Employee Performance Mapping – SQL Project Submission

Submitted by: Mohsin Bashir Najar
Date: 23/02/2025
Course Name: Data Acquisition and Manipulation using SQL
Project Name: Employee Performance Mapping

Database Setup

Task #1
Created Database, Make Database Active, Created Tables & Imported Data

```
CREATE DATABASE IF NOT EXISTS employee;
USE employee;
```

Created Tables:

```
CREATE TABLE project_table (
        project_id VARCHAR(255) PRIMARY KEY,
  project_name VARCHAR(255) NOT NULL,
  domain VARCHAR(255) NOT NULL,
  start_date DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  closure_date DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  dev_qtr ENUM("Q1", "Q2", "Q3", "Q4") NOT NULL,
  status VARCHAR(255) NOT NULL,
  CONSTRAINT project_table_project_name_unique UNIQUE(project_name)
);
```

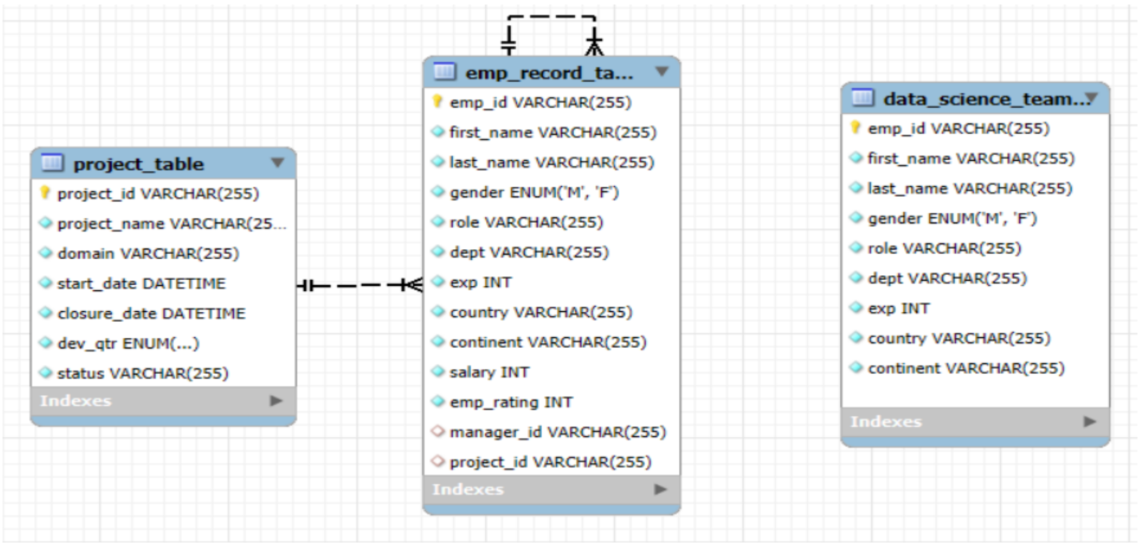
```
CREATE TABLE emp_record_table (
        emp_id VARCHAR(255) PRIMARY KEY,
        first_name VARCHAR(255) NOT NULL,
         last_name VARCHAR(255) NOT NULL,
          gender ENUM('M', 'F') NOT NULL,
        role VARCHAR(255) NOT NULL,
        dept VARCHAR(255) NOT NULL,
         exp INT NOT NULL,
        country VARCHAR(255) NOT NULL,
        continent VARCHAR(255) NOT NULL,
        salary INT NOT NULL,
        emp_rating INT NOT NULL,
        manager_id VARCHAR(255),
        project_id VARCHAR(255),
        CONSTRAINT emp_record_table_exp_check CHECK(exp >= 0),
        CONSTRAINT emp_record_table_salary_check CHECK(salary > 2500),
	CONSTRAINT emp_record_table_emp_rating_check CHECK(emp_rating
BETWEEN 1 AND 5),
	CONSTRAINT emp_record_table_project_id_fk FOREIGN KEY(project_id)
REFERENCES project_table(project_id) ON DELETE CASCADE ON UPDATE CASCADE,
	CONSTRAINT emp_record_table_manager_id_fk FOREIGN KEY(manager_id)
REFERENCES emp_record_table(emp_id)
);

CREATE TABLE data_science_team (
	emp_id VARCHAR(255) PRIMARY KEY,
	first_name VARCHAR(255) NOT NULL,
	last_name VARCHAR(255) NOT NULL,
	gender ENUM('M', 'F') NOT NULL,
	role VARCHAR(255) NOT NULL,
	dept VARCHAR(255) NOT NULL,
	exp INT NOT NULL,
	country VARCHAR(255) NOT NULL,
	continent VARCHAR(255) NOT NULL,
	CONSTRAINT data_science_team_exp_check CHECK(exp >= 0)
);
```

Imported Tables After Data Cleaning

emp_record_table.csv
proj_table.csv
data_science_team.csv

Task #2
Created ER Diagram Using Reverse Engineering



SQL Queries & Outputs

This section includes the required queries along with the corresponding screenshots of the outputs.

