Oracle®

Data Masking and Subsetting Guide

Oracle Enterprise Manager 24ai
F85906-03
November 2024

Oracle Data Masking and Subsetting Guide, Oracle Enterprise Manager 24ai

F85906-03

Primary Authors: Preethy P G, Bert Rich

Contributing Authors: Jim Garrison, Tulika Das, Dinesh Rajasekharan

Contributors: Eric Paapanen, Vipin Samar, Gopal Mulagund, John Kati, Amoghavarsha Ramappa

Contents

6 Frequently Asked Questions

Index

Preface

This preface contains the following topics:

Audience

This document provides information about how to mask and subset data for non-production usage such as development and testing using Oracle Data Masking and Subsetting Pack (DMS) of Oracle Enterprise Manager (EM) Database Plug-in. This document is intended for database administrators (DBAs), database designers, application administrators, application owners, business owners, testers, and developers.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Related Documents

For more information about some of the topics discussed in this document, see the following documents in the Oracle documentation set:

- *Oracle Database 2 Day DBA*
- *Oracle Database 2 Day + Performance Tuning Guide*
- *Oracle Database Administrator's Guide*
- *Oracle Database Concepts*
- *Oracle Database Performance Tuning Guide*
- *Oracle Database SQL Tuning Guide*
- *Oracle Database Installation Guide for Linux*

- *Enterprise Manager Command Line Interface*
- *Oracle Enterprise Manager Installation Guide*
- *Enterprise Manager Cloud Control Advanced Installation and Configuration Guide*
- Enterprise Manager Licensing Information

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.

Frequently Asked Questions

For frequently asked or common questions regarding Oracle Data Masking and Subsetting, see: **Frequently Asked Questions**

What's New in Oracle Data Masking and Subsetting

This preface contains:

Versioning of Oracle Data Masking and Subsetting

The Oracle Data Masking and Subsetting version is based on the current version of the Oracle Enterprise Manager Database Plug-in.

What's new in Oracle Data Masking and Subsetting Release 24ai

This section describes the features introduced in Oracle Data Masking and Subsetting Release 24ai.

Modernized UI for Data Discovery and Data Masking with Oracle Javascript Extension Toolkit (JET)

- **Unified Console Experience**: Easily switch between key DMS components—Overview, Data Discovery, Data Masking, and Data Subsetting—all within a single, streamlined interface.
- **Simplified Navigation**: Access features with a reorganized menu, enabling direct and intuitive navigation within each component.
- **Workflow Diagrams**: Visualize typical workflows with newly introduced, clean and easy-to-understand diagrams.
- **Enterprise-Level Overview Dashboard**: Gain insights with the all-new Data Masking and Subsetting Overview dashboard, offering metrics to effectively manage discovery and masking activities across your databases.

UI Performance Enhancements for Data Discovery and Data Masking

- **Optimized Workflow Execution**: Perform workflows without unnecessary delays by loading only the data required for the requested operation, reducing resource usage and user wait times.

- **Responsive Design**: A UI that seamlessly adapts to devices of various sizes, ensuring a consistent experience across desktops, tablets, and smartphones.
- **Faster Workflow Execution**: Enhanced flow execution times for operations like creating masking formats or definitions, leveraging Oracle JET UI and optimizations such as lazy loading.
- **Improved Filtering**: Enable faster, client-side searches by eagerly loading data, removing dependency on server-side filtering.
- **Support for Bulk Operations**: Simplify repetitive tasks with bulk operations, such as adding sensitive columns in bulk, minimizing clicks and effort.
- **Intuitive Masking Format Creation**: Simplify the creation of custom masking formats with drag-and-drop support for reorganizing format entries.

Changes in Oracle Data Masking and Subsetting Release 13.5

This section describes the features introduced in Oracle Data Masking and Subsetting Release 13.5.

Data Masking and Subsetting for Autonomous Databases

Data Masking and Subsetting functionalities are now available for Autonomous Databases in Oracle Enterprise Manager (13c Release 5 Update 10). You can use this comprehensive solution to minimize data exposure in non-production environments by discovering and masking sensitive production data.

Enhancements and Additions to Sensitive Types

Oracle Data Masking and Subsetting Pack provides a comprehensive solution to manage secure production data for test and development environments. The sensitive data is identified through sensitive data discovery, where you can define search patterns specific to your business. You can also leverage the sensitive types that are available out-of-the-box. The pre-defined types that were updated include:

- Credit Card Number
- Email ID
- IP Address
- ISBN_10
- ISBN_13
- National Insurance Number
- Phone Number
- Social Insurance Number
- Social Security Number
- Universal Product Code

Additionally, the following new types are now supported:

- Age

- Card Expiration Date
- Card Security Code
- Card Security PIN
- Date of Birth
- Ethnicity
- First Name
- Full Name
- Gender
- India Aadhaar Number
- India PAN
- Last Name
- MAC Address
- Mexico CURP Code
- Place of Birth
- Postal Code
- Race
- Religion
- Sexual Orientation

Sensitive types can also be added manually.

1

Introduction

This chapter provides an introduction to the core concepts of the Oracle Data Masking and Subsetting Pack.

> **Note:**
>
> For Oracle Data Masking and Subsetting licensing information, please refer to Oracle Database Licensing Guide.

What to Know

About DMS

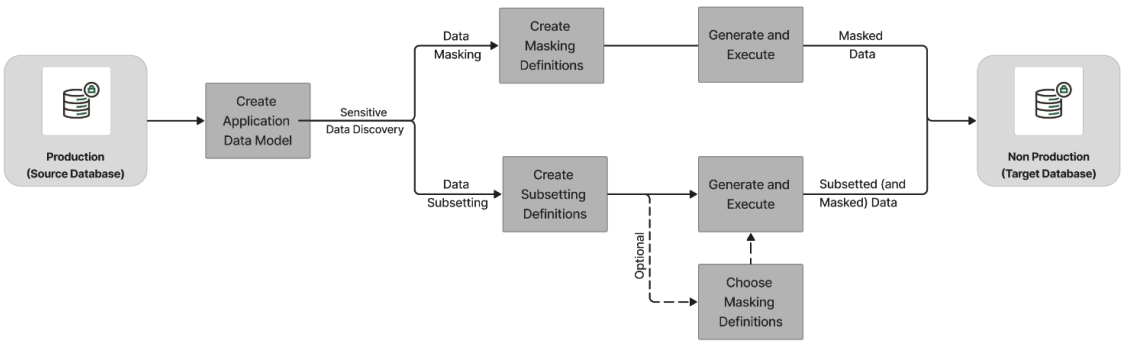
Oracle Data Masking and Subsetting unlocks the value of data without increasing risk and minimizes storage costs by enabling secure and cost-effective data provisioning for various scenarios, including test and development environments. It protects sensitive data by discovering and masking it, ensuring that only relevant data is shared. This approach speeds up regulatory compliance and maximizes data value by safely sharing realistic, functionally intact data with stakeholders. It also reduces costs by extracting and sharing only the necessary data, thereby lowering time, storage, and infrastructure expenses.

Major Components of Oracle Data Masking and Subsetting

Oracle Data Masking and Subsetting consists of the following major components:

- **Data Discovery**
- **Data Masking**
- **Data Subsetting**

Figure 1-1 Oracle Data Masking and Subsetting Overview



> **Note:**
>
> It is recommended that you understand the concepts of Oracle Data Masking and Subsetting mentioned in this chapter prior to the implementation. If you are already aware of these concepts or want to start masking and subsetting data, please refer to the chapters below this chapter:
>
> - Start with the **Prerequisites** chapter to understand the privileges, roles, and storage requirements.
> - Refer to the **Data Discovery** chapter to discover and model sensitive columns.
> - Refer to the **Data Masking** chapter to define and execute masking operation.
> - Refer to the **Data Subsetting** chapter to define and execute subsetting operation.

Why DMS?

Challenges

1. **How to Locate Sensitive Data at Short and Long Term?** Finding sensitive data across a sprawling landscape of numerous applications, databases, and environments can be a daunting task. Organizations must implement robust discovery tools and methodologies to ensure all sensitive data is accurately identified both initially and on an ongoing basis.
2. **How to Accurately Protect Sensitive Data?** Sensitive data comes in various shapes and forms, such as Credit Card Number, Ethnicity, Phone Number, Date of Birth, and more. Protecting such diverse data types necessitates sophisticated masking and techniques tailored to each specific data type to ensure comprehensive data security.
3. **Is the Protected Data Usable?** For developers, testers, applications, and more, it's crucial that protected data remains usable. Ensuring that data masking and subsetting solutions preserve the functionality and realism of the data is key to maintaining productivity and the integrity of testing and development processes.
4. **Will the Applications Continue to Work?** Developing and maintaining data protection solutions in an ever-changing IT landscape, encompassing both on-premises and cloud environments, requires continuous adaptation. Ensuring that applications continue to work seamlessly despite these changes is vital for ongoing operations and development.
5. **Limit Sensitive Data Proliferation**: Sensitive data often proliferates across various environments such as Test, Dev, QA, Training, Research, Cloud, and more. Implementing strict controls and monitoring to limit the spread of sensitive data is essential for reducing exposure and maintaining data security.
6. **Save Storage Costs**: Managing data storage efficiently in non-production environments, such as test/dev and large data warehouses, is crucial for controlling costs. By extracting and sharing only relevant data, organizations can significantly reduce their storage expenses while maintaining necessary access to data.
7. **Share What is Necessary**: When sharing data with subscribers, auditors, courts, partners, testers, developers, and more, it is essential to ensure that only necessary information is provided and sensitive information is masked. This minimizes the risk of unnecessary data exposure while complying with regulatory and operational requirements.
8. **Right to be Forgotten/Erasure (GDPR in Europe)**: Compliance with regulations such as GDPR in Europe mandates the right to be forgotten, requiring organizations to effectively

erase personal data upon request. Implementing efficient data management processes to handle such requests is critical for regulatory compliance and maintaining customer trust.

Use Cases

1. **Data Sharing:** When outsourcing or collaborating with third parties, sharing full datasets can pose a risk. Data Masking and Subsetting solution helps by removing sensitive or non-essential data so that only relevant, non-sensitive data is shared, protecting sensitive information like intellectual property, financial, and personal data.
2. **Secure Application Testing:** Developers need real data for effective testing. Data Masking and Subsetting allows them to use real-world data by obfuscating sensitive information like personal identifiers and financial details. This helps in robust testing without compromising data privacy.
3. **Analytics without compromise:** By masking sensitive information, businesses can share data securely for analytics without exposing personal data. This enables effective data-driven insights while maintaining privacy and compliance.
4. **Regulatory Compliance:** With regulations like GDPR and CCPA, organizations are required to handle sensitive data securely. Oracle's solution helps anonymize and protect sensitive information in non-production environments, aiding compliance efforts and minimizing the risk of penalties.

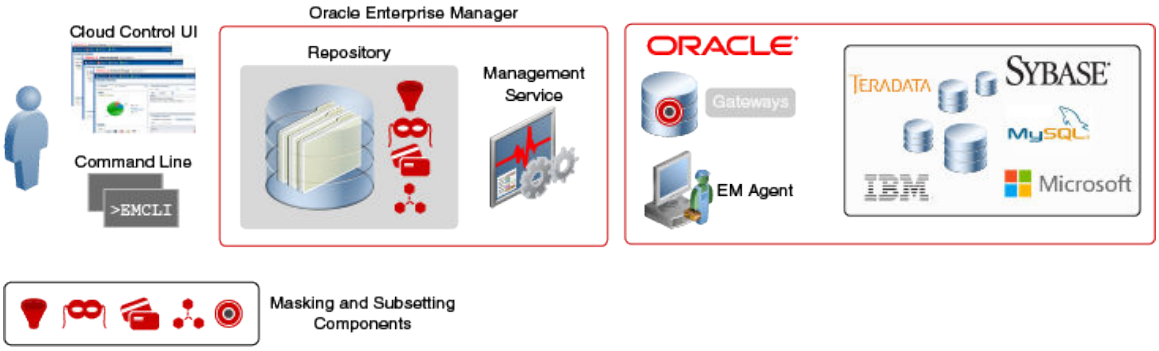
Benefits

- **Sensitive Data Protection**: Ensure the confidentiality of sensitive information, such as Personally Identifiable Information (PII) or financial data, by masking it before it's used in non-production environments, thereby safeguarding against unauthorized access.
- **Compliance Boundary Management**: Ensure enterprise-wide compliance with regulations and laws for managing sensitive data and only allowing access to non-sensitive data in testing and development environments, thereby reducing the risk of compliance violations.
- **Cost-Effective Test and Development Environments**: Reduce storage costs in test and development environments by subsetting data. By only including relevant subsets of data necessary for testing and development purposes, storage requirements are minimized without sacrificing the quality of testing.
- **Automated Sensitive Data Discovery**: Automatically identify sensitive data and its relationships within databases. This feature streamlines compliance efforts by providing a clear understanding of where sensitive data resides and how it's connected within the database.
- **Comprehensive Masking and Subsetting Capabilities**: Access a wide range of masking formats, subsetting techniques, and application templates. This empowers users to tailor data masking and subsetting strategies to their specific requirements, ensuring comprehensive data protection and optimization.
- **Flexible Deployment Options**: Mask and subset data either directly in the database or from exported files. This flexibility accommodates diverse data environments and preferences, allowing users to choose the method that best suits their infrastructure and workflows.

Architecture

Oracle Data Masking and Subsetting is part of the Oracle Enterprise Manager infrastructure. Organizations using Oracle Enterprise Manager do not need to download and install the Oracle Data Masking and Subsetting Pack separately. Oracle Enterprise Manager provides unified browser-based user interface for administration. All Data Masking and Subsetting objects are centrally located in the Oracle Enterprise Manager repository, which facilitates centralized creation and administration of Data Discovery, Masking Definitions and Data Subsetting Definitions. In addition to its intuitive cloud control Graphical User Interface (GUI), Oracle Enterprise Manager also provides Command Line Interface (EMCLI) to automate select Data Masking and Subsetting tasks.

Figure 1-2 Oracle Data Masking and Subsetting Architecture

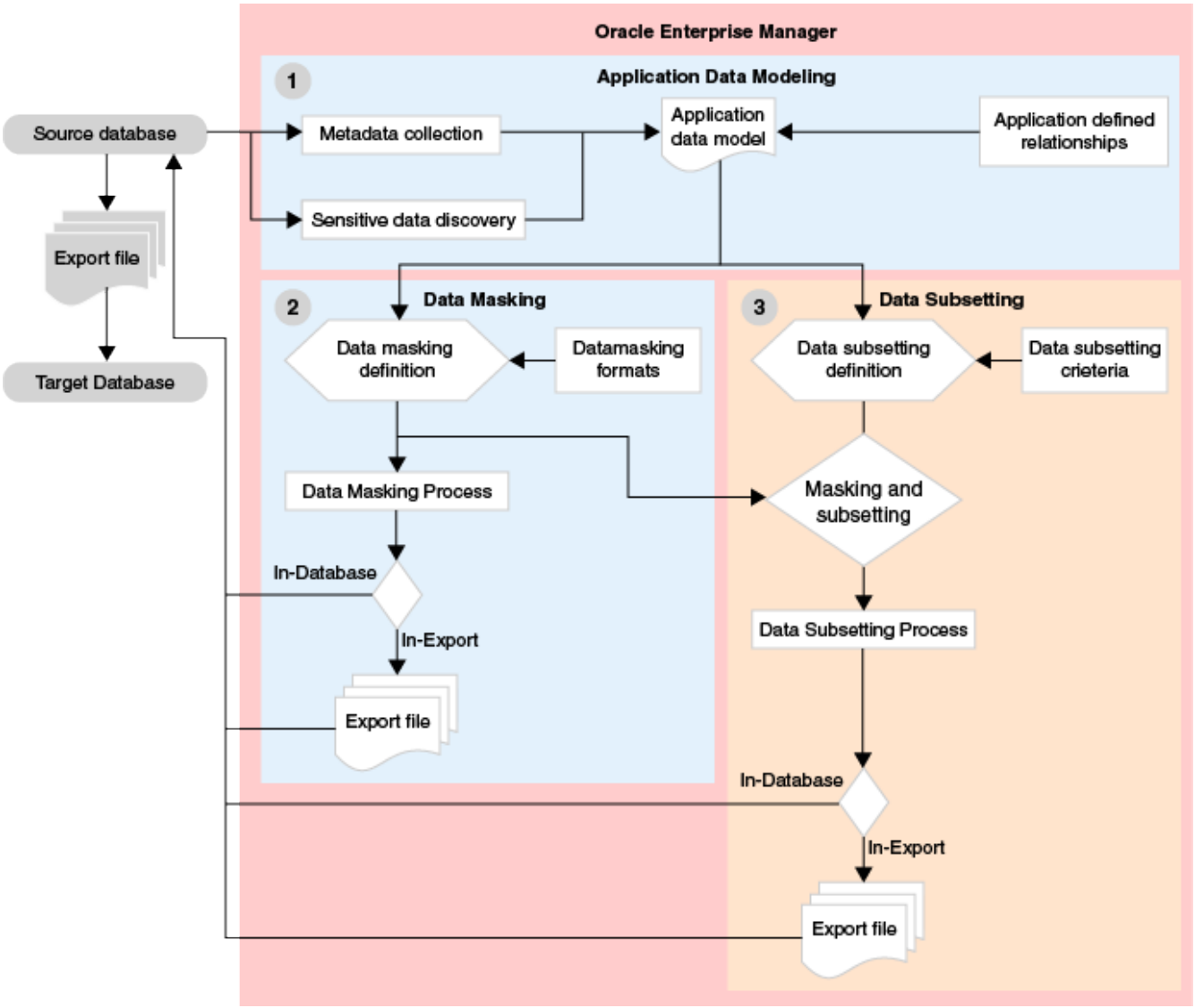


For more details on Oracle Enterprise Manager Architecture, please refer to the *Oracle Enterprise Manager Introduction Guide*.

