ORACLE

Oracle GoldenGate Advantages

Comparison of Change Data Capture (CDC) Techniques for the Oracle Database

July, 2021, Version 2.1

Public

Purpose statement

This document provides an overview of how Oracle GoldenGate works, and why it is the best overall choice for the Oracle Database.

When choosing a technology for Change Data Capture (CDC) and data replication, it is crucial to understand in depth how the data events are being captured. Every database has one or more different options to detect data events – each different option comes with trade-offs that should be understood and carefully considered. For example, the Oracle Database over the years has had as many as six different ways to detect data events.

In this document we will discuss technical differences between CDC approaches that include database triggers, single-threaded LogMiner API, XStream API and GoldenGate for use with Oracle Database. We will also briefly discuss the risks associated with unsupported third-party disk log readers.

Disclaimer

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

Table of contents

Purpose statement	**2**
Disclaimer	**2**
Oracle GoldenGate	**4**
How GoldenGate Works	**5**
Anatomy of a GoldenGate Transaction	**6**
Options for Detecting Events in Oracle Database	**6**
Changed Data Using Database Triggers	**7**
Changed Data Using Oracle Database LogMiner	**8**
Challenges with LogMiner	8
Changed Data Using XStreams API	**10**
Important XStream Facts	10
Changed Data Using Oracle GoldenGate	**11**
General Advantages of Oracle GoldenGate:	11
Brief Note Regarding Third-Party Disk Log Readers	**13**
Regarding GoldenGate for Apache Kafka or Object Storage	**14**
Summary – Oracle GoldenGate is the best CDC for Oracle Database	**15**

Oracle GoldenGate has been recognized by customers and analysts as the most functionally complete, highest performing, and most trusted data integration and database replication solution. As a proven heterogeneous solution, it integrates with 100's of combinations non-Oracle databases, data stores and clouds. For the Oracle Database, GoldenGate is the only scalable, complete, and fully supported solution. Other frameworks and tools for capturing data events from Oracle Database are incomplete, non-scalable, and/or unsupported. For example, LogMiner is a single-threaded diagnostic API with limited capabilities. Third-party data integration technology that depends on the LogMiner API cannot come close to matching Oracle GoldenGate for use with the Oracle Database.

The focus and purpose of this paper is to explain the options you may have for capturing data events from the Oracle Database, how GoldenGate fits in, and why it may be your best option.

Oracle GoldenGate

A San Francisco startup founded in the 1990's, GoldenGate's original purpose was to provide business continuity and data high availability for networked ATM/cash machines running from Tandem NonStop databases.

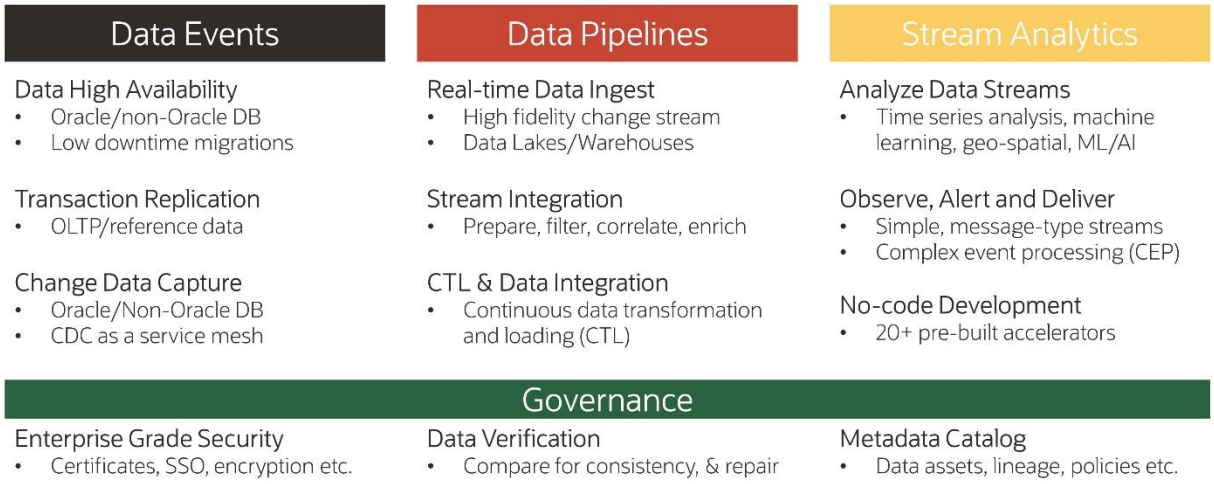

Figure 1: Oracle GoldenGate platform capabilities

Today Oracle GoldenGate has expanded to become a mature platform of integration and analytical capabilities serving Oracle and non-Oracle data sources. The reliability of guaranteed transaction delivery has enabled GoldenGate to be the anchor technology to deliver Oracle MAA 'Platinum Tier' service level. This kind of reliability extends to non-Oracle databases such as NonStop, SQL Server, DB2 iSeries, mainframes and other supported data stores. Many thousands of global banks, retailers, telecoms, healthcare companies etc. run their operational data platforms on the trusted foundation of Oracle GoldenGate.

GoldenGate is a real-time data replication platform that can detect data events and route them across networks at very low latencies. The GoldenGate technology is used for geographic sharding of operational databases, low-downtime data migrations, multi-active (online) data stores, real-time data ingestion to cloud, data lakes, and data warehouses etc. Since 2015 GoldenGate has been increasingly focused on polyglot big data and noSQL data payloads and has been completely refactored for native microservices 'as a service' deployments.

