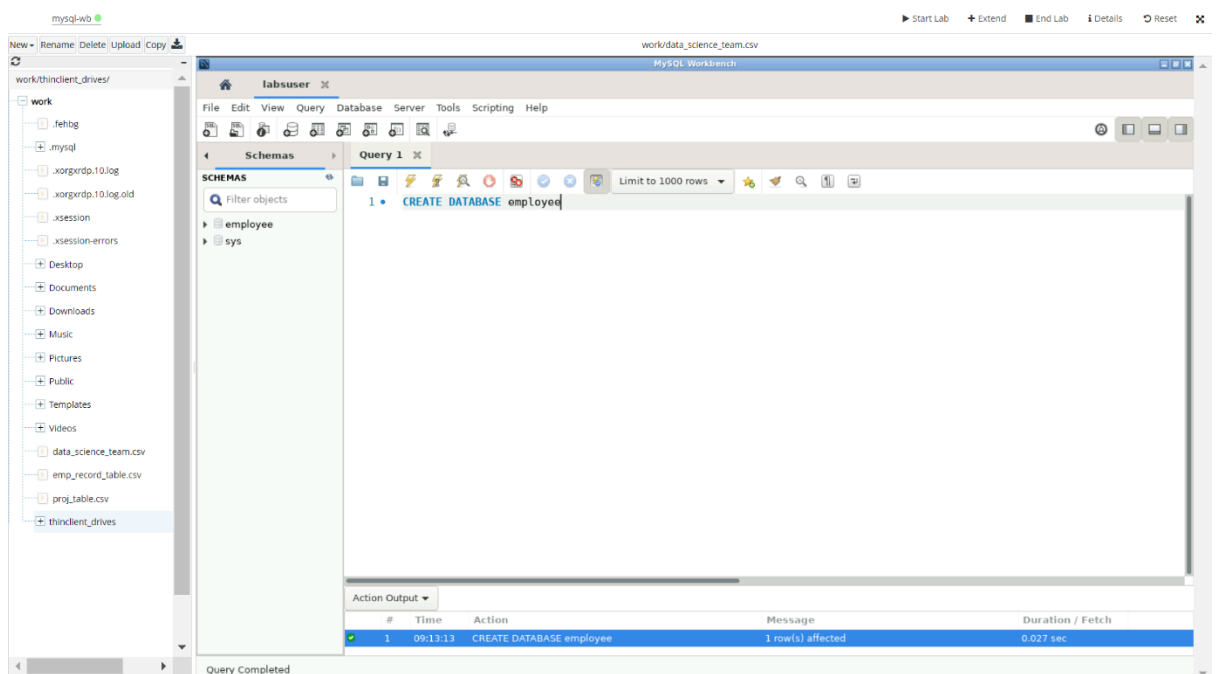
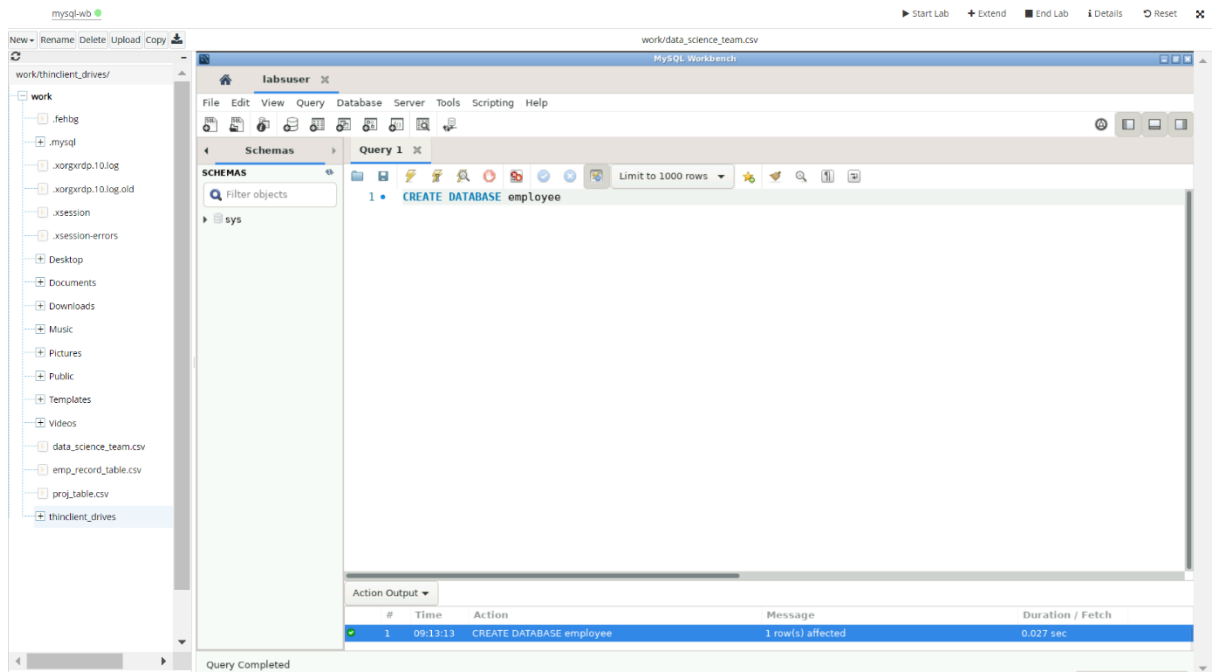
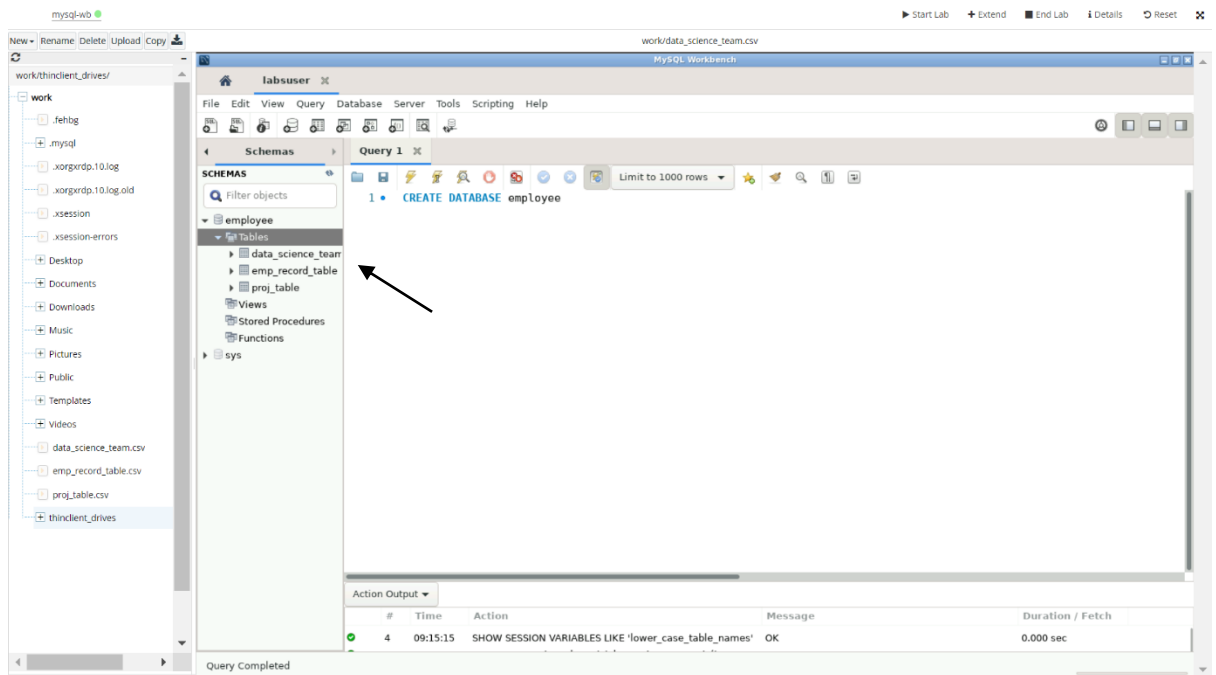


