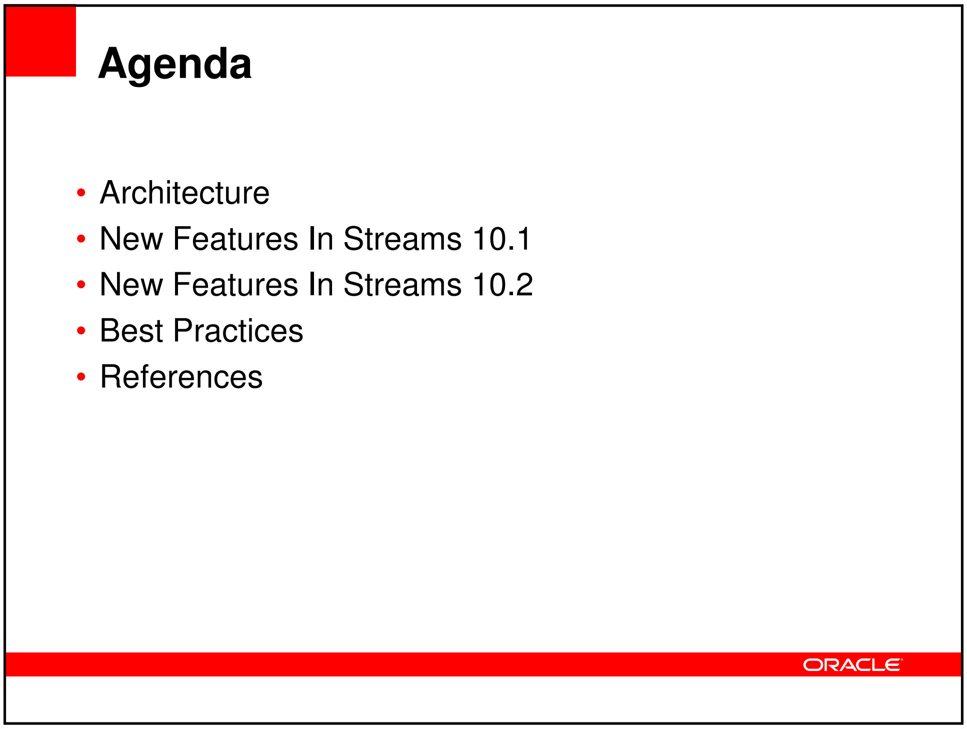
ORACLE

ORACLE

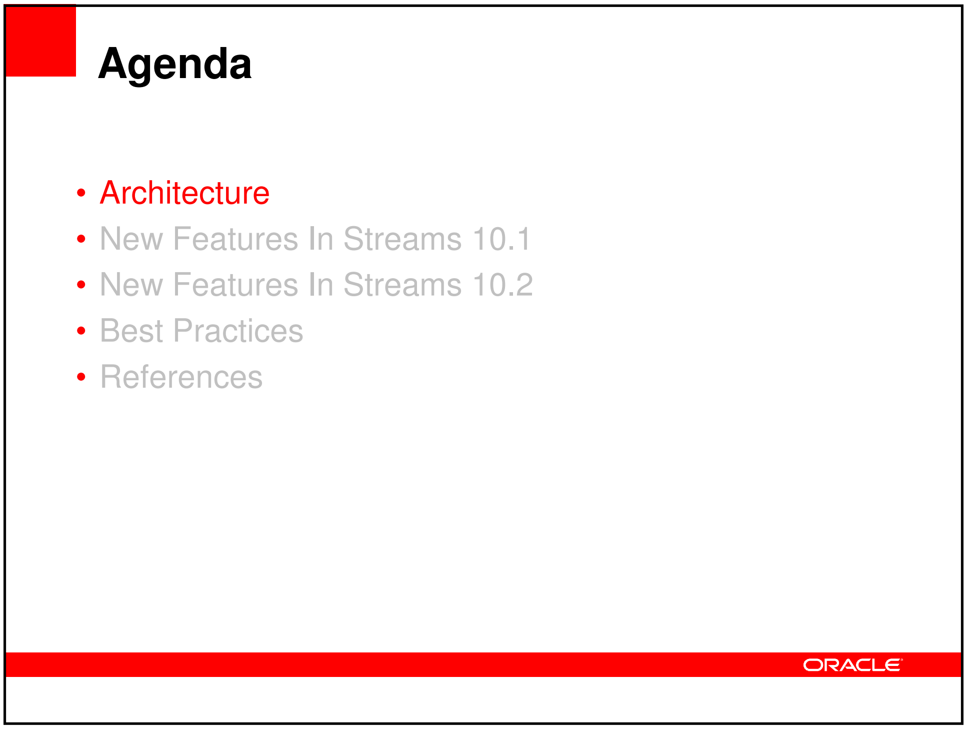
Streams - Overview

Sanjeev Joglekar
Technical Project Manager



Agenda

- Architecture
- New Features In Streams 10.1
- New Features In Streams 10.2
- Best Practices
- References



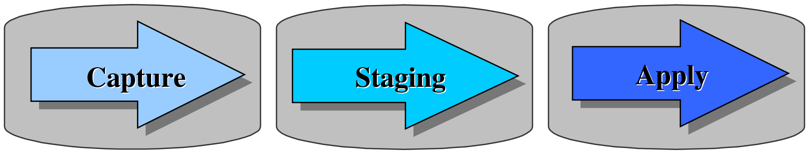
Agenda

- Architecture
- New Features In Streams 10.1
- New Features In Streams 10.2
- Best Practices
- References

Oracle Streams

- Introduced in Oracle 9.2
- New solution for information sharing
  - Replication
  - Message queuing
  - Data warehouse loading
- Additional features over Advanced Replication
  - Data transformation
  - Between Oracle and third-party databases