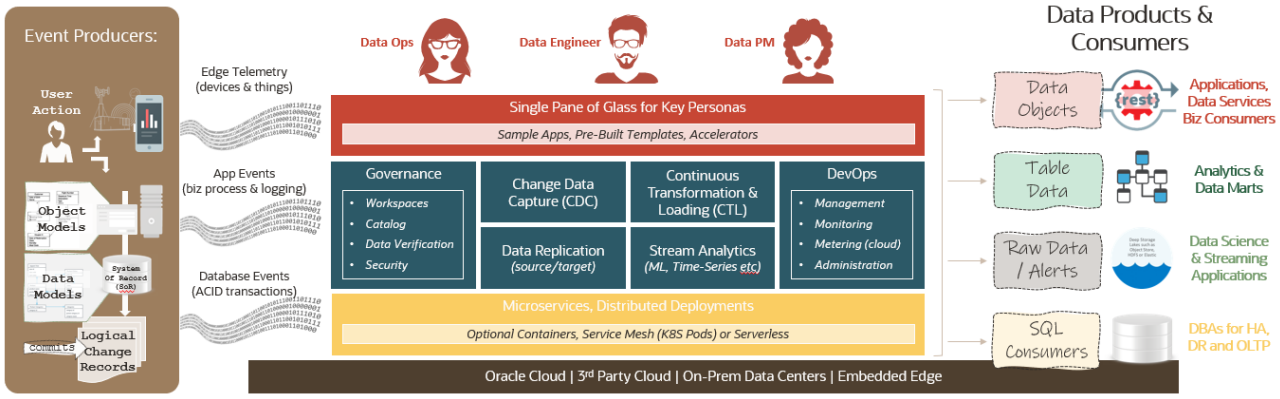

Figure 2: Oracle GoldenGate platform. Logical architecture and key components

In 2018 the GoldenGate platform added Data Pipelines and Stream Analytics with a robust complex event processing (CEP) core engine that scales to billions of events per second while preserving ordered data processing down to the nano-second scale. This event engine can use very powerful semantics for transformations or analytics and runs on an open-source Apache Spark for massively parallel processing (MPP).

With these new capabilities, GoldenGate provides high-value data products directly to data consumers. In the past, GoldenGate would have mainly been used to deliver low-latency raw data to data pipelines or other data products. Today, GoldenGate can push raw data events as well as provide high-value data products.

How GoldenGate Works

At heart, the GoldenGate CDC capabilities are used for detecting and transmitting data events. Data events typically include database DML (data manipulation language) and DDL (data definition language). On the Oracle Database, GoldenGate can also detect and transmit procedure events, which is useful when keeping databases in sync. GoldenGate has been designed from the ground up to be a trusted provider of data events and can be trusted for MAA (maximum availability architecture) use cases, as well as for DW (data warehouse), Data Lake and data streaming (eg; Kafka-based) scenarios that are mission-critical.

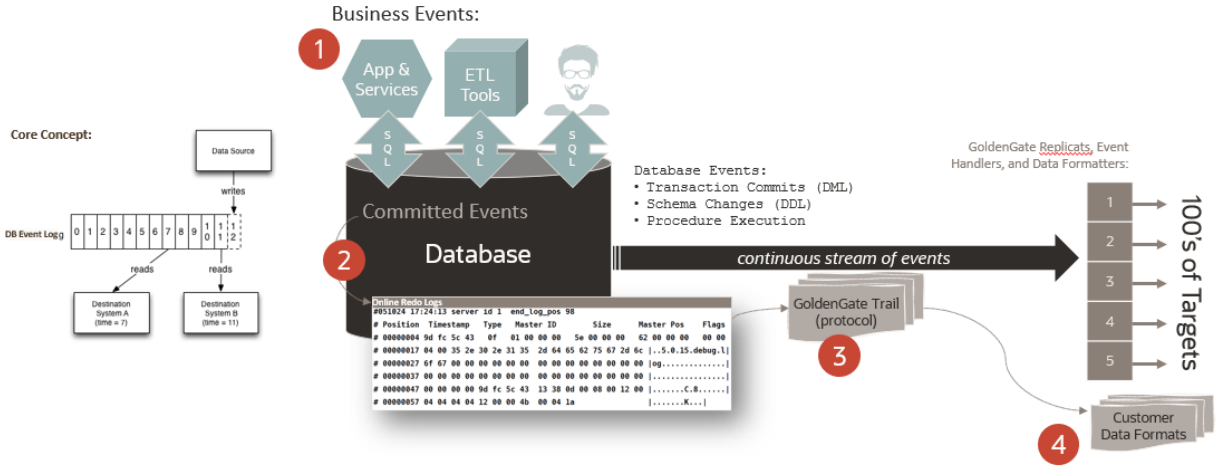

Figure 3: Change detection and propagation in Oracle GoldenGate

Anatomy of a GoldenGate Transaction

As database log events are captured, they are converted into the GoldenGate canonical ledger called Trail. All the data event logs of the 100's of combinations of supported sources/platforms are converted into this single canonical Trail format. It is the Trail which is distributed via the GG Distribution Microservice. Each Trail may have one or more 'paths' and Trail consumers may initiate their own access to a path, or GoldenGate can be configured to push Trail downstream into a Receiver Microservice.

