

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**  
**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**  
**KHOA CÔNG NGHỆ THÔNG TIN**



**CHUYÊN ĐỀ**  
**HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU**

**Seminar**  
**Class 21HTTT**  
**Group 6**

21127373 - Lê Thanh Khôi Nguyên  
21127591 - Nguyễn Hiền Đạt  
21127157 - Dương Phước Sang  
21127085 - Phan Trung Kiên

**Supervisors:**

Tuấn Nguyễn Hoài Đức  
Hồ Chí Minh - 3/2024

# CONTENT

<b>I</b>	<b>Virtual Private Database (VPD)</b>	<b>2</b>
1	VPD concepts . . . . .	2
1.1	Components of an Oracle Virtual Private Database Policy . . . . .	3
2	Column-level VPD and Column Masking . . . . .	4
	Column-masking Behavior . . . . .	7
3	VPD Security Policy . . . . .	10
3.1	Apply Policies on table, view, synonym . . . . .	10
3.2	Organize Policies as Groups . . . . .	11
3.3	Default Policies . . . . .	16
4	Configure VPD for Fine-grained Access Control . . . . .	17
4.1	Use the DBMS_RLS Package and the Create Context command . . .	17
4.2	Use the Oracle Policy Manager Interface . . . . .	24
5	Limit the scope of VPD use (in which Schemas) . . . . .	27
<b>II</b>	<b>Application Context</b>	<b>29</b>
1	Application Context features and setup . . . . .	29
2	Session-based / Local / Global Application Context . . . . .	32
2.1	Session-based application contexts . . . . .	32
2.2	Local Application context: . . . . .	35
2.3	Global Application Context: . . . . .	40
2.4	Client Session-Based Application Context . . . . .	45
3	How to use Application Context for Fine-grained Access Control . . . . .	46
3.1	Thực hiện chính sách: . . . . .	46
4	Check the policy effect: View V \$ VPD_POLICY . . . . .	50
<b>III</b>	<b>Tài liệu tham khảo</b>	<b>51</b>

MSSV	Họ và Tên	Nội dung phân công	Hoàn thành
21127373	Lê Thanh Khôi Nguyên	VPD Security Policy & Use the Oracle Policy Manager Interface	100%
21127591	Nguyễn Hiến Đạt	VPD concept & Column-level VPD and Column Masking & Use the DBMS_RLS Package and the Create Context command	100%
21127157	Dương Phước Sang	Session-based/Local/Global Application Context & Check the policy effect: View V\$VPD_POLICY	100%
21127085	Phan Trung Kiên	Limit the scope of VPD use (in which Schemas) & Application Context features and setup & How to use Application Context for Fine-grained Access Control	100%

**Table .1** Bảng phân công

# I. Virtual Private Database (VPD)

## 1. VPD concepts

Oracle Virtual Private Database tạo ra các chính sách bảo mật để kiểm soát quyền truy cập cơ sở dữ liệu ở cấp hàng và cột.

Về cơ bản, Oracle Virtual Private Database thêm một điều kiện động WHERE vào một câu lệnh SQL được thực thi đối với bảng (table), chế độ xem (view), hoặc từ đồng nghĩa (synonym) mà một chính sách bảo mật Oracle Virtual Private Database đã được áp dụng.

Oracle Virtual Private Database cho phép bảo mật chi tiết trên bảng (table), chế độ xem (view) hoặc từ đồng nghĩa (synonym) của cơ sở dữ liệu. Bạn có thể đính kèm chính sách bảo mật vào các đối tượng này và chúng tự động áp dụng khi truy cập dữ liệu, đảm bảo bảo mật không bị vượt qua.

Khi bạn truy cập vào một bảng (table), chế độ xem (view) hoặc từ đồng nghĩa (synonym) được bảo vệ bởi chính sách Oracle Virtual Private Database, hệ thống Oracle Database sẽ tự chỉnh sửa câu lệnh SQL của bạn. Việc sửa đổi này thêm một điều kiện WHERE (được gọi là điều kiện tiên quyết) từ một hàm thực hiện chính sách bảo mật. Oracle Database thực hiện việc sửa đổi này một cách tự động và không ảnh hưởng đến người dùng, sử dụng bất kỳ điều kiện nào từ hàm. Bạn có thể sử dụng chính sách Oracle Virtual Private Database với các câu lệnh SELECT, INSERT, UPDATE, INDEX và DELETE.

Ví dụ khi người dùng sử dụng câu lệnh truy xuất dữ liệu từ bảng:

```
SELECT * FROM OE.ORDERS;
```

Chính sách Oracle Virtual Private Database tự động thêm vào câu lệnh một mệnh đề WHERE. Ví dụ:

```
SELECT * FROM OE.ORDERS  
WHERE SALES_REP_ID = 159;
```

Trong ví dụ này, người dùng chỉ có thể xem các đơn đặt hàng do Đại diện Bán hàng 159 thực hiện.

Nếu bạn muốn lọc người dùng dựa trên thông tin phiên của họ, như ID của người dùng đó, thì bạn có thể tạo mệnh đề WHERE để sử dụng ngữ cảnh ứng dụng (Application context). Ví dụ:

```
SELECT * FROM OE.ORDERS  
WHERE SALES_REP_ID = SYS_CONTEXT('USERENV','SESSION_USER');
```

### 1.1. *Components of an Oracle Virtual Private Database Policy*

Một chính sách VPD sử dụng một hàm để tạo ra mệnh đề động WHERE, và một chính sách để gắn hàm vào các đối tượng để bảo vệ.

#### **Hàm để Tạo Ra Mệnh Đề Động WHERE**

Hàm Oracle Virtual Private Database (VPD) xác định các hạn chế mà bạn muốn áp dụng.

Để tạo ra mệnh đề WHERE động của Oracle Virtual Private Database (VPD), bạn phải tạo ra một hàm (không phải là một thủ tục) xác định những hạn chế này. Hàm này là một hàm quyền của người xác định.

Thường thì, người quản trị bảo mật tạo ra hàm này trong schema của họ. Đối với hành vi phức tạp hơn, như bao gồm các lời gọi tới các hàm khác hoặc thêm kiểm tra để theo dõi các lần đăng nhập không thành công, tạo ra các hàm này trong một package.

Hàm phải có các hành vi sau:

- Chúng ta cần nhận vào tên schema và tên đối tượng (bảng, view, hoặc synonym). Phải xác định các tham số đầu vào để lưu thông tin này, nhưng không ghi rõ tên schema và đối tượng trong hàm. Chính sách bạn tạo từ gói DBMS\_RLS sẽ đưa ra tên của schema và đối tượng mà chính sách sẽ dùng. Bạn cần tạo tham số cho schema trước, sau đó mới là tham số cho đối tượng.
- Phải cung cấp một giá trị trả về cho mệnh đề WHERE sẽ được tạo ra. Giá trị trả về cho mệnh đề WHERE luôn là kiểu dữ liệu VARCHAR2.
- Phải tạo ra một mệnh đề WHERE hợp lệ. Trong đó mệnh đề WHERE của nó giống nhau cho tất cả người dùng đăng nhập.
- Tuy nhiên, trong hầu hết các trường hợp, bạn có thể muốn tạo ra câu lệnh WHERE khác nhau cho từng người dùng, nhóm người dùng, hoặc ứng dụng. Ví dụ, nếu một người quản lý đăng nhập, câu lệnh WHERE có thể được tạo ra dựa trên quyền của người quản lý đó. Bạn có thể làm điều này bằng cách sử dụng một ngữ cảnh ứng dụng (Application context), nơi lưu trữ thông tin phiên của người dùng, trong việc tạo ra câu lệnh WHERE.
- Bạn có thể tạo hàm Oracle Virtual Private Database mà không cần sử dụng bối cảnh ứng dụng. Nhưng nếu sử dụng, chính sách Oracle Virtual Private Database sẽ mạnh mẽ hơn.

Bối cảnh ứng dụng giúp đặt quyền truy cập an toàn cho người dùng dựa trên thông tin phiên làm việc, như ID người dùng.

- Ngoài ra, bạn có thể nhúng các lời gọi C hoặc Java để truy cập thông tin hệ điều hành hoặc để trả về các mệnh đề WHERE từ một tệp hệ điều hành hoặc nguồn khác.
- Bạn không thể chọn một bảng từ trong hàm của chính sách liên quan. Bạn có thể định nghĩa một chính sách cho một bảng, nhưng bạn không thể chọn bảng đó từ chính sách của bảng đó.
- Phải là hàm đơn giản. Hàm VPD chỉ cần dựa vào ngữ cảnh ứng dụng và các tham số được đưa vào để tạo ra mệnh đề WHERE. Hàm này không nên dựa vào các biến của gói.

### **Chính Sách để Gắn Hàm vào Các Đối Tượng Bạn Muốn Bảo Vệ**

Chính sách Oracle Virtual Private Database liên kết hàm VPD với một bảng, view, hoặc synonym.

Bạn sử dụng gói DBMS\_RLS để tạo chính sách. Nếu không phải là SYS, bạn cần được cấp quyền EXECUTE để sử dụng gói này. Gói DBMS\_RLS bao gồm các thủ tục để quản lý chính sách và thiết lập kiểm soát truy cập chi tiết. Ví dụ, bạn có thể sử dụng thủ tục DBMS\_RLS.ADD\_POLICY để áp dụng chính sách cho một bảng. Trong đó, bạn có thể thiết lập kiểm soát truy cập chi tiết, như khi nào chính sách có hiệu lực khi một người dùng thực hiện lệnh SELECT hoặc UPDATE trên bảng hoặc view.

Sự kết hợp giữa việc tạo hàm và sau đó áp dụng nó vào một bảng hoặc view được gọi là việc tạo chính sách Oracle Virtual Private Database.

## **2. Column-level VPD and Column Masking**

VPD cho phép bạn bảo vệ dữ liệu theo hàng khi một cột quan trọng được truy vấn. Bạn có thể dùng VPD cho bảng (table) và chế độ xem (view), nhưng không dùng cho từ đồng nghĩa (synonyms). Bạn chỉ định tên cột quan trọng bằng tham số sec\_relevant\_cols trong thủ tục DBMS\_RLS.ADD\_POLICY, chính sách bảo mật sẽ được áp dụng mỗi khi cột được truy vấn, dù rõ ràng hay ngầm định.

Ví dụ, người dùng ngoài phòng HR thường chỉ được phép xem số Bảo hiểm Xã hội của họ. Khi một nhân viên bán hàng khởi tạo truy vấn sau:

```
SELECT fname, lname, ssn FROM emp;
```

Hàm thực hiện chính sách bảo mật trả về tiền đề ssn='my\_ssn' và cơ sở dữ liệu sẽ sửa

đổi lại truy vấn và thực thi như sau:

```
SELECT fname, lname, ssn FROM emp WHERE ssn = 'my_ssn';
```

Nếu truy vấn liên quan đến một cột nhạy cảm, thì mặc định VPD giới hạn số hàng trả về. Nếu sử dụng chức năng ẩn cột, kích hoạt bằng tham số `sec_relevant_cols_opt` trong thủ tục `DBMS_RLS.ADD_POLICY`, tất cả hàng sẽ được hiển thị, kể cả hàng liên quan đến cột nhạy cảm. Nhưng cột nhạy cảm sẽ được hiển thị như giá trị `NULL`.

Để hiểu rõ hơn, ta xem xét kết quả truy vấn của nhân viên bán hàng, như trong ví dụ trước. Nếu sử dụng tính năng ẩn cột, nhân viên bán hàng không chỉ xem được hàng chứa thông tin và số Bảo hiểm Xã hội của mình, mà còn xem được tất cả các hàng từ `emp`, nhưng các giá trị cột `ssn` sẽ không có (`NULL`). Điều này hoàn toàn khác với các chính sách VPD khác, chỉ trả về một phần các hàng.

### **Adding Policies for Column-Level VPD.**

VPD cấp cột, có thể áp dụng cho một bảng (table) hoặc một chế độ xem (view), cho phép bạn thiết lập bảo mật khi một cột liên quan đến bảo mật được tham chiếu trong một truy vấn, dẫn đến bảo mật cấp dòng. VPD cấp cột không thể áp dụng cho một synonym.

Nó có thể được cấu hình để tạo ra hai hành vi riêng biệt như sau:

- Default Behavior
- Column-masking Behavior

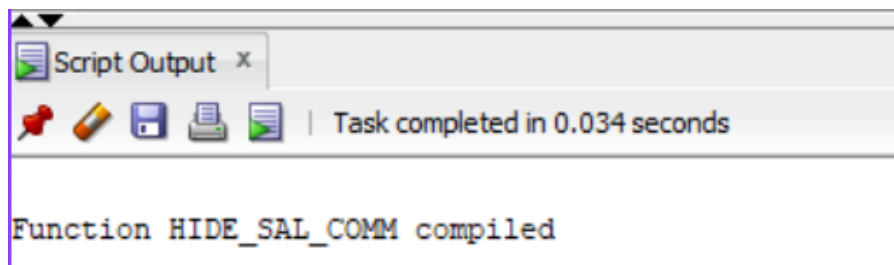
Chính sách VPD trong trường hợp này được thiết lập để ngăn người dùng trong bộ phận bán hàng (`deptno=30`) thấy lương và hoa hồng của những người trong các bộ phận khác. Để thực hiện điều này, chúng ta sẽ tạo một hàm chính sách VPD bằng PL/SQL và sau đó áp dụng nó bằng gói `DBMS_RLS`.

Ví dụ Tạo và Thêm Một Chính Sách VPD Cấp Cột:

Tạo một hàm chính sách không tiết lộ lương của nhân viên ngoài bộ phận bán hàng (`deptno=30`).

```
CREATE OR REPLACE FUNCTION hide_sal_comm (  
    v_schema IN VARCHAR2,  
    v_objname IN VARCHAR2)  
  
    RETURN VARCHAR2 AS  
    con VARCHAR2 (200);  
  
BEGIN  
    con := 'deptno=30';  
    RETURN (con);  
END hide_sal_comm;
```

Hệ thống sẽ báo

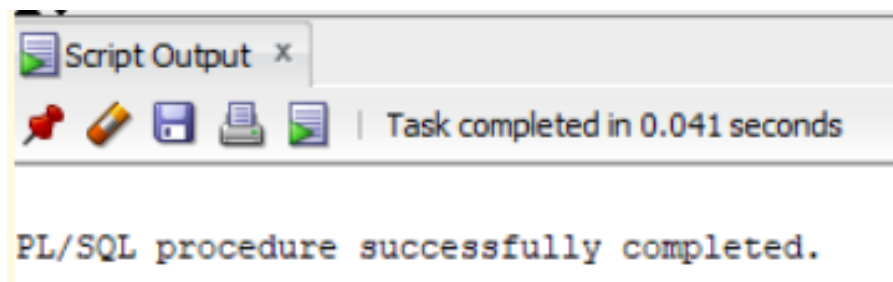


Sau đó, chính sách được thêm bằng gói DBMS\_RLS như sau:

```
BEGIN  
    DBMS_RLS.ADD_POLICY (  
        object_schema => 'scott',  
        object_name   => 'emp',  
        policy_name    => 'hide_sal_policy',  
        policy_function => 'hide_sal_comm',  
        sec_relevant_cols => 'sal,comm');  
END;
```

Hệ thống báo thành công





### Default Behavior

Default Behavior của VPD Column Level là hạn chế số lượng hàng được trả về cho một truy vấn tham chiếu đến các cột chứa thông tin nhạy cảm. Các cột liên quan đến bảo mật này được chỉ định bằng tham số `sec_relevant_cols` của thủ tục `DBMS_RLS.ADD_POLICY`.