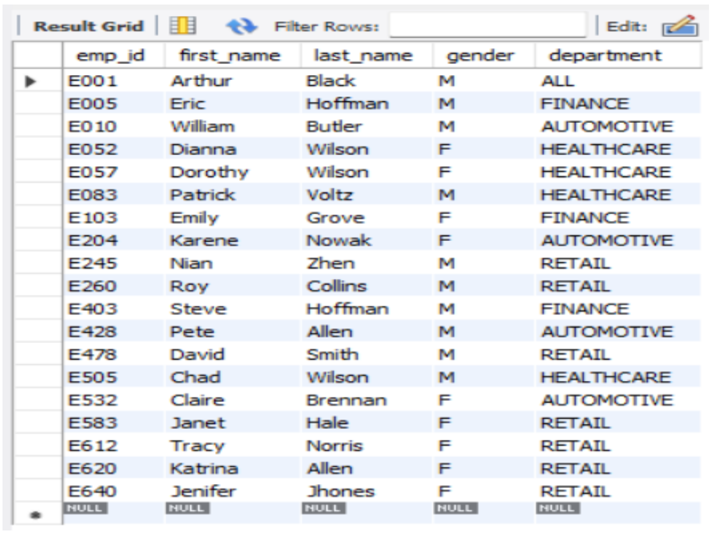
Task #3

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.

Query

SELECT emp_id, first_name, last_name, gender, dept AS department
FROM emp_record_table;

Screenshot of Output



Result Grid | Filter Rows: | Edit:

emp_id	first_name	last_name	gender	department
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
E103	Emily	Grove	F	FINANCE
E204	Karene	Nowak	F	AUTOMOTIVE
E245	Nian	Zhen	M	RETAIL
E260	Roy	Collins	M	RETAIL
E403	Steve	Hoffman	M	FINANCE
E428	Pete	Allen	M	AUTOMOTIVE
E478	David	Smith	M	RETAIL
E505	Chad	Wilson	M	HEALTHCARE
E532	Claire	Brennan	F	AUTOMOTIVE
E583	Janet	Hale	F	RETAIL
E612	Tracy	Norris	F	RETAIL
E620	Katrina	Allen	F	RETAIL
E640	Jenifer	Jhones	F	RETAIL
NULL	NULL	NULL	NULL	NULL

Task #4

Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPARTMENT, and EMP_RATING if the EMP_RATING is:

Less than two

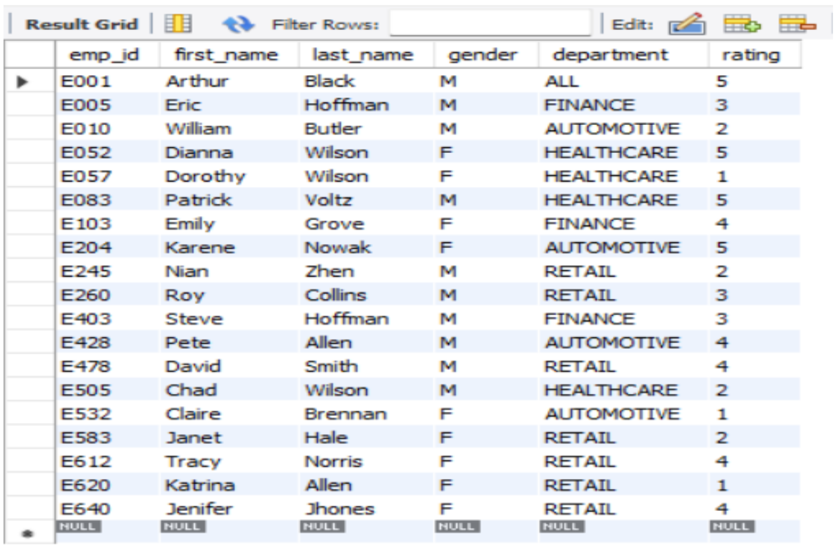
Greater than four

Between two and four

Query

SELECT emp_id, first_name, last_name, gender, dept AS department,
            emp_rating AS rating
FROM emp_record_table
WHERE emp_rating < 2 OR emp_rating > 4 OR emp_rating BETWEEN 2 AND 4;

