

FINE-GRAINED ACCESS CONTROL

Chuyên đề hệ
quản trị cơ sở dữ
liệu nâng cao

Nhóm 6

GIỚI THIỆU THÀNH VIÊN

Lê Thanh Khôi Nguyên

21127373

Dương Phước Sang

21127157

Nguyễn Hiển Đạt

21127591

Phan Trung Kiên

21127085





MỤC LỤC



1. Virtual Private Database

- VPD concept
- Column-level VPD and column-masking
- VPD security policy
- Configure VPD for FGAC
- Limit the scope of VPD



2. Application context

- Application context features and setup
- Session-based/Local/Global Application Context
- Use Application context for FGAC
- Check the policy effect



1. VPD

1.1 VPD CONCEPT

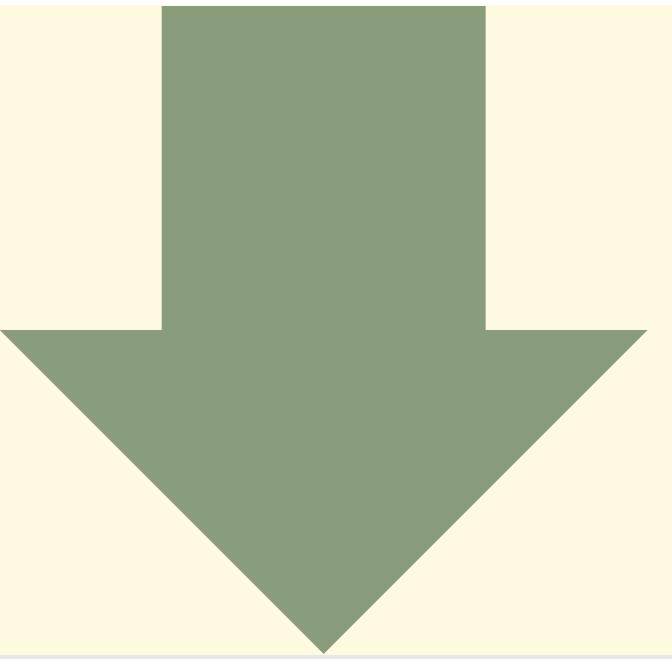


1.1 VPD CONCEPTS

- VPD tạo ra các chính sách bảo mật kiểm soát quyền truy cập ở cấp dòng và cột.
- Khi có 1 câu truy vấn trên đối tượng được bảo vệ, VPD thêm một điều kiện động WHERE vào câu lệnh SQL.

1.1 VPD CONCEPTS

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



```
SELECT * FROM OE.ORDERS  
WHERE SALES REP ID = 159;
```



1.1 VPD CONCEPTS

Các thành phần của VPD policy:

- Function để tạo ra mệnh đề động WHERE.
- Thủ tục để gắn function trên vào đối tượng cần được bảo vệ.

1.1 VPD CONCEPTS

Các thành phần của VPD policy:

- **Function** để tạo ra mệnh đề động WHERE.
- Policy để gắn function trên vào đối tượng cần được bảo vệ.

1.1 VPD CONCEPTS

Function phải đảm bảo các yếu tố sau:

- Nhận 2 tham số đầu vào là tên schema và tên đối tượng cần được bảo vệ. 2 tham số có kiểu dữ liệu là VARCHAR2.
- Trả về một mệnh đề WHERE hợp lệ. Kiểu dữ liệu của mệnh đề này là kiểu VARCHAR2.

1.1 VPD CONCEPTS

Các thành phần của VPD policy:

- Function để tạo ra mệnh đề động WHERE.
- Thủ tục để gắn function trên vào đối tượng cần được bảo vệ.

1.1 VPD CONCEPTS

- Thủ tục sẽ đăng ký function vừa tạo ở trên cho đối tượng được bảo vệ.
- Sử dụng package DBMS_RLS để tạo policy. Quyền sử dụng DBMS_RLS không được gán cho mọi người dùng, người quản trị cần có quyền EXECUTE ON DBMS_RLS để có thể sử dụng.

DEMO

Đầu tiên ta tạo policy function:

```
[-] Create or replace function hide_sal_comm (
    v_schema in varchar2,
    v_object in varchar2)
    return varchar2 as con varchar2(200);
Begin
    con := 'deptno=30';
    return con;
End hide_sal_comm;
```

Script Output x

Task completed in 0.034 seconds

Function HIDE_SAL_COMM compiled

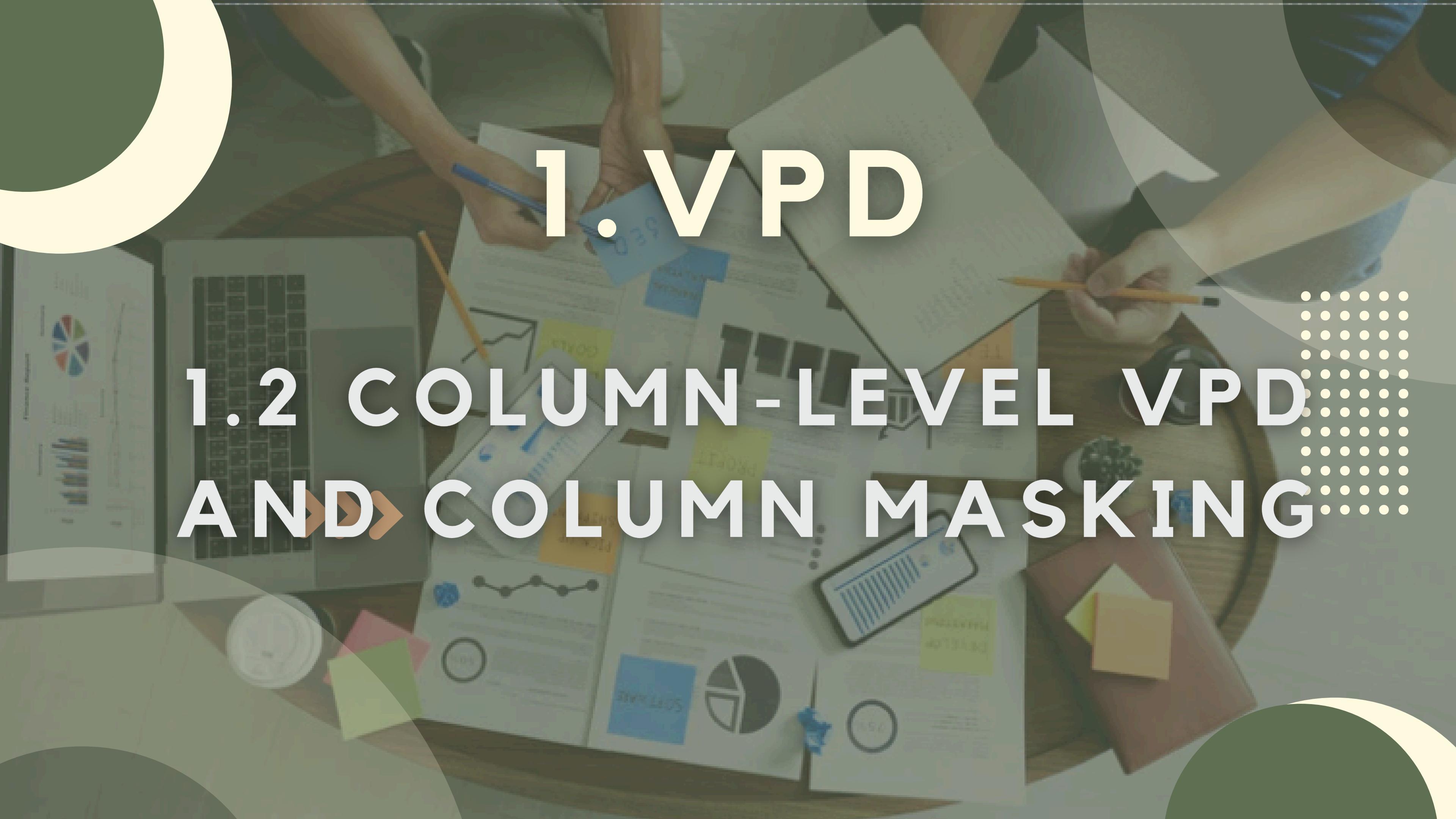
DEMO

Sau đó, cấu hình chính sách bằng thủ tục DBMS_RLS.ADD_POLICY như sau:

```
|- Begin
  DBMS_RLS.ADD_POLICY (
    object_schema  => 'scott',
    object_name    => 'emp',
    policy_name    => 'hide_sal_policy',
    policy_function => 'hide_sal_comm');
End;

Script Output  x
  | Task completed in 0.041 seconds
PL/SQL procedure successfully completed.
```

- object_schema: Tên schema chứa đối tượng cần được bảo vệ.
- object_name: Tên đối tượng cần được bảo vệ.
- policy_name: Tên chính sách.
- policy_function: Tên function vừa tạo ở trên.



1. VPD

1.2 COLUMN-LEVEL VPD AND> COLUMN MASKING

1.2 COLUMN-LEVEL VPD AND COLUMN MASKING

- .VPD cho phép bảo vệ dữ liệu theo dòng khi câu truy vấn tham chiếu đến cột được bảo vệ.
- Chỉ định tên cột cần bảo vệ bằng cách dùng tham số sec_relevant_cols của thủ tục DBMS_RLS.ADD_POLICY.

DEMO

Giả sử ta có một chính sách bảo vệ trên bảng EMP của SCOTT quy định một user chỉ được xem lương của những người thuộc department của họ (deptno = 30). Các cột liên quan đến chính sách bảo mật này gồm cột sal và comm.

DEMO

Đầu tiên ta tạo policy function:

```
[-] Create or replace function hide_sal_comm (
    v_schema in varchar2,
    v_object in varchar2)
    return varchar2 as con varchar2(200);
Begin
    con := 'deptno=30';
    return con;
End hide_sal_comm;
```

Script Output x

Task completed in 0.034 seconds

Function HIDE_SAL_COMM compiled

DEMO

Sau đó, cấu hình chính sách bằng thủ tục DBMS_RLS.ADD_POLICY 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;
Script Output x
| Task completed in 0.04 seconds
PL/SQL procedure successfully completed.
```

- object_schema: Tên schema chứa đối tượng cần được bảo vệ.
- object_name: Tên đối tượng cần được bảo vệ.
- policy_name: Tên chính sách.
- policy_function: Tên function vừa tạo ở trên.
- sec_relevant_cols: Tên cột cần bảo vệ.

DEMO

Kết quả sau khi áp dụng chính sách bảo mật

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	

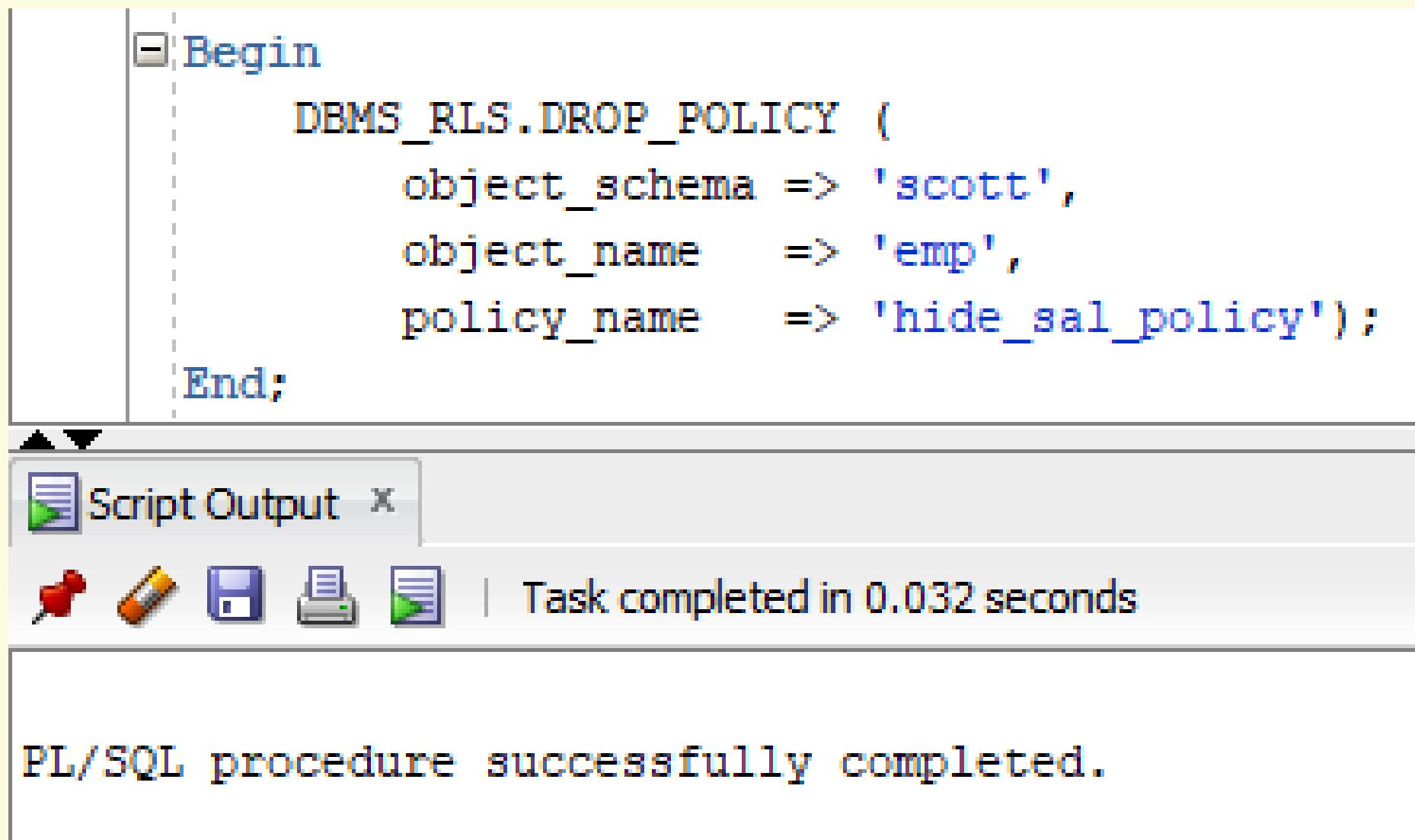
6 rows selected.

DEMO

- Nếu một câu truy vấn tham khảo đến cột được bảo vệ, việc áp dụng chính sách trên sẽ hạn chế số dòng dữ liệu trả về.
- Trong khi đó nhu cầu thực tế chúng ta muốn xem cả thông tin của những người dùng ở những phòng ban khác ngoại trừ lương của họ.
- Ta có thể làm việc này với column-masking. Với column-masking, thông tin của những người ở những phòng ban khác sẽ được hiển thị đầy đủ, nhưng cột salary và comm của họ sẽ có giá trị NULL.

DEMO

Đầu tiên xóa chính sách bảo mật tạo ở trên



```
Begin
    DBMS_RLS.DROP_POLICY (
        object_schema => 'scott',
        object_name   => 'emp',
        policy_name   => 'hide_sal_policy');
End;
```

Script Output X

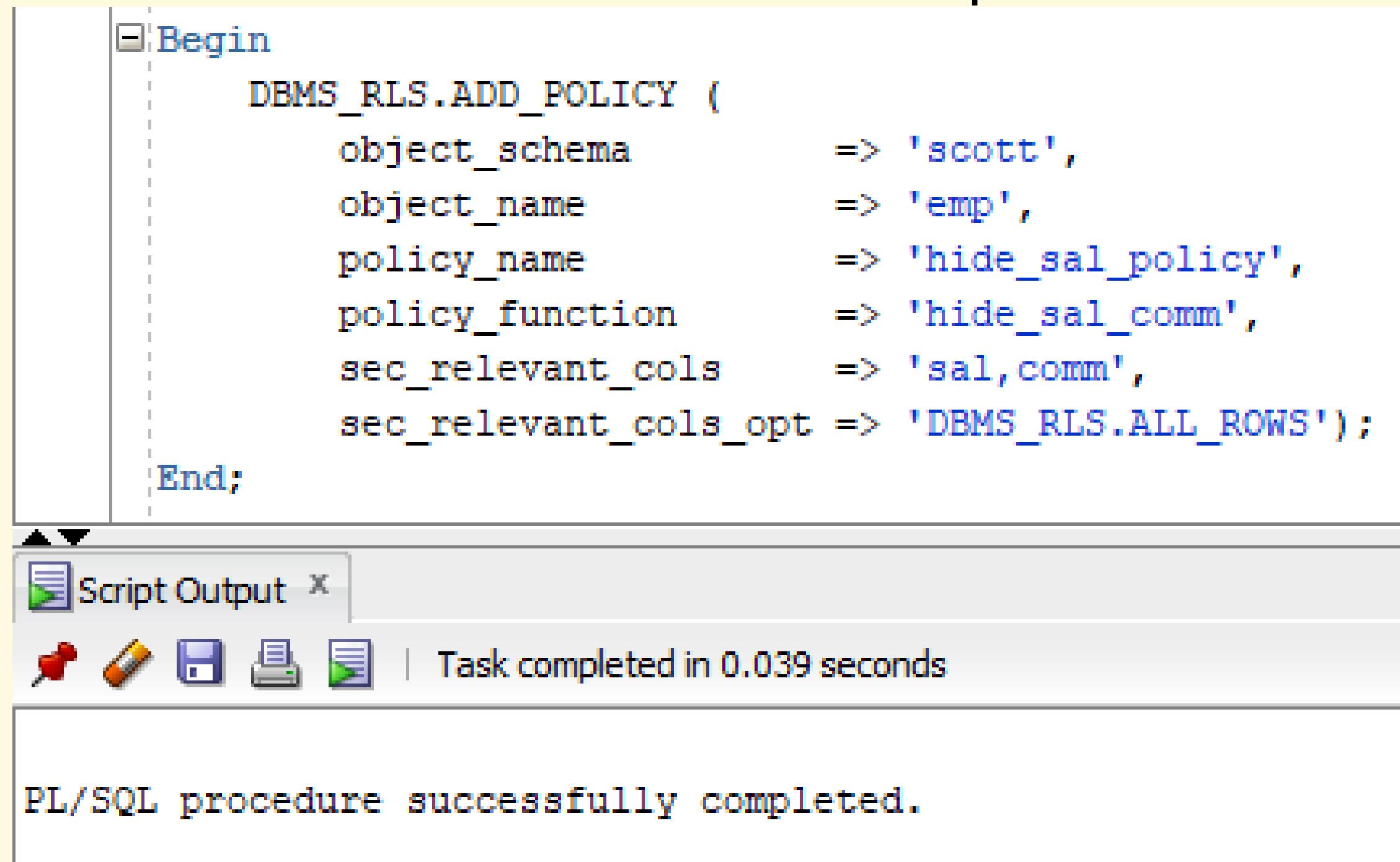
Task completed in 0.032 seconds

PL/SQL procedure successfully completed.

policy_name: Tên chính sách được đặt ở thủ tục
DBMS_RLS.ADD_POLICY

DEMO

Thêm lại chính sách với thay đổi ở tham số
sec_relevant_cols_opt



