



Oracle Database 19c Detective Controls

Detective Controls

Introduction

- We will cover the following:
 - Implement Oracle Audit Vault
 - Implement Database Firewall
 - Oracle Virtual Private Database (VPD)
 - Data Redaction
 - Oracle Privilege Analysis
 - Enumerate Oracle Database Auditing Solutions
 - Oracle Database Vault
 - Database Vault and Data Pump
 - Oracle Application Context
 - Explain Oracle Compliance Framework
 - Configure Compliance Framework rules



Oracle Database 19c Detective Controls

Oracle Audit Vault

Oracle Audit Vault

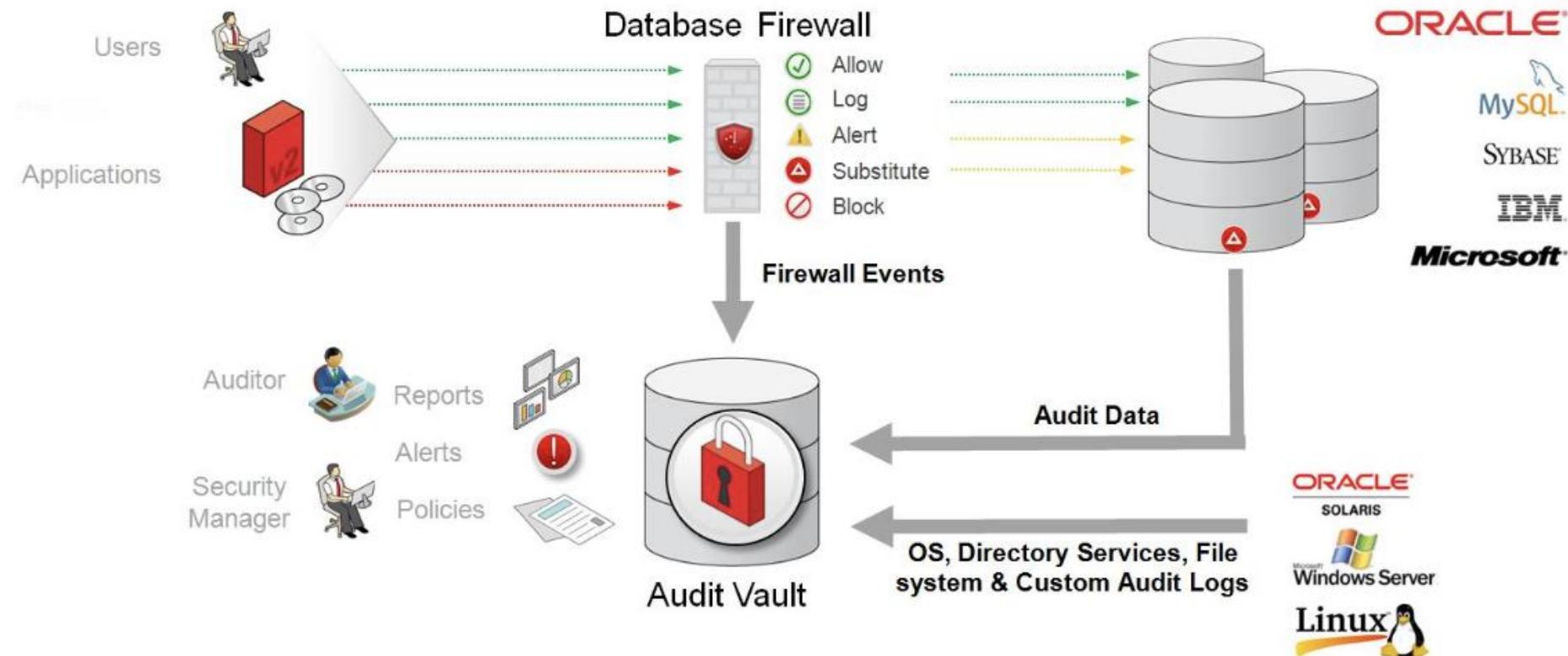
- What is Audit Vault and Firewall?
 - Continuous auditing of Oracle and non-Oracle databases
 - Audits:
 - File system
 - Application
 - Audit specific data

- What is Audit Vault and Firewall?
 - Firewall
 - First line of network defense
 - Prevents SQL injection
 - Application bypass
 - Enforces expected behavior

Oracle Audit Vault

- Consolidates audit data from thousands of databases as well as monitors SQL traffic
- Prevents un authorized SQL statements or out of policy statements
- Supports most major Databases
 - DB2
 - SQL Server
 - MySQL

Oracle Audit Vault



- Why Audit Vault and Firewall?
 - Today's world everything is audited
 - Audit only tracks attempts
 - Most organizations are not just Oracle

- Why Audit Vault and Firewall?
 - Oracle Audit Vault and Firewall consolidates all auditing to one centralized location
 - Reports may be established
 - Alerts can be established
 - Real time alerts and monitoring

- Difference between Audit Vault and Firewall
 - Firewall determines if an event may go through the firewall
 - Audit tracks the event
 - Firewall is proactive
 - Firewall has 3 states
 - Permit
 - Block
 - Modify

- Firewall identifies and attempt and blocks

SQL Injection Report

Go Actions ▾

Saved Report = "SQL Injection" Command Text does not contain 'TEXTSIZE' Event Time is in the last 24 hours Location = 'Network' Name = 'CRM Database'

Event Time	Type	Name	Client IP	User Name	Target Object	Command Text	Location	Action Taken
11/28/2012 2:18:35 AM	Microsoft SQL Server	CRM Database	10.240.169.211	crmapp		DISCONNECTED	Network	pass
11/28/2012 2:18:33 AM	Microsoft SQL Server	CRM Database	10.240.169.211	crmapp	credit_card	select * from credit_card where ssn = ##### or '#'='#'	Network	block
11/28/2012 2:18:27 AM	Microsoft SQL Server	CRM Database	10.240.169.211	crmapp	credit_card	select * from credit_card where ssn = #####	Network	pass
11/28/2012 2:18:23	Microsoft	CRM	10.240.169.211	crmapp		CONNECTED,LOGIN	Network	pass

- Audit Vault

- Database repository that stores consolidated audit information
- Enabled via lightweight agents
- Audit information from many different:
 - Databases
 - File systems
 - Applications
- Single source of audit data

Oracle Audit Vault



<input type="text"/> Go		Actions +						
1		1						
Event Time	Class	Type	Name ▾	Client IP	User Name	Command Class	Command Text	Location
11/27/2012 4:08:27 PM	Database	Oracle Database	target1	10.240.114.167	avadmin	DDL	create user joedba identified by HIDDEN	Network
11/28/2012 6:28:46 PM	OS	Microsoft Windows	msw	10.240.169.211	Windows Administrator	LOGON		Event Log
11/28/2012 3:07:53 AM	Database	Oracle Database	Sales DB		SYSTEM	GRANT	grant dba to appsdba	Audit Table
11/28/2012 3:07:50 AM	Database	Oracle Database	Sales DB		SYSTEM	CREATE	create user appsdba identified by *	Audit Table
11/28/2012 2:18:27 AM	Database	Microsoft SQL Server	CRM Database	10.240.169.211	crmapp	SELECT	select * from credit_card where ssn = '#####'	Network

Oracle Audit Vault

- How is Audit Vault implemented
 - Audit Vault Server
 - Audit Vault agents
 - Similar to OEM

The screenshot shows the Oracle Audit Vault Server interface. The top navigation bar includes Home, Secured Targets (which is selected), Reports, Policy, and Settings. Below the navigation is a breadcrumb trail: Home > Secured Targets. On the left, a sidebar titled 'Manage' lists Targets, Groups, Access Rights, Entitlement Snapshots, Manage Snapshots, Manage Labels, Quick Links (Audit Trails, Enforcement Points), and a Copy button. The main content area displays a table of secured targets:

Name	Type	Description
CRM Database	Microsoft SQL Server	
Sales DB	Oracle Database	
Solaris11	Oracle Solaris	
mssql1	Microsoft SQL Server	
msw	Microsoft Windows	
s1c03sew-db01	Oracle Database	
s1c03jh-db02	Oracle Database	
s1c03jh-os	Oracle Solaris	



Oracle Database 19c Detective Controls

Oracle Database Firewall

Database Firewall

- Specifically what is Oracle's Database Firewall?
 - A network monitoring component outside the database that monitors the inbound SQL traffic and serves as a first line of defense against SQL injection threats and other unauthorized SQL statements
 - Monitors data access, enforces access policies, highlights anomalies and helps protect against network based attacks originating from outside or inside the organization

Database Firewall

- What does Database Firewall do?
 - Deploy Database Firewall in active monitoring mode to protect a database assets or in passive monitoring mode to alert security operations personnel of unexpected activity, and/or supplemental auditing to address compliance requirements
 - Passive monitoring mode, Database Firewall observes database traffic and analyzing SQL interactions

Database Firewall

- What does Database Firewall do?
 - Data from Database Firewall is logged to the Audit Vault, providing reports to span information observed on the network alongside audit information from the database, operating systems, and directories
 - In active monitoring mode, Database Firewall transparently intercepts SQL traffic coming from database clients acting as an application layer firewall, analyzes the security of the SQL payload in TCP packets before forwarding it on to the database

- How does Database Firewall work?
- White List Policy
 - Enforces security using a set of approved SQL statements along with the conditions under which they were executed including the username, IP address, time of day, and program name
 - Compares SQL traffic with the approved white list and then based upon the policy, it chooses to alert, substitute, or block the SQL statement

Database Firewall

- White List Policy
 - The approved SQL or white list is learned over time by monitoring database traffic
 - The period of time varies depending upon organization

- Black List Policy Enforcement
 - A black list model that blocks specific SQL statements
 - Evaluate various factors such as username, IP address, time of day and program, before making the decision
- Exception List Policy Enforcement
 - Exception lists policies override white list and black list policies by allowing custom bypass policies to be created for specific activities
 - Exception list policies could be used to enable a specific remote administrator coming from a predetermined IP address to diagnose a particular application performance issue without being bound by the white list or the blacklist

- Unauthorized SQL is handled in different ways
 - Terminates the connection/blocks all incoming traffic from that connection
 - Blocks the SQL statement
 - Modifies the request
 - Alerts on out of policy SQL Statement

Database Firewall

ORIGINAL STATEMENT (FRAUDULENT)	SUBSTITUTED STATEMENT	DATABASE RESPONSE (RESULT)
SELECT * FROM tbl_users;	SELECT * FROM tbl_users WHERE 'a' = 'b';	No record found
DROP TABLE tbl_accounts;	SELECT * FROM aaabbbccc;	Error. Table not known
UPDATE tbl_accounts SET accounts = '123' WHERE user = 'Fred';	SELECT DUAL SET 'Fred';	Error. Incorrect Syntax.

- Additional Audit Reporting
 - Audit Vault and Database Firewall have predefined reports
 - Reports can display consolidated audit information from databases, operating systems, and directories, providing a holistic picture of activities across the enterprise
 - Reports can include information on database account management, roles and privileges, object management, and stored procedure changes

Database Firewall

- Additional Audit Reporting
 - Reports can be interactive, through an Auditor Console web interface, or PDF or XLS report files
 - PDF and XLS report definitions can be used to schedule automatic report generation

- Compliance Reporting
 - Payment Card Industry Data Security Standard (PCI-DSS)
 - Gramm-Leach-Bliley Act (GLBA)
 - Health Insurance Portability and Accountability Act (HIPAA),
 - Sarbanes-Oxley Act (SOX)
 - European Union Data Protection Act (DPA)

Database Firewall



Home > Reports > Compliance Reports

Built-in Reports

- Audit Reports
- Compliance Reports**
- Specialized Reports

Custom Reports

- Uploaded Reports
- Interactive Reports

Report Workflow

- Report Schedules
- Generated Reports

Quick Links

- Audit Trails

Payment Card Industry (PCI) Reports

Gramm-Leach-Bliley Act (GLBA) Reports

Health Insurance Portability and Accountability Act (HIPAA) Reports

Sarbanes-Oxley Act (SOX) Reports

Data Protection Act (DPA) Reports

To associate Secured Target(s) with this Compliance Category, click on the Go button

Activity Overview	Digest of all captured audit events for a specified period of time
Data Access	Details of audited read access to data for a specified period of time
Data Modification	Details of audited data modifications for a specified period of time
Database Schema Changes	Details of audited DDL activity for a specified period of time

Database Firewall

The screenshot shows the Oracle Audit Vault Server interface. At the top, there is a navigation bar with links for Home, Secured Targets, Reports (which is the active tab), Policy, and Settings. On the right side of the header, there is a user profile icon labeled 'avauditor' and links for Help and Logout. Below the header, the page title is 'Home > Reports'. On the left, there is a sidebar with sections for Built-in Reports (Audit Reports, Compliance Reports, Specialized Reports, Custom Reports, Uploaded Reports, Interactive Reports), Report Workflow (Report Schedules, Generated Reports), Quick Links (Audit Trails, Enforcement Points), and a section for Alerts. The main content area is titled 'Activity Reports' and contains a list of reports with descriptions and icons:

Report Type	Description	Icon (File, Magnifying Glass, Calendar)
Activity Overview	Digest of all captured audit events for a specified period of time	[Icon]
Data Access	Details of audited read access to data for a specified period of time	[Icon]
Data Modification	Details of audited data modifications for a specified period of time	[Icon]
Data Modification Before-After Values	Details of audited data modifications for a specified period of time showing before and after values	[Icon]
Database Schema Changes	Details of audited DDL activity for a specified period of time	[Icon]
All Activity	Details of all captured audit events for a specified period of time	[Icon]
Failed Logins	Details of audited failed user logins for a specified period of time	[Icon]
User Login and Logout	Details of audited successful user logins and logouts for a specified period of time	[Icon]
Entitlements Changes	Details of audited entitlement related activity for a specified period of time	[Icon]
Audit Settings Changes	Details of observed user activity targeting audit settings for a specified period of time	[Icon]
Secured Target Startup and Shutdown	Details of observed startup and shutdown events for a specified period of time	[Icon]

Below the 'Activity Reports' section, there are three collapsed sections: 'Alert Reports', 'Entitlement Reports', and 'Stored Procedure Audit Reports'.

- Entitlements Reporting
 - Describe the types of access that users have to an Oracle database
 - Information about the users, roles, profiles, and privileges used
 - Compare different snapshots to find how the entitlement information has changed over time
- Stored Procedure Audit Reports
 - Monitor any changes made to the Stored Procedures on the protected databases
 - Shows all stored procedure operations, deleted and created procedures, as well as modification history

- Alerts and Notifications

- Detect and alert on activities that may indicate attempts to gain unauthorized access and/or abuse system privileges
- Generate alerts on network activity
- Alerts can be associated with any database event including system events such as changes to application tables, creating privileged users
- Events when someone attempts to access sensitive business information and is blocked by an Oracle Database Vault policy

Database Firewall

Modify Alert

Name * Possible Brute Force Attack Alert

Secured Target Type Oracle Database

Severity * Critical

Threshold (times) * 10

Duration (min) * 1

Group By (Field) - Select F

Status * Enabled

Description

Condition *

:SECURED_TARGET_NAME = 'AV Server'

Condition - Available Fields

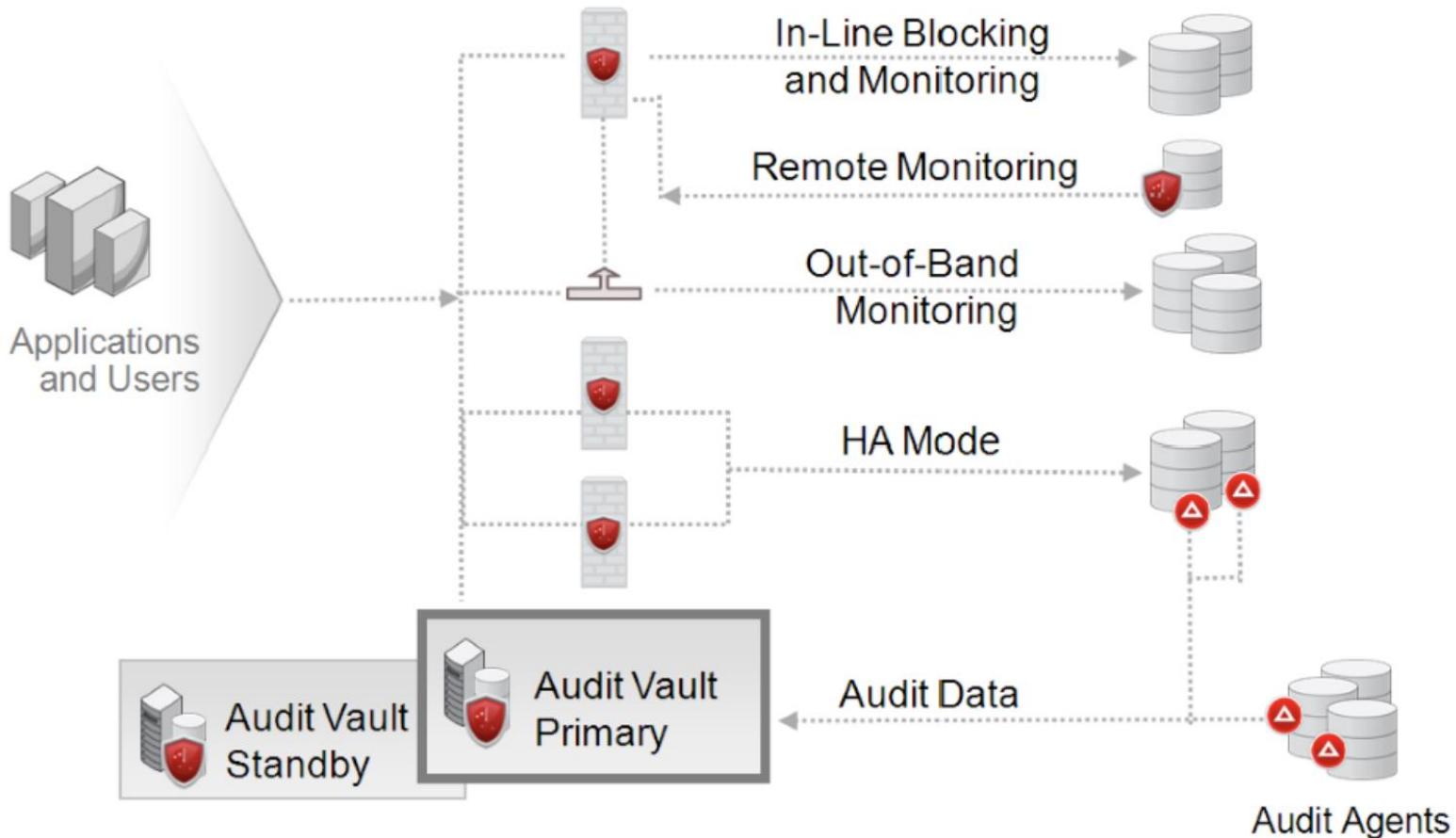
ACTION_TAKEN
AV_TIME
CLIENT_HOST_NAME
CLIENT_IP
CLUSTER_TYPE
COMMAND_CLASS
ERROR_CODE
ERROR_MESSAGE
EVENT_NAME
EVENT_STATUS
EVENT_TIME
NETWORK_CONNECTION
OSUSER_NAME
SECURED_TARGET_NAME
TARGET_OBJECT
TARGET_OWNER
TARGET_TYPE
THREAT_SEVERITY
USER_NAME

34 of 4000

Database Firewall

- Deployment
 - Deployed as a transparent network bridge, simply inserted into the network in a segment that lies between database clients/application servers and the databases being protected
 - Supports a local server-side, monitor-only agent to ensure flexibility in the choice of the network point at which the traffic is monitored

Database Firewall



ORACLE
MySQL

SYBASE

IBM

Microsoft

- Audit Agents Deployment
 - Distributed as JAR files to the target systems and require no additional manual configuration or updates once they have been distributed
- Policy Authoring and Management
 - Define a white list, black list, or exception list of SQL statements for a given database



Oracle Database 19c Detective Controls

Virtual Private Database

- In this chapter, we will cover the following:
 - Creating different policy functions
 - Creating Oracle Virtual Private Database row-level policies
 - Creating column-level policies
 - Creating a driving context
 - Creating policy groups
 - Setting context as a driving context
 - Adding a policy to a group
 - Exempting users from VPD policies

- Oracle **Virtual Private Database (VPD)** is a security feature, introduced in Oracle Database 8i.
- Discretionary access control (DAC) grants/restricts access to data at an object level.
- VPD enables you more granular control over security of your data.

- There are five types of policies based on how often a policy function is evaluated:

- DBMS_RLS.DYNAMIC
- DMBS_RLS STATIC
- DBMS_RLS.SHARED_STATIC
- DBMS_RLS.CONTEXT_SENSITIVE
- DBMS_RLS.SHARED_CONTEXT_SENSITIVE

Introduction

- Steps to implement the VPD policy

Create a PL/SQL package that sets application context



Create an application context



Create a policy function



Create a VPD policy

Introduction

- A driving context is an application context that has at least one attribute and its purpose is to determine which group of policies will be applied.

