Cursors and Exceptions:

Practice 1:

• Write a PL/SQL block that takes a department number from a user and increases the salary of all the employees belonging to the department by 10%. The block should display on the screen how many records are updated.

```
SET SERVEROUTPUT ON
DECLARE
   DETAILS EMPLOYEE_Kushmakar%ROWTYPE;
   EID EMPLOYEE_Kushmakar.ID%TYPE;
   ANS NUMBER := 0;
   DID EMPLOYEE_Kushmakar.DEPID%TYPE := &DID;
   CURSOR UPDATE_SALARY_Kushmakar
IS
   SELECT * FROM Employee_Kushmakar WHERE DEPID = DID;
BEGIN
   OPEN UPDATE_SALARY_Kushmakar;
   LOOP
      FETCH UPDATE_SALARY_Kushmakar INTO DETAILS;
      UPDATE EMPLOYEE_Kushmakar SET SALARY = SALARY + (SALARY/10) WHERE
ID = DETAILS.ID;
      EXIT WHEN UPDATE_SALARY_Kushmakar%NOTFOUND;
      ANS := ANS+1;
   END LOOP;
   DBMS_OUTPUT.PUT_LINE (ANS ||' '||'ROWS UPDATED');
```

```
   CLOSE UPDATE_SALARY_Kushmakar;
END;
/
```

Practice 2:

• Write a PL/SQL block to display the employee name, salary and their corresponding grades (by making use of the "salgrade" table) for the first five employees recorded in the "emp" table by making use of :

• Simple for loop with "EXIT WHEN" condition

• The "WHILE" condition

• Cursor FOR LOOP

Practice 3:

• Write a PL/SQL block that displays names and salaries of all CLERK's recorded in the "emp" table by making use of a cursor.

```
set serveroutput on
declare
   details EMPLOYEE_Kushmakar%rowtype;
   cursor c2_Kushmakar
is
   select * from EMPLOYEE_Kushmakar  where job = 'SDE';
```

```
begin
   open c2_Kushmakar;
   loop
   fetch c2_Kushmakar into details;
   exit when c2_Kushmakar%notfound;
   dbms_output.put_line('name ' ||details.name||' salary '||details.salary);
   end loop;
   close c2_Kushmakar;
end;
/
```

Practice 4:

• Write a PL/SQL block that accepts a value of a job from user and displays the names, department numbers and salaries of the employees from the "emp" table having that job. The block makes use of a parameterized cursor. The user input is passed on to the cursor as a parameter.

```
set serveroutput on
declare
   jobb EMPLOYEE_Kushmakar.job%type := '&job';
   details EMPLOYEE_Kushmakar%rowtype;
   cursor c3(ejob in EMPLOYEE_Kushmakar.job%type)
is
   select * from EMPLOYEE_Kushmakar  where job = jobb;
```

```
begin
   open c3(jobb);
   loop
   fetch c3 into details;
   exit when c3%notfound;
   dbms_output.put_line('name: '||details.name||' dept no.: '||details.depid||'
salary: '||details.salary);
   end loop;
   close c3;
end;
/
```

Practice 5

• Write a PL/SQL block that updates the salaries of the employees as per the following rules.

• If the job is CLERK and deptno is 10 then increase the salary by 20%

• If the job is MANAGER and deptno is 20 then increase the salary by 5%

• For all the other cases increase the salary by 10%

2/3

The block makes use of a cursor for performing the updates and ensures that the appropriate locks are taken on the data retrieved by the cursor, as the data is to be updated.

```
set serveroutput on
declare
   cursor c4
is
   select salary,job,id from EMPLOYEE_Kushmakar;
   v_sal EMPLOYEE_Kushmakar.salary%type;
   v_job EMPLOYEE_Kushmakar.job%type;
   v_deptno EMPLOYEE_Kushmakar.id%type;
begin
   open c4;
   loop
      fetch c4 into v_sal,v_job,v_deptno;
      exit when c4%notfound;
   if (v_job ='ASE' and v_deptno=1) then
      update EMPLOYEE_Kushmakar
      set salary = salary + salary*20/100;
      dbms_output.put_line(v_sal);
   elsif (v_job='SDE' and v_deptno=2) then
      update EMPLOYEE_Kushmakar
      set salary = salary + salary*5/100;
      dbms_output.put_line(v_sal);
   else
      update EMPLOYEE_Kushmakar
      set salary = salary + salary*10/100;
```

```
      dbms_output.put_line(v_sal);
   end if;
   end loop;
   close c4;
end;
/
select * from dept_Kushmakar;
```

Practice 6

• Write a PL/SQL block that retrieves information from the “dept” and “emp” table

Department Number :10 Department Name : XXXX

_____

_____

EMPLOYEE NUMBER EMPLOYEE NAMESALARY JOB_ID

Department Number :20 Department Name : XXXX

_____

_____

EMPLOYEE NUMBER EMPLOYEE NAMESALARY JOB_ID

……and so on for all the departments recorded in the “dept” table.

