Oracle CODASYL DBMS™

Database Administration Reference Manual

Release 7.4.1

August 2022
Build: August 9, 2022

ORACLE®

Oracle CODASYL DBMS Database Administration Reference Manual

Oracle Corporation - Worldwide Headquarters, 2300 Oracle Way, Austin, TX 78741, United States

Primary Author: DBMS Engineering and Documentation group

Contents

7 Supported Data Types

8 Conditional Expressions

9 DBO Utility Commands

10 Load/Unload Utility Commands

11 DBALTER Utility Commands

12 DRU Commands

A Configuration File Management and User Defined Events

Index

Tables

Send Us Your Comments

Oracle CODASYL DBMS
Database Administration Reference Manual, 7.4.1

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title, release date, chapter, section, and page number (if available).

Please direct all comments, and corrections to this email address: **infordb_us@oracle.com**.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

Oracle CODASYL DBMS is a CODASYL-compliant general purpose database management system (DBMS) based on the March 1981 Working Document of the ANSI Data Definition Language Committee.

This manual provides reference information about the following:

- The data definition language (DDL) and its compiler
- The load/unload format and sequence language
- The OpenVMS data types supported by Oracle CODASYL DBMS
- The conditional expression
- The command language of the Database Operator (DBO) utility
- The command language of the DBALTER file patching utility
- The command language of the Database Restructuring Utility (DRU)
- The logical names used by Oracle CODASYL DBMS

Intended Audience

This manual is intended for a database administrator or person performing that function, either for a multi-user database, or for a single-user application.

Before reading this manual, you should read the *Introduction to Oracle CODASYL DBMS*, *Oracle CODASYL DBMS Database Design Guide*, *Oracle CODASYL DBMS Database Load/Unload Guide*, *Oracle CODASYL DBMS Database Maintenance and Performance Guide*, and the *Oracle CODASYL DBMS Database Security Guide*.

Familiarity with the OpenVMS operating system is assumed. Knowledge of a high-level programming language is helpful.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Document Structure

This manual contains 12 chapters and an index:

Chapter 1	Provides an introduction to information in the manual about the database administrative utilities.
Chapter 2	Describes in detail the syntax and semantics of the schema data definition.
Chapter 3	Describes in detail the syntax and semantics of the storage schema data definition.
Chapter 4	Describes in detail the syntax and semantics of the subschema data definition.
Chapter 5	Describes in detail the syntax and semantics of the security schema data definition.
Chapter 6	Describes the syntax and semantics of the DDL compiler. The DDL compiler uses the OpenVMS command language (DCL).
Chapter 7	Describes the OpenVMS data types.
Chapter 8	Describes the conditional text that can be used in several of the definitions.
Chapter 9	Describes in detail the syntax and semantics of the DBO utility commands.
Chapter 10	Describes in detail the syntax and semantics of the load/unload format language and the load/unload sequence language.
Chapter 11	Describes the DBALTER syntax and commands.
Chapter 12	Describes the DRU syntax and commands.

Related Documents

The other manuals in the Oracle CODASYL DBMS documentation set are:

- *Introduction to Oracle CODASYL DBMS*
- *Oracle CODASYL DBMS Installation Guide*
- *Oracle CODASYL DBMS Database Load/Unload Guide*
- *Oracle CODASYL DBMS Database Design Guide*
- *Oracle CODASYL DBMS Database Maintenance and Performance Guide*
- *Oracle CODASYL DBMS Database Security Guide*
- *Oracle CODASYL DBMS Programming Guide*
- *Oracle CODASYL DBMS Programming Reference Manual*
- *Oracle CODASYL DBMS Release Notes*

Associated Documents

The other manuals referred to in this manual are:

- Documentation for VSI DATATRIEVE
- OpenVMS documentation for Language-Sensitive Editor and Source Code Analyzer

Conventions

The following conventions are used in this manual:

Symbol	Meaning
WORD	An uppercase word in a syntax format is a keyword. You must include it in the statement if the clause is used.
WORD	An underlined uppercase word means that you must include the word in the clause, statement, or command.
word	A lowercase word in a syntax format indicates a syntax element that you supply.
[ ]	Brackets enclose optional clauses from which you can choose one or none.
{ }	Braces enclose clauses from which you must choose one alternative.
CTRL/x	This symbol tells you to press the CTRL (control) key and hold it down while pressing a letter key.
. . .	Horizontal ellipsis points mean you can repeat the previous item.
. . .	Vertical ellipsis points in an example mean information not directly related to the example has been omitted.

References to Products

The Oracle CODASYL DBMS documentation set to which this manual belongs often refers to products by their abbreviated names:

- VSI DATATRIEVE software is referred to as DATATRIEVE.
- VMS Language-Sensitive Editor software is referred to as LSE.

1
Introduction

This book is intended for use by the database administrator (DBA) or person performing that function. The DBA function is to design, analyze, modify, and maintain the logical database and the database management system. The DBA also provides for the integrity and security of the data in the database. The information contained in this book is reference material describing the commands and syntax needed to perform the DBA function.

You can find tutorial information on designing a database in the *Oracle CODASYL DBMS Database Design Guide*. Once you have designed the database, you use the Oracle CODASYL DBMS data definition language (DDL) to define and compile the following four elements:

- Schema
- Subschema
- Security schema
- Storage schema

The schema describes the logical elements of the database (data items, record types, and set types) and their relationships to each other. The storage schema describes the physical structure of the database, how record types are stored, and how set types are represented. The subschema describes a subset of the schema allowing application programs to access only the portion of the database needed. The security schema describes a subset of the schema. It is used to control database access by permitting and denying users access. You can find tutorial information on general database security in the *Oracle CODASYL DBMS Database Security Guide*.

Four chapters describe the syntax for defining the schema, subschema, security schema, and storage schema. Chapter 6 describes the DDL compiler syntax for compiling the schema definitions.

You specify a data type for a schema, subschema, and storage schema record. A data type identifies the size of the operand and the significance of the bits in the operand. Chapter 7 describes each of the available data types in detail.

You can specify conditional expressions in the schema. Conditional expressions are Boolean expressions that consist of a simple relational expression or combinations of expressions. Conditional expressions are evaluated at run time, and the control paths depend on whether the condition is true or false. Chapter 8 describes the use, structure, and evaluation of conditional expressions.

To manage your database, Oracle CODASYL DBMS provides the Database Operator (DBO) utility. Some DBO commands control operation of the database, while others create, maintain, and delete databases. Chapter 9 describes these commands and their parameters and qualifiers. You can find tutorial information

on maintenance and performance for a database in the *Oracle CODASYL DBMS Database Maintenance and Performance Guide*.

The DBO commands, DBO/LOAD and DBO/UNLOAD, are used to load, unload, and reload data into a database. This operation requires that you create FORMAT and SEQUENCE files. The FORMAT file describes the data to be exchanged between database records and OpenVMS Record Management Service (RMS) files. The SEQUENCE file determines the order of the exchange. Chapter 10 describes the syntax for these files. See the *Oracle CODASYL DBMS Database Load/Unload Guide* for examples of how to use the Load/Unload utility.

The DBO/ALTER command invokes the interactive DBALTER utility. DBALTER provides a low-level patch capability for Oracle CODASYL DBMS databases and allows you to move a database from one device to another. Chapter 11 describes the full syntax and usage of the DBALTER environment. See the *Oracle CODASYL DBMS Database Maintenance and Performance Guide* for a detailed tutorial introduction to the DBALTER utility.

The DBO/RESTRUCTURE command invokes the interactive Database Restructure Utility (DRU). DRU allows you to change certain database characteristics without unloading and reloading the database. Chapter 12 describes the full syntax and usage of the DRU environment level. For tutorial information on using DRU, see the *Oracle CODASYL DBMS Database Maintenance and Performance Guide*.

You should use only Oracle CODASYL DBMS utilities to manipulate any Oracle CODASYL DBMS files. If you use any other utilities, you might corrupt your database. If you do corrupt your database, you should restore the database from your latest backup file and roll forward from your .AIJ file or files.

2
Schema Definition

Use the schema definition of the Oracle CODASYL DBMS data definition language (DDL) to define the logical structure of a database. The schema describes all logical elements of the database and their relationships to each other.

This chapter is divided into four sections. Each section describes one entry of the schema definition:

- SCHEMA entry

  Provides the syntax for naming a database schema
- Schema AREA entry

  Provides the syntax for naming one or more database areas
- Schema RECORD entry

  Provides the syntax for defining schema record types
- Schema SET entry

  Provides the syntax for defining schema set types

The following syntax presents the major schema divisions. The area, record, and set clauses must be repeated for each area, record type, and set type defined in the schema:

- SCHEMA NAME IS schema-name
- {AREA NAME IS area-name} . . .
- {RECORD NAME IS record-name} . . .
- [SET NAME IS set-name] . . .

A schema must include at least the schema identification, one area definition, and one record definition.

If you have DEC Language-Sensitive Editor (LSE) installed on your system, you can use LSE templates to write your schema. There is an LSE template available for the data definition language (DDL). To invoke LSE, enter the LSEDIT command followed by the name of the file you want to edit:

```
LSEDIT filename
```

If you use the .DDL file type the template is automatically invoked for the editing session.

Once you are using the template, a placeholder appears on the first line of the editing buffer. A placeholder is text that can be expanded into code. In the DDL template, the placeholder is the following:

```
[database_schema]
```

Expand this placeholder to write a schema.

See the VMS Language-Sensitive Editor and Source Code Analyzer User Manual and *Introduction to Oracle CODASYL DBMS* to learn more about using LSE.

2.1 SCHEMA Entry

The SCHEMA entry assigns a name to the logical representation of the database and must be the first entry in the schema definition file.

Format

SCHEMA NAME IS schema-name

Arguments

SCHEMA NAME IS schema-name
Names the logical representation of the database. There is a 31-character limit on schema name.

Example

```
*  AUTHOR:    NILS JACOBSCHUTZ     -+
*  DATE:      19-APR-1981           |----❶
*  FACILITY:  TROY, NEW YORK        |
*  REMARKS:   PARTS DATABASE       -+

SCHEMA NAME IS PARTS --------------------❷
```

❶ These four lines are comments. They identify the author, date, and facility and provide some further information about the schema. Comments are not required by the syntax but provide useful documentation for future database administrator's maintenance of the database.

❷ This line identifies the schema as PARTS.

2.2 Schema AREA Entry

The schema AREA entry names a database area. A database is composed of one or more areas. It is good practice to place the AREA entry after the SCHEMA entry.

Format

AREA NAME IS area-name

Arguments

AREA NAME IS area-name
Names the database area. There is a 31-character limit on area name.

Example

```
AREA NAME IS MAKE
AREA NAME IS BUY
AREA NAME IS MARKET
AREA NAME IS PERSONNEL
```

These four lines identify the four areas of the database as MAKE, BUY, MARKET, and PERSONNEL.

2.3 Schema RECORD Entry

The schema RECORD entry names and describes a record type. It includes information about areas that can contain occurrences of the record type. It is good practice to place the RECORD entries after the AREA entry.

Format

```
RECORD NAME IS record-name
[ WITHIN { area-name... } ]
         { ANY AREA     }

[ UNIQUE data-item... ]...

[ CHECK IS condition-test ]

[ ITEM NAME IS data-item                  ]
[ [ OCCURS unsigned-integer TIMES ]       ]
[ [ CHECK IS condition-test ]             ]...
[ TYPE IS data-type                       ]
[ [ DEFAULT VALUE IS literal-value ]      ]
```

Defaults

WITHIN ANY AREA

If you do not use the optional WITHIN clause, record types can be stored in ANY AREA.

Arguments

RECORD NAME IS record-name

Names a record type in the schema. There is a 31-character limit on record name. Because a 6-character prefix is added to each record name at compile time, the practical limit is 25 characters.

There must be a record entry for each record type in the schema.

WITHIN

If the schema specifies more than one area by name or the run unit does not specify an area when it stores a record occurrence, the Database Control System (DBCS) chooses the area. Otherwise, the record occurrence is stored in the area specified by the run unit.

WITHIN area-name
DBCS stores occurrences of this record type in the specified area.

WITHIN ANY AREA
If the run unit does not specify a valid area, DBCS chooses the area from all valid areas at run time and stores the record in the chosen area. This is the default.

UNIQUE data-item
Specifies one or more data items that must be unique among the record occurrences.

Note

Overuse of the UNIQUE clause slows performance. Every time you store or modify a record, the value must be validated against each UNIQUE clause.

The following example illustrates the use of the UNIQUE clause. Adding a UNIQUE clause to the EMP_ID record type ensures that each employee has a unique identification number:

```
RECORD NAME IS EMPLOYEE
    WITHIN PERSONNEL
*   Owner of sets :  MANAGES
*                    RESPONSIBLE_FOR
*   Member of sets:  ALL_EMPLOYEES
*                    CONSISTS_OF
        UNIQUE EMP_ID
    ITEM EMP_ID TYPE IS UNSIGNED NUMERIC 5
    ITEM EMP_LAST_NAME TYPE IS CHARACTER 20
    ITEM EMP_FIRST_NAME TYPE IS CHARACTER 10
    ITEM EMP_PHONE TYPE IS UNSIGNED NUMERIC 7
        DEFAULT IS 5559191
    ITEM EMP_LOC TYPE IS CHARACTER 5
```

CHECK IS condition-test
Checks the validity of a record occurrence against a Boolean expression. DBCS evaluates this expression at run time when a record occurrence is stored or modified. DBCS does not allow records to be stored or modified if the result of the condition test is false.

See Chapter 8 for more information on condition tests.

ITEM NAME IS data-item
Names a data item in a record. This clause must be repeated for each data item type.

OCCURS unsigned-integer TIMES
Identifies a data item type as a repeating item. Specify the number of times it occurs.

The integer value (greater than 0) supplied is the same for each occurrence.

CHECK IS condition-test
Checks the validity of an item against a Boolean expression. DBCS evaluates this expression at run time when a record occurrence is stored or the item is modified. DBCS does not allow record storage or modification operations if the result of the condition test is false.

See Chapter 8 for more information on condition tests.

TYPE IS data-type

Specifies the data type for the data item type.

See Chapter 7 for information about the data types and their uses.

DEFAULT VALUE IS literal-value

Provides a default value for a data item type. The default value must be the same class as the data item type (either string or any of the numeric data types). A valid move from the default value to the data item type must be possible.

The default value is used when a record occurrence is stored if a subschema does not have access to all data item types for a record type. If there is a default value in such a case, DBCS supplies that value. Otherwise, an error occurs.

Nonnumeric character string literals must be enclosed within quotation marks (" ").

Examples

1.
```
RECORD NAME IS PART ----------------------------❶
  WITHIN  MAKE    |-------------------------------❷
          BUY     |
  CHECK IS (PART_PRICE GT ".000" AND   |---------❸
           PART_COST GT ".000")        |
    ITEM PART_ID       TYPE IS CHARACTER 8   |
    ITEM PART_DESC     TYPE IS CHARACTER 50  |---❹
    ITEM PART_STATUS   TYPE IS CHARACTER 1   |
          DEFAULT VALUE IS "G" ------------------❺
          CHECK IS (PART_STATUS EQ "N" OR    -+
                 PART_STATUS EQ "R" OR        |--❻
                 PART_STATUS EQ "G")         -+
```

This example uses the WITHIN clause, the record CHECK clause, and a data item type default value:

❶ Names the PART record type.

❷ Specifies that PART records will be stored in the MAKE and BUY areas.

❸ Causes DBCS to check the validity of a record occurrence before storing or modifying it. It checks that the values of the PART_PRICE and PART_COST items are greater than 0.

See Chapter 8 for more information on using condition tests.

❹ Names the data item types that make up a record type. Each ITEM clause also specifies the data type for that data item type. For example, PART_ID contains 8 alphanumeric characters.

❺ Specifies the default value to be stored if the subschema for a run unit does not have access to that data item type. If a run unit does not have access to PART_STATUS, DBCS stores the value G.

❻ Causes DBCS to check the validity of the data item occurrence before storing a record occurrence. It checks that the value of PART_STATUS is N, R, or G.

See Chapter 8 for more information on using the CHECK clause.

2.
```
RECORD NAME IS VENDOR ------------------------❶
  WITHIN MARKET ------------------------------❷
    ITEM VEND_ID       TYPE IS CHARACTER 8                  -+
    ITEM VEND_NAME     TYPE IS CHARACTER 40                  |
    ITEM VEND_CONTACT  TYPE IS CHARACTER 30                  |--❸
    ITEM VEND_ADDRESS  TYPE IS CHARACTER 15                  |
                            OCCURS 3 TIMES --------❹         |
    ITEM VEND_PHONE    TYPE IS UNSIGNED NUMERIC 10          -+
```

This example uses the WITHIN clause and the OCCURS clauses:

❶ Names the VENDOR record type.

❷ Limits the VENDOR record type to the area called MARKET.

❸ Names the data item types that make up the record type. Each ITEM clause also specifies the data type for that data item type. For example, VEND_PHONE contains UNSIGNED NUMERIC data with a length of 10.

❹ Specifies the number of times VEND_ADDRESS can occur. The OCCURS clause allows exactly three lines of address information for each vendor.

3.
```
RECORD NAME IS COMPONENT ----------------------❶
  WITHIN MAKE ----------------------------------❷
    ITEM COMP_SUB_PART    TYPE IS CHARACTER 8            -+
    ITEM COMP_OWNER_PART  TYPE IS CHARACTER 8             |
    ITEM COMP_MEASURE     TYPE IS CHARACTER 1             |------❸
      CHECK IS (COMP_MEASURE EQ "L" OR  -+                |
                COMP_MEASURE EQ "G" OR   |-----❹          |
                COMP_MEASURE EQ "M" OR   |                |
                COMP_MEASURE EQ "U")    -+                |
    ITEM COMP_QUANTITY    TYPE IS UNSIGNED NUMERIC 5 -2  -+
```

This example uses the WITHIN clause and the CHECK clause on an item:

❶ Names the COMPONENT record type.

❷ Limits the COMPONENT record type to the area MAKE.

❸ Names the data item types that make up the record type. Each ITEM clause also specifies the data type for that data item type. For example, COMP_SUB_PART contains 8 alphanumeric characters.

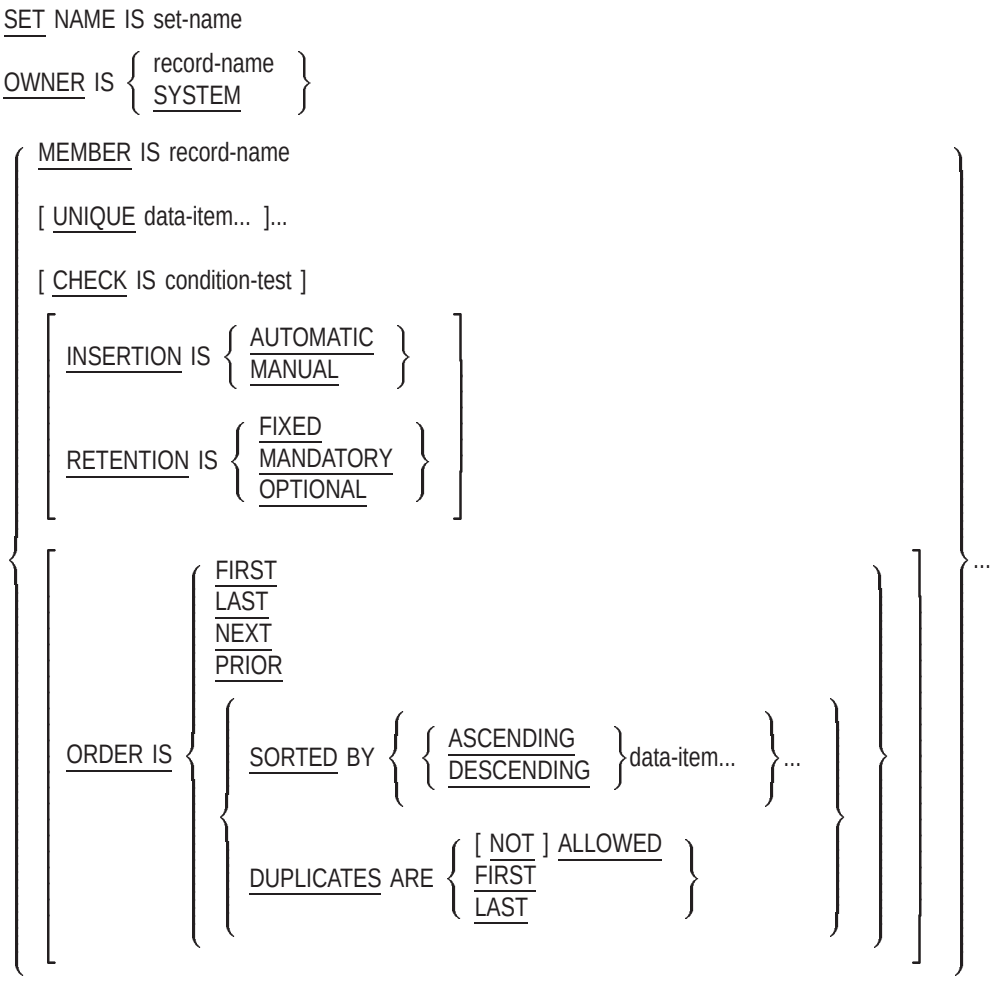
❹ Causes DBCS to check the validity of the data item occurrence before storing or modifying it. It checks that the values of COMP_MEASURE are L, G, or M.

See Chapter 8 for more information on using the CHECK clause.

2.4 Schema SET Entry

This schema SET entry names and describes a set type. It defines ownership, membership, how record occurrences become members of set occurrences, and under what conditions record occurrences can be removed from set occurrences. It is good practice to place SET entries after the RECORD entries.

Format



Set types, with the exception of SORTED set types, can have one or more members. Repeat the INSERTION, RETENTION, UNIQUE, and ORDER clauses for each member record type.

Defaults

INSERTION IS MANUAL
RETENTION IS OPTIONAL
If you specify either an INSERTION or a RETENTION option, but not both, you get an error. If you omit both INSERTION and RETENTION, the DDL compiler supplies MANUAL OPTIONAL for that member.

ORDER IS
If you do not choose an ORDER option, the order is indeterminate. At any time DBCS can change the ordering criteria for members in sets where the ORDER option has not been specified.

Arguments

SET NAME IS set-name

Names a set type in the schema. There is a 31-character limit on set name. Because a 6-character prefix is added to each set name at compile time, the practical limit is 25 characters.

There must be a SET entry for each set type in the schema.

OWNER IS record-name

Identifies the specified record type as the owner of the set type. Each occurrence of this record type establishes an occurrence of the set type. The set occurrence established does not need members to exist in the database. Set occurrences can be empty.

OWNER IS SYSTEM

Names SYSTEM as the owner of the set type. Because there is only one system, there is only one occurrence of each SYSTEM-owned set type in the database.

MEMBER IS record-name

Identifies a record type as a member of a set type.

Each set type must have one or more members. Each member record type should have its own member clause. The member record type cannot also be the owner record type of the set type.

UNIQUE data-item

Specifies one or more data items that must be unique among the member occurrences within each set occurrence. You can use the UNIQUE clause to disallow duplicate member records with respect to the values of the fields defined in the UNIQUE clause. You should use the UNIQUE clause on CALC keys to improve performance. This allows DBCS to use efficient methods of validating a newly stored, modified, or connected record occurrence.

A schema MEMBER clause should not contain conflicting UNIQUE and SORTED . . . DUPLICATES clauses. For example, the following member clause is contradictory because of the DUPLICATES ARE LAST clause:

```
MEMBER IS CUSTOMER_RECORD
    INSERTION IS AUTOMATIC
        RETENTION IS FIXED
    UNIQUE CUSTOMER_ID
    ORDER IS
        SORTED BY ASCENDING CUSTOMER_ID
        DUPLICATES ARE LAST
```

Contradictory ORDER SORTED and UNIQUE clauses do not cause compilation errors, but cause errors if you try to store records with duplicate keys. The order of precedence for the clauses in determining set membership is:

1. ORDER IS SORTED
2. MEMBER . . . UNIQUE
3. RECORD . . . UNIQUE

If an attempted record storage operation results in an error on the ORDER SORTED clause, the UNIQUE tests will never occur. In the previous example, a member occurrence with a duplicate CUSTOMER_ID would result in a UNIQUE clause error. A record instance must have a unique CUSTOMER_ID to be included in the set, even though duplicate CUSTOMER_NAME values are allowed.

Note

Overuse of the UNIQUE clause slows performance. Every time you store or modify a record, the value must be validated against each UNIQUE clause.

You should use the MEMBER UNIQUE clause in conjunction with CALC keys. You can then use performance optimizations to validate the uniqueness of a newly stored, modified, or inserted record occurrence in the set.

CHECK IS condition-test

Checks the validity of a member occurrence against a Boolean expression. DBCS evaluates this expression at run time when a member record occurrence is connected into a set occurrence (either by STORE, CONNECT, or RECONNECT) and when the owner or member is modified. DBCS does not allow records to be stored or modified if the result of the condition test is false. See Chapter 8 for more information on using condition tests.

INSERTION IS AUTOMATIC

Automatically inserts a member record occurrence into a set occurrence when the member record occurrence is stored.

INSERTION IS MANUAL

Requires the DML CONNECT statement to insert a member record occurrence into a set occurrence.

RETENTION IS FIXED

Once the record occurrence is connected to an occurrence of a set, it remains a member of that set occurrence until it is erased from the database. Its position in that set occurrence can change.

RETENTION IS MANDATORY

Once the record occurrence is connected to an occurrence of this set type, it must remain a member of some occurrence of this set type until it is deleted from the database by using the ERASE command. It can be moved to another occurrence of the same set type with the DML RECONNECT statement.

RETENTION IS OPTIONAL

An occurrence of this record type can exist in the database without being a member of any set occurrence. The record occurrence can be removed, reconnected, reordered, or connected to occurrences of any set type at any time.

ORDER IS FIRST

New member record occurrences are inserted before all other member record occurrences.

ORDER IS LAST
New member record occurrences are inserted after all other member record occurrences.

ORDER IS NEXT
New member record occurrences are inserted immediately after the current position in the set occurrence. The DML program selects this position.

ORDER IS PRIOR
New member record occurrences are inserted immediately before the current position in the set occurrence. The DML program selects this position.

SORTED BY ASCENDING data-item
SORTED BY DESCENDING data-item
Names one or more data item types (separated by a space, tab, or line break) in the member record type as a sort key. The sort key can be up to 255 bytes long. DBCS sorts new member record occurrences using occurrences of the data item types that make up the sort key.

New member record occurrences can be sorted in increasing (ASCENDING) or decreasing (DESCENDING) order.

During a retrieval operation (FETCH or FIND), if the first data item occurrence in the sort key for two different record occurrences is the same, DBCS uses the occurrence of the next data item type in the sort key to attempt to distinguish between the two record occurrences. This process continues for each data item type in the sort key. If there are no more data item types in the sort key and the duplicate record occurrences have not yet been distinguished, DBCS uses the information provided by the DUPLICATES clause to determine record placement.

DUPLICATES ARE ALLOWED
DUPLICATES ARE FIRST
DUPLICATES ARE LAST
Determines the validity of having two or more record occurrences with the same sort key. You can allow or disallow duplicate member record occurrences (ALLOWED or NOT ALLOWED), allow duplicates and order them sorted before or after all other duplicates (FIRST or LAST), or allow duplicates without specifying how they should be sorted (ALLOWED).

A set type with SORTED order can have only one member record type. Multimember or forked, SORTED set types result in a DDL compiler error.

Examples

1.
```
SET NAME IS ALL_CLASS -------❶
  OWNER IS SYSTEM -----------❷
  MEMBER IS CLASS -----------❸
    INSERTION IS AUTOMATIC --❹
    RETENTION IS FIXED ------❺
```

This example is a SYSTEM-owned set type:

❶ Names the ALL_CLASS set type.

❷ Identifies the owner as SYSTEM. A SYSTEM-owned set type has only one occurrence in the database.

❸ Identifies the CLASS record type as a member of the ALL_CLASS set type.

❹ Specifies that CLASS record occurrences are automatically inserted into the set occurrence upon storage.

❺ Specifies that CLASS record occurrences must remain a member of the set occurrence in which they were originally stored. Because there can be no other occurrences of a SYSTEM-owned set type, the choice of RETENTION is required.

2.
```
SET NAME IS PART_INFO ----------❶
  OWNER IS PART ----------------❷
  MEMBER IS SUPPLY -------------❸
    INSERTION IS AUTOMATIC  |----❹
    RETENTION IS FIXED      |
      ORDER IS NEXT ----------❺
  MEMBER IS QUOTE --------------❻
    INSERTION IS AUTOMATIC  |----❼
    RETENTION IS FIXED      |
      ORDER IS NEXT ----------❽
```

This example describes a set type with more than one member:

❶ Names the PART_INFO set type.

❷ Identifies the owner as PART. Each occurrence of PART in the database defines an occurrence of this set type.

❸ Identifies the SUPPLY record type as a member of the PART_INFO set type.

❹ Specifies the INSERTION and RETENTION for SUPPLY record occurrences. SUPPLY record occurrences are automatically connected to an occurrence of PART_INFO when stored and remain a member until deleted from the database by using the ERASE command.

❺ Specifies that new occurrences of SUPPLY are to be stored in the NEXT position in each occurrence of type, PART_INFO.

❻ Identifies the QUOTE record as a member of the PART_INFO set type.

❼ Specifies the INSERTION and RETENTION for QUOTE record occurrences. SUPPLY record occurrences are automatically connected to an occurrence of PART_INFO when stored and remain a member until deleted from the database by using the ERASE command.

❽ Specifies that new occurrences of QUOTE are to be stored in the NEXT position in each occurrence of type, PART_INFO.

3.
```
SET NAME IS CONSISTS_OF --------❶
  OWNER IS DIVISION ------------❷
  MEMBER IS EMPLOYEE -----------❸
    INSERTION IS MANUAL --------❹
    RETENTION IS OPTIONAL ------❺
    UNIQUE EMP_ID --------------❻
  ORDER IS SORTED BY
      ASCENDING EMP_LAST_NAME EMP_FIRST_NAME ----❼
      DUPLICATES ARE LAST
```

This example shows a set type that is SORTED by the values of two data item types:

❶ Names the CONSISTS_OF set type.

❷ Identifies DIVISION as the owner of the set type. Each occurrence of DIVISION in the database defines an occurrence of this set type.

❸ Identifies the EMPLOYEE record type as a member of the CONSISTS_OF set type.

❹ Specifies that EMPLOYEE record occurrences must be manually connected to occurrences of the CONSISTS_OF set type.

❺ Specifies that EMPLOYEE record occurrences can exist in the database without belonging to this set.

❻ Specifies that each EMPLOYEE within an occurrence of the CONSISTS_OF set must have a unique EMP_ID.

❼ Specifies that the two data item types, EMP_LAST_NAME and EMP_FIRST_NAME, are used by DBCS to produce the key upon which EMPLOYEE record occurrences are sorted. If DBCS encounters any new record occurrences with the same sort key value as an existing record occurrence, it adds the new record occurrence after all other DUPLICATES.

4.
```
SET NAME IS PART_USED_ON --------❶
  OWNER IS PART -----------------❷
  MEMBER IS COMPONENT -----------❸
    CHECK IS PART_ID                        |-----❹
    IN PART EQ COMP_SUB_PART IN COMPONENT   |
    INSERTION IS AUTOMATIC ------❺
    RETENTION IS FIXED ----------❻
    ORDER IS NEXT ---------------❼
```

This example demonstrates the use of the CHECK clause on a member of a set type:

❶ Names the PART_USED_ON set type.

❷ Identifies PART as the owner of the set type. Each occurrence of PART in the database defines an occurrence of this set type.

❸ Identifies the COMPONENT record type as a member of the PART_USED_ON set type.

❹ DBCS checks that the values of PART_ID and COMP_SUB_PART are the same before storing or modifying the record.

See Chapter 8 for more information on using the CHECK clause.

❺ Specifies that COMPONENT record occurrences are automatically added to the PART_USED_ON set type upon storage.

❻ Specifies that COMPONENT record occurrences must remain a member of the set occurrence in which they were originally stored.

❼ Specifies that new occurrences of COMPONENT are to be stored in the NEXT position in each occurrence of the PART_USED_ON set type.

3
Storage Schema Definition

Use the storage schema definition of the Oracle CODASYL DBMS data definition language (DDL) to define the physical structure of the database. The storage schema describes how record types are stored and how set types are represented.

This chapter is divided into three sections. Each section describes an entry of the storage schema definition:

- STORAGE SCHEMA entry

  Provides the syntax for naming a storage schema and associating it with a schema

- Storage schema RECORD entry

  Provides the syntax for describing how occurrences of record types are stored

- Storage schema SET entry

  Provides the syntax for describing how occurrences of set types are represented in the database

The following syntax presents the major storage schema divisions. The storage record and storage set entries can be repeated in the storage schema for each schema record and set type. If you do not provide an entry for a schema record or set type, the DDL compiler includes the record or set type information in the storage schema using default values:

- STORAGE SCHEMA NAME IS storage-schema-name
- [RECORD NAME IS record-name]...
- [SET NAME IS set-name]...

If you have DEC Language-Sensitive Editor (LSE) installed on your system, you can use LSE templates to write your storage schema. There is an LSE template available for the data definition language (DDL). To invoke LSE, enter the LSEDIT command followed by the name of the file you want to edit:

```
LSEDIT filename
```

If you use the .DDL file type, the template is automatically invoked for the editing session.

Once you are using the template, a placeholder appears on the first line of the editing buffer. A placeholder is text that can be expanded into code. In the DDL template, the placeholder is the following:

```
[database_schema]
```

Expand this placeholder to write a storage schema.

See the VMS Language-Sensitive Editor and Source Code Analyzer User Manual and *Introduction to Oracle CODASYL DBMS* to learn more about using LSE.

3.1 STORAGE SCHEMA Entry

The STORAGE SCHEMA entry names the storage schema for a database and associates it with the schema defined for the database.

Format

STORAGE SCHEMA NAME IS storage-schema-name

FOR schema-name SCHEMA

Arguments

STORAGE SCHEMA NAME IS storage-schema-name
Names the storage schema. There is a 31-character limit on schema name.

FOR schema-name SCHEMA
Identifies the schema for which the storage schema is being defined.

Examples

1. `STORAGE SCHEMA NAME IS PARTST1 FOR PARTS SCHEMA`

   This line is an example of a user-written storage schema entry. It names the storage schema (PARTST1) and identifies the schema for which it was written (PARTS).

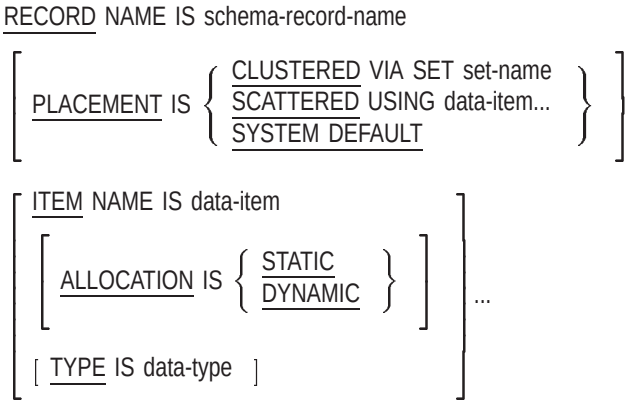
2. `STORAGE SCHEMA NAME IS DEFAULT_STORAGE_SCHEMA FOR PARTS SCHEMA`

   This line is an example of the storage schema entry produced by the default storage schema generator for the PARTS schema. The default storage schema generator names all storage schemas DEFAULT_STORAGE_SCHEMA and associates them with the appropriate schema. In this case, the storage schema is for the PARTS schema.

3.2 Storage Schema RECORD Entry

The storage schema RECORD entry defines the physical storage characteristics of a schema record type. If you do not include a schema record type in the storage schema, the compiler includes the record type using default values.

Format



Defaults

PLACEMENT IS

If you do not specify PLACEMENT for a record type, the DDL compiler chooses a PLACEMENT option on the basis of schema and storage schema set type characteristics.

ALLOCATION IS STATIC

If you do not choose an ALLOCATION option, the DDL compiler provides the STATIC option.

TYPE IS data-type

If you do not specify a storage schema data type for anything other than a decimal string data type (see Chapter 7), the DDL compiler provides the data type used in the schema for that data item. For strings defined as a decimal type in the schema, the DDL compiler uses PACKED DECIMAL in the storage schema.

Arguments

RECORD NAME IS schema-record-name

Identifies the schema record type for which the storage characteristics are being defined. There is a 31-character limit on record name. Because a 6-character prefix is added to each record name at compile time, the practical limit is 25 characters.

PLACEMENT IS

Determines the database page on which to store occurrences of this record type. Storage positions are clustered around a set owner occurrence or calculated by DBCS.

CLUSTERED VIA SET set-name

DBCS tries to store member record occurrences on the same page as the owner occurrence for the specified set type. The member record type must be an AUTOMATIC member of the specified set type. If no room is left on the page chosen by DBCS, the member record occurrence is stored on a page as close as possible to the storage location of the set owner record occurrence.

This option clusters owners and members even if they are in different areas. This method results in fast access when searching through occurrences of a given set type.

SCATTERED USING data-item

DBCS computes a storage page from the occurrence of one or more data item types in the record type. These data item types are supplied by the data item clause. If no room is left on the computed page, the record occurrence is stored as close to that page as possible.

As long as the value for each data item is unique, this method evenly distributes record occurrences of a given type throughout all pages of the storage areas in which the record type can be stored. The SCATTERED USING option is useful for looking up individual records in CALC sets. It does not, however, provide for fast access paths to such record occurrences.

SYSTEM DEFAULT

The SYSTEM DEFAULT entry lets DBCS decide where to place the record. Letting DBCS decide where to place the record will hamper performance. Oracle recommends that you select what is applicable for your application.

ITEM NAME IS data-item

Specifies a schema data item type. This option can be used to specify a data item type that is different from the data item type specified in the schema. See Chapter 7 for information about supported data types and their uses.

ALLOCATION IS STATIC

All occurrences of this data item occupy the same amount of space. DBCS does not compress STATIC data item occurrence values.

ALLOCATION IS DYNAMIC

The data item occurrence value is compressed when stored. If its value is the same as the schema default value for the item occurrence, it occupies no space. Otherwise, each run of 3 or more identical bytes is represented as 2 bytes in storage.

TYPE IS data-type

Specifies the data type. See Chapter 7 for information about the data types and their uses.

Examples

1.
```
RECORD NAME IS PART ---------------------------❶
  PLACEMENT IS CLUSTERED VIA ALL_PARTS --------❷
    ITEM PART_ID TYPE IS CHARACTER 8              -+
    ITEM PART_DESC TYPE IS CHARACTER 50            |
    ITEM PART_STATUS TYPE IS CHARACTER 1           |
    ITEM PART_PRICE TYPE IS PACKED DECIMAL 9 -3    |--❸
    ITEM PART_COST TYPE IS PACKED DECIMAL 9 -3     |
    ITEM PART_SUPPORT TYPE IS CHARACTER 2         -+
```

These examples show the two PLACEMENT options:

❶ Identifies PART as the schema record type for which the storage characteristics are being defined.

❷ Specifies that member occurrences of the PART record type are to be stored as close as possible to the owner occurrence of the set type ALL_PARTS. Because the mode of ALL_PARTS is CALC (see Section 3.3), retrieving a PART record occurrence by the CALC key usually takes only one disk access.

❸ Specifies data item types for purposes of internal documentation or to override the schema assignment for those data item types.

2.
```
RECORD NAME IS DIVISION ----------------------❶
  PLACEMENT IS SCATTERED USING DIV_NAME -----❷
  ITEM DIV_NAME TYPE IS CHARACTER 20 --------❸
```

❶ Identifies DIVISION as the schema record type for which the storage characteristics are being defined.

❷ Specifies that occurrences of the DIVISION record type are stored by DBCS on the basis of the occurrence value of the data item type, DIV_NAME. This procedure tends to distribute DIVISION record occurrences throughout the database.

❸ Specifies a data item type (DIV_NAME) for purposes of internal documentation or to override the schema assignment for that data item type.

3.3 Storage Schema SET Entry

The storage schema SET entry describes how a schema set type is represented in the database, giving the access path to a record type in the set type. If you do not include an entry for a given schema set type in the storage schema definition, the DDL compiler includes the set type by default.

Format

```
SET NAME IS set-name

[ MODE IS { CALC {MEMBER IS record-name KEY IS data-item...}... }
          { CHAIN                                               }
          { INDEX [ NODE SIZE IS unsigned-integer BYTES ]       }

  [ OMIT PRIOR ] ]
```

Default

SET NAME IS set-name

If you provide a SET entry for a set type defined in the schema, the DDL compiler chooses a mode based on schema characteristics of the set type. There is a 31-character limit on set name. Because a 6-character prefix is added to each set name at compile time, the practical limit is 25 characters. The DDL compiler assigns a default mode by the following rules:

1. SORTED set types must have INDEX or CHAIN mode. Specifying CHAIN mode for a SORTED set results in a nonindexed sorted set that offers

performance advantages when there are a small number of set occurrences (generally, less than 100 set occurrences). Specifying INDEX mode results in a B-tree structure. INDEX mode is the default for sorted sets.

2. Set types with ORDER of FIRST, LAST, NEXT, and PRIOR must be CHAIN mode.
3. Set types owned by a record type for which no ORDER was specified in the schema must be CHAIN mode.
4. SYSTEM-owned set types for which no ORDER was specified in the schema are given a mode of CALC. You can also use CHAIN mode for SYSTEM-owned sets, but CHAIN mode is not provided by default.

Arguments

SET NAME IS set-name
Identifies the associated schema set type. There is a 31-character limit on set name. Because a 6-character prefix is added to each set name at compile time, the practical limit is 25 characters.

MODE IS
Describes how set types are represented in the database. The MODE can be CALC, CHAIN, or INDEX. The mode you specify depends on some of the characteristics of the associated schema set type.

CALC MEMBER IS record-name KEY IS data-item
DBCS uses the occurrence values of one or more data item types, specified by data-item, to find the required record occurrence. Retrieval of members is fast if DBCS knows all the KEY values. Otherwise, DBCS searches the area allowed only for SYSTEM-owned set types with no ORDER specified.

CHAIN
The owner record occurrence is at the beginning of a chain, followed by linked member record occurrences. Each member has pointers to the next and prior record occurrences in a chain, as well as back to the owner. You can specify CHAIN for either sorted or unsorted sets.

If ORDER IS SORTED is specified in the schema and if MODE IS CHAIN is specified in the storage schema, the set will have a nonindexed sorted chain structure.

Using a nonindexed sorted set eliminates the overhead of maintaining a B-tree and contention for B-tree nodes. A nonindexed sorted set is usually most advantageous when set occurrences have a small number of members. It performs best when data is inserted in descending order. Searching begins at the current set when possible. Retrieval is best when scanning the entire set, as in report generation. Single lookup transactions result in walking the set.

INDEX [NODE SIZE IS unsigned-integer BYTES]
INDEX mode builds a B-tree structure using the sort keys defined in the schema. These keys, together with pointers, are grouped into index nodes, the size of which is determined by unsigned-integer. INDEX mode is allowed only with SORTED set types.

See the *Oracle CODASYL DBMS Database Design Guide* for formulas to determine the average, minimum, and maximum node size.

OMIT PRIOR
NEXT and OWNER pointers are provided for member record occurrences but PRIOR pointers are not. This makes backward set walking very difficult.

Examples

1.
```
SET NAME IS ALL_PARTS ----❶
  MODE IS CALC
    MEMBER IS PART -------❷
      KEY IS PART_ID
* Member is PART ---------❸
```

These examples demonstrate the different modes:

❶ Identifies ALL_PARTS as the schema set type for which the storage characteristics are being defined.

❷ Indicates that DBCS calculates the location of the set owner occurrence and the related member occurrences from an occurrence of the PART_ID data item type of the PART record type.

❸ Comment line identifies PART as a member. Comment lines are inserted when you use DBO/EXTRACT/OPTION=FULL or DBO/DUMP/OPTION=FULL commands.

2.
```
SET NAME IS PART_INFO -----❶
  MODE IS CHAIN -----------❷
* Member is SUPPLY    |-----❸
* Member is QUOTE     |
```

❶ Identifies PART_INFO as the schema set type for which the storage characteristics are being defined.

❷ Indicates that the owner and member record occurrences are linked sequentially.

❸ Comment lines identify SUPPLY and QUOTE as members. Comment lines are inserted when you use DBO/EXTRACT/OPTION=FULL or DBO/DUMP/OPTION=FULL commands.

3.
```
SET NAME IS ALL_VENDORS -------❶
  MODE IS INDEX
    NODE SIZE IS 400 BYTES ----❷
* Member is VENDOR ------------❸
```

❶ Identifies ALL_VENDORS as the schema set type for which the storage characteristics are being defined.

❷ Indicates the set type is indexed; each index node is 400 bytes long.

❸ Comment line inserted by the DBO/EXTRACT or DBO/DUMP utilities.

4
Subschema Definition

Use the subschema definition of the Oracle CODASYL DBMS data definition language (DDL) to provide a subset of the schema for user applications. The subschema identifies the elements of the schema used by the application.

This chapter is divided into five sections. Each section describes an entry of the subschema:

- SUBSCHEMA entry

  Provides the syntax for naming the subschema and associating it with a schema

- Subschema ALIAS entry

  Provides the syntax for renaming schema areas, record types, data item types, or set types for the subschema

- Subschema REALM entry

  Provides the syntax for identifying realms in the subschemas

- Subschema RECORD entry

  Provides the syntax and argument descriptions for including schema record types in a subschema

- Subschema SET entry

  Provides the syntax and argument descriptions for including schema set types in a subschema

The following syntax presents the major subschema divisions. The ALIAS, REALM, RECORD, and SET clauses must be repeated for each alias, realm, record type, and set type you want to include in a subschema:

- SUBSCHEMA NAME IS . . .
- [ALIAS] . . .
- {REALM realm-name IS} . . .
- {RECORD NAME IS record-name} . . .
- [SET NAME IS set-name] . . .

A subschema must include at least the SUBSCHEMA entry, one REALM entry, and one RECORD entry.

If you have DEC Language-Sensitive Editor (LSE) installed on your system, you can use LSE templates to write your subschema. There is an LSE template available for the data definition language (DDL). To invoke LSE, type the LSEDIT command followed by the name of the file you want to edit:

```
LSEDIT filename
```

If you use the .DDL file type the template is automatically invoked for the editing session.

Once you are using the template, a placeholder appears on the first line of the editing buffer. A placeholder is text that can be expanded into code. In the DDL template, the placeholder is the following:

```
[database_schema]
```

Expand this placeholder to write a subschema.

See the VMS Language-Sensitive Editor and Source Code Analyzer User Manual and *Introduction to Oracle CODASYL DBMS* to learn more about using LSE.

4.1 SUBSCHEMA Entry

The SUBSCHEMA entry names a subset of the schema and associates it with a schema.

Format

SUBSCHEMA NAME IS subschema-name

FOR schema-name SCHEMA

Arguments

SUBSCHEMA NAME IS subschema-name
Names the subschema. There is a 31-character limit on subschema name.

FOR schema-name SCHEMA
Identifies the associated schema.

Examples

1.
```
SUBSCHEMA NAME IS PARTSS1 FOR PARTS SCHEMA ----------------❶
* Subschema version number : 16-MAR-1981 15:06:26:07    |--❷
* Schema version number : 16-MAR-1981 15:04:50:03       |
```

This example illustrates the SUBSCHEMA entry for a user-written subschema:

❶ Names the subschema (PARTSS1) and associates it with the PARTS schema.

❷ Comment lines tell you when the subschema and schema were compiled. Comment lines are inserted when you use DBO/EXTRACT/OPTION=FULL or DBO/DUMP/OPTION=FULL commands.

2.
```
SUBSCHEMA NAME IS DEFAULT_SUBSCHEMA FOR PARTS SCHEMA
```

This example illustrates the SUBSCHEMA entry for a subschema produced by the default subschema generator. It names the default subschema (DEFAULT_SUBSCHEMA) and associates it with the PARTS schema.

4.2 ALIAS Entry

The ALIAS entry allows you to rename schema elements in the subschema. Such renaming becomes necessary if there is a conflict of reserved words, or when user-supplied names violate language syntax rules.

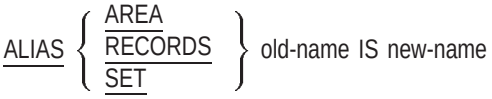
The ALIAS entries must follow the subschema identification entry and precede all other entries.

There are two ALIAS formats:

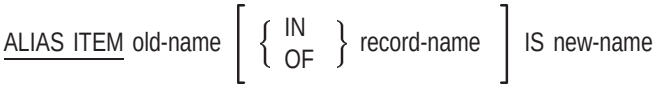
- Format 1 renames areas, record types, and set types.
- Format 2 renames an item from a specified record type.

After the last ALIAS entry, any run unit reference to the renamed entities must use the new name. The old name is no longer valid.

Format 1



Format 2



Default

ALIAS ITEM
If you do not qualify the data item type with the record name clause, the DDL compiler renames any data item type it finds with that name.

Arguments

ALIAS AREA old-name IS new-name
Renames a schema area specified by old-name to new-name.

ALIAS RECORDS old-name IS new-name
Renames a schema record type specified by old-name to new-name.

ALIAS SET old-name IS new-name
Renames a schema set type specified by old-name to new-name.

ALIAS ITEM old-name IN record-name IS new-name
Renames schema item type old-name in record-name to new-name.

Examples

1.
```
ALIAS RECORD DIVISION IS WK_GROUP ----------------❶
ALIAS ITEM DIV_NAME OF DIVISION IS GROUP_NAME ----❷
ALIAS RECORD QUOTE IS PR_QUOTE -------------------❸
```

This example renames two record types and an item for use in the subschema:

❶ Renames the DIVISION record type to WK_GROUP.

❷ Renames DIV_NAME in the DIVISION record type to GROUP_NAME. If DIV_NAME were not qualified by DIVISION, all data items named DIV_NAME would be renamed in this subschema.

❸ Renames the QUOTE record type to PR_QUOTE.

2.
```
ALIAS SET PART_INFO IS PART_SUPPLY
```

This example renames the PART_INFO set to PART_SUPPLY.

4.3 REALM Entry

The REALM entry groups one or more schema areas into a realm.

Format

REALM realm-name [IS area-name...]

Default

REALM realm-name
If you omit the area name parameter, the compiler uses the schema area with the same name as realm name if one matches. If no schema area exists with that name, an error occurs. There is a 31-character limit on realm name. Because a 6-character prefix is added to each realm name at compile time, the practical limit is 25 characters.

Argument

REALM realm-name IS area-name
Names the realm to be created from areas. Specifies the areas that make up the realm.

A schema area cannot be in more than one subschema REALM.

Examples

1.
```
REALM BUY ------------❶
REALM MAKE -----------❷
REALM PERSONNEL ------❸
```

This example does not list schema areas for the REALM entry, which associates the named realms with any schema areas of the same name:

❶ Defines the realm BUY as consisting of a schema area named BUY.

❷ Defines the realm MAKE as consisting of a schema area named MAKE.

❸ Defines the realm PERSONNEL as consisting of a schema area named PERSONNEL.

2.
```
REALM BUY
    IS BUY ---------❶

REALM MAKE
    IS MAKE --------❷

REALM PERSONNEL
    IS PERSONNEL ---❸
```

This example associates one schema area with one subschema realm:

❶ Makes the schema area BUY available to this subschema as the realm BUY.

❷ Makes the schema area MAKE available to this subschema as the realm MAKE.

❸ Makes the schema area PERSONNEL available to this subschema as the realm PERSONNEL.

3.
```
REALM MARKETS ------❶
    IS BUY      -+
       MAKE      |--❷
       MARKET   -+
```

This example groups several schema areas into one subschema realm.

❶ Names the realm, MARKETS.

❷ Makes three schema areas, BUY, MAKE, and MARKET, available to this subschema in the realm MARKETS.

4.4 Subschema RECORD Entry

The subschema RECORD entry includes a schema record type in the subschema.

4.4 Subschema RECORD Entry

Format

```
RECORD NAME IS record-name

  [ ALL ITEMS                    ]
  [ { item-entry  }...           ]
  [ { group-entry }              ]

Where item-entry consists of:

  ITEM NAME IS { FILLER    }
               { data-item }
  [ TYPE IS data-type ]
  [ OCCURS unsigned-integer TIMES ]
  [ QUERY_NAME IS query-name ]
  [ QUERY_HEADER IS query-header ]
  [ EDIT_STRING IS edit-string ]

Where group-entry consists of:

  GROUP NAME IS group-name
    [ OCCURS unsigned-integer TIMES ]
    [ QUERY_NAME IS query-name ]

    { item-entry  }...
    { group-entry }
  ENDGROUP [ group-name ]
```

Subschema record subentries allow you to tailor schema record types to the needs of the subschema.

If the run units using this subschema will store or modify occurrences of this record type, include in the subschema all set types of which this record type is an AUTOMATIC member. Otherwise, DBCS will not be able to find a position to STORE occurrences of this record type in its set types. However, if the AUTOMATIC sets of which this record type is a member are SYSTEM-owned with an order of FIRST, LAST, SORTED, or the system default, this restriction does not apply.

A subschema record type can consist of no data item types (for example, when you use a record type as a stepping-stone to other record types) or one or more elementary data item types or group item types. An elementary data item type is one that cannot be logically subdivided (ITEM NAME IS . . .). A group item type contains subordinate elementary data item types, other group item types, or any combination of group and elementary data item types. The item entry can contain up to 50 groups entries.

The subschema record entry also supports DEC DATATRIEVE query names, query headers, and edit strings. Features specific to DATATRIEVE let you tailor a subschema to the needs of DATATRIEVE users of Oracle CODASYL DBMS.

Default

TYPE IS data-type
If you do not specify a data type for a data item type, the compiler provides the data type used in the schema for that data item type. You must specify a data type if you specify FILLER for the item name clause.

Arguments

RECORD NAME IS record-name
Includes a schema record type in the subschema. There is a 31-character limit on record name. Because a 6-character prefix is added to each record name at compile time, the practical limit is 25 characters.

ALL ITEMS
Includes in the subschema all data item types defined in the schema. The data types defined for data item types in the schema are automatically included.

item-entry
Consists of one or more schema data item types or FILLER. The data-item identifier names the schema data item type to be included in the record type. FILLER reserves space in the subschema record type layout that will not be used by Oracle CODASYL DBMS. Using FILLER allows a user to exactly define the layout of a subschema record type, possibly to match up with the format of an existing file.

The TYPE IS data-type line of this entry specifies the data type. You must specify a data type for FILLER; the data type clause is optional for data item types defined in the schema. Data types do not have to agree between the schema and subschema. See Chapter 7 for information about supported data types and their uses.

group-entry
Consists of any combination of elementary and group item types. The only restriction is that a group item type cannot be empty. All group or elementary data item types that follow this entry in a group item type become part of the group identified by group name. Grouping schema data item types moves all of the component item types into a run unit's user work area at once, thus eliminating multiple requests for individual data item types.

The group entry can be nested up to 50 times. Each group entry must include at least one subordinate entry consisting of elementary or group item types and end with ENDGROUP. The ENDGROUP line of the entry can also specify the group name as a check.

Use the OCCURS line of the group entry to structure a repeating group. See the *Oracle CODASYL DBMS Database Design Guide* for more information on repeating groups.

TYPE IS data-type
Specifies the data type for the data item type.

See Chapter 7 for information about the data types and their uses.

OCCURS unsigned-integer TIMES
Identifies a data item type as a repeating item. Specify the number of times it occurs.

The integer value (greater than 0) supplied is the same for each occurrence.

QUERY_NAME IS query-name
Allows you to assign another name to a data item type. The name must meet these requirements:

- Must be 1 to 31 characters long

- Must start with an alphabetic character and can contain alphabetic characters and underscores
- Cannot be a reserved word

This clause is for DATATRIEVE applications only. See the documentation for DEC Language-Sensitive Editor and DEC Source Code Analyzer for more information on query names.

QUERY_HEADER IS query-header

Specifies the column header to be used when the data item value is printed. You can specify a list of query-header character string literals; each must be enclosed within quotation marks (" ").

This clause is for DATATRIEVE applications only. See the documentation for DEC Language-Sensitive Editor and DEC Source Code Analyzer for more information on query headers.

EDIT_STRING IS edit-string

Specifies the format of the data item value when it is printed. The edit-string is one or more edit string characters enclosed in quotation marks (" ").

This optional clause is for DATATRIEVE applications only. See the documentation for DEC Language-Sensitive Editor and DEC Source Code Analyzer for more information on edit strings.

Examples

1.
```
RECORD NAME IS COMPONENT ----------------------------❶
* Within areas     : MAKE          -+
* Member of sets   : PART_USES      |----------------❷
*                    PART_USED_ON  -+
  ITEM COMP_SUB_PART TYPE IS CHARACTER 8             -+
  ITEM COMP_OWNER_PART TYPE IS CHARACTER 8            |
  ITEM COMP_MEASURE TYPE CHARACTER 1                  |--❸
  ITEM COMP_QUANTITY TYPE IS UNSIGNED NUMERIC 5 -2   -+
```

This example names the schema data item types to be included in the subschema:

❶ Names the schema record type, COMPONENT, as a subschema record type.

❷ Comment lines identify MAKE as the valid area for COMPONENT record types and identify the sets in which COMPONENT participates. Comment lines are inserted when you use the DBO/EXTRACT/OPTION=FULL or DBO/DUMP/OPTION=FULL commands.

❸ Names the schema data item types that are available to run units using this subschema. Each item clause also specifies the data type for that data item type. For example, COMP_QUANTITY contains UNSIGNED NUMERIC data with a length of 5 and a scale of -2.

2.
```
RECORD NAME IS COMPONENT
  ITEM COMP_SUB_PART TYPE IS CHARACTER 8
  ITEM COMP_OWNER_PART TYPE IS CHARACTER 8
  ITEM COMP_MEASURE TYPE CHARACTER 1
  ITEM COMP_QUANTITY TYPE IS SIGNED LONGWORD
```

This example defines the same record type for a FORTRAN application. This record type redefines the data type for COMP_QUANTITY to UNSIGNED LONGWORD. Oracle CODASYL DBMS automatically performs the necessary data conversion.

See Chapter 7 for more information on data types and equivalence.

3.
```
RECORD NAME IS EMPLOYEE ------------------❶
* Within areas     : PERSONNEL -----------❷
* Owner of sets    : MANAGES           |--❸
*                    RESPONSIBLE_FOR   |
* Member of sets   : ALL_EMPLOYEES   |----❹
*                    CONSISTS_OF     |
  ITEM EMP_ID TYPE IS UNSIGNED NUMERIC 5
  GROUP NAME IS EMP_NAME                        -+
     ITEM EMP_LAST_NAME TYPE IS CHARACTER 20     |--❺
     ITEM EMP_FIRST_NAME TYPE IS CHARACTER 10   -+
  ENDGROUP EMP_NAME ----------------------------❻
  ITEM EMP_PHONE TYPE IS UNSIGNED NUMERIC 7   |--❼
  ITEM EMP_LOC TYPE IS CHARACTER 5            |
```

This example structures a group item type from schema data items:

❶ Names the schema record type, EMPLOYEE, as a subschema record type.

❷ Comment line identifies PERSONNEL as a valid area for the COMPONENT record type. Comment lines are inserted when you use DBO/EXTRACT/OPTION=FULL or DBO/DUMP/OPTION=FULL commands.

❸ Comment lines identify the sets which EMPLOYEE owns as MANAGES and RESPONSIBLE_FOR. Comment lines are inserted when you use DBO/EXTRACT/OPTION=FULL or DBO/DUMP/OPTION=FULL commands.

❹ Comment lines identify the sets of which EMPLOYEE is a member as ALL_EMPLOYEES and CONSISTS_OF. Comment lines are inserted when you use DBO/EXTRACT/OPTION=FULL or DBO/DUMP/OPTION=FULL commands.

❺ Structures a GROUP named EMP_NAME from two schema data item types, EMP_LAST_NAME and EMP_FIRST_NAME.

❻ Ends the group named EMP_NAME.

❼ Names other schema data item types that are available to run units using this subschema. Each item clause also specifies the data type for that data item type. For example, EMP_PHONE contains UNSIGNED NUMERIC data with a length of 7.

4.
```
RECORD NAME IS COMPONENT --------------❶
*   Within areas   : MAKE          -+
*   Member of sets : PART_USES      |--❷
*                    PART_USED_ON  -+
  ITEM COMP_SUB_PART TYPE IS CHARACTER 8              -+
  ITEM COMP_OWNER_PART TYPE IS CHARACTER 8             |
  ITEM COMP_MEASURE TYPE IS CHARACTER 1                |-----❸
    QUERY_HEADER IS "Unit of measure" --------❹        |
  ITEM COMP_QUANTITY TYPE IS UNSIGNED NUMERIC 5 -2     |
    QUERY_HEADER IS "Quantity" -----------❺           |
    EDIT_STRING IS "Z99.99" -----------------❻       -+
```

This example demonstrates the use of the clauses specific to DATATRIEVE:

❶ Names the schema record type, COMPONENT, as a subschema record type.

❷ Comment lines identify MAKE as a valid area for the COMPONENT record type and identify the sets in which COMPONENT participates. Comment lines are inserted when you use the DBO/EXTRACT/OPTION=FULL or DBO/DUMP/OPTION=FULL commands.

❸ Names the schema data item types available to run units using this subschema. Each item clause also specifies the data type for that data item type. For example, COMP_SUB_PART contains alphanumeric data with a length of 8.

❹ Inserts a DATATRIEVE query header ("Unit of measure") in the COMP_MEASURE data item type.

❺ Inserts a DATATRIEVE query header ("Quantity") in the COMP_QUANTITY data item type.

❻ Inserts a DATATRIEVE edit string ("Z99.99") in the COMP_QUANTITY data item type, describing how it should be formatted.

See the documentation for DEC Language-Sensitive Editor and DEC Source Code Analyzer for more information on using these DATATRIEVE features.

4.5 Subschema SET Entry

The subschema SET entry includes a schema set in the subschema. If you include a schema set type in a subschema, you must also include the owner record type and at least one member record type.

Format

SET NAME IS set-name

Argument

SET NAME IS set-name
Includes a schema set in the subschema. There is a 31-character limit on set name. Because a 6-character prefix is added to each set name at compile time, the practical limit is 25 characters.

Example

```
SET NAME IS ALL_VENDORS ------------------------❶
*  Owner is SYSTEM ----------------------------❷
*  Member is VENDOR ---------------------------❸
*    Insertion is AUTOMATIC retention is MANDATORY --❹
*    Order is SORTED BY      -+
*      DESCENDING VEND_NAME   |--❺
*      DUPLICATES ARE LAST   -+
```

This example demonstrates the use of the subschema SET entry. Only the first line is required to include a schema set in the subschema. All other lines are comments used for internal documentation of the subschema. Comment lines are inserted when you use DBO/EXTRACT/OPTION=FULL or DBO/DUMP/OPTION=FULL commands.

❶ Names ALL_VENDORS as a schema set to be included in this subschema.

❷ Comment line identifies ALL_VENDORS as a SYSTEM-owned set.

❸ Comment line identifies VENDOR as a member.

❹ Comment line identifies the INSERTION and RETENTION for the VENDOR record type.

❺ Comment lines identify the order of VENDOR record type.

5
Security Schema Definition

Use the security schema definition of the Oracle CODASYL DBMS data definition language (DDL) to control the execution of data manipulation language (DML) statements at run time. The areas of control are the ability to grant or deny permission to execute the following:

- READY open-mode statements (usage combinations)
- DML set-level commands
- DML record-level commands
- DML item-level commands

The definition of a security schema must include a SECURITY SCHEMA entry and can include one or more AREA, RECORD, or SET entries. If you do not include AREA, RECORD, or SET entries in the security schema definition, values from the security schema entry are used as defaults.

For example, the access mode default for a security schema is DENY ALL, but you can change it to GRANT CONCURRENT UPDATE. The new access mode becomes the default for all areas, set types, record types, item types, and member record types in the associated schema. You can override the security schema access mode by redefining access privileges in other individual security entries (area, record, or set).

This chapter is divided into four sections. Each section describes one entry of the security schema:

- SECURITY SCHEMA entry

  Provides the syntax for identifying an existing schema, associating the schema with the security schema, and defining security attributes that are global to the security schema.

- Security schema AREA entry

  Provides the syntax for identifying areas in the associated schema and for defining the security attributes of those areas. AREA definitions override global security schema definitions.

- Security schema RECORD entry

  Provides the syntax for identifying record types in the associated schema and for defining the security attributes of those record types. RECORD definitions override global security schema definitions. RECORD entries also provide the syntax for identifying data item types and for defining the security attributes of those data item types. ITEM definitions override RECORD definitions.

- Security schema SET entry

Provides the syntax for identifying existing set types in the associated schema and for defining the security attributes of those set types. SET definitions override global security schema definitions. SET entries also provide the syntax for identifying member record types and for defining the security attributes of those member record types. MEMBER definitions override SET definitions.

Note

Repeat the AREA, RECORD, and SET entries for each area, record type, and set type respectively for which you want to provide specific security controls.

The following syntax presents the major divisions of a security schema:

- SECURITY SCHEMA NAME IS security-schema-name
- [AREA NAME IS area-name]...
- [RECORD NAME IS record-name]...
- [SET NAME IS set-name]...

If you have DEC Language-Sensitive Editor (LSE) installed on your system, you can use LSE templates to write your security schema. There is an LSE template available for the data definition language (DDL). To invoke LSE, type the LSEDIT command followed by the name of the file you want to edit:

```
LSEDIT filename
```

If you use the .DDL file type, the template is automatically invoked for the editing session.

Once you are using the template, a placeholder appears on the first line of the editing buffer. A placeholder is text that can be expanded into code. In the DDL template, the placeholder is the following:

```
[database_schema]
```

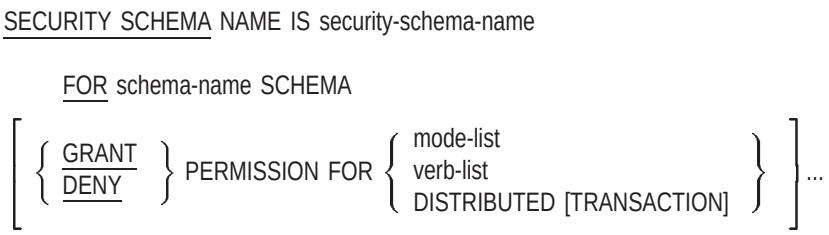
Expand this placeholder to write a security schema.

See the documentation for DEC Language-Sensitive Editor and DEC Source Code Analyzer and *Introduction to Oracle CODASYL DBMS* to learn more about using LSE.

5.1 SECURITY SCHEMA Entry

The SECURITY SCHEMA entry names a security schema, associates that security schema with a schema, and defines the global security attributes of the schema definitions.

Format



Defaults

DENY PERMISSION FOR mode-list
Prevents the execution of all open modes.

DENY PERMISSION FOR verb-list
Prevents the execution of all DML commands.

Arguments

SECURITY SCHEMA NAME IS security-schema-name
Names the security schema. There is a 31-character limit on security schema name.

FOR schema-name SCHEMA
Names the schema associated with the security schema.

GRANT
Permits execution of the corresponding DML operation at run time.

DENY
Prevents execution of the corresponding DML operation at run time.

PERMISSION FOR mode-list
Names a READY mode combination whose use you want to control. The list can contain one or more of the following combinations:

ALL MODES
ALL RETRIEVAL
ALL UPDATE
BATCH MODES
BATCH RETRIEVAL
BATCH UPDATE
CONCURRENT MODES
CONCURRENT RETRIEVAL
CONCURRENT UPDATE
EXCLUSIVE MODES
EXCLUSIVE RETRIEVAL
EXCLUSIVE UPDATE
PROTECTED MODES
PROTECTED RETRIEVAL
PROTECTED UPDATE

PERMISSION FOR verb-list
Lists record-level or set-level DML commands whose use you want to control. The list can contain one or more of the following record-level commands:

ERASE
FIND
IF
MODIFY

GET STORE

The list can contain one or more of the following set-level commands:

CONNECT IF
DISCONNECT RECONNECT

PERMISSION FOR DISTRIBUTED [TRANSACTION]
Enables distributed transactions.

Examples

1.
```
SECURITY SCHEMA NAME IS READ_ONLY FOR PARTS SCHEMA ---❶
  GRANT ALL RETRIEVAL --------------------------------❷
```

❶ Names the security schema, READ_ONLY, and associates it with the PARTS schema.

❷ Lists the usage mode privileges. This example allows only retrieval access to data in the PARTS database.

2.
```
SECURITY SCHEMA NAME IS RSTRD_MODES FOR PARTS SCHEMA ---❶
  DENY  BATCH MODES        -+
        EXCLUSIVE MODES     |
        PROTECTED MODES     |--------------------------❷
  GRANT CONCURRENT MODES   -+
```

❶ Names the security schema, RSTRD_MODES, and associates it with the PARTS schema.

❷ Lists the allow and access mode privileges. This example prevents UPDATE and RETRIEVAL access in the BATCH, EXCLUSIVE, and PROTECTED allow modes to data in the PARTS database. It permits concurrent update and retrieval.

5.2 Security Schema AREA Entry

A schema is composed of one or more areas. The AREA entry for a security schema identifies an existing area and defines its security attributes. You must use a separate AREA entry for each area you want to control. Security attributes defined by an AREA entry override those defined by a SECURITY SCHEMA entry.

Format

AREA NAME IS area-name

[{ GRANT / DENY } PERMISSION FOR mode-list] ...

Default

PERMISSION FOR mode-list
Determined by the security schema mode list entry.

Arguments

AREA NAME IS area-name
Names an area from the associated schema. There is a 31-character limit on area name.

GRANT
Permits execution of the corresponding DML operation at run time.

DENY
Prevents execution of the corresponding DML operation at run time.

PERMISSION FOR mode-list
Names a READY usage mode combination whose use you want to control. The list can contain one or more of the following combinations:

ALL MODES
ALL RETRIEVAL
ALL UPDATE
BATCH MODES
BATCH RETRIEVAL
BATCH UPDATE
CONCURRENT MODES
CONCURRENT RETRIEVAL
CONCURRENT UPDATE
EXCLUSIVE MODES
EXCLUSIVE RETRIEVAL
EXCLUSIVE UPDATE
PROTECTED MODES
PROTECTED RETRIEVAL
PROTECTED UPDATE

Example

```
AREA NAME IS MAKE ----------------❶
  DENY CONCURRENT MODES   -+
       PROTECTED MODES     |------❷
       EXCLUSIVE MODES    -+

AREA NAME IS BUY -----------------❸
  DENY CONCURRENT RETRIEVAL   -+
       BATCH RETRIEVAL         |--❹
       PROTECTED RETRIEVAL    -+
```

❶ Names an existing area, MAKE, in the PARTS schema.

❷ Lists the security attributes of the MAKE area. These entries further restrict execution of READY statement modes listed in Example 1 of the SECURITY SCHEMA entry to allow only BATCH RETRIEVAL of data in the MAKE area.

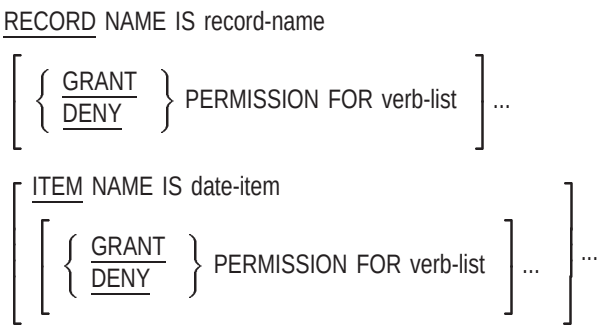
❸ Names an existing area, BUY, in the PARTS schema.

❹ Overrides the GRANT ALL RETRIEVAL access mode defined in Example 1 of the SECURITY SCHEMA entry to allow only the exclusive retrieval of data in the BUY area.

5.3 Security Schema RECORD Entry

This security schema RECORD entry controls a schema record type.

Format



Use the RECORD entry to define security attributes that are global to a record type and to define security attributes that are specific to a data item type.

Use a separate RECORD entry for each record type whose use you want to control. Within a RECORD entry, use separate ITEM entries for each data item type whose use you want to control. Security attributes defined by a RECORD entry override those defined by a SECURITY SCHEMA entry. Security attributes defined by an ITEM entry override those defined by a RECORD entry.

Defaults

PERMISSION FOR verb-list
Determined by the security schema verb list entry.

ITEM NAME IS data-item
Identifies all data item types in the record. There is a 31-character limit on data-item.

Arguments

RECORD NAME IS record-name
Names an existing record type in the schema. There is a 31-character limit on record name. Because a 6-character prefix is added to each record name at compile time, the practical limit is 25 characters.

GRANT
Permits execution of the corresponding DML operation at run time.

DENY
Prevents execution of the corresponding DML operation at run time.

PERMISSION FOR verb-list
Lists the record-level DML commands whose use you want to control. The list can contain one or more of the following commands:

ERASE MODIFY

FIND STORE
GET

ITEM NAME IS data-item
Names a data item type. Repeat this clause for each data item type you want to secure. There is a 31-character limit on data-item.

GRANT
Permits execution of the corresponding DML operation at run time.

DENY
Prevents execution of the corresponding DML operation at run time.

PERMISSION FOR verb-list
Lists the item-level DML commands whose use you want to control. The list can contain one or more of the following commands:

GET
MODIFY

Examples

1.
```
RECORD NAME IS PART ---------❶
  ITEM NAME IS PART_DESC ----❷
    GRANT MODIFY ------------❸
```

The following examples illustrate how to override the DENY default in a record level definition:

❶ Names an existing record type, PART, in the PARTS schema.

❷ Names an existing data item type, PART_DESC, in the PART record type.

❸ Permits modification of the PART_DESC data item type. It overrides the record-level default, DENY MODIFY, for the PART_DESC data item type.

2.
```
RECORD NAME IS PART ------❶
  GRANT GET --------------❷
```

❶ Names an existing record type, PART, in the PARTS schema.

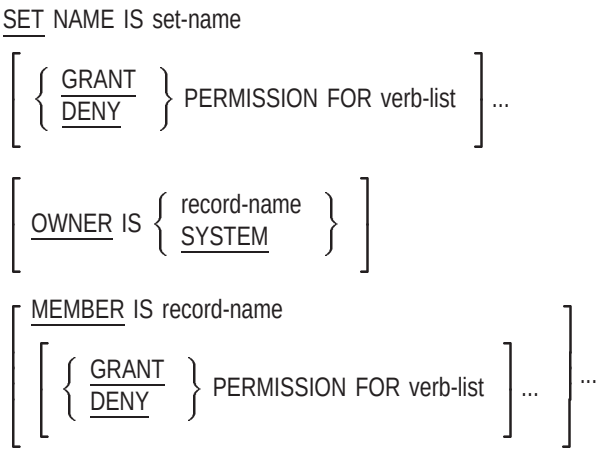
❷ Allows retrieval access to occurrences of the record type. If a GET or FETCH is performed at run time, a GRANT GET is required to override the DENY default.

5.4 Security Schema SET Entry

The security schema SET entry controls a schema set.

5.4 Security Schema SET Entry

Format



Use the SET entry to define security attributes that are global to a set type and to define security attributes that are specific to a member record type.

Use a separate SET entry for each set type whose use you want to control. Within a SET entry, use separate MEMBER entries for each member record type you want to control. Security attributes defined by a SET entry override those defined by a SECURITY SCHEMA entry, and security attributes defined by a MEMBER entry override those defined by a SET entry.

Default

PERMISSION FOR verb-list
Determined by the security schema verb list entry.

Arguments

SET NAME IS set-name
Names an existing set type in the schema. There is a 31-character limit on set name. Because a 6-character prefix is added to each set name at compile time, the practical limit is 25 characters.

GRANT
Permits execution of the corresponding DML operation at run time.

DENY
Prevents execution of the corresponding DML operation at run time.

PERMISSION FOR verb-list
Lists a set-level DML verb whose use you want to control. The list can contain one or more of the following commands:

CONNECT

DISCONNECT

RECONNECT

OWNER IS
Because DBCS uses a set name to determine OWNER information, the OWNER clause only documents the syntax entry. There can be only one OWNER clause for a SECURITY SET entry.

OWNER IS record-name
Names an existing record type that is the owner of a set type.

OWNER IS SYSTEM
Names the SYSTEM as the owner of a set type.

MEMBER IS record-name
Names an existing record type that is a member of this set type. Each set entry can have zero, one, or more MEMBER clauses. Repeat the MEMBER clause for each member record type you want to secure.

GRANT
Permits execution of the corresponding DML operation at run time.

DENY
Prevents execution of the corresponding DML operation at run time.

PERMISSION FOR verb-list
Lists a set-level DML verb whose use you want to control. The list can contain one or more of the following commands:

CONNECT
DISCONNECT
RECONNECT

Example

```
SET NAME IS PART_USED_ON ------------❶
  OWNER IS PART ----------------------❷
  MEMBER IS COMPONENT ----------------❸
    DENY PERMISSION FOR CONNECT -----❹
```

❶ Names an existing set type, PART_USED_ON, in the PARTS schema.

❷ Names the OWNER record type as PART.

❸ Names an existing member record type, COMPONENT, in the PART_USED_ON set type.

❹ Prevents insertion of the member record type, COMPONENT, into the PART_USED_ON set type.

6
DDL Compiler Commands

This chapter describes the syntax necessary for using the DDL compiler.

Use the DDL compiler to:

- Compile a schema, storage schema, subschema, and security schema.
- Generate a default storage schema, default subschema, and default security schema from an existing schema.
- Modify an existing schema. You can also modify storage schemas and security schemas associated with the schema.

You can direct Oracle CODASYL DBMS to store these definitions in either Oracle CDD/Repository or in a metadata export file (.DBM).

If you have DEC Language-Sensitive Editor (LSE) installed on your system, you can use LSE templates to write your schema, subschemas, storage schemas, and security schemas.

There is an LSE template available for the data definition language (DDL). To invoke LSE, type the LSEDIT command followed by the name of the file you want to edit:

```
LSEDIT filename
```

If you use the .DDL file type, the template is automatically invoked for the editing session.

Once you are using the template, a placeholder appears on the first line of the editing buffer. A placeholder is text that can be expanded into code. In the DDL template, the placeholder is the following:

```
[database_schema]
```

See the VMS Language-Sensitive Editor and Source Code Analyzer User Manual and *Introduction to Oracle CODASYL DBMS* to learn more about using LSE.

The following sections discuss the various options of the DDL compiler.

6.1 DDL/COMPILE Command

The DDL/COMPILE command compiles the DDL source file.

The DDL compiler first reads the source file to determine whether to compile a schema, storage schema, subschema, or security schema. This determination affects the type of compilation, how the compilation is stored, and whether or not certain default qualifier values apply.

For instance, the positional qualifier /DEFAULT applies only when compiling a DDL schema source file.

6.1 DDL/COMPILE Command

Format

DDL/COMPILE source-file-spec [,...]

Command Qualifiers	Defaults
/[NO]CONFIRM	/NOCONFIRM
/[NO]DEFAULT[=option,...]	/DEFAULT=(STORAGE_SCHEMA, SUBSCHEMA,SECURITY_SCHEMA)
/DIAGNOSTICS[=file-spec]	/DIAGNOSTICS=source-file-name.DIA
/EXPORT_FILE[=file-spec]	See description
/[NO]LIST [,...]	/NOLIST
/[NO]LOAD_CDD	/LOAD_CDD
/PATH=cdd-repository-path	/PATH=CDD$DEFAULT
/[NO]REPLACE	/NOREPLACE
/[NO]WARNINGS	/WARNINGS

Parameter

source-file-spec [,...]
Specifies a source file to be compiled. The default file type is .DDL.

Command Qualifiers

/CONFIRM
/NOCONFIRM
Used only in conjunction with the /REPLACE qualifier. Use /CONFIRM to protect against unintended deletions. This qualifier causes a delete confirmation prompt to be displayed.

/DEFAULT
/DEFAULT=STORAGE_SCHEMA
/DEFAULT=SUBSCHEMA
/DEFAULT=SECURITY_SCHEMA
/NODEFAULT
/NODEFAULT=STORAGE_SCHEMA
/NODEFAULT=SUBSCHEMA
/NODEFAULT=SECURITY_SCHEMA
Applies only when the source file describes a schema. It requests or suppresses compilation of a default storage schema, subschema, and security schema for the compiled schema. The following list describes the qualifiers of the source-file-spec parameter:

- /DEFAULT

  The storage schema, subschema, and security schema defaults are compiled. If you are compiling a schema, this form is the default. This is the same as specifying:

  ```
  /DEFAULT=(STORAGE_SCHEMA,SUBSCHEMA,SECURITY_SCHEMA)
  ```

- /DEFAULT=STORAGE_SCHEMA

  The default storage schema is compiled but not the default subschema or the default security schema.

- /DEFAULT=SUBSCHEMA

  The default subschema is compiled but not the default storage schema or the default security schema.

- /DEFAULT=SECURITY_SCHEMA

  The default security schema is compiled but not the default storage schema or the default subschema.

- /NODEFAULT

  No default compilation occurs.

- /NODEFAULT=SUBSCHEMA

  The default subschema is not compiled, but the default storage schema and the default security schema are compiled.

- /NODEFAULT=STORAGE_SCHEMA

  The default storage schema is not compiled, but the default subschema and the default security schema are compiled.

- /NODEFAULT=SECURITY_SCHEMA

  The default security schema is not compiled, but the default storage schema and the default subschema are compiled.

/DIAGNOSTICS[=file-spec]
Produces an ASCII diagnostics file. If you do not specify a file specification, DDL uses the source file name with the file type .DIA as the name for the diagnostics file name.

/EXPORT_FILE[=file-spec]
Produces a metadata file, file type .DBM, that can be used with the DBO/CREATE command to create a database. If you do not provide a file specification, Oracle CODASYL DBMS creates the file name by using the schema name it finds in the source file and appending a file extension of .DBM.

It is invalid to specify the /EXPORT_FILE and /PATH qualifiers in the same DDL command.

/LIST[=file-spec]
/NOLIST
Determines whether or not compilation output is placed in a listing file and identifies the output file if applicable. The /LIST qualifier is the default for batch jobs.

The /NOLIST qualifier suppresses the output listing and is the default.

The /LIST qualifier directs compilation output to a file with the file specification source-file-spec.LIS.

The /LIST=file-spec qualifier directs compilation output to the file you specify.

/LOAD_CDD
/NOLOAD_CDD
If you specify /LOAD_CDD, the compilation output is stored in either the export file specified with the /EXPORT_FILE qualifier, or in the Oracle CDD/Repository directory you specify with the /PATH qualifier. By default, data is loaded into the CDD$DEFAULT directory.

If you specify /NOLOAD_CDD, the compilation process performs a syntax check but does not load the information in either an export file or the Oracle CDD/Repository directory. In this case, the /PATH or /EXPORT_FILE qualifier is ignored if it is present.

/PATH=cdd-repository-path
Specifies the Oracle CDD/Repository path to the Oracle CDD/Repository directory in which the compiled source is to be loaded. In the case of a storage schema, subschema, or security schema, this is the repository directory in which the associated schema is stored.

If you do not specify a path, the logical name CDD$DEFAULT defines the path to the repository directory.

It is invalid to specify the /PATH and the /EXPORT_FILE qualifiers in the same DDL command.

/REPLACE
/NOREPLACE
Meaningful only on compilations for which /LOAD_CDD is in effect.

Using /REPLACE, existing schemas, storage schemas, subschemas, and security schemas are deleted as necessary, unless they are used in existing databases.

The /NOREPLACE qualifier is the default. This option states that this compilation must not replace an existing compilation of the same name.

Using /NOREPLACE results in an error if you attempt a compilation that would have the same name and type as an existing compilation in the same Oracle CDD/Repository directory. In this case, no replacement occurs and the new compilation is not loaded into Oracle CDD/Repository.

If there are no such conflicts, this qualifier has no effect.

/WARNINGS
/NOWARNINGS
Use /WARNINGS to instruct the DDL compiler to produce diagnostic messages for warning and information level conditions. The /WARNINGS qualifier is the default.

Use /NOWARNINGS to override the default.

Examples

1. `$ DDL/COMPILE PARTS`

   This example is a simple case of a schema compilation in which a default storage schema, default subschema, and default security schema are compiled.

2. `$ DDL/COMPILE PARTS/NODEFAULT`

   This example demonstrates the use of the /NODEFAULT qualifier on DDL/COMPILE to suppress the generation of a default storage schema, default subschema, and default security schema.

3. `$ DDL/COMPILE PARTST1`

   This example compiles the PARTS1 storage schema.

4. `$ DDL/COMPILE/NOLOAD_CDD PARTS`

   This example compiles a schema but does not load Oracle CDD/Repository. The DDL compiler performs syntactic and semantic checking only.

6.2 DDL/GENERATE Command

The DDL/GENERATE command generates the default storage schema, subschema, and security schema for an existing schema.

Format

DDL/GENERATE schema-name [,...]

Command Qualifiers	Defaults
/[NO]DEFAULT[=option,...]	/DEFAULT=(STORAGE_SCHEMA, SUBSCHEMA,SECURITY_SCHEMA)
/EXPORT_FILE[=file-spec]	See description
/[NO]REPLACE	/NOREPLACE
/[NO]WARNINGS	/WARNINGS

Parameter

schema-name [,...]
Specifies the name of a compiled schema.

Command Qualifiers

/DEFAULT
/DEFAULT=STORAGE_SCHEMA
/DEFAULT=SUBSCHEMA
/DEFAULT=SECURITY_SCHEMA
/NODEFAULT
/NODEFAULT=STORAGE_SCHEMA
/NODEFAULT=SUBSCHEMA
/NODEFAULT=SECURITY_SCHEMA
Requests or suppresses compilation of a default storage schema, subschema, and security schema for the compiled schema. The following list describes the qualifiers of the schema name parameter:

- /DEFAULT

  The storage schema, subschema, and security schema defaults are compiled. This is the same as specifying:

  ```
  $ DDL/COMPILE/DEFAULT=(STORAGE_SCHEMA,SUBSCHEMA,SECURITY_SCHEMA)
  ```

- /DEFAULT=STORAGE_SCHEMA

  The default storage schema is compiled but not the default subschema or the default security schema.

- /DEFAULT=SUBSCHEMA

  The default subschema is compiled but not the default storage schema or the default security schema.

- /DEFAULT=SECURITY_SCHEMA

  The default security schema is compiled but not the default storage schema or the default subschema.

- /NODEFAULT

  No default compilation occurs.

- /NODEFAULT=SUBSCHEMA

  The default subschema is not compiled, but the default storage schema and the default security schema are compiled.

- /NODEFAULT=STORAGE_SCHEMA

  The default storage schema is not compiled, but the default subschema and the default security schema are compiled.

- /NODEFAULT=SECURITY_SCHEMA

  The default security schema is not compiled, but the default storage schema and the default subschema are compiled.

/EXPORT_FILE[=file-spec]
Inserts the requested default metadata into an existing .DBM file. The .DBM file can be used with the DBO/CREATE command to create a database.

If you do not provide a file specification, Oracle CODASYL DBMS expects to find a .DBM file with the same name as that of the schema specified on the command line. If you specify a list of schema names on the command line, do not provide a file-spec with the /EXPORT_FILE qualifier; because you can specify only one file specification with the /EXPORT_FILE qualifier, the existing .DBM files that correspond to each schema must be named schema-name.DBM.

If the .DBM file or files are not found, an error is returned.

/REPLACE
/NOREPLACE
Using /REPLACE, existing default storage schemas, default subschemas, and default security schemas are deleted as necessary, unless they are used in existing databases.

The /NOREPLACE qualifier is the default. This option states that this compilation must not replace an existing default compilation of the same type for that schema. If there are no such conflicts, this qualifier has no effect.

/WARNINGS
/NOWARNINGS
Using /WARNINGS instructs the DDL compiler to produce diagnostic messages for warning and information level conditions. The /WARNINGS qualifier is the default. Use /NOWARNINGS to override the default.

Examples

1. `$ DDL/GENERATE PARTS/DEFAULT`

   Generates a default storage schema, default subschema, and default security schema from an existing schema.

2. `$ DDL/GENERATE PARTS/REPLACE/DEFAULT=SUBSCHEMA`

   Generates a default subschema from an existing schema. The /REPLACE qualifier implies that there are no existing databases for the default subschema.

3.
```
$ DDL/GENERATE PARTS/REPLACE/DEFAULT=(SUBSCHEMA,STORAGE_SCHEMA)
or
$ DDL/GENERATE PARTS/REPLACE/NODEFAULT=SECURITY_SCHEMA
```

   Both alternatives in Example 3 first generate a default subschema and a default storage schema from an existing schema and then delete existing default subschemas and default storage schemas as necessary. The first alternative uses an explicit approach that requests the generation of the subschema and storage schema. The second alternative uses an implicit approach that suppresses the generation of the security schema.

6.3 DDL/MODIFY Command

The DDL/MODIFY command modifies an existing schema, storage schema, or security schema.

When modifying a schema, you can also modify storage schemas and security schemas associated with that schema; in this case, changes made to the schema are reflected in the storage schemas and security schemas that you specify.

DDL/MODIFY can add new item, record, and set definitions to the DDL source and recompile the source. Then you can incorporate the new DDL data definitions into existing databases with the DBO/MODIFY command.

Subschemas cannot be modified with DDL/MODIFY. Subschemas must be recompiled when changed and then incorporated into databases that use them.

Format

DDL/MODIFY source-file-spec [,...]

Command Qualifiers	Defaults
/EXPORT_FILE[=file-spec]	See description
/[NO]LIST[=file-spec]	/NOLIST
/[NO]LOAD_CDD	/LOAD_CDD
/PATH=cdd-repository-path	/PATH=CDD$DEFAULT
/[NO]SECURITY_SCHEMA	See description
/[NO]STORAGE_SCHEMA	See description
/[NO]WARNINGS	/WARNINGS

Parameter

source-file-spec
Specifies the source file containing the modified schema, storage schema, or security schema. Default file type is .DDL.

Command Qualifiers

/EXPORT_FILE[=file-spec]
Specifies an existing .DBM file into which metadata modifications are entered.

If you do not provide a file specification, Oracle CODASYL DBMS expects to find a .DBM file whose name matches the schema name found in the source file you specified on the command line. If the .DBM file is not found, an error is returned.

It is invalid to specify the /EXPORT_FILE and /PATH qualifiers in the same DDL command.

/LIST[=file-spec]
/NOLIST
Determines whether or not compilation output is placed in a listing file and identifies the output file if applicable. The /LIST qualifier is the default for batch jobs.

The /NOLIST qualifier suppresses the output listing and is the default.

The /LIST qualifier directs compilation output to a file with the file specification source-file-spec.LIS.

The /LIST=file-spec qualifier directs compilation output to the file you specify.

/LOAD_CDD
/NOLOAD_CDD
If you specify /LOAD_CDD, the compilation output is stored in either the export file specified with the /EXPORT_FILE qualifier, or in the Oracle CDD/Repository directory you specify with the /PATH qualifier. By default, data is loaded into the CDD$DEFAULT directory.

If you specify /NOLOAD_CDD, the compilation process performs a syntax check but does not load the information in either an export file or the Oracle CDD/Repository directory. In this case, the /PATH or /EXPORT_FILE qualifier is ignored if it is present.

/PATH=cdd-repository-path
Specifies the Oracle CDD/Repository path to the Oracle CDD/Repository directory in which the compiled source is to be loaded. In the case of a storage schema or security schema, this is the repository directory in which the associated schema exists.

If you do not specify a path, the logical name CDD$DEFAULT defines the path to the repository directory.

It is invalid to specify the /EXPORT_FILE and /PATH qualifiers in the same DDL command.

/SECURITY_SCHEMA[=(security-schema-name[,...]])
/NOSECURITY_SCHEMA
Specifies which security schema, if any, is to be updated to match the changes to the schema. If no value is supplied to /SECURITY_SCHEMA, DEFAULT_SECURITY_SCHEMA is modified.

Note

The /[NO]STORAGE_SCHEMA and /[NO]SECURITY_SCHEMA qualifiers are valid only when the source-file-spec specifies a schema DDL file. These qualifiers help the database administrator maintain the necessary one-to-one correspondence between schema elements and storage or security schema elements.

/STORAGE_SCHEMA[=(storage-schema-name[,...]])
/NOSTORAGE_SCHEMA
Specifies which storage schema, if any, is to be updated to match the changes to the schema. If no value is supplied to /STORAGE_SCHEMA, DEFAULT_STORAGE_SCHEMA is modified.

/WARNINGS
/NOWARNINGS
Use /WARNINGS to instruct the DDL compiler to produce diagnostic messages for warning and information level conditions. The /WARNINGS qualifier is the default.

Use /NOWARNINGS to override the default.

Examples

1. `$ DDL/MODIFY PARTS/LIST`

   This example modifies a schema called PARTS and produces a listing.

2. `$ DDL/MODIFY PARTST1/PATH="CDD$TOP.DBMS.USER"`

   This example modifies a storage schema called PARTST1 and specifies the Oracle CDD/Repository directory in which the associated schema is stored.

3. `$ DDL/MODIFY READ_ONLY/PATH="CDD$TOP.DBMS.SECR"`

   This example modifies a security schema called READ_ONLY and specifies the Oracle CDD/Repository directory in which the associated schema is stored.

4. The following steps show a typical context for using DDL/MODIFY. For simplicity, it is assumed that the Oracle CDD/Repository pathnames are the same for the modified versions of the metadata.

5.

   a. Use DDL/COMPILE to compile a schema and generate the default storage schema, security schema, and subschema into Oracle CDD/Repository. The compiled schema and defaults are known collectively as Oracle CDD/Repository metadata.

   b. Use DBO/CREATE to create a database from the Oracle CDD/Repository metadata generated in step a. Having created the database, you can delete the metadata from Oracle CDD/Repository.

   c. When you want to modify the database schema and storage schema or produce additional security schemas, use DBO/DUMP to get the DDL source from the database root file (or use DBO/EXTRACT to get source from Oracle CDD/Repository). To get the schema source for the database PARTS, you can enter the following command:

      ```
      $ DBO/DUMP/SCHEMA/OUT=PARTS_SCHEMA.DDL PARTS
      ```

      If you are making structural changes to the schema, it is advisable to make those changes first because you can make parallel changes to the storage schema and security schema automatically during the DDL/MODIFY and DBO/MODIFY sequence.

   d. Use a text editor to change the DDL source. Be sure to change the name of the default security schema, storage schema, or subschema.

e. Recompile the source to Oracle CDD/Repository with DDL/MODIFY. To ensure that storage schemas and security schemas are modified, use the /STORAGE_SCHEMA and /SECURITY_SCHEMA qualifiers when modifying the schema as follows:

```
$ DDL/MODIFY/STOR=*/SECUR=* PARTS_SCHEMA
```

Note you cannot change subschemas with DDL/MODIFY. You must edit them separately and use DDL/COMPILE to recompile them.

f. Use DBO/MODIFY to incorporate the data structure modifications into the database as follows:

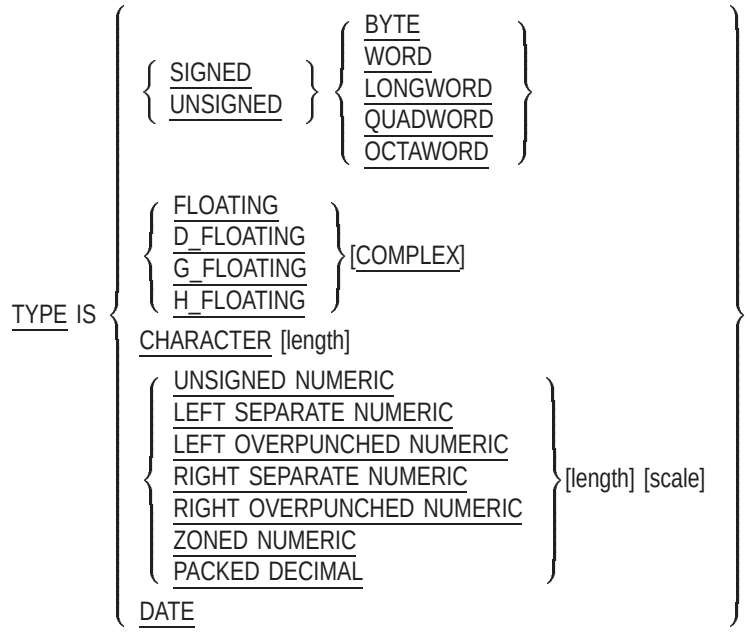
```
$ DBO/MODIFY/SCHEMA/SECUR=* PARTS
```

Using the /SCHEMA qualifier in the DBO/MODIFY command replaces the schema and storage schema.

7

Supported Data Types

Oracle CODASYL DBMS allows you to use all the data types supported by OpenVMS, including INTEGER, FLOATING POINT, CHARACTER STRING, DECIMAL STRING, and DATE classes. Most of these types can be subdivided into data types of different sizes and formats. The following sections describe each of the data types.



Because of the large number of supported data types, the argument descriptions are grouped by type in the following sections:

- INTEGER data types
- FLOATING POINT data types
- CHARACTER STRING data types
- DECIMAL data types
- DATE data types

Note

See the OpenVMS documentation for additional information about the supported data types.

7.1 INTEGER Data Types

The INTEGER data types are used to represent in a binary format quantities that have no scaling. These quantities can be treated as either SIGNED or UNSIGNED. When treated as SIGNED quantities, INTEGERS are represented in two's complement form, with values ranging from -2^{n-1} to $2^{(n-1)-1}$, where *n* is the number of bits of the data type.

When treated as UNSIGNED quantities, INTEGERS range from 0 through 2^{n-1} where there are (*n* bits in the representation. The OpenVMS instruction sets support INTEGER data types of 8, 16, 32, 64, and 128 bit sizes. These are called byte, word, longword, quadword, and octaword INTEGERS, respectively.

Oracle CODASYL DBMS supports the following INTEGER data types as SIGNED and UNSIGNED:

BYTE
Specifies an 8-bit UNSIGNED or SIGNED INTEGER. UNSIGNED bytes have a range of 0 to 255. SIGNED bytes have a range of –128 to 127.

WORD
Specifies a 16-bit UNSIGNED or SIGNED INTEGER. UNSIGNED bytes have a range of 0 to 65535. SIGNED bytes have a range of –32768 to 32767.

LONGWORD
Specifies a 32-bit UNSIGNED or SIGNED INTEGER. UNSIGNED bytes have a range of 0 to 2^{32-1}. SIGNED bytes have a range of -2^{31} to 2^{31-1}.

QUADWORD
Specifies a 64-bit UNSIGNED or SIGNED INTEGER. UNSIGNED bytes have a range of 0 to 2^{64-1}. SIGNED bytes have a range of -2^{63} to 2^{63-1}.

OCTAWORD
Specifies a 128-bit UNSIGNED or SIGNED INTEGER. UNSIGNED bytes have a range of 0 to 2^{128-1}. SIGNED bytes have a range of -2^{127} to 2^{127-1}.

7.2 FLOATING POINT Data Types

The FLOATING POINT data types are used to represent approximations to quantities that have no scaling. Floating-point data is stored in a scientific notation as a power of 2 times a fraction in the range 0.5 (inclusive) to 1.0 (exclusive). The data representation consists of three fields: the sign, the power of 2 exponent, and the fractional magnitude. The VAX, Alpha and Standard 64 Integrity architectures support in the instruction set FLOATING POINT data types of 32, 64 (two formats), and 128 bit sizes. These are called FLOATING, DOUBLE FLOATING, and QUADRUPLE FLOATING, respectively.

Oracle CODASYL DBMS supports the following floating-point data types in both unqualified and COMPLEX form; the /COMPLEX qualifier specifies a complex number in which the lower addressed quantity signifies the real part and the higher addressed quantity represents the imaginary part.

Due to the differences in native floating-point support and run-time libraries in the two environments, results from floating-point operations in Oracle CODASYL DBMS may differ between the OpenVMS VAX, OpenVMS Alpha and HPE OpenVMS Industry Standard 64 for Integrity Servers environments.

FLOATING
Specifies a single-precision, 32-bit F_floating-point number in the range of about 10^{-38} to 10^{38}, with a precision of about 7 decimal digits.

S_FLOATING
Specifies a single-precision, 32-bit S_floating-point (IEEE) number in the range of about 10^{-38} to 10^{38}, with a precision of about 7 decimal digits. Range is 1.17549435e-38 through 3.40282347e38.

D_FLOATING
Specifies a double-precision, 64-bit D_floating-point number in the range of about 10^{-38} to 10^{38}, with a precision of about 16 decimal digits on OpenVMS VAX.

On OpenVMS Alpha and HPE OpenVMS Industry Standard 64 for Integrity Servers systems, D_floating numbers deliver 15 decimal digits of precision. Alpha and I64 architecture processors implement D_floating-point data in software rather than hardware. There is a loss of 3 fraction bits compared to VAX processors.

G_FLOATING
Specifies a double-precision, 64-bit G_floating-point number in the range of about 10^{-308} to 10^{308}, with a precision of about 15 decimal digits. Range is 0.5562684646268004e-308 through 0.898846567431157854 07e308.

T_FLOATING
Specifies a double-precision, 64-bit T_floating-point (IEEE) number in the range of about 10^{-308} to 10^{308}, with a precision of about 15 decimal digits. Range is 2.2250738585072013e-308 through 1.7976931348623158e308.

H_FLOATING
Specifies a quadruple-precision, 128-bit H_floating-point number in the range of about 10^{-4932} to 10^{4932}, with a precision of about 33 decimal digits.

On OpenVMS Alpha and HPE OpenVMS Industry Standard 64 for Integrity Servers systems, H_floating-point data is not supported. You must convert H_ floating-point data to another data type, for example: G_floating, using DRU on an OpenVMS VAX system. If the data is not converted, a run-time error occurs when the data is referenced on an OpenVMS Alpha and and HPE OpenVMS Industry Standard 64 for Integrity Servers systems systems.

7.3 CHARACTER STRING Data Type

The CHARACTER STRING is a data type used to represent strings of characters such as names, data records, or text. Rather than performing arithmetic or logical operations on character strings, the important operations include copying, concatenating, searching, and translating the string.

Oracle CODASYL DBMS supports the following CHARACTER STRING data type:

CHARACTER length
Specifies an alphanumeric character with a length from 0 to 16,383 characters.

length
If you do not specify a value for length, the DDL compiler supplies a value of 1.

7.4 DECIMAL STRING Data Types

The DECIMAL STRING data types are used to represent fixed scale quantities in a form close to their external representation. The DECIMAL STRING data types include formats in which each decimal digit occupies 1 byte (character) and a more compact form in which 2 decimal digits are packed into 1 byte. These are called numeric and packed decimal strings, respectively. Because the numeric string form precisely represents many external data arrangements, it appears in several representations.

The length of a string and the number of bytes it occupies is specified by the length clause. Length must be in the range 0 to 31 digits. Scaling is specified by the scale clause; the scale clause uses the signed, power-of-10 multiplier to convert the internal form to the external form. For example, the digits 123 with a scale of -2 represent the number 1.23.

Oracle CODASYL DBMS supports the following DECIMAL STRING data types:

UNSIGNED NUMERIC length scale
Specifies an unsigned numeric ASCII string. The actual storage is equal to the number of bytes specified in the length clause.

LEFT SEPARATE NUMERIC length scale
Specifies a signed numeric ASCII string in which the leftmost byte is set aside for the sign and length refers to the number of digits. The actual storage, in bytes, is 1 byte longer than length.

LEFT OVERPUNCHED NUMERIC length scale
Specifies a signed numeric ASCII string in which the leftmost digit is combined with the sign of the number. The length is the number of digits, which is the number of bytes the string occupies.

RIGHT SEPARATE NUMERIC length scale
Specifies a signed numeric ASCII string in which the rightmost byte is set aside for the sign and length refers to the number of digits. The actual storage, in bytes, is 1 byte longer than length.

Note

The data types RIGHT OVERPUNCHED NUMERIC and ZONED NUMERIC are the same except for the coding of the rightmost byte. See the OpenVMS documentation for more information on the representation of these two data types.

RIGHT OVERPUNCHED NUMERIC length scale
Specifies a signed numeric ASCII string in which the rightmost digit is combined with the sign of the number. The length is the number of digits, which is the number of bytes the string occupies.

ZONED NUMERIC length scale
Specifies that one digit is represented in storage by one addressable byte. The rightmost byte is also a combination of the last digit and the sign of the number.

PACKED DECIMAL length scale
Specifies a signed decimal number in which each digit occupies 4 bits of storage. A sign is appended to the string, which requires an additional 4 bits of storage. The actual length of the string, therefore, is $length/2 + 1$, rounded down to the nearest byte.

length
If you do not specify a value for length, the DDL compiler supplies a value of 10.

7.5 DATE Data Types

The DATE data type is an unsigned quadword (64 bits) that represents a date and time as the number of 100-nanosecond "ticks" since the OpenVMS system base date and time (00:00 hours, November 17, 1858). Although the time-and-date stamp is stored with 100-nanosecond precision, it is entered or displayed with 10-millisecond precision when treated as a text string. The DATE data type can range from the OpenVMS system base date and time to December 31, 9999 23:59:59.99.

7.5.1 Defining a DATE Data Item

You can define a DATE data item in the schema, storage schema, and subschema using the clause TYPE IS DATE.

7.5.2 DATE Data Type Conversion

A date can be represented as either a CHARACTER, a DATE, or an UNSIGNED QUADWORD data type in the schema, the subschema, and the storage schema. You can use some system services and run-time library routines within programs to convert a date from text to numeric representation. The system services that you can use are: $ASCTIM, $BINTIM, and $NUMTIM. The run-time library routines that you can use are LIB$SYS_ASCTIM and LIB$DATE_TIME. This allows the user to manipulate the date with arithmetic operations on the date's quadword form. These routines can also give the program more flexibility in formatting the date's text string by allowing the program to create its own date string format. This is especially true for displaying dates that have been retrieved from the database. For applications where no such date manipulation is anticipated and the possible Oracle CODASYL DBMS date text formats are acceptable, DATE data items in the schema and storage schema can be defined as CHARACTER 23 data items in the subschemas.

7.5.3 Entering Date and Time as Character Strings

If you treat the storage schema DATE item as a CHARACTER 23 item in the subschema, you can enter the date and time into a DATE item as a text string. Interactive DBQ receives all input as text and converts text to the type used in the storage schema automatically.

If you enter the date and time through callable DBQ, FDML, or COBOL DML, the DATE item appears as a string to the user, but is stored in the database as a DATE data item.

Character input to a DATE data item consists of an optional date field and an optional time field. The time field, when used, must be separated from the date field by a space, and must contain at least the hours field and a colon delimiter. Delimiters can include any number of blank and tab characters.

DATE data input is flexible, allowing many formats. The date and time fields have separate formats and both can be abbreviated to some extent. The following shows the legal text syntax in Backus-Naur Form (BNF):

```
date text format :: {date word | [date] [time]}
date::
    { [MMM delim day [delim year]] |
      [day delim MMM [delim year]] |
      [month_num delim day [delim year]] |
      [day delim month_num [delim year]] |
      [month_num delim year delim day] |
      [day delim year delim month_num] |
      [year delim month_num delim day] |
      [year delim day delim month_num] |
      [month_num DD [YY]] |
      [day MM [YY]] |
      [month_num YY DD] |
      [day YY MM] |
      [year MM DD] |
      [year DD MM] }

date word    :: {null date | YESTERDAY | TODAY | TOMORROW | NOW}

null date    :: [blank | tab] ...

MMM          :: first three or more letters of month name
month_num    :: one- or two-digit month field
MM           :: two-digit month field
day          :: one- or two-digit day field
DD           :: two-digit day field
year         :: {YYYY | YY | Y}
YYYY         :: four-digit year field
YY           :: two-digit year field for years xxYY
Y            :: one-digit year field for years xx00 to xx09

delim        :: {space, comma, dash, slash, period}
space        :: {blank | tab}[...]
comma        :: [space],[space]
dash         :: [space]-[space]
slash        :: [space]/[space]
period       :: [space].[space]

time         :: space hours colon [mins [colon [secs [period
                      [partial secs]]]]]

colon        :: [space]:[space]
hours        :: two-digit hour field
mins         :: two-digit minute field
secs         :: two-digit second field
partial secs :: two-digit fractional second field
```

7.5.3.1 Entering the Date Field

For the date field, there are three major choices:

- DD-MMM-YYYY

  The MMM field must contain at least three leading letters of the month's name. If you enter more than three letters, MMM is truncated to three letters. DD specifies the day of the month and may be one or two digits. YYYY specifies the year, and may be one, two, or four digits. If you specify the year as one or two digits, the current century is assumed. For example, a YY of 82 indicates the year 1982 and a Y of 8 indicates the year 1908.

  A delimiter can be a comma, a slash, a blank, a tab, a dash, or a period with any number of blanks and tabs.

For example, you can enter 9 june 82 and it will be interpreted as 9 JUN 1982.

- MMM-DD-YYYY

  The rules for entering a date in MMM-DD-YYYY format are the same as for DD-MMM-YYYY, except that the month and day fields are transposed. For example, you can enter September 3, 1982.

- NN-NN-NN

  This format can also be of the form NNNNNN (no delimiters). If you omit delimiters, the second and third NN fields must each contain two digits (that is, single-digit values in these positions need a leading zero).

  With or without delimiters, you must tell the system the order of the three NN fields. A logical name, SYS$DATE_INPUT, can be defined to specify the default date format. The definition is a 3-character code containing the letters D, M, and Y in any order, indicating the expected order of the day, month, and year fields. Normally, you would define this as a systemwide logical name. For example:

  ```
  $ DEFINE/SYSTEM SYS$DATE_INPUT DMY
  ```

  This defines the system default format as the first NN representing the day; the middle NN, the month; and the last NN, the year.

  To define a system logical name, you need the SYSNAM privilege.

  If the logical name is not defined, MDY is the default.

  You can override the format defined by the system logical name by defining SYS$DATE_INPUT for your own process or group, omitting the /SYSTEM qualifier. For example:

  ```
  $ DEFINE SYS$DATE_INPUT YMD
  ```

7.5.3.2 Special DATE Input and Defaults

Some date formats can be abbreviated. For example, to enter the current date (today), you can omit the date field and enter the time. If the current year is desired, you can omit the second delimiter and the year field with the DD-MMM-YYYY and MMM-DD-YYYY formats and also with types MDY and DMY of format NN-NN-NN. Using other formats, all three date fields are required. You can also specify date using the date words:

YESTERDAY
TODAY
TOMORROW
NOW

Any of the first three words yields the appropriate date, with the time field set to 0. NOW also includes the current time. If only a blank field is entered as the date, the item is set to 0 (that is, base time is used).

7.5.3.3 Entering the Time Field

The general format of the time field is:

space hours : minutes : seconds . partial_seconds

The time fields are entered and filled from left to right. Any fields not entered default to 0. The space, colon, and period delimiters can contain any number of blank and tab characters.

The hours field must be preceded by a space and followed by a colon when a previous date field is present. No space is necessary when entering time without a previous date field.

7.5.4 DATE Output

Although there are many input formats, the only output format is the standard format used by the OpenVMS time system services. The format is:

```
DD-MMM-YYYY HH:MM:SS.FF
```

The only exception to this is the output of the base time of 17-Nov-1858 00:00:00.00. This time is interpreted as the null time and the output for it is all blanks.

If the output field is too long, it is truncated from the right.

7.6 Errors, Equivalence, and Conversions

Run-time conversion of data is performed between the storage schema and the subschema; the schema data type does not matter in this case. You should pay particular attention to the validity of conversion between the storage schema and the subschema. For example, the storage schema specifies a data item 2 digits long, but the subschema allows 40 digits for that same item. An error results when a run unit using that subschema attempts to store a value 40 digits long. Furthermore, numbers that are larger than the specified range of a data type or where the absolute values are too large result in an error.

Because DBCS routinely rounds down numbers, overflow and truncation problems that involve the low-order digit do not result in an error. For the same reason, a number too close to 0 will be rounded to 0.

Although Oracle CODASYL DBMS supports all Oracle data types, not all languages support all the data types. This fact must be taken into consideration when you write a subschema to be used by anything other than interactive DBQ. If a data type for a schema data item is not supported by a language, make sure that any subschema written for that language contains a supported data type. For example, the schema might contain the following data item definition:

```
ITEM NAME IS TRUNNION TYPE IS UNSIGNED QUADWORD
```

The data type UNSIGNED QUADWORD, however, is not supported by Pascal. A subschema written for a Pascal application, therefore, would have to override the schema definition for that data item with a data type that is supported by Pascal.

Table 7–1 through Table 7–10 summarize the data types supported by the following languages: Ada, BASIC, BLISS, C, COBOL, DIBOL, FORTRAN, MACRO-32, Pascal, and PL/I.

Table 7–1 Ada Data Types

Generic Data Type	Ada Data Type
SIGNED Integer	
Signed Byte	SHORT_SHORT_INT
Signed Word	SHORT_INT

(continued on next page)

Table 7–1 (Cont.) Ada Data Types

Generic Data Type	Ada Data Type
SIGNED Integer	
Signed Longword	INTEGER
Signed Quadword	Not supported
Signed Octaword	Not supported
UNSIGNED Integer	
Unsigned Byte	UNSIGNED_BYTE
Unsigned Word	UNSIGNED_WORD
Unsigned Longword	UNSIGNED_LONGWORD
Unsigned Quadword	UNSIGNED_QUADWORD
Unsigned Octaword	Not supported
FLOATING	
F_Floating	F_FLOAT
D_Floating	D_FLOAT
G_Floating	G_FLOAT
H_Floating	H_FLOAT
FLOATING COMPLEX	
F_Floating Complex	Not supported
D_Floating Complex	Not supported
G_Floating Complex	Not supported
H_Floating Complex	Not supported
CHARACTER and DATE	
Character	STRING
Date	Not supported
NUMERIC	
Unsigned Numeric	Not supported
Left Separate Numeric	Not supported
Right Separate Numeric	Not supported
Left Overpunched Numeric	Not supported

(continued on next page)

Table 7–1 (Cont.) Ada Data Types

Generic Data Type	Ada Data Type
NUMERIC	
Right Overpunched Numeric	Not supported
Zoned Numeric	Not supported
Packed Decimal	Not supported

Table 7–2 BASIC Data Types

Generic Data Type	BASIC Data Type
SIGNED Integer	
Signed Byte	BYTE
Signed Word	WORD or INTEGER
Signed Longword	LONG or INTEGER
Signed Quadword	Not supported
Signed Octaword	Not supported
UNSIGNED Integer	
Unsigned Byte	Not supported
Unsigned Word	Not supported
Unsigned Longword	Not supported
Unsigned Quadword	Not supported
Unsigned Octaword	Not supported
FLOATING	
F_Floating	SINGLE
D_Floating	DOUBLE
G_Floating	GFLOAT
H_Floating	HFLOAT
FLOATING COMPLEX	
F_Floating Complex	Not supported
D_Floating Complex	Not supported
G_Floating Complex	Not supported
H_Floating Complex	Not supported

(continued on next page)

Table 7–2 (Cont.) BASIC Data Types

Generic Data Type	BASIC Data Type
CHARACTER and DATE	
Character	STRING VARIABLE
Date	Not supported
NUMERIC	
Unsigned Numeric	Not supported
Left Separate Numeric	Not supported
Right Separate Numeric	Not supported
Left Overpunched Numeric	Not supported
Right Overpunched Numeric	Not supported
Zoned Numeric	Not supported
Packed Decimal	DECIMAL (length, scale)[1]

[1]The scale is the number of decimal places such that 1≤length≤31 and 0≤scale≤length.

Table 7–3 BLISS Data Types

Generic Data Type	BLISS Data Type
SIGNED Integer	
Signed Byte	SIGNED BYTE
Signed Word	SIGNED WORD
Signed Longword	SIGNED LONGWORD
Signed Quadword	VECTOR[n,BYTE][1]
Signed Octaword	VECTOR[n, BYTE][1]
UNSIGNED Integer	
Unsigned Byte	UNSIGNED BYTE
Unsigned Word	UNSIGNED WORD
Unsigned Longword	UNSIGNED LONGWORD
Unsigned Quadword	VECTOR[n ,BYTE][1]
Unsigned Octaword	VECTOR[n, BYTE][1]
FLOATING	
F_Floating	VECTOR[n, BYTE][1]
D_Floating	VECTOR[n, BYTE][1]
G_Floating	VECTOR[n, BYTE][1]

[1]The n is the length in bytes.