Đối với một ví dụ về hành vi mặc định của VPD Column Level, ta hãy xem xét người dùng của bộ phận bán hàng có quyền `SELECT` trên bảng `emp`, được bảo vệ bằng chính sách VPD cấp cột đã được tạo trong Ví dụ trên. Khi người dùng sử dụng câu truy vấn sau:

```
SELECT ENAME, d.dname, job, sal, comm
FROM emp e, dept d
WHERE d.deptno = e.deptno;
```

Cơ sở dữ liệu sẽ trả về một bảng gồm các hàng như sau:

ENAME	DNAME	JOB	SAL	COMM
ALLEN	SALES	SALESMAN	1600	300
WARD	SALES	SALESMAN	1250	500
MARTIN	SALES	SALESMAN	1250	1400
BLAKE	SALES	MANAGER	2850	
TURNER	SALES	SALESMAN	1500	0
JAMES	SALES	CLERK	950	

6 rows selected.

Chỉ các hàng mà người dùng có quyền truy cập vào tất cả các cột mới được hiển thị.

*Column-masking Behavior* Column-masking Behavior khác với Default behavior của VPD Column Level, Column-masking Behavior hiển thị tất cả các hàng, nhưng trả về các giá trị của cột nhạy cảm như `NULL`. Để bao gồm Column-masking trong chính sách của bạn, bạn cần thiết lập tham số `sec_relevant_cols_opt` của thủ tục `DBMS_RLS.ADD_POLICY` thành

dbms\_ols.ALL\_ROWS. Và cũng thiết lập tham số hành vi mặc định khác.

Ví dụ dưới đây cho thấy VPD Column Level với Column-masking Behavior. Nó sử dụng cùng một chính sách VPD như ví dụ trên nhưng với sec\_relevant\_cols\_opt được chỉ định là dbms\_ols.ALL\_ROWS.

Ví dụ Thêm Một Chính Sách VPD Column Level với Column-masking Behavior:  
chính sách ALL\_ROWS

Thêm

```
BEGIN
  DBMS_OLS.ADD_POLICY(
    object_schema      => 'scott',
    object_name        => 'emp',
    policy_name        => 'hide_sal_policy',
    policy_function     => 'hide_sal_comm',
    sec_relevant_cols  => ' sal,comm',
    sec_relevant_cols_opt => dbms_ols.ALL_ROWS);
END;
```

Giả sử một người dùng của bộ phận bán hàng có quyền SELECT trên bảng emp chạy truy vấn sau:

```
SELECT ENAME, d.dname, job, sal, comm
FROM emp e, dept d
WHERE d.deptno = e.deptno;
```

Cơ sở dữ liệu sẽ trả về tất cả các hàng được chỉ định trong truy vấn, nhưng với một số giá trị được che giấu do chính sách VPD:

ENAME	DNAME	JOB	SAL	COMM
CLARK	ACCOUNTING	MANAGER		
KING	ACCOUNTING	PRESIDENT		
MILLER	ACCOUNTING	CLERK		
JONES	RESEARCH	MANAGER		
FORD	RESEARCH	ANALYST		
ADAMS	RESEARCH	CLERK		
SMITH	RESEARCH	CLERK		
SCOTT	RESEARCH	ANALYST		
WARD	SALES	SALESMAN	1250	500
TURNER	SALES	SALESMAN	1500	0
ALLEN	SALES	SALESMAN	1600	300
JAMES	SALES	CLERK	950	
BLAKE	SALES	MANAGER	2850	
MARTIN	SALES	SALESMAN	1250	1400

14 rows selected.

Lưu ý rằng Column-masking đã trả về tất cả các hàng được yêu cầu bởi truy vấn của người dùng bán hàng, nhưng đã làm cho cột SAL và COMM thành NULL cho nhân viên ở ngoài bộ phận bán hàng.

Các lưu ý sau áp dụng cho Column-masking:

- Column-masking chỉ áp dụng cho các câu lệnh SELECT.
- Các điều kiện Column-masking được tạo ra bởi hàm chính sách phải là các biểu thức Boolean đơn giản, không giống như các vị từ VPD thông thường.
- Đối với các ứng dụng thực hiện tính toán hoặc không mong đợi các giá trị NULL, hãy sử dụng VPD cấp cột tiêu chuẩn, chỉ định sec\_relevant\_cols thay vì Column-masking.
- Column-masking được sử dụng với UPDATE AS SELECT sẽ chỉ cập nhật các cột mà người dùng được phép nhìn thấy.
- Đối với một số truy vấn, Column-masking có thể ngăn một số hàng không hiển thị. Ví dụ:

```
SELECT * FROM employees  
WHERE salary = 10
```

- Truy vấn này có thể không trả về hàng nào nếu cột lương trả về giá trị NULL, vì tùy chọn Column-masking đã được thiết lập.

### 3. VPD Security Policy

#### 3.1. *Apply Policies on table, view, synonym*

Gói PL/SQL DBMS\_RLS có thể áp dụng một Policies vào một vào bảng (table), chế độ xem (view), hoặc từ đồng nghĩa (synonym).

Để áp dụng một chính sách vào một vào bảng (table), chế độ xem (view), hoặc từ đồng nghĩa (synonym) sử dụng thủ tục DBMS\_RLS.ADD\_POLICY.

Người dùng phải chỉ định bảng, chế độ xem hoặc đồng nghĩa mà họ đang thêm chính sách, và một tên cho chính sách. Người dùng cũng có thể chỉ định thông tin khác, chẳng hạn như các loại câu lệnh chính sách kiểm soát (SELECT, INSERT, UPDATE, DELETE, CREATE INDEX hoặc ALTER INDEX).

Có các chú ý cho việc áp dụng chính sách:

- Oracle khuyến nghị áp dụng chính sách VPD cho các chế độ xem một cách tương tự như cho các đối tượng cơ sở của chúng.
- Trước khi thêm chính sách VPD vào một đối tượng cơ sở, kiểm tra xem có các đối tượng phụ thuộc không. Nếu có, các đối tượng này sẽ trở thành không hợp lệ khi chính sách VPD được áp dụng, và cần phải được biên dịch lại bằng cách sử dụng câu lệnh ALTER ... COMPILE. Tuy nhiên, việc này có thể ảnh hưởng đến hiệu suất hệ thống, nên nên thực hiện trong giờ ngoài giờ cao điểm hoặc khi hệ thống không hoạt động.
- Lưu ý rằng mỗi đối tượng chỉ có thể có tối đa 255 chính sách VPD.

Ví dụ Áp dụng một Chính sách Oracle vào Một Bảng

Ví dụ dưới cho thấy cách sử dụng DBMS\_RLS.ADD\_POLICY để áp dụng một chính sách Oracle VPD có tên là secure\_update vào bảng HR.EMPLOYEES. Hàm được gắn vào chính sách là check\_updates.

```
BEGIN
  DBMS_RLS.ADD_POLICY (
    object_schema    => 'hr',
    object_name      => 'employees',
    policy_name      => 'secure_update',
    policy_function   => 'check_updates',
    ...
```

Nếu hàm được tạo bên trong một gói, hãy bao gồm tên gói. Ví dụ:

```
policy_function => 'pkg.check_updates',
```

Mặc dù người dùng có thể xác định một chính sách đối với một bảng, nhưng không thể chọn bảng đó từ trong chính sách đã được xác định cho bảng.

### 3.2. *Organize Policies as Groups*

Người dùng có thể nhóm nhiều chính sách bảo mật lại với nhau và áp dụng chúng cho một ứng dụng.

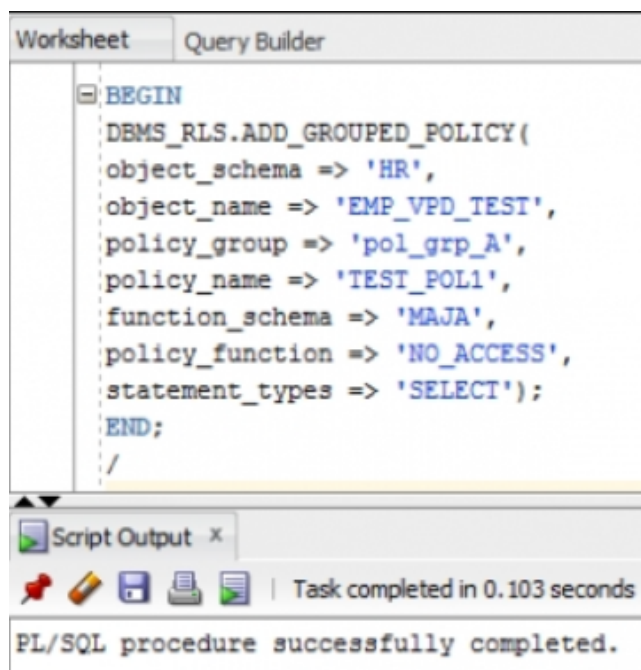
Một nhóm chính sách là một tập hợp các chính sách bảo mật liên quan đến một ứng dụng cụ thể. Bằng cách chỉ định một ngữ cảnh ứng dụng (còn được gọi là ngữ cảnh thực thi hoặc ngữ cảnh chính sách), người dùng xác định nhóm chính sách nào sẽ được áp dụng. Khi người dùng truy cập vào bảng, view, hoặc cột tương ứng, Oracle Database sẽ tra cứu ngữ cảnh ứng dụng để xác định nhóm chính sách đang có hiệu lực và thực thi tất cả các chính sách trong nhóm đó.

Nhóm chính sách giúp xác định chính xác những quy tắc nào sẽ được áp dụng khi nhiều ứng dụng chia sẻ cùng một đối tượng. Điều này giúp quản lý và duy trì bảo mật một cách dễ dàng hơn.

Công ty A lưu trữ bảng BENEFIT cho Công ty B và Công ty C. Hai ứng dụng khác nhau là Nhân sự và Tài chính truy cập vào bảng này với các chính sách bảo mật khác nhau. Ứng dụng Nhân sự ủy quyền dựa trên cấp bậc, trong khi ứng dụng Tài chính ủy quyền dựa trên phòng ban. Để không cần phải phát triển chung các chính sách, mỗi ứng dụng có thể áp dụng chính sách riêng bằng cách xác định một ngữ cảnh ứng dụng.

Để thực hiện điều này, cần tổ chức các chính sách bảo mật thành các nhóm. Bằng cách tham chiếu đến application context, Oracle Database xác định nhóm chính sách nào nên có hiệu lực tại thời điểm chạy. Máy chủ thực thi tất cả các chính sách thuộc về nhóm chính sách

đó.



```
BEGIN
DBMS_RLS.ADD_GROUPED_POLICY(
object_schema => 'HR',
object_name => 'EMP_VPD_TEST',
policy_group => 'pol_grp_A',
policy_name => 'TEST_POL1',
function_schema => 'MAJA',
policy_function => 'NO_ACCESS',
statement_types => 'SELECT');
END;
/
```

Script Output x

Task completed in 0.103 seconds

PL/SQL procedure successfully completed.

**object\_schema:** Schema của đối tượng mà chính sách sẽ được áp dụng. Đây là một chuỗi ký tự đại diện cho tên của schema.

**object\_name:** Tên của đối tượng mà chính sách sẽ được áp dụng. Đây là một chuỗi ký tự đại diện cho tên của đối tượng (ví dụ: bảng, view) trong cơ sở dữ liệu.

**policy\_name:** Tên của chính sách cụ thể. Đây là một chuỗi ký tự đại diện cho tên của chính sách phân quyền.

**policy\_function\_schema:** Schema chứa hàm hoặc điều kiện được sử dụng để áp dụng chính sách. Đây là một chuỗi ký tự đại diện cho tên của schema.

**policy\_function:** Tên của hàm hoặc điều kiện logic được sử dụng để xác định quyền truy cập. Đây là một chuỗi ký tự đại diện cho tên của hàm hoặc điều kiện.

**statement\_types:** Loại câu lệnh SQL mà chính sách sẽ áp dụng (SELECT, INSERT, UPDATE, DELETE). Đây là một mảng chứa các giá trị từ 1 đến 4 để biểu thị các loại câu lệnh.

**update\_check:** Cờ chỉ định xem chính sách sẽ kiểm tra điều kiện cập nhật hay không. Đây là một giá trị boolean (TRUE hoặc FALSE).

**enable:** Cờ chỉ định xem chính sách sẽ được kích hoạt ngay lập tức sau khi thêm hay không. Đây là một giá trị boolean (TRUE hoặc FALSE).

**static\_policy:** Cờ chỉ định xem chính sách là tĩnh hay không (tĩnh có nghĩa là nó không sử dụng các hàm phân quyền). Đây là một giá trị boolean (TRUE hoặc FALSE).

**policy\_type:** Loại của chính sách (e.g., "GROUP", "STATIC"). Đây là một chuỗi ký tự đại diện cho loại chính sách.

### Creation of a New Oracle Virtual Private Database Policy Group

Thủ tục DBMS\_RLS.ADD\_GROUPED\_POLICY được sử dụng để thêm một chính sách VPD vào một nhóm chính sách VPD.

Để chỉ định những chính sách nào sẽ có hiệu lực, bạn có thể thêm một ngữ cảnh thực thi bằng cách sử dụng thủ tục DBMS\_RLS.ADD\_POLICY\_CONTEXT. Nếu ngữ cảnh thực thi trả về một nhóm chính sách không xác định thì sẽ xảy ra lỗi.

Nếu không có ngữ cảnh thực thi được xác định, Oracle Database sẽ thực thi tất cả các chính sách bảo mật. Tương tự, nếu ngữ cảnh thực thi là NULL, thì tất cả các chính sách từ tất cả các nhóm chính sách sẽ được thực thi. Điều này có nghĩa là một ứng dụng truy cập vào dữ liệu không thể bỏ qua bất kỳ chính sách bảo mật nào được áp dụng.

