

## **PROJECT WRITE-UP:**

### **SCIENCEQTECH EMPLOYEE PERFORMANCE MAPPING.**

#### **Objectives:**

The main goal of this project is to assist the HR department in finalizing employee performance mapping for the annual appraisal cycle. This involves analyzing employee details, their performance ratings, and the projects they have worked on.

Specific tasks include: Identifying the maximum salary of employees. Ensuring all job roles meet the organization's profile standards. Calculating bonuses to estimate extra costs. Providing data to ensure employees receive necessary training. Tools and Techniques Database Management: Creation and manipulation of databases. SQL Queries: To extract and manipulate data. ER Diagrams: To visualize relationships between different data tables. Stored Procedures and Functions: For efficient data processing. Indexing: To optimize query performance.

#### **Dataset Description:**

The dataset includes the following features: -

**emp\_record\_table:** It contains the information of all the employees.

- EMP\_ID – ID of the employee
- FIRST\_NAME – First name of the employee
- LAST\_NAME – Last name of the employee
- GENDER – Gender of the employee
- ROLE – Post of the employee
- DEPT – Field of the employee
- EXP – Years of experience the employee has
- COUNTRY – Country in which the employee is presently living
- CONTINENT – Continent in which the country is
- SALARY – Salary of the employee
- EMP\_RATING – Performance rating of the employee
- MANAGER\_ID – The manager under which the employee is assigned
- PROJ\_ID – The project on which the employee is working or has worked on

**Proj\_table:** It contains information about the projects.

- PROJECT\_ID – ID for the project
- PROJ\_Name – Name of the project
- DOMAIN – Field of the project
- START\_DATE – Day the project began
- CLOSURE\_DATE – Day the project was or will be completed
- DEV\_QTR – Quarter in which the project was scheduled
- STATUS – Status of the project currently

**Data\_science\_team:** It contains information about all the employees in the Data Science team.

- EMP\_ID – ID of the employee
- FIRST\_NAME – First name of the employee
- LAST\_NAME – Last name of the employee
- GENDER – Gender of the employee
- ROLE – Post of the employee
- DEPT – Field of the employee
- EXP – Years of experience the employee has
- COUNTRY – Country in which the employee is presently living
- CONTINENT – Continent in which the country is

## **PROJECT TASKS:**

**Q 1.** 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.

### **Solution:**

#### **1. Launch MySQL Workbench:**

Connect to mysql database. On the left side of MySQL Workbench, right-click and select 'Create Schema'. Name the new schema as **employee**, then click 'Apply'. Navigate to the newly created schema “**employee**”, click on it, and proceed to Tables option. Right-click on 'Tables'.

#### **2. Navigate to Import Wizard:**

- Go to File -> Import Data - > From File... **import data\_science\_team.csv**.

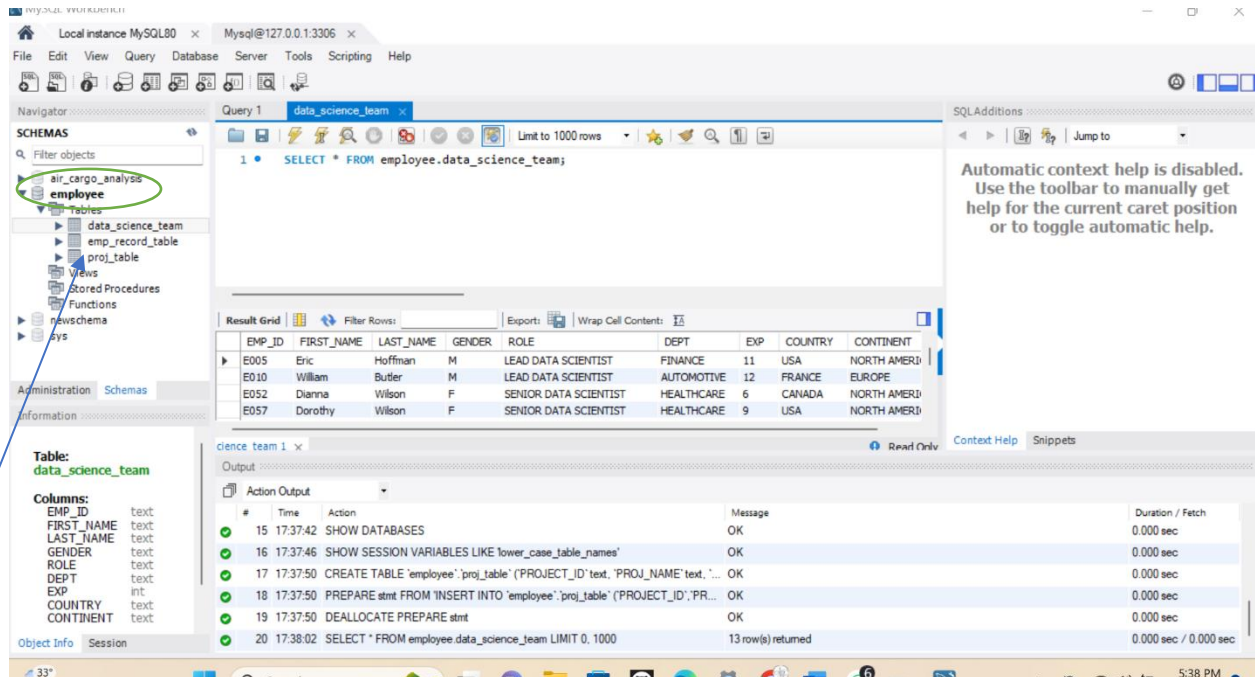
### 3. Specify CSV File:

- Select your CSV file “**import data\_science\_team.csv**” using the file browser.

### 6. Execute Import:

- Click Import or Next to execute the import process.

Same process repeats from second step for importing other CSV files such as “**proj\_table.csv**” and “**emp\_record\_table.csv**” file.



**Q 2. Create an ER diagram for the given **employee** database.**

### **Solution:**

To create an ER (Entity-Relationship) diagram for the given employee database, we will identify the entities, their attributes, and the relationships between them.

