# ScienceQTech Employee Performance Mapping

ScienceQtech is a startup that works in the Data Science field. ScienceQtech has worked on fraud detection, market basket, self-driving cars, supply chain, algorithmic early detection of lung cancer, customer sentiment, and the drug discovery field. With the annual appraisal cycle around the corner, the HR department has asked you (Junior Database Administrator) to generate reports on employee details, their performance, and on the project that the employees have undertaken, to analyze the employee database and extract specific data based on different requirements.

## Objective:

To facilitate a better understanding, managers have provided ratings for each employee which will help the HR department to finalize the employee performance mapping. As a DBA, you should find the maximum salary of the employees and ensure that all jobs are meeting the organization's profile standard. You also need to calculate bonuses to find extra cost for expenses. This will raise the overall performance of the organization by ensuring that all required employees receive training.

## Task 1:

| **Create Database and import tables** | |
|---|---|
| Create a database named employee, then import **data_science_team.csv proj_table.csv** and **emp_record_table.csv** into the **employee** database from the given resources | |
| | |
| **SQL Query** | **Comment** |
| CREATE DATABASE employee; | Create a database |
| USE DATABASE employee; | Use the created database |
| Import tables "**emp_record_table.csv**", "**proj_table.csv**", "**data_science_team.csv**" using the Table Data Import Wizard. | |
| | |
| **Outcome** | |
| Database is created and Tables are imported using "Table Data Import Wizard" | |

## Task 2:

| | |
|---|---|
| **ER Diagram** | |
| Create an ER diagram for the given **employee** database | |
| | |
| **SQL Query** | **Comment** |
| | |
| **Outcome / Conclusion** | |
| ER Diagram is created using www.gliffy.com | |

## Task 3:

| | |
|---|---|
| **SELECT Statement** | |
| Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER, and DEPARTMENT from the employee record table, and make a list of employees and details of their department | |
| | |
| **SQL Query** | **Comment** |
| SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT<br>FROM emp_record_table; | Fetch the mentioned columns from the table emp_record_table and display |
| | |
| **Outcome / Conclusion** | |
| List with the required columns. | |

## Task 4:

| **WHERE Clause** | |
|---|---|
| Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPARTMENT, and EMP_RATING if the EMP_RATING is:<br>• less than two<br>• greater than four<br>• between two and four | |
| | |
| **SQL Query** | **Comment** |
| SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING<br>FROM emp_record_table<br>WHERE EMP_RATING < 2; | Fetch the required columns from emp_record_table for employees having rating less than 2 |
| SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING<br>FROM emp_record_table<br>WHERE EMP_RATING > 4; | Fetch the required columns from emp_record_table for employees having rating greater than 4 |
| SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING<br>FROM emp_record_table<br>WHERE EMP_RATING BETWEEN 2 AND 4; | Fetch the required columns from emp_record_table for employees having rating between 2 and 4 |
| | |
| **Outcome / Conclusion** | |
| Lists containing the required fields | |

## Task 5:

| **CONCAT** | |
|---|---|
| Write a query to concatenate the FIRST_NAME and the LAST_NAME of employees in the Finance department from the employee table and then give the resultant column alias as NAME | |
| | |
| **SQL Query** | **Comment** |
| SELECT CONCAT(FIRST_NAME, ‘ ‘, LAST_NAME) AS NAME<br>FROM emp_record_table<br>WHERE DEPT = “FINANCE”; | Concatenate the First_name and last_name with a space in between for all employees in Finance Department |
| | |
| **Outcome / Conclusion** | |
| Column with the name “NAME”, containing the full name of all employees of Finance Department | |

## Task 6:

| COUNT, SELF JOIN and GROUP BY | |
|---|---|
| Write a query to list only those employees who have someone reporting to them. Also, show the number of reporters (including the President) | |
| | |
| **SQL Query** | **Comment** |
| SELECT E.EMP_ID, E.FIRST_NAME, E.LAST_NAME, E.ROLE, COUNT(R.EMP_ID) AS NUMBER_OF_REPORTERS<br>FROM emp_record_table AS E, emp_record_table AS R<br>WHERE E.MANAGER_ID = R.EMP_ID<br>GROUP BY E.EMP_ID; | Fetch the required columns from emp_record_table where the employee is a manager and create a new column containing the count of employees under each manager. |
| | |
| **Outcome / Conclusion** | |
| List of all managers along with number of employees reporting to them. | |