Người dùng có thể áp dụng nhiều ngữ cảnh thực thi cho cùng một bảng, chế độ xem hoặc từ đồng nghĩa, và mỗi ngữ cảnh sẽ được xử lý một cách độc lập. Điều này cho phép bạn cấu hình nhiều bộ chính sách hoạt động để thực thi.

Để tạo các nhóm chính sách, người quản trị phải thực hiện hai công việc:

- Thiết lập một ngữ cảnh thực thi để xác định nhóm chính sách có hiệu lực.
- Thêm các chính sách vào các nhóm chính sách theo yêu cầu.

Ví dụ, trong một công ty lưu trữ có hai ứng dụng Benefits và Financial chia sẻ một số đối tượng cơ sở dữ liệu. Cả hai ứng dụng đều sử dụng chính sách SUBSCRIBER trong nhóm chính sách SYS\_DEFAULT. Truy cập dữ liệu được phân chia trước tiên theo ID của người đăng ký, sau đó là dựa vào ứng dụng mà người dùng đang truy cập (được xác định bằng một ngữ cảnh thực thi). Nếu Công ty A muốn áp dụng một chính sách tùy chỉnh cho việc truy cập dữ liệu của mình, chúng ta có thể thêm một ngữ cảnh thực thi bổ sung (như COMPANY A SPECIAL). Điều này đảm bảo rằng nhóm chính sách bổ sung này chỉ được áp dụng cho việc truy cập dữ liệu của Công ty A. Không nên áp dụng điều này dưới chính sách SUBSCRIBER vì chính sách này chỉ liên quan đến Công ty A, và việc tách riêng chính sách lưu trữ cơ bản khỏi các chính sách khác sẽ hiệu quả hơn.

Lưu ý: Cần thiết lập các cấu trúc dữ liệu cũng như cấp quyền sau đây để các ví dụ trong phần này hoạt động



```
DROP USER finance CASCADE;
CREATE USER finance IDENTIFIED BY welcome2;
GRANT RESOURCE TO apps;
DROP TABLE apps.benefit;
CREATE TABLE apps.benefit (c NUMBER);
```

Bước 1: Thiết lập một Ngữ cảnh Thực thi

Bắt đầu bằng cách tạo một không gian tên cho ngữ cảnh thực thi. Ví dụ:

```
CREATE CONTEXT appsctx USING apps.apps_security_init;
```

Tạo package quản lý ngữ cảnh thực thi. Ví dụ:

```
CREATE OR REPLACE PACKAGE apps.apps_security_init IS
PROCEDURE setctx (policy_group VARCHAR2);
END;

CREATE OR REPLACE PACKAGE BODY apps.apps_security_init AS
PROCEDURE setctx ( policy_group varchar2 ) IS
BEGIN

REM Do some checking to determine the current application.
REM You can check the proxy if using the proxy authentication feat
REM Then set the context to indicate the current application.
.
.
.
DBMS_SESSION.SET_CONTEXT('APPSCTX','ACTIVE_APPS', policy_group);
END;
END;
```

Xác định ngữ cảnh thực thi cho bảng APPS.BENEFIT.

```
BEGIN
DBMS_RLS.ADD_POLICY_CONTEXT('apps','benefit','APPSCTX','ACTIVE_APPS');
END;
```

Bước 2: Thêm một Chính sách vào Nhóm Chính sách Mặc định

Tạo một hàm bảo mật để trả về một điều kiện để chia dữ liệu theo công ty.



```
CREATE OR REPLACE FUNCTION by_company (sch varchar2, tab varchar2)
RETURN VARCHAR2 AS
BEGIN
    RETURN 'COMPANY = SYS_CONTEXT(''ID'', 'MY_COMPANY')';
END;
```

Do các chính sách trong SYS\_DEFAULT luôn được thực thi (ngoại trừ SYS, hoặc người dùng có quyền hạn EXEMPT ACCESS POLICY), chính sách bảo mật này, sẽ luôn được áp dụng bất kể ứng dụng đang chạy. Hàm APPS.APPS\_SECURITY\_INIT.BY\_COMPANY trả về điều kiện để đảm bảo rằng người dùng chỉ có thể xem dữ liệu liên quan đến công ty của họ:

```
BEGIN
DBMS_RLS.ADD_GROUPED_POLICY('apps', 'benefit', 'SYS_DEFAULT',
'security_by_company',
'apps', 'by_company');
```

Bước 3: Thêm một Chính sách vào Nhóm Chính sách HR

Đầu tiên, tạo nhóm HR:

```
CREATE OR REPLACE FUNCTION hr.security_policy
RETURN VARCHAR2
AS
BEGIN
    RETURN 'SYS_CONTEXT(''ID'', 'TITLE') = 'MANAGER' ';
END;
```

Sau đó tạo nhóm chính sách và thêm một chính sách có tên HR\_SECURITY vào nhóm chính sách HR. Hàm HR.SECURITY\_POLICY trả về điều kiện để thực thi bảo mật trên bảng APPS.BENEFIT:

```
BEGIN
DBMS_RLS.CREATE_POLICY_GROUP('apps', 'benefit', 'HR');
DBMS_RLS.ADD_GROUPED_POLICY('apps', 'benefit', 'HR',
'hr_security', 'hr', 'security_policy');
END;
```

Bước 4: Thêm một Chính sách vào Nhóm Chính sách FINANCE

Tạo chính sách FINANCE:

```
CREATE OR REPLACE FUNCTION finance.security_policy
RETURN VARCHAR2
AS
BEGIN
    RETURN ('SYS_CONTEXT(''ID'', ''DEPT'') = ''FINANCE'' ');
END;
```

Tạo một nhóm chính sách có tên FINANCE và thêm chính sách FINANCE vào nhóm FINANCE:

```
BEGIN
DBMS_RLS.CREATE_POLICY_GROUP('apps','benefit','FINANCE');
DBMS_RLS.ADD_GROUPED_POLICY('apps','benefit','FINANCE',
'finance_security','finance','security_policy');
END;
```

Kết quả là, khi cơ sở dữ liệu được truy cập, ứng dụng khởi tạo ngữ cảnh thực thi sau xác thực. Ví dụ, với HR:

```
execute apps.security_init.setctx('HR');
```

### 3.3. *Default Policies*

Trong một nhóm chính sách bảo mật, người dùng có thể chỉ định một chính sách bảo mật là chính sách mặc định.

Trong các tình huống khi bạn cần phân vùng các chính sách bảo mật theo ứng dụng để đảm bảo áp dụng liên tục, việc sử dụng các nhóm chính sách là rất hữu ích. Các chính sách bảo mật mặc định có thể cho phép các nhà phát triển áp dụng bảo mật trong mọi trường hợp, trong khi phân vùng chúng theo ứng dụng (bằng cách sử dụng các nhóm bảo mật) cho phép bạn đặt ưu tiên cao hơn cho bảo mật cụ thể của ứng dụng so với các chính sách mặc định. Để triển khai các chính sách bảo mật mặc định, bạn chỉ cần thêm chúng vào nhóm chính sách SYS\_DEFAULT.

Trong nhóm chính sách này, các chính sách được xác định cho một bảng, view, hoặc đồng nghĩa cụ thể sẽ được thực thi theo nhóm chính sách được chỉ định bởi ngữ cảnh thực thi. Một ngữ cảnh thực thi là một ngữ cảnh ứng dụng chỉ định nhóm chính sách đang có hiệu lực.

Nhóm chính sách SYS\_DEFAULT có thể có hoặc không có các chính sách. Tuy nhiên, không thể loại bỏ nhóm chính sách SYS\_DEFAULT. Nếu thử loại bỏ, Oracle Database sẽ hiển

thì một lỗi.

Nếu bạn thêm các chính sách liên kết với hai hoặc nhiều đối tượng cho nhóm chính sách SYS\_DEFAULT, mỗi đối tượng sẽ có một nhóm chính sách SYS\_DEFAULT riêng biệt được liên kết với nó. Ví dụ, bảng emp trong schema scott có một nhóm chính sách SYS\_DEFAULT, và bảng dept trong schema scott có một nhóm chính sách SYS\_DEFAULT khác được liên kết với nó. Ta có thể thấy chúng được tổ chức như một cấu trúc cây như sau:

```
SYS_DEFAULT
- policy1 (scott/emp)
- policy3 (scott/emp)
SYS_DEFAULT
- policy2 (scott/dept)
```

Người dùng có thể tạo các nhóm chính sách có cùng tên. Khi họ chọn một nhóm chính sách cụ thể, tên schema cùng với các đối tượng liên kết sẽ được hiển thị trong bảng thuộc tính ở phía bên phải của màn hình. Điều này giúp dễ dàng quản lý và theo dõi các chính sách được áp dụng cho từng đối tượng.

## 4. Configure VPD for Fine-grained Access Control

### 4.1. Use the DBMS\_RLS Package and the Create Context command

Để cấu hình VPD cho kiểm soát truy cập cấp dòng trong Oracle bằng cách sử dụng gói DBMS\_RLS và lệnh Create Context:

- Tạo Hàm Chính Sách Bảo Mật:

Tạo một hàm PL/SQL để định nghĩa chính sách bảo mật, quyết định điều kiện kiểm soát truy cập dựa trên thông tin phiên.

- Liên Kết Hàm Chính Sách với Bảng:

Sử dụng thủ tục ADD\_POLICY trong gói DBMS\_RLS để liên kết hàm chính sách bảo mật với một bảng cụ thể.

- Tạo Bối Cảnh:

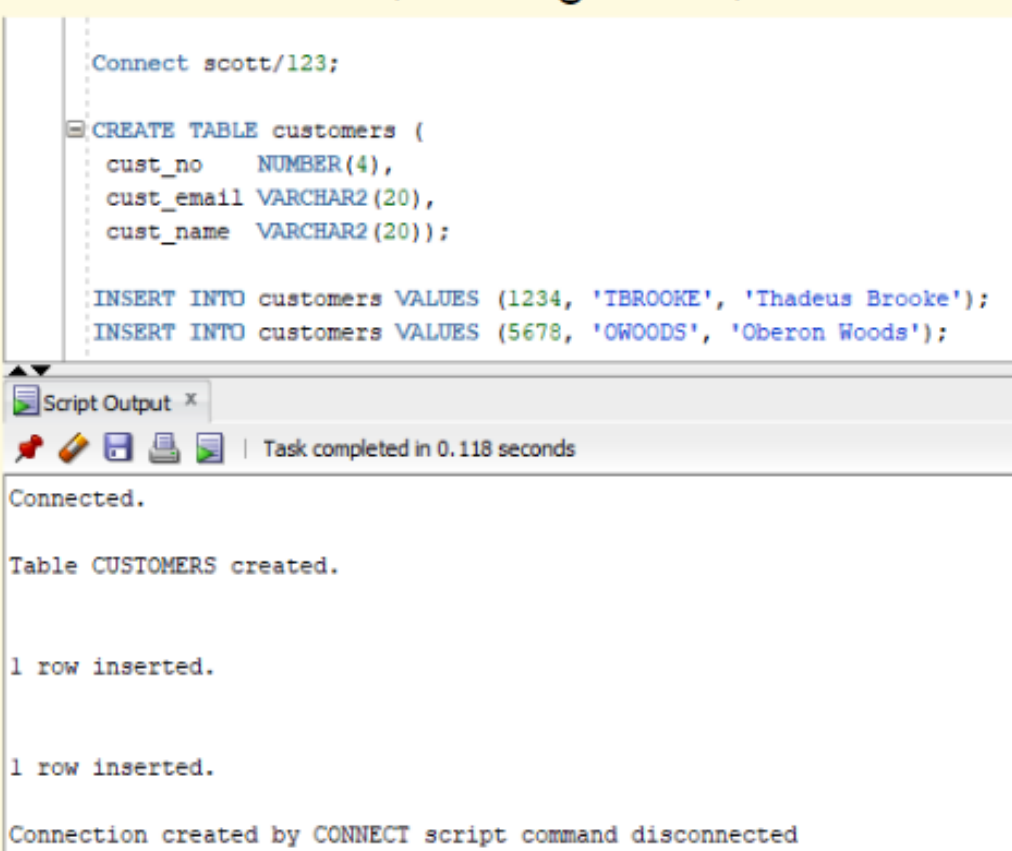
Tạo một bối cảnh để lưu trữ thông tin phiên sử dụng lệnh CREATE CONTEXT.

- Đặt Bối Cảnh:

Sử dụng thủ tục SET\_CONTEXT để đặt giá trị cho bối cảnh được tạo.

Ví dụ

Bước 1: Tạo bảng dữ liệu



```
Connect scott/123;

CREATE TABLE customers (
  cust_no    NUMBER(4),
  cust_email VARCHAR2(20),
  cust_name  VARCHAR2(20));

INSERT INTO customers VALUES (1234, 'TBROOKE', 'Thadeus Brooke');
INSERT INTO customers VALUES (5678, 'OWOODS', 'Oberon Woods');
```

Script Output x

Task completed in 0.118 seconds

Connected.

Table CUSTOMERS created.

1 row inserted.

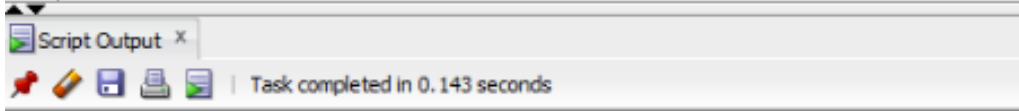
1 row inserted.

Connection created by CONNECT script command disconnected

```
Connect scott/123;

CREATE TABLE orders_tab (
  cust_no NUMBER(4),
  order_no NUMBER(4));

INSERT INTO orders_tab VALUES (1234, 9876);
INSERT INTO orders_tab VALUES (5678, 5432);
INSERT INTO orders_tab VALUES (5678, 4592);
```



Script Output x

Task completed in 0.143 seconds

Table ORDERS\_TAB created.

1 row inserted.

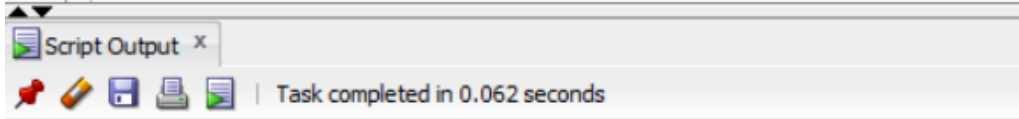
1 row inserted.

1 row inserted.