### **Entities and Attributes:**

1. emp\_record\_table:

- EMP\_ID (Primary Key)
- FIRST\_NAME
- LAST\_NAME
- GENDER
- ROLE
- DEPT
- EXP
- COUNTRY
- CONTINENT
- SALARY
- EMP\_RATING
- MANAGER\_ID (Foreign Key referencing EMP\_ID)
- PROJ\_ID (Foreign Key referencing Proj\_table.PROJECT\_ID)

2. Proj\_table:

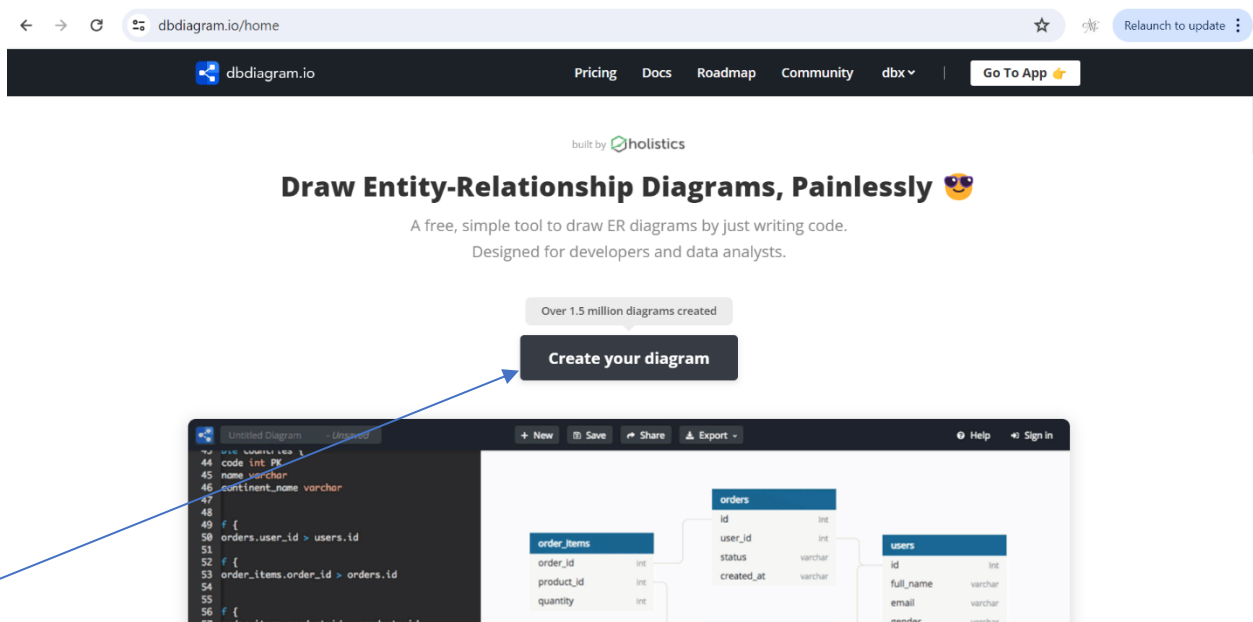
- PROJECT\_ID (Primary Key)
- PROJ\_Name
- DOMAIN
- START\_DATE
- CLOSURE\_DATE
- DEV\_QTR
- STATUS

3. Data science team:

- EMP\_ID (Primary Key, Foreign Key referencing emp\_record\_table.EMP\_ID)

- FIRST\_NAME
- LAST\_NAME
- GENDER
- ROLE
- DEPT
- EXP
- COUNTRY
- CONTINENT

I used [dbDiagram.io](https://dbdiagram.io) to create an ER diagram, and here is the link <https://dbdiagram.io>.

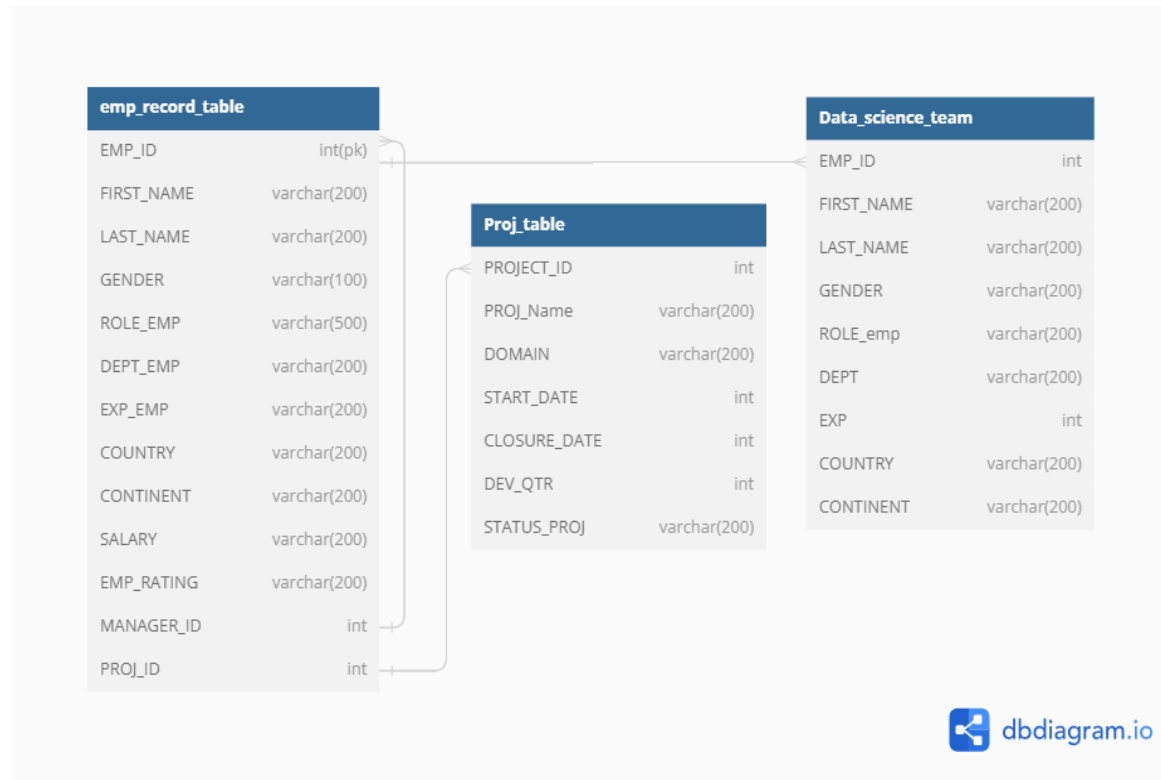


### Relationships:

- One-to-Many relationship between [emp\\_record\\_table](#) and [Proj\\_table](#) (one employee can work on many projects, one project can have many employees).
- One-to-Many self-relationship in [emp\\_record table](#) for [MANAGER ID](#) (one manager can manage many employees).