Introduction

- Steps to implement policy groups

Determine the default policies



Create and set the driving context



Create a policy group for each application



Add policies to the appropriate policy groups

Creating different policy functions

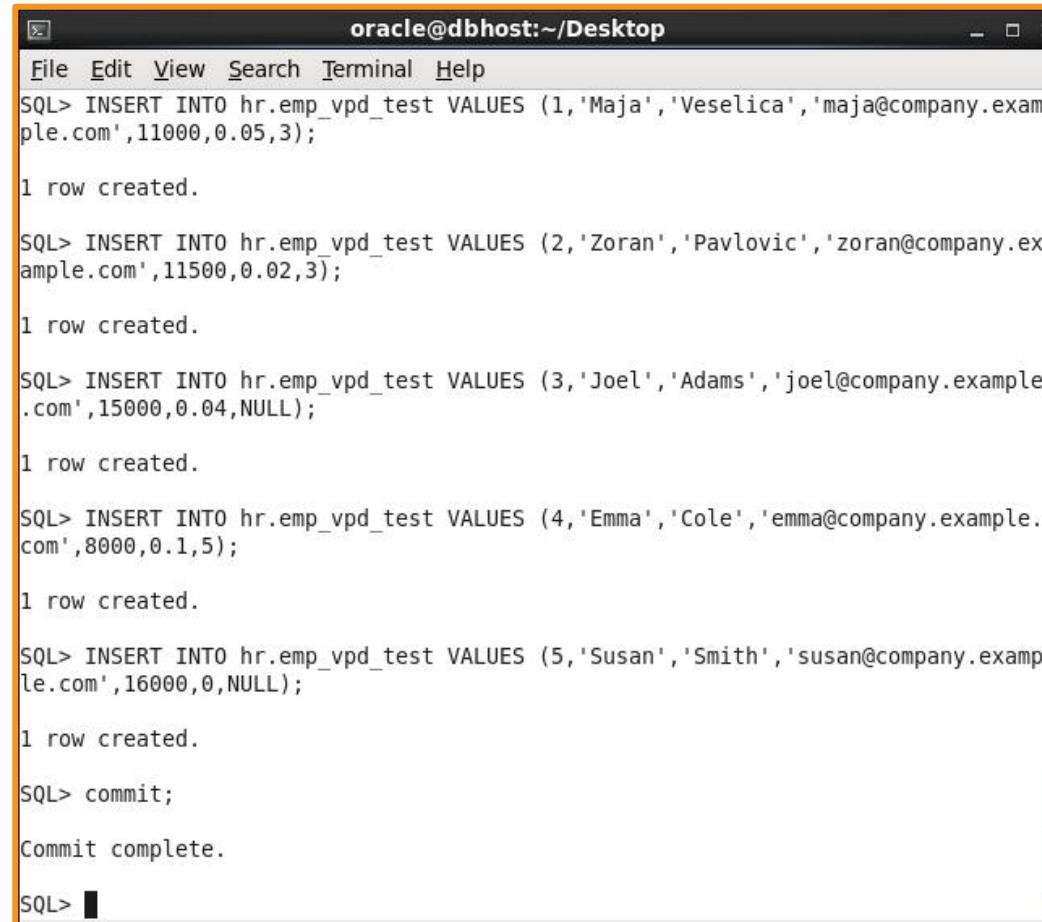
- The purpose of a policy function is to return a predicate that will be applied in WHERE clause of the statement (except for INSERT operation).
- You'll create several simple policy functions, based on different business and security requirements.
- You'll need to create the table `hr.emp_vpd_test`, insert several values into that table, and create several user.

```
SQL> CREATE TABLE hr.emp_vpd_test (
  2 emp_id NUMBER(6) NOT NULL,
  3 first_name VARCHAR2(30) NOT NULL,
  4 last_name VARCHAR2(30) NOT NULL,
  5 email VARCHAR2(30) NOT NULL,
  6 salary NUMBER(8,2),
  7 comm_pct NUMBER(2,2),
  8 mgr_id NUMBER(6));
```

Table created.

Creating different policy functions

- Test data in the table `hr.emp_vpd_test`



```
oracle@dbhost:~/Desktop
File Edit View Search Terminal Help
SQL> INSERT INTO hr.emp_vpd_test VALUES (1,'Maja','Veselica','maja@company.example.com',11000,0.05,3);
      1 row created.

SQL> INSERT INTO hr.emp_vpd_test VALUES (2,'Zoran','Pavlovic','zoran@company.example.com',11500,0.02,3);
      1 row created.

SQL> INSERT INTO hr.emp_vpd_test VALUES (3,'Joel','Adams','joel@company.example.com',15000,0.04,NULL);
      1 row created.

SQL> INSERT INTO hr.emp_vpd_test VALUES (4,'Emma','Cole','emma@company.example.com',8000,0.1,5);
      1 row created.

SQL> INSERT INTO hr.emp_vpd_test VALUES (5,'Susan','Smith','susan@company.example.com',16000,0,NULL);
      1 row created.

SQL> commit;
Commit complete.

SQL> ■
```

Creating different policy functions

- Create a policy function that uses the application context you created, whereas other policy functions you created use built-in application contexts.

A policy function can be part of a package or is standalone.

Creating different policy functions

- To test whether a policy function where a user can't access data in a table and other users can access entire data in the table:
 1. Connect to the database as the user **maja** and execute the following statement:

```
SQL> select no_access('a','b') from dual;  
NO_ACCESS('A','B')  
-----  
1=1
```

Creating different policy functions

2. Grant the user **susan** execute on **no_access** function and connect to the database as the user **susan**.

```
SQL> grant execute on no_access to susan;  
Grant succeeded.  
  
SQL> connect susan  
Enter password:  
Connected.
```

3. Execute the following statement:

```
select maja.no_access('a','b') from dual;
```

```
SQL> select maja.no_access('a','b') from dual;  
MAJA.NO_ACCESS('A','B')  
-----  
1=2
```

Creating different policy functions

4. Connect to the database as the user **maja** and revoke execute on **no_access** function from the user **susan**.

```
SQL> connect maja
Enter password:
Connected.
SQL> revoke execute on no_access from susan;

Revoke succeeded.
```

Creating Oracle Virtual Private Database row-level policies

- Oracle VPD row-level policies restrict users' access per row for a protected object.
- This means that two users who execute the same query against, for example, a table may, as a result, receive different number of rows.

Creating Oracle Virtual Private Database row-level policies

- You define two VPD policies on the same table, and they are both enabled.
- The first one only restricts the user **susan** from accessing the table, whereas the other one affects all users connected to the database.
- If **susan** connects to the database, both policies will determine whether she can access the data and if **yes**, which data.
- The way the policies are defined, she won't be able to view data in the table.

Creating column-level policies

- When you create a column-level VPD policy, you define sensitive columns, and if those columns are referenced in a query, statement will be rewritten.
- To create a column-level VPD policy, you also use the **DBMS_RLS.ADD_POLICY** procedure.

Creating column-level policies

1. Create the `test_col` VPD policy.
2. The user is granted the role (`HREMP_TEST`) that will allow him to view entire data.
3. Restrict displayed rows by `TEST_POL2`, so user can view only his data.

Creating column-level policies

4. Disable the TEST_POL2 policy using the DBMS_RLS.ENABLE_POLICY procedure. The syntax is:

```
DBMS_RLS.ENABLE_POLICY (
object_schema IN VARCHAR2 NULL,
object_name   IN VARCHAR2,
policy_name   IN VARCHAR2,
enable        IN BOOLEAN TRUE)
```

5. Other user can view all rows, but cannot view salary and comm_pct, because he doesn't have the HREMP_TEST role.

Creating a driving context

- Having multiple VPD policies is harder to manage, and it can lead to unexpected/unwanted results.
- For example, you have two applications and want to create two policy groups.
- If the first application accesses the table, the `test_pol1` and `test_col` policies should be enforced, and if second application accesses the table, the `test_pol2` policies should be applied.

Creating a driving context

- You'll create an application context and set it.
- You'll need an existing user who can create an application context.

Creating policy groups

- You'll create two policy groups that will be applied to table `hr.emp_vpd_test`.
- You'll need an existing user who has appropriate privileges.

Setting context as a driving context

- You'll make an existing application context a driving context (you'll associate it with the protected object).
- You'll need an existing application context and an existing user who has appropriate privileges.

Adding policy to a group

- Create VPD policies as part of a policy group.
- You'll need an existing user who has appropriate privileges. Drop all VPD policies using the **DBMS_RLS.DROP_POLICY** procedure.
- Drop policies

```
SQL> BEGIN
 2  DBMS_RLS.DROP_POLICY('HR','EMP_VPD_TEST','TEST_POL1');
 3  DBMS_RLS.DROP_POLICY('HR','EMP_VPD_TEST','TEST_POL2');
 4  DBMS_RLS.DROP_POLICY('HR','EMP_VPD_TEST','TEST_COL');
 5  END;
 6  /
```

PL/SQL procedure successfully completed.

Exempting users from VPD policies

- VPD policies are not enforced for users who connect as **sysdba**, during direct path export, and for users who have the **EXEMPT ACCESS POLICY** privilege.
- You'll connect to the database as **SYS** user and grant **EXEMPT ACCESS POLICY** to an existing user.



Oracle Database 19c Detective Controls

Data Redaction

We will cover the following:

- Creating a redaction policy when using full redaction
- Creating a redaction policy when using partial redaction
- Creating a redaction policy when using random redaction
- Creating a redaction policy when using regular expression redaction
- Using Oracle Enterprise Manager Cloud Control 19c to manage redaction policies
- Changing the function parameters for a specified column
- Adding a column to the redaction policy
- Enabling, disabling, and dropping a redaction policy
- Exempting users from data redaction policies

- **Oracle Data Redaction** is a new security feature, introduced in Oracle Database 19c.
- Oracle Data Redaction masks sensitive data just before the results of the SQL query are returned to the application that issued the query.
- Data stored in the database is **NOT** changed in any way.

- Oracle Data Redaction and Oracle Data Masking are both used to mask sensitive data, but these solutions are completely different.
- **Oracle Data Masking** enables organizations to use production data in development and test environments by changing production data with realistic data (transformation is done by using masking rules).

- When you implement Oracle Data Redaction, you have to decide the following:

- What data should be redacted
- Which redaction method is most suitable for the identified data
- In which situations the redaction should take place

The parts of a redaction policy

WHAT	Schema, object, column
HOW	Redaction type, required parameters
WHEN	SQL expression

The types of redaction

None	Full	Partial	Regular Expression	Random
<ul style="list-style-type: none">Redaction is NOT applied	<ul style="list-style-type: none">Columns are redacted to constant values depending on column data type	<ul style="list-style-type: none">User-specified positions are replaced by a user-specified character	<ul style="list-style-type: none">Pattern for matching and replacing is defined and used for redaction	<ul style="list-style-type: none">Preserves data typesRandomizes output

- To view which data redaction policies are defined and whether they are enabled, you can query the **redaction_policies** view.
- Also, it is very useful to query the **redaction_columns** view, which shows which columns will be masked and what type of redaction will be used.
- We assume that database is up and running, and each user has at least a **create session** privilege.

Creating a redaction policy when using full redaction

- You will create a redaction policy on the **income_level** column, find the default values for different data types, and change the default value for the **varchar2** data type.

Creating a redaction policy when using full redaction

- You'll need the following:

- An existing user who can view data in OE.CUSTOMERS sample table but doesn't have exempt redaction policy privilege (for example, oe)
- To connect as a **SYS** user to the database
- To restart the database

Creating a redaction policy when using full redaction

1. In order to manage redaction policies, you need to connect to a database as a user who has an execute privilege on the **dbms_redact** package.
2. Define redaction policy **CUST_POL**.
3. Creating a new redaction policy is done by using the **ADD_POLICY** procedure in the **DBMS_REDACT** package.

Creating a redaction policy when using full redaction

4. The **WHAT** section defines on which the column redaction policy should be applied.
5. The **HOW** section defines the redaction type.
6. The **WHEN** section defines the conditions when protected data will be masked.

Creating a redaction policy when using full redaction



SQL> SELECT * FROM EMPLOYEES;

NAME	SALARY
frank	0
emily	0
grace	0

EMPLOYEES

NAME	SALARY
frank	8800
emily	9600
grace	11300

```
DBMS_REDACT.ADD_POLICY  
(object_schema => 'GLDB'  
object_name => 'EMPLOYEES'  
policy_name => 'SAL_POLICY'  
column_name => 'SALARY'  
function_type => DBMS_REDACT.FULL,  
expression => '7=7');
```

Creating a redaction policy when using full redaction

```
oracle@dbhost:~/Desktop
File Edit View Search Terminal Help
SQL> connect / as sysdba
Connected.
SQL> desc REDACTION_VALUES_FOR_TYPE_FULL
Name          Null?    Type
-----        -----
NUMBER_VALUE   NOT NULL NUMBER
BINARY_FLOAT_VALUE NOT NULL BINARY_FLOAT
BINARY_DOUBLE_VALUE NOT NULL BINARY_DOUBLE
CHAR_VALUE      VARCHAR2(1)
VARCHAR_VALUE    VARCHAR2(1)
NCHAR_VALUE     NCHAR(1)
NVARCHAR_VALUE  NVARCHAR2(1)
DATE_VALUE      NOT NULL DATE
TIMESTAMP_VALUE NOT NULL TIMESTAMP(6)
TIMESTAMP_WITH_TIME_ZONE_VALUE NOT NULL TIMESTAMP(6) WITH TIME ZONE
BLOB_VALUE       BLOB
CLOB_VALUE       CLOB
NCLOB_VALUE      NCLOB
SQL> select date_value from redaction_values_for_type_full;
DATE_VALU
-----
01-JAN-01
```

1

To find out default values for other data types, query **REDACTION_VALUES_FOR_TYPE_FULL**. Finding the default value for **DATE** data type is depicted

How to change the default value



```
oracle@dbhost:~/Desktop
File Edit View Search Terminal Help
SQL> connect secmgr
Enter password:
Connected.
SQL> exec dbms_redact.update_full_redaction_values (varchar_val => 'T')
BEGIN dbms_redact.update_full_redaction_values (varchar_val => 'T'); END;
*
ERROR at line 1:
ORA-00942: table or view does not exist
ORA-06512: at "SYS.DBMS_REDACT", line 363
ORA-06512: at line 1
```

1

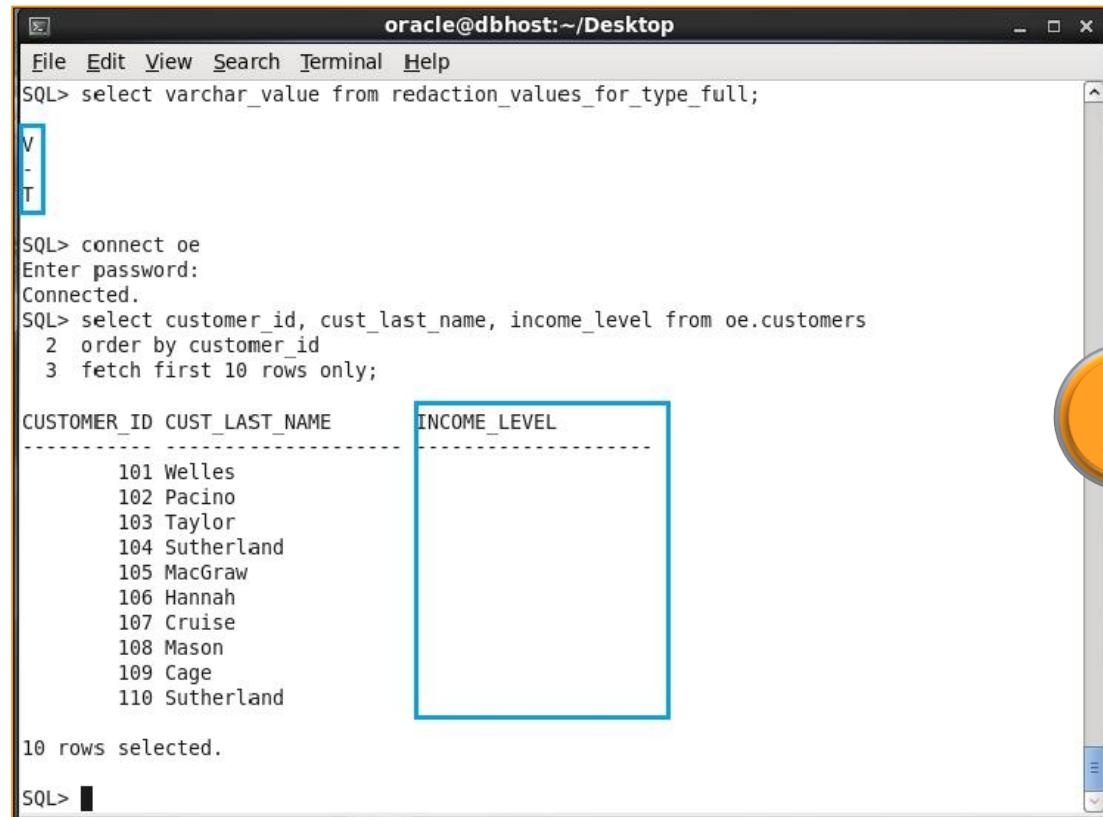
An unsuccessful change of default value

Connect to the database as the **SYS** user and change the default value.

2

```
SQL> exec dbms_redact.update_full_redaction_values (varchar_val => 'T')
PL/SQL procedure successfully completed.
```

How to change the default value



```
oracle@dbhost:~/Desktop
File Edit View Search Terminal Help
SQL> select varchar_value from redaction_values_for_type_full;
V
-
T

SQL> connect oe
Enter password:
Connected.
SQL> select customer_id, cust_last_name, income_level from oe.customers
  2 order by customer_id
  3 fetch first 10 rows only;

CUSTOMER_ID CUST_LAST_NAME           INCOME_LEVEL
-----  -----
      101 Welles
      102 Pacino
      103 Taylor
      104 Sutherland
      105 MacGraw
      106 Hannah
      107 Cruise
      108 Mason
      109 Cage
      110 Sutherland

10 rows selected.

SQL>
```

3

Verify that the default value is changed and that there is **no effect before you restart the database.**

How to change the default value



```
oracle@dbhost:~/Desktop
File Edit View Search Terminal Help
SQL> connect / as sysdba
Connected.
SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> startup
ORACLE instance started.

Total System Global Area 1073741824 bytes
Fixed Size          2932632 bytes
Variable Size       692060264 bytes
Database Buffers   373293056 bytes
Redo Buffers        5455872 bytes
Database mounted.
Database opened.
SQL> connect oe
Enter password:
Connected.

SQL> select customer_id, cust_last_name, income_level from oe.customers
  2 order by customer_id
  3 fetch first 10 rows only;

CUSTOMER_ID CUST_LAST_NAME      INCOME_LEVEL
-----  -----
  101 Welles                      T
  102 Pacino                     T
  103 Taylor                      T
  104 Sutherland                  T
  105 MacGraw                     T
  106 Hannah                      T
  107 Cruise                      T
  108 Mason                       T
  109 Cage                        T
  110 Sutherland                  T

10 rows selected.
```

4

Restart the database and verify that the modified default value is displayed.

Creating a redaction policy when using partial redaction

- You will implement partial redaction on columns of two different types: Number and Varchar2.
- Partial redaction means that only part of the data in a specified column will be masked whereas the other part of the data will be visible to the user.

Creating a redaction policy when using partial redaction

- In order to manage redaction policies and also to create some test tables, you can connect to a database as a user who has a dba role.
- If you just need to manage redaction policies, you can connect with user who has the execute privilege on the **dbms_redact** package.

Creating a redaction policy when using partial redaction

- Creation of redaction policy for number type column:
 1. Create a redaction policy named **a_tbl_partial**.
 2. Creating a new redaction policy is done by using the **ADD_POLICY** procedure in the **DBMS_REDACT** package.
 3. A policy consists of several distinct sections, which column our redaction policy should be applied, redaction type, function parameters, condition when protected data will be masked.
 4. Evaluate using the following expression:
SYS_CONTEXT("SYS_SESSION_ROLES","MYROLE") = "FALSE"

Creating a redaction policy when using partial redaction

- Creation of redaction policy for the **Varchar2** column type.
 1. The difference is on function parameters and on condition, which is always **TRUE**.

Creating a redaction policy when using partial redaction



SQL> SELECT * FROM CUSTOMERS;

NAME	CREDIT_CARD
tom	#####9132
steve	#####5691
john	#####5806

CUSTOMERS	
NAME	CREDIT_CARD
tom	3455647456589132
steve	3734982321225691
john	3472586894975806

```
DBMS_REDACT.ADD_POLICY  
(object_schema => 'ERNESTO'  
object_name => 'CUSTOMERS'  
policy_name => 'CCN_POLICY'  
column_name => 'CREDIT_CARD'  
function_type => DBMS_REDACT.PARTIAL,  
function_parameters => 'VVVVVVVVVVVVVVVV,  
VVVVVVVVVVVVVVVV, #,1,12'  
expression => '1=1');
```

Creating a redaction policy when using partial redaction

- Even though users can't see unmasked data, they can use redacted columns in **where** clause:

```
SQL> select * from ernesto.customers;
NAME          CREDIT_CARD
-----
tom           #####9132
steve         #####5691
john          #####5806
SQL> select * from ernesto.customers where credit_card like
      '34%';
NAME          CREDIT_CARD
-----
tom           #####9132
john          #####5806
```

Creating a redaction policy when using random redaction

- Random redaction type is usually used for the number and date-time data types because for these data types, it is hard to make a distinction between the redacted (random) and real data.
- You will create redaction policy **EMP_POL** using random redaction type on **hr.employees** table, column salary, by using SQL*Plus.
- You will modify the **EMP_POL** redaction policy.

Creating a redaction policy when using random redaction

- To complete this, you'll need:

- An existing user who can view data in the HR.EMPLOYEES sample table but doesn't have an exempt redaction policy privilege (for example, hr)
- The secmgr user created in the Creating a redaction policy using full redaction recipe or another user who can create redaction policies (has execute on the dbms_redact package)

Creating a redaction policy when using random redaction

1. Create the redaction policy **EMP_POL** by using the procedure **ADD_POLICY** in the **DBMS_REDACT** package.
2. Define that random redaction type will be used to redact data.
3. Policy expression, specifies the blacklist (which contains only user HR).
4. To define a whitelist change operator **=** to operator **\diamond** and define left and right operand according to your needs.

Creating a redaction policy when using random redaction

- When the number data type is redacted using random redaction type, the redacted value will belong to the interval $[0, |n|]$, where $|n|$ is the absolute value of the original data.
- The only exception to this rule is when original data is an integer between **-1** and **9**, and in that case, the redacted value will belong to the interval $[0, 9]$.

Creating a redaction policy when using regular expression redaction

- A regular expression redaction type enables you to create and implement flexible redaction rules.
- You define patterns that will be used in order to match and replace data, as well as some other parameters of the search.
- You will create the redaction policy **SHORT_POL**, which will be used to mask customers' phone numbers.