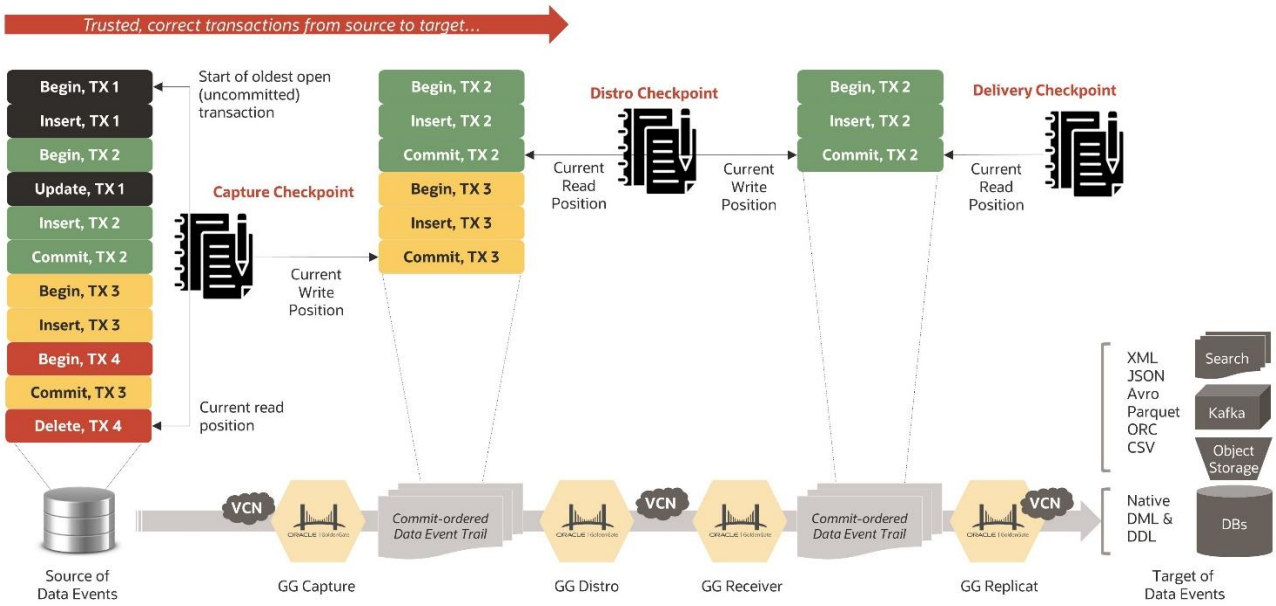

Figure 4: Anatomy of a GoldenGate transaction, preserving ACID properties in a distributed architecture

GoldenGate is aware of all source operations as they hit the database logs, and each different kind of database handles operations differently. As a general rule, GoldenGate emits the data events once they are committed on the source system. In database systems, many operations can be grouped together in a single commit. In fact, in some long running transactions, millions of objects might be modified as part of a single transaction that takes many hours to complete and is interleaved with thousands of smaller uncommitted operations. GoldenGate groups together, isolates and preserves the consistency of these transactions as they move across the networks.

Options for Detecting Events in Oracle Database

Over the years, Oracle Database has had many APIs that can be leveraged for changed data capture (CDC). Today, there are handful of techniques that remain a strategic part of the Oracle ecosystem:

- **Database Triggers** are a stored PL/SQL block, which may be associated with a Table and execute when DML statements such as INSERT, UPDATE, or DELETE occur. Using Triggers to detect events require a detailed knowledge of the schema and application which is prone to employee turnover and poor documentation. This highly invasive strategy can be both incomplete and incur lots of runtime overhead. Nevertheless, many ETL vendors continue to support using CDC with Database Triggers. (documentation: https://docs.oracle.com/en/database/oracle/oracle-database/19/lnpls/CREATE-TRIGGER-statement.html#GUID-AF9E33F1-64D1-4382-A6A4-EC33C36F237B)
- **Oracle LogMiner** is a component of an Oracle Database that enables users to query online and archived redo log files through a SQL interface. LogMiner is a popular option with many ETL and Replication vendors because it is an API of the database that is free to use without any additional licensing. With LogMinor activated, there may be some noticeable degradation in the update performance of Oracle Database.

(documentation: https://docs.oracle.com/en/database/oracle/oracle-database/19/sutil/oracle-logminer-utility.html#GUID-3417B738-374C-4EE3-B15C-3A66E01AE2B5)

- **Oracle XStream** consists of Oracle Database components and application programming interfaces (APIs) that enable client applications to receive data changes from an Oracle database and send data changes to an Oracle database. XStream is licensed with Oracle GoldenGate and is used by ISV application partners and non-Oracle replication tools to support high-speed CDC from the Oracle Database. (documentation: https://docs.oracle.com/en/database/oracle/oracle-database/19/xstrm/introduction-to-xstream.html#GUID-644C0567-E409-4611-9D9E-C5C51FBF2DE0)
- **Oracle GoldenGate** is the best-performing and most feature-rich way to detect changes and replicate transactions from the Oracle Database. A modern microservices architecture makes GoldenGate an ideal solution for modernizing operational IT systems, applications and analytic data platforms. (documentation: https://docs.oracle.com/en/middleware/goldengate/core/21.1/index.html)

This document will also discuss one non-strategic alternative for detecting changes in Oracle Database:

- **Unsupported Disk Log Readers** are third-party proprietary tools that mine the Oracle Database logs directly from disk-based storage. These approaches reverse engineer the Oracle Database data files and APIs. This approach is unsupported by Oracle, poses measurable security risks and may be in violation of Oracle license agreements. Later in this document we will explain the risks involved with this technique.

Oracle GoldenGate is the recommended approach for CDC and replication with Oracle Databases. It supports more than 300 combinations of sources and targets including all versions of Oracle Database at all patch levels. GoldenGate uses its own deeply integrated log capture API for detecting log events and reading log records from Oracle DBs, enabling the least invasive, fastest and most scalable solutions.

