Oracle Direct Seminar

ORACLE®

効果的な集計処理ことはじめ

日本オラクル株式会社

Oracle Direct

Oracle Direct Seminar

アジェンダ

- 集計処理今昔
- Oracle Databaseの集計処理
- 効果的な集計処理
- まとめ

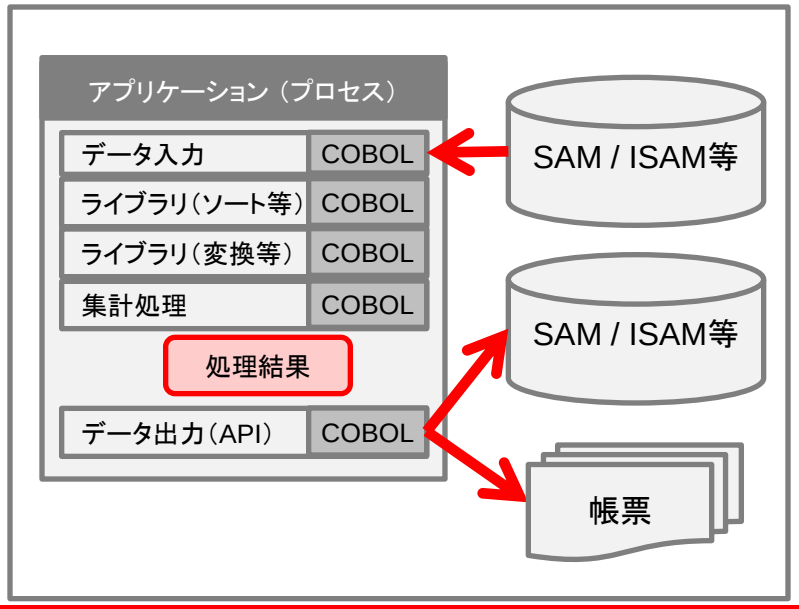
Copyright© 2011, Oracle. All rights reserved.

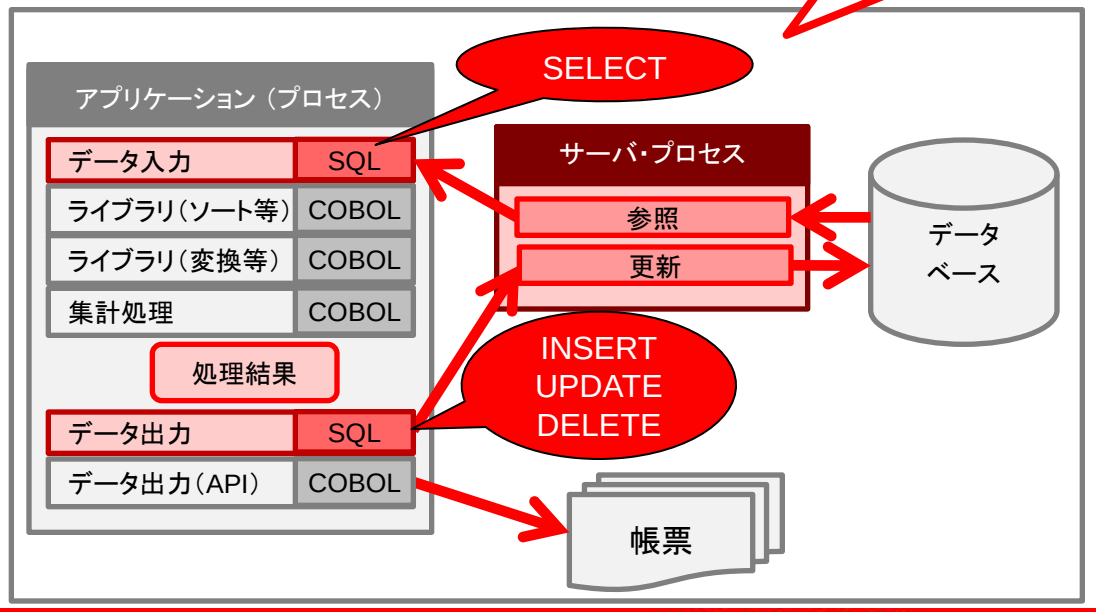
集計処理今昔

RDBMSを利用しない集計処理
初期のRDBMSを利用した集計処理（Pro*C、Pro*COBOLアプリ）

- サーバ上で動作可能な実行モジュールを作成
  - COBOLやC言語などのコンパイラ言語で記述されることが多い
  - サーバ側で提供されるライブラリやAPIを利用してアプリケーションを作成
  - 集計処理のロジックをコンパイラ言語で記述
  - データを1件づつ読み込んで処理をおこなうことが多い（直列処理）