```
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',
        sec_relevant_cols_opt => 'DBMS_RLS.ALL_ROWS');
End;
```

Script Output | Task completed in 0.039 seconds

PL/SQL procedure successfully completed.

DEMO

Kết quả sau khi áp dụng chính sách:

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

- Mặc định policy sẽ được áp dụng cho tất cả câu lệnh Select, Insert, Delete, Update. Người quản trị có thể sử dụng tham số STATEMENT_TYPES để chỉ rõ policy áp dụng cho loại câu lệnh nào.
- SYS không bị ảnh hưởng bởi bất cứ chính sách bảo mật nào. Đó là do SYS có quyền EXEMPT ACCESS POLICY, đây là quyền mặc định có của những user có quyền SYSDBA.
- Nhiều policy có thể cùng áp dụng cho cùng một đối tượng. Khi đó hệ quản trị sẽ kết hợp các policy lại với nhau theo phép AND.

1. VPD

1.3 VPD SECURITY POLICY



1.3.1 ORGANIZE POLICIES AS GROUPS

1.3.2 DEFAULT POLICIES

>>>



1.3.1 ORGANIZE POLICIES AS GROUPS

1.3.2 DEFAULT POLICIES

>>>

1.3.1 ORGANIZE POLICIES AS GROUPS

- Một nhóm chính sách (policy group) là một tập hợp các chính sách thuộc về một ứng dụng cụ thể.
- Ta có thể chỉ định một application context để xác định nhóm chính sách đang có hiệu lực. Khi người dùng truy cập vào đối tượng, Oracle sẽ tra cứu application context để xác định nhóm chính sách đang có hiệu lực và thực thi tất cả chính sách trong nhóm đó.

1.3.1 ORGANIZE POLICIES AS GROUPS

- Nhóm chính sách giúp xác định chính xác những chính sách 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.

1.3.1 ORGANIZE POLICIES AS GROUPS

VÍ DỤ

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.

1.3.1 ORGANIZE POLICIES AS GROUPS

VÍ DỤ

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.

1.3.1 ORGANIZE POLICIES AS GROUPS

TẠO GROUP POLICY

```
BEGIN
    DBMS_RLS.CREATE_POLICY_GROUP(
        object_schema => 'HR',
        object_name => 'EMP_VPD_TEST',
        policy_group => 'POL_GRP_A');
END;
```

 **Script Output** X

     | Task completed in 0.037 seconds

PL/SQL procedure successfully completed.

1.3.1 ORGANIZE POLICIES AS GROUPS

THÊM POLICY VÀO GROUP POLICY

```
DECLARE
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 0.037 seconds

PL/SQL procedure successfully completed.



1.3.1 ORGANIZE POLICIES AS GROUPS

1.3.2 DEFAULT POLICIES

>>>

1.3.2 DEFAULT POLICIES

- Trong một nhóm chính sách bảo mật, người dùng có thể chỉ định một chính sách bảo mật làm chính sách mặc định.
- Chính sách bảo mật mặc định cho phép thực thi bảo mật dưới mọi điều kiện. Để triển khai chính sách bảo mật mặc định, ta thêm chính sách vào nhóm chính sách SYS_DEFAULT.

1.3.2 DEFAULT POLICIES

- Nhóm chính sách SYS_DEFAULT có thể không chứa chính sách bảo mật nào.
- Nhóm chính sách SYS_DEFAULT không thể bị xóa.
- Nếu ta thêm các chính sách liên quan đến hai hoặc nhiều đối tượng vào nhóm chính sách SYS_DEFAULT, thì mỗi đối tượng sẽ có một nhóm chính sách SYS_DEFAULT riêng được liên kết với nó.

1.3.2 DEFAULT POLICIES

VÍ DỤ

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. Các chính sách được tổ chức trong cấu trúc cây như sau:

SYS_DEFAULT

- policy1 (scott/emp)
- policy3 (scott/emp)

SYS_DEFAULT

- policy2 (scott/dept)



1. VPD

1.4 CONFIGURE VPD FOR FINE-GRAINED ACCESS CONTROL

1.4.1 USE THE DBMS_RLS PACKAGE AND THE CREATE CONTEXT COMMAND

- 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.

1.4.1 USE THE DBMS_RLS PACKAGE AND THE CREATE CONTEXT COMMAND

- 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.

DEMO

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
```

DEMO

Bước 1: Tạo bảng dữ liệu

```
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);

Table ORDERS_TAB created.

1 row inserted.

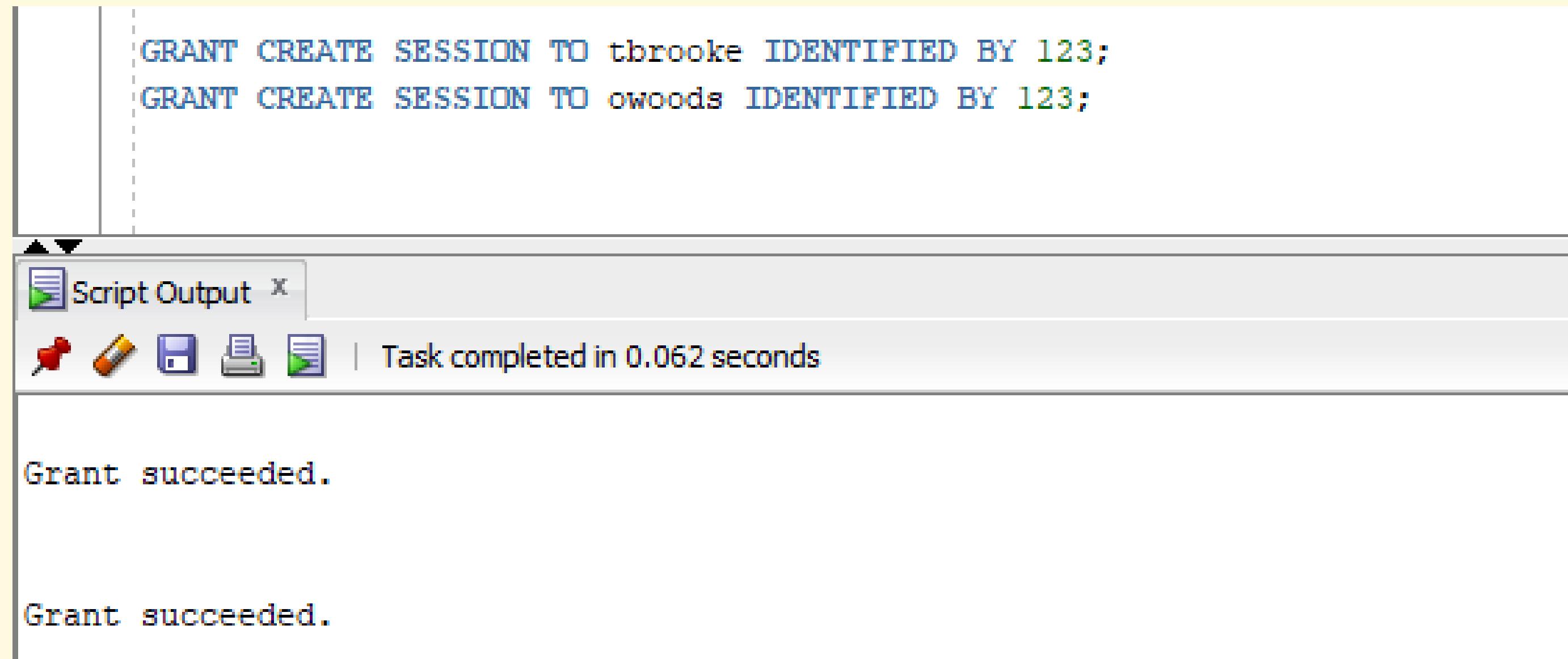
1 row inserted.

1 row inserted.

Connection created by CONNECT script command disconnected
```

DEMO

Bước 1: Tạo người dùng



The screenshot shows a SQL developer interface with a script editor and an output window.

Script Editor (Top):

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

Output Window (Bottom):

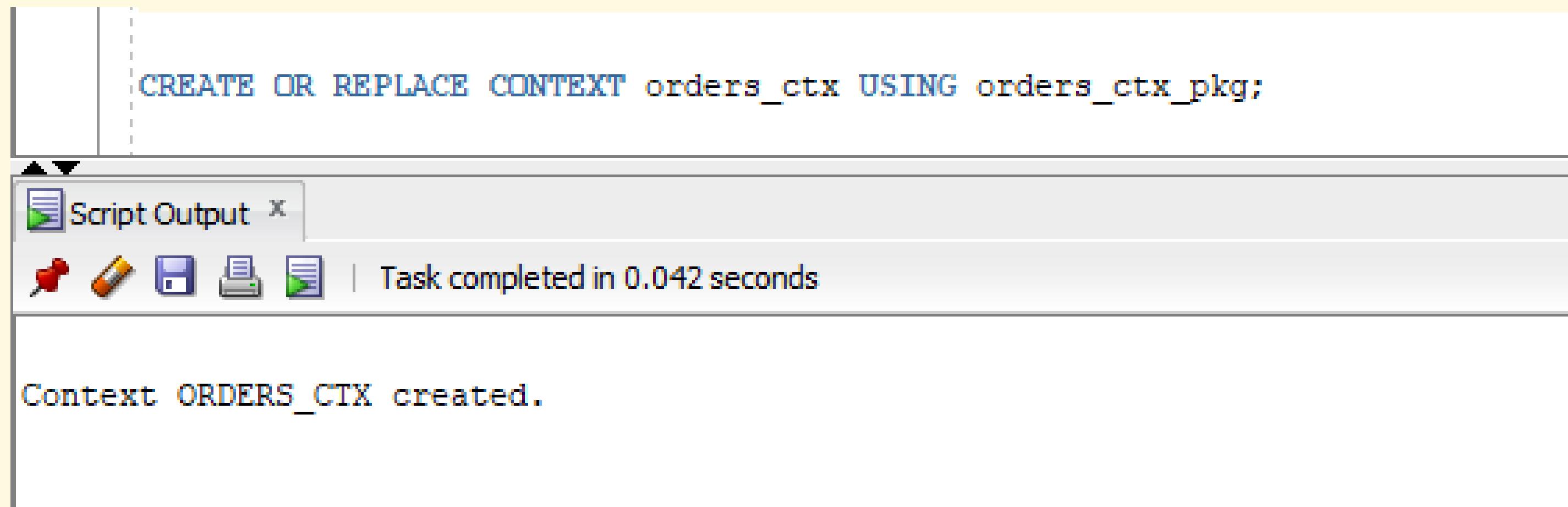
Script Output X

Task completed in 0.062 seconds

```
Grant succeeded.  
Grant succeeded.
```

DEMO

Bước 2: Tạo Application context



The screenshot shows a SQL developer interface. In the top-left pane, a script is being run with the following command:

```
CREATE OR REPLACE CONTEXT orders_ctx USING orders_ctx_pkg;
```

In the bottom pane, the output of the script is displayed. The output window title is "Script Output". The output message is:

Context ORDERS_CTX created.

Below the output message, a status bar indicates: "Task completed in 0.042 seconds".

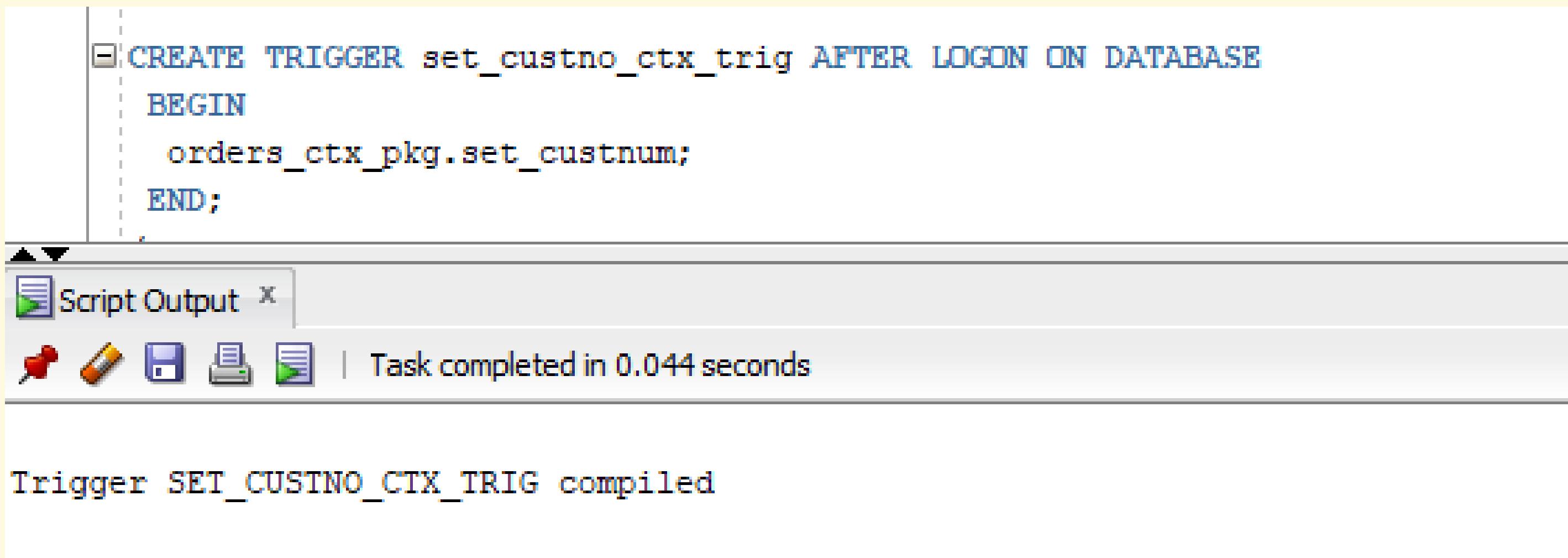
DEMO

Bước 3: Tạo Package

```
CREATE OR REPLACE PACKAGE orders_ctx_pkg IS
  PROCEDURE set_custnum;
END;
/
CREATE OR REPLACE PACKAGE BODY orders_ctx_pkg IS
  PROCEDURE set_custnum
  AS
    custnum NUMBER;
  BEGIN
    SELECT cust_no INTO custnum FROM SCOTT.CUSTOMERS
      WHERE cust_email = SYS_CONTEXT('USERENV', 'SESSION_USER');
    DBMS_SESSION.SET_CONTEXT('orders_ctx', 'cust_no', custnum);
  END set_custnum;
END;
/
Script Output X
✖️ 🖍️ 📁 🖨️ 🎵 | Task completed in 0.115 seconds
Package ORDERS_CTX_PKG compiled
Package Body ORDERS_CTX_PKG compiled
```

DEMO

Bước 4: Tạo Trigger



The screenshot shows the Oracle SQL Developer interface. In the top-left pane, a SQL script is being run, displaying the following code:

```
CREATE TRIGGER set_custno_ctx_trig AFTER LOGON ON DATABASE
BEGIN
  orders_ctx_pkg.set_custnum;
END;
```

Below the script, the "Script Output" tab is active, showing the results of the execution:

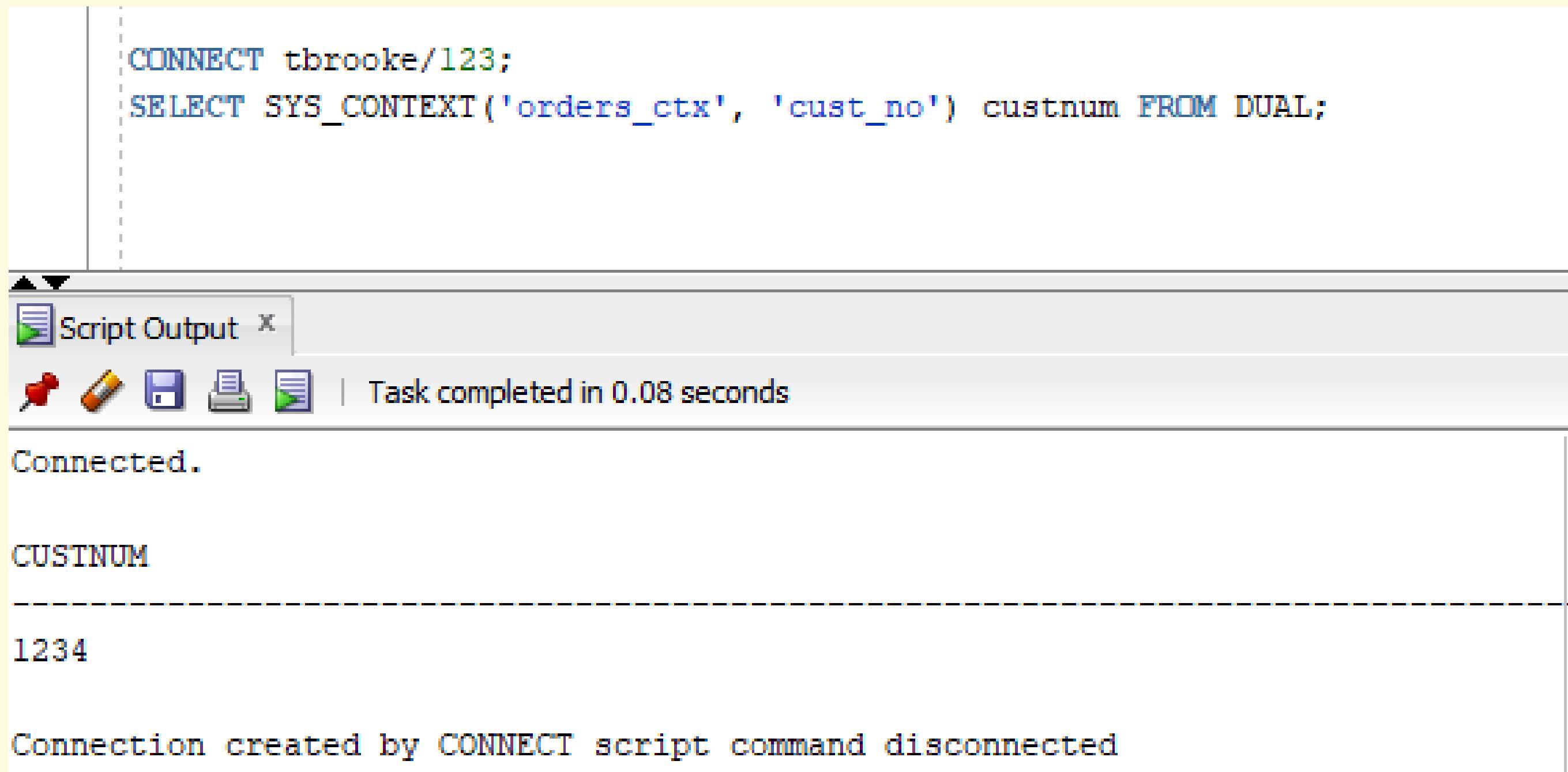
- Task completed in 0.044 seconds

At the bottom of the output pane, a message indicates the trigger has been compiled:

Trigger SET_CUSTNO_CTX_TRIG compiled

DEMO

Bước 4: Tạo Trigger



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

Script Output X

Connected.

CUSTNUM

1234

Connection created by CONNECT script command disconnected

DEMO

Bước 5: Tạo PL/SQL Function

```
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;
```

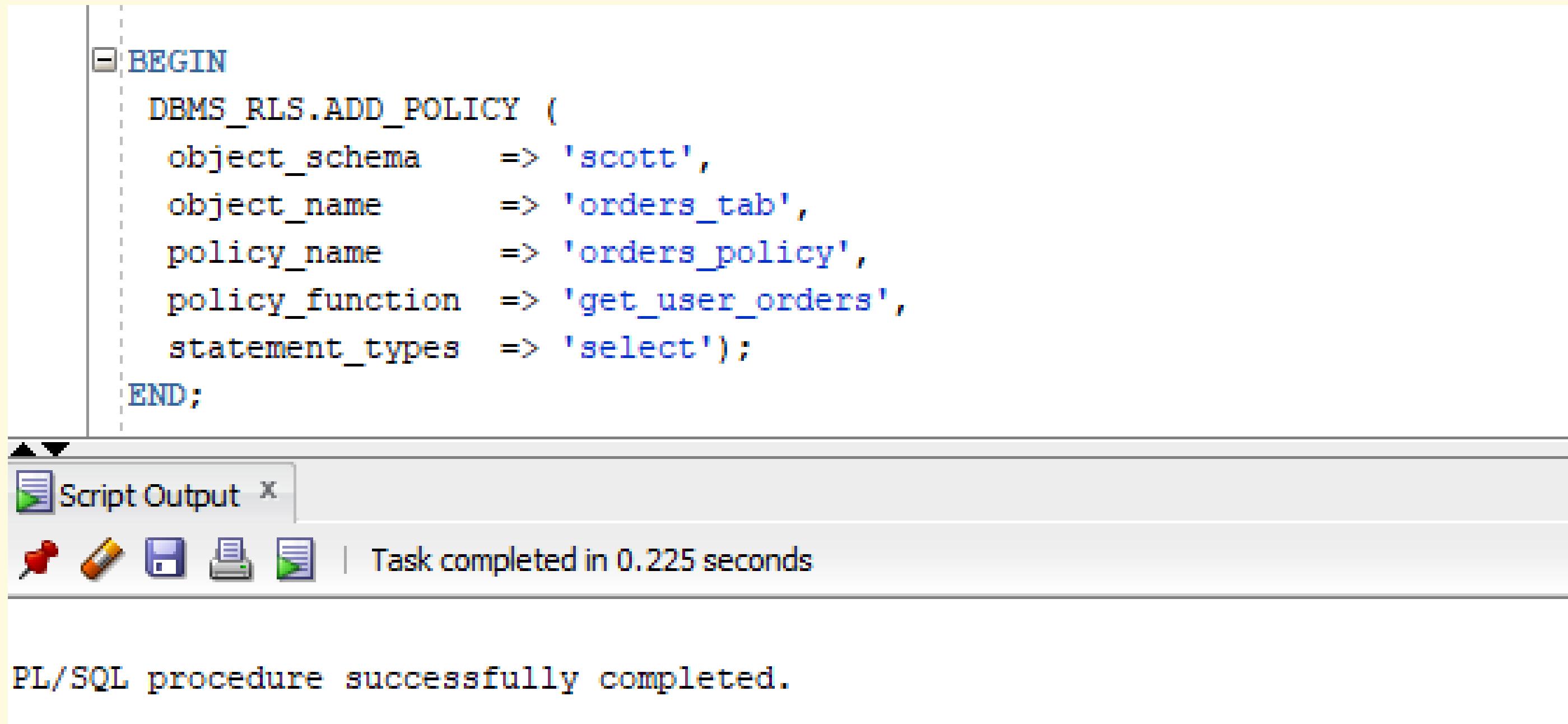
Script Output x

Task completed in 0.039 seconds

Function GET_USER_ORDERS compiled

DEMO

Bước 6: Tạo chính sách bảo mật



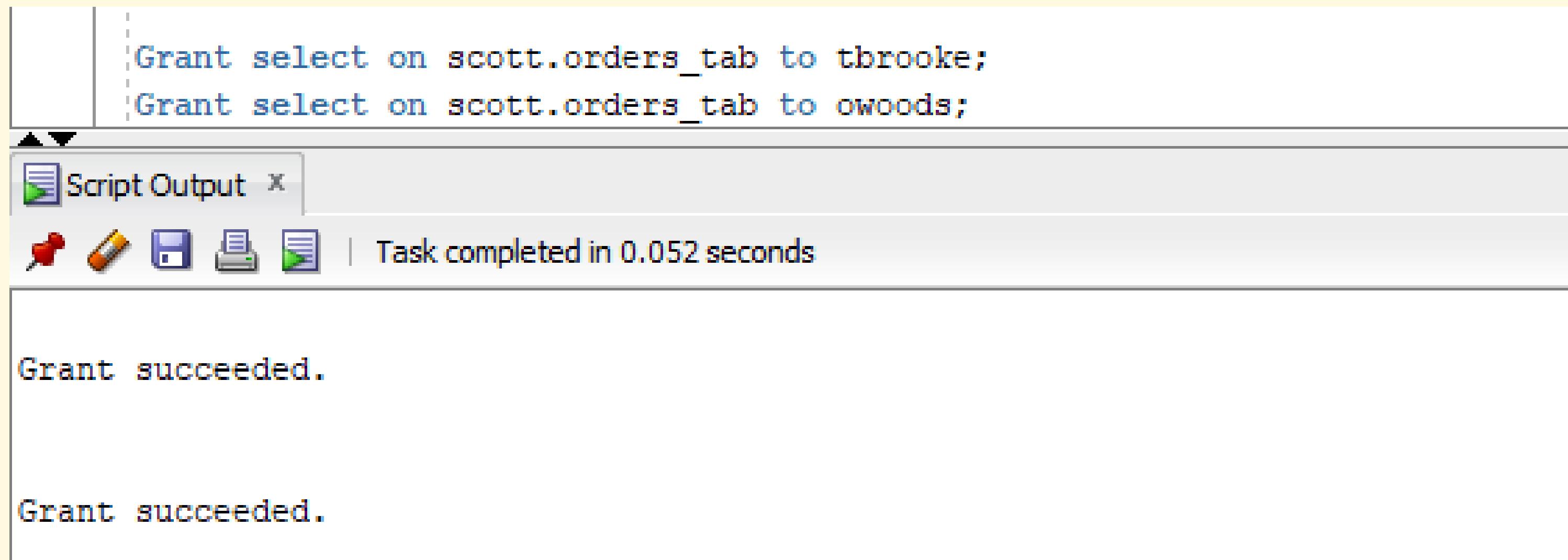
```
begin
  dbms_rls.add_policy (
    object_schema    => 'scott',
    object_name      => 'orders_tab',
    policy_name      => 'orders_policy',
    policy_function  => 'get_user_orders',
    statement_types   => 'select');
end;
```

Script Output | Task completed in 0.225 seconds

PL/SQL procedure successfully completed.

DEMO

Bước 7: Thử kết quả



The screenshot shows a database management interface with a script editor and a results pane. The script editor contains the following SQL statements:

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

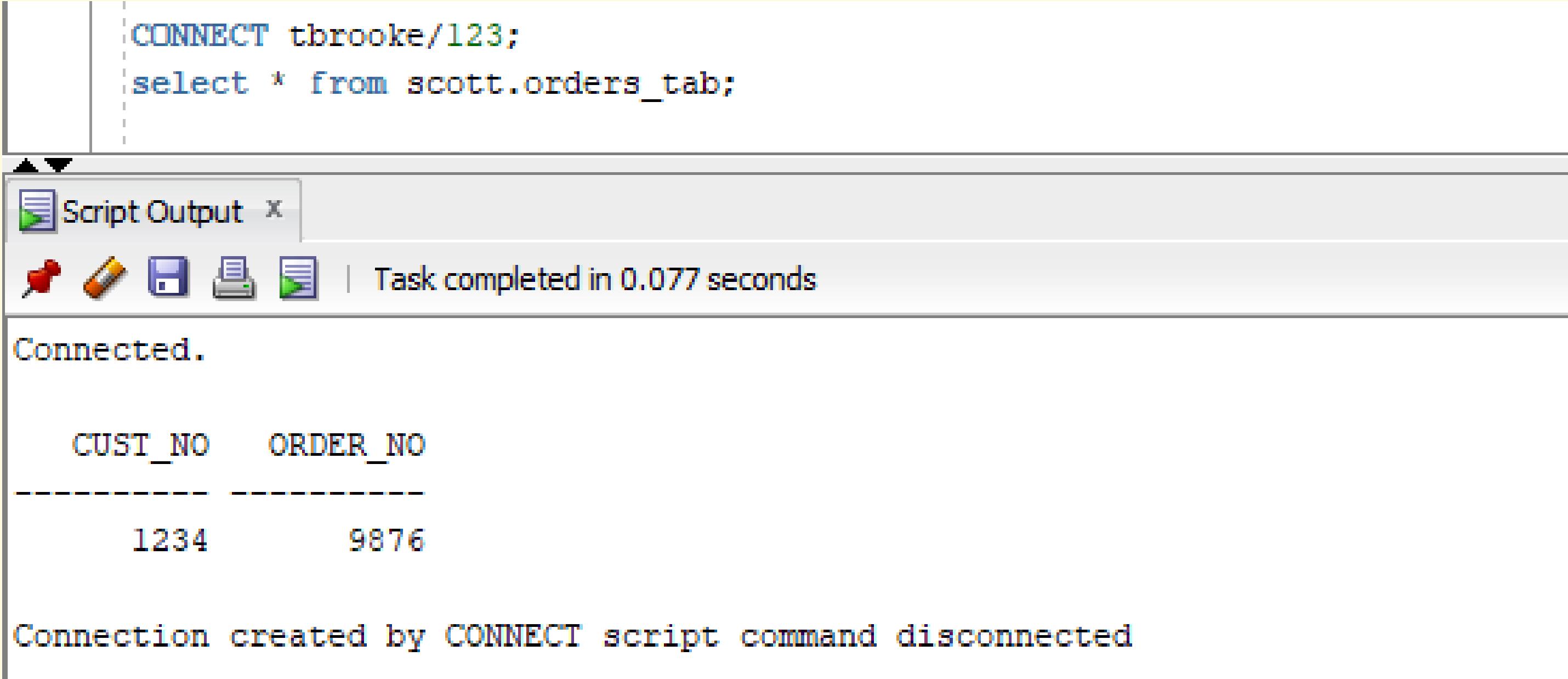
The results pane below shows the output of the execution:

```
Grant succeeded.
Grant succeeded.
```

The results pane includes a toolbar with icons for script, save, and print, and a status message: "Task completed in 0.052 seconds".

DEMO

Bước 7: Thử kết quả



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

Script Output X

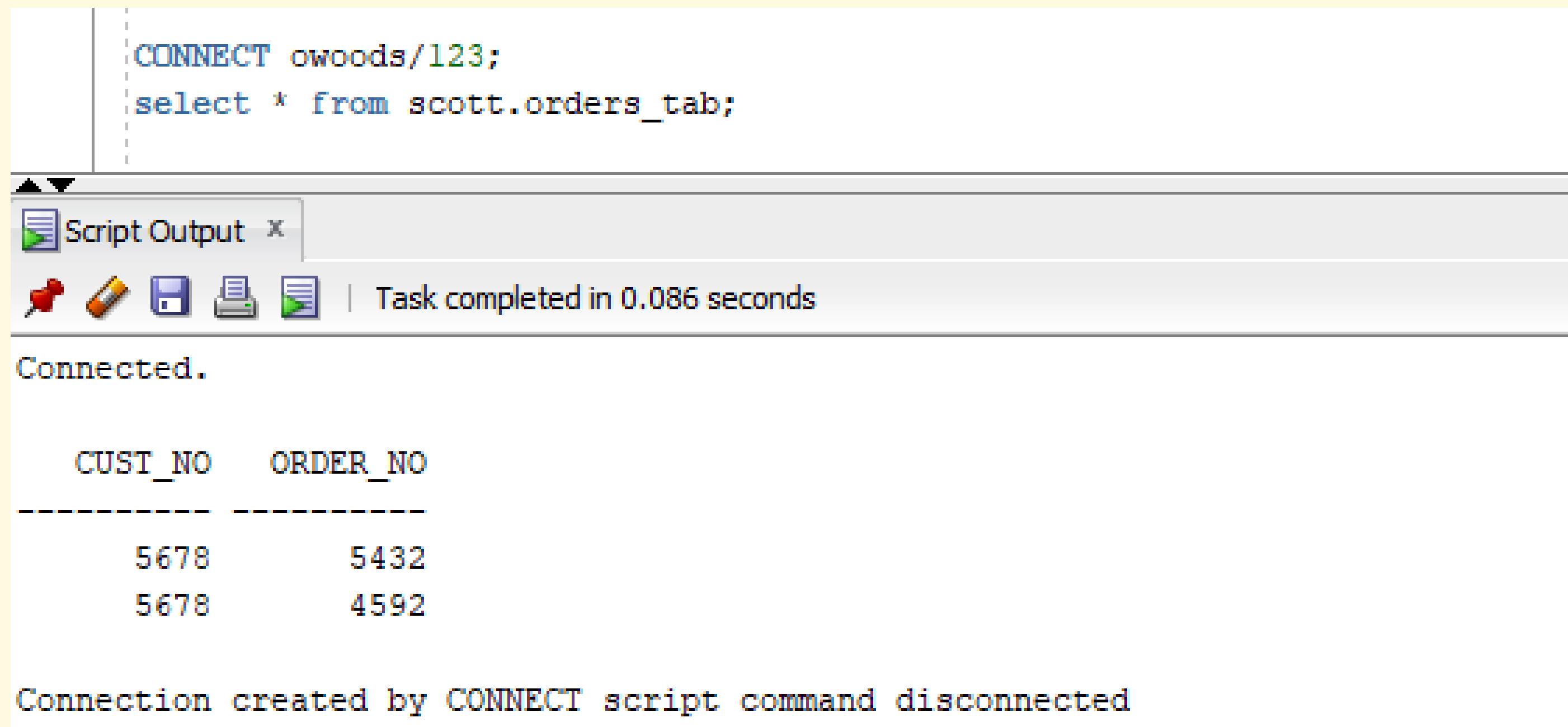
Connected.

CUST_NO	ORDER_NO
1234	9876

Connection created by CONNECT script command disconnected

DEMO

Bước 7: Thử kết quả



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

1. VPD

1.5 LIMIT THE SCOPE OF
VPD USE (IN WHICH
SCHEMAS)

1.5 LIMIT THE SCOPE OF VPD USE (IN WHICH SCHEMAS)

- Phạm vi sử dụng của VPD có thể được giới hạn bằng cách áp dụng chúng vào các Schema cụ thể trong cơ sở dữ liệu.
- Bằng cách áp dụng VPD vào các Schema cụ thể, người dùng có thể kiểm soát truy cập vào dữ liệu trong các Schema đó dựa trên các chính sách được xác định trước.

1.5 LIMIT THE SCOPE OF VPD USE (IN WHICH SCHEMAS)

- Cách giới hạn phạm vi sử dụng của VPD cho các schema cụ thể:
 1. Xác định các schema mà người dùng muốn áp dụng chính sách VPD.
 2. Tạo các chính sách VPD.
 3. Áp dụng các chính sách.
 4. Kiểm tra các chính sách.

DEMO

Bước 1: Xem các schema

USERNAME
1 SYS
2 SYSTEM
3 XS\$NULL
4 OJVMSYS
5 LBACSYS
6 OUTLN
7 DBSNMP
8 APPQOSSYS
9 GGSYS
10 ANONYMOUS
11 DBSFWUSER
12 CTXSYS
13 DVSYS
14 DVF
15 AUDSYS
16 GSMADMIN_INTERNAL
17 OLAPSYS
18 MDSYS
19 XDB
20 WMSYS
21 GSMCATUSER
22 DG006
23 MDDATA
24 LYNN
25 NV030
26 NV003
27 NV015