Methodology

The following diagram explains the Oracle Data Masking and Subsetting workflow.

Figure 1-3 Oracle Data Masking and Subsetting Workflow



The following steps describe the Oracle Data Masking and Subsetting Workflow:

1. **Create an Application Data Model** — To begin using Oracle Data Masking and Subsetting, you must create an Application Data Model (ADM). ADMs capture application metadata, referential relationships, and allow discovery of sensitive data from the source database.
2. **Create a Data Masking Definition** — After an ADM is created, the next step is to create a data masking definition. A masking definition includes information regarding the columns to be masked and the format to use for masking each of these columns.
3. **Create a Data Subsetting Definition** — Create a data subsetting definition to define the conditions and parameters to subset data.

Deployment Options

Oracle Data Masking and Subsetting provides the following modes for masking and subsetting data:

- **In-Database mode** directly masks and subsets data within a non-production database with minimal or zero impact on production environments. As In-Database mode permanently changes the data in a database, this deployment mode is recommended for non-production environments such as staging, test or development databases instead of production databases.
- **In-Export mode** masks and subsets the data in near real-time while extracting the data from a database. The masked and subsetted data that is extracted is written to data pump export files, which can be further imported into test, development or QA databases. In general, In-Export mode is used for production databases. In-Export method of masking and subsetting is a unique offering from Oracle that sanitizes sensitive information within the product perimeter.
- **Heterogeneous mode** Oracle Data Masking and Subsetting can mask and subset data in non-Oracle databases. Target production data is first copied from the non-Oracle environment into Oracle Database using an Oracle Database Gateway, and is then masked and subsetted within the Oracle Database, and is finally copied back to the non-Oracle environment. This approach is very similar to the steps used in various ETL (Extract, Transform, and Load) tools, except that the Oracle Database is the intermediary that transforms the data. Oracle Database Gateways enable Oracle Data Masking and Subsetting to operate on data from Oracle MySQL, Microsoft SQL Server, Sybase SQL Server, IBM DB2 (UDB, 400, z/OS), IBM Informix, and Teradata.

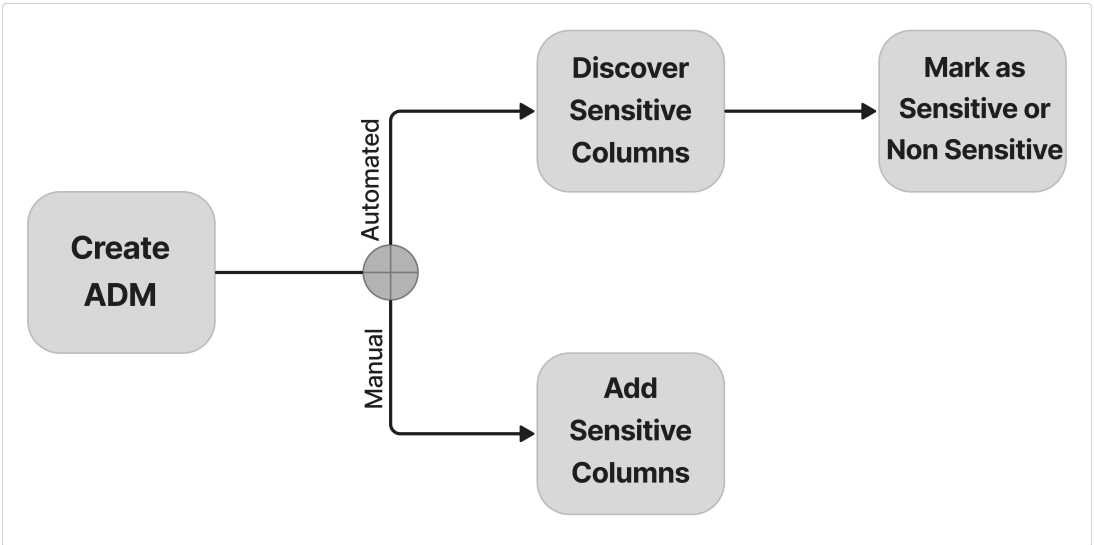
> **Note:**
>
> For information on licensing, please refer to Oracle Database Licensing Guide.

Data Discovery

The Data Discovery component of Data Masking and Subsetting on Enterprise Manager helps you scan and manage sensitive data in your target database. This component supports the creation of Application Data Models (ADM) and Sensitive Types to facilitate data management.

A sensitive type identifies the type of sensitive data, such as national insurance number, using a combination of data patterns in column names, column data, and column comments. Although Oracle provides pre-defined sensitive types to facilitate in identifying sensitive data, users can also create their own custom sensitive types according to their requirements. Automated discovery procedures leverage sensitive types to pattern match the column name, column comment and sample data in the database table columns while scanning for sensitive information.

Figure 1-4 ADM Workflow



Creating an ADM is an important step in discovering and managing sensitive data. The ADM keeps a record of applications, tables, and the relationships between table columns that are declared in the data dictionary, imported from application metadata, or user-specified.

Oracle provides predefined sensitive types that leverage pattern matching in column names, comments, and data content. Users can also create their own sensitive types. Automated discovery uses these types to scan and sample data, identifying sensitive data such as national insurance numbers.

Oracle Data Masking Application Templates deliver pre-identified sensitive types, their relationships, and industry-standard best practice masking techniques out-of-the box for packaged applications such as Oracle E-Business Suite and Oracle Fusion Applications. Use the Self Update feature to get the latest masking and subsetting templates available from Oracle.

Data Masking

Data Masking protects sensitive data by replacing it with realistic, yet fictitious data. This ensures that data remains usable while safeguarding sensitive information.

Masking formats are the building blocks of a data masking definition which define the format for masking different sensitive columns according to different business requirements

Oracle provides a library containing ready-to-use masking formats for masking data, or users can create custom masking formats as required. Masking Definitions associate table columns with the appropriate masking formats to be used for masking the data.

Figure 1-5 Data Masking Workflow

Data Subsetting

Data Subsetting involves creating smaller, manageable sets of data from larger enterprise-class applications while retaining the data's integrity and usability. Oracle Data Masking and Subsetting simplifies this process through goal-based and condition-based subsetting techniques.

Subsetting Definitions allow users to set rules for how data is extracted. For example, goals can be defined by relative table size, such as extracting a 1% subset from a table with 1 million rows. Conditions can be defined based on specific criteria, such as time (e.g., excluding user records created before a certain year) or region (e.g., extracting only Asia-Pacific data for development purposes). These conditions are specified using a SQL `WHERE` clause.

Figure 1-6 Data Subsetting Workflow

2

Prerequisites

This chapter helps you prepare with the prerequisites and other important things that you must consider before installing and using Oracle Data Masking and Subsetting.

> **Note:**
>
> You must be licensed for the Oracle Data Masking and Subsetting Pack in order to use any of its features. For licensing information, please refer to the Oracle Database Licensing Guide.

Oracle Data Masking and Subsetting Access Rights

To use Oracle Data Masking and Subsetting in Oracle Enterprise Manager, users need specific privileges:

User Privileges

Privilege	Description
Application Data Model (`DB_ADM_ADMIN`)	For managing and using the application data model feature
Data Masking Definition (`DB_MASK_ADMIN`)	For managing and using the data masking feature.
Data Subset Definition (`DB_SUBSET_ADMIN`)	For managing and using the data subsetting feature.

Default Access: By default, all Enterprise Manager administrators can access the Oracle Data Management and Subsetting pages, including the Application Data Models, Sensitive Types, Data Subset Definitions, Data Masking Definitions, and Data Masking Formats, because of the `TDM_ACCESS` privilege, which is part of the `PUBLIC` role. The Super Administrator can revoke this privilege to restrict access, removing these items from the Cloud Control console.

Privilege Access Model

Enterprise Manager allows Super Administrators to restrict Test Data Management (TDM) object access to authorized users only, by granting Operator or Designer privileges.

Operator Privileges

Privilege	View	Edit	Delete	Additional Permissions
ADM Operator	✓			-

Data Subset Definition Operator	✓	Can create data subsets to export files, create on a database, and save the subset script
Data Masking Definition Operator	✓	Can generate data masking scripts, schedule jobs, and export definitions

Users with Designer privileges can manage, enhance, and modify Test Data Management (TDM) objects and can grant/revoke privileges to others.

Designer Privileges Summary

Privilege	View	Edit	Delete	Additional Permissions
ADM Designer/ Data Subset Definition Designer/ Data Masking Definition Designer	✓	✓	✓	Grant/revoke privileges to others

Access Control Procedures

To assign privileges on Application Data Models, Data Masking Definitions, and Data Subsetting Definitions, follow these procedures:

- **Modify an Existing ADM**
- **Granting Privileges on a Subset Definition**

Additional Information:

The following privileges are required for a database user to perform data masking operation in a normal database where Database Vault is **not** enabled:

EM Privileges

- Other Resource Privileges:
  - Job System —> Create (`CREATE_JOB`)
  - Named Credential —> Create new Named Credential (`CREATE_CREDENTIAL`)
    Other Resource Privileges

Resource Type	Privilege	Description
Application Data Model	Use Application Data Models in EM	Ability to use Application Data Models in EM
Data Masking Definition	Use Data Masking Definitions in EM	Ability to Use Data Masking Definitions in EM
Data Subset Definition	Use Data Subset Definitions in EM	Ability Use Data Subset Definitions in EM

Resource Type	Privilege	Description
Job System	Create	Ability to submit jobs, create library jobs, create deployment procedure instance, and create deployment procedure configuration
Named Credential	Create new Named Credentials	Ability to create new Named Credentials

- Target Privileges:
  - Execute Command Anywhere (**PERFORM_OPERATION_ANYWHERE**)
  - View any Target AND Connect to any viewable target (**VIEW_ANY_TARGET** AND **CONNECT_ANY_VIEW_TARGET**) OR grant "Connect" privileges under Target Instance Privileges to specific instances

Target Privileges

Privilege Name	Target Types	Description
Execute command anywhere	Host	Execute any OS command at any Agent
View any Target	All	Ability to view all managed targets in Enterprise Manager
Connect to any viewable target	All	Ability to connect and manage any viewable target

DB Privileges and Roles

- CREATE SESSION
- SELECT_CATALOG_ROLE
- CREATE TABLE
- CREATE PROCEDURE
- CREATE SEQUENCE
- CREATE TYPE
- CREATE TABLESPACE
- CREATE ANY DIRECTORY
- DROP ANY DIRECTORY
- UNLIMITED TABLESPACE
- DATAPUMP_EXP_FULL_DATABASE
- DATAPUMP_IMP_FULL_DATABASE
- EXECUTE ANY PROCEDURE
- ALTER SESSION
- EXECUTE ON SYS.DBMS_CRYPTO
- EXECUTE ON SYS.DBMS_RANDOM
- EXECUTE ON SYS.UTL_RECOMP
- ANALYZE ANY

- ADMINISTER SQL TUNING SET
- EXECUTE ON SYS.DBMS_AQADM
- ALTER ANY INDEX
- ALTER SYSTEM
- SELECT ANY TABLE
- CONNECT
- DBA
- RESOURCE
- CREATE VIEW
- CREATE SYNONYM
- CREATE DATABASE LINK
- SELECT ANY DICTIONARY
- EXECUTE ANY TYPE

> **Note:**
>
> `GRANT ANY OBJECT PRIVILEGE WITH GRANT OPTION` privilege is also required for EBS and FA application suites.

Ensure the database user performing the masking has object/system privileges to create, drop, alter, select, insert and compile the objects being masked (in case the objects are owned by another schema).

For Basic set privileges to perform Data masking operation in a Database Vault enabled environment, please refer the following Oracle documentation URL: *Oracle Database Vault Administrator's Guide*

Application Data Model (ADM):

1. `ADD_AUTH_TO_REALM`: Grants the user permission to deploy ADM packages.
   Example: If you want to grant this privilege to `SCOTT` user then login as user with `dv_owner` role and execute the following SQL command:

   ```
   SQL>exec dbms_macadm.add_auth_to_realm('Oracle Enterprise Manager','SCOTT',
   NULL, dbms_macutl.g_realm_auth_owner);
   ```

In-Place: Masking and Subsetting:

1. Grant `EXECUTE` on `UTL_FILE` to allow the user to read and write files.
2. Authorize *<user>* in realm protecting *object*: (Needed even if object owner is performing masking/subsetting)
   Example: `SQL>exec dbms_macadm.add_auth_to_realm('<Realm Name>','<user>', NULL, dbms_macutl.g_realm_auth_owner);`

Export: Masking and Subsetting:

1. Grant Execute on `UTL_FILE` to *<user>* ;
2. Realm Authorization:

a. Object owner:
If we want to perform export masking or subsetting on `TEST.EMPLOYEE` and we execute the job as `TEST`, then we don't need to authorize `TEST` in any realm protecting `TEST.EMPLOYEE`. Without granting authorization, `TEST` can export its own tables.

b. Admin User:
i) Grant authorization to Admin user in realm protecting `TEST.EMPLOYEE`.

ii) Authorize Admin user to export tables under `TEST` schema:

Example: `SQL>exec dbms_macadm.authorize_datapump_user('<Admin User>','TEST');`

Storage Requirements

Although Oracle Data Masking and Subsetting objects such as data models, masking and subsetting definitions consume a negligible amount of storage space, depending on the amount of data being stored over a period of time, you may need to allocate additional storage space to Oracle Enterprise Manager's repository database.

This section details the storage recommendations for masking and subsetting.

- In-Database Masking:
  - 3X of additional space in the user tablespace (X being the largest table in size)
  - 2X of additional space in temporary tablespace.
- In-Export Masking:
  - 2X additional space in the user tablespace (X being the largest table in size)
  - 2X of additional space in temporary tablespace
  - Sufficient disk space to store the generated export dump file.
- In-Export Subsetting:
  - X additional space in the user tablespace (X being the largest table in size)
  - Sufficient space to store the generated dump files.
- In-Database Subsetting:
  - 2X additional space in the user tablespace (X being the largest table in size)
  - 2X additional space in temporary tablespace.

> **Note:**
>
> The recommended storage requirement for integrated masking and subsetting is the sum total of the storage requirement for masking and subsetting as mentioned above.

3

Data Discovery

Introduction

Sensitive Data Discovery is the first feature offered as part of Data Masking and Subsetting on Enterprise Manager. This feature lets us create or manage Application Data Models and Sensitive Types. Scanning and tagging of sensitive data and modeling of referential relationships are incorporated within an Application Data Model (ADM) whereas sensitive types are important in terms of describing what kind of sensitive data you want the feature to search for.

Let us get started with Application Data Models.

Application Data Models

ADM Overview

Under Data Discovery, creating ADM is the first step in discovering and managing your sensitive data where it can store automatically discovered or manually added sensitive columns. ADM stores the list of applications (schemas), objects (tables or editioning views) and columns as well as relationships between columns. The ADM maintains sensitive types and their associated columns, and is used by subsequent features, such as data subsetting and data masking, to securely produce test data. Creating an ADM is a prerequisite for data masking and data subsetting operations.

> **Note:**
>
> Please refer to how we use the terms Application, Schema, Object, and Table in the following sections. Below is a brief description of each:
>
> - **Application:** An application groups related schemas and objects, providing a high-level organizational view for managing data masking and subsetting configurations.
> - **Schema:** A logical collection of database objects, typically associated with a user, that includes tables, views, indexes, and more.
> - **Object:** Any entity within a schema that holds data or defines a structure, such as a table, view, or index.
> - **Table:** A structured set of data organized in rows and columns within a schema.