Streams Elements

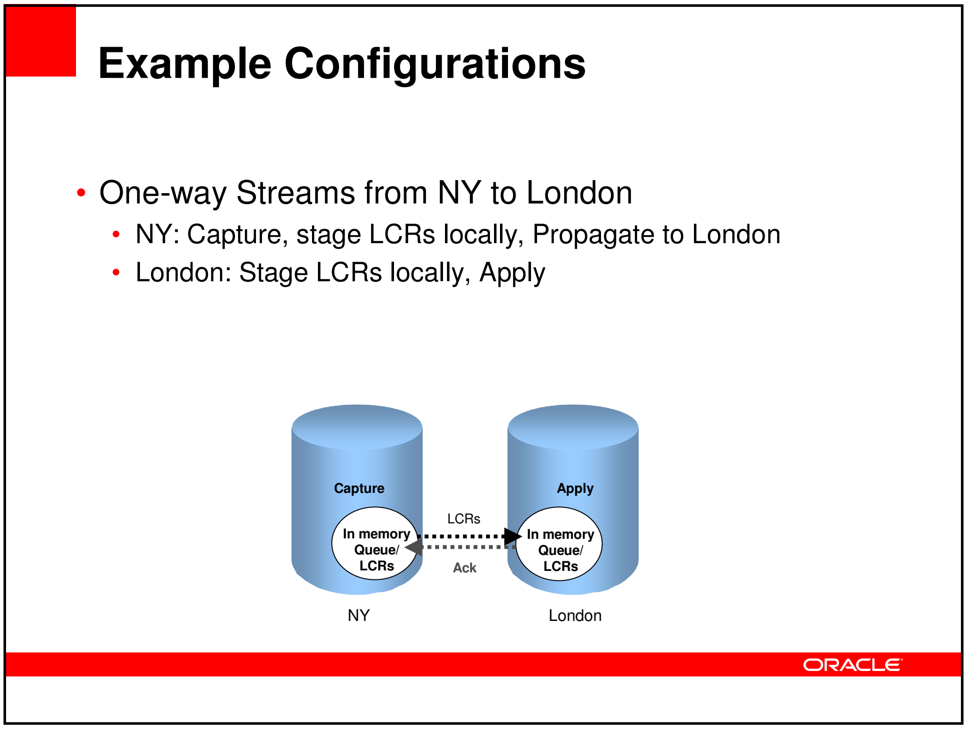


Capture

Staging

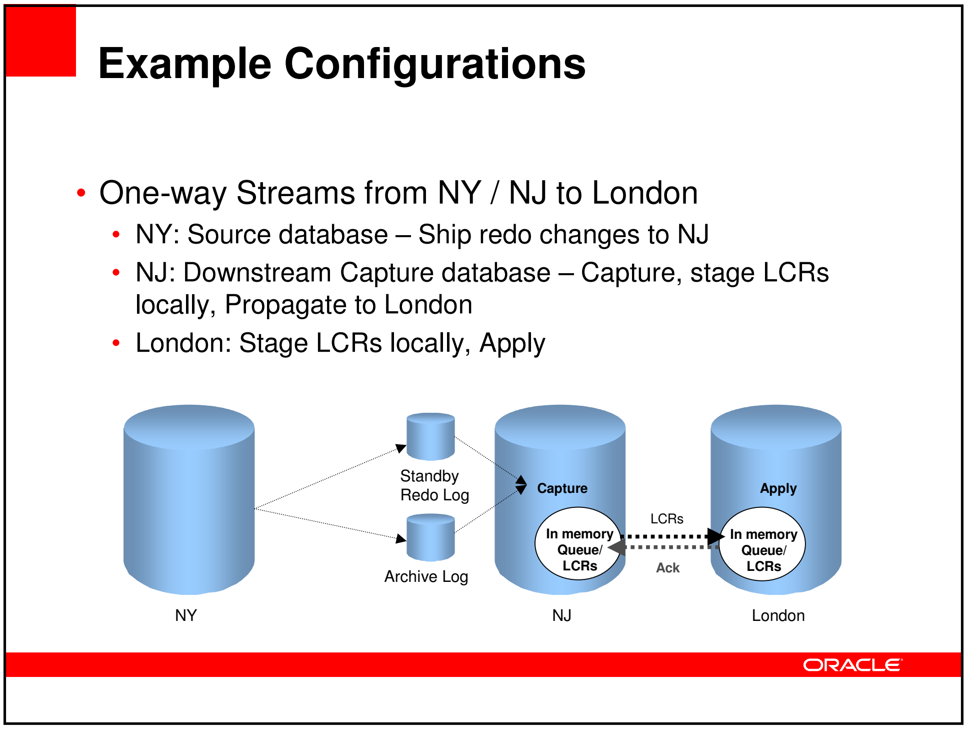
Apply

- Capture
  - Runs on Capture database
- Staging and Propagation
  - Staging: Store changes locally in SGA
  - Propagation: Send changes to Apply or another Staging database
- Apply
  - Runs on Apply database



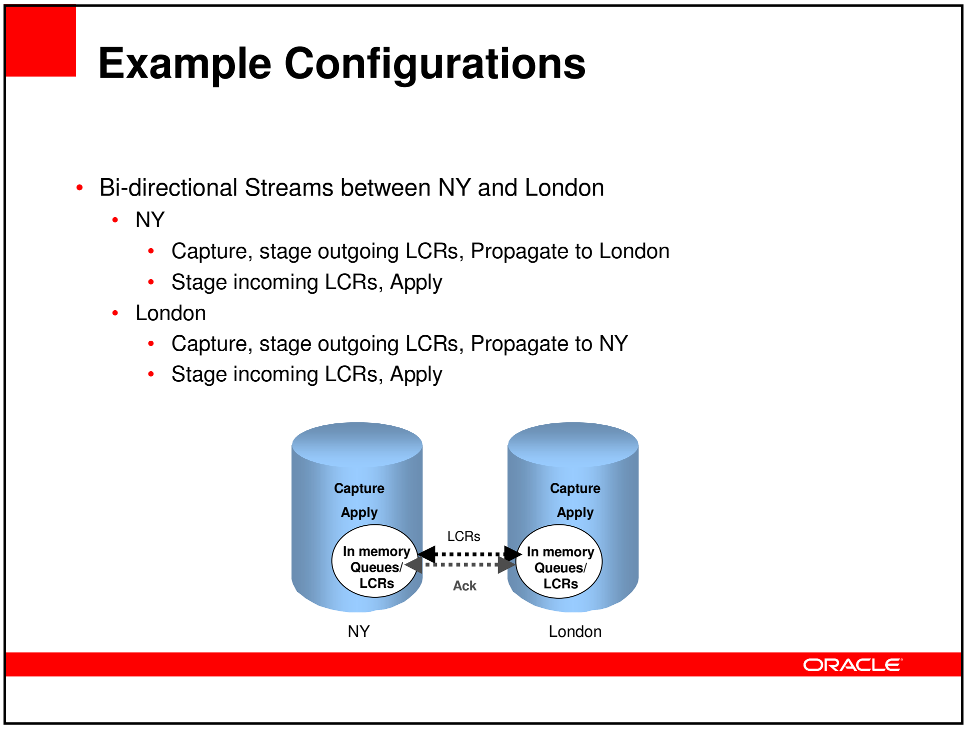
Example Configurations

- One-way Streams from NY to London
  - NY: Capture, stage LCRs locally, Propagate to London
  - London: Stage LCRs locally, Apply



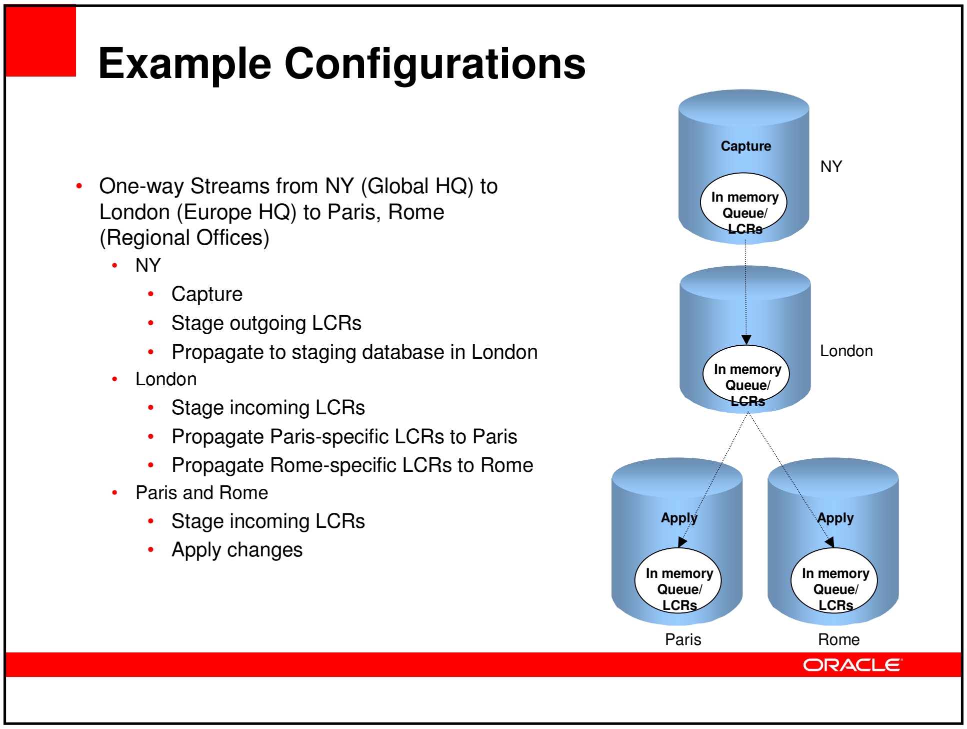
Example Configurations

- One-way Streams from NY / NJ to London
  - NY: Source database – Ship redo changes to NJ
  - NJ: Downstream Capture database – Capture, stage LCRs locally, Propagate to London
  - London: Stage LCRs locally, Apply



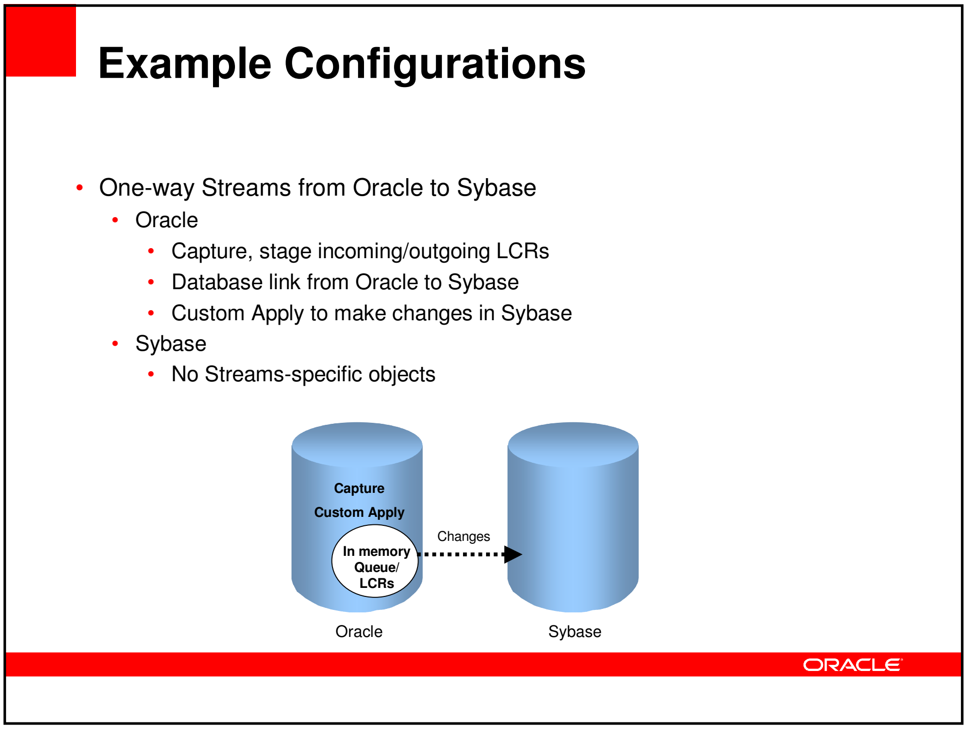
Example Configurations

- Bi-directional Streams between NY and London
  - NY
    - Capture, stage outgoing LCRs, Propagate to London
    - Stage incoming LCRs, Apply
  - London
    - Capture, stage outgoing LCRs, Propagate to NY
    - Stage incoming LCRs, Apply