Creating a redaction policy when using regular expression redaction

- You'll need:

- An existing user who can view data in the SH.CUSTOMERS sample table but doesn't have an exempt redaction policy privilege (for example, sh)
- The secmgr user you created in the Creating redaction policy using full redaction recipe or another user who can create redaction policies (has execute on dbms_redact package)

Creating a redaction policy when using regular expression redaction

- When creating redaction policies that use regular expression redaction type, you can choose between redaction shortcuts and the creation of custom regular expressions.

Using Oracle Enterprise Manager Cloud Control 19c to manage redaction policies

- You will perform several tasks with Data Redaction policies using Oracle Enterprise Manager Cloud Control 19c, including creation, modification, and deletion.
- You need Enterprise Manager Cloud Control 19c and **HR** sample schema in the database.

Changing the function parameters for a specified column

- There are several ways in which you can change an existing redaction policy. You will:

- Change the function parameters for a specified column (the `a_tbl_partial` policy)
- Add a column (`commission_pct` in the `hr.employees` table) to the redaction policy `EMP_POL`

Changing the function parameters for a specified column

- Also, it is possible to remove column from the redaction policy, alter the policy expression, and modify the type of redaction for a specified column.
- You concluded that the `a_tbl_partial` redaction policy doesn't satisfy the requirements for your application anymore because it redacts first four digits with 0 and leading zeros are not displayed in the application.
- You decide to alter the `a_tbl_partial` policy.

Changing the function parameters for a specified column

- Before doing this, you should have completed the **Creating a redaction policy when using partial redaction** section.
- You will use the **secmgr** user you created in the **Creating a redaction policy when using full redaction** section.

Add a column to the redaction policy

- You have to modify the existing redaction policy in order to redact more than one column in the table.
- In the table **HR.EMPLOYEES**, besides the column **SALARY**, you want to redact the column **COMMISSION_PCT**.
- You will modify the redaction policy **EMP_POL**.
- You decide that you want to use full redaction type for the column **COMMISSION_PCT**.

Add a column to the redaction policy

- Before doing this, you should have completed the **Creating redaction policy when using random redaction** section.
- You will use the **secmgr** user you created in the **Creating redaction policy when using full redaction** section.

Add a column to the redaction policy

- Use the procedure `ALTER_POLICY` in the PL/SQL package `DMBS_REDACT` to change redaction policies.
- Specify value for the `ACTION` parameter, which defines what kind of change will happen.

Enabling, disabling, and dropping redaction policy

- You will perform the three basic tasks: enabling, disabling, and dropping the same redaction policy (**CUST_POL**).
- Also, you will check which redaction policies exist in the database and whether they are enforced (enabled).

Enabling, disabling, and dropping redaction policy

- Before doing this, you should have completed the **Creating a redaction policy when using full redaction** section.

Exempting users from data redaction policies

- You will create a user and then exempt that user from Data Redaction.
- This user will be exempted from all redaction policies in the database.

Exempting users from data redaction policies

- Before doing this, you should have completed the **Creating a redaction policy when using the partial redaction section**.

Exempting users from data redaction policies

- There is a new system privilege that is used to control which users will be exempted from data redaction in Oracle Database.
- This privilege is **EXEMPT REDACTION POLICY**.
- Users who are granted this privilege will be able to see clear (unmasked) data in the whole database if they have (select) privilege to access that data.

Exempting users from data redaction policies

- Backup/restore as well as import and export operations are not subject to data redaction.
- However, data redaction policies are included in export and import operations.



Oracle Database 19c Detective Controls

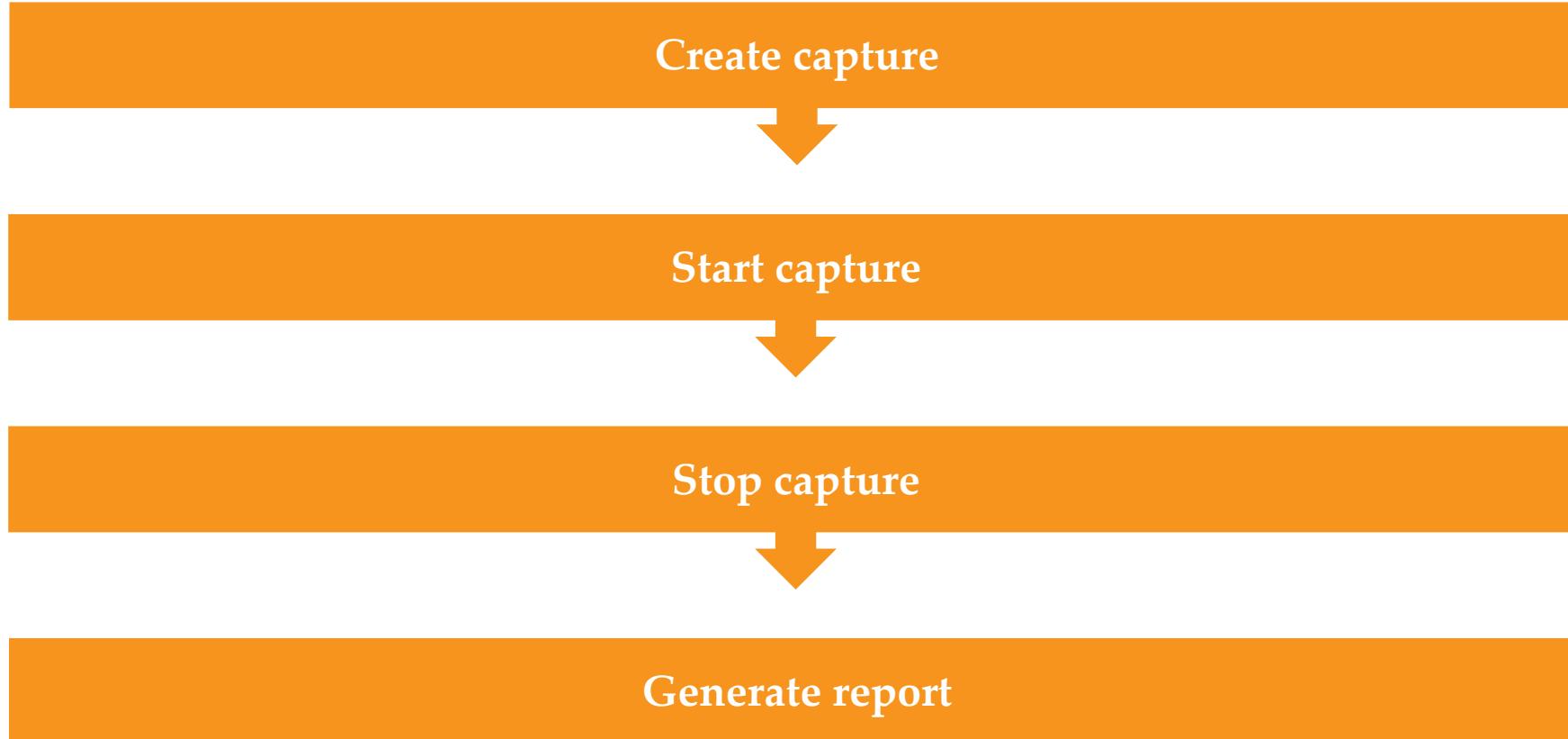
Privilege Analysis

In this chapter, we will cover the following tasks:

- Creating a database analysis policy
- Creating a role analysis policy
- Creating a context analysis policy
- Creating a combined analysis policy
- Starting and stopping privilege analysis
- Reporting on used system privileges
- Reporting on used object privileges
- Reporting on unused system privileges
- Reporting on unused object privileges
- How to revoke unused privileges
- Dropping the analysis

- **Privilege analysis** is a new security feature, introduced in Oracle Database 19c.
- Privilege analysis is very useful to implement and maintain the least privilege principle by identifying both privileges that users are actually using (used privileges) and those that are only granted to them (unused privileges).

The steps to analyze the used and unused privileges



Introduction

- In this chapter, it is assumed that all users have a **create session** privilege, and in the following table, other privileges and roles granted to the users and roles are listed:

USER/ROLE	HR.EMPLOYEES	OE.ORDERS	ROLES/SYS.PRIVS.
BARBARA			P1_ROLE
NICK			DBA
ALAN	SELECT, INSERT, UP DATE, DELETE		
STEVE			P2_ROLE
P1_ROLE	SELECT		
P2_ROLE		SELECT, INSERT, UP DATE, DELETE	SELECT ANY TABLE, CREATE TABLE

- Depending on your needs, you can create and use four different types of privilege analysis policies that differ in the scope of the analysis. This scope can be:

- An entire database
- Role-based
- Context-based
- Role- and context-based

Creating database analysis policy

- You'll learn to create database privilege analysis policy.
- It analyzes privileges in the whole database (except privileges used by **SYS** user).
- You can use SQL*Plus and Enterprise Manager Cloud Control 12.1.0.3+ (in our case, EM19cR4) to create privilege analysis policies.

Creating database analysis policy

- You'll need an existing user who can create a privilege analysis policy (has CAPTURE_ADMIN role and SELECT ANY DICTIONARY privilege), for example, SYSTEM user.

Creating database analysis policy

1. Create database-wide policy that will capture privileges, which are used by users (except the SYS user).

2. However, to start gathering data about privilege usage, you have to enable the policy.

Creating database analysis policy



1

Login to EM as a user who has appropriate privileges and select **Privilege Analysis** from **Security** drop-down menu:

Creating database analysis policy

Log in to the database as **SYSTEM** user or a user who has appropriate privileges (**CAPTURE_ADMIN** role and **SELECT ANY DICTIONARY** privilege).

2

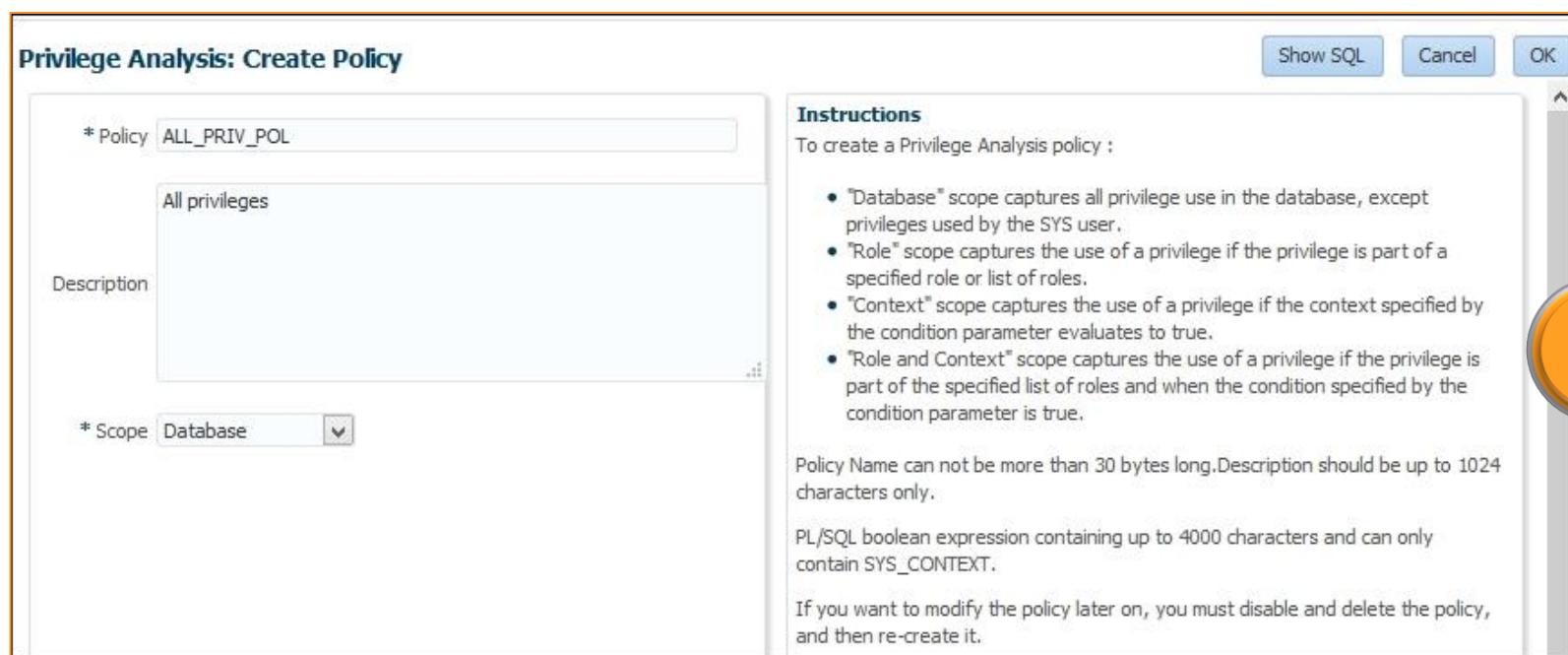


Policies					
		Actions		View	
		Create	Start Capture	Stop Capture	Generate Report
Policy	Active	Type	First Start Time	Last End Time	Capture Scope
ORA\$DEPENDENCY		Database			

3

Click on the **Create** button in the **Policy** section

Creating database analysis policy



4

To create a database policy, choose that scope is **Database**, name the policy, and optionally write a description. Click on the **OK** button.

Creating database analysis policy

You should receive a confirmation message and see your newly created policy listed in the table

5

The screenshot shows the Oracle Database Control interface. At the top, a confirmation message reads: "Privilege Analysis policy ALL_PRIV_POL has been created successfully." Below this, the "Privilege Analysis" section provides a brief description of what it does. The main area is titled "Policies" and contains a table with two rows. The columns are: Policy, Active, Type, First Start Time, Last End Time, Total Capture Duration, and User. The first row for "ALL_PRIV_POL" is active (indicated by a green checkmark) and set to "Database". The second row for "ORA\$DEPENDENCY" is also set to "Database".

Policy	Active	Type	First Start Time	Last End Time	Total Capture Duration	User
ALL_PRIV_POL	Yes	Database				
ORA\$DEPENDENCY	No	Database				

Creating role analysis policy

- You'll create a **role analysis policy** using SQL*Plus and Enterprise Manager Cloud Control 19c (EM).
- The usage of directly and indirectly granted privileges to the roles listed in the policy, will be captured if the roles are active for the session.

Creating role analysis policy

- You'll need an existing user who can create a privilege analysis policy (has a CAPTURE_ADMIN role and a SELECT ANY DICTIONARY privilege), for example, SYSTEM user.

Creating role analysis policy

Log in to the database as **SYSTEM** user or a user who has appropriate privileges (**CAPTURE_ADMIN** role and **SELECT ANY DICTIONARY** privilege).



Login to EM as a user who has appropriate privileges and select **Privilege Analysis** from **Security** drop-down menu:



Click on the **Create** button in the **Policy** section

Creating role analysis policy

Privilege Analysis: Create Policy

* Policy ROLE_PRIV_POL

Usage of privileges granted through listed roles

Description

* Scope Role

Available Roles

DV_PUBLIC
DV_PATCH_ADMIN
DV_STREAMS_ADMIN
DV_GOLDENGATE_ADMIN
DV_XSTREAM_ADMIN
DV_GOLDENGATE_REDO_ACCESS
DV_AUDIT_CLEANUP
DV_DATAPUMP_NETWORK_LINK
DV_REALM_RESOURCE
DV_REALM_OWNER
P2_ROLE

* Roles

Selected Roles

DBA
P1_ROLE

4

Name the policy, select roles, optionally write a description, and click on OK button

Creating role analysis policy

You should receive a confirmation message and see your newly created policy listed in the table

5

The screenshot shows a confirmation message and a table of policies.

Confirmation: Privilege Analysis policy ROLE_PRIV_POL has been created successfully.

Privilege Analysis: Describes the purpose of Privilege Analysis, mentioning it enables finding information about privilege usage for a database according to a specified condition, such as privileges to run an application module or privileges used in a given user session. It analyzes both system privileges and object privileges. To monitor the privileges used for a user's action, you must create and enable a Privilege Analysis policy. Afterward, you can generate a report that describes the used and unused privileges and then from there, revoke (and regrant) privileges as necessary. However, you cannot use privilege analysis to analyze the use of SYS user privileges. Privilege analysis is licensed as part of Oracle Database Vault, but you do not need to enable Database Vault to use it.

Policies:

Actions	View	Create	Start Capture	Stop Capture	Generate Report	Delete
Policy	Active	Type	First Start Time	Last End Time	Capture Scope	T
ALL_PRIV_POL		Database				
ORA\$DEPENDENCY		Database				
ROLE_PRIV_POL		Role				

Creating context analysis policy

- You'll create a context analysis policy.
- After the policy is enabled, it will capture privileges when the condition specified in the policy evaluates to **true**.
- You'll need an existing user who can create a privilege analysis policy (has the **CAPTURE_ADMIN** role and the **SELECT ANY DICTIONARY** privilege), for example, the **SYSTEM** user.

Creating context analysis policy

Log in to the database as **SYSTEM** user or a user who has appropriate privileges (**CAPTURE_ADMIN** role and **SELECT ANY DICTIONARY** privilege).



Login to EM as a user who has appropriate privileges and select **Privilege Analysis** from **Security** drop-down menu:



Click on the **Create** button in the **Policy** section

Creating context analysis policy

Privilege Analysis: Create Policy

* Policy CONT_PRIV_POL

Privileges used by Steve

Description

* Scope Context

SYS_CONTEXT ('USERENV','CURRENT_SCHEMA') = 'SYSTEM'

* Condition

Examples:

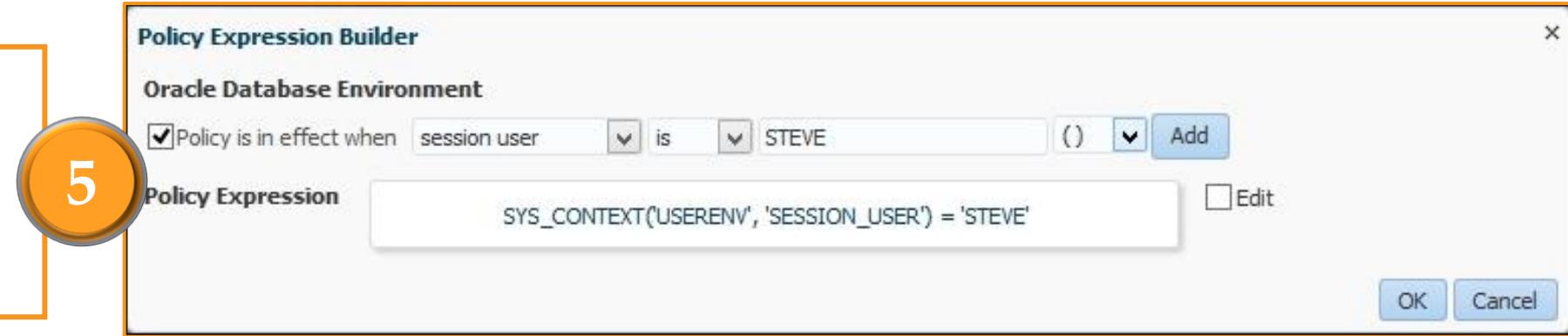
SYS_CONTEXT ('USERENV', 'HOST') NOT IN ('sales_24','sales_12')
SYS_CONTEXT ('USERENV','CURRENT_SCHEMA') = 'SYS'

4

Name the policy and optionally write a description

Creating context analysis policy

Click on the Build Context Expression button. You can enter expression manually or use the built-in help. Click on the OK button.



Creating context analysis policy

Privilege Analysis: Create Policy

* Policy

Privileges used by Steve

Description

* Scope

SYS_CONTEXT('USERENV', 'SESSION_USER') = 'STEVE'

* Condition

Examples:

SYS_CONTEXT ('USERENV', 'HOST') NOT IN ('sales_24','sales_12')
SYS_CONTEXT ('USERENV','CURRENT_SCHEMA') = 'SYS'

6

Make sure that you chose options you wanted and then click on the **OK** button

Creating context analysis policy

You should receive a confirmation message and see your newly created policy listed in the table.

7

The screenshot shows a confirmation message in a yellow box: "Privilege Analysis policy CONT_PRIV_POL has been created successfully." Below this, the "Privilege Analysis" section is described, explaining its purpose and how it monitors privilege usage. At the bottom, a table titled "Policies" lists four entries:

Policy	Active	Type	First Start Time	Last End	Capture
ALL_PRIV_POL		Database			
CONT_PRIV_POL		Context			
ORA\$DEPENDENCY		Database			
ROLE_PRIV_POL		Role			

Creating combined analysis policy

- You'll create a combined analysis policy.
- This type of policy defines that the usage of directly and indirectly granted privilege to specified roles will be gathered if roles are enabled in the session and the context condition is satisfied.
- The context condition can consist of one or more conditions (you can use the AND or OR Boolean operators).

Creating combined analysis policy

- You'll need an existing user who can create a privilege analysis policy (has the CAPTURE_ADMIN role and the SELECT ANY DICTIONARY privilege), for example, the SYSTEM user.

Creating combined analysis policy

Log in to the database as **SYSTEM** user or a user who has appropriate privileges (**CAPTURE_ADMIN** role and **SELECT ANY DICTIONARY** privilege).