-One-to-One relationship between emp\_record\_table and Data\_science\_team (each data science team member is a unique employee).

Here is the source code for the ER diagram [https://dbdiagram.io/d/ER\\_DIAGRAM\\_1-667edf579939893dae8d8fd7](https://dbdiagram.io/d/ER_DIAGRAM_1-667edf579939893dae8d8fd7).



Here is the textual representation of the ER diagram:

[emp\_record\_table]

EMP\_ID (PK)

FIRST\_NAME

LAST\_NAME

GENDER

ROLE\_EMP

DEPT\_EMP

EXP\_EMP

COUNTRY

CONTINENT

SALARY

EMP\_RATING

MANAGER\_ID (FK -> emp\_record\_table.EMP\_ID)

PROJ\_ID (FK -> Proj\_table.PROJECT\_ID)

| 1

|

|

| \*

[Proj\_table]

PROJECT\_ID (PK)

PROJ\_Name

DOMAIN

START\_DATE

CLOSURE\_DATE

DEV\_QTR

STATUS

| 1

|

|

|

| 1

[Data\_science\_team]

EMP\_ID (PK, FK -> emp\_record\_table.EMP\_ID)

FIRST\_NAME

LAST\_NAME

GENDER

ROLE\_emp

DEPT

EXP

COUNTRY

CONTINENT

Q 3. 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.

**Solution:**

SELECT

EMP\_ID, FIRST\_NAME, LAST\_NAME, GENDER, DEPT

FROM

emp\_record\_table;

(PRESS CONTROL + ENTER)



Query 1   data\_science\_team   emp\_record\_table

Limit to 1000 rows

```

2 • SELECT
3     EMP_ID,
4     FIRST_NAME,
5     LAST_NAME,
6     GENDER,
7     DEPT
8 FROM
9     emp_record_table;

```

Result Grid   Filter Rows:   Export:   Wrap Cell Content:

	EMP_ID	FIRST_NAME	LAST_NAME	GENDER	DEPT
▶	E001	Arthur	Black	M	ALL
	E005	Eric	Hoffman	M	FINANCE
	E010	William	Butler	M	AUTOMOTIVE
	E052	Dianna	Wilson	F	HEALTHCARE
	E057	Dorothy	Wilson	F	HEALTHCARE
	E083	Patrick	Voltz	M	HEALTHCARE

Q4. Write a query to fetch EMP\_ID, FIRST\_NAME, LAST\_NAME, GENDER, DEPARTMENT, and EMP\_RATING if the EMP\_RATING is:

- less than two:

**Solution:**

```

SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
FROM emp_record_table WHERE EMP_RATING < 2;

```

Query 1   data\_science\_team   emp\_record\_table

Limit to 1000 rows

```

10 • SELECT
11     EMP_ID, FIRST_NAME, LAST_NAME,
12     GENDER,
13     DEPT,
14     EMP_RATING
15 FROM
16     emp_record_table
17 WHERE
18     EMP_RATING < 2;

```

Result Grid   Filter Rows:   Export:   Wrap Cell Content:

	EMP_ID	FIRST_NAME	LAST_NAME	GENDER	DEPT	EMP_RATING
▶	E057	Dorothy	Wilson	F	HEALTHCARE	1
	E532	Claire	Brennan	F	AUTOMOTIVE	1
	E620	Katrina	Allen	F	RETAIL	1

2. greater than four:

**Solution:**

```
SELECT

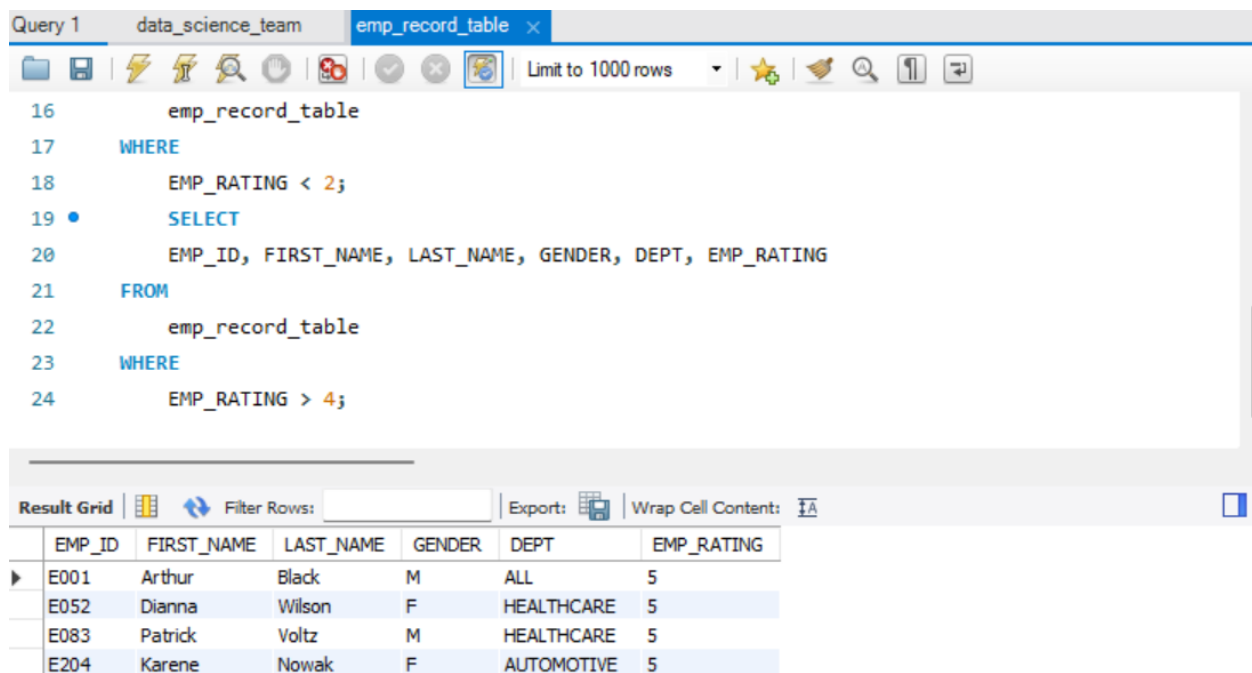
    EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING

FROM

    emp_record_table

WHERE

    EMP_RATING > 4;
```



