Parth Shukla
Lab 7
190905104

CursorName %ISOPEN / FOUND / NOT FOUND:
1. The HRD manager has decided to raise the salary of all the Instructors in a given department number by 5%. Whenever, any such raise is given to the instructor, a record for the same is maintained in the salary_raise table. It includes the Instuctor Id, the date when the raise was given and the actual raise amount. Write a PL/SQL block to update the salary of each Instructor and insert a record in the salary_raise table.
salary_raise(Instructor_Id, Raise_date, Raise_amt)

```
create table salary_raise (instructor_id varchar(5),raise_date date,raise_amt number(5),foreign key (instructor_id) references instructor (id) on delete cascade);

declare
   cursor c1 is select * from instructor;
   info instructor%rowtype;
begin
   open c1;
   loop
      fetch c1 into info;
      exit when c1%NOTFOUND;
      update instructor set salary = 1.05 * salary where id = info.id;
      insert into salary_raise values(info.id,(select sysdate from dual),info.salary*0.05);
   end loop;
   close c1;
end;
/

select * from salary_raise;
```

INSTRUCTOR_ID	RAISE_DATE	RAISE_AMT
10101	09-JUN-21	3250
12121	09-JUN-21	4500
15151	09-JUN-21	2000
22222	09-JUN-21	4750
32343	09-JUN-21	3000
33456	09-JUN-21	4350
45565	09-JUN-21	3750
58583	09-JUN-21	3100
76543	09-JUN-21	4000
76766	09-JUN-21	3600
83821	09-JUN-21	4600
98345	09-JUN-21	4000

Download CSV

CursorName%ROWCOUNT:

2. Write a PL/SQL block that will display the ID, name, dept_name and tot_cred of the first 10 students with lowest total credit.

```
declare
   cursor c1 is select * from student order by tot_cred;
   info student%rowtype;
begin
   open c1;
   loop
      fetch c1 into info;
      exit when c1%ROWCOUNT>10;
      dbms_output.put_line('Student ID: '|| info.id);
      dbms_output.put_line('Student Name: '|| info.name);
      dbms_output.put_line('Department: '|| info.dept_name);
      dbms_output.put_line('Total Credits: '|| info.tot_cred);
      dbms_output.put_line('---------');
   end loop;
   close c1;
end;
/
```

```
Statement processed.
Student ID: 70557
Student Name: Snow
Department: Physics
Total Credits: 0
---------
Student ID: 12345
Student Name: Shankar
Department: Comp. Sci.
Total Credits: 32
---------
Student ID: 55739
Student Name: Sanchez
Department: Music
Total Credits: 38
---------
Student ID: 45678
Student Name: Levy
Department: Physics
Total Credits: 46
---------
Student ID: 54321
Student Name: Williams
Department: Comp. Sci.
Total Credits: 54
---------
Student ID: 44553
Student Name: Peltier
Department: Physics
Total Credits: 56
---------
Student ID: 76543
Student Name: Brown
Department: Comp. Sci.
Total Credits: 58
---------
Student ID: 76653
Student Name: Aoi
Department: Elec. Eng.
Total Credits: 60
---------
Student ID: 19991
Student Name: Brandt
Department: History
Total Credits: 80
```

Cursor For Loops:

3. Print the Course details and the total number of students registered for each course along with the course details - (Course-id, title, dept-name, credits, instructor_name, building, room-number, time-slot-id, tot_student_no)

```
declare
   cursor c1 is with stu as (select * from (student natural join takes natural join section)),ins as
(select * from (instructor natural join teaches natural join section))
              select
course_id,title,ins.dept_name,credits,ins.name,ins.building,ins.room_number,ins.time_slot_id,co
unt(*) as no_of_students from stu inner join  ins using(course_id,sec_id,semester,year) natural
join course
              group by
(course_id,title,ins.dept_name,credits,ins.name,ins.building,ins.room_number,ins.time_slot_id);
begin
   for info in c1
      loop
         dbms_output.put_line('Course ID: '|| info.course_id);
         dbms_output.put_line('Title: '|| info.title);
         dbms_output.put_line('Department: '|| info.dept_name);
         dbms_output.put_line('Credits: '|| info.credits);
         dbms_output.put_line('Instructor Name: '|| info.name);
         dbms_output.put_line('Building: '|| info.building);
         dbms_output.put_line('Room Number: '|| info.room_number);
         dbms_output.put_line('Time Slot ID: '|| info.time_slot_id);
         dbms_output.put_line('Total Students: '|| info.no_of_students);
         dbms_output.put_line('-------');
      end loop;
end;
/
```

```
Statement processed.
Course ID: HIS-351
Title: World History
Department: History
Credits: 3
Instructor Name: El Said
Building: Painter
Room Number: 514
Time Slot ID: C
Total Students: 1
-------
Course ID: BIO-301
Title: Genetics
Department: Biology
Credits: 4
Instructor Name: Crick
Building: Painter
Room Number: 514
Time Slot ID: A
Total Students: 1
-------
Course ID: CS-101
Title: Intro. to Computer Science
Department: Comp. Sci.
Credits: 4
Instructor Name: Srinivasan
Building: Packard
Room Number: 101
Time Slot ID: H
Total Students: 6
-------
Course ID: PHY-101
Title: Physical Principles
Department: Physics
Credits: 4
Instructor Name: Einstein
Building: Watson
Room Number: 100
```