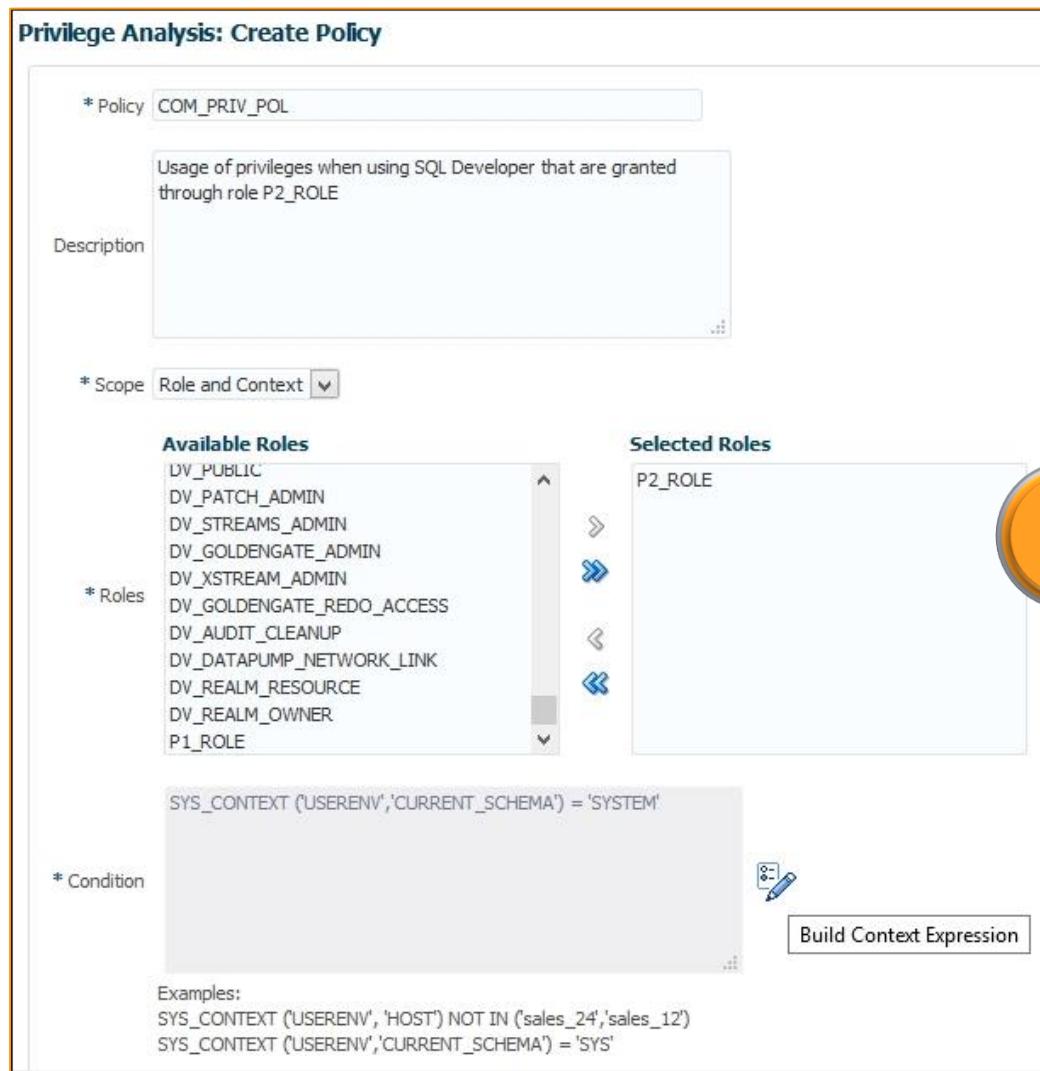


Login to EM as a user who has appropriate privileges and select **Privilege Analysis** from **Security** drop-down menu:



Click on the **Create** button in the **Policy** section

Creating combined analysis policy



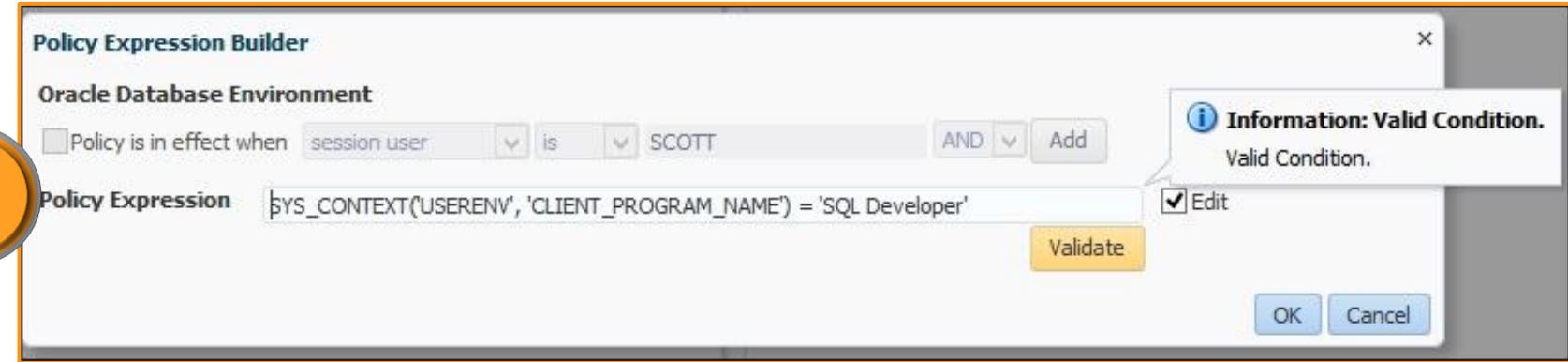
4

Name the policy, select roles, and optionally write a description.
Click on **Build Context Expression**

Creating combined analysis policy

Manually write the policy expression. Click on the **Validate** button and then on the **OK** button

5



Creating combined analysis policy

The screenshot shows the Oracle Database Vault Policies page. At the top, there is a confirmation message: "Privilege Analysis policy COM_PRIV_POL has been created successfully." Below this, the "Privilege Analysis" section provides a brief overview of what it does. The main area is titled "Policies" and contains a table with the following data:

Policy	Active	Type	First Start Time	Last End	Capture
ALL_PRIV_POL		Database			
COM_PRIV_POL		Role and Context			
CONT_PRIV_POL		Context			
ORA\$DEPENDENCY		Database			
ROLE_PRIV_POL		Role			

A large orange arrow points from the top right towards the confirmation message.

6

You should receive a confirmation message and see your newly created policy listed in the table.

Starting and stopping privilege analysis

- To start capturing privileges, you'll enable privilege analysis policies you created previously.
- You'll need an existing user who can manage privilege analysis policies (has the CAPTURE_ADMIN role and the SELECT ANY DICTIONARY privilege), for example, the SYSTEM user.

Starting and stopping privilege analysis

1. Start capturing privileges according to the policy **ALL_PRIV_POL**.
2. Then, execute several statements as the user **ALAN**.
3. The point of those statements is to generate records.

Starting and stopping privilege analysis

4. Stop capturing the privilege usage.
5. Populate `DBA_USED_XXX` and `DBA_UNUSED_XXX` data dictionary views.

Starting and stopping privilege analysis

Log in to the database as **SYSTEM** user or a user who has appropriate privileges (**CAPTURE_ADMIN** role and **SELECT ANY DICTIONARY** privilege).

1

Login to EM as a user who has appropriate privileges and select **Privilege Analysis** from **Security** drop-down menu:

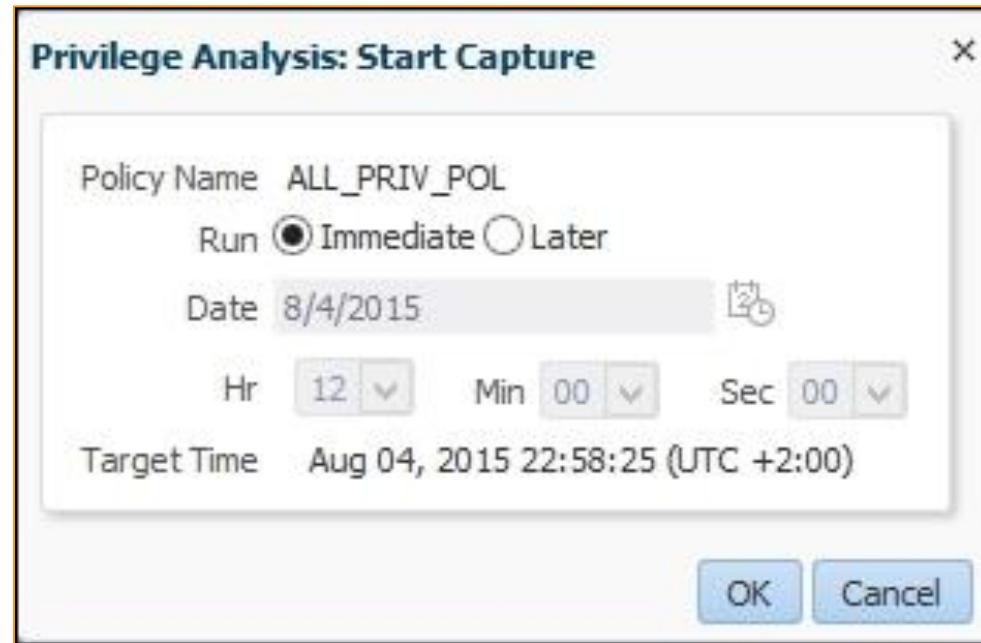
2

Policies					
Actions		View		Actions	
		Create	Start Capture	Stop Capture	Generate Report
Policy	Active	Type	First Start Time	Last End Time	Cap
ALL_PRIV_POL		Database			
COM_PRIV_POL		Role and Context			
CONT_PRIV_POL		Context			
ORA\$DEPENDENCY		Database			
ROLE_PRIV_POL		Role			

3

Select the database policy and click on the **Start Capture** button

Starting and stopping privilege analysis



4

You can either start capture immediately or schedule it. Leave the defaults and click on the **OK** button

Starting and stopping privilege analysis

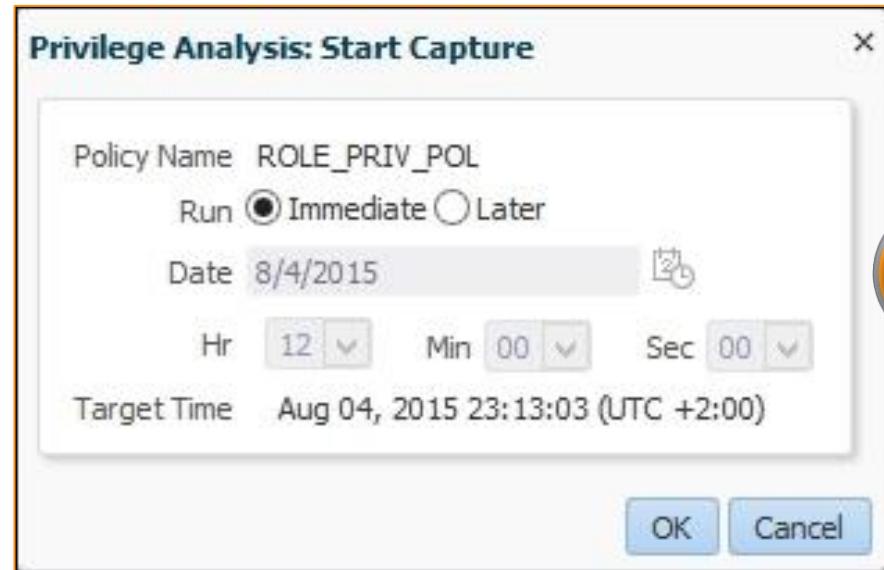
You should receive a confirmation message and see that your policy is active

5

The screenshot shows the Oracle Database Control interface with a yellow border. At the top, there's a confirmation message: "Confirmation Started capture for Privilege Analysis policy ALL_PRIV_POL." Below it, the "Privilege Analysis" section provides a brief overview of what it does. In the center, there's a table titled "Policies" with the following data:

Policy	Active	Type	First Start Time	Capture Scope
ALL_PRIV_POL	Yes	Database	Aug 04, 2015 11:00 PM	Last End Time
COM_PRIV_POL		Role and Context		
CONT_PRIV_POL		Context		
ORA\$DEPENDENCY		Database		
ROLE_PRIV_POL		Role		

Starting and stopping privilege analysis



6

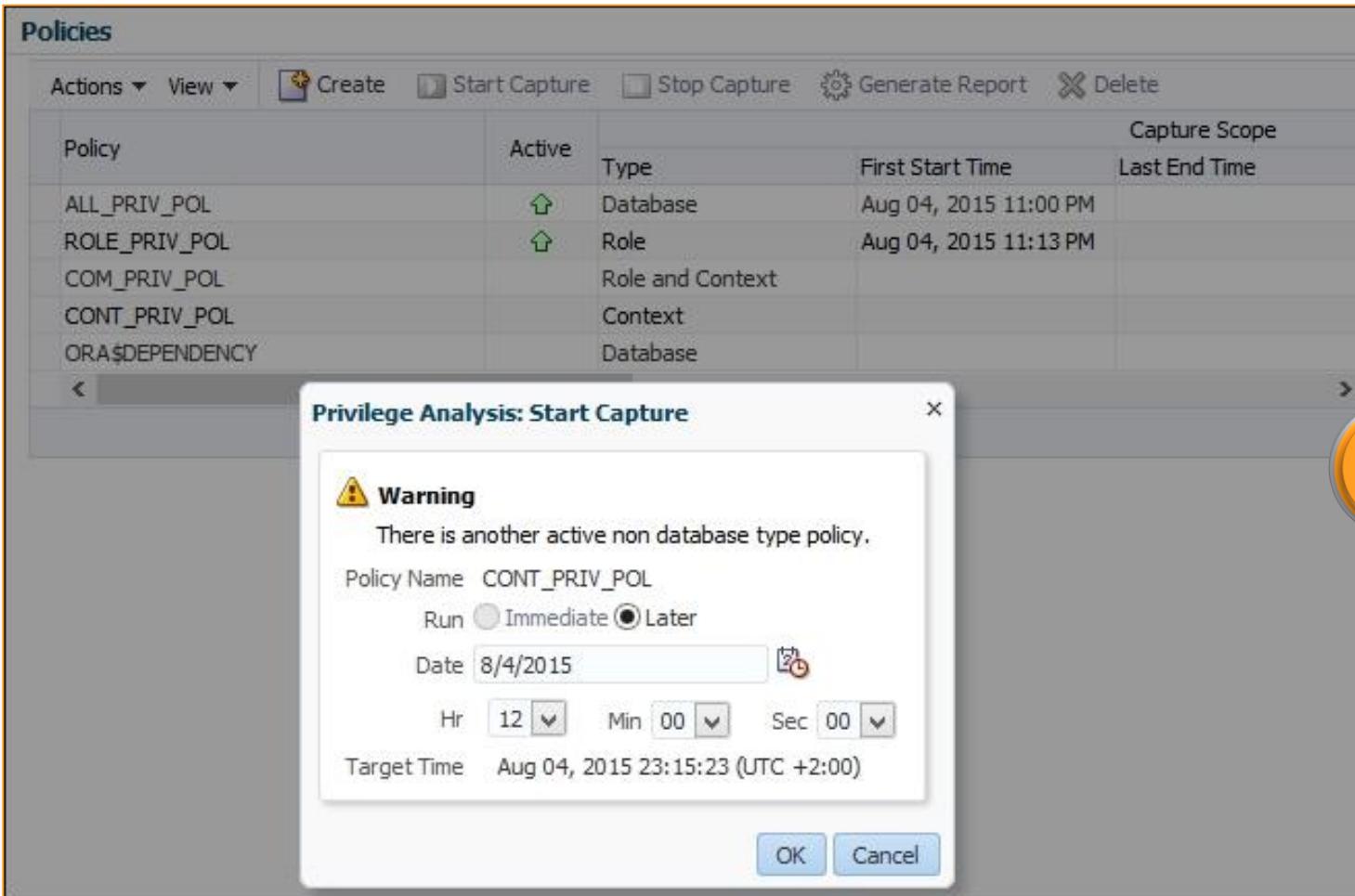
Select the role policy and click on **Start Capture**

You should see under the **Policies** section that both policies are active

7

Policies				
Actions ▾ View ▾		Create	Start Capture	Stop Capture
Policy	Active	Type	First Start Time	Capture Scope
ALL_PRIV_POL	↑	Database	Aug 04, 2015 11:00 PM	
ROLE_PRIV_POL	↑	Role	Aug 04, 2015 11:13 PM	
COM_PRIV_POL		Role and Context		
CONT_PRIV_POL		Context		
ORA\$DEPENDENCY		Database		

Starting and stopping privilege analysis



8

Verify that you can't enable another non-database policy while role policy is active. Select **CONT_PRIV_POL** and click on the **Start Capture** button. You'll receive warning message, and you'll only be able to schedule job to run at later point in time

Starting and stopping privilege analysis

To disable capture, select an active policy (for example, ALL_PRIV_POL) and click on the button Stop Capture

9

Policies	Actions	View	Create	Start Capture	Stop Capture	Generate Report	Delete
Policy	Active	Type	Stop Capture Time	Capture Scope	Last End Time	Total Capture Duration	
ALL_PRIV_POL	Active	Database	Aug 04, 2015 11:00 PM			7hr 26Min	
ROLE_PRIV_POL	Active	Role	Aug 04, 2015 11:13 PM			7hr 13Min	
COM_PRIV_POL		Role and Context					
CONT_PRIV_POL		Context					
ORA\$DEPENDENCY		Database					

Privilege Analysis: Stop Capture

Policy Name: ALL_PRIV_POL

Run Immediate Later

Date: 8/5/2015

Hr: 12 Min: 00 Sec: 00

Target Time: Aug 05, 2015 06:28:58 (UTC +2:00)

Generate Report

OK Cancel

10

Choose to immediately stop capture and tick generate report checkbox. Click on the OK button

Starting and stopping privilege analysis

The screenshot shows the Oracle Database Control interface with the 'Policies' tab selected under 'Privilege Analysis'. A confirmation message box is displayed at the top left:

Confirmation
Capture for Privilege Analysis policy ALL_PRIV_POL has been stopped and Oracle Scheduler Job "I_ALL_PRIV_POL" has been submitted to generate report.

Privilege Analysis

Privilege Analysis enables you to find information about privilege usage for a database according to a specified condition, such as privileges to run an application module or privileges used in a given user session. It analyzes both system privileges and object privileges. To monitor the privileges used for a user's action, you must create and enable a Privilege Analysis policy. Afterward, you can generate a report that describes the used and unused privileges and then from there, revoke (and regrant) privileges as necessary. However, you cannot use privilege analysis to analyze the use of SYS user privileges. Privilege analysis is licensed as part of Oracle Database Vault, but you do not need to enable Database Vault to use it.

Policies

Policy	Active	Capture Scope			
		Type	First Start Time	Last End Time	Total Capture Duration
ROLE_PRIV_POL	Role	Role	Aug 04, 2015 11:13 PM		7hr 17Min
COM_PRIV_POL	Role and Context	Role and Context			
CONT_PRIV_POL	Context	Context			
ORA\$DEPENDENCY	Database	Database			
ALL_PRIV_POL	Database	Database	Aug 04, 2015 11:00 PM	Aug 05, 2015 06:30 AM	7hr 30Min

11

You should receive confirmation message that capture has been stopped and that job has been submitted

Starting and stopping privilege analysis

Refresh page

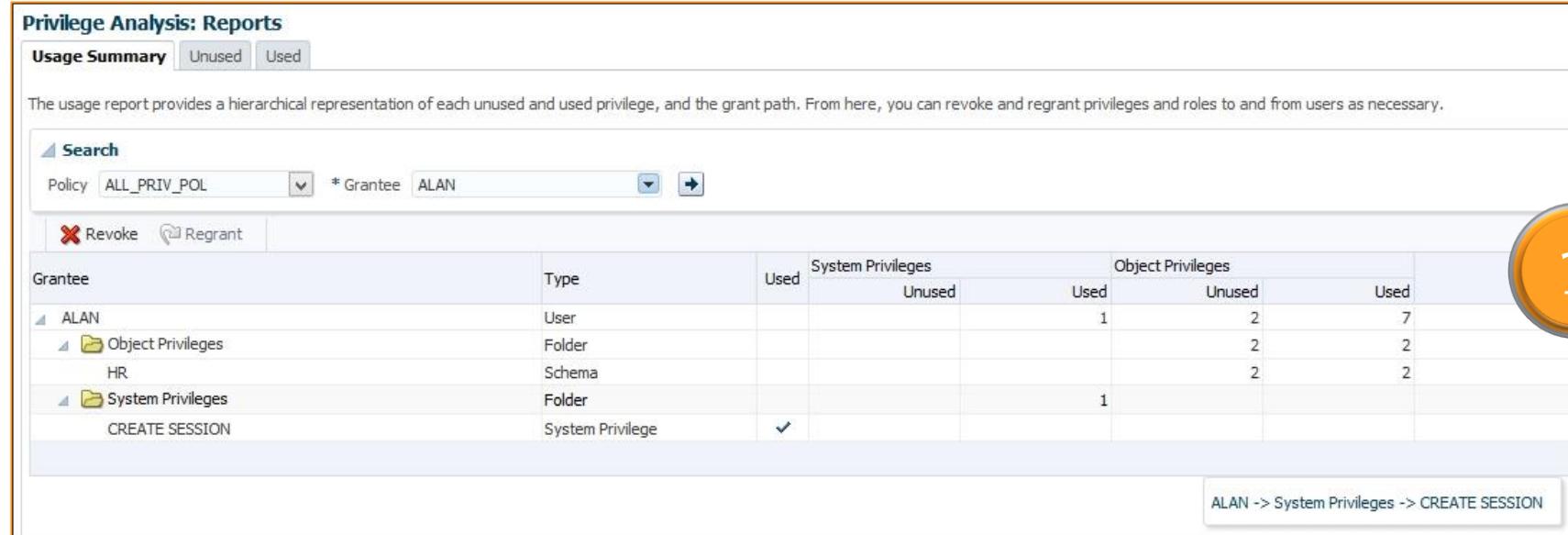
12

Policies												
Policy	Active	Type	Capture Scope			Total Capture Duration	Unused Privileges			Scheduled Jobs	Most Recent Job	
			First Start Time	Last End Time	Users		System	Object	Public		Status	Type
ROLE_PRIV_POL	▲	Role	Aug 04, 2015 11:13 PM			7Hr 18Min						
COM_PRIV_POL		Role and Context										
CONT_PRIV_POL		Context										
ORA\$DEPENDENCY		Database					19					
ALL_PRIV_POL		Database	Aug 04, 2015 11:00 PM	Aug 05, 2015 06:30 AM	20	7Hr 30Min	492	13683			✓ Report Generation	

Reporting on used system privileges

- You'll view collected data about the usage of system privileges during a capture interval.
- You'll need an existing user who can create a privilege analysis policy (has the CAPTURE_ADMIN role and the SELECT ANY DICTIONARY privilege), for example, the SYSTEM user.

