**SCIENCEQTECH EMPLOYEE MAPPING PROJECT**

TASK 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.

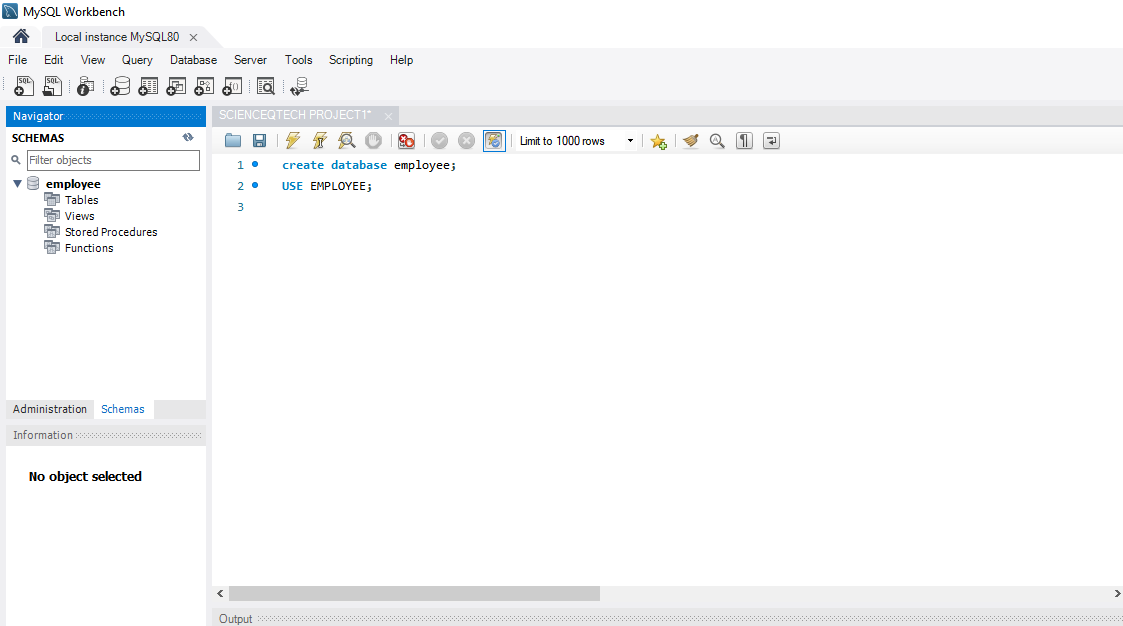
SOL

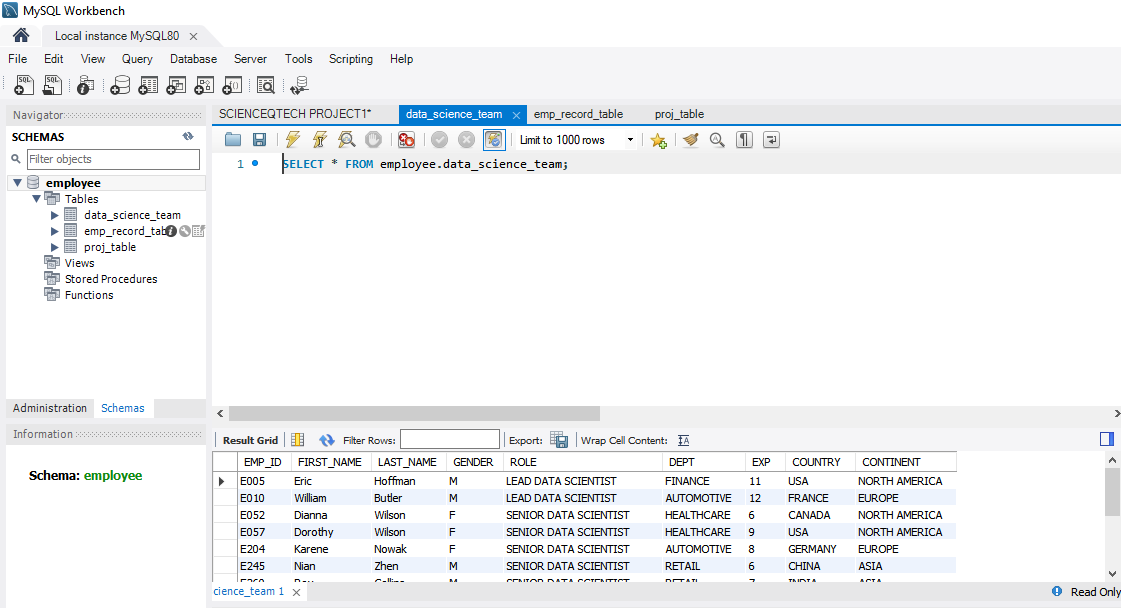
1. CREATED DATABASE EMPLOYEE USING QUERY

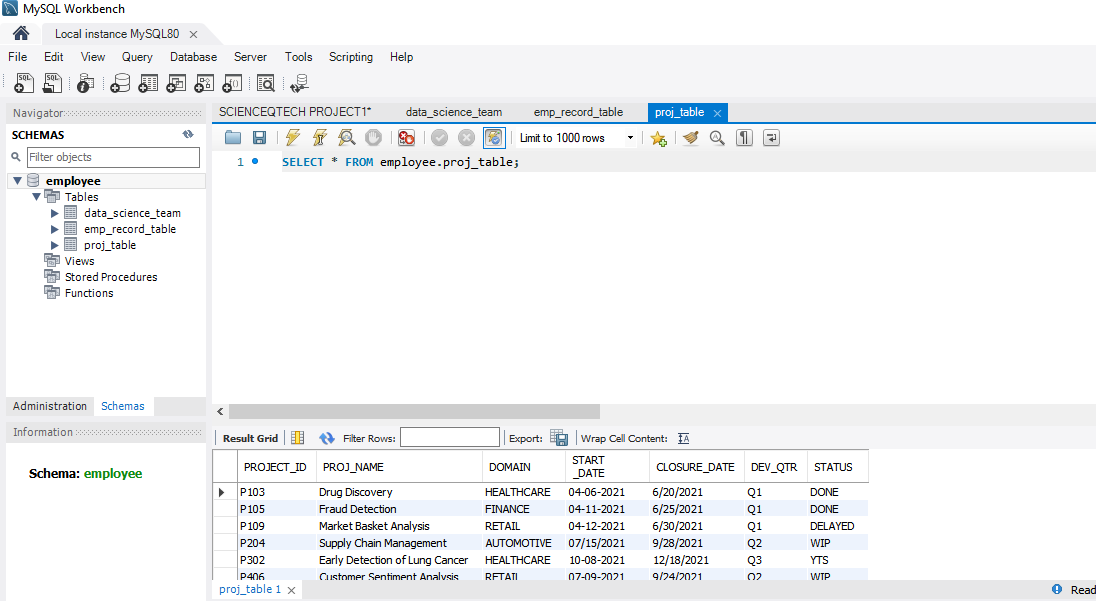
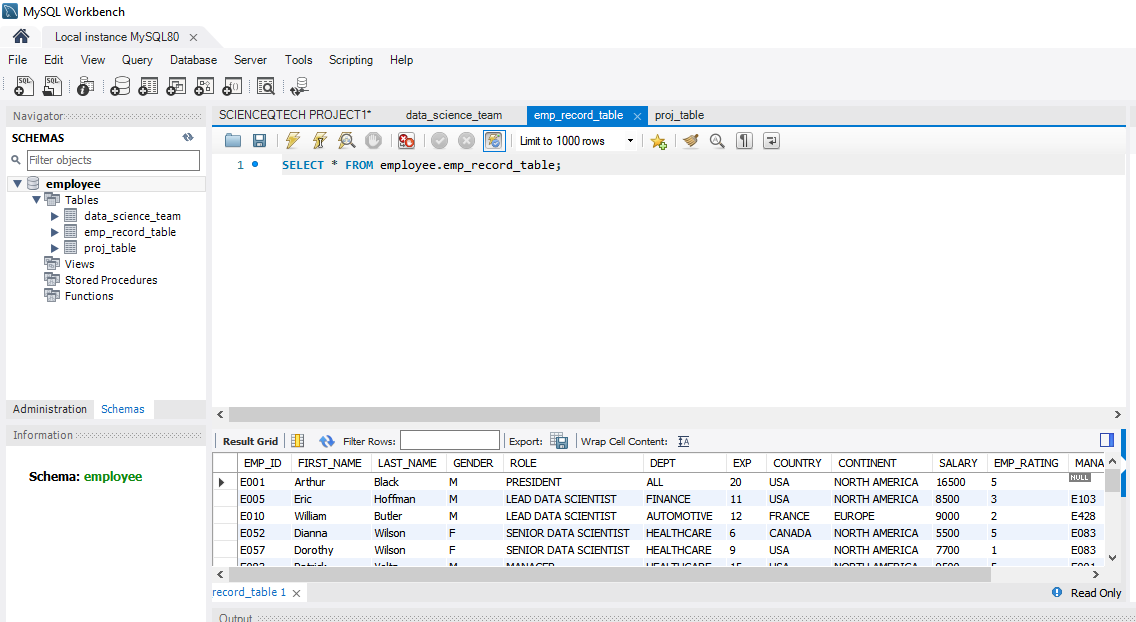
CREATE DATABASE EMPLOYEE;