(continued on next page)

Table 7–3 (Cont.) BLISS Data Types

Generic Data Type	BLISS Data Type
FLOATING	
H_Floating	VECTOR[n, BYTE][1]
FLOATING COMPLEX	
F_Floating Complex	VECTOR[n, BYTE][1]
D_Floating Complex	VECTOR[n, BYTE][1]
G_Floating Complex	VECTOR[n, BYTE][1]
H_Floating Complex	VECTOR[n, BYTE][1]
CHARACTER and DATE	
Character	VECTOR[n, BYTE][1]
Date	VECTOR[8, BYTE]
NUMERIC	
Unsigned Numeric	VECTOR[n, BYTE][1]
Left Separate Numeric	VECTOR[n, BYTE][1]
Right Separate Numeric	VECTOR[n, BYTE][1]
Left Overpunched Numeric	VECTOR[n, BYTE][1]
Right Overpunched Numeric	VECTOR[n, BYTE][1]
Zoned Numeric	VECTOR[n, BYTE][1]
Packed Decimal	VECTOR[n, BYTE][1]

[1]The n is the length in bytes.

Table 7–4 COBOL Data Types

Generic Data Type	COBOL Data Type
SIGNED Integer	
Signed Byte	Not supported[1]
Signed Word	PIC S9(4) COMP
Signed Longword	PIC S9(9) COMP

[1]COBOL has no equivalent for this data type. A warning diagnostic is issued for such an item that is part of a record description entry. The compiler treats that item as if it had been specified as an alphanumeric data item with the same number of bytes.

(continued on next page)

Table 7–4 (Cont.) COBOL Data Types

Generic Data Type	COBOL Data Type
SIGNED Integer	
Signed Quadword	PIC S9(18) COMP
Signed Octaword	Not supported[1]
UNSIGNED Integer	
Unsigned Byte	Not supported[1]
Unsigned Word	Not supported[3]
Unsigned Longword	PIC 9(9) COMP[3]
Unsigned Quadword	PIC 9(18) COMP[3]
Unsigned Octaword	Not supported[1]
FLOATING	
F_Floating	(no PIC) COMP-1
D_Floating	(no PIC) COMP-2
G_Floating	Not supported[1]
H_Floating	Not supported[1]
FLOATING COMPLEX	
F_Floating Complex	Not supported[1]
D_Floating Complex	Not supported[1]
G_Floating Complex	Not supported[1]
H_Floating Complex	Not supported[1]
CHARACTER and DATE	
Character	PIC X(n)
Date	Not supported[2]
NUMERIC	
Unsigned Numeric	PIC S9(m)V9(n)[4]
Left Separate Numeric	PIC S9(m)V9(n)[4] SIGN LEADING SEPARATE

[1]COBOL has no equivalent for this data type. A warning diagnostic is issued for such an item that is part of a record description entry. The compiler treats that item as if it had been specified as an alphanumeric data item with the same number of bytes.

[2]COBOL has no exact equivalent for this data type. A warning diagnostic is issued for such an item that is part of a record description entry. The compiler treats that item as if it had been specified as PIC S9(11)V(7) COMP. (This gives the item units of seconds.)

[3]COBOL has no exact equivalent for this data type. A warning diagnostic is issued for such an item that is part of a record description entry. The compiler treats that item as if it had been specified as corresponding unsigned COMP data type.

[4]The number of digits cannot be greater than 18.

(continued on next page)

Table 7–4 (Cont.) COBOL Data Types

Generic Data Type	COBOL Data Type
NUMERIC	
Right Separate Numeric	PIC S9(m)V9(n)[1] SIGN TRAILING SEPARATE
Left Overpunched Numeric	PIC S9(m)V9(n)[1] SIGN LEADING
Right Overpunched Numeric	PIC S9(m)V9(n)[1] SIGN TRAILING
Zoned Numeric	Not supported[1]
Packed Decimal	PIC S9(m)V9(n) COMP-3[4]

[1]COBOL has no equivalent for this data type. A warning diagnostic is issued for such an item that is part of a record description entry. The compiler treats that item as if it had been specified as an alphanumeric data item with the same number of bytes.

[4]The number of digits cannot be greater than 18.

Table 7–5 C Data Types

Generic Data Type	C Data Type
SIGNED Integer	
Signed Byte	CHAR
Signed Word	SHORT INT
Signed Longword	INT
Signed Quadword	Not supported
Signed Octaword	Not supported
UNSIGNED Integer	
Unsigned Byte	UNSIGNED CHAR
Unsigned Word	UNSIGNED SHORT INT
Unsigned Longword	UNSIGNED INT
Unsigned Quadword	Not supported
Unsigned Octaword	Not supported
FLOATING	
F_Floating	FLOAT
D_Floating	DOUBLE
G_Floating	Not supported
H_Floating	Not supported

(continued on next page)

Table 7–5 (Cont.) C Data Types

Generic Data Type	C Data Type
FLOATING COMPLEX	
F_Floating Complex	Not supported
D_Floating Complex	Not supported
G_Floating Complex	Not supported
H_Floating Complex	Not supported
CHARACTER and DATE	
Character	CHAR[n]
Date	Not supported
NUMERIC	
Unsigned Numeric	Not supported
Left Separate Numeric	Not supported
Right Separate Numeric	Not supported
Left Overpunched Numeric	Not supported
Right Overpunched Numeric	Not supported
Zoned Numeric	Not supported
Packed Decimal	Not supported

Table 7–6 DIBOL Data Types

Generic Data Type	DIBOL Data Type
SIGNED Integer	
Signed Byte	I 1
Signed Word	I 2
Signed Longword	I 4
Signed Quadword	I 8
Signed Octaword	I 16
UNSIGNED Integer	
Unsigned Byte	Not supported
Unsigned Word	Not supported
Unsigned Longword	Not supported
Unsigned Quadword	Not supported
Unsigned Octaword	Not supported

(continued on next page)

Table 7–6 (Cont.) DIBOL Data Types

Generic Data Type	DIBOL Data Type
FLOATING	
F_Floating	Not supported
D_Floating	Not supported
G_Floating	Not supported
H_Floating	Not supported
FLOATING COMPLEX	
F_Floating Complex	Not supported
D_Floating Complex	Not supported
G_Floating Complex	Not supported
H_Floating Complex	Not supported
CHARACTER and DATE	
Character	An[1]
Date	Not supported
NUMERIC	
Unsigned Numeric	Not supported
Left Separate Numeric	Not supported
Right Separate Numeric	Not supported
Left Overpunched Numeric	Not supported
Right Overpunched Numeric	Not supported
Zoned Numeric	Dn[1]
Packed Decimal	Plength.scale[2]

[1]The n is the length in bytes.

[2]Scale is the number of decimal places, such that 1≤length≤31 and 0≤scale≤length.

Table 7–7 FORTRAN Data Types

Generic Data Type	FORTRAN Data Type
SIGNED Integer	
Signed Byte	BYTE
Signed Word	INTEGER*2

(continued on next page)

Table 7–7 (Cont.) FORTRAN Data Types

Generic Data Type	FORTRAN Data Type
SIGNED Integer	
Signed Longword	INTEGER*4 or INTEGER
Signed Quadword	Not supported
Signed Octaword	Not supported
UNSIGNED Integer	
Unsigned Byte	Not supported
Unsigned Word	Not supported
Unsigned Longword	Not supported
Unsigned Quadword	Not supported
Unsigned Octaword	Not supported
FLOATING	
F_Floating	REAL or REAL*4
D_Floating	REAL*8[1]
G_Floating	REAL*8[1]
H_Floating	REAL*16
FLOATING COMPLEX	
F_Floating Complex	COMPLEX or COMPLEX*8
D_Floating Complex	COMPLEX*16[1]
G_Floating Complex	COMPLEX*16[1]
H_Floating Complex	COMPLEX*32
CHARACTER and DATE	
Character	CHARACTER*n[2]
Date	Not supported
NUMERIC	
Unsigned Numeric	Not supported
Left Separate Numeric	Not supported
Right Separate Numeric	Not supported
Left Overpunched Numeric	Not supported
Right Overpunched Numeric	Not supported

[1]A compiler qualifier might be necessary. This data type might not be compatible with another data type if used in the same program. See the *FORTRAN Reference Manual* for more information.

[2]The n $\leq$ 32,767.

(continued on next page)

Table 7–7 (Cont.) FORTRAN Data Types

Generic Data Type	FORTRAN Data Type
NUMERIC	
Zoned Numeric	Not supported
Packed Decimal	Not supported

Table 7–8 MACRO-32 Data Types

Generic Data Type	MACRO-32 Data Type
SIGNED Integer	
Signed Byte	.BLKB n[1]
Signed Word	.BLKB n[1]
Signed Longword	.BLKB n[1]
Signed Quadword	.BLKB n[1]
Signed Octaword	.BLKB n[1]
UNSIGNED Integer	
Unsigned Byte	.BLKB n[1]
Unsigned Word	.BLKB n[1]
Unsigned Longword	.BLKB n[1]
Unsigned Quadword	.BLKB n[1]
Unsigned Octaword	.BLKB n[1]
FLOATING	
F_Floating	.BLKB n[1]
D_Floating	.BLKB n[1]
G_Floating	.BLKB n[1]
H_Floating	.BLKB n[1]
FLOATING COMPLEX	
F_Floating Complex	.BLKB n[1]
D_Floating Complex	.BLKB n[1]
G_Floating Complex	.BLKB n[1]
H_Floating Complex	.BLKB n[1]

[1]The n is the length in bytes.

(continued on next page)

Table 7–8 (Cont.) MACRO-32 Data Types

Generic Data Type	MACRO-32 Data Type
CHARACTER and DATE	
Character	.BLKB n[1]
Date	.BLKB 8
NUMERIC	
Unsigned Numeric	.BLKB n[1]
Left Separate Numeric	.BLKB n[1]
Right Separate Numeric	.BLKB n[1]
Left Overpunched Numeric	.BLKB n[1]
Right Overpunched Numeric	.BLKB n[1]
Zoned Numeric	.BLKB n[1]
Packed Decimal	.BLKB n[1]

[1]The n is the length in bytes.

Table 7–9 Pascal Data Types

Generic Data Type	Pascal Data Type
SIGNED Integer	
Signed Byte	Not supported
Signed Word	Not supported
Signed Longword	INTEGER
Signed Quadword	Not supported
Signed Octaword	Not supported
UNSIGNED Integer	
Unsigned Byte	Not supported
Unsigned Word	Not supported
Unsigned Longword	UNSIGNED
Unsigned Quadword	Not supported
Unsigned Octaword	Not supported
FLOATING	
F_Floating	SINGLE
D_Floating	DOUBLE

(continued on next page)

Table 7–9 (Cont.) Pascal Data Types

Generic Data Type	Pascal Data Type
FLOATING	
G_Floating	Not supported
H_Floating	QUADRUPLE
FLOATING COMPLEX	
F_Floating Complex	Not supported
D_Floating Complex	Not supported
G_Floating Complex	Not supported
H_Floating Complex	Not supported
CHARACTER and DATE	
Character	PACKED ARRAY OF CHAR [1..n][1]
Date	Not supported
NUMERIC	
Unsigned Numeric	Not supported
Left Separate Numeric	Not supported
Right Separate Numeric	Not supported
Left Overpunched Numeric	Not supported
Right Overpunched Numeric	Not supported
Zoned Numeric	Not supported
Packed Decimal	Not supported

[1]The n is the length in bytes.

Table 7–10 PL/I Data Types

Generic Data Type	PL/I Data Type
SIGNED Integer	
Signed Byte	FIXED BINARY (7)
Signed Word	FIXED BINARY (15)
Signed Longword	FIXED or FIXED BINARY (31)
Signed Quadword	Not supported
Signed Octaword	Not supported

(continued on next page)

Table 7–10 (Cont.) PL/I Data Types

Generic Data Type	PL/I Data Type
UNSIGNED Integer	
Unsigned Byte	Not supported
Unsigned Word	Not supported
Unsigned Longword	Not supported
Unsigned Quadword	Not supported
Unsigned Octaword	Not supported
FLOATING	
F_Floating	FLOAT BINARY (24)
D_Floating	FLOAT BINARY (53)
G_Floating	FLOAT BINARY (53)
H_Floating	FLOAT BINARY (113)
FLOATING COMPLEX	
F_Floating Complex	Not supported
D_Floating Complex	Not supported
G_Floating Complex	Not supported
H_Floating Complex	Not supported
CHARACTER and DATE	
Character	CHARACTER (n)[1] CHARACTER VARYING (n)[1]
Date	Not supported
NUMERIC	
Unsigned Numeric	PIC '(m)9'[2]
Left Separate Numeric	PIC 'S(m)9'[2]
Right Separate Numeric	PIC '(m)9S'[2]
Left Overpunched Numeric	PIC 'T(m)9'[2]
Right Overpunched Numeric	PIC '(m)9T'[2]
Zoned Numeric	Not supported
Packed Decimal	FIXED DECIMAL (length,scale)[3]

[1]The n is the number of digits. A compiler qualifier might be necessary for CHARACTER and CHARACTER VARYING. This data type might not be compatible with another data type if used in the same program. See the *PL/I Reference Manual* for more information.

[2]The m is the number of digits.

[3]The scale is the number of decimal spaces such that $1 \leq length \leq 31$ and $0 \leq scale \leq length$.

8
Conditional Expressions

This chapter discusses the use, structure, and evaluation of conditional expressions. Conditional expressions are used as arguments in:

- DDL schema CHECK clauses
- Load/Unload format language WHERE and WHILE clauses
- DML IF tests and WHERE clauses

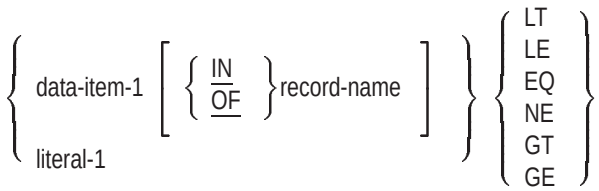
Conditional expressions are evaluated at run time by DBCS. DBCS tests the conditions in the expression and selects alternate control paths depending on whether the condition has a truth value of true or false.

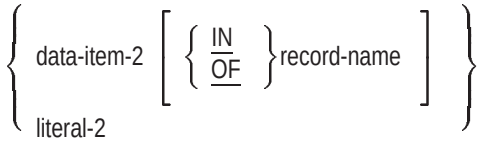
8.1 Anatomy of a Conditional Expression

The conditional expressions are Boolean expressions that consist of a simple relational expression (for example, X is greater than Y) or combinations of such expressions. Because the more complex combinations involve an understanding of relational expressions, the discussion of simple relational expressions precedes the combinations.

8.1.1 Relational Expressions

The relational expression is the core of a conditional expression. It consists of two operands separated by a relational operator. The operands must be schema data items or literal expressions. A literal can be a numeric or nonnumeric character string whose value is specified by the characters it contains. All literals must be contained in quotation marks (" ").





Relational operators specify the type of comparison. These operators are listed in Table 8–1.

Table 8–1 Relational Operators

Relational Operator	Meaning
EQ	Equal to
GE	Greater than or equal to
GT	Greater than
LE	Less than or equal to
LT	Less than
NE	Not equal to

Notes to Table 8–1

- The relational operators must be typed exactly as listed.
- A space must precede and follow each relational operator.

A relational expression is true if the specified relation exists between the operands. For example, the expression X GT Y is true if the value of X is indeed greater than the value of Y.

The following rules apply to relational expressions:

- The relational expression must contain at least one reference to a variable. For example, the following expression contains the variable PART_STATUS:

  ```
  PART_STATUS EQ "N"
  ```

  The following expression, however, is not allowed:

  ```
  5 EQ 5
  ```

- The relational expression can compare operands that have different data classes. For example, if one operand contains alphanumeric data and the other operand contains numeric data, DBCS converts the alphanumeric data to a number for the purposes of the comparison.

- Both data item types, item_name_1 and item_name_2, cannot reference arrays (data item occurrences with an OCCURS clause). If either of the data item types references an array, the array is evaluated as a conjunction of simple conditions. Each occurrence of a data item type in the array, therefore, must be true for the expression to be true.

  For example, the schema defines the data item VEND_ADDRESS (for vendor's address) as occurring three times:

  ```
  VEND_ADDRESS     TYPE IS CHARACTER 15
                   OCCURS 3 TIMES
  ```

  If there was also a CHECK clause on that item, it might specify that VEND_ADDRESS must contain a value:

  ```
  CHECK VEND_ADDRESS NE " "
  ```

  With this CHECK clause each of the three occurrences of VEND_ADDRESS must contain data for a condition check on the data item type to be true.

  In evaluating this condition, DBCS assumes that the conjunction is enclosed in parentheses and treats it as a simple condition.

- You cannot use floating-point COMPLEX data types in a relational expression involving the relational operators LT, LE, GT, or GE. Such data types can be used, however, with EQ or NE.

- You should make sure that a relational expression does not contain:
  - Logically invalid expressions, such as:

    ```
    5 EQ 6
    ```

  - Conditions that are always false or always true, such as:

    ```
    (X LT 0) AND (X GT 0)
    ```

    or

    ```
    (X LT 0) OR (X GT 0) OR (X EQ 0)
    ```

8.1.2 Combinations of Relational Expressions

Complex condition tests are possible using logical operators to link relational expressions or combinations of such expressions.

alternative [OR alternative]...

alternative =

simple-condition [AND simple-condition]...

simple-condition =

{ condition-test
  NOT condition-test
  relational-expression }

Table 8–2 describes the logical operators AND, NOT, and OR.

Table 8–2 Logical Operators

Logical Operator	Meaning
AND	Logical conjunction. The expression is true if both conditions are true. It is false if one or both connected conditions are false.
NOT	Logical negation or reversal of truth value. The expression is true if the following condition is false. It is false if the following condition is true.
OR	Logical inclusive OR. The expression is true if one or both of the connected conditions are true. It is false if both conditions are false.

The following rules apply to a relational expression:

- A condition test specifies one or more alternatives. The truth value of the condition test is true if any of the alternatives is true.
- Each alternative, in turn, consists of one or more simple conditions. The alternative is true if all of the simple conditions are true.
- A simple condition consists of a relational expression or statement of another condition. A conditional expression can be recursive; the statement condition_test refers back to the condition test, the highest level of a conditional expression. The simple condition is true if:
  - The relational expression is true.

- NOT is omitted and the condition test is true.
- NOT is specified and the condition test is false.

8.1.3 Examples of Condition Tests

A relational expression, in itself a condition, consists of two operands separated by a relational operator. For example:

```
X GT Y
```

Such a construction might be used in an item level CHECK clause, such as:

```
PART_PRICE GT "0"
```

Two such expressions might be joined by an OR in a record CHECK clause:

```
PART_PRICE GT "0" OR PART_STATUS NE "N"
```

Similarly, relational expressions can be joined by a logical AND:

```
(PART_PRICE GT "0" AND PART_STATUS NE "N")
```

The operator OR can also join combinations of conditions made up of relational expressions, as in the following:

```
(PART_PRICE GT "0" AND PART_STATUS NE "N")
OR PART_COST LT PART_PRICE
```

8.2 Evaluating Condition Tests

The truth value of a condition depends on:

- The effect of logical operators on the individual truth values of simple conditions
- The intermediate truth values of logically connected or logically negated conditions

8.2.1 Effects of Logical Operators

Table 8–3 depicts the effects of logical operators on two simple conditions. A and B represent the truth values of those two simple conditions.

Table 8–3 Truth Table for Evaluating Conditions

A	B	A AND B	A OR B	NOT A	NOT B
False	False	False	False	True	True
False	True	False	True	True	False
True	False	False	True	False	True
True	True	True	True	False	False

The truth value of a complex condition often depends on whether or not the condition is enclosed in parentheses. For example, the truth value of the following expression changes depending on the placement of parentheses:

```
A OR B AND C
```

Table 8–4 illustrates how this expression would be evaluated if A and B were true and C were false. Note AND is evaluated before OR.

Table 8–4 Truth Table Without Parentheses

A	B	C	Intermediate State B AND C	Truth Value A OR B AND C
True	True	False	False	True

The addition of parentheses, however, changes the truth value, as the following example demonstrates:

```
(A OR B) AND C
```

Table 8–5 illustrates how this expression would be evaluated if A and B were true and C were false. Note OR is evaluated before AND.

Table 8–5 Truth Table with Parentheses

A	B	C	Intermediate State A OR B	Truth Value (A OR B) AND C
True	True	False	True	False

Complex expressions containing the logical operator NOT must be used with parentheses. For example, `NOT (A OR B)` is a valid complex expression; `NOT A OR B` is an invalid complex expression.

Each left parenthesis must have a matching right parenthesis. Each right parenthesis must have a matching left one.

8.2.2 Order of Evaluation

DBCS evaluates a conditional expression in a depth-first manner: it first assesses the condition at its lowest level and proceeds left-to-right, evaluating ANDs before ORs.

DBCS uses the OpenVMS standard comparison routines for comparing text strings. See the OpenVMS documentation for more information on these routines.

9
DBO Utility Commands

This chapter describes individual Database Operator utility (DBO) commands and their parameters and qualifiers. DBO utility command syntax follows the conventions of the OpenVMS command language (DCL). Commands consist of words, generally verbs, that have parameters and qualifiers to define the action to be performed.

Syntax representations and most examples represent interactive processing, not batch processing. During interactive processing, DBO prompts you for required parameters. However, you must include all required parameters in commands entered through command files for batch processing. In addition, the /CONFIRM qualifier is the default on interactive delete or initialization operations whereas /NOCONFIRM is the default on delete or initialize operations during batch processes.

- **Parameters**

  Parameters define the file, Oracle CDD/Repository entity, or database storage areas on which the command will act. In most cases, you can omit the parameter from the command line and enter it in response to a prompt. In a few cases, the command uses a default parameter if you do not specify one.

  One or more spaces separate command parameters and their qualifiers from the command word. Many commands allow multiple instances of parameters separated by commas.

- **Qualifiers**

  Qualifiers always begin with a slash (/) followed by a qualifier word. In some cases, an equal sign (=) and a qualifier argument follow the qualifier word. A qualifier argument can be simple (a value or a keyword) or compound (a string of values or keywords separated by commas with the entire string enclosed in parentheses). Some DBO commands have no qualifiers, while other DBO commands have many qualifiers. Each DBO command description includes a general format that contains a list of valid qualifiers for the command and a corresponding list of default qualifiers, if applicable. Some qualifiers have no defaults; other qualifiers have defaults that depend on context.

  A default value for a qualifier indicates what qualifier you imply if you omit the qualifier completely. Do not assume that defaulting (omitting) a qualifier is the same as specifying it with a default argument. In some cases this is true, but you should regard that as a coincidence for a particular case. The list of qualifiers and defaults shows only command syntax. To understand the action of a qualifier and how it interacts with other qualifiers, you must read the descriptive text for that qualifier.

  For example, the /SUBSCHEMA qualifier is used in several DBO commands and can have:

  – No argument: /SUBSCHEMA

- A simple argument: /SUBSCHEMA=PARTSS1
- A compound argument: /SUBSCHEMA=(PARTSS1,PARTSS3,PARTSS4)
- Negative qualifier: /NOSUBSCHEMA

All DBO qualifiers that accept a list can also accept a command file specification. This is especially useful when you have a long list of arguments. For example, suppose you are specifying a long list of tape labels to the /LABEL qualifier of the DBO/BACKUP/MULTITHREAD command. Instead of typing the labels each time, you can include the labels in an indirect command file and specify the command file following the qualifier. The following example shows this usage:

```
$ DBO/BACKUP/MULTITHREAD/LABEL="@LABEL.EXT" DBMS$USER:[PARTS]PARTS -
_$ $111$MUA2:PARTS
```

The following example shows the contents of a sample indirect command file:

```
TAPE01         ! label for tape 1
TAPE02
TAPE03
<EOF>
```

Note

Do not begin each line in the indirect command file with a dollar sign ($). You can embed comments in the file by preceding the comment with the exclamation mark (!).

Command qualifiers influence the overall action of a command. In most cases, their effect is the same, regardless of their placement in the command string. However, there are some exceptions. For instance, the action qualifiers in the DBO/GRANT_COMMAND and DBO/PERMIT_USER commands must be positioned immediately following the command keyword. Such exceptions are documented in the qualifier descriptions but are not always apparent in the general command format.

Command and area qualifiers can apply either globally to all instances of the parameter or locally to individual area-name parameters. They define the characteristics of the file that contains the area's page.

Parameter qualifiers (also referred to as area qualifiers, depending on the type of parameter they qualify) affect the treatment of parameters in the command. If the command includes multiple instances of a given type of parameter, the placement of parameter qualifiers affects their scope of influence as follows:

- If you position the parameter qualifier after a particular parameter, the qualifier affects only that parameter. In this example, /SCHEMA applies only to TRIALDB:

  ```
  $ DBO/DUMP PARTSDB,TRIALDB/SCHEMA,TESTZ
  ```

  This is local use of a parameter qualifier.

- If you position the parameter qualifier before the first parameter, the qualifier applies to all instances of the parameter. For example, in this DBO/DUMP command, the /SCHEMA qualifier applies to all three root-file-spec parameters:

```
$ DBO/DUMP/SCHEMA PARTSDB,TRIALDB,TESTZ
```

This is global use of a parameter qualifier. Not all parameter qualifiers can be used globally; to identify such qualifiers, read the description of the qualifier.

- Local parameter qualifiers override contradictory global qualifiers in most cases. For example:

```
$ DBO/DUMP/SUBSCHEMA=*/SCHEMA PARTSDB,TRIALDB/SUBSCHEMA=TRYSS
```

Here, the qualifier /SUBSCHEMA=TRYSS, appended locally to the root file specification TRIALDB, overrides the global /SUBSCHEMA=* qualifier. For database PARTSDB, DBO dumps the schema and all subschemas. For database TRIALDB, DBO dumps the schema and the subschema named TRYSS.

- Several DBO utility commands have special rules for positional qualifiers. In such cases, the command description includes these rules, which override the rules presented in this introductory section.

- **Description**

The description section for each command describes the command in detail. It tells you what the command does, its purpose, when and where you might use it, and what operations should be performed before or after the command. Sometimes particular qualifiers are highlighted to help you better understand what the command can do for you.

- **DBO Commands**

A DBO command invokes the Oracle CODASYL DBMS Database Operator (DBO) utility and performs a DBO operation.

A DBO command with a null qualifier invokes an indirect command file that you must specify as a parameter. For example, if you create the command file TEST.COM, containing a DBO/DUMP PARTS command, specify the command file by entering DBO TEST.COM at the DCL prompt. All other DBO commands consist of the DCL keyword DBO, followed immediately by one of the DBO command qualifiers. Each DBO command keyword qualifier has its own set of qualifiers and parameters.

You can include all parameters for any DBO command on the command line. If a DBO command is too long to be entered at the terminal, you can use an indirect command file. When invoking a DBO indirect command file, you must include the indirect command file parameter on the command line; that is, failure to include it is an error. In all other cases, DBO prompts you for any required parameters that are not on the command line and have no default. For some commands, DBO provides (without prompting) a default parameter value if the command line does not include an explicit value.

Some DBO commands control operation of the Oracle CODASYL DBMS system, while others create, maintain, and delete databases. The DBO command qualifiers can be found in Table 9–1.

Table 9–1 DBO Command Qualifiers

/ALTER	/CREATE	/LOAD	/REPORT

(continued on next page)

Table 9–1 (Cont.) DBO Command Qualifiers

/ANALYZE	/DELETE	/MODIFY	/RESOLVE
/AUDIT	/DUMP	/MONITOR	/RESTORE
/BACKUP	/EXPORT	/MOVE_AREA	/SERVER
/CACHE	/EXTRACT	/OPEN	/SET
/CHECKPOINT	/GRANT_COMMAND	/OPTIMIZE	/SHOW
/CLOSE	/INITIALIZE	/PERMIT_USER	/UNLOAD
/CONVERT	/INTEGRATE	/RECLAIM	/VERIFY
/COPY_DATABASE	/LIBRARIAN	/RECOVER	

9.1 DBO Indirect-Command-File

Executes a DBO indirect command file.

Format

DBO dbo-indirect-command-file-spec

Description

A DBO indirect command file, unlike a standard OpenVMS indirect command file, contains only one DBO command. When you enter a DBO command interactively, the size of the DCL command buffer restricts the command length to 256 characters.

For most DBO commands, this length is sufficient. However, some DBO commands (such as DBO/CREATE and DBO/RESTORE) can be very long because of their parameters, each with long strings of positional qualifiers.

You can process a DBO command containing up to 1024 characters and 128 tokens by placing it in a special formatted DBO indirect command file. The file can contain no other command. The overall format of the indirect command file is:

DBO/dbo-command line-1 - line-2 - ...line-n

The DBO command in the file must not begin with a dollar sign ($). You must use a hyphen, the continuation character, at the end of every line except the last.

Do not try to execute a DBO indirect command file as an OpenVMS indirect command file. If you reach the character or token limit, try the following:

- Abbreviate qualifiers to four letters. Shorter abbreviations are possible in many cases but might be incompatible with future releases.
- On DBO/CREATE or DBO/RESTORE commands, use defaults on those qualifiers that you can later adjust with DBO/MODIFY commands.
- Use the global /FILE qualifier to specify device and directory. Use local /FILE qualifiers only for those area files for which you do not want the global specification to apply.

Parameter

dbo-indirect-command-file-spec
Specifies the DBO-format indirect command file to be executed. This type of indirect command file is DBO specific; it is not the same as an OpenVMS indirect command file that you use in a batch job. Do not precede the file specification with an at sign (@). The default file type is .COM.

9.2 DBO/ALTER Command

Invokes the interactive ALTER utility, used to examine or modify selected parts of database files when you suspect corruption.

Format

```
DBO/ALTER  [ root-file-spec ]
```

Description

DBO/ALTER invokes DBALTER, a low-level patch utility you can use to:

- Bind DBALTER to a database for alteration
- Fetch a page from a storage area
- Display information from a page
- Change data and structural information on a page
- Change database file specifications to establish a database on a new device or device set
- Specify numeric radix conversion for new data (default is decimal)
- Move data from one part of a page to another part
- Have a page verified for data integrity
- Write pages, with or without changes, back to the database
- Reset the corruption indication bit
- Tell DBALTER to ignore (roll back) changes that have not been committed
- Commit changes to the database
- Unbind from a database

When you type DBO/ALTER, DBO displays the following prompt:

```
DBALTER>
```

The DBALTER> prompt means you are at DBALTER command level. You cannot enter DCL-level commands until you exit DBALTER.

Chapter 11 describes the DBALTER commands. See the *Oracle CODASYL DBMS Database Maintenance and Performance Guide* for a detailed tutorial introduction to the DBALTER utility.

Parameter

root-file-spec
Specifies the root file of a database containing the storage areas you intend to alter. The default file type is .ROO.

You can omit this parameter and then use the DBALTER BIND command to specify the database you want to alter.

9.3 DBO/ANALYZE Command

Displays space usage, record, and set statistics for a database.

Format

DBO/ANALYZE root-file-spec

Command Qualifiers	Defaults
/AREAS=area-name [,...]	/AREAS=*
/BINARY_FILE=file-spec	
/BUFFERS=integer	/BUFFERS=root-value
/END=integer	/END=last-page
/INTERVAL_LOG=integer	/INTERVAL_LOG=50
/LENGTH_BUFFER=integer	/LENGTH_BUFFER=root-value
/[NO]LOG	Current DCL verify value
/[NO]ONLINE	/NOONLINE
/OUTPUT=file-spec	SYS$OUTPUT
/[NO]PAGE_USAGE	/NOPAGE_USAGE
/[NO]SETS[=set-name [,...]]	/SETS
/START=integer	/START=1

Description

DBO/ANALYZE provides a maintenance tool for database administration. You can use it to:

- Show how many database pages are relatively full and how many are relatively empty
- Show space usage for a storage area
- Show record storage statistics
- Analyze storage sets by means of:
  - a table showing statistics for all the sets in an area
  - a histogram for each set type showing the number of members for each set occurrence
  - a histogram for each set type showing the number of disk reads necessary to traverse each set occurrence
- Experiment with various numbers of buffers and buffer lengths to determine the effect on disk read statistics
- Direct command output to a file

A histogram (bar graph) shows page space usage statistics for each area you request. Record space usage statistics are also shown automatically. Set space usage statistics are shown unless you specify the /NOSETS qualifier. When you request set statistics, you can experiment with different buffer parameters for set cluster read analysis.

By default, DBO/ANALYZE runs in PROTECTED RETRIEVAL mode and read-only areas default to BATCH RETRIEVAL mode. If you are not doing set analysis, it runs in CONCURRENT RETRIEVAL mode.

Parameter

root-file-spec
Specifies the root file of a database to be analyzed. The default file type is .ROO.

Command Qualifiers

/AREAS=area-name [,...]
Specifies one or more database storage areas to analyze. If you specify an area name for which there is no schema area clause, an error occurs.

The default, /AREAS=*, analyzes all storage areas for the database.

/BINARY_FILE=file=spec
Produces a formatted output file containing data collected during the analyze session. This data file can be parsed by a user-written application program to extract information of interest.

The output file consists of a header record plus one data record for each database area, set, or record analyzed. For example, a full DBO/ANALYZE of the PARTS database produces the header record followed by 4 AREA records, 13 SET records, and 8 RECORD records in the output file.

The record format representing each entity type is different and can be distinguished according to the first field of the record, a longword integer, as described in Table 9–2.

Table 9–2 Record Format Identifiers

Value	Record Format
0	Header Record
1	Area-description Record
2	Record-description Record
3	Set-description Record

Use the /BINARY_FILE qualifier in conjunction with the /AREA and /[NO]SETS qualifiers to limit output.

The layout for each record format can be found in an Oracle CDD/Repository record definition file: DBOANA$ANALYZE_BINARY_FORMAT.CDO. This file, as well as a sample application program, DBOANA$DUMP_BINARY.C can be found in the IVP directory SYS$COMMON:[SYSTEST.DBM].

/BUFFERS=integer
Specifies a hypothetical number of page buffers for the DBO/ANALYZE command to use when walking through sets. You can use this qualifier to see how different buffer counts would influence set cluster read statistics. If you specify fewer buffers than you already have, the cluster read statistics of the set name tables tend to be higher than before, and the cluster read histograms tend to show different distributions than before. All other statistics are unaffected by specifying fewer buffers.

The default is the integer set by DBO/CREATE or the last DBO/MODIFY command for the database.

This qualifier's sole purpose is to test and determine the effect of a hypothetical number of buffers on set read performance. This qualifier does not affect the number of buffers in the database root file or the number of buffers used by the DBO/ANALYZE command.

/END=integer
Specifies the range of database pages within each area to be analyzed. The /END qualifier specifies the end of the page range. By default, a database analysis starts with the first page of the database and ends with the last page of the database. You can override the default using the /START and /END qualifiers to specify the page range.

/INTERVAL_LOG=integer
Specifies how frequently the log message, DBO-I-LOGANAPAG, should be issued during analysis. The default is every 50 database pages.

/LENGTH_BUFFER=integer
Specifies the hypothetical number of disk blocks per database page buffer for the DBO/ANALYZE command to use when doing set walking analysis. You can use this qualifier to see how different buffer lengths would influence set cluster read statistics. If you specify shorter buffers than you already have, the cluster read statistics of the set name tables tend to be higher, and the cluster read histograms tend to show different distributions than with longer buffers. All other statistics are unaffected.

The default is the value set by DBO/CREATE or the last DBO/MODIFY command for the database.

This qualifier's sole purpose is to test and determine the effect of a hypothetical buffer length on set read performance. This qualifier does not affect the length of buffers in the database root file or the length of buffers used by the DBO/ANALYZE command.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request log output and /NOLOG to prevent it. The default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.) If the DCL verify option is set, the /INTERVAL_LOG qualifier can be used without specifying /LOG. If the DCL verify option is not set, an error is returned.

The /LOG qualifier allows you to monitor the progress of the database analysis. This should be especially useful on large databases.

The log message, DBO-I-BEGINANA, is issued at the beginning of analysis. The log message, DBO-I-LOGANAPAG, is issued during analysis at the specified interval. The default interval is 50 pages and can be changed with the /INTERVAL_LOG qualifier. The log message, DBO-I-ENDANA, is issued at the end of analysis.

/ONLINE
/NOONLINE
Allows database analysis without the need to shut down the database. Because online analysis uses snapshots to create a consistent view of the database, snapshots must be allowed and enabled for all storage areas being analyzed. See DBO/MODIFY and DBO/RESTORE for information on enabling snapshots, and DBO/CREATE for information on allowing snapshots. When you specify the /ONLINE qualifier, users can continue to update the database during the procedure. The online analysis procedure uses the snapshot file to retrieve the previous version of any record that has been modified since the analysis began.

By specifying /ONLINE, you can prevent the loss of time involved in waiting for write operations, thus improving performance.

If you use /NOONLINE, users cannot be bound to the database for updates. To use /NOONLINE, you do not need to allow or enable snapshots.

The default is /NOONLINE.

/OUTPUT=file-spec
Names the file where output will be sent. The default file type is .LIS. The default output is SYS$OUTPUT.

/PAGE_USAGE
/NOPAGE_USAGE
When analyzing a database, you can determine the fullness of database pages through an additional diagram generated by the /PAGE_USAGE qualifier. The diagram gives you a good idea of how well the record clustering is working in the database. For every database page analyzed, a line is written to the output file, causing DBO/ANALYZE to run slower. For optimal performance, use the /START and /END qualifiers to specify a range of database pages. The default is /NOPAGE_USAGE.

/SETS[=set-name[,...]]
/NOSETS
Specify /SETS if you want DBO to analyze all your database storage sets. Specify one or more storage sets if you want DBO to analyze specific storage sets. If you specify /NOSETS, pages and records are analyzed, but storage sets are not analyzed. The default is /SETS.

/START=integer
When analyzing a database, you can specify the range of database pages to be analyzed. The /START qualifier specifies the start of the page range. By default, database analysis starts with the first page of the database and ends with the last page of the database. You can override the default using the /START and /END qualifiers to specify the page range.

Examples

1. `$ DBO/ANALYZE/AREA=BUY/NOSETS PARTS`

   This command produces a page usage histogram and a record type summary page. The analysis provides data on the size, length, and fragmentation of the records within the storage area BUY.

2. `$ DBO/ANALYZE/AREA=BUY PARTS`

   This command produces a page usage histogram and a record type summary page. The command also produces a set type summary for the storage area, an index summary, and histograms showing ranges of members per occurrence and disk reads per occurrence for each set in the storage area BUY.

3. `$ DBO/ANALYZE/BUFFER=10/LENGTH_BUFFER=10/AREA=BUY PARTS`

   This command sets the hypothetical number of buffers to 10 and the hypothetical length of each buffer to 10 for the area BUY of the PARTS database. If the area BUY has 10 buffers of 10 pages each, the reduced buffering tends to increase the number of clustered read operations needed to read some of the sets. If you specify larger buffering parameters (such as /BUFFER=20 and /LENGTH_BUFFER=16), the clustered read statistics become smaller.

9.4 DBO/AUDIT Commands

There are three forms of the DBO/AUDIT command, each having its own qualifiers and parameters. These are:

DBO/AUDIT/ANALYZE root-file-spec

This format extracts Oracle CODASYL DBMS audit records from the OpenVMS security audit log file for the specified database.

DBO/AUDIT/LIST root-file-spec [object-name,...]

This format displays current auditing characteristics for a database.

DBO/AUDIT/SET/OBJECT=option

This format establishes or changes discretionary auditing characteristics. You can specify the following options for the /OBJECT qualifier:

- COMMAND root-file-spec [command-type[,...]]

  Specifies securable DBO commands for the specified database.

- DATABASE root-file-spec

  Sets and modifies auditing characteristics for a database. The root-file-spec parameter specifies the name of the database whose auditing characteristics you want to change. The default file type is .ROO.

- SECURITY_SCHEMA root-file-spec security-schema-name[,...]

  Establishes or changes discretionary auditing characteristics for security schemas of the specified database.

These formats are described in the following sections.

9.4.1 DBO/AUDIT/ANALYZE Command

Extracts audit records generated by Oracle CODASYL DBMS.

Format

DBO/AUDIT/ANALYZE root-file-spec

Command Qualifiers	Defaults
/BEFORE[=time]	
/BINARY	See description
/BRIEF	
/FULL	
/IGNORE=(criteria[,...])	
/JOURNAL[=file-spec[,...]]	See description
/OUTPUT[=file-spec]	SYS$OUTPUT
/SELECT=(criteria[,...])	See description
/SINCE[=time]	See description
/SUMMARY	

Description

This command is the Oracle CODASYL DBMS equivalent of the OpenVMS DCL ANALYZE/AUDIT command.

Parameter

root-file-spec
Specifies the root file name of the database from which you want to extract audit records.

Command Qualifiers

/BEFORE[=time]
Specifies the time used to select audit records. Audit records dated earlier than the specified time are selected. You can specify an absolute time, delta time, or a combination of the two. Observe the syntax rules for date and time described in the OpenVMS user documentation. If you specify /BEFORE without specifying the time, midnight of the previous day is used.

By default, all audit records in the OpenVMS security audit log file are examined. You must specify /BEFORE to discard audit records created after a specific time.

/BINARY
When /BINARY is specified, image copies of the selected audit records are written to the output file specified in the /OUTPUT qualifier. Each audit record consists of a number of data packets. The first is known as a header packet and describes the rest of the data packets. The number of data packets depends on the event type. Audit reduction and reporting tools can be written to process the binary output file. See the *Oracle CODASYL DBMS Database Security Guide* for a description of the binary audit record formats.

If you omit the qualifier, ASCII records are written to the output file. If you specify /BINARY and omit the /OUTPUT qualifier, the selected audit records are written to an output file named DBMS.AUDIT$JOURNAL. You cannot use the /BINARY and /FULL qualifiers together.

/BRIEF
Displays selected audit records in an abbreviated format. If you do not specify /FULL, this is the default.

/FULL
Controls whether or not a full format is used in ASCII displays. If you omit the qualifier, audit records are displayed in the brief format.

You cannot use the /BINARY and /FULL qualifiers together.

/IGNORE=(criteria[,...])
Specifies criteria used to ignore audit records from the OpenVMS security audit log file. For a list of criteria, see the /SELECT qualifier description.

/JOURNAL[=file-spec[,...]]
Specifies one or more OpenVMS security audit log files as input to be processed by the DBO/AUDIT/ANALYZE command. If you omit the /JOURNAL qualifier, data is processed from the default OpenVMS security audit log file SYS$MANAGER:SECURITY_AUDIT.AUDIT$JOURNAL. You must set the default to SYS$MANAGER if you do not specify a full file-spec for journal.

/OUTPUT[=file-spec]
Specifies the file to which the output is written. If you omit the qualifier, selected ASCII records are written to SYS$OUTPUT. If you omit the device and directory specification, the current device and directory specification is used. If you omit the file name and type, the default file name AUDIT.LIS is used. If the output is binary (/BINARY) and you omit the /OUTPUT qualifier, the binary information is written to the file DBMS.AUDIT$JOURNAL.

/SELECT=(criteria[,...])
Specifies the criteria to be used to select audit records. If you omit the /SELECT qualifier, all audit records for the specified database are selected.

The criteria list for the /IGNORE and /SELECT qualifiers is as follows:

- COMMAND=(option[,...])

  Specifies one or more DBO commands as selection criteria. The valid command names are:

  - ALTER
  - ANALYZE
  - AUDIT
  - BACKUP
  - CLOSE
  - COPY_DATABASE
  - DELETE
  - DUMP
  - EXPORT
  - GRANT_COMMAND
  - INITIALIZE
  - INTEGRATE

- LOAD
- MODIFY
- MOVE_AREA
- OPEN
- OPTIMIZE
- PERMIT_USER
- RECOVER
- RESOLVE
- RESTORE
- UNLOAD
- VERIFY

- EVENT_TYPE=(event_type[,...])

  Specifies one or more event types as selection criteria for audit records from the OpenVMS security audit log file. The valid event types are:

  - AUDIT

    Auditing of audit changes

  - DACCESS

    Auditing of object access requiring discretionary access

  - DBO

    Auditing of securable DBO commands

  - PROTECTION

    Auditing of permission modifications

  See the *Oracle CODASYL DBMS Database Security Guide* for more information on these event types.

- NAME_OBJECT=(object-name[,...])

  Specifies one or more instances of a record name or a set name as the selection criteria from the OpenVMS security audit log file. Use the following syntax:

```
$ DBO/AUDIT/ANALYZE/SELECT=(NAME_OBJECT=(MAKE,BUY), -
_$ TYPE_OBJECT=(AREA,RECORD)) PARTS
```

- NODENAME=(nodename [,...])

  Only audit records from the specified node or list of nodes are included in the output file generated.

- SEC_SUBCODE=(option[,...])

  Specifies a security completion subcode as a selection criteria. The valid security subcodes are:

  - DBMS_PRIVILEGE

    Access was granted by an Oracle CODASYL DBMS privilege (security schema access or DBO command CAL).

  - VMS_BYPASS

    Access was granted by means of an OpenVMS BYPASS privilege.

  - VMS_SECURITY

    Access was granted by means of an OpenVMS SECURITY privilege.

- STATUS=(status[,...])

  Specifies one or both of the security event completion status values. The valid status values are:

  - CODE=value

    Security event completion status values

  - SUCCESS
  - FAILURE

- TERMINAL=(terminal [,...])

  Specifies a terminal name or a list of terminal names as selection criteria from the OpenVMS security audit log file.

- TYPE_OBJECT=(object-type[,...])

  Specifies one or more object-types as the selection criteria from the OpenVMS security audit log file. The following Oracle CODASYL DBMS object types are supported:

  - AREA
  - DATABASE
  - ITEM
  - RECORD
  - SET

  You can specify more than one object clause within the /OBJECT qualifier. Use the following syntax:

```
$ DBO/AUDIT/ANALYZE/SELECT=(TYPE_OBJECT=AREA=(MAKE,BUY), -
_$ TYPE_OBJECT=RECORD=*)
```

- USERNAME=(username [,...])

  Specifies a particular username or list of usernames as the selection criteria. Only those audit records associated with the specified username are extracted from the security audit log file.

- VERB=(option[,...])

  Specifies one or more DML verbs as selection criteria from the OpenVMS security audit log file. The valid verb names, which can be listed, are separated by commas. They are:

  - BIND
  - DISTRIBUTED_TRANSACTION
  - ERASE
  - FIND
  - GET
  - IF
  - MODIFY
  - STORE

/SINCE[=time]
Specifies the time used to select audit records. Audit records with the same date or later than the specified time are selected. You can specify an absolute time, delta time, or a combination of the two. Observe the syntax rules for date and time described in the OpenVMS user documentation.

If you specify /SINCE without specifying the time, midnight of the current day is used.

/SUMMARY
Specifies that a summary of the selected audit records be produced after all audit records are processed.

You can use the /SUMMARY qualifier alone or in combination with the /BRIEF, /BINARY, or /FULL qualifier.

Examples

1.
```
$ DBO/AUDIT/ANALYZE/SUMMARY PARTS
Total records read:    511         Records Selected:   511
Audit Changes:         3           Protection Changes: 1
Utility Command Count: 15          Object Accesses:    453
Access Successes:      450         Access Failures:    3
```

This command shows a summary listing of all database-related audit records for the PARTS database.

2.
```
$ DBO/AUDIT/ANALYZE PARTS

Date/Time              Type    Subtype Node Username ID     Status
-------------------------------------------------------------------
1-SEP-1991 16:00:02.37 AUDIT          WOOBY SYSTEM 20800AE4 00000001
1-SEP-1991 16:00:03.47 PROTECT PERMIT WOOBY SYSTEM 20800AE4 0028900C
1-SEP-1991 16:00:08.82 DACCESS AREA   WOOBY FLYNN  20800AB2 0028900C
1-SEP-1991 16:00:10.45 DACCESS RECORD WOOBY FLYNN  20800AB2 0028900C
1-SEP-1991 16:05:34.76 DBO     VERIFY WOOBY SYSTEM 20800AE4 00000001
```

This example shows a sample brief listing from the DBO/AUDIT/ANALYZE command.

3.
```
$ DBO/AUDIT/ANALYZE/FULL PARTS
Security audit on WOOBY, system id: 19681
Database name:    DBMS$DISK:[SMITH]PARTS.ROO;1
Auditable event:  DACCESS check
Event time:       1-SEP-1991 16:00:24.22
PID:              47202996
Terminal:         TWA101:
Username:         WOOBY
Process name:     SMITH
UIC:              [DBMS,SMITH]
Final status:    %DBM-F-SECURVIO, security violation, operation denied
Security subcode:
Object type:      DATABASE
Object name:      DBO Command
Operation code:   DBO/PERMIT_USER
TSN:              0
```

This example shows a single audit record from a full OpenVMS security audit log file.

```
4.  $ DBO/AUDIT/ANALYZE PARTS/OUTPUT=AUDIT.DAT/BINARY -
    _$ /SELECT=(STATUS=FAILURE,USERNAME=SMITH) -
    _$ /SINCE=YESTERDAY/OUTPUT=AUDIT.DAT
```

In this example, the DBO/AUDIT/ANALYZE/BINARY command is used to create a binary file of selected audit record images from the OpenVMS security audit log file. All audit records generated by user SMITH with a failure status that have been generated since midnight yesterday are placed in the file named AUDIT.DAT. A user-written program can read and process the binary file, generating a report from the file.

9.4.2 DBO/AUDIT/LIST Command

Displays current auditing characteristics for a database.

Format

DBO/AUDIT/LIST root-file-spec [object-name,...]

Command Qualifiers	Default
/OBJECT[=option]	/OBJECT=DATABASE
/OUTPUT=file-spec	SYS$OUTPUT

Description

This command displays auditing characteristics of the database, securable DBO commands, and security schemas.

Parameters

root-file-spec
Specifies the root file name for which you want to display security audit characteristics. The default file type is .ROO.

object-name,...
If the /OBJECT qualifier has COMMAND as its value, a list of DBO commands can be given for which auditing characteristics should be displayed. If the /OBJECT qualifier has SECURITY_SCHEMA as its value, a list of security schemas can be given for which auditing characteristics should be displayed. If the /OBJECT qualifier has DATABASE as its value, no object parameters can be listed on the command line. If no object-name parameter is specified, security characteristics for all securable DBO commands and security schemas are displayed depending on the value of the /OBJECT qualifier.

Command Qualifiers

/OBJECT[=option,...]
Specifies the object of the list command. The valid options are:

- COMMAND

  Displays security audit information for securable DBO commands.

- DATABASE

  Displays security audit information at the database level. This is the default.

- SECURITY_SCHEMA

Displays security audit information for security schemas. If one or more security schema names are specified, only those specified names are listed.

If the /OBJECT qualifier contains no value or is omitted, the default behavior displays the security audit information for the database.

/OUTPUT=file-spec
Specifies the file that will receive the output. The default file type is .LIS. The default output is SYS$OUTPUT.

Examples

1.
```
$ DBO/AUDIT/LIST PARTS
Security characteristics for: DBMS$DISK:[SMITH]PARTS.ROO;1

   Security auditing STOPPED for:
       PROTECTION (disabled)
       DBO (disabled)
       AUDIT (enabled)
       DACCESS (disabled)

   Security alarms STARTED for:
       PROTECTION (enabled)
       DBO (enabled)
       AUDIT (enabled)
       DACCESS (disabled)

   Audit flush is disabled
```

This example shows the use of the DBO/AUDIT/LIST command to display the current auditing characteristics for the PARTS database.

2.
```
$ DBO/AUDIT/LIST/OBJECT=COMMAND PARTS
%DBO-I-LOGCALAUD, DBO/ALTER auditing SUCCESS,FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/ANALYZE auditing FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/BACKUP auditing FAILURE by ALARM
%DBO-I-LOGCALAUD, DBO/CLOSE auditing FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/DELETE auditing SUCCESS,FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/DUMP auditing FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/GRANT_COMMAND auditing SUCCESS,FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/INITIALIZE auditing SUCCESS,FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/LOAD auditing FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/MODIFY auditing SUCCESS,FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/OPEN auditing FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/PERMIT_USER auditing SUCCESS,FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/RECOVER auditing FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/RESTORE auditing FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/UNLOAD auditing FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/VERIFY auditing is off
%DBO-I-LOGCALAUD, DBO/INTEGRATE auditing FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/RESOLVE auditing FAILURE by AUDIT
%DBO-I-LOGCALAUD, DBO/AUDIT auditing SUCCESS,FAILURE by AUDIT
```

This example displays the discretionary access (DACCESS) security characteristics of all securable DBO commands in the parts database. The success or failure of a DBO/AUDIT command will generate an audit report. The failure of a DBO/BACKUP command will generate an alarm message. The success or failure of a DBO/DELETE command will generate an audit report and an alarm message.

```
3.  $ DBO/AUDIT/LIST/OBJECT=SECURITY_SCHEMA PARTS READ_ONLY
    %DBO-I-LOGSECAUD, security schema READ_ONLY auditing FAILURE by AUDIT,ALARM
```

This example displays the current auditing characteristics of the READ_ONLY security schema in the PARTS database.

9.4.3 DBO/AUDIT/SET/OBJECT=COMMAND Command

Establishes or changes discretionary access auditing characteristics for securable DBO commands.

Format

DBO/AUDIT/SET/OBJECT=COMMAND root-file-spec [command-type[,...]]

Command Qualifiers	Defaults
/LOG	
/STATUS=(option[,...])	See description
/TYPE=(option[,...])	See description

Description

This command allows you to set or change discretionary event-type auditing characteristics for securable DBO commands. The auditing characteristics define the condition under which DACCESS event-type auditing can take place for a given securable DBO command. For auditing to take place, the DACCESS event type must be enabled, auditing must be started for the database, and the /STATUS and /TYPE qualifiers must be specified.

Parameters

root-file-spec
Specifies the root file name for which you want to change security audit characteristics. The default file type is .ROO.

command-type[,...]
Specifies one or more securable DBO command names for which you want to change the auditing characteristics.

Command Qualifiers

/LOG
Indicates that the processing of the DBO/AUDIT/SET command is to be reported to SYS$OUTPUT.

/STATUS=(option[,...])
Specifies whether DACCESS event-type auditing should be performed when users of the specified command(s) are granted access to the command, denied access to the command, or both. If the /STATUS qualifier is not specified, auditing characteristics for the specified command are turned off.

The following keywords can be used:

- SUCCESS

Generates a DACCESS security event if the user is successfully granted access to the specified command(s). (If audit reporting is enabled and started.)

- FAILURE

  Generates a DACCESS security event if the user is denied access to the specified command(s). (If audit reporting is enabled and started.)

/TYPE=(option[,...])
Specifies whether DACCESS auditing for the specified commands should generate an audit record, alarm message, or both. If the /TYPE qualifier is not specified, auditing characteristics for the specified commands are turned off.

The following are valid options:

- ALARM

  Sends security alarm messages to all terminals enabled as security operator terminals.

- AUDIT

  Records security audit records in the OpenVMS security audit log file.

Example

```
$ DBO/AUDIT/SET/OBJECT=COMMAND/LOG -
_$  /TYPE=AUDIT/STATUS=(SUCCESS,FAILURE) PARTS AUDIT,GRANT_COMMAND
%DBO-I-MODCALAUD, modified audit characteristics for
  DBO/AUDIT command
%DBO-I-MODCALAUD, modified audit characteristics for
  DBO/GRANT_COMMAND command
```

This example shows the use of the DBO/AUDIT/SET/OBJECT=COMMAND command. Security auditing characteristics are established for the DBO/AUDIT and DBO/GRANT_COMMAND commands. Whether the user is granted or denied access, an audit record will be generated.

9.4.4 DBO/AUDIT/SET/OBJECT=DATABASE Command

Modifies auditing characteristics of a database.

Format

DBO/AUDIT/SET/OBJECT=DATABASE root-file-spec

Command Qualifiers	Defaults
/DISABLE[=(option[,...])]	
/ENABLE[=(option[,...])]	
/[NO]FLUSH	See description
/LOG	
/START	
/STOP	
/TYPE[=(option[,...])]	/TYPE=(ALARM, AUDIT)

Description

Alters the auditing characteristics of a database. This command is the Oracle CODASYL DBMS equivalent of the OpenVMS DCL SET AUDIT command. Because Oracle CODASYL DBMS security auditing uses OpenVMS system-level auditing mechanisms, certain auditing characteristics (for example, /FAILURE_MODE) can only be set and modified using the OpenVMS SET AUDIT command, which requires OpenVMS SECURITY privilege. Because both /SET and /OBJECT=DATABASE are defaults for the DBO/AUDIT command, you can specify DBO/AUDIT/SET/OBJECT=DATABASE simply by specifying DBO/AUDIT. For example, you could use the following command to start auditing on the PARTS database:

```
DBO/AUDIT/START PARTS
```

Note

The DBO/AUDIT/SET/OBJECT=DATABASE command can be invoked while users are bound to the database. You can start and stop auditing or enable and disable events while users are bound to the database. However, you must have exclusive access to the root file to invoke the DBO/AUDIT command with an object type of COMMAND or SECURITY_SCHEMA.

Parameter

root-file-spec
Specifies the root file of the database whose auditing characteristics you want to change. The default file type is .ROO.

Command Qualifiers

/DISABLE[=(option[,...])]
Specifies one or more event types for which you want to disable security auditing. For a list of options, see the /ENABLE qualifier description. If you specify the /DISABLE qualifier without listing any options, all event types are disabled.

When processing the DBO/AUDIT command, Oracle CODASYL DBMS processes the /DISABLE qualifier last. If you accidentally specify both /ENABLE and /DISABLE for the same event type in the same command, the /DISABLE qualifier takes precedence.

/ENABLE[=(option[,...])]
Enables security auditing. Specifies one or more event types you want to enable. To enable alarm messages or audit records for all event types, specify the keyword ALL. You can also specify the appropriate keywords to selectively enable alarm messages or audit records for event types that are currently disabled. If you do not specify any keywords, all event types are enabled.

In processing the DBO/AUDIT command, Oracle CODASYL DBMS processes the /DISABLE qualifier last. If you accidentally specify both /ENABLE and /DISABLE for the same event type in the same command, the /DISABLE qualifier takes precedence.

The possible event types that may be specified in the keyword list for both the /ENABLE and /DISABLE qualifiers are as follows:

- ALL

  All possible event types except AUDIT cannot be disabled.

- AUDIT

  Auditing of audit changes. AUDIT cannot be disabled.

- DACCESS

  Auditing of object access requiring permission.

- DBO

  Auditing of securable DBO commands.

- PROTECTION

  Auditing of permission modifications.

/FLUSH
/NOFLUSH
Indicates whether or not forced writes of audit records are currently enabled for the database. Forced writes cause Oracle CODASYL DBMS to flush the audit record immediately out to disk at the time the audit record is produced (as opposed to waiting for the AUDIT_SERVER to flush the audit records at a rate defined using the OpenVMS SET AUDIT command).

/LOG
Indicates that the processing of the DBO/AUDIT/SET command is to be reported to SYS$OUTPUT.

/START
Starts security auditing for a database.

When the /START qualifier is specified, auditing starts immediately for all event types that are currently enabled. Any subsequent security events result as records in the OpenVMS security audit log file or alarm messages sent to security-enabled terminals, or both, depending on what you have specified for your database.

/STOP
Stops security auditing for a database. The /STOP qualifier by itself stops both alarm messages and audit records. You can also supply the /TYPE=ALARM or /TYPE=AUDIT qualifier to stop only alarm messages or audit reports.

When the /STOP qualifier is specified, alarm messages and audit records for all event-type classes are immediately stopped. The audit events remain enabled, and you can start them again by using the /START qualifier.

/TYPE=(option[,...])
Causes other qualifiers in the command line (/START, /STOP, /ENABLE, /DISABLE) to affect different auditing modes, based on the following options:

- ALARM

  Sends security alarm messages to all terminals enabled as security operator terminals.

- AUDIT

Records security audit records in the OpenVMS security audit log file.

If you omit the /TYPE qualifier, the default action performs both types of audit reporting.

Example

```
$ DBO/AUDIT/SET/TYPE=ALARM/ENABLE=(DBO,PROTECTION)/LOG PARTS
%DBO-I-LOGMODVAL,     modified security audit event class flags
$ !
$ ! Show that alarms are enabled, but not yet started
$  DBO/AUDIT/LIST PARTS.ROO
Security characteristics for: DBMS$DISK:[SMITH]PARTS.ROO;1

   Security auditing STOPPED for:
       PROTECTION (disabled)
       DBO (disabled)
       AUDIT (enabled)
       DACCESS (disabled)

   Security alarms STOPPED for:
       PROTECTION (enabled)
       DBO (enabled)
       AUDIT (enabled)
       DACCESS (disabled)

   Audit flush is disabled

$ ! Start alarms for the enabled DBO and PROTECTION classes:
$ DBO/AUDIT/SET/START/TYPE=ALARM/LOG PARTS
%DBO-I-LOGMODFLG, enabled security alarm
$ !
$ ! Show that alarms are started for the DBO
$ ! and protection classes
$ DBO/AUDIT/LIST PARTS
Security characteristics for: DBMS$DISK:[SMITH]PARTS.ROO;1

   Security auditing STOPPED for:
       PROTECTION (disabled)
       DBO (disabled)
       AUDIT (enabled)
       DACCESS (disabled)

   Security alarms STARTED for:
       PROTECTION (enabled)
       DBO (enabled)
       AUDIT (enabled)
       DACCESS (disabled)

   Audit flush is disabled
```

This example shows the use of the DBO/AUDIT command. The default for DBO/AUDIT is /SET/OBJECT=DATABASE. Alarm messages are defined for commands that modify database object protection and use securable DBO commands.

9.4.5 DBO/AUDIT/SET/OBJECT=SECURITY_SCHEMA Command

Establishes or changes discretionary auditing characteristics for security schemas.

Format

DBO/AUDIT/SET/OBJECT=SECURITY_SCHEMA root-file-spec security-schema-name[,...]

Command Qualifiers	Defaults
/LOG	
/STATUS=(option[,...])	See description
/TYPE=(option[,...])	See description

Description

This command is used to modify the discretionary access checking auditing characteristics for access through security schemas.

Parameters

root-file-spec
Specifies the root file name for which you want to display security audit characteristics.

security-schema-name[,...]
Specifies one or more security schemas for which you want to initially set or to change the DACCESS event-type security auditing characteristics.

Command Qualifiers

/LOG
Indicates that the processing of the DBO/AUDIT/SET command is to be reported to SYS$OUTPUT.

/STATUS=(option[,...])
Specifies whether DACCESS event-type auditing is performed when users of the specified command(s) are granted access to the command, denied access to the command, or both. If the /STATUS qualifier is not specified, auditing characteristics for the specified security schema are turned off.

The following keywords may be used:

- SUCCESS

  If audit reporting is enabled and started, generates a DACCESS security event if the user is successfully granted access to the specified command(s).

- FAILURE

  If audit reporting is enabled and started, generates a DACCESS security event if the user is denied access to the specified command(s).

/TYPE=(option[,...])
Specifies whether DACCESS auditing for the specified commands generates an audit record, alarm message, or both. If the /TYPE qualifier is not specified, auditing characteristics for the specified security schema are turned off.

The following are valid options:

- ALARM

  Sends security alarm messages to all terminals enabled as security operator terminals.

- AUDIT

Records security audit records in the OpenVMS security audit log file.

Example

```
$ DBO/AUDIT/SET/OBJECT=SECURITY_SCHEMA/LOG -
_$  PARTS READ_ONLY/TYPE=AUDIT/STATUS=FAILURE
%DBO-I-MODSECAUD, modified audit characteristics for
  READ_ONLY security schema
```

In this example, each user mapped to the READ_ONLY security schema generates an audit record, whenever the user incurs a security violation while accessing any record, item, or set within the PARTS database.

9.5 DBO/BACKUP Commands

There are three forms of the DBO/BACKUP command: DBO/BACKUP, DBO/BACKUP/MULTITHREAD, and DBO/BACKUP/AFTER_JOURNAL. The three commands have the same parameters but different qualifiers. No one of these commands backs up both the database files and the after-image journal (.AIJ) files.

DBO/BACKUP and DBO/BACKUP/MULTITHREAD are two different ways of backing up a database or storage area. A backup file produced by one of these commands can be restored only by the corresponding restore command. A backup produced by DBO/BACKUP must be restored with the DBO/RESTORE command. A backup produced by DBO/BACKUP/MULTITHREAD must be restored with the DBO/RESTORE/MULTITHREAD command.

The DBO/BACKUP command performs a full or incremental backup of a database to a file on disk or tape. However, the tape handling capabilities in the Multithreaded Backup utility are superior to those capabilities in the Single-Threaded Backup utility.

The DBO/BACKUP/MULTITHREAD command performs a full or incremental backup of a database to a file on a single disk or single or multiple tapes. The Multithreaded Backup utility supports the use of multiple tape drives. Multithreaded backup to tape significantly decreases the amount of time required to perform full and incremental backups.

The DBO/BACKUP /AFTER_JOURNAL command backs up one or more after-image journal files concurrently to a single .AIJ backup file, either on disk or on tape. The backup .AIJ file is an actual, usable .AIJ file that can be applied to the appropriate Oracle CODASYL DBMS database in a rollforward operation.

The DBO/BACKUP /AFTER_JOURNAL command can be used while users are bound to the database.

Note

Use only the DBO/BACKUP command to back up all Oracle CODASYL DBMS databases. Do not back up a database by using any other method, such as the OpenVMS Backup Utility. The database root is updated only when the DBO/BACKUP command is used.

9.5.1 DBO/BACKUP Database Command

Creates a backup copy of the database or storage area and places it in a file.

Format

DBO/BACKUP root-file-spec backup-file-spec

Command Qualifiers	Defaults
/[NO]AREA=area-name [,...]	
/[NO]CHECKSUM	/NOCHECKSUM
/[NO]CONTINUE	/NOCONTINUE
/[NO]INCREMENTAL	/NOINCREMENTAL
/[NO]LOG	Current DCL verify value
/[NO]ONLINE	/NOONLINE
/VERIFY=option	/VERIFY=SETS

Description

You can perform a full or incremental backup of the entire database. Oracle recommends using the Single-Threaded Backup utility when backing up databases or specified storage areas to disk. You can use single-threaded backup to back up databases or storage areas to tape; however, using multithread backup to tape would be much faster and more reliable due to the use of tape redundancy blocks. See DBO/BACKUP/MULTITHREADED for information on multithreaded backup.

DBO writes backup files in compressed format to save space. Free space in the root file and on each database page is not written to the backup file.

In the event of subsequent damage to the database, you can specify backup files in a DBO/RESTORE command to restore the database to its condition when you backed it up.

Parameters

root-file-spec
Specifies the root file of a database to be backed up. The default file type is .ROO.

backup-file-spec
Specifies the backup file for the database.

It is good practice to write backup files (as well as journal files) to a device other than the device where the root, storage area, and snapshot files of the database are located. This way, if there is a problem with the database disk, you can still restore the database from backup. However, failure to place the backup copy on another device does not constitute an error.

Oracle recommends that you do not perform backups to ANSI tapes using single-threaded backup. However, if you choose to perform a backup to ANSI tapes, use a backup file name of 17 characters or less. Backup truncates longer file names because OpenVMS enforces the ANSI file-name-length limit. The default file type is .DBB.

Command Qualifiers

/AREA=area-name [,...]
/NOAREA=area-name [,...]
Specifies one or more database storage areas for incremental backup. The /AREA qualifier must be used in conjunction with the /INCREMENTAL qualifier to back up database pages changed since the last full backup. Areas specified with the /NOAREA qualifier are not backed up. The backup is labeled as a partial backup in the backup file header.

/CHECKSUM
/NOCHECKSUM
During the backup operation, DBO reads in enough data to fill a buffer. When the buffer is full, DBO calculates a checksum value based on the data contained in the buffer. The checksum is stored in the CHECKSUM field of the buffer header. Then the full buffer and buffer header are written out to the backup file.

This checksum value is used with the DBO/RESTORE/VERIFY command to verify the backup file. When verifying the backup file, DBO reads in the same block of data, filling the buffer and buffer header. It calculates the checksum value and compares the calculated value to the value stored in the CHECKSUM field of the buffer header. If the two values are the same, DBO continues to verify the backup file. If the two values are different, or if a checksum value is not found, an error occurs.

The default is /NOCHECKSUM.

/CONTINUE
/NOCONTINUE
Specifies whether or not the backup of the database should continue even if errors are found during verification. Use /CONTINUE with the /VERIFY qualifier. If you specify /CONTINUE, the backup of the database continues despite errors found. The default is /NOCONTINUE. If you specify /NOCONTINUE, the backup of the database stops if errors are found.

/INCREMENTAL
/NOINCREMENTAL
Specifies an incremental backup be performed. Only those database pages that have changed since the last full backup of the database are backed up. The first backup of your database must be a full backup. After that, you can use the /INCREMENTAL qualifier to perform incremental backups. The default, /NOINCREMENTAL, backs up the entire database.

Used in conjunction with the /AREA qualifier, backs up the database pages changed since the last full backup for only the areas specified.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request log output and /NOLOG to prevent it. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/ONLINE
/NOONLINE
Allows full or incremental backups without the need to shut down the database. To use /ONLINE, you must allow and enable snapshot files for all storage areas.

When you specify the /ONLINE qualifier, users can continue to update the database during the procedure. The backup procedure uses the snapshot file to retrieve the previous version of any record that has been modified since the backup procedure began.

If you use /NOONLINE, the default, users cannot be bound to the database. The offline backup process has EXCLUSIVE access to the database and does not require snapshot files to work. To use /NOONLINE, you do not need to allow or enable snapshots.

/VERIFY=option

Verifies the database while performing an online full, incremental, or by-area backup.

The backup and verification processes use snapshots to get a consistent view of the database. Therefore, snapshots must be allowed and enabled and you must specify the /ONLINE qualifier.

The verification options available are:

- PAGES

  Verifies database page format

- SEGMENTS

  Verifies both database storage segments and database page formats

- SETS

  Verifies database storage set linkages, database storage segments, and database page formats

SETS is the default. If the section in question is verified without error, then DBO backs up that section. It the section contains an error, DBO does not back up that section unless you have used the /CONTINUE qualifier.

Examples

1.
```
$ DBO/BACKUP PARTS DISK:[BACKUPS]PARTSBKUP
```

   This command performs a full backup of the PARTS database. The full backup file will be in DISK:[BACKUPS]PARTSBKUP.DBB.

2.
```
$ DBO/BACKUP/INCREMENTAL PARTS DISK:[BACKUPS]PARTSINBU
```

   This command performs an incremental backup of the PARTS database.

3.
```
$ DBO/BACKUP/ONLINE PARTS DISK:[BACKUPS]PARTSBKUP
%DBO-I-QUIETPT, waiting for database quiet point
```

   To back up the database without first closing it, specify the /ONLINE qualifier. The backup will not begin until a quiet point is reached.

4.
```
$ DBO/BACKUP/ONLINE/VERIFY=SETS/CONTINUE PARTS -
_$ DISK:[BACKUPS]PARTSBKUP
```

   This command verifies the database at the set level while performing an online backup. If an error occurs, the backup will continue.

9.5.2 DBO/BACKUP/AFTER_JOURNAL Command

Creates a backup copy of the database after-image journal (.AIJ) file or files.

Oracle CODASYL DBMS supports two types of journaling mechanisms: one that employs a single, extensible .AIJ file and another that employs multiple, fixed-size .AIJ files. Which type of journaling mechanism is being used at the time the backup operation starts can affect how you should specify the backup command.

The backup .AIJ file is an actual, usable .AIJ file that can be applied to the appropriate Oracle CODASYL DBMS database in a rollforward operation.

The DBO/BACKUP /AFTER_JOURNAL command can be used while users are bound to the database.

Format

DBO/BACKUP/AFTER_JOURNAL root-file-spec backup-file-spec

Command Qualifiers	Defaults
/ACCEPT_LABEL	See description
/ACTIVE_IO=max-writes	/ACTIVE_IO=3
/BUFFER_SIZE=integer	See description
/COMPRESSION[=ZLIB[=level]]	Not compressed
/[NO]CONTINUOUS=[integer]	/NOCONTINUOUS
/CRC[=AUTODIN_II | CHECKSUM	/See description
/DENSITY=number	See description
/[NO]EDIT_FILENAME[=options]	/NOEDIT_FILENAME
/ENCRYPT=(option[,...])	Not encrypted
/FORMAT={OLD_FILE | NEW_TAPE}	/FORMAT=OLD_FILE
/[NO]GROUP_SIZE[=interval]	See description
/IDENTIFIER=user-id	
/[NO]INTERVAL=integer	/NOINTERVAL
/LABEL=label-name-list	See description
/[NO]LOG	Current DCL verify value
/PROTECTION=file-protection	See description
/[NO]QUIET_POINT	/QUIET_POINT
/[NO]RENAME	/NORENAME
/[NO]REWIND	/NOREWIND
/[NO]SEQUENCE= (first-sequence [,last-sequence])	/NOSEQUENCE
/TAPE_EXPIRATION=date-time	/TAPE_EXPIRATION=now
/[NO]THRESHOLD=integer	/NOTHRESHOLD
/UNTIL=time	
/[NO]UPDATE	/NOUPDATE
/[NO]WAIT	/NOWAIT

Description

The backup .AIJ file you create can be used with the DBO/RECOVER command to recover (roll forward) journaled transactions. In some cases, you may have to issue additional RECOVER commands: one for the backup .AIJ file and a second for the more recent .AIJ files.

Oracle CODASYL DBMS supports two types of .AIJ file configurations:

1. A configuration that uses a single, extensible .AIJ file

This method is the default for compatibility with prior versions of Oracle CODASYL DBMS.

When an extensible .AIJ file is used, one .AIJ file is written to and extended, as needed, by the number of blocks specified when the .AIJ file was created. The .AIJ file continues to be extended until it is backed up.

The DBO/BACKUP /AFTER_JOURNAL command copies transactions recorded in the current .AIJ file (always on a disk device) to the backup .AIJ file (which may be on a tape or disk device). Upon completion, the current .AIJ file is truncated and used again. During periods of high-update activity, the truncation of the active .AIJ file may not be performed because of conflicting access to the .AIJ file by other users. But the storage allocated to the active .AIJ file is still used again when the backup completes.

2. A configuration that uses two or more fixed-size .AIJ files

   When fixed-size .AIJ files are used, the database maintains multiple .AIJ files, however, only one .AIJ file is written to at a time. This .AIJ file is considered the *current* journal. When this .AIJ file is filled, an automatic switchover occurs to allow journaling to continue in another .AIJ file. The last .AIJ file will ultimately switch over to the first .AIJ file.

   The DBO/BACKUP /AFTER_JOURNAL command copies only those .AIJ files to the backup file that are not being actively accessed. They may be copied to tape or a disk device. Therefore the problem of contention seen with an extensible .AIJ file during high update activity is not seen when fixed-size .AIJ files are used.

   Once a specified .AIJ file has been completely backed up, it is initialized and marked as available for reuse.

Note

Which method is employed, fixed-size .AIJ files or an extensible .AIJ file, cannot be set explicitly by the user. Any event that reduces the number of journal files to one results in an extensible journal being used. Any event that increases the accessible .AIJ files to two or more results in fixed-size .AIJ files being used.

Because some of the DBO/BACKUP /AFTER_JOURNAL qualifiers are valid when only one or the other journaling mechanism is employed, you may need to issue a DBO/DUMP command to determine which journaling mechanism is currently being employed before you issue a DBO/BACKUP /AFTER_JOURNAL command.

Also note that once a backup operation begins, .AIJ file modification is not allowed until the backup operation is complete. However, if the type of journaling changes between the time you issue a DBO/DUMP command and the time you issue the DBO/BACKUP /AFTER_JOURNAL command, you will receive an error message if you have specified qualifiers that are only valid with a particular type of journaling mechanism. (The /THRESHOLD qualifier, for example, is valid only when the extensible journaling mechanism is being used.)

If you back up the .AIJ file or files to tape, you must mount the backup media using the DCL MOUNT command before you issue the DBO/BACKUP /AFTER_JOURNAL command. If you specify the default, /FORMAT=OLD_RMS, the DBO/BACKUP /AFTER_JOURNAL command uses RMS to write to the tape and the tape must be mounted as an OpenVMS volume. (That is, do not specify the /FOREIGN qualifier with the MOUNT command.) If you specify /FORMAT=NEW_TAPE, the DBO/BACKUP /AFTER_JOURNAL command writes backup files in a format similar to that used by DBO/BACKUP, and you must mount the tape as a FOREIGN volume.

If you back up an .AIJ file to disk, you can then use the OpenVMS Backup Utility (BACKUP) to archive the .AIJ backup file.

Note

If an AIJ backup process fails or is terminated prematurely, OpenVMS might discard the resulting .AIJ backup file because the backup operation was not completed. However, all .AIJ backup files, including those produced by a failed backup process, are necessary to recover a database. If an .AIJ backup file of a failed backup process is discarded, the database is not recoverable from that point forward. This is especially important if you use a magnetic tape as the AIJ backup media; in this case, preserve the magnetic tape and do not reuse it.

The DBO/BACKUP /AFTER_JOURNAL command can be used in a batch job to avoid tying up an interactive terminal for long periods of time. The /CONTINUOUS, /INTERVAL, /THRESHOLD, and /UNTIL qualifiers control the duration and frequency of the backup process. When you use the /CONTINUOUS qualifier, the command can occupy a terminal indefinitely. Therefore, it is good practice to issue the command through a batch process when executing a continuous after-image journal backup. However, remember that the portion of the command procedure that follows the DBO/BACKUP /AFTER_JOURNAL command will not be executed until the time specified by the /UNTIL qualifier.

When the DBO/BACKUP /AFTER_JOURNAL command completes, it records information about the state of the backup files in the global process symbols shown in the following list. You can use these symbols in DCL command procedures to help automate the backup operation.

- DBM$AIJ_SEQNO

  Contains the sequence number of the last .AIJ backup file written to tape. This symbol is synonymous to DBM$AIJ_BACKUP_SEQNO and is maintained for compatibility with earlier versions of Oracle CODASYL DBMS.

- DBM$AIJ_CURRENT_SEQNO

  Contains the sequence number of the currently active .AIJ file. A value of -1 indicates that after-image journaling is disabled.

- DBM$AIJ_NEXT_SEQNO

  Contains the sequence number of the next .AIJ file that needs to be backed up. This symbol always contains a positive integer value (which may be 0).

- DBM$AIJ_LAST_SEQNO

Contains the sequence number of the last .AIJ file available for backup that is different from the current sequence number if fixed-size journaling is being used. A value of -1 indicates that no journal file has ever been backed up.

If the value of the DBM$AIJ_NEXT_SEQNO symbol is greater than the value of the DBM$AIJ_LAST_SEQNO symbol, then no more .AIJ files are currently available for the backup operation.

- DBM$AIJ_BACKUP_SEQNO

  Contains the sequence number of the last .AIJ file backed up by the backup operation. This symbol is set at the completion of an AIJ backup operation. A value of -1 indicates that this process has not yet backed up an .AIJ file.

  The DBO/BACKUP /AFTER_JOURNAL command provides an informational message that describes the exact sequence number for each AIJ backup operation.

- DBM$AIJ_COUNT

  Contains the number of available .AIJ files.

Parameters

root-file-spec
Specifies the root file of a database. An error results if you specify a database that does not have after-image journaling enabled. The default file type is .ROO.

backup-file-spec
Specifies the file specification for the after-image journal backup file. The default file type is .AIJ. A file specification for the .AIJ backup file is required. When you use the /RENAME qualifier, specify a null string ("") for the backup-file-spec parameter.

The default file extension is .AIJ unless you specify the /FORMAT=NEW_TAPE qualifier. In this case, the default file extension is .AIJ_DBF.

Command Qualifiers

/ACCEPT_LABEL
Specifies that DBO should keep the current tape label it finds on a tape during a backup operation even if that label does not match the default label or that specified with the /LABEL qualifier. Operator notification does not occur unless the tape's protection, owner, or expiration date prohibit writing to the tape. However, a message is logged (assuming logging is enabled) and written to the backup journal file (assuming you have specified the /JOURNAL qualifier) to indicate that a label is being preserved and which drive currently holds that tape.

This qualifier is particularly useful when your backup operation employs numerous previously used (and thus labeled) tapes and you want to preserve the labels currently on the tapes.

If you do not specify this qualifier, and you are performing a backup operation to tape, the default behavior of DBO is to notify the operator each time it finds a mismatch between the current label on the tape and the default label (or the label you specify with the /LABEL qualifier).

/ACTIVE_IO=max-writes
Specifies the maximum number of write operations to a backup device that the DBO/BACKUP /AFTER_JOURNAL command will attempt simultaneously. This is not the maximum number of write operations in progress; that value is the product of active system I/O operations and the number of devices being written to simultaneously.

The value of the /ACTIVE_IO qualifier can range from 1 to 5. The default value is 3. Values larger than 3 can improve performance with some tape drives.

/BUFFER_SIZE=integer
Specifies the maximum record size for the backup file. The size can vary between 2048 and 65,024 bytes. The default value is device dependent. The appropriate buffer size is a compromise between tape capacity and error rate.

/COMPRESSION
/COMPRESSION=ZLIB
/NOCOMPRESSION
Allows you to specify the compression method to use before writing data to the AIJ backup file. This may increase CPU usage during the backup, but may be justified by a smaller AIJ backup file when written as a disk file, or is being backed up over a busy network, or is being backed up to a tape drive that does not do its own compression. You probably do not want to specify the /COMPRESSION qualifier when you are backing up an AIJ file to a tape drive that does its own compression; in some cases doing so can actually result in a larger file.

If you specify the /COMPRESSION qualifier without a value, the default is /COMPRESSION=ZLIB=6.

The level value (ZLIB=level) is an integer between 1 and 9 specifying the relative compression level with one being the least amount of compression and nine being the greatest amount of compression. Higher levels of the compression use increased CPU time while generally providing better compression. The default compression level of 6 is a balance between compression effectiveness and CPU consumption.

Note that the /COMPRESSION qualifier need only be used with the DBO/BACKUP /AFTER_JOURNAL command as the DBO /RECOVER command determines whether a backup file has been compressed from information saved within the backup file.

/CONTINUOUS=[integer]
/NOCONTINUOUS
Specifies whether or not the AIJ backup process operates continuously, under control of the /INTERVAL and /UNTIL qualifiers, or the iteration-count directive, as specified by “integer”. One or the other must be specified with the /CONTINUOUS qualifier to indicate when the AIJ backup is to terminate. The default value is NOCONTINUOUS.

If you specify /CONTINUOUS, DBO does not terminate the backup process after backing up all .AIJ files. Instead, the backup process follows the directive of the /INTERVAL and /UNTIL qualifiers, or the iteration count.

If you specify /INTERVAL=integer, it waits for the period of time specified by the /INTERVAL qualifier. After that time interval, the backup process tests to see if the threshold set by the /THRESHOLDS qualifier has been reached. The backup process performs backup as needed, and then waits again until the next interval break.

The /CONTINUOUS=integer qualifier specifies the number of passes the backup utility will make through the complete set of after-image journal files; the default value is 1. For example, if the database contains 4 active after-image journal files, and you specify /CONTINUOUS=2, a total of 8 journal files will be backed up.

When you specify the /CONTINUOUS qualifier, the backup process occupies the terminal (that is, no DCL prompt occurs) until the process terminates. Therefore, you should normally execute the command from a batch process.

If you do not specify /CONTINUOUS, the backup process determines if the threshold set by the /THRESHOLD qualifier has been reached, performs backup as needed, and stops. You must enter another DBO/BACKUP /AFTER_JOURNAL command to resume AIJ backup.

If you specify /NOCONTINUOUS, the default, the backup process stops as soon as it completely backs up the after-image journal files. Use of this qualifier implies an iteration-count of 1.

If you specify the /CONTINUOUS qualifier, Oracle CODASYL DBMS does not terminate the backup process after truncating the current .AIJ file (when an extensible journal is used) or after switchover to a new journal file (when fixed-size journal files are used). Instead, the backup process waits for the period of time that you specify in the argument to the /INTERVAL qualifier. After that time interval, the backup process tests to determine if the threshold has been reached (for an extensible journal file) or if the journal file is full (for fixed-size journal files). It then performs backup operations as needed, and then waits again until the next interval break, unless the number of iterations or the conditions specified with the /UNTIL qualifier has been reached.

The /CONTINUOUS qualifier is not recommended when you are backing up to tape. If your tape operations complete successfully, you do not want the backup operation to continue in an infinite loop.

Using DCL STOP to terminate a backup operation to tape may result in an incomplete or corrupt backup file.

/CRC[=AUTODIN_II]
Uses the AUTODIN-II polynomial for the 32-bit CRC calculation and provides the most reliable end-to-end error detection. This is the default for NRZ/PE (800/1600 bits/inch) tape drives.

Typing CRC is sufficient to select the /CRC=AUTODIN_II qualifier. It is not necessary to type the entire qualifier.

/CRC=CHECKSUM
Uses one's complement addition, which is the same computation used to do a checksum of the database pages on disk. This is the default for GCR (6250 bits/inch) tape drives and for TA78, TA79, and TA81 tape drives.

The /CRC=CHECKSUM qualifier allows detection of errors.

/DENSITY=number
Specifies the density at which the output volume is to be written. The default value is the format of the first volume (the first tape you mount). You do not need to specify this qualifier unless your tape drives support data compression or more than one recording density.

The /DENSITY qualifier is applicable only to tape drives. DBO returns an error message if this qualifier is used and the target device is not a tape drive.

Specify the /DENSITY qualifier as follows:

- For TA90E, TA91, and TA92 tape drives, specify the number in bits per inch as follows:
  - DENSITY = 70000 to initialize and write tapes in the compacted format.
  - DENSITY = 39872 or DENSITY = 40000 for the noncompacted format.
- For SCSI (Small Computer System Interface) tape drives, specify DENSITY = 1 to initialize and write tapes using the drive's hardware data compression scheme.
- For other types of tape drives, you can specify a supported density value between 800 and 160000 bits per inch.
- For all tape drives, specify DENSITY = 0 to initialize and write tapes at the drive's standard density.

/EDIT_FILENAME=[options]
/NOEDIT_FILENAME

Appends to the backup file any or all of the values specified by the following options:

- Vno

  The journal sequence number of the first journal in the backup operation.
- Year

  The current year (A.D.) expressed as a 4-digit integer.
- Month

  The current month expressed as a 2-digit integer (01 to 12).
- Hour

  The current hour of the day expressed as a 2-digit integer (00 to 23).
- Minute

  The current minute of the hour expressed as a 2-digit integer (00 to 59).
- Day_Of_Year

  The current day of the year expressed as a 3-digit integer (001 to 366).
- Day_Of_Month

  The current day of the month expressed as a 2-digit integer (01 to 31).
- Day_Of_Week

  The current day of the week expressed as a 1-digit integer (1 to 7). Sunday is expressed as 1; Saturday is expressed as 7.
- Julian_Date

  The number of days passed since 17-Nov-1858.

If you specify more than one option, place a comma between each option.

The edit is performed in the order specified. For example, the file BACKUP.AIJ and the qualifier /EDIT_FILENAME=(Hour, Minute, Month, Day_Of_Month, Vno) will create a file with the name BACKUP_160504233.AIJ when journal 3 is backed up at 4:05 P.M. on April 23rd.

You can make the name more readable by inserting quoted strings between each /EDIT_FILENAME option. For example, the following qualifier will add the string "$30_0155-2" to the AIJ file name if the day of the month is the 30th, the time is 1:55 A.M. and the version number is 2:

```
/EDIT_FILENAME=("$",DAY_OF_MONTH,"_",HOUR,MINUTE,"-",VNO)
```

This qualifier is useful for creating meaningful file names for your backup files and makes file management easier.

The default is the /NOEDIT_FILENAME qualifier.

/ENCRYPT=({VALUE=|NAME=}[,ALGORITHM=])

The /ENCRYPT qualifier specifies the encryption key and algorithm so that the DBO/BACKUP /AFTER_JOURNAL command can encrypt the after image journal backup.

Specify a key value as a string or, the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters see HELP ENCRYPT.

This feature requires the OpenVMS Encrypt product to be installed and licensed on this system.

Encryption Key

If you cannot remember the encryption key you have effectively lost all data in the encrypted file. The encryption key used on the backup command will be REQUIRED on the RESTORE command of that backup file. It is also used by the DBO/DUMP/AFTER_IMAGE command.

/FORMAT=OLD_FILE
/FORMAT=NEW_TAPE

Specifies the format in which the backup file is to be written. Oracle Corporation recommends that you specify the /FORMAT=OLD_FILE qualifier (or accept the default) when you back up your AIJ file to disk and that you specify the /FORMAT=NEW_TAPE qualifier when you back up your AIJ file to tape.

If you specify the default, the /FORMAT=OLD_FILE qualifier, the DBO/BACKUP command writes the file in a format that is optimized for a file-structured disk.

If you specify the /FORMAT=NEW_TAPE qualifier, the DBO command writes the file in a format that is optimized for tape storage, including ANSI/ISO labeling and end-to-end error detection and correction. When you specify the /FORMAT=NEW_TAPE qualifier and back up the AIJ file to tape, you must mount the backup media by using the DCL MOUNT command before you issue the DBO/BACKUP /AFTER_JOURNAL command. The tape must be mounted as a foreign volume. If you mount the tape as an OpenVMS volume (that is, you do not mount it as a foreign volume) and you specify the /FORMAT=NEW_TAPE qualifier, you receive a DBO-F-MOUNTFOR error.

The following tape qualifiers have meaning only when used in conjunction with the /FORMAT=NEW_TAPE qualifier:

- /ACCEPT_LABEL
- /ACTIVE_IO
- /BUFFER_SIZE

- /CRC
- /DENSITY
- /EDIT_FILENAME
- /GROUP_SIZE
- /IDENTIFIER
- /LABEL
- /MEDIA_LOADER
- /PROTECTION
- /REWIND
- /TAPE_EXPIRATION

The default file specification when you specify the /FORMAT=NEW_TAPE qualifier is .AIJ_DBF. The default file specification when you specify the /FORMAT=OLD_FILE qualifier is .AIJ.

Although Oracle Corporation recommends that you specify the /FORMAT=NEW_TAPE qualifier for AIJ backup operations to tape and the /FORMAT=OLD_FILE qualifier for AIJ backup operations to disk, DBO does not enforce this recommendation. This is to provide compatibility with prior versions of Oracle CODASYL DBMS.

/GROUP_SIZE[=interval]
/NOGROUP_SIZE
Specifies the frequency at which XOR recovery blocks are written to tape. The group size can vary from 0 to 100. Specifying a group size of 0 or specifying the /NOGROUP_SIZE qualifier results in no XOR recovery blocks being written. The /GROUP_SIZE qualifier is applicable only to tape, and its default value is device dependent. DBO returns an error message if this qualifier is used and the target device is not a tape drive.

/IDENTIFIER=user-id
Specifies the owner of the tape volume set. The owner is the user who will be permitted to restore the database. The user-id parameter must be one of the following types of OpenVMS identifier:

- A user identification code (UIC) in [group-name,member-name] alphanumeric format
- A UIC in [group-number,member-number] numeric format
- A general identifier, such as SECRETARIES
- A system-defined identifier, such as DIALUP

The /IDENTIFIER qualifier cannot be used with a backup operation to disk. When used with tapes, the /IDENTIFIER qualifier applies to all continuation volumes. Unless the /REWIND qualifier is also specified, the /IDENTIFIER qualifier is not applied to the first volume. If the /REWIND qualifier is not specified, the backup operation appends the file to a previously labeled tape; thus, the first volume can have a protection different from the continuation volumes.

/INTERVAL=integer
/NOINTERVAL

Specifies the number of seconds for which the backup process waits. Use this qualifier in conjunction with the /CONTINUOUS qualifier and the extensible journaling method. The interval determines how often to test the active .AIJ file to determine if it contains more blocks than the value of the /THRESHOLD qualifier.

If you specify the /INTERVAL without specifying the number of seconds, or if you omit this qualifier, the default number of seconds is 60.

Oracle recommends using the default (/INTERVAL=60) initially because the interval that you choose can affect the performance of the database. In general, you can arrive at a good interval time on a given database only by judgment and experimentation.

If you specify the /NOINTERVAL qualifier, the active .AIJ file is tested repeatedly with no interval between finishing one cycle and beginning the next.

You must specifying the /CONTINUOUS qualifier if you specify either the /INTERVAL or /NOINTERVAL qualifier.

If you specify both the /INTERVAL and /NOCONTINUOUS qualifiers, the /INTERVAL qualifier is ignored.

/LABEL=label-name-list

Specifies the 1- to 6-character string with which the volumes of the backup file are to be labeled. The /LABEL qualifier is applicable only to tape volumes. You must specify one or more label names when you use the /LABEL qualifier.

If you do not specify the /LABEL (or /ACCEPT_LABEL) qualifier, DBO labels the first tape used for a backup operation with the first 6 characters of the backup file name. Subsequent default labels are the first 4 characters of the backup file name appended with a sequential number. For example, if your backup file is PARTS.ROO, the default tape labels are PARTS, PART01, PART02, and so on.

When you reuse tapes, DBO compares the label currently on the tape to the label or labels you specify with the /LABEL qualifier. If there is a mismatch between the existing label and a label you specify, DBO sends a message to the operator asking if the mismatch is acceptable (unless you also specify the /ACCEPT_ LABEL qualifier).

If desired, you can explicitly specify the list of tape labels for multiple tapes. If you list multiple tape label names, separate the names with commas and enclose the list of names within parentheses. If you are reusing tapes be certain that you load the tapes so that the label DBO expects and the label on each tape will match, or be prepared for a high level of operator intervention.

If you specify fewer labels than are needed, DBO will generate labels based on the format you have specified. For example, if you specify /LABEL=TAPE01, DBO labels subsequent tapes as TAPE02, TAPE03, and so on up to TAPE99. Thus, many volumes can be preloaded in the cartridge stacker of a tape drive. The order is not important because DBO relabels the volumes. An unattended backup operation is more likely to be successful if all the tapes used do not have to be mounted in a specific order.

Once the backup operation is complete, externally mark the tapes with the appropriate label so the order can be maintained for the restore operation. Be particularly careful if you are allowing DBO to implicitly label second and subsequent tapes and you are performing an unattended backup operation. Remove the tapes from the drives in the order in which they were written. Apply labels to the volumes following the logic of implicit labeling (for example, TAPE02, TAPE03, and so on).

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request log output and /NOLOG to prevent it. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/OUTPUT=file
/OUTPUT=SYS$OUTPUT
Data is written to the file specified by the /OUTPUT qualifier, or to the default output file location, SYS$OUTPUT. The default file type is .LIS.

/PROTECTION=file-protection
Specifies the system file protection for the backup file produced by the DBO/BACKUP /AFTER_JOURNAL command.

The default file protection varies, depending on whether or not you back up the file to disk or tape. This is because tapes do not allow delete or execute access and the system account always has both read and write access to tapes. In addition, a more restrictive class accumulates the access rights of the less restrictive classes. If you do not specify the /PROTECTION qualifier, the default protection is as follows:

- S:RWED,O:RE,G,W if the backup is to disk
- S:RW,O:R,G,W if the backup is to tape

If you specify the /PROTECTION qualifier explicitly, the differences in protection applied for backups to tape or disk as noted in the preceding paragraph are applied. Thus, if you specify PROTECTION=(S,O,G:W,W:R), that protection on tape becomes (S:RW,O:RW,G:RW,W:R).

/QUIET_POINT
/NOQUIET_POINT
Specifies whether or not the quiet-point lock will be acquired when performing an AIJ backup operation. The default is /QUIET_POINT. Use of the /QUIET_POINT qualifier is meaningful only when performing a full backup operation; that is, a backup that makes a complete pass through all available .AIJ files. A full AIJ backup operation can be performed regardless of whether or not an extensible or a fixed-size AIJ journaling mechanism is being employed.

For each AIJ backup operation, an AIJ sequence number is assigned. This labeling distinguishes each .AIJ backup file from previous .AIJ backup files. During a rollforward operation, it is important to apply the .AIJ backup files in the proper sequence. The DBO/RECOVER command checks the database root file structure and displays a message telling you the AIJ sequence number with which to begin the rollforward.

The quiet point is a state where all write transactions have either been committed or rolled back and no update transactions are in progress. In that state, the AIJ backup operation is guaranteed to back up everything completed before the quiet point. The backup file can then be used to produce a recovered database that is in the same state as when the quiet point was reached.

When fixed-size journaling is employed, the /QUIET_POINT qualifier is only relevant when the active .AIJ file is being backed up. In this case, a quiet point is acquired only once, regardless of the number of .AIJ files being backed up.

There is no natural quiet point if there is someone writing or waiting to write to the database at any given time. (A natural quiet point is one that is not instigated by the use of the QP Lock.) The AIJ backup operation may never be able to capture a state that does not have uncommitted data in the database. As a result, the /NOQUIET_POINT qualifier creates .AIJ backup files that are not independent of one another. If you apply one .AIJ backup file to the database without applying the next .AIJ backup file in sequence, the backup will not be applied completely.

The /WAIT and /SEQUENCE qualifiers are invalid when used with the /QUIET_POINT qualifier.

Note

When FAST COMMIT is enabled, and an extensible .AIJ file configuration is used, the backup process must compress and retain some fraction of the original .AIJ file. This fraction may approach 100% of the original size. Therefore, be sure to reserve enough space to duplicate the maximum size .AIJ file.

Oracle recommends that you schedule journal file backups with sufficient frequency and check the free space and journal file size periodically, so that you know when you are approaching a critical situation in terms of free space. (This is good practice whether or not you have FAST COMMIT enabled.)

However, if you issue the DBO/BACKUP /AFTER_JOURNAL command with FAST COMMIT enabled and find that you have insufficient space for the .AIJ file, you have the following options:

- Delete unneeded files to create sufficient space on the disk where the .AIJ file is located.
- Temporarily disable FAST COMMIT and back up the .AIJ file.
- Close the database, disable after-image journaling, enable a new after-image journal and perform a backup. (The database can be opened either before or after the backup.)
- Close the database. Create a bound volume set or stripe set that is large enough for the .AIJ file and copy the .AIJ file there. Use the DBO/MODIFY/AIJ_OPTIONS command to change the .AIJ file name (or redefine the logical name if one was used to locate the journal file), then open the database again.

/RENAME
/NORENAME
Creates and initializes a new .AIJ file and creates the backup file by renaming the original .AIJ file. The effect is that the original .AIJ file is the backup file and the new .AIJ file is the new after-image journal file.

When the /RENAME qualifier is used, the backup operation is faster than when /NORENAME, the default, is specified, because the duration of the backup is the total time required to rename and initialize the .AIJ file; the data copy portion of the backup (reading and writing) is eliminated. However, the disk containing the .AIJ file must have sufficient space for both the new and original .AIJ files. Note also that the .AIJ backup file name cannot be specified.

When you use the /RENAME qualifier, specify a null string ("") for the backup-file-spec parameter.

Note

If there is insufficient space for both the new and original .AIJ files when the /RENAME qualifier is specified, after-image journaling shutdown is invoked which will result in a complete database shutdown.

The /RENAME qualifier can be used with both fixed-size and extensible journaling.

The /NORENAME qualifier copies the contents of the .AIJ file to tape or disk and initializes the original .AIJ file for reuse. The /NORENAME qualifier results in a slower backup operation than when /RENAME is specified, but it does not require space on the journal disk for both new and original .AIJ files.

/NORENAME is the default.

/REWIND
/NOREWIND
Specifies that the magnetic tape that contains the backup file will be rewound before processing begins. The tape will be initialized according to the /LABEL and /DENSITY qualifiers. The /NOREWIND qualifier is the default and causes the backup file to be created starting at the current logical end-of-tape (EOT).

This qualifier is applicable only to tape drives.

/SEQUENCE=(n[,m])
Specifies that the journal files with sequence numbers from n to m inclusive are to be backed up. The values n and m are interpreted or interpolated as follows:

- If /SEQUENCE = (33, 35) is specified, then .AIJ file with sequence numbers 33, 34, and 35 are backed up.
- If /SEQUENCE = (53, 53) is specified, then the .AIJ file with sequence number 53 is backed up.
- If /SEQUENCE = (53) is specified, then the .AIJ file with sequence number 53 is backed up.
- If /SEQUENCE = (55,53) is specified, then the .AIJ file with sequence numbers 53, 54, and 55 are backed up.

- If the /SEQUENCE qualifier is specified without a value list, both n and m are set at the sequence number of the next journal file that needs to be backed up.

The following qualifiers cannot be used or have no effect when used with the /SEQUENCE qualifier: /QUIET_POINT, /THRESHOLD, /CONTINUOUS, /INTERVAL, or /UNTIL. Furthermore, fixed-size after-image journal files must be in use when this qualifier is specified.

/TAPE_EXPIRATION=date-time
Specifies the expiration date of the backup file. The default for this qualifier is now (the current time), meaning that the tape can be written over immediately. The /TAPE_EXPIRATION qualifier cannot be used with a backup operation to disk.

/THRESHOLD=integer
/NOTHRESHOLD
Specifies the approximate limit on the number of disk blocks for the .AIJ file. The default is /NOTHRESHOLD. This qualifier is invalid when multiple, fixed-size journaling is enabled. This qualifier can only be used when extensible journaling is enabled. It cannot be used with fixed-size journaling.

This qualifier sets an approximate limit on the size of the primary .AIJ file. When the size of the primary .AIJ file exceeds the threshold, you cannot initiate new transactions until the backup process finishes backing up and truncating (resetting) the primary .AIJ file. During the backup, existing transactions can continue to write to the .AIJ file. Before new transactions can start, all activity issuing from existing transactions (including activity occurring after the threshold was exceeded) must be written from the primary .AIJ file to the backup .AIJ file. At that time, the primary .AIJ file is completely truncated.

An appropriate value for /THRESHOLD depends on the activity of your database, how much disk space you want to use, whether or not the backup will be continuous, and how long you are willing to wait for the backup to complete.

If you use /NOTHRESHOLD, the default, each backup cycle completely backs up the primary .AIJ file. Oracle recommends using the default /NOTHRESHOLD.

/UNTIL=absolute-time
Specifies the future time and date to stop the continuous backup process. The default is /NOUNTIL if you are using multiple .AIJ files. There is no default if you are using a single .AIJ file. If you specify /CONTINUOUS, you must specify /UNTIL, unless you are specifying the iteration-count (CONTINUOUS=integer.)

Note

The /UNTIL qualifier accepts all date and time strings from OpenVMS including international dates. See the OpenVMS documentation for the Run-Time Library (LIB$) for more information. The date and time strings specified must be enclosed within quotation marks because they can contain spaces and other DCL syntax characters such as commas. A colon to separate the date and time is no longer valid.

/UPDATE
/NOUPDATE
Updates the global process symbols that provide AIJ sequence number state information. (For more information about these symbols, see the description section of the DBO/BACKUP /AFTER_JOURNAL command in this chapter.) The /UPDATE qualifier should only be used by itself to set up the backup environment before performing any AIJ backup. If specified with any other qualifier, it will be ignored. The default value is /NOUPDATE.

You can use this qualifier in a DCL script to establish the current backup environment and decide whether or not an AIJ backup is needed, as shown in the following example:

```
DBO/BACKUP/AFTER_JOURNAL/UPDATE PARTS
```

/WAIT
/NOWAIT
Indicates that the backup operation is to wait for an after-image journal file to become ready for backup. The default is /NOWAIT. You must use this qualifier with the /SEQUENCE qualifier. It is implied when you use the /CONTINUOUS qualifier.

For example, it is possible to specify a range of after-image journal files using the /SEQUENCE qualifier. Some of these journal files may not yet be ready for backup. By default, the AIJ backup operation is terminated when it encounters a journal file that is not ready for backup. But if you specify the /WAIT qualifier, the AIJ backup will wait until the journal file is ready for backup.

Use the /INTERVAL qualifier to specify the period at which the backup utility checks to see if the journal file is ready for backup.

Examples

1.
```
$ DBO/BACKUP/AFTER_JOURNAL/CONTINUOUS/THRESHOLD=500 -
_$ /INTERVAL=300/UNTIL="20-JUN-1993 06:00:00.00" -
_$ PARTS DISK12:[PARTS_AIJ]BU_PARTS.AIJ
```

   Assuming that you have enabled after-image journaling for the PARTS database, this command backs up the extensible .AIJ file continuously. Every 300 seconds, the backup process tests to determine if the .AIJ file on disk has reached the threshold size of 500 blocks. If not, transaction processing continues normally for one or more 300-second intervals until the threshold test indicates that the .AIJ file has reached a size of at least 500 blocks. When the .AIJ file reaches that size, DBO allows existing transactions to continue to write to the .AIJ file but does not allow new transactions to start.

   Assuming that the .AIJ file contains 550 blocks, Oracle CODASYL DBMS moves those 550 blocks to the backup journal file and deletes them from the journal file. Then, the backup process determines if the transactions already in progress have written more journal records to the journal file during the backup operation. If so, DBO moves those journal records to the backup file.

   After Oracle CODASYL DBMS completely moves the journal file, it truncates the journal file to its initial allocation. Oracle CODASYL DBMS then allows new transactions to start and the backup process resumes threshold testing at 300-second intervals. The backup process continues until the time and date specified by the /UNTIL qualifier.

2.
```
$ DBO/BACKUP/AFTER_JOURNAL/LOG PARTS PARTS_BKUP_AIJ1.AIJ
%DBO-I-LOGCREBCK, created backup file
     DISK1:[DBMS.AIJ]PARTS_BKUP_AIJ1.AIJ;1
%DBO-I-LOGBCKAIJ, backing up AIJ file
     DISK1:[DBMS.AIJ]PARTS.AIJ;1
%DBO-I-AIJBCKSEQ, backing up current AIJ file sequence number 0
%DBO-I-LOGAIJBCK, backed up 8 committed transactions at 07:49:42.18
    .
    .
!More transactions committed to the database
    .
    .
$ DBO/BACKUP/AFTER_JOURNAL/LOG PARTS PARTS_BKUP_AIJ2.AIJ
%DBO-I-LOGCREBCK, created backup file
     DISK1:[DBMS.AIJ]PARTS_BKUP_AIJ2.AIJ;1
%DBO-I-LOGBCKAIJ, backing up AIJ file
     DISK1:[DBMS.AIJ]PARTS.AIJ;1
%DBO-I-AIJBCKSEQ, backing up current AIJ file sequence number 1
%DBO-I-LOGAIJBCK, backed up 2 committed transactions at 07:51:28.35
```

The commands in this example show the backing up of multiple .AIJ files in sequence. Note that a number of transactions were committed to the database between backup operations.

3.
```
$ DBO/BACKUP/AFTER/LOG/RENAME PARTS
$_ Backup: ""
%DBO-W-DATACMIT, unjournaled changes made; database may not
  be recoverable
%DBO-I-AIJBCKBEG, beginning after-image journal backup operation
%DBO-I-LOGBCKAIJ, backing up after-image journal ONE
%DBO-I-AIJBCKSEQ, backing up after-image journal sequence number 0
%DBO-I-LOGCREAIJ, created after-image journal file
  DISK1:[USER1]ONE.AIJ;3
%DBO-I-AIJBCKEND, after-image journal backup operation
     completed successfully
%DBO-I-LOGAIJJRN, backed up 1 after-image journal at 09:23:30.43
%DBO-I-LOGAIJBLK, backed up 127 after-image journal blocks at
  09:23:30.44
```

This command creates the backup file by renaming the current .AIJ file and creating a new .AIJ file with the same name as the original .AIJ file. Now the file specification ONE.AIJ;3 is the current after-image journal file, and the file specification ONE.AIJ;2 is the after-image journal backup file.

4.
```
$ DBO/BACKUP/AFTER_JOURNAL PARTS AIJ_BCK -
_$ /REWIND/TAPE_EXPIRATION=4-NOV-1997 -
_$ /PROTECTION=(S:RW, O:RWE, G:R, W:R) -
_$ /CRC/ACCEPT_LABEL/FORMAT=NEW_TAPE -
_$ /IDENTIFIER=[DBMS,STEWART]
```

This command backs up an .AIJ file to tape. It specifies the following:

- The tape should be rewound before any data is written to it. (/REWIND)
- The tape will expire on November 4, 1997, which means that it can be overwritten anytime after that date. (/TAPE_EXPIRATION)
- The protection on the file is specified. (/PROTECTION)
- CRC=AUTODIN_II error detection is used. (/CRC)

- If the tape is already labeled, that tape label will be used. If the tape is not already labeled, AIJ_BC will be used for the first tape label, AIJ_01 will be used for the second label, AIJ_02 will be used for the third label, and so on. This is default label naming scheme for a backup file with this name. (/ACCEPT_LABEL, /LABEL default)
- The file will be written in the tape-optimized format. (/FORMAT=NEW_TAPE)
- User [DBMS,STEWART] is owner of the tape volume set. (/IDENTIFIER)

9.5.3 DBO/BACKUP/MULTITHREAD Command

Creates a backup copy of a database or storage area and places it in a file.

Format

DBO/BACKUP/MULTITHREAD root-file-spec backup-file-spec

Command Qualifiers	Defaults
/ACTIVE_IO=integer	/ACTIVE_IO=3
/[NO]AREA=area-name [,...]	
/BUFFER_SIZE=integer	See description
/[NO]CHECKSUM_VERIFICATION	/NOCHECKSUM_VERIFICATION
/COMPRESSION[=ZLIB[=level]]	Not compressed
/[NO]CRC[=option]	See description
/DENSITY=integer	See description
/ENCRYPT=(option[,...])	Not encrypted
/EXTEND_QUANTITY=number-blocks	/EXTEND_QUANTITY=2048
/[NO]GROUP_SIZE=interval	/GROUP_SIZE=10
/IDENTIFIER=option	See description
/INCREMENTAL=option	/INCREMENTAL=COMPLETE
/[NO]JOURNAL[=file-spec]	/NOJOURNAL
/LABEL=label ([,...])	See description
/LOCK_TIMEOUT=integer	
/[NO]LOG	Current DCL verify value
/MASTER	See description
/[NO]MEDIA_LOADER	
/[NO]ONLINE	/NOONLINE
/PAGE_BUFFERS=integer	/PAGE_BUFFER=3
/PROTECTION=file-protection	/PROTECTION=(S:RW,O:RW,G,W)
/[NO]QUIET_POINT	/QUIET_POINT
/NORECORD	/RECORD
/RESTORE_OPTIONS=file-spec	/No options file is written
/[NO]REWIND	/NOREWIND
/[NO]SCAN_OPTIMIZATION	See description
/TAPE_EXPIRATION=absolute-time	/TAPE_EXPIRATION=now

Description

You can perform a full or incremental backup of the entire database, or a by-area backup of the storage areas. Oracle recommends using the Multithreaded Backup utility when backing up databases or specified storage areas to tape. You can use multithreaded backup to back up database or storage areas to disk; however, multithreaded backup to disk is slower then single-threaded backup to disk. See Section 9.5.1 for information on single-threaded backup to disk.

The multithreaded backup achieves high performance by reading pages from several areas and writing to several master tape drives simultaneously. A master tape drive has a controller and can control several slave drives at the same time. In addition, high-density tapes are utilized more efficiently by allocating block size according to tape density. Certain tape drives provide further performance improvements due to support for their tape cache mechanisms.

The tape characteristics of each volume are checked when attempting a multithreaded backup. If DBO encounters an error when attempting to mount the tape, either DBO terminates or the problem is reported and the operation is retried. The tape must be write enabled. If this check fails, you are asked to correct the problem, reload the tape, and indicate when you are finished.

To use the multithreaded backup command, check the setting of the authorize parameter, FILLM. This parameter sets the total number of files that can be opened at one time. Set it to at least 2 times the number of areas plus the number of master tape drives plus 1.

For example:

```
FILLM = AREAS * 2 + MASTER + 1
```

Parameters

root-file-spec
Specifies the root file of a database to be backed up. The default file type is .ROO.

backup-file-spec
Specifies the backup file for the database.

It is good practice to write backup files to a device other than the device where the root, storage area, and snapshot files of the database are located. This way, if there is a problem with the database disk, you can still restore the database from backup. However, failure to place the backup copy on another device does not constitute an error.

When you perform a backup to ANSI tapes, use a backup file name of 17 characters or less. Backup truncates longer file names because OpenVMS enforces the ANSI file name length limit. The default file type is .DBF.

Command Qualifiers

/ACCEPT_LABEL
Specifies that DBO should keep the current tape label it finds on a tape during a backup operation even if that label does not match the default label or that specified with the /LABEL qualifier. Operator notification does not occur unless the tape's protection, owner, or expiration date prohibit writing to the tape. However, a message is logged (assuming logging is enabled) and written to the backup journal file (assuming you have specified the /JOURNAL qualifier) to indicate that a label is being preserved and which drive currently holds that tape.

This qualifier is particularly useful when your backup operation employs numerous previously used (and thus labeled) tapes and you want to preserve the labels currently on the tapes.

If you do not specify this qualifier, and you are performing a backup operation to tape, the default behavior of DBO is to notify the operator each time it finds a mismatch between the current label on the tape and the default label (or the label you specify with the /LABEL qualifier).

/ACTIVE_IO=integer
Specifies the maximum number of write operations to a backup device that DBO will attempt simultaneously. This is not the maximum number of write operations in progress.

The value of the /ACTIVE_IO qualifier can range from 1 to 7. For TA90 and TA90E tape drives, the maximum value is 7 and the default is 5. For all other tape drives the maximum value is 5 and the default is 3. Values larger than 3 might improve performance when using streaming tape drives. You should start with the default of 3 and experiment to find the best setting for your configuration.

The TA90, TA90E, and TF857 tape drives and their stack loaders are supported. The TA90E data compaction function is also supported.

/AREA=area-name [,...]
/NOAREA=area-name [,...]
Specifies one or more database storage areas for by-area backup. After-image journaling must be enabled. Areas specified with the /NOAREA qualifier are not backed up. The backup is labeled as a partial backup in the file header. To incrementally back up an area include the /INCREMENTAL qualifier.

/BUFFER_SIZE=integer
Specifies the maximum record size for the backup file. The size can range from 2048 to 65,024 bytes. In general, you should not override the record size selected by multithreaded backup as it determines a size that is efficient for the tape drive being used.

/CHECKSUM_VERIFICATION
/NOCHECKSUM_VERIFICATION
Verifies the checksum stored on each database page while backing it up. This provides error detection when reading database pages. You can save significant CPU resources by specifying the /CHECKSUM_VERIFICATION qualifier only when you are experiencing disk, HSC, or CI port hardware problems.

The default is /NOCHECKSUM_VERIFICATION. When you specify this qualifier, DBO does not verify the checksum on the database pages.

/COMPRESSION
/COMPRESSION=ZLIB
/NOCOMPRESSION
Allows you to specify the compression method to use before writing data to the backup file. This may increase CPU usage during the backup, but may be justified by a smaller backup file when written as a disk file, or is being backed up over a busy network, or is being backed up to a tape drive that does not do its own compression. You probably do not want to specify the /COMPRESSION qualifier when you are backing up an file to a tape drive that does its own compression; in some cases doing so can actually result in a larger file.

If you specify the /COMPRESSION qualifier without a value, the default is /COMPRESSION=ZLIB=6.

The level value (ZLIB=level) is an integer between 1 and 9 specifying the relative compression level with one being the least amount of compression and nine being the greatest amount of compression. Higher levels of compression use increased CPU time while generally providing better compression. The default compression level of 6 is a balance between compression effectiveness and CPU consumption.

Note that the /COMPRESSION qualifier need only be used with the DBO/BACKUP /MULTITHREAD command as the DBO /RESTORE command determines whether a backup file has been compressed from information saved within the backup file.

/CRC[=option]
/NOCRC
Causes software cyclic redundancy check (CRC) coding to be computed and stored in the data blocks of the backup file.

If you do not specify this qualifier, Oracle CODASYL DBMS will check the hardware and select the appropriate checksum calculations for reasonable safety. The /CRC qualifier has two options:

- AUTODIN_II

  Uses the AUTODIN_II polynomial for 32-bit cyclic redundancy check (CRC) calculation and provides the most reliable error detection. This is the default for the NRZ/PE (800/1600 bits/inch) tape drives.

  If your CPU has the CRC instruction implemented in microcode, it will be used. If your CPU does not have the CRC instruction implemented in microcode, DBO automatically uses a high-performance emulation routine instead of the CRC instruction.

  If you retain your tapes longer than 3 years, you should always use the AUTODIN_II option for best reliability.

- CHECKSUM

  Uses one's complement addition, which is the same computation used to checksum the database pages on disk. This is the default for TA78, TA79, and TA81 tape drives. These HSC drives have adequate error-detection capability, but CI contention may cause data underruns and unrecoverable restore errors unless error detection is employed.

  The /CRC=CHECKSUM option adequately detects data underrun errors and is about six times faster than the CRC microcode instruction.

If you specify /CRC and do not specify an option, AUTODIN_II is the default. The overall effect of the /CRC=AUTODIN_II, /CRC=CHECKSUM, and /NOCRC qualifiers is to raise tape reliability to a par with disk reliability.

/DENSITY=integer
Specifies the density in bits/inch at which the output volume is written. The default is the system default for the tape drive specified.

