

Oracle GoldenGate 12c: Fundamentals for Oracle

Student Guide

D84357GC10

Edition 1.0

January 2014

D85201

ORACLE®

Author

Steve Friedberg

**Technical Contributors
and Reviewers**

Mack Bell
Joseph deBuzna
Pete Daly
Joe Greenwald
Steven George
Susan Jang
Naoki Kato
Juan Quezada Nunez
Randy Richeson
Doug Reid
Ranbir Singh
Jinyu Wang
Volker Zell

Editors

Aju Kumar
Richard Wallis

Graphic Designer

Seema Bopaiah

Publishers

Pavithran Adka
Jayanthi Keshavamurthy
Srividya Rameshkumar

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Disclaimer

This document contains proprietary information and is protected by copyright and other intellectual property laws. You may copy and print this document solely for your own use in an Oracle training course. The document may not be modified or altered in any way. Except where your use constitutes "fair use" under copyright law, you may not use, share, download, upload, copy, print, display, perform, reproduce, publish, license, post, transmit, or distribute this document in whole or in part without the express authorization of Oracle.

The information contained in this document is subject to change without notice. If you find any problems in the document, please report them in writing to: Oracle University, 500 Oracle Parkway, Redwood Shores, California 94065 USA. This document is not warranted to be error-free.

Restricted Rights Notice

If this documentation is delivered to the United States Government or anyone using the documentation on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS

The U.S. Government's rights to use, modify, reproduce, release, perform, display, or disclose these training materials are restricted by the terms of the applicable Oracle license agreement and/or the applicable U.S. Government contract.

Trademark Notice

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Contents

1 Introduction

- Objectives 1-2
- Oracle GoldenGate 12c 1-3
- Key Capabilities and Technology Differentiators 1-4
- Value Propositions for Oracle GoldenGate 1-5
- Oracle GoldenGate Topologies 1-6
- Oracle Middleware for Business Intelligence 1-7
- Oracle Data Integrator EE and Oracle GoldenGate 1-8
- Oracle GoldenGate for Real-Time Data Warehousing 1-9
- Oracle GoldenGate Solutions for Oracle Database 1-10
- Oracle GoldenGate for Oracle Database Eliminate Down Time for Migrations and Application Upgrades 1-11
- Oracle GoldenGate for Oracle Database Eliminate Down Time During Oracle Database Upgrades 1-12
- Oracle GoldenGate for Oracle Database Eliminate Unplanned Down Time with Active Data Guard 1-13
- Oracle GoldenGate for Oracle Database Improve Production System Performance and Lower Costs 1-14
- Oracle GoldenGate for Oracle Database Offload Redo Logs 1-15
- Oracle GoldenGate for Operational Reporting 1-16
- Oracle GoldenGate for Oracle Database Increase Return on Investment (ROI) on Existing Servers and Synchronize Global Data 1-17
- Quiz 1-18
- Summary 1-20

2 Technology Overview

- Objectives 2-2
- Roadmap 2-3
- Oracle GoldenGate: Modular Building Blocks 2-4
- Roadmap 2-10
- Supported Databases 2-11
- Supported Operating Systems 2-12
- Roadmap 2-13
- Oracle GoldenGate Product Line 2-14
- Oracle GoldenGate Veridata 2-15

Oracle GoldenGate Director: Overview	2-16
Oracle GoldenGate Director	2-17
Oracle GoldenGate Monitor: Overview	2-18
Oracle GoldenGate Monitor	2-19
Enterprise Manager Cloud Control 12c	2-20
Adapter Integration Options for Oracle GoldenGate	2-21
Oracle GoldenGate for Flat File	2-22
Oracle GoldenGate Application Adapter for Java	2-23
JMS Messaging Capture	2-24
Quiz	2-25
Summary	2-27

3 Oracle GoldenGate Architecture

Objectives	3-2
Roadmap	3-3
Uses of Oracle GoldenGate	3-4
Oracle GoldenGate Components	3-5
Oracle GoldenGate Logical Architecture	3-6
Oracle GoldenGate Process Groups	3-7
Process-Group Naming Conventions	3-8
GGSCI	3-10
Parameter Files	3-11
Roadmap	3-12
Change Data Capture (Extract) and Delivery	3-13
Change Data Capture (Extract) and Delivery Using a Data Pump	3-15
Extract Flavors	3-16
Distributed Topologies for Integrated Extract: Same Machine	3-17
Distributed Topologies for Integrated Extract: Different Machine, Real-Time	3-18
Distributed Topologies for Integrated Extract: Different Machine, Non-Real-Time	3-19
Supported Features Based on Source DB Version	3-20
Roadmap	3-21
Online Versus Batch Operation	3-22
Running an Initial Load	3-23
Initial Load	3-24
Roadmap	3-25
Checkpoints: Capture	3-26
Checkpoints: Pump	3-27
Checkpoints: Delivery	3-28

Commit Sequence Number (CSN) 3-29
Discussion Questions 3-30
Quiz 3-31
Summary 3-32

4 Installing Oracle GoldenGate

Objectives 4-2
Roadmap 4-3
Software System Requirements 4-4
Operating System Requirements 4-5
Downloading Oracle GoldenGate 4-7
Roadmap 4-8
Installation on UNIX, Linux, or z/OS Systems 4-9
Oracle Universal Installer GUI (New with 12c) 4-10
Installation on Windows Systems 4-12
Oracle GoldenGate Directories 4-13
Oracle GoldenGate Documentation 4-15
Roadmap 4-16
GGSCI Command Interface 4-17
GGSCI Commands 4-18
GGSCI Examples 4-22
Obey Files 4-23
Oracle GoldenGate 12c: Miscellaneous New Features 4-24
Running Oracle GoldenGate from the OS Shell 4-25
Discussion Questions 4-26
Summary 4-27
Practice 4 Overview: Installing Oracle GoldenGate 4-28

5 Configuration Overview and Preparing the Environment

Objectives 5-2
Roadmap 5-3
Configuring Oracle GoldenGate 5-4
Character Set: National Language Support (NLS) 5-5
Mixed-Case Object Names: ‘Single’ and “Double” Quotation Marks 5-6
Configuring Oracle GoldenGate 5-7
Preparing the Environment: Oracle Database 5-8
Using Command Security 5-9
Sample CMDSEC Statements 5-10
Handling TCP/IP Errors 5-11
tcperrs File 5-12
Roadmap 5-13

TranData Command	5-14
Preparing the Environment: Oracle Database	5-15
Preparing the Environment: Manager Overview	5-16
Preparing the Environment: Configuring Manager	5-17
Preparing the Environment: Sample Manager Parameter File	5-18
Roadmap	5-19
Preparing the Environment: Overview of Source Definitions	5-20
Preparing the Environment: Running defgen	5-21
Quiz	5-23
Summary	5-25
Practice 5 Overview: Configuration Overview and Preparing the Environment	5-26

6 Configuring Change Capture: Extract

Objectives	6-2
Roadmap	6-3
Step 2: Change Capture	6-4
Extract: Overview	6-5
Roadmap	6-6
Data Pump: Overview	6-7
Data Pumps: One-to-Many Trails	6-9
Data Pumps: One-to-Many Target Systems	6-10
Roadmap	6-11
Setting Up Change Capture (Extract)	6-12
Add Extract Command	6-13
Add Extract: Examples	6-14
Editing Extract Parameters	6-15
User ID and Password Aliases (New with 12c)	6-16
Passive Alias Extract	6-17
Converting to Integrated Capture	6-19
Roadmap	6-20
Overview of Trails	6-21
Adding a Local or Remote Trail	6-22
Starting the Extract	6-23
Primary Extract Configuration for Oracle	6-24
Data Pump Configuration for Oracle	6-25
Roadmap	6-26
Automatic Storage Management (ASM)	6-27
Ensuring ASM Connectivity	6-28

ASM and DBLogReader	6-29
Discussion Questions	6-30
Summary	6-32
Practice 6 Overview: Configuring Change Capture	6-33

7 Configuring Change Delivery: Replicat

Objectives	7-2
Roadmap	7-3
Step 4: Change Delivery (Replicat)	7-4
Replicat: Overview	7-5
Replicat “Classic”	7-6
Replicat “Integrated” a.k.a. “Integrated Delivery” (New with 12c)	7-7
Coordinated Replicat (New with 12c)	7-8
Roadmap	7-9
Change Delivery Tasks	7-10
CheckpointTable	7-11
Sample Configuration	7-12
Roadmap	7-13
Avoiding Collisions with Initial Load	7-14
Handling Collisions with Initial Load	7-15
Roadmap	7-16
Converting to or from Integrated Apply (New with 12c)	7-17
Roadmap	7-18
Obtaining Process Information Through GGSCI	7-19
Process Report Files	7-21
Sample Extract Process Report	7-22
Discard Files	7-23
Using the ggserr.log Error Log	7-24
Using the System Logs	7-25
Discussion Questions	7-26
Quiz	7-27
Summary	7-29
Practice 7 Overview: Configuring Change Delivery	7-30

8 Extract Trails and Files

Objectives	8-2
Roadmap	8-3
Overview of Extract Trails and Files	8-4
Extract Trails and Files Distribution	8-5
Extract Trails and Files Contents	8-6
Extract Trails and Files Cleanup	8-7

Trail Format	8-8
Record Header Area	8-9
Record Data Area	8-10
Setting the Compatibility Level	8-11
Roadmap	8-12
Alternative Trail Formats	8-13
Logical Change Records (LCRs)	8-14
FormatASCII	8-15
FormatASCII Sample Output	8-16
FormatSQL	8-17
FormatSQL Sample Output	8-18
FormatXML	8-19
FormatXML Sample Output	8-20
Roadmap	8-21
logdump Utility	8-22
Opening a Trail	8-23
Setting Up a View	8-24
Viewing the Trail File Header	8-25
Viewing Trail Records	8-26
Viewing Canonical Trail Records	8-27
Counting Records in the Trail	8-28
Filtering by a File Name	8-30
Locating a Hex Data Value	8-31
Saving Records to a New Trail	8-33
Keeping a Log of Your Session	8-34
Roadmap	8-35
Overview of the reverse Utility	8-36
Overall Process of the reverse Utility	8-37
reverse: Overall Process	8-38
Sample Parameter Files	8-39
Discussion Questions	8-40
Summary	8-41
Practice 8 Overview: Using Extract Trails and Files	8-42

9 Configuring Initial Load

Objectives	9-2
Roadmap	9-3
Step 3: Initial Load	9-4
Initial Load: Advantages of Oracle GoldenGate Methods	9-5
Initial Load: Resource Limitations	9-6
Prerequisites for Initial Load	9-7

Initial Load: Oracle GoldenGate Methods	9-8
Roadmap	9-9
Initial Load: File to Replicat	9-10
Initial Load: File to Database Utility	9-11
Roadmap	9-13
Initial Load: Direct Load	9-14
Initial Load: Direct Bulk Load (to Oracle)	9-16
Discussion Questions	9-17
Summary	9-18
Practice 9 Overview: Configuring the Initial Load	9-19

10 Oracle GoldenGate Parameters

Objectives	10-2
Roadmap	10-3
Oracle GoldenGate Parameter Files	10-4
Using Parameter Files	10-5
GLOBALS Versus Process Parameters	10-6
GLOBALS Parameters	10-7
Roadmap	10-8
Manager Parameters: Overview	10-9
Sample Manager Parameter File	10-10
Manager Parameter Categories	10-11
Managing Trail Files	10-12
Roadmap	10-13
Extract Parameter: Overview	10-14
Extract Parameter Defaults	10-15
Sample Extract Parameter File	10-16
Extract Parameter Categories	10-17
Extract Example: Table Parameter	10-19
Extract Example: TranLogOptions Parameter	10-20
Roadmap	10-22
Replicat Parameters: Overview	10-23
Replicat Parameter Defaults	10-24
Sample Replicat Parameter File	10-25
Replicat Parameter Categories	10-26
Replicat Example: Map Parameter	10-28
DBOptions	10-30
Discussion Questions	10-31
Summary	10-32
Practice 10 Overview: Modifying Parameters	10-33

11 Data Selection and Filtering

- Objectives 11-2
- Roadmap 11-3
- Data Mapping and Manipulation: Overview 11-4
- Types of Definition Files 11-5
- Data Selection: Overview 11-6
- Roadmap 11-7
- Data Selection: Where Clause 11-8
- Data Selection: Where Clause Examples 11-10
- Roadmap 11-11
- Data Selection: Filter Clause 11-12
- Data Selection: Filter Clause Examples 11-13
- Roadmap 11-14
- Data Selection: Range Function 11-15
- Data Selection: Range Function Examples 11-16
- Coordinated Apply (New with 12c) 11-18
- Roadmap 11-19
- Column Mapping: Overview 11-20
- Column Mapping: Example 11-21
- Column Mapping: Building History 11-22
- Data Transformation Using Functions 11-23
- Functions: Performing Tests on Column Values 11-24
- @IF Function 11-25
- Functions: Working with Dates 11-26
- @Date Function 11-27
- Functions: Working with Strings and Numbers 11-28
- @StrCat Function 11-30
- @StrExt Function 11-31
- Other Functions 11-32
- Roadmap 11-33
- SQLEXEC: Overview 11-34
- SQLEXEC: Basic Functionality 11-36
- SQLEXEC: DBMS and Data Type Support 11-37
- SQLEXEC: Usage with a LOOKUP Stored Procedure 11-39
- SQLEXEC: Usage with a SQL Query 11-41
- SQLEXEC: Usage in a Table or Map Statement 11-42
- SQLEXEC: Usage as a Stand-Alone Statement 11-43
- Quiz 11-44
- Summary 11-46
- Practice 11 Overview: Data Selection and Filtering 11-47

12 Additional Transformation Topics

Objectives 12-2
Roadmap 12-3
Macros: Overview 12-4
Creating Macros 12-5
Invoking a Macro 12-6
Reusing Parameter Sets 12-7
Creating Macro Libraries 12-9
Tracing Macro Expansion 12-10
Roadmap 12-11
User Tokens: Overview 12-12
Environmental Values Available to @GETENV 12-13
User Tokens Display 12-14
Using User Tokens 12-15
Viewing User Tokens in Logdump 12-16
Roadmap 12-17
User Exits: Overview 12-18
Uses for User Exits 12-19
User Exits: High-Level Processing Logic 12-20
Implementing User Exits 12-21
User Exit Parameters 12-22
Sample User Exits 12-23
Calling User Exits 12-24
Roadmap 12-25
Oracle Sequences: Overview 12-26
Quiz 12-28
Summary 12-30
Practice 12 Overview: Data Transformation 12-31

13 Configuration Options

Objectives 13-2
Roadmap 13-3
BatchSQL: Overview 13-4
BatchSQL Syntax 13-5
BatchSQL Results 13-7
Roadmap 13-8
Compression Options 13-9
Example of Compression 13-10
Compression and Exadata 13-11
Roadmap 13-12
Encryption: Overview 13-13

Message Encryption	13-15
Options: Message Encryption	13-17
Trail or Extract File Encryption	13-18
Trail Encryption with Wallet (new with 12c)	13-19
Password Encryption: Method 1	13-20
Password Encryption: Method 2	13-21
Password Encryption: Method 3	13-22
Summary of Password Encryption	13-23
Roadmap	13-24
Event Marker System	13-25
Uses for Event Actions	13-26
Event Actions Flowchart	13-27
EventActions Order	13-28
Implementing Event Actions: Examples	13-29
Event Actions: Heartbeat Example	13-30
Event Actions: Automated Switchover Example	13-31
Event Actions: Automated Synchronization Example	13-32
Quiz	13-33
Summary	13-35
Practice 13 Overview: Configuration Options	13-36

14 Bidirectional Replication

Objectives	14-2
Roadmap	14-3
Bidirectional Flowchart	14-4
Capabilities of a Bidirectional Configuration	14-5
Bidirectional Configuration Considerations	14-6
Roadmap	14-7
Preventing Data Looping	14-8
Loop Detection Techniques	14-10
Roadmap	14-12
Conflict Avoidance and Detection and Resolution	14-13
Conflict Detection by CompareCols	14-14
GetUpdateBefores, GetBeforeCols, CompareCols, and ResolveConflict	14-15
Conflict Detection by Filter	14-17
Conflict Resolution	14-18
Conflict Resolution: Example	14-19
Conflict Resolution by Applying Net Differences	14-20
ResolveConflict Built-in Methods	14-21
Conflict Resolution: Custom Methods	14-22
Roadmap	14-23

Oracle Sequence Numbers	14-24
Truncate Table Operations	14-25
Quiz	14-26
Summary	14-28
Practice 14 Overview: Configuring Bidirectional Replication	14-29

15 DDL Replication

Objectives	15-2
Roadmap	15-3
Overview of DDL Replication	15-4
DDL Replication Requirements and Restrictions	15-5
Characteristics for DDL Replication	15-8
Roadmap	15-9
DDL Scopes	15-10
DDL Parameter	15-11
DDL String Substitution	15-13
DDL Error Handling	15-14
DDLOptions for Oracle	15-15
Mapping Schemas	15-17
Roadmap	15-18
Supporting DDL in an Active-Active Bidirectional Configuration	15-19
Activating Oracle DDL Capture	15-20
DDL Setup Scripts Classic versus Integrated	15-22
Quiz	15-23
Summary	15-25
Practice 15 Overview: Configuring DDL Replication	15-26

1

Introduction

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

- Describe the features and functionality of Oracle GoldenGate 12c (version 12.1.2)
- Identify key capabilities and differentiators
- Describe Oracle GoldenGate high-availability and disaster tolerance solutions
- Describe Oracle GoldenGate real-time data integration solutions



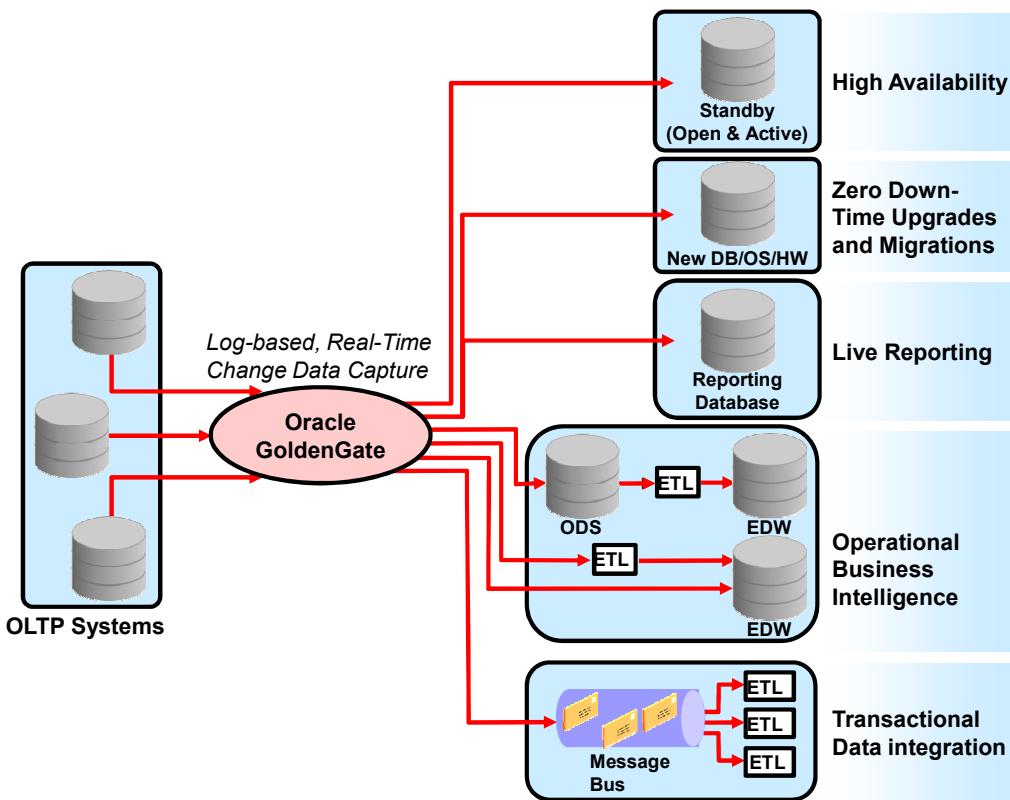
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Scenario

Suppose that you are a database administrator (DBA) for a bank that has offices in the Americas and in Europe. There are separate Oracle 12c databases on both continents. You need to replicate some tables from West (schema AMER) to East (schema EURO). To do this, you try a proof-of-concept with Oracle GoldenGate for Oracle 12c.

The constraint that makes this just proof-of-concept (rather than production) is that you are doing this on one PC. In the real world, you would use separate East and West host machines.

Oracle GoldenGate 12c



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

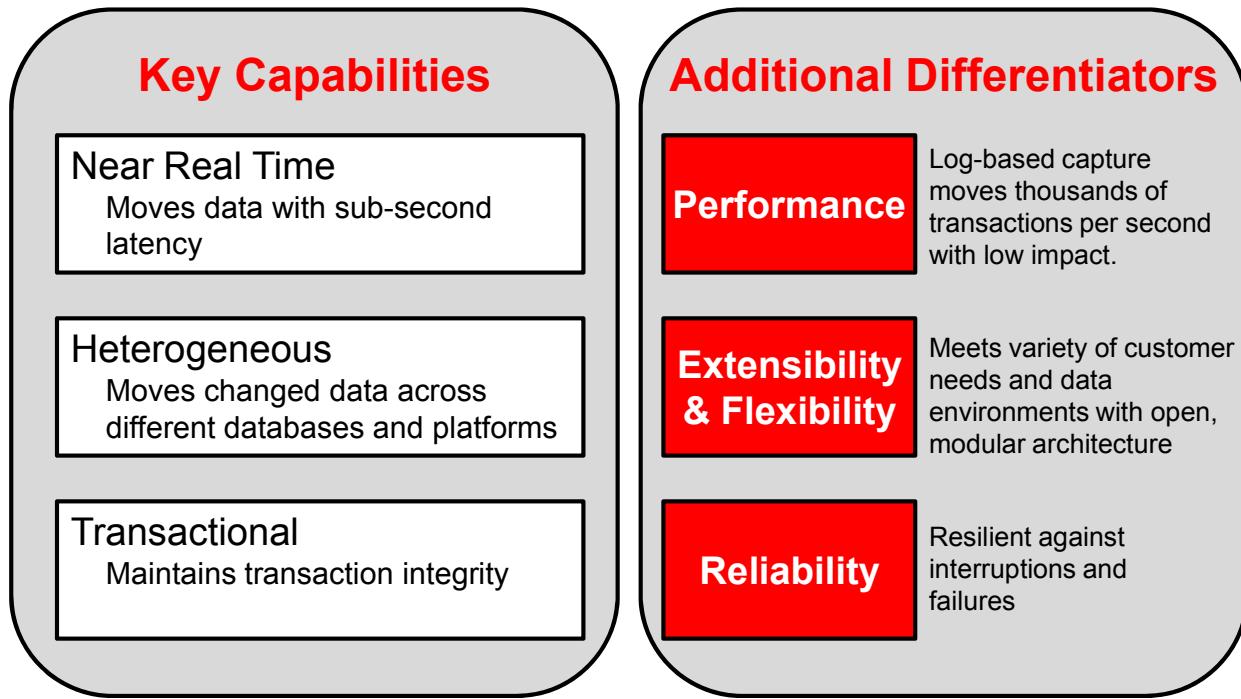
Glossary

- **DB:** database
- **EDW :** Enterprise Data Warehouse
- **ETL:** Extract, Transform, and Load
- **HW:** Hardware (Intel 32-bit, Intel 64-bit, SPARC, and so on)
- **ODS:** Operational Data Store
- **OLTP:** Online Transaction Processing
- **OS:** operating system (Linux, Windows, and so on)

Oracle GoldenGate 12c provides low-impact capture, routing, transformation, and delivery of database transactions across heterogeneous environments in near-real time.

Oracle GoldenGate enables the exchange and manipulation of data at the transaction level among multiple, heterogeneous platforms across the enterprise. It moves committed transactions from redo logs and maintains transaction integrity with sub-second latency. Its reliability is especially critical for enabling continuous-availability solutions, including high availability, zero down time for upgrades and migrations, live reporting, operational business intelligence, and transactional data integration.

Key Capabilities and Technology Differentiators



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate:

- Is a middleware product designed to work in a heterogeneous environment with different databases
- Moves only committed data across platforms, which allows for sub-second latency. This is different from an Oracle database, which writes committed and uncommitted changes to the redo logs.
- Can move changes across a TCP/IP network and does not require Oracle Net
- Uses its own system of checkpoint files to maintain transaction integrity, and does not use a concept of multiplexing like an Oracle database
- Can quickly move data to a standby database that can support disaster recovery. However, it does not provide an “automatic failover” capability like Oracle Data Guard.
- Uses its own commit sequence number (CSN) to identify a transaction, which is based on the Oracle Database SCN

Value Propositions for Oracle GoldenGate

Oracle GoldenGate 12c:

- Delivers continuous operations
- Lowers IT costs
- Improves efficiencies
- Reduces risk
- Reduces barriers to sharing data

ORACLE®

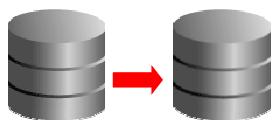
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate can save time and money because it:

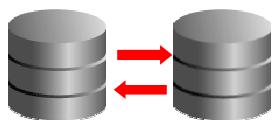
- Delivers continuous operations for mission-critical applications to eliminate unplanned and planned down time and related costs
- Lowers IT costs through heterogeneous support for multiple platforms to leverage lower-cost infrastructure for query offloading
- Improves efficiencies through improved performance, scalability of real-time feeds, and data distribution
- Reduces risk by ensuring data integrity and reliability between source and target systems
- Reduces barriers to sharing data because it has no application impact for real-time data acquisition; provides improved visibility and business insight

Oracle GoldenGate Topologies

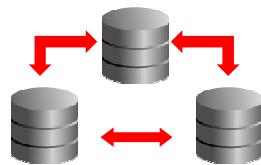
Unidirectional Query Offloading



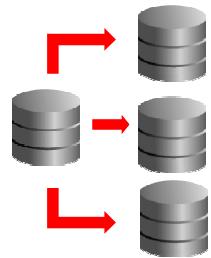
Bidirectional Standby DB or Active-Active for HA



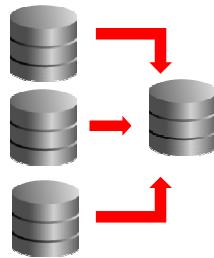
Peer-to-Peer Load Balancing, Multimaster



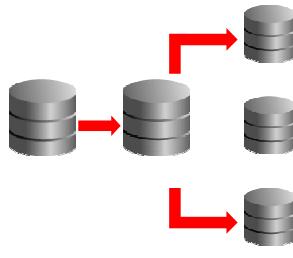
Broadcast Data Distribution



Integration/Consolidation Data Warehouse



Cascading Data Marts

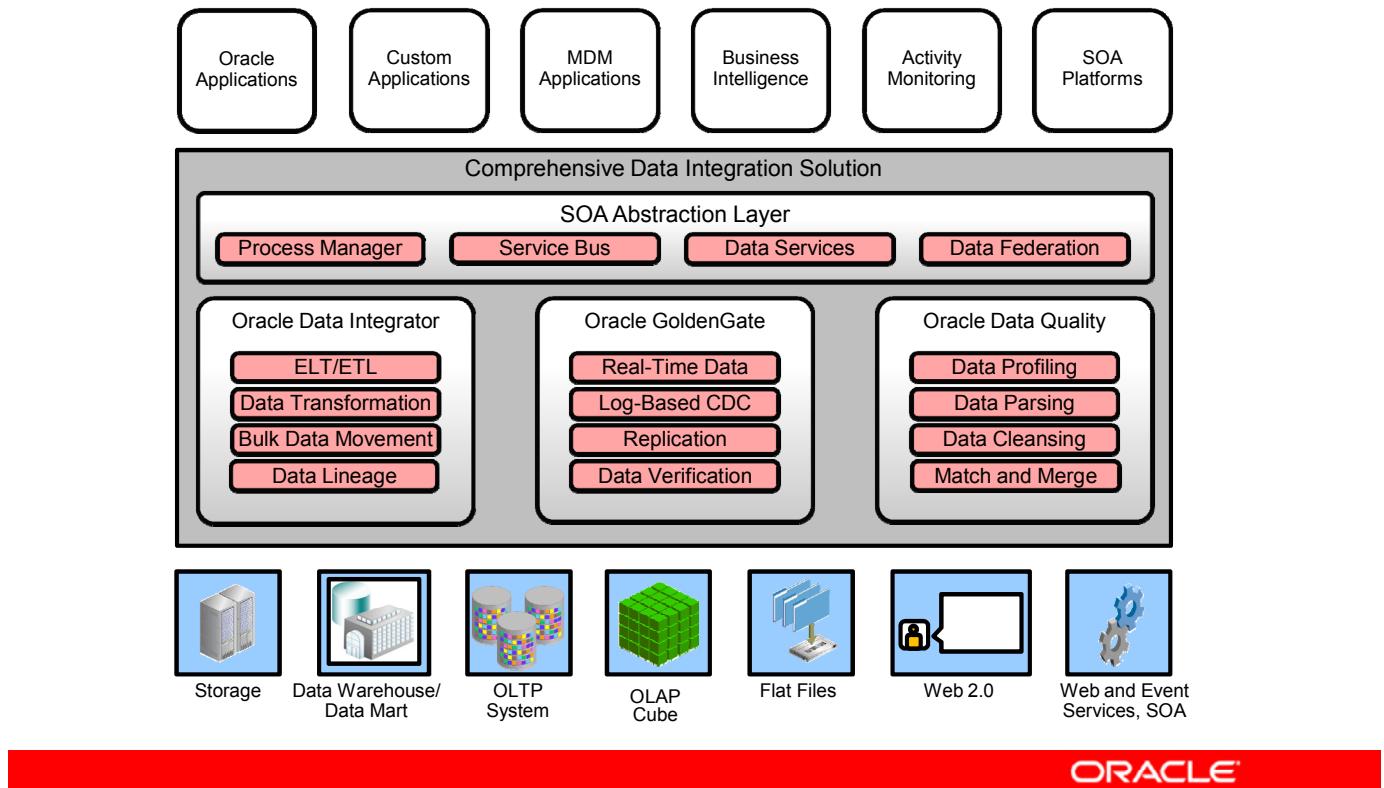


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate has a modular architecture that provides the flexibility to extract and replicate selected data records, transactional changes, and changes to data definition language (DDL) across a variety of topologies. With this flexibility, plus the filtering, transformation, and custom processing features of Oracle GoldenGate, you can support numerous business requirements, including:

- Business continuance and high availability
- Initial load and database migration
- Data integration
- Decision support and data warehousing

Oracle Middleware for Business Intelligence



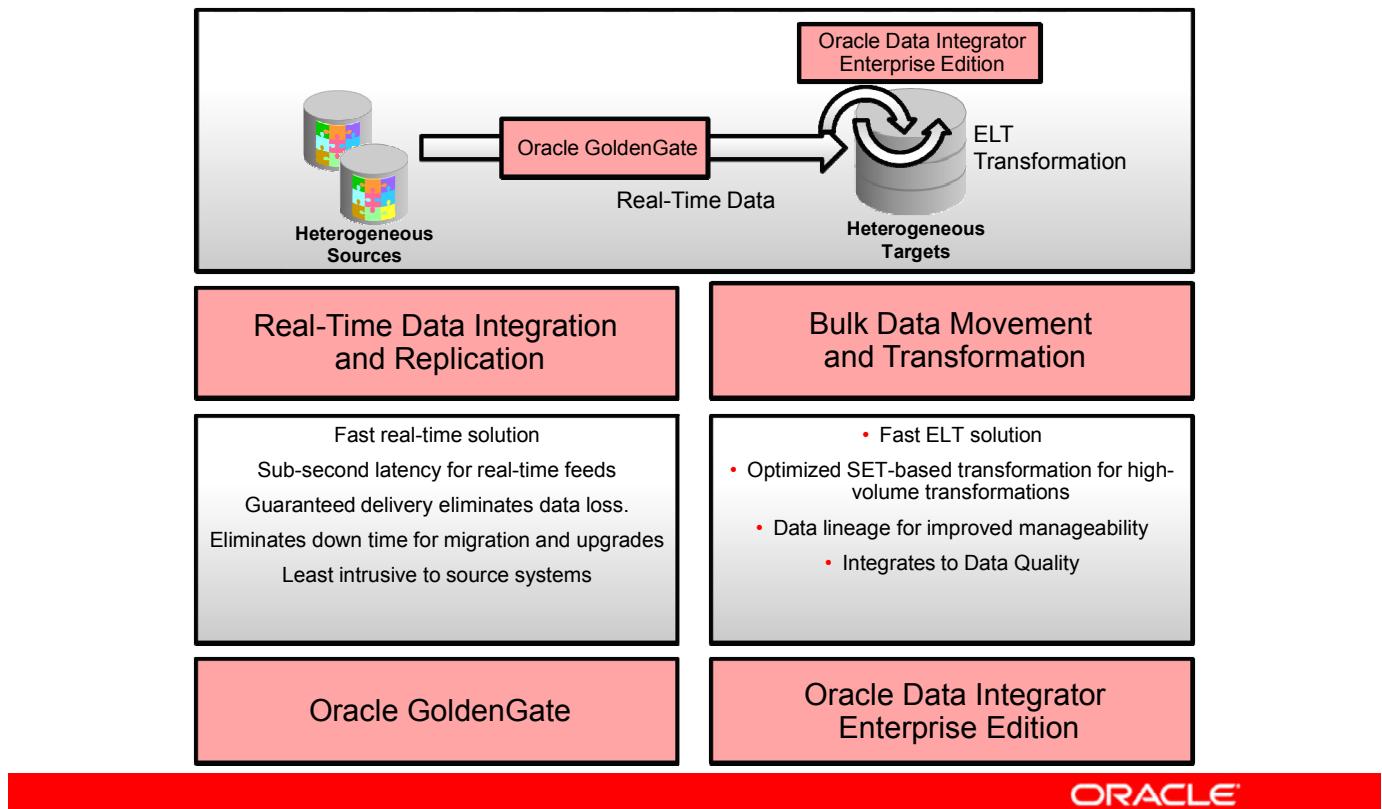
ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Integrating data and applications throughout the enterprise, and presenting them in a unified view, is a complex task. Not only are there broad disparities in technologies, data structures, and application functionality, but there are also fundamental differences in integration architectures. Some integration needs are data oriented, especially those involving large data volumes. Other integration projects lend themselves to an event-driven architecture (EDA) or a service-oriented architecture (SOA), for asynchronous or synchronous integration.

Oracle offers comprehensive solutions for data integration that help move data in bulk across heterogeneous sources and targets. With the best-in-class extract, load, transform (ELT) technology of Oracle Data Integrator (ODI) EE and best-in-class data cleansing technology (Oracle Data Profiling and Data Quality), these solutions can be integrated together with SOA approaches to build reusable data services. Oracle GoldenGate now completes the picture with the addition of real-time change data capture and replication for high availability.

Oracle Data Integrator EE and Oracle GoldenGate



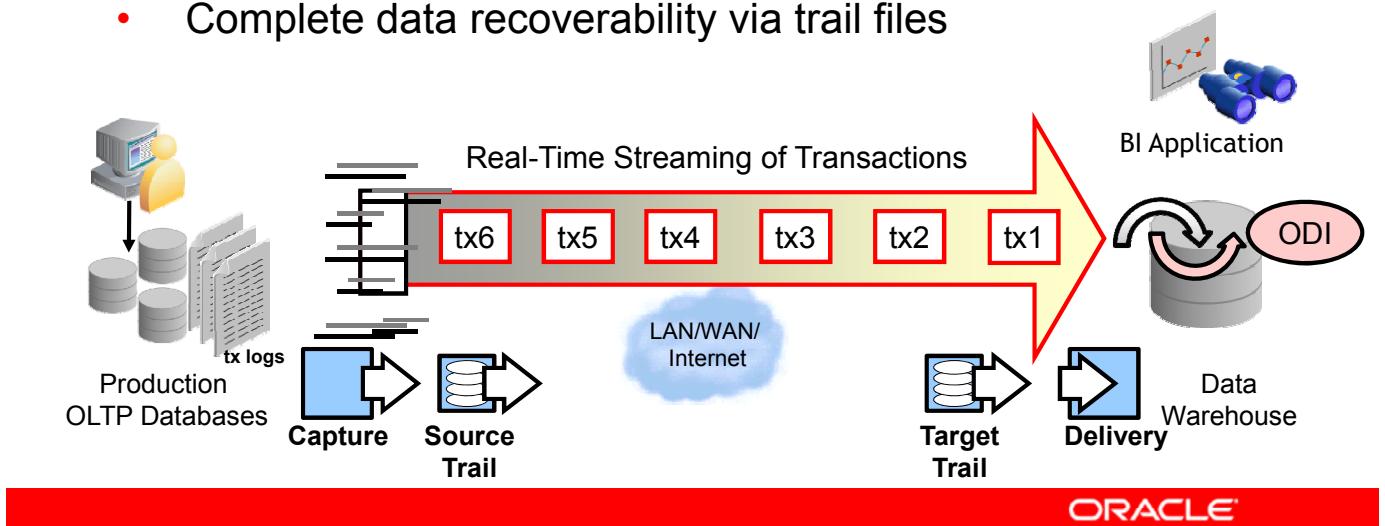
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate and Oracle Data Integrator EE are combined to deliver real-time data warehousing. This combination brings fast real-time data integration and fast bulk-data transformation. Combining ODI EE with Oracle GoldenGate's real-time low-impact change data capture helps customers use integrated data quality and fast bulk transformations. Oracle GoldenGate feeds the staging area of the data warehouse, and ODI EE performs bulk transformations to feed the user tables in the DW for a low total cost of ownership (TCO) and high-performance Real-Time Data Warehouse (RTDW) solution.

Oracle GoldenGate for Real-Time Data Warehousing

- Sub-second data latency
- Minimal overhead and no batch windows
- High-performance, in-database transformations
- Read-consistent changed data with referential integrity
- Complete data recoverability via trail files



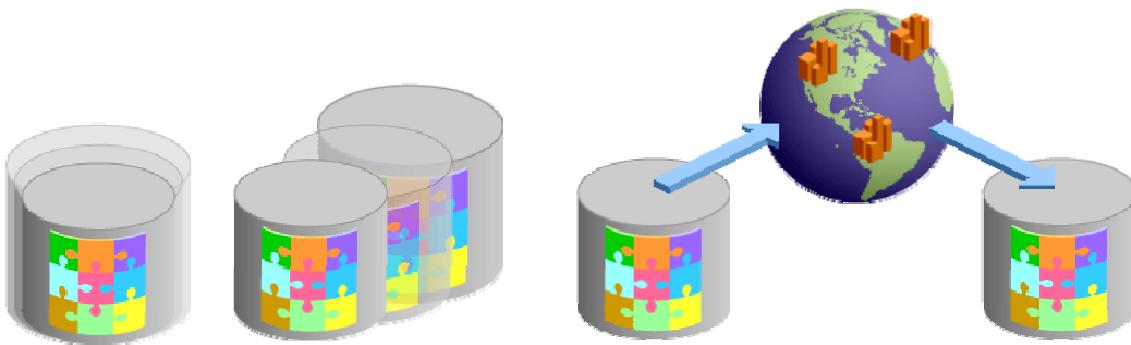
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

For real-time data warehousing environments, Oracle GoldenGate captures and delivers changed data to the data warehouse or operational data store in near real time. Because this is log based, there is minimal impact on the source, there are no batch windows, and it moves the new transactions in a source system in sub-seconds. During the movement phase, each transaction's commit boundaries are maintained to ensure data integrity. ODI performs complex transformations within the database for maximum performance.

The other benefit of this approach is data recoverability in case there is an outage during data movement. This is an important requirement because data latency decreases in feeding the analytical environment. Oracle GoldenGate's trail files that store the changed data are persistent, so they can be reapplied (if needed) to the target and also to the source system.

Oracle GoldenGate Solutions for Oracle Database

- Continuous availability via active-active databases
- Zero down-time upgrades, migrations, and maintenance
- Offloading queries from legacy systems to Oracle databases
- Global Data Synchronization for distributed systems



ORACLE®

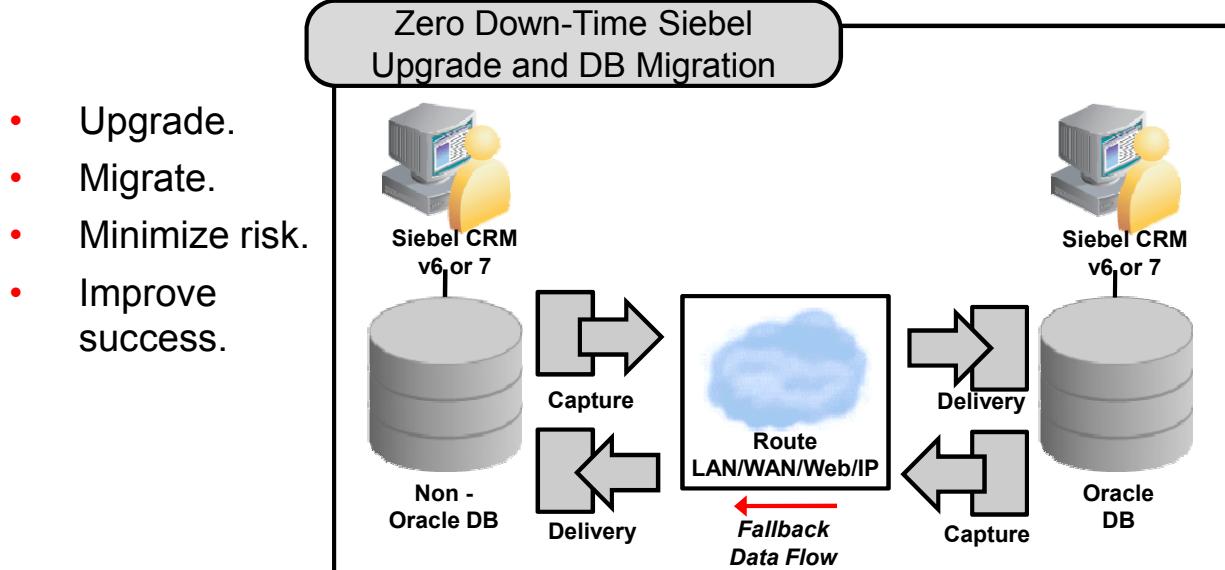
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Migration and maintenance can take many forms:

- Migrate from non-Oracle databases to Oracle 12c.
- Upgrade Oracle Database versions 8*i*, 9*i*, 10*g*, or 11*g* to 12c.
- Upgrade or migrate the database server or operating system (OS).
- Perform database maintenance.
- Perform application upgrades (Siebel CRM).

Oracle GoldenGate for Oracle Database

Eliminate Down Time for Migrations and Application Upgrades



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can use Oracle GoldenGate to:

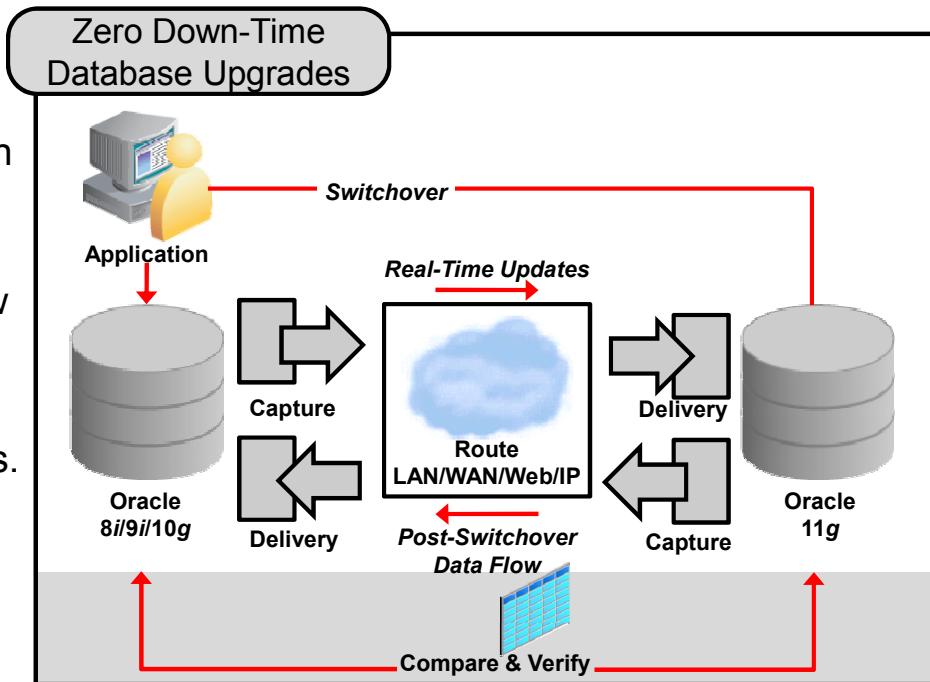
- Upgrade, migrate, and maintain a database, hardware, an OS, or an application
- Minimize risk with the fallback option
- Improve success with phased user migration

Whenever a database, an application, an OS, or hardware must be upgraded or migrated, Oracle GoldenGate enables zero down-time upgrades by synchronizing the new system with the existing one so that users can immediately switch over as soon as the data is fully synced. It also offers a fallback option to go back to the old system if needed for any reason.

Oracle GoldenGate for Oracle Database

Eliminate Down Time During Oracle Database Upgrades

- Zero DB down time for upgrades
- Leverage new features of 12c.
- Minimize risks.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can use Oracle GoldenGate to:

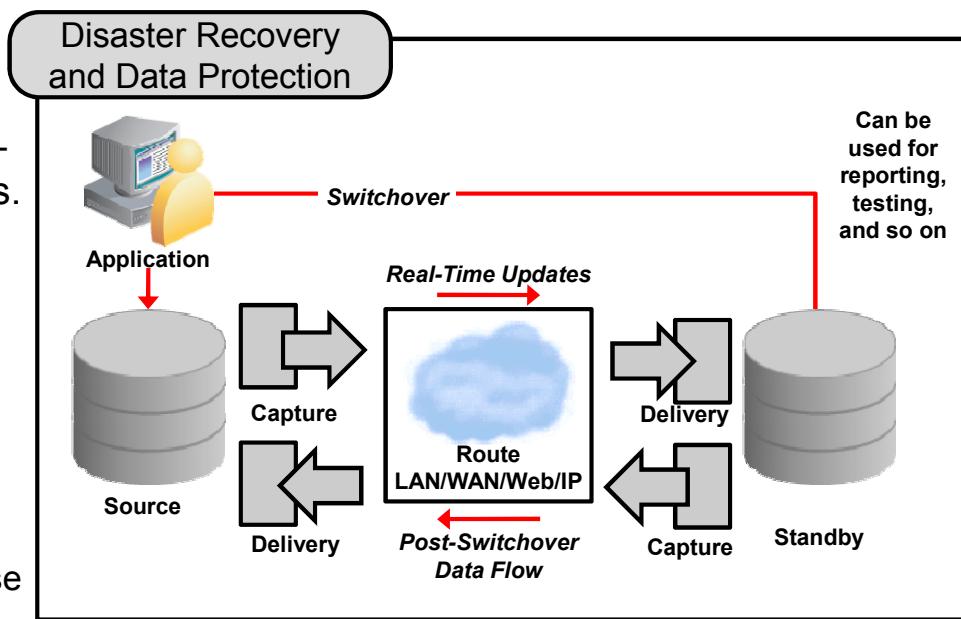
- Upgrade from 8i, 9i, 10g, or 11g to 12c with zero database down time
- Leverage new features of Oracle Database 12c without affecting business operations
- Minimize risks by using the fallback option

A key use case in eliminating planned down time is database upgrades. Oracle GoldenGate enables zero database down time for upgrades, supporting upgrades from versions 8i, 9i, 10g, and 11g to 12c. The solution enables zero database down time by synchronizing databases in different versions and enabling immediate switchover. Customers can also use the two database versions concurrently to make the upgrade completely transparent to application users, as well as to validate data consistency.

Oracle GoldenGate for Oracle Database

Eliminate Unplanned Down Time with Active Data Guard

- Use Active Data Guard for Oracle-Oracle databases.
- Use Oracle GoldenGate for:
 - Non-Oracle platforms
 - Active-active configurations
 - Cross-OS and Oracle database version requirements



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

For disaster recovery, Active Data Guard is a viable option for Oracle databases. Oracle GoldenGate supplements it with non-Oracle platforms, active-to-active configurations, and cross-OS or cross-database versions. Oracle GoldenGate can support a physical standby, logical standby, or live standby database system.

Oracle GoldenGate keeps a live standby database system in sync continuously to enable immediate switchover to the standby system when needed. The secondary system is open for read-only as well as write uses. A post-switchover data flow from standby to primary is also provided. Any data that is processed by the standby during the outages is moved to the primary as soon as it is back online.

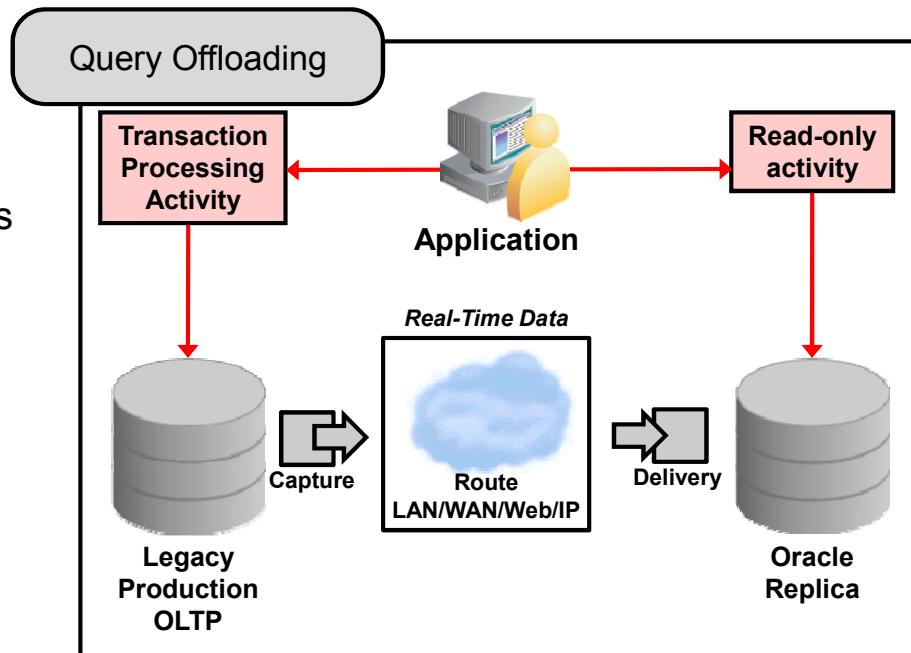
Oracle GoldenGate can also be used *with* Active Data Guard. For example, Active Data Guard can protect a primary database that Oracle GoldenGate is using for a source. Active Data Guard can also be used to protect a primary database that Oracle GoldenGate is using for a target.

Oracle GoldenGate for Oracle Database

Improve Production System Performance and Lower Costs

Offload queries from production systems in:

- Heterogeneous configurations
- Active-active environments



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Heterogeneous configurations may include:

- Different OS or database version, or different type of database
- Legacy system query off-load

Active-active environments may also be doing bidirectional replication.

Oracle GoldenGate can also be used to move data from a CPU-bound source machine to a different target machine that is not CPU bound.

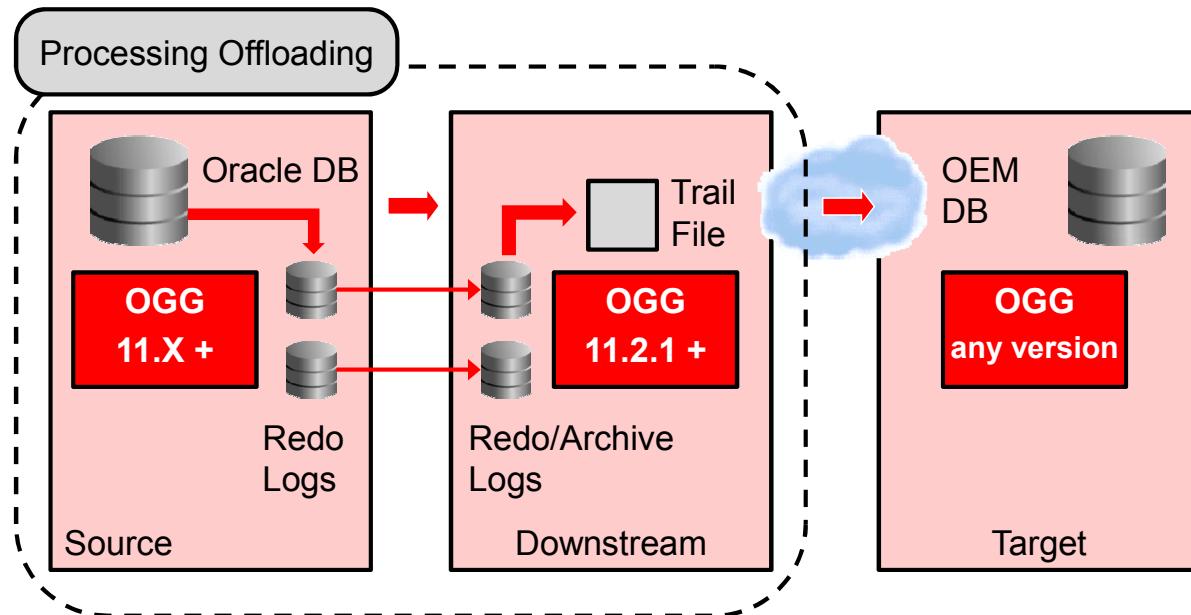
To handle semi-availability issues, Oracle GoldenGate offers the approach of offloading queries thus improving the performance of the production systems by assigning expensive queries to a secondary lower-cost platform.

A good example that requires this solution is an airline. Typically, there are many users who query the airline's database but who may not actually confirm a booking. Allowing these users to perform their queries on a different, continuously synchronized database improves production system performance and extends the life cycle of existing OLTP investments.

Oracle GoldenGate for Oracle Database

Offload Redo Logs

Capture can be offloaded from the source DB to an intermediate host by copying the redo logs.



ORACLE

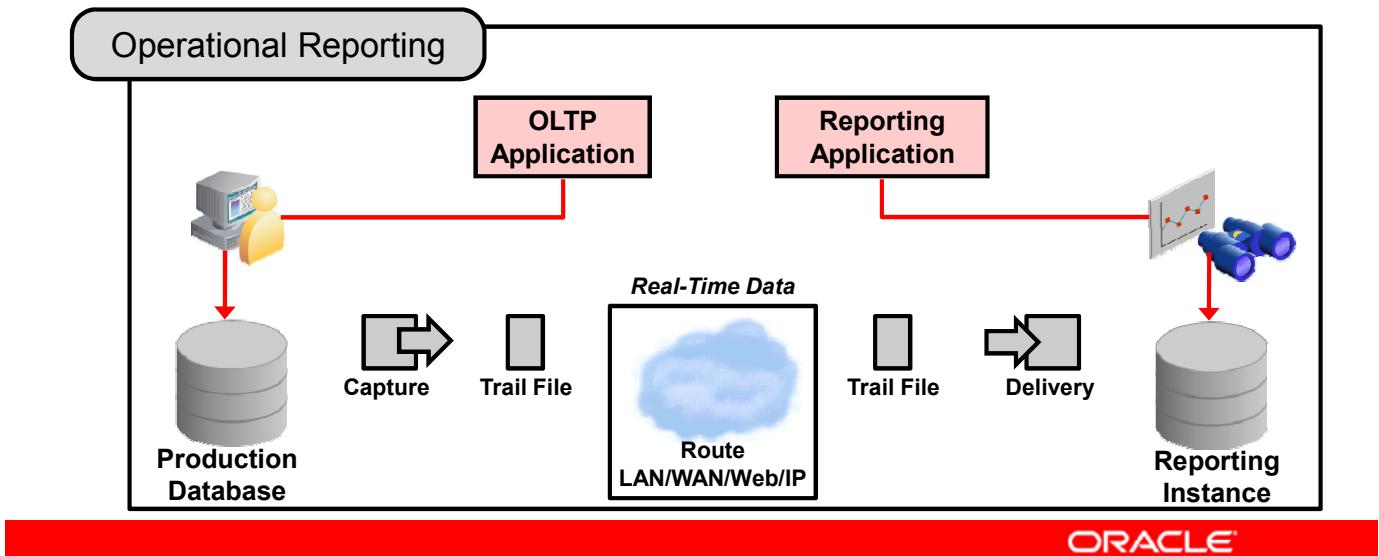
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate (“OGG” in the graphic in the slide) version 11, with the plus sign (+) to show offloading, is the *minimum* version; you can have a later version, such as Oracle GoldenGate12c. Note that the Oracle GoldenGate and Oracle Database release numbers are not necessarily in sync. That is, you can run Oracle GoldenGate 12.1.2 against Oracle Database 11.2.1.

There are certain combinations of versions that are required to support the downstream extraction. The details are covered in the lesson titled “Oracle GoldenGate Architecture.”

Oracle GoldenGate for Operational Reporting

- Sub-second data latency
- No performance degradation for the source system
- Read-consistent changed data with referential integrity
- Complete data recoverability via trail files



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Operational reporting from a single source system or running queries on the source production system can both affect performance. As a result, the best practice is to offload reporting to a lower-cost system.

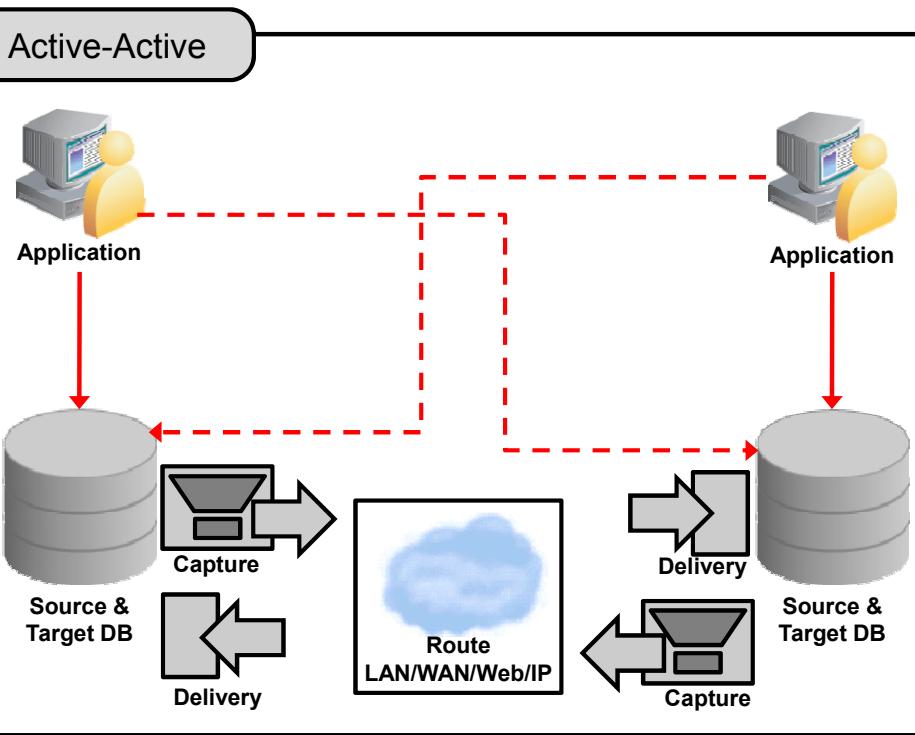
Oracle GoldenGate enables real-time reporting from a lower-cost system by keeping the reporting database in sync with the OLTP. Oracle GoldenGate can move just the data that is needed for reporting, and the heterogeneity enables the use of lower-cost systems.

An example is the offloading from a mainframe to Oracle databases on Linux.

Oracle GoldenGate for Oracle Database

Increase Return on Investment (ROI) on Existing Servers and Synchronize Global Data

- Use secondary systems for transactions.
- Enable continuous availability during unplanned and planned outages.
- Synchronize data across data centers around the globe.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Active-to-active or multimaster database configurations also help to eliminate system performance issues by allowing transaction load distribution between completely parallel systems, which Oracle GoldenGate replicates. Active-to-active configuration also enables continuous system availability because of the ability to work with the remaining databases if one database fails. Data can be filtered to move only certain schemas, tables, or rows.

Quiz

Which three statements are true about Oracle GoldenGate?

- a. Oracle GoldenGate is an Oracle Database product that supports other Oracle products.
- b. Oracle GoldenGate is a middleware product that does not require an Oracle database.
- c. Oracle GoldenGate captures changes from Oracle Redo logs or non-Oracle transaction logs and moves them to another database.
- d. Oracle GoldenGate can support high availability.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b, c, d

Quiz

Oracle GoldenGate is a middleware software for business intelligence, and it is designed to support a heterogeneous database environment.

- a. True
- b. False



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: a

Summary

In this lesson, you should have learned how to describe:

- Oracle GoldenGate features and functionality
- Oracle GoldenGate solutions for real-time business intelligence
- Oracle GoldenGate for continuous availability



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

There is no practice for Lesson 1.

Technology Overview

2

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

- List the building blocks that make up Oracle GoldenGate functionality
- List the supported databases and platforms
- Describe the Oracle GoldenGate product line:
 - Oracle GoldenGate Veridata
 - Management Pack for Oracle GoldenGate
 - Oracle GoldenGate Director
 - Oracle GoldenGate Monitor
 - Oracle GoldenGate for Flat File
 - Oracle GoldenGate for Java

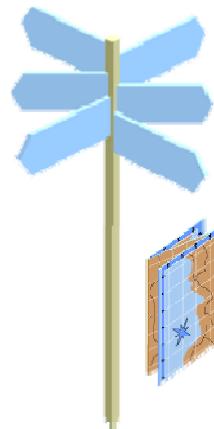


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Note: There are separate courses for other Oracle GoldenGate (OGG) products, such as Veridata and Management Pack. OGG could connect to MySQL, DB2, Hadoop, and so on. This course covers only OGG connected to Oracle 12c.

Roadmap

- Building Blocks of Oracle GoldenGate
 - Extract
 - Trail
 - Replicat
- Supported Platforms
- Product Line

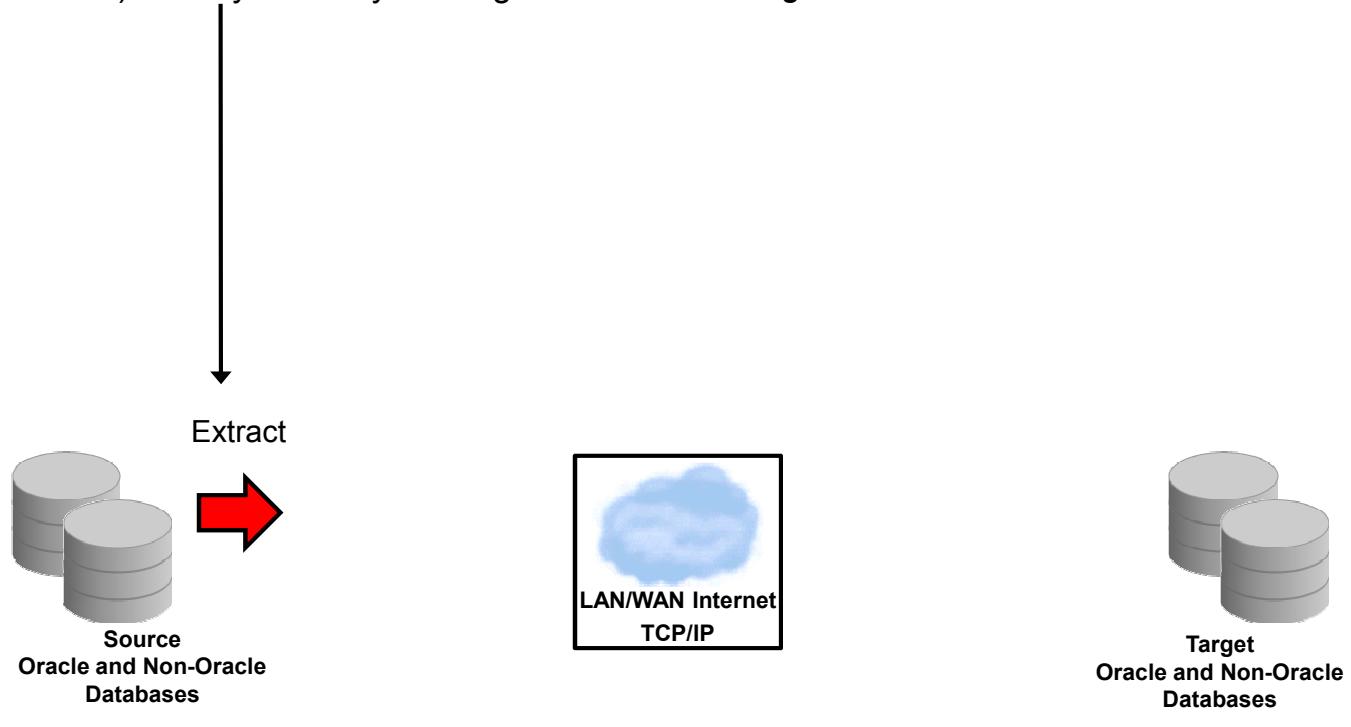


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate: Modular Building Blocks

Extract: Committed transactions are captured (and can be filtered) as they occur by reading the *transaction logs*.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Extract

Oracle GoldenGate can move data between heterogeneous databases for both the source and the target. The software operates at the database level, and the Capture component is typically installed on the source database server outside of the DBMS. Oracle GoldenGate's Extract process reads native transaction logs and captures transactions as soon as they commit, and takes the transactions outside of the database system to be queued. Oracle GoldenGate moves only changed, committed transactional data, allowing it to operate with extremely high performance and very low impact.

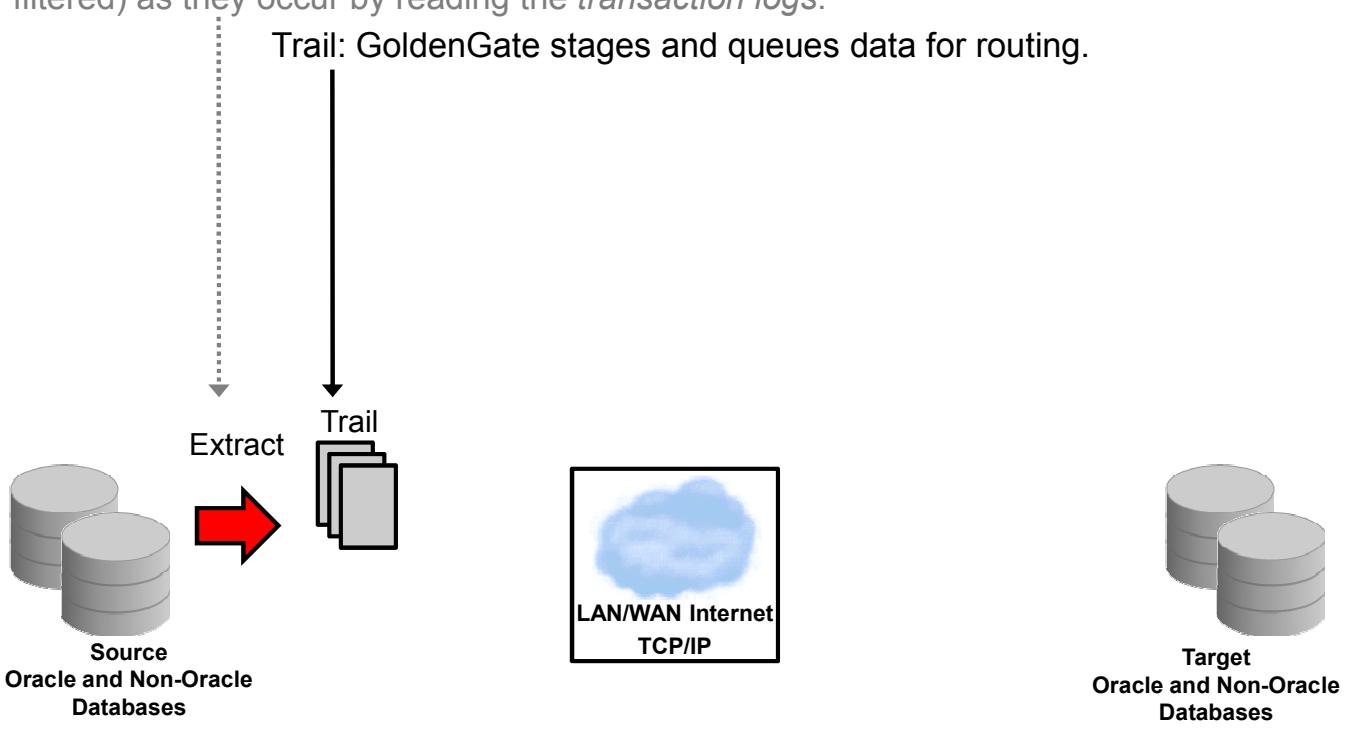
Oracle GoldenGate allows filtering at the schema, table, column, or row level. Row-level transformations can be applied either at this capture stage or later when delivering.

Oracle GoldenGate can do a "classic" Extract for any platform. However, if the platform is Oracle Database of a certain version, it can also do an optional "integrated" Extract (bypassing the logs) by using direct APIs, which is more efficient.

Oracle GoldenGate: Modular Building Blocks

Extract: Committed transactions are captured (and can be filtered) as they occur by reading the *transaction logs*.