Reporting on used system privileges



The usage report provides a hierarchical representation of each unused and used privilege, and the grant path. From here, you can revoke and regrant privileges and roles to and from users as necessary.

Search

Policy ALL_PRIV_POL * Grantee ALAN

Revoke **Regrant**

Grantee	Type	Used	System Privileges		Object Privileges		
			Unused	Used	Unused	Used	
ALAN	User		1	2	7		
Object Privileges	Folder			2	2		
HR	Schema			2	2		
System Privileges	Folder		1				
CREATE SESSION	System Privilege	✓					

ALAN -> System Privileges -> CREATE SESSION

1

In EM 19c, after you have generated the report, select the policy and from Actions dropdown menu, select Reports. The Usage Summary report will open

Reporting on used system privileges

Click on the tab **Used** and choose **All** for Match radio button, **Policy: ALL_PRIV_POL**, **User Name: ALAN**, and click on the **Search** button.

2

SL#	Policy	User Name	Used Role	System Privileges	Path	Admin/Grant Option
1	ALL_PRIV_POL	ALAN	ALAN	CREATE SESSION	ALAN	False
2	ALL_PRIV_POL	ALAN	PUBLIC		PUBLIC	False
3	ALL_PRIV_POL	ALAN	PUBLIC		PUBLIC	False
4	ALL_PRIV_POL	ALAN	PUBLIC		PUBLIC	False
5	ALL_PRIV_POL	ALAN	PUBLIC		PUBLIC	False
6	ALL_PRIV_POL	ALAN	ALAN		ALAN	False
7	ALL_PRIV_POL	ALAN	PUBLIC		PUBLIC	False
8	ALL_PRIV_POL	ALAN	ALAN		ALAN	False

Reporting on used system privileges

The screenshot shows the 'Privilege Analysis: Reports' interface. The 'Used' tab is selected. The search criteria are set to 'All' for 'Policy' (ROLE_PRIV_POL), 'User Name' (NICK), and 'System Privileges'. The results table displays four rows of data:

SL#	Policy	User Name	Used Role	Path	System Privileges	Object Owner	Object Name
1	ROLE_PRIV_POL	NICK	EM_EXPRESS_BASIC	NICK,DBA,EM_EXPRESS_ALL,EM_EXPRESS_BASIC	CREATE SESSION		
2	ROLE_PRIV_POL	NICK	DATAPUMP_EXP_FULL_DATABASE	NICK,DBA,DATAPUMP_EXP_FULL_DATABASE,EXP_FULL_D...	CREATE TABLE		
3	ROLE_PRIV_POL	NICK	DATAPUMP_EXP_FULL_DATABASE	NICK,DBA,DATAPUMP_EXP_FULL_DATABASE	CREATE TABLE		
4	ROLE_PRIV_POL	NICK	OLAP_DBIA	NICK,DBA,OLAP_DBIA	SELECT ANY TABLE	OE	CUSTOMERS

Columns Hidden: 9

3

If you haven't generated report for the role policy, do it now and return to this tab (the Used tab). Find all records generated by ROLE_PRIV_POL for user Nick (who has a DBA role).

Reporting on used object privileges

- You'll view collected data about the usage of object privileges during the capture interval.
- You'll need an existing user who can create a privilege analysis policy (has the CAPTURE_ADMIN role and the SELECT ANY DICTIONARY privilege), for example, the SYSTEM user.

Reporting on used object privileges

Privilege Analysis: Reports

Usage Summary Unused **Used**

Used Privileges

Search

Match All Any

Policy ALL_PRIV_POL

User Name ALAN

System Privileges

Object Name

Advanced Saved Search UsedPrevVOCriteria

Search Reset Save...

View Export to Spreadsheet

SL#	Policy	User Name	Object Privileges	Used Role	Path	Object Owner	Object Name
1	ALL_PRIV_POL	ALAN	SELECT	ALAN	ALAN	HR	EMPLOYEES
2	ALL_PRIV_POL	ALAN	DELETE	ALAN	ALAN	HR	EMPLOYEES
3	ALL_PRIV_POL	ALAN	SELECT	PUBLIC	PUBLIC	SYS	DUAL
4	ALL_PRIV_POL	ALAN	SELECT	PUBLIC	PUBLIC	SYS	DUAL
5	ALL_PRIV_POL	ALAN	EXECUTE	PUBLIC	PUBLIC	SYS	DBMS_APPLICATION_INFO
6	ALL_PRIV_POL	ALAN	EXECUTE	PUBLIC	PUBLIC	SYS	DBMS_OUTPUT
7	ALL_PRIV_POL	ALAN	SELECT	PUBLIC	PUBLIC	SYSTEM	PRODUCT_PRIVS
8	ALL_PRIV_POL	ALAN		ALAN	ALAN		

Columns Hidden 9

1

In EM 19c, after you have generated the report, select the policy, and from **Actions** drop-down menu, select **Reports**. The **Usage Summary** report will open. Click on the **Used** tab and verify that the user **Alan** has used the **SELECT** and **DELETE** privileges while **ALL_PRIV_POL** has been active

Reporting on unused system privileges

- You'll view the collected data about the unused system privileges during the capture interval.
- You'll need an existing user who can create a privilege analysis policy (has the CAPTURE_ADMIN role and the SELECT ANY DICTIONARY privilege), for example, the SYSTEM user.

Reporting on unused system privileges

- To view report about the unused system privileges in EM19c, see instructions to view the used system privileges and under **Privilege Analysis: Reports**, choose the **Unused** tab instead of the **Used** tab.

Reporting on unused object privileges

- You'll view collected data about the unused object privileges during the capture interval.
- You'll need an existing user who can create a privilege analysis policy (has the CAPTURE_ADMIN role and the SELECT ANY DICTIONARY privilege), for example, the SYSTEM user.

Reporting on unused object privileges

Privilege Analysis: Reports

Usage Summary **Unused** Used

Unused Privileges

Search Advanced Saved Search DbaUnusedPrvsVOCriteria

Match: All Any

Policy: ALL_PRIV_POL

Grantee: ALAN

System Privileges:

Object Owner:

Object Name:

Search Reset Save...

View Export to Spreadsheet

SL#	Policy	Grantee	Grantee Type	System Privileges	Object Privileges	Object Owner	Object Name	Object Type	Path
1	ALL_PRIV_POL	ALAN	USER		INSERT	HR	EMPLOYEES	TABLE	ALAN
2	ALL_PRIV_POL	ALAN	USER		UPDATE	HR	EMPLOYEES	TABLE	ALAN

Columns Hidden 3

SL#	Policy	Grantee	Grantee Type	System Privileges	Object Privileges	Object Owner	Object Name	Object Type	Path
1	ALL_PRIV_POL	ALAN	USER		INSERT	HR	EMPLOYEES	TABLE	ALAN
2	ALL_PRIV_POL	ALAN	USER		UPDATE	HR	EMPLOYEES	TABLE	ALAN

1

In EM 19c, after you have generated the report, select the policy, and from **Actions** drop-down menu, select **Reports**. The **Usage Summary** report will open. Click on the **Unused** tab and verify that the user **Alan** hasn't used the **INSERT** and **UPDATE** privileges while **ALL_PRIV_POL** has been active.

How to revoke unused privileges

- You can manually revoke unused privileges one by one from users, write your own scripts to complete that task, or use Enterprise Manager Cloud Control 19c.
- You'll use EM19c to efficiently revoke unused privileges based on reports you generated in the previous recipes.

How to revoke unused privileges

- In EM 19c, there is another excellent option to create a new role based on privilege analysis results.
- This way, you won't change an existing role (and affect other users and roles who have that role), but create a new one and afterwards revoke the old role and grant a newly created one.

How to revoke unused privileges

Privilege Analysis: Create Role

Use this feature to create a new role from Privilege Analysis results. The role can have used or unused system/object privileges and roles.

Note: If the logged in user does not have sufficient privileges, then the user will be prompted to provide SYSDBA credentials.

* Policy Name ALL_PRIV_POL

* Role Name

Unused

Used

Directly Granted System Privileges All None Customize

Directly Granted Object Privileges All None Customize

Directly Granted Roles All None Customize

Cancel

OK

1

You can select it from the **Actions** menu (**Create Role**). The configuration part of a process for creating a new role is shown

Dropping the analysis

- In this recipe, you'll drop an existing privilege analysis policy.
- It has to be disabled before dropping; otherwise, you'll receive an error.
- You'll need an existing user who can manage privilege analysis policies (has the **CAPTURE_ADMIN** role and the **SELECT ANY DICTIONARY** privilege), for example, the **SYSTEM** user and an existing privilege analysis policy.

Dropping the analysis

- In EM 19c under the **Policies** section, select policy you want to drop and click on the **Delete** button.



Oracle Database 19c Detective Controls

Unified Auditing

In this chapter, we will cover the following tasks:

- Enabling the Unified Auditing mode
- Configuring whether loss of audit data is acceptable
- Which roles do you need to have to be able to create audit policies and to view audit data?
- Auditing RMAN operations
- Auditing Data Pump operations
- Auditing Database Vault operations
- Creating audit policies to audit privileges, actions, and roles under specified conditions
- Enabling an audit policy
- Finding information about audit policies and audited data
- Auditing application contexts
- Purging audit trail
- Disabling and dropping audit policies

- Unified Auditing is a new feature in Oracle Database 19c, and it introduces new auditing architecture.
- Some of the characteristics of unified auditing are:
 - A single audit trail
 - Being based on a read-only table
 - Extensible Audit Framework for additional columns
 - The separation of audit administration with new roles
 - Auditing performance is better, especially when used in the queued-write mode

Traditional Audit Trails

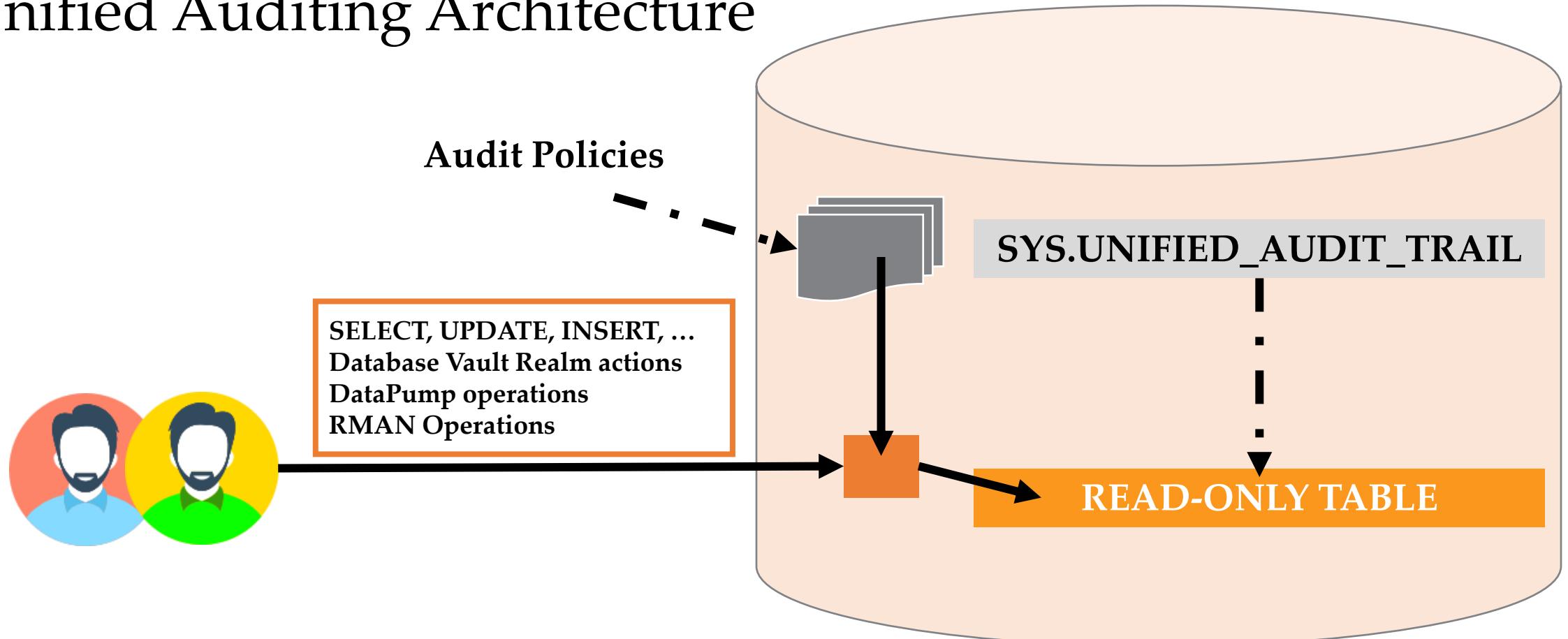
SYS.AUD\$
SYS.FGA_LOG\$
V\$XML_AUDIT_TRAIL
DBA_COMMON_AUDIT_TRAIL
DVSYS.AUDIT_TRAIL\$
OS files



Unified Audit Trails

SYS.UNIFIED_AUDIT_TRAIL

- Unified Auditing Architecture



Enabling Unified Auditing mode

- In Oracle Database 19c, unified auditing is not enabled by default.
- The process of enabling it is simple and equivalent to enabling of other database options.
- You'll need to shut down the database.

Enabling Unified Auditing mode

- When database is upgraded to 19c, by default, it uses the traditional way of auditing.
- However, when you directly install a new database 19c, default auditing is set to mixed auditing mode.
- In both cases, the procedure to enable the unified auditing mode is the same.

Enabling Unified Auditing mode

- After you enable the unified auditing mode, traditional auditing doesn't work anymore.
- Old audit instance parameters (`AUDIT_TRAIL`, `AUDIT_FILE_DEST`, `AUDIT_SYSLOG_LEVEL`, and `AUDIT_SYS_OPERATIONS`) are disregarded.
- Predefined unified audit policies that are enabled by default are:
 - `ORA_SECURECONFIG` (database versions: 12.1.0.1, 12.1.0.2)
 - `ORA_LOGON_FAILURES` (Oracle Database 12.1.0.2)

Enabling Unified Auditing mode

- Predefined unified audit policies

Predefined audit policy	Oracle Database 12.1.0.1	Oracle Database 12.1.0.2
ORA_RAS_POLICY_MGMT	Yes	Yes
ORA_DATABASE_PARAMETER	Yes	Yes
ORA_RAS_SESSION_MGMT	Yes	Yes
ORA_ACCOUNT_MGMT	Yes	Yes
ORA_SECURECONFIG	Yes	Yes
ORA_LOGON_FAILURES	No	Yes
ORA_CIS_RECOMMENDATIONS	No	Yes
ORA_DV_AUDPOL	No	Yes

Enabling Unified Auditing mode

- If you execute the following statement in both 12.1.0.1 and 12.1.0.2 database versions, as a user who has the **audit_admin** or **dba** role:

```
SQL> select audit_option from audit_unified_policies  
where policy_name='ORA_SECURECONFIG'  
order by 1;
```

Enabling Unified Auditing mode

- In Oracle Database 19cR1 **Standard Edition (SE)**, when you enable unified auditing mode and query the v\$option view to verify that it's enabled you may see the following:

PARAMETER	VALUE
<hr/>	
Unified Auditing	FALSE

Configuring whether loss of audit data is acceptable

- You'll learn to set whether audit data is queued in memory or is immediately written to audit trail.
- You'll need an existing user who has the **audit_admin** role.

Configuring whether loss of audit data is acceptable

- The default value for a write mode is the queued-write mode.
- In this mode, audit data is stored in SGA queues and later automatically persisted in the read-only table in the **AUDSYS** schema in the **SYSAUX** tablespace.
- You can also manually flush content of memory queues to the disk:

```
SQL>EXEC SYS.DBMS_AUDIT_MGMT.FLUSH_UNIFIED_AUDIT_TRAIL;
```

Which roles do you need to have to be able to create audit policies and to view audit data?

- In this recipe, you're going to create two users (for example, **jack** and **jill**).
- Jack's job is to implement auditing requirements and to make sure that auditing is functioning properly.
- Jill is an auditor and her job is to analyze audit data.
- You'll need an existing user who has the DBA role.

Which roles do you need to have to be able to create audit policies and to view audit data?



AUDIT_ADMIN

Manages audit configuration & audit trail.



AUDIT_VIEWER

Analysis audit data.

Which roles do you need to have to be able to create audit policies and to view audit data?

- Grant the **AUDIT_ADMIN** role to the newly created user **jack** because that role enables him to create, alter, enable, disable, and drop audit policies, view audit data, and manage the unified audit trail.
- Grant the **AUDIT_VIEWER** role to the user **jill** because that role enables her to view audit data.

Which roles do you need to have to be able to create audit policies and to view audit data?

- To test what can and can't be done as a user who has the **audit_viewer** role, connect to the database as **jill** and try to create the unified audit policy **jill_policy**:

```
SQL> connect jill
```

```
SQL> create audit policy jill_policy  
actions delete on oe.orders;  
actions delete on oe.orders  
*
```

```
ERROR at line 2:  
ORA-00942: table or view does not exist
```

Which roles do you need to have to be able to create audit policies and to view audit data?

- Even if you grant object privileges on the **oe.orders** table to **jill**, she won't be able to create unified audit policy because she doesn't have the **audit_admin** role or the **AUDIT SYSTEM** system privilege:

```
SQL> conn maja
```

```
SQL> grant select,delete on oe.orders to jill;
```

```
SQL> connect jill
```

Which roles do you need to have to be able to create audit policies and to view audit data?

```
SQL> create audit policy jill_policy  
actions delete on oe.orders;  
actions delete on oe.orders  
*  
ERROR at line 2:  
ORA-01031: insufficient privileges
```

Which roles do you need to have to be able to create audit policies and to view audit data?

- Revoke select and delete on the `oe.orders` table from Jill:

```
SQL> connect maja
```

```
SQL> revoke select,delete on oe.orders from jill;
```

```
Revoke succeeded.
```

Which roles do you need to have to be able to create audit policies and to view audit data?

- Grant the AUDIT SYSTEM privilege to jill and again try to create the audit policy jill_policy:

```
SQL> grant audit system to jill;  
SQL> connect jill
```

```
SQL> create audit policy jill_policy  
actions delete on oe.orders;  
Audit policy created.
```

Which roles do you need to have to be able to create audit policies and to view audit data?

- Drop the unified audit policy **jill_policy** and revoke the **AUDIT SYSTEM** privilege from **jill**:

```
SQL> drop audit policy jill_policy;
```

Audit Policy dropped.

```
SQL> connect maja
```

```
SQL> revoke audit system from jill;
```

Which roles do you need to have to be able to create audit policies and to view audit data?

- View audit data:

```
SQL> connect jill
```

```
SQL> select dbusername, action_name from unified_audit_trail  
where unified_audit_policies='ORA_SECURECONFIG';
```

- Also, a user who has the audit_viewer role can access information about defined and enabled unified audit policies.

Which roles do you need to have to be able to create audit policies and to view audit data?

- Throughout this chapter, you'll use a user who has the **audit_admin** role (for example, **jack**).
- So only test you'll do right now is to enable the predefined audit policy **ORA_ACCOUNT_MGMT** and then to disable it:

```
SQL> connect jack
```

```
SQL> audit policy ora_account_mgmt;  
Audit succeeded.  
SQL> noaudit policy ora_account_mgmt;  
Noaudit succeeded.
```

Auditing RMAN operations

- You'll see that RMAN operations are audited by default.
- In this, we assume that database is in the ARCHIVELOG mode.
- You'll need an existing user who has the SYSBACKUP privilege (for example, `tom`) and an existing user who has the dba role (for example, `maja`).

Auditing RMAN operations

- When the mixed or unified auditing mode is enabled, RMAN operations are automatically audited.
- Perform the backup of the tablespace **EXAMPLE**.
- Then intentionally cause a problem by removing the datafile.
- Afterwards, perform restore and recover RMAN operations.

Auditing RMAN operations

- The whole point of the example is to execute several RMAN operations.
- In the `unified_audit_trail` data dictionary view, there are several columns that contain data pertaining to the RMAN events.
- Their names start with RMAN, so it's easy to find them.

Auditing RMAN operations

- You should get similar result to this one:

DBUSERNAME	RMAN_OPERATION
TOM	Backup
TOM	List
TOM	Restore
TOM	Recover

Auditing Data Pump operations

- You can audit Data Pump export, import, or both export and import operations by creating audit policies.
- You'll need an existing user who has the `audit_admin` role (for example, **jack**).

Auditing Data Pump operations

- Also, it is assumed that directory for export operations (for example, `my_dir`) is created and a user (for example, `maja`) who is going to perform the Data Pump export has read and write privileges on the directory.

```
SQL> CREATE DIRECTORY my_dir AS '/u01/app/oracle/oradata/export';
```

```
SQL> grant read, write ON DIRECTORY my_dir to maja;
```

Auditing Database Vault operations

- You'll learn to audit Oracle Database Vault events.
- You'll need to use Oracle Database 19c, which has Oracle Database Vault enabled and at least some of the components configured.
- Also, you'll need an existing user who has the **audit_admin** role.

Auditing Database Vault operations

- To create an audit policy that captures Oracle Database Vault events, specify **ACTIONS COMPONENT = DV <action> ON <object>**.
- Define the audit policy **dbv_policy** that encapsulates the rules.
- In the unified audit trail, Oracle Database Vault-specific audit data is stored in the columns whose name starts with **DV_**.

Auditing Database Vault operations

- When you are using Oracle Database Vault, you can also additionally secure your auditing infrastructure by creating a realm around the **AUDIT_ADMIN** and **AUDIT_VIEWER** roles.
- This allows you to control who can grant those roles.

Creating audit policies to audit privileges, actions and roles under specified conditions

- You will create several unified audit policies.
- You'll need two existing users:

- A user who has the `audit_admin` role (for example, `jack`)
- A user who has the `create session` privilege (for example, `john`)

Creating audit policies to audit privileges, actions and roles under specified conditions

- Also, you should create the roles **hr_role** and **oe_role** as stated here and grant **hr_role** to the user **john**.

```
SQL> create role hr_role;
SQL> grant select any table, create table to hr_role;
SQL> grant insert on hr.departments to hr_role;
SQL> create role oe_role;
SQL> grant drop any table to oe_role;
SQL> grant select, update on oe.orders to oe_role;
SQL> grant oe_role to hr_role;
SQL> grant hr_role to john;
```

Creating audit policies to audit privileges, actions and roles under specified conditions

- When you create a unified audit policy, it is stored in the first-class object owned by **SYS** schema.
- Create the audit policy **my_policy1**.
- Create the audit policy **role_con_policy**.

