ORACLE®

ORACLE®

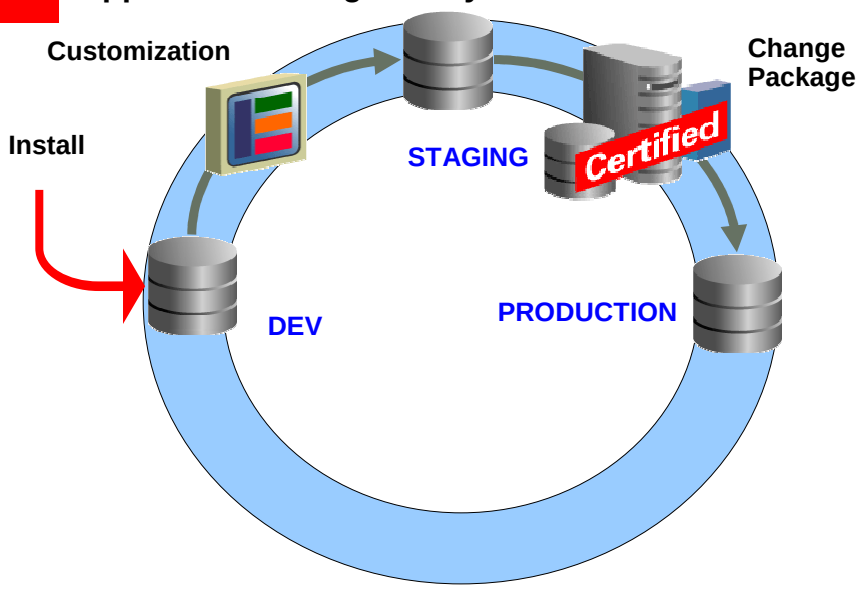
Application Change Management and Data Masking

Jagan R. Athreya (jagan.athreya@oracle.com)
Director of Database Manageability
Oracle Corporation

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.
The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

ORACLE

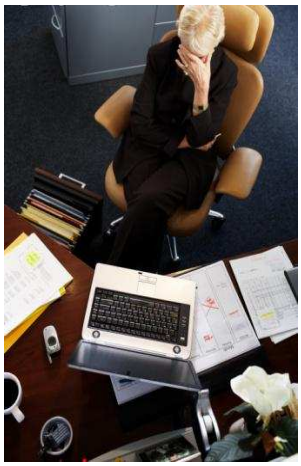
Application Change Lifecycle



ORACLE

Change is constant

- **Production Compliance**:
  - How to avoid downtime due to out-of-band changes to production databases
- **Application Upgrades:**
  - How to accelerate application upgrade cycles through faster upgrade and testing of application modules
- **Environment Synchronization:**
  - How to keep application environments in sync to ensure consistent performance in test, development, staging & production
- **Custom Development:**
  - How to improve application developer productivity by rapidly synchronizing changes in a distributed development environment.

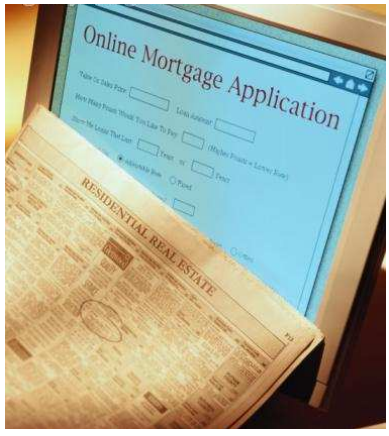


ORACLE

Cost of Application Upgrades

- Research study on application upgrades
  - $1,839 per business user
  - 1 man-week of labor per business user
- Company with 5000 business users will spend
  - $9.2 million
  - 20.7 man-years