SQLで
『データの集合』
を扱っていない



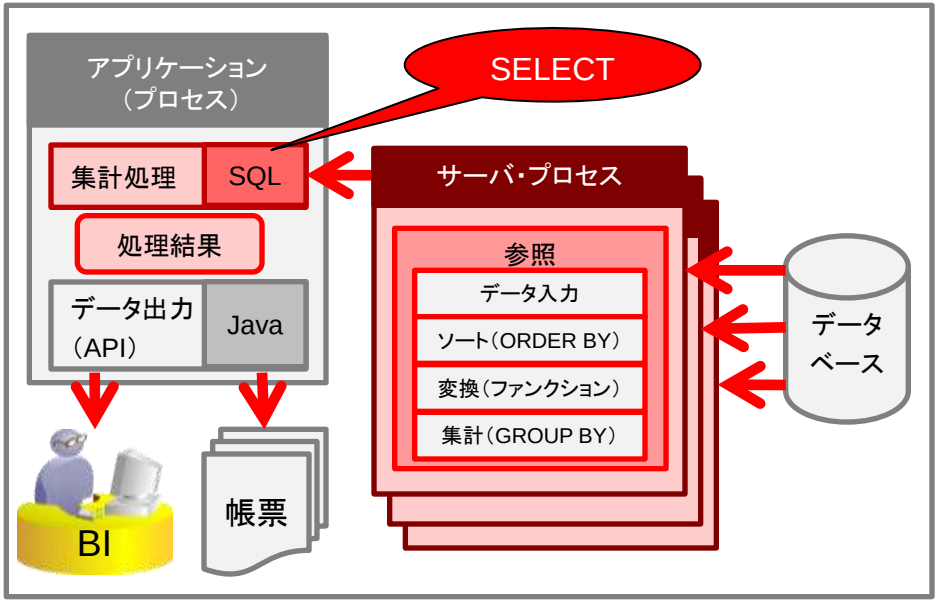


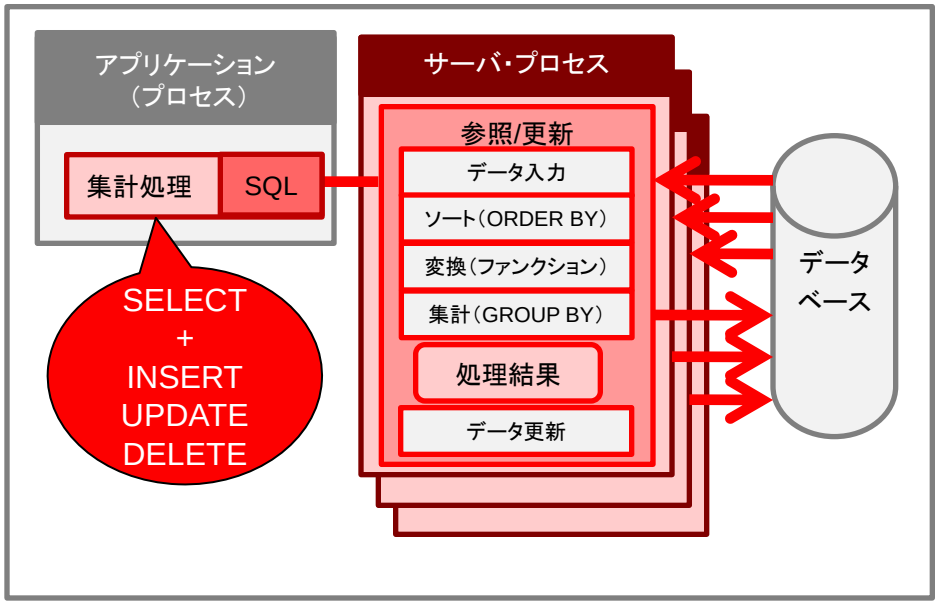
Copyright© 2011, Oracle. All rights reserved.

集計処理今昔

SQLを活用した集計処理：SQLで『データの集合』を扱う

- SQLを実行可能なインターフェースがあれば特に実行モジュールの形式は問わない
  - 集計ロジックをSQLに移管することによる、アプリケーションのメンテナンス性向上
  - H/W、RDBMS側の進化（パラレル処理等による性能向上）の恩恵を最大限享受
  - 集計処理結果を様々な方法で容易に利用可能（Business Intelligenceなど）





Copyright© 2011, Oracle. All rights reserved.

集計処理今昔
差異のまとめ

	これまでの集計処理	SQLを活用した集計処理
アプリケーション開発言語	コンパイラ言語（C、COBOL） プリコンパイラ製品（Pro*C、Pro*COBOL）	SQLを実行できる言語またはインターフェース（SQL*Plus等）
集計ロジック	コンパイラ言語で記述されることが多い	SQLで表現
SQL実行回数	集計処理中のループ処理内にて何度も実行	1～数回
処理の実行形態	単体での実行 処理範囲を分割することによる並列処理も可能だが、分割単位を細かく設定する必要あり	Oracle Databaseのパラレル機能による並列処理が可能
RDBMSの進化への適合	RDBMSの基本機能を利用 単純なSQLの繰り返し処理が多く、基本的には索引処理の繰り返し	SQLのオプティマイザ性能向上、パラレル機能による並列処理、Exadataを利用することによる劇的な速度向上が見込める
H/W進化への適合	プロセッサ速度の向上による処理時間短縮	プロセッサ速度の向上、およびマルチCPU、マルチCore化に対する並列処理による処理時間短縮
利用形態	帳票を介した利用形態が多い	帳票のみならず、Business Intelligence機能等での活用が容易

Copyright© 2011, Oracle. All rights reserved.

アジェンダ

- 集計処理今昔
- Oracle Databaseの集計処理
  - SQL内部にロジック(IF .. THEN .. ELSE)を組み込む
  - GROUP BY拡張
  - ピボット操作
  - 分析関数
  - MODEL句
- 効果的な集計処理
- まとめ

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理

主要な機能

- GROUP BY句
- GROUP BY拡張：SQL99
  - ROLLUP, CUBE　Oracle 8i ～
  - GROUPING SETS　Oracle 9i ～
- ピボット操作　Oracle 11g ～
- 分析関数（ウィンドウ・ファンクション）：SQL2003
  - 集計ウィンドウ・ファンクション　Oracle 8i ～
  - FIRST/LAST関数、ヒストグラム関数、等　Oracle 9i ～
- MODEL句　Oracle 10g ～

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理

SQL内部にロジック（IF .. THEN .. ELSE）を組み込む（その1）

- CASE式（Oracle 8i ～ ※1）

単純CASE式

```
SELECT cust_last_name,
       CASE credit_limit
         WHEN  100 THEN 'Low'
         WHEN 5000 THEN 'High'
         ELSE 'Medium'
       END AS credit
  FROM customers
 ORDER BY cust_last_name, credit;
```

検索CASE式

```
SELECT AVG(
         CASE
           WHEN e.salary > 2000
             THEN e.salary
           ELSE 2000
         END
       ) "Average Salary"
  FROM employees e;
```

- DECODE ファンクション

```
SELECT product_id,
       DECODE(warehouse_id, 1, 'Southlake',
                            2, 'San Francisco',
                            3, 'New Jersey',
                            4, 'Seattle',
                               'Non domestic') "Location"
  FROM inventories
 ORDER BY product_id, "Location";
```

※1 PL/SQL構文として
CASEを利用できるように
なるのはOracle 9i以降

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理

SQL内部にロジック(IF .. THEN .. ELSE)を組み込む (その2)

- 集計関数とDECODEを利用した例:

```
SELECT TO_CHAR("データ作成日",'YYYYMM') "年月",
       "取引区分",
       COUNT(*) "件数",
       SUM(DECODE("予算措置区分",'1',"処分損益額",
                                 '2',"処分損益額" * -1,
                                     "処分損益額")) "金額"
  FROM "支払いD"
 WHERE TO_CHAR("データ作成日",'YYYYMMDD')  <= '20110531'
   AND "みなし区分" != '9'
 GROUP BY TO_CHAR("データ作成日",'YYYYMM'),"取引区分"
HAVING SUM(DECODE("予算措置区分",'1',"処分損益額",'2',"処分損益額"*-1,"処分損益額")) != 0
```

- 特定の区分に従い、集計する数字(金額等)を加工することで、単一のSQLで求めるべき数字を取得する

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理

これまでのサブ・トータル（小計）の取得

- カテゴリ単位で集計したい
  - 単一のSQLで記述できるが、複数のSQLをUNIONする必要あり

```
SELECT job, deptno, SUM(sal)
  FROM emp GROUP BY job,deptno
UNION ALL
SELECT job, NULL deptno, SUM(sal)
  FROM emp GROUP BY job
UNION ALL
SELECT NULL job, deptno, SUM(sal)
  FROM emp GROUP BY deptno
UNION ALL
SELECT NULL job, NULL deptno, SUM(sal)
  FROM emp
 ORDER BY job,deptno
```

①

何度もUNIONするのは非効率的！

これまで通り、通常のGROUP BYの結果を元にサブ・トータルを算出する？

JOB	DEPTNO	SUM(SAL)
ANALYST	20	6000
ANALYST		6000
CLERK	10	1300
CLERK	20	1900
CLERK	30	950
CLERK		4150
MANAGER	10	2450
MANAGER	20	2975
MANAGER	30	2850
MANAGER		8275
PRESIDENT	10	5000
PRESIDENT		5000
SALESMAN	30	5600
SALESMAN		5600
	10	8750
	20	10875
	30	9400
全体の合計 →		29025

JOB 単位の小計

DEPTNO 単位の合計

①結果

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理
GROUP BY拡張（ROLLUP その1）

- GROUP BYの結果に加えて、指定した集計軸の並びに沿ったサブ・トータルおよび総合計を得る

構文 ※1

```
GROUP BY ROLLUP(expr1, expr2, expr3)
```

集計の単位：
(expr1, expr2, expr3)
(expr1, expr2)
(expr1)
全体

①

```
SELECT job, deptno, SUM(sal)
  FROM emp
 GROUP BY ROLLUP(job,deptno)
 ORDER BY job,deptno
```

