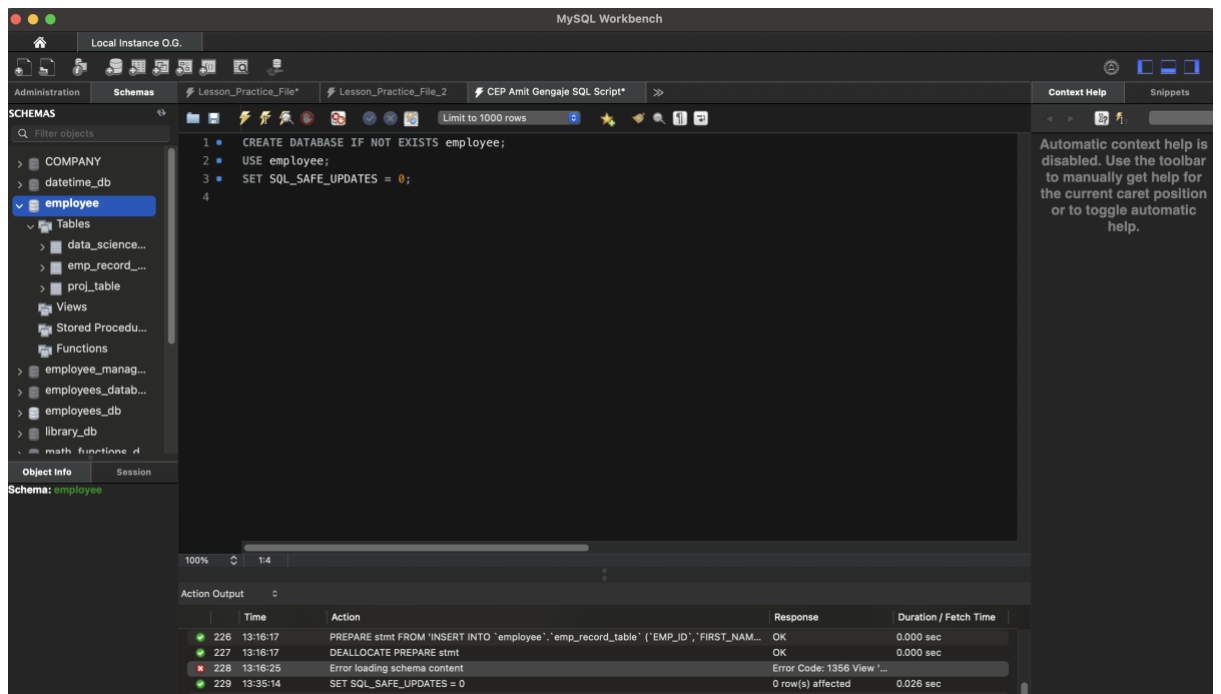
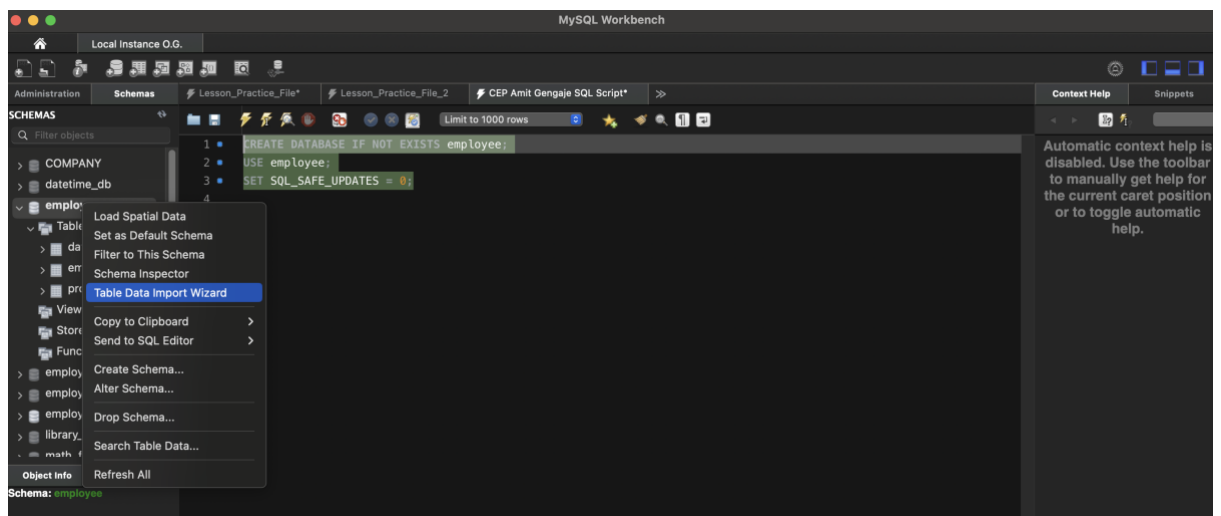


