Nama	: Afina Putri Dayanti
NIM	: 825200049
Jurusan	: Sistem Informasi
Mata Kuliah	: Database Design and Management (Praktikum)

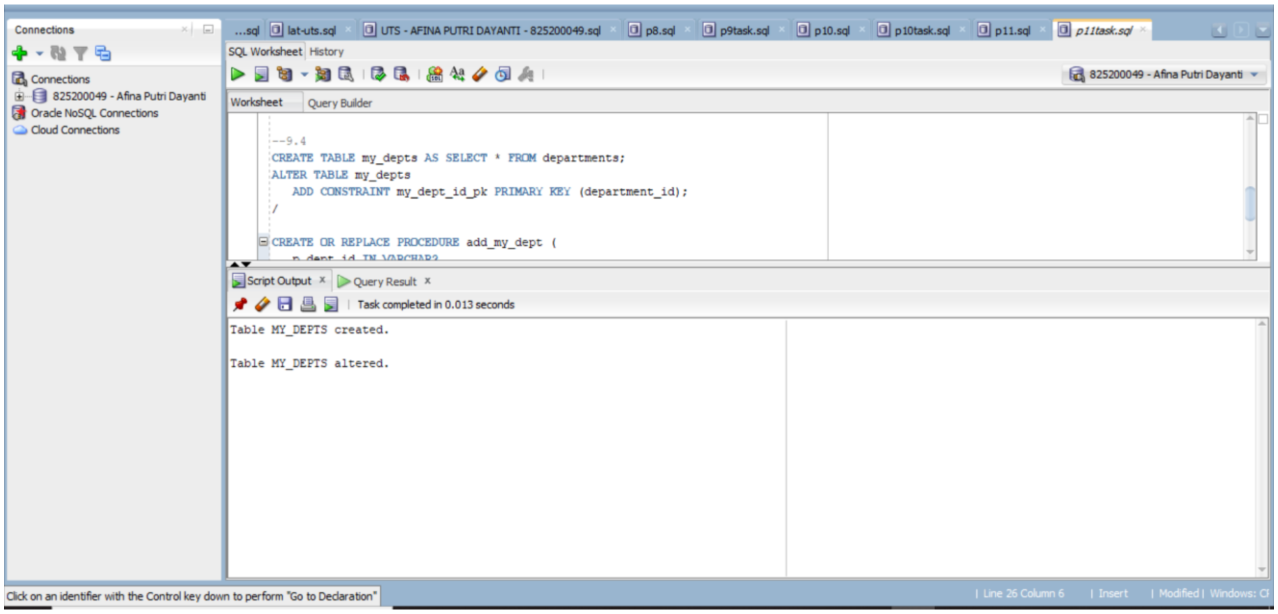
Vocabulary

Identify the vocabulary word for each definition below:

USER_SOURCE	The dictionary table that contains source code for all of the subprograms that you own.
USER_OBJECTS	The dictionary table that contains the names and types of procedures and functions that you own.
ALL_SOURCE	The dictionary table that contains source code for subprograms that are owned by others who have granted you the EXECUTE privilege.

Try It / Solve It

1. Complete the steps below to see how exceptions are propagated.
   a. Execute the following two SQL statements to create a duplicate of the DEPARTMENTS table, with department_id as the primary key.
   CREATE TABLE my_depts AS SELECT * FROM departments;
   ALTER TABLE my_depts
     ADD CONSTRAINT my_dept_id_pk PRIMARY KEY (department_id);
   Answer :