JOB	DEPTNO	SUM(SAL)
ANALYST	20	6000
ANALYST		6000
CLERK	10	1300
CLERK	20	1900
CLERK	30	950
CLERK		4150
MANAGER	10	2450
MANAGER	20	2975
MANAGER	30	2850
MANAGER		8275
PRESIDENT	10	5000
PRESIDENT		5000
SALESMAN	30	5600
SALESMAN		5600
		29025

①結果

※1：GROUP BY句で処理できる列は、拡張機能使用の有無にかかわらず255列以内です

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理

GROUP BY拡張（ROLLUP その2）

- 部分的ROLLUP
  - 総合計は不要、サブ・トータル（小計）を得る

構文 ※1

```
GROUP BY expr1, ROLLUP(expr2, expr3)
```

集計の単位：
(expr1, expr2, expr3)
(expr1, expr2)
(expr1)

①

```
SELECT job, deptno, SUM(sal)
  FROM emp
 GROUP BY job, ROLLUP(deptno)
 ORDER BY job,deptno
```

JOB	DEPTNO	SUM(SAL)
ANALYST	20	6000
ANALYST		6000
CLERK	10	1300
CLERK	20	1900
CLERK	30	950
CLERK		4150
MANAGER	10	2450
MANAGER	20	2975
MANAGER	30	2850
MANAGER		8275
PRESIDENT	10	5000
PRESIDENT		5000
SALESMAN	30	5600
SALESMAN		5600

①結果

※1：GROUP BY句で処理できる列は、拡張機能使用の有無にかかわらず255列以内です

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理
GROUP BY拡張（CUBE その1）

- GROUP BYの結果に加えて、
  多次元クロス・タブの結果を得る

構文 ※1

```
GROUP BY CUBE(expr1, expr2, expr3)
```

集計の単位:

(expr1, expr2, expr3)

(expr1, expr2), (expr1,expr3), (expr2,expr3)

(expr1), (expr2), (expr3)

全体

```
SELECT job, deptno, SUM(sal)
  FROM emp
 GROUP BY CUBE(job,deptno)
 ORDER BY job,deptno
```

①

JOB	DEPTNO	SUM(SAL)
ANALYST	20	6000
ANALYST		6000
CLERK	10	1300
CLERK	20	1900
CLERK	30	950
CLERK		4150
MANAGER	10	2450
MANAGER	20	2975
MANAGER	30	2850
MANAGER		8275
PRESIDENT	10	5000
PRESIDENT		5000
SALESMAN	30	5600
SALESMAN		5600
	10	8750
	20	10875
	30	9400
		29025

①結果

※1: GROUP BY句で処理できる列は、拡張機能使用の有無にかかわらず255列以内です

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理
GROUP BY拡張（CUBE その2）

- 部分的CUBE
  - 総合計は不要、クロス・タブの結果を得る

構文 ※1

```
GROUP BY expr1, CUBE(expr2, expr3)
```

集計の単位：expr1 と CUBE内に指定したカラムの組合せ
(expr1, expr2, expr3)
(expr1, expr2), (expr1,expr3)
(expr1)

```
SELECT job, deptno, SUM(sal)
  FROM emp
 GROUP BY job, CUBE(deptno)
 ORDER BY job,deptno
```

①

JOB	DEPTNO	SUM(SAL)
ANALYST	20	6000
ANALYST		6000
CLERK	10	1300
CLERK	20	1900
CLERK	30	950
CLERK		4150
MANAGER	10	2450
MANAGER	20	2975
MANAGER	30	2850
MANAGER		8275
PRESIDENT	10	5000
PRESIDENT		5000
SALESMAN	30	5600
SALESMAN		5600

①結果

※1：GROUP BY句で処理できる列は、拡張機能使用の有無にかかわらず255列以内です

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理
GROUP BY拡張（GROUPING SETS）

- ROLLUPやCUBEは独自の意味を持っていますが、GROUPING SETSは任意の集約単位による集約が可能

構文

```
GROUP BY GROUPING SETS(…)
```

()は総合計を意味する

```
GROUP BY ROLLUP(a,b,c)
```

＝

```
GROUP BY GROUPING SETS
  ((a,b,c), (a,b), ())
```

```
GROUP BY CUBE(a,b,c)
```

＝

```
GROUP BY GROUPING SETS
  ((a,b,c), (a,b), (a,c),(b,c)
   (a), (b), (c), ())
```

詳細はマニュアル「Oracle Databaseデータ・ウェアハウス・ガイド 11gリリース2（11.2）」を参照してください
http://download.oracle.com/docs/cd/E16338_01/server.112/b56309/aggreg.htm#i1007462

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理

ピボット操作（PIVOT）

- クロス・タブ・レポートを作成
  - JOB毎の部門への配属人数内訳

これまでのピボット操作

```
SELECT job,
       SUM(CASE deptno WHEN 10 THEN 1 ELSE 0 END)) AS d10,
       SUM(CASE deptno WHEN 20 THEN 1 ELSE 0 END)) AS d20,
       SUM(CASE deptno WHEN 10 THEN 1 ELSE 0 END)) AS d30
  FROM emp
 GROUP BY job
 ORDER BY job
```

```
SELECT job, d10, d20, d30
  FROM (SELECT job, deptno, empno FROM emp)
 PIVOT (
   COUNT(empno) FOR deptno
      IN (10 AS d10, 20 AS d20, 30 AS d30)
 )
 ORDER BY job;
```

集計関数

PIVOTを使った操作

ピボット操作の結果

JOB	D10	D20	D30
ANALYST	0	2	0
CLERK	1	2	1
MANAGER	1	1	1
PRESIDENT	1	0	0
SALESMAN	0	0	4

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理

ピボット操作（UNPIVOT）

- UNPIVOTを使うことでデータを
  列から行へ回転させることができます

PIVOTを使った操作

```
SELECT job, d10, d20, d30
  FROM (SELECT job, deptno, empno FROM emp)
 PIVOT (
   COUNT(empno) FOR deptno
     IN (10 AS d10, 20 AS d20, 30 AS d30)
 )ORDER BY job;
```

UNPIVOTを使った操作

```
SELECT job, deptno, cnt
  FROM (
        SELECT job, d10, d20, d30
          FROM (SELECT job, deptno, empno FROM emp)
         PIVOT ( COUNT(empno) FOR deptno
            IN (10 AS d10, 20 AS d20, 30 AS d30)
         )
        )
UNPIVOT INCLUDE NULLS (
  cnt FOR deptno IN (d10, d20, d30)
)ORDER BY job, deptno
```