Example Configurations

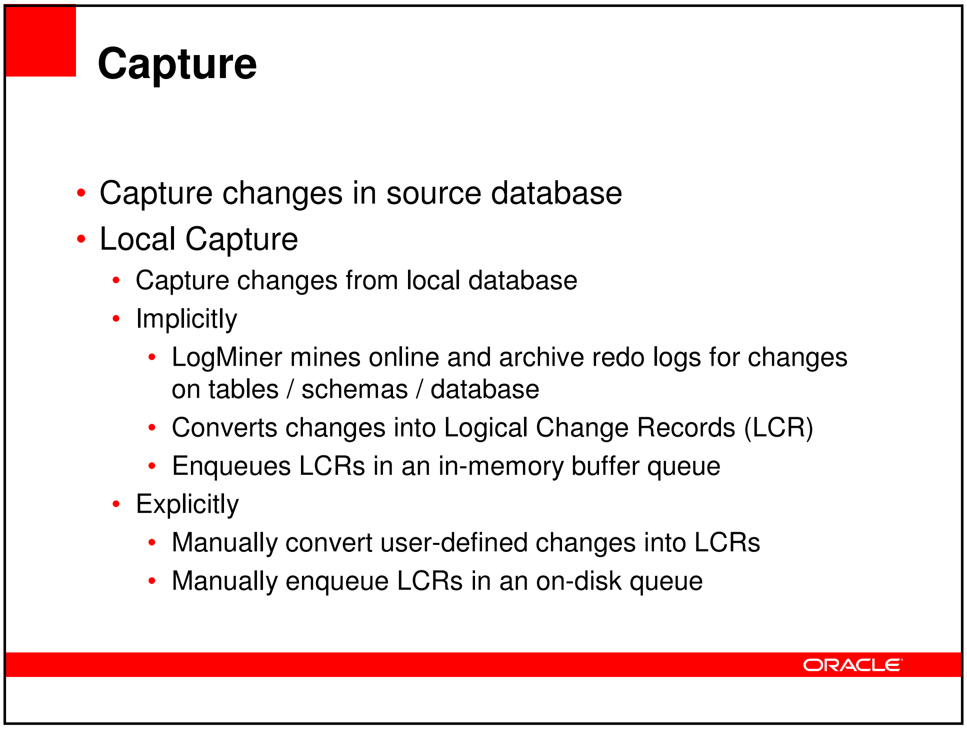
- One-way Streams from NY (Global HQ) to London (Europe HQ) to Paris, Rome (Regional Offices)
  - NY
    - Capture
    - Stage outgoing LCRs
    - Propagate to staging database in London
  - London
    - Stage incoming LCRs
    - Propagate Paris-specific LCRs to Paris
    - Propagate Rome-specific LCRs to Rome
  - Paris and Rome
    - Stage incoming LCRs
    - Apply changes



Example Configurations

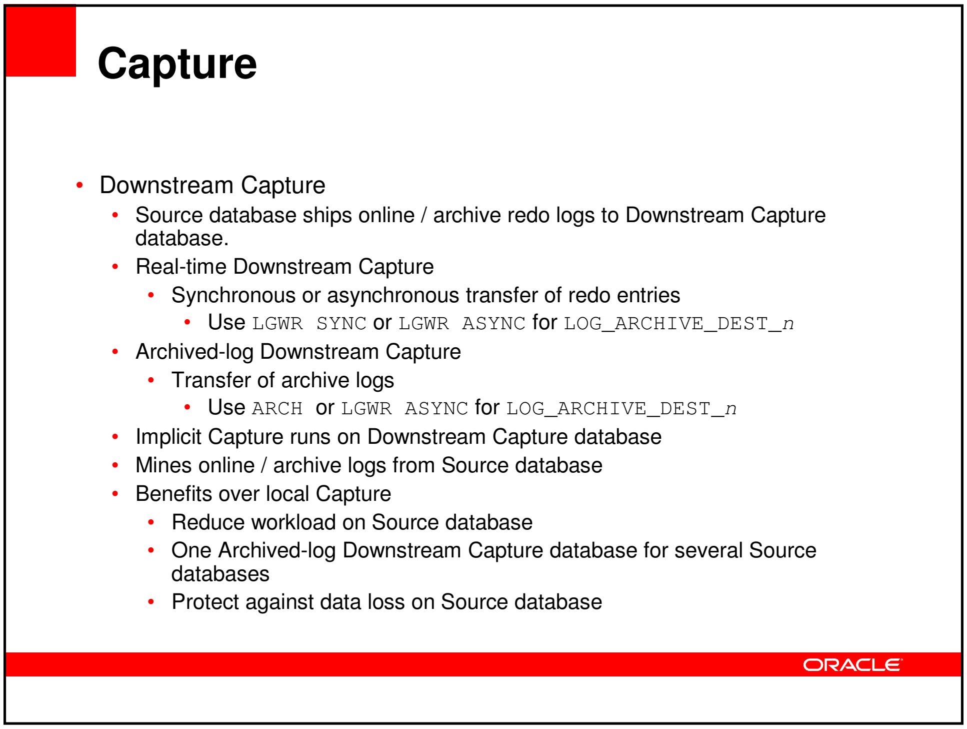
- One-way Streams from Oracle to Sybase
  - Oracle
    - Capture, stage incoming/outgoing LCRs
    - Database link from Oracle to Sybase
    - Custom Apply to make changes in Sybase
  - Sybase
    - No Streams-specific objects

Capture
Custom Apply
In memory Queue/ LCRs
Changes
Oracle
Sybase



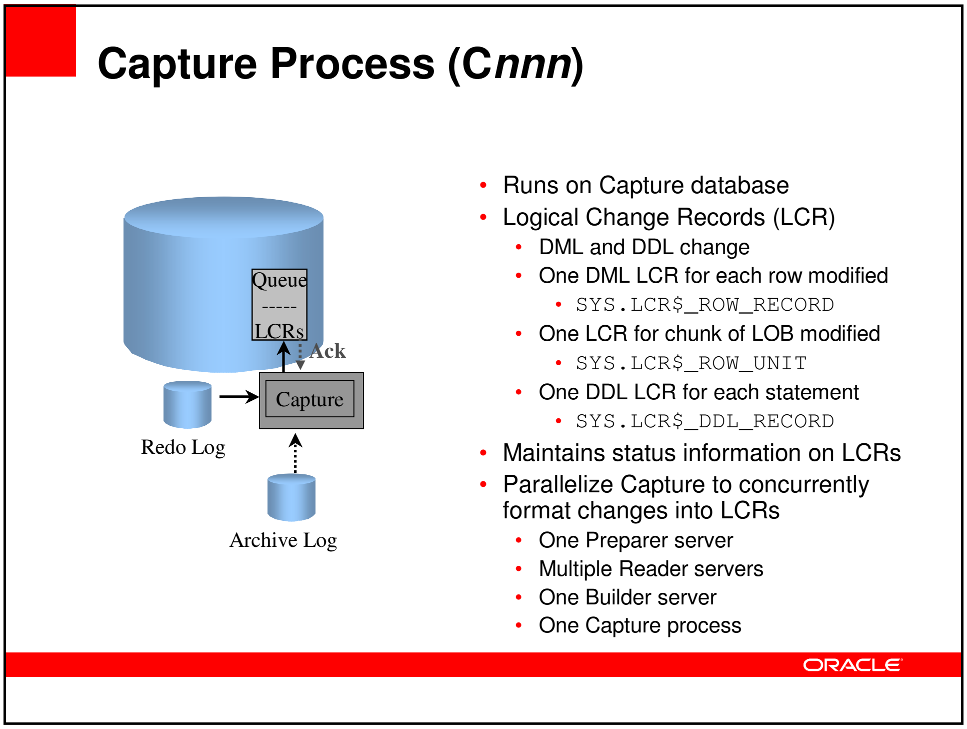
Capture

- Capture changes in source database
- Local Capture
  - Capture changes from local database
  - Implicitly
    - LogMiner mines online and archive redo logs for changes on tables / schemas / database
    - Converts changes into Logical Change Records (LCR)
    - Enqueues LCRs in an in-memory buffer queue
  - Explicitly
    - Manually convert user-defined changes into LCRs
    - Manually enqueue LCRs in an on-disk queue



Capture

- Downstream Capture
  - Source database ships online / archive redo logs to Downstream Capture database.
  - Real-time Downstream Capture
    - Synchronous or asynchronous transfer of redo entries
      - Use `LGWR SYNC` or `LGWR ASYNC` for `LOG_ARCHIVE_DEST_`*`n`*
  - Archived-log Downstream Capture
    - Transfer of archive logs
      - Use `ARCH` or `LGWR ASYNC` for `LOG_ARCHIVE_DEST_`*`n`*
  - Implicit Capture runs on Downstream Capture database
  - Mines online / archive logs from Source database
  - Benefits over local Capture
    - Reduce workload on Source database
    - One Archived-log Downstream Capture database for several Source databases
    - Protect against data loss on Source database



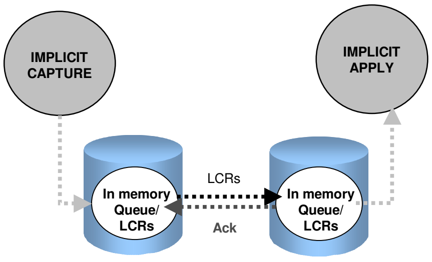
Capture Process (C*nnn*)

- Runs on Capture database
- Logical Change Records (LCR)
  - DML and DDL change
  - One DML LCR for each row modified
    - `SYS.LCR$_ROW_RECORD`
  - One LCR for chunk of LOB modified
    - `SYS.LCR$_ROW_UNIT`
  - One DDL LCR for each statement
    - `SYS.LCR$_DDL_RECORD`
- Maintains status information on LCRs
- Parallelize Capture to concurrently format changes into LCRs
  - One Preparer server
  - Multiple Reader servers
  - One Builder server
  - One Capture process

Staging

- Staging area = In-memory buffer queue
  - Oracle 9.2: within Shared Pool
    - `SHARED_POOL_SIZE, _FIRST_SPARE_PARAMETER`
  - Oracle 10g: within Streams Pool
    - `STREAMS_POOL_SIZE`
- Queue based on new data type `SYS.ANYDATA`
  - Accommodates any data type
- Capture process stages LCRs in buffer queue before propagating them to Apply or Staging database.
  - Messages purged when consumed or received by subscribers
- Apply process stages LCRs in buffer queue before applying them locally.