## Task 7:

| **UNION** | |
|---|---|
| Write a query to list down all the employees from the healthcare and finance departments using union. Take data from the employee record table | |
| | |
| **SQL Query** | **Comment** |
| SELECT * FROM emp_record_table WHERE DEPT = “HEALTHCARE”<br>UNION<br>SELECT * FROM emp_record_table WHERE DEPT = “FINANCE”; | Fetch all records from emp_record_table for employees in Healthcare and Finance department |
| | |
| **Outcome / Conclusion** | |
| Table containing details of employees of Healthcare and Finance Department | |

## Task 8:

| **PARTITION** | |
|---|---|
| Write a query to list down employee details such as EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPARTMENT, and EMP_RATING grouped by dept. Also include the respective employee rating along with the max emp rating for the department | |
| | |
| **SQL Query** | **Comment** |
| SELECT EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, EMP_RATING, MAX(EMP_RATING) OVER (PARTITION BY ROLE) AS MAX_EMP_RATING FROM emp_record_table; | Fetch the required columns and calculate maximum EMP_RATING for each Role from emp_record_table |
| | |
| **Outcome / Conclusion** | |
| Table containing details of employees along with maimum rating in their respective departments | |

## Task 9:

| **MIN() & MAX()** | |
|---|---|
| Write a query to calculate the minimum and the maximum salary of the employees in each role. Take data from the employee record table | |
| | |
| **SQL Query** | **Comment** |
| SELECT ROLE, MIN(SALARY), MAX(SALARY)<br>FROM emp_record_table<br>GROUP BY ROLE; | Group the data by roles and fetch minimum and maximum salary for each role |
| | |
| **Outcome / Conclusion** | |
| Table containing each role and it's minimum and maximum salary. | |

## Task 10:

| **RANK()** | |
|---|---|
| Write a query to assign ranks to each employee based on their experience. Take data from the employee record table | |
| | |
| **SQL Query** | **Comment** |
| SELECT EMP_ID, DIRST_NAME, LAST_NAME, EXP, RANK()<br>OVER (ORDER BY EXP) AS EXP_RANK<br>FROM emp_record_table; | Fetch the required columns and add a column EXP_RANK containing the rank of employees according to their experience. |
| | |
| **Outcome / Conclusion** | |
| Table containing required columns and rank of each employee | |

## Task 11:

| **CREATE VIEW** | |
|---|---|
| Write a query to create a view that displays employees in various countries whose salary is more than six thousand. Take data from the employee record table | |
| | |
| **SQL Query** | **Comment** |
| CREATE VIEW emp_salary AS<br>SELECT EMP_ID, FIRST_NAME, LAST_NAME, COUNTRY, SALARY<br>FROM emp_record_table<br>WHERE SALARY > 6000; | Create a View containing details of employees having salary more than 6000. |
| SELECT * FROM emp_salary | Display all the details from the view created above |
| | |
| **Outcome / Conclusion** | |
| Table containing required columns for employees having salary more than 6000. The view is save for future use | |

## Task 12:

| | |
|---|---|
| **NESTED SELECT** | |
| Write a nested query to find employees with experience of more than ten years. Take data from the employee record table | |
| | |
| **SQL Query** | **Comment** |
| SELECT * FROM emp_record_table WHERE EMP_ID IN ( SELECT EMP_ID FROM emp_record_table WHERE EXP > 10); | Fetch all details from emp_record_table where the EMP_ID is present in the list of EMP_IDs who have experience more than 10 years |
| | |
| **Outcome / Conclusion** | |
| Table containing all details of employees having experience more than 10 years. | |

## Task 13:

| | |
|---|---|
| **STORED PROCEDURE** | |
| Write a query to create a stored procedure to retrieve the details of the employees whose experience is more than three years. Take data from the employee record table | |
| | |
| **SQL Query** | **Comment** |
| DELIMITER //<br>CREATE PROCEDURE emp_exp()<br>BEGIN<br>SELECT * FROM emp_record_table<br>WHERE EXP > 3;<br>END // | Initialise a delimiter //<br>Create a procedure that fetch all data of employees having experience more than 3 years |
| DELIMITER ; | Reset Delimiter to ; |
| | |
| **Outcome / Conclusion** | |
| Store Procedure is created | |