  to perform an upgrade



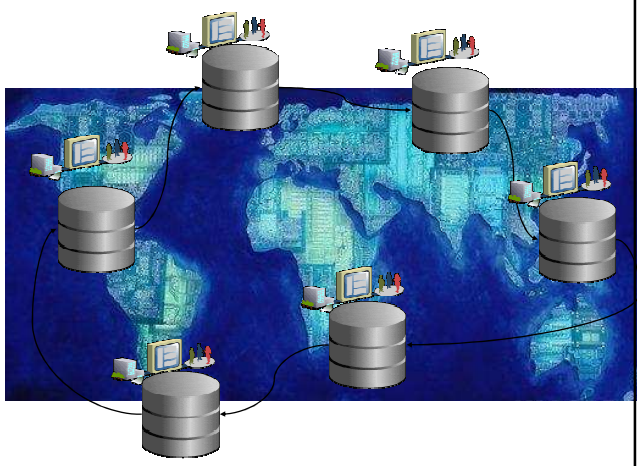
ORACLE

Change Management

ORACLE

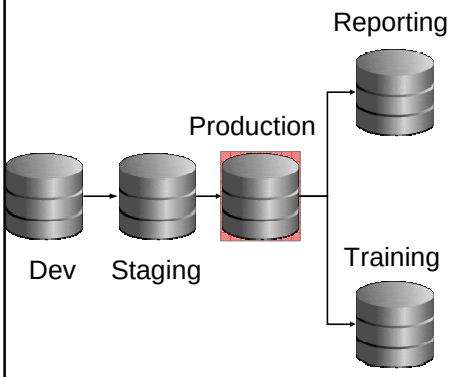
Automation of Developer Changes

- Developers make schema changes all the time
- Changes to schema need to be promoted to all development teams rapidly



ORACLE

Promoting Development Changes to Production



- Tracking changes to production databases
- Promoting changes from Testing to Staging to Production
- Transferring schema & other database changes to training, reporting environments

ORACLE

Managing Application Upgrades

Customization:
Invoice Check Printing Module

Dependencies:

- **PO_VENDORS table**
- **AP_CHECKS_PKG package & package body**

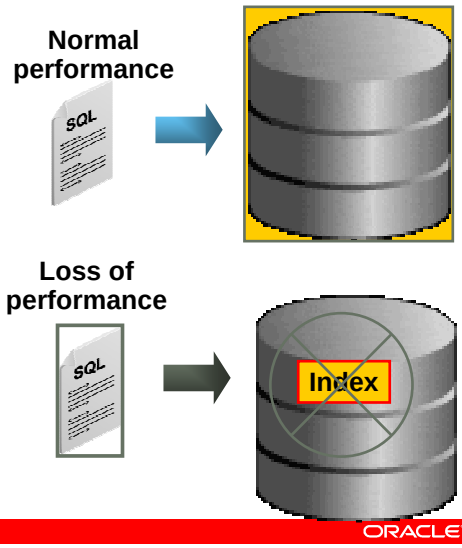
Steps

1. **Create Baseline based on dependencies**
2. **Upgrade application (EBusiness Suite 11.5.09 to 11.5.10)**
3. **Compare upgraded application schema with baseline**
4. **Identify changes**
5. Upgrade customization modules accordingly

ORACLE

Maintaining Production Compliance

- Identify out-of-band changes to production database
- ***Best practice***: weekly comparison report of **current** production against gold master baseline of production schema



ORACLE

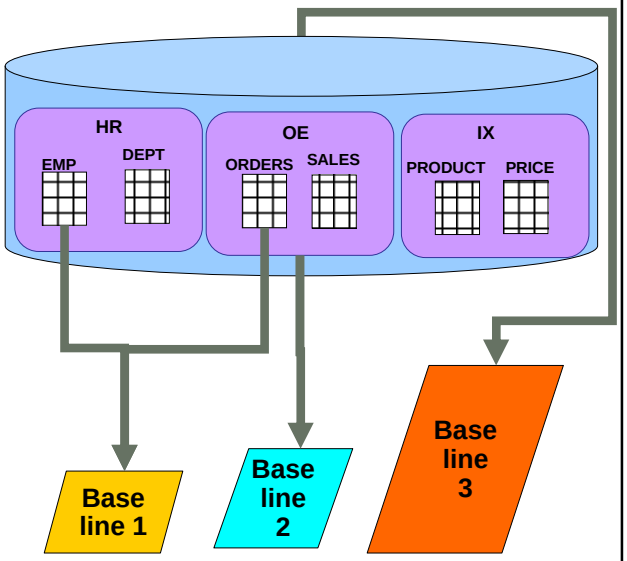
Change Management Concepts

- Source
  - Database or Baseline from where change is captured or propagated
- Target
  - Database to which change is to be Synchronized
- Baseline
  - Captured snapshot of various object definitions in a database
- Compare
  - Schemas can be compared between two sources.
- Synchronize
  - Schemas changes can be propagated from source to target
- Data Copy
  - Data can be propagated or preserved in source or target