Trail: GoldenGate stages and queues data for routing.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Trail

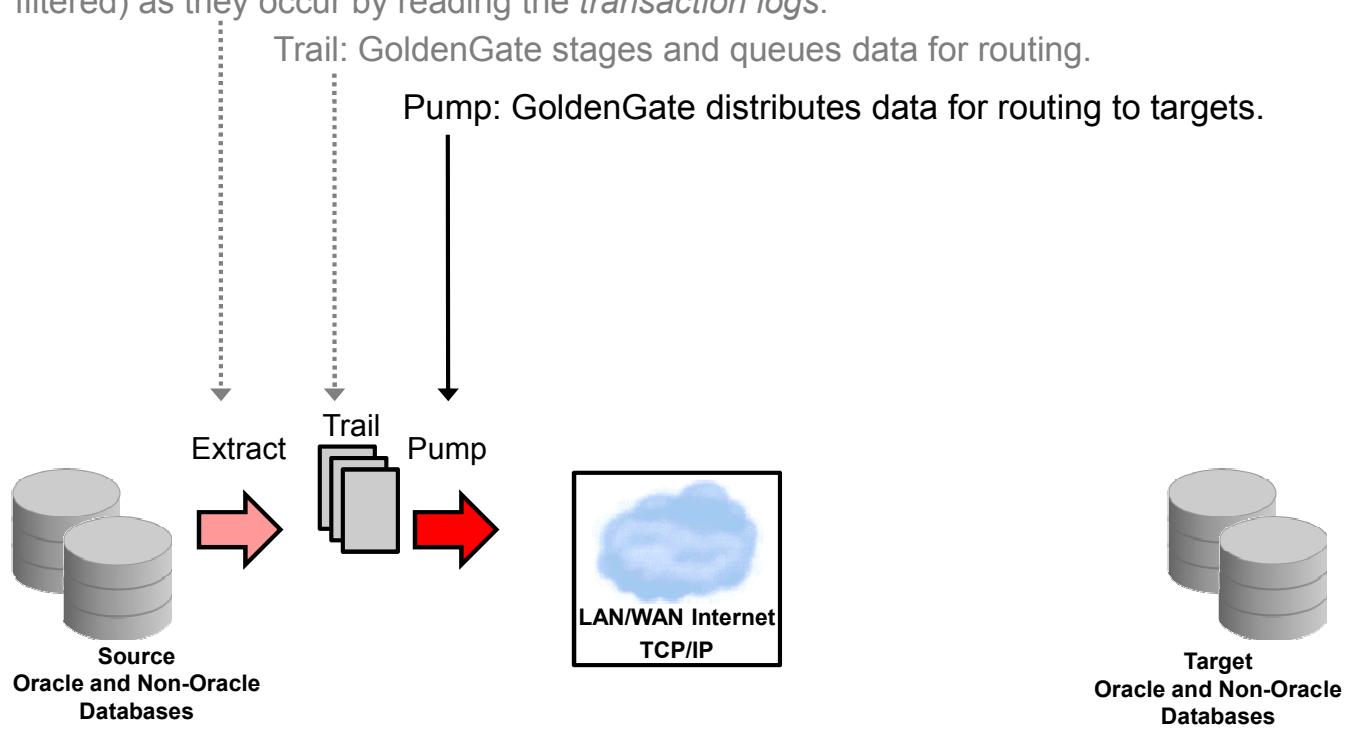
After the capture, Oracle GoldenGate converts the committed transactions into a canonical (universal) data format in “trail” files. Using source and target trail files, it ensures that data integrity is maintained—even if there is a system error or an outage.

Oracle GoldenGate: Modular Building Blocks

Extract: Committed transactions are captured (and can be filtered) as they occur by reading the *transaction logs*.

Trail: GoldenGate stages and queues data for routing.

Pump: GoldenGate distributes data for routing to targets.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Pump

Another Capture component can be used to pump the data to multiple targets and is used for better recoverability. This configuration enhances the fault tolerance and reliability of the overall GoldenGate environment. If there is a network failure (between the source and the target systems), the Oracle GoldenGate Capture component can continue to capture transactions, because the data can be queued up locally in the trail files on the source, enhancing the recoverability in case of database failures.

Note: This is *not* the database feature called Data Pump.

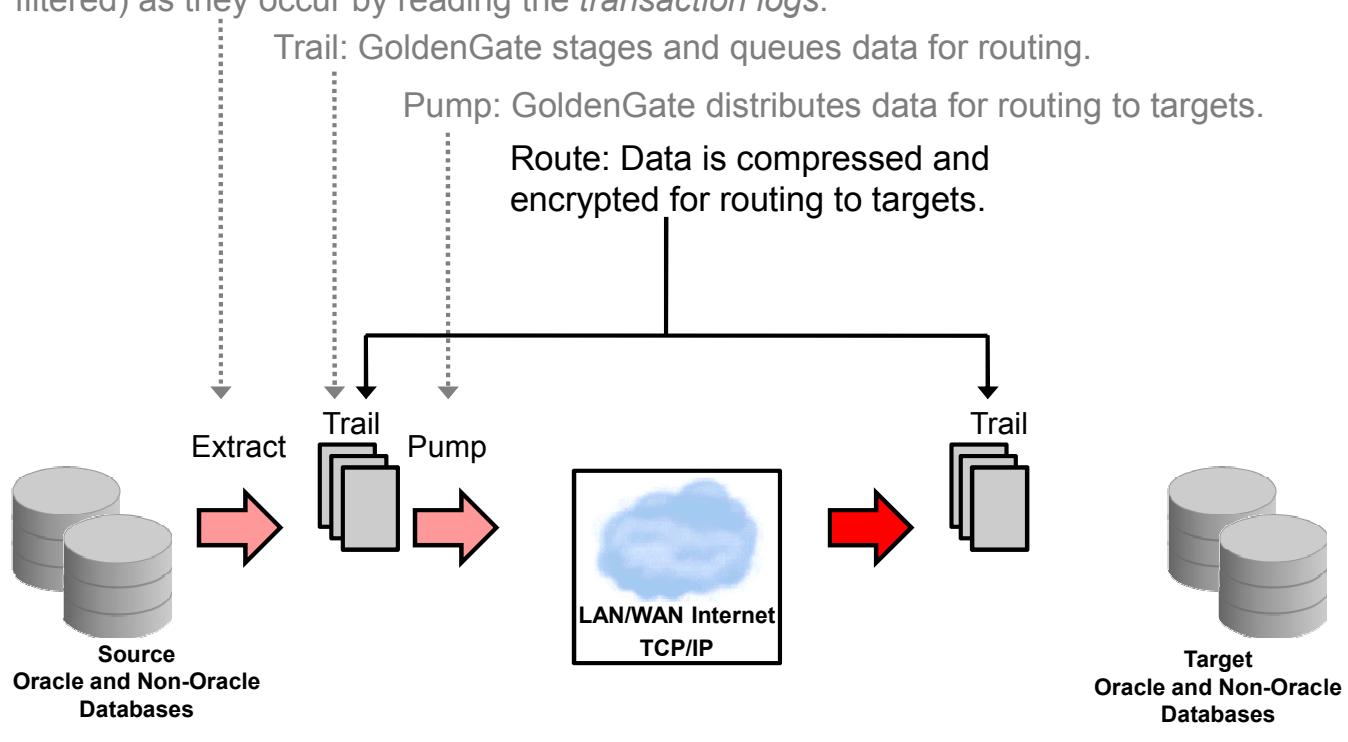
Oracle GoldenGate: Modular Building Blocks

Extract: Committed transactions are captured (and can be filtered) as they occur by reading the *transaction logs*.

Trail: GoldenGate stages and queues data for routing.

Pump: GoldenGate distributes data for routing to targets.

Route: Data is compressed and encrypted for routing to targets.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Route

The data is then routed from the trail files via TCP/IP (IPv4 or IPv6) to the target systems. During this routing process, data compression and encryption can be applied and thousands of transactions can be moved per second without distance limitations.

Oracle GoldenGate: Modular Building Blocks

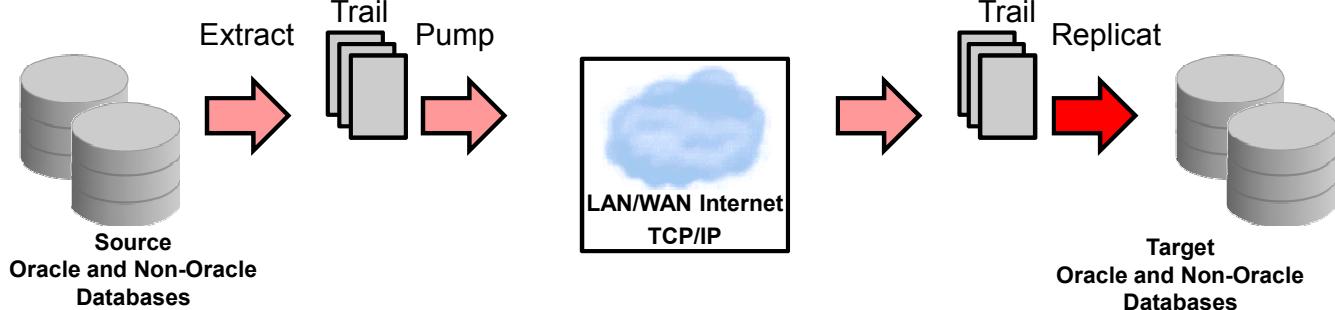
Extract: Committed transactions are captured (and can be filtered) as they occur by reading the *transaction logs*.

Trail: GoldenGate stages and queues data for routing.

Pump: GoldenGate distributes data for routing to targets.

Route: Data is compressed and encrypted for routing to targets.

Replicat: Replicat applies data with transaction integrity, transforming the data as required.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Replicat

The other key component of Oracle GoldenGate is the Replicat (or Delivery) module that is installed on the target database server. Queued transactions that are stored in the trail files are applied to the target by using native SQL calls. If necessary, basic transformations at the row level can be applied at either delivery or capture.

Oracle GoldenGate can do a “classic” Replicat (delivery) for any supported platform. However, if the platform is Oracle Database of a certain version, it can also do an optional “integrated” Replicat (bypassing the SQL apply) by using direct APIs, which is more efficient.

Oracle GoldenGate: Modular Building Blocks

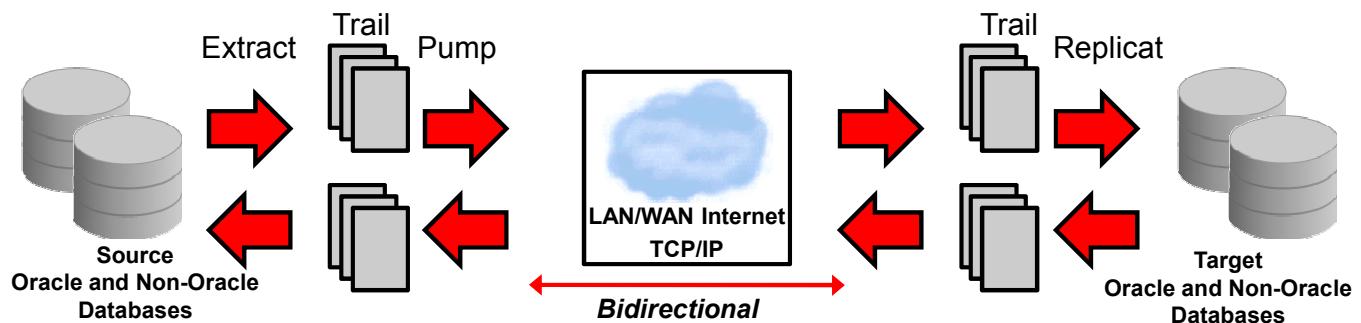
Extract: Committed transactions are captured (and can be filtered) as they occur by reading the *transaction logs*.

Trail: GoldenGate stages and queues data for routing

Pump: GoldenGate distributes data for routing to targets.

Route: Data is compressed and encrypted for routing to targets.

Replicat: Replicat applies data with transaction integrity, transforming the data as required



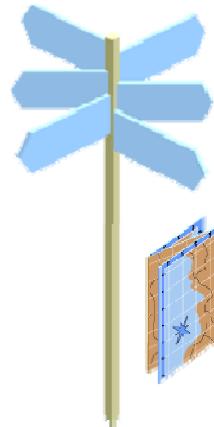
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Bidirectional Data Movement

This data movement can also be done bidirectionally. Oracle GoldenGate has built-in loop detection to differentiate between new transactions and data that is replicated.

Roadmap

- Building Blocks of Oracle GoldenGate
- Supported Platforms
 - Databases
 - Operating Systems
- Product Line

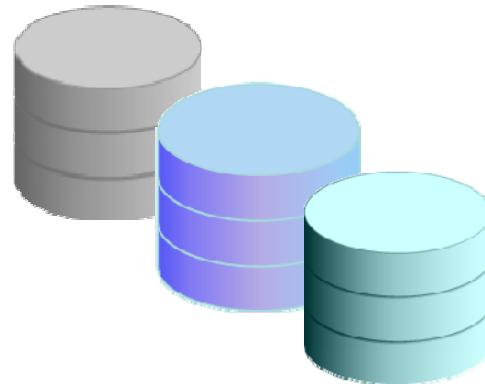


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Supported Databases

- Oracle GoldenGate
Capture:
 - Oracle
 - MySQL
 - IBM DB2
 - Microsoft SQL Server
 - Sybase ASE
 - Teradata
 - Enscribe
 - SQL/MP
 - SQL/MX
 - JMS message queues
- Oracle GoldenGate
Delivery:
All listed to the left, plus:
 - TimesTen
 - Flat File products



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate is ideal for heterogeneous environments—not just to support different versions of the same database or operation system/hardware, but to replicate and integrate data across vendor systems. Oracle GoldenGate supports log-based capture of changed data from nearly all major database vendors.

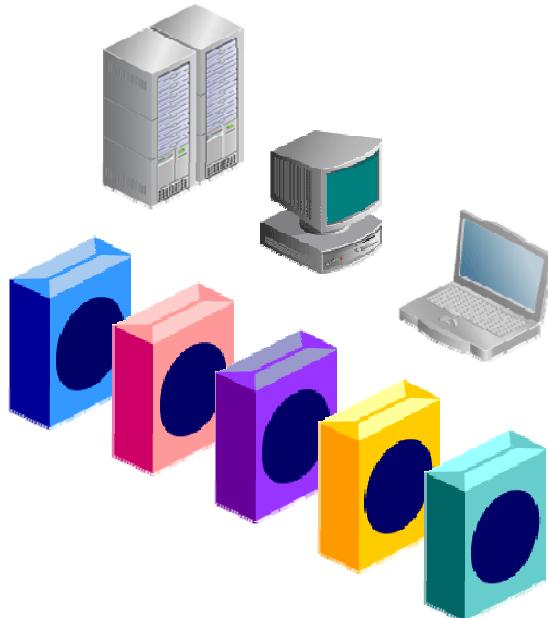
Replicating and delivering that data are also available to an even wider range of targets, including open source databases, several data warehouse appliances, ETL servers, and Java Message Service (JMS) message queues to support service-oriented architecture (SOA) and event-driven architecture (EDA).

Other systems are supported for delivery when using the Flat File Adapter or the Application Adapter for Java. For example, there is support for delivery to Netezza and Greenplum systems when using the Flat File Adapter.

Not all databases are supported on all versions of Oracle GoldenGate. As always, check the Oracle website (oracle.com) for the latest certification matrix.

Supported Operating Systems

- Linux
- Windows
- Oracle Solaris
- HP NonStop
- HP-UX
- HP OpenVMS
- IBM AIX
- IBM z/OS
- IBM iSeries
- z/linux



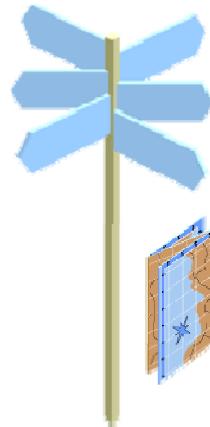
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Not all operating systems are supported on all versions of Oracle GoldenGate. As always, check the Oracle website for the latest certification matrix.

Roadmap

- Building Blocks of Oracle GoldenGate
- Supported Platforms
- Product Line
 - Oracle GoldenGate Veridata
 - Management Pack for GoldenGate
 - Director
 - Monitor
 - Oracle GoldenGate Application Adapters



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate Product Line

Product	Description
Oracle GoldenGate Veridata	Add-on capability to validate data in replicated systems
Management Pack for GoldenGate	Add-on Management Pack includes: <ul style="list-style-type: none"> • Oracle GoldenGate Director • Oracle GoldenGate Monitor
Oracle GoldenGate Application Adapters	Prepackaged application content, such as for Flat File and Java adapters



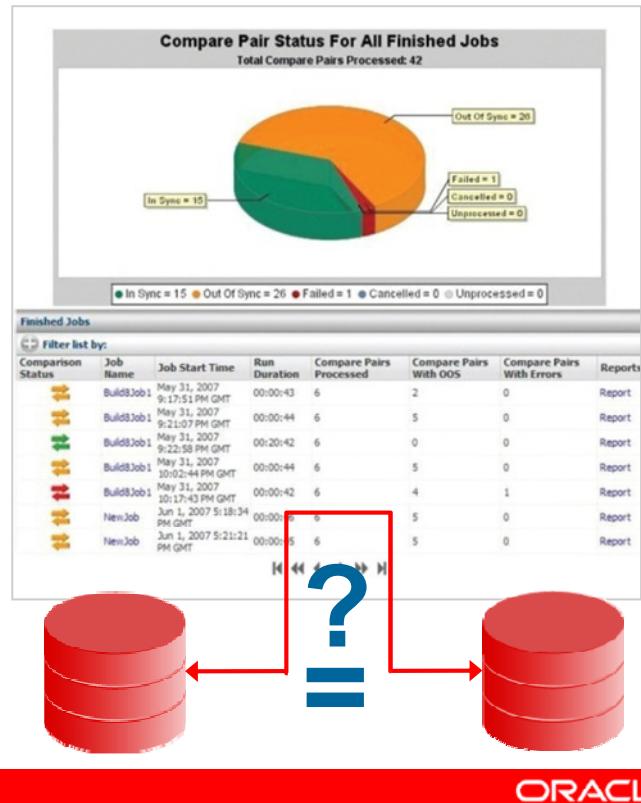
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Management Pack for Oracle GoldenGate is a server-based product that provides a graphical interface for designing, configuring, managing, monitoring, and reporting on the Oracle GoldenGate components that are implemented across a business organization. It includes Oracle GoldenGate Director (for creating solutions graphically) and Oracle GoldenGate Monitor (for monitoring Oracle GoldenGate objects and sending alerts).

Oracle GoldenGate Veridata

Comparisons:

- High-speed
- Low-impact
- Non-disruptive

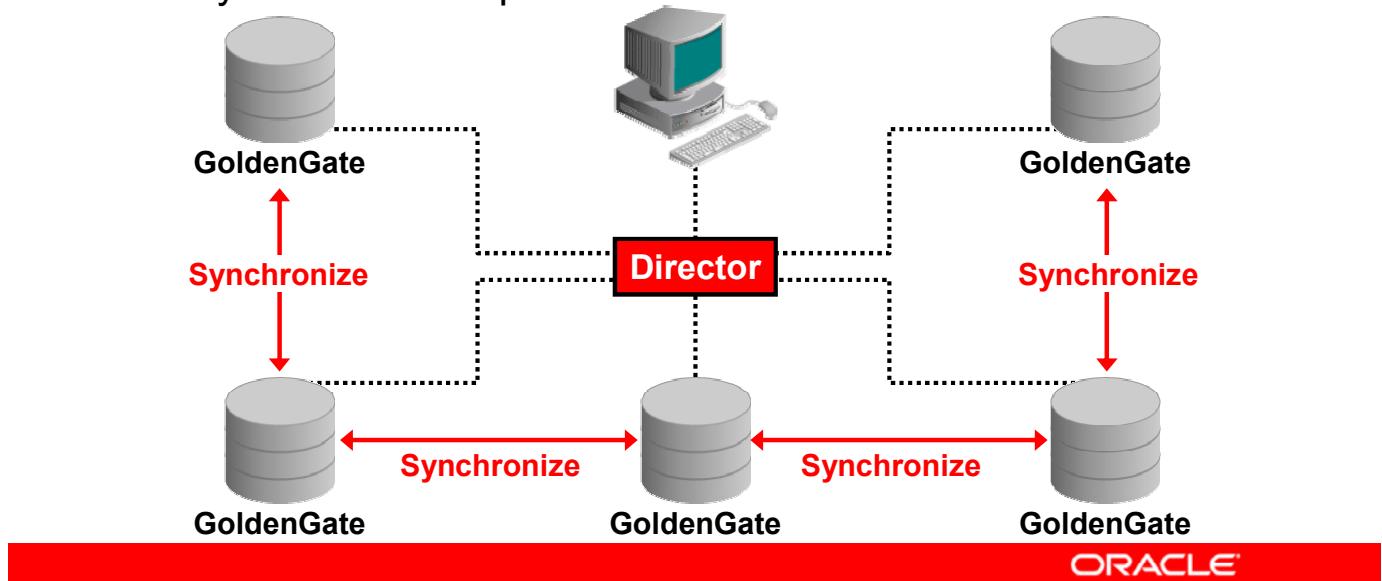


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate Veridata performs high-speed, low-impact data comparisons between homogeneous and heterogeneous databases. It identifies and reports data discrepancies between heterogeneous databases without interrupting their availability or the business processes that they support. One typical use is to ensure that there are no data discrepancies in standby systems or in new environments (that users may be migrating to) before a switchover. By comparing data sets and ensuring that they are consistent, Veridata enables companies to decrease their exposure to risk—especially for data that is used for regulatory compliance.

Oracle GoldenGate Director: Overview

Oracle GoldenGate Director is a graphical enterprise application that offers a visual way to define, configure, manage, and report on all Oracle GoldenGate transactional data synchronization processes.



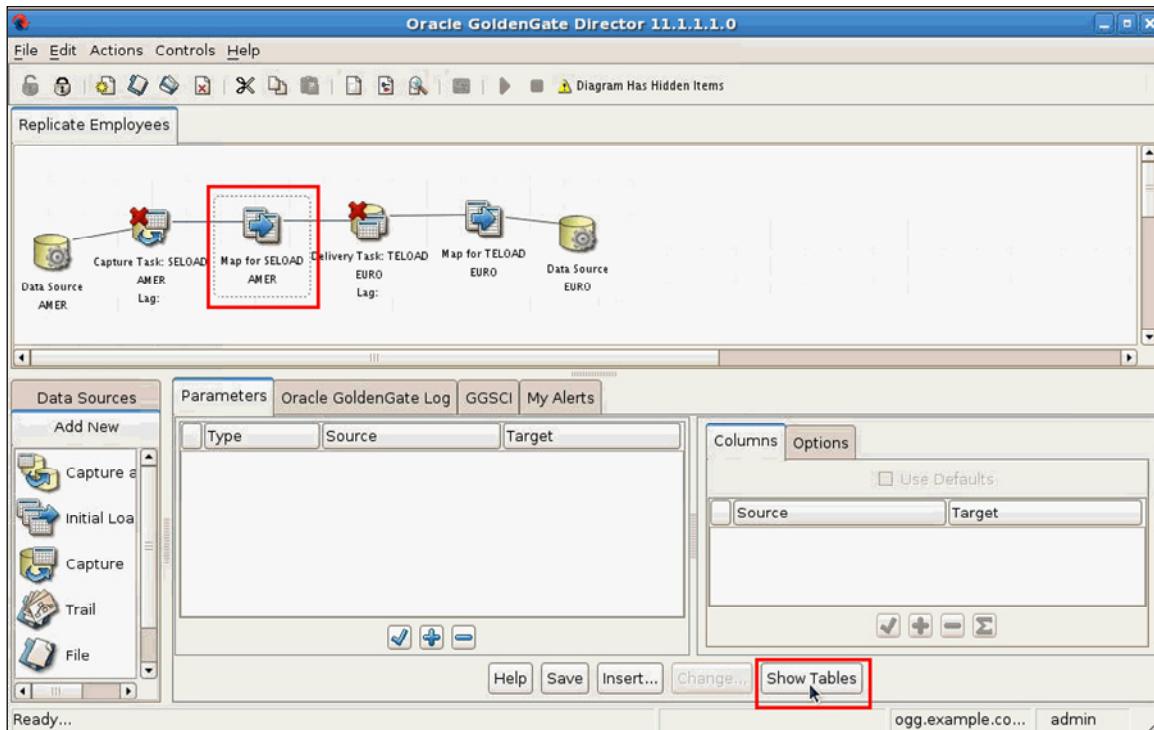
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate Director is a multitiered graphical enterprise application that provides a visual way to define, configure, manage, and report on all Oracle GoldenGate processes.

Oracle GoldenGate Director provides:

- A client application that can be used remotely
- A web interface that is accessible through most browsers

Oracle GoldenGate Director

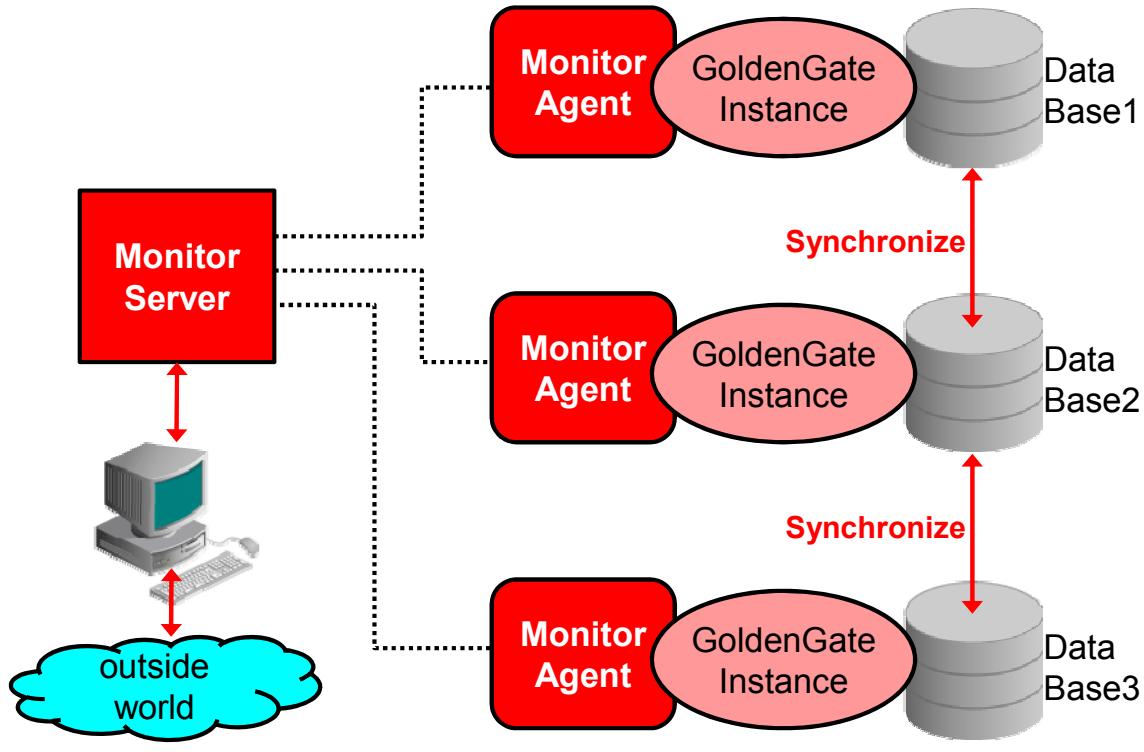


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate Director is one half of the separately licensed Management Pack. It can perform configuration as well as monitoring and reporting. By clicking an individual icon, you can view parameters, logs, and reports for that item. Note that Director refers to Capture instead of Extract.

Oracle GoldenGate Monitor: Overview



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate Monitor is the other half of the Management Pack. Monitor is a tool that displays on a web browser the configuration, statistics, history, and alerts of Oracle GoldenGate instances and processes. The Oracle GoldenGate instances must be at least version 11.1.1.1.1 (pronounced “11-quad-one”); Oracle GoldenGate Monitor version 11.1.1.1 (pronounced “11-triple-one”) cannot read back-level instances.

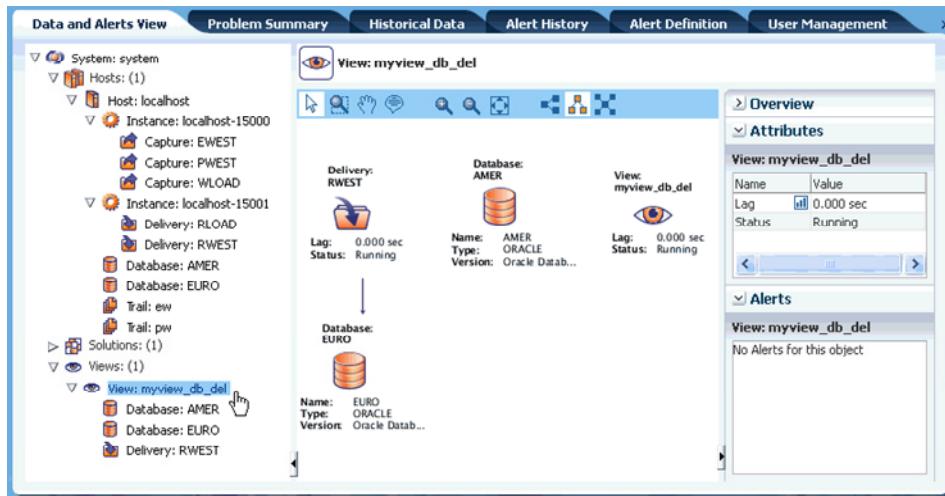
Unlike Oracle GoldenGate Director, Oracle GoldenGate Monitor cannot make any changes to the configurations of instances. However, Oracle GoldenGate Monitor provides excellent means of notifying external interested parties of alerts using either email or SNMP traps, or internal notification by running another program locally using a command-line interface.

The synchronization between databases is at least unidirectional, and may or may not be bidirectional.

Many of the pieces of Monitor (such as the `dirbdb`, `dirwww`, `jagent`, and `jdk` directories) now ship with the base Oracle GoldenGate product 12.1.2 and later.

Oracle GoldenGate Monitor

- Oracle GoldenGate Monitor can view process status, statistics, and events.



- The status of the Manager, Extract, and Replicat processes can also be viewed through various files.

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The external monitoring software options require the `GLOBALS` file to have the `EnableMonitoring` parameter.

As a subsystem becomes active or inactive, the icon and icon color change. Oracle GoldenGate Monitor is available in the Oracle Management Pack for Oracle GoldenGate. It is covered in the *Oracle GoldenGate 11g Management Pack: Overview* course (course code: D73248GC10).

These processes can also be viewed through various commands and files:

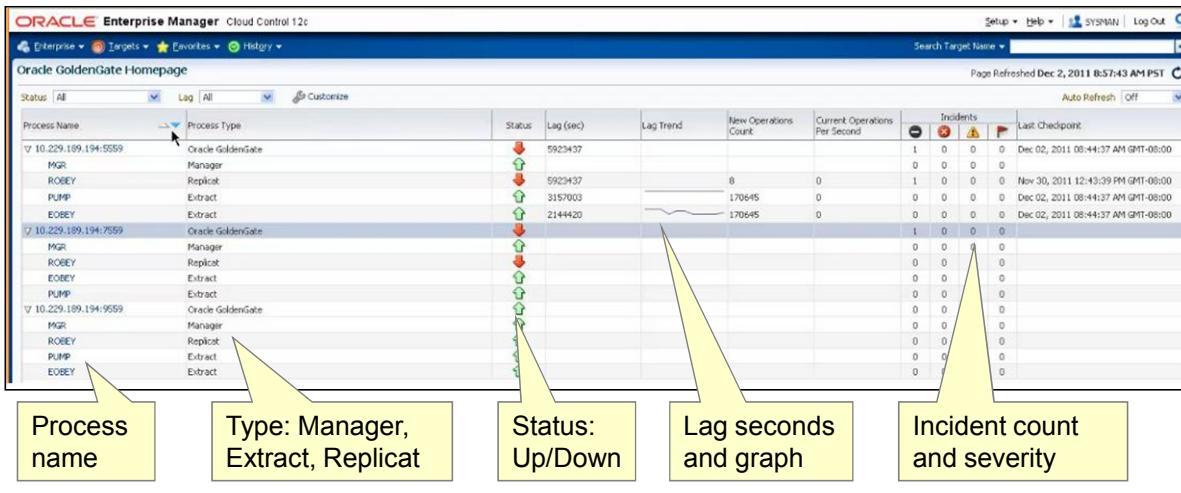
- GGSCI information commands
- The `ggserr.log` file (known as the *error log*)
- Process reports (`dirrpt/*.rpt`)
- The discard file (`dirrpt/*.dsc`)
- The Event Viewer on Windows systems or the `syslog` on UNIX systems to view errors at the operating-system level

The discard file can be located anywhere that you want. But, by convention, it is in the same folder as the other reports.

Enterprise Manager Cloud Control 12c

Enterprise Manager:

- Requires plug-ins and agents
- Auto-discovers targets
- Monitors (rather than configures) GoldenGate



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

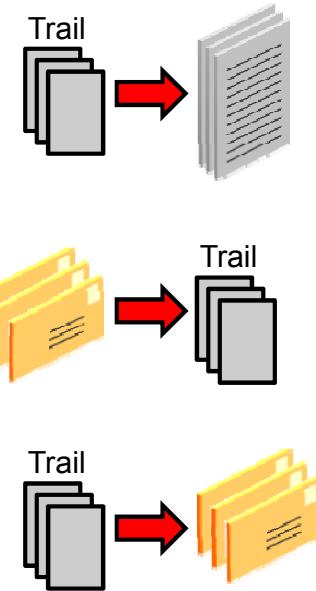
Although it is not technically part of the Oracle GoldenGate product line, Oracle Enterprise Manager can be used to manage Oracle GoldenGate. This slide shows three different instances of Oracle GoldenGate being monitored by Oracle Enterprise Manager 12c. Each installation is uniquely identified by a host IP address and port combination. *Port number* represents the port number of the Oracle GoldenGate Manager process.

Oracle Enterprise Manager has its own training curriculum.

Adapter Integration Options for Oracle GoldenGate

Oracle GoldenGate adapters integrate with installations of the core product to:

- Read an Oracle GoldenGate trail and write transactions to a flat file that can be used by other applications
- Read JMS messages and deliver them as an Oracle GoldenGate trail
- Read an Oracle GoldenGate trail and deliver transactions to a JMS provider or to another messaging system or custom application

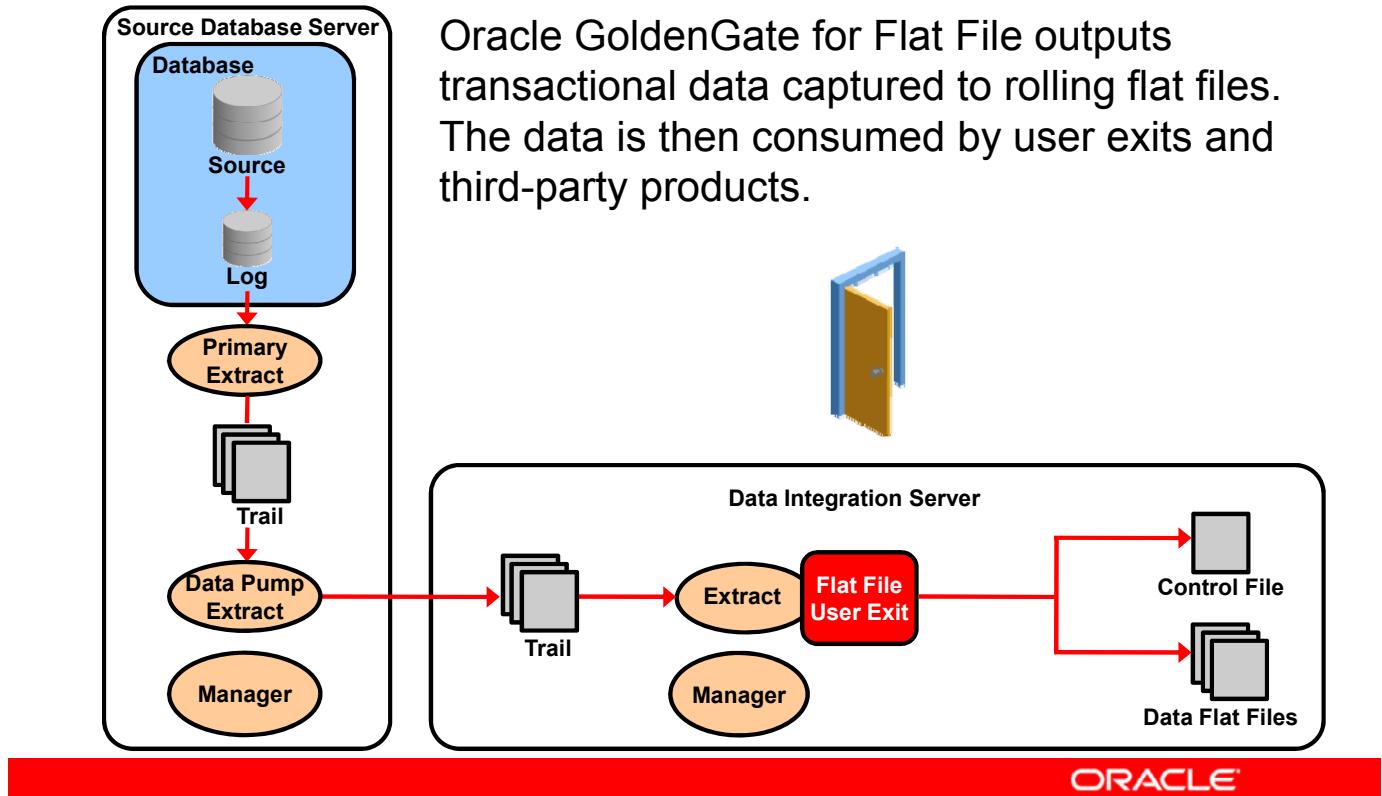


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The adapters can also be used with other Oracle SOA Suite components.

Each of the adapters listed in the slide is discussed in detail in the following slides.

Oracle GoldenGate for Flat File



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

ORACLE

Oracle GoldenGate for Flat File is used to output transactional data captured by Oracle GoldenGate to rolling flat files (a series of sequential flat files) to be consumed by a third-party product. Oracle GoldenGate for Flat File is implemented as a *user exit* provided as a shared library (.so or .dll) that integrates into the Oracle GoldenGate Extract process. The user exit supports two modes of output:

- **DSV:** Delimiter Separated Values (for example, tabs or commas)
- **LDV:** Length Delimited Values

It can output data in the following ways:

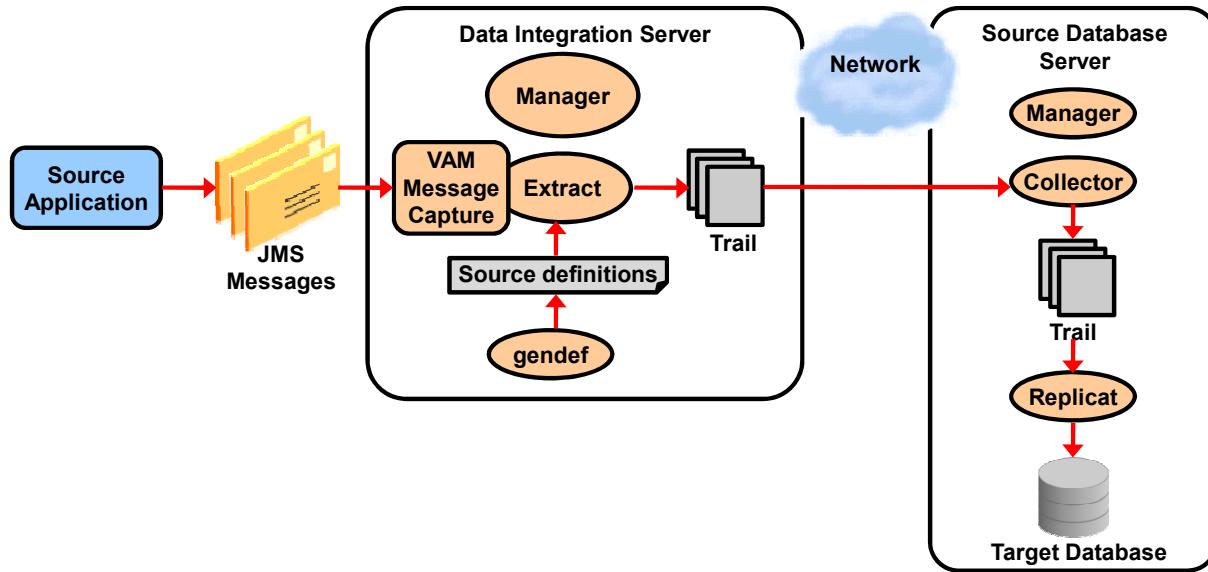
- All to one file
- One file per table
- One file per operation code

The Flat File User Exit is used with properties files. There are properties files for Siebel, Abinitio, Greenplum, Netezza, and comma-delimited formats.

User exits in general are covered in the lesson titled “Additional Transformation Topics.”

Oracle GoldenGate Application Adapter for Java

The Oracle GoldenGate Java API delivers transactional data to targets such as a JMS, writing to disk or integrating with custom applications.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Through the Oracle GoldenGate Java API, transactional data captured by Oracle GoldenGate can be delivered to targets other than a relational database (such as a JMS), writing files to disk or integrating with a custom application's Java API.

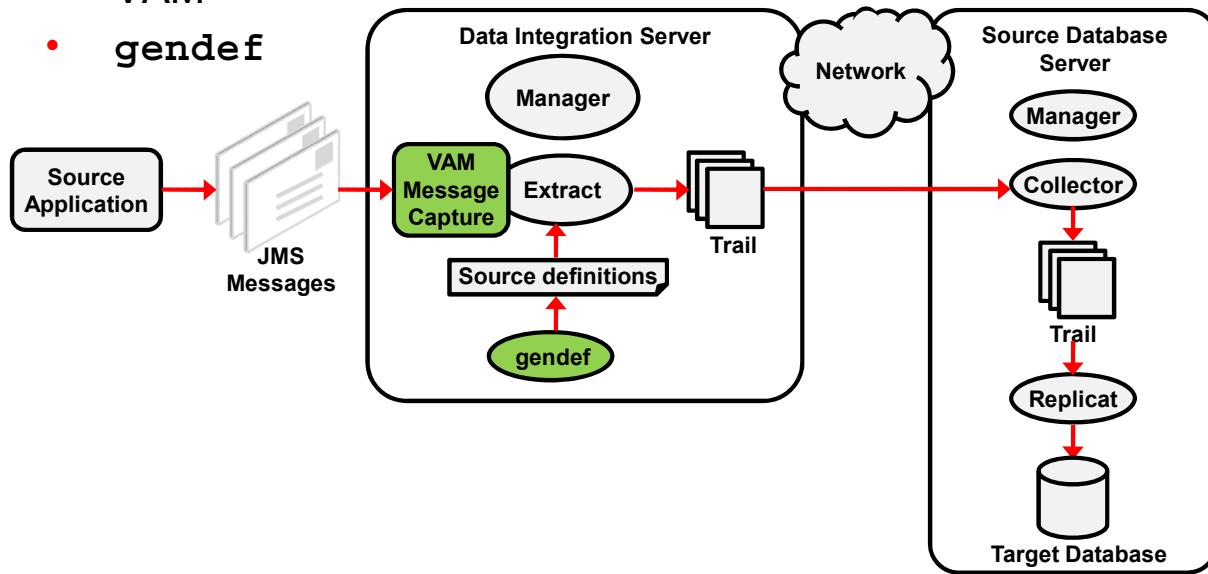
The Oracle GoldenGate messaging capture adapter connects to JMS messaging to parse messages and send them through a Vendor Access Module (VAM) interface to an Oracle GoldenGate Extract that builds an Oracle GoldenGate trail of message data. This enables JMS messages to be delivered to an Oracle GoldenGate system that is running for a target database.

Application Adapter for Java supports Oracle Coherence HotCache, but Coherence 12.1.2 works only with Oracle GoldenGate Core 11.2.1 (not 12c) at the time this course was published.

JMS Messaging Capture

Oracle GoldenGate JMS message capture requires two additional components:

- VAM
- **gendef**



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate JMS message capture requires two components:

- The dynamically linked shared Vendor Access Module (VAM) library that is attached to the Oracle GoldenGate Extract process
- A separate utility, **gendef**, that uses the message-capture properties file and parser-specific data definitions to generate (create) the source definitions file

Oracle GoldenGate for Java includes a **gendef** utility (not to be confused with the **defgen** column utility) that generates an Oracle GoldenGate source definitions file from the properties defined in a properties file. It creates a normalized definition of tables based on the property settings and other parser-specific data definition values.

Quiz

Which three statements are true?

- a. The Extract process can capture committed changes from the transaction logs and write them to a trail file.
- b. The Data Pump process can capture committed changes from the transaction logs and write them to a trail file.
- c. The Data Pump process can move data from a local trail file to a remote trail file.
- d. The Replicat process can move changes from a local trail file to the database.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: a, c, d

Quiz

Which three products make up the Oracle GoldenGate product line?

- a. Oracle GoldenGate Veridata
- b. Management Pack for Oracle GoldenGate
- c. Oracle GoldenGate Application Adapters (that is, Flat File and Java adapters)
- d. Enterprise Manager
- e. Oracle Streams



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: a, b, c

These are all separately purchased products.

Summary

In this lesson, you should have learned how to:

- List the building blocks that make up Oracle GoldenGate functionality
- List the supported platforms and infrastructure
- Describe the Oracle GoldenGate product line
 - Oracle GoldenGate Veridata
 - Management Pack for Oracle GoldenGate
 - Oracle GoldenGate for Flat File
 - Oracle GoldenGate for Java



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

There is no practice for Lesson 2.

Oracle GoldenGate Architecture

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

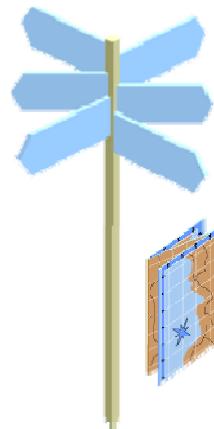
- Describe the Oracle GoldenGate logical architecture and components
- Describe Oracle GoldenGate process groups
- Describe Oracle GoldenGate files
- Explain change capture and delivery
- Explain initial data load
- Compare batch and online operations
- Explain Oracle GoldenGate checkpointing
- Describe the commit sequence number (CSN)



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Roadmap

- Architecture
 - Components
 - Process Groups
 - Files
- Extracts
- Initial Loads
- Checkpoints

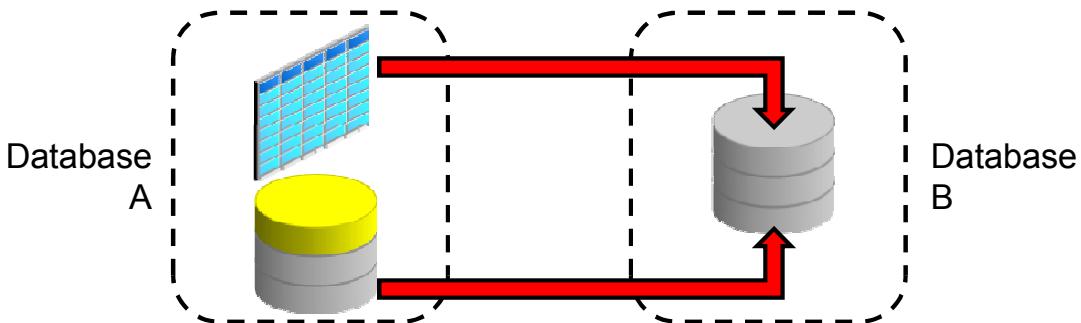


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Uses of Oracle GoldenGate

- **Primary use:** Change data capture and delivery from database transaction logs
- **Optional use:** Initial load directly from database tables
 - This is useful for synchronizing heterogeneous databases.
 - Database-specific methods may be preferable for homogeneous configurations.



ORACLE®

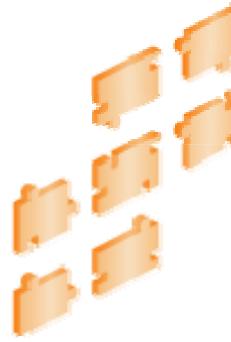
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The exception to the “database-specific” generalization is for Oracle-to-Oracle homogenous environments. In those cases, Oracle GoldenGate is the recommended method of synchronization.

“Transaction logs” in an Oracle database context are the REDO logs. (You should not confuse these with the undo logs.)

Oracle GoldenGate Components

- Manager
- Collector
- Extract
- Data Pump
- Replicat
- Trails or extract files
- Checkpoints
- Wallet



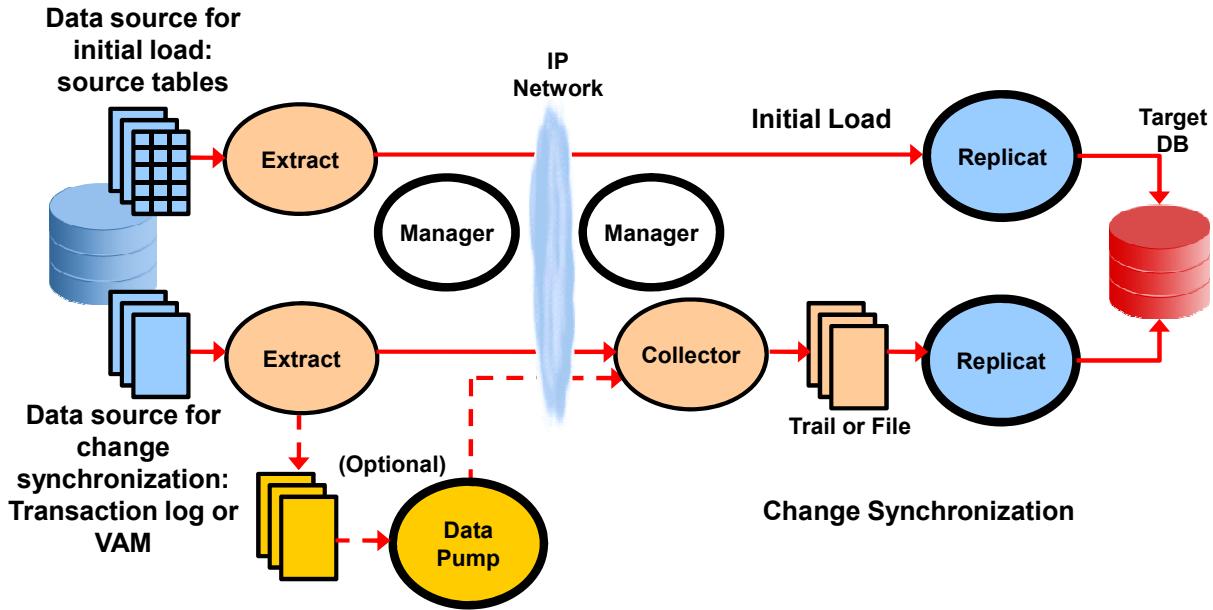
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Each of these components is covered in detail in the following slides and in later lessons in this course.

- **Manager:** Is required to start and stop the other processes, but is not required for the ongoing operation of another process. That is, you can start the Manager, use the Manager to start the Extract, and then stop and restart the Manager (perhaps to pick up new Manager configuration parameters) without affecting the running Extract.
- **Server Collector:** Also known simply as *Collector*
- **Extract:** Also known as *Change Data Capture*
- **Data Pump:** Optional; highly recommended
- **Replicat**
- **Trails or extract files:** Can optionally be compressed and/or encrypted
- **Checkpoints:** Optional; highly recommended
- **Wallet:** Used for secure common storage of user IDs and passwords; optional and highly recommended

Oracle GoldenGate Logical Architecture



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

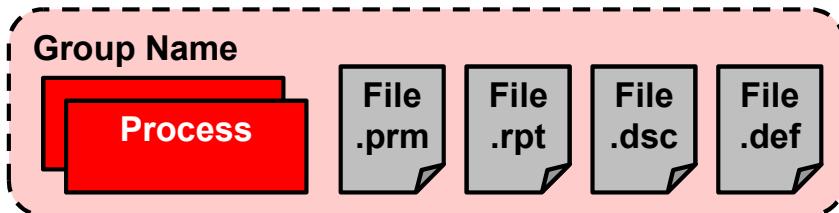
The diagram in the slide illustrates the logical architecture of Oracle GoldenGate for initial data loads and for the replication of ongoing database changes. This is the basic configuration. Variations of this model are recommended depending on business needs.

Vendor Access Module (VAM) is usually used only by non-Oracle databases, such as Teradata.

In Oracle GoldenGate version 11.2.1 and later, all network calls use IPv6-based APIs, which support either IPv4 or IPv6 networks as appropriate.

Oracle GoldenGate Process Groups

- A process *group* consists of the following:
 - Process (either Extract or Replicat)
 - Parameter file
 - Checkpoint file
 - Any other files associated with the process
- Groups can be defined by using the Add Extract and Add Replicat commands.
- Each process group must have a unique name.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To differentiate among multiple Extract or Replicat processes on a system, you define processing groups. For example, to replicate different sets of data in parallel, you create two Replicat groups.

A processing group consists of a process (either Extract or Replicat), its parameter file (*.prm), its checkpoint file, and any other files associated with the process, such as Report (*.rpt), Discard (*.dsc), and Column Definition (*.def) ASCII files. For Replicat, a group also includes a checkpoint table (if one is being used). You define groups by using the Add Extract and Add Replicat commands in the GoldenGate Software Command Interface (GGSCI).

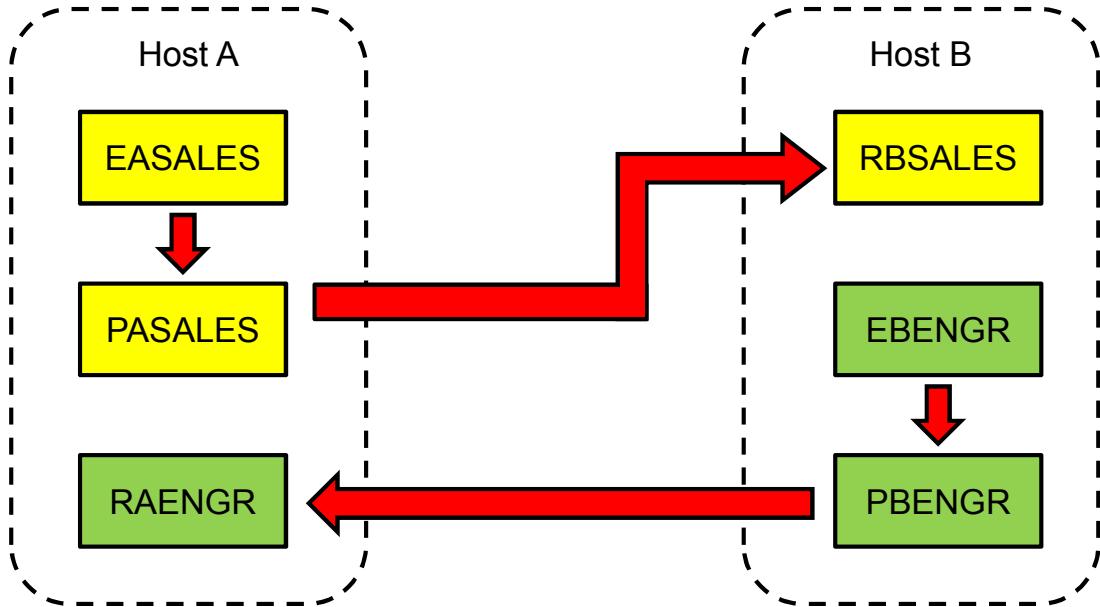
In version 11.2.1 and later, the Oracle GoldenGate GGSCI command interface fully supports up to 5,000 concurrent Extract and Replicat processes for each instance of Oracle GoldenGate Manager.

In version 12.1, the discard file creation is now automatic by default; in previous versions, you had to define the discard file manually.

In 12c, an integrated process causes additional processes to be created in the database.

Process-Group Naming Conventions

You can use any convention you choose. Here is an example:



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can use any OS-supported naming convention you want for the process groups and their associated parameter files (covered in later slides). For example, suppose that you have host A and host B, and you have the engineering and sales departments on both hosts. You may want to use a naming convention such as *P**H**t**a**s**k*, where:

P is the process type:

- E=Extract
- P=Pump
- R=Replicat
- I=Initial Load

H is the host:

- A=left, West
- B=right, East

task is the name of the task, department, or schema:

- SALES
- ENGR
- whatever

When naming groups, you can use up to eight characters, including non-alphanumeric characters such as the underscore (_). Any character can be used, as long as the character set of the local operating system supports it and the operating system allows that character to be in a file name. This is because a group is identified by its associated checkpoint file.

The following characters are not allowed in a group name: { \ / : * ? " < > | }.

On HP UX, Linux, and Solaris, it is possible to create a file name with a colon (:) or an asterisk (*), although it is not recommended.

In general, group names are not case-sensitive. For example, `finance`, `Finance`, and `FINANCE` are all considered to be the same. However, in Linux, the group name (and its parameter file name if explicitly defined in the `ADD` command) must be all **UPPERCASE** or all lowercase. Mixed-case group names and parameter file names will result in errors when starting the process.

Use only one word. Do not use the word `port` as a group name. However, you can use the string `port` as part of the group name. You can include a number in a group name. However, be aware that using a numeric value at the end of a group name (such as `fin10`) can cause duplicate report file names and errors, because the writing process appends a number to the end of the group name when generating a report. You can place a numeric value at the beginning of a group name, such as `10_fin`, `10fin`, and so forth.

GGSCI

- Processes are added and started using the Oracle GoldenGate Software Command Interface (GGSCI) with the group name.
- The GGSCI is used to issue a complete range of commands that configure, control, and monitor Oracle GoldenGate.
- Commands can be typed or run from a script.
- The script can include OS shell commands.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

GGSCI commands can be collected together in an ASCII text script called an Obey (.oby) file. The extension .oby is by convention; it can be any extension. You create and maintain these scripts with any text editor, such as Notepad or gedit or vi.

There are GUI interfaces as well, but they are part of the Oracle GoldenGate Management Pack (a separate product that is covered in the course titled *Oracle GoldenGate 11g Management Pack: Overview*).

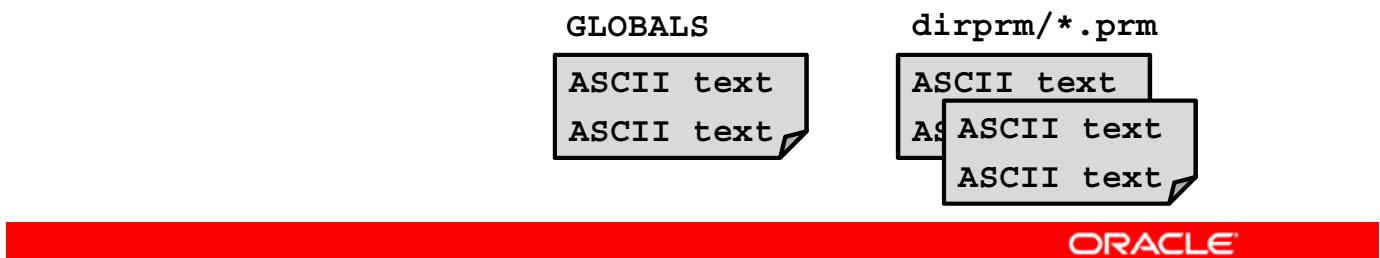
There is no formal collection of commands to run at startup, but you can create a startup.oby file (this name is a convention; you can name the file anything you like) with useful environment and setup commands in it. You would run it first by entering:

```
GGSCI 1> Obey startup.oby
```

Similarly, you may want to make a shutdown.oby file with cleanup commands in it. The last command in it would be Exit.

Parameter Files

- Most Oracle GoldenGate functionality is controlled by the use of parameters that are specified in text files.
- Oracle GoldenGate uses two types of parameter files:
 - **GLOBALS file:** Stores parameters that relate to the Oracle GoldenGate instance as a whole
 - **Run-time parameter file:** Is coupled with a specific process (such as Extract)
- By default, parameter files are in `dirprm` in the Oracle GoldenGate directory.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

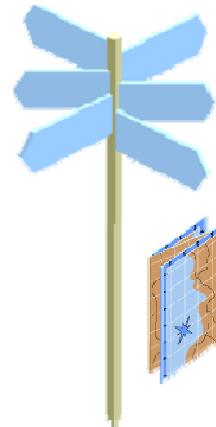
The **GLOBALS** file (must be uppercase with no extension) stores parameters that relate to the Oracle GoldenGate instance as a whole. This is in contrast to runtime parameters, which are coupled with a specific process such as Extract. The parameters in the **GLOBALS** file apply to all processes in the Oracle GoldenGate instance, but can be overridden by specific process parameters. After the **GLOBALS** parameters are set, they are rarely changed, and there are far fewer of them than runtime parameters.

A **GLOBALS** parameter file is required only in certain circumstances and, when used, must be created from the command shell before starting any Oracle GoldenGate processes, including GGSCI. The GGSCI program reads the **GLOBALS** file and passes the parameters to processes that need them.

Run-time parameters (extension `.prm`) give you control over the various aspects of Oracle GoldenGate synchronization, such as data selection and mapping, DDL and sequence selection, error resolution, logging, and so on. There can be only one active parameter file for the Manager process or an Extract or Replicat group.

Roadmap

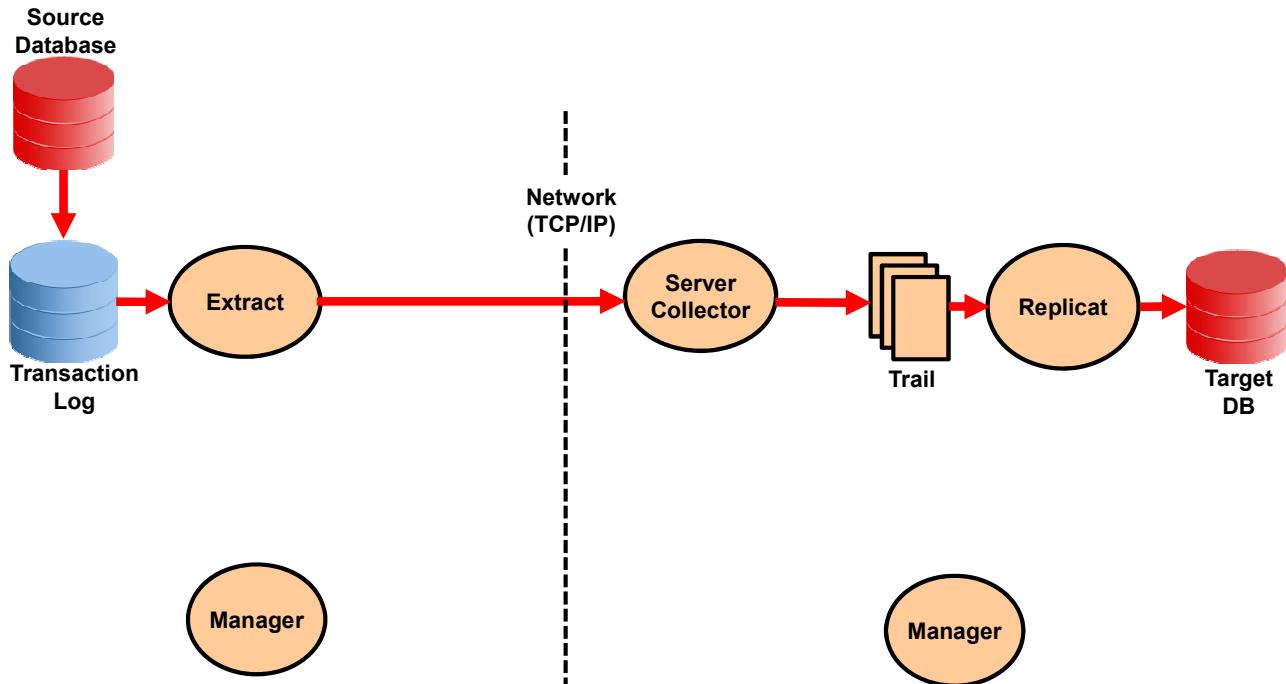
- Architecture
- Extracts
 - Classic
 - Integrated
- Initial Loads
- Checkpoints



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Change Data Capture (Extract) and Delivery



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

By default, Oracle GoldenGate 10.4 and later have one or more *dynamic* Server Collectors. With a dynamic Server Collector, there is a one-to-one relationship between the Extract and the Server Collector. Earlier releases used a default *static* Server Collector, and there was a many-to-one relationship between the Extracts and the Server Collector.

On the source system:

1. An Extract process captures transactional changes from transaction logs.
2. The Extract process sends data across a TCP/IP network to the target system.

On the target system:

1. A Server Collector process reassembles and writes the data to an Oracle GoldenGate trail.
2. A Replicat process reads the trail and applies it to the target database. (This can be concurrent with data capture or performed later.)

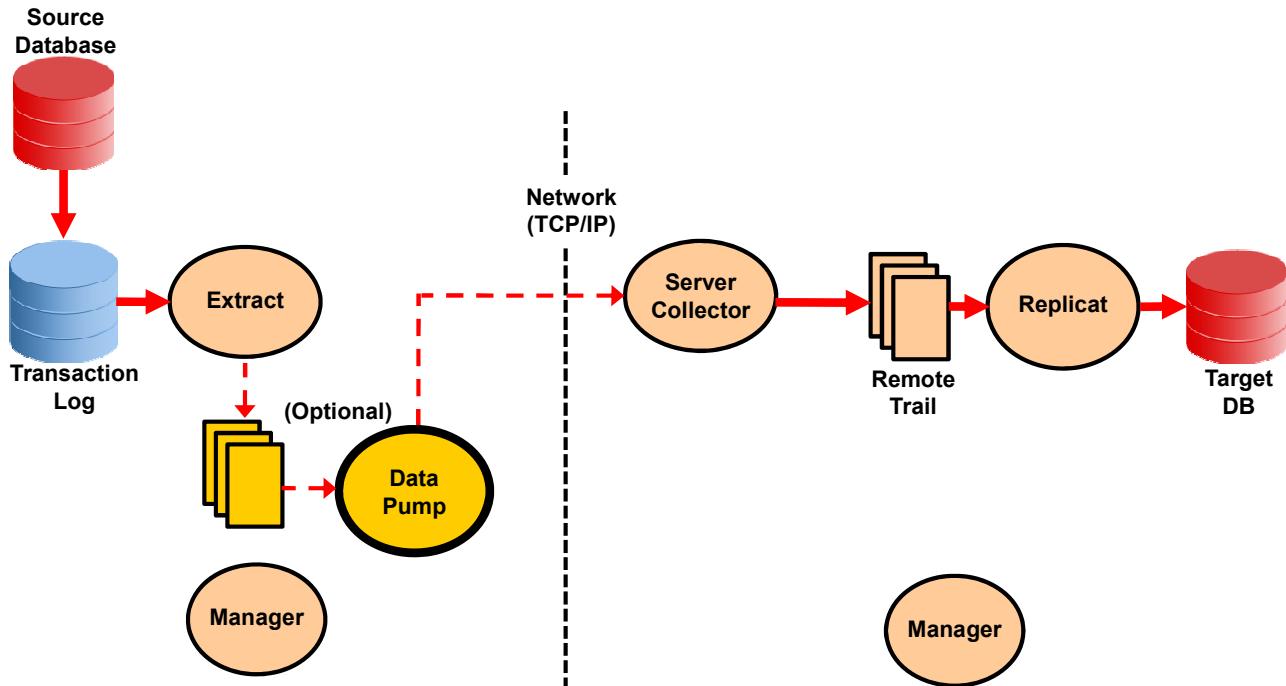
Manager processes on both systems control activities such as starting, monitoring, and restarting processes; allocating data storage; and reporting errors and events.

Real Application Clusters (RAC)

In an Oracle RAC configuration with Oracle GoldenGate, the Extract (capture) spawns an extract reader thread for every node on the source database. The coordinator thread receives, interprets, and orders redo from all readers. The `THREADS n` option is required in classic capture mode for Oracle RAC to specify the number of redo log threads being used by the cluster. Extract reads and coordinates each thread to maintain transactional consistency. Manual thread maintenance is not required for integrated capture mode.

Only one OGG RAC extract node copy is active at a time; the other nodes are in standby.

Change Data Capture (Extract) and Delivery Using a Data Pump



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Although the data pump Extract is optional, using one is considered to be a common best practice. The data pump Extract works as follows:

- On the source system:
 1. An Extract process captures transactional changes from the database transaction log.
 2. The Extract process writes the data to a local Oracle GoldenGate trail. (This preserves the captured data if the network or target trail fails.)
 3. A second Extract process (called a *data pump*) sends the data across the network to the target system.
- On the target system:
 1. A Server Collector process reassembles and writes the data to an Oracle GoldenGate trail.
 2. A Replicat process reads the trail and applies it to the target database. (This can be concurrent with data capture or performed later.)

Manager processes on both systems control activities such as starting, monitoring, and restarting processes; allocating data storage; and reporting errors and events.

Extract Flavors

Integrated Extract:

- Is an Oracle GoldenGate Extract for Oracle databases
- Is multithreaded
- Relies on Oracle's internal log parsing and processing implementation
- Supports more data types
- Supports downstream topologies
- Is available with OGG version 11.2.1 and later

Classic Extract:

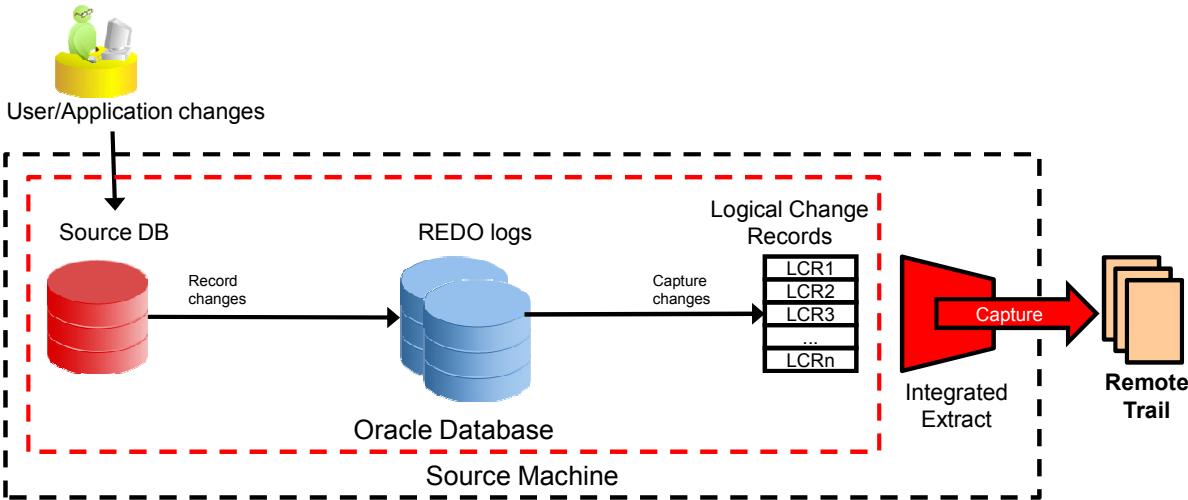
- Is traditional REDO log-based extract for Oracle
- Works for all supported DB platforms and versions



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The location of the REDO logs is flexible with some of the newest Oracle database versions. The stated direction is that new features will come out for the Integrated mode, whereas the Classic mode will continue as is. This course assumes the Classic Extract mode for all practices. Integrated Extract is covered in more detail in the advanced Oracle GoldenGate courses.