Screenshot of Output



Result Grid | Filter Rows: | Edit:

emp_id	first_name	last_name	gender	department	rating
E001	Arthur	Black	M	ALL	5
E005	Eric	Hoffman	M	FINANCE	3
E010	William	Butler	M	AUTOMOTIVE	2
E052	Dianna	Wilson	F	HEALTHCARE	5
E057	Dorothy	Wilson	F	HEALTHCARE	1
E083	Patrick	Voltz	M	HEALTHCARE	5
E103	Emily	Grove	F	FINANCE	4
E204	Karene	Nowak	F	AUTOMOTIVE	5
E245	Nian	Zhen	M	RETAIL	2
E260	Roy	Collins	M	RETAIL	3
E403	Steve	Hoffman	M	FINANCE	3
E428	Pete	Allen	M	AUTOMOTIVE	4
E478	David	Smith	M	RETAIL	4
E505	Chad	Wilson	M	HEALTHCARE	2
E532	Claire	Brennan	F	AUTOMOTIVE	1
E583	Janet	Hale	F	RETAIL	2
E612	Tracy	Norris	F	RETAIL	4
E620	Katrina	Allen	F	RETAIL	1
E640	Jenifer	Jhones	F	RETAIL	4
NULL	NULL	NULL	NULL	NULL	NULL

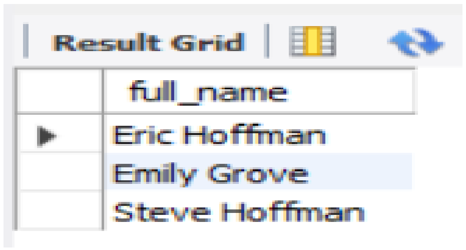
Task #5

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.

Query

```
SELECT  CONCAT_WS(' ', first_name, last_name) full_name
FROM  emp_record_table
WHERE dept = 'FINANCE';
```

Screenshot of Output



Result Grid

full_name
Eric Hoffman
Emily Grove
Steve Hoffman

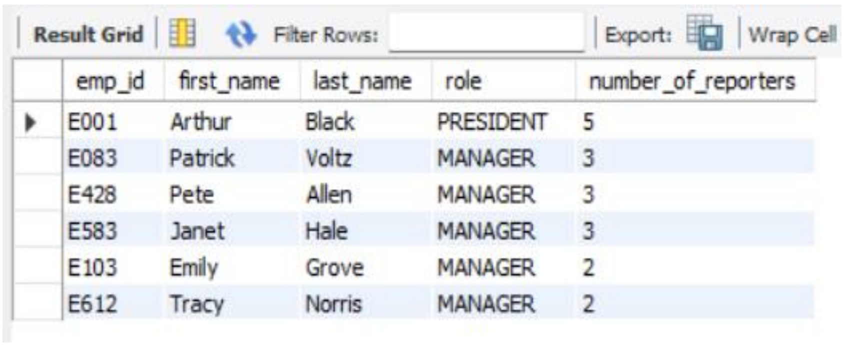
Task #6

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

Query

```
SELECT e.emp_id, e.first_name, e.last_name, e.role,
        COUNT(m.emp_id) number_of_reporters
FROM  emp_record_table e
INNER JOIN emp_record_table m
ON e.emp_id = m.manager_id
GROUP BY e.emp_id;
```

Screenshot of Output



emp_id	first_name	last_name	role	number_of_reporters
E001	Arthur	Black	PRESIDENT	5
E083	Patrick	Voltz	MANAGER	3
E428	Pete	Allen	MANAGER	3
E583	Janet	Hale	MANAGER	3
E103	Emily	Grove	MANAGER	2
E612	Tracy	Norris	MANAGER	2

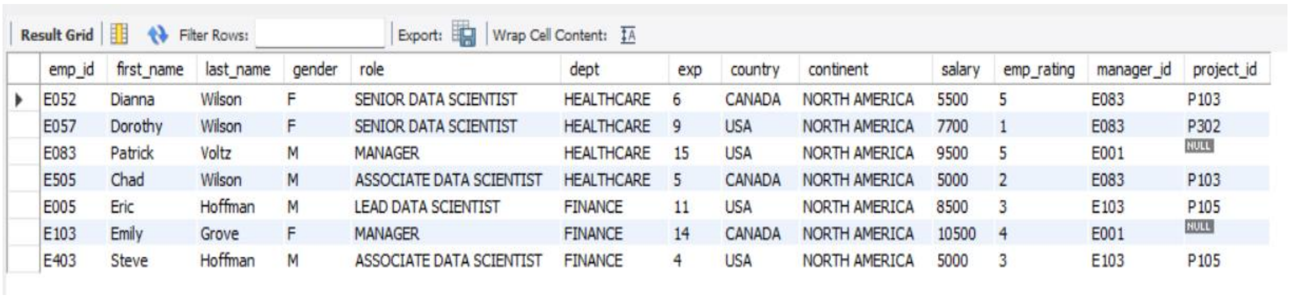
Task #7

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

Query

SELECT * FROM emp_record_table WHERE dept = 'healthcare'
UNION
SELECT * FROM emp_record_table WHERE dept = 'finance';