IMPORTED TABLES FORM RESOURCES.

SCREENSHOTS

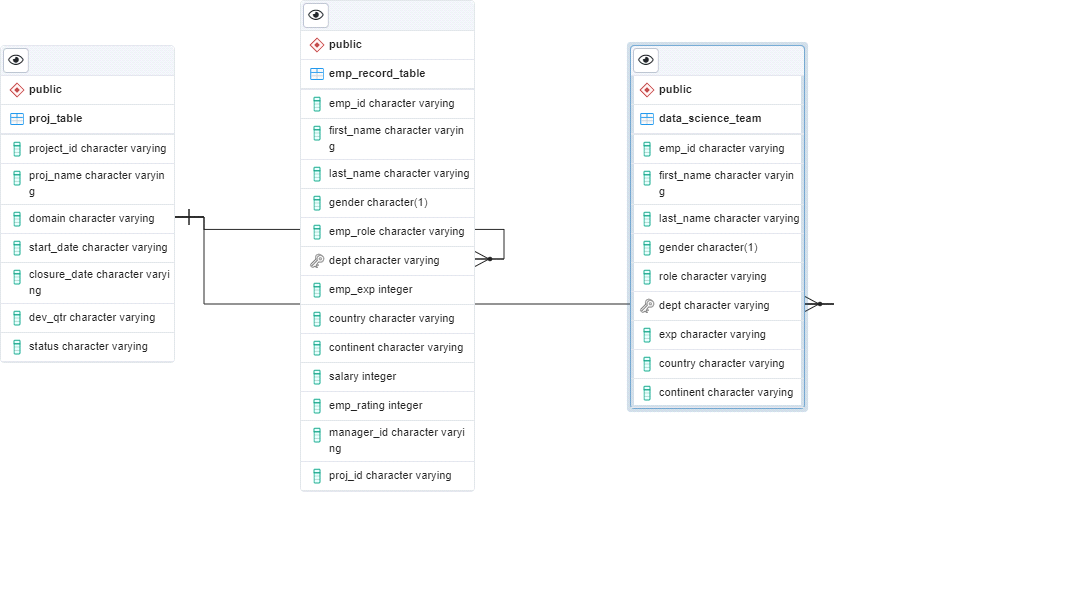






TASK 2

1. Create an ER diagram for the given **employee**database.



**TASK 3**

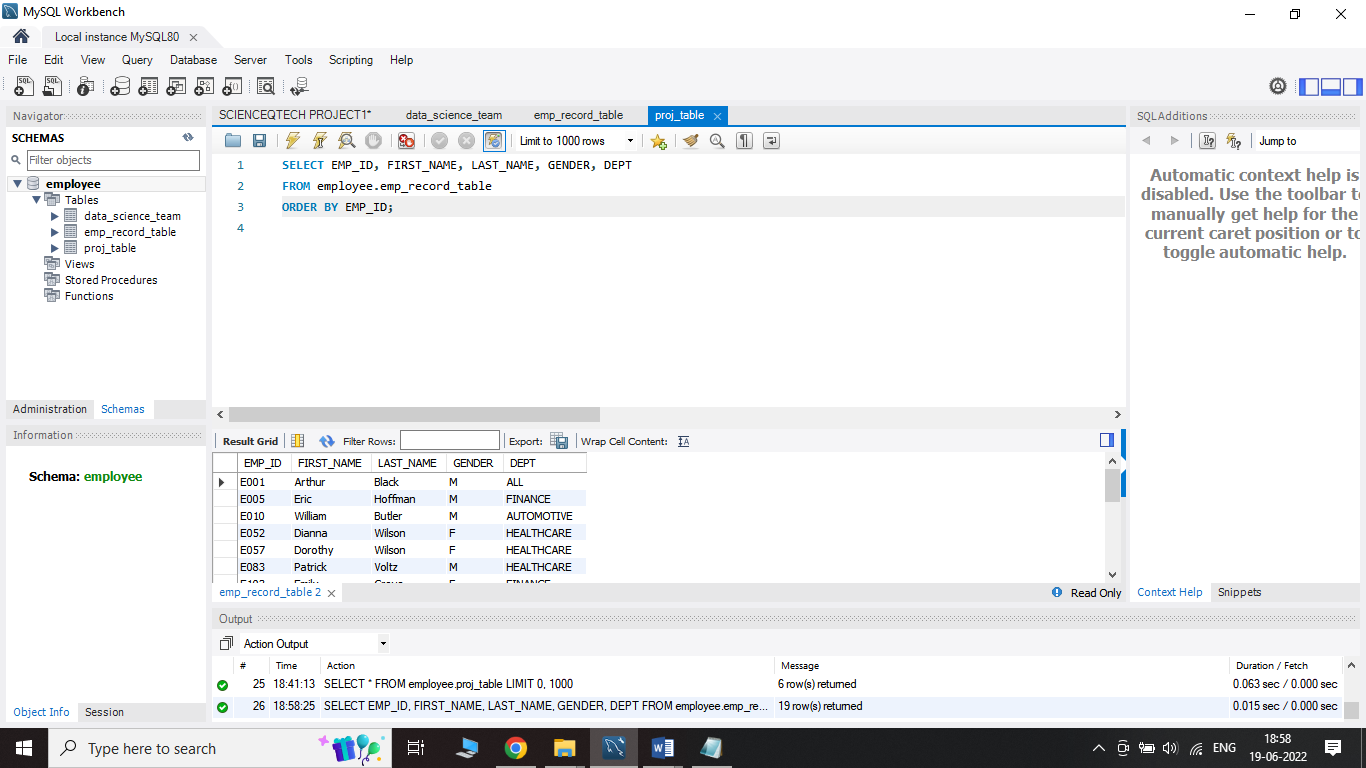
1. Write a query to fetch EMP\_ID, FIRST\_NAME, LAST\_NAME, GENDER, and DEPARTMENT from the employee record table, and make a list of employees and details of their department.