Query 1 data\_science\_team emp\_record\_table x

Limit to 1000 rows

```
16    emp_record_table
17    WHERE
18    EMP_RATING < 2;
19    SELECT
20    EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
21    FROM
22    emp_record_table
23    WHERE
24    EMP_RATING > 4;
```

Result Grid Filter Rows: Export: Wrap Cell Content:

	EMP_ID	FIRST_NAME	LAST_NAME	GENDER	DEPT	EMP_RATING
▶	E001	Arthur	Black	M	ALL	5
	E052	Dianna	Wilson	F	HEALTHCARE	5
	E083	Patrick	Voltz	M	HEALTHCARE	5
	E204	Karene	Nowak	F	AUTOMOTIVE	5

3. between two and four:

**Solution:**

```
SELECT

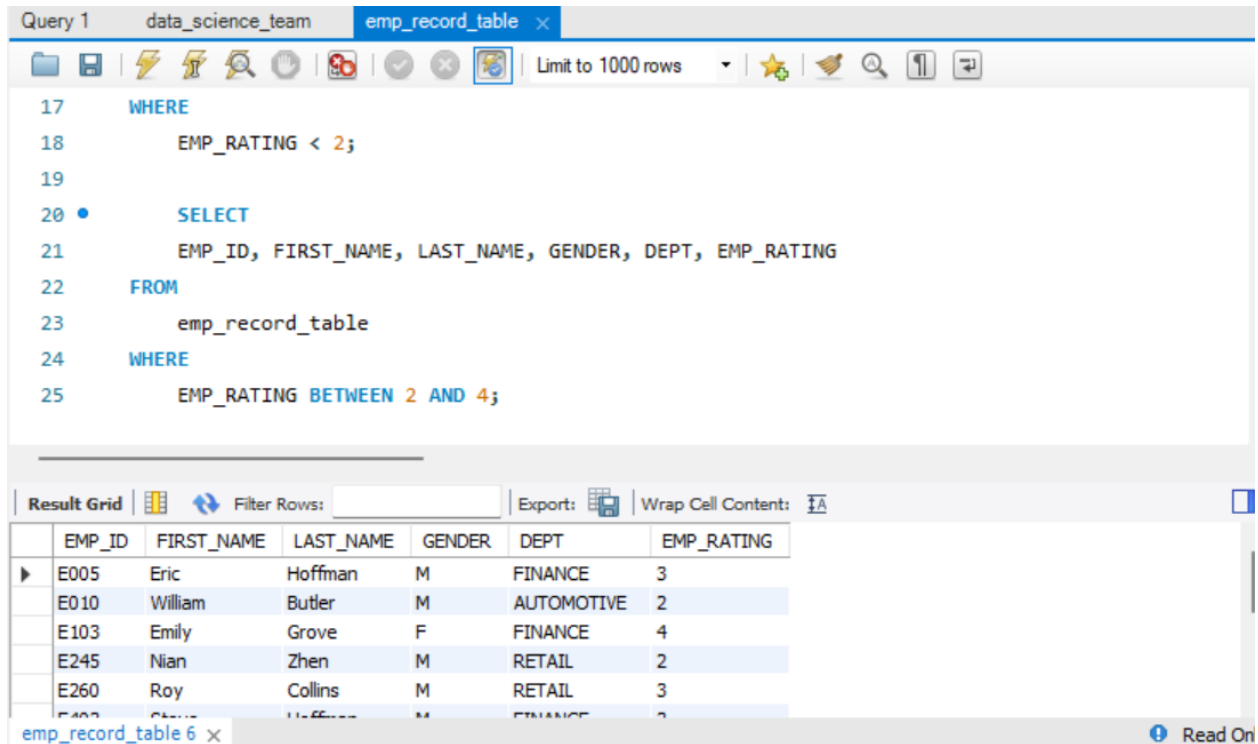
    EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING

FROM
```

emp\_record\_table

WHERE

EMP\_RATING BETWEEN 2 AND 4;



Query 1 data\_science\_team emp\_record\_table x

Limit to 1000 rows

```
17 WHERE
18     EMP_RATING < 2;
19
20 • SELECT
21     EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
22 FROM
23     emp_record_table
24 WHERE
25     EMP_RATING BETWEEN 2 AND 4;
```

Result Grid Filter Rows: Export: Wrap Cell Content:

	EMP_ID	FIRST_NAME	LAST_NAME	GENDER	DEPT	EMP_RATING
▶	E005	Eric	Hoffman	M	FINANCE	3
	E010	William	Butler	M	AUTOMOTIVE	2
	E103	Emily	Grove	F	FINANCE	4
	E245	Nian	Zhen	M	RETAIL	2
	E260	Roy	Collins	M	RETAIL	3
	E402	Steve	Hoffman	M	FINANCE	3

emp\_record\_table 6 x Read On

Q 5. 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.

