Workforce Payroll Management System

Summary:

1. **Employee Information**
   Maintaining accurate employee data is essential for communication, event invitations, and collecting feedback.

2. **Salary Records**
   Tracking salary details helps HR and management monitor payroll and enables the company to assess employee costs.

3. **Work Location**
   Recording work locations supports regional growth planning, targeted hiring, and market expansion strategies.

4. **Projects**
   Maintaining project records, including employee assignments and related salaries, ensures smooth project management and cost analysis.

PL/SQL Features Used in the Project:

1. Designed Explicit Cursors to display employees' hourly pay linked to their respective accounts, and implemented a Ref Cursor to retrieve employees belonging to a specific department.
2. Created a Container Database (CDB) and Pluggable Database (PDB) with users to efficiently manage data based on specific areas of interest.
3. Implemented the pre-defined exception CURSOR_ALREADY_OPEN to demonstrate exception handling by showing the error triggered when opening an already open cursor.
4. Developed Relational Views, Inline Views, and Materialized Views to meet diverse business requirements.
5. Created an Index on the AccountDetails table to enhance query performance.
6. Designed an Entity-Relationship (E-R) Diagram to visualize and understand entity relationships within the payroll management system for any organization.

List of Entities:

Employee

The Employee table stores all personal details of each employee, covering comprehensive information related to that individual.

Salary

The Salary table records both current and historical salary details of employees. It helps managers and HR track salary grade changes and promotion dates.

Department

The Department table maintains data about all departments within the company that an employee can belong to.

Account Details

The Account Details table stores information about the bank accounts linked by employees for salary credit purposes.

Attendance

The Attendance table captures data on employee attendance, including the total number of hours worked each week.

Project

The Project table holds information on all projects the company is currently handling as well as upcoming projects.

Education

The Education table tracks each employee's academic qualifications, including all degrees earned.

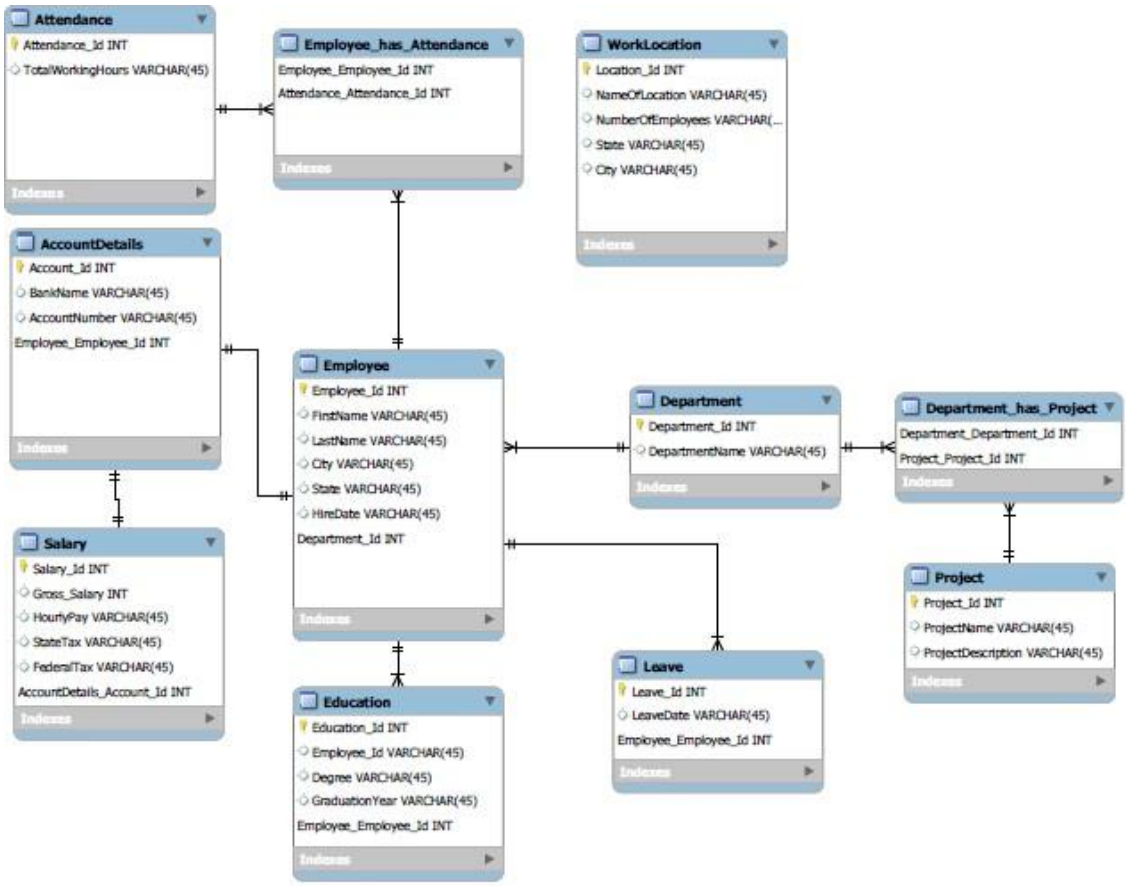
Work Location

The Work Location table stores details about office locations, including city, state, and the number of employees assigned to each location.

