17. Cursor (Any Two) a) The bank manager has decided to activate all those accounts which were previously marked as inactive for performing no transaction in last 365 days. Write a PL/SQ block (using implicit cursor) to update the status of account, display an approximate message based on the no. of rows affected by the update. (Use of %FOUND, %NOTFOUND, %ROWCOUNT)

SQL> create table bank\_manager(

2 id number(3) not null primary key,

3 inactive\_days number(3)

4 );

Table created.

SQL> insert into bank\_manager (id, inactive\_days) values (01,256);

1 row created.

SQL> insert into bank\_manager (id, inactive\_days) values (02,456);

1 row created.

SQL> insert into bank\_manager (id, inactive\_days) values (03,545);

1 row created.

SQL> insert into bank\_manager (id, inactive\_days) values (04,222);

1 row created.

SQL> insert into bank\_manager (id, inactive\_days) values (05,120);

1 row created.

SQL> insert into bank\_manager (id, inactive\_days) values (06,03);

1 row created.

SQL> select \* from bank\_manager;

ID INACTIVE\_DAYS

---------- -------------

1 256

2 456

3 545

4 222

5 120

6 3

6 rows selected.

SQL> alter table bank\_manager add status number(2) ;

Table altered.

SQL> select \* from bank\_manager;

ID INACTIVE\_DAYS STATUS

---------- ------------- ----------

1 256

2 456

3 545

4 222

5 120

6 3

6 rows selected.

SQL> edit

Wrote file afiedt.buf

1 declare

2 total\_rows number(3);

3 begin

4 update bank\_manager set status = 1 where inactive\_days>356;

5 if sql%notfound then

6 dbms\_output.put\_line('No Record Found');

7 elsifsql%found then

8 total\_rows := sql%rowcount;

9 dbms\_output.put\_line('Account Updated: '||total\_rows);

10 end if;

11\* end;

SQL> /

PL/SQL procedure successfully completed.

SQL> set serveroutput on;

SQL> /

Account Updated: 2

PL/SQL procedure successfully completed.

SQL> select \* from bank\_manager;

ID INACTIVE\_DAYS STATUS

---------- ------------- ----------

1 256

2 456 1

3 545 1

4 222

5 120

6 3

6 rows selected.

18. TRIGGER:

CREATE TABLE Employee

(

Id INT PRIMARY KEY,

Name VARCHAR(45),

Salary INT,

Gender VARCHAR(12),

DepartmentId INT

)

CREATE TABLE Audit2

(

Salary INT

) ;

INSERT INTO Employee VALUES (1,'Steffan', 82000, 'Male', 3);

INSERT INTO Employee VALUES (2,'XYZ', 79000, 'Female', 4);

CREATE OR REPLACE TRIGGER display\_salary\_changes

BEFORE DELETE OR INSERT OR UPDATE ON Employee

FOR EACH ROW

WHEN (NEW.ID > 0)

DECLARE

sal\_diff number;

BEGIN

dbms\_output.put\_line('Old salary: ' || :OLD.salary);

sal\_diff:= :OLD.salary;

dbms\_output.put\_line('New salary: ' || :NEW.salary);

insert into Audit2 values(sal\_diff);

END;

update Employee set salary=85080 where id=2;

select \* from Audit2;