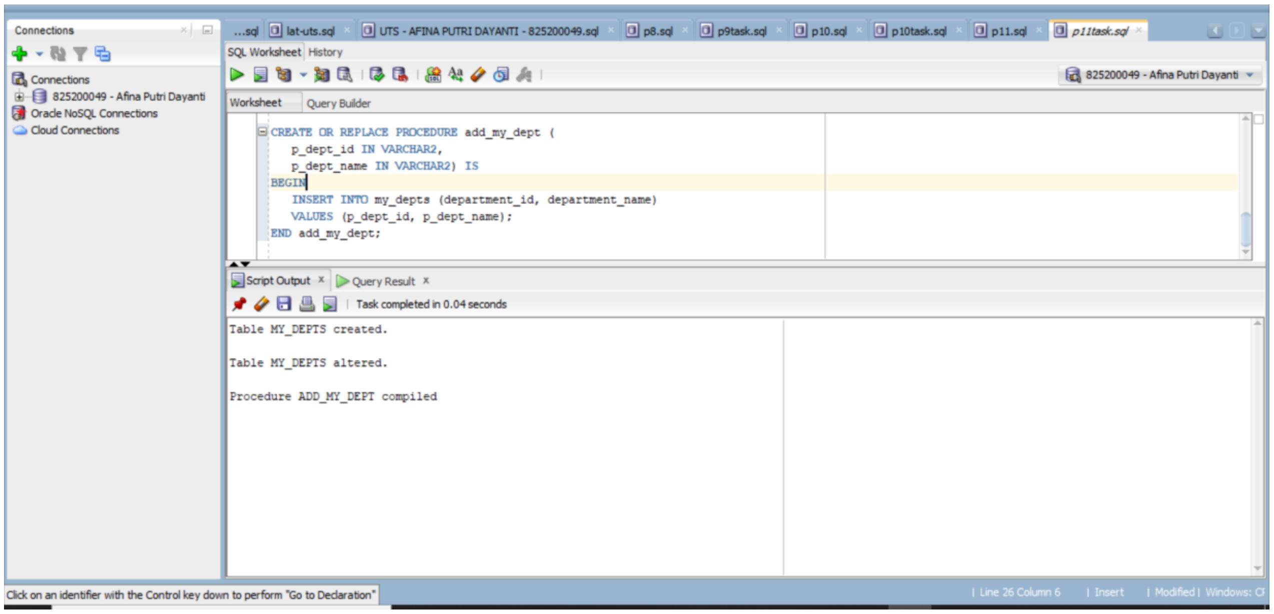
```
--9.4
CREATE TABLE my_depts AS SELECT * FROM departments;
ALTER TABLE my_depts
   ADD CONSTRAINT my_dept_id_pk PRIMARY KEY (department_id);
/

CREATE OR REPLACE PROCEDURE add_my_dept (
```

Task completed in 0.013 seconds

```
Table MY_DEPTS created.

Table MY_DEPTS altered.
```

   b. Examine the following code and create the procedure. Save your work (you will need to modify the procedure code later).
   CREATE OR REPLACE PROCEDURE add_my_dept (
     p_dept_id IN VARCHAR2,
     p_dept_name IN VARCHAR2) IS
   BEGIN

    INSERT INTO my_depts (department_id, department_name)
    VALUES (p_dept_id, p_dept_name);
END add_my_dept;
Answer :

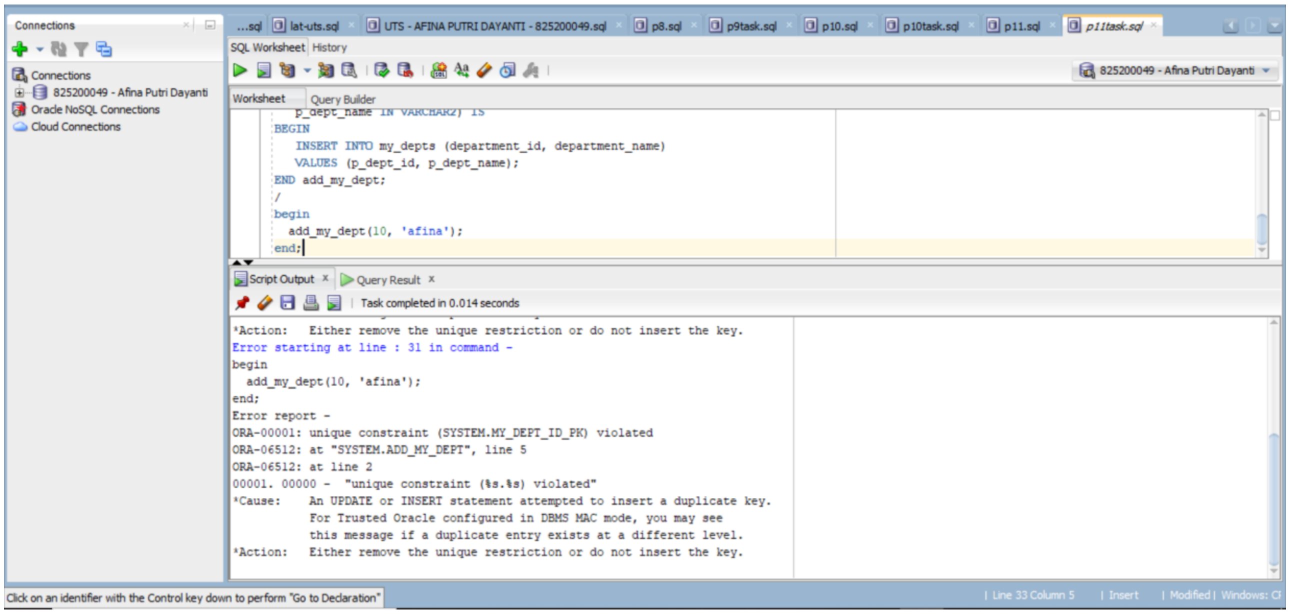


```
CREATE OR REPLACE PROCEDURE add_my_dept (
   p_dept_id IN VARCHAR2,
   p_dept_name IN VARCHAR2) IS
BEGIN
   INSERT INTO my_depts (department_id, department_name)
   VALUES (p_dept_id, p_dept_name);
END add_my_dept;
```

```
Table MY_DEPTS created.

Table MY_DEPTS altered.

Procedure ADD_MY_DEPT compiled
```

