****

Description: Views

Date:22/09/2023

****

<u>Query:</u>

- **Create a view called MovieDetails that combines information from the Movies, Directors, and Stars tables to display movie titles, directors' names, and the names of stars who acted in those movies.**

```
SQL>  create view moviedetails as select m.title as movies,d.name as directors,s.name as
stars from movies m,directors d,stars s,moviesdirectors md,moviesstars ms where
m.id=md.moviesid and m.id=ms.moviesid and md.directorsid=d.id and ms.starsid=s.id;

View created.

SQL> select * from moviedetails;

MOVIES                              DIRECTORS                 STARS
------------------------------------- ----------------------------- -----------------------------------
----
Rdx                                  Nahas Hidayath            Shane Nigam
premam                               Alphonse Puthran          Nivin Pauly
Usthad Hotel                         Anwar Rasheed             Dulquer Salman
Home                                 Rojin Thomas              Indrans
Avatar                               James Cameroon            Sully
Once upon a time in Hollywood        Quentin Tarantino         Leonardo Dicaprio
Ashiqui 2                            Mohit Suri                Aditya Roy kapur
Chennai express                      Rohit Shetty              Shah Rukh Khan
Vaaranam Aayiram                     Gautham Vasudev Menon     Surya
Mersal                               Atlee                     Vijay

10 rows selected.
```

- **Create a view called HighlyRatedMovies that displays movies with IMDb ratings greater than 8.0, including their titles and ratings.**

```
SQL> create view highlyratedmovies as select title,imdbrating from movies where
imdbrating>8.0;

View created.

SQL> select * from highlyratedmovies;
```

```
TITLE                          IMDBRATING
-------------------------------------- ----------
premam                            12
Usthad Hotel                       9
Oppenheimer                        8.6
Driven                             9
Home                               9.1
Ashiqui 2                          14
Vaaranam Aayiram                   8.2

7 rows selected.
```

- **Create a view called DirectorMovies that provides a list of directors along with the number of movies they have directed.**

```
SQL>  create view directormovies as select d.name
directors,count(md.moviesid)no_of_movies from directors d,moviesdirectors md where
d.id=md.directorsid group by d.name;

View created.

SQL> select * from directormovies;

DIRECTORS                   NO_OF_MOVIES
---------------------------- ------------
Mohit Suri                         1
James Cameroon                     1
Anwar Rasheed                      1
Rojin Thomas                       1
Atlee                              1
Gautham Vasudev Menon              1
Nahas Hidayath                     1
Alphonse Puthran                   1
Rohit Shetty                       1
Quentin Tarantino                  1

10 rows selected.
```

- **Create a view called StarMovies that displays stars' names and the titles of movies they have acted in.**

```
SQL> create view starmovies as select s.name,m.title from movies m,stars s,moviesstars
ms where m.id=ms.moviesid and s.id=ms.starsid;

View created.

SQL> select * from starmovies;
```

```
NAME                           TITLE
-------------------------------------- --------------------------------------
Shane Nigam                       Rdx
Nivin Pauly                       premam
Dulquer Salman                    Usthad Hotel
Indrans                           Home
Sully                             Avatar
Leonardo Dicaprio                 Once upon a time in Hollywood
Aditya Roy kapur                  Ashiqui 2
Shah Rukh Khan                    Chennai express
Surya                             Vaaranam Aayiram
Vijay                             Mersal

10 rows selected.
```

- **Create a view called LongestMovies that lists the titles of movies with the longest runtimes (duration).**

```
SQL> create view longestmovies as select title,duration from(select title,runtime as duration from  movies order by runtime desc)where rownum<=5;

View created.

SQL> select * from longestmovies;

TITLE                              DURATION
-------------------------------------- ----------
Avatar                                3.2
Oppenheimer                           3
Home                                  2.8
Vaaranam Aayiram                      2.8
Once upon a time in Hollywood    2.8
```

- **Create a view called LanguageDistribution that shows the distribution of movies based on the languages they were released in, including the count of movies for each language.**

```
SQL> create view languagedistribution as select language,count(*)movies from movies group by language;

View created.

SQL> select * from languagedistribution;
```

```
LANGUAGE                              MOVIES
-------------------------------------- ----------
malayalam                              5
                                       1
english                                7
tamil                                  4
hindi                                  2
```

- **Create a view called GrossEarnings that displays movies with their titles and gross earnings, sorted by earnings in descending order.**

```
SQL> create view grossearnings as select title,gross as grossearnings from movies order
by gross desc;

View created.

SQL> select * from grossearnings;

TITLE                                  GROSSEARNINGS
-------------------------------------- -------------
Mersal                                 2600000000
Oppenheimer                            730000000
Avatar                                 230000000
Voice of Sathyanathan                  109500000
Chennai express                        109000000
Ashiqui 2                              109000000
premam                                 76000000
Usthad Hotel                           41000000
Vaaranam Aayiram                       40000000
Once upon a time in Hollywood          37000000
Dejavu                                 15000000

TITLE                                  GROSSEARNINGS
-------------------------------------- -------------
Rdx                                    14000000
The Marshes                            8000000
Diary                                  4000000
Open grave                             3000000
Driven                                 3000000
Home                                   3000000
Open the door please                   100000
Santhosham                             .24

19 rows selected.
```