Figure 3-1 Overview of an Application Data Model

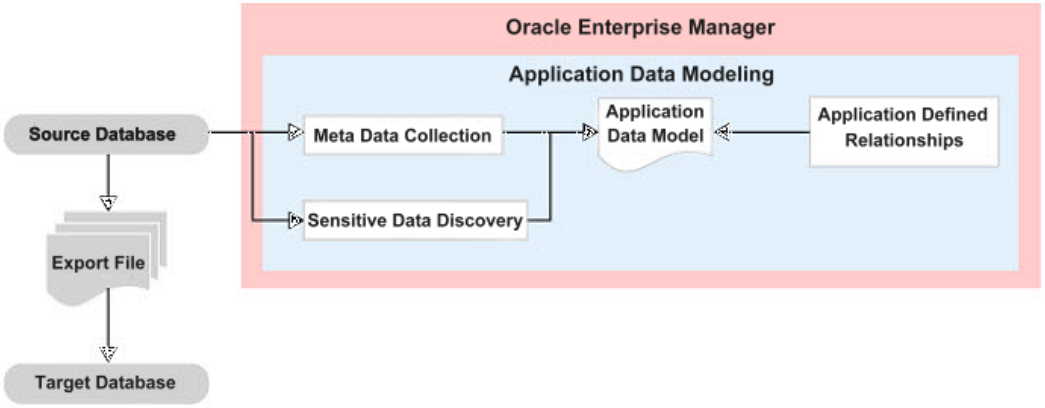
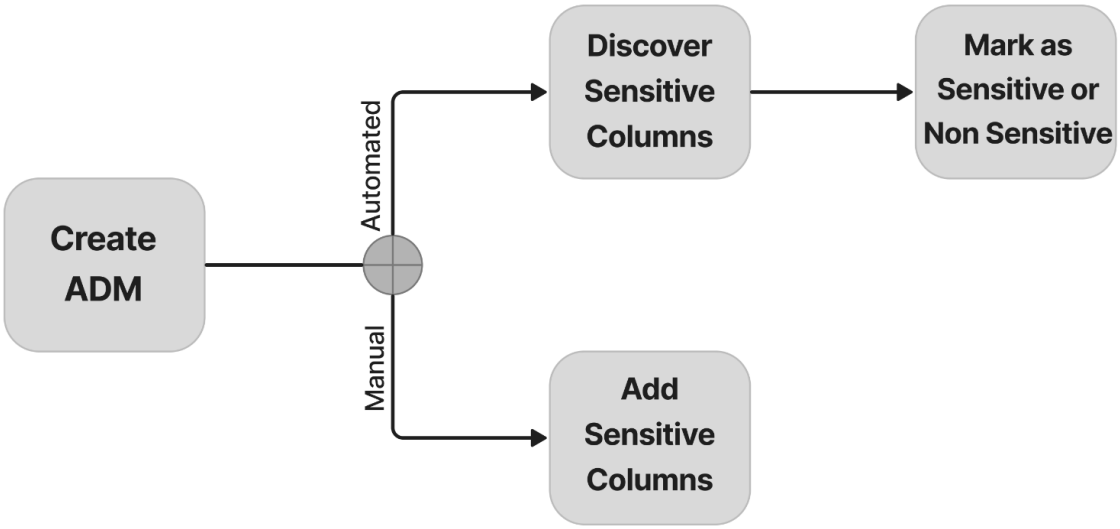


Figure 3-2 ADM Workflow



The steps shown in the workflow diagram can be referenced in the following topics:

- **Create a New ADM**
- **Modify an Existing ADM**
- **Verifying an ADM**
- **Importing and Exporting an ADM**
- **Create a New Sensitive Type**

Before proceeding, ensure you have all the necessary privileges mentioned in the **Prerequisites** chapter.

When you create an Application Data Model, the PL/SQL metadata collection packages are automatically deployed on the target database. The Database user must have DBA privileges to auto-deploy the packages.

We recommend that you create an Application Data Model for the first time with a less privileged user.

Create a New ADM

This step allows you to define and structure the relationships between your target database, application, and schema. This setup is essential for accurately modeling and managing the data within your application environment.

To create an Application Data Model:

1. Navigate to the Enterprise Manager Named Credentials page and create a named credential that can be used to perform DMS operations on the target database.
   For detailed information regarding the creation of a named credential, see Cloud Control Security Guide - Named Credentials
2. From the Application Data Models page, review the overview diagram for a quick glance at the steps involved in creating an ADM with sensitive data.
3. Click **Create**. A pop-up window requesting general properties information appears.
4. Specify a name for the ADM to be created.
5. Select the **Target Type**, **Target Database** and **Database Named Credential** from the drop down list.
6. Select an **Application Suite**:
   - If you select **Custom Application Suite**:
     - By default, metadata collection is enabled for creating the ADM.
   - If you select **Oracle Application Suite**:
     - Oracle E-Business Suite – Provide the database credentials for APPS user (or equivalent), and click **Submit** to create the ADM.
     - Oracle Fusion Applications – Provide database credentials for FUSION user (or equivalent), and click **Submit** to create the ADM.

   Please note the following points about metadata collections:
   - The metadata collection for the selected application suite populates the ADM with the applications and tables in the suite.
   - The ADM can collect metadata for one or more schemas. An ADM application typically represents a schema. Each schema you select becomes an ADM application, and the ADM becomes populated with the tables in the schema, particularly in the case of custom applications. However, please note that multiple applications can also map to a single schema, as in the case of Fusion Applications. The actual mapping depends on the application metadata discovered by the metadata collection job.
7. Select the schemas you want to include as applications in the ADM being created. The drop-down only displays a limited number of possible schemas in the chosen database. Start searching in order to find and select the schema(s) of interest

   > **Note:**
   >
   > For accurate results, ensure that you execute `dbms_stats.gather_table_stats` to gather the stats of all the tables.
8. Select the following "Relationship Discovery Type":

- **Database Level (Dictionary-Based) Relationships** — Use this feature to discover relationships between database table columns predefined in the data dictionary as referential integrity constraints (primary key and foreign key). For example, the `ORDERS` table's `CUSTOMER_ID` column has a foreign key constraint referring to the `CUSTOMER_ID` primary key in the `CUSTOMERS` table.
- **Application Level (Non-Dictionary) Relationships** — Use this feature to discover parent-child relationships between database table columns that are not predefined in the data dictionary as referential integrity constraints (primary key and foreign key). For example, the `EMPLOYEE_CONTACTS` table's `EMPLOYEE_ID` column is related to the `EMPLOYEES` table's `EMPLOYEE_ID` column, but no foreign key constraint is defined in the database.

  Click **Continue**.

9. Specify the parameters for scheduling the metadata collection job. You can choose to either run the ADM creation job immediately or schedule it to start at a later time.
10. Proceed if you checked **Application Level** discovery in Step 8, otherwise skip to step 11.
    a. Click **Next.**
    b. Select a Sampling option:
       - **Sampling Column Names**: This option identifies application-defined parent-child relationships using column names. Column name patterns can be specified using regular expressions. For example: `EMPID.*` and `DEPT_EMPID.*`.
       - **Sampling Column Data**: This option identifies application-defined parent-child relationships using data or value patterns. Data patterns can be specified using regular expressions. For example a 10 digit US telephone number such as 123-456-7890 can be specified as [1-9]{3}[-][1-9]{3}[-][1-9]{4}.
         Oracle Data Masking and Subsetting will match the values in the column to identify the potential primary key and foreign key using the name and (or) data pattern specified by the user and return the results.

         The potential foreign keys are verified for containment within the potential primary key column, that is, the values of potential foreign key must already be present in the potential primary key. A containment test is done to meet 90% accuracy, that is, if the foreign key column is 90% contained in the primary key, a match is flagged. This test is done to include any orphan rows that might be present in applications such as Oracle's E-Business Suite and Oracle Fusion Applications.
11. Click **Create** to submit the **Create Application Data Model** job. The ADM you created appears in the Application Data Models page. The application data model is locked and cannot be edited when the metadata is being collected. Use the Most Recent Job Status table column to monitor the status of the metadata collection job.

Modify an Existing ADM

You can modify an existing ADM to view or add sensitive columns, applications, objects, privileges and referential relationships:

Adding or Removing an Application From the ADM

1. Select an Application Data Model, click on the **Actions** button, click on **Modify** and click **Applications**.
2. The Application subpage appears displaying the applications discovered during the metadata collection process.
3. To create a new Application, click **Add**. The Add Application pop-up window appears.

4. Specify a name for the application, a nick name/short name, description for the application, and Database Named Credentials.
5. Type the name of the schema you want to add and click the search icon. Alternatively, click the search icon without any text to retrieve all schemas.
6. Select a schema from the list (you may need to type to filter and view all options).
7. Click **Add** to include the newly created application to the data model.
   The application now appears along with the defined schema.
8. To remove an application, select the application from the Applications page and click **Delete**.
   Similarly, you can modify the objects and table types under the Actions menu.

Viewing the Referential Relationships

To view referential relationships:

1. Select an Application Data Model, click on the **Actions** button, click on **Modify** and click **Referential Relationships**.
2. A dialog opens with the list of parent and dependent key relationships. The following types of referential relationships are supported:
   - *Dictionary-defined*
     This is the referential relationship that the metadata collection extracted, resulting from primary key and foreign key relationship. You can remove relationship from the ADM if desired.
   - *Non-Dictionary Based*

     This is the referential relationship that is not defined in the Oracle data dictionary, and is achieved by matching the column names and column values of potential foreign keys with column names and column values of potential primary keys along with their data types.
   - *User-defined*

     This is the referential relationship that is not defined in the Oracle data dictionary, and user can create manually based on their requirement.

Adding and Removing Referential Relationships

To manually add or remove a referential relationship:

1. Select an Application Data Model, click on the **Actions** button, click on **Modify** and click **Referential Relationships**.
2. Click on the **Add** button under the Parent Referential Relations section.
   The Add Referential Relationship pop-up window appears.
3. Select the requisite Database Named Credential, Parent Key and Dependent Key information.
4. The new dependent column now appears in the referential relationships list.

Automatically Discover Sensitive Data

To discover sensitive columns:

1. Select an Application Data Model, click on the **Actions** button, click on **Modify** and click **Discover Sensitive Columns**.
2. Click **Schedule** and fill in the details for Database Named Credentials, Applications, Sensitive Types and click **Submit**.

3. Once the job is submitted, refresh until the status changes to Succeeded.
4. Highlight the succeeded job by clicking on the row. Notice the sensitive columns appeared under Discovered columns.
5. The Sensitive Status for the columns by default is Undefined. To set the sensitive status of any column, select the row for the column in Discovered Columns table and click **Mark Sensitive** (or **Mark Not Sensitive** if it is a false positive).
6. Click **Close** to return to the Application Data Models page.

Manually Add Sensitive Columns

To add/remove sensitive columns:

1. Select an Application Data Model, click on the **Actions** button, click on **Modify** and click **Manage Sensitive Columns**.
2. A dialog appears with the list of all sensitive columns currently part of the selected ADM. Now, click **Add**.
   The Add Sensitive Column pop-up appears.
3. Provide the required information and an optional Sensitive Type, then click **Add**.
   The sensitive column now appears in the table of Sensitive Columns.

Viewing the Discovery Results

To view the discovered sensitive columns:

1. Select an Application Data Model, click on the **Actions** button, click on **Modify** and click **Discover Sensitive Columns**.
2. Select one of the discovery jobs to view the sensitive columns discovered by the selected job.

Associating a Database to an ADM:

1. Select an Application Data Model, click on the **Actions** button, click on **Modify** and click **Associated Databases**.
   This dialog lists all of the databases associated with this ADM and the schemas assigned to each application per database. You can add more databases that give you a choice of data sources during masking and subsetting operations.
2. Click **Add**, then select a target type and target database from the popup.
   The selected database now appears in the Database section of the Associated Databases dialog.
3. Click on **Associate**.
4. To change a schema, select the associated database on the left, select the application on the right for which the schema is to be changed, then click **Update**.
5. Select the missing schema from the list in the pop-up, then click **Update**.

Assigning Privileges to an Existing ADM

You can grant privileges on an Application Data Model that you create so that others can have access. To do so, you must be an Enterprise Manager Administrator with at least Designer privileges on the ADM.

To assign privileges to an existing ADM:

1. From the Application Data Models page, select the ADM to which you want to grant privileges and click on **Action**.

2. From the Actions menu, select **Modify** then **Privileges** then **Grant**, and select one of the following for Grant Privilege:
   - Operator – to grant Operator privileges on the ADM to selected roles or administrators, which means the grantees can view and copy but not edit and delete the definition.
   - Designer – to grant Designer privileges on the ADM to selected roles or administrators, which means the grantees can view, edit, and delete the definition.
3. Filter by name, if desired. Make your selections and click **Grant**.
   The selected names now have privileges on the ADM.
4. Use the **Revoke** action if you want to deny any privileges that were previously granted.

Verifying an ADM

After you have created an ADM, the ADM Status column can indicate Valid, Invalid, Needs Verification, or Needs Upgrade.

- Invalid status – Verify the target database to update the referential relationships in the application data model with those found in the data dictionary, and to also determine if each item in the application data model has a corresponding object in the database.
- Needs Verification status – You have imported an Oracle supplied template and you must verify the ADM before you can use it. This is to ensure that necessary referential relationships from data dictionary are pulled into the ADM.

To verify a target database:

1. Select the ADM to be verified.
2. From the Actions menu, select **ADM Verification** then **Verify**.
3. Select the Associated Database to be verified as well as its credentials
4. Once again select the ADM, select **ADM Verification**, and then select **Results**.
5. Monitor the status of the verify job submitted earlier by periodically clicking on the **Refresh** button.
6. Click **Verify** to schedule a verification job.
7. After the job completes successfully, click the Target Database and note any issues listed.
8. Fix the object problems, rerun the Verification Job, then check that the Target Database Status is now Valid.

Importing and Exporting an ADM

You can share an ADM with other Enterprise Manager environments that use a different repository by exporting it and then importing it into the new repository.

An exported ADM is by definition in the XML file format required for import. You can edit an exported ADM XML file prior to import. When exporting an ADM for subsequent import, it is best to have one that uses most or all of the features—applications, tables, table types, referential relationships, sensitive columns. This way, if you are going to edit the exported file prior to import, it is clear which XML tags are required and where they belong in the file.

- Importing an ADM
- Exporting an ADM

> **Note:**
>
> There are Enterprise Manager CLI verbs to export and import an ADM if you want to perform these operations remotely or script them. See: Enterprise Manager Cloud Control Command Line Interface

Importing an ADM

1. From the Application Data Models page, click on the **Import** option shown on top of the table.
2. In the pop-up that appears, specify a name for the ADM, the Target Database you want to assign to the ADM, and the XML file which contains the ADM to be imported
3. Click **Import**.
   The ADM now appears on the Application Data Models page.

> **Note:**
>
> After importing an ADM, you may want to discover sensitive columns or run a verification job. In the process of performing these tasks, the PL/SQL metadata collection packages are automatically deployed on the target database. The Database user must have DBA privileges to auto-deploy the packages.

Exporting an ADM as an XML File

1. From the Application Data Models page, select the ADM you want to export.
2. From the Actions menu, select **Export** then select **Export to File**.
3. In the popup, navigate the location at which the exported XML file should be saved and click **Save**.

Export sensitive metadata from an ADM to Transparent Sensitive Data Protection (TSDP) enabled database

1. From the Application Data Models page, select the ADM to export.
2. From the Actions menu, select **Export**, then select **Export to TSDP Catalog**.
3. The Application Data Models page displays a table of associated databases. Select a database and click the **Export** button.
4. In the Export pop-up that appears, provide credentials for the selected database and click **Export**.
   A message appears on the Application Data Models page confirming that the sensitive data was copied to the database.

For detailed information on TSDP, see Oracle Database Security Guide.

Sensitive Types

Oracle provides predefined sensitive types based on which sensitive data is discovered. You can either choose an existing sensitive type to discover data or create a new one.

Create a New Sensitive Type

To create a new sensitive type:

1. Under Data Discovery, select **Sensitive Types**.

   The Sensitive Types page appears.

2. Click **Create**.

   The Create Sensitive Type pop-up appears.

3. Specify a required name and regular expressions for the Column Name, Column Comment, and/or Column Data search patterns.

   - The 'Or' Search Type means that any of the patterns can match for a candidate sensitive column.
   - The 'And' Search Type means that all of the patterns must match for a candidate sensitive column.

   If you do not provide expressions for any of these parameters, the sensitive data discovery job will not search for the entity.

4. Click **Create**.

   The sensitive column appears in the table in the Sensitive Types page.

You can also create a new sensitive type based on an existing type:

1. Under Data Discovery, select **Sensitive Types**.

   The Sensitive Types page appears.

2. Select either a sensitive type you have already defined, or select one from the out-of-box types that the product provides.

3. Click **Create Like**.

   The Create Sensitive Type pop-up appears.

4. Specify a required name and alter the existing expressions for the Column Name, Column Comment, and Column Data search patterns to suit your needs.

5. Click **Create**.

   The sensitive column appears in the table in the Sensitive Types page.

> **Note:**
>
> If we export an ADM with custom sensitive types, the custom sensitive type information gets listed in the exported XML.
>
> If an EM doesn't have a particular sensitive type, and the imported XML has that sensitive type info, the new sensitive type gets created in the EM automatically when import happens.
>
> It might be possible that the XML doesn't have a particular sensitive type information, even when the sensitive columns of that ADM are defined using that particular sensitive type. After importing that XML, the sensitive types of those sensitive columns would be "UNDEFINED".

4

Data Masking

This chapter provides conceptual information about the features that comprise Oracle Data Masking, and procedural information about performing the task sequence including Masking Formats and Masking Definitions. Data masking presupposes that you have created an Application Data Model (ADM) with sensitive columns defined.

Introduction

Enterprises run the risk of breaching sensitive information when copying production data into non-production environments for the purposes of application development, testing, data sharing or analysis. Oracle Data Masking helps reduce this risk by irreversibly replacing the original sensitive data with fictitious data so that production data can be shared safely with non-production users. Accessible through Oracle Enterprise Manager, Data Masking provides end-to-end secure automation for provisioning test databases from production in compliance with regulations.

Data masking (also known as data scrambling and data anonymization) is the process of replacing sensitive information copied from production databases to non-production databases with realistic, but scrubbed, data based on masking rules. Data masking is ideal for virtually any situation when confidential or regulated data needs to be shared with non-production users. These users may include internal users such as application developers, or external business partners such as offshore testing companies, suppliers and customers. These non-production users need to access some of the original data, but do not need to see every column of every table, especially when the information is protected by government regulations.

Data masking enables organizations to generate realistic and fully functional data with similar characteristics as the original data to replace sensitive or confidential information. This contrasts with encryption or Virtual Private Database, which simply hides data, and the original data can be retrieved with the appropriate access or key. With data masking, the original sensitive data cannot be retrieved or accessed.

