Lecture Topics

Stored Functions
Exceptions
Cursors

Objectives

After covering this material, students should be able to

- Describe and demonstrate to deal with common PL/SQL exceptions
- Describe how exceptions can be handled using Late Raising exception handling rather than Early Raising techniques
- Contrast Stored Procedures and Stored Functions
- Demonstrate the use of Explicit cursors in Select statements

References

Lecture Material

Week 2 lecture (available via Blackboard / Lectopia)

Reference material (see subject handout for the full details of these books):

PL/SQL:
http://download.oracle.com/docs/cd/B19306_01/appdev.102/b14261/toc.htm
http://docstore.mik.ua/orelly/oracle/prog2/ch15_01.htm

Terms

You must be familiar with the following terms and concepts:

Stored Procedures	Stored Functions	Parameters
Return values	Exceptions	Oracle Exception
Early Raising	Late Raising	Explicit Cursors

Tutorial Questions

Assume that this code is successfully executed.

```
create table student (
stuid integer primary key,
stuname varchar2(20),
gender varchar2(1) );
insert into student values (1,'Tom','M');
insert into student values (2,'Clare','F');
insert into student values (3,'Fred','F');

insert into student values (4,'Tom','M');
```

Assume that this code is also successfully executed.

```
CREATE OR REPLACE FUNCTION GetGender(pStuName VARCHAR2)
                                                RETURN VARCHAR2 AS
  vGender    student.gender%TYPE;
  vRetValue  VARCHAR2(100);
BEGIN
  SELECT     gender INTO vGender
  FROM       student
  WHERE      stuName = pStuName;
  vRetValue := vGender;
  RETURN vRetValue;
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    vRetValue := 'No matching student found';
    RETURN vRetValue;
  WHEN TOO_MANY_ROWS THEN
    vRetValue := 'Too many matching students found';
    RETURN vRetValue;
END;
```

1. **In relation to the above code, answer these questions**

   a. **How many parameters does this Stored Function have?**
   ⇨ One which is pStuName.

   b. **What is the return type of this Stored Function?**
   ⇨ Varchar2

   c. **What is the data type of the variable named pStuName?**
   ⇨ Varchar2

2. **What is the output of the following anonymous blocks**

a. F

b. No matching Student found

c. Too many matching student found

3. Lab Questions

Attempt **all** of these lab questions / tasks

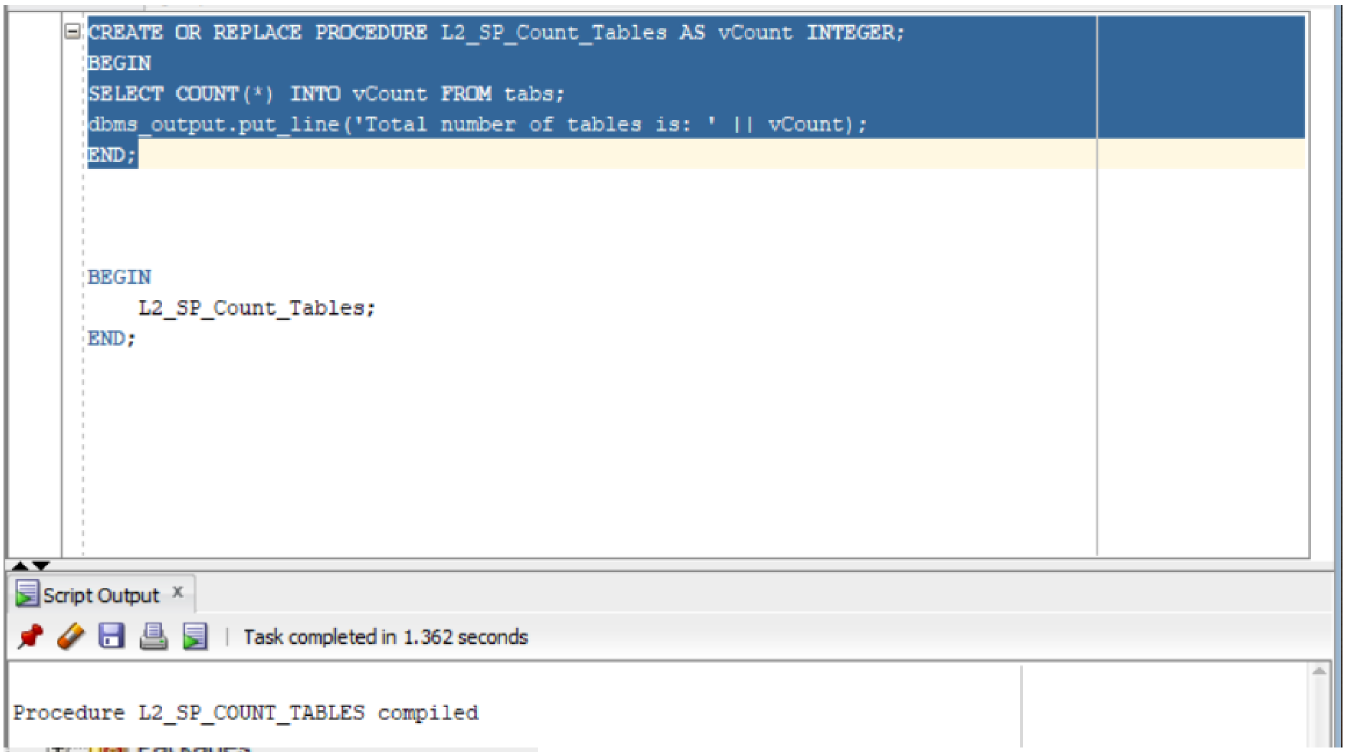
If you do not finish the tasks during the lab session, you should **complete** them **before** the beginning of the next lab.