Creating audit policies to audit privileges, actions and roles under specified conditions

- The Role **HR_ROLE** has to exist at the time audit policy **role_con_policy** is created because if it doesn't exist you will get an error message:

```
ERROR at line 2:  
ORA-01919: role 'HR_ROLE' does not exist
```

- Create the audit policy **hr_policy**, the way it is written, audit records will be generated for **select**, **insert**, and **update** operations on all objects and for delete operations on **hr.departments**.

Creating audit policies to audit privileges, actions and roles under specified conditions

- Create the audit policy **oe_policy**, which will be used in order to audit all actions on table orders in oe schema.
- In Oracle Database 12.1.0.1 due to the bug, audit records for this policy are not generated.
- Workaround is to specify one by one actions instead of using keyword **ALL**.

Enabling audit policy

- You will learn to use different options to enable unified audit policies.
- You'll need an existing user who has the **audit_admin** role (for example, **jack**) and several other existing users (for example, **john**, **maja**, and **zoran**).

Enabling audit policy

- Define **BY** list, which means that only user(s) listed on that list will be affected by the policy.
- Define that audit policy **hr_policy** is applied to all users, but only successful operations will generate audit records.
- Define **EXCEPT** list.

Enabling audit policy

- Enable audit policy using default options, which means that `role_con_policy` will affect all users for both successful and unsuccessful events.
- You can't use both `BY` and `EXCEPT` lists for the same policy statement.

Finding information about audit policies and audited data

- You will view audited data and find information about unified audit policies.

Finding information about audit policies and audited data

- You'll need three existing users:

- A user who has `audit_admin` role (for example, `jack`)
- A user who has `hr_role` and `oe_role` (for example, `john`), created in recipe `Creating audit policies to audit privileges, actions and roles under specified conditions`
- A user who has `admin_viewer` role (for example, `jill`)

- Also, you'll need to connect to the database as **SYS** user.

Auditing application contexts

- You will configure auditing of information contained in an application context.
- You'll need an existing (or predefined) application context and a user who has the **audit_admin** role (for example, **jack**).

Auditing application contexts

- Audit custom application contexts:

NAMESPACE	ATTRIBUTE	USER_NAME
USERENV	HOST	JILL
USERENV	SERVICE_NAME	ALL USERS
USERENV	SESSION_USER	ALL USERS

Auditing application contexts

- If needed, execute the following statement as the user **jack**:

```
SQL>EXEC SYS.DBMS_AUDIT_MGMT.FLUSH_UNIFIED_AUDIT_TRAIL;
```

Auditing application contexts

- Audit records:

APPLICATION_CONTEXTS

(USERENV,SERVICE_NAME=SYS\$USERS); (USERENV,SESSION_USER=JACK)
(USERENV,SERVICE_NAME=SYS\$USERS); (USERENV,SESSION_USER=JILL);
(USERENV,HOST= dbhost.orapassion.com)

Auditing application contexts

- To disable auditing of application contexts, you should use the **NOAUDIT** command:

```
SQL> connect jack
```

```
SQL> NOAUDIT CONTEXT NAMESPACE USERENV  
ATTRIBUTES HOST BY jill;
```

Purging audit trail

- You can clean up audit data manually or by scheduling clean up job.
- You'll need a user who has the **audit_admin** role (for example, **jack**).

Purging audit trail

- By default, `USE_LAST_ARCH_TIMESTAMP` is set to `TRUE`.
- It means that only records created before that time will be deleted.
- If you set that parameter to `FALSE`, all records will be deleted.

Purging audit trail

- In multitenant environment, use CONTAINER clause as well (CONTAINER => DBMS_AUDIT_MGMT.CONTAINER_CURRENT or DBMS_AUDIT_MGMT.CONTAINER_ALL).

Disabling and dropping audit policies

- You will learn to disable and drop audit policies.
- You'll need an enabled unified audit policy (for example, `oe_policy`) and a user who has the `audit_admin` role (for example, `jack`).

Disabling and dropping audit policies

- Check that the audit policy `oe_policy` is enabled.
- Disable it.
- To be able to drop audit policy, you have to disable it first.
- Drop the audit policy `oe_policy`.



Oracle Database 19c Detective Controls

Database Vault

We will cover the following tasks:

- Registering Database Vault
- Preventing users from exercising system privileges on schema objects
- Securing roles
- Preventing users from executing a specific command on a specific object
- Creating a rule set
- Creating a secure application role
- Using Database Vault to implement that administrators cannot view data
- Running Oracle Database Vault reports
- Disabling Database Vault
- Re-enabling Database Vault

Introduction

- 
- You are going to learn to create and appropriately use realms, rules, rule sets, command rules, factors, and secure application roles.

- We assume that database is up and running, and each user has at least **create session** a privilege.
- Also, you will use Oracle Enterprise Manager Cloud Control 19c.
- A **SYS** user, because he is the most powerful user, will be used to test that security is correctly enforced (even for him).

Registering Database Vault

In Oracle Database 19c process of configuring and enabling Database Vault is different than in Oracle Database 11g.

You will learn to register Oracle Database Vault in multitenant environment in two situations:

- When Oracle Database 19c is already installed
- During the installation of Oracle Database 19c

Registering Database Vault

- You'll need an existing common user who has a privilege to create users and grant `create session` and set container privileges (for example, `c##maja`).

Registering Database Vault

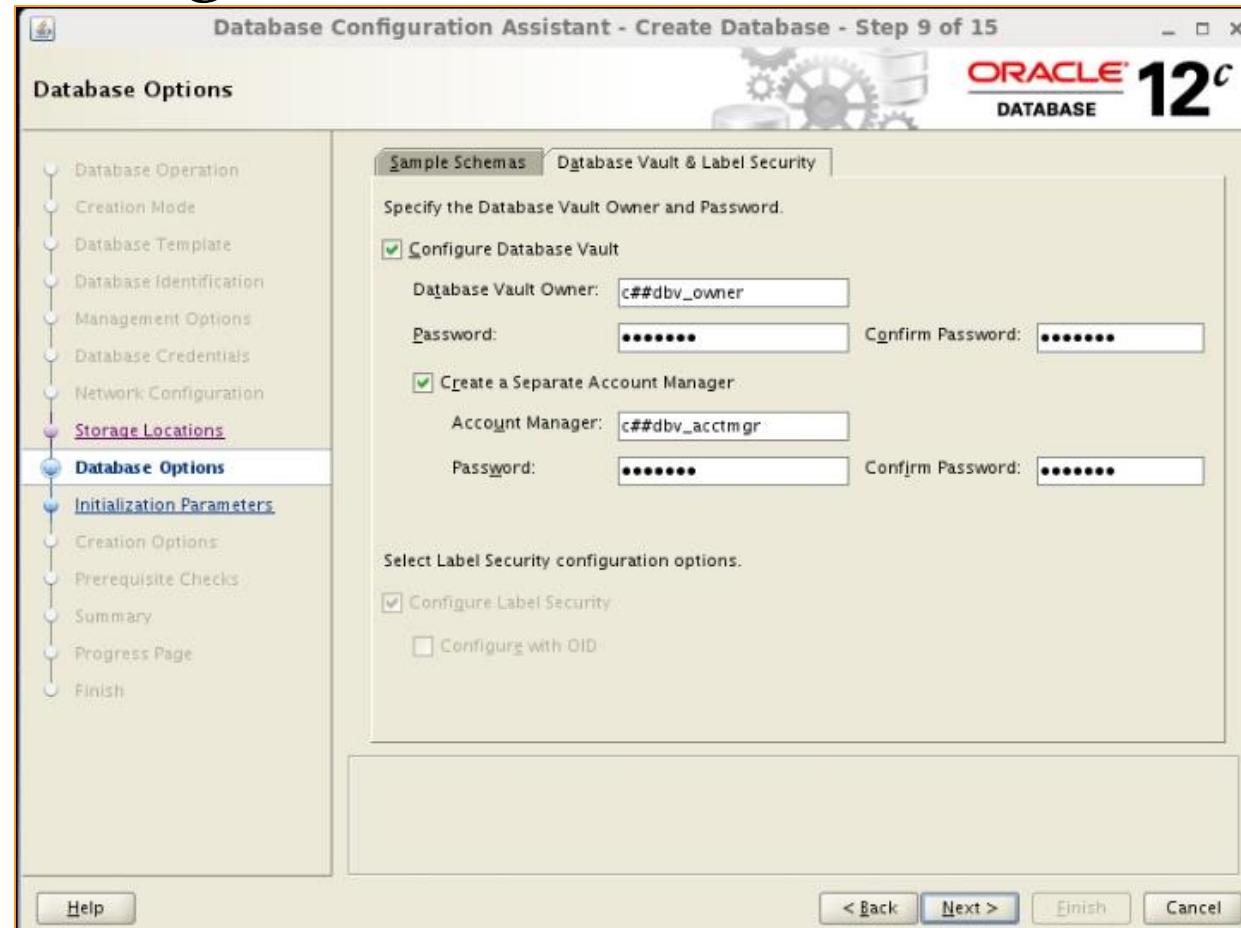
- After you register Oracle Database Vault with Oracle Database 19c, there are number of changes in the Oracle Database.
- Some database parameters change values, separation of duties is enabled by revoking privileges from some roles and by creating new users.

Registering Database Vault

- You use **Database Configuration Assistant (DBCA)** when you configure Database Vault during the database installation.
- When you get to **Database Options**, click on tab **Database Vault & Label Security**.
- Select both available checkboxes and fill out text fields to create users: **Database Vault Owner** and **Account Manager**.

Registering Database Vault

- Using DBCA to register



Preventing users from exercising system privileges on schema objects

- To prevent users to exercise system privileges (such as **select any table**), you are going to first create a **realm** and then you are going to change it to a **mandatory realm**.
- The mandatory realm further restricts access to protected objects.
- Schema owners and users with object privileges cannot access mandatory realm-secured objects if they are not authorized in realm.

Preventing users from exercising system privileges on schema objects

- You'll need an existing common user who has a DBA role in the pluggable database **PDB1** (for example, **c##ernesto**).

Preventing users from exercising system privileges on schema objects

- The difference between **participant** and **owner** of the realm is that a realm participant can only exercise system privileges on realm-secured objects.
- Whereas an owner besides that can grant object privileges on realm-secured objects to other users and roles.

Preventing users from exercising system privileges on schema objects

- Verify that the **SYS** user can't create a user, after Database Vault is registered:

```
SQL> connect sys@pdb1 as sysdba
```

Enter password:
Connected.

```
SQL> create user usr1 identified by oracle;  
create user usr1 identified by oracle
```

*

```
ERROR at line 1:  
ORA-01031: insufficient privileges
```

Preventing users from exercising system privileges on schema objects

- Verify that after you created realm **HR_Realm**, the **SYS** user can't access data in the table **HR.EMPLOYEES**.

```
SQL> connect sys@pdb1 as sysdba  
Enter password: Connected.  
  
SQL> select count(*) from hr.employees;  
  
select count(*) from hr.employees  
*ERROR at line 1:  
ORA-01031: insufficient privileges
```

Preventing users from exercising system privileges on schema objects

- This is the expected behavior because realm protects secured objects from users who try to use their system privileges.
- In our example, **SYS** tried to use **SELECT ANY TABLE**, and because he doesn't have direct object privilege (**SELECT** on **HR.EMPLOYEES**), he is restricted from selecting data in the table **HR.EMPLOYEES**.

Preventing users from exercising system privileges on schema objects

```
SQL> conn c##ernesto@pdb1
```

Enter password: Connected.

```
SQL> select count(*) from hr.employees;
```

COUNT(*)

107

Preventing users from exercising system privileges on schema objects

- After mandatory realm is created, the user **usr1** can't access data in the table **HR.EMPLOYEES** because he/she is not authorized in the realm.

```
Enter password:  
Connected.
```

```
usr1@CDB1> select count(*) from hr.employees;
```

```
select count(*) from hr.employees  
*
```

```
ERROR at line 1:  
ORA-01031: insufficient privileges
```

Preventing users from exercising system privileges on schema objects

- The same principle applies even to the schema owner (HR)

```
SQL> connect usr1@pdb1  
SQL> connect hr@pdb1
```

Enter password:
Connected.

```
SQL> select count(*) from hr.employees;
```

```
select count(*) from hr.employees  
*
```

ERROR at line 1:
ORA-01031: insufficient privileges

- In Preventing users from exercising system privileges on schema objects, you secured the table **HR.EMPLOYEES** by creating the **HR_**realm realm, and afterwards, you edit it and made it mandatory.
- You'll learn to protect roles using a realm and a mandatory realm.
- You'll need to use a **SYS** user

Securing roles

- After we created a realm, the **SYS** user (or any user that is not authorized in realm) cannot grant the realm-protected role:

```
SQL> connect usr1@pdb1
SQL> connect hr@pdb1
```

```
SQL> connect sys@pdb1 as sysdba
```

```
Enter password:  
Connected.
```

```
SQL> grant role1 to usr1;
```

```
grant role1 to usr1  
*
```

```
ERROR at line 1:  
ORA-47410: Realm violation for GRANT on ROLE1
```

Securing roles

- However, user **c##ernesto** is authorized in realm as owner, so he can grant this role:

```
SQL> connect c##ernesto@pdb1
```

Enter password:

Connected.

```
SQL> grant role1 to usr1;
```

Grant succeeded.

Securing roles

- The **SYS** user can grant or revoke privileges from role even though the role is protected by realm.
- After we make the realm mandatory, this is no longer possible:

```
SQL> connect sys@pdb1 as sysdba
Enter password:
Connected.

SQL> revoke drop any synonym from role1;
revoke drop any synonym from role1
*
ERROR at line 1:
ORA-47410: Realm violation for REVOKE on ROLE1

SQL> grant update any table to role1;
grant update any table to role1
*
ERROR at line 1:
ORA-47410: Realm violation for GRANT on ROLE1
```

Preventing users from executing specific command on specific object

- You'll learn to create command rules.
- A command rule defines a protected database operation on a specific database object (for example, **UPDATE** on all tables in **HR** schema).
- The evaluation of associated rule set determines if statement will be allowed (executed) or blocked.

Preventing users from executing specific command on specific object

- In order to execute **UPDATE** on the table **ORDERS** in schema **OE**, rule set **Disabled** needs to evaluate **TRUE**.
- However, because rule set **DISABLED** will evaluate **FALSE** always, consequently, this command is disabled for all users in the database:

```
SQL> connect sys@pdb1 as sysdba
Enter password:
Connected.

SQL> UPDATE OE.ORDERS SET ORDER_MODE = 'TEST' WHERE ORDER_ID < 3000;
UPDATE OE.ORDERS SET ORDER_MODE = 'TEST' WHERE ORDER_ID < 3000
*
ERROR at line 1:
ORA-01031: insufficient privileges
```

Creating a rule set

- A rule set is a group of rules, which will be evaluated as a whole, using only AND or only OR operator.
- The Boolean result of logical evaluation is used in other Oracle Database Vault components to grant or deny certain actions (for example, deleting data from a table).
- You'll learn to create rules and rule sets.
- You are going to use Enterprise Manager Cloud Control 19c.

Creating a rule set

- To use rule set you have created in this, create command rule for **UPDATE** operation on schema **SCOTT**, table **EMP**.
- And choose that condition for evaluation whether update operation will be executed is defined by rule set **Working Hours**.

Creating a rule set

- Create command rule using your rule set



The screenshot shows a 'Create Command Rule' dialog box. At the top left is the title 'Create Command Rule'. Below it is a descriptive text: 'This page allows you to create or edit a command rule that can be associated with an existing Database Vault rule set.' On the right side of the header are three buttons: 'Show SQL', 'Cancel', and 'OK', with 'OK' being the one highlighted by a red box. The main area contains several input fields:

- * Command: UPDATE (with a magnifying glass icon to its right)
- Status: Enabled (radio button) Disabled (radio button)
- * Applicable Object Owner: SCOTT (with a magnifying glass icon to its right)
- * Applicable Object Name: EMP (with a magnifying glass icon to its right)
- * Rule Set: Working Hours (with a magnifying glass icon to its right, and this entire row is highlighted with a red box)

Creating a rule set

- Check time and day (your result will be different).
- In this case, it's **NOT** a work day, so rule set will evaluate to **false**.

```
SQL> !date
```

```
Sun Jun 14 02:12:02 CEST 2015
```

Creating a rule set

- Try to increase salary by **300** for the employee whose **empno** is **7902**

```
SQL> UPDATE SCOTT.EMP SET SAL = SAL + 300 WHERE EMPNO = 7902;
```

```
UPDATE SCOTT.EMP SET SAL = SAL + 300 WHERE EMPNO = 7902
```

```
*
```

```
ERROR at line 1:
```

```
ORA-01031: insufficient privileges
```

Creating a rule set

- Check time and day (your result will be different).
- In this case, it is a work day and it is during working hours, so rule set will evaluate to **true**

```
SQL>!date  
Mon Jun 15 14:27:24 CEST 2015
```

```
SQL> UPDATE SCOTT.EMP SET SAL = SAL + 300 WHERE EMPNO = 7902;  
1 row updated.
```

Creating a secure application role

- A secure application role is a database role whose enablement depends on the evaluation of a specified condition.
- You'll learn to create secure application role using Oracle Database Vault.
- The condition that determines whether the role will be enabled is specified by rule set (you can use built-in rule set or create your own).

Creating a secure application role

- You are going to test behavior of the secure application role.
- Connect to pluggable database pdb1 as a user who has the Oracle Database Vault Account Manager role and create the user **usr2**.

```
SQL> connect c##dbv_acctmgr/oraDVA123@pdb1
```

Connected.

```
SQL> create user usr2 identified by oracle1;
```

User created.

Creating a secure application role

- Connect to the pluggable database `pdb1` as a `SYS` user and grant a `create session` privilege to `usr2` and select and update privileges to the role `cust_role`

```
SQL> connect sys/oracle@pdb1 as sysdba
Connected.

SQL> grant create session to usr2;
Grant succeeded.

SQL> grant select on oe.customers to cust_role;
Grant succeeded.

SQL> grant update on oe.customers to cust_role;
Grant succeeded.
```

Creating a secure application role

- Connect to pluggable database **pdb1** as **usr2** and view information about machine you are accessing the database from.
- In this example, we are using a built-in factor to get that information.

```
SQL> connect usr2/oracle1@pdb1  
  
SQL> select dvf.f$machine from dual;  
  
F$MACHINE  
-----  
host01.challengeernesto.com
```

Creating a secure application role

- Set **cust_role** by using the PL/SQL package **DBMS_MACSEC_ROLES**:

```
SQL> EXEC DBMS_MACSEC_ROLES.SET_ROLE('CUST_ROLE');
```

PL/SQL procedure successfully completed.

Creating a secure application role

- View number of rows in the table **OE.CUSTOMERS**:

```
SQL> select count(*) from oe.customers;
```

COUNT(*)

319

- When the same user tries to connect from another machine, he or she won't be able to set the role, which in turn means that he or she won't be able to view data in the table **OE.CUSTOMERS**:

Creating a secure application role

```
SQL> connect usr2/oracle1@pdb1
Connected.
SQL> select dvf.f$machine from dual;
F$MACHINE
-----
host02.challengezoran.com
SQL> EXEC DBMS_MACSEC_ROLES.SET_ROLE('CUST_ROLE');
BEGIN DBMS_MACSEC_ROLES.SET_ROLE('CUST_ROLE'); END;
*
ERROR at line 1:
ORA-47305: Rule Set violation on SET ROLE (CUST_ROLE)
ORA-06512: at "DVSYS.DBMS_MACUTL", line 49
ORA-06512: at "DVSYS.DBMS_MACUTL", line 398
ORA-06512: at "DVSYS.DBMS_MACSEC", line 306
ORA-06512: at "DVSYS.ROLE_IS_ENABLED", line 4
ORA-06512: at "DVSYS.DBMS_MACSEC_ROLES", line 55
ORA-06512: at line 1

SQL> select count(*) from oe.customers;
select count(*) from oe.customers
*
ERROR at line 1:
ORA-00942: table or view does not exist
```

Using Database Vault to implement that administrators cannot view data

- You will use multiple components (realms, command rules, and rule sets) to secure data in database from administrators.

Using Database Vault to implement that administrators cannot view data

- We can show that the user `orders_dba` in fact can manage objects in `OE` schema (for instance, he can create and drop a table `test`)
- This is because he is authorized in realm that protects `oe` schema:

```
SQL> connect orders_dba@pdb1
```

Enter password:
Connected.

```
SQL> create table oe.test(a int);
```

Table created.

```
SQL> drop table oe.test;
```

Table dropped.

Using Database Vault to implement that administrators cannot view data

- However, the user **orders_dba** cannot view data that resides inside objects in **OE** schema
- Select on objects in this schema is restricted to users that have the role **ORD_USR_ROLE** using command rule:

```
SQL> select count(*) from oe.orders;
```

```
select count(*) from oe.orders  
*
```

ERROR at line 1:
ORA-01031: insufficient privileges

Using Database Vault to implement that administrators cannot view data

- The user **orders_user** has the role **ORD_USER_ROLE** and he or she can select data from table in OE schema:

```
SQL> connect orders_user@pdb1
```

Enter password:

Connected.

```
SQL> select count(*) from oe.orders;
```

COUNT(*)

105

Using Database Vault to implement that administrators cannot view data

- An example of adding a new user to the system and authorizing him to access the data:
- Because separation of duties is implemented, there are several users that need to grant certain privileges.

Using Database Vault to implement that administrators cannot view data

- Only account manager can create users in database:

```
SQL> connect c##dbv_acctmgr@pdb1
```

Enter password:
Connected.

```
SQL> create user orders_user2 identified by oracle3;
```

User created.