c. What do you think would happen if you execute this procedure to insert department_id 10 (which already exists)? Write and execute an anonymous block to test your theory.
Answer : It will give an error.

```
begin
  add_my_dept(10, 'afina');
end;
```



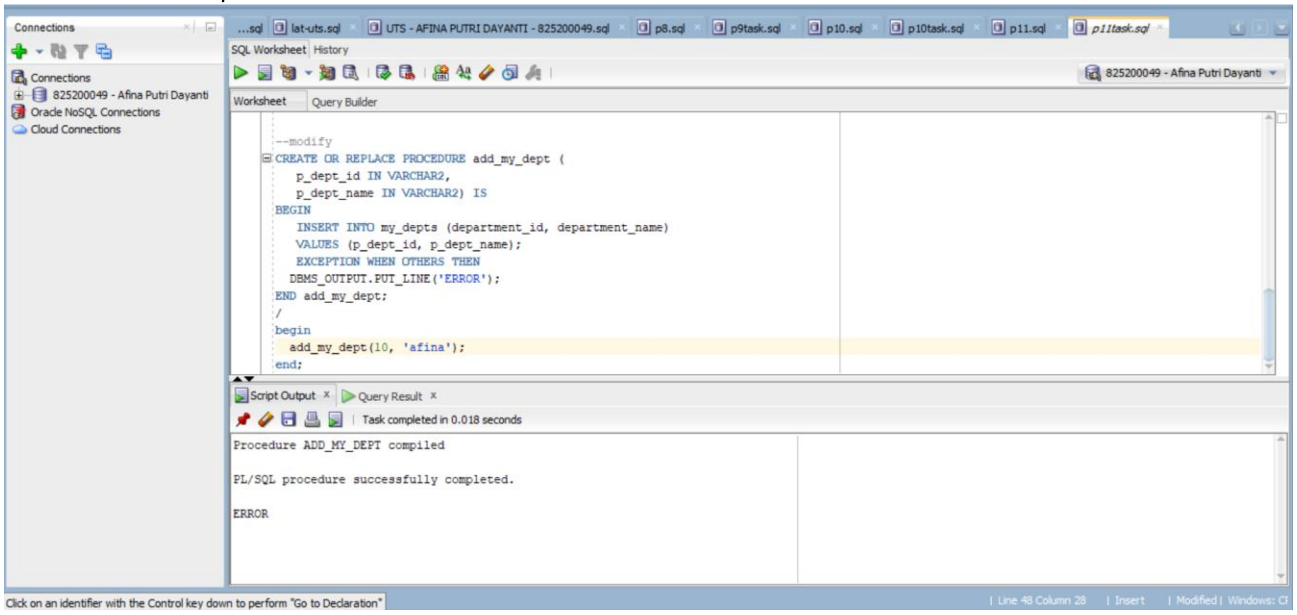
```
   p_dept_name IN VARCHAR2) IS
BEGIN
   INSERT INTO my_depts (department_id, department_name)
   VALUES (p_dept_id, p_dept_name);
END add_my_dept;
/
begin
  add_my_dept(10, 'afina');
end;
```

```
*Action:   Either remove the unique restriction or do not insert the key.
Error starting at line : 31 in command -
begin
  add_my_dept(10, 'afina');
end;
Error report -
ORA-00001: unique constraint (SYSTEM.MY_DEPT_ID_PK) violated
ORA-06512: at "SYSTEM.ADD_MY_DEPT", line 5
ORA-06512: at line 2
00001. 00000 -  "unique constraint (%s.%s) violated"
*Cause:    An UPDATE or INSERT statement attempted to insert a duplicate key.
           For Trusted Oracle configured in DBMS MAC mode, you may see
           this message if a duplicate entry exists at a different level.
*Action:   Either remove the unique restriction or do not insert the key.
```