The data compaction function of some tape drives affects the use of the /DENSITY qualifier in the following ways. If you do not specify the /NOREWIND or /DENSITY qualifiers, all the tapes are written in the compaction mode in which the first volume was initialized. If you specify /REWIND and /DENSITY=39872 or /DENSITY=4000, the tapes are written not compacted regardless of how they were originally initialized. If you specify /REWIND and /DENSITY=7000, the tapes are written compacted regardless of how they were originally initialized.

/ENCRYPT=({VALUE=|NAME=}[,ALGORITHM=])
The /ENCRYPT qualifier specifies the encryption key and algorithm so that the DBO/BACKUP /MULTITHREAD can encrypt the database backup.

Specify a key value as a string or, the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters see HELP ENCRYPT.

This feature requires the OpenVMS Encrypt product to be installed and licensed on this system.

Encryption Key

If you cannot remember the encryption key you have effectively lost all data in the encrypted file. The encryption key used on the backup command will be REQUIRED on the RESTORE command of that backup file. It is also used by the DBO/DUMP/AFTER_IMAGE command.

/EXTEND_QUANTITY=number-blocks
Sets the size, in blocks, by which the backup file can be extended. The minimum value for the number-blocks parameter is 1; the maximum value is 65535. This qualifier is only valid for backup operations to disk. Using this qualifier with a backup operation to tape generates an error. The default extend size is 2048 blocks.

/GROUP_SIZE=interval
/NOGROUP_SIZE
Specifies the number of blocks to place in each redundancy group. There is an XOR recovery block for each redundancy group. The group size can be from 0 to 100. Specifying a group size of 0 or specifying /NOGROUP_SIZE results in no XOR recovery blocks being written. The default is 10.

This qualifier can be used only for backup to tape.

/IDENTIFIER=option
Specifies the owner of the tapes. The owner is the user who is permitted to restore the database.

Specify UICs in the standard OpenVMS UIC format, which allows alphanumeric group and member values. Wildcards are allowed in both the group and member fields. Specify OpenVMS identifiers in the standard format, which allows 1 to 31 alphanumeric characters and contains at least one nonnumeric character. Refer to the OpenVMS system management documentation and the OpenVMS System Services documentation for more information.

The default for /IDENTIFIER is the user identifier of the process that performed the DBO/BACKUP/MULTITHREAD command.

/INCREMENTAL=option
Specifies an incremental backup be performed. Valid options are:

- BY_AREA

  Specifies an incremental by-area backup be performed. This option must be used in conjunction with the /AREA qualifier to specify the areas for incremental backup. Only those database pages that have changed since the last by-area backup are backed up. The following message is displayed when an incremental backup does not include all areas:

  ```
  %DBO-I-NOTALLARE, Not all areas will be included in this backup file
  ```

- COMPLETE

Specifies an incremental backup of the entire database be performed. Only those database pages that have changed since the last full backup of the database are backed up. This is the default.

Used with the /AREA qualifier, it will back up the database pages changed since the last full backup for the area specified.

/LABEL=label ([,...])
Specifies the 1- to 6-character string with which the volumes of the backup file are labeled. You can specify a list of tape labels for multiple type volumes.

This qualifier can be used only with the /REWIND qualifier for backup to tape.

If the volume label disagrees with the label specified, you receive an error message. If the tape is not the first volume or if you specified the /REWIND qualifier in the command, DBO checks the owner, file protection, and expiration date of the tape and issues an error message if a problem was found.

If any of these checks fails, you are asked what you want to do with the tape volume. You must select one of the following options:

- QUIT

  This option cancels the operation and causes you to exit from DBO.

- RETRY

  This option repeats the checks after dismounting and remounting the same tape.

- UNLOAD

  This option dismounts and unloads the tape so you can load the correct tape on the drive.

- INITIALIZE [AS label-name]

  This option requests that the tape be rewound and relabeled.

The DBO/BACKUP/MULTITHREAD command works correctly with unlabeled or nonstandard formatted tapes when the /REWIND qualifier is specified. However, tapes that have never been used or initialized and nonstandard tapes sometimes produce errors that make OpenVMS mount attempts fail repeatedly. In this situation, DBO cannot continue until you use the OpenVMS INITIALIZE command to correct the error.

/LOCK_TIMEOUT=integer
Specifies the length of time the database should wait before returning a lock timeout message. The value expressed represents the number of seconds.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request log output and /NOLOG to prevent it. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.) If you use DBO/BACKUP/MULTITHREAD in BATCH mode, fatal errors print to the log file. It is especially important to check the log file for errors. If the backup terminates due to a fatal error, then the backup is partial and thus corrupt.

/MASTER
Specifies a tape drive to be a MASTER tape drive. This is a positional qualifier specified with a tape drive. When /MASTER is used, it must be used on the first tape drive specified. When specified, all additional tape drives become a slave for that tape drive until the end of the command line, or until the next /MASTER qualifier, whichever comes first.

The /MASTER qualifier allows users to specify which tape drives are master tape drives and which tape drives are slaves. By specifying this, the user eliminates any uncertainty about which drive will be the master and which drives will be slaves. However, if the /MASTER qualifier is used on a tape drive that is not physically a master tape drive, then the output performance of BACKUP will decrease.

/MEDIA_LOADER
/NOMEDIA_LOADER
Use the /MEDIA_LOADER qualifier to specify that the tape device receiving the backup file has a loader or stacker. Use the /NOMEDIA_LOADER qualifier to specify that the tape device does not have a loader or stacker.

If a tape device has a loader or stacker and DBO detects its presence, DBO uses the loader or stacker, if needed, during the backup operation. However, occasionally DBO does not detect that a tape device has a loader or stacker. Therefore, when the first backup tape fills, DBO issues a request to the operator for the next tape, instead of requesting the next tape from the loader or stacker.

If you find that DBO is not detecting that your tape device has a loader or stacker, specify the /MEDIA_LOADER qualifier. If you find that DBO expects a loader or stacker when it should not, specify the /NOMEDIA_LOADER qualifier.

/ONLINE
/NOONLINE
Allows full, incremental, or by-area database backups without the need to shut down the database. To use /ONLINE, you must allow and enable snapshot files for all storage areas. When you specify the /ONLINE qualifier, users can continue to update the database during the procedure. The backup procedure uses the snapshot file to retrieve the previous version of any record that has been modified since the backup procedure began.

If you use /NOONLINE, the default, users cannot be bound to the database. The offline backup process has EXCLUSIVE access to the database and does not require snapshot files to work. To use /NOONLINE, you do not need to allow or enable snapshots.

/PROTECTION=file-protection
Specifies the file protection for tapes. If you do not specify the /PROTECTION qualifier, the backup file receives the default protection of S:RW,O:RW,G,W.

The protection you assign to tapes can be different than the protection you assign to OpenVMS files on disk. Tapes do not allow DELETE or EXECUTE access, and SYSTEM and OWNER always receive READ and WRITE access. Also, a more restrictive class receives the access of less restrictive classes. For example, if you specify /PROTECTION=(S,O,G:W,W:R), that protection on tape becomes (S:RW,O:RW,G:RW,W:R).

For tapes, access is written and checked at the volume level. Therefore, unless you specify /REWIND on the backup, the volume label is not written on the first tape, and the first tape retains its original protection. The volume labels on subsequent volumes are always written, so they always get the specified protection and identifier. Tapes do not allow changes of the volume labels without losing everything previously written to the tape. If you want to append the backup to an existing tape volume, you must get the identifier and protection from OpenVMS and specify it in the BACKUP command. Similarly, if you fail to specify /REWIND, the restore does not check the volume label on the first tape unless the tape is at beginning-of-tape (BOT).

/QUIET_POINT
/NOQUIET_POINT
Specifies whether or not to require a database quiet point before the online backup process can begin. /QUIET_POINT is the default.

/RECORD
/NORECORD
The /RECORD qualifier is set by default. Using the /NORECORD qualifier allows you to avoid the modification of the database with recent backup information. Hence the database appears as if it had not been backed up at this time.

The main purpose of this qualifier is to allow a backup of a Hot Standby database without modifying the database files.

The /NORECORD qualifier can be negated with the /RECORD qualifier.

/RESTORE_OPTIONS=file-spec
Generates an options file designed to be used with the /OPTIONS qualifier of the DBO/RESTORE command. The file generated by the /RESTORE_OPTIONS qualifier references:

- All storage areas for a full backup operation
- Only those storage areas specified for a by-area backup operation

By default, a /RESTORE_OPTIONS qualifier is not created. If you specify the /RESTORE_OPTIONS qualifier and a file specification, but not a file type, DBO uses the file type .OPT, by default.

/REWIND
/NOREWIND
Specify the /REWIND qualifier to have the tape rewound and initialized before processing begins.

Specify the /NOREWIND qualifier to have the backup procedure write to the tape starting at the end of the current volume. This is the default.

This qualifier can be used only for backup to tape.

/SCAN_OPTIMIZATION
/NOSCAN_OPTIMIZATION
Specifies whether or not DBO should employ scan optimizations during incremental backup operations. This feature is referred to as fast incremental backup.

If this feature is disabled, an incremental backup requires that every page of every area be read to determine if the page has been updated since the last full backup. The fast incremental backup feature provides an algorithm that identifies pages that have not been updated without reading and processing each page. This information is stored on each SPAM page. This provides a substantial performance improvement when database activity is sufficiently low.

However, there is a cost in recording this information in the database. In some circumstances the cost might be too high, particularly if you do not intend to use incremental backup operations.

The SCAN_OPTIMIZATION qualifier has different effects, depending on the type of backup operation you perform. In brief, you can enable or disable the scan optimization setting only when you issue a full offline backup command (or by using the DBO/CREATE or DBO/MODIFY command with the /BACKUP_CONTEXT=SCAN_OPTIMIZATION qualifier), and you can specify whether to use the data produced by a scan optimization only when you issue an incremental backup command. An online incremental backup command can be issued with the /SCAN_OPTIMIZATION qualifier.

The following list describes this behavior in more detail:

- Incremental backup operation

  If the /NOSCAN_OPTIMIZATION qualifier is supplied, the optimization will not be used, whether or not the database is recording the updated regions. If the /SCAN_OPTIMATION qualifier is supplied, or if the qualifier is omitted, the optimization will only be used when the recording of updated regions is being performed.

- Full backup operation off line

  If the /SCAN_OPTIMIZATION qualifier is supplied, the recording of updated regions will be enabled for this database. If the /NOSCAN_OPTIMIZATION qualifier is supplied, the recording of updated regions will be disabled for this database. If the qualifier is not supplied, the recording state remains unchanged.

- Full backup operation on line

  The /[NO]SCAN_OPTIMIZATION qualifier is ignored and an informational message to this effect is returned.

The DBO/DUMP/HEADER command reports whether or not scan optimization is enabled.

/TAPE_EXPIRATION=absolute-time

Specifies the expiration date and time for tapes. The default for this qualifier is NOW (the current time), meaning that the tape can be overwritten immediately.

There are many valid forms for absolute time specifications. The OpenVMS user documentation describes absolute time specification in detail. The following form is sufficient for most purposes:

DD-MMM-YYYY

Specifies the time 00:00 of the specified date. For example, 12-APR-1990 specifies that the backup file can be overwritten without question starting on that day.

Examples

1.
```
$ INIT $111$MUA2: PARTS
$ MOUNT/FOREIGN $111$MUA2: PARTS
$ DBO/BACKUP/MULTITHREAD/LOG -
_$ DISK1:[DB]PARTS $111$MUA2:PARTS

%DBO-I-BCKTXT_01, Thread 1 uses devices $111$MUA2:
%DBO-I-BCKTXT_08, Thread 1 was assigned file DISK1:[DB]MAKE.DBS;1
%DBO-I-BCKTXT_08, Thread 1 was assigned file DISK1:[DB]BUY.DBS;1
%DBO-I-BCKTXT_08, Thread 1 was assigned file
  DISK1:[DB]MARKET.DBS;1
%DBO-I-BCKTXT_08, Thread 1 was assigned file
  DISK1:[DB]PERSONNEL.DBS;1
%DBO-I-BCKTXT_00, Backed up root file DISK1:[PARTS]PARTS.ROO;1
%DBO-I-BCKTXT_02, Full backup of storage area DISK1:[DB]MAKE.DBS;1
%DBO-I-BCKTXT_02, Full backup of storage area DISK1:[DB]BUY.DBS;1
%DBO-I-BCKTXT_02, Full backup of storage area
  DISK1:[DB]MARKET.DBS;1
%DBO-I-BCKTXT_02, Full backup of storage area
  DISK1:[DB]PERSONNEL.DBS;1
%DBO-I-BCKTXT_02, Full backup of storage area DISK1:[DB]MAKE.DBS;1
%DBO-I-BCKTXT_04,     ignored 1 space management page
%DBO-I-BCKTXT_07,     backed up 100 data pages
%DBO-I-BCKTXT_02, Full backup of storage area DISK1:[DB]BUY.DBS;1
%DBO-I-BCKTXT_04,     ignored 1 space management page
%DBO-I-BCKTXT_07,     backed up 100 data pages
%DBO-I-BCKTXT_02, Full backup of storage area
  DISK1:[DB]MARKET.DBS;1
%DBO-I-BCKTXT_04,     ignored 1 space management page
%DBO-I-BCKTXT_07,     backed up 100 data pages
%DBO-I-BCKTXT_02, Full backup of storage area
  DISK1:[DB]PERSONNEL.DBS;1
%DBO-I-BCKTXT_04,     ignored 1 space management page
%DBO-I-BCKTXT_07,     backed up 100 data pages
```

This command performs a full backup of the PARTS database.

2.
```
$ DBO/BACKUP/MULTITHREAD PARTS MUA0:PARTS/MASTER,-
_$ MUA1:,MUA2:,MUA3:/MASTER,MUA4:
```

In this example, there will be two output threads. One will be MUA0: (with MUA1: as its slave), and the other will be MUA3: (with MUA4: as its slave).

3.
```
$ DBO/BACKUP/MULTITHREAD PARTS -
_$ MUA0:PARTS/MASTER,MUA1:,MUA2:,MUA3:,MUA4:
```

In this example, there will be one output thread, MUA0:, with four slaves (MUA1: through MUA4:).

4.
```
$ DBO/BACKUP/MULTITHREADED PARTS PARTS.DBF -
_$ /AREA=(MAKE, BUY) /RESTORE_OPTIONS=BYAREA.OPT
%DBO-I-NOTALLARE, Not all areas will be included in
  this backup file
$ TYPE BYAREA.OPT
!  Options file for database DISK5:[DBUSER]PARTS.ROO;1
!  Created  4-NOV-1996 14:05:29.79
! Created by BACKUP command
```

```
MAKE -
        /file=DISK1:[AREA]MAKE.DBS;1 -
        /blocks_per_page=2 -
        /extension=ENABLED -
        /read_write -
        /space -
        /thresholds=(70,85,95) -
        /snapshot=(allocation=102, -
                   file=DISK2:[SNAPS]MAKE.SNP;1)

BUY -
        /file=DISK3:[AREA]BUY.DBS;1 -
        /blocks_per_page=2 -
        /extension=ENABLED -
        /read_write -
        /space -
        /thresholds=(70,85,95) -
        /snapshot=(allocation=102, -
                   file=DISK4:[SNAPS]BUY.SNP;1)

$ DBO/RESTORE/MULTITHREADED -
_$ PARTS.DBF/AREA/OPTIONS=BYAREA.OPT
```

This example demonstrates the use of the /RESTORE_OPTIONS qualifier. The first command backs up selected areas of the PARTS database and creates an options file. The second command shows the contents of the options file. The last command demonstrates the use of the options file with the DBO/RESTORE/MULTITHREADED command.

5.
```
$ DBO/BACKUP/MULTITHREADED PARTS PARTS.DBF -
_$ /SCAN_OPTIMIZATION
$ DBO/BACKUP/MULTITHREADED/AREA=MAKE/INCREMENTAL -
_$ /SCAN_OPTIMIZATION PARTS MAKE_ONLY.DBF
%DBO-I-NOTALLARE, Not all areas will be included in this backup file
```

This example shows two commands. In the first command, the recording of data for fast incremental backup is enabled with the /SCAN_OPTIMIZATION qualifier. In the second command, the data recorded since the full backup operation is used to perform the incremental backup operation.

6.
```
$ encrypt/create_key/aes HAMLET "And you yourself shall keep the key of it"
$ dbo/backup/multi/encrypt=(algorithm=AES,name:HAMLET)/log -
_$ PARTS.ROO prt.dbb_encv
%DBO-I-BCKTXT_00, Backed up root file DEV1:[DEV.DBMS_TESTING]PARTS.ROO;1
%DBO-I-BCKTXT_02, Starting full backup of storage area (PERSONNEL)
 DEV1:[DEV.DBMS_TESTING]PERSONNEL.DBS;1 at 13-MAY-2022 21:06:35.09
%DBO-I-BCKTXT_02, Starting full backup of storage area (MARKET)
 DEV1:[DEV.DBMS_TESTING]MARKET.DBS;1 at 13-MAY-2022 21:06:35.10
%DBO-I-BCKTXT_02, Starting full backup of storage area (BUY)
 DEV1:[DEV.DBMS_TESTING]BUY.DBS;1 at 13-MAY-2022 21:06:35.10
%DBO-I-BCKTXT_02, Starting full backup of storage area (MAKE)
 DEV1:[DEV.DBMS_TESTING]MAKE.DBS;1 at 13-MAY-2022 21:06:35.10
%DBO-I-BCKTXT_12, Completed full backup of storage area (PERSONNEL)
 DEV1:[DEV.DBMS_TESTING]PERSONNEL.DBS;1 at 13-MAY-2022 21:06:35.12
%DBO-I-BCKTXT_12, Completed full backup of storage area (MARKET)
 DEV1:[DEV.DBMS_TESTING]MARKET.DBS;1 at 13-MAY-2022 21:06:35.14
%DBO-I-BCKTXT_12, Completed full backup of storage area (BUY)
 DEV1:[DEV.DBMS_TESTING]BUY.DBS;1 at 13-MAY-2022 21:06:35.14
%DBO-I-BCKTXT_12, Completed full backup of storage area (MAKE)
 DEV1:[DEV.DBMS_TESTING]MAKE.DBS;1 at 13-MAY-2022 21:06:35.15
%DBO-I-COMPLETED, BACKUP operation completed at 13-MAY-2022 21:06:35.15
$
```

This example shows the use of the ENCRYPT DCL command to define a process

wide name to be used by the DBO commands. This name is specified on the DBO/BACKUP /ENCRYPT qualifier to generate an encrypted (secure) backup of the full database. Ensure that the encryption key is preserved and the association with this backup file is known. Without there is no way to use DBO/RESTORE or DBO/DUMP/BACKUP.

9.6 DBO/CACHE Commands

There are three forms of the DBO/CACHE command, each having its own qualifiers. These are:

DBO/CACHE/ADD root-file-spec cache-name [,...]

This format adds the specified record caches to the specified database.

DBO/CACHE/DELETE root-file-spec cache-name [,...]

This format deletes the specified record caches from the specified database.

DBO/CACHE/MODIFY root-file-spec cache-name [,...]

This format modifies the specified record caches in the specified database.

These formats are described in the following sections.

9.6.1 DBO/CACHE/ADD Command

Defines record caches for the specified database. A cache definition includes a record cache name, the size of the record cache (determined by the allocation and length of the cache), where in memory the record cache will reside, the record replacement strategy, and parameters for the cache backing store file.

Format

DBO/CACHE/ADD root-file-spec cache-name[,...]

Command Qualifier	Default
/[NO]LOG	/NOLOG

Command or Area Qualifiers	Defaults
/CHECKPOINT=(option[,...])	See description
/FILE=(option[,...])	See description
/LARGE_MEMORY	/NOLARGE_MEMORY
/MEMORY_MAPPING=(option[,...])	/MEMORY_MAPPING=PROCESS
/[NO]REPLACEMENT	/REPLACEMENT
/SLOTS=(option[,...])	See description
/WINDOW_COUNT=n	/WINDOW_COUNT=10

Description

A record cache is a section of globally accessible memory that contains copies of records. Record caching provides the ability to store frequently accessed records in memory, reducing disk I/O. The records remain in memory even when the associated page has been transferred (flushed) back to disk. Record caching has the following advantages:

- Reduced database page read and write operations
- Improved response time
- Much lower overhead to access a record in a record cache than for a page in a global or local buffer
- Shorter code path when a record is found in the record cache
- Efficient use of system resources (memory) for shared data

Adding a record cache requires that a slot for that cache be reserved in the database. By default, one slot is reserved in the database when it is created. If you want to add more than one record cache, use the DBO/MODIFY/RESERVE=CACHE command. See Section 9.20.1 for details.

Interchangeable Terms: Record and Row

Often the term *row* cache or caching will be used instead of *record* cache or caching. The meaning is the same; these terms are interchangeable.

Parameters

root-file-spec
Specifies the root file of the database to which you want to add one or more record caches.

cache-name[,...]
Specifies the name or names to be assigned to the caches you are creating.

Command Qualifiers

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Use the /LOG qualifier to display the name of each cache as it is added to the database; use the /NOLOG qualifier to prevent this display. The /NOLOG qualifier is the default.

Command or Area Qualifiers

/CHECKPOINT=(option[,...])
The /CHECKPOINT qualifier specifies the source records and target for checkpoint operations for the row cache. If the keyword ALL_ROWS is specified, then all rows in the cache are written during the checkpoint process.

If the keyword UPDATED_ROWS is specified, then just the added or the updated rows are written during the checkpoint process.

If the target of the checkpoint operation is BACKING_FILE, then the RCS process writes the row cache entries to the backing (.DBC) files. If the target is DATABASE, then the added or updated rows (only UPDATED_ROWS is allowed) are written back to the database. Upon recovery from a node failure, the database recovery process has no data on the contents of the row cache. Therefore, it does not repopulate the row caches in memory.

This /CHECKPOINT qualifier overrides the database-level CHECKPOINT clause.

The /CHECKPOINT qualifier options are as follows:

- ALL_ROWS

  The options ALL_ROWS and UPDATED_ROWS are mutually exclusive. The options ALL_ROWS and DATABASE are mutually exclusive.

- UPDATED_ROWS

  The options UPDATED_ROWS and ALL_ROWS are mutually exclusive.

- DATABASE

  The options DATABASE and BACKING_FILE are mutually exclusive. The options DATABASE and ALL_ROWS are mutually exclusive.

- BACKING_FILE

  The options BACKING_FILE and DATABASE are mutually exclusive.

/FILE=(option[,...])

Provides options that allow you to specify the directory, allocation, and extension of the cache backing store file. Oracle CODASYL DBMS writes to the cache backing store file when the row cache server (RCS) process checkpoints. Oracle CODASYL DBMS automatically generates a file name with a file extension of .DBC.

The /FILE qualifier options are as follows:

- DIRECTORY=directory-spec

  Specifies the directory location for the cache backing store file for the specified record cache. By default, this file is placed in the same directory as the database root file.

  The database system generates a file name automatically for each row cache at checkpoint time. Specify a device name and directory name only.

- ALLOCATION=n

  Specifies the initial size of the .DBC file. The default allocation is 40 percent of the record cache size. (The cache size is determined by multiplying the number of rows in the cache by the row length.)

  This clause is ignored if the row cache is defined to checkpoint to the database.

- EXTENSION=n

  Specifies the number of pages by which the .DBC file can be extended after the initial allocation is reached. The default extension is 127 multiplied by the number of rows in the cache.

  This clause is ignored if the row cache is defined to checkpoint to the database.

/[NO]LARGE_MEMORY

This qualifier specifies whether or not large memory is used to manage the row cache. Very large memory (VLM) allows Oracle CODASYL DBMS to use as much physical memory as is available. It provides access to a large amount of physical memory through small virtual address windows.

Use LARGE_MEMORY only when both of the following are true:

- You have enabled row caching.

- You want to cache large amounts of data, but the cache does not fit in the virtual address space.

The default is /NOLARGE_MEMORY.

/MEMORY_MAPPING=(option[,...])
Specifies whether cache global sections are created in system space or process space. The default is PROCESS.

When you use cache global sections created in the process space, you and other users share physical memory and the OpenVMS operating system maps a row cache to a private address space for each user. As a result, all users are limited by the free virtual address range and each uses a percentage of memory in overhead. If many users are accessing the database, the overhead can be high.

When many users are accessing the database, consider using SYSTEM. This gives users more physical memory because they share the system space memory and there is none of the overhead associated with the process space memory.

A description of the available options is provided in the following list:

– PROCESS

  Specifies that you want the record cache created in the process global section. When you use global sections created in the process space, you and other users share physical memory. The OpenVMS operating system maps a record cache to a private address space for each user. This is the default MEMORY_ MAPPING option.

– SYSTEM

  Specifies that you want the record cache created in the system space global section in the OpenVMS system space. This means that the system space global section is fully resident, or pinned, in memory; it does not affect the quotas of the working set of a process.

  System space is critical to the overall system. System space buffers are not paged; therefore, they use physical memory, reducing the amount of physical memory available for other system tasks. This may be an issue if your system is constrained by memory.

  Consider allocating small caches that contain heavily accessed data in system space buffers. When a row cache is stored in a system space buffer, there is no process overhead and data access is very fast because the data does not need to be mapped to user windows. The Hot Row Information screen in the DBO/SHOW STATISTICS utility displays a list of the most frequently accessed rows for a specific row cache.

– RESIDENT

  {{

/REPLACEMENT
/NOREPLACEMENT
Enables row replacement for a particular cache; the /NOREPLACEMENT qualifier disables row replacement for a particular cache.

These qualifiers provide you with some control over what happens when a row cache becomes full. If row replacement is enabled for a particular row cache, then new rows replace the oldest, unused, unmarked rows when the cache is full. If row replacement is disabled, then new rows are not placed in the cache when the cache is full; they will always be retrieved from disk.

Row replacement is enabled by default.

/SLOTS=(option[,...])

Allows you to set parameters for the record caches you are adding. Specify these parameters with the following options:

- COUNT=n

  Specifies the maximum number of records that can be stored in a cache. As the record cache grows, more recently referenced records are retained in the record cache area and those not referenced recently are discarded.

  If you do not specify the COUNT option, Oracle CODASYL DBMS sets the default count to 1000. The minimum value is 1 and the maximum value is 2 billion.

- LENGTH=n

  Specifies the length in bytes of each record allocated to the record cache. Records are not cached if they are too large for the record cache area. The maximum length you can specify for a record in the record cache area is 65,535 bytes.

  If you do not specify the LENGTH option, Oracle CODASYL DBMS sets the length to 256 by default. The minimum value you can specify for *n* is 16 and the maximum value is 65535.

  The value specified will be rounded up to the next multiple of 4 bytes to ensure LONGWORD alignment.

- SWEEP_COUNT=n

  Specifies the number of added or modified cache rows that will be written back to the database to make space available in the row cache for subsequent transactions that insert rows into the caches. Oracle Corporation recommends that you initially specify the number of sweep rows to be between 10 and 30 percent of the total number of rows in the cache, then monitor performance and adjust the number of sweep rows if necessary. The default setting is 3000 rows.

- RESERVE_COUNT=n

  Specifies the maximum number of cache rows that each user can reserve. The default is 20 rows.

  The RESERVE_COUNT clause is used also when searching for available slots in a row cache. The entire row cache is not searched on the initial pass. This clause is used as the maximum number of rows that are searched for an available slot. If at least one available slot is found, the insert operation can proceed. If no available slots are found in this initial search, Oracle CODASYL DBMS continues searching through the cache until it finds one.

  If you do not specify the RESERVE_COUNT option, Oracle CODASYL DBMS sets the RESERVE_COUNT to 20 by default. The minimum value you can specify for *n* is 0 and the maximum value is 65535.

- WORKING_SET_COUNT=n

  Specifies the number of records per process that are not eligible for record replacement (when record replacement is enabled).

  If you do not specify the WORKING_SET_COUNT option, Oracle CODASYL DBMS sets the working set to 10 by default. The minimum value you can specify for *n* is 1 and the maximum value is 100.

Examples

1.
```
$ DBO/CACHE/ADD WKD1:PARTS.ROO CACHE_1 -
_$ /SLOTS=(ALLOCATION=500, LENGTH=512, RESERVE_COUNT=100,WORKING_SET=20) -
_$ /REPLACEMENT/MEMORY_MAPPING=(PROCESS) -
_$ /FILE=(DIR=DISK1:[BACKING_STORE], ALLOC=200, EXTEN=200), CACHE_2
```

   This example first adds two record cache areas to the PARTS database. Cache area CACHE_1 is given the following settings:

   - The cache is allocated 500 slots.
   - The largest record that can be stored in a slot is 512 bytes.
   - The maximum number of slots that can be reserved per process is 100 and each process will have a working set of 20.
   - Record replacement is enabled.
   - The cache will be stored in the process global section.
   - The cache backing store files (.DBC) will be placed in the DISK1:[BACKING_STORE] directory with an initial allocation of 200 blocks that can be extended in 200-block increments.

   Cache area CACHE_2 accepts the default settings for all of these attributes.

   This example assumes that at least one record cache slot was reserved (in addition to the one default slot reserved when the database was created). Use the DBO/CREATE/RESERVE=CACHE or DBO/MODIFY/RESERVE=CACHE commands to reserve record cache slots.

   The next step is to link these row caches to a storage area with the DBO/MODIFY/CACHE command. See Section 9.20.1 for details.

9.6.2 DBO/CACHE/DELETE Command

Deletes a record cache previously added to the database.

Format

DBO/CACHE/DELETE root-file-spec cache-name[,...]

Command Qualifiers	Defaults
/[NO]LOG	/NOLOG
/[NO]RESTRICT	/NORESTRICT

Description

The DBO/CACHE/DELETE command deletes an existing cache for a database, and if necessary, disassociates it from the area to which it is related. Executing this command requires exclusive access to the database.

Parameters

root-file-spec
Specifies the database root file from which the specified record cache or caches are to be deleted.

cache-name[,...]
Specifies the record cache or caches that you want to delete from the specified database.

Command Qualifiers

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Use the /LOG qualifier to display the name of each cache as it is deleted from the database; use the /NOLOG qualifier to prevent this display. The /NOLOG qualifier is the default.

/RESTRICT
/NORESTRICT
When /RESTRICT is specified Oracle CODASYL DBMS will abort if any storage area uses the cache. Otherwise, the cache will be deleted and any storage areas will be updated to remove reference to the cache.

Examples

1. `$ DBO/CACHE/DELETE PARTS CACHE_2`

   This command deletes the record cache named CACHE_2 from the PARTS database.

9.6.3 DBO/CACHE/MODIFY Command

Allows you to modify an existing record cache definition.

Format

DBO/CACHE/MODIFY root-file-spec cache-name[,...]

Command Qualifier	Default
/[NO]LOG	/NOLOG

Command or Area Qualifiers	Defaults
/CHECKPOINT=(option[,...])	See description
/FILE=(option[,...])	
/LARGE_MEMORY	/NOLARGE_MEMORY
/MEMORY_MAPPING=(option[,...])	
/[NO]REPLACEMENT	/REPLACEMENT
)	
/SLOTS=(option[,...])	
/WINDOW_COUNT=n	/WINDOW_COUNT=10

Description

The DBO/CACHE/MODIFY command allows you to modify characteristics of a cache definition, such as the size of the record cache (determined by the allocation and length of the cache), where in memory the record cache will reside, the record replacement strategy, and parameters for the cache backing store file.

Parameters

root-file-spec
Specifies the root file of the database to which you want to modify one or more record caches.

cache-name[,...]
Specifies the name or names of the caches you want to modify.

Command Qualifiers

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Use the /LOG qualifier to display the name of each cache as it is modified; use the /NOLOG qualifier to prevent this display. The /NOLOG qualifier is the default.

Command or Area Qualifiers

/CHECKPOINT=(option[,...])
The /CHECKPOINT qualifier specifies the source records and target for checkpoint operations for the row cache. If the keyword ALL_ROWS is specified, then all rows in the cache are written during the checkpoint process.

If the keyword UPDATED_ROWS is specified, then just the added or the updated rows are written during the checkpoint process.

If the target of the checkpoint operation is BACKING_FILE, then the RCS process writes the row cache entries to the backing (.DBC) files. If the target is DATABASE, then the added or updated rows (only UPDATED_ROWS is allowed) are written back to the database. Upon recovery from a node failure, the database recovery process has no data on the contents of the row cache. Therefore, it does not repopulate the row caches in memory.

This /CHECKPOINT qualifier overrides the database-level CHECKPOINT clause.

The /CHECKPOINT qualifier options are as follows:

- ALL_ROWS

  The options ALL_ROWS and UPDATED_ROWS are mutually exclusive. The options ALL_ROWS and DATABASE are mutually exclusive.

- UPDATED_ROWS

  The options UPDATED_ROWS and ALL_ROWS are mutually exclusive.

- DATABASE

  The options DATABASE and BACKING_FILE are mutually exclusive. The options DATABASE and ALL_ROWS are mutually exclusive.

- BACKING_FILE

  The options BACKING_FILE and DATABASE are mutually exclusive.

/FILE=(options)
Provides options that allow you to modify the directory, allocation, and extension of the cache backing store file. Oracle CODASYL DBMS writes to the cache backing store file when the row cache server (RCS) process checkpoints. Oracle

CODASYL DBMS automatically generates a file name with a file extension of .DBC.

The /FILE qualifier options are as follows:

- DIRECTORY=directory-spec

  Specifies the directory location for the cache backing store file for the specified record cache. By default, this file is placed in the same directory as the database root file.

  The database system generates a file name automatically for each row cache at checkpoint time. Specify a device name and directory name only.

- ALLOCATION=n

  Specifies the initial size of the .DBC file. The default allocation is 40 percent of the record cache size. (The cache size is determined by multiplying the number of rows in the cache by the row length.)

  This clause is ignored if the row cache is defined to checkpoint to the database.

- EXTENSION=n

  Specifies the number of pages by which the .DBC file can be extended after the initial allocation is reached. The default extension is 127 multiplied by the number of rows in the cache.

  This clause is ignored if the row cache is defined to checkpoint to the database.

/[NO]LARGE_MEMORY

This qualifier specifies whether or not large memory is used to manage the row cache. Very large memory (VLM) allows Oracle CODASYL DBMS to use as much physical memory as is available. It provides access to a large amount of physical memory through small virtual address windows.

Use LARGE_MEMORY only when both of the following are true:

- You have enabled row caching.
- You want to cache large amounts of data, but the cache does not fit in the virtual address space.

The default is /NOLARGE_MEMORY.

/MEMORY_MAPPING=(option[,...])

Specifies whether cache global sections are created in system space or process space. The default is PROCESS.

When you use cache global sections created in the process space, you and other users share physical memory and the OpenVMS operating system maps a row cache to a private address space for each user. As a result, all users are limited by the free virtual address range and each uses a percentage of memory in overhead. If many users are accessing the database, the overhead can be high.

When many users are accessing the database, consider using SYSTEM. This gives users more physical memory because they share the system space memory and there is none of the overhead associated with the process space memory.

A description of the available options is provided in the following list:

– PROCESS

Specifies that you want the record cache re-created in the process global section. When you use global sections created in the process space, you and other users share physical memory. The OpenVMS operating system maps a record cache to a private address space for each user. This is the default MEMORY_MAPPING option.

– SYSTEM

Specifies that you want the record cache re-created in the system space global section in the OpenVMS system space. This means that the system space global section is fully resident, or pinned, in memory; it does not affect the quotas of the working set of a process.

System space is critical to the overall system. System space buffers are not paged; therefore, they use physical memory, reducing the amount of physical memory available for other system tasks. This may be an issue if your system is constrained by memory.

Consider allocating small caches that contain heavily accessed data in system space buffers. When a row cache is stored in a system space buffer, there is no process overhead and data access is very fast because the data does not need to be mapped to user windows. The Hot Row Information screen in the DBO/SHOW STATISTICS utility displays a list of the most frequently accessed rows for a specific row cache.

– RESIDENT

{{

/REPLACEMENT
/NOREPLACEMENT

Enables row replacement for a particular cache; the /NOREPLACEMENT qualifier disables row replacement for a particular cache.

These qualifiers provide you with some control over what happens when a row cache becomes full. If row replacement is enabled for a particular row cache, then new rows replace the oldest, unused, unmarked rows when the cache is full. If row replacement is disabled, then new rows are not placed in the cache when the cache is full; they will always be retrieved from disk.

/SLOTS=option[,...]

Allows you to reset parameters for the record caches you are modifying. Specify these parameters with the following options:

- COUNT=n

  Specifies the maximum number of records that can be stored in a cache. As the record cache grows, more recently referenced records are retained in the record cache area and those not referenced recently are discarded. The minimum value is 1 and the maximum value is 2 billion.

- LENGTH=n

  Specifies the length in bytes of each record allocated to the record cache. Records are not cached if they are too large for the record cache area. The maximum length you can specify for a record in the record cache area is 65,535 bytes.

  The minimum value you can specify for *n* is 16 and the maximum value is 65535.

  The value specified will be rounded up to the next multiple of 4 bytes to ensure LONGWORD alignment.

- SWEEP_COUNT=n

  Specifies the number of added or modified cache rows that will be written back to the database to make space available in the row cache for subsequent transactions that insert rows into the caches. Oracle Corporation recommends that you initially specify the number of sweep rows to be between 10 and 30 percent of the total number of rows in the cache, then monitor performance and adjust the number of sweep rows if necessary. The default setting is 3000 rows.

- RESERVE_COUNT=n

  Specifies the maximum number of cache rows that each user can reserve. The default is 20 rows.

  The RESERVE_COUNT clause is used also when searching for available slots in a row cache. The entire row cache is not searched on the initial pass. This clause is used as the maximum number of rows that are searched for an available slot. If at least one available slot is found, the insert operation can proceed. If no available slots are found in this initial search, Oracle CODASYL DBMS continues searching through the cache until it finds one.

  If you do not specify the RESERVE_COUNT option, Oracle CODASYL DBMS sets the RESERVE_COUNT to 20 by default. The minimum value you can specify for *n* is 0 and the maximum value is 65535.

- WORKING_SET_COUNT=n

  Specifies the number of records per process that are not eligible for record replacement (when record replacement is enabled).

  If you do not specify the WORKING_SET_COUNT option, Oracle CODASYL DBMS sets the working set to 10 by default. The minimum value you can specify for *n* is 1 and the maximum value is 100.

Example

```
$ DBO/CACHE/MODIFY PARTS CACHE_1 -
_$ /SLOTS=(COUNT=1000, LENGTH=1024)
```

This command increases the number of slots and the length of each slot in the record cache named CACHE_1.

9.7 DBO/CHECKPOINT Command

When fast commit is enabled, requests that each active database process (on each node) flush updated database pages from its buffer pool to disk.

Format

DBO/CHECKPOINT root-file-spec

Command Qualifier	Default
/[NO]WAIT	/WAIT

9.7 DBO/CHECKPOINT Command

Description

The DBO/CHECKPOINT command is useful only if the database fast commit feature has been enabled. If the fast commit feature is disabled, this command does nothing. See Section 9.20.1 for information on enabling the fast commit feature.

Usually, each process performs a checkpoint operation after a certain set of thresholds has been exceeded. The DBO/CHECKPOINT command allows you to spontaneously force each process to perform a checkpoint operation.

Performing a checkpoint operation is useful for several purposes. A checkpoint operation with the /WAIT qualifier causes all updated database pages to be flushed to disk. A checkpoint operation also improves the redo performance of the database recovery (DBR) process (although the per-process parameters should have already been properly initialized with this goal in mind).

When the DBO/CHECKPOINT/WAIT command completes (the system prompt is returned), all active processes have successfully performed a checkpoint operation.

When the system prompt is returned after you issue the DBO/CHECKPOINT/NOWAIT command, there is no guarantee that all active processes have successfully performed a checkpoint operation.

Parameter

root-file-spec
The root file specification for the database you want to checkpoint. You can use either a full or partial file specification, or a logical name.

If you specify only a file name, Oracle CODASYL DBMS looks for the database in the current default directory. If you do not specify a file extension, Oracle CODASYL DBMS assumes a file extension of .ROO.

Command Qualifiers

/WAIT
/NOWAIT
Specifies whether or not the system prompt is to be returned before the checkpoint operation completes.

When you specify the /WAIT qualifier, the system prompt is not returned to you until all processes have flushed updated database pages to disk. When you specify the /NOWAIT qualifier, the system prompt is returned immediately, before all processes have flushed database pages to disk. In addition, when you specify the /NOWAIT qualifier, there is no guarantee that all processes will flush their database pages to disk.

The /NOWAIT qualifier is useful when it is more essential that the system prompt be returned immediately than it is to be certain that all processes have checkpointed.

The /WAIT qualifier is the default.

Example

```
$ DBO/MODIFY PARTS/FAST_COMMIT=ENABLED
$ DBO/CHECKPOINT PARTS
```

In this example, the first command enables fast commit and the second comand causes all the active database processes on all nodes to immediately perform a checkpoint operation.

```
$ DBO/MODIFY PARTS/FAST_COMMIT=ENABLED
$ DBO/CHECKPOINT/NOWAIT PARTS
```

In this example, the first command enables fast commit. The second command requests that all the active database processes on all nodes perform a checkpoint operation and that the system prompt be returned to you immediately. In this case, there is no guarantee that all processes will actually perform a checkpoint operation.

9.8 DBO/CLOSE Command

Closes an open database.

Format

DBO/CLOSE root-file-spec [,...]

Command Qualifiers	Defaults
/[NO]ABORT=option	/ABORT=FORCEX
/[NO]CLUSTER	/NOCLUSTER
/[NO]STATISTICS	/NOSTATISTICS
/[NO]WAIT	/NOWAIT

Description

The DBO/CLOSE command closes an open database. You can close the database immediately by specifying the /ABORT qualifier, or you can allow current users to finish their transactions by specifying /NOABORT.

If you use a DBO/OPEN command to open a database, you must later use a DBO/CLOSE command to close it.

If you have specified manual opening for your database, you must use the DBO/OPEN command to manually open the database before any users can bind to it and the DBO/CLOSE command to manually close the database. You can also use the DBO/CLOSE command to close a database that is open because a process has bound to it. A root file is considered open if it has been specified in a previous DBO/OPEN command or BIND statement.

Use the DBO/SHOW USERS command to display information about databases in current use on your node.

Parameter

root-file-spec [,...]
Specifies one or more open root files. The default file type is .ROO.

Command Qualifiers

/ABORT=option
/NOABORT
Specifies whether to close the database immediately or allow the process to run down.

The /ABORT qualifier has two options:

- DELPRC

  When you use the DELPRC or delete process option, run units are not recovered. The processes and any subprocesses of all database users are deleted, thereby unbinding the processes from the database. The .RUJ files are left in the directories to be recovered on the next bind to the database. Meanwhile, the database is in an inconsistent state. The database must be recovered before you can perform any exclusive operation, such as backup or verify.

- FORCEX

  When you use the FORCEX or forced exit option, run units are recovered and no .RUJ files are left in the directories. The option cannot force an exit of a database process with a spawned subprocess or a suspended process. It aborts batch jobs that are using the database. Therefore, any exclusive operation will work. FORCEX is the default.

The DELPRC and FORCEX options are based on OpenVMS system services $DELPRC and $FORCEX. Refer to OpenVMS System Services documentation and online help.

With the /NOABORT option, users already bound to the database can continue, and the root file global sections remain mapped until all users unbind. No new users are allowed to bind to the database. When all current images terminate, DBO closes the database.

/CLUSTER
/NOCLUSTER
Closes a database on all nodes of a VMScluster that currently have the database open. This is equivalent to issuing a DBO/CLOSE command on every node in the cluster. This command works only from a node on which the database is open. If you enter this command on a node on which the database is not open, you receive an error message.

The /NOCLUSTER qualifier is the default unless /WAIT is also specified in which case /CLUSTER is implicit.

/[NO]STATISTICS
This qualifier indicates that the current statistics information should be preserved. The default is /NOSTATISTICS, which indicates that statistics information is not preserved upon a database close. You cannot use the /STATISTICS qualifier across a cluster, thus the qualifiers /STATISTICS and /CLUSTER are disallowed.

The statistics information is stored in a node-specific database file, located in the database directory. The file is named database_node.RDS. For example, the PARTS database open on node MYNODE would use a statistics file named PARTS_MYNODE.RDS.

See the DBO/OPEN/STATISTICS command for further details on the save statistics.

/WAIT
/NOWAIT
This qualifier causes DBO to close and recover the database before the system prompt is returned to you. You should always specify the /WAIT qualifier, unless you are attempting to recover from some failure. When you specify the /WAIT qualifier, DBO performs all the auxiliary actions required to close and recover the database systemwide and it does not return with the prompt until those actions have been completed.

When you specify the /NOWAIT qualifier, the database might not be closed when the system prompt returns. In this case, you can receive errors if you attempt to access the database after you issue the DBO/CLOSE/NOWAIT command and the system prompt is returned, but before the monitor has actually closed the database.

The /NOWAIT qualifier is the default.

Examples

1.
```
$ DBO/CLOSE/CLUSTER PARTS/WAIT
```

   This command closes the PARTS database on all nodes of the VMScluster. The system prompt will not be returned until the database is closed and all the auxiliary actions required to close and recover the database systemwide have been completed.

2.
```
$ DBO/CLOSE/WAIT/NOCLUSTER PARTS
%DCL-W-CONFLICT, illegal combination of command elements - check documentation
\WAIT\
```

   This examples shows that the qualifiers /WAIT and /NOCLUSTER are incompatible.

9.9 DBO/CONVERT Commands

There are two forms of the DBO/CONVERT command:

- DBO/CONVERT converts a database created with a previous version to run with the current version of Oracle CODASYL DBMS.
- DBO/CONVERT/CDD converts the metadata in a schema in Oracle CDD/Repository to be consistent with the current version of Oracle CODASYL DBMS.

Both commands may be necessary after an upgrade of Oracle CODASYL DBMS. For example, the format of the data definition control blocks (DDCBs) changed with Oracle CODASYL DBMS Version 5.0. The DDCBs in a database root file are converted when you enter the DBO/CONVERT command to convert the database to the current structure level. Because copies of the DDCBs may

also be maintained in Oracle CDD/Repository, you may also need to enter the DBO/CONVERT/CDD command to convert the pre-Version 5.0 schema to the current structure level.

The two conversion commands are described in the following sections.

9.9.1 DBO/CONVERT Command

Converts a database created with a previous version of Oracle CODASYL DBMS to run with the current version.

Format

DBO/CONVERT root-file-spec [,...]

Command Qualifier	Default
/[NOCOMMIT]	/COMMIT
/[NO]LOG	Current DCL verify value
/[NO]ROLLBACK	/NOROLLBACK

Description

Once you convert the database to the current format, it can no longer run under earlier versions of Oracle CODASYL DBMS. Certain database files in the current version are structured differently from those in previous versions. DBO/CONVERT performs the necessary changes to allow a database created by Oracle CODASYL DBMS Version 2.2 or higher to run under the current version. To convert a database, you must have BYPASS privileges.

Before converting your database, you should roll back all existing .RUJ files. To do this, simply bind to the database.

Parameter

root-file-spec [,...]
Specifies the root files of one or more databases that are currently in the previous format. The default file type is .ROO.

Command Qualifier

/COMMIT
/NOCOMMIT
Specifies that the database is to be converted to the current structure level permanently; you will not be able to roll back the conversion after the convert operation completes.

The /NOCOMMIT qualifier also specifies that the database is to be converted to the current structure level; however, if the need arises later, you can roll back the database to the previous structure level with the /ROLLBACK qualifier.

To convert a database using the /NOCOMMIT qualifier, the database must be at Version 5.0, at least. Databases at a structure level prior to Version 5.0 can only be converted with the /COMMIT qualifier.

The /COMMIT qualifier is the default.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request log output and /NOLOG to prevent it. The default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/ROLLBACK
/NOROLLBACK
Specifies that a database that has been converted, but not committed, be reverted to the structure level at which the database existed when the DBO/CONVERT/NOCOMMIT command was issued. Issue the DBO/CONVERT/ROLLBACK command from the current version level of the existing database, not from the prior version level. After issuing the rollback command, the database will not be accessible from the current version of Oracle CODASYL DBMS. If the prior version is no longer available on your system, you will need to reinstall it in order to access the database.

After-image journaling is disabled by the DBO/CONVERT/ROLLBACK command and the following message is displayed:

```
DBO-I-CANTENAAIJ, The After-image Journal is now DISABLED. You
must enable it manually.
```

It is your responsibility to restart after-image journaling after issuing the DBO/CONVERT/ROLLBACK command.

Note

Oracle Corporation strongly recommends that you back up the database before and after issuing a DBO/CONVERT command. A database converted using the DBO/CONVERT utility may not be recoverable if a full database backup is not made immediately. Subsequently restoring the database using the backup file created prior to the conversion will leave the database in an unrecoverable state.

Example

```
$ DBO/CONVERT/LOG PARTSDB,NEWPTS
```

This command converts the root files of the databases PARTSDB and NEWPTS to the current Oracle CODASYL DBMS format and displays a log of the conversions on SYS$OUTPUT.

```
$ DBO/SHOW VERSION
Executing DBO for Oracle CODASYL DBMS V7.0-00
$ DBO/CONVERT/NOCOMMIT PARTS/LOG
%DBO-I-LOGOPNROO, opened root file DISK1:[USER1]PARTS.ROO;1
%DBO-I-LOGMODSTR,     activated after-image journal "AIJ_ONE"
%DBO-W-DOFULLBCK, full database backup should be done to ensure
  future recovery
%DBO-I-LOGCONVRT, database root converted to current structure level
$ DBO/CONVERT/ROLLBACK PARTS/LOG
%DBO-I-LOGOPNROO, opened root file DISK1:[USER1]PARTS.ROO;1
%DBO-I-CANTENAAIJ, The After-image Journal is now DISABLED. You must
  enable it manually.
%DBO-I-LOGROLLBCK, Database rolled back to structure level 61.0
```

This example first converts a database created under struture level 6.1 to structure level 7.0 and specifies that the conversion should not be committed. Oracle CODASYL DBMS performs the conversion and activates the next available fixed-size .AIJ file.

Next, the conversion is rolled back. Oracle CODASYL DBMS rolls back the conversion and notifies the user that this action has resulted in disabling after-image journaling.

9.9.2 DBO/CONVERT/CDD Command

Converts the metadata in each specified schema in Oracle CDD/Repository to be consistent with the current version of Oracle CODASYL DBMS.

Format

DBO/CONVERT/CDD schema-name [,...]

Command Qualifier	Default
/[NO]LOG	Current DCL verify value

Description

The DBO/CONVERT/CDD command converts the metadata in Oracle CDD/Repository using the schema as a parameter. This command also converts the subschemas, storage schemas, and security schemas in Oracle CDD/Repository.

Parameter

schema-name [,...]
Specifies one or more schemas that are currently in the previous format.

Command Qualifier

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request log output and /NOLOG to prevent it. The default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

Example

```
$ DBO/CONVERT/CDD/LOG PARTS
```

This command converts the PARTS schema to the current Oracle CODASYL DBMS format and displays a log of the conversions on SYS$OUTPUT.

9.10 DBO/COPY_DATABASE Command

Allows you to create a duplicate database.

Format

DBO/COPY_DATABASE root-file-spec [area-name[, . . .]]

Command Qualifiers	Defaults
/[NO]AFTER_JOURNAL[=file-spec]	See description
/[NO]AIJ_OPTIONS[=journal-opts-file]	See description
/[NO]CDD_INTEGRATE	/CDD_INTEGRATE
/[NO]CHECKSUM_VERIFICATION	/CHECKSUM_VERIFICATION
/CLUSTER_NODES=integer	
/DIRECTORY=directory-spec	
/[NO]DUPLICATE	/NODUPLICATE
/GLOBAL_BUFFERS= MAXIMUM_PER_USER=integer,[NO]ENABLED	See description
/LOCK_TIMEOUT=integer	
/[NO]LOG	Current DCL verify value
/[NO]ONLINE	/NOONLINE
/OPEN_MODE=option	See description
/OPTION=file-spec	
/PAGE_BUFFERS=integer	/PAGE_BUFFERS=3
/PATH=cdd-repository-path	See description
/[NO]QUIET_POINT	QUIET_POINT
/ROOT=root-file-spec	
/USERS=integer	

Command or Area Qualifiers	Defaults
/BLOCKS_PER_PAGE=integer	
/FILE=file-spec	
/SNAPSHOTS[=option(, . . .)]	
/THRESHOLDS=(pct1[,pct2[,pct3]])	

Description

The DBO/COPY_DATABASE command allows you to modify certain area parameters when the copy operation is performed. All the files are processed simultaneously. The DBO/COPY_DATABASE command eliminates the need for intermediate storage media.

The DBO/COPY_DATABASE command is securable.

Command Parameters

root-file-spec
Specifies the name of the database root file for the database you want to duplicate.

area-name [, . . .]
Specifies the name of one or more storage areas whose parameters you are changing. The area name parameter is optional unless you are using the DBO/COPY_DATABASE command to modify the parameters of one or more storage areas.

Command Qualifiers

/AFTER_JOURNAL[=file-spec]
/NOAFTER_JOURNAL
Specifies a valid device and directory name for the new after-image journal file and enables after-image journaling for the database. The default file type is .AIJ. Do not specify a version number on this file specification. If you use /NOAFTER_JOURNAL, DBO does not enable after-image journaling for the new database. The default is to retain the current journaling state and to create a new journal file.

/AIJ_OPTIONS[=journal-opts-file]
/NOAIJ_OPTIONS
Specifies how DBO is to handle after-image journaling and .AIJ file creation, using the following rules:

- If you specify the /AIJ_OPTIONS qualifier and provide a journal-opts-file, DBO enables journaling and creates the .AIJ file or files you specify for the database copy. If only one .AIJ file is created for the database copy, it will be an extensible .AIJ file. If two or more .AIJ files are created for the database copy, they will be fixed-size .AIJ files (as long as at least two .AIJ files are always available).
- If you specify the /AIJ_OPTIONS qualifier, but do not provide a journal-opts-file, DBO disables journaling and does not create any new .AIJ files.
- If you specify the /NOAIJ_OPTIONS qualifier, DBO disables journaling and does not create any new .AIJ files.
- If you do not specify an /AFTER_JOURNAL, /NOAFTER_JOURNAL, /AIJ_OPTIONS, or /NOAIJ_OPTIONS qualifier, DBO disables after-image journaling and does not create a new .AIJ file.

You can only specify one, or none, of the following after-image journal qualifiers in a single DBO command: /AFTER_JOURNAL, /NOAFTER_JOURNAL, /AIJ_OPTIONS, /NOAIJ_OPTIONS.

If you specify a journal-opts-file, it should appear in the following format:

```
JOURNAL [IS] {ENABLED | DISABLED} -
[RESERVE n] -
[ALLOCATION [IS] n] -
[EXTENT [IS] n] -
[BACKUPS [ARE] {MANUAL|AUTOMATIC} [FILE filename]] -
[OVERWRITE [IS] {ENABLED|DISABLED}] -
[SHUTDOWN_TIMEOUT [IS] n] -
[NOTIFY [IS] {ENABLED|DISABLED}] -
[CACHE [IS] {ENABLED FILE filename|DISABLED}]

ADD [JOURNAL] journal-name -
FILE filename -
[BACKUP_FILE filename] -
[ALLOCATION [IS] n]
```

See the DBO/SHOW AFTER_JOURNAL command for information on the format of a journal-opts-file.

/CDD_INTEGRATE
/NOCDD_INTEGRATE
Specifies whether or not you want DBO to update the database instance information to reflect the existence of the database you are creating. The default is /CDD_INTEGRATE.

If you specify /NOCDD_INTEGRATE, DBO copies the database but does not record it in Oracle CDD/Repository. Therefore, the database you are creating will not appear in the output of a DBO/REPORT/FULL command.

/CHECKSUM_VERIFICATION
/NOCHECKSUM_VERIFICATION
Requests that the page checksum be verified for each page copied. The default is to perform this verification.

/CLUSTER_NODES=integer
Specifies the maximum number of nodes that can access the database in a VMScluster.

/DIRECTORY=directory-spec
Specifies the destination for the copied areas.

/DUPLICATE
/NODUPLICATE
Generates a new database with the same content but with a different identity from that of the original database. For this reason, journal files may not be interchanged between the original and the duplicate database. This qualifier creates copies of your databases that are not expected to evolve independently in time. In this case, being able to exchange journal files may be a security breach, and a possible source of corruption. You can create a duplicate database when you use the /DUPLICATE qualifier or create the original database again when you use the /NODUPLICATE qualifier. The default is /NODUPLICATE.

/GLOBAL_BUFFERS=MAXIMUM_PER_USER=integer,ENABLED
NOENABLED
Specifies the maximum number of global buffers a run unit can allocate. The default is 5 global buffers.

/LOCK_TIMEOUT=integer
Specifies a timeout interval or maximum time in seconds to wait for the quiet-point lock request when the operation is performed on line.

The default is the minimum of the default set for the database at its creation and the value of the logical name DBM$BIND_LOCK_TIMEOUT_INTERVAL.

/LOG
/NOLOG
Specifies whether or not to report the process of the command on SYS$OUTPUT. Specify /LOG to request log output and /NOLOG to prevent it. The default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/ONLINE
/NOONLINE
Specifies that the copy operation be performed while other users are bound to the database. The areas to be moved are locked for read-only access, so the operation is compatible with all but exclusive access.

/OPEN_MODE=option
Specifies the mode in which the database can be opened for access. If you specify the AUTOMATIC option, the default, the first bind statement issued causes the database to open automatically. If you specify the MANUAL option, you must use the DBO/OPEN command to open the database and the DBO/CLOSE command to close the database. You can change options using DBO/MODIFY.

/OPTION=file-spec
Specifies a file whose text defines areas and their qualifiers. This feature makes it easier to define database characteristics without exceeding the character and token limits for DCL-level commands.

The format of the text in the file is:

{area-name [qualifier]...}...

There is no limit on the number of area-names; however, each area entry in the options file cannot exceed 256 characters and 128 tokens. Use of /OPTIONS supersedes the need for DBO indirect command files. The default file type is .DBO.

The /OPTION qualifier cannot be used in conjunction with the area-name parameter.

/PAGE_BUFFERS=integer
Specifies the number of buffers to be allocated for each file to be copied. The number of buffers used is twice the number specified; half are used for reading the file and half for writing the copy. Values specified may range from 1 to 5. The default value is 3. Larger values may improve performance, but they increase memory use.

/PATH=cdd-repository-path
Specifies an Oracle CDD/Repository location containing the schema with which the database will be integrated. If you do not specify /PATH, DBO uses the schema path in the root file.

/QUIET_POINT
/NOQUIET_POINT
Allows you to specify that a database copy operation is to occur either immediately or when a quiet point for database activity occurs. A quiet point is defined as a point where no active update transactions are in progress in the database.

When you specify the /NOQUIET_POINT qualifier, DBO proceeds with the copy operation as soon as the /COPY_DATABASE command is issued, regardless of any update transaction activity in progress in the database. Because DBO must acquire concurrent-read locks on all physical and logical areas, the copy operation fails if there are any active transactions with exclusive locks on a storage area. However, once DBO has successfully acquired all concurrent-read storage area locks, it should not encounter any further lock conflicts. If a transaction that causes DBO to request exclusive locks is started while the copy operation is proceeding, that transaction either waits or gets a lock conflict error, but the copy operation continues unaffected. If you intend to use the /NOQUIET_POINT qualifier with a copy procedure that previously specified the /QUIET_POINT qualifier (or did not specify either the /QUIET_POINT or /NOQUIET_POINT qualifier), you should examine any applications that execute concurrently with the copy operation. You might need to modify your applications or your copy procedure to handle the lock conflicts that can occur when you specify the /NOQUIET_POINT qualifier.

When you specify the /QUIET_POINT qualifier, the copy operation begins when a quiet point is reached. Other update transactions issued after the database copy operation begins are prevented from executing until after the root file for the database has been copied (copying of the database storage areas begins after the root file is copied).

The default is the /QUIET_POINT qualifier.

/ROOT=root-file-spec
Specifies the location for the database root file of the duplicate database.

/USERS=integer
Specifies the number of users who can access the database concurrently.

File or Area Qualifiers

/BLOCKS_PER_PAGE=integer
Specifies a new page size for the storage area.

/FILE=file-spec
Specifies a new location for the storage area. The default file type is .DBS.

/SNAPSHOTS [=(option[, . . .])]
/NOSNAPSHOTS
Allows you to select snapshot options for storage areas. These options can be global (affecting the entire database) or local (affecting one or more specific areas). Local qualification overrides global qualification.

The following options are available for each storage area that has snapshots allowed:

- ALLOCATION=integer

Defines the initial number of database pages in the snapshot file. The pages in the snapshot file are the same size as the pages in the corresponding storage area.

- FILE=file-spec

  Identifies the snapshot file name for this storage area. You can use /FILE=file-spec to specify any part of the full file specification for the snapshot file. The default file type is .SNP. If you omit the version number, DBO uses the latest version. If you omit any other portions of the file specification, DBO uses the corresponding parts of the storage area file specification. (See the DBO/CREATE/FILE option.)

  By default, DBO creates the snapshot file in the same directory as the storage area file. DBO derives the name of the snapshot file from the name of the storage area file using a file type of .SNP and the latest version.

/THRESHOLDS=(pct1[,pct2[,pct3]])

Specifies one, two, or three new threshold values for the storage area. The threshold values determine how much free space can be guaranteed to exist on a database page when storing records. Each threshold value represents a percentage of fullness on a data page. When a data page reaches the percentage of fullness defined by a given threshold value, the SPAM entry for the data page is updated to contain that threshold value. For example, if a record is large enough to occupy 50 percent of a database page, the Database Control System (DBCS) examines the threshold values to determine which pages have room for the new record.

Examples

1.
```
$ DBO/COPY_DATABASE PARTS/DIRECTORY=DISK1:[PARTS.WORK]
```

   This example makes a duplicate copy of the PARTS database and stores it in the DISK1:[PARTS.WORK] directory.

2.
```
$ DBO/COPY_DATABASE PARTS
$ DIR/DATE PARTS.ROO

Directory DISK1:[TEMP]

PARTS.ROO;2          15-MAY-1992 16:27:50.98
PARTS.ROO;1          09-MAY-1992 12:29:28.00
```

   This example shows a simple copy of a database within a user's directory. In this instance, the new database is not uniquely different from the original database. It has the same content and identity as the original database. Use the DCL DIRECTORY command to verify that the database was copied.

3.
```
$ DBO/COPY_DATABASE/DUPLICATE PARTS
```

   This example shows a duplication of a database within a user's directory using the /DUPLICATE qualifier. In this instance, the duplicated database is uniquely different from the original database. It has the same content as the original database but its identity is different. As a result, journal files may not be exchanged between the original database and the duplicate database. Use the DCL DIRECTORY command to verify that the database was duplicated.

9.11 DBO/CREATE Command

Creates an empty database.

Format

DBO/CREATE schema-name [area-name [,...]]

Command Qualifiers	Defaults
/[NO]ADJUSTABLE_LOCKING=(n1[,n2...[n8]])	/ADJUSTABLE_LOCKING=(10,10,10)
/[NO]AFTER_JOURNAL=file-spec	/NOAFTER_JOURNAL
/AIJ_OPTIONS=option[,...]	See description
/BACKUP_OPTIONS=[NO]SCAN_OPTIMIZATION	/BACKUP_OPTIONS=SCAN_OPTIMIZATION
/[NO]BATCH_WRITE[=(option[,...])]	/BATCH_WRITE
/BUFFERS=integer	See description
/CACHE=option[,...]	See description
/[NO]CDD_INTEGRATE	/CDD_INTEGRATE
/CLOSE=option	
/CLUSTER_NODES=integer	/CLUSTER_NODES=16
/DBR_BUFFERS=integer	See description
/[NO]DEFERRED_SNAPSHOTS	/NODEFERRED_SNAPSHOTS
/DIRECTORY=directory-spec	
/[NO]FAST_COMMIT=(options[,...])	/NOFAST_COMMIT
/[NO]GLOBAL_BUFFERS=(options[,...])	/GLOBAL_BUFFERS=NOENABLED
/[NO]HOLD_RETRIEVAL_LOCKS	/NOHOLD_RETRIEVAL_LOCKS
/IMPORT_FILE=file-spec	See description
/JOURNAL_OPTIONS=option[,...]	See description
/LENGTH_BUFFER=integer	See description
/LOCK_OPTIONS=option[,...]	See description
/[NO]LOG	Current DCL verify value
/[NO]MULTITHREAD=option[,...]	/NOMULTITHREAD
/OPEN=option	AUTOMATIC
/OPTIONS=file-spec	
/PREFETCH=([NO]ENABLED,DEPTH=integer)	See description
/[NO]PLT	/NOPLT
/RECOVER_JOURNAL=file-spec	See description
/RESERVE=(options[,...])	See description
/ROOT=root-file-spec	See description
/RUJ_OPTIONS=(option[,...])	See description
/[NO]SECURITY_SCHEMAS[=(security-schema-name[,...])]	/SECURITY_SCHEMAS=*
/SERVER=AFTER_JOURNAL=MANUAL | AUTOMATIC	MANUAL
/[NO]STATISTICS	/STATISTICS
/STORAGE_SCHEMA=storage-schema-name	/STORAGE_SCHEMA=DEFAULT_STORAGE_SCHEMA
/[NO]SUBSCHEMAS[=subschema-name[,...]]	/SUBSCHEMAS=*
/TIMEOUT=LOCK=integer	
/TRANSACTION=(option[,...])	See description
/USERS=integer	/USERS=10
/[NO]WAIT_RECORD_LOCKS	/WAIT_RECORD_LOCKS

Command or Area Qualifiers	Defaults
/ALLOCATION=integer	/ALLOCATION=100

/BLOCKS_PER_PAGE=integer	/BLOCKS_PER_PAGE=2
/[NO]CHECKSUM_PAGES	/CHECKSUM_PAGES
/EXPANSION=(argument-string)	See description
/[NO]EXTENSION=integer	/EXTENSION=100
/FILE=file-spec	See description
/INTERVAL=integer	/INTERVAL=256
/[NO]SNAPSHOTS [=(option[,...])]	/SNAPSHOTS
/[NO]SPACE_MANAGEMENT	/SPACE_MANAGEMENT
/THRESHOLDS= (pct1[,pct2[,pct3]])	/THRESHOLDS=(70,85,95)

Description

When you create a database, the parameters and qualifiers of the DBO/CREATE command tell the system:

- Schema, storage schema, security schemas, and subschemas the database will use initially
- Name of the database
- Number of users who can access the database at any one time
- Locking characteristics for transactions against the database
- Whether to use the manual or automatic database open option
- Size and number of buffers
- File specifications for database storage area files
- Page size for each area file
- If area files can be extended and, if so, the extended size
- Disk storage allocation for each area file
- Whether or not to integrate Oracle CDD/Repository metadata information in the root file with database instance information in Oracle CDD/Repository
- Space area management (SPAM) information for database areas, including definitions of page fullness percentage thresholds and SPAM page intervals for each database area
- If snapshots will be deferred
- If snapshots will be used for areas and, if so, what initial options will be used
- Where to place the recovery-unit journal files
- If after-image journaling will be used and, if so, how many slots to reserve for after-image journal (.AIJ) files in the database root file, the file specification of the first .AIJ file, various global journaling characteristics for the database, and specific characteristics for the first .AIJ file created
- How many slots to reserve for record-level caches
- To enable or disable fast incremental backup
- File containing area definition information
- To enable fast commit for transactions

A successful DBO/CREATE command produces a root file (default file type .ROO) and for each schema area, a database storage file (default file type .DBS). In addition, it can produce snapshot files (default file type .SNP) and an after-image journal file (default file type .AIJ). By default, snapshot files are allowed for each area but not enabled unless you specify the /SNAPSHOTS=ENABLED qualifier.

If you use /NOSNAPSHOTS, DBO does not allow snapshots for the storage area. You can later enable snapshots by restoring your database from backup using the /SNAPSHOTS=(ALLOWED) option on the DBO/RESTORE command. It is recommended that you use /SNAPSHOTS.

With the DBO/CREATE command, you can specify the default placement of the recovery-unit journal (.RUJ) files.

A database contains command authorization lists (CALs) and a user execution list (UEL) in its root file. There is one CAL for each securable DBO command. The CALs determine which users are allowed to execute securable DBO commands on the database. The UEL maps database users to security schemas, thus securing data against unauthorized access despite the user's subschema view.

After database creation, DBO considers the UIC of the user who issued the DBO/CREATE command to be the database owner. By default, a newly created database denies access to all users except the database owner. The database owner is mapped to the NULL security schema, which allows the owner to bind to the database and access data. Also, the owner is mapped to each securable DBO command. This limits only the owner to securable DBO commands until the command authorization list (CAL) is altered with the DBO/GRANT_COMMAND command.

You can reinitialize database storage area files with the DBO/INITIALIZE command, change any database attributes with the DBO/MODIFY command, and change some database attributes with the DBO/RESTORE command. For more information on the database attributes that can be changed, see DBO/MODIFY and DBO/RESTORE.

Parameters

schema-name
Identifies the schema to use in creating a database. If the schema resides in Oracle CDD/Repository, specify the full repository pathname. If the schema resides in a metadata (.DBM) file, specify the schema name only.

area-name [,...]
Identifies one or more areas defined in the schema by an AREA clause.

Command Qualifiers

/ADJUSTABLE_LOCKING=(n1[,n2...[,n8]])
/NOADJUSTABLE_LOCKING
Specifies the intermediate page-range lock levels for the adjustable granularity record lock feature in Oracle CODASYL DBMS. It minimizes the number of OpenVMS locks needed during the execution of your program by allowing you to specify intermediate page range locking rather than locking at the area and record levels.

By default, Oracle CODASYL DBMS uses three intermediate lock levels of 10, 100, and 1000 pages. It is recommended that you start with the default setting of /ADJUSTABLE_LOCKING and that you modify the settings only if you encounter performance problems.

If you specify /NOADJUSTABLE_LOCKING, Oracle CODASYL DBMS performs area-level and record-level locking only. No intermediate page ranges will be locked.

/AFTER_JOURNAL=file-spec
/NOAFTER_JOURNAL
Specifies a valid device and directory name for an after-image journal file. The default file type is .AIJ. Do not specify a version number on this file specification.

If you want multiple .AIJ files, this qualifier creates a new .AIJ file when used with the /AIJ_OPTIONS=CREATE qualifier. You must provide the device, directory, and file name. This qualifier *does not* enable after-image journaling. If you use /JOURNAL_OPTIONS=NOENABLED, DBO does not enable journaling for the database.

Note

In order to be compatible with previous versions of Oracle CODASYL DBMS, the /AFTER_JOURNAL=filename-spec qualifier still provides the same function as before. That is, you can still use this qualifier to create one .AIJ file and enable after-image journaling for the database if you want to use only one .AIJ file. However, using the qualifier this way limits you to the pre-Version 6.0 journaling function. It is recommended that you use the extended Version 6.0 syntax to create .AIJ files and enable journaling, regardless of how many you create. See the qualifier descriptions for /JOURNAL_OPTIONS=ENABLED and /AIJ_OPTIONS=CREATE for more information.

You can create only one .AIJ file on the DBO/CREATE command line. To create subsequent .AIJ files after creating the database, use the DBO/MODIFY command with the /AFTER_JOURNAL and /AIJ_OPTIONS=CREATE qualifiers.

The command in the following example creates a database called PARTS, reserves slots for 10 .AIJ files in the database root file, and creates the first .AIJ file with a user-defined name of AIJ_1. After-image journaling is not enabled at this point.

```
$ DBO/CREATE/RESERVE=(AFTER_JOURNAL=10) -
_$   /AIJ_OPTIONS=(CREATE,NAME=AIJ_1 -)
_$   /AFTER_JOURNAL=DBM$DISK:[USER1]AIJ_1.AIJ PARTS
```

/AIJ_OPTIONS=option [,...]
Performs operations on an .AIJ file. The following options are available:

- ALLOCATION=integer

  Specifies the initial number of 512-byte disk blocks for an .AIJ file. The minimum number of blocks allowed is 512; this is also the default.

- BACKUP=[backup-file-spec]

  Specifies a file name for the output file when the after-image journal is backed up.

If you use the BACKUP option without a backup file specification, the backup file name will be the current version of the original .AIJ file name. Because you are creating a new version of the .AIJ file rather than actually performing a backup operation, you can save overhead. You can then perform an OpenVMS backup on the older version of the .AIJ file, then delete that version. If you do this, however, be sure you have adequate disk space to hold both versions of the .AIJ file.

The NOBACKUP option is disallowed on the DBO/CREATE/AIJ_OPTIONS command.

- CREATE

  Creates a new .AIJ file. You must also use the /AFTER_JOURNAL=file-name-spec to specify the file location. It is also required that you use the NAME option to name the new file. This will simplify and shorten your command lines.

- EXTENSION=integer

  Specifies the number of 512-byte disk blocks necessary by which to extend an .AIJ file when it is full. This option is meaningful only when you are using a single, extensible after-image journal file. It is ignored if you are using multiple journal files. However, if you set the option and then go back to using a single, extensible journal file, the journal file will have the previously specified extension value. The minimum number of blocks allowed is 512; this is also the default.

- NAME=name

  Specifies a name that describes the .AIJ file you are creating with the /AFTER_JOURNAL=file-spec qualifier.

/BACKUP_OPTIONS=SCAN_OPTIMIZATION
/BACKUP_OPTIONS=NOSCAN_OPTIMIZATION

Allows you to enable or disable fast incremental backup operations. When you specify the /BACKUP_OPTIONS=SCAN_OPTIMIZATION qualifier, DBO records the identity of pages that have not been updated since the last full backup operation. This feature, called fast incremental backup, means that during an incremental backup operation, DBO need not read every page of every area to determine if the page has been updated since the last full backup operation.

However, there is a cost in recording this information in the database. In some circumstances the cost might be too high, particularly if you do not intend to use incremental backup operations. If you want to disable the recording of pages changed since the last full backup operation, specify the /BACKUP_OPTIONS=NOSCAN_OPTIMIZATION qualifier.

The database setting for fast incremental backup can also be enabled or disabled with the DBO/MODIFY/BACKUP_OPTIONS=[NO]SCAN_OPTIMIZATION command and the DBO/BACKUP/MULTITHREADED/[NO]SCAN_OPTIMIZATION command. See Section 9.20.1 and Section 9.5.3 for details.

The default is the /BACKUP_OPTIONS=SCAN_OPTIMIZATION qualifier.

/BATCH_WRITE[=(options[,...])]
/NOBATCH_WRITE

Eliminates the process stalls due to most disk write operations to the database, thereby improving performance for update applications. By default, the feature is enabled.

- MAXIMUM_SIZE=n

  Specifies the maximum number of buffers used for batch write operations. The default and maximum values are the number of buffers allocated to the user. The minumum is 2, maximum is 65535.

  When MAXIMUM_SIZE is not defined for a database, write operations occur in large bursts when the user's buffers are full. Use the MAXIMUM_SIZE parameter to control the size and frequency of batch write operations, evening the I/O behavior on your system to avoid these bursts.

  This parameter does not affect the size of batch write operations for snapshot pages or write operations used for commit or checkpoint operations.

  See the *Oracle CODASYL DBMS Programming Reference Manual* for information on the logical name, DBM$BIND_BATCH_MAX.

- CLEAN_BUFFER_COUNT=n

  Specifies the number of clean buffers to be maintained at the end of a process' least recently used (LRU) queue of buffers for replacement. In the asynchronous batch write process, a set of clean (or empty) buffers must be available at all times at the end of the LRU queue, so that as buffers are filled and written, there are others that can replace them. The minimum is 2, the maximum is 65535. If you specify 0 or no value for CLEAN_BUFFER_COUNT, the default will be 5 clean buffers.

  This parameter is used in conjunction with MAXIMUM_SIZE to determine the size and frequency of buffer transfers. For example, if MAXIMUM_SIZE is defined as 6 and CLEAN_BUFFER_COUNT is 4, then this specifies that 6 modified buffers be written asynchronously whenever the number of remaining clean buffers for the process is less than 4.

  The values for CLEAN_BUFFER_COUNT and MAXIMUM_SIZE should be determined through trial and error to provide the best performance. In general, performance is better when the value for CLEAN_BUFFER_COUNT is not set too high, because a high value can increase CPU usage. However, you also want to anticipate asynchronous batch write operations and have Oracle CODASYL DBMS start them while there are still enough clean buffers to allow processing to continue without a stall. Therefore, do not set the CLEAN_BUFFER_COUNT value too low.

  See the *Oracle CODASYL DBMS Programming Reference Manual* for information on the logical name, DBM$BIND_CLEAN_BUF_CNT.

- [NO]ENABLED

  Enables asynchronous batch write operations for a database. ENABLED is the default.

  See the *Oracle CODASYL DBMS Programming Reference Manual* for information on the logical name, DBM$BIND_ABW_DISABLED.

/BUFFERS=integer

Specifies the number of database page buffers each run unit can allocate, if your database is using local buffering. The default is 10. If your database is using global buffering, specify the default number of buffers a run unit allocates.

/CACHE=(option[,...])

Specifies a default directory and checkpoint options for caches created using the DBO/CACHE/ADD command.

- CHECKPOINT=(option[,...]))

The CHECKPOINT keyword for the /CACHE qualifier specifies the source records and target for checkpoint operations for the row cache. If the keyword ALL_ROWS is specified, then all rows in the cache are written during the checkpoint process.

If the keyword UPDATED_ROWS is specified, then just the added or the updated rows are written during the checkpoint process. The keywords ALL_ROWS and UPDATED_ROWS are mutually exclusive.

If the target of the checkpoint operation is BACKING_FILE, then the RCS process writes the row cache entries to the backing (.DBC) files. If the target is DATABASE, then the added or updated rows (only UPDATED_ROWS is allowed) are written back to the database. Upon recovery from a node failure, the database recovery process has no data on the contents of the row cache. Therefore, it does not repopulate the row caches in memory. The keywords BACKING_FILE and DATABASE are mutually exclusive.

The keyword TIMED specifies the frequency (in seconds) with which the Row Cache Server (RCS) process checkpoints the contents of the row caches back to disk.

The frequency of RCS checkpointing is important in determining how much of an .AIJ file must be read during a recovery operation following a node failure. It also affects the frequency with which marked records get flushed back to the database for those row caches that checkpoint to the database. The default is every 15 minutes (900 seconds).

This CHECKPOINT keyword overrides the database-level CHECKPOINT clause.

- DIRECTORY=directory-spec

  The DIRECTORY keyword specifies the name of the directory to which row cache backing file information is written. The database system generates a file name (row-cache-name.DBC) automatically for each row cache at checkpoint time. Specify a device name and directory name only. By default, the location is the directory of the database root file. The backing files are temporary. They are deleted when the row cache server (RCS) is shut down, unless the system logical name, DBM$BIND_RCS_KEEP_BACKING_FILES, is defined.

- [NO]ENABLED

  The ENABLED or NOENABLED keywords specifies whether or not the row caching feature is enabled. Enabling row caching does not affect database operations until a cache is created and assigned to one or more storage areas.

  When row caching is disabled, all previously created and assigned caches remain and will be available if row caching is enabled again.

/CDD_INTEGRATE
/NOCDD_INTEGRATE

Specifies whether or not you want DBO to update the database instance information to reflect the existence of the database you are creating. The default is /CDD_INTEGRATE.

If you specify /NOCDD_INTEGRATE, DBO creates the database but does not record it in Oracle CDD/Repository. Therefore, the database you are creating will not appear in the output of a DBO/REPORT/FULL command.

/CLOSE=option
Specifies the mode in which the database is to be closed.

- MANUAL

  The database must be closed manually by issuing the DBO/CLOSE command.

- AUTOMATIC

  The database will be automatically closed when there are no users accessing the database. This mode is applicable only when DBO/MODIFY/OPEN is set to AUTOMATIC.

- TIMED_AUTOMATIC=integer

  The database will be closed automatically after there has been no activity for the specified minutes. The default is 0 minutes.

/CLUSTER_NODES=integer
Specifies the maximum number of nodes that can access a database simultaneously in a cluster. You can modify this number using DBO/MODIFY. The default is 16. The maximum number supported is 96 nodes.

/DBR_BUFFERS=integer
Specifies the number of buffers to be used by the database recovery process (DBR) when it clears uncommitted changes in the database by using the .RUJ file. The larger the number of DBR buffers, the faster the recovery process. Setting the number of buffers high and setting a user's working set extent low can lead to a high page fault rate. If this occurs, increase the working set extent quota for the user. The user name shown in the monitor log for the DBR process is the process name of the person who started the monitor.

The more memory on the system, the more buffers it can handle. The default number of DBR buffers is 10. However, if your database is using global buffering, this value is overridden by the maximum number of global buffers per run unit set with the /GLOBAL_BUFFERS=MAXIMUM_PER_USER qualifier. The length of a DBR buffer is set by the /LENGTH_BUFFER qualifier.

/DEFERRED_SNAPSHOTS
/NODEFERRED_SNAPSHOTS
Causes update transactions to write records to snapshot files only if a snapshot transaction (BATCH RETRIEVAL) is in progress or waiting to start. When you are using deferred snapshots, batch retrieval transactions wait for update transactions to begin writing to the snapshot files before starting. (Batch retrieval transactions wait until all current update transactions commit or roll back.) Once a batch retrieval transaction is attempted, all subsequent update transactions write before-images of the records they modify to the snapshot files. After the batch retrieval transaction completes, future update transactions do not write to the snapshot file.

You must specify deferred snapshots for the entire database. However, you can allow or enable snapshots for individual areas.

If you use the default, /NODEFERRED_SNAPSHOTS, then an update transaction for that area will always write to the area snapshot file, even when no snapshot transaction is in progress or waiting. Using /DEFERRED_SNAPSHOTS saves overhead incurred by writing to the snapshot file only when a snapshot transaction or batch retrieval job is in progress or waiting.

You can use the DBO/MODIFY command to specify /DEFERRED_SNAPSHOTS or /NODEFERRED_SNAPSHOTS.

/DIRECTORY=directory-spec
Specifies a global device and directory file specification portion for the root, database storage, and snapshot files of the database. The /DIRECTORY qualifier is useful to shorten commands when the /FILE, /ROOT, and /SNAPSHOTS qualifiers all specify the same device and directory.

For example, the following two commands are equivalent:

```
$ DBO/CREATE/FILE=DISK:[USER]/ROOT=DISK:[USER]/SNAP=FILE=DISK:[USER] PARTS
$ DBO/CREATE/DIRECTORY=DISK:[USER] PARTS
```

You can override the /DIRECTORY file specification with a global or local /FILE qualifier.

/FAST_COMMIT=(options[,...])
/NOFAST_COMMIT
Specifies the commit scheme for fast commit. By default, fast commit is disabled. The following options are available:

- [NO]ENABLED

  Specifies whether or not fast commit is to be used.

- BLOCKS_PER_CHECKPOINT=integer

  Specifies an AIJ block checkpoint interval value. The valid range of integers is 0 to 2,147,483,647. The default value is 512 blocks.

- COMMIT_TO_JOURNAL=(options[,...])

  Specifies the commit scheme for optimizing fast commit. The following options are available:

  – [NO]ENABLED

    Specifies whether or not the commit to journal optimization scheme is used for fast commit.

  – TRANSACTION_INTERVAL=integer

    Sets the size of a group of transaction sequence numbers (TSNs). The default size is 256. The minimum size is 8 and the maximum size is 512.

- TIME_PER_CHECKPOINT=n

  Specifies a time checkpoint interval value, expressed in seconds. The valid range of integers is 0 to 2,147,483,647. By default, no time interval value is specified. If you do not specify a value, no checkpoint is triggered by time.

/GLOBAL_BUFFERS=(options[,...])
/NOGLOBAL_BUFFERS
Specifies the buffering scheme for database pages. The following options are available:

- BUFFERS=integer

  Specifies the total number of global buffers for a database. Valid values are from 2 to 32768. The default value is equal to 5 times the maximum number of users allowed to access the database simultaneously, as specified with the /USER qualifier.

- [NO]ENABLED

Specify /GLOBAL_BUFFERS=ENABLED to use global buffers. Specify /GLOBAL_BUFFERS=NOENABLED to use local buffers. Local buffering is the default.

- MAXIMUM_PER_USER=integer

  Specifies the maximum number of global buffers a run unit can allocate. The default is 5 global buffers.

- PAGE_TRANSFER={DISK | MEMORY}

  Specify PAGE_TRANSFER=MEMORY to enable optimized page transfer. Specify PAGE_TRANSFER=DISK to disable it.

  When the memory page transfers feature is enabled, a process does not need to write a modified page to disk before another process accesses the page. In other words, pages in a process' allocate set can contain committed updates from another process that have not been written to disk.

  If a database has the required characteristics for using memory page transfers, the performance gain of update-intensive applications can be significant. This is because of the number of input/output (I/O) operations saved by not writing updates to disk, and the ability to share pages among processes.

  The memory page transfers feature is available for any database with the following characteristics:

  – Global buffers are enabled.

  – After-image journaling is enabled.

  – Fast-commit processing is enabled.

  – The database can be accessed only from a single node (the /CLUSTER_NODES value for the database is 1).

By default, global buffers are disabled.

/HOLD_RETRIEVAL_LOCKS
/NOHOLD_RETRIEVAL_LOCKS

Specifies whether or not to hold retrieval locks until the end of the transaction. Specify /HOLD_RETRIEVAL_LOCKS to hold all record retrieval locks for each transaction until the end of the transaction.

Specify /NOHOLD_RETRIEVAL_LOCKS, the default, to release record retrieval locks for records not in a keeplist or serving as a currency indicator.

/IMPORT_FILE=file-spec

Specifies a metadata file to be used when defining a database. See the DBO /EXPORT command for information on copying metadata from an existing database.

When creating a database based on definitions stored in an exported metadata file, the target storage schema defaults to the storage schema found in the metadata file rather than to DEFAULT_STORAGE_SCHEMA.

/JOURNAL_OPTIONS=options

Specifies after-image journaling characteristics for all .AIJ files in the database when you are using multiple .AIJ files. You can override this qualifier for a specified .AIJ file with the /AIJ_OPTIONS qualifier. The database-wide options you can specify are:

- ALLOCATION=integer

  Specifies the initial number of 512-byte disk blocks for .AIJ files. The minimum number of blocks allowed is 512; this is also the default.

- BACKUP=[backup-file-spec]

  Specifies the default name for the output file when .AIJ files are backed up.

  If you use the BACKUP option without a backup file specification, the backup file name will be the current version of the original .AIJ file name. Because you are creating a new version of an .AIJ file rather than actually performing a backup operation, you can save overhead. You can then perform an OpenVMS backup on the older versions of the .AIJ file, then delete those versions. If you do this, however, be sure you have have adequate disk space to create the various versions of .AIJ files.

  The NOBACKUP option is disallowed with the DBO/CREATE/JOURNAL_ OPTIONS command.

- [NO]ENABLED

  Enables after-image journaling for the database. You must have at least one accessible journal file to enable journaling. The NOENABLED option disables journaling for the database, but does not delete an .AIJ file if you have created it on the DBO/CREATE command line.

- EXTENSION=integer

  Specifies the number of disk blocks a fixed-size journal file will extend if it is converted to an extensible journal. The default and minimum number that can be stored is 512 blocks.

- [NO]NOTIFY=(operator-name)

  Designates any of 16 OPCOM operators to receive after-image journaling information. If a catastrophic event occurs, for example if the AIJ switch-over operation cannot complete, or if the .AIJ file cannot be extended, the system operator will receive event notification messages from the database. The operator names you can specify are:

  - CENTRAL
  - DISK
  - CLUSTER
  - SECURITY
  - OPER1, OPER2, OPER3, . . . OPER12

  For example, to have OPER2 receive .AIJ event notification messages for the PARTS database, enter the following command:

  ```
  $ DBO/CREATE/JOURNAL_OPTIONS=(NOTIFY=OPER2) PARTS
  ```

  For more information on using OPCOM, see the OpenVMS documentation on system management.

- [NO]OVERWRITE

  Specifies that .AIJ files can be written over without first being backed up. The /NOOVERWRITE qualifier specifies that only an .AIJ file that has been backed up can be written over. You should only use this option when database availability or performance requirements are more important than database recoverability. The default option is NOOVERWRITE.

- [NO]SHUTDOWN=integer

  Specifies the time, in minutes, until database shutdown occurs in the event of an AIJ problem.

- [NO]SPOOLER

  Enables the monitor-invoked AIJ Backup Server (ABS) for the database. The ABS automatically backs up a single .AIJ file to disk, then unbinds and terminates. This option applies when you are using multiple .AIJ files. You must also use the BACKUP=file-spec option to set a file specification for the backup file.

  The default NOSPOOLER option specifies that the ABS not be started by the monitor. You can dump the header file and check the journaling section to see if the ABS has been enabled.

The command in the following example shows how to use the /RESERVE, /AIJ_OPTIONS, and /JOURNAL_OPTIONS qualifiers with the DBO/CREATE command. Only qualifiers related to after-image journaling are shown.

```
$ DBO/CREATE -
_$ /RESERVE=(AFTER_JOURNAL=10) -
_$ /AIJ_OPTIONS=(CREATE,NAME=AIJ_1) -
_$ /AFTER_JOURNAL=(AIJ$DISK:[USER1]AIJ1.AIJ) -
_$ /JOURNAL_OPTIONS=(ENABLED,ALLOCATION=1024,BACKUP=AIJ1_BACKUP.AIJ, -
_$ NOTIFY=OPER1,SPOOLER) -
_$ PARTS
```

The command in the previous example performs the following after-image journal operations when creating the PARTS database:

- Reserves 10 slots in the database root file.
- Creates one .AIJ file, with the file specification AIJ$DISK:[USER1]AIJ_1.AIJ and the user-specified name AIJ_1.
- Allocates 1024 blocks for the .AIJ file.
- Specifies a default backup file name of AIJ1_BACKUP.AIJ for the .AIJ file.
- Enables after-image journaling for the database.
- Designates the OPCOM operator name, OPER1, to receive after-image journal information, and notify the operator if certain catastrophic events are occurring.
- Enables the monitor-invoked Automatic Backup Server (ABS) for the database.

The DBO/CREATE command in the previous example also enables the following journaling characteristics by default:

- If the DBA decides to use only one .AIJ file, that file is given an extension value of 512 blocks.
- If multiple .AIJ files are used, .AIJ switchover will not write over active journal files.

/LENGTH_BUFFER=integer

Defines the number of disk blocks in a database page buffer. The default is 5 times the largest page size in your database.

/LOCK_OPTIONS=option[,...]
Enables either adjustable record and page locking, the default locking mechanism for Oracle CODASYL DBMS, or two-phase page locking. (The two-phase page locking feature was introduced in Version 6.0.) It also enables carry-over lock optimization, which operates independently of record and page locking.

This qualifier replaces /LOCK_OPTIMIZATION.

The /LOCK_OPTIONS qualifier takes the following options:

- RECORD_LEVEL

  Enables adjustable record and page locking. (Whenever adjustable record locking is used, adjustable page locking is also automatically enabled.)

  Adjustable record and page locking is the default lock mechanism for Oracle CODASYL DBMS. It is enabled automatically with the DBO/CREATE command, unless you specify the PAGE_LEVEL option. Although it is not necessary to specify the /LOCK_OPTIONS=RECORD_LEVEL qualifier with the DBO/CREATE command, the function is there if you want to use it.

- PAGE_LEVEL

  Enables two-phase page locking. Two-phase page locking allows transactions to hold page locks per storage area and not request record locks.

  By default, two-phase page locking is disabled. If you want to enable it at database creation, you must specify /LOCK_OPTIONS=PAGE_LEVEL with the DBO/CREATE command.

- [NO]CARRYOVER

  Enables or disables carry-over lock optimization. This option replaces the /LOCK_OPTIMIZATION qualifier. Carry-over locks are enabled by default. If you want to disable them at database creation, you must specify the NOCARRYOVER option.

  Carry-over lock optimization operates independently of record and page locking.

  For more information on adjustable record locking, adjustable page locking, two-phase page locking, and carry-over lock optimization, see the *Oracle CODASYL DBMS Database Design Guide*.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to print the name and attributes of each file as it is created and /NOLOG to prevent it. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/MULTITHREAD=option[,...]
/NOMULTITHREAD
Creates database storage area files in parallel.

- BUFFERS=n

  Number of buffers to use during the creation of storage areas. Value can be a minimum of 1 buffer and a maximum of 255. The default is 10 buffers.

- THREADS=n

Determines the maximum number of areas to initialize in parallel. The minimum is 1 and the maximum is 65535. The default value is 0.

/OPEN=option
Specifies the mode in which the database can be opened for access. If you specify the AUTOMATIC option, the default, the first bind statement issued causes the database to open automatically. If you specify the MANUAL option, you must use the DBO/OPEN command to open the database and the DBO/CLOSE command to close the database. You can change options using DBO/MODIFY.

/OPTIONS=file-spec
Specifies a file whose text defines areas and their qualifiers. This feature makes it easier to define database characteristics without exceeding the character and token limits for DCL-level commands.

The format of the text in the file is:

{area-name [qualifier]...}...

DBO treats the text as part of the DBO/CREATE command. Thus, commas are required to separate area names, and hyphens are needed to continue command lines. Because the area qualifiers are positionally dependent, the location of the /OPTIONS qualifier is also important.

There is no limit on the number of area-names; however, each area entry in the options file cannot exceed 256 characters and 128 tokens. Use of /OPTIONS supersedes the need for DBO indirect command files. The default file type is .DBO.

Command line qualifiers override corresponding qualifiers in the options file.

/PREFETCH=([NO]ENABLED,DEPTH=integer)
Allows you to specify options to affect asynchronous prefetch operations. These options are:

- ENABLED or NOENABLED

  Use the ENABLED option to enable, and the NOENABLED option to disable, asynchronous prefetching of database pages. Asynchronous prefetching is enabled by default.

- DEPTH=integer

  Use the DEPTH option to set the number of buffers into which Oracle CODASYL DBMS can prefetch database pages. The default is the number of database buffers divided by 4, or the value 8, whichever is smaller.

See the *Oracle CODASYL DBMS Programming Reference Manual* for more information on asynchronous prefetching.

/PLT
/NOPLT
Lock partitioning allows you to specify whether more than one lock tree is used for the database or whether all lock trees for a database are mastered by one database resource tree.

When partitioned lock trees (PLT) are enabled for a database, locks for storage areas are separated from the database resource tree and all locks for each storage area are independently mastered on the VMScluster node that has the highest traffic for that resource. OpenVMS determines the node that is using each resource the most and moves the resource hierarchy to that node.

You should not enable lock partitioning for single-node systems, because all lock requests are local on single-node systems.

By default, lock partitioning is disabled.

/RECOVER_JOURNAL=file-spec
Specifies the default device and directory of the .RUJ files. Generally, you need to specify only a device and directory for this qualifier; DBO appends the file type .RUJ to the root file name.

The default location for RUJ files is [DBM$RUJ] on the user's default login device.

In a VMScluster environment, you can place all .RUJ files in a single, central directory ensuring the continued availability of all .RUJ files. OpenVMS has a utility to limit how many versions of a file can be in one directory. Setting a limit on the directory with .RUJ files can cause database corruption.

/RESERVE=(option[,...])
Allows you to reserve slots for after-image journal (.AIJ) files or record-level caches using the following options:

- AFTER_JOURNAL=integer

  Reserves slots in the database root file for a predetermined number of after-image journal files. Exclusive database access is required. A full database backup must be performed after this operation.

  Because this is not an online operation, if you plan to use multiple .AIJ files, it is important to determine ahead of time how many slots you will need. When you create or convert a database, Oracle CODASYL DBMS provides one slot by default if you do not specify the /RESERVE qualifier. It is recommended that you reserve more slots than you need. You cannot decrease the number of slots set aside once they are reserved. There is no default for this qualifier.

- CACHE=integer

  Reserves slots in the database root file for a predetermined number of pointers to row caches. Exclusive database access is required. You must perform a full database backup after this operation.

  Because this is not an online operation, if you plan to use multiple row caches, it is important to determine ahead of time how many slots you will need. If you reserve a sufficient number of slots for row caches, you can add row caches while the database is on line without interrupting database activity.

  If you do not reserve any slots for row caches, Oracle CODASYL DBMS reserves one slot when the database is created, restored or converted. You cannot decrease the number of reserved slots for row caching after you have reserved them.

/ROOT=root-file-spec
Specifies the name of the root file. The default file type is .ROO. If you omit the /ROOT qualifier, the root file is created in your default directory using the first nine alphanumeric characters of the schema name and the file type .ROO.

A database root file must be on a public (shared) disk. It must not be on a private disk.

/RUJ_OPTIONS=(BUFFER={LOCAL|GLOBAL})
This qualifier permits the assignment of recovery-unit journal (RUJ) buffers created for each process, (typically allocated to local virtual memory), to be assigned to a shared global section. This means the recovery process can occur in global memory and possibly avoid disk access.

This buffer memory can be defined as GLOBAL to improve row cache performance for recovery. If row caching is disabled, then buffer memory is always LOCAL.

/SECURITY_SCHEMAS [=(security-schema-name [,...])]
/NOSECURITY_SCHEMAS
Specifies which security schemas, if any, to store in the created root file. The different forms are:

- /SECURITY_SCHEMAS [=(security-schema-name [,...])]

  DBO stores all security schemas in the name list in the root file. If you name a security schema that does not exist in Oracle CDD/Repository, an error occurs.

- /SECURITY_SCHEMAS

  DBO stores the default security schema in the root file. An error occurs if the default security schema has not previously been compiled or generated.

- /SECURITY_SCHEMAS=*

  DBO stores all security schemas existing for this schema in the root file. If there are no security schemas, it is not an error. This is the default.

- /NOSECURITY_SCHEMAS

  DBO does not store any security schemas in the root file, regardless of the security schemas stored for the schema in Oracle CDD/Repository.

/SERVER=AFTER_JOURNAL=AUTOMATIC
/SERVER=AFTER_JOURNAL=MANUAL
Enables the AIJ Logging Server (ALS) and determines whether the ALS is started automatically or manually.

The /SERVER=AFTER_JOURNAL=AUTOMATIC qualifier enables the ALS and specifies that it be automatically started by the monitor when the database is opened.

The /SERVER=AFTER_JOURNAL=MANUAL qualifier enables the ALS and specifies that it be started and stopped manually, on line. If you set the ALS to the manual online mode, use the DBO/SERVER/AFTER_JOURNAL command to start or stop the ALS. MANUAL is the default.

/STATISTICS
/NOSTATISTICS
Determines whether or not database statistics are enabled. By default, statistics are enabled.

/STORAGE_SCHEMA=storage-schema-name
Identifies a storage schema presently stored under the named schema. If there is no such storage schema, an error occurs.

If you omit this qualifier, DBO uses the default storage schema. An error occurs if the default storage schema has not previously been compiled or generated.

/SUBSCHEMAS [=(subschema-name [,...])]
/NOSUBSCHEMAS

Identifies which subschemas, if any, to store in the created root file. The different forms are as follows:

- /SUBSCHEMAS [=(subschema-name [,...])]

  DBO stores all subschemas in the name list in the root file. If you name a subschema that does not exist in Oracle CDD/Repository, an error occurs.

- /SUBSCHEMAS

  DBO stores only the default subschema in the root file. An error results if the default subschema has not previously been compiled or generated.

- /SUBSCHEMAS=*

  DBO stores all subschemas existing for this schema in the root file. If there are no subschemas it is not an error, but you will not be able to use the database. This is the default.

- /NOSUBSCHEMAS

  DBO does not store any subschemas in the root file, regardless of the subschemas stored for the schema in Oracle CDD/Repository.

/TIMEOUT=LOCK=integer

Specifies the length of time (in seconds) the database should wait before returning a lock timeout message. You can use this qualifier in conjunction with the DBM$BIND_LOCK_TIMEOUT_INTERVAL logical name and with the WAIT option on the READY statement to set the lock timeout duration.

/TRANSACTION=(options...)

This qualifier allows the database administrator to define the setting for prestarted transactions.

- PRESTART=ENABLED
  PRESTART=NOENABLED

  The default when a database is created is to have prestarted transactions enabled. This is a performance feature which allows Oracle CODASYL DBMS to automatically establish a new transaction during the regular commit processing. When the application starts a transaction the I/O has already been perform and so reduced I/O to the root file.

  Use NOENABLED to disable this feature.

- PRESTART=TIMEOUT:n

  The database administrator may define a timeout for a prestarted transaction.

  ```
  $ DBO/MODIFY/TRANSACTION=PRESTART=(TIMEOUT=n) dbroot
  ```

  here 'n' is a value in seconds (range 0:3600). This value represents the number of seconds to wait before aborting the prestarted transaction. Timing out the prestarted transaction may prevent snapshot file growth in environments where servers stay bound to the database with long periods of inactivity.

  Additionally, a process will be forced to obtain a new transaction sequence number (TSN) if the same TSN has been reused throughout the duration of the prestarted transaction timeout interval. This permits processes that constantly reuse TSNs to periodically obtain a new TSN, thus preventing excessive snapshot growth.

/USERS=integer
Defines the number of users who can access the database at the same time. Users include DBO commands. The maximum number of users is 16368. Setting this value unnecessarily high causes performance to degrade. The default is 10.

The minimum value is five (5) to allow for various optional database servers, such as the ABS (Automatic Backup Server), RCS (Row Cache Server) or ALS (AIJ Log Server) to access the database.

/WAIT_RECORD_LOCKS
/NOWAIT_RECORD_LOCKS
Specifies whether to wait on record lock conflicts or treat lock conflicts as errors.

Specify /WAIT_RECORD_LOCKS, the default, to wait until the lock is granted or a deadlock condition occurs.

Specify /NOWAIT_RECORD_LOCKS to return an error message whenever a record lock conflict occurs.

Command or Area Qualifiers

/ALLOCATION=integer
Specifies the initial number of data pages in a disk storage file. The default is 100.

This page count does not include SPAM pages. SPAM pages are automatically allocated according to the /SPACE_MANAGEMENT and /INTERVAL qualifiers.

The initial allocation determines the initial CALC range of the area. You can add overflow pages to database areas by automatic /EXPANSION or /EXTENSION, and you can increase the CALC range with DBO/MODIFY.

/BLOCKS_PER_PAGE=integer
Specifies the number of disk blocks in each page of an area file. All pages in any one storage area have the same size. The maximum size is 63 blocks. The size that you choose can only be modified by backing up and restoring the database. The default is 2 blocks (1024 bytes).

/CHECKSUM_PAGES
/NOCHECKSUM_PAGES
Determines whether or not page checksums are calculated when pages are read from or written to storage areas. Use the /CHECKSUM_PAGES qualifier to enable checksum calculations and use the /NOCHECKSUM_PAGES qualifier to disable checksum calculations. You can disable, then subsequently enable checksum calculations without error. However, once you disable checksum calculations, corrupt pages may not be detected even if checksum calculations are enabled again.

The /CHECKSUM_PAGES qualifier is the default.

/EXPANSION=argument-string
Defines how many data pages, if any, are added automatically to a storage area file when its present allocation is full. Automatic extension adds only overflow space to a file; it does not extend the file's CALC range. DBO inserts SPAM pages as needed while observing the specified /INTERVAL value.

The /EXPANSION qualifier takes the following arguments:

- ENABLED and NOENABLED turn area extension on and off. ENABLED is the default. If you specify NOENABLED, DBO reports an exception to your application program when the storage area becomes full but does not extend the area.
- MINIMUM specifies the minimum number of pages necessary to extend the area. The default is 100.
- MAXIMUM specifies the maximum number of pages necessary to extend the area. By default, the area can extend to an unlimited number of pages.
- PERCENT specifies a percentage of the current area size necessary to extend the area. The default is PERCENT=0, which results in a fixed extension of 100 pages (based on the MINIMUM=100 default).
- SPREAD or NOSPREAD is meaningful only if the area resides in a multiple volume disk set.

  If the area resides in a multiple volume disk set, specifying NOSPREAD specifies that any extensions to the area will be created on the first volume of the disk set. If you specify SPREAD and the area resides in a multiple volume disk set, the extension spreads across the multiple volumes. NOSPREAD is the default.

  DBO reports a device-full error when all volumes in the disk set are full.

DBO determines the size of the extension by the MINIMUM, MAXIMUM, and PERCENT values and by the current size of the area. The MINIMUM and MAXIMUM values define a range of allowable extension sizes. The utility first calculates the percentage of the current area size and then determines if that value is within the MINIMUM and MAXIMUM value range. The extension size is either the MINIMUM number of pages, the MAXIMUM number, or that size produced by calculating the percentage of the current area size, provided the percentage falls between the specified MINIMUM and MAXIMUM values. For example, you might specify the following qualifier for an area:

```
/EXPANSION=(MIN=100,MAX=500,PERCENT=25)
```

If the area currently contains 300 pages, the percentage calculation yields a proposed extension value of 75. This is less than the minimum extension value of 100. DBO therefore extends the file by 100 pages. If the area currently contains 500 pages, the percentage calculation yields a proposed extension value of 125. This is greater than the minimum extension value of 100 and less than the maximum of 500. DBO therefore extends the file by 125 pages. If the area contains 2000 or more pages, DBO extends it by the defined maximum of 500 pages regardless of the specified percentage. For a larger extension, you must specify a higher MAXIMUM value.

/EXTENSION=integer
/NOEXTENSION

Defines a constant number of data pages to add automatically to a storage area file when full. The default is 100 data pages. Automatic extension adds only overflow space to a file; it does not extend the file's CALC range. DBO inserts SPAM pages as needed while observing the specified /INTERVAL value.

The /EXPANSION qualifier replaces the /[NO]EXTENSION qualifier. The /[NO]EXTENSION qualifier remains solely for compatibility with previous versions of Oracle CODASYL DBMS. If you include both the /EXPANSION and /EXTENSION qualifier in a command, the /EXPANSION qualifier overrides the /EXTENSION qualifier.

/FILE=file-spec
Identifies the database file name for this storage area. The default file type is .DBS. You can use /FILE to specify any part of the full file specification for a storage area file.

If you do not specify a database file, DBO creates one in the current default directory with the file specification STORAGE_A.DBS, where STORAGE_A is the first 9 characters of the storage area name.

You can place database storage areas on private disks.

Global use of /FILE is a convenient way to specify a default device and directory for database files, shortening the command. The global file specification becomes the default specification for all storage area files, even for areas not specified. Any part of a complete file specification omitted from a local /FILE qualifier on an area parameter defaults to the file specification given in the global /FILE qualifier.

For example, suppose you want to create the PARTS database on disk DISK: under directory [USER.DBMS]. If your default directory is DISK:[USER], you would enter:

```
$ DBO/CREATE/FILE=DISK:[USER.DBMS] PARTS MAKE/FILE=PTMAKE
```

DBO stores all files of the created database in DISK:[USER.DBMS] and the area MAKE in the file DISK:[USER.DBMS]PTMAKE.DBS. DBO also stores all other storage area files in DISK:[USER.DBMS] using the area file specification defaults.

/INTERVAL=integer
Defines the number of data pages between SPAM pages in the physical storage file and, thus, the maximum number of data pages each SPAM page will manage. The default, also the minimum interval, is 256 data pages.

Unless you specify /NOSPACE_MANAGEMENT, the first page of each database area is a SPAM page. The integer determines where DBO inserts subsequent SPAM pages, provided there are enough pages in the area to require more SPAM pages.

/SNAPSHOTS [=(option[,...])]
/NOSNAPSHOTS
Allows you to select snapshot options for storage areas. These options can be global (affecting the entire database) or local (affecting one or more specific areas). Local qualification overrides global qualification.

The /NOSNAPSHOTS qualifier allows snapshots for a database storage area but initially disables them. (See the following option descriptions for more information.) DBO reserves space on each storage area page for snapshot information. You can thus use DBO/MODIFY later to enable snapshots for the storage area. Even if you disable them again, space will always be reserved for snapshots.

Specify the qualifier /SNAPSHOTS=option to allow snapshots and simultaneously enable them for a database. Batch retrieval transactions require the use of this qualifier.

You must allow and enable snapshots to use the DBO/BACKUP/ONLINE command.

Specifying /NOSNAPSHOTS disallows snapshots for the storage area, leaving more space in the storage area for storing data. (You can later allow and enable snapshots for this storage area by restoring the area from backup and specifying the /SNAPSHOTS=(ALLOWED) option on the DBO/RESTORE command.)

The following options are available for each storage area that has snapshots allowed:

- ALLOCATION=integer

  Defines the initial number of database pages in the snapshot file. (The pages in the snapshot file are the same size as the pages in the corresponding storage area.) The default is 1 with additional pages added as needed.

- [NO]ENABLED

  Specifies whether to enable or disable snapshot activity at this time. The /SNAPSHOTS qualifier requests snapshot capability for the storage area. The default, NOENABLED option, disables snapshot files temporarily. If you specify the ENABLED option, snapshot files are immediately enabled. You can enable snapshots later using the DBO/MODIFY command.

- EXPANSION=([NO]ENABLED, MINIMUM=integer, MAXIMUM=integer, PERCENT=integer, [NO]SPREAD)

  Defines how many data pages are added automatically to the snapshot file when its present allocation is full. This qualifier does not affect the current size of the snapshot file.

  Arguments for this option include:

  – ENABLED and NOENABLED turn area extension on and off. ENABLED is the default.

  – MINIMUM specifies the minimum number of pages necessary to extend the snapshot file when full. The default is 100.

  – MAXIMUM specifies the maximum number of pages necessary to extend the snapshot file when full. By default, the snapshot file can extend to an unlimited number of pages.

  – PERCENT specifies a percentage of the current snapshot file size necessary to extend the snapshot file when full. If the number of pages calculated is less than the MINIMUM or greater than the MAXIMUM, DBO extends the snapshot file by the MINIMUM or the MAXIMUM number of pages.

  – SPREAD or NOSPREAD is meaningful only if the snapshot file resides in a multiple volume disk set. If you specify SPREAD, the extension is spread evenly across the multiple volumes. NOSPREAD is the default. DBO reports a device-full error when all volumes in the disk set are full.

  DBO determines the size of the extension by the MINIMUM, MAXIMUM, and PERCENT values and by the current size of the snapshot file. The MINIMUM and MAXIMUM values define a range of allowable extension sizes. The utility first calculates the percentage of the current area size and

then determines if that value is within the MINIMUM and MAXIMUM value range. The extension size is either the MINIMUM number of pages, the MAXIMUM number, or that number produced by calculating the percentage of the current area size, provided the percentage falls between the specified MINIMUM and MAXIMUM values.

- [NO]EXTENSION=integer

  Defines how many data pages to add to the snapshot file when its present allocation is full. By specifying EXTENSION=0 or NOEXTENSION, you prevent automatic extension.

  This option remains solely for compatibility with previous versions of Oracle CODASYL DBMS. EXPANSION values override EXTENSION values.

- FILE=file-spec

  Identifies the snapshot file name for this storage area. You can use /FILE=file-spec to specify any part of the full file specification for the snapshot file. The default file type is .SNP. If you omit the version number, DBO uses the latest version. If you omit any other portions of the file specification, DBO uses the corresponding parts of the storage area file specification. (See the DBO/CREATE/FILE option.)

  By default, DBO creates the snapshot file in the same directory as the storage area file. DBO derives the name of the snapshot file from the name of the storage area file using a file type of .SNP and the latest version.

The /SNAPSHOTS qualifier is the default. The default allows snapshots for a database storage area, but initially disables them.

/SPACE_MANAGEMENT
/NOSPACE_MANAGEMENT

Enables or suppresses the SPAM utility. You cannot change this qualifier with DBO/MODIFY; your choice at database creation time remains in effect for the life of the database, unless you unload and reload. (You can, however, use this qualifier with DBO/MODIFY for a new area being added to the database at a later time.)

It is strongly recommended that you always use the SPAM option. This significantly improves Oracle CODASYL DBMS database update performance as the data in the storage area increases. The SPAM pages take up little space compared to the performance benefit they provide, especially for update-intensive applications.

An area using space area management has a number of SPAM pages in addition to data pages. The number of SPAM pages depends on the /INTERVAL qualifier for each area and does not count against data page allocation or expansion values.

The default is /SPACE_MANAGEMENT.

/THRESHOLDS=(pct1[,pct2[,pct3]])

Specifies one, two, or three threshold values. The threshold values determine how much free space can be guaranteed to exist on a database page when storing records. Each threshold value represents a percentage of fullness on a data page. When a data page reaches the percentage of fullness defined by a given threshold value, the SPAM entry for the data page is updated to contain that threshold value. For example, if a record is large enough to occupy 50 percent of a database page, the Database Control System (DBCS) examines the threshold values to determine which pages have room for the new record.

If you omit /THRESHOLDS for an area, the default thresholds are 70, 85, and 95. If you specify only one or two values, unspecified values default to 100.

Examples

1.
```
$ DBO/CREATE PARTS
```

   This command creates the PARTS database using all the defaults.

   The database root file name is the same as the schema name. Defaults are provided (in part) to allow easy creation of test or sample databases, usually by individuals learning to use Oracle CODASYL DBMS.

2.
```
$ DBO/CREATE/ROOT=CARPARTS/CLUSTER_NODES=5/USERS=50/BUFFERS=20 -
_$ /ALLOCATION=1000 /EXPANSION=(MINIMUM=500,MAXIMUM=1000, -
_$ PERCENT=50,SPREAD) /OPEN=MANUAL -
_$ /ADJUSTABLE_LOCKING=(10,20,30) -
_$ /RECOVER_JOURNAL=DRA1:[DBMSRUJS] -
_$ /WAIT_RECORD_LOCKS -
_$ /SNAPSHOTS=(ALLOCATION=200,NOENABLE) PARTS
```

   This command creates the database CARPARTS from the schema PARTS. It allows a maximum of 5 nodes on the VMScluster, 50 simultaneous users, 20 buffers per run unit, and larger allocation and expansion sizes than the defaults would provide. The SPREAD argument in the /EXPANSION qualifier causes the system to try to spread expansion space over a multiple volume disk set previously defined by the system manager.

   Because the command specifies /OPEN=MANUAL, database administrators have to use the DBO/OPEN command to open the database and DBO/CLOSE to close it. The /ADJUSTABLE_LOCKING qualifier creates three intermediate levels in the record lock tree, with 10 pages in each lock subrange. The first level consists of ranges of 10 pages, the second consists of ranges of 200 (10*20) pages, and the third consists of ranges of 6000 (10*20*30) pages.

   The command also provides for a recovery-unit journal file and allows waiting on record lock conflicts (instead of error messages). The database has snapshot capability; snapshots are disabled at this time but can be enabled by the DBO/MODIFY command.

   All these qualifiers are used globally, so each area created will have the same characteristics.

   See the *Oracle CODASYL DBMS Database Design Guide* for additional examples that explain the use of the DBO/CREATE command in detail.

3.
```
$ DBO/CREATE/LOG /RESERVE=AFTER_JOURNAL=20 -
$ /JOURNAL_OPTIONS=(ENABLED,SPOOLER,NOTIFY=CENTRAL,SHUTDOWN=30) -
_$ /AIJ_OPTIONS=(CREATE,NAME=ONE) -
_$ /AFTER_JOURNAL=DISK2:[USER1]ONE.AIJ PARTS
%DBO-I-LOGMODROO, modifying root file $DISK1:[USER1]PARTS.ROO;1
%DBO-I-LOGMODVAL,     reserved 20 additional after-image journals
%DBO-W-DOFULLBCK, full database backup should be done to ensure
     future recovery
%DBO-I-LOGMODVAL,     modified AIJ shutdown time to 30 minutes
%DBO-I-LOGMODFLG,     enabled after-image journal operator
   notification
%DBO-I-LOGMODFLG,     enabled after-image journal spooler
%DBO-I-LOGCREAIJ, created after-image journal file
  DISK2:[USER1]ONE.AIJ;1
%DBO-I-LOGMODSTR,     created after-image journal "ONE"
%DBO-I-LOGMODSTR,     activated after-image journal "ONE"
%DBO-I-LOGMODFLG,     enabled after-image journaling
%DBO-W-DOFULLBCK, full database backup should be done to ensure
     future recovery
```

This command creates the PARTS database and reserves 20 slots for after-image journal files in the database root file (in addition to the one slot that is reserved by default when the database is created).

The /JOURNAL_OPTIONS qualifier enables after-image journaling with the ENABLED argument. It also sets the following global journaling characteristics for the database: the SPOOLER argument enables the automatic backup spooler for automatic backup of .AIJ files; the NOTIFY argument specifies that the operator CENTRAL be notified of various AIJ events; and the SHUTDOWN argument specifies database users be notified of database shutdown 30 minutes ahead of time.

The /AIJ_OPTIONS qualifier creates one after-image journal and assigns it the name ONE so you do not have to specify the entire file location when working with the file.

The /AFTER_JOURNAL qualifier specifies the location on the disk of the after-image file just created. Note that it is placed on a different disk than the root file to protect against head crashes.