Distributed Topologies for Integrated Extract: Same Machine



Supports:

- Exadata
- Compression
- IOT, XML, LOBs natively

Requires:

- Database 10.2 or later

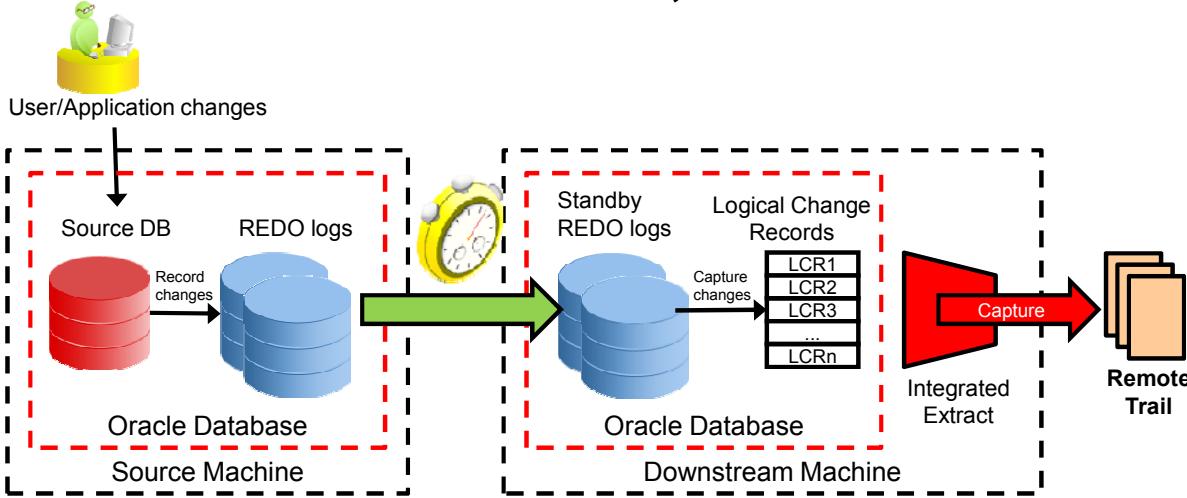
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In the typical configuration, Oracle GoldenGate is installed on the same machine or host as Oracle Database. This is similar to the REDO-based Classic Extract configuration. Oracle's REDO parsing and processing system captures the database changes in the form of Logical Change Records (LCRs). Later this LCR data is read by the consumer thread of Integrated Extract and processed by the producer thread of the same Integrated Extract. After processing, the records will be written to trail files.

This real-time configuration is suitable for deployments in which customers do not mind keeping the Oracle GoldenGate Extract process running on the same machine as the Oracle Database instance. This deployment configuration is simple to configure and easy to manage, and it is familiar to existing Oracle GoldenGate customers.

Distributed Topologies for Integrated Extract: Different Machine, Real-Time



Source requires:

- Database 10.2 or later
- If not 11.2.0.3, then must run with downstream

Downstream requires:

- Database 11.2.0.3 or later

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

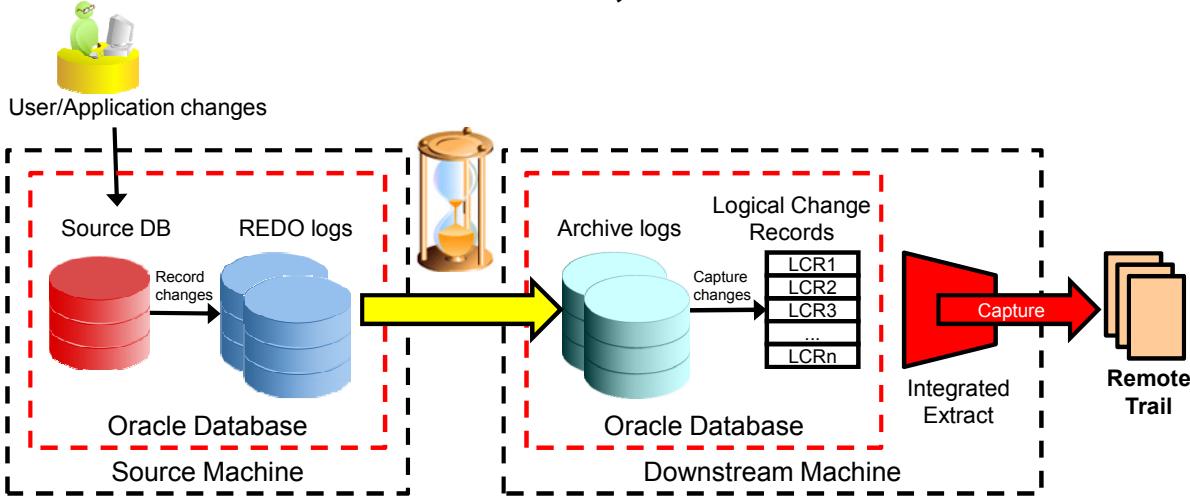
While keeping real-time mode in consideration, Integrated Extract supports another configuration called the *downstream* configuration, where the source database instance and Integrated Extract processes are on different machines.

In this deployment configuration, Oracle Database needs to be installed on both the source machine and the downstream machine. Oracle Database on the downstream machine is used to hold minimal data or metadata specific to the Oracle internal log processing module. From the source machine, change data records from REDO logs are *shipped continuously* to the downstream machine as *Standby REDO logs*. These Standby REDO logs are processed by the Oracle internal log processing module and are available as Logical Change Records. Later this LCR data is read by the consumer thread of Integrated Extract and processed by the producer thread of the same Integrated Extract. After processing, the records are written in to trail file.

Note that change data records from the source database will not be persisted in the downstream database. As mentioned earlier, the main purpose of the downstream database is to hold some state-specific data or metadata, which is minimal in nature and is specific to the log processing module. Change data records from the REDO logs of the source database are simply transported or shipped continuously as Standby REDO logs on the downstream machine.

If the source is not Oracle Database version 11.2.0.3, not all new features are available. For a list of available features, see the section titled “Supported Features Based on Source DB Version” later in this lesson.

Distributed Topologies for Integrated Extract: Different Machine, Non-Real-Time



Useful for offloading processing overhead

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

For non-real-time mode, archive logs can be processed in the configuration shown in this slide.

Only downstream configuration is supported. In this configuration, Oracle Database needs to be installed on both the source machine and the downstream machine. Oracle Database on the downstream machine is used to hold minimal data (or metadata) specific to Oracle internal log processing module.

REDO logs generated on the source machine will be available as Archive logs on the downstream machine. This is similar to how you would offload archive log processing using Classic Extract.

Benefits of this configuration:

- Production databases that are sensitive in nature will not be significantly disturbed. No additional software is required to install on the production machine.
- Using a downstream mining database for capture may be desirable to offload the capture overhead and any other overhead from transformation or other processing from the production server.
- The production database version can be independent of the downstream database.

Supported Features Based on Source DB Version

DB Version	12.1.0	11.2.0.4	11.2.0.2	11.1.0.7	10.2
XML Types	Y	Y	N	N	N
REDO-based Secure Files	Y	Y	N	N	N
Fetch	Y	Y	Y	N	N
XA-RAC	Y	Y	Y	N	N
Compression	Y	Y	Y	N	N
TDE/TSE	Y	Y	Y	Y	N
Integrated Apply	Y	Y	N	N	N
Large VARCHAR2	Y	N	N	N	N
Multitenant CDB	Y	N	N	N	N

Depending on Oracle Database versions, not all new features are supported.



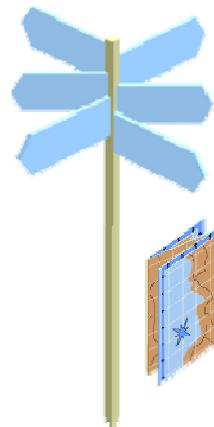
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

- **12.1.0:** New 12c features include CDB, Integrated Replicat (Apply), and more datatypes such as Large VARCHAR2 (32 KB).
- **11.2.0.4:** All new 11g features are supported. In particular, the compression supports Exadata Hybrid Columnar Compression (EHCC) and other compression types such as OLTP and Segment compression.
- **11.2.0.2:** There is no support for XML Binary or XML Object Relational (XML-OR), or for REDO-based secure files.
- **11.1.0.7:** There is support only for Transparent Data Encryption (TDE) and Tablespace Encryption (TSE).
- **10.2:** This *does not* support new Oracle GoldenGate features, such as XML Types, Fetch support for Secure files, ADTs, VARRAYS, Nested Tables, Object Tables, Compression, XA transactions, Capture Support for REDO-based secure files and TDE/TSE. Those features are not available in this configuration.

All source configurations presume a downstream Oracle Database version 11.2.0.3 or later. The target database can be anything, including non-Oracle.

Roadmap

- Architecture
- Extracts
- Initial Loads
 - DB-Specific Methods
 - GoldenGate-Specific Methods
- Checkpoints



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Online Versus Batch Operation

- Change data capture and delivery can be run either *continuously* (online) or as a *special run* (batch run) to capture changes for a specific period of time.
- The initial load is always a *special run* (batch run).
- An initial load takes a copy of the entire source data set, transforms it if necessary, and applies it to the target tables so that the movement of transaction data begins from a synchronized state.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can use Oracle GoldenGate to perform a stand-alone batch load to populate database tables for migration or other purposes. The other method is to load data into database tables as part of an initial synchronization run in preparation for change synchronization with Oracle GoldenGate.

The initial load can be performed from an active source database. Users and applications can access and update data while the load is running. You can also perform an initial load from a quiesced (temporarily inactive) source database if you delay access to the source tables until the target load is completed.

Running an Initial Load

Oracle GoldenGate can be used to load data in the following ways:

- **Direct load:** Extract sends data directly to Replicat to apply using SQL.
- **Direct bulk load:** Replicat uses the Oracle SQL*Loader API.
- **File to Replicat:** Extract writes to a file that Replicat applies using SQL.
- **File to database utility:** Extract writes to a file formatted for a DB bulk load utility.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

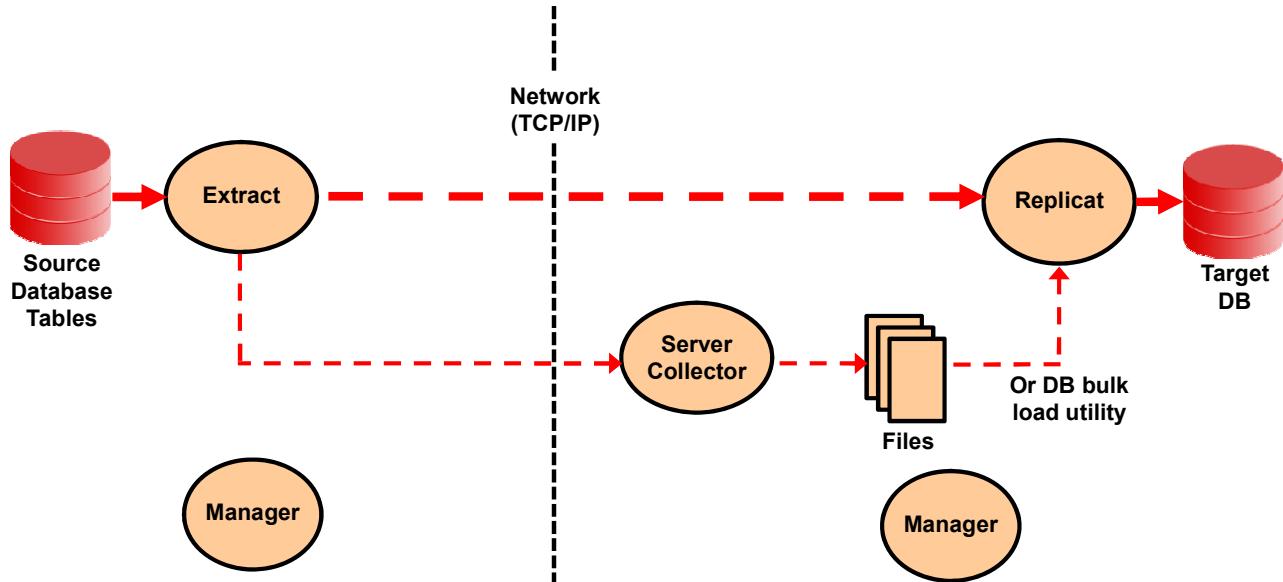
Oracle GoldenGate can be used to load data from the tables of the source database (rather than from the transaction logs) for an initial load. This feature is useful for relatively small tables and for initial loads in a heterogeneous environment. As an alternative, consider using other non-GoldenGate tools for larger databases or tables. For example, a DBA might consider cloning a source database by using the Recovery Manager (RMAN) duplicate features or by using the Oracle Data Pump utility.

The Extract process writes to a trail file and can optionally write to an ASCII, SQL, and XML file by using the `FormatASCII`, `FormatSQL`, and `FormatXML` Extract options, respectively. Do not use `FormatASCII`, `FormatSQL`, or `FormatXML` if the data will be processed by the Replicat process, because Replicat expects the default canonical format.

Note: For more information about Data Pump and the SQL*Loader utility, see the Oracle website:

<http://www.oracle.com/technetwork/database/enterprise-edition/index-093639.html>

Initial Load



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

On the source system:

1. An Extract process captures source data directly from tables.
2. The Extract process sends data in large blocks across a TCP/IP network to the target system.

On the target system, one of the following scenarios may occur:

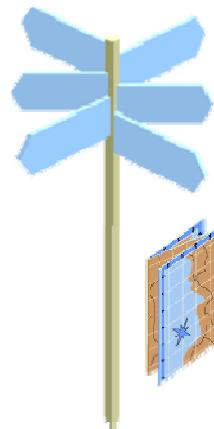
- **Direct load:** Replicat reads the data stream and concurrently applies the data to the target database by using SQL.
- **Direct bulk load (Oracle):** Replicat can apply the data by using the Oracle SQL*Loader API to improve performance.
- **File to Replicat:** Server Collector reassembles and writes the data to Extract files. Replicat applies the data to the target database by using SQL.
- **File to database utility:** Server Collector reassembles and writes the data to files formatted for a bulk loader, which applies the data to the target database.

Manager processes on both systems control activities such as starting, monitoring, and restarting processes; allocating data storage; and reporting errors and events.

This topic is covered in more detail in the lesson titled “Configuring the Initial Load.”

Roadmap

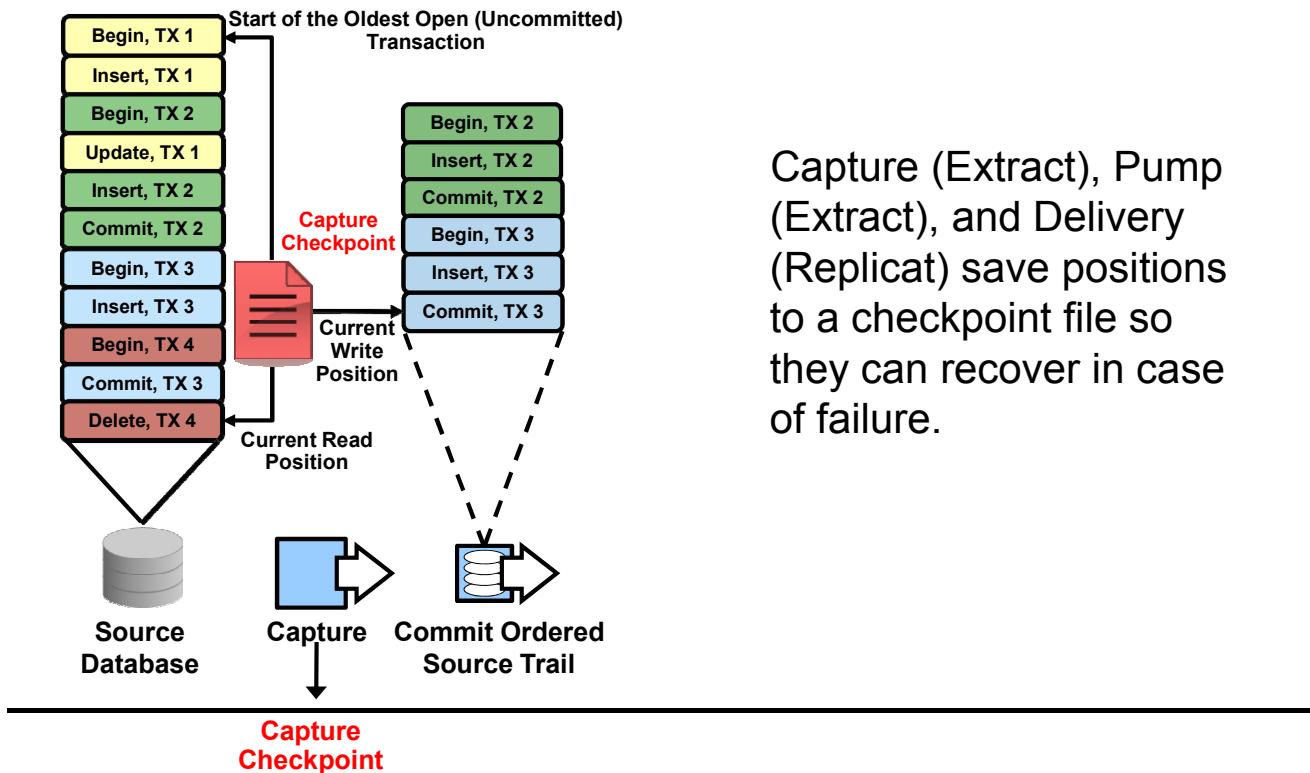
- Architecture
- Extracts
- Initial Loads
- Checkpoints
 - Capture
 - Pump
 - Delivery



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Checkpoints: Capture



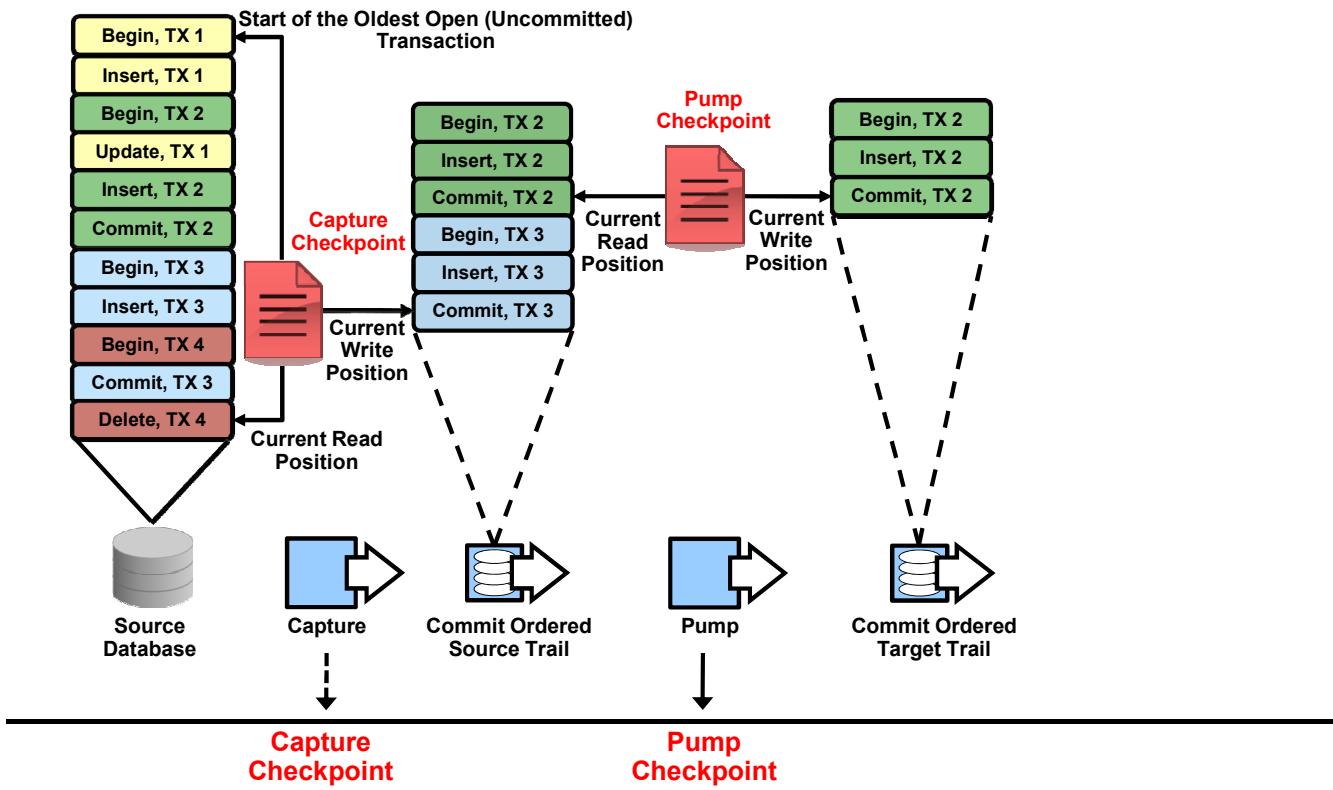
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Checkpoints are used during online change synchronization to store the current read and write position of a process. Checkpoints ensure that data changes marked for synchronization are extracted, and they prevent redundant extractions. They provide fault tolerance by preventing the loss of data if the system, the network, or an Oracle GoldenGate process needs to be restarted. Capture (Extract), Pump (Extract), and Delivery (Replicat) save positions to a checkpoint file so they can recover in case of failure. Extract creates checkpoints for its positions in the data source and in the trail. Replicat creates checkpoints for its position in the trail.

Checkpoint information is maintained in checkpoint files in the `dirchk` subdirectory of the Oracle GoldenGate directory. This is not to be confused with the optional checkpoint *table* stored on the database.

Checkpoints: Pump

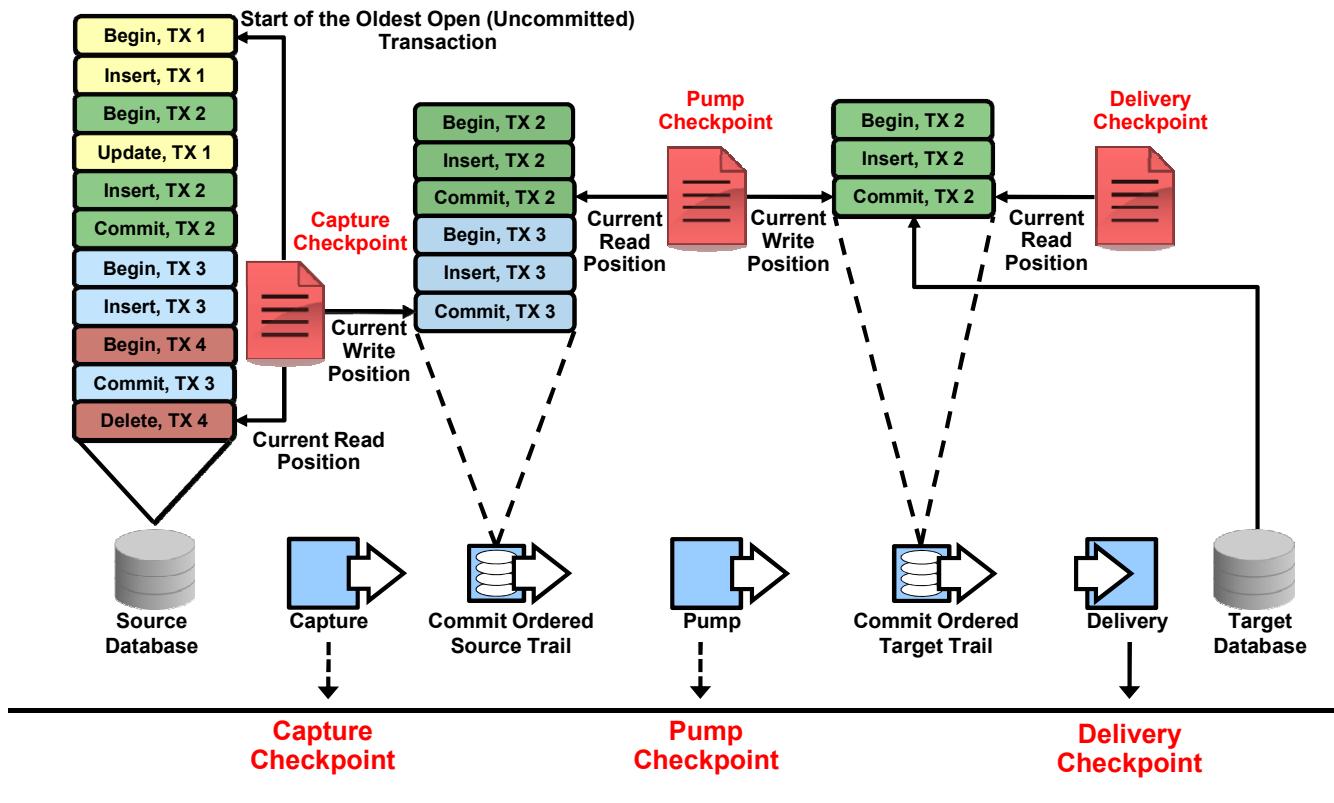


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The optional pump checkpoint adds an extra layer of reliability.

Checkpoints: Delivery



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Checkpoints help Oracle GoldenGate ensure that data is processed at the destination in the same order in which it was committed on the source. Checkpoints also prevent redundant extractions.

Commit Sequence Number (CSN)

A CSN:

- Is an identifier that Oracle GoldenGate constructs to identify a source transaction
- Uniquely identifies a particular point in time



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

A commit sequence number (CSN) is an identifier that Oracle GoldenGate constructs to identify a source transaction for the purpose of maintaining transactional consistency and data integrity.

It uniquely identifies a particular point in time at which a transaction commits to the database. Each kind of database management system generates some kind of unique serial number of its own at the completion of each transaction; this number uniquely identifies that transaction. A CSN captures this same identifying information and represents it internally as a series of bytes, but the CSN is processed in a platform-independent manner.

A comparison of any two CSN numbers, each of which is bound to a transaction-commit record in the same log stream, reliably indicates the order in which the two transactions are completed.

Discussion Questions

1. How is Oracle GoldenGate different from simply replicating database operations?
2. What is the purpose of checkpointing?



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

1. Log-based change data capture, decoupled from database architecture; real-time, heterogeneous, and transactional
2. Recovery (if an Oracle GoldenGate process, network, or system goes down)

Quiz

Which of the following statements is true?

- a. Oracle GoldenGate checkpoints are stored in an Oracle control file.
- b. The CKPT background process records the Oracle GoldenGate checkpoint.
- c. The Oracle GoldenGate checkpoint supports Oracle GoldenGate recovery.
- d. There is likely to be only one Oracle GoldenGate checkpoint file.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: c

Summary

In this lesson, you should have learned how to:

- Describe the uses of Oracle GoldenGate
- Describe Oracle GoldenGate parameters, process groups, and GGSCI commands
- Explain change capture and delivery (with and without a data pump)
- Explain initial data load
- Compare batch and online operations
- Explain Oracle GoldenGate checkpointing



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

There is no practice for Lesson 3.

Installing Oracle GoldenGate



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

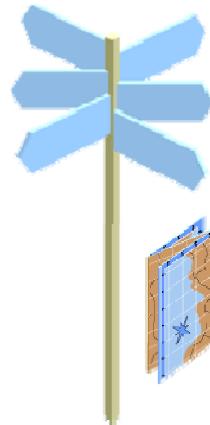
- Download the required Oracle GoldenGate Media Pack
- Install Oracle GoldenGate on Linux, UNIX, and Windows systems
- Locate and use OGG documentation
- Run Oracle GoldenGate commands from the Oracle GoldenGate Software Command Interface (GGSCI)
- Identify the types of GGSCI commands
- Use `Obey` files to automate tasks



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Roadmap

- Download
 - Software System Requirements
 - OS Requirements
 - Download Screens
- Install
- GGSCI



ORACLE®

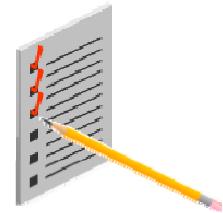
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Software System Requirements

Software	OGG 11.1.1	OGG 11.2.1	OGG 12.1
DB 9.2	Y		
DB 10.1	Y		
DB 10.2	Y	Y	
DB 11.1	Y	Y	Y
DB 11.2	Y	Y	Y
DB 12.1			Y

Operating system:

Many combinations are certified for database versions and operating systems. Check the documentation for certification requirements.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Supported platforms for Oracle databases:

- Oracle GoldenGate version 11.1 (and later) supports DML and DDL on Oracle Database 9.2 and later.

For the latest OS and DB Certification matrix, see:

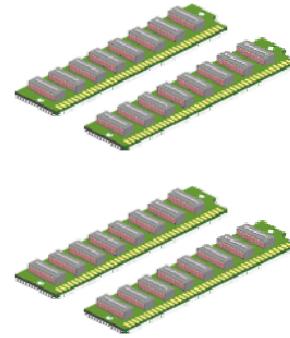
<http://www.oracle.com/technetwork/middleware/fusion-middleware/documentation/fmw-1212certmatrix-1970069.xls>

or, more generally (including back-level):

<http://www.oracle.com/technetwork/middleware/ias/downloads/fusion-certification-100350.html>

Operating System Requirements

- Memory requirements:
 - RAM required by Oracle GoldenGate depends on the number of concurrent processes running.
 - Each Extract and Replicat process needs approximately 25 MB to 55 MB of memory.
 - Swap space must be sufficient for each Oracle GoldenGate Extract and Replicat process.
 - Also allow RAM for Database SGA.
- Disk requirements
- Network requirements



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Supported operating systems for OGG 12c:

- **Linux x86-64:** Oracle Linux 5 and 6, Red Hat Enterprise Linux 5 and 6
- **Solaris x86-64:** Versions 10 and 11
- **SPARC 64-bit:** Versions 10 and 11
- **Windows x64:** Versions 2008 R2 and 2012

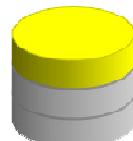
The amount of memory that is required for Oracle GoldenGate depends on the number of concurrent processes that is running. At minimum, there is a primary Extract process that captures source data, a secondary Extract data-pump process that transfers data across the network, and one or more Replicat processes that apply the replicated data to the target.

The Oracle GoldenGate GGSCI command interface fully supports up to 5,000 concurrent Extract and Replicat processes per instance of Oracle GoldenGate. An instance of Oracle GoldenGate equates to one Manager process, which is the main controller process. Each Extract and Replicat process needs approximately 25 MB to 55 MB of memory (or more) depending on the size of the transactions and the number of concurrent transactions.

By default, integrated capture Extract requires 1 GB Max_SGA_Size on Oracle Database in Streams_Pool_Size *per extract*. That can add up to a lot when time is short.

Operating System Requirements

- Memory requirements
- Disk requirements
 - 450 MB to 800 MB
 - Trail files: You can assign a separate disk for trail files. The default size is 100 MB each, but to optimize space, use the PurgeOldExtracts parameter.
- Network requirements
 - Configure to use TCP/IP.
 - Configure one port for communication between the Manager process and other GoldenGate processes.
 - Configure a range of ports for local Oracle GoldenGate communications.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Assign the following free disk space:

- 450 MB to 800 MB, depending on the database and platform. This includes space for the compressed download file and space for the uncompressed files. You can delete the download file after the installation is complete.
- 40 MB for the working directories and binaries for each instance of Oracle GoldenGate that you are installing on the system. For example, to install two builds of Oracle GoldenGate in two separate directories, allocate 80 MB of space.
- Additional disk space on any system that hosts Oracle GoldenGate trails, which are files that contain the working data. The space that is consumed by the trails varies depending on the volume of data that will be processed. A good starting point is 1 GB.

For Network TCP/IP requirements:

- Configure the system to use TCP/IP, including DNS. IPv6 is supported by default.
- Configure the network with the host names or IP addresses of all systems that will be hosting Oracle GoldenGate processes and to which Oracle GoldenGate will be connecting. Host names are easier to use.
- Oracle GoldenGate requires the following unreserved and unrestricted TCP/IP ports:
 - One port for communication between the Manager process and other Oracle GoldenGate processes
 - A range of ports for local Oracle GoldenGate communications. This can be the default range starting at port 7840 or a customized range of up to 256 other ports.

Downloading Oracle GoldenGate

1. In a browser, go to <http://edelivery.oracle.com>.
2. Enter your personal information.
3. In the Fusion Middleware Media Pack, search for platform.
4. Select and download the Oracle GoldenGate Media Pack.

The screenshot shows two consecutive pages from the Oracle Software Delivery Cloud.

Page 1: A search interface for "Fusion Middleware Media Pack". It has dropdown menus for "Select a Product Pack" (set to "Oracle Fusion Middleware") and "Platform" (set to "Linux x86-64"). A "Go" button is below. Below the buttons is a table titled "Results" with columns: Select, Description, Release, Part Number, Updated, and # Parts / Size. One row is selected, showing "Oracle GoldenGate on Oracle v12.1.2 Media Pack for Linux x86-64" with details: Release 12.1.2.0.0, Part Number B74844-01, Updated OCT-17-2013, and # Parts / Size 1 / 325M. A cursor arrow points to this row.

Page 2: A detailed view of the selected media pack. It shows the title "Oracle GoldenGate on Oracle v12.1.2 Media Pack v1 for Linux x86-64" and two buttons: "Readme" and "View Digest". Below is another table with columns: Select, Name, Part Number, and Size (Bytes). A single row is listed: "Download" (button), "Oracle GoldenGate V12.1.2.0.0 for Oracle on Linux x86-64", "V40146-01", and "325M". A cursor arrow points to the "Download" button. To the right of the second page is a blue downward-pointing arrow and a small icon of a computer monitor and server tower.

ORACLE®

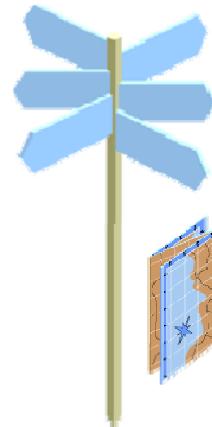
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

1. Go to Oracle Software Delivery Cloud at <http://edelivery.oracle.com>.
2. Personal information includes agreeing to the terms and conditions.
3. Search for:
 - Oracle Fusion Middleware
 - Linux, Windows, or other
 - Platform OS bits (32 or 64)
 - Version
4. Select the Media Pack and download for Oracle 12c.

Older versions of Oracle GoldenGate are available for older versions of Oracle Database and non-Oracle databases.

Roadmap

- Download
- Install
 - OUI (GUI)
 - Linux
 - Windows
 - Directories
 - Documentation
- GGSCI



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Installation on UNIX, Linux, or z/OS Systems

1. Extract the OGG .zip file to /tmp.
2. Run the Oracle Universal Installer:

```
[OS prompt] unzip 121200_fbo_ggs_* /tmp  
[OS prompt] cd /tmp/fbo_ggs_Linux_x64_shiphome/Disk1  
[OS prompt] ./runInstaller
```

3. From the Oracle GoldenGate HOME directory, run the GGSCI program.

```
[OS prompt] cd $OGG_HOME  
[OS prompt] ./ggsci  
Oracle GoldenGate Command Interpreter for Oracle  
Version 12.1.2.0.0 17185003 OGGCORE_12.1.2.0.0_PLATFORMS_130924.1316_FBO  
Linux, x64, 64bit (optimized), Oracle 12c on Sep 25 2013 02:33:54  
Operating system character set identified as UTF-8.  
Copyright (C) 1995, 2013, Oracle and/or its affiliates. All rights reserved.  
GGSCI (host) 1> Help
```



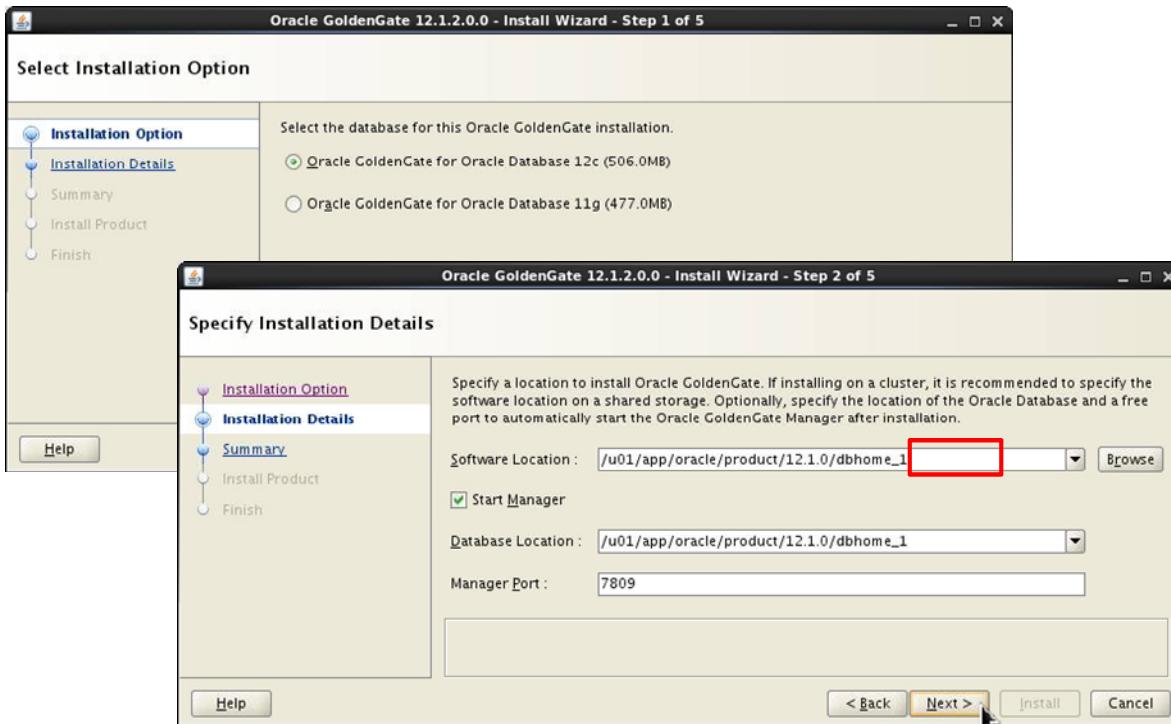
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Optionally, it is helpful to define the \$OGG_HOME environment variable and place it in .profile or .bashrc for each session.

To install Oracle GoldenGate, use the gzip and tar options that are appropriate for your operating system. If you are installing Oracle GoldenGate into a cluster environment, make certain that the Oracle GoldenGate binaries and files are installed on a file system that is available to all cluster nodes. After installing Oracle GoldenGate, configure the Oracle GoldenGate Manager process within the cluster application (as directed by the vendor's documentation) so that Oracle GoldenGate will fail over properly with the other applications.

The Manager process is the master control program for all Oracle GoldenGate operations. An Oracle GoldenGate instance is a single installation of Oracle GoldenGate.

Oracle Universal Installer GUI (New with 12c)



ORACLE®

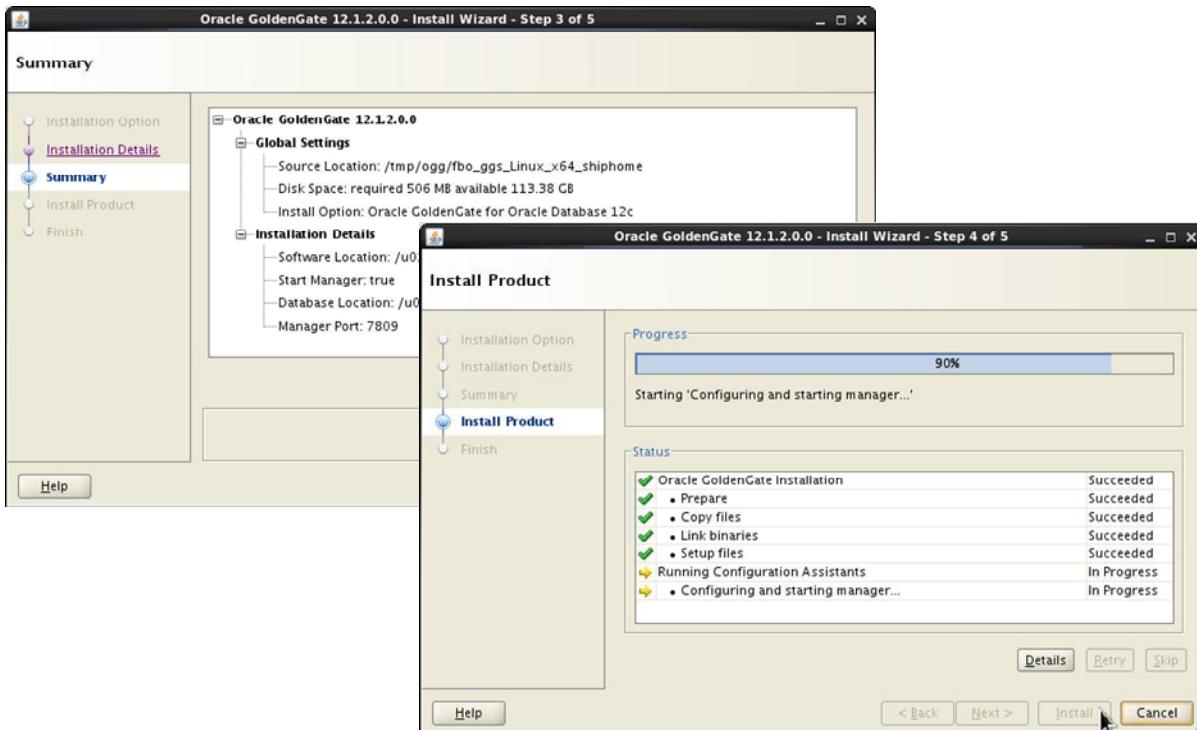
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The primary advantage of using Oracle Universal Installer (OUI) is that it registers Oracle GoldenGate with `oraInventory` for later audit purposes. The disadvantage is that it takes up more disk space to install. The previous 11g methods of installing by unzipping are mentioned in the documentation but are not available.

Do not accept the default location. At least, make it one more level down with `/ogg` or something else. Otherwise, the dozens of GoldenGate files and directories will be mixed in with the database code. Even though the `$ORACLE_HOME` default location says 12.1.0, this version of GoldenGate is 12.1.2.

Although it is helpful for the Install Wizard to offer to start the Manager, you will probably need to modify the Manager parameter files anyway, and that requires stopping and restarting the Manager. For this reason, there is no reason to start the Manager here but then have to stop and restart it immediately after the Install Wizard ends. You will need not just one port but a range of ports.

Oracle Universal Installer GUI (New with 12c)



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Step 3 (Summary) gives you the opportunity to create and save a response file for making repeated silent installations. The final installation step runs the `Create Subdirs GGSCI` command.

`runInstaller` does not give you the chance to modify, upgrade, or remove any code. The installer is not directly suited for RAC or cluster installation. There are several Tech notes in *My Oracle Support Knowledge Base* (<https://support.oracle.com/>) on how to do that manually.

The installer creates `$OGG_HOME/deinstall/deinstall.sh` for removing the code and inventory entries. The deinstaller runs in text (not GUI) mode.

Installation on Windows Systems

1. Download the Oracle GoldenGate Media Pack .zip file to C:\GGS.
2. Unzip the .zip file to the C:\GGS folder or /tmp folder. (GGS is an example.)
3. Configure a Windows Service Name for the Manager process in a GLOBALS parameter file (required only if there are multiple Manager processes on the server):

```
C:\GGS> Install AddService AddEvents
```

4. Run the command shell and change to the new Oracle GoldenGate directory. Then create subdirectories:

```
GGSCI> Create Subdirs
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Do not install Oracle GoldenGate into a folder that contains spaces in its name (for example, “GoldenGate Software”). The application references path names, and the operating system does not support path names that contain spaces, whether they are within quotation marks or not.

The INSTALL ADDSERVICE ADDEVENTS command will install event messages into the Windows system registry and will install a Windows Service. Oracle GoldenGate needs only one Manager on each node, but you can install multiple Managers. For example, you can install a “Test” Manager and a “Production” Manager. Multiple Manager processes require multiple installations and unique Windows service names. You may want to create a Windows service for the Manager; otherwise, if the user logs outs, the Manager process stops. You may also want to delete a service. To delete a service, enter INSTALL DELETESERVICE DELETEEVENTS.

Oracle GoldenGate Directories

Directory	Contents
dirbdb	* Berkeley DB for Monitor facility
dirchk	Oracle GoldenGate checkpoint files
dircrd	* Credential Store for users and passwords
dirdat	Oracle GoldenGate trail and Extract files
dirdef	Column definitions produced by <code>defgen</code> to translate heterogeneous data
dirjar	Java executables to support OGG Monitor and other services
dirpcs	Process status files
dirprm	Parameter files
dir rpt	Process report files
dirs q l	SQL scripts
dirtmp	Temp storage for transactions that exceed allocated memory
dirwlt	* Wallet for masterkey; works with <code>dircrd</code>
dirwww	* Web artifacts for Monitoring

A few other directories are not shown. * New with 12c



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Some key directories:

- **dirchk:** Contains the checkpoint files created by Extract and Replicat processes, which store current read and write positions to support data accuracy and fault tolerance. These files are written in internal Oracle GoldenGate format and should not be modified. The file name format is `<group_name><sequence_number>. <ext>`, where `<sequence_number>` is a sequential number appended to aged files and `<ext>` is either `cpe` for Extract checkpoint files or `cpr` for Replicat checkpoint files. Examples include `ext1.cpe` and `rep1.cpr`.
- **dircrd:** The default Credential store location for encrypted user IDs and passwords. You can change this location.
- **dirdat:** The default location for Oracle GoldenGate trail files and extract files created by Extract processes to store records of extracted data for further processing, either by the Replicat process or another application or utility. `dirdat` is written in internal GoldenGate format and should not be modified.
The file name format is a user-defined two-character prefix followed by either a six-digit sequence number (used by trail files; for example, `rt000001`) or the user-defined name of the associated Extract process group (used by extract files; for example, `finance`).

- **dirdef:** The default location for data definition files created by the `defgen` utility to contain source or target data definitions used in a heterogeneous synchronization environment. These files are written in plain text ASCII format.
The file name format is a user-defined name specified in the `defgen` parameter file (for example, `defs.dat`). These files can be edited to add definitions for newly created tables. If you are unsure how to edit a definitions file, contact technical support.
- **dirjar:** Contains the Java executable files that support Oracle GoldenGate Monitor
- **dirpcs:** The default location for status files. The file name format is `<group>.<ext>`, where `<group>` is the name of the group and `<ext>` is an extension of `pce` (Extract), `pcr` (Replicat), or `pcm` (Manager).
These files are created only while a process is running. The file shows the program name, the process name, the port, and process ID that is running. Do not edit these files. Examples include `mgr.pcm` and `ext.pce`.
- **dirprm:** The default location for Oracle GoldenGate parameter files created by Oracle GoldenGate users to store run-time parameters for Oracle GoldenGate process groups or utilities. These files are written in plain text ASCII format.
The file name format is `<group_name/user-defined_name>.prm` or `mgr.prm`. These files can be edited to change Oracle GoldenGate parameter values. They can be edited directly from a text editor, such as `gedit` or `Notepad`, or by using the `EDIT PARAMS` command in GGSCI. Examples include `defgen.prm` and `finance.prm`.
- **dirrpt:** The default location for process report files created by Extract, Replicat, and Manager processes to report statistical information about a processing run. These files are written in plain text ASCII format.
The file name format is `<group_name><seq_num>.rpt`, where `<seq_num>` is a sequential number appended to aged files. Do not edit these files. Examples include `fin2.rpt` and `mgr4.rpt`.
- **dirsq1:** The default location for SQL scripts
- **dirtmp:** The default location for storing large transactions when the size exceeds the allocated memory size. Do not edit these files.
- **dirwlt:** The default Oracle Wallet location for the masterkey; you can change this location. Existing GUI Wallet utilities that come with other products, such as the Oracle Database “Oracle Wallet Manager” (`owm`), do not work on this version of the Wallet.
- **dirwww:** Artifacts for Oracle Monitoring software; separately licensed
- **jagent** and **jdk:** Used for Java bits of the Monitor facility

Oracle GoldenGate Documentation

- Installation and setup guides (by database)
- Administration Guide
- Reference Guide
- Error Messages Reference
- Logdump Reference
- Upgrading



Note: You can download the current documentation from
<http://docs.oracle.com/goldengate/1212/gg-winux/index.html>
and older versions from
<http://www.oracle.com/technetwork/middleware/goldengate/documentation/index.html>



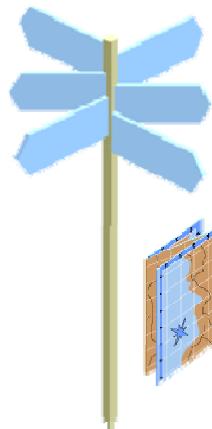
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate Documentation for Windows and UNIX Platforms

- **Oracle GoldenGate installation and setup guides:** Are available for each database that is supported by Oracle GoldenGate and include database-specific configuration information
- **Administering Oracle GoldenGate for Windows and UNIX:** Introduces Oracle GoldenGate components and explains how to plan for, configure, and implement Oracle GoldenGate on the Windows and UNIX platforms
- **Reference for Oracle GoldenGate for Windows and UNIX:** Provides detailed information about Oracle GoldenGate parameters, commands, and functions for the Windows and UNIX platforms
- **Error Messages Reference for Oracle GoldenGate for Windows and UNIX:** Includes suggested fixes for common errors
- **Logdump Reference for Oracle GoldenGate:** Used for viewing trail files
- **Upgrading Oracle GoldenGate for Windows and UNIX:** Provides information about migrating from 11g to 12c
- **Management Pack for Oracle GoldenGate: Oracle GoldenGate Director Administrator's Guide**
- **Management Pack for Oracle GoldenGate: Oracle GoldenGate Director Release Notes**

Roadmap

- Download
- Install
- GGSCI
 - Keyboard
 - Obey files
 - New commands



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

GGSCI Command Interface

- GGSCI is the command interface that executes Oracle GoldenGate commands.
- Start GGSCI from the Oracle GoldenGate installation directory:

```
[OS prompt] cd <GoldenGate_install_location>
[OS prompt] ./ggsci
```

- For the help Summary page:

```
GGSCI> Help
```

- For help on a specific command:

```
GGSCI> Help <command> <object>
GGSCI> Help Add Extract
```

- To re-execute a command:

```
GGSCI> !
```

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Golden Gate Software Command Interface (GGSCI) provides online help for all commands. The following is an example of the information that is returned when you enter HELP Status Extract. Use Status Extract to determine whether or not Extract groups are running.

Syntax

```
Status Extract <group_name>
[, Tasks]
[, AllProcesses]
```

- *<group_name>* is the name of a group or a wildcard (*) to specify multiple groups.
- AllProcesses displays the status of all Extract processes, including tasks.
- Tasks displays the status of all Extract tasks.

Examples

- Status Extract finance
- Status Extract fin*

GGSCI Commands

	Manager	Extract	Replicat	ER	ExtTrail	RmtTrail	Trandata	Checkpoint Table	Trace Table
Add		X	X		X	X	X	X	X
Alter		X	X		X	X			
Cleanup		X	X					X	
Delete		X	X	X	X	X	X	X	X
Info	X	X	X	X	X	X	X	X	X
Kill		X	X	X					
Lag		X	X	X					
Refresh	X								
Send	X	X	X	X					
Start	X	X	X	X					
Stats		X	X	X					
Status	X	X	X	X					
Stop	X	X	X	X					

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objects

- **Manager, Extract, Replicat:** Oracle GoldenGate processes
- **ER:** Multiple Extract and Replicat processes
- **ExtTrail:** Local Extract trail
- **RmtTrail:** Remote trail
- **TranData:** Enables supplemental logging of Primary Keys and/or indexes for a schema or table in the transaction logs. Also, use SchemaTranData for all the present and future tables in a schema.
- **CheckpointTable:** Checkpoint table (on target database)
- **TraceTable:** Oracle trace table (on target database)

Commands

- **Add:** Creates an object or enables TranData or SchemaTranData capture
- **Alter:** Changes the attributes of an object
- **Cleanup:** Deletes the run history of a process or removes records from a checkpoint table

- **Delete:** Deletes an object or disables TranData or SchemaTranData capture
- **Info:** Displays information about an object (status, and so on)
- **Kill:** Forces a process to stop (no restart)
- **Lag:** Displays the lag between when a record is processed by the process and the source record timestamp
- **Refresh:** Refreshes Manager parameters (except port number) without stopping Manager
- **Send:** Sends commands to a running process
- **Start:** Starts a process
- **Stats:** Displays statistics for one or more processes
- **Status:** Indicates whether a process is running
- **Stop:** Stops a process gracefully

GGSCI Commands

	Commands
Parameters	<code>Set Editor</code> , <code>Edit Params</code> , <code>View Params</code>
Database	<code>DBLogin</code> , <code>MiningDBLogin</code> , <code>Encrypt Password</code> , <code>List Tables</code>
DDL	<code>DumpDDL [SHOW]</code>
Miscellaneous	<code>!command</code> , <code>Create Subdirs</code> , <code>FC</code> , <code>Help</code> , <code>History</code> , <code>Info All</code> , <code>Obey</code> , <code>Shell</code> , <code>Show</code> , <code>Versions</code> , <code>View GGSevt</code> , <code>View Report</code>

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Parameter Commands

- `Set Editor`: Changes the default text editor for the current GGSCI session from Notepad (Windows) or `vi` (Linux) to any ASCII editor
- `Edit Params`: Edits a parameter file with an implied extension of `.prm`
- `View Params`: Displays the contents of a parameter file with an implied extension of `.prm`

Database Commands

- `DBLogin` and `MiningDBLogin`: Establish a database connection through GGSCI to a local or remote database
- `Encrypt Password`: Encrypts a database login password in a parameter file
- `List Tables`: Lists all tables in the database that match a wildcard string (asterisks and question marks)

DDL Commands

- `DumpDDL`: Saves the Oracle GoldenGate DDL history table to file
- `Show`: Displays the DDL information in standard output format

Miscellaneous Commands

- ***!command***: Executes a previous GGSCI command without modification
- **Create Subdirs**: Creates default directories within the Oracle GoldenGate home directory
- **FC**: Edits a previously issued GGSCI command
- **Help**: Displays information about a GGSCI command
- **History**: Lists the most recent GGSCI commands issued
- **Info All**: Displays the status and lag for all Oracle GoldenGate online processes on a system (Info All AllProcesses also displays tasks.)
- **Obey**: Runs a file containing a list of GGSCI commands
- **Shell**: Runs shell commands from within GGSCI
- **Show**: Displays the Oracle GoldenGate environment
- **Versions**: Displays OS and database versions
- **View GGSEvt**: Displays the Oracle GoldenGate event/error log
- **View Report**: Displays a process report for Extract or Replicat

GGSCI Examples

- Start a Manager process:

```
GGSCI> Start Mgr
```

- Add an Extract group:

```
GGSCI> Add Extract myext, TranLog, Begin Now
```

- Add a local trail:

```
GGSCI> Add ExtTrail /ggs/dirdat/lt, Extract myext
```

- Start an Extract group:

```
GGSCI> Start Extract myext
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In this example, a primary extract named `myext` extracts database changes from the transaction logs starting with records generated at the time the group was created. The changes will be written to a local trail (`lt`). The extract is then started. When an extract is started, it creates a new trail file rather than appending to an existing trail file.

Obey Files

- Obey files are used to automate a series of frequently used GGSCI commands.
- Create and save a text file that contains the commands, with one command per line:
[OS prompt] **more myscript.oby**

```
Start Mgr
Add Extract myext, TranLog, Begin Now
Add ExtTrail /ggs/dirdat/l1, Extract myext
Start Extract myext
```

- Then use the GGSCI Obey command to run the file:

```
GGSCI> Obey myscript.oby
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Obey can also appear in an Extract or Replicat parameter file for nesting other parameters. For nesting parameters, the file extension is .prm rather than .oby. The Obey command does not look for a specific file extension. If a file extension is part of the name, it must be used as in the following example:

```
GGSCI> Obey myfilename.oby
```

The AllowNested command enables the use of nested Obey files. A nested Obey file contains another Obey file. An attempt to run a nested Obey file in the default mode of NoAllowNested causes an error. The maximum number of nested files is 16 levels deep.

Oracle GoldenGate 12c: Miscellaneous New Features

- Multitenant Container Databases (CDB)
 - Pluggable Databases (PDB) within CDB
 - Three-part object names
- New Oracle Database `init.ora` parameter:
 - `ENABLE_GOLDENGATE_REPLICATION = TRUE`
- `SHOWSYNTAX` command now works with Dynamic SQL.
- Better wildcard naming support
- Automatic discard file creation
- Large `VARCHAR2` support (32 KB)
- Security enhancements:
 - Wallet Key
 - Credential Store



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Each of these features is explained in later lessons as they are encountered.

Running Oracle GoldenGate from the OS Shell

- You can also start Oracle GoldenGate processes from the OS command shell when running a batch job or initial load, as in the following example:

```
Shell> cd <GoldenGate_install_location>
Shell> ./extract paramfile <filepath> reportfile
      <filepath> [-p <port>]
Shell> ./replicat paramfile <filepath> reportfile
      <filepath>
```

- This is especially useful in scheduling Oracle GoldenGate batch jobs to run during off-peak hours using a command-line capable scheduler.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Manager must be running when you issue the following commands:

- *<filepath>* specifies the fully qualified name of the parameter and report files.
- paramfile can be abbreviated to pf.
- reportfile can be abbreviated to rf.

When you run Extract and Replicat from the command line, you do not get any feedback to stdout; you must look in `reportfile` to see the results.

Discussion Questions

1. Where can you find the Oracle GoldenGate software for downloading?
2. What is GGSCI?
3. Where can you view Oracle GoldenGate command syntax?
4. What is an Obey file? Why would you use it?



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

GoldenGate Commands: Discussion Points

1. <http://edelivery.oracle.com>
2. Oracle GoldenGate Software Command Interface, which is a text-based prompt
3. In the help or in the *Reference Guide* (online or downloaded)
4. An ASCII text file containing a sequence of Oracle GoldenGate commands or parameters. You use it for easy reuse of common command sequences (similar to a script).

Summary

In this lesson, you should have learned how to:

- Download the required Oracle GoldenGate Media Pack
- Install Oracle GoldenGate on Linux, UNIX, and Windows systems
- Locate and use the Oracle GoldenGate documentation
- Identify the types of GGSCI commands
- Run Oracle GoldenGate commands from the GGSCI prompt, from Obey files, and from the OS shell



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice 4 Overview: Installing Oracle GoldenGate

This practice covers the following topics:

- Preparing for the course practices
- Installing Oracle GoldenGate
- Introducing the GGSCI command-line interface



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Configuration Overview and Preparing the Environment

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

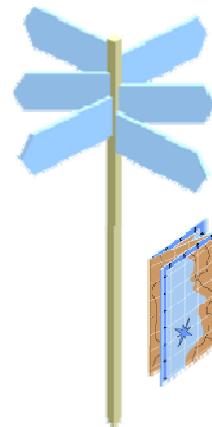
- Describe the high-level configuration steps
- Configure and start the Manager process
- Configure supplemental logging
 - On the database
 - At the table level
- Generate source definitions files
- Prepare a source database for transaction capture



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Roadmap

- Preparing the Environment
 - NLS
 - Character Set
 - Mixed-Case Object Names
 - Roles and Permissions
 - Network Troubleshooting
- Supplemental Logging: TranData
- Column Definitions: defgen



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Configuring Oracle GoldenGate

Oracle GoldenGate can be deployed in four steps:

1. Preparing the environment:
 - a) Decide NLS and mixed-case object name issues.
 - b) Set up a database user to access Oracle GoldenGate.
 - c) Enable supplemental logging.
 - d) Start the Manager process.
 - e) If you are using a heterogeneous source and/or target, create source definitions.
2. Change capture (Extract)
3. Initial load
4. Change delivery (Replicat)



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The installation of Oracle GoldenGate includes all of the components required to run and manage Oracle GoldenGate processing and Oracle GoldenGate utilities. The Manager process must be running on each system before Extract or Replicat can be started, and must remain running while those processes are running so that resource management functions are performed.

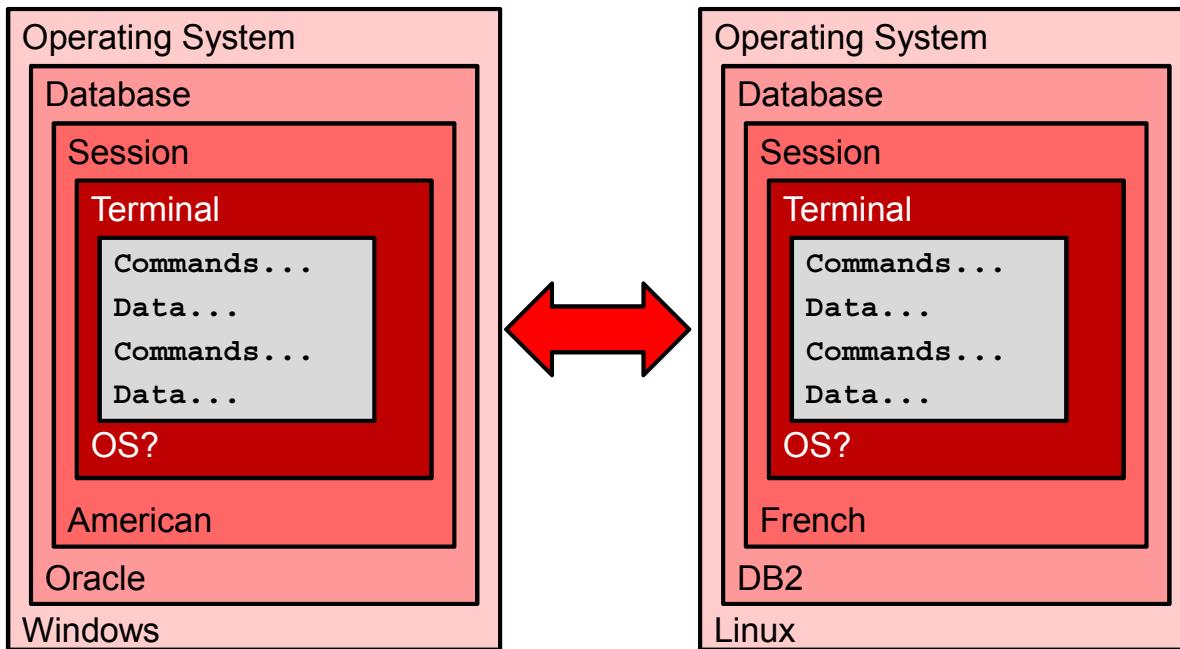
The source definitions file contains the definitions of the source tables and is required on the target system in heterogeneous configurations. Replicat refers to the file when transforming data from the source to the target.

To reconstruct an update operation, Oracle GoldenGate needs more information than the transaction logs provide by default. Adding supplemental log data forces the log writer to also include the primary keys to logs instead of just the columns that have changed.

This lesson covers Step 1: “Preparing the environment.” The remaining steps are covered in subsequent lessons.

Character Set: National Language Support (NLS)

Possible character set mismatches:



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

If everything is the same, character mapping is not a problem. However, there can be numerous differences at many places:

- OS
 - Windows vs. Linux
 - Unicode vs. Codepage
 - Multibyte Character Support vs. Fixed Byte
- Database
 - Oracle vs. OEM (for example, DB2)
 - ASCII vs. EBCDIC
 - SessionCharSet command in Oracle GoldenGate DBLogin and GLOBALS. This parameter supports Sybase, Teradata, and MySQL. The database character set for other databases (for example, Oracle) is obtained programmatically.
- Session
 - USA vs. other Latin (for example, French or Spanish)
 - Non-Latin vs. other non-Latin (for example, Chinese vs. Arabic)
- Terminal
 - CharSet command in Oracle GoldenGate parameters; otherwise, assumes local OS character set

Mixed-Case Object Names: ‘Single’ and “Double” Quotation Marks

Case-sensitivity for Oracle, DB2, and SQL/MX:

- *Unquoted* object name: Non-case-sensitive stores in *UPPER*

```
CREATE TABLE abc ( txt char(10) );
INSERT INTO abc VALUES ( 'text value' );

SELECT txt FROM abc WHERE txt = 'text value';
SELECT object_name FROM all_objects
WHERE object_name = 'ABC';
```

- *Quoted* object name: Case-sensitive stores in *MixedCase*

```
CREATE TABLE "abc" ( txt char(10) );
INSERT INTO "abc" VALUES ( 'text value' );

SELECT "txt" FROM "abc" WHERE "txt" = 'text value';
SELECT object_name FROM all_objects
WHERE object_name = 'abc';
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

White space in a quoted name is another issue. As a general rule, you should not have white space in an object name.

Stand-alone wildcards (asterisks) inherit case-sensitivity. Wildcards with other characters require you to state the case-sensitivity with “double quotation marks.”

Other databases may behave differently:

Teradata

- Regardless of unquoted or quoted object name, non-case-sensitivity always stores in *Mixed case* (as is).

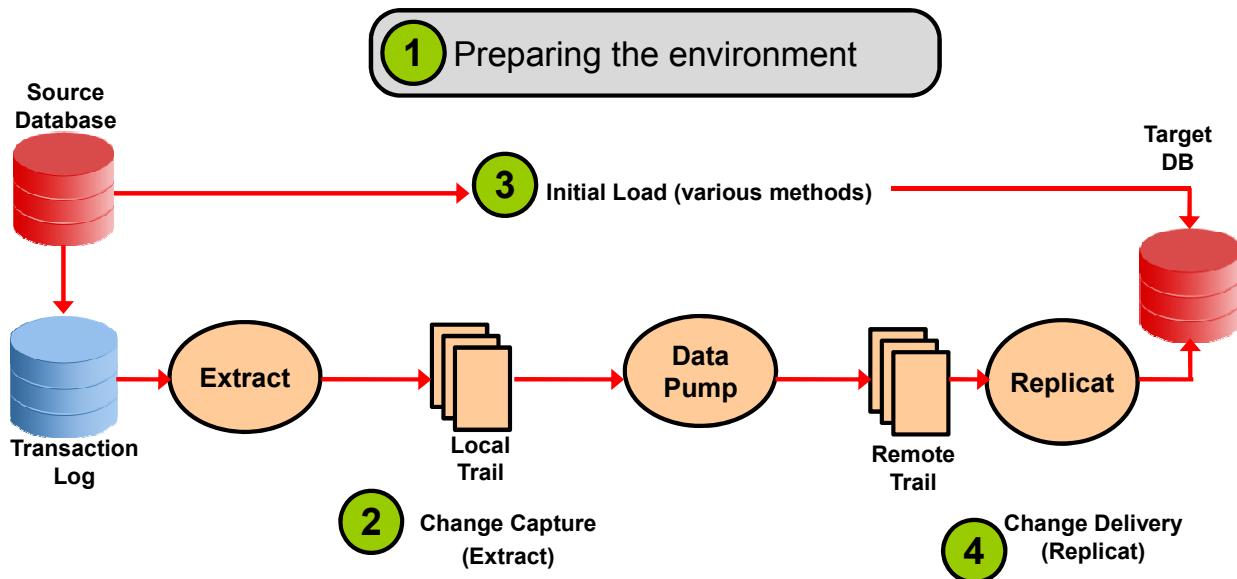
SQL Server

- You can configure case-sensitivity at the database level.
- Quoted and unquoted names are stored in *Mixed case* for both non-case-sensitive and case-sensitive databases.

MySQL

- You can configure case-sensitivity at the database level.
- Column, index, stored procedure, and triggers are not configurable, and are always non-case-sensitive.

Configuring Oracle GoldenGate



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate can be deployed quickly and easily in four steps. This lesson covers the first step; the remaining steps are covered in subsequent lessons.

1. Prepare the environment.
 - Install the Oracle GoldenGate software on source and target.
 - Set up a GoldenGate database user.
 - Enable supplemental logging.
 - For a heterogeneous source and/or target, generate source definitions so Replicat can process trail data.
2. Configure and start change capture to GoldenGate trail files (Extract processes: primary and data pump).
3. Perform initial load to synchronize databases by database-specific or Oracle GoldenGate methods. The timing of the initial load can happen at several places: before or after Step 2. If you have an initially empty table, the initial load step can be skipped altogether.
4. Configure and start change delivery (Replicat process).

For the purposes of teaching, these steps are presented in a different order.

Preparing the Environment: Oracle Database

To ensure access by Oracle GoldenGate processes, create a database user with the following privileges:

User Privilege	Extract (Source Side)	Replicat (Target Side)
CREATE SESSION, ALTER SESSION	X	X
RESOURCE	X	X
CONNECT	X	X
SELECT ANY DICTIONARY	X	X
FLASHBACK ANY TABLE or FLASHBACK ON <owner.table>	X	
SELECT ANY TABLE or SELECT ON <owner.table>	X	X
INSERT, UPDATE, DELETE ON <target tables>		X
CREATE TABLE		X
EXECUTE on DBMS_FLASHBACK package	X	



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Database Access

Although not required, Oracle GoldenGate recommends creating a user specifically for the Oracle GoldenGate application, with all of the privileges listed in the slide. To ensure that processing can be monitored accurately, do not permit other users or processes to operate as the Oracle GoldenGate user.