Connection created by CONNECT script command disconnected

### Tạo người dùng

```
GRANT CREATE SESSION TO tbrooke IDENTIFIED BY 123;
GRANT CREATE SESSION TO owoods IDENTIFIED BY 123;
```



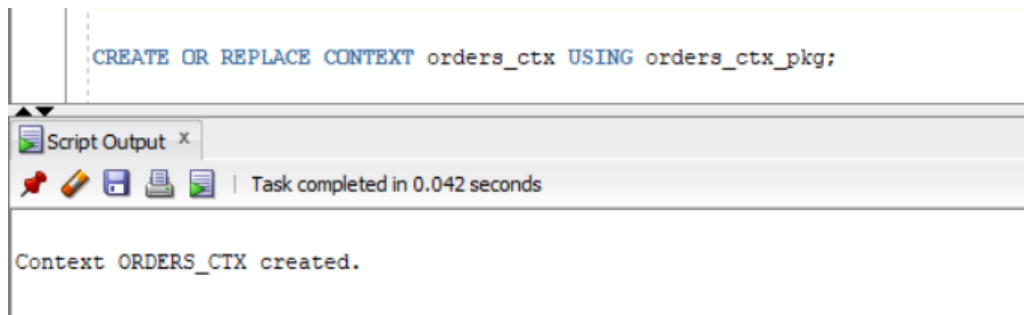
Script Output x

Task completed in 0.062 seconds

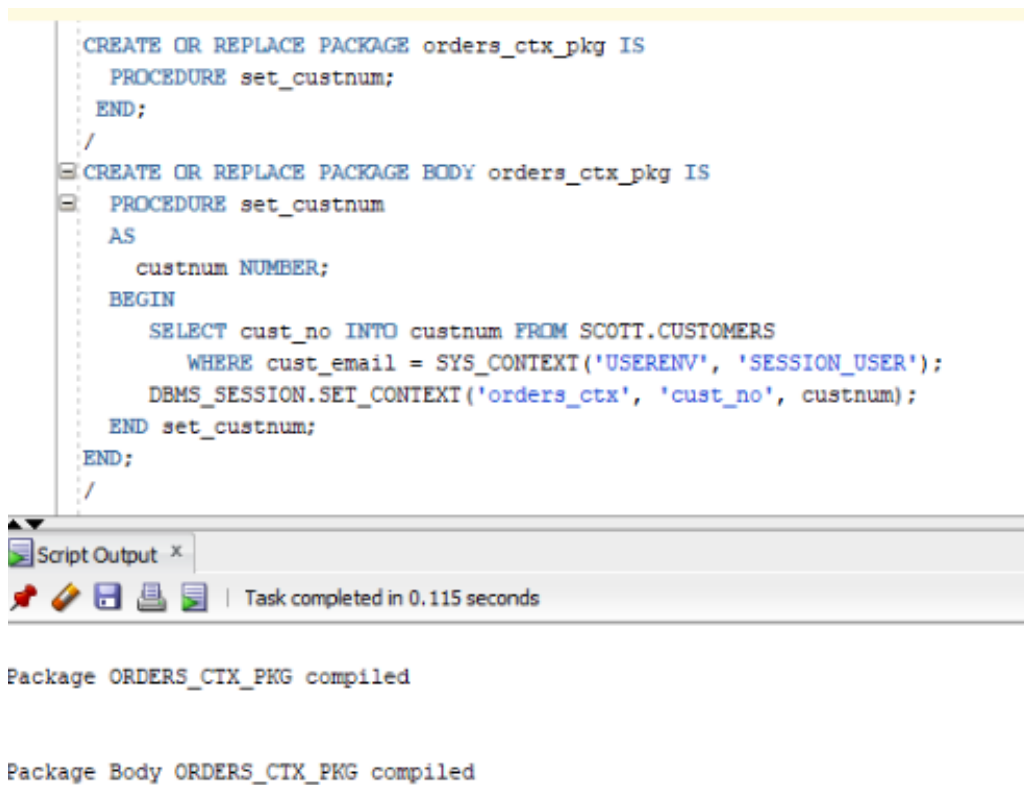
Grant succeeded.

Grant succeeded.

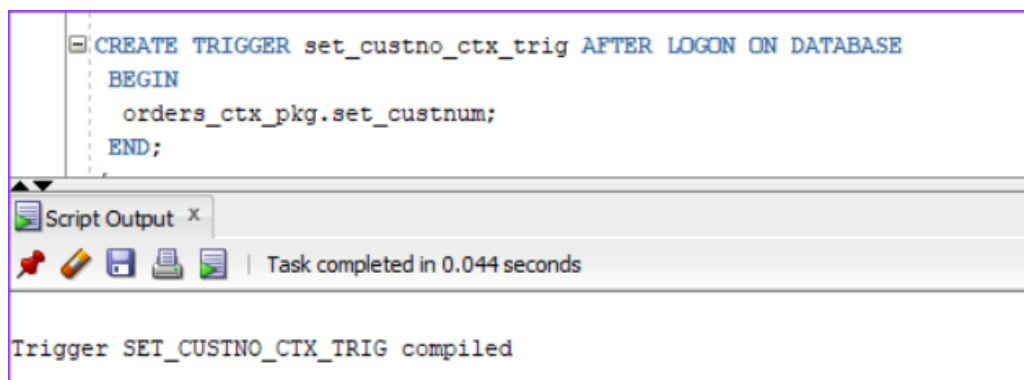
### Bước 2: Tạo Application context



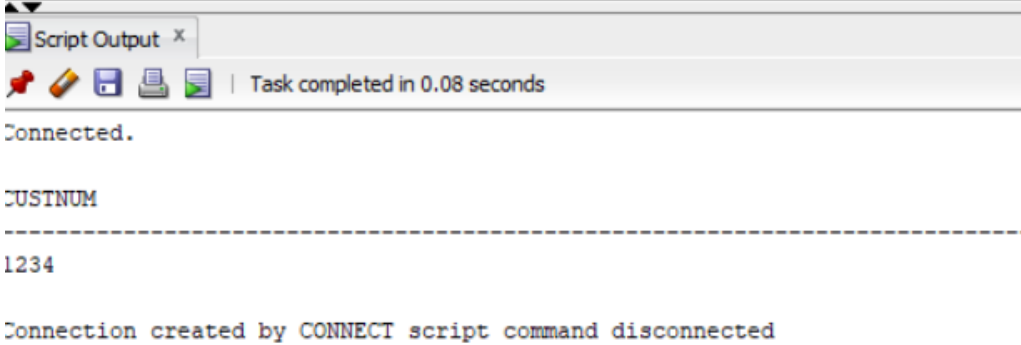
### Bước 3: Tạo package



### Bước 4: Tạo trigger



```
CONNECT tbrooke/123;  
SELECT SYS_CONTEXT('orders_ctx', 'cust_no') custnum FROM DUAL;
```



Script Output x

Task completed in 0.08 seconds

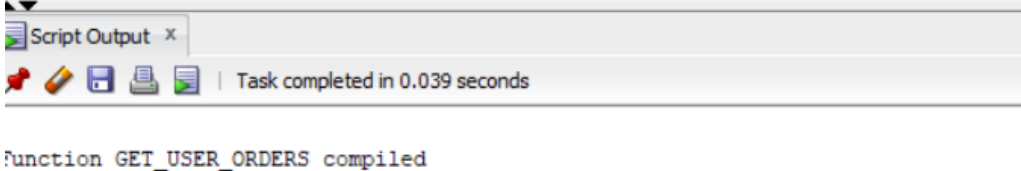
Connected.

CUSTNUM
1234

Connection created by CONNECT script command disconnected

### Bước 5: Tạo Hàm Chính Sách Bảo Mật

```
CREATE OR REPLACE FUNCTION get_user_orders(  
  schema_p IN VARCHAR2,  
  table_p IN VARCHAR2)  
RETURN VARCHAR2  
AS  
  orders_pred VARCHAR2 (400);  
BEGIN  
  orders_pred := 'cust_no = SYS_CONTEXT(''orders_ctx'', ''cust_no'')';  
RETURN orders_pred;  
END;
```



Script Output x

Task completed in 0.039 seconds

Function GET\_USER\_ORDERS compiled

### Bước 6: Liên Kết Hàm Chính Sách với Bảng

```
BEGIN
  DBMS_RLS.ADD_POLICY (
    object_schema => 'scott',
    object_name   => 'orders_tab',
    policy_name   => 'orders_policy',
    policy_function => 'get_user_orders',
    statement_types => 'select');
END;
```

Script Output x

Task completed in 0.225 seconds

PL/SQL procedure successfully completed.

## Bước 7: Test

```
Grant select on scott.orders_tab to tbrooke;
Grant select on scott.orders_tab to owoods;
```

Script Output x

Task completed in 0.052 seconds

Grant succeeded.

Grant succeeded.

Hai bảng trả ra hai kết quả khác nhau do VPD đã áp dụng chính sách lên bảng

```
CONNECT owoods/123;
select * from scott.orders_tab;
```

Script Output x

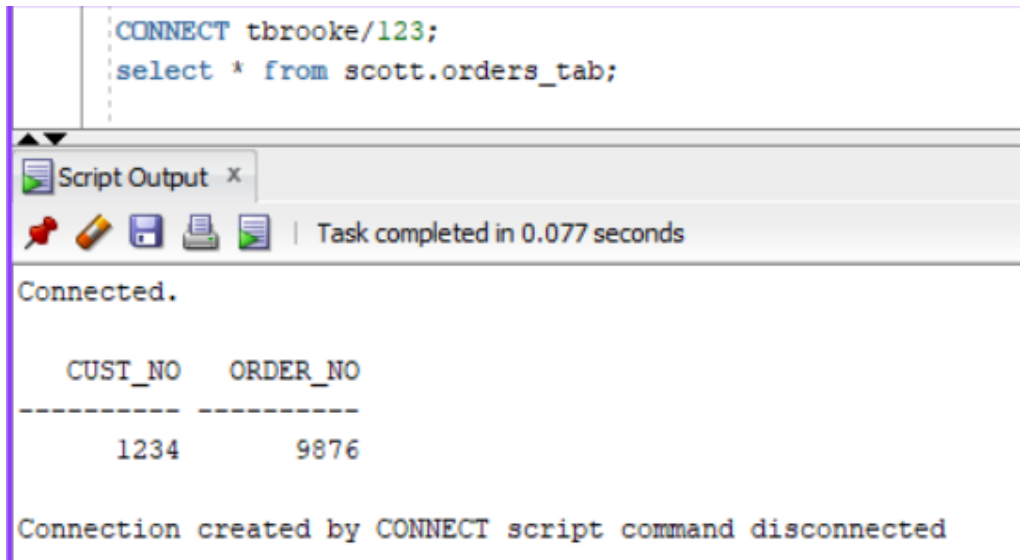
Task completed in 0.086 seconds

Connected.

CUST_NO	ORDER_NO
5678	5432
5678	4592

Connection created by CONNECT script command disconnected





```
CONNECT tbrooke/123;  
select * from scott.orders_tab;
```

Script Output x

Task completed in 0.077 seconds

Connected.

CUST_NO	ORDER_NO
1234	9876

Connection created by CONNECT script command disconnected

- Các thủ tục trong gói DBMS\_RLS

*Xử lý các chính sách cá nhân:*

- **DBMS\_RLS.ADD\_POLICY**: Thêm một chính sách vào một bảng, chế độ xem hoặc từ đồng nghĩa.
- **DBMS\_RLS.ENABLE\_POLICY**: Kích hoạt (hoặc vô hiệu hóa) một chính sách đã được thêm vào một bảng, chế độ xem hoặc từ đồng nghĩa trước đó.
- **DBMS\_RLS.ALTER\_POLICY**: Sửa đổi một chính sách hiện có để liên kết hoặc tách rời các thuộc tính với chính sách.
- **DBMS\_RLS.REFRESH\_POLICY**: Vô hiệu hóa các con trỏ liên quan với các chính sách không tĩnh.
- **DBMS\_RLS.DROP\_POLICY**: Loại bỏ một chính sách từ một bảng, chế độ xem hoặc từ đồng nghĩa.

*Xử lý các nhóm chính sách:*

- **DBMS\_RLS.CREATE\_POLICY\_GROUP**: Tạo một nhóm chính sách.
- **DBMS\_RLS.ALTER\_GROUPED\_POLICY**: Sửa đổi một nhóm chính sách.
- **DBMS\_RLS.DELETE\_POLICY\_GROUP**: Xóa một nhóm chính sách.
- **DBMS\_RLS.ADD\_GROUPED\_POLICY**: Thêm một chính sách vào nhóm chính sách cụ thể.

- **DBMS\_RLS.ENABLE\_GROUPED\_POLICY**: Kích hoạt một chính sách trong một nhóm.
- **DBMS\_RLS.REFRESH\_GROUPED\_POLICY**: Phân tích lại các lệnh SQL liên quan đến một chính sách được làm mới.
- **DBMS\_RLS.DISABLE\_GROUPED\_POLICY**: Vô hiệu hóa một chính sách trong một nhóm.
- **DBMS\_RLS.DROP\_GROUPED\_POLICY**: Xóa một chính sách là thành viên của nhóm đã chỉ định.

*Xử lý bối cảnh ứng dụng:*

- **DBMS\_RLS.ADD\_POLICY\_CONTEXT**: Thêm bối cảnh cho ứng dụng đang hoạt động.
- **DBMS\_RLS.DROP\_POLICY\_CONTEXT**: Xóa bối cảnh cho ứng dụng.

#### ***4.2. Use the Oracle Policy Manager Interface***

Quản trị viên bảo mật có nhiều lựa chọn để quản lý các chính sách ứng dụng, bao gồm việc sử dụng các lệnh WebLogic Scripting Tool (WLST) hoặc Fusion Middleware Control. Trong khi các lệnh WLST đòi hỏi thực thi thủ công, Fusion Middleware Control cung cấp một giao diện đồ họa nhưng phức tạp và đòi hỏi sự quen thuộc với các tài liệu bảo mật cấp thấp và kiến thức về các tên và khái niệm thường chỉ quen thuộc với các nhà phát triển (như tên lớp quyền hoặc tên tác vụ).

Authorization Policy Manager là công cụ đơn giản hóa việc tạo, cấu hình và quản lý chính sách ứng dụng so với hai công cụ khác với các tính năng sau:

- Tên và mô tả dễ hiểu về các tài liệu bảo mật.
- Tính linh hoạt trong việc tổ chức vai trò ứng dụng dựa trên các tham số khác nhau như kinh doanh hoặc sản phẩm.
- Giao diện đồ họa thống nhất giúp tìm kiếm, tạo, duyệt và chỉnh sửa các tài liệu bảo mật một cách thuận tiện.
- Khả năng chỉ định phần của ứng dụng mà một vai trò có thể quản lý, tăng tính linh hoạt và kiểm soát đối với các nhiệm vụ quản lý.

## Tạo chính sách

Quy trình tạo chính sách ứng dụng dựa trên một vai trò ứng dụng được mô tả, cùng với các cách thay thế khác như dựa trên một nguyên tắc, một quyền hoặc một tài nguyên, được cung cấp qua menu "New Policy".