## 1. Input SQL Queries:

```
CREATE DATABASE IF NOT EXISTS employee;  
USE employee;  
SET SQL_SAFE_UPDATES = 0;
```

Used the Table Data Import Wizard to import the three CEP dataset CSV files as tables “data\_science\_team”, “emp\_record\_table” and “proj\_table”

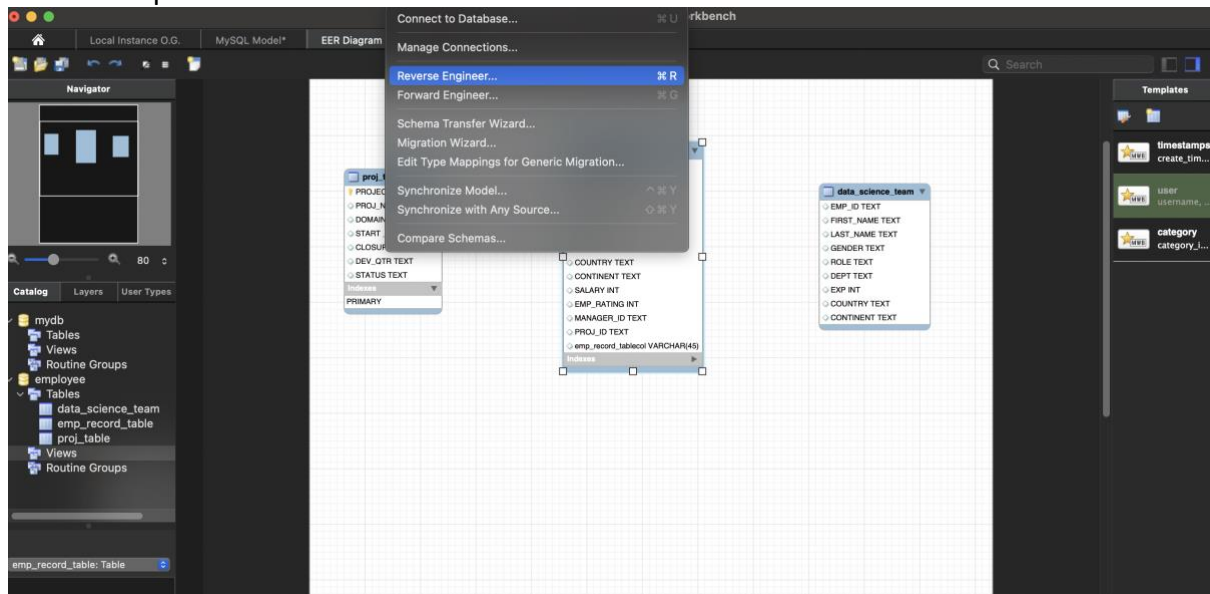
Output:



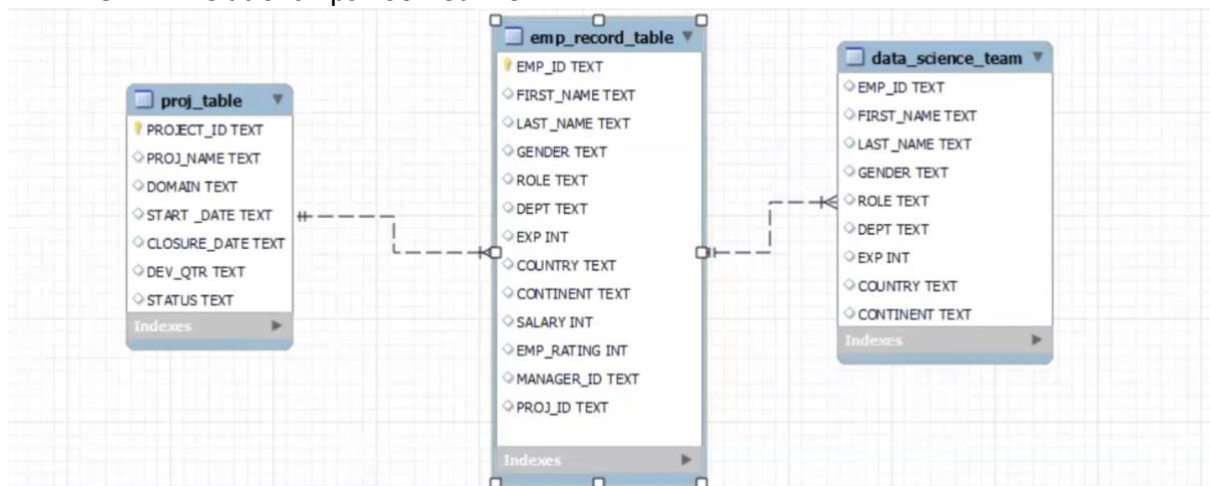
## 2. Input SQL Action:

Database tab -> Reverse Engineer DB -> Select Schemas (Chose schema created above) -> kept clicking next

Output:



ER DIAGRAM Relationships Zoomed View:



### 3. Input SQL Query:

SELECT emp\_id, first\_name, last\_name, gender, dept FROM emp\_record\_table;

Output:

The screenshot shows the SQL Developer interface with a SQL script in the main editor. The script contains the following statements:

```
1 CREATE DATABASE IF NOT EXISTS employee;
2 USE employee;
3 SET SQL_SAFE_UPDATES = 0;
4 SELECT * FROM emp_record_table;
5 SELECT emp_id, first_name, last_name, gender, dept FROM emp_record_table;
```

The 'Result Grid' displays the output of the last query, showing columns: emp\_id, first\_name, last\_name, gender, and dept. The data is as follows:

emp_id	first_name	last_name	gender	dept
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

The 'Action Output' pane at the bottom shows the execution log:

	Time	Action	Response	Duration / Fetch Time
229	13:35:14	SET SQL_SAFE_UPDATES = 0	0 row(s) affected	0.026 sec
230	08:12:22	Error loading schema content	Error Code: 1356 View '...	
231	12:05:36	SELECT * FROM emp_record_table LIMIT 0, 1000	19 row(s) returned	0.059 sec / 0.00032...
232	17:07:37	SELECT emp_id, first_name, last_name, gender, dept FROM emp_record_table LIMIT...	19 row(s) returned	0.042 sec / 0.00030...

Result Grid Zoom:

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
E640	Jenifer	Jhones	F	RETAIL

#### 4. Input SQL Query:

```
SELECT emp_id, first_name, last_name, gender, dept, emp_rating FROM  
emp_record_table WHERE  
emp_rating <2  
OR emp_rating >4  
OR (emp_rating >=2 AND emp_rating <=4);
```

#### Output:

The screenshot shows the SQL Developer interface with a SQL script in the main editor and its results in the Result Grid.

**SQL Script:**

```
1 CREATE DATABASE IF NOT EXISTS employee;  
2 USE employee;  
3 SET SQL_SAFE_UPDATES = 0;  
4 SELECT * FROM emp_record_table;  
5 SELECT emp_id, first_name, last_name, gender, dept FROM emp_record_table;  
6 SELECT emp_id, first_name, last_name, gender, dept, emp_rating FROM emp_record_table WHERE  
7 emp_rating <2  
8 OR emp_rating >4  
9 OR (emp_rating >=2 AND emp_rating <=4);  
10
```

**Result Grid:**

emp_id	first_name	last_name	gender	dept	emp_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

**Action Output:**

	Time	Action	Response	Duration / Fetch Time
27	19:11:12	SELECT * from emp_record_table WHERE first_name = "Eric" LIMIT 0, 1000	1 row(s) returned	0.0036 sec / 0.00001...
28	19:12:49	SELECT EMP_ID, FIRST_NAME, LAST_NAME, SALARY, EMP_RATIN...	19 row(s) returned	0.0045 sec / 0.0000...
29	19:15:07	SELECT * , avg(salary) over (partition by country) AS country_wise, avg(salary)...	19 row(s) returned	0.026 sec / 0.000038...
30	19:28:09	SELECT emp_id, first_name, last_name, gender, dept, emp_rating FROM emp_r...	19 row(s) returned	0.050 sec / 0.00034...