9.12 DBO/DELETE Commands

The DBO/DELETE command deletes database and Oracle CDD/Repository information. This command has six distinct forms that you can use to:

1. Delete a root file and its associated database storage area files, recovery-unit journal files, and snapshot files, while updating Oracle CDD/Repository database instance information to reflect the deletion
2. Delete a schema from Oracle CDD/Repository
3. Delete a security schema from Oracle CDD/Repository
4. Delete a storage schema from Oracle CDD/Repository
5. Delete a subschema from Oracle CDD/Repository
6. Delete database instance information from Oracle CDD/Repository

You must have appropriate Oracle CDD/Repository privileges to delete Oracle CDD/Repository information. The individual responsible for managing Oracle CDD/Repository at your installation can give you Oracle CDD/Repository privileges.

Database root files need not be integrated with Oracle CDD/Repository. You can delete a schema or storage schema without deleting all databases that use it. In addition, you can delete subschemas and security schemas from Oracle CDD/Repository without deleting them from databases. You can also delete database instance information from Oracle CDD/Repository. If you later want to modify the metadata in your database, you might need to use a DBO/INTEGRATE or DBO/MODIFY/CDD_INTEGRATE operation to reestablish the metadata in Oracle CDD/Repository.

9.12.1 DBO/DELETE Root File Command

Deletes a database and optionally updates instance information in Oracle CDD/Repository.

Format

DBO/DELETE root-file-spec [,...]

Command Qualifiers	Defaults
/[NO]CDD_INTEGRATE	/CDD_INTEGRATE
/[NO]CONFIRM	See description
/[NO]LOG	Current DCL verify value

Description

This command deletes the database for the root file you specify. DBO deletes the root file (default file type .ROO) and all its associated database files (default file type .DBS), the recovery-unit journal file (default file type .RUJ), and snapshot files (default file type .SNP) if present. This command also updates database instance information (unless the /NOCDD_INTEGRATE qualifier is specified) for schemas in Oracle CDD/Repository to show that the database no longer exists.

Use /CONFIRM to protect against unintended deletions. This qualifier causes DBO to display a confirmation prompt for each root file you propose to delete.

Parameter

root-file-spec [,...]
Specifies one or more root files you want to delete. The default file type is .ROO.

Command Qualifiers

/CDD_INTEGRATE
/NOCDD_INTEGRATE
Specify /NOCDD_INTEGRATE to delete the specified database without updating the instance information in Oracle CDD/Repository. The default, /CDD_INTEGRATE, will delete the specified database and update the instance information in Oracle CDD/Repository.

/CONFIRM
/NOCONFIRM
Specify /CONFIRM, the default, to have DBO prompt for confirmation before deleting the root file and its database area files. This is the default for interactive processing.

Specify /NOCONFIRM to delete the root file and its database files without prompting. This is the default for batch processing.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to list the file specification of each deleted file on SYS$OUTPUT and /NOLOG to prevent this list. If you specify neither, the default is the setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

Example

```
$ DBO/DELETE/LOG PARTS
DISK:[USER.PARTS]PARTS.ROO;1, delete? [N]: Y
%DBO-I-LOGFILACC, deleted file DISK:[USER.PARTS]MAKE.DBS;1
%DBO-I-LOGFILACC, deleted file DISK:[USER.PARTS]BUY.DBS;1
%DBO-I-LOGFILACC, deleted file DISK:[USER.PARTS]MARKET.DBS;1
%DBO-I-LOGFILACC, deleted file DISK:[USER.PARTS]PERSONNEL.DBS;1
%DBO-I-LOGFILACC, deleted file DISK:[USER.PARTS]MAKE.SNP;1
%DBO-I-LOGFILACC, deleted file DISK:[USER.PARTS]BUY.SNP;1
%DBO-I-LOGFILACC, deleted file DISK:[USER.PARTS]MARKET.SNP;1
%DBO-I-LOGFILACC, deleted file DISK:[USER.PARTS]PERSONNEL.SNP;1
%DBO-I-LOGFILACC, deleted file DISK:[USER.PARTS]PARTS.ROO;1
%DBO-I-LOGPTHNAM, using CDD schema path _CDD$TOP.VIA.DBMS.USER.PARTS
```

This command deletes a database root file and its associated storage area and snapshot files, and requests log output.

9.12.2 DBO/DELETE/INSTANCE Command

Deletes all database instance information from a schema and from storage schemas, subschemas, and security schemas under that schema.

Format

DBO/DELETE/INSTANCE schema-name [,...]

Command Qualifiers	Defaults
/[NO]CONFIRM	See description
/[NO]LOG	Current DCL verify value

Description

The DBO/DELETE/INSTANCE command deletes database instance information from Oracle CDD/Repository for each schema you specify. When you delete instance information from a schema, you also delete instance information from all subschemas, security schemas, and storage schemas stored under that schema in Oracle CDD/Repository. Following deletion, a DBO/REPORT/FULL command on the same Oracle CDD/Repository path would not show the database root file

in the Oracle CDD/Repository instance information. However, the database still exists and you can still bind to it. Databases using the schema are unaffected, except their root files are no longer integrated with Oracle CDD/Repository database instance information.

Use /CONFIRM to protect against unintended deletions. This qualifier causes DBO to display a confirmation prompt for each security schema you propose to delete.

Parameter

schema-name
A schema whose database instance information is to be deleted from Oracle CDD/Repository; specify using the correct Oracle CDD/Repository pathname.

Command Qualifiers

/CONFIRM
/NOCONFIRM
Specify /CONFIRM, the default, to have DBO prompt for confirmation before deleting instance information. This is the default for interactive processing.

Specify /NOCONFIRM to delete the instance information without a confirmation inquiry. This is the default for batch processing.

/LOG
/NOLOG
Specify whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to have DBO report the path you are using on SYS$OUTPUT and /NOLOG to prevent this list. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

Example

```
$ DBO/DELETE/INSTANCE/CONFIRM/LOG PARTS
%DBO-I-LOGPTHNAM, using CDD schema path _CDD$TOP.PARTS
DISK:[USER.PARTS]PARTS.ROO;1, delete instance? [N]: Y
%DBO-I-DELOLDINF, deleting old instance information used
%DBO-I-CDDINSTNC,    by DISK:[USER.PARTS]PARTS.ROO;1
(20-FEB-1987 16:50:34:31)
```

This command deletes database instance information for the PARTS schema from Oracle CDD/Repository (using the default path) and requests log output.

The /LOG qualifier causes an informational message identifying the schema path being used. Following deletion, a DBO/REPORT/FULL command on the same path would show no database named DISK:[USER.PARTS]PARTS.ROO in Oracle CDD/Repository instance information, although the database still exists and you can still bind to it.

9.12.3 DBO/DELETE/SCHEMA Command

Deletes one or more schemas from Oracle CDD/Repository along with Oracle CDD/Repository metadata stored under the deleted schema.

DBO/DELETE/SCHEMA Command

Format

DBO/DELETE/SCHEMA schema-name [,...]

Command Qualifiers	Defaults
/[NO]CONFIRM	See description
/[NO]LOG	Current DCL verify value

Description

The DBO/DELETE/SCHEMA command deletes each specified schema from Oracle CDD/Repository. When you delete a schema, you also delete all subschemas, security schemas, and storage schemas stored under that schema in Oracle CDD/Repository.

You receive a warning message when instance information for the schema indicates the schema is currently in use by one or more databases.

If you also want to delete databases that use the schema, you must do so in a separate operation, using the DBO/DELETE root file command described previously.

Use /CONFIRM to protect against unintended deletions. This qualifier causes DBO to display a confirmation prompt for each schema you propose to delete.

Parameter

schema-name [,...]
Specifies rules for specifying Oracle CDD/Repository pathnames.

Command Qualifiers

/CONFIRM
/NOCONFIRM
Specify /CONFIRM to have DBO prompt for confirmation before deleting each specified schema. This is the default for interactive processing.

Specify /NOCONFIRM to delete the schema without confirmation. This is the default for batch processing.

/LOG
/NOLOG
Specify whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to list the deletion activity on SYS$OUTPUT and /NOLOG to prevent this listing. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

Example

```
$ DBO/DELETE/SCHEMA/LOG PARTS
%DBO-I-LOGPTHNAM, using CDD schema path _CDD$TOP.VIA.DBMS.USER.PARTS
%DBO-I-MAYBEUSED, schema _CDD$TOP.VIA.DBMS.USER.PARTS may
                  still be in use
%DBO-I-CDDINSTNC,     by SYS$32T:[USER.PARTS]PARTS.ROO;1
                     (10-OCT-1985 11:19:03.89)
schema _CDD$TOP.VIA.DBMS.USER.PARTS, delete from CDD? [N]: Y
%DBO-I-LOGDELCDD, deleted schema _CDD$TOP.VIA.DBMS.USER.PARTS from CDD
```

This command deletes the schema PARTS and requests log output. The %DBO-I-CDDINSTNC message occurs only when database instance information is present in the schema.

9.12.4 DBO/DELETE/SECURITY_SCHEMA Command

Deletes one or more security schemas from Oracle CDD/Repository.

Format

DBO/DELETE/SECURITY_SCHEMA schema-name security-schema-name [,...]

Command Qualifiers	Defaults
/[NO]CONFIRM	See description
/[NO]LOG	Current DCL verify value

Description

The DBO/DELETE/SECURITY_SCHEMA command deletes each specified security schema from Oracle CDD/Repository.

Use /CONFIRM to protect against unintended deletions. This qualifier causes DBO to display a confirmation prompt for each security schema you propose to delete.

Parameters

schema-name
Specifies the schema containing the security schemas to be deleted from Oracle CDD/Repository. Follow the rules for specifying Oracle CDD/Repository pathnames.

security-schema-name
Specifies the security schema belonging to the specified schema.

Command Qualifiers

/CONFIRM
/NOCONFIRM
Specify /CONFIRM, the default, to have DBO prompt for confirmation before deleting a security schema. This is the default for interactive processing.

Specify /NOCONFIRM to delete the security schema from Oracle CDD/Repository without inquiry. This is the default for batch processing.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to list the Oracle CDD/Repository path on SYS$OUTPUT and /NOLOG to prevent this list. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

Example

```
$ DBO/DELETE/SECURITY_SCHEMA/LOG PARTS TEST_SEC_SCHEMA
%DBO-I-LOGPTHNAM, using CDD schema path _CDD$TOP.VIA.DBMS.USER.PARTS
%DBO-I-MAYBEUSED, security schema TEST_SEC_SCHEMA may still be in use
%DBO-I-CDDINSTNC,     by SYS$32T:[USER.PARTS]PARTS.ROO;1
                     (10-OCT-1985 11:19:03.89)
security schema TEST_SEC_SCHEMA, delete from CDD? [N]: Y
%DBO-I-LOGDELCDD, deleted security schema TEST_SEC_SCHEMA from CDD
```

This command deletes the security schema TEST_SEC_SCHEMA and requests log output.

9.12.5 DBO/DELETE/STORAGE_SCHEMA Command

Deletes one or more storage schemas from Oracle CDD/Repository.

Format

DBO/DELETE/STORAGE_SCHEMA schema-name storage-schema-name [,...]

Command Qualifiers	Defaults
/[NO]CONFIRM	See description
/[NO]LOG	Current DCL verify value

Description

The DBO/DELETE/STORAGE_SCHEMA command deletes each specified storage schema from Oracle CDD/Repository.

Use /CONFIRM to protect against unintended deletions. This qualifier causes DBO to display a confirmation prompt for each storage schema you propose to delete.

Parameters

schema-name
Specifies the schema containing the storage schema to be deleted from Oracle CDD/Repository. You must follow the rules for specifying Oracle CDD/Repository pathnames.

storage-schema-name [,...]
Specifies one or more of the storage schemas belonging to the specified schema.

Command Qualifiers

/CONFIRM
/NOCONFIRM
Specify /CONFIRM, the default, to have DBO prompt for confirmation of each storage schema deletion. This is the default for interactive processing.

Specify /NOCONFIRM to delete the specified storage schema without a confirmation inquiry. This is the default for batch processing.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to have DBO report the deletion activity on SYS$OUTPUT and /NOLOG to prevent this list. If you specify neither, the default is the current setting of the DCL verify option. (The SET VERIFY command controls the DCL verify option.)

Example

```
$ DBO/DELETE/STORAGE_SCHEMA/LOG PARTS DEFAULT_STORAGE_SCHEMA
%DBO-I-LOGPTHNAM, using CDD schema path _CDD$TOP.VIA.DBMS.USER.PARTS
%DBO-I-MAYBEUSED, storage schema DEFAULT_STORAGE_SCHEMA may still be in use
DBO-I-CDDINSTNC,     by SYS$32T:[USER.PARTS]PARTS.ROO;1
                     (10-OCT-1985 11:19:03.89)
storage schema DEFAULT_STORAGE_SCHEMA, delete from CDD? [N]: N
```

This command deletes the storage schema DEFAULT_STORAGE_SCHEMA and requests log output. DBO does not actually delete the storage schema in this example because the operator responded negatively to the confirmation prompt.

9.12.6 DBO/DELETE/SUBSCHEMA Command

Deletes one or more subschemas from Oracle CDD/Repository.

Format

DBO/DELETE/SUBSCHEMA schema-name subschema-name [,...]

Command Qualifiers	Defaults
/[NO]CONFIRM	See description
/[NO]LOG	Current DCL verify value

Description

The DBO/DELETE/SUBSCHEMA command deletes each specified subschema from Oracle CDD/Repository.

Use /CONFIRM to protect against unintended deletions. This qualifier causes DBO to display a confirmation inquiry for each subschema you specify.

Parameters

schema-name
Specifies the schema containing the subschema to be deleted from Oracle CDD/Repository. It must follow the rules for specifying Oracle CDD/Repository pathnames.

subschema-name [,...]
Specifies one or more of the subschemas that belong to the specified schema.

Command Qualifiers

/CONFIRM
/NOCONFIRM
Specify /CONFIRM, the default, to have DBO prompt for confirmation before deleting a subschema from Oracle CDD/Repository. This is the default for interactive processing.

Specify /NOCONFIRM to delete the subschema from Oracle CDD/Repository without a confirmation inquiry. This is the default for batch processing.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to have DBO list a message identifying Oracle CDD/Repository path and /NOLOG to prevent this list. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

Example

```
$ DBO/DELETE/SUBSCHEMA/LOG PARTS PARTSS4,PARTSS1
%DBO-I-LOGPTHNAM, using CDD schema path _CDD$TOP.VIA.DBMS.USER.PARTS
%DBO-I-MAYBEUSED, subschema PARTSS4 may still be in use
%DBO-I-CDDINSTNC,     by SYS$32T:[USER.PARTS]PARTS.ROO;1
                     (10-OCT-1985 11:19:03.89)
subschema PARTSS4, delete from CDD? [N]: Y
%DBO-I-LOGDELCDD, deleted subschema PARTSS4 from CDD
%DBO-I-MAYBEUSED, subschema PARTSS1 may still be in use
%DBO-I-CDDINSTNC,     by SYS$32T:[USER.PARTS]PARTS.ROO;1
                     (10-OCT-1985 11:19:03.89)
subschema PARTSS1, delete from CDD? [N]: Y
%DBO-I-LOGDELCDD, deleted subschema PARTSS1 from CDD
```

This command deletes the subschemas PARTSS4 and PARTSS1 and requests log output.

9.13 DBO/DUMP Commands

There are seven forms of the DBO/DUMP command, each having its own qualifiers and parameters. These are:

DBO/DUMP root-file-spec

This format dumps information from root files, storage area files, and snapshot files.

DBO/DUMP/AFTER_JOURNAL aij-file-spec

This format dumps information from after-image journal (.AIJ) files.

DBO/DUMP/BACKUP backup-file-spec

This format dumps header block information and information about each storage area contained in the backup file. This format can only be used for single-threaded backups.

DBO/DUMP/BACKUP/MULTITHREAD backup-file-spec

This format dumps information contained in backup files created with the DBO/BACKUP /MULTITHREAD command.

DBO/DUMP/CACHE_FILE cache-filename

This format dumps information contained within a caching backing (.DBC) files.

DBO/DUMP/EXPORT metadata-file-spec

This format dumps information contained within a metadata file.

DBO/DUMP/RECOVERY_JOURNAL ruj-file-spec

This format dumps information from recovery-unit journal (.RUJ) files.

These formats are described in the following sections.

9.13.1 DBO/DUMP Database Command

Lists information from root files, storage area files, and snapshot files.

Format

DBO/DUMP root-file-spec [,...]

Command Qualifiers	Defaults
/[NO]AREAS=area-name [,...]	/NOAREAS
/END=integer	/END=last-page
/FILES	
/[NO]HEADER [=(detail-opt,type-opts)]	See description
/OPTIONS=(type [,...])	/OPTIONS=NORMAL
/OUTPUT=file-spec	SYS$OUTPUT
/[NO]SCHEMA	/NOSCHEMA
/[NO]SECURITY_SCHEMAS [=(security-schema-name[,...])]	/NOSECURITY_SCHEMAS
/[NO]SNAPSHOTS=file-spec [,...]	/NOSNAPSHOTS
/STATE=option	/STATE=BLOCKED
/START=integer	/START=1
/[NO]STORAGE_SCHEMA	/NOSTORAGE_SCHEMA
/[NO]SUBSCHEMAS [=(subschema-name [,...])]	/NOSUBSCHEMAS
/USERS	

Note

The /START and /END qualifiers apply only when the /AREAS or /SNAPSHOTS qualifier is specified.

Description

Depending on your selection of qualifiers, the DBO/DUMP command can list:

- A formatted display of any number of pages in any storage area file of the database.
- Database header information listed by default.

- Current users of the database and optionally just the users that are blocked in current transactions.
- Any or all DDL source code the database uses. This output is identical to that of the DBO/EXTRACT command except that the source code is taken from the root file instead of Oracle CDD/Repository.
- Database root and area file specifications and whether an area is read only.
- A formatted display of any number of pages in any snapshot area file.

Parameter

root-file-spec [,...]
Specifies one or more root files whose header information, source code, storage area file pages, or snapshot file pages you want to dump. The default file type is .ROO.

Command Qualifiers

/AREAS=area-name [,...]
/NOAREAS
Specifies whether or not to dump database storage area pages. If you use the /AREAS=area-name qualifier, you must specify one or more storage areas in the database to dump. DBO does not dump storage area pages if you omit the /AREAS qualifier.

The /AREAS=area-name [,...] qualifier identifies one or more storage areas in the database. Each area-name must be defined in an AREA clause of the schema source code. If you specify an area-name that is not so defined, an error occurs.

If you specify /AREAS=*, DBO dumps all storage areas for the database, starting with the first one defined in the schema.

If you specify /NOAREAS, the default, DBO does not dump any of the storage areas.

Use the /START and /END qualifiers to define the limits of a page dump. These qualifiers will apply to each area named.

/END=integer
When dumping a database, you can specify the range of area or snapshot pages to be dumped. The /END qualifier specifies the end of the range. By default, a dump starts with the first page and ends with the last page. You can override the default using the /START and /END qualifiers to specify the page range. This qualifier works only if used with /SNAPSHOTS or /AREAS.

/FILES
Lists the file specification of the root file, each storage area file, and each snapshot area file. The output is the same as that of an OpenVMS DIRECTORY/NOHEAD/NOTRAIL command.

/HEADER[=(detail-opt, type-opts)]
/NOHEADER
Indicates whether or not to include the database header in the output. Specify the /HEADER qualifier to include all database header information in the output. Specify the /NOHEADER qualifier to suppress the database header listing. Specify the /HEADER=(detail-opt, type-opts) qualifier to limit the output from the header to specific items of interest. Use the detail-opt options (BRIEF

or DETAIL) to limit the amount of output. Use the type-opt options to limit the output to specific types of information. Table 9–3 summarizes the valid parameters for the /HEADER qualifier and the effects of specifying each.

Table 9–3 DBO/DUMP Command /HEADER Parameter List Options

Option	Effect
ALL	Generates the full output of all the header information. If you specify this and other /HEADER options, the other options are ignored. This is the default option.
AREAS	Displays information about active storage areas and snapshot areas.
AUDIT	Displays information about security auditing.
BACKUP	Displays information about backup and recovery operations.
BRIEF	Generates a summary of the requested database root file information.
BUFFERS	Displays information about database buffers.
CDD	Displays information about the data dictionary.
CORRUPT_PAGE	Displays the corrupt page table (CPT).
DETAIL	Generates a complete report of the requested database root file information. This is the default.
FAST_COMMIT	Displays information about whether or not fast commit is enabled or disabled, commit-to-AIJ optimization is enabled or disabled, the AIJ checkpoint intervals, and the transaction interval.
HOT_STANDBY	Displays information regarding Hot Standby databases.
JOURNALING	Displays information about recovery-unit and after-image journaling.
LOCKING	Displays information about fanout factors and database locking, such as whether or not adjustable record locking, carry-over lock optimization, and lock tree partitioning are enabled or disabled.
PARAMETERS	Displays basic root file header information.
ROOT_RECORD	Displays information on database maintenance operations, including backup, restore, verify, alter, and restructure operations.
ROW_CACHES	Displays information about row caches.
SCHEMA	Displays information about the schema.
SECURITY_SCHEMA	Displays information about security schemas.
SEQUENCE_NUMBERS	Displays database sequence numbers.

(continued on next page)

Table 9–3 (Cont.) DBO/DUMP Command /HEADER Parameter List Options

Option	Effect
STORAGE_SCHEMA	Displays information about the storage schema.
SUBSCHEMA	Displays information about subschemas.
USERS	Displays information about active database users.

If you specify both the DETAIL and BRIEF parameters, DETAIL takes precedence. If you specify the ALL parameter and other detail-opt parameters, the ALL option takes precedence. If you specify the BRIEF or DETAIL option only, the default for the type-opts parameter is ALL. If you specify type-opts parameters, but do not specify a detail-opt parameter, the default for the detail-opt parameter is DETAIL.

/OPTIONS=(type [,...])
Specifies the level of information the output will include about schemas, storage schemas, subschemas, security schemas, and database headers. Three types of output are available:

- NORMAL

  Output includes only the source language information. This is the default.

- FULL

  In addition to the NORMAL information, output includes commentary and set participation information.

- DEBUG

  In addition to NORMAL and FULL information, output includes internal information about the data. You can identify internal area, set record, and item numbers using the DEBUG option. In general, DEBUG information is useful only for diagnostic support purposes.

If you specify more than one option, you must enclose the types in parentheses:

```
/OPTIONS=(NORMAL,FULL)
```

The /OPTIONS qualifier has no effect when used with the /AREAS or /SNAPSHOTS qualifiers.

/OUTPUT=file-spec
Specifies the file where output will be sent. The default file type is .LIS. The default output is SYS$OUTPUT.

/SCHEMA
/NOSCHEMA
Specifies whether or not to dump the schema source. Specify /SCHEMA to request the schema source and /NOSCHEMA, the default, to suppress it.

/SECURITY_SCHEMAS [=(security-schema-name [,...])]
/NOSECURITY_SCHEMAS
Indicates which security schema source you want to dump, if any. The options are:

- /SECURITY_SCHEMAS [=(security-schema-name [,...])]

DBO dumps a security schema source list for each security schema you name. An error occurs if the specified security schema does not exist in the database.

- /SECURITY_SCHEMAS

  DBO dumps the DEFAULT_SECURITY_SCHEMA source if it exists in the database.

- /SECURITY_SCHEMAS=*

  DBO dumps a source list for every security schema in the database.

- /NOSECURITY_SCHEMAS

  DBO does not dump any security schema source. This is the default.

/SNAPSHOTS=(file-spec [,...])
/NOSNAPSHOTS
Specifies which snapshot files to dump. The options are:

- /SNAPSHOTS=(area-name,...)

  DBO dumps the snapshot files for each area you name. If you specify an area name for which snapshots are not allowed, an informational message is displayed.

- /SNAPSHOTS=*

  DBO dumps all snapshot files for the database.

- /NOSNAPSHOTS

  DBO does not dump any of the snapshot files. This is the default.

You can use /START and /END with this qualifier to dump specific pages. If you omit the /SNAPSHOTS qualifier, DBO does not dump snapshot files.

/STATE=option
Specifies a list of blocked users of distributed transactions. Use this qualifier in conjunction with the /USERS qualifier. Currently, BLOCKED is the only option available and displays the following information about each blocked distributed transaction:

- Process identification (PID)
- Run-unit identification
- Monitor identification
- Oracle CODASYL DBMS transaction identifier (TID)
- Name of the recovery-unit journal file
- Transaction sequence number (TSN)
- DECdtm transaction identifier
- Name of the node on which the failure occurred
- Name of the node initiating the transaction

Use the DBO/RESOLVE command to resolve blocked distributed transactions.

/START=integer
Specifies the range of area or snapshot pages to be dumped. The /START qualifier specifies the start of the range. By default, a dump starts with the first page and ends with the last page. You can override the default using the /START and

/END qualifiers to specify the page range. This qualifier works only if used with the /SNAPSHOTS or /AREAS qualifier.

/STORAGE_SCHEMA
/NOSTORAGE_SCHEMA
Specifies whether or not to dump the storage schema source list. Use /STORAGE_SCHEMA to produce a source list of the database storage schema. Use /NOSTORAGE_SCHEMA, the default, if you do not want to produce a storage schema source list.

/SUBSCHEMAS [=(subschema-name [,...])]
/NOSUBSCHEMAS
Indicates which subschema source you want to dump, if any. The options are:

- /SUBSCHEMAS [=(subschema-name [,...])]

  DBO dumps a subschema source list for each subschema you name. An error occurs if the specified subschema does not exist in the database.

- /SUBSCHEMAS

  DBO dumps the DEFAULT_SUBSCHEMA source if it exists in the database.

- /SUBSCHEMAS=*

  DBO dumps a source list for every subschema in the database.

- /NOSUBSCHEMAS

  DBO does not dump any subschema source. This is the default.

/USERS
Lists all active users of your database. If you are in a VMScluster environment, this list includes all VMScluster users. If used in conjunction with the /STATE qualifier, it lists blocked users of distributed transactions.

Examples

1.
```
$ DBO/DUMP PARTS/OUTPUT=DUMPPARTS
```

   This command dumps the root file header of database PARTS to the file DUMPPARTS.LIS.

2.
```
$ DBO/DUMP/SUBSCHEMA=PARTSS3/OPTION=DEBUG PARTS
```

   This command dumps the source list of subschema PARTSS3 with DEBUG commentary.

3.
```
$ DBO/DUMP PARTS/AREAS=(MAKE,BUY)-
_$ /END=10/SNAPSHOTS=(MAKE,BUY)
```

   This command dumps the first 10 pages of areas MAKE and BUY in the database PARTS, as well as the snapshot files for those two areas.

4.
```
$ DBO/DUMP/USERS/STATE=BLOCKED PARTS
Blocked user with process ID 20601875 (remote access server)
       Run Unit ID is 1
       Monitor ID is 1
       Transaction ID is 1
       Recovery journal filename is "DBM$DISK:[USERA]PARTS.RUJ;5"
       Transaction sequence number is 40
       DECdtm Tid is  00202078 0020207C 00202080 00202084
       Failure occurred on GREEN
       Parent node is YELLOW
```

   This command dumps the information for a remote blocked distributed transaction.

5.
```
$ DBO/DUMP/HEADER=(BRIEF,ROW_CACHE) PARTS

*----------------------------------------------------------------
* Oracle CODASYL DBMS V7.0-00             1-NOV-1996 10:43:52.19
*
* Dump of database header
*     Database: DISK1:[DBUSER]PARTS.ROO;1
*
*----------------------------------------------------------------

Database Parameters:
    Root filename is "DISK1:[DBUSER]PARTS.ROO;1"
    Row Caches...
     - Active row cache count is 2
     - Reserved row cache count is 7
     - Sweep interval is 1 second
     - Default cache file directory is database directory

Row cache "CACHE_1"
    Cache ID number is 1
    Cache-size in different sections of memory...
     - Total normal or system memory requirement is 284384 bytes
     - Total very large memory requirement...
        - Physical memory requirement is 284384 bytes
        - Virtual memory address space requirement is approximately
            153600 bytes

Row cache "CACHE_2"
    Cache ID number is 2
    Cache-size in different sections of memory...
     - Total normal or system memory requirement is 309984 bytes
     - Total very large memory requirement...
        - Physical memory requirement is 309984 bytes
        - Virtual memory address space requirement is approximately
          102400 bytes
```

   This command displays the record-level caching settings for the PARTS database. This is an abbreviated display of the settings. To view the full settings, use the DETAIL option in place of the BRIEF option with the /HEADER qualifier.

9.13.2 DBO/DUMP/AFTER_JOURNAL Command

Dumps a binary .AIJ file in an ASCII format.

DBO/DUMP/AFTER_JOURNAL Command

Format

DBO/DUMP/AFTER_JOURNAL aij-file-spec

Command Qualifiers	Defaults
/AREA=n	None
/ACTIVE_IO=max-reads	/ACTIVE_IO=3
/DATA	/NODATA
/ENCRYPT=(option[,...])	See description
/END=integer	/END=last-page
/FIRST=options	None
/FORMAT={OLD_FILE|NEW_TAPE}	/FORMAT=OLD_FILE
/LABEL=label-name-list	
/LAST=options	None
/LINE=n	None
/ONLY=options	None
/OPTIONS=[NO]STATISTICS	/OPTIONS=STATISTICS
/[NO]OUTPUT=file-spec	/OUTPUT=SYS$OUTPUT
/PAGE=n	None
/REWIND	/NOREWIND
/START=integer	/START=1
/STATE=option	/STATE=PREPARED

Description

The DBO/DUMP/AFTER_JOURNAL command specifies an .AIJ file as its parameter, not a root file. It is a separate command from the DBO/DUMP command used to dump database areas and root file header information.

DBO keeps the AIJ information in a binary file. The command translates the .AIJ file into an ASCII display format.

The dump always includes the header of the .AIJ file. You can use the /NODATA qualifier to exclude data blocks from the dump entirely or you can use the /START and /END qualifiers to restrict the data block dump to a specific series of blocks. If you omit these qualifiers, DBO includes all data blocks.

Parameter

aij-file-spec
Specifies an .AIJ file. The default file type is .AIJ.

Command Qualifiers

/ACTIVE_IO=max-reads
Specifies the maximum number of read operations from a backup device that the DBO/DUMP/AFTER_JOURNAL command will attempt simultaneously. This is not the maximum number of read operations in progress; that value is the product of active system I/O operations and the number of devices being written to simultaneously.

The value of the /ACTIVE_IO qualifier can range from 1 to 5. The default value is 3. Values larger than 3 can improve performance with some tape drives.

/AREA=integer
This qualifier is used together with /PAGE and /LINE to specify a DBKEY. Only AIJ records that match that DBKEY will be displayed. For example:

```
$ DBO/DUMP/AFTER/AREA=1/PAGE=2/LINE=1/NODAT/OPT=NOSTATISTICS RND0.AIJ
   .
   .
   .
379/1906      TYPE=D, LENGTH=36, TAD=31-JUL-1999 16:21:48.74 CSM=00
              TID=8730, TSN=0:699, AIJBL_START_FLG=01, FLUSH=00, SEQUENCE=60
               ERASE: PDBK=1:2:1, LDBID=0, PSN=0, FLAGS=00
```

/DATA
/NODATA
Specifies whether to dump AIJ data blocks or just the record headers.

Specify /DATA, the default, to dump one or more AIJ data blocks (in addition to the record headers) in an ASCII display format.

Specify /NODATA to dump only the record headers of the .AIJ file.

/ENCRYPT=({VALUE=|NAME=}[,ALGORITHM=])
The /ENCRYPT qualifier specifies the encryption key and algorithm so that the dump after image journal command can decrypt the save set file of a after image journal backup. Failure to use the /ENCRYPT qualifier when the database back is encrypted will result in an error: *%DBO-F-ENCRYPTSAVSET, save set is encrypted, /ENCRYPT must be specified*. Likewise, the correct encryption details must be provided or an error will be reported and the dump command will fail.

Specify a key value as a string or the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters type HELP ENCRYPT at the OpenVMS prompt.

This feature requires the OpenVMS Encrypt product to be installed and licensed on your system.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/END=integer
Specifies the number of the last data block to dump. The default integer is the number of the last data block in the file. This qualifier can be used together with /START to defined a range of blocks to process.

/FIRST=options
The /FIRST and /LAST qualifiers can be used either together or separately, to limit the range to be displayed, based on the keyword and its value (BLOCK, RECORD, TSN, TID, and TIME).

The BLOCK keyword will cause the DBO/DUMP /AFTER_JOURNAL command to display the AIJ blocks. The RECORD keyword will cause AIJ records to be displayed. You can also specify the TID, TSN, or TIME keywords to limit the range of the display. For example:

```
$ DBO/DUMP/AFTER RND0.AIJ;1 -
    /FIRST=(TIME="31-JUL-1999 16:17:17") -
    /LAST=(TIME="31-JUL-1999 16:17:19")
```

Be cautious when searching for TSNs or TIDs, as the AIJ file is not ordered by these values. For example, if you want to search for a specific TSN, then use the /ONLY qualifier, not the /FIRST or /LAST qualifier.

The options consists of a list of one or more of the following keywords:

- BLOCK=block#: Specifies the first or last block in the AIJ file.

- RECORD=record#: Specifies the first or last record in the AIJ file. This is the same as the existing /START and /END qualifiers, which are still supported, but deprecated.
- TID=tid: Specifies the first, last, or specific TID in the AIJ file.
- TIME=date_time: Specifies the first or last date/time in the AIJ file, using the standard date/time format.
- TSN=tsn: Specifies the first, last, or specific TSN in the AIJ file, using the standard [n:]m TSN format.

/FORMAT=OLD_FILE
/FORMAT=NEW_TAPE
Specifies whether or not the backup or optimized .AIJ file was written in the old (disk-optimized) or the new (tape-optimized) format. If you enter the DBO/DUMP/AFTER_JOURNAL command without the /FORMAT qualifier, the result is the old disk-optimized format. You must specify the same /FORMAT qualifier as was used with the DBO/BACKUP/AFTER_JOURNAL command or the DBO/OPTIMIZE/AFTER_JOURNAL command. If your .AIJ file resides on disk, you should use the /FORMAT=OLD_FILE qualifier.

If you specified the /FORMAT=OLD_FILE qualifier when you optimized or backed up the .AIJ file to tape, you must mount the backup media with the DCL MOUNT command before you issue the DBO/DUMP/AFTER_JOURNAL command. Because DBO will use RMS to read the tape, the tape must be mounted as an OpenVMS volume (that is, do not specify the /FOREIGN qualifier to the DCL MOUNT command).

If you specify the /FORMAT=NEW_TAPE qualifier, you must mount the backup media using the DCL MOUNT/FOREIGN command before you issue the DBO/DUMP/AFTER_JOURNAL command.

The following qualifiers have meaning only when used in conjunction with the /FORMAT=NEW_TAPE qualifier:

- /OPTIONS=STATISTICS
- /ACTIVE_IO
- /LABEL
- /REWIND

/LABEL=label-name-list
Specifies the 1- to 6-character string with which the volumes of the backup file were labeled. The /LABEL qualifier is applicable only to tape volumes. You must specify one or more label names when you use the /LABEL qualifier.

You can specify a list of tape labels for multiple tapes. If you list multiple tape label names, separate the names with commas and enclose the list of names within parentheses.

In a normal dump after-journal operation, the /LABEL qualifier you specify should be the same /LABEL qualifier you specified with the DBO/BACKUP/AFTER_JOURNAL command.

/LAST=option
The /FIRST and /LAST qualifiers can be used either together or separately, to limit the range to be displayed, based on the keyword and its value (BLOCK, RECORD, TSN, TID, and TIME).

The BLOCK keyword will cause the DBO/DUMP/AFTER_JOURNAL command to display the AIJ blocks. The RECORD keyword will cause AIJ records to be displayed. You can also specify the TID, TSN, or TIME keywords to limit the range of the display.

Be cautious when searching for TSNs or TIDs, as the AIJ file is not ordered by these values. For example, if you want to search for a specific TSN, then use the /ONLY qualifier, not the /FIRST or /LAST qualifier.

The options consists of a list of one or more of the following keywords:

- BLOCK=block#: Specifies the first or last block in the AIJ file.
- RECORD=record#: Specifies the first or last record in the AIJ file. This is the same as the existing /START and /END qualifiers, which are still supported, but deprecated.
- TID=tid: Specifies the first, last, or specific TID in the AIJ file.
- TIME=date_time: Specifies the first or last date/time in the AIJ file, using the standard date/time format.
- TSN=tsn: Specifies the first, last, or specific TSN in the AIJ file, using the standard [n:]m TSN format.

/LINE=integer

This qualifier is used together with /AREA and /PAGE to specify a DBKEY. Only AIJ records that match that DBKEY will be displayed.

/ONLY=options

The /ONLY qualifier can be used with either the TSN or TID keyword to specify that only the AIJ records associated with a specified TSN or TID will be displayed. For example:

```
$ DBO/DUMP/AFTER/OPT=NOSTAT/ONLY=(TSN=165) RND0.AIJ
   .
   .
   .
5/22             TYPE=D, LENGTH=246, TAD=31-JUL-1999 16:17:17.52, CSM=00
    TID=5456, TSN=0:165, AIJBL_START_FLG=01, FLUSH=01, SEQUENCE=2
        MODIFY: PDBK=1:8:0, LDBID=0, PSN=0, FLAGS=00, LENGTH=16

                            2001  0000  line 0 (1:8:0) SYSTEM record
                          01 000C  0002  12 bytes in 1 set/dynamic item
                          0007 0B  0005  11 bytes, storage set type 7
                          0291 0B  0008   next  1:49:1
                            01 01  000B   owner 1:8:1
                          0291 0B  000D   prior 1:49:1
   .
   .
   .
15/89            TYPE=R, LENGTH=14, TAD=31-JUL-1999 16:17:26.86, CSM=00
    TID=5456, TSN=0:165, AIJBL_START_FLG=01, FLUSH=01, SEQUENCE=12
```

The options consists of a list of one or more of the following keywords:

- TID=tid: Specifies the first, last, or specific TID in the AIJ file.
- TSN=tsn: Specifies the first, last, or specific TSN in the AIJ file, using the standard [n:]m TSN format.

/OPTION=STATISTICS
/OPTION=NOSTATISTICS
Specifies that you want DBO to include statistics on how frequently database pages are referenced by the data records in a backed up .AIJ file. In addition, if the database root file is available, the output created by the /OPTIONS=STATISTICS qualifier includes the value to specify for the /AIJ_BUFFERS qualifier of the DBO/RECOVER command. If several .AIJ files will be used in your recovery operation, perform a DBO/DUMP/AFTER_JOURNAL command on each .AIJ file and add the recommended AIJ_BUFFER values. Use the total as the value you specify with the /AIJ_BUFFERS qualifier.

If the database root file is not available, the /OPTION=STATISTICS qualifier does not provide a value for the /AIJ_BUFFERS qualifier of the DBO/RECOVER command. However, it does provide the statistics on the frequency with which each page is accessed.

The value specified for the /AIJ_BUFFERS qualifier does not guarantee that no buffers will be replaced during the AIJ rollforward. Buffer replacement strategy is affected by asynchronous prefetch, asynchronous batch writes, and the contents of the buffers before AIJ rollforward started. The /AIJ_BUFFERS recommendation gives the exact number of buffers required by the data records in the specific .AIJ file. Using fewer buffers means more I/O, but not necessarily worse performance. Using more buffers means your process may possibly do more paging, reducing performance.

Specify the /OPTION=NOSTATISTICS qualifier to suppress .AIJ file statistics generation.

Use the /OPTIONS=STATISTICS qualifier with the /NOOUTPUT qualifier if you only want the AIJ_BUFFER value, and not the .AIJ file information. Use the /OPTIONS=STATISTICS qualifier with the /OUTPUT=file-spec qualifier to write the statistical information to the specified output file. In any case, you must specify the /FORMAT=NEW_TAPE qualifier to display the statistical information (which means that the AIJ backup file must have been created with the /FORMAT=NEW_TAPE qualifier).

The default for the DBO/DUMP/AFTER_JOURNAL command is the /OPTION=STATISTICS qualifier.

/OUTPUT=file-spec
Names the file where output will be sent. The default file type is .LIS. The default output is SYS$OUTPUT.

/PAGE=integer
This qualifier is used together with /AREA and /LINE to specify a DBKEY. Only AIJ records that match that DBKEY will be displayed.

/REWIND
/NOREWIND
Specifies that the magnetic tape that contains the backup file will be rewound before processing begins. The /NOREWIND qualifier is the default and causes the dump file to start at the current logical end-of-tape (EOT).

This qualifier is applicable only to tape drives.

/START=integer
Specifies the number of the first data block to dump. The default is the first block, 1. This qualifier can be used together with /END to defined a range of blocks to process.

/STATE=option
Specifies a list of all records associated with blocked distributed transactions (transactions that have been prepared but not yet committed). Currently, PREPARED is the only option available.

Example

```
$ DBO/DUMP/AFTER_JOURNAL DISK1:[USER]PARTSJ
```

This command dumps the after-image journal file PARTSJ, located on DISK1 in the USER directory.

```
$ DBO/DUMP/AFTER_JOURNAL/STATE=PREPARED PARTS.AIJ
*------------------------------------------------------
* Oracle CODASYL DBMS V7.0-00      8-NOV-1996 12:32:38.31
*
* Dump of After Image Journal
*     Filename: DISK1:[USER]PARTS.AIJ;1
*
*------------------------------------------------------

1       TYPE=O, LENGTH=293, TAD=21-DEC-1993 15:08:01.36
    FACILITY="DBMSAIJ ", VER=60.1
    RT_TAD= 8-MAY-1989 10:49:10.98, COMMIT_TSN=72
    RT_FILNAM=DBM$DISK:[USER1]PARTS.ROO;12
2       TYPE=D, LENGTH=110, TAD=21-DEC-1993 15:08:02.04
3       TYPE=C, LENGTH=13, TAD=21-DEC-1993 15:08:02.11
4       TYPE=R, LENGTH=13, TAD=21-DEC-1993 15:08:17.68
5       TYPE=D, LENGTH=510, TAD=21-DEC-1993 15:15:43.83
6       TYPE=D, LENGTH=28, TAD=21-DEC-1993 15:15:43.83
7       TYPE=D, LENGTH=55, TAD=21-DEC-1993 15:15:43.83
8       TYPE=V, LENGTH=126, TAD=21-DEC-1993 15:15:43.94
    TSN=96

000000007E62CC83009340468A046A60        TID: ''j..F@...Ìb~....'
000000007E72F4E500933EA5BE0FD120        TM LOG_ID: ' Ñ.¥>..åôr~....'
00000000000000000000000120400B5E        RM LOG_ID: '^.@ ............'
202020325243535F534D424402440028        RM_NAME: '(.D.DBMS_SCR2   '
  0000000100010000011A0291000000        RM_NAME: '.............'
                      4E45455247        NODE NAME: 'GREEN'
                    574F4C4C4559        PARENT NODE NAME: 'YELLOW'
```

This example specifies a list of all records associated with blocked distributed transactions.

```
$ DBO/DUMP/AFTER PARTS.AIJ/FORMAT=NEW/OPTION=STATISTICS/NOOUTPUT
*---------------------------------------------------------------------
* Oracle CODASYL DBMS V7.0-00                  29-OCT-1996 16:24:00.80
*
* Dump of After Image Journal
*     Filename: DISK1:[DBUSER]AIJ_WORK0UPTPAHYHJV.AIJ_BCK;
*
*---------------------------------------------------------------------

Use "/AIJ_BUFFERS=4" when recovering this AIJ journal
```

```
2 recovery buffers referenced 2 times (2:2-6): 66.6%
2 recovery buffers referenced 1 time (2:12-16): 33.3%
Journal effectiveness: 60.0%

2 data records
6 data modification records
10 total modification records
1 commit record
```

This example specifies the /OPTIONS=STATISTICS qualifier to display the value to specify with the /AIJ_BUFFERS qualifier to the DBO/RECOVER command. It also shows how frequently each recovery buffer was accessed.

```
$ DBO/DUMP/AFTER PARTS.AIJ/FORMAT=NEW/OPTION=STATISTICS/NOOUTPUT
*-------------------------------------------------------------------------
* Oracle CODASYL DBMS V7.0-00                    29-OCT-1996 16:25:06.98
*
* Dump of After Image Journal
*     Filename: DISK1:[DBUSER]AIJ_WORK0UM3BZ0YHJV.AIJ_BCK;
*
*-------------------------------------------------------------------------

Database rootfile not available; displaying page information
5 pages affected by this AIJ journal

1 page referenced 2 times (1:50): 33.3%
4 pages referenced 1 time (2:2): 66.6%
Journal effectiveness: 60.0%

2 data records
6 data modification records
10 total modification records
1 commit record
```

This example is the same as the previous example, except in this case the database root file is not available. Therefore, Oracle CODASYL DBMS is not able to provide a suggested value for the DBO/RECOVER command's /AIJ_BUFFERS qualifier, but does provide statistics on how frequently pages were referenced. Namely, pages (2:2-6) were referenced 2 times, which means that 66.6% of all data records in the dumped after-image journal file reference these pages.

9.13.3 DBO/DUMP/BACKUP Command

Dumps header block and information about each storage area contained in the backup file. This can only be used for single-threaded backups.

Format

DBO/DUMP/BACKUP backup-file-spec

Command Qualifier	Default
/HEADER	See description

Description

The DBO/DUMP/BACKUP command specifies a backup file as its parameter, not a root file. It is a separate command from the DBO/DUMP command used to dump database areas and root file header information.

Parameter

backup-file-spec
Specifies a backup file. The default file type is .DBB.

Command Qualifier

/HEADER
Lists only the database header information in the output.

Example

```
$ DBO/DUMP/BACKUP DISK1:[USER]PARTS_BCK.DBB
```

This command dumps the backup file PARTS_BCK.DBB, located on DISK1 in the USER directory.

9.13.4 DBO/DUMP/BACKUP/MULTITHREAD Command

Dumps information contained within a backup file created using the DBO/BACKUP/MULTITHREAD command.

Format

DBO/DUMP/BACKUP/MULTITHREAD backup-file-spec

Command Qualifiers	Defaults
/ACTIVE_IO=integer	See description
/BUFFER_SIZE=integer	
/ENCRYPT=(option[,...])	
/LABEL=(label[,...])	
/OPTIONS=(option[,...])	
/OUTPUT=file-spec	SYS$OUTPUT
/PROCESS=(option[,...])	
/RESTORE_OPTIONS=file-spec	No file written
/[NO]REWIND	/NOREWIND
/SKIP=(option[,...])	

Description

The DBO/DUMP/BACKUP/MULTITHREAD command specifies a backup file as its parameter, not a root file. It is a separate command from the DBO/DUMP command used to dump database areas and root file header information and the DBO/DUMP/BACKUP command used to dump single-threaded database backup files.

Parameter

backup-file-spec
Specifies a backup file. The default file type is .DBF.

Command Qualifier

/ACTIVE_IO=integer
Specifies the maximum number of read operations to the backup file that the process attempts simultaneously. The maximum number of read operationss may range from 1 to 7. For TA90 and TA90E tape drives, the maximum value is 7 and the default is 5. For all other tape drives the maximum value is 5 and the default is 3. TA90, TA90E, and TF857 tape drives and stack loaders are supported.

/BUFFER_SIZE=integer
Specifies the maximum record size for the backup file. The size can range from 2048 to 65,024 bytes. In general, you should not override the record size selected by multithreaded backup as it determines a size that is efficient for the tape label being used.

/ENCRYPT=({VALUE=|NAME=}[,ALGORITHM=])
Specify a key value as a string or, the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters see HELP ENCRYPT.

The /ENCRYPT qualifier specifies the encryption key and algorithm so that the dump command can decrypt the save set file of a database backup. Failure to use the /ENCRYPT qualifier when the database back is encrypted will result in an error: *%DBO-F-ENCRYPTSAVSET, save set is encrypted, /ENCRYPT must be specified*. Likewise, the correct encryption details must be provided or an error will be reported and the dump command will fail.

This feature requires the OpenVMS Encrypt product to be installed and licensed on this system.

/LABEL=(label[,...])
Specifies the labels of the backup tape volumes to be processed. You can specify a string from 1 to 6 characters long.

/OPTIONS=(option[,...])
Specifies options for the dump process. The following are valid options:

- BLOCKS

  Used without the DATA option, the BLOCKS option dumps the backup file's block structure only.

- DATA

  In conjunction with BLOCKS and/or RECORDS options, the DATA option dumps the contents of the backup file's blocks and/or records.

- FULL

  Equivalent to specifying ROOT, BLOCKS, and RECORDS. The contents of the backup file's root file, block structure, and record structure are dumped when FULL is specified.

- MEDIA_CHECK

  The backup file is read and its integrity verified. Useful in detecting media errors. No dump output is generated unless problems are found. This is always done unless /PROCESS or /SKIP is specified. Output may be generated if other options are specified.

- NORMAL

Dumps the root file and area information only. This is the default.

- RECORDS

  Used without the DATA option, the RECORDS option dumps the backup file's record structure only.

- ROOT=(option,...)

  Dumps all the root file contents or, optionally, only the schemas, subschemas, security schemas, or storage schemas. Options are:

  - ALL

    Dumps the database root file contents as recorded in the backup file.

  - SCHEMA

    Dumps only the schemas in the database root file.

  - SUBSCHEMA

    Dumps only the subschemas in the database root file.

  - SECURITY_SCHEMA

    Dumps only the security schemas in the database root file.

  - STORAGE_SCHEMA

    Dumps only the storage schemas in the database root file.

/OUTPUT=file-spec

Specifies output be written to the specified file. The default file type is .LIS. The default output is SYS$OUTPUT.

/PROCESS=(option[,...])

Specifies the number of BLOCKS, RECORDS, and/or VOLUMES of the backup file to be dumped by the dump process. Valid options include:

- BLOCKS=integer

  Specifies the number of BLOCKS to dump

- RECORDS=integer

  Specifies the number of RECORDS to dump

- VOLUMES=integer

  Specifies the number of VOLUMES to dump

/RESTORE_OPTIONS=file-spec

Generates an options file designed to be used with the /OPTIONS qualifier of the DBO/RESTORE command. The file generated by the /RESTORE_OPTIONS qualifier references:

- All storage areas for a full backup operation
- Only those storage areas specified for a by-area backup operation

The restore options file is created after the root information has been read from the backup file.

By default, a /RESTORE_OPTIONS qualifier is not created. If you specify the /RESTORE_OPTIONS qualifier and a file specification, but not a file type, DBO uses the file type .OPT, by default.

/REWIND
/NOREWIND
Specifies /REWIND to have the tape containing the backup file be rewound before processing begins. /NOREWIND is the default.

The /REWIND and /NOREWIND qualifiers are applicable only to tape devices. You should use these qualifiers only when the target device is a tape device.

/SKIP=(option[,...])
Specifies the number of BLOCKS, RECORDS, and/or VOLUMES of the backup file to be ignored before the dump process begins dumping the file. Valid options include:

- BLOCKS=integer

  Specifies the number of BLOCKS to skip

- RECORDS=integer

  Specifies the number of RECORDS to skip

- VOLUMES=integer

  Specifies the number of VOLUMES to skip

Examples

1.
```
$ MOUNT/FOREIGN $111$MUA2:
%MOUNT-I-MOUNTED, PARTS mounted on _$111$MUA2: (STAR)
$ DBO/DUMP/BACKUP/MULTI $111$MUA2:PARTS
Database Parameters:
    Root filename is "DBMS$DISK:[PARTS]PARTS.ROO;1"
   .
   .
   .
    Buffers...
      - Global buffers are disabled
   .
   .
   .
Storage area
   .
   .
   .
    Area ID number is 1
    Filename is "DBMS$DISK:[PARTS]MAKE.DBS;1"
   .
   .
   .
Snapshot area for storage area
    Area ID number is 5
    Filename is "DBMS$DISK:[PARTS]MAKE.SNP;1"
   .
   .
   .
$ DISMOUNT $111$MUA2:
```

This example dumps the root information contained within the backup file of the PARTS database created with the DBO/BACKUP/MULTITHREAD command.

2.
```
$ DBO/DUMP/BACKUP/MULTITHREAD PARTS.DBF /RESTORE_OPTIONS=PARTS.OPT-
_$ /OPTIONS=NORMAL/OUTPUT=DUMP.LIS
$ TYPE PARTS.OPT

  Options file for database DISK1:[DBUSER]PARTS.ROO;1
  Created  4-NOV-1996 14:42:04.10
  Created by DUMP BACKUP command

MAKE -
        /file=DISK2:[AREA]MAKE.DBS;1 -
        /blocks_per_page=2 -
        /extension=ENABLED -
        /read_write -
        /space -
        /thresholds=(70,85,95) -
        /snapshot=(allocation=2, -
                   file=DISK3:[SNAPS]MAKE.SNP;1)

BUY -
        /file=DISK3:[AREA]BUY.DBS;1 -
        /blocks_per_page=2 -
        /extension=ENABLED -
        /read_write -
        /space -
        /thresholds=(70,85,95) -
        /snapshot=(allocation=2, -
                   file=DISK4:[SNAPS]BUY.SNP;1)

$ DBO/RESTORE/MULTITHREADED/AREA PARTS.DBF/OPTIONS=PARTS.OPT
```

This example demonstrates the use of the /RESTORE_OPTIONS qualifier. The first command performs a dump operation on the backup file of the PARTS database and creates a restore options file. The second command shows the contents of the options file. The last command demonstrates the use of the options file with the DBO/RESTORE/MULTITHREADED command.

9.13.5 DBO/DUMP/CACHE_FILE Command

Dumps information contained within a cache backing store file.

Format

DBO/DUMP/CACHE_FILE cache-filename

Command Qualifiers	Defaults
/[NO]DATA	/DATA
/OUTPUT=file-spec	SYS$OUTPUT

Description

This command displays or writes to a specified output file information from cache files, and refines what is listed depending upon the use of the /[NO]DATA qualifier.

A row cache backing store file name has a format similar to the following:

```
personnel_10_0.DBC_0C1H85848NO00063228L;1
```

0C1H85848NO00063228L represents the device name and file ID of the root file for the database. The file type is always prefixed with .DBC_. All row cache backing store files for a database have this same exact file type. Another database using the same location for cache backing store files would use a different file type (perhaps .DBC_4D87HD234FSD0063228L).

Parameters

cache-filename
File specification of the cache backing store file to dump.

Command Qualifiers

/DATA
/NODATA
The /[NO]DATA qualifier specifies whether or not you want to display data blocks of the cache file, or just the cache file header.

The /DATA qualifier is the default. It causes the display of the cache file data blocks in addition to the file header, in an ASCII display format. The /NODATA qualifier limits the display to the record headers of the cache file.

/OUTPUT
The /OUTPUT qualifier specifies the file to which the cache data will be written. The default output file location is SYS$OUTPUT. The default file type is .LIS.

9.13.6 DBO/DUMP/EXPORT Command

Dumps information contained within a metadata file.

Format

DBO/DUMP/EXPORT metadata-file-spec

Command Qualifiers	Defaults
/[NO]HEADER	See description
/OPTIONS=(type [,...])	/OPTIONS=NORMAL
/OUTPUT=file-spec	SYS$OUTPUT
/[NO]SCHEMA	/NOSCHEMA
/[NO]SECURITY_SCHEMAS [=(security-schema-name[,...])]	/NOSECURITY_SCHEMAS
/[NO]STORAGE_SCHEMA	/NOSTORAGE_SCHEMA
/[NO]SUBSCHEMAS [=(subschema-name [,...])]	/NOSUBSCHEMAS

Description

Depending on your selection of qualifiers, the DBO/DUMP/EXPORT command lists:

- Metadata header information listed by default.
- Any or all DDL source code the database uses. This output is identical to that of the DBO/EXTRACT command except that the source code is taken from the metadata file instead of the Oracle CDD/Repository dictionary.

Parameters

metadata-file-spec
Specifies the metadata file whose metadata you want to dump. The default file type is .DBM.

Command Qualifiers

/HEADER
/NOHEADER
Indicates whether or not to include a list of the metadata header information in the output.

Specify /HEADER to request dumping the metadata header from the metadata file. Specify /NOHEADER to suppress the metadata header list.

If the DBO/DUMP command does not call for a DDL source code list, a storage dump, a list of database file specifications, or a dump of snapshot files, the /HEADER qualifier is the default. Otherwise, /NOHEADER is the default.

/OPTIONS=(type [,...])
Specifies the level of information the output will include about schemas, storage schemas, subschemas, security schemas, and database headers. Three types of output are available:

- NORMAL

  Output includes only the DDL source code information. This is the default.

- FULL

  In addition to the NORMAL information, output includes commentary and set participation information.

- DEBUG

  In addition to NORMAL and FULL information, output includes internal information about the data. You can identify internal area, set record, and item numbers using the DEBUG option. In general, DEBUG information is useful only for diagnostic support purposes.

/OUTPUT=file-spec
Names the file where output will be sent. The default file type is .LIS. The default output is SYS$OUTPUT.

/SCHEMA
/NOSCHEMA
Specifies whether or not to dump the schema source. Specify /SCHEMA to request the schema source and /NOSCHEMA, the default, to suppress it.

/SECURITY_SCHEMAS [=(security-schema-name [,...])]
/NOSECURITY_SCHEMA
Indicates which security schema source you want to dump, if any. The options are:

- /SECURITY_SCHEMAS [=(security-schema-name [,...])]

  DBO dumps the security schema source list for each security schema you name. An error occurs if the specified security schema does not exist in the metadata file.

- /SECURITY_SCHEMAS

DBO dumps the DEFAULT_SECURITY_SCHEMA source if it exists in the metadata file.

- /SECURITY_SCHEMAS=*

  DBO dumps a source list for every security schema in the metadata file.

- /NOSECURITY_SCHEMAS

  DBO does not dump any security schema source. This is the default.

/STORAGE_SCHEMA
/NOSTORAGE_SCHEMA
Specifies whether or not to dump the storage schema source list. Use /STORAGE_SCHEMA to produce a source list of the database storage schema. Use /NOSTORAGE_SCHEMA, the default, if you do not want to produce a storage schema source list.

/SUBSCHEMAS [=(subschema-name [,...])]
/NOSUBSCHEMAS
Indicates which subschema source you want to dump, if any. The options are:

- /SUBSCHEMAS [=(subschema-name [,...])]

  DBO dumps a subschema source list for each subschema you name. An error occurs if the specified subschema does not exist in the metadata file.

- /SUBSCHEMAS

  DBO dumps the DEFAULT_SUBSCHEMA source if it exists in the metadata file.

- /SUBSCHEMAS=*

  DBO dumps a source list for every subschema in the metadata file.

- /NOSUBSCHEMAS

  DBO does not dump any subschema source. This is the default.

Examples

1.
```
$ DBO/DUMP/EXPORT/SUB=PARTSS1/OPTIONS=NORMAL EXPORTED_PARTS.DBMS
*------------------------------------------------------------
* Oracle CODASYL DBMS V7.0-00          11-NOV-1996 14:38:10.32
*
* Dump of metadata file subschema
*     Filename: DBM$DISK:[SMITH]EXPORTED_PARTS.DBM;1
*
*------------------------------------------------------------

SUBSCHEMA NAME IS PARTSS1 FOR PARTS SCHEMA

ALIAS RECORD DIVISION IS WK_GROUP

ALIAS ITEM DIV_NAME OF DIVISION IS GROUP_NAME

ALIAS RECORD QUOTE IS PR_QUOTE
```

```
REALM BUY
    IS BUY
   .
   .
   .
RECORD NAME IS CLASS
    ITEM CLASS_CODE TYPE IS CHARACTER 2
   .
   .
   .
```

This command dumps the metadata file for the EXPORTED_PARTS metadata file.

2. `$ DBO/DUMP/EXPORT/SUB=PARTSS1/OPTIONS=DEBUG XPARTS.DBMS`

```
*-------------------------------------------------------------
* Oracle CODASYL DBMS V7.0-00           1-NOV-1996 14:39:27.86
*
* Dump of metadata file subschema
*    Filename: DBM$DISK:[SMITH]EXPORTED_PARTS.DBM;1
*
*-------------------------------------------------------------

*   Block addr: 004BED68, DDBLK offset : 00000000 (0.)
SUBSCHEMA NAME IS PARTSS1 FOR PARTS SCHEMA
*  Subschema version number : 5-SEP-1996 17:36:44.76
*  Schema version number : 5-SEP-1996 17:36:33.78 (1)
*  Currency Indicator : 0
*  Block addr: X'001FD330', offset (DDBLK section): X'00000000' (0.)

ALIAS RECORD DIVISION IS WK_GROUP

ALIAS ITEM DIV_NAME OF DIVISION IS GROUP_NAME

ALIAS RECORD QUOTE IS PR_QUOTE


*   Block addr: 004BEDB4, DDBLK offset : 0000004C (76.)
REALM BUY
*   Subschema realm ID : 1 (0001)
   IS BUY
*  User Id : 1
*  Currency Indicator : 1

*   Block addr: 004BEDC8, DDBLK offset : 00000060 (96.)
   .
   .
   .
RECORD NAME IS CLASS
*   Subschema record ID : 1 (0001)
*  Within areas   : MAKE
*                   BUY
*  Owner of sets  : CLASS_PART
*  Member of sets : ALL_CLASS
*  Corresponding schema record type : CLASS
*  Currency Indicator : 5
*  Maximum length : 23 bytes

*   Block addr: 004BEE04, DDBLK offset : 0000009C (156.)
   ITEM CLASS_CODE TYPE IS CHARACTER 2
*       Subschema item/group ID : 1 (0001)
*       DSC addr: 004BFBA4, DDNAME offset : 0000003C (60.)
*      0 bytes from start of record
*      Descriptor length field: 2
*      Part of subschema record type : CLASS
*      Corresponding schema data item : CLASS_CODE
```

```
*   Block addr: 004BEE38, DDBLK offset : 000000D0 (208.)
    .
    .
    .
```

This command dumps the source list of the EXPORTED_PARTS metadata file with DEBUG commentary.

9.13.7 DBO/DUMP/RECOVERY_JOURNAL Command

Dumps a binary recovery-unit journal (.RUJ) file in an ASCII format.

Format

DBO/DUMP/RECOVERY_JOURNAL ruj-file-spec

Command Qualifiers	Defaults
/[NO]DATA	/DATA
/OUTPUT=file-spec	SYS$OUTPUT

Description

The DBO/DUMP/RECOVERY_JOURNAL command specifies an .RUJ file as its parameter, not a root file. It is a separate command from the DBO/DUMP command used to dump database areas and root file header information.

DBO keeps the RUJ information in a binary file. This command translates the .RUJ file into an ASCII display format.

Parameter

ruj-file-spec
Specifies an .RUJ file. The default file type is .RUJ.

Command Qualifiers

/DATA
/NODATA
Specifies whether to dump RUJ data blocks or just the record headers.

Specify /DATA, the default, to dump one or more RUJ data blocks (in addition to the file header) in an ASCII display format.

Specify /NODATA to dump only the record headers of the .RUJ file.

/OUTPUT=file-spec
Names the file where output will be sent. The default file type is .LIS. The default output is SYS$OUTPUT.

Example

```
$ DBO/DUMP/RECOVERY_JOURNAL DISK1:[USER]PARTSRC
```

This command dumps the recovery-unit journal file PARTSRC, located on DISK1 in the USER directory.

9.14 DBO/EXPORT Command

Copies data definitions from an existing database root file to a specified metadata file.

Format

DBO/EXPORT root-file-spec metadata-file-spec

Command Qualifiers	Defaults
/[NO]LOG	Current DCL verify value
/[NO]SECURITY_SCHEMAS [=security-schema-name[,...]]	/SECURITY_SCHEMAS=*
/[NO]SUBSCHEMAS [=subschema-name[,...]]	/SUBSCHEMAS=*

Description

The DBO/EXPORT command creates a simple RMS sequential file or metadata file containing the metadata from an existing database root file. By default, the schema, storage schema, all security schemas, and all subschemas are exported. The metadata file can then be distributed to a remote site for inclusion into local database instances.

Parameters

root-file-spec
Specifies the root file of the database from which you want to create a metadata file. The default file type is .ROO.

metadata-file-spec
Specifies the metadata file for the database. The default file type is .DBM. If you do not provide a full file specification for the export file, the missing components (the device and directory) will be derived from the root file specification.

Command Qualifiers

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request log output and /NOLOG to prevent this reporting. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/SECURITY_SCHEMAS [=(security-schema-name [,...])]
/NOSECURITY_SCHEMAS
Specifies the security schema metadata to be copied. Form variants are:

- /SECURITY_SCHEMAS [=(security-schema-name [,...])]

  Copies one or more security schemas from the root file to the metadata file.

- /SECURITY_SCHEMAS

  Copies the DEFAULT_SECURITY_SCHEMA from the root file to the metadata file.

- /SECURITY_SCHEMAS=*

Copies all security schemas from the root file to the metadata file.

- /NOSECURITY_SCHEMAS

  Specifies that no security schemas are to be copied.

/SUBSCHEMAS [=(subschema-name [,...])]
/NOSUBSCHEMAS
Specifies the subschema metadata to be copied. Form variants are:

- /SUBSCHEMAS [=(subschema-name [,...])]

  Copies one or more subschemas from the root file to the metadata file.

- /SUBSCHEMAS

  Copies the DEFAULT_SUBSCHEMA from the root file to the metadata file.

- /SUBSCHEMAS=*

  Copies all subschemas from the root file to the metadata file.

- /NOSUBSCHEMAS

  Specifies that no subschemas are to be copied.

Example

```
$ DBO/EXPORT/LOG PARTS EXPORTED_PARTS
%DBO-I-LOGEXPDDL, exported schema PARTS
%DBO-I-LOGEXPDDL, exported storage schema PARTST1
%DBO-I-LOGEXPDDL, exported subschema PARTSS3
%DBO-I-LOGEXPDDL, exported subschema PARTSS2
%DBO-I-LOGEXPDDL, exported subschema PARTSS4
%DBO-I-LOGEXPDDL, exported subschema PARTSS1
%DBO-I-LOGEXPDDL, exported security schema DEFAULT_SECURITY_SCHEMA
```

This command creates a metadata file for the PARTS database with the file name EXPORTED_PARTS.DBM.

9.15 DBO/EXTRACT Command

Extracts DDL source information from Oracle CDD/Repository and produces DDL source files.

Format

DBO/EXTRACT schema-name [,...]

Command Qualifiers	Defaults
/OPTIONS=type [,...]	/OPTIONS=NORMAL
/OUTPUT=file-spec	SYS$OUTPUT
/[NO]SCHEMA	See description
/[NO]SECURITY_SCHEMAS [=(security-schema-name[,...])]	/NOSECURITY_SCHEMAS
/[NO]STORAGE_SCHEMAS [=(storage-schema-name[,...])]	/NOSTORAGE_SCHEMAS
/[NO]SUBSCHEMAS [=(subschema-name[,...])]	/NOSUBSCHEMAS

Description

Depending on your selection of qualifiers, DBO/EXTRACT can produce a source file for the schema you specify, along with source files for any or all storage schemas, security schemas, and subschemas stored under that schema in Oracle CDD/Repository. This output shows how data was defined in the original source file.

An extracted source file often provides a good baseline file for reformulating data definitions. For example, you can use a text editor to modify a default storage schema or subschema into a more optimal definition.

The DBO/DUMP command can also produce source files. DBO/DUMP extracts source from the root file, not from Oracle CDD/Repository.

Parameter

schema-name [,...]
Identifies one or more schemas in Oracle CDD/Repository from which to extract source. Each schema name must follow the rules for forming Oracle CDD/Repository pathnames. If you specify a schema name without a full Oracle CDD/Repository pathname, Oracle CODASYL DBMS searches your CDD$DEFAULT directory for the schema.

Command Qualifiers

/OPTIONS=type [,...]
Specifies the type and amount of information the output will include. Three types of output are available:

- NORMAL

  Output includes only the source language information. This is the default.

- FULL

  In addition to the NORMAL information, output includes commentary, such as cross-checking information about set participation for records.

- DEBUG

  In addition to NORMAL and FULL information, output includes internal information about the data. In general, this information is useful for diagnostic support purposes.

If you specify more than one type of /OPTIONS output, you must enclose the options in parentheses:

```
/OPTION=(NORMAL,FULL,DEBUG)
```

This produces the same result as:

```
/OPTION=DEBUG
```

/OUTPUT=file-spec
Names the file where output will be sent. The default file type is .LIS. The default output is SYS$OUTPUT.

/SCHEMA
/NOSCHEMA
Indicates whether or not to extract the schema source. Specify /SCHEMA to request the schema DDL and /NOSCHEMA to suppress it.

If you do not request at least one security schema, storage schema, or subschema source output, /SCHEMA is the default. However, if you request a security schema, storage schema, or subschema DDL, /NOSCHEMA is the default.

/SECURITY_SCHEMAS [=(security-schema-name [,...])]
/NOSECURITY_SCHEMAS
Indicates which security schema source you want to extract, if any. Form variants are:

- /SECURITY_SCHEMAS [=(security-schema-name [,...])]

  DBO produces a source file containing all the security schemas you name. An error results if the specified security schema does not exist in Oracle CDD/Repository.

- /SECURITY_SCHEMAS

  DBO produces a source file containing the DEFAULT_SECURITY_SCHEMA if it exists in Oracle CDD/Repository.

- /SECURITY_SCHEMAS=*

  DBO produces a source version of every security schema currently loaded for the schema.

- /NOSECURITY_SCHEMAS

  DBO does not produce any security schema source. This is the default.

/STORAGE_SCHEMAS [=(storage-schema-name [,...])]
/NOSTORAGE_SCHEMAS
Indicates which storage schema DDLs you want to extract, if any. Form variants are:

- /STORAGE_SCHEMAS [=(storage-schema-name [,...])]

  DBO produces a storage schema source file containing all storage schemas you name. An error results if there is no such storage schema in Oracle CDD/Repository.

- /STORAGE_SCHEMAS

  DBO produces a storage schema source file containing the DEFAULT_STORAGE_SCHEMA if it exists in Oracle CDD/Repository.

- /STORAGE_SCHEMAS=*

  DBO produces a source list of every storage schema currently stored under the schema.

- /NOSTORAGE_SCHEMAS

  DBO does not produce any storage schema DDL. This is the default.

/SUBSCHEMAS [=(subschema-name [,...])]
/NOSUBSCHEMAS
Indicates which subschema source lists you want to extract, if any. Form variants are:

- /SUBSCHEMAS [=(subschema-name [,...])]

  DBO produces a source file containing all the specified subschemas. An error results if the specified subschema does not exist in Oracle CDD/Repository.

- /SUBSCHEMAS

DBO produces a source file containing the DEFAULT_SUBSCHEMA if it exists in Oracle CDD/Repository.

- /SUBSCHEMAS=*

  DBO produces a source version of every subschema currently loaded for the schema.

- /NOSUBSCHEMAS

  DBO does not produce any subschema source. This is the default.

Example

```
$ DBO/EXTRACT/SCHEMA/OUTPUT=PARTSOUT/SUBSCHEMAS=(PARTSS1,PARTSS2) PARTS
```

This command extracts NORMAL source output from Oracle CDD/Repository for the schema PARTS and for two subschemas under the schema PARTS. DBO writes the source to the file PARTSOUT.DDL.

9.16 DBO/GRANT_COMMAND Commands

Adds, deletes, or lists entries in the command authorization lists (CALs) of a database.

Format

DBO/GRANT_COMMAND [/action] root-file-spec [command-type [,...]]

Action Qualifiers	Defaults
/ADD	
/DELETE	
/LIST	

Description

The DBO/GRANT_COMMAND provides the only means of accessing or changing the security mechanisms for DBO commands that can be secured. For a list of securable DBO commands see the *Oracle CODASYL DBMS Database Security Guide*. To secure access there is one OpenVMS access control list (ACL) for each command. Each CAL consists of a set of access control entries (ACEs). A CAL is a root file structure whose purpose is to restrict the use of a securable DBO command to specific users. Each database contains CALs in its root file. Specifically, a CAL is a list of user identification codes (UICs) and OpenVMS identifiers. Each ACE is a UIC or OpenVMS identifier value. A user cannot execute a securable DBO command on the database unless an ACE for that user exists in the CAL for that command or unless the CAL is empty.

Immediately after database creation, the CAL for each securable DBO command contains the UIC of the owner (the user who issued the DBO/CREATE command). This limits only the owner to securable DBO commands until the CALs are altered.

9.16 DBO/GRANT_COMMAND Commands

Each DBO/GRANT_COMMAND instance performs one of three types of action: listing CAL entries, adding CAL entries, or deleting CAL entries. An action qualifier specifies which of these actions will be performed. Each action qualifier requires a distinct command format. These three formats are described in the following subsections.

Parameters

root-file-spec
Specifies the root file of the database whose CALs you want to reference. The default file type is .ROO.

command-type [,...]
Specifies one or more DBO commands whose CAL you want to list or change. Valid command-type values are listed in Table 9–4.

Table 9–4 DBO Command Types

ALTER	EXPORT	OPEN
ANALYZE	GRANT_COMMAND	PERMIT_USER
AUDIT	INITIALIZE	RECOVER
BACKUP	INTEGRATE	RESOLVE
CLOSE	LOAD	RESTORE
COPY_DATABASE	MANAGER	UNLOAD
DELETE	MODIFY	USER
DUMP	MOVE_AREA	VERIFY

The command-type USER refers to the set of DBO commands that cannot change the physical database, display data pages, or provide the means to do either. The USER commands are ANALYZE and VERIFY.

The command-type MANAGER refers to all securable DBO commands.

The DBO commands CONVERT, CREATE, DUMP/EXPORT, EXTRACT, MONITOR, REPORT, SHOW, and WORK_AREA are not securable. Because the CALs are part of the root file, DBO commands that do not access the database cannot be secured. However, other means of protection exist for these commands. CREATE, EXTRACT, REPORT, and WORK_AREA access Oracle CDD/Repository metadata, which is protected by Oracle CDD/Repository security mechanisms. MONITOR requires WORLD user privilege, which provides adequate protection. CONVERT can be used only with BYPASS privilege.

Action Qualifiers

/ADD
/DELETE
/LIST
Specifies the type of operation to be performed on the CAL of each command-type specified. You must select only one operation qualifier and position it before all command-type parameters.

The /ADD qualifier adds one or more new ACEs to the CAL specified. You must use the /IDENTIFIER qualifier to specify the UICs for each new ACE. Users with those UICs will be allowed to perform the command for the specified command-type.

See Section 9.16.1 for a complete description of the DBO/GRANT_ COMMAND/ADD command.

The /DELETE qualifier deletes one or more ACEs from the CAL specified. You can specify ACEs by using the /IDENTIFIER qualifier (specifying the UIC values of the ACEs you want to delete) or the /ENTRY qualifier (specifying the ordinal position of the ACE in the CAL). Deleting an ACE denies use of the specified command-type to users with those UICs, provided the deletion does not leave the CAL empty.

See Section 9.16.2 for a complete description of the DBO/GRANT_ COMMAND/DELETE command.

The /LIST qualifier is the default operation. It lists to SYS$OUTPUT all ACEs for the specified CAL. If you prefer, you can write the ACE list to a file by specifying /OUTPUT=file-spec.

See Section 9.16.3 for a complete description of the DBO/GRANT_ COMMAND/LIST command.

9.16.1 DBO/GRANT_COMMAND/ADD Command

Adds entries to the CALs of the database according to parameters and qualifiers you specify.

Format

DBO/GRANT_COMMAND/ADD root-file-spec command-type [,...]

Command Qualifier	Default
/[NO]LOG	Current DCL verify value

Parameter Qualifier	Defaults
/ENTRY=integer	/ENTRY=1
/IDENTIFIER=option	

Description

This command adds one or more ACEs to all or to selected CALs. Use the /IDENTIFIER qualifier to specify UICs or OpenVMS identifiers of the users who will be granted the use of the command. You can specify the location of the new CAL entries using the /ENTRY qualifier.

Parameters

root-file-spec
Specifies the root file of the database whose CALs you want to reference. The default file type is .ROO.

command-type [,...]
Specifies one or more DBO commands that refer to one or more CALs to which you want to add entries. See Table 9–4 for a list of valid command types.

Command Qualifier

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT.

Specify /LOG to request that the DBO/GRANT_COMMAND/ADD command list each CAL entry addition; specify /NOLOG to prevent the list. If you specify neither qualifier, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

Parameter Qualifiers

/ENTRY=integer
Specifies an ordinal position in the CAL where you intend to add an ACE.

Positioning of entries is critical when wildcard UICs or OpenVMS identifiers exist in the CAL. Because OpenVMS searches the ACL from the beginning until it finds an ACE matching the user, it is good practice to keep ACEs for individual UICs or OpenVMS identifiers ahead of wildcard ACE entries. If a wildcard ACE came before an ACE for a specific user, that user would always be matched to the wildcard ACE. Use DBO/GRANT_COMMAND/LIST to see the positions of the entries already present.

When you add an entry at position N, the position numbers of existing entries in the CAL that are greater than or equal to N are incremented by 1. For example, if the CAL contains five entries (numbered 1 through 5) and you add an entry specifying /ENTRY=3, the new entry becomes entry 3 and the entries previously numbered 3 through 5 become entries 4 through 6.

If you do not specify an /ENTRY position, the new ACE becomes the first entry of the CAL.

If you specify an integer larger than the existing number of entries, the new entry becomes the last entry of the CAL. DBO reduces the entry number if necessary. After each entry addition, the highest entry number in the CAL equals the number of entries in that CAL.

/IDENTIFIER=(UIC,...)
/IDENTIFIER=(VMS_ID,...)
Identifies one or more users by their UIC or OpenVMS identifier. When adding one or more ACEs to a CAL, you must use the /IDENTIFIER qualifier to identify the set of users who will be allowed to execute the command to which the CAL pertains.

If a UIC or OpenVMS identifier string contains duplicate values, an informational message results and DBO ignores the duplicates. Nevertheless, you can specify the same string for more than one positional /IDENTIFIER parameter qualifier if the command contains multiple command-type parameters.

Specify UICs in the standard OpenVMS UIC format, which allows alphanumeric group and member values. Wildcards are allowed in both the group and member fields. Specify OpenVMS identifiers in the standard format, which allows 1 to 31 alphanumeric characters and contains at least one nonnumeric character. Refer to the OpenVMS documentation for System Managers and the OpenVMS documentation for System Services for more information.

Example

```
$ DBO/GRANT_COMMAND/ADD/IDENTIFIER=([DBMS,HARRY],[SPEC,DEBI]) PARTS USER
```

This command adds entries for two users to each of the USER CALs.

The command-type parameter USER refers to the CALs for DBO/ANALYZE and DBO/VERIFY. Because /ENTRY is not specified, each new entry is added as entry 1. The UICs are processed left to right; therefore, an entry one for [DBMS,HARRY] is inserted into the CALs for DBO/ANALYZE and DBO/VERIFY. It will then become displaced to entry 2 when similar entries are added for [SPEC,DEBI].

See the *Oracle CODASYL DBMS Database Security Guide* for additional examples.

9.16.2 DBO/GRANT_COMMAND/DELETE Command

Deletes entries from the CALs of a database according to the parameters and qualifiers you specify.

Format

DBO/GRANT_COMMAND/DELETE root-file-spec command-type [,...]

Command Qualifiers	Defaults
/[NO]CONFIRM	See description
/[NO]LOG	Current DCL verify value

Parameter Qualifiers	Defaults
/ENTRY=integer	/ENTRY=1
/IDENTIFIER=option	

Description

This command deletes one or more entries from one or more CALs. You can specify entries for deletion in the following ways:

- Use the /ENTRY qualifier to specify the position of the entry in the CAL.
- Use the /IDENTIFIER qualifier to specify one or more UICs or OpenVMS identifiers; entries containing those UICs or OpenVMS identifiers are deleted.
- Use neither /ENTRY nor /IDENTIFIER qualifiers and delete all entries in the CALs named in the command-type parameters.

Unless you request otherwise (by specifying the /NOCONFIRM qualifier), the command displays a confirmation inquiry for each entry you specify before actually deleting it.

Parameters

root-file-spec
Specifies the root file of the database whose CALs you want to reference. The default file type is .ROO.

command-type [,...]
Specifies one or more DBO commands that refer to one or more CALs from which you want to delete entries. See Table 9–4 for a list of valid command types.

Command Qualifiers

/CONFIRM
/NOCONFIRM
Specify /CONFIRM to prompt for each entry you specify, thus reducing the risk of an unintended deletion. This is the default for interactive processing. Specify /NOCONFIRM to perform all deletions the command requests without preliminary inquiry. This is the default for batch processing.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT.

Specify /LOG to request that DBO/GRANT_COMMAND report each CAL deletion, and specify /NOLOG to prevent this list. If you specify neither qualifier, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

Parameter Qualifiers

/ENTRY=integer
Specifies an ordinal position in the CAL where you intend to delete an ACE. (You can first enter a DBO/GRANT_COMMAND/LIST command to learn the order of the ACEs.) If you use /ENTRY, you get an error message if you try to use /IDENTIFIER in the same command line. The default is /ENTRY=1.

/IDENTIFIER=UIC,...
/IDENTIFIER=VMS_ID,...
Identifies one or more users by their UIC or OpenVMS identifier. If you specify a UIC value for which no ACE exists in the CAL, an informational message results. When deleting ACEs, the /IDENTIFIER qualifier is optional.

If a UIC or OpenVMS identifier string contains duplicate values, an informational message results and DBO ignores the duplicates. Nevertheless, you can specify the same string for more than one positional /IDENTIFIER parameter qualifier, if the command contains multiple command-type parameters. If you use /IDENTIFIER, you get an error message if you try to use /ENTRY in the same command line.

Specify UICs in the standard OpenVMS UIC format, which allows alphanumeric group and member values. Wildcards are allowed in both the group and member fields. Specify OpenVMS identifiers in the standard format, which allows 1 to 31 alphanumeric characters and contains at least one nonnumeric character. Refer to the OpenVMS documentation for System Managers and the OpenVMS documentation for System Services for more information.

Examples

1.
```
$ DBO/GRANT_COMMAND/DELETE PARTS USER
```

   This command issues a delete confirmation message for each entry in the DBO/ANALYZE and DBO/VERIFY CALs.

   The command-type parameter USER refers to two commands: DBO/ANALYZE and DBO/VERIFY. You can see each entry and decide whether to keep it or delete it. The /CONFIRM qualifier is specified by default on /DELETE operations.

2.
```
$ DBO/GRANT/DELETE PARTS ALTER/IDENTIFIER=(SAM), -
_$ PERMIT/IDENTIFIER=(SAM,JANET)
delete DBO/ALTER entry for [DBMS,SAM]? Y
delete DBO/PERMIT entry for [DBMS,SAM]? Y
delete DBO/PERMIT entry for [DBMS,JANET]? Y
```

   This command shows the use of positional /IDENTIFIER qualifiers in multiple command-type parameters. Entries for the user [SAM] are deleted from the CALs for ALTER and PERMIT following the delete confirmation inquiry, as is the entry for [JANET] in the CAL for PERMIT.

3.
```
$ DBO/GRANT/DELETE/IDENTIFIER=(SAM) PARTS ALTER,PERMIT/IDENTIFIER=(JANET)
delete DBO/ALTER entry for [DBMS,SAM]? Y
delete DBO/PERMIT entry for [DBMS,JANET]? Y
```

   Example 3 differs from Example 2 in that the positional identifier qualifier overrides the global.

   See the *Oracle CODASYL DBMS Database Security Guide* for additional examples.

9.16.3 DBO/GRANT_COMMAND/LIST Command

Lists all entries in all or specified CALs.

Format

DBO/GRANT_COMMAND/LIST root-file-spec [command-type [,...]]

Command Qualifier	Default
/OUTPUT=file-spec	SYS$OUTPUT

Description

This command lists all ACEs currently in one or more CALs. You can specify a set of CALs by means of command-type parameters. The default command type is a wildcard, which is the same as specifying the MANAGER command type. This lists the CALs of all securable DBO commands.

Parameters

root-file-spec
Specifies the root file of the database for which the ACEs are to be listed. The default file type is .ROO. This parameter is required.

command-type [,...]
Specifies one or more DBO commands whose ACEs will be listed. See Table 9–4 for a list of valid command types.

Command Qualifier

/OUTPUT=file-spec
Specifies the file to which output is written. If you omit this qualifier, selected ASCII records are written to SYS$OUTPUT. The default file type is .LIS.

Example

```
$ DBO/GRANT_COMMAND/LIST PARTS
%DBO-I-LOGCALACE, DBO/ALTER (entry 1) is granted to [SNOW,USER]
%DBO-I-LOGCALACE, DBO/ANALYZE (entry 1) is granted to [SNOW,USER]
%DBO/I-LOGCALACE, DBO/BACKUP (entry 1) is granted to [SNOW,USER]
   .
   .
   .
```

This command requests a list of the users granted permission to use the securable DBO commands. Only UIC [SNOW,USER] has permission to use the commands listed.

9.17 DBO/INITIALIZE Command

Reinitializes database storage area files for a database.

Format

DBO/INITIALIZE root-file-spec [area-name[,...]]

Command Qualifiers	Defaults
/[NO]CONFIRM	See description
/[NO]LOG	Current DCL verify value
/TSN	

Command or Area Qualifiers	Defaults
/ALLOCATION=integer	
/BLOCKS_PER_PAGE=integer	
/INTERVAL=integer	
/SNAPSHOTS=allocation=integer	
/[NO]SPACE_MANAGEMENT	/SPACE_MANAGEMENT
/THRESHOLDS=(x1[,x2[,x3]])	

Description

Useful during application development, the DBO/INITIALIZE command empties all storage area files or only the specified areas associated with the database. All data previously stored in those areas is lost. If a storage area has been expanded since it was created, the CALC range is reset to the area's current size.

You can change certain area characteristics by initializing one or more areas. These characteristics are the interval, page size (blocks_per_page), area allocation, and snapshot file allocation.

Note

Before initializing an area, check its DBO/DUMP output to be sure that it does not contain records that point to other areas. Initializing such an area causes you to lose the pointers permanently. This corrupts the database.

Parameters

root-file-spec
Specifies the root file of a database whose storage areas are to be initialized. The default file type is .ROO.

area-name
Specifies one or more areas to be initialized. The area should not require recovery and should not contain cross-area pointers. If you do not specify any areas, DBO initializes the entire database. The default file type for areas is .DBS.

Command Qualifiers

/CONFIRM
/NOCONFIRM
Specify /CONFIRM to have DBO prompt you for confirmation before initializing each specified database or area. This is the default for interactive processing.

Specify /NOCONFIRM to initialize each database you specify without receiving a confirmation prompt. This is the default for batch processing.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to display the initialization process and /NOLOG to prevent this list. If you specify neither, the default is the setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.) When you initialize area files, use /LOG to receive a message supplying the name of the database whose areas are being initialized.

/TSN
Initializes the transaction sequence number (TSN) value. Each time a transaction is initiated against a database, DBCS issues it a TSN. The numbers are incremented sequentially over the life of the database. Immediately after resetting the TSN, you should perform a full database backup. Should your database become corrupt you will be able to roll forward any .AIJ files.

The maximum TSN value is 4,294,967,295. When your TSN is coming close to the maximum, DBO/BACKUP displays a warning message during your regular database backups telling you that you must reset the TSN. You can also see how close you are to the maximum by dumping the database using the DEBUG option:

```
$ DBO/DUMP/OPTION=DEBUG/OUTPUT=PARTS.LIS PARTS
```

Search the output file for NEXT_TSN to see the TSN that will be used for the next transaction. When you decide to reset the TSN, perform a full backup of the database. Do this to protect yourself in case of any failure during the TSN initialization. Then, once the TSN initialization is complete, you must perform another full backup before accessing the database. The reason for this post-TSN

initialization backup is because subsequent .AIJ files will be based on the new TSNs. The .AIJ backup files must be using the same TSNs. You cannot roll forward a database from an old (pre-TSN initialization) backup file with new (post-TSN initialization) .AIJ files. When you initialize the TSN, the .AIJ files can no longer be used.

When initializing TSNs, the operation is very slow. To increase the performance of the operation, you should increase the size of the buffers and reduce the number of buffers for this operation only. The optimal configuration would be one very large buffer (up to 128 blocks).

Command or Area Qualifiers

/ALLOCATION=integer
Defines the number of data pages in an area file. This page count does not include space area management (SPAM) pages. The /ALLOCATION qualifier can be used globally or locally. Local specifications override global specifications.

/BLOCKS_PER_PAGE=integer
Specifies the number of disk blocks in each page of the area file. The maximum size is 63 blocks.

/INTERVAL=integer
Specifies the number of data pages between SPAM pages in the physical storage file and thus the maximum number of data pages each SPAM page will manage. The minimum interval is 256 data pages. The maximum interval depends on page size and should not exceed the following calculation:

```
Maximum interval = ((blocks_per_page * 512) - 22) * 4
```

Unless you specify /NOSPACE_MANAGEMENT, the first page of each database area is a SPAM page. The interval integer determines where subsequent SPAM pages will be inserted.

/SNAPSHOTS=allocation=integer
Allows snapshots for a storage area. DBO reserves space on each page in the storage area for snapshot information and creates a snapshot file for the storage area. You can temporarily enable or disable snapshots using the DBO/MODIFY/SNAPSHOTS=[NO]ENABLED command. Oracle recommends that you allow snapshots for all storage areas. Certain useful maintenance tools (such as online backup) require that snapshots be allowed and enabled. The ALLOCATION option specifies the number of pages in the snapshot file. DBO will extend or truncate the initialized file by the number of pages specified. Integer is the initial allocation. The default is 1. If you do not use ALLOCATION, the snapshot file remains the same size as the original setting.

/SPACE_MANAGEMENT
/NOSPACE_MANAGEMENT
Enables or suppresses the space area management utility. A database using space area management has a number of SPAM pages in each area in addition to data pages. The number of SPAM pages depends on the /INTERVAL qualifier for each area and does not count against the value specified in /ALLOCATION or /EXTENSION. Specify /NOSPACE_MANAGEMENT to suppress space area management. If you do not wish to change current SPAM settings, you do not have to specify thresholds or intervals. These values automatically default to the area's current value. This is a positional storage area qualifier. Oracle strongly recommends that you use the space area management page option. SPAM pages

significantly improve database update performance as an area fills up with data. The default is /SPACE_MANAGEMENT.

/THRESHOLDS=(x1[,x2[,x3]])
Specifies one, two, or three threshold values. The threshold values determine how much free space can be guaranteed to exist on a database page. Each threshold value represents a percentage of fullness on a data page. When a data page reaches the percentage of fullness defined by a given threshold value, the SPAM page entry for the data page is updated to contain that threshold value. If a record is large enough to occupy 50 percent of a database page, DBCS examines the threshold values to determine which pages have room for the new record. DBO ignores this qualifier if you specify /NOSPACE_MANAGEMENT. The *Oracle CODASYL DBMS Database Design Guide* contains detailed guidelines on selecting threshold values.

Examples

1.
```
$ DBO/INITIALIZE/LOG/ALLOCATION=100 -
_$ /BLOCKS_PER_PAGE=2/INTERVAL=256 -
_$ /SPACE_MANAGEMENT/THRESHOLDS=(70,85,95) PARTS BUY,MAKE
```

   This command reinitializes areas BUY and MAKE of the PARTS database. It sets allocation to 100, blocks-per-page to 2, and the interval between SPAM pages to 256 data pages. In addition, it enables space area management and sets thresholds to 70, 85, and 95. (The example returns areas BUY and MAKE to the default settings of the DBO/CREATE command.)

2.
```
$ DBO/INITIALIZE/LOG PARTS MAKE/ALLOCATION=150, -
_$ BUY/SNAPSHOTS=(allocation=2)
```

   This command also reinitializes areas MAKE and BUY of the PARTS database but sets different characteristics for the two areas. For MAKE, the command changes the allocation to 150. For BUY, the command changes the snapshot file allocation to 2.

9.18 DBO/INTEGRATE Command

Checks and verifies information in the database root file header against database instance information in Oracle CDD/Repository, updating Oracle CDD/Repository when necessary.

Format

DBO/INTEGRATE root-file-spec

Command Qualifiers	Defaults
/[NO]INSTANCE	/INSTANCE
/[NO]LOG	Current DCL verify value
/PATH=cdd-repository-path	See description

9.18 DBO/INTEGRATE Command

Description

The DBO/INTEGRATE command updates database instance information in Oracle CDD/Repository. It validates the schema, storage schema, all security schemas, and all subschemas in the root file against their respective compilations in Oracle CDD/Repository. If you do not specify an Oracle CDD/Repository pathname with the /PATH qualifier, DBO uses the schema path in the root file. DBO makes changes in Oracle CDD/Repository as needed to ensure that the root file header and Oracle CDD/Repository instance information reflect the composition of the database. In addition, the command compiles schemas into Oracle CDD/Repository that exist in the root file but not at the current Oracle CDD/Repository directory.

If the compilation exists, but the major/minor version numbers are not compatible, DBO cannot integrate the database with the schema in Oracle CDD/Repository. You must then attempt to integrate the database with another location in Oracle CDD/Repository.

The major version number is a timestamp showing when the schema was first compiled; the minor version number refers to the number of times the schema has been modified with the DDL/MODIFY command. If Oracle CDD/Repository compilation exists, if the major/minor version numbers (defined by the DDL compiler) are compatible, and if instance information does not already exist in Oracle CDD/Repository, DBO inserts the instance information for the database into Oracle CDD/Repository. You can use the DBO/DUMP/HEADER command to display the major and minor version numbers.

In the full-development version of Oracle CODASYL DBMS, if you specify an Oracle CDD/Repository path to a schema that does not exist in Oracle CDD/Repository, the DBO/INTEGRATE command compiles compatible versions of the schema, storage schema, security schemas, and subschemas into Oracle CDD/Repository as necessary to make the database valid. This operation includes insertion of database instance information.

Because this command opens many files, it might exceed your open file quota. If so, have the system manager increase your open file quota.

Parameter

root-file-spec
Specifies the root file of the database to integrate with Oracle CDD/Repository. The default file type is .ROO. Wildcards are allowed.

Command Qualifiers

/INSTANCE
/NOINSTANCE
Specifies whether or not to create a database instance entity in the repository as part of the integration procedure. The default is /INSTANCE.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to list all actions performed on Oracle CDD/Repository by the integration procedure and /NOLOG to prevent this list. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/PATH=cdd-repository-path
Specifies an Oracle CDD/Repository location containing the schema with which the database will be integrated. If you do not specify /PATH, DBO uses the schema path in the root file.

The Oracle CDD/Repository path named in the root file header is the default. If you specify the /PATH parameter, the DBO/INTEGRATE command integrates the database in the specified Oracle CDD/Repository directory and changes the path in the root file.

This is a positional qualifier.

Example

```
$ DBO/INTEGRATE PARTS
```

This command integrates the schema, security schema, storage schema, and subschemas with their compilations in Oracle CDD/Repository.

9.19 DBO/LOAD Commands

There are two forms of the DBO/LOAD command, each having its own qualifiers and parameters. These are:

DBO/LOAD

This format loads or reloads a database from OpenVMS RMS files.

DBO/LOAD/CONTINUE

This format recovers and continues an interrupted load operation.

These formats are described in the following sections.

9.19.1 DBO/LOAD Command

Performs an initial load of a database from OpenVMS RMS files or performs a reload of a database from OpenVMS RMS files produced by a previous DBO/UNLOAD command.

Format

DBO/LOAD root-file-spec

Command Qualifiers	Defaults
/ANALYZE	
/FILE=file-spec	/FILE=SYS$DISK:[]
/FORMAT=lfl-file-spec	/FORMAT=root-file-spec.LFL
/[NO]LOG	Current DCL verify value
/[NO]QUIET_POINT=integer	/QUIET_POINT=1000
/READY=(allow-mode,access-mode)	/READY=(E,U)
/RETRIEVAL_READY =(allow-mode,access-mode)	/RETRIEVAL_READY=(P,R)
/[NO]SAVE=save-file-spec	/SAVE=root-file-spec.LSV
/[NO]SELECTIVE	/NOSELECTIVE

/SEQUENCE=lsl-file-spec	/SEQUENCE=root-file-spec.LSL
/SUBSCHEMA=subschema-name	/SUBSCHEMA=DEFAULT_SUBSCHEMA

Description

The LOAD utility allows you to introduce OpenVMS RMS file data into an existing database. In addition, you can use it as an aid to database restructuring.

Before loading data from OpenVMS RMS files, you need the following:

- A database root file with a schema, subschema, and storage schema in Oracle CDD/Repository. The subschema must include all records that you intend to load.
- One or more OpenVMS RMS files containing the data you intend to introduce. Each file must contain only one type of record and be sorted in the order it will be processed.
- A load format language (.LFL) file that identifies the input files containing the records and defines the set relationships of the records.

  The .LFL file must include record description information to define the format of the OpenVMS RMS input records.
- A load sequence file in load sequence language (.LSL) that describes the load order of input records.

A security schema is optional. The DBO/LOAD command will support its use. The load operation can be restarted if necessary. DBO uses the save file, created by default during the load operation, to restart the load. See the DBO/LOAD/CONTINUE command for more information.

Chapter 10 shows the syntax of .LFL and .LSL files and explains briefly how they relate to the DBO/LOAD command. See the *Oracle CODASYL DBMS Database Load/Unload Guide* for detailed tutorial information on loading and unloading a database.

Parameter

root-file-spec
Specifies the root file of the database to be loaded. The DBO/LOAD command takes the schema, subschema, and storage schema from the root file. The default file type is .ROO.

Command Qualifiers

/ANALYZE
Generate an analysis that shows which realms will be readied in which mode. This is useful in predicting conflicting areas between parallel load or unload operations before you attempt to load or unload the database.

This qualifier is only valid when used with the /SELECTIVE qualifier.

/FILE=file-spec
Globally specifies the device and directory for input record data files. If you do not specify /FILE, DBO uses the default device and directory.

/FORMAT=lfl-file-spec
Specifies the name of a file whose contents are in load format language. The default file specification is:

root-file-spec.LFL

Identifies the record and input source OpenVMS RMS file for that record. It also defines the record format and the set linkages. The format of this file is described in Chapter 10.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request reporting of DBO/LOAD activity and /NOLOG to prevent this reporting. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/QUIET_POINT=integer
/NOQUIET_POINT
Specifies how many stores can occur between quiet points (that is, the number of DML STORE operations performed between COMMIT RETAINING statements). The default is 1000. The /NOQUIET_POINT qualifier commits transactions just once at the end of the load operation.

/READY=(allow-mode,access-mode)
Sets the area lock for the areas you are loading. Valid allow-modes are EXCLUSIVE and BATCH. The only valid access-mode is UPDATE. The default is /READY=(EXCLUSIVE,UPDATE). If you use /READY=(BATCH,UPDATE), you cannot use the DBO/LOAD/CONTINUE restart capability.

/RETRIEVAL_READY=(allow-mode,access-mode)
Specifies the lock mode for the areas that DBO/LOAD accesses only for retrieval (such as in walking a set hierarchy.) The valid allow-modes are EXCLUSIVE, PROTECTED (default), and BATCH. The valid access-modes are RETRIEVAL (default) and UPDATE.

This qualifier is only valid when used with the /SELECTIVE qualifier.

/SAVE=save-file-spec
/NOSAVE
Identifies the save file in which DBO records the progress of the load operation. DBO uses this save file for restart in the event of an abnormal termination of the load operation. The default file specification is:

root-file-spec.LSV

See the description of the DBO/LOAD/CONTINUE command in the following section for details about the content and use of the save file. Specifying /NOSAVE prevents the generation of this file.

/SELECTIVE
/NOSELECTIVE
Readies only the required realms with the required mode. The /SELECTIVE qualifier also works with the /CONTINUE qualifier so you can change back and forth between readying all the realms and only those necessary when you continue the load.

If the /SELECTIVE qualifier is used with the /LOG qualifier, the load operation produces an informational message showing which areas are readied in which mode.

Use the /NOSELECTIVE qualifier to ready all realms. This is the default.

/SEQUENCE=lsl-file-spec
Specifies an .LSL file whose contents describe the load sequence criteria that determines the order of record loading. The default file specification is:

root-file-spec.LSL

The format of this file is described in Chapter 10.

/SUBSCHEMA=subschema-name
Specifies a subschema through which the utility loads the records. DEFAULT_SUBSCHEMA is the default. An error results if you specify a nonexistent subschema, either explicitly or by default.

Example

```
$ DEFINE F "SYS$COMMON:[SYSTEST.DBM]"
$ DBO/LOAD/SELECTIVE -
_$ /FORMAT=F:DBMPARTS/SEQUENCE=F:DBMPARTS/FILE=F: PARTS
```

These commands define a logical name and load data into the PARTS database. The format and sequence files are in SYS$COMMON:[SYSTEST.DBM].

9.19.2 DBO/LOAD/CONTINUE Command

Recovers and continues an interrupted load operation using a save file.

Format

DBO/LOAD/CONTINUE save-file-spec

Command Qualifiers	Defaults
/[NO]LOG	Current DCL verify value
/[NO]QUIET_POINT=integer	
/[NO]SELECTIVE	/NOSELECTIVE

Description

If a load operation is interrupted for any reason, such as a power failure, you can restart the load operation with a DBO/LOAD/CONTINUE command. Specify the save file as a command parameter. Your original DBO/LOAD command writes a save file to support restart unless you have specified the /NOSAVE qualifier. As the load operation progresses, the save file receives information about the load such as:

- The DBO/LOAD command line including the root file specification and all qualifiers and qualifier arguments
- The location of the current input record in each input file
- All database currencies

You can update save file information at intervals specified by the /QUIET_POINT qualifier (every 1000 records by default). The load operation waits while DBO updates the save file. After writing the save file, DBO commits the database with a COMMIT RETAINING statement and then continues the load operation. Transactions not committed are rolled back by the automatic recovery procedure prior to restart.

When you issue the DBO/LOAD/CONTINUE command, specify the save file as the parameter. The save file contains the original DBO/LOAD command line as well as currency information. Unless you want to change the settings of the original load operation, you need not use any qualifiers with the DBO/LOAD/CONTINUE command. If you want, you can change the specifications for quiet point and logging.

Parameter

save-file-spec
Specifies a save file produced by a previous DBO/LOAD operation. When you append the /CONTINUE qualifier to a DBO/LOAD command, DBO automatically treats the parameter as a save file specification. The default file type is .LSV.

Command Qualifiers

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to list the load operation and /NOLOG to prevent this listing. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/QUIET_POINT=integer
/NOQUIET_POINT
Specifies how many stores can occur between quiet points (that is, the number of DML STORE operations performed between COMMIT RETAINING statements). The value specified with DBO/LOAD/CONTINUE can supersede the value in the original DBO/LOAD command.

The /NOQUIET_POINT qualifier commits transactions just once at the end of the load operation.

/SELECTIVE
/NOSELECTIVE
Readies only the required realms with the required mode. You can change back and forth between readying all the realms and only those necessary when you continue the load.

If the /SELECTIVE qualifier is used with the /LOG qualifier, the load operation produces an informational message showing which areas are readied in which mode.

Use the /NOSELECTIVE qualifier to ready all realms. This is the default.

Example

```
$ DBO/LOAD/CONTINUE/SELECTIVE PARTS.LSV
```

This command continues an interrupted load for the PARTS database by using the save file produced by default during the initial load operation.

9.20 DBO/MODIFY Commands

There are two forms of the DBO/MODIFY command, each having its own qualifiers and parameters. These are:

DBO/MODIFY root-file-spec [area-name [,...]]

This format changes the attributes of a database.

DBO/MODIFY/RESTRUCTURE root-file-spec

This format invokes interactive DRU.

These formats are described in the following sections.

9.20.1 DBO/MODIFY Database Command

Changes the attributes of a database from those assigned by a previous DBO/CREATE or DBO/MODIFY command.

Format

DBO/MODIFY root-file-spec [area-name [,...]]

Command Qualifiers	Defaults
/[NO]ADJUSTABLE_LOCKING =(n1[,n2...[,n8]])	
/[NO]AFTER_JOURNAL=file-spec	See description
/AIJ_OPTIONS=option[,...]	See description
/BACKUP_OPTIONS =[NO]SCAN_OPTIMIZATION	=SCAN_OPTIMIZATION
/BATCH_WRITE=(options,[...])	See description
/BUFFERS=integer	
/[NO]CDD_INTEGRATE	
/CLOSE=option	
/CLUSTER_NODES=integer	
/DBR_BUFFERS=integer	
/[NO]DEFERRED_SNAPSHOTS	
/DIRECTORY=directory-spec	
/FAST_COMMIT=(option[,...])	
/GLOBAL_BUFFERS=(option[,...])	
/[NO]HOLD_RETRIEVAL_LOCKS	
/IMPORT_FILE=file-spec	See description
/JOURNAL_OPTIONS=option[,...]	See description
/LENGTH_BUFFER=integer	
/[NO]LOCK_OPTIONS =(option[,...])	See description
/[NO]LOG	Current DCL verify value
/OPEN=option	
/OPTIONS=file-spec	
/PREFETCH =([NO]ENABLED,DEPTH=integer)	See description
/[NO]PLT	/NOPLT
/[NO]RECOVER_JOURNAL=file-spec	
/RESERVE=(option[,...])	See description
/RUJ_OPTIONS=(option[,...])	See description
/SCHEMA	

/[NO]SECURITY_SCHEMAS [=security-schema-name[,...]]	
/SERVER=AFTER_JOURNAL =MANUAL|AUTOMATIC	=MANUAL
/[NO]STATISTICS	
/STORAGE_SCHEMA	
/[NO]SUBSCHEMAS [=subschema-name[,...]]	
/TIMEOUT=LOCK=integer	
/TRANSACTION=(option[,...])	See description
/USERS=integer	
/[NO]WAIT_RECORD_LOCKS	

Command or Area Qualifiers	**Defaults**
/[NO]ALLOCATION=integer	
/BLOCKS_PER_PAGE=integer	
/CACHE =([NO]ENABLED, [NAME=cache-name])	=NOENABLED
/[NO]CHECKSUM_PAGES	/CHECKSUM_PAGES
/EXPANSION=option[,...]	
/[NO]EXTENSION=integer	
/FILE=file-spec	
/INTERVAL=integer	
/[NO]READ_ONLY	
/READY=(allow-mode,access-mode)	
/[NO]RECALC	
/SNAPSHOTS=(option[,...])	
/[NO]SPACE_MANAGEMENT	
/THRESHOLDS=(pct1[,pct2[,pct3]])	

Description

When you modify a database, use the qualifiers of the DBO/MODIFY command to:

- Incorporate a modified schema, subschema, security schema, or storage schema in the database
- Add new areas to the database
- Add and delete database subschemas
- Add and delete database security schemas
- Change the number of users who can access the database at one time
- Change locking characteristics for transactions against the database
- Change access mode for the database
- Specify file specifications for new database area files
- Specify page sizes for database area files being added to the database
- Change size or number of buffers
- Start or stop after-image journaling
- Reserve slots in the database root file for .AIJ files, row caches, and reload areas
- Change global after-image journaling characteristics for the database
- Change specified characteristics for individual .AIJ files

- Change the default placement of the recovery-unit journal (.RUJ) file
- Integrate Oracle CDD/Repository information in the root file header with database instance information in Oracle CDD/Repository
- Change storage allocation or extension values or both for one or more storage area files
- Change the snapshot options for one or more areas
- Specify deferred snapshots
- Recalculate the CALC chains in an extended area
- Ready area locks
- Specify space area management information for new database areas, including definitions of page fullness percentage thresholds and space area management (SPAM) page intervals for each new area
- Change space area management threshold values for new or existing areas
- Change the number of allowable CPU nodes concurrent in a VMScluster environment
- Enable fast commit
- Change characteristics of asynchronous batch write operations
- Enable or disable asynchronous prefetching of database pages
- Enable or disable calculations of page checksums as pages are read from or written to storage area or snapshot files
- Link a record-level cache to a specified area or all areas of the database
- Enable or disable record-level caching
- Enable or disable fast incremental backup operations

Unless otherwise stated, defaults for the following qualifiers are the values set by the previous DBO/CREATE or DBO/MODIFY command.

Note

Most DBO/MODIFY operations are not written to the .AIJ file. To be sure subsequent database updates are fully recoverable, you should perform a full database backup immediately after every major modification done with DBO/MODIFY. This will cause the current .AIJ files to become obsolete.

The major modifications are modifying schema, including adding areas; modifying the number of users; allocating an area; and changing from no expansion to any expansion setting. Failure to back up the database after these changes might result in the inability to fully recover your database from backup and .AIJ files should it become corrupt.

Parameters

root-file-spec
Specifies the root file of the database to be modified. The default file type is .ROO.

area-name [,...]
Specifies one or more storage areas of the database for which you are setting or changing storage area attributes, or that you are reserving for a reload operation.

Command Qualifiers

/ADJUSTABLE_LOCKING=(n1[,n2...[,n8]])
/NOADJUSTABLE_LOCKING
Specifies the intermediate page-range locking levels for the adjustable granularity record-locking feature in Oracle CODASYL DBMS. It minimizes the number of OpenVMS locks needed during the execution of your program by allowing you to specify intermediate page-range locking rather than locking at the area and record levels.

By default, Oracle CODASYL DBMS uses three intermediate locking levels of 10, 100, and 1000 pages. Oracle recommends changing the locking levels from the DBO/CREATE default settings only when performance problems related to locking arise.

If you specify /NOADJUSTABLE_LOCKING, Oracle CODASYL DBMS performs area-level and record-level locking only.

/AFTER_JOURNAL=file-spec
/NOAFTER_JOURNAL
Specifies a valid device and directory name for an after-image journal file. The default file type is .AIJ. Do not specify a version number on this file specification.

If you want multiple .AIJ files, this qualifier creates a new .AIJ file when used with the /AIJ_OPTIONS=CREATE qualifier. You must provide the device, directory, and file name. Any previously created .AIJ files remain in the database. It *does not* enable after-image journaling.

Note

In order to be compatible with previous versions of Oracle CODASYL DBMS, the /AFTER_JOURNAL=file-spec qualifier still provides the same functions as before. That is, you can still use this qualifier to create one .AIJ file and enable after-image journaling for the database if you want to use only one .AIJ file. However, using the qualifier this way limits you to pre-Version 6.0 journaling functions. Oracle recommends that you use the extended Version 6.0 syntax to create .AIJ files and enable journaling, regardless of how many .AIJ files you use. See the qualifier descriptions for /JOURNAL_OPTIONS=ENABLED and /AIJ_OPTIONS=CREATE for more information.

The command in the following example creates a new after-image journal file in the PARTS database, and gives it the user-specified name AIJ_1. The file specification is DISK1:[USER1]AIJ_1.AIJ.

```
$ DBO/MODIFY/AIJ_OPTIONS=(CREATE,NAME=AIJ_1) -
_$ /AFTER_JOURNAL=DISK1:[USER1]AIJ_1.AIJ PARTS
```

The /NOAFTER_JOURNAL qualifier disables after-image journaling for the database only when you are using a single, extensible .AIJ file. You need this qualifier only if after-image journaling was previously enabled.

Note

In order to be compatible with previous versions of Oracle CODASYL DBMS, the /NOAFTER_JOURNAL qualifier still provides the same function as before. That is, if you have only one .AIJ file in your database, you can still use this qualifier to disable after-image journaling. However, Oracle recommends that you use the extended Version 6.0 syntax to disable after-image journaling, regardless of how many .AIJ files you have. See the qualifier description for /JOURNAL_OPTIONS=NOENABLED for more information.

/AIJ_OPTIONS=options [,...]

Performs operations on a specified .AIJ file. The following options are available:

- ALLOCATION=integer

  Specifies the initial number of 512-byte disk blocks for an .AIJ file. The minimum number of blocks allowed is 512; this is also the default.

- BACKUP=[backup-file-spec]

  Specifies a file name for the output file when the after-image journal is backed up. In the following example, the file name AIJ_BACKUP is specified as the backup file name for a journal named AIJ_1. When you back up the AIJ_1 journal, the backup file name will be AIJ_BACKUP.AIJ.

```
$ DBO/MODIFY/AIJ_OPTIONS=(MODIFY,NAME=AIJ_1, -
_$  BACKUP=AIJ_BACKUP.AIJ)
```

  If you use the BACKUP option without a backup file specification, the backup file name will be the current version of the original .AIJ file name. Because you are creating a new version of the .AIJ file rather than actually performing a backup operation, you can save overhead. You can then perform an OpenVMS backup on the older version of the .AIJ file, then delete that version. If you do this, however, be sure you have adequate disk space to hold both versions of the .AIJ file.

  In the next example, assuming PARTS.AIJ is version 1 of an .AIJ file, the backup file name will be PARTS.AIJ;1 and the new .AIJ file will be PARTS.AIJ;2.

```
$ DBO/MODIFY/AIJ_OPTIONS=(NAME=PARTS.AIJ,BACKUP)
```

- [NO]BACKUP

  Removes a previous backup file specification.

- CREATE

  Creates a new .AIJ file. You must also use the /AFTER_JOURNAL=file-spec qualifier to specify the file location. It is recommended that you also use the NAME option to create a pointer to that file. This will simplify and shorten your command lines. For example, the following command creates a new journal with the file specification DISK1:[USER1]AIJ1.AIJ. The name AIJ_1 points to that file.

```
$ DBO/MODIFY/AIJ_OPTIONS=(CREATE,NAME=AIJ_1) -
_$  /AFTER_JOURNAL=DISK1:[USER1]AIJ1.AIJ
```

- DELETE

  Deletes an .AIJ file name and the file it points to. You must also use the NAME option to indicate the file you want to delete. For example, the following command deletes the AIJ_1 file name and its associated file:

  ```
  $ DBO/MODIFY/AIJ_OPTIONS=(DELETE,NAME=AIJ_1)
  ```

- EXTENSION=integer

  Specifies the number of 512-byte disk blocks by which to extend an .AIJ file when it is full. This option is meaningful only when you are using a single, extensible after-image journal file. It is ignored if you are using multiple journal files. However, if you set the option and then go back to using a single, extensible journal file, the journal file will have the previously specified extension value. The minimum number of blocks allowed is 512; this is also the default.

- MODIFY

  Modifies a characteristic for an .AIJ file. The characteristics you can modify are values for the following /AIJ_OPTIONS options: ALLOCATE, BACKUP=backup-filename-spec, EXTENSION, SUPPRESS, and UNSUPPRESS. You must use the NAME option with the MODIFY option to indicate the file you want to modify. For example, the following command changes the number of blocks to allocate to a file named AIJ_1:

  ```
  $ DBO/MODIFY/AIJ_OPTIONS=(MODIFY,NAME=AIJ_1,ALLOCATION=512)
  ```

- NAME=name

  Specifies a name that describes an .AIJ file. This lets you use a short name on a command line instead of typing the entire file specification. You must use this option to identify the file you want to affect when using the CREATE, MODIFY, or DELETE options with the /AIJ_OPTIONS qualifier. The following command creates an .AIJ file with the file specification DISK1:[USER1]AIJ1.AIJ and specifies the name AIJ_1 for that file:

  ```
  $ DBO/MODIFY/AIJ_OPTIONS=(CREATE,NAME=AIJ_1) -
  _$  /AFTER_JOURNAL=DISK1:[USER1]AIJ1.AIJ
  ```

  The command in the following example modifies the number of disk blocks allocated to the AIJ_1 journal:

  ```
  $ DBO/MODIFY/AIJ_OPTIONS=(MODIFY,NAME=AIJ_1,ALLOCATION=1024)
  ```

- [UN]SUPPRESSED=[NO]ONLINE

  Specifies that an .AIJ file be made inaccessible. The UNSUPPRESSED option makes a previously suppressed file accessible again. Both options use the keywords ONLINE or NOONLINE to indicate whether you want to perform the operation on line or off line.

  NOONLINE is the default for both options. You must use this option if you are suppressing the last available .AIJ file. You cannot suppress the last available .AIJ file on line. The following command makes available the previously suppressed file named AIJ_1, and specifies that this be done on line:

  ```
  $ DBO/MODIFY/AIJ_OPTIONS=(MODIFY,NAME=AIJ_1,UNSUPPRESSED=ONLINE)
  ```

The next example shows the command to suppress a previously unsuppressed .AIJ file named JOURNAL1. Because this is the last available .AIJ file in the database, the NOONLINE option is used:

```
$ DBO/MODIFY/AIJ_OPTIONS=(MODIFY,NAME=JOURNAL1,SUPPRESSED=NOONLINE)
```

You cannot unsuppress a previously suppressed journal file that has had hard data loss.

/BACKUP_OPTIONS=SCAN_OPTIMIZATION
/BACKUP_OPTIONS=NOSCAN_OPTIMIZATION

Allows you to enable or disable fast incremental backup operations. When you specify the /BACKUP_OPTIONS=SCAN_OPTIMIZATION qualifier, DBO records the identity of pages that have not been updated since the last full backup operation. This feature, called fast incremental backup, means that during an incremental backup operation, DBO need not read every page of every area to determine if the page has been updated since the last full backup operation.