```
SELECT username  
FROM dba_users;
```

USERNAME
25 NV030
26 NV003
27 NV015
28 NV021
29 NV027
30 SYSBACKUP
31 GSMUSER
32 NV034
33 NV035
34 REMOTE_SCHEDULER_AGENT
35 AMY
36 ADMIN
37 NV001
38 NV025
39 NV031
40 NV032
41 DG002
42 NV009
43 GSMROOTUSER
44 SYSRAC
45 FRED
46 THEOTA
47 NV033
48 NV038
49 JOE
50 DG001
51 ANNII

DEMO

Bước 2: Tạo hàm chính sách

```
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;
```

DEMO

Bước 3: Áp dụng chính sách

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

```
select *
from admin.empholiday;
```

	EMPNO	NAME	HOLIDAY
	1	2 Annu	22-MAY-24

DEMO

Bước 3: Áp dụng chính sách

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



2. APPLICATION CONTEXT

2.1 APPLICATION CONTEXT FEATURES: AND SETUP



2.1.1 FEATURES

2.1.2 SETUP





2.1.1 FEATURES

2.1.2 SETUP



2.1.1 FEATURES

- Application Context là một tập các cặp name - value được lưu trữ trong bộ nhớ.

2.1.1 FEATURES

- Quản lý Thông Tin Phiên.
- Bảo Mật và Quyền Truy Cập.
- Tích Hợp Ứng Dụng.
- Audit Logging.



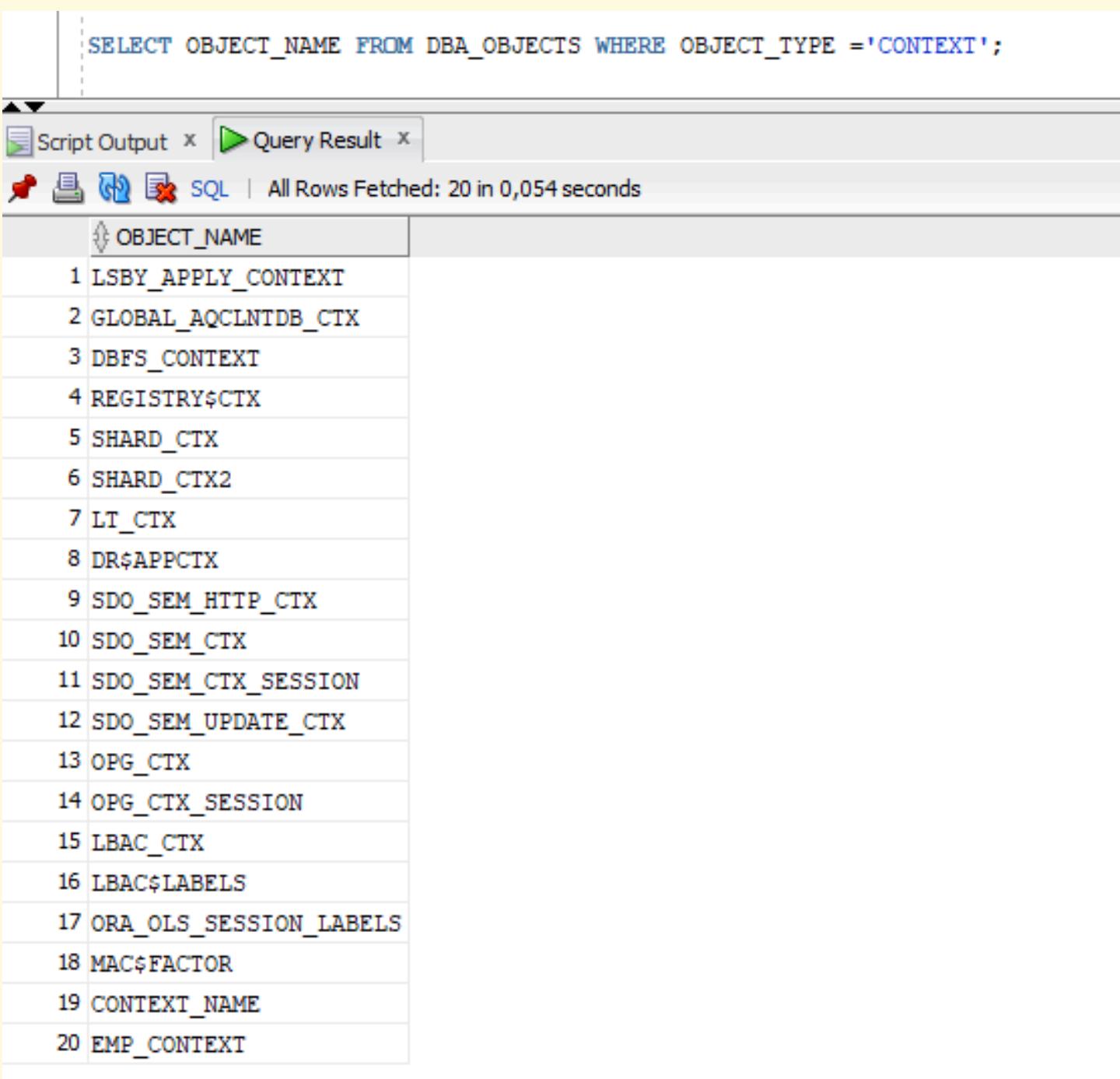
2.1.1 FEATURES

2.1.2 SETUP



DEMO

Bước 1: Tạo Application context



SELECT OBJECT_NAME FROM DBA_OBJECTS WHERE OBJECT_TYPE ='CONTEXT';

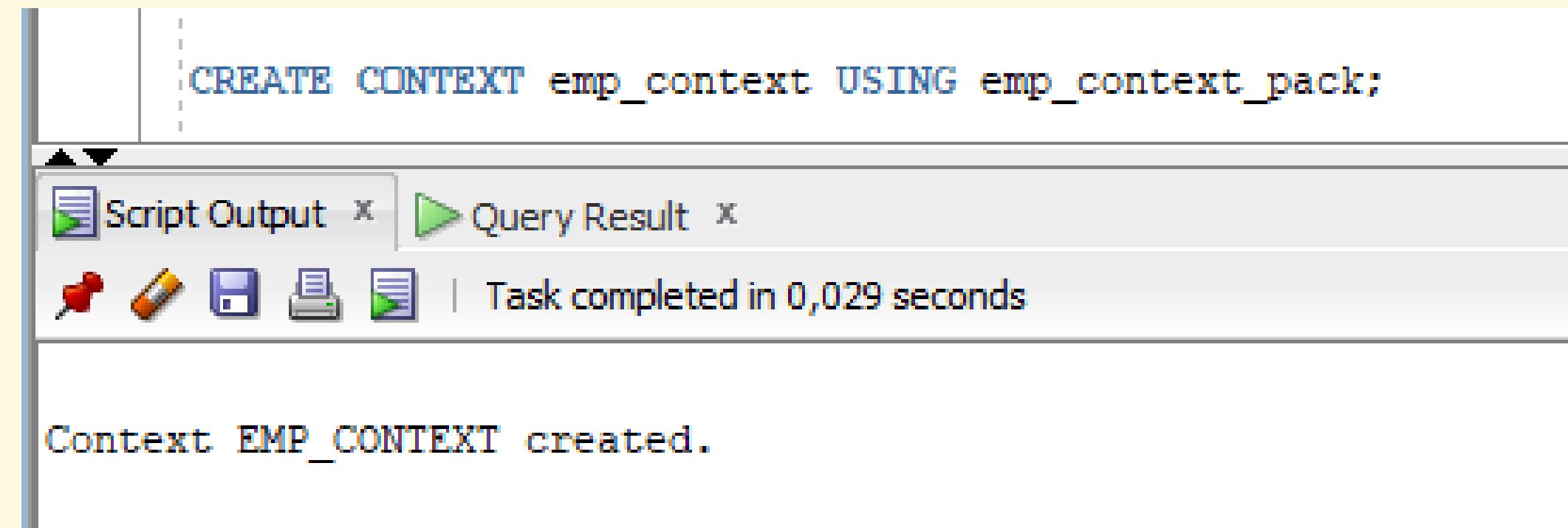
Script Output x Query Result x

SQL | All Rows Fetched: 20 in 0,054 seconds

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

DEMO

Bước 1: Tạo Application context



The screenshot shows a SQL developer interface. In the top-left pane, a script is being run with the command:

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

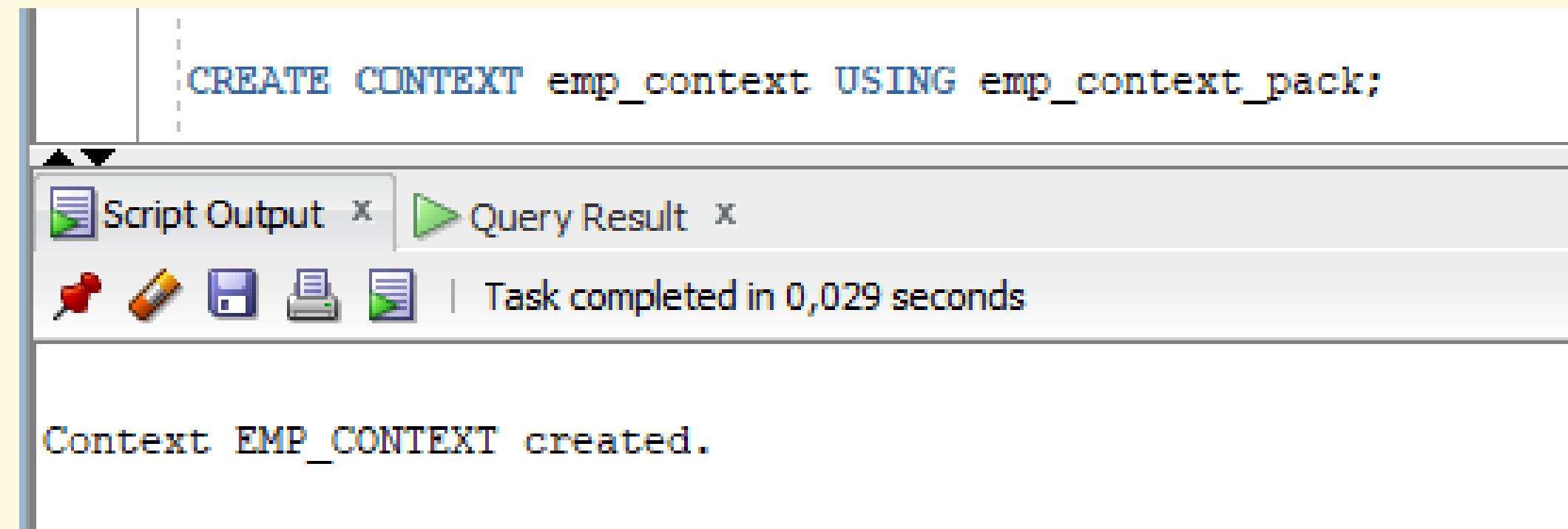
In the bottom pane, the output shows:

```
Context EMP_CONTEXT created.
```

The interface includes tabs for "Script Output" and "Query Result", and a toolbar with various icons.

DEMO

Bước 1: Tạo Application context



CREATE CONTEXT emp_context USING emp_context_pack;

Script Output x Query Result x

Task completed in 0,029 seconds

Context EMP_CONTEXT created.

The screenshot shows the Oracle SQL Developer interface. In the top-left, a code editor window contains the SQL command to create an application context. Below it is a toolbar with icons for script, edit, and file operations. The main pane shows the output of the command, indicating the context was created successfully and the task completed in 0.029 seconds. A message at the bottom of the output pane says 'Context EMP_CONTEXT created.'

	—	—	—
18	MAC\$FACTOR		
19	CONTEXT_NAME		
20	EMP_CONTEXT		

2. APPLICATION CONTEXT

2.2 TYPES OF APPLICATION CONTEXTS



A background photograph showing a person's hands on a laptop keyboard. The image is partially obscured by a dark, semi-transparent overlay. On the left side of this overlay, there are three large, stylized, overlapping circles in shades of green, yellow, and white. The text is positioned on the right side of the overlay.

2.2.1 LOCAL APPLICATION CONTEXT

2.2.2 SESSION-BASED APPLICATION CONTEXT

2.2.3 GLOBAL APPLICATION CONTEXTS



A background photograph showing a person's hands on a laptop keyboard. A large, semi-transparent circular magnifying glass effect is centered over the hands and keyboard. The image has a warm, slightly blurred aesthetic.

2.2.1 LOCAL APPLICATION CONTEXT

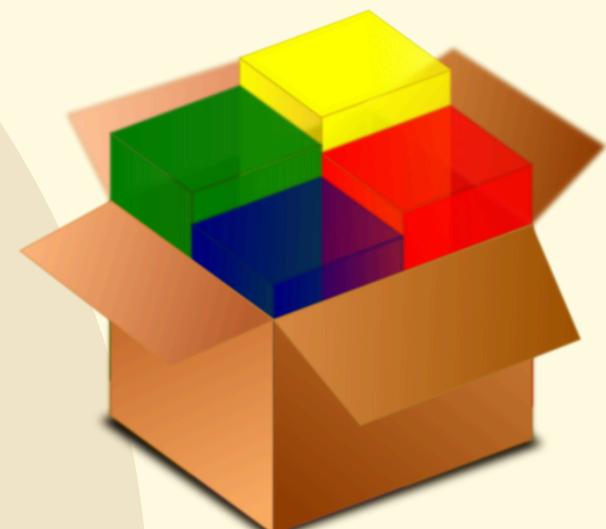
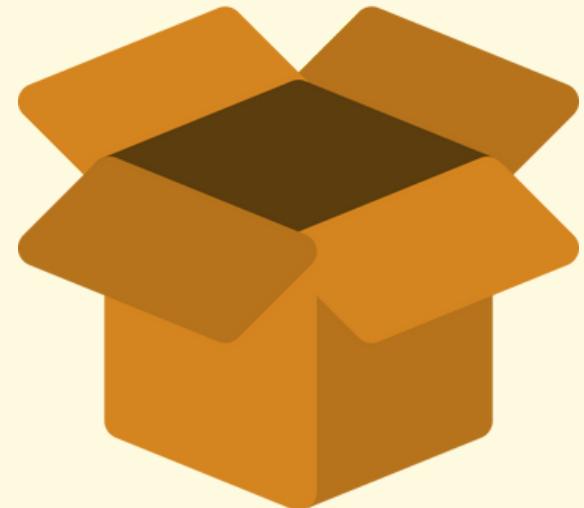
2.2.2 SESSION-BASED APPLICATION CONTEXT

2.2.3 GLOBAL APPLICATION CONTEXTS

>>>

NAMESPACE

HRAPP



Sử dụng câu lệnh **CREATE CONTEXT** để:

- Tạo một context trong một namespace.
- Liên kết một package với context.

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

Sử dụng thủ tục **SET_CONTEXT** để:

- Tạo các thuộc tính.
- Thiết lập giá trị cho các thuộc tính.