Using Database Vault to implement that administrators cannot view data

- The **SYS** user is one of the few users who are authorized to grant a **create session** privilege

```
SQL> connect sys@pdb1 as sysdba
```

```
Enter password:  
Connected.
```

```
SQL> grant create session to orders_user2;
```

```
Grant succeeded.
```

Using Database Vault to implement that administrators cannot view data

- Because c##ernesto is the only authorized user in realm that protects the role ORD_USR_ROLE
- He is the only user that can grant that role:

```
SQL> connect c##ernesto@pdb1
```

Enter password:

Connected.

```
SQL> grant ord_usr_role to orders_user2;
```

Grant succeeded.

Using Database Vault to implement that administrators cannot view data

- **Orders_dba** is the only user that is authorized in realm that protects **OE** schema, so he is the only user that can grant object privileges on objects in **OE** schema.

```
SQL> connect orders_dba@pdb1
```

Enter password:

Connected.

```
SQL> grant select on oe.orders to orders_user2;
```

Grant succeeded.

```
SQL> grant update on oe.orders to orders_user2;
```

Grant succeeded.

Using Database Vault to implement that administrators cannot view data

- After a user is granted all necessary privileges, he or she is able to connect to the database and select data from table in secured schema.

```
SQL> connect orders_user2@pdb1
```

Enter password:

Connected.

```
SQL> select count(*) from oe.orders;
```

COUNT(*)

```
-----
```

```
105
```

Running Oracle Database Vault reports

- You will intentionally violate some security controls in order to have data for reports.

Disabling Database Vault

- You will disable Database Vault in two ways: Using Enterprise Manager 19c Cloud Control and command line.

Re-enabling Database Vault

- You will enable previously disabled Database Vault in two ways: Using Enterprise Manager 19c Cloud Control and command line.



Oracle Database 19c Detective Controls

Database Vault and Data Pump

In this chapter, we will cover the following tasks:

- Exporting data using Oracle Data Pump in the Oracle Database Vault environment
- Creating factors in Oracle Database Vault
- Using TDE in a multitenant environment

- An Oracle Database Vault component **factor** is a named variable, which can have one or more values, assigned in several ways.
- The actual value of factor is named **identity**.
- Each factor has a **factor type**.

- A factor type is used only for classification purposes.
- Factors are building blocks for configuring security policies.
- They can be used in rules/rule sets.
- You can configure factors by using Oracle Enterprise Manager or the Database Vault API.

- In Oracle Database 19c, it is possible to perform Oracle Data Pump regular and transportable export and import operations in the Oracle Database Vault environment.
- You'll export data that resides in a schema that is protected by a realm.

- It is assumed that:

- You are using Oracle Database 12.1.0.2 (the traditional architecture) on Linux
- Sample schemas are installed (you'll use HR schema in this recipe)
- Database Vault is enabled and configured (a Database Vault owner is user dbv_owner, account manager is user dbv_acctmgr, and realm that protects HR schema is created).

- This is one way how you can create **HR** realm:

```
SQL> connect dbv_owner
SQL> BEGIN
  DBMS_MACADM.CREATE_REALM(
    realm_name  => 'HR Realm',
    description => 'Protects HR schema',
    enabled      => DBMS_MACUTL.G_YES,
    audit_options=> DBMS_MACUTL.G_REALM_AUDIT_OFF,
    realm_type   => 0);
END;
/
```

PL/SQL procedure successfully completed.

```
SQL> BEGIN
  DBMS_MACADM.ADD_OBJECT_TO_REALM(
    realm_name  => 'HR Realm',
    object_owner => 'HR',
    object_name  => '%',
    object_type  => '%');
END;
/
```

PL/SQL procedure successfully completed.

- A directory for export operations (for example, `dp_dir`) is created and a user (for example, `piter`) who is going to perform Data Pump export has read and write privileges on the directory.
- Also, the `DATAPUMP_EXP_FULL_DATABASE` role and the `CREATE TABLE` and `UNLIMITED TABLESPACE` privileges have been granted to the user:

```
SQL> connect system
SQL> CREATE DIRECTORY dp_dir AS '/u01/app/oracle/oradata/dp_exp';
SQL> connect dbv_acctmgr
SQL> create user piter identified by T2abc_4z1;
SQL> grant create session to piter;
SQL> connect / as sysdba
SQL> grant create table, unlimited tablespace to piter;
SQL> grant read, write ON DIRECTORY dp_dir to piter;
SQL> grant DATAPUMP_EXP_FULL_DATABASE to piter;
```

- To be able to export data that is protected by Database Vault mechanisms, user has to be authorized.
- You can authorize a user to perform export and import operations:
 1. On specific database object in a schema, such as table.

```
SQL> EXEC DBMS_MACADM.AUTHORIZE_DATAPUMP_USER ('AMY', 'HR',  
'EMPLOYEES');
```

2. On specific schema.

3. For entire database.

```
SQL> EXEC DBMS_MACADM.AUTHORIZE_DATAPUMP_USER ('KIM');
```

```
SQL> grant DV_OWNER to kim;
```

- If you want to encrypt Oracle Data Pump export, using its encryption features, Oracle Advanced Security option has to be enabled.
- To encrypt export, specify appropriate value for **ENCRYPTION** parameter.
- These are allowed values for the parameter:

ENCRYPTION=[ALL|DATA_ONLY|ENCRYPTED_COLUMNS_ONLY|MATERIALIZE_ONLY|NONE]

Creating factors in Oracle Database Vault

- You'll create three factors (**Day**, **Holiday**, and **NonWorkingDay**).
- The factor **Day** will return name of the day based on **sysdate**.

Creating factors in Oracle Database Vault

- It is assumed that:

- You are using Oracle Database 12.1.0.2 (the traditional architecture) on Linux and Oracle Enterprise Manager Cloud Control 19c
- Database Vault is enabled and configured (the Database Vault owner is the user `dbv_owner` and account manager is the user `dbv_acctmgr`).
- The user `dbv_owner` has been granted the `SELECT ANY DICTIONARY` privilege

Creating factors in Oracle Database Vault

- The user **piter** exists, the function **piter.get_function** has been created, and **DVSYS** has been granted the **EXECUTE** privilege on the function:

```
SQL> connect system
SQL> create or replace function piter.get_holiday
  2 return varchar2
  3 IS
  4 holiday varchar2(10);
  5 begin
  6 IF (RTRIM(TO_CHAR(SYSDATE,'DD-MON')) IN ('1-JAN', '4-JUL',
  '15-NOV')) THEN
  7 holiday := 'TRUE';
  8 ELSE
  9 holiday := 'FALSE';
 10 END IF;
 11 RETURN holiday;
 12 end;
 13 /
Function created.
SQL> grant execute on piter.get_holiday to dvsys;
Grant succeeded.
```

Creating factors in Oracle Database Vault

- The identity of a factor can be assigned by:
 - Method
 - Constant
 - Factors
- The process of assigning identity to a factor is named factor identification.

Creating factors in Oracle Database Vault

- You created two factors (**Day** and **Holiday**) whose identities were assigned by methods and one factor (**NonWorkingDay**) whose identity was assigned by factors.
- Factors can be evaluated.
- Because you created factors that can change during a session, you chose evaluation by access.

Creating factors in Oracle Database Vault

- Verify that the factor **NonWorkingDay** got the value **COMPANY_HOLIDAY**.
- In our case, **COMPANY_HOLIDAY** was matched, so **WEEKEND** wasn't evaluated.
- In general, it is better to try to avoid overlapping conditions (if possible) because maintenance is easier and the risk of making a mistake is smaller.

Creating factors in Oracle Database Vault

- From Factors page in EM19c verify that you can't delete the factor **Holiday** because you use it to resolve identities for the factor **NonWorkingDay**

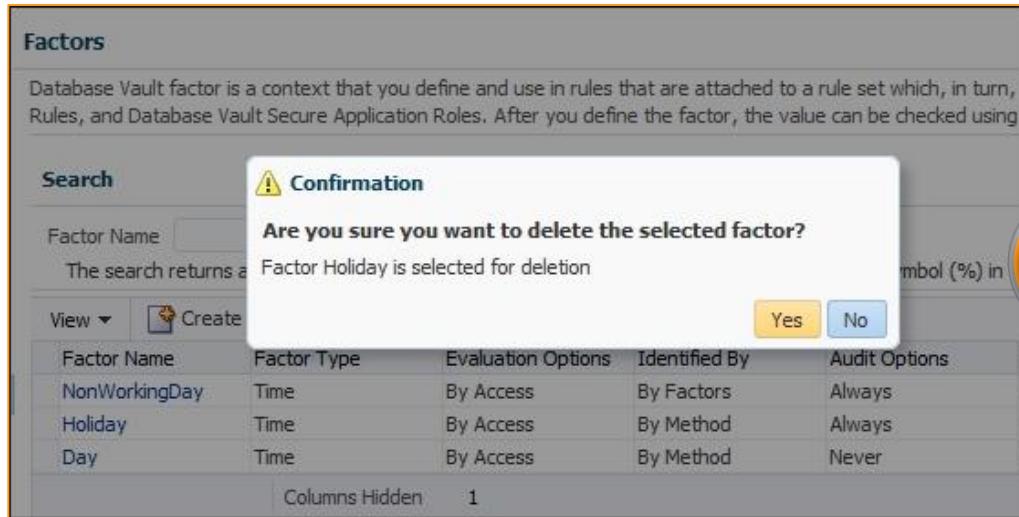


Factor Name	Factor Type	Evaluation Options	Identified By	Audit Options	Fail Options
NonWorkingDay	Time	By Access	By Factors	Always	Show Error Message
Holiday	Time	By Access	By Method	Always	Show Error Message
Day	Time	By Access	By Method	Never	Show Error Message

1

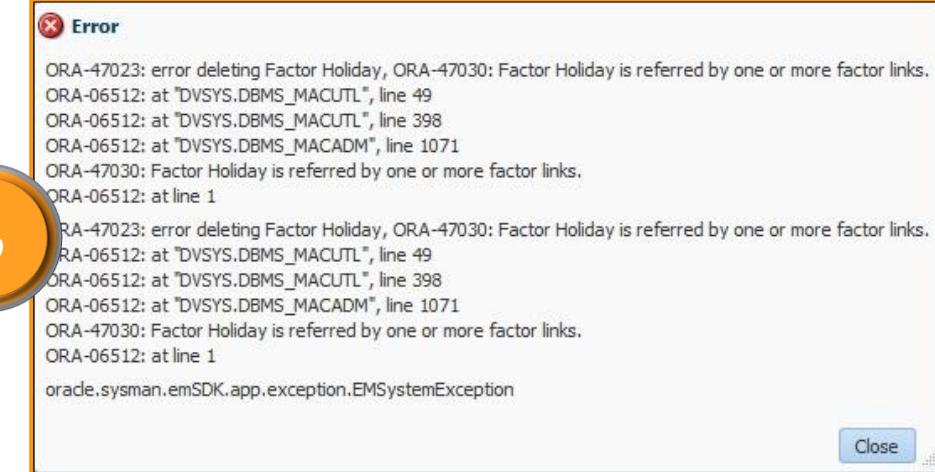
Select the factor **Holiday** and click on the **Delete** button.

Creating factors in Oracle Database Vault



2

Click on the button Yes



You will receive an error message

3

Using TDE in a multitenant environment

- You will perform different operations using Transparent Data Encryption in a multitenant environment.
- It is assumed that:
 - You have two container databases (the multitenant architecture), version 12.1.0.2 in the same host.
 - You have at least one pluggable database in each container database
 - You have sample schemes installed.

Using TDE in a multitenant environment

- Creation of keystore and master key in root container.
- Opening and creation of master key in the pluggable database.
- Another way of creation of the keystore and master key (in the second container database).

Using TDE in a multitenant environment

- User with the **SYSKM** system privilege is used, and opening of keystore as well as the creation of master keys are done by using **container=all** clause.
- Because there is only one keystore per container database but multiple master keys, if database needs to be unplugged and plugged into another container database, a master key needs to be exported and imported into the target database also.

Using TDE in a multitenant environment

- Export a master key and unplug the database.
- Plugging this database into another container database (cdb2).
- We are importing a master key.
- After importing a master key, restart that pluggable database.



Oracle Database 19c Detective Controls

Application Contexts

In this chapter, we will cover the following tasks:

- Exploring and using built-in contexts
- Creating an application context
- Setting application context attributes
- Using an application context

- An **application context** is a memory container that holds a set of key-value pairs.
- Also, an application context is a namespace because in different application contexts, attributes that have the same name can exist.

The steps to implement a local application context

Create an application context

Create a PL/SQL package that sets the context

Set application context attributes by calling the package

Use application context attributes in your application

Exploring and using built-in contexts

- The **USERENV** application context is a built-in context that contains information about the current session.
- You'll learn to retrieve values from built-in contexts.
- You'll need an existing user who can get values from built-in namespaces by using the **SYS_CONTEXT** function (for example, user **maja**).

Exploring and using built-in contexts

- You use the `SYS_CONTEXT` function to get values of several parameters from the `USERENV` context.
- You can use that function in both SQL and PL/SQL statements.

Exploring and using built-in contexts

- The **UNIFIED_AUDIT_SESSIONID** attribute (parameter) is introduced in Oracle Database 12.1.0.2.
- The value of that parameter is **unified audit session ID** if the database uses unified auditing mode or mixed auditing mode, and **NULL** if the database uses traditional auditing.

The value of the UNIFIED_AUDIT_SESSIONID

ATTRIBUTE
UNIFIED_AUDIT_SESSIONID



VALUE

**unified audit sesión ID,
unified or mixed auditing
mode**

**NULL,
traditional auditing mode**

Exploring and using built-in contexts

- Another built-in context is **SYS_SESSION_ROLES**.
- You can use it to check whether a specified role is currently enabled for the session.
- Afterwards, you'll verify that **zoran** has the **test_role** role by using the **SYS_CONTEXT** function.
- When working in the multitenant environment, some useful attributes are **CON_ID**, **CON_NAME**, and **CDB_NAME**.

Exploring and using built-in contexts

```
SQL> create role test_role;
Role created.

SQL> grant select on hr.employees to test_role;
Grant succeeded.

SQL> grant test_role to ernesto
Grant succeeded.

SQL> select sys_context('SYS_SESSION_ROLES', 'TEST_ROLE') from dual;
SYS_CONTEXT('SYS_SESSION_ROLES', 'TEST_ROLE')
-----
FALSE

SQL> connect ernesto
Enter password:
Connected.
SQL> select sys_context('SYS_SESSION_ROLES', 'TEST_ROLE') from dual;
SYS_CONTEXT('SYS_SESSION_ROLES', 'TEST_ROLE')
-----
TRUE
```

Creating an application context

- In this recipe, you'll create a local application context (for example, `sh_client`).
- In the next techniques, you will use it to store clients' identifiers.
- You'll need an existing user who can create an application context (it needs the `CREATE ANY CONTEXT` privilege or a DBA role), for example, the user `maja`.

Creating an application context

- Create application context `sh_client` and defined that the PL/SQL package `sh_ctx_pkg` will be used to create and set application context attributes.
- At this moment, attributes aren't set in the application context.

Setting application context attributes

- You'll create the PL/SQL package (for example, `sh_ctx_pkg`) that will set application context attributes for the application context you created in the previous technique (for example, `sh_client`).
- Also, you'll create a logon trigger.

Setting application context attributes

- You'll need an existing user who can create `sh_ctx_pkg`.
- Make sure that the user has direct privileges on the `sh.customers` table (even if he/she has a DBA role).

Setting application context attributes

- Create a logon trigger so that every user who connects to the database is going to have an application context set.
- It is very important to note that if you try to set or change key-value pairs outside the package you specified when you created application context, you will receive the error **insufficient privileges**.

```
SQL> exec DBMS_SESSION.SET_CONTEXT('sh_client','cust_id',101);
BEGIN DBMS_SESSION.SET_CONTEXT('sh_client','cust_id',101); END;
*
ERROR at line 1:
ORA-01031: insufficient privileges
ORA-06512: at "SYS.DBMS_SESSION", line 122
ORA-06512: at line 1
```

Using an application context

- In this, you'll see one possible usage (in SQL) of the application contexts.
- Create a new user (for example, **sofia**).
- Make sure that his or her e-mail in the format **user@company.example.com** is unique.
- Grant him or her privileges: **create session** and **select** on **sh.customers** table.

Using an application context

- New user

```
SQL> create user sofia identified by Q14be7NP;  
User created.  
  
SQL> grant create session to sofia;  
Grant succeeded.  
  
SQL> grant select on sh.customers to sofia;  
Grant succeeded.
```

Using an application context

- Insert data about him or her into the sh.customers table.

```
SQL> insert into sh.customers values (80000,'Sofia','Smith','F',1979,'Married','Albert Embankment 19','SE1 7HD','London',11111,'England','1111',52790,'1111111','12',30,'Sofia@company.example.com',10000,1,1,sysdate,sysdate,'T');

1 row created.

SQL> commit;
```



Oracle Database 19c Detective Controls

Oracle Compliance Framework

Oracle Compliance Framework

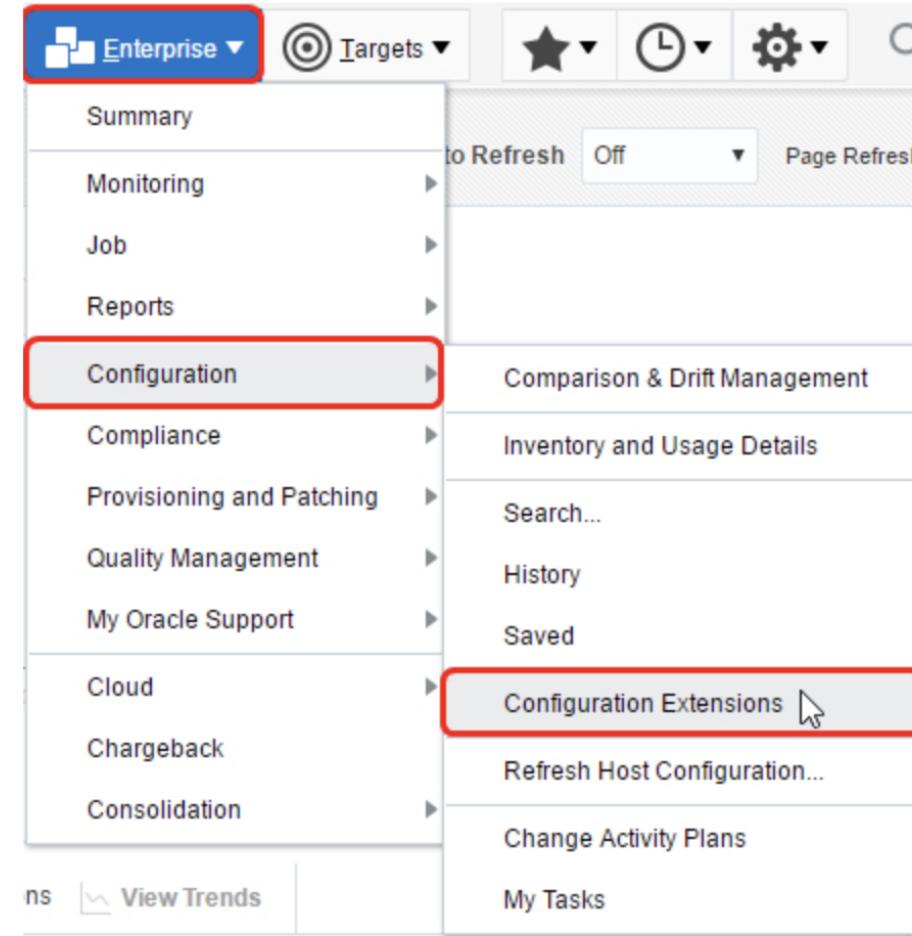
- Corporate requirements
- Government requirements
- Built in security rules
- Out-of-the box solutions

- Built in Compliance Rules
 - Government STIGS
 - Software Technical Implementation Guides
 - Deploy code and security against the STIG
 - Pre-defined compliance score

Oracle Compliance Framework

- Part of Oracle's software lifecycle management
- Installed via OEM 19c or 13c Cloud Control
- Databases will be compared against the STIG

Oracle Compliance Framework



Oracle Compliance Framework

The screenshot shows the Oracle Enterprise Manager Cloud Control 13c interface. On the left, there's a sidebar with various navigation options. A large orange arrow points from the top right towards this sidebar.

