Final Project Proposal
Employee Payroll Management System

Name SMART DARE **NUID: 001826636**

Name of Project: Employee Payroll Management System

Summary:

1. Employee Information

Employee data is very essential in order to maintain a proper record of the employees and there personal information for various purposes like contacting them for inviting for certain summit, feedback of the company from the employee data

2. Maintaining Salary

Very important to keep this data which will help not only the managers and the HR to keep a track of the employee salaries but also help the company or its board to analyze what amount they are spending on a particular employee of a particular company

3. Work Location

It is very much important for an organization small or big to have a record of all the work locations they operate from to see how they can develop in that particular region and also increase the hiring in that region so that the organization can increase there Market Outreach that area.

4. Projects

In order to be successful company should be involved in various projects, so they also need to maintain the record of the salaries each employee is being paid for a particular type of project he/she is working on

PL/SQL features used in the project:

1. Created Explicit Cursors which shows the hourly pay of the employees associated with there Accounts and Ref cursor showing the employees who are a part of a particular department

2. Create a CDB and a PDB with users to manage the data according to the area of interest

3. Implement pre-defined exception cursor_already_open to demonstrate the understanding of the exceptional handling concept which shows what error will populate when we try to open a cursor which is already open

4. Also, created Relational, Inline and Materialized Views satisfying various business requirements

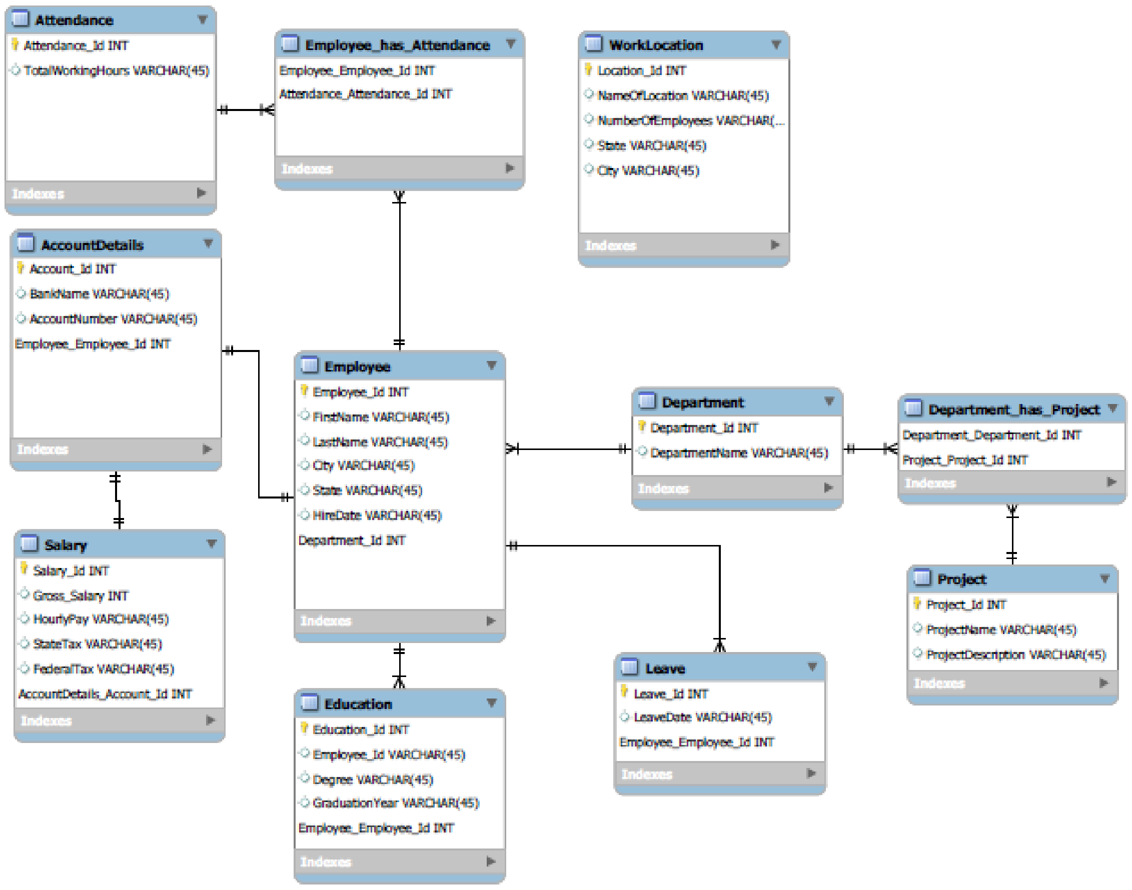
5. Created Index on AccountDetails table

6. Built an E-R Diagram to know how the entities are related in the payroll management system for any company

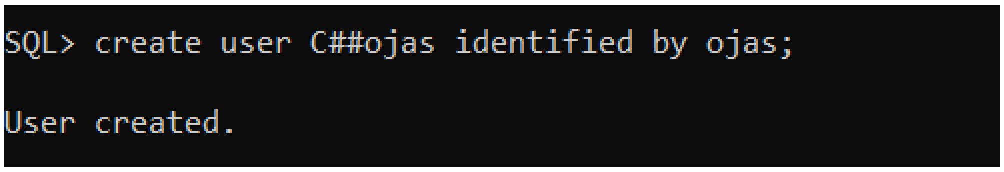
List of Entities:

Employee
Employee table will include all the personal details of the employee and would be very much cover overall information of that particular employee
Salary
Salary Table will cover all the current and previous salaries an employee had or currently has. This table will help a manager/ an HR to analyze which employee has been given promotion on which date or when did his salary grade changed
Department
Department Table maintains the data of the all the possible departments an employee can belong to
Account Details
Account Details Table will maintain the data regarding the accounts which the employee has connected with the company for his/her salary to be credited
Attendance
This table includes all the data of the employees attendance which includes the number of hours an employee has worked in a week
Project
This table includes the data of all the projects a particular company is working on or the projects on which the company is going to work in the future
Education
The Education Table keeps the track of the education of the employee including his degrees achieved until now
Work Location
The name of the table tells you most of the things. This table includes the location of the office, which city is it located, which state it is in and also tracks the number of employees in a particular location
Leave
Leave table keeps the record of the number of leaves an employee takes or has taken over the course of any month or an year

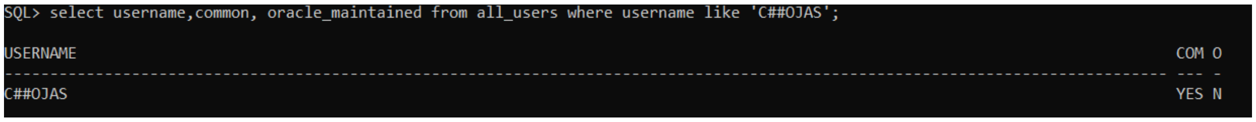
E-R Diagram


Attendance
Attendance_Id INT
TotalWorkingHours VARCHAR(45)
Indexes
Employee_has_Attendance
Employee_Employee_Id INT
Attendance_Attendance_Id INT
Indexes
WorkLocation
Location_Id INT
NameOfLocation VARCHAR(45)
NumberOfEmployees VARCHAR(...
State VARCHAR(45)
City VARCHAR(45)
Indexes
AccountDetails
Account_Id INT
BankName VARCHAR(45)
AccountNumber VARCHAR(45)
Employee_Employee_Id INT
Indexes
Employee
Employee_Id INT
FirstName VARCHAR(45)
LastName VARCHAR(45)
City VARCHAR(45)
State VARCHAR(45)
HireDate VARCHAR(45)
Department_Id INT
Indexes
Department
Department_Id INT
DepartmentName VARCHAR(45)
Indexes
Department_has_Project
Department_Department_Id INT
Project_Project_Id INT
Indexes
Salary
Salary_Id INT
Gross_Salary INT
HourlyPay VARCHAR(45)
StateTax VARCHAR(45)
FederalTax VARCHAR(45)
AccountDetails_Account_Id INT
Indexes
Project
Project_Id INT
ProjectName VARCHAR(45)
ProjectDescription VARCHAR(45)
Indexes
Leave
Leave_Id INT
LeaveDate VARCHAR(45)
Employee_Employee_Id INT
Indexes
Education
Education_Id INT
Employee_Id VARCHAR(45)
Degree VARCHAR(45)
GraduationYear VARCHAR(45)
Employee_Employee_Id INT
Indexes

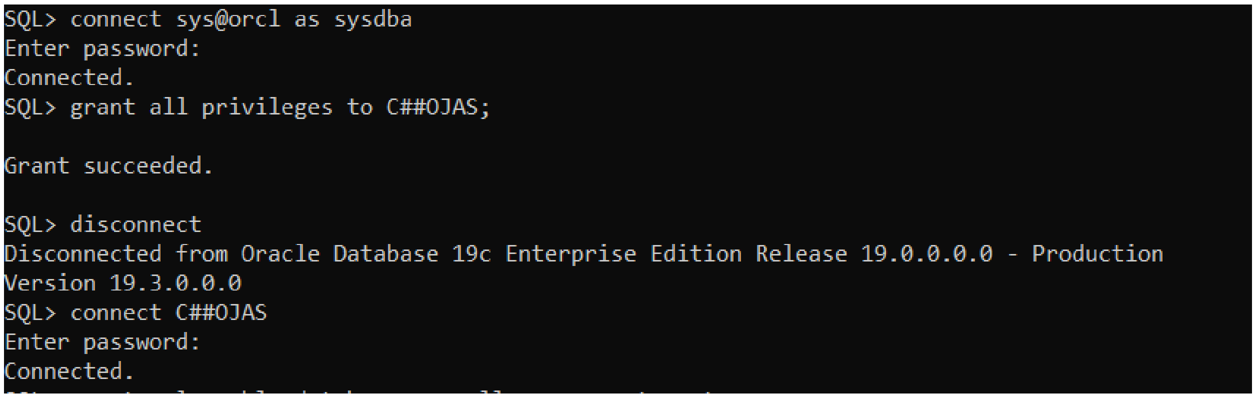
1.Created Common User on sysdba



```
SQL> create user C##ojas identified by ojas;

User created.
```