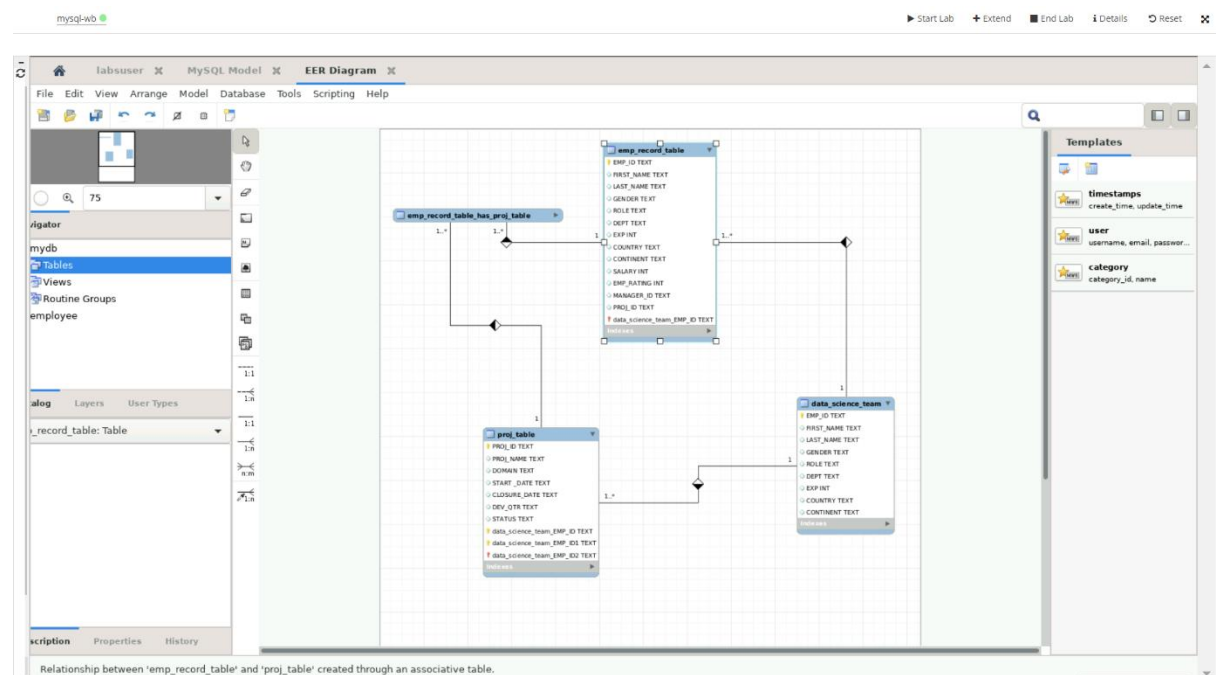
The task to be performed:

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.





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



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.

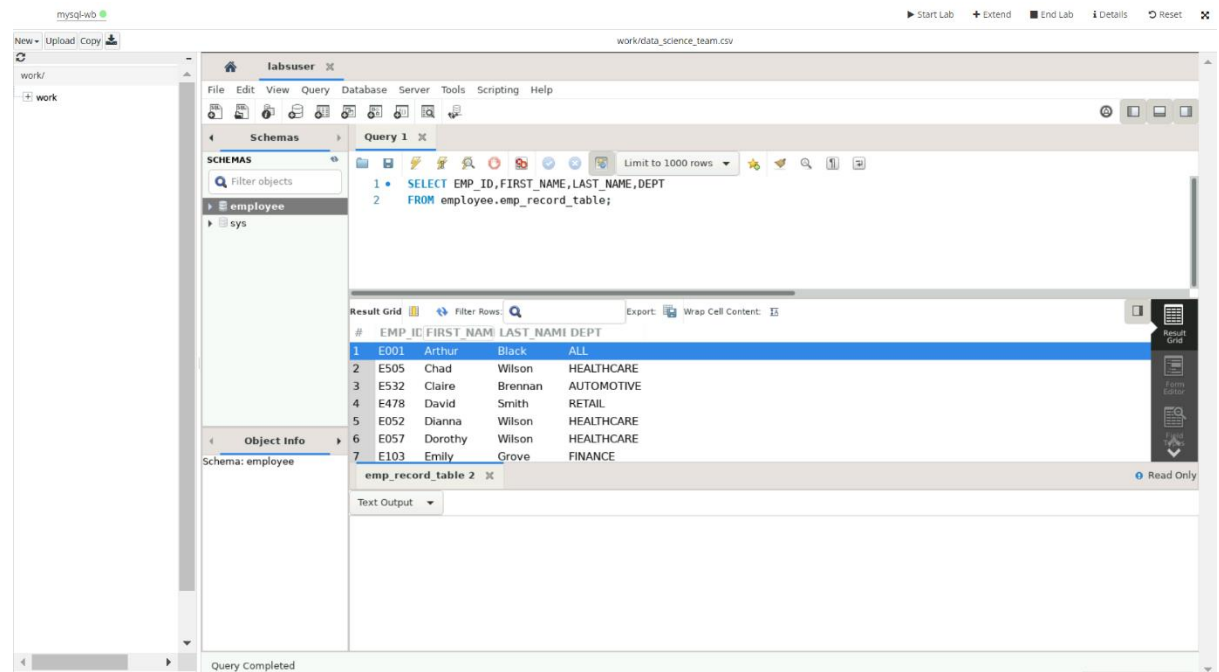
Query to fetch details from employee record table:

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT
```

FROM employee.emp_record_table;

List of employees and details of their department:

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, DEPT
FROM employee.emp_record_table;
```