Để tạo một chính sách ứng dụng dựa trên một vai trò ứng dụng cụ thể, thực hiện như sau:

- 1. Chọn Policies dưới ứng dụng mà bạn muốn tạo chính sách, và nhấp đúp vào nó hoặc nhấp Open để hiển thị trang Search - Policies.
- 2. Trong trang đó, đưa tab Principal lên phía trước và chỉ định các tham số cho một Search, để định vị và chọn các nguyên tắc (vai trò ứng dụng, vai trò bên ngoài, hoặc người dùng) trên đó dựa chính sách đang được tạo.
- 3. Trong tab Function Security, ở khu vực dưới cùng của trang, chọn Entitlement Policies hoặc Resource Based Policies (tùy theo loại chính sách muốn tạo), sau đó nhấp New Policy để hiển thị một trang chính sách Untitled.

Nếu tạo một chính sách dựa trên quyền lợi, sau đó trong trang Không tiêu đề:

a. Thêm các nguyên tắc vào chính sách - Sử dụng nút Add ở đầu bảng Principal, hoặc, tùy chọn khác, thực hiện một tìm kiếm đơn giản về các vai trò ứng dụng, các vai trò bên ngoài, hoặc người dùng, và kéo và thả các mục từ kết quả tìm kiếm vào bảng Principal.

b. Thêm một quyền lợi vào chính sách - Sử dụng nút Add ở đầu bảng Entitlement, hoặc tùy chọn khác, thực hiện một tìm kiếm đơn giản về các quyền lợi ứng dụng, và kéo và thả một quyền lợi từ kết quả tìm kiếm vào bảng Entitlement.

c. Nhấn Save.

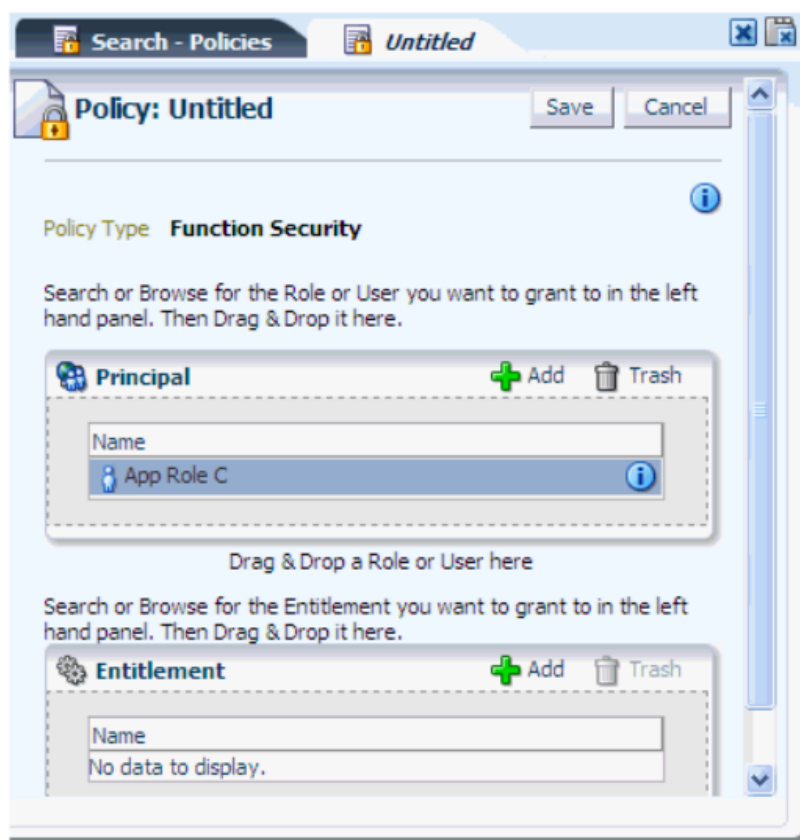
Nếu tạo một chính sách dựa trên tài nguyên, sau đó trong trang Untitled:

a. Thêm các nguyên tắc vào chính sách - Sử dụng nút Add ở đầu bảng Principal, hoặc tùy chọn khác, thực hiện một tìm kiếm đơn giản về các vai trò ứng dụng, các vai trò bên ngoài, hoặc người dùng, và kéo và thả các mục từ kết quả tìm kiếm vào bảng Principal.

b. Thêm các trường hợp tài nguyên vào chính sách - Sử dụng nút Add trong bảng Resources, hoặc tùy chọn khác, thực hiện một tìm kiếm đơn giản về các trường hợp tài nguyên ứng dụng, và kéo và thả một trường hợp tài nguyên từ kết quả tìm kiếm vào bảng Tài nguyên.

c. Đối với mỗi trường hợp tài nguyên được thêm, chọn một trường hợp tài nguyên và chỉ định các hành động được phép bằng cách chọn các hộp kiểm thích hợp trong khu vực Actions ở dưới cùng của trang.

d. Nhấn Save.



### Chỉnh sửa chính sách

Các chính sách dựa trên quyền lợi không thể được sửa đổi.

Để sửa đổi hoặc xem một chính sách dựa trên tài nguyên, thực hiện các bước sau:

1. Xác định chính sách ứng dụng dựa trên tài nguyên để sửa đổi hoặc xem bằng cách sau:

- Bằng cách khớp với một vai trò ứng dụng trong chính sách.
- Bằng cách khớp tên tài nguyên trong chính sách.

2. Chọn chính sách và nhấp Open để mở trang cho chính sách.

3. Trong trang đó, sửa đổi các thuộc tính của chính sách khi cần thiết.

4. Nhấn Apply để lưu các thay đổi.

## 5. Limit the scope of VPD use (in which Schemas)

VPD (Virtual Private Database) là một tính năng trong Oracle Database, được sử dụng để triển khai các chính sách bảo mật dữ liệu bằng cách giới hạn quyền truy cập đến dữ liệu cho người dùng hoặc ứng dụng.

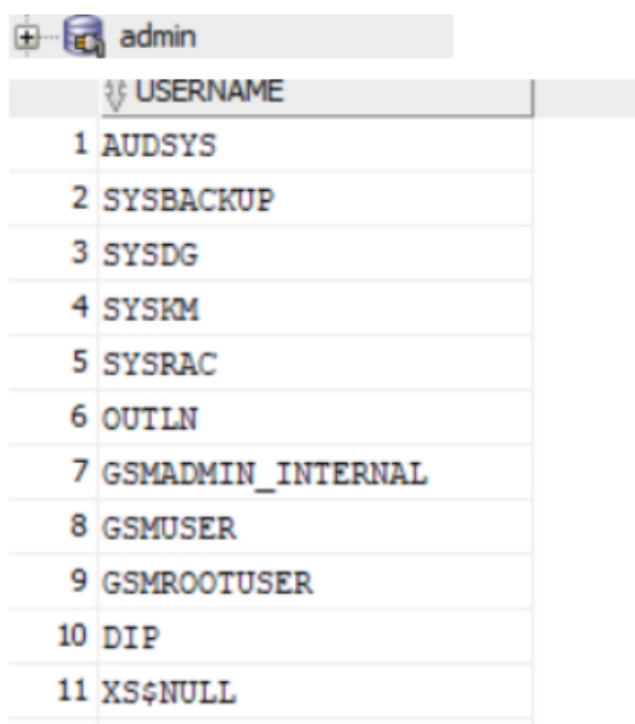
Việc giới hạn phạm vi sử dụng VPD được thực hiện thông qua việc áp dụng chính sách VPD trên các bảng hoặc view cụ thể trong cơ sở dữ liệu.

Đối với việc giới hạn phạm vi sử dụng VPD, bạn có thể chỉ định các schema cụ thể mà chính sách VPD được áp dụng. Điều này có nghĩa là bạn có thể chỉ định rằng chính sách VPD chỉ được kích hoạt trên các bảng hoặc view trong một schema nhất định, và không áp dụng cho các schema khác trong cơ sở dữ liệu.

Đây là cách bạn có thể giới hạn phạm vi sử dụng của VPD cho các Schema cụ thể:

- **Xác định các Schema:** Xác định các schema mà người dùng muốn áp dụng các chính sách VPD. Các schema này chứa các bảng hoặc view mà người dùng muốn bảo vệ bằng VPD.

Ví dụ: Schema admin



	USERNAME
1	AUDSYS
2	SYSBACKUP
3	SYSDG
4	SYSKM
5	SYSRAC
6	OUTLN
7	GSMADMIN_INTERNAL
8	GSMUSER
9	GSMROOTUSER
10	DIP
11	XS\$NULL

- **Tạo các Chính Sách VPD:** Trong mỗi schema, tạo các chính sách VPD trên các bảng hoặc view mà người dùng muốn bảo vệ. Các chính sách này xác định các điều kiện dưới đó các dòng được lọc hoặc che dấu dựa trên ngữ cảnh phiên của người dùng.

```
create or replace FUNCTION admin.PC1_FUNCTION
(P_SCHEMA VARCHAR2, P_OBJ VARCHAR2)
RETURN VARCHAR2
AS
    user varchar2(100);
begin
    user:= sys_context('userenv','session_user');
    IF user = 'ANNU' THEN
        RETURN 'Name = ''Annu''';
    ELSIF user = 'HANN' THEN
        RETURN '1 = 1';
    ELSE |
        RETURN '1 = 0';
    END IF;
end;
```

- Áp dụng các Chính Sách: Sau khi các chính sách được tạo, chúng được áp dụng vào các bảng hoặc view tương ứng trong schema.

```
BEGIN
    dbms_rls.add_policy(
        OBJECT_SCHEMA => 'admin',
        OBJECT_NAME => 'EMPHOLIDAY',
        POLICY_NAME => 'pc1',
        FUNCTION_SCHEMA => 'admin',
        POLICY_FUNCTION => 'PC1_FUNCTION',
        STATEMENT_TYPES => 'SELECT'
    );
END;
```

- Kiểm Tra các Chính Sách: Đảm bảo rằng các chính sách hoạt động như mong đợi bằng cách kiểm tra chúng với các vai trò người dùng và ngữ cảnh khác nhau.

Nếu không có tên OBJECT\_SCHEMA thì PL/SQL block sẽ thêm chính sách cho bảng 'EMPHOLIDAY' trong schema 'admin'.

```
BEGIN
    dbms_rls.add_policy(
        OBJECT_SCHEMA => '',
        OBJECT_NAME => 'EMPHOLIDAY',
        POLICY_NAME => 'pcl',
        FUNCTION_SCHEMA => 'admin',
        POLICY_FUNCTION => 'PC1_FUNCTION',
        STATEMENT_TYPES => 'SELECT'
    );
END;
```

## II. Application Context

Application Context mang đến những lợi ích trong việc kiểm soát quyền truy cập của người dùng vào dữ liệu. Cung cấp một nơi lưu trữ bảo mật cho những giá trị ngữ cảnh ứng dụng. Nâng cao hiệu quả sử dụng Fine-grained Access Control.

Là một cơ chế cho phép bạn lưu trữ và chia sẻ dữ liệu tùy chỉnh giữa các phiên làm việc hoặc truy vấn khác nhau. Nó là một cách để truyền dữ liệu từ một phần của ứng dụng Oracle đến một phần khác mà không cần phải truyền qua các tham số hoặc biến trên một cấp độ cao hơn như các câu lệnh SQL hoặc các biến liên kết.

Là một tập các cặp name - value được lưu trữ trong bộ nhớ. Oracle Database lưu trữ giá trị application context trong một bộ đệm dữ liệu an toàn. Bộ đệm này có sẵn trong User Global Area (UGA) hoặc System (thường được gọi là “Shared”) Global Area (SGA). Bằng cách này, các giá trị application context được truy xuất trong suốt phiên. Vì application context lưu trữ các giá trị trong bộ đệm dữ liệu này nên nó sẽ tăng hiệu suất cho ứng dụng. Người dùng có thể sử dụng application context riêng lẻ, với VPD policies hoặc với fine-grained access control policies khác.

Một ứng dụng có thể sử dụng Application Context để truy cập vào thông tin phiên của một người dùng, chẳng hạn như ID người dùng hay các thông tin chi tiết khác, sau đó dữ liệu này được chuyển tới cơ sở dữ liệu một cách an toàn. Chúng ta có thể sử dụng thông tin này để cho phép hoặc ngăn người dùng truy cập dữ liệu thông qua ứng dụng.

### 1. Application Context features and setup

Tính Năng của Application Context:

- Quản lý Thông Tin Phiên: Application Context cho phép lưu trữ thông tin về phiên làm

việc hiện tại của người dùng. Điều này bao gồm các thông tin như ID người dùng, ngôn ngữ, múi giờ, và các thiết lập phiên khác. Khi thông tin này được lưu trữ trong Context, nó có thể được truy cập một cách dễ dàng từ bất kỳ đâu trong cơ sở dữ liệu, giúp ứng dụng hoạt động hiệu quả hơn với các thông tin phiên đó.

- **Bảo Mật và Quyền Truy Cập:** Một trong những chức năng quan trọng nhất của Application Context là trong việc quản lý bảo mật. Thông tin như vai trò của người dùng, quyền truy cập vào dữ liệu, hoặc các hạn chế khác có thể được lưu trữ trong context và sử dụng để kiểm soát quyền truy cập vào dữ liệu. Điều này giúp bảo vệ dữ liệu khỏi truy cập trái phép hoặc không ủng hộ.
- **Tích Hợp Ứng Dụng:** Application Context cung cấp một cách tiện lợi để truy cập và chia sẻ thông tin giữa các thành phần khác nhau của ứng dụng. Thông tin như thông tin phiên, cài đặt người dùng, hoặc các thông tin tùy chỉnh khác có thể được sử dụng trong các truy vấn SQL hoặc trong việc xử lý dữ liệu của ứng dụng.
- **Audit Logging:** Thông tin được lưu trữ trong Application Context cũng có thể được sử dụng để ghi lại các hoạt động của người dùng trong hệ thống. Bằng cách này, nó giúp trong việc theo dõi và kiểm tra tính toàn vẹn của dữ liệu cũng như hành vi của người dùng.

Setup:

Kiểm tra các application context đang có bằng câu lệnh.

```
SELECT OBJECT_NAME FROM DBA_OBJECTS WHERE OBJECT_TYPE = 'CONTEXT';
```

Tránh đặt tên trùng với context đã có.