- **Create a view called IndustryHitMovies that shows the titles and release dates of industry hit movies.**

```
SQL> create view industryhitmovies as select title,releasedate from industryhit;

View created.

SQL> select * from industryhitmovies;

TITLE                              RELEASEDA
---------------------------------- ---------
Rdx                                25-AUG-23
Ashiqui 2                          26-APR-13
Vaaranam Aayiram                   14-NOV-08
Oppenheimer                        21-JUL-23
Open grave                         14-AUG-13
Neram                              10-MAY-13
Kadal                              31-JAN-13
Bheed                              24-MAR-13
Anjam Pathira                      10-JAN-20
Cold Case                          30-JUN-21
Forensic                           28-FEB-20

11 rows selected.
```

- **Create a view called MovieVotes that displays movies along with their titles and the number of votes they have received.**

```
SQL> create view movievotes as select title,votes from movies;

View created.

SQL> select * from movievotes;

TITLE                                   VOTES
---------------------------------- ----------
Rdx                                       443
premam                                  21991
Usthad Hotel                              452
Oppenheimer                               400
Voice of Sathyanathan                      90
Driven                                    104
Dejavu                                    100
Santhosham                                440
Diary                                     200
Open the door please                       90
Open grave                                100
```

```
TITLE                              VOTES
-------------------------------------- ----------
The Marshes                          89
Home                                 95
Avatar                              100
Once upon a time in Hollywood        92
Ashiqui 2                            84
Chennai express                     120
Vaaranam Aayiram                     49
Mersal                              300

19 rows selected.
```

- **Create a view called CertifiedMovies that lists movies with their titles and certificates (e.g., U/A, U).**

```
SQL> create view certifiedmovies as select title,certificate from movies;

View created.

SQL> select * from certifiedmovies;

TITLE                            CERTIFICATE
-------------------------------------- --------------------
Rdx                                U/A
premam                             U/A
Usthad Hotel                       U/A
Oppenheimer                        U/A
Voice of Sathyanathan              U
Driven                             U/A
Dejavu                             U/A
Santhosham                         U/A
Diary                              U/A
Open the door please               U/A
Open grave                         U/A

TITLE                            CERTIFICATE
-------------------------------------- --------------------
The Marshes                        A
Home                               U/A
Avatar                             A
Once upon a time in Hollywood      A
Ashiqui 2                          U
Chennai express                    U/A
Vaaranam Aayiram                   U/A
Mersal                             U/A

19 rows selected.
```

Activity # 4
4.Practice PL/SQL

****

Description : Introduction to PL/SQL

Date: 22/09/2023

****

<u>Query:</u>

- **Write a PL/SQL code block to calculate the area of a circle for a value of radius varying from 3 to 7. Store the radius and corresponding values of calculated area in an empty table named Areas, consisting of two columns Radius and Area.**

```
SQL> create table area(radius number(10,2),area number(10,2));

Table created.

declare
i number;
a number(10,2);
begin
for i in 3..7
loop
a:=3.14*i*i;
insert into Areas values(i,a);
end loop;
dbms_output.put_line('Area is :'||a);
end;


SQL> @"C:\Users\KHADEEJA C N\Desktop\plsql\area.sql"
 12  /
Area is :153.86

PL/SQL procedure successfully completed.

SQL> select * from Areas;
```

```
    RADIUS       AREA
---------- ----------
         3      28.26
         4      50.24
         5       78.5
         6     113.04
         7     153.86
```

- **Write a PL/SQL block of code for inverting a number accepted from the console.**

```
declare
num number;
rev number;
rem number;
begin
rev:=0;
num:=&num;
while num!=0
loop
rem:=mod(num,10);
rev:=rev*10+rem;
num:=trunc(num/10);
end loop;
dbms_output.put_line('Inverse is :'||rev);
end;
```

```
SQL> @"C:\Users\KHADEEJA C N\Desktop\plsql\inverse.sql"
 16  /
Enter value for num: 456
old   7: num:=&num;
new   7: num:=456;
Inverse is :654

PL/SQL procedure successfully completed.
```