Changed Data Using Database Triggers

Using triggers can be an acceptable way to detect DML events for a small number of tables, and for use cases that do not have a lot of transaction volume. However, each trigger places new objects in the database and is essentially running a stored procedure for each event that is triggered – thus, there can be quite a bit of overhead on the application and the database when using triggers extensively. Few high-end tools will use triggers, but nonetheless some mainstream ETL tools still provide developer tools that setup and configure triggers as part of an ETL flow, this is something to beware of as it could negatively affect performance of the database.

This example creates a DML trigger that uses conditional predicates to determine which of its four possible triggering statements fired it:

```
CREATE OR REPLACE TRIGGER t
  BEFORE
    INSERT OR
    UPDATE OF salary, department_id OR
    DELETE
  ON employees
BEGIN
  CASE
    WHEN INSERTING THEN
      DBMS_OUTPUT.PUT_LINE('Inserting');
    WHEN UPDATING('salary') THEN
      DBMS_OUTPUT.PUT_LINE('Updating salary');
```

```
            WHEN UPDATING('department_id') THEN
              DBMS_OUTPUT.PUT_LINE('Updating department ID');
            WHEN DELETING THEN
              DBMS_OUTPUT.PUT_LINE('Deleting');
          END CASE;
        END; /
```

Figure 5: example DML trigger

Changed Data Using Oracle Database LogMiner

Historically Oracle created LogMiner and maintained it for support and diagnostic use cases. Although many third-party vendors use this API for change data capture, that use case was not the original intent of the API.

Oracle redo logs capture all changes made to user data or to the database dictionary so that database recovery operations can be performed and to conform with relational database ACID (Atomicity, Consistency, Integrity, Durability) properties.

LogMiner is an embedded utility within the Oracle database that Oracle support uses to view the contents of redo log files. It is an auditing and diagnosis tool as well as a data analysis tool. As a DBA, LogMiner uses PL/SQL procedures and functions to find changed records in redo log files.

Among the key capabilities of LogMiner are:

- Tracks when a logical corruption to a database, such as errors made at the application level, may have begun. It is important to know exactly when an error was made so that you know when to initiate time-based or change-based recovery. (documentation: https://docs.oracle.com/en/database/oracle/oracle-database/21/sutil/oracle-logminer-utility.html#GUID-CF064432-57A0-4891-ABE5-800DF327615A)
- Perform fine-grained recovery using a set of reversed SQL statements in order to return a table to its initial state
- Database tuning using its trend analysis capabilities. Determine which table received the most updates and inserts in order to have a historical perspective on disk access statistics
- Check transactional consistency because it can track not only the DDL and DML but also the order of the execution and which user executed the statements

LogMiner is a powerful utility for the Oracle Database, but it also has some challenges to be aware of when attempting to stretch the use cases for LogMiner beyond what it was originally designed for.

Challenges with LogMiner

The following section lists some potential pain-points that any customer should be aware of when using LogMiner directly, or any third-party CDC tool based on LogMiner:

- Log Miner was designed as a diagnostic tool for the database redo logs. It will perform sub-optimally when used to mine the redo logs in an ongoing fashion.
- Log Miner is single-threaded and not designed for low overhead while retrieving the redo records.
- Log Miner will stop when it reached the current System Commit Number (SCN). New requests will start from the beginning of the logs similarly to a full-table-scan (FTS) operation. (documentation: https://docs.oracle.com/en/database/oracle/oracle-database/21/sutil/oracle-logminer-utility.html#GUID-319446A8-6FEC-42CE-A6A4-582CA65377CF)
- It may be cumbersome or error prone when dealing with rollbacks or partial rollbacks when using Log miner for CDC. When an application session is aborted (for instance, it exits suddenly), a rollback should occur for any transaction failure. If a transaction is cancelled or fails, then the database must cleanup the uncommitted

work that was done by this transaction so that other transactions can progress. This cleanup involves rolling back the uncommitted work. From LogMiner point of view, the rollback statement is reported by itself as SQL_REDO and not SQL_UNDO. For a SQL which rolls back, no undo SQL is generated, and the rollback flag is set.

- Beginning with Oracle Database 12.2, LogMiner `continuous_mine` option was deprecated and is no longer available in Oracle Database 19.1 and onward. Any third-party tools dependent on continuous mining will no longer work with database 19c and higher. For further details and to review the full list of decommissioned features please review the documentation. (documentation: https://docs.oracle.com/en/database/oracle/oracle-database/19/upgrd/behavior-changes-deprecated-desupport-oracle-database.html#GUID-4B922F38-109D-42DF-B2AD-698E9FD8A8DE)
- Datatypes since DB 12.2 will generate unsupported sql_redo and sql_undo, including: JSON, OSON, BFILE, Nested tables, Objects with nested tables, Tables with identity columns, Temporal validity columns, PKREF columns, Long identifiers, PKOID columns, Nested table attributes and stand-alone nested table columns (datatype compatibility is with DBMS_ROLLING, at Oracle Database 12.2+ versions)

A key fact that is crucial to understand is that each vendor who implements their own LogMiner based client may have their own distinct limitations and liabilities – the use of LogMiner does not guarantee any particular level of uniformity or common expectations for service levels (SLAs) and KPIs.