	OBJECT_NAME
1	LSBY_APPLY_CONTEXT
2	GLOBAL_AQCLNTDB_CTX
3	DBFS_CONTEXT
4	REGISTRY\$CTX
5	SHARD_CTX
6	SHARD_CTX2
7	LT_CTX
8	DR\$APPCTX
9	SDO_SEM_HTTP_CTX
10	SDO_SEM_CTX
11	SDO_SEM_CTX_SESSION
12	SDO_SEM_UPDATE_CTX
13	OPG_CTX
14	OPG_CTX_SESSION
15	LBAC_CTX
16	LBAC\$LABELS
17	ORA_OLS_SESSION_LABELS
18	MAC\$FACTOR
19	CONTEXT_NAME
20	EMP_CONTEXT

Tạo application context với câu sql sau:

```
CREATE CONTEXT context_name USING package_name;
```

Ví dụ:

```
CREATE CONTEXT emp_context USING emp_context_pack;
```

## 2. Session-based / Local / Global Application Context

Có 3 loại application context. Hình sau sẽ tóm tắt lại sự khác nhau giữa các loại application contexts:

Application Context Type	Stored in UGA	Stored in SGA	Supports Connected User Database Links	Supports Centralized Storage of Users' Application Context	Supports Sessionless Multitier Applications
Database session-based application context initialized locally	Yes	No	No	No	No
Database session-based application context initialized externally	Yes	No	Yes	No	No
Database session-based application context initialized globally	Yes	No	No	Yes	No
Global application context	No	Yes	No	No	Yes
Client session-based application context	Yes	No	Yes	No	Yes

### 2.1. Session-based application contexts

- Loại này truy xuất dữ liệu được lưu trữ trong cơ sở dữ liệu của bộ nhớ đệm trong phiên người dùng. Có 2 loại session-based application context:
  - Loại được khởi tạo bên ngoài: Khởi tạo application context từ một ứng dụng Oracle Call Interface (OCI) hay một quy trình job queue hoặc một đường dẫn cơ sở dữ liệu của người dùng đã được kết nối.

Cách khởi tạo này có thể gia tăng hiệu suất bởi vì application context được lưu trữ trong khu vực toàn cục của người dùng (UGA).

Bạn phải sử dụng một loại namespace đặc biệt để khởi tạo Application Context theo phiên từ bên ngoài.

Một application context có thể ghi nhận sự khởi tạo các thuộc tính và giá trị từ tài nguyên bên ngoài. Nó bao gồm một Oracle Call Interface (OCI), một quy trình job queue, hoặc một đường dẫn cơ sở dữ liệu.

Mặc dù bất cứ chương trình phía máy khách nào mà sử dụng Oracle Call Interface đều có thể khởi tạo loại namespace này, bạn có thể sử dụng triggers đăng nhập để xác minh các giá trị.

1. Câu lệnh SQL CREATE CONTEXT có thể khởi tạo 1 cơ sở dữ liệu session-based application context từ bên ngoài:

```
CREATE CONTEXT ext_ctx USING ext_ctx_pkg INITIALIZED EXTERNALLY;
```



The **CREATE CONTEXT** SQL statement can create an externalized database session-based application context.

## 2. Khởi tạo giá trị Application Context từ một Server cấp trung.

Server cấp trung có thể khởi tạo các giá trị application context thay cho các người dùng cơ sở dữ liệu. Trong quá trình này, các thuộc tính context được truyền cho phiên từ xa tại thời điểm khởi tạo và cơ sở dữ liệu từ xa chấp nhận các giá trị nếu không gian tên được khởi tạo bên ngoài.

Ví dụ: có một ứng dụng ba tầng tạo phiên người dùng thông qua OCI hoặc JDBC/OCI có thể truy cập thuộc tính **PROXY\_USER** trong **USERENV**. Thuộc tính này cho phép bạn xác định xem phiên người dùng có được tạo bởi ứng dụng cấp trung hay không. Bạn có thể cho phép người dùng chỉ truy cập dữ liệu đối với các kết nối mà người dùng được ủy quyền. Nếu người dùng kết nối trực tiếp với cơ sở dữ liệu thì họ sẽ không thể truy cập bất kỳ dữ liệu nào. Bạn có thể sử dụng thuộc tính **PROXY\_USER** từ namespace **USERENV** trong **VPD** của Oracle để đảm bảo rằng người dùng chỉ truy cập dữ liệu thông qua một ứng dụng cấp trung cụ thể.

- Loại được khởi tạo toàn cục: Khi một cơ sở dữ liệu session-based application sử dụng các thuộc tính và các giá trị từ một vị trí tập trung, nó có thể được sử dụng toàn cục chẳng hạn như từ LDAP directory.

Để sử dụng ứng dụng ứng dụng bảo mật được khởi tạo toàn cục, bạn trước hết phải cấu hình Enterprise User Security. Sau đó, bạn phải cấu hình các giá trị application context cho người dùng trong cơ sở dữ liệu và thư mục.

Khi một người dùng toàn cục (enterprise user) kết nối tới cơ sở dữ liệu, Enterprise User Security sẽ xác minh danh tính của người dùng đang kết nối với cơ sở dữ liệu. Sau khi xác thực, vai trò người dùng chung và application context sẽ được truy xuất từ thư mục. Khi người dùng đăng nhập vào cơ sở dữ liệu, các vai trò toàn cục và application context ban đầu đã được cài đặt.

Bạn có thể định cấu hình và lưu trữ ngữ cảnh ứng dụng ban đầu cho người dùng, chẳng hạn như department name hay title trong thư mục LDAP.

Cách khởi tạo:

### 1. Tạo application context trong cơ sở dữ liệu:

```
CREATE CONTEXT hr USING hrapps.hr_manage_pkg INITIALIZED GLOBALLY;
```



## 2. Tạo và thêm các mục mới trong thư mục LDAP:

Sau đây là ví dụ về các mục được thêm vào thư mục LDAP. Các mục này tạo một thuộc tính có tên Title với giá trị thuộc tính Manager cho ứng dụng HR và gán tên người dùng user1 và user2. Trong phần sau đây, cn=example để cập tên của domain:

```
dn: cn=OracleDBAppContext,cn=example,cn=OracleDBSecurity,cn=Products,cn=OracleContext,ou=Americas,o=oracle,c=US
changetype: add
cn: OracleDBAppContext
objectclass: top
objectclass: orclContainer

dn: cn=hr,cn=OracleDBAppContext,cn=example,cn=OracleDBSecurity,cn=Products,cn=OracleContext,ou=Americas,o=oracle,c=US
changetype: add
cn: hr
objectclass: top
objectclass: orclContainer

dn: cn=Title,cn=hr, cn=OracleDBAppContext,cn=example,cn=OracleDBSecurity,cn=Products,cn=OracleContext,ou=Americas,o=oracle,c=US
changetype: add
cn: Title
objectclass: top
objectclass: orclContainer

dn: cn=Manager,cn=Title,cn=hr, cn=OracleDBAppContext,cn=example,cn=OracleDBSecurity,cn=Products,cn=OracleContext,ou=Americas,o=oracle,c=US
changetype: add
cn: Manager
objectclass: top
objectclass: groupofuniquenames
objectclass: orclDBApplicationContext
uniquemember: CN=user1,OU=Americas,O=Oracle,L=Redwoodshores,ST=CA,C=US
uniquemember: CN=user2,OU=Americas,O=Oracle,L=Redwoodshores,ST=CA,C=US
```

Nếu người dùng tồn tại một đối tượng LDAP inetOrgPerson, thì sau đó kết nối sẽ truy xuất các thuộc tính từ inetOrgPerson và gán chúng cho namespace SYS\_LDAP\_USER\_DEFAULT. Lưu ý rằng ngữ cảnh chỉ được điền với các giá trị không phải NULL là một phần của lớp đối tượng inetOrgPerson.

Một ví dụ về inetOrgPerson entry:

```
dn: cn=user1,ou=Americas,O=oracle,L=redwoodshores,ST=CA,C=US
changetype: add
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: user1
sn: One
givenName: User
initials: UO
title: manager, product development
uid: uone
mail: uone@us.example.com
telephoneNumber: +1 650 555 0105
employeeNumber: 00001
employeeType: full time
```

 Copy

Kết nối tới cơ sở dữ liệu:

Khi user1 kết nối với cơ sở dữ liệu thuộc example domain, user1 sẽ có Title được đặt thành Manager. Mọi thông tin liên quan đến user1 sẽ được truy xuất từ thư mục LDAP. Giá trị có thể được lấy bằng cú pháp sau:

```
SYS_CONTEXT('namespace','attribute name')
```

 Copy

Ví dụ:

```
DECLARE
  tmpstr1 VARCHAR2(30);
  tmpstr2 VARCHAR2(30);
BEGIN
  tmpstr1 = SYS_CONTEXT('HR','TITLE');
  tmpstr2 = SYS_CONTEXT('SYS_LDAP_USER_DEFAULT','telephoneNumber');
  DBMS_OUTPUT.PUT_LINE('Title is ' || tmpstr1);
  DBMS_OUTPUT.PUT_LINE('Telephone Number is ' || tmpstr2);
END;
```

 Copy

## 2.2. Local Application context:

- Khởi tạo cục bộ application context, theo phiên của người dùng. Cách tạo và sử dụng:

**Bước 1:** Tạo tài khoản người dùng và đảm bảo người dùng 'SCOTT' đang hoạt động.

1. Đăng nhập với tư cách người dùng 'SYS' và kết nối bằng quyền quản trị SYSDBA.

```
sqlplus sys as sysdba
Enter password: password
```

 Copy

Enter password: *password*

2. Trong môi trường ứng dụng nhiều người dùng, kết nối tới PDB thích hợp:

```
CONNECT SYS@hrpdb AS SYSDBA
Enter password: password
```

 Copy

Enter password: *password*

Để tìm PDBs có sẵn, chạy lệnh `show pdbs`. Để kiểm tra PDB hiện tại, chạy lệnh `show`.

3. Tạo tài khoản người dùng cục bộ `sysadmin_ctx` để quản trị cơ sở dữ liệu session-based application context:

```
CREATE USER sysadmin_ctx IDENTIFIED BY
password;

GRANT CREATE SESSION, CREATE ANY CONTEXT,
CREATE PROCEDURE, CREATE TRIGGER, ADMINISTER
DATABASE TRIGGER TO sysadmin_ctx;

GRANT READ ON HR.EMPLOYEES TO sysadmin_ctx;

GRANT EXECUTE ON DBMS_SESSION TO
sysadmin_ctx;
```

4. Tạo tài khoản người dùng sau cho Lisa Ozer, người có lozer trong email trong bảng HR.EMPLOYEES:

```
GRANT CREATE SESSION TO LOZER IDENTIFIED BY password;
```

 Copy

5. Người dùng SCOTT sẽ được sử dụng trong hướng dẫn này, nên sẽ phải truy vấn vào dữ liệu bảng DBA\_USERS để đảm bảo user SCOTT được mở:

```
SELECT USERNAME, ACCOUNT_STATUS FROM DBA_USERS WHERE USERNAME = 'SCOTT';
```

 Copy

Nếu user SCOTT trong bảng DBA\_USERS bị khoá hoặc hết hạn, thì sử dụng câu lệnh sau để mở khoá và tạo mật khẩu:

```
ALTER USER SCOTT ACCOUNT UNLOCK IDENTIFIED BY password;
```

 Copy

## **Bước 2:** Tạo cơ sở dữ liệu Session-based application contexts

Với tư cách là người dùng sysadmin\_ctx, ta có thể tạo được cơ sở dữ liệu Session-based application contexts.

1. Đăng nhập bằng quyền sysadmin\_ctx:

```
CONNECT sysadmin_ctx -- Or, CONNECT sysadmin_ctx@hrpdb  
Enter password: password
```

## 2. Tạo application context bằng cách sử dụng câu lệnh:

```
CREATE CONTEXT empno_ctx USING set_empno_ctx_pkg;
```

Bước 3: Tạo một Package để truy xuất dữ liệu theo phiên và đặt lại Application Context

Tiếp theo, chúng ta tạo một gói PL/SQL để truy xuất dữ liệu theo phiên và đặt lại application context.

```
CREATE OR REPLACE PACKAGE set_empno_ctx_pkg IS  
    PROCEDURE set_empno;  
END;  
/  
CREATE OR REPLACE PACKAGE BODY set_empno_ctx_pkg IS  
    PROCEDURE set_empno  
    IS  
        emp_id HR.EMPLOYEES.EMPLOYEE_ID%TYPE;  
    BEGIN  
        SELECT EMPLOYEE_ID INTO emp_id FROM HR.EMPLOYEES  
            WHERE email = SYS_CONTEXT('USERENV', 'SESSION_USER');  
        DBMS_SESSION.SET_CONTEXT('empno_ctx', 'employee_id', emp_id);  
    EXCEPTION  
        WHEN NO_DATA_FOUND THEN NULL;  
    END;  
END;  
/
```

Package trên tạo một thủ tục gọi là set\_empno thực hiện các hành động sau:

- emp\_id HR.EMPLOYEES.EMPLOYEE\_ID%TYPE khai báo một biến emp\_id để lưu ID nhân viên của người dùng thực hiện đăng nhập. Biến được khai báo sử dụng cùng một loại dữ liệu với cột EMPLOYEE\_ID trong bảng HR.EMPLOYEES.
- SELECT EMPLOYEE\_ID INTO emp\_id FROM HR.EMPLOYEES thực hiện một câu lệnh SELECT để sao chép ID nhân viên được lưu trữ trong cột dữ liệu employee\_id từ bảng HR.EMPLOYEES vào biến emp\_id.
- WHERE email = SYS\_CONTEXT('USERENV', 'SESSION\_USER') sử dụng mệnh đề WHERE để tìm tất cả ID nhân viên khớp với tài khoản email của người dùng trong phiên. Hàm SYS\_CONTEXT sử dụng USERENV context được định nghĩa trước để

truy xuất ID người dùng trong phiên, có giá trị giống với dữ liệu từ cột email. Chẳng hạn như ID người dùng và địa chỉ email của Lisa Ozer có giá trị giống nhau là: lozer.

- DBMS\_SESSION.SET\_CONTEXT('empno\_ctx', 'employee\_id', 'emp\_id') sử dụng thủ tục DBMS\_SESSION.SET\_CONTEXT để tinh chỉnh application context:

‘empno\_ctx’: Gọi application context empno\_ctx.

‘employee\_id’: Tạo giá trị thuộc tính cho cặp name-value của empno\_ctx application context, bằng cách đặt tên nó là member\_id.

‘emp\_id’: Đặt giá trị cho thuộc tính employee\_id thành giá trị được lưu trong biến emp\_id.

- EXCEPTION ... WHEN\_NO\_DATA\_FOUND thêm một exception hệ thống WHEN NO\_DATA\_FOUND để catch bất kỳ lỗi no data found nào có thể xảy ra từ câu lệnh SELECT. Nếu không có exception này, package vừa được thiết lập và trigger đăng nhập vẫn sẽ hoạt động tốt và thiết lập application context nếu cần, nhưng khi đó bất kỳ người dùng không phải quản trị viên hệ thống nào ngoài những người dùng được liệt kê trong bảng HR.EMPLOYEES sẽ không thể đăng nhập vào cơ sở dữ liệu. Nhưng giả sử một số là người dùng cơ sở dữ liệu hợp lệ, họ vẫn có thể đăng nhập vào hệ thống. Khi thông tin application context được thiết lập, bạn có thể sử dụng thông tin phiên này như một cách để kiểm soát quyền truy cập của người dùng vào một ứng dụng cụ thể.