Configuration Extensions

Overview

Actions ▾ (highlighted with a red box)

- Create
- Create Like
- Edit
- View Details
- Delete
- Manage Deployments
- Manage Parsers
- Enable Facet Synchronization
- Export
- Import** (highlighted with a red box and a cursor icon)
- Create Custom Target Type
- Add New Custom Target

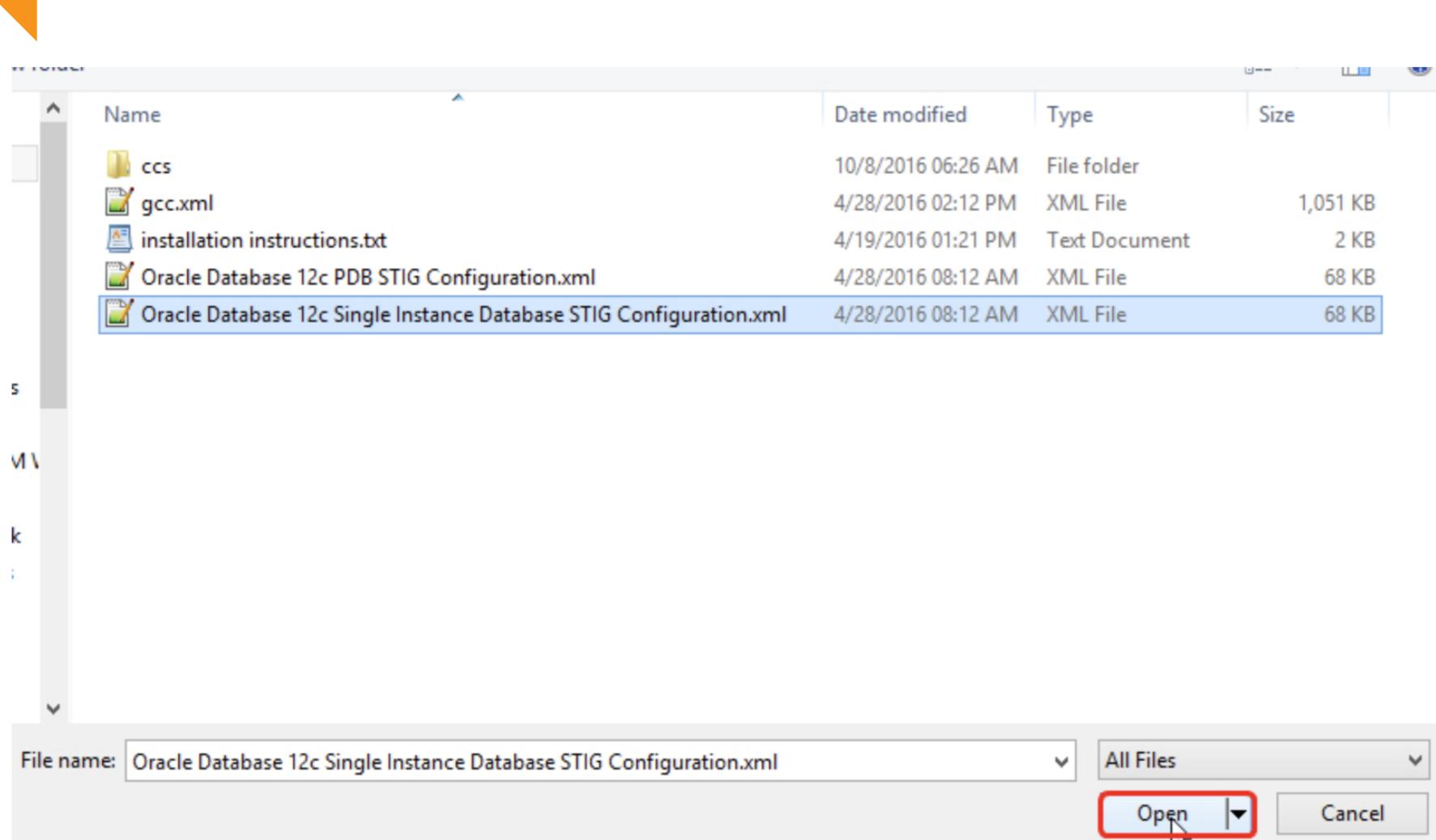
Import dialog box (highlighted with a red box)

* File **Choose file** (highlighted with a red box)

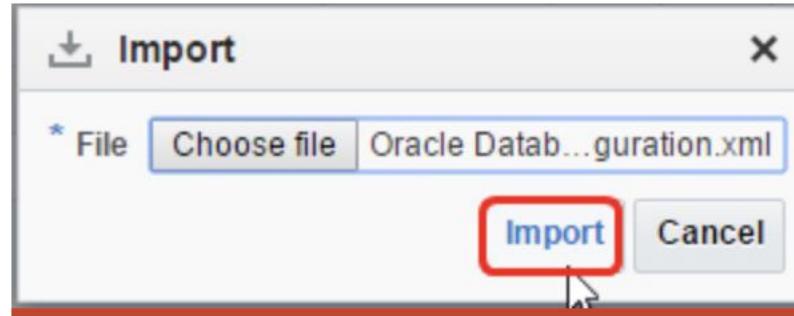
No file chosen

Import Cancel

Oracle Compliance Framework



Oracle Compliance Framework



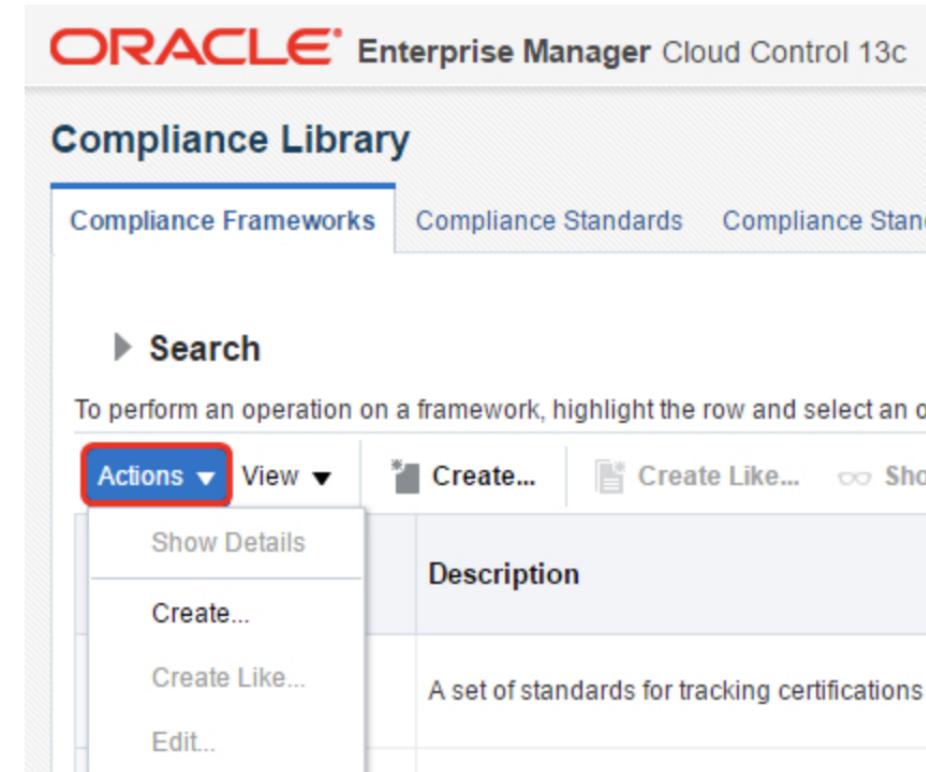
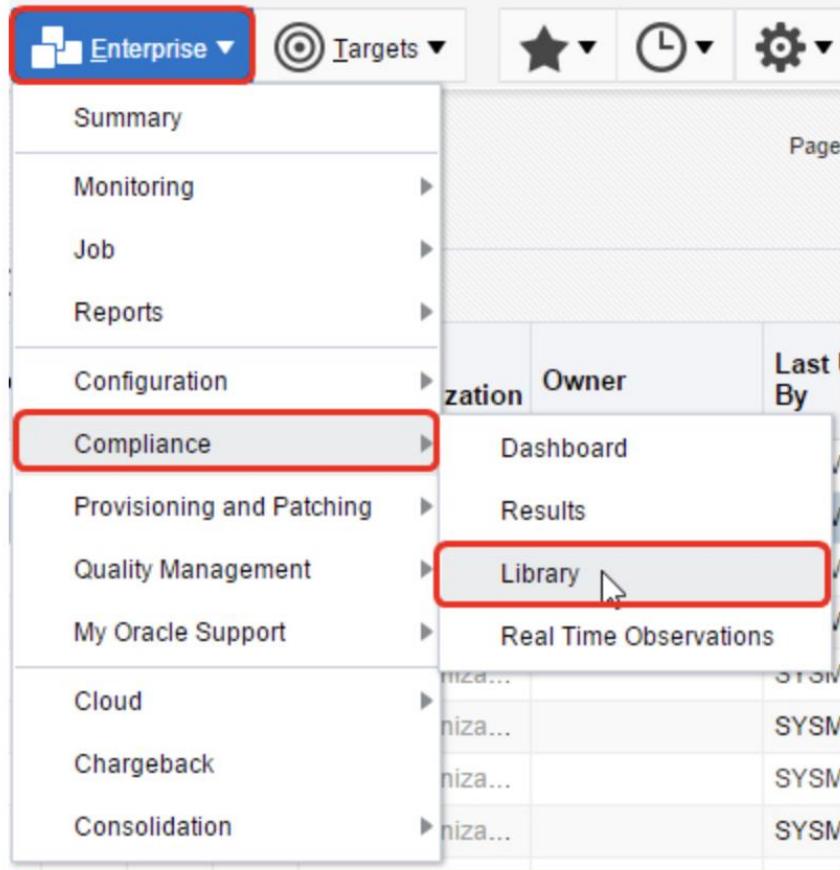
ORACLE® Enterprise Manager Cloud Control 13c

Configuration Extensions

Overview

Name	Description	Target Type	Version	Deployments	Facet Synchronization	Owner	Last Updated By	Last Updated On
IDM Config Tool		Oracle Access ...	1	✓	No synchroniza...	SYSMAN	SYSMAN	Oct 6, 2016 12:06:37 PM
STIG Configuration	Configuration Extension for STIG rules	Database Insta...	1	✓	No synchroniza...	SYSMAN	SYSMAN	Oct 6, 2016 12:09:22 PM
STIG Configuration	Configuration Extension for STIG rules	Cluster Database	1	✓	No synchroniza...	SYSMAN	SYSMAN	Oct 6, 2016 12:09:22 PM
Oracle Database 11gR2 ...	Configuration Extension for Oracle 11.2g Database STIG	Database Insta...	1	✓	No synchroniza...	SYSMAN	SYSMAN	Oct 6, 2016 12:09:30 PM
Oracle Cluster Database ...	Configuration Extension for Oracle 11.2g Database STIG	Cluster Database	1	✓	No synchroniza...	SYSMAN	SYSMAN	Oct 6, 2016 12:09:30 PM
Orachk Collections		Host	1	✓	No synchroniza...	SYSMAN	SYSMAN	Oct 6, 2016 12:14:19 PM
Orachk Collections		Analytics System	1	✓	No synchroniza...	SYSMAN	SYSMAN	Oct 6, 2016 12:14:20 PM
Orachk Collections		Oracle Exalogic	1	✓	No synchroniza...	SYSMAN	SYSMAN	Oct 6, 2016 12:14:20 PM
Orachk Collections		Cluster	1	✓	No synchroniza...	SYSMAN	SYSMAN	Oct 6, 2016 12:14:20 PM
Oracle Database 12c Sin...	Configuration Extension for Oracle 12c Single Instance Dat...	Database Insta...	1	✓	No synchroniza...	SYSMAN	SYSMAN	Oct 7, 2016 12:44:16 PM
Oracle Database 12c PD...	Configuration Extension for Oracle 12c PDB STIG	Pluggable Data...	1	✓	No synchroniza...	SYSMAN	SYSMAN	Oct 7, 2016 12:44:57 PM

Oracle Compliance Framework



Oracle Compliance Framework

The screenshot shows the Oracle Enterprise Manager Cloud Control 13c interface. The title bar reads "ORACLE® Enterprise Manager Cloud Control 13c". Below it, the "Compliance Library" is displayed. A navigation bar at the top includes tabs for "Compliance Frameworks" (which is selected), "Compliance Standards", and "Compliance Standard". A search bar with the placeholder "Search" is present. Below the search bar, a message says "To perform an operation on a framework, highlight the row and select an operation from the Actions menu." An "Actions" dropdown menu is open, showing options like "View", "Create...", "Create Like...", "Show Details", "Import...", "Edit...", "Delete", "Export...", and "Recovery". The "Import..." option is highlighted with a red box. To the right of the dropdown, there are two rows of framework details. The first row's description is "A set of standards for tracking certifications" and its note is "This framework consolidates all the Exachk". The second row's description is "This framework consolidates all the Exachk" and its note is also "This framework consolidates all the Exachk".

Oracle Compliance Framework

Name	Date modified	Type	Size
ccs	10/8/2016 06:26 AM	File folder	
gcc.xml	4/28/2016 02:12 PM	XML File	1,051 KB
installation instructions.txt	4/19/2016 01:21 PM	Text Document	2 KB
Oracle Database 12c PDB STIG Configuration.xml	4/28/2016 08:12 AM	XML File	68 KB
Oracle Database 12c Single Instance Database STIG Configuration.xml	4/28/2016 08:12 AM	XML File	68 KB

Oracle Compliance Framework

The screenshot shows the Oracle Enterprise Manager Cloud Control 13c interface. At the top, there is a modal dialog titled "Import Compliance Framework" with the sub-instruction "Select the file for import." It contains a file input field set to "gcc.xml" and an "Overwrite Option" checkbox labeled "Import with overwrite". The "OK" button is highlighted with a red box. Below the dialog, the main page header reads "ORACLE Enterprise Manager Cloud Control 13c". The top navigation bar includes links for "Enterprise", "Targets", "SYSMAN", and other system icons. The main content area is titled "Compliance Library" and features a table with columns: "Compliance Frameworks", "Compliance Standards", "Compliance Standard Rules", and "Real-time Monitoring Facets". A "Search" section is present with a "Create..." button. A confirmation dialog box is overlaid on the page, titled "Confirmation", stating "Compliance content imported successfully." with an "OK" button. The table data includes a row for "Oracle 12c Database STIG" with a detailed description: "A set of standards to ensure compliance of Oracle Single Instance Databases and Oracle 12c Pluggable Databases." The row also lists "Production" as the state, "SYSMAN" as the author, "Security, 12c" as keywords, and the date "Oct 7, 2016 1:05:15 PM GMT-07:00".

Oracle Compliance Framework

- Generate scripts and sql to set and create STIG rules
- Internally monitor tables and activity to meet the requirements
- Grant privileges to the user STIGTOOL

- Setting STIG RULES

```
begin
stig_custom_report.set_stig_rules(1);
stig_custom_report.set_stig_rules(2, 'pdb');
end;
/
exit;
```

- Create STIG Package

```
-- DDL for Package Body STIG_CUSTOM_REPORT

CREATE OR REPLACE PACKAGE BODY "STIGTOOL"."STIG_CUSTOM_REPORT" AS
/* STIG_CUSTOM_REPORT v1.2 - Initial Release
   v1.3 - Changed schema to STIGTOOL
*/
procedure update_vuln(p_stig_key NUMBER, p_target VARCHAR2, p_vuln_rule_id VARCHAR2, p_status VARCHAR2,
p_comment VARCHAR2, p_finding VARCHAR2) AS
d DATE;
BEGIN
  d := sysdate;
  INSERT INTO STIG_RULE_STATUS (stig_key, rule_id, target, status, findings, comments, date_updated)
  VALUES (p_stig_key, p_vuln_rule_id, p_target, p_status, p_finding, p_comment, d);
END update_vuln;

PROCEDURE get_vul_detail(p_rule_id IN VARCHAR2, p_target IN VARCHAR2, p_status OUT VARCHAR2, p_finding OUT
VARCHAR2, p_comments OUT VARCHAR2) IS
/*
For the vul_id, pull the status, finding, and comments from OEM views
*/
vcount INTEGER;
rcount INTEGER;
rfound BOOLEAN;
v_rule_type INTEGER;
valid_values VARCHAR2(4000);
v_suppressed BOOLEAN;
```

- Grant STIGTOOL Prvs

```
ALTER USER STIGTOOL QUOTA UNLIMITED ON USERS;
GRANT CREATE TABLE TO STIGTOOL;
GRANT CREATE PROCEDURE TO STIGTOOL;
grant select on sysman.mgmt$compliance_standard_rule to stigtool;
grant select on sysman.mgmt$events to stigtool;
grant select on sysman.mgmt$target to stigtool;
grant select on sysman.mgmt$event_annotation to stigtool;
grant select on sysman.mgmt$cs_target_assoc to stigtool;
exit;
```

- Generate STIG Report

```
CREATE TABLE "STIGTOOL"."STIG_DOC_TO_TARGET"
(   "RULE_ID" VARCHAR2(20 BYTE),
    "STIG_KEY" NUMBER(4,0),
    "DOC_ID" NUMBER(5,0),
    "TARGET" VARCHAR2(50 BYTE),
    "DATE_VALID_UNTIL" DATE,
    "SUPPRESS" VARCHAR2(20 BYTE)
) SEGMENT CREATION IMMEDIATE
PCTFREE 10 PCTUSED 40 INITTRANS 1 MAXTRANS 255
NOCOMPRESS LOGGING
STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645
PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1
BUFFER_POOL DEFAULT FLASH_CACHE DEFAULT CELL_FLASH_CACHE DEFAULT)
TABLESPACE "USERS" ;
```

- Generate STIG Report

STIG_RULE_STATUS Example Rows

RULE_ID	STATUS	STIG_KEY	FINDINGS	COMMENTS
SV-75913r1_rule	Open		1 EXECUTE privilege granted to PUBLIC on... Violation Count is 9.	
SV-75919r1_rule	Open		1sql92_security parameter is set to FALSE. Violation Count is 1.	
SV-76019r1_rule	Open		1global_names parameter is set to FALSE.; Violation Count is 1.	
SV-76025r1_rule	Open		1SQLNET.ALLOWED_LOGON_VERSION not set i... Violation Count is 1.	
SV-76031r1_rule	Open		1User XDB requires non-default password... Violation Count is 30.	
SV-76035r1_rule	Open		1No network encryption is used.; Violation Count is 1.	
SV-76053r2_rule	Open		1User XSSNULL is assigned profile DEFAU... Violation Count is 36.	
SV-76093r2_rule	Open	1		Rule not found in Enterprise Manager.

RULE_ID	STATUS	STIG_KEY	FINDINGS	COMMENTS
SV-75899r1_pdbrule	Open		2 Database is not compliant to rule SV-7... Manual Rule.	
SV-75899r1_rule	Open		1Database is not compliant to rule SV-7... Manual Rule.	

RULE_ID	STATUS	STIG_KEY	FINDINGS	COMMENTS
SV-75927r1_rule	Not a ...	1		Additional Documentation Example: Each value is required .

RULE_ID	STATUS	STIG_KEY	FINDINGS	COMMENTS
SV-75909r1_rule	Not a Finding		1 Automated Violation Supressed... Violation Count is 3. REDO Log Risk Acceptable:	

RULE_ID	STATUS	STIG_KEY	FINDINGS	COMMENTS
SV-76271r1_pdbrule	Open		2 Database is not compliant to ... Manual Rule.	
SV-76271r1_rule	Open		1Database is not compliant to ... Manual Rule.	
SV-76273r1_pdbrule	Open		2Database is not compliant to ... Manual Rule.	
SV-76273r1_rule	Not a finding	1		Internal Supercluster Rate Limit: For RAC setup,



Oracle Database 19c Detective Controls

Compliance Framework Rules

Compliance Framework rules

- Understanding the rules
- Implementing the rules
- Provides advice on the rules
- Real time compliance monitoring

Compliance Framework rules

- Compliance features
 - Provides the ability to evaluate the compliance of targets and systems as they relate to business best practices for configuration, security and storage.
 - Database configuration data can now be managed within the new configuration and compliance standards frameworks.

Compliance Framework rules

- Compliance Standards
 - A compliance standard evaluation can provide information related to platform compatibility, known issues affecting other customers with similar configurations, security vulnerabilities, patch recommendations, and more
 - A compliance standard is Enterprise Manager's representation of a compliance control that must be tested against a set of IT infrastructure to determine if the control is being followed

Compliance Framework rules

- Compliance Rules
 - Repository of Rules
 - Real time monitoring of rules
 - Agent Rules
 - Manual Rules
 - Weblogic Rules

Compliance Framework rules

- Compliance Reasons
 - Database Change
 - Configuration Best Practices for Oracle RAC Database
 - Configuration Best Practices for Oracle Database
 - Configuration Best Practices for Oracle Pluggable Database

Compliance Framework rules

- Compliance Evaluation
 - Via Oracle's Enterprise Manager Cloud Control (13c)
 - Imported using the software library
 - Compliance reports may be generated

Compliance Framework rules

- Creating Rules
 - Utilize existing compliance standards
 - Create rules based on existing rules (STIGS)
 - Rules can be edited, added, or removed
 - Alerts may be established to track rules.

Compliance Framework rules

- Terms
 - Compliance Framework
 - A compliance framework is an organized list of control areas that need to be followed for a company to stay in compliance in their industry
 - A single framework control area maps to one or more compliance standards

Compliance Framework rules

- Terms
 - Compliance Standard
 - A compliance standard is a collection of checks or rules that follow broadly accepted best practices
 - This ensures that IT infrastructure, applications, business services and processes are organized, configured, managed, and monitored properly. A compliance standard evaluation can provide information related to platform compatibility, known issues affecting other customers with similar configurations, security vulnerabilities, patch recommendations, and more
 - A compliance standard is also used to define where to perform real-time change monitoring

Compliance Framework rules

- Terms
 - Compliance Standard Rule
 - A compliance standard rule is a specific test to determine if a configuration data change affects compliance
 - Types of rules
 - Repository
 - Used to perform a check against any metric collection data in the Management Repository

Compliance Framework rules

- Terms
 - Types of rules
 - WebLogic Server Signature Rule
 - Used to check a WebLogic target to support best practice configurations.
 - Real-time Monitoring Rule
 - Used to monitor actions to files, processes, and database entities in real-time as the changes occur
 - Agent-Side Rule
 - Used to perform configuration checks on the agent and upload violations into the Management Repository.

Compliance Framework rules

- Terms
 - Types of rules
 - Manual Rule
 - Checks that must be performed but cannot be automated
 - Compliance Standard Rule Folder
 - Compliance standard rule folders are hierarchical structures that contain compliance standard rules

Compliance Framework rules

- Terms
- Importance
 - Importance is a setting that the user can make when mapping compliance frameworks, standards, and rules
- Score
 - A target's compliance score for a compliance standard is used to reflect the degree of the target's conformance with respect to the compliance standard. The compliance score is in the range of 0% to 100% inclusive. A compliance score of 100% indicates that a target fully complies with the compliance standard

Compliance Framework rules

- Terms
- Real-time Facets
 - The real-time monitoring rule definition includes facets that specify what is important to monitor for a given target type, target properties, and entity type
- Real-Time Observations
 - Observations are the actions that were seen on a host or target that were configured to be monitored through real-time monitoring rules. Each distinct user action results in one observation

Compliance Framework rules

- Terms
- Observation Audit Status
 - Every observation has an audit status that determines if the observation was authorized, or unauthorized, or neither (unaudited)
- Observation Bundles
 - Single observations are not reported from the Management Agent to the server. They are instead bundled with other observations against the same target, rule, and user performing the action