Screenshot of Output



emp_id	first_name	last_name	gender	role	dept	exp	country	continent	salary	emp_rating	manager_id	project_id
E052	Dianna	Wilson	F	SENIOR DATA SCIENTIST	HEALTHCARE	6	CANADA	NORTH AMERICA	5500	5	E083	P103
E057	Dorothy	Wilson	F	SENIOR DATA SCIENTIST	HEALTHCARE	9	USA	NORTH AMERICA	7700	1	E083	P302
E083	Patrick	Voltz	M	MANAGER	HEALTHCARE	15	USA	NORTH AMERICA	9500	5	E001	NULL
E505	Chad	Wilson	M	ASSOCIATE DATA SCIENTIST	HEALTHCARE	5	CANADA	NORTH AMERICA	5000	2	E083	P103
E005	Eric	Hoffman	M	LEAD DATA SCIENTIST	FINANCE	11	USA	NORTH AMERICA	8500	3	E103	P105
E103	Emily	Grove	F	MANAGER	FINANCE	14	CANADA	NORTH AMERICA	10500	4	E001	NULL
E403	Steve	Hoffman	M	ASSOCIATE DATA SCIENTIST	FINANCE	4	USA	NORTH AMERICA	5000	3	E103	P105

Task #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.

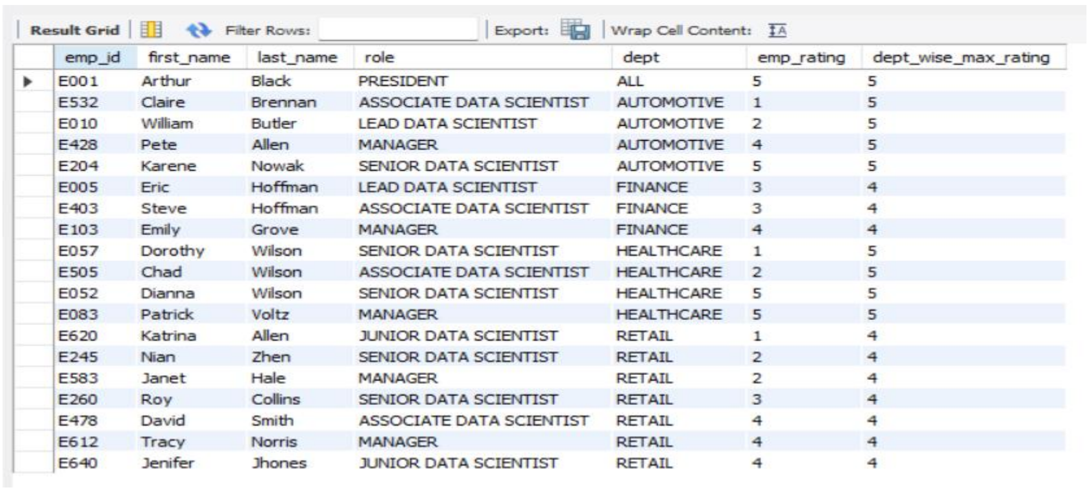
Query

```
SELECT e.EMP_ID, e.FIRST_NAME, e.LAST_NAME, e.ROLE, e.DEPT, e.EMP_RATING,
        d.Max_Dept_Rating
FROM emp_record_table e
JOIN (
  SELECT DEPT, MAX(EMP_RATING) AS Max_Dept_Rating
  FROM emp_record_table
  GROUP BY DEPT
) d
ON e.DEPT = d.DEPT;
```

OR

```
SELECT emp_id, first_name, last_name, role, dept, emp_rating,
            max(emp_rating) OVER(PARTITION BY dept) AS dept_wise_max_rating
FROM  emp_record_table
ORDER BY dept, emp_rating;
```