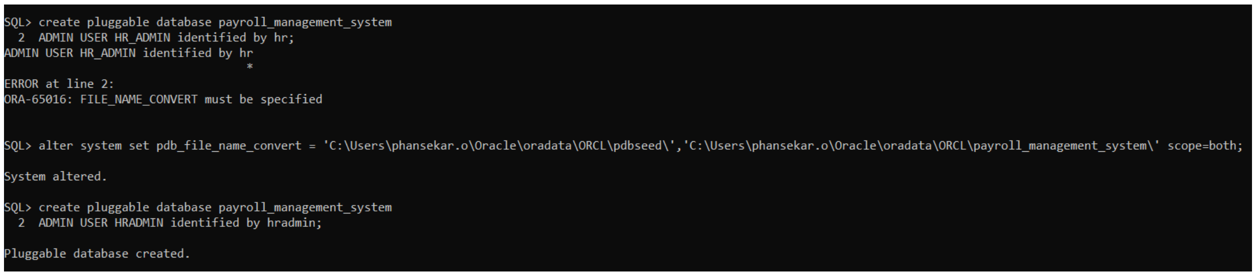


```
SQL> select username,common, oracle_maintained from all_users where username like 'C##OJAS';

USERNAME                                                                                                                   COM O
-------------------------------------------------------------------------------------------------------------------------- --- -
C##OJAS                                                                                                                    YES N
```

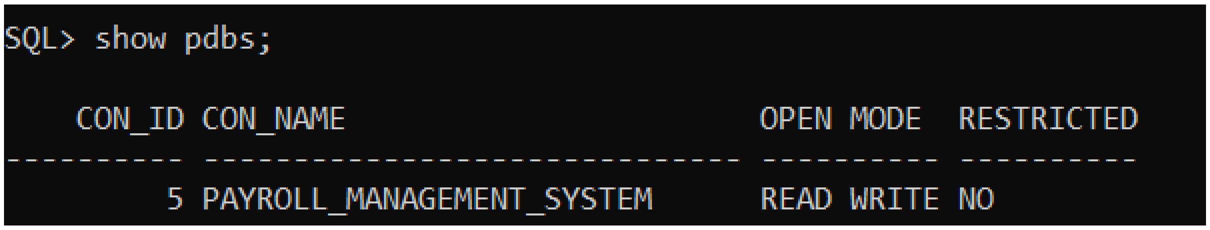


```
SQL> connect sys@orcl as sysdba
Enter password:
Connected.
SQL> grant all privileges to C##OJAS;

Grant succeeded.

SQL> disconnect
Disconnected from Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.3.0.0.0
SQL> connect C##OJAS
Enter password:
Connected.
```

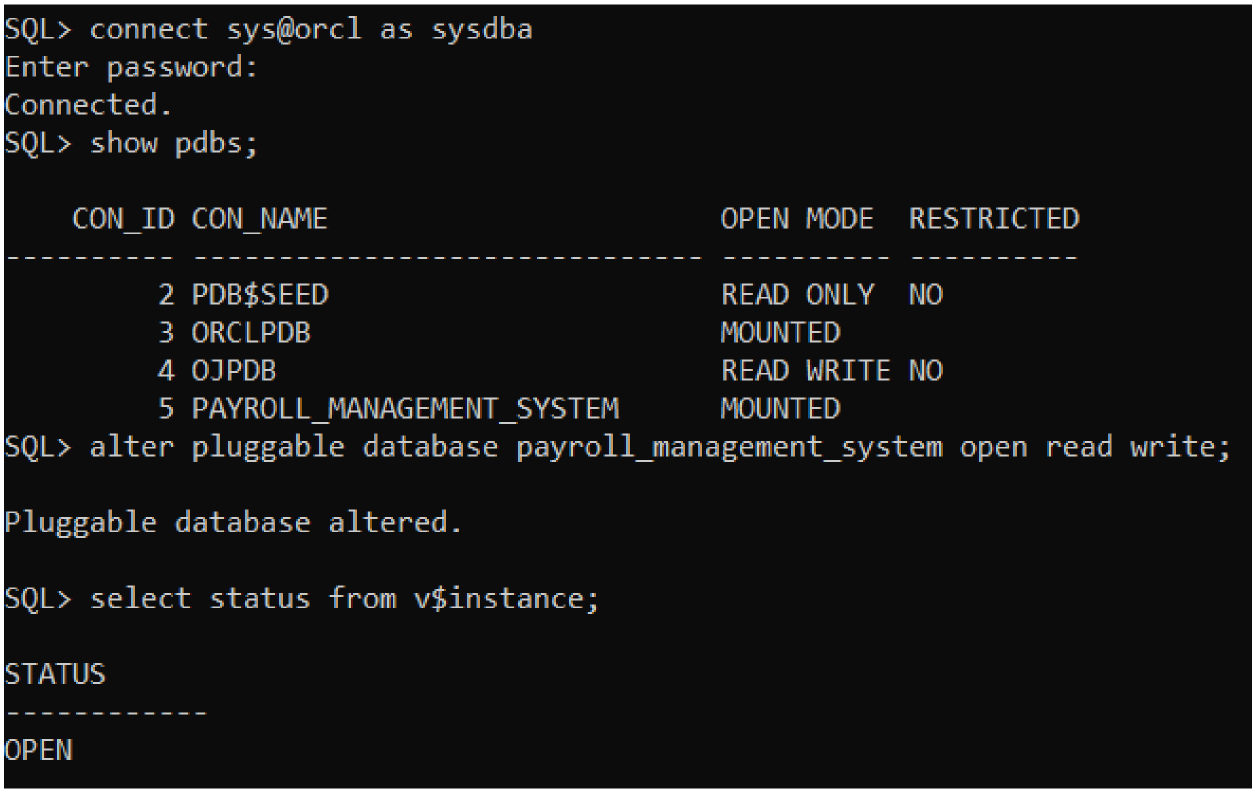
2. Create Pluggable Database



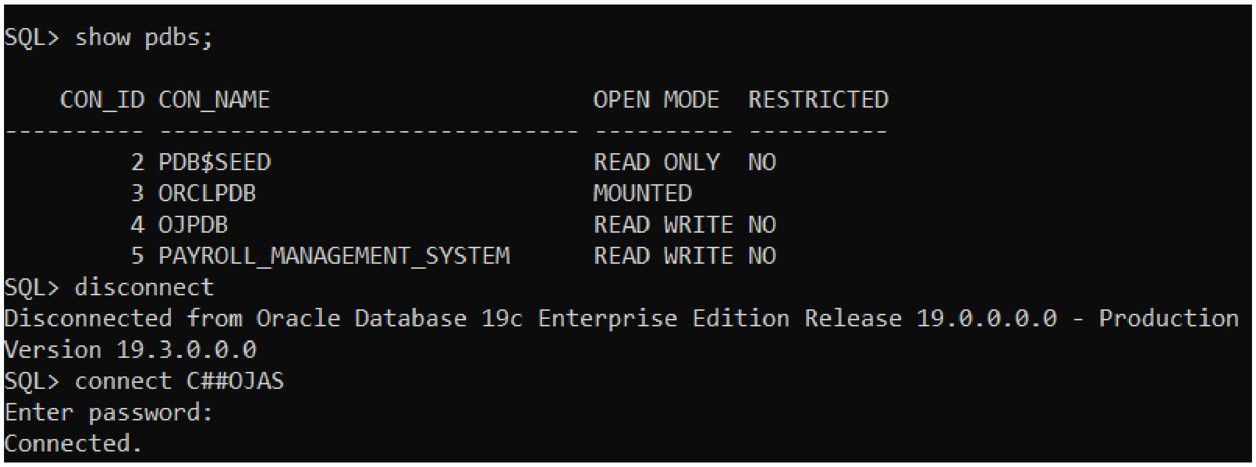
```
SQL> create pluggable database payroll_management_system
  2  ADMIN USER HR_ADMIN identified by hr;
ADMIN USER HR_ADMIN identified by hr
                                   *
ERROR at line 2:
ORA-65016: FILE_NAME_CONVERT must be specified


SQL> alter system set pdb_file_name_convert = 'C:\Users\phansekar.o\Oracle\oradata\ORCL\pdbseed\','C:\Users\phansekar.o\Oracle\oradata\ORCL\payroll_management_system\' scope=both;

System altered.

SQL> create pluggable database payroll_management_system
  2  ADMIN USER HRADMIN identified by hradmin;

Pluggable database created.
```



