Employee Performance and HR Analytics Report

Project Report: Employee Performance and HR Analytics

Name: Pabitra Mandal

---

1. Project Overview

Objective:

The primary objective of this project is to analyze employee-related data to gain insights into employee performance, salary trends, department-wise analysis, and identify factors influencing employee retention and performance.

Tools Used:

- Microsoft SQL Server
- SQL Scripts

---

2. Database Design

Database Name:

HRAnalytics

Tables Created:

1. Departments
2. Employees
3. PerformanceReviews
4. SalaryHistory

---

3. Key Insights

1. Average Salary by Department

- Engineering department has the highest average salary.

- HR department has the lowest average salary.

2. Employees with Low Performance Score

- Identified employees with performance scores below average (<3).
- Necessary actions or training plans can be suggested for improvement.

3. Salary Change History

- Tracked salary growth over time for specific employees.
- Used for compensation analysis and performance appraisal.

4. Top Performers

- Identified employees scoring 4 or 5 in performance reviews within the last year.
- These employees can be considered for rewards or promotions.

5. Average Employee Tenure

- Calculated the average employee tenure in years.
- Useful for understanding employee retention.

6. Employees Earning Above Average Salary

- Listed employees whose salaries exceed the company's average salary.

7. Manager-wise Employee Count

- Identified the number of employees managed by each manager.
- Helpful for workload distribution and managerial performance evaluation.

---

4. Conclusion

This project provides a comprehensive analysis of employee performance and HR-related metrics using SQL. The insights derived from the analysis will help the organization make data-driven decisions regarding employee management, compensation, and performance improvement.

Employee Performance and HR Analytics Report

---

5. **Appendix: Full SQL Scripts**

- ****Create Database****

```
CREATE DATABASE HRAnalytics;
USE HRAnalytics;
```

- ****Create Tables****

1. **Create Departments Table**

```
CREATE TABLE Departments (
  DepartmentID INT PRIMARY KEY,
  DepartmentName VARCHAR(100)
);
```

2. **Create Employees Table**

```
CREATE TABLE Employees (
  EmployeeID INT PRIMARY KEY,
  Name VARCHAR(100),
  DepartmentID INT,
  HireDate DATE,
  Salary DECIMAL(10, 2),
  ManagerID INT,
  FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID)
);
```

3. **Create PerformanceReviews Table**

```
CREATE TABLE PerformanceReviews (
  ReviewID INT PRIMARY KEY,
  EmployeeID INT,
  ReviewDate DATE,
  Score INT,            -- Rating from 1 to 5
  Feedback TEXT,
  FOREIGN KEY (EmployeeID) REFERENCES Employees(EmployeeID)
);
```

4. **Create SalaryHistory Table**

```
CREATE TABLE SalaryHistory (
  SalaryHistoryID INT PRIMARY KEY,
  EmployeeID INT,
  Salary DECIMAL(10, 2),
  ChangeDate DATE,
  FOREIGN KEY (EmployeeID) REFERENCES Employees(EmployeeID)
);
```

- ****Inserting Sample Data****

Now, let's insert some sample data to work with.

1. **Insert Departments :**

```
INSERT INTO Departments (DepartmentID, DepartmentName) VALUES
```

(1, 'Sales'),

(2, 'Engineering'),

(3, 'Marketing'),

(4, 'HR');

2. **Insert Employees :**

INSERT INTO Employees (EmployeeID, Name, DepartmentID, HireDate, Salary, ManagerID) VALUES

(1, 'Alice Johnson', 1, '2020-01-15', 60000, NULL),

(2, 'Bob Smith', 2, '2018-05-22', 75000, 1),

(3, 'Charlie Brown', 1, '2021-06-10', 55000, 1),

(4, 'David Clark', 3, '2019-08-03', 65000, 2),

(5, 'Eve Davis', 2, '2017-03-12', 85000, 2),

(6, 'Frank Wright', 4, '2022-11-25', 50000, NULL);

3. **Insert Performance Reviews :**

INSERT INTO PerformanceReviews (ReviewID, EmployeeID, ReviewDate, Score, Feedback) VALUES

(1, 1, '2023-12-01', 4, 'Excellent performance, consistent results.'),

(2, 2, '2023-11-15', 5, 'Outstanding leadership and project success.'),

(3, 3, '2023-06-30', 3, 'Needs improvement in project management.'),

(4, 4, '2023-09-01', 4, 'Good performance, meets expectations.'),

(5, 5, '2023-07-20', 5, 'Exceptional results, great strategic vision.'),

(6, 6, '2023-12-05', 2, 'Performance below expectations, requires improvement.');

4. **Insert Salary History :**

INSERT INTO SalaryHistory (SalaryHistoryID, EmployeeID, Salary, ChangeDate) VALUES

(1, 1, 60000, '2020-01-15'),

(2, 2, 70000, '2018-05-22'),

(3, 3, 55000, '2021-06-10'),

(4, 4, 65000, '2019-08-03'),

(5, 5, 85000, '2017-03-12'),

(6, 6, 50000, '2022-11-25');

- ****SQL Queries for Data Analysis****

- ****Table select queries****

SELECT * FROM Departments;

SELECT * FROM Employees;

SELECT * FROM PerformanceReviews;

SELECT * FROM SalaryHistory;

1. We want to see the average salary of employees in each department.

```
SELECT
    d.DepartmentName,
    AVG(e.Salary) AS AverageSalary
FROM
    Employees e
JOIN
```

```
    Departments d
ON
    e.DepartmentID = d.DepartmentID
GROUP BY
    d.DepartmentName;
```

2. Identify employees who scored below average in performance reviews.

```
SELECT
    e.Name,
    pr.Score,
    pr.Feedback
FROM
    Employees e
JOIN
    PerformanceReviews pr
ON
    e.EmployeeID = pr.EmployeeID
WHERE
    pr.Score < 3
ORDER BY
    pr.Score ASC;
```

3. Display the salary change history for a specific employee.

```
SELECT
    e.Name,
    sh.Salary,
    sh.ChangeDate
FROM
    SalaryHistory sh
JOIN
    Employees e
ON
    sh.EmployeeID = e.EmployeeID
WHERE
    e.Name = 'Bob Smith'
ORDER BY
    sh.ChangeDate DESC;
```

4. Find the top performers (score of 4 or 5) in the last year.

```
SELECT
    e.Name,
    pr.Score,
    pr.Feedback
FROM
    Employees e
JOIN
```

```
    PerformanceReviews pr
ON
    e.EmployeeID = pr.EmployeeID
WHERE
    pr.ReviewDate >= '2023-01-01' AND pr.Score >= 4
ORDER BY
    pr.Score DESC;
```

5. Calculate the average tenure (years) of employees in the company.

```
SELECT
    AVG(DATEDIFF(DAY, HireDate, GETDATE()) / 365.0) AS AverageTenure
FROM
    Employees e;
```

6. List employees who earn more than the average salary in the company.

```
SELECT
    e.Name,
    e.Salary,
    d.DepartmentName
FROM
    Employees e
JOIN
```

```
    Departments d
ON
    e.DepartmentID = d.DepartmentID
WHERE
    e.Salary > (SELECT AVG(Salary) FROM Employees)
ORDER BY
    e.Salary DESC;
```

7. Count the number of employees managed by each manager.

```
SELECT
    e.Name AS Manager,
    COUNT(emp.EmployeeID) AS EmployeesManaged
FROM
    Employees e
LEFT JOIN
    Employees emp
ON
    e.EmployeeID = emp.ManagerID
GROUP BY
    e.Name
ORDER BY
    EmployeesManaged DESC;
```