Screenshot of Output



emp_id	first_name	last_name	role	dept	emp_rating	dept_wise_max_rating
E001	Arthur	Black	PRESIDENT	ALL	5	5
E532	Claire	Brennan	ASSOCIATE DATA SCIENTIST	AUTOMOTIVE	1	5
E010	William	Butler	LEAD DATA SCIENTIST	AUTOMOTIVE	2	5
E428	Pete	Allen	MANAGER	AUTOMOTIVE	4	5
E204	Karene	Nowak	SENIOR DATA SCIENTIST	AUTOMOTIVE	5	5
E005	Eric	Hoffman	LEAD DATA SCIENTIST	FINANCE	3	4
E403	Steve	Hoffman	ASSOCIATE DATA SCIENTIST	FINANCE	3	4
E103	Emily	Grove	MANAGER	FINANCE	4	4
E057	Dorothy	Wilson	SENIOR DATA SCIENTIST	HEALTHCARE	1	5
E505	Chad	Wilson	ASSOCIATE DATA SCIENTIST	HEALTHCARE	2	5
E052	Dianna	Wilson	SENIOR DATA SCIENTIST	HEALTHCARE	5	5
E083	Patrick	Voltz	MANAGER	HEALTHCARE	5	5
E620	Katrina	Allen	JUNIOR DATA SCIENTIST	RETAIL	1	4
E245	Nian	Zhen	SENIOR DATA SCIENTIST	RETAIL	2	4
E583	Janet	Hale	MANAGER	RETAIL	2	4
E260	Roy	Collins	SENIOR DATA SCIENTIST	RETAIL	3	4
E478	David	Smith	ASSOCIATE DATA SCIENTIST	RETAIL	4	4
E612	Tracy	Norris	MANAGER	RETAIL	4	4
E640	Jenifer	Jhones	JUNIOR DATA SCIENTIST	RETAIL	4	4

Task #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.

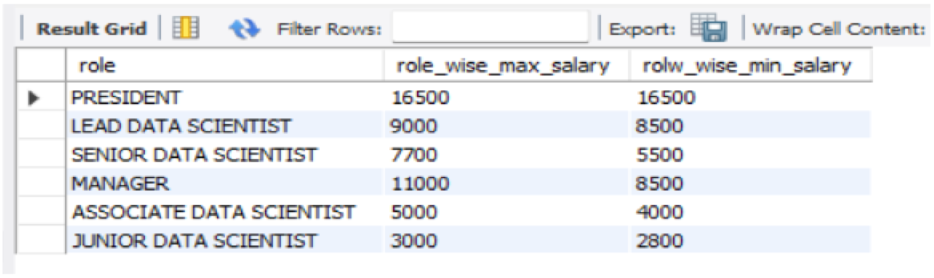
Query

```
SELECT role,
            max(salary) role_wise_max_salary,
            min(salary) rolw_wise_min_salary
FROM  emp_record_table
GROUP BY role;
```

OR

```
SELECT role,
           max(salary) OVER w role_wise_max_salary,
           min(salary) OVER w rolw_wise_min_salary
FROM emp_record_table
WINDOW w AS (PARTITION BY role);
```

Screenshot of Output

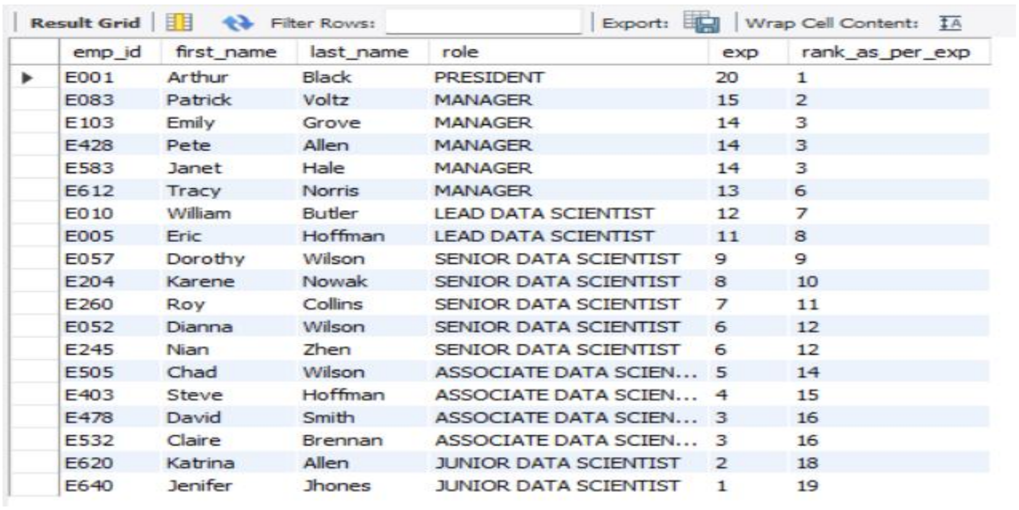


role	role_wise_max_salary	rolw_wise_min_salary
PRESIDENT	16500	16500
LEAD DATA SCIENTIST	9000	8500
SENIOR DATA SCIENTIST	7700	5500
MANAGER	11000	8500
ASSOCIATE DATA SCIENTIST	5000	4000
JUNIOR DATA SCIENTIST	3000	2800

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

Query
```
SELECT emp_id, first_name, last_name, role, exp,
        RANK() OVER(ORDER BY exp DESC) rank_as_per_exp
FROM  emp_record_table;
```

