

ScienceQtech Employee Performance Mapping.

Course-end Project 1

Submitted By:

Sagar Sarolia

MS Data Science Job guarantee Aug 2022 Cohort 2

Email: saroliasagar@yahoo.com

INTRODUCTION

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.

Task1: 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.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' panel with a search filter and a tree view showing the 'employee' database and its sub-objects (Tables, Views, Stored Procedures, Functions). The main query editor contains a SQL script to create the 'employee' database, use it, and import data from three CSV files into tables named 'data_science_team', 'emp_record_table', and 'proj_table'. Below the query editor, the 'Result Grid' is visible, showing a table with 8 columns: PROJECT_ID, PROJ_NAME, DOMAIN, START_DATE, CLOSURE_DATE, DEV_QTR, and STATUS. The table contains 6 rows of project data.

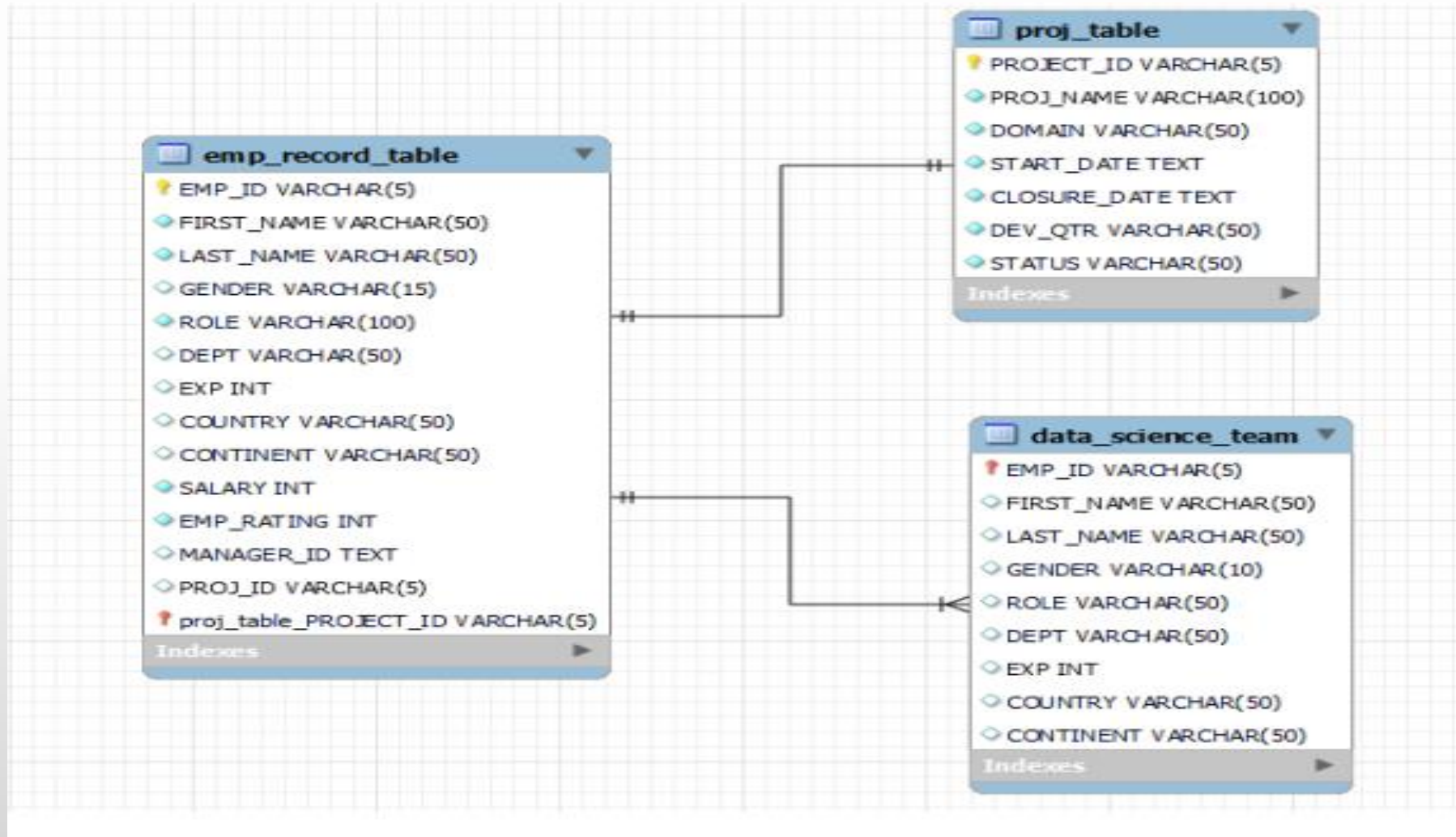
```
1  /*Create a database named employee, then import data_science_team.csv proj_table.csv and emp_record_table.csv
2  into the employee database from the given resources.*/
3  create database employee;
4  use employee;
5
6  /*Importing Data in the following Tables in employee database*/
7  select * from data_science_team;
8  select * from emp_record_table;
9  select * from proj_table;
```

PROJECT_ID	PROJ_NAME	DOMAIN	START_DATE	CLOSURE_DATE	DEV_QTR	STATUS
P103	Drug Discovery	HEALTHCARE	04-06-2021	6/20/2021	Q1	DONE
P105	Fraud Detection	FINANCE	04-11-2021	6/25/2021	Q1	DONE
P109	Market Basket Analysis	RETAIL	04-12-2021	6/30/2021	Q1	DELAYED
P204	Supply Chain Management	AUTOMOTIVE	07/15/2021	9/28/2021	Q2	WIP
P302	Early Detection of Lung Cancer	HEALTHCARE	10-08-2021	12/18/2021	Q3	YTS
P406	Customer Sentiment Analysis	RETAIL	07-09-2021	9/24/2021	Q2	WIP

Schema: employee

Object Info Session data_science_team 5 emp_record_table 6 proj_table 7 x Read Only

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



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.

MySQL Workbench interface showing a query and its result grid. The query is as follows:

```
9 • select * from proj_table;
10 • /*Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER, and DEPARTMENT from the employee record table,
11 • and make a list of employees and details of their department.*/
12
13 • select EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT
14 • from emp_record_table;
15 • select concat(First_name, ' ', last_name) as employees_Name, dept
16 • from emp_record_table;
```

The result grid displays the following data:

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
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

MySQL Workbench interface showing a query and its result grid. The query is as follows:

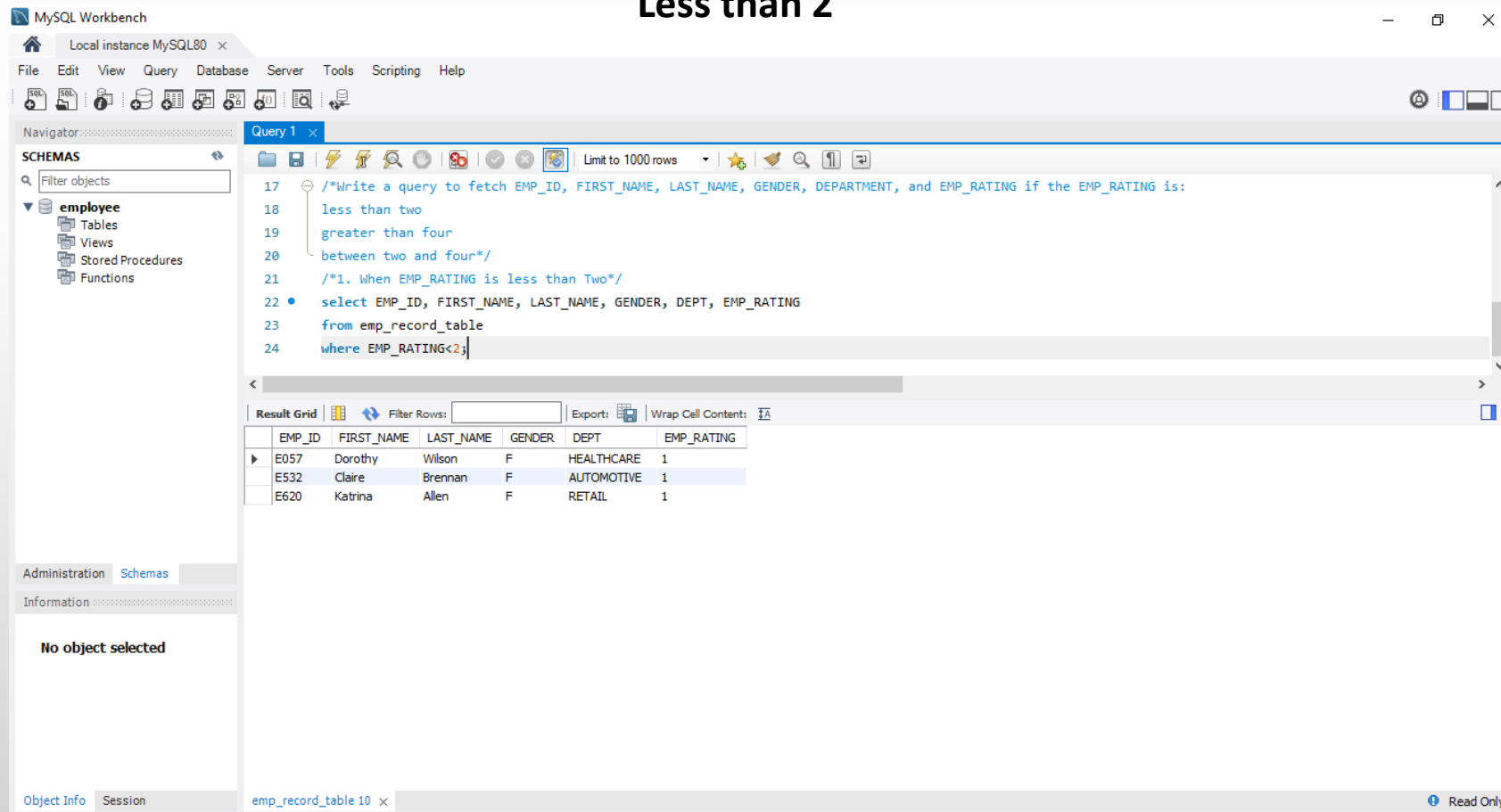
```
9 • select * from proj_table;
10 • /*Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER, and DEPARTMENT from the employee record table,
11 • and make a list of employees and details of their department.*/
12
13 • select EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT
14 • from emp_record_table;
15 • select concat(First_name, ' ', last_name) as employees_Name, dept
16 • from emp_record_table;
```

The result grid displays the following data:

employees_Name	dept
Arthur Black	ALL
Eric Hoffman	FINANCE
William Butler	AUTOMOTIVE
Dianna Wilson	HEALTHCARE
Dorothy Wilson	HEALTHCARE
Patrick Voltz	HEALTHCARE
Emily Grove	FINANCE
Karene Nowak	AUTOMOTIVE
Nian Zhen	RETAIL
Roy Collins	RETAIL
Steve Hoffman	FINANCE
Pete Allen	AUTOMOTIVE
David Smith	RETAIL
Chad Wilson	HEALTHCARE
Claire Brennan	AUTOMOTIVE
Janet Hale	RETAIL
Tracy Norris	RETAIL
Katrina Allen	RETAIL

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

Less than 2



The screenshot shows the MySQL Workbench interface. The 'Query 1' tab is active, displaying a SQL query. The query is as follows:

```
17 /*Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPARTMENT, and EMP_RATING if the EMP_RATING is:
18 less than two
19 greater than four
20 between two and four*/
21 /*1. When EMP_RATING is less than Two*/
22 • select EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
23 from emp_record_table
24 where EMP_RATING<2;
```

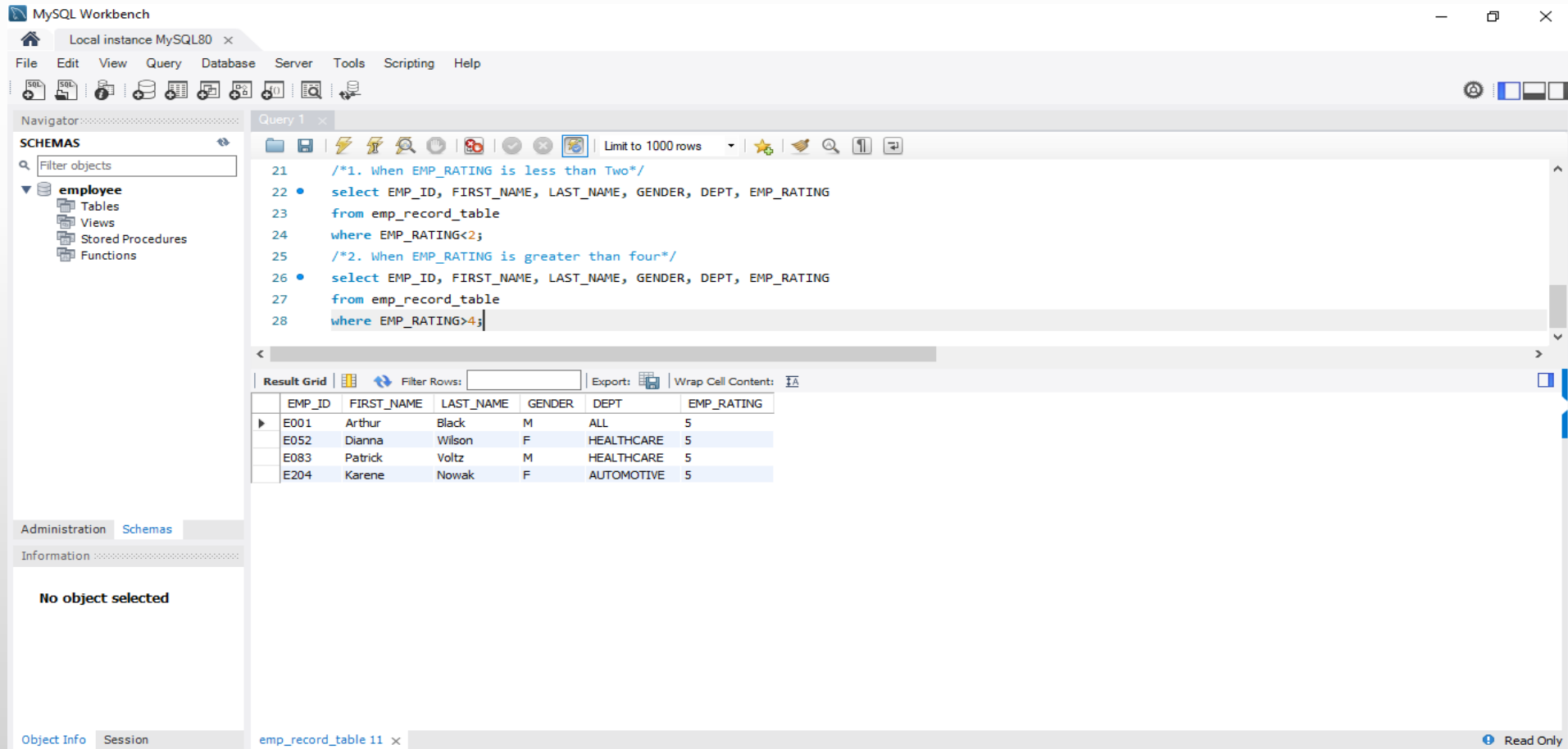
Below the query editor, the 'Result Grid' is visible, showing the results of the query. The results are as follows:

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

The interface also shows the 'SCHEMAS' panel on the left, with the 'employee' schema selected. The 'Administration' and 'Information' tabs are visible at the bottom.

