1. Write a program in PL/SQL to show the uses of static PL/SQL statement.

Ans. DROP TABLE emp\_temp;

CREATE TABLE emp\_temp AS

SELECT employee\_id, first\_name, last\_name

FROM employees;

DECLARE

emp\_id emp\_temp.employee\_id%TYPE := 285;

emp\_f\_name emp\_temp.first\_name%TYPE := 'Sam';

emp\_l\_name emp\_temp.last\_name%TYPE := 'Sarkar';

BEGIN

INSERT INTO emp\_temp (employee\_id, first\_name, last\_name)

VALUES (emp\_id, emp\_f\_name, emp\_l\_name);

UPDATE emp\_temp

SET first\_name = 'Sam'

WHERE employee\_id = emp\_id;

DELETE FROM emp\_temp

WHERE employee\_id = emp\_id

RETURNING first\_name, last\_name

INTO emp\_f\_name, emp\_l\_name;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE (emp\_f\_name || ' ' || emp\_l\_name);

END;

/

1. Write a program in PL/SQL to show the uses of CURVAL and NEXTVAL with a sequence name.

DECLARE

seq\_value NUMBER;

BEGIN

seq\_value := emp\_seq.NEXTVAL;

DBMS\_OUTPUT.PUT\_LINE ( 'Initial sequence value: ' || TO\_CHAR(seq\_value) );

INSERT INTO emp\_temp (employee\_id, first\_name, last\_name)

VALUES (emp\_seq.NEXTVAL,' Max', 'White');

INSERT INTO emp\_temp1 VALUES (emp\_seq.CURRVAL,

'June', 'England');

seq\_value := emp\_seq.CURRVAL;

DELETE FROM emp\_temp

WHERE employee\_id = seq\_value;

UPDATE emp\_temp

SET employee\_id = emp\_seq.NEXTVAL

WHERE first\_name = 'Max'

AND last\_name = 'White';

seq\_value := emp\_seq.CURRVAL;

DBMS\_OUTPUT.PUT\_LINE ( 'Ending sequence value: ' || TO\_CHAR(seq\_value) );

END;

/

1. Write a program in PL/SQL to find the number of rows effected by the use of SQL%ROWCOUNT attributes of an implicit cursor.

Ans.

DECLARE

total\_rows number(2);

BEGIN

UPDATE customers

SET salary = salary + 500;

IF sql%notfound THEN

dbms\_output.put\_line('no customers selected');

ELSIF sql%found THEN

total\_rows := sql%rowcount;

dbms\_output.put\_line( total\_rows || ' customers selected ');

END IF;

END;

/

1. Write a program in PL/SQL to show the uses of implicit cursor without using any attribute.

DECLARE

emp\_first\_name VARCHAR2(35);

emp\_last\_name VARCHAR2(35);

zemp\_id NUMBER:=&employee\_id;

BEGIN

SELECT first\_name,

last\_name

INTO emp\_first\_name, emp\_last\_name

FROM employees

WHERE employee\_id = zemp\_id;

dbms\_output.Put\_line ('Employee name: '

|| emp\_first\_name

||' ' ||emp\_last\_name);

EXCEPTION

WHEN no\_data\_found THEN

dbms\_output.Put\_line ('There is no employee with the ID '||to\_char(zemp\_id));

END;

1. Write a program in PL/SQL to show the uses of SQL%FOUND to determine if a DELETE statement affected any rows.

Ans. DROP TABLE emp\_temp;

CREATE TABLE emp\_temp AS

SELECT employee\_id, first\_name, last\_name

FROM employees;

CREATE OR REPLACE PROCEDURE test\_proc ( z\_emp\_id NUMBER) AUTHID DEFINER AS

BEGIN

DELETE FROM emp\_temp

WHERE employee\_id = z\_emp\_id;

IF SQL%FOUND THEN

DBMS\_OUTPUT.PUT\_LINE ( 'Delete succeeded for employee\_id: ' || z\_emp\_id);