**QUERY**

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

FROM employee.emp\_record\_table

ORDER BY EMP\_ID;



**TASK 4**

1. 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

**SOL**

**QUERY**

#EMPLOYEE RATING LESS THAN TWO

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

FROM employee.emp\_record\_table

WHERE EMP\_RATING< 2;

#EMPLOYEE RATING GREATER THAN FOUR

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

FROM employee.emp\_record\_table

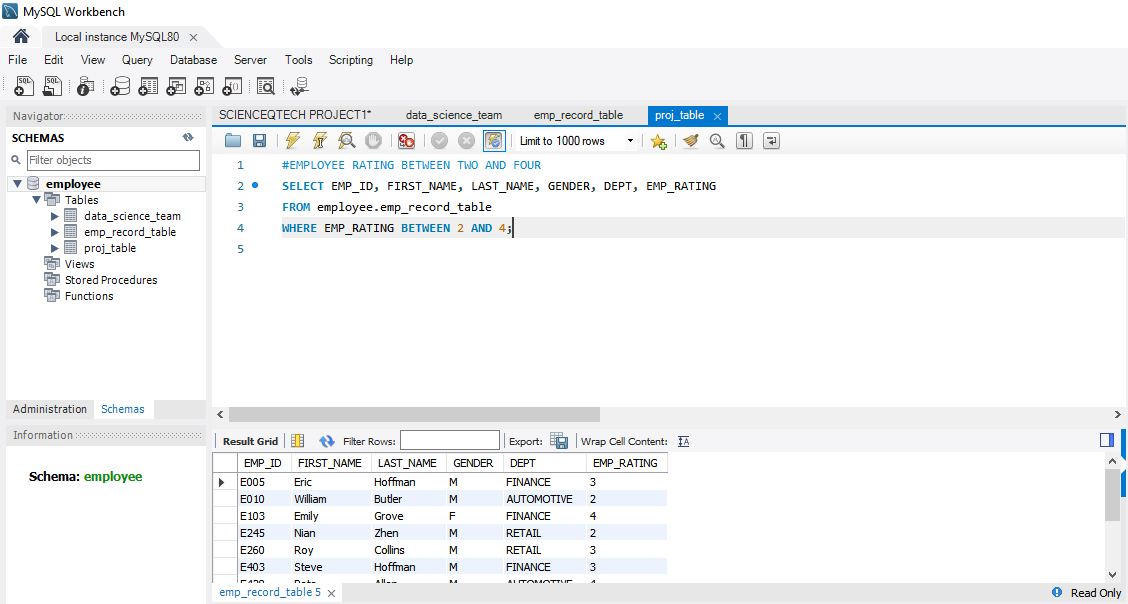
WHERE EMP\_RATING> 4;

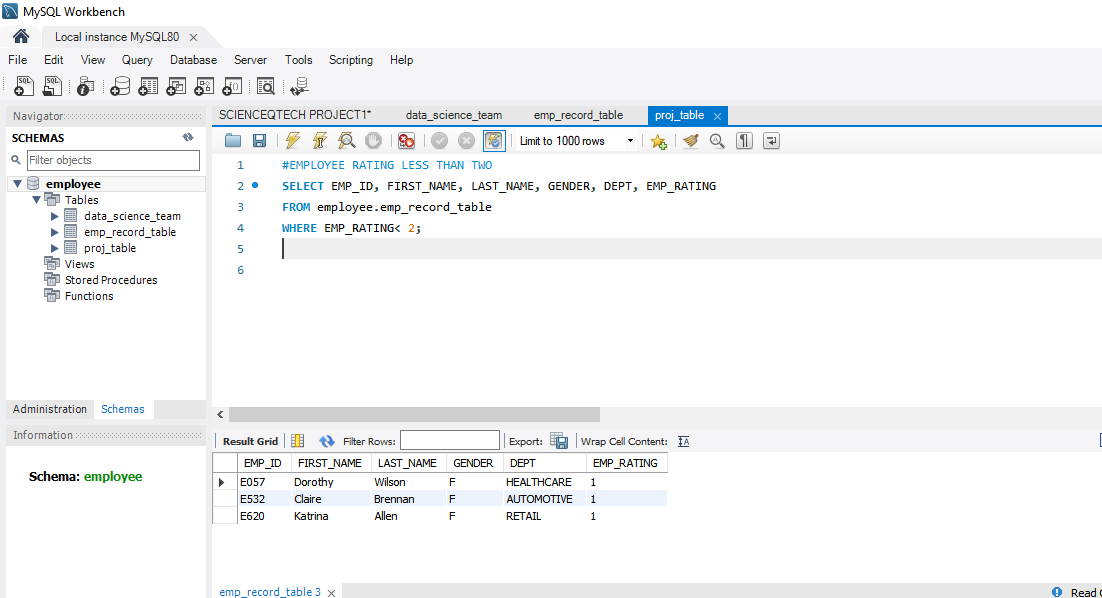
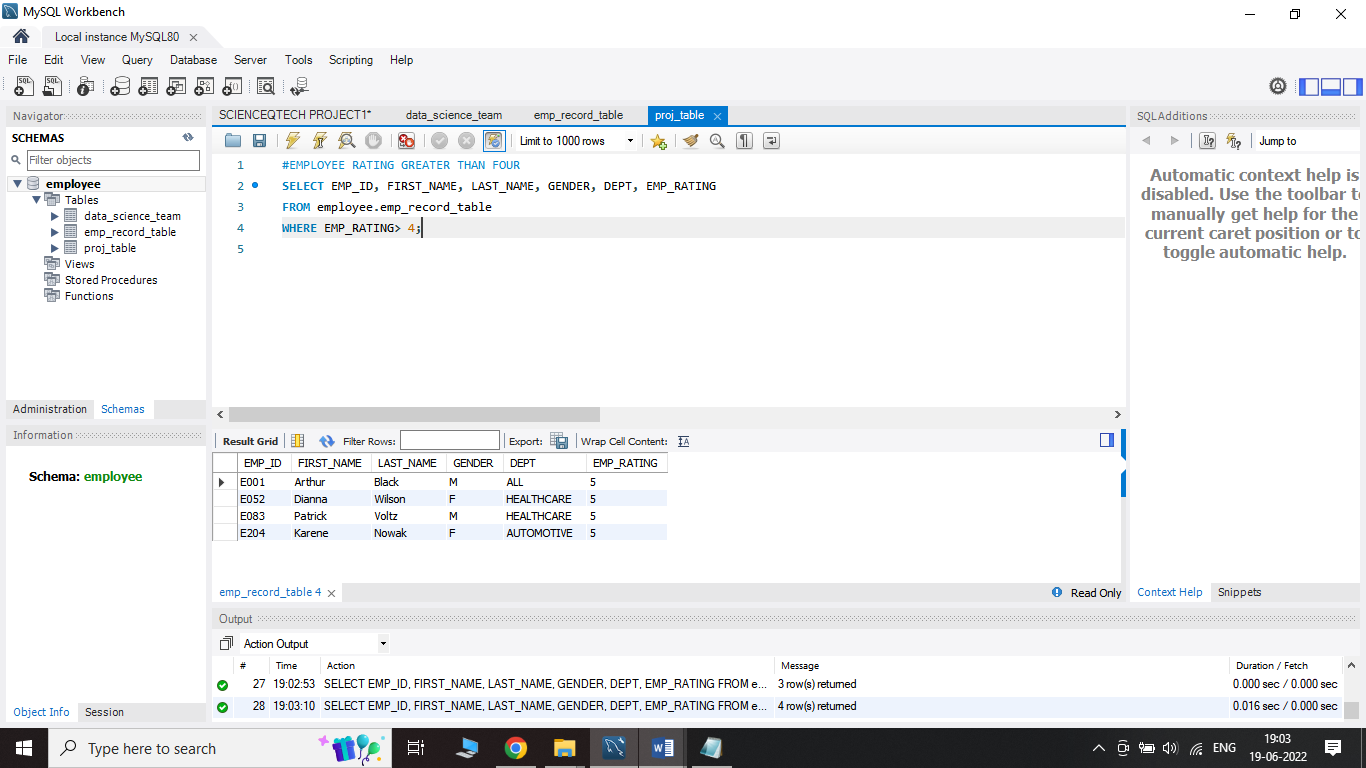
#EMPLOYEE RATING BETWEEN TWO AND FOUR