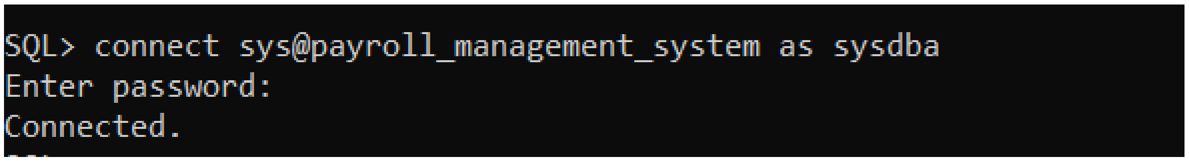
```
SQL> show pdbs;

    CON_ID CON_NAME                       OPEN MODE  RESTRICTED
---------- ------------------------------ ---------- ----------
         5 PAYROLL_MANAGEMENT_SYSTEM      READ WRITE NO
```



```
SQL> connect sys@orcl as sysdba
Enter password:
Connected.
SQL> show pdbs;

    CON_ID CON_NAME                       OPEN MODE  RESTRICTED
---------- ------------------------------ ---------- ----------
         2 PDB$SEED                       READ ONLY  NO
         3 ORCLPDB                        MOUNTED
         4 OJPDB                          READ WRITE NO
         5 PAYROLL_MANAGEMENT_SYSTEM      MOUNTED
SQL> alter pluggable database payroll_management_system open read write;

Pluggable database altered.

SQL> select status from v$instance;

STATUS
------------
OPEN
```

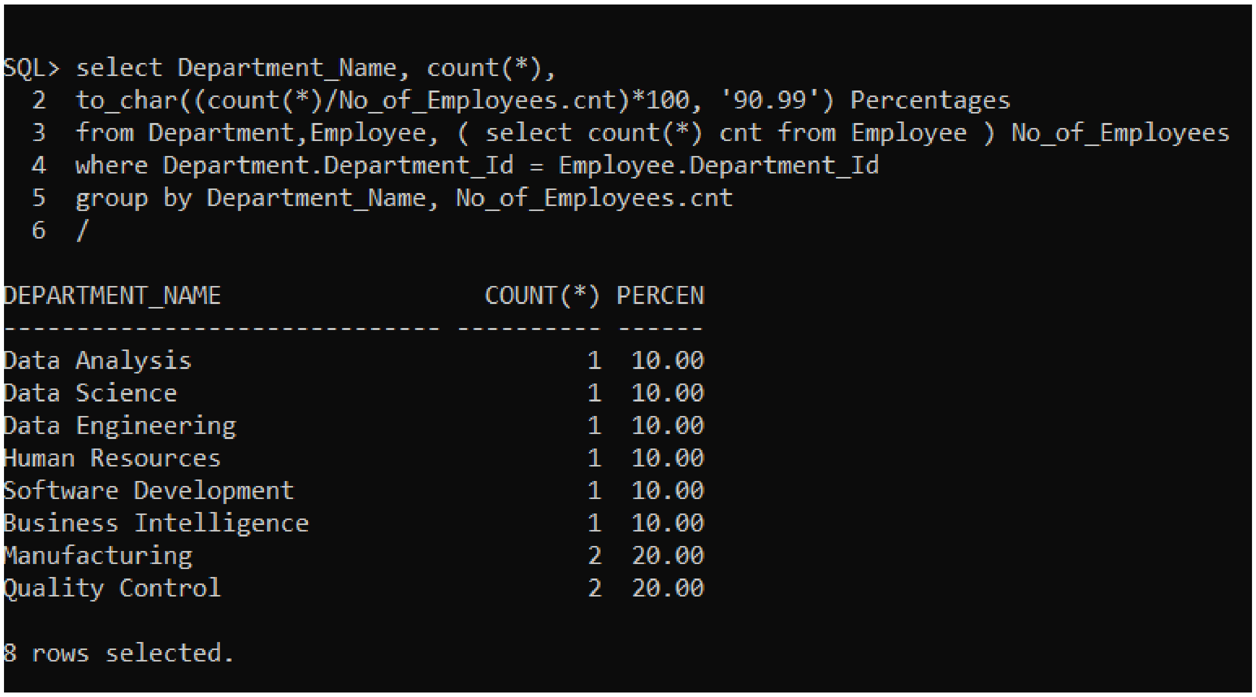


```
SQL> show pdbs;

    CON_ID CON_NAME                       OPEN MODE  RESTRICTED
---------- ------------------------------ ---------- ----------
         2 PDB$SEED                       READ ONLY  NO
         3 ORCLPDB                        MOUNTED
         4 OJPDB                          READ WRITE NO
         5 PAYROLL_MANAGEMENT_SYSTEM      READ WRITE NO
SQL> disconnect
Disconnected from Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.3.0.0.0
SQL> connect C##OJAS
Enter password:
Connected.
```



```
SQL> connect sys@payroll_management_system as sysdba
Enter password:
Connected.
```

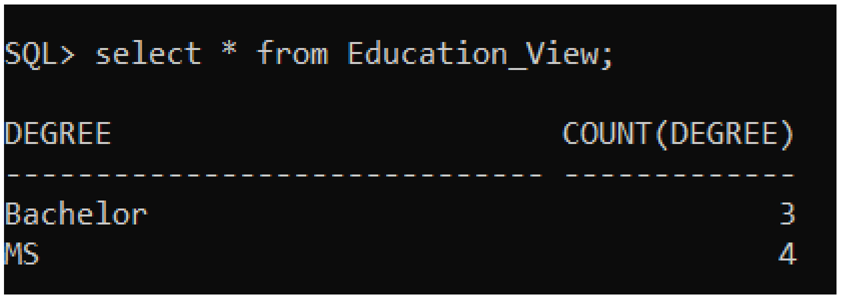
3. Inline View



```
SQL> select Department_Name, count(*),
  2  to_char((count(*)/No_of_Employees.cnt)*100, '90.99') Percentages
  3  from Department,Employee, ( select count(*) cnt from Employee ) No_of_Employees
  4  where Department.Department_Id = Employee.Department_Id
  5  group by Department_Name, No_of_Employees.cnt
  6  /

DEPARTMENT_NAME                  COUNT(*) PERCEN
------------------------------ ---------- ------
Data Analysis                           1  10.00
Data Science                            1  10.00
Data Engineering                        1  10.00
Human Resources                         1  10.00
Software Development                    1  10.00
Business Intelligence                   1  10.00
Manufacturing                           2  20.00
Quality Control                         2  20.00

8 rows selected.
```

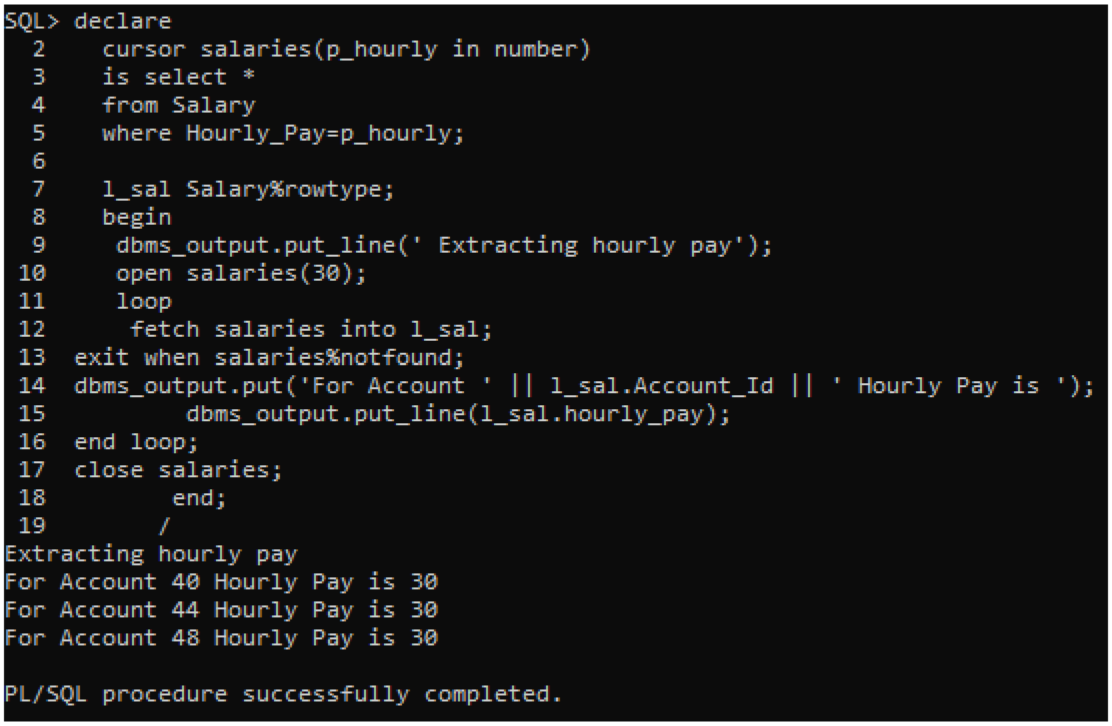
4. Materialized Views

-- Number of Employees with different degrees



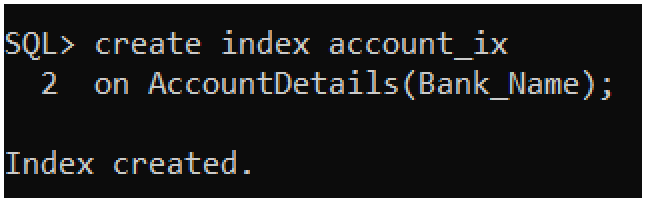
```
SQL> select * from Education_View;

DEGREE                         COUNT(DEGREE)
------------------------------ -------------
Bachelor                                   3
MS                                         4
```