Names, addresses, phone numbers, and credit card details are some of the examples of data that require protection of the information content from inappropriate visibility. Live production database environments contain valuable and confidential data access to this information is tightly controlled. However, each production system usually has replicated development copies, and the controls on such test environments are less stringent. This greatly increases the risks that the data might be used inappropriately. Data masking can modify sensitive database records so that they remain usable, but do not contain confidential or personally identifiable information. Yet, the masked test data resembles the original in appearance to ensure the integrity of the application.

Related Oracle Security Offerings

Table 4-1 Related Offerings

Oracle Solutions	Description
Oracle Data Masking and Subsetting	Software for creating masked and subsetted copies of production data for use in non-production environments such as testing and development databases.
Oracle Data Safe (Data Discovery/ Data Masking)	Discover and mask sensitive data with a cloud service that supports Oracle Databases everywhere: in the Oracle Cloud, on-premises, and third-party clouds.
Oracle Label Security	Implements Multi-Level Security (MLS), enabling rows with differing sensitivity to reside in the same table. Explicitly labels rows with group, compartment, and sensitivity levels then matches them with user labels.
Oracle Data Redaction	Redacts sensitive data from query results before display through client applications. Enforces redaction at runtime, with low overhead, and according to conditions set in policies.

Data Masking Components

Data Masking consists of two main components:

- **Masking Formats**

  A masking format represents the definition on how to mask some given data. For example, an IP Address masking format could be a 3 random digits from 0 to 255 followed by a dot (.) fixed string repeated 4 times (without a trailing dot). A masking format can either be one that you create, or one from the list of Oracle-supplied default masking formats.

  As a matter of best practice, organizations should create masking formats for all commonly regulated information so that the formats can be applied to the sensitive data regardless of which database the sensitive data resides in. This ensures that all sensitive data is consistently masked across the entire organization.

- **Masking Definitions**

  A masking definition defines a data masking operation to be implemented on one or more tables in a database. Masking definitions associate table columns with formats to use for masking the data. They also maintain the relationship between columns that are not formally declared in the database using related columns.

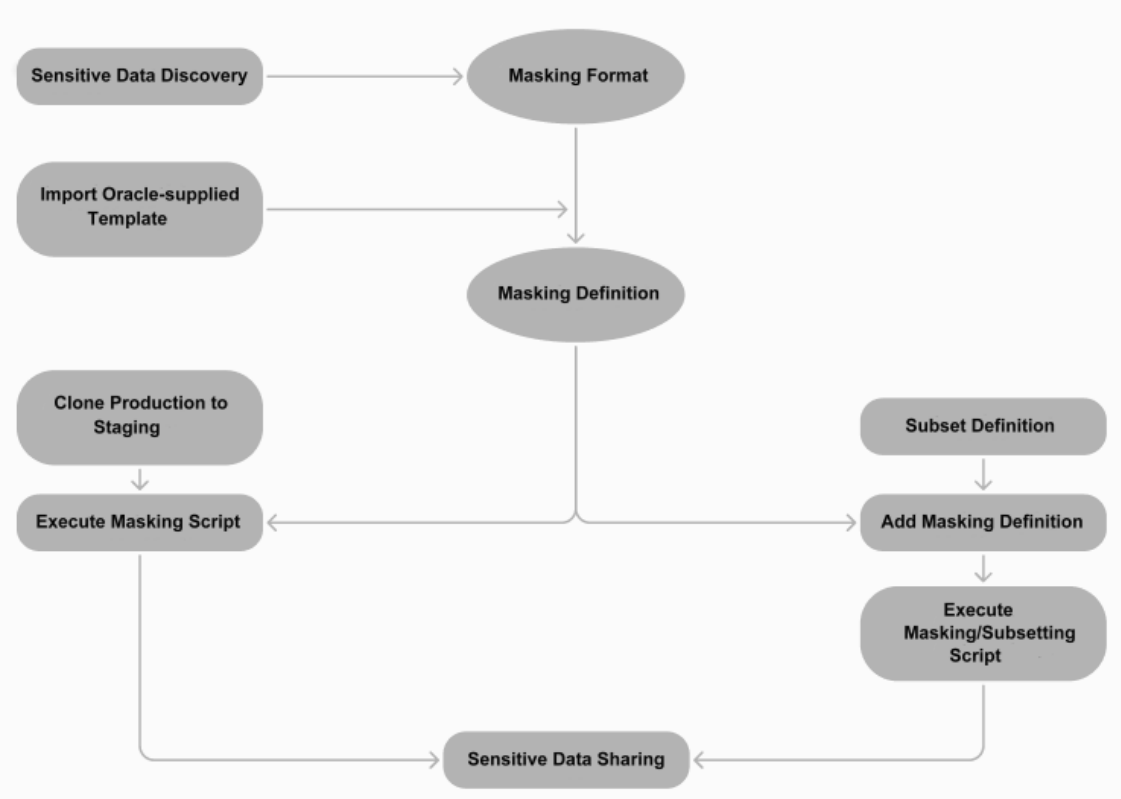
  You can create a new masking definition or use an existing definition for a masking operation. To create a masking definition, you specify the column of the table and how data should be masked. If the columns being masked are involved in unique, primary key, or foreign key constraints, data masking generates the values so that the constraints are not violated. Masking ensures uniqueness per character using decimal arithmetic. For example, a numeric string of length 5 generates a maximum of only 99999 unique values. Similarly, a numeric string of length 1 generates a maximum of only 9 unique values.

You can export masking definitions to files and import them on other systems. This is important when the test and production sites reside on different Oracle Management Systems or on entirely different sites.

Recommended Data Masking Workflow

The following figure shows that the production database is cloned to a staging region and then masked there. During the masking process, the staging and test areas are tightly controlled like a production site.

Figure 4-1 Recommended Data Masking Workflow



The workflow diagram illustrates the process of data masking and subsetting within Oracle environments, with the following steps:

1. **Sensitive Data Discovery**: Identifies sensitive data that requires protection.
2. **Masking Format**: Involves creating masking formats, either through predefined Oracle-supplied templates or custom masking rules.
3. **Masking Definition**: Consolidates sensitive columns and their associated masking formats, which define how to mask the sensitive data in these columns.
4. **Clone Production to Staging**: A copy of production data is cloned to a staging environment where masking operations are performed.
5. **Execute Masking Script**: Applies the defined masking rules to the cloned data, ensuring sensitive data is protected.
6. **Subset Definition**: A definition to subset data, limiting data exposure.

7. **Add Masking Definition**: Combines the masking rules with the subsetting definition.
8. **Execute Masking/Subsetting Script**: Executes both masking and subsetting on the data set.
9. **Sensitive Data Sharing**: The masked and subsetted data can then be safely shared without revealing sensitive information.

Data masking is an iterative and evolving process handled by the security administrator and implemented by the database administrator. When you first configure data masking, try out the masking definition on a test system, then add a greater number of columns to the masking definition and test it to make sure it functions correctly and does not break any application constraints. During this process, you should exercise care when removing all embedded references to the real data while maintaining referential integrity.

After data masking is configured to your satisfaction, you can use the existing definition to repeatedly mask after cloning. The masking definition, however, would need to evolve as new schema changes require new data and columns to be masked.

After the masking process is complete, you can distribute the database for wide availability. If you need to ship the database to another third-party site, you are required to use the Data Pump Export utility, and then ship the dump file to the remote site.

You can also perform inline, or at the source, data masking while creating a subset definition.

Masking Formats

A masking definition requires one or more masking formats for any columns included in the masking definition. When adding columns to a masking definition, you can either create masking formats manually or import them from the format library. It is often more efficient to work with masking formats from the format library.

A masking format is made up of one or more format entries, where a format entry is a predefined type of supported masking method such as Fixed String, Random String, Shuffle etc. In essence, it describes how to transform an input value to an output value. The masking format is a combination of one or more of these transformations which chain together to take some input value and output a final masked value.

This section covers the following topics:

- **Creating a New Masking Format**
- **Oracle-supplied Predefined Masking Formats**
- **Masking Format Categories and Characteristics**
- **Format Entry Options to Customize Masking Format**
- **Providing User-defined and Post-processing Functions**
- **Patterns of Format Definitions**

Creating a New Masking Format

This section describes how to create new masking formats using Enterprise Manager. To create a masking format:

1. From the Targets menu, select **Databases**, then click the **Security** menu and navigate to **Data Masking and Subsetting** then **Data Masking**. You can also access the menu through the database home page by selecting **Security** then **Data Masking and Subsetting** then **Data Masking**.

Go to **Masking Formats** under **Data Masking**. The Masking Format Library page appears with predefined formats that Oracle Enterprise Manager provides.

2. Click **Create**.
   The Create Masking Format page appears, where you can define a masking format.
3. Provide a required name for the new masking format, add a description, select Sensitive Type, and then select a **Custom Format Entry** type from the drop-down list.
   A section appears that enables you to provide input for the format entry you have selected. For instance, if you select Array List, the subsequent box enables you to enter a list of comma-separated values, such as New York, New Jersey, and New Hampshire.
4. Continue adding additional format entries as needed.
5. When done, provide an optional user-defined or post-processing function, then click **Create** to include this customized masking format in the library.
   The Masking Format page reappears with your newly created format displayed in the Format Library table. You can use this format later to mask a column of the same sensitive type.

Creating a Masking Format from an Existing Format

After you have created at least one format, you can use this format as a template, where you can implement most of the masking format using a different name and changing the entries as needed, rather than needing to create a new format from scratch.

1. Navigate to the Masking Formats page and select the format you want to use as a template.
2. Click **Actions** and choose **Create Like**.
3. Enter a name for the new format and select a Sensitive Type.
4. Customize the format entries as needed and click **Create**.

Oracle-supplied Predefined Masking Formats

Enterprise Manager provides several out-of-box predefined formats. All predefined formats and built-in formats are random.

Masking Format Categories and Characteristics

Data masking formats have characteristics. Some common characteristics include combinable, uniqueness, reversible, and deterministic:

- **Combinable**: A masking format is considered combinable when it can be combined with other basic masking formats or predefined masking formats though the use of conditions.
- **Uniqueness**: A masking format is characterized as having uniqueness if it ensures uniqueness of the generated masked data. These types of masking formats are useful for masking columns with uniqueness constraints.
- **Reversible**: A masking format that is characterized as reversible can retrieve original column data from masked data. Data masking usually means permanently replacing the data and ensuring that no one can retrieve the original data. But, sometimes you might want to see the original data.
- **Deterministic**: A deterministic masking format generates consistent output for a given input across databases and data masking jobs. Deterministic masking helps to maintain data integrity across multiple applications and preserve system integrity in a single sign-on environment.

MASKING FORMAT	DESCRIPTION	SAMPLE MASKED DATA	COMBINABLE	UNIQUENESS
Visa Credit Card Number	~10 billion unique Visa credit card numbers	• 4929677281270400 • 4929026969239255	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column
USA Phone Number Formatted	~2.7 billion unique USA phone numbers	• 305-557-9243 • 540-140-3020	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column
USA Phone Number	~2.7 billion unique USA phone numbers	• 6823682394 • 3208603360	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column
UPC Number Formatted	~100 billion UPC numbers	• 1-12345-12345-1 • 2-38193-12983-9	Yes	Yes. The number of distinct values in the specified range must be greater than or equal to the number of values in the column

UPC Number	~100 billion UPC numbers	• 27790 96306 32 • 49225 98451 10	Yes	Yes. The number of distinct values in the specified range must be greater than or equal to the number of values in the column
Social Security Number Formatted	~718 million unique US Social Security Numbers	• 058-20 -6521 • 337-71 -7394	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column
Social Security Number	~718 million unique US Social Security Numbers	• 08771 9720 • 17747 0222	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column
Social Insurance Number Formatted	~1 billion unique Canadian Social Insurance Numbers	• 861-11 5-129 • 186-91 5-534	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column

Social Insurance Number	~1 billion unique Canadian Social Insurance Numbers	• 300580727 • 780632832	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column
National Insurance Number Formatted	Generates unique UK National Insurance Numbers	• BH 85 53 00 D • RX 81 61 33 B	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column
MasterCard Credit Card Number	~10 billion unique MasterCard credit card number	• 5353780646712850 • 5259381925706856	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column
ISBN (Thirteen Digit) Formatted	~2 billion unique ISBN numbers	• 978-3-31553 3-40-1 • 979-3-34300 5-63-4	Yes	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column

ISBN (Thirteen Digit)	~2 billion unique ISBN numbers	• 979274629491 1 • 978907535890 2	Yes	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column
ISBN (Ten Digit) Formatted	~1 billion unique ISBN numbers	• 8-13-132012-X • 7-69-626483-4	Yes	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column
ISBN (Ten Digit)	~1 billion unique ISBN numbers	• 9890454203 • 7851091314	Yes	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column
Generic Credit Card Number Formatted	~10 billion unique generic credit card numbers	• 3647-1229-3400-9783 • 3088-2317-5970-6344	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column

Generic Credit Card Number	~10 billion unique generic credit card numbers	• 5595470359501553 • 3826796343439260	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column
Finnish Social Security Numbers	~2.4 billion unique Finnish Social Security Numbers	• 080309A635C • 290558-119J	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column
Discover Card Credit Card Number	~10 billion unique Discover Card credit card numbers	• 6011793517281170 • 6011061023216399	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column

Auto Mask Format	Masking by scrambling the characters and numbers. This format preserves the input length, position of the characters and numbers, case of the character (upper or lower), and special characters in the input.	• Input: ABCD_343_ddg • Output: FHDT_657_tte	Yes	Not applicable
American Express Credit Card Number	~10 billion unique American Express credit card numbers	• 379133064473287 5 • 348187001742431 6	Yes, if the generated masked values passes the post processing function validation	Yes, if the number of distinct values that can be generated by the LMF is greater than the number of distinct values in the column

> **Note:**
>
> The **Deterministic** and **Reversible** format characteristic values for all Masking Formats in the table above are "No".

For more information on Masking Format Characteristics, see: Characteristics of Masking Formats

Credit Card Numbers

Out of the box, the format library provides many different formats for credit cards. The credit card numbers generated by these formats pass the standard credit card validation tests by the applications, thereby making them appear like valid credit card numbers.

Some of the masking formats provided include coverage for credit card types such as:

- Mastercard
- Visa
- American Express
- Discover

- Generic credit card numbers (can cover all card types)

You may want to use different styles for storing credit card numbers, such as:

- Pure numbers
- 'Space' for every four digits
- 'Hyphen' (-) for every four digits, and so forth

To implement the masked values in a certain format style, you can set the `DM_CC_FORMAT` variable of the `DM_FMTLIB` package.

United States Social Security Numbers

Out of the box, you can generate valid U.S. Social Security (SSN) numbers. These SSNs pass the normal application tests of a valid SSN.

You can affect the format style by setting the `DM_SSN_FORMAT` variable of the `DM_FMTLIB` package. For example, if you set this variable to '-', the typical social security number would appear as '123-45-6789'.

ISBN Numbers

Using the format library, you can generate either 10-digit or 13-digit ISBN numbers. These numbers adhere to standard ISBN number validation tests. All of these ISBN numbers are random in nature. Similar to other format definitions, you can affect the "style" of the ISBN format by setting values to `DM_ISBN_FORMAT`.

UPC Numbers

Using the format library, you can generate valid UPC numbers. They adhere to standard tests for valid UPC numbers. You can affect the formatting style by setting the `DM_UPC_FORMAT` value of the `DM_FMTLIB` package.

Canadian Social Insurance Numbers

Using the format library, you can generate valid Canadian Social Insurance Numbers (SINs). These numbers adhere to standard tests of Canadian SINs. You can affect the formatting style by setting the `DM_CN_SIN_FORMAT` value of the `DM_FMTLIB` package.

North American Phone Numbers

Out of the box, the format library provides various possible U.S. and Canadian phone numbers. These are valid, realistic looking numbers that can pass standard phone number validation tests employed by applications. You can generate the following types of numbers:

- Any North American phone numbers
- Any Canadian phone number
- Any U.S.A. phone number

UK National Insurance Numbers

Using the format library, you can generate valid unique random UK National Insurance Numbers (NINs). These numbers adhere to standard tests of UK NINs. A typical national insurance number would appear as 'GR 12 56 34 RS'.

Auto Mask

This format scrambles characters and numbers into masked characters and numbers and while retaining the format and length of the data, including special characters; for example, 'ABCD_343-ddg' masked as 'FHDT_657-tte'.

Format Entry Options to Customize Masking Format

Format entry options are as follows:

1. **Array List**
   Accepts a list of values as input and maps each value in the list to a value in the input column. The number of values in the list should be greater than or equal to the number of distinct values in the masked column. The values in the user-provided list are ordered randomly before mapping them to the original column values. For example, if the original column contains values [10,20,30,40,50] and the Array List specified by the user is [99,100,101,102,103], the first masking run could produce the mapping [10,101], [20,103], [30,100], [40,99], [50,102] and a different masking run can produce [10,100], [20,99], [30,101], [40,102], [50,103].

   A mapping table is created. The CTAS that creates the mapping table queries from:

   - The original table to fetch the column values being masked and a row number for each column value. The row number is derived from the Oracle-supplied `ROW_NUMBER` function.
   - The user-passed list of values — values in the user-passed array list are converted into a table-like record set using the `SQL TABLE` function. A row number is also retrieved corresponding to each value in the record set. The row number is derived from the `ROWNUM` pseudo column. The values in the record set are randomly ordered using `DBMS_RANDOM.VALUE` function.
   - The mapping table CTAS then joins the row numbers in both sub-queries to map the original value (from sub-query in step 1 above) and a value from the user-list (sub-query in step 2 above). Multiple executions of the CTAS will create a mapping table with different original-masked value mappings because of the random ordering of the user list in step 2.