However, there is a cost in recording this information in the database. In some circumstances the cost might be too high, particularly if you do not intend to use incremental backup operations. If you want to disable the recording of pages changed since the last full backup operation, specify the /BACKUP_OPTIONS=NOSCAN_OPTIMIZATION qualifier.

You can also enable or disable the database setting for fast incremental backup with the DBO/MODIFY/BACKUP_OPTIONS=[NO]SCAN_OPTIMIZATION and DBO/CREATE/BACKUP_OPTIONS=[NO]SCAN_OPTIMIZATION commands. See Section 9.11 and Section 9.5.3 for details.

The default is the /BACKUP_OPTIONS=SCAN_OPTIMIZATION qualifier.

/BATCH_WRITE=(options,[...])

Allows you to specify options that affect asynchronous batch write operations. These options are:

- ENABLED and NOENABLED

  Use the ENABLED option to enable, or the NOENABLED option to disable, asynchronous batch write operations. If you do not specify this option, asychronous batch write operations are enabled by default.

- CLEAN_BUFFER_COUNT=integer

  Use the CLEAN_BUFFER_COUNT option and specify an integer value to specify the number of clean buffers to be maintained at the end of a process' least recently used (LRU) queue of buffers for replacement. In the asynchronous batch write process, a set of clean (or empty) buffers must be available at all times at the end of the LRU queue, so that as buffers are filled and written, there are others that can replace them.

  The default CLEAN_BUFFER_COUNT value is 5.

- MAXIMUM_SIZE=integer

  Use the MAXIMUM_SIZE option and specify an integer value to specify the maximum number of buffers used for asynchronous batch write operations.

  The default MAXIMUM_SIZE value is determined as follows:

  1. Let x be the larger of the following values:
     - The number of database buffers divided by 5
     - The value 2

2. The default value is the smaller of the following values:
   - The value x
   - The value 10

See the *Oracle CODASYL DBMS Programming Reference Manual* for more information on asynchronous batch write operations.

/BUFFERS=integer
Specifies the number of database page buffers for each run unit. If your database is using global buffering, this value is overridden by the maximum number of global buffers per run-unit set with the /GLOBAL_BUFFERS=MAXIMUM_PER_USER qualifier.

/CDD_INTEGRATE
/NOCDD_INTEGRATE
Specifies whether or not database instance information in Oracle CDD/Repository includes your modifications to the database. If you specify /CDD_INTEGRATE, which is the default, DBO incorporates your modifications into the database instance information in Oracle CDD/Repository. If you specify /NOCDD_ INTEGRATE, DBO does not incorporate your modifications into the instance information in Oracle CDD/Repository.

/CLOSE=option
Specifies the mode in which the database is to be closed.

- MANUAL

  The database must be closed manually by issuing the DBO/CLOSE command.

- AUTOMATIC

  The database will be automatically closed when there are no users accessing the database. This mode is applicable only when DBO/MODIFY/OPEN is set to AUTOMATIC.

- TIMED_AUTOMATIC=integer

  The database will be closed automatically after there has been no activity for the specified minutes. The default is 0 minutes.

/CLUSTER_NODES=integer
Specifies the maximum number of CPU nodes in a VMScluster configuration that can access a database simultaneously. The maximum number supported is 96 nodes.

/DBR_BUFFERS=integer
Specifies the number of buffers to be used by the database recovery process (DBR) when it clears uncommitted changes in the database by using the .RUJ file. The larger the number of DBR buffers, the faster the recovery process. Setting the number of buffers high and a user's working set extent low can lead to a high page fault rate. If this occurs, increase the working set extent quota for the user. The user name shown in the monitor log for the DBR process is the process name of the person who started the monitor.

The more memory on the system, the more buffers it can handle. The default number of DBR buffers is 10. If your database is using global buffering, this value is overridden by the maximum number of global buffers per run-unit set with the /GLOBAL_BUFFERS=MAXIMUM_PER_USER qualifier. The length of a DBR buffer is set by the /LENGTH_BUFFER qualifier.

/DEFERRED_SNAPSHOTS
/NODEFERRED_SNAPSHOTS
Specify /DEFERRED_SNAPSHOTS to cause update transactions to write records to snapshot files only if a snapshot transaction (BATCH RETRIEVAL) is in progress or waiting to start. When you are using deferred snapshots, BATCH RETRIEVAL transactions wait for all current update transactions to complete before starting. (BATCH RETRIEVAL transactions wait until all current update transactions COMMIT or ROLLBACK.) After a BATCH RETRIEVAL transaction is attempted, all subsequent update transactions write before-images of the records they modify to the snapshot files. After the BATCH RETRIEVAL transaction completes, future update transactions do not write to the snapshot file.

You must specify deferred snapshots for the entire database. However, you can allow or enable snapshots for individual areas.

Specifying /NODEFERRED_SNAPSHOTS causes an update transaction to always write to the area snapshot file, even when no snapshot transaction is in progress or waiting. Using /DEFERRED_SNAPSHOTS saves overhead incurred by writing to the snapshot file only when a snapshot transaction or BATCH RETRIEVAL job is in progress or waiting.

/DIRECTORY=directory-spec
Specifies a global device and directory for new areas being added to the database. The /DIRECTORY qualifier is useful to shorten commands when the /FILE and /SNAPSHOTS qualifiers all specify the same device and directory. The global file specification becomes the default specification for all areas being added.

For example, the following two commands are equivalent:

```
DBO/MODIFY/FILE=DISK:[USER]/SNAP=DISK:[USER] PARTS
DBO/MODIFY/DIRECTORY=DISK:[USER] PARTS
```

You can override the /DIRECTORY file specification with a global or local /FILE qualifier.

/FAST_COMMIT=(options[,...])
Specifies the commit scheme for fast commit. The following options are available:

- [NO]ENABLED

  Specifies whether or not fast commit is to be used.

- BLOCKS_PER_CHECKPOINT=integer

  Specifies an AIJ block checkpoint interval value. The valid range of integers is 0 to 2,147,483,647. If you specify 0 blocks, a checkpoint is not triggered. The default value is 512 blocks.

- COMMIT_TO_JOURNAL=(options[,...])

  Specifies the commit scheme for optimizing fast commit. The following options are available:

  - [NO]ENABLED

    Specifies whether or not fast commit optimization is to be used.

  - TRANSACTION_INTERVAL=integer

    Sets the size of a block of transaction sequence numbers (TSNs). The default size is 256. The minimum size is 8 and the maximum size is 512.

- TIME_PER_CHECKPOINT=n

  Specifies a time checkpoint interval value, expressed in seconds. The valid range of integers is 0 to 2,147,483,647. If you specify 0 seconds, a checkpoint is not triggered. By default, no time interval value is specified. If you do not specify a value, no checkpoint is triggered by time.

/GLOBAL_BUFFERS=(options[,...])

Specifies the buffering scheme for database pages. The following options are available:

- BUFFERS=integer

  Specifies the total number of global buffers for a database. Valid values are from 2 to 32768. The default value is equal to 5 times the maximum number of users allowed to access the database simultaneously, as specified with the /USER qualifier.

- [NO]ENABLED

  Specifies whether or not global buffers are to be used.

- MAXIMUM_PER_USER=integer

  Specifies the maximum number of global buffers a run unit can allocate. The default is 5.

- PAGE_TRANSFER={DISK | MEMORY}

  Specify PAGE_TRANSFER=MEMORY to enable optimized page transfer. Specify PAGE_TRANSFER=DISK to disable it.

  When the memory page transfers feature is enabled, a process does not need to write a modified page to disk before another process accesses the page. In other words, pages in a process' allocate set can contain committed updates from another process that have not been written to disk.

  If a database has the required characteristics for using memory page transfers, the performance gain of update-intensive applications can be significant. This is because of the number of input/output (I/O) operations saved by not writing updates to disk, and the ability to share pages among processes.

  The memory page transfers feature is available for any database with the following characteristics:

  - Global buffers are enabled.
  - After-image journaling is enabled.
  - Fast-commit processing is enabled.
  - The database can be accessed only from a single node (the /CLUSTER_NODES value for the database is 1).

/HOLD_RETRIEVAL_LOCKS
/NOHOLD_RETRIEVAL_LOCKS

Specifies whether or not to hold retrieval locks until the end of the transaction. Specify /HOLD_RETRIEVAL_LOCKS to hold all retrieval locks for each transaction until the end of the transaction.

Specify /NOHOLD_RETRIEVAL_LOCKS to release record retrieval locks for records that are not in a keeplist or serving as a currency indicator.

/IMPORT_FILE=file-spec

Incorporates the specified metadata into the target database. The target storage schema name defaults to the storage schema name found in the metadata file. To specify another storage schema name use the /STORAGE_SCHEMA qualifier. When the /IMPORT_FILE qualifier is specified, the Oracle CDD/Repository dictionary is not integrated by default.

See the DBO/EXPORT command for information on copying metadata from an existing database.

/JOURNAL_OPTIONS=options

Specifies after-image journaling characteristics for all .AIJ files in the database when you are using multiple .AIJ files. You can override this qualifier for a specified .AIJ file with the /AIJ_OPTIONS qualifier. The database-wide options you can specify are:

- ALLOCATION=integer

  Specifies the initial number of 512-byte disk blocks for .AIJ files. The minimum number of blocks allowed is 512; this is also the default.

- [NO]BACKUP[=backup-file-spec]

  Specifies the default name for the output file when .AIJ files are backed up. In the following example, the name PARTS_BACKUP.AIJ is specified as the output file for all .AIJ backups.

  ```
  $ DBO/MODIFY/JOURNAL_OPTIONS=(MODIFY,BACKUP=PARTS_BACKUP.AIJ)
  ```

  The NOBACKUP option removes a previous backup file specification.

  If you use the BACKUP option without a backup file specification, the backup file name will be the current version of the original .AIJ file name. Because you are creating a new version of an .AIJ file rather than actually performing a backup operation, you can save overhead. You can then perform an OpenVMS backup on the older versions of the .AIJ files, then delete those versions. If you do this, however, be sure you have adequate disk space to create the various versions of .AIJ files.

- [NO]ENABLED

  Enables after-image journaling for the database. You must have at least one accessible journal file to enable journaling. The NOENABLED option disables journaling for the database, but does not delete .AIJ files. The following command enables after-image journaling for the PARTS database:

  ```
  $ DBO/MODIFY/JOURNAL_OPTIONS=ENABLED PARTS
  ```

- EXTENSION=integer

  Specifies the number of disk blocks to be stored in the AIJFB in case a fixed-size journal file is converted to an extensible journal file by the removal of another journal file. The default and minimum number that can be stored is 512 blocks.

- [NO]NOTIFY=(operator-name)

  Designates any of 16 OPCOM operators to receive after-image journaling information. If a catastrophic event occurs, for example if the AIJ switch-over operation cannot complete, or if the .AIJ file cannot be extended, the system operator will receive event notification messages from the database. The operator names you can specify are:

  – CENTRAL

- DISK
- CLUSTER
- SECURITY
- OPER1, OPER2, OPER3, . . . OPER12

For example, to have OPER2 receive .AIJ event notification messages for the PARTS database, enter the following command:

```
$ DBO/MODIFY/JOURNAL_OPTIONS=(NOTIFY=OPER2) PARTS
```

For more information on using OPCOM, see the OpenVMS documentation on system management.

- [NO]OVERWRITE

  Specifies whether or not the AIJ switch-over operation overwrites an active journal file when no unmodified journal files exist. The default option is NOOVERWRITE.

  This qualifier works only when you have two or more journal files. It is ignored if specified on a database with a single extensible .AIJ file.

  After-image journal files are used for database recovery in case of media failure and for transaction recovery as part of the fast commit feature. Using the OVERWRITE option implies that hard data loss has occurred, and means the database will not be recoverable. You should only use this option when database availability or performance requirements are more important that database recoverability.

  For example, in some environments only the fast commit feature is of interest and a small set of journal files can be used as a circular fast commit log with no backup of the contents required. The OVERWRITE option instructs Oracle CODASYL DBMS to write over journal records that would normally be used for media recovery. The resulting set of journal files is unable to be used by DBO/RECOVER for media recovery.

  You can see whether or not a journal file is being written over with the DBO/SHOW STATISTICS AIJ Information screen, as shown in the following example:

```
Node: NODEA    Oracle CODASYL DBMS V7.0-00                 11-NOV-1996 14:51:39
Rate: 3.00 Seconds              AIJ Information            Elapsed: 00:01:01.20
Page: 1 of 1          DISK$1:[USER1.WORK]PARTS.ROO;1         Mode: Online
------------------------------------------------------------------------------
Journaling: Enabled   Shutdown:   30  Notify: Enabled   State: Accessible
ALS: Manual     ABS: Enabled   ACE: Disabled  FC: Disabled  CTJ: Disabled
```

```
After-Image.Journal.Name....... SeqNum AIJsize CurrEOF Status. State.......
ONE                                  2 *BACKUP DENIED* Written Overwritten
TWO                                  3     513       2 Current Overwritten
Available AIJ slot 1
Available AIJ slot 2
Available AIJ slot 3
Available AIJ slot 4
Available AIJ slot 5
Available AIJ slot 6
Available AIJ slot 7
Available AIJ slot 8
Available AIJ slot 9
Available AIJ slot 10
Available AIJ slot 11
Available AIJ slot 12
--------------------------------------------------------------------------------
```

- [NO]ROTATE

  Forces a switch over to the next available .AIJ file. Use this option if you want to switch quickly to another .AIJ file because of disk errors. The NOROTATE option causes no rotation or switchover to occur.

  The following command forces a switchover to the next available journal file in the PARTS database:

  ```
  $ DBO/MODIFY/JOURNAL_OPTIONS=ROTATE PARTS
  ```

  If a switch over to the next AIJ file cannot complete because the next AIJ file is unavailable the database enters the "AIJ suspended" state. If the DBM$BIND_ALS_CREATE_AIJ system logical is either not defined or defined as 1 (true) then a new permanent "emergency" AIJ file will automatically be created for the switch over to terminate the AIJ suspended state.

- [NO]SHUTDOWN=integer

  Specifies the time, in minutes, until database shutdown occurs in the event of an AIJ problem.

- [NO]SPOOLER

  Enables the monitor-invoked AIJ Backup Server (ABS) for the database. The ABS automatically backs up a single .AIJ file to disk, then unbinds and terminates. You must also use the BACKUP=file-spec option to set a file specification for the backup file.

  The default NOSPOOLER option specifies that the ABS not be started by the monitor. You can dump the header file and check the journaling section to see if the ABS has been enabled.

/LENGTH_BUFFER=integer

Defines the number of disk blocks in a database page buffer.

/LOCK_OPTIONS=option[,...]

Enables either adjustable record locking and adjustable page locking, the default locking mechanism for Oracle CODASYL DBMS, or two-phase page locking, which was a new feature for Version 6.0. It also enables carry-over lock optimization, which operates independently of record and page locking.

This qualifier replaces /LOCK_OPTIMIZATION.

The /LOCK_OPTIONS qualifier takes the following options:

- RECORD_LEVEL

Specifically enables adjustable record locking and implicitly enables adjustable page locking.

Adjustable record locking and adjustable page locking are the default locking mechanisms for Oracle CODASYL DBMS. They are enabled automatically with the DBO/CREATE command, unless you specify the PAGE_LEVEL option. To disable them, you must specify the /LOCK_OPTIONS=PAGE_LEVEL qualifier with the DBO/MODIFY command.

- PAGE_LEVEL

  Enables two-phase page locking. Two-phase page locking allows transactions to hold page locks per storage area and not request record locks. For more information see the *Oracle CODASYL DBMS Database Design Guide*.

  By default, two-phase page locking is disabled. If you enable it and then want to disable it, you must enable adjustable record and page locking by specifying the RECORD_LEVEL option with the /LOCK_OPTIONS qualifier.

- [NO]CARRYOVER

  Enables or disables carry-over lock optimization. This option replaces the /LOCK_OPTIMIZATION qualifier. Carry-over locks are enabled by default. To disable carry-over lock optimization, use the NOCARRYOVER option with the DBO/MODIFY command.

  This is an online database modification.

  Carry-over lock optimization operates independently of record and page locking.

For more information on adjustable record locking, adjustable page locking, two-phase page locking, and carry-over lock optimization, see the *Oracle CODASYL DBMS Database Design Guide*.

/LOG
/NOLOG

Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to display the name of each attribute as it is modified and /NOLOG to prevent this display. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/OPEN=option

Specifies the mode in which the database can be opened for access. If you specify the AUTOMATIC option, the first bind statement issued causes the database to open automatically. If you specify the MANUAL option, you must use the DBO/OPEN command to open the database and the DBO/CLOSE command to close the database.

The default is to leave the mode unchanged.

/OPTIONS=file-spec

Specifies a file whose text defines areas and their qualifiers. This feature makes it easier to define database characteristics without exceeding the character and token limits for DCL-level commands.

The format of the text in the file is:

{area-name [qualifier]...}...

Commas are optional for separating area names, but hyphens are required to continue command lines. For example:

```
PERSONNEL       /SNAPSHOT=(ALLOCATION=111) -    ! Correct
                /THRESHOLDS=(71,81,91)

PERSONNEL       /SNAPSHOT=(ALLOCATION=111)      ! Incorrect
                /THRESHOLDS=(71,81,91)


PERSONNEL       /SNAPSHOT=(ALLOCATION=111)      ! Correct
MAKE            /SNAPSHOT=(ALLOCATION=121)

PERSONNEL       /SNAPSHOT=(ALLOCATION=111),     ! Also correct
MAKE            /SNAPSHOT=(ALLOCATION=121)

PERSONNEL       /SNAPSHOT=(ALLOCATION=111) -    ! Correct, only if
MAKE            /SNAPSHOT=(ALLOCATION=121)      ! total number of characters
                                                ! does not exceed 256.
```

Because the area qualifiers are positionally dependent, the location of the /OPTIONS qualifier is also important.

There is no limit on the number of area-names; however, each area entry in the options file cannot exceed 256 characters and 128 tokens. Use of /OPTIONS supersedes the need for DBO indirect command files. The default file type is .DBO.

Command line qualifiers override corresponding qualifiers in the options file.

/PREFETCH=([NO]ENABLED,DEPTH=integer)

Allows you to specify options to affect asynchronous prefetch operations. These options are:

- ENABLED or NOENABLED

  Use the ENABLED option to enable, and the NOENABLED option to disable, asynchronous prefetching of database pages. Asynchronous prefetching is enabled by default.

- DEPTH=integer

  Use the DEPTH option to set the number of buffers into which Oracle CODASYL DBMS can prefetch database pages. The default is the number of database buffers divided by 4, or the value 8, whichever is smaller.

See the *Oracle CODASYL DBMS Programming Reference Manual* for more information on asynchronous prefetching.

/PLT
/NOPLT

Lock partitioning allows you to specify whether more than one lock tree is used for the database or whether all lock trees for a database are mastered by one database resource tree.

When partitioned lock trees (PLT) are enabled for a database, locks for storage areas are separated from the database resource tree and all locks for each storage area are independently mastered on the VMScluster node that has the highest traffic for that resource. OpenVMS determines the node that is using each resource the most and moves the resource hierarchy to that node.

You should not enable lock partitioning for single-node systems, because all lock requests are local on single-node systems.

By default, lock partitioning is disabled.

/RECOVER_JOURNAL=file-spec
/NORECOVER_JOURNAL

Specifies the default device and directory of the .RUJ files. Generally, you only need to specify a device and directory for this qualifier; DBO assigns the root file type .RUJ to the .RUJ file.

In a VMScluster environment, you can place all .RUJ files in a single, central directory ensuring the continued availability of all .RUJ files. OpenVMS has a utility to limit how many versions of a file can be in one directory. Setting a limit on the directory with .RUJ files can cause database corruption.

If you use /NORECOVER_JOURNAL, the default .RUJ file name is reset, as if /RECOVER_JOURNAL had not been specified in the DBO/CREATE command.

Never destroy .RUJ files when the database is in an inconsistent state. If the .RUJ is destroyed when the database is inconsistent, or if the .RUJ is missing or corrupt, you will have to use DBO/RESTORE and DBO/RECOVER to return the database to a usable condition.

/RESERVE=(option[,...])

Allows you to reserve slots for after-image journal (.AIJ) files, record-level caches, or to reserve a reload area, using the following options:

- AFTER_JOURNAL=integer

  Reserves slots in the database root file for a predetermined number of after-image journal files. Exclusive database access is required. You must perform a full database backup after this operation.

  Because this is not an online operation, if you plan to use multiple .AIJ files, it is important to determine ahead of time how many slots you will need. When you create or convert a database, Oracle CODASYL DBMS provides one slot by default. It is recommended that you reserve more slots than you need. You cannot decrease the number of slots set aside once they are reserved.

- CACHE=integer

  Reserves slots in the database root file for a predetermined number of pointers to row caches. Exclusive database access is required. You must perform a full database backup after this operation.

  Because this is not an online operation, if you plan to use multiple row caches, it is important to determine ahead of time how many slots you will need. If you reserve a sufficient number of slots for row caches, you can add row caches while the database is on line without interrupting database activity.

  If you do not reserve any slots for row caches, Oracle CODASYL DBMS reserves one slot when the database is created, restored or converted. You cannot decrease the number of reserved slots for row caching after you have reserved them.

- RELOAD_AREA

  Reserves a reload area in preparation for an online reload operation. A reserved reload area is an area that cannot be referenced by schema WITHIN clauses or subschema REALM clauses. Currently, it is used only by an online reload operation. Reserving a reload area is an offline operation that is not journaled. After a reload area is reserved, perform a full database backup operation to ensure that the database is recoverable.

The /RESERVE=RELOAD_AREA qualifier cannot be used with any other metadata-related qualifiers, such as /SCHEMA, /STORAGE_SCHEMA, /SUBSCHEMAS, or /SECURITY_SCHEMAS, because each of them modifies the metadata in the database.

/RUJ_OPTIONS=(BUFFER={LOCAL|GLOBAL})
This qualifier permits the assignment of recovery-unit journal (RUJ) buffers created for each process, (typically allocated to local virtual memory), to be assigned to a shared global section. This means the recovery process can occur in global memory and possibly avoid disk access.

This buffer memory can be defined as GLOBAL to improve row cache performance for recovery. If row caching is disabled, then buffer memory is always LOCAL.

/SCHEMA
Replaces the present schema, security schema, and storage schema with a modified schema, security schema, and storage schema. (By specifying /SCHEMA, you imply /STORAGE_SCHEMA as well.) To successfully issue a DBO/MODIFY command, you must previously have modified the corresponding schema, security schema, and storage schema in Oracle CDD/Repository with DDL/MODIFY.

/SECURITY_SCHEMAS [=(security-schema-name [,...])]
/NOSECURITY_SCHEMAS
Specifies the security schemas to add to or delete from the root file. You can use /SECURITY_SCHEMAS and /NOSECURITY_SCHEMAS entries in the same command. Form variants are:

- /SECURITY_SCHEMAS [=(security-schema-name [,...])]

  Adds one or more security schemas, replacing existing security schemas of the same name, if applicable. DBO retrieves security schemas from the Oracle CDD/Repository node of the schema on which the root file is based. (The root file header information includes a pointer to this Oracle CDD/Repository node.)

- /SECURITY_SCHEMAS

  Adds the DEFAULT_SECURITY_SCHEMA to the root file.

- /SECURITY_SCHEMAS=*

  Adds to the root file all security schemas currently stored under the schema.

- /NOSECURITY_SCHEMAS [=security_schema_name [,...]]

  Deletes security schemas from the root file. You can name one or more security schemas for deletion. When you delete a security schema from a database, you automatically delete all user execution list entries for that security schema.

- /NOSECURITY_SCHEMAS

  Deletes the DEFAULT_SECURITY_SCHEMA from the root file.

- /NOSECURITY_SCHEMAS=*

  Deletes from the root file all security schemas currently stored under the schema.

/SERVER=AFTER_JOURNAL=AUTOMATIC
/SERVER=AFTER_JOURNAL=MANUAL
Enables the AIJ log server (ALS) and determines whether the ALS is started automatically or manually.

The /SERVER=AFTER_JOURNAL=AUTOMATIC qualifier enables the ALS and specifies that it be automatically started by the monitor when the database is opened.

The /SERVER=AFTER_JOURNAL=MANUAL qualifier enables the ALS and specifies that it be started and stopped manually, on line.

If you set ALS to the manual online mode, use the DBO/SERVER AFTER_ JOURNAL command to stop or start the ALS.

/STATISTICS
/NOSTATISTICS
Determines whether or not database statistics are enabled. If the /STATISTICS qualifier is not specified on the command line, nothing is modified.

/STORAGE_SCHEMA
Replaces the present storage schema with a modified storage schema. You must have modified the corresponding storage schema in Oracle CDD/Repository with DDL/MODIFY.

/SUBSCHEMAS [=(subschema-name [,...])]
/NOSUBSCHEMAS
Specifies the subschemas to add to or delete from the root file. You can use both /SUBSCHEMAS and /NOSUBSCHEMAS qualifiers in the same command.

Form variants are:

- /SUBSCHEMAS [=(subschema-name [,...])]

  Adds one or more subschemas to the database. By default, DBO retrieves subschemas from the Oracle CDD/Repository node of the schema on which the root file is based. (The root file header information includes a pointer to this Oracle CDD/Repository node.) If you specify the /IMPORT_FILE qualifier, then DBO retrieves subschemas from the specified metadata (.DBM) file. (See Section 9.14 for information on metadata files.)

  You cannot update an existing subschema by using the /SUBSCHEMAS qualifier with the DBO/MODIFY command. If you want to update an existing subschema without changing the name, you must first delete the subschema using DBO/MODIFY/NOSUBSCHEMAS=subschema-name and then add it back using DBO/MODIFY/SUBSCHEMAS=subschema-name.

- /SUBSCHEMAS

  Adds the DEFAULT_SUBSCHEMA to the root file.

- /SUBSCHEMAS=*

  Adds to the root file all subschemas currently stored under the schema.

- /NOSUBSCHEMAS [=(subschema-name, [,...])]

  Deletes one or more subschemas from the root file.

- /NOSUBSCHEMAS

  Deletes the DEFAULT_SUBSCHEMA from the root file.

- /NOSUBSCHEMAS=*

  Deletes from the root file all subschemas currently stored under the schema.

/TIMEOUT=LOCK=integer
Specifies the length of time (in seconds) the database should wait before returning a lock timeout message. You can use this qualifier in conjunction with the DBM$BIND_LOCK_TIMEOUT_INTERVAL logical name and with the WAIT option on the READY statement to set the lock timeout duration.

/TRANSACTION=(options...)
This qualifier allows the database administrator to adjust the setting for prestarted transactions.

- PRESTART=ENABLED
  PRESTART=NOENABLED

  The default when a database is created is to have prestarted transactions enabled. This is a performance feature which allows Oracle CODASYL DBMS to automatically establish a new transaction during the regular commit processing. When the application starts a transaction the I/O has already been perform and so reduced I/O to the root file.

  Use NOENABLED to disable this feature.

- PRESTART=TIMEOUT:n

  The database administrator may define a timeout for a prestarted transaction.

  ```
  $ DBO/MODIFY/TRANSACTION=PRESTART=(TIMEOUT=n) dbroot
  ```

  here 'n' is a value in seconds (range 0:3600). This value represents the number of seconds to wait before aborting the prestarted transaction. Timing out the prestarted transaction may prevent snapshot file growth in environments where servers stay bound to the database with long periods of inactivity.

  Additionally, a process will be forced to obtain a new transaction sequence number (TSN) if the same TSN has been reused throughout the duration of the prestarted transaction timeout interval. This permits processes that constantly reuse TSNs to periodically obtain a new TSN, thus preventing excessive snapshot growth.

/USERS=integer
Defines the number of users who can access the database simultaneously. Each DBO command counts as a user. The maximum number of users is 16368. Setting this value unnecessarily high causes performance to degrade. The default is 10.

The minimum value is five (5) to allow for various optional database servers, such as the ABS (Automatic Backup Server), RCS (Row Cache Server) or ALS (AIJ Log Server) to access the database.

/WAIT_RECORD_LOCKS
/NOWAIT_RECORD_LOCKS
Specifies whether to wait on record lock conflicts or treat lock conflicts as errors.

Specify /WAIT_RECORD_LOCKS to wait until the lock is granted or a deadlock condition is returned.

Specify /NOWAIT_RECORD_LOCKS to return an error message whenever a record lock conflict occurs.

Command or Area Qualifiers

/ALLOCATION=integer
/NOALLOCATION
Specifies a new size for the database file containing the storage area. The integer defines the number of data pages. The new size must not be less than the present size, including extensions to the file that might have occurred since the database was created or last modified. (You can use DBO/MODIFY to enlarge a file by increasing the number of pages, but you cannot use it to enlarge the pages.)

The /NOALLOCATION qualifier when used positionally on a specific area negates the global /ALLOCATION value.

You can use /ALLOCATION for a new storage area to specify the initial number of data pages in the storage area. The default allocation for a new area is 100.

/CACHE=(option[,...])
Specifies how record-level caching is to be employed. /CACHE takes the following options:

- CHECKPOINT=(option[,...]))

  The CHECKPOINT keyword for the /CACHE qualifier specifies the source records and target for checkpoint operations for the row cache. If the keyword ALL_ROWS is specified, then all rows in the cache are written during the checkpoint process.

  If the keyword UPDATED_ROWS is specified, then just the added or the updated rows are written during the checkpoint process. The keywords ALL_ROWS and UPDATED_ROWS are mutually exclusive.

  If the target of the checkpoint operation is BACKING_FILE, then the RCS process writes the row cache entries to the backing (.DBC) files. If the target is DATABASE, then the added or updated rows (only UPDATED_ROWS is allowed) are written back to the database. Upon recovery from a node failure, the database recovery process has no data on the contents of the row cache. Therefore, it does not repopulate the row caches in memory. The keywords BACKING_FILE and DATABASE are mutually exclusive.

  The keyword TIMED specifies the frequency (in seconds) with which the Row Cache Server (RCS) process checkpoints the contents of the row caches back to disk.

  The frequency of RCS checkpointing is important in determining how much of an .AIJ file must be read during a recovery operation following a node failure. It also affects the frequency with which marked records get flushed back to the database for those row caches that checkpoint to the database. The default is every 15 minutes (900 seconds).

  This CHECKPOINT keyword overrides the database-level CHECKPOINT clause.

- DIRECTORY=directory-spec

  The DIRECTORY keyword specifies the name of the directory to which row cache backing file information is written. The database system generates a file name (row-cache-name.DBC) automatically for each row cache at checkpoint time. Specify a device name and directory name only. By default, the location is the directory of the database root file. The backing files are temporary. They are deleted when the row cache server (RCS) is shut down,

unless the system logical name, DBM$BIND_RCS_KEEP_BACKING_FILES, is defined.

The NODIRECTORY keyword removes any defined location for this cache.

- [NO]ENABLED

  The ENABLED or NOENABLED keywords specifies whether or not the row caching feature is enabled. Enabling row caching does not affect database operations until a cache is created and assigned to one or more storage areas.

  When the row caching is disabled, all previously created and assigned caches remain and will be available if row caching is enabled again.

  To enable or disable row caching, you must have exclusive access to the database (it is an offline operation).

- NAME=cache-name

  – Specify the /CACHE=(ENABLED, NAME=cache-name) qualifier to specify that caching be enabled for the named cache.

  – Specify the /CACHE=(NOENABLED, NAME=cache-name) qualifier to specify that caching be disabled for the named cache.

  If the /CACHE qualifier appears after a storage area name, the named cache is linked to that storage area. If the qualifier appears before any storage area name (or if a storage area name is not specified), the named cache is linked to the entire database.

/BLOCKS_PER_PAGE=integer

Specifies the number of disk blocks in each page of a new area file that you are adding to the database. All pages in any one storage area have the same size. The maximum size is 63 blocks. The size that you choose can only be modified by backing up and restoring the database.

The default is 2 blocks (1024 bytes).

For an existing storage area, DBO ignores this qualifier.

/CHECKSUM_PAGES
/NOCHECKSUM_PAGES

Determines whether or not page checksums are calculated when pages are read from or written to storage areas. Use the /CHECKSUM_PAGES qualifier to enable checksum calculations and use the /NOCHECKSUM_PAGES qualifier to disable checksum calculations. You can disable and subsequently enable checksum calculations again without error. However, once checksum calculations have been disabled, corrupt pages may not be detected even if checksum calculations are subsequently enabled again.

The /CHECKSUM_PAGES qualifier is the default.

/EXPANSION=option[,...]

Defines how many data pages, if any, DBO will add automatically to a storage area file when its present allocation is full. Automatic extension adds only overflow space to a file; it does not extend the file's CALC range. DBO inserts SPAM pages as needed while observing the specified /INTERVAL value.

The /EXPANSION qualifier takes the following options:

- ENABLED and NOENABLED turn area extension on and off. If you specify NOENABLED, DBO reports an exception to your application program when the storage area becomes full but does not extend the area.

- MINIMUM specifies the minimum number of pages by which to extend the area when it is full.
- MAXIMUM specifies the maximum number of pages by which to extend the area when full. You can extend the area to an unlimited number of pages.
- PERCENT specifies a percentage of the current area size by which to extend the area when it is full.
- SPREAD or NOSPREAD is meaningful only if the area resides in a multiple volume disk set. If you specify SPREAD and the area resides in a multiple volume disk set, the extension will be spread across the multiple volumes. DBO reports a device-full error when all volumes in the disk set are full.

DBO determines the size of the expansion by the MINIMUM, MAXIMUM, and PERCENT values and the current size of the area. The MINIMUM and MAXIMUM values define a range of allowable extension sizes. The utility first calculates the percentage of the current area size and then determines if that value is within the MINIMUM and MAXIMUM value range. The extension size is either the MINIMUM number of pages, the MAXIMUM number, or the size produced by calculating the percentage of the current area size, provided the percentage falls between the specified MINIMUM and MAXIMUM values. For example, you might specify the following qualifier for an area:

```
/EXPANSION=(MIN=100,MAX=500,PERCENT=25)
```

If the area currently contains 300 pages, the percentage calculation yields a proposed extension value of 75. This is less than the minimum extension value of 100. DBO therefore extends the file by 100 pages.

If the area currently contains 500 pages, the percentage calculation yields a proposed extension value of 125. This is greater than the minimum extension value of 100 and less than the maximum of 500. DBO therefore extends the file by 125 pages.

If the area contains 2000 or more pages, DBO will extend it by the defined MAXIMUM of 500 pages. For a larger extension, you must specify a higher MAXIMUM value.

/EXTENSION=integer
/NOEXTENSION

Defines a constant number of data pages to add automatically to a storage area file when full. Automatic extension adds only overflow space to a file; it does not extend the file's CALC range. DBO inserts SPAM pages as needed while observing the specified /INTERVAL value.

The /EXPANSION qualifier replaces the /[NO]EXTENSION qualifier. The /[NO]EXTENSION qualifier remains solely for compatibility with previous versions of Oracle CODASYL DBMS. If you include both the /EXPANSION and /EXTENSION qualifier in a command, the /EXPANSION qualifier will override the /EXTENSION qualifier.

/FILE=file-spec

Identifies the database file name for a new storage area. The default file type is .DBS. You can use /FILE to specify any part of the full file specification for a storage area file. For an existing storage area, DBO ignores this qualifier.

If you do not specify a database file, DBO creates one in the current default directory with the following file specification:

STORAGE_A.DBS

Where STORAGE_A is the first 9 characters of the storage area name.

You can place database storage areas on private disks.

Global use of /FILE is a convenient way to specify a default device and directory for the database files, shortening the command. The global file specification becomes the default specification for all storage area files, even for areas not specified. Any part of a complete file specification omitted from a local /FILE qualifier on an area parameter defaults to the file specification given in the global /FILE qualifier.

For example, assume that you want to modify the PARTS database on disk DISK3: under directory [USER.DBMS]. If your default directory is DISK1:[USER], you can enter:

```
$ DBO/MODIFY/FILE=DISK3:[USER.DBMS] PARTS MAKE/FILE=PTMAKE
```

DBO stores all files of the database in DISK3:[USER.DBMS] and the area MAKE in the file DISK3:[USER.DBMS]PTMAKE.DBS. DBO also stores all other storage area files in DISK3:[USER.DBMS], using the area file specification defaults.

/INTERVAL=integer
Defines the number of data pages between SPAM pages in the physical storage file of a new area and, thus, the maximum number of data pages each SPAM page will manage. The default, also the minimum interval, is 256 data pages.

Unless you specify /NOSPACE_MANAGEMENT, the first page of each database area is a SPAM page. The integer determines where DBO inserts subsequent SPAM pages provided there are enough pages in the area to require more SPAM pages.

This qualifier works only on a new area that you are adding to the database.

/READ_ONLY
/NOREAD_ONLY
Defines an area to be read-only. Transactions can ready read-only areas in any retrieval mode. Transactions cannot ready read-only areas in update mode. This is useful for distributing copies of the database and to let users read the database while doing exclusive operations like DBO/BACKUP.

/READY=(allow-mode,access-mode)
Sets the area lock for an area for which you have specified the /RECALC qualifier. Valid allow-modes are EXCLUSIVE, the default, and BATCH. The only valid access-mode is UPDATE, which is also the default. If you use /READY=BATCH, DBO does not perform recovery-unit journaling and marks the storage area as corrupt if the RECALC operation terminates. If you do not use /RECALC, DBO ignores this qualifier. The default is /READY=(EXCLUSIVE,UPDATE).

/RECALC
/NORECALC
Calls for expansion of the area's CALC range to the present allocation. The /RECALC qualifier affects only the CALC chains; it does not cause any physical reclustering of CALC set records.

The /RECALC qualifier requires EXCLUSIVE UPDATE or BATCH UPDATE. You can use the /READY qualifier to set the allow- and access-modes.

The /NORECALC qualifier turns off or negates the effect of /RECALC when used positionally.

/SNAPSHOTS=(option[,...])
Modifies snapshot options for a database storage area that had snapshot files allowed when you created the database.

For a new area, you can use /SNAPSHOTS or /NOSNAPSHOTS to specify the snapshot options for the new area. (See the DBO/CREATE/SNAPSHOTS command.)

Specifying /NOSNAPSHOTS disallows snapshots for the storage area specified, leaving more space in the storage area for storing data. (You can later allow and enable snapshots for this storage area by restoring the database using the /SNAPSHOTS=(ALLOWED) option on the DBO/RESTORE command.)

Specify /SNAPSHOTS=option[,...] to allow snapshots and simultaneously enable them for a database or new storage area file. Batch retrieval transactions require the use of this qualifier. In addition, online analysis, online backup, and online verify require that snapshots be allowed and enabled.

The following options are available:

- ALLOCATION=integer

  Changes the initial number of database pages in the snapshot file. (The pages in the snapshot file are the same size as the pages in the corresponding storage area.) For a new area, the default is one page.

  You can reclaim snapshot file space with DBO/MODIFY by specifying the /SNAPSHOTS qualifier with an allocation smaller than the current snapshot file size. Snapshot pages will be deleted from the beginning of the file. This is useful when a snapshot file has been extended a number of times and contains a large number of old snapshot pages that are no longer useful. For example, if you have a very large snapshot file for the area BUY in the PARTS database, you can shorten it to include only the last 200 pages that were allocated to it as follows:

  ```
  $ DBO/MODIFY PARTS MAKE/SNAPSHOTS=(ALLOCATION=200)
  ```

- [NO]CHECKSUM_PAGES

  Specifies whether to enable or disable calculations of page checksums when pages are read from or written to snapshot files. Specify the CHECKSUM_PAGES option to enable checksum calculations and specify the NOCHECKSUM_PAGES option to disable checksum calculations for snapshot files.

  You can disable and subsequently enable checksum calculations again without error. However, once checksum calculations have been disabled, corrupt pages may not be detected even if checksum calculations are subsequently enabled again.

  The CHECKSUM_PAGES option is the default.

- [NO]ENABLED

  Specifies whether to enable or disable snapshot file activity, assuming that the database was created with snapshots allowed. A DBO/MODIFY command can enable or disable snapshot activity but cannot add snapshot capability (that is, allow snapshots) to a database. You can add snapshot capability by restoring your database from backup using the /SNAPSHOTS=(ALLOWED) option with the DBO/RESTORE command.

ENABLED, the default option for new areas, enables snapshot files immediately.

- EXPANSION=([NO]ENABLED,MINIMUM=integer, MAXIMUM=integer,PERCENT=integer,[NO]SPREAD)

  Changes the number of database pages that are added automatically to the snapshot file when its present allocation is full. This parameter does not affect the current size of the snapshot file.

  Arguments for this option include:

  – ENABLED and NOENABLED turn area extension on and off. ENABLED is the default for new areas.

  – MINIMUM specifies the minimum number of pages by which to extend the snapshot file when it is full. The default for new areas is 100.

  – MAXIMUM specifies the maximum number of pages by which to extend the snapshot file when it is full. By default, snapshot files of new areas can extend to an unlimited number of pages.

  – PERCENT specifies a percentage of the current snapshot file size by which to extend the snapshot file. If the number of pages calculated is less than MINIMUM or greater than MAXIMUM, DBO extends the snapshot file by the MINIMUM or MAXIMUM number of pages.

  – SPREAD or NOSPREAD is meaningful only if the snapshot file resides in a multiple-volume disk set. If you specify SPREAD, the extension will be spread evenly across the multiple volumes. NOSPREAD is the default for new areas.

  DBO determines the size of the extension by the MINIMUM, MAXIMUM, and PERCENT values and by the current size of the snapshot file. The MINIMUM and MAXIMUM values define a range of allowable extension sizes. The utility first calculates the percentage of the current area size and then determines if that value is within the MINIMUM and MAXIMUM value range. The extension size is either the MINIMUM number of pages, the MAXIMUM number, or that produced by calculating the percentage of the current area size, provided the percentage falls between the specified MINIMUM and MAXIMUM values.

- [NO]EXTENSION=integer

  Defines a constant number of database pages to add automatically to the snapshot file when it is full. By specifying EXTENSION=0 or NOEXTENSION, you prevent automatic extension.

  This option remains solely for compatibility with previous versions of Oracle CODASYL DBMS. EXPANSION values override EXTENSION values.

/SPACE_MANAGEMENT
/NOSPACE_MANAGEMENT

Enables or suppresses the SPAM utility for areas being added to the database. The default is /SPACE_MANAGEMENT. To suppress space area management, you must specify /NOSPACE_MANAGEMENT. You cannot enable or disable SPAM pages with a subsequent DBO/MODIFY command; your choice when you create the area remains in effect for the life of the database unless you unload and reload your database.

Oracle strongly recommends that you always use the space area management option. It significantly improves Oracle CODASYL DBMS database update performance as the data in the storage area increases. The SPAM pages take up little space compared to the performance benefits they provide, especially for update-intensive applications.

An area using space area management has a number of SPAM pages in addition to data pages. The number of SPAM pages depends on the /INTERVAL qualifier for each area and does not count against data page allocation or expansion values.

This qualifier only works for new areas being added to the database.

/THRESHOLDS=(pc1[,pc2[,pc3]])
Specifies one, two, or three threshold values for an existing area or for a new area being added to the database. The threshold values determine how much free space can be guaranteed to exist on a database page. Each threshold value represents a percentage of fullness on a data page. When a data page reaches the percentage of fullness defined by a given threshold value, the SPAM entry for the data page is updated to contain that threshold value. For example, if a record is large enough to occupy 50 percent of a database page, the Database Control System (DBCS) examines the threshold values to determine which pages have room for the new record.

If you omit /THRESHOLDS for an area, the defaults for a new area are 70, 85, and 95. If you specify only one or two values, unspecified values default to 100.

Examples

1.
```
$ DBO/MODIFY-
_$ /RESERVE=(AFTER_JOURNAL=10) -
_$ /JOURNAL_OPTIONS=(ENABLED,ALLOCATION=1024, -
_$ BACKUP=AIJ1_BACKUP.AIJ, -
_$ CACHE=CTS$DISK:[USER1.ESE50]ACE.AIJ,NOTIFY=OPER1,SPOOLER) -
_$ /AIJ_OPTIONS=(CREATE,NAME=AIJ_1) -
_$ /AFTER_JOURNAL=(AIJ$DISK:[USER1]AIJ_1.AIJ) -
_$ PARTS
```

The command in the previous example shows how to use the /RESERVE, /AIJ_OPTIONS, and /JOURNAL_OPTIONS qualifiers with the DBO/MODIFY command. Only qualifiers related to after-image journaling are shown. The command performs the following after-image journaling operations for the PARTS database:

- Reserves 10 slots in the database root file, in addition to the one slot automatically reserved by default when the database was created, for a total of 11 slots.
- Allocates 1024 blocks for any .AIJ files that might be created.
- Specifies a default backup file name of AIJ1_BACKUP.AIJ for the .AIJ file or files.
- Enables after-image journaling for the database.
- Designates the OPCOM operator name, OPER1, to receive after-image journaling information, and notify the operator if certain catastrophic events are occurring.
- Enables the monitor-invoked Automatic Backup Server (ABS) for the database.

- Creates one .AIJ file, using the AIJ settings established by the /JOURNAL_OPTIONS qualifier, with the file specification AIJ$DISK:[USER1]AIJ_1.AIJ and the user-specified name AIJ_1. This .AIJ file takes up one slot of the 11 slots reserved.

2.
```
$ DBO/MODIFY/SECURITY=(CLERK1,CLERK2) PARTS -
_$ BUY/ALLOCATE=4000/RECALC -
_$ /EXPANSION=(SPREAD,ENABLE,MIN=1000,MAX=2000,PERCENT=20)
```

   This command adds two security schemas to the database PARTS. For the area BUY, the command increases the allocation and expansion values. The SPREAD argument requires that the area BUY be on a multiple volume disk set. The /RECALC qualifier expands the CALC range to the new allocation using the default READY mode of EXCLUSIVE UPDATE.

3.
```
$ DBO/MODIFY/SCHEMA PARTS NEW/SNAPSHOTS=ENABLED
```

   This command adds a new area (NEW) to the PARTS database.

   This command enables snapshots for the area NEW, while accepting the remaining default area characteristics. Before issuing this command, you must first extract the schema source from the root file, edit it to add the new area, and enter a DDL/MODIFY command specifying the modified source file as the parameter. To access the new area, you must first add it to at least one subschema by editing the source, entering a DDL/COMPILE/REPLACE command and then entering another DBO/MODIFY command. See the *Oracle CODASYL DBMS Database Maintenance and Performance Guide* for an expanded example.

4.
```
$ DBO/MODIFY/PREFETCH=NOENABLED PARTS
```

   This command disables asynchronous prefetching of database pages for the PARTS database.

5.
```
$ DBO/MODIFY PARTS MAKE/NOCHECKSUM_PAGES/SNAPSHOTS=NOCHECKSUM_PAGES
```

   This command directs Oracle CODASYL DBMS to not calculate page checksums when pages are read from or written to an area or a snapshot file for that area.

6.
```
$ DBO/MODIFY/RESERVE=RELOAD_AREA PARTS
%DBO-W-DOFULLBCK, full database backup should be done to ensure
future recovery
```

   This command reserves a reload area in the PARTS database in preparation for an online reload operation.

7.
```
$ DBO/MODIFY PARTS MAKE/CACHE=(ENABLED, NAME=CACHE_1)
```

   This command links the CACHE_1 record cache to the MAKE area and enables record-level caching for the MAKE area.

8.
```
$ dbo/modify/transaction=prestart=(enabled,timeout=120) SAMPLE
%DBO-I-LOGMODROO, modifying root file USER2:[DB_TEST]SAMPLE.ROO;1
%DBO-I-LOGMODFLG,     enabled prestarted transactions
%DBO-I-LOGMODFLG,     modified prestarted transaction timeout to 120 seconds
```

   This command modifies the database to enable prestarted transactions and to set a timeout of 2 minutes (120 seconds). The timeout takes affect when there has been no activity from the application in that interval.

9.20.2 DBO/MODIFY/RESTRUCTURE Command

Invokes the interactive Database Restructuring Utility (DRU).

Format

DBO/MODIFY/RESTRUCTURE [root-file-spec] [area-name]

Command Qualifiers	Defaults
/READY=(allow-mode,access-mode)	/READY=(EXCLUSIVE,UPDATE)
/[NO]ONLINE	/NOONLINE

Command or Area Qualifiers	Defaults
/[NO]ALLOCATION=integer	See the DBO/MODIFY command
/BLOCKS_PER_PAGE=integer	See the DBO/MODIFY command
/FILE=file-spec	See the DBO/MODIFY command
/INTERVAL=integer	See the DBO/MODIFY command
/READY= (allow-mode,access-mode)	See the DBO/MODIFY command
/SNAPSHOTS=(option[,...])	See the DBO/MODIFY command
/[NO]SPACE_MANAGEMENT	See the DBO/MODIFY command
/THRESHOLDS= (pct1[,pct2[,pct3]])	See the DBO/MODIFY command

Description

The DBO/MODIFY/RESTRUCTURE command invokes DRU and allows you to:

- Reload areas on line or off line
- Remove areas from record WITHIN clauses
- Change set insertion and retention to AUTOMATIC MANDATORY or AUTOMATIC FIXED
- Specify that duplicates are not allowed on sorted sets
- Change the sort key and sort order on sorted sets
- Change set ORDER and MODE clauses
- Change B-tree node size, reset the B-tree fill factor, and, for SYSTEM-owned sets, change the B-tree page location
- Change item data type and length
- Remove data items from a database

When you enter DBO/MODIFY/RESTRUCTURE, DBO displays the following prompt:

```
DRU>
```

The DRU> prompt means you are at DRU command level. You cannot enter DCL commands until you exit DRU. Chapter 12 describes the DRU commands. See the *Oracle CODASYL DBMS Database Maintenance and Performance Guide* for a detailed tutorial to DRU.

Parameter

root-file-spec
Specifies the root file of the database containing the storage area you intend to restructure. The default file type is .ROO. You can omit this parameter and use the DRU BIND command to specify the database you want to restructure.

area-name
Specifies the name of an area you intend to reload. This parameter is useful only when you intend to reload an area with the DRU reload utility.

Command Qualifiers

/ONLINE
/NOONLINE
Specifies that the restructuring operation is to be performed while other users are bound to the database. Currently, the only restructuring operation that can be performed on line is a DRU reload operation.

The /NOONLINE qualifier specifies that the restructuring operation is to be performed without other users bound to the database. This is the default.

/READY=(allow-mode, access-mode)
Sets the allow-modes and the access-mode for the areas being modified. Valid allow-modes are EXCLUSIVE, which is the default and BATCH. The only valid access mode is UPDATE, which is also the default. If you use use /READY=BATCH, DRU does not perform recovery-unit journaling which makes change executions faster. However, if DRU change executions are terminated for any reason, then the affected storage areas are corrupt.

Examples

1.
```
$ DBO/MODIFY/RESTRUCTURE PARTS
%DBO-I-DBBOUND, bound to database "DISK:[USER]PARTS.ROO;1
DRU>
```

This command invokes DRU and binds to the PARTS database.

2.
```
$ DBO/MODIFY/RESTRUCTURE/ONLINE PARTS -
_$ MAKE/BLOCKS=3/ALLOCATION=200
%DBO-I-DBBOUND, bound to database "DISK:[USER]PARTS.ROO;1"
DRU>
```

This command invokes DRU and binds to the PARTS database. It directs DRU to reset the blocks and allocation for the MAKE area when the restructure operation is performed, and it specifies that the restructure operation will be an online operation. Currently, the only online operation permitted is an area reload operation.

9.21 DBO/MONITOR Command

Starts and stops the Oracle CODASYL DBMS monitor and can also create a new version of the monitor log file for the database monitor running on your node.

Format

DBO/MONITOR option

Command Qualifiers	Defaults
/[NO]ABORT=option	/NOABORT
/OUTPUT=file-spec	SYS$OUTPUT
/PRIORITY=integer	/PRIORITY=15
/[NO]SWAP	/NOSWAP
/[NO]WAIT	/NOWAIT

Description

The Oracle CODASYL DBMS monitor process is a detached process (usually running under the SYSTEM account) that controls database access, initiates the automatic recovery procedure, and maintains a log of database activity. A monitor process must be running on your system for you to use Oracle CODASYL DBMS. Usually, you will include the command to start the Oracle CODASYL DBMS monitor in your system startup command file and the command to stop the monitor in your system shutdown command file. You can also start and stop the monitor interactively using the DBO/MONITOR command. To use DBO/MONITOR, you need WORLD privilege.

Parameter

option
Specifies one of the following monitor operations:

- START

  Starts the Oracle CODASYL DBMS monitor process

- STOP

  Stops the Oracle CODASYL DBMS monitor process normally, either with process rundown or an immediate termination

- REOPEN_LOG

  Closes the current Oracle CODASYL DBMS monitor log file and opens a new version with the same name without stopping the monitor

Command Qualifiers

/ABORT=option
/NOABORT
Specifies how the monitor handles current processes before it stops. This qualifier is valid only with the STOP option. You can use the following options:

- The FORCEX option, or forced exit, stops the monitor immediately without allowing current processes to continue. This is the default for the /ABORT qualifier and can be specified as follows:

  ```
  $ DBO/MONITOR STOP/ABORT
  ```

- The DELPRC option, or delete process, stops the monitor and deletes the processes of all database users on the current node.

- The /NOABORT qualifier stops the monitor after allowing current processes to continue and run down. New users on the node will not be allowed to bind to any database, but existing database users will be able to complete their sessions normally. Once existing database users terminate, the database monitor shuts down. The default is /NOABORT.

/OUTPUT=file-spec
Specifies a device, directory, and file name for the monitor log. SYS$SPECIFIC:DBMMON.LOG; is the default. You can use this qualifier to save space on the system disk.

/PRIORITY=integer
Sets the base priority of the Oracle CODASYL DBMS monitor process. By default, the monitor runs at the highest interactive priority (15). This qualifier is valid only when used with the START option. Oracle does not recommend using /PRIORITY to decrease the base priority of the monitor. If you are running real-time processes that use Oracle CODASYL DBMS, then the priority of the database monitor should be greater (by at least 1) than the highest priority of any database user process.

/SWAP
/NOSWAP
Enables or disables swapping of the Oracle CODASYL DBMS monitor process. The default is /NOSWAP. This qualifier is valid only when used with the START option. If you allow the monitor process to swap, thrashing could occur and database users' processes can hang. Oracle does not recommend using the /SWAP qualifier. This qualifier remains solely for compatibility with previous versions of Oracle CODASYL DBMS.

/WAIT
/NOWAIT
Retains control of your process until the Oracle CODASYL DBMS monitor has actually stopped on that node. At this point, you are returned to the DCL prompt. Use this qualifier when you want to be sure the monitor has stopped. You can also use this qualifier to be sure that any $FORCEX or $DELPRCs requested were sent. This qualifier is valid only with the STOP option.

Specify /NOWAIT, the default, to have Oracle CODASYL DBMS return control to you immediately after you issue the DBO/MONITOR STOP command.

Example

```
$ DBO/MONITOR REOPEN_LOG
```

This command creates a new version of the database monitor log file. This is especially useful if the monitor log has grown too large and you want to start a new version of the file and delete the old version without shutting down the database monitor.

9.22 DBO/MOVE_AREA Command

Moves one or more storage areas to different disks or directories. You can also choose to move the database root file to a different location.

Format

DBO/MOVE_AREA root-file-spec area-name[, . . .]

Command Qualifiers	Defaults
/[NO]AFTER_JOURNAL[=file-spec]	See description
/[NO]AIJ_OPTIONS[=journal-opts-file]	See description
/[NO]AREA	/AREA
/[NO]CDD_INTEGRATE	CDD_INTEGRATE
/[NO]CHECKSUM_VERIFICATION	/CHECKSUM_VERIFICATION
/CLUSTER_NODES=integer	
/[NO]CONFIRM	/NOCONFIRM
/DIRECTORY=directory-spec	
/[NO]LOG	Current DCL verify value
/[NO]ONLINE	/NOONLINE
/OPTION=file-spec	
/OUTPUT=output-file	/OUTPUT=SYS$OUTPUT
/PAGE_BUFFERS=integer	/PAGE_BUFFERS=3
/PATH=cdd-repository-path	See description
/ROOT=root-file-spec	
/ROW_CACHE_OPTIONS=rc-opts-file	See description
/THREADS=n	/THREADS=??
/USERS=integer	

Command or Area Qualifiers	Defaults
/ATTRIBUTES	See description
/BLOCKS_PER_PAGE=integer	No changes
/EXTENSION=n	
/FILE=file-spec	
/LIVE	
/NOLIVE	See description
/READ_ONLY	See description
/READ_WRITE	See description
/SNAPSHOTS[=(option, . . .)]	
/NOSNAPSHOTS	See description
/THRESHOLDS=(pct1[,pct2[,pct3]])	

Description

The DBO/MOVE_AREA command lets you modify certain area parameters. All the files are processed simultaneously during the move operation. The DBO/MOVE_AREA command eliminates the need for intermediate storage media.

Command Parameters

root-file-spec
Specifies the name of the database root file for the database whose storage areas you want to move.

area-name [, . . .]
Specifies the name of one or more storage areas that you want to move.

Command Qualifiers

/AFTER_JOURNAL[=file-spec]
/NOAFTER_JOURNAL
Creates a new after-image journal file and turns on journaling. The /NOAFTER_JOURNAL qualifier turns off journaling. The default is to retain the current journaling state and to create a new journal file. You facilitate recovery by creating a new journal file because a single journal file cannot be applied across a move operation that changes an area page size.

/AIJ_OPTIONS[=journal-opts-file]
/NOAIJ_OPTIONS
Specifies how DBO is to handle after-image journaling and .AIJ file creation, using the following rules:

- If you specify the /AIJ_OPTIONS qualifier and provide a journal-opts-file, DBO enables journaling and creates the .AIJ file or files you specify for the database copy. If only one .AIJ file is created for the database copy, it will be an extensible .AIJ file. If two or more .AIJ files are created for the database copy, they will be fixed-size .AIJ files (as long as at least two .AIJ files are always available).
- If you specify the /AIJ_OPTIONS qualifier, but do not provide a journal-opts-file, DBO disables journaling and does not create any new .AIJ files.
- If you specify the /NOAIJ_OPTIONS qualifier, DBO retains the original journal setting (enabled and disabled) and retains the original .AIJ file.
- If you do not specify an /AFTER_JOURNAL, /NOAFTER_JOURNAL, /AIJ_OPTIONS, or /NOAIJ_OPTIONS qualifier, DBO retains the original journal setting (enabled or disabled) and the original .AIJ file state.

Note that you cannot use the DBO/MOVE_AREA command with the AIJ_OPTIONS qualifier to alter the journal configuration. However, you can use it to define a new after-image journal configuration. When you use it to define a new after-image journal configuration, it does not delete the journal files in the original configuration. Those can still be used for recovery.

The /AIJ_OPTIONS qualifier is valid only when no users are bound to the database and only when the root file is moved.

You can only specify one, or none, of the following after-image journal qualifiers in a single DBO command: /AFTER_JOURNAL, /NOAFTER_JOURNAL, /AIJ_OPTIONS, /NOAIJ_OPTIONS.

If you specify a journal-opts-file, it should appear in the following format:

```
JOURNAL [IS] {ENABLED | DISABLED} -
[RESERVE n] -
[ALLOCATION [IS] n] -
[EXTENT [IS] n] -
[BACKUPS [ARE] {MANUAL|AUTOMATIC} [FILE filename]] -
[OVERWRITE [IS] {ENABLED|DISABLED}] -
[SHUTDOWN_TIMEOUT [IS] n] -
[NOTIFY [IS] {ENABLED|DISABLED}] -
[CACHE [IS] {ENABLED FILE filename|DISABLED}]

ADD [JOURNAL] journal-name -
FILE filename -
[BACKUP_FILE filename] -
[ALLOCATION [IS] n]
```

See the DBO/SHOW AFTER_JOURNAL command for information on the format of a journal-opts-file.

/AREA
/NOAREA
Controls whether or not specific storage areas are moved. If you specify the /AREA qualifier, only the storage areas specified with the /OPTION qualifier or the area-name parameter are moved. If you specify /NOAREA, all the storage areas in the database are moved. The default is /AREA.

/CDD_INTEGRATE
/NOCDD_INTEGRATE
Specifies whether or not you want DBO to update the database instance information to reflect the existence of the database you are creating. The default is /CDD_INTEGRATE.

If you specify /NOCDD_INTEGRATE, DBO copies the database but does not record it in Oracle CDD/Repository. Therefore, the database you are creating will not appear in the output of a DBO/REPORT/FULL command.

Use this qualifier only in conjunction with the /ROOT qualifier.

/CHECKSUM_VERIFICATION
/NOCHECKSUM_VERIFICATION
Requests that the page checksum be verified for each page moved. The default is to perform this verification.

/CLUSTER_NODES=integer
Specifies the maximum number of nodes that can access the database in a VMScluster.

Use this qualifier only in conjunction with the /ROOT qualifier.

/DIRECTORY=directory-spec
Specifies the new location for the areas.

/LOG
/NOLOG
Specifies whether or not to report the process of the command on SYS$OUTPUT. Specify /LOG to request log output and /NOLOG to prevent it. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/ONLINE
/NOONLINE
Allows the specified storage areas to be moved without taking the database off line. This qualifier can only be used when the /AREA qualifier is used. The default is /NOONLINE. You cannot move a database root file when the database is on line. The /ONLINE qualifier cannot be used in conjunction with the /ROOT qualifier.

/OPTION=file-spec
Specifies a file whose text defines areas and their qualifiers. This feature makes it easier to define database characteristics without exceeding the character and token limits for DCL-level commands.

The format of the text in the file is:

{area-name [qualifier]...}...

There is no limit on the number of area-names; however, each area entry in the options file cannot exceed 256 characters and 128 tokens. Use of /OPTIONS supersedes the need for DBO indirect command files. The default file type is .DBO.

The /OPTION qualifier cannot be used in conjunction with the area-name parameter.

/PAGE_BUFFERS=integer
Specifies the number of buffers to be allocated for each file to be moved. The number of buffers used is twice the number specified; half is used for reading the file and half for writing the copy. Values specified may range from 1 to 5. The default value is 3. Larger values may improve performance, but they increase memory usage.

/PATH=cdd-repository-path
Specifies an Oracle CDD/Repository location containing the schema with which the database will be integrated. If you do not specify /PATH, DBO uses the schema path in the root file.

Use this qualifier only in conjunction with the /ROOT qualifier.

/ROOT=root-file-spec
Specifies the new location for the database root file. If not specified, the database root file is not moved.

/USERS=integer
Specifies the number of users who can access the database concurrently.

Use this qualifier only in conjunction with the /ROOT qualifier.

Command or Area Qualifiers

/BLOCKS_PER_PAGE=integer
Specifies a new page size for the storage area. You cannot decrease the page size of a storage area. If you change page sizes, Oracle recommends that you perform a database backup after a move operation. Performing the backup simplifies recovery if that becomes necessary.

/FILE=file-spec
Specifies the new location for the storage area. The default file type is .DBS.

/SNAPSHOTS [=(option[, . . .])]
Allows you to select snapshot options for storage areas. These options can be global (affecting the entire database) or local (affecting one or more specific areas). Local qualification overrides global qualification.

The following options are available for each storage area that has snapshots allowed:

- ALLOCATION=integer

  Defines the initial number of database pages in the snapshot file. The pages in the snapshot file are the same size as the pages in the corresponding storage area.

- FILE=file-spec

  Identifies the snapshot file name for this storage area. You can use /FILE=file-spec to specify any part of the full file specification for the snapshot file. The default file type is .SNP. If you omit the version number, DBO uses the latest version. If you omit any other portions of the file specification, DBO uses the corresponding parts of the storage area file specification. (See the DBO/CREATE/FILE option.)

  By default, DBO creates the snapshot file in the same directory as the storage area file. DBO derives the name of the snapshot file from the name of the storage area file using a file type of .SNP and the latest version.

/THRESHOLDS=(pct1[,pct2[,pct3]])
Specifies one, two, or three new threshold values for the storage area. The threshold values determine how much free space can be guaranteed to exist on a database page when storing records. Each threshold value represents a percentage of fullness on a data page. When a data page reaches the percentage of fullness defined by a given threshold value, the SPAM entry for the data page is updated to contain that threshold value. For example, if a record is large enough to occupy 50 percent of a database page, the Database Control System (DBCS) examines the threshold values to determine which pages have room for the new record.

Example

```
$ DBO/MOVE_AREA PARTS MARKET/DIRECTORY=DISKA:[SAMPLE]
```

If a storage area is on a disk that is logging error messages, you can move the storage area to another disk using the DBO/MOVE_AREA command. This example moves the MARKET storage area (MARKET.DBS and MARKET.SNP) of the PARTS database to the DISKA:[SAMPLE] directory.

9.23 DBO/OPEN Command

Opens a root file and maps its global section.

Format

DBO/OPEN root-file-spec [,...]

Command Qualifier	Default
/CACHE=NOENABLED	See description
/GLOBAL_BUFFERS=(options[,...])	See description
/[NO]STATISTICS	/NOSTATISTICS
/[NO]WAIT	/NOWAIT

Description

Once you have used DBO/OPEN to open a database, the database remains open and mapped until closed by a DBO/CLOSE command and all users have exited the database with the UNBIND or EXIT commands.

Entering a DBO/OPEN command before binding to a database results in more efficient bind operations. If the database were corrupted and .RUJ files were present, the database would be recovered.

If you modify a database for automatic opening, users can bind to that database at any time without first issuing a DBO/OPEN command. However, using the DBO/OPEN command saves mapping time when users bind to the database. If you modify the database for manual opening, the DBO/OPEN command must be entered before users can bind to the database.

Parameter

root-file-spec [,...]
Specifies one or more root files to open. If the root file is open, you get an informational message. An error returns if you specify a nonexistent root file. The default file type is .ROO.

Qualifiers

/CACHE=NOENABLED
This qualifier disables row caching. This qualifier is provided for use with Hot Standby databases. Row caching cannot be enabled on a Hot Standby database while replication is active. If it is enabled, the Hot Standby feature will not start.

/GLOBAL_BUFFERS=(options[,...])
Specifies a unique global buffer size for the target database on the node from which the DBO/OPEN command is issued by the database administrator. The following are valid options:

- BUFFERS=integer

  Specifies the number of global buffers for a database. Valid values are from 2 to 32768. The default value is equal to 5 times the value of the maximum number of users allowed to access the database simultaneously, as specified with the DBO/CREATE/USER or DBO/MODIFY/USER commands.

- MAXIMUM_PER_USER=integer

  Specifies the maximum numbers of global buffers a run unit can allocate. The default is the number database page buffers defined by the /BUFFERS qualifier.

/[NO]STATISTICS
This qualifier permits you to load previously saved statistics information (using DBO/CLOSE/STATISTICS) when the database is opened. The default is /NOSTATISTICS, which indicates statistics information is not loaded on database open.

The statistics information is stored in a node-specific database file, located in the database directory. The file is named database_node.RDS. For example, the PARTS database open on node MYNODE would use a statistics file named PARTS_MYNODE.RDS.

Note that clusterwide statistics information is not stored in the statistics file. This lets you decide on which nodes the statistics information should be initially loaded at database open time.

The statistics files may not be renamed or relocated since they contain node-specific information. The statistics files may be deleted, if they are no longer needed. They are not required in order to open a database.

The statistics files will not be loaded if the physical schema of the database has changed since the statistics file was created. This means that the addition or deletion of storage areas, and record caches will invalidate the statistics files. This restriction prevents incorrect statistics information from being loaded when intervening physical changes occurred to the database. Closing the database will update the statistics files and make them valid again.

If you specified the /STATISTICS qualifier on the DBO/OPEN command, then the statistics information will be automatically preserved in the event of abnormal database closure.

/[NO]WAIT
This qualifier specifies whether or not the system prompt should be returned before the database is completely open and available. Specify the /WAIT qualifier if you want the system prompt returned when the database is completely open and available. Specify /NOWAIT if you want the system prompt returned immediately, regardless of the state of the open operation. The /NOWAIT qualifier is the default.

Example

```
$ DBO/OPEN PARTS
```

This command opens the PARTS database.

9.24 DBO/OPTIMIZE Command

Creates an optimized after-image journal (.OAIJ) backup file.

Format

DBO/OPTIMIZE aij-file-spec oaij-file-spec

Command Qualifiers	Defaults
/ACCEPT_LABEL	See description
/ACTIVE_IO=max-reads	/ACTIVE_IO=3
/AFTER_JOURNAL	/AFTER_JOURNAL
/BUFFER_SIZE=integer	See description
/COMPRESSION[=ZLIB[=level]]	/NOCOMPRESSION
/[NO]CRC[=AUTODIN_II]	See description
/DENSITY=number	See description
/ENCRYPT=(option[,...])	
/FORMAT={OLD_FILE|NEW_TAPE}	/FORMAT=OLD_FILE
/[NO]GROUP_SIZE[=interval]	See description
/IDENTIFIER=user-id	
/LABEL=label-name-list	Internally generated labels
/[NO]LOG	Current DCL verify value
/[NO]MEDIA_LOADER	See description
/OUTPUT=file-spec	SYS$OUTPUT
/PROTECTION=file-protection	See description
/[NO]REWIND	/NOREWIND
/TAPE_EXPIRATION=date-time	The current time
/[NO]TRACE	/NOTRACE

9.24 DBO/OPTIMIZE Command

Description

This command optimizes an after-image journal (.AIJ) backup file (created with the /QUIET_POINT qualifier) for database recovery (rollforward) operations by eliminating unneeded and duplicate journal records, and by ordering journal records.

An optimized AIJ (.OAIJ) backup file created by the DBO/OPTIMIZE/AFTER_JOURNAL command provides better recovery performance for your database than an unoptimized .AIJ backup file. A benefit of this improved recovery performance is that the database is made available to users sooner.

The DBO/OPTIMIZE/AFTER_JOURNAL command is used to read an .AIJ file on disk and write the .OAIJ file to tape or disk.

Note

Because an .OAIJ file is not functionally equivalent to the original .AIJ file, the original .AIJ file should not be discarded after it has been optimized.

Parameter

aij-file-spec
Specifies the name of the .AIJ backup file to be optimized. You cannot specify an .OAIJ file, or an active .AIJ file.

The default file type is .AIJ.

oaij-file-spec
Specifies the name of the optimized .AIJ file to be produced by the DBO/OPTIMIZE/AFTER_JOURNAL command.

The default file type is .OAIJ.

Command Qualifiers

/AFTER_JOURNAL
Specifies that an .AIJ file is to be optimized. This is the default.

/ACCEPT_LABEL
Specifies that DBO should keep the current tape label it finds on a tape during a backup operation even if that label does not match the default label or that specified with the /LABEL qualifier. Operator notification does not occur unless the tape's protection, owner, or expiration date prohibit writing to the tape. However, a message is logged (assuming logging is enabled) to indicate that a label is being preserved and which drive currently holds that tape.

This qualifier is particularly useful when your backup operation employs numerous previously used (and thus labeled) tapes and you want to preserve the labels currently on the tapes.

If you do not specify this qualifier, the default behavior of DBO is to notify the operator each time it finds a mismatch between the current label on the tape and the default label (or the label you specify with the /LABEL qualifier).

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/ACTIVE_IO=max-writes
Specifies the maximum number of write operations to an OAIJ device that the DBO/OPTIMIZE/AFTER_JOURNAL command will attempt simultaneously. This is not the maximum number of write operations in progress; that value is the product of active system I/O operations and the number of devices being written to simultaneously.

The value of the /ACTIVE_IO qualifier can range from 1 to 5. The default value is 3. Values larger than 3 can improve performance with some tape drives.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/BUFFER_SIZE=integer
Specifies the maximum record size for the .OAIJ file. The size can vary between 2048 and 65,024 bytes. The default value is device dependent. The appropriate buffer size is a compromise between tape capacity and error rate.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/COMPRESSION
/COMPRESSION=ZLIB
/NOCOMPRESSION
Allows you to specify the compression method to use before writing data to the AIJ backup file. This may increase CPU usage during the backup, but may be justified by a smaller AIJ backup file when written as a disk file, or is being backed up over a busy network, or is being backed up to a tape drive that does not do its own compression. You probably do not want to specify the /COMPRESSION qualifier when you are backing up an AIJ file to a tape drive that does its own compression; in some cases doing so can actually result in a larger file.

If you specify the Compression qualifier without a value, the default is /COMPRESSION=ZLIB=6.

The level value (ZLIB=level) is an integer between 1 and 9 specifying the relative compression level with one being the least amount of compression and nine being the greatest amount of compression. Higher levels of the compression use increased CPU time while generally providing better compression. The default compression level of 6 is a balance between compression effectiveness and CPU consumption.

Note that the /COMPRESSION qualifier need only be used with the DBO/BACKUP/AFTER_JOURNAL command as the DBO /RECOVER command determines whether a backup file has been compressed from information saved within the backup file.

/CRC
/CRC=AUTODIN_II
/CRC=CHECKSUM
/NOCRC
Makes tape reliability equal to that of a disk. The /CRC and /CRC=AUTODIN_II qualifiers are synonomous. Typing CRC is sufficient to select the /CRC=AUTODIN_II qualifier. It is not necessary to type the entire qualifier. The /CRC=AUTODIN_II qualifier uses the AUTODIN-II polynomial for the 32-bit CRC calculation and provides the most reliable end-to-end error detection. This is the default qualifier for NRZ/PE (800/1600 bits/inch) tape drives.

The /CRC=CHECKSUM qualifier uses one's complement addition, which is the same computation used to do a checksum of the database pages on disk. This is the default qualifier for GCR (6250 bits/inch) tape drives and for TA78, TA79, and TA81 tape drives.

The /NOCRC qualifier disables end-to-end error detection. This is the default qualifier for TA90 (IBM 3480 class) drives.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

Note

The overall effect of the /CRC=AUTODIN_II, /CRC=CHECKSUM, and /NOCRC default qualifiers is to make tape reliability equal to that of a disk. If you retain your tapes longer than 1 year, the /NOCRC default might not be adequate. For tapes retained longer than 1 year, use the /CRC=CHECKSUM qualifier.

If you retain your tapes longer than 3 years, you should always use the /CRC=AUTODIN_II qualifier.

Tapes retained longer than 5 years could be deteriorating and should be copied to fresh media.

/DENSITY=number

Specifies the density at which the output volume is to be written. The default value is the format of the first volume (the first tape you mount). You do not need to specify this qualifier unless your tape drives support data compression or more than one recording density.

The /DENSITY qualifier is applicable only to tape drives. DBO returns an error message if this qualifier is used and the target device is not a tape drive.

Specify the /DENSITY qualifier as follows:

- For TA90E, TA91, and TA92 tape drives, specify the number in bits per inch as follows:
  - DENSITY = 70000 to initialize and write tapes in the compacted format.
  - DENSITY = 39872 or DENSITY = 40000 for the noncompacted format.
- For SCSI (Small Computer System Interface) tape drives, specify DENSITY = 1 to initialize and write tapes using the drive's hardware data compression scheme.
- For other types of tape drives, you can specify a supported density value between 800 and 160000 bits per inch.
- For all tape drives, specify DENSITY = 0 to initialize and write tapes at the drive's standard density.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/ENCRYPT=({VALUE=|NAME=}[,ALGORITHM=])

The /ENCRYPT qualifier specifies the encryption key and algorithm so that the optimize after image journal command can decrypt the save set file of a database backup. Failure to use the /ENCRYPT qualifier when the database back is encrypted will result in an error: *%DBO-F-ENCRYPTSAVSET, save set is*

encrypted, /ENCRYPT must be specified. Likewise, the correct encryption details must be provided or an error will be reported and the optimize will fail.

Specify a key value as a string or, the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters see HELP ENCRYPT.

This feature requires the OpenVMS Encrypt product to be installed and licensed on this system.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/FORMAT=OLD_FILE
/FORMAT=NEW_TAPE
Specifies the format of the files written by the DBO/OPTIMIZE/AFTER_ JOURNAL command.

If you specify the default, the /FORMAT=OLD_FILE qualifier, the DBO/OPTIMIZE command writes the file in a format that is optimized for a file-structured disk. If you specify the /FORMAT=NEW_TAPE qualifier, the DBO command writes the file in a format that is optimized for tape storage, including ANSI/ISO labeling and end-to-end error detection and correction.

When you specify the /FORMAT=NEW_TAPE qualifier and optimize the .AIJ file to tape, you must mount the backup media by using the DCL MOUNT/FOREIGN command before you issue the DBO/OPTIMIZE/AFTER_JOURNAL command. The tape must be mounted as a foreign volume. If you mount the tape as an OpenVMS volume (that is, you do not mount it as a foreign volume) and you specify the /FORMAT=NEW_TAPE qualifier, you receive a DBO-F-MOUNTFOR error.

The following tape qualifiers have meaning only when used in conjunction with the /FORMAT=NEW_TAPE qualifier:

- /ACCEPT_LABEL
- /ACTIVE_IO
- /BUFFER_SIZE
- /CRC
- /DENSITY
- /GROUP_SIZE
- /IDENTIFIER
- /LABEL
- /MEDIA_LOADER
- /PROTECTION
- /REWIND
- /TAPE_EXPIRATION

/GROUP_SIZE[=interval]
/NOGROUP_SIZE
Specifies the frequency at which XOR recovery blocks are written to tape. The group size can vary from 0 to 100. Specifying a group size of 0 or specifying the /NOGROUP_SIZE qualifier results in no XOR recovery blocks being written. The /GROUP_SIZE qualifier is applicable only to tape, and its default value is device

dependent. DBO returns an error message if this qualifier is used and the target device is not a tape drive.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/IDENTIFIER=user-id
Specifies the owner of the tape volume set. The owner is the user who will be permitted to restore the database. The user-id parameter must be one of the following types of OpenVMS identifier:

- A user identification code (UIC) in [group-name,member-name] alphanumeric format
- A UIC in [group-number,member-number] numeric format
- A general identifier, such as SECRETARIES
- A system-defined identifier, such as DIALUP

The /IDENTIFIER qualifier cannot be used with a backup operation to disk. When used with tapes, the /IDENTIFIER qualifier applies to all continuation volumes. Unless the /REWIND qualifier is also specified, the /IDENTIFIER qualifier is not applied to the first volume. If the /REWIND qualifier is not specified, the backup operation appends the file to a previously labeled tape, so the first volume can have a protection different from the continuation volumes.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/LABEL=label-name-list
Specifies the 1- to 6-character string with which the volumes of the .OAIJ file are to be labeled. The /LABEL qualifier is applicable only to tape volumes. You must specify one or more label names when you use the /LABEL qualifier.

You can specify a list of tape labels for multiple tapes. If you list multiple tape label names, separate the names with commas and enclose the list of names within parentheses.

Use the label that you specify for the DBO optimize after-journal operation when you issue the DBO/RECOVER command.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request that summary information about the optimize operation be reported on SYS$OUTPUT and /NOLOG to prevent this reporting. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/MEDIA_LOADER
/NOMEDIA_LOADER
Specifies that the tape device from which the .AIJ file is being read has a loader or stacker. Use the /NOMEDIA_LOADER qualifier to specify that the tape device does not have a loader or stacker.

By default, if a tape device has a loader or stacker, DBO should detect it. However, occasionally DBO does not detect that a tape device has a loader or stacker. Therefore, after reading the first tape, DBO issues a request to the operator for the next tape, instead of requesting the next tape from the loader or

stacker. Similarly, sometimes DBO behaves as though a tape device has a loader or stacker when actually it does not.

If you find that DBO is not detecting that your tape device has a loader or stacker, specify the /MEDIA_LOADER qualifier. If you find that DBO expects a loader or stacker when it should not, specify the /NOMEDIA_LOADER qualifier.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/OUTPUT=file-spec
Names the file where output from the /LOG and /TRACE qualifiers is sent. The default file type is .LIS. The default output is SYS$OUTPUT.

/PROTECTION=file-protection
Specifies the system file protection for the .OAIJ file produced by the DBO/OPTIMIZE/AFTER_JOURNAL command.

The default file protection varies, depending on whether or not you write the .OAIJ file to disk or tape. This is because tapes do not allow delete or execute access and the system account always has both read and write access to tapes. In addition, a more restrictive class accumulates the access rights of the less restrictive classes. If you do not specify the /PROTECTION qualifier, the default protection is as follows:

- S:RWED,O:RE,G,W if the .OAIJ file is written to disk
- S:RW,O:R,G,W if the .OAIJ file is written to tape

If you specify the /PROTECTION qualifier explicitly, the differences in protection applied for backups to tape or disk as noted in the preceding paragraph are applied. Thus, if you specify PROTECTION=(S,O,G:W,W:R), that protection on tape becomes (S:RW,O:RW,G:RW,W:R).

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/REWIND
/NOREWIND
Specifies that the magnetic tape that contains the .OAIJ file will be rewound before processing begins. The tape will be initialized according to the /LABEL and /DENSITY qualifiers. The /NOREWIND qualifier is the default and causes the .OAIJ file to be created starting at the current logical end-of-tape (EOT).

This qualifier is applicable only to tape drives.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/TAPE_EXPIRATION=date-time
Specifies the expiration date of the .OAIJ file. The default for this qualifier is now (the current time), meaning that the tape can be written over immediately.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/TRACE
/NOTRACE
Sends detailed information about each optimization operation to SYS$OUTPUT.

If you use the /NOTRACE qualifier, which is the default, detailed information about each optimization is not sent to SYS$OUTPUT.

Examples

1.
```
$ DBO/OPTIMIZE/AFTER_JOURNAL/FORMAT=OLD_FILE -
_$ PARTS.AIJ PARTS.OAIJ
```

   This command creates an .OAIJ file named PARTS.OAIJ and writes it to disk.

2.
```
$ MOUNT/FOREIGN $111$MUA0:
$ DBO/OPTIMIZE/AFTER_JOURNAL/LOG/REWIND -
$ /LABEL=(PARTS_AIJ_1) /FORMAT=NEW_TAPE PARTS.AIJ -
$ $111$MUA0:PARTS.OAIJ
```

   This command creates the optimized .AIJ file, PARTS.OAIJ and writes it to tape.

9.25 DBO/PERMIT_USER Commands

Adds, deletes, or lists user execution list (UEL) entries.

Format

DBO/PERMIT_USER[/action] root-file-spec [security-schema-name [,...]]

Action Qualifiers	Defaults
/ADD	
/DELETE	
/LIST	/LIST

Description

The DBO/PERMIT_USER command provides the only means of accessing the database UEL and listing or changing its contents. To secure access to the database, there is one OpenVMS access control list (ACL), which Oracle CODASYL DBMS calls UEL. The UEL consists of a set of access control entries (ACEs). The UEL maps specific database users to specific security schemas. Thus, it allows you to control database access despite the user's subschema view. Each database contains a UEL in its root file. Each ACE in the UEL contains two items: a user identification code or OpenVMS identifier and a security schema identifier. This pairing maps a particular UIC value or OpenVMS identifier to a specific security schema.

Immediately after database creation, access is denied to all users except the creator or database owner. The database owner is mapped to the NULL security schema. This allows only the owner to bind to the database and access data until the UEL is altered.

The NULL security schema is not a true security schema. It serves as a placeholder in the UEL and cannot be dumped with the DBO/DUMP command. Anyone mapped to the NULL security schema is granted full DML access to the database.

A DML program takes on different access rights when different users mapped to different security schemas run the program. Consequently, users mapped to different security schemas can run a program against the same database with the same subschema view and get different results depending on what information they could access.

Each DBO/PERMIT_USER command instance performs one of three types of action: adding UEL entries, deleting UEL entries, or listing UEL entries. An action qualifier specifies which of these actions will be performed. Each action qualifier requires a distinct command format. The following sections describe these three formats.

Parameters

root-file-spec
Specifies the root file of the database whose UEL you want to reference. The default file type is .ROO.

security-schema-name [,...]
Specifies one or more security schemas whose access control entries (ACEs) you want to list or change. You can specify an asterisk (*) wildcard to refer to all security schemas.

You can also specify NULL. The NULL security schema is not a true security schema. It serves as a placeholder in the UEL and cannot be dumped with the DBO/DUMP command. Anyone mapped to the NULL security schema is granted full DML access to the database.

Action Qualifiers

/ADD
/DELETE
/LIST
Specifies the type of operation to be performed on the UEL. You must position the operation qualifier globally (before the security-schema-name parameters) to refer to all security schemas named in the command.

The /ADD qualifier adds one or more new ACEs to the UEL. See the DBO/PERMIT_USER/ADD command for the complete description of this command.

The /DELETE qualifier deletes one or more ACEs from the UEL. See the DBO/PERMIT_USER/DELETE command for the complete description of this command. For a detailed description of the command with examples of commands and output, see the *Oracle CODASYL DBMS Database Security Guide*.

The default qualifier is /LIST. It lists all ACEs in the UEL on SYS$OUTPUT. If you prefer, you can place the ACE list in a file by specifying /OUTPUT=file-spec. See the DBO/PERMIT_USER/LIST command for a complete description of this command.

9.25.1 DBO/PERMIT_USER/ADD Command

Adds one or more access control entries to the UEL according to parameters and qualifiers that you specify.

DBO/PERMIT_USER/ADD Command

Format

DBO/PERMIT_USER/ADD root-file-spec security-schema-name [,...]

Command Qualifier	Default
/[NO]LOG	Current DCL verify value

Parameter Qualifiers	Defaults
/ENTRY=integer /IDENTIFIER=option	/ENTRY=1

Description

Use the DBO/PERMIT_USER/ADD command to secure your database and to associate specific users with security schemas. When you add entries to the UEL, remember that a user can be mapped to only one security schema. You must specify at least one security schema and one or more users that you want to map to that security schema. If you specify more than one security schema parameter, you must specify users by means of /IDENTIFIER qualifiers local to each security schema.

Use the /IDENTIFIER qualifier to specify the UICs or OpenVMS identifiers you want to map to the security schema. For example:

```
$ DBO/PERMIT/ADD/IDENTIFIER=([PERS,AL],[BKKP,*]) PERSONLDB CLERICAL2
```

This command adds two ACEs to the UEL for database PERSONLDB, mapping the UICs [PERS,AL] and [BKKP,*] to security schema CLERICAL2.

The /ENTRY qualifier is always global; you can specify only one entry per command and DBO inserts all new entries at the position you specify. Unless you specify the /ENTRY qualifier, DBO adds entries at the beginning of the UEL.

Parameters

root-file-spec
Specifies the root file of the database whose UEL you want to reference. The default file type is .ROO.

security-schema-name [,...]
Specifies one or more security schemas to which you intend to map users. You can also specify NULL. The NULL security schema is not a true security schema. It serves as a placeholder in the UEL and cannot be dumped with the DBO/DUMP command. Anyone mapped to the NULL security schema is granted full DML access to the database.

Command Qualifier

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request that DBO/PERMIT_USER report each UEL addition and /NOLOG to prevent this reporting. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

Parameter Qualifiers

/ENTRY=integer

Specifies an ordinal position in the UEL where you intend to add the new UEL entries. Positioning of entries in the UEL is critical. It is good practice to first enter a DBO/PERMIT_USER/LIST command to learn the order of the existing entries.

All entries should be mapped to the correct security schema before being mapped to the group's default security schema. Additionally, you should keep the ACEs for individual UICs or OpenVMS identifiers ahead of wildcard UIC or OpenVMS identifier entries. OpenVMS searches an ACL from the beginning until it finds an ACE that matches the user. If a wildcard ACE comes before an ACE for a specific user, that user will always match to the wildcard ACE. Thus you want to place the ACE for [x,y] before [x,*].

If you use /ENTRY with /ADD, the integer specifies the position of the new entry. If you do not specify an /ENTRY position, DBO inserts each new entry at the default position 1. If the integer is greater than the number of entries presently in the UEL, the new entry becomes last on the list.

/IDENTIFIER=(UIC,...)
/IDENTIFIER=(VMS_ID,...)

Identifies one or more users by their UIC or OpenVMS identifier who are mapped to the security schema. If you specify more than one security schema, an error results.

Specify UICs in the standard UIC format, which allows alphanumeric group and member values. Wildcards are allowed in both the group and member fields. Specify OpenVMS identifiers in the standard format, which allow 1 to 31 alphanumeric characters and contains at least 1 nonnumeric character. Refer to the OpenVMS system management documentation and the *OpenVMS System Services Reference Manual* for more information.

Examples

1.
```
$ DBO/PERMIT/ADD/ID=([PERS,AL],[BKKP,*]) PERSONLDB CLERICAL2
```

   This command adds two ACEs to the beginning of the UEL in database PERSONLDB, mapping the UICs [PERS,AL] and [BKKP,*] to security schema CLERICAL2.

   Unless you specify the /ENTRY qualifier, DBO adds entries at the beginning of the UEL.

2.
```
$ DBO/PERMIT/ADD PARTS -
_$ CLERK1/ENTRY=99/IDENTIFIER=([INV,JANE],[INV,SAM])
```

   This command maps users [INV,JANE] and [INV,SAM] to the security schema CLERK1.

   If the UEL has fewer than 99 entries, DBO places the two new entries at the end of the UEL; otherwise, DBO inserts them as entries 99 and 100 and increases higher entry numbers in the UEL by 2.

9.25.2 DBO/PERMIT_USER/DELETE Command

Deletes one or more access control entries (ACEs) from the UEL of the database according to the parameters and qualifiers you specify.

Format

DBO/PERMIT_USER/DELETE root-file-spec [security-schema-name [,...]]

Command Qualifiers	Defaults
/[NO]CONFIRM	See description
/[NO]LOG	Current DCL verify value

Parameter Qualifiers	Defaults
/ENTRY=integer	/ENTRY=1
/IDENTIFIER=option	

Description

There are three ways to specify entries for deletion:

- Use an /ENTRY qualifier to specify an entry by its position number. If an entry with that position number exists, DBO deletes it.
- Use an /IDENTIFIER qualifier to specify one or more OpenVMS identifier values. DBO deletes entries containing those identifiers.
- Specify a set of security schemas (or a security schema wildcard, indicating all security schemas in the root file). DBO deletes entries mapping users to those security schemas.

Observe the following rules when specifying UEL entries for deletion:

- You cannot use the security-schema-name parameter if the command contains an /ENTRY or /IDENTIFIER qualifier. However, you must use the security-schema-name parameter if you omit both /ENTRY and /IDENTIFIER. To specify a security-schema-name parameter, include it on the command line following the root-file-spec or enter both parameters in response to prompts.
- Specify an /ENTRY qualifier, an /IDENTIFIER qualifier, or a security schema parameter to identify entries for deletion. The security schema parameter can consist of one security schema name, a series of security schema names separated by commas, or a wildcard.
- A syntax error results if you include more than one /ENTRY qualifier, more than one /IDENTIFIER qualifier, or both an /ENTRY and an /IDENTIFIER qualifier in one DBO/PERMIT_USER/DELETE command. A syntax error also results if you include an /ENTRY or /IDENTIFIER qualifier with one or more security schema parameters. These syntax errors occur even when the multiple specifications do not conflict.

Unless you request otherwise (by specifying the /NOCONFIRM qualifier), the command displays a delete confirmation inquiry for each entry you specify. You must respond to this inquiry before the command can proceed.

Note

You must specify either an /ENTRY qualifier, an /IDENTIFIER qualifier, or a security schema parameter or wildcard. Otherwise, a fatal error will result. You are not prompted for a security schema parameter in this case. The prompt for security schema occurs only after you have responded to a prompt for the root file.

Parameters

root-file-spec
Specifies the root file of the database whose UEL you want to reference. The default file type is .ROO.

security-schema-name [,...]
Specifies one or more security schemas whose ACE you want to delete. If you specify a security-schema-name, you cannot specify the /ENTRY or /IDENTIFIER qualifiers.

You can also specify NULL. The NULL security schema is not a true security schema. It serves as a placeholder in the UEL and cannot be dumped with the DBO/DUMP command. Anyone mapped to the NULL security schema is granted full DML access to the database.

Command Qualifiers

/CONFIRM
/NOCONFIRM
Specify /CONFIRM to have DBO prompt you for confirmation of each UEL entry that you specify for deletion. This is the default for interactive processing.

Specify /NOCONFIRM to suppress the prompt. This is the default for batch processing.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request that DBO/PERMIT_USER report each UEL entry deletion and /NOLOG to prevent this reporting. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

Parameter Qualifiers

/ENTRY=integer
Specifies an ordinal position in the ACL where you intend to delete an ACE. It is good practice to first enter a DBO/PERMIT_USER/LIST command to learn the order of the ACEs. OpenVMS searches the ACL from the beginning until it finds an ACE matching the user. If a wildcard ACE comes before an ACE for a specific user, that user will always be matched to the wildcard ACE. Thus, you should keep ACEs for individual UICs or OpenVMS identifiers ahead of wildcard ACEs. If you do not specify an /ENTRY position, DBO inserts each row entry at the default position 1.

If you use the /IDENTIFIER qualifier or specify a security-schema-name in the same command line, an error results.

/IDENTIFIER=UIC,...
/IDENTIFIER=VMS_ID,...
Identifies one or more users by their UIC or OpenVMS identifier whose current UEL entries you want to delete. If you specify a UIC or OpenVMS identifier value for which no ACE exists in the UEL, an informational message results.

If you use the /ENTRY qualifier or specify a security-schema-name in the same command line, an error results.

Specify UICs in the standard UIC format, which allows alphanumeric group and member values. Wildcards are allowed in both the group and member fields. Specify OpenVMS identifiers in the standard format, which allows 1 to 31 alphanumeric characters and contains at least 1 nonnumeric character. Refer to the OpenVMS system management documentation and the *OpenVMS System Services Reference Manual* for more information.

Examples

1.
```
$ DBO/PERMIT_USER/DELETE/NOCONFIRM PERSONLDB *
```

   This command empties the UEL of the database PERSONLDB. To peruse the entire UEL and delete entries selectively, omit the /NOCONFIRM qualifier.

2.
```
$ DBO/PERMIT/DELETE/LOG PARTS CLERK1,BASELINE
delete entry for [ADM,BILL]? [N]: Y
%DBO-I-LOGPERDEL, revoked database access permission for [ADM,BILL]
delete entry for [DBMS,*]? [N]: N
delete entry for [DBMS,DONNA]? [N]: Y
%DBO-I-LOGPERDEL, revoked database access permission for [DBMS,DONNA]
delete entry for [PAYROLL,*]? [N]: N
```

   This command specifies all UEL entries that map to either of two security schemas, CLERK1 or BASELINE, and uses the default confirm feature to delete entries selectively.

   See the *Oracle CODASYL DBMS Database Security Guide* for additional examples.

9.25.3 DBO/PERMIT_USER/LIST Command

Lists the entries in the UEL.

Format

DBO/PERMIT_USER/LIST root-file-spec

Command Qualifier	Default
/OUTPUT=file-spec	SYS$OUTPUT

Description

This command lists all entries currently in the UEL. The list always appears on SYS$OUTPUT. In addition, you can write the list to a file by including the /OUTPUT qualifier.

Parameter

root-file-spec
Specifies the root file of the database whose UEL you want to list. The default file type is .ROO.

Command Qualifier

/OUTPUT=file-spec
Specifies the file to which the output is written. If you omit this qualifier, selected ASCII records are written to SYS$OUTPUT. The default file type is .LIS.

Example

```
$ DBO/PERMIT_USER/LIST PARTS
%DBO-I-LOGPERLSEC, entry 1: [SNOW,USER] uses DEFAULT_SECURITY_SCHEMA
%DBO-I-LOGPERLSEC, entry 2: [SNOW,USER1] uses NEW_SECURITY_SCHEMA
                          .
                          .
                          .
```

This command lists database users and the security schemas to which they are mapped.

9.26 DBO/RECLAIM Command

Reclaims deleted database keys.

Format

DBO/RECLAIM root-file-spec

Command Qualifiers	Defaults
/AREA=area_name	all areas
/FORCE_SPAM_UPDATE	/NOFORCE_SPAM_UPDATE
/[NO]LOG	/NOLOG
/PAGE_SKIP_LIMIT[=n]	
/RETRY_TIMEOUT[=n]	
/SPAM_SKIP_LIMIT[=n]	

Description

The DBO /RECLAIM command reclaims erased record dbkeys and locked space from database pages. It is designed for customer sites where database users attach to a database in DBKEY SCOPE IS FINISH mode.

By default DBKEY SCOPE is COMMIT. This implies that held DBKEY values are valid for the duration of the transaction, but a DBKEY of an erased record may be re-used by another transaction, and possibly a different database user. The logical name DBM$BIND_DBK_SCOPE_FINISH must be defined as 1 to change the scope to FINISH. In this case the dbkeys of erased records are not re-used and take up space on pages. (See DBO /SHOW LOGICAL_NAME /DESCRIPTION for further details of this logical name).

9.26 DBO/RECLAIM Command

The DBO /RECLAIM command reads and updates all pages in a storage area. Where possible, locked lines (aka dbkeys) and locked free space are released so they will be available for later allocation.

The DBO /RECLAIM command runs online (does not require exclusive access). However, if there are any users connected to the database in DBKEY SCOPE IS FINISH mode, the DBO /RECLAIM operation will have greatly reduced effect. In order to release all possible locked space, there should be no users bound to the database in DBKEY SCOPE IS FINISH mode. Further, to allow database page locked space to be reclaimed, the database session that owns the locked space must be unbound from the database. This can be accomplished by having each database user disconnect and reconnect to the database.

To avoid lock contention with other database users DBO /RECLAIM sequentially fetches data pages in a storage area but does not process data pages in the storage area that are currently locked in an incompatible mode by other users. However, once DBO /RECLAIM is processing a page, it can conflict with other database users who must wait for the page that DBO is processing. DBO /RECLAIM is designed to run in the background and avoid lock contention with other users; the qualifier /PAGE_SKIP_LIMIT enable an optional feature that will detect that other users are currently locking pages in the same storage area or SPAM interval that DBO /RECLAIM is processing and skip to the next storage area or SPAM interval based on a percent of the pages that DBO /RECLAIM is not able to process in a SPAM interval or storage area when reaching a specified limit.

Parameter

root-file-spec [,...]
Specifies one open root files. The default file type is .ROO.

Command Qualifiers

/AREA=area_names
all areas
Indicates the storage areas to be reclaimed. The default is to reclaim all storage areas. Values for /AREA include a single area name, a comma separated list of area names enclosed in parentheses or an asterisk (*) meaning all areas.

/FORCE_SPAM_UPDATE
/NOFORCE_SPAM_UPDATE
If /FORCE_SPAM_UPDATE is specified then SPAM thresholds are updated even if the page had no locked lines or if the SPAM threshold didn't appear to change. This allows DBO /RECLAIM to perform an online update after modifying the SPAM threshold values for a storage area.

/LOG
/NOLOG
Causes log messages to be issues to report progress for each storage area.

/PAGE_SKIP_LIMIT[=n]
If this qualifier is specified without the qualifier /SPAM_SKIP_LIMIT the current storage area will be skipped by DBO /RECLAIM once this percent of pages in the area have been ignored by DBO /RECLAIM due to lock contention.

If this qualifier is specified with the qualifier /SPAM_SKIP_LIMIT the current SPAM interval will be abandoned and DBO will skip to the next SPAM interval in the current storage area once this percent of pages in the current SPAM interval have been ignored by DBO /RECLAIM due to lock contention.

The minimum value that can be specified is 1 percent, the maximum value that can be specified is 100 percent, the default value is 25 percent.

/RETRY_TIMEOUT[=n]
Before completing, DBO /RECLAIM will perform one retry of the processing of any storage areas that have been skipped when /PAGE_SKIP_LIMIT or /SPAM_SKIP_LIMIT have been specified. The /RETRY_TIMEOUT qualifier specifies a wait time in seconds before DBO /RECLAIM reprocesses the storage areas where pages and SPAM intervals were previously skipped in order to allow more time for page contention to decrease. This qualifier can only be specified if the /PAGE_SKIP_LIMIT qualifier is also specified. The minimum value that can be specified is 0 seconds, the maximum value that can be specified is 3600 seconds (one hour), the default value is 60 seconds.

/SPAM_SKIP_LIMIT[=n]
This qualifier specifies the percent of SPAM intervals that can be skipped in the current storage area due to lock contention before the current storage area is skipped and the next storage area is processed.

This qualifier can only be specified if the /PAGE_SKIP_LIMIT qualifier is also specified. The minimum value that can be specified is 1 percent, the maximum value that can be specified is 100 percent, the default value is 25 percent.

Example

```
$ dbo/modify parts MARKET/threshold=(25,76,88)
$ dbo/reclaim parts/force_spam/area=MARKET/log
%DBO-I-RCLMAREA, Reclaiming area MARKET
%DBO-I-RCLMPRCT, 101 pages processed of 101 total pages for area MARKET,
    approximately 100 %
 ELAPSED:    0 00:00:00.05  CPU: 0:00:00.01  BUFIO: 15  DIRIO: 52  FAULTS: 138
```

This example modifies the THRESHOLD for the storage area MARKET, and then uses DBO /RECLAIM to update the SPAM thresholds according to the new configuration.

```
$ DBO/RECLAIM/LOG/PAGE_SKIP_LIMIT=4/SPAM_SKIP_LIMIT=1/RETRY_TIMEOUT=20 PARTS
%DBO-I-RCLMAREA, Reclaiming area DBMS_USER3:[DB]MAKE.DBS;1
%DBO-I-RCLMPRCT, 101 pages processed of 101 total pages for area
    DBMS_USER3:[DB]MAKE.DBS;1, approximately 100 %
%DBO-I-RCLMAREA, Reclaiming area DBMS_USER3:[DB]BUY.DBS;1
%DBO-I-RCLMPRCT, 101 pages processed of 101 total pages for area
    DBMS_USER3:[DB]BUY.DBS;1, approximately 100 %
%DBO-I-RCLMAREA, Reclaiming area DBMS_USER3:[DB]MARKET.DBS;1
%DBO-I-RCLMPRCT, 101 pages processed of 101 total pages for area
    DBMS_USER3:[DB]MARKET.DBS;1, approximately 100 %
%DBO-I-RCLMAREA, Reclaiming area DBMS_USER3:[DB]PERSONNEL.DBS;1
%DBO-I-RCLMPRCT, 101 pages processed of 101 total pages for area
    DBMS_USER3:[DB]PERSONNEL.DBS;1, approximately 100 %
ELAPSED:  0 00:00:00.36  CPU: 0:00:00.04  BUFIO: 42  DIRIO: 125  FAULTS: 135
```

DBO /RECLAIM uses /PAGE_SKIP_LIMIT=4 to reduce lock contention by detecting conflicts and moving to different pages, even subsequent SPAM intervals.

9.27 DBO/RECOVER Command

Completes a database reconstruction by processing past transactions from after-image journal (.AIJ) files or optimized after-image journal (.OAIJ) files against a database restored from a backup file.

Format

DBO/RECOVER [aij-file-spec [,...]]

Command Qualifiers	Defaults
/ACTIVE_IO=max-reads	/ACTIVE_IO=3
/AIJ_BUFFERS=integer	See description
/AREAS=area-name [,...]	
/[NO]AUTOMATIC	/NOAUTOMATIC
/CONFIRM	/NOCONFIRM
/ENCRYPT=(option[,...])	See description
/FORMAT={OLD_FILE|NEW_TAPE}	/FORMAT=OLD_FILE
/JUST_CORRUPT	
/LABEL=label-name-list	None
/[NO]LOG	Current DCL verify value
/[NO]MEDIA_LOADER	See description
/[NO]ONLINE	/NOONLINE
/[NO]OUTPUT	SYS$OUTPUT
/RESOLVE	
/[NO]REWIND	/NOREWIND
/ROOT=root-file-spec	
/STATE=type	
/[NO]TRACE	/NOTRACE
/UNTIL=absolute-time	

Description

You can use the DBO/RECOVER command to apply the contents of after-image journal files to a restored copy of your database. Oracle CODASYL DBMS rolls forward the transactions in .AIJ files into the restored copy of the database.

The DBO/RECOVER command accepts a list of .AIJ or .OAIJ file names. Unless you specify the /NOAUTOMATIC qualifier, the DBO/RECOVER command attempts to automatically complete the recovery operation by applying the journal files currently associated with the database in the current journal configuration if they are in the recovery sequence. For example, if you specify the following DBO command, Oracle CODASYL DBMS will recover not only AIJ1, but also AIJ2, AIJ3, and so on, for all journal files in the recovery sequence:

```
$ DBO/RECOVER AIJ1
```

However, note that this automatic recovery feature means that if you want to specify a termination condition, you must specify the /UNTIL qualifier.

If you have backed up your .AIJ files (using the DBO/BACKUP/AFTER_ JOURNAL command), these .AIJ files are no longer part of the current journal configuration and automatic recovery will not take place because Oracle CODASYL DBMS does not know where to find the .AIJ files. (There is one exception to this rule: if the only .AIJ file that has been backed up is the first .AIJ file in the recovery sequence, then automatic recovery will occur. You specify

the backed up .AIJ file on the DBO command line and DBO knows where the remaining on-disk .AIJ files reside.)

When automatic recovery does not, or cannot occur, you must specify the complete list of .AIJ files on the DBO/RECOVER command line to return your database to the desired state.

If your backup files were created using the /NOQUIET_POINT qualifier, you must provide the names of all the .AIJ files in just one command. In addition, you must be careful to apply these .AIJ files to the database in the order in which they were created. Oracle CODASYL DBMS checks the validity of the journal file entries against your database and applies only appropriate transactions. If none of the transactions applies, you will receive a warning message.

You can access your database for retrieval of data between recovery steps, but you must not perform additional updates if you want to perform more recovery steps.

If a system failure causes a recovery step to abort, you can simply issue the DBO/RECOVER command again. Oracle CODASYL DBMS scans the .AIJ file until it finds the first transaction that has not yet been applied to your restored database. DBO begins recovery at that point.

Note that the DBO/RECOVER command permits a full after-image journal rollforward operation only on an unmodified database. Attempting to perform a full AIJ rollforward operation on a modified database results in the following exception:

```
%DBO-F-DBMODIFIED, database has been modified; full AIJ roll-forward
    not possible
```

Parameter

aij-file-spec [,...]
Specifies a list of after-image journal (.AIJ) files to be applied to the database. You must apply the .AIJ files in the order in which they were created by the DBO/BACKUP/AFTER_JOURNAL command. In other words, the oldest .AIJ file must be first in the list.

The default file type is .AIJ.

Command Qualifiers

/ACTIVE_IO=max-reads
Specifies the maximum number of read operations from a backup device that the DBO/RECOVER command will attempt simultaneously. This is not the maximum number of read operations in progress; that value is the product of active system I/O operations and the number of devices being written to simultaneously.

The value of the /ACTIVE_IO qualifier can range from 1 to 5. The default value is 3. Values larger than 3 can improve performance with some tape drives.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/AIJ_BUFFERS=integer
Specifies the number of buffers to be used by the rollforward process. The default is 20 buffers. However, if your database is using global buffering, this value is overridden by the maximum number of global buffers per run unit. The maximum number of global buffers per run unit is set with the /GLOBAL_BUFFERS=MAXIMUM_PER_USER qualifier on the DBO/CREATE and DBO/MODIFY commands.

Use the DBO/DUMP/AFTER_JOURNAL/OPTION=STATISTICS command to display a recommended value for the /AIJ_BUFFERS qualifier.

/AREAS=area-name [,...]
Specifies one or more areas you want to roll forward. A recovery operation by area rolls the specified areas forward to current time or the time specified by the /UNTIL qualifier. If you have restored the database by area, you do not need to specify the area to be recovered again on the DBO/RECOVER command line. (Such areas are marked inconsistent until they are rolled forward to the current time of the database. The area cannot be readied until it is rolled forward.) However, if you do not respecify the area, DBO will recover the areas (as necessary) up to the current time and then roll forward all areas to the latest point on the .AIJ file.

/AUTOMATIC
/NOAUTOMATIC
Indicates that DBO/RECOVER is to use the existing after-journal files to perform the rollforward operation, if appropriate. This qualifier is useful for full, by-area, and by-page AIJ recovery operations.

After AIJ information has been restored to the database root data structures the DBO/RECOVER utility has all the information it needs to perform an automatic AIJ recovery operation.

The .AIJ file specification is optional with the /AUTOMATIC qualifier. You can also use the /ROOT command qualifier to indicate that the specified database is to be recovered automatically.

For example, the following command indicates that automatic after-image journal recovery is to start after the specified journal file has been recovered, if possible.

```
$ DBO/RECOVER/AUTOMATIC PARTS.AIJ
```

The next command indicates that automatic after-image recovery is to start with whatever journal file is appropriate to recover the specified database. It is not necessary to specify any after-image journal files if the /ROOT command qualifier is specified.

```
$ DBO/RECOVER/AUTOMATIC/ROOT=PARTS.ROO
```

/CONFIRM
/NOCONFIRM
Specify /CONFIRM to have DBO prompt for confirmation that the intended .AIJ file is ready to be rolled forward.

Use the /CONFIRM qualifier in conjunction with the /RESOLVE qualifier to specify whether or not to have DBO prompt you for confirmation of each distributed transaction state you alter.

/CONFIRM is the default for interactive processing.

Specify /NOCONFIRM to suppress the confirmation. /NOCONFIRM is the default for batch processing.

/ENCRYPT=({VALUE=|NAME=}[,ALGORITHM=])
The /ENCRYPT qualifier specifies the encryption key and algorithm so that the recovery can decrypt the save set file of a database after image journal backup. Failure to use the /ENCRYPT qualifier when the backup is encrypted will result in an error: *%DBO-F-ENCRYPTSAVSET, save set is encrypted, /ENCRYPT must*

be specified. Likewise, the correct encryption details must be provided or an error will be reported and the restore will fail.

Specify a key value as a string or, the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters see HELP ENCRYPT.

This feature requires the OpenVMS Encrypt product to be installed and licensed on this system.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/FORMAT=OLD_FILE
/FORMAT=NEW_TAPE
Specifies whether or not the backed up or optimized .AIJ file was written in the old (disk-optimized) or the new (tape-optimized) format. You must specify the same /FORMAT qualifier as was used with the DBO/BACKUP/AFTER_JOURNAL command or the DBO/OPTIMIZE/AFTER_JOURNAL command. If your .AIJ file resides on disk, you should use the /FORMAT=OLD_FILE qualifier.

If you specified the /FORMAT=OLD_FILE qualifier when you optimized or backed up the .AIJ file to tape, you must mount the backup media with the DCL MOUNT command before you issue the DBO/RECOVER/AFTER_JOURNAL command. Because DBO will use RMS to read the tape, the tape must be mounted as an OpenVMS volume (that is, do not specify the /FOREIGN qualifier to the DCL MOUNT command).

If you specify the /FORMAT=NEW_TAPE qualifier, you must mount the backup media using the DCL MOUNT/FOREIGN command before you issue the DBO/DUMP/AFTER_JOURNAL command.

The following tape qualifiers have meaning only when used in conjunction with the /FORMAT=NEW_TAPE qualifier:

- /ACTIVE_IO
- /LABEL
- /MEDIA_LOADER
- /REWIND

The /FORMAT=OLD_FILE qualifier is the default.

/JUST_CORRUPT
Allows you to perform a by-page or by-area recovery of the database, depending on the extent of database corruptions. It specifies the pages and areas logged inconsistent in the Corrupt Page Table (CPT) to be recovered with the DBO/RECOVER command.

If you perform this type of recovery operation on line, only the pages being recovered will be locked. The remainder of the database can be read and updated by applications.

You cannot use the /AREAS qualifier with the /JUST_CORRUPT qualifier. The /AREAS qualifier implies recovery for all named areas and pages in those areas.

If you specify the /UNTIL qualifier with the /JUST_CORRUPT qualifier, the recovery operation recovers a page until it is consistent with its storage area and ignores the time specified by the /UNTIL qualifier.

/LABEL=label-name-list
Specifies the 1- to 6-character string with which the volumes of the backup file are labeled. The /LABEL qualifier is applicable only to tape volumes. You must specify one or more label names when you use the /LABEL qualifier.

You can specify a list of tape labels for multiple tapes. If you list multiple tape label names, separate the names with commas and enclose the list of names within parentheses.

In a normal AIJ recovery operation, the /LABEL qualifier you specify should be the same /LABEL qualifier you specified with the AIJ backup operation.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request that summary information about the recovery operation be reported on SYS$OUTPUT and /NOLOG to prevent this reporting. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/MEDIA_LOADER
/NOMEDIA_LOADER
Specifies that the tape device from which the .AIJ file is being read has a loader or stacker. Use the /NOMEDIA_LOADER qualifier to specify that the tape device does not have a loader or stacker.

By default, if a tape device has a loader or stacker, DBO should detect it. However, occasionally, DBO does not detect that a tape device has a loader or stacker. Therefore, after reading the first tape, DBO issues a request to the operator for the next tape, instead of requesting the next tape from the loader or stacker. Similarly, sometimes DBO behaves as though a tape device has a loader or stacker when it actually does not.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/ONLINE
/NOONLINE
Allows the specified storage areas to be recovered without taking the database off line. Access is restricted only to the area being recovered. The rest of the database is available for access. This qualifier can only be used in conjunction with the /AREAS qualifier. The default is /NOONLINE.

/OUTPUT
Directs output, including that from the /TRACE and /LOG qualifiers, to a file. The default file type is .LIS. If the /OUTPUT qualifier is not specified, output is directed to SYS$OUTPUT.

/RESOLVE
Alters the state of incomplete distributed transactions in the .AIJ file that are associated with blocked users. You must alter the state in every .AIJ file affected.

The /RESOLVE qualifier does the following:

- Displays identification information for a blocked user
- Alters the state of the user

- Prompts you for the state to which the user is to be altered (COMMIT or ROLLBACK), unless you specified the state with the /STATE qualifier
- Prompts for confirmation of the state you entered, unless you specified the /CONFIRM qualifier
- Continues with subsequent blocked users until it has displayed information for all blocked users

Use the /STATE qualifier with the /RESOLVE qualifier to eliminate the state prompt for the blocked users.

/REWIND
/NOREWIND
Specifies that the magnetic tape that contains the backup file will be rewound before processing begins. The /NOREWIND qualifier is the default and causes reading of the backup file to start at the current logical end-of-tape (EOT).

This qualifier is applicable only to tape drives.

Use this qualifier only in conjunction with the /FORMAT=NEW_TAPE qualifier.

/ROOT=root-file-spec
Specifies a copy of a database instead of the original whose file specification is in the .AIJ file. Use /ROOT to specify the new location of your database root file. This lets you roll forward a database copy (possibly residing on a different disk).

/STATE=type
Specifies the state to which all blocked users are altered.

State types are:

- COMMIT to change the state of a user to committed
- ABORT to change the state of a user to aborted

If you do not specify the /STATE qualifier with the /RESOLVE qualifier, you are prompted for a state of each blocked user.

/TRACE
/NOTRACE
Sends detailed information about each recovery operation to SYS$OUTPUT.

If you use the /NOTRACE qualifier, the default, detailed information about each recovery will not be sent to SYS$OUTPUT.

/UNTIL=absolute-time
Defines a date and time in absolute-time format. Use /UNTIL to limit the recovery to those transactions in the after-image journal file bearing a starting time and date stamp no later than the specified absolute time. Thus, you can specify recovery up to the point you think the error occurred. This qualifier cannot be used when recovering with optimized after-image journal (.OAIJ) files.

If the absolute-time string contains both a date and a time of day, you must enclose the entire string in quotation marks (" "). For example, these two commands are valid:

```
$ DBO/RECOVER/UNTIL=10-FEB-1991
$ DBO/RECOVER/UNTIL="10-FEB-1991 12:00"
```

This command is not valid and will result in an error:

```
$ DBO/RECOVER/UNTIL=10-AUG-1991 12:00
```

Examples

1.
```
$ DBO/RECOVER/UNTIL="22-MAY-1991 13:35"/TRACE DISK:[USER]PARTSJ
```

   This command requests recovery from the after-image journal file PARTSJ.AIJ located on DISK in the USER directory. It specifies that recovery should continue until 1:35 p.m. on May 22, 1991. Additionally, it requests the display of detailed information about each recovery operation to SYS$OUTPUT.

2.
```
$ DBO/RECOVER/RESOLVE/LOG/NOCONFIRM PARTS.AIJ
%DBO-I-LOGOPNAIJ, opened journal file DBM$DISK:[USERA]PARTS.AIJ;2
%DBO-I-LOGRECDB, recovering database file DBM$DISK:[USERA]PARTS.ROO;1
    TSN=96

000000007E62CC83009340468A046A60        TID: ''j..F@...Ìb~....'
000000007E72F4E500933EA5BE0FD120        TM LOG_ID: ' Ñ.¥>..åôr~....'
00000000000000000000000120400B5E        RM LOG_ID: '^.@ ............'
202020325243535F534D424402440028        RM_NAME: '(.D.DBMS_SCR2   '
  0000000100010000011A0291000000        RM_NAME: '..............'
                      4E45455247        NODE NAME: 'GREEN'
                    574F4C4C4559        PARENT NODE NAME: 'YELLOW'

Do you wish to COMMIT/ABORT/IGNORE this transaction:COMMIT
%DBO-I-LOGRECOVR, 1 transaction committed
%DBO-I-AIJSUCCES, database recovery completed successfully
```

   This example shows that the state of records in the PARTS.AIJ file are set to COMMIT.