These tasks will help you prepare for working on your assignment.

1. Copy this code and paste it into a SQL Developer worksheet.

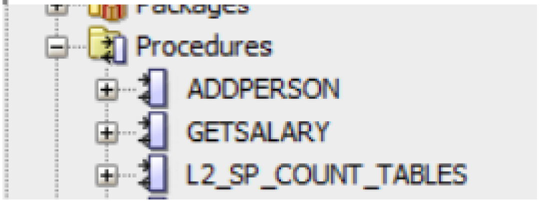
```
CREATE OR REPLACE PROCEDURE L2_SP_Count_Tables
  AS vCount INTEGER;
BEGIN
  SELECT COUNT(*) INTO vCount FROM tabs;
  dbms_output.put_line('Total number of tables is: ' ||

vCount); END;
```

- Run the script to **compile** the code and create a Stored Procedure.
- On the **left side of the screen**, find the list of all stored procedures linked to your connection
- You may need to Right Click on the heading **Procedures** and click **Refresh**

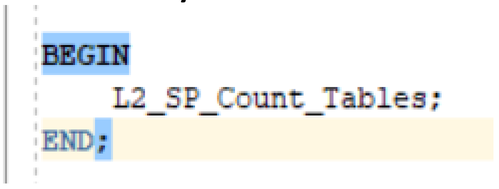


```
CREATE OR REPLACE PROCEDURE L2_SP_Count_Tables AS vCount INTEGER;
BEGIN
SELECT COUNT(*) INTO vCount FROM tabs;
dbms_output.put_line('Total number of tables is: ' || vCount);
END;



BEGIN
    L2_SP_Count_Tables;
END;
```

```
Procedure L2_SP_COUNT_TABLES compiled
```



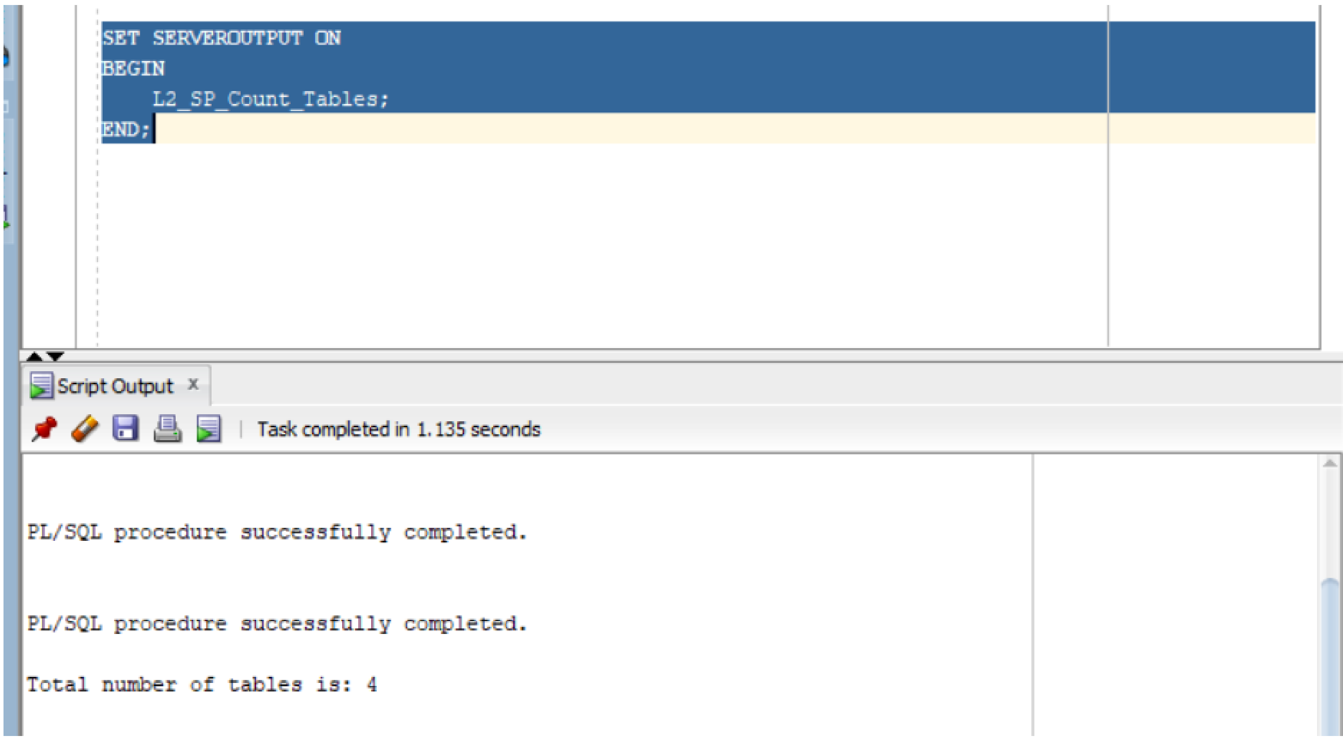
2. Write an **anonymous block** that executes this SP



```
BEGIN
    L2_SP_Count_Tables;
END;
```

3. **Execute** your anonymous block so that the SP named **L2_SP_Count_Tables** is executed

   ****Hint. If you don't see any output, ensure that you execute the statement **set serveroutput on;****



```
SET SERVEROUTPUT ON
BEGIN
    L2_SP_Count_Tables;
END;
```