**Solution:**

SELECT

EMP\_ID,

CONCAT(FIRST\_NAME, ' ', LAST\_NAME) AS NAME,

GENDER, DEPT, EMP\_RATING

FROM

emp\_record\_table

WHERE DEPT = 'FINANCE';

Query 1   data\_science\_team   emp\_record\_table   x   emp\_record\_table

Limit to 1000 rows

```

26
27 •   SELECT
28     EMP_ID,
29     CONCAT(FIRST_NAME, ' ', LAST_NAME) AS NAME,
30     GENDER, DEPT, EMP_RATING
31   FROM
32     emp_record_table
33   WHERE
34     DEPT = 'FINANCE';

```

Result Grid   Filter Rows:   Export:   Wrap Cell Content:

	EMP_ID	NAME	GENDER	DEPT	EMP_RATING
▶	E005	Eric Hoffman	M	FINANCE	3
	E103	Emily Grove	F	FINANCE	4
	E403	Steve Hoffman	M	FINANCE	3

Q 6. Write a query to list only those employees who have someone reporting to them. Also, show the number of reporters (including the President).

**Solution:**

```

SELECT

    EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, EMP_RATING,

    COUNT(EMP_ID) AS NUM_REPORTERS

FROM

    emp_record_table

GROUP BY

    EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, EMP_RATING

ORDER BY

    NUM_REPORTERS DESC;

```

Query 1   data\_science\_team   emp\_record\_table   x   emp\_record\_table

Limit to 1000 rows

```

185 •   SELECT
186     EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, EMP_RATING,
187     COUNT(EMP_ID) AS NUM_REPORTERS
188   FROM
189     emp_record_table
190
191   GROUP BY
192     EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, EMP_RATING
193   ORDER BY
194     NUM_REPORTERS DESC;
195

```

Result Grid   Filter Rows:   Export:   Wrap Cell Content:

	EMP_ID	FIRST_NAME	LAST_NAME	ROLE	DEPT	EMP_RATING	NUM_REPORTERS
▶	E001	Arthur	Black	PRESIDENT	ALL	5	1
	E005	Eric	Hoffman	LEAD DATA SCIENTIST	FINANCE	3	1
	E010	William	Butler	LEAD DATA SCIENTIST	AUTOMOTIVE	2	1
	E052	Dianna	Wilson	SENIOR DATA SCIENTIST	HEALTHCARE	5	1
	E057	Dorothy	Wilson	SENIOR DATA SCIENTIST	HEALTHCARE	1	1

Q 7. Write a query to list down all the employees from the healthcare and finance departments using union. Take data from the employee record table.

**Solution:**

SELECT

EMP\_ID, FIRST\_NAME, LAST\_NAME, GENDER, DEPT, EMP\_RATING

FROM

emp\_record\_table

WHERE

DEPT = 'HEALTHCARE'

UNION

SELECT

EMP\_ID, FIRST\_NAME, LAST\_NAME, GENDER, DEPT, EMP\_RATING

FROM

emp\_record\_table

WHERE

DEPT = 'FINANCE';

The screenshot shows a SQL query editor with a query window and a results grid. The query window contains the following SQL code:

```
45
46 •   SELECT
47     EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
48   FROM
49     emp_record_table
50  WHERE
51     DEPT = 'HEALTHCARE'
52  UNION
53  SELECT
54     EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
55  FROM
56     emp_record_table
57  WHERE
58     DEPT = 'FINANCE';
```

The results grid displays the following data:

EMP_ID	FIRST_NAME	LAST_NAME	GENDER	DEPT	EMP_RATING
E052	Dianna	Wilson	F	HEALTHCARE	5
E057	Dorothy	Wilson	F	HEALTHCARE	1
E083	Patrick	Voltz	M	HEALTHCARE	5
E505	Chad	Wilson	M	HEALTHCARE	2
E005	Eric	Hoffman	M	FINANCE	3

Q 8. 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.

**Solution:**

SELECT

EMP\_ID, FIRST\_NAME, LAST\_NAME, ROLE, DEPT, EMP\_RATING,

max(EMP\_RATING) over (partition by DEPT) AS MAX\_DEPT\_RATING

FROM

emp\_record\_table

GROUP BY

EMP\_ID, FIRST\_NAME, LAST\_NAME, ROLE, DEPT, EMP\_RATING

ORDER BY

DEPT, EMP\_RATING DESC;

The screenshot shows a SQL query editor with a query window titled 'Query 1' and a result grid window titled 'Result 70'. The query is as follows:

```
175 • SELECT
176     EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, EMP_RATING,
177     max(EMP_RATING) over (partition by DEPT) AS MAX_DEPT_RATING
178 FROM
179     emp_record_table
180 GROUP BY
181     EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, EMP_RATING
182 ORDER BY
183     DEPT, EMP_RATING DESC;
184
185
```

The result grid shows the following data:

EMP_ID	FIRST_NAME	LAST_NAME	ROLE	DEPT	EMP_RATING	MAX_DEPT_RATING
E001	Arthur	Black	PRESIDENT	ALL	5	5
E204	Karene	Nowak	SENIOR DATA SCIENTIST	AUTOMOTIVE	5	5
E428	Pete	Allen	MANAGER	AUTOMOTIVE	4	5
E010	William	Butler	LEAD DATA SCIENTIST	AUTOMOTIVE	2	5
E532	Claire	Brennan	ASSOCIATE DATA SCIENTIST	AUTOMOTIVE	1	5

Q 9. Write a query to calculate the minimum and the maximum salary of the employees in each role. Take data from the employee record table.