In general, the following permissions are necessary for the Oracle GoldenGate user:

- On the source system, the user must have permissions to read the data dictionary or catalog tables.
- On the source system, the user must have permissions to select data against the tables.
- On the target system, the user must have the same permissions as the Oracle GoldenGate user on the source system plus additional privileges to perform DML on the target tables.

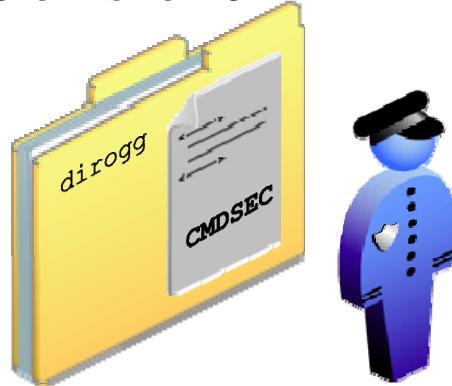
In addition, execute the following command in SQL*Plus as SYSDBA:

```
EXEC DBMS_GOLDENGATE_AUTH.GRANT_ADMIN_PRIVILEGE('GGUSER');
```

In this syntax, *GGUSER* is the database user ID that is used in GGSCI DBLogin commands. You can grant fewer privileges, but the default is both capture and apply, integrated and classic.

Using Command Security

- Command security can be established for Oracle GoldenGate to control which users have access to which Oracle GoldenGate functions.
- Security levels are defined by the operating system's user groups.
- To implement security, create a **CMDSEC** file.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can establish command security for Oracle GoldenGate to control which users have access to which Oracle GoldenGate functions. For example, you can allow certain users to issue the `Info` and `Status` commands, while preventing their use of the `Start` and `Stop` commands.

To implement security for Oracle GoldenGate commands, you create a `CMDSEC` text file in the Oracle GoldenGate directory. On a Linux or UNIX OS, the file name must be in uppercase with no extension. Without this file, the default access to all Oracle GoldenGate commands is granted to all users.

Sample CMDSEC Statements

File Contents	Explanation
#GG command security	Comment line
Status Replicat * Smith NO	STATUS Replicat is denied to user Smith.
Status * dpt1 * YES	Except for the preceding rule, all users in the dpt1 group are granted all STATUS commands.
Start Replicat * * NO	Except for the preceding rule, START Replicat is denied to all users.
* Extract 200 * NO	All Extract commands are denied to all groups with the ID 200.
* * root root YES	The root user is granted any command.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

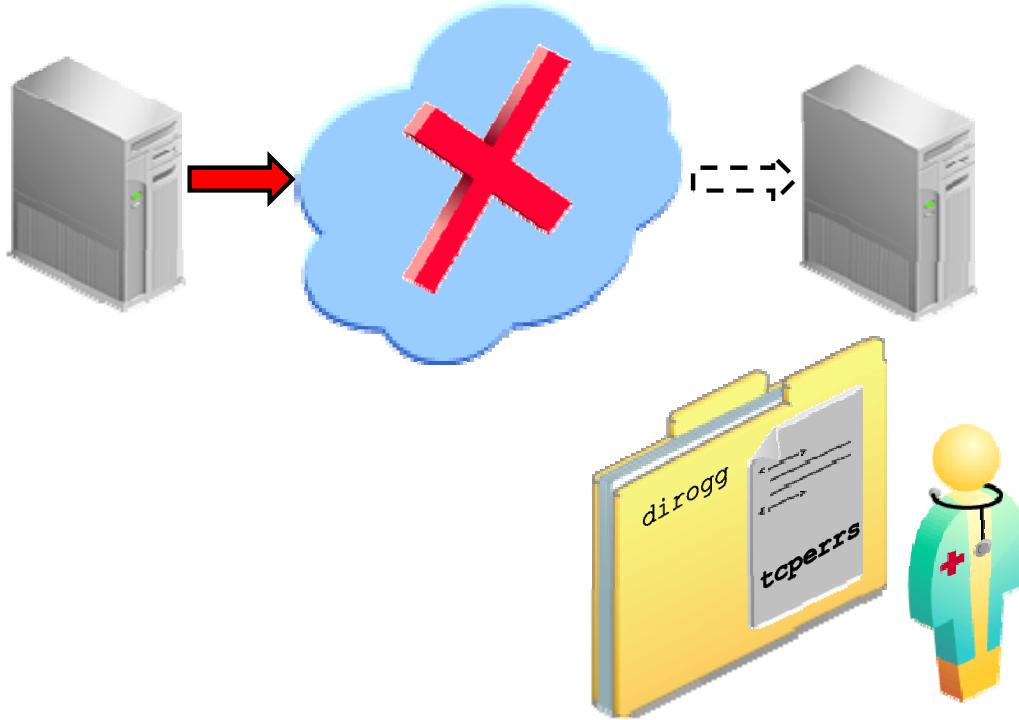
The file format for each line is:

<command_name> <command_object> <OS_group> <OS_user> <YES | NO>

To implement command security:

1. Open a new ASCII text file.
2. Create one or more security rules for each command that you want to restrict, with one rule per line. Order the rules from the most specific (those with no wildcards) to the least specific. Security rules are processed from the top of the CMDSEC file downward. The first rule that is satisfied is the one that determines whether or not access is allowed. Separate each of the following components with spaces or tabs.
3. Save the file as CMDSEC (using uppercase letters on a Linux or UNIX system) in the Oracle GoldenGate home directory.

Handling TCP/IP Errors



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

A number of issues can pertain specifically to the TCP/IP connection, including:

- Bottlenecks
- Connection refused error
- Not enough bandwidth

The `tcperrs` file (TCP Errors), which is located in the Oracle GoldenGate installation directory of the target system, can help with troubleshooting TCP/IP errors.

tcperrs File

```
# TCP/IP error handling parameters
# Default error response is abend
#
# Error          Response    Delay (csecs)  Max Retries
#
ECONNABORTED      RETRY        1000           10
#ECONNREFUSED     ABEND         0              0
ECONNREFUSED      RETRY        1000           12
ECONNRESET        RETRY        500            10
ENETDOWN          RETRY        3000           50
ENETRESET         RETRY        1000           10
ENOBUFS           RETRY        100             60
ENOTCONN          RETRY        100             10
EPIPE              RETRY        500             10
ESHUTDOWN         RETRY        1000            10
ETIMEDOUT         RETRY        1000            10
NODYNPORTS        RETRY        100             10
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

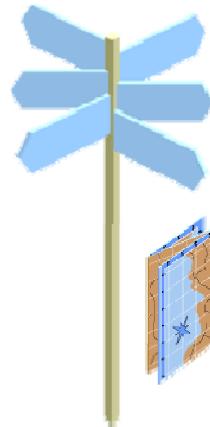
Error handling for TCP/IP connections is guided by the `tcperrs` file on the target system. It is recommended that you set the response values for the errors in this file to `RETRY`. (The default is `ABEND`.) This file also provides options for setting the number of retries and the delay between attempts. This file is in the Oracle GoldenGate directory. The `tcperrs` file contains default responses to basic errors.

The defaults are adequate to get started with a typical installation. Later, you can change the response to `RETRY` if you are having intermittent network problems.

For additional information about networking errors, see the follow-on course: *Oracle GoldenGate 11g Troubleshooting and Tuning*.

Roadmap

- Preparing the Environment
- Supplemental Logging: TranData
 - Commands to Do to the Database
 - Commands to Do to Add Transaction Data
 - Preparing the Manager
- Column Definitions: defgen



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

TranData Command

- By default, the database logs only those column values that change.
- Before Oracle GoldenGate can start capturing real-time data, the Oracle database must be set to log the table key values whenever it logs a row change. In this way, the key values are available in the redo logs.
 - This is required for Oracle GoldenGate so that it can locate the correct row on the target for update and delete operations.
 - This task is accomplished via the GGSCI command:
 - Add TranData or
 - Add SchemaTranData

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Add TranData enables supplemental logging of key values in the transaction logs whenever a row change occurs. If Add TranData is not used, updates and deletes will fail to be propagated to the target system and the Replicat may abend.

TranData and SchemaTranData do the same thing, except TranData operates at the table level (including wildcards) while SchemaTranData operates at the schema level.

Suppose that a schema named myschema has tables A, B, and C, and you run the following command:

```
GGSCI> Add TranData myschema.*
```

The result is that it adds the three tables' transaction data. Later, you create another table myschema.D, but now that table is not added to the TranData list because Add has already happened and the wildcard was already resolved.

On the other hand, suppose that you have another schema named yourschema and it has tables E, F, and G. You run the following command:

```
GGSCI> Add SchemaTranData yourschema
```

It adds the three tables just like TranData would. Later, you create another table myschema.H, and that table and all future tables for yourschema are automatically added to the TranData list.

Preparing the Environment: Oracle Database

- On the source database
 - Enable minimal supplemental logging at the database level:


```
SQL> ALTER DATABASE ADD SUPPLEMENTAL LOG DATA;
SQL> ALTER DATABASE FORCE LOGGING;
SQL> ALTER SYSTEM SWITCH LOGFILE;
```
 - Enable supplemental logging at the table level:


```
GGSCI> DBLogin UserID <login>, Password <pswd>
GGSCI> Add TranData <OWNER>. <TABLE1>
GGSCI> Add TranData <OWNER>. <TABLE2>
```
 - Enable archive logging as a secondary data source in case the online logs recycle before Extract is finished with them.
- On the target database:
 - Ensure access by Oracle GoldenGate processes.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle Logs

Oracle GoldenGate reads the online logs by default (or it reads the archived logs if it starts falling “behind”).

It is recommended that archive logging be enabled and that you keep the archived logs on the system for the longest time possible (until they are all processed). This prevents the need to resync data if the online logs recycle before all data has been processed.

Enabling and forcing minimal supplemental logging at the database level allows Oracle GoldenGate to properly capture updates to primary keys and chained rows.

The DBLogin user ID and password in clear text can be stored encrypted in the Wallet.

Wallets are covered in the lesson titled “Configuring Change Capture: Extract.” The user ID could (or should) have an Oracle SID appended as a suffix (for example, ogguser@mysid).

Preparing the Environment: Manager Overview

- The Manager provides a command-line interface to perform a variety of tasks:
 - Starting, stopping, and monitoring Oracle GoldenGate processes
 - Setting the parameters that configure Oracle GoldenGate processes
 - Error and lag reporting
 - Resource management
 - Trail file management
- The Manager process must be running on each system before Extract or Replicat can be started.
- Manager parameters are entered in the `mgr.prm` file under the `dirprm` directory.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Manager process performs system management and monitoring tasks on Windows and UNIX, including the following:

- Starting Server Collector processes to collect data from remote Extract processes
- Threshold reporting (for example, when Extract falls behind on the transaction logs)
- Purging trails

Manager Parameters

Enter the Manager parameters in the `dirprm/mgr.prm` file in the Oracle GoldenGate installation directory. A `mgr.prm` file must exist; otherwise, you get an error when you try to start the Manager. A `mgr.prm` file is created by Oracle Universal Installer (OUI). On a multi-host environment, it is recommended that you make all Manager ports unique; on a single host environment, it is *required* that you make all Manager ports unique.

Error and Informational Reporting

Manager reports critical and informational events to the `ggserr.log` file in the Oracle GoldenGate installation directory.

Preparing the Environment: Configuring Manager

- Create the parameter file by using GGSCI commands:

```
GGSCI> Edit Param Mgr
```

- Start the Manager by using GGSCI:

```
GGSCI> Start Mgr
```

- Determine which port the Manager is using:

```
GGSCI> Info Mgr
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You must start the Manager before most other configuration tasks are performed in GGSCI. This can be achieved by using either Start Manager or Start Mgr. GGSCI commands and parameter keywords are not case-sensitive. The following are all equivalent in any GGSCI session, regardless of the OS:

```
start mgr
START MGR
Start Mgr
sTaRt MgR
```

However, the report file name produced *is* case-sensitive: MGR.rpt.

On Windows, you can also start and stop the Manager through the standard Windows services control applet (in Control Panel). If GoldenGate is to be used in a cluster environment, each service in Windows must be set up on each server.

Putting Start Mgr (and some other useful commands) in a startup.oby file is a good practice.

Preparing the Environment: Sample Manager Parameter File

```
-- Created by Joe Admin on 10/11/2013
Port 7809
DynamicPortList 8001, 8002, 9500-9520
PurgeOldExtracts /ggs/dirdat/aa*, UseCheckpoints
PurgeOldExtracts /ggs/dirdat/bb*, UseCheckpoints, &
MinKeepDays 5
Autostart ER *
AutoRestart Extract *, WaitMinutes 2, Retries 5
LagReportHours 1
LagInfoMinutes 3
LagCriticalMinutes 5
```

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

This parameter file has the Manager listening on port 7809. Ports 8001 and 8002, and those ports in the range 9500 through 9520, will be assigned to the dynamic processes started by the Manager. It is not smart enough to test or skip busy ports.

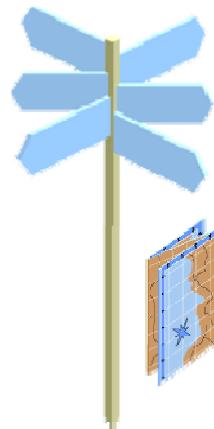
The Manager process will also recycle Oracle GoldenGate trails that match the file name of /ggs/dirdat/aa* and /ggs/dirdat/bb*. The Manager process can be configured to delete trail files after all processing by any Extracts and Replicats has completed (which is determined by the UseCheckpoints option). However, bb* trails will not be purged until there has been no activity for five days.

The Manager will automatically start any Extract and Replicat process at startup and will attempt to restart any Extract process that abends after waiting two minutes, but only up to five attempts.

The Manager will report lag information every hour, but only for processes that have three and five minutes of latency. The message will be flagged *informational* for lags of three minutes and *critical* for any process that has a lag greater than five minutes. Oracle GoldenGate Monitor can be set to watch for lag values that exceed any user-specified threshold, and then to send either an email to a group of users and/or an SNMP trap to a target.

Roadmap

- Preparing the Environment
- Supplemental Logging: TranData
- Column Definitions: defgen
 - Prepare the Source Side
 - Copy to the Target Side



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Preparing the Environment: Overview of Source Definitions

- When capturing, transforming, and delivering data across disparate systems and databases, Oracle GoldenGate must understand both the source and target layouts.
- The `defgen` utility produces a file containing a definition of the layouts of the source files and tables.
- This source definition file is used to interpret layouts for data that is stored in Oracle GoldenGate trails.
- At startup, Replicat reads the definition file specified with the `SourceDefs` parameter.

**ORACLE**

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

When capturing, transforming, and delivering data across disparate systems and databases, Oracle GoldenGate must understand both the source and target layouts. Understanding column names and data types is instrumental to Oracle GoldenGate's data synchronization functions.

The `defgen` utility program produces a file containing a definition of the layouts of the source files and tables. The output definitions are saved in an edit file and transferred to all target systems in text format. Replicat and Collector read the definitions at process startup and use the information to interpret the data from Oracle GoldenGate trails.

When transformation services are required on the source system, Extract can use a definition file containing the target layouts rather than source layouts.

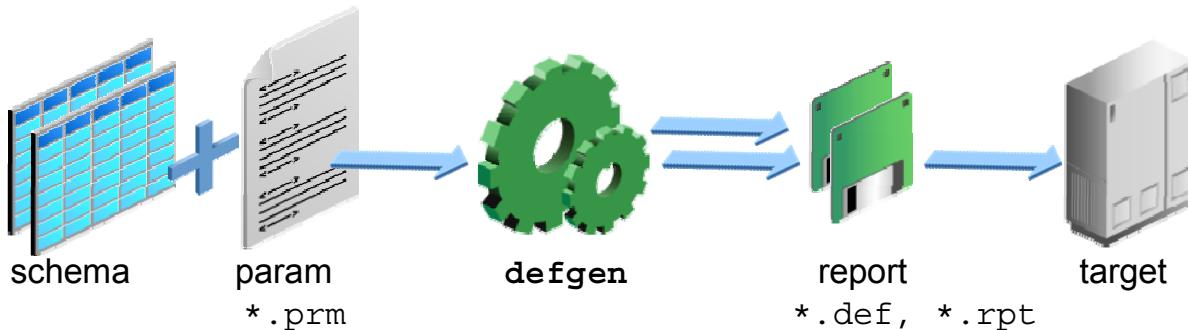
Note: The user should never modify the `defgen` output.

Preparing the Environment: Running defgen

- defgen is initiated from the command prompt:

```
defgen paramfile <paramfile> [ reportfile <reportfile> ]
```

- Definitions are saved to the file that is specified in the parameter file. This file must be transferred to the target system as a text file.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

UNIX Example

```
cd <OGG_HOME>
./defgen paramfile dirprm/defgen.prm reportfile
dirrpt/defgen.rpt
```

Windows Example

```
cd <OGG_HOME>
defgen paramfile dirprm\defgen.prm reportfile
dirrpt\defgen.rpt
```

In this syntax, `<OGG_HOME>` is the Oracle GoldenGate installation directory (for example, `/home/user01/ogg` or `C:\OGG`).

Character Set Configuration (Optional)

Use the `CharSet` parameter to generate the definitions file in the specified character set:

```
DefsFile ./dirdef/source.def, Purge, CharSet UTF-8
```

Add the `UpdateCS` option:

```
[OS prompt] defgen ./dirdef/source.def UpdateCS ISO-8859-1
```

Sample defgen Output

```

*+- Defgen version 2.0, Encoding UTF-8
*
* Definitions created/modified 2013-04-13 11:36
*
* Field descriptions for each column entry:
*
*   1    Name
*   2    Data Type
*   3    External Length
*   4    Fetch Offset
*   5    Scale
*   6    Level
*   7    Null
*   8    Bump if Odd
*   9    Internal Length
*  10   Binary Length
*  11   Table Length
*  12   Most Significant DT
*  13   Least Significant DT
*  14   High Precision
*  15   Low Precision
*  16   Elementary Item
*  17   Occurs
*  18   Key Column
*  19   Sub Data Type
*
Database type: ORACLE
Character set ID: UTF-8
National character set ID: UTF-16
Locale: neutral
Case sensitivity: 14 14 14 14 14 14 14 14 14 14 14 14 14 11 14 14 14
*
Definition for table WEST.ACCTOUNT
Record length: 58
Syskey: 0
Columns: 2
ACCOUNT_NUMBER    134      11          0 0 0 1 0      8      8      8 0 0 0 0 0
1 0 1 3
ACCOUNT_BALANCE   64       40          12 2 0 1 0     40     40     40 0 0 0 0 0
1 0 0 3
End of definition
*
Definition for table WEST.BRANCH
Record length: 24
Syskey: 0
Columns: 2
BRANCH_NUMBER    134      11          0 0 0 1 0      8      8      8 0 0 0 0 1
0 1 3
BRANCH_ZIP       134      8           12 0 0 1 0      8      8      8 0 0 0 0 1
0 0 3
End of definition

```

Quiz

Add TranData enables supplemental logging of key values in the transaction logs whenever a row change occurs.

- a. True
- b. False



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: a

If you need additional columns replicated other than just the key columns, you can specify them by using the `Cols` attribute.

Quiz

Users of both Extract and Replicat require the DBA role for the database.

- a. True
- b. False



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Although the DBA role will certainly work for Extract and Replicat, you do not require quite that much privilege.

Summary

In this lesson, you should have learned how to:

- Describe the high-level configuration steps
- Configure and start the Manager process
- Generate a source definitions file
- Prepare a source database for transaction capture



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice 5 Overview: Configuration Overview and Preparing the Environment

This practice covers the following topics:

- Preparing your environment
- Creating the GLOBALS parameter file
- Configuring and starting Oracle GoldenGate Manager
- Using the TranData option



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Configuring Change Capture: Extract

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

ORACLE®

Objectives

After completing this lesson, you should be able to:

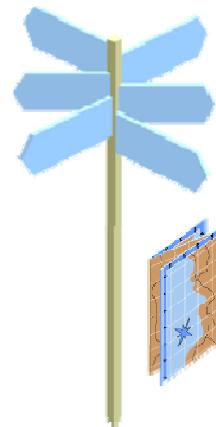
- Describe what the Extract process does
- Configure, start, and stop an Extract process
- Configure, start, and stop an Extract data pump
- Add local and remote trails
- Configure Extract to access logs on Oracle Automatic Storage Management (ASM)



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Roadmap

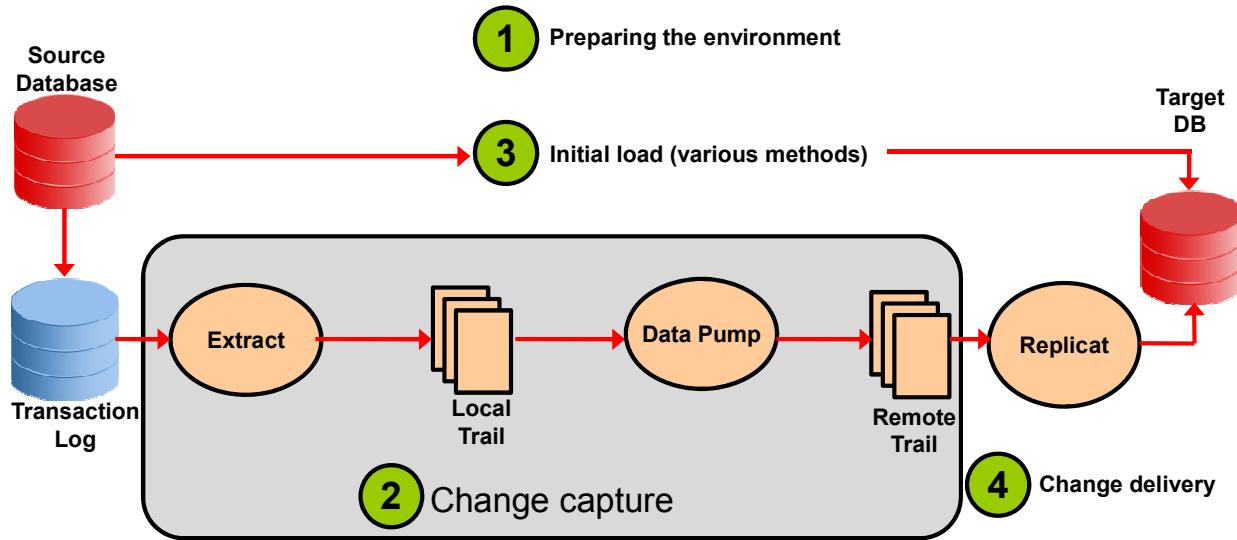
- Extract
 - Primary
 - Secondary
- Data Pump
- Starting Process Group
- Trails
- ASM



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Step 2: Change Capture



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Recall from the preceding lesson the steps for configuring Oracle GoldenGate:

1. Preparing the environment
2. Change capture (Extract)
3. Initial load
4. Change delivery (Replicat)

The graphic in this slide represents a high-level overview of Step 2: "Change capture." Each component in the gray box (labeled "2. Change capture") in the slide diagram is discussed in detail in this lesson.

The data pump is optional but recommended as a best practice.

Disclaimer: Normally, you would make sure that the target tables are ready to receive the new transactions. If both source and target are empty tables, there is nothing additional to do.

If you choose to use a homogeneous database-specific utility to do the initial table copy, there is nothing for Oracle GoldenGate to do. If you want to do the initial table load with Oracle GoldenGate, you normally do a one-time initial load. Because initial load is done only once, and even then optionally, it will be presented *after* the ongoing capture and delivery. This is not the normal sequence that you would run in a production shop, but it makes the teaching and understanding easier.

Extract: Overview

- Extract captures all the changes that are made to objects that you configure for synchronization.
- When a transaction is committed, Extract sends the data for that transaction to the trail for propagation to the target system.
- A primary Extract can be configured to:
 - Capture changed data from database logs or archive logs
 - Capture changed data from JMS
 - Capture data directly from source tables for initial data load
 - Send the data to be written to a local or remote trail or file
- A secondary Extract, called a *data pump*, can be configured to distribute data from local trails to remote systems.

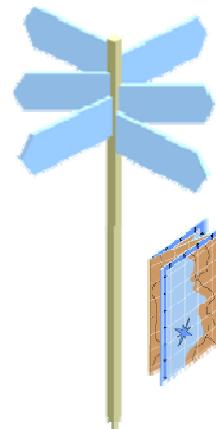


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Extract captures all the changes that are made to objects that you configure for synchronization. Extract stores the changes until it receives commit records or rollbacks for the transactions that contain them. When a rollback is received, Extract discards the data for that transaction. When a commit is received, Extract sends the data for that transaction to the trail for propagation to the target system. All the log records for a transaction are written to the trail as a sequentially organized transaction unit. This design ensures both speed and data integrity.

Roadmap

- Extract
- Data Pump
 - Overview
 - Topologies
- Starting Process Group
- Trails
- ASM



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Data Pump: Overview

- An Extract data pump can write to one or multiple remote trails and/or servers (not to be confused with Oracle Data Pump).
- A data pump:
 - Reads the local trail
 - Manipulates the data or passes it through without change
 - Sends the data to one or more targets
- A data pump is useful:
 - As a safeguard against network and target failures
 - To break complex data filtering and transformation into phases
 - To consolidate data from many sources
 - To synchronize one source with multiple targets



ORACLE

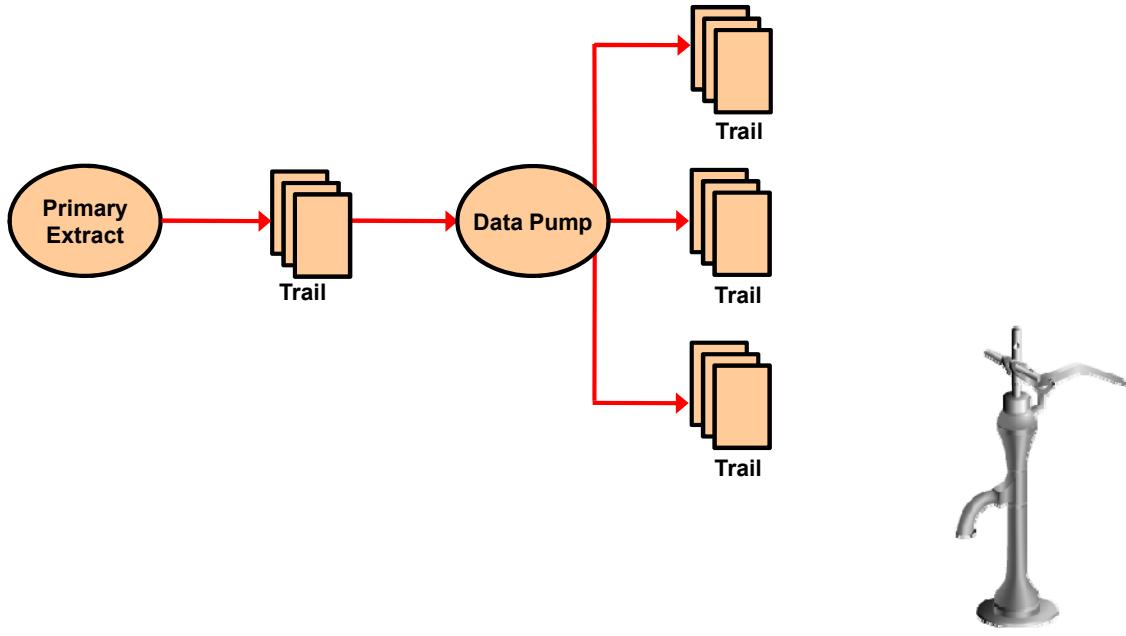
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

For most business cases, it is a best practice to use a data pump. Some reasons for using a data pump include the following:

- **Protection against network and target failures:** In a basic GoldenGate configuration, with only a trail on the target system, there is no space on the source system to store the data that Extract continuously extracts into memory. If the network or the target system becomes unavailable, the primary Extract could run out of memory and ABEND. However, with a trail and data pump on the source system, captured data can be moved to disk, preventing the ABEND. When connectivity is restored, the data pump extracts the data from the source trail and sends it to the target systems.
- **Breaking down complex data filtering and transformation phases:** You can configure a data pump to perform the first transformation either on the source system or on the target system, and then use another data pump or the Replicat group to perform the second transformation.

- **Consolidating data from many sources to a central target:** You can store extracted data on each source system and use data pumps on each system to send the data to a trail on the target system. Dividing the storage load between the source and target systems reduces the need for massive amounts of space on the target system to accommodate data arriving from multiple sources.
- **Synchronizing one source with multiple targets:** When sending data to multiple target systems, you can configure data pumps on the source system for each one. If network connectivity to any of the targets fails, data can still be sent to the other targets.

Data Pumps: One-to-Many Trails

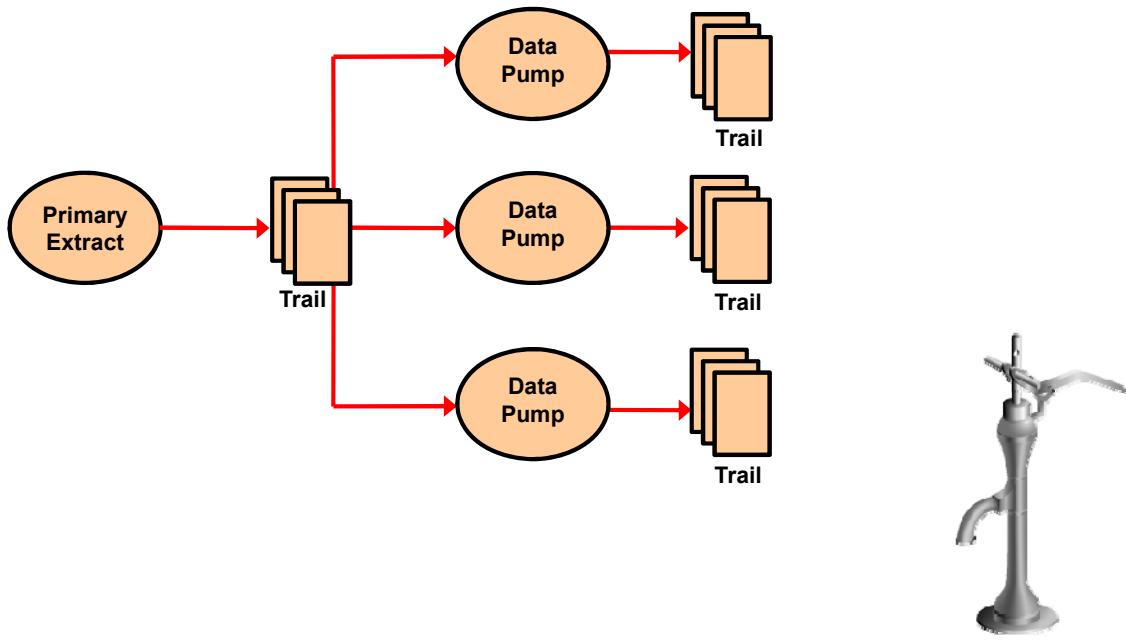


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

A data pump can be set up to duplicate or selectively route the data to multiple trails. However, if the trails are on multiple target systems and the communication to one of the systems goes down, the Extract may exhaust its retries and shut down, causing the updates to all targets to stop.

Data Pumps: One-to-Many Target Systems



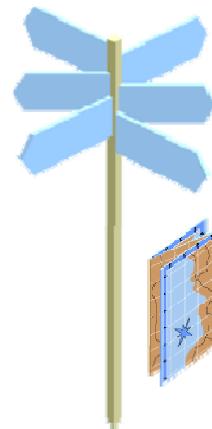
ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate supports synchronization of a source database to any number of target systems. For this configuration, Oracle Corporation recommends using data pump Extract groups to ensure that if network connectivity to any of the targets fails, data can still be sent to the other targets.

Roadmap

- Extract
- Data Pump
- Starting Process Group
 - Add Extracts
 - Create Wallet Aliases
 - Passive Alias Extracts
 - Converting to (and from) Integrated
- Trails
- ASM



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Setting Up Change Capture (Extract)

On the source system:

1. Add a primary Extract. This Extract reads from the transaction logs located on the source and has an associated parameter file.
2. (Optional) Add a local trail and a data pump Extract that reads from the transaction logs located on the source and has an associated parameter file.
3. Add a remote trail.
4. Start the Extract processes.

(The target system is a Replicat, which is covered later.)



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To configure Extract to capture changes from transaction logs:

- 1a. Set up a parameter file for Extract with the GGSCI `Edit Param` command.
- 1b. Set up an initial Extract checkpoint in the logs with the GGSCI `Add Extract` command.
 - Alternatively, the Extract can be registered using the `LogRetention` option, which enables an Extract group in classic capture mode to work with Oracle Recovery Manager (RMAN) to retain the logs that Extract needs for recovery.
2. (Optional) Create a local trail by using the GGSCI `Add ExtTrail` command and a data pump Extract (and parameter file) reading from the local trail.
3. Set up a remote trail by using the GGSCI `Add RmtTrail` command.
- 4a. Start the Server Collector process on the target system, or let the Manager start the Server Collector dynamically.
- 4b. Start Extract by using the GGSCI `Start Extract` command, as in this example:
`GGSCI> Start Extract finance`
 - GGSCI sends this request to the Manager process, which in turn starts Extract. Manager monitors the Extract process and restarts it, when appropriate, if it goes down.

Add Extract Command

- For a regular, passive, or data pump Extract, use the GGSCI Add Extract command. (This automatically creates a checkpoint.)

```
GGSCI> Add Extract <group_name>
      , <data_source_options>
      , <starting_point>
      [, <processing options>]
```

- For an alias Extract:

```
GGSCI> Add Extract <group_name>
      , RmtHost {<host_name> | <IP address>}
      , {MgrPort <port>} | {Port <port>}
      [, RmtName <name>]
      [, Desc "<description>"]
```

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Passive Alias Extract is covered in more detail in later slides.

Add Extract: Examples

```
GGSCI > DBLogin UserID myuser, Password mypswd

GGSCI 1> Add Extract finance, TranLog, Begin Now
EXTRACT added.

GGSCI 2> Add Extract atms, TranLog, Begin 2013-01-31 08:00
EXTRACT added.

GGSCI 3> Add Extract pump, ExtTrailSource c:\ggs\dirdat\lt
EXTRACT added.

GGSCI 4> Add Extract load, SourceIsTable
EXTRACT added.

GGSCI 5> Info All
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

All of these examples presume that a `DBLogin` has already been done.

1. Create an Extract group named `finance` that extracts database changes from the transaction logs. Start extracting with records generated at the time that you add the Extract group.
2. Create an Extract group named `atms` that extracts database changes from the transaction logs. Start extracting with records generated at 8:00 AM on January 31, 2013.
3. Create a data pump Extract group named `pump` that reads from the Oracle GoldenGate Windows trail `c:\ggs\dirdat\lt`.
4. Create an initial-load Extract named `load`.
5. After you create all of these Extracts, it is customary to enter `Info All` to see if they are all present. They should be `Status=STOPPED` at this point.

Editing Extract Parameters

- To edit a parameter file for an online Extract group, issue the following command:

```
GGSCI> Edit Params <group name>
```

- Then modify the following parameters in the file:

```
--Some Comment Goes Here.  
Extract extwest  
ExtTrail ./dirdat/ew  
UserID ogguser@mysid, Password oracle_4U  
--UserIDAlias oggalias  
TranLogOptions ExcludeUser ogguser  
Table WEST.*;  
Table SALES.INVENTORY;
```

 **ORACLE**

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In Linux, the default GGSCI editor opens `vi`. You can change that by entering:

```
GGSCI> Set Editor gedit
```

Or you can choose any other editor. That editor choice stays in effect only for that session. If you exit GGSCI and restart it, you have to enter the `set editor` command again.

The example in the slide shows a primary local Extract. A companion data pump is not shown.

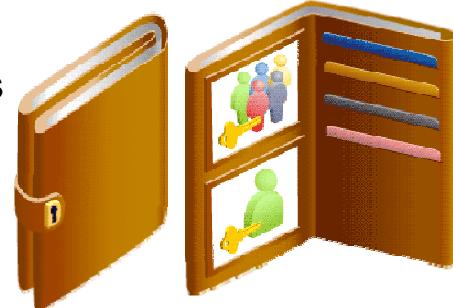
Although the example in the slide shows `UserID` and `Password` stored in clear text (obviously a security concern), you have the option to store `UserID` and `Password` in the Oracle Wallet and have only an alias configured in the parameter file. See the lesson titled “Configuration Options” for additional information about other uses of the Wallet for encryption.

The `$ORACLE_SID` variable needs to be set somewhere. It could be set in the Extract parameter file, in the alias, in the `.bashrc` profile, or by using `oraenv`. There is no harm in setting it twice.

User ID and Password Aliases (New with 12c)

- Oracle Wallet:

- Contains user IDs and passwords
 - Is used in Extracts, ASM, and so on



```
GGSCI> Create Wallet
GGSCI> Add CredentialStore
GGSCI> Alter CredentialStore Add User oggadmin
      Password Welcome1 Alias oggalias
GGSCI> Info CredentialStore
GGSCI> DBLogin UserIDAlias oggalias
```

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Info command lists the user IDs and aliases, but not the passwords.

The following are the additional commands that are not shown in the slide:

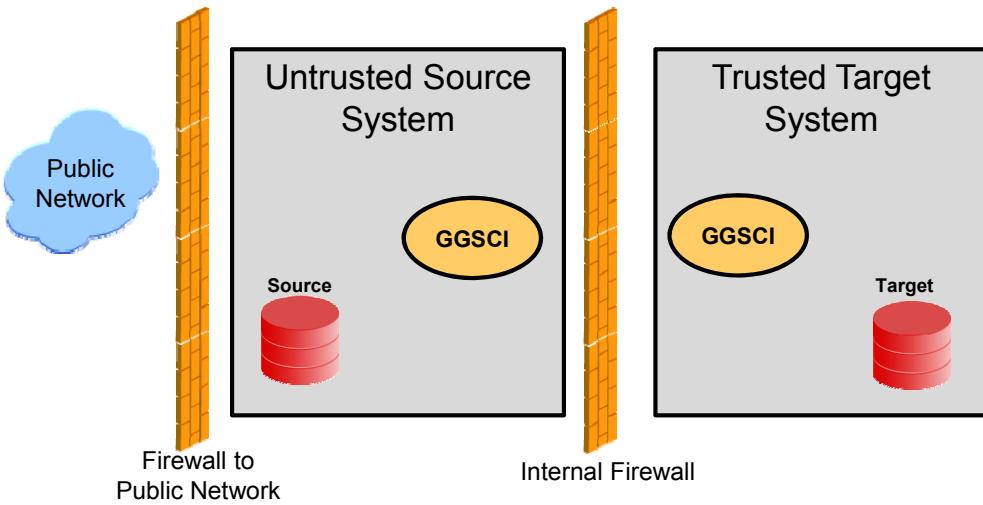
```
GGSCI> Alter CredentialStore Replace User userid ...
GGSCI> Alter CredentialStore Delete User userid ...
```

Optionally, you can add “Domain *mydom*” at the end of each of the Alter commands to maintain multiple groups of credentials in the same wallet. As another option, the *userid* can include an Oracle SID in the form of @*mysid* as a suffix.

The parameter file on the previous page would now look like:

```
Extract extwest
ExtTrail ./dirdat/ew
UserIDAlias oggalias
TranLogOptions ExcludeUser ogguser
Table WEST.*;
Table SALES.INVENTORY;
```

Passive Alias Extract



ORACLE®

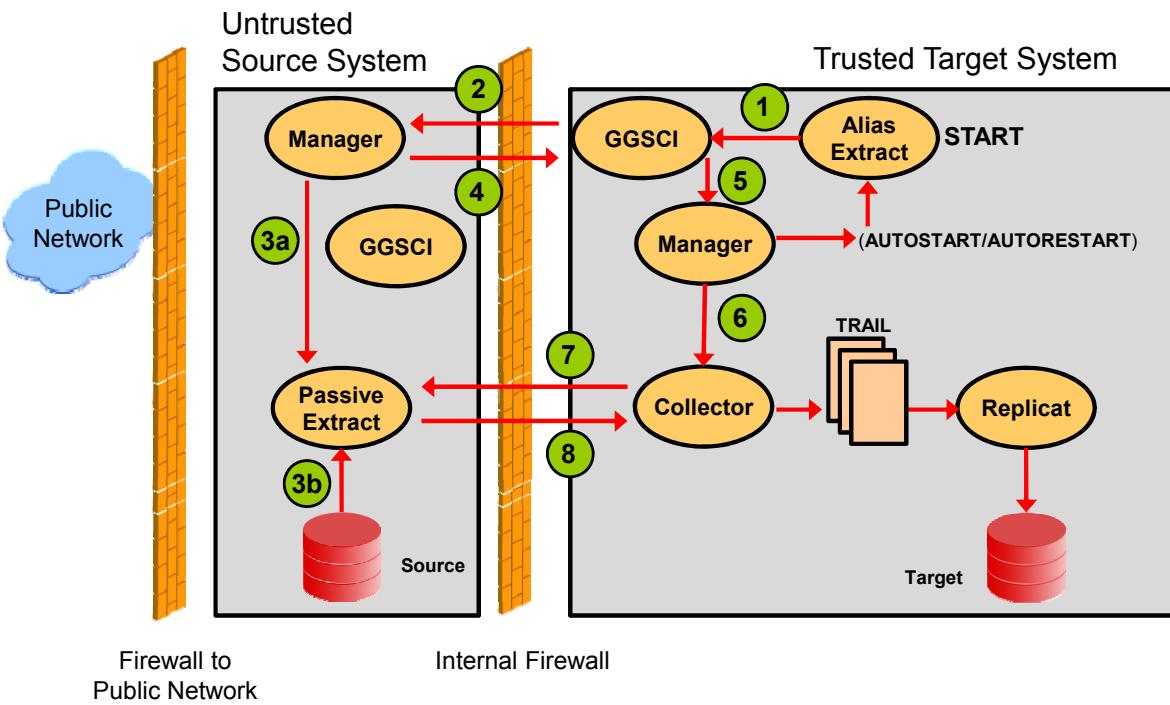
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

When a target system resides inside a trusted intranet zone, initiating connections from the source system (the standard Oracle GoldenGate method) may violate security policies if the source system is in a less trusted zone. It also may violate security policies if a system in a less-trusted zone contains information about the ports or IP address of a system in the trusted zone, such as that normally found in an Oracle GoldenGate Extract parameter file. In this kind of intranet configuration, you can use a passive-alias Extract configuration.

Connections are initiated from the target system inside the trusted zone by an alias Extract group, which acts as an alias for a regular Extract group on the source system, known in this case as the passive Extract. After a connection between the two systems is established, data is processed and transferred across the network by the passive Extract group in the usual way.

Unlike a Primary Extract group, the alias Extract group on the trusted target does not perform any data processing activities. Its sole purpose is to initiate and terminate connections to the less trusted source. In this capacity, the alias Extract group does not use a parameter file nor does it write processing checkpoints. A checkpoint file is created for this group, but it is used only to determine whether the passive Extract group is running or not and to record information required for the remote connection.

Passive Alias Extract



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

1. An Oracle GoldenGate user starts the alias Extract on the trusted system, or an Autostart or AutoRestart parameter causes it to start.
2. GGSCI on the trusted system sends a message to the Manager on the less-trusted system to start the associated passive Extract. The host name or IP address and port number of the Manager on the trusted system are sent to the less-trusted system.
3. On the less-trusted system, the Manager finds an open port (according to rules in the DynamicPortList Manager parameter) and starts the passive Extract, which listens on the specified port.
4. The Manager on the less-trusted system returns that port to GGSCI on the trusted system.
5. GGSCI on the trusted system sends a request to the Manager on that system to start a Collector process on that system.
6. The target Manager starts the Collector process and passes it the port number where Extract is listening on the less-trusted system.
7. Collector on the trusted system opens a connection to the passive Extract on the less-trusted system.
8. Data is sent across the network from the passive Extract to the Collector on the target and is written to the trail in the usual manner for processing by Replicat.

Converting to Integrated Capture

- Convert to Integrated:

```
GGSCI> DBLogin UserIDAlias <useralias>
GGSCI> Register Extract my_capt Database
GGSCI> Stop Extract my_capt
GGSCI> Alter Extract my_capt, Upgrade Integrated Tranlog
GGSCI> Start Extract my_capt
```

- In the parameter file:

```
--Extract my_capt
:
TranLogOptions IntegratedParams (max_sga_size 900, &
    parallelism 3)
:
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

After you register the extract, GGSCI reports back an SCN number. The extract has to proceed past that number before the extract can be stopped and altered to Integrated mode. In a busy database, that happens in a few seconds on its own. In a quiet database, you need to force the SCN to increment by doing some trivial DML through the Extract, doing a commit, making sure it replicated, and *then* issuing the command to Stop and Alter the Extract.

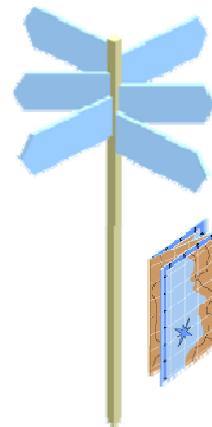
To convert from integrated capture back to Classic mode:

```
GGSCI> DBLogin UserId <user>, Password <pswd>
GGSCI> Stop Extract my_capt
GGSCI> Unregister Extract my_capt Database
GGSCI> Alter Extract my_capt, Downgrade Integrated Tranlog
GGSCI> Start Extract my_capt
```

There are many other possible topology scenarios (covered in the follow-on advanced course) with downstream capture and logmining.

Roadmap

- Extract
- Data Pump
- Starting Process Group
- Trails
 - Local
 - Remote
 - Data Pumps (Part II)
- ASM

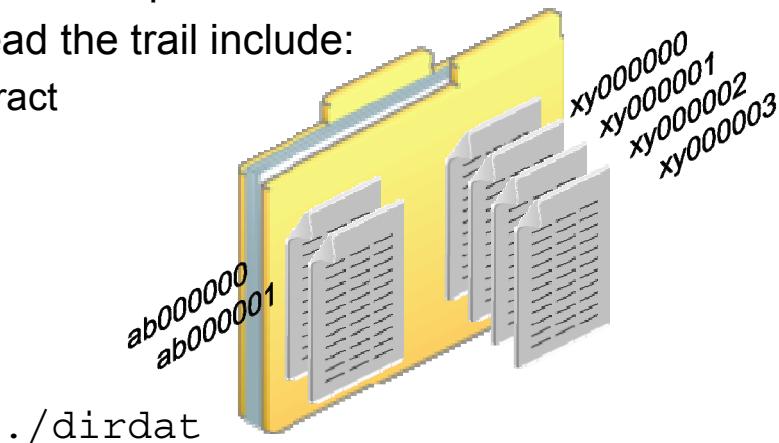


ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Overview of Trails

- Trails are used in Oracle GoldenGate to support the continuous extraction and replication of database changes.
- A trail can exist on the source or target system, or on an intermediary system.
- Only one primary Extract process writes to a trail.
- Processes that read the trail include:
 - Data pump Extract
 - Replicat



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Trails are stored in the `dirdat` subdirectory of the Oracle GoldenGate directory. When created, all file names in a particular trail begin with the same two characters. Then, as additional files are needed, each name is appended with a unique six-digit serial number.

Adding a Local or Remote Trail

- Add a local or remote trail with the GGSCI command:

```
GGSCI> Add ExtTrail | RmtTrail <trail_name>
      , Extract <group_name>
[, Megabytes <n>]
```

- If you are using a data pump:
 - The primary Extract needs a local trail (ExtTrail)
 - The data pump Extract needs a remote trail (RmtTrail)

```
GGSCI> Add ExtTrail c:\ggs\dirdat\aa,
      Extract finance, Megabytes 10
GGSCI> Add RmtTrail c:\ggs\dirdat\bb,
      Extract parts, Megabytes 5
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

On the local system, a trail is known as an *Extract trail* (or *local trail*). On a remote system, it is known as a *remote trail*. The primary Extract writes to a local trail. The data pump Extract sends changes to a remote trail. The default trail file size is 100 MB. In this example, the parts Extract file is set to 5 MB.

Starting the Extract

- Start an Extract process with the GGSCI command:

```
GGSCI> Start Extract <group_name>
```

```
GGSCI> Info Extract <group_name>
```

- If the output trail is remote, this normally triggers the target Manager process to start a Server Collector process with default parameters.



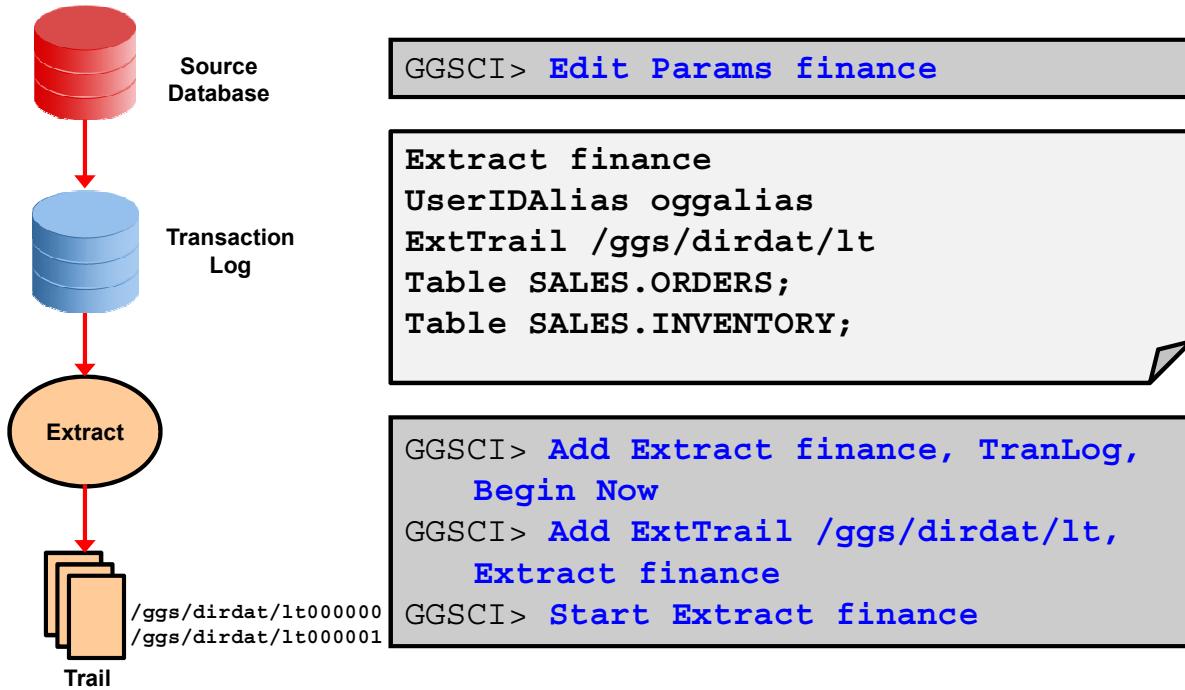
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You will not know if the `Start` fails until you run the `INFO ALL` command. You should look for a `Status` of `RUNNING`.

It is possible for users to start a Server Collector statically and modify the parameters, but this option is rarely used.

Primary Extract Configuration for Oracle



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The primary Extract makes a local Extract trail (lt). Although you could locate the local Extract trail file anywhere, by convention it is usually located in the Oracle GoldenGate installation directory. If you are following the convention, the path is simply ./dirdat to indicate relative to the installation directory.

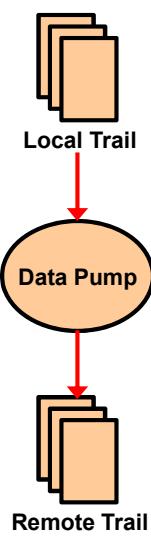
Make sure that you are logged in to the database before trying to register a new Extract:

```
GGSCI> DBLogin UserID myusername, Password mypasswd
```

As mentioned earlier, you have the option to store UserIDs and Passwords in the Oracle Wallet and have only an alias configured in the parameter file. If you did not use an alias, the syntax for that second parameter line is:

```
UserID gguser, Password mypswd
```

Data Pump Configuration for Oracle



```
GGSCI> Edit Params mypump
```

```
Extract mypump
Passthru
RmtHost <target>, MgrPort <port>
RmtTrail ./dirdat/rt
Table SALES.ORDERS;
Table SALES.INVENTORY;
```

```
GGSCI> Add Extract mypump,
ExtTrailSource ./dirdat/lt
GGSCI> Add RmtTrail ./dirdat/rt,
Extract mypump
GGSCI> Start Extract mypump
GGSCI> Info Extract mypump
```

ORACLE

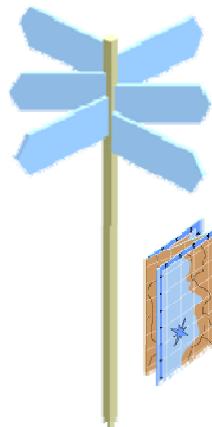
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The `Passthru` parameter is used on a data pump if you do not need to perform any data transformations or user exit processing. Add the data pump Extract with a local trail (`lt`) as the source, and add the remote trail (`rt`) as the destination.

Notice that no `UserID` login is needed in `Passthru` mode.

Roadmap

- Extract
- Data Pump
- Starting Process Group
- Trails
- ASM
 - Definition
 - Connectivity

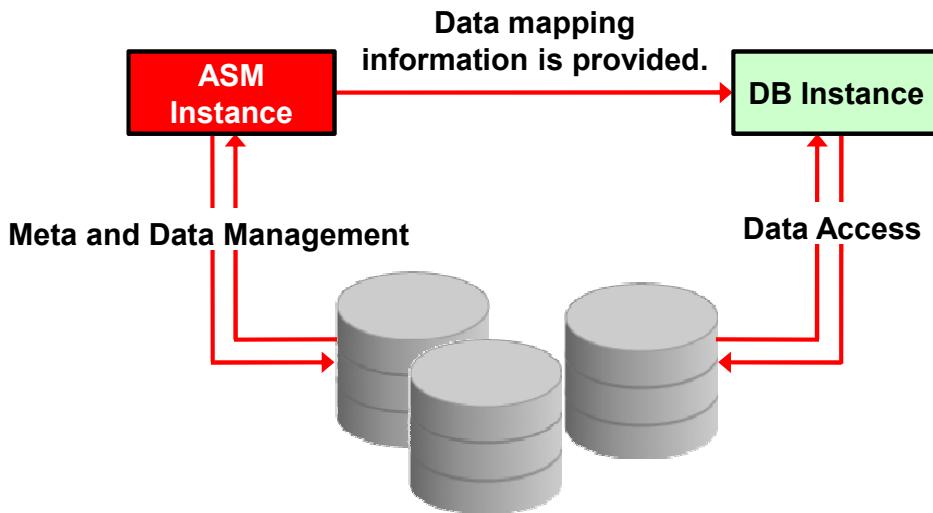


ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Automatic Storage Management (ASM)

Automatic Storage Management (ASM) enables a disk group to be designated for Oracle database files, control files, and backup files.



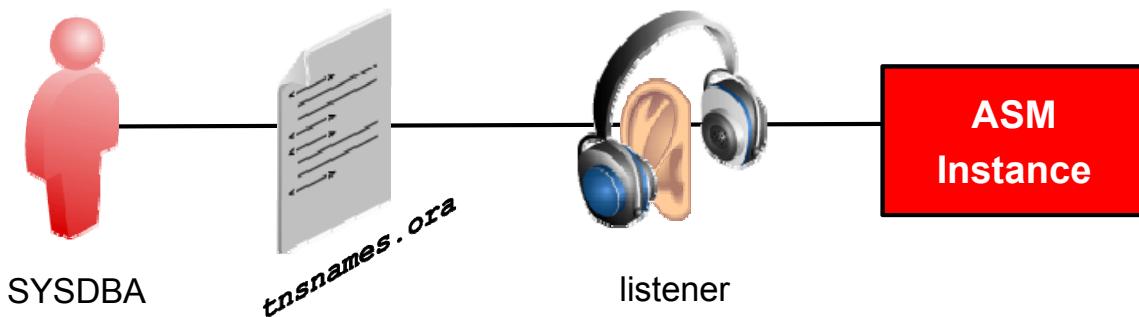
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

ASM enables a disk group to be designated as storage for Oracle database files, control files, and backup files. A disk group consists of multiple disk drives and is managed as a unit by ASM. Any file stored in a disk group is automatically striped over all the disk drives in the group. ASM provides clustering capabilities and is available not only for single SMP machine, but across multiple nodes of Oracle Real Application Clusters. To generate maximum performance, ASM automatically and dynamically distributes I/O loads among all the disk drives, even if the data usage pattern is rapidly changing.

Ensuring ASM Connectivity

To connect Oracle GoldenGate to an ASM instance:



```
TranLogOptions  
  ASMUser SYS@<ASM_instance>, ASMPassword <password>  
  ...
```

ORACLE®

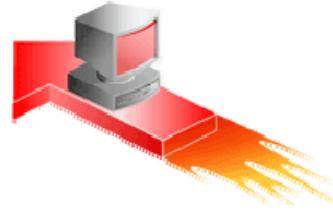
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To ensure that Oracle GoldenGate can connect to an ASM instance, verify the following:

- A user with SYSDBA privileges in the ASM instance must be used.
- Confirm that the ASM instance is listed in the `tnsnames.ora` file.
- Confirm that the Oracle listener is listening for new connections to the ASM instance.
- Use the `TranLogOptions` parameter with the `ASMUser` and `ASMPassword` options for ASM.

ASM and DBLogReader

- For Classic Extract only
- Causes newer API to be used
- Potential performance improvements



```
TranLogOptions
```

```
...
```

```
DBLogReader, DBLogReaderBufSize nnn ...
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You may use the `DBLogReader` option with Oracle Database to cause Extract to use a newer ASM API that is available as of Oracle Database 10.2.0.5 and later 10g R2 versions, and with Oracle Database 11.2.0.2 and later 11g R2 and 12c versions (but not in Oracle Database 11g R1 versions).

This API uses the database server to access the redo and archive logs. The database must contain the libraries that contain the API modules and must be running.

When used, `DBLogReader` enables Extract to use a read size of up to 4 MB. This is controlled with the `DBLogReaderBufSize` option. The maximum read size when using the default OCI buffer is 28672 bytes. The size is controlled by the `ASMBufSize` option.

A larger buffer may improve the performance of Extract when the redo rate is high.

When using `DBLogReader`, do not use the `ASMUser` and `ASMPassword` options of `TranLogOptions`. The API uses the user and password that are specified with the `UserID` parameter.

More information about ASM can be found in the Oracle Database 11g and 12c documentation at <http://www.oracle.com/technetwork/products/cloud-storage/index.html>.

Discussion Questions

1. What does Extract do?
2. Where does Extract capture transactional changes?
3. What parameters tell Extract where to send data?
4. What commands are used to create and start an Extract group?
5. What command option is used to set the maximum size of an Oracle GoldenGate trail file before it rolls to the next file?



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answers

1. Extract captures incremental changes from database transaction logs. It can also save source data from the tables themselves or other Oracle GoldenGate trails. It writes the captured data to Oracle GoldenGate trails or files.
2. From transaction logs (or archive logs) (except for Teradata)
3. ExtTrail, ExtFile
RmtHost with RmtTrail, RmtFile, or RmtTask
4. Edit Params
Add Extract

```
        Add {ExtTrail | RmtTrail | ExtFile | RmtFile}  
        Start Extract
```
5. The Megabytes <megabytes> option in the Add ExtTrail or Add RmtTrail command

Discussion Questions

6. What is a data pump?
7. What is the advantage of using a data pump?
8. Why might you use multiple data pumps for one source trail?
9. What parameter is used to identify the remote target system?
10. What other parameter is commonly used on data pumps?



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answers

6. A secondary Extract process that reads from a local trail and distributes that data to a remote system
7. Allows a local trail on the source system, which is useful for recovery if the network or target system fails
8. To send to multiple target systems (so if one goes down, they do not all go down); to separate out different tables; for parallel processing (faster)
9. RmtHost is used to identify the name or IP address of the remote system and the port that is being used.
10. The PASSTHRU parameter is used on a data pump (unless you need to perform data transformation or user exit processing).

Summary

In this lesson, you should have learned how to:

- Configure and start an Extract process
- Add local and remote trails
- Configure and start a data pump
- Describe the parameters to connect to an ASM instance



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice 6 Overview: Configuring Change Capture

This practice covers the following topics:

- Setting up the Extract and ExtTrail
- Setting up an Extract data pump and the remote trail
- Starting the two Extracts



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Configuring Change Delivery: Replicat

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

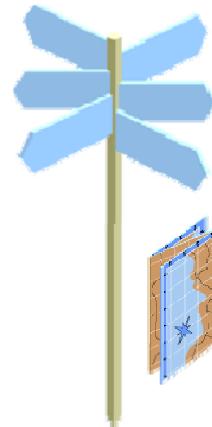
- Explain what the Replicat process does
- Configure and start a Replicat process
- Configure Replicat to handle collisions between changes and the initial load data
- Troubleshoot a started solution



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Roadmap

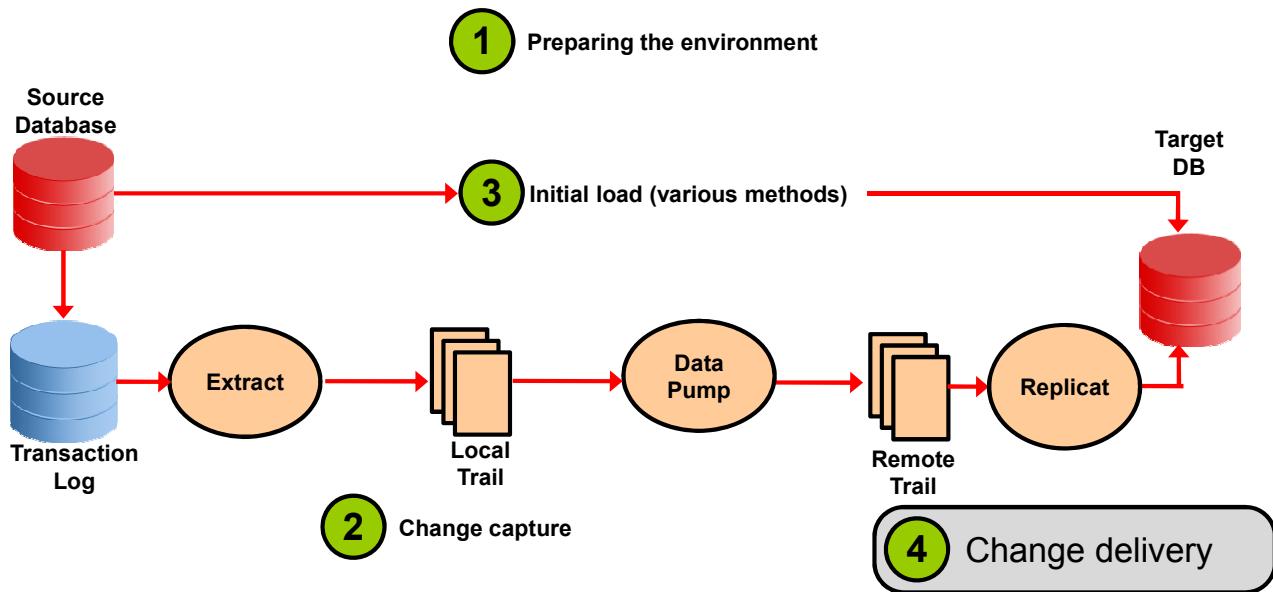
- Replicat
 - Diagram
 - Overview
 - Classic
 - Integrated
 - Coordinated
- Checkpoint
- Initial Load
- Converting to Integrated
- Troubleshooting



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Step 4: Change Delivery (Replicat)



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Step 4 in the GoldenGate configuration process is change delivery. (For the purposes of this course, we skip the optional Step 3.)

After the initial table data is loaded, you are ready to deliver just the changes.

Disclaimer

Normally, you would make sure that the target tables are ready to receive the *new* transactions. If both source and target are empty tables, there is nothing additional to do.

If you choose to use a homogeneous database-specific utility to do the initial table copy (for example SQL*Loader or RMAN DUPLICATE), there may be nothing for Oracle GoldenGate to do. If you want to do the initial table load with Oracle GoldenGate, you would normally do a one-time initial load. Because that initial load is done only once, and even then optionally, it will be presented *after* the ongoing capture and delivery. This is not the normal sequence that you would run in a production shop, but it makes the teaching and understanding easier.

Replicat: Overview

- The Replicat process runs on the target system.
- Multiple Replicat processes can be used with multiple Extract processes in parallel to increase throughput.
- Replicat can:
 - Read data out of Oracle GoldenGate trails
 - Perform data filtering by table, row, or operation
 - Perform data transformation
 - Perform database operations just as your application performed them



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

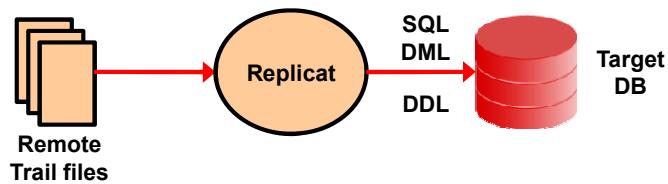
Oracle GoldenGate trails are temporary queues for the Replicat process. Each record header in the trail provides information about the database change record. Replicat reads these trail files sequentially and then processes `INSERTS`, `UPDATES`, and `DELETES` that meet your criteria. Alternatively, you can filter out the rows that you do not want to deliver, as well as perform data transformation before applying the data.

Replicat supports a high volume of data-replication activity. As a result, network activity is block based, not record-at-a-time. Replicat uses native calls to the database for optimal performance. You can configure multiple Replicat processes for increased throughput.

When replicating, Replicat preserves the boundaries of each transaction so that the target database has the same degree of integrity as the source. Small transactions can be grouped into larger transactions to improve performance. Replicat uses a checkpointing scheme, so changes are processed exactly once. After a graceful stop or a failure, processing can be restarted without repetition or loss of continuity.

Replicat “Classic”

- Trail files contain OGG canonical format statements.
- Data applied through “normal” SQL-type channels
- Available for any supported platform, multiple vendors
- Heterogeneous, that is, source and target could be almost any database



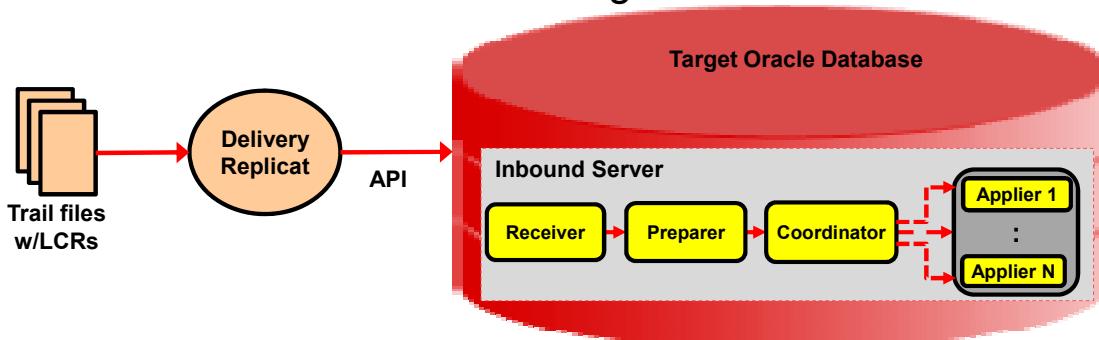
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

- Trail files and the ASCII canonical format are covered in the lesson titled “Extract Trails and Files.”
- The universal trail files are used to construct SQL DML and/or DDL statements customized to the target database vendor syntax. The SQL statements then apply the data. The target database often is not aware that the data is coming from Oracle GoldenGate; it appears to be regular plain old SQL commands as if a user typed them in.
- DDL replication, if enabled, is supported on either an Oracle or a Teradata environment.
- This mode is available on almost every OEM platform (including Oracle) and every version (with some rare exceptions). You can start a Replicat in Classic and then later convert it to Integrated if the platform permits it.

Replicat “Integrated” a.k.a. “Integrated Delivery” (New with 12c)

- Trail files contain data in either canonical or LCR format.
- Data applied through API, very efficient, bypasses overhead
- Available for only latest Oracle Database platforms
- Source can be anything, target must be Oracle database.
- Source can be Classic or Integrated.