5. Explicit Cursor



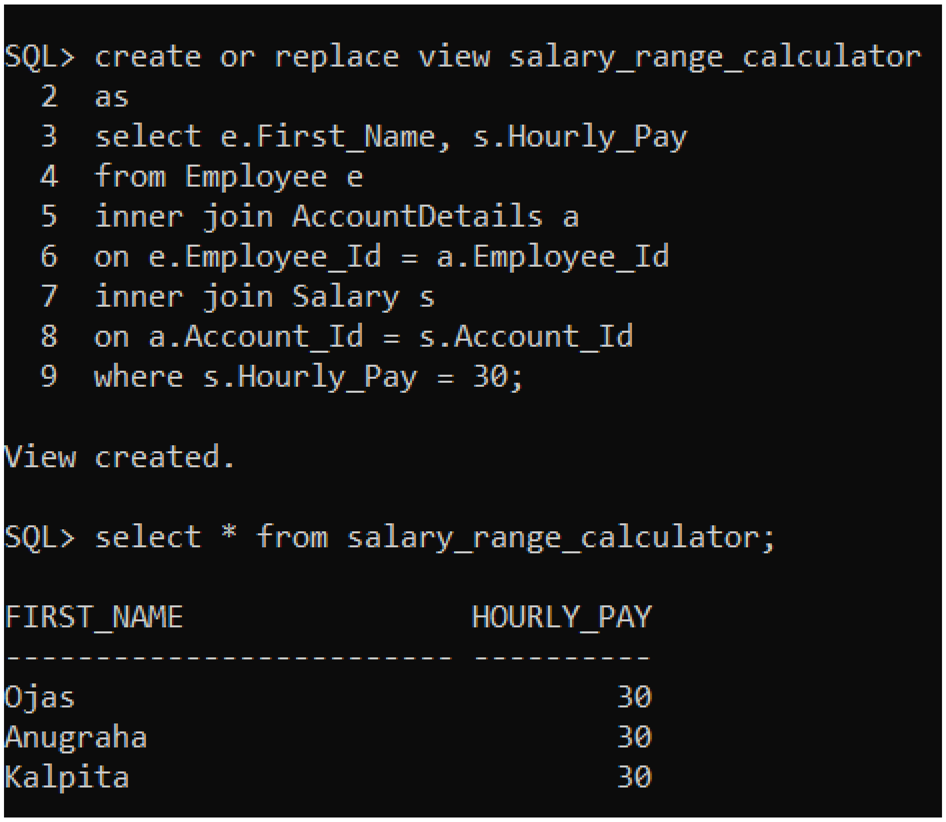
```
SQL> declare
       cursor salaries(p_hourly in number)
       is select *
       from Salary
       where Hourly_Pay=p_hourly;

       l_sal Salary%rowtype;
       begin
        dbms_output.put_line(' Extracting hourly pay');
        open salaries(30);
        loop
         fetch salaries into l_sal;
     exit when salaries%notfound;
     dbms_output.put('For Account ' || l_sal.Account_Id || ' Hourly Pay is ');
             dbms_output.put_line(l_sal.hourly_pay);
     end loop;
     close salaries;
            end;
           /
Extracting hourly pay
For Account 40 Hourly Pay is 30
For Account 44 Hourly Pay is 30
For Account 48 Hourly Pay is 30

PL/SQL procedure successfully completed.
```

6. Index



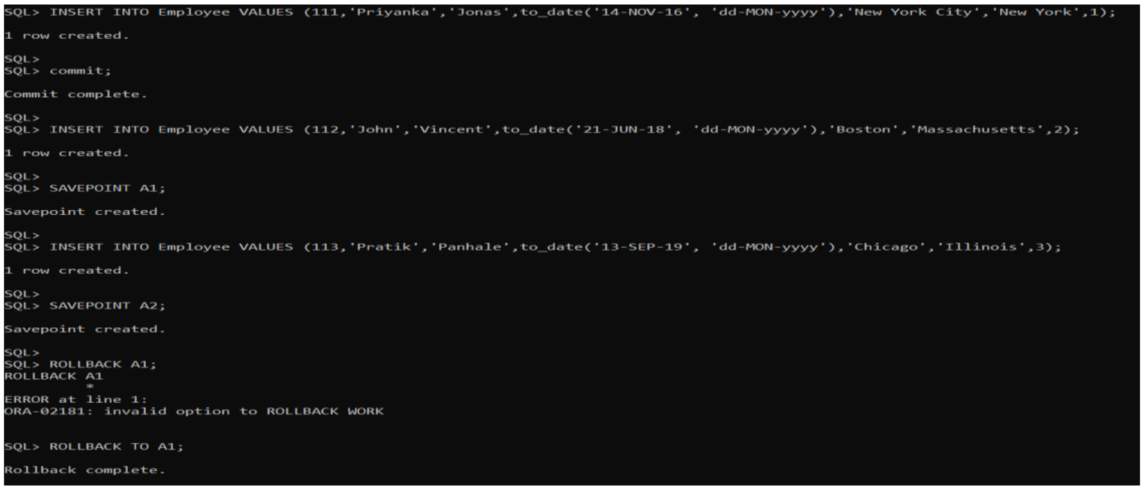
```
SQL> create index account_ix
     on AccountDetails(Bank_Name);

Index created.
```

7. Relational Views



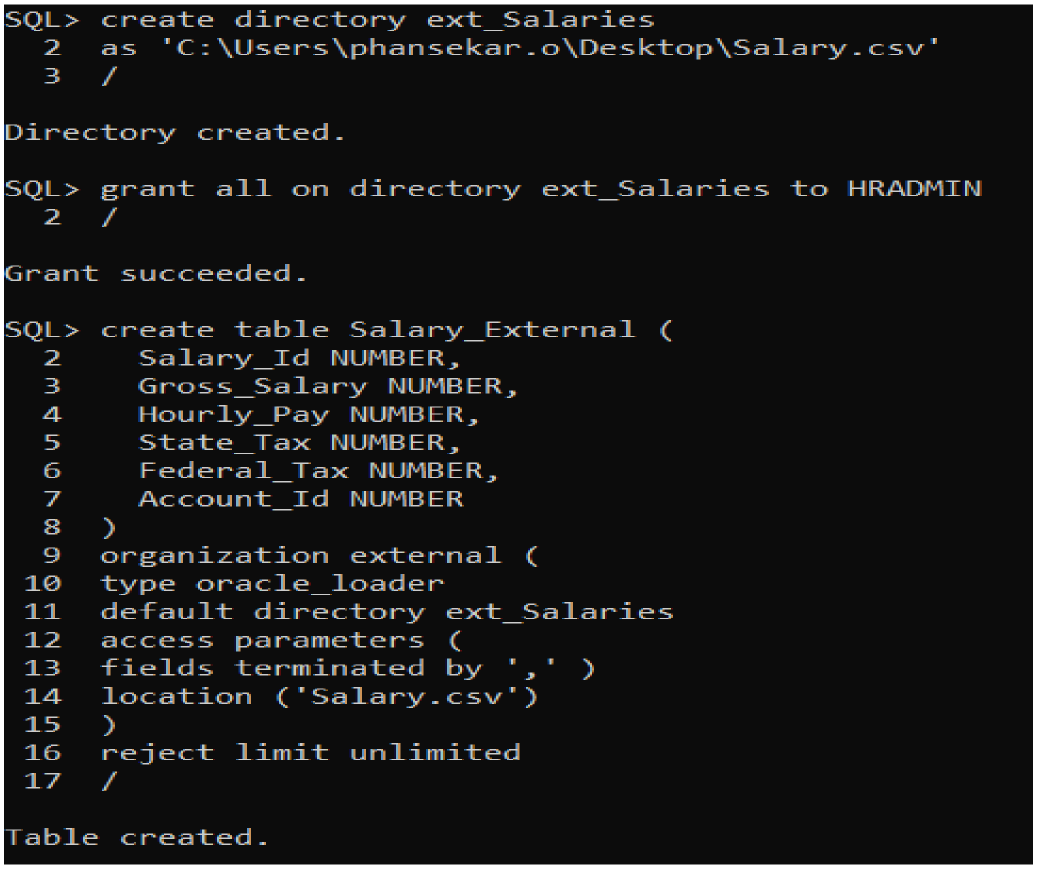
```
SQL> create or replace view salary_range_calculator
     as
     select e.First_Name, s.Hourly_Pay
     from Employee e
     inner join AccountDetails a
     on e.Employee_Id = a.Employee_Id
     inner join Salary s
     on a.Account_Id = s.Account_Id
     where s.Hourly_Pay = 30;

View created.

SQL> select * from salary_range_calculator;

FIRST_NAME                HOURLY_PAY
------------------------- ----------
Ojas                              30
Anugraha                          30
Kalpita                           30
```

8. Transaction

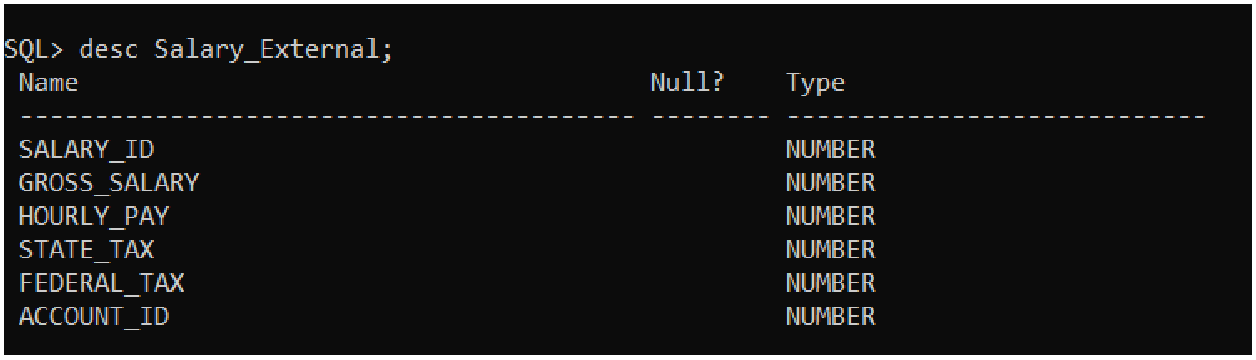


```
SQL> INSERT INTO Employee VALUES (111,'Priyanka','Jonas',to_date('14-NOV-16', 'dd-MON-yyyy'),'New York City','New York',1);

1 row created.

SQL>
SQL> commit;

Commit complete.

SQL>
SQL> INSERT INTO Employee VALUES (112,'John','Vincent',to_date('21-JUN-18', 'dd-MON-yyyy'),'Boston','Massachusetts',2);

1 row created.

SQL>
SQL> SAVEPOINT A1;

Savepoint created.

SQL>
SQL> INSERT INTO Employee VALUES (113,'Pratik','Panhale',to_date('13-SEP-19', 'dd-MON-yyyy'),'Chicago','Illinois',3);

1 row created.

SQL>
SQL> SAVEPOINT A2;

Savepoint created.

SQL>
SQL> ROLLBACK A1;
ROLLBACK A1
         *
ERROR at line 1:
ORA-02181: invalid option to ROLLBACK WORK


SQL> ROLLBACK TO A1;

Rollback complete.
```