For example, a third-party vendor who launched CDC support with LogMiner in summer of 2021 had the following limitations:

- Tables larger than 100 GB can't be backfilled.
- Oracle multi-tenant architecture (CDB/PDB) is not supported.
- Oracle Autonomous Database is not supported.
- Events from tables that do not have a primary key won't contain the information required to perform an update on the consumer side.
- Events have a size limitation of 3 MB.
- Index-organized tables (IOTs) are not supported.
- For columns of type BFILE, only the path to the file will be replicated. The contents of the file will not be replicated.
- Columns of data types ANYDATA, BLOB, CLOB, LONG/LONG RAW, NCLOB, UDT, UROWID, XMLTYPE are not supported, and will be replaced with NULL values.
- For Oracle 11g, tables that have columns of data types ANYDATA or UDT are not supported, and the entire table won't be replicated.
- Oracle Label Security (OLS) is not replicated.
- If a schema changes, some events from the new schema may be read while the old schema is still applied.
- Not all changes to the source schema can be detected automatically, in which case data corruption may occur. The following schema changes may cause data corruption or failure to process the events downstream:
    - Dropping columns
    - Adding columns to the middle of a table
    - Changing the data type of a column
    - Reordering columns

    - Dropping tables (relevant if the same table is then recreated with new data added)
    - Truncating tables
- Does not support replicating views.
- Materialized views created while the stream is running aren't backfilled automatically.

Log Miner remains focused on diagnostic use cases, and customers who choose third-party CDC tools that depend on LogMiner should proceed with caution, within the known limitations of the framework. Use cases for online workloads that exceed hundreds of transactions per second may result in latency, memory and scalability issues. For small and low volume databases, LogMiner might be "good enough" for simple workloads. However, in many cases LogMiner will be paired with additional third-party tooling that can incur additional lifecycle challenges or unnecessary load on the source database.

Anyone using third-party CDC tools with LogMiner should carefully measure the impact and load requirements placed on the source database itself, and make sure that the datatype coverage, functional limitations and security concerns are ok for the business.

Changed Data Using XStreams API

XStream is a strategic part of the Oracle Database, providing a low-level API that enables transaction streams to be replicated "in" or "out" of the database. Its purpose is to provide client applications with the ability to insert and extract logical change records (an LCR contains the DMLs and/or DDLs that are sent from Source to Target) into an implicit or explicit stream. XStream APIs are intended to help power-users, technical partners, and third-party tools with high performance CDC, streaming and replication with the Oracle Database directly.

XStream consists of two major features: *XStream Out* and *XStream In*. XStream Out provides Oracle Database components and APIs that enable you to share data changes made to an Oracle database with other systems. XStream Out can retrieve both data manipulation language (DML) and data definition language (DDL) changes from the redo log and send these changes to a client application that uses the API. (documentation: https://docs.oracle.com/en/database/oracle/oracle-database/19/xstrm/introduction-to-xstream.html#GUID-5939CB6C-8BA9-4594-8F96-B0453D246722)

In general, the XStream APIs are a very powerful, modern and up-to-date access layer for high-speed transactions replicating in or out of the Oracle Database. However, there are some important topics to be aware of.

Important XStream Facts

- **Licensing**, the use of XStream APIs require GoldenGate licenses for any database where XStream is used
- **Automation & lifecycle** of tools using XStream cannot be guaranteed. Because XStream is a low-level API, any higher level tooling (eg; a third-party CDC tool) must implement its own software automation and lifecycle management for working with users, connections, dictionaries, transactions, and any supplemental logging or checkpoint semantics. Individual tools from non-Oracle vendors may vary in quality and capabilities.
- **Recovery semantics** for any databases that XStream APIs are used with. Any source or target database may fail for any number of reasons, and a third-party client that is connected and using XStream will need to implement its own logic for gracefully handing restart situations. If this checkpointing and recovery is done incorrectly, the third-party CDC tool may skip some records, write duplicate records, or otherwise produce inconsistent data sets across the replicated data stores. This situation is also important when replicating transactions from Oracle databases into non-relational targets such as Apache Kafka or object storage.
- **Consistency of third-party tools** is not guaranteed by XStream. There is no magical way to guarantee high quality CDC when third-party tools are merely clients to the XStream API. Implementations may vary from

one vendor to the next, and open source frameworks may have altogether different implementations on top of XStream. Be alert to any vendor consistency issues with their use of XStream, even if Oracle Database customers implement a variety of CDC tools which all use XStream, the individual capabilities, quality and results will differ between the tools.

Changed Data Using Oracle GoldenGate

Oracle GoldenGate leverages its own highly optimized, native API to integrate closely with the Oracle Database. The GoldenGate API is deeply integrated within the core of the database to optimize for high speed data events and to take advantage of database new features as they become available. GoldenGate is the only replication framework that is certified for Oracle's Platinum Tier of the maximum availability architecture (MAA). (documentation: https://www.oracle.com/a/tech/docs/maa-goldengate-hub.pdf)

As with the MAA technical paper noted above, GoldenGate is often run in a 'hub' configuration. This is advantageous for single point-projects and for simplifying the administration of GoldenGate components.

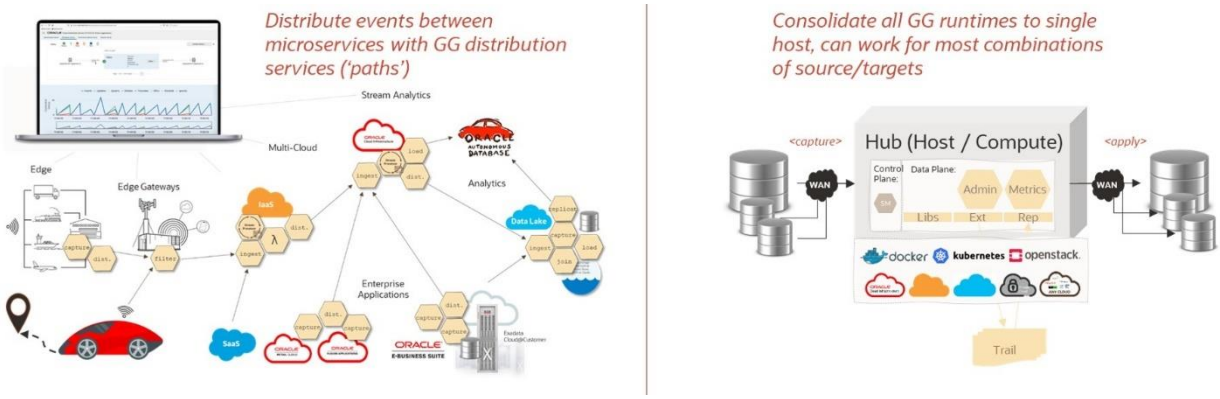

Figure 6: Oracle GoldenGate may be deployed as a Hub or in a microservices based Data Mesh