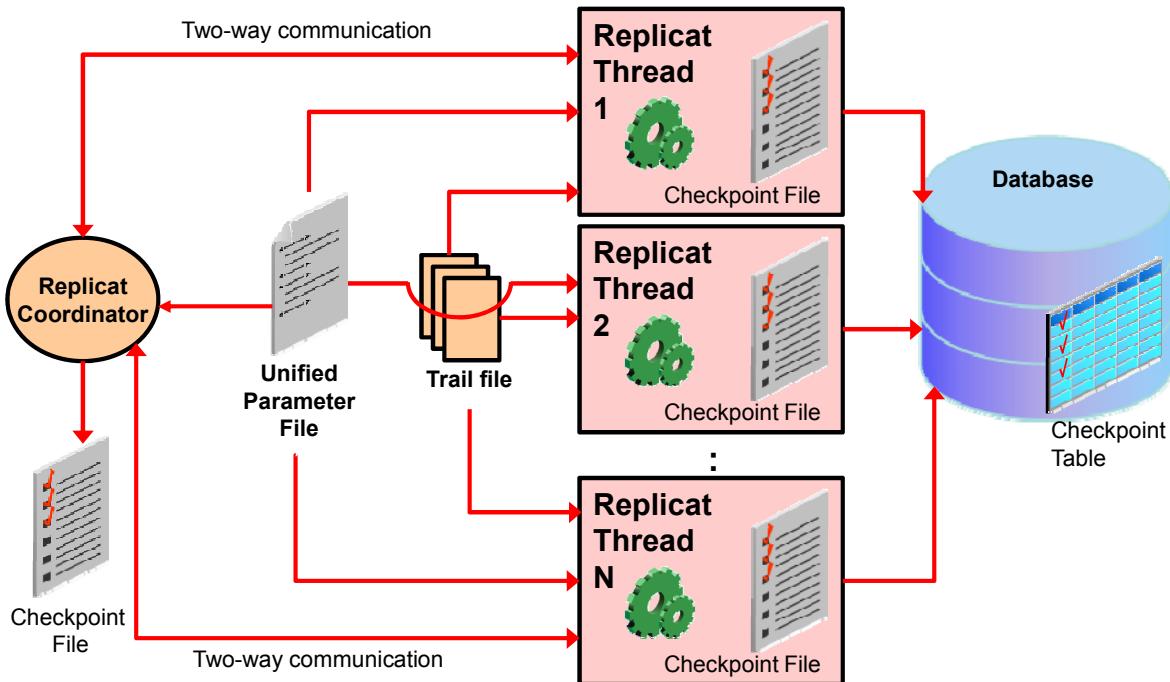


ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

- Logical Change Records (LCRs) are created by Integrated Extract, whereas canonical is created by Classic Extract.
- Integrated Replicat uses Lightweight Streaming API from the Replicat process to an Inbound Server, and converts canonical to LCR format if needed:
 - **Receiver:** Reads trails in LCR format
 - **Preparer (Reader):** Computes the dependencies between the transactions (primary key, unique indexes, foreign key), grouping transactions and sorting in dependency order
 - **Coordinator:** Coordinates transactions; maintains the order between applier processes
 - **Appliers:** Performs changes for assigned transactions, including conflict detection and error handling
- There are several new database `sysdba` tables and views named `v$GG_APPLY_*` and `DBA_APPLY_*`.
- Integrated Replicat is supported only on Oracle Database versions 11.2.0.4 and later.
- Integrated Replicat does not support GGSCI parameters such as `BULKLOAD`, `GENLOADFILES`, `RMTTASK`, and `SPECIALRUN`; they cause an abort. There is limited functionality for `SHOWSYNTAX` which does not abort but does not show interactive commands either.

Coordinated Replicat (New with 12c)



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Coordinated Replicat is also known as *coordinated apply*. This is independent of Integrated Replicat, and it works on most versions and most vendors' databases. The main feature is that all of the Replicat threads can apply in parallel, though there may be cases where a full barrier synchronization (serialization) is desirable.

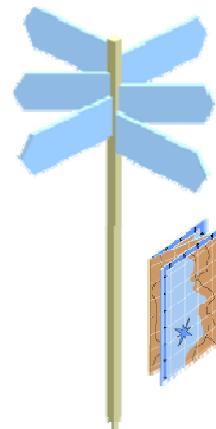
- **Coordinator:** It is responsible for thread creation and shutdown, dependency coordination, and statistics aggregation.
- **Multiple checkpoints:** If there are n threads, there are $n + 1$ checkpoint files, one for each thread and a coordinator's checkpoint file, plus the checkpoint *table* in the database.
- **Single parameter file:** This greatly simplifies ease of configuration, maintenance, and use.
- **Single trail file:** You do not need to figure out how to split up the work. The Replicat coordinator process does that.

By default, the coordinated Replicat sets up a maximum of 25 threads (there are actually 26, including one coordinator thread plus 25 Replicat threads), but you can increase or decrease this maximum.

Examples of Coordinated Apply are shown in the lesson titled “Data Selection and Filtering.” Do not use initial load `SpecialRun` or `ExtFile` with Coordinated Replicat options.

Roadmap

- Replicat
- Checkpoint
 - Adding Checkpoints
 - Using Checkpoints
- Initial Load
- Converting to Integrated
- Troubleshooting



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Change Delivery Tasks

On the target system:

1. Create a checkpoint table in the target database (best practice):

```
GGSCI> DBLogin UserIDAlias myalias  
GGSCI> Add CheckpointTable  
GGSCI> Info CheckpointTable
```

2. Create a parameter file for Replicat:

```
GGSCI> Edit Params
```

3. Create a Replicat group:

```
GGSCI> Add Replicat <params>
```

4. Start the Replicat process:

```
GGSCI> Start Replicat
```



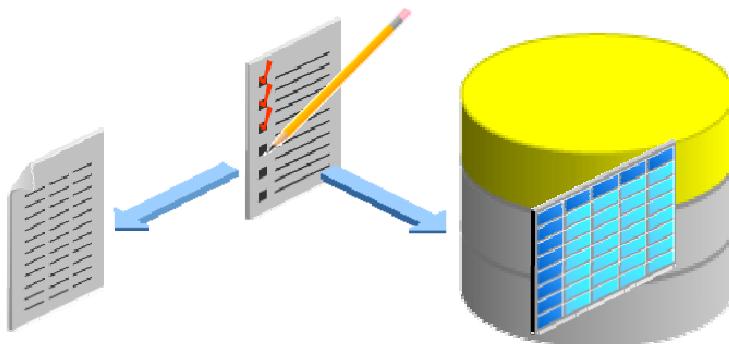
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Replicat reads the Oracle GoldenGate trail and applies changes to the target database. Like Extract, Replicat uses checkpoints to store the current read and write position and is added and started using the processing group name.

CheckpointTable

You can provide checkpoint instructions by:

- Specifying a default checkpoint table in the **GLOBALS** file
- Using **CheckpointTable** or **NoDBCcheckpoint** in the **Add Replicat** command to override the default
- Using file system or database for storing the checkpoint table



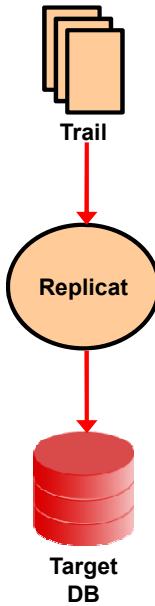
ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Replicat maintains checkpoints that provide a known position in the trail from which to start after an expected or unexpected shutdown. By default, a record of these checkpoints is maintained in a trail file on disk in the Oracle GoldenGate directory. Optionally, the checkpoint record can also be maintained in a checkpoint table in the target database.

Using database checkpointing is recommended because it enables the checkpoint to be included within Replicat's transaction, which improves recovery in certain situations. The checkpoint table remains small because rows are deleted when no longer needed, and it does not affect database performance. It can reside in a schema of your choice, but Oracle Corporation recommends using one that is dedicated to Oracle GoldenGate.

Sample Configuration



```

GGSCI> DBLogin UserIDAlias myalias
GGSCI> Add CheckpointTable mycheckpt
GGSCI> Edit Params repord
  
```

```

-- Some Comment here.

Replicat repord
UserIDAlias myalias
AssumeTargetDefs
DiscardFile /ggs/dirrpt/REPORD.dsc, Append
Map SALES.ORDERS, Target USSALES.USORDERS;
Map SALES.INVENTORY, Target USSALES.USINVENTORY;
  
```

```

GGSCI> Add Replicat repord, ExtTrail
/ggs/dirdat/rt
GGSCI> Start Replicat repord
  
```

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In this example, `DBLogin UserID` and `Password` log the user in to the database to add the checkpoint table. Note that `DBLogin` and `UserID` in the Replicat might be different.

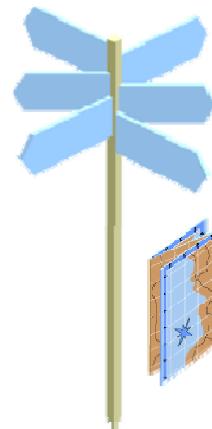
For the Replicat parameters, `UserID` and `Password` provide the credentials to access the database and `AssumeTargetDefs` is used when the source and target systems have the same data definition with identical columns. If they do not, you need a `SourceDefs` parameter referencing a source definitions file generated by DEFGEN.

`DiscardFile` creates a log file to receive records that cannot be processed. `MAP` establishes the relationship between the source table and the target table. `Add Replicat` names the Replicat group `REPORD` and establishes a local trail (`ExtTrail`) with the two-character identifier `rt` residing in the `dirdat` directory.

As always, the `Map` statements end with a semicolon; the other statements do not.

Roadmap

- Replicat
- Checkpoint
- Initial Load
 - Collisions
- Converting to Integrated
- Troubleshooting



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Avoiding Collisions with Initial Load

- If the source database remains active during an initial load, you must either avoid or handle any collisions when updating the target with interim changes.
- If you can back up, restore, or clone the database at a point in time, you can avoid collisions by starting Replicat to read trail records from a specific transaction commit sequence number (CSN):

```
GGSCI> Start Replicat <group> AtCSN | AfterCSN <csn>
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

During the initial load, updates may occur to the source during the load process. To try to avoid these collisions, perform the following steps:

1. Use a standby copy of the source database for the initial load.
2. After the initial load completes, note the highest CSN number of the standby database. The CSN varies by database. (For example, for Oracle Database, it is the system change number [SCN].)
3. Start Replicat to read from the next CSN:

```
Start Replicat <group> AtCSN <csn> | AfterCSN <csn> |  
SkipTransaction
```

- **AtCSN <csn>** causes Replicat to skip transactions in the trail until it finds a transaction that contains the specified CSN. **<csn>** must be in the format that is native to the database.
- **AfterCSN <csn>** causes Replicat to skip transactions in the trail until it finds the first transaction after the one that contains the specified CSN.
- **SkipTransaction** causes Replicat to skip the first transaction in the trail after startup. All operations in that first transaction are excluded.

Handling Collisions with Initial Load

- If you cannot avoid collisions, you must *handle* them.
- The Replicat **HandleCollisions** parameter can be used.
 - When Replicat encounters a duplicate-record error on an insert, it writes the change record over the initial data load record.
 - When Replicat encounters a missing-record error for an update or delete, the change record is discarded.
- After all of the change data generated during the load has been replicated, turn off **HandleCollisions**:

```
GGSCI> Send Replicat <group> NoHandleCollisions  
GGSCI> Edit Param <group>
```

Temporary

Permanently remove parameter

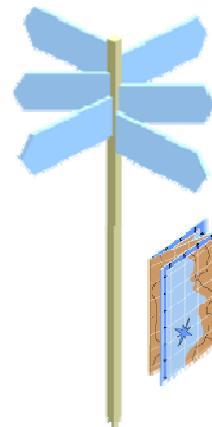
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

`HandleCollisions` processing requires that each target table have a primary key or unique index. If you cannot create a temporary primary key or unique index through your application, use the `KeyCols` argument of the `Table` or `Map` parameter to designate columns as a substitute key. Otherwise, the source database must be quiesced for the initial load.

Roadmap

- Replicat
- Checkpoint
- Initial Load
- Converting to Integrated
- Troubleshooting



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

This is optional and is covered in more detail in the *Oracle GoldenGate 11g Advanced Configuration for Oracle* course.

Converting to or from Integrated Apply (New with 12c)

- Convert to Integrated:

```
GGSCI> DBLogin UserIdAlias myalias
GGSCI> Stop Replicat my_repl
GGSCI> Alter Replicat my_repl, Integrated
GGSCI> Start Replicat my_repl
```

- Convert from Integrated:

```
GGSCI> DBLogin UserIdAlias myalias
GGSCI> Info CheckpointTable
GGSCI> Stop Replicat my_repl
GGSCI> Alter Replicat my_repl, NonIntegrated,
      CheckPointTable <tablename>
GGSCI> Start Replicat my_repl
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In Integrated mode, the checkpoint *file* is updated to indicate Integrated; the database checkpoint *table* is not used.

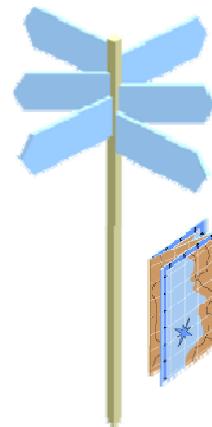
In NonIntegrated (classic) mode, you must have added a database checkpoint table, which can be specified in the GLOBALS parameter file.

Unlike the Extract, you do not normally need to Register or Unregister a Replicat. However, you might need to do this if you have to forcibly remove the Replicat (that is, if you forgot to do a DBLogin before doing a Delete.)

Do not use Initial Load SpecialRun or ExtFile with the Integrated Replicat options.
All this integrated information applies only to Oracle Database 11.2.0.4 and later.

Roadmap

- Replicat
- Checkpoint
- Initial Load
- Converting to Integrated
- Troubleshooting
 - GGSCI Process Information
 - Report Files
 - Log Files



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

After everything is started, the `Info All` process status (see the next slide) ideally changes from STOPPED to RUNNING. However, you may see the status change from STOPPED to ABEND. If the process fails to start, this section discusses some troubleshooting techniques.

For additional details, see the follow-on course *Oracle GoldenGate 11g Troubleshooting and Tuning*.

Obtaining Process Information Through GGSCI

GGSCI>

- **Info {Extract | Replicat} <group> [Detail]**
- **Info Manager**
- **Info All**
- **Stats {Extract | Replicat} <group>**
- **Status {Extract | Replicat} <group>**
- **Status Manager**
- **Lag {Extract | Replicat} <group>**



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The primary way to view processing information is through GGSCI (as shown in previous lessons). The slide shows the key commands to view process information.

- **Info {Extract | Replicat} <group> [Detail]**: Run status, checkpoints, approximate lag, and environmental information
- **Info Manager | Mgr**: Run status and port number
- **Info All**: Info output for all Oracle GoldenGate processes on the system
- **Stats {Extract | Replicat} <group>**: Statistics for operations processed
- **Status {Extract | Replicat} <group>**: Run status (starting, running, stopped, and abended)
- **Status Manager | Mgr**: Run status
- **Lag {Extract | Replicat} <group>**: Latency between last record processed and time stamp in the data source

Obtaining Process Information Through GGSCI

GGSCI>

- **Info {ExtTrail | RmtTrail} <path_name>**
- **Send Manager**
- **Send {Extract | Replicat}**
- **View Report <group>**
- **View GGSEvt**
- **<command> ER <wildcard>**



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The slide shows the additional commands to view process information:

- **Info {ExtTrail | RmtTrail} <path_name>**: Name of the associated process, position of the last data processed, and maximum file size
 - **Send Manager**: Run status, information about child processes, port information, and trail purge settings
 - **Send {Extract | Replicat}**: Depending on the process, information about memory pool, lag, TCP stats, long-running transactions, process status, recovery progress, and more
 - **View Report <group>**: Contents of the process report
 - **View GGSEvt**: Contents of the Oracle GoldenGate error log
 - **<command> ER <wildcard>**: Information dependent on the <command> type:
 - Info
 - Lag
 - Send
 - Stats
 - Status
- <wildcard> is either * or ?.

Process Report Files

Process reports (depending on the process) enable you to view the following:

- Parameters in use
- Table and column mapping
- Database information
- Runtime messages and errors
- Runtime statistics for the number of operations processed

These reports can be viewed with:

- GGSCI > **View Report**
- Any text editor
- Oracle Management Pack for Oracle GoldenGate



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Every Extract, Replicat, and Manager process generates a report file at the end of each run. The report can help you diagnose problems that occurred during the run, such as invalid mapping syntax, SQL errors, and connection errors.

To view a process report, use any of the following:

- A standard shell command for viewing a text file, such as `more`, `less`, or `cat`
- Oracle Management Pack for Oracle GoldenGate (Oracle GoldenGate Director and/or Oracle GoldenGate Monitor)
- The `view Report` command in GGSCI:

```
View Report {<group> | <file_name> | Mgr}
```

Sample Extract Process Report

```
*****
**          Running with the following parameters          **
*****
2013-04-10 12:58:48  INFO      OGG-03035  Operating system character set
identified as UTF-8. Locale: en_US, LC_ALL:.

-- WEST
Extract pwconf
RmtHost easthost, MgrPort 15001, Compress
RmtTrail ./dirdat/pf
Passthru
Table west.*;
-- cannot use GetBeforeCols on the data pump above with Passthru...

2013-04-10 12:58:53  INFO      OGG-01226  Socket buffer size set to 27985
(flush size 27985).

*****
**          Run Time Messages          **
*****
Opened trail file ./dirdat/wf000000 at 2013-04-10 12:58:53
Wildcard TABLE resolved (entry west.*):
  Table "WEST"."PRODUCTS";
PASSTHRU mapping resolved for source table WEST.PRODUCTS
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Because this is for the `pwconf` process group, the report is in `dirrpt/pwconf.rpt`. The normal report is much longer than is shown in the slide. The parameter file `pwconf.prm` for this process is included in the report.

To view the report, you can use the following within GGSCI:

GGSCI> **View Report groupname**

Alternatively, you can view the file by using any editor.

Discard Files

```
Oracle GoldenGate Delivery for Oracle process started, group RWCONF
discard file opened: 2013-04-10 12:43:13
ORA-20017: asta0009 6144935
ORA-06512: at "LON.STARTASTA0009_INSERT", line 31
ORA-04088: error during execution of trigger 'LON.STARTASTA0009_INSERT'
Operation failed at seqno 45 rba 12483311
Problem replicating PRODTAB.ASTA0009 to ASTA0009
Error occurred with insert record (target format)...
*
A_TIMESTAMP = 2013-05-15 13:18:32
RELA_PERSON_NR = 3618047
RELA_BEZART = 1
RELA_BEZCODE = 01
RELA_AZ_BAFL = 2819220
RELA_STEMPEL = 0
AKTION = I
OK = 1.0000
NOTOK = -1.0000
*
```

The Oracle logo, consisting of the word "ORACLE" in white capital letters on a red background.

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Discard files are used to capture information about the Oracle GoldenGate operations that have failed.

The Discard file reports information such as:

- The database error message
- The sequence number of the data source or trail file
- The relative byte address of the record in the data source or trail file
- The details of the discarded operation, such as the column values of a DML statement or the text of a DDL statement

A Discard file can be used for both Extract and Replicat.

To use a Discard file, include the `DiscardFile` parameter in the Extract or Replicat parameter file. Parameters are covered in the lesson titled “Oracle GoldenGate Parameters.” Starting with OGG 12.1.2, `DiscardFile` is the default.

Using the ggserr.log Error Log

```

2013-04-06 11:10:02  INFO    OGG-00987 Oracle GoldenGate Command Interpreter for
Oracle: GGSCI command (oracle): edit param mgr.
2013-04-06 11:11:00  INFO    OGG-00987 Oracle GoldenGate Command Interpreter for
Oracle: GGSCI command (oracle): start mgr.
2013-04-06 11:11:01  INFO    OGG-00983 Oracle GoldenGate Manager for Oracle,
mgr.prm: Manager started (port 15000).
2013-04-06 11:57:24  INFO    OGG-00987 Oracle GoldenGate Command Interpreter for
Oracle: GGSCI command (oracle): add trandata west.account.
2013-04-06 11:59:25  INFO    OGG-00987 Oracle GoldenGate Command Interpreter for
Oracle: GGSCI command (oracle): add schematrandata hr.
2013-04-06 11:59:25  INFO    OGG-01788 Oracle GoldenGate Command Interpreter for
Oracle: SCHEMATRANLDA has been added on schema hr.
2013-04-06 11:59:45  ERROR   OGG-01780 Oracle GoldenGate Command Interpreter for
Oracle: Missing/Invalid argument(s) on ADD/INFO/DELETE SCHEMATRANLDA command.
2013-04-06 12:00:03  ERROR   OGG-01780 Oracle GoldenGate Command Interpreter for
Oracle: Missing/Invalid argument(s) on ADD/INFO/DELETE SCHEMATRANLDA command.
2013-04-06 12:00:09  INFO    OGG-01786 Oracle GoldenGate Command Interpreter for
Oracle: Schema level supplemental logging is disabled on schema GGUSER.
2013-04-06 12:00:09  INFO    OGG-01786 Oracle GoldenGate Command Interpreter for
Oracle: Schema level supplemental logging is disabled on schema WEST.
2013-04-06 12:00:09  INFO    OGG-01786 Oracle GoldenGate Command Interpreter for
Oracle: Schema level supplemental logging is disabled on schema BI.
:
:

```

ERROR vs INFO



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can use the Oracle GoldenGate error log to view the following:

- A history of GGSCI commands
- The Oracle GoldenGate processes that started and stopped
- The processing that was performed
- The errors that occurred
- Informational and warning messages

Because the error log shows events as they occurred in sequence, it is a good tool for detecting the cause (or causes) of an error. For example, you might discover that someone stopped a process or a process failed to make a TCP/IP or database connection.

To view the error log, use any of the following:

- A standard shell command to view the ggserr.log file within the root Oracle GoldenGate directory
- Oracle Management Pack for Oracle GoldenGate (Oracle GoldenGate Director and/or Oracle GoldenGate Monitor)
- The view GGSEVT command in GGSCI
- An external table

Using the System Logs

- Oracle GoldenGate writes errors that are generated at the level of the operating system:
 - Event Viewer on Windows
 - `syslog` on UNIX and Linux
- Use the `SYSLOG` parameter to control the types of messages that Oracle GoldenGate sends to the system logs on a Windows or UNIX system.
- By using the `SYSLOG` parameter, messages can be filtered to:
 - Include all Oracle GoldenGate messages
 - Suppress all Oracle GoldenGate messages
 - Include information, warning, or error messages, or any combination of these types



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate writes errors that are generated at the level of the operating system to the Event Viewer in Windows or to the `syslog` in UNIX and Linux. Oracle GoldenGate events are basically in the same format in the UNIX, Linux, and Windows system logs. The Oracle GoldenGate errors that appear in the system logs also appear in the Oracle GoldenGate error log.

You can use `SYSLOG` as a `GLOBALS` parameter, a Manager parameter, or both. When present in the `GLOBALS` parameter file, `SYSLOG` controls message filtering for all Oracle GoldenGate processes on the system. When present in the Manager parameter file, `SYSLOG` controls message filtering only for the Manager process. If `SYSLOG` is used in both the `GLOBALS` and Manager parameter files, the Manager setting overrides the `GLOBALS` setting for the Manager process. This enables you to use separate settings for Manager and all the other Oracle GoldenGate processes.

Discussion Questions

1. What does Replicat do?
2. When is **AssumeTargetDefs** valid?
3. How does Replicat know the layout of the source tables when source and target schemas differ?
4. What commands are used to create and start a Replicat group?
5. Which GGSCI command creates an Oracle GoldenGate checkpoint table on the target database?
6. What is the purpose of **DiscardFile**?
7. Which parameter manages conflicts between initial load and change replication?
Where is it specified?



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answers

1. Replicat reads change data from Oracle GoldenGate trails and applies it to a target database via SQL commands.
2. When the source and target table structures (column order, data type and length) are identical
3. Replicat uses the source definitions created by defgen.
4. Add CheckpointTable (optional)
Edit Params
Add Replicat
Start Replicat
5. Add CheckpointTable (must be logged in to the database)
6. DiscardFile identifies operations that could not be processed by Replicat.
7. HandleCollisions, which is specified in the Replicat parameter file for change delivery. Turn it off after initial load data is processed.

Quiz

Which of the following are tools that you would use to determine whether all Oracle GoldenGate processes are up and running?

- a. `tcperrs`
- b. `CMDSEC`
- c. `GGSCI`
- d. Process report
- e. Oracle GoldenGate Monitor



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: c, d, e

Quiz

Which files are used to capture information about the Oracle GoldenGate operations that have failed?

- a. Discard files
- b. Purge trail files
- c. `tcperrs` files



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: a

Summary

In this lesson, you should have learned how to:

- Explain what the Replicat process does
- Configure and start a Replicat process
- Configure Replicat to handle collisions between changes and the Initial Load data
- Troubleshoot a started solution



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice 7 Overview: Configuring Change Delivery

This practice covers the following topics:

- Setting up the checkpoint table on the target system
- Setting up Replicat delivery
- Generating DML data
- Starting Oracle GoldenGate processes
- Stopping processes and checking statistics



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

8

Extract Trails and Files

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

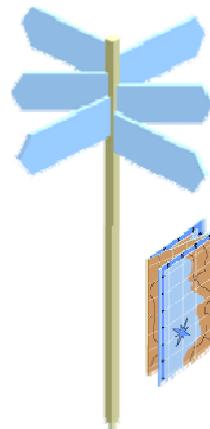
- Describe and contrast Extract trails and files
- Describe the formats that Extract trails and files can have
- View Extract trails and files with `logdump`
- Reverse the sequence of operations in an Extract trail or file (to back out changes)



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Roadmap

- Trail Format
 - Local and Remote
 - Cleanup
 - Record Header Area
 - Record Data Area
- Alternative Trail Formats
- logdump
- reverse



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Overview of Extract Trails and Files

- Extract writes data to any of the following:
 - Local trail (`ExtTrail`) on the local system
 - Local file (`ExtFile`) on the local system
 - Remote trail (`RmtTrail`) on a remote system
 - Remote file (`RmtFile`) on a remote system
- Extract trails and files are unstructured, with variable length records.
 - I/O is performed using large block writes.
- Extract writes checkpoints for trails during change capture:
 - This guarantees that no data is lost during restart.
 - Multiple Replicat processes may process the same trail.
- Extract does not write checkpoints for files.

**ORACLE**

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate trails support the continuous extraction and replication of database changes and stores them temporarily on disk. A trail can exist on the source or target system, or on an intermediary system, depending on how you configure Oracle GoldenGate. A trail can reside on any platform that Oracle GoldenGate supports.

Extract Trails and Files Distribution

- Extract can write:
 - To local trails, and then distribute over IP with a data pump to remote trails
 - To multiple trails:
 - For distribution to multiple systems/disk storage devices
 - For parallel processing by downstream processes
- Trails and files can be transported online using TCP/IP or sent in batch using any file transfer method.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

When transporting trails via TCP/IP, a Server Collector process on the target platform collects, writes, and checkpoints blocks of records in one or more extract files.

Extract Trails and Files Contents

- Each record in the trail contains an operation that has been committed in the source database.
- Committed transactional order is preserved.
- Operations in a transaction are grouped together in the order in which they were applied.
- By default, only the primary key and changed columns are recorded.
- Flags indicate the first and last records in each transaction.



ORACLE®

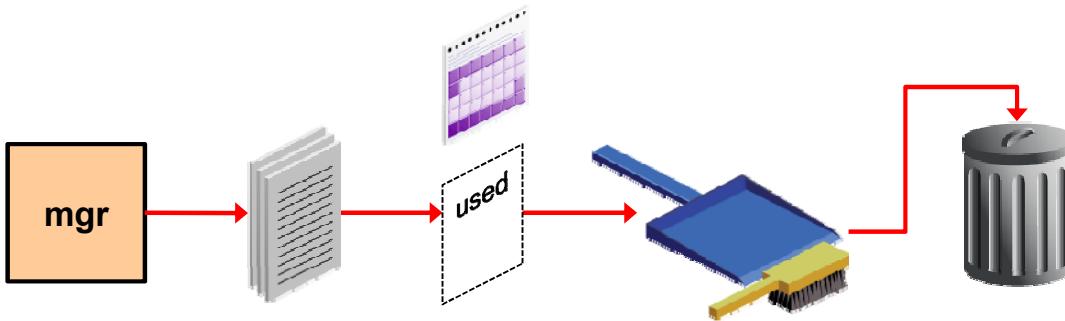
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can examine the contents of trail files directly by using the `logdump` utility if you need to troubleshoot. `logdump` is covered later in this lesson.

Extract Trails and Files Cleanup

Trail files can be purged after they are consumed:

- The temporary storage requirement is small if processes keep pace.
- Configure the Manager to purge used trail data (best practice).



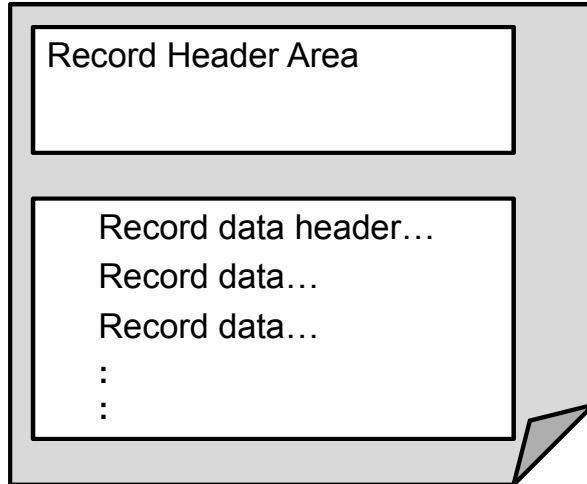
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can configure more than one Replicat to process a trail. After all of the data has been consumed, Replicat can then purge the data using the `MinKeepDays` parameter. As long as Replicat remains current, your temporary storage requirements for trails can be very low. If multiple Replicat processes are configured against a single trail, you can instruct the Manager to purge the data in the trail as soon as all checkpoints have been resolved. As long as replication processes keep pace, temporary storage requirements can be kept quite low.

Trail Format

- By default, trails are formatted in canonical format, allowing them to be exchanged rapidly and accurately among heterogeneous databases.
- Each trail file contains the following:
 - Record header area
 - Record data area



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate Trails

Trail files are unstructured files containing variable length records. They are unstructured and written in large blocks for best performance. Trail files contain:

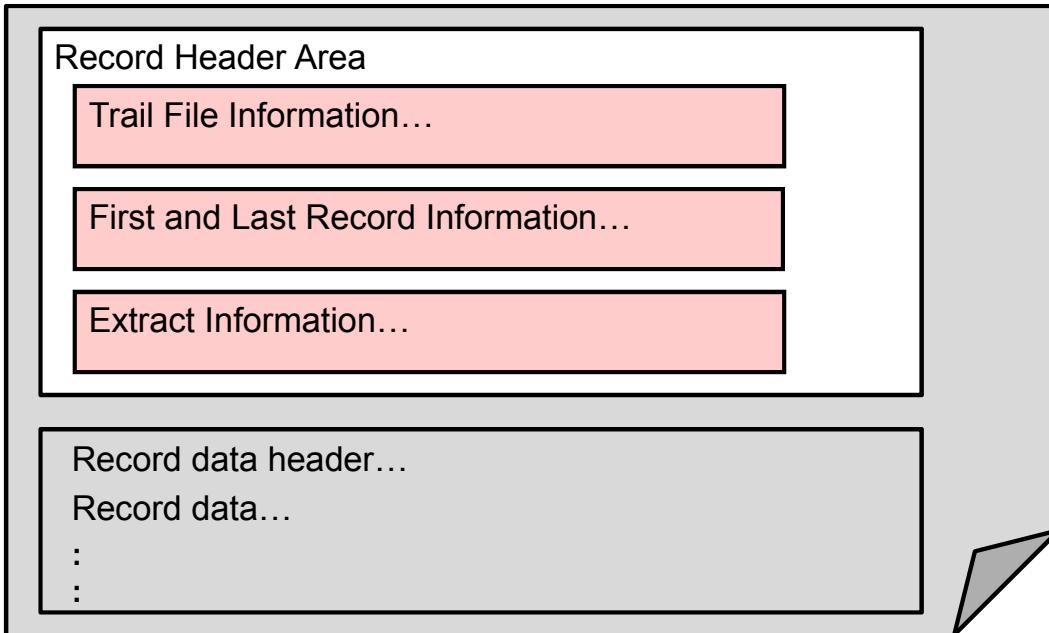
- **Record header area:** Stored at the beginning of the file and contains information about the trail file itself
- **Record data area:** Contains a header area as well as a data area

Checkpoints

Both Extract and Replicat maintain checkpoints into the trails. Checkpoints provide persistent processing whenever a failure occurs. Each process resumes where the last checkpoint was saved, guaranteeing that no data is lost. One Extract can write to one or many trails. Each trail can then be processed by one or many Replicat processes.

Record Header Area

Trail file:



ORACLE®

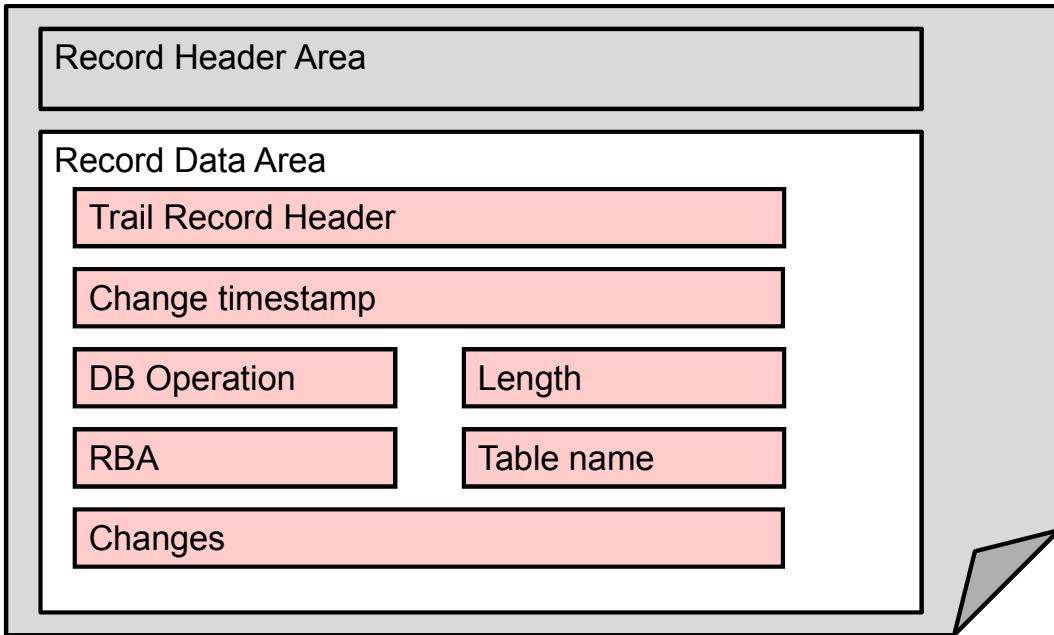
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Each trail file has a record header area that contains:

- Trail File Information
 - Compatibility level
 - Character set (globalization function with version 11.2.1 and later)
 - Creation time
 - File sequence number
 - File size
- First and Last Record Information
 - Timestamp
 - Commit Sequence Number (CSN)
- Extract Information
 - Oracle GoldenGate version
 - Group name
 - Hostname and Hardware type
 - OS type and version
 - DB type, version, and character set

Record Data Area

Trail file:



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The trail file header and the data area of the Oracle GoldenGate trail record contain:

- Trail record header
- The time that the change was written to the Oracle GoldenGate file
- The type of database operation (Insert, Update, Delete)
- The length of the record
- The relative byte address within the trail file
- The table name
- The data changes in hex format
- Optional user token area

The contents of the record header and data areas are discussed (with an example) in the section “Viewing Trail Records” later in this lesson.

Setting the Compatibility Level

- The `<major>. <minor>` setting identifies the trail file format version numbers used by Oracle GoldenGate.
- This allows customers to use different versions of Oracle GoldenGate Extract, trail files, and Replicat together.
- Set in the Extract ExtFile, ExtTrail, RmtFile, or RmtTrail parameter:

```
:  
RmtTrail /ggs/dirdat/ex, Format Release 10.4  
:
```

- The input and output trails of a data pump must have the same compatibility level.

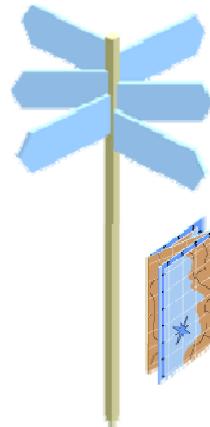


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

It is recommended that all instances of Oracle GoldenGate be the same version to take advantage of the new functionality. However, this is not required.

Roadmap

- Trail Format
- Alternative Trail Formats
 - Logical Change Records (LCRs)
 - ASCII
 - SQL
 - XML
- logdump
- reverse



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Alternative Trail Formats

- Instead of the default canonical format, alternative formats can be used to output data.
- This is beneficial if database load utilities or other programs are used that require different input.
- These alternative formats include:
 - Logical Change Records (LCRs)
 - FormatASCII
 - FormatSQL
 - FormatXML



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

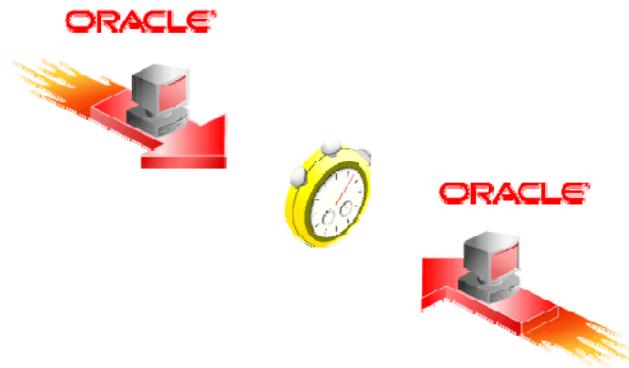
The benefits of each alternative format are discussed in the next few slides.

Logical Change Records (LCRs)

Instead of applying SQL-type statements (slower), LCRs enable the APIs to interact with Oracle Database directly (faster).

However, LCRs are:

- Oracle-to-Oracle only
- Integrated Capture only
- 11g and later only



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The integrated mode (and, therefore, the LCR trail) is the platform to which new features will be added. New features may or may not be added to the Classic Capture mode, which produces the canonical trails for Oracle and non-Oracle databases. The specific 11g version depends on the topology, not all 11g versions are supported in all topologies.

LCRs are not an option for non-Oracle databases, such as IBM DB2, Microsoft SQL Server, and so on, and also do not apply to non-flagship Oracle databases, such as MySQL or TimesTen.

Use of Integrated Replicat will convert canonical format to LCR format at the target using the Lightweight Streaming API.

FormatASCII

- Output is in external ASCII format.
- FormatASCII is required by the file-to-database-utility initial-load method.
- The FormatASCII statement must be before the extract files or trails statements that are listed in the parameter file.
- FormatASCII can format data for popular database load utilities:
 - FormatASCII, BCP
 - FormatASCII, SQLLOADER
- Data cannot be processed by Oracle GoldenGate Replicat because Replicat expects the default canonical format.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Use the FormatASCII parameter to output data in external ASCII format instead of the default Oracle GoldenGate canonical format. Using FormatASCII, you can format output that is compatible with most database load utilities and other programs that require ASCII input.

Using the FormatASCII, BCP option provides output that is compatible with SQL Server's BCP, DTS, or SQL Server Integration Services (SSIS) bulk-load utility. The FormatASCII, SQLLOADER option produces a fixed-length, ASCII-formatted file that is compatible with the Oracle SQL*Loader utility or the IBM Load Utility program.

FormatASCII Sample Output

Sample transaction:

```
INSERT INTO customer VALUES ("Eric", "San Fran", 550);
UPDATE customer SET balance=100 WHERE custname="Eric";
COMMIT;
```

Example 1. FormatASCII without options produces:

```
B,1997-02-17:14:09:46.421335,8,1873474,
I,A,TEST.CUSTOMER,CUSTNAME,'Eric',LOCATION,'San Fran',
BALANCE,550,
V,A,TEST.CUSTOMER,CUSTNAME,'Eric',BALANCE,100,
C,
```

Example 2. FormatASCII, NONAMES, DELIMITER ' | '
produces:

B	1997-02-17:14:09:46.421335		8		1873474						
I	A	CUSTOMER		'Eric'		'San Fran'		550			
V	A	CUSTOMER		CUSTNAME		'Eric'		BALANCE		100	
C											

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In Example 1:

- The transaction begins (B)
- A record is added (I)
- A record is updated (V)
- The transaction commits (C)

Note that Example 2 returns column names for the CUSTNAME and BALANCE columns because the record is a compressed update and PLACEHOLDERS was not used.

FormatSQL

- Output is in external SQL DML format.
- FormatSQL generates SQL statements (`INSERT`, `UPDATE`, and `DELETE`) that can be applied to both SQL and Enscribe tables.
- Data cannot be processed by Oracle GoldenGate Replicat because Replicat expects the default canonical format.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Every record in a transaction is contained between the begin and commit indicators. Each combination of commit timestamp and relative byte address (RBA) is unique. The output can be customized with optional arguments.

FormatSQL Sample Output

```
B,2008-11-11:13:48:49.000000,1226440129,155,
DELETE FROM TEST.TCUSTMER WHERE CUST_CODE='JANE';
DELETE FROM TEST.TCUSTMER WHERE CUST_CODE='WILL';
DELETE FROM TEST.TCUSTORD WHERE CUST_CODE='JANE' AND
ORDER_DATE='1995-11-11:13:52:00' AND PRODUCT_CODE='PLANE' AND
ORDER_ID='256';
DELETE FROM TEST.TCUSTORD WHERE CUST_CODE='WILL' AND
ORDER_DATE='1994-09-30:15:33:00' AND PRODUCT_CODE='CAR' AND
ORDER_ID='144';
INSERT INTO TEST.TCUSTMER (CUST_CODE,NAME,CITY,STATE) VALUES
('WILL','BG SOFTWARE CO.', 'SEATTLE', 'WA');
INSERT INTO TEST.TCUSTMER (CUST_CODE,NAME,CITY,STATE) VALUES
('JANE','ROCKY FLYER INC.', 'DENVER', 'CO');
INSERT INTO TEST.TCUSTORD
(CUST_CODE, ORDER_DATE, PRODUCT_CODE, ORDER_ID, PRODUCT_PRICE, PRO-
DUCT_AMOUNT, TRANSACTION_ID) VALUES ('WILL', '1994-09-
30:15:33:00', 'CAR', '144', 17520.00, 3, '100');
INSERT INTO TEST.TCUSTORD
(CUST_CODE, ORDER_DATE, PRODUCT_CODE, ORDER_ID, PRODUCT_PRICE, PRO-
DUCT_AMOUNT, TRANSACTION_ID) VALUES ('JANE', '1995-11-
11:13:52:00', 'PLANE', '256', 133300.00, 1, '100');
C,
```

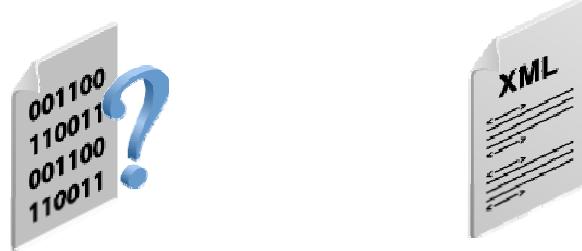


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

This SQL format is not compact, but it is very portable. The FormatSql parameter can be used to monitor what is being captured. It can also be used to create a SQL script that can be used in SQL*Plus. For example, a SQL script can be created to apply missed transactions in a recovery.

FormatXML

- Output is in XML format.
- The `NoBinaryChars` parameter is used with `FormatXML`. (Contact Oracle Support for additional guidance for using this parameter.)
- Data cannot be processed by Oracle GoldenGate Replicat because Replicat expects the default canonical format.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

`NoBinaryChars` can cause Oracle GoldenGate to interpret a binary character to be the end of the data in that column, even if there is other data remaining in the column.

Format XML Sample Output

```
<transaction timestamp="2008-11-11:14:33:12.000000">
  <dbupdate table="TEST.TCUSTMER" type="insert">
    <columns>
      <column name="CUST_CODE" key="true">ZEKE</column>
      <column name="NAME">ZEKE'S MOTION INC.</column>
      <column name="CITY">ABERDEEN</column>
      <column name="STATE">WA</column>
    </columns>
  </dbupdate>
  <dbupdate table="TEST.TCUSTMER" type="insert">
    <columns>
      <column name="CUST_CODE" key="true">ZOE</column>
      <column name="NAME">ZOE'S USED BICYCLES</column>
      <column name="CITY">ABERDEEN</column>
      <column name="STATE">WA</column>
    </columns>
  </dbupdate>
  <dbupdate table="TEST.TCUSTMER" type="insert">
    <columns>
      <column name="CUST_CODE" key="true">VAN</column>
      <column name="NAME">VAN'S BICYCLES</column>
      <column name="CITY">ABERDEEN</column>
      <column name="STATE">WA</column>
    </columns>
  </dbupdate>
</transaction>
```

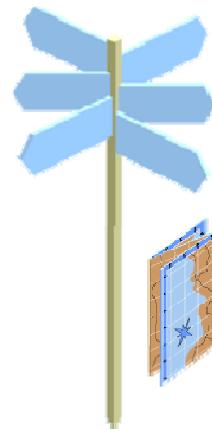


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

This XML format is not very compact, but it is very portable.

Roadmap

- Trail Format
- Alternative Trail Formats
- logdump
 - Opening
 - Viewing
 - Filtering
- reverse



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

logdump Utility

- The logdump utility enables you to:
 - Display or search for information that is stored in Oracle GoldenGate trails or extract files
 - Save a portion of an Oracle GoldenGate trail to a separate trail file
- To start logdump from the Oracle GoldenGate installation directory:

```
[OS prompt] ./logdump
```

- To access help:

```
Logdump 1> help
```



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

logdump provides access to Oracle GoldenGate trails, which are unstructured files with a variable record length. Each record in the trail contains a header, known as the GGS Header (unless the NoHeaders Extract parameter was used), an optional user token area, and the data area.

For more information about the logdump utility, see the *Oracle GoldenGate Troubleshooting and Tuning Guide*.

Opening a Trail

```
Logdump> open dirdat/rt000000  
Current LogTrail is /ggs/dirdat/rt000000
```

Response/confirmation



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The syntax to open a trail is:

Open *<file_name>*

In this syntax, *<file_name>* is either the relative name or a fully qualified name of the file, including the file sequence number. logdump reads one trail file at a time.

Setting Up a View

- To view the trail file header:

```
Logdump 1> fileheader on
```

- To view the record header with the data:

```
Logdump 2> ghdr on
```

- To add column information:

```
Logdump 3> detail on
```

- To add hex and ASCII data values to the column list:

```
Logdump 4> detail data
```

- To control how much record data is displayed:

```
Logdump 5> reclen 280
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

fileheader [on | off | detail] controls whether or not the trail file header is displayed.

ghdr [on | off] controls whether or not the record header is displayed with each record. Each record contains a header that includes information about the transaction environment. Without arguments, **ghdr** displays the status of header display (ON or OFF).

detail {on | off | data}

DETAIL ON displays a list of columns that includes the column ID, length, and value in hex and ASCII.

DATA adds hex and ASCII data values to the column list.

DETAIL OFF turns off detailed display.

Usertoken: By default, the name of the token and its length are displayed. Use the **USERTOKEN DETAIL** option to show the actual token data. User tokens are discussed in the lesson titled “Additional Transformation Topics.”

reclen controls how much of the record data is displayed. You can use **reclen** to control the amount of scrolling that must be done when records are large, while still showing enough data to evaluate the record. Data beyond the specified length is truncated.

Viewing the Trail File Header

```

Logdump 14662 > fileheader detail
Logdump 14663 > pos 0
Reading forward from RBA 0
Logdump 14664 > n

TokenID x46 'F' Record Header      Info x00 Length 587
TokenID x30 '0' TrailInfo          Info x00 Length 303
TokenID x31 '1' MachineInfo       Info x00 Length 103
TokenID x32 '2' DatabaseInfo      Info x00 Length 88
TokenID x33 '3' ProducerInfo      Info x00 Length 85
TokenID x34 '4' ContinuityInfo    Info x00 Length 4
TokenID x5a 'Z' Record Trailer    Info x00 Length 587
2008/07/18 13:40:26.034.631 FileHeader           Len 587 RBA 0
Name: *FileHeader*
3000 012f 3000 0008 660d 0a71 3100 0006 0001 3200 | 0.../0....f..q1.....2.
0008 0000 0016 3300 000c 02f1 7834 eac7 7f3f 3400 | .....3.....x4....?4.
0037 0031 7572 693a 7465 6c6c 7572 6961 6e3a 3a68 | .7.luri:tellurian::h
6f6d 653a 6d63 6361 7267 6172 3a67 6773 3a67 6773 | ome:mccargar:ggs:ggs
4f72 6163 6c65 3a73 6f75 7263 6536 0000 1700 112e | Oracle:source6.....
2f64 6972 6461 742f 6572 3030 3030 3030 3700 0005 | /dirdat/er0000007...
0138 0000 0800 01e2 4039 0000 0c00 0000 0000 001d | .8.....@9.....
GroupID x30 '0' TrailInfo          Info x00 Length 303
3000 012f 3000 0008 660d 0a71 3100 0006 0001 3200 | 0.../0....f..q1.....2.
etc.

```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

If the file header [on|detail] option is used, the file header data is displayed.

Viewing Trail Records

- To go to the first record and to move from one record to another in sequence:

```
Logdump 6 > pos 0
```

```
Logdump 7 > next
```

Or just type n.

- To position at an approximate starting point and locate the next good header record:

```
Logdump 8 > pos <approximate RBA>
```

```
Logdump 9 > scanforheader
```

Or just type sfh.

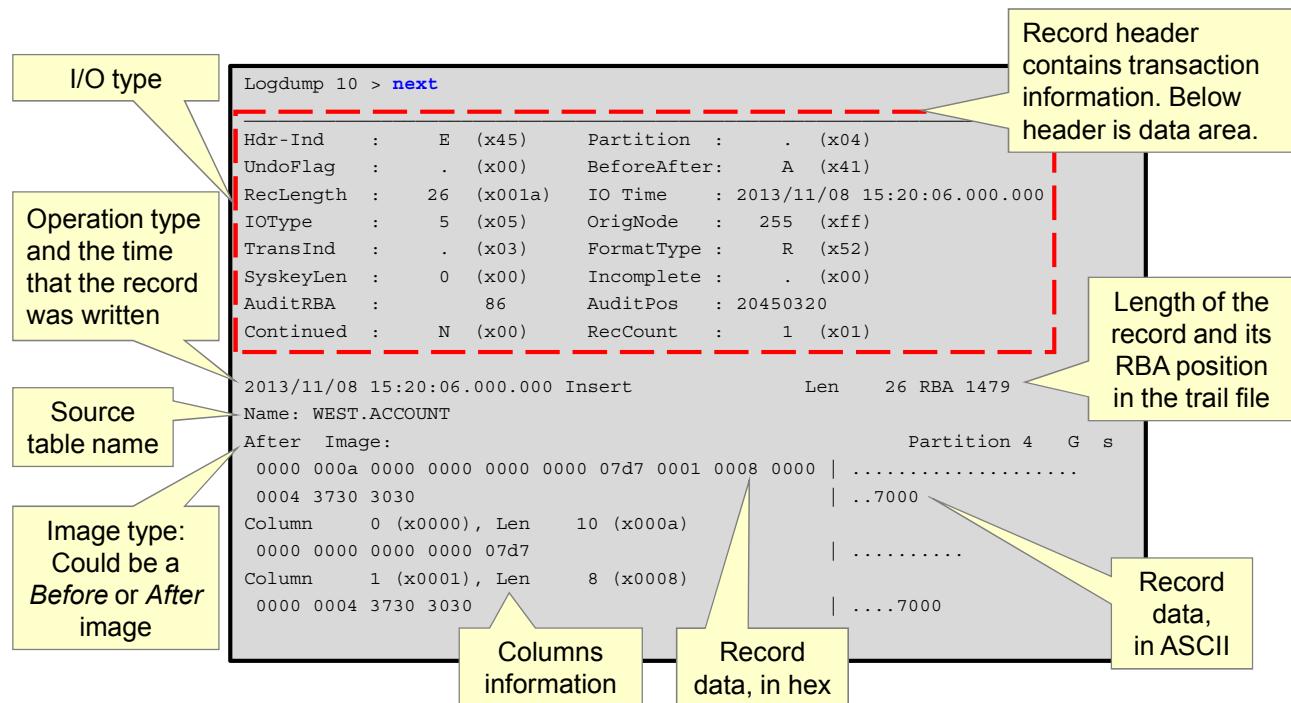
ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

position | pos [<RBA> | 0 | FIRST]: You can position on the first record using 0 or FIRST or on a relative byte address.

scanforheader | sfh [prev]: Use scanforheader to go to the next record header. Adding the prev option will display the previous header. Before using this command, use the ghdr on command to show record headers.

Viewing Canonical Trail Records



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate trail files are unstructured. The Oracle GoldenGate record header provides metadata of the data contained in the record and includes the following information:

- The operation type, such as an insert, an update, or a delete
- The transaction indicator (TransInd): 00 beginning, 01 middle, 02 end, or 03 whole of transaction
- The before or after indicator for updates
- Transaction information, such as the transaction group and commit timestamp
- The time that the change was written to the Oracle GoldenGate file
- The type of database operation
- The length of the record
- The relative byte address within the Oracle GoldenGate file
- The table name

The change data is shown in hex and ASCII format. If before images are configured to be captured (for example, to enable a procedure to compare before values in the WHERE clause), a before image also would appear in the record. The format varies slightly in different GoldenGate versions.

Counting Records in the Trail

```
Logdump> count

LogTrail /ggs/dirdat/rt000000 has 4828 records
Total Data Bytes           334802
  Avg Bytes/Record         69
  Delete                   900
  Insert                   3902
  FieldComp                26
  Before Images             900
  After Images              3928

Average of 25 Transactions
  Bytes/Trans .....      22661
  Records/Trans ...        193
  Files/Trans .....          8
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The basic output, without options, shows the following:

- The RBA where the count began
- The number of records in the file
- The total data bytes and average bytes per record
- Information about the operation types
- Information about the transactions

Counting Records in the Trail

TCUSTMER	
Total Data Bytes	10562
Avg Bytes/Record	55
Delete	300
Insert	1578
FieldComp	12
Before Images	300
After Images	1590
TCUSTORD	
Total Data Bytes	229178
Avg Bytes/Record	78
Delete	600
Insert	2324
Field Comp	14
Before Images	600
After Images	23388

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Syntax:

```
COUNT
[, DETAIL]
[, END [TIME] <time_string>]
[, FILE <specification>]
[, INT [ERVAL] <minutes>]
[, LOG] <wildcard>
[, START [TIME] <time_string>]
```

COUNT options enable you to:

- Show table detail without using the DETAIL command first
- Set a start and end time for the count
- Filter the count for a table, data file, trail file, or extract file
- Specify a time interval for counts

Filtering by a File Name

```

Logdump 7 > filter include filename TCUST*
Logdump 8 > filter match all
Logdump 9 > n

Hdr-Ind      : E (x45) Partition : . (x00)
UndoFlag     : . (x00) BeforeAfter: A (x41)
RecLength    : 56 (x0038) IO Time   : 2002/04/30 15:56:40.814
IOTYPE       : 5 (x05) OrigNode   : 108 (x6c)
TransInd     : . (x01) FormatType: F (x46)
SyskeyLen    : 0 (x00) Incomplete : . (x00)
AuditRBA     : 105974056

2002/04/30 15:56:40.814 Insert Len 56 Log RBA 1230
File: TCUSTMER Partition 0
After Image:
 3220 2020 4A61 6D65 7320 2020 2020 4A6F 686E 736F | 2 James
Johnso
 6E20 2020 2020 2020 2020 2020 2020 4368 6F75 6472 | n Choudr
 616E 7420 2020 2020 2020 2020 4C41                 LA

Filtering suppressed      18 records

```

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Use the `Filter` option to filter the display based on one or more criteria. `Filename` specifies a SQL table, NonStop data file, or group of files. An asterisk (*) wildcard can be used to specify multiple tables. You can string multiple `Filter` commands together, separating each one with a semicolon, as in the following example:

```
Filter INCLUDE FILENAME fin.act*; Filter RECTYPE 5; Filter MATCH ALL
```

To avoid unexpected results, avoid stringing filter options together with one `Filter` command. For example, the following would be *incorrect*:

```
Filter INCLUDE FILENAME fin.act*; RECTYPE 5; MATCH ALL
```

Without arguments, `Filter` displays the current filter status (ON or OFF) and any filter criteria that are in effect.

Locating a Hex Data Value

```

Logdump 27 > filter inc hex /68656C20/
Logdump 28 > pos 0
Current position set to RBA
Logdump 29 > n

-----  

Hdr-Ind   :     E  (x45)  Partition  :      .  (x00)  

UndoFlag  :     .  (x00)  BeforeAfter:      B  (x42)  

RecLength :    56  (x0038) IO Time    : 2002/04/30  

           16:22:14.205  

IOType    :     3  (x03)  OrigNode   :    108  (x6c)  

TransInd  :     .  (x01)  FormatType:      F  (x46)  

SyskeyLen :     0  (x00)  Incomplete :      .  (x00)  

AuditRBA  : 109406324

2002/04/30 16:22:14.205 Delete          Len  56  Log  

      RBA 64424
File: TCUSTMER          Partition 0
Before Image:  

  3620 2020 4A61 6D65 7320 2020 2020 4A6F 686E 736F | 6  James  

  Johnso  

  6E20 2020 2020 2020 2020 2020 2020 4574 6865 6C20 | n  Ethel  

  2020 2020 2020 2020 2020 2020 4C41  

Filtering suppressed  545 records

```

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The example in the slide includes a hex range.

The Filter command can INCLUDE | EXCLUDE the following options:

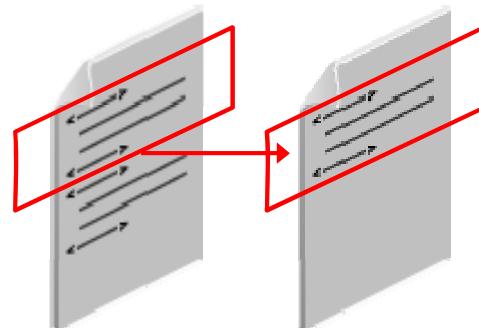
- AUDITRBA <rba> [<comparison operator>]
- CLEAR {<filter_spec> | ALL}
- ENDTIME <time_string>
- FILENAME <name> [, <name>]
- HEX <hex_string> [<byte_range>] [, ...]
- INT16 <16-bit_integer> | INT32 <32-bit_integer>
- IOTYPE <operation type> [, <operation type>]
- MATCH {ANY | ALL}
- DISABLE | OFF
- ENABLE | ON
- CSN | LogCSN

```
PROCESS <process_name>
RBA <byte address> [<comparison operator>] [...]
RECLEN <length> [<comparison operator>]
RECTYPE {<type_number> | <type_name>}
SHOW
STARTTIME <time_string>
STRING [BOTH] [B],<text> [<column_range>]
[[B],<text> [<column_range>]] [...]
SYSKEY <system key> [<comparison operator>] [...]
TRANSIND <indicator> [<comparison operator>]
TYPE <type>
UNDOFLAG <type> [<comparison operator>]
```

Saving Records to a New Trail

To save the 10 records from the current position in the file, issue the following command:

```
Logdump> save newtrail 10 records
```



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Use `Save` to write a subset of the records to a new trail or an extract file. By saving a subset to a new file, you can work with a smaller file. Saving to another file also enables you to extract valid records that can be processed by Oracle GoldenGate, while excluding records that may be causing errors.

`Save` options enable you to overwrite an existing file, save a specified number of records or bytes, suppress comments, use the old or new trail format, set the transaction indicator (first, middle, end, only), and clean out an existing file before writing new data to it.

Syntax:

```
Save <file_name> [!] {<n> records | <n> bytes} [NoComment]  
[OldFormat | NewFormat]  
[TRANSIND <indicator>]  
[Truncate]
```

Keeping a Log of Your Session

- To start and stop the logging of a logdump session, use the `Log` option:

```
Logdump> Log to MySession.txt
```

- When enabled, logging remains in effect for all sessions of Logdump until it is disabled with the `Log Stop` command:

```
Logdump> Log Stop
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

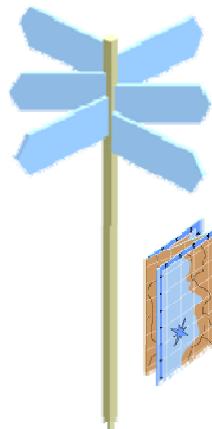
Use `Log` to start and stop the logging of `logdump` sessions. When enabled, logging remains in effect for all sessions of `logdump` until it is disabled with the `Log STOP` command. Without arguments, `Log` displays the status of logging (ON or OFF). An alias for `Log` is `Out`.

Syntax :

```
Log {<file_name> | Stop}
```

Roadmap

- Trail Format
- Alternative Trail Formats
- logdump
- reverse
 - Overview
 - Parameters



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Overview of the `reverse` Utility

The `reverse` utility:

- Reorders operations within Oracle GoldenGate trails in reverse sequence
- Provides the ability to selectively back out of certain operations (such as corrupt data or accidental delete operations) while keeping the rest of the application alive
- Is used to restore a database to a specific point in time, providing the ability to back out all operations during regression testing to restore the original test baseline
- Is an OS command rather than a GGSCI command

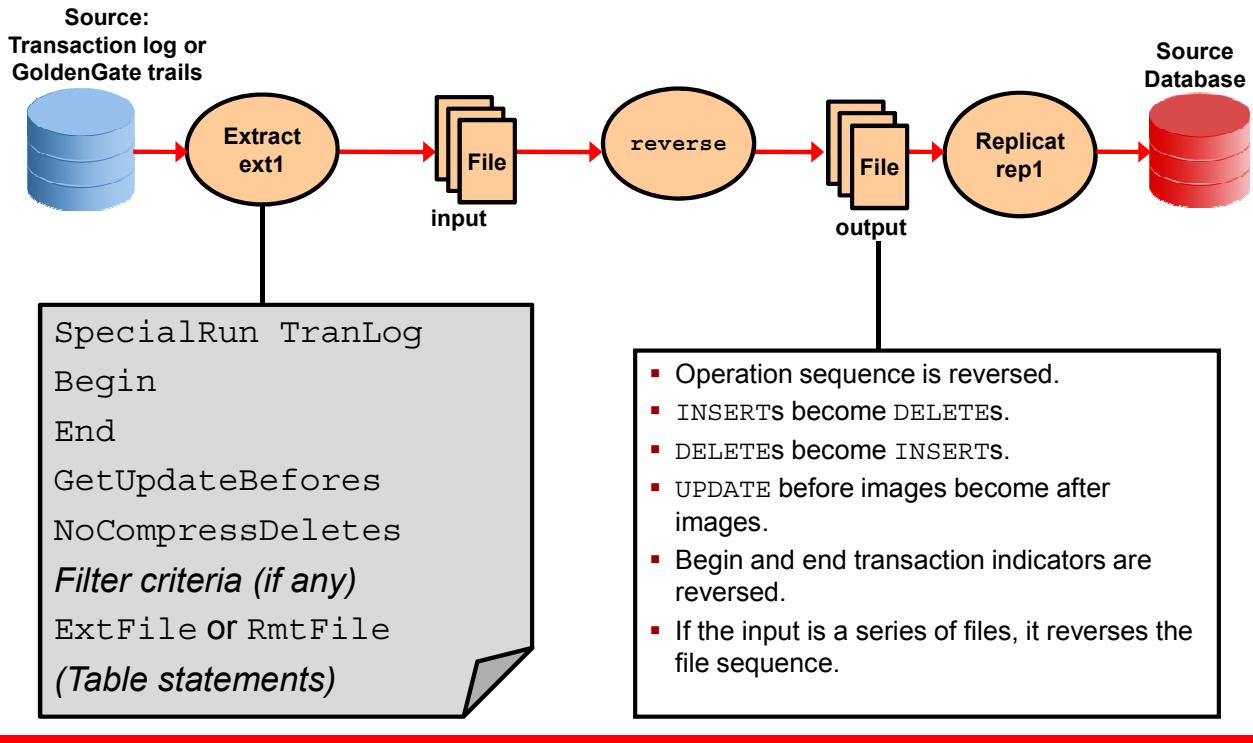


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The `reverse` utility uses before images to undo database changes for specified tables, records, and time periods. It enables you to perform a selective backout, unlike other methods that require restoring the entire database. An alternative to using the `reverse` utility is an Oracle Flashback table feature.