d. Modify your procedure to handle the exception in a generic WHEN OTHERS exception handler.
Answer :

```
CREATE OR REPLACE PROCEDURE add_my_dept (
    p_dept_id IN VARCHAR2,
    p_dept_name IN VARCHAR2) IS
  BEGIN
    INSERT INTO my_depts (department_id, department_name)
    VALUES (p_dept_id, p_dept_name);
    EXCEPTION WHEN OTHERS THEN
   DBMS_OUTPUT.PUT_LINE('ERROR');
  END add_my_dept;
```

e. Now what do you think would happen if you execute this procedure for department_id 10 (which already exists)? Test it again as in step C.
Answer : The exception was executed "ERROR"



```
--modify
CREATE OR REPLACE PROCEDURE add_my_dept (
   p_dept_id IN VARCHAR2,
   p_dept_name IN VARCHAR2) IS
BEGIN
   INSERT INTO my_depts (department_id, department_name)
   VALUES (p_dept_id, p_dept_name);
   EXCEPTION WHEN OTHERS THEN
  DBMS_OUTPUT.PUT_LINE('ERROR');
END add_my_dept;
/
begin
  add_my_dept(10, 'afina');
end;
```

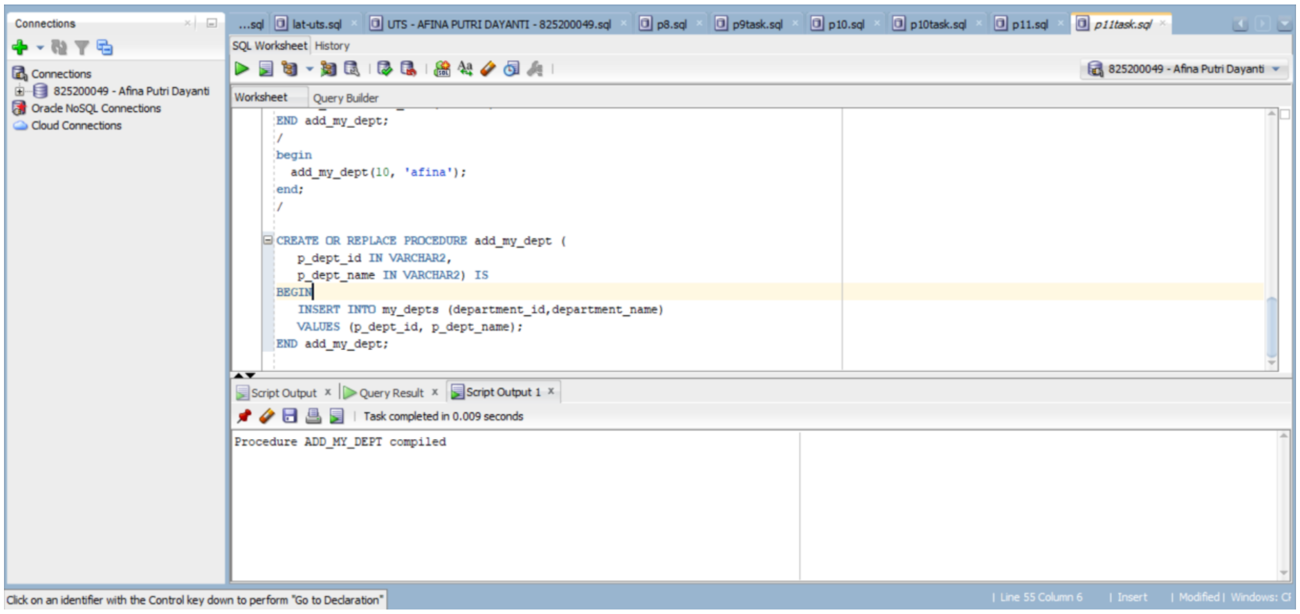
Task completed in 0.018 seconds

```
Procedure ADD_MY_DEPT compiled

PL/SQL procedure successfully completed.

ERROR
```

f. Modify the procedure code to leave out the exception section again. Run the code.

```
CREATE OR REPLACE PROCEDURE add_my_dept (
  p_dept_id IN VARCHAR2,
  p_dept_name IN VARCHAR2) IS
BEGIN
  INSERT INTO my_depts (department_id,department_name)
  VALUES (p_dept_id, p_dept_name);
END add_my_dept;
```