Screenshot of Output



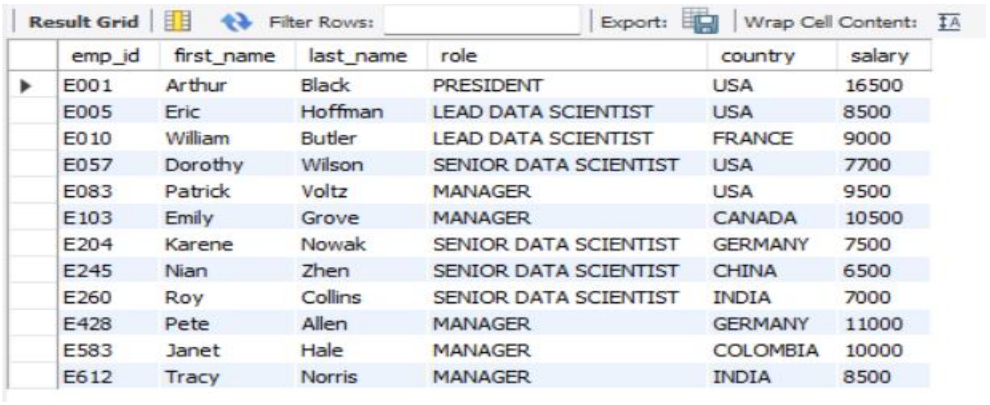
emp_id	first_name	last_name	role	exp	rank_as_per_exp
E001	Arthur	Black	PRESIDENT	20	1
E083	Patrick	Voltz	MANAGER	15	2
E103	Emily	Grove	MANAGER	14	3
E428	Pete	Allen	MANAGER	14	3
E583	Janet	Hale	MANAGER	14	3
E612	Tracy	Norris	MANAGER	13	6
E010	William	Butler	LEAD DATA SCIENTIST	12	7
E005	Eric	Hoffman	LEAD DATA SCIENTIST	11	8
E057	Dorothy	Wilson	SENIOR DATA SCIENTIST	9	9
E204	Karene	Nowak	SENIOR DATA SCIENTIST	8	10
E260	Roy	Collins	SENIOR DATA SCIENTIST	7	11
E052	Dianna	Wilson	SENIOR DATA SCIENTIST	6	12
E245	Nian	Zhen	SENIOR DATA SCIENTIST	6	12
E505	Chad	Wilson	ASSOCIATE DATA SCIEN...	5	14
E403	Steve	Hoffman	ASSOCIATE DATA SCIEN...	4	15
E478	David	Smith	ASSOCIATE DATA SCIEN...	3	16
E532	Claire	Brennan	ASSOCIATE DATA SCIEN...	3	16
E620	Katrina	Allen	JUNIOR DATA SCIENTIST	2	18
E640	Jenifer	Jhones	JUNIOR DATA SCIENTIST	1	19

Task #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.

Query
```
CREATE VIEW employees_salary_gt_6000 AS
SELECT emp_id, first_name, last_name, role, country, salary
FROM  emp_record_table
WHERE salary > 6000;

SELECT * FROM employees_salary_gt_6000;
```

Screenshot of Output



emp_id	first_name	last_name	role	country	salary
E001	Arthur	Black	PRESIDENT	USA	16500
E005	Eric	Hoffman	LEAD DATA SCIENTIST	USA	8500
E010	William	Butler	LEAD DATA SCIENTIST	FRANCE	9000
E057	Dorothy	Wilson	SENIOR DATA SCIENTIST	USA	7700
E083	Patrick	Voltz	MANAGER	USA	9500
E103	Emily	Grove	MANAGER	CANADA	10500
E204	Karene	Nowak	SENIOR DATA SCIENTIST	GERMANY	7500
E245	Nian	Zhen	SENIOR DATA SCIENTIST	CHINA	6500
E260	Roy	Collins	SENIOR DATA SCIENTIST	INDIA	7000
E428	Pete	Allen	MANAGER	GERMANY	11000
E583	Janet	Hale	MANAGER	COLOMBIA	10000
E612	Tracy	Norris	MANAGER	INDIA	8500

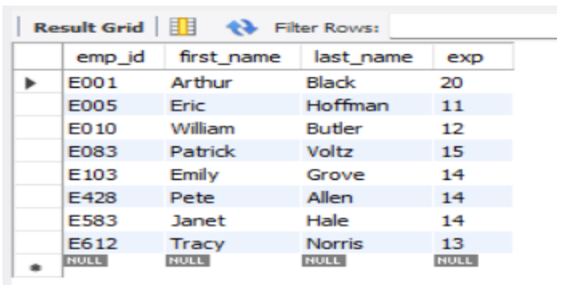
Task #12

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

Query

```
SELECT emp_id, first_name, last_name, exp
FROM  emp_record_table
WHERE exp > (SELECT 10);
```