Overall Process of the reverse Utility



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To use Oracle GoldenGate reverse, run:

- Extract to extract the before data
- The `reverse` utility to perform the reversal of the transactions
- Replicat to apply the restored before data to the target database

Oracle GoldenGate `reverse` has the following restrictions:

- If the database does not store the before images of LONG or LOB columns, Oracle GoldenGate might not be able to roll back those columns. A before image is required to reverse update and delete operations.
- Tables with XMLTYPE columns are not supported by Oracle GoldenGate Reverse.
- If the *before* image of your non-LOB or non-LONG data is not selected into your trail file, you should consider using Extract against the transaction logs to reverse the transactions of interest to you.

reverse: Overall Process

- Run extract from either Oracle GoldenGate trails or the source database transaction logs:

```
[OS] ./extract paramfile dirprm/ext1.prm
```

- Run reverse to produce the reordered output:

```
[OS] ./reverse dirdat/input.dat dirdat/output.dat
```

- Run replicat to back out operations:

```
[OS] ./replicat paramfile dirprm/repl.prm
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Reverse utility is run from the OS shell.

- Syntax: REVERSE <sourcefile_or_trail> <targetfile_or_trail>
- Example: C:\GGS> reverse dirdat\rt dirdat\nt

Sample Parameter Files

- Extract parameters (dirprm/ext1.prm):

```
SpecialRun, TranLog
UserID user, Password uassword
Begin 2011-05-30 17:00
End 2011-05-30 18:00
GetUpdateBefores
NoCompressDeletes
RmtHost Target, Mgrport 7809
RmtFile /ggs/dirdat/input.Dat
Table HR.SALES;
Table HR.ACCOUNTS;
```

- Replicat parameters (dirprm/repl1.prm):

```
SpecialRun
End Runtime
UserID user, PASSWORD password
ExtFile /ggs/dirdat/Output.Dat
AssumeTargetDefs
Map HR.SALES, Target HR.SALES;
Map HR.ACCOUNTS, Target HR.ACCOUNTS;
```

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

- SpecialRun indicates a one-time batch process that runs from Begin date and time until End date and time.
- TranLog specifies the transaction log as the data source.
- GetUpdateBefores is used to include before images of update records, which contain record details before an update (as opposed to after images).
- NoCompressDeletes causes Extract to send all column data to the output, instead of sending only the primary key. It enables deletes to be converted back to inserts.
- End Runtime causes the Extract or Replicat to terminate when it reaches process startup time.

Discussion Questions

1. What is a trail?
2. In what formats are Extract trails and files written?
3. Which Oracle GoldenGate utility enables you to view trail contents?



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answers

1. A trail is a series of files on disk where Oracle GoldenGate stores data for further processing.
2. Oracle GoldenGate trail format (canonical), LCR, ASCII, SQL, XML
3. logdump

Summary

In this lesson, you should have learned how to:

- Describe and contrast Extract trails and files
- Describe the formats that Extract trails and files can have
- View Extract trails and files with `logdump`
- Reverse the sequence of operations in an Extract trail or file (to back out changes)



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice 8 Overview: Using Extract Trails and Files

This practice covers using the `logdump` utility.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

9

Configuring Initial Load

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

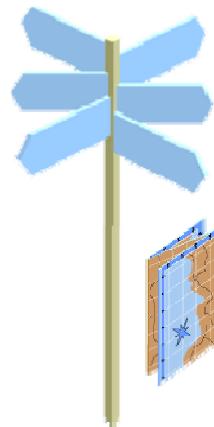
- Describe Oracle GoldenGate initial load methods
- Identify the prerequisites before initiating an initial load
- Explain the advantages of Oracle GoldenGate methods
- Configure an initial load by using Oracle GoldenGate



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Roadmap

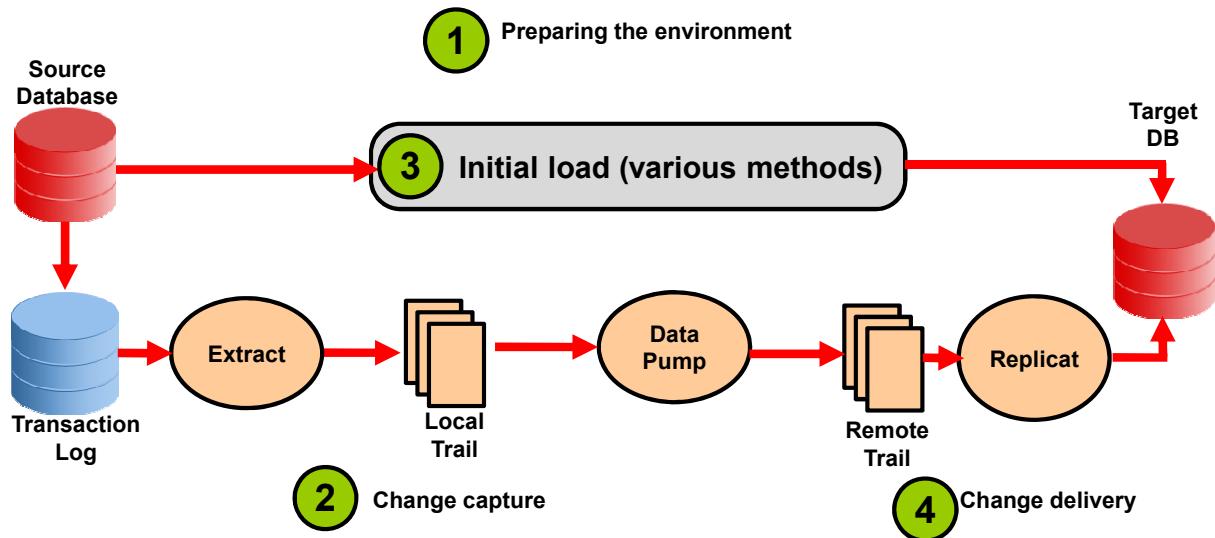
- Introduction
 - Advantages
 - Limitations
 - Prerequisites
- File Based
- Direct Based



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Step 3: Initial Load



ORACLE®

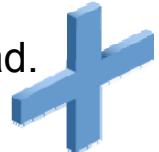
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You have already prepared the environment and configured change capture. The next step is to configure the initial load. The target table exists but may be empty. You need to do DML (INSERT) but not DDL (CREATE). These INSERTs are reading not from the change logs but from the original source table itself.

Note: This initial load could have been done as Step 2 or 3. Because the practice uses a new set of tables, it is okay to do this after the lesson titled “Configuring Change Delivery.”

Initial Load: Advantages of Oracle GoldenGate Methods

- Work across heterogeneous database types and platforms.
- No application down time is required.
- Read directly from source tables without locking tables.
- Fetch data in arrays to speed performance.
- Use parallel processing with `Where` clauses or the `@Range` function.
- Distribute data over multiple network controllers.
- There are flexible load alternatives, including native bulk load utilities.
- Change delivery can handle collisions with initial load.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

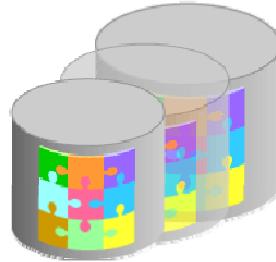
You start change synchronization first during the initial load process. Change synchronization keeps track of ongoing transactional changes while the load is being applied.

Avoiding and handling collisions with initial load is discussed in the lesson titled “Configuring Change Delivery.”

Coordinated Apply can often be used in place of `@RANGE`.

Initial Load: Resource Limitations

- How close are your systems?
- How large are your tables?
- What are the outage time constraints?
- How much disk space do you have to store changes?



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Suppose you are tracking something with a large amount of historical data. If your tables are huge (for example, in the terabyte range), even if the ongoing changes are small, the initial load can be challenging.

Prerequisites for Initial Load

- Disable DDL processing.
- Prepare the target tables.
- Configure the Manager process.
- Create a data definitions file (if the source and target databases have dissimilar definitions).
- Create change-synchronization groups (for capture and replication transactional changes during the initial load).



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Proper prerequisite planning results in a smooth initial load.

Do not use initial load `SpecialRun` or `ExtFile` with the `Integrated` or `Coordinated Replicat` options.

Before executing an initial load, do the following:

- Disable DDL extraction and replication on the table being loaded. You can re-enable it after the initial load is complete.
- Prepare the target tables with:
 - `ALTER TABLE ... DROP UNUSED COLUMNS;`
GoldenGate supports unused columns, but avoid them before an initial load. There is no reason to drag around unused data.
 - `ALTER TABLE ... READ ONLY;`
(Optional) Place the table in read-only mode (if practical) until the initial load is complete. However, this is not required.

Initial Load: Oracle GoldenGate Methods

GoldenGate Method	Extract writes to	Load Method
File to Replicat	Trail (canonical format)	Replicat via SQL
File to database utility	Formatted text file	Database utility
Direct load	Replicat (directly)	Replicat via SQL
Direct bulk load	Replicat (directly)	Replicat via SQL*Loader API

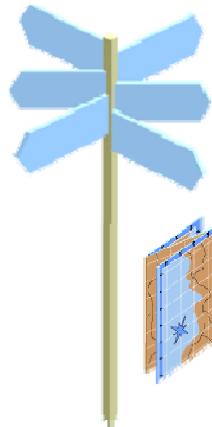


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In an initial load, all the data is selected directly from the source tables rather than the transaction log. Therefore, in an initial load, data values for all columns (including virtual columns) are written to the trail or sent to the target, depending on the method that is being used.

Roadmap

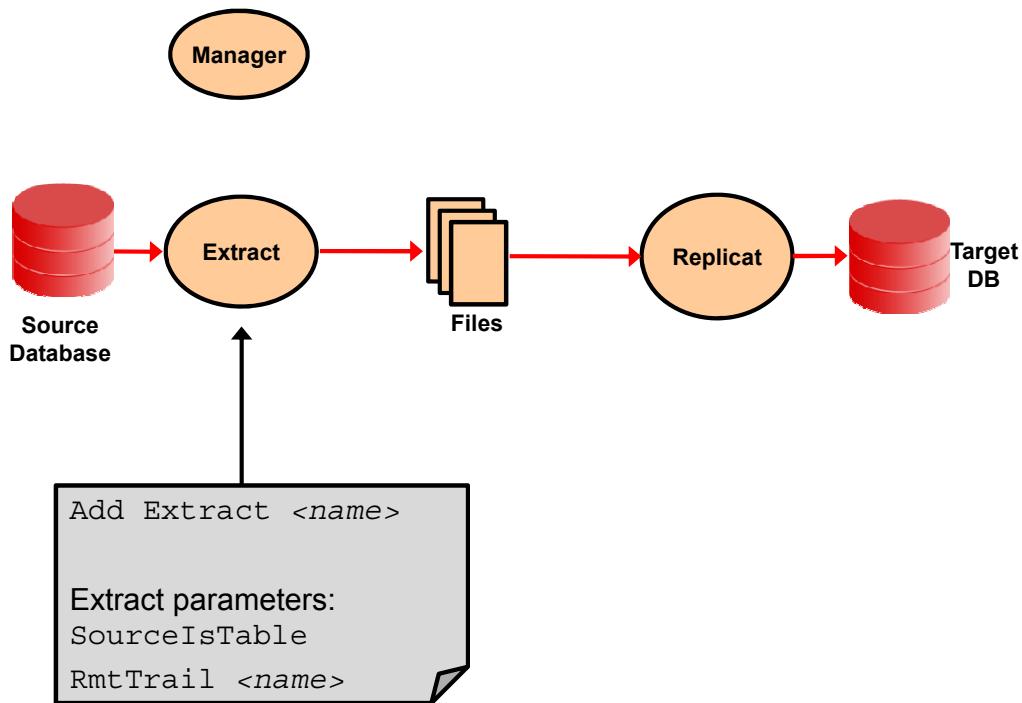
- Introduction
- File Based
 - To Replicat
 - To Database
- Direct Based



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Initial Load: File to Replicat



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

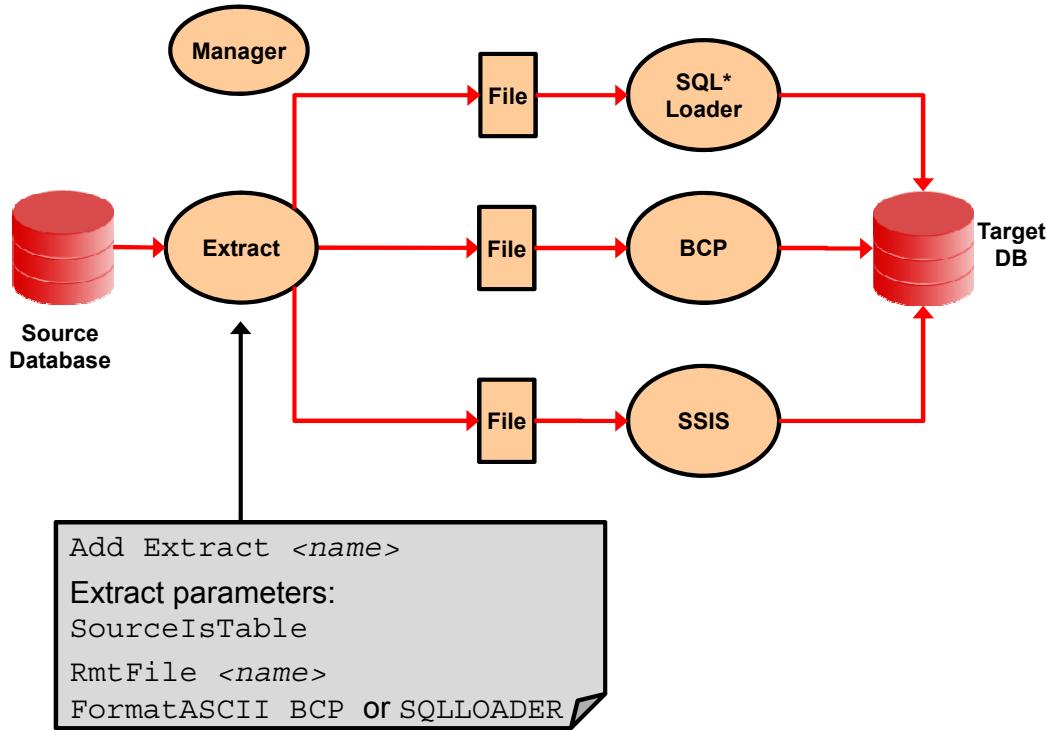
To use Replicat to establish the target data, you use an initial load Extract to extract source records from the source tables and write them to an extract file in canonical format. From the file, an initial load Replicat loads the data by using the database interface. During the load, the change-synchronization groups extract and replicate incremental changes, which are then reconciled with the results of the load.

During the load, the records are applied to the target database one record at a time, so this method is considerably slower than any of the other initial load methods. This method permits data transformation to be done on either the source or the target system. The “file to Replicat” method supports the extraction of LONG and LOB data.

Extract Parameters

SourceIsTable instructs Extract to read the source tables directly rather than from the transaction log. To format the output for processing by Replicat, use RmtTrail or RmtFile. You can use Replicat to perform additional data transformation before loading the data.

Initial Load: File to Database Utility



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To use a database bulk-load utility, use an initial load Extract to extract source records from the source tables and write them to an extract file in plain text ASCII format. The file can be read by Oracle SQL*Loader, Microsoft BCP, DTS, or SQL Server Integration Services (SSIS) utility, or IBM Load Utility (`LOADUTIL`). During the load, the change synchronization groups extract and replicate incremental changes, which are then reconciled with the results of the load. As part of the load procedure, Oracle GoldenGate uses the initial load Replicat to create run and control files required by the database utility.

Any data transformation must be performed by the initial load Extract on the source system, because the control files are generated dynamically and cannot be preconfigured with transformation rules.

Extract Parameters

`SourceIsTable` instructs Extract to read the source tables directly rather than the transaction log.

Oracle GoldenGate checkpoints are not maintained when `SpecialRun` is used. To format the output for native bulk utilities (such as SSIS, BCP, or SQL*Loader), use `RmtFile` or `RmtTrail` and `FormatASCII` with appropriate options, such as `BCP` or `SQLLOADER`.

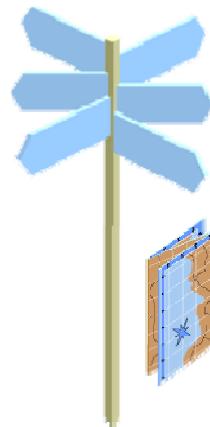
Replicat Parameters

Replicat uses the GenLoadFiles parameter when using the file-to-database-utility initial load method to generate run and control files that are compatible with:

- Oracle SQL*Loader
- Microsoft BCP, DTS, or SQL Server Integration Services (SSIS) utility
- IBM Load Utility (`LOADUTIL`)

Roadmap

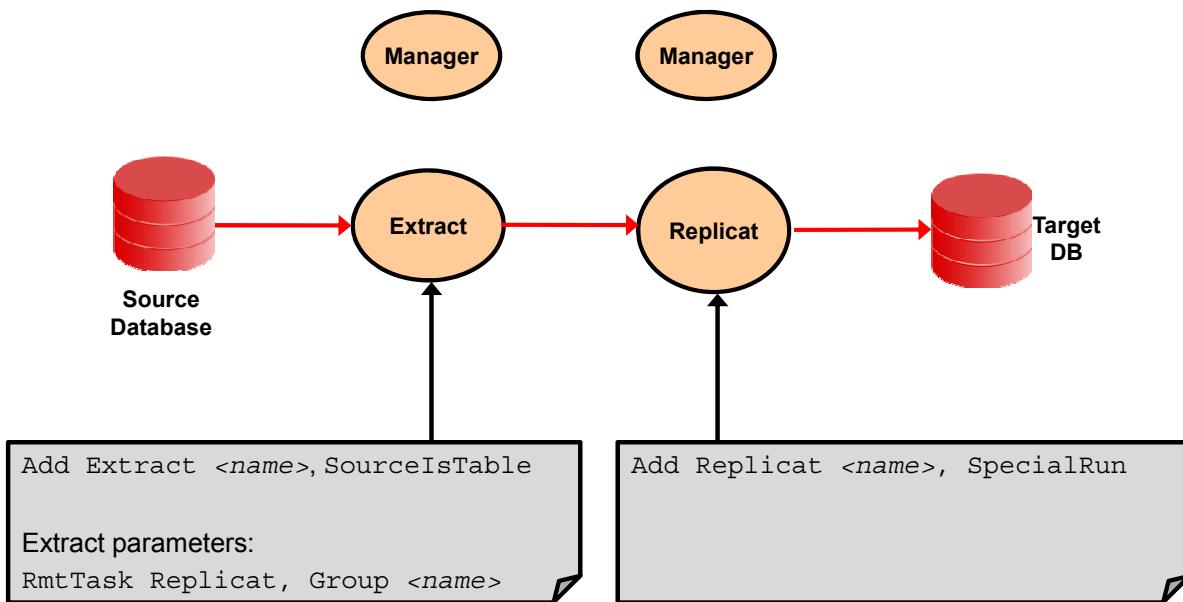
- Introduction
- File Based
- Direct Based
 - Direct Load
 - Direct Bulk Load



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Initial Load: Direct Load



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To use an Oracle GoldenGate direct load, you run an Oracle GoldenGate initial load Extract to extract the source records and send them directly to an initial load Replicat task. A task is started dynamically by the Manager process and does not require the use of a Collector process or file. The initial load Replicat task delivers the load in large blocks to the target database. Transformation and mapping can be done by Extract, Replicat, or both. During the load, the change-synchronization groups extract and replicate incremental changes, which are then reconciled with the results of the load.

The direct load method does not support tables that have columns that contain LOBS, LONGS, user-defined types (UDT), or any other large data type that is greater than 4 KB in size. Transformation can be done with Extract or Replicat.

Extract Parameters

For this example, `RmtTask` (instead of `RmtFile` in the Queue Data method) is used. `RmtTask` instructs the Manager process on the target system to start a Replicat process with a group name specified in the `Group` clause.

If the initial load is from a compressed table, there are some additional constraints:

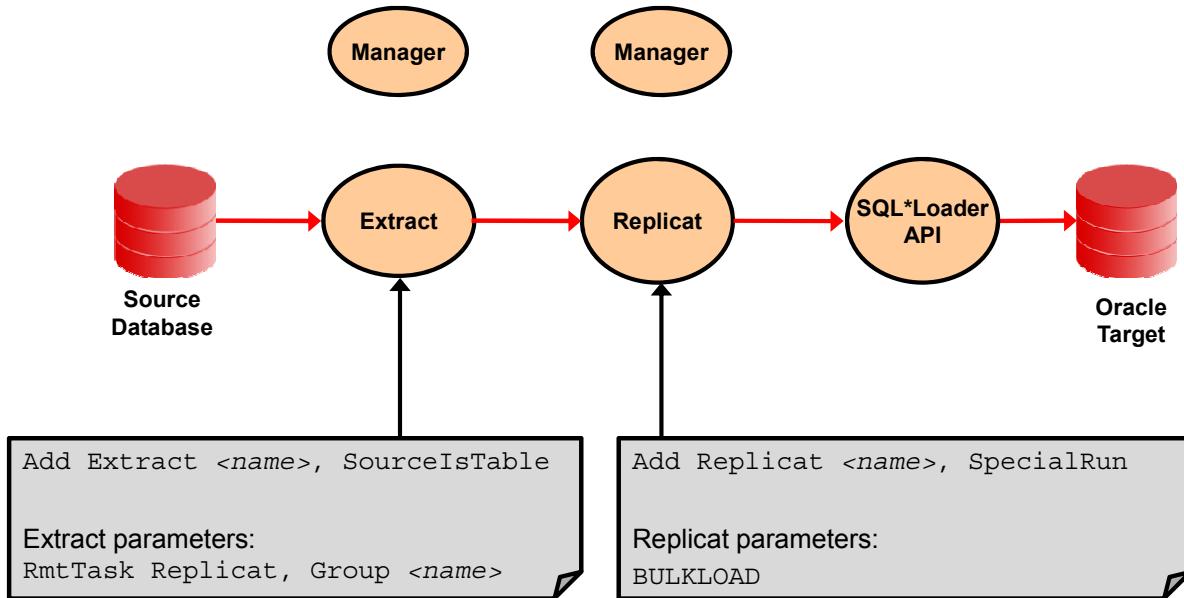
- The Extract must be Integrated (instead of Classic).
- You need to specify the `InsertAppend` command to cause the `Append` hint for the target table.

Execution

When you add Extract and Replicat:

- `SourceIsTable` instructs Extract to read the source tables directly rather than the transaction log. `SpecialRun` on Replicat specifies a one-time batch processing where checkpoints are not maintained.
- The initial data load is then started using the GGSCI command `Start Extract`. The Replicat process is automatically started by the Manager process. The port used by the Replicat process can be controlled by using the `DynamicPortList Manager` parameter.

Initial Load: Direct Bulk Load (to Oracle)



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To use the Oracle SQL*Loader utility to establish the target data, you run an Oracle GoldenGate initial load Extract to extract the source records and send them directly to an initial load Replicat task.

A task is a process that is started dynamically by the Manager process and does not require the use of a Collector process or file. The initial load Replicat task interfaces with the API of SQL*Loader to load data as a direct-path bulk load.

Data mapping and transformation can be done by either the initial load Extract or initial load Replicat, or by both. During the load, the change-synchronization groups extract and replicate incremental changes, which are then reconciled with the results of the load.

The direct bulk load method is the fastest method using Oracle GoldenGate for initial data load. It sends data in large blocks to the Replicat process, which communicates directly with SQL*Loader through an API.

Replicat Parameters

The BULKLOAD parameter distinguishes the direct bulk load method from the direct load method.

Discussion Questions

1. What are the Oracle GoldenGate methods for initial load?
2. Which Oracle GoldenGate command arguments specify that Extract and Replicat run as batch tasks (for example, for initial load)?



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answers

1. File to Replicat: Extract writes to a file for Replicat to load via SQL.
File to database utility: Extract writes to ASCII files formatted for database utilities to load.
Direct load: Extract writes directly to Replicat, which loads via SQL.
Direct bulk load: (Oracle only) Extract writes directly to Replicat, which loads through the SQL*Loader API.
2. Add Extract with SourceIsTable
Add Replicat with SpecialRun

Summary

In this lesson, you should have learned how to:

- Describe Oracle GoldenGate initial load methods
- Explain the advantages of Oracle GoldenGate methods
- Configure an initial load by using Oracle GoldenGate



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice 9 Overview: Configuring the Initial Load

This practice covers the following topics:

- Setting up the initial load by using the “file to Replicat” method
- Setting up the initial data load by using the “direct load” method
- Putting it all together
- Converting from Classic mode to Integrated mode



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

10

Oracle GoldenGate Parameters

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

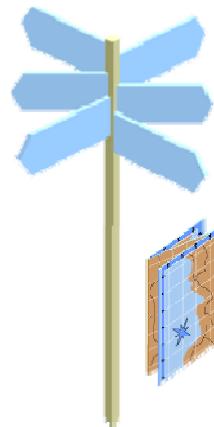
- Edit parameter files
- Compare GLOBALS parameters with process parameters
- Describe commonly used parameters for:
 - GLOBALS
 - Manager
 - Extract
 - Replicat
 - Both Extract and Replicat



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Roadmap

- GLOBALS
 - Overview
 - Examples
- Manager
- Extract
- Replicat



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate Parameter Files

There are two types of parameter files:

- **GLOBALS file (one):** Stores parameters that relate to the Oracle GoldenGate instance as a whole
- **Runtime parameter files (many):** Are coupled with a specific process (such as Extract or Replicat)



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Most Oracle GoldenGate functionality is controlled by means of parameters specified in parameter files. A parameter file is an ASCII file that is read by an associated process. Oracle GoldenGate uses two types of parameter files:

- A GLOBALS file, which cannot have any extension (beware of text editors that try to append .txt to the filename upon creation.) There is at most one GLOBALS file per instance.
- Runtime parameter files, which have an extension of .prm. Examples include:
 - Extract (both primary and data pump)
 - Replicat
 - Manager

There can be many (hundreds or more) parameter files per instance.

Using Parameter Files

- To create a parameter file in GGSCI, use the EDIT PARAM option.
- The GLOBALS parameter file is identified by its file path:

```
GGSCI> Edit Param ./GLOBALS
```

- Manager and utility parameter files are identified by keywords:

```
GGSCI> Edit Param mgr  
GGSCI> Edit Param defgen
```

- Extract and Replicat parameter files are identified by the process group name:

```
GGSCI> Edit Param <group_name>
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Before editing, set your default editor to whatever you prefer:

```
GGSCI> Set Editor notepad  
GGSCI> Set Editor vi  
GGSCI> Set Editor gedit
```

Notepad is the default editor for Windows; vi is the default for Linux. Whatever you set is valid only for that session; it is not retained after exit and there is no way to permanently change the defaults.

You can edit any of the files outside GGSCI as well. You can create the GLOBALS file manually by using any text editor. Make sure that the file name is in uppercase and has no extension. (Editors usually append .txt to the file name, which will break the process.)

GLOBALS Versus Process Parameters

- GLOBALS parameters apply to all processes.
 - These are set when Manager starts.
 - They reside in `<OGG_HOME>/GLOBALS`.
- Process parameters apply to a specific process (Manager, Extract, Server Collector, Replicat, and utilities).
 - These are set when the process starts.
 - They override GLOBALS settings.
 - They reside by default in the `dirprm` directory in files named `<processname>.prm`.
 - Most apply to all the tables that are processed, but some can be specified at the table level.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Parameters manage all Oracle GoldenGate components and utilities, enabling you to customize your data management environment to suit your needs.

Processwide Parameters

These apply to all the tables that are specified in the parameter file for the process. These parameters can appear anywhere in the parameter file, and each should be listed only once in the file. If they are listed more than once, only the last instance of the parameter is active. All other instances are ignored.

Table-Specific Parameters

These control processing for the tables that are specified with a `Table` or `Map` statement. Table-specific parameters enable you to designate one set of processing rules for some tables, while designating other rules for other tables. Table-specific parameters take effect in the order in which each parameter is listed in the parameter file. There are two implementations for file-specific parameters:

- Toggling the parameter on and off for one or more `Table` or `Map` statements
- Adding the parameter within a `Map` statement so that it applies only to that table or file

GLOBALS Parameters

- GLOBALS parameters:
 - Control things that are common to all processes in an Oracle GoldenGate instance
 - Can be overridden by parameters at the process level
- Exit GGSCI for new parameters to take effect.
- After the GLOBALS parameters are set, they are rarely changed.
- Some of the most common parameters include:
 - **MgrServName ggsmanager1**: Defines a unique Manager service name on Windows
 - **CheckPointTable oggadmin.ggschkpt**: Defines the default table name used for the Replicat checkpoint table



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

CheckPointTable: Defines the table name used for Replicat's checkpoint table. You can use any schema.name; the name shown is only by convention.

MgrServName: Is valid only for Windows. It defines the name of the Manager service that is used for starting or stopping the Manager process. This service name is also used when you run the INSTALL utility to add the Windows service.

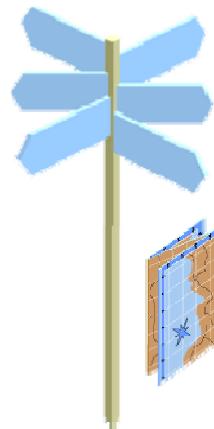
Parameters for Oracle DDL Replication

- **GGSchema**: Specifies the name of the schema that contains the database objects that support DDL synchronization for Oracle
- **DDLTable**: Specifies a nondefault name for the DDL history table that supports DDL synchronization for Oracle
- **MarkerTable**: Specifies a nondefault name for the DDL marker table that supports DDL synchronization for Oracle

After you add or change any GLOBALS parameters, you must exit GGSCI for the new parameters to take effect.

Roadmap

- GLOBALS
- Manager
 - Overview
 - Examples
- Extract
- Replicat

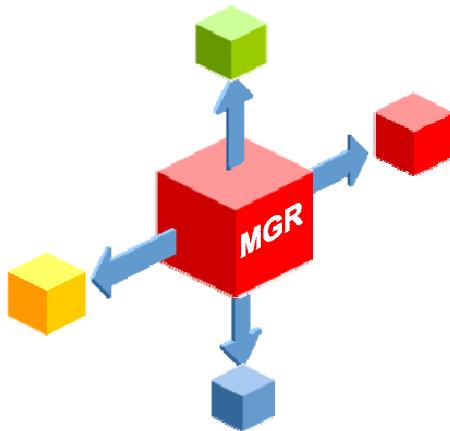


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Manager Parameters: Overview

- The Manager is the Oracle GoldenGate parent process.
- The Manager controls other Oracle GoldenGate processes, resources, user interface, and reporting of thresholds and errors.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You always need to specify unique ports* for each Manager process. The remaining default parameter settings usually suffice. The Manager is not required for the operation of Oracle GoldenGate replication; it is required only for starting, stopping, and changing processes. This allows you to alter and bounce the Manager settings without service disruption.

* **Note:** Actually, a combination of IP address and port makes the Manager unique. So 10.0.0.1:7000 and 10.0.0.2:7000 are considered unique, but it is a good practice to keep the ports themselves dedicated and unique.

Sample Manager Parameter File

```
-- Some Comment with leading double-dashes.  
-- Created by Joe Admin on 10/11/2013.  
Port 7809  
DynamicPortList 9001-9100  
Autostart ER *  
AutoRestart Extract *, WaitMinutes 2, Retries 5  
LagReportHours 1  
LagInfoMinutes 3  
LagCriticalMinutes 5  
PurgeOldExtracts /ggs/dirdat/rt*, UseCheckpoints
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

- **--**: If you copy and paste text into parameter files, beware of editors that try to turn a double-minus into two en dashes (different character) or one big em dash.
- **Port**: Establishes the TCP/IP port number on which Manager listens for requests
- **DynamicPortList**: Specifies the ports that Manager can dynamically allocate
- **Autostart**: Specifies the processes that are to be automatically started when Manager starts
- **AutoRestart**: Specifies the processes to be restarted after abnormal termination
- **LagReportHours**: Sets the interval, in hours, at which Manager checks the lag for Extract and Replicat processing. Alternatively, this can be set in minutes.
- **LagInfoMinutes**: Specifies the interval at which Extract and Replicat will send an informational message to the event log. Alternatively, this can be set in seconds or hours.
- **LagCriticalMinutes**: Specifies the interval at which Extract and Replicat will send a critical message to the event log. Alternatively, this can be set in seconds or hours.
- **PurgeOldExtracts**: Purges the Oracle GoldenGate trails that are no longer needed, based on option settings

Manager Parameter Categories

Category	Parameter Summary
General	Allows comments in the parameter file; filters messages that are written to the system logs
Port Management	Establishes the TCP/IP port; specifies a time to wait before assigning a port number
Process Management	Determines the processes and how long after a failure they are restarted
Event Management	Reports processes that stop abnormally; determines the information that is reported to the error log
Database Login	Provides login information
Maintenance	Maintains Trails, including trail data that is no longer needed

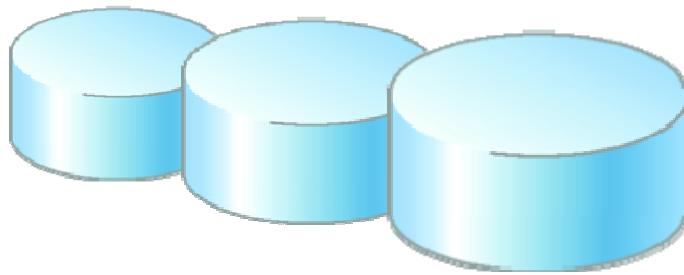


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

For a complete list of Manager parameters and definitions, see the *Oracle GoldenGate Windows and UNIX Reference Guide 12c*.

Managing Trail Files

- Use the PurgeOldExtracts parameter in the Manager parameter file to purge trail files when Oracle GoldenGate has finished processing them.
- The Manager parameter (rather than the Extract or Replicat version of PurgeOldExtracts) is preferred because it enables the trail files to be managed in a more centralized fashion.



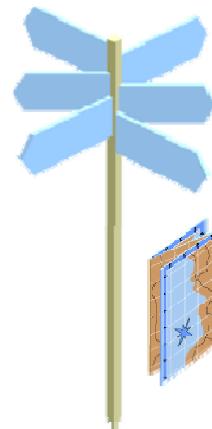
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Trail files, if not managed properly, can consume a significant amount of disk space.

Roadmap

- GLOBALS
- Manager
- Extract
 - Overview
 - Examples
- Replicat



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Extract Parameter: Overview

Extract parameters specify the following:

- Group name (associated with a checkpoint file)
- Where to send data:
 - Local system
 - Multiple remote systems
 - One-to-many Oracle GoldenGate trails
- What is being captured:
 - Which tables
 - Which rows and columns
 - Which operations
- Which column mapping to apply
- Which data transformations to apply



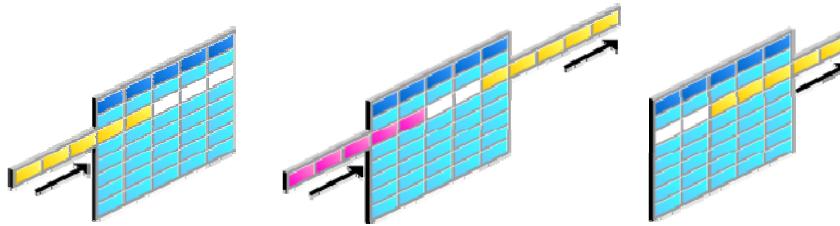
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Extract process captures either full data records or transactional data changes depending on configuration parameters, and then sends the data to a target system to be applied to target tables or processed further by another process such as a load utility.

Extract Parameter Defaults

Extract parameters can be modified or can assume a default value. For insert, update, and delete operations, data can be captured with certain specifications:

- Send data without transformation
- Buffer transactions, either/or
 - Until a block is full
 - Until time elapses
- Based on average transaction volumes



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

For insert, update, and delete operations, data can be captured with the following specifications:

- Committed data only
- Full image for inserts
- Only primary key and changed columns for updates
- Only primary key for deletes
- Only after-image of update

Sample Extract Parameter File

You can use either of the following commands:

- [OS prompt] **more dirprm/somename.prm**
- GGSCI> **View Params somename**

```
-- Created by Joe Admin on 10/11/2013.  
Extract somename  
-- UserIDAlias oggalias  
RmtHost mytarget.example.com, MgrPort 7809  
RmtTrail /ggs/dirdat(rt  
Passthru  
Table SALES.ORDERS;  
Table SALES.INVENTORY;
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

This particular Extract shows a data pump because it has a remote trail. There is another Extract that writes to a local trail.

- **UserID** and **Password** or **UserIDAlias** supply database credentials. (**SourceDB** is not required for Oracle.)
- If running with **Passthru**, the **UserID** lines are not required for data pumps, only for primary extracts.
- **RmtHost** specifies the target system; the **MgrPort** option specifies the port where Manager is running.
- **RmtTrail** specifies the Oracle GoldenGate path and trail file prefix on the target system. The system will append 00000, 00001, 00002, and so on to the file name prefix.
- **Table** specifies a source table for which activity will be extracted.

Extract Parameter Categories

Category	Parameter Summary
General	Verifies parameter file syntax and retrieves variables that were set by other parameters
Processing Method	Determines when a processing run begins and ends
Database Login	Provides login information
Selecting and Mapping Data	Determines the information that is extracted and the format
Routing Data	Provides the location where data is written
Formatting Data	Formats data in a format other than the default Oracle GoldenGate format



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

For a complete list of Extract parameters and definitions, see the *Oracle GoldenGate Windows and UNIX Reference Guide 12c*, section 2.1.3 and following.

General: RecoveryOptions, SourceDB, TCPSourceTimer,
UpdateRecordFormat, Obey, SetEnv.

Processing Method: DBOptions, Extract, GetApplOps, GetReplicates, PassThru,
RmtTask, SourceIsTable, VAM.

Database Login: UserID, UserIDAlias.

Selecting and Mapping Data: CompressDeletes, CompressUpdates, FetchOptions,
LocalSupCols, Sequence, Table, Map, TableExclude,
TargetDefs, TrailCharASCII.

Routing Data: ExtFile, ExtTrail, RmtFile, RmtHost, RmtTrail.

Formatting Data: FormatASCII, FormatSQL, FormatXML, NoHeaders.

There may be other parameters not shown. Not all parameters are supported in all environments.

Extract Parameter Categories

Category	Parameter Summary
Custom Processing	Determines whether to invoke a user exit routine or a macro
Reporting	Displays what information is included in statistical displays
Error Handling	Contains records that cannot be processed and error handling for DDL extraction
Tuning	Controls how long data is buffered before writing to a trail; controls memory allocations
Maintenance	Specifies how often trail files are created or purged
Security	Indicates whether data encryption is enabled in a trail or file

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Extract's behavior can be influenced by many of the documented parameters. Some of these parameters can be used only in the Extract parameter file, whereas others can be used in the Extract, Replicat, or Manager parameter file. These parameters are categorized based on what they can do. The categories include General, Processing Method, Database Login, Selection and Mapping, Formatting, Custom Processing, Reporting, Tuning, Maintenance, and Security. Although you may never memorize all the parameters, you should become familiar with these categories and know where to look in the documentation for more details.

Custom Processing: CUserExit, Include, Macro, SQLExec.

Reporting: CmdTrace, List, Report, StatOptions, ReportCount, Trace.

Error Handling: DDLError, DiscardFile.

Tuning: BR, CacheMgr, FlushSecs, ThreadOptions, AllocFiles, CheckpointSecs, DBOptions, EOFDelay, NumFiles.

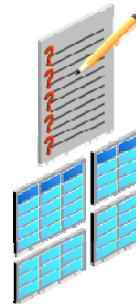
Maintenance: DiscardRollover, PurgeOldExtracts, ReportRollover, Rollover.

Security: EncryptTrail, DecryptTrail.

There may be other parameters not shown. Not all parameters are supported in all environments.

Extract Example: Table Parameter

- Use the Table parameter in an Extract parameter file to specify objects for extraction.
- Some of the Table options do the following:
 - Select and filter records.
 - Select and map columns.
 - Transform data.
 - Designate key columns.
 - Define user tokens.
 - Trim trailing spaces.
 - Pass a parameter to a user exit.
 - Execute stored procedures and queries.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

There must be a Table statement for each source table from which you will be extracting data. Use wildcards (asterisks and question marks) to specify multiple tables with one Table statement, as in the following examples:

```
Table acct*;  
Table myschema.acct*;  
Table plug?.my*.acct*;
```

Version 12c supports table name wildcarding even in three-part names for container (pluggable) databases.

Extract Example: TranLogOptions Parameter

- Use the TranLogOptions parameter to control database-specific aspects of log-based extraction.
- Several options control the archive log.
 - To specify an alternative log format:

```
TranLogOptions AltArchivedLogFormat log_%t_%s_%r.arc
```

- To specify an alternative archive log location:

```
TranLogOptions AltArchiveLogDest /oradata/archive/log2
```

- To cause Extract to read from the archived logs exclusively:

```
TranLogOptions ArchivedLogOnly
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Many of the TranLogOptions parameters are vendor specific.

Use the TranLogOptions parameter to control aspects of the way that Extract interacts with transaction logs. You can use multiple TRANLOGOPTIONS statements in the same parameter file or you can specify multiple options within the same TranLogOptions statement.

For example, ArchivedLogOnly causes Extract to read from Oracle archived logs exclusively, without querying or validating the logs from system views such as v\$log and v\$archived_log. This is called “Archived Log Only” mode (ALO). ALO mode can be used to support a physical or logical standby database. For requirements and more information, see the *Oracle GoldenGate Oracle Installation and Setup Guide*. By default, Extract does not use Archived Log Only mode even if the database that it connects to is a physical standby database.

Extract Example: TranLogOptions Parameter

Additionally, there are options for loop prevention.

- To specify the name of the Replicat database user so that those transactions are not captured by Extract:

```
TranLogOptions ExcludeUser ggsrep
```

- To specify the transaction name of the Replicat database user so that those transactions are not captured by Extract:

```
TranLogOptions ExcludeTrans "ggs_repl"
```

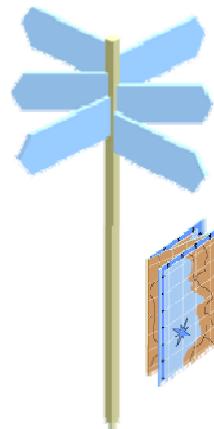


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

These TranLogOptions for loop detection are helpful for bidirectional models. Loop prevention is covered in the lesson titled “Bidirectional Replication.”

Roadmap

- GLOBALS
- Manager
- Extract
- Replicat
 - Overview
 - Examples
 - Options common to both Replicat and Extract



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Replicat Parameters: Overview

Replicat parameters specify the following:

- A group name that is also associated with a checkpoint file
- A list of source-to-target relationships:
 - Optional row-level selection criteria
 - Optional column mapping facilities
 - Optional transformation services
 - Optional stored procedure or SQL query execution
- Error handling
- Various optional parameter settings



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Replicat process runs on the target system, reads the extracted data, and replicates it to the target tables. Replicat reads extract and log files sequentially, and processes the inserts, updates, and deletes specified by selection parameters. Replicat reads extracted data in blocks to maximize throughput.

Optionally, you can filter the rows that you do not want to deliver, as well as perform data transformation before replicating the data. Parameters control the way Replicat processes—how it maps data, uses functions, and handles errors. You can configure multiple Replicat processes for increased throughput and identify each by a different group name.

Replicat Parameter Defaults

Replicat parameters can be modified or can assume a default value:

- Apply all insert, update, or delete operations.
- Smart transactional grouping is possible.
 - 1,000 source operations are grouped into a single target transaction.
- Process ABENDS on any operational failure:
 - Rollback of transactions to the last good checkpoint
 - Optional error handling
 - Optional mapping to secondary table for exceptions



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Replicat supports a high volume of data replication activity. As a result, network activity is block based rather than one record at a time. The SQL operations that are used to replicate operations are compiled once and executed many times, resulting in virtually the same performance as precompiled operations.

Replicat preserves the boundaries of each transaction while processing, but small transactions can be grouped into larger transactions to improve performance. Like Extract, Replicat uses checkpoints so that processing can be restarted without repetition or loss of continuity after a graceful stop or a failure.

Sample Replicat Parameter File

```
-- Created by Joe Admin on 10/11/2013.

Replicat salesrpt
SetEnv (ORACLE_SID = 'myorcl')
UserID ggsuser@myorcl, Password ggspass
-- UserIDAlias oggalias
AssumeTargetDefs
DiscardFile /ggs/dirrpt/SALESRPT.dsc, Append
Map HR.STUDENT, Target HR.STUDENT
    Where (STUDENT_NUMBER < 400000);
Map HR.CODES, Target HR.CODES;
Map SALES.ORDERS, Target SALES.ORDERS,
    Where (STATE = "CA" AND OFFICE = "LA");
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

- **Replicat** names the group that links the process, checkpoints, and log files together.
- Indicate the SID in either the Set Environment or the User ID (probably, not both).
- **UserID** and **Password** provide the credentials to access the database.
- **AssumeTargetDefs** specifies that the table layout is identical on the source and target.
- **DiscardFile** identifies the file to receive records that cannot be processed for any reason. Records will be appended or the file will be purged at the beginning of the run, depending on the options. Starting with version 12.1.2, creation of the discard file is the default and does not need to be specified. If you do not specify this parameter, the file name will be the process name. If it is a Coordinated Replicat, the file name will be the process name plus the thread ID. If the file already exists, you must specify Append or Purge, else you will get an error. See also **DiscardRollover**. By default, the discard file rolls over for each new process start. Rolled file names are the process name plus a one-digit sequence number starting with 0 (similar to the reports).
- **Map** links the source tables to the target tables and applies mapping, selection, error handling, and data transformation, depending on options. Because the **Map** statements are terminated with a semi-colon, they may span multiple lines in the parameter text file without any special continuation characters.

Replicat Parameter Categories

Category	Parameter Summary
General	Verifies parameter file syntax and retrieves variables that were set by other parameters
Processing Method	Determines when a processing run begins and ends
Database Login	Provides login information
Selecting, Converting, and Mapping Data	Specifies the information that is replicated and the format
Routing Data	Defines the name of the Extract file or trail that contains the data to be replicated



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

For a complete list of Replicat parameters and definitions, see the *Oracle GoldenGate Windows and UNIX Reference Guide 12c*.

General: TargetDB, HaveUDFWithinChar, Obey, SetEnv.

Processing Method: Begin, BulkLoad, End, GenLoadFiles, Replicat, SpecialRun.

Database Login: UserID, UserIDAlias.

Selecting, Converting, and Mapping Data: AllowNoUpdates, ApplyNoUpdates, AssumetargetDefs, InsertAllRecords, InsertDeletes, InsertMissingUpdates, InsertUpdates, Table, Map, MapExclude, SourceCharSet, SourceTimeZone, SpacesToNull, UpdateDeletes, UseDedicatedCoordinationThread.

Routing Data: ExtFile, ExtTrail.

There may be other parameters not shown. Not all parameters are supported in all environments.

Replicat Parameter Categories

Category	Parameter Summary
Custom Processing	Determines whether to invoke a user exit routine or a macro
Reporting	Displays the information that is included in statistical displays and the number of records processed
Error Handling	Determines how Replicat manages errors for duplicate or missing records
Tuning	Controls the parameters for how fast Replicat can process data and memory allocations
Maintenance	Specifies how often discard files are created or obsolete trail files are purged
Security	Indicates whether to decrypt data in a trail or an extract file



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Custom Processing: CUserExit, Include, Macro, MacroChar, SQLExec.

Reporting: CmdTrace, List, Report, StatOptions, ReportCount, Trace, WarnRate.

Error Handling: DDLLError, DiscardFile, HandleCollisions, HandleTPKUpdates, OverrideDups, RestartCollisions, RepError, ShowSyntax.

Tuning: BatchSQL, CoordStatInterval, CoordTimer, DeferApplyInterval, GroupTransOps, InsertAppend, MaxDiscardRecs, MaxSQLStatements, MaxTransOps, NumFiles, TransactionTimeout.

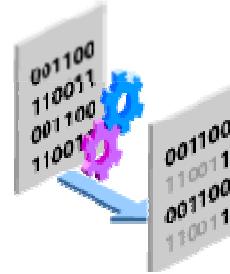
Maintenance: DiscardRollover, PurgeOldExtracts, ReportRollover.

Security: EncryptTrail, DecryptTrail.

There may be other parameters not shown. Not all parameters are supported in all environments.

Replicat Example: Map Parameter

- The `Map` parameter establishes a relationship between one source and one target table.
- With the `Map` parameter, particular subsets of data can be replicated to the target table.
- `Map` also enables users to map certain fields or columns from the source record to the target record format (“column mapping”).



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The `Map` parameter establishes a relationship between one source and one target table. Insert, update, and delete records originating in the source table are replicated in the target table. The first `<table spec>` is the source table. With `Map`, you can replicate particular subsets of data to the target table [for example, `Where (STATE = "CA")`]. In addition, `Map` enables the user to map certain fields or columns from the source record to the target record format (“column mapping”). You can also include a `Filter` command with built-in functions to evaluate data for more complex filtering criteria.

The syntax for the `Map` parameter includes the following:

- `Map <table spec>` specifies the source object.
- `Target <table spec>` specifies the target object.
- `Def <definitions template>` specifies a source-definitions template.
- `TargetDef <definitions template>` specifies a target-definitions template.
- `ColMap` maps records between different source and target columns.
- `EventActions (<action>)` triggers an action based on a record that satisfies a specified filter criterion or (if no filter condition) on every record.

- **ExceptionsOnly** specifies error handling within an exceptions Map statement.
- **ExitParam** passes a parameter in the form of a literal string to a user exit.
- **Filter** selects records based on a numeric operator. **Filter** provides more flexibility than **Where**.
- **HandleCollisions | NoHandleCollisions** reconciles the results of changes made to the target table by an initial load process with those applied by a change-synchronization group.
- **InsertAllRecords** applies all row changes as inserts.
- **InsertAppend | NoInsertAppend** controls whether or not Replicat uses an Oracle APPEND hint for INSERT statements.
- **KeyCols** designates columns that uniquely identify rows.
- **RepError** controls how Replicat responds to errors when executing the **Map** statement.
- **SQLExec** executes stored procedures and queries.
- **TrimSpaces | NoTrimSpaces** controls whether trailing spaces are trimmed or not when mapping CHAR to VARCHAR columns.
- **Where** selects records based on conditional operators.
- ; terminates the **Map** statement and is required.

DBOptions

DBOptions is used:

- For Extract or Replicat
- After TargetDB, SourceDB, or UserID

Some options are valid for:

- Oracle only
- OEM only
- Multiple vendors

Example:

```
Replicat ename
UserIDAlias oggalias
DBOptions DeferRefConst SuppressTriggers
:
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

DBOptions is a parameter for Extract and Replicat that is placed after the TargetDB or SOURCEDB parameter statement and/or the UserID statement to specify database options. Some of the two dozen options are:

- [SuppressTriggers | NoSuppressTriggers]: For Oracle databases to disable triggers on the target database
- [DeferRefConst]: To defer deferrable constraints
- [FetchBatchSize <num_recs>]: To change the fetch array size the same way you do in SQL*Plus in Oracle
- [LimitRows | NoLimitRows]: To prevent multiple rows from being updated or deleted by the same Replicat SQL statement when the target table does not have a primary or unique key
- [SPTHread | NoSPTHread]: To create a separate database connection thread for stored procedures

Discussion Questions

1. What are some typical Manager parameters?
2. What are some typical Extract parameters?
3. What are some typical Replicat parameters?
4. Where are Oracle GoldenGate parameters documented?



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answers

1. Port, DynamicPortList, Autostart, AutoRestart, Lag parameters, PurgeOldExtracts, and so on. These can be found in the section titled “Manager Parameter Categories.”
2. Extract with group name, database login parameters; ExtTrail or RmtHost and RmtTrail; Table, and so on. These can be found in the section titled “Extract Parameter Categories.”
3. Replicat with group name, database login parameters; SourceDefs or AssumeTargetDefs, DiscardFile, Map, and so on. These can be found in the section titled “Replicat Parameter Categories.”
4. *Oracle GoldenGate Windows and UNIX Reference Guide 12c*

Summary

In this lesson, you should have learned how to:

- Edit parameter files
- Compare **GLOBALS** parameters with process parameters
- Describe commonly used parameters for **GLOBALS**, Manager, Extract, and Replicat



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice 10 Overview: Modifying Parameters

This practice covers modifying the following:

- Source Manager parameters
- Target Manager parameters
- Extract parameters on the source database



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

11

Data Selection and Filtering

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

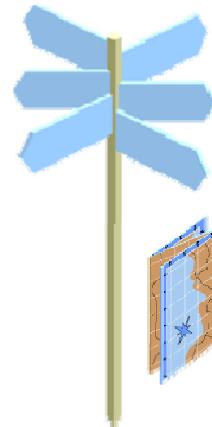
- Select and filter data for replication
- Map columns between different schemas
- Use built-in @ functions
- Use SQLLEXEC to interact directly with a database



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Roadmap

- Overview
 - Mapping
 - Manipulation
 - Definition Files
- Where
- Filter
- @Range
- Mapping
- SQLEXEC



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Data Mapping and Manipulation: Overview

All data selection, mapping, and manipulation is done by using options of the Table (Extract) and Map (Replicat) parameters.



ORACLE®

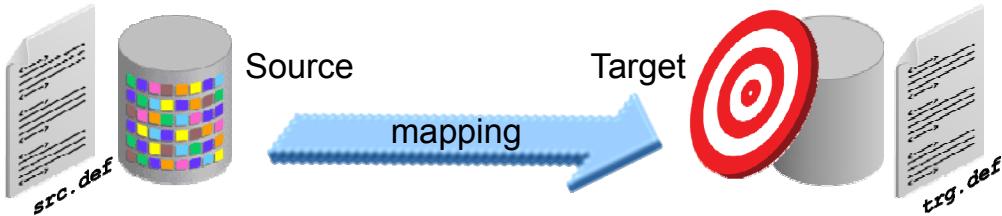
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Data can be integrated between different source and target tables by:

- Selecting records and columns
- Selecting and converting operations
- Mapping dissimilar columns
- Using transaction history
- Testing and transforming data
- Using tokens

Types of Definition Files

- Configuring column mapping or transformation on the target system requires a *source-definitions file*.
- Configuring column mapping or transformation on the source system requires a *target-definitions file*.
- Configuring column mapping or transformation on an intermediary system that contains neither a source database nor a target database requires a *source-definitions file* and a *target-definitions file* on that system.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

- If you are configuring Oracle GoldenGate to perform column mapping or transformation on the target system, a *source-definitions file* is required. The source-definitions file contains the definitions of the source tables. You transfer this file to the target system. Replicat refers to these definitions, plus the queried target definitions, to perform the conversions.
- If you are configuring Oracle GoldenGate to perform column mapping or transformation on the source system, a *target-definitions file* is required. The target-definitions file contains the definitions of the target tables. You transfer the file to the source system. A primary Extract or a data pump refers to these definitions, plus the queried source definitions, to perform the conversions.
- If you are configuring Oracle GoldenGate to perform column mapping or transformation on an intermediary system that contains *neither* a source database nor a target database, you must provide both a *source-definitions file* and a *target-definitions file* on that system.

Data Selection: Overview

Oracle GoldenGate provides the ability to select or filter data based on a variety of levels and conditions:

Parameter/Clause	Selects
Table or Map	Table
Where	Row
Filter	Row, Operation, Range
Table Cols ColsExcept	Columns



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Table Selection

The Map (Replicat) or Table (Extract) parameter is used to select a table:

```
Map sales.tcustord, Target sales.tord;
```

ROWS Selection

The following Where option can be used with Map or Table to select rows for the "AUTO" product type:

```
Where (PRODUCT_TYPE = "AUTO");
```

OPERATIONS Selection

The following can be used with Map or Table to select rows with amounts greater than zero only for update and delete operations:

```
Filter (ON UPDATE, ON DELETE, amount > 0);
```

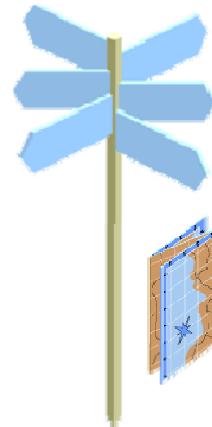
COLUMNS Selection

The Cols and ColsEXCEPT options of the Table parameter enable selection of columns. Use Cols to select columns for extraction and use ColsEXCEPT to select all columns except those designated by ColsEXCEPT. Here is an example:

```
Table sales.tcustord, Target sales.tord, ColsEXCEPT (facility_number);
```

Roadmap

- Overview
- Where
 - Overview
 - Compared to Filter
 - Compared to SQL WHERE
 - Examples
- Filter
- @Range
- Mapping
- SQLEXEC

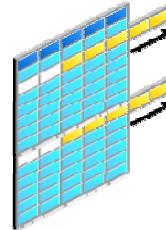


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Data Selection: Where Clause

- The Where () clause is the simplest form of selection.
- The Where () clause appears on either the Map or the Table parameter and must be enclosed in parentheses.
- You cannot use the Where () clause to:
 - Perform arithmetic operations
 - Refer to trail header and user token values
- Use the Filter clause for more complex selections with built-in functions.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Oracle GoldenGate Where clause (shown in mixed case) is similar but *not* identical to the SQL WHERE clause (shown in *uppercase*). In particular, certain operations that are permitted in the SQL WHERE clause are not permitted in the GoldenGate Where clause, such as math. If you need a full SQL query including WHERE, see the section about SQLEXEC later in this lesson.

Examples of using the Where () clause:

```
Map SALES.TCUSTORD, Target SALES.TORD,  
Where (product_amount > 10000);  
Map SALES.TCUSTORD, Target SALES.TORD,  
Where (product_type = "AUTO");
```

Data Selection: Where Clause

You can use `Where` to perform an evaluation for:

<u>Element Description</u>	<u>Example</u>
Columns	<code>PRODUCT_AMT, LAST_NAME</code>
Comparison operators	<code>=, <>, >, <, >=, <=</code>
Numeric values	<code>-123, 5500.123</code>
Literal strings	<code>"AUTO", "Ca"</code>
Field tests	<code>@NULL, @PRESENT, @ABSENT</code>
Conjunctive operators	<code>AND, OR</code>



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Arithmetic operators and floating-point data types are not supported by `Where`.

Data Selection: Where Clause Examples

Returns:

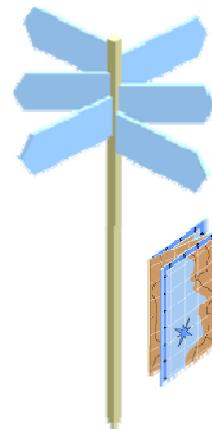
- Only rows where the STATE column has a value of CA
`Where (STATE = "CA");`
- Only rows where the AMOUNT column has a value of NULL
(Note that if AMOUNT is not part of the update, the result is false.)
`Where (AMOUNT = @NULL);`
- Only rows where AMOUNT is a part of the operation and has a value that is not NULL
`Where (AMOUNT @PRESENT AND AMOUNT <> @NULL);`
- Only rows where the account identifier is greater than CORP-ABC
`Where (ACCOUNT_ID > "CORP-ABC");`

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Roadmap

- Overview
- Where
- Filter
 - Overview
 - Compared to Where
 - Examples
- @Range
- Mapping
- SQLEXEC



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Data Selection: Filter Clause

- The Filter clause:
 - Provides complex evaluations to include or exclude data selection
 - Appears on either the Map or the Table parameter and must be enclosed in parentheses
- With Filter, you can:
 - Deploy other Oracle GoldenGate built-in functions
 - Use multiple Filters on one statement
 - If any filter fails, the entire filter clause fails.
 - Include multiple option clauses (for example, on insert or update)
 - Raise a user-defined error for exception processing



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

When multiple filters are specified for a given Table or Map entry, the filters are executed until one fails or until all are passed. Failure of any filter results in a failure for all filters.

Filters can be qualified with operation type so that you can specify different filters for inserts, updates, and deletes.

The Filter RAISEERROR option creates a user-defined error number if the Filter clause is true. In the following example, error 9999 is generated when the BEFORE time stamp is earlier than the CHECK time stamp. This also selects only update operations.

```
Filter (ON UPDATE, BEFORE.TIMESTAMP < CHECK.TIMESTAMP, RAISEERROR 9999) ;
```

Data Selection: Filter Clause Examples

- The following example includes rows where the price multiplied by the amount exceeds 10,000:

```
Filter ((PRODUCT_PRICE * PRODUCT_AMOUNT) > 10000);
```
- The following example includes rows containing the string "JOE":

```
Filter (@StrFind(NAME, "JOE") > 0);
```
- The following example executes the Filter clause for both updates and deletes, but not inserts:

```
Filter (ON UPDATE, ON DELETE, @Compute  
(PRODUCT_PRICE * PRODUCT_AMOUNT) > 10000);
```

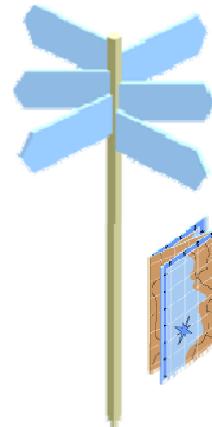


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Filter clause can be used to perform arithmetic and to call Oracle GoldenGate functions.

Roadmap

- Overview
- Where
- Filter
- **@Range**
 - Overview
 - Examples
 - Compare to Coordinated Apply
- Mapping
- SQLEXEC

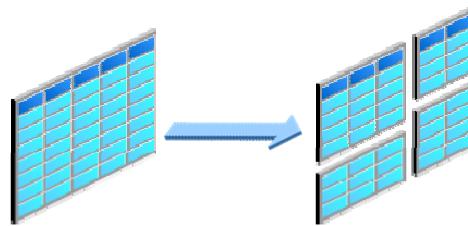


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Data Selection: Range Function

- The @Range function divides a workload into multiple, randomly distributed groups of data.
- It guarantees that the same row is always processed by the same process group.
- It determines the group in which the range falls by computing a hash against the primary key or user-defined columns.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

@Range helps divide the rows of any table across two or more Oracle GoldenGate processes. For example, the @Range function can be used to split the workload based on different key ranges. This is beneficial for a heavily accessed table and multiple Replicat processes.

The user specifies both a range that applies to the current process and the total number of ranges (generally the number of processes). @Range computes a hash value of all the columns specified, or if no columns are specified, the primary key columns of the source table. A remainder of the hash and the total number of ranges is compared with the ownership range to determine whether or not @Range returns true or false. Note that the total number of ranges will be adjusted internally to optimize even distribution across the number of ranges.

Note: @Range cannot be used if primary key updates are performed on the database.

Data Selection: Range Function Examples

- For transaction volume beyond the capacity of a single Replicat, the following example shows three Replicat groups, each processing one-third of the data.
 - Hashing each operation by primary key to a particular Replicat guarantees the original sequence of operations.
- Replicat #1:

```
Map SALES.ACCOUNT,
Target SALES.ACCOUNT, Filter (@Range (1,3));
```

Replicat #2:

```
Map SALES.ACCOUNT,
Target SALES.ACCOUNT, Filter (@Range (2,3));
```

Replicat #3:

```
Map SALES.ACCOUNT,
Target SALES.ACCOUNT, Filter (@Range (3,3));
```

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The example in the slide shows three Replicat processes, with each Replicat group processing one-third of the data in the GoldenGate trail based on the primary key.

The following example uses one Extract process and processes the load into two trails:

```
RmtTrail /ggs/dirddat/aa
Table FIN.ACCT, Filter (@Range (1, 2));
RmtTrail /ggs/dirddat/bb
Table FIN.ACCT, Filter (@Range (2, 2));
```

Note: Because no columns were defined in the second example on which to base the range calculation, the primary key columns are used.

Data Selection: Range Function Examples

```
RmtTrail /ggs/dirdat/aa
Table SALES.REP, Filter (@Range (1,3));
Table SALES.ACOUNT, Filter (@Range (1,3,REP_ID));

RmtTrail /ggs/dirdat/bb
Table SALES.REP, Filter (@Range (2,3));
Table SALES.ACOUNT, Filter (@Range (2,3,REP_ID));

RmtTrail /ggs/dirdat/cc
Table SALES.REP, Filter (@Range (3,3));
Table SALES.ACOUNT, Filter (@Range (3,3,REP_ID));
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Two tables in the SALES schema, REP and ACCOUNT, which are related by REP_ID, require three Replicats to handle the transaction volumes.

By hashing the REP_ID column, related rows are always processed to the same Replicat.

Coordinated Apply (New with 12c)

```
--rep1_old.prm
Map sales.acct, Target sales.acct,
Filter (@RANGE (1,3,ID));
```

```
--rep2_old.prm
Map sales.acct, Target sales.acct,
Filter (@RANGE (2,3,ID));
```

```
--rep3_old.prm
Map sales.acct, Target sales.acct,
Filter (@RANGE (3,3,ID));
```

Replicat (normal)

```
--rep_new.prm
Map sales.acct, Target sales.acct,
ThreadRange (5-8, ID);
```

Coordinated Replicat



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Also known as *Coordinated Replicat*, this feature is compatible with all databases from all vendors. However, certain features are available only with Oracle Database, such as full barrier coordination (serialized DDL).

The Coordinated Apply may replace the need for the @Range and Threads combination.

The lowest thread is the full barrier condition thread. With full barrier coordination, specified by adding the Coordinated parameter to the Map statement, all threads must wait until the barrier thread completes. This could be a significant performance issue if left unmonitored.

The Coordinated Replicat group is created by adding the Coordinated and MaxThreads parameters:

```
GGSCI> Add Replicat rep_new, Coordinated, exttrail ..., MaxThreads 10
```

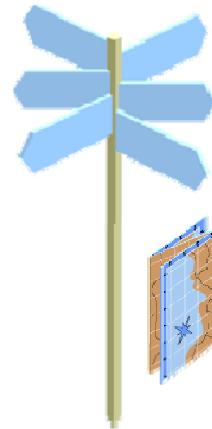
This example creates 11 threads:

- rep_new1 (the Coordinator thread)
- rep_new101 (thread one, the barrier thread, created by default)
- rep_new102, rep_new103, ..., rep_new1010 (the remaining threads)

If MaxThreads is not specified, the default is 25. Note that SpecialRun and ShowSyntax parameters are disabled for coordinated apply.

Roadmap

- Overview
- Where
- Filter
- @Range
- Mapping
 - Column Mapping
 - @ Functions
 - Tests
 - Strings
 - Numbers
- SQLEXEC

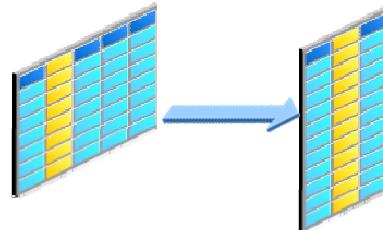


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Column Mapping: Overview

- Oracle GoldenGate provides the capability to map columns from one table to another.
- Data can be transformed between dissimilar database tables.
 - Use ColMap to map target columns from your source columns.
- Oracle GoldenGate automatically matches source to target column names with UseDefaults.
- Mapping can be applied when either extracting or replicating data.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Extract and Replicat provide the capability to transform data between two dissimilarly structured database tables or files. These features are implemented with the ColMap clause in the Table and Map parameters.

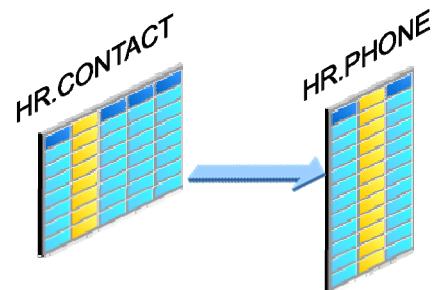
Data Type Conversions

Numeric fields are converted from one type and scale to match the type and scale of the target. If the scale of the source is larger than that of the target, the number is truncated on the right. If the target scale is larger than the source, the number is padded with zeros.

Varchar and character fields can accept other character, varchar, group, and datetime fields, or string literals enclosed in quotation marks. If the target character field is smaller than that of the source, the character field is truncated on the right.

Column Mapping: Example

```
:  
:  
Map HR.CONTACT, Target HR.PHONE,  
    ColMap (USEDEFAULTS,  
        NAME = CUST_NAME,  
        PHONE_NUMBER = @StrCat( "(", AREA_CODE, ")" ,  
            PH_PREFIX, "-", PH_NUMBER ) );
```



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The example in the slide does the following:

- Moves the `HR.CONTACT CUST_NAME` column value to the `HR.PHONE NAME` column
- Concatenates `HR.CONTACT AREA_CODE, PH_PREFIX, and PH_NUMBER` with quote and hyphen literals to derive the `PHONE_NUMBER` column value
- Automatically maps other `HR.CONTACT` columns to the `HR.PHONE` columns that have the same name

Column Mapping: Building History

This example uses special values to build a history of operations data:

```
:  
:  
InsertAllRecords  
Map SALES.ACCOUNT, Target REPORT.ACCTHISTORY,  
    ColMap (USEDEFAULTS,  
            TRAN_TIME = @GetEnv("GGHEADER", "COMMITTIMESTAMP") ,  
            OP_TYPE = @GetEnv("GGHEADER", "OPTYPE") ,  
            BEFORE_AFTER_IND =  
                @GetEnv ("GGHEADER", "BEFOREAFTERINDICATOR") ,  
            );
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

InsertAllRecords causes Replicat to insert every change operation performed on a record as a new record in the database. The initial insert and subsequent updates and deletes are maintained as point-in-time snapshots.

- **ColMap** uses the `@GetEnv` function to get historical data from the GoldenGate trail header.
- **TRAN_TIME** picks up the commit time stamp for the date of the transaction.
- **OP_TYPE** indicates whether it is an insert, an update, or a delete operation.
- **BEFORE_AFTER_IND** indicates whether it is storing a “before” or an “after” image.

Data Transformation Using Functions

- Oracle GoldenGate provides the capability to transform columns by using a set of built-in functions.
- Transformation functions can be applied for either Extract or Replicat.
- You can use additional functions by calling your own logic through user exits.
- Functions are identified with the @ prefix.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Using column conversion functions, you can:

- Perform string and number conversion
- Extract portions of strings or concatenate columns
- Compare strings or numbers
- Perform a variety of date mappings
- Use single or nested IF statements to evaluate numbers, strings, and other column values to determine the appropriate value and format for target columns

Functions: Performing Tests on Column Values

@Function	Description
Case	Enables a user to select a value depending on a series of value tests
Eval	Enables a user to select a value depending on a series of independent tests
If	Selects one of two values depending on whether a conditional statement returns TRUE or FALSE
ColStat	Tests whether a column value is missing, NULL, or invalid
ColTest	Tests whether a column value is present, missing, NULL, or invalid
ValOneOf	Returns TRUE if a column contains one of a list of values



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The functions listed in the slide select a value based on tests against the current value.

@IF Function

Variable = @IF(*condition*, *true_val*, *false_val*)

- This function can be used with other Oracle GoldenGate functions to begin a conditional argument that tests for one or more exception conditions.
- Examples of @IF functions:

- The following returns an amount only if the AMT column is greater than zero. Otherwise, it returns zero.

```
AMOUNT_COL = @IF (AMT > 0, AMT, 0)
```

- The following returns WEST if the STATE column is CA, AZ, or NV. Otherwise, it returns EAST.

```
REGION = @IF (@VALONEOF (STATE, "CA", "AZ",  
                           "NV"), "WEST", "EAST")
```

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The @IF function is used to return one of two values based on a condition.

Functions: Working with Dates

@Function	Description
Date	Returns a date from a variety of sources in a variety of output formats
DateDiff	Returns the difference between two dates or times
DateNow	Returns the current date and time



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The functions listed in the slide return dates in various formats and calculate the difference between two dates.

@Date Function

- The @Date function is used to return dates and times in a variety of formats to the target column based on the format passed into the source column.
- Examples of @Date functions:
 - The following converts year, month, and day columns to a date:

```
date_col = @Date ("YYYY-MM-DD", "YY",
                  date1_yy, "MM", date1_mm, "DD", date1_dd)
```

- The following converts a numeric column that is stored as YYYYMMDDHHMISS to a Julian time stamp:

```
julian_ts_col = @Date ("JTS", "YYYYMMDDHHMISS",
                        numeric_date)
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The @Date function is used to return dates and times in a variety of formats to the target column based on the format passed into the source column. @Date converts virtually any type of input to a valid SQL date. @Date can also be used to extract portions of a date column or to compute a numeric timestamp column based on a date.

Functions: Working with Strings and Numbers

@Function	Description
Compute	Returns the result of an arithmetic expression
NumBin	Converts a binary string into a number
NumStr	Converts a string into a number
StrCat	Concatenates two or more strings
StrCmp	Compares two strings to determine whether they are equal, or whether the first is less or greater than the second
StrEq	Tests to see whether two strings are equal; returns 1 for equal and 0 if not equal
StrExt	Extracts selected characters from a string
StrFind	Finds the occurrence of a string within a string
StrLen	Returns the length of a string



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

These functions convert, compare, extract, trim, and otherwise manipulate strings and numbers.

Functions: Working with Strings and Numbers

@Function	Description
StrLTrim	Trims the leading spaces in a column
StrNCat	Concatenates one or more strings up to a specified number of characters per string
StrNCmp	Compares two strings up to a certain number of characters
StrNum	Converts a number into a string, with justification and zero-fill options
StrRTrim	Trims the trailing spaces in a column
StrSub	Substitutes one string for another within a column
StrTrim	Trims both leading and trailing spaces in a column
StrUp	Changes a string to uppercase



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

@StrCat Function

- The @StrCat function is used to concatenate one or more strings or string (character) columns.
- @StrCat function examples:
 - The following concatenates the LASTNAME and FIRSTNAME columns, separated by a semicolon:

```
NAME = @StrCat (LASTNAME, ";" ,FIRSTNAME)
```

- The following concatenates a country code, area code, and local phone number into an international phone number with hyphens between the components:

```
INTL_PHONE = @StrCat (COUNTRY_CODE, "-",
                      AREA_CODE, "-",
                      LOCAL_PHONE)
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The last example in the slide assumes that the phone “number” is made up of strings.

@StrExt Function

- The @StrExt function is used to extract a portion of a string.
- Example
 - Using three @StrExt functions to extract a phone number into three different columns:

```
AREA_CODE = @StrExt (PHONE, 1, 3),  
PREFIX    = @StrExt (PHONE, 4, 6),  
PHONE_NO  = @StrExt (PHONE, 7, 10)
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The previous example assembled a phone number. This example breaks a compound phone number into its parts.

Note: This is a *string* Extract, which is not related to a *database* Extract.

Other Functions

@Function	Description
Binary	Keeps source data in its original binary format in the target when the source column is defined as character
BinToHex	Converts a binary string to a hexadecimal string
GetEnv	Returns information about the GoldenGate environment, trail file header, trail record header, last replicated operation, and lag; can retrieve the commit time stamp in local time or GMT
GetVal	Extracts parameters from a stored procedure as input to a Filter or ColMap clause
HexToBin	Converts a hexadecimal string to a binary string
Range	Divides a workload into multiple groups of data, while ensuring that the same row is always sent to the same process. Range uses a hash against primary key or user-defined columns.
Token	Maps environmental values that are stored in the user token area to the target column

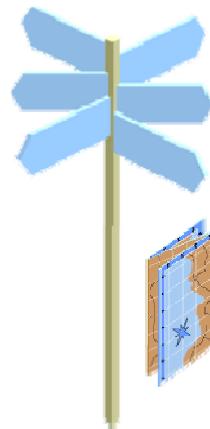


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

These functions work with various number types, converting from one type to another.
@Range was covered earlier in this lesson.

Roadmap

- Overview
- Where
- Filter
- @Range
- Mapping
- SQLEXEC
 - Overview
 - With Procedures
 - With SQL Query
 - Standalone



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

SQLEXEC: Overview

- The SQLEXEC parameter extends Oracle GoldenGate capabilities by enabling Extract and Replicat to communicate with the database through SQL queries or stored procedures.
- SQLEXEC also extends data integration beyond what can be done with Oracle GoldenGate functions.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The SQLEXEC option enables both Extract and Replicat to communicate with the user's database via SQL queries or stored procedures. SQLEXEC can be used to interface with a virtually unlimited set of functionality supported by the underlying database.

Stored Procedure Capabilities

Stored procedures extend the functionality of popular databases such as Oracle Database, DB2, SQL Server, Sybase, and Teradata. Users write stored procedures to perform custom logic, typically involving the database in some way, by using languages such as Oracle's PL/SQL and Microsoft's Transact-SQL.