同じSELECT文

PIVOT結果

JOB	D10	D20	D30
ANALYST	0	2	0
CLERK	1	2	1
MANAGER	1	1	1
PRESIDENT	1	0	0
SALESMAN	0	0	4

UNPIVOT結果

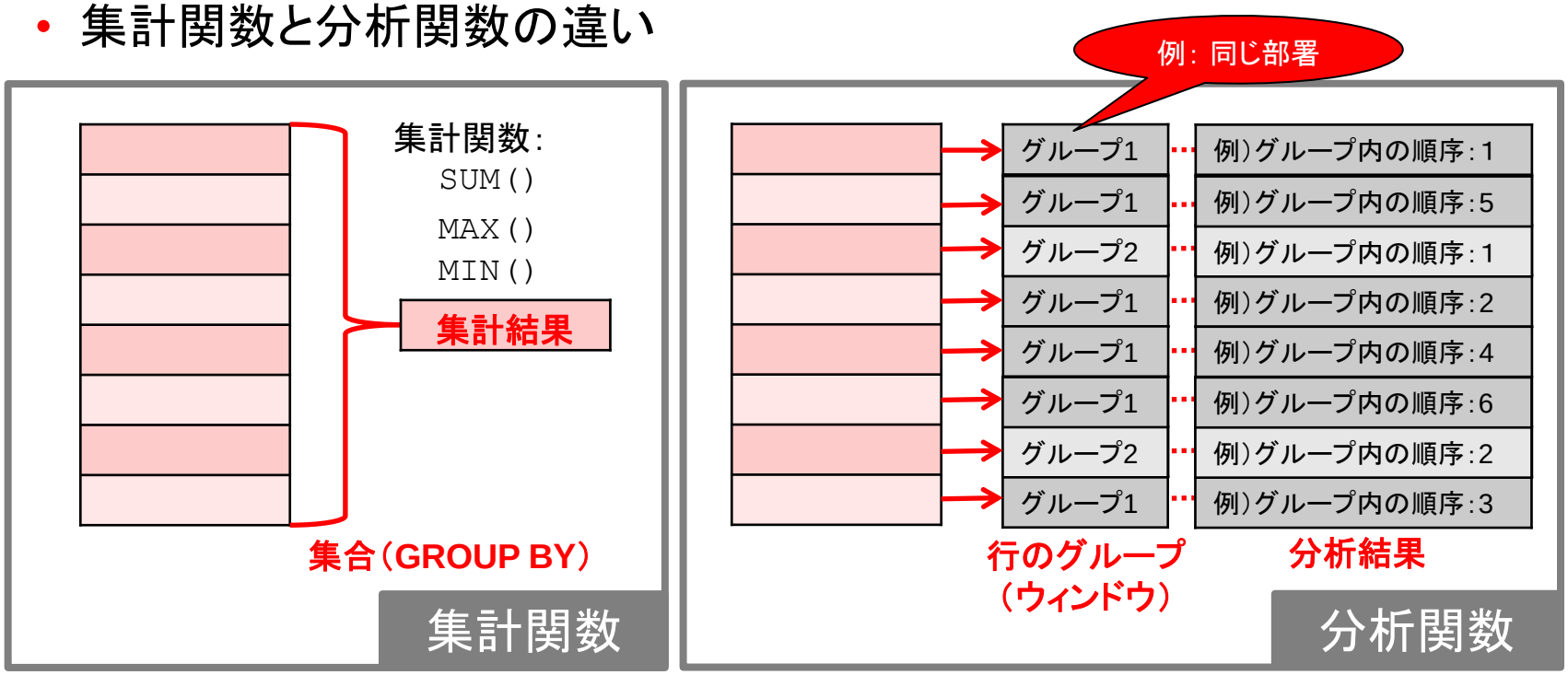
JOB	DEP	CNT
ANALYST	D10	0
ANALYST	D20	2
ANALYST	D30	0
CLERK	D10	1
CLERK	D20	2
CLERK	D30	1
MANAGER	D10	1
MANAGER	D20	1
MANAGER	D30	1
PRESIDENT	D10	1
PRESIDENT	D20	0
PRESIDENT	D30	0
SALESMAN	D10	0
SALESMAN	D20	0
SALESMAN	D30	4

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理

分析関数（ウィンドウ関数）

- 行のグループ（=ウィンドウ）に基づいて集計値を計算する関数（=ファンクション）
  - 集計関数と分析関数の違い



Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理

分析関数 :結果セット・パーティション

- 行グループ内でソートした結果に対して分析関数を適用できます

RANK() OVER を利用

```
SELECT ename,
       deptno,
       sal,
       RANK() OVER (PARTITION BY deptno
                    ORDER BY sal DESC) AS rnk
  FROM emp
ORDER BY deptno, sal desc;
```

ENAME	DEPTNO	SAL	RNK
KING	10	5000	1
CLARK	10	2450	2
MILLER	10	1300	3
FORD	20	3000	1
SCOTT	20	3000	1
JONES	20	2975	3
ADAMS	20	1100	4
SMITH	20	800	5
BLAKE	30	2850	1
ALLEN	30	1600	2
TURNER	30	1500	3
WARD	30	1250	4
MARTIN	30	1250	4
JAMES	30	950	6

ソート

PARTITION 設定しない場合は、全体の行に対してRANK()をおこなう

行のグループ（ウィンドウ）

結果セット・パーティションの単位（deptno）

※ 分析関数で利用するパーティションは、表パーティション機能とは無関係です

Copyright© 2011, Oracle. All rights reserved.

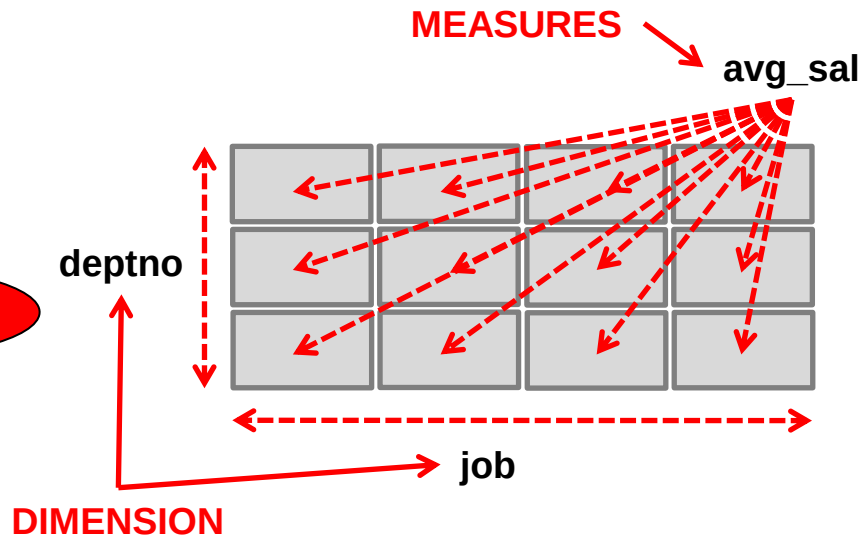
Oracle Databaseの集計処理
MODEL句

- 問合せ結果から多次元配列を作成し、この配列に式（ルール）を適用して新しい値を計算する機能
  - SQLには組み込みにくい計算（時系列分析等）を組み込む

構文

```
SELECT 文 …
MODEL
DIMENSION BY (deptno, job)
MEASURES(avg_sal)
RULES(
   …,      -- RULE1
   …,      -- RULE2
   …       -- RULE3
)
ORDER BY …
```