- **Write a PL/SQL code block that will accept an account number from the user and debit an amount of Rs.2000 from the account if the account has a minimum balance of 500 after the amount is debited. The process is fired on the Accounts table.**

```
SQL> create table accounts(accno number(10),balance number(10,2));

Table created.
```

```
SQL> insert into accounts values(101,5000);

1 row created.

SQL> insert into accounts values(102,1000);

1 row created.

SQL> insert into accounts values(103,10000);

1 row created.

SQL> insert into accounts values(104,3000);

1 row created.

SQL> insert into accounts values(105,500);

1 row created.

SQL> select * from accounts;

     ACCNO    BALANCE
---------- ----------
       101       5000
       102       1000
       103      10000
       104       3000
       105        500


declare
ano number;
b number(10,5);
begin
ano:=&ano;
select balance into b from accounts where accno=ano;
if b-2000<500 then
dbms_output.put_line('Transaction not possible,insufficient balance');
else
update accounts set balance=balance-2000 where accno=ano;
dbms_output.put_line('Transaction successful');
end if;
end;

SQL> @"C:\Users\KHADEEJA C N\Desktop\plsql\accounts.sql"
 14  /
Enter value for ano: 101
old   5: ano:=&ano;
new   5: ano:=101;
```

```
Transaction successful

PL/SQL procedure successfully completed.

SQL> @"C:\Users\KHADEEJA C N\Desktop\plsql\accounts.sql"
 14  /
Enter value for ano: 102
old   5: ano:=&ano;
new   5: ano:=102;
Transaction not possible,insufficient balance

PL/SQL procedure successfully completed.

SQL> select * from accounts;

     ACCNO    BALANCE
---------- ----------
       101       3000
       102       1000
       103      10000
       104       3000
       105        500
```

- **Write a PL/SQL block of code that updates the salaries of Maria Jacob and Albert by Rs. 2000/- and Rs.2500/- respectively. Then check to see that the total salary does not exceed 75000. If the total salary is greater than 75000, then undo the updates made to salaries of both. (Use savepoint, rollback and commit).**

```
SQL> create table salary(id number(5),name varchar2(40),sal number(12,2));

Table created.

SQL> insert into salary values(101,'Mariya Jacob',10000);

1 row created.

SQL> insert into salary values(102,'Albert',20000);

1 row created.

SQL> insert into salary values(103,'Khadeeja',35000);

1 row created.

SQL> insert into salary values(104,'Faris',6000);
```

```
1 row created.

SQL> select * from salary;

       ID NAME                                            SAL
---------- -------------------------------------- ----------
      101 Mariya Jacob                                 10000
      102 Albert                                       20000
      103 Khadeeja                                     35000
      104 Faris                                        6000

declare
total number;
begin
savepoint s1;
update salary set sal=sal+2000 where id =102;
update salary set sal=sal+2500 where id =103;
select sum(sal) into total from salary;
dbms_output.put_line('total : '|| total);
if total>75000 then
        rollback to s1;
else
        commit;
end if;
end;

SQL> @"C:\Users\KHADEEJA C N\Desktop\plsql\salary.sql"
 13  /
total : 75500

PL/SQL procedure successfully completed.

SQL> select * from salary;

       ID NAME                                            SAL
---------- -------------------------------------- ----------
      101 Mariya Jacob                                 10000
      102 Albert                                       20000
      103 Khadeeja                                     35000
      104 Faris                                        6000
```

****

Description : Illustration of cursors

Date: 22/09/2023

****

Query:

Illustration of Implicit cursor.

- **Write a PL/SQL block to accept an employee number and update the salary of that employee to raise the salary by 0.15. Display appropriate message based on the existence of the record in the employee table.**

```
SQL> create table employee(empno number(5),empname varchar2(40),job
varchar2(40),salary number(12,2));

Table created.

SQL> insert into employee values(101,'Khadeeja','analyst',50000);

1 row created.

SQL> insert into employee values(102,'Faris','developer',100000);

1 row created.

SQL> insert into employee values(103,'Sahala','designer',40000);

1 row created.

SQL>  insert into employee values(104,'Aparna','analyst',60000);

1 row created.

SQL> insert into employee values(105,'Arun','engineer',80000);

1 row created.

SQL> select * from employee;

     EMPNO EMPNAME                                  JOB                                          SALARY
---------- ---------------------------------------- ---------------------------------------- ----------
       101 Khadeeja                                 analyst                                       50000
       102 Faris                                    developer                                    100000
       103 Sahala                                   designer                                      40000
```

```
      104 Aparna                              analyst                          60000
      105 Arun                                analyst                          80000
```

```
declare
begin
update employee set salary=salary+(salary*0.15) where empno=&empno;
if sql%found then
dbms_output.put_line(sql%rowcount ||'record is updated');
end if;
end;
```

```
SQL> @"C:\Users\KHADEEJA C N\Desktop\plsql\implicit1.sql"
  8  /
Enter value for empno: 101
old   3: update employee set salary=salary+(salary*0.15) where empno=&empno;
new   3: update employee set salary=salary+(salary*0.15) where empno=101;
1record is updated

PL/SQL procedure successfully completed.

SQL> select * from employee;

    EMPNO EMPNAME                            JOB                              SALARY
---------- ---------------------------------------- ---------------------------------------- ----------
      101 Khadeeja                             analyst                         57500
      102 Faris                               developer                        100000
      103 Sahala                              designer                         40000
      104 Aparna                              analyst                          60000
      105 Arun                                analyst                          80000
```