Answer :



```
END add_my_dept;
/
begin
  add_my_dept(10, 'afina');
end;
/

CREATE OR REPLACE PROCEDURE add_my_dept (
   p_dept_id  IN VARCHAR2,
   p_dept_name IN VARCHAR2) IS
BEGIN
   INSERT INTO my_depts (department_id,department_name)
   VALUES (p_dept_id, p_dept_name);
END add_my_dept;
```

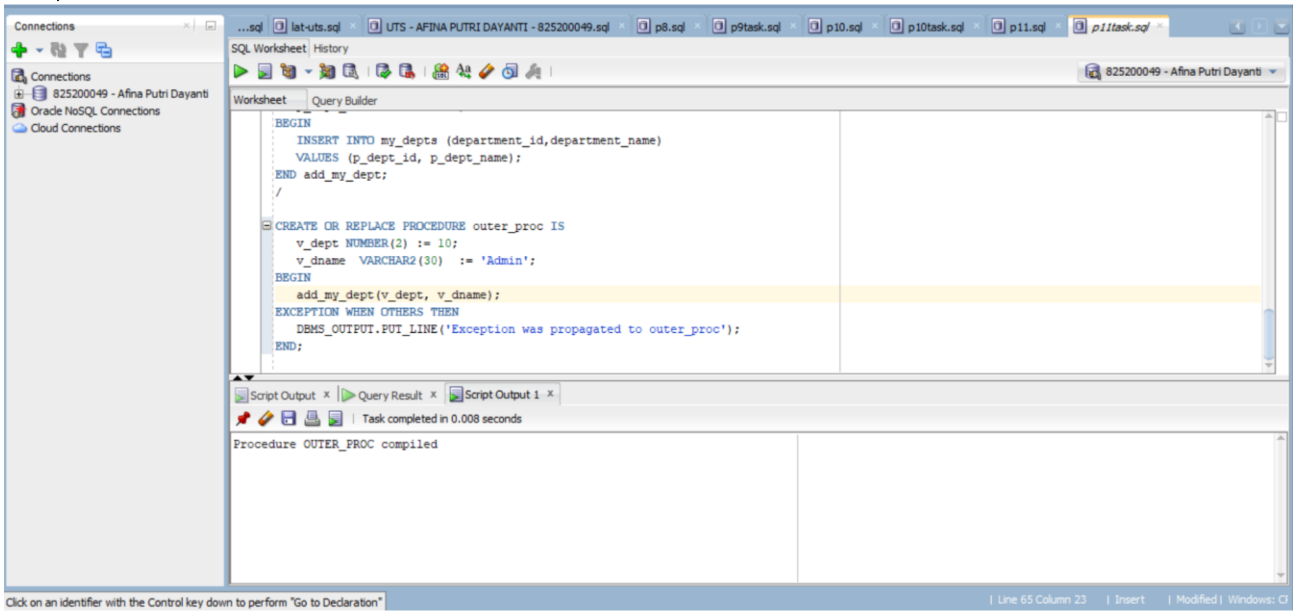
Task completed in 0.009 seconds

```
Procedure ADD_MY_DEPT compiled
```

Click on an identifier with the Control key down to perform "Go to Declaration"

g. Execute the following code to create a new procedure called outer_proc which calls add_my_dept, passing department_id 10 to it:

```
CREATE OR REPLACE PROCEDURE outer_proc IS
  v_dept      NUMBER(2)      := 10;
  v_dname  VARCHAR2(30)  := 'Admin';
BEGIN
  add_my_dept(v_dept, v_dname);
EXCEPTION WHEN OTHERS THEN
  DBMS_OUTPUT.PUT_LINE('Exception was propagated to outer_proc');
END;
```



```
BEGIN
   INSERT INTO my_depts (department_id,department_name)
   VALUES (p_dept_id, p_dept_name);
END add_my_dept;
/

CREATE OR REPLACE PROCEDURE outer_proc IS
   v_dept NUMBER(2)  := 10;
   v_dname   VARCHAR2(30)   := 'Admin';
BEGIN
   add_my_dept(v_dept, v_dname);
EXCEPTION WHEN OTHERS THEN
   DBMS_OUTPUT.PUT_LINE('Exception was propagated to outer_proc');
END;
```