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

FROM employee.emp\_record\_table

WHERE EMP\_RATING BETWEEN 2 AND 4;

****

****

**TASK 5**

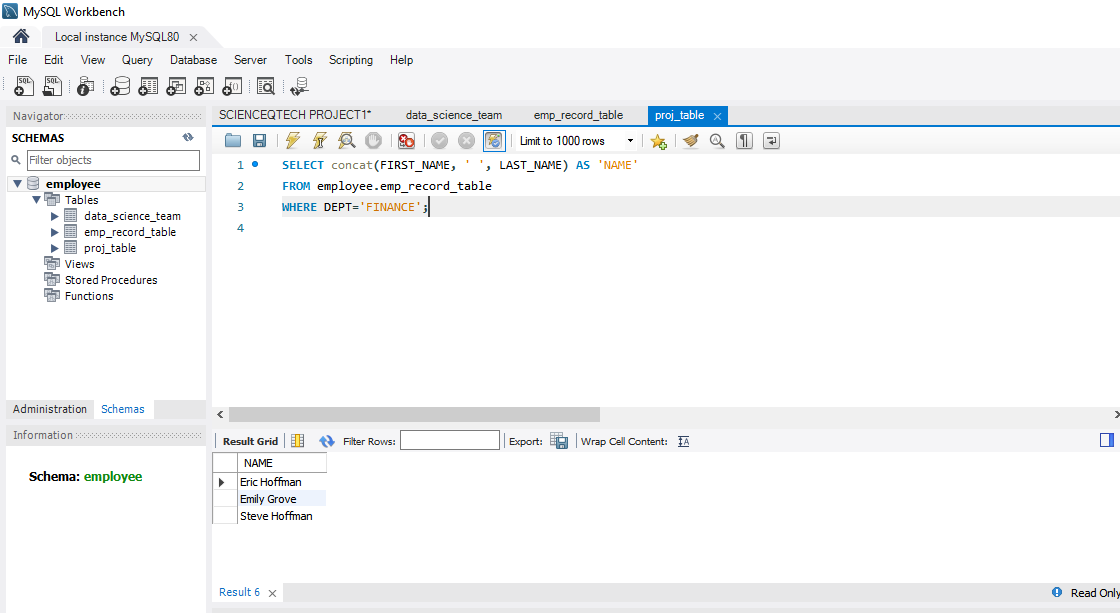
1. Write a query to concatenate the FIRST\_NAME and the LAST\_NAME of employees in the Finance department from the employee table and then give the resultant column alias as NAME.

**QUERY**

SELECT concat(FIRST\_NAME, ' ', LAST\_NAME) AS 'NAME'

FROM employee.emp\_record\_table

WHERE DEPT='FINANCE';

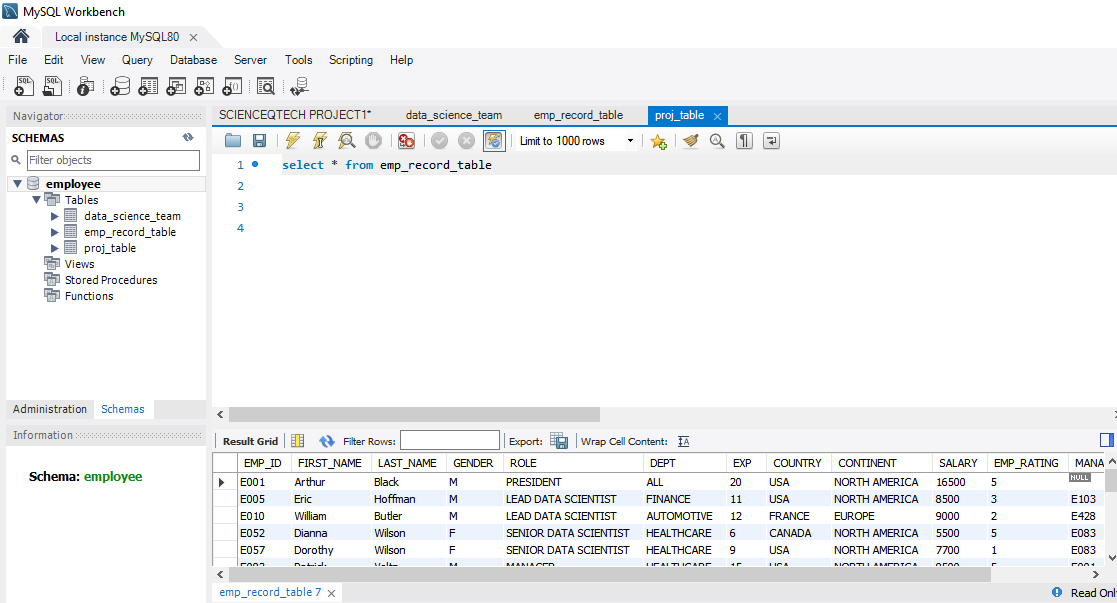


**TASK 6**

1. 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 \* from emp\_record\_table

****

**TASK 7**

1. 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 EMP\_ID, concat(FIRST\_NAME, ' ', LAST\_NAME) AS 'NAME'