Propagation

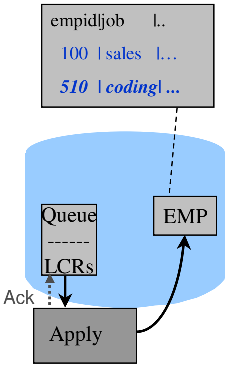


- Propagation job on Capture site
- Job Queue processes (J*nnn*)
- Propagate from one staging area to another
- Queue forwarding
  - Propagate changes without applying locally
- Apply forwarding
  - Apply changes locally, recapture, and propagate
- Maintain status information on LCRs

Apply

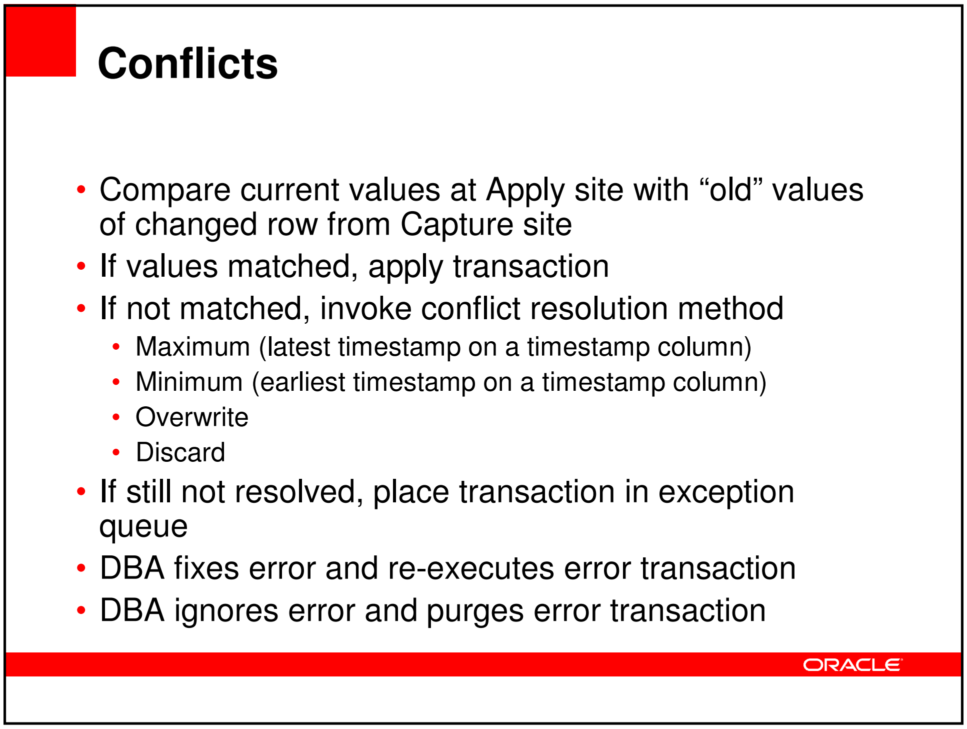
- Apply LCRs at Apply database
- Implicitly
  - Convert LCRs from staging area into transactions
  - Execute transactions
- Explicitly through Apply handlers
  - Manually dequeue LCRs
  - Convert into transactions
  - Execute transactions
  - Example: Apply changes to Sybase database over Transparent Gateway / Database link

Apply Process (A*nnn*)



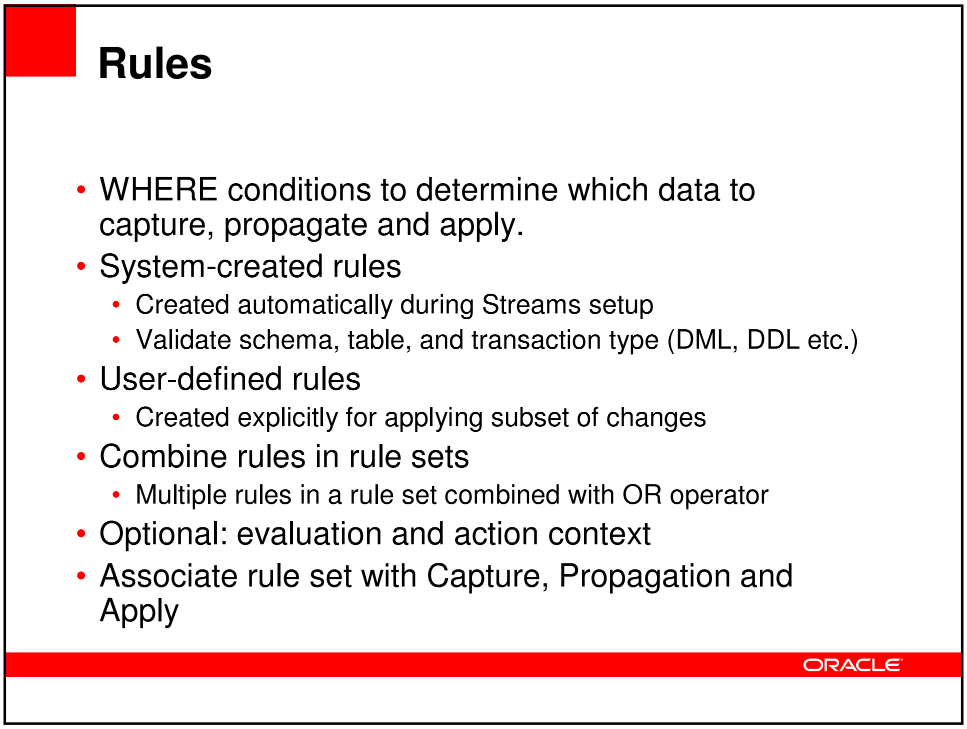
empid	job	..
100	sales	...
510	*coding*	*...*

- Runs on Apply database
- Assembles transactions from LCRs
- Parallelize execution of transactions
  - One Reader server
  - One Coordinator process (Apply)
  - Multiple Apply servers
- Report unresolved errors in exception queue
- Apply only once in bi-directional Streams setup
- Maintains status information on LCRs
- Conflict detection and resolution



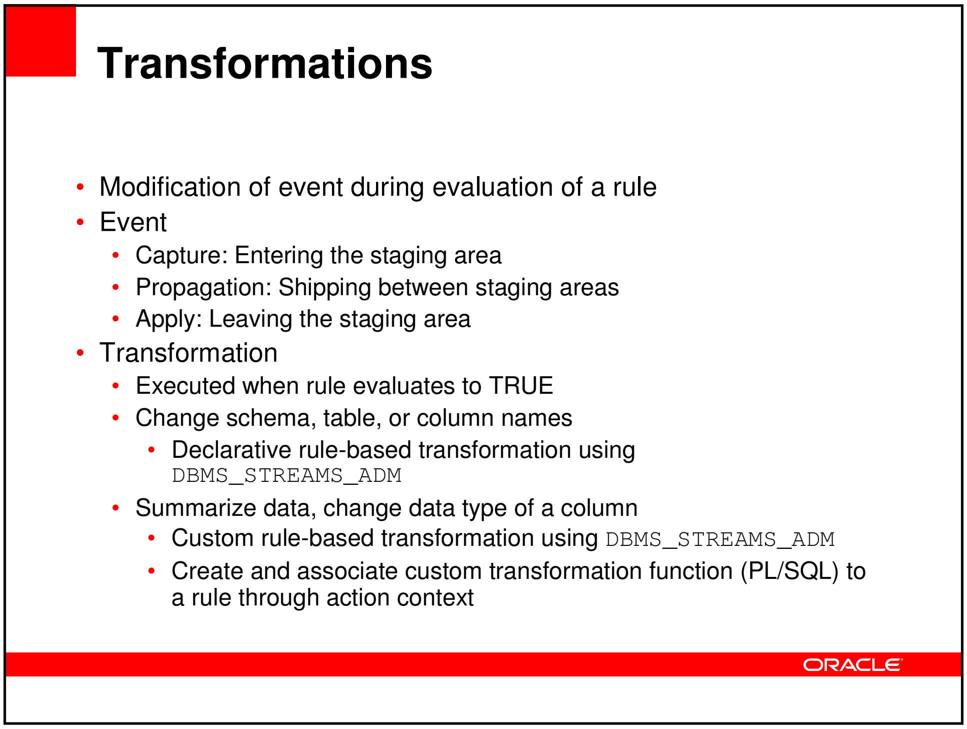
Conflicts

- Compare current values at Apply site with “old” values of changed row from Capture site
- If values matched, apply transaction
- If not matched, invoke conflict resolution method
  - Maximum (latest timestamp on a timestamp column)
  - Minimum (earliest timestamp on a timestamp column)
  - Overwrite
  - Discard
- If still not resolved, place transaction in exception queue
- DBA fixes error and re-executes error transaction
- DBA ignores error and purges error transaction