**Solution:**

SELECT

ROLE,

MIN(SALARY) AS MIN\_SALARY,

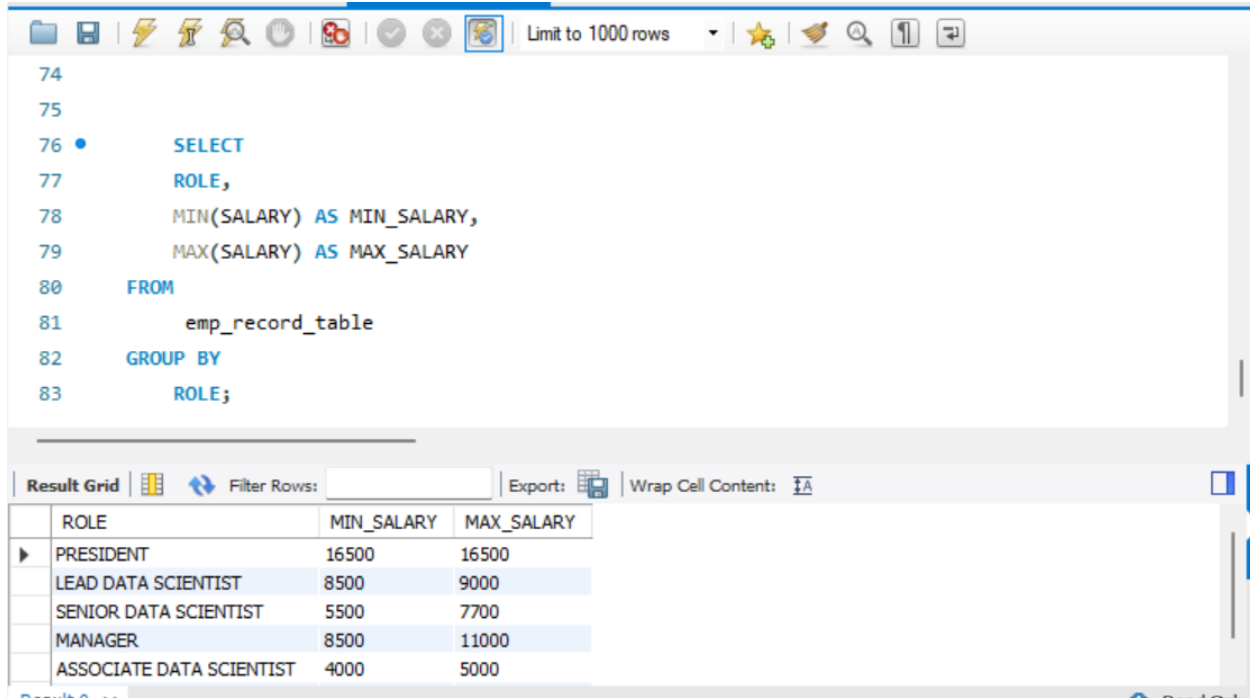
MAX(SALARY) AS MAX\_SALARY

FROM

emp\_record\_table

GROUP BY

ROLE;



```
74
75
76 • SELECT
77     ROLE,
78     MIN(SALARY) AS MIN_SALARY,
79     MAX(SALARY) AS MAX_SALARY
80 FROM
81     emp_record_table
82 GROUP BY
83     ROLE;
```

ROLE	MIN_SALARY	MAX_SALARY
PRESIDENT	16500	16500
LEAD DATA SCIENTIST	8500	9000
SENIOR DATA SCIENTIST	5500	7700
MANAGER	8500	11000
ASSOCIATE DATA SCIENTIST	4000	5000

Q 10. Write a query to assign ranks to each employee based on their experience. Take data from the employee record table.

**Solution:**

SELECT

EMP\_ID, FIRST\_NAME, LAST\_NAME,

ROLE,

DEPT, EXP AS EXPERIENCE,

RANK() OVER (ORDER BY EXP DESC) AS EXPERIENCE\_RANK

FROM

emp\_record\_table;



Query 1   data\_science\_team   emp\_record\_table   x   emp\_record\_table

Limit to 1000 rows

```

81  emp_record_table
82  GROUP BY
83  ROLE;
84  •  SELECT
85  EMP_ID, FIRST_NAME, LAST_NAME,
86  ROLE,
87  DEPT, EXP AS EXPERIENCE,
88  RANK() OVER (ORDER BY EXP DESC) AS EXPERIENCE_RANK
89  FROM
90  emp_record_table;

```

Result Grid   Filter Rows:   Export:   Wrap Cell Content:   Read Only

	EMP_ID	FIRST_NAME	LAST_NAME	ROLE	DEPT	EXPERIENCE	EXPERIENCE_RANK
▶	E001	Arthur	Black	PRESIDENT	ALL	20	1
	E083	Patrick	Voltz	MANAGER	HEALTHCARE	15	2
	E103	Emily	Grove	MANAGER	FINANCE	14	3
	E428	Pete	Allen	MANAGER	AUTOMOTIVE	14	3
	E583	Janet	Hale	MANAGER	RETAIL	14	3

Result 10

Q 11. 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.

**Solution:**

```

CREATE VIEW high_salary_employees_by_country AS

SELECT FIRST_NAME, LAST_NAME, SALARY, COUNTRY

FROM emp_record_table

WHERE SALARY > 6000;

SELECT * FROM high_salary_employees_by_country;

```

Query 1   data\_science\_team   emp\_record\_table   x   emp\_record\_table

Limit to 1000 rows

```
87     DEPT, EXP AS EXPERIENCE,  
88     RANK() OVER (ORDER BY EXP DESC) AS EXPERIENCE_RANK  
89 FROM  
90     emp_record_table;  
91  
92 •   CREATE VIEW high_salary_employees_by_country AS  
93     SELECT FIRST_NAME, LAST_NAME, SALARY, COUNTRY  
94     FROM emp_record_table  
95     WHERE SALARY > 6000;  
96 •   SELECT * FROM high_salary_employees_by_country;
```

Result Grid   Filter Rows:   Export:   Wrap Cell Content:

	FIRST_NAME	LAST_NAME	SALARY	COUNTRY
▶	Arthur	Black	16500	USA
	Eric	Hoffman	8500	USA
	William	Butler	9000	FRANCE
	Dorothy	Wilson	7700	USA
	Patrick	Voltz	9500	USA

Q 12. Write a nested query to find employees with experience of more than ten years. Take data from the employee record table.