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

Greater than 4



The screenshot shows the MySQL Workbench interface. The 'Navigator' pane on the left displays the 'employee' schema with tables, views, stored procedures, and functions. The 'Query 1' editor contains the following SQL code:

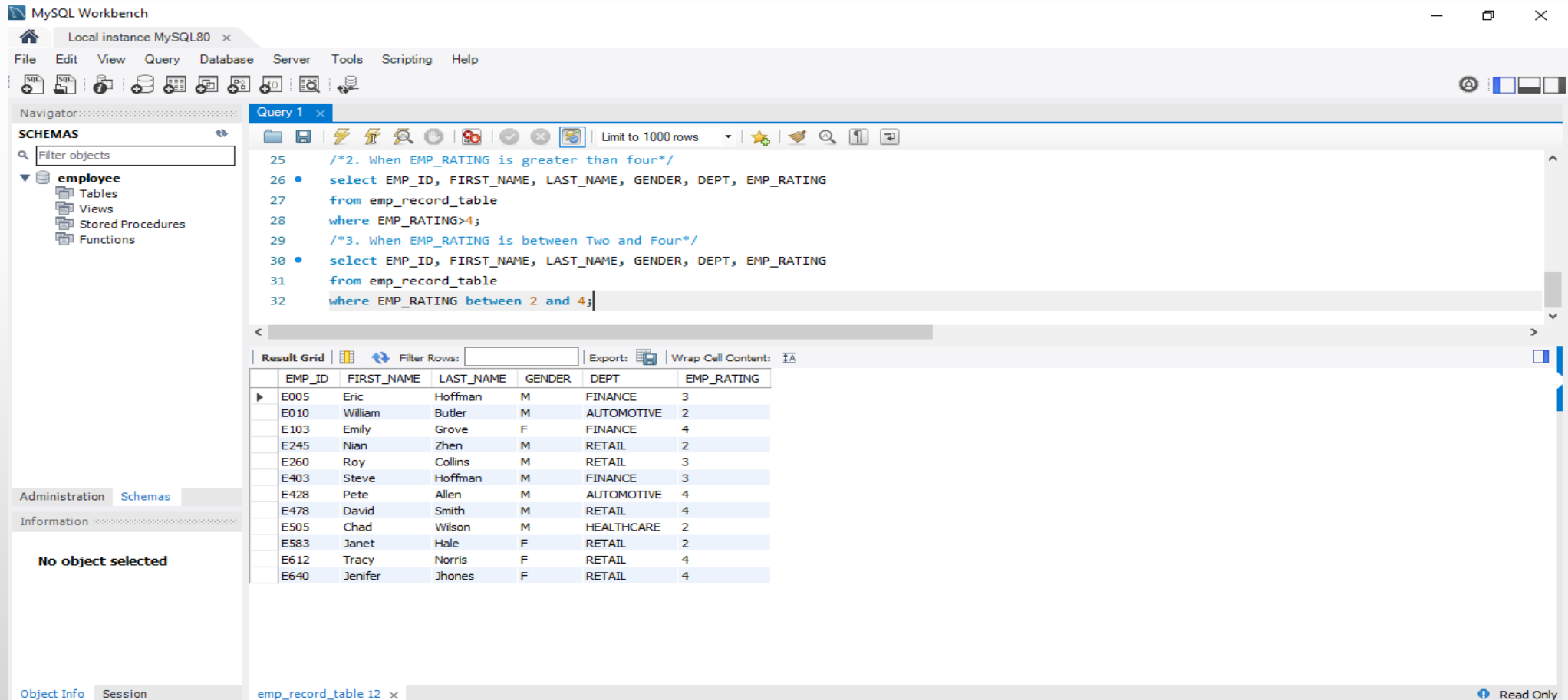
```
21  /*1. When EMP_RATING is less than Two*/
22  •  select EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
23      from emp_record_table
24      where EMP_RATING<2;
25  /*2. When EMP_RATING is greater than four*/
26  •  select EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
27      from emp_record_table
28      where EMP_RATING>4;
```

The 'Result Grid' pane shows the results of the query. It displays a table with 7 columns: EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, and EMP_RATING. The results are as follows:

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

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

Between 2 and 4



The screenshot shows the MySQL Workbench interface. The 'Query 1' tab is active, displaying a SQL query that filters employees based on their rating. The query is as follows:

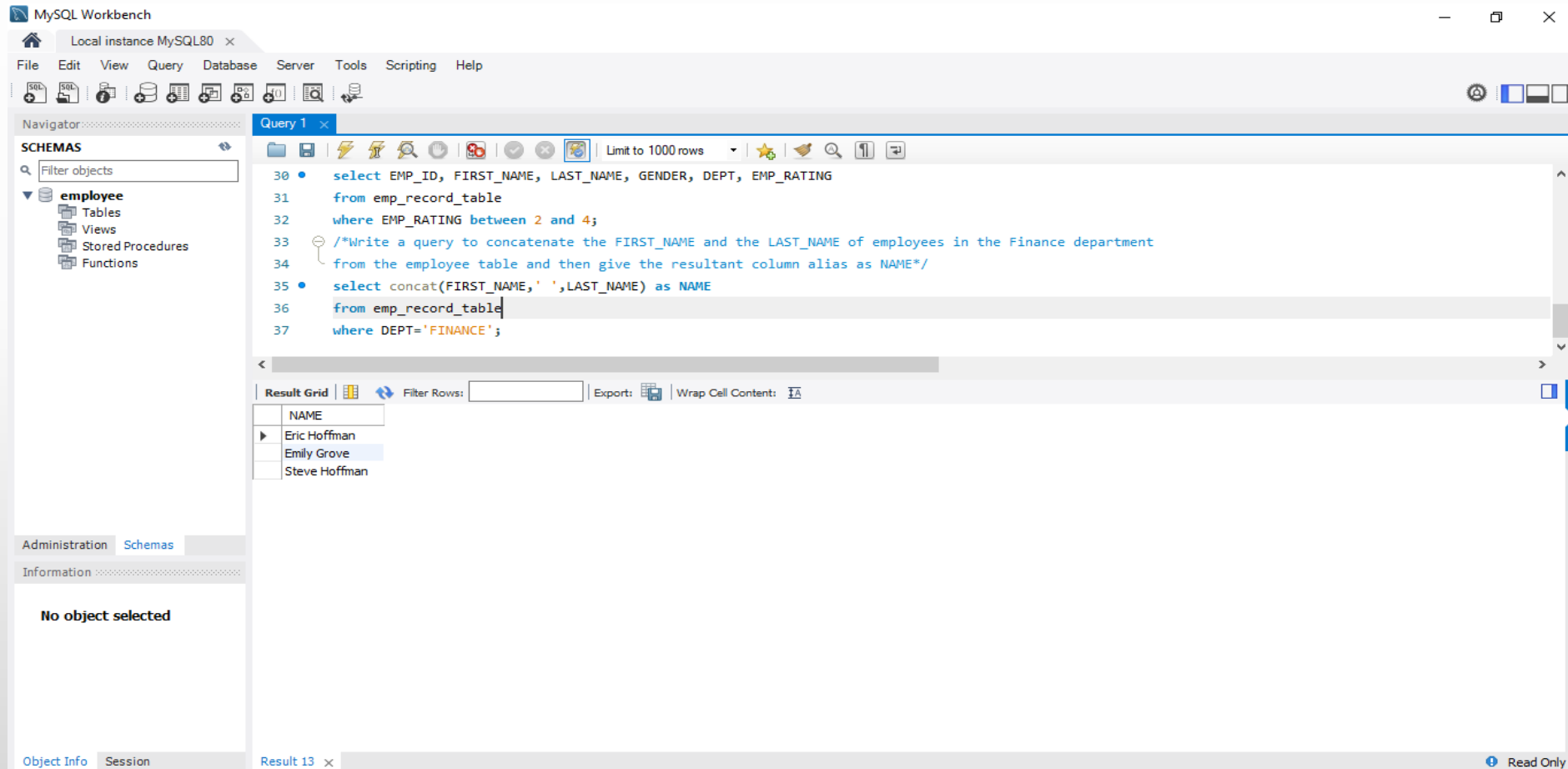
```
/*2. When EMP_RATING is greater than four*/
select EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
from emp_record_table
where EMP_RATING > 4;

/*3. When EMP_RATING is between Two and Four*/
select EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
from emp_record_table
where EMP_RATING between 2 and 4;
```

The 'Result Grid' shows the output of the query, displaying 12 rows of employee data. The columns are EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, and EMP_RATING. The data is as follows:

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
E403	Steve	Hoffman	M	FINANCE	3
E428	Pete	Allen	M	AUTOMOTIVE	4
E478	David	Smith	M	RETAIL	4
E505	Chad	Wilson	M	HEALTHCARE	2
E583	Janet	Hale	F	RETAIL	2
E612	Tracy	Norris	F	RETAIL	4
E640	Jenifer	Jhones	F	RETAIL	4

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.



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' panel with a tree view showing 'employee' and its sub-items: 'Tables', 'Views', 'Stored Procedures', and 'Functions'. The main editor window, titled 'Query 1', contains the following SQL code:

```
30 • select EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
31 from emp_record_table
32 where EMP_RATING between 2 and 4;
33 /*Write a query to concatenate the FIRST_NAME and the LAST_NAME of employees in the Finance department
34 from the employee table and then give the resultant column alias as NAME*/
35 • select concat(FIRST_NAME, ' ', LAST_NAME) as NAME
36 from emp_record_table
37 where DEPT='FINANCE';
```

Below the query editor, the 'Result Grid' is visible, showing the output of the query. The grid has a single column named 'NAME' and three rows of data:

NAME
Eric Hoffman
Emily Grove
Steve Hoffman

The bottom of the interface shows the 'Administration' and 'Schemas' tabs, and the 'Information' panel displays 'No object selected'.