ORACLE

Capture Application Baseline

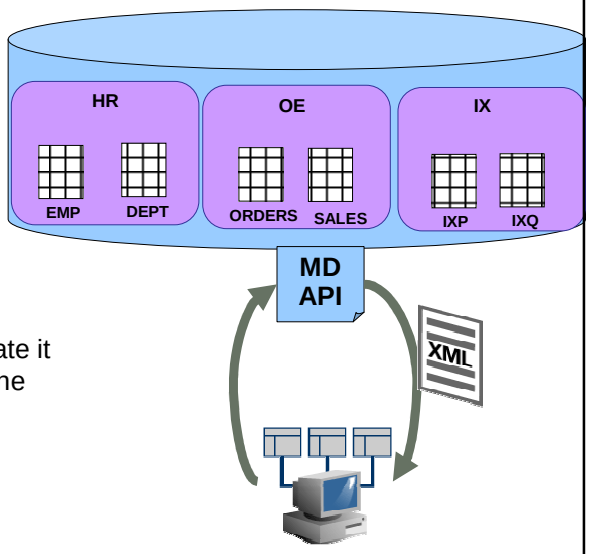
- Capture information on all schema objects for an application
  - Tables, Views, Indexes
  - Procedures, Packages, Triggers
  - Users, init.ora
- Capture scope
  - Database
  - Schema
  - Objects
- Version each baseline



ORACLE

Capture Baseline Workflow

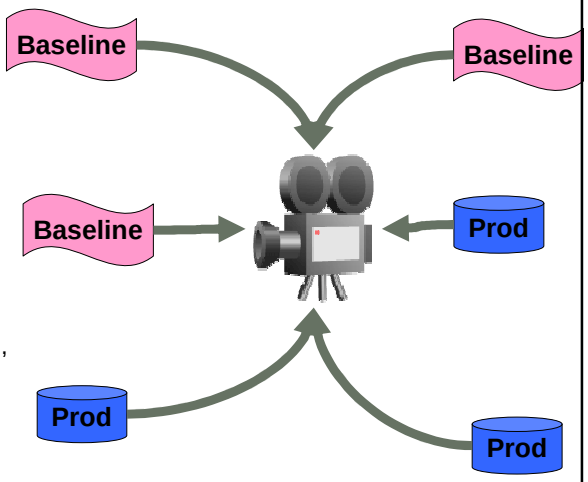
- Figure out the list of objects in the capture scope
- Use Meta Data API to capture XML representations.
- Store XML in EM repository and associate it to the captured baseline



ORACLE

Compare Baselines or Databases

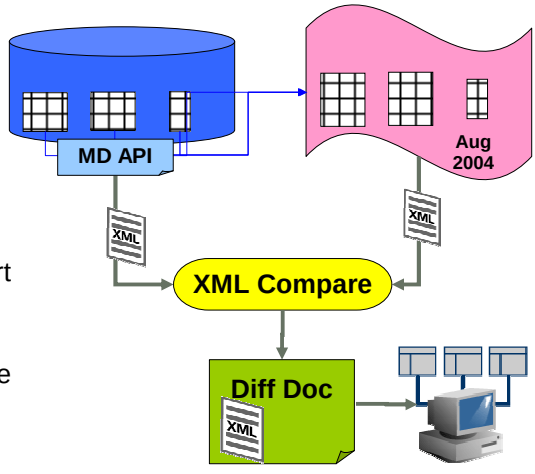
- Compare Modes
  - Baselines to Baselines
  - Baselines to Database
  - Database to Database
- Compare Scope
  - Schemas
  - Object types (Tables, Index etc)
  - Non Schema Objects (Users, Grants, Profiles etc)
  - Storage Parameters (Table spaces, extent size etc)



ORACLE

Comparison Workflow

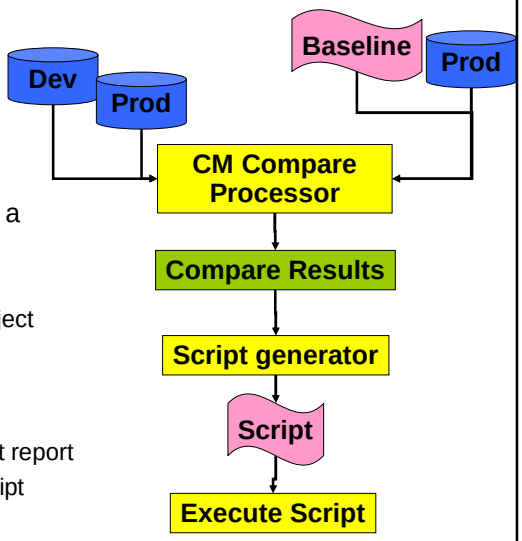
- Use existing baseline or capture a temporary baseline
- For each object do an XML comparison to generate a XML difference document
- Generate a comparison report from XML difference document
- On subsequent compares use last DDL time to limit comparisons to objects with changes