RULEは計算式



多次元配列のイメージ

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの集計処理

MODEL句の例

Q) 部門(deptno)、職種(job)毎の平均給与、および
仮の部門(deptno=40)の職種(job='MANAGER')として
すべての部門の職種(job='MANAGER')の平均金額を表示

```
SELECT * FROM (
SELECT deptno,job,AVG(sal) avg_sal
  FROM emp GROUP BY deptno,job )
MODEL
DIMENSION BY (deptno, job)
MEASURES(avg_sal)
RULES(
  avg_sal[40, 'MANAGER']
   = ROUND(AVG(avg_sal)[ANY, 'MANAGER'])
)
 ORDER BY deptno,job;
```

MODEL句によるSQL

	ANALYST	CLERK	MANAGER	PRESIDENT	SALESMAN
10		1300	2450	5000	
20	3000	950	2975		
30		950	2850		1400
40			2758		

多次元配列のイメージ

RULEによる導出

DEPTNO	JOB	AVG_SAL
10	CLERK	1300
10	MANAGER	2450
10	PRESIDENT	5000
20	ANALYST	3000
20	CLERK	950
20	MANAGER	2975
30	CLERK	950
30	MANAGER	2850
30	SALESMAN	1400
40	MANAGER	2758

RULE次第で上書き可能

実行結果

Copyright© 2011, Oracle. All rights reserved.

アジェンダ

- 集計処理今昔
- Oracle Databaseの集計処理
- 効果的な集計処理
  - 集計処理をおこなうSQLの比較
  - 分析処理をおこなうSQLの比較
- まとめ

Copyright© 2011, Oracle. All rights reserved.

集計処理をおこなうSQLの比較

集計処理を4種類のSQLで記述してみる

CUBEが
適切なケース

Q) 社員の所属している職種(job)毎、部門(deptno)毎の賃金(sal)合計、
職種毎の賃金合計、部門毎の賃金合計、全社員の賃金合計を表示する

```
SELECT job, deptno, SUM(sal)
  FROM emp GROUP BY job,deptno
UNION ALL
SELECT job, NULL deptno, SUM(sal)
  FROM emp GROUP BY job
UNION ALL
SELECT NULL job, deptno, SUM(sal)
  FROM emp GROUP BY deptno
UNION ALL
SELECT NULL job, NULL deptno, SUM(sal)
  FROM emp
 ORDER BY job,deptno
```

UNION ALL

```
SELECT job, deptno, SUM(sal)
  FROM emp
 GROUP BY CUBE(job,deptno)
 ORDER BY job,deptno;
```

CUBE

```
SELECT job, deptno, SUM(sal)
  FROM emp
 GROUP BY ROLLUP(job),
          ROLLUP(deptno)
 ORDER BY job,deptno;
```

ROLLUP

```
SELECT job,deptno, SUM(sal)
  FROM emp
 GROUP BY GROUPING SETS (
  (job, deptno),
  (job),
  (deptno),
  ()
 )
 ORDER BY job,deptno;
```

GROUPING SETS

結果

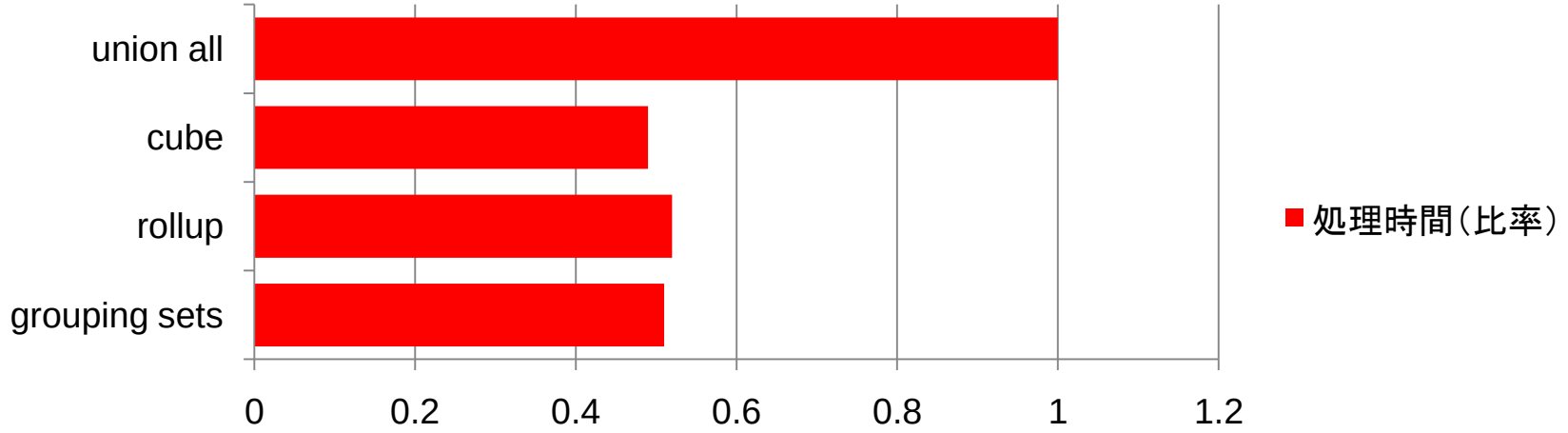
JOB	DEPTNO	SUM(SAL)
ANALYST	20	6000
ANALYST		6000
CLERK	10	1300
CLERK	20	1900
CLERK	30	950
CLERK		4150
MANAGER	10	2450
MANAGER	20	2975
MANAGER	30	2850
MANAGER		8275
PRESIDENT	10	5000
PRESIDENT		5000
SALESMAN	30	5600
SALESMAN		5600
	10	8750
	20	10875
	30	9400
		29025

Copyright© 2011, Oracle. All rights reserved.

集計処理をおこなうSQLの比較

4種類のSQLの処理時間比較

- 件数：10,000,000
- deptnoのバリエーション：80



- テーブルにアクセスする回数が多い（UNION ALL）と性能は悪い
- GROUP BY拡張も若干のばらつきがみられる

Copyright© 2011, Oracle. All rights reserved.

集計処理をおこなうSQLの比較

4種類のSQLの実行計画（抜粋）

UNION ALL

OPERATION	OBJECT_NAME	OPTIONS
SELECT STATEMENT		
SORT		ORDER BY
UNION-ALL		
HASH		GROUP BY
TABLE ACCESS	EMP	FULL
HASH		GROUP BY
TABLE ACCESS	EMP	FULL
HASH		GROUP BY
TABLE ACCESS	EMP	FULL
SORT		AGGREGATE
TABLE ACCESS	EMP	FULL

ROLLUP

OPERATION	OBJECT_NAME	OPTIONS
SELECT STATEMENT		
SORT		ORDER BY
SORT		GROUP BY ROLLUP
SORT		GROUP BY ROLLUP
TABLE ACCESS	EMP	FULL

CUBE

OPERATION	OBJECT_NAME	OPTIONS
SELECT STATEMENT		
SORT		GROUP BY
GENERATE		CUBE
SORT		GROUP BY
TABLE ACCESS	EMP	FULL

GROUPING SETS

OPERATION	OBJECT_NAME	OPTIONS
SELECT STATEMENT		
TEMP TABLE TRANSFORMATION		
MULTI-TABLE INSERT		
SORT		GROUP BY ROLLUP
TABLE ACCESS	EMP	FULL
DIRECT LOAD INTO	SYS_TEMP_0FD9D6616_2288404	
DIRECT LOAD INTO	SYS_TEMP_0FD9D6617_2288404	
LOAD AS SELECT	SYS_TEMP_0FD9D6617_2288404	
SORT		GROUP BY ROLLUP
TABLE ACCESS	SYS_TEMP_0FD9D6616_2288404	FULL
SORT		ORDER BY
VIEW		
VIEW		
UNION-ALL		
TABLE ACCESS	SYS_TEMP_0FD9D6616_2288404	FULL
TABLE ACCESS	SYS_TEMP_0FD9D6617_2288404	FULL

- GROUP BY拡張はすべて同じ実行計画ではなく、それぞれ最適化されている
- 今回、わずかであるが性能が良かったCUBEは指定カラムのすべての組合せでサブ・トータルが導出されるため、実際の利用局面はほとんど無い
- 可能な限り、ROLLUPおよびGROUPING SETSを利用する

Copyright© 2011, Oracle. All rights reserved.