FROM employee.emp\_record\_table

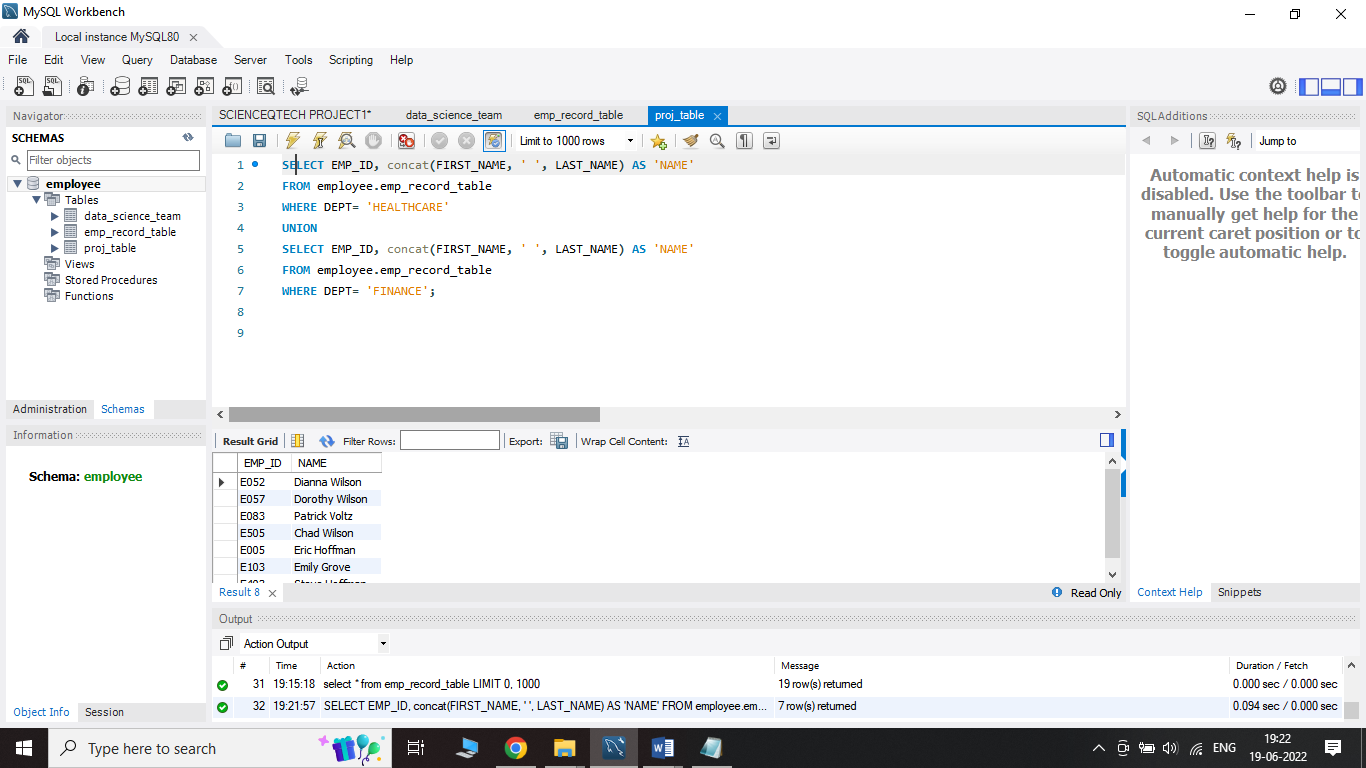
WHERE DEPT= 'HEALTHCARE'

UNION

SELECT EMP\_ID, concat(FIRST\_NAME, ' ', LAST\_NAME) AS 'NAME'

FROM employee.emp\_record\_table

WHERE DEPT= 'FINANCE';



**TASK 8**

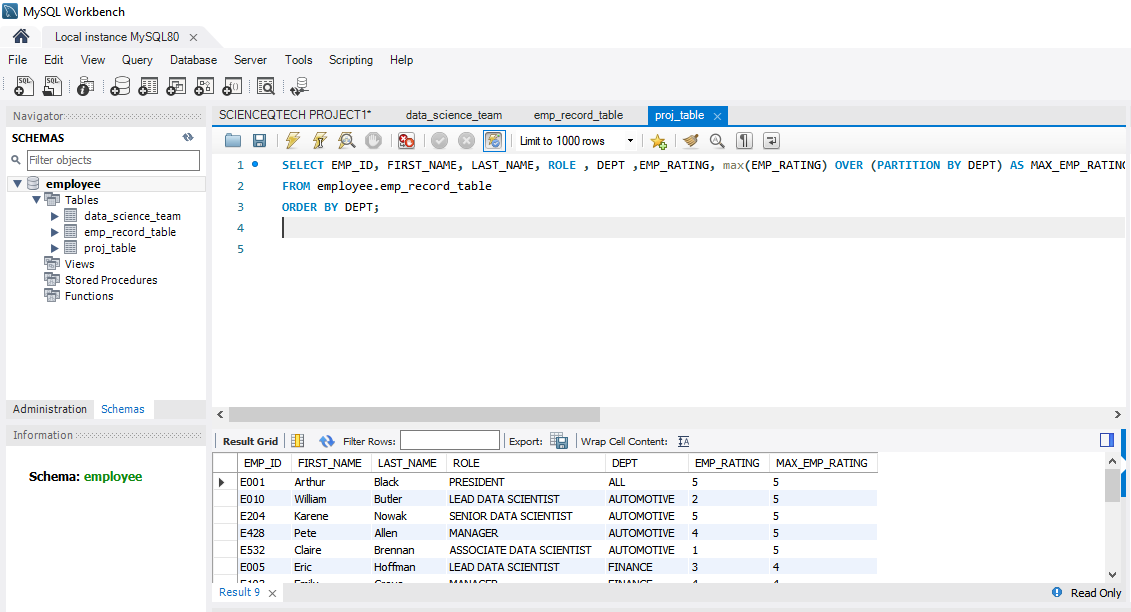
1. Write a query to list down employee details such as EMP\_ID, FIRST\_NAME, LAST\_NAME, ROLE, DEPARTMENT, and EMP\_RATING grouped by dept. Also include the respective employee rating along with the max emp rating for the department.

**QUERY**

SELECT EMP\_ID, FIRST\_NAME, LAST\_NAME, ROLE , DEPT ,EMP\_RATING, max(EMP\_RATING) OVER (PARTITION BY DEPT) AS MAX\_EMP\_RATING

FROM employee.emp\_record\_table

ORDER BY DEPT;