Extract and Replicat enable stored procedure capabilities to be leveraged for Oracle Database, SQL Server, and DB2. Combining industry-standard stored procedure languages with extraction and replication functions brings a familiar, powerful interface to virtually unlimited functionality.

Stored procedures can also be used as an alternative method for inserting data into the database, aggregating data, denormalizing or normalizing data, or any other function that requires database operations as input. Extract and Replicat can support stored procedures that only accept input, or procedures that produce output as well. Output parameters can be captured and used in subsequent map and filter operations.

SQL Query Capabilities

In addition to stored procedures, Extract and Replicat can execute specified database queries that either return results (`SELECT` statements) or update the database (`INSERT`, `UPDATE`, and `DELETE` statements).

SQLEXEC: Basic Functionality

- Execute a stored procedure or SQL query by using the SQLEXEC clause of the Table or Map parameter.
- (Optional) Extract output parameters from the stored procedure or SQL query as input to a Filter or ColMap clause by using the @GETVAL function.
- Use SQLEXEC at the root level (without input/output parameters) to call a stored procedure, run a SQL query, or issue a database command.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Before you define the SQLEXEC clause, a database logon must be established. This is done via the SourceDB or UserID parameter for Extract and the TargetDB or UserID parameter for Replicat.

When using SQLEXEC, a mapping between one or more input parameters and source columns or column functions must be supplied. When you supply at least one SQLEXEC entry for a given Replicat Map entry, a target table is not required.

SQL EXEC: DBMS and Data Type Support

- SQL EXEC is available for the following databases:

Oracle Database	SQL Server
Teradata	Sybase
DB2	
- The stored procedure interface supports the following data types for input and output parameters:

Oracle	DB2	SQL Server/Sybase/Teradata
CHAR	CHAR	CHAR
VARCHAR2	VARCHAR2	VARCHAR
DATE	DATE	DATETIME
All numeric types	All numeric types	All numeric types
LOBs up to 200 bytes	BLOB data types	



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The stored procedure interface for Oracle Database currently supports the following input and output parameter types:

- CHAR
- VARCHAR2
- DATE
- All available numeric data types
- LOB data types (BLOB and CLOB) where the length is less than 200 bytes
- The ANSI equivalents of the preceding types

The stored procedure interface for SQL Server currently supports the following input and output parameter types:

- CHAR
- VARCHAR
- DATETIME
- All available numeric data types
- Image and text data types where the length is less than 200 bytes

TIMESTAMP parameter types are not supported natively, but you can specify other data types for parameters and convert the data to the TIMESTAMP format within the stored procedure.

The stored procedure interface for DB2 currently supports the following input and output parameter types:

- CHAR
- VARCHAR
- DATE
- All available numeric data types
- BLOB data types

The stored procedure interface for Sybase currently supports all data types except TEXT, IMAGE, and UDT.

The stored procedure interface for Teradata version 12 and later supports all data types that are supported by Oracle GoldenGate.

SQLEXEC: Usage with a LOOKUP Stored Procedure

The following example uses SQLEXEC to run a stored procedure named `lookup` that performs a query to return a description based on a code:

```
CREATE OR REPLACE PROCEDURE lookup
  (code_param IN VARCHAR2,
   desc_param OUT VARCHAR2)
BEGIN
  SELECT desc_col INTO desc_param
  FROM lookup_table
  WHERE code_col = code_param;
END;
```



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Mapping can be augmented with a simple database lookup procedure in Extract or Replicat. The example in the slide illustrates the stored procedure to perform a table lookup.

Note: This is a regular SQL WHERE instead of an Oracle GoldenGate Where (as shown earlier in this lesson). For this reason, this WHERE can perform complex math.

SQLEXEC: Usage with a LOOKUP Stored Procedure

Content of the Map statement:

- Data is mapped from the ACCOUNT table to the NEWACCT table.
- When processing any rows from ACCOUNT, Extract performs the LOOKUP stored procedure before executing the column map.
- Values returned in desc_param are mapped to the newacct_val column by using the @GETVAL function:

```
Map HR.ACOUNT, Target HR.NEWACCT, &
SQLEXEC (SPNAME lookup,
          PARAMS (code_param = account_code)), &
ColMap  (USEDEFAULTS, newacct_id = account_id,
          newacct_val = @GETVAL(lookup.desc_param));
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The example in the slide illustrates how a stored procedure can be used for mapping in a Replicat parameter file.

SQLEXEC: Usage with a SQL Query

For an Oracle database, the following example performs a SQL query directly to return the description. @GETVAL is used to retrieve the return parameter:

```
:  
:  
Map HR.ACOUNT, Target HR.NEWACCT, &  
    SQLEXEC (id lookup, &  
        QUERY " SELECT desc_param FROM lookup_table  
        WHERE code_col = :code_param ", &  
        PARAMS (code_param = account_code)), &  
    ColMap (USEDEFAULTS, newacct_id = account_id,  
        newacct_val = @GETVAL(lookup.desc_param));
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The example parameter file entries illustrate a mapping using a simple SQL query to look up the account description.

Note: The ampersand (&) is used as a continuation character in Oracle GoldenGate parameter files. It must be placed at the end of each line of a parameter statement that spans multiple lines. Most examples in the documentation show the ampersand in its proper place; however, some examples of multiline statements may omit the ampersand to meet the space constraints of the publication format.

SQLLEXEC: Usage in a Table or Map Statement

- When used within a Table or Map statement, SQLLEXEC can pass and accept parameters.
- It can be used for procedures and queries, but not for database commands.
- To execute a procedure in a Table or Map statement:

```
SQLLEXEC (SPNAME <sp_name>,
           [ID <logical_name>,
            {PARAMS <param_spec> | NOPARAMS})
```

- To execute a query in a Table or Map statement:

```
SQLLEXEC (ID <logical_name>, QUERY ' <sql_query> ',
           {PARAMS <param_spec> | NOPARAMS})
```

Note the spaces.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Note: Spaces are required within the ' single straight quotation marks ' around <sql_query>. Although you may get SQLLEXEC to work without the spaces, it is recommended that you include spaces within the quotes.

SQLEXEC: Usage as a Stand-Alone Statement

- When SQLEXEC is used as a stand-alone parameter statement in the Extract or Replicat parameter file, it can execute a stored procedure, query, or database command.
- For these situations, SQLEXEC does not need to be tied to a specific table, and it can be used to perform general SQL operations.

Parameter Syntax	Purpose
<code>SQLEXEC "exec <procedure name>()"</code>	Executes a stored procedure
<code>SQLEXEC "<sql query>"</code>	Executes a query
<code>SQLEXEC "<database command>"</code>	Executes a database command



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

When SQLEXEC is used as a stand-alone parameter statement in the Extract or Replicat parameter file, it can execute a stored procedure, query, or database command. As such, it need not be tied to any specific table, and can be used to perform general SQL operations.

For example, if the Oracle GoldenGate database user account is configured to time out when idle, you could use SQLEXEC to execute a query at a defined interval, so that Oracle GoldenGate does not appear idle.

As another example, you could use SQLEXEC to issue an essential database command (for example, to disable target triggers). A stand-alone SQLEXEC statement cannot accept input parameters or return output parameters.

Quiz

SQLLEXEC can communicate only via a SQL query.

- a. True
- b. False



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Quiz

What is required if you are configuring column mapping or transformation on the target system?

- a. Source-definitions file
- b. Target-definitions file
- c. Both a source-definitions file and a target-definitions file
- d. Neither, because transformations never occur on the target



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: a

Summary

In this lesson, you should have learned how to:

- Select and filter data for replication
- Map columns between different schemas
- Use built-in @ functions
- Use SQLEXEC to interact directly with a database



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice 11 Overview: Data Selection and Filtering

This practice covers increasing performance by:

- Splitting replication loads
- Coordinated Apply



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

12

Additional Transformation Topics

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

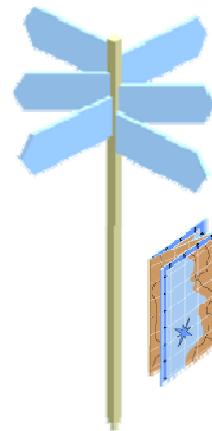
- Create and invoke macros
- Set and retrieve user tokens
- Run user exits in GoldenGate processes
- Replicate Oracle sequences



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Roadmap

- Macros
 - Creating
 - Invoking
 - Parameters
 - Tracing
- Tokens
- User Exits
- Sequences



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Macros: Overview

- Macros make coding easier and more efficient.
- You can use macros in the following ways:
 - Write once and use many times (such as the UserID/Password line).
 - Consolidate multiple statements.
 - Eliminate the need for redundant column specifications.
 - Invoke other macros.
 - Create macro libraries and share across parameter files.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Macros make it easier and more efficient to build parameters (not related to your individual parameter *files*, but commands that can be used across all parameter files).

By using Oracle GoldenGate macros in parameter files, you can easily configure and reuse parameters, commands, and functions. The slide describes how to use macros for a variety of operations to enable the easier and more efficient building of parameters.

Oracle GoldenGate macros work with the following parameter files:

- Manager
- Extract
- Replicat
- GLOBALS

Note: Do not use macros to manipulate data for tables that are being processed by a data pump Extract in Passthru mode.

Creating Macros

- Macros can be defined in any parameter file or library.
- Macro and parameter names must begin with a macro character. (The default is the pound [#] character.)
- Macro statements include the following:
 - Macro name
 - Optional parameter list
 - Macro body

```
Macro #<macro_name>
Params (#<param1>, #<param2>, ...)
BEGIN
<macro_body>
End;
```



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

- **<macro_name>** is the name of the macro. **<macro_name>** must begin with the # character, as in `#macro1`. If the # macro character is used elsewhere in the parameter file, such as in a table name, you can change it to something else with the `MacroCHAR` parameter. Macro names are not case-sensitive.
- **Params (<p1>, <p2> . . .)** describes each of the parameters to the macro. Names must begin with the macro character, as in `#param1`. When the macro is invoked, it must include a value for each parameter named in the `Params` statement. Parameter names are optional and not case-sensitive. `BEGIN` indicates the beginning of the body of the macro and must be specified before the macro body.
- **<macro_body>** represents one or more statements to be used as parameter file input. It can include simple parameter statements, such as `Col1 = Col2`; more complex statements that include parameters, such as `Col1 = #val2`; or invocations of other macros, such as `#colmap (Col1, #sourcecol)`.
- **End;** ends the macro definition.

Invoking a Macro

1. Define a macro:

```

:
Macro #make_date
Params (#year, #month, #day)
BEGIN
    @DATE("YYYY-MM-DD", "CC", @IF(#year < 50, 20, 19),
    "YY", #year, "MM", #month, "DD", #day)
End;

```

2. Invoke the macro:

```

Map SALES.ACCT, Target REPORT.ACCOUNT,
ColMap
( TargetCol1 = SourceCol1,
Order_Date = #make_date(Order_YR,Order_MO,Order_DAY),
Ship_Date = #make_date(Ship_YR,Ship_MO,Ship_DAY)
);

```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The example in the slide demonstrates defining a macro named `#make_date` and calling the macro two different times, with each instance sending a different set of source column values to determine the target column values. The steps are as follows:

1. Define the macro, either inline within a single Extract/Replicat .prm file, or in a library to be included (covered later in this lesson).
2. To invoke the macro, place an invocation statement in the parameter file wherever you want the macro to occur. Use of a macro library would allow you to invoke it from several different parameter files.

Note that the order and ship dates are determined as the result of calling the `make_date` routine to populate the target columns.

Reusing Parameter Sets

1. Define a macro:

```
Macro #option_defaults
  Begin
    GetInserts
    GetUpdates
    GetDeletes
    InsertDeletes
  End;
```

2. Invoke the macro:

```
#option_defaults ()
IgnoreUpdates
Map SALES.SRCTAB, Target SALES.TARGTAB;
#option_defaults ()
Map SALES.SRCTAB2, Target SALES.TARGTAB2;
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Another use of macros is to create a set of frequently used commands. For example, GETINSERTS, GETUPDATES, GETDELETES, and INSERTDELETES may be referenced as a macro within multiple Map statements (as shown in the slide).

Reusing Parameter Sets

3. The macro expands to the following:

```
GetInserts  
GetUpdates  
GetDeletes  
InsertDeletes  
IgnoreUpdates  
Map SALES.SRCTAB, Target SALES.TARGTAB;  
GetInserts  
GetUpdates  
GetDeletes  
InsertDeletes  
Map SALES.SRCTAB2, Target SALES.TARGTAB2;
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Note that the macro's result is altered by the `IgnoreUpdates` parameter for the first `Map` statement.

Creating Macro Libraries

- You can create a macro library that contains one or more macros.
- By using a macro library, you can define a macro once and then use it in many parameter files.
- In this parameter file, use the `Include` command at the beginning of the parameter file:

```
Include /ggs/dirprm/mdateplib.mac
Replicat rep
AssumeTargetDefs
UserIDAlias myoggalias
Map FIN.ACCT_TAB, Target FIN.ACCOUNT;
```

- Macros listing can be toggled by using the `List` and `NoList` parameters.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To use a macro library, use the `Include` parameter at the beginning of a parameter file. The syntax is:

```
Include <macro_name>
```

If the library should include both Extract and Replicat macros, you are responsible for maintaining a copy on both the source and target `/dirprm` directories.

Use the `List` and `NoList` parameters to toggle between showing and hiding the output of libraries. Because of this, `NoList` can be used to reduce the size of the report and the space needed to store the reports.

Tracing Macro Expansion

- Trace macro expansion with the CmdTrace parameter.
- Options for the CmdTrace parameter include ON, OFF (default value), and Detail.
- For this example, tracing is enabled before #testmac is invoked and then it is disabled after the macro's execution:

```
Replicat rep
Macro #testmac
BEGIN
Col1 = Col2,
Col3 = Col4,
End;
...
CmdTrace ON
Map TEST.TABLE1, Target TEST.TABLE2,
ColMap (#testmac);
CmdTrace OFF
```



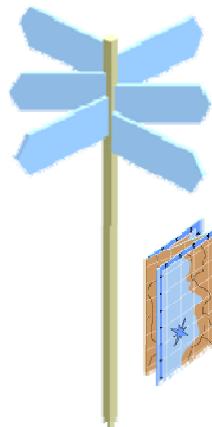
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The macro processor enables tracing of macro expansion for debugging purposes via the CmdTrace command. When CmdTrace is enabled, the macro processor displays macro expansion steps in the process's report file.

The ON option enables tracing, OFF disables it, and DETAIL produces additional details.

Roadmap

- Macros
- Tokens
 - Overview
 - Environmental Variables
 - logdump
- User Exits
- Sequences



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

User Tokens: Overview

- Set token values through a Table TOKENS clause and @GETENV functions.

Example (set for each transaction):

```
Table SALES.PRODUCT,
TOKENS (TKN1 = @GETENV('GGENVIRONMENT', 'OSUSERNAME') ,
TKN2 = @GETENV('GGHEADER', 'COMMITTIMESTAMP') );
```

- Use token values to populate target columns through a Map ColMap clause and @TOKEN functions.

Example:



```
Map SALES.PRODUCT, Target SALES.PRODUCT_HISTORY,
ColMap (USEDDEFAULTS,
OSUSER      = @TOKEN('TKN1'),
TRANSTIME   = @TOKEN('TKN2'));
```

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

User tokens are environment values that are captured and stored in the trail record for replication to target columns or other purposes.

User tokens differ from a macro in the sense that a macro is a set of functions, whereas a user token is a set of variables (or data). To set a token, use the TOKENS option of the Extract Table parameter to define a user token and associate it with data. Tokens enable you to extract and store data within the user token area of a trail record header. You can set tokens to values returned by the @GETENV function (for example, values from the GoldenGate header or environment).

You can use token data in column maps, stored procedures called by SQLEXEC, or macros. For example, use the @TOKEN function in the ColMap clause of a Replicat Map statement to map a token to a target column. If you have both a token and a macro and they are the same, they are both executed. Mapping statements take precedence over tokens. Tokens consist of supplemental data that you can use.

The quotation marks around the tokens must be 'straight single' quotation marks. This is a change from previous versions of Oracle GoldenGate.

Environmental Values Available to @GETENV

Source	Option	Returns values for/from
General	'LAG' 'LASTERR' 'JULIANTIMESTAMP' 'RECSOUTPUT'	Lag (in unit specified) Last failed operation Julian time stamp Number of records written to trail
GoldenGate	'GGENVIRONMENT' 'GGFILEHEADER' 'GGHEADER' 'RECORD'	GoldenGate environment Trail file header Trail record header Trail record location
Database	'DBENVIRONMENT' 'TRANSACTION'	Database environment Source transaction
Operating System	'OSVARIABLE'	OS environmental variable



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

For a detailed list of all parameters and their options, see the *Oracle GoldenGate Windows and UNIX Reference Guide 12c*.

User Tokens Display

Values are stored in the Oracle GoldenGate record header by using a TOKENS clause and @GETENV functions:

```
Extract extdemo
Table SALES.PRODUCT, TOKENS (
(Token name) TKN-OSUSER = @GETENV ('GGENVIRONMENT', 'OSUSERNAME'),
TKN-DOMAIN      = @GETENV ('GGENVIRONMENT', 'DOMAININNAME'),
TKN-COMMIT-TS   = @GETENV ('GGHEADER', 'COMMITTIMESTAMP'),
TKN-BA-IND      = @GETENV ('GGHEADER', 'BEFOREAFTERINDICATOR'),
TKN-TABLE        = @GETENV ('GGHEADER', 'TABLENAME'),
TKN-OP-TYPE     = @GETENV ('GGHEADER', 'OPTYPE'),
TKN-LENGTH       = @GETENV ('GGHEADER', 'RECORDLENGTH'),
TKN-DB-VER       = @GETENV ('DBENVIRONMENT', 'DBVERSION')
);
```



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The example in the slide shows how to store details in the Oracle GoldenGate trail header. Using the TOKENS clause of the Table parameter, the user defines a token identifier (for example, TKN-OSUSER) and specifies the environment category and value by using the @GETENV function.

Using User Tokens

```
Map SALES.ORDER, Target REPORT.ORDER_HISTORY,  
  ColMap (USEDEFAULTS,  
    TKN_NUMRECS = @TOKEN ('TKN-NUMRECS') ;  
  
Map SALES.CUSTOMER, Target REPORT.CUSTOMER_HISTORY,  
  ColMap (USEDEFAULTS,  
    TRAN_TIME = @TOKEN ('TKN-COMMIT-TS'),  
    OP_TYPE = @TOKEN ('TKN-OP-TYPE'),  
    BEFORE_AFTER_IND = @TOKEN ('TKN-BA-IND'),  
    TKN_ROWID = @TOKEN ('TKN-ROWID')) ;
```



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Tokens are retrieved through a Map ColMap clause and @TOKEN functions.

The example in the slide demonstrates how to retrieve values that have been stored as tokens in the GoldenGate trail header. Using the @TOKEN function on the Map parameter, specify the token identifier (for example, TKN-GROUP-NAME) value to use for the target column specification.

Viewing User Tokens in Logdump

```
logdump 2> UserToken On
logdump 3> UserToken Detail
logdump 4> Next

User tokens:
    TKN-HOST          : jemhadar
    TKN-GROUP         : EXTORA
    TKN-BA_IND        : AFTER
    TKN-COMMIT_TS    : 2003-03-24 17:08:59.000000
    TKN-POS           : 3604496
    TKN-RBA           : 4058
    TKN-TABLE         : SOURCE.CUSTOMER
    TKN-OPTYPE        : INSERT
    TKN-LENGTH        : 57
    TKN-TRAN_IND     : BEGIN
    TKN-LAG_SEC       : 1
    TKN-LAG_MIN       : 0
    TKN-LAG_MSEC      : 1229
    TKN-NUMRECS       : 8
    TKN-DBNAME        : ORA901
    TKN-DB_USER       : GGOODRIC
    TKN-DB_VER        : 9.0.1.0.0
    TKN-INAME         : ora901
    TKN-ROWID         : AAABBAABAAAB0BAAF
```

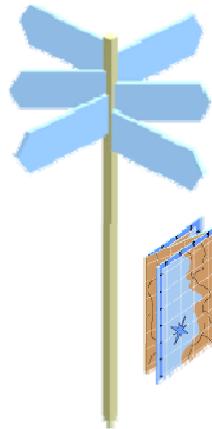


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

After environment values are stored in the trail header, Logdump can display them when the UserTOKEN ON option is used. The UserTOKEN DETAIL option provides additional information.

Roadmap

- Macros
- Tokens
- User Exits
 - Overview
 - Implementing
 - Parameters
 - Samples
- Sequences



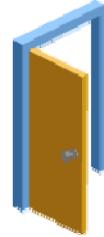
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

User Exits: Overview

User exits:

- Are custom logic written by the customer in C or C++
- Are invoked at different points in Extract or Replicat processing (through the `CUserExit` parameter)
- Enable you to extend or customize the functionality of data movement and integration beyond what is supported through mapping, functions, or `SQLEXEC`
- Can perform an unlimited number of functions



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

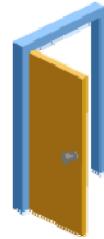
At different points during Extract and Replicat processing, routines can be created in C and invoked to perform an unlimited number of functions. You can employ user exits as an alternative to, or in conjunction with, the column-conversion functions that are available in GoldenGate. User exits can be a better alternative to the built-in functions because, with a user exit, data is processed once (when extracted) rather than twice (extracted and then read again to perform the transformation).

User exits cannot be used for tables that are being processing by a data pump Extract in Passthru mode.

Uses for User Exits

User exits can be used to:

- Perform arithmetic operations or data transformations beyond those possible with GoldenGate built-in functions
- Perform additional table lookups or clean up invalid data
- Respond to events in custom ways (for example, by sending a formatted email message or paging a supervisor based on some field value)
- Accumulate totals and gather statistics
- Perform conflict detection or custom handling of errors or discards
- Enable Oracle GoldenGate for Flat File

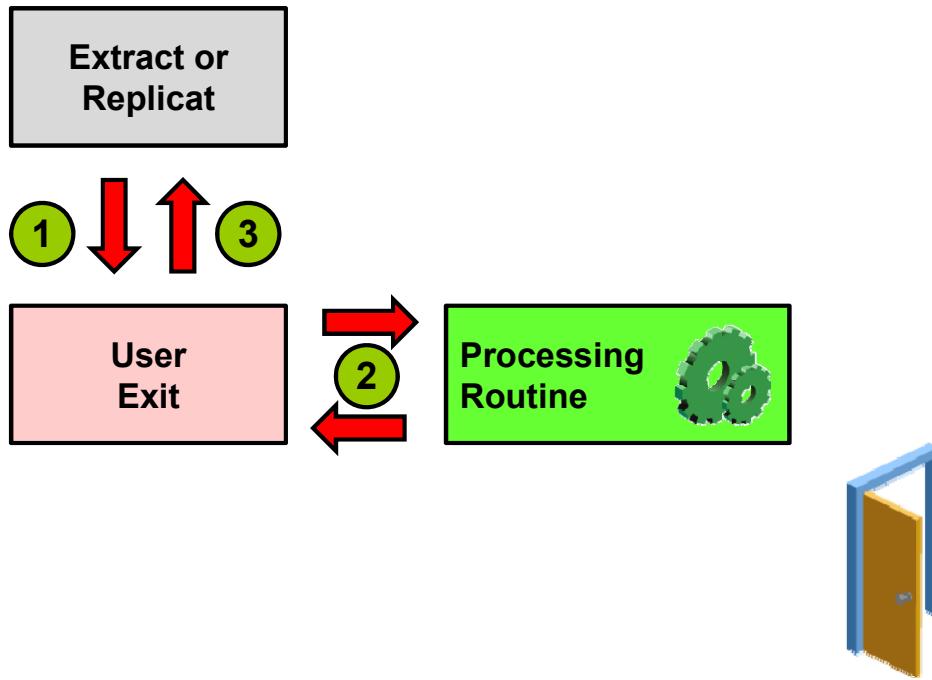


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The gathering of statistics and the sending of event emails based on thresholds are best done by using Oracle GoldenGate Monitor, which is part of the Oracle Management Pack for Oracle GoldenGate.

User Exits: High-Level Processing Logic



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

A user exit:

1. Accepts different events and information from Extract or Replicat
2. Passes the information to an appropriate paragraph or a routine for processing
3. Returns a response and information to the caller

In C, create either a shared object (UNIX systems) or a DLL (Windows), and create or export a routine to be called from Extract or Replicat. This routine is the communication point between Oracle GoldenGate and your routines.

Implementing User Exits

- Windows: Create a DLL in C and create a routine to be called from Extract or Replicat.
- UNIX: Create a shared object in C and create a routine to be called from Extract or Replicat.
- The routine must accept the following parameters:
 - Exit_Call_Type
 - Exit_Call_Result
 - Exit_Parms
- In the source for the DLL or shared object, include the `usrdecs.h` file (in the GoldenGate installation directory).
- Call the `ERCallback` function from the shared object to retrieve record and application context information.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To implement user exits on Windows, perform the following steps:

1. **On Windows systems:** Create a user exit DLL in C and export a routine to be called from Extract or Replicat.
On UNIX systems: Create a shared object in C and create a routine to be called from Extract or Replicat. This routine is the communication point between Extract or Replicat and your routines.

You can define the name of the routine, but it must accept the user exit parameters:

```
Exit_Call_Type
Exit_Call_Result
Exit_Parms
```

2. In the source for the DLL or shared object, include the `usrdecs.h` file. This file contains type definitions, return status values, callback function codes, and other definitions.
3. Include callback routines in the user exit when applicable. Callback routines retrieve record and application context information, and modify the contents of data records.

Extract and Replicat export an `ERCallback` function to be called from the user exit routine.

The user exit must explicitly load the callback function at run time by using appropriate Windows or UNIX API calls.

User Exit Parameters

- `Exit_Call_Type`
- `Exit_Call_Result`
- `Exit_Parms`
- `Exit_Call_Begin_Trans`
- `Exit_Call_End_Trans`
- `Exit_Call_Abort_Trans`
- `ERCallback`



ORACLE®

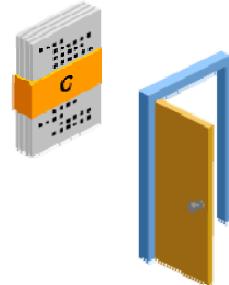
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

- **Exit_Call_Type:** During processing, indicates when the Extract or Replicat process calls the user exit: at start processing, stop processing, begin transaction, end transaction, process record, process marker, discard record, fatal error, or call result
- **Exit_Call_Result:** Provides a response to the routine: OK, ignore, stop, ABEND, or skip record
- **Exit_Parms:** Supplies information to the routine: calling program path and name, function parameter, or “more records” indicator
- **Exit_Call_Begin_Trans:** Called just before the start of a Replicat transaction, or just before the begin record of a transaction is read by data pump
- **Exit_Call_End_Trans:** Called just after the last record in a Replicat transaction, or just after an end record of a transaction is read by data pump
- **Exit_Call_Abort_Trans:** Called when Replicat or data pump reads a Restart Abend record from the trail
- **ERCallback:** Implements a callback routine. Callback routines retrieve record and Oracle GoldenGate context information and modify the contents of data records.

For a detailed list of all parameters and their options, see the *Oracle GoldenGate Windows and UNIX Reference Guide 12c*.

Sample User Exits

- Sample user exit files are located in `<OGG_HOME>/UserExitExamples`.
- Each directory contains the `.c` file as well as makefiles and a `readme.txt` file.
- You will also need `<OGG_HOME>/demo*.sql` to create sample tables.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

- `exitdemo.c` shows how to initialize the user exit, issue callbacks at given exit points, and modify data. The demo is not specific to any database type.
- `exitdemo_passthru.c` shows how the PASSTHRU option of the CUserExit parameter can be used in an Extract data pump.
- `exitdemo_more_recs.c` shows an example of how to use the same input record multiple times to generate several target records.
- `exitdemo_lob.c` shows an example of how to get read access to LOB data.
- `exitdemo_pk_befores.c` shows how to access the before and after image portions of a primary key update record, as well as the before images of regular updates (non-key updates). It also shows how to get target row values with SQLLEXEC in the Replicat parameter file as a means for conflict detection. The resulting fetched values from the target row are mapped to the target record when the target record enters the user exit.

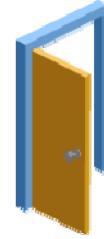
Calling User Exits

- You can call a user exit from Extract or Replicat by using the `CUserExit` parameter.
- Example parameter file syntax on UNIX systems:

```
CUserExit eruserexit.so MyUserExit
```

- Example parameter file syntax on Windows systems:

```
CUserExit eruserexit.dll MyUserExit
```



ORACLE®

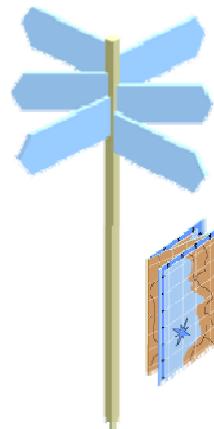
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To call a user exit, include the `CUserExit` parameter in your Extract or Replicat parameter file. This parameter accepts the name of the shared object or DLL and the name of the exported routine that is to be called from Extract or Replicat. You can specify the full path of the shared object or DLL or let the operating system's standard search strategy locate the shared object. The parameter also accepts options to:

- Use a user exit with a data pump that is operating in `Passthru` mode
- Get before images for update operations
- Supply a startup string, such as a startup parameter

Roadmap

- Macros
- Tokens
- User Exits
- Sequences
 - Overview
 - Sanity Check Increment/Decrement



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

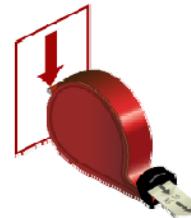
Oracle Sequences: Overview

- Oracle GoldenGate supports the replication of Oracle sequence values.
- Use the Extract Sequence parameter to extract sequence values from the transaction log for propagation to a trail and delivery to other databases. Here is an example:

```
:  
Sequence HR.EMPLOYEES_SEQ;
```

- Use the Replicat Map parameter to apply sequence values to the target, as in the following example:

```
:  
Map HR.EMPLOYEES_SEQ,  
Target PAYROLL.EMPLOYEES_SEQ;
```



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle sequence numbers are used for a variety of purposes, such as to maintain a count, determine order of entry, or generate a primary key value.

Oracle sequences are supported in the following configurations:

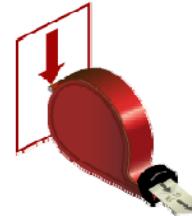
- Oracle GoldenGate online and batch (SpecialRun) change synchronization methods support the replication of sequence values.
- Oracle GoldenGate initial load methods (configurations containing SourceIsTable) do not support the replication of sequence values.
- Oracle GoldenGate does not support the replication of sequence values in a bidirectional configuration.

Similar to the Table command, Sequence ends with a semicolon.

Oracle Sequences: Overview

- Source increment: +10 (positive)
- Source Values: 110, 120, 130, 140, 150...
- Target Values: 120, 130, 140, 140, 140...
 - ✓ ✓ ✓ ✗ ✗

- Source decrement: -10 (negative)
- Source Values: 950, 940, 930, 920, 910...
- Target Values: 940, 930, 920, 930, 940...
 - ✓ ✓ ✓ ✗ ✗



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The default Replicat `CheckSequenceValue` parameter ensures that target sequence values are one of the following

- Higher than the source values (if the increment interval is positive)
- Lower than the source values (if the increment interval is negative)

You can change this default behavior if there are no gaps in the sequence updates (for example, from a trail corruption or process failure) and you want to improve the performance of Oracle GoldenGate.

Gaps are possible in the values of the sequences that Oracle GoldenGate replicates because gaps are inherent, and expected, in the way that sequences are maintained by the database. However, the target values are always greater than those of the source unless the `NoCheckSequenceValue` parameter is used. So a *skipped* value is not necessarily an error!

`CheckSequenceValue` can be used to support a live standby database.

Quiz

User _____ are a set of variables.

- a. Macros
- b. Tokens
- c. Exits
- d. Procedures



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Quiz

User exits can be used for tables that are being processed by a data pump Extract in Passthru mode.

- a. True
- b. False



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Summary

In this lesson, you should have learned how to:

- Create and invoke macros
- Set and retrieve user tokens
- Run user exits in Oracle GoldenGate processes
- Replicate Oracle sequences



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice 12 Overview: Data Transformation

This practice covers the following topics:

- Modifying an existing set of macros
- Using user tokens



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

13

Configuration Options

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

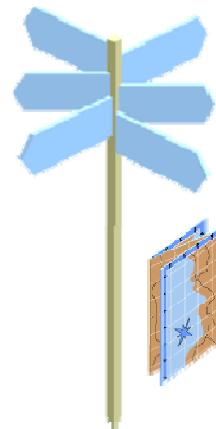
- Use BatchSQL to speed up delivery
- Compress data across the network
- Encrypt messages, trails, and passwords
- Automatically trigger actions based on Event records



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Roadmap

- BatchSQL
 - Syntax
 - Results
- Compression
- Encryption
- Event Actions

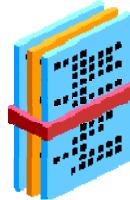


ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

BatchSQL: Overview

- The BatchSQL parameter enhances the performance of Replicat by organizing SQL statements into arrays (rather than individual operations) and applying them at an accelerated rate. Operations containing the same table, operation type (I, U, D), and column list are grouped into a batch.
- Each statement type is prepared once, cached, and executed many times with different variables.
- Referential integrity is preserved.
- BatchSQL can be used with change capture or initial loads.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Operations containing the same table, operation type (Insert, Update, or Delete), and column list are grouped into a batch. For example, each of the following is an example of a batch:

- Inserts to table A
- Inserts to table B
- Updates to table A
- Updates to table B
- Deletes from table A
- Deletes from table B

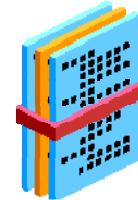
Oracle GoldenGate analyzes parent-child foreign key referential dependencies in the batches before executing them. If referential dependencies exist for statements that are in different batches, more than one statement per batch may be required to maintain referential integrity.

BatchSQL Syntax

BatchSQL is implemented as a parameter in Replicat:

```

:
:
BatchSQL
    [BatchErrorMode | NoBatchErrorMode]
    [BatchesPerQueue <n>]
    [BatchTransOps <n>]
    [BytesPerQueue <n>]
    [OpsPerBatch <n>]
    [OpsPerQueue <n>]
    [Trace]
```



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

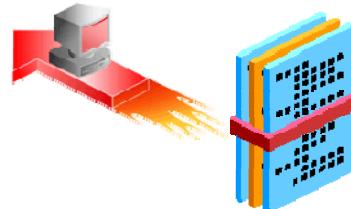
Underscore in a syntax diagram indicates the default value.

- BatchErrorMode and NoBatchErrorMode: Set the response of Replicat to errors
 - In NoBatchErrorMode, Replicat aborts the transaction on an error, temporarily disables BatchSQL, and retries in normal mode.
 - In BatchErrorMode, Replicat attempts to resolve errors without reverting to normal mode. HandleCollisions is required to prevent Replicat from exiting on an error.
- BatchesPerQueue <n | 50>: Sets a maximum number of batches per queue before flushing all batches
Note: A queue is a thread of memory containing captured operations that are waiting to be batched. By default, there is one buffer queue, but you can change this with NumThreads.
- BatchTransOps <n | 1000>: Controls the size of a batch; set to the default or higher
- BytesPerQueue <n | 20000000>: Sets the maximum number of bytes to hold in a queue before flushing batches. The default is 20 MB.

- OpsPerBatch <n | 1200>: Sets the maximum number of rows that can be prepared for one batch before flushing
 - OpsPerQueue <n | 1200>: Sets the maximum number of row operations that can be queued for all batches before flushing
 - Trace: Enables tracing of BatchSQL activity to the console and report file
- To attain optimum balance between the efficiency of memory in the buffer and the use of memory, use the following options to control when a buffer is flushed: BatchesPerQueue, BytesPerQueue, OpsPerBatch, OpsPerQueue, and BatchTransOps. (The last buffer flush for a target transaction occurs when the threshold set with BatchTransOps is reached.)

BatchSQL Results

- Smaller row changes will show a higher gain in performance than larger row changes.
- At 100 bytes of data per row change, BatchSQL has been known to improve Replicat's performance by as much as 400 to 500 percent, but actual performance benefits vary depending on the mix of operations.
- At around 5,000 bytes of data per row change, BatchSQL benefits diminish.



ORACLE®

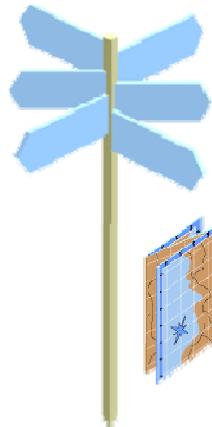
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Some statement types cannot be processed in batches and must be processed as exceptions. When BatchSQL encounters them, it flushes everything in the batch, applies the exceptions in the normal manner of one at a time, and then resumes batch processing. Transaction integrity is maintained. Statements treated as exceptions include:

- Statements containing LOB or LONG data
- Statements containing rows longer than 25 KB in length
- Statements where the target table has one or more unique keys in addition to the primary key. Such statements cannot be processed in batches because BatchSQL does not guarantee correct ordering for nonprimary keys if their values could change content.

Roadmap

- BatchSQL
- Compression
 - Overview
 - Examples
 - EHCC
- Encryption
- Event Actions



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Compression Options

- Oracle GoldenGate provides optional data compression when sending data over TCP/IP.
- Automatic decompression is performed by the Server Collector on a remote system.
- The compression threshold enables the user to set the minimum block size to compress.
- Oracle GoldenGate uses zlib compression.
More information can be found at
www.zlib.net.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Compression can be initiated in either the Extract or Data Pump.

Compression ratios depend on the type of data being compressed. Some data is already precompressed (such as JPEGs).

This TCP compression in transit is unrelated to storage compression within the database itself.

Example of Compression

- Compression is specified on the Extract or Pump RmtHost parameter:

```
RmtHost <host> | <ip address>, MgrPort <port>
[, Compress ]
[, CompressThreshold <bytesize> ]
```

- Compress specifies that outgoing blocks of captured changes are compressed.
 - CompressThreshold sets the minimum byte size for which compression occurs (the default is 1000 bytes).
- Example:

```
RmtHost newyork, MgrPort 7809, Compress,
CompressThreshold 750
```

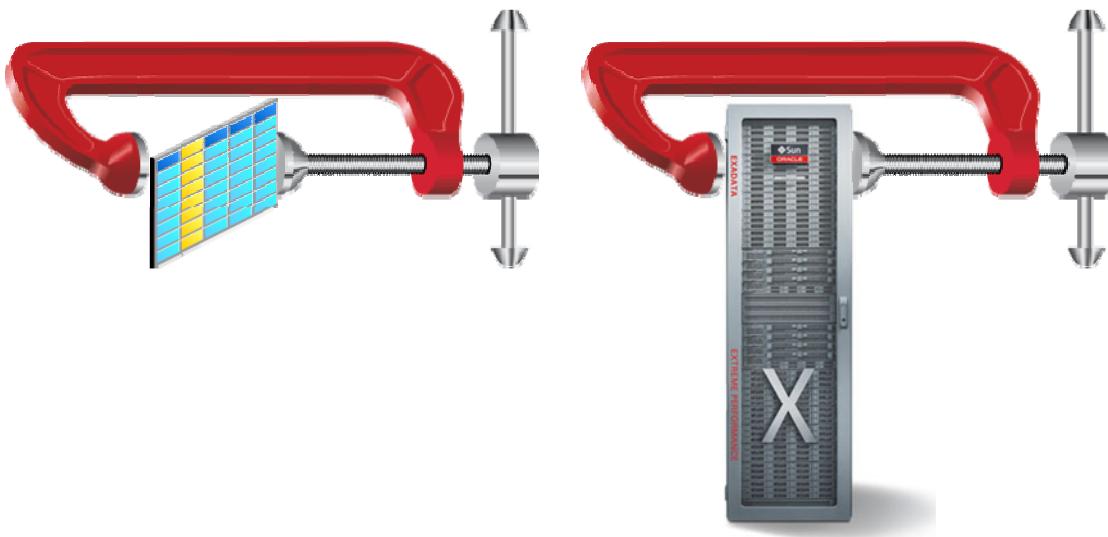


Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The destination Server Collector decompresses the data stream before writing it to the remote file or remote trail. This typically results in compression ratios of at least 4:1 (and sometimes much better). However, compression can require significant CPU resources.

Compression and Exadata

- EHCC
- InsertAppend in Replicat



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate supports Oracle Exadata Database Machine as follows:

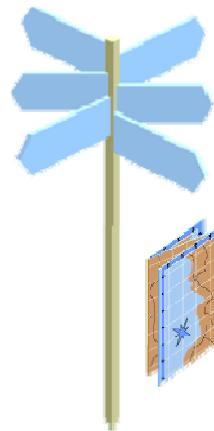
- Oracle GoldenGate can capture from Exadata in either classic capture or integrated capture mode, but to capture from Exadata with **Hybrid Columnar Compression (EHCC)**, Extract must be in integrated capture mode and the source database compatibility must be set to 11.2 or later.
- Oracle GoldenGate can replicate data from any supported database to Exadata. In general, the configuration of Oracle GoldenGate to operate with Exadata is the same as any other Oracle GoldenGate configuration.

Replicating to Exadata with EHCC Enabled

To ensure successful delivery of insert operations to Oracle Exadata with EHCC, use the **InsertAppend** parameter in the **Replicat** parameter file. This causes Replicat to use an **Append** hint for inserts so that they remain compressed. Without this hint, the record is inserted uncompressed.

Roadmap

- BatchSQL
- Compression
- Encryption
 - Overview
 - Message
 - File
 - Password
- Event Actions



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Encryption: Overview

Encrypting and decrypting data can be done by using the following methods:

- Message encryption
 - Messages that are sent over TCP/IP are encrypted.
 - FIPS (AES) and/or Blowfish is used.
 - The data is automatically decrypted by Server Collector before saving the data to the trail.
- Trail or extract file encryption
 - Oracle GoldenGate uses 256-key byte substitution.
 - Only the record data is encrypted in a trail or extract file.
 - The data is decrypted by a downstream data pump or Replicat.
- The Encrypted Database password is generated by using:
 - A default key
 - A user-defined key



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The Federal Information Processing Standards (FIPS) standardization was developed by the United States Federal Government. These standards encompass existing security standards such as Data Encryption Standard (DES) and Advanced Encryption Standard (AES).

AES is a symmetric-key encryption standard that is used by governments and other organizations that require a high degree of data security. AES supports the following ciphers:

- AES128
- AES192
- AES 256

FIPS-supported encryption was added to Oracle GoldenGate with version 11.2.1.0.0.

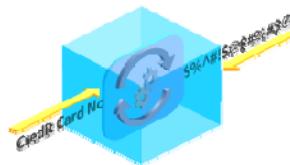
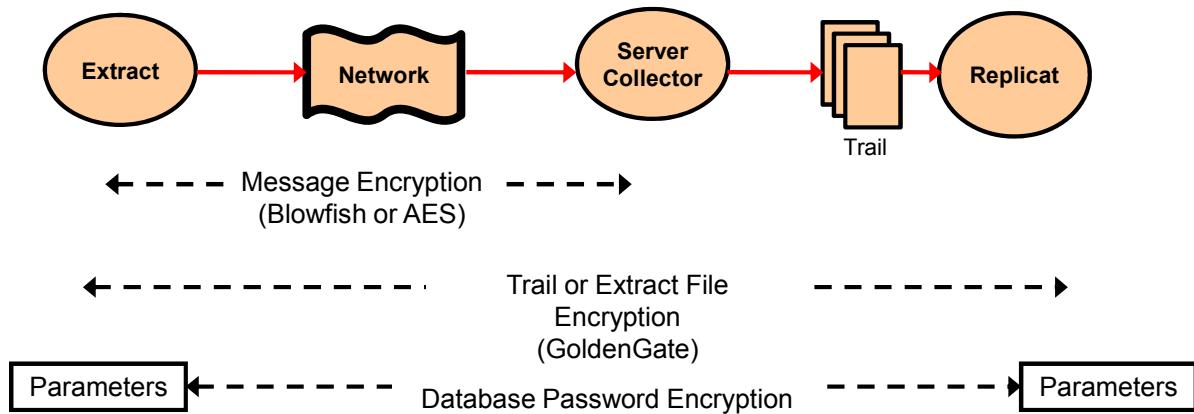
Blowfish is a symmetric 64-bit block cipher from Counterpane Internet Security. Blowfish is not FIPS-compliant and, therefore, not recommended except for backward compatibility.

Note: For details, see <http://www.schneier.com/blowfish.html> and <http://bt.counterpane.com/>.

The master key and wallet method is the preferred method on platforms that support it. It is not supported for the iSeries, z/OS, and NonStop platforms.

ENCKEYS method is valid for all Oracle GoldenGate-supported databases and platforms. Blowfish must be used on the iSeries, z/OS, and NonStop platforms.

Encryption: Overview



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Encryption can be at several levels:

- **Message:** The transmission across the TCP/IP network is encrypted.
- **File:** The contents of the trail or extract files are encrypted end-to-end.
- **Database Password:** The contents of the database can be encrypted.

The keys are either stored in flat file `ENCKEYS` (which works on all platforms) or in Oracle Wallet (which works on only some platforms). Each type of encryption is shown in the following slides.

Message Encryption

- Run the Oracle GoldenGate keygen utility to generate random hex keys:

```
[OS prompt] ./keygen <key_length> <number_of_keys>
```

- Enter your key names and values in an ASCII text file named ENCKEYS (uppercase, no file extension) in the Oracle GoldenGate installation directory:

```
##Key name      Key value
superkey       0x420E61BE7002D63560929CCA17A4E1FB
secretkey      0x027742185BBF232D7C664A5E1A76B040
```

- Copy the ENCKEYS file to the source and target Oracle GoldenGate installation directories.

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The keygen utility and ENCKEYS are needed only if you do not use the Wallet (covered later in this lesson).

<key_length> can be up to 256 bits. Example syntax:

```
[OS prompt]$ ./keygen 128 4
0xA100BF1FDAFE440C0D410901341B7D58
0x28677E0C59BA41177B0F5B23B8C11335
0xAECD3D79D8753E22E9DDAC453C68AA11
0x3534FD6557313B2D57ACFE67BF0E416E
```

If you prefer to use a literal key instead of using keygen, enter the literal key in quotation marks as the key value in an ENCKEYS file:

```
##Key name      Key value
mykey          "DailyKey "
```

Message Encryption

4. In the Extract parameter files, use the RmtHost Encrypt and KeyName parameters:

```
RmtHost <process>, Port <port>,
Encrypt <encrypt_type>, KeyName <keyname>
```

Example:

```
RmtHost west, Port 7809,
Encrypt AES256, KeyName superkey
```

5. Configure a static Server Collector and start it manually with the Encrypt and KeyName parameters:

```
server -p <port> -Encrypt <encrypt_type>
-KeyName <keyname>
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

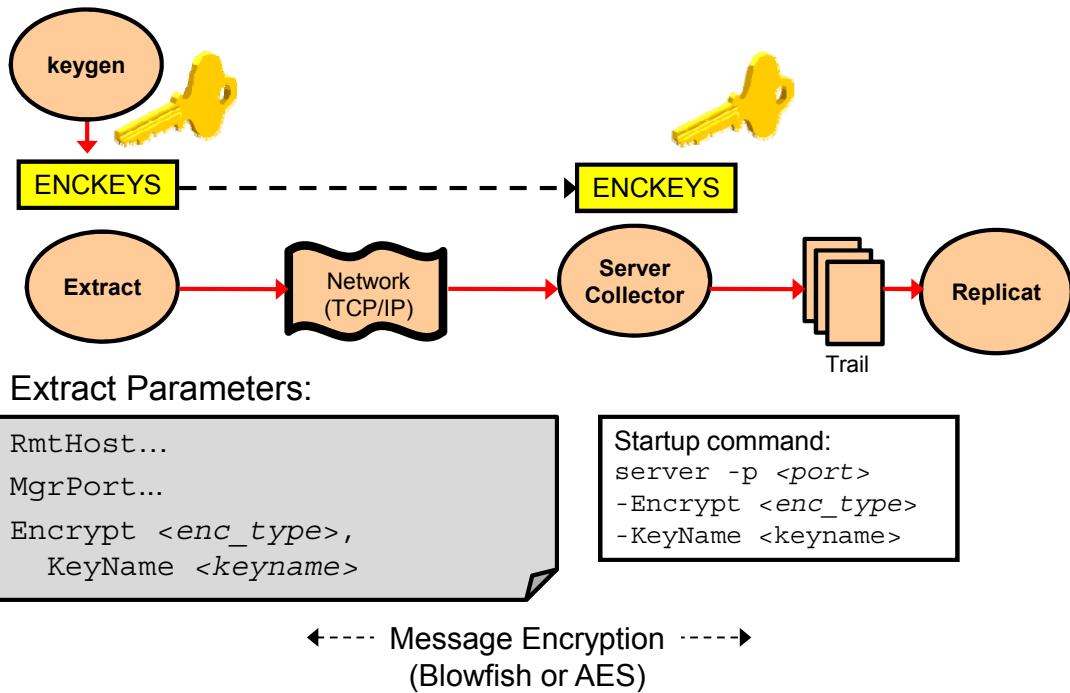
For the remote host RmtHost, use either MgrPort if you use a dynamic collector (the typical way to do things) or Port if you use a static collector, as shown in the slide.

<encrypt_type> can be:

- AES128
- AES192
- AES256
- BLOWFISH (avoid using except for backward compatibility)

The ENCKEYS file contains one or many keys, one of which you named superkey. If you omit the KeyName clause, it defaults to the masterkey in the wallet (covered later in this lesson).

Options: Message Encryption



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

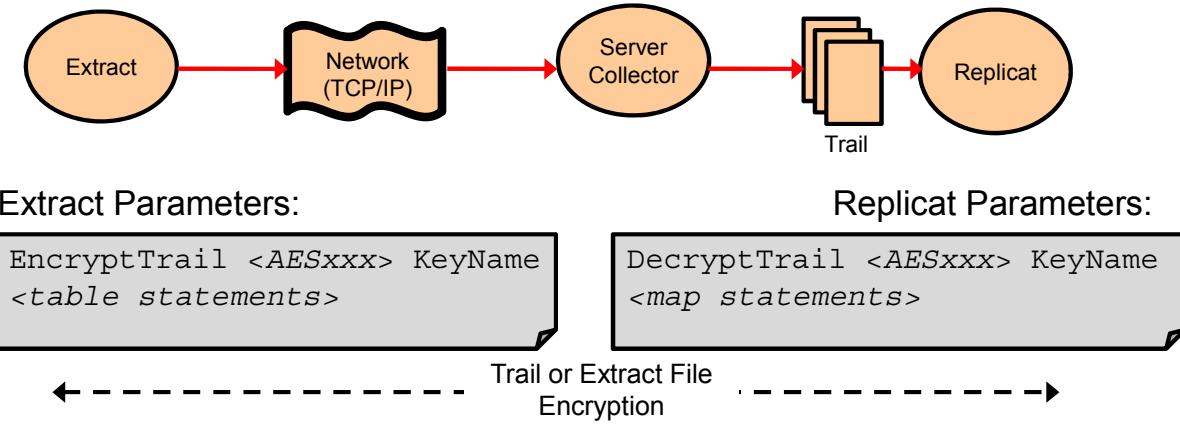
Now the messages go back and forth encrypted.

<enc_type> is any of the following encryption types:

- AES128
- AES192
- AES256
- BLOWFISH (avoid using except for backward compatibility)

If you omit the KeyName clause, it defaults to the masterkey in the wallet (covered later in this lesson).

Trail or Extract File Encryption



Extract Parameters:

```
EncryptTrail <AESxxx> KeyName
<table statements>
```

Replicat Parameters:

```
DecryptTrail <AESxxx> KeyName
<map statements>
```

←----- Trail or Extract File Encryption -----→

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Only the data records are encrypted in the trail.

Set EncryptTrail before the tables that you want to be encrypted, and start NoEncryptTrail before the other tables.

Downstream data pumps can also decrypt the trail for transformation and pass it on either encrypted or decrypted.

<AESxxx> can be any of the following encryption types:

- AES128
- AES192
- AES256

The KeyName <keyname> refers to a key name in the ENCKEYS file.

If you omit the AES specification, Oracle GoldenGate defaults to using the 256-key byte substitution, which is very weak. If you omit the KeyName clause, it defaults to the master key in the wallet (covered later in this lesson).

Trail Encryption with Wallet (new with 12c)

- Oracle Wallet
 - Contains master keys
 - Is used in encryption for trail or TCP/IP
 - Creates keys used with AES
- Common commands:


```
GGSCI> Open Wallet
GGSCI> Add MasterKey {mykey}
GGSCI> Renew MasterKey {mykey}
GGSCI> Info MasterKey { ALL | mykey }
```
- Rare commands:


```
GGSCI> Delete MasterKey ALL
GGSCI> Purge Wallet
```



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

There are at least two wallets in Oracle GoldenGate. The wallet that holds Oracle Database user IDs and passwords for Extract parameter files is stored in the credential store file `dircrd/cwallet.sso`. The wallet file in `dirwlt/cwallet.sso` holds the master keys used with trail or TCP/IP encryption, not with database users. After you create a master key, you only need to add one line `EncryptTrail <enc_type>` in the Pump (or Extract if there is no Pump) and nothing in the Replicat:

```
Extract mypump
EncryptTrail AES256
-- the matching decrypt in the replicat is automatic and is not
specified.
RmtHost targethost, MgrPort 15001, Compress
RmtTrail ./dirdat/pe
Passthru
Table MYSHEMA.*;
```

Similar to `ENCKEYS`, the master keys wallet created on the source host must be either stored on a centrally available disk or copied to all GoldenGate target hosts.

Password Encryption: Method 1

1. Generate an encrypted password with an Oracle GoldenGate default key code:

```
GGSCI> Encrypt Password <password> EncryptKey Default
```

Example:

```
GGSCI> Encrypt Password mypswd EncryptKey Default
No key specified, using default key...
Encrypted password: AACAAAAAAAAAAOARAQIDGEEXAFAQJ
```

2. Paste the encrypted password in the Extract or Replicat Password parameter, as in the following example:

```
:
SourceDB mysource, USERID joe, Password
AACAAAAAAAAAAOARAQIDGEEXAFAQJ, EncryptKey DEFAULT
:
```

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

There are three ways to encrypt passwords:

- **Method 1:** As shown in the slide, use an Oracle-supplied default key and no ENCKEYS file. This method is repeatable and, therefore, susceptible to a reverse dictionary crack, so it is not recommended. Notice that there is also no encryption type specified, so it defaults to Blowfish, which is not a FIPS-compliant method of encryption.
- **Method 2:** A user-defined custom key using the ENCKEYS file (less susceptible) is shown in the next slide.
- **Method 3:** Oracle Wallet (new with OGG 12c)

Password Encryption: Method 2

1. Generate an encrypted password with a cipher strength (*encrypt_type*) and user-defined key:

```
GGSCI> Encrypt Password <password> <encrypt_type>
          EncryptKey <keyname>
```

2. Enter the key name and value in the ENCKEYS file.
3. Paste the encrypted password in the Extract or Replicat Password parameter.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

1. <encrypt_type>: Specify AES128, AES192, AES256, or Blowfish. If you do not specify anything, the default is to use Blowfish, which is the weakest of the four encryption types.

<keyname>: Comes from the ENCKEYS file

Example of generating an encrypted password with a user-defined key:

```
GGSCI> Encrypt Password MyPass AES256 EncryptKey drkey
Encrypted password: AACAAAAAAAAAAIAJFGBNEYGTGSBSHVB
```

2. Example of key name and value from ENCKEYS:

```
##Key name  Key value
drkey      0x11DF0E2C2BC20EB335CB98F05471A737
```

3. Example of pasted encrypted password:

```
SourceDB mysource, UserID joe, Password
AACAAAAAAAAAAIAJFGBNEYGTGSBSHVB, EncryptKey drkey
```

Password Encryption: Method 3

This method is only for Oracle Database passwords. It is not for TCP/IP encryption or for trail encryption.

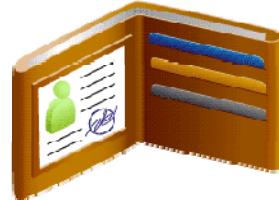
```
GGSCI> Create Wallet  
GGSCI> Add CredentialStore  
GGSCI> Alter CredentialStore Add User ogguser@mysid  
          Password mypswd Alias ggalias  
GGSCI> Info CredentialStore
```

In Extract or Replicat .prm files, replace all occurrences of:

UserID ogguser@mysid, Password mypswd

With:

UserIDAlias ggalias



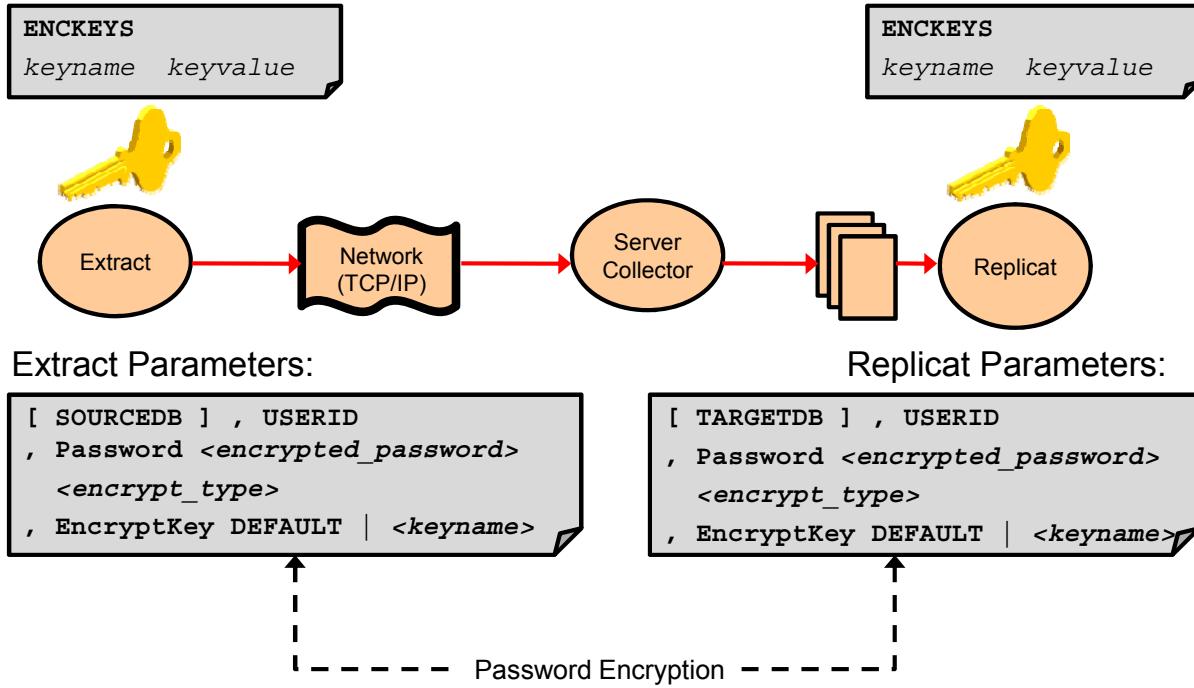
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

If available, this is the preferred method of password encryption.

Note that this is a different wallet than the one used for trails and TCP/IP encryption.

Summary of Password Encryption



ORACLE

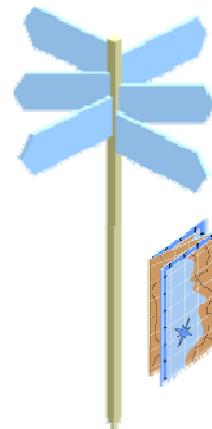
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

There are two ways to create the password; either way involves copying and pasting the encrypted value into the parameter files on both the source and the target.

Note that there is no comma between the password and its type.

Roadmap

- BatchSQL
- Compression
- Encryption
- Event Actions
 - Rules
 - Heartbeat
 - Automated Actions



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Event Marker System

- Oracle GoldenGate provides an event marker system that enables Oracle GoldenGate processes to take a defined action based on an event record in the transaction log or in the trail (depending on the data source of the process).
- The event record is one of the following:
 - A record in a data table that satisfies a specific filter condition for which you want an action to occur
 - A record that you write to a dedicated event table when you want an action to occur

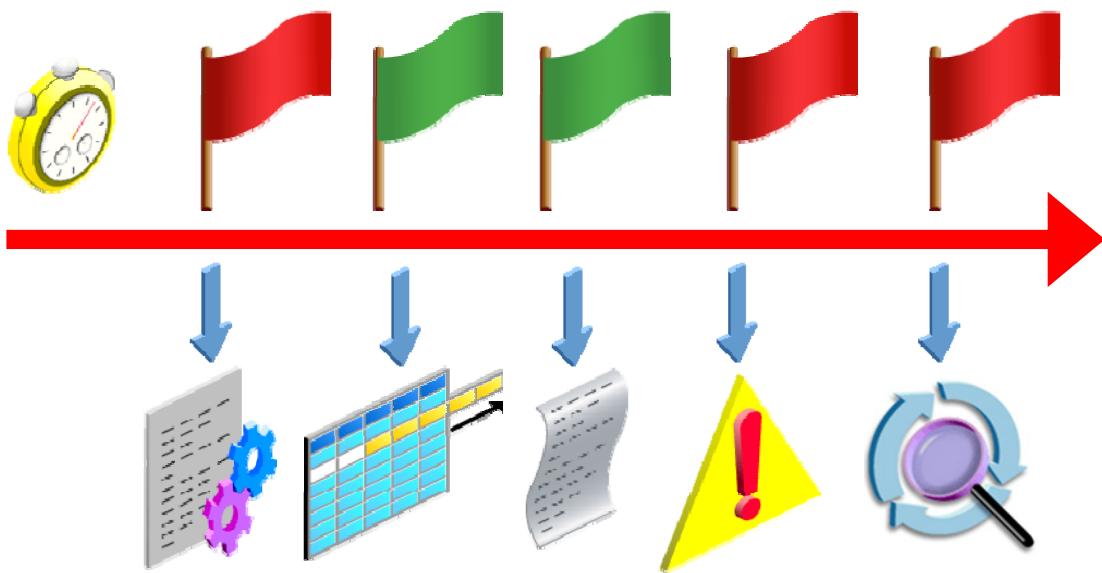


ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can use the event marker system to customize Oracle GoldenGate processing based on database events.

Uses for Event Actions



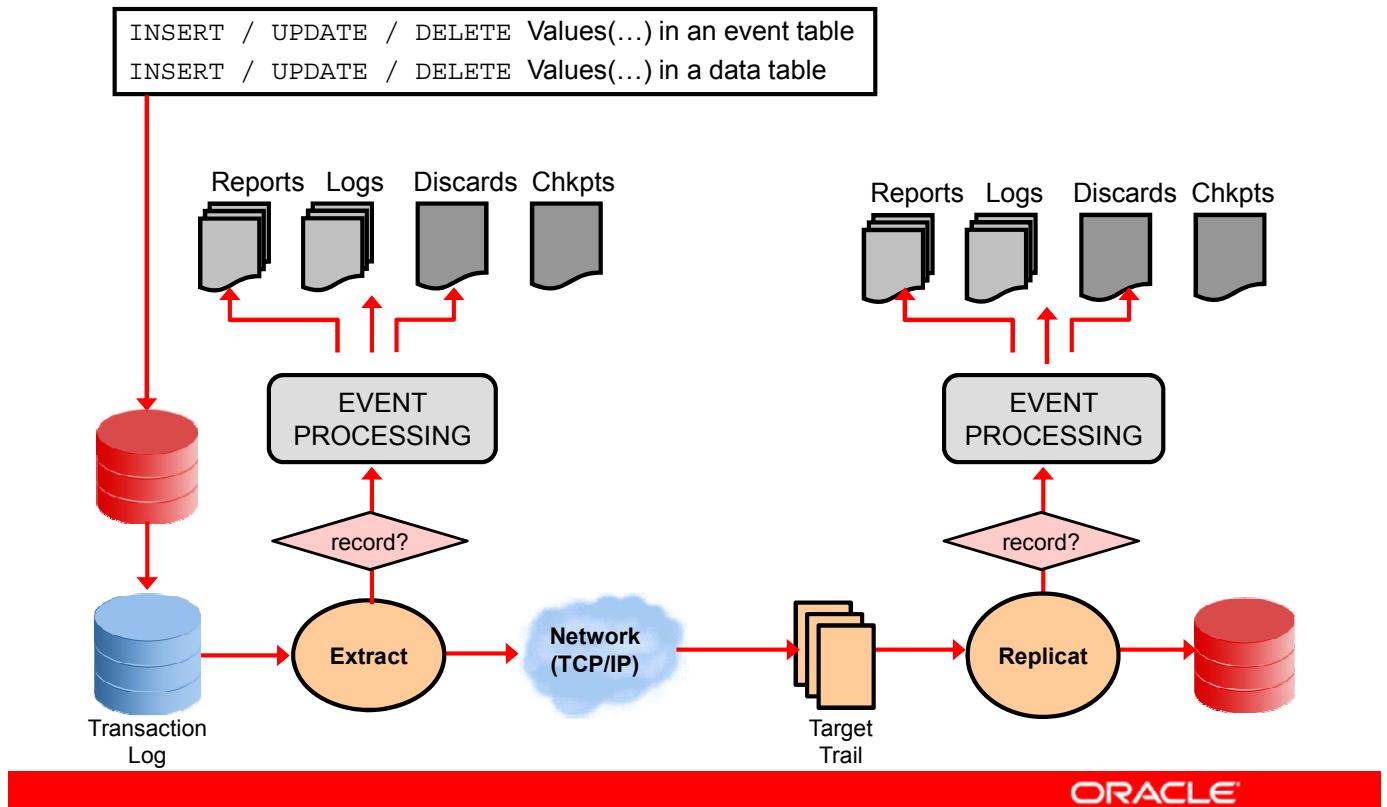
ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Examples of actions that you might take after detecting an event record:

- Start or stop a GoldenGate process (there are several flavors of “stop”: graceful, abort, and so on).
- Ignore or discard the current record.
- Log an informational or warning message to the report file, Oracle GoldenGate error log, and system event log.
- Generate a report file.
- Roll over the trail file.
- Run a shell command in the OS (for example, to switch an application, start batch processes, or start end-of-day reporting).
- Activate tracing.
- Write a checkpoint before or after writing the record to the trail.

Event Actions Flowchart



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The event can trigger in an Extract or a Replicat. One transaction can contain many rows, and several of those rows could each trigger multiple events.

Events can trigger:

- Reports
- Writing to a log
- Discard of records
- Writing a checkpoint

New with version 11.2 is the ability to trigger from a DDL event as well as a DML event.

Similar to database triggers, the amount of overhead from excessive events should be monitored.

EventActions Order

Before	Process the Record...	After
Trace Log Checkpoint Before Ignore Discard Shell Rollover	...unless you specified either Ignore or Discard	Report Suspend Abort Checkpoint After ForceStop Stop



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Many (but not all) EventActions options can be combined. You probably will need to combine two or more actions to achieve your goals.

The entire EventActions statement is parsed first, and only then are the specified options executed according to which one takes precedence over the other. In the table in the slide, the actions in the Before column occur before the record is written to the trail or applied to the target (depending on the process, Extract or Replicat). Actions that are listed in the After column are executed after the record is processed. If you do not want to process the record at all but only perform the actions, specify Ignore or Discard in addition to the other actions.

It is possible that a single transaction contains two or more records that trigger an event action. In such a case, there could be multiple executions of certain EventActions. For example, if two qualifying records trigger two successive Rollover actions, it causes Extract to roll over the trail twice, leaving one of the two files essentially empty.

Trace has several modifiers: Transaction, DDL, and Purge or Append.

The Checkpoint option has three modifiers: Before and After (listed in the slide) and Both. Checkpoint cannot be combined with Abort.

Implementing Event Actions: Examples

- Use a separate event table to manage events.
Whenever a record is written to the event table, the trail file is rolled over:

```
Table MYSOURCE.EVENT_TABLE, EventActions (Rollover);
```

- Use data values to trigger events.
Any record where account_no = 100 is discarded and a log message is written:

```
Map MYSOURCE.ACCOUNT, Target MYTARGET.ACCOUNT,
Filter (account_no = 100), EventActions (Discard, Log);
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate supports actions based on the event record condition.

There are two ways to configure event actions. Perform either of the following steps:

- Add the EventActions option to a Table or Map statement.
- In the same Table or Map statement where you specified the event record, include the EventActions parameter.