9. External Table

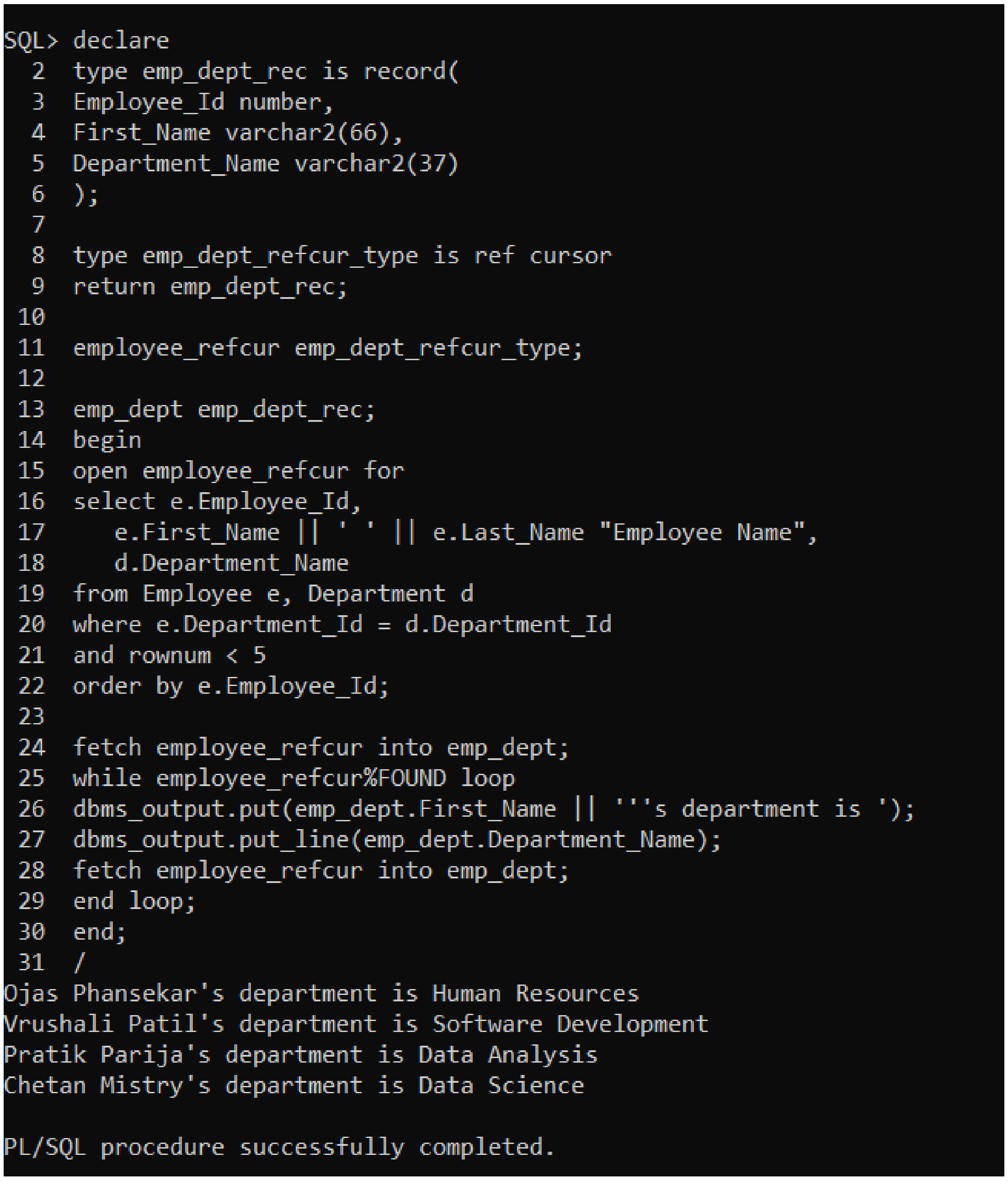


```
SQL> create directory ext_Salaries
     as 'C:\Users\phansekar.o\Desktop\Salary.csv'
     /

Directory created.

SQL> grant all on directory ext_Salaries to HRADMIN
     /

Grant succeeded.

SQL> create table Salary_External (
       Salary_Id NUMBER,
       Gross_Salary NUMBER,
       Hourly_Pay NUMBER,
       State_Tax NUMBER,
       Federal_Tax NUMBER,
       Account_Id NUMBER
     )
     organization external (
     type oracle_loader
     default directory ext_Salaries
     access parameters (
     fields terminated by ',' )
     location ('Salary.csv')
     )
     reject limit unlimited
     /

Table created.
```



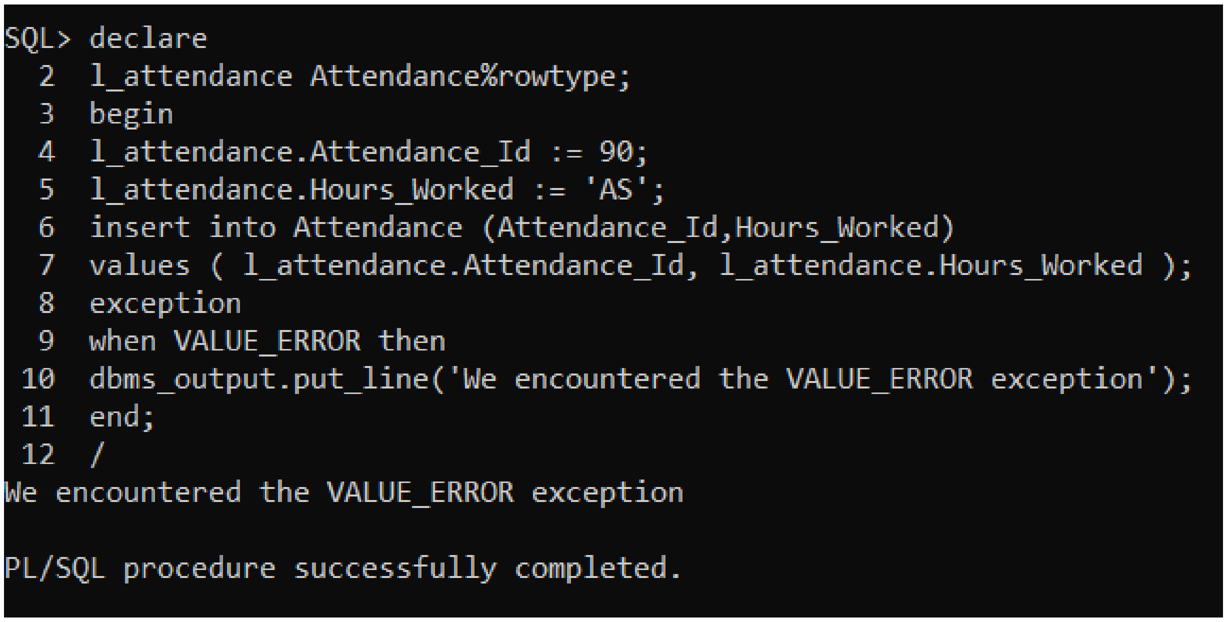
```
SQL> desc Salary_External;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SALARY_ID                                          NUMBER
 GROSS_SALARY                                       NUMBER
 HOURLY_PAY                                         NUMBER
 STATE_TAX                                          NUMBER
 FEDERAL_TAX                                        NUMBER
 ACCOUNT_ID                                         NUMBER
```