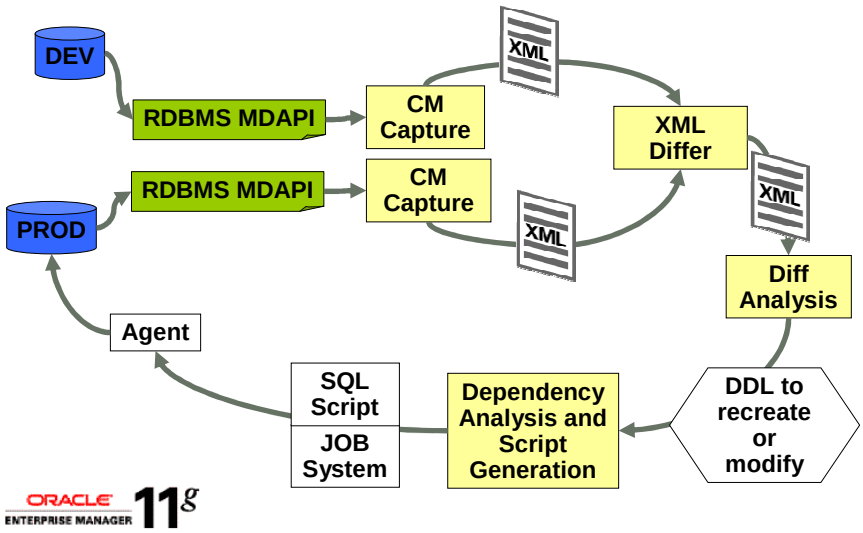
ORACLE

Synchronization

- Synchronization modes
  - Unattended Synchronize
  - Interactive Synchronize
- Synchronize source can be baseline or database
- Synchronize target is always a database
- Interactive Mode
  - Specify scope (schemas, object types to sync)
  - View Comparison results
  - Exclude objects from sync
  - Generate a script and impact report
  - Execute Synchronization script



Change Manager Sync Processing



Identify schema changes: Before and Now

Scenario: Compare test and production schema

Before

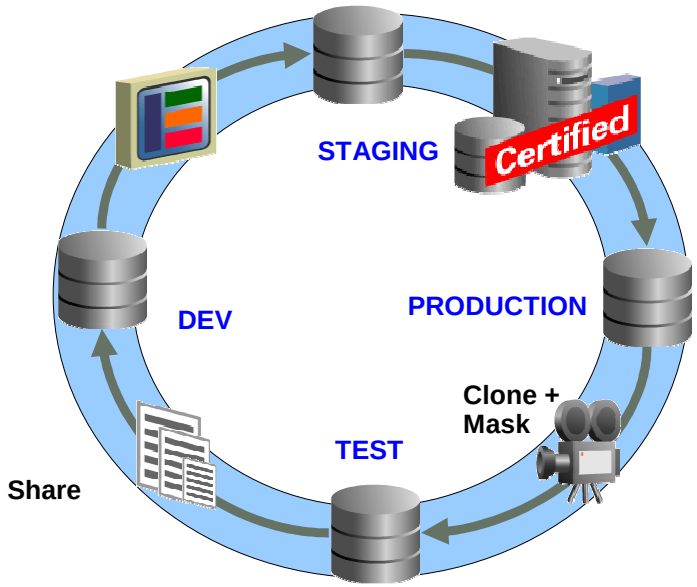
1. Identify databases, test and production
2. Get access to application schema
3. Write script to get table column parameters
4. Run script on first database
5. Run script on second database
6. Visually compare results
7. Write new script for indexes
8. Re-run script
9. Repeat for all other database objects
10. Generate a report listing changes
11. Write script to apply changes
12. Apply changes
13. Repeat above steps to verify changes
14. Re-run this for training and production
15. Re-run this for development and production

Change Management

1. Run Dictionary Comparison job
2. Identify all schema and initialization parameters differences in report.

ORACLE

Application Change Lifecycle



ORACLE

Business Drivers for Data sharing

Application Testing

- Offshore or in-house application development
- Offshore or in-house software QA

Data sharing

- Claims processing
- Offline reporting
- Data archival and retrieval
- Marketing analysis of customer data

- California Database Security Breach Notification Act
- Sarbanes Oxley Act of 2002
- US HIPAA Act
- US Graham-Leach-Bliley Financial Services Modernization Act
- EU Data Protection Directive

ORACLE

Data Masking

ORACLE

Data masking concepts

What

- The act of anonymizing customer, financial, or company confidential data to create new, legible data which retains the data's properties, such as its width, type, and format.