3.
```
$ DBO/DUMP/HEADER=CORRUPT PARTS

*------------------------------------------------------------------------
* Oracle CODASYL DBMS V7.0-00                    15-OCT-1996 14:50:21.78
*
* Dump of database header
*     Database: DISK:[USER]PARTS.ROO;1
*
*------------------------------------------------------------------------

Database Parameters:
    Root filename is "DISK:[USER]PARTS.ROO;1"

Dump of Corrupt Page Table:

Entries for storage area MAKE
-----------------------------

    Page 1
        - AIJ recovery sequence number is 0
        - Live area ID number is 1
        - Consistency transaction sequence number is 0:1
        - State of page is: inconsistent

$ DBO/RECOVER PARTS.AIJ /AREAS=MAKE /ONLINE/JUST_CORRUPT/LOG
```

```
%DBO-I-AIJBADPAGE, inconsistent page 1 from storage area
  DISK:[USER]MAKE.DBS;1 needs AIJ sequence number 0
%DBO-I-LOGRECDB, recovering database file _$111$DUA388:[USER]PARTS.ROO;1
%DBO-I-LOGOPNAIJ, opened journal file DISK2:[USER]PARTS.AIJ;1
%DBO-I-AIJONEDONE, AIJ file sequence 0 roll-forward operations
  completed
%DBO-I-LOGRECOVR, 3 transactions committed
%DBO-I-LOGRECOVR, 0 transactions rolled back
%DBO-I-LOGRECOVR, 0 transactions ignored
%DBO-I-AIJNOACTIVE, there are no active transactions
%DBO-I-AIJSUCCES, database recovery completed successfully
%DBO-I-AIJALLDONE, after-image journal roll-forward operations completed
%DBO-I-LOGSUMMARY, total 3 transactions committed
%DBO-I-LOGSUMMARY, total 0 transactions rolled back
%DBO-I-LOGSUMMARY, total 0 transactions ignored
%DBO-I-AIJSUCCES, database recovery completed successfully
%DBO-I-AIJGOODPAGE, page 1 from storage area DISK:[USER]MAKE.DBS;1
  is now consistent
%DBO-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence
 number needed will be 0

$ DBO/DUMP/HEADER=CORRUPT PARTS
*------------------------------------------------------------------------
* Oracle CODASYL DBMS V7.0-00                    15-OCT-1996 14:54:03.63
*
* Dump of database header
*     Database: DISK:[USER]PARTS.ROO;1
*
*------------------------------------------------------------------------

Database Parameters:
    Root filename is "DISK:[USER]PARTS.ROO;1"

Dump of Corrupt Page Table:

Corrupt page table is empty.
```

This example shows the status of a CPT. It then recovers the inconsistent page in the MAKE area. The DBO/DUMP command issued after the recovery operation shows that the CPT is now empty.

9.28 DBO/REPORT Command

Displays Oracle CDD/Repository information either about schemas and their respective storage schemas, security schemas, and subschemas or about schema data used by subschemas.

Format

DBO/REPORT [cdd-repository-path]

Command Qualifiers	Defaults
/BRIEF	/BRIEF
/[NO]FULL	/NOFULL
/OUTPUT=filename	SYS$OUTPUT

9.28 DBO/REPORT Command

Description

After the Oracle CDD/Repository path parameter has been evaluated, all schemas within the scope of that path are included in the output. Descendant metadata of the schemas (that is, storage schemas, subschemas, and security schemas) are also included in the report.

Parameter

cdd-repository-path
Specifies a pathname to an Oracle CDD/Repository directory or schema. It must be a valid Oracle CDD/Repository pathname. The default pathname is CDD$DEFAULT.

If you do not specify a full Oracle CDD/Repository pathname, beginning with CDD$TOP, DBO appends the Oracle CDD/Repository pathname you enter to the CDD$DEFAULT name.

If the Oracle CDD/Repository pathname refers to an Oracle CDD/Repository directory, DBO includes all schemas under that directory in the report. If the Oracle CDD/Repository pathname refers to a schema, DBO includes only that schema in the report.

Command Qualifiers

/BRIEF
Produces the same information as the /NOFULL qualifier. It remains in the syntax solely to provide compatibility with previous versions of Oracle CODASYL DBMS.

/FULL
/NOFULL
Displays the dictionary directory path to the schema, the names of the schema, storage schemas, security schemas, and subschemas at the specified dictionary directory, and the time and date stamp for each. The /NOFULL qualifier is the default.

Specify /FULL to display the /NOFULL information Repository database instance information for all reported schemas, security schemas, storage schemas, and subschemas.

/OUTPUT=file-spec
Names the file where output will be sent. The default file type is .LIS. The default output is SYS$OUTPUT.

Example

```
$ DBO/REPORT PARTS
```

```
Dictionary Directory: _CDD$TOP.SMITH
PARTS                                  20-FEB-1987 16:48:26.70 (1)
  Storage schemas:
    DEFAULT_STORAGE_SCHEMA             20-FEB-1987 16:49:02.37 (1)
    PARTST1                            20-FEB-1987 16:58:04.37 (1)
  Security schemas:
    DEFAULT_SECURITY_SCHEMA            20-FEB-1987 16:49:14.49 (2)
    TEST_SEC_SCHEMA                    20-FEB-1987 16:53:53.52 (1)
  Subschemas:
    DEFAULT_SUBSCHEMA                  25-FEB-1987 15:50:02.38
    PARTSS1                            20-FEB-1987 16:56:38.12
    PARTSS3                            20-FEB-1987 17:02:48.11
   .
   .
   .
```

This example shows the DBO/REPORT command and a sampling of the output.

9.29 DBO/RESOLVE Command

Resolves blocked users of distributed transactions for the specified database.

Format

DBO/RESOLVE root-file-spec

Command Qualifiers	Defaults
/[NO]CONFIRM	/CONFIRM
/[NO]LOG	/NOLOG
/PARENT_NODE=(node-name[,...])	
/PROCESS=(process-id[,...])	
/STATE=option	
/TSN=(tsn-number[,...])	

Description

The DBO/RESOLVE command resolves blocked users of distributed transactions by committing or rolling back the users. Before using the DBO/RESOLVE command, use the DBO/DUMP/USERS/STATE=BLOCKED command (described in Section 9.13.1) to get a list of blocked users. To determine how to resolve the blocked user, consult with the database administrators of the other databases involved in the distributed transaction. Human collaboration is necessary because the distributed transaction might have completed on one node before the coordinator became unreachable to the other nodes. When this occurs, the blocked user should be resolved to the same state as other users of the same distributed transaction.

The DBO/RESOLVE command performs the following actions:

- Displays identification information for each blocked user.
- Alters the state of the user. Once you change the state of a blocked user you cannot change it again.
- Prompts for the state to which the user is to be altered (COMMIT or ABORT), unless you specified the state with the /STATE qualifier.
- Prompts for confirmation of the state entered, unless you specified the /NOCONFIRM qualifier.

- Continues displaying information and prompting for subsequent blocked users until it has displayed information for all blocked users on that node.

Use the /PARENT_NODE, /PROCESS, or /TSN qualifiers to limit the number of blocked users DBO displays. If you do not specify one of these qualifiers, you will be prompted for each blocked user. Use the /STATE=BLOCKED qualifier of the DBO/DUMP/USER command as described in Section 9.13.1 to determine values for these qualifiers. The DBO/RESOLVE command is secured by its own Command Authorization List (CAL).

Parameter

root-file-spec
Specifies the root file of the database.

Command Qualifiers

/CONFIRM
/NOCONFIRM
Specify /CONFIRM to have DBO prompt for confirmation. This is the default for interactive processing.

Specify /NOCONFIRM to suppress the prompt. This is the default for batch processing.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request that summary information about the resolve operation is reported to SYS$OUTPUT. Specify /NOLOG to prevent this reporting. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/PARENT_NODE=(node-name[, . . .])
Specifies one or more node names to limit the selection of blocked users to those originating from the specified node. Do not use this qualifier in conjunction with the /PROCESS or /TSN qualifiers.

/PROCESS=(process-id[, . . .])
Specifies one or more process identifiers to limit the selection of blocked users to those associated with the specified process. Do not use this qualifier in conjunction with the /PARENT_NODE or /TSN qualifiers.

/STATE=option
Specifies the state to which all selected blocked users are altered. Two options are available:

- COMMIT to change the state of a user to committed
- ABORT to change the state of a user to aborted

If you do not specify the /STATE qualifier, you are prompted for the state of each blocked user.

/TSN=(tsn-number[, . . .])
Specifies one or more transaction sequence numbers (TSN) of the blocked user whose state you want to modify. Do not use this qualifier in conjunction with the /PARENT_NODE or /PROCESS qualifiers.

Example

```
$ DBO/RESOLVE/LOG/CONFIRM  PARTS
Blocked user with process ID 204005DA (remote access server)
    Stream ID is 1
    Monitor ID is 1
    Transaction ID is 2
    Recovery journal filename is "DBM$DISK:[USER1]PARTS.RUJ;1"
    Transaction sequence number is 96
    DECdtm Tid is  001A2920 001A2924 001A2928 001A292C
    Failure occurred on YELLOW
    Parent node is GREEN

Do you wish to COMMIT/ABORT/IGNORE this transaction: COMMIT
Do you really want to COMMIT this transaction? [N]: YES
%DBO-I-LOGRESOLVE, blocked transaction with TSN 96 committed
```

This example commits the transaction.

9.30 DBO/RESTORE Commands

There are two forms of the DBO/RESTORE command, each having its own qualifiers. These are:

DBO/RESTORE backup-file-spec [area-name [,...]]

This format restores a database using a backup file produced by DBO/BACKUP.

DBO/RESTORE/MULTITHREAD backup-file-spec [area-name [,...]]

This format restores a database using a backup file produced by DBO/BACKUP/MULTITHREAD.

These formats are described in the following sections.

9.30.1 DBO/RESTORE Command

Restores a database to its condition at the time of a full or incremental backup. The backup file must have been produced by the DBO/BACKUP command. Unless otherwise specified, DBO reintegrates the root file header information with Oracle CDD/Repository database instance information. Area names are verified before the restore operation begins to eliminate the possibility of root file and database inconsistencies.

Format

DBO/RESTORE backup-file-spec [area-name [,...]]

Command Qualifiers	Defaults
/[NO]AFTER_JOURNAL=file-spec	/AFTER_JOURNAL
/AREA	
/BUFFERS=integer	/BUFFERS=5
/[NO]CDD_INTEGRATE	/CDD_INTEGRATE
/[NO]CONFIRM	
/DIRECTORY=directory-spec	
/[NO]INCREMENTAL	/NOINCREMENTAL
/[NO]LOG	Current DCL verify value

/[NO]NEW_VERSION	/NONEW_VERSION
/PATH=cdd-repository-path	
/[NO]RECOVERY= [(AIJ_BUFFERS=integer)]	/NORECOVERY
/ROOT=root-file-spec	
/[NO]VERIFY	/NOVERIFY

Command or Area Qualifiers	Defaults
/[NO]BLOCKS_PER_PAGE= integer	/NOBLOCKS_PER_PAGE
/FILE=file-spec	
/SNAPSHOT=(option[,...])	
/THRESHOLDS=(pct1[,pct2[,pct3]])	

Description

The DBO/RESTORE command rebuilds a database from a backup file produced by a DBO/BACKUP command. The database is restored to the condition it was in when the backup was performed. For information on DBO/BACKUP, see the Section 9.5.1.

You can perform a full or incremental restore of the entire database. You can specify only one backup file parameter in a DBO/RESTORE command. If this parameter is a full backup file, you must not use the /INCREMENTAL qualifier. You must use the /INCREMENTAL qualifier if the parameter names an incremental backup file. You must use the /AREA and /INCREMENTAL qualifiers if the parameter names an incremental backup file for specific areas.

If DBO/RESTORE encounters a tape mounted /FOREIGN, it displays a message similar to the following:

```
%DBO-F-MOUNTNOFOR, device MUA0: must not be mounted /FOREIGN
```

If an invalid backup file name is specified on the DBO/RESTORE command, an error message is produced.

Parameters

backup-file-spec
Specifies a backup file produced by a previous DBO/BACKUP command. The default file type is .DBB.

Note

When restoring from tape, a backup file name longer than 17 characters produces an error. Backup truncates a backup file name because OpenVMS enforces the ANSI file name length limit of 17 characters. A workaround is to restore the database using the first 17 characters of the file name followed by a period (.) to indicate there is no file type.

area-name [,...]
Specifies one or more areas of the database that you want to restore or apply the area qualifiers.

Command Qualifiers

/AFTER_JOURNAL=file-spec
/NOAFTER_JOURNAL
Specifies a valid device and directory for a new after-image journal file and enables after-image journaling for the restored version of the database. You can specify a new device and directory for the after-image journal file. If you do not specify a device and directory, you will receive a warning message. To protect yourself against media failures, put the after-image journal file on a different device from your database files and your root file.

To restore a database by area, you must have after-image journaling enabled.

Specify /NOAFTER_JOURNAL to disable after-image journaling if it is currently enabled.

If you omit both qualifiers, information in the backup file is used to determine whether or not to enable after-image journaling. If after-image journaling was enabled at backup time, a new version of the default .AIJ file is created.

You cannot use either the /AFTER_JOURNAL or the /NOAFTER_JOURNAL qualifiers with the /AREA qualifier.

/AREA
Specifies that only the areas listed in the command are to be restored. You must have after-image journaling enabled to restore a database by area. When you restore by area, you should also recover the area to ensure that it is consistent with the rest of the database.

You cannot use the /AREA qualifier with either the /AFTER_JOURNAL or the /NOAFTER_JOURNAL qualifiers.

/BUFFERS=integer
Specifies the number of buffers used while restoring the database. The maximum number of buffers is 32. Careful selection of the value specified for the /BUFFERS qualifier helps I/O performance. Settings up through 10 are recommended. The default is /BUFFERS=5.

/CDD_INTEGRATE
/NOCDD_INTEGRATE
Specify /CDD_INTEGRATE, the default, to reintegrate the root file header information with Oracle CDD/Repository instance information as part of the restoration.

If you specify the /NOCDD_INTEGRATE qualifier, no reintegration occurs during the restoration, although you can reintegrate with a subsequent DBO/INTEGRATE command. Oracle recommends that you delay integration with Oracle CDD/Repository until after the restore finishes successfully.

/CONFIRM
/NOCONFIRM
Specify /CONFIRM to have DBO notify you of the name of the database on which you are performing the incremental restoration. Using /CONFIRM, you can be sure you have specified the correct root file name to which the incremental backup file is applied. This is the default for interactive processing.

Confirmation is especially important on an incremental restore if you have changed the root file name or created a new version of the database during restoration from the full backup file. A new version always requires the /ROOT qualifier on the incremental restore.

Specify /NOCONFIRM to have DBO apply the incremental backup to the database without prompting for confirmation. This is the default for batch processing.

DBO ignores this qualifier unless you also specify /INCREMENTAL.

/DIRECTORY=directory-spec
Specifies a global device and directory file specification portion for the root, database storage, and snapshot files of the database.

For example, the following two commands are equivalent:

```
$ DBO/RESTORE/FILE=DISK:[USER]/ROOT=DISK:[USER] -
_$ /SNAP=FILE=DISK:[USER] DBBUFULL
$ DBO/RESTORE/DIRECTORY=DISK:[USER] DBBUFULL
```

You can override the /DIRECTORY file specification with a global or local /FILE or /ROOT qualifier.

/INCREMENTAL
/NOINCREMENTAL
Specifies an incremental restore be performed. Use the /INCREMENTAL qualifier only when you have first issued a DBO/RESTORE command naming the full backup file and when the time and date stamps indicate that the full backup file was produced before the incremental backup file. The default, /NOINCREMENTAL, restores a database from a full backup file. Used in conjunction with the /AREA qualifier, it restores the database pages changed since the last full backup for only the areas specified.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request log output and /NOLOG to prevent it. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/NEW_VERSION
/NONEW_VERSION
Specifies whether or not new versions of database files should be produced if the device contains previous versions of database files.

If you use /NEW_VERSION, the new versions are produced. If you use /NONEW_VERSION, the default, an error occurs if an old copy exists of any of the database files being restored.

/PATH=cdd-repository-path
Specifies an Oracle CDD/Repository location containing the schema with which the database will be integrated. If you do not specify /PATH, DBO uses the schema pathname in the root file.

The /PATH qualifier is ignored if you use the /NOCDD_INTEGRATE qualifier to prevent attempts to integrate the database into Oracle CDD/Repository.

/ROOT=root-file-spec
Specifies the root file specification of a restored database. This gives the database a new location, a new name, or both. The default file type is .ROO.

DBO derives the file specification for the restored root file by starting with the /ROOT qualifier, then taking missing file specification components from the /DIRECTORY qualifier, and ultimately taking any other remaining file specification components from the original root file specification.

/VERIFY
/NOVERIFY
Verifies that the backup file is readable and that the records were written to the file in the correct order. When execution is successful, a completion message is logged. You must have specified /CHECKSUM when you created the database backup file. The DBO/RESTORE/VERIFY command does not actually restore the database from backup. It only verifies the backup file.

During the backup operation, DBO calculates a value, called a checksum, based on the data contained in that database record. The checksum is stored in the CHECKSUM field of the record header.

When verifying the backup file, DBO recalculates the checksum value from the database record. Then DBO compares the calculated value to the value stored in the CHECKSUM field in the record header. If the two values are the same, DBO continues to verify the next record. If the two values are different or if a checksum value is not found, an error occurs.

Command or Area Qualifiers

/BLOCKS_PER_PAGE=integer
/NOBLOCKS_PER_PAGE
Allows you to restore the database with larger page sizes than existed in the original database. This creates new free space on each page in the area and does not interfere with record clustering. DBO ignores this qualifier when it specifies an integer less than or equal to the current page size of the area.

If you use /NOBLOCKS_PER_PAGE, the default, DBO takes the page size for the area from the page size specified for the database you backed up.

/FILE=file-spec
Assigns a new file specification to the area name parameter it qualifies. It can specify a fully qualified file specification and can be used to restore all or selected database files in a different disk or directory.

Global use of /FILE is a convenient way to shorten the command string by specifying a default device and directory for the restored database storage files. The value of the global file specification becomes the default specification for all storage area files, including areas not specified. Any part of a complete file specification omitted from a local /FILE qualifier on an area parameter defaults to the value given in the global /FILE qualifier.

If you do not specify a database file name for a storage area, the file name in the backup file is the default. Similarly, if you specify no directory (using either a global or local /FILE qualifier), the directory in the backup file is the default.

/RECOVERY[=(AIJ_BUFFERS=integer)]
/NORECOVERY
Specifies that Oracle CODASYL DBMS perform a database restore and AIJ rollforward in a single operation. The optional AIJ_BUFFERS=integer argument specifies the number of buffers to be used by the rollforward process.

The DBO/RESTORE utility assumes that automatic recovery is desired. That is, if your restore operation includes AIJ information that was not captured in a previous backup, DBO/RESTORE performs automatic AIJ recovery. If your restore operation does not include such AIJ information, DBO/RESTORE does not perform automatic AIJ recovery, because automatic recovery is assumed. You must specify /NORECOVERY if you plan to restore an incremental backup. Because automatic recovery is assumed, you must specify /NORECOVERY if you plan to restore an incremental backup.

For example, the following command specifies that the database is to be restored from the full database backup file PARTS.DBB and that the database is to be automatically recovered using the restored AIJ information, if possible.

```
$ DBO/RESTORE/RECOVERY PARTS.DBB
```

/SNAPSHOTS=(option[,...])
Specifies a positional qualifier. If you use it globally, DBO restores all snapshot area files on the device and directory named using the FILE option. If used locally, the /SNAPSHOTS qualifier overrides a global /SNAPSHOTS qualifier for a particular area.

The /SNAPSHOTS qualifier allows you to select snapshot options for storage areas. The option, FILE, can be used whether or not snapshots are allowed. The options ALLOCATION, ENABLED, EXPANSION, and EXTENSION can only be used if snapshots are currently disabled. The following options are available for each storage area:

- ALLOCATION=integer

  Defines the initial number of database pages in the snapshot file. (The pages in the snapshot file are the same size as the pages in the corresponding storage area.) The default is 1, with additional pages added as needed. You must specify at least the first five letters of ALLOCATION. Otherwise, DBCS cannot distinguish between ALLOCATION and ALLOW and an error occurs.

- ALLOWED

  Specifies that snapshots are to be added to an area or areas that previously disallowed snapshots. You must use the full word ALLOW when entering this option. If only the first four letters are used, DBCS cannot distinguish ALLOW from ALLOCATION and an error occurs.

  When snapshots are added to the database, more room is needed on each database page to allow for snapshot activity. Consequently, DBO automatically increases the /BLOCKS_PER_PAGE setting by one. If this increment still does not allow enough room, you get an error message. If this happens, increase the setting for /BLOCKS_PER_PAGE by one more.

  You cannot add snapshots during an incremental or multithreaded restore operation.

- ENABLED
  NOENABLED

Specifies whether to enable or disable snapshot activity at this time. The /SNAPSHOTS qualifier requests snapshot capability for the storage area. The default NOENABLED option disables snapshot files temporarily. If you specify the ENABLED option, snapshot files are immediately enabled. You can enable snapshots later using the DBO/MODIFY command.

- EXPANSION=([NO]ENABLED,MINIMUM=integer, MAXIMUM=integer, PERCENT=integer,[NO]SPREAD)

  Defines how many data pages are added automatically to the snapshot file when its present allocation is full. This qualifier does not affect the current size of the snapshot file.

  Arguments for this option are:

  – ENABLED and NOENABLED turn area extension on and off. ENABLED is the default.

  – MINIMUM specifies the minimum number of pages required to extend the snapshot file when full. The default is 100.

  – MAXIMUM specifies the maximum number of pages required to extend the snapshot file when full. By default, the snapshot file can extend to an unlimited number of pages.

  – PERCENT specifies a percentage of the current snapshot file size required to extend the snapshot file when full. If the number of pages calculated is less than the MINIMUM or greater than the MAXIMUM, DBO extends the snapshot file by the MINIMUM or the MAXIMUM number of pages.

  – SPREAD or NOSPREAD is meaningful only if the snapshot file resides in a multiple-volume disk set. If you specify SPREAD, the extension will be spread evenly across the multiple volumes. NOSPREAD is the default. DBO reports a device-full error when all volumes in the disk set are full.

  DBO determines the size of the extension by the MINIMUM, MAXIMUM, and PERCENT values and by the current size of the snapshot file. The MINIMUM and MAXIMUM values define a range of allowable extension sizes. The utility first calculates the percentage of the current area size and then determines if that value is within the MINIMUM and MAXIMUM value range. The extension size is either the MINIMUM number of pages, the MAXIMUM number, or that number produced by calculating the percentage of the current area size, provided the percentage falls between the specified MINIMUM and MAXIMUM values.

- EXTENSION=integer
  NOEXTENSION

  Defines how many data pages to add to the snapshot file when its present allocation is full. By specifying EXTENSION=0 or NOEXTENSION, you prevent automatic extension. This option remains solely for compatibility with previous versions of Oracle CODASYL DBMS. EXPANSION values override EXTENSION values.

- FILE=file-spec

  Identifies the snapshot file name for this storage area. You can use /FILE=file-spec to specify any part of the full file specification for the snapshot file. The default file type is .SNP. If you omit the version number, DBO uses the latest version. If you omit any other portions of the file specification, DBO uses the corresponding parts of the storage area file specification.

By default, DBO creates the snapshot file in the same directory as the storage area file. The file specification for the restored snapshot file is derived by starting with the local /SNAPSHOTS=FILE qualifier for the storage area, if one exists. Missing file specification components are then taken from the global /SNAPSHOTS=FILE qualifier, if one exists. Then the file specification specified by the /DIRECTORY qualifier (if any) is used. Any remaining file specification components are taken from the original snapshot file specification at the time of the backup. DBO derives the name of the snapshot file from the name of the storage area file using a file type of .SNP and the latest version.

When you do not specify /SNAPSHOTS, the /FILE qualifier (if specified) determines the placement of snapshot files. In this case, DBO places the snapshot file in the same device and directory as its corresponding area file. If you omit both qualifiers, DBO restores snapshot areas according to the information stored in the backup file.

If you specify /SNAPSHOTS for a database or area that was not created with snapshots, DBO ignores the qualifier and no error message occurs.

/THRESHOLDS=(pct1[,pct2[,pct3]])

Changes a storage area's percentage fullness threshold. You can adjust space area management thresholds to improve future space usage in the storage area. Each threshold value represents a percentage of fullness on a data page. When a data page reaches the percentage of fullness defined by a given threshold value, the space area management (SPAM) entry for the data page is updated to contain that threshold value. A record occupies a certain percentage of a database page; DBCS examines the threshold values to determine which pages will have room for the new record. If you do not use /THRESHOLDS, DBO uses the storage area's original thresholds. DBO ignores this qualifier if the area does not have SPAM pages. The *Oracle CODASYL DBMS Database Design Guide* contains tutorial information on selecting threshold values.

Examples

1.
```
$ DBO/RESTORE/NEW_VERSION/AFTER_JOURNAL=DISK:[USER]PARTSAIJ -
_$ /NOCDD_INTEGRATE/LOG PARTSBCK -
_$ MAKE/THRESHOLDS=(65,75,80)/BLOCKS_PER_PAGE=3
```

   This command restores the database PARTS from the backup file PARTSBCK, requesting a new version of the database file.

   The command changes the after-image journal file location and name to DISK:[USER]PARTSAIJ, prevents reintegration with Oracle CDD/Repository, and displays the progress of the restore operation. For the storage area MAKE, the command changes the SPAM threshold values to 65, 75, and 80 and increases the number of blocks-per-page to 3.

2.
```
$ DBO/RESTORE/INCREMENTAL PARTSBCKINC
```

   This command restores the incremental backup file PARTSBCKINC.

3.
```
$ DBO/RESTORE/AREA PARTS_FULL MAKE
```

   This example restores the area MAKE using the backup file PARTS_FULL.

9.30.2 DBO/RESTORE/MULTITHREAD Command

Restores a database from multiple disks and tapes to its condition at the time of a full, incremental, or by-area backup. The backup file must have been produced by the DBO/BACKUP/MULTITHREAD command. Unless otherwise specified, DBO reintegrates the root file header information with Oracle CDD/Repository database instance information. Area names are verified before the restore operation begins to eliminate the possibility of root file and database inconsistencies. The root file creation date for the database to be incrementally restored is compared to that of the database on tape to avoid consistency errors. If the dates are different, a meaningful error is issued and the restore operation does not occur.

Format

DBO/RESTORE/MULTITHREAD backup-file-spec [area-name [,...]]

Command Qualifiers	Defaults
/ACTIVE_IO=integer	See description
/[NO]AFTER_JOURNAL=file-spec	See description
/[NO]AIJ_OPTIONS[=journal-opts-file]	See description
/AREA	See description
/BUFFER_SIZE=integer	
/[NO]CDD_INTEGRATE	/CDD_INTEGRATE
/CLUSTER_NODES=integer	See description
/[NO]CONFIRM	See description
/DIRECTORY=directory-spec	
/DUPLICATE	
/ENCRYPT=(option[,...])	See description
/GLOBAL_BUFFERS= global-buffer-options	See description
/INCREMENTAL	See description
/JOURNAL[=file-spec]	See description
/JUST_CORRUPT	See description
/LABEL=(label([,...]))	
/LOADER_SYNCHRONIZATION	See description
/LOCAL_BUFFERS= local-buffer-options	See description
/[NO]LOG	Current DCL verify value
/[NO]NEW_VERSION	/NONEW_VERSION
/[NO]MEDIA_LOADER	/See description
/[NO]ONLINE	/NOONLINE
/OPEN_MODE=option	See description
/OPTIONS=file-spec	
/PAGE_BUFFERS=integer	/PAGE_BUFFERS=3
/PATH=cdd-repository-path	
/[NO]RECOVERY= [(AIJ_BUFFERS=integer)]	/NORECOVERY
/[NO]REWIND	/NOREWIND
/ROOT=root-file-spec	
/USERS=integer	/USERS=10
/VOLUMES=n	See description

Command or Area Qualifiers	Defaults
/[NO]BLOCKS_PER_PAGE= integer	/NOBLOCKS_PER_PAGE

/FILE=file-spec	
/JUST_CORRUPT	See description
/MASTER	See description
/SNAPSHOT=option	
/THRESHOLDS=(pct1[,pct2[,pct3]])	

Description

The DBO/RESTORE/MULTITHREAD command rebuilds a database from a backup file produced by a DBO/BACKUP/MULTITHREAD command. The database is restored to the condition it was in when the backup was performed. You can use the multithreaded restore utility to restore a database from disk or tape. For information on DBO/BACKUP/MULTITHREAD see Section 9.5.3.

You can perform a full or incremental restore of the entire database, a by-area restore of storage areas, or a by-page restore of selected storage areas. You can specify only one backup file parameter in a DBO/RESTORE/MULTITHREAD command. If this parameter is a full backup file, you must not use the /INCREMENTAL qualifier. You must use the /INCREMENTAL qualifier if the parameter names a database incremental backup file. You must use the /AREA qualifier if the parameter names a by-area backup file.

Certain tape drives provide further performance improvements due to support for their tape cache mechanisms.

The tape characteristics of each volume are checked when attempting a multithreaded restore. If DBO encounters an error when attempting to mount the tape, either DBO terminates or the problem is reported and the operation is retried.

If a backup file is corrupt, multithreaded restore terminates with the message:

```
%DBO-F-BACFILCOR, Backup file is corrupt.
```

Parameters

backup-file-spec

Specifies a backup file produced by a previous command. The default file type is .DBF.

If you a restoring a backup file from tape and you use multiple tape drives, the backup-file-spec parameter must include the tape device specifications. Separate the device specifications with commas as follows:

```
$  DBO/RESTORE /REWIND $111$MUA0:PARTS_FULL_NOV30.DBF,$112$MUA1:
```

Note

When restoring from tape, a backup file name longer than 17 characters produces an error. Backup truncates a backup file name because OpenVMS enforces the ANSI file name length limit of 17 characters. A workaround is to restore the database using the first 17 characters of the file name followed by a period (.) to indicate there is no file type.

area-name [,...]

Specifies one or more areas of the database in which you want to restore or apply the area qualifiers.

Command Qualifiers

/ACTIVE_IO=integer
Specifies the maximum number of read operations from a backup device that DBO will attempt simultaneously. This is not the maximum number of read operations in progress; that value is the product of active system I/O operations and the number of devices being read from simultaneously.

The value of the /ACTIVE_IO qualifier can range from 1 to 7. For TA90 and TA90E tape drives, the maximum value is 7 and the default is 5. For all other tape drives the maximum value is 5 and the default is 3. Values larger than 3 might improve performance when using streaming tape drives. You should start with the default of 3 and experiment to find the best setting for your configuration. 3480/3490 and DLT type drives and their stack loaders are supported. The TA90E data compaction function is also supported.

/AFTER_JOURNAL=file-spec
/NOAFTER_JOURNAL
Specifies a valid device and directory for a new after-image journal file and enables after-image journaling for the restored version of the database. You can specify a new device and directory for the after-image journal file. If you do not specify a device and directory, you receive a warning message. To protect yourself against media failures, put the after-image journal file on a different device from your database files and your root file.

To restore a database by area, you must have after-image journaling enabled.

Specify /NOAFTER_JOURNAL to disable after-image journaling if it is currently enabled.

If you omit both qualifiers, information in the backup file is used to determine whether or not to enable after-image journaling. If after-image journaling is enabled at backup time, a new version of the default .AIJ file is created.

You cannot use either the /AFTER_JOURNAL or the /NOAFTER_JOURNAL qualifiers with the /AREA qualifier.

/AIJ_OPTIONS=journal-opts
/NOAIJ_OPTIONS
Specifies how DBO is to handle after-image journaling and .AIJ file creation, using the following rules:

- If you specify the /AIJ_OPTIONS qualifier and provide a journal-opts file, DBO creates the .AIJ file or files you specify for the restored database. If only one .AIJ file is created for the restored database, it will be an extensible .AIJ file. If two or more .AIJ files are created for the restored database, they will be fixed-size .AIJ files (as long as at least two .AIJ files are always available). Depending on what is specified in the options file, after-image journaling may either be disabled or enabled.
- If you specify the /AIJ_OPTIONS qualifier, but do not provide a journal-opts file, DBO disables journaling and does not create any new .AIJ files.
- If you specify the /NOAIJ_OPTIONS qualifier, DBO reuses the original .AIJ file configuration and recovers the journaling state (enabled or disabled) from the backed up .AIJ file.

- If you do not specify an /AFTER_JOURNAL, /NOAFTER_ JOURNAL, /AIJ_ OPTIONS, or /NOAIJ_OPTIONS qualifier, DBO recovers the journaling state (enabled or disabled) and tries to reuse the .AIJ file or files. (This is the same as specifying the /NOAIJ_OPTIONS qualifier.)

If the original database is lost or corrupted but the journal files are unaffected, you would typically restore the database without the use of either the /AIJ_ OPTIONS or the /AFTER_JOURNAL qualifier.

If you specify a journal options file, it should appear in the following form:

```
JOURNAL [IS] {ENABLED | DISABLED} -
[RESERVE n] -
[ALLOCATION [IS] n] -
[EXTENT [IS] n] -
[BACKUPS [ARE] {MANUAL|AUTOMATIC} [FILE filename]] -
[OVERWRITE [IS] {ENABLED|DISABLED}] -
[SHUTDOWN_TIMEOUT [IS] n] -
[NOTIFY [IS] {ENABLED|DISABLED}] -
[CACHE [IS] {ENABLED FILE filename|DISABLED}]

ADD [JOURNAL] journal-name -
FILE filename -
[BACKUP_FILE filename] -
[ALLOCATION [IS] n]
```

Note the following rules regarding this syntax:

- As shown in the preceding format, you can use the DCL continuation character (-), a hyphen, at the end of each line in the JOURNAL and ADD clauses. Although continuation characters are not required if you can fit each clause (JOURNAL or ADD clause) on a single line, using them will improve readability.
- Because the JOURNAL clause and the ADD clause are two separate clauses, a continuation character should not be used between the last option in the JOURNAL clause and the ADD clause (or clauses).
- The journal options file can only contain one JOURNAL clause, but it can contain several ADD clauses. However, the number of ADD clauses cannot exceed the number of reservations made with the RESERVE option in the JOURNAL clause. In addition, if you are enabling journaling, you must add at least one journal file.
- You can only specify one of each option (for example, one EXTENT clause, one CACHE clause, and so on) for the JOURNAL IS clause.

The clauses and options for the journal options file have the following meaning:

- JOURNAL IS ENABLED

  Enables after-image journaling. At least one ADD clause must follow. If this option is omitted, the current journaling state is maintained.

- JOURNAL IS DISABLED

  Disables after-image journaling. You can specify other options or ADD clauses but they will not take effect until journaling is enabled. The ADD clause is optional. If this option is omitted, the current journaling state is maintained.

- RESERVE n

  Allocates space for an after-image journal definition for a maximum of *n* after-image journal files. By default, no reservations are made.

- ALLOCATION IS n

  Specifies the default size of each .AIJ file (in blocks). If this option is omitted, the default allocation size is 512 blocks.

- EXTENT IS n

  Specifies the default size to extend a journal file if it is, or becomes, an extensible journal (in blocks). (If the number of available after-image journal files falls to one, extensible journaling will be employed.) If this option is omitted, the default extent size is 512 blocks.

- BACKUPS ARE MANUAL

  Enables manual backup operations. Automatic backup operations are not enabled. This is the default behavior.

- BACKUPS ARE AUTOMATIC [FILE filename]

  Enables automatic backup operations. Automatic backup operations are triggered by the filling of a journal file. The backup file will have the specified file name unless a different file is specified in the ADD clause. If this option is omitted, backup operations are manual.

  The FILE filename phrase is optional. If it is omitted, the automatic backup operation will only be triggered when a journal file for which a backup file has been specified becomes full. (See the description of the ADD clause.)

- OVERWRITE IS ENABLED

  Writes over journal files before they have been backed up. If this option is omitted, writing over files is disabled.

- OVERWRITE IS DISABLED

  Disables writing over journal files before they have been backed up. If this option is omitted, writing over files is disabled.

- SHUTDOWN_TIMEOUT IS n

  Sets the delay from the time a journal failure is detected until the time the database aborts all access and shuts itself down. The value of *n* is in minutes.

  If this option is omitted, the shutdown timeout is 60 minutes.

- NOTIFY IS ENABLED

  Enables operator notification when the journal state changes. If this option is omitted, operator notification is disabled. If enabled, the cluster central operator is notified when the journal state changes.

- NOTIFY IS DISABLED

  Disables operator notification when the journal state changes. If this option is omitted, operator notification is disabled.

- ADD[JOURNAL]journal-name

  Specifies the name for an after-image journal object reserved in the JOURNAL clause. The journal-name is the name of the journal object. A journal object is the journal file specification plus all the attributes (allocation, extent, and so on) given to it in the journal clause. This option is required.

- FILE filename

  Specifies the file specification for the after-image journal file being added. This option is required.

- BACKUP_FILE filename

Specifies the backup file name for automatic backup operations. Note that it is valid to specify a BACKUP_FILE in the ADD clause if you have specified BACKUPS ARE MANUAL in the JOURNAL clause (and provided a file name to serve in the event that you enable automatic backup operations).

/AREA
Specifies that only the areas listed in the command are to be restored. You must have after-image journaling enabled. When you restore an area, you should also recover the area to ensure that it is consistent with the rest of the database. See the Section 9.27 for information on the DBO/RECOVER command.

You cannot use the /AREA qualifier with either the /AFTER_JOURNAL or the /NOAFTER_JOURNAL qualifiers.

/BUFFER_SIZE=integer
Specifies the maximum record size for the backup file. The size can range from 2048 to 65,024 bytes. In general, you should not override the record size selected by multithreaded backup because it determines an efficient size for the tape drive being used.

/CDD_INTEGRATE
/NOCDD_INTEGRATE
Specify /CDD_INTEGRATE, the default, to reintegrate the root file header information with Oracle CDD/Repository instance information as part of the restore operation.

If you specify the /NOCDD_INTEGRATE qualifier, no reintegration occurs during the restore operation, although you can reintegrate with a subsequent DBO/INTEGRATE command. Oracle recommends that you delay integration with Oracle CDD/Repository until after the restore operation finishes successfully.

/CLUSTER_NODES=integer
Specifies the maximum number of nodes in a VMScluster configuration that can access a database simultaneously. You can modify this number using DBO/MODIFY. The minimum is 16. The maximum number supported is 96 nodes. If /CLUSTER_NODES is not supplied, the value that was in effect when the database was backed up is used.

/CONFIRM
/NOCONFIRM
Specify /CONFIRM to have DBO notify you of the name of the database on which you are performing the incremental restore operation. Using /CONFIRM, you can be sure you have specified the correct root file name to which the incremental backup file will be applied. This is the default for interactive processing.

Confirmation is especially important on an incremental restore operation if you have changed the root file name or created a new version of the database during restore operation from the full backup file. A new version always requires the /ROOT qualifier on the incremental restore operation.

Specify /NOCONFIRM to have DBO apply the incremental backup to the database without prompting for confirmation. This is the default for batch processing.

DBO ignores this qualifier unless you also specify /INCREMENTAL.

/DIRECTORY=directory-spec

Specifies a global device and directory file specification portion for the root, database storage, and snapshot files of the database.

For example, the following two commands are equivalent:

```
$ DBO/RESTORE/MULTITHREAD/FILE=DISK:[USER]/ROOT=DISK:[USER] -
_$ /SNAP=FILE=DISK:[USER] DBBUFULL
$ DBO/RESTORE/MULTITHREAD/DIRECTORY=DISK:[USER] DBBUFULL
```

You can override the /DIRECTORY file specification with a global or local /FILE or /ROOT qualifier.

/DUPLICATE

Specifies a new copy of a database is to be created. This qualifier ensures that any new AIJ activity can only be applied to the new database and not to the original database. This will protect the integrity of the original database from any changes to the copied database. This is useful when performing testing on a database to improve performance or when you install a database at a remote site or system.

/ENCRYPT=({VALUE=|NAME=}[,ALGORITHM=])

The /ENCRYPT qualifier specifies the encryption key and algorithm so that the restore can decrypt the save set file of a database backup. Failure to use the /ENCRYPT qualifier when the database back is encrypted will result in an error: *%DBO-F-ENCRYPTSAVSET, save set is encrypted, /ENCRYPT must be specified*. Likewise, the correct encryption details must be provided or an error will be reported and the restore will fail.

Specify a key value as a string or, the name of a predefined key. If no algorithm name is specified the default is DESCBC. For details on the Value, Name and Algorithm parameters see DCL HELP ENCRYPT.

This feature requires the OpenVMS Encrypt product to be installed and licensed on this system.

/GLOBAL_BUFFERS=global-buffer-options

Allows you to change the default global buffer parameters when you restore a database. The following options are available:

- ENABLED

  Use this option to enable global buffering for the database being restored.

- NOENABLED

  Use this option to disable global buffering for the database being restored. This is the default.

- TOTAL=integer

  Use this option to specify the number of buffers that will be available for all users. The minimum value you can specify is 2; the maximum value you can specify is the global buffer count stored in the .ROO file.

- USER_LIMIT=integer

  Use this option to specify the maximum number of buffers available to each user.

/INCREMENTAL
Specifies that the backup file on which the restore operation is being performed is an incremental backup file. You must specify the /INCREMENTAL qualifier if the backup file was created with the DBO/BACKUP/INCREMENTAL command.

By default, the DBO/RESTORE operation assumes that the backup file is a full backup file.

/JOURNAL=file-spec
Allows you to specify a journal file to be used to improve tape performance by a restore operation (including a by-area or just-corrupt restore operation).

The backup operation creates the journal file and writes to it a description of the backup operation. This description contains identification of the tape drives, the tape volumes and their contents. The /JOURNAL qualifier directs the DBO/RESTORE/MULTITHREADED command to read the journal file and select only the useful tape volumes.

The journal file must be the one created at the time the backup operation was performed. If the wrong journal file is supplied, DBO returns an informational message and does not use the specified journal file to select the volumes to be processed.

If you omit the /LABEL qualifier, the restore operation creates a list of volume labels from the contents of the journal file.

A by-area restore operation also constructs a list of useful tape volume labels from the journal file; only those volumes are mounted and processed.

/LABEL=(label ([,...]))
Specifies the 1- to 6-character string with which the volumes of the backup file are labeled. You can specify a list of tape labels for multiple tape volumes.

This qualifier can be used only with the /REWIND qualifier for backup to tape.

If the volume label disagrees with the label specified, you receive an error message. If the tape is not the first volume or if you specified the /REWIND qualifier in the command, DBO checks the owner, file protection, and expiration date of the tape and issues an error message if a problem was found.

If any of these checks fails, you are asked what you want to do with the tape volume. You must select one of the following options:

- QUIT

  This option cancels the operation and causes you to exit from DBO.

- RETRY

  This option repeats the checks after dismounting and remounting the same tape.

- UNLOAD

  This option dismounts and unloads the tape so you can load the correct tape on the drive.

- OVERRIDE

  This option overrides the failed checks and uses the tape only if you have read access to the tape.

/LOADER_SYNCHRONIZATION
Allows you to preload tapes in order to minimize operator support.

The DBO/BACKUP/MULTITHREAD and DBO/RESTORE/MULTITHREAD commands offer concurrent tape drive operation. However, maximal performance is achieved at the cost of not being able to preload the drive loader or stacker and have DBO use the tapes in the order loaded. The trade-off must be made between operator support time and tape I/O time.

If DBO is only writing to or reading from one drive at a time, preloading is not a problem. But if more than one drive is being used concurrently, DBO must use the tapes in the order specified by the /LABEL qualifier. However, for maximal performance, no drive should remain idle, and the next identified volume must be placed on the first drive to become idle. Because the order in which the drives become idle depends on many uncontrollable factors and cannot be predetermined, the drives cannot be preloaded with tapes.

When you specify the /LOADER_SYNCHRONIZATION qualifier, the restore operation waits until all concurrent tape operations have concluded before assigning the next set of tape volumes. This ensures that the tapes can be loaded into the loaders or stackers in the order required by the restore operation.

This mode of operation has a cost in reduced performance, but because the actual cost is very sensitive to the hardware configuration and the system load, it is unpredictable. A 5% to 20% additional elapsed time for the operation is typical. You must determine whether the benefit of a lower level of operator support compensates for this loss of performance. The /LOADER_SYNCHRONIZATION qualifier is most useful for use with large backup operations.

The /LOADER_SYNCHRONIZATION qualifier has no effect unless the /VOLUMES qualifier is also specified.

/LOCAL_BUFFERS=local-buffer-options

Allows you to change the default local buffer parameters when you restore a database. The following options are available:

- NUMBER=number-buffers

  Use this option to specify the number of local buffers that will be available for all users. You must specify a number between 2 and 32,767 for the number-buffers parameter.

- SIZE=buffer-blocks

  The size (specified in blocks) for each buffer. You must specify a number between 2 and 128 for the buffer-blocks parameter.

  If you specify a value smaller than the size of the largest page defined, DBO will automatically adjust the size of the buffer to hold the largest page defined. For example, if you specify /LOCAL_BUFFERS=SIZE=8 and the largest page size for the storage areas in your database is 64 blocks, DBO will automatically interpret the /LOCAL_BUFFERS=SIZE=8 as though it were /LOCAL_BUFFERS=SIZE=64.

  The value specified for the SIZE parameter determines the number of blocks for each buffer, regardless of whether local buffering or global buffering is enabled for the database.

If you do not specify a /LOCAL_BUFFERS option, the database will be restored with the values that were in effect when the database was backed up.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request log output and /NOLOG to prevent it. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/MASTER
Allows you to explicitly state how drives should be used when they are to be accessed concurrently. This is is a positional qualifier that designates a tape drive as a master tape drive. It can only be used on the backup file.

When the /MASTER qualifier is used, it must be used on the first drive specified. All additional drives become slaves to that master until the end of the command line, or until the next /MASTER qualifier, whichever comes first.

If the /MASTER qualifier is used on a drive that does not have an independent I/O path (not a hardware master), performance will decrease.

If the /MASTER qualifier is not used, and concurrent tape access is requested, DBO will use the same automatic configuration procedure it employs with the backup operation to select the master drives.

Use of the /MASTER qualifier is an error if concurrent tape access is not specified. See the /VOLUMES qualifier for further information on specifying concurrent tape access.

/MEDIA_LOADER
/NOMEDIA_LOADER
Specifies that the tape device from which DBO is reading the backup file has a loader or stacker. Use the /NOMEDIA_LOADER qualifier to specify that the tape device does not have a loader or stacker.

By default, if a tape device has a loader or stacker, DBO should detect it. However, occasionally DBO does not detect that a tape device has a loader or stacker. Therefore, after reading the first tape, DBO issues a request to the operator for the next tape, instead of requesting the next tape from the loader or stacker. Similarly, sometimes DBO behaves as though a tape device has a loader or stacker when actually it does not.

If you find that DBO is not detecting that your tape device has a loader or stacker, specify the /MEDIA_LOADER qualifier. If you find that DBO expects a loader or stacker when it should not, specify the /NOMEDIA_LOADER qualifier.

/NEW_VERSION
/NONEW_VERSION
Specifies whether or not new versions of database files should be produced if the device contains previous versions of database files.

If you use /NEW_VERSION, the new versions are produced. The default is /NONEW_VERSION. If you use /NONEW_VERSION, an error occurs if an old copy exists of any of the database files being restored.

/ONLINE
/NOONLINE
Specifies that the restore operation be performed while other users are bound to the database. The /ONLINE qualifier can be specified only with the /AREA or /JUST_CORRUPT qualifier. The pages to be restored are locked for exclusive

access, so the operation is not compatible with any other use of the data in the specified pages.

The default is the /NOONLINE qualifier.

/OPEN_MODE=option

Allows you to change the mode for opening a database when you restore that database. The following options are available:

- AUTOMATIC

  Allows users to invoke the database immediately after it is restored.

- MANUAL

  Specifies that a DBO/OPEN command must be used to open the database before users can invoke the database.

If you do not specify the /OPEN_MODE qualifier, the database will be restored with the open mode of the database that was in effect when the database was backed up.

/OPTIONS=file-spec

Contains storage area names, followed by the storage area qualifiers that you want applied to that storage area. Do not separate the storage area names with commas. Instead, put each storage area name on its own line in the file. The storage area qualifiers that you can include in the options file are /BLOCKS_PER_PAGE, /FILE, /SNAPSHOT, and /THRESHOLDS. You can use the DCL line continuation character, a hyphen (-), or the comment character (!) in the options file. The default file type is .OPT.

In the following example, the options file requests that the BUY.DBS, MAKE.DBS, and MARKET storage area files and the BUY.SNP, MAKE.SNP, and MARKET.SNP snapshot files be restored in the USER3:[CORPORATE] directory. Other database storage area and snapshot files not specified in the options file are restored to the disk and directory specified in the backup file, PARTS:

```
$ SET DEFAULT USER1:[SUPERS]
$ TYPE OPTIONS_FILE.OPT
$ ! Temporary options file for restore
BUY /FILE=USER3:[CORPORATE]BUY.DBS -
           /SNAPSHOT=FILE=USER3:[CORPORATE]BUY.SNP
MAKE /FILE=USER3:[CORPORATE]MAKE.DBS -
           /SNAPSHOT=FILE=USER3:[CORPORATE]MAKE.SNP
MARKET /FILE=USER3:[CORPORATE]MARKET.DBS -
           /SNAPSHOT=FILE=USER3:[CORPORATE]MARKET.SNP
$ DBO/RESTORE/MULTITHREAD/NOCDD_INTEGRATE/OPTIONS=OPTIONS_FILE.OPT PARTS
```

/PAGE_BUFFERS=n

Specifies the maximum number of buffers Oracle CODASYL DBMS will use during the DBO/RESTORE/MULTITHREAD operation while the database files are being created. The value of the /PAGE_BUFFERS qualifier may range from 1 to 5. The default is 3 buffers. Values larger than 3 buffers may improve performance, especially during incremental restore operations. When the DBO/RESTORE/MULTITHREAD command enters the stage of reconstructing internal structures at the end of the restore operation, a high /PAGE_BUFFERS value can be useful for very large databases.

However, the cost of using these extra buffers is that memory use is high. Thus, the trade-off during a restore operation is memory use against performance. For systems where physical memory is not a limitation, try setting the /PAGE_BUFFERS value higher during restore operations.

/PATH=cdd-repository-path
Specifies an Oracle CDD/Repository location containing the schema with which the database will be integrated. If you do not specify /PATH, DBO uses the schema pathname in the root file.

The /PATH qualifier is ignored if you use the /NOCDD_INTEGRATE qualifier to prevent attempts to integrate the database into Oracle CDD/Repository.

/RECOVERY[=(AIJ_BUFFERS=integer)
/NORECOVERY
Specifies that Oracle CODASYL DBMS perform a multithreaded restore operation and AIJ rollforward in a single operation. The optional AIJ_BUFFERS=integer argument specifies the number of buffers to be used by the rollforward process.

The DBO/RESTORE/MULTITHREAD command assumes that automatic recovery is desired. That is, if your restore operation includes AIJ information that was not captured in a previous backup, DBO/RESTORE performs automatic AIJ recovery. If your restore operation does not include such AIJ information, DBO/RESTORE/MULTITHREAD does not perform automatic AIJ recovery. Because automatic recovery is assumed, you must specify /NORECOVERY if you plan to restore an incremental backup.

For example, the following command specifies that the database is to be restored from the full database backup file PARTS.DBF and that the database is to be automatically recovered using the restored AIJ information, if possible. The AIJ restore process will use 10 buffers.

```
$ DBO/RESTORE/MULTITHREAD/RECOVERY=(AIJ_BUFFERS=10) PARTS.DBF
```

/REWIND
/NOREWIND
Specifies that the tape is to be rewound before processing begins.

Specify the /NOREWIND qualifier to have the restore procedure start looking for the backup file at the present position on the tape. This is the default.

/ROOT=root-file-spec
Specifies the root file specification of a restored database. This gives the database a new location, a new name, or both. The default file type is .ROO.

DBO derives the file specification for the restored root file by starting with the /ROOT qualifier, then taking missing file specification components from the /DIRECTORY qualifier, and ultimately taking any other remaining file specification components from the original root file specification.

/USERS=integer
Defines the number of users who can access the database simultaneously. Users include DBO commands. The maximum number of users is 2032. If omitted, the backed up value is used. Setting this value unnecessarily high causes performance to degrade and the root file to be very large. The minimum is 10.

/VOLUMES=n
Allows you to specify that concurrent tape access is to be used to accelerate the restore operation.

The /VOLUMES qualifier indicates concurrent tape access and specifies the number of tape volumes in the backup file. The number of volumes must be specified accurately for the restore operation to complete.

If the /VOLUMES qualifier is not specified, the restore operation does not use concurrent tape access.

Command or Area Qualifiers

/BLOCKS_PER_PAGE=integer
/NOBLOCKS_PER_PAGE
Allows you to restore the database with larger page sizes than existed in the original database. This creates new free space on each page in the area and does not interfere with record clustering. DBO ignores this qualifier when it specifies an integer less than or equal to the current page size of the area.

If you use /NOBLOCKS_PER_PAGE, the default, DBO takes the page size for the area from the page size specified for the database you backed up.

/FILE=file-spec
Assigns a new file specification to the area name parameter it qualifies. It can specify a fully qualified file specification and can be used to restore all or selected database files in a different disk or directory.

Global use of /FILE is a convenient way to shorten the command string by specifying a default device and directory for the restored database storage files. The value of the global file specification becomes the default specification for all storage area files, including areas not specified. Any part of a complete file specification omitted from a local /FILE qualifier on an area parameter defaults to the value given in the global /FILE qualifier.

If you do not specify a database file name for a storage area, the file name in the backup file is the default. Similarly, if you specify no directory (using either a global or local /FILE qualifier), the directory in the backup file is the default.

/JUST_CORRUPT
Restores all corrupt storage areas and pages logged in the Corrupt Page Table (CPT), if they are present in the backup file. In addition, if any snapshot files associated with the specified areas are marked corrupt, they are also cleared from the CPT.

Corrupt and inaccessible areas and pages are logged in the CPT, which can be seen in the output of the DBO/DUMP command. You can perform this type of restore operation on line.

The following example uses /JUST_CORRUPT as a command qualifier to restore, on line, all storage areas and pages logged corrupt in the CPT:

```
$ DBO/RESTORE/MULTITHREAD/JUST_CORRUPT/ONLINE PARTS.DBF
```

The backup file used for this type of restore operation must be a full database backup or a full area backup. It cannot be an incremental backup. SPAM pages cannot be restored because they are not backed up.

After Oracle CODASYL DBMS restores corrupt areas and pages, it changes their status in the CPT from corrupt to inconsistent. Inconsistent pages are still not accessible, so you must recover them to make them consistent. You can use the AIJ recovery utility with the /JUST_CORRUPT qualifier to recover the inconsistent areas and pages and make them consistent and available to database users.

The /JUST_CORRUPT qualifier can also be used as an area qualifier to restore a specific area. All corrupt pages, or the entire area if numerous pages are corrupt, are restored during a DBO/RESTORE/MULTITHREAD operation for the specified area. The /AREA qualifier is required when using the /JUST_CORRUPT qualifier as an area qualifier.

The following example uses /JUST_CORRUPT as an area qualifier. In this example, the command restores all pages from the area MAKE that are logged corrupt in the CPT from the parts.dbb backup file:

```
$ DBO/RESTORE/MULTITHREAD/AREA PARTS.DBF MAKE/JUST_CORRUPT
```

/SNAPSHOTS=option

Assigns a file specification to a snapshot file associated with the area name parameter it qualifies. In general, use this qualifier to place snapshot files on devices apart from the live-page files. The valid options are:

- FILE=file-spec

  Specifies any part of the full file-spec for a restored snapshot file

- ALLOCATION=integer

  Specifies a new allocation for a restored snapshot file

The /SNAPSHOTS qualifier is positional. If you use it globally, DBO restores all snapshot area files on the device and directory named in the FILE=file-spec argument, except those areas specified with a local /SNAPSHOTS qualifier naming a different device and directory. A local /SNAPSHOTS qualifier overrides a global /SNAPSHOTS qualifier for a particular area.

When you do not specify /SNAPSHOTS, the /FILE qualifier (if specified) determines the placement of snapshot files. In this case, DBO places the snapshot file in the same device and directory as its corresponding area file. If you omit both qualifiers, DBO restores snapshot areas according to the information stored in the backup file.

If you specify /SNAPSHOTS for a database or area that was not created with snapshots, DBO ignores the qualifier and no error message occurs.

Note

You cannot add snapshots using a multithreaded restore operation; although, you can add snapshots through a single-threaded restore operation.

/THRESHOLDS=(pct1[,pct2[,pct3]])

Changes a storage area's percentage fullness threshold. You can adjust space area management thresholds to improve future space usage in the storage area. Each threshold value represents a percentage of fullness on a data page. When a data page reaches the percentage of fullness defined by a given threshold value, the SPAM entry for the data page is updated to contain that threshold value.

A record occupies a certain percentage of a database page, DBCS examines the threshold values to determine which pages have room for the new record. If you do not use /THRESHOLDS, DBO uses the storage area's original thresholds. DBO ignores this qualifier if the area does not have space area management pages. The *Oracle CODASYL DBMS Database Design Guide* contains tutorial information on selecting threshold values.

Examples

1.
```
$ MOUNT/FOREIGN $111$MUA2:
%MOUNT-I-MOUNTED, PARTS mounted on _$111$MUA2: (TREK)
$ DBO/RESTORE/MULTI/LOG $111$MUA2:PARTS
%DBO-I-RESTXT_04, Thread 1 uses devices $111$MUA2:
%DBO-I-RESTXT_00, Restored root file DISK1:[PARTS]PARTS.ROO;1
%DBO-I-LOGMODDDL,  restored schema DBM$DISK:[PARTS_CDD]PARTS
%DBO-I-LOGMODDDL,  restored storage schema DEFAULT_STORAGE_SCHEMA
%DBO-I-LOGMODDDL,  restored subschema DEFAULT_SUBSCHEMA

%DBO-I-LOGMODDDL,  restored security schema DEFAULT_SECURITY_SCHEMA
%DBO-I-LOGRESSST, restored storage area DISK1:[PARTS]MAKE.DBS;1
%DBO-I-LOGRESSST, restored storage area DISK1:[PARTS]BUY.DBS;1
%DBO-I-LOGRESSST, restored storage area DISK1:[PARTS]MARKET.DBS;1
%DBO-I-LOGRESSST, restored storage area DISK1:[PARTS]PERSONNEL.DBS;1
%DBO-I-LOGRESSST, restored storage area DISK1:[PARTS]MAKE.DBS;1
%DBO-I-RESTXT_05,     rebuilt 1 space management page
%DBO-I-RESTXT_08,     restored 100 data pages
%DBO-I-LOGRESSST, restored storage area DISK1:[PARTS]BUY.DBS;1
%DBO-I-RESTXT_05,     rebuilt 1 space management page
%DBO-I-RESTXT_08,     restored 100 data pages

%DBO-I-LOGRESSST, restored storage area DISK1:[PARTS]MARKET.DBS;1
%DBO-I-RESTXT_05,     rebuilt 1 space management page
%DBO-I-RESTXT_08,     restored 100 data pages
%DBO-I-LOGRESSST, restored storage area DISK1:[PARTS]PERSONNEL.DBS;1
%DBO-I-RESTXT_05,     rebuilt 1 space management page
%DBO-I-RESTXT_08,     restored 100 data pages
%DBO-I-RESTXT_01, Initialized snapshot file DISK1:[PARTS]MAKE.SNP;1
%DBO-I-LOGINIFIL,     contains 2 pages, each page is 2 blocks long
%DBO-I-RESTXT_01, Initialized snapshot file DISK1:[PARTS]BUY.SNP;1
%DBO-I-LOGINIFIL,     contains 2 pages, each page is 2 blocks long
%DBO-I-RESTXT_01, Initialized snapshot file DISK1:[PARTS]MARKET.SNP;1
%DBO-I-LOGINIFIL,     contains 2 pages, each page is 2 blocks long

%DBO-I-RESTXT_01, Initialized snapshot file DISK1:[PARTS]PERSONNEL.SNP;1
%DBO-I-LOGINIFIL,     contains 2 pages, each page is 2 blocks long
%DBO-I-AIJNOTON, AIJ journalling was not active when database
  was backed up
%DBO-I-AIJRECOVR, AIJ recovery should start with file sequence 0
%DBO-I-LOGINTDDL, integrated root file DISK1:[PARTS]PARTS.ROO;1
%DBO-I-LOGINTDDL, integrated schema PARTS
%DBO-I-LOGINTDDL, integrated storage schema DEFAULT_STORAGE_SCHEMA
%DBO-I-LOGINTDDL, integrated subschema DEFAULT_SUBSCHEMA
%DBO-I-LOGINTDDL, integrated security schema DEFAULT_SECURITY_SCHEMA
```

This example restores the PARTS database using a multithread restore operation.

2.
```
$ DBO /RESTORE /RECOVERY /NOCDD/LOG/DIR=TEST$DATABASE TEST.DBB
%DBO-I-LOGOPNDBB, opened backup file KODD$:[JONES.WORK.AIJ]MUT.DBB;1
%DBO-I-LOGRESROO, restoring root file KODD$:[JONES.WORK.AIJ]MUT.ROO;1
%DBO-I-LOGMODDDL,     restored schema _CDD$TOP.SMITH.MUT.MUT
%DBO-I-LOGMODDDL,     restored storage schema DEFAULT_STORAGE_SCHEMA
%DBO-I-LOGMODDDL,     restored subschema DEFAULT_SUBSCHEMA
%DBO-I-LOGMODDDL,     restored security schema DEFAULT_SECURITY_SCHEMA
%DBO-I-LOGRESDBS, restoring storage area
     KODD$:[JONES.WORK.AIJ]MAINAREA.DBS;1
%DBO-I-LOGRESPAG,     restored 100 data pages, each page
  is 2 blocks long
%DBO-I-LOGRESSPM,     restored 1 spam page, each page
  is 2 blocks long
%DBO-I-LOGINISNP, initialized snapshot file
     KODD$:[JONES.WORK.AIJ]MAINAREA.SNP;1
%DBO-I-LOGINIFIL,     contains 2 pages, each page
  is 2 blocks long
%DBO-I-LOGRESDBS, restoring storage area
     KODD$:[JONES.WORK.AIJ]OTHERAREA.DBS;1
%DBO-I-LOGRESPAG,     restored 100 data pages, each page
  is 2 blocks long
%DBO-I-LOGRESSPM,     restored 1 spam page, each page
  is 2 blocks long
%DBO-I-LOGINISNP, initialized snapshot file
     KODD$:[JONES.WORK.AIJ]OTHERAREA.SNP;1
%DBO-I-LOGINIFIL,     contains 2 pages, each page is 2 blocks long
%DBO-I-AIJRSTBEG, Restoring after-image journal state information
%DBO-I-AIJRSTJRN, Restoring journal "MUT0" information
%DBO-I-AIJRSTSEQ, Journal sequence number is "1"
%DBO-I-AIJRSTSUC, Journal "MUT0" successfully restored from file
     "$111$DUA29:[JONES.WORK.AIJ]MUT0.AIJ;1"
%DBO-I-AIJRSTJRN, Restoring journal "MUT1" information
%DBO-I-AIJRSTSEQ, Journal sequence number is "5"
%DBO-I-AIJRSTSUC, Journal "MUT1" successfully restored from file
     "$111$DUA29:[JONES.WORK.AIJ]MUT1.AIJ;1"
%DBO-I-AIJRSTJRN, Restoring journal "MUT2" information
%DBO-I-AIJRSTSEQ, Journal sequence number is "4"
%DBO-I-AIJRSTSUC, Journal "MUT2" successfully restored from file
     "$111$DUA29:[JONES.WORK.AIJ]MUT2.AIJ;1
%DBO-I-AIJRSTJRN, Restoring journal "MUT3" information
%DBO-I-AIJRSTSEQ, Journal sequence number is "3"
%DBO-I-AIJRSTSUC, Journal "MUT3" successfully restored from file
     "$111$DUA29:[JONES.WORK.AIJ]MUT3.AIJ;1
%DBO-I-AIJRSTJRN, Restoring journal "MUT4" information
%DBO-I-AIJRSTSEQ, Journal sequence number is "2"
%DBO-I-AIJRSTSUC, Journal "MUT4" successfully restored from file
     "$111$DUA29:[JONES.WORK.AIJ]MUT4.AIJ;1
%DBO-I-AIJRSTEND, after-image journal state restoration complete
%DBO-I-AIJWASON, AIJ journalling was active when database was
   backed up
%DBO-I-AIJRECFUL, full database recovery starts with AIJ
     file sequence 1
%DBO-F-AIJENBOVR, enabling AIJ journaling would overwrite
   an existing
     journal
%DBO-I-LOGMODFLG,     disabled after-image journaling
%DBO-I-AIJRSTEND, after-image journal state restoration
  complete
%DBO-I-LOGRECDB, recovering database file
  KODD$:[JONES.WORK.AIJ]MUT.ROO;1
%DBO-I-AIJAUTOREC, starting automatic after-image journal recovery
%DBO-I-LOGOPNAIJ, opened journal file
  $111$DUA29:[JONES.WORK.AIJ]MUT0.AIJ;1
%DBO-I-AIJONEDONE, AIJ file sequence 1 roll-forward
  operations completed
```

```
%DBO-I-LOGRECOVR, 0 transactions committed
%DBO-I-LOGRECOVR, 0 transactions rolled back
%DBO-I-LOGRECOVR, 4 transactions ignored
%DBO-I-AIJNOACTIVE, there are no active transactions
%DBO-I-AIJSUCCES, database recovery completed successfully
%DBO-I-AIJNXTSEQ, to continue this AIJ file recovery, the sequence
  number needed will be 2
%DBO-I-LOGOPNAIJ, opened journal file
  $111$DUA29:[JONES.WORK.AIJ]MUT4.AIJ;1
%DBO-I-AIJONEDONE, AIJ file sequence 2 roll-forward operations
  completed
%DBO-I-LOGRECOVR, 0 transactions committed
%DBO-I-LOGRECOVR, 0 transactions rolled back
%DBO-I-LOGRECOVR, 0 transactions ignored
%DBO-I-AIJNOACTIVE, there are no active transactions
%DBO-I-AIJSUCCES, database recovery completed successfully
%DBO-I-AIJNXTSEQ, to continue this AIJ file recovery, the
  sequence number needed will be 3
%DBO-I-LOGOPNAIJ, opened journal file
  $111$DUA29:[JONES.WORK.AIJ]MUT3.AIJ;1
%DBO-I-AIJONEDONE, AIJ file sequence 3 roll-forward
  operations completed
%DBO-I-LOGRECOVR, 0 transactions committed
%DBO-I-LOGRECOVR, 0 transactions rolled back
%DBO-I-LOGRECOVR, 0 transactions ignored
%DBO-I-AIJNOACTIVE, there are no active transactions
%DBO-I-AIJSUCCES, database recovery completed successfully
%DBO-I-AIJNXTSEQ, to continue this AIJ file recovery, the sequence
  number needed will be 4
%DBO-I-LOGOPNAIJ, opened journal file
  $111$DUA29:[JONES.WORK.AIJ]MUT2.AIJ;1
%DBO-I-AIJONEDONE, AIJ file sequence 4 roll-forward
  operations completed
%DBO-I-LOGRECOVR, 0 transactions committed
%DBO-I-LOGRECOVR, 0 transactions rolled back
%DBO-I-LOGRECOVR, 0 transactions ignored
%DBO-I-AIJNOACTIVE, there are no active transactions
%DBO-I-AIJSUCCES, database recovery completed successfully
%DBO-I-AIJNXTSEQ, to continue this AIJ file recovery, the sequence
  number needed will be 5
%DBO-I-LOGOPNAIJ, opened journal file
  $111$DUA29:[JONES.WORK.AIJ]MUT1.AIJ;1
%DBO-W-AIJDEVDIR, AIJ filename "MUT5.AIJ" does not include device
  and directory
%DBO-W-AIJDEVDIR, AIJ filename "MUT6.AIJ" does not include device
  and directory
%DBO-I-AIJONEDONE, AIJ file sequence 5 roll-forward operations
  completed
%DBO-I-LOGRECOVR, 5 transactions committed
%DBO-I-LOGRECOVR, 0 transactions rolled back
%DBO-I-LOGRECOVR, 0 transactions ignored
%DBO-I-AIJNOACTIVE, there are no active transactions
%DBO-I-AIJSUCCES, database recovery completed successfully
%DBO-I-AIJNXTSEQ, to continue this AIJ file recovery, the sequence
  number needed will be 6
%DBO-I-AIJALLDONE, after-image journal roll-forward operations
  completed
%DBO-I-LOGSUMMARY, total 5 transactions committed
%DBO-I-LOGSUMMARY, total 0 transactions rolled back
%DBO-I-LOGSUMMARY, total 4 transactions ignored
%DBO-I-AIJSUCCES, database recovery completed successfully
%DBO-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence
  number needed will be 6
%DBO-I-AIJNOENABLED, after-image journaling has not yet been enabled
```

The command in this example performs an automatic AIJ restore and

recovery operation. It shows the full database restore operation of a database containing several .AIJ files, and the automatic recovery of all applicable after-image journal files.

3.
```
$ DBO/RESTORE/MULTI PARTS.DBF/AREA MAKE/JUST_CORRUPT
%DBO-I-RESTXT_04, Thread 1 uses devices DISK1:
%DBO-I-LOGRESSST, restored storage area DISK1:[DBUSER]MAKE.DBS;1
%DBO-I-LOGRESSST, restored storage area DISK1:[DBUSER]MAKE.DBS;1
%DBO-I-RESTXT_05,     rebuilt 1 space management page
%DBO-I-RESTXT_08,     restored 0 data pages
%DBO-I-AIJWASON, AIJ journalling was active when database was
  backed up
%DBO-I-AIJRECFUL, full database recovery starts with AIJ file
  sequence 0
%DBO-I-AIJBADPAGE, inconsistent page 1 from storage area
  DISK1:[DBUSER]MAKE.DBS;1 needs AIJ sequence number 0
%DBO-I-LOGRECDB, recovering database file DISK1:[DBUSER]PARTS.ROO;1
%DBO-I-AIJAUTOREC, starting automatic after-image journal recovery
%DBO-I-LOGOPNAIJ, opened journal file DISK1:[DBUSER]PARTS.AIJ;1
%DBO-I-AIJONEDONE, AIJ file sequence 0 roll-forward operations
  completed
%DBO-I-LOGRECOVR, 3 transactions committed
%DBO-I-LOGRECOVR, 0 transactions rolled back
%DBO-I-LOGRECOVR, 0 transactions ignored
%DBO-I-AIJNOACTIVE, there are no active transactions
%DBO-I-AIJSUCCES, database recovery completed successfully
%DBO-I-AIJALLDONE, after-image journal roll-forward operations
  completed
%DBO-I-LOGSUMMARY, total 3 transactions committed
%DBO-I-LOGSUMMARY, total 0 transactions rolled back
%DBO-I-LOGSUMMARY, total 0 transactions ignored
%DBO-I-AIJSUCCES, database recovery completed successfully
%DBO-I-AIJGOODPAGE, page 1 from storage area DISK:[DBUSER]MAKE.DBS;1
 is now consistent
%DBO-I-AIJFNLSEQ, to start another AIJ file recovery, the sequence
 number needed will be 0
```

This example performs a /JUST_CORRUPT restore operation on the MAKE area which was found to have corrupt pages. Because after-image journaling was enabled, DBO attempts and succeeds at applying an automatic recovery.

9.31 DBO/SERVER Commands

There are five forms of the DBO/SERVER command, each having its own qualifiers. These commands are:

DBO/SERVER AFTER_JOURNAL REOPEN_OUTPUT root-file-spec [,...]

This format allows you to close and reopen the output file specified with a DBO/SERVER AFTER_JOURNAL START command.

DBO/SERVER AFTER_JOURNAL START root-file-spec [,...]

This format allows you to start the AIJ log server.

DBO/SERVER AFTER_JOURNAL STOP root-file-spec [,...]

This format allows you to stop the AIJ log server.

DBO/SERVER BACKUP_JOURNAL RESUME root-file-spec [,...]

This format allows you to resume AIJ backup operations that were suspended with the DBO/SERVER BACKUP_JOURNAL SUSPEND command.

DBO/SERVER BACKUP_JOURNAL SUSPEND root-file-spec [,...]

This format allows you to temporarily suspend AIJ backup operations.

These commands are described in the following sections.

9.31.1 DBO/SERVER AFTER_JOURNAL REOPEN_OUTPUT Command

Allows you to close the current AIJ log server (ALS) output file for the specified database and open a new one. This allows you to see the current contents of the original ALS output file.

Format

DBO/SERVER AFTER_JOURNAL REOPEN_OUTPUT root-file-spec [,...]

Description

The DBO/SERVER AFTER_JOURNAL REOPEN_OUTPUT command allows you to reopen an ALS output file that was previously created with a DBO/SERVER AFTER_JOURNAL START command with the /OUTPUT qualifier. (The ALS output file is opened for exclusive access by the ALS process.) If the /OUTPUT qualifier was not specified when the ALS was started, the logical name DBM$BIND_ALS_OUTPUT_FILE can be defined and the DBO/SERVER AFTER_JOURNAL REOPEN_OUTPUT command will create a new output file.

Reopening the output file results in the current output file being closed and a new output file being created. The new output file has the same file name as the original output file, but its version number is incremented by one.

The ALS is an optional process that transfers log data to the after-image journal (.AIJ) file. All database servers deposit transaction log data in a cache located in the database global section. If the ALS is active, it continuously flushes the log data to disk. Otherwise, server processes might block temporarily if the cache in the global section is full.

Command Parameters

root-file-spec
Specifies the database root file for which you want to reopen the ALS output file.

Example

```
$ DBO/OPEN PARTS
$ DBO/SERVER AFTER_JOURNAL START PARTS/OUT=ALS
$ ! Database updates occur
$ DBO/SERVER AFTER_JOURNAL REOPEN_OUTPUT PARTS
$ ! View the ALS.OUT;-1 file:
$ TYPE ALS.OUT;-1

 ------------------------------------------------------------------

  - 7-OCT-1996 11:01:30.41 -
  Oracle CODASYL DBMS V7.0 database utility started

 ------------------------------------------------------------------
```

In this example the first DBO command opens the database so that the ALS can be started. In the second command, the ALS is started and an output file is specified. The last DBO command reopens the ALS output file, so you can view the data that is contained in the ALS output file so far.

9.31.2 DBO/SERVER AFTER_JOURNAL START Command

Allows you to manually start the AIJ log server (ALS) for the specified database and specify a file for the AIJ log server output.

Format

DBO/SERVER AFTER_JOURNAL START root-file-spec [,...]

Command Qualifier	Default
/OUTPUT=file-spec	See description

Description

The AIJ Log Server (ALS) is designed to improve high performance system throughput by reducing server stalls, eliminating AIJ lock requests for servers, and eliminating the need to tune commit timers. ALS dedicates one process per database; the ALS process writes log data to the .AIJ file in a single I/O operation. Database servers place log data in a global section and the ALS writes the data to disk on behalf of all the servers. Use ALS only when you have determined that AIJ processing is causing a bottleneck.

When the ALS is not enabled, write requests to the .AIJ file are performed by obtaining a global AIJ lock; choosing a group committer from all servers; and having the group committer write AIJ data from all servers to disk.

Command Parameters

root-file-spec
Specifies the database root file for which you want to start the ALS.

Command Qualifiers

/OUTPUT=file-spec
Specifies the file for the ALS output file. Use this qualifier in anticipation of issuing a DBO/SERVER/AFTER_JOURNAL/REOPEN_OUTPUT command. By specifying the output file you will know the location of, and therefore can view, the ALS output file.

By default, the ALS output file is not available to the user.

Example

```
$ DBO/OPEN PARTS
$ DBO/SERVER AFTER_JOURNAL/OUTPUT=ALS.LOG START PARTS
$ DBO/SHOW USERS
Oracle CODASYL DBMS V7.0 on node MYNODE 7-OCT-1996 11:01:51.88
    - monitor started  6-OCT-1996 01:34:19.15
    - monitor log filename is "DISK1:[DBMMON_LOGS]DBMMON701_MYNODE.LOG;269"
    - 255 monitor buffers available (256 maximum)
database DISK1:[MYDB]PARTS.ROO;1
    - First opened  7-OCT-1996 11:01:23.42
    * database is opened by an operator
    - current after-image journal file is DISK2:[MYAIJ]PARTS.AIJ;1
    - AIJ Log Server is active
 . . .
```

This command opens the database, starts the AIJ log server and then issues a DBO/SHOW USERS command to verify that the AIJ log server is active.

9.31.3 DBO/SERVER AFTER_JOURNAL STOP Command

Allows you to manually stop the AIJ log server (ALS) for the specified database and specify a file for the AIJ log server output.

Format

DBO/SERVER AFTER_JOURNAL STOP root-file-spec [,...]

Command Qualifier	Default
/OUTPUT=file-spec	See description

Description

The AIJ log server (ALS) is designed to improve high performance system throughput by reducing server stalls, eliminating AIJ lock requests for servers, and eliminating the need to tune commit timers. ALS dedicates one process per database; the ALS process writes log data to the .AIJ file in a single I/O operation. Database servers place log data in a global section and the ALS writes the data to disk on behalf of all the servers. Use ALS only when you have determined that AIJ processing is causing a bottleneck.

When the ALS is not enabled, write requests to the .AIJ file are performed by obtaining a global AIJ lock; choosing a group committer from all servers; and having the group committer write AIJ data from all servers to disk.

Command Parameters

root-file-spec
Specifies the database root file for which you want to stop the ALS.

Command Qualifiers

/OUTPUT=file-spec
Specifies the file for the ALS output file. Use this qualifier in anticipation of issuing a DBO/SERVER/AFTER_JOURNAL/REOPEN_OUTPUT command. By specifying the output file you will know the location of, and therefore can view, the ALS output file.

By default, the ALS output file is not available to the user.

Example

```
$ DBO/SERVER AFTER_JOURNAL STOP PARTS
```

This command stops the AIJ log server.

9.31.4 DBO/SERVER BACKUP_JOURNAL RESUME Command

Allows you to reinstate the ability to perform AIJ backup operations after they have been manually suspended with the DBO/SERVER BACKUP_JOURNAL SUSPEND command.

Format

DBO/SERVER BACKUP_JOURNAL RESUME root-file-spec [,...]

Command Qualifier	Default
/[NO]LOG	/NOLOG

Description

When you issue the DBO/SERVER BACKUP_JOURNAL SUSPEND command, after-image journal (AIJ) backup operations are temporarily suspended. Use the DBO/SERVER BACKUP_JOURNAL RESUME command to reinstate the ability to back up .AIJ files.

The DBO/SERVER BACKUP_JOURNAL RESUME command must be issued from the same node from which AIJ backup operations were originally suspended. If you attempt to resume AIJ backup operations from another database node, the following errors are returned:

```
%DBM-F-CANTRESUMEABS, error resuming AIJ backup operations
-DBM-F-DBNOTACTIVE, database is not being used, or must be manually
  opened first
```

Note

If you are using the Hot Standby Option and suspend AIJ backup operations, starting master database replication on the node from which AIJ backup operations were suspended will automatically restart AIJ backup operations.

Command Parameters

root-file-spec
Specifies the database root file for which you want to resume AIJ backup operations.

Command Qualifier

/LOG
/NOLOG
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the /LOG qualifier to request log output and the /NOLOG qualifier to

prevent it. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

Example

```
$ DBO/SERVER BACKUP_JOURNAL RESUME PARTS
```

This example demonstrates how to reinstate the ability to perform backup operations previously suspended with the DBO/SERVER BACKUP_JOURNAL SUSPEND command.

9.31.5 DBO/SERVER BACKUP_JOURNAL SUSPEND Command

Allows you to temporarily suspend AIJ backup operations on all database nodes. While suspended, you cannot back up .AIJ files manually (with the DBO/BACKUP/AFTER_JOURNAL command) nor will the AIJ backup server (ABS) perform AIJ backup operations.

Format

DBO/SERVER BACKUP_JOURNAL SUSPEND root-file-spec [,...]

Command Qualifier	Default
/[NO]LOG	/NOLOG

Description

Suspending AIJ backup operations is temporary and nonpersistent. That is, the suspension is not stored in the database root file. If the node from which AIJ backup operations are suspended were to fail, then AIJ backup operations would be automatically resumed when the database was opened. This could be on node reboot, or upon opening the database on another node in a cluster.

This command temporarily suspends AIJ backup operations during the time when invoking the AIJ backup would prevent subsequent commands from operating properly. For example, if you are using the Hot Standby Option, AIJ backups would prevent the replication from starting.

The solution to this problem is to use the DBO/SERVER BACKUP_JOURNAL SUSPEND command to suspend AIJ backup operations before the database backup begins until after replication commences.

AIJ backup operations are suspended until any one of the following events occurs:

- The database is closed on the node from which AIJ backup operations were suspended.
- The node fails from which AIJ backup operations were suspended.
- Database replication is started as a master database on the node from which AIJ backup operations were suspended.
- AIJ backup operations are explicitly resumed on the node from which AIJ backup operations were suspended. (This occurs when you issue the DBO/SERVER BACKUP_JOURNAL RESUME command. See Section 9.31.4 for details.)

Output from the DBO/SHOW USERS command indicates if AIJ backup operations have been suspended.

Command Parameters

root-file-spec
Specifies the database root file for which you want to suspend AIJ backup operations.

Command Qualifier

/LOG
/NOLOG
Specifies whether the processing of the command is reported to SYS$OUTPUT. Specify the /LOG qualifier to request log output and the /NOLOG qualifier to prevent it. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

Example

```
$ DBO/SERVER BACKUP_JOURNAL SUSPEND PARTS
$ DBO/SHOW USERS PARTS
Oracle CODASYL DBMS V7.0 on node MYNODE 7-OCT-1996 13:11:32.08
 . . .
After-image backup operations temporarily suspended from this node
$ DBO/BACKUP/AFTER_JOURNAL PARTS PARTS.AIJ
%DBO-F-LCKCNFLCT, lock conflict on AIJ backup
```

In this example first AIJ backup operations are suspended, then the DBO/SHOW/USERS command is issued to confirm that suspension has occurred. If you attempt an AIJ backup operation, you receive the %DBO-F-LCKCNFLCT error message.

9.32 DBO/SHOW Command

Displays current information about version numbers, active databases, active users, active recovery units, monitor statistics, or visible OpenVMS locks related to database activity on your VMScluster node.

Format

DBO/SHOW option [root-file-spec]

Description

The DBO/SHOW command has eight output options: AFTER_JOURNAL, CORRUPT_PAGES, LOCKS, LOGICAL_NAMES, STATISTICS, SYSTEM, USERS, and VERSION. The option determines which command qualifiers apply and if you need to use a root file specification parameter.

The STATISTICS option includes syntax and output that are entirely different from that of the other options. Consequently, the various options of the DBO/SHOW command are described separately in the following sections.

The DBO/SHOW command cannot be secured with the DBO/GRANT_COMMAND command, although it may access a root file.

Parameters

option
Specifies the type of information to display. The options include:

- AFTER_JOURNAL
- CORRUPT_PAGES
- LOCKS
- LOGICAL_NAMES
- STATISTICS
- SYSTEM
- USERS
- VERSION

root-file-spec
Specifies the root file of the database on which you want information. This parameter is meaningful when you select AFTER_JOURNAL, CORRUPT_ PAGES, STATISTICS, USERS, or LOCKS as the option. The default file type is .ROO.

9.32.1 DBO/SHOW AFTER_JOURNAL Command

Displays the after-image journal configuration in the form required for the /AIJ_OPTIONS qualifier. You can use the /AIJ_OPTIONS qualifier with the DBO/COPY_DATABASE, DBO/MOVE_AREA, and the DBO/RESTORE/MULTITHREADED commands.

In addition, this command optionally initializes the following global process symbols:

- DBM$AIJ_BACKUP_SEQNO
- DBM$AIJ_COUNT
- DBM$AIJ_CURRENT_SEQNO
- DBM$AIJ_ENDOFFILE
- DBM$AIJ_FULLNESS
- DBM$AIJ_LAST_SEQNO
- DBM$AIJ_NEXT_SEQNO
- DBM$AIJ_SEQNO

Format

DBO/SHOW AFTER_JOURNAL root-file-spec

Command Qualifiers	Defaults
/[NO]BACKUP_CONTEXT	/BACKUP_CONTEXT
/OUTPUT[=file-spec]	SYS$OUTPUT

Description

The output of the DBO/SHOW AFTER_JOURNAL command displays the form required by the /AIJ_OPTIONS qualifier for the DBO/COPY_DATABASE, DBO/MOVE_AREA, and DBO/RESTORE/MULTITHREADED commands. When you issue the DBO/SHOW AFTER_JOURNAL command, you may not see all the options unless you specified them when you created your after-image journal file configuration (for example, with the DBO/MODIFY/AIJ_OPTIONS/JOURNAL_ OPTIONS command).

When you use the output from the DBO/SHOW AFTER_JOURNAL command as a template for the /AIJ_OPTIONS qualifier of the DBO/COPY_DATABASE, DBO/MOVE_AREA, and DBO/RESTORE/MULTITHREADED commands, note the following regarding the syntax:

- You can use the DCL continuation character (-) at the end of each line in the JOURNAL and ADD clauses. Although continuation characters are not required if you can fit each clause on a single line, using them might improve readability.
- The JOURNAL IS clause must precede the ADD clause.
- Because the JOURNAL clause and the ADD clause are two separate clauses, a continuation character should not be used between the last option in the JOURNAL clause and the ADD clause (or clauses).
- The journal options file can contain one JOURNAL clause only, but it can contain several ADD clauses. However, the number of ADD clauses cannot exceed the number of reservations made for .AIJ files. In addition, if you are enabling journaling, you must add at least one journal file.
- You can specify only one of each option (for example, one EXTENT clause, one CACHE clause, and so on) for the JOURNAL IS clause.

The clauses and options have the following meaning:

- JOURNAL IS ENABLED

  Enables after-image journaling. At least one ADD clause must follow. If this option is omitted, the current journaling state is maintained.

- JOURNAL IS DISABLED

  Disables after-image journaling. You can specify other options or ADD clauses but they do not take effect until journaling is enabled. The ADD clause is optional. If this option is omitted, the current journaling state is maintained.

- RESERVE n

  Allocates space for an .AIJ file name for a maximum of *n* .AIJ files. By default, no reservations are made.

- ALLOCATION IS n

  Specifies the size (in blocks) of each .AIJ file. If this option is omitted, the default allocation size is 512 blocks. The maximum allocation size you can specify is 8 million blocks.

- EXTENT IS n

Specifies the maximum size to extend an .AIJ file if it is, or becomes, an extensible .AIJ file (in blocks). (If the number of available after-image journal files falls to 1, extensible journaling is employed.)

If there is insufficient free space on the .AIJ file device, the file is extended using a smaller extension value than specified. However, the minimum, and default, extension size is 512 blocks.

- OVERWRITE IS ENABLED

  Enables writing over journal files before they have been backed up. If this option is omitted, writing over files is disabled.

  This option is ignored if only one .AIJ file is available. When you specify this option, it is activated only when two or more .AIJ files are, or become, available.

- OVERWRITE IS DISABLED

  Disables writing over journal files before they have been backed up. If this option is omitted, writing over files is disabled.

- SHUTDOWN_TIMEOUT IS n

  Sets the delay from the time a journal failure is detected until the time the database aborts all access and shuts itself down. The value *n* is in minutes.

  If this option is omitted, the shutdown timeout is 60 minutes. The maximum value you can specify is 4320 minutes.

- NOTIFY IS ENABLED

  Enables operator notification when the journal state changes. If this option is omitted, operator notification is disabled.

- NOTIFY IS DISABLED

  Disables operator notification when the journal state changes. If this option is omitted, operator notification is disabled.

- BACKUPS ARE MANUAL

  Requires manual backup operations; automatic backup operations are not enabled. This is the default behavior.

- BACKUPS ARE AUTOMATIC [file filename]

  Requires automatic backup operations. Automatic backup operations are triggered by the filling of a journal file. The backup file will have the specified file name unless a different file name or an edit string is specified in the ADD clause. If this option is omitted, backup operations are manual.

- EDIT STRING IS (edit-string-options)

  Specifies a default edit string to apply to the backup file when an .AIJ file is backed up automatically.

- QUIET_POINT

  Specifies that the after-image journal backup operation is to acquire the quiet-point lock prior to performing an .AIJ backup operation for the specified database.

- NOQUIET_POINT

  Specifies that the after-image journal backup operation will not acquire the quiet-point lock prior to performing an .AIJ backup operation for the specified database.

- CACHE IS ENABLED [file filename]

  Specifies that a journal cache file should be used. The cache file must reside on a nonvolatile, solid-state disk. If it does not, caching is ineffective. By default, caching is disabled.

- CACHE IS DISABLED

  Specifies that a journal cache file should not be used. This is the default behavior.

- ADD

  Specifies the name and location of the journal file and the backup file generated by automatic backup operations.

  The ADD clauses are as follows:

  – ADD [JOURNAL] journal-name

    Specifies the name for the after-image journal file described in the JOURNAL clause. The journal-name is the name of the journal object. A journal object is the journal file specification plus all the attributes (allocation, extent, and so on) given to it in the journal clause.

  – FILE file-specification

    Provides the full file specification and version number of the .AIJ file named in the ADD clause. This line of output is provided because the next line (file filename) provides the string that the user entered when he or she created the .AIJ file. For example, if the user entered a file name only, and this line of output was not provided, you would have to issue the DBO/DUMP command to determine in which directory the file resides.

  – FILE filename

    Specifies the file name for the .AIJ file being added. This option is required.

  – ALLOCATION IS n

    Specifies the size of the .AIJ file (in blocks). If this option is omitted, the default allocation size is 512 blocks.

  – BACKUP_FILE filename

    Specifies the backup file name for automatic backup operations. It is not valid to specify a BACKUP_FILE clause in the ADD clause if you have specified BACKUPS ARE MANUAL in the JOURNAL clause; DBO returns an error if you attempt to do so.

  – EDIT STRING IS (edit-string-options)

    Specifies an edit string to apply to the backup file when the .AIJ file is backed up automatically.

Parameters

root-file-spec
Specifies the root file of the database for which you want after-image journal information displayed. The default file type is .ROO.

Command Qualifiers

/BACKUP_CONTEXT
/NOBACKUP_CONTEXT

Specifies that the following symbols be initialized, unless you have issued a DCL SET SYMBOL/SCOPE=(NOLOCAL, NOGLOBAL) command:

- DBM$AIJ_SEQNO

  Contains the sequence number of the last .AIJ backup file written to tape. This symbol has a value identical to DBM$AIJ_BACKUP_SEQNO.

- DBM$AIJ_CURRENT_SEQNO

  Contains the sequence number of the currently active .AIJ file. A value of –1 indicates that after-image journaling is disabled.

- DBM$AIJ_NEXT_SEQNO

  Contains the sequence number of the next .AIJ file that needs to be backed up. This symbol always contains a positive integer value (which may be 0).

- DBM$AIJ_LAST_SEQNO

  Contains the sequence number of the last .AIJ file available for a backup operation, which is different from the current sequence number if fixed-size journaling is being used. A value of –1 indicates that no journal has ever been backed up.

  If the value of the DBM$AIJ_NEXT_SEQNO symbol is greater than the value of the DBM$AIJ_LAST_SEQNO symbol, then no more .AIJ files are currently available for the backup operation.

- DBM$AIJ_BACKUP_SEQNO

  Contains the sequence number of the last .AIJ file backed up (completed) by the backup operation. This symbol is set at the completion of an .AIJ backup operation. A value of –1 indicates that this process has not yet backed up an .AIJ file.

- DBM$AIJ_COUNT

  Contains the number of available .AIJ files.

- DBM$AIJ_ENDOFFILE

  Contains the end of file block number for the current .AIJ file.

- DBM$AIJ_FULLNESS

  Contains the percent fullness of the current .AIJ file.

The /NOBACKUP_CONTEXT qualifier specifies that the preceding symbols will not be initialized.

The /NOBACKUP_CONTEXT qualifier is the default.

These are string symbols, not integer symbols, even though their equivalence values are numbers. Therefore performing arithmetic operations with them produces unexpected results.

If you need to perform arithmetic operations with these symbols, first convert the string symbol values to numeric symbol values using the OpenVMS F$INTEGER lexical function. For example:

```
$ SEQNO_RANGE = F$INTEGER(DBM$AIJ_LAST_SEQNO) - F$INTEGER(DBM$AIJ_NEXT_SEQNO)
```

/OUTPUT[=file-spec]
Specifies the name of the file where output is sent. The default is SYS$OUTPUT. The default output file extension is .LIS, if you specify only a file name.

Examples

1.
```
$ DBO/SHOW AFTER_JOURNAL PARTS/BACKUP_CONTEXT
JOURNAL IS ENABLED -
    RESERVE 5 -
    ALLOCATION IS 512 -
    EXTENT IS 512 -
    OVERWRITE IS DISABLED -
    SHUTDOWN_TIMEOUT IS 60 -
    NOTIFY IS DISABLED -
    BACKUPS ARE MANUAL -
    CACHE IS DISABLED
ADD JOURNAL DBM$JOURNAL -
! FILE DISK:[USER]PARTS.AIJ;1
    FILE SYS$DISK:[]PARTS.AIJ
ADD JOURNAL AIJ_ONE -
! FILE DISK1:[USER]AIJ_ONE.AIJ;1
    FILE DISK1:[USER]AIJ_ONE
ADD JOURNAL AIJ_TWO -
! FILE DISK2:[USER]AIJ_TWO.AIJ;1
    FILE DISK2:[USER]AIJ_TWO
ADD JOURNAL AIJ_THREE -
! FILE DISK3:[USER]AIJ_THREE.AIJ;1
    FILE DISK3:[USER]AIJ_THREE
```

This command displays the current .AIJ file configuration and specifies that the AIJ symbols be initialized. You could copy and edit this .AIJ file configuration to use as the template for input to the /AIJ_OPTIONS qualifier of the DBO/COPY_DATABASE, DBO/MOVE_AREA, or the DBO/RESTORE/MULTITHREADED command.

2.
```
$ SHOW SYM *DBM$AIJ*

DBM$AIJ_COUNT == "4"          ! Number of available .AIJ files
DBM$AIJ_CURRENT_SEQNO == "0"  ! Sequence # of currently active AIJ
DBM$AIJ_ENDOFFILE == "10"     ! The EOF block # for current .AIJ file
DBM$AIJ_FULLNESS == "1"       ! Percent fullness of the current .AIJ file
DBM$AIJ_LAST_SEQNO == "-1"    ! No .AIJ file has ever been backed up
DBM$AIJ_NEXT_SEQNO == "0"     ! Sequence # of next .AIJ file needing backup
```

This command shows the settings of the symbols initialized in the previous example.

3.
```
DBO/BACKUP/AFTER_JOURNAL PART PARTS_BACK.AIJ
%DBO-I-AIJBCKBEG, beginning after-image journal backup operation
%DBO-I-OPERNOTIFY, system operator notification: Oracle CODASYL DBMS Database
DIS1:[USER]PARTS.ROO;1 Event Notification AIJ backup operation started

%DBO-I-AIJBCKSEQ, backing up after-image journal sequence number 0
%DBO-I-LOGBCKAIJ, backing up after-image journal DBM$JOURNAL at 10:43:51.74
%DBO-I-QUIETPT, waiting for database quiet point
%DBO-I-LOGCREBCK, created backup file DISK:[USER]PARTS_BACKUP.AIJ;1
%DBO-I-OPERNOTIFY, system operator notification: Oracle CODASYL DBMS Database
DISK:[USER]PARTS.ROO;1 Event Notification AIJ backup operation completed

%DBO-I-AIJBCKEND, after-image journal backup operation completed successfully
%DBO-I-LOGAIJJRN, backed up 1 after-image journal at 10:43:53.26
%DBO-I-LOGAIJBLK, backed up 254 after-image journal blocks at 10:43:53.26
```

```
SHOW SYM *DBM$AIJ*

DBM$AIJ_BACKUP_SEQNO == "0"   ! 0 is sequence # of last .AIJ file backed up
DBM$AIJ_COUNT == "4"          ! Number of available .AIJ files is still 4
DBM$AIJ_CURRENT_SEQNO == "1"  ! Current AIJ sequence # is now 1
DBM$AIJ_ENDOFFILE == "1"      ! The EOF block # is now 1
DBM$AIJ_FULLNESS == "0"       ! The .AIJ file is now empty
DBM$AIJ_LAST_SEQNO == "0"     ! Last AIJ available for backup now sequence # 1
DBM$AIJ_NEXT_SEQNO == "1"     ! Next .AIJ for backup now sequence # 1
DBM$AIJ_SEQNO == "0"          ! 0 is sequence # of last .AIJ file backed up
```

This command shows how the values for the AIJ symbols change after an .AIJ file has been backed up.

9.32.2 DBO/SHOW LOCKS Command

Displays information about the OpenVMS locks related to database activity on a node.

Format

DBO/SHOW LOCKS

Command Qualifiers	Defaults
/LOCK=(lock_id[,...])	
/MODE=(option)	
/OPTIONS=option	
/OUTPUT=file-spec	SYS$OUTPUT
/PROCESS=(process_id[,...])	

Description

The DBO/SHOW LOCKS command displays current information about lock activity and contention for all active databases on your node. This command uses the OpenVMS $GETLKI system service and is intended for diagnostic purposes. Executing DBO/SHOW LOCKS requires some overhead.

A process requesting a lock is in one of three states: owning, blocking, or waiting. A process is waiting when the lock request is incompatible with existing granted locks. A process is blocking when the lock request is granted and other waiting locks exist. A process owns a lock when the lock request is granted and there are no blocked or waiting lock requests. The DBO/SHOW LOCKS command provides information about processes that own, are blocking, or are waiting for locks.

In a VMScluster, the DBO/SHOW LOCKS command displays detailed information for the current node, although some general clusterwide information can be obtained.

Use the qualifiers individually or in combination to display the required output. See Table 9–5 for all possible qualifier combinations and the type of output they produce. If you do not specify any qualifiers a complete list of locks is displayed. The volume of information from this report can be quite large. Because this command runs in EXEC mode, you cannot terminate it by pressing Ctrl/C or Ctrl/Y. Therefore, you should use the /OUTPUT qualifier instead of SYS$OUTPUT to direct output to a file. Each output contains a heading that indicates what qualifiers, if any, were used to generate the output.

Table 9–5 Lock Qualifier Combinations

Object	/MODE Argument	/OPTION Argument	Output
/PROCESS			Locks for the specified processes
/PROCESS	BLOCKING		Processes blocking the specified processes
/PROCESS	WAITING		Processes waiting for the specified processes
/PROCESS		ALL	Process locks for the specified processes
/PROCESS		FULL	Special process locks for the specified processes
/PROCESS	BLOCKING, WAITING		Processes blocking and waiting for the specified processes
/PROCESS	BLOCKING	FULL	Special process locks blocking the specified processes
/PROCESS	WAITING	FULL	Special process locks waiting for the specified processes
/PROCESS	BLOCKING, WAITING	FULL	Special process locks blocking and waiting for the specified processes
/PROCESS		ALL, FULL	Process and special process locks for the specified processes
/LOCK			Locks for the specified locks
/LOCK	BLOCKING		Processes blocking the specified locks
/LOCK	WAITING		Processes waiting for the specified locks
/LOCK		FULL	Special process locks for the specified locks
/LOCK	BLOCKING	FULL	Special process locks blocking the specified locks
/LOCK	WAITING	FULL	Special process locks waiting for the specified locks
/LOCK	BLOCKING, WAITING		Processes blocking and waiting for the specified locks
/LOCK	BLOCKING, WAITING	FULL	Special process locks blocking and waiting for the specified locks
	BLOCKING		Lock requests that are blocked
	WAITING		Lock requests that are waiting
	BLOCKING, WAITING		Lock requests that are blocking and waiting
/PROCESS /LOCK			Locks for specified processes and locks
/PROCESS /LOCK	BLOCKING		Processes blocking the specified processes and locks
/PROCESS /LOCK	WAITING		Processes waiting for the specified processes and locks

(continued on next page)

Table 9–5 (Cont.) Lock Qualifier Combinations

Object	/MODE Argument	/OPTION Argument	Output
/PROCESS /LOCK	BLOCKING, WAITING		Processes blocking and waiting for the specified processes and locks
/PROCESS /LOCK	BLOCKING	FULL	Special process locks blocking the specified processes and locks
/PROCESS /LOCK	WAITING	FULL	Special process locks waiting for the specified processes and locks
/PROCESS /LOCK		ALL	Process locks for the specified processes and locks
/PROCESS /LOCK		FULL	Special process locks for the specified processes and locks
/PROCESS /LOCK	BLOCKING	FULL	Special process locks blocking the specified processes and locks
/PROCESS /LOCK		ALL, FULL	Process and special process locks for the specified processes and locks

By default, information is displayed only for processes for which you have access privileges. You need GROUP privilege to display information about other processes with the same group UIC number. You need WORLD privilege to display information about other processes in the system that do not have the same UIC number.

When you specify a list of process or lock identifiers, make sure the processes or locks are local to the node on which the command is issued. If there are no locks on your node, you will get the following message:

```
%DBO-I-NOLOCKSOUT, No locks on this node with the specified qualifiers.
```

See the *Oracle CODASYL DBMS Database Maintenance and Performance Guide* for information on how to interpret DBO/SHOW LOCKS output.

Command Qualifiers

/LOCK =(lock-id[,...])
Specifies one or more lock identifiers for which information is displayed. The lock identifier is an 8-digit hexadecimal number, and it must be local to the node on which the command is issued.

/MODE=(option[,...])
Specifies one or more lock modes to be displayed. Valid options include:

- BLOCKING

  Displays the processes whose locks are blocking the lock requests for other processes.

  The first line of output identifies a process that is waiting for a lock request to be granted. Subsequent lines of output identify those processes that are preventing the lock request from being granted. When multiple processes are waiting for the same lock resource, multiple sets of process-specific information, one for each waiting process, are displayed.

- WAITING

Displays the processes whose lock requests are waiting due to conflicting granted locks for other processes.

A requesting process can appear to be waiting for other lock requesters. This condition occurs when there are many processes waiting for the same lock resource. Depending upon the sequence of processes in the wait queue, certain waiting processes appear to be blocking other waiting processes because they will eventually be granted the lock first.

The first line of output identifies a process that has been granted a resource lock. Subsequent lines of output identify those processes that are waiting for lock requests on the same resource. When multiple processes are blocking the same lock resource, multiple sets of process-specific information, one for each blocking process, are displayed.

/OPTIONS=option

Specifies the detail of information to be displayed. Valid options include:

- ALL

  Indicates that information about all process locks be displayed. Information is displayed for all other processes that have a need to know the lock held by a specific process. This method is an easy way to display all of a process' locks and to see what other processes are also using the same resource. This option can only be specified when the /PROCESS qualifier is also specified and the /MODE qualifier is not specified.

- FULL

  Indicates that information about all process locks and special process locks, such as monitors, be displayed. By default, these special database processes are not displayed because they increase the size of the output.

  When the FULL option is specified, all application processes will be legitimately waiting on the special database processes. This is a perfectly normal condition and in no way indicates inappropriate operation of the database.

/OUTPUT=file-spec

Names the file where output is sent. The default file type is .LIS. The default output is SYS$OUTPUT.

/PROCESS=(process-id[,...])

Specifies one or more process identifiers about which information is displayed. The process identifier is an 8-digit hexadecimal number, and it must be local to the node on which the command is issued.

Examples

1.
```
$ DBO/SHOW LOCKS

=========================================================================
           DBO/SHOW LOCKS OUTPUT Information
=========================================================================

-------------------------------------------------------------------------
Resource Name: TSN block 11
Granted Lock Count: 10, Parent Lock ID: 05B402D5,  Lock Access Mode: 1,
Resource Type: Global,  Lock Value Block: 0
```

```
          -Master Node Info-  --Lock Mode Information--     -Remote Node Info-
ProcessID Lock ID   SystemID  Requested Granted   Queue     Lock ID   SystemID
20400151  00040B18  00010004  PR        PR        GRANT     00040B18  00010002
20800146  0008127A  00010004  PR        PR        GRANT     0008127A  00010004
2060012B  00041D69  00010004  PR        PR        GRANT     00041D69  00010003
20A0012E  00AC0007  00010004  PR        PR        GRANT     00AC0007  00010005
20200142  000C0C64  00010004  PR        PR        GRANT     000C0C64  00010001
2040011D  000809AD  00010004  NL        NL        GRANT     000809AD  00010002
2080011D  00080D63  00010004  NL        NL        GRANT     00080D63  00010004
2060011B  0008175C  00010004  NL        NL        GRANT     0008175C  00010003
20A0011C  00180867  00010004  NL        NL        GRANT     00180867  00010005
2020011C  000809B0  00010004  NL        NL        GRANT     000809B0  00010001
------------------------------------------------------------------------------
Resource Name: snapshot cursor 4
Granted Lock Count: 1,  Parent Lock ID: 12980DC8,   Lock Access Mode: 1,
Resource Type: Global,  Lock Value Block: 0

          -Master Node Info-  --Lock Mode Information--   -Remote Node Info-
ProcessID Lock ID   SystemID  Requested Granted   Queue   Lock ID   SystemID
20A00128  3648000A  00010005  NL        NL        GRANT   3648000A  00010005
------------------------------------------------------------------------------
Resource Name: client ..
Granted Lock Count: 5,  Parent Lock ID: 05B402D5,   Lock Access Mode: 1,
Resource Type: Global,  Lock Value Block: 0

          -Master Node Info-  --Lock Mode Information--     -Remote Node Info-
ProcessID Lock ID   SystemID  Requested Granted   Queue     Lock ID   SystemID
20400151  004C0C07  00010004  PR        PR        GRANT     004C0C07  00010002
20800146  00381384  00010004  PR        PR        GRANT     00381384  00010004
2060012B  00C81284  00010004  PR        PR        GRANT     00C81284  00010003
20A0012E  0090001F  00010004  PR        PR        GRANT     0090001F  00010005
20200142  00DC0BEE  00010004  PR        PR        GRANT     00DC0BEE  00010001
------------------------------------------------------------------------------
```

This example displays information about all locks in the OpenVMS locks database for the current node.

2.
```
$ DBO/SHOW LOCKS/OPTIONS=FULL
================================================================================
          DBO/SHOW LOCKS/OPTIONS=FULL Information
================================================================================

------------------------------------------------------------------------------
Resource Name: record 0:-4
Granted Lock Count: 1,  Parent Lock ID: 00080BC8,   Lock Access Mode: 1,
Resource Type: Global,  Lock Value Block: 0

          -Master Node Info-  --Lock Mode Information--     -Remote Node Info-
ProcessID Lock ID   SystemID  Requested Granted   Queue     Lock ID   SystemID
20400151  02C4001A  00010001  EX        EX        GRANT     02C4001A  00010002
------------------------------------------------------------------------------
Resource Name: TSN block 11
Granted Lock Count: 20, Parent Lock ID: 10DC0560,   Lock Access Mode: 1,
Resource Type: Global,  Lock Value Block: 0

          -Master Node Info-  --Lock Mode Information--     -Remote Node Info-
ProcessID Lock ID   SystemID  Requested Granted   Queue     Lock ID   SystemID
27A00054  000401B7  00010001  NL        NL        GRANT     000401B7  0001003D
27400054  000401F6  00010001  NL        NL        GRANT     000401F6  0001003A
```

This example displays information shown when using the /OPTIONS=FULL qualifier.

```
3.  $ DBO/SHOW LOCKS/MODE=WAITING
    ================================================================================
    SHOW LOCKS/WAITING Information
    ================================================================================

    --------------------------------------------------------------------------------
    Resource: storage area 1

              ProcessID Process Name        Lock ID   System ID Requested Granted
              --------- ---------------     --------- --------- --------- -------
    Blocker:  2100A311  "_TWA9:".........  18E90582  00010008  PW        PW
    Waiting:  2100B712  "_TWA7:".........  55383CA8  00010008  PW        NL
```

This example displays information shown when using the /MODE=WAITING qualifier.

```
4.  $ DBO/SHOW LOCKS/PROCESS=2100B712/MODE=BLOCKING
    ================================================================================
    SHOW LOCKS/PROCESS/BLOCKING Information
    ================================================================================

    --------------------------------------------------------------------------------
    Resource: storage area 1

              ProcessID Process Name        Lock ID   System ID Requested Granted
              --------- ---------------     --------- --------- --------- -------
    Waiting:  2100B712  "_TWA7:".........  55383CA8  00010008  PW        NL
    Blocker:  2100A311  "_TWA9:".........  18E90582  00010008  PW        PW
```

This example displays information shown when using the /PROCESS and /MODE=BLOCKING qualifiers.

9.32.3 DBO/SHOW LOGICAL_NAMES Command

Displays information about the OpenVMS logical names related to Oracle CODASYL DBMS.

Format

DBO/SHOW LOGICAL_NAMES logical-name

Command Qualifiers	Defaults
/DESCRIPTION	No description shown
/OUTPUT[=file-spec]	SYS$OUTPUT
/UNDEFINED	No undefined logical names are shown

Description

The DBO/SHOW LOGICAL_NAMES command displays the definitions of logical names known by various components of Oracle CODASYL DBMS. You can specify just one logical name, or use wildcards to select logical names of interest. The output format is similar to that of the DCL SHOW LOGICALS command.

Command Parameters

logical-name
Use this paramter to display the definition of a logical name. If you omit the logical name, the definitions of all logical names known to Oracle CODASYL DBMS are displayed. The logical-name string can contain OpenVMS style wildcard characters (*) and (%) to select more than one logical name.

Command Qualifiers

/DESCRIPTION

The /DESCRIPTION qualifier retrieves a brief description of the logical name and displays it along with the current definition. If wildcards are used for the logical name, then any matching logical names will also include output of the description.

The following example shows the use of the /DESCRIPTION qualifier, a wildcard logical name specification and the use of the UNDEFINED qualifier to include output; even for logical names not defined for this process or system.

```
$ DBO/SHOW LOG *SOR*/UNDEF/DESC
  "DBM$BIND_OPT_SORT_THRESHOLD" = Undefined

       This logical name is used by DBO/OPTIMIZE/AFTER_JOURNAL.

       This logical name may be defined to limit the number of records
       sorted at one time, which reduces the size of the sort workfiles
       when device space is limited. The efficiency of optimizing the
       journal is not as good as sorting all records in one pass, but
       the sort may be faster and device space for sort workfiles is
       greatly reduced. The optimized file will be a little larger if
       the sort is limited to a specified number of records.

          Default: no limit
          Minimum: 1 (if zero is specified the limit is assumed to be
          unlimited)
          Maximum: no limit

       See also the logical name DBM$BIND_OPT_TXN_THRESHOLD.


  "DBM$SORT_FILES" = Undefined

       Use this logical name to specify the number of temporary work
       files that a database sort operation is to use if the work files
       are required. The default number of work files is 2 and the
       maximum number is 10. For example:

       $ DEFINE DBM$SORT_FILES   3

       To control the placement of the work files define the various
       SORTWORKn logical names. The SORTWORKn logicals identify the
       device and directory of the work files. These logical names
       must point to a device that has a directory (allowing WRITE
       protection) with the same name as the user's home directory.

          Table: PROCESS
          Values: value between 2 (default) and 10
```

/OUTPUT=file-name

Specifies the name of the file where output is to be sent. The default is SYS$OUTPUT. The default output file type is .lis, if you specify a file name.

/UNDEFINED

Use the /UNDEFINED qualifier to display a list of both defined and undefined logicals.

Examples

The definitions of all logical names are maintained in a HELP library called SYS$HELP:DBODISPLAY74.HLB. The DCL HELP command can also be used to query this help library.

1.
```
$ HELP/LIBR=SYS$HELP:DBODISPLAY74.HLB DBMS_LOGICAL_NAMES DBMS$RUJ

DBMS_LOGICAL_NAMES

  DBMS$RUJ

       By default the Recovery Unit Journal file is located on the
       device on which the attached users default directory is located
       and in a special directory [DBM$RUJ]. This placement is used to
       avoid accidental deletion of this important file.

       However, you can use this logical name to locate the .ruj file on
       a different device and directory from the default location. This
       can help to reduce contention on the default device.

Topic?
```

2.
```
$ DBO/SHOW LOGICAL
  "DBM$BIND_DBR_LOG_FILE" = "DBR_PID.OUT" (LNM$SYSTEM_TABLE)
  "DBM$BIND_DBR_LOG_HEADER" = "0" (LNM$SYSTEM_TABLE)
  "DBM$DEMO" = "DBM$TOP:[DEMO]" (LNM$SYSTEM_TABLE)
  "DBM$MAILBOX_CHANNEL74" = "MBA12211:" (LNM$SYSTEM_TABLE)
  "DBM$MON_DIRECTORY" = "$1$DGA231:[DBMMON_LOGS]" (LNM$SYSTEM_TABLE)
```

This example displays defined logical names on the current system known to Oracle CODASYL DBMS.

3.
```
$ DBO/SHOW LOGICAL *ABS*/UNDEFINED ! Display them all
  "DBM$BIND_ABS_GLOBAL_STATISTICS" = Undefined
  "DBM$BIND_ABS_LOG_FILE" = "ABS_PID.OUT" (LNM$SYSTEM_TABLE)
  "DBM$BIND_ABS_LOG_HEADER" = Undefined
  "DBM$BIND_ABS_OVERWRITE_ALLOWED" = Undefined
   .
   .
   .
```

This example displays both defined and undefined logical names that contain the string ABS.

9.32.4 DBO/SHOW STATISTICS Command

Displays Oracle CODASYL DBMS processing statistics for a database on a single node.

Format

DBO/SHOW STATISTICS root-file-spec

Command Qualifiers	Defaults
/ALARM=interval	/ALARM=0
/[NO]BROADCAST	See description
/CLUSTER=(node-list)	/NOCLUSTER
/COLUMNS=integer	See description

/CONFIGURE=file-spec	None used
/CYCLE=seconds	/NOCYCLE
/DBKEY_LOG=file-spec	None
/DEADLOCK_LOG=file-spec	/NODEADLOCK_LOG
/FLUSH_INTERVAL=seconds	/FLUSH_INTERVAL=60
/HISTOGRAM	/HISTOGRAM
/HOT_STANDBY_LOG=file-spec	/NOHOT_STANDBY_LOG
/INPUT=file-spec	None
/[NO]INTERACTIVE	See description
/LOCK_TIMEOUT_LOG=file-spec	/NOLOCK_TIMEOUT_LOG
/[NO]LOG	See description
/MULTIPAGE_MAXIMUM=integer	/NOMULTIPAGE_MAXIMUM
/[NO]NOTIFY=([NO]ALL|operator-class-list)	/NONOTIFY
/OPCOM_LOG=file-spec	/NOOPCOM_LOG
/OPTIONS=(option[,...])	/OPTIONS=BASE
/OUTPUT=file-spec	
/PROMPT_TIMEOUT=seconds	/NOPROMPT_TIMEOUT
/REOPEN_INTERVAL=minutes	None
/REPORT_SCREEN=(option[,...])	See description
/RESET	/Statistics not reset
/ROWS=integer	See description
/SCREEN=screen-name	See description
/STALL_LOG=file-spec	None
/TIME=integer	/TIME=3
/UNTIL="absolute-time"	
/WRITE_REPORT_DELAY=integer	/NOWRITE_REPORT_DELAY

Description

The DBO /SHOW STATISTICS command dynamically samples processing activity statistics on a database for your current node. You can display the statistics on your terminal, write them to a formatted binary file, and write them to a text or report file. The statistics show activity only from the OpenVMS Cluster node on which you execute the command. The DBO /SHOW STATISTICS command in one of two modes: online and replay. In online mode, you can display or record current activity on a database. In replay mode, you can examine a previously recorded binary statistics file.

If you use the /INPUT qualifier, DBO executes in replay mode. In replay mode, DBO generates an interactive display from a previously recorded binary statistics file (/INPUT).

If you omit the /INPUT qualifier, you must specify a root-file-spec. DBO then executes in online mode. In online mode, DBO can generate an interactive display (/INTERACTIVE) and can also record statistics in a binary file (/OUTPUT).

The DBO interactive display is made up of numerous display pages. DBO allows you to control the interactive display by means of menus, using arrow keys or letters to select options. You can display information for most display pages in the form of a histogram or a numerical chart, although certain display pages, for instance Stall Messages and Transaction Durations, have special formats. In addition, you can produce time plots for individual statistical fields. See the OpenVMS Help utility and the *Oracle CODASYL DBMS Database Maintenance and Performance Guide* for information on interpreting the interactive displays.