   > **Note:**
   >
   > If masking column is either Date or Timestamp, user can mask the column using Array List format. The Date/Timestamp formats supported are
   >
   > `yyyy-MM-dd`
   >
   > and
   >
   > `yyyy-MM-DDTHH:mm:ssTZD`

   For example: "2015-01-18" and "2015-01-18T03:25:46Z".

2. **Delete**
   Deletes a row based on a condition. If the condition matches, then the row is deleted on the target. A mapping table is created. The "`DELETE_VAL`" column in the mapping table is set to 1 for rows that are candidates to be deleted. For example, we are masking the

SALARY column and the masking definition has conditions on the EMPID column and formats defined as:

```
EMPID < 100
      DELETE
EMPID < 200
        RANDOM NUMBERS [Start Value:1 End Value:100]
DEFAULT
        PRESERVE ORIGINAL DATA
```

The mapping table will have the DELETE_VAL column set to 1 for SALARY rows with EMPID < 100. DELETE_VAL for all other rows is set to 0. The final masking CTAS SQL which joins the original table and the mapping table to create the masked table filters out rows with DELETE_VAL set to 1. Therefore, the rows in the original table that match the join condition are effectively “deleted”.

3. **Encrypt**
   The Encrypt masking format encrypts column data using Triple DES (3DES). The format of the column data after encryption is similar to that of the original values. For example, if you mask nine-digit numbers, the encrypted values also have nine digits. Encrypt is a deterministic and the only reversible masking format. It is helpful when businesses need to mask and send their data to a third party for analysis, reporting, or any other business processing purpose. After the processed data is received from the third party, the original data can be recovered (decrypted) using the same seed value that was used to encrypt the data.

   You provide a regular expression to mask character or numeric type column. The specified regular expression must match all the original values in the column. If a value does not match the regular expression exactly, the masking format may no longer produce one-to-one mapping. Therefore, to ensure uniqueness, all the values must match the regular expression. The encrypted values also match the specified regular expression. Encrypt supports encryption of strings of fixed widths. It supports a subset of the regular expression language and does not support * or + syntax in regular expressions.

   > **Note:**
   >
   > The maximum value of a user provided regular expression cannot exceed 64 bits.

   You also provide a seed value that is used to generate a key for encryption and decryption. The seed value has to be provided at the time of submitting a data masking job. It can be any string containing alphanumeric characters.

   If your masking definition has a sensitive column using Encrypt, you are shown the decrypt option while submitting a data masking job. Choosing this option, you can decrypt the encrypted column values by providing the same seed used to encrypt the sensitive column.

4. **Fixed Number**
   This format does not use a lookup or a mapping table. It assigns a fixed number value to a string/number column.

   The type of column applicable to this entry is a NUMBER column or a STRING column. For example, if you mask a column that has a social security number, one of the entries can be Fixed Number '900'. This format is combinable.

5. **Fixed String**
   This format does not use a lookup or a mapping table. It assigns a fixed string value to a string column. For example, if you mask a column that has a License Plate Number, one of the entries can be Fixed String 'CA'. This format is combinable.

6. **Null Value**
   Masks the column with a value of NULL. It does not use a lookup or a mapping table.

7. **Preserve Original Data**
   Preserves the original column value. Used in conditional masking with a combination of other formats where only a subset of values needs to be masked based on a condition.

8. **Random Dates**
   The format creates a mapping table. The mapping table CTAS contains code to generate random dates within a user specified date range. A random date is generated using the following logic:

   ```
   TO_DATE(start_date','YYYY-DD-MM HH24:MI:SS') +
   mask_util.genrnd(0, <#of days between the specified date range>)
   ```

   This format is combinable at the end after data range.

9. **Random Decimal Numbers**
   If used as part of a mixed random string, these have limited usage for generating unique values. This masking format generates unique values within the specified range. For example, a starting value of 5.5 and ending value of 9.99 generates a decimal number ranging from 5.5 to 9.99, both inclusive. This masking format is combinable.

10. **Random Digits**
    This format generates unique values within the specified range. For example, for a random digit with a length of [5,5], an integer between [0, 99999] is randomly generated, left padded with '0's to satisfy the length and uniqueness requirement. This is a complementary type of random number, which will not be padded. When using random digits, the random digit pads to the appropriate length in a string. It does not pad when used for a number column. This format is combinable.

    Data masking ensures that the generated values are unique, but if you do not specify enough digits, you could run out of unique values in that range.

11. **Random Numbers**

    If used as part of a mixed random string, these have limited usage for generating unique values. This format generates unique values within the specified range. For example, a starting value of 100 and ending value of 200 generates an integer number ranging from 100 to 200, both inclusive. This format is combinable.

12. **Random Strings**

    This format generates unique values within the specified range. For example, a starting length of 2 and ending length of 6 generates a random string of 2 - 6 characters in length. This format is combinable.

13. **Regular Expression**

    The format uses a lookup table. No mapping table is created. The PL/SQL function that implements the format is invoked directly from the final CTAS which creates the masked table. The lookup table has two columns to store the regular expression and the replacement value specified by the user. The SQL `REGEXP_REPLACE` function is used to implement this format.

The function has the signature:

```
regexp_replace(column_value, regex, replacement_val);
```

For example, phone numbers in the format

```
 nnn.nnn.nnnn
```

can be masked using a regex

```
[1-9]{3}[.][0-9]{3}[.][0-9]{4}
```

with a replacement value

```
***.***.****
```

The format invokes the regexp_replace for each regexp-replacement value pair. If the phone number column was masked using regular expression:

```
EMPID < 100
  Regular Expression      Regex: [1-9]{3}[.][0-9]{3}[.][0-9]{4}
Replacement Value: 999.444.5555
  Regular Expression      Regex: [9]{3}[.][4]{3}[.][5]{4}
Replacement Value: ***.***.****
```

Each column value matching the first regular expression is first replaced with `999.444.5555`, this value then matches the second regular expression and is replaced with `***.***.****`. The example is not a real world use case. The behavior probably is a side effect of how the format is implemented, the real use case of specifying multiple regular expression formats to mask a column is to handle cases when the data in the column could match multiple regular expressions. For example:

```
EMPID < 100
  Regular Expression      Regex: [1-9]{3}[.][0-9]{3}[.][0-9]{4}
Replacement Value: ***.***.****
  Regular Expression      Regex: [1-9]{3}[.][0-9]{3}[.][0-9]{3}
Replacement Value: ***.***.***
```

can be used to mask column values that store 10 digit - nnn.nnn.nnnn - or 9 digit - nnn.nnn.nnn - phone numbers.

14. **Shuffle**

This format does not use a lookup or a mapping table. The final CTAS which creates the mapping table includes a sub query to order the column contents randomly using the DBMS_RANDOM.VALUE function. If shuffle is used with a grouping column, the `PARTITION` clause is used to partition by the grouping column and the column is ordered randomly within each partition. The implementation is similar to that of Array List and Table Column. The random ordering for the shuffle format occurs on the column being “shuffle masked”, whereas in Array List, it is on the user-passed list, and in Table Column, the ordering is on the user-specified column.

15. **Substitute**

    The format creates a mapping table. It uses a user specified “substitution” table as a source for masked values. The format uses the Oracle supplied hash based partitioning function ORA_HASH to map a column value to its mask value in a lookup (substitution) table. Processing involves querying the substitution table to get a count of distinct values in the mask column. This count - `n` - is then used as the `max_bucket` parameter of `ORA_HASH` to hash the original column values into `n` buckets. For example, if we are masking `EMPLOYEE.SALARY` and using `SUBST.SUB_COL` column as the substitution column, we first get the count of distinct values in `SUB_COL`. The mapping table CTAS then queries:

    a. The original column, `EMPLOYEE.SALARY`

    b. The user provided substitution table to fetch all the distinct values in `SUBST.SUB_COL` and also fetches the `ROWNUM` associated with each row

    The CTAS SQL then joins 1 and 2 using `ORA_HASH` and equating its output to the `ROWNUM` from step 2. The `SELECT` part of the CTAS SQL is listed below. `max_bckt` is the count of distinct values in the substitution column `SUBST.SUB_COL`:

```
select s.orig_val,
       a0.new_val
from ( select orig_val
       from (select "SALARY" orig_val
             from "TESTU"."EMPLOYEE")
       group by orig_val) s,
     (select rownum rn,
             SUB_COL new_val
      from (select distinct SUB_COL
            from TESTU.SUBST
            order by SUB_COL)) a0
 where ora_hash(s.orig_val, max_bckt, seed)+1 = a0.rn
```

16. **SQL Expression**

    The format does not create a mapping table. It allows a user to use a SQL Expression for masking a column. Data masking uses this expression to generate masked values to replace the original values. The expression is invoked directly from the masking CTAS SQL. The SQL Expression can consist of one or more values, operators, and SQL functions that evaluates to a value. It can also contain substitution columns (columns from the same table as the masked column). Some examples of valid expressions:

```
1.    dbms_random.string('u', 8) || '@company.com'
2.    %first_name% || '.' || %last_name% || '@company.com'
```

17. **Substring**

    The format creates a mapping table. The mapping table CTAS invokes the Oracle SUBSTR function on the input column. The format accepts a start position and length as input, extracts that data from the input column using `SUBSTR`, and uses that as a mask value.

18. **Table Column**

    The format creates a mapping table. The format maps original column values to column values in a user specified table. The processing is similar to the array list format. The values in the user specified table are randomly ordered using `DBMS_RANDOM.VALUE` before mapping each value to the original column. Unlike the Substitute format, the format is not deterministic since the substitution column is randomly ordered.

19. **Truncate**

    The format truncates all rows in a table. It does not create a mapping table. If one of the columns in a table is masked using this format, so no other mask formats can be specified for any of the other columns.

20. **User Defined Function**

    The format creates a mapping table. The return value of the user defined function is used to mask the column. The function is invoked as part of the mapping table CTAS. The function has a fixed signature:

```
 function userdef_func(rowid varchar2, col_name varchar2, orig_val
varchar2) returns varchar2;
```

21. **Post Processing Function**

    Optionally, some formats can also include a post-processing function that takes the output after executing all the format entries, runs the function and produces the final masked value.

    The format allows users to use a custom function to process column values after they are masked using standard data masking formats. For example, the `SALARY` column can be masked with a SQL expression first, and a post processing function can be applied on the masked values to add a currency symbol, like '$'. The function has a fixed signature:

```
function post_proc_func(rowid varchar2, column_name varchar2, mask_value
varchar2) returns varchar2;
```

    The `ROWID` input allows a user to fetch column values from the masked table. The function could use these values to mask the input column value, basically to transform the column further after a standard format is applied on the column. This format creates a mapping table. The post processing function gets invoked as part of the mapping table `CTAS SQL`. The input to the mask_value argument of the function is the masked value of the original column. For example, say we are masking the `SALARY` column and the mask definition has conditions on the `EMPID` column and formats are defined this way:

```
EMPID < 100
      RANDOM NUMBERS [START:100000 END: 10000000]
      POST PROCESSING FUNCTION ppf
EMPID < 200
        FIXED NUMBER 100000
DEFAULT
        PRESERVE ORIGINAL DATA
```

> **Note:**
>
> Refer to **Frequently Asked Questions** for recommended customize masking formats using above entry options.

Deterministic Masking Using the Substitute Format

You may occasionally need to consistently mask multiple, distinct databases. For instance, if you run HR, payroll, and benefits that have an employee ID concept on three separate

databases, the concept may be consistent for all of these databases, in that an employee's ID can be selected to retrieve the employee's HR, payroll, or benefits information. Based on this premise, if you were to mask the employee's ID because it actually contains his/her social security number, you would have to mask this consistently across all three databases.

Deterministic masking provides a solution for this problem. You can use the Substitute format to mask employee ID column(s) in all three databases. The Substitute format uses a table of values from which to substitute the original value with a mask value. As long as this table of values does not change, the mask is deterministic or consistent across the three databases.

Using the Shuffle Format

A shuffle format is available, but it does not keep the original data distribution if the column values are not unique or when conditional masking is used. For example, in an Original Table with two columns, EmpName and Salary, the Salary column might have distinct values like 10, 90, and 20.

Original Table

EmpName	Salary
A	10
B	90
C	10
D	10
E	90
F	20

If you mask the Salary column with this format, each of the original values is replaced with one of the values from this set. Assume that the shuffle format replaces 10 with 20, 90 with 10, and 20 with 90.

Mapping Table (Non-preservation)

EmpName	Salary
10	20
90	10
20	90

The result is a shuffled Salary column as shown in the Masked Table, but the data distribution is changed. While the value 10 occurs three times in the Salary column of the Original Table, it occurs only twice in the Masked Table.

Masked Table (Non-preservation)

EmpName	Salary
A	20
B	10
C	20
D	20
E	10
F	90

If the salary values had been unique, the format would have maintained data distribution.

Using Group Shuffle

Group shuffle enables you to perform a shuffle within discrete units, or groups, where there is a relationship among the members of the group. Consider the case of shuffling the salaries of employees. This table illustrates the group shuffle mechanism, where employees are categorized as managers (M) or workers (W), and salaries are shuffled within job category.

Employee	Job Category	Salary	Shuffled Salary
Alice	M	90	88
Bill	M	88	90
Carol	W	72	70
Denise	W	57	45
Eddie	W	70	57
Frank	W	45	72

Using Conditional Masking

To demonstrate how conditional masking can handle duplicate values, add another job category, assistant (A), where the employee in this category, George, earns the same as Frank. Assume the following conditions:

- If job category is M, replace salary with a random number between 1 and 10.
- If job category is W, set salary to a fixed number (01).
- Default is to preserve the existing value.

Applying these conditions results in the masked values shown in the following table:

Employee	Job Category	Salary	Conditional Result
Alice	M	90	5
Bill	M	88	7
Carol	W	72	01
Denise	W	57	01
Eddie	W	70	01
Frank	W	45	01
George	A	45	45

Conditional masking works when there are duplicate values provided there are no dependent columns or foreign keys. If either of these is present, a "bleeding condition" results in the first of two duplicate values becoming the value of the second. So, in the example, Frank's salary is not preserved, but becomes 01.

Providing User-defined and Post-processing Functions

If desired, you can provide user-defined and post-processing functions on the Create Masking Format page. A user-defined choice is available in the Custom Format Entry list, and a post-processing function field is available at the bottom of the page.

- **User-defined functions**

  To provide a user-defined function, select **User Defined Function** from the drop-down list to access the input fields.

  A user-defined function passes in the original value as input, and returns a mask value. The data type and uniqueness of the output values must be compatible with the original output values. Otherwise, a failure occurs when the job runs. Combinable, a user-defined function is a PL/SQL function that can be invoked in a `SELECT` statement. Its signature is returned as:

```
Function udf_func (rowid varchar2, column_name varchar2, original_value varchar2)
return varchar2;
```

  - `rowid` is the min (`rowid`) of the rows that contain the value `original_value` 3rd argument.
  - `column_name` is the name of the column being masked.
  - `original_value` is the value being masked.

  That is, it accepts the original value as an input string, and returns the mask value.

  Both the input and output values are varchar2. For instance, a user-defined function to mask a number could receive 100 as input, the string representation of the number 100, and return 99, the string representation of the number 99. Values are cast appropriately when inserting to the table. If the value is not castable, masking fails.

- **Post-processing functions**

  To provide a post-processing function, enter it in the **Post Processing Function** field.

  A post-processing function has the same signature as a user-defined function, but passes in the mask value the masking engine generates, and returns the mask value that should be used for masking, as shown in the following example:

```
Function post_proc_udf_func (rowid varchar2, column_name varchar2, mask_value
varchar2) return varchar2;
```

  - `rowid` is the min (rowid) of the rows that contain the value `mask_value`.
  - `column_name` is the name of the column being masked.
  - `mask_value` is the value being masked.

Patterns of Format Definitions

All of the predefined format definitions adhere to these typical patterns:

- Generate a random number or random digits.
- Perform post-processing on the above-generated value to ensure that the final result is a valid, realistic value.

For example, a valid credit card number must pass Luhn's check. That is, the last digit of any credit card number is a checksum digit, which is always computed. Also, the first few digits indicate the card type (MasterCard, Amex, Visa, and so forth). Consequently, the format definition of a credit card would be as follows:

- Generate random and unique 10-digit numbers.
- Using a post-processing function, transform the values above to a proper credit card number by adding well known card type prefixes and computing the checksum digit.

This format is capable of generating 10 billion unique credit card numbers.

Masking Definitions

A masking definition defines a data masking operation to be implemented on one or more tables in a database. Masking definitions associate table columns with masking formats to use for masking the data.

Prerequisites

Before creating a masking definition, review requirements and advisory information here: **Prerequisites**

- Ensure the format you select does not violate check constraints and does not break any applications that use the data.
- For triggers and PL/SQL packages, data masking recompiles the object.
- Exercise caution when masking partitioned tables, especially if you are masking the partition key. In this circumstance, the row may move to another partition.
- Data Masking does not support clustered tables, masking information in object tables, XML tables, and virtual columns. Relational tables are supported for masking.
- If objects are layered on top of a table such as views, materialized views, and PL/SQL packages, they are recompiled to be valid.

If you plan to mask a test system intended for evaluating performance, try to preserve the production statistics and SQL profiles after masking by adding a pre-masking script to export the SQL profiles and statistics to a temporary table, then restoring after masking completes.