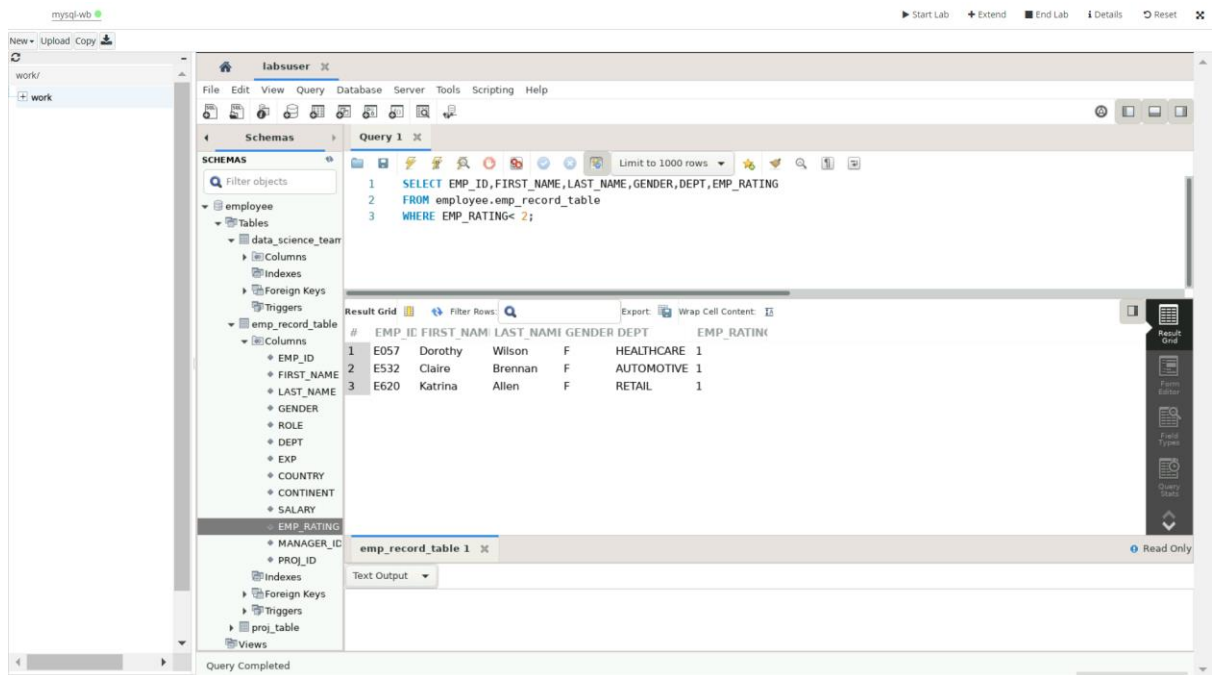


4. Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPARTMENT, and EMP_RATING if the EMP_RATING is:

- **less than two**

Query:

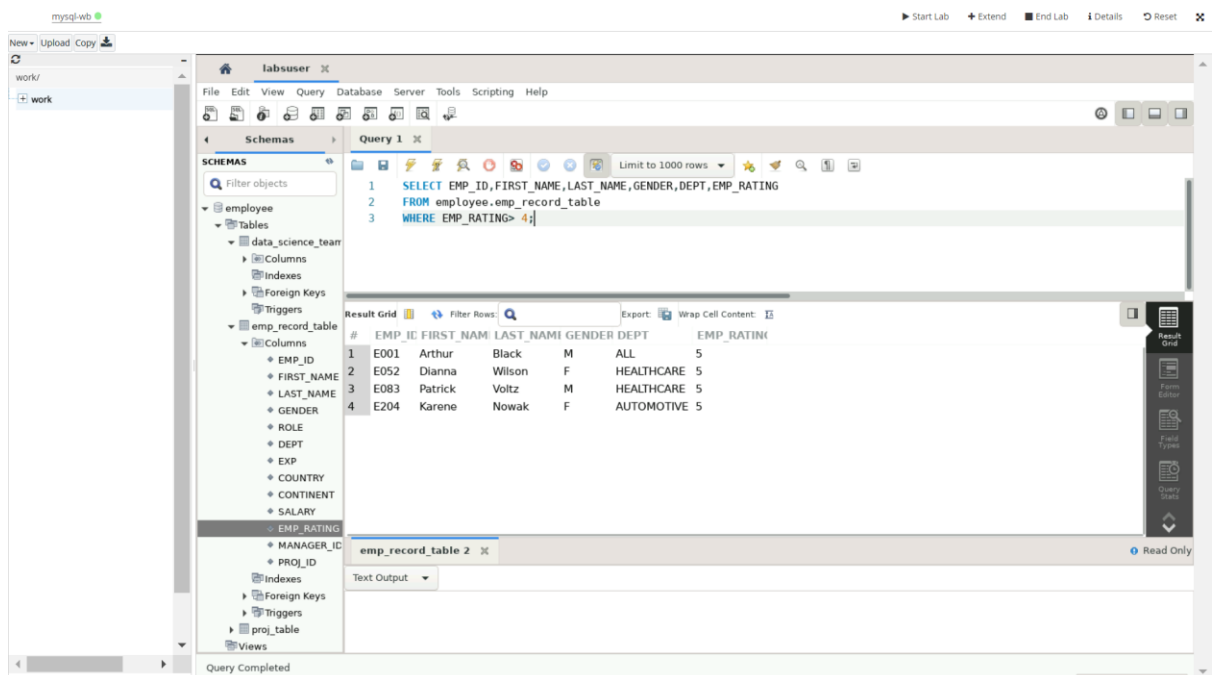
```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
FROM employee.emp_record_table
WHERE EMP_RATING < 2;
```



- **greater than four**

Query:

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
FROM employee.emp_record_table
WHERE EMP_RATING > 4;
```



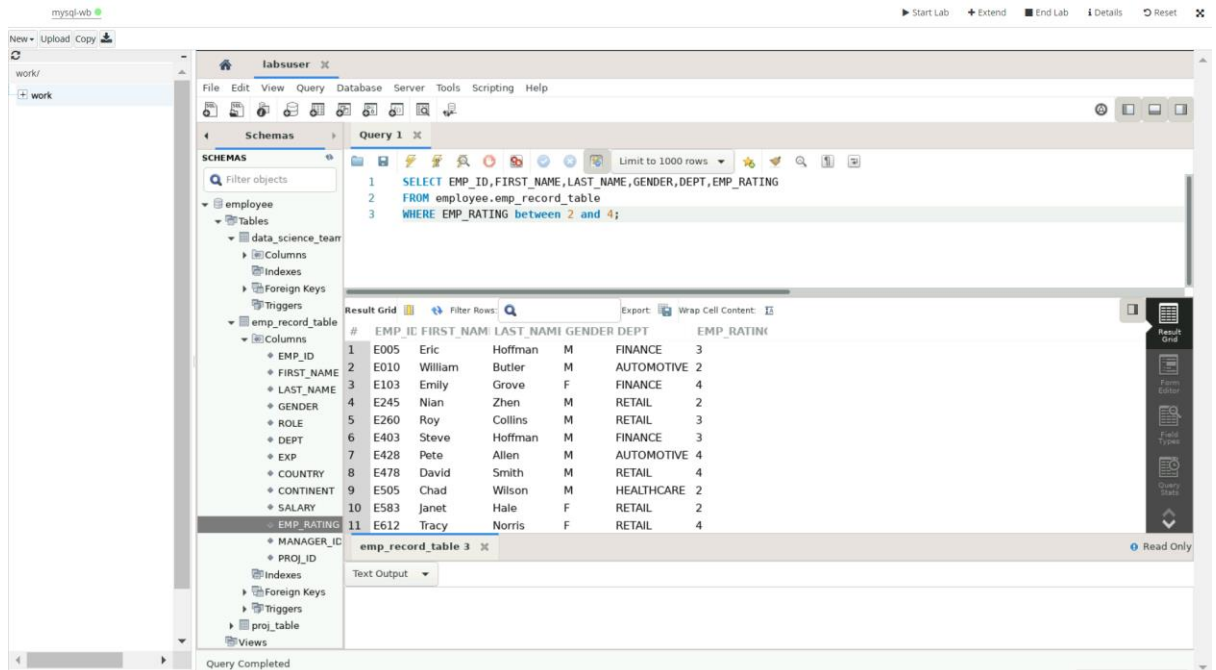
- **between two and four**

Query:

```

SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
FROM employee.emp_record_table
WHERE EMP_RATING BETWEEN 2 AND 4;

```



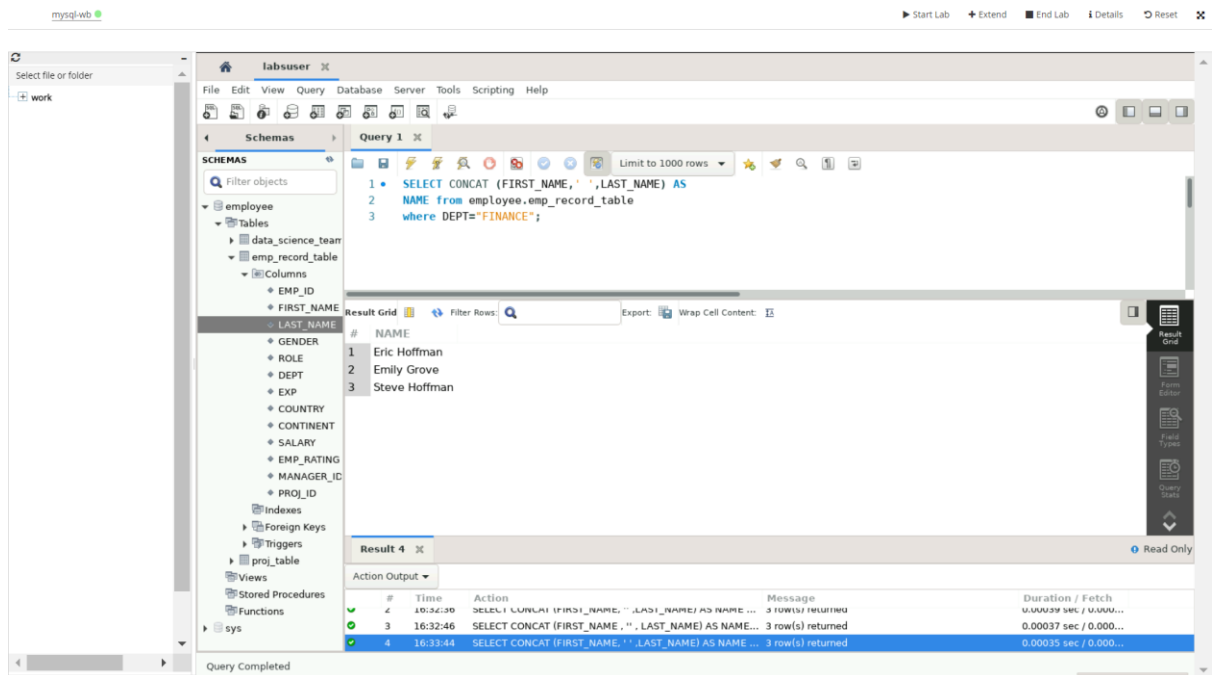
5. Write a query to concatenate the **FIRST_NAME** and the **LAST_NAME** of employees in the Finance department from the employee table and then give the resultant column alias as **NAME**.

QUERY:

```

SELECT CONCAT (FIRST_NAME, ' ', LAST_NAME) AS
NAME FROM employee.emp_record_table
WHERE DEPT=" FINANCE";

```



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 MANAGER_ID, COUNT(EMP_ID)
```

```
FROM employee.emp_record_table
```

```
GROUP BY MANAGER_ID
```

```
ORDER BY COUNT(EMP_ID);
```