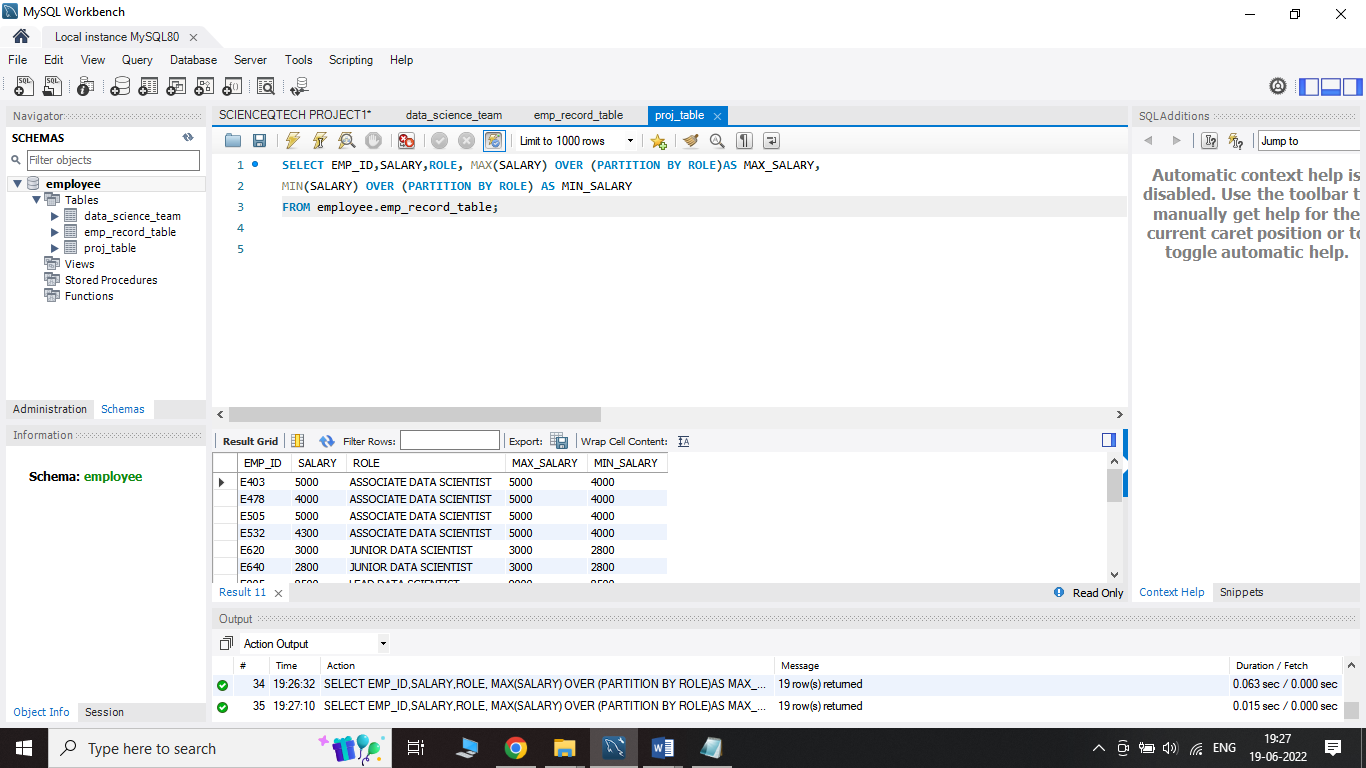
**TASK 9**

1. 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 EMP\_ID, SALARY, ROLE, MAX(SALARY) OVER (PARTITION BY ROLE) AS MAX\_SALARY, MIN(SALARY) OVER (PARTITION BY ROLE) AS MIN\_SALARY

FROM employee.emp\_record\_table;



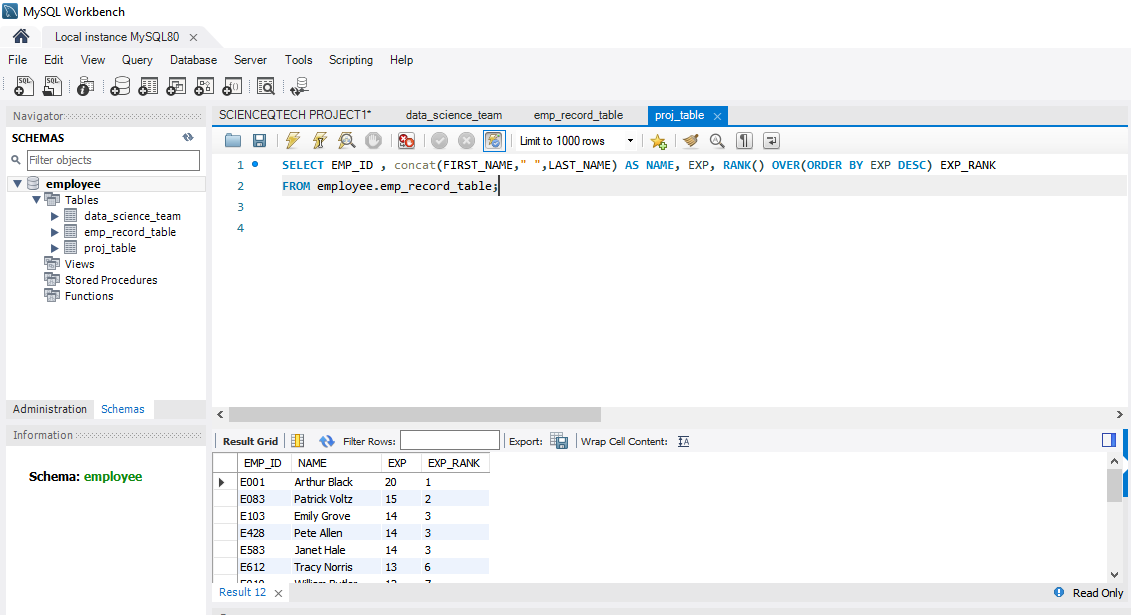
**TASK 10**

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

**QUERY**

SELECT EMP\_ID , concat(FIRST\_NAME," ",LAST\_NAME) AS NAME, EXP, RANK() OVER(ORDER BY EXP DESC) EXP\_RANK

FROM employee.emp\_record\_table;



**TASK 11**

1. 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**

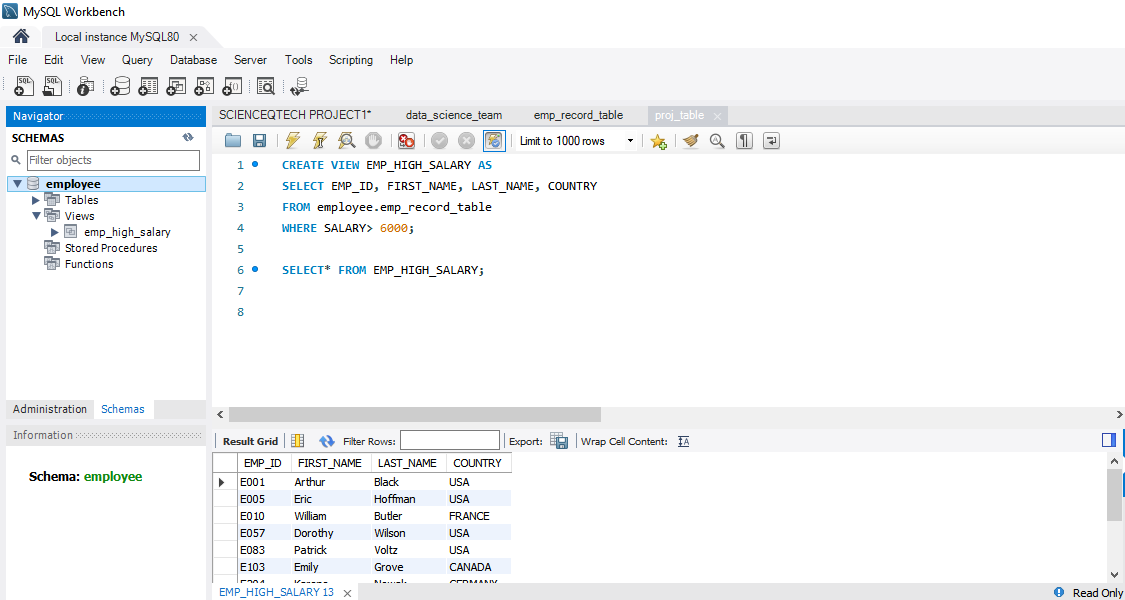
CREATE VIEW EMP\_HIGH\_SALARY AS

SELECT EMP\_ID, FIRST\_NAME, LAST\_NAME, COUNTRY

FROM employee.emp\_record\_table

WHERE SALARY> 6000;

SELECT\* FROM EMP\_HIGH\_SALARY;