Use the /OUTPUT qualifier to direct statistical output to a file. The output is a formatted binary file and does not produce a legible printed listing. To read the output, you must use the DBO /SHOW STATISTICS command with the /INPUT qualifier. You can also write your own program to reformat the file. The *Oracle CODASYL DBMS Database Maintenance and Performance Guide* describes the format of this binary statistics file in detail.

The /NOINTERACTIVE qualifier suppresses the interactive display. Use this when you want to generate binary statistics output but do not want an online display.

DBO maintains database statistics in a global section on each OpenVMS Cluster node. DBO resets the statistics to zero when you exit or close the database on your OpenVMS Cluster node. If you want to keep the database open for an extended time to observe processing activity, you can explicitly open the database by using the DBO/OPEN command. In addition, running a DBO /SHOW STATISTICS command keeps the database open even when there are no users accessing the database. Most displayed values can be reset using an online command.

There is a menu at the bottom of the statistics display page. Not all menu choices appear on every display page. The choices are:

Alarm	- specify a duration that a process must stall before it appears on the Stall Messages screen
Bell	- activate or deactivate the alarm bell
Brief	- display an abbreviated set of statistics
Configure	- configure the statistics screen
Exit	- return to DCL
Filter	- filter stall messages
Full	- display all available statistics
Graph	- display a histogram graph
Help	- display online help
Input	- display the Select Input Control menu
Lock	- display submenu of lock IDs
Menu	- display main menu
Normal	- return to normal screen display
Numbers	- display numeric statistics
Options	- select report options
Pause	- pause or resume the statistics display
Refresh	- refreshes the screen
Reset	- reset the statistics
Set_rate	- set the display update rate
Step	- advance one record in the input file while in paused mode
Time_plot	- plot a field's value by time
Unreset	- reverse effect of Reset option
Update	- change the value of a database attribute
Write	- write the screen to DBO.SCR

X_plot	- plot rate of current screen
Yank	- select statistics field
Zoom	- display detailed information about a specific item
>next_page	- display the next page
<prev_page	- display the previous page

Note

The following choices are mutually exclusive: **B**rief and **F**ull, and **G**raph and **N**umbers. Only one of each appears in the menu at a time.

To make a selection, press the first letter of the desired choice. You can also use these additional keys as follows:

D key	- Select a new display.
Space bar	- Initiate search for the next data screen that has ongoing activity within the current submenu group.
Plus (+) key	- Move forward the number of screens specified by the value you enter at the prompt. If you enter the value 0, move to the last screen.
Minus (-) key	- Move backward the number of screens specified by the value you enter at the prompt. If you enter the value 0, move to the first screen.
help or ? or PF2	- Display help.
up or down arrow	- Highlight a menu option.
Return, Enter, or Select	- Select a menu option.
Ctrl/W	- Repaint the screen.
Ctrl/Z	- Cancel a menu or return to DCL
← or left arrow	- Move to previous display.
→ or right arrow	- Move to next display.
!	- Invoke the Notepad facility
$	- Invoke DCL command facility
#	- Display submenu for current screen

Parameter

root-file-spec
Specifies the root file of the database on which you want statistics. The default file type is .ROO.

If you do not use the /INPUT qualifier, you must specify a root-file-spec. However, if you use the /INPUT qualifier to supply a prerecorded binary statistics file, you cannot specify a root-file-spec.

Command Qualifiers

/ALARM=interval
Establishes an alarm interval (in seconds) for the Stall Messages screen from the command line. This is useful when you plan to submit the DBO /SHOW STATISTICS command as a batch job.

Use this qualifier in conjunction with the /NOTIFY qualifier to notify an operator or set of operators of stalled processes.

The default value is 0 seconds, which is equivalent to disabling notification.

/BROADCAST
/NOBROADCAST
Specifies whether or not to broadcast messages. The /BROADCAST qualifier is the default, if broadcasting certain messages has been enabled with DCL SET BROADCAST. If broadcasting has been disabled with the DCL SET BROADCAST=none command, messages are not broadcast, even if you specify the DBO /SHOW STATISTICS command with the /BROADCAST qualifier.

Specify the /NOBROADCAST qualifier if broadcasting has been enabled with the DCL SET BROADCAST command but you do not want broadcast messages displayed while you are running the DBO /SHOW STATISTICS command.

/CLUSTER=(node-list)
/NOCLUSTER
Specifies the list of remote nodes from which statistics collection and presentation are to be performed. The collected statistics are merged with the information for the current node and displayed using the usual statistics screens.

The following list summarizes usage of the /CLUSTER qualifier:

- If the /CLUSTER qualifier is specified by itself, remote statistics collection is performed on all cluster nodes on which the database is currently open.
- If the /CLUSTER=(node-list) qualifier is specified, remote statistics collection is performed on the specified nodes only, even if the database is not yet open on those nodes.
- If the /CLUSTER qualifier is not specified, or the /NOCLUSTER qualifier (the default) is specified, cluster statistics collection is not performed. However, you can still enable clusterwide statistics collection online using the Tools menu.

You can specify up to 96 different cluster nodes with the Cluster qualifier. The current node is always included in the list of nodes from which statistics collection is to be performed.

It is not necessary to have the DBO /SHOW /STATISTICS command running on the specified remote nodes or to have the database open on the remote nodes. These events are automatically handled by the feature.

The following example shows the use of the /CLUSTER qualifier to initiate statistics collection and presentation from two remote nodes:

```
$ DBO /SHOW STATISTICS /CLUSTER=(PROD, DEPTS) PARTS
```

Remote nodes can also be added and removed online at run time. Use the Cluster Statistics option located in the Tools menu. The Tools menu is displayed by using the exclamation point (!) on-screen menu option.

See the *RMU Show Statistic DBA Handbook* (available on our Oracle Rdb and DBMS documentation web page) for information about the Cluster Statistics Collection and Presentation feature.

Both DECnet and TCP/IP network protocols are supported. By default, the DECnet protocol is used. To explicitly specify which network protocol to use, define the DBM$BIND_STT_NETWORK_TRANSPORT to DECNET or TCPIP respectively. The DBM$BIND_STT_NETWORK_TRANSPORT logical name must be defined to the same definition on both the local and cluster nodes. The DBM$BIND_STT_NETWORK_TRANSPORT logical name can be specified in LNM$FILE_DEV on the local node but must be specified in the LNM$SYSTEM_TABLE on all remote nodes.

/COLUMNS=integer

The dimensions of the DBO /SHOW STATISTICS command window will default to the user's display size or, in the case of non-interactive mode, 132 columns and 66 rows.

DBO /SHOW STATISTICS command allows you to specify the horizontal dimension using the /COLUMNS=n qualifier. Use this in conjunction with the /ROWS=n qualifier.

/CONFIGURE=file-spec

Specifies the name of a human-readable configuration file to be processed by the DBO /SHOW STATISTICS command. The configuration file can be created using any editor, or it can be automatically generated from the DBO /SHOW STATISTICS command using the current run-time configuration settings. The default configuration file type is .cfg.

If you specify the /CONFIGURE=file-spec qualifier, the configuration file is processed by the DBO /SHOW STATISTICS command prior to opening the database or the binary input file. If you do not specify this qualifier, all of the variables are the defaults based on command-line qualifiers and logical names.

The configuration file is processed in two passes. The first pass occurs before the database is opened and processes most of the configuration file entries. The second pass occurs after the database is opened and processes those variables that are database-dependent, such as the CUSTOM_LINE_n variable.

When a user selects "Custom Statistics", uses the tool "!" and selects "Save current configuration", then all custom statistics are saved in CUSTOM_LINE_n variables so that they get loaded in the next DBO /SHOW STATISTICS command session.

Please refer to Appendix A, Configuration File Management and User Defined Events for more information about configuration files.

/CYCLE=seconds
/NOCYCLE

Provides an automated, continual cycle through the set of screens at the current menu level, at the time interval you specify in seconds. The default if you do not specify the interval or if you do not specify the qualifier, is for no automated cycling to occur; you must migrate from screen to screen manually.

When you specify the /CYCLE qualifier, the DBO /SHOW STATISTICS command continues to operate normally. You can change screen modes or change submenus at will; however, the /CYCLE operation continues at the current menu level.

The specified screen cycle rate cannot be less than the screen refresh rate. That is, the /CYCLE qualifier must be greater than or equal to the /TIME qualifier. Additionally, if you manually change the refresh rate using the Set_rate menu option, you cannot alter the screen cycle time until you specify a refresh rate that is less than the specified /CYCLE interval.

You can use the Notepad facility (!) to stop online cycling.

/DBKEY_LOG=file-spec
/NODBKEY_LOG
Logs the records accessed during a given processing period by the various bound processes. The file-spec is the name of the file to which all accessed dbkeys are logged.

The accessed dbkeys are written in a human-readable format similar to that of the DBKEY Information screen.

The dbkey information is written at the current screen refresh rate, determined using the /TIME command qualifier or on line using the Set_rate onscreen menu option. Obviously, using a larger refresh rate will minimize the size of the file, but result in a large number of missed dbkey messages. Using a smaller refresh rate will produce a large log file, but will contain a much finer granularity of dbkey messages.

You do not need to display the DBKEY Information screen in order to record the dbkey messages to the dbkey log. The dbkey log is maintained regardless of which screen, if any, is displayed.

You can easily construct a dbkey logging server with the /DBKEY_LOG command qualifier. The following DCL command provides this function:

```
$ DBO/SHOW STAT/NOHIST/TIME=1/NOINTER/DBKEY=DBKEY.LOG PARTS -
    /NOBROADCAST/UNTIL="15:15:00"
```

/DEADLOCK_LOG=file-spec
/NODEADLOCK_LOG
Records the last deadlock for the processes. There is no method to record each lock deadlock as it occurs.

The file-spec in the qualifier is the name of the file to which you want all lock deadlock messages to be logged. The lock deadlock messages are written in human-readable format similar to the Lock Timeout History and Lock Deadlock History screens.

The header region of the lock deadlock log contains three lines:

- Line 1 indicates that the DBO /SHOW STATISTICS command created the log file.
- Line 2 identifies the database.
- Line 3 identifies the date and time the log was created.

The main body of the stall log contains three columns:

- The first column contains the process ID and stream ID that experienced the lock deadlock.
- The second column contains the time the deadlock occurred; however, the date is not displayed.

- The third column contains the deadlock message describing the affected resource. This message is similar to the originating stall message.

For example:

```
2EA00B52:34 14:25:46.14 - waiting for page 5:751 (PR)
```

If any lock deadlocks are missed for a particular process (usually because the recording interval is too large), the number of missed lock deadlocks is displayed in brackets after the message. For example:

```
2EA00B52:34 14:25:46.14 - waiting for page 5:751 (PR) [1 missed]
```

Only one message is logged for each deadlock.

The lock deadlock messages are written at the specified screen refresh rate, determined by specifying the /TIME qualifier, or online using the Set_rate on-screen menu option. Using a larger refresh rate minimizes the size of the file, but results in a large number of missed deadlock messages. Using a smaller refresh rate produces a large log file, but contains a much finer granularity of deadlock messages.

Using the /TIME=1 or /TIME=50 qualifier produces a reasonable log while minimizing the impact on the system.

The affected LockID is not displayed, because this is meaningless information after the lock deadlock has completed.

Use the Tools menu (displayed when you press the exclamation point (!) key from any screen) to enable or disable the lock timeout and lock deadlock logging facility while the DBO /SHOW STATISTICS command is running. However, note that the lock timeout log and lock deadlock log are not available during binary file replay.

/FLUSH_INTERVAL=seconds
/NOFLUSH_INTERVAL

The FLUSH interval is important because it insures that, if a system failure occurs, important data is not lost from the DBO /SHOW STATISTICS command output files (the binary file used to record database statistics for later replay, the record access dbkey log file, the process deadlock log file, the lock timeout log file, the OPCOM messages log file, the Hot Standby log file, the online analysis log file, and the stall messages log file). These periodic buffer data flushes initiated by the /FLUSH_INTERVAL qualifier will help to minimize this loss of diagnostic data.

The default value is 60 seconds if no /FLUSH_INTERVAL is specified. The minimum flush interval that can be specified is 0 seconds. The maximum flush interval that can be specified is 3600 seconds (1 hour). Specifying 0 seconds for the flush interval is equivalent to specifying /NOFLUSH_INTERVAL. If the flush interval is active and less than the statistics collection interval, to avoid unnecessary buffer flushes, the flush interval will be set to the statistics collection interval, which has a default of 3 seconds and can be set by the existing /TIME qualifier, or by typing an "S" if "Set rate" is displayed at the bottom of the statistics screen. The currrent collection interval is displayed following "Rate:" in the statistics screen header.

/HISTOGRAM
/NOHISTOGRAM

Specifies whether the DBO /SHOW STATISTICS command initially displays statistics in histogram or tabular numeric format. The default is /HISTOGRAM.

To initially display statistics in tabular numeric format, use the /NOHISTOGRAM qualifier, as shown in the following example:

```
$ DBO/SHOW STATISTICS/NOHISTOGRAM PARTS

Node: NODEA Oracle CODASYL DBMS V7.0                        7-NOV-1996 17:05:21
Rate: 3.00 Seconds          Summary IO Statistics       Elapsed: 00:00:11.74
Page: 1 of 1       ABC$DISK:[DATABASE]PARTS.ROO;1          Mode: Online

statistic.....  rate.per.second.............. total.......  average......
name..........  max..... cur..... avg....... count.......  per.trans....

transactions                0        0        0.0            0            0.0
verb successes              0        0        0.0            0            0.0
verb failures               0        0        0.0            0            0.0

synch data reads            0        0        0.0            0            0.0
synch data writes           0        0        0.0            0            0.0
RUJ file reads              0        0        0.0            0            0.0
RUJ file writes             0        0        0.0            0            0.0
AIJ file reads              0        0        0.0            0            0.0
AIJ file writes             0        0        0.0            0            0.0
root file reads             9        0        1.4           16            0.0
root file writes            0        0        0.0            0            0.0

Display_menu Exit Graph Help Options  Reset Set_rate  Time_plot  Write_screen
```

To switch to histogram format, enter the "Graph" on-screen menu option.

To initially display statistics in the histogram format, use the /HISTOGRAM qualifier with DBO /SHOW STATISTICS command. This is the default.

/HOT_STANDBY_LOG=file-spec
/NOHOT_STANDBY_LOG
Specifies the name of the Hot Standby log file. The "Start hot standby logging" option of the Tools menu (enter !) can be used to specify the name of the Hot Standby log file at runtime.

/INPUT=file-spec
Specifies the prerecorded binary file from which you can read the statistics. This file must have been created by an earlier DBO /SHOW STATISTICS command session that specified the /OUTPUT qualifier. The default file type is .DAT. The binary input file can be used only on the same version of software that was used to create the output file.

You cannot specify a root-file name with the /INPUT qualifier. In addition, you cannot use the /UNTIL, /OUTPUT, or /NOINTERACTIVE qualifier with the /INPUT qualifier.

You can use /TIME to change the rate of the display. This does not change the computed times as recorded in the original session; it simply changes the speed at which the statistics are replayed. For example, you can record a session at /TIME=60. This session gathers statistics once a minute. You can then replay the session at /TIME=1. The replay changes the display once a second, thus replaying 10 hours of statistics in 10 minutes.

If you do not specify the /INPUT qualifier, you must specify the root-file-spec parameter.

/INTERACTIVE
/NOINTERACTIVE
Displays the statistics dynamically on your terminal. The /INTERACTIVE qualifier is the default for interactive processes.

Specify /NOINTERACTIVE with the /OUTPUT qualifier to generate a binary statistics file without generating a terminal display. The /NOINTERACTIVE qualifier is the default for batch processes.

When running interactively, you can use the menu interface to select display options.

/LOCK_TIMEOUT_LOG=file-spec
/NOLOCK_TIMEOUT_LOG
Records the last lock timeout message for the processes. There is no method to record each lock timeout as it occurs. The lock timeout messages are written in human-readable format.

The header region of the lock timeout log contains three lines:

- Line 1 indicates that the DBO /SHOW STATISTICS command created the log file.
- Line 2 identifies the database.
- Line 3 identifies the date and time the log was created.

The main body of the stall log contains three columns:

- The first column contains the process ID and stream ID that experienced the lock timeout.
- The second column contains the time the timeout occurred; however, the date is not displayed.
- The third column contains the timeout message describing the affected resource. This message is similar to the originating stall message.

For example:

```
2EA00B52:34 14:25:46.14 - waiting for page 5:751 (PR)
```

If any lock timeouts are missed for a particular process (usually because the recording interval is too large), the number of missed lock timeouts is displayed in brackets after the message. For example:

```
2EA00B52:34 14:25:46.14 - waiting for page 5:751 (PR) [1 missed]
```

Only one message is logged for each lock timeout.

The lock timeout messages are written at the specified screen refresh rate, determined by specifying the /TIME qualifier, or online using the Set_rate on-screen menu option. Using a larger refresh rate minimizes the size of the file, but results in a large number of missed lock timeout messages. Using a smaller refresh rate produces a large log file, but contains a much finer granularity of lock timeout messages.

Using the /TIME=1 or /TIME=50 qualifier appears to produce a reasonable log while minimizing the impact on the system.

The affected LockID is not displayed because this is meaningless information after the lock timeout has completed.

Note that you do not need to be displaying the Lock Timeout History or Lock Deadlock History screens to record the stall messages to the stall log. These logs are maintained regardless of which screen, if any, is displayed.

Use the Tools menu (displayed when you press the exclamation point (!) key from any screen) to enable or disable the lock timeout and lock deadlock logging facility while the DBO /SHOW STATISTICS command is running. However, note that the lock timeout log and lock deadlock log are not available during binary file replay.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request a listing of the creation of a binary statistics file and /NOLOG to prevent this listing. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

DBO creates this binary statistics file only if you have used the /OUTPUT qualifier. DBO ignores the /LOG qualifier if you specify /INTERACTIVE.

/MULTIPAGE_MAXIMUM=integer
/NOMULTIPAGE_MAXIMUM
By default DBO /SHOW STATISTICS command will write all pages for each display when writing to a report file. For some databases, for example, those with a large number of row cache slots, this output may be quite lengthy. To assist in reducing the size of the output report file, the /MULTIPAGE_MAXIMUM=n qualifier can be used.

When specified, the /MULTIPAGE_MAXIMUM qualifier accepts a number representing the maximum number of pages to write for any multi-page display. For example, if /MULTIPAGE_MAXIMUM=5 is specified, only the first five pages of any multi-page display are written to the output file. If /MULTIPAGE_ MAXIMUM is not specified, the default behavior of "unlimited" is assumed and all pages for multi-page displays are written to the output file.

/NOTIFY
/NOTIFY=[NO]ALL
/NOTIFY=operator-classes
/NONOTIFY
Notifies the specified system operator or operators when a stall process exceeds the specified alarm interval. Notification is made by broadcasting a message and ringing a bell at the terminal receiving the message.

The valid operator classes are: CENTRAL, CLUSTER, DISKS, SECURITY, and OPER1 through OPER12.

The various forms of the /NOTIFY qualifier have the following effects:

- If you specify the /NOTIFY qualifier without the operator-classes parameter, the CENTRAL and CLUSTER operators are notified by default.
- If you specify the /NONOTIFY or /NOTIFY=NOALL qualifiers, operator notification is disabled.
- If you specify the /NOTIFY=ALL qualifier, all operator classes are enabled.
- If you specify the /NOTIFY=operator-classes qualifier, the specified classes are enabled. (If you specify more than one operator class, enclose the list in parentheses and separate each class name with a comma.)

For example, issuing the DBO /SHOW STATISTICS command with the /NOTIFY=(OPER1, OPER2) qualifier sends a notification message to system operator classes OPER1 and OPER2 if the alarm threshold is exceeded while monitoring the Stall Messages screen.

The specified system operator(s) are notified only when the alarm threshold is first exceeded. For instance, if three processes exceed the alarm threshold, the specified operator(s) are notified only once. If another process subsequently exceeds the alarm threshold while the other processes are still displayed, the specified system operator(s) are not notified.

However, if the longest-duration stall is resolved and a new process then becomes the newest stall to exceed the alarm threshold, then the specified system operator(s) will be notified of the new process.

To receive operator notification messages, the following three OpenVMS DCL commands must be issued:

- SET TERM /BROADCAST
- SET BROADCAST=OPCOM
- REPLY /ENABLE=(operator-classes)

The operator-classes specified in the DCL REPLY /ENABLE command must match those specified in the /NOTIFY qualifier to the DBO /SHOW STATISTICS command.

If you specify the /NOINTERACTIVE qualifier, bell notification is disabled, but the broadcast message remains enabled.

/OPCOM_LOG=file-spec
/NOOPCOM_LOG
Specifies the name of the file where OPCOM messages broadcast by attached database processes will be sent.

When recording OPCOM messages, it is possible to occasionally miss a few messages for a specific process. When this occurs, the message "n missed" will be displayed in the log file.

You can record specific operator classes of OPCOM messages if you specify the /OPTION=VERBOSE qualifier. The /OPTION=VERBOSE qualifier records only those messages that can be received by the process executing the DBO /SHOW STATISTICS command. For example, if the process is enabled to receive operator class CENTRAL, then if you specify /OPCOM_LOG=OPCOM.LOG the /OPTION=VERBOSE qualifier records all CENTRAL operator messages. Conversely, specifying only the /OPCOM_LOG=OPCOM.LOG qualifier records all database-specific OPCOM messages generated from this node. Because the output is captured directly from OpenVMS, the operator-specific log file output format is different from the database-specific contents.

/OPTIONS=(option[,...])
Allows you to select the collection of by-file or by-area statistics information. This qualifier is only applicable when used in conjunction with the /OUTPUT qualifier.

The /OPTIONS qualifier provides several keywords that allow you to indicate which statistics information is to be collected. The keywords are as follows:

- [NO]ALL

This keyword can be negated and indicates whether or not all collectible statistics are to be collected. The ALL keyword indicates that all statistics information is to be collected; the NOALL keyword indicates that only the base statistics information is to be collected.

- [NO]AREA

  This keyword can be negated and indicates whether or not the by-area statistics information are to be collected, in addition to the base statistics information.

  When you specify the AREA keyword, statistics for all existing storage areas are written to the binary output file. You cannot selectively choose specific storage areas whose statistic information is to be collected.

- BASE

  This keyword cannot be negated and indicates that only the base set of statistics are to be collected.

- COMPRESS

  Compresses the statistics records written to the output file specified by the Output qualifier. While replaying the statistics, the DBO /SHOW STATISTICS command determines if a record was written using compression or not. If the record was written using compression it is automatically decompressed.

  If compression is used, the resultant binary file can be read only by the DBO /SHOW STATISTICS command. The format and contents of a compressed file are not documented or accessible to other applications.

- CONFIRM

  Indicates that you wish to confirm before exiting from the utility. You can also specify the Confirm option in the configuration file using the CONFIRM_EXIT variable. A value of TRUE indicates that you want to confirm before exiting the utility and a value of FALSE (the default) indicates you do not want to confirm before exiting the utility.

- LOG_STALL_ALARM

  If LOG_STALL_ALARM is present when using the /STALL_LOG qualifier to write stall messages to a log file and the /ALARM qualifier to set an alarm interval, only those stalls exceeding the /ALARM specified duration are written to the stall log output file.

- LOG_STALL_LOCK

  If you use the /STALL_LOG qualifier to write stall messages to a log file, use the NOLOG_STALL_LOCK option to prevent lock information from being written to the log file. If you use or omit the LOG_STALL_LOCK option, lock information is written to the log file.

- SCREEN_NAME

  Allows you to identify a screen capture by screen name. If you issue an DBO /SHOW STATISTICS command with the /OPTIONS=SCREEN_NAME qualifier, the screen capture is written to a file that has the name of the screen with all spaces, brackets, and slashes replaced by underscores. The file has an extension of .SCR. For example, if you use the /OPTION=SCREEN_NAME qualifier and select the Write option on the Screen Transaction Duration (Read/Write), the screen is written to a file named TRANSACTION_DURATION_READ_WRITE.SCR.

- ROW_CACHE

  This keyword indicates that Row Cache Statistics screen be written to the binary output file. This screen provides summary information for a specific record cache.

- UPDATE

  This keyword indicates that you intend to change the value of a database attribute using the Database Dashboard. You must have the OpenVMS WORLD, BYPASS, and SYSNAM privileges when you issue the DBO /SHOW STATISTICS command to update values through the Database Dashboard.

  Use the UPDATE option carefully. Oracle CODASYL DBMS does not perform error checking on the updated values.

- VERBOSE

  Causes the stall message logging facility to report a stall message at each interval, even if the stall message has been previously reported.

  Note

  Use of the /OPTIONS=VERBOSE qualifier can result in an enormous stall messages log file. Ensure that adequate disk space exists for the log file when you use this qualifier.

  You can enable or disable the stall messages logging VERBOSE option at run time by using the Tools menu and pressing the exclamation point (!) key.

  You can also specify the Verbose option in the configuration file by using the STALL_LOG_VERBOSE variable. Valid keywords are ENABLED or DISABLED.

  Lock information is displayed only once per stall, even in verbose mode, to minimize the output file size.

The default for the /OPTIONS qualifier is /OPTIONS=BASE. You cannot omit the base statistics from the output. When you specify more than one keyword, the list of keywords is processed in the order specified, with a cumulative effect. For example:

/OPTIONS=(AREA,NOALL,BASE)	selects BASE
/OPTIONS=(ALL,NOAREA)	selects BASE
/OPTIONS=(NOAREA,AREA)	selects BASE and AREA

The following command creates the binary output file including the by-area statistics:

```
$ DBO/SHOW STATISTIC/OUTPUT=MAIN.STATS/OPTIONS=(BASE,AREA) PARTS
```

The size of the by-area statistics output largely depends on the total number of storage areas in the database. If the database contains a large number of storage areas, it may not be advisable to use the /OPTIONS=AREA qualifier.

Note that stall messages are not currently collected in the binary output file.

/OUTPUT=file-spec

Specifies a binary statistics file into which DBO writes the statistics. You must reformat the file with a program to obtain legible output.

You can replay the statistics collected in this file by using the DBO /SHOW STATISTICS command with the /INPUT qualifier.

Note

Statistics from the Stall Messages display are *not* collected in the binary output file.

/PROMPT_TIMEOUT=seconds
/NOPROMPT_TIMEOUT
Allows you to specify the user prompt timeout interval, in seconds. The default value is 60 seconds.

If you specify the /NOPROMPT_TIMEOUT qualifier or the /PROMPT_TIMEOUT=0, the DBO /SHOW STATISTICS command does not time out any user prompts. Note that this can cause your database to hang.

Note

Oracle Corporation recommends that you do not use the /NOPROMPT_TIMEOUT qualifier or the /PROMPT_TIMEOUT=0 qualifier unless you are certain that prompts will always be responded to in a timely manner.

If the /PROMPT_TIMEOUT qualifier is specified with a value greater than 0 but less than 10 seconds, the value 10 is used. The user prompt timeout interval can also be specified using the PROMPT_TIMEOUT configuration variable.

/REOPEN_INTERVAL=minutes
Causes the DBO /SHOW STATISTICS command to periodically close the current output file (specified with the /OUTPUT qualifier) and open a new version of the file without requiring you to close the utility. Using this qualifier, you can view data written to the output file while the DBO /SHOW STATISTICS command is running.

The minutes parameter specifies the inverval at which the output file is to be closed and reopened. If there is no database activity during the specified interval, a new version of the output file is not created. Note that if you issue the DCL PURGE command, you lose all but the most recent version of the output file. Furthermore, if you use the DCL SET FILE/VERSION_LIMIT command, older versions of the output file are automatically deleted.

This qualifier is valid only when the /OUTPUT qualifier is also specified.

/REPORT_SCREEN=(INCLUDE="string")
/REPORT_SCREEN=(EXCLUDE="string")
Allows individual screens to be included or excluded from the STATISTICS.RPT file by specifying a full screen name or parts of a screen name using the REPORT_SCREEN qualifier. The specification may include wild card character "*" to match multiple characters and "%" to match a single character.

This qualifier /REPORT_SCREEN supports these keywords:

- INCLUDE="string"

  /REPORT_SCREEN=(INCLUDE="string")

This keyword will include in the report only those screens with a screen name matching all or part of the specified "string". The string may contain wild card characters.

- EXCLUDE="string"

  /REPORT_SCREEN=(EXCLUDE="string")

  This keyword will exclude from the report only those screens with a screen name matching all or part of the specified "string". The string may contain wild card characters.

INCLUDE and EXCLUDE options cannot both be specified in the same DBO /SHOW STATISTICS command. The REPORT_SCREEN qualifier is only valid when the WRITE_REPORT_DELAY qualifier is used.

To include or exclude all the "SUMMARY" screens from the report, the commands would be similar to these examples.

```
$ DBO/SHOW STATISTICS -
    /WRITE_REPORT_DELAY=120 -
    /REPORT_SCREEN=INCLUDE="SUMMARY*" -
    DATABASE.ROO
$
$ DBO/SHOW STATISTICS -
    /WRITE_REPORT_DELAY=120 -
    /REPORT_SCREEN=EXCLUDE="Summary*" -
    DATABASE.ROO
$
```

The string matching is case insensitive. That is, the strings "SUMMARY*" and "Summary*" will be treated the same and will match the following screen header titles containing "Summary".

- Summary IO Statistics
- Summary Locking Statistics
- Summary Object Statistics
- Summary Tx Statistics

/RESET

Specifies that you want the DBO /SHOW STATISTICS command to reset your display to zero. The /RESET qualifier has the same effect as selecting the reset option from the interactive screen (except when you specify the /RESET qualifier, values are reset before being initially displayed).

Note that this qualifier resets the values being displayed to your output device only, it does *not* reset the values in the database global section nor does it affect the data collected in an output file.

The default behavior of the DBO /SHOW STATISTICS command is to display each change in values that has occurred since the database was opened. To display only the value changes that have occurred since the DBO /SHOW STATISTICS command was issued specify the /RESET qualifier, or immediately select the on-screen reset option when statistics are first displayed.

The /RESET qualifier does not affect the values that are written to the binary output file (created when you specify the /OUTPUT qualifier). Specify the /RESET qualifier when you replay the output file if you want the replay to display only the change in values that occurred between the time the you issued the DBO

/SHOW STATISTICS command (with the /OUTPUT qualifier) and the monitoring session ended.

/ROWS=integer
The dimensions of the DBO /SHOW STATISTICS command window will default to the user's display size or, in the case of non-interactive mode, 132 columns and 66 rows.

DBO /SHOW STATISTICS command allows you to specify the vertical dimension using the /ROWS=n qualifier. Use this in conjunction with the /COLUMNS=n qualifier.

/SCREEN=screen-name
Specifies the first screen to be displayed. This is particularly useful when you are using the DBO /SHOW STATISTICS command to interactively monitor stalled processes. For example, the following command automatically warns the system operator of excessive stalls:

```
$ DBO/SHOW STATISTICS-
_$      /ALARM=5-
_$      /NOTIFY=OPER12-
_$      /SCREEN="Stall Messages" -
_$      PARTS
```

The following list describes the syntax of the screen-name argument:

- You can use any unique portion of the desired screen name for the screen-name argument. For example, the following has the same results as the preceding example:

  ```
  $ DBO/SHOW STATISTICS-
  _$      /ALARM=5-
  _$      /NOTIFY=OPER12-
  _$      /SCREEN="Stall" -
  _$      PARTS
  ```

- The match is case insensitive, however, whatever unique portion of the screen you supply must be an exact match to the equivalent portion of the actual screen name.

  For example Screen="Stall" is equivalent to Screen="STALL"; however Screen="Stalled" is not.

- If the specified screen-name does not match any known screen name, the display starts with the Summary IO Statistics screen (the default first screen). No error message is produced.

- If the screen name contains spaces, enclose the screen-name within quotation marks.

- You can not specify the "by-lock" or "by-area" screens.

If you specify the /NOINTERACTIVE qualifier, the /SCREEN qualifier is ignored.

/STALL_LOG=file-spec
/NOSTALL_LOG
Specifies that stall messages are to be written to the specified file. This can be useful when you notice a great number of stall messages being generated, but do not have the resources on hand to immediately investigate and resolve the problem. The file generated by the /STALL_LOG qualifier can be reviewed later so that the problem can be traced and resolved.

Only one message per stall is displayed. Stall messages are written to the file at the same rate as the screen refresh rate. (The refresh rate is set with the /TIME qualifier or from within the Performance Monitor with the Set_rate on-screen menu option.) Specifying a large refresh rate minimizes the size of the file, but results in a large number of missed stall messages. Specifying a small refresh rate produces a large log file, but contains more of the stall messages generated.

The LockID is not displayed because this is meaningless information after the stall has completed.

You do not need to be displaying the Stall Messages screen to record the stall messages to the log file. The stall log is maintained regardless of which screen, if any, is displayed.

With the /STALL_LOG qualifier, you can construct a stall logging server. The following DCL command provides this function:

```
$ DBO/SHOW STAT/NOHIST/TIME=1/NOINTER/STALL=STALL.LOG PARTS -
    /NOBROADCAST/UNTIL="+15:00"
```

By default, stall messages are not logged to a file.

/TIME=integer
Defines the statistics collection interval in seconds. You can specify any integer from 1 to 180 (1 second to 3 minutes). The default is 3 seconds. If you are running DBO interactively, DBO updates the screen display at the specified interval.

If you are also using the /OUTPUT qualifier, DBO writes a binary statistics record to the output file at the specified interval. DBO does not write a statistics record to this file if no database activity has occurred since it wrote the last record. You can use /TIME to change the rate of the display. This does not change the computed times as recorded in the original session. For example, you can record a session at /TIME=60. This session gathers statistics once a minute. You can then replay the session at /TIME=1. The replay changes the display once a second, thus replaying 10 hours of statistics in 10 minutes.

If you specify a negative number for the /TIME qualifier, DBO interprets the number as hundredths of a second. For example, /TIME=-20 specifies an interval of 20/100 or 1/5 of a second.

/UNTIL="absolute-time"
Indicates when you want the statistics collection to end. At the specified time, the DBO /SHOW STATISTICS command terminates and control returns to DCL command level. Whether you are running DBO in a batch job or interactively, the statistics collection terminates at the time specified.

If you do not use the /UNTIL qualifier, the DBO /SHOW STATISTICS command continues until you terminate it manually. When running interactively, terminate the DBO /SHOW STATISTICS command by pressing Ctrl/Z or by selecting EXIT from the menu. When running DBO from a terminal with the /NOINTERACTIVE qualifier, terminate the DBO /SHOW STATISTICS command by pressing Ctrl/C or Ctrl/Y and then selecting EXIT. When running the DBO /SHOW STATISTICS command in a batch job, terminate the operation by deleting the batch job.

/WRITE_REPORT_DELAY=integer
/NOWRITE_REPORT_DELAY
This qualifier specifies that statistics are to be collected for "n" seconds (default of 60 seconds) and then a report file written and then the DBO /SHOW STATISTICS

command will exit. /WRITE_REPORT_DELAY implies /NOINTERACTIVE. It can be negated with /NOWRITE_REPORT_DELAY.

Examples

1. `$ DBO/SHOW STATISTICS/OUTPUT=STATS PARTS`

   This command creates a binary file called STATS from which you will be able to read the statistics generated from the database PARTS.

2. `$ DBO/SHOW STATISTICS/INPUT=STATS.DAT`

   This command reads the binary file STATS.DAT, created in the previous example, and displays the output on the terminal screen.

9.32.5 DBO/SHOW SYSTEM Command

Displays summary information on all databases accessed on a single node.

Format

DBO/SHOW SYSTEM

Command Qualifier	Default
/OUTPUT=file-spec	SYS$OUTPUT

Description

This command displays summary information on all databases accessed on your current node. It is similar to DBO/SHOW USERS except that it has no root file parameter. Thus, you can use it to see systemwide user statistics only. See the description of DBO/SHOW USERS in the next section.

Command Qualifier

/OUTPUT=file-spec
Names the file where output will be sent. The default file type is .LIS. The default output is SYS$OUTPUT.

Example

```
$ DBO/SHOW SYSTEM
```

This command displays a summary of all database activity on the current node.

9.32.6 DBO/SHOW USERS Command

Displays detailed information on one or all databases accessed on a single node.

This command includes information on whether or not system space global sections are enabled and displays the active values for the global buffer parameters.

Format

DBO/SHOW USERS [root-file-spec]

Command Qualifier	Default
/OUTPUT=file-spec	SYS$OUTPUT

Description

The DBO/SHOW USERS command displays detailed information on one or all databases accessed on your current node. It is similar to DBO/SHOW SYSTEM. However, you can use a root-file-spec parameter to limit the display of information to one database.

Parameter

root-file-spec
Specifies the root file of the database for which you want information. This parameter is optional. If you specify it, DBO shows only the users of that database. Otherwise, DBO shows all users of all active databases. The default file type is .ROO.

Command Qualifier

/OUTPUT=file-spec
Names the file where output will be sent. The default file type is .LIS. The default output is SYS$OUTPUT.

Example

```
$ DBO/SHOW/OUTPUT=DBUSE USERS
```

This command lists current USERS information in the file DBUSE.LIS.

9.32.7 DBO/SHOW VERSION Command

Displays information about the currently executing database version.

Format

DBO/SHOW VERSION [root-file-spec]

Command Qualifier	Default
/OUTPUT=file-spec	SYS$OUTPUT

Description

The DBO/SHOW VERSION command displays the version number of the Oracle CODASYL DBMS monitor currently executing on your node.

Parameter

root-file-spec
Specifies the root file of the database on which you want information. This parameter is optional. If you specify it, DBO displays the version of that database. If you omit it, DBO displays the version of the current Oracle CODASYL DBMS monitor. The default file type is .ROO.

Command Qualifier

/OUTPUT=file-spec
Names the file where output will be sent. The default file type is .LIS. The default output is SYS$OUTPUT.

Example

```
$ DBO/SHOW VERSION
Executing DBO for Oracle CODASYL DBMS V7.4-10 on IA64 V8.4-2L3
```

This command requests the version of the Oracle CODASYL DBMS monitor running on the local node. The response will include the Oracle CODASYL DBMS version, and the current OpenVMS version.

9.33 DBO/UNLOAD Commands

There are two forms of the DBO/UNLOAD command, each having its own qualifiers and parameters. There are:

DBO/LOAD

This format unloads data from a database.

DBO/UNLOAD/CONTINUE

This format recovers and continues an interrupted unload operation.

These formats are described in the following sections.

9.33.1 DBO/UNLOAD Command

Unloads data from a database to output source files.

Format

DBO/UNLOAD root-file-spec

Command Qualifiers	Defaults
/ANALYZE	
/FILE=file-spec	/FILE=SYS$DISK:[]
/FORMAT=lfl-file-spec	/FORMAT=root-file-spec.LFL
/[NO]LOG	Current DCL verify value
/[NO]QUIET_POINT=integer	/QUIET_POINT=1000
/READY=(allow-mode,access-mode)	/READY=(E,R)
/[NO]SAVE=save-file-spec	/SAVE=root-file-spec.USV
/[NO]SELECTIVE	/NOSELECTIVE

/SEQUENCE=usl-file-spec	/SEQUENCE=root-file-spec.USL
/SUBSCHEMA=subschema-name	/SUBSCHEMA=DEFAULT_SUBSCHEMA

Description

Use the DBO/UNLOAD command as a general-purpose data extraction tool in conjunction with the DBO/LOAD command to unload and then reload a database.

To use DBO/UNLOAD, you must already have a schema, subschema, storage schema, and root file.

The DBO/UNLOAD command extracts data from a database according to the view defined by the subschema view and the format (.LFL) file. DBO/UNLOAD creates a homogeneous output source file for each record named in the unload sequence language (.USL) file. The /FILE qualifier gives the file specification under which the homogeneous source files will be stored. The default file specification is:

```
LFL_rec_name.DAT
```

In the previous file specification, LFL_rec_name refers to the first nine alphanumeric characters of the LFL record name.

All record instances of a given record type are put in one file.

You can edit the .LFL file before you use it to reload the database. Thus, you can use DBO/LOAD and DBO/UNLOAD to modify the database. You can also use the files to move records from one database to another.

The unload operation can be restarted, if necessary. DBO uses the save file, created by default during the unload operation, to restart the unload procedure. See DBO/UNLOAD/CONTINUE for more information.

Chapter 10 shows the commands of the .LFL and .USL files and explains briefly how they relate to the DBO/UNLOAD command. See the *Oracle CODASYL DBMS Database Load/Unload Guide* for tutorial information on loading and unloading database files.

Parameter

root-file-spec
Specifies the root file of the database to be unloaded. The DBO/UNLOAD command takes the schema, subschema, and storage schema from the root file. The default file type is .ROO.

Command Qualifiers

/ANALYZE
Generates an analysis that shows which realms will be readied in which mode. This is useful in predicting conflicting areas between parallel unload operations before you attempt to unload the database.

This qualifier is only valid when used with the /SELECTIVE qualifier.

/FILE=file-spec
Specifies globally the device and directory for the record files that are unloaded. If you do not specify /FILE, DBO uses the default device and directory.

/FORMAT=lfl-file-spec
Specifies the name of the .LFL file that defines the data and set linkages. This file can be used as input if you use DBO/LOAD to reload the database. (The unload format language is the same as the load format language; thus, the same file can be used for both load and unload operations.) The default file specification is:

root-file-spec.LFL

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request reporting of DBO/UNLOAD activity and /NOLOG to prevent this reporting. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/QUIET_POINT=integer
/NOQUIET_POINT
Specifies how frequently a COMMIT RETAINING is to be issued. The integer gives the number of find operations that will occur between unload quiet points. The default is 1000.

Specify /NOQUIET_POINT to delay the COMMIT until DBO finishes the UNLOAD operation.

The /NOQUIET_POINT qualifier is incompatible with the BATCH RETRIEVAL usage mode.

/READY=(allow-mode,access-mode)
Sets the area lock for the areas you are unloading. Valid allow-modes are EXCLUSIVE, BATCH, and PROTECTED. The only valid access-mode is RETRIEVAL. The default is /READY=(EXCLUSIVE, RETRIEVAL).

/SAVE=save-file-spec
/NOSAVE
Specifies the name of a save file. This save file records the progress of the unload operation. This record is used for recovery if the unload operation terminates abnormally. The default file specification is:

root-file-spec.USV

Specifying /NOSAVE prevents the generation of this file.

See the description of the DBO/UNLOAD/CONTINUE command in the next section for details about the contents and use of the save file.

/SELECTIVE
/NOSELECTIVE
Readies only the required realms with the required mode. The /SELECTIVE qualifier also works with the /CONTINUE qualifier so you can change back and forth between readying all the realms and only those necessary when you continue the unload operation.

If the /SELECTIVE qualifier is used with the /LOG qualifier, the unload operation produces an informational message showing which areas are readied in which mode.

Use the /NOSELECTIVE qualifier to ready all realms. This is the default.

/SEQUENCE=usl-filename
Specifies the file containing the unload sequence language (USL), which the DBO/UNLOAD command uses to specify the order of the unload.

The default file specification is:

root-file-spec.USL

/SUBSCHEMA=subschema-name
Specifies the subschema through which DBO/UNLOAD unloads the record files. The default is /SUBSCHEMA=DEFAULT_SUBSCHEMA.

Example

```
$ DBO/UNLOAD/LOG/SEQUENCE=DBMPARTS/FORMAT=DBMPARTS PARTS.ROO
```

This command unloads the PARTS database and logs the progress of the unload operation to SYS$OUTPUT. Additionally, it specifies a format file, DBMPARTS.LFL, and a sequence file, DBMPARTS.USL, both of which are in the current default directory.

9.33.2 DBO/UNLOAD/CONTINUE Command

Recovers and continues an interrupted unload operation from a save file.

Format

DBO/UNLOAD/CONTINUE save-file-spec

Command Qualifiers	Defaults
/[NO]LOG	Current DCL verify value
/[NO]QUIET_POINT=integer	
/[NO]SELECTIVE	/NOSELECTIVE

Description

If an unload operation is interrupted for any reason, such as a power failure, you can restart the unload operation with a DBO/UNLOAD/CONTINUE command. Specify the save file as a command parameter. Your original DBO/UNLOAD command writes a save file to support restart unless you specified the /NOSAVE qualifier. As the unload operation progresses, the save file receives information about the unload, such as:

- The DBO/UNLOAD command line, including the root file specification and all qualifiers and qualifier arguments
- The location of the current output record for each output file
- All database currencies

You can update save file information at intervals specified by the /QUIET_POINT qualifier (every 1000 records by default). The unload operation waits while DBO updates the save file. After writing the save file, DBO commits the database with a COMMIT RETAINING statement. Then, DBO continues the unload operation.

When you issue the DBO/UNLOAD/CONTINUE command, specify the save file as the parameter. The save file contains the original DBO/UNLOAD command line as well as currency information. Unless you want to change the settings of the original load operation, you need not use any qualifiers with the DBO/UNLOAD/CONTINUE command. If you want, you can change the values of the /QUIET_POINT and /LOG qualifiers.

Parameter

save-file-spec
Specifies a save file produced by a previous DBO/UNLOAD operation. When you append the /CONTINUE qualifier to a DBO/UNLOAD command, DBO automatically treats the parameter as a save file specification. The default file type is .USV.

Command Qualifiers

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request that each unload operation be logged to SYS$OUTPUT and /NOLOG to prevent this listing. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/QUIET_POINT=integer
/NOQUIET_POINT
Specifies the number of fetch operations that can occur between quiet points. This value is equal to the number of DML FIND commands performed between COMMIT RETAINING commands. The value specified with a DBO/UNLOAD/CONTINUE command supersedes the value in the original DBO/UNLOAD command.

The /NOQUIET_POINT qualifier commits transactions just once at the end of the unload operation.

/SELECTIVE
/NOSELECTIVE
Readies only the required realms with the required mode. The /SELECTIVE qualifier allows you to change back and forth between readying all the realms and only those necessary when you continue the unload.

If the /SELECTIVE qualifier is used with the /LOG qualifier, the unload operation produces an informational message showing which areas are readied in which mode.

Use the /NOSELECTIVE qualifier to ready all realms. This is the default.

Example

```
$ DBO/UNLOAD/CONTINUE/SELECTIVE PARTS
```

This example continues an interrupted unload for the PARTS database by using the save file produced during the initial unload operation.

9.34 DBO/VERIFY Command

Checks and verifies the internal integrity of database data structures and, optionally, the time and date (TAD), path and instance information, version numbers, root file, and the .AIJ file, if after-image journaling is enabled.

Format

DBO/VERIFY root-file-spec

Command Qualifiers	Defaults
/AREAS=area-name [,...]	/AREAS=*
/[NO]CDD	/CDD
/[NO]CONTINUE	/CONTINUE
/END=integer	/END=last-page
/INCREMENTAL	
/INTERVAL_LOG=integer	See description
/[NO]LOG	Current DCL verify value
/ONLINE	
/[NO]OPTIMIZE [=(option[,NAME=set-file])]	/NOOPTIMIZE
/OUTPUT=file-spec	SYS$OUTPUT
/PAGES	
/READY=(allow-mode,access-mode)	
/[NO]ROOT	/NOROOT
/SEGMENTS	
/[NO]SETS[=(set-name [,...])]	/SETS
/START=integer	/START=1
/[NO]SUMMARY	/NOSUMMARY

Description

You should verify your database after any kind of serious system malfunction and as part of routine maintenance. The DBO/VERIFY command checks and verifies the internal structures of a database. It can check database pages, storage segments, records, and set linkages. If space area management (SPAM) is in use, DBO/VERIFY checks SPAM pages for consistency.

In addition, when you specify the /OPTIMIZE qualifier, DBO/VERIFY verifies orphan records and the correctness of the B-tree node entries against the member records. (Orphan records are records that are connected improperly and cannot be reached through their sets, or are records that are not connected at all to their non-optional sets.) See the description of the /OPTIMIZE qualifier for more information.

The optimized verify operation is several times faster than a regular (non-optimized) verify.

The DBO/VERIFY command does not attempt to determine whether data within records is reasonable or plausible.

The DBO/VERIFY command verifies each area and lists the areas containing corrupt or inconsistent data. If an area is marked as corrupt or inconsistent, a message is logged listing each corrupt area, and the verify operation continues. The DBO/VERIFY command can also perform the following in the root file:

- Checks that checksum in the root file equals zero (0).

- Checks that TAD is not equal to 0 and that it is within a valid range.
- Verifies file ID entry for live pages and snapshot pages in each storage area. This entry includes the following checks:
  - Checksumflag is true
  - Checksum equals 0
  - Corrupt flag is not set
  - Verify prologue page for area
  - Detect duplicate area names
- Validates the .AIJ file by performing the following checks:
  - Verify the .AIJ file exists
  - Check for the open record
  - Check for valid TAD
  - Check that block type equals "O"
  - Verify that block (1) is correct
  - Verify that the file name is the same name as in the root file
  - Verify that the TAD is the same TAD as in the root file
  - Verify that the root file name is the same as in the root file

Parameter

root-file-spec
Specifies the root file of the database to verify. The default file type is .ROO.

Command Qualifiers

/AREAS=area-name [,...]
Specifies the areas of the database to verify. The /AREAS qualifier specifies storage areas for verification. Each area-name must be the object of an AREA clause in the schema on which the root file is based. The default, /AREAS=*, causes DBO to verify all storage areas of the database.

The /ROOT and /AREA qualifiers are mutually exclusive.

/CDD
/NOCDD
The /CDD qualifier:

- Checks schema version and database instance information to ensure that the version numbers match
- Checks storage schema version and database instance information
- Ensures schema major and minor version numbers match
- Checks subschema version and database instance information

/CONTINUE
/NOCONTINUE
Specifies whether or not the verification of the database should continue if errors are found. If you specify /CONTINUE, the verification of the database

continues in spite of errors found. The default is /NOCONTINUE. If you specify /NOCONTINUE, the verification of the database stops if errors are found.

/END=integer
Specifies the end of the page range. When verifying a database, you can specify the range of database pages within each area to be verified. By default, a database verification starts with the first page of the database and ends with the last page of the area to be verified. You can override the default using the /START and /END qualifiers to specify the page range.

/INCREMENTAL
Verifies database pages that have changed since the last full or incremental verify. DBO stores timestamps in the root file for both full and incremental verifies. To determine which pages have changed since the last verify operation, DBO compares these timestamps with the page timestamps. An incremental verify operation can take considerably less time than a full verify operation, allowing you to perform incremental verify operations more frequently than full verify operations.

/INTERVAL_LOG=integer
Specifies how often DBO logs verified pages. The value of /INTERVAL_LOG is the number of pages between the log messages that show the progress of the verify operation. This is useful for restarting a verify operation. Specify a positive integer for the /INTERVAL_LOG qualifier to control how often messages are issued during the verify process. If you do not specify an integer, pages are not logged. Use the /INTERVAL_LOG qualifier only in conjunction with the /LOG qualifier.

/LOG
/NOLOG
Specifies whether or not to report the processing of the command to SYS$OUTPUT. Specify /LOG to request that each verify operation be displayed and /NOLOG to prevent this display. If you specify neither, the default is the current setting of the DCL verify option. (The DCL SET VERIFY command controls the DCL verify option.)

/ONLINE
Allows database verification without the need to shut down the database. Because online verification uses snapshots to create a consistent view of the database, snapshots must be allowed and enabled for all storage areas for successful online verification. See DBO/CREATE, DBO/MODIFY, and DBO/RESTORE for information on allowing and enabling snapshots. When you specify the /ONLINE qualifier, users can continue to update the database during the procedure. The verification procedure uses the snapshot file to retrieve the previous version of any record that has been modified since the verification began. When /ONLINE is used, the access mode defaults to /READY=(BATCH,RETRIEVAL).

/OPTIMIZE[=(option[,NAME=set-file])]
Specifies that the DBO/VERIFY command is to walk areas sequentially, extract set information (such as dbkeys, pointer clusters, and B-tree nodes) into set files, sort the set files using the OpenVMS Sort utility and then verify the set files.

The optimized VERIFY command is several times faster than the normal VERIFY command.

The DBO/VERIFY command constructs default set file names using the database name, timestamp, set ID, and area ID. Use the NAME parameter to specify a meaningful name instead of the timestamp in the set file name. The file type is .DBV.

By using options to the /OPTIMIZE qualifier, you can break down the optimized verification operation into two distinct parts:

- /OPTIMIZE=EXTRACT

  This qualifier extracts set information to the set files. It dumps all pointer clusters and B-tree nodes to the set files for the sets specified, or by default, for all sets. Use the NAME parameter if you do not want to use the default set name. See the example in the following section on OPTIMIZE=VERIFY.

- OPTIMIZE=VERIFY

  This qualifier verifies the set files generated by the prior DBO/VERIFY/OPTIMIZE=EXTRACT command. The database is bound very briefly at the beginning, then it is immediately unbound. The verification stage checks the integrity of the set information in the set files.

  Even when you use the default set file name during the extraction phase, you must specify the set file name using the NAME parameter during the verification phase; there is no default set file name in the verification phase.

  If the NAME parameter is not used in the extraction phase, then the timestamp is used to construct the set file name. For example:

```
$ DBO/VERIFY/OPTIMIZE=EXTRACT PARTS
$ DIRECTORY *.DBV

PARTS$009AC3C3A4A0FF84_A4_S12_B.DBV;1
PARTS$009AC3C3A4A0FF84_A4_S13.DBV;1
PARTS$009AC3C3A4A0FF84_A4_S5.DBV;1
   .
   .
   .
$ DBO/VERIFY/OPTIMIZE=(VERIFY, NAME=009AC3C3A4A0FF84) PARTS
```

  If the NAME parameter is used in the extraction phase, then the specified name is used to construct the set file name. That set file name is then used in the verification phase, as follows:

```
$ DBO/VERIFY/OPTIMIZE=(EXTRACT, NAME=FOO) PARTS
$ DIRECTORY *.DBV

PARTS$FOO_A4_S12_B.DBV;1
PARTS$FOO_A4_S13.DBV;1
PARTS$FOO_A4_S5.DBV;1
   .
   .
   .
$ DBO/VERIFY/OPTIMIZE=(VERIFY, NAME=FOO) PARTS
```

If no options are specified to the DBO/VERIFY/OPTIMIZE command, then both extraction and verification phases are done together, which is 10 to 20 percent faster than the separate operations.

If you verify only specified areas, then the areas owned by those areas are also verified. For example, if the area walk encounters an owner record R1 in area A, and R1 owns R2 which exists in area B, then area B is verified along with area A. This could result in some areas being verified more than once if the database is verified by areas. To minimize the duplication, specify areas that are logically grouped together, be processed together.

This feature uses more disk space than a non-optimized verify operation for temporary files during the verification. For a full verification, the space required is typically 10 to 20 percent of a full database backup file size. However, if the database has a very large number of sets, the disk requirements could approach up to 50 percent of the full backup file size.

The following qualifiers of DBO/VERIFY cannot be used with the DBO/VERIFY/OPTIMIZE command:

/END
/START
/INCREMENTAL
/ROOT

Also, the following qualifiers cannot be used with the DBO/VERIFY/OPTIMIZE=VERIFY command:

- /CDD
- /PAGES
- /SEGMENTS
- /READY
- /ONLINE

/OUTPUT=file-spec
Names the file where output will be sent. The default file type is .LIS. The default output is SYS$OUTPUT.

/PAGES
Verifies the database page format of all pages of each area being verified. The VERIFY command performs the following actions with the page headers:

- Checks the range to ensure the page number is within the expected range
- Verifies the area ID of the page is correct
- Verifies checksum
- Checks the TAD to verify it is not equal to zero (0)

The VERIFY command performs the following actions in a line-by-line check:

- Checks the amount of locked and free space
- Checks the SPAM thresholds if the area has SPAMs
- Checks the line index
- Checks the system record ID
- Sorts the line index and checks for overlapping or holes on the page

/READY=(allow-mode,access-mode)
Sets the area lock for the areas being verified. The allow-mode defaults to PROTECTED except for read-only areas, which default to BATCH. The access-mode defaults to RETRIEVAL.

If /ONLINE is used, the access-mode defaults to /READY=(BATCH, RETRIEVAL).

/ROOT
/NOROOT
Specifies whether or not the root file (and the .AIJ file if after-image journaling is enabled) is to be verified. By default, the root file is not verified.

The /ROOT and /AREA qualifiers are mutually exclusive.

/SEGMENTS
Verifies both the database page format and the database storage segments of each area to be verified. If you specify the /SEGMENTS qualifier, the DBO/VERIFY command performs the following checks:

- Checks the SPAM thresholds if the database segment has SPAMs
- Examines the line index and rebuilds all fragmented records
- Verifies the B-tree node, if one exists, or verifies the record database ID
- Maps the record type to the storage area
- Verifies each pointer to the cluster
- Verifies the length of the data portion of each database record

/SETS[=(set-name [,...])
/NOSETS
Verifies the database storage set linkages, as well as the database storage segments and the database page formats. Specify set names when changes affect only a few sets or while verifying database corruption. If you are also using the /AREA qualifier, be sure to list the area from which the sets originate. If you do not specify the area from which the sets originate, those sets are not verified. The default, /SETS, verifies all sets. If you specify the /NOSETS qualifier, the database is verified at the page and segment level.

/START=integer
Specifies the start of the page range. When verifying a database, you can specify the range of database pages to be verified. By default, a database verification starts with the first page of the database and ends with the last page of each area to be verified. You can override this by using the /START and /END qualifiers to specify the page range.

/SUMMARY
/NOSUMMARY
Specifies whether or not a summary message of all errors displays upon completion of the verify operation. Both fatal and informational messages are displayed.

Examples

1. `$ DBO/VERIFY PARTS/AREAS=(MAKE,BUY)`

   This command verifies the pages, segments, and sets of the areas MAKE and BUY in the PARTS database.

2.
```
$ DBO/VERIFY/LOG/NOCDD PARTS
%DBO-I-DBBOUND, bound to database "DBMS$DISK:[WORK]PARTS.ROO;1"
%DBO-I-NOCDDVFY, CDD metadata not being verified
%DBO-I-OPENAREA, opened storage area MAKE for protected retrieval
%DBO-I-PAGERANGE, verification page range 1 to 101 of area MAKE
%DBO-I-BEGINSEGV, beginning segment verification of MAKE area
%DBO-I-ENDSEGV, completed segment verification of MAKE area
%DBO-S-NOPAGERRS,  no page format errors encountered
%DBO-S-NOSEGERRS,  no segment format errors encountered
%DBO-S-NOSETERRS,  no set chain errors encountered

%DBO-E-AREACORRPT, Area DBMS$DISK:[WORK]BUY.DBS;1 is marked as
  corrupt
%DBO-E-AREAINCON, Area DBMS$DISK:[WORK]BUY.DBS;1 is marked as
  inconsistent
%DBO-I-OPENAREA, opened storage area BUY for protected retrieval
%DBO-I-PAGERANGE, verification page range 1 to 101 of area BUY
%DBO-I-BEGINSEGV, beginning segment verification of BUY area
%DBO-I-OPENAREA, opened storage area MARKET for protected
  retrieval
%DBO-I-ENDSEGV, completed segment verification of BUY area
%DBO-S-NOPAGERRS,  no page format errors encountered
%DBO-S-NOSEGERRS,  no segment format errors encountered
%DBO-S-NOSETERRS,  no set chain errors encountered

%DBO-I-OPENAREA, opened storage area MARKET for protected
  retrieval
%DBO-I-PAGERANGE, verification page range 1 to 101 of
  area MARKET
%DBO-I-BEGINSEGV, beginning segment verification of MARKET area
%DBO-I-ENDSEGV, completed segment verification of MARKET area
%DBO-S-NOPAGERRS,  no page format errors encountered
%DBO-S-NOSEGERRS,  no segment format errors encountered
%DBO-S-NOSETERRS,  no set chain errors encountered
%DBO-I-OPENAREA, opened storage area PERSONNEL for protected
  retrieval
%DBO-I-PAGERANGE, verification page range 1 to 101 of
  area PERSONNEL
%DBO-I-BEGINSEGV, beginning segment verification of PERSONNEL area
%DBO-I-ENDSEGV, completed segment verification of PERSONNEL area
%DBO-S-NOPAGERRS,  no page format errors encountered
%DBO-S-NOSEGERRS,  no segment format errors encountered
%DBO-S-NOSETERRS,  no set chain errors encountered
%DBO-I-CLOSAREAS, releasing protected retrieval lock on all
  storage areas
```

   This example verifies all the areas of the PARTS database and lists the areas that are corrupt and inconsistent.

3.
```
$ DBO/VERIFY/SUMMARY PARTS
%DBO-I-SUMPAGERR, 0 page errors encountered
%DBO-I-SUMSEGERR, 0 segment errors encountered
%DBO-I-SUMSETERR, 0 set errors encountered
```

   This example shows the successful verification of the pages, segments, and sets in the PARTS database.

4.
```
$ DBO/VERIFY/LOG/INTERVAL_LOG=10 PARTS
%DBO-I-DBBOUND, bound to database "DBMS$DISK:[USER]PARTS.ROO;2"
%DBO-I-OPENAREA, opened storage area MAKE for protected retrieval
%DBO-I-PAGERANGE, verification page range 1 to 51 of area MAKE
%DBO-I-BEGINSEGV, beginning segment verification of MAKE area
%DBO-I-VERPAGNO, just verified page 10 of MAKE area
%DBO-I-VERPAGNO, just verified page 20 of MAKE area
%DBO-I-VERPAGNO, just verified page 30 of MAKE area
%DBO-I-VERPAGNO, just verified page 40 of MAKE area
%DBO-I-VERPAGNO, just verified page 50 of MAKE area
%DBO-I-ENDSEGV, completed segment verification of MAKE area
%DBO-S-NOPAGERRS,       no page format errors encountered
%DBO-S-NOSEGERRS,       no segment format errors encountered
%DBO-S-NOSETERRS,       no set chain errors encountered
%DBO-I-OPENAREA, opened storage area BUY for protected retrieval
%DBO-I-PAGERANGE, verification page range 1 to 101 of area BUY
%DBO-I-BEGINSEGV, beginning segment verification of BUY area
%DBO-I-OPENAREA, opened storage area MARKET for protected retrieval
%DBO-I-VERPAGNO, just verified page 10 of BUY area
%DBO-I-VERPAGNO, just verified page 20 of BUY area
%DBO-I-VERPAGNO, just verified page 30 of BUY area

                ******  SYSTEM CRASH  ******

$ DBO/VERIFY/LOG/INTERVAL_LOG=10/AREA=BUY/START=30 PARTS
%DBO-I-DBBOUND, bound to database "DBMS$DISK:[USER]PARTS.ROO;2"
%DBO-I-OPENAREA, opened storage area BUY for protected retrieval
%DBO-I-PAGERANGE, verification page range 30 to 101 of area BUY
%DBO-I-BEGINSEGV, beginning segment verification of BUY area
%DBO-I-VERPAGNO, just verified page 30 of BUY area
%DBO-I-OPENAREA, opened storage area MAKE for protected retrieval
%DBO-I-OPENAREA, opened storage area MARKET for protected retrieval
%DBO-I-VERPAGNO, just verified page 40 of BUY area
%DBO-I-VERPAGNO, just verified page 50 of BUY area
%DBO-I-VERPAGNO, just verified page 60 of BUY area
%DBO-I-VERPAGNO, just verified page 70 of BUY area
%DBO-I-VERPAGNO, just verified page 80 of BUY area
%DBO-I-VERPAGNO, just verified page 90 of BUY area
%DBO-I-VERPAGNO, just verified page 100 of BUY area
%DBO-I-ENDSEGV, completed segment verification of BUY area
%DBO-S-NOPAGERRS,       no page format errors encountered
%DBO-S-NOSEGERRS,       no segment format errors encountered
%DBO-S-NOSETERRS,       no set chain errors encountered
%DBO-I-CLOSAREAS, releasing protected retrieval lock on all
  storage areas
```

```
$ DBO/VERIFY/LOG/INTERVAL_LOG=10/AREA=(PERSONNEL,MARKET) PARTS
%DBO-I-DBBOUND, bound to database "DBMS$DISK:[USER]PARTS.ROO;2"
%DBO-I-OPENAREA, opened storage area PERSONNEL for protected
  retrieval
%DBO-I-PAGERANGE, verification page range 1 to 77 of area PERSONNEL
%DBO-I-BEGINSEGV, beginning segment verification of PERSONNEL area
%DBO-I-VERPAGNO, just verified page 10 of PERSONNEL area
%DBO-I-VERPAGNO, just verified page 20 of PERSONNEL area
%DBO-I-VERPAGNO, just verified page 30 of PERSONNEL area
%DBO-I-VERPAGNO, just verified page 40 of PERSONNEL area
%DBO-I-VERPAGNO, just verified page 50 of PERSONNEL area
%DBO-I-VERPAGNO, just verified page 60 of PERSONNEL area
%DBO-I-VERPAGNO, just verified page 70 of PERSONNEL area
%DBO-I-ENDSEGV, completed segment verification of PERSONNEL area
%DBO-S-NOPAGERRS,       no page format errors encountered
%DBO-S-NOSEGERRS,       no segment format errors encountered
%DBO-S-NOSETERRS,       no set chain errors encountered
%DBO-I-OPENAREA, opened storage area MARKET for protected retrieval
%DBO-I-PAGERANGE, verification page range 1 to 46 of area MARKET
%DBO-I-BEGINSEGV, beginning segment verification of MARKET area
%DBO-I-VERPAGNO, just verified page 10 of MARKET area
%DBO-I-VERPAGNO, just verified page 20 of MARKET area
%DBO-I-VERPAGNO, just verified page 30 of MARKET area
%DBO-I-VERPAGNO, just verified page 40 of MARKET area
%DBO-I-ENDSEGV, completed segment verification of MARKET area
%DBO-S-NOPAGERRS,       no page format errors encountered
%DBO-S-NOSEGERRS,       no segment format errors encountered
%DBO-S-NOSETERRS,       no set chain errors encountered
%DBO-I-CLOSAREAS, releasing protected retrieval lock on all
  storage areas
```

In this example, the database has four areas. The interval is set to 10, so after each 10 pages are verified (10, 20, 30, . . .) DBO logs a message. The system crashes after logging the messages for the first area (MAKE) and pages 1–30 of the second area (BUY). The MAKE area and pages 1–30 of the BUY area do not have to be reverified. The verification for the BUY area is restarted at page 30. Then the verification continues for the third (PERSONNEL) and fourth (MARKET) areas.

5.
```
$ DBO/VERIFY/OPTIMIZE=EXTRACT PARTS
%DBO-I-DBBOUND, bound to database "DISK:[USER]PARTS.ROO;1"
%DBO-W-NOCDDINFO, Metadata not loaded into CDD
%DBO-I-OPENAREA, opened storage area MAKE for protected retrieval
%DBO-I-OPENAREA, opened storage area BUY for protected retrieval
%DBO-I-OPENAREA, opened storage area MARKET for protected retrieval
%DBO-I-OPENAREA, opened storage area PERSONNEL for protected retrieval
%DBO-I-PAGERANGE, verification page range 1 to 101 of area MAKE
%DBO-I-BEGINSEGV, beginning segment verification of MAKE area
%DBO-I-OPNSETFIL, opening and building set files walking area sequentially
%DBO-I-CLOSETFIL, closing set files for all sets
%DBO-I-ENDSEGV, completed segment verification of MAKE area
%DBO-S-NOPAGERRS,       no page format errors encountered
%DBO-S-NOSEGERRS,       no segment format errors encountered
%DBO-S-NOSETERRS,       no set chain errors encountered
%DBO-I-PAGERANGE, verification page range 1 to 101 of area BUY
%DBO-I-BEGINSEGV, beginning segment verification of BUY area
   .
   .
   .
%DBO-I-CLOSAREAS, releasing protected retrieval lock on all storage areas
```

```
$ DIR *.DBV
PARTS$009A9E1E3A618D6B_A1_S1.DBV;1
PARTS$009A9E1E3A618D6B_A1_S13.DBV;1
PARTS$009A9E1E3A618D6B_A1_S2.DBV;1

$ DBO/VERIFY/OPTIMIZE=(VERIFY, NAME=009A9E1E3A618D6B) PARTS
%DBO-I-DBBOUND, bound to database "DISK:[USER]PARTS.ROO;1"
%DBO-I-SRTSETFIL, merging and sorting set files for set ALL_CLASS
%DBO-I-VERSETFIL, verifying set files for set ALL_CLASS
%DBO-S-NOSETERRS,       no set chain errors encountered
%DBO-I-SRTSETFIL, merging and sorting set files for set ALL_PARTS
%DBO-I-VERSETFIL, verifying set files for set ALL_PARTS
%DBO-S-NOSETERRS,       no set chain errors encountered
   .
   .
   .
%DBO-I-VERSETFIL, verifying set files for set RESPONSIBLE_FOR
%DBO-S-NOSETERRS,       no set chain errors encountered
```

This example uses the /OPTIMIZE qualifier to perform an optimized verification on the PARTS database in two steps. The first DBO command performs the extract phase of the optimized verification. The DCL DIRECTORY command is issued to find the default name for the set files associated with this operation. The first portion of the file name following the PARTS$ prefix is used as an argument to the NAME qualifier in the last DBO/VERIFY command. This last command verifies the set files generated by the previous DBO/VERIFY/OPTIMIZE=EXTRACT command. Equivalent results could have been achieved in a one-step process by issuing just the DBO/VERIFY/OPTIMIZE command, without any arguments to the /OPTIMIZE qualifier.

9.35 DBO/WORK_AREA Command

Translates subschema syntax into data definition syntax for a variety of commonly used application programming languages.

Format

DBO/WORK_AREA schema-name [subschema-name]

Command Qualifiers	Defaults
/LANGUAGE=type	/LANGUAGE=MACRO
/RECORD	

Parameter Qualifiers	Defaults
/[NO]DECLARATIONS [=file-spec]	/DECLARATIONS
/INVOKE_ROOT [=file-spec]	
/OUTPUT=file-spec	SYS$OUTPUT
/ROOT=root-file-spec	
/STREAM [=stream-number]	

Description

The DBO/WORK_AREA command is a tool to help application programmers match data structures in their programs to database records. It saves application programming time and ensures accurate data structure mapping between the database and the program.

The subschema must belong to the schema you specify. You can specify only one target language per DBO/WORK_AREA command, and DBO translates the subschema to the data definition syntax for that language. This translated version of a subschema is called a user work area (UWA). In addition, you can specify a stream number for the UWA and specify the root file to which the UWA will be bound.

Note the DML precompiler supports programs written in the following programming languages: Ada, BASIC, BLISS, C, COBOL, DIBOL, Pascal, and PL/I. The DML precompiler automatically creates work areas. Use the DBO/WORK_AREA command to create a UWA only when you have either written your program in MACRO or used callable Database Query (DBQ) routines without also using embedded DML statements in the same program.

The UWA contents are different on OpenVMS Alpha and OpenVMS VAX. Application programs generated for use with OpenVMS Alpha must use a UWA generated from DBO/WORK_AREA running on an OpenVMS Alpha system. Likewise, application programs generated for use on OpenVMS VAX must use a UWA generated from DBO/WORK_AREA running on an OpenVMS VAX system.

Parameters

schema-name
Specifies the schema to which the subschema belongs.

subschema-name
Specifies a subschema for which a UWA will be produced. The default is DEFAULT_SUBSCHEMA.

Command Qualifiers

/LANGUAGE=type
Specifies the programming language into which you want the subschema source translated. The type argument must refer to one of the following source programming languages:

Ada
BASIC
BLISS
CC (refers to C)
COBOL
DIBOL
MACRO
Pascal
PLI (refers to PL/I)

MACRO is the default language. For all languages except MACRO, the output is provided in a form that can be inserted into a program by the language compiler using a REQUIRE or INCLUDE commands.

For MACRO, the default output is a single macro definition named DBMUWA, which can be inserted into a macro library.

The language you specify does not necessarily support all data definitions in the subschema. If a subschema contains data types and scaling factors not compatible with the programming language, the output will contain errors. Such errors do not prevent the use of the UWA, but application programs must not address the incompatible data.

Chapter 7 includes complete tables outlining data restrictions for the various programming languages.

/RECORD
Generates the UWA in BASIC record structures (instead of the unique ITEM name method). Specify this qualifier only in conjunction with the qualifier /LANGUAGE=BASIC. If you omit the /RECORD qualifier when specifying BASIC, the unique ITEM method is used.

Parameter Qualifiers

/DECLARATIONS [=file-spec]
/NODECLARATIONS
Specifies how to treat declarations for DBQ routines and status values.

To include declarations in the /OUTPUT file, use the /DECLARATIONS qualifier without a file specification. This is the default.

To write declarations to a separate file, specify /DECLARATIONS=file-spec. The /OUTPUT file receives the rest of the UWA but not the declarations.

To exclude declarations from the UWA, specify /NODECLARATIONS.

/INVOKE_ROOT [=root-file-spec]
Specifies the root file to be used when translating subschema syntax into data definition syntax. If a root-file-spec is not specified, then the /ROOT=root-file-spec qualifier will be used to specify the target database root file. If the /ROOT qualifier is not supplied, then the schema-name parameter is used as the root file name.

Another way to direct the UWA to be fabricated from the target database root file rather than from the dictionary is to define the DBM$DML_INVOKE_ROOT logical name. When the logical name is defined, the schema-name parameter is used as the name of the target database root file. If the schema-name is different from the target root file, then the DBM$DML_INVOKE_ROOT logical name cannot be used. In these cases, the /INVOKE_ROOT qualifier must be used.

See the *Oracle CODASYL DBMS Programming Reference Manual* for a description of this and other Oracle CODASYL DBMS logical names.

/OUTPUT=file-spec
Names the file where output will be sent. The default output is SYS$OUTPUT. The default file type depends on the language, as shown in the following table.

Programming Language	File Type
Ada	.ADA
BASIC	.BAS
BLISS	.B32
COBOL	.COB
CC	.C
DIBOL	.DBL

Programming Language	File Type
MACRO	.MAR
Pascal	.PAS
PL/I	.PLI

/ROOT=root-file-spec
Specifies the root file to which the UWA will be bound when you also specify /STREAM. If you do not specify /ROOT with /STREAM, DBO uses the schema name as the default root file name.

You can use /ROOT only if /LANGUAGE is BLISS, C, MACRO, PL/I, or Pascal.

DBO ignores the /ROOT qualifier when you do not specify /STREAM.

/STREAM [=stream-number]
Identifies the UWA with a stream name.

The /STREAM qualifier, in combination with the /ROOT qualifier, identifies the pairing of a root file with a subschema. It gives a UWA a stream name for DML programs using functions that require identification of UWAs. This need arises in programs that use multiple streams or DBQ functions that refer to stream names.

You can use /STREAM only if /LANGUAGE is BLISS, C, MACRO, PL/I, or Pascal.

If you omit /STREAM, the UWA produced for the subschema will have no stream (the default UWA) and the user cannot bind to multiple databases.

If you specify /STREAM without an explicit stream number, the stream name defaults to the subschema's time and date stamp. Therefore, all users of the subschema have the same UWA.

If you specify /STREAM with a stream number (/STREAM=stream-number), the number you specify is the basis of the UWA's stream name.

Example

```
$ DBO/WORK_AREA PARTS -
_$ PARTSS4/LANGUAGE=PASCAL/OUTPUT=UWADEF.PAS /STREAM=27/ROOT=NEWPTS
```

This command translates a subschema into Pascal source. At run time, the UWA stream name is based on the number 27. Using streams allows a user to perform multiple bind operations.

10

Load/Unload Utility Commands

The Load/Unload utility consists of the following components:

- DBO/LOAD command
- DBO/UNLOAD command
- Load/unload format language (FORMAT)
- Load/unload sequence language (SEQUENCE)

This chapter describes the syntax of FORMAT and SEQUENCE files. See also Chapter 9 for the syntax of the DBO/LOAD and DBO/UNLOAD commands. See also the *Oracle CODASYL DBMS Database Load/Unload Guide* for examples of how to use the Load/Unload utility.

The FORMAT file describes the data to be exchanged between database records and OpenVMS Record Management Service (RMS) files. The SEQUENCE files determine the order of the exchange.

A DBO/LOAD or DBO/UNLOAD command must always use a FORMAT file and a SEQUENCE file. You can specify these with the /FORMAT and the /SEQUENCE qualifiers.

Loading or unloading is accomplished through a subschema view, which you can specify with the /SUBSCHEMA qualifier or default to DEFAULT_SUBSCHEMA. The contents of the FORMAT file, the SEQUENCE file, and the subschema view determine the outcome of the load or unload operation.

If you have DEC Language-Sensitive Editor (LSE) installed on your system, you can use LSE templates to write load sequence files and load format files.

There are LSE templates available for the load format language (LFL) and the load sequence language (LSL). To invoke LSE, type the LSEDIT command followed by the name of the file you want to edit:

```
$ LSEDIT filename.LFL
$ LSEDIT filename.LSL
```

If you use the .LFL or .LSL file types, the template is automatically invoked for the editing session.

Once you are using the template, a placeholder appears on the first line of the editing buffer. A placeholder is text that can be expanded into code.

In the LFL template, the placeholder is:

```
[RECORD | SET]...
```

In the LSL template, the placeholder is:

```
[sequence_entry]...
```

See the VMS Language-Sensitive Editor and Source Code Analyzer User Manual and *Introduction to Oracle CODASYL DBMS* to learn more about using LSE.

10.1 Load/Unload Format Language

The Load and Unload utilities use FORMAT files to describe data. A load/unload format language has four types of entries:

- RECORD entry
- ITEM (subordinate to RECORD) entry
- SET entry
- MEMBER (subordinate to SET) entry

10.1.1 RECORD Entry

Defines a load/unload record and specifies the format of each record. Each record to be loaded/unloaded must be defined.

Format

```
RECORD NAME IS record-name
  [ NOT STORED ]
  [ FOR subschema-record-name ]
  [ IN data-file-spec ]
  [ item-entry ]...
```

Arguments

RECORD NAME IS record-name
Identifies the input record to be loaded or output record to be unloaded.

NOT STORED
Indicates to the Load/Unload utility that the record is not in the subschema view. The Load utility cannot store the record; the Unload utility cannot unload the record directly, but creates it while unloading other records in order to use it for a later reload. The NOT STORED clause can identify set-significant records or create new records. Set-significant records are key data items in OpenVMS RMS files that establish correct database set relationships.

FOR subschema-record-name
Identifies the subschema record that corresponds to this input or output record. If no subschema record name is specified, the default subschema record is the same name as the input or output record.

IN data-file-spec
Specifies a load input or unload output file containing occurrences of this record type. If no data-file-spec is specified, the default file specification consists of the default device and directory defined by the global /FILE qualifier, followed by the first 9 alphanumeric characters of the record name and the file type .DAT.

10.1.2 ITEM Entry

Defines a data item for a load/unload RECORD entry. All items to be loaded/unloaded must be defined.

Format

```
ITEM NAME IS input-item
  TYPE IS data-type
  [ NOT STORED ]
  [ FOR subschema-item-name ]
  [ OCCURS integer TIMES ]
```

Arguments

ITEM NAME IS input-item
Identifies the input item to be loaded or output item to be unloaded.

TYPE IS data-type
Describes the format of each item in the input or output record. Data types are the same in FORMAT language as in the data definition language (DDL).

NOT STORED
Indicates that the item does not reside in the database. A NOT STORED item does not have a corresponding subschema item. Use this to identify set-significant items or to add new items to unloaded records.

FOR subschema-item-name
Identifies the subschema item that corresponds to this input or output item. If you specify no subschema item name, the default subschema item has the same name as the input or output item.

OCCURS integer TIMES
Describes repeating items. A stored item must occur the same number of times in a FORMAT file as it does in the subschema, schema, or both.

10.1.3 SET Entry

Names a set for the SEQUENCE file and specifies set order and insertion information.

Format

```
SET NAME IS set-name
[ OWNER IS { SYSTEM
           { owner-record-name } ]
[ member-entry ]...
```

Arguments

SET NAME IS set-name
Identifies a set that must be in the database subschema.

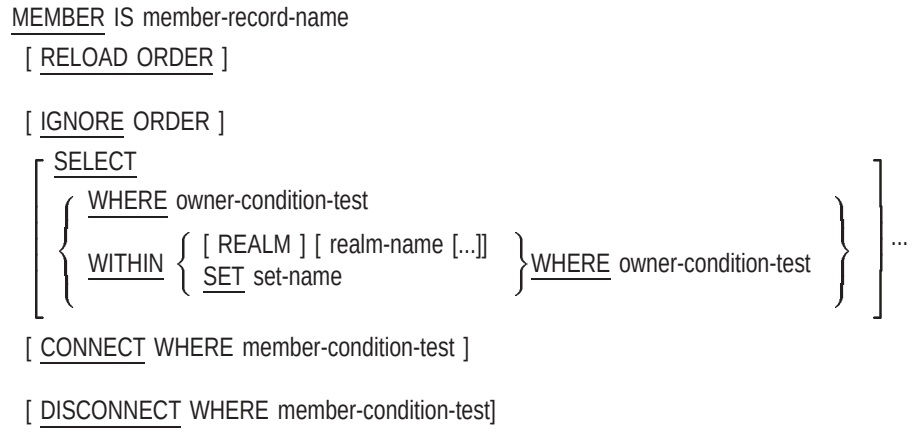
OWNER IS SYSTEM
Identifies the SYSTEM as the owner of the set.

OWNER IS owner-record-name
Identifies the owner record of the set. Must be in the database subschema. The OWNER clause need not be specified.

10.1.4 MEMBER Entry

Names a member for the SEQUENCE file and specifies set order and insertion information. All arguments except the RELOAD ORDER clause pertain to the Load utility only.

Format



Arguments

MEMBER IS member-record-name
Identifies the member records of the set (which must be in the database subschema). The member record must be mentioned if a SELECT, CONNECT, DISCONNECT, RELOAD ORDER, or IGNORE ORDER clause is used for it. Otherwise, the member clause is optional if you defined the member in a prior RECORD entry.

RELOAD ORDER
Indicates that the order of each member within the set should be preserved during an unload operation. This is useful only for ORDER IS FIRST or PRIOR and DUPLICATES ARE FIRST or ALLOWED.

If RELOAD ORDER is specified, record occurrences are unloaded in a PRIOR direction to the OpenVMS RMS file. This allows the original set order to be established again during a subsequent load operation. The RELOAD ORDER clause is effective only for a set through which you unload the member records.

IGNORE ORDER

Indicates that the initial order of members within the set can be ignored during a load operation. This is useful for ORDER IS NEXT or PRIOR, both of which are based on set currency. The IGNORE ORDER argument is also useful when it is used with the set member whose owner is selected by the SELECT clause. Because load selects the owner of the set, it must do an extra find operation to establish proper set currency.

If IGNORE ORDER is specified, schema order of NEXT results in actual order of FIRST, and schema order of PRIOR results in actual order of LAST.

SELECT

Selects an owner record for the record being loaded by using its WHERE expression. You can use the SELECT clauses for each non-SYSTEM-owned set to find owner records for the member records you are loading. The owner records must be already stored.

If you do not use the SELECT clauses, all member records are connected to the current owner record. This is not a problem if you are loading owner and member records hierarchically. However, if owner records are already stored in the database, SELECT clauses are necessary to find the correct owner record occurrences for the member records you are loading separately.

For an automatic set, the SELECT clause always locates an owner record. For a MANUAL set, the SELECT clause locates an owner record only when the CONNECT WHERE expression is true.

If you moved the value of an item from an owner record to a member record using the unload MOVE clause in a previous unload operation, then you can now reconnect the member record to its owner record using the SELECT and CONNECT clauses with the moved item.

The CONNECT WHERE expression checks if the value of the moved item is not zero to determine if the record was a member record. The SELECT WHERE expression then uses the item to locate the owner record.

SELECT clauses construct database find operations out of the set-significant information you supply in an owner condition test. There are two types of SELECT clauses:

- Those that walk a set or hierarchy (SELECT WITHIN SET clauses)
- Those that scan one or more database areas (SELECT and SELECT WITHIN REALM clauses)

Use SELECT WITHIN SET wherever possible to optimize the performance of your SELECT clauses.

SELECT clauses can be used singly or in combination. Different rules apply when you use a single SELECT clause than when you use a list of SELECT clauses, as discussed next.

WHERE owner-condition-test

Evaluates set-significant data items to construct a database find operation.

If you are using a single SELECT clause, the owner condition test selects the owner record occurrence for the member record being loaded.

If you are using a list of SELECT clauses, the first owner condition test selects the owner record occurrence for the member record being loaded. The next owner condition test selects the owner record occurrence for the first SELECT clause's set. The next SELECT clause selects the owner of the second SELECT clause's set, and so on, until the top of the hierarchy is reached. Multiple owner condition tests provide the Load utility with a path to follow within a set hierarchy.

Although you list the hierarchy in reverse order, the Load utility traverses the hierarchy from the top down.

One data item in each owner condition test must be a stored item of the record being found. Any other items in the condition test might be stored items of the record being found or any items of the member record being loaded.

You may have to add set-significant items to your member records specifically for use in multiple-level owner condition tests.

An owner condition test can contain only EQ and AND operators.

SELECT WITHIN REALM realm-name WHERE
SELECT WHERE

Scans one or more database areas to find an owner for the member. If the SELECT or SELECT WITHIN REALM is the only SELECT clause you use, the Load utility performs a sequential scan of the realm or realms in which the owner record is located.

The SELECT clause scans those realms that can contain the record, while the SELECT WITHIN REALM clause lets you specify the realm or realms to scan for the owner record.

Only one SELECT or SELECT WITHIN REALM clause may be used for any member entry. If you use a combination of SELECT clauses, the SELECT or SELECT WITHIN REALM must be the last SELECT WITHIN clause in the list. This clause is used to locate the top record of the hierarchy.

SELECT WITHIN SET set-name WHERE

Identifies a set or hierarchy within which to find an owner for a member.

A single SELECT WITHIN SET clause names the set within which to find the owner of the member being loaded.

A single SELECT WITHIN SET clause must name a system-owned set, or use the set through which the load operation is performed. If such a set is not established, you get an error message.

Multiple SELECT WITHIN SET clauses identify a hierarchy within which to find the owner of the member being loaded.

You can list as many SELECT WITHIN SET clauses as there are levels in the hierarchy you are tracing. SELECT WITHIN SET clauses must be listed in reverse hierarchical order. If a SELECT WITHIN SET clause names a system-owned set, it must be the last SELECT WITHIN clause in the list.

Use the shortest hierarchical path possible so that the Load utility has to find the least number of records in order to locate the correct owner record occurrence. For example, a single SELECT WITHIN SET clause that uses a system-owned set is an optimal use of the SELECT WITHIN SET clause.

CONNECT
Connects a member to a set with manual insertion mode. See the description for the SELECT clause for additional information.

DISCONNECT
Disconnects a member from a set with AUTOMATIC INSERTION and OPTIONAL RETENTION modes.

CONNECT WHERE
Performs a test on the member record you are loading to determine whether or not it should be inserted in the set. The set must have MANUAL INSERTION mode.

Member records are connected to the current owner record as established by sequence or by SELECT clause.

DISCONNECT WHERE
Tests the member record you are loading to determine whether or not it should be disconnected from the set. The set must have an AUTOMATIC INSERTION mode and an OPTIONAL RETENTION mode.

Member records are disconnected from the current owner record as established by sequence or by SELECT clause.

member-condition-test
Denotes which data values of the member record are used to connect the record to a set or disconnect the record from a set. The member condition test is defined to be a full Boolean expression.

The member condition test can reference any item of the member record being loaded.

If no member condition test is specified, all members are connected or disconnected.

10.2 Load/Unload Sequence Language

The Load and Unload utilities use SEQUENCE files to describe the order of data. The load/unload sequence language has three types of entries:

- SEQUENCE NAME entry
- LOOP entry
- Sequence RECORD entry

10.2.1 SEQUENCE NAME Entry

Defines a procedural order for loading or unloading records. The format of a SEQUENCE NAME entry is the same for either a load or an unload sequence.

SEQUENCE NAME Entry

Format

```
SEQUENCE NAME IS sequence-name { loop-entry              }...
                               { sequence-record-entry   }
```

Arguments

SEQUENCE NAME IS sequence-name
Groups together multiple sequence LOOP and record entries. A SEQUENCE file must contain at least one SEQUENCE NAME and RECORD entry and can contain any number of SEQUENCE NAME and record or LOOP entries.

10.2.2 LOOP Entry

Loads or unloads a hierarchy. You can nest loops to load or unload nested hierarchies. The format of a LOOP entry is the same for either a load or unload sequence.

Format

```
LOOP
 [EXIT ON record-name]

 [EXHAUSTIVE]
 { loop-entry              }...
 { sequence-record-entry   }
ENDLOOP
```

Arguments

EXIT ON record-name
Names a RECORD entry within the LOOP that terminates the loop. The EXIT ON clause terminates a loop when the indicated RECORD entry fails to load or unload a record. Use the EXIT ON clause only when you want the loop to exit on a RECORD entry other than the first.

EXHAUSTIVE
Indicates that the LOOP will continue processing until each of the RECORD entries within it fails to load or unload a record.

Use an EXHAUSTIVE clause to load or unload within the same loop records that have unequal numbers of occurrences and records that do not have a hierarchical association.

loop-entry
Nested loop entries are allowed.

sequence-record-entry
See Section 10.2.3 for a description.

A LOOP entry without an EXHAUSTIVE or EXIT ON clause is nonexhaustive. The LOOP entry will continue loading or unloading records only as long as the first RECORD entry within it successfully loads or unloads a record.

10.2.3 Sequence RECORD Entry

Identifies one record in the sequence. See Section 10.2.3.1 for load sequence record entry and Section 10.2.3.2 for unload sequence record entry.

10.2.3.1 Load Sequence RECORD Entry

Identifies one RECORD entry in the load sequence. Each sequence entry represents a record in the current hierarchy.

Format

```
{ ONE  } record-name
{ ALL  }

[ WHILE condition-test ]

[ WITHIN {realm-name [ WHERE condition-test ]} ... ]

[ LINK {record-name TO {set-name [...]} WHERE condition-test} ... ]
```

Arguments

ONE
ALL record-name
Identifies the record to be loaded.

WHILE condition-test
Gives an end condition for loading multiple records.

If ONE is specified, then only one of the records is loaded at this time. The record chosen is the next record in the file. If ALL is specified, the next record in the file is loaded as long as it satisfies the condition test.

You can specify a condition test for each record to select the record occurrences to be loaded. The condition test can reference the current record being loaded, as well as the current record for any other record type. This allows set membership to depend on other records as well as the owner and member.

A current record is the most recent occurrence loaded for a given record type.

WITHIN realm-name WHERE condition-test
Identifies the realm in which to load the record. The default realm is the first realm in the corresponding schema record's WITHIN clause.

You can use a condition test for each realm to determine which occurrences of the record to store in that realm. The condition test can mention any current record.

LINK record-name TO set-name WHERE condition-test
Links records into manual sets where the member record occurrences are stored before the owner record occurrences. When the owner is stored first, use the SELECT clause in the load format language.

The record name identifies the member record to be linked to the sets specified in the set name. The record must be listed as a member of the specified sets in the subschema definition. Use the LINK clause to find member records that have been previously stored in the database.

A condition test must be used for each record to determine which occurrences of the member records should be found in the database. The condition test must mention at least one stored item of the member record being found, and may mention any items of the owner record being loaded.

WHILE condition-test
WHERE condition-test
Denotes which data values of the record are used to select particular record occurrences or to select the area in which to store the record. The condition test is defined to be a full Boolean expression.

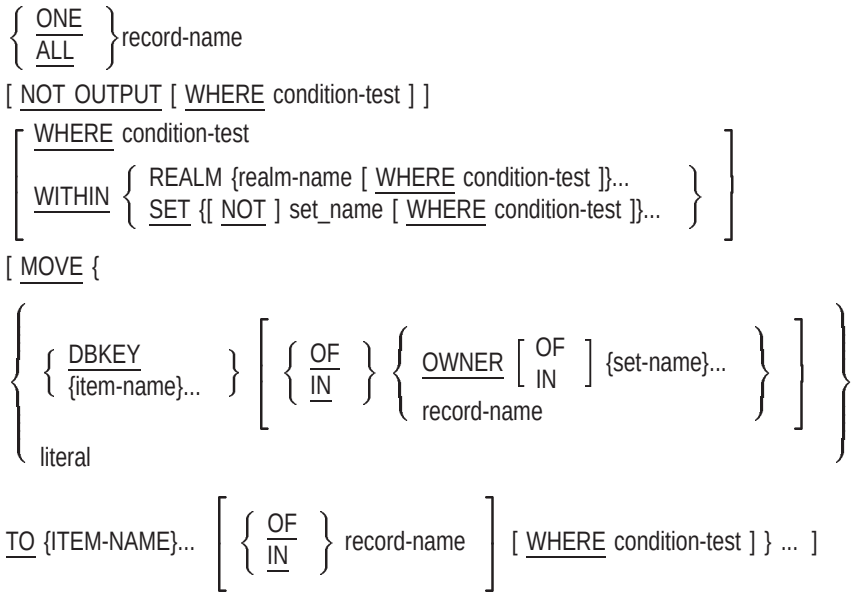
The items referenced by the condition test are from the load record format. The items do not necessarily exist in the subschema record definition.

The items used in the condition test may or may not be stored. Items are from the current record, that is, from the most recently loaded occurrence.

10.2.3.2 Unload Sequence RECORD Entry

Identifies one RECORD entry in the unload sequence. Each sequence entry represents a record in the current hierarchy.

Format



Arguments

ONE
ALL record-name
Identifies the record to be unloaded.

If ONE is specified, then only one of the records gets unloaded at this time. The record chosen is the next record that satisfies the WHERE condition test. If ALL is specified, the next record is unloaded as long as it satisfies the WHERE condition test. If no WHERE condition test is specified, the next record within the database is unloaded.

WHERE condition-test
Unloads only specified record occurrences at a time. The WHERE condition test gives a condition for unloading multiple records. If a global WHERE is used, the WITHIN clause cannot be used.

WITHIN REALM
WITHIN SET
Identifies the realm from which to unload the record. If the realm is not specified, it defaults to all realms in the subschema that contain the record.

The SET clause allows a set selection expression that denotes which set to walk to unload the records. The first set in the set name mentioned without a NOT is walked, while the record's membership in any other set is evaluated with an IF MEMBER test. A record is unloaded only if it is a member of each of the other sets on the list. When NOT is used in front of a set, the record is unloaded only if it is not a member of the set.

A WITHIN clause refers only to the current record.

A sequence RECORD entry cannot contain both SET and REALM clauses.

You can use a WHERE condition test for each realm or set name to specify which occurrences of the record to unload. The condition test must mention at least one item in the record being unloaded.

If the WITHIN REALM or WITHIN SET clause is used, the global WHERE clause cannot be used.

NOT OUTPUT [WHERE condition-test]
Indicates that the record is available for pass-through access but should not be written to the output file.

Use a NOT OUTPUT clause to read records that have been already unloaded without unloading them again.

Items on NOT OUTPUT records are available for use with the MOVE clause.

A WHERE condition test selects occurrences of the record for output. The condition test might mention any current record.

MOVE ... TO {item-name}... WHERE condition-test
Assigns data values to items in the output format that were not stored in the database. The item values of the target record (moved to) must be of the record specified by the ONE or ALL clause.

The source record items (moved from) might be:

- A literal value

- The dbkey of the current record or of the owner record in a set hierarchy
- One or more item values of the current record or of the owner record in a set hierarchy

In a set hierarchy, the current record is a member of the first set, and the owner of the first set is a member of the second set, and so on. If a set hierarchy is specified, the owner of the last (topmost) set is selected as a source record for MOVE.

If more than one item is moved, there must be the same number of source and target items.

A WHERE condition test can specify records to which you want to assign new item values. The condition test might mention items on any current record.

If an unload sequence file specifies a MOVE clause to move an item from an owner record to a member record and the member is not connected to the owner, then the item is filled with zeroes, regardless of the data type.

11

DBALTER Utility Commands

This chapter describes the DBALTER utility for Oracle CODASYL DBMS. DBALTER provides a low-level patch capability for Oracle CODASYL DBMS databases.

Use DBALTER only after you fully understand the internal data structures, know the information the database should contain, and know the full effects of the DBALTER commands. Because of the power of DBALTER and the cascading effects it can have, Oracle recommends that you experiment on a copy of the damaged database with logging enabled.

To enter the DBALTER utility, enter the following:

```
DBO/ALTER [root-file-spec]
```

The optional root file parameter identifies the database you wish to alter. If you specify this parameter, you automatically bind to the specified database. If you do not specify this parameter, you must use the DBALTER BIND command. See Section 11.2 for more information on the BIND command.

The DBO/ALTER command responds with the following prompt:

```
DBALTER>
```

This prompt means that the system expects DBALTER command input. The DBALTER commands are:

AREA . . . PAGE	HELP	RADIX
BIND	LOG	ROLLBACK
COMMIT	MAKE_CONSISTENT	UNBIND
DEPOSIT	MOVE	UNCORRUPT
DISPLAY	NOLOG	UPDATE CHECKSUM
EXIT	PAGE	VERIFY

When you enter DBALTER and the database needs to be recovered, DBALTER displays a message to that effect. If you receive this message, you should recover the database and verify it again before making any changes in DBALTER. If you happen to make changes in DBALTER before recovering the database, there is a chance that the recovery procedure will write over any changes you have made.

The remainder of this chapter describes the full syntax and use of these commands. For detailed information on using the DBALTER utility, see the *Oracle CODASYL DBMS Database Maintenance and Performance Guide*.

11.1 AREA . . . PAGE Command

Specifies either a storage area or a storage page, or both, of the current bound database that you intend to alter.

Only one page of one area is accessible to DBALTER at a time. This command switches you from one page of the currently bound database to another.

When you bind to a database, DBALTER automatically makes Area 1 the current area and Page 1 of that area the current page. To work on any other area and page, you must use the AREA . . . PAGE command.

If you specify AREA but not PAGE, DBALTER makes the area you specify current and fetches Page 1 of that area.

If you specify both AREA and PAGE, DBALTER makes the area you specify current and fetches the page you specify from the new current area.

If you specify an area or page that does not exist, an error occurs.

Format

```
AREA { storage-area-name   }    [ PAGE page-number ]
     { storage-area-number }
```

Arguments

storage-area-name
Specifies an area of the current database by the storage area name, which is the name given by the AREA clause in the schema.

storage-area-number
Specifies an area of the current database by the storage area number, which is assigned when the database is created and is given on the first line of a page display.

page-number
Identifies the area page to be altered. Express it as an integer from 1 to the number of pages in the area.

Suppose you have been altering area 2, page 23 and now want to work on some other area or page of the database. The following examples show some possible commands.

Examples

1. `DBALTER> AREA BUY`

   This example specifies the area BUY of the currently bound database. No page number has been specified, so DBALTER fetches page 1 of the area.

2. `DBALTER> AREA 4`

   This example is similar to the previous example, except it specifies the area by its area number instead of its area name. If area 4 is the area named BUY in the schema, either command produces the same result.

3. DBALTER> AREA 3 PAGE 100

   This example fetches area 3, page 100.

11.2 BIND Command

Binds DBALTER to a database, putting an EXCLUSIVE UPDATE lock on the database. No other user can access the database while DBALTER BIND is in effect.

The database specified by root-file-spec is bound to DBALTER. Then you can alter the pages of its storage area files. Area 1, page 1 of the database is fetched automatically and remains the current page until you issue an AREA . . . PAGE command.

If the DBO/ALTER command includes a root-file-spec parameter, the database to which the root-file-spec refers is bound as part of DBALTER startup. In this one case, the BIND command is unnecessary. Otherwise, no commands changing database pages are allowed until a BIND command naming that database is issued.

Only one database BIND can be active at one time. Before binding to another database for altering, you must UNBIND from the current one.

Format

BIND [root-file-spec]

Argument

root-file-spec
Specifies the root file of the database whose pages you wish to alter. The default file type is .ROO. It must be enclosed in quotation marks (" ") if it contains non-alphanumeric characters.

Examples

1.
```
$ DBO/ALTER
DBALTER> BIND PARTS
%DBO-I-DBBOUND, bound to database "DISK:[USER]PARTS.ROO;1"
```

   This example enters DBALTER command level, then binds to the PARTS database.

2.
```
$ DBO/ALTER PARTS
%DBO-I-DBBOUND, bound to database "DISK:[USER]PARTS.ROO;1"
```

   This example enters DBALTER command level and binds to the PARTS database in a single command.

11.3 COMMIT command

Writes all page changes back to the database since the last COMMIT or ROLLBACK command was entered.

The results of DEPOSIT and MOVE commands are kept in virtual memory until you enter COMMIT or ROLLBACK. When you enter COMMIT, all pages that you changed since the last COMMIT or ROLLBACK command are written to the database in their new form.

Changes are not permanent until you issue a COMMIT command. If you issue an EXIT command while altered but uncommitted pages exist, an error results. EXIT is not allowed until you first issue either a COMMIT or a ROLLBACK command.

Format

COMMIT

Example

```
DBALTER> COMMIT
```

This example commits all page changes entered since the last COMMIT or ROLLBACK.

11.4 DEPOSIT Command

Alters specified data fields on the current database page.

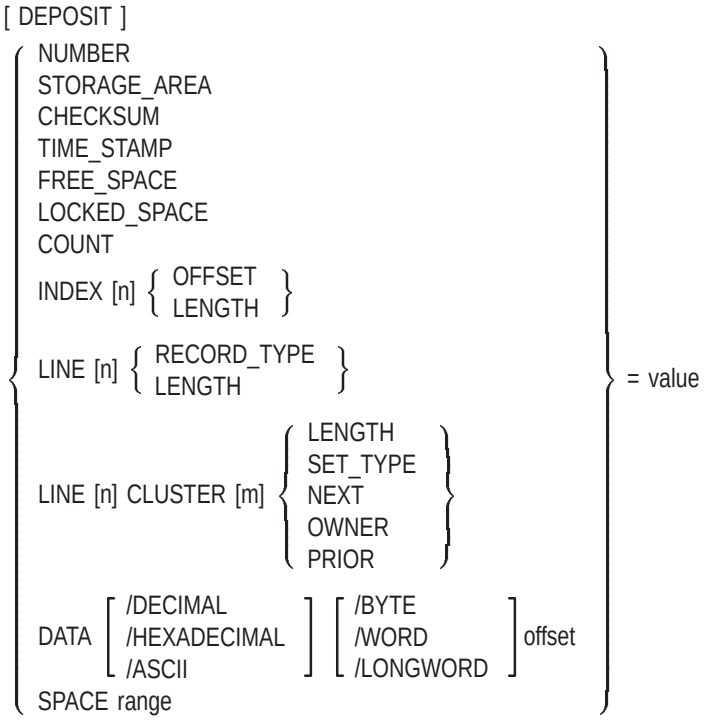
DEPOSIT is the default at DBALTER command level. If no other command is present following the prompt, DBALTER automatically parses the command line as a DEPOSIT.

Specify the field to be patched just as you would specify a field to be displayed in the DISPLAY command.

The patch field specification must be followed by an equal sign (=) and a string of characters specifying the new value of the patched field.

Do not use the DEPOSIT command immediately after a ROLLBACK command. The ROLLBACK command removes currency indicators. For this reason you must respecify your location by using either the DISPLAY or AREA . . . PAGE command immediately after a ROLLBACK command, but before a DEPOSIT command.

Format



Arguments

NUMBER
Deposits a value for the 4-byte page number field.

STORAGE_AREA
Deposits a value for the 2-byte storage area identification.

CHECKSUM
Deposits a value for the 4-byte page checksum field.

TIME_STAMP
Deposits a value for the 8-byte time-and-date-stamp field.

FREE_SPACE
Deposits a value for the 2-byte field indicating how much free space remains on the page.

LOCKED_SPACE
Deposits a value for the 2-byte field indicating how much free space is allocated for exclusive use by a run unit.

COUNT
Deposits a value for the 2-byte field showing the number of line index entries. If this number is 1, the page contains only the SYSTEM record.

Note

In the next three arguments, the integers denoting INDEX, LINE, and CLUSTER are zero based. For example, INDEX 0 refers to the first index, and LINE 3 CLUSTER 2 refers to the third cluster of the fourth line.

The integers *n* and *m* are optional. The present value of the relevant pointer is the default in each case.

References to INDEX, LINE, and CLUSTER are invalid if the current page is a SPAM page.

INDEX n
Deposits a value for the offset field, the length field, or both for the line index indicated by *n*. For example, if you enter DEPOSIT INDEX 3 OFFSET, the offset address field from the fourth line index is deposited.

LINE n
Deposits information for an individual storage segment. You can deposit a value for the record-type field, the pointer cluster length, or the entire content of the storage segment line indicated by *n*. Each storage segment contains a set of pointer clusters.

LINE n CLUSTER m
Deposits information from an individual pointer cluster.

DATA offset
Deposits 1, 2, or 4 bytes of data in the radix specified. If you do not specify a radix, the default radix is assumed. See the description of the RADIX command in this chapter for information on how to set a default radix.

SPACE range
Deposits a value for a specified range of SPAM entries; it is valid only if the current page is a SPAM page. SPACE is the only option that you can use in DISPLAY and DEPOSIT commands that access a SPAM page. The range value can be an asterisk (*), referring to all entries, or a set of consecutive entries that you describe as follows:

```
lower-data-page-number[:higher-data-page-number]
```

Each entry on a SPAM page consists of 2 bits, containing a value 0 through 3, that represents a fullness threshold. For example, if the *n*th SPAM entry contains a 2, it means that the *n*th data page in the interval has reached a percentage of fullness greater than the second threshold for the area, but less than or equal to the third threshold.

value
Specifies the new value of the field you are patching. The value is deposited in the default radix unless you specify otherwise in one of these ways:

- With a prior RADIX command.
- By specifying HEXADECIMAL or DECIMAL in a DEPOSIT DATA command.
- By enclosing ASCII data in quotation marks (" "). Timestamps must always be enclosed in quotation marks because they include punctuation characters.

Example

```
DBALTER> DEPOSIT DATA/WORD 0012=032c
```

This example deposits data 032c with a length of 2 bytes into offset location 0012.

See the *Oracle CODASYL DBMS Database Maintenance and Performance Guide* for examples of how to use the DEPOSIT command.

11.5 DEPOSIT FILE Command

Puts a new file specification or uncorrupt flag value into the root file for an area file or snapshot file.

Format

[DEPOSIT] FILE { area-name } [SNAPSHOT]

{ SPECIFICATION = file-spec }

Arguments

area-name
Specifies the area name of the area or snapshot file whose file specification or uncorrupt flag field you are changing in the root file.

SNAPSHOT
Specifies that the file whose root file fields you are changing is a snapshot file.

SPECIFICATION = file-spec
Specifies the full file specification that the area or snapshot file will have.

Example

See the *Oracle CODASYL DBMS Database Maintenance and Performance Guide* for examples of how to use the DEPOSIT command.

11.6 DEPOSIT ROOT Command

Enters a new file specification for the root file. This will be the file specification after it is committed to the database. The change does not actually occur until you have committed all changes and ended the DBALTER session.

Note

After committing a changed root file name, the database cannot be accessed until you copy it to the location (using the exact, fully qualified file specification) that you specify on the DEPOSIT ROOT command. After that, users bind to the database at the new location.

In addition, this command allows you to remove or rename recovery-unit journal (.RUJ) file references in the database root file. Removing the .RUJ file references from the root file permits users to bind to the database in situations where the .RUJ files have been accidently deleted. However, this action marks the database as "eternally corrupt," which results in a warning message in all future DBO functions against the database. Therefore, users must realize that the database is corrupt when they bind. The database must be restored and recovered from clean backup files to guarantee the consistency of the database.

Format

```
[ DEPOSIT ] ROOT { SPECIFICATION =root-file-spec }
                 { USER n RUJ_FILENAME=file-spec }
```

Argument

root-file-spec
Specifies the full file specification that the root file will have after it has been committed and the DBALTER session is completed.

USER n RUJ_FILENAME=file-spec
Specifies the new file specification for the .RUJ file for the user specified, where *n* is a valid user number. The file specification can be the null string ("").

Example

```
$ DBO/ALTER PARTS
DBALTER>DISPLAY ROOT USER *
Active user 0
    Process ID is 2020012A
    Stream ID is 1
    Monitor ID is 1
    Transaction ID is 2
    Recovery journal filename is "DISK1:[DBM$RUJ]PARTS$000197FF4EF3.RUJ;1"
DBALTER>DEPOSIT ROOT USER 0 RUJ_FILENAME = ""
              ***** WARNING! *****

  THE DBO ALTER COMMIT COMMAND WILL RESULT IN
  THE DATABASE BEING MARKED CORRUPT WITHOUT
  THE ABILITY TO UNMARK IT. EVER.

AN DBO ALTER ROLLBACK COMMAND WILL NOT
CORRUPT THE DATABASE.
              ***** WARNING! *****
Active user 0
Recovery journal filename is ""
```

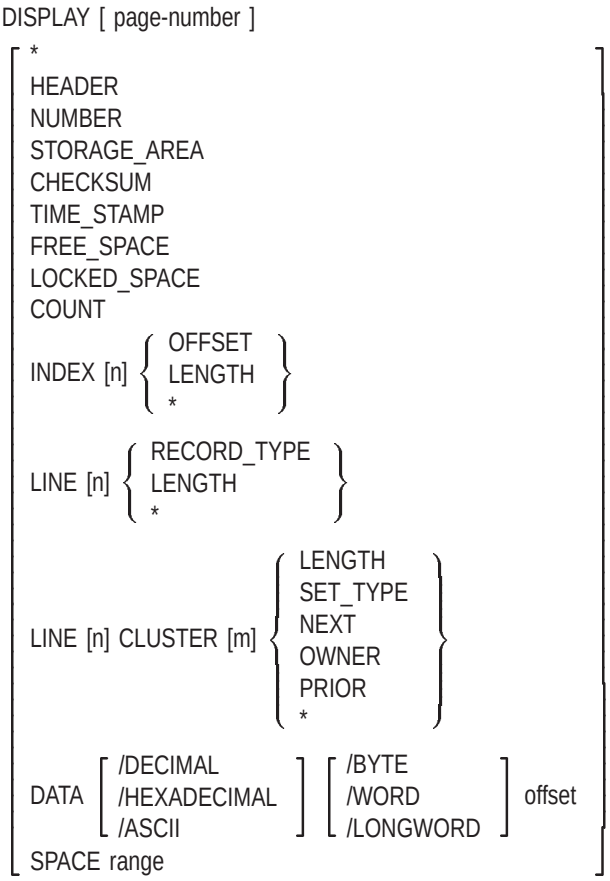
This example demonstrates the use of the DBO/ALTER DISPLAY ROOT command with the USER keyword to display the .RUJ file of all active users. Assume that for some reason the database can no longer access the displayed .RUJ file and therefore users can no longer access the database. The example uses the DBO/ALTER DEPOSIT ROOT command to enter the null string for this user's .RUJ file. The database becomes accessible again. (However, it should be restored and recovered from a backup file as soon as possible.)

See the *Oracle CODASYL DBMS Database Maintenance and Performance Guide* for additional examples of how to use the DEPOSIT command.

11.7 DISPLAY Command

Requests display of data fields from a database page. An individual DISPLAY command can include only one of the display options shown.

Format



Arguments

page-number
Identifies the page whose information you want to display. The current page is the default.

DISPLAY *
Displays the entire page.

HEADER
Displays the entire page header.

NUMBER
Shows the 4-byte page number field.

STORAGE_AREA
Displays the 2-byte storage area identification.

CHECKSUM
Displays the 4-byte page checksum field.

TIME_STAMP
Displays the 8-byte time-and-date-stamp field.

FREE_SPACE
Displays the 2-byte field indicating how much free space remains on the page.

LOCKED_SPACE
Displays the 2-byte field indicating how much free space is allocated for exclusive use by a run unit.

COUNT
Displays the 2-byte field showing the number of line index entries. If this number is 1, the page contains only the SYSTEM record.

Note

In the next three arguments, the integers denoting INDEX, LINE, and CLUSTER are zero based. For example, INDEX 0 refers to the first index, and LINE 3 CLUSTER 2 refers to the third cluster of the fourth line.

The integers *n* and *m* are optional. The present value of the relevant pointer is the default in each case.

References to INDEX, LINE, and CLUSTER are invalid if the current page is a SPAM page.

INDEX n
Displays the offset field, the length field, or both from the line index indicated by *n*. For example, if you enter DISPLAY INDEX 3 OFFSET, the offset address field from the fourth line index is displayed.

LINE n
Displays information from an individual storage segment. You can display the record-type field or the pointer cluster length, or the entire content of the storage segment line indicated by *n*. Each storage segment contains a set of pointer clusters.

LINE n CLUSTER m
Displays information from an individual pointer cluster.

DATA offset
Displays 1, 2, or 4 bytes of data in the radix specified. If you do not specify radix, the default radix is assumed. See the description of the RADIX command in this chapter for information on how to set a default radix.

SPACE range
Displays a specified range of SPAM entries; it is valid only if the current page is a SPAM page. SPACE is the only option that you can use in DISPLAY and DEPOSIT commands that access a SPAM page. The range value can be an asterisk (*), referring to all entries, or a set of consecutive entries that you describe as follows:

lower-data-page-number[:higher-data-page-number]

Each entry on a SPAM page consists of 2 bits, containing a value 0 through 3 that represents a fullness threshold. For example, if the *n*th SPAM entry contains a 2, it means that the *n*th data page in the interval has reached a percentage of fullness greater than the second threshold for the area, but less than or equal to the third threshold.

Example

```
DBALTER> DISPLAY 94
                   0003 0000005E  0000  page 94, physical area 3
                        3CB90DA0  0006  checksum = 3CB90DA0
               0091E1AE 96AFAE20  000A  time stamp = 30-DEC-1988 11:25:59.17
                        0000 0322  0012  802 free bytes, 0 locked
                             0002  0016  2 lines
                        0005 03E4  0018  line 0: offset 03E4, 5 bytes
                        009A 034A  001C  line 1: offset 034A, 154 bytes

                         00000000  0020  line 0: TSN 0
                         00000001  0024  line 1: TSN 1

00000000000000000000000000000001  0028  free space '................'
00000000000000000000000000000000  0038  free space '................'
                                  ::::  (48 duplicate lines)
                            0000  0348  free space '..'

                            0004  034A  line 1: record type 4
                         02 0015  034C  21 bytes in 2 sets/dynamic items
                         0004 0A  034F  10 bytes, storage set type 4
                           EF 0A  0352   next  3:77:1
                         0A7F 0B  0354   owner 3:5:0
                           A4 0A  0357   prior 3:2:1
                         000A 0A  0359  10 bytes, storage set type 10
                         0D64 0B  035C   next  3:52:5
                              00  035F   owner 3:94:1
                         0B01 0B  0360   prior 3:14:2
                                  ....  129 bytes of static data
4E4920444C554F473039383736353433  0363   data '34567890GOULD IN'
20202020202020202020202020202043  0373   data 'C               '
20202020202020202020202020202020  0383   data '                '
20202020444C414E4F44205245594F42  0393   data 'BOYER DONALD    '
30312020202020202020202020202020  03A3   data '              10'
4E414D20594157204E41495050412030  03B3   data '0 APPIAN WAY MAN'
2020202020484E202C52455453454843  03C3   data 'CHESTER, NH     '
10515503062020202020202020202020  03D3   data '           ..UQ.'
                             0C   03E3   data '.'

                           2001   03E4  line 0: SYSTEM record
                         00 0001  03E6  1 byte in 0 sets/dynamic items
                      0000000000  03E9  padding '.....'

                         FFFFFFFF  03EE  snap page pointer -1
                         00000001  03F2  snap pointer TSN 1
             00000000000000000000  03F6  MBZ '..........'
```

This example displays page 94.

See the *Oracle CODASYL DBMS Database Maintenance and Performance Guide* for examples of how to use the DISPLAY command.

11.8 DISPLAY FILE Command

Displays the file specification, uncorrupt flag value, or both in the root file for an area file or snapshot file.

If you display a field that you have changed during the current DBALTER session, the field is displayed as marked.

Format

```
DISPLAY FILE [ * | area-name ] [ SNAPSHOT ]

[ * | SPECIFICATION | UNCORRUPT ]
```

Arguments

*** (asterisk)**
Displays all file characteristics.

area-name
Specifies the area name of the area for which you want to display information from the root file.

SNAPSHOT
Displays information about a snapshot file.

SPECIFICATION
Displays the full file specification for the data storage or snapshot file of the area.

UNCORRUPT
Displays the current setting of the uncorrupt flag. This applies only to area files.

Example

See the *Oracle CODASYL DBMS Database Maintenance and Performance Guide* for examples of how to use the DISPLAY command.