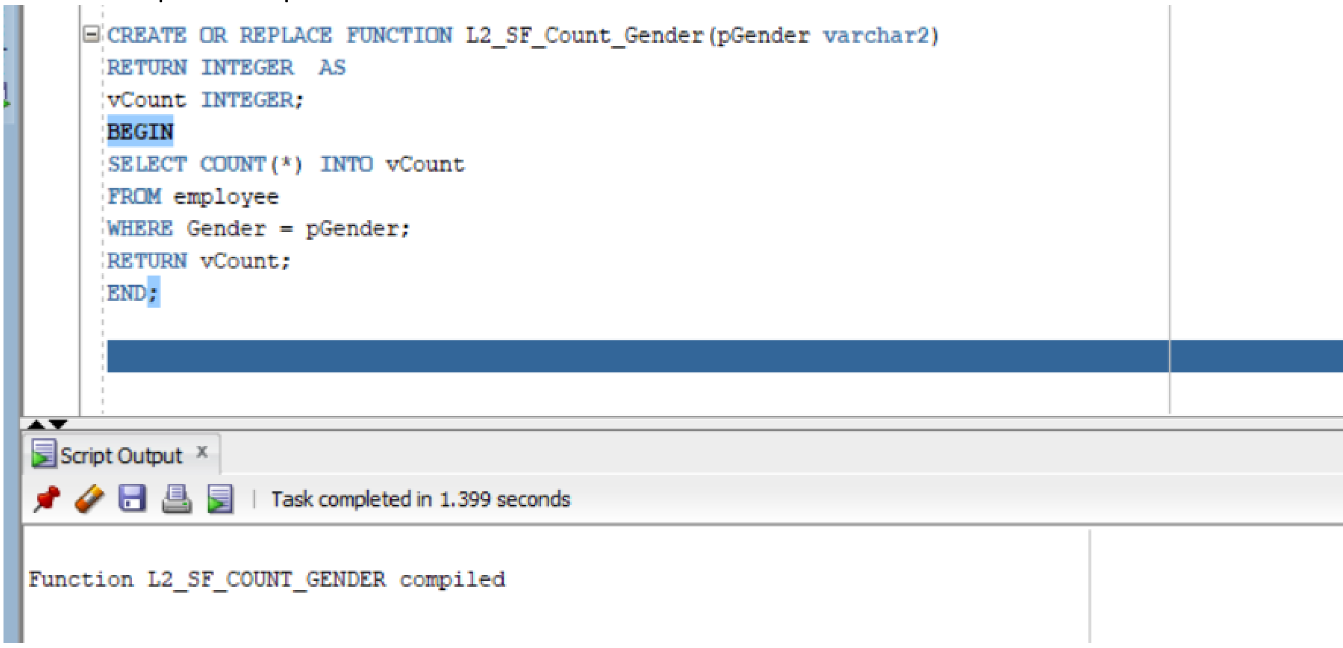
```
PL/SQL procedure successfully completed.


PL/SQL procedure successfully completed.

Total number of tables is: 4
```

Use the employee table from week 1 lab work for the following

4. Copy this code and paste it into a SQL Developer worksheet.

```
CREATE OR REPLACE FUNCTION L2_SF_Count_Gender(pGender varchar2)
                                              RETURN INTEGER   AS
  vCount INTEGER;
BEGIN
  SELECT COUNT(*) INTO vCount
  FROM employee
  WHERE Gender = pGender;
  RETURN vCount;
END;
```

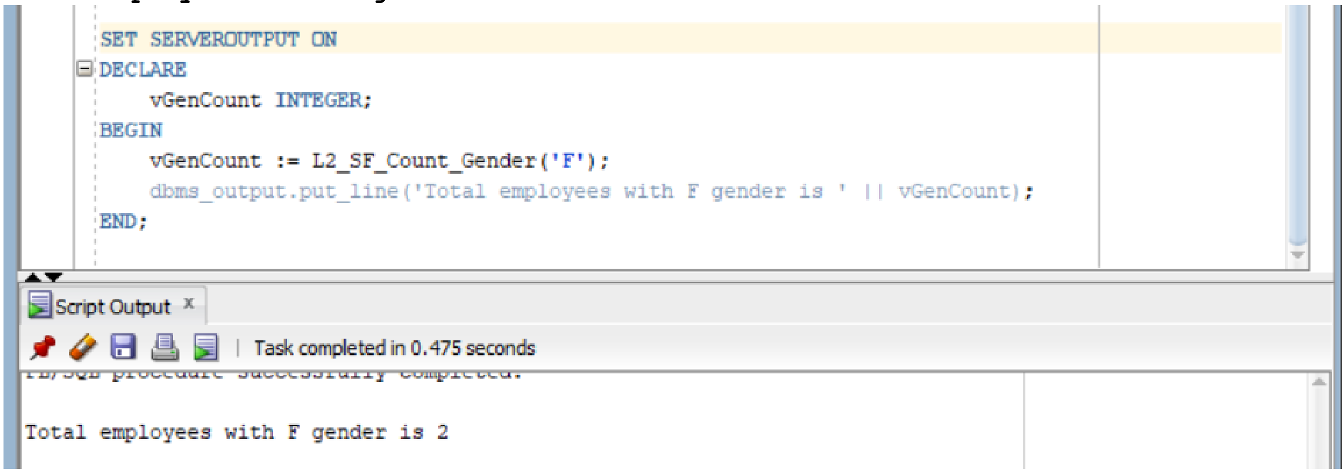
5. Run the script to compile the code and create a Stored Procedure.



```
CREATE OR REPLACE FUNCTION L2_SF_Count_Gender(pGender varchar2)
RETURN INTEGER  AS
vCount INTEGER;
BEGIN
SELECT COUNT(*) INTO vCount
FROM employee
WHERE Gender = pGender;
RETURN vCount;
END;
```

```
Function L2_SF_COUNT_GENDER compiled
```