4. Find all students who take the course with Course-id: CS101 and if he/ she has less than 30 total credit (tot-cred), deregister the student from that course. (Delete the entry in Takes table)

```
declare
   cursor c1 is select * from (student natural join takes) where course_id = 'CS-101';
begin
   for info in c1
   loop
      if info.tot_cred < 30 then
         delete from takes where id = info.id and course_id = 'CS-101';
```

```
            dbms_output.put_line('Deregistered : '|| info.name);
        end if;
    end loop;
end;
/
-- Check
select * from (student natural join takes) where course_id = 'CS-101';
```

ID	NAME	DEPT_NAME	TOT_CRED	COURSE_ID	SEC_ID	SEMESTER	YEAR	GRADE
00128	Zhang	Comp. Sci.	102	CS-101	1	Fall	2009	A
12345	Shankar	Comp. Sci.	32	CS-101	1	Fall	2009	C
45678	Levy	Physics	46	CS-101	1	Fall	2009	F
45678	Levy	Physics	46	CS-101	1	Spring	2010	B+
54321	Williams	Comp. Sci.	54	CS-101	1	Fall	2009	A-
76543	Brown	Comp. Sci.	58	CS-101	1	Fall	2009	A
98765	Bourikas	Elec. Eng.	98	CS-101	1	Fall	2009	C-

Download CSV

Where Current of:
5. Alter StudentTable(refer Lab No. 8 Exercise) by resetting column LetterGrade to F. Then write a PL/SQL block to update the table by mapping GPA to the corresponding letter grade for each student.

```
update studenttable set lettergrade = 'F';
select * from studenttable;

declare
    cursor c1 is select * from studenttable for update;
    grade studenttable.lettergrade%type;
begin
    for info in c1
        loop
            if info.gpa < 4 then
                grade := 'F';
            elsif info.gpa < 5 then
                grade := 'E';
            elsif info.gpa < 6 then
                grade := 'D';
            elsif info.gpa < 7 then
```

```
            grade := 'C';
        elsif info.gpa < 8 then
            grade := 'B';
        elsif info.gpa < 9 then
            grade := 'A';
        elsif info.gpa <=10 then
            grade := 'A+';
        else
            grade := null;
        end if;
        update studenttable set lettergrade = grade where current of c1;
    end loop;
end;
/

select * from studenttable;
```

ROLLNO	GPA	LETTERGRADE
1	5.8	F
2	6.5	F
3	3.4	F
4	7.8	F
5	9.5	F

Download CSV

5 rows selected.

Statement processed.

ROLLNO	GPA	LETTERGRADE
1	5.8	D
2	6.5	C
3	3.4	F
4	7.8	B
5	9.5	A+

Download CSV

5 rows selected.

Parameterized Cursors:
6. Write a PL/SQL block to print the list of Instructors teaching a specified course.

```
declare
   cursor c1(c_id teaches.course_id%type) is select * from (instructor natural join teaches)
where course_id = c_id;
   course teaches.course_id%type;
begin
   -- course := '&Course_ID'
   course := 'CS-101';
   dbms_output.put_line('Teaching ' || course);
   for info in c1(course)
      loop
         dbms_output.put_line(info.name);
      end loop;
end;
/
```

```
Statement processed.
Teaching CS-101
Srinivasan
Katz
```

7. Write a PL/SQL block to list the students who have registered for a course taught by his/her advisor.

```
declare
   cursor c1(a_id advisor.i_id%type,c_id takes.course_id%type) is select * from ((student s
natural join takes t) inner join advisor a on (id=a.s_id)) where course_id = c_id and a_id=i_id;
   cursor c2 is select * from (instructor natural join teaches);
begin
   for ins_info in c2
      loop
         for info in c1(ins_info.id,ins_info.course_id)
            loop
               dbms_output.put_line(info.name);
               dbms_output.put_line(info.course_id);
            end loop;
      end loop;
end;
/
```

```
Statement processed.
Shankar
CS-101
Shankar
CS-315
Shankar
CS-347
Peltier
PHY-101
Zhang
CS-101
Brown
CS-101
Brown
CS-319
Tanaka
BIO-101
Tanaka
BIO-301
Aoi
EE-181
```

Transactions (COMMIT / ROLLBACK / SAVEPOINT):

8. Write a PL/SQL block that updates the salary of ‘Biology’ department instructors by 20%. Subsequently, check the whether the department budget can support the raise. If not, undo the raise given to the instructors

```
select * from instructor where dept_name='Biology';

declare
   cursor c1 is select * from instructor where dept_name='Biology' for update;
   sumsal number(20) := 0;
   dept_budget department.budget%type;
begin
   savepoint beforeraise;
   for info in c1
      loop
         sumsal := sumsal + info.salary * 1.2;
         update instructor set salary = 1.2 * salary where current of c1;
      end loop;
      select budget into dept_budget from department where dept_name='Biology';
      -- dbms_output.put_line(dept_budget);
      if sumsal > dept_budget then
         rollback to beforeraise;
         dbms_output.put_line('Budget exceeded');
      end if;
```

```
end;
/

select * from instructor where dept_name='Biology';
```

ID	NAME	DEPT_NAME	SALARY
76766	Crick	Biology	72000

Download CSV

```
Statement processed.
90000
```

ID	NAME	DEPT_NAME	SALARY
76766	Crick	Biology	86400

Download CSV