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).

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'employee' database selected. The main editor window shows a SQL query for 'Query 1' with the following code:

```
37 where DEPT='FINANCE';
38 /*Write a query to list only those employees who have someone reporting to them.
39 Also, show the number of reporters (including the President).*/
40 select concat(e.first_name, ' ', e.last_name) as employees, concat(m.first_name, ' ', m.last_name) as manager
41 from emp_record_table e
42 inner join emp_record_table m
43 on m.emp_id=e.manager_id
44 group by employees;
```

Below the query editor, the 'Result Grid' tab is active, displaying the results of the query. The results are shown in a table with two columns: 'employees' and 'manager'.

employees	manager
Tracy Norris	Arthur Black
Janet Hale	Arthur Black
Pete Allen	Arthur Black
Emily Grove	Arthur Black
Patrick Voltz	Arthur Black
Chad Wilson	Patrick Voltz
Dorothy Wilson	Patrick Voltz
Dianna Wilson	Patrick Voltz
Steve Hoffman	Emily Grove
Eric Hoffman	Emily Grove
Claire Brennan	Pete Allen
Karene Nowak	Pete Allen
William Butler	Pete Allen
David Smith	Janet Hale
Roy Collins	Janet Hale
Nian Zhen	Janet Hale
Jenifer Jhones	Tracy Norris
Katrina Allen	Tracy Norris

The bottom status bar indicates 'Result 14' and 'Read Only'.

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.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' panel with a tree view containing 'employee', 'Tables', 'Views', 'Stored Procedures', and 'Functions'. The main editor window, titled 'Query 1', contains the following SQL code:

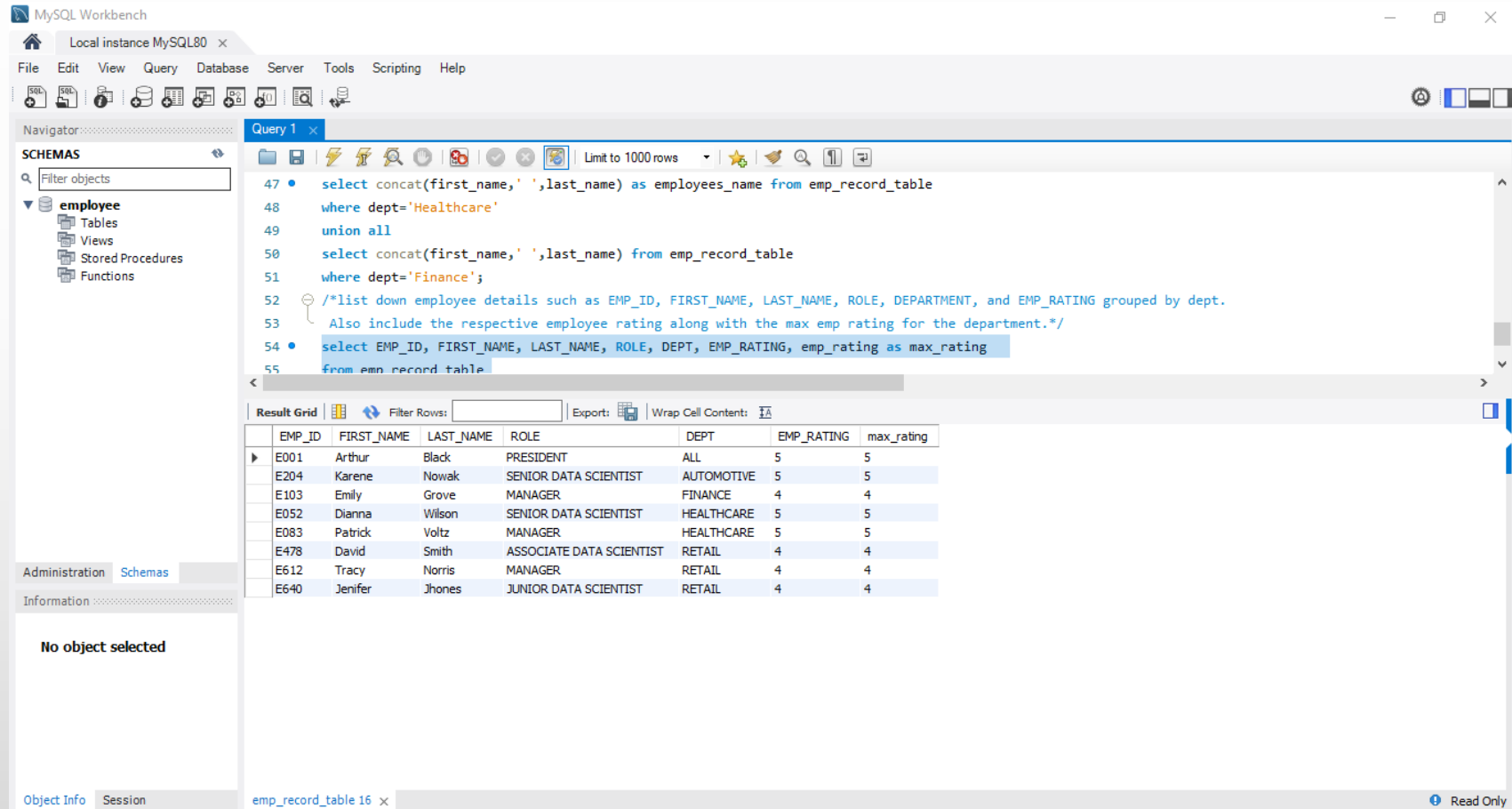
```
44 group by employees;
45 /*Write a query to list down all the employees from the healthcare and finance departments using union.
46 Take data from the employee record table. */
47 • select concat(first_name, ' ',last_name) as employees_name from emp_record_table
48 where dept='Healthcare'
49 union all
50 select concat(first_name, ' ',last_name) from emp_record_table
51 where dept='Finance';
```

Below the query editor, the 'Result Grid' tab is active, showing the results of the query. The results are displayed in a table with one column, 'employees_name', and seven rows of employee names.

employees_name
Dianna Wilson
Dorothy Wilson
Patrick Voltz
Chad Wilson
Eric Hoffman
Emily Grove
Steve Hoffman

The bottom status bar indicates 'Result 15' and 'Read Only'.

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.



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'employee' expanded, showing 'Tables', 'Views', 'Stored Procedures', and 'Functions'. The main editor shows a SQL query with line numbers 47 to 55. The query is as follows:

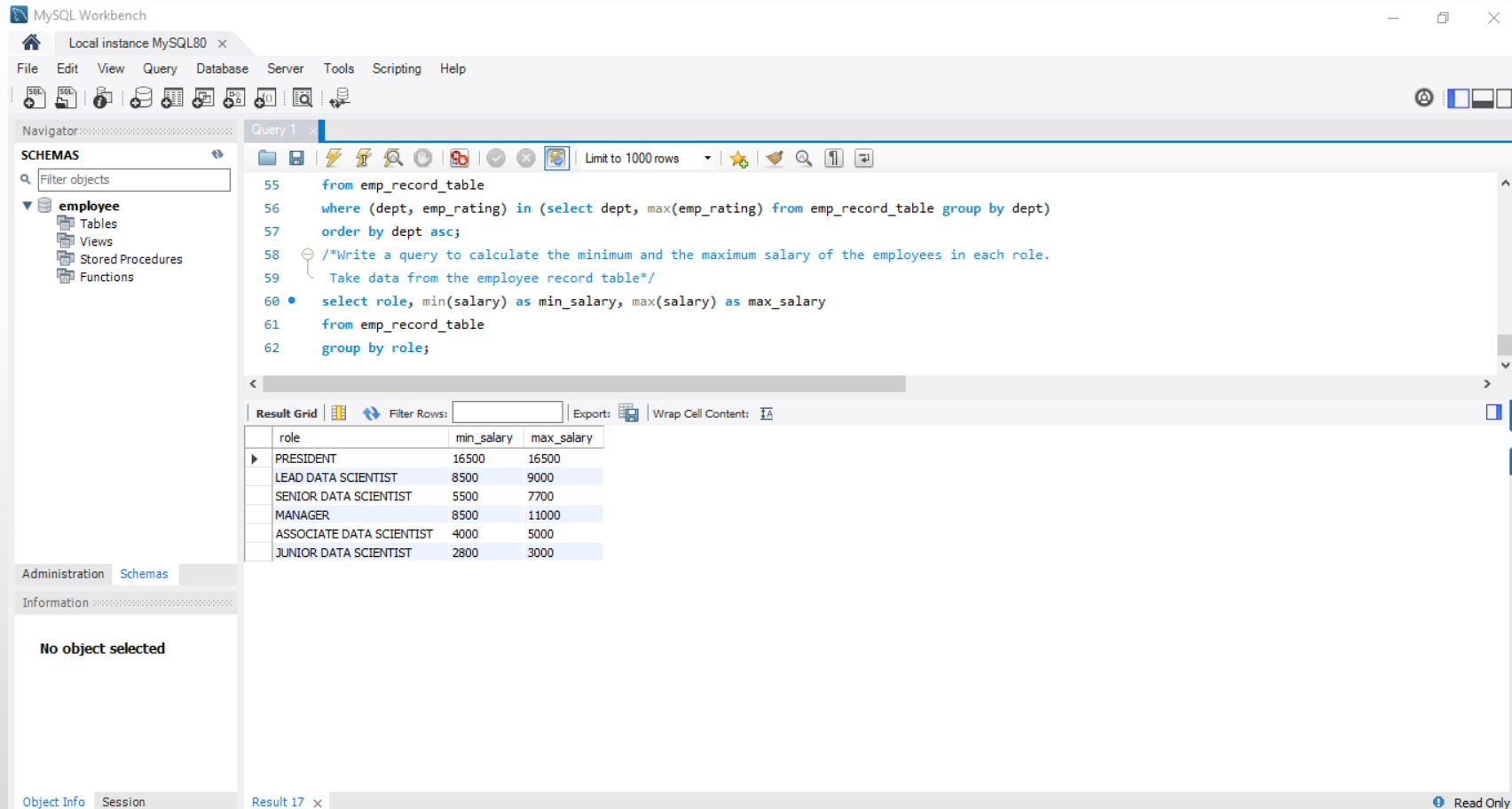
```
47 • select concat(first_name, ' ', last_name) as employees_name from emp_record_table
48   where dept='Healthcare'
49   union all
50   select concat(first_name, ' ', last_name) from emp_record_table
51   where dept='Finance';
52   /*list down employee details such as EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPARTMENT, and EMP_RATING grouped by dept.
53   Also include the respective employee rating along with the max emp rating for the department.*/
54 • select EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, EMP_RATING, emp_rating as max_rating
55   from emp_record_table
```

Below the query editor, the 'Result Grid' is displayed, showing a table with 7 columns: EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, EMP_RATING, and max_rating. The table contains 8 rows of data.

EMP_ID	FIRST_NAME	LAST_NAME	ROLE	DEPT	EMP_RATING	max_rating
E001	Arthur	Black	PRESIDENT	ALL	5	5
E204	Karene	Nowak	SENIOR DATA SCIENTIST	AUTOMOTIVE	5	5
E103	Emily	Grove	MANAGER	FINANCE	4	4
E052	Dianna	Wilson	SENIOR DATA SCIENTIST	HEALTHCARE	5	5
E083	Patrick	Voltz	MANAGER	HEALTHCARE	5	5
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

The bottom status bar shows 'Object Info', 'Session', and 'emp_record_table 16 x' with a 'Read Only' indicator.

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.



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' panel with a tree view showing 'employee' and its sub-items: 'Tables', 'Views', 'Stored Procedures', and 'Functions'. The main editor window, titled 'Query 1', contains the following SQL code:

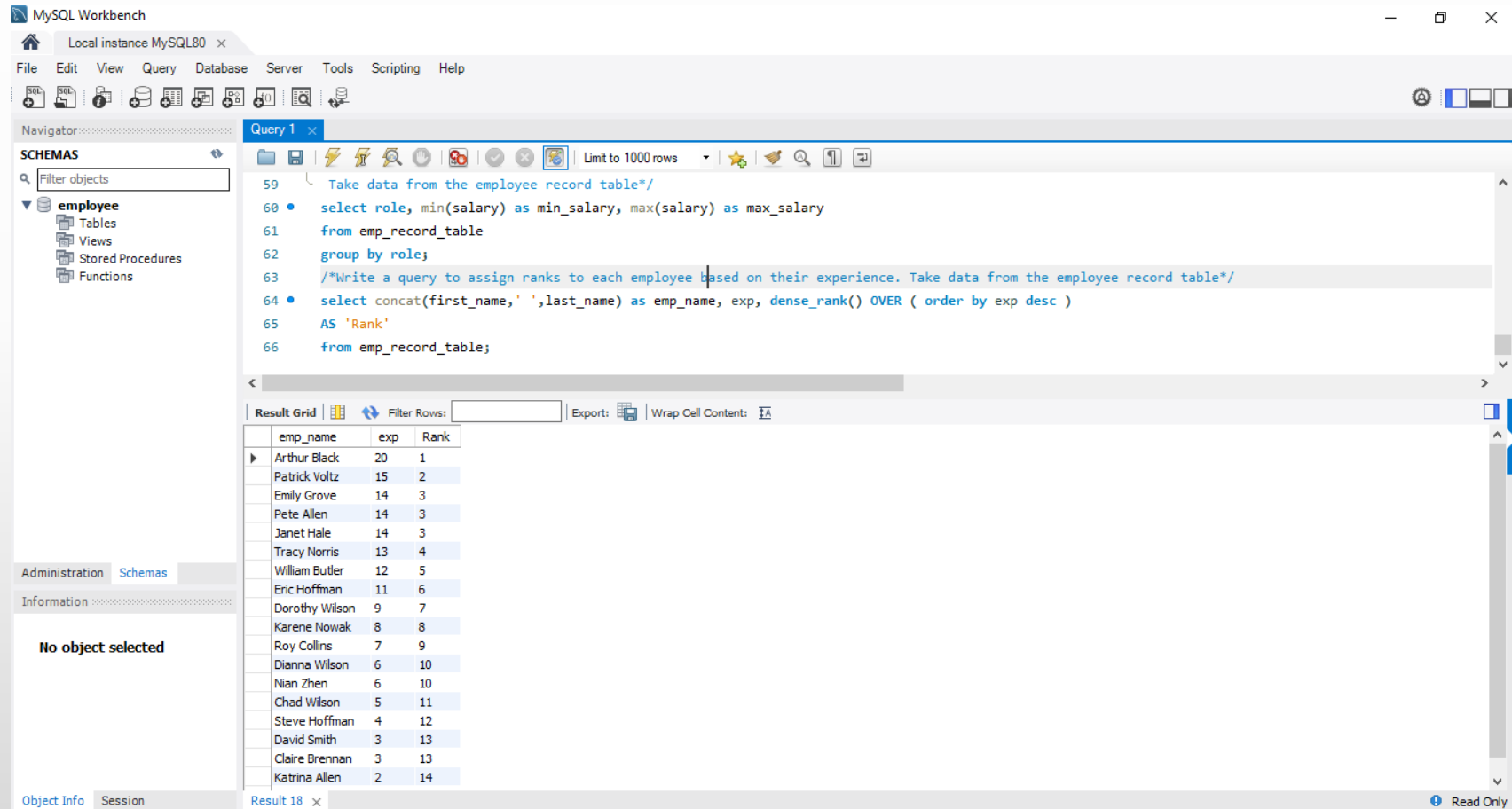
```
55 from emp_record_table
56 where (dept, emp_rating) in (select dept, max(emp_rating) from emp_record_table group by dept)
57 order by dept asc;
58 /*Write a query to calculate the minimum and the maximum salary of the employees in each role.
59 Take data from the employee record table*/
60 • select role, min(salary) as min_salary, max(salary) as max_salary
61 from emp_record_table
62 group by role;
```

Below the query editor, the 'Result Grid' tab is active, displaying the results of the query in a table format. The table has four columns: 'role', 'min_salary', and 'max_salary'. The results are as follows:

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
JUNIOR DATA SCIENTIST	2800	3000

The bottom status bar indicates 'Object Info', 'Session', 'Result 17', and 'Read Only'.