Leave

The Leave table records the number of leaves an employee has taken or applied for within a month or a 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
  2    cursor salaries(p_hourly in number)
  3    is select *
  4    from Salary
  5    where Hourly_Pay=p_hourly;
  6
  7    l_sal Salary%rowtype;
  8    begin
  9     dbms_output.put_line(' Extracting hourly pay');
 10     open salaries(30);
 11     loop
 12      fetch salaries into l_sal;
 13  exit when salaries%notfound;
 14  dbms_output.put('For Account ' || l_sal.Account_Id || ' Hourly Pay is ');
 15          dbms_output.put_line(l_sal.hourly_pay);
 16  end loop;
 17  close salaries;
 18         end;
 19        /
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
  2  on AccountDetails(Bank_Name);

Index created.
```

7. Relational Views

```
SQL> create or replace view salary_range_calculator
  2  as
  3  select e.First_Name, s.Hourly_Pay
  4  from Employee e
  5  inner join AccountDetails a
  6  on e.Employee_Id = a.Employee_Id
  7  inner join Salary s
  8  on a.Account_Id = s.Account_Id
  9  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
  2  as 'C:\Users\phansekar.o\Desktop\Salary.csv'
  3  /

Directory created.

SQL> grant all on directory ext_Salaries to HRADMIN
  2  /

Grant succeeded.

SQL> create table Salary_External (
  2    Salary_Id NUMBER,
  3    Gross_Salary NUMBER,
  4    Hourly_Pay NUMBER,
  5    State_Tax NUMBER,
  6    Federal_Tax NUMBER,
  7    Account_Id NUMBER
  8  )
  9  organization external (
 10  type oracle_loader
 11  default directory ext_Salaries
 12  access parameters (
 13  fields terminated by ',' )
 14  location ('Salary.csv')
 15  )
 16  reject limit unlimited
 17  /

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
  2  type emp_dept_rec is record(
  3  Employee_Id number,
  4  First_Name varchar2(66),
  5  Department_Name varchar2(37)
  6  );
  7
  8  type emp_dept_refcur_type is ref cursor
  9  return emp_dept_rec;
 10
 11  employee_refcur emp_dept_refcur_type;
 12
 13  emp_dept emp_dept_rec;
 14  begin
 15  open employee_refcur for
 16  select e.Employee_Id,
 17     e.First_Name || ' ' || e.Last_Name "Employee Name",
 18     d.Department_Name
 19  from Employee e, Department d
 20  where e.Department_Id = d.Department_Id
 21  and rownum < 5
 22  order by e.Employee_Id;
 23
 24  fetch employee_refcur into emp_dept;
 25  while employee_refcur%FOUND loop
 26  dbms_output.put(emp_dept.First_Name || '''s department is ');
 27  dbms_output.put_line(emp_dept.Department_Name);
 28  fetch employee_refcur into emp_dept;
 29  end loop;
 30  end;
 31  /
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
  2  l_attendance Attendance%rowtype;
  3  begin
  4  l_attendance.Attendance_Id := 90;
  5  l_attendance.Hours_Worked := 'AS';
  6  insert into Attendance (Attendance_Id,Hours_Worked)
  7  values ( l_attendance.Attendance_Id, l_attendance.Hours_Worked );
  8  exception
  9  when VALUE_ERROR then
 10  dbms_output.put_line('We encountered the VALUE_ERROR exception');
 11  end;
 12  /
We encountered the VALUE_ERROR exception

PL/SQL procedure successfully completed.
```

12. Procedure

```
SQL> CREATE OR REPLACE PROCEDURE Unimportant_Locations(l_NOFEmployees IN Number)
  2  IS
  3    l_wl NUMBER;
  4    l_emp NUMBER;
  5
  6  BEGIN
  7    SELECT COUNT(*) INTO l_wl
  8    FROM Work_Location
  9    WHERE Number_Of_Employees LIKE l_NOFEmployees;
 10
 11
 12    select count(*)
 13    into l_emp
 14    from Employee e
 15    inner join Work_Location w
 16    on e.Employee_Id = w.Employee_Id
 17    where w.Number_Of_Employees LIKE l_NOFEmployees;
 18
 19    IF l_wl < 5 THEN
 20      DELETE FROM Work_Location
 21      WHERE Number_Of_Employees = l_NOFEmployees;
 22  END IF;
 23
 24    EXCEPTION WHEN no_data_found THEN
 25    DBMS_OUTPUT.PUT_LINE('No Such Data Available');
 26  END;
 27  /

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
  2    cursor salaries(p_hourly in number)
  3    is select *
  4    from Salary
  5    where Hourly_Pay=p_hourly;
  6
  7    l_sal Salary%rowtype;
  8    begin
  9     dbms_output.put_line('Getting hourly pay');
 10     open salaries(30);
 11     loop
 12      fetch salaries into l_sal;
 13  exit when salaries%notfound;
 14  dbms_output.put('For Account ' || l_sal.Account_Id || ' Hourly Pay is ');
 15          dbms_output.put_line(l_sal.hourly_pay);
 16  end loop;
 17  open salaries(30);
 18  exception
 19  when CURSOR_ALREADY_OPEN then
 20  dbms_output.put_line('No Need to open cursor again');
 21  close salaries;
 22         end;
 23        /
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.
```