mysql-wb

Start Lab + Extend End Lab Details Reset

labuser

File Edit View Query Database Server Tools Scripting Help

Schemas

Filter objects

employee

data_science_team

emp_record_table

Columns

EMP_ID

FIRST_NAME

LAST_NAME

GENDER

ROLE

DEPT

EXP

COUNTRY

CONTINENT

SALARY

EMP_RATING

MANAGER_ID

PROJ_ID

Indexes

Foreign Keys

Triggers

proj_table

Views

Stored Procedures

Functions

sys

Query 1

1 SELECT MANAGER_ID, COUNT(EMP_ID)

2 FROM employee.emp_record_table

3 GROUP BY MANAGER_ID

4 ORDER BY COUNT(EMP_ID);

Limit to 1000 rows

Result Grid

Filter Rows

Export: Wrap Cell Content

#	MANAGER_ID	COUNT(EMP_ID)
1	E000	1
2	E103	2
3	E612	2
4	E428	3
5	E083	3
6	E583	3
7	E001	5

Result 10

Action Output

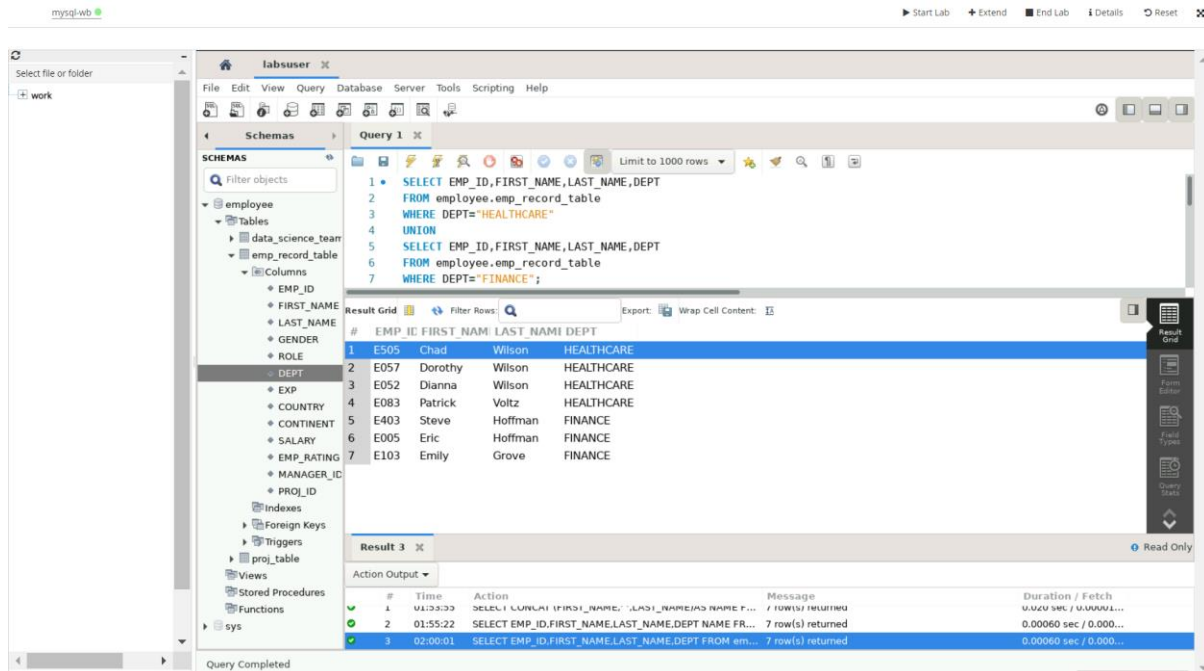
#	Time	Action	Message	Duration / Fetch
13	17:27:34	SELECT MANAGER_ID, FIRST_NAME, LAST_NAME, COUNTRY...	19 row(s) returned	0.00048 sec / 0.000...
14	17:28:17	SELECT MANAGER_ID, FIRST_NAME, LAST_NAME, COUNTRY...	19 row(s) returned	0.00048 sec / 0.000...
15	17:28:45	SELECT MANAGER_ID, COUNT(EMP_ID) FROM employee.e...	7 row(s) returned	0.00040 sec / 0.000...

Query Completed

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 EMP_ID, FIRST_NAME, LAST_NAME, DEPT
FROM employee.emp_record_table
WHERE DEPT="HEALTHCARE"
UNION
SELECT EMP_ID, FIRST_NAME, LAST_NAME, DEPT
FROM employee.emp_record_table
WHERE DEPT="FINANCE";
```



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

```
FROM employee.emp_record_table
```

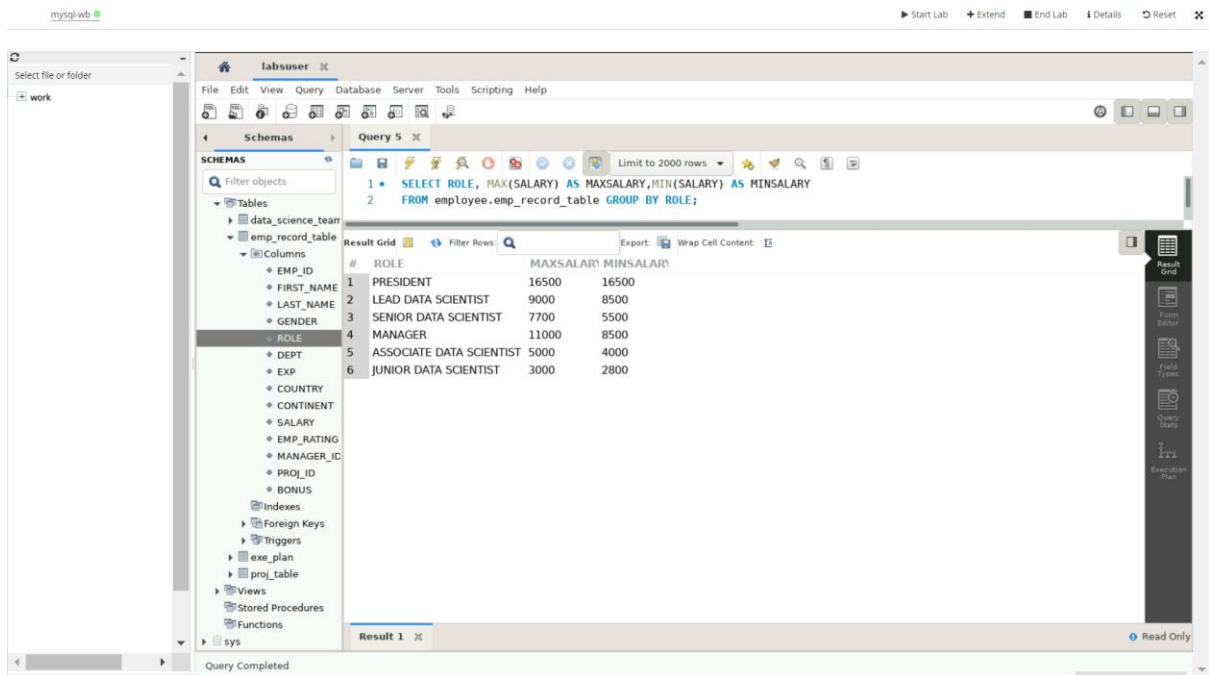
```
WHERE (EMP_RATING,DEPT) IN (SELECT MAX(EMP_RATING),DEPT FROM
employee.emp_record_table GROUP BY DEPT);
```

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

QUERY:

```
SELECT ROLE, MAX(SALARY) AS MAXSALARY, MIN(SALARY) AS MINSALARY
```

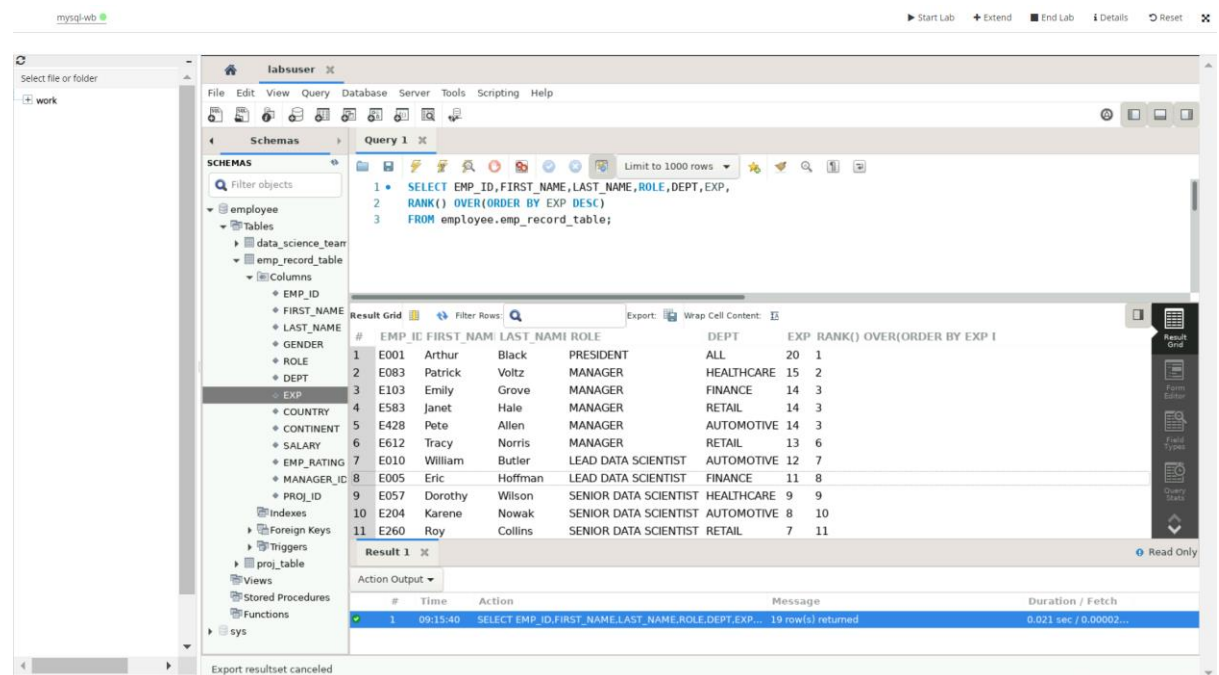
```
FROM employee.emp_record_table GROUP BY ROLE;
```

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, DEPT, EXP,
RANK () OVER (ORDER BY EXP DESC)
FROM employee.emp_record_table;
```