**TASK 12**

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

**QUERY**

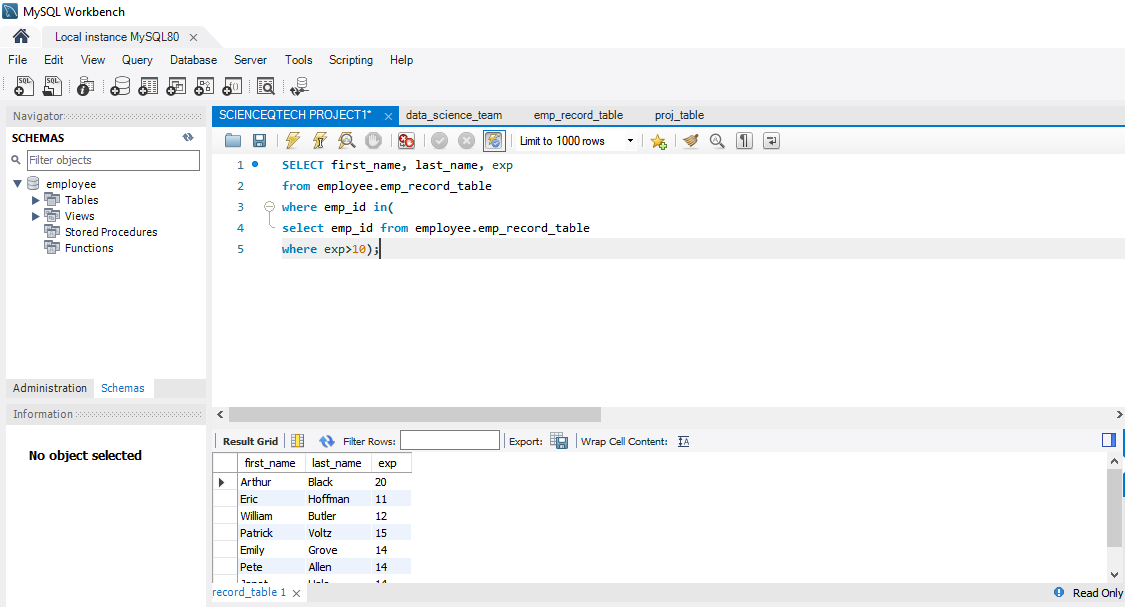
SELECT first\_name, last\_name, exp

from employee.emp\_record\_table

where emp\_id in(

select emp\_id from employee.emp\_record\_table

where exp>10);



**TASK 13**

1. 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 **&&**

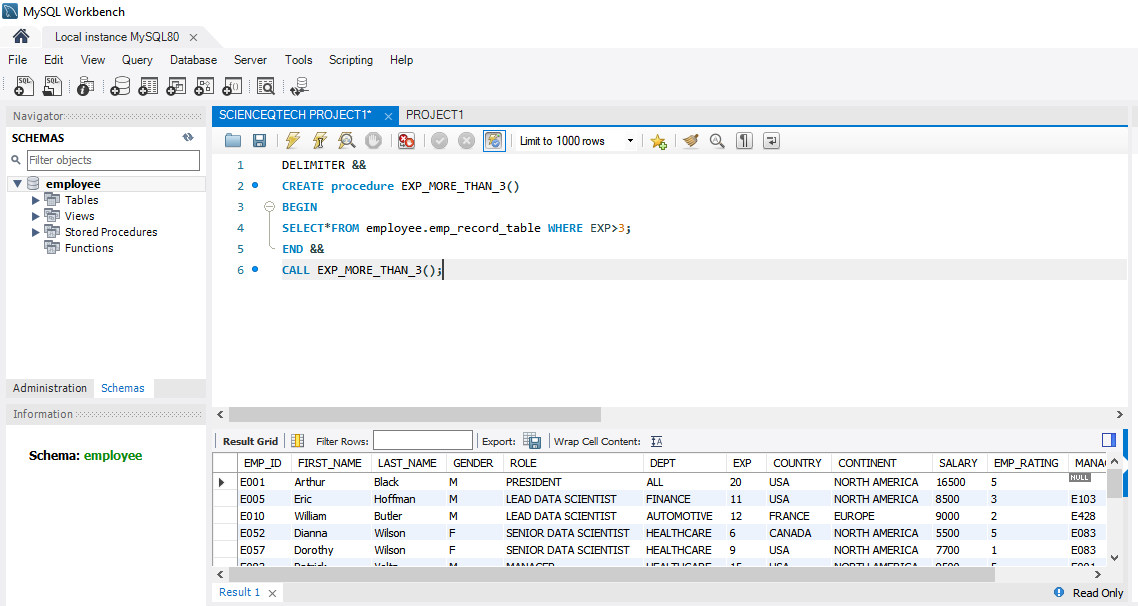
**CREATE procedure EXP\_MORE\_THAN\_3()**

**BEGIN**

**SELECT\*FROM employee.emp\_record\_table WHERE EXP>3;**

**END &&**

**CALL EXP\_MORE\_THAN\_3();**

****

**TASK 14**

1. 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'.

**QUERY**

DELIMITER \\

CREATE FUNCTION DESIGNATION(EXP INT)

RETURNS VARCHAR(100)

deterministic

BEGIN

declare DESIGNATION VARCHAR(100);

IF EXP <= 2 THEN SET DESIGNATION = 'JUINOR DATA SCIENTIST';

ELSEIF EXP <= 5 THEN SET DESIGNATION ='ASSOCIATE DATA SCIENTIST';

ELSEIF EXP <= 10 THEN SET DESIGNATION=' SENIOR DTA SCIENTIST';

ELSEIF EXP <=12 THEN SET DESIGNATION ='LEAD DATA SCIENTIST';

ELSEIF EXP>12 THEN SET DESIGNATION ='MANAGER';

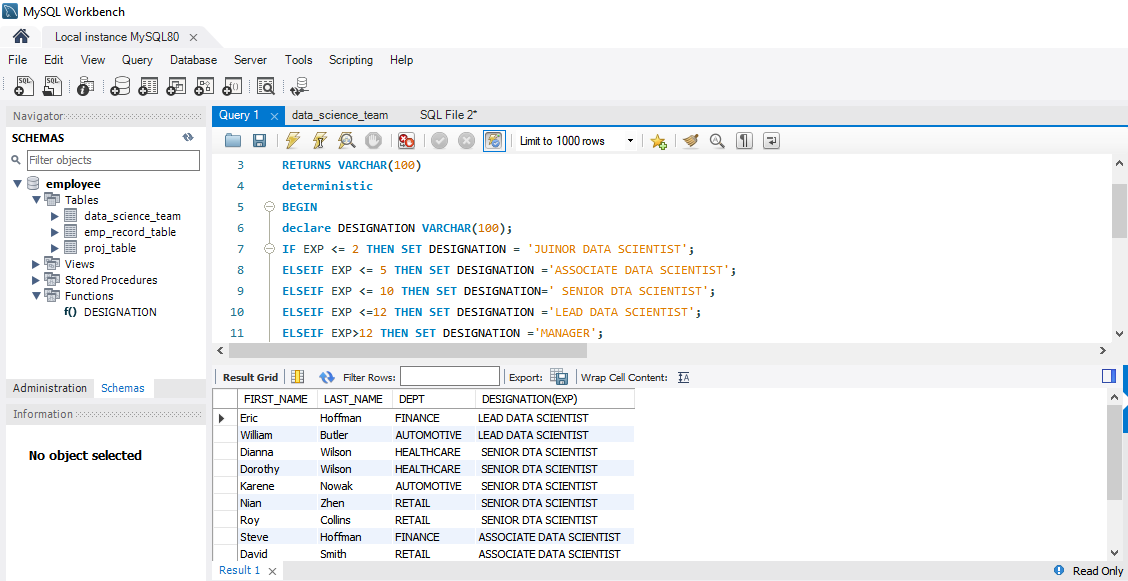
ELSE SET DESIGNATION= 'INVALID';