6. Write an **anonymous block** that executes this SP
   Your code must display the output in this format:
   `Total employees with F gender is 2`



```
SET SERVEROUTPUT ON
DECLARE
    vGenCount INTEGER;
BEGIN
    vGenCount := L2_SF_Count_Gender('F');
    dbms_output.put_line('Total employees with F gender is ' || vGenCount);
END;
```

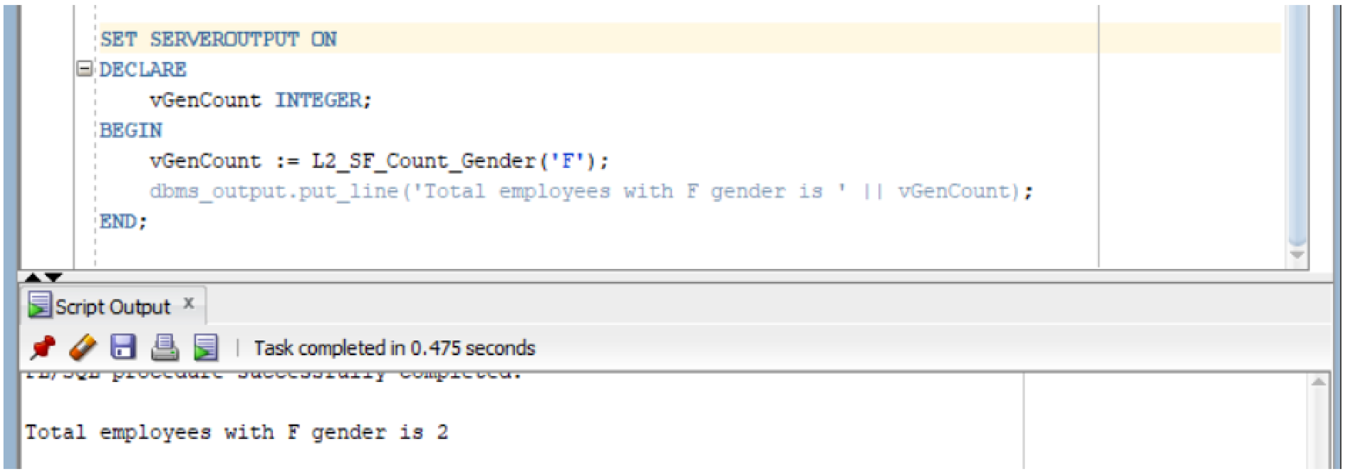
```
PL/SQL procedure successfully completed.

Total employees with F gender is 2
```

7. Execute your anonymous block so that the SP named **L2_SF_Count_Gender** is executed

   Hint. If you don't see any output, ensure that you execute the statement **set serveroutput on;**

   - Test your code.
     - Pass the value F.
     - Pass the value M.
     - Pass the value X.

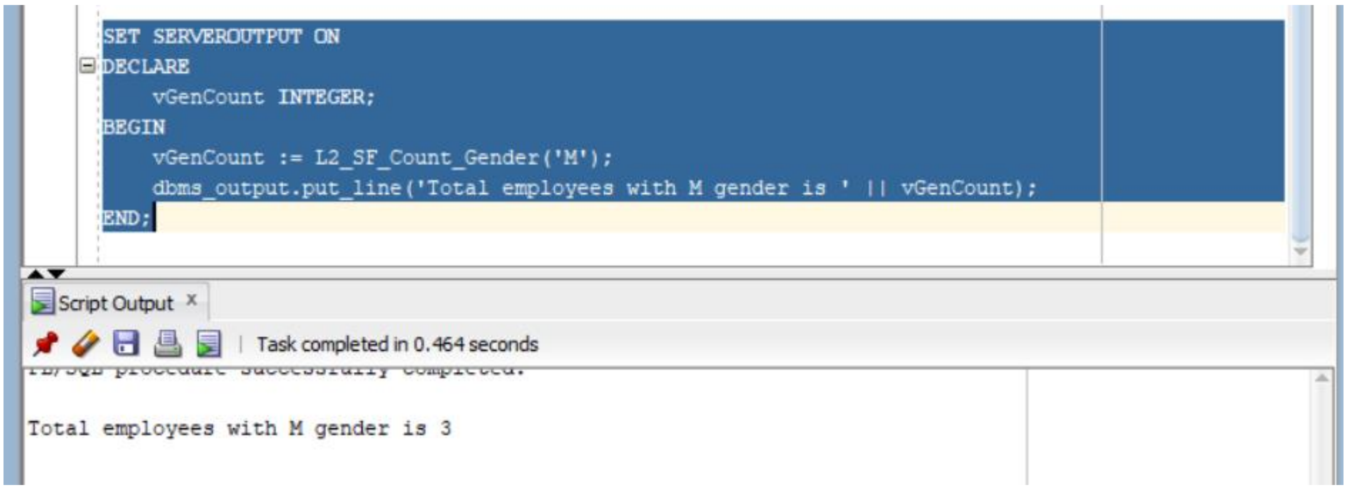
Value F:



```
SET SERVEROUTPUT ON
DECLARE
    vGenCount INTEGER;
BEGIN
    vGenCount := L2_SF_Count_Gender('F');
    dbms_output.put_line('Total employees with F gender is ' || vGenCount);
END;
```

```
PL/SQL procedure successfully completed.

Total employees with F gender is 2
```