Screenshot of Output



Result Grid | Filter Rows:

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
E428	Pete	Allen	14
E583	Janet	Hale	14
E612	Tracy	Norris	13
NULL	NULL	NULL	NULL

Task #13

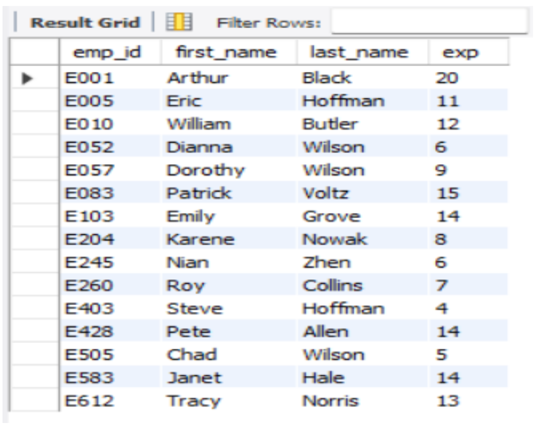
Write a query to create a stored procedure to retrieve the details of the employees whose experience is more than three years. Take data from the employee record table.

Query

```
DELIMITER $$
CREATE PROCEDURE get_employees_with_exp_gt_3()
BEGIN
        SELECT emp_id, first_name, last_name, exp
        FROM emp_record_table
        WHERE exp > 3;
END $$
DELIMITER ;

CALL get_employees_with_exp_gt_3();
```

Screenshot of Output



Result Grid | Filter Rows:

emp_id	first_name	last_name	exp
E001	Arthur	Black	20
E005	Eric	Hoffman	11
E010	William	Butler	12
E052	Dianna	Wilson	6
E057	Dorothy	Wilson	9
E083	Patrick	Voltz	15
E103	Emily	Grove	14
E204	Karene	Nowak	8
E245	Nian	Zhen	6
E260	Roy	Collins	7
E403	Steve	Hoffman	4
E428	Pete	Allen	14
E505	Chad	Wilson	5
E583	Janet	Hale	14
E612	Tracy	Norris	13

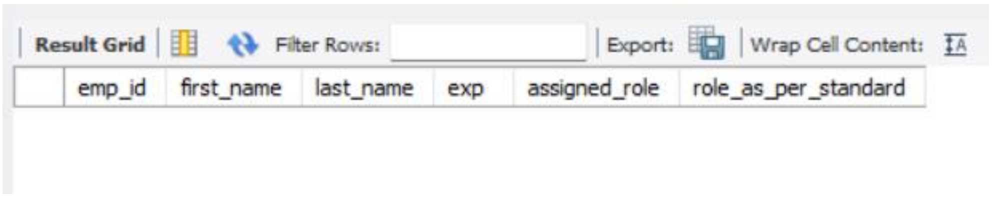
Task #14
Write a query using stored functions in the project table to check whether the job profile assigned to each employee in the data science team matches the organization's set standard. The standard being:

For an employee with experience less than or equal to 2 years
assign 'JUNIORDATA 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'.

Query
```
DELIMITER $$
CREATE FUNCTION check_job_profile_standard(exp INT)
RETURNS VARCHAR(255)
DETERMINISTIC
BEGIN
    DECLARE profile VARCHAR(255);
    IF exp <= 2 THEN
        SET profile = 'JUNIOR DATA SCIENTIST';
    ELSEIF exp > 2 AND exp <= 5 THEN
        SET profile = 'ASSOCIATE DATA SCIENTIST';
    ELSEIF exp > 5 AND exp <= 10 THEN
        SET profile = 'SENIOR DATA SCIENTIST';
    ELSEIF exp > 10 AND exp <= 12 THEN
        SET profile = 'LEAD DATA SCIENTIST';
    ELSEIF exp > 12 AND exp <= 16 THEN
        SET profile = 'MANAGER';
    ELSE
        SET profile = 'UNKNOWN ROLE';
    END IF;
    RETURN profile;
END $$
DELIMITER ;
```

```
SELECT emp_id, first_name, last_name, exp, role assigned_role,
        check_job_profile_standard(exp) role_as_per_standard
FROM  data_science_team
WHERE role != check_job_profile_standard(exp);
```

Screenshot of Output



emp_id	first_name	last_name	exp	assigned_role	role_as_per_standard

Task #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.