LAST_NAME	SSN	SALARY
AGUILAR	203-33-3234	40,000
BENSON	323-22-2943	60,000
D'SOUZA	989-22-2403	80,000
FIORANO	093-44-3823	45,000

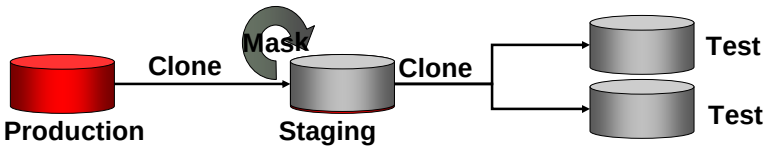
Why

- To protect confidential data in test environments when the data is used by developers or offshore vendors
- When customer data is shared with 3rd parties without revealing personally identifiable information

LAST_NAME	SSN	SALARY
ANSKEKSL	111–23-1111	40,000
BKJHHEIEDK	111-34-1345	60,000
KDDEHLHESA	111-97-2749	80,000
FPENZXIEK	111-49-3849	45,000

ORACLE

Data Masking



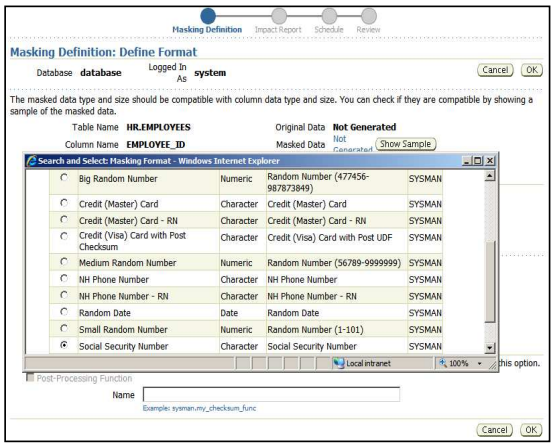
Major features

- Automatic database referential integrity when masking primary keys
  - Implicit – database enforced
  - Explicit – application enforced
- Data mask format library
- View sample data before masking
- Application masking templates
- Define once; execute multiple times

ORACLE

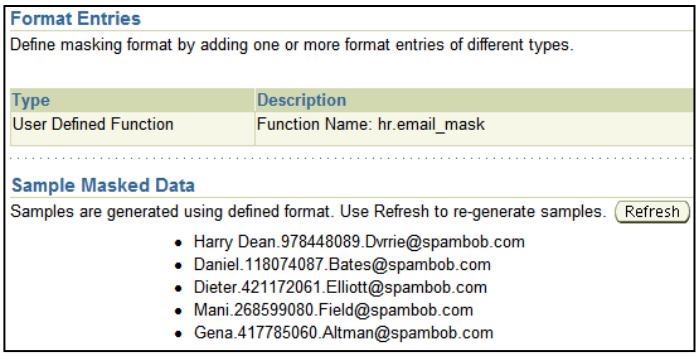
Format Libraries

- Mask Primitives
  - Random Number
  - Random String
  - Random Date within range
  - Shuffle
  - Sub string of original value
  - Table Column
- User Defined Function
  - National Identifiers
  - Social Security Numbers
  - Credit Card Numbers



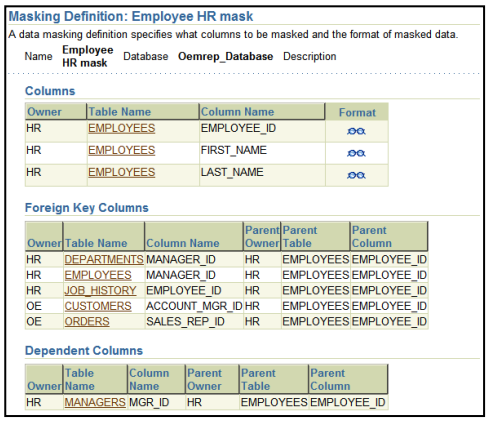
User-defined mask formats

Email notification testing



Masking Definitions

- Associates formats with database
  - Maps formats to table columns being masked
  - Defines dependent columns
  - Associated Database target
- Automatically identifies Foreign key relationships
- Can specify undeclared constraints as related columns
- Import-from or export-to XML
- "Create like" to apply to similar databases



Masking Definition: Employee HR mask

A data masking definition specifies what columns to be masked and the format of masked data.