**Solution:**

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, EXP  
  
FROM emp_record_table  
  
WHERE EXP > 10;
```

Query 1   data\_science\_team   emp\_record\_table   x   emp\_record\_table

Limit to 1000 rows

```
90      emp_record_table;
91
92 •      CREATE VIEW high_salary_employees_by_country AS
93      SELECT  FIRST_NAME, LAST_NAME, SALARY, COUNTRY
94      FROM emp_record_table
95      WHERE   SALARY > 6000;
96 •      SELECT * FROM high_salary_employees_by_country;
97 •      SELECT EMP_ID, FIRST_NAME, LAST_NAME, EXP
98      FROM emp_record_table
99      WHERE  EXP > 10;
```

Result Grid   Filter Rows:   Export:   Wrap Cell Content:

	EMP_ID	FIRST_NAME	LAST_NAME	EXP
▶	E001	Arthur	Black	20
	E005	Eric	Hoffman	11
	E010	William	Butler	12
	E083	Patrick	Voltz	15
	E103	Emily	Grove	14

Q 13. 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.

**Solution:**

```
CREATE PROCEDURE GetEmployeesWithExperience()

BEGIN

    SELECT

        EMP_ID, FIRST_NAME, LAST_NAME, GENDER, ROLE, DEPT, EXP,
        COUNTRY, CONTINENT, SALARY, EMP_RATING, MANAGER_ID, PROJ_ID

    FROM

        emp_record_table

    WHERE

        EXP > 3;

END // DELIMITER;

CALL GetEmployeesWithExperience();
```

Query 1   data\_science\_team   emp\_record\_table   x   emp\_record\_table

Limit to 1000 rows

```

123 • CREATE PROCEDURE GetEmployeesWithExperience()
124 BEGIN
125     SELECT
126         EMP_ID, FIRST_NAME, LAST_NAME, GENDER, ROLE, DEPT, EXP, COUNTRY, CONTINENT, SALARY,
127         EMP_RATING, MANAGER_ID, PROJ_ID
128     FROM
129         emp_record_table
130     WHERE
131         EXP > 3;
132 END //
133
134 DELIMITER ;

```

Result Grid   Filter Rows:   Export:   Wrap Cell Content:   Read Only

	EMP_ID	FIRST_NAME	LAST_NAME	GENDER	ROLE	DEPT	EXP	COUNTRY	CONTINENT
▶	E001	Arthur	Black	M	PRESIDENT	ALL	20	USA	NORTH AMERICA
	E005	Eric	Hoffman	M	LEAD DATA SCIENTIST	FINANCE	11	USA	NORTH AMERICA
	E010	William	Butler	M	LEAD DATA SCIENTIST	AUTOMOTIVE	12	FRANCE	EUROPE
	E052	Dianna	Wilson	F	SENIOR DATA SCIENTIST	HEALTHCARE	6	CANADA	NORTH AMERICA

Result 50 x   Read Only

Output

Action Output

#	Time	Action	Message
106	11:05:31	CREATE PROCEDURE GetEmployeesWithExperience() BEGIN SELECT ...	0 row(s) affected

Q 14. 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.

The standard being:

For an employee with experience less than or equal to 2 years assign 'JUNIOR DATA SCIENTIST',

For an employee with the experience of 2 to 5 years assign 'ASSOCIATE DATA SCIENTIST',

For an employee with the experience of 5 to 10 years assign 'SENIOR DATA SCIENTIST',

For an employee with the experience of 10 to 12 years assign 'LEAD DATA SCIENTIST',

For an employee with the experience of 12 to 16 years assign 'MANAGER'.

**Solution:**

```
CREATE FUNCTION CheckJobProfile(exp INT, role VARCHAR(50))
```

```
RETURNS VARCHAR(50)
```

DETERMINISTIC

BEGIN

DECLARE expected\_role VARCHAR(50);

IF exp <= 2 THEN

SET expected\_role = 'JUNIOR DATA SCIENTIST';

ELSEIF exp > 2 AND exp <= 5 THEN

SET expected\_role = 'ASSOCIATE DATA SCIENTIST';

ELSEIF exp > 5 AND exp <= 10 THEN

SET expected\_role = 'SENIOR DATA SCIENTIST';

ELSEIF exp > 10 AND exp <= 12 THEN

SET expected\_role = 'LEAD DATA SCIENTIST';

ELSEIF exp > 12 AND exp <= 16 THEN

SET expected\_role = 'MANAGER';

ELSE

SET expected\_role = 'UNKNOWN'; -- For experience greater than 16 years

END IF;

IF role = expected\_role THEN

RETURN 'MATCHES';

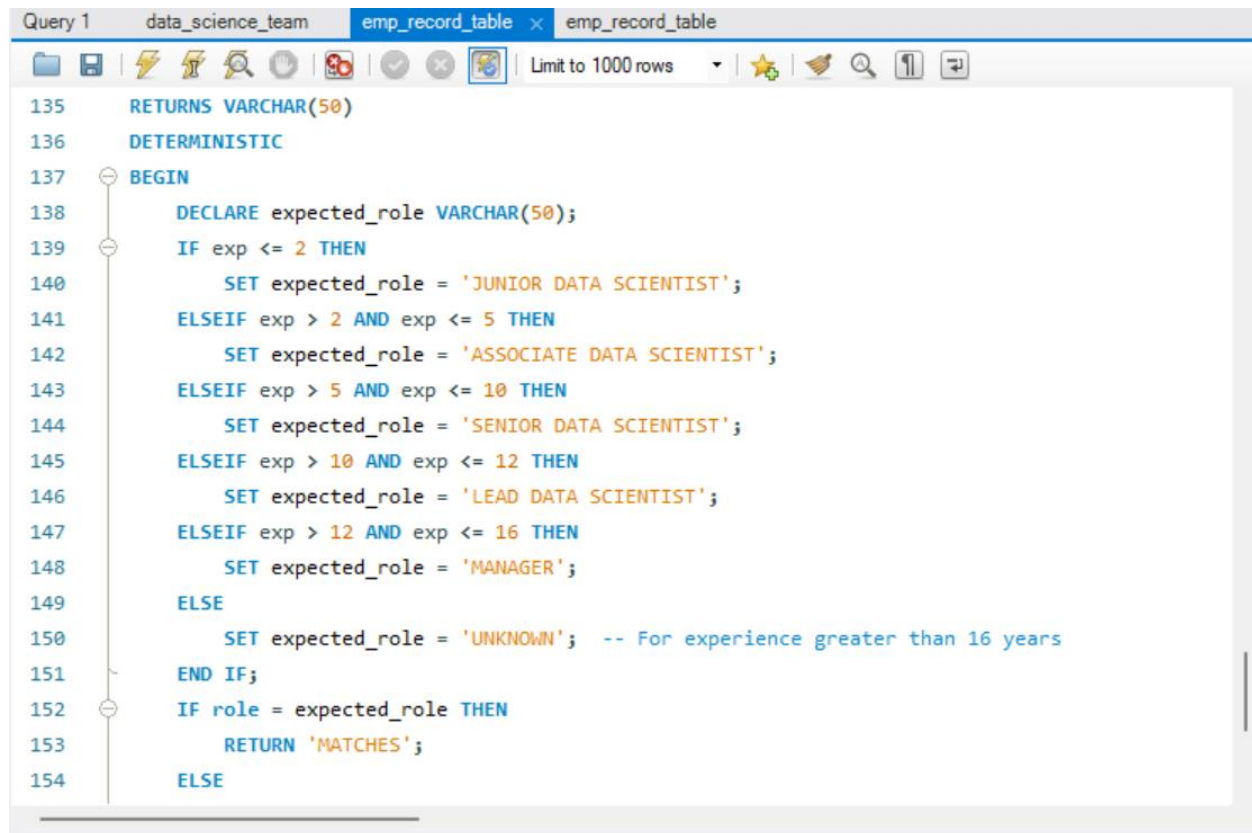
ELSE

RETURN 'DOES NOT MATCH';

END IF;

END //

DELIMITER ;

A screenshot of a SQL IDE window titled 'Query 1' with tabs for 'data\_science\_team', 'emp\_record\_table', and 'emp\_record\_table'. The code is a PL/SQL function definition. It starts with 'RETURNS VARCHAR(50)' and 'DETERMINISTIC'. The function body begins with 'BEGIN', followed by a 'DECLARE' statement for 'expected\_role VARCHAR(50);'. An 'IF' statement checks 'exp <= 2'. If true, 'SET expected\_role = 'JUNIOR DATA SCIENTIST';'. An 'ELSEIF' clause checks 'exp > 2 AND exp <= 5', setting 'expected\_role' to 'ASSOCIATE DATA SCIENTIST'. Another 'ELSEIF' clause checks 'exp > 5 AND exp <= 10', setting it to 'SENIOR DATA SCIENTIST'. A fourth 'ELSEIF' clause checks 'exp > 10 AND exp <= 12', setting it to 'LEAD DATA SCIENTIST'. A fifth 'ELSEIF' clause checks 'exp > 12 AND exp <= 16', setting it to 'MANAGER'. An 'ELSE' clause sets 'expected\_role = 'UNKNOWN';' with a comment '-- For experience greater than 16 years'. The 'IF' statement ends with 'END IF;'. Then, another 'IF' statement checks 'role = expected\_role'. If true, it returns 'MATCHES';'. Otherwise, it returns an empty string (implied by the 'ELSE' without a return statement, though the code shows 'ELSE' without a return, which might be a typo for 'RETURN '';'). The function ends with 'END'.

--For check function is created

SHOW FUNCTION STATUS

WHERE Db = 'employee'

AND Name = 'CheckJobProfile';

SELECT CheckJobProfile(3, 'ASSOCIATE DATA SCIENTIST') AS result;

The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and a 'Limit to 1000 rows' dropdown. The query window contains the following SQL code:

```
154     ELSE
155         RETURN 'DOES NOT MATCH';
156     END IF;
157 END //
158
159 DELIMITER ;
160
161 •     SELECT CheckJobProfile(3, 'ASSOCIATE DATA SCIENTIST') AS result;
162
163 •     SHOW FUNCTION STATUS
164     WHERE Db = 'employee'
165     AND Name = 'CheckJobProfile';
```

Below the query window is the 'Result Grid' section. It includes a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The grid itself has two columns: 'result' and 'MATCHES'. The 'MATCHES' column is currently expanded, showing a list of matches.

Q 15. 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.

**Solution:**

```
CREATE INDEX idx_first_name ON emp_record_table(FIRST_NAME(50));
```

```
EXPLAIN SELECT * FROM emp_record_table WHERE FIRST_NAME = 'Eric';
```

```
SHOW INDEX FROM emp_record_table;
```

```
SELECT * FROM emp_record_table WHERE FIRST_NAME = 'Eric';
```

Query 1 data\_science\_team emp\_record\_table x emp\_record\_table

Limit to 1000 rows

```

166
167 • CREATE INDEX idx_first_name ON emp_record_table(FIRST_NAME(50));
168
169 • EXPLAIN SELECT * FROM emp_record_table WHERE FIRST_NAME = 'Eric';
170 • show INDEX FROM emp_record_table;
171
172
173
174
175
176

```

Result Grid Filter Rows: Export: Wrap Cell Content:

	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Ex
▶	1	SIMPLE	emp_record_table	NULL	ref	idx_first_name	idx_first_name	203	const	1	100.00	Usi

Q 16. 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).

**Solution:**

```

SELECT
    EMP_ID, FIRST_NAME, LAST_NAME, SALARY, EMP_RATING,
    (0.05 * SALARY * EMP_RATING) AS BONUS
FROM
    emp_record_table;

```



Query 1   data\_science\_team   emp\_record\_table x emp\_record\_table

Limit to 1000 rows

```

101 • SELECT
102     EMP_ID,
103     FIRST_NAME,
104     LAST_NAME,
105     SALARY,
106     EMP_RATING,
107     (0.05 * SALARY * EMP_RATING) AS BONUS
108 FROM
109     emp_record_table;
110
...

```

Result Grid   Filter Rows:   Export:   Wrap Cell Content:   Read Only

	EMP_ID	FIRST_NAME	LAST_NAME	SALARY	EMP_RATING	BONUS
▶	E001	Arthur	Black	16500	5	4125.00
	E005	Eric	Hoffman	8500	3	1275.00
	E010	William	Butler	9000	2	900.00
	E052	Dianna	Wilson	5500	5	1375.00
	E057	Dorothy	Wilson	7700	1	385.00

Result 13 x

Q 17. Write a query to calculate the average salary distribution based on the continent and country. Take data from the employee record table.

### Solution:

```

SELECT

    CONTINENT, COUNTRY,

    AVG(SALARY) AS AVERAGE_SALARY

FROM

    emp_record_table

GROUP BY

    CONTINENT, COUNTRY

ORDER BY

    CONTINENT, COUNTRY;

```

Query 1

data\_science\_team

emp\_record\_table

emp\_record\_table

</