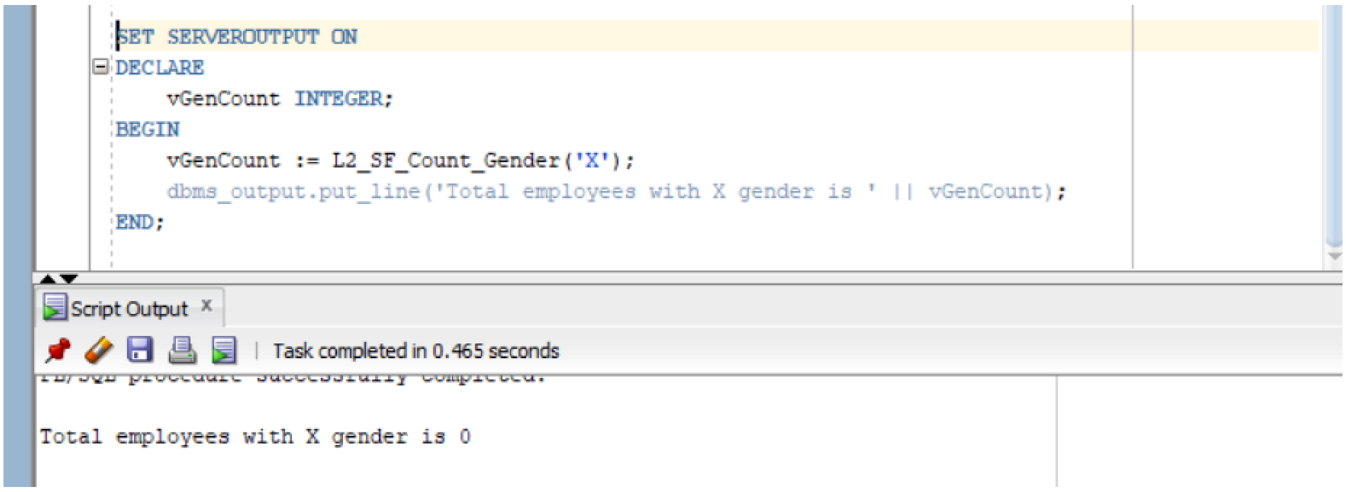
Value M:



```
SET SERVEROUTPUT ON
DECLARE
    vGenCount INTEGER;
BEGIN
    vGenCount := L2_SF_Count_Gender('M');
    dbms_output.put_line('Total employees with M gender is ' || vGenCount);
END;
```

```
PL/SQL procedure successfully completed.

Total employees with M gender is 3
```

Value X:



```
SET SERVEROUTPUT ON
DECLARE
    vGenCount INTEGER;
BEGIN
    vGenCount := L2_SF_Count_Gender('X');
    dbms_output.put_line('Total employees with X gender is ' || vGenCount);
END;
```

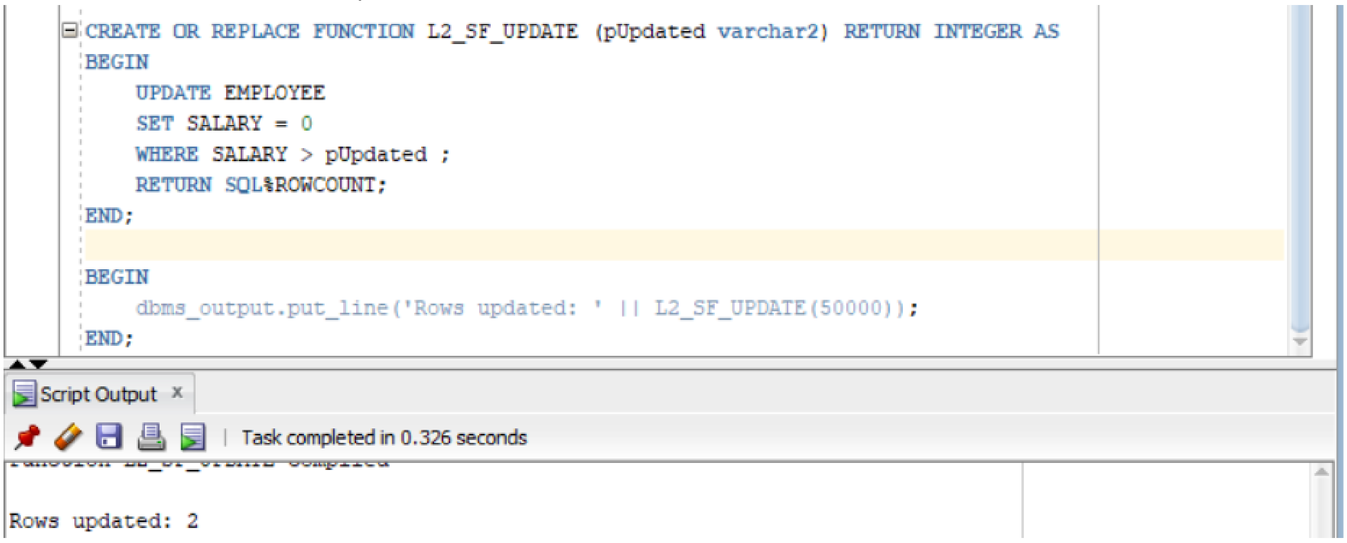
Script Output ×

Task completed in 0.465 seconds

```
PL/SQL procedure successfully completed.

Total employees with X gender is 0
```

8. Write a SF named L2_SF_Update that sets the salary value to zero for employees who have a salary greater than a parameter value.