ELSE

DBMS\_OUTPUT.PUT\_LINE ('No employee of ID '|| z\_emp\_id||'is found.');

END IF;

END;

/

BEGIN

test\_proc(175);

test\_proc(444);

END;

1. Write a program in PL/SQL to show the uses of SQL%NOTFOUND to determine if a UPDATE statement affected any rows.

Ans DROP TABLE emp\_temp;

CREATE TABLE emp\_temp AS

SELECT employee\_id, first\_name, last\_name,email

FROM employees;

DECLARE

z\_emp\_id NUMBER:=&employee\_id;

BEGIN

UPDATE emp\_temp

SET email = 'not available'

WHERE employee\_id = z\_emp\_id;

IF SQL%NOTFOUND THEN

DBMS\_OUTPUT.PUT\_LINE ('No employee of ID '|| z\_emp\_id||' is found.');

ELSE

DBMS\_OUTPUT.PUT\_LINE ( 'Update succeeded for employee\_id: ' || z\_emp\_id);

END IF;

END;

/

1. Write a program in PL/SQL to create a table-based record using the %ROWTYPE attribute.

Ans… DECLARE

e\_employee employees%ROWTYPE;

z\_emp\_id NUMBER:=&employee\_id;

BEGIN

SELECT \*

INTO e\_employee

FROM employees

WHERE employee\_id = z\_emp\_id;

dbms\_output.Put\_line (e\_employee.first\_name

||' '

||e\_employee.last\_name

||' has an ID of '||z\_emp\_id);

EXCEPTION

WHEN no\_data\_found THEN

Raise\_application\_error(-20001, 'The Employee '

|| 'is not in the database');

END;

/

1. Write a program in PL/SQL to display a table based detail information for the employee of ID 149 from the employees table.

Ans… DECLARE

e\_employee employees%ROWTYPE;

BEGIN

SELECT \*

INTO e\_employee -- INTO clause always notifies only single row can be fetch

FROM employees

WHERE employee\_id = 149;

dbms\_output.Put\_line('Employee Details : ID:'

||e\_employee.employee\_id

||' Name: '

||e\_employee.first\_name

||' '

||e\_employee.last\_name

||' Salary: '

||e\_employee.salary);

END;

/

1. Write a program in PL/SQL to display a cursor based detail information of employees from employees table.

Ans.

DECLARE

e\_empid employees.employee\_id%TYPE;

e\_empname employees.first\_name%TYPE;

e\_salary employees.salary%TYPE;

CURSOR employee\_cursor IS

SELECT employee\_id,

first\_name,

salary

FROM employees;

BEGIN

OPEN employee\_cursor;

LOOP

FETCH employee\_cursor

INTO e\_empid,

e\_empname,

e\_salary;

EXIT

WHEN employee\_cursor%NOTFOUND;

IF (e\_salary > 8000) THEN

dbms\_output.Put\_line(e\_empid

|| ' '

|| e\_empname

|| ' '

|| e\_salary);

ELSE

dbms\_output.Put\_line(e\_empname

|| ' salary is less then 8000');

END IF;

END LOOP;

CLOSE employee\_cursor;

END;

1. Write a program in PL/SQL to retriev the records from the employees table and display them using cursors.

Ans. DECLARE

e\_empid employees.employee\_id%TYPE;

e\_empname employees.first\_name%TYPE;

e\_salary employees.salary%TYPE;

CURSOR employee\_cursor IS

SELECT employee\_id,

first\_name,

salary

FROM employees;

BEGIN

OPEN employee\_cursor;

LOOP

FETCH employee\_cursor

INTO e\_empid,

e\_empname,

e\_salary;

EXIT

WHEN employee\_cursor%NOTFOUND;

IF (e\_salary > 5000) THEN

dbms\_output.Put\_line(e\_empid

|| ' '

|| e\_empname

|| ' '

|| e\_salary);

ELSE

dbms\_output.Put\_line(e\_empname);

END IF;

END LOOP;

CLOSE employee\_cursor;

END;

/