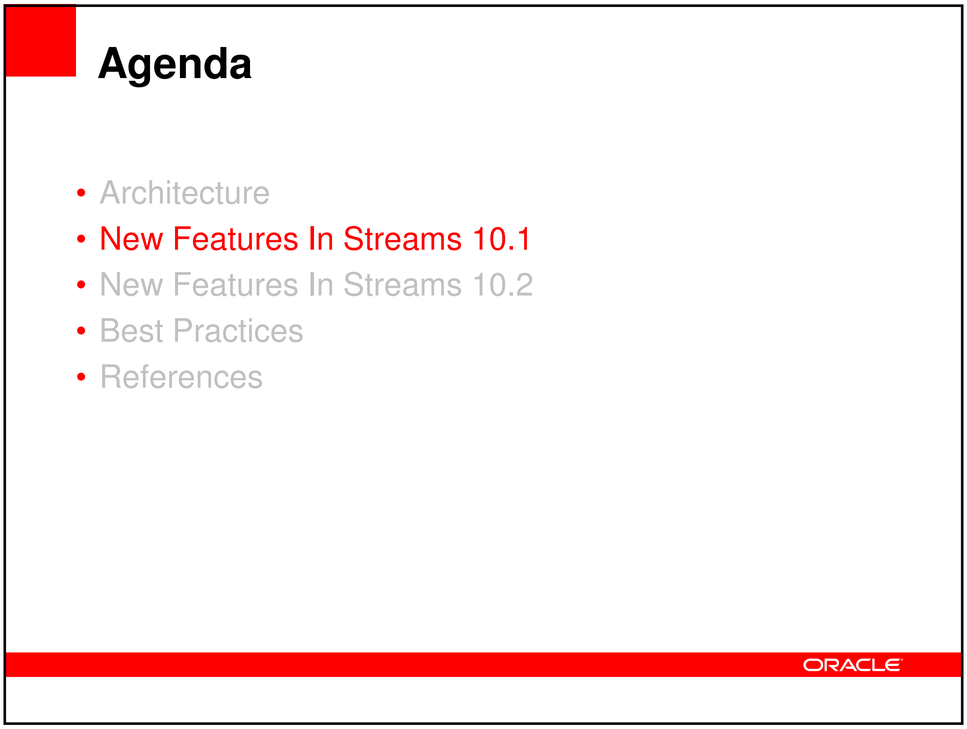
Rules

- WHERE conditions to determine which data to capture, propagate and apply.
- System-created rules
  - Created automatically during Streams setup
  - Validate schema, table, and transaction type (DML, DDL etc.)
- User-defined rules
  - Created explicitly for applying subset of changes
- Combine rules in rule sets
  - Multiple rules in a rule set combined with OR operator
- Optional: evaluation and action context
- Associate rule set with Capture, Propagation and Apply



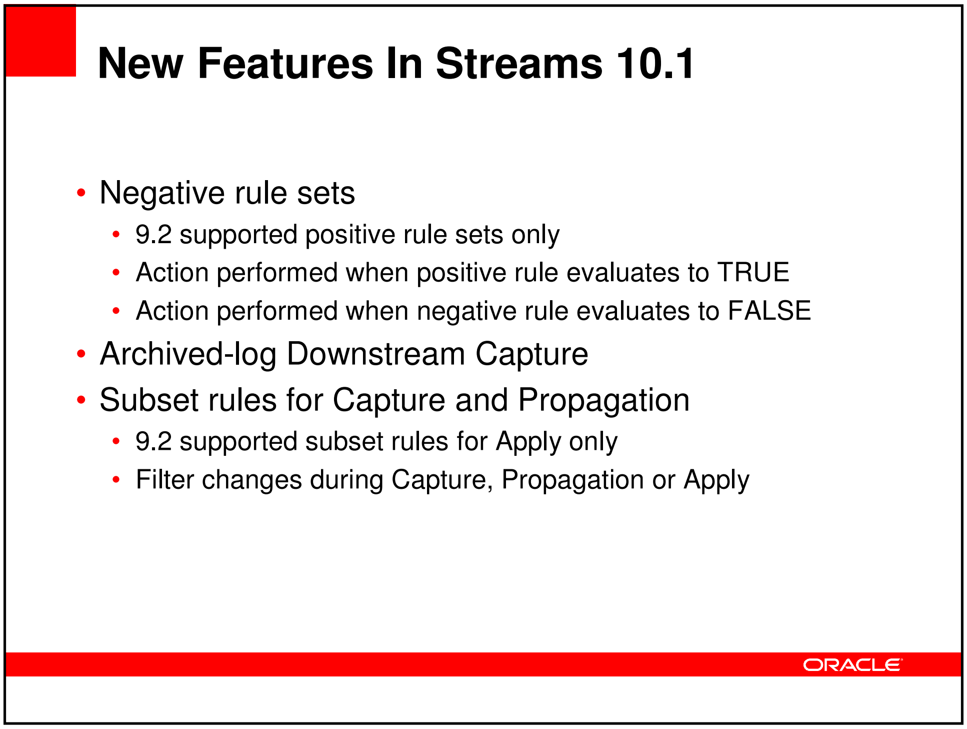
Transformations

- Modification of event during evaluation of a rule
- Event
  - Capture: Entering the staging area
  - Propagation: Shipping between staging areas
  - Apply: Leaving the staging area
- Transformation
  - Executed when rule evaluates to TRUE
  - Change schema, table, or column names
    - Declarative rule-based transformation using `DBMS_STREAMS_ADM`
  - Summarize data, change data type of a column
    - Custom rule-based transformation using `DBMS_STREAMS_ADM`
    - Create and associate custom transformation function (PL/SQL) to a rule through action context



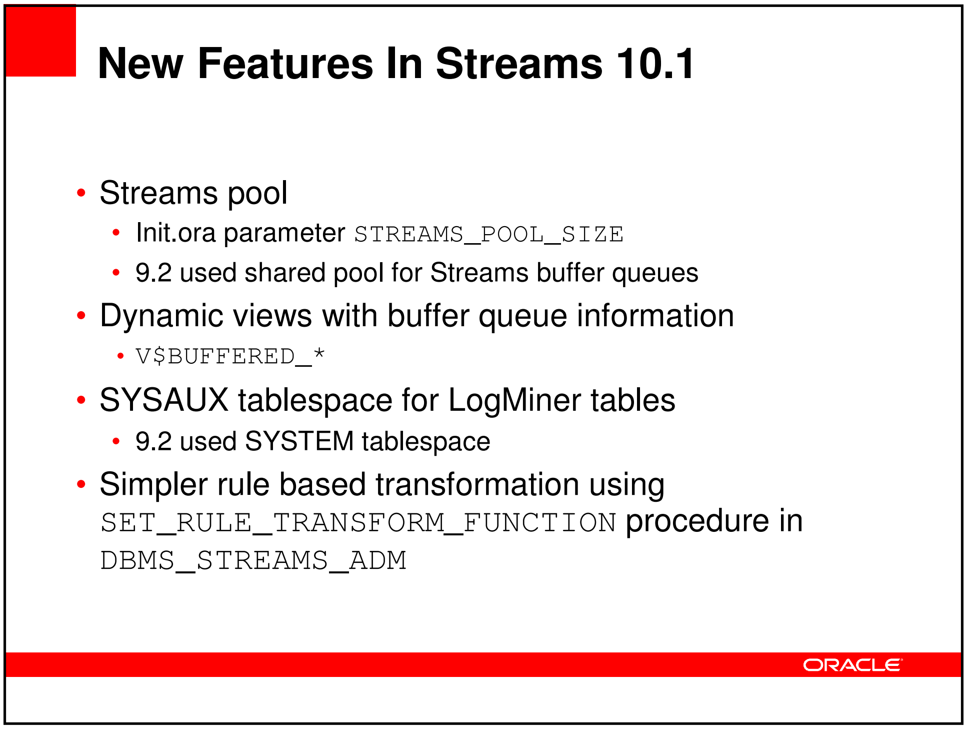
Agenda

- Architecture
- New Features In Streams 10.1
- New Features In Streams 10.2
- Best Practices
- References