Name **Employee HR mask** Database **Oemrep_Database** Description

Columns

Owner	Table Name	Column Name	Format
HR	EMPLOYEES	EMPLOYEE_ID	
HR	EMPLOYEES	FIRST_NAME	
HR	EMPLOYEES	LAST_NAME	

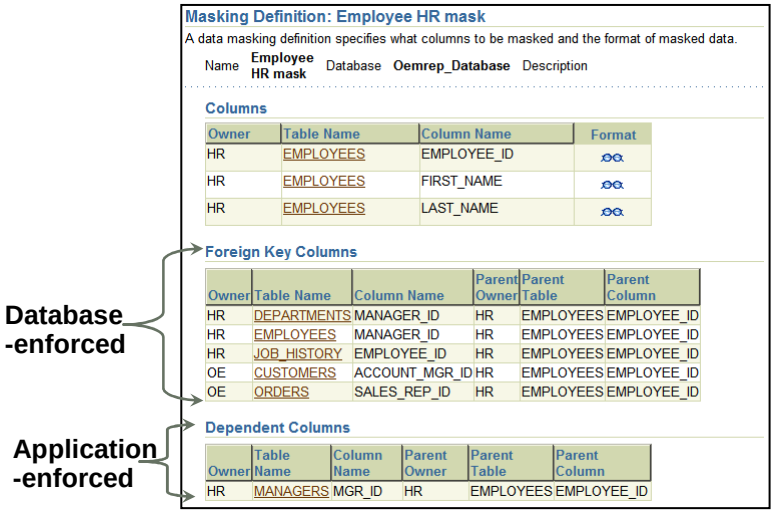
Foreign Key Columns

Owner	Table Name	Column Name	Parent Owner	Parent Table	Parent Column
HR	DEPARTMENTS	MANAGER_ID	HR	EMPLOYEES	EMPLOYEE_ID
HR	EMPLOYEES	MANAGER_ID	HR	EMPLOYEES	EMPLOYEE_ID
HR	JOB_HISTORY	EMPLOYEE_ID	HR	EMPLOYEES	EMPLOYEE_ID
OE	CUSTOMERS	ACCOUNT_MGR_ID	HR	EMPLOYEES	EMPLOYEE_ID
OE	ORDERS	SALES_REP_ID	HR	EMPLOYEES	EMPLOYEE_ID

Dependent Columns

Owner	Table Name	Column Name	Parent Owner	Parent Table	Parent Column
HR	MANAGERS	MGR_ID	HR	EMPLOYEES	EMPLOYEE_ID

Referential Integrity Enforcement



Masking Definition: Employee HR mask

A data masking definition specifies what columns to be masked and the format of masked data.

Name **Employee HR mask** Database **Oemrep_Database** Description

Columns

Owner	Table Name	Column Name	Format
HR	EMPLOYEES	EMPLOYEE_ID	
HR	EMPLOYEES	FIRST_NAME	
HR	EMPLOYEES	LAST_NAME	

Foreign Key Columns

Owner	Table Name	Column Name	Parent Owner	Parent Table	Parent Column
HR	DEPARTMENTS	MANAGER_ID	HR	EMPLOYEES	EMPLOYEE_ID
HR	EMPLOYEES	MANAGER_ID	HR	EMPLOYEES	EMPLOYEE_ID
HR	JOB_HISTORY	EMPLOYEE_ID	HR	EMPLOYEES	EMPLOYEE_ID
OE	CUSTOMERS	ACCOUNT_MGR_ID	HR	EMPLOYEES	EMPLOYEE_ID
OE	ORDERS	SALES_REP_ID	HR	EMPLOYEES	EMPLOYEE_ID

Dependent Columns

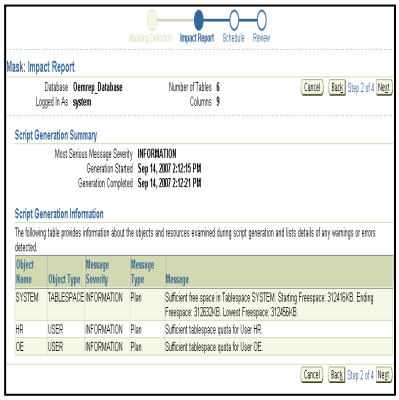
Owner	Table Name	Column Name	Parent Owner	Parent Table	Parent Column
HR	MANAGERS	MGR_ID	HR	EMPLOYEES	EMPLOYEE_ID