分析処理をおこなうSQLの比較

知りたい情報を同じ行に並べる

Q) 社員の情報と共に、その社員の属する部門、JOBの人数および社員数を表示する

EMPNO	ENAME	DEPTNO	部門人数	職種人数	社員数
7369	SMITH	20	5	4	14
7499	ALLEN	30	6	4	14
7521	WARD	30	6	4	14
7566	JONES	20	5	3	14
7654	MARTIN	30	6	4	14
7698	BLAKE	30	6	3	14
7782	CLARK	10	3	3	14
7788	SCOTT	20	5	2	14
7839	KING	10	3	1	14
7844	TURNER	30	6	4	14
7876	ADAMS	20	5	4	14
7900	JAMES	30	6	4	14
7902	FORD	20	5	2	14
7934	MILLER	10	3	4	14

14行が選択されました。

実行結果

- 3種類のSQLで表現可能
  - 性能が良いのはどれか？
  - 注意点は無いか？

1) SELECTリストの中でスカラ・サブクエリを利用

2) インライン・ビューとの結合

3) 分析関数を利用

Copyright© 2011, Oracle. All rights reserved.

分析処理をおこなうSQLの比較

3種類のSQL文

```
SELECT e.empno, e.ename, e.deptno,
       (SELECT COUNT(*) FROM emp ed WHERE e.deptno = ed.deptno) "部門人数",
       (SELECT COUNT(*) FROM emp ej WHERE e.job = ej.job) "職種人数",
       (SELECT COUNT(*) FROM emp ) "社員数"
  FROM emp e ORDER BY e.empno;
```

SELECTリストの中でスカラ・サブクエリを利用

```
SELECT e.empno, e.ename, e.deptno,
       ed.cnt   "部門人数",
       ej.cnt   "職種人数",
       et.cnt   "社員数"
  FROM emp e,
       (SELECT deptno, COUNT(*) cnt FROM emp GROUP BY deptno) ed,
       (SELECT job,    COUNT(*) cnt FROM emp GROUP BY job) ej,
       (SELECT         COUNT(*) cnt FROM emp) et
 WHERE e.deptno = ed.deptno
   AND e.job    = ej.job
 ORDER BY e.empno;
```

インライン・ビューとの結合

```
SELECT e.empno, e.ename, e.deptno,
       COUNT(*) OVER (PARTITION BY deptno) "部門人数",
       COUNT(*) OVER (PARTITION BY job)    "職種人数",
       COUNT(*) OVER ()                    "社員数"
  FROM emp e ORDER BY e.empno;
```

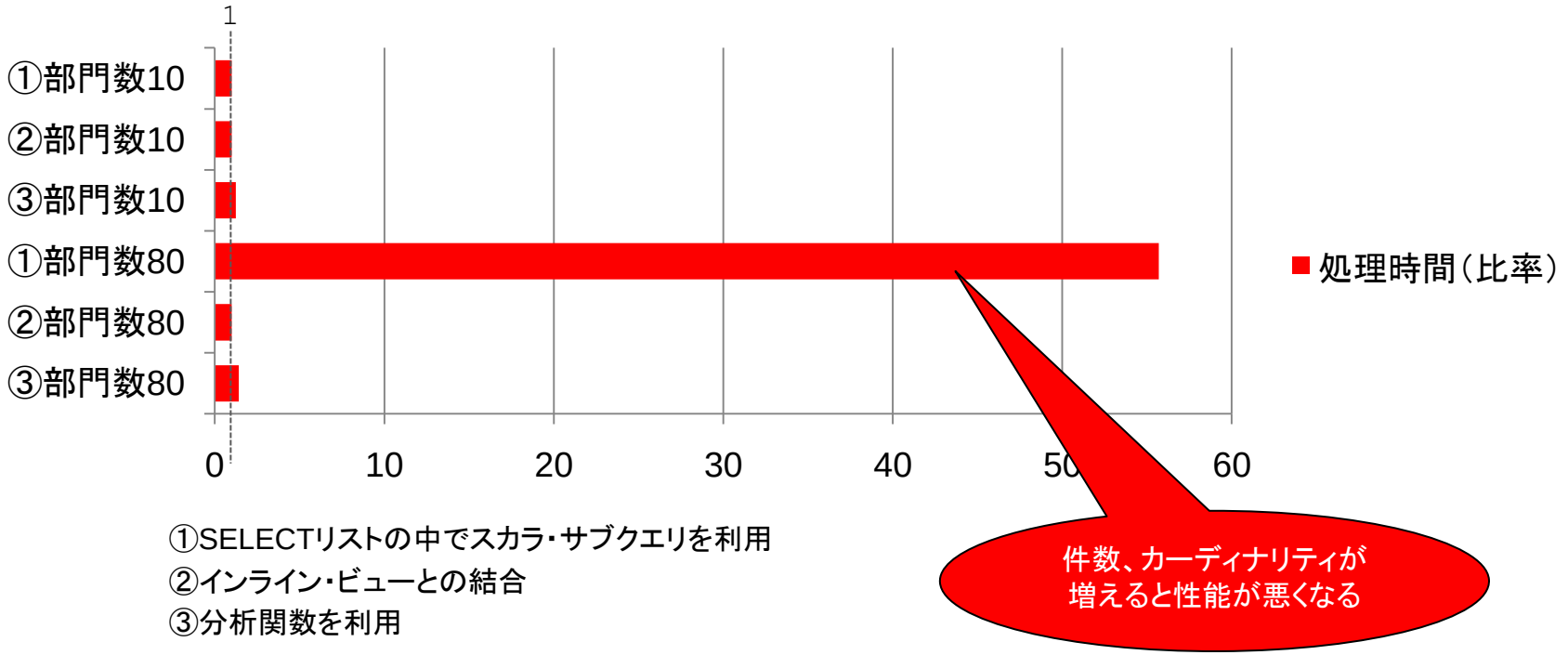
分析関数を利用

Copyright© 2011, Oracle. All rights reserved.

分析処理をおこなうSQLの比較

3種類のSQL文の性能比較

- 件数：1,000,000
- deptnoのバリエーション：10、80



Copyright© 2011, Oracle. All rights reserved.