```
DBMS_SESSION.SET_CONTEXT('emp  
no_ctx', 'employee_id', emp_id);
```



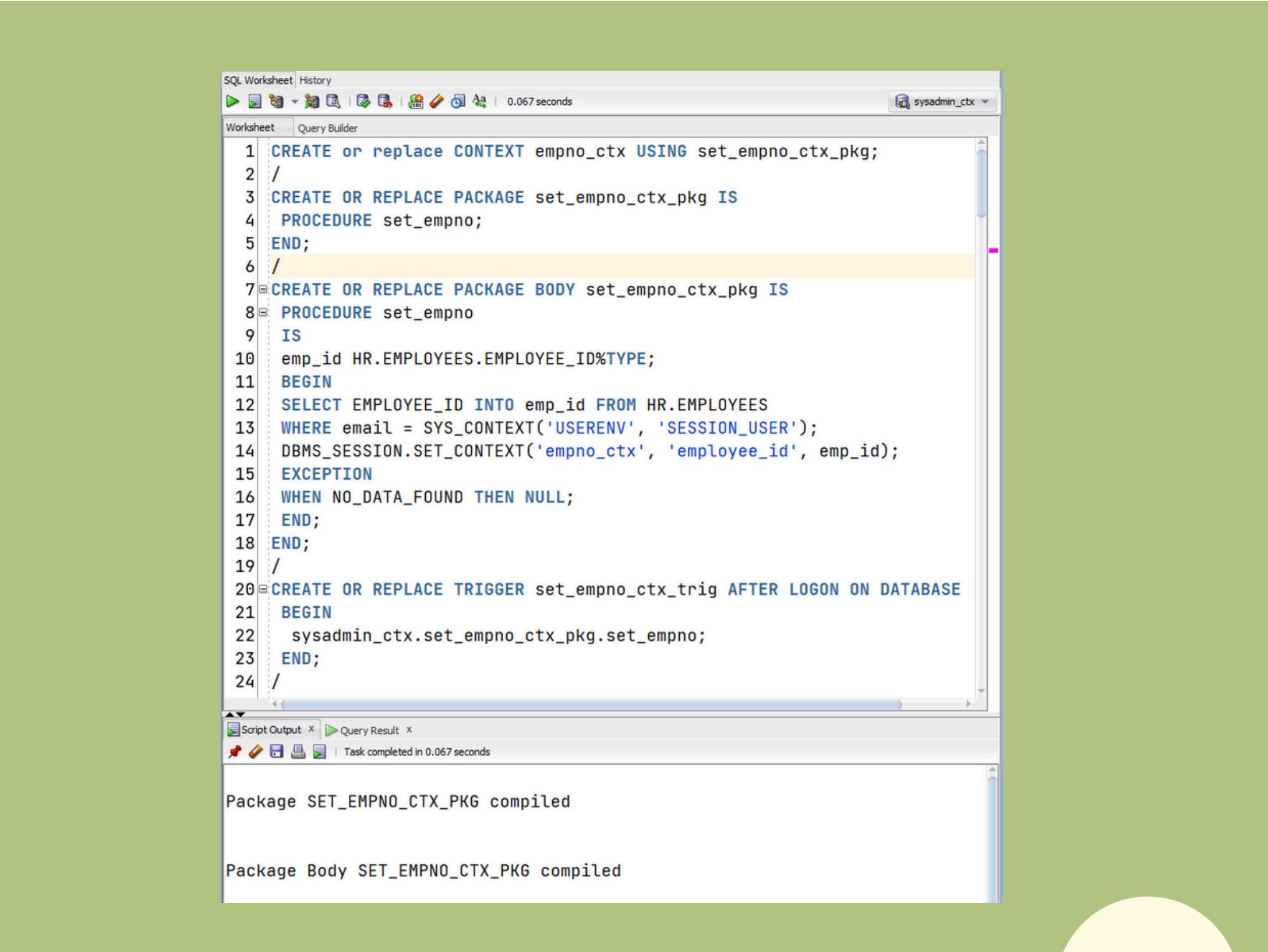
2.2.2 LOCAL APPLICATION CONTEXT

Triển khai một bối cảnh ứng dụng cục bộ

1. Tạo một bối cảnh ứng dụng.
2. Tạo một package PL/SQL để thiết lập bối cảnh.
3. Thiết lập thuộc tính cho bối cảnh.
4. Truy xuất thuộc tính bối cảnh trong ứng dụng.



TRIỀN KHAI MỘT BỐI CẢNH ỨNG DỤNG CỤC BỘ



The screenshot shows an Oracle SQL Worksheet interface. The top bar includes 'SQL Worksheet' and 'History' tabs, and a toolbar with various icons. The connection is set to 'sysadmin_ctx'. The main area displays the following PL/SQL code:

```
1 CREATE or replace CONTEXT empno_ctx USING set_empno_ctx_pkg;
2 /
3 CREATE OR REPLACE PACKAGE set_empno_ctx_pkg IS
4   PROCEDURE set_empno;
5 END;
6 /
7 CREATE OR REPLACE PACKAGE BODY set_empno_ctx_pkg IS
8   PROCEDURE set_empno
9   IS
10    emp_id HR.EMPLOYEES.EMPLOYEE_ID%TYPE;
11   BEGIN
12    SELECT EMPLOYEE_ID INTO emp_id FROM HR.EMPLOYEES
13    WHERE email = SYS_CONTEXT('USERENV', 'SESSION_USER');
14    DBMS_SESSION.SET_CONTEXT('empno_ctx', 'employee_id', emp_id);
15  EXCEPTION
16    WHEN NO_DATA_FOUND THEN NULL;
17  END;
18 END;
19 /
20 CREATE OR REPLACE TRIGGER set_empno_ctx_trig AFTER LOGON ON DATABASE
21 BEGIN
22   sysadmin_ctx.set_empno_ctx_pkg.set_empno;
23 END;
24 /
```

The code creates a context 'empno_ctx' using the 'set_empno_ctx_pkg'. It defines a package with a procedure 'set_empno' that retrieves the employee ID from the 'EMPLOYEES' table based on the user's email and sets it in the context. A trigger 'set_empno_ctx_trig' is created to call this procedure whenever a user logs on to the database.

The bottom panel shows the 'Script Output' and 'Query Result' tabs. The 'Script Output' tab displays the message 'Package SET_EMPNO_CTX_PKG compiled'. The 'Query Result' tab shows the message 'Package Body SET_EMPNO_CTX_PKG compiled'.

BƯỚC 1: TẠO MỘT BỐI CẢNH ỨNG DỤNG

Tạo một bối cảnh

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

- Đặt tên cho bối cảnh là empno_ctx
- Liên kết bối cảnh với package set_empno_ctx_pkg

Chỉ có thể thiết lập các thuộc tính của bối cảnh trong:

- Package được đặt tên ở trong câu lệnh CREATE CONTEXT
- Một hàm chức năng được liên kết với một chính sách.

Thiết lập các thuộc tính trong package bằng cách:

```
DBMS_SESSION.SET_CONTEXT.
```

BƯỚC 2: TẠO MỘT PACKAGE PL/SQL ĐỂ THIẾT LẬP BỐI CẢNH

TẠO THỦ TỤC SET_EMPNO_CTX_PKG.SET_EMPNO

- Sử dụng SYS_CONTEXT để xác định tên người dùng đăng nhập

```
SYS_CONTEXT('USERENV', 'SESSION_USER')
```

- Sử dụng tên người dùng để xác định ID nhân viên

```
SELECT EMPLOYEE_ID INTO emp_id
  FROM HR.EMPLOYEES WHERE email =
    SYS_CONTEXT('USERENV', 'SESSION_USER');
```

- Sử dụng SET_CONTEXT để thiết lập một thuộc tính của bối cảnh

```
DBMS_SESSION.SET_CONTEXT(
  'empno_ctx', 'employee_id', emp_id);
```

BƯỚC 3: GỌI PACKAGE ĐÃ TẠO

TẠO MỘT TRIGGER ĐĂNG NHẬP GỌI TỚI THỦ TỤC
`SET_EMPNO_CTX_PKG.SET_EMPNO`

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

BƯỚC 4: KIỂM TRA THUỘC TÍNH BỐI CẢNH TRONG ỨNG DỤNG

Để trả về một giá trị thuộc tính, sử dụng câu lệnh:

```
sys_context('empno_ctx', 'employee_id')
```

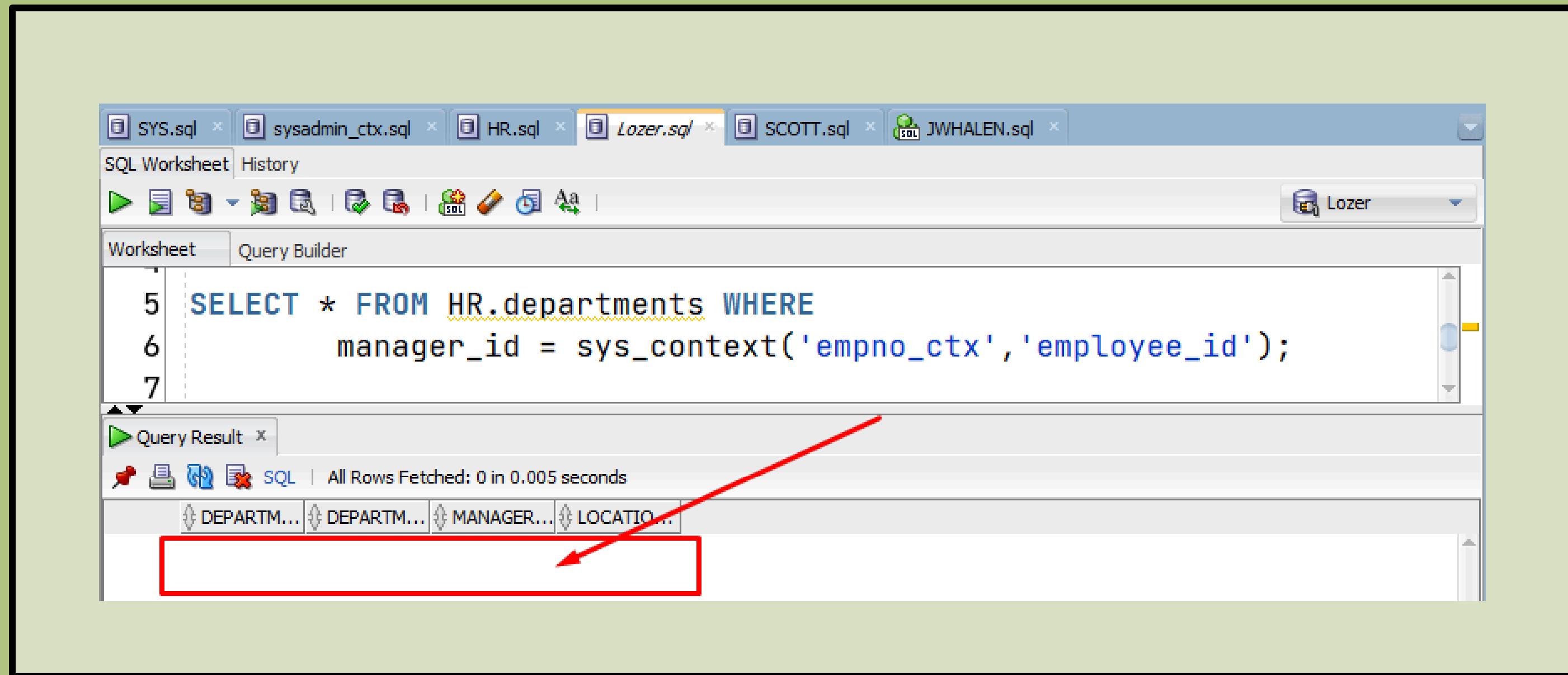
Câu lệnh gồm hai tham số:

- Tên của bối cảnh
- Tên của thuộc tính

Câu lệnh SELECT:

```
SELECT * FROM HR.departments  
WHERE manager_id =  
      sys_context('empno_ctx','employee_id');
```

KẾT QUẢ THỰC THI

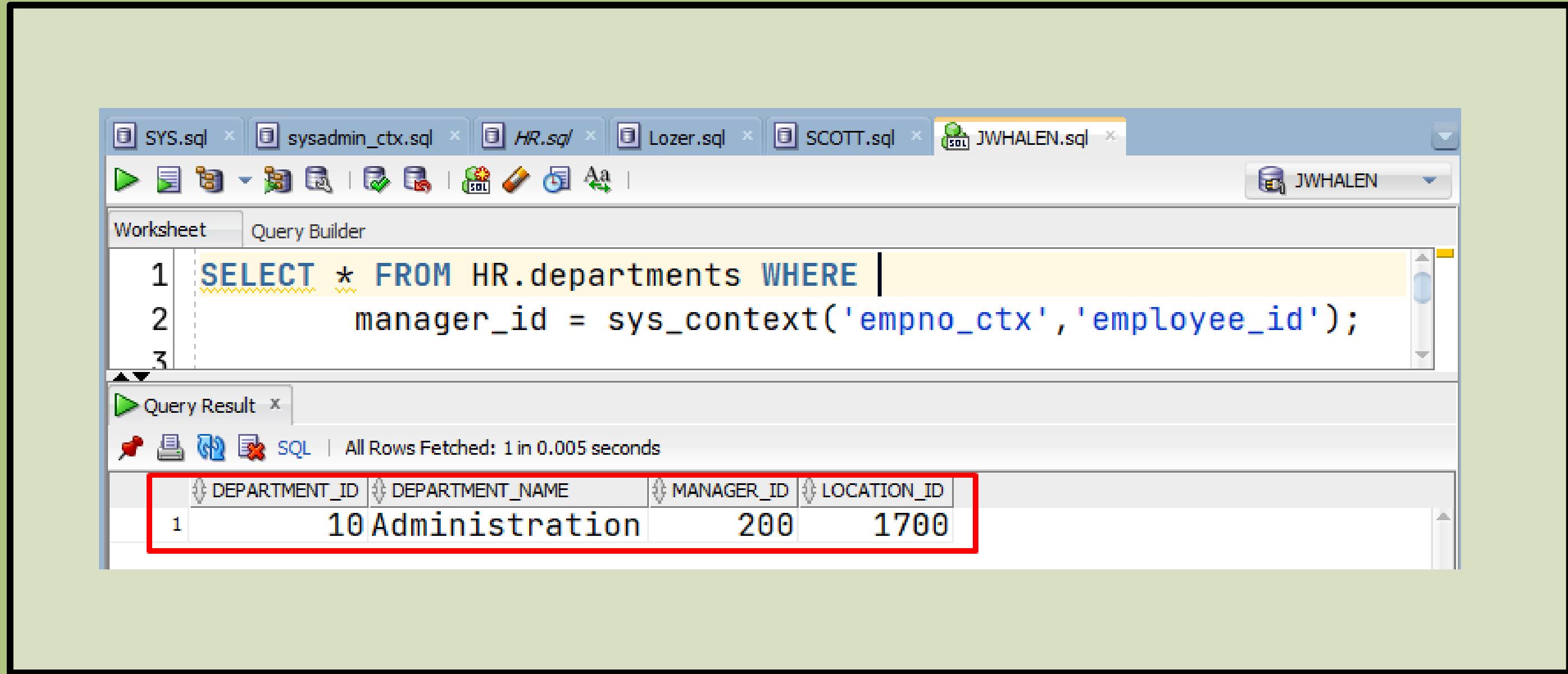


Screenshot of an Oracle SQL Worksheet interface. The title bar shows multiple open files: SYS.sql, sysadmin_ctx.sql, HR.sql, Lozer.sql (which is the active tab), SCOTT.sql, and JWHALEN.sql. The toolbar includes standard SQL and database management icons. The main area is divided into 'Worksheet' and 'Query Builder' tabs, with 'Worksheet' selected. The query window contains the following SQL code:

```
5  SELECT * FROM HR.departments WHERE
6      manager_id = sys_context('empno_ctx','employee_id');
7
```

The 'Query Result' window below shows the results of the query. The table has four columns: DEPARTMENT_ID, DEPARTMENT_NAME, MANAGER_ID, and LOCATION_ID. The MANAGER_ID column is highlighted with a red box, and a red arrow points from this box to the empty data row below it. The status bar at the bottom of the results window indicates: All Rows Fetched: 0 in 0.005 seconds.

KẾT QUẢ THỰC THI



The screenshot shows a SQL developer interface with the following details:

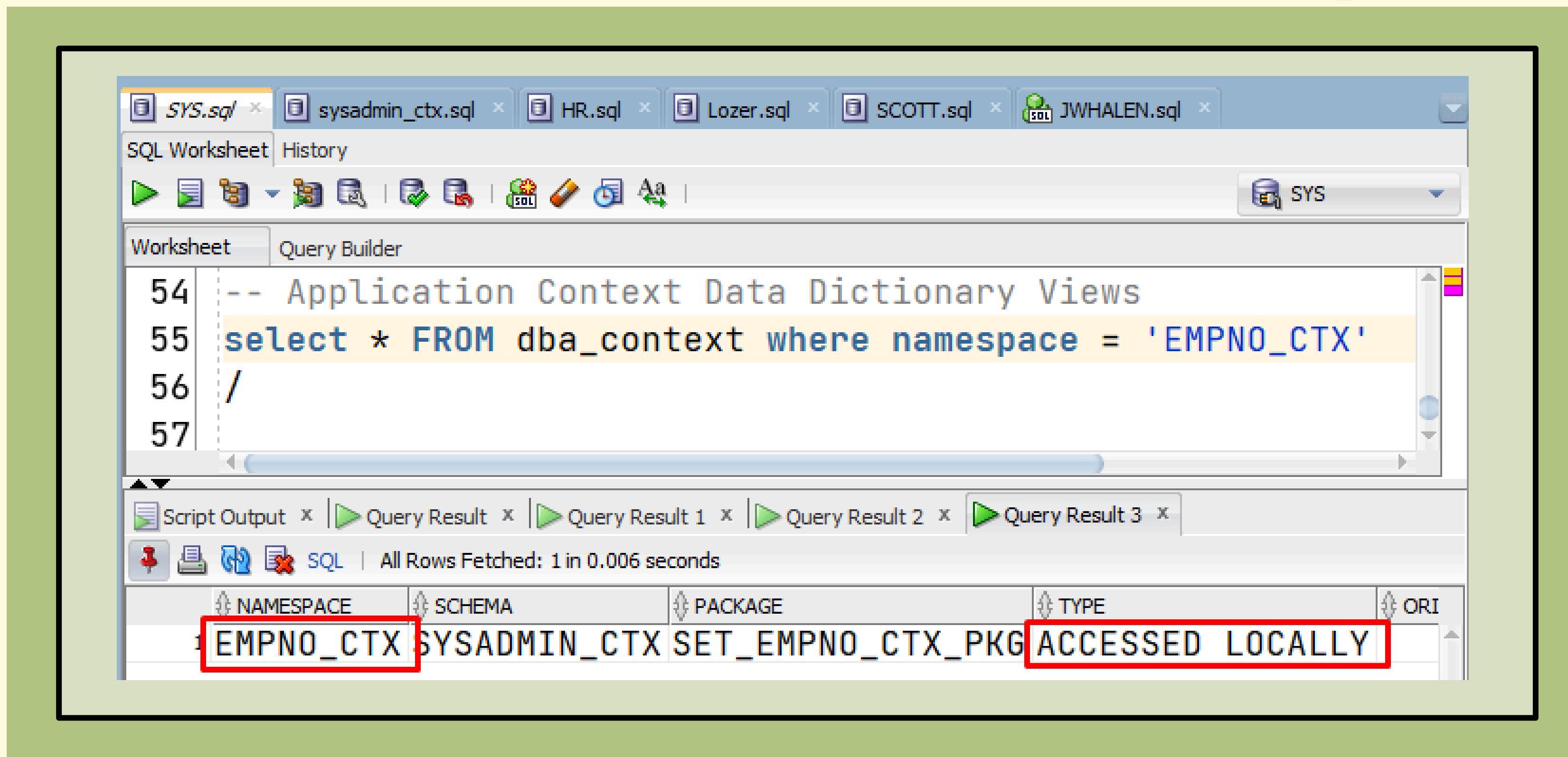
- Toolbar:** Includes icons for Run, Stop, Refresh, Paste, Copy, Paste Special, Find, Replace, and others.
- Tab Bar:** Shows multiple tabs: SYS.sql, sysadmin_ctx.sql, HR.sql, Lozer.sql, SCOTT.sql, and JWHALEN.sql (selected).
- Worksheet:** Displays the SQL query:

```
1 | SELECT * FROM HR.departments WHERE |
2 |       manager_id = sys_context('empno_ctx','employee_id');
```
- Query Result:** Shows the execution results in a table format:

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10 Administration	200	1700
- Message:** "All Rows Fetched: 1 in 0.005 seconds"

DATA DICTIONARY VIEWS

DBA_CONTEXT



The screenshot shows the Oracle SQL Developer interface. The top menu bar has tabs for 'SYS.sql', 'sysadmin_ctx.sql', 'HR.sql', 'Lozer.sql', 'SCOTT.sql', and 'JWHALEN.sql'. The 'SQL Worksheet' tab is selected. The toolbar below includes icons for running, saving, and zooming. The connection dropdown shows 'SYS'. The main workspace displays the following SQL code:

```
54 -- Application Context Data Dictionary Views
55 select * FROM dba_context where namespace = 'EMPNO_CTX'
56 /
57
```

The results pane shows the output of the query:

NAMESPACE	SCHEMA	PACKAGE	TYPE	ORI
EMPNO_CTX	SYSADMIN_CTX	SET_EMPNO_CTX_PKG	ACCESSED LOCALLY	

The 'NAMESPACE' and 'TYPE' columns are highlighted with red boxes.



2.2.1 LOCAL APPLICATION CONTEXT

2.2.2 SESSION-BASED APPLICATION CONTEXT

2.2.3 GLOBAL APPLICATION CONTEXTS

>>>

CATEGORIES OF DATABASE SESSION-BASED APPLICATION CONTEXTS

2.2.1.1 INITIALIZED EXTERNALLY

2.2.1.2 INITIALIZED GLOBALLY

CATEGORIES OF DATABASE SESSION-BASED APPLICATION CONTEXTS

2.2.1.1 INITIALIZED EXTERNALLY

2.2.1.2 INITIALIZED GLOBALLY



2.2.2.1 INITIALIZED EXTERNALLY

Lấy giá trị mặc định từ người dùng.

- Các giá trị mặc định có thể là gợi ý hoặc tuỳ chọn, trở nên đáng tin cậy sau khi xác thực.
- Chấp nhận việc khởi tạo các giá trị ngữ cảnh từ quy trình job queue.

Lấy giá trị từ các tài nguyên bên ngoài.

- Tự động truyền các giá trị ngữ cảnh trong namespace application context được khởi tạo bên ngoài
- Khôi phục các giá trị ngữ cảnh trong namespace application context được khởi tạo bên ngoài
- Cơ chế khởi tạo các giá trị ngữ cảnh nằm trong namespace application context được khởi tạo bên ngoài

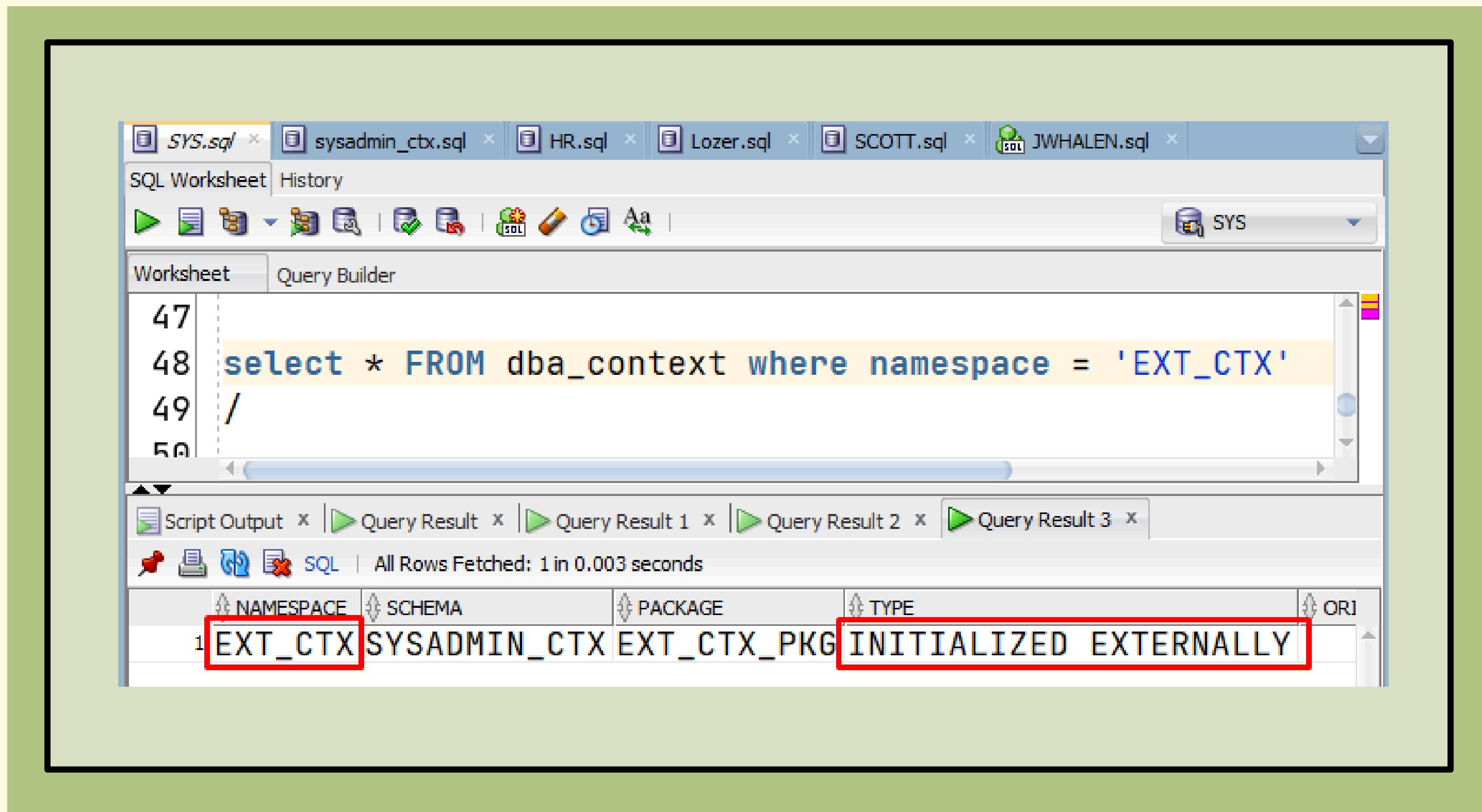
CREATE CONTEXT ext_ctx USING ext_ctx_pkg INITIALIZED EXTERNALLY;

Khởi tạo giá trị bối cảnh ứng dụng từ máy chủ trung gian.

- Các máy chủ trung gian có thể khởi tạo các giá trị ngữ cảnh, chẳng hạn như PROXY_USER, thay cho người dùng
- Cho phép xác định các chính sách truy cập dữ liệu an toàn, hạn chế truy cập trực tiếp vào cơ sở dữ liệu và áp dụng kết nối thông qua các ứng dụng hoặc proxy trung gian cụ thể.

DATA DICTIONARY VIEWS

DBA_CONTEXT



The screenshot shows the Oracle SQL Developer interface. The top menu bar has tabs for 'SYS.sql', 'sysadmin_ctx.sql', 'HR.sql', 'Lozer.sql', 'SCOTT.sql', and 'JWHALEN.sql'. The main window is a 'Worksheet' tab, showing the following SQL query:

```
47
48 select * FROM dba_context where namespace = 'EXT_CTX'
49 /
50
```

The results of the query are displayed in a table below:

1	2	3	4	5
NAMESPACE	SCHEMA	PACKAGE	TYPE	ORI
1 EXT_CTX	SYSADMIN_CTX	EXT_CTX_PKG	INITIALIZED EXTERNALLY	

The 'NAMESPACE' and 'TYPE' columns are highlighted with red boxes. The 'SCHEMA' column contains 'SYSADMIN_CTX' and the 'PACKAGE' column contains 'EXT_CTX_PKG'. The 'TYPE' column contains the value 'INITIALIZED EXTERNALLY'.

CATEGORIES OF DATABASE SESSION-BASED APPLICATION CONTEXTS

2.2.1.1 INITIALIZED EXTERNALLY

2.2.1.2 INITIALIZED GLOBALLY

Initialized Globally



About Initialized Globally

Giúp quản lý bối cảnh cho số lượng lớn người dùng và cơ sở dữ liệu dễ dàng hơn.



Initialized Globally with LDAP

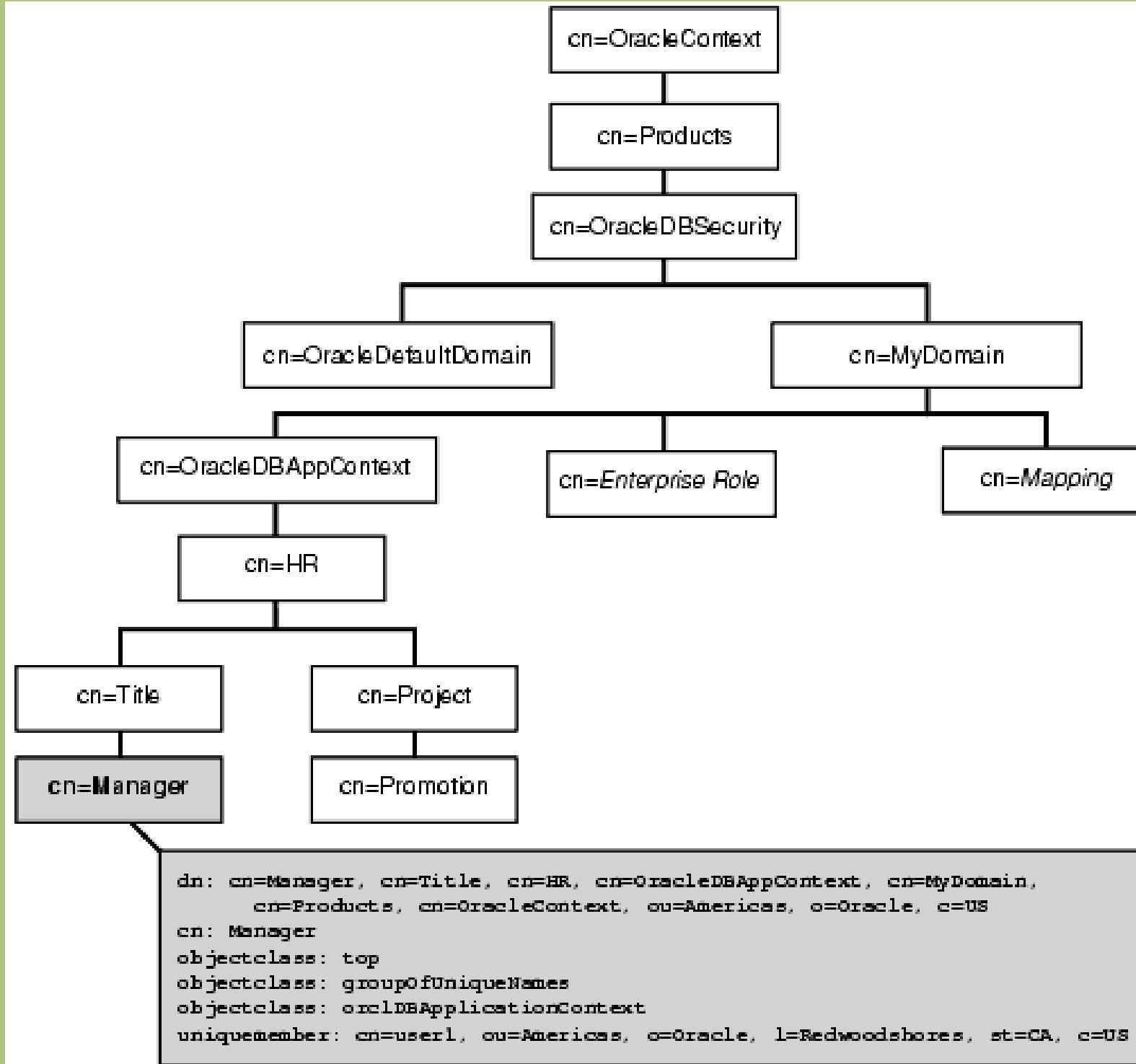
Một giao thức truy cập thư mục tiêu chuẩn, có thể mở rộng và hiệu quả



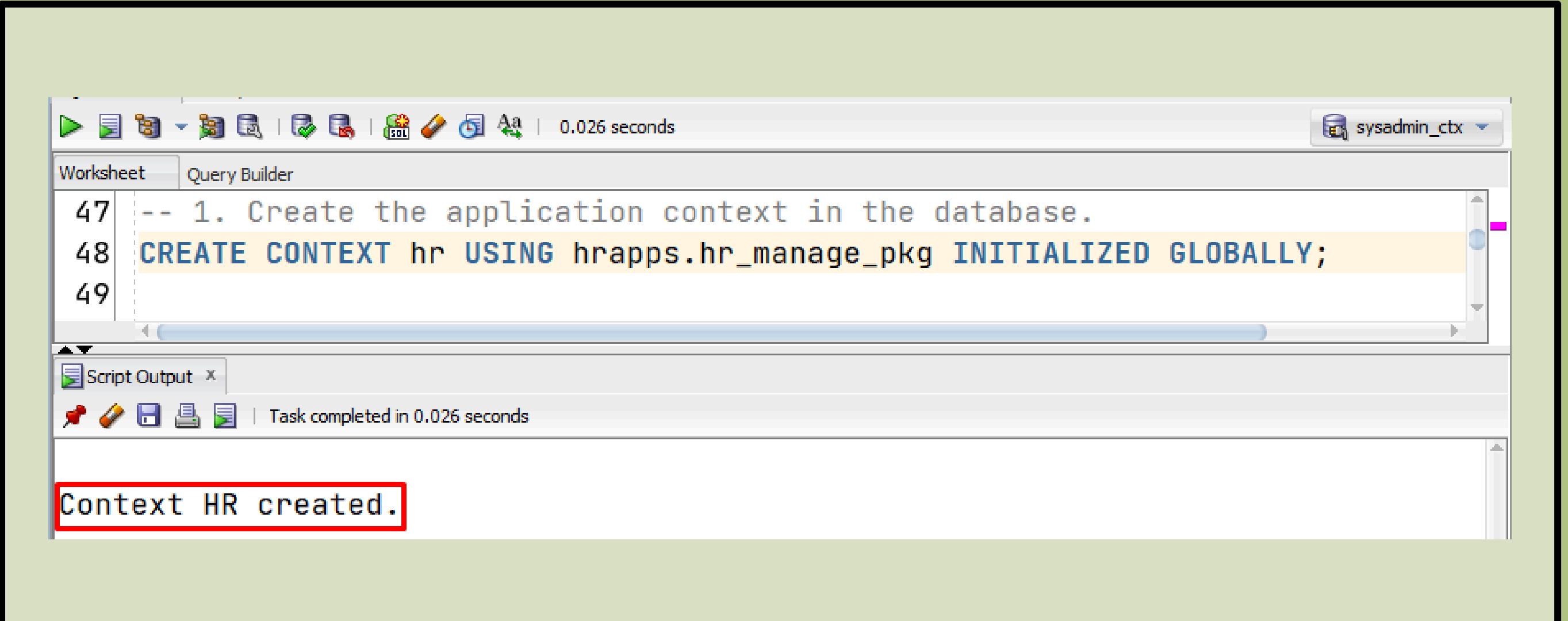
How it works?

Enterprise User Security

LOCATION OF APPLICATION CONTEXT IN LDAP DIRECTORY INFORMATION TREE (DIT)



INITIALIZING APPLICATION CONTEXT GLOBALLY



The screenshot shows the Oracle SQL Developer interface. The top menu bar includes 'File', 'Edit', 'Tools', 'Help', and a 'Database' dropdown set to 'sysadmin_ctx'. The toolbar contains various icons for database operations. The main area is a 'Worksheet' tab with the following SQL code:

```
47 -- 1. Create the application context in the database.
48 CREATE CONTEXT hr USING hrapps.hr_manage_pkg INITIALIZED GLOBALLY;
49
```

The code is executed, and the 'Script Output' tab shows the result:

```
Context HR created.
```

CREATE AND ADD NEW ENTRIES IN THE LDAP DIRECTORY.

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

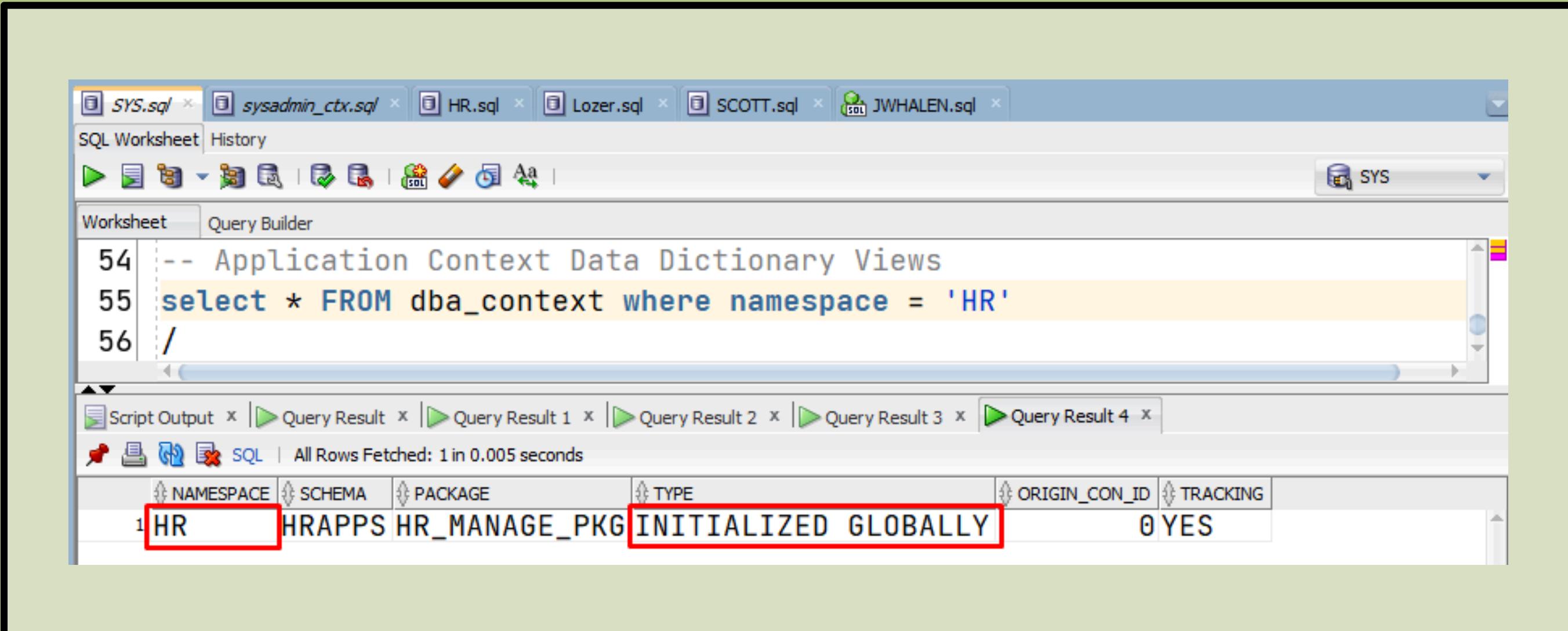
dn: cn=hr,cn=OracleDBApplicationContext,cn=example,cn=OracleDBSecurity,cn=Products,cn=OracleContext,ou=Americas,o=oracle,c=US
changetype: add
cn: hr
objectclass: top
objectclass: orclContainer

dn: cn=Title,cn=hr, cn=OracleDBApplicationContext,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=OracleDBApplicationContext,cn=example,cn=OracleDBSecurity,cn=Products,cn=OracleContext,ou=Americas,o=oracle,c=US
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
```

DATA DICTIONARY VIEWS

DBA_CONTEXT



The screenshot shows an Oracle SQL Worksheet interface. The top menu bar has tabs for 'SYS.sql', 'sysadmin_ctx.sql', 'HR.sql', 'Lozer.sql', 'SCOTT.sql', and 'JWHALEN.sql'. The 'Worksheet' tab is selected. The SQL code in the worksheet pane is:

```
54 -- Application Context Data Dictionary Views
55 select * FROM dba_context where namespace = 'HR'
56 /
```

The results pane shows a table with the following data:

1	NAMESPACE	SCHEMA	PACKAGE	TYPE	ORIGIN_CON_ID	TRACKING
1	HR	HRAPPS	HR_MANAGE_PKG	INITIALIZED GLOBALLY	0	YES

The 'NAMESPACE' and 'TYPE' columns are highlighted with red boxes.

A photograph of a person's hands resting on a laptop keyboard. The hands are positioned as if the person is about to type. The laptop is open, and the keyboard is clearly visible. The background is slightly blurred, and there are large, semi-transparent circles in various colors (yellow, green, blue) overlaid on the image, creating a modern, digital feel.

2.2.1 LOCAL APPLICATION CONTEXT

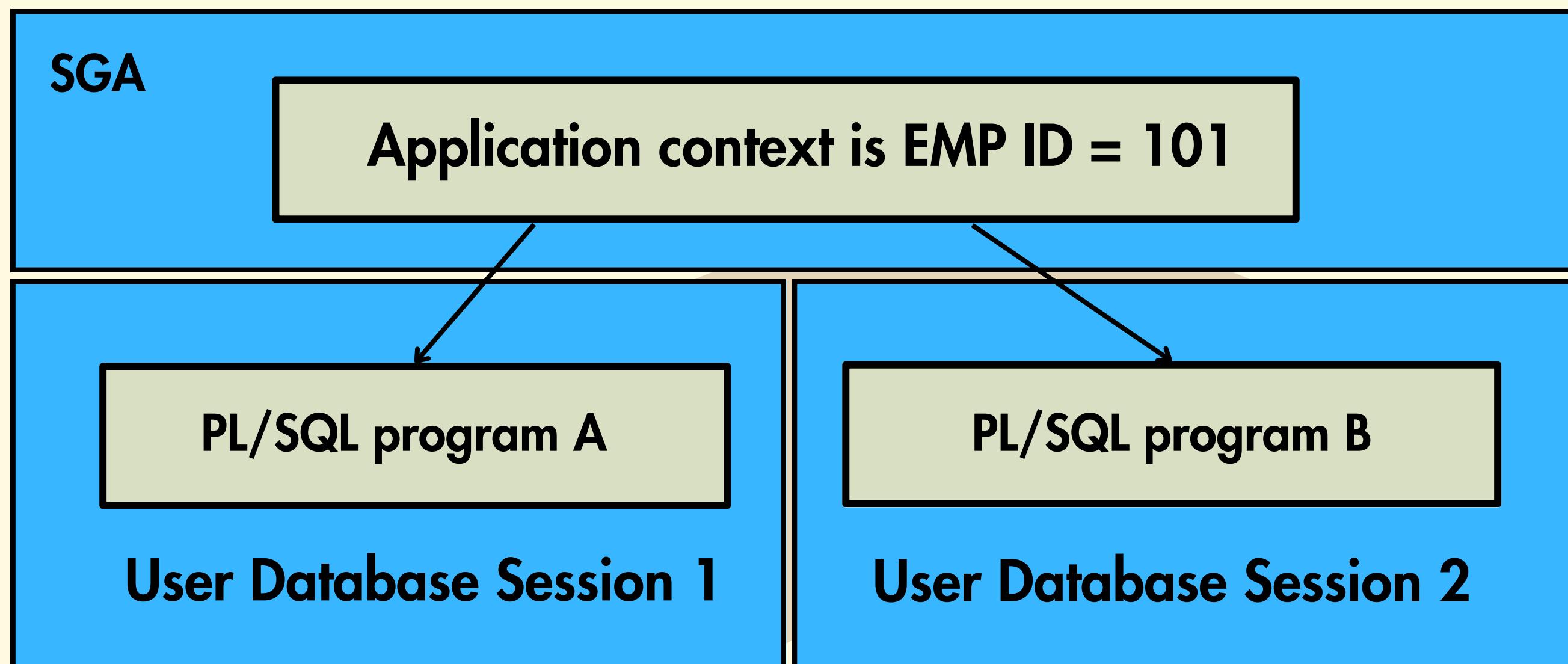
2.2.2 SESSION-BASED APPLICATION CONTEXT

2.2.3 GLOBAL APPLICATION CONTEXTS

>>>

2.2.3 APPLICATION CONTEXT ACCESSED GLOBALLY

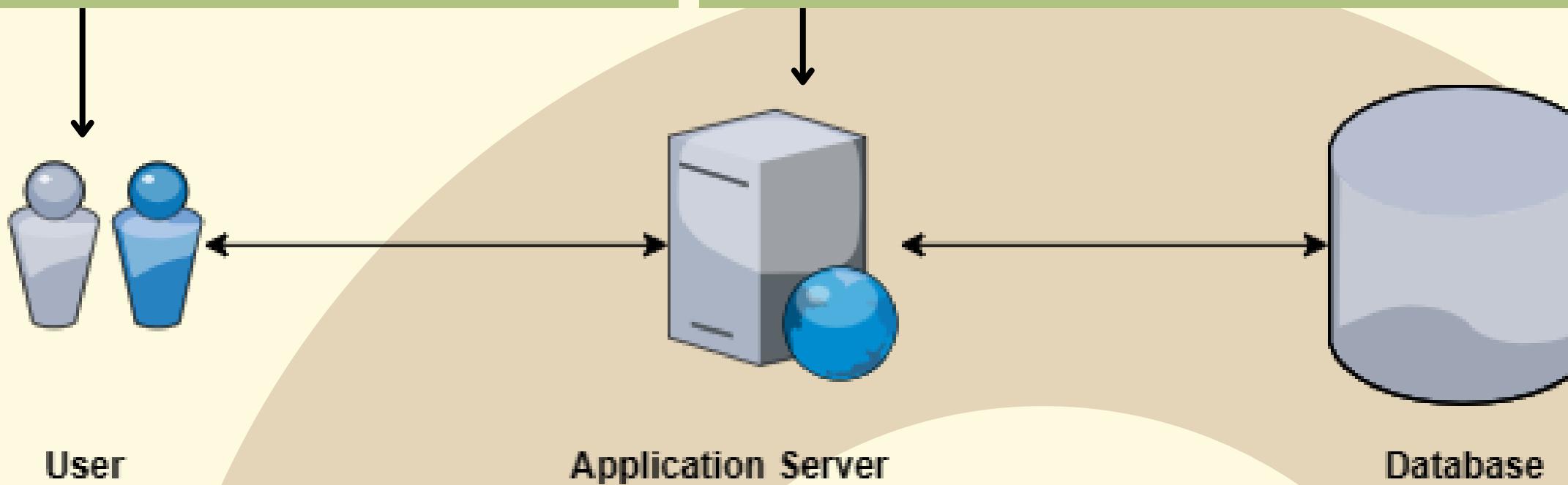
- Chia sẻ bối cảnh giữa các phiên
- Đơn giản hóa kết nối từ tầng trung gian
- Sử dụng mã định danh khách để xác định người dùng phiên



HOW THE APPLICATION CONTEXT ACCESSED GLOBALLY WORKS

2. Đăng nhập
6. Tạo request mới
8. Đăng xuất

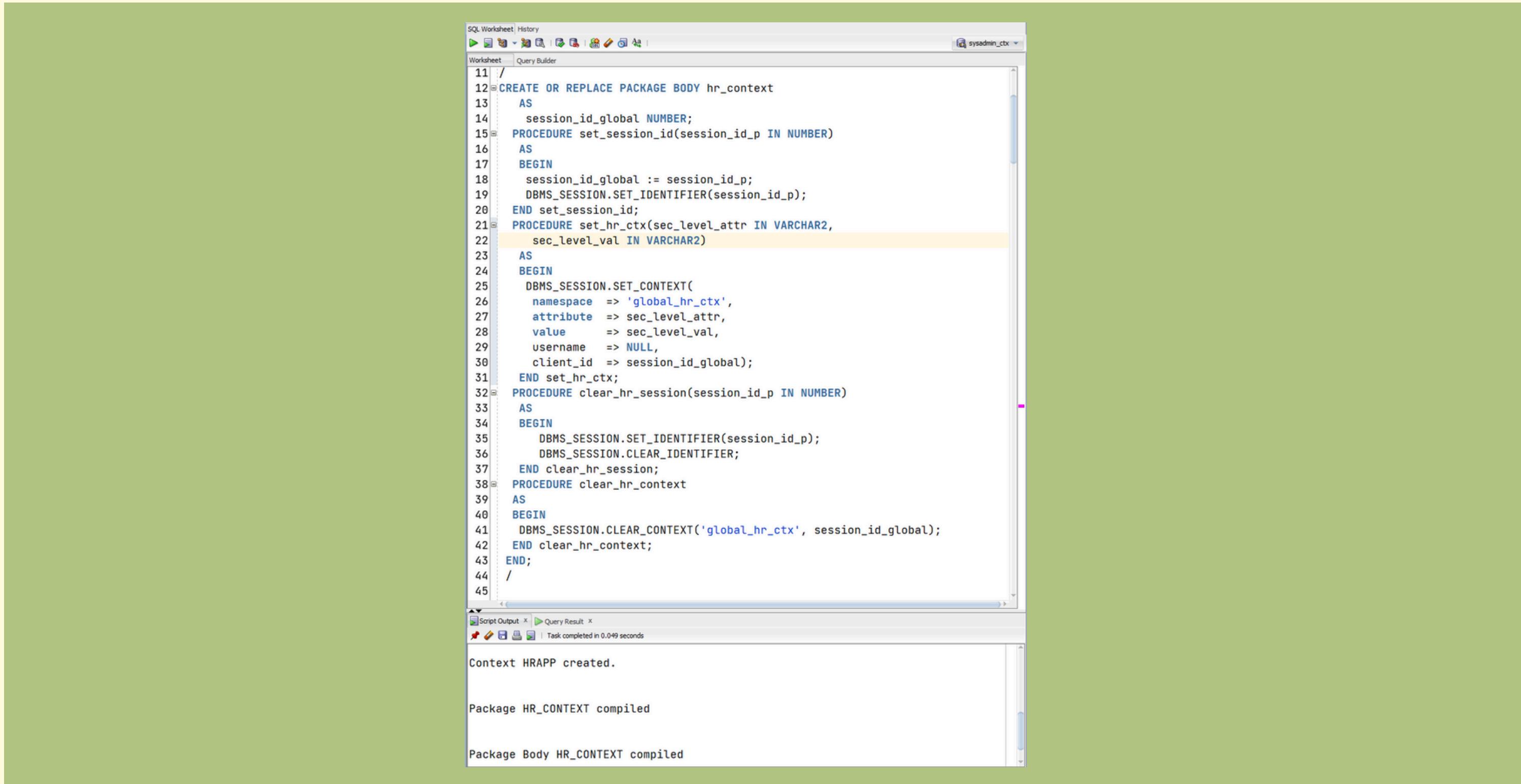
1. Xây dựng nhóm kết nối
3. Thiết lập phiên
4. Xử lý yêu cầu.
5. Hoàn tất yêu cầu
7. Xử lý yêu cầu thứ hai
9. Xoá bối cảnh



TRIỂN KHAI BỐI CẢNH ỨNG DỤNG TOÀN CỤC

- 1. Tạo bối cảnh ứng dụng được truy cập trên toàn cục.**
- 2. Sửa đổi chương trình thiết lập phiên:**
 - Thiết lập bối cảnh ứng dụng.
 - Thiết lập mã định danh phiên máy khách.
 - Xóa mã định danh khách hàng khi yêu cầu kết thúc.
- 3. Sửa đổi chương trình ứng dụng xử lý các request tiếp theo trong cùng một phiên:**
 - Thiết lập mã định danh phiên máy khách từ phiên đăng nhập.
 - Xóa mã định danh khách hàng khi yêu cầu kết thúc.
- 4. Tạo hoặc điều chỉnh chương trình ứng dụng để xóa bối cảnh khi kết thúc một phiên.**

TẠO BỐI CẢNH ỨNG DỤNG ĐƯỢC TRUY CẬP TRÊN TOÀN CỤC.