## Task 14:

| **STORED FUNCTION** | |
|---|---|
| Write a query using stored functions in the project table to check whether the job profile assigned to each employee in the data science team matches the organization's set standard | |
| **SQL Query** | **Comment** |
| DELIMITER && | Initialise a delimiter // |
| CREATE FUNCTION check_profile ( | Create a function that takes two inputs EXP and ROLE and returns a VARCHAR. |
|   EXP INT, | |
|   ROLE VARCHAR(25)) | |
| RETURNS VARCHAR(50) DETERMINISTIC | |
| BEGIN | |
|   DECLARE check_status VARCHAR(20); | Declare a VARCHAR "check_status" and set its value as "MATCH" or "NO MATCH" considering if the conditions are met. |
|   IF (EXP <= 2 AND ROLE = "JUNIOR DATA SCIENTIST") | |
|   THEN SET check_status = "MATCH"; | |
|   ELSEIF (EXP > 2 AND EXP <= 5 AND ROLE = "ASSOCIATE DATA SCIENTIST") | |
|   THEN SET check_status = "MATCH"; | |
|   ELSEIF (EXP > 5 AND EXP <= 10 AND ROLE = "SENIOR DATA SCIENTIST") | |
|   THEN SET check_status = "MATCH"; | |
|   ELSEIF (EXP > 10 AND EXP <= 12 AND ROLE = "LEAD DATA SCIENTIST") | |
|   THEN SET check_status = "MATCH"; | |
|   ELSEIF (EXP > 12 AND EXP <=16 AND ROLE = "MANAGER") | |
|   THEN SET check_status = "MATCH"; | |
|   ELSE | |
|   SET check_status = "NO MATCH"; | |
|   END IF; | |
| RETURN(check_status) | Return the variable check_status |
| END && | |
| DELIMITER ; | Reset Delimiter to ; |
| SELECT FIRST_NAME, LAST_NAME, ROLE, EXP, check_profile(EXP, ROLE) | Fetch the required columns and create a new column containing the data returned by above created function. |
| FROM emp_record_table | |
| **Outcome / Conclusion** | |
| Store Function is created | |

## Task 15:

| **INDEX** | |
|---|---|
| Create an index to improve the cost and performance of the query to find the employee whose FIRST_NAME is ‘Eric’ in the employee table after checking the execution plan | |
| | |
| **SQL Query** | **Comment** |
| CREATE INDEX ind_first_name<br>ON emp_record_table(FIRST_NAME); | Create an index for FIRST_NAME |
| SELECT * FROM emp_record_table<br>WHERE FIRST_NAME = “Eric”; | Fetch data for employees having first name as Eric |
| | |
| **Outcome / Conclusion** | |
| Index is created<br>Data of Eric is displayed | |

## Task 16:

| BONUS CALCULATION | |
|---|---|
| Write a query to calculate the bonus for all the employees, based on their ratings and salaries (Use the formula: 5% of salary * employee rating) | |
| | |
| **SQL Query** | **Comment** |
| SELECT EMP_ID, FIRST_NAME, LAST_NAME, SSALARY, EMP_RATING, (SALARY*0.05*EMP_RATING) AS BONUS FROM emp_record_table; | Fetch details of employees and calculate their Bonus and display in a new column |
| | |
| **Outcome / Conclusion** | |
| Table containg Employee bonus is displayed | |

## Task 17:

| **AVG()** | |
|---|---|
| Write a query to calculate the average salary distribution based on the continent and country. Take data from the employee record table | |
| | |
| **SQL Query** | **Comment** |
| SELECT CONTINENT, AVG(SALARY) AS AVERAGE_SALARY_BY_CONTINENT<br>FROM emp_record_table<br>GROUP BY CONTINENT; | Fetch the continents and average salary in each continent |
| SELECT COUNTRY, AVG(SALARY) AS AVERAGE_SALARY_BY_COUNTRY<br>FROM emp_record_table<br>GROUP BY COUNTRY; | Fetch the country and average salary in each country |
| | |
| **Outcome / Conclusion** | |
| Tables containing continents and average salary in each continent.<br>Tables containing country and average salary in each country. | |