11.9 DISPLAY ROOT Command

Displays the current root file specification of the database, or the current recovery-unit journal (.RUJ) file specification for a specified user of the database, depending on the keywords you use, as follows:

- Use this statement with the SPECIFICATION keyword to be sure that you have assigned the database file specification that you intended.
- Use this statement with the USER n keyword to display the .RUJ file name associated with a specified user or all users. Substitute a valid user number for *n* to receive information on a specific user; substitute the asterisk (*) for *n* to receive information on all users.

Format

```
DISPLAY ROOT [ * | SPECIFICATION [ root-file-spec] | USER n ]
```

Arguments

*** (asterisk)**
Displays all root file characteristics.

root-file-spec
Displays the current file specification of the root file. If the file specification has been changed during the current DBALTER session, it is displayed as marked.

If you omit the root-file-spec parameter, the current field is redisplayed.

USER n
Specifies the user for which you want the associated .RUJ file displayed. If *n* is a valid user number, the .RUJ file for that user is displayed. If *n* is the asterisk character (*), the .RUJ files for all current users are displayed.

Example

See the *Oracle CODASYL DBMS Database Maintenance and Performance Guide* for examples of how to use the DISPLAY command.

11.10 EXIT Command

Terminates the DBALTER session and returns you to the DCL command level. You can also press Ctrl/Z to end a DBALTER session.

If EXIT is issued and altered but uncommitted pages exist, you are told to issue either a COMMIT or a ROLLBACK command. DBALTER does not exit until COMMIT or ROLLBACK has accounted for all altered pages.

EXIT performs an implicit NOLOG command if a LOG file is open.

Format

```
EXIT
```

Example

```
DBALTER> EXIT
$
```

This example exits DBALTER command level and returns you to the DCL command level.

11.11 HELP Command

Provides information about DBALTER commands, terminology, and concepts. If you enter HELP at the DBALTER prompt without specifying a topic, DBALTER displays a list of topics on which help is available. If you enter HELP followed by a topic, you get a brief description of that topic.

Format

HELP [topic]

Argument

topic
Identifies the topic you want explained.

Example

```
DBALTER> HELP

  Information available:

  AREA-PAGE  BIND       COMMIT     DEPOSIT    DISPLAY    ERRORS     EXIT
  HELP       INTEGRATE  LOG        MAKE_CONSISTENT       MOVE       NOLOG
  PAGE       RADIX      ROLLBACK   UNBIND     UNCORRUPT  UPDATE     VERIFY

Topic? VERIFY

  Verifies the current page.

  Format:

      VERIFY;
  The current page  is  statically  verified,  i.e.,  no  sets  are
  walked.  Error messages are issued if the page is corrupt.

Topic?
DRU>
```

This example requests two layers of help. The first layer is for DBALTER, and a list of topics is displayed. The second layer is for the topic VERIFY, and information about the VERIFY command is displayed. Help then prompts for another topic. To return to the DBALTER prompt, press Return.

11.12 LOG Command

Keeps an audit trail of all or part of a DBALTER session. All DBALTER commands and their results are logged until you enter a NOLOG or EXIT command to close the log file.

Format

LOG file-spec

Argument

file-spec
Specifies a file to contain the audit trail log. If file-spec contains any non-alphanumeric characters, it must be enclosed in quotation marks (" "). The default file type is .LIS.

Examples

1. `DBALTER> LOG AUDIT`

   This example creates the file AUDIT.LIS and begins audit trail logging.

2. `DBALTER> LOG "AUDIT.TRL"`

   This example creates the file AUDIT.TRL and begins audit trail logging. The quotation marks (" ") are necessary because the file specification contains a non-alphanumeric character.

11.13 MAKE_CONSISTENT Command

Resets an area's inconsistent flag. This command also removes the area's (and associated snapshot file) entries from the corrupt page table (CPT), allowing you to use the database.

When an area is restored from backup on a per-area basis, it will not reflect data that has been updated since the backup. The transaction level of the restored area reflects the transaction level of the backup file, not the transaction level of the database. Therefore, the transaction level of the restored area differs from that of the database. DBCS marks the area by setting a flag in the area file to inconsistent.

You can perform a recovery by area to upgrade the transaction level of the restored area to that of the database. (After-image journaling must be enabled in order to restore by area.) If you know for certain that no updates have been made to the database since the backup, you can use the MAKE_CONSISTENT command in DBALTER to change the setting of the flag from inconsistent to consistent.

Note

The DBO/ALTER MAKE_CONSISTENT command is not the recommended method for clearing CPT entries for either data or snapshot files. The command should be used only as a last resort.

Format

MAKE_CONSISTENT { storage-area-name | storage-area-number }

Arguments

storage-area-name
Specifies an area of the database by storage area name, which is the name given by the AREA clause in the schema.

storage-area-number
Specifies an area of the database by storage area number, which is assigned when the database is created and is given on the first line of a page display.

Example

```
DBALTER> MAKE_CONSISTENT BUY

                    ***** WARNING! *****

 BEWARE ATTEMPTING TO MAKE CONSISTENT A STORAGE AREA
 WITHOUT FIRST VERIFYING IT WITH /OPTIONS=SETS!

 A DBALTER ROLLBACK COMMAND WILL LEAVE THIS AREA
 MARKED INCONSISTENT SO THAT YOU CAN PERFORM A VERIFY
 TO INSURE THAT IT IS ACTUALLY CONSISTENT.

Area BUY now marked consistent.
```

This example resets the indication flag from inconsistent to consistent for the area BUY.

11.14 MOVE Command

Moves data (defined by beginning and ending offset addresses) from one page location to another.

The number of bytes moved is:

(old-offset-end) - (old-offset-start) + 1

The sending field, defined by the old-offset arguments, remains unchanged.

The receiving field, defined by new-offset and the length of the sending field, is replaced by the contents of the sending field.

Other information in the page is unchanged.

Format

```
MOVE old-offset-start : old-offset-end new-offset
```

Arguments

old-offset-start
Specifies the hexadecimal offset address of the first byte in the data sequence to be moved.

old-offset-end
Specifies the hexadecimal offset address of the last byte in the data sequence to be moved.

new-offset
Specifies the hexadecimal offset address of the first byte in the sequence of bytes receiving the moved data.

Example

```
DBALTER> MOVE 34A:34E 354
```

This example moves data from offset locations 34A through 34E to the starting offset location of 354.

See the *Oracle CODASYL DBMS Database Maintenance and Performance Guide* for examples of how to use the MOVE command.

11.15 NOLOG Command

Stops DBALTER logging. NOLOG stops audit trail logging if a previous LOG command is still in effect. The EXIT command performs an implicit NOLOG if LOG is still active; thus, you need not enter a NOLOG command before exiting the DBALTER session.

Format

```
NOLOG
```

Example

```
DBALTER> NOLOG
```

This example stops audit trail logging.

11.16 PAGE Command

Fetches a page from the current area. This command fetches a different page of the same area. You can specify a page number or you can fetch the next page in sequence. If you enter PAGE without a page number and you are already at the highest numbered page of the area, the fetch sequence wraps to Page 1.

Format

```
PAGE [ page-number ]
```

Argument

page-number
Identifies a page to be altered in the current database area. The default is the current page number plus 1.

Example

```
DBALTER> PAGE 14
```

This example fetches page 14 of the current area.

11.17 RADIX Command

Sets the default radix for entering numeric data. This command does not change the radix for specifying page offsets. Page offsets must always be specified in hexadecimal radix.

Format

RADIX { DECIMAL / HEXADECIMAL }

Arguments

DECIMAL
Sets the default radix in decimal numbers.

HEXADECIMAL
Sets the default radix in hexadecimal numbers.

Example

```
DBALTER> RADIX HEXADECIMAL
```

This example sets the default radix for entering numeric data to hexadecimal.

11.18 ROLLBACK Command

Ignores all page changes since the last COMMIT or ROLLBACK command. Altered pages are stored in virtual memory until you issue a COMMIT or ROLLBACK command. ROLLBACK tells DBALTER to ignore all changes since the last COMMIT or ROLLBACK.

Do not follow a ROLLBACK command with a DEPOSIT command without first respecifying your location with a DISPLAY or PAGE command.

DBALTER does not allow EXIT until all altered pages are either committed or rolled back.

Format

ROLLBACK

Example

```
DBALTER> ROLLBACK
```

This example rolls back all page changes entered since the last COMMIT or ROLLBACK.

11.19 UNBIND Command

Releases a previous database BIND command. The EXCLUSIVE UPDATE lock is released, and other users can access the database again. The database to which DBALTER is currently bound is released, with the EXCLUSIVE UPDATE lock discontinued. No database page alterations are allowed until another BIND command is issued.

Format

UNBIND

Example

```
DBALTER> UNBIND
```

This command unbinds from the current database.

11.20 UNCORRUPT Command

Resets the area's corruption flag, thus allowing you to use the uncorrupted sections of a corrupted storage area. Storage areas are most often corrupted by attempting an Oracle CODASYL DBMS (not DBALTER) rollback with one or more storage areas opened in BATCH UPDATE.

The UNCORRUPT command allows you to access a database that is in an uncertain condition. Accordingly, when you enter UNCORRUPT, the following warning message is displayed:

```
BEWARE ATTEMPTING TO UNCORRUPT A STORAGE AREA
WITHOUT FIRST VERIFYING IT WITH OPTION=SETS!
```

Format

UNCORRUPT { storage-area-name | storage-area-number }

Arguments

storage-area-name
Specifies an area of the current database by storage area name, which is the name given by the AREA clause in the schema.

storage-area-number
Specifies an area of the current database by the storage area number, which is assigned when the database is created and is given on the first line of a page display.

Example

```
DBALTER> UNCORRUPT MAKE

               ***** WARNING! *****

   BEWARE ATTEMPTING TO UNCORRUPT A STORAGE AREA
   WITHOUT FIRST VERIFYING IT WITH /OPTIONS=SETS!

   A DBALTER ROLLBACK COMMAND WILL LEAVE THIS AREA
   MARKED CORRUPT SO THAT YOU CAN PERFORM A VERIFY
   TO INSURE THAT IT IS ACTUALLY NOT CORRUPT.

 Area MAKE now marked uncorrupt.
```

This example resets the MAKE area's corruption indication flag.

11.21 UPDATE CHECKSUM Command

Sets the checksum value of the current page to the correct value.

Format

```
UPDATE CHECKSUM
```

Example

```
DBALTER> PAGE 1
DBALTER> VERIFY
%DBO-F-PAGCKSBAD, area MAKE, page 1
    contains an invalid checksum
    expected: 33AA0012, found 12345678
%DBO-W-PAGCPTERR, Area MAKE, page 1
    page state mismatch.
    On-page page state is corrupt, page is logged consistent.
DBALTER> UPDATE CHECKSUM
DBALTER> VERIFY
DBALTER> COMMIT
```

This example verifies page 1 in area MAKE and determines that it contains an invalid checksum. The user updates the checksum with the UPDATE CHECKSUM command and verifies the page again.

11.22 VERIFY Command

Verifies the current page statically. In a static verification, no sets are walked. This is similar to a DBO/VERIFY/OPTIONS=SEGMENT command, except DBALTER VERIFY checks only the current page. If the database uses space area management (SPAM) and the current page is a data storage page, the SPAM

entry for the current page is checked for correct value. If the current page is a SPAM page, the page is checked for valid format, but the individual SPAM entries are not checked for correctness. Error messages are issued if the page is corrupt.

If a data or SPAM page was previously logged as corrupt in the corrupt page table (CPT), but is no longer corrupt, issuing the DBO/ALTER VERIFY command clears the corruption entry from the CPT for this page. DBO/ALTER VERIFY does not clear entries from the CPT that are logged as inconsistent.

Format

VERIFY

Example

```
DBALTER> AREA 3 PAGE 100
DBALTER> VERIFY
```

This example verifies area 3 page 100.

```
$ DBO/VERIFY/LOG/AREA=MAKE/NOCDD PARTS
   .
   .
   .
%DBO-F-PAGCKSBAD, area MAKE, page 1
        contains an invalid checksum
        expected: 346BA012, found 00A98AD3
%DBO-W-PAGINCPT, Area MAKE, page 1
        page has been logged corrupt.
$ DBO/ALTER PARTS
DBALTER> VERIFY
%DBO-F-PAGCKSBAD, area MAKE, page 1
        contains an invalid checksum
        expected: 346BA012, found 00A98AD3
%DBO-W-PAGCPTERR, Area MAKE, page 1
        page state mismatch.
        On-page page state is corrupt, page is logged consistent.
DBALTER> UPDATE CHECKSUM
DBALTER> COMMIT
DBALTER> EXIT
$ DBO/VERIFY/LOG/AREA=MAKE/NOCDD PARTS
%DBO-I-DBBOUND, bound to database "DISK1:[DBUSER]PARTS.ROO;1"
%DBO-I-NOCDDVFY, CDD metadata not being verified
%DBO-I-OPENAREA, opened storage area MAKE for protected retrieval
%DBO-I-PAGERANGE, verification page range 1 to 101 of area MAKE
%DBO-I-BEGINSEGV, beginning segment verification of MAKE area
%DBO-I-OPENAREA, opened storage area BUY for protected retrieval
%DBO-I-ENDSEGV, completed segment verification of MAKE area
%DBO-S-NOPAGERRS,       no page format errors encountered
%DBO-S-NOSEGERRS,       no segment format errors encountered
%DBO-S-NOSETERRS,       no set chain errors encountered
%DBO-I-CLOSAREAS, releasing protected retrieval lock on all storage areas
```

This example demonstrates how the DBALTER VERIFY command removes pages that are no longer corrupt from the corrupt page table. The first DBO/VERIFY command shows that a page has been logged as corrupt in the CPT. The DBALTER UPDATE CHECKSUM command is used to fix the corruption. The second DBO/VERIFY command removes the page from the corrupt page table.

12

DRU Commands

This chapter describes the Database Restructuring Utility (DRU) of Oracle CODASYL DBMS. With DRU, you can change certain database characteristics without unloading and reloading the database. These types of changes affect the schema and storage schema contained in the root file. These changes might also affect existing database supporting structures such as set pointer clusters. The changes you can make include:

- Removing areas from record WITHIN clauses
- Changing set insertion and retention to AUTOMATIC MANDATORY or AUTOMATIC FIXED
- Specifying that duplicates are not allowed on sorted sets
- Changing the sort key and sort order on sorted sets
- Changing set ORDER and MODE clauses
- Changing B-tree node size, resetting the B-tree fill factor, and, for system-owned sets, changing the B-tree page location
- Removing data items from a database
- Changing data type and length
- Changing allocation modes
- Removing DEFAULT values
- Reloading an area

To enter the DRU environment, use the following syntax:

```
DBO/MODIFY/RESTRUCTURE [root-file-spec]
```

The optional root-file-spec parameter identifies the database to access. If you specify this parameter, you automatically bind to the specified database. If you do not specify this parameter, you must use the DRU BIND command. See Section 12.2 for more information on the BIND command.

A file named DRUINI.DRU is an indirect command file that executes immediately and automatically each time you access DRU. You can define the logical name DRUINI to point to DRUINI.DRU in another directory.

DRU responds with the following prompt:

```
DRU>
```

This prompt means that the system expects DRU command input. The DRU commands are:

ANALYZE	DEFINE	MACRO	SET

BIND	EDIT	OPEN	SHOW
CLOSE	EXECUTE	PREPARE	STOP
CREATE	EXIT	REMOVE	UNBIND
CLEANUP	HELP	REVERSE	

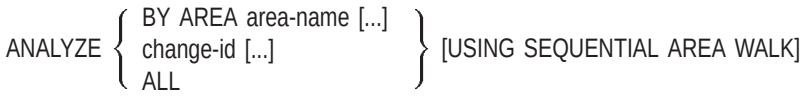
Before you make any restructuring changes, Oracle recommends that you make a full backup copy of the database. Making a backup before restructuring allows you to restore the database to its previous state in case any problems occur during the restructuring. DRU contains complete journaling and recovery capabilities. This means that if you need to restore a database from a backup file that was created before the restructuring change was executed, you can roll the database forward from the .AIJ file to recover the changes.

The remainder of this chapter describes the full syntax and use of these commands. For detailed information on using DRU, see the *Oracle CODASYL DBMS Database Maintenance and Performance Guide*.

12.1 ANALYZE Command

Simulates the execution of a set of restructuring changes and displays information. The output shows whether the change will walk a set or area and indicates the name of the set or area used. If a change will be converted from set walking to area walking, this is indicated. Also included is what change forced the conversion. The display also shows what areas will be readied and in what mode. ANALYZE does not start any transactions or ready any areas. Before you can analyze changes, you must first bind to the database and open a previously created change file.

Format



Arguments

BY AREA area-name [...]
Identifies one or more areas to be analyzed. These areas are readied in EXCLUSIVE UPDATE mode and contain the records that are changed. Only changes that affect the specified area(s) are analyzed.

change-id [...]
Identifies one or more unique names that identify changes, as you defined with the DEFINE command, in the current change file.

ALL
Analyzes all changes in the change file.

USING SEQUENTIAL AREA WALK
Specifies that areas to be analyzed are walked sequentially by area rather than by set walks. This option should be used prior to issuing the EXECUTE USING SEQUENTIAL AREA WALK command.

Examples

1.
```
DRU> SHOW RETENTION_CHANGE

Change file entries for DISK:[USER]PARTS_CHANGES.RCF;1

RETENTION_CHANGE - MODIFY ALL_PARTS_ACTIVE
       RETENTION IS MANDATORY


DRU> ANALYZE RETENTION_CHANGE

================================================================================
Oracle CODASYL DBMS Restructuring Analysis Report
    Database filename is DISK:[USER]PARTS.ROO;1
    Change filename is DISK:[USER]PARTS_CHANGES.RCF;2
================================================================================

================================================================================
                        Changes being analyzed for execution
================================================================================

Execution of RETENTION_CHANGE will be executed via a set walk of the sets
    ALL_CLASS
    CLASS_PART

================================================================================
    Areas affected or used for access and their ready modes
================================================================================

Storage area MAKE is an affected area readied for EXCLUSIVE UPDATE

Storage area BUY is an affected area readied for EXCLUSIVE UPDATE
```

   This example analyzes the change RETENTION_CHANGE.

2.
```
DRU> ANALYZE BY AREA MAKE
%DBO-F-NOENTQUAL, No entries qualified for execution
```

   This example attempts to analyze the changes in the MAKE area. Because no changes affect just the MAKE area, no entries qualified.

3.
```
DRU> ANALYZE BY AREA MAKE BUY

================================================================================
Oracle CODASYL DBMS Restructuring Analysis Report
    Database filename is DISK:[USER]PARTS.ROO;1
    Change filename is DISK:[USER]PARTS_CHANGES.RCF;1
================================================================================

================================================================================
                        Changes being analyzed for execution
================================================================================

Execution of WITHIN_CHANGE will be executed via an area walk of the
                                    storage areas
    MAKE
    BUY

================================================================================
            Areas affected or used for access and their ready modes
================================================================================

Storage area MAKE is an affected area readied for EXCLUSIVE UPDATE

Storage area BUY is an affected area readied for EXCLUSIVE UPDATE
```

   This example analyzes the MAKE and BUY areas. Only the

WITHIN_CHANGE will be executed.

12.2 BIND Command

Binds to the database you want to access. You must bind to a database before you can proceed with any DRU commands. If you specify the root file as an argument to the DBO/MODIFY/RESTRUCTURE command, DRU automatically binds to the database.

You can bind to only one database at a time. Before binding to another database for restructuring, you must unbind from the current one. See Section 12.19 for information on the UNBIND command.

Format

BIND root-file-spec

Argument

root-file-spec
Identifies the database root file to access. The root file specification must be enclosed in quotation marks (" ") if it contains non-alphanumeric characters. If you use a logical name, it cannot be a DRU reserved word. See the *Oracle CODASYL DBMS Quick Reference Guide* for a list of DRU reserved words. The default file type is .ROO.

Examples

1.
```
$ DBO/MODIFY/RESTRUCTURE
DRU> BIND PARTS
%DBO-I-DBBOUND, bound to database "DISK:[USER]PARTS.ROO;1"
```

   This example enters the database restructuring utility, then binds to the PARTS database.

2.
```
$ DBO/MODIFY/RESTRUCTURE PARTS
%DBO-I-DBBOUND, bound to database "DISK:[USER]PARTS.ROO;1"
```

   This example enters the database restructuring utility and binds to the PARTS database in a single command.

12.3 CLEANUP Command

Cleans up the database after all records have been reloaded from a reload operation.

The CLEANUP command requires exclusive access to the database, but it is not necessary to perform the cleanup immediately upon completion of the reload. Instead, you can wait until a convenient time to close the database and run the CLEANUP command. This command performs the following tasks:

- Checks to ensure that the state information stored in DROOT is set to RELOADED

- Removes the old area name from the subschema realm and the schema WITHIN clause
- Changes the original area to a reserved reload area
- Deletes the original area file
- Clears the DROOT reload state
- Journals the changes to the .AIJ file if after-image journaling is enabled

Format

```
CLEANUP change-id
```

Arguments

change-id
Specifies a unique name that identifies the reload area, as you defined it with the DRU DEFINE command, for which cleanup is needed.

Example

```
DRU> CLEANUP MAKE_ONLINE
```

This command cleans up after the reload operation described by the MAKE_ ONLINE change-id.

12.4 CLOSE Command

Closes the current change file. Only one change file can be open at a time. You must close the current change file before you can open a different change file.

Format

```
CLOSE
```

Example

```
DRU> CLOSE
```

This command closes the current change file.

12.5 CREATE Command

Opens and creates a restructuring change file for the currently bound database. A change file contains header information about the database and definitions of the restructuring changes that you make with the DEFINE command. Multiple change files can exist for a database; however, only one change file can be open at a time.

12.5 CREATE Command

Format

CREATE change-file

Argument

change-file
Identifies the name of the change file to create. The default file type is .RCF. The default protection of a change file is such that unprivileged users can delete it.

Example

```
DRU> CREATE PARTS_CHANGES
%DBO-I-LOGFILACC, Created change file DISK:[USER]PARTS_CHANGES.RCF;1
```

This example creates the change file PARTS_CHANGES.RCF.

12.6 DEFINE Command

Defines one or more restructuring changes in a change file.

Use the DEFINE command to define these restructuring changes:

- Change allocation mode
- Change B-tree node size, reset the B-tree fill factor, and, for system-owned sets, change the B-tree page location
- Change data type and length
- Change set insertion and retention to AUTOMATIC MANDATORY or AUTOMATIC FIXED
- Change set ORDER and MODE clauses
- Change the sort key and sort order on sorted sets
- Reload an area
- Remove AREAS from record WITHIN clauses
- Remove data items from your database
- Remove the DEFAULT clause
- Specify that duplicates are not allowed on sorted sets

Before you can define a change, you must first create and open a change file, or open a previously created change file. Changes in the change file do not take effect until they are selected for execution.

Format

DEFINE change-id action target object [target object ...]

Arguments

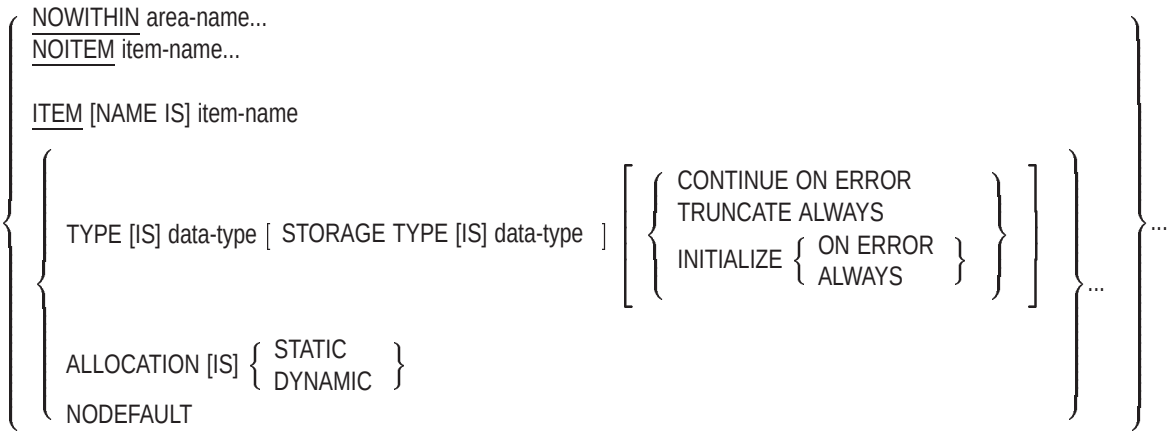
change-id
Specifies a unique name to identify a change in the current change file. The change-id identifies this particular change in the ANALYZE, EXECUTE, REMOVE, and SHOW commands. The change-id can be up to 31 characters and must start with an alphanumeric character.

action
Identifies the action to be taken. MODIFY and RELOAD AREA are the only actions available.

target
Identifies the set, record, or area name to be changed.

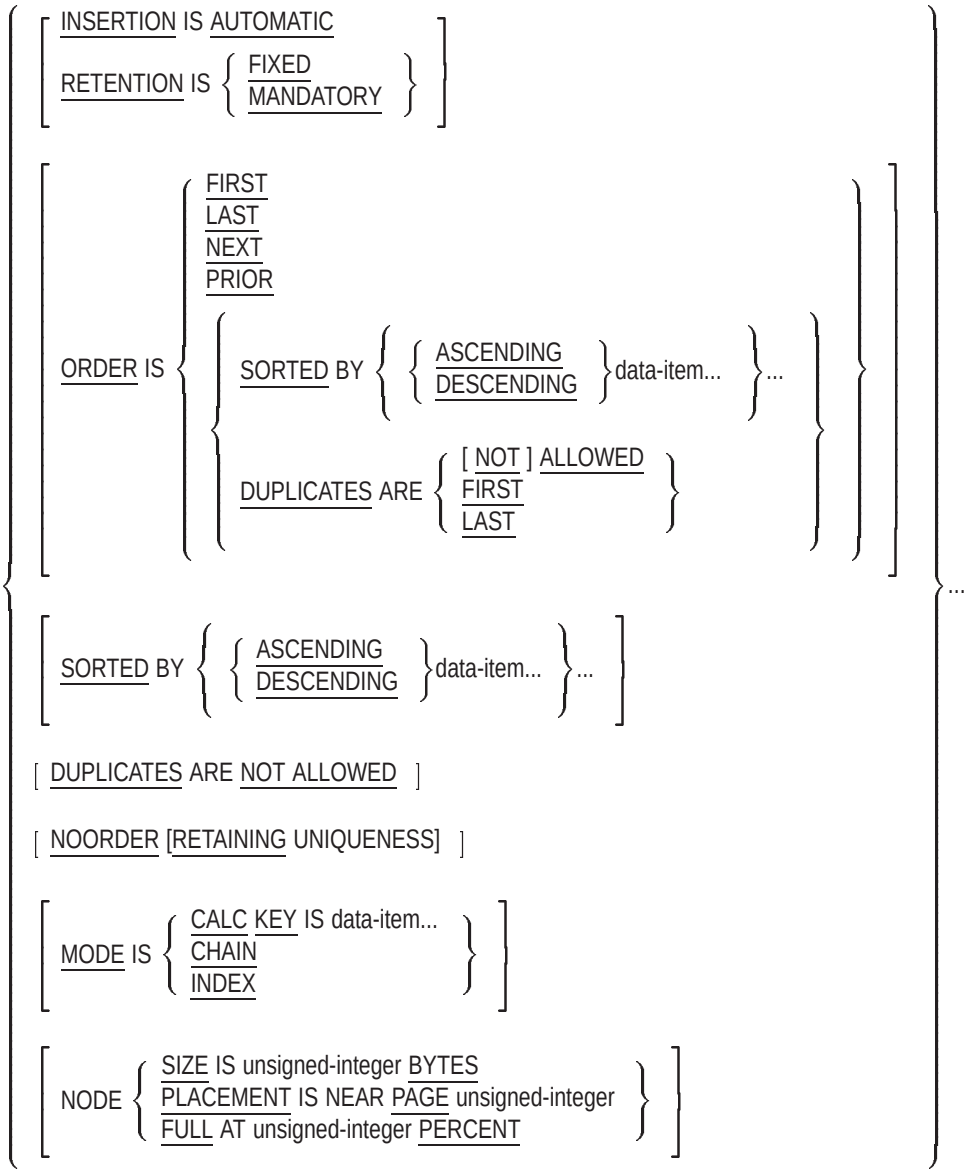
object
For the MODIFY action, specifies the new DDL information for the record, set, or area name specified.

The syntax for a record change definition is:

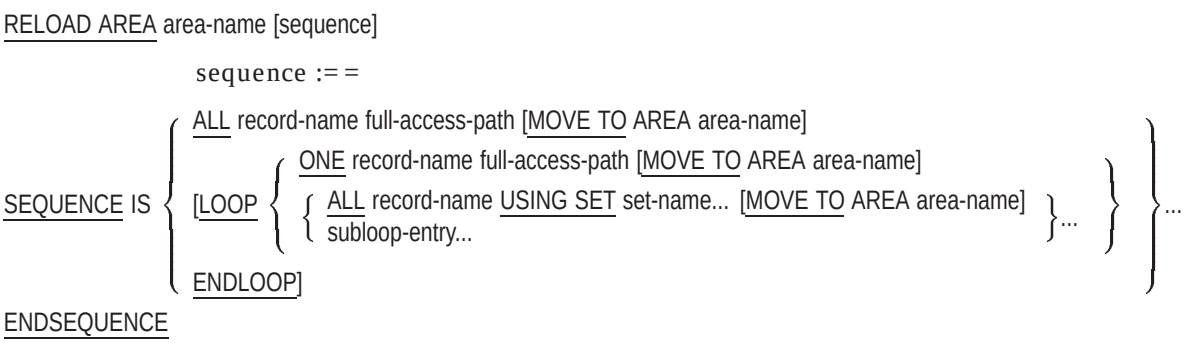


The syntax for a set change definition is:

12.6 DEFINE Command

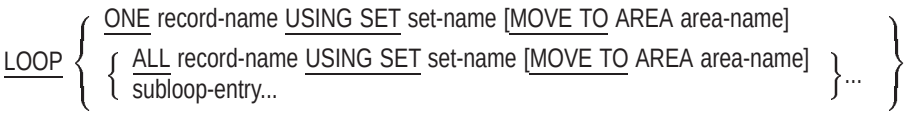


The syntax for reloading an area is:



full-access-path :==

And where subloop-entry is:



Examples

1.
```
DRU> DEFINE WITHIN_CHANGE MODIFY CLASS NOWITHIN MAKE
```

The DEFINE command defines a change that when executed disallows the CLASS record from being stored in the MAKE area.

2.
```
DRU> DEFINE RETENTION_CHANGE MODIFY ALL_PARTS_ACTIVE RETENTION IS MANDATORY
```

The DEFINE command defines a change that will change the retention from OPTIONAL to MANDATORY. Insertion remains AUTOMATIC.

3.
```
DRU> DEFINE ORDER_MODE_CHANGE MODIFY CONSISTS_OF SORTED BY ASCENDING -
DRU>_ EMP_ID DUPLICATES ARE NOT ALLOWED MODE IS CHAIN
DRU>
```

The DEFINE command defines a change that when executed will change the sort key to be EMP_ID, disallow duplicates, and change the set mode from index to chain.

4.
```
DRU> DEFINE CHG_VEND_ID MODIFY VENDOR -
DRU>_ ITEM VEND_ID -
DRU>_ ALLOCATION IS STATIC -
DRU>_ TYPE IS UNSIGNED NUMERIC 10 -
DRU>_ STORAGE TYPE IS PACKED DECIMAL 10 -
DRU>_ NOITEM VEND_CONTACT VEND_ADDRESS
```

The DEFINE command changes the allocation mode and data type of the item VEND_ID and removes the two items, VEND_CONTACT and VEND_ADDRESS, from the record.

5.
```
DRU> DEFINE CHNG_AREA RELOAD AREA MAKE -
DRU>_ SEQUENCE -
DRU>_  LOOP -
DRU>_   ONE CLASS USING SET ALL_CLASS -
DRU>_   LOOP -
DRU>_    ONE PART USING SET CLASS_PART -
DRU>_    ALL COMPONENT USING SET PART_USES -
DRU>_        MOVE TO AREA NEW_AREA -
DRU>_   ENDLOOP -
DRU>_  ENDLOOP -
DRU>_ ENDSEQUENCE
```

The DEFINE command defines a change that when executed reloads the CLASS and PART records within the MAKE area, and moves the COMPONENT records from the MAKE area to the NEW_AREA area.

12.7 EDIT command

Calls the VAX EDT editor, allowing you to edit DRU commands previously issued within the current session. DRU stores all commands except EDIT, SET EDIT, and HELP, in the DRU edit save buffer until you exit from the utility. You can use EDT to modify DEFINE commands stored in the DRU edit save buffer or create new DEFINE commands.

By default, the maximum number of commands in the DRU edit save buffer is 50. You can use the SET EDIT KEEP command to override this default. See Section 12.16 for information on the SET command. If you do not specify an argument, EDIT recalls the last DRU line entered.

If you exit from the editor, you are returned to the DRU prompt and lines in the edit save buffer are not executed. If you quit from the editor, you are returned to the DRU prompt. No work done during the EDT session remains.

Format

```
EDIT [ integer ]
     [ *       ]
```

Arguments

integer
Specifies the number of lines you want to recall. If you specify zero, the resulting editing buffer will be empty.

*** (asterisk)**
Allows you to edit all lines that are currently saved within the editing save buffer.

Example

```
DRU> BIND PARTS
%DBO-I-DBBOUND, bound to database "DISK:[USER]PARTS.ROO;1"
DRU> CREATE PARTS_CHANGES
%DBO-I-LOGFILACC, Created change file DISK:[USER]PARTS_CHANGES.RCF;1
DRU> OPEN PARTS_CHANGES
DRU> DEFINE WITHIN_CHANGE MODIFY CLASS NOWITHIN MAKE
DRU> DEFINE RETENTION_CHANGE MODIFY ALL_PARTS_ACTIVE RETENTION IS FIXED
```

Suppose you entered the previous commands. Then you realize that you actually wanted MANDATORY RETENTION for ALL_PARTS_ACTIVE. Delete the change using the REMOVE command:

```
DRU> REMOVE RETENTION_CHANGE
1 change(s) removed
```

Enter the EDIT command:

```
DRU> EDIT 2
```

The editing buffer now contains:

```
DEFINE RETENTION_CHANGE MODIFY ALL_PARTS_ACTIVE RETENTION IS FIXED
REMOVE RETENTION_CHANGE
```

Using EDT, modify the DEFINE command and delete the REMOVE command:

```
DEFINE RETENTION_CHANGE MODIFY ALL_PARTS_ACTIVE RETENTION IS MANDATORY
```

When you exit, the change file will contain the modified change:

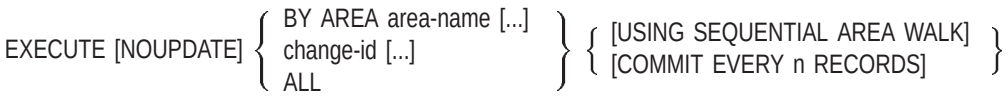
```
DRU> SHOW CHANGES

Change file entries for DISK:[USER]PARTS_CHANGES.RCF;2

WITHIN_CHANGE - MODIFY CLASS
       NOWITHIN MAKE

RETENTION_CHANGE - MODIFY ALL_PARTS_ACTIVE
       RETENTION IS MANDATORY
```

12.8 EXECUTE Command

Applies changes to the database you are bound to as specified by the DEFINE command.

The first EXECUTE command after each bind operation requests confirmation. See Section 12.16 for details.

Format



Arguments

NOUPDATE
Specifies that the changes not be written to a disk. Use this option when specifying data type or allocation mode changes to determine if the changes are feasible and if the changes cause records to fragment. If you specify any other changes, a warning message is returned and the change is removed.

Using the EXECUTE NOUPDATE argument will ready the areas in protected retrieval (PR) mode.

BY AREA area-name [...]
Identifies one or more areas to be executed. These areas are readied in EXCLUSIVE UPDATE mode and contain the records that will be changed. Only changes that affect the specified area(s) are executed.

change-id [...]
Specifies one or more unique names that identify changes, as you defined them with the DEFINE command, in the current change file.

ALL
Executes all changes in the change file.

USING SEQUENTIAL AREA WALK
Specifies that DRU is to walk areas rather than perform a set walk as selected by the DRU optimizer with this option. Before specifying USING SEQUENTIAL AREA WALK, you should analyze the DRU changes using the ANALYZE command to determine the scanning modes selected by the DRU optimizer.

Specifying USING SEQUENTIAL AREA WALK will force all the selected changes to walk areas sequentially. Depending on the characteristics of the changes, an area walk may be more efficient.

COMMIT EVERY n RECORDS
Specifies, during an online area reload operation, how often to commit changes to the database. If you do not specify the COMMIT clause, the reload utility performs a commit operation after each record has been loaded.

Examples

1.
```
DRU> EXECUTE RETENTION_CHANGE

%DBO-I-BEGINPHASE, beginning set walk phase
%DBO-I-OPEMAREA, opened storage area MAKE for exclusive update
%DBO-I-OPEMAREA, opened storage area BUY for exclusive update
%DBO-I-EXECHANGE, executing RETENTION_CHANGE
%DBO-S-CHGSUCCESS, execution of RETENTION_CHANGE completed successfully
```

   In this example the change, RETENTION_CHANGE, is set-walked and executes successfully.

2.
```
DRU> EXECUTE ORDER_MODE_CHANGE WITHIN_CHANGE

%DBO-I-BEGINPHASE, beginning set walk phase
%DBO-I-OPENAREA, opened storage area PERSONNEL for exclusive update
%DBO-I-OPENAREA, opened storage area MAKE for exclusive update
%DBO-I-OPENAREA, opened storage area BUY for exclusive update
%DBO-I-EXECHANGE, executing WITHIN_CHANGE
%DBO-E-SETWITHIN, ALL_CLASS set OWNER found in nowithin area at 1:2:0
%DBO-E-RECWITHIN, CLASS record found in nowithin area at 1:2:9
%DBO-E-RECWITHIN, CLASS record found in nowithin area at 1:2:7
%DBO-E-RECWITHIN, CLASS record found in nowithin area at 1:2:5
%DBO-E-RECWITHIN, CLASS record found in nowithin area at 1:2:3
%DBO-E-RECWITHIN, CLASS record found in nowithin area at 1:2:1
%DBO-E-CHGFAILED, execution of WITHIN_CHANGE failed

%DBO-I-BEGINPHASE, beginning area walk phase
%DBO-I-OPENAREA, opened storage area PERSONNEL for exclusive update
%DBO-I-OPENAREA, opened storage area MAKE for exclusive update
%DBO-I-OPENAREA, opened storage area BUY for exclusive update
%DBO-I-EXECHANGE, executing ORDER_MODE_CHANGE
%DBO-S-CHGSUCCESS, execution of ORDER_MODE_CHANGE completed successfully
```

   In this example the change, WITHIN_CHANGE, is set-walked and fails execution. The change, ORDER_MODE_CHANGE, is area-walked and executes successfully.

3.

```
DRU> EXECUTE NOUPDATE ALL USING SEQUENTIAL AREA WALK
%DBO-I-BEGINPHASE, beginning area walk phase
%DBO-I-OPENAREA, opened storage area A1 for protected retrieval
%DBO-I-OPENAREA, opened storage area A2 for protected retrieval
%DBO-I-OPENAREA, opened storage area A3 for protected retrieval
%DBO-I-EXECHANGE, executing AA
%DBO-I-EXECHANGE, executing BB
%DBO-I-EXECHANGE, executing CC
%DBO-W-RECFRAG, R3 at 1:4:1 may be fragmented
%DBO-W-RECFRAG, R1 at 1:4:2 may be fragmented
%DBO-W-RECFRAG, R3 at 1:5:1 may be fragmented
%DBO-W-RECFRAG, R1 at 1:5:2 may be fragmented
%DBO-I-TOTDSKSPA, total 301 bytes will be freed for the area A1
%DBO-I-TOTDSKSPA, total 102 bytes will be freed for the area A2
%DBO-I-TOTDSKSPA, total 0 bytes will be freed for the area A3
%DBO-S-CHGCMPNOUP, execution of AA completed with NOUPDATE mode
%DBO-S-CHGCMPNOUP, execution of BB completed with NOUPDATE mode
%DBO-S-CHGCMPNOUP, execution of CC completed with NOUPDATE mode
%DBO-I-REVALIDAT, Re-validating unexecuted valid change entries...
```

In this example, all the changes are executed with NOUPDATE mode and areas, then the sets are walked sequentially. Using the NOUPDATE option shows the possible record fragmentations and the total number of disk blocks used or freed.

4.
```
DRU> EXECUTE MAKE_ONLINE COMMIT EVERY 5 RECORDS
%DBO-I-RELOADBEG, beginning of reload operation
   .
   .
   .
%DBO-I-LOGONLNUM, reloaded COMPONENT up to 1:100:9 at 15-SEP-1995
   15:10:44.62; total 147
%DBO-I-CHGRELOAD, database completely reloaded for MAKE_ONLINE
```

This example executes a reload of the MAKE area as defined by the MAKE_ ONLINE change-id. It directs DRU to perform a commit operation after every 5 records are loaded into the reload area.

12.9 EXIT Command

Terminates the DRU session and returns you to the DCL command level. You can also press Ctrl/Z to end a DRU session.

Format

EXIT

Example

```
DRU> EXIT
$
```

This example exits from the DRU environment and returns you to the DCL command level.

12.10 HELP Command

Provides information about DRU commands, terminology, and concepts. If you enter HELP at the DRU prompt without specifying a topic, DRU displays a list of topics on which help is available. If you enter HELP followed by a topic, you get a brief description of that topic. You can enter HELP immediately after receiving an error message to display an explanation of the error.

Format

HELP [topic]

Argument

topic
Identifies the topic you want explained.

Examples

1.
```
DRU> HELP

  Information available:

  ANALYZE    BIND      CLOSE      Comments  Continuation         CREATE
  DCL_command          DEFINE     DRUINI    EDIT       EXECUTE   EXIT
  HELP       Indirect  Introduction         MACRO      OPEN      REMOVE
  SET        SHOW       UNBIND

Topic? CLOSE

CLOSE

   Use the CLOSE command to close the current change file.

   CLOSE

   Only one change file can be open at a time.  In order to open a
   different change file, the current file must first be closed by
   issuing this command.

Topic?
DRU>
```

This example requests two layers of help. In the first layer, DRU and a list of topics are displayed. In the second layer, the topic CLOSE and information about the CLOSE command are displayed. Help then prompts for another topic. To return to the DRU prompt, press Return.

2.
```
DRU> CREATE PARTS_CHANGES
%DBM-F-NOT_BOUND, database is not bound
DRU> HELP

DBO

  DBO_ERRORS

    NOT_BOUND

       database is not bound

       Explanation:  You have not bound to a database yet, or you have
       unbound the database and have not bound to another one yet.
```

```
        User Action:  Bind to a database before continuing.
```

In this example, the user tries to create a change file. The Oracle CODASYL DBMS fatal error message "database is not bound" is displayed. The user then enters HELP at the DRU prompt to see information on the error message.

12.11 MACRO Command

Creates a DRU indirect command file with a default file type of .RCF. The format of the MACRO utility is:

MACRO macro-filename

DRU prompts you for input with the macro> prompt. Enter as many lines as you need, and they will be written to the specified macro file name. Specify the end of the MACRO by entering a semicolon (;) or by pressing Ctrl/Z to terminate the command.

To execute the macro, enter the macro file name preceded by the at sign (@) at the DRU prompt:

@macro-filename

To delete the macro file, use the DCL DELETE command.

Format

MACRO macro-filename

Argument

macro-filename
Identifies the name of the macro file. The default file type is .DRU.

Example

```
DRU> MACRO PARTS_CHANGES
.....macro> DEFINE WITHIN_CHANGE MODIFY CLASS NOWITHIN MAKE
.....macro> DEFINE RETENTION_CHANGE MODIFY ALL_PARTS_ACTIVE RETENTION IS FIXED
.....macro> ;
DRU> @PARTS_CHANGES
```

This example creates a macro file containing two DEFINE commands. Then the macro is executed.

12.12 OPEN Command

Opens an existing change file. You must be bound to a database before you can open a change file. Only one change file can be open at a time. To open a different change file, the current change file must first be closed. See Section 12.5 for information on creating a change file. See Section 12.4 for information on closing a change file.

12.12 OPEN Command

Upon opening a change file, any unexecuted changes are revalidated. If a change is found to be no longer valid due to results of execution, the change is marked INVALID. Invalid changes should be removed from the change file. See Section 12.14 for information on the REMOVE command.

Format

OPEN change-file

Argument

change-file
Identifies the name of the change file to open. The default file type is .RCF.

Example

```
DRU> OPEN PARTS_CHANGES
```

This example opens the change file PARTS_CHANGES.RCF.

12.13 PREPARE Command

Prepares a reload area for execution. This step in a reload area operation requires exclusive access to the database, but you need not perform the preparation immediately before reloading an area. You can perform the prepare operation at your convenience (such as when the database must be taken off line for another reason) at any time before the reload area operation.

During the prepare operation, DRU:

- Defines a target area using a reserved reload area
- Adds the target area to the subschema realm area list and to the records' WITHIN lists
- Creates a target area file using the area qualifiers specified in the DBO/MODIFY/RESTRUCTURE/ONLINE command
- Moves system-owned chain and index set owner records to the target area
- Builds the system-owned index set B-tree on the target area
- Saves the PREPARE state to DROOT
- Journals the changes to the .AIJ file if after-image journaling is enabled

Format

PREPARE change-id

Arguments

change-id
Specifies a unique name that identifies the change, as you defined it with the DEFINE command, to be prepared.

Example

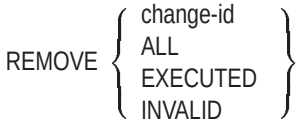
```
DRU> PREPARE MAKE_ONLINE
```

This example prepares a reload area for execution as previously defined in the MAKE_ONLINE change.

12.14 REMOVE Command

Removes one or more changes from the current change file for the currently bound database.

Format



Arguments

change-id
Specifies a unique name that identifies the change, as you defined it with the DEFINE command, to be removed from the current change file.

ALL
Removes all changes in the current change file.

EXECUTED
Removes all changes in the current change file that have been executed.

INVALID
Removes all changes in the current change file that are marked INVALID.

Examples

1.
```
DRU> REMOVE PARTS_CHANGES
1 change(s) removed
```

   This example removes the change PARTS_CHANGES from the current change file.

2.
```
DRU> REMOVE ALL
4 change(s) removed
```

   The current change file contains four changes. This example removes all four changes.

12.15 REVERSE Command

Rolls back a reload operation by reversing the direction of the reload from the target storage area back to the original storage area. This command can be issued only after the target storage area is prepared using the DRU PREPARE command, and before the database is cleaned up using the DRU CLEANUP command.

After the direction of the reload operation is reversed, you must issue the DRU EXECUTE and CLEANUP commands to complete the rollback of the reload. The DRU EXECUTE command is required even when records have not been reloaded, because new records might have been stored in the target area after the PREPARE command released the exclusive lock to the database.

The REVERSE command requires a brief moment of exclusive access to the database.

Format

```
REVERSE change-id
```

Arguments

change-id
Specifies a unique name that identifies the reload, as you defined it with the DEFINE command, to be reversed.

Example

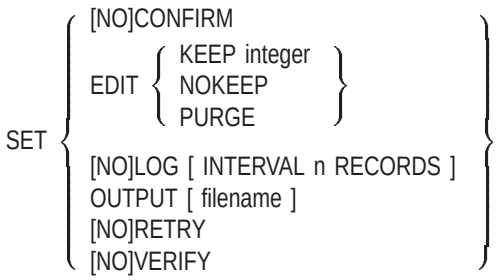
```
DRU> REVERSE MAKE_ONLINE
```

This example reverses the reload operation specified by the MAKE_ONLINE change.

12.16 SET Command

Lets you customize the environment of a DRU session.

Format



Arguments

[NO]CONFIRM
Enables or disables confirmation questions about DRU operations. The first EXECUTE command after each of the commands listed in Table 12–1 requests confirmation before continuing.

Table 12–1 DRU Confirmations

Command	Confirmation
EXECUTE	Is database backed up?
PREPARE	Is database backed up?
CLEANUP	Is database backed up?
REVERSE	Is database backed up?
EXECUTE offline	SEQUENCE not specified. Continue with default?
PREPARE	SEQUENCE not specified. Continue with default?
EXECUTE online	RELOAD definition different. Restart from beginning?
PREPARE	Metadata cannot be changed until CLEANUP. Continue?
EXECUTE offline	Online RELOAD available. Continue with offline RELOAD?

In a batch job, the confirmation message is not written to the log file, and the next line after the EXECUTE command is regarded as a reply.

To suppress the confirmation, either use the SET NOCONFIRM command before the EXECUTE command, or create a DRU initialization file with the SET NOCONFIRM command in it.

The default is the CONFIRM option.

EDIT
Allows you to control the number of lines saved in the edit save buffer. See the description of the DRU EDIT command for related information.

There are three forms of the SET EDIT command:

SET EDIT KEEP integer

Instructs DRU to save the specified number of previously issued DRU commands.

SET EDIT NOKEEP

Instructs DRU not to save any previously executed commands. This command is equivalent to SET EDIT KEEP 0.

SET EDIT PURGE

Purges all commands in the editing save buffer while retaining the value of the KEEP argument.

[NO]LOG [INTERVAL n RECORDS]
Displays the all messages during the processing of a DRU EXECUTE command. If you specify NOLOG, no messages are displayed. The default is LOG.

For reload, a log interval can be changed using INTERVAL. The default interval is 1000 records for the offline reload and 100 for the online reload.

OUTPUT [filename]
Specifies a file for DRU output. The default file type is .LIS. DRU creates either a new file with the specified name or a new version of an existing file. Certain output that normally appears on your screen is directed to the file you specify. Output is directed to the specified file on execution of any of the following DRU commands:

ANALYZE
EXECUTE
MACRO
SHOW

The default is to direct output to your terminal. If you previously specified an output file and you want to direct output back to your terminal, enter SET OUTPUT.

[NO]RETRY
Enables automatic execution retry during the processing of a DRU EXECUTE command. The NORETRY argument turns off automatic retry. The default is RETRY.

[NO]VERIFY
Echoes the contents of indirect command files as the files are executed. The default is the current value of the DCL SET VERIFY command for your process.

Example

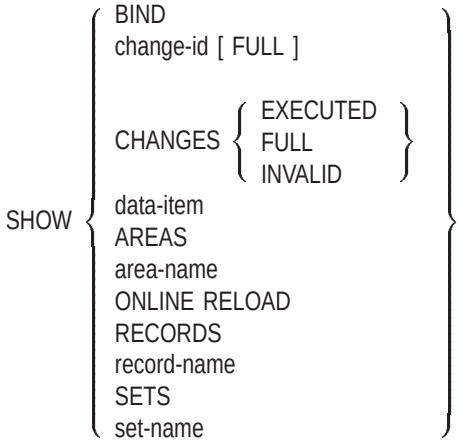
```
DRU> SET OUTPUT ANALYZED_CHANGES
DRU> ANALYZE WITHIN_CHANGE RETENTION_CHANGE ORDER_MODE_CHANGE
```

This example analyzes three changes and sends the output to the file ANALYZED_CHANGES.LIS.

12.17 SHOW Command

Displays information about your schema, storage schema, change file, or a reload operation in progress.

Format



Arguments

BIND
Displays information about the currently bound database.

change-id
Specifies a unique name that identifies the change, as you defined it in the DEFINE command, in the current change file.

change-id FULL
Displays the areas affected for the specified change.

CHANGES
Shows changes made to the DRU change file. Use one of the following options to display output:

- EXECUTED - Displays all changes in the current change file that have been executed for all changes in the current change file.
- FULL - Displays the areas affected by the current change file.
- INVALID - Displays all changes in the current change file that are marked INVALID.

data-item
Identifies a data item of the schema definition.

AREAS
Displays the names of the areas in your schema.

area-name
Identifies an area of the schema definition. All record types that exist in the area are displayed.

ONLINE RELOAD
Displays information about a reload operation in progress. (You can use the SHOW change-id or SHOW CHANGES form of this command to receive the same information if the change file can be opened and is not locked by other processes.)

RECORDS
Displays the names of the record types in your schema.

record-name
Identifies a record name of the schema definition.

SETS
Displays the names of the set types in your schema.

set-name
Identifies a set name of the schema definition.

Examples

1.
```
DRU> SHOW CHANGES FULL

Change file entries for DISK:[USER]PARTS_CHANGES.RCF;1

WITHIN_CHANGE - MODIFY CLASS
       NOWITHIN MAKE

        Areas affected :
                MAKE
                BUY

RETENTION_CHANGE - MODIFY ALL_PARTS_ACTIVE
       RETENTION IS FIXED

        Areas affected :
                MAKE
                BUY
```

   This example displays areas affected for the current change file.

2.
```
DRU> SHOW ALL_PARTS_ACTIVE
SET NAME IS ALL_PARTS_ACTIVE
    OWNER IS SYSTEM
    MEMBER IS PART
        INSERTION IS AUTOMATIC RETENTION IS OPTIONAL
    MODE IS CALC
        MEMBER IS PART
            KEY IS PART_DESC
```

   This example displays the schema and storage schema information for the ALL_PARTS_ACTIVE set.

3.
```
DRU> SHOW ONLINE RELOAD
!  Online reload status: has not been prepared yet or already cleaned up
```

   This example displays the status of a reload area operation in progress. The displayed text indicates that either the reload operation has not yet been prepared, or it has already been cleaned up.

12.18 STOP Command

Stops a reload area operation and terminates it gracefully. This command completes any in-flight record move, commits the transaction, and saves the dbkey of the current record to the root file. A stopped reload operation can be restarted at any time by issuing the DRU EXECUTE command.

The STOP command does not require a change-id, as other reload operation commands do, because the change file might be locked by the other process that is executing the reload.

Format

STOP [ONLINE RELOAD]

Example

```
DRU> STOP ONLINE RELOAD
```

This example stops an online reload operation in progress at the next commit point.

12.19 UNBIND Command

Terminates access to the current database and releases all resources that DBCS has acquired during the DRU session.

Format

UNBIND

Example

```
DRU> UNBIND
```

This command unbinds from the current database.

A

Configuration File Management and User Defined Events

Most database products provide tools for the purposes of analyzing the performance characteristics of the database and the application. These tools often require constant manual attention, or extensive post-processing of recorded data, to detect critical events that might be detrimental to the smooth operations of the database.

In some cases, the database administrator is not always available to constantly monitor the utility output, especially for 24 x 7 production databases. Post-processing of recorded data is certainly not timely enough to prevent or resolve potential realtime database downtime situations, particularly with real-world mission-critical requirements. The database administrator needs the ability to be "notified" automatically when a time-critical or interesting events occur on the database.

The DBO /SHOW /STATISTICS utility is one of the most powerful and useful tools available to the database administrator for the purposes of analyzing performance characteristics of a database. However, the ability to analyze performance problems is just one aspect of the role of a performance analysis tool. The database administrator needs to be able to detect problems in a timely manner, analyze the problem, and immediately institute corrective actions, also in a timely manner.

The DBO /SHOW /STATISTICS utility includes a "Configuration File" facility, which provides an extremely powerful and easy-to-use "User-Defined Event" facility that provide database administrators with the information necessary to detect, analyze and correct performance problems in the database.

The configuration file is used to persistently define user defined "events" that can react to the changing statistics being monitored for a database.

A.1 Configuration Files

The DBO /SHOW /STATISTICS utility provides the ability to import and export a configuration file at runtime. The configuration file is a human-readable file that was created or that can be automatically generated from the DBO Show Statistics utility using the current setup as the default values.

The DBO /SHOW /STATISTICS utility processes the configuration file prior to opening the database or the binary input file, if you specify the optional CONFIGURE=config_file qualifier. If you do not specify the CONFIGURE qualifier, all of the variables default to values based on command-line qualifiers and logical names. The default configuration file type is .CFG.

The configuration file is processed in two passes. The first pass occurs before the database is opened and processes most of the configuration file entries. The second pass occurs after the database is open, and processes those variables that are database dependent, such as the "CUSTOM_LINE_n" variables.

A.1.1 Creating Configuration Files

The DBO /SHOW /STATISTICS utility configuration file is created using the editor of your choice. The configuration file typically resides in the database directory, although it can reside anywhere the database administrator desires.

You can also manually export a new configuration file by selecting the Tools menu, which you obtain using the exclamation point (!), and then selecting the "Save current configuration" option. You will be prompted to enter the name of the new configuration file. The default configuration file type is .CFG. For portability, comments are generated using the pound sign (#) character. The configuration file will be generated using the current settings as the configuration variable values.

A.1.2 Configuration File Syntax

Each entry in the configuration file uses the following general format:

```
variable = value;
```

The equal sign (=) separating the variable and the value is required. Also note that each definition is terminated using a semicolon (;).

You can specify a comment using either the pound sign (#) or exclamation point (!) characters, and it continues to the end of the current line. A comment can occur anywhere in a line, but always terminates that line. Blank lines are ignored.

The "variable=value;" syntax is free-format. That is, spacing is not relevant to the parsing of the token. For example, the entry

```
STALL_LOG="STALL.LOG";
```

could be entered in the configuration file as:

```
STALL_LOG
="
STALL.LOG"
;
```

However, for readability reasons, this type of formatting is not recommended. Also, multiple entries can be put on a single line, although this is also not recommended, for readability reasons. For example, the following entry is permitted:

```
STALL_LOG="STALL.LOG"; DBKEY_LOG="DBKEY.LOG";
```

Really long configuration file entries can be continued on the next line by terminating the line with a back-slash (\). Lines can be continued multiple times, up to 2048 characters, even within quoted values; for example:

```
EVENT_DESCRIPTION="ENABLE 'pages checked'
 MAX_CUR_TOTAL \
 INITIAL 7 \
 EVERY 11 \
 LIMIT 100 \
 INVOKE DB_ALERT";
```

The line-continuation feature is not limited to the EVENT_DESCRIPTION variable; it can be used for any configuration variable. As an added bonus, comments can be embedded in continued lines if they start at the beginning of the next line. For example, consider the following two event descriptions:

```
EVENT_DESCRIPTION="ENABLE ' (Asynch. reads)'
 MAX_CUR_TOTAL \
! this will work as expected AREA EMPIDS_OVER \
 INITIAL 6 EVERY 10 LIMIT 100 INVOKE DB_ALERT";

EVENT_DESCRIPTION="ENABLE ' (Asynch. reads)'
 MAX_CUR_TOTAL \
 AREA EMPIDS_OVER ! this will NOT work as expected \
 INITIAL 6 EVERY 10 LIMIT 100 INVOKE DB_ALERT";
```

Note that the comment in the second event description takes precedence over the line continuation character.

A.1.3 Importing Configuration Files

A new configuration file can be imported at any time by selecting the Tools menu, which you can obtain using the exclamation point (!), and then selecting the "Import configuration settings" option. You will then be prompted to enter the name of the new configuration file. The default configuration file type is .CFG.

A.1.4 Nested Configuration Files

Configuration files can be nested to any depth, using the special "INCLUDE" variable. This command variable will be described in detail below. The use of nested configuration files allows you to create hierarchies of configuration variable definitions.

When using nested configuration files, remember that the last definition of a variable is the one that is available for use by the DBO /SHOW /STATISTICS utility. The use of nested configuration files is a good method for "grouping" related configuration file objects, such as events, into their own configuration file.

Note that, when including other configuration files, error messages for those included configuration files are displayed in the corresponding log file. For example, if the configuration file CONFIG.CFG includes the configuration file EVENTS.CFG, any errors or log messages pertaining to the EVENTS.CFG configuration file will be placed in the EVENTS.LOG log file, while all other log messages will be placed in the CONFIG.LOG log file.

A.1.5 Configuration Variable Semantics

The "variable" is either a pre-defined configuration parameter known to the DBO Show Statistics utility, or a user-defined variable whose value will presumably be used later as the value of some other pre-defined configuration parameter. The complete list of pre-defined variables is described in Section A.3, Configuration Parameters . All variable names must start with an alphabetic character. The "variable" name is case-insensitive.

All variables known to the DBO /SHOW /STATISTICS utility have defaults; these defaults are described later. It is not necessary to explicitly specify every variable. It is generally recommended that you only explicitly specify those variables whose default value is not acceptable to your application. Most configuration file variables replace existing command-line qualifiers or logical names.

A.1.6 Variable Types

Configuration variables have seven types. These types dictate how the corresponding atomic value is available for use. The variable types are:

- Numeric: This is the most common type of configuration variable. Some, but not all, numeric variables have minimum and maximum values. Also, some, but not all, numeric variables have scale values, which dictate the number of fractional digits allowed. For example, a variable with a scale of "2" can be specified in terms of hundredths. A scale of "0" designates only a whole number is allowed (0 decimal positions).
- Boolean: These types of variables can use the values TRUE and FALSE, or ENABLED and DISABLED interchangeably. These values are not quoted. You can also use the numeric values "0" and "1".
- String: String variables specify a quoted value, which are typically file names. Null string values are specified using an empty set of quotes ("").
- Date: Date variables specify a quoted value in the standard operating system date format.
- Command: Command variables cause an action specified by the quoted value to be performed. The most common variable is the "PRINT", "PROMPT" and "INQUIRE" variables. Command variables are not automatically exported.
- Control: Control variables are most commonly used to document interesting values. They do not actually do anything, but they can be useful as the value of another variable. The most common variable is the "DATABASE" variable.
- User Defined: User defined variables are used as the values of other types.

A.1.7 Configuration Value Semantics

The value is the numeric or quoted-string value to be assigned to the respective variable. The value, when quoted, is case-sensitive.

The values for numeric variables can be specified using either whole numbers or floating-point numbers, depending on the respective "scale" of the particular variable. The scale of a variable indicates the number of digits to the right of the decimal place. Obviously, if you specify a value whose fractional portion exceeds the scale, then the value will be truncated. For example, if you specify "4.25" for a variable whose scale is "1" (meaning 1 digit to the right of the decimal point)

then only "4.2" would be used. All values are truncated according to the specified scale.

A.1.7.1 Quoted Variable Evaluation

When a variable's value is quoted using double quotes (""), the evaluation of the value is treated as a hierarchy of values. The evaluation hierarchy is: variable name, logical name, atomic value. This hierarchy means that every value is first iteratively treated as another variable, then iteratively treated as a logical name, then finally as an atomic value.

Note that if you can also specify a single-word string value without the double quotes, if so desired. For example, the value "FOOBAR" could be specified using either double quotes or no quotes at all, since "FOOBAR" is a single-word string. However, the value "here and now" must be enclosed with either single- or double quotes. Quotes can also be nested, as long as they alternate.

A.1.7.2 Hierarchical Variable Evaluation

The hierarchical evaluation semantics allows you to specify the value of one variable as the value of another.

In the following example, the user-defined variable TEST is defined with the value "0". The pre-defined numeric variable CYCLE is defined to the value "TEST", which because it is a variable will be evaluated as the value "0". Finally, the pre-defined command variable PRINT will be evaluated as "CYCLE", which because it is also a variable, will be evaluated as the value "0".

```
TEST = 0;
CYCLE = "TEST";
PRINT = "CYCLE";
```

It is interesting to note that, even though the CYCLE variable is of type "numeric", you can specify a quoted string value as long as it ultimately evaluates to a numeric value.

It is also interesting to note that the hierarchical evaluation of values prevents a quoted value from being the same as the variable name. For instance, in the following configuration file entry the value "INCLUDE" obviously cannot be evaluated as the variable name "INCLUDE", since this would be recursive.

```
INCLUDE="INCLUDE";
```

A.1.7.3 Logical Name Evaluation

Using logical names as the value of variables is very useful for certain variables, such as the INCLUDE variable (see Section A.1.4, Nested Configuration Files). This allows users to customize their configuration files through the use of logical names while still utilizing a centralized configuration file hierarchy.

A.1.7.4 Variable Keyword Evaluation

Some variables have special keywords for their values. These keywords must be specified in upper-case, exactly as they are described in Section A.3, Configuration Parameters . Be especially careful when specifying user-defined variables whose name conflicts with any of the valid keywords. When using atomic values for keywords, it is recommended to use single quotes to avoid inadvertent evaluation of the keyword as another variable.

When a variable's value is quoted using single quotes ('), the value is treated as a simple string. For more information and examples on the difference between single quoted and double quoted strings, refer to Section A.1.7.5, Print and Prompt Command Variables.

A.1.7.5 Print and Prompt Command Variables

The PRINT and PROMPT variables are useful for debugging and tracing execution of the configuration file. For instance, the following example demonstrates how to display a variable's default value, and then the new value initialized from the configuration file:

```
PRINT="CYCLE";
CYCLE=5;
PRINT="CYCLE";
```

The output from the previous example is:

```
# This file was generated by the SHOW STATISTICS utility.
# Oracle Rdb on OpenVMS Release 7.4.0.0.0 - Production
# Generated on 15-AUG-2018 16:25:13.04
line 3: variable "PRINT" value "CYCLE = 0"
line 5: variable "PRINT" value "CYCLE = 5"
```

The PRINT variable is also useful for tracing execution of the configuration file, which may be very useful when using nested configuration files. For example, the PRINT command can be used to display simple messages to the log file:

```
PRINT = "Now initializing CYCLE variable";
CYCLE = 5;
```

The output from the previous example is:

```
# This file was generated by the SHOW STATISTICS utility.
# Oracle Rdb on OpenVMS Release 7.4.0.0.0 - Production
# Generated on 22-NOV-2019 12:29:14.13
line 1: variable "PRINT" value "Now initializing CYCLE variable"
```

Care should be taken when printing single-word messages, as these may be evaluated as variables. Consider the following example:

```
HERE = "there";
PRINT = "here";
PRINT = 'here';
PRINT = "here and there"
```

In this example, the value of the first PRINT variable is "there" not "here", because "here" is a user-defined variable. The value of the second PRINT variable is "here" because it is a single quoted simple string. Finally, the value of the third PRINT variable is "here and there" because it is an atomic string.

Note that the PROMPT variable does not work during an import operation.

A.1.7.6 Variable Value Redirection

Redirecting is an interesting use of a variable, especially a variable whose value is input by the user. Consider the following configuration file example:

```
OVER = "AND OUT";
THERE = "OVER";
HELLO = "THERE";
FOO = "HELLO";
```

Setting the variable "FOO" to the value "HELLO" causes an implicit redirection of its value to be the final value of all of its own value(s). This results in the "FOO" variable being set to the value "AND OUT", which is the final value of all values.

The REDIRECT command variable is extremely useful following an INQUIRE command, where the user entered the name of another variable to use. By redirecting the variable's value, you can essentially implement a menu mechanism. For example, consider the following configuration file example:

```
VALUE1 = "10";
VALUE2 = "20";
VALUE3 = "30";
PROMPT = "Enter VALUE1, VALUE2 or VALUE3";
INQUIRE = "VALUE";
REDIRECT = "VALUE";
PRINT = "VALUE";
```

Note how this differs from setting the VALUE variable to itself.

A.1.8 Log File

If any problem occurs processing the specified configuration file, a log file is automatically created. The log file's name is the full file specification of the configuration file with the .CFG file type replaced with a .LOG file type. For example, if the configuration file name is "CONFIG.CFG" then the corresponding log file name would be "CONFIG.LOG". Log file entries have the following format:

```
line #: message
```

The following example shows a problem trying to include a nested configuration file; the offending command is on line 4 of the configuration file.

```
# This file was generated by the SHOW STATISTICS utility.
# Oracle Rdb on OpenVMS Release 7.4.0.0.0 - Production
# Generated on 22-NOV-2019 12:35:14.13
line 4: INCLUDE="config.cfg";
line 4: command "INCLUDE" failed %COSI-E-FLK, file currently locked by another user
```

You can also specify the LOG qualifier, which lists each processed variable to the corresponding log file. This qualifier is highly recommended until you become more familiar with using the configuration files.

Note that, when including other configuration files, error messages for those included configuration files are displayed in the corresponding log file. For example, if the configuration file CONFIG.CFG includes the configuration file EVENTS.CFG, any errors or log messages pertaining to the EVENTS.CFG configuration file will be placed in the EVENTS.LOG log file, while all other log messages will be placed in the CONFIG.LOG log file.

A.1.9 Configuration File Example

The following is an example of a possible configuration file:

```
# This file was generated by the SHOW STATISTICS utility.
# Oracle Rdb on OpenVMS Release 7.4.0.0.0 - Production
DATABASE = "DISK$:[TESTER.WORK.STATS]MF_PERSONNEL.RDB;1"
# Generated on 10-OCT-2019 09:27:31.14
EVENT_DESCRIPTION="ENABLE transactions MAX_RATE INITIAL 5 EVERY 2 INVOKE EVENT";
EVENT_DESCRIPTION="DISABLE transactions MAX_CUR_RATE";
EVENT_DESCRIPTION="ENABLE transactions MIN_CUR_RATE INITIAL 5 EVERY 2 LIMIT 5";

ALARM = 0;
BROADCAST = FALSE;
CYCLE = 0;
INTERACTIVE = TRUE;
HISTOGRAM = FALSE;
REOPEN_INTERVAL = 0;
REFRESH_INTERVAL = 1;
NOTIFY = "OPER1,OPER11";
STALL_LOG = "stall.log";
TIMEOUT_LOG = "timeout.log";
DEADLOCK_LOG = "deadlock.log";
DBKEY_LOG = "";
TX_DURATION = "TOTAL";
LOGICAL_AREA = "INDIVIDUAL";
STALL_MESSAGE = "ACTUAL";
ACTIVE_USER = "ACTUAL";
CHECKPOINT_TX = "ACTUAL";
CHECKPOINT_SORT = "OLDEST_CHECKPOINT";
AIJ_ARBS_PER_IO = 99.9;
AIJ_BKGRD_ARB_RATIO = 50.5;
AIJ_BLKS_PER_IO = 2.5;
AIJ_SEC_TO_EXTEND = 60.5;
BTR_FETCH_DUP_RATIO = 15.5;
BTR_LEF_FETCH_RATIO = 25.5;
DBR_RATIO = 15.5;
FULL_BACKUP_INTRVL = 6;
GB_IO_SAVED_RATIO = 85.5;
GB_POOL_HIT_RATIO = 85.5;
LB_PAGE_HIT_RATIO = 75.5;
MAX_HASH_QUE_LEN = 2;
MAX_LOCK_STALL = 2.5;
MAX_TX_DURATION = 15.5;
PAGES_CHECKED_RATIO = 10.5;
RECS_FETCHED_RATIO = 20.5;
RECS_STORED_RATIO = 20.5;
RUJ_SYNC_IO_RATIO = 10.5;
VERB_SUCCESS_RATIO = 25.5;
```

A.2 User-Defined Events

An event is an identification of a particular statistic value upon which the DBO /SHOW /STATISTICS utility is to perform some user-defined action. In other words, an event is signaled when a statistic's value exceeds a user-defined set of thresholds.

Through the use of events, the database administrator is able to be automatically notified by a DBO /SHOW /STATISTICS utility server running on behalf of a particular database. An event is defined by specifying a threshold against a specific statistic, and optionally specifying the attributes the event is to have.

A.2.1 Event Definition Syntax

The DBO /SHOW /STATISTICS utility configuration file is a human-readable file, created using the editor of your choice. The configuration file typically resides in the database directory, although it can reside anywhere that you desire.

Each entry in the configuration file uses the general format variable=value; The equal-sign (=) separating the variable and value is required. Also note that each definition is terminated with a semicolon character (;).

User-defined events are specified using the EVENT_DESCRIPTION variable. Events themselves are not named; rather, they are defined on behalf of a specific statistic for a given threshold. The EVENT_DESCRIPTION variable's value is a free-format description of the user-specified event. The event definition consists of three required position-dependent components and an optional component; the following example describes the general format:

```
EVENT_DESCRIPTION="operation \
        statistic_name \
        threshold_name \
        [ attribute_list ]";
```

A.2.1.1 "Operation" Clause

The event "operation" clause identifies the action to be performed for the EVENT_ DESCRIPTION operation. The keyword ENABLE is available for use to enable a new event or change an existing event definition. The keyword DISABLE is available for use to disable a previously defined event; this keyword is typically used when importing a new configuration file.

Even though an event may be enabled, it may not be active. In order for an event to be active, you must also specify either a program to be invoked or one or more operator classes to be notified. This requirement is discussed in Section A.2.1.4.8 and Section A.2.1.4.9. At runtime, events can only be disabled by the DBO /SHOW /STATISTICS utility, or by importing a new configuration file that explicitly disables an event.

A.2.1.2 "Statistics Name" Clause

The event "statistic name" clause identifies the particular valid statistic field for which the event will be enabled or disabled. Note that some statistic names are only valid when certain database attributes are enabled, such as global buffers or row caching. The names of a particular statistic field can be found on the individual screen of interest. When statistic field names contain multiple words, such as "process attaches", it is required that the statistic name be either single or double quoted; failure to quote the statistic name may result in a syntax error.

Certain statistic fields have leading blanks. This whitespace is considered part of the statistic name, and is necessary to identify a unique statistic. However, the use of leading blanks is often difficult to discern in the configuration file. Therefore, the underscore character (_) or dash character (-) can be used in place of spaces in statistic names that have leading spaces. For example, the statistic field name " file extend" can also be specified as "_ _ file_extend" or "- - file-extend". This is useful for improving the readability of difficult statistic field names.

Aggregated Statistic Names: Most general events are defined using the summary statistics screens. However, it is sometimes necessary to define an event on a specific table or index, or even a partition of a table. This type of drill-down event is specified using the AREA attribute (described later) to specify the name of a particular storage area. When this clause is specified, the statistic field selected must be from the "IO Statistics (by file)" or "Locking Statistics (by file)" screens. The identified statistic name can also be qualified with the LAREA attribute (described later) to specify the name of a particular logical area, such as a table, b-tree index, hash index, or blob. When this clause is specified, the statistic field selected must be from the "Logical Area" screens. Furthermore, if the selected logical area is partitioned across multiple storage areas, the AREA clause can also be used to identify a specific partition to define the event against.

The following table explains the semantics of specifying the AREA and LAREA clauses to qualify a statistic field name:

Table A–1 AREA and LAREA clauses

AREA	LAREA	Description
No	No	General Statistic Field
Yes	No	Storage Area Statistic Field
No	Yes	Logical Area Statistic Field, all partitions
Yes	Yes	Logical Area Statistic Field, single partition

The AREA and LAREA clauses are attributes and, as such, must follow the "Threshold Name" component of the event definition.

Unnamed Statistics: Sometimes, it is necessary to define an event for an unnamed statistic, such as the transaction duration information. This is achieved using the special "CELL number" statistic name syntax. By specifying a cell number instead of a statistic name, you are able to specify events on any statistic, even if the statistic name is not available. You can also use the ANNOTATE attribute to specify a statistic name for the selected cell number. If you do not use the ANNOTATE attribute, the statistic name will be the specified text "CELL number".

The cell numbers are documented in the text file named SYS$LIBRARY:RMU$SHOW_STATISTICS74.CDO. For example, the comment describes the statistic and includes the cell number in brackets.

```
define field RMU$RT_VERB_SUCCESS
        description is /* total verbs executed                          [2] */
        datatype SIGNED QUADWORD.
```

A.2.1.3 "Threshold Name" Clause

The event "threshold name" clause identifies the particular event threshold for which the specified statistic will be enabled or disabled. The thresholds are essentially the columns in the numeric version of the statistics screens.

Up to eight different thresholds can be specified for a particular statistic field, although each individual event name must be specified in its own EVENT_ DESCRIPTION variable definition. The "threshold name" clauses are the following:

- MAX_RATE: The maximum current occurrence-per-second rate collected. This threshold only increases as each event is signaled.
- MAX_CUR_TOTAL: The maximum total value collected since the database was opened. This threshold only increases as each event is signaled.
- MIN_CUR_RATE: The lowest rate currently being sustained. This threshold remains constant.
- MAX_CUR_RATE: The highest rate currently being sustained. This threshold remains constant.
- MIN_AVG_RATE: The lowest average rate. This threshold only decreases as each event is signaled.
- MAX_AVG_RATE: The highest average rate. This threshold only increases as each event is signaled.
- MIN_PER_TX: The lowest per-transaction rate. This threshold only decreases as each event is signaled.
- MAX_PER_TX: The highest per-transaction rate. This threshold only increases as each event is signaled.

A.2.1.4 "Attribute List" Clause

The optional event "attribute list" clause provides additional characteristics for enabled event thresholds; in general, these attributes are ignored when disabling an event. Any or all of the event attributes can be specified for each event name within the same EVENT_DESCRIPTION variable definition. The attribute list clauses are the following:

A.2.1.4.1 AREA storage_area_name Defines the name of a particular storage area. When this clause is specified, the statistic field selected must be from the "IO Statistics (by file)" or "Locking Statistics (by file)" screens, unless the LAREA clause is also specified. This clause is used when disabling the event.

A.2.1.4.2 LAREA logical_area_name Defines the name of a particular logical area, such as a table, b-tree index, hash index, or blob. When this clause is specified, the statistic field selected must be from the "Logical Area" screens. This clause is used when disabling the event. This clause cannot be used if the NOLOGICAL_AREA qualifier is specified.

A.2.1.4.3 INITIAL value Defines the initial value of the current event threshold; the default value is "0" for MAX_xxx thresholds and "very big number" for MIN_xxx thresholds. The default value guarantees that at least one event will be signaled, thereby initializing the new current threshold value.

A.2.1.4.4 EVERY value Defines the value by which the initial threshold will be incremented or decremented when an event is signaled. If this value is "0", the default value, for any event except the MIN_CUR_RATE and MAX_CUR_RATE events, then the event will be signaled just once.

A.2.1.4.5 LIMIT value Defines the maximum number of times the event may be signaled. If the value is "0", which is the the default, events may be signaled indefinitely if the EVERY clause is specified with a non-zero value.

A.2.1.4.6 SKIP value Defines the number of event notifications to ignore before performing an actual notification. This clause is useful for the MIN_CUR_RATE and MAX_CUR_RATE events, as the thresholds for these events are not reset upon being signaled. The default value of "0" ensures that all events are notified.

A.2.1.4.7 ANNOTATE stat_name Defines the name of a statistic specified using the "CELL number" syntax. The maximum length of the specified statistic name is 18 characters. This attribute can also be used even if a statistic name is specified instead of its cell number.

A.2.1.4.8 NOTIFY oper_class_list Defines the quoted comma-separated list of operators to be notified for all events defined on the specified statistic. Valid operator keywords are CENTRAL, DISKS, CLUSTER, SECURITY and OPER1 through OPER12.

A.2.1.4.9 INVOKE program_name Defines the user-supplied program to be invoked for all events defined on the specified statistic. The program name is specified as a DCL process global symbol known to the DBO /SHOW /STATISTICS utility.

A.2.1.5 Description Readability

Really long entries in the configuration file lines can be continued on the next line by terminating the line with a back-slash (\). The continuation character as the last character of a line is available for use to indicate that the configuration entry is continued on the next line exactly as if it were entered as a single line. Lines can be continued practically indefinitely, up to 2048 characters, even within quoted values. The following example demonstrates how to define a multi-line event description:

```
EVENT_DESCRIPTION="ENABLE 'pages checked' \
        MAX_CUR_TOTAL \
        INITIAL 7 \
        EVERY 11 \
        LIMIT 100 \
        INVOKE DB_ALERT";
```

The continuation character is not limited to the EVENT_DESCRIPTION variable; it can be used for any configuration variable. Comments can be embedded in continued lines if they start at the beginning of the next line. The following example demonstrates two event descriptions containing embedded comments. The comment in the second event description takes precedence over the line continuation character:

```
EVENT_DESCRIPTION="ENABLE ' (Asynch. reads)' \
        MAX_CUR_TOTAL \
        AREA EMPIDS_OVER \
! this will work as expected
        INITIAL 6 EVERY 10 LIMIT 100 \
        INVOKE DB_ALERT";

EVENT_DESCRIPTION="ENABLE ' (Asynch. reads)' \
        MAX_CUR_TOTAL ! this will NOT work as expected \
        AREA EMPIDS_OVER \
        INITIAL 6 EVERY 10 LIMIT 100 \
        INVOKE DB_ALERT";
```

A.2.2 Event Semantics

In order for an event to be active, you must specify either one or both of the NOTIFY or INVOKE attribute clauses. When using the INVOKE attribute clause, the program must be specified by defining a process-global symbol pointing to the DCL command procedure or image to be invoked. The INVOKE program and NOTIFY operator classes apply to all events defined for the statistic field. Therefore, these clauses only need to be defined once per statistic field, no matter how many events thresholds are defined for that statistic. Specifying multiple programs or operator classes results in only the last specified attribute being used.

Once an event has been signaled, it will only be re-signaled if the EVERY attribute clause was specified with a non-zero value. The current threshold value, originally initialized to the INITIAL value, will be increased for MAX_xxx thresholds and decreased for MIN_xxx thresholds. The exception to this rule is the "current rate" thresholds MIN_CUR_RATE and MAX_CUR_RATE, which are never increased nor decreased. The MIN_xxx thresholds disable themselves once the INITIAL value reaches "0", while the MAX_xxx thresholds never disable themselves.

Once an event has been disabled, it can only be re-enabled by importing a new configuration file or manually using the "Statistics Event Information" screen "Re-enable all disabled events" configuration submenu option; individual events cannot be re-enabled online.

A.2.3 How User-Defined Events Work

The user-defined events are analyzed by the DBO /SHOW /STATISTICS utility at the specified screen refresh rate. The default screen refresh rate of three seconds is ideal for most databases, but using a one-second refresh rate will produce a finer granularity event signaling mechanism. Multiple events defined for the same statistic field may cause the specified program to be invoked multiple times, once for each affected event.

As the DBO /SHOW /STATISTICS utility identifies a statistic field whose current value or average value is changing, it examines any defined event thresholds established for that statistic field. This manner of examination minimizes the impact of event analysis, since the analysis is performed as part of the normal statistics collection process. When the utility determines that a specified event threshold has been exceeded, an event is signaled. The signaling of the event means that any specified programs will be invoked and any specified operators will be notified. The event notification occurs immediately.

If you defined a program that will be invoked when an event is signaled, the program will be invoked with eight parameters. Some of the parameters contain multiple words; be sure to quote them if the parameters are passed to other utilities. The parameters passed to invoked programs are the following:

Table A–2 Invoked Program Parameters

Parameter	Description
P1	the "date and time" the event occurred. This parameter contains embedded blanks.
P2	the statistic field name. This parameter may contain embedded blanks.
P3	the event name.
P4	the current event numeric value, expressed to the nearest tenth.
P5	the word "above" or "below."
P6	the current event threshold value.
P7	the event occurrence count.
P8	the optional physical area and/or logical area name for the statistic field.

The following example contains log file sample output where an event for a partitioned logical area was signaled. Note that when both the storage area and logical area names are specified, they are separated by a period (.).

```
pages checked MAX_CUR_TOTAL 6.0 above 4.0 count is 1 area is
EMPIDS_MID.EMPLOYEES

pages checked MAX_CUR_TOTAL 32820.0 above 5.0 count is 1 area is
EMPIDS_OVER.EMPLOYEES
```

A.2.3.1 Run-time Event Status Information

The current runtime status of the user-defined events can be examined using the new "Statistics Event Information" screen, located in the "Database Parameter Info" submenu. Note that you do not have to be viewing this screen to signal events. Note that the physical area and logical area identifiers are only displayed in "Full" mode.

A.2.4 User-Defined Event Example

Nothing demonstrates a feature better than a real-life example explained in step-by-step detail.

For the purposes of this example, suppose the database administrator wants to be sent email whenever a database freeze occurs. A database freeze occurs when an application process on the database prematurely terminates (i.e. "dies"). Such an event results in all application activity being temporarily suspended until the recovery operation for the terminated process has been completed. This is a very significant and serious runtime event that should be immediately detected.

Using events, notifying the database administrator when a process terminates prematurely is very easy to accomplish. The following steps describe how this can be achieved using the DBO /SHOW /STATISTICS "User-Defined Events" feature:

- Identity the Statistic Name: You are going to define a new event, so you need to specify the ENABLE operation keyword.

- Identity the Operation: A database freeze is represented by the "process failures" statistic on the "Recovery Statistics" screen, which is located in the "Journaling Information" submenu. This statistic is available even if you are not using after-image journaling.
- Identity the Threshold Name: The current number of processes that have failed, thereby causing a database freeze, is represented by the MAX_CUR_ TOTAL threshold.
- Identity the Event Attributes: This is probably the hardest part of defining an event. You want to be alerted to any process failure, so you have to set the INITIAL attribute to "0". Since you want to be notified on each and every process failure, then you have to set the EVERY attribute to "1" and the LIMIT attribute to "0".
- Define How You Will be Alerted: Since you want to be sent mail, use the INVOKE clause. Invoking a program on OpenVMS requires that you define a DCL process-global symbol to identify the actual DCL script, as the following example shows:

```
DBR_LOGGER == "@SYS$SYSTEM:DBR_LOGGER.COM"
```

- Write the Program: Since you want to be sent mail with a nice description of what event was actually signaled, use the simple DCL script in the following example:

```
$ open /write dbr_logger sys$scratch:dbr_logger.tmp
$ write dbr_logger " ’’’p1’ ’’p2’ ’’p3’ ’’p4’ ", -
        " ’’p5’ ’’p6’ (count is ’’p7’) area is ’’p8’ "
$ close dbr_logger
$ mail sys$scratch:dbr_logger.tmp -
        RDB_DBA_USER -
        /subject="DBR notification"
$ delete /nolog sys$scratch:dbr_logger.tmp;*
$ exit
```

- Create the Configuration File: The following example contains the final event description as you would enter it in the configuration file:

```
EVENT_DESCRIPTION="ENABLE ’process failures’ \
        MAX_CUR_TOTAL \
        INITIAL 0 EVERY 1 LIMIT 0 \
        INVOKE DBR_LOGGER";
```

- Invoke the DBO /SHOW /STATISTICS Utility: Be sure to use the CLUSTER command qualifier if you want to be notified of cluster-wide events. The following example demonstrates the command to perform this:

```
$ DBO/Show Statistics -
        /CONFIG=CONFIG.CFG -
        /NOINTERACTIVE -
        /UNTIL=15:00:00 -
        MF_PERSONNEL
```

A.2.5 Statistics Event Information Screen

This screen displays information about user-defined events and resides in the "Database Parameter Info" menu. These are not database parameters in the tradition sense; however, they control the operation of the DBO /SHOW /STATISTICS utility, so they are presented as a database parameter. You cannot use the information contained in this screen on the "Custom Statistics" screen.

The user-defined event information is presented in no particular order.

A.2.5.1 Screen Fields

The following screen fields are available when both the "Brief" and "Full" on-screen menu options are selected:

- Statistic: This field identifies the name of the statistic for which the event is defined.
- Event: This field identifies the name of the event for the statistic.
- State: This field identifies the state of the event. The state field has the following values: ENABLED if the event is active and DISABLED if the event is no longer active.
- Threshold: This field defines the "current" event threshold; the default value is "0" for MAX_xxx thresholds and "very big number" for MIN_xxx thresholds. The default value guarantees that at least one event will be signaled, thereby initializing the new "current" threshold value.
- Every: This field defines the value by which the initial threshold will be incremented or decremented when an event is signaled. If this value is "0", the default value, for any event except the MIN_CUR_RATE and MAX_CUR_RATE events, then the event will be signaled just once.
- Current: This field identifies the current value of the statistic field. This is presented for reference only.
- Cnt: This field identifies the number of times the user-defined event as been triggered.

The following fields are available only when the "Full" on-screen menu option is selected:

- Program or Operator Notification: This field defines the user-supplied program to be invoked for all events defined on the specified statistic. The program name is specified as a DCL process global symbol known to the DBO /SHOW /STATISTICS utility. If a program is not specified, this field defines the quoted comma-separated list of operators to be notified for all events defined on the specified statistic. Valid operator keywords are CENTRAL, DISKS, CLUSTER, SECURITY and OPER1 through OPER12.
- Parea: This field identifies the storage area identifier for events defined on partitioned logical area or specific storage area statistics. The value "0" indicates the storage area is not used.
- Larea: This field identifies the logical area identifier, if any. The value "0" indicates the logical area is not used.

- Rem: This field defines the number of event notifications remaining before performing an actual notification. This clause is extremely useful for the MIN_CUR_RATE and MAX_CUR_RATE events, as the thresholds for these events are not reset upon being signaled. The default value of "0" ensures that all events are notified.
- Limit: This field defines the maximum number of times the event may be signaled. If the value is "0", the default value, events may be signaled indefinitely, if the EVERY clause is specified with a non-zero value.

A.2.5.2 On-Screen Menu Options

This section discusses the on-screen menu options available to this screen. Remember that the Exit, Help (?), Menu, Set_rate, Write and (!) on-screen menu options are available on all screens.

- Brief: You select brief mode by typing "B". In brief mode, one line per event is displayed, providing the following information: statistic name, event name, event state, threshold value, every value, current value and trigger count.
- Full: You select full mode by typing "F". In full mode, two lines per event is displayed, providing the following information in addition to the information displayed for the brief mode: event program or operator notification classes, storage area identifier, logical area identifier, remaining count and limit value.

A.2.6 Summary

As applications increasingly require 24 x 7 availability, the rate at which database administrators are expected to react to potential downtime situations also increases. The "User-Defined Events" feature provides the means by which database administrators are able to be automatically alerted when such critical situations arise, thereby allowing timely corrective actions.

A.3 Configuration Parameters

The following is the complete list of configuration parameters, in alphabetical order. For more information on these variables and their respective semantics, please refer to the "Online Analysis" facility chapter in Volume 1 of the *RMU Show Statistics DBA Handbook: Methods and Internals*.

Table A–3 Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
ACTIVE_USER	String				
	Specifies the "Active User Stall Messages" screen configuration options. The valid keywords are ACTUAL and ELAPSED.				
AIJ_ARBS_PER_IO	Numeric	2	0		1

(continued on next page)

Table A–3 (Cont.) Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
	Specifies the default online analysis value of AIJ request blocks per AIJ I/O. The DBM$BIND_STATS_AIJ_ARBS_PER_IO logical name allows you to override the default value of AIJ request blocks per AIJ I/O. You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "AIJ Analysis" screen.				
AIJ_BKGRD_ARB_RATIO	Numeric	50	0		1
	Specifies the default online analysis value for the background AIJ request block threshold. The DBM$BIND_STATS_AIJ_BKGRD_ARB_RATIO logical name allows you to override the default value for the background AIJ request block threshold. You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "AIJ Analysis" screen.				
AIJ_BLKS_PER_IO	Numeric	2	0		1
	Specifies the default online analysis value of blocks per AIJ I/O. The DBM$BIND_STATS_AIJ_BLKS_PER_IO logical name allows you to override the default value of blocks per AIJ I/O. You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "AIJ Analysis" screen.				
AIJ_SEC_TO_EXTEND	Numeric	60	0		1
	Specifies the default online analysis value of seconds to AIJ extend. The DBM$BIND_STATS_AIJ_SEC_TO_EXTEND logical name allows you to override the default value of seconds to AIJ extend. You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "AIJ Analysis" screen.				
ALARM	Numeric	0	0		0
	Establishes an alarm interval (in seconds) for the Stall Messages screen from the command line. This is useful when you plan to submit the DBO /SHOW /STATISTICS utility as a batch job. This variable supersedes the ALARM=seconds qualifier.				
AUTO_ACTIVE_DETECT	Boolean	FALSE	FALSE	TRUE	
	When the AUTO_NODE_DETECT variable is set to the value TRUE, specifies whether or not to actively or passively detect new nodes joining the cluster. Active detection may incur an I/O operation per screen refresh. Passive detection relies on other users on the current node to passively refresh the node information.				
AUTO_NODE_DETECT	Boolean	FALSE	FALSE	TRUE	
	Specifies whether or not to automatically detect new nodes joining the cluster. When the utility detects a new node joining the cluster, the cluster statistics menu will be automatically displayed.				
AUTO_RECONNECT	Boolean	FALSE	FALSE	TRUE	
	Specifies whether or not to automatically reconnect to disconnected nodes.				

(continued on next page)

Table A–3 (Cont.) Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
BROADCAST	Boolean	TRUE	FALSE	TRUE	
	Specifies whether or not to broadcast messages. The Broadcast variable is enabled by default, if broadcasting of certain messages has been enabled with DCL SET BROADCAST. If broadcasting has been disabled with the DCL SET BROADCAST=NONE command, broadcast messages are not displayed, even if you specify the DBO /SHOW /STATISTICS utility with the Broadcast variable. This variable supersedes the [NO]BROADCAST qualifier.				
BTR_FETCH_DUP_RATIO	Numeric	15	0		1
	Specifies the default online analysis value for when the average number of sorted index duplicate node fetches exceeds this threshold. The default sorted index duplicate fetch threshold is 15%. You can also override the default value with the DBM$BIND_STATS_BTR_FETCH_DUP_RATIO logical name. You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Index Analysis" screen.				
BTR_LEF_FETCH_RATIO	Numeric	25	0		1
	Specifies the default online analysis value for when the average number of sorted index leaf-node retrievals is less than the corresponding threshold. This may be an indication that the sorted index contains too many levels. You may want to adjust the fan-out factors during index creation to use a larger sorted that contains fewer levels. This may help improve overall sorted index throughput and utilization. The default lock stall threshold is 25%. You can also override the default value with the DBM$BIND_STATS_BTR_LEF_FETCH_RATIO logical name. You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Index Analysis" screen.				
BTR_STORE_DUP_RATIO	Numeric	15	0		1
	Specifies the default online analysis value for when the average number of sorted index duplicate node stores exceeds this threshold. The default sorted index duplicate store threshold is 15%. You can also override the default value with the DBM$BIND_STATS_BTR_STORE_DUP_RATIO logical name. You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Index Analysis" screen.				
CHECKPOINT_ALARM	Numeric	0	0		0
	Establishes an alarm interval (in seconds) for the Checkpoint Information screen. This is useful when you plan to submit the DBO /SHOW /STATISTICS utility as a batch job.				
CHECKPOINT_BLOCK_COUNT	Numeric	512	0		0

(continued on next page)

Table A–3 (Cont.) Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
	Checkpoint Old. This analysis determines whether or not the checkpoint size exceeds a user-specified threshold, expressed in AIJ blocks. The number of AIJ blocks indicates a physical process recovery duration, but also impacts other components, such as AIJ backup and Row Cache. This configuration variable can be defined to specify a different threshold in the configuration file. The logical DBM$BIND_STATS_CHECKPOINT_BLOCK_COUNT can be defined to specify a different threshold at utility startup.				
CHECKPOINT_SORT	String Specifies the "Checkpoint" screen sort configuration options. The valid keywords are OLDEST_CHECKPOINT, OLDEST_TRANSACTION and OLDEST_QUIET_POINT.				
CHECKPOINT_TX	String Specifies the "Checkpoint" screen transaction configuration options. The valid keywords are ACTUAL and ELAPSED.				
CLUSTER_NODES	String Identifies the set of nodes that are to participate in statistics collection for the current session. The node name(s) should be comma separated. The keyword ALL_OPEN indicates that statistics should be collected from all nodes on which the database is currently open. Note that the ALL_OPEN keyword is never automatically generated. This variable supersedes the CLUSTER=node_list qualifier.				
CONFIRM_EXIT	Boolean When the CONFIRM_EXIT variable is set to the value TRUE, specifies that the user will confirm exiting from the utility. This variable is only useful when the INTERACTIVE variable is TRUE.	FALSE	FALSE	TRUE	
CUSTOM_LINE_n	String Specifies the name of the statistic field located on line "n of" the "Custom Statistics" screen. Statistics may be entered for lines "1" through "36", although the number of lines that can actually be displayed depends on your terminal. The statistic name must be specified exactly as it appears on its home screen, including leading spaces. Duplicate statistics as well as duplicated line numbers are detected. Note that the specified custom statistics fields are not evaluated until after the database has been opened. Opening the database activates the various screens, which determines the set of custom statistic fields that can be specified. Therefore, some custom statistics fields may not always be available, depending on which database attributes (for instance "global buffers) are active.			18	
CYCLE	Numeric Specifies the interval (in seconds) to automatically migrate to the next screen in the current sub-menu. This variable supersedes the CYCLE=seconds qualifier.	0	0		0
DASHBOARD_UPDATE	Boolean	DISABLED	DISABLED	ENABLED	

(continued on next page)

Table A–3 (Cont.) Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
	Specifies whether or not dashboard updates are permitted, if you have the proper privileges.				
	This variable supersedes the OPTION=[NO]UPDATE qualifier.				
DATABASE	Control				
	This control variable identifies the database from which the configuration file was generated. This variable is for documentation purposes only, and is primarily useful only when the LOG qualifier is specified.				
DBKEY_LOG	String			255	
	Logs the records accessed during a given processing period by the various attached processes. The file specification is the name of the file to which all accessed DBKEYs are logged. When importing a configuration file which specifies a log file, even if the same log file is specified, a new log file will be automatically created. This variable supersedes the DBKEY_LOG=dbkey_log qualifier.				
DBR_RATIO	Numeric	15	0		1
	Specifies the default online analysis value of the Database Recovery process ("DBR") invocation threshold.				
	The DBM$BIND_STATS_DBR_RATIO logical name allows you to override the default value of the Database Recovery process ("DBR") invocation threshold.				
	You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "RUJ Analysis" screen.				
DEADLOCK_FULL_DISPLAY	Boolean	DISABLED	DISABLED	ENABLED	
	Specifies whether or not the "Lock Deadlock History" screen is to display all processes or just those with deadlock messages.				
DEADLOCK_LOG	String			255	
	Specifies that lock deadlock message are to be written to the specified file. This can be useful when you notice a great number of lock deadlock messages being generated, but do not have the resources on hand to immediately investigate and resolve the problem. The file generated by the DEADLOCK_LOG variable can be reviewed later so that the problem can be traced and resolved.				
	When importing a configuration file which specifies a log file, even if the same log file is specified, a new log file will be automatically created. This variable supersedes the DEADLOCK_LOG=deadlock_log qualifier.				
EVENT_DESCRIPTION	Command				
	This command describes a Show Statistic "event" and either enables a new event or disables an active event. Please refer to Section A.2, User-Defined Events for more information.				
FULL_BACKUP_INTRVL	Numeric	6	0		0
	Specifies the online analysis full database backup threshold.				
	The DBM$BIND_STATS_FULL_BACKUP_INTRVL logical name allows you to override the full database backup threshold. You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Recovery Analysis" screen.				

(continued on next page)

Table A–3 (Cont.) Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
FULL_DISPLAY	Boolean	FALSE	FALSE	TRUE	
	Directs DBO to display full information displays for those screens that have the "Brief" and "Full" onscreen-menu options.				
GB_IO_SAVED_RATIO	Numeric	85	0		1
	Specifies the online analysis global buffer I/O-saved default threshold.				
	The DBM$BIND_STATS_GB_IO_SAVED_RATIO logical name allows you to override the global buffer I/O-saved default threshold.				
	You can also set the global buffer I/O-saved threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Buffer Analysis" screen.				
GB_POOL_HIT_RATIO	Numeric	85	0		1
	Specifies the online analysis global buffer pool hit default threshold.				
	The DBM$BIND_STATS_GB_POOL_HIT_RATIO logical name allows you to override the global buffer pool hit default threshold.				
	You can also set the global buffer pool hit threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Buffer Analysis" screen.				
HISTOGRAM	Boolean	FALSE	FALSE	TRUE	
	Directs DBO to display the initial statistics screen in the numbers play mode. The HISTOGRAM variable value TRUE specifies the graph display mode. The HISTOGRAM variable value FALSE specifies the numbers display mode.				
	This variable supersedes the [NO]HISTOGRAM qualifier.				
INCLUDE	Command				
	This command temporarily switches processing to the quoted string value that defines a new configuration file. This configuration file, in turn, may also specify the INCLUDE command to switch to yet another configuration file. When processing of the command completes, processing of the current configuration file continues. Remember that variable definitions are superseded by subsequent references. The INCLUDE variable detects infinite recursion. The INCLUDE variable is not exported.				
INQUIRE	Command				
	This command prompts the user to enter a value for the specified variable. This command is typically used after the PROMPT command to prompt the user to enter a value for certain variables.				
	This command does not work when a configuration file is imported. The INQUIRE variable is not exported.				
INTERACTIVE	Boolean	TRUE	FALSE	TRUE	

(continued on next page)

Table A–3 (Cont.) Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
	Displays the statistics dynamically to your terminal. The Interactive qualifier is the default when you execute the DBO /SHOW /STATISTICS utility from a terminal. You can use the NOINTERACTIVE qualifier with the Output qualifier to generate a binary statistics file without generating a terminal display. The NOINTERACTIVE qualifier is the default when you execute the DBO /SHOW /STATISTICS utility from a batch job.				
	Note that most of these variable are not interesting when the INTERACTIVE variable is set to FALSE. This variable supersedes the [NO]INTERACTIVE qualifier.				
LB_PAGE_HIT_RATIO	Numeric	75	0		1
	Specifies the online analysis LB/AS page hit default threshold.				
	The DBM$BIND_STATS_LB_PAGE_HIT_RATIO logical name allows you to override the LB/AS page hit default threshold. You can also set the local buffer pool hit threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Buffer Analysis" screen.				
LONG_TX_SECONDS	Numeric	1			0
	Specifies the long-running transaction threshold, expressed in seconds. The default value is “1” second.				
	DBO /SHOW /STATISTICS emits OPCOM messages to indicate transactions that exceed the interval specified by the LONG_TX_ SECONDS at intervals of 1 minute. The messages are delivered to the OPCOM classes specified by the NOTIFY variable in the configuration file. This is useful when you plan to submit the DBO /SHOW /STATISTICS utility in a batch job.				
MAX_HASH_QUE_LEN	Numeric	2	0		0
	Specifies the online analysis hash table queue length default threshold.				
	The DBM$BIND_STATS_MAX_HASH_QUE_LEN logical name allows you to override the hash table queue length default threshold.				
	You can also set the hash table queue length threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Transaction Analysis" screen.				
MAX_LOCK_STALL	Numeric	2	0		1
	Specifies the online analysis lock stall default threshold.				
	The DBM$BIND_STATS_MAX_LOCK_STALL logical name allows you to override the lock stall default threshold. You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Locking Analysis" screen.				
MAX_TX_DURATION	Numeric	15	0		1
	Specifies the online analysis transaction duration default threshold.				
	The DBM$BIND_STATS_MAX_TX_DURATION logical name allows you to override the transaction duration default threshold.				
	You can also set the transaction duration threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility Transaction Analysis screen.				
NOTIFY	String				

(continued on next page)

Table A–3 (Cont.) Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
	Notifies the specified system operator or operators when a stall process exceeds the specified alarm interval by issuing a broadcast message and ringing a bell at the terminal receiving the message. The valid operator classes are: CENTRAL, CLUSTER, DISKS, SECURITY, and OPER1 through OPER12. Multiple operator classes can be comma-separated; for example 'OPER11,OPER12'. Be sure to use single-quotes for this variable.				
	This variable supersedes the NOTIFY=oper_classes qualifier.				
ONLINE_ANALYSIS_LOG	String			255	
	The DBO /SHOW /STATISTICS utility provides an "Online Analysis" log file. The online analysis log file provides hard copy of all of the analysis performed by all of the "Online Analysis" facility screens, without having to actually display any of the screens.				
	This configuration variable identifies the name of the log file to contain the online analysis information.				
	There is no command qualifier to directly enable the online analysis log file; the configuration file should be used instead via the /CONFIG command qualifier.				
	The online analysis is performed at the specified screen refresh rate. It is possible to generate a considerable number of entries in the online analysis log file. Therefore, it is recommended that the online analysis log file be used in a non-interactive batch job with a reasonable refresh rate of five, ten or thirty seconds.				
OPCOM_LOG	String			255	
	This configuration variable identifies the name of the log file to contain to record various OPCOM messages broadcast by attached database processes.				
	If a single process is attached to the database multiple times, the OPCOM messages are displayed for the attach that issued the broadcast.				
	When recording OPCOM messages, it is possible to occasionally miss a few messages for a specific process. When this occurs, the message "n missed" will be displayed in the log file.				
OUTPUT	String			255	
	Specifies that the collected statistics are to be written to the specified binary output file.				
	This variable supersedes the OUTPUT=binary_file qualifier.				
PAGES_CHECKED_RATIO	Numeric	10	0		1
	Specifies the online analysis pages checked default threshold.				
	The default pages checked threshold is 10%. You can also override the default value with the DBM$BIND_STATS_PAGES_CHECKED_RATIO logical name.				
	You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Record Analysis" screen.				
PRINT	Command				

(continued on next page)

Table A–3 (Cont.) Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
	This command prints the "value" to the log file. This variable is useful if you do not want to use the LOG qualifier, but want to display the value of a variable. This variable is also useful for displaying the "initial" value of a variable before it is changed in the configuration file. Interesting uses of the PRINT command have already been described in this handbook. The PRINT variable is not exported.				
PROC_ASTLM_RATIO	Numeric	25	0		1
	Specifies the online analysis process ASTLM default threshold.				
	The DBM$BIND_STATS_PROC_ASTLM_RATIO logical name allows you to override the process ASTLM default threshold. You can also set the process ASTLM threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Process Analysis" screen.				
	Note: the configuration variable PROC_ASTLM is a synonym for PROC_ASTLM_RATIO but is now deprecated.				
PROC_BIOLM_RATIO	Numeric	25	0		1
	Specifies the online analysis process BIOLM default threshold.				
	The DBM$BIND_STATS_PROC_BIOLM_RATIO logical name allows you to override the process BIOLM default threshold. You can also set the process BIOLM threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Process Analysis" screen.				
	Note: the configuration variable PROC_BIOLM is a synonym for PROC_BIOLM_RATIO but is now deprecated.				
PROC_DIOLM_RATIO	Numeric	25	0		1
	Specifies the online analysis process DIOLM default threshold.				
	The DBM$BIND_STATS_PROC_DIOLM_RATIO logical name allows you to override the process DIOLM default threshold.				
	You can also set the process DIOLM threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Process Analysis" screen.				
	Note: the configuration variable PROC_DIOLM is a synonym for PROC_DIOLM_RATIO but is now deprecated.				
PROC_ENQLM_RATIO	Numeric	25	0		1
	Specifies the online analysis process ENQLM default threshold.				
	The DBM$BIND_STATS_PROC_ENQLM_RATIO logical name allows you to override the process ENQLM default threshold. You can also set the process ENQLM threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Process Analysis" screen.				
	Note: the configuration variable PROC_ENQLM is a synonym for PROC_ENQLM_RATIO but is now deprecated.				
PROC_PGFLLM_RATIO	Numeric	25	0		1

(continued on next page)

Table A–3 (Cont.) Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
	Specifies the online analysis process PGFLLM default threshold.				
	The DBM$BIND_STATS_PROC_PGFLLM_RATIO logical name allows you to override the process PGFLLM default threshold.				
	You can also set the process PGFLLM threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Process Analysis" screen.				
	Note: the configuration variable PROC_PGFLLM is a synonym for PROC_PGFLLM_RATIO but is now deprecated.				
PROMPT	Command				
	This command prints the "value" to the terminal. This variable is typically used in conjunction with the INQUIRE variable, but can also be useful if you do not want to use the LOG qualifier, but want to display the value of a variable.				
	This command does not work when a configuration file is imported. Interesting uses of the PROMPT command have already been described in this handbook. The PROMPT variable is not exported.				
RECOVERY_SORT	String				
	Specifies the "Transaction Recovery Duration Estimate" screen sort configuration options. The valid keywords are LONGEST_ TRANSACTION, LONGEST_UNDO, LONGEST_REDO and LONGEST_FREEZE.				
RECS_FETCHED_RATIO	Numeric	20	0		1
	Specifies the online analysis records fetched default threshold.				
	The DBM$BIND_STATS_RECS_FETCHED_RATIO logical name allows you to override the records fetched default threshold. You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Record Analysis" screen.				
RECS_STORED_RATIO	Numeric	20	0		1
	Specifies the online analysis records stored default threshold.				
	The DBM$BIND_STATS_RECS_STORED_RATIO logical name allows you to override the records stored default threshold. You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Record Analysis" screen.				
REDIRECT	Command				
	This command forces the redirection of another variable's value into itself. This is the means by which variable indirection can be achieved. Command variables and control variables cannot be redirected, but all other variables can be. For example, if the variable HELLO has the value "THERE", and the variable FOO has the value "HELLO", then the command REDIRECT="FOO" causes the variable FOO to have the value "THERE". This command variable is typically used after defining the variable's value using the INQUIRE command. The REDIRECT variable is not exported.				
REFRESH_INTERVAL	Numeric	3	0		0

(continued on next page)

Table A–3 (Cont.) Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
	Specifies the statistics collection interval in seconds. If you omit this qualifier, a sample collection is made every 3 seconds. The integer has a normal range of 1 to 180 (1 second to 3 minutes). However, if you specify a negative number for the Time qualifier, the DBO /SHOW /STATISTICS utility interprets the number as hundredths of a second. For example, REFRESH_INTERVAL="-20" specifies an interval of 20/100 or 1/5th of a second.				
	This variable supersedes the TIME=seconds qualifier.				
REOPEN_INTERVAL	Numeric	0	0		0
	After the specified interval, closes the current output file and opens a new output file without requiring you to exit from the DBO /SHOW /STATISTICS utility. The new output file has the same name as the previous output file, but the version number is incremented by 1.				
	This variable supersedes the REOPEN_INTERVAL=seconds qualifier.				
REPORT_INTERVAL	Numeric	30	0		0
	Specifies the automatic screen capture interval in seconds. If you omit this qualifier, automatic screen capture is disabled.				
RESET	Boolean	FALSE	FALSE	TRUE	
	Specifies whether or not the statistics are to be automatically reset prior to being displayed. This variable is always exported with the value FALSE regardless of its initial value.				
	This variable supersedes the [NO]RESET qualifier.				
RUJ_FILE_SIZE	Numeric	256	0		1
	RUJ File Size. This analysis determines whether or not the process RUJ file size exceeds a user-specified threshold, expressed in blocks. The number of RUJ blocks is an indicator a transaction recovery duration.				
	The default RUJ file size threshold is 256 blocks.				
	This configuration variable can be defined to specify a different threshold in the configuration file.				
	The logical name DBM$BIND_STATS_RUJ_FILE_SIZE can be defined to specify a different threshold at utility startup.				
RUJ_SYNC_IO_RATIO	Numeric	10	0		1
	Specifies the online analysis synchronous RUJ I/O default threshold.				
	The DBM$BIND_STATS_RUJ_SYNC_IO_RATIO logical name allows you to override the synchronous RUJ I/O default threshold.				
	You can also set this threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "RUJ Analysis" screen.				
SCREEN	String			255	
	Specifies that name of the initial screen to be displayed.				
	This variable supersedes the SCREEN=screen_name qualifier.				
SPAM_PAG_FET_RATIO	Numeric	80	0		1

(continued on next page)

Table A–3 (Cont.) Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
	Specifies the default online analysis value for when the average number of SPAM page fetches exceeds this threshold for the total number of SPAM page fetches. The percentage is computed using the total number of SPAM pages fetched to store a record divided by the total number of SPAM page fetches for any reason. In most real-world applications, new records stored exceed the number of record modifications or deletions. The default SPAM fetched threshold is 80%. You can also override the default value with the DBM$BIND_STATS_SPAM_PAG_FET_RATIO logical name. You can also set this threshold selecting the "SPAM pages fetched threshold" option from the configuration submenu.				
SPAM_REC_STO_RATIO	Numeric	20	0		1
	Specifies the default online analysis value of the SPAM records stored threshold. The default SPAM record stored threshold is 20%. You can also override the default value with the DBM$BIND_STATS_SPAM_REC_STO_ RATIO logical name. You can also set this threshold selecting the "SPAM records stored threshold" option from the configuration submenu.				
STALL_INVOKE	String			155	

	By default OPCOM notifications from DBO /SHOW /STATISTICS occur when a process stalls for more than the ALARM seconds. This configuration variable provides the ability to invoke a procedure from DBO /SHOW /STATISTICS when this event occurs.
	The user can specify the procedure to invoke by defining a DCL symbol that invokes the procedure just like one would do to invoke a procedure for a user defined event and assign it to STALL_INVOKE.
	Similar to INVOKE for user defined events, the following parameters give more information about the stall.
	P1 - gives the name of the database.
	P2 - gives the date and time of the invocation.
	P3 - gives the process identification of the stalled process.
	P4 - gives the stream identification.
	P5 - gives you the value of ALARM seconds.
	P6 - gives the resource waiting.
	P7 - is currently undefined.
	P8 - is currently undefined.
	Example DBO config file,
	`ALARM = 300;`
	`TIME = 60;`
	`STALL_INVOKE = "invoke_stall_cmd";`
	`INTERACTIVE = FALSE;`
	`NOTIFY = "1";`				

STALL_LOG	String			255	

(continued on next page)

Table A–3 (Cont.) Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
	Specifies that stall message are to be written to the specified file. This can be useful when you notice a great number of stall messages being generated, but do not have the resources on hand to immediately investigate and resolve the problem. The file generated by the STALL_ LOG variable can be reviewed later so that the problem can be traced and resolved. When importing a configuration file which specifies a log file, even if the same log file is specified, a new log file will be automatically created.				
	This variable supersedes STALL_LOG=stall_log qualifier.				
STALL_LOG_VERBOSE	Boolean	FALSE	FALSE	TRUE	
	When This configuration variable when set to the value TRUE, specifies that the Stall Messages Log file output will be in "verbose" mode, meaning that all stalled processes will be displayed at each refresh interval, instead of just once per stalled event.				
	This variable supersedes the OPTION=VERBOSE qualifier.				
STALL_MESSAGE	String				
	Specifies the "Stall Messages" screen configuration options. The valid keywords are ACTUAL and ELAPSED.				
TIMEOUT_FULL_DISPLAY	Boolean	DISABLED	DISABLED	ENABLED	
	Specifies whether or not the "Lock Timeout History" screen is to display all processes or just those with deadlock messages.				
TIMEOUT_LOG	String			255	
	Specifies that lock timeout message are to be written to the specified file. This can be useful when you notice a great number of lock timeout messages being generated, but do not have the resources on hand to immediately investigate and resolve the problem. The file generated by the TIMEOUT_LOG variable can be reviewed later so that the problem can be traced and resolved. When importing a configuration file which specifies a log file, even if the same log file is specified, a new log file will be automatically created.				
	This variable supersedes the TIMEOUT_LOG=timeout_log qualifier.				
TX_DBR_FREEZE_DURATION	Numeric	15	0		0
	Database freeze duration. This analysis determines whether or not the entire database freeze duration exceeds a user-defined threshold, expressed in seconds. The database freeze duration includes both transaction rollback, process recovery (transaction REDO) and DBR processing. The default database freeze threshold is 15 seconds.				
	This configuration variable can be defined to specify a different threshold in the configuration file.				
	The logical DBM$BIND_DBR_FREEZE_DURATION can be defined to specify a different threshold at utility startup.				
TX_DISPLAY	String				
	Specifies the "Transaction Duration" screen display interval. The valid keywords are MINUTES for 0-15 minutes and SECONDS for 0-15 seconds.				
TX_DURATION	String				

(continued on next page)

Table A–3 (Cont.) Configuration Parameters

Variable Name	Type	Default Value	Min. Value	Max. Value	Scale
	Specifies the "Transaction Duration" screen configuration options. The valid keywords are TOTAL, READ_WRITE and READ_ONLY.				
TX_REDO_DURATION	Numeric	10	0		0
	Process Recovery Duration. This analysis determines whether or not the recovery of a process failure (transaction REDO) exceeds a user-defined threshold, expressed in seconds. The default process recovery threshold is 10 seconds.				
	This configuration variable can be defined to specify a different threshold in the configuration file.				
	The logical DBM$BIND_TX_REDO_DURATION can be defined to specify a different threshold at utility startup.				
TX_UNDO_DURATION	Numeric	5	0		0
	Transaction Rollback Duration. This analysis determines whether or not the transaction rollback duration exceeds a user-defined threshold, expressed in seconds. The default transaction rollback threshold is 5 seconds.				
	This configuration variable can be defined to specify a different threshold in the configuration file.				
	The logical DBM$BIND_TX_UNDO_DURATION can be defined to specify a different threshold at utility startup.				
UNSET	Command				
	This command removes a user defined variable from the symbol table. It is not necessary to unset user defined variables prior to changing their value. It is only necessary to unset a variable if you do not wish it to exist for some reason. Note that using the single-quote is not supported for this command because variables can be specified using double-quotes only. The UNSET variable is not exported.				
VERB_SUCCESS_RATIO	Numeric	25	0		1
	Specifies the online analysis verb success default threshold.				
	The DBM$BIND_STATS_VERB_SUCCESS_RATIO logical name allows you to override the verb success default threshold. You can also set the verb success threshold from the configuration sub-menu in the DBO /SHOW /STATISTICS utility "Transaction Analysis" screen.				

Index

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y