Query for Creating Index

```
CREATE INDEX idx_first_name ON emp_record_table(first_name);
```

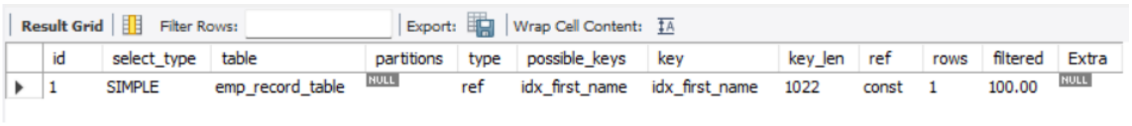
Query for Searching 'Eric'

```
SELECT emp_id, first_name, last_name, role, dept, country, salary
FROM  emp_record_table
WHERE first_name = 'eric';
```

Execution Plan Check

```
EXPLAIN SELECT emp_id, first_name, last_name, role, dept, country, salary
FROM  emp_record_table
WHERE first_name = 'eric';
```

Screenshot of Output



id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	emp_record_table	NULL	ref	idx_first_name	idx_first_name	1022	const	1	100.00	NULL

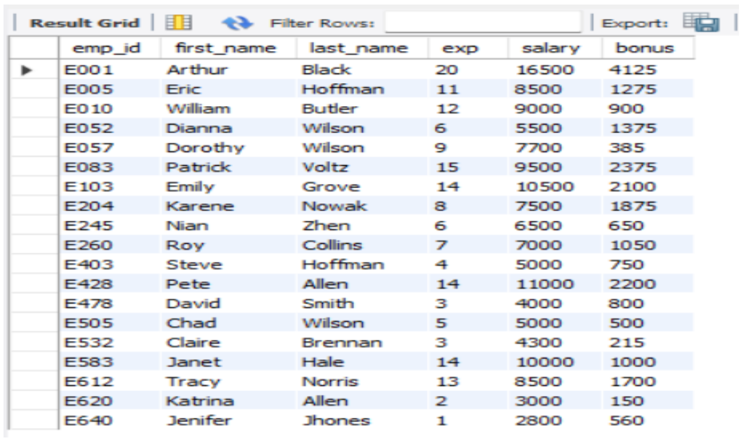
Task #16

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).

Query

```
SELECT emp_id, first_name, last_name, exp, salary,
        round(((5/100) * salary) * emp_rating) AS bonus
FROM  emp_record_table;
```

Screenshot of Output



emp_id	first_name	last_name	exp	salary	bonus
E001	Arthur	Black	20	16500	4125
E005	Eric	Hoffman	11	8500	1275
E010	William	Butler	12	9000	900
E052	Dianna	Wilson	6	5500	1375
E057	Dorothy	Wilson	9	7700	385
E083	Patrick	Voltz	15	9500	2375
E103	Emily	Grove	14	10500	2100
E204	Karene	Nowak	8	7500	1875
E245	Nian	Zhen	6	6500	650
E260	Roy	Collins	7	7000	1050
E403	Steve	Hoffman	4	5000	750
E428	Pete	Allen	14	11000	2200
E478	David	Smith	3	4000	800
E505	Chad	Wilson	5	5000	500
E532	Claire	Brennan	3	4300	215
E583	Janet	Hale	14	10000	1000
E612	Tracy	Norris	13	8500	1700
E620	Katrina	Allen	2	3000	150
E640	Jenifer	Jhones	1	2800	560

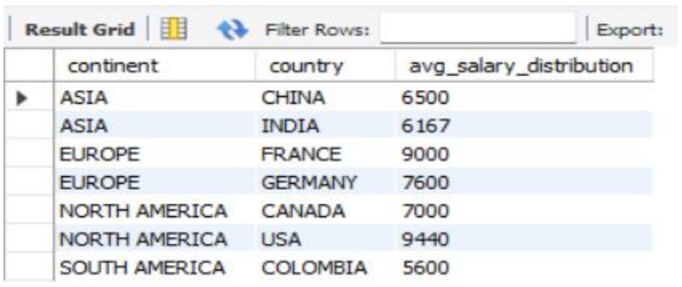
Task #17

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

Query

```
SELECT continent,ncountry,
            ROUND(AVG(salary)) avg_salary_distribution
FROM  emp_record_table
GROUP BY continent, country
ORDER BY continent, country;
```

Screenshot of Output



continent	country	avg_salary_distribution
ASIA	CHINA	6500
ASIA	INDIA	6167
EUROPE	FRANCE	9000
EUROPE	GERMANY	7600
NORTH AMERICA	CANADA	7000
NORTH AMERICA	USA	9440
SOUTH AMERICA	COLOMBIA	5600

Performance Optimization

Use of Indexing: Improved search performance for employee names.
Use of Views: Ensured filtered data retrieval without duplicating data.
Use of Stored Functions: Automated job title assignments based on experience.

Conclusion

This project helped in applying SQL techniques like joins, subqueries, indexing, views, and functions to analyze employee performance efficiently. The optimization techniques ensured better query performance.