Creating a Masking Definition

1. From the Targets menu, select **Databases**, then click the **Security** menu and navigate to **Data Masking and Subsetting** then **Data Masking**. Alternatively, you can right-click on your specific database and go to **Security** then **Data Masking and Subsetting** then **Data Masking**.

   The Masking Definitions page appears, where you can create new or manage existing masking definitions.

2. Click **Create** to go to the Create Masking Definition page.

   A masking definition includes information regarding table columns and the format for each column. You can choose which columns to mask, leaving the remaining columns intact.

3. Provide required details, including Name, Application Data Models, Associated Database, and Database Named Credentials.

4. Click **Next** to go to the Sensitive Columns and Masking Formats page, where you can choose which sensitive columns in the ADM you want to mask.

   The sensitive columns appear in the first section whereas columns added to Masking Definition appear below in the next section.

5. Select the desired sensitive columns you want to mask.

   Either click **Define Format and Add** to define the format for the column now or you can choose **Add Column** to add the column to the Masking Definition and define the format later.

   Also, Define Format and Add features can save significant time. When you select multiple columns to add that have the same data type, you do not need to define the format for each column one by one. For instance, if you search for Social Security numbers (SSN) and the search yields 100 SSN columns, you could select them all, then click **Define Format and Add** to import the SSN format for all of them.

   Optionally, below are the other masking options available:

   - Group Masking: If you want to mask selected columns as a group, enable **Group Masking**. The columns that you want to mask as a group must all be from the same

table. Any formats defined while this checkbox is selected will be applied to all the selected columns as a group.

- Select all filtered rows: Enable the **Select all filtered rows** check box if you want to mask more than one column together rather than separately.
  After you define the masking format for the grouped columns and return to this page, the Column Group in the table shows an identical number for each entry row in the table for all members of the group. For example, if you have defined your first group containing four columns, each of the four entries in this page will show a number 1 in the Column Group column. This helps you to distinguish which columns belong to which column groups.

6. Do one of the following:
   - If you chose **Add Column** in the previous step and do not wish to provide the masking format for now:
     You will eventually need to define the column's masking format. When you are ready to provide the masking format, go to the Actions menu for the Masking Definition and select **Edit** to define the format. Read the following instructions to define the formats.
   - If you chose **Define Format and Add** in the previous step:
     The Define Format and Add page appears, where you can define the format for all selected columns, as explained below:
     - Choose the masking format either from the predefined masking formats drop-down or customize your own. Click **Import**.
     - You can provide one or more format entries.
     - You can generate Sample Data by clicking **Generate**.
     - When you have finished formatting the column, click **Add**. The sensitive columns you selected and assigned masking formats to earlier now appear in the next section. Click **Next**.
7. Review the Data Masking options. Before clicking **Next**, you can also provide the pre- and post-masking script. Refer to **Selecting Data Masking Advanced Options** for more information regarding Pre Mask Scripts and Post Mask Scripts.
8. Review the final page and click **Create** to return to the Masking Definitions page. The new masking definition will appear at the top of the table.

Selecting Data Masking Advanced Options

Data Masking Options

The data masking options include:

- Disable redo log generation during masking:

  Masking disables redo logging and flashback logging to purge any original unmasked data from logs. However, in certain circumstances when you only want to test masking, roll back changes, and retry with more mask columns, it is easier to uncheck this box and use a flashback database to retrieve the old unmasked data after it has been masked. You can use Enterprise Manager to flashback a database.

> **Note:**
>
> Disabling this option compromises security. You must ensure this option is enabled in the final mask performed on the copy of the production database.

- Refresh statistics after masking:

  If you have already enabled statistics collection and would like to use special options when collecting statistics, such as histograms or different sampling percentages, it is beneficial to turn off this option to disable default statistics collection and run your own statistics collection jobs.

- Drop temporary tables created during masking:

  Masking creates temporary tables that map the original sensitive data values to mask values. In some cases, you may want to preserve this information to track how masking changed your data. Note that doing so compromises security. These tables must be dropped before the database is available for unprivileged users.

- Decrypt encrypted columns:

  This option decrypts columns that were previously masked using Encrypt format. To decrypt a previously encrypted column, the seed value must be the same as the value used to encrypt.

  Decrypt only recovers the original value if the original format used for the encryption matches the original value. If the originally encrypted value did not conform to the specified regular expression, when decrypted, the encrypted value cannot reproduce the original value.

- Use parallel execution when possible:

  Oracle Database can make parallel various SQL operations that can significantly improve their performance. Data Masking uses this feature when you select this option. You can enable Oracle Database to automatically determine the degree of parallelism, or you can specify a value. For more information about using parallel execution and the degree of parallelism, see the Oracle Database Data Warehousing Guide.

- Recompile invalid dependent objects after masking:

  The masking process re-creates the table to be masked and as a consequence, all existing dependent objects (packages, procedures, functions, MViews, Views, Triggers) become invalid. You can specify that the masking process recompile these invalid objects after creating the table, by selecting the check box. Otherwise, invalid objects are not recompiled using `utl_comp` procedures at the end of masking.

  If you choose this option, indicate whether to use serial or parallel execution. You can enable Oracle Database to automatically determine the degree, or you can specify a value. For more information about using parallel execution and the degree of parallelism, see the Oracle Database Data Warehousing Guide.

Random Number Generator

The random number generation options include:

- Favor Speed:

  The `DBMS_RANDOM` package is used for random number generation.

- Favor Security:

The `DBMS_CRYPTO` package is used for random number generation. Additionally, if you use the Substitute format, a seed value is required when you schedule the masking job or database clone job.

Pre Mask and Post Mask Scripts

Pre and post masking scripts are free flow scripts where a user can execute SQL statements and PL/SQL stored program units (procedures, functions, packages, anonymous block) before and after masking as required.

Use the Pre Mask Script text box to specify any user-specified SQL script that must run before masking starts.

Use the Post Mask Script text box to specify any user-specified SQL script that must run after masking completes.

Generating Masking Script

To schedule a script generation job:

1. Select the masking definition to generate a script for, then click on the **Actions** button and select **Manage Masking Script** then **Generate Masking Script**.
2. Select a Data Masking option:
   - **In-Database Masking** – to replace sensitive data in-place with masked data on a specified database (usually copied from production).
   - **In-Export Masking** – to export masked data from the specified source database (usually production) using Oracle Data Pump.
   - You can choose both options; that is, a script to mask the database directly and a script to create a masked dump.
3. Select an Associated Database and Database Named Credentials from the drop-down list.
4. Change the default job name to something meaningful, if desired, and provide an optional job description.
5. Specify to start the job immediately or at a later specified date and time, then click **Generate**.
   A message confirms that the Generate script job has been submitted successfully. Click on the **Refresh** button to see the latest job status.

Scheduling Masking Job

To set up the data masking job and schedule its execution:

1. Select the masking definition for which a script has been generated, then click **Actions** and select **Schedule Masking**.
2. Select a Data Masking Option radio button to indicate your preference:
   - **In-Database Masking** – Replaces sensitive data directly in the specified database (usually a copy from production) with masked data. This option is intended for non-production environments only.

> **Note:**
>
> You must enable the check box indicating that the selected database is not a production database in order to proceed.

- **In-Export Masking** – Exports masked data from the specified source database (usually production) using Oracle Data Pump. This option is safe for production environments, as it does not modify customer data. However, it creates temporary tables that will be dropped once the masking operation is completed. Your selection will update the checkbox text below the radio buttons and other areas on the page.

3. From the dropdown menus, select the required Associated Database, Database Named Credentials, and Host Named Credentials.
4. Choose a Tablespace for Temporary Objects. There are three options:
   a. Default Tablespace
   b. Temporary Tablespace
   c. Custom Tablespace

   Select the appropriate option based on your needs.
5. Enter a Job Name or accept the default pre-filled value.
6. Specify whether to start the job immediately or at a later date and time. Then, click **Next**.
7. If you selected In-Database Masking in Step 2:
   a. Enter the script file location and name (pre-filled), and then click **Submit**.
8. If you selected In-Export Masking in Step 2:
   a. Enter the Export Filename (pre-filled).
   b. Specify a directory to save the mask dump files to by selecting an option from **Type of Export Directory**. The available options are a Database Directory, Custom Directory or an External Directory.
   c. Select whether to export the masked data only or the entire database along with the masked data through **Data to Export**. Exporting the entire database might significantly increase the overall job execution time.
   d. Enter the Maximum Export File Size (MB) (pre-filled). Optionally, choose to **Enable Export File Compression** and/or **Enable Export File Encryption** according to your needs. If Export File Encryption is enabled, enter and confirm an encryption password.
   e. Specify the Maximum Number of Threads (pre-filled).
   f. Generate Data Dump Logs is enabled by default but you can choose to turn it off as necessary.
   g. Click **Submit**.
9. A message confirms that the job has been scheduled and you can use the refresh button above the table to update the job status.

5

Data Subsetting

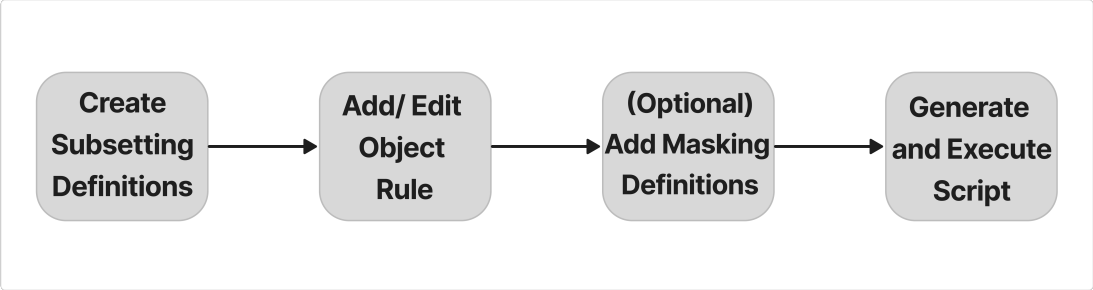
This chapter provides conceptual information about the components that comprise Oracle Data Subsetting and procedural information about performing the task sequence, including methods for defining subsets. Data Subsetting presupposes that you have created an Application Data Model (ADM).

> **Note:**
>
> You must have the Oracle Data Masking and Subsetting Pack license to use Data Subsetting features.

Data Subsetting Workflow

This flowchart covers the steps to perform data subsetting. Please note that several additional associated features, such as Importing/Exporting Subset Templates, are not highlighted here.

Figure 5-1 Data Subsetting Workflow



This diagram illustrates the data subsetting workflow in Oracle Data Masking and Subsetting (DMS). The process begins with creating subsetting definitions to identify and extract specific data sets. Next, users can add or edit object rules to define how the data is filtered. Optionally, masking definitions can be added to ensure data anonymization during the subsetting process. The final step involves generating and executing the script to apply the defined rules and extract the subsetted data. This streamlined workflow enables users to securely subset and if needed, mask data in a single process.

Data Subsetting Definitions

The procedure described in this section enables you to create a subset database, after which you can perform other tasks, such as editing the properties of the subset definition or exporting a subset definition.

The interface also allows you to perform inline, or at the source, masking while creating the subset definition.

Before proceeding, review requirements and advisory information here: **Prerequisites**

Creating a Subsetting Definition

To create a data subset definition:

1. From the main menu, navigate to **Targets**, then select **Databases**. Next, go to **Security** then **Data Masking and Subsetting** and choose **Data Subsetting**.
2. Open the **Actions** menu in the Data Subsetting Definitions page, then select **Create**, or just click the **Create** icon.
3. Define the data subset definition properties:
   a. Provide the requisite information in the General pop-up that appears, then click **Continue**.

      You can select any source database associated with the Application Data Model.

      If you are performing masking within the subset definition, you must select the same ADM and target used in creating the masking definition.

   b. Provide a job name, credentials, and specify a schedule in the Schedule Application Detail Collection pop-up that appears, then click **Submit**.

      If you want to use new credentials, choose the **New Credentials** option. Otherwise, choose the **Preferred Credentials** or **Named Credentials** option.

   The space estimate collection job runs, and then displays the Data Subset Definitions page. Your definition appears in the table, and the Most Recent Job Status column should indicate Scheduled, Running, or Succeeded, depending on the schedule option selected and time required to complete the job.

   Add Object (Subsetting) Rule
4. Select the definition within the table, then click the **Edit** icon at the top, or go to the **Actions** menu and choose **Edit**.

   The Database Login page appears.
5. Select either **Named Credentials** or **New Credentials** if you have not already set preferred credentials, then click **Login**.
6. In the Applications subpage of the Edit page, move applications from the Available list to the Selected list as follows:
   - If you intend only to mask the data (no subsetting), select all applications.
   - If you intend only to subset the data (no masking), select specific applications as appropriate.
   - If you intend both to subset and mask the data, the applications selected must include those that the masking definitions require.

   The names of application suites, applications, or application modules are maintained in the Application Data Model.
7. Click the **Object Rules** tab.

> **Note:**
>
> If you are masking only, set the **Default Object Rows** option to include all rows and skip to Step 13. The Column Mask Rules tab, Rule Parameters tab, and additional features on the **Object Rules** tab pertain specifically to subsetting.

You can add rules here to define the data to include in the subset.

8. Select **Actions**, then **Create** to display the Object Rule pop-up, or just click the **Create** icon.
   a. Select the application for which you want to provide a rule. Associate the rule with all objects, a specific object, or a specific type of object.
   b. If you select **All Objects**, in the Rows to Include section, select all rows or some rows by specifying a percentage portion of the rows.
   c. If you select **Specified** as the object type, the tables from the selected Application appear in the drop-down list.

      In the Rows to Include section, select all rows, or some rows by specifying a percentage portion of the rows. For finer granularity, you could specify a Where clause, such as where region_id=6.

      If the selected table is partitioned, click **Add/Remove Partitions** to choose the partitions and sub-partitions from which the objects must be included in the subset.
   d. In the Include Related Rows section, do one of the following:
      - Select **Ancestor and Descendant Objects**

        This rule impacts the parent and child columns, and ensures that referential integrity is maintained, and that child columns are also selected as part of the subset.
      - Select **Ancestor Objects Only**

        This option will be disabled if the global rule is set to “Include All Rows”, and will be enabled if it is set to “Include No Rows”. This rule only impacts the parent columns, and ensures that referential integrity is maintained.
      - Select **Descendant Objects Only**

        This option will be disabled if the global rule is set to “Include No Rows”, and will be enabled if it is set to “Include All Rows”.

      If you disable the Include Related Rows check box, referential integrity might not be maintained. However, you can define additional rules on the related tables to restore the referential integrity. You can disable this check box whether or not you specify a Where clause.
   e. If you want to specify a Where clause, go to the next step. Otherwise, skip to Step 9.
   f. Provide a rule parameter, if desired, for the clause.

      For instance, if you specify a particular value for an employee ID as employee_id=:emp_id, you could enter query values for the default of 100:
      - Select the **Rows Where** button and enter employee_id=:emp_id.
      - Click **OK** to save the rule and return to the **Object Rules** tab.

If this is a new rule, a warning appears stating that "Rule parameters corresponding to the bind variables 'emp_id' should be created before generating subset."

- Select the table rule, click the **Rule Parameters** tab, then click **Create**.

  The Rule Parameter Properties pop-up appears.

- Enter emp_id for the Name and 100 for the Value.

> **Note:**
>
> The colon (:) preceding emp_id is only present in the Where clause, and not required when creating a new rule parameter.

- Click **OK** to save the properties, which now appear in the Rule Parameters tab.
- Skip to Step 10.

9. Click **OK** to save the rule and return to the **Object Rules** tab.

   The new rule is displayed in the list. The related objects are displayed in the table below. Related rows from the objects are included in the subset to provide referential integrity in the subset database.

10. In the "Related Objects" section of the **Object Rules** tab, you can manage the size of the subset by controlling the levels of ancestors and descendants within the subset. Notice that each node in the table has a check box. By default, all nodes are included in the subset, as indicated by the check mark. Deselect the check box to exclude a node from the subset. The deselection option is disabled for parent rows (the join columns to the right identify parent and child rows). In addition, you can make these other refinements to subset content:

    - Click **Allow Excluding Parent Objects**. This enables the check marks that were grayed out. You can now selectively exclude parent rows from the subset by deselecting the check box.
    - Select a node within the table and click **Add Descendants** to include related rows. In the dialog that opens, make appropriate selections and click **OK**.

    As you make these refinements, columns on the right reflect the effect on space estimates for the subset. The "Related Objects" section also denotes the processing order of the ancestor and descendant tables, including the detailed impact of including each object. When you are done with the refinements, go to the Space Estimates tab to see a finer granularity of the impact on the overall size of the subset.

11. In the Default Object Rows section of the **Object Rules** tab, choose whether you want to include or exclude the objects not affected by the defined rules in the subset.

    When you select the **Include All Rows** option, all of the rows for the object are selected as part of the subset.

    This is a global rule and applies to the entire subset. You can only select the **Include All Rows** option when all of the rules have a scope of None. A scope of None is established when you uncheck the Include Related Rows option in the Object Rule pop-up.

> **Note:**
>
> For a subset definition that has column rules (see Step 12), be sure to use object rules to include the corresponding objects. You can use the Default Object Rows option to include all objects not affected by object rules, if required.

12. Optional: Click the **Column Mask Rules** tab to perform masking as part of the subset definition.

    a. Click **Create** and enter search criteria to filter on columns within the schema. These would typically be vertical columns such as CLOB AND BLOB columns.

    > **Note:**
    >
    > If you are using column mask rules instead of masking definitions (see Step 13), you can select no more than 10 columns in a given table. This restriction applies to the export method but not to the in-place delete method.