10. Ref cursor



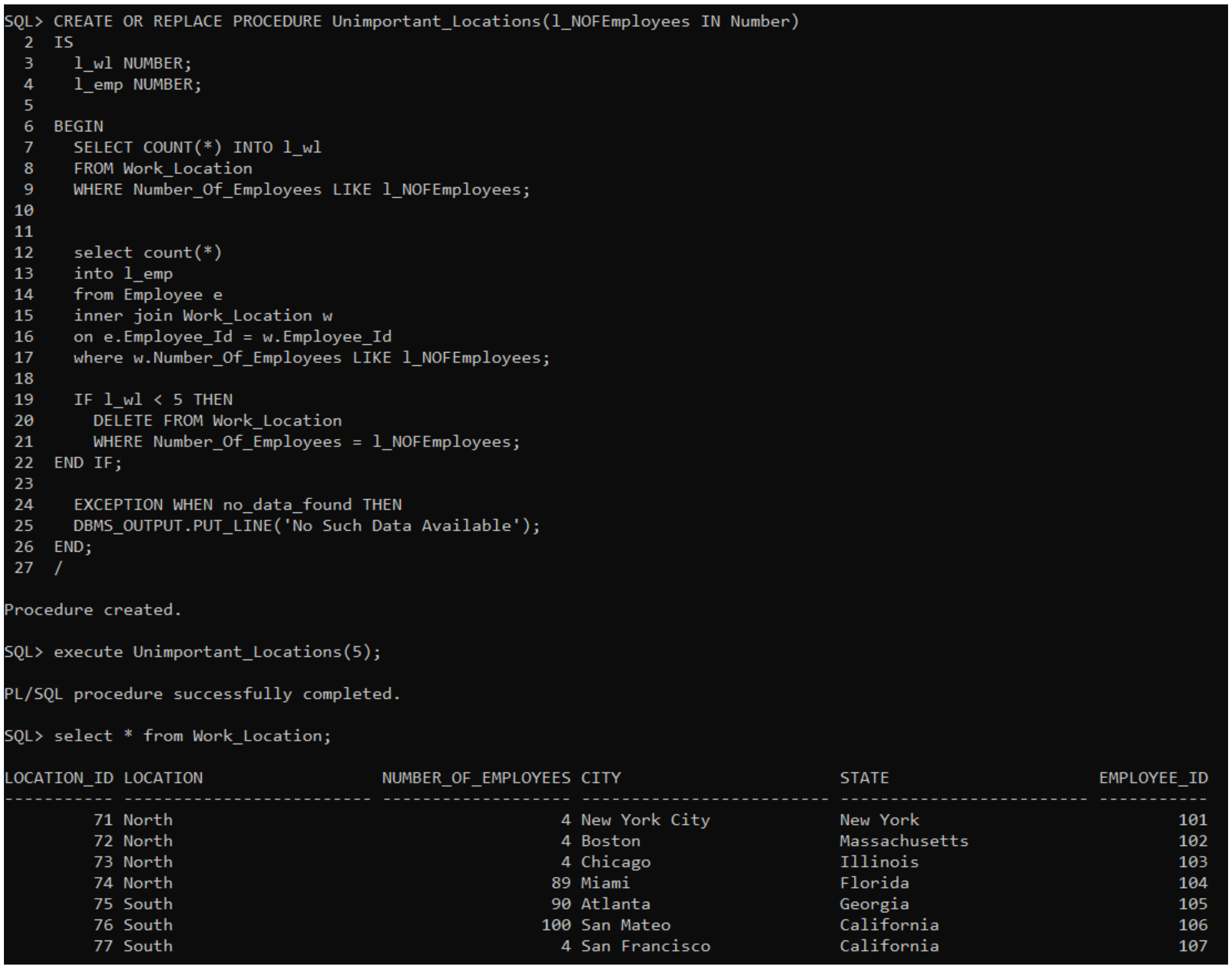
```
SQL> declare
type emp_dept_rec is record(
Employee_Id number,
First_Name varchar2(66),
Department_Name varchar2(37)
);

type emp_dept_refcur_type is ref cursor
return emp_dept_rec;

employee_refcur emp_dept_refcur_type;

emp_dept emp_dept_rec;
begin
open employee_refcur for
select e.Employee_Id,
   e.First_Name || ' ' || e.Last_Name "Employee Name",
   d.Department_Name
from Employee e, Department d
where e.Department_Id = d.Department_Id
and rownum < 5
order by e.Employee_Id;

fetch employee_refcur into emp_dept;
while employee_refcur%FOUND loop
dbms_output.put(emp_dept.First_Name || '''s department is ');
dbms_output.put_line(emp_dept.Department_Name);
fetch employee_refcur into emp_dept;
end loop;
end;
/
Ojas Phansekar's department is Human Resources
Vrushali Patil's department is Software Development
Pratik Parija's department is Data Analysis
Chetan Mistry's department is Data Science

PL/SQL procedure successfully completed.
```

11. Pre-defined Exception



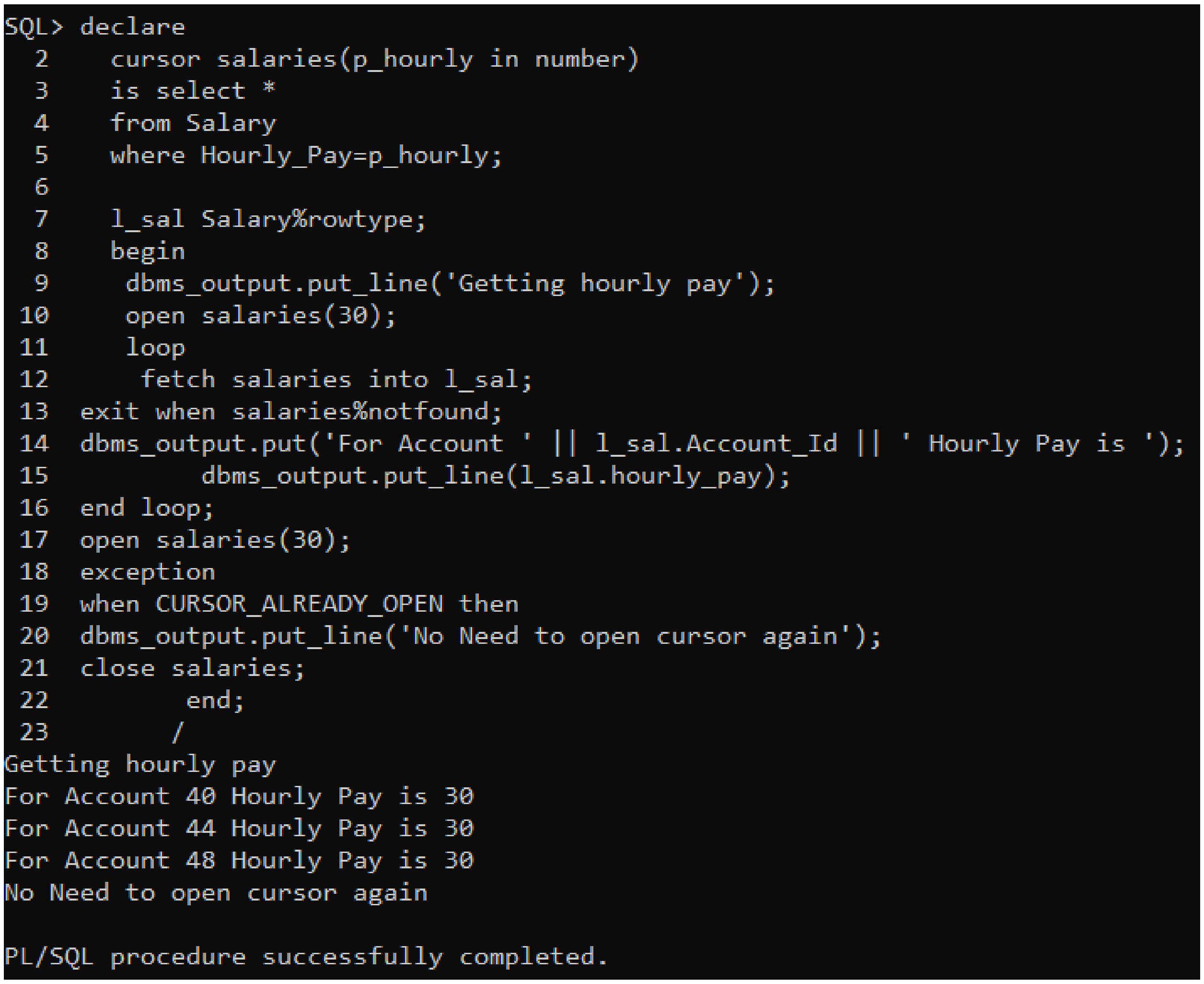
```
SQL> declare
l_attendance Attendance%rowtype;
begin
l_attendance.Attendance_Id := 90;
l_attendance.Hours_Worked := 'AS';
insert into Attendance (Attendance_Id,Hours_Worked)
values ( l_attendance.Attendance_Id, l_attendance.Hours_Worked );
exception
when VALUE_ERROR then
dbms_output.put_line('We encountered the VALUE_ERROR exception');
end;
/
We encountered the VALUE_ERROR exception

PL/SQL procedure successfully completed.
```

12. Procedure



```
SQL> CREATE OR REPLACE PROCEDURE Unimportant_Locations(l_NOFEmployees IN Number)
     IS
       l_wl NUMBER;
       l_emp NUMBER;

     BEGIN
       SELECT COUNT(*) INTO l_wl
       FROM Work_Location
       WHERE Number_Of_Employees LIKE l_NOFEmployees;


       select count(*)
       into l_emp
       from Employee e
       inner join Work_Location w
       on e.Employee_Id = w.Employee_Id
       where w.Number_Of_Employees LIKE l_NOFEmployees;

       IF l_wl < 5 THEN
         DELETE FROM Work_Location
         WHERE Number_Of_Employees = l_NOFEmployees;
     END IF;

       EXCEPTION WHEN no_data_found THEN
       DBMS_OUTPUT.PUT_LINE('No Such Data Available');
     END;
     /

Procedure created.

SQL> execute Unimportant_Locations(5);

PL/SQL procedure successfully completed.

SQL> select * from Work_Location;

LOCATION_ID LOCATION                  NUMBER_OF_EMPLOYEES CITY                      STATE                     EMPLOYEE_ID
----------- ------------------------- ------------------- ------------------------- ------------------------- -----------
         71 North                                       4 New York City             New York                          101
         72 North                                       4 Boston                    Massachusetts                     102
         73 North                                       4 Chicago                   Illinois                          103
         74 North                                      89 Miami                     Florida                           104
         75 South                                      90 Atlanta                   Georgia                           105
         76 South                                     100 San Mateo                 California                        106
         77 South                                       4 San Francisco             California                        107
```