分析処理をおこなうSQLの比較

3種類のSQL文の実行計画（抜粋）

SELECTリストの中でスカラ・サブクエリを利用

OPERATION	OBJECT_NAME	OPTIONS
SELECT STATEMENT		
SORT		AGGREGATE
TABLE ACCESS	EMP	FULL
フィルタ述語		
ED.DEPTNO=:B1		
SORT		AGGREGATE
TABLE ACCESS	EMP	FULL
フィルタ述語		
EJ.JOB=:B1		
SORT		AGGREGATE
INDEX	PK_EMP	FULL SCAN
SORT		ORDER BY
TABLE ACCESS	EMP	FULL

分析関数を利用

OPERATION	OBJECT_NAME	OPTIONS
SELECT STATEMENT		
WINDOW		SORT
WINDOW		SORT
WINDOW		SORT
TABLE ACCESS	EMP	FULL

インライン・ビューとの結合

OPERATION	OBJECT_NAME	OPTIONS
SELECT STATEMENT		
SORT		ORDER BY
HASH JOIN		
アクセス述語		
E.JOB=EJ.JOB		
HASH JOIN		
アクセス述語		
E.DEPTNO=ED.DEPTNO		
NESTED LOOPS		
VIEW		
SORT		AGGREGATE
INDEX	PK_EMP	FULL SCAN
VIEW		
SORT		GROUP BY
TABLE ACCESS	EMP	FULL
TABLE ACCESS	EMP	FULL
VIEW		
HASH		GROUP BY
TABLE ACCESS	EMP	FULL

- SELECTリストの中でスカラ・サブクエリはなるべく利用しない（特にサブクエリで参照する表のデータ量が多い場合）
- 分析関数をつかうと、**理解しやすいSQL**が記述できるので利用を推奨

Copyright© 2011, Oracle. All rights reserved.

分析処理をおこなうSQLの比較

分析関数を利用した場合のWHERE句は注意（その1）

Q）社員の情報と共に、その社員の属する部門、JOBの人数および社員数を表示する

なお、deptno=10のものだけを表示させる

```
SELECT e.empno, e.ename, e.deptno,
       ed.cnt   "部門人数",
       ej.cnt   "職種人数",
       et.cnt   "社員数"
  FROM emp e,
       (SELECT deptno, COUNT(*) cnt
          FROM emp GROUP BY deptno) ed,
       (SELECT job,    COUNT(*) cnt
          FROM emp GROUP BY job) ej,
       (SELECT         COUNT(*) cnt
          FROM emp) et
 WHERE e.deptno = ed.deptno
   AND e.job    = ej.job
   AND e.deptno = 10
 ORDER BY e.empno;
```

インライン・ビューとの結合

```
SELECT e.empno, e.ename, e.deptno,
   COUNT(*) OVER (PARTITION BY deptno) "部門人数",
   COUNT(*) OVER (PARTITION BY job)    "職種人数",
   COUNT(*) OVER ()                    "社員数"
  FROM emp e
WHERE e.deptno = 10
 ORDER BY e.empno;
```

分析関数を利用

絞り込みの条件追加

Copyright© 2011, Oracle. All rights reserved.

分析処理をおこなうSQLの比較

分析関数を利用した場合のWHERE句は注意（その2）

- 分析関数の場合、結果が異なるものとなってしまうので注意

```
EMPNO ENAME    DEPTNO   部門人数   職種人数    社員数
----- ------- ------   ------   -------    -----
 7782 CLARK        10        3          3        14
 7839 KING         10        3          1        14
 7934 MILLER       10        3          4        14

3行が選択されました。
```

インライン・ビューとの結合

```
EMPNO ENAME    DEPTNO   部門人数   職種人数    社員数
----- ------- ------   ------   -------    -----
 7782 CLARK        10        3          1         3
 7839 KING         10        3          1         3
 7934 MILLER       10        3          1         3

3行が選択されました。
```

分析関数を利用

分析関数はWHERE句の後に適用される為

```
SELECT e.empno, e.ename, e.deptno,
       COUNT(*) OVER (PARTITION BY deptno) "部門人数",
       COUNT(*) OVER (PARTITION BY job)    "職種人数",
       COUNT(*) OVER ()                    "社員数"
  FROM emp e
 WHERE e.deptno = 10
 ORDER BY e.empno;
```

分析関数を利用

```
SELECT * FROM (
SELECT e.empno, e.ename, e.deptno,
       COUNT(*) OVER (PARTITION BY deptno) "部門人数",
       COUNT(*) OVER (PARTITION BY job)    "職種人数",
       COUNT(*) OVER ()                    "社員数"
  FROM emp e
)
WHERE deptno = 10
ORDER BY empno
;
```

全体をインライン・ビュー化

分析関数を利用

Copyright© 2011, Oracle. All rights reserved.

アジェンダ

- 集計処理今昔
- Oracle Databaseの集計処理
- 効果的な集計処理
- まとめ

Copyright© 2011, Oracle. All rights reserved.

まとめ

- SQLを効果的に利用することで、効率的に集計処理をおこなうことができます
- Oracle Databaseでは効率的に集計処理をおこなう仕組みが以前のバージョンより提供されています
  - GROUP BY拡張（ROLLUP、CUBE、GROUPING SETS）
  - ピボット操作
  - 分析関数
  - MODEL句
- GROUP BY拡張の例、分析関数の利用例を説明しました

Copyright© 2011, Oracle. All rights reserved.

Appendix

- 参考資料、参考情報
  - マニュアル
    - Oracle Databaseデータ・ウェアハウス・ガイド 11gリリース2（11.2）
      - http://download.oracle.com/docs/cd/E16338_01/server.112/b56309/toc.htm
  - 書籍
    - Anthony Molinaro著「SQLクックブック」（株）オライリージャパン
      - http://www.oreilly.co.jp/books/9784873113159/
      - （Oracle Database のバージョンは10g）
  - MODEL句
    - Oracle Database 10g の SQL MODEL句
      - http://www.oracle.co.jp/grid/papers/db/SQLModel.pdf

Copyright© 2011, Oracle. All rights reserved.

SQL実行例に関する補足事項

- SQL実行例の動作確認はOracle Database 11g R2（11.2.0.2）Enterprise Editionにておこなっています
- SQL実行例の大部分はサンプル・スキーマSCOTTのEMP表を利用しています

	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	7369	SMITH	CLERK	7902	80-12-17	800	(null)	20
2	7499	ALLEN	SALESMAN	7698	81-02-20	1600	300	30
3	7521	WARD	SALESMAN	7698	81-02-22	1250	500	30
4	7566	JONES	MANAGER	7839	81-04-02	2975	(null)	20
5	7654	MARTIN	SALESMAN	7698	81-09-28	1250	1400	30
6	7698	BLAKE	MANAGER	7839	81-05-01	2850	(null)	30
7	7782	CLARK	MANAGER	7839	81-06-09	2450	(null)	10
8	7788	SCOTT	ANALYST	7566	87-04-19	3000	(null)	20
9	7839	KING	PRESIDENT	(null)	81-11-17	5000	(null)	10
10	7844	TURNER	SALESMAN	7698	81-09-08	1500	0	30
11	7876	ADAMS	CLERK	7788	87-05-23	1100	(null)	20
12	7900	JAMES	CLERK	7698	81-12-03	950	(null)	30
13	7902	FORD	ANALYST	7566	81-12-03	3000	(null)	20
14	7934	MILLER	CLERK	7782	82-01-23	1300	(null)	10

Copyright© 2011, Oracle. All rights reserved.