    Click **OK**.

    b. Select a row or rows in the column search results and click **Manage Masking Formats**.

    c. In the pop-up dialog, select a masking format and value to apply to the columns. For multiselection, the same format must be appropriate for all columns. If you select multiple columns, ensure that the column rule format you choose is applicable to the selected columns. Use the columns (flags) not null and unique to enforce compliance.

    Click **OK** to apply the masking format to the columns.

13. Optional: **Click the Data Masking Definitions** tab to include masking definitions as part of the subsetting operation or to perform at the source data masking only.

    a. Click **Add**.

    b. In the pop-up dialog, enter search criteria to retrieve appropriate definitions. Be sure to select the desired radio button (All or Any). All formats except compound masking are supported for inline masking.

    After doing a search, the search results appear in the data masking table. Select a masking definition and click **OK**.

14. Click the **Space Estimates** tab.

    - Note the value in the Estimated Subset Size MB column. The space estimates depend on optimizer statistics, and the actual distribution of data can only be calculated if histogram statistics are present.
    - Whenever you add new rules, recheck the space estimates for updated values.
    - Data in the Space Estimates subpage is sorted with the largest applications appearing at the top.

> **Note:**
>
> Space estimates are accurate only if “dbms_stats.gather_table_stats” is used. Also, space estimates do not reflect the effect of data masking, if used. If you provide a Where clause and subsequent rule parameter properties, the Space Estimates subpage is updated with the value contained in the Rule Parameters tab.

15. Optional: click the **Pre/Post Subset Script** tab.
    - You can specify a pre-subset script to run on the subset database before you select subset data.
    - You can specify a post-subset script to run on the subset database after you assemble the subset data.
    - Either script type runs on the source database.

> **Note:**
>
> Ensure that the PL/SQL block defined the pre-subset or post-subset script includes a “/” at the end.

16. Click **Return**.

    The definition is complete and displayed in the Data Subsetting Definitions table.

You can now proceed with script generation. Alternatively, you may want to save the script for future use. In either case, you must decide whether to export data to a dump file or delete data from a target database.

Tip: If you have a very large database of 4 terabytes, for instance, and you want to export a small percentage of the rows, such as 10%, it is more advantageous to use the export method. Using the in-place delete method would require 3.6 terabytes of data, which would not perform as quickly as the export method.

The in-place delete method is recommended when the amount of data being deleted is a small percentage of the overall data size.

Generating a Subset Script

To prepare and submit a job to generate a subset script:

1. Select the definition within the table, open the **Actions** menu, then select **Generate Subset**.
   The Generate Subset pop-up appears.
2. Select a target database that is either the same target database you used to create the subset model, or database containing similar table schema and objects.
3. Decide if you want to create a subset by writing subset data to export files, or by deleting data from a target database.

> **Note:**
>
> Choosing to delete data creates an in-place subset by removing or deleting unwanted data from a cloned copy of the production database, rather than a production database. Only data satisfying the rules are retained. You should never use this option on a production database.

Select either **Named Credentials** or **New Credentials** if you have not already set preferred credentials.

If you have defined any parameters from the Rule Parameters tab, they appear in the table at the bottom. You can change a parameter value by clicking on the associated field in the Value column.

4. Click **Continue** to access the Parameters pop-up. The contents of the pop-up depend on whether you chose the export or delete option in the previous step.
   For Writing Subset Data to Export Files, provide the requisite information, then click **Continue** to schedule the job.
   - Specify a subset directory where to save the export dump. The drop-down list consists of directory objects for which you have access. Alternatively, you can select a custom directory path. Click the check box if you want to speed the process by using an external directory. Recommended default: `DATA_PUMP_DIR`.
   - Specify appropriate values if you want to override the defaults: enter a name for the export file; specify a maximum file size in megabytes; specify the maximum number of threads of active execution operating on behalf of the export job. This enables you to consider trade-offs between resource consumption and elapsed time.
   - Specify whether you want to export only the subsetted data or the entire database along with the subsetted data.
   - Select whether to enable dump file compression and encryption. Enter and confirm an encryption password, if appropriate. Log file generation is selected by default.

   For Deleting Data From a Target Database, provide the requisite information, then click **Continue** to schedule the job.
   - Specify a subset directory where to save the subset scripts. The drop-down list consists of directory objects for which you have access. Alternatively, you can select a custom directory path. Recommended default: `DATA_FILE_DIR`.
   - You must enable the check box indicating that the selected target is not a production database in order to proceed.
5. Click **Continue** to review the estimated storage requirement for generating the subset script.

> **Note:**
>
> If the required storage for generating the subset script is much more than the available space, allocate the necessary space to ensure that you have enough space to proceed with generating the subset script.

6. Specify a name for the subset job, and provide a description. Schedule the job to run immediately or at a later point of time, then click **Submit**.
   The Data Subset Definitions page reappears, and the Most Recent Job Status column shows that the subset job is running, and subsequently that it has succeeded.

After performing this procedure, you can now create a subset database with the generated export files at any time.

Download a Subset Script

To download a subset script:

1. Select the definition within the table, open the **Actions** menu, then select **Download Subset Script**. The Subset Mode pop-up appears.
2. Select a target database that is either the same target database you used to create the subset model, or database containing similar table schema and objects.
3. Decide if you want to create a subset by writing subset data to export files, or by deleting data from a target database.
   Choosing to delete data creates an in-place subset by removing/deleting unwanted data from a cloned copy of the production database, rather than a production database. Only data satisfying the rules are retained. You should never use this option on a production database.

   Select either **Named Credentials** or **New Credentials** if you have not already set preferred credentials.

   If you have defined any parameters from the Rule Parameters tab, they appear in the table at the bottom. You can change a parameter value by clicking on the associated field in the Value column.
4. Click **Continue** to access the Parameters pop-up. The contents of the pop-up depend on whether you chose the export or delete option in the previous step.
   For Writing Subset Data to Export Files, provide the requisite information, then click **Continue** to schedule the job.
   - Specify a subset directory where to save the export dump. The drop-down list consists of directory objects for which you have access. Alternatively, you can select a custom directory path. Click the check box if you want to speed the process by using an external directory. Recommended default: `DATA_PUMP_DIR`.
   - Specify appropriate values if you want to override the defaults: enter a name for the export file; specify a maximum file size in megabytes; specify the maximum number of threads of active execution operating on behalf of the export job. This enables you to consider trade-offs between resource consumption and elapsed time.
   - Specify whether to export only the subsetted data or the entire database along with the subsetted data.
   - Select whether to enable dump file compression and encryption. Enter and confirm an encryption password, if appropriate. Log file generation is selected by default.

   For Deleting Data From a Target Database, provide the requisite information, then click Continue to schedule the job.
   - Specify a subset directory where to save the subset scripts. The drop-down list consists of directory objects for which you have access. Alternatively, you can select a custom directory path. Recommended default: `DATA_FILE_DIR`.
   - You must enable the check box indicating that the selected target is not a production database in order to proceed.
5. Click **Continue**. A progress indicator tracks script generation. When complete, the Files table lists the results of script generation.
6. Click **Download**. In the File Download pop-up that appears, click **Save**.

7. In the Save As pop-up that appears, navigate to a file location and click **Save**.

   The file containing the scripts (`SubsetBundle.zip`) now appears at the specified location on your desktop.

To run the saved script at a later time:

1. Copy the ZIP file to the target database and extract it to a directory on which you have the requisite privileges.
2. Change directory to where you extracted the files.
3. Execute the following script from the SQL command line: `subset_exec.sql`
   Note that if you want to change generated parameter settings, you can do so by editing the following file in a text editor prior to executing the script: `subset_exec_params.lst`

Synchronizing a Subset Definition with an Application Data Model

Changes to an ADM, adding referential relationships or deleting tables, for example, can render a subset definition stale. The Subset Definitions page clearly indicates this condition with a lock icon next to the subset name and an invalid status. Also, most menu items on the Actions menu are disabled. To revert the status to valid and unlock the subset definition, you have to synchronize the definition with its associated ADM.

1. On the Subset Definitions page, select the locked subset definition.
2. From the Actions menu, select **Synchronize**.
3. Complete the job submission dialog, then click **Submit**.
   When the job completes, the subset definition is unlocked and available for use.

Importing and Exporting Subset Templates and Dumps

A subset template is an XML file that contains the details of the subset, consisting of the application, subset rules, rule parameters, and pre-scripts or post-scripts. When you create a subset definition and specify that you want to write subset data to export files, the export files become a template that you can subsequently import for reuse. You would import the template to perform subset operations on a different database.

Typically, the workflow is that you would first import a previously exported ADM template, which is another XML file, while creating an ADM. You would then import the related subset template while creating a data subset definition. You could alternatively select an existing ADM (skipping the import ADM flow) while importing the subset template.

Importing a Subset Definition

There are three methods of import:

- Importing a Subset Definition XML File From Your Desktop
- Importing a Subset Dump
- Importing a Subset Definition XML File From the Software Library

Importing a Subset Definition XML File From Your Desktop

1. From the Actions menu, select **Import**, then select **File from Desktop**.
2. In the pop-up that appears:

a. Specify a name for the subset definition.
b. The ADM on which the subset is based.
c. A source database.
d. The location on your desktop from which you want to import the subset definition.
e. Click **Continue**.

3. In the pop-up that appears:
   a. Enter a descriptive job name (or accept the default).
   b. Provide credentials.
   c. Schedule the job.
   d. Click **Submit**.

After the job runs successfully, the imported subset appears in the list of subsets in the table on the Data Subset Definitions page.

Importing a Subset Dump

1. From the **Actions** menu, select **Import**, then select **Subset Dump**.
2. In the pop-up that appears:
   a. Select a target database.
   b. Provide database and host credentials, then click **Login**.
   c. Specify the location of the dump file, which can be in a selected directory on the target database or at a custom path on the target. Note that if the original export action used an external location for the dump files, the location must be specified as well.
   d. Click **Continue**.
3. In the pop-up that appears:
   a. Select whether to import both metadata and data, or data only. If data only, indicate if you want to truncate, that is, overlay existing data or append to existing data.
   b. Enable OID transform during type or object creation to create a new OID. Creating a new OID will break the REF columns that point to the object.
   c. Specify whether to use the default tablespace on the target database or remap the tablespaces from the existing tablespace to another tablespace.
   d. Perform tablespace remapping as necessary.
   e. Specify whether you want to retain the existing tables in the destination schema or drop the existing tables in the destination schema.
   f. Perform schema remapping as necessary.
   g. Select log file options.
   h. Click **Continue**.
4. In the pop-up that appears:
   a. Enter a descriptive job name (or accept the default).
   b. Schedule the job.
   c. Click **Submit**.

The job reads the dump files and loads the data into the selected target database.

Importing a Subset Definition XML File From the Software Library

1. From the **Actions** menu, select **Import**, then select **File from Software Library**.
2. In the Import Data Subset Definition from Software Library pop-up that appears:
   a. Select the desired subset definition XML file on the left.
   b. Provide the ADM properties on the right.
   c. Click **Continue**.
3. In the pop-up that appears:
   a. Enter a descriptive job name (or accept the default).
   b. Provide credentials.
   c. Schedule the job.
   d. Click **Submit**.

After the job runs successfully, the imported subset appears in the list of subsets in the table on the Data Subset Definitions page.

Exporting a Subset Definition

There are two methods of export:

- Exporting a Subset Definition as an XML File to Your Desktop
- Exporting a Subset Definition From the Software Library

Exporting a Subset Definition as an XML File to Your Desktop

1. From the Data Subset Definitions page, select the subset definition you want to export.
2. From the **Actions** menu, select **Export**, then select **Selected Subset Definition**.
3. In the File Download pop-up that appears, click **Save**.
4. In the Save As pop-up that appears, navigate to a file location and click **Save**.
   The system converts the subset definition into an XML file that now appears at the specified location on your desktop.

Exporting a Subset Definition From the Software Library

1. From the **Actions** menu, select **Export**, then select **File from Software Library**.
2. In the Export Subset Definition from Software Library pop-up that appears, select the desired subset definition and click **Export**.
3. In the File Download pop-up that appears, click **Save**.
4. In the Save As pop-up that appears, navigate to a file location and click **Save**.
   The system converts the subset definition into an XML file that now appears at the specified location on your desktop.

After the job runs successfully, the subset template appears in the list of subsets in the table on the Data Subset Definitions page.

Granting Privileges on a Subset Definition

You can grant privileges on a subset definition that you create so that others can have access. To do so, you must be an Enterprise Manager Administrator with at least Designer privileges on the subset definition.

1. On the Data Subsetting Definitions page, select a definition to which you want to grant privileges.
2. From the **Actions** menu, select **Grant**, then select as follows:
   - Operator – to grant Operator privileges on the subset definition to selected roles or administrators, which means the grantees can view and copy but not edit the definition.
   - Designer – to grant Designer privileges on the subset definition to selected roles or administrators, which means the grantees can view and edit the definition.
3. In the dialog that opens, select the type (administrator or role, or both). Search by name, if desired. Make your selections and click **Select**.
   Those selected now have privileges on the subset definition.
4. Use the Revoke action if you want to deny privileges previously granted.

See "**Access Control Procedures**" for more information on privileges within the test data management area.

Lifecycle Management

This section discusses the lifecycle management of Application Data Models, Data Masking, and Data Subsetting definitions.

In the event of an Enterprise Manager user being dropped or modified, the user can reassign the Application Data Models, Data Masking and Data Subsetting definitions to another user in the system. However, if the user doesn't reassign the Application Data Models, Data Masking and Data Subsetting definitions to another user, these definitions are automatically reassigned to the SYSMAN user.

When you try to reassign the Application Data Model, Data Masking and Data Subsetting definitions to another user in the system, and if the reassigned user already has a definition with the same name, the original definitions are renamed.

For example: User A has an Application Data Model, ADM1, and user B also has an Application Data Model, named ADM1. If user B is being dropped, and you choose to assign its definitions to user A, the original definition ADM1 is renamed to ADM1_B. The original definitions with the same name are renamed by suffixing "_" and adding the user name that is being dropped. After the reassignment, user A will now have both definitions ADM1 and ADM1_B.

6

Frequently Asked Questions

This FAQ chapter includes a product overview, key components and features, deployment and administration details, and answers to common customer questions with resource links.

Product Overview

Growing security threats and ever-expanding privacy regulations have made it necessary to limit the exposure of sensitive data. Copying production data for non-production purposes such as development and data analytics proliferates sensitive data, expanding the security and compliance boundary and increasing the likelihood of data breaches. Oracle Data Masking and Subsetting provides a flexible solution that discovers, masks, and subsets sensitive data, allowing organizations to safely share data across their non-production environments.

What is Data Masking and Subsetting?

Data masking replaces sensitive data such as credit card numbers with fictitious yet realistic-looking data. Data subsetting is the process of retaining or extracting a selected portion of data from a larger database.

Why do I need to mask and subset data?

Copying production data to non-production, outsourced, partner, and cloud environments for test, development, and other purposes proliferates sensitive information such as credit card numbers and social security numbers. This increases the risk of a data breach as non-production environments are generally not as protected or monitored as diligently as production environments. For this reason, data privacy standards such as PCI-DSS recommend rendering sensitive production data unreadable when used for test and development.

Subsetting extracts only the necessary information from a large database for sharing with internal and external teams, reducing the resources required to store and manage that data in test and development. Masking and subsetting sensitive data in non-production environments helps improve security and minimize compliance and infrastructure costs.

How does masking improve security and minimize compliance costs?

Masking sensitive data in test and development environments reduces the overall compliance boundary, restricting it to only production environments. Rendering masked data unreadable in these environments limits the risk of a data breach and helps minimize compliance costs.

Why are data masking and subsetting important for cloud computing?

Organizations understand the advantage of leveraging a cloud platform for test and development. However, they are concerned about uploading sensitive on-premises production data to the cloud because of data privacy and compliance reasons. Other concerns are the storage cost associated with the cloud platform and the network cost due to data transfers.

Oracle Data Masking and Subsetting addresses these concerns by enabling cloud users to mask sensitive data on-premises before uploading it to the cloud. The product helps reduce storage and network costs by extracting a subset of production data for upload to the cloud.

Is data masking reversible?

Data masking usually means permanently replacing the data and ensuring that no one can retrieve the original data. But, sometimes you might want to see the original data. Reversible masking is helpful when businesses need to mask and send their data to a third party for analysis, reporting, or some other business purpose. After the processed data is received from the third party, the original data can be recovered. The Encrypt masking format is the only format that supports reversible masking, enabling the masked data to be reverted to its original form when necessary.

Oracle also offers a product called Data Redaction. Unlike data masking, Data Redaction does not alter the actual data. Instead, it modifies the output during presentation, keeping the original data intact. This feature allows you to control who can see the original data and who sees the redacted version.

Components and Features

What are the main components of Oracle Data Masking and Subsetting?

The main components are:

- **Data Discovery** provides automated procedures to discover sensitive columns and parent-child relationships. The discovery results are stored as an application data model, which is reusable across multiple databases.
- **Data Masking** assists in mapping masking formats to discovered sensitive columns, creating reusable masking scripts. It also provides a workflow to mask data.
- **Data Subsetting** helps create reusable goal/condition-based subsetting rules on a database. It also provides a workflow to generate subsets.
- **Masking Format** library provides a comprehensive set of predefined masking formats to mask sensitive data such as credit card numbers, national identifiers, and phone numbers. It also allows creating new masking formats to meet domain-specific requirements.