Database -enforced

Application -enforced

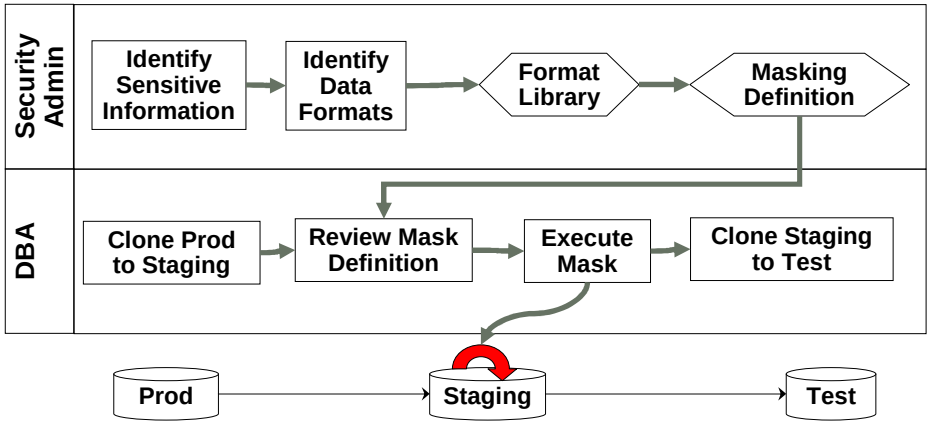
Pre-Masking Validation

- Ensure uniqueness can be maintained
- Ensure formats match column data types
- Check Space availability
- Warn about Check Constraints
- Check presence of default Partitions



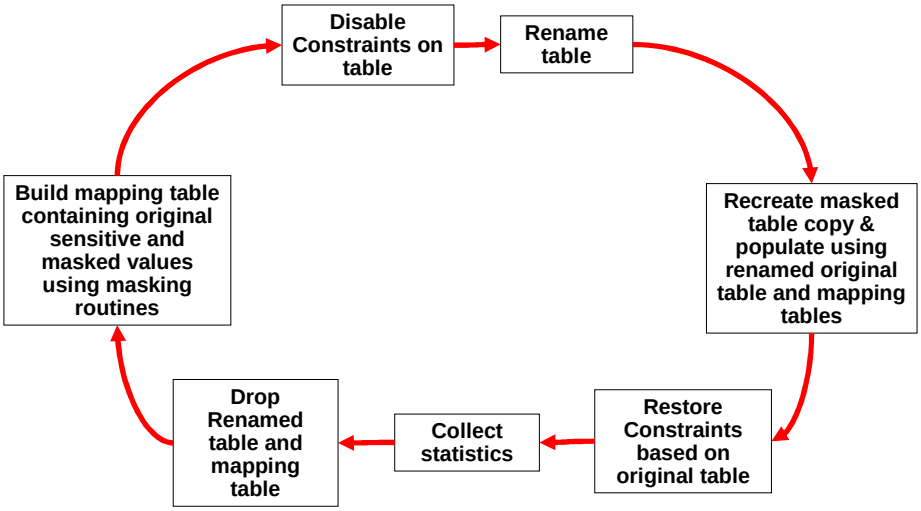
ORACLE

Masking Workflow



ORACLE

Data Masking Internals



ORACLE

Performance

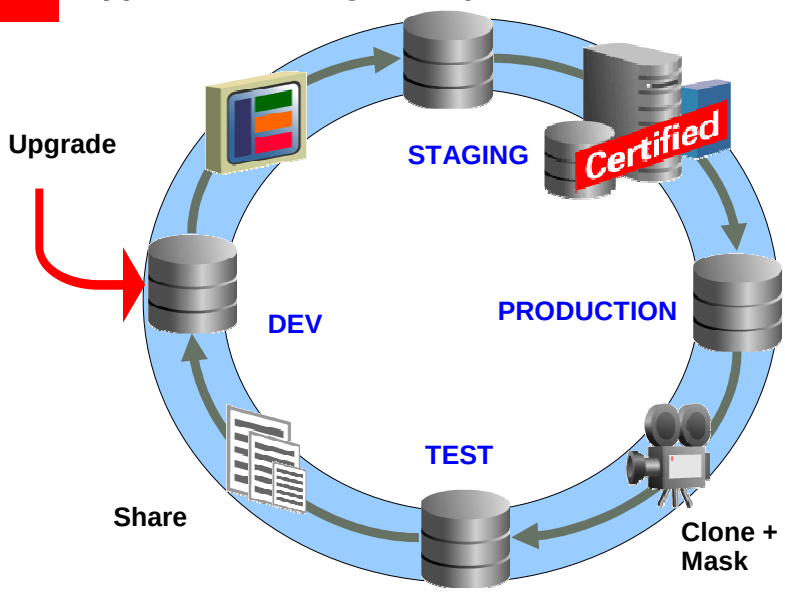
- Optimizations
  - SQL Parallelism for tables > 1 million rows
  - Statistics collection before & after masking
  - CTAS statement with NOLOGGING