- **The HRD manager decides to raise the salary of employees working as 'analyst' by 0.15. Write a cursor to update the salary of the employees. Display the no. of employee records that has been modified.**

```
declare
begin
update employee set salary=salary+(salary*0.15) where job='analyst';
if sql%found then
dbms_output.put_line(sql%rowcount ||'record is updated');
end if;
end;
```

```
SQL> @"C:\Users\KHADEEJA C N\Desktop\plsql\implicit2.sql"
  8  /
3record is updated

PL/SQL procedure successfully completed.
```

```
SQL> select * from employee;

    EMPNO EMPNAME                              JOB                                     SALARY
---------- -------------------------------------- -------------------------------------- ----------
       101 Khadeeja                          analyst                                  66125
       102 Faris                             developer                               100000
       103 Sahala                            designer                                 40000
       104 Aparna                            analyst                                  69000
       105 Arun                              analyst                                  92000
```

Illustration of explicit cursor.

- **Write an explicit cursor to display the name,department, salary of the first 5 employees getting the highest salary.**

```
declare
cursor empcursor is select empname,deptname,salary from (select
empname,deptname,salary from employee e,department d where e.deptno=d.deptno order
by salary desc) where rownum<=5;
vemp empcursor%rowtype;
begin
dbms_output.put_line('******employee details******');
open empcursor;
fetch empcursor into vemp;
while empcursor%found
loop
dbms_output.put_line('Empname:'|| vemp.empname);
dbms_output.put_line('Empname:'|| vemp.deptname);
dbms_output.put_line('Salary:'|| vemp.salary);
dbms_output.put_line('*********************************');
fetch empcursor into vemp;
end loop;
close empcursor;
end;

SQL> @ C:\Users\CCL35\Desktop\pls\expilicit_record1.sql;
 50  /
******employee details******
Empname:ARNOLD LEONARD AMON
Empname:COMPUTER SERVICE DIVISION
Salary:152750
*********************************
Empname:DONA ANICE SIBY
Empname:COMPUTER SERVICE DIVISION
Salary:46500
*********************************
Empname:PHILIP VINCENT
```

```
Empname:PLANNING
Salary:41250
*********************************
Empname:ALFRIN LUIZ
Empname:SUPPORT SERVICES
Salary:40175
*********************************
Empname:SHILVY K K
Empname:INFORMATION CENTER
Salary:39250
*********************************

PL/SQL procedure successfully completed.
```

- **The HRD manager decides to raise the salary of employees working as 'analyst' by 0.15. Whenever any such raise is given to the employees, a record for the same is maintained in the emp_raise table. It includes the employee number, the date when the raise was given and actual raise. Write a PL/SQL block to update the salary of the employees and insert a record in the emp_raise table.**

  Emp_raise(empcode, raisedate,raise_amt)

```
declare
v_raise_amt number;
v_raisedate date:=sysdate;
cursor empcursor is select * from employee where job='analyst';
vemp empcursor%rowtype;
begin
open empcursor;
fetch empcursor into vemp;
while empcursor%found
loop
update employee set salary=salary+salary*0.15 where empno=vemp.empno;
insert into emp_raise(empcode,raisedate,raise_amt)
values(vemp.empno,v_raisedate,vemp.salary*0.15);
fetch empcursor into vemp;
end loop;
close empcursor;
dbms_output.put_line('salary updated and records inserted in the new table successfully');
end;

SQL> @ "C:\Users\KHADEEJA C N\Desktop\record
2023_dbms\plsql\explicit_record2.sql"
 18  /
```

```
PL/SQL procedure successfully completed.

SQL> select * from employee;

EMPNO          EMPNAME              JOB           SALARY        DEPARTMENT
---------- -------------------------------------- -------------------------------------- ---------- -----
 101            Khadeeja             analyst        69000          marketing
 102            Faris                developer      100000         IT
 103            Sahala               designer       40000          production
 104           Aparna                analyst        80500          marketing
 105           Arun                 analyst         11500          marketing
 106           Mariya                engineer       75000         IT
 107           Ravi                  clerk          45000          finance

7 rows selected.

SQL> select * from emp_raise;

EMPCODE       RAISEDATE  RAISE_AMT
--------------- --------- ----------
101              23-SEP-23      9000
104              23-SEP-23     10500
105              23-SEP-23      1500
```