The screenshot shows an Oracle SQL Worksheet interface. The main area displays the following PL/SQL code:

```
11  /
12  CREATE OR REPLACE PACKAGE BODY hr_context
13  AS
14      session_id_global NUMBER;
15  PROCEDURE set_session_id(session_id_p IN NUMBER)
16  AS
17      BEGIN
18          session_id_global := session_id_p;
19          DBMS_SESSION.SET_IDENTIFIER(session_id_p);
20      END set_session_id;
21  PROCEDURE set_hr_ctx(sec_level_attr IN VARCHAR2,
22                      sec_level_val IN VARCHAR2)
23  AS
24      BEGIN
25          DBMS_SESSION.SET_CONTEXT(
26              namespace  => 'global_hr_ctx',
27              attribute  => sec_level_attr,
28              value      => sec_level_val,
29              username   => NULL,
30              client_id  => session_id_global);
31      END set_hr_ctx;
32  PROCEDURE clear_hr_session(session_id_p IN NUMBER)
33  AS
34      BEGIN
35          DBMS_SESSION.SET_IDENTIFIER(session_id_p);
36          DBMS_SESSION.CLEAR_IDENTIFIER;
37      END clear_hr_session;
38  PROCEDURE clear_hr_context
39  AS
40      BEGIN
41          DBMS_SESSION.CLEAR_CONTEXT('global_hr_ctx', session_id_global);
42      END clear_hr_context;
43  END;
44  /
45
```

The code is being compiled, as indicated by the output window at the bottom:

```
Context HRAPP created.

Package HR_CONTEXT compiled

Package Body HR_CONTEXT compiled
```

BƯỚC 1: TẠO BỐI CẢNH ỨNG DỤNG ĐƯỢC TRUY CẬP TRÊN TOÀN CỤC.

Tạo bối cảnh bằng cách:

```
CREATE CONTEXT hrapp  
USING hr_context ACCESSED GLOBALLY;
```

Mệnh đề ACCESSED GLOBALLY chỉ ra rằng ngữ cảnh có thể được truy cập từ nhiều phiên.

BƯỚC 2: THIẾT LẬP PHIÊN

1. Nhận một giá trị duy nhất để sử dụng làm mã định danh khách hàng
2. Thiết lập bối cảnh ứng dụng:

```
dbms_session.set_context  
(context, attr, value, username, client_id);
```

```
dbms_session.set_context  
('hrapp','id','phall','APPSMGR', 12345 );  
dbms_session.set_context  
('hrapp','dept','sales','APPSMGR', 12345 );
```

3. Đặt mã định danh phiên máy khách:

```
dbms_session.set_identifier( 12345 );
```

4. Lưu mã định danh khách hàng trong cookie.

BƯỚC 3: XỬ LÝ CÁC REQUEST TIẾP THEO

1. Truy xuất mã định danh máy khách
2. Thiết lập mã định danh khách hàng cho phiên:

```
dbms_session.set_identifier( 12345 );
```

3. Xóa mã định danh khách hàng khi yêu cầu kết thúc:

```
dbms_session.clear_identifier()
```

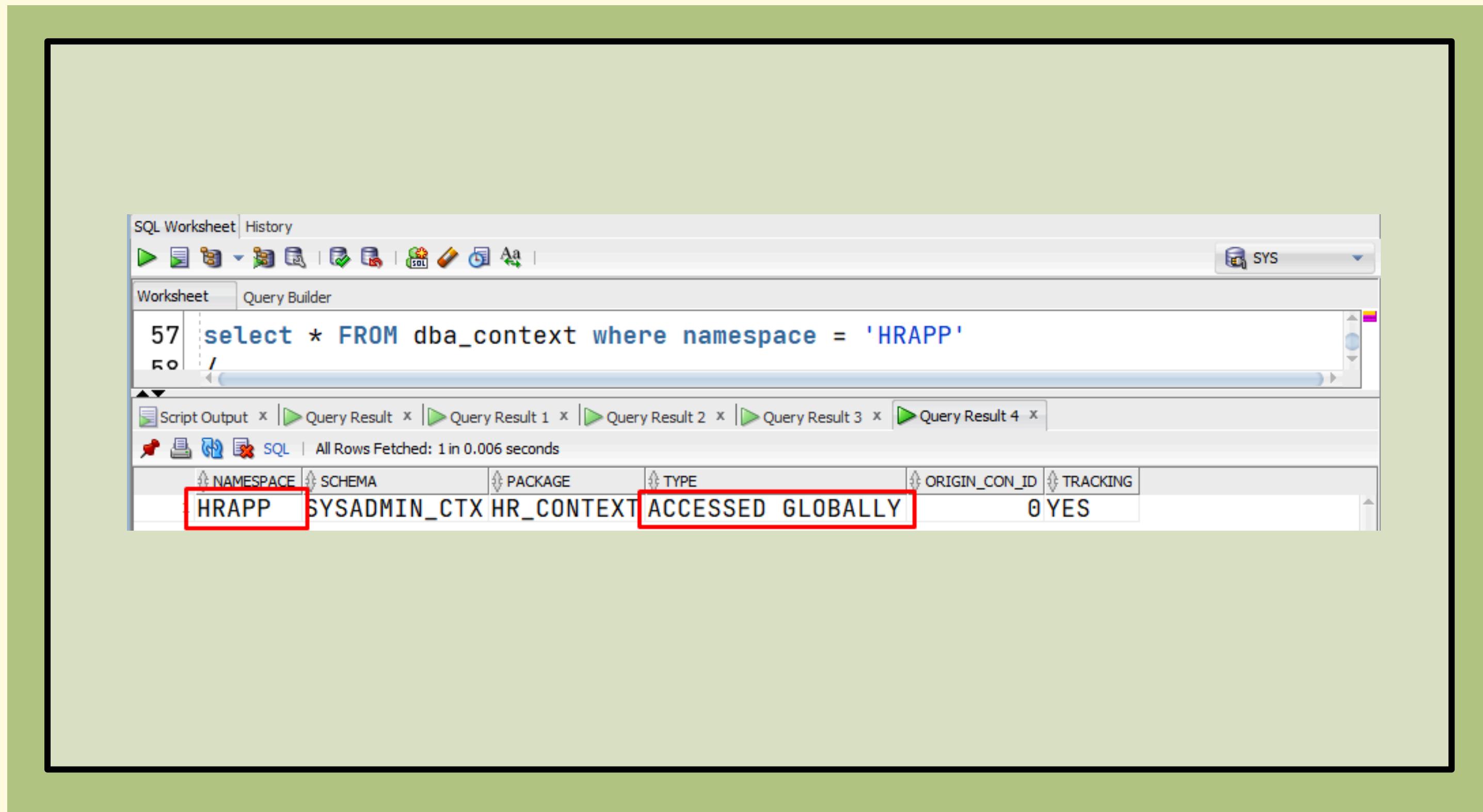
BƯỚC 4: KẾT THÚC PHIÊN

1. Truy xuất mã định danh máy khách
2. Xoá bối cảnh khi kết thúc phiên:

```
EXEC dbms_session.clear_context
  ('HRAPP', '12345');
```

DATA DICTIONARY VIEWS

DBA_CONTEXT



```
SQL Worksheet History
Worksheet Query Builder
57 | select * FROM dba_context where namespace = 'HRAPP'
58 | /
Script Output x | Query Result x | Query Result 1 x | Query Result 2 x | Query Result 3 x | Query Result 4 x
SQL | All Rows Fetched: 1 in 0.006 seconds
NAMESPACE SCHEMA PACKAGE TYPE ORIGIN_CON_ID TRACKING
HRAPP SYSADMIN_CTX HR_CONTEXT ACCESSED GLOBALLY 0 YES
```

2. APPLICATION CONTEXT

2.3 HOW TO USE APPLICATION CONTEXT FOR FINE-GRAINED ACCESS CONTROL



DEMO

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

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

CREATE OR REPLACE PACKAGE apps.oe_ctx AS
  PROCEDURE set_cust_num;
END;

CREATE OR REPLACE PACKAGE BODY apps.oe_ctx AS
  PROCEDURE set_cust_num IS
    custnum NUMBER;
    BEGIN
      SELECT cust_no INTO custnum FROM customers WHERE cust_name =
        SYS_CONTEXT('USERENV', 'SESSION_USER');
      /* SET cust_num attribute in 'order_entry' context */
      DBMS_SESSION.SET_CONTEXT('order_entry', 'cust_num', custnum);
      DBMS_SESSION.SET_CONTEXT('order_entry', 'cust_num', custnum);
    END set_cust_num;
END;
```

DEMO

Bước 2: Tạo ngữ cảnh ứng dụng

```
CREATE CONTEXT Order_entry USING apps.oe_ctx;
```

DEMO

Bước 3: Truy cập ngữ cảnh ứng dụng bên trong gói

```
CREATE OR REPLACE PACKAGE Oe_security AS
  FUNCTION Custnum_sec (D1 VARCHAR2, D2 VARCHAR2)
  RETURN VARCHAR2;
END;
```

DEMO

Bước 3: Truy cập ngữ cảnh ứng dụng bên trong gói

```
3 CREATE OR REPLACE PACKAGE BODY Oe_security AS
4
5     /* limits select statements based on customer number: */
6     FUNCTION Custnum_sec (D1 VARCHAR2, D2 VARCHAR2) RETURN VARCHAR2
7     IS
8         D_predicate VARCHAR2 (2000);
9         BEGIN
10            D_predicate := 'cust_no = SYS_CONTEXT("order_entry", "cust_num")';
11            RETURN D_predicate;
12        END Custnum_sec;
13    END Oe_security;
```

DEMO

Bước 4: Tạo chính sách bảo mật mới

```
CONNECT sys/11052003 AS sysdba;
ALTER SESSION SET "_ORACLE_SCRIPT" = TRUE;
CREATE USER secusr IDENTIFIED BY secusr;
GRANT CONNECT TO secusr;
ALTER SESSION SET "_ORACLE_SCRIPT" = FALSE;
```

DEMO

Bước 4: Tạo chính sách bảo mật mới

```
|- BEGIN
  DBMS_RLS.ADD_POLICY(
    OBJECT_SCHEMA    => 'scott',
    OBJECT_NAME      => 'orders_tab',
    POLICY_NAME      => 'oe_policy',
    FUNCTION_SCHEMA  => 'secusr',
    POLICY_FUNCTION  => 'oe_security.custnum_sec',
    STATEMENT_TYPES  => 'select'
  );
END;
```

2.APPLICATION CONTEXT

2.4 CHECK THE POLICY EFFECT

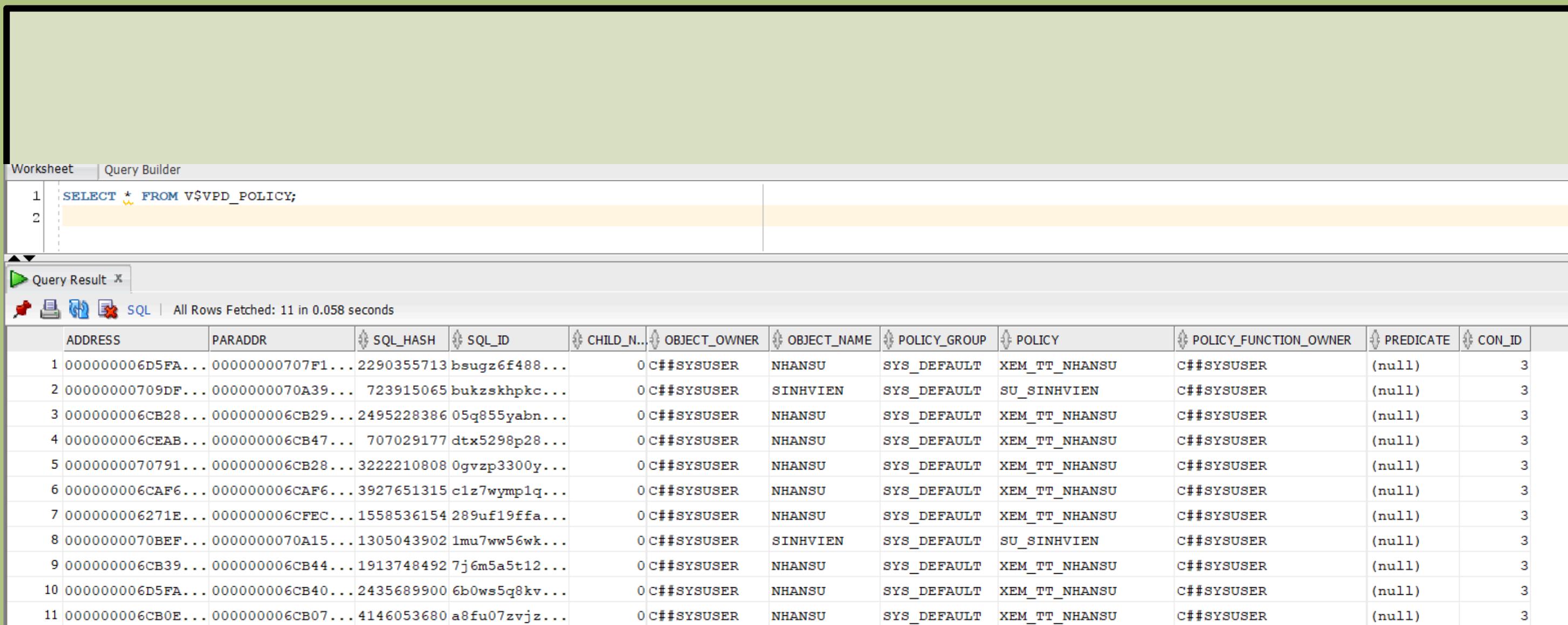


2.4 CHECK THE POLICY EFFECT

Column	Datatype	Description
ADDRESS	RAW (4 8)	Cursor address
PARADDR	RAW (4 8)	Parent cursor address
SQL_HASH	NUMBER	SQL hash number
SQL_ID	VARCHAR2 (13)	SQL identifier
CHILD_NUMBER	NUMBER	Cursor's child number under the parent
OBJECT_OWNER	VARCHAR2 (128)	Owner of the object with the policy
OBJECT_NAME	VARCHAR2 (128)	Name of the object with the policy
POLICY_GROUP	VARCHAR2 (128)	Name of the policy group
POLICY	VARCHAR2 (128)	Name of the policy
POLICY_FUNCTION_OWNER	VARCHAR2 (128)	Owner of the policy function
PREDICATE	VARCHAR2 (4000)	Predicate for the policy (truncated to 4000 bytes in length)
CON_ID	NUMBER	<p>The ID of the container to which the data pertains. Possible values include:</p> <ul style="list-style-type: none">• 0: This value is used for rows containing data that pertain to the entire CDB. This value is also used for rows in non-CDBs.• 1: This value is used for rows containing data that pertain to only the root• <i>n</i>: Where <i>n</i> is the applicable container ID for the rows containing data



2.4 CHECK THE POLICY EFFECT



The screenshot shows the Oracle SQL Developer interface. The top bar has tabs for 'Worksheet' and 'Query Builder', with 'Worksheet' selected. The main area is a 'Query Result' window. The query executed is:

```
1 | SELECT * FROM V$VPD_POLICY;
```

The results show 11 rows of data, each representing a policy entry. The columns are:

ADDRESS	PARADDR	SQL_HASH	SQL_ID	CHILD_N...	OBJECT_OWNER	OBJECT_NAME	POLICY_GROUP	POLICY	POLICY_FUNCTION_OWNER	PREDICATE	CON_ID
1 000000006D5FA...	00000000707F1...	2290355713	bsugz6f488...		0C##SYSUSER	NHANSU	SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
2 00000000709DF...	0000000070A39...	723915065	bukzskhpkc...		0C##SYSUSER	SINHVIEN	SYS_DEFAULT	SU_SINHVIEN	C##SYSUSER	(null)	3
3 000000006CB28...	000000006CB29...	2495228386	05q855yabn...		0C##SYSUSER	NHANSU	SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
4 000000006CEAB...	000000006CB47...	707029177	dtx5298p28...		0C##SYSUSER	NHANSU	SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
5 0000000070791...	000000006CB28...	3222210808	0gvzp3300y...		0C##SYSUSER	NHANSU	SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
6 000000006CAF6...	000000006CAF6...	3927651315	c1z7wymp1q...		0C##SYSUSER	NHANSU	SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
7 000000006271E...	000000006CFEC...	1558536154	289uf19ffa...		0C##SYSUSER	NHANSU	SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
8 0000000070BEF...	0000000070A15...	1305043902	1mu7ww56wk...		0C##SYSUSER	SINHVIEN	SYS_DEFAULT	SU_SINHVIEN	C##SYSUSER	(null)	3
9 000000006CB39...	000000006CB44...	1913748492	7j6m5a5t12...		0C##SYSUSER	NHANSU	SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
10 000000006D5FA...	000000006CB40...	2435689900	6b0ws5q8kv...		0C##SYSUSER	NHANSU	SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3
11 000000006CB0E...	000000006CB07...	4146053680	a8fu07zvjjz...		0C##SYSUSER	NHANSU	SYS_DEFAULT	XEM_TT_NHANSU	C##SYSUSER	(null)	3



THANK
YOU