Test results

- Linux x86 4 CPU: Single core Pentium 4 (Northwood) [D1]
- Memory: 5.7G
- Column scalability
  - 215 columns masked across 100 tables
  - 60GB Database
  - 20 minutes
- Rows scalability
  - 100 million row table, 6 columns masked
  - Random Number
  - 1.3 hours

ORACLE

Application Change Lifecycle



ORACLE

Online Application Upgrade

- Large, mission critical applications are often unavailable for tens of hours while a patch or an upgrade is installed
- Oracle Database 11*g* Release 2 introduces revolutionary new capabilities that allow online application upgrade with *uninterrupted availability* of the application
- The pre-upgrade application and the post-upgrade application can be used at the same time
- End-user sessions therefore enjoy hot rollover

ORACLE

The challenge

- The installation of the upgrade into the production database must not perturb live users of the pre-upgrade application
  - Many objects must be changed in concert. The changes must be made *in privacy*
- Transactions done by the users of the pre-upgrade application must by reflected in the post-upgrade application
- For hot rollover, we also need the *reverse* of this:
  - Transactions done by the users of the *post*-upgrade application must by reflected in the *pre*-upgrade application

ORACLE

The solution: edition-based redefinition

- 11.2 brings these revolutionary new features: the *edition*, the *editioning view*, and the *crossedition trigger*
  - Code changes are installed in the privacy of a new *edition*.
  - Data changes are made safely by writing only to new columns or new tables not seen by the old edition
    - An *editioning view* exposes a different projection of a table into each edition to allow each to see just its own columns
    - A *crossedition trigger* propagates data changes made by the old edition into the new edition's columns, or (in hot-rollover) vice-versa

ORACLE

The solution: editions

- 11.2 introduces the new nonschema object type, *edition* – objects are versioned within an edition
- A database must have at least one edition
- You create a new edition as the child of an existing edition – and an edition can't have more than one child
- A database session specifies which edition to use

  (of course, the database has a default edition)

ORACLE

Editioning views

- Think of it like a noneditionable *physical* table body with an editionable *logical* table spec
- Of course, you can't have more than one editioning view for a particular table in a particular edition
- Application code should refer only to the logical world
- You can create table-style triggers (*before* or *after statement* or *each row*) on an editioning view using the "logical" column names
- A SQL optimizer hint can request an index on the physical table by specifying the "logical" column names

ORACLE

The solution: crossedition triggers

- The new crossedition trigger has special firing rules
- You create crossedition triggers in the *Post_Upgrade* edition
  - The paradigm is: don't interfere with the *Pre_Upgrade* edition
- The firing rules rules assume that
  - Pre-upgrade columns are changed – by ordinary application code – only by sessions using the *Pre_Upgrade* edition
  - Post-upgrade columns are changed only by sessions using the *Post_Upgrade* edition

ORACLE

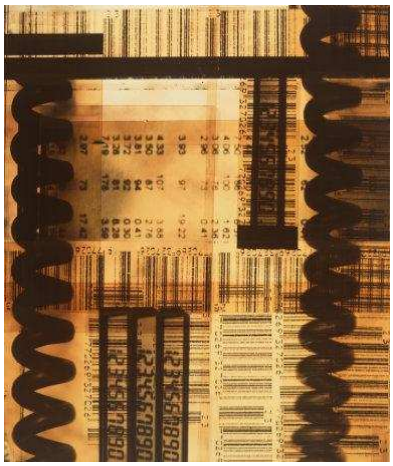
Change Management

- Application Managers: Manage application upgrades effectively and efficiently
- DBAs: Helps with audit, compliance and management reporting
- Developers: Eliminate errors/data loss and down time when making changes

ORACLE

Data Masking

- Helps sharing of production data in compliance with data privacy policies.
- Delivers uniform application of mask formats across all enterprise data
- Increase DBA productivity by automating the discovery and masking of sensitive data.



ORACLE

Online Application Upgrade

- Large, mission critical applications can now be continuously available while a patch or an upgrade is installed
- The pre-upgrade application and the post-upgrade application can be used at the same time
- End-user sessions therefore enjoy hot rollover
- The pre-upgrade application is retired only when no sessions any longer are using it

ORACLE


ORACLE®