OTN×ダイセミ　でスキルアップ!!

・一般的な技術問題解決方法などを知りたい！
・セミナ資料など技術コンテンツがほしい！

Oracle Technology Network(OTN)を御活用下さい。

http://forums.oracle.com/forums/main.jspa?categoryID=484

一般的技術問題解決にはOTN掲示版の
「データベース一般」をご活用ください

※OTN掲示版は、基本的にOracleユーザー有志からの回答となるため100%回答があるとは限りません。
ただ、過去の履歴を見ると、質問の大多数に関してなんらかの回答が書き込まれております。

http://www.oracle.com/technetwork/jp/ondemand/index.html

過去のセミナ資料、動画コンテンツはOTNの
「OTNセミナー　オンデマンド コンテンツ」へ

※ダイセミ事務局にダイセミ資料を請求頂いても、お受けできない可能性がございますので予めご了承ください。
ダイセミ資料はOTNコンテンツ　オン　デマンドか、セミナ実施時間内にダウンロード頂くようお願い致します。

ORACLE

Copyright© 2011, Oracle. All rights reserved.

OTNセミナー　オンデマンド　コンテンツ

ダイセミで実施された技術コンテンツを動画で配信中!!

ダイセミのライブ感はそのままに、お好きな時間で受講頂けます。

twitter

最新情報つぶやき中

oracletechnetjp

- 人気コンテンツは？
- お勧め情報
- 公開予告　など

OTN トップページ　http://www.oracle.com/technetwork/jp/index.html

ページ左「基本リンク」>「OTN セミナー　オンデマンド」

※掲載のコンテンツ内容は予告なく変更になる可能性があります。
期間限定での配信コンテンツも含まれております。お早めにダウンロード頂くことをお勧めいたします。

Copyright© 2011, Oracle. All rights reserved.

Oracle エンジニアのための技術情報サイト

オラクルエンジニア通信

http://blogs.oracle.com/oracle4engineer/

twitter

最新情報つぶやき中

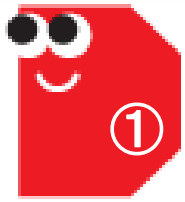
oracletechnetjp

- 技術資料
  - ダイセミの過去資料や製品ホワイトペーパー、スキルアップ資料などを多様な方法で検索できます
  - キーワード検索、レベル別、カテゴリ別、製品・機能別
- コラム
  - オラクル製品に関する技術コラムを毎週お届けします
  - 決してニッチではなく、誰もが明日から使える技術の「あ、そうだったんだ！」をお届けします

オラクルエンジニア通信

Copyright© 2011, Oracle. All rights reserved.

Oracle Databaseの価格ご存知ですか？



問題：

Oracle Databaseの最小構成はいくらでしょうか？

ヒント：

Oracle Standard Edition Oneを

5Named User Plus(指名ユーザ) というのが最小構成です。

問題：

Real Applications Clusters(RAC) Optionはいくらでしょうか？

ヒント：

RACはOracle Database Enterprise EditionのOptionです。

答えはこちら↓ ログイン不要の簡単見積もり

ライセンス見積もりヘルプ

検索

見積もり Start!

Copyright© 2011, Oracle. All rights reserved.

ITプロジェクト全般に渡る無償支援サービス

Oracle Direct Conciergeサービス

■パフォーマンス診断サービス
- Webシステム　ボトルネック診断サービス NEW
- データベースパフォーマンス 診断サービス

■移行支援サービス
- SQL Serverからの移行支援サービス
- DB2からの移行支援サービス
- Sybaseからの移行支援サービス
- MySQLからの移行支援サービス
- Postgre SQLからの移行支援サービス
- Accessからの移行支援サービス
- Oracle Application ServerからWeblogicへ移行支援サービス NEW

■システム構成診断サービス
- Oracle Database構成相談サービス
- サーバー統合支援サービス
- 仮想化アセスメントサービス
- メインフレーム資産活用相談サービス
- BI EEアセスメントサービス
- 簡易業務診断サービス

■バージョンアップ支援サービス
- Oracle Databaseバージョンアップ支援サービス
- Weblogic Serverバージョンアップ支援サービス NEW
- Oracle Developer/2000(Froms/Reports) Webアップグレード相談サービス

オラクル社のエンジニアが 直接ご支援します
お気軽にご活用ください!

オラクル　無償支援　検索

Copyright© 2011, Oracle. All rights reserved.

Oracle Enterprise Cloud Summit

Solid foundation. Elastic cloud.

オラクル・エンタープライズ・クラウド・サミット
クラウド環境のための強固な情報基盤

2011年5月25日（水）10:00－16:50（開場9:30）
ザ・プリンスパークタワー東京

ORACLE
CLOUD COMPUTING

多くのご要望にお応えし、セッション追加決定

～今こそお伝えしたい、事業継続性を確保するためにオラクルが出来ること～

- C-1 事業継続性と最大可用性の視点から考えるクラウド環境構築の秘訣
- C-2 今から始めるBCP/BCM対策、Oracleだからできること

■内　　容	基調講演×2、個別セッション×12
■定　　員	400名
■対　　象	CIO、経営企画・情報システム部門マネージャ、開発者・管理者、情報システムアーキテクト
■入 場 料	無料
■主　　催	日本オラクル株式会社

■基調講演　Roadmap to Cloud

ソフトバンクモバイル株式会社
取締役専務執行役員兼CISO
阿多　親市 様

日本オラクル株式会社
常務執行役員 クラウド＆EA統括本部長
三澤　智光

お申込み　>>>　http://www.oracle.co.jp/oecs2011/

1日5組限定！

製品無償評価サービス

提供シナリオー例

・データベースチューニング　・無停止アップグレード

・アプリケーション性能・負荷検証　・Webシステム障害解析

インストールすることなく、すぐに体験いただけます

- サービスご提供までの流れ
  1. お問合せフォームより「製品評価サービス希望」と必要事項を明記し送信下さい
  2. 弊社より接続方法手順書およびハンズオン手順書を送付致します
  3. 当日は、弊社サーバー環境でインターネット越しに製品を体感頂けます

※サービスご提供には事前予約が必要です

Web問い合わせフォーム

「ダイデモ」をキーワードに検索することで申し込みホームページにアクセスできます

http://www.oracle.com/jp/direct/services/didemo-195748-ja.html

ORACLE

Copyright© 2011, Oracle. All rights reserved.

あなたにいちばん近いオラクル

Oracle Direct

まずはお問合せください

Oracle Direct　検索

システムの検討・構築から運用まで、ITプロジェクト全般の相談窓口としてご支援いたします。

システム構成やライセンス/購入方法などお気軽にお問い合わせ下さい。

Web問い合わせフォーム

専用お問い合わせフォームにてご相談内容を承ります。

https://secure.oracle.co.jp/direct/inquiry-form.php

※こちらから詳細確認のお電話を差し上げる場合がありますので、ご登録されている連絡先が最新のものになっているか、ご確認下さい。

フリーダイヤル

0120－155－096

※月曜~金曜　9:00~12:00、13:00~18:00

（祝日および年末年始除く）

ORACLE

Copyright© 2011, Oracle. All rights reserved.