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



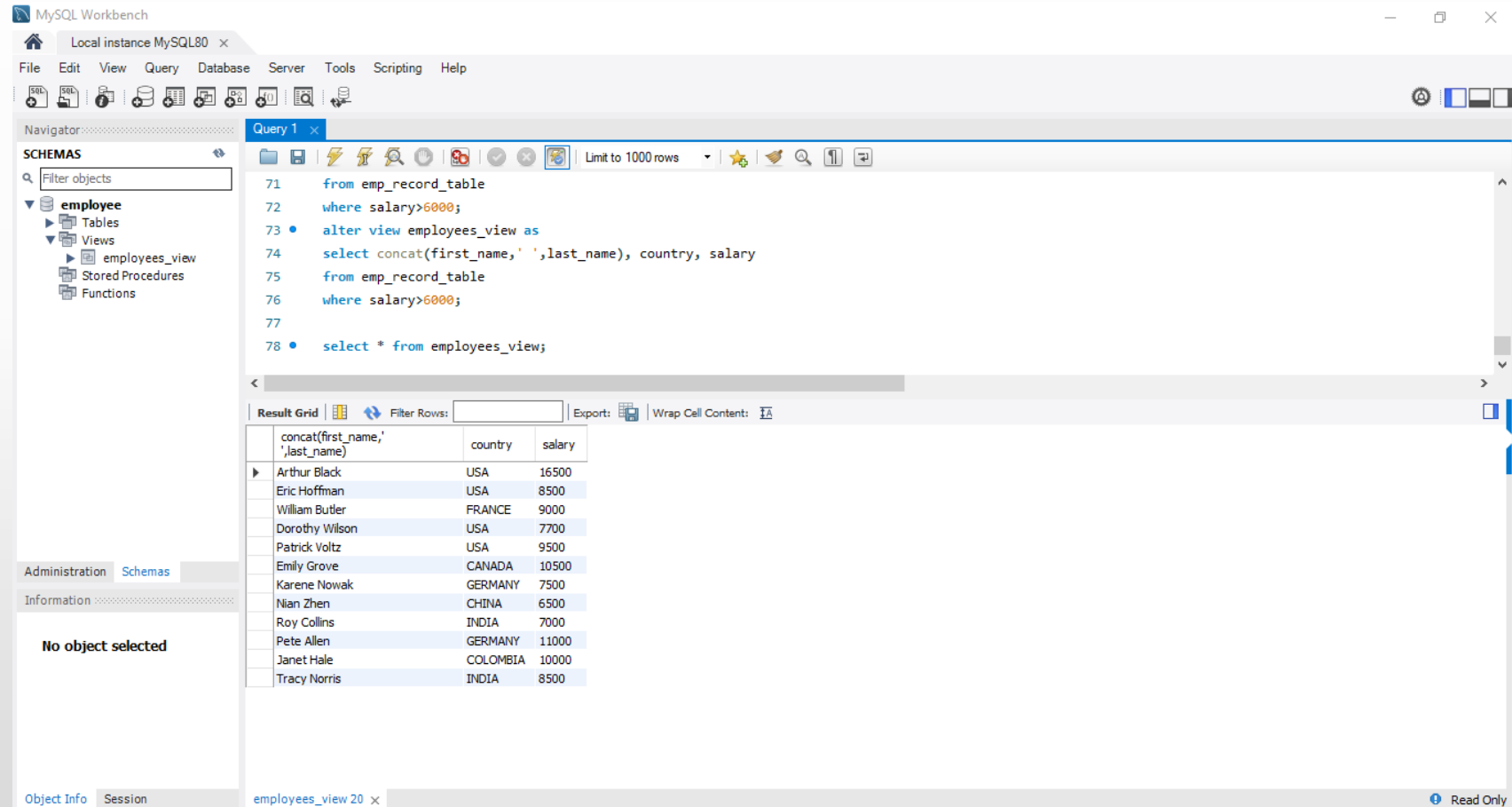
The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'employee' selected. The main editor shows a SQL query in 'Query 1' that assigns ranks to employees based on their experience. The query is as follows:

```
59  /* Take data from the employee record table*/
60  select role, min(salary) as min_salary, max(salary) as max_salary
61  from emp_record_table
62  group by role;
63  /*Write a query to assign ranks to each employee based on their experience. Take data from the employee record table*/
64  select concat(first_name, ' ', last_name) as emp_name, exp, dense_rank() OVER ( order by exp desc )
65  AS 'Rank'
66  from emp_record_table;
```

The 'Result Grid' at the bottom displays the output of the query, showing a list of employees with their experience and assigned rank. The results are as follows:

emp_name	exp	Rank
Arthur Black	20	1
Patrick Voltz	15	2
Emily Grove	14	3
Pete Allen	14	3
Janet Hale	14	3
Tracy Norris	13	4
William Butler	12	5
Eric Hoffman	11	6
Dorothy Wilson	9	7
Karene Nowak	8	8
Roy Collins	7	9
Dianna Wilson	6	10
Nian Zhen	6	10
Chad Wilson	5	11
Steve Hoffman	4	12
David Smith	3	13
Claire Brennan	3	13
Katrina Allen	2	14

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.



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'employee' database selected, showing 'Tables' and 'Views'. The 'Views' folder is expanded, showing 'employees_view'. The main editor window displays a SQL query in 'Query 1' tab:

```
71 from emp_record_table
72 where salary>6000;
73 • alter view employees_view as
74 select concat(first_name,' ',last_name), country, salary
75 from emp_record_table
76 where salary>6000;
77
78 • select * from employees_view;
```

Below the query editor, the 'Result Grid' tab is active, showing the output of the query. The grid has four columns: 'concat(first_name, ' ', last_name)', 'country', and 'salary'. The data is as follows:

concat(first_name, ' ', last_name)	country	salary
Arthur Black	USA	16500
Eric Hoffman	USA	8500
William Butler	FRANCE	9000
Dorothy Wilson	USA	7700
Patrick Voltz	USA	9500
Emily Grove	CANADA	10500
Karene Nowak	GERMANY	7500
Nian Zhen	CHINA	6500
Roy Collins	INDIA	7000
Pete Allen	GERMANY	11000
Janet Hale	COLOMBIA	10000
Tracy Norris	INDIA	8500

The bottom status bar shows 'Object Info', 'Session', and 'employees_view 20 x' with a 'Read Only' indicator.

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

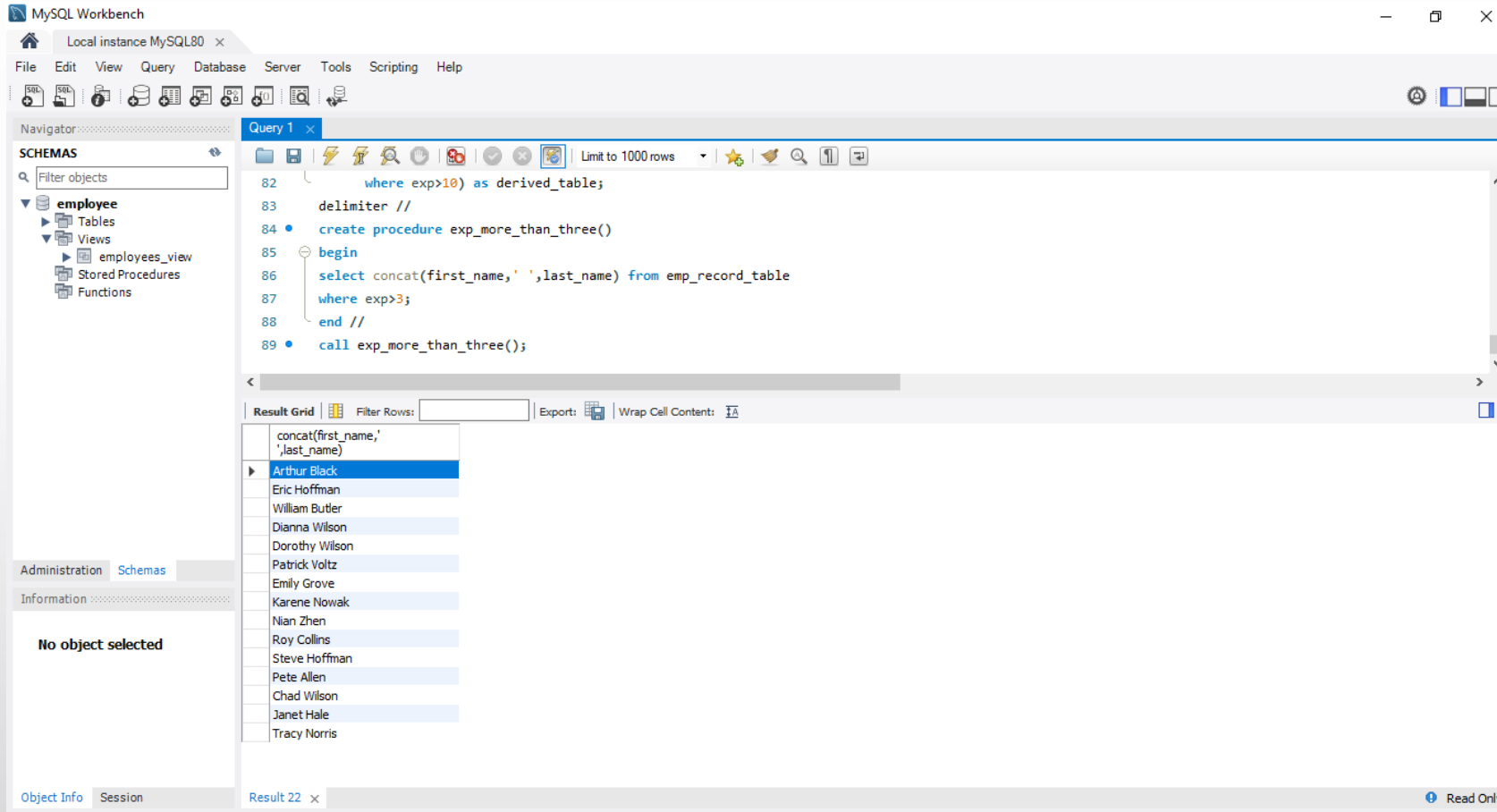
The screenshot shows the MySQL Workbench interface with a query editor and a result grid. The query is a nested SELECT statement designed to find employees with more than 10 years of experience by joining the `employees_view` with a subquery derived from `emp_record_table`.

```
76 where salary>6000;  
77  
78 select * from employees_view;  
79 select concat(first_name, ' ', last_name) as employee_name, exp  
80 from (select *  
81 from emp_record_table  
82 where exp>10) as derived_table;  
83
```