Bước 4: Tạo một Logon Trigger cho Package:

Logon Trigger sẽ được kích hoạt khi người dùng đăng nhập.

Với tư cách là người dùng sysadmin\_ctx, ta sẽ tạo một trigger đăng nhập cho thủ tục package set\_empno\_ctx\_pkg.set\_empno:

```
CREATE TRIGGER set_empno_ctx_trig AFTER LOGON ON DATABASE
BEGIN
    sysadmin_ctx.set_empno_ctx_pkg.set_empno;
END;
/
```

Bước 5: Test Application Context:

Đến bước này tất cả các thành phần đã được hoàn chỉnh, đây là lúc để chúng ta thực hiện test Application Context.

1. Đăng nhập với người dùng lozer:

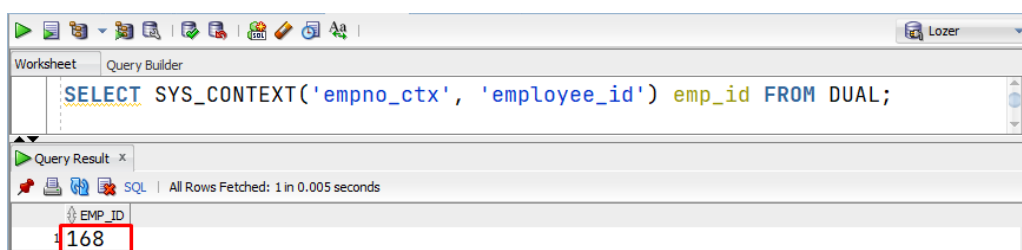


```
CONNECT lozer -- Or, CONNECT lozer@hrpdb
Enter password: password
```

Khi người dùng lozer đăng nhập, application context empno\_ctx sẽ thu thập ID nhân viên của người đó. Chúng ta có thể kiểm tra lại bằng cách:

```
SELECT SYS_CONTEXT('empno_ctx', 'employee_id') emp_id FROM DUAL;
```

Kết quả sẽ nhận được:



The screenshot shows the SQL Developer interface with the user 'Lozer' selected. The query 'SELECT SYS\_CONTEXT('empno\_ctx', 'employee\_id') emp\_id FROM DUAL;' is entered in the Query Builder. The Query Result pane shows 'All Rows Fetched: 1 in 0.005 seconds'. The result table has one column 'EMP\_ID' and one row with the value '168', which is highlighted with a red box.

EMP_ID
168

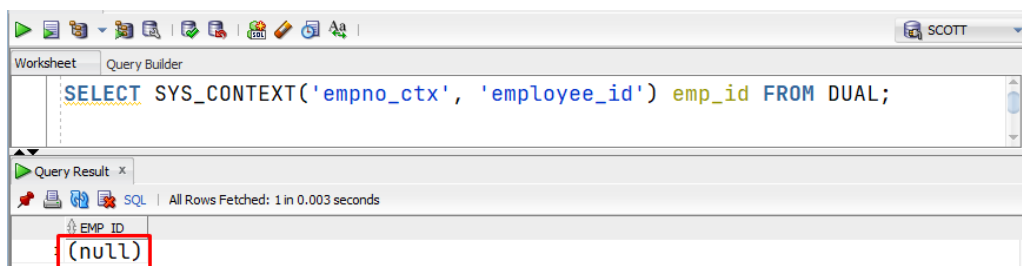
## 2. Đăng nhập với người dùng SCOTT

```
CONNECT SCOTT -- Or, CONNECT SCOTT@hrpdb
Enter password: password
```

Vì người dùng SCOTT không nằm trong danh sách nhân viên trong bảng HR.EMPLOYEES, nên application context empno\_ctx không thể xác định được ID nhân viên của người dùng này.

```
CONNECT SCOTT -- Or, CONNECT SCOTT@hrpdb
Enter password: password
```

Kết quả sẽ nhận được:



The screenshot shows the SQL Developer interface with the user 'SCOTT' selected. The same query is entered. The Query Result pane shows 'All Rows Fetched: 1 in 0.003 seconds'. The result table has one column 'EMP\_ID' and one row with the value '(null)', which is highlighted with a red box.

EMP_ID
(null)

Từ lúc này, ứng dụng có thể sử dụng thông tin phiên của người dùng để xác định mức độ truy cập của người dùng có thể có tới cơ sở dữ liệu. Chúng ta có thể sử dụng Oracle VPD để thực hiện việc này.

Bước 6: Xóa các thành phần của hướng dẫn này:

Khi không cần sử dụng các thành phần trong hướng dẫn này, bạn có thể xóa chúng.

1. Kết nối bằng vai trò SYS với đặc quyền SYSDBA

```
CONNECT SYS AS SYSDBA -- Or, CONNECT SYS@hrpdb AS SYSDBA  
Enter password: password
```

2. Drop người dùng sysadmin\_ctx và lozer:

```
DROP USER sysadmin_ctx CASCADE;  
DROP USER lozer;
```

3. Drop application context:

```
DROP CONTEXT empno_ctx;
```

Mặc dù sysadmin\_ctx tạo ra application context, nhưng nó vẫn được sở hữu bởi SYS schema.

4. Nếu muốn, chúng ta có thể lock hoặc expire SCOTT trừ khi có người khác muốn sử dụng tài khoản này:

```
ALTER USER SCOTT PASSWORD EXPIRE ACCOUNT LOCK;
```

### 2.3. Global Application Context:

- Chúng ta có thể sử dụng Global Application Context để truy cập giá trị ứng dụng thông qua phiên cơ sở dữ liệu, bao gồm một môi trường Oracle Real Application Clusters.

- Có 3 cách để sử dụng cho Global Application Context:
  - Bạn phải chia sẻ các giá trị ứng dụng trên toàn cục cho tất cả người dùng cơ sở dữ liệu. Bạn có thể cần phải vô hiệu hóa quyền truy cập vào một ứng dụng dựa trên một tình huống cụ thể. Trong trường hợp này, các giá trị mà application context thiết lập không dành riêng cho người dùng và cũng không dựa trên dữ liệu riêng tư của người dùng. Application Context xác định một tình huống, chẳng hạn như để chỉ ra phiên bản của mô-đun ứng dụng đang chạy.
  - Bạn có người dùng cơ sở dữ liệu phải chuyển từ ứng dụng này sang ứng dụng khác. Trong trường hợp này, ứng dụng thứ hai mà người dùng đang chuyển sang có các yêu cầu truy cập khác với ứng dụng đầu tiên.
  - Bạn phải xác thực những người dùng không có cơ sở dữ liệu, tức là những người dùng chưa biết đến cơ sở dữ liệu.
- Các thành phần của Global Application Context:
  - Global Application Context: sử dụng câu lệnh SQL CREATE CONTEXT để tạo global application context và bao gồm mệnh đề ACCESSED GLOBALLY trong câu lệnh. Câu lệnh này đặt tên cho application context và liên kết nó với một thủ tục PL/SQL được thiết kế để thiết lập dữ liệu cho application context. Global application context được khởi tạo và lưu trữ trong lược đồ cơ sở dữ liệu của quản trị viên bảo mật, người tạo ra nó.
  - Một PL/SQL package để thiết lập các thuộc tính: Package này phải chứa một thủ tục sử dụng thủ tục DBMS\_SESSION.SET\_CONTEXT để thiết lập application context.
  - Một ứng dụng middle-tier để nhận và đặt ID phiên máy khách. Đối với những người dùng không có cơ sở dữ liệu yêu cầu xác thực ID phiên khách, bạn có thể sử dụng lệnh gọi Oracle Call Interface (OCI) trong ứng dụng middle-tier để truy xuất và đặt dữ liệu phiên của họ.

- Tạo một Global Application Context:

Câu lệnh SQL CREATE CONTEXT để tạo application context toàn cục, và được lưu trữ trong SYS schema. Mặc dù bất cứ người dùng nào được cấp quyền CREATE ANY CONTEXT và DROP ANY CONTEXT đều có thể tạo và drop application context toàn cục nhưng nó vẫn thuộc sở hữu của SYS schema.

Giống như application context cục bộ, application context toàn cục được tạo và lưu trữ trong lược đồ cơ sở dữ liệu của quản trị viên bảo mật. Bạn phải có quyền hệ thống

CREATE ANY CONTEXT trước khi tạo được một application context toàn cục, và quyền DROP ANY CONTEXT trước khi bạn có thể drop được context với câu lệnh DROP CONTEXT. Cơ sở dữ liệu Oracle liên kết context với tài khoản lược đồ đã tạo ra nó, nhưng nếu bạn loại bỏ người dùng này, context vẫn tồn tại trong lược đồ SYS. Bằng cách đăng nhập với người dùng SYS, bạn có thể drop application context.

```
CREATE OR REPLACE CONTEXT global_hr_ctx USING  
hr_ctx_pkg ACCESSED GLOBALLY;
```

- Tạo một PL/SQL package để quản lý Application Context:

Về Package quản lý Global Application Context:

- Vai trò của package PL/SQL mà bạn liên kết với global application context là sử dụng gói DBMS\_SESSION để thiết lập và xóa các giá trị global application context.
- Bạn phải có đặc quyền EXECUTE cho gói DBMS\_SESSION trước khi sử dụng các thủ tục của nó. Gói này thuộc quyền sở hữu của SYS schema.
- Không giống như các gói PL/SQL được sử dụng để thiết lập local application context, bạn không bao gồm hàm SYS\_CONTEXT để lấy dữ liệu phiên của người dùng. Bạn không cần thêm chức năng này vì chủ sở hữu phiên được ghi trong USERENV, giống nhau đối với mọi người dùng đang kết nối.

Chia sẻ giá trị Global Application Context cho tất cả người dùng cơ sở dữ liệu: tất cả người dùng có tài khoản cơ sở dữ liệu sẽ có khả năng truy cập vào dữ liệu trong cơ sở dữ liệu.

```
dn: cn=user1,ou=Americas,O=oracle,L=redwoodshores,ST=CA,C=US  
changetype: add  
objectClass: top  
objectClass: person  
objectClass: organizationalPerson  
objectClass: inetOrgPerson  
cn: user1  
sn: One  
givenName: User  
initials: UO  
title: manager, product development  
uid: uone  
mail: uone@us.example.com  
telephoneNumber: +1 650 555 0105  
employeeNumber: 00001  
employeeType: full time
```

 Copy

- Sử dụng thủ tục DBMS\_SESSION.SET\_CONTEXT để đặt giá trị cho các tham số namespace, thuộc tính và giá trị. Giá trị sec\_level được chỉ định khi ứng dụng cơ

sở dữ liệu chạy thủ tục hr\_ctx\_pkg.set\_hr\_ctx. Giá trị tên người dùng và client\_id không được đặt cách này cho phép tất cả người dùng (người dùng cơ sở dữ liệu) có quyền truy cập vào các giá trị phù hợp với cài đặt trên toàn máy chủ.

Thiết lập Global Context cho người dùng cơ sở dữ liệu di chuyển qua lại giữa các ứng dụng:

- Để thiết lập global application context cho người dùng cơ sở dữ liệu di chuyển qua lại giữa các ứng dụng, đặc biệt khi các ứng dụng có yêu cầu truy cập khác nhau, cần sử dụng tham số username trong quy trình SET\_CONTEXT. Tham số này chỉ định rằng cùng một lược đồ sẽ được sử dụng cho tất cả các phiên.

```
88 -- Package to Manage Global Application Context Values
89 -- for a User Moving Between Applications
90 CREATE OR REPLACE PACKAGE hr_ctx_pkg
91 AS
92     PROCEDURE set_hr_ctx(sec_level IN VARCHAR2, user_name IN VARCHAR2);
93     PROCEDURE clear_hr_context;
94 END;
95
96 CREATE OR REPLACE PACKAGE BODY hr_ctx_pkg
97 AS
98     PROCEDURE set_hr_ctx(sec_level IN VARCHAR2, user_name IN VARCHAR2)
99     AS
100     BEGIN
101         DBMS_SESSION.SET_CONTEXT(
102             namespace => 'global_hr_ctx',
103             attribute  => 'job_role',
104             value       => sec_level,
105             username    => user_name);
106     END set_hr_ctx;
107
108     PROCEDURE clear_hr_context
109     AS
110     BEGIN
111         DBMS_SESSION.CLEAR_CONTEXT('global_hr_ctx');
112     END clear_hr_context;
113 END;
114
115 BEGIN
116     hr_ctx_pkg.set_hr_ctx('clerk', 'scott');
117 END;
```

Thiết lập Global Context cho người dùng không tồn tại trong cơ sở dữ liệu:

- Khi một người dùng không có cơ sở dữ liệu, hay là một người dùng không có liên kết với cơ sở dữ liệu (chẳng hạn như người dùng ứng dụng Web), khi bắt đầu một phiên máy khách, máy chủ ứng dụng sẽ tạo ID phiên máy khách. Khi ID này được thiết lập trên máy chủ ứng dụng, nó phải được chuyển đến phía máy chủ cơ sở dữ liệu. Sử dụng thủ tục DBMS\_SESSION.SET\_IDENTIFIER để đặt ID phiên máy khách. Để đặt ngữ cảnh, cần đặt tham số client\_id trong thủ tục DBMS\_SESSION.SET\_CONTEXT ở phía máy chủ. Điều này cho phép quản lý Global Application Context, tuy nhiên mỗi khách hàng chỉ nhìn thấy bối cảnh ứng dụng được chỉ định của mình.

```

122 -- Package to Manage Global Application Context Values for Nondatabase Users
123 CREATE OR REPLACE PACKAGE hr_ctx_pkg
124 AS
125     PROCEDURE set_session_id(session_id_p IN NUMBER);
126     PROCEDURE set_hr_ctx(sec_level_attr IN VARCHAR2,
127         sec_level_val IN VARCHAR2);
128     PROCEDURE clear_hr_session(session_id_p IN NUMBER);
129     PROCEDURE clear_hr_context;
130 END;
131 CREATE OR REPLACE PACKAGE BODY hr_ctx_pkg
132 AS
133     session_id_global NUMBER;
134     PROCEDURE set_session_id(session_id_p IN NUMBER)
135     AS
136     BEGIN
137         session_id_global := session_id_p;
138         DBMS_SESSION.SET_IDENTIFIER(session_id_p);
139     END set_session_id;
140     PROCEDURE set_hr_ctx(sec_level_attr IN VARCHAR2,
141         sec_level_val IN VARCHAR2)
142     AS
143     BEGIN
144         DBMS_SESSION.SET_CONTEXT(
145             namespace => 'global_hr_ctx',
146             attribute => sec_level_attr,
147             value      => sec_level_val,
148             username   => USER,
149             client_id  => session_id_global);
150     END set_hr_ctx;
151     PROCEDURE clear_hr_session(session_id_p IN NUMBER)
152     AS
153     BEGIN
154         DBMS_SESSION.SET_IDENTIFIER(session_id_p);
155         DBMS_SESSION.CLEAR_IDENTIFIER;
156     END clear_hr_session;
157     PROCEDURE clear_hr_context
158     AS
159     BEGIN
160         DBMS_SESSION.CLEAR_CONTEXT('global_hr_ctx', session_id_global);
161     END clear_hr_context;
162 END;

```

## 2.4. Client Session-Based Application Context

Trong Client Session-Based Application Contexts, cần sử dụng các hàm Oracle Call Interface (OCI) để thiết lập và xóa thông tin phiên người dùng, sau đó được lưu trữ trong User Global Area (UGA). Ưu điểm của loại application context này là một ứng dụng riêng lẻ có thể kiểm tra dữ liệu phiên của người dùng không có cơ sở dữ liệu cụ thể, thay vì yêu cầu cơ sở dữ liệu thực hiện nhiệm vụ này.