13. Predefined Exception and Explicit Cursor



```
SQL> declare
       cursor salaries(p_hourly in number)
       is select *
       from Salary
       where Hourly_Pay=p_hourly;

       l_sal Salary%rowtype;
       begin
        dbms_output.put_line('Getting hourly pay');
        open salaries(30);
        loop
         fetch salaries into l_sal;
     exit when salaries%notfound;
     dbms_output.put('For Account ' || l_sal.Account_Id || ' Hourly Pay is ');
             dbms_output.put_line(l_sal.hourly_pay);
     end loop;
     open salaries(30);
     exception
     when CURSOR_ALREADY_OPEN then
     dbms_output.put_line('No Need to open cursor again');
     close salaries;
            end;
           /
Getting hourly pay
For Account 40 Hourly Pay is 30
For Account 44 Hourly Pay is 30
For Account 48 Hourly Pay is 30
No Need to open cursor again

PL/SQL procedure successfully completed.
```

Appendix:

Create Table Statements

Employee

```
--------------------------------------------------
CREATE TABLE Employee(
  Employee_Id NUMBER(6),
  First_Name VARCHAR2(25),
  Last_Name VARCHAR2(25),
  Hire_Date DATE,
  City VARCHAR2(25),
  State VARCHAR2(25),
  CONSTRAINT EMPLOYEE_PK PRIMARY KEY (Employee_Id));
--------------------------------------------------
```

Department

```
--------------------------------------------------
  CREATE TABLE Department(
  Department_Id NUMBER,
  Department_Name VARCHAR2(30),
  CONSTRAINT DEPARTMENT_PK PRIMARY KEY (Department_Id)
  );

  ------------------------------------------------
```

Salary

```
  ------------------------------------------------

  CREATE TABLE Salary(
  Salary_Id NUMBER,
  Gross_Salary NUMBER,

  Hourly_Pay NUMBER,

  State_Tax NUMBER,

  Federal_Tax NUMBER,

  Account_Id NUMBER,

  CONSTRAINT SALARY_PK PRIMARY KEY (Salary_Id),

  FOREIGN KEY (Account_Id)

      REFERENCES ACCOUNTDETAILS(Account_Id)

  );


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

  DepartmentProject Bridge

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

  CREATE TABLE DepartmentProject(

  Department_Id NUMBER,

  Project_Id NUMBER,

  CONSTRAINT DEPTPROJECT_PK PRIMARY KEY (Department_Id,Project_Id),

  FOREIGN KEY (Department_Id)

      REFERENCES Department(Department_Id),

  FOREIGN KEY (Project_Id)

      REFERENCES Project(Project_Id)

  );

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

  Project

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

  CREATE TABLE Project(

  Project_Id NUMBER,

  Project_Name VARCHAR2(50),

  Project_Description VARCHAR2(50),
  CONSTRAINT Project_PK PRIMARY KEY (Project_Id)
  );

  --------------------------------------------------
  AccountDetails
  --------------------------------------------------

  CREATE TABLE AccountDetails(
  Account_Id NUMBER,
  Bank_Name VARCHAR2(50),
  Account_Number VARCHAR2(50),
  Employee_Id NUMBER,
  CONSTRAINT Account_PK PRIMARY KEY (Account_Id),
  FOREIGN KEY (Employee_Id)
      REFERENCES Employee(Employee_Id)
  );
  --------------------------------------------------
  Education
  --------------------------------------------------
  CREATE TABLE Education(
  Education_Id NUMBER,
  Employee_Id NUMBER,
  Degree VARCHAR(30),
  Graduation_Year NUMBER(4),
  CONSTRAINT Location_PK PRIMARY KEY (Education_Id),
  FOREIGN KEY (Employee_Id)
      REFERENCES Employee(Employee_Id)
  );
  --------------------------------------------------
```

Leave

```
--------------------------------------------------


CREATE TABLE Leave(
Leave_Id NUMBER,
Employee_Id NUMBER,
Leave_date DATE,
CONSTRAINT Leave_PK PRIMARY KEY (Leave_Id),
FOREIGN KEY (Employee_Id)
    REFERENCES Employee(Employee_Id)
);

--------------------------------------------------
```

EmployeeAttendance Bridge

```
--------------------------------------------------


CREATE TABLE Employee_Attendance(
Employee_Id NUMBER,
Attendance_Id  NUMBER,
CONSTRAINT DEPARTMENTPROJECT_PK  PRIMARY  KEY (Employee_Id,Attendance_Id),
FOREIGN KEY (Employee_Id)
    REFERENCES Employee(Employee_Id),
FOREIGN KEY (Attendance_Id)
    REFERENCES Attendance(Attendance_Id)
);

--------------------------------------------------
```

Attendance

```
--------------------------------------------------
```

```
CREATE TABLE Attendance(
Attendance_Id  NUMBER,
Hours_Worked  NUMBER,
CONSTRAINT Attendance_PK  PRIMARY  KEY (Attendance_Id)
);

---------------------------------------------------
WorkLocation
---------------------------------------------------
CREATE TABLE Work_Location(
Location_Id NUMBER,
Location VARCHAR2(25),
Number_Of_Employees  NUMBER,
City VARCHAR2(25),
State VARCHAR2(25),
CONSTRAINT Loc_PK PRIMARY KEY (Location_Id)
);
```

Insert Statements

```
INSERT INTO Employee VALUES (101,'Ojas','Phansekar',to_date('14-APR-16',  'dd-MON-yyyy'),'New York City','New York',1);

INSERT INTO Employee VALUES (102,'Vrushali','Patil',to_date('21-JUN-18',  'dd-MON-yyyy'),'Boston','Massachusetts',2);

INSERT INTO Employee VALUES (103,'Pratik','Parija',to_date('13-SEP-19',  'dd-MON-yyyy'),'Chicago','Illinois',3);

INSERT INTO Employee VALUES (104,'Chetan','Mistry',to_date('12-APR-11',  'dd-MON-yyyy'),'Miami','Florida',4);

INSERT INTO Employee VALUES (105,'Anugraha','Varkey',to_date('16-AUG-17',  'dd-MON-yyyy'),'Atlanta','Georgia',5);
INSERT INTO Employee VALUES (106,'Rasagnya','Reddy',to_date('25-JUL-18', 'dd-MON-yyyy'),'San
Mateo','California',6);

INSERT INTO Employee VALUES (107,'Aishwarya','Boralkar',to_date('18-DEC-10', 'dd-MON-yyyy'),'San
Francisco','California',7);

INSERT INTO Employee VALUES (108,'Shantanu','Savant',to_date('27-NOV-15', 'dd-MON-
yyyy'),'Seattle','Washington',8);

INSERT INTO Employee VALUES (109,'Kalpita','Malvankar',to_date('24-APR-16', 'dd-MON-
yyyy'),'Boston','Massachusetts',8);

INSERT INTO Employee VALUES (110,'Saylee','Bhagat',to_date('21-MAY-14', 'dd-MON-yyyy'),'San
Francisco','California',7);



INSERT INTO Department VALUES (1,'Human Resources');

INSERT INTO Department VALUES (2,'Software Development');

INSERT INTO Department VALUES (3,'Data Analysis');

INSERT INTO Department VALUES (4,'Data Science');

INSERT INTO Department VALUES (5,'Business Intelligence');

INSERT INTO Department VALUES (6,'Data Engineering');

INSERT INTO Department VALUES (7,'Manufacturing');

INSERT INTO Department VALUES (8,'Quality Control');


INSERT INTO Project VALUES (21,'Dev','Whatever');

INSERT INTO Project VALUES (22,'Prod','do something');

INSERT INTO Project VALUES (23,'Test','focus');

INSERT INTO Project VALUES (24,'Nothing','do nothing');

INSERT INTO Project VALUES (25,'Research','focus on everything');

INSERT INTO Project VALUES (26,'Next Steps','find some way out');


INSERT INTO AccountDetails VALUES (40,'Santander','S12344',101);

INSERT INTO AccountDetails VALUES (41,'Santander','S12345',102);
INSERT INTO AccountDetails  VALUES (42,'Santander','S12346',103);
INSERT INTO AccountDetails  VALUES (43,'Santander','S12347',104);
INSERT INTO AccountDetails  VALUES (44,'Chase','C12344',105);
INSERT INTO AccountDetails  VALUES (45,'Chase','C12345',106);
INSERT INTO AccountDetails  VALUES (46,'Chase','C12347',107);
INSERT INTO AccountDetails  VALUES (47,'Chase','C12334',108);
INSERT INTO AccountDetails  VALUES (48,'BOFA','C12378',109);
INSERT INTO AccountDetails  VALUES (49,'BOFA','C12390',110);

INSERT INTO Education  VALUES (10,101,'MS',2017);
INSERT INTO Education  VALUES (11,102,'MS',2019);
INSERT INTO Education  VALUES (12,104,'MS',2011);
INSERT INTO Education  VALUES (13,108,'MS',2015);
INSERT INTO Education  VALUES (14,109,'Bachelor',2013);
INSERT INTO Education  VALUES (15,107,'Bachelor',2008);
INSERT INTO Education  VALUES (16,106,'Bachelor',2007);

INSERT INTO Leave VALUES (51,104,to_date('1-DEC-19',  'dd-MON-yyyy'));
INSERT INTO Leave VALUES (52,108,to_date('2-DEC-19',  'dd-MON-yyyy'));
INSERT INTO Leave VALUES (53,109,to_date('3-DEC-19',  'dd-MON-yyyy'));
INSERT INTO Leave VALUES (54,107,to_date('4-DEC-19',  'dd-MON-yyyy'));
INSERT INTO Leave VALUES (55,106,to_date('5-DEC-19',  'dd-MON-yyyy'));
INSERT INTO Leave VALUES (56,104,to_date('6-DEC-19',  'dd-MON-yyyy'));
INSERT INTO Leave VALUES (57,108,to_date('7-DEC-19',  'dd-MON-yyyy'));
INSERT INTO Leave VALUES (58,109,to_date('7-DEC-19',  'dd-MON-yyyy'));
INSERT INTO Leave VALUES (59,107,to_date('8-DEC-19',  'dd-MON-yyyy'));
INSERT INTO Leave VALUES (60,106,to_date('9-DEC-19',  'dd-MON-yyyy'));
INSERT INTO Attendance  VALUES (90,10);
INSERT INTO Attendance  VALUES (91,20);
INSERT INTO Attendance  VALUES (92,30);
INSERT INTO Attendance  VALUES (93,40);
INSERT INTO Attendance  VALUES (94,45);
INSERT INTO Attendance  VALUES (95,56);
INSERT INTO Attendance  VALUES (96,58);

INSERT INTO Work_Location VALUES (71,'North',4,'New York City','New York',101);
INSERT INTO Work_Location VALUES (72,'North',4,'Boston','Massachusetts',102);
INSERT INTO Work_Location VALUES (73,'North',4,'Chicago','Illinois',103);
INSERT INTO Work_Location VALUES (74,'North',89,'Miami','Florida',104);
INSERT INTO Work_Location VALUES (75,'South',90,'Atlanta','Georgia',105);
INSERT INTO Work_Location VALUES (76,'South',100,'San  Mateo','California',106);
INSERT INTO Work_Location VALUES (77,'South',4,'San  Francisco','California',107);
INSERT INTO Work_Location VALUES (78,'South',2,'Seattle','Washington',108);
INSERT INTO Work_Location VALUES (79,'South',25,'Alpharetta','Georgia',109);
INSERT INTO Work_Location VALUES (80,'South',20,'Keene','New  Hampshire',110);
INSERT INTO Work_Location VALUES (81,'South',22,'Hampton','New  Hampshire',109);

INSERT INTO Employee_Attendance  VALUES (101,90);
INSERT INTO Employee_Attendance  VALUES (102,91);
INSERT INTO Employee_Attendance  VALUES (103,92);
INSERT INTO Employee_Attendance  VALUES (104,93);
INSERT INTO Employee_Attendance  VALUES (105,94);
INSERT INTO Employee_Attendance  VALUES (106,95);
INSERT INTO Employee_Attendance  VALUES (107,96);
INSERT INTO Employee_Attendance  VALUES (108,91);
INSERT INTO Employee_Attendance  VALUES (109,92);
INSERT INTO Employee_Attendance  VALUES (110,93);

INSERT INTO DepartmentProject  VALUES (1,21);
INSERT INTO DepartmentProject  VALUES (2,22);
INSERT INTO DepartmentProject  VALUES (3,23);
INSERT INTO DepartmentProject  VALUES (4,24);
INSERT INTO DepartmentProject  VALUES (5,25);
INSERT INTO DepartmentProject  VALUES (6,26);
INSERT INTO DepartmentProject  VALUES (7,21);
INSERT INTO DepartmentProject  VALUES (8,24);

INSERT INTO Salary  VALUES (1,57600,30,200,1000,40);
INSERT INTO Salary  VALUES (2,76800,40,300,1300,41);
INSERT INTO Salary  VALUES (3,96000,50,400,1500,42);
INSERT INTO Salary  VALUES (4,115200,60,500,1700,43);
INSERT INTO Salary  VALUES (5,57600,30,200,1000,44);
INSERT INTO Salary  VALUES (6,76800,40,300,1300,45);
INSERT INTO Salary  VALUES (7,96000,50,400,1500,46);
INSERT INTO Salary  VALUES (8,115200,60,500,1700,47);
INSERT INTO Salary  VALUES (9,57600,30,200,1000,48);
INSERT INTO Salary  VALUES (10,76800,40,300,1300,49);
```