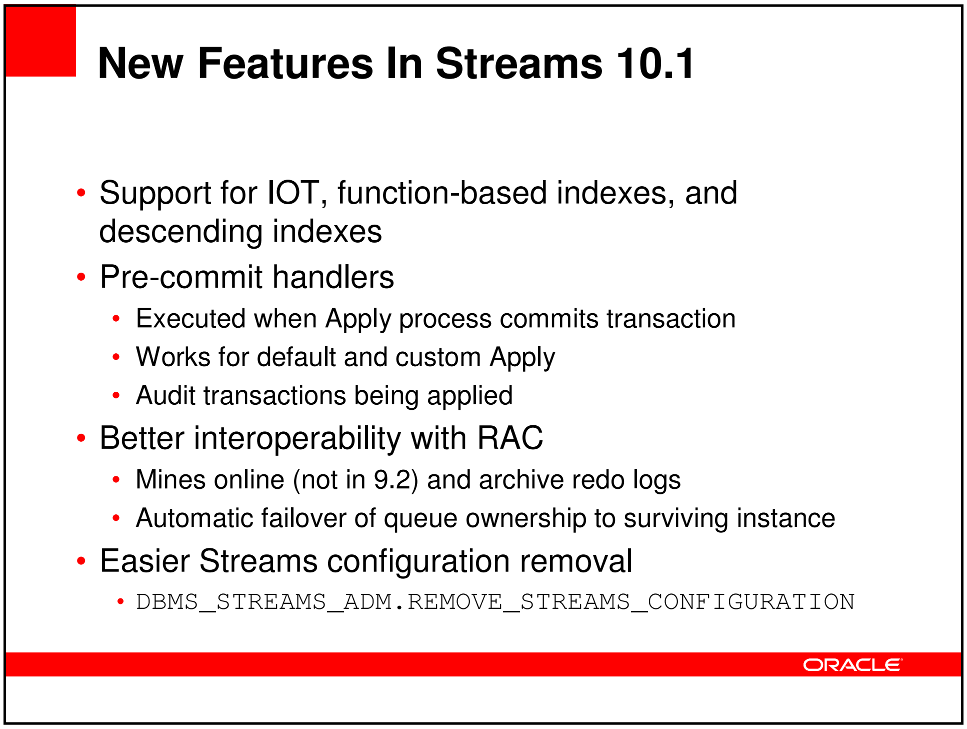
New Features In Streams 10.1

- Negative rule sets
  - 9.2 supported positive rule sets only
  - Action performed when positive rule evaluates to TRUE
  - Action performed when negative rule evaluates to FALSE
- Archived-log Downstream Capture
- Subset rules for Capture and Propagation
  - 9.2 supported subset rules for Apply only
  - Filter changes during Capture, Propagation or Apply



New Features In Streams 10.1

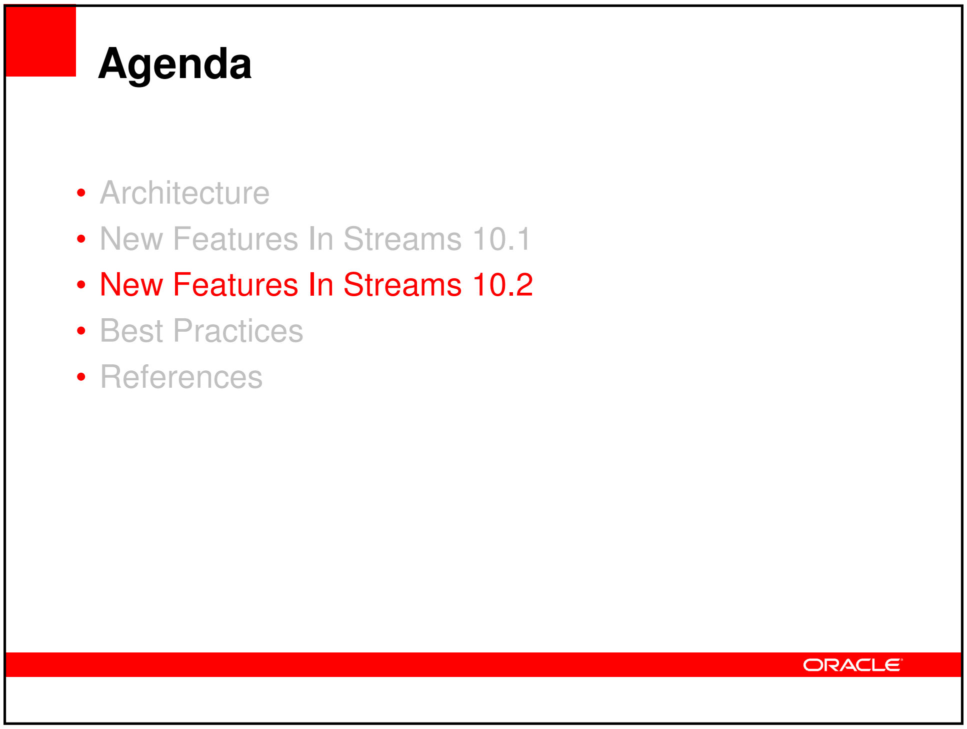
- Streams pool
  - Init.ora parameter `STREAMS_POOL_SIZE`
  - 9.2 used shared pool for Streams buffer queues
- Dynamic views with buffer queue information
  - `V$BUFFERED_*`
- SYSAUX tablespace for LogMiner tables
  - 9.2 used SYSTEM tablespace
- Simpler rule based transformation using `SET_RULE_TRANSFORM_FUNCTION` procedure in `DBMS_STREAMS_ADM`



New Features In Streams 10.1

- Support for IOT, function-based indexes, and descending indexes
- Pre-commit handlers
  - Executed when Apply process commits transaction
  - Works for default and custom Apply
  - Audit transactions being applied
- Better interoperability with RAC
  - Mines online (not in 9.2) and archive redo logs
  - Automatic failover of queue ownership to surviving instance
- Easier Streams configuration removal
  - `DBMS_STREAMS_ADM.REMOVE_STREAMS_CONFIGURATION`