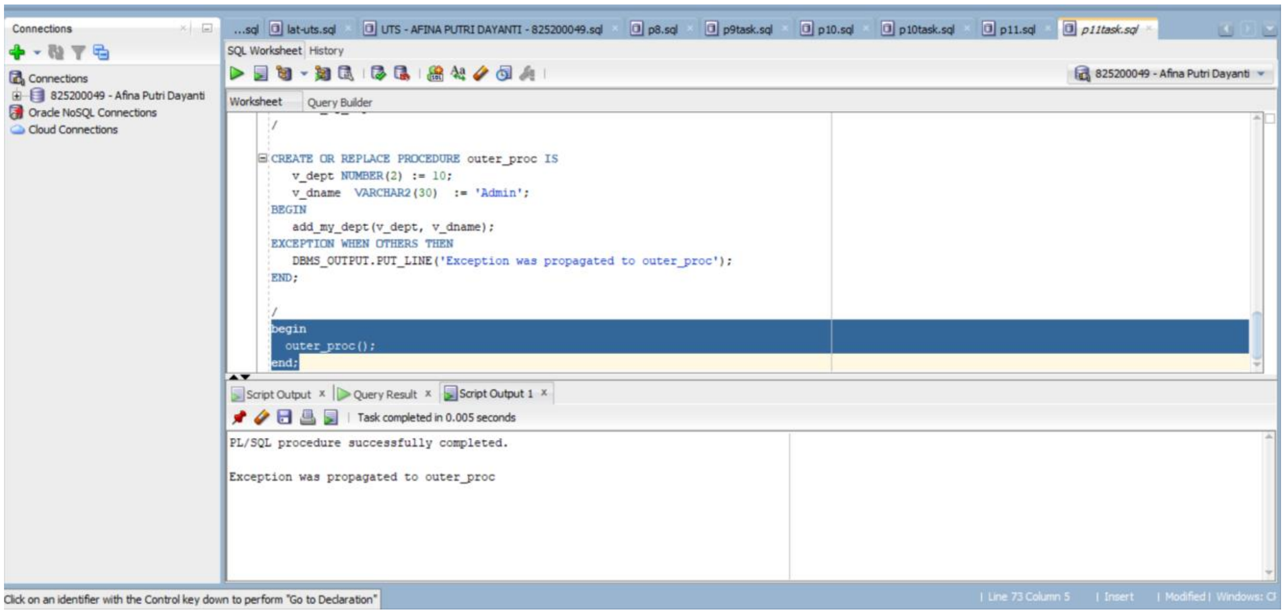
Task completed in 0.008 seconds

```
Procedure OUTER_PROC compiled
```

Click on an identifier with the Control key down to perform "Go to Declaration"

h. Execute outer_proc from an anonymous block. What happens and why?

Answer : The exception from add_my_dept was propagated to outer_proc and handled there.