The result grid displays the following data:

employee_name	EXP
Arthur Black	20
Eric Hoffman	11
William Butler	12
Patrick Voltz	15
Emily Grove	14
Pete Allen	14
Janet Hale	14
Tracy Norris	13

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.



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'employee' database expanded, showing 'Tables', 'Views' (including 'employees_view'), 'Stored Procedures', and 'Functions'. The main editor window, titled 'Query 1', contains the following SQL code:

```
82     where exp>10) as derived_table;
83 delimiter //
84 • create procedure exp_more_than_three()
85 begin
86     select concat(first_name, ' ', last_name) from emp_record_table
87     where exp>3;
88 end //
89 • call exp_more_than_three();
```

Below the editor, the 'Result Grid' tab is active, showing the output of the query. The first column is labeled 'concat(first_name, ' ', last_name)'. The results are as follows:

concat(first_name, ' ', last_name)
Arthur Black
Eric Hoffman
William Butler
Dianna Wilson
Dorothy Wilson
Patrick Voltz
Emily Grove
Karene Nowak
Nian Zhen
Roy Collins
Steve Hoffman
Pete Allen
Chad Wilson
Janet Hale
Tracy Norris

The bottom status bar indicates 'Object Info', 'Session', and 'Result 22 x'.

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.

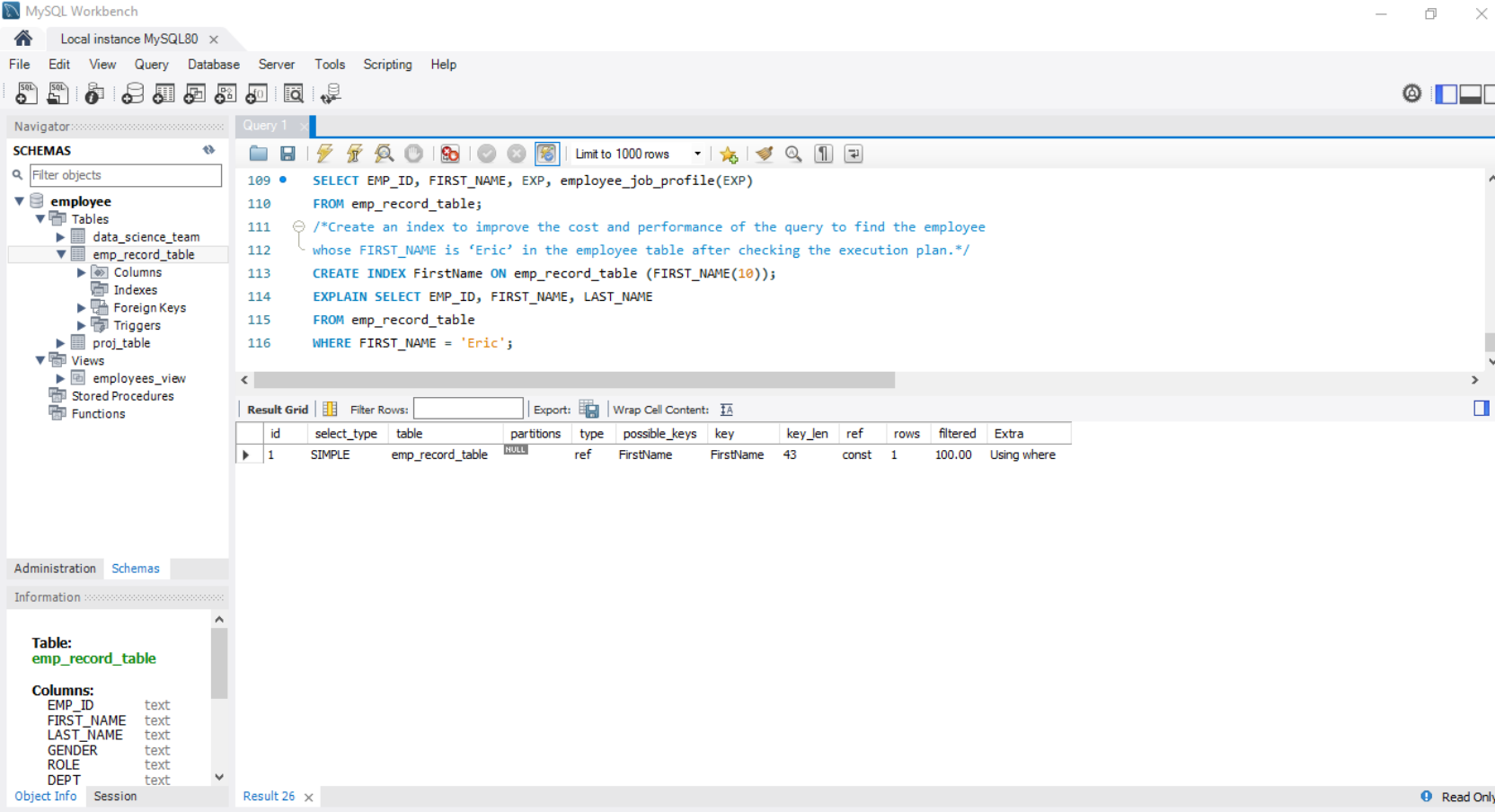
The screenshot displays the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows the 'employee' database with its tables, views, stored procedures, and functions. The main editor window, titled 'Query 1', contains a SQL script that defines a stored function 'employee_job_profile' and a query to use it. The function logic is as follows:

```
103 SET employee_job_profile = 'LEAD DATA SCIENTIST';
104 ELSEIF EXP BETWEEN 12 AND 16 THEN
105 SET employee_job_profile = 'MANAGER';
106 END IF;
107 RETURN (employee_job_profile);
108 END$$
109 SELECT EMP_ID, FIRST_NAME, EXP, employee_job_profile(EXP)
110 FROM emp_record_table;
```

Below the editor, the 'Result Grid' shows the output of the query. It lists 23 rows of employee data, including their ID, name, experience level, and the job profile assigned by the function.

EMP_ID	FIRST_NAME	EXP	employee_job_profile(EXP)
E001	Arthur	20	NULL
E005	Eric	11	LEAD DATA SCIENTIST
E010	William	12	LEAD DATA SCIENTIST
E052	Dianna	6	SENIOR DATA SCIENTIST
E057	Dorothy	9	SENIOR DATA SCIENTIST
E083	Patrick	15	MANAGER
E103	Emily	14	MANAGER
E204	Karene	8	SENIOR DATA SCIENTIST
E245	Nian	6	SENIOR DATA SCIENTIST
E260	Roy	7	SENIOR DATA SCIENTIST
E403	Steve	4	ASSOCIATE DATA SCIENT...
E428	Pete	14	MANAGER
E478	David	3	ASSOCIATE DATA SCIENT...
E505	Chad	5	ASSOCIATE DATA SCIENT...
E532	Claire	3	ASSOCIATE DATA SCIENT...
E583	Janet	14	MANAGER
E612	Tracy	13	MANAGER
E620	Katrina	2	JUNIOR DATA SCIENTIST

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.