How does the product preserve the relational integrity of the data in an application?

Masking parent table that has child relationships also automatically masks the children to preserve referential integrity. For example - if there is a foreign key dependency, on a column that is being masked, we want to make sure that after masking, the foreign key constraint is not violated and appropriately mask the child table. Oracle Data Masking and Subsetting does the following to minimize the disruption of applications post-masking and subsetting:

- The product uses automated discovery procedures to gather referential integrity or parent-child relationships between the columns before the masking and subsetting process.
- During the masking and subsetting process, parent and child columns are processed consistently to preserve the integrity between these columns.
- When masking a parent table that has child relationships, the tool automatically masks the child tables to preserve referential integrity.

Can masking support multi-byte or international characters?

Several masking options support multi-byte or international characters, such as UTF-8. The suitable masking formats include Array List, Shuffle, Substitute, Table Column, and User Defined Function.

Does the product include predefined masking formats?

Yes, Oracle Data Masking and Subsetting provides out-of-the-box masking formats covering a broad range of sensitive data, such as national identifiers of multiple countries, credit card numbers of various vendors, phone numbers, and more.

Which masking techniques are supported by the product?

Some options include generating fixed/random characters or numbers, replacing them with null values, substituting data from a random list or table column, and SQL or regular expression-based masking. You also have several advanced options to meet complex business requirements, such as:

- **Shuffle Masking** randomly shuffles data within a table/ view. For example, columns containing salaries can be shuffled to break the employee-salary mapping.
- **Encryption** encrypts sensitive data using a cryptographic key while preserving the data's format. It's a reversible masking option, and you can decrypt your data using the same key. This feature is useful when masked data sent to a third party has to be merged with further updates.
- **Conditional Masking** masks column data using different masking formats based on user-defined conditions. For example, in a column, the US identifiers can be masked using the Social Security Number format and the UK identifiers using the National Insurance Number format.
- **Compound Masking** masks related columns as a group, ensuring the masked data across the related columns retain the same. For example, address fields such as city, state,
- **Deterministic Masking** generates consistent masked output for a given input across application schemas and databases.
- **User-defined PL/SQL Masking** enables you to define custom masking logic or migrate your existing masking scripts.

Oracle also offers a product called Data Redaction. Unlike data masking, Data Redaction does not alter the actual data. Instead, it modifies the output during display time, keeping the original data intact. This feature allows you to control who can see the original data and who sees the redacted or masked version.

Which subsetting techniques are supported by the product?

Oracle Data Masking and Subsetting simplifies the task of subsetting through its goal or condition-based subsetting techniques. A goal can be a relative table size, such as extracting a 1% subset of a table containing 10 billion rows. Condition-based subsetting is useful for creating a subset using the data itself. For example, you can use a time-based condition, such as discarding all user records created before a particular year. Another example is a region-based condition, where you might extract only Asia Pacific information to support new application development.

Does Oracle Data Masking and Subsetting work with packaged applications like Oracle E-Business Suite and Oracle Fusion Applications?

As Oracle Data Masking and Subsetting is a database-centric solution, it works for all supported databases regardless of the application. However, care is required when setting up data models and masking and subsetting definitions to avoid misconfigurations that could break complex applications.

Can I mask and subset databases running in Oracle Cloud?

You can mask databases in the Oracle Cloud but subsetting is not yet supported. Oracle Data Masking for cloud databases works much like on-premises databases.

What masking formats support deterministic masking?

Currently, Substitute and Encrypt masking formats provide deterministic masking transformation.

- **Encrypt Masking Format**: This transformation encrypts and decrypts the original data using a secure key string. The input data format is preserved during encryption and decryption. This transformation uses industry-standard 3DES algorithm. This transformation is helpful when businesses need to mask and send their data to a third-party for analysis, reporting, or any other business processing purpose. After the processed data is received from the third-party, the original data can be recovered using the same key string that was used to encrypt the data.
- **Substitute Masking Format:** The Substitute format uses a table of values from which to substitute the original value with a mask value. As long as this table of values does not change, the mask is deterministic or consistent across the three databases.

How does reversibility work with Encrypt masking format?

Reversibility can be performed using Encrypt Masking format which is Key Based Reversible Masking. It uses 3DES algorithm where a seed value is provided to encrypt and decrypt the data.

If I use Oracle Data Masking and Subsetting in Enterprise Manager to generate the masking script, can I run these scripts directly on the Database Server for masking?

Yes, you can run the generated scripts directly on the database but the officially supported and recommended way to perform data masking is through EM.

If a customer masks data on one server (which is on-premises) and then the customer moves this data to another Oracle Database (e.g., on OCI cloud or any other third-party cloud), can Oracle Data Masking and Subsetting perform the decryption on this new server?

Yes, the customer can mask and subset databases in Oracle Database Cloud Service (DBCS). Oracle Data Masking and Subsetting for cloud databases works much like it does for on-premises databases. Oracle Data Masking and Subsetting license is included in DBCS High Performance, Extreme Performance, and Exadata Service.

Deployment and Administration

How do I download and install Oracle Data Masking and Subsetting?

Oracle Data Masking and Subsetting is pre-installed with Oracle Enterprise Manager. To use Oracle Data Masking and Subsetting, you must have the Data Masking and Subsetting license pack.

What are the different ways to mask and subset data?

The product provides two modes to mask and subset data:

- **In-Database Masking and Subsetting:** The target data is first copied (cloned) to a separate location. Oracle Data Masking and Subsetting operates on the cloned data. After processing is complete, the resulting masked data can be cloned and distributed for non-production.

> **Note:**
>
> In-Database masking and subsetting directly operates on the underlying data, on the original copy of the data and is ***NOT*** recommended to be performed on production databases.

- **In-Export Masking and Subsetting:** The masking and subsetting rules are applied while the data is extracted from the target database, and the resulting data is written to Oracle Data Pump dump files. In this mode of operation, Oracle Data Masking and Subsetting can run directly on the production system and unmasked data does not leave the production database. After processing is complete, the dump file containing masked data can be imported into non-production databases.

How does Oracle Data Masking and Subsetting compare to other similar Oracle offerings?

Oracle Solutions	Description
Oracle Data Masking and Subsetting	Software for creating masked and subsetted copies of production data for use in non-production environments such as testing and development databases.
Oracle Data Safe (Data Discovery/ Data Masking)	Discover and mask sensitive data with a cloud service that supports Oracle Databases everywhere: in the Oracle Cloud, on-premises, and third-party clouds.
Oracle Data Redaction	Redacts sensitive data from query results before display through client applications. Enforces redaction at runtime, with low overhead, and according to conditions set in policies.
Oracle Label Security	Implements Multi-Level Security (MLS), enabling rows with differing sensitivity to reside in the same table. Explicitly labels rows with group, compartment, and sensitivity levels then matches them with user labels.

Can Data Masking and Subsetting run without Enterprise Manager?

All Data Masking and Subsetting objects are centrally located in the Oracle Enterprise Manager repository, which facilitates centralized creation and administration of Application Data Models, Data Masking and Subsetting rules or definitions. So, we recommend using it through Enterprise Manager.

Recurring Customer Queries

Recurring Issues	Alternate Questions	Response
Masking Reversibility	Can masked data be restored to its original state? Is it possible to reverse data masking to retrieve the original data? Does data masking allow for the original data to be recovered? Can masked data be decrypted back to its original form? Is there a way to revert masked data to its initial state?	Data masking usually means permanently replacing the data to ensure that no one can retrieve the original data. We recommend that masking should only be performed directly on non-production databases or in-export to ensure critical data isn't lost. If the original data is required even in a non-production environment, a backup should be taken before masking. However, sometimes you might need to view the original data. The "Encrypt masking format" offered by DMS is the only format that supports masking reversibility, allowing the masked data to be reverted to its original form when necessary. Oracle also offers a product called Data Redaction. Unlike data masking, Data Redaction does not alter the actual data. Instead, it modifies the output during display time, keeping the original data intact. This feature allows you to control who can see the original data and who sees the redacted or masked version.
Privileges	What all privileges are required to use Data Masking and Subsetting? What are minimum privileges required to use DMS features?	Go to the Prerequisites chapter for details on privileges: **Prerequisites** For Basic set privileges to perform Data masking operation in a Database Vault enabled environment, please refer the following Oracle documentation URL: *Oracle Database Vault Administrator's Guide*

Recurring Issues	Alternate Questions	Response
Masking Performance	1. Are there ways to optimize data masking performance?/ How can I make the data masking process more efficient? 2. What are some of the ways to calculate/ analyze the masking performance?	**1. Improving Masking Performance in Oracle Data Masking and Subsetting** Based on the best practices and known issues from Oracle Data Masking and Subsetting Performance Tuning, here are key steps to enhance masking performance: **Updating Statistics** • **Action**: Regularly update database statistics using `DBMS_STATS.GATHER_TABLE_STATS` • **Outcome**: Improved query execution plans and overall masking efficiency. **Indexing** • **Action**: Create and rebuild indexes on frequently queried columns. • **Outcome**: Faster data retrieval and improved masking performance. **2. Analyzing Masking Performance in Oracle Data Masking and Subsetting** *Performance Calculation:* The total time for masking can be calculated as: "Time taken for cloning tables to be masked"+"Time taken for CTAS of each table"+"Time taken to disable and enable existing indexes and constraints"+"Time taken to apply masking logic (X)" The variable 'X' depends on the masking format. For example, fixed string masking is faster, while encryption takes longer. *Hardware Impact:* Masking performance is directly proportional to the underlying hardware capabilities. *Parallelism:* Adjust the PARALLEL_MAX_SERVERS parameter based on the formula: `{PARALLEL_MAX_SERVERS} = {CPU_COUNT} x {PARALLEL_THREADS_PER_CPU} x (2 { if PGA_AGGREGATE_TARGET > 0 else 1}) x 5` *Performance Testing:* Use masking definitions to analyze the performance impact on a test system. This analysis can provide insights into

Recurring Issues	Alternate Questions	Response
		the time taken to mask data and any changes in the execution plan. *Masking Methods:* In-Export Masking: Involves additional time for exporting the table to a dump file. In-Place Masking: Involves additional cloning time from the source to the staging environment. The total time for In-Export masking is the sum of the export time and the time taken to apply the masking logic.
New DMS Customer	1. What resources are available for Data Masking and Subsetting (DMS)? 2. Do we need a separate license for Data Masking and Subsetting (DMS) in Enterprise Manager (EM)? 3. Is there a scheduling mechanism to schedule jobs on a weekly/ monthly basis instead of running them manually? 4. How can the new rows getting inserted into columns be automatically masked? 5. Currently, masked values are being updated for all rows in a column. How can we configure it to mask only when the column is not null?	1. Resources: Documentation 2. In order to use the DMS feature, the Data Masking and Subsetting License Pack must be purchased 3. No; currently all jobs must be scheduled manually but multiple jobs can be manually scheduled in the future to achieve the desired weekly/monthly cadence 4. Currently, DMS does not support masking at the time of insertion 5. Add a new masking rule with the condition "column_name is not null" and specify the desired masking format under this new masking rule. In total, there would be 2 rules, first rule with the "column_name is not null" with the desired masking format and there would be the second default condition rule which could just be "Preserve Original Data"

Recurring Issues	Alternate Questions	Response
Masking Formats	What are some of the examples for Personal Information (PI) Columns and their recommended Masking formats?	**Here is the list of available basic Masking Formats:** Array List Delete Encrypt Fixed Number Fixed String Null Value Preserve Original Data Random Decimal Numbers Random Digits Random Dates Random Strings Random Numbers SQL Expression Regular Expression Post-Processing Function Shuffle Subsitute Substring Table Column Truncate User-Defined Function
Regular Expression Basics	What are some of the examples of Regex? How can we implement Regex for a sensitive column?	Refer to tables **Basic Building Blocks of Regex** and **Regular Expression Example**.

Table 6-1 Basic Building Blocks of Regex

Basic Characters	Examples	Definition
. (dot)	Pattern: b.t Matches: "bat", "bet", "bit", "bot", "but", etc.	Matches any single character except newline (\n).
^	Pattern: ^start Matches: "start" at the beginning of a line.	Matches the start of the string or line.
$	Pattern: end$ Matches: "end" at the end of a line.	Matches the end of the string or line.
|	Pattern: cat|dog Matches: "cat" or "dog"	Alternation, matches either the expression before or after the |.
[]	Pattern: [aeiou] Matches: Any single vowel character ("a", "e", "i", "o", "u")	Character class, matches any one of the characters inside the brackets.
()	Pattern: (ab) Matches: "ab"	Grouping, groups multiple tokens together and captures matched text.

Table 6-1 (Cont.) Basic Building Blocks of Regex

Basic Characters	Examples	Definition
\	Pattern: \d Matches: The pattern \d matches any digit (0-9)	Escape character, used to escape a special character or indicate a special sequence.

Table 6-2 Regular Expression Example

Regular Expression	Explanation	Example
[A-Za-z]{3}	Matches exactly three alphabetic characters (uppercase or lowercase).	Fgy
[0-9]{3}	Matches exactly three digits.	674
[A-Za-z]{3}+[0-9]{3}	Matches exactly three alphabetic characters (uppercase or lowercase) followed by exactly three digits.	Fgy674
[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}	This pattern matches a typical email address format. It includes character classes ([a-zA-Z0-9._%+-]), the + quantifier for one or more occurrences, the @ literal character, and the . (dot) for domain separation.	f@8.co
(?:https?://)?(?:www\.)?[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}	(?:https?://)?: Non-capturing group that matches "http://" or "https://", optionally. optionally.\. (?:www\.)?: Non-capturing group that matches "www." [a-zA-Z0-9.-]+: Matches one or more alphanumeric characters, dots, or hyphens (for the domain name). [a-zA-Z]{2,}: Matches a dot followed by at least two alphabetic characters (for the top-level domain).	https://www.example.com
#([a-fA-F0-9]{6}|[a-fA-F0-9]{3})	This pattern matches a hexadecimal color code starting with #, followed by either six ({6}) or three ({3}) characters from the set [a-fA-F0-9]. It uses () for grouping and {} for specifying repetitions.	#FfF011
\d{4}-\d{2}-\d{2}	This pattern matches a date in the format YYYY-MM-DD. It uses \d for digits and {} for specifying exact repetitions.	2024-07-16

Masking Format Examples for Common Sensitive Types

Column Name	Masking Format	Masking Format Entries	Original Data	Masked Data

CustomerID	Fixed String Random Strings	Fixed String: 'F' Random Strings: Start Length: 7 End Length: 7 Random Digits: Start Length: 3 End Length: 3	Fjg860	Faqjdnfh623
Age	Shuffle	Shuffle	26	34
FirstName	Random Strings	Random Strings: Start Length: 1 End Length: 6	Kate	aaabgd
LastName	Substring	Substring: Start Position: 1 Length: 3	Wilson	Wil
DateOfBirth	Random Dates	Random Dates: Start Date: 07/18/50 End Date: 07/18/20	7/3/1998	8/5/1963
SocialSecurityNumber (SSN)	Random Number Post-Processing Function	Random Number: Start Integer: 20000000 End Integer: 738999999 Post-Processing Function: `DM_FMTLIB.MGMT_DM_GEN_SSN_FH`	049-66-6786	136-76-4899
EmailAddress	SQL Expression	SQL Expression: %FirstName% || '.' || %LastName% || '@company.com '	ktewilson@gmail.com	Kate.Wilson@company.com

PhoneNumber	Random Digits Random Number Post-Processing Function	Random Digits: Start Length: 7 End Length: 7 Random Number: Start Integer: 1 End Integer: 279 Post-Processing Function: `DM_FMTLIB.MGMT_DM_GEN_PH_USA_FH`	6033550232	619-989-9213
AddressLine1	Shuffle (Choose Group Masking option)	Shuffle Columns	303 Madison	26 West
AddressLine2			San Street	Wheelock St
City			Redwood City	Hanover
PostalCode			94002	3755
Country			United States	United States
Gender	Array List	Array List: Female, Male	Female	Male
MaritalStatus	Null Value	Null Value	Single	N/A
Nationality	Truncate	Truncate	American	N/A
PassportNumber	Substitute	Regular Expression: [A-Z]{1}[0-9]{7}	P2378283	V4697997
DriverLicenseNumber	Encrypt	Regular Expression: [D][0-9]{7}	D7482464	D8208541
CreditCardNumber	Random Digits Post-Processing Function	Random Digits: Start Length: 10 End Length: 10 Post-Processing Function: `DM_FMTLIB.MGMT_DM_GEN_VC`	4111111111111110	4556771680150140
BankAccountNumber	Substring	Substring: Start Position: 3 Length: 3	123456789	345
AccountHolderName	Table Column	Schema: HR Table: New_table Column: LastName	Kate Wilson	Wilson

TaxIdentificationNumber (TIN)	Fixed String Random Digits	Fixed String: TIN Random Digits: Start Length: 1 End Length: 6	TIN123456	TIN232163
EmploymentStatus	Preserve Original Data		Employed	Employed
MedicalRecordNumber	Fixed String	MRNxxxxxx	MRN123456	MRNxxxxxx
UserName	SQL Expression	SQL Expression: %FirstName% || '_' || %LastName%	katewilson1980	Kate_Wilson
Password	delete	N/A	xYdkI34b	N/A

Where can I find more information on Oracle Data Masking and Subsetting?

For more information, such as product data sheet, documentation, customer references, and blog, please visit the Oracle Data Masking and Subsetting page on oracle.com:

https://www.oracle.com/security/database-security/data-masking/

Index