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.

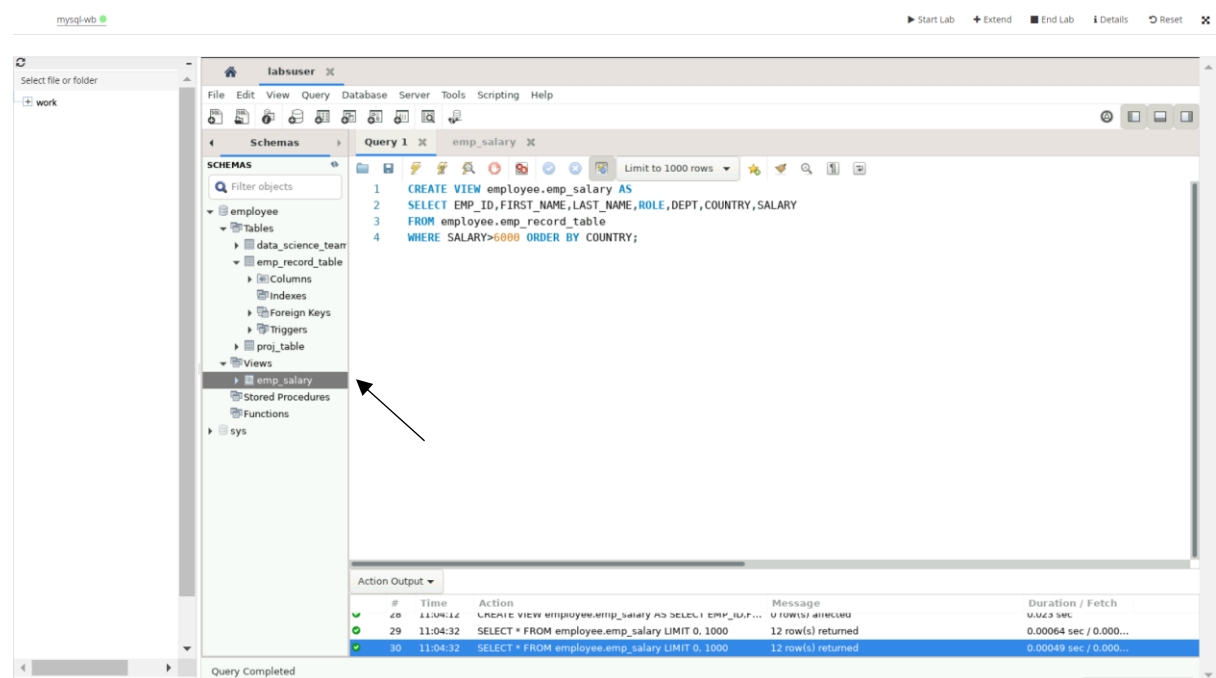
QUERY- To create a view

```
CREATE VIEW employee.emp_salary AS
```

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, COUNTRY, SALARY
```

```
FROM employee.emp_record_table
```

```
WHERE SALARY > 6000 ORDER BY COUNTRY;
```



QUERY- To see the view contents:

```
SELECT * FROM employee.emp_salary;
```

The screenshot shows the MySQL Workbench interface with the 'emp_salary' table selected in the Schemas pane. The query editor contains the statement: `SELECT * FROM employee.emp_salary;`. The Result Grid displays 12 rows of data with columns: #, EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, COUNTRY, and SALARY. The Action Output pane at the bottom shows the execution details for the query.

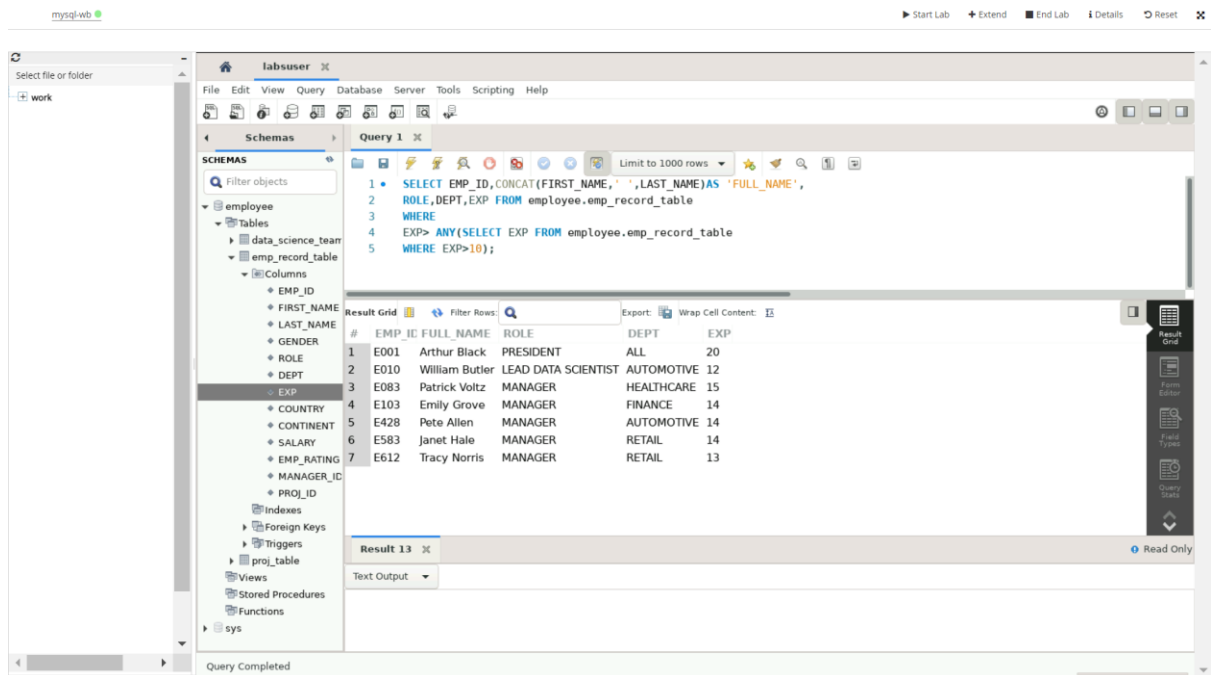
#	EMP_ID	FIRST_NAME	LAST_NAME	ROLE	DEPT	COUNTRY	SALARY
1	E103	Emily	Grove	MANAGER	FINANCE	CANADA	10500
2	E245	Nian	Zhen	SENIOR DATA SCIENTIST	RETAIL	CHINA	6500
3	E583	Janet	Hale	MANAGER	RETAIL	COLOMBIA	10000
4	E010	William	Butler	LEAD DATA SCIENTIST	AUTOMOTIVE	FRANCE	9000
5	E204	Karene	Nowak	SENIOR DATA SCIENTIST	AUTOMOTIVE	GERMANY	7500
6	E428	Pete	Allen	MANAGER	AUTOMOTIVE	GERMANY	11000
7	E260	Roy	Collins	SENIOR DATA SCIENTIST	RETAIL	INDIA	7000
8	E612	Tracy	Norris	MANAGER	RETAIL	INDIA	8500
9	E001	Arthur	Black	PRESIDENT	ALL	USA	16500
10	E005	Eric	Hoffman	LEAD DATA SCIENTIST	FINANCE	USA	8500
11	E057	Dorothy	Wilson	SENIOR DATA SCIENTIST	HEALTHCARE	USA	7700

#	Time	Action	Message	Duration / Fetch
28	11:04:12	CREATE VIEW employee.emp_salary AS SELECT EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, EXP FROM employee.emp_record_table	VIEW CREATED SUCCESSFULLY	0.00000 sec / 0.000...
29	11:04:32	SELECT * FROM employee.emp_salary LIMIT 0, 1000	12 row(s) returned	0.00064 sec / 0.000...
30	11:04:32	SELECT * FROM employee.emp_salary LIMIT 0, 1000	12 row(s) returned	0.00049 sec / 0.000...

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, CONCAT (FIRST_NAME, ' ', LAST_NAME) AS 'FULL_NAME',
ROLE, DEPT, EXP FROM employee.emp_record_table
WHERE
EXP > ANY (SELECT EXP FROM employee.emp_record_table
WHERE EXP > 10);
```



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:

To create a stored procedure-

DELIMITER &&

CREATE PROCEDURE get_experience ()

BEGIN

SELECT * FROM employee.emp_record_table

WHERE EXP > 3;

END &&

To call the for the stored procedure-

CALL get_experience();

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'.

DELIMITER \$\$

drop function employee.role;

CREATE FUNCTION role (exp int)

RETURNS VARCHAR (2255) DETERMINISTIC

BEGIN DECLARE role VARCHAR (2255);

IF experience <= 2 THEN SET role = 'JUNIOR DATA SCIENTIST';

ELSEIF experience <= 5 THEN SET role = 'ASSOCIATE DATA SCIENTIST';

ELSEIF experience <= 10 THEN SET role = 'SENIOR DATA SCIENTIST';

ELSEIF experience <= 12 THEN SET role = 'LEAD DATA SCIENTIST';

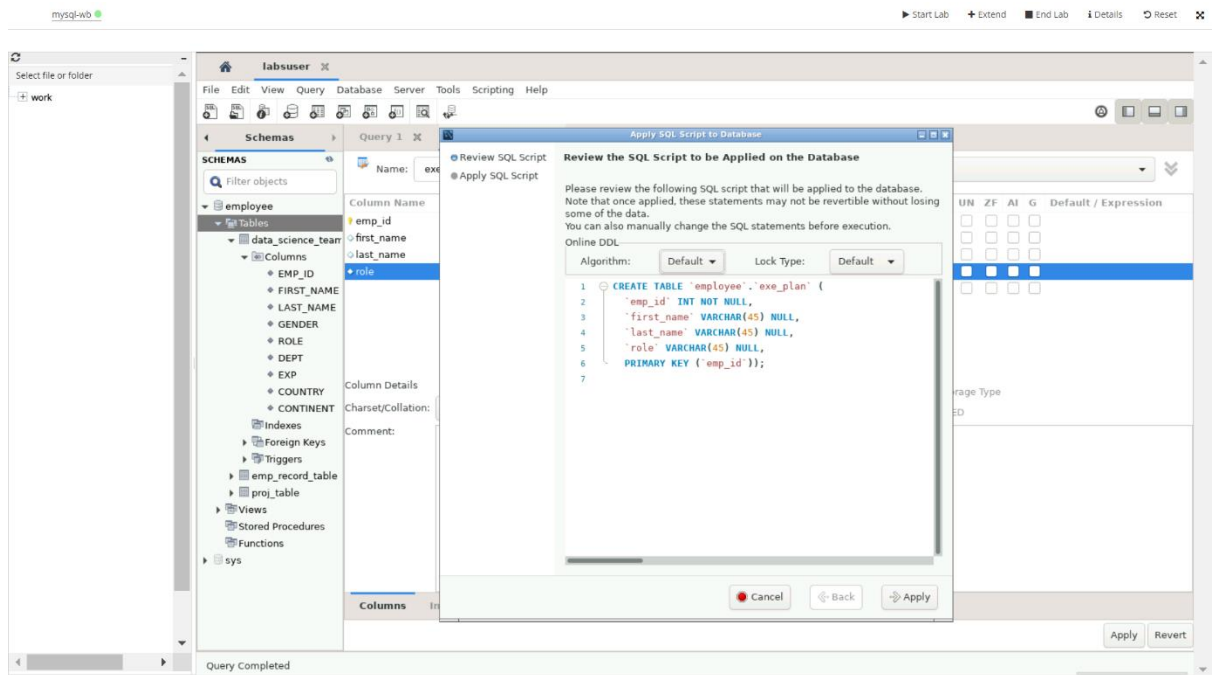
ELSEIF experience > 12 THEN SET role = 'MANAGER'

END IF; RETURN (role); END\$\$ DELIMITER \$\$;