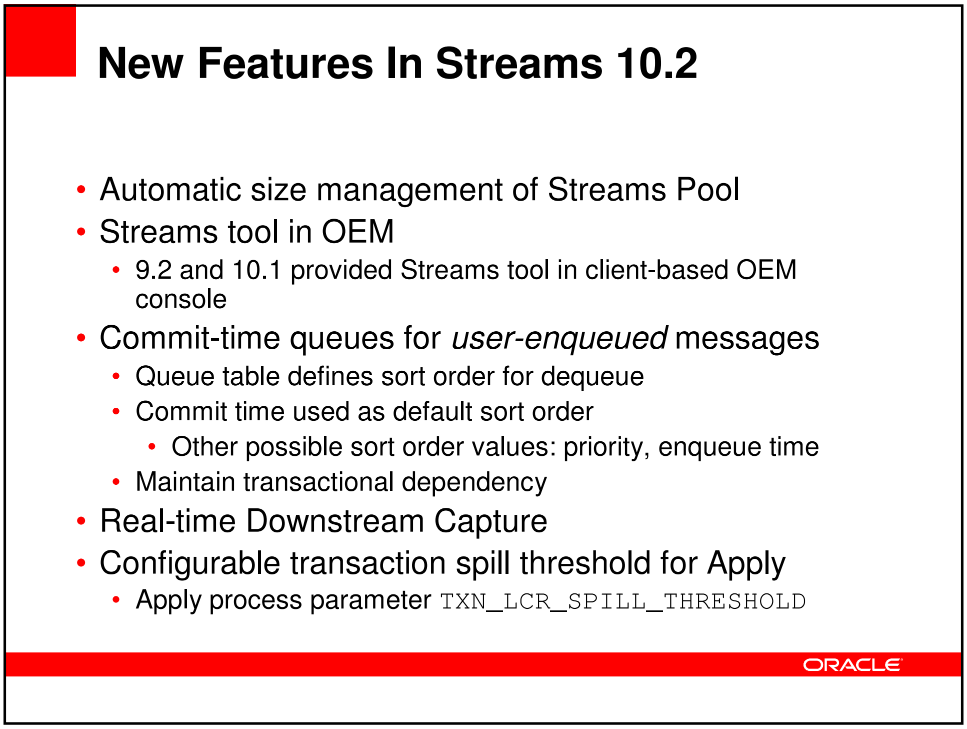
ORACLE



Agenda

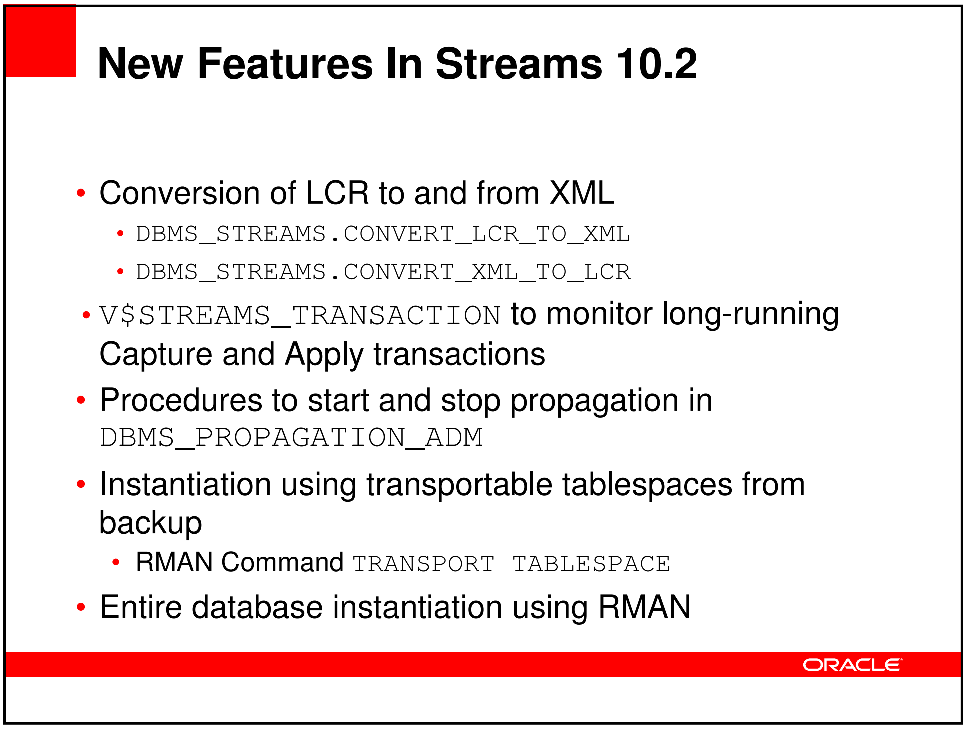
- Architecture
- New Features In Streams 10.1
- New Features In Streams 10.2
- Best Practices
- References

ORACLE



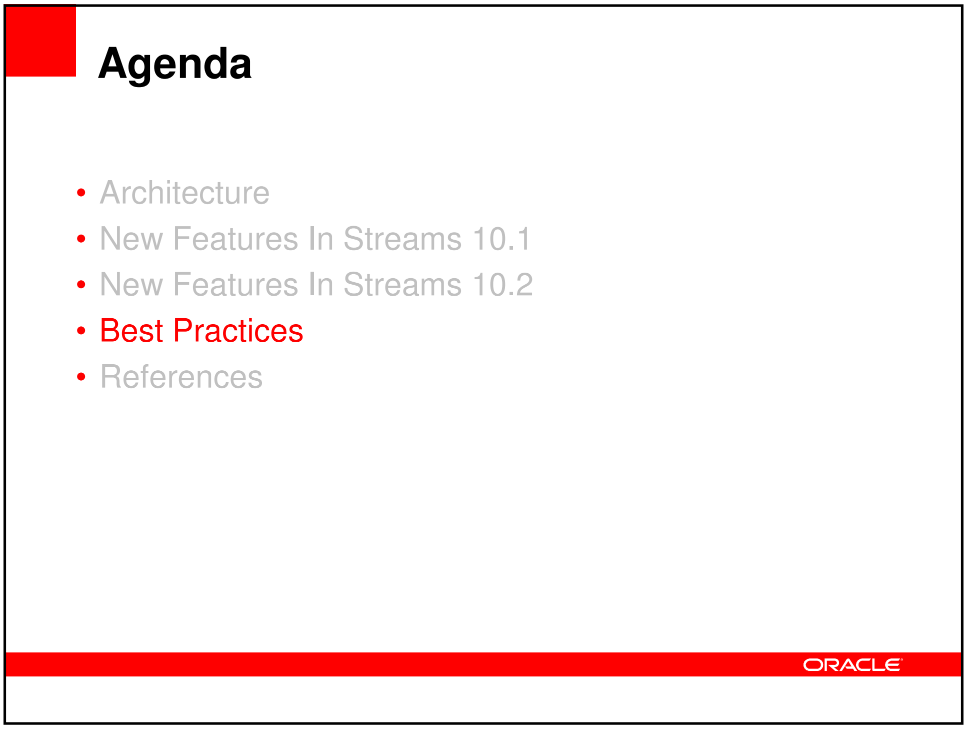
New Features In Streams 10.2

- Automatic size management of Streams Pool
- Streams tool in OEM
  - 9.2 and 10.1 provided Streams tool in client-based OEM console
- Commit-time queues for *user-enqueued* messages
  - Queue table defines sort order for dequeue
  - Commit time used as default sort order
    - Other possible sort order values: priority, enqueue time
  - Maintain transactional dependency
- Real-time Downstream Capture
- Configurable transaction spill threshold for Apply
  - Apply process parameter `TXN_LCR_SPILL_THRESHOLD`



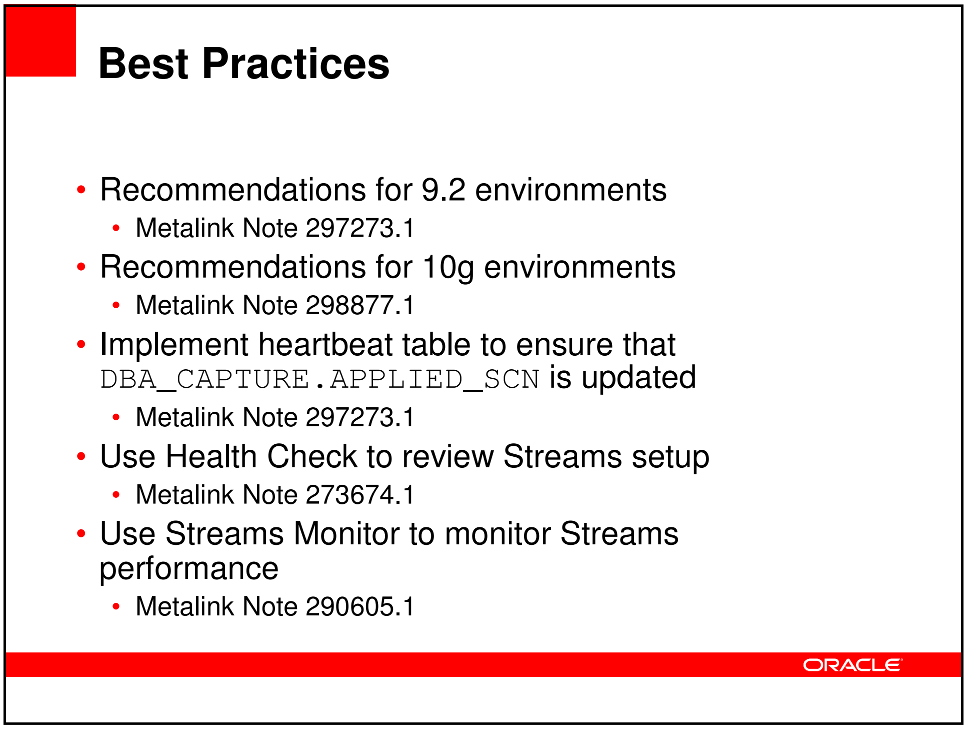
New Features In Streams 10.2

- Conversion of LCR to and from XML
  - `DBMS_STREAMS.CONVERT_LCR_TO_XML`
  - `DBMS_STREAMS.CONVERT_XML_TO_LCR`
- `V$STREAMS_TRANSACTION` to monitor long-running Capture and Apply transactions
- Procedures to start and stop propagation in `DBMS_PROPAGATION_ADM`
- Instantiation using transportable tablespaces from backup
  - RMAN Command `TRANSPORT TABLESPACE`
- Entire database instantiation using RMAN



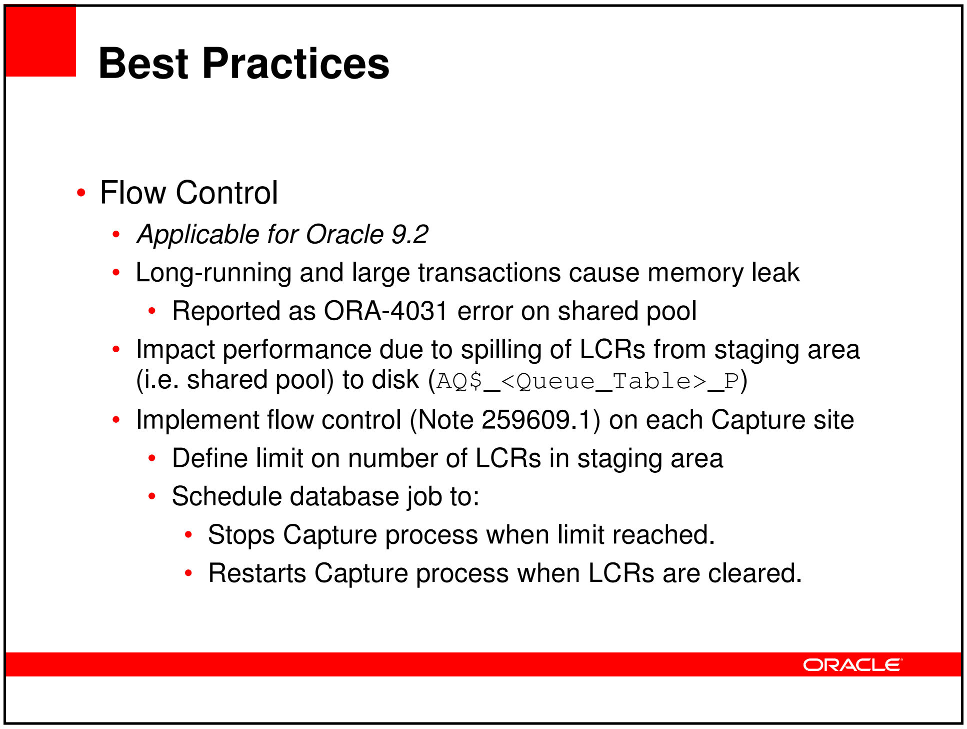
Agenda

- Architecture
- New Features In Streams 10.1
- New Features In Streams 10.2
- Best Practices
- References