   E.g. if a block calls L2_SP_Reset(50000), then all employee who earn more than 50000 will have their salary set to zero. The function must return a value indicated how many rows were updated.



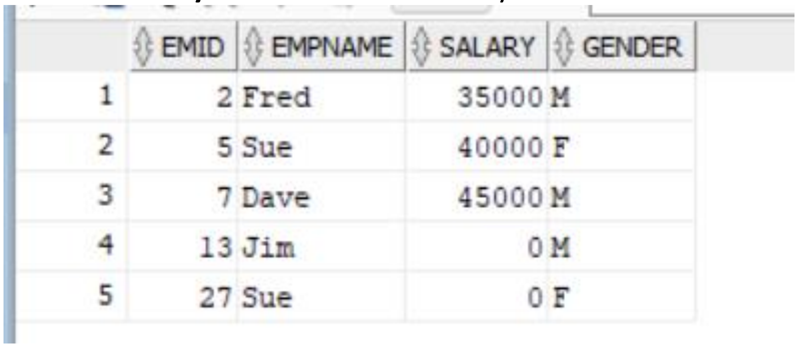
```
CREATE OR REPLACE FUNCTION L2_SF_UPDATE (pUpdated varchar2) RETURN INTEGER AS
BEGIN
    UPDATE EMPLOYEE
    SET SALARY = 0
    WHERE SALARY > pUpdated ;
    RETURN SQL%ROWCOUNT;
END;

BEGIN
    dbms_output.put_line('Rows updated: ' || L2_SF_UPDATE(50000));
END;
```

Script Output ×

Task completed in 0.326 seconds

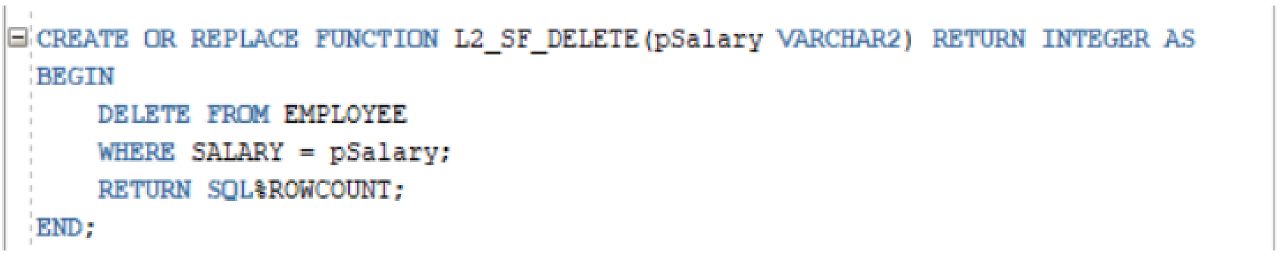
```
Function L2_SF_UPDATE compiled

Rows updated: 2
```

9. Write an **anonymous block** that tests your code



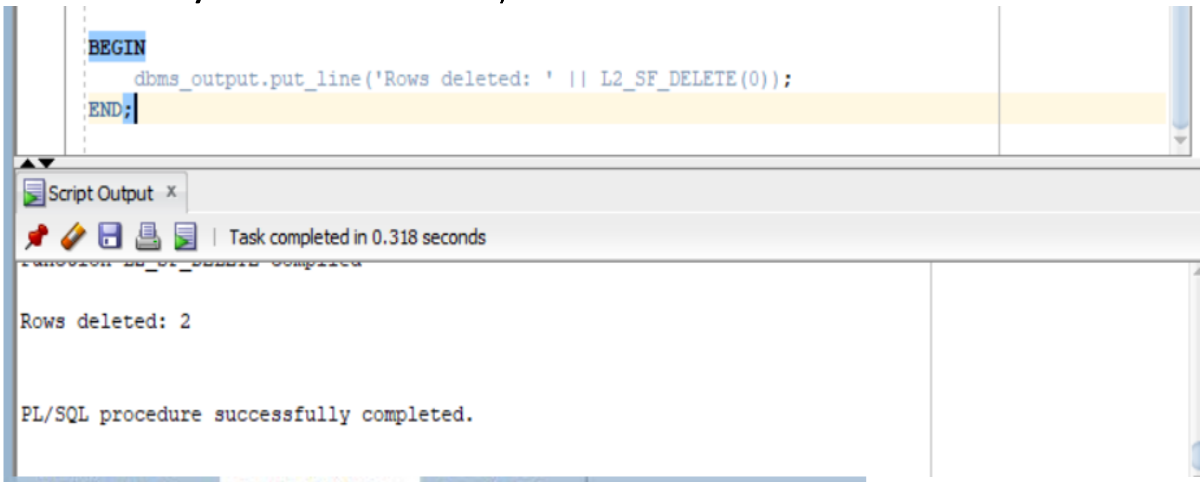
	EMID	EMPNAME	SALARY	GENDER
1	2	Fred	35000	M
2	5	Sue	40000	F
3	7	Dave	45000	M
4	13	Jim	0	M
5	27	Sue	0	F

10. Write a SF named L2_SF_DELETE that deletes employees whose is equal to zero. The function must return a value indicated how many rows were deleted.