Để cấu hình client session-based application context, cần sử dụng hàm OCIAppCtxSet OCI. Một client session-based application context sử dụng namespace CLIENTCONTEXT, có thể cập nhật bởi bất kỳ máy khách OCI nào hoặc bởi gói DBMS\_SESSION hiện có cho application context. Cơ sở dữ liệu Oracle không thực hiện kiểm tra bảo mật gói hoặc đặc quyền cho loại này. Bất kỳ người dùng nào cũng có thể đặt, xóa hoặc thu thập thông tin trong namespace CLIENTCONTEXT vì nó không được bảo vệ bởi bảo mật dựa trên gói.

- Thiết lập giá trị trong CLIENTCONTEXT namespace:

Sử dụng cú pháp sau đối với OCI:

```
err = OCIAppCtxSet((void *) session_handle, (dvoid *) "CLIENTCONTEXT", (ub4) 13,
                  (dvoid *) attribute_name, length_of_attribute_name
                  (dvoid *) attribute_value, length_of_attribute_value, errhp,
                  OCI_DEFAULT);
```

- session\_handle: Đại diện cho không gian tên OCISessionHandle.
- attribute\_name: Tên của thuộc tính. Ví dụ, trách nhiệm, có độ dài là 14.
- attribute\_value: Giá trị của thuộc tính. Ví dụ, quản lý, có độ dài là 7.

- Truy xuất CLIENTCONTEXT namespace:

Sử dụng Oracle Call Interface OCIStmtExecute:

```
SELECT SYS_CONTEXT('CLIENTCONTEXT', 'Attribute-1') FROM dual;

SELECT VALUE FROM SESSION_CONTEXT

WHERE NAMESPACE='CLIENTCONTEXT' AND ATTRIBUTE='attribute-1';
```

### 3. How to use Application Context for Fine-grained Access Control

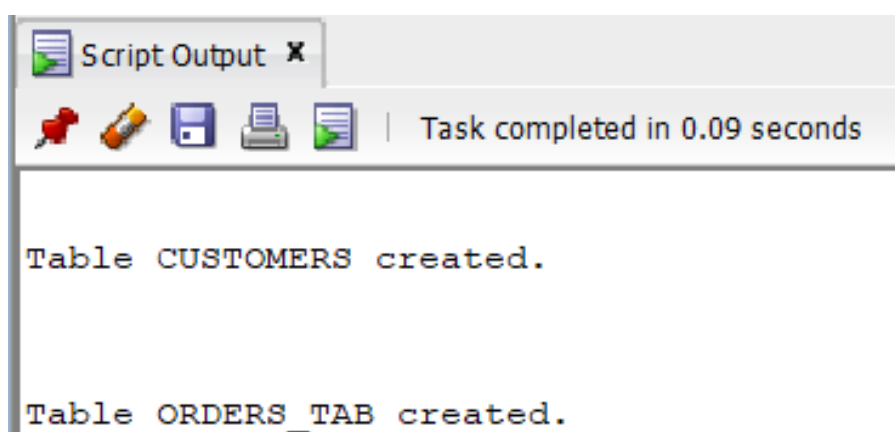
#### 3.1. Thực hiện chính sách:

- Quy trình trong ví dụ này giả định mối quan hệ một-một giữa người dùng và khách hàng. Nó tìm số khách hàng của người dùng (), và lưu vào bộ nhớ cache số khách hàng trong ngữ cảnh ứng dụng. Sau đó, bạn có thể tham khảo thuộc tính của context () nhập lệnh bên trong hàm security policy. *Cust\_numcust\_numoe\_ctx*

**Bước 1.** Tạo một gói PL / SQL đặt ngữ cảnh cho ứng dụng

Bạn có thể cần thiết lập cấu trúc dữ liệu sau để các ví dụ sau hoạt động:

```
CREATE TABLE apps.customers (cust_no NUMBER(4), cust_name  
VARCHAR2(20));  
CREATE TABLE scott.orders_tab (order_no NUMBER(4));
```



Tạo package như sau:



```

8
9 CREATE OR REPLACE PACKAGE apps.oe_ctx AS
10     PROCEDURE set_cust_num;
11 END;
12
13 CREATE OR REPLACE PACKAGE BODY apps.oe_ctx AS
14     PROCEDURE set_cust_num IS
15         custnum NUMBER;
16     BEGIN
17         SELECT cust_no INTO custnum FROM customers WHERE cust_name =
18             SYS_CONTEXT('USERENV', 'SESSION_USER');
19         /* SET cust_num attribute in 'order_entry' context */
20         DBMS_SESSION.SET_CONTEXT('order_entry', 'cust_num', custnum);
21         DBMS_SESSION.SET_CONTEXT('order_entry', 'cust_num', custnum);
22     END set_cust_num;
23 END;
24
25

```

Script Output x

Task completed in 0.043 seconds

Package Body OE\_CTX compiled

## Bước 2. Tạo ngữ cảnh ứng dụng

Tạo ngữ cảnh ứng dụng bằng cách nhập:

```

27 CREATE CONTEXT Order_entry USING apps.oe_ctx;

```

Script Output x

Task completed in 0.06 seconds

Context ORDER\_ENTRY created.

Ngoài ra, bạn có thể sử dụng Oracle Policy Manager để tạo ngữ cảnh ứng dụng.

## Bước 3. Truy cập ngữ cảnh ứng dụng bên trong gói

Truy cập ngữ cảnh ứng dụng bên trong gói thực hiện chính sách bảo mật trên đối tượng cơ sở dữ liệu.

Bạn có thể cần thiết lập các cấu trúc dữ liệu sau để một số ví dụ nhất định hoạt động:

```

5
6 CREATE OR REPLACE PACKAGE Oe_security AS
7 FUNCTION Custnum_sec (D1 VARCHAR2, D2 VARCHAR2)
8 RETURN VARCHAR2;
9 END;
10

```

Script Output x

Task completed in 0.993 seconds

Package OE\_SECURITY compiled

Thân gói nối một vị ngữ động vào các câu lệnh trên bàn. Vị ngữ này giới hạn các đơn đặt hàng được trả về số khách hàng của người dùng bằng cách truy cập thuộc tính ngữ cảnh, thay vì truy vấn con vào bảng khách hàng. *SELECTORDERS\_TABcust\_num*

```

12 CREATE OR REPLACE PACKAGE BODY Oe_security AS
13 /* limits select statements based on customer number: */
14 FUNCTION Custnum_sec (D1 VARCHAR2, D2 VARCHAR2) RETURN VARCHAR2
15 IS
16     D_predicate VARCHAR2 (2000);
17 BEGIN
18     D_predicate := 'cust_no = SYS_CONTEXT(''order_entry'', ''cust_num'');'
19     RETURN D_predicate;
20 END Custnum_sec;
21 END Oe_security;
22
23

```

Script Output x

Task completed in 0.187 seconds

Package Body OE\_SECURITY compiled

#### Bước 4. Tạo chính sách bảo mật mới

Bạn có thể cần thiết lập các cấu trúc dữ liệu sau để một số ví dụ nhất định hoạt động:

```
CONNECT sys/xIcf1T9u AS sysdba;
CREATE USER secur IDENTIFIED BY secur;
```

Tạo chính sách như sau:

```
BEGIN
DBMS_RLS.ADD_POLICY ('scott', 'orders_tab', 'oe_policy', 'secur',
                    'oe_security.custnum_sec', 'select');
END;
```

Câu lệnh này thêm một chính sách có tên vào bảng để xem trong lược đồ . Các tệp . hàm thực hiện chính sách, được lưu trữ trong lược đồ và chỉ áp dụng cho các câu lệnh. *OE\_POLICYORDERS\_TABSCOTTSECUSROE\_SECURITY.CUSTNUM\_SECSECUSRSELECT*

Bây giờ, bất kỳ tuyên bố chọn nào của khách hàng trên bảng sẽ tự động trả về chỉ đơn đặt hàng của khách hàng đó. Nói cách khác, vị ngữ động sửa đổi câu lệnh của người dùng từ điều này: *ORDERS\_TAB*

```
SELECT * FROM Orders_tab
```

Thành như sau:

```
SELECT * FROM Orders_tab
WHERE Custno = SYS_CONTEXT('order_entry', 'cust_num')
```

- Lưu ý những điều sau đây liên quan đến ví dụ này:

Trong thực tế, bạn có thể có một số vị ngữ dựa trên vị trí của người dùng. Ví dụ: đại diện bán hàng sẽ chỉ có thể xem hồ sơ cho khách hàng của mình và nhân viên nhập đơn hàng sẽ có thể xem bất kỳ đơn đặt hàng nào của khách hàng. Bạn có thể mở rộng hàm để trả về các vị ngữ khác nhau dựa trên giá trị ngữ cảnh vị trí của người dùng. *custnum\_sec*

Việc sử dụng ngữ cảnh ứng dụng trong gói kiểm soát truy cập chi tiết có hiệu quả cung cấp cho bạn một biến ràng buộc trong một câu lệnh phân tích cú pháp. Chẳng hạn:

```
SELECT * FROM Orders_tab
WHERE Custno = SYS_CONTEXT('order_entry', 'cust_num')
```

Điều này được phân tích cú pháp và tối ưu hóa hoàn toàn, nhưng việc đánh giá giá trị thuộc tính của người dùng cho ngữ cảnh diễn ra khi thực thi. Điều này có nghĩa là bạn nhận được lợi ích của một câu lệnh được tối ưu hóa thực thi khác nhau cho mỗi người dùng thực thi câu lệnh. *CUST\_NUMORDER\_ENTRY*

*Bạn có thể đặt các thuộc tính ngữ cảnh của mình dựa trên dữ liệu từ bảng hoặc bảng cơ sở dữ liệu hoặc từ máy chủ thư mục bằng LDAP (Giao thức truy cập thư mục nhẹ).*

#### 4. Check the policy effect: View V \$ VPD\_POLICY

*V\$VPD\_POLICY* hiển thị tất cả các chính sách bảo mật chi tiết cho PDB hiện tại. Dạng VIEW này hữu ích cho việc tìm kiếm các chính sách được áp dụng cho câu lệnh SQL.

Cột	Kiểu dữ liệu	Mô tả
ADDRESS	RAW(4   8)	Địa chỉ con trở
PARADDR	RAW(4   8)	Địa chỉ con trở cha
SQL_HASH	NUMBER	Số băm SQL
SQL_ID	VARCHAR2(13)	Định danh SQL
CHILD_NUMBER	NUMBER	Số thứ tự con trở trong con
OBJECT_OWNER	VARCHAR2(128)	Chủ sở hữu của đối tượng có chính sách
OBJECT_NAME	VARCHAR2(128)	Tên của đối tượng có chính sách
POLICY_GROUP	VARCHAR2(128)	Tên nhóm chính sách
POLICY	VARCHAR2(128)	Tên của chính sách
POLICY_FUNCTION_OWNER	VARCHAR2(128)	Chủ sở hữu của hàm chính sách
PREDICATE	VARCHAR2(4000)	Tiền đề cho chính sách (cắt ngắn thành 4000 byte)

**Table II.2** Bảng miêu tả các cột trong bảng dữ liệu

Ví dụ thực thi câu lệnh

Worksheet

Query Builder

1

2

SELECT \* FROM V\$VPD\_POLICY;

Query Result

SQL | All Rows Fetched: 11 in 0.058 seconds

	ADDRESS	PARADDR	SQL_HASH	SQL_ID	CHILD_N...	OBJECT_OWNER	OBJECT_NAME
1	000000006D5FA...	00000000707F1...	2290355713	bsugz6f488...	0	C##SYSUSER	NHANSU
2	00000000709DF...	0000000070A39...	723915065	bukzskhpkc...	0	C##SYSUSER	SINHVIENT
3	000000006CB28...	000000006CB29...	2495228386	05q855yabn...	0	C##SYSUSER	NHANSU
4	000000006CEAB...	000000006CB47...	707029177	dtx5298p28...	0	C##SYSUSER	NHANSU
5	0000000070791...	000000006CB28...	3222210808	0gvzp3300y...	0	C##SYSUSER	NHANSU
6	000000006CAF6...	000000006CAF6...	3927651315	c1z7wymplq...	0	C##SYSUSER	NHANSU
7	000000006271E...	000000006CFEC...	1558536154	289uf19ffa...	0	C##SYSUSER	NHANSU
8	0000000070BEF...	0000000070A15...	1305043902	1mu7ww56wk...	0	C##SYSUSER	SINHVIENT
9	000000006CB39...	000000006CB44...	1913748492	7jm5a5t12...	0	C##SYSUSER	NHANSU
10	000000006D5FA...	000000006CB40...	2435689900	6b0ws5q8kv...	0	C##SYSUSER	NHANSU
11	000000006CB0E...	000000006CB07...	4146053680	a8fu07zvjjz...	0	C##SYSUSER	NHANSU

POLICY_GROUP	POLICY	POLICY_FUNCTION_OWNER	PREDICATE	CON_ID
SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
SYS_DEFAULT	SU_SINHVIENT	C##SYSUSER	(null)	3
SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
SYS_DEFAULT	SU_SINHVIENT	C##SYSUSER	(null)	3
SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3

### III. Tài liệu tham khảo

- Using Application Contexts to Retrieve User Information - 11g Release 2 (11.2)
- Implementing Application Context and Fine-Grained Access Control
- Using Application Contexts to Retrieve User Information

- Application contexts in a web environment
- Virtual Private Databases (VPD)
- Oracle Database PL/SQL Packages and Types Reference - DBMS\_RLS
- Implementing Virtual Private Databases (VPD)
- Securing Sensitive Data with Oracle VPD
- Securing Sensitive Data with Oracle VPD
- Using Oracle VPD to Control Data Access
- CREATE CONTEXT
- Oracle 12c Application Context