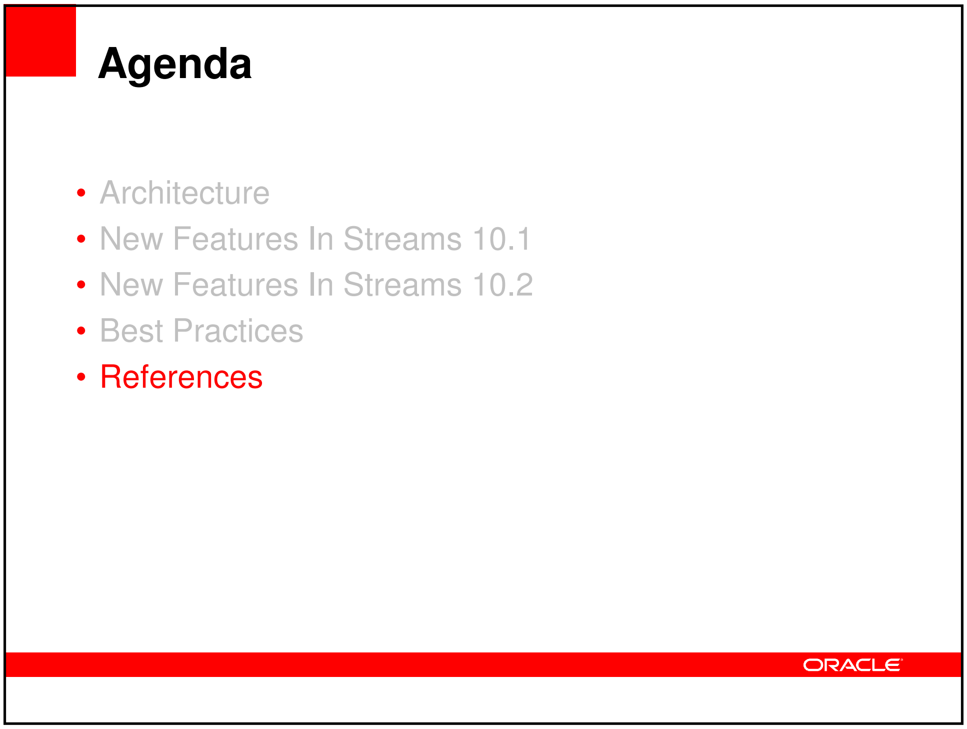
Best Practices

- Recommendations for 9.2 environments
  - Metalink Note 297273.1
- Recommendations for 10g environments
  - Metalink Note 298877.1
- Implement heartbeat table to ensure that `DBA_CAPTURE.APPLIED_SCN` is updated
  - Metalink Note 297273.1
- Use Health Check to review Streams setup
  - Metalink Note 273674.1
- Use Streams Monitor to monitor Streams performance
  - Metalink Note 290605.1



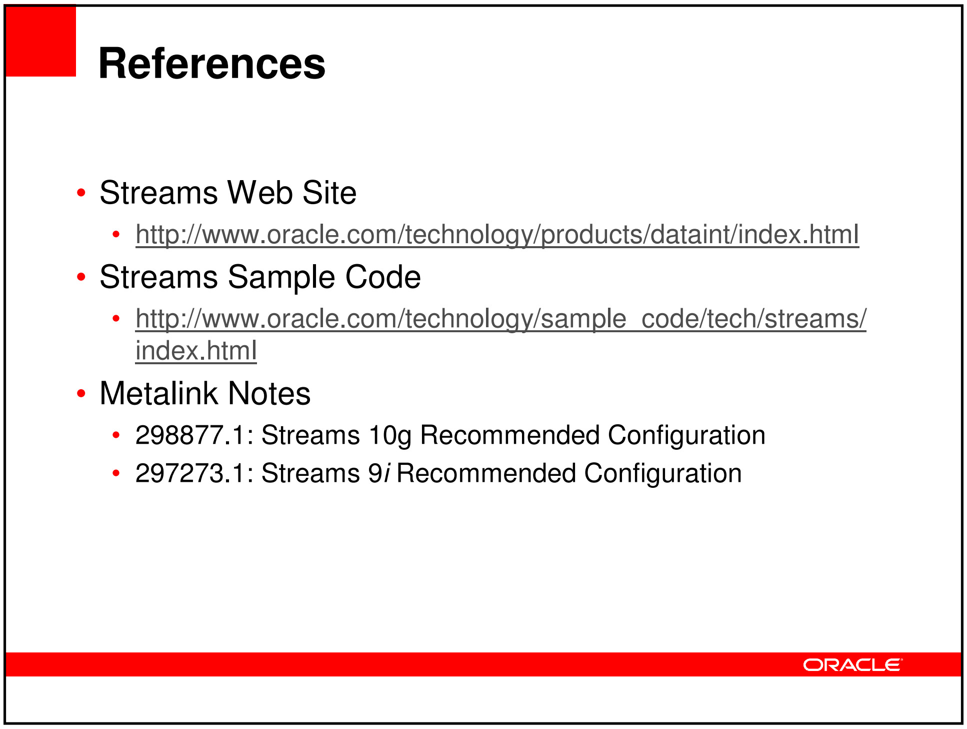
Best Practices

- Flow Control
  - *Applicable for Oracle 9.2*
  - Long-running and large transactions cause memory leak
    - Reported as ORA-4031 error on shared pool
  - Impact performance due to spilling of LCRs from staging area (i.e. shared pool) to disk (`AQ$_<Queue_Table>_P`)
  - Implement flow control (Note 259609.1) on each Capture site
    - Define limit on number of LCRs in staging area
    - Schedule database job to:
      - Stops Capture process when limit reached.
      - Restarts Capture process when LCRs are cleared.



Agenda

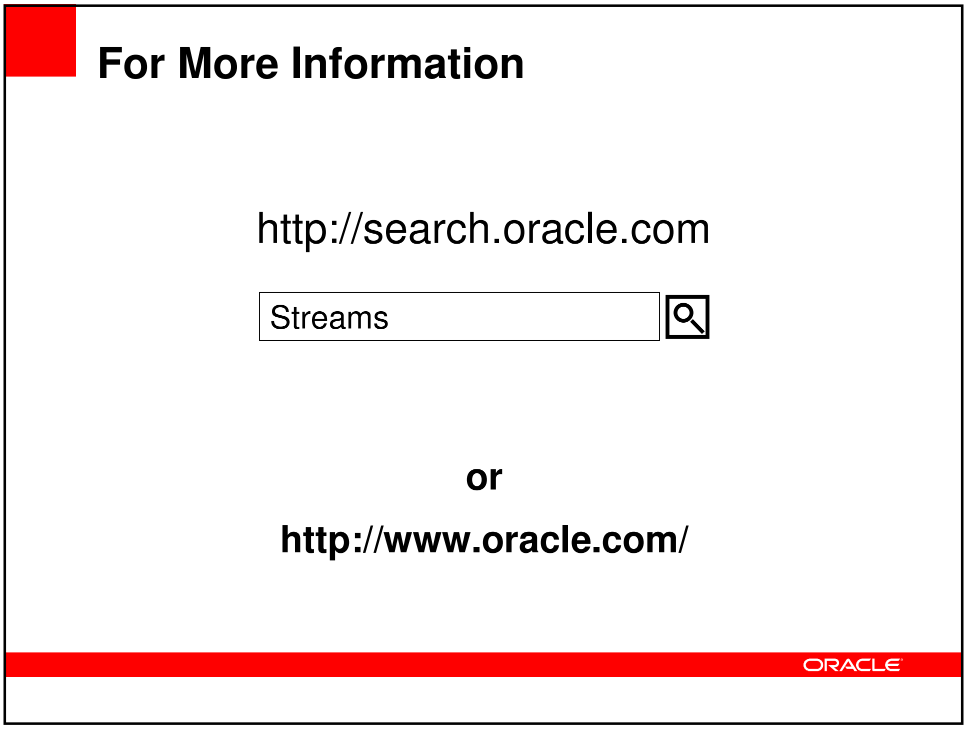
- Architecture
- New Features In Streams 10.1
- New Features In Streams 10.2
- Best Practices
- References



References

- Streams Web Site
  - http://www.oracle.com/technology/products/dataint/index.html
- Streams Sample Code
  - http://www.oracle.com/technology/sample_code/tech/streams/index.html
- Metalink Notes
  - 298877.1: Streams 10g Recommended Configuration
  - 297273.1: Streams 9*i* Recommended Configuration

ORACLE



For More Information

http://search.oracle.com

Streams 🔍

or

http://www.oracle.com/

ORACLE

ORACLE®