Oracle GoldenGate also has the ability to run as a mesh for larger enterprises, distributed environments or multi-cloud replication projects. With the Data Mesh approach, customers may distribute events between microservices layers/applications using GoldenGate distribution services.

General Advantages of Oracle GoldenGate:

- Oracle GoldenGate is deeply integrated with the Oracle Database (logging, redo, backup and recovery layer, security, administration, data guard, data vault, autonomous database etc.)
- Oracle GoldenGate is the most stable, performant and reliable CDC / replication solution for Oracle
- High performance: Oracle GoldenGate is capable of sub-second latency for data movement featuring low-impact capture, routing, transformation, and delivery of transactional data
- GoldenGate is the only CDC / replication tool that keeps pace with new and emerging innovations (security, features, datatypes, etc) in the core Oracle Database.
- GoldenGate supports a wide range of use cases from conventional OLTP data replication and high availability (unidirectional, bi-directional, peer-to-peer and more) to data lake ingestion, or multi-cloud ingestion, SaaS application replication, messaging replication. During the last five years GoldenGate has been expanding into the stream processing patterns for use cases like data pipelines, data transformation, time-series analysis, predictive analysis, geospatial, real-time analytics etc.
- Oracle GoldenGate is easier to use when capturing from Oracle Databases, especially when dealing with options like Oracle Real Application Clusters (RAC) and Oracle Automated Storage Management (ASM).

Further, GoldenGate is the only CDC / replication technology certified for Exadata, Exadata Cloud Service and Exadata Cloud at Customer.

- Due to being the only CDC / replication tool certified for Oracle's Platinum Tier of maximum availability architecture, customers can be confident that GoldenGate is tested for the most stringent RPO (recovery point objective) and RTO (recovery time objective) scenarios – our customers trust GoldenGate with their most valuable and sensitive data

From an operational point of view GoldenGate has both a command line and a graphical web user interface that supports database administrators, system administrators, and DevOps teams. Benefits of GoldenGate microservices include:

- Simple to use web interface
- Fast and easy installation or provisioning
- Command line access via AdminClient that can run anywhere
- RESTful APIs for DevOps
- Offers built-in monitoring but also can be easily adapted to roll your own monitoring

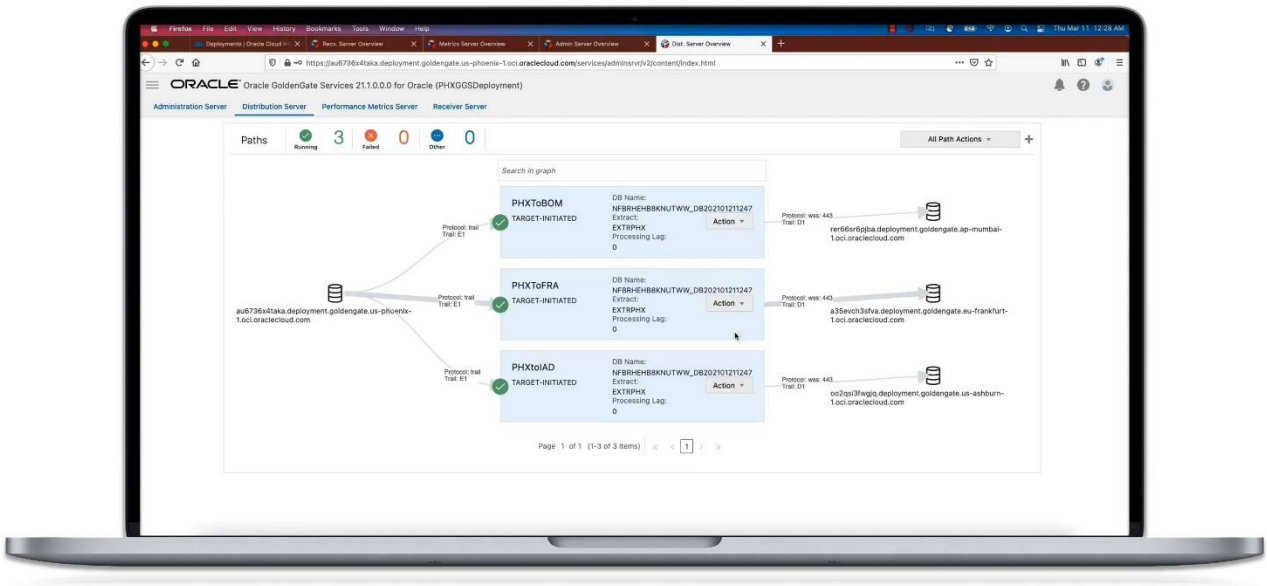

Figure 7: simple and easy user interface for Oracle GoldenGate (both on-premises or cloud-native)

Banking and finance customers have always demanded the most rigorous security framework. GoldenGate security capabilities include:

- Data can be encrypted on disk and over the wire protocol
- Encryption profiles encapsulate the configuration information used to retrieve a master key from Oracle Key Vault
- Native support for DMZ environments and private cloud environments.
- Supports Oracle Transparent Data Encryption (TDE) and Tablespace Encryption (TSE), through all versions of the database (third-party vendors with proprietary disk readers can't support this)
- Rich support for certificates across distributed services, including support for mTLS 1.3

- APIs are protected by Cross Site Request Forgery (CSRF) authentication. The default configuration is to enforce CSRF-token based protection.