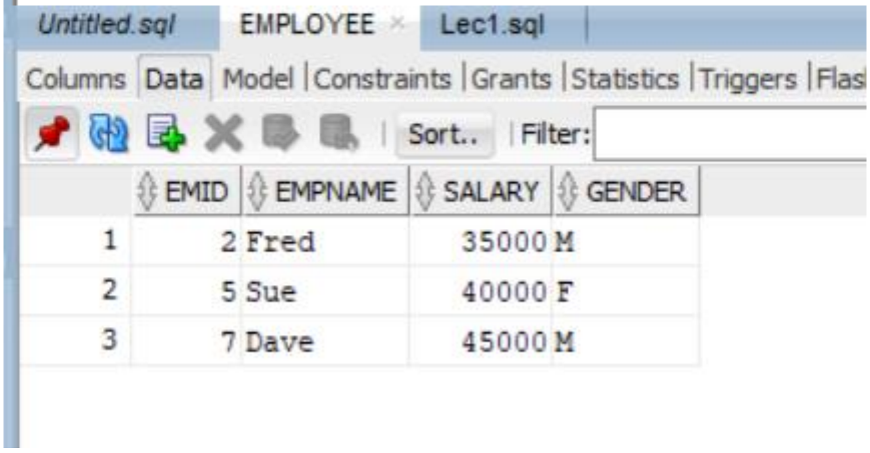
```
CREATE OR REPLACE FUNCTION L2_SF_DELETE(pSalary VARCHAR2) RETURN INTEGER AS
BEGIN
    DELETE FROM EMPLOYEE
    WHERE SALARY = pSalary;
    RETURN SQL%ROWCOUNT;
END;
```

11. Write an **anonymous block** that tests your code



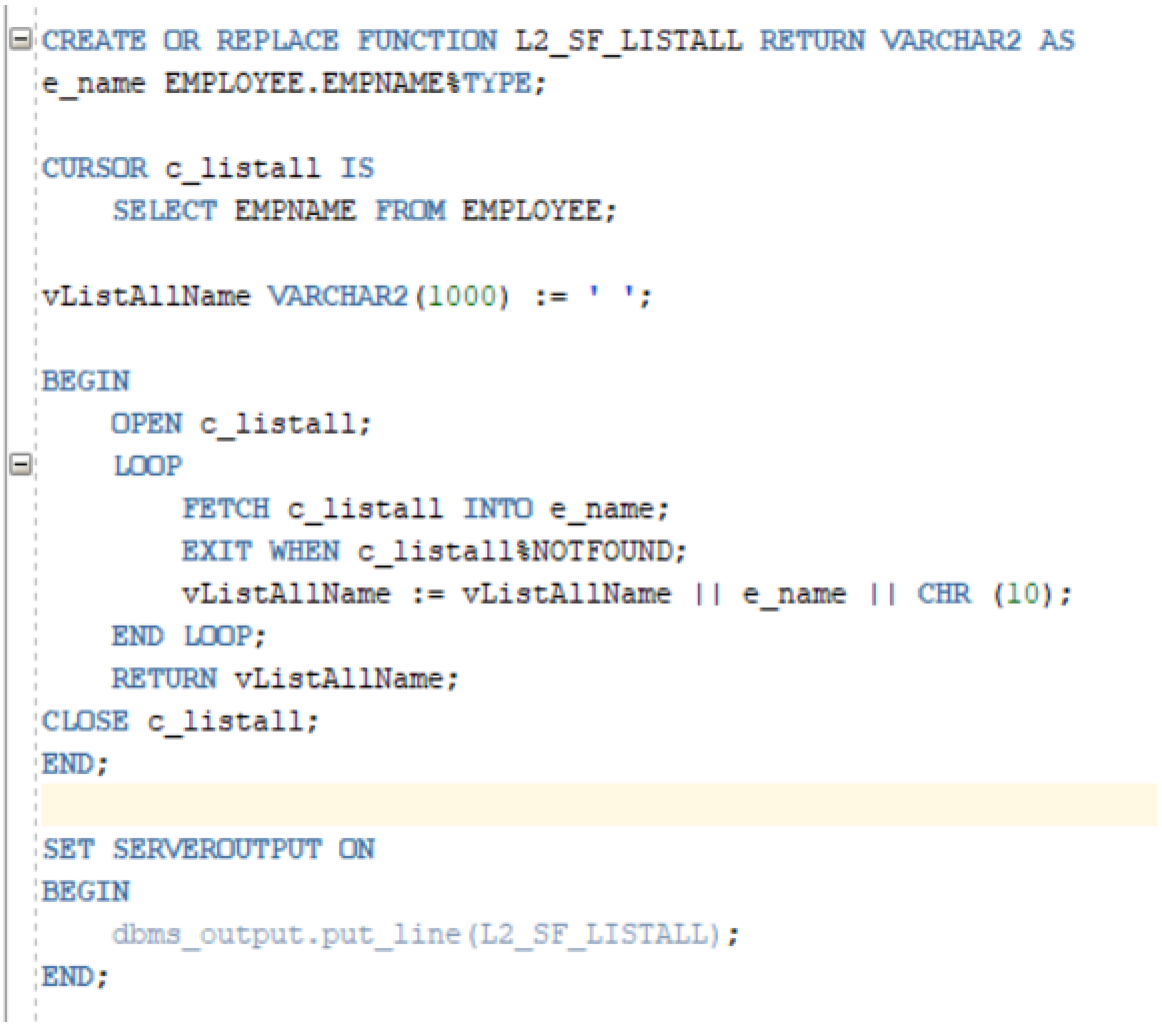
```
BEGIN
   dbms_output.put_line('Rows deleted: ' || L2_SF_DELETE(0));
END;
```

```
Rows deleted: 2


PL/SQL procedure successfully completed.
```