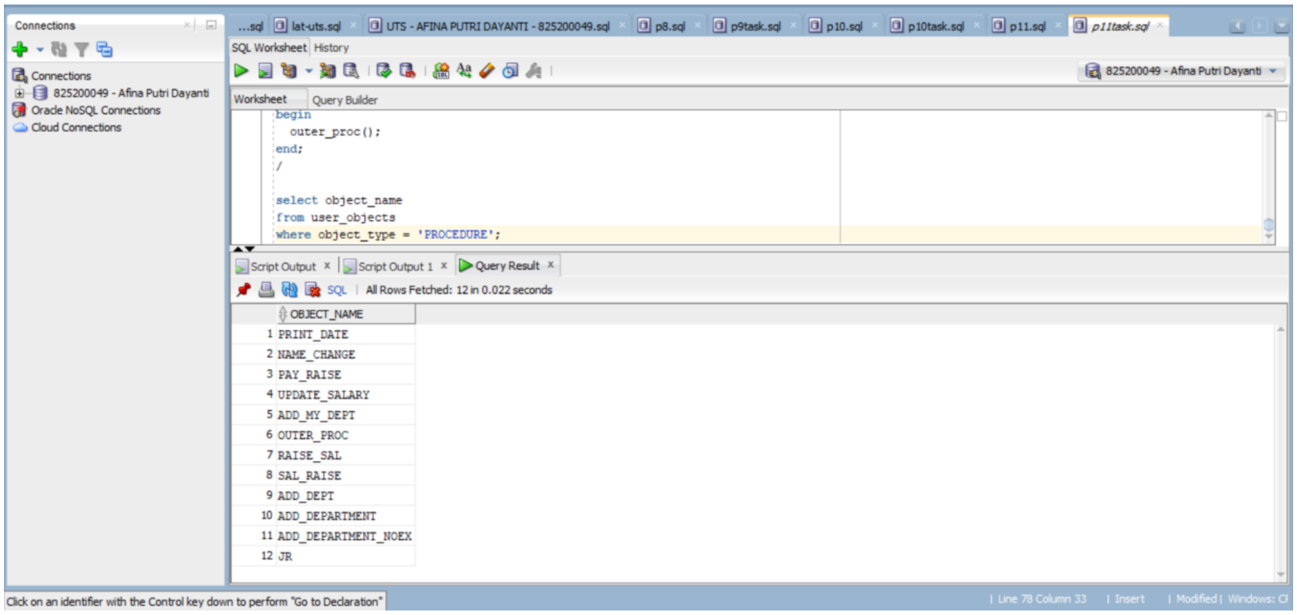
```
/

CREATE OR REPLACE PROCEDURE outer_proc IS
   v_dept NUMBER(2) := 10;
   v_dname  VARCHAR2(30)  := 'Admin';
BEGIN
   add_my_dept(v_dept, v_dname);
EXCEPTION WHEN OTHERS THEN
   DBMS_OUTPUT.PUT_LINE('Exception was propagated to outer_proc');
END;

/
begin
  outer_proc();
end;
```

Task completed in 0.005 seconds

```
PL/SQL procedure successfully completed.

Exception was propagated to outer_proc
```

2. Write and execute a SELECT statement to list the names of all the procedures you have created so far.
Answer :

```
select object_name
from user_objects
where object_type = 'PROCEDURE';
```



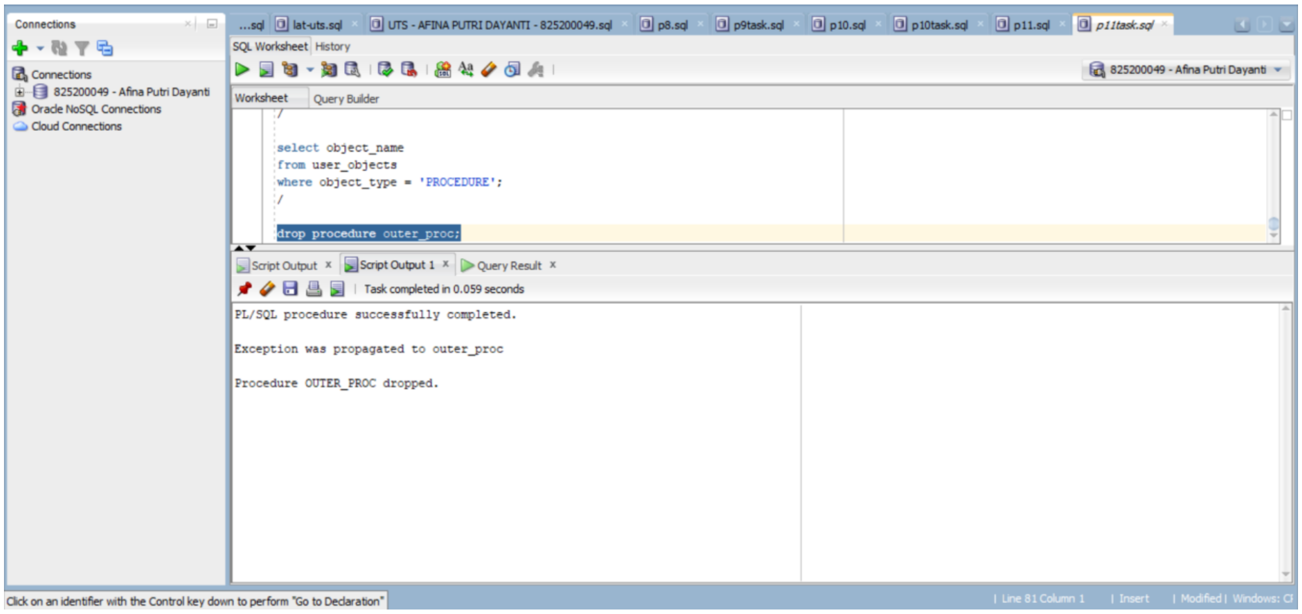
```
begin
  outer_proc();
end;
/

select object_name
from user_objects
where object_type = 'PROCEDURE';
```