Brief Note Regarding Third-Party Disk Log Readers

Over the years some vendors have chosen to create proprietary software that reverse engineers Oracle Database logs directly to find data changes. Vendors who have chosen this strategy have avoided supported APIs and utilities, such as LogMiner and XStream. This puts the data, data security, and ongoing support at risk.

These proprietary log parsers have these shortcomings:

1. **Unsupported and Undocumented** – Oracle does not document the content and structure of the redo logs / archive logs and the use of 3rd party tools that reverse engineer these logs is unsupported and in violation of the Oracle Master Agreement (pt.8) and the Oracle Software License Agreement (pt.9)
2. **Significant Functional Limitations** – since these readers are reverse engineering disk logs, they only see a fraction of the semantics (the meaning) of what the database engine is performing. This leads to many incompatibilities with Exadata, data compression (EHCC), table encryption, access to security keys, datatype incompatibilities etc. Vendors with binary log readers will force customers to disable this functionality in the database in order to process the data.
3. **Governance and Security Risk for End Users** – administrative tools that require direct-host access to database logs and have the power to use user supplied encryption keys pose a significant compliance risk for a number of policies such as SOC2, Basel, BCBS239 and others.
4. **Lack of Lifecycle Integration** – for mission-critical systems, there is no supported or certified Oracle MAA configuration that provides for automation of failover and restart with 3rd party binary log parsers.
5. **Risk for Unplanned Outage** – since this is an unsupported integration, Oracle may change the log formats, encryption, key vaults, etc at any time, causing unplanned outages and complex rollbacks when Database patches are applied.
6. **Problems with Cloud Databases** – Oracle managed cloud databases (e.g. Oracle Database Cloud Service, Oracle Autonomous Database, Oracle Exadata Cloud Service, and Oracle Exadata Cloud at Customer) may not be supported. In particular, it is not supported to install 3rd party binary log parsers from Oracle-managed cloud hosts. Further, Exadata cloud systems require encrypted tables, which are not readable by binary log parsers in Database 19.1 and higher.
7. **Mediocre Performance and Scale** – compared to GoldenGate's native access to Oracle database engine, any direct disk-based access to the redo logs causes unnecessary I/O issues and limitations on parallelism and compute utilization. Real-world workloads may run 2x-5x faster and handle 3x-10x more data volume when using native integrated GoldenGate extracts. These problems are further exacerbated when ASM storage is remotely accessed.
8. **Ownership Rights and Restrictions** – the Oracle Master Agreement expressly forbids causing or permitting others to reverse engineer data structures of the software, see section 3.4 of the Online Transactional Oracle Master Agreement, here: https://www.oracle.com/a/ocom/docs/lic-online-toma-us-eng-v091120.pdf
9. **Database Software License** – the Oracle Software License Agreement also expressly forbids causing or permitting others to reverse engineer data structures of the software, see section D, 3rd bullet, here: https://www.oracle.com/us/corporate/contracts/olsa-services/olsa-renewals-en-us-v053012-1867431.pdf

Regarding GoldenGate for Apache Kafka or Object Storage

GoldenGate is a great platform for replicating data into and from Apache Kafka and Object Stores.

Oracle GoldenGate was the first CDC / replication tool to support Kafka, and the LinkedIn team who invented Kafka have made extensive use of their GoldenGate to Kafka implementation over the years. Several of the largest Kafka implementations in the world depend on GoldenGate every day for petabytes worth of changed data being ingested into their event streams. GoldenGate customers can feel confident that the solution will scale to the most demanding and mission-critical Oracle to Kafka use cases on the planet.