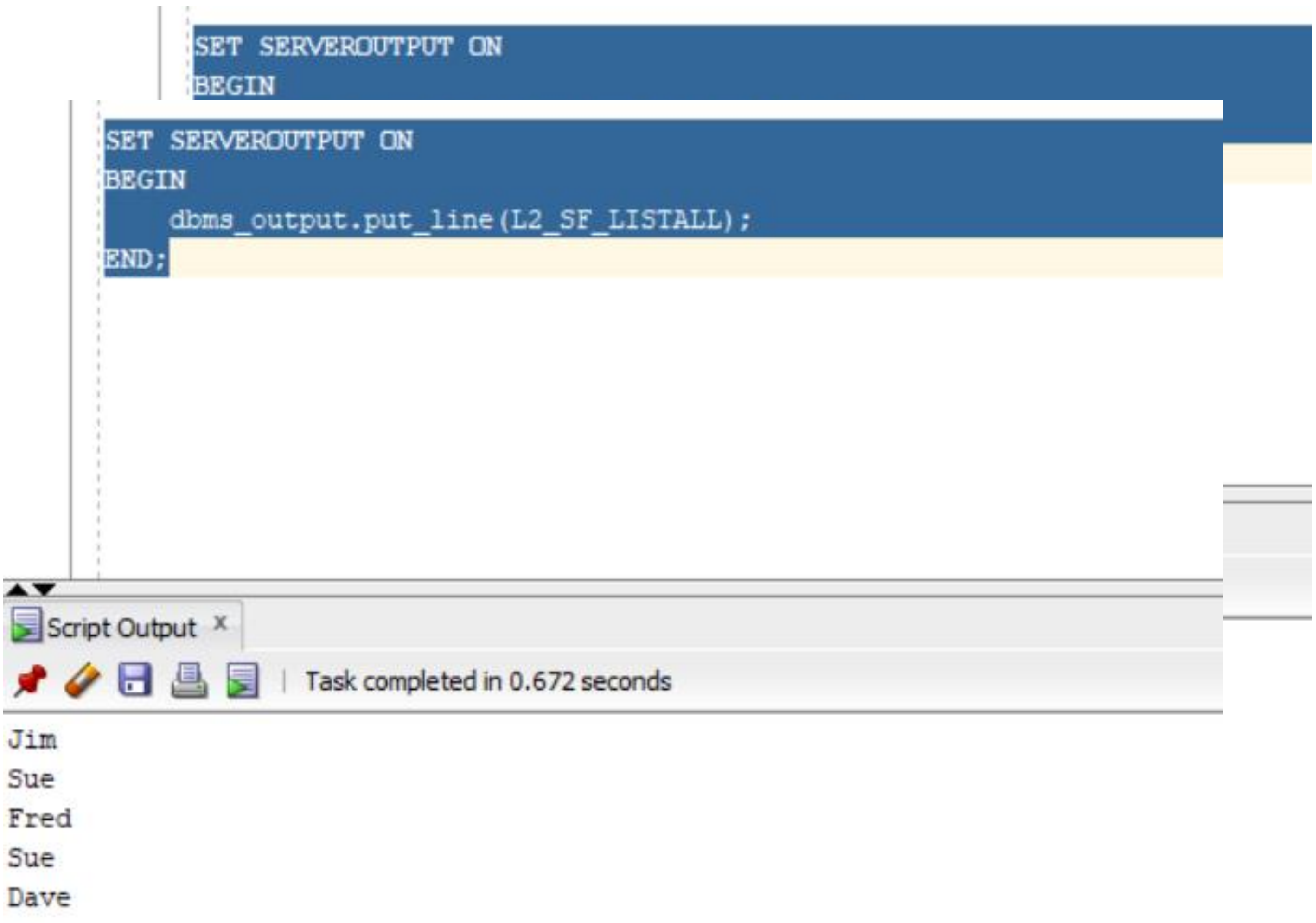
	EMID	EMPNAME	SALARY	GENDER
1	2	Fred	35000	M
2	5	Sue	40000	F
3	7	Dave	45000	M

12. Write a SF named L2_SP_LISTALL that lists all employee names. You an explicit cursor to do this.



```
CREATE OR REPLACE FUNCTION L2_SF_LISTALL RETURN VARCHAR2 AS
e_name EMPLOYEE.EMPNAME%TYPE;

CURSOR c_listall IS
    SELECT EMPNAME FROM EMPLOYEE;

vListAllName VARCHAR2(1000) := ' ';

BEGIN
    OPEN c_listall;
    LOOP
        FETCH c_listall INTO e_name;
        EXIT WHEN c_listall%NOTFOUND;
        vListAllName := vListAllName || e_name || CHR (10);
    END LOOP;
    RETURN vListAllName;
CLOSE c_listall;
END;

SET SERVEROUTPUT ON
BEGIN
    dbms_output.put_line(L2_SF_LISTALL);
END;
```

13. Write an **anonymous block** that tests your code



```
SET SERVEROUTPUT ON
BEGIN
```

```
SET SERVEROUTPUT ON
BEGIN
    dbms_output.put_line(L2_SF_LISTALL);
END;
```

```
Jim
Sue
Fred
Sue
Dave
```

Continue work on your Assignment.