New with version 11.2 is the ability for Oracle GoldenGate to pass values to an external shell script as a result of an action. The syntax is:

```
EventActions Shell
(
    "<command>",
    VAR <variable> = {<column_name> | <expression>}
    [, ...] [, ...]
)
```

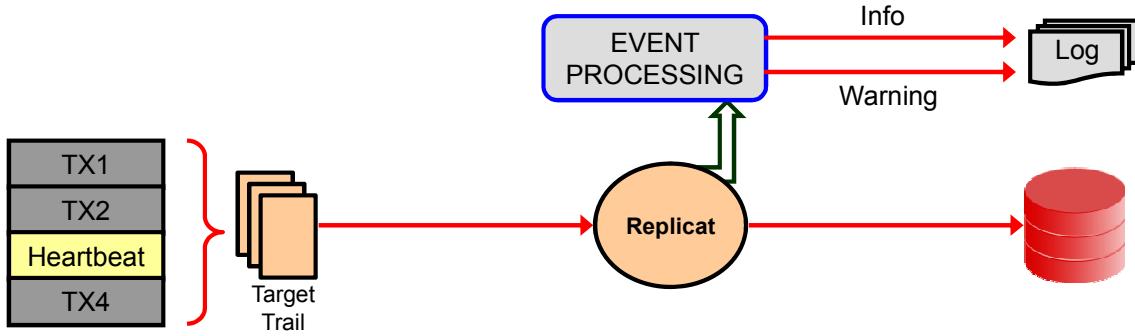
Event Actions: Heartbeat Example

```

Map SOURCE.HEARTBEAT, Target TARGET.HEARTBEAT,
  Filter (@DateDiff ("SS", hb_timestamp, @DateNow() > 60
AND @DateDiff ("SS", HBTIMESTAMP, @Datenow() < 120),
  EventActions (LOG);

Map SOURCE.HEARTBEAT, Target TARGET.HEARTBEAT,
  Filter (@DateDiff ("SS", hb_timestamp, @DateNow() > 120),
  EventActions (Log Warning);

```



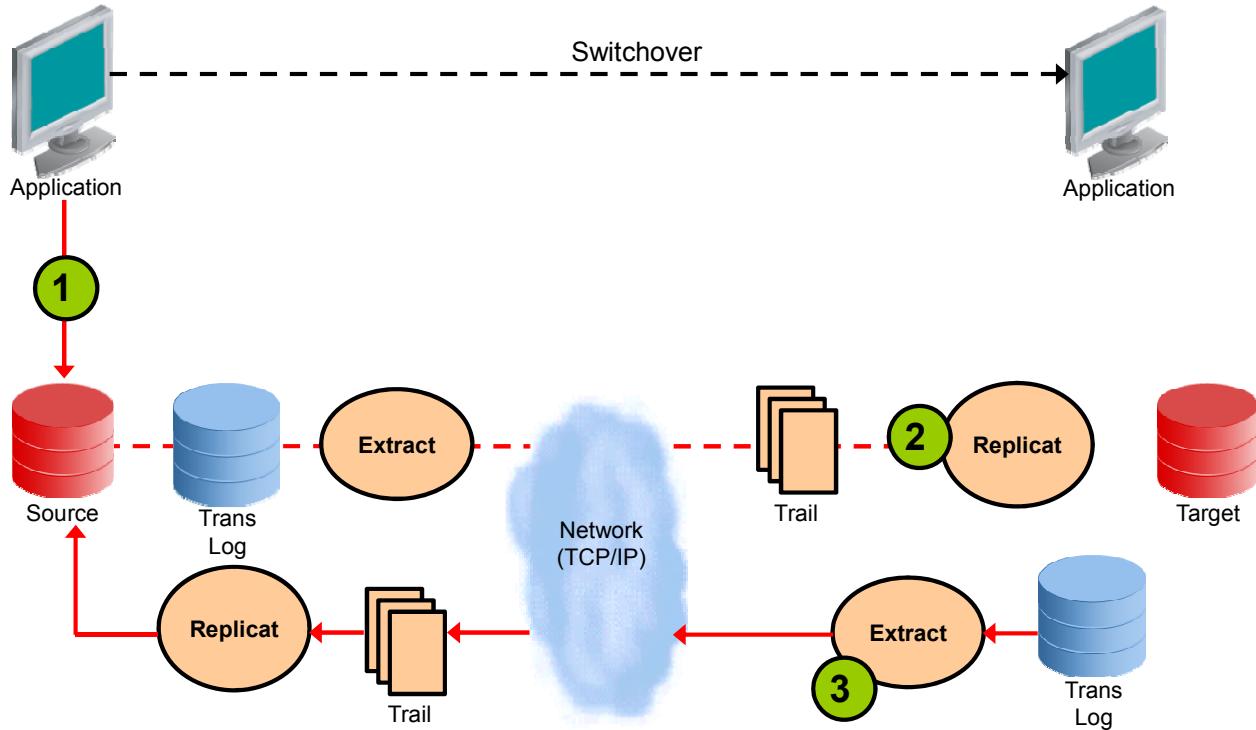
ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In the example in the slide, a table named HEARTBEAT is periodically updated with the current time in the source database. Assume for this example that the source database writes a heartbeat every 20 seconds to the table. The event is to log a heartbeat on the target after every 60 seconds. If it misses two or three heartbeats, that is okay. If it misses more than six heartbeats (in this example, two minutes), log a warning.

The "SS" parameter in @DateDiff gives you the date difference in seconds, as opposed to "mi" minutes or "hh" hours, and so on.

Event Actions: Automated Switchover Example

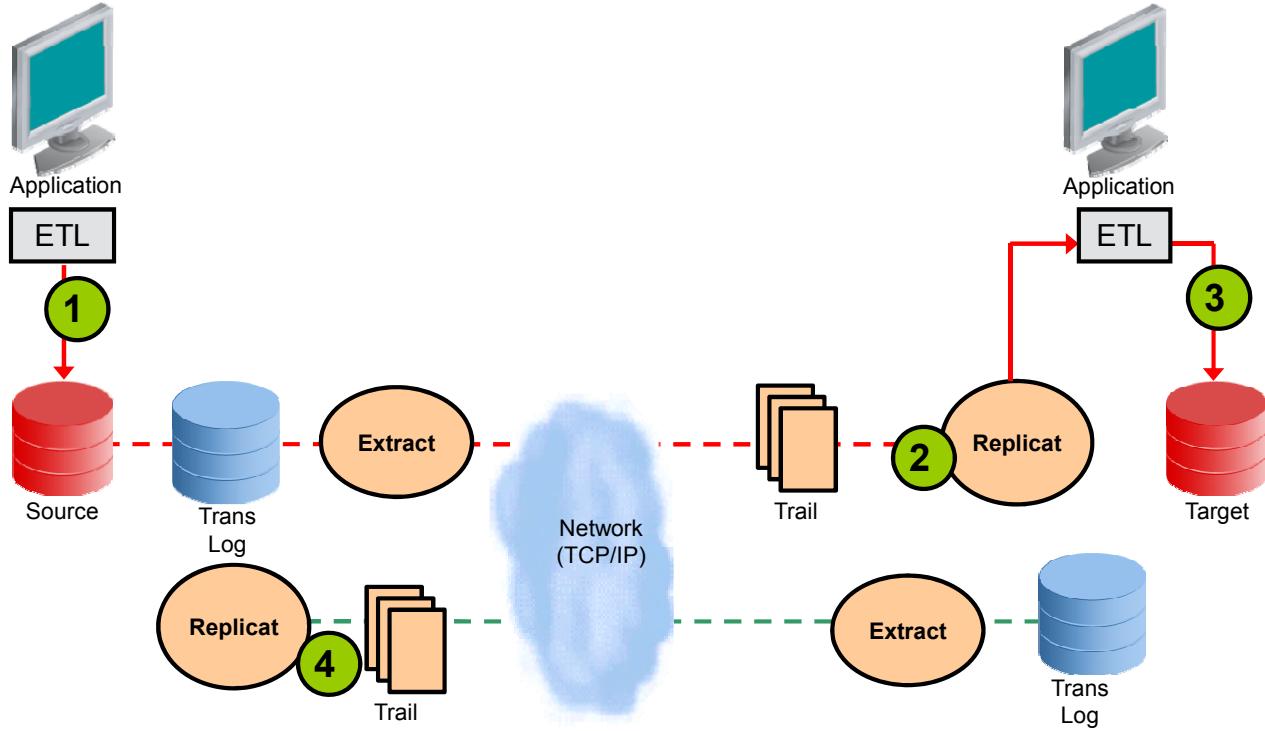


ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

1. The user writes an event record at the planned outage point. This is read by Extract through the transaction log.
2. When Replicat reads the event record, it triggers an event action (runs a custom script to switch the application to the target database).
3. The Extract on the target, which is already configured and running, starts capturing transactions.

Event Actions: Automated Synchronization Example



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

1. When a batch load is starting, the ETL process writes an event record. Extract reads the record and performs a checkpoint before and after the record.
2. When Replicat reads the event record, it requests the second ETL process to start at the right point.
3. The ETL performs checkpoints before and after the record.
4. When the second ETL process is completed, it generates an event record that is read by Extract on the target. When Replicat on the source receives the event record, it triggers a custom script to start the application based on the status of the batch process on the source.

Quiz

Key names and values can be stored in:

- a. GLOBALS
- b. ENCKEYS
- c. SourceDefs
- d. DefsFile



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Quiz

Only the EncryptTrail extract parameter should be set to encrypt and decrypt the records in the trail file.

- a. True
- b. False



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Summary

In this lesson, you should have learned how to use the following:

- BatchSQL to speed up delivery
- Data compression across the network
- Message, trail, and password encryption
- Event records to automatically trigger actions



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice 13 Overview: Configuration Options

This practice covers the following topics:

- Setting up the database and the source definitions file
- Performing Oracle GoldenGate encryption
- Generating transactions and validating results



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Bidirectional Replication

14

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

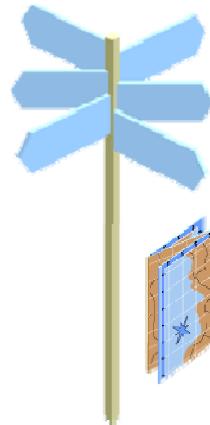
- Describe bidirectional configuration
- Detect and avoid loops
- Handle bidirectional issues such as loop detection
- Detect and resolve conflicts
- Handle identity types



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Roadmap

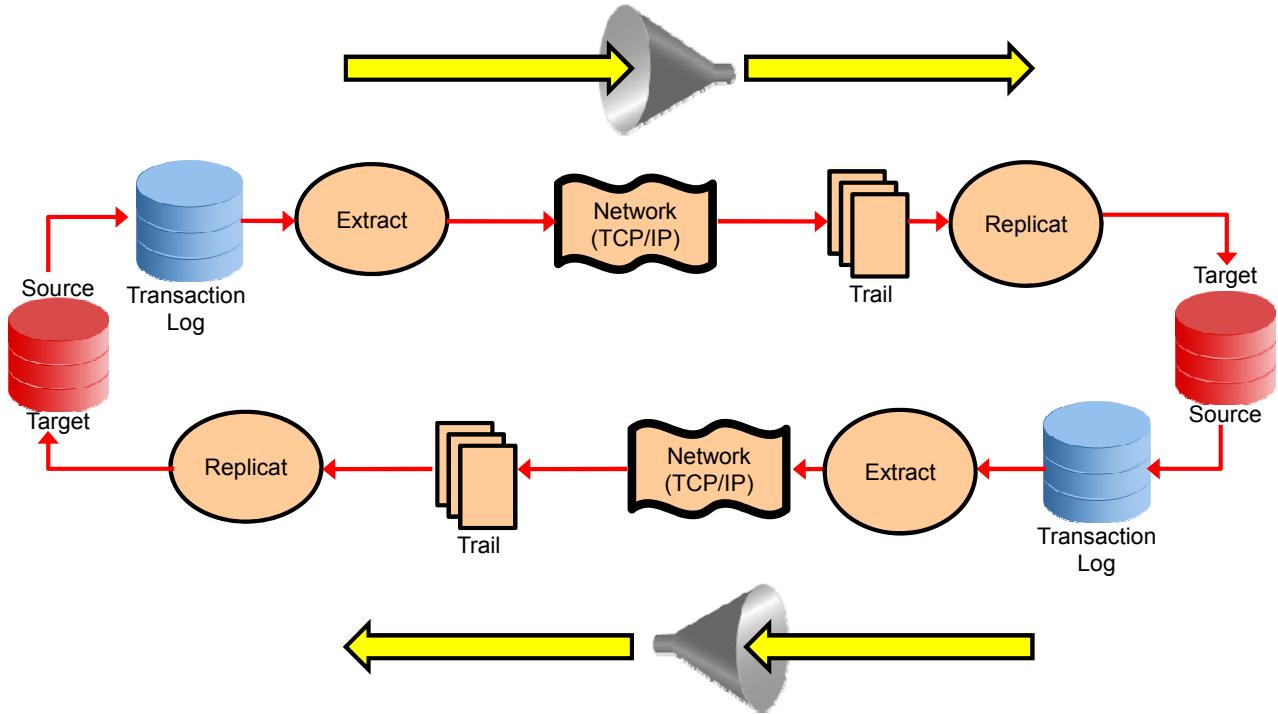
- Bidirectional Topics
 - Capabilities
 - Issues
- Looping
- Conflict Detection and Resolution
- Miscellaneous



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Bidirectional Flowchart



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In the diagram in the slide, changes from the database on the left of the illustration are extracted and sent over the network to be replicated in the database on the right.

The diagram at the bottom of the slide indicates the reverse: Changes from the database at the right are extracted and sent to the database at the left.

In a bidirectional environment, there must be some kind of a filter (represented by the funnel) to ignore certain transactions to prevent a loop.

Capabilities of a Bidirectional Configuration

- Bidirectional configurations are available for both homogeneous and heterogeneous configurations.
- Bidirectional configurations support distributed processing.
- Both sides are active.
- Oracle GoldenGate's low latency reduces the risk of conflicts.
- Oracle GoldenGate provides loop detection.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate supports bidirectional replication between two databases, whether the configuration is for homogeneous or heterogeneous synchronization. In a bidirectional configuration, both sides of the database may be live and processing application transactions at the same time. It also may be that the target application is standing by, waiting to be used if there is a failover. For this, operations are captured and queued to be synchronized back to the primary database after it becomes available.

Configuring Oracle GoldenGate for bidirectional replication may be as straightforward as configuring a mirror set of Extract and Replicat groups moving in the opposite direction. It is important, however, to thoroughly discuss special considerations for your environment to guarantee data accuracy. The following slides cover some of the bidirectional concerns as well as known issues relating to file and data types.

Bidirectional Configuration Considerations

- Loop detection
 - Detect whether Oracle GoldenGate or the application performed the operation.
- Conflict avoidance, detection, and resolution
 - Detect whether an update occurred on both the source and the target before the changes were applied by Oracle GoldenGate.
 - Determine the business rules for handling collisions.
- Sequence numbers and identity data types
- Truncate table operations



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Loops

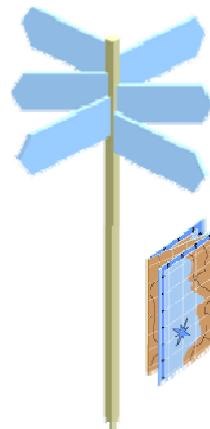
Because there are both Extract and Replicat processes operating on the same tables in bidirectional synchronization, Replicat's operations must be prevented from being sent back to the source table by Extract. If they are re-extracted, they will be re-replicated, beginning an endless loop. Loop detection is sometimes called ping-pong detection.

Conflicts

Because Oracle GoldenGate is an asynchronous solution, conflict management is required to ensure data accuracy in the event that the same row is changed in two or more databases at (or about) the same time. For example, User A on Database A updates a row, and then User B on Database B updates that same row. If User B's transaction occurs before User A's transaction is synchronized to Database B, there will be a conflict on the replicated transaction.

Roadmap

- Bidirectional Topics
- Looping
 - Preventing
 - Detecting
- Conflict Detection and Resolution
- Miscellaneous



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Preventing Data Looping

- The following sample sequence demonstrates the problem that occurs with UPDATE without loop detection:
 1. A row is updated on System A.
 2. The update operation is captured and sent to System B.
 3. The row is updated on System B.
 4. The update operation is captured and sent to System A.
 5. The row is updated on System A.
- Without loop detection, this loop continues endlessly.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

To avoid looping, Replicat's operations must be prevented from being sent back to the source table by Extract. This is accomplished by configuring Extract to ignore transactions issued by the Replicat user. This slide illustrates why data looping must be prevented. It is bad enough when the update says `SET mycol=100` over and over again, but it is worse when the update says `SET mycol=mycol+10` over and over again—which will approach infinity!

Preventing Data Looping

Another example involves the following `INSERT` sequence:

1. A row is inserted on System A.
2. The insert is captured and sent to System B.
3. The row is inserted on System B.
4. The insert is captured and sent to System A.
5. The insert is attempted on System A, but the operation fails with a conflict on the primary key causing synchronization services to HALT.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Because of the constraint that the primary key must be unique, the insert fails in this example so that it does not trigger looping. However, the failed insert causes Replicat to ABEND, which stops all replication services. Thus, this is another example of the need to recognize and not extract Replicat transactions.

Scenario

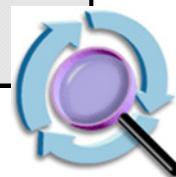
1. User `west` on database AMER inserts a row into table `west.branch`, primary key 100.
2. User `gguamer` on database AMER Extract `ewest` reads the redo log and generates the trail to replicate it to database EURO. The row is marked as originating from user `west`.
3. User `ggueuro` on database EURO Replicat `reast` reads the trail file and inserts the new row to table `east.branch`. The insert succeeds as this is a new row. The row is marked as originating from user `gueuro` even though it goes into schema `east`.
4. User `gueuro` on database EURO Extract `eeast` reads the redo log with the insert and generates the trail to replicate it to database AMER. The row is marked as originating from user `gueuro`.
5. User `gguamer` on database AMER Replicat `rwest` reads the trail file and attempts to insert the same row again into `west.branch` with primary key 100. The insert fails with Duplicate Key violation. An insert or delete would fail; an update would potentially continue in an infinite loop. Either scenario is bad.

Loop Detection Techniques

The loop detection technique depends on the source database:

- Oracle Database
 - Use `ExcludeUser` or `ExcludeUserID` (Oracle 10g and later) or `ExcludeTag` (12c or later).
 - Execute Replicat with a unique database `username` or Oracle `userID`.
 - Suppress capture of Replicat transactions with one of the Extract parameters:

```
TranLogOptions ExcludeUser <username>
TranLogOptions ExcludeUserID <OracleUserID>
TranLogOptions ExcludeTag <nn> [, <nn> ]
```



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

ExcludeUser and ExcludeUserID: This eliminates the overhead of creating and writing to the trace table (next page).

New with 12c is `ExcludeTag` and its matching `SetTag`.

Syntax sample in Extract: `SetTag 00`

Syntax sample in Replicat: `TranLogOptions ExcludeTag 00, 55, FF`

Scenario repeated from the previous page (same Steps 1–3, new step 4):

4. Extract parameter file has `TranLogOptions ExcludeUser ggueuro`. User `ggueuro` on database EURO Extract reads the redo logs with the insert, sees that the row is marked as originating from user `ggueuro`, and excludes it. Loop solved!

You do this `ExcludeUser` in the practice for this lesson. Note that the loop was fixed on the Extract side, rather than the Replicat side, which means that the data does not have to make any “unnecessary” trips.

Alternatively, if the users are unpredictable or complicated, you could have used `SetTag` and `ExcludeTag` parameters instead to do the same thing. The difference is that you have to explicitly `SetTag` on the Extract side, whereas the user is automatically known to the Extract. `SetTag` can be any hex number. `ExcludeTag` can be a comma-separated list of hex numbers.

Loop Detection Techniques

- Use an Oracle trace table.
 - Add a trace table to detect Oracle GoldenGate operations with the GGSCI command:

```
GGSCI> Add TraceTable <OWNER>. <TABLE_NAME>
```

- If you are not using the default table name, add a parameter in both Extract and Replicat:

```
TraceTable <OWNER>. <TABLE_NAME>
```



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

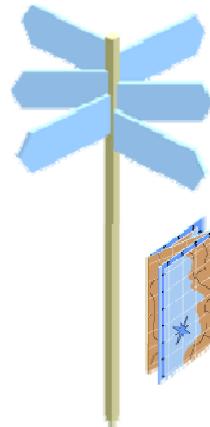
Oracle trace table: Loop detection can be accomplished by creating a table in the Oracle environment known as the *trace table*. The Replicat process updates this table when a transaction is committed, providing a mechanism that Extract can use to detect a Replicat operation. When Replicat starts up, it looks for a `TraceTable` parameter, and if one exists, updates the trace table name as every transaction is committed. If no `TraceTable` parameter is present, Replicat looks for a default table, `GGS_TRACE`, owned by the logged-in `UserID`. If it exists, it will automatically update the table. If `GGS_TRACE` does not exist, nothing is updated.

To create the `GGS_TRACE` table, use the GGSCI `Add TraceTable` command. To create a table with a different table name, use `Add TraceTable [<owner>] . <table_name>`. If `<owner>` is omitted, it assumes the logged-in `USERID`. GGSCI also provides `INFO` and `DELETE` commands for the `TraceTable` entity. To use these commands, you must first log in to the database by using `DBLogin`.

The Extract process behaves in a similar manner as Replicat. As it is started, it looks for the `TraceTable` parameter and uses the table name as the table to use to detect if Replicat performed an operation. If no parameter is present, it looks to see whether `GGS_TRACE` is part of the transaction.

Roadmap

- Bidirectional Topics
- Looping
- Conflict Detection and Resolution
 - CompareCols
 - Filter
 - Built-in Methods
 - Custom Methods
- Miscellaneous



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Conflict Avoidance and Detection and Resolution

- Conflicts can be minimized:
 - By low latency
 - At the application level by assuring that records are always updated on only one system
- Oracle GoldenGate can capture both the before and after images of updates so that you can compare these before applying updates.
 - Several built-in CDR commands are available for Insert, Update, and Delete.
 - You can write logic to do the compares in a filter or user exit.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Low latency reduces the risk of encountering a conflict where the same record is updated on both systems. This is the best overall method encouraged in bidirectional configurations. For every conflict you avoid, there is one less conflict that you need to resolve.

Conflict detection/resolution (CDR) can also be addressed at the application or system level by assuring that records are always updated on only one system. For example, all updates to card holders (or account numbers) 0 through 1000000 are applied on System A, whereas updates to 1000001 and higher are applied on System B. Oracle GoldenGate for UNIX and Windows (only) provides the capability to capture the before and after values of columns so that comparisons may be made on the target database before applying the values. Additional SQL procedures can be written based on your application requirements.

Conflict detection/resolution requires Oracle GoldenGate version 11.2.1 or later.

Conflict Detection by CompareCols

- Is used in the Map statement on the Replicat side
- Determines which rows pass on to the WHERE clause

The diagram illustrates a conflict detection scenario. It shows three tables: Source BEFORE, Source UPDATE, and Target BEFORE. The Source BEFORE table has two rows: (100, 'widget', 'vendorY', 50.00) and (110, 'gizmo', 'vendorZ', 60.00). The Source UPDATE table shows the row for item 100 updated to (100, 'widget', 'vendorY', 70.00). The Target BEFORE table has two rows: (100, 'widget', 'vendorX', 50.00) and (110, 'gazmoo', 'vendorY', 60.00). A red curved arrow points from the Col_C value of 50.00 in the Source BEFORE row to the Col_C value of 50.00 in the Target BEFORE row, indicating a comparison between the source and target values.

	Col_Key	Col_A	Col_B	Col_C
Source BEFORE	100	widget	vendorY	50.00
	110	gizmo	vendorZ	60.00

	Col_Key	Col_A	Col_B	Col_C
Source UPDATE	100	widget	vendorY	70.00

	Col_Key	Col_A	Col_B	Col_C
Target BEFORE	100	widget	vendorX	50.00
	110	gazmoo	vendorY	60.00

CompareCols (ON UPDATE KEYINCLUDING col_a)

ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Suppose that you have a source row that contains values:

(100, 'widget', 'vendorY', 50.00)

You want to update Col_C to 70 and then replicate that row. However, for whatever reason, the target is out of sync with the source. Notice that item 100 has a different vendor on source and target; therefore, a conflict exists. The two BEFORE images do not match. Because of business reasons, the application does not care if Col_B does not compare, it only cares if Col_A does not compare. In this case, the update proceeds.

Suppose in another case, the names are different, the target BEFORE for item 110 has a name of "gazmoo" versus "gizmo," and you are trying to update Col_C to 70. Because Col_A is being compared in addition to the key, that row would be in conflict and you have a choice of what to do:

- Retain the before value.
- Overwrite with the new value.
- Check another column, such as a time stamp, to decide which value to use.
- Reject the row into the discard file.

GetUpdateBefores, GetBeforeCols, CompareCols, and ResolveConflict

```
-- WEST
Extract EWCONF
:
GetUpdateBefores
:
Table WEST.PRODUCTS GetBeforeCols(On Update All);
```

```
-- WEST data pump
Extract PWCONF
:
Table WEST.*;
```

```
-- EAST
Replicat RECONF
:
Map WEST.PRODUCTS, Target EAST.PRODUCTS,
CompareCols(On Update All), ResolveConflict (UpdateRowExists, ...);
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

All four keywords need to be used in the following places for CDR to work:

- **GetUpdateBefores**: To do the *before* and *after* testing, the trail needs to ship both the images in the record. By default, it does not do that. This enables the before image.
- **GetBeforeCols**: This specifies the columns for which the before image is captured and written to the trail on an update or a delete operation. For updates, the before image of the specified columns is included in the trail whether or not any given column is modified. To use this parameter, supplemental logging must be enabled for any database that does not log before image values by default.
- **CompareCols**: This specifies the columns that are included in Replicat's `Where` clause with their before image values. Usually application-specific critical columns are specified in this list, to check if a conflict exists. A conflict is a mismatch between the *before* image of a record in the trail and the *current* data in the target table.
- **ResolveConflict**: This specifies how Replicat handles conflicts on operations made to the tables in the `MAP` statement in a bidirectional configuration. Use `ResolveConflict` multiple times in a `MAP` statement to specify different resolutions for different operations and conflict types. Specific modes of resolution are covered in the slide titled "ResolveConflict Built-in Methods."

The full parameter files are the following:

```
-- WEST
Extract EWCONF
ExtTrail ./dirdat/wf
UserIDAlias ggualias
GetUpdateBefores
TranLogOptions ExcludeUser gguser
Table WEST.PRODUCTS GetBeforeCols(On Update All);

-- WEST
Extract PWCONF
RmtHost easthost, MgrPort 15001, Compress
RmtTrail ./dirdat/pf
Passthru
Table WEST.*;
-- cannot use GetBeforeCols on the data pump above with Passthru...

-- EAST
Replicat RECONF
AssumeTargetDefs
DiscardFile ./dirrpt/reconf.dsc, Purge
UserIDAlias ggualias
AllowDupTargetMap
IgnoreUpdates
Map WEST.* , Target EAST.*;
GetUpdates
IgnoreInserts
IgnoreDeletes
Map WEST.PRODUCTS, Target EAST.PRODUCTS,
CompareCols(On Update All),
ResolveConflict (UpdateRowExists,
(delta_resolution_method, UseDelta, Cols (qty_in_stock)),
(max_resolution_method, UseMax (modified_ts), Cols (vendor,
modified_ts)),
(Default, Overwrite));
```

Conflict Detection by Filter

```
REPERROR 9999, EXCEPTION
Map SRCTAB, Target TARGTAB,
    SQLEXEC (ID CHECK, ON UPDATE, BEFOREFILTER,
              QUERY "SELECT COUNTER FROM TARGTAB"
              "WHERE PKCOL = :P1",
              PARAMS (P1 = PKCOL)),
    Filter (ON UPDATE, BEFORE.COUNTER <> CHECK.COUNTER,
            RAISEERROR 9999);

InsertAllRecords
Map SRCTAB, Target TARGET_EXCEPT, EXCEPTIONSONLY,
    ColMap (USEDEFAULTS, ERRTYPE="Conflict Detected");
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The code in the slide performs the following steps when an update is encountered:

- Before the update filter is executed, perform a query to retrieve the present value of the COUNTER column.
- If the first filter is passed successfully, ensure that the value of COUNTER before the update occurred matches the value in the target before performing the update.
- If the update filter fails, raise error 9999.

Note: The REPERROR clause for error 9999 ensures that the exceptions map to TARGET_EXCEPT is executed.

Conflict Resolution

Addressing conflict resolution depends on your business rules.

Example:

- Apply the net difference instead of the after value.
- Map the data to an exception table for manual resolution.
- Accept or ignore changes based on date and time.
- Accept or ignore changes based on comparing other columns.

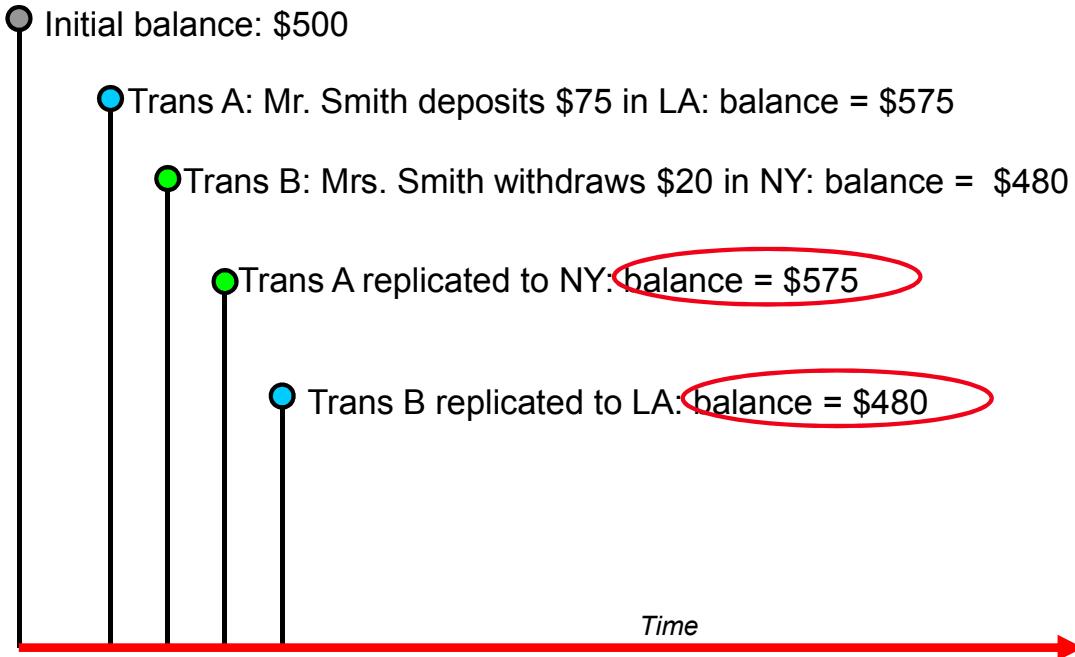


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The next slide shows an example of one form of conflict resolution.

Conflict Resolution: Example



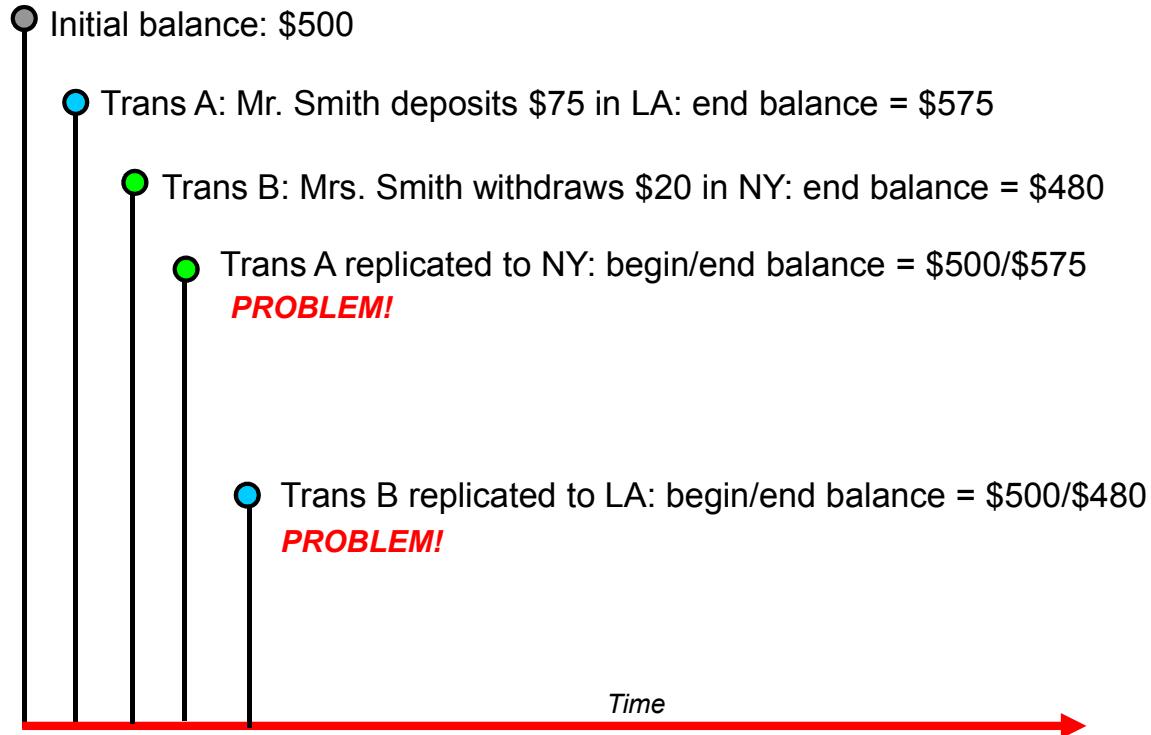
At end of day, correct balance should be $\$500 + \$75 - \$20 = \$555!$

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

When the two transactions occur at different times, the deposit transaction is replicated to New York before the withdrawal transaction is extracted. But when the two transactions occur at around the same time, they are applied independently, resulting in an incorrect balance on both systems.

Conflict Resolution by Applying Net Differences



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

A conflict is detected between the trans A beginning balance (500) and the current NY balance (480), so apply the net difference:

$$\text{Trans A end balance (575)} - \text{Trans A begin balance (500)} = 75$$

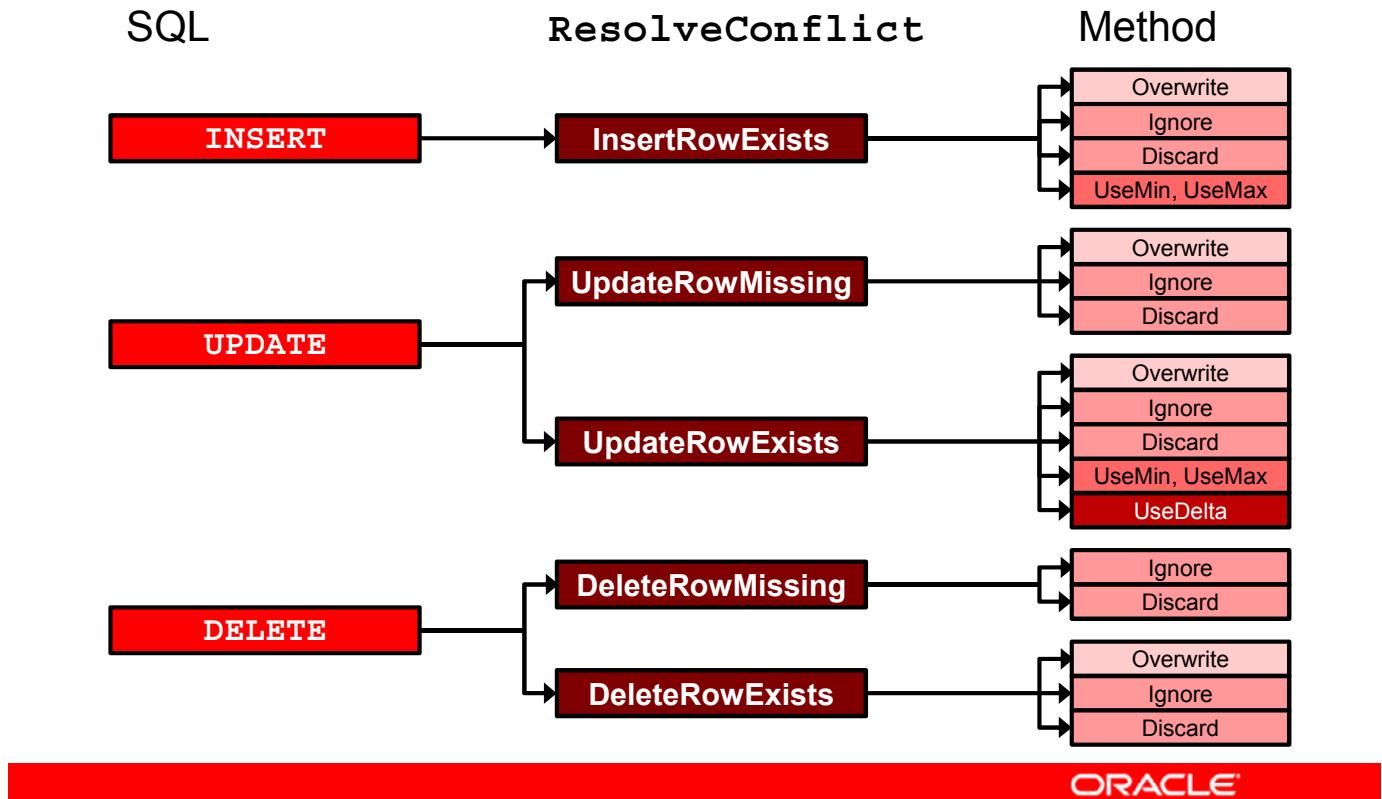
This results in a NY end balance of \$555.

A conflict is detected between the trans B beginning balance (500) and the current LA balance (575), so apply the net difference:

$$\text{Trans B end balance (480)} - \text{Trans B begin balance (500)} = -20$$

This results in an LA end balance of \$555.

ResolveConflict Built-in Methods



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Example

This is from Practice 14. A similar test using custom coding is in the next slide.

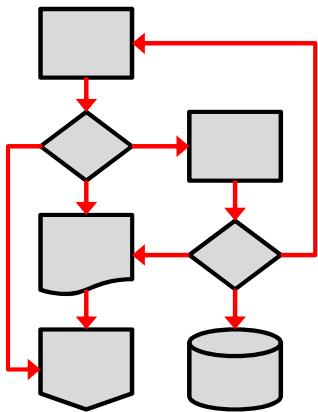
```

Replicat reconf
AssumeTargetDefs
DiscardFile ./dirrpt/reconf.dsc, Purge
UserIDAlias ggualias
AllowDupTargetMap
IgnoreUpdates
Map WEST.* , Target EAST.*;
GetUpdates
IgnoreInserts
IgnoreDeletes
Map WEST.PRODUCTS, Target EAST.PRODUCTS,
  CompareCols(On Update All),
  ResolveConflict (UpdateRowExists,
    (delta_resolution_method, UseDelta, Cols (qty_in_stock)),
    (max_resolution_method, UseMax (modified_ts), Cols (vendor, modified_ts)),
    (Default, Overwrite));
  
```

Conflict Resolution: Custom Methods

Use custom methods if:

- Your version of Oracle GoldenGate does not support ResolveConflict
- Testing rules are more complex than column compare



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

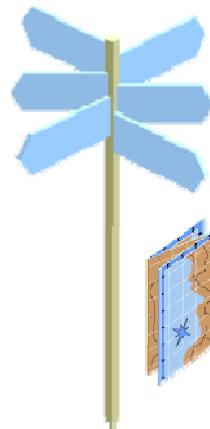
Example

```

Replicat reconf
AssumeTargetDefs
DiscardFile ./dirrpt/reconf.dsc, Purge
UserIDAlias gqualias
AllowDupTargetMap
IgnoreUpdates
Map WEST.* , Target EAST.* ;
GetUpdates
IgnoreInserts
IgnoreDeletes
Map WEST.PRODUCTS, Target WEST.PRODUCTS
SQLEXEC (id conflict, query
  "SELECT qty_in_stock FROM west.products WHERE product_number=:vpnum",
  params (vpnum = product_number)),
  colmap (product_number=product_number,
  product_name = product_name,
  qty_in_stock = @if(conflict.qty_in_stock <> BEFORE.qty_in_stock,
  @compute(conflict.qty_in_stock-(BEFORE.qty_in_stock - qty_in_stock)),
  qty_in_stock)
);
  
```

Roadmap

- Bidirectional Topics
- Looping
- Conflict Detection and Resolution
- Miscellaneous
 - Sequences
 - Truncates

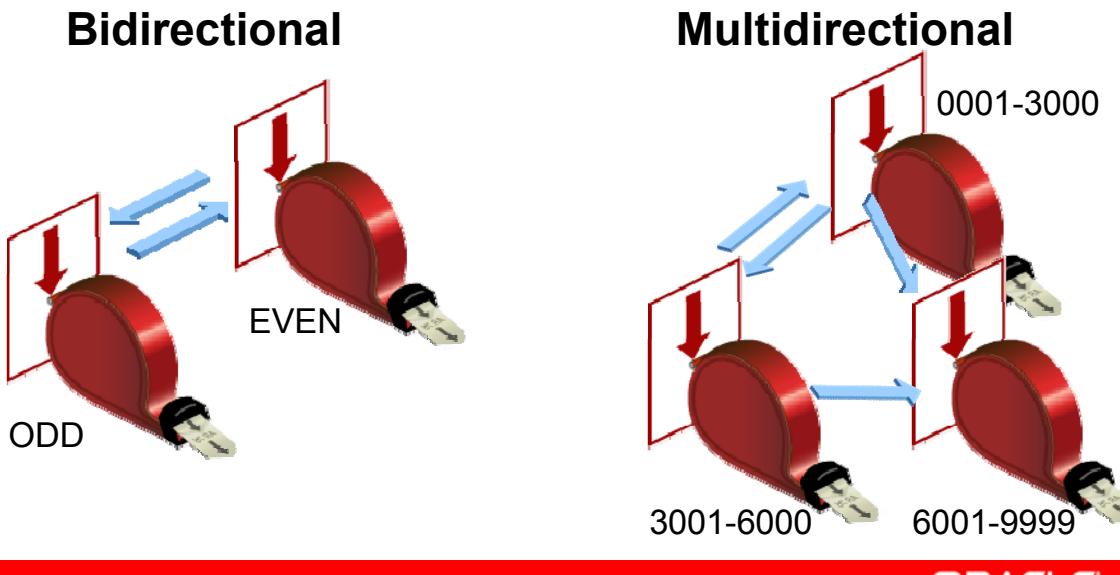


ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle Sequence Numbers

Oracle GoldenGate does not support the replication of sequence values in a bidirectional configuration.



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

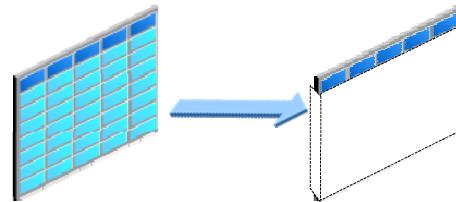
Oracle sequence numbers are used for a variety of purposes, such as to maintain a count, determine the order of entry, or generate a primary key value.

- **Bidirectional:** Use odd and even values to ensure that source and target sequence numbers are unique.
- **Multidirectional:** Each system must use a starting value and increment based on the number of systems.

Truncate Table Operations

- The TRUNCATE table operations cannot be detected for loops.
- Ensure that GetTruncates is ON for only one direction.
- Use IgnoreTruncates (default) for the other direction.
- Change database security so that truncates can be issued on only one system.

```
SQL> TRUNCATE mytable;
```



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Bidirectional replication of TRUNCATEs is not supported, but you can configure these operations to be replicated in one direction, while data is replicated in both directions. To replicate TRUNCATEs (if supported by Oracle GoldenGate for the database) in an active-active configuration, the TRUNCATEs must originate from only one database, and only from the same database each time.

Configure the environment as follows:

- Configure all database roles so that they cannot execute TRUNCATE from any database other than the one that is designated for this purpose.
- On the system where TRUNCATE will be permitted, configure the Extract and Replicat parameter files to contain the GetTruncates parameter.
- On the other system, configure the Extract and Replicat parameter files to contain the IgnoreTruncates parameter. No TRUNCATEs should be performed on this system by applications that are part of the Oracle GoldenGate configuration.

Quiz

Oracle GoldenGate supports bidirectional synchronization only for homogeneous environments.

- a. True
- b. False



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: b

Quiz

Which Oracle Database parameters can be used instead of a trace table to allow loop detection?

- a. ExcludeUser
- b. IgnoreTruncates
- c. ExcludeUserID
- d. UserExclude



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: a, c

Summary

In this lesson, you should have learned how to:

- Handle loop detection in a bidirectional environment
- Describe conflict resolution by applying net differences
- Decide whether to perform conflict resolution by using built-in `ResolveConflict` or custom coding
- Describe the limitations of using Oracle sequence numbers and truncate operations in a bidirectional configuration



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice 14 Overview: Configuring Bidirectional Replication

This practice covers the following topics:

- Two-way active-active data replication
- Conflict detection and resolution



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

15

DDL Replication

Data Description Language

ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Objectives

After completing this lesson, you should be able to:

- Describe the concept of DDL replication
- Explain the requirements for Oracle DDL replication
- Describe Oracle GoldenGate options for DDL replication
- Explain how to activate DDL capture



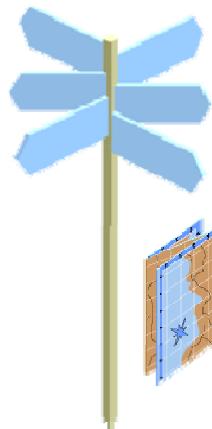
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

DDL: Data description language (for example, CREATE, ALTER, DROP)

DML: Data manipulation language (for example, INSERT, UPDATE, DELETE)

Roadmap

- Requirements and Restrictions
 - Naming
 - Wildcards
 - Sizes
- Syntax
- Active-Active Bidirectional



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Overview of DDL Replication

- DDL replication is available for all Oracle versions supported by Oracle GoldenGate DML synchronization.
- DDL in relation to DML:
 - DDL can be active with or without DML synchronization.
 - The same Extract/Replicat should be processing both DML and DDL to avoid timing issues.
- DDL operations are recognized differently by Extract and Replicat:
 - **Source:** DDL is disabled by default. Extract must be configured to enable.
 - **Target:** DDL is enabled by default to maintain data integrity.
 - Replicat must be configured to ignore or filter DDL.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate supports the synchronization of DDL operations from one database to another. The two supported environments for DDL synchronization in GoldenGate 12c are Oracle Database and Teradata. DDL synchronization can be active when:

- Business applications are actively accessing and updating the source and target objects
- Oracle GoldenGate transactional data synchronization is active

The components that support the replication of DDL and the replication of transactional data changes (DML) are independent of each other. Therefore, you can synchronize:

- Just DDL changes
- Just DML changes
- Both DDL and DML

DDL Replication Requirements and Restrictions

- Source and target data definitions must be identical: AssumeTargetDefs.
- wildcardResolve must remain set to Dynamic (default).
- Data pumps must be configured in Passthru mode.
- Data manipulation is not supported by the data pumps.
- Oracle GoldenGate user exits are not supported for DDL activity.
- Restrictions exist for DDL operations that involve user-defined types and LOB data.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

DDL support is valid only with log-based extraction. Other criteria include:

- **Object definitions:** Source and target object definitions must be identical (AssumeTargetDefs specified for Replicat). Neither data nor DDL conversion is supported.
- **Wildcard resolution:** Standard Oracle GoldenGate wildcards (*) can be used with certain parameter options when synchronizing DDL operations. WildcardResolve is now set by default to DYNAMIC and must remain so for DDL support.
- **DDL and data pumps:** When Extract data pumps are being used, tables for which DDL is being replicated must be configured in Passthru mode. DDL filtering, manipulation, and error handling is not supported by data pumps.
- **User exits:** Oracle GoldenGate user exit functionality is not supported for use with DDL synchronization activities (user exit logic cannot be triggered based on DDL operations.) User exits can be used with concurrent DML processing.
- **LOB data:** With LOB data, Extract might fetch a LOB value from a Flashback Query, and Oracle does not provide Flashback capability for DDL (except DROP). When a LOB is fetched, the object structure reflects current metadata, but the LOB record in the transaction log reflects old metadata. Refer to the *Oracle GoldenGate Reference Guide* for information about this topic.

DDL Replication Requirements and Restrictions

- DDL on objects in Table or Map statements inherit the limitations on allowed characters in those parameters.
- Restrictions exist for DDL operations that involve stored procedures.
- Schema names using Oracle reserved names are ignored.
- There are shared schema/tablespace restrictions.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

- **User-defined types:** DDL operations that involve user-defined types generate implied DML operations on both the source and target. To avoid SQL errors that would be caused by redundant operations, Oracle GoldenGate does not replicate those DML operations. If DML is being replicated for a user-defined type, Extract must process all of those changes before DDL can be performed on the object. Because UDT data might be fetched by Extract, the reasons for this rule are similar to those that apply to LOB columns.
- **SQLEXEC:** Objects that are affected by a stored procedure must exist with the correct structure before the execution of SQL. Consequently, DDL that affects the structure must happen before the SQLEXEC statement executes.
- **Stand-alone SQLEXEC statements:** Objects affected by a stand-alone SQLEXEC statement must exist before Oracle GoldenGate processes start. This means that DDL support must be disabled for these objects; otherwise, DDL operations could change or delete the object before SQLEXEC executes.
- **Shared Schema/Tablespace:** The trigger script `ddl_setup.sql` will fail if the tablespace for this schema is shared by any other users. However, it will not fail if the default tablespace does not have `AUTOEXTEND` set to `ON`, which is the recommended setting. This restriction does not apply for Integrated Extract, which does not use triggers.

DDL Replication Requirements and Restrictions

- The GetTruncates parameter should not be used with full DDL support.
- Table name cannot be longer than 16 characters plus quotation marks for ALTER TABLE RENAME.
- ALTER TABLE ..MOVE TABLESPACE
 - It is supported when tablespace is all SMALLFILE or BIGFILE.
 - Stop Extract before issuing a MOVE TABLESPACE.
- ALTER DATABASE and ALTER SYSTEM are not captured.
- DDL statements larger than 2 MB require special handling.



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

GetTruncates: GoldenGate supports the synchronization of TRUNCATES as a stand-alone function (independently of full DDL synchronization) or as part of full DDL synchronization. If you are using DDL synchronization, disable stand-alone TRUNCATE synchronization to avoid errors caused by duplicate operations.

Table names: ALTER TABLE RENAME fails if the old or new table name is longer than 18 characters (16 for the name and two for the quotation marks). Oracle allows only 18 characters for a rename, because of the ANSI limit for identifiers.

Long DDL statements: Oracle GoldenGate supports the capture and replication of Oracle DDL statements of up to 2 MB in length (including some internal Oracle GoldenGate maintenance information). Extract will skip statements that are greater than the supported length, but the `ddl_ddl2file.sql` script can be used to save the skipped DDL to a text file in the `USER_DUMP_DEST` directory of Oracle. To use the new support, the DDL trigger must be reinstalled in INITIALSETUP mode, which removes all of the DDL history. For more information, see the *Oracle GoldenGate for Oracle Installation and Setup Guide*.

Characteristics for DDL Replication

- New names must be specified in Table and Map statements.
- Extract sends all DDL to each trail when writing to multiple trails.
- Oracle GoldenGate supports all DDL operations up to 2 MB in size on the following objects:

clusters	roles	triggers
functions	sequences	types
indexes	synonyms	users
packages	tables	views
procedure	tablespaces	materialized views

**ORACLE®**

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

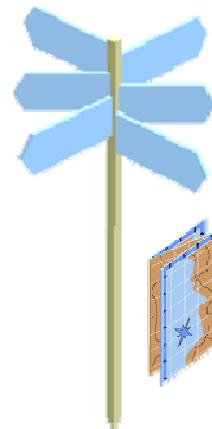
To work around remote permissions issues that may arise when different users are being used on the source and target, `RENAME` will always be converted to equivalent `ALTER TABLE RENAME` for Oracle.

New names must be specified in Table/Map statements to do the following:

- Replicate DML operations on tables resulting from a `CREATE` or `RENAME`
- `CREATE USER` and then move new or renamed tables into that schema

Roadmap

- Requirements and Restrictions
- Syntax
 - Scope
 - Parameters
 - Strings
 - Mapping
- Active-Active Bidirectional

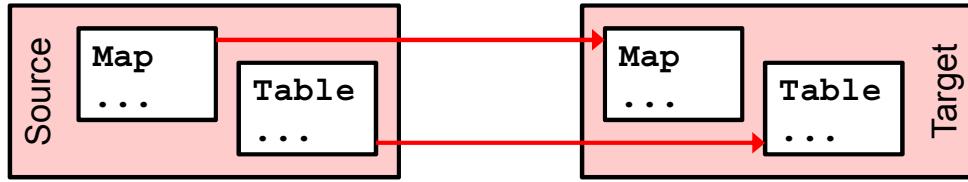


ORACLE®

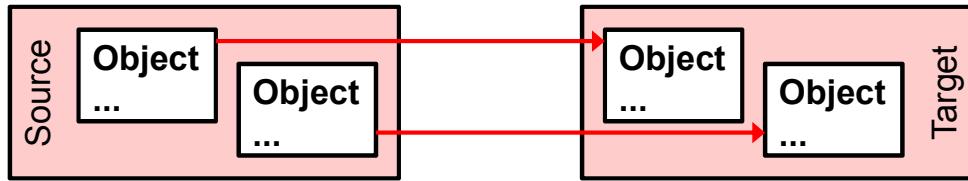
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

DDL Scopes

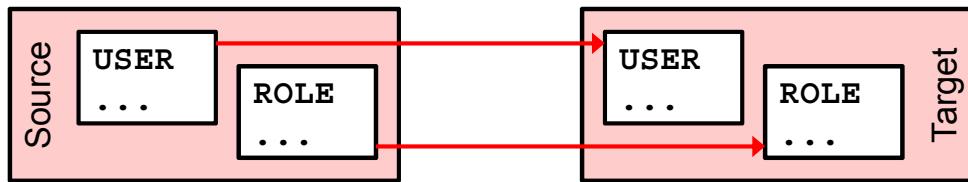
- **Mapped**



- **UnMapped**



- **OTHER**



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Database objects are classified into *scopes*. A scope is a category that defines how DDL operations on an object are handled by Oracle GoldenGate. The use of scopes enables granular control over the filtering of DDL operations, string substitutions, and error handling. The three flavors of scopes are:

- **Mapped**

- Is specified in a Table or Map statement
- Operations: CREATE, ALTER, DROP, RENAME, GRANT*, REVOKE*
- Objects: TABLE*, INDEX, TRIGGER, SEQUENCE*, MATERIALIZED VIEW*

- **UnMapped**

- Does not have a Table or Map statement
- Same list of operations and objects as Mapped

- **OTHER**

- Table or Map statements do not apply.
- DDL operations other than those listed above
- Examples: CREATE USER, CREATE ROLE, ALTER TABLESPACE

* Operations are for only those objects with an asterisk.

DDL Parameter

The DDL parameter, which is valid for Extract and Replicat, enables DDL support and filters the operations:

```
DDL [
{Include | Exclude}
  [, Mapped | UnMapped | OTHER | ALL]
  [, Optype <type>]
  [, ObjType '<type>']
  [, ObjName "<name>"] Note 'single' versus "double" quotation marks. They must be this way.
  [, InStr '<string>']
  [, InStrComments '<comment_string>']
]
[...]
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Only one DDL parameter can be used in a parameter file, but you can combine multiple inclusion and exclusion options to filter the DDL to the required level. When combined, multiple option specifications are linked logically as AND statements. All criteria specified with multiple options must be satisfied for a DDL statement to be replicated.

Example:

```
DDL &
INCLUDE UNMAPPED &
OPTYPE alter &
OBJTYPE 'table' &
OBJNAME users.tab* &
INCLUDE MAPPED OBJNAME * &
EXCLUDE MAPPED OBJNAME "temporary.tab"
```

In the previous example, if the object name "temporary.tab" was not case-sensitive (that is, if the real object name is TEMPORARY.TAB), you can leave the "quotation marks" and simply say temporary.tab, as was done with users.tab*.

- **Include | Exclude:** Identifies the beginning of an *inclusion* or *exclusion* clause. **Include** includes specified DDL for capture or replication. **Exclude** excludes specified DDL from being captured or replicated. The inclusion or exclusion clause must consist of the **Include** or **Exclude** keyword followed by any valid combination of other options of the **DDL** parameter. An **Exclude** must be accompanied by a corresponding **Include** clause. An **Exclude** takes priority over any **Includes** that contain the same criteria. You can use multiple inclusion and exclusion clauses.
- **Mapped | UnMapped | Other | ALL:** Applies **Include** or **Exclude** based on the DDL operation scope
 - **Mapped** applies to DDL operations that are of **Mapped** scope.
 - **UnMapped** applies to DDL operations that are of **UnMapped** scope.
 - **OTHER** applies to DDL operations that are of **OTHER** scope.
 - **ALL** applies to DDL operations of all scopes. DDL **Exclude ALL** maintains up-to-date metadata on objects, while blocking the replication of the DDL operations themselves.
- **OpType <type>:** Applies **Include** or **Exclude** to a specific type of DDL operation. For **<type>**, use any DDL command that is valid for the database, such as **CREATE**, **ALTER**, and **RENAME**.
- **ObjType '<type>':** Applies **Include** or **Exclude** to a specific type of database object. For **<type>**, use any object type that is valid for the database, such as **TABLE**, **INDEX**, **TRIGGER**, **USER**, or **ROLE**. Enclose the object type within single quotation marks.
- **ObjName "<name>":** Applies **Include** or **Exclude** to the name of an object, for example, a table name. Provide a double-quoted string as input. Wildcards can be used. If you do not qualify the object name for Oracle, the owner is assumed to be the GoldenGate user. When using **ObjName** with **Mapped** in a Replicat parameter file, the value for **ObjName** must refer to the name specified with the **Target** clause of the **Map** statement. For DDL that creates triggers and indexes, the value for **ObjName** must be the name of the base object, not the name of the trigger or index. For **RENAME** operations, the value for **ObjName** must be the new table name. Make sure that “double quotation marks” and ‘single quotation marks’ are used correctly.
- **InStr '<string>':** Applies **Include** or **Exclude** to DDL statements that contain a specific character string within the command syntax itself, but not within comments. Enclose the string within single quotation marks. The search string is not case-sensitive.
- **InStrComments '<comment_string>':** Applies **Include** or **Exclude** to DDL statements that contain a specific character string within a comment, but not within the DDL command itself. By using **InStrComments**, you can use comments as a filtering agent. Enclose the string within single quotation marks. The search string is not case-sensitive. You can combine **InStr** and **InStrComments** options to filter on a string in the command syntax and in the comments.

DDL String Substitution

- The `DDLSUBST` parameter substitutes strings in a DDL operation.
- Multiple statements can be used.
- In the following example, the string '`cust`' is replaced with the string '`customers`' for tables owned by "`fin`".

```
:
DDLSUBST 'cust' With 'customers'
Include All ObjType 'table' ObjName "fin.*"
:
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

You can substitute strings within a DDL operation while it is being processed by Oracle GoldenGate. This feature provides a convenience for changing and mapping directory names, comments, and other things that are not directly related to data structures.

Syntax:

```
DDLSUBST '<search_string>' With '<replace_string>'
  [ {Include | Exclude}
  [, All | Mapped | UnMapped | OTHER]
  [, OpType <type>]
  [, ObjType <type>]
  [, ObjName "<name>"]
  [, InStr '<string>']
  [, InStrComments '<comment_string>']
  ]
  [...]
```

As with the previous command `DLL`, the object name `ObjName` requires “double-quotes” only if it is case-sensitive.

DDL Error Handling

- The `DDLError` parameter contains specific error-handling rules to handle a full range of anticipated errors.
- Extract syntax:

```
DDLError [RestartSkip <num_skips>]  
[SkipTriggerError <num_errors>]
```

- Replicat syntax:

```
DDLError  
<error> | Default {<response>}  
[RetryOp MaxRetries <n> [RetryDelay <delay>]]  
{Include <clause> | Exclude <clause>}  
[, IgnoreMissingTables | AbendOnMissingTables]  
[, RestartCollisions | NoRestartCollisions]
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

In the syntax in the slide, `<response>` can be `Ignore`, `ABEND`, or `Discard`.

- Use the Extract option of the `DDLError` parameter to handle errors on objects found by Extract for which metadata cannot be found.
- Use the Replicat options of the `DDLError` parameter to handle errors that occur when DDL is applied to the target database.

DDLOptions for Oracle

```
DDLOptions
[, AddTranData [ABEND | RetryOp <RetryDelay <seconds>
                 MaxRetries <retries>]
[, DefaultUserPassword <password> <encrypt_type>
   [EncryptKey {Default | <keyname>}]]
[, GetApplOps | IgnoreApplOps]
[, GetReplicates | IgnoreReplicates]
[, MapDerived | NoMapDerived]
[, MapSessionSchema] <src_schema> Target <trgt_schema>
[, NoCrossRename]
[, Password EncryptKey [Default | EncryptKey <keyname>]
[, RemoveComments {Before | After}]
[, ReplicatePassword | NoReplicatePassword]
[, Report | NoReport]
[, UpdateMetadata]
[, UseOwnerForSession]
```



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

AddTranData is used to:

- Enable Oracle's supplemental logging automatically for new tables created with a CREATE TABLE
- Update supplemental logging for tables affected by an ALTER TABLE to add or drop columns
- Update supplemental logging for tables that are renamed
- Update supplemental logging for tables where unique or primary keys are added or dropped

DefaultUserPassword is valid for Replicat. It specifies a different password for a replicated {CREATE | ALTER} User <name> IDENTIFIED BY <password> statement from the one used in the source statement. The password may be entered as clear text or encrypted using the default or a user-defined <keyname> from ENCKEYS. When using DefaultUserPassword, use the NoReplicatePassword option of DDLOptions for Extract. <encrypt_type> is AES128, AES192, AES256, Blowfish.

GetApplOps | IgnoreApplOps are valid for Extract. This controls whether or not DDL operations produced by business applications *except* Replicat are included in the content that Extract writes to a trail or file. The default is GetApplOps.

GetReplicates | **IgnoreReplicates** is valid for Extract. It controls whether or not DDL operations produced by Replicat are included in the content that Extract writes to a trail or file. The default is `IgnoreReplicates`.

MapDerived | **NoMapDerived** is valid for Replicat. It controls how derived Object (for example, indexes) names are mapped. With `MapDerived`, if a `Map` statement exists for the derived object, that is used. Otherwise, the name is mapped to the name specified in the `Target` clause of the `Map` statement for the base object. `MapDerived` is the default. `NoMapDerived` overrides any explicit `Map` statements that contain the name of the derived object and prevents name mapping.

MapSessionSchema is valid for Replicat on Oracle. It enables a source session schema to be mapped to (transformed to) a different session schema on the target. Wildcards are not supported. You can use multiple `MapSessionSchema` parameters to map different schemas.

NoCrossRename is valid for Extract on Oracle. It assumes that tables excluded from the GoldenGate configuration will not be renamed to names that are in the configuration. `NoCrossRename` improves performance by eliminating processing that otherwise is required to keep track of excluded tables in case they get renamed to an included name.

Password EncryptKey is valid for Extract. It directs Extract to encrypt all passwords in source DDL before writing the DDL to the trail.

RemoveComments BEFORE removes comments before the DDL operation is processed by Extract or Replicat. AFTER removes comments after they are used for string substitution.

ReplicatePassword is valid for Extract. It applies to the password in a {`CREATE` | `ALTER`} `USER <user> IDENTIFIED BY <password>` command. By default, GoldenGate uses the source password in the target `CREATE` or `ALTER` statement. To prevent the source password from being sent to the target, use `NoReplicatePassword`.

Report | **NoReport** is valid for Extract and Replicat. It controls whether or not expanded DDL processing information is written to the report file. The default of `NoReport` reports basic DDL statistics. `Report` adds the parameters being used and a step-by-step history of the operations that were processed.

DefaultUserPassword is valid for Replicat. It specifies a different password for a replicated {`CREATE` | `ALTER`} `USER <name> IDENTIFIED BY <password>` statement from the one used in the source statement. The password may be entered as clear text or encrypted using the default or a user-defined `<keyname>` from `ENCKEYS`. When using `DefaultUserPassword`, use the `NoReplicatePassword` option of `DDLOptions` for Extract.

UpdateMetadata is valid for Replicat. Use in an active-active bidirectional configuration. This parameter notifies Replicat on the system where DDL originated that this DDL was propagated to the other system, and that Replicat should not update its object metadata cache to match the new metadata.

UseOwnerForSession is valid for Replicat. Forces the schema of an unqualified object in the Replicat DDL statement to be that of the Replicat session schema, instead of the schema in an `ALTER SESSION SET CURRENT_SCHEMA` statement, which is the default behavior.

Mapping Schemas

MapSessionSchema enables the mapping of a source session schema to another schema on the target:

Extract

```
DDL Include ObjName "SRC.*" &
Include ObjName "SRC1.*"
Table SRC.*;
Table SRC1.*;
```

Replicat

```
:
DDLOptions MapSessionSchema SRC Target DST
DDLOptions MapSessionSchema SRC1 Target DST1
Map SRC.* , Target DST.*;
Map SRC1.* , Target DST1.*;
DDL Include ObjName "DST.*" &
Include ObjName "DST1.*"
```

ORACLE

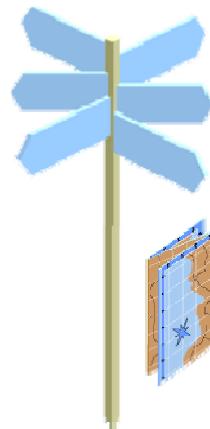
Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The slide shows DDL capture and mapping configurations for Extract and Replicat. The MapSessionSchema parameter enables a source session schema to be mapped to a different session schema on the target. The example in the slide shows how MapSessionSchema works to allow the mapping of a source session schema to another schema on the target.

The naming conventions can be extended to three parts for multitenant databases.

Roadmap

- Requirements and Restrictions
- Syntax
- Active-Active Bidirectional
 - Metadata
 - Markers
 - Triggers



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Supporting DDL in an Active-Active Bidirectional Configuration

- The `UpdateMetadata` parameter notifies Replicat on the system where DDL originated that this DDL was propagated to the other system.
- This functionality keeps Replicat metadata cache synchronized with the current metadata of the local database:

```
DDLOptions UpdateMetadata
```

- This is valid for Replicat (Oracle only).



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Oracle GoldenGate offers support for DDL in an active-active bidirectional configuration. The `UpdateMetadata` option of the `DDLOptions` parameter notifies Replicat on the system where DDL originated that this DDL was propagated to the other system, and that Replicat should now update its object metadata cache to match the new metadata. This keeps Replicat's metadata cache synchronized with the current metadata of the local database. This is for the Oracle platform only.

Activating Oracle DDL Capture

- Process differs for older versus newer database versions.
- DDL marker table: GGS_MARKER
 - Stores DDL information
- DDL history table: GGS_DDL_HIST
 - Stores object metadata history
- DDL trigger: GGS_DDL_TRIGGER_BEFORE
 - Activates when there is a DDL operation
 - Writes to the marker and history tables
- User role: GGS_GGSUser_ROLE
 - Establishes the role needed for DDL replication
 - Should be the user that executes Extract



ORACLE

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

These database objects are used for the replication of operations. They are either installed manually (by running SQL*Plus scripts for database versions 11.2.0.3 and earlier) or are automatically built-in (database versions 11.2.0.4 or later).

DDL marker table: The DDL Marker table receives only inserts and its rows can be periodically purged. The name can be set in the GLOBALS parameter file, but if it is not, the default name is GGS_MARKER. Do not delete the DDL marker table if you plan to continue processing DDL. The marker table and the DDL trigger are interdependent. If you attempt to remove the marker table without first removing the trigger, the following error will be generated:

```
"ORA-04098: trigger 'SYS.GGS_DDL_TRIGGER_BEFORE' is invalid and
failed re-validation"
```

DDL history table: The DDL History table receives inserts, updates, deletes. It contains the SQL statement that was issued by the user. The default table name is GGS_DDL_HIST. Caution must be used if purging this table.

DDL trigger: The DDL trigger is installed with some packages. The default trigger name is GGS_DDL_TRIGGER_BEFORE.

User Role: The default user role name is GGS_GGSUser_ROLE. In older versions of the database, you needed to grant this role even if the user has the DBA role.

Other supporting database objects:

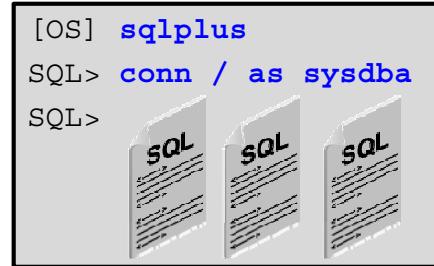
- Internal setup table
- DumpDDL tables (for viewing DDL history)
- ddl_pin (pins tracing for performance evaluation)
- Sequence used for a column in the marker table

Specify the schema name in GLOBALS : GGSHEMA <schema_name>. Note that these tables grow large, so it is recommended that you specify cleanup in the Manager parameters.

DDL Setup Scripts

Classic versus Integrated

- Classic
 - @marker_setup.sql
 - @ddl_setup.sql
 - @role_setup_.sql
 - grant role to user
 - @ddl_enable.sql
- Integrated
 - No need to run the scripts
 - Requires Oracle database 11.2.0.4 or later
- DDL Trigger
 - Enable
 - Disable



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

The DDL setup scripts are included with Oracle GoldenGate. If you are using Oracle Database 11.2.0.3 or earlier, or are running in Classic mode, then the scripts need to be run from SQL*Plus as sysdba.

In Integrated mode, you do not need to run the setup scripts, nor do you need to do anything with the DDL Trigger. (It will be ignored.)

- The DDL Trigger must be Enabled for:
 - Classic mode DDL replication (any version database)
 - Oracle Databases older than 11.2.0.3
- The DDL Trigger may be Disabled for:
 - Integrated mode DDL replication. This does not require the DDL trigger, but can optionally use it with the ForceClassicMetaData parameter.
 - Replication of only DML

Quiz

Which of the following statements is true?

- a. Oracle GoldenGate supports the synchronization of only DDL changes.
- b. Oracle GoldenGate supports the synchronization of only DML changes.
- c. Oracle GoldenGate supports the synchronization of both DDL and DML changes.



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: c

Quiz

What parameter in DDLOptions supports DDL replication in an active-active bidirectional configuration?

- a. MapSessionSchema
- b. IgnoreApp1Ops
- c. UpdateMetadata
- d. NoCrossRename



ORACLE®

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Answer: c

Summary

In this lesson, you should have learned how to:

- Explain how DDL operations are categorized into scopes
- Use the `DDL` and `DDLOptions` parameters
- Run the necessary scripts to set up the database for DDL capture



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Practice 15 Overview: Configuring DDL Replication

This practice covers the following topics:

- Setting up DDL replication on the database
- Setting up DDL replication for Oracle GoldenGate
- Viewing DDL verification and statistics



Copyright © 2014, Oracle and/or its affiliates. All rights reserved.