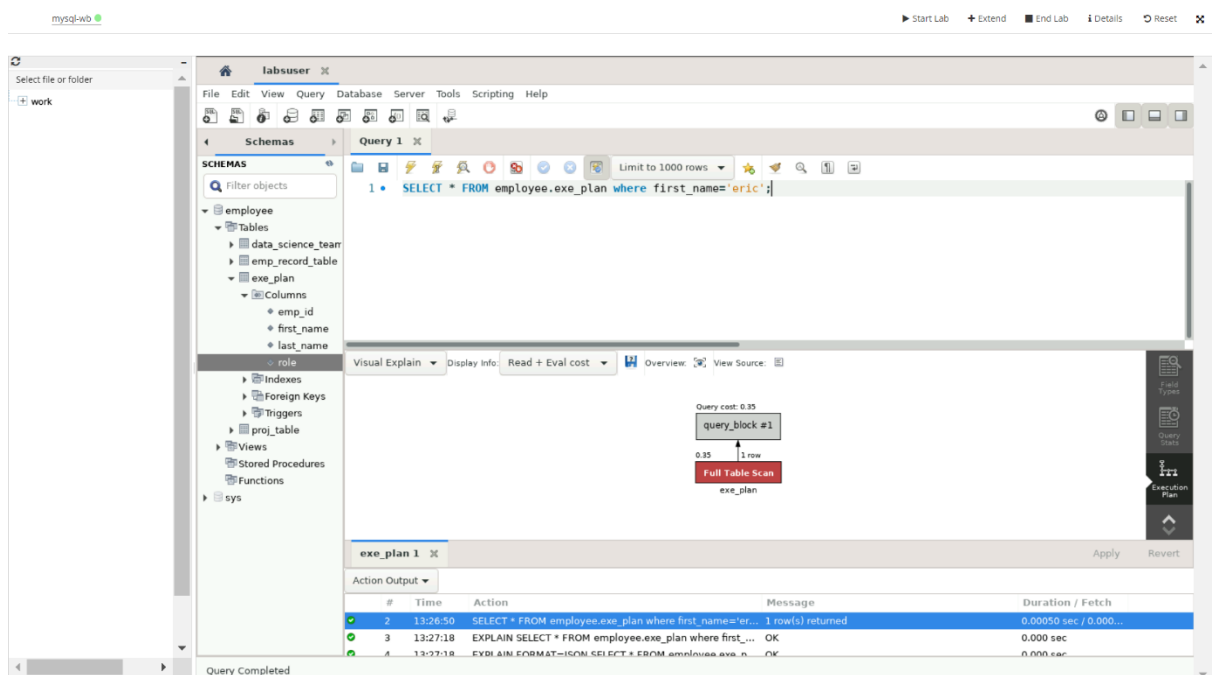
SELECT first_name, last_name, dept, Role(exp) as designation FROM employee.data_science_team
ORDER BY exp;

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.

CREATION OF TABLE-



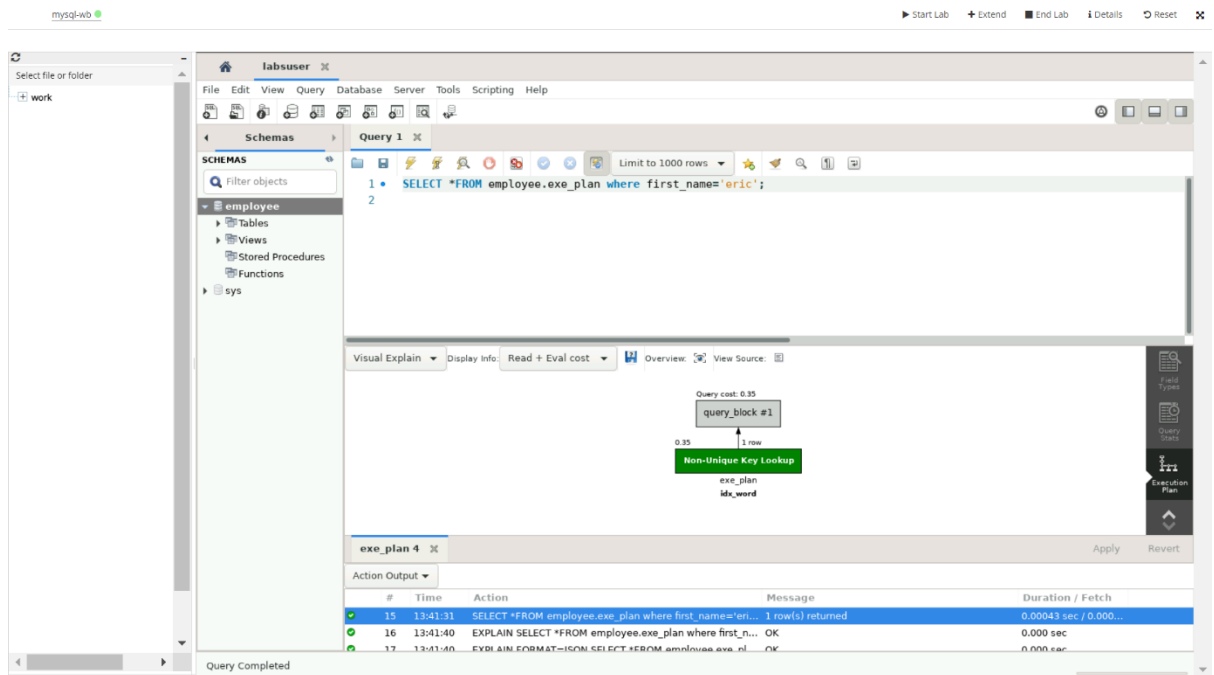
Query cost and performance before creating index-



Query cost and performance after creating index-

Index created by-

CREATE INDEX index_word on employee.exe_plan (FIRST_NAME);



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:

ALTER TABLE employee.emp_record_table

ADD COLUMN BONUS DOUBLE

GENERATED ALWAYS AS (5%(SALARY)*EMP_RATING) STORED;

SELECT * FROM employee.emp_record_table;

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 COUNTRY, CONTINENT, AVG(SALARY) AS AVERAGE_SALARY

FROM employee.emp_record_table GROUP BY COUNTRY, CONTINENT;

mysql-wb

Start Lab + Extend End Lab Details Reset

labuser

File Edit View Query Database Server Tools Scripting Help

Schemas Query 4

Filter objects

tables

data_science_team

emp_record_table

Columns

EMP_ID

FIRST_NAME

LAST_NAME

GENDER

ROLE

DEPT

EXP

COUNTRY

CONTINENT

SALARY

EMP_RATING

MANAGER_ID

PROJ_ID

BONUS

Indexes

Foreign Keys

Triggers

exe_plan

proj_table

Views

Stored Procedures

Functions

sys

Query 4

```
1 SELECT COUNTRY,CONTINENT,AVG(SALARY) AS AVERAGE_SALARY
2 FROM employee.emp_record_table GROUP BY COUNTRY,CONTINENT;
```

Result Grid

Filter Rows

Export: Wrap Cell Content

#	COUNTRY	CONTINENT	AVERAGE_SALAR
1	USA	NORTH AMERICA	9440.0000
2	FRANCE	EUROPE	9000.0000
3	CANADA	NORTH AMERICA	7000.0000
4	GERMANY	EUROPE	7600.0000
5	CHINA	ASIA	6500.0000
6	INDIA	ASIA	6166.6667
7	COLOMBIA	SOUTH AMERICA	5600.0000

Result 3

Read Only

Query Completed