(Hint : In a loop, use a cursor to retrieve the deptno and the dname from the DEPT table, pass the deptno to retrieve the required column values from the “EMP” table, for those employees who work

in that department.)

Same using parameterized cursors

Exceptions

```
declare
   cursor c_dept is select depid,depname from dept_Kushmakar;
   cursor c_emp(p_deptno number) is select * from employee_Kushmakar where
id=p_deptno;

begin
   for v_dept in c_dept
   loop
      dbms_output.put_line('deptno='||v_dept.depid||'  '|| 'deptname=' ||
v_dept.depname);

      for v_emp in c_emp(v_dept.depid)
      loop

      dbms_output.put_line('name=' ||v_emp.name||'  '|| 'enumber='
||v_emp.id||'  '|| 'salary=' ||v_emp.salary||'  '
      || 'job=' ||v_emp.job);

      end loop;
   end loop;
```

end;

/

Practice 7

• Create a table named MESSAGES (err_message VARCHAR2(250))

• Write a PL/SQL block that accepts a salary value from a user and displays name of the employee having the salary value, on the screen.

• If the salary entered returns more than one row, handle the exception with an appropriate Exception handler and insert into the MESSAGES table the message

“ More than one employee with salary of <input salary>”

• If the salary entered does not return any rows, handle the exception with an appropriate Exception handler and insert into the MESSAGES table the message

“ No employee with salary of < input salary>”

3/3

• If the salary entered returns only one row, insert into the MESSAGES table the employee’s name

• Handle any other exception with an appropriate Exception handler and insert into the MESSAGES table the message “ Some other error occurred”.

create table MESSAGES_Kushmakar(err_message VARCHAR2(250));

```
DECLARE
   SAL employee_Kushmakar.salary%TYPE;
   D employee_Kushmakar%ROWTYPE;
BEGIN
   SAL := &X;
   SELECT * INTO D FROM employee_Kushmakar WHERE
   salary=SAL;
   DBMS_OUTPUT.put_line('NAME IS '|| D.NAME || '
   SALARY IS ' || SAL);
   EXCEPTION
   WHEN TOO_MANY_ROWS THEN
   insert into MESSAGES_Kushmakar values(' More than one
   employee with salary of ' ||'SAL');
   WHEN NO_DATA_FOUND THEN
   insert into MESSAGES_Kushmakar values( ' NO employee
   with salary of ' || 'SAL');
   WHEN OTHERS THEN
   insert into MESSAGES_Kushmakar values( ' SOME OTHER
   ERROR OCCURED');
END;
select * from MESSAGES_Kushmakar;
```

Practice 8

• Write a PL/SQL block that accepts all the column values for dept table as user inputs and inserts a record in the dept table. The block should give a name DUPLICATE_DEPT to the error for duplicate value of the primary key, deptno. (Use pragma EXCEPTION_INIT) The block should write a handler for handling the exception fired when a duplicated value is entered for deptno. The handler should have code for displaying an appropriate message on the screen when DUPLICATE_DEPT is fired.

```
DECLARE
   D_ID dept_Kushmakar.depid%TYPE := '&Department_ID';
   DNAME dept_Kushmakar.depname%TYPE := '&DEPNAME';
   LOC dept_Kushmakar.LOC%TYPE := '&LOC';
   DETAILS dept_Kushmakar%ROWTYPE;
   DUPLICATE_DEP EXCEPTION;
BEGIN
   SELECT * INTO DETAILS FROM dept_Kushmakar WHERE depid =
   D_ID;
   IF SQL%FOUND THEN
   RAISE DUPLICATE_DEP;
   END IF;
   EXCEPTION
   WHEN DUPLICATE_DEP THEN
   DBMS_OUTPUT.PUT_LINE('DEPARTMENT ID ALREADY EXIST IN THE
```

```
   TABLE');
   WHEN NO_DATA_FOUND THEN
   INSERT INTO dept_Kushmakar VALUES (D_ID , DNAME , LOC);
END;
```

Practice 9

• Write a PL/SQL BLOCK to check for more than one President (Job column) in the "emp" table. Create a user defined exception named DUPLICATE_PRESIDENT that should be raised when more than one president is found in the "emp" table. The block should handle the exception by displaying a message "MORE THAN ONE PRESIDENT" on the screen.

```
DECLARE
   cnt NUMBER:=0;
   DUPLICATE_PRESIDENT EXCEPTION;
BEGIN
   SELECT COUNT(*) INTO cnt FROM employee_Kushmakar WHERE
   job='President';
   IF cnt >1 THEN
      RAISE DUPLICATE_PRESIDENT;
   END IF;
   EXCEPTION WHEN DUPLICATE_PRESIDENT THEN
   DBMS_OUTPUT.PUT_LINE('MORE THAN ONE PRESIDENT');
END;
```