END IF;

RETURN (DESIGNATION);

END \\

SELECT FIRST\_NAME, LAST\_NAME , DEPT, DESIGNATION(EXP) FROM EMPLOYEE.DATA\_SCIENCE\_TEAM;



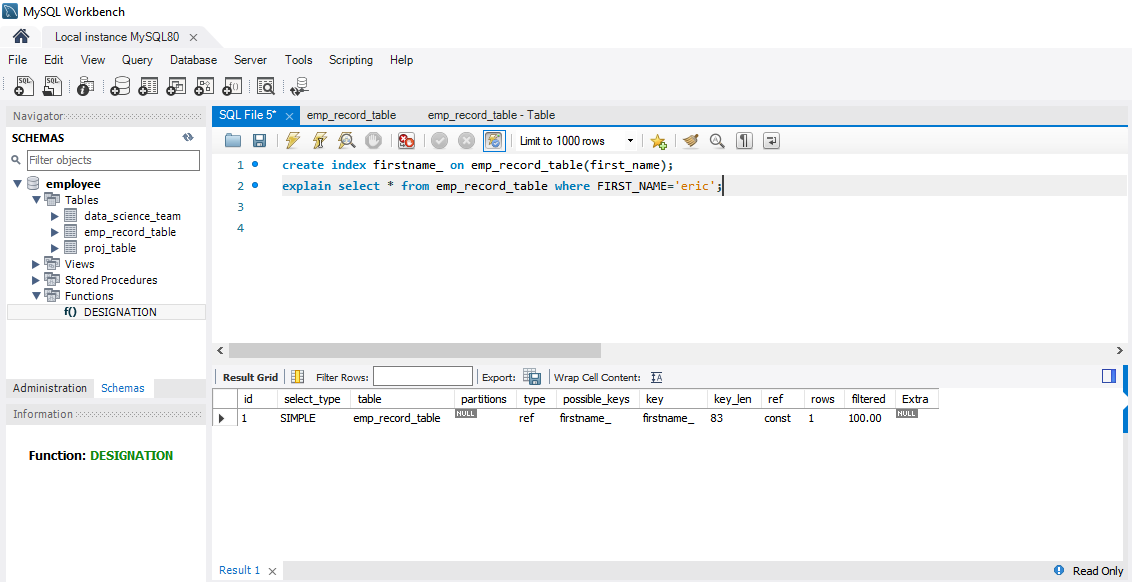
**TASK 15**

1. 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**

create index firstname\_ on emp\_record\_table(first\_name);

explain select \* from emp\_record\_table where FIRST\_NAME='eric';

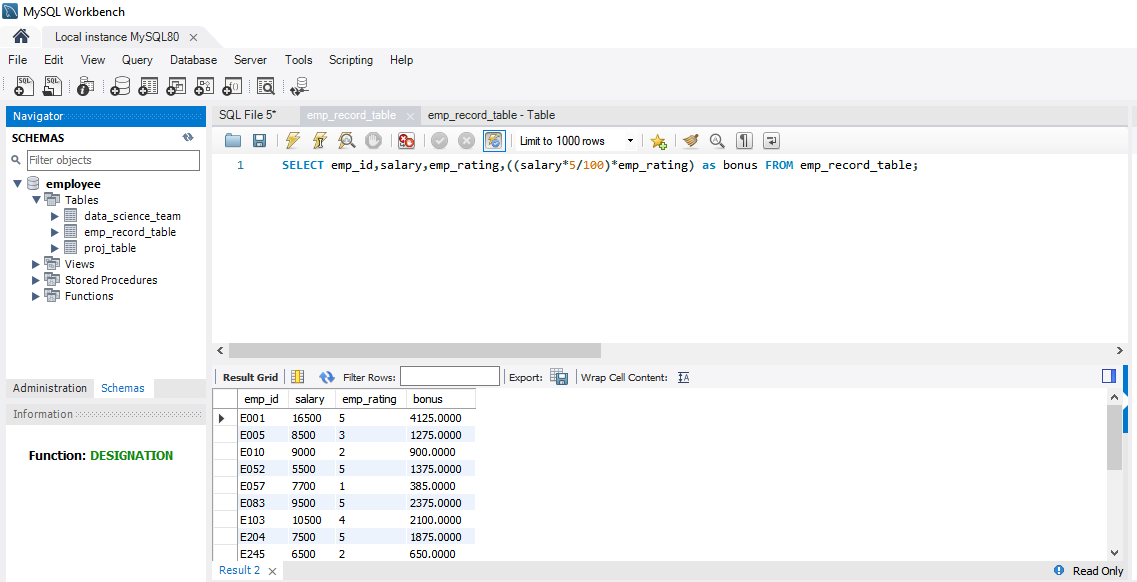


**TASK 16**

1. 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,salary,emp\_rating,((salary\*5/100)\*emp\_rating) as bonus FROM emp\_record\_table;



**TASK 17**

1. 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, avg(Salary) FROM emp\_record\_table group by continent;

SELECT country, avg(Salary) FROM emp\_record\_table group by country;

select continent,

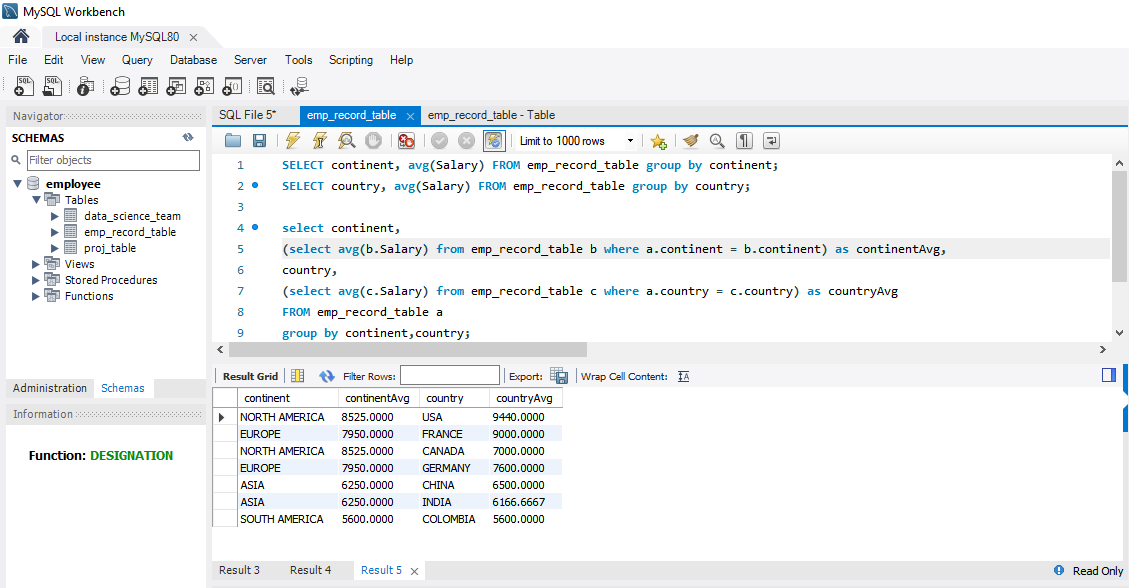
(select avg(b.Salary) from emp\_record\_table b where a.continent = b.continent) as continentAvg,

country,

(select avg(c.Salary) from emp\_record\_table c where a.country = c.country) as countryAvg

FROM emp\_record\_table a

group by continent,country;



END OF PROJECT