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'employee' schema expanded, showing tables like 'emp_record_table'. The main editor contains a SQL query (Query 1) with the following code:

```
109 • SELECT EMP_ID, FIRST_NAME, EXP, employee_job_profile(EXP)
110 FROM emp_record_table;
111 /*Create an index to improve the cost and performance of the query to find the employee
112 whose FIRST_NAME is 'Eric' in the employee table after checking the execution plan.*/
113 CREATE INDEX FirstName ON emp_record_table (FIRST_NAME(10));
114 EXPLAIN SELECT EMP_ID, FIRST_NAME, LAST_NAME
115 FROM emp_record_table
116 WHERE FIRST_NAME = 'Eric';
```

Below the query editor, the 'Result Grid' is displayed, showing the execution plan for the query. The grid has columns: id, select_type, table, partitions, type, possible_keys, key, key_len, ref, rows, filtered, and Extra. The first row of the plan is:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	emp_record_table	NONE	ref	FirstName	FirstName	43	const	1	100.00	Using where

The bottom of the interface shows the 'Information' tab with details for the table 'emp_record_table', including its columns: EMP_ID, FIRST_NAME, LAST_NAME, GENDER, ROLE, and DEPT.

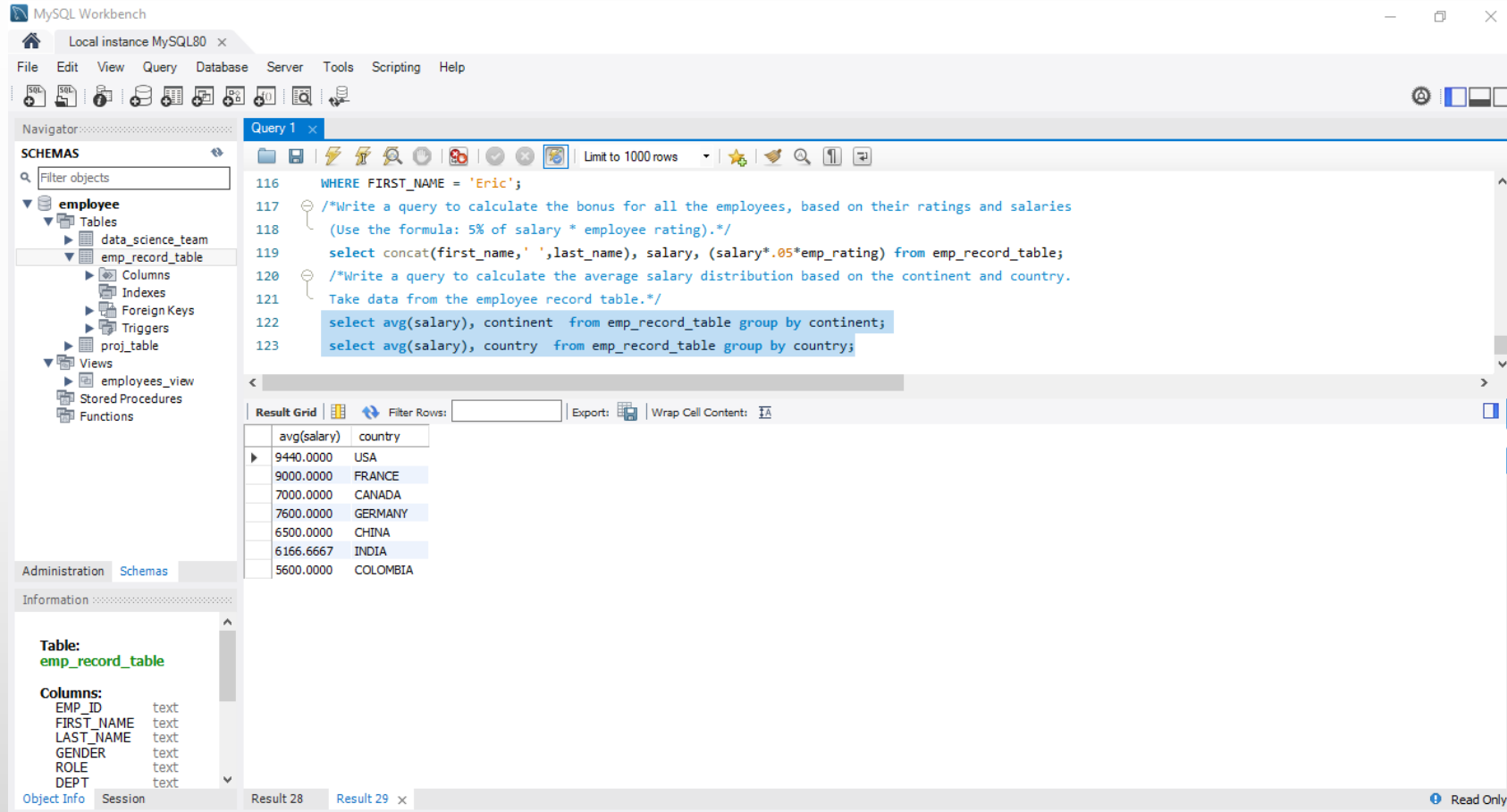
Task 16: Write a query to calculate the bonus for all the employees, based on their ratings and salaries (Use the formula: 5% of salary * employee rating).

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'employee' database selected, showing tables like 'emp_record_table'. The main editor window contains a SQL query with line numbers 112 to 119. The query includes a comment about Eric's first name, an index creation statement, an EXPLAIN statement, and a SELECT statement that calculates a bonus as 5% of salary multiplied by employee rating. The 'Result Grid' at the bottom shows the output of the query, with columns for concatenated first and last names, salary, and the calculated bonus.

```
112  whose FIRST_NAME is 'Eric' in the employee table after checking the execution plan.*/
113  CREATE INDEX FirstName ON emp_record_table (FIRST_NAME(10));
114  EXPLAIN SELECT EMP_ID, FIRST_NAME, LAST_NAME
115  FROM emp_record_table
116  WHERE FIRST_NAME = 'Eric';
117  /*Write a query to calculate the bonus for all the employees, based on their ratings and salaries
118  (Use the formula: 5% of salary * employee rating).*/
119  select concat(first_name,' ',last_name), salary, (salary*.05*emp_rating) from emp_record_table;
```

concat(first_name,' ',last_name)	salary	(salary*.05*emp_rating)
Arthur Black	16500	4125.00
Eric Hoffman	8500	1275.00
William Butler	9000	900.00
Dianna Wilson	5500	1375.00
Dorothy Wilson	7700	385.00
Patrick Voltz	9500	2375.00
Emily Grove	10500	2100.00
Karene Nowak	7500	1875.00
Nian Zhen	6500	650.00
Roy Collins	7000	1050.00
Steve Hoffman	5000	750.00
Pete Allen	11000	2200.00
David Smith	4000	800.00
Chad Wilson	5000	500.00
Claire Brennan	4300	215.00
Janet Hale	10000	1000.00
Tracy Norris	8500	1700.00

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



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'employee' database expanded, showing tables like 'emp_record_table'. The main editor window contains a SQL query with two SELECT statements. The first statement calculates the average salary by continent, and the second statement calculates the average salary by country. The 'Result Grid' at the bottom shows the output of the second query, displaying average salaries for various countries.

```
116 WHERE FIRST_NAME = 'Eric';
117 /*Write a query to calculate the bonus for all the employees, based on their ratings and salaries
118 (Use the formula: 5% of salary * employee rating).*/
119 select concat(first_name, ' ', last_name), salary, (salary*.05*emp_rating) from emp_record_table;
120 /*Write a query to calculate the average salary distribution based on the continent and country.
121 Take data from the employee record table.*/
122 select avg(salary), continent from emp_record_table group by continent;
123 select avg(salary), country from emp_record_table group by country;
```

avg(salary)	country
9440.0000	USA
9000.0000	FRANCE
7000.0000	CANADA
7600.0000	GERMANY
6500.0000	CHINA
6166.6667	INDIA
5600.0000	COLOMBIA

Table: emp_record_table

Columns:

- EMP_ID text
- FIRST_NAME text
- LAST_NAME text
- GENDER text
- ROLE text
- DEPT text