All Rows Fetched: 12 in 0.022 seconds

	OBJECT_NAME
1	PRINT_DATE
2	NAME_CHANGE
3	PAY_RAISE
4	UPDATE_SALARY
5	ADD_MY_DEPT
6	OUTER_PROC
7	RAISE_SAL
8	SAL_RAISE
9	ADD_DEPT
10	ADD_DEPARTMENT
11	ADD_DEPARTMENT_NOEX
12	JR

3. Delete the last procedure you created: outer_proc.
Answer :

```
drop procedure outer_proc;
```

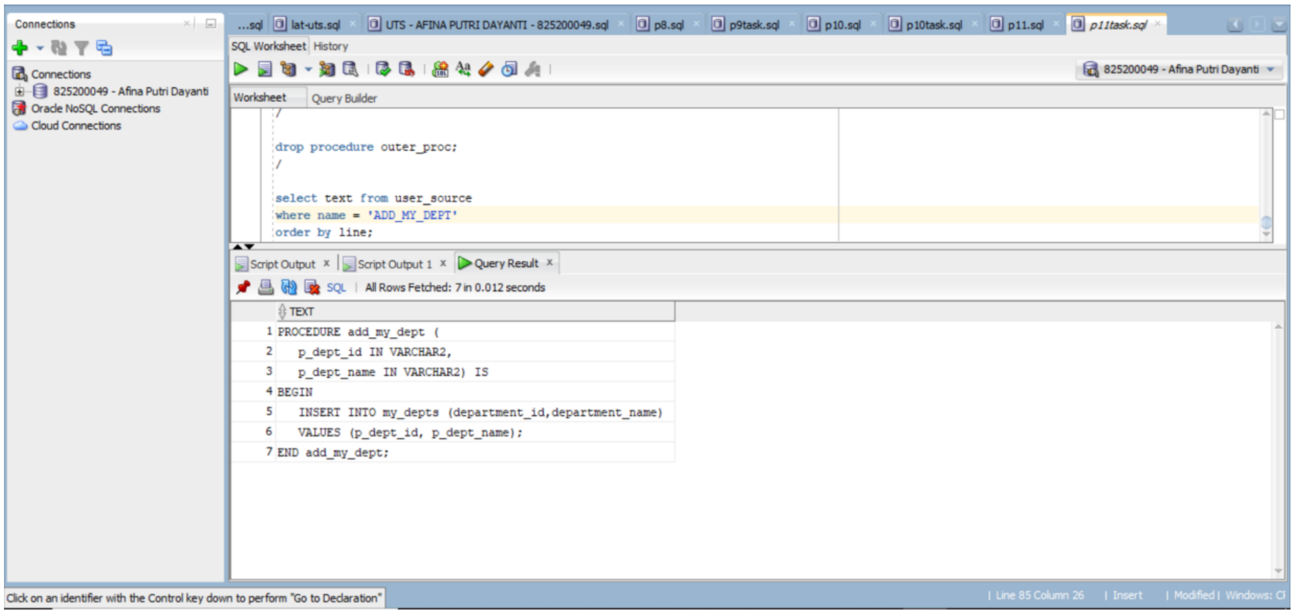


```
/

select object_name
from user_objects
where object_type = 'PROCEDURE';
/

drop procedure outer_proc;
```

```
PL/SQL procedure successfully completed.

Exception was propagated to outer_proc

Procedure OUTER_PROC dropped.
```

4. Write and execute a SELECT statement to list the source code of your add_my_dept procedure. Make sure your SELECT statement list the lines of code in the correct order.
Answer :

```
select text from user_source
where name = 'ADD_MY_DEPT'
order by line;
```



```
/

drop procedure outer_proc;
/

select text from user_source
where name = 'ADD_MY_DEPT'
order by line;
```

All Rows Fetched: 7 in 0.012 seconds

	TEXT
1	PROCEDURE add_my_dept (
2	p_dept_id IN VARCHAR2,
3	p_dept_name IN VARCHAR2) IS
4	BEGIN
5	INSERT INTO my_depts (department_id,department_name)
6	VALUES (p_dept_id, p_dept_name);
7	END add_my_dept;