Inline View

```
select Department_Name,  count(*),
to_char((count(*)/No_of_Employees.cnt)*100,  '90.99') Percentages
from Department,Employee,  ( select count(*) cnt from Employee ) No_of_Employees
where Department.Department_Id  = Employee.Department_Id
group by Department_Name,  No_of_Employees.cnt
```

```
/
```

Materialized View

Number of Employees with different degrees

```
---------------------------------------------
create materialized view Education_View
        build immediate
        refresh on commit
        as
        select Degree, count(Degree)
        from Education
        group by Degree;
```

Procedure

Locations with less number of employees

```
CREATE OR REPLACE PROCEDURE Unimportant_Locations(l_NOFEmployees IN Number)
IS
 l_wl NUMBER;
 l_emp NUMBER;

BEGIN
 SELECT COUNT(*) INTO l_wl
 FROM Work_Location
 WHERE Number_Of_Employees LIKE l_NOFEmployees;


 select count(*)
 into l_emp
 from Employee e
 inner join Work_Location w
 on e.Employee_Id = w.Employee_Id
 where w.Number_Of_Employees  LIKE l_NOFEmployees;

 IF l_wl < 5 THEN
   DELETE FROM Work_Location
   WHERE Number_Of_Employees  = l_NOFEmployees;
          END IF;

 EXCEPTION WHEN no_data_found THEN
 DBMS_OUTPUT.PUT_LINE('No Such Data Available');
END;
```

Explicit Cursor

```
declare
              cursor salaries(p_hourly  in number)
              is select *
              from Salary
              where Hourly_Pay=p_hourly;

              l_sal Salary%rowtype;
              begin
               dbms_output.put_line('  Extracting hourly pay');
               open salaries(30);
               loop
                fetch salaries into l_sal;
                         exit when salaries%notfound;
                         dbms_output.put('For  Account ' || l_sal.Account_Id  || ' Hourly Pay is ');
       dbms_output.put_line(l_sal.hourly_pay);
```

```
                end loop;
                close salaries;
      end;
     /
```

Pre-Defined Exception

```
declare
        l_attendance  Attendance%rowtype;
        New_Exception exception;
begin
        l_attendance.Attendance_Id  := 90;
        l_attendance.Hours_Worked  := 'AS';
        insert into Attendance (Attendance_Id,Hours_Worked)
        values ( l_attendance.Attendance_Id,  l_attendance.Hours_Worked  );
exception
        when VALUE_ERROR then
                dbms_output.put_line('We  encountered  the VALUE_ERROR exception');
end;
/
```

Explicit Cursor and Pre-Defined Cursor Together

```
declare
         cursor salaries(p_hourly  in number)
         is select *
         from Salary
         where Hourly_Pay=p_hourly;

         l_sal Salary%rowtype;
         begin
```

```
            dbms_output.put_line('Getting  hourly pay');
            open salaries(30);
            loop
             fetch salaries into l_sal;
                        exit when salaries%notfound;
                        dbms_output.put('For  Account ' || l_sal.Account_Id || ' Hourly Pay is ');
        dbms_output.put_line(l_sal.hourly_pay);
                        end loop;
                        open salaries(30);
                        exception
                        when CURSOR_ALREADY_OPEN then
                        dbms_output.put_line('No  Need to open cursor again');
                        close salaries;
       end;
      /
```

External Table

```
create table Salary_External  (
  Salary_Id NUMBER,
  Gross_Salary NUMBER,
  Hourly_Pay NUMBER,
  State_Tax NUMBER,
  Federal_Tax NUMBER,
  Account_Id NUMBER
)
organization external (
            type oracle_loader
            default directory ext_Salaries
            access parameters (
		fields terminated by ',' )
	location ('Salary.csv')
)
reject limit unlimited
/
```

Ref Cursor

```
declare
type emp_dept_rec is record(
	Employee_Id number,
	First_Name varchar2(66),
	Department_Name varchar2(37)
	);

	type emp_dept_refcur_type is ref cursor
		return emp_dept_rec;

	employee_refcur emp_dept_refcur_type;

	emp_dept emp_dept_rec;
begin
	open employee_refcur for
		select e.Employee_Id,
			e.First_Name || ' ' || e.Last_Name "Employee Name",
			d.Department_Name
		from Employee e, Department d
		where e.Department_Id = d.Department_Id
		and rownum < 5
		order by e.Employee_Id;
```

```
        fetch employee_refcur into emp_dept;

        while employee_refcur%FOUND loop

                dbms_output.put(emp_dept.First_Name || '''s department is ');

                dbms_output.put_line(emp_dept.Department_Name);

                fetch employee_refcur into emp_dept;

        end loop;

end;

/
```

Transaction

```
INSERT INTO Employee VALUES (111,'Priyanka','Jonas',to_date('14-NOV-16', 'dd-MON-yyyy'),'New York
City','New York',1);


commit;


INSERT INTO Employee VALUES (112,'John','Vincent',to_date('21-JUN-18', 'dd-MON-
yyyy'),'Boston','Massachusetts',2);


SAVEPOINT A1;


INSERT INTO Employee VALUES (113,'Pratik','Panhale',to_date('13-SEP-19', 'dd-MON-
yyyy'),'Chicago','Illinois',3);


SAVEPOINT A2;


ROLLBACK A1;
```

Relational View

```
create or replace view salary_range_calculator
as
select e.First_Name, s.Hourly_Pay
        from Employee e
        inner join AccountDetails a
        on e.Employee_Id = a.Employee_Id
        inner join Salary s
        on a.Account_Id = s.Account_Id
        where s.Hourly_Pay = 30;
```