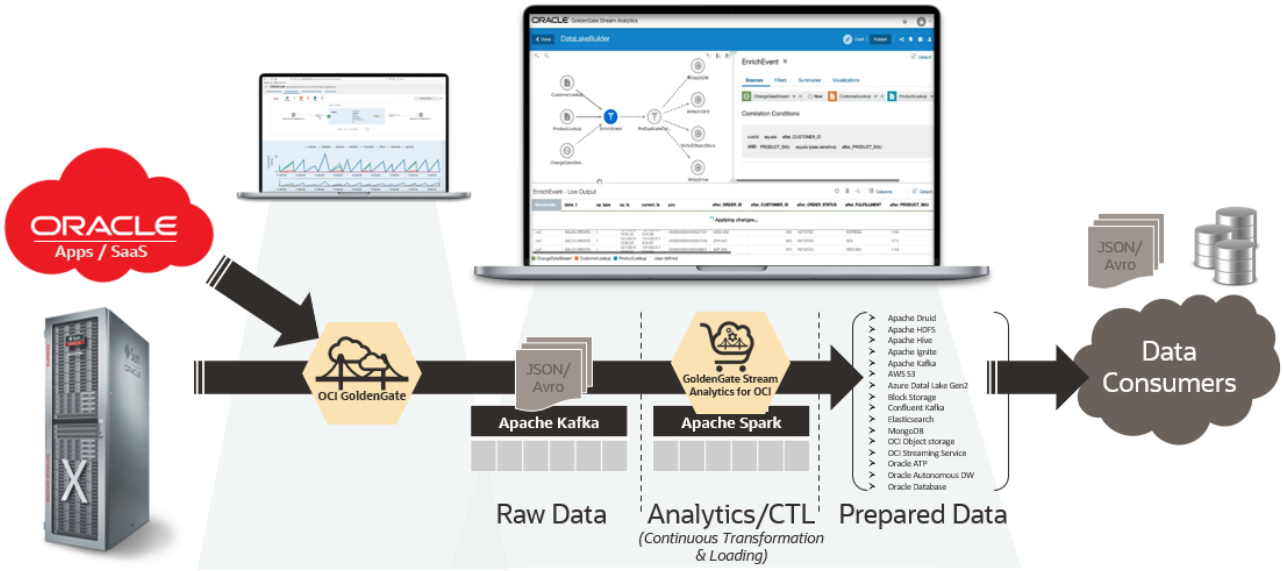

Figure 8: Using GoldenGate for CDC, Data Integration and Stream Analytics

The high-level objective is to stream database events from source databases to Kafka. But in doing this, it is crucial to preserve the transaction boundaries (DML and DDL) for transaction integrity. This is important because messaging systems like Kafka do not have ACID (Atomicity, Consistency, Integrity and Durability) properties. When the GoldenGate target is an ACID capable store (eg; relational databases, Apache Hive, MongoDB, etc) the user can be certain that the data events are correctly handled as they are written and merged into the target data store. This provides the huge benefit of reliable 'synchronized data' among ACID capable data stores.

In the case where GoldenGate is writing data events into a non-ACID store (eg; Apache Kafka, object storage, search-based indices etc) it often becomes necessary for GoldenGate to take an extra step to 'decorate' the payloads with extra metadata so that the consumer of the data events can reconstruct the data events correctly, keep track of schema changes, and decipher the position of readers in case of the need to perform rollback operations.

For example, the kind of metadata that GoldenGate users often want to see in their data events include:

- Transaction before and after images
- Source operation type (insert, update, delete, truncate, create, alter, drop)
- Source commit sequence number (SCN, LSN, RBA, etc.)
- Source commit timestamp
- Source metadata such as table and column definitions
- GoldenGate target trail sequence number
- GoldenGate target trail RBA
- Metadata versions

- Source database name & type
- Source operation log position (if available)
- Source transaction user & name (if available)

GoldenGate can send transactions to 100's of different data store and platform combinations. Specifically for Kafka, the GoldenGate output formats support a variety of non-relational payloads such as: CSV, Fixed-length, XML, JSON, Avro, Parquet, ORC. Increasingly, advanced filetypes are trying to inject high levels of schema evolution and other relational-like attributes, for example as with Avro in Kafka registries and Parquet in metastore catalogs. But none of these payload formats are anywhere near capable of handling the richness and semantics of ACID capable DBs.

Ultimately, developers are left to stitch together the isolation and consistency on their non-ACID targets, but the GoldenGate metadata goes a long way towards making this simpler. If you choose to use GoldenGate for Stream Analytics, there are built-in patterns that support GoldenGate metadata out-of-the-box by default (e.g. for partial/full supplemental logging), this further simplifies and 'bakes in' best practices for dealing with consistency and isolation within stream processing. These holistic stream processing and stream analytics use cases are increasingly becoming critical for modern data platforms that are dynamic and real-time.

Summary – Oracle GoldenGate is the best CDC for Oracle Database

It should not be a surprise that the Oracle engineering teams are best equipped to also provide the best overall CDC and replication solution for the Oracle Database. GoldenGate is regularly recognized by customers and analysts as the most functionally complete, highest performing, most trusted, and reliable data integration solution for the Oracle Database, as well as for the 100's of additional supported data platforms.

In this document we have explained numerous limitations and considerations for third-party CDC technologies working with Oracle databases. This table summarizes those CDC approaches.

[best] ●●●⊙○ [worst]	GoldenGate	XStream	LogMiner	Triggers	Disk Readers
Developer Experience					
Low Code Development	●	○	○	○	3rd party
Microservices APIs & Mesh Deployments	●	○	○	○	○
Built-in Monitoring and Metrics	●	○	○	○	3rd party
Elastic, Native Cloud Service	●	Oracle ADB	Oracle ADB	Oracle ADB	○
Best for Oracle Databases					
Highest Performance (capture and replication)	●	◉	⊙	○	storage-bound
Remote Capture/Apply	●	●	●	●	◉
Datatype Support	●	●	⊙	◉	⊙
Database HA / Recovery Capabilities	●	◉	◉	⊙	⊙
Certified for Oracle Max Availability	●	○	○	○	○
Multitenant, Container & Pluggable DB Support	●	◉	⊙	◉	⊙
Oracle Compression Support	●	●	●	●	○
Security (cloud DBs, TDE, wallets, vaults etc)	●	●	●	●	old DBs only
Heterogeneous Data Ecosystem					
Non-Oracle Databases	●	○	○	○	varies
Mainframes (DB2/z, Tandem, etc)	●	○	○	○	varies
Big Data, NoSQL, Kafka etc.	●	○	○	○	varies
Multi-cloud (hosting & data store support)	●	○	○	○	varies
Other					
Supported (Oracle will take SR on issues)	yes	yes	yes	yes	no
Licensing	GoldenGate	GoldenGate	Oracle DB	Oracle DB	unlicensed

Figure 9: Comparison of Oracle GoldenGate to alternative change capture APIs and approaches

No other third-party technology can come close to GoldenGate.

We hope that this analysis will help you choose the right replication solutions for your business use and address your requirements for change data capture streams.

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

blogs.oracle.com facebook.com/oracle twitter.com/oracle

Authors: Oracle Product Development, Ionut Bruma