#### Result Grid Zoom:

emp_id	first_name	last_name	gender	dept	emp_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

5. Input SQL Query:

```
SELECT CONCAT(first_name, ' ', last_name) NAME  
FROM emp_record_table WHERE dept = 'FINANCE';
```

Output:

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'employee' schema selected. The main editor window contains the following SQL script:

```
1 CREATE DATABASE IF NOT EXISTS employee;  
2 USE employee;  
3 SET SQL_SAFE_UPDATES = 0;  
4 SELECT * FROM emp_record_table;  
5 SELECT emp_id, first_name, last_name, gender, dept FROM emp_record_table;  
6 SELECT emp_id, first_name, last_name, gender, dept, emp_rating FROM emp_record_table WHERE  
7 emp_rating <2  
8 OR emp_rating >4  
9 OR (emp_rating >2 AND emp_rating <=4);  
10 SELECT CONCAT(first_name, ' ', last_name) NAME  
11 FROM emp_record_table WHERE dept = 'FINANCE';
```

Below the script, the 'Result Grid' shows the output of the last query (Result 4):

NAME
Eric Hoffman
Emily Grove
Steve Hoffman

The 'Action Output' pane at the bottom shows the execution log:

	Time	Action	Response	Duration / Fetch Time
231	12:05:36	SELECT * FROM emp_record_table LIMIT 0, 1000	19 row(s) returned	0.059 sec / 0.00032...
232	17:07:37	SELECT emp_id, first_name, last_name, gender, dept FROM emp_record_table LIMIT...	19 row(s) returned	0.042 sec / 0.00030...
233	17:11:54	SELECT emp_id, first_name, last_name, gender, dept, emp_rating FROM emp_record_...	15 row(s) returned	0.051 sec / 0.00033...
234	17:14:21	SELECT CONCAT(first_name, ' ', last_name) NAME FROM emp_record_table WHERE...	3 row(s) returned	0.057 sec / 0.00030...

6. Input SQL Query:

```
SELECT manager_id, count(*)  
FROM emp_record_table  
GROUP BY manager_id  
ORDER BY manager_id;
```

Output:

The screenshot shows the MySQL Workbench interface with the following components:

- SCHEMAS Panel:** A tree view on the left showing the database structure. The 'employee' schema is selected, showing tables like 'data\_science...', 'emp\_record...', 'proj\_table', and views like 'Stored Procedu...'. The 'Object Info' tab is active, showing the 'Schema: employee'.
- SQL Editor:** The main area contains a SQL script with five queries:
 

```

5 SELECT emp_id, first_name, last_name, gender, dept FROM emp_record_table;
6 SELECT emp_id, first_name, last_name, gender, dept, emp_rating FROM emp_record_table WHERE
7   emp_rating <2
8   OR emp_rating >4
9   OR (emp_rating >2 AND emp_rating <=4);
10 SELECT CONCAT(first_name, ' ', last_name) NAME
11 FROM emp_record_table WHERE dept = 'FINANCE';
12 SELECT manager_id, count(*)
13 FROM emp_record_table
14 GROUP BY manager_id
15 ORDER BY manager_id;
```
- Result Grid:** The bottom panel shows the results of the queries. The first query (Result 5) is selected, displaying a table with columns 'manager\_id' and 'count(\*)'. The data is as follows:
 

manager_id	count(*)
NULL	1
E001	5
E083	3
E103	2
E428	3
E583	3
E612	2
- Action Output:** The bottom-most panel shows a log of executed queries with their execution times and row counts. The last three entries are:
 

	Time	Action	Response	Duration / Fetch Time
232	17:07:37	SELECT emp_id, first_name, last_name, gender, dept FROM emp_record_table LIMIT...	19 row(s) returned	0.042 sec / 0.00030...
233	17:11:54	SELECT emp_id, first_name, last_name, gender, dept, emp_rating FROM emp_record_...	15 row(s) returned	0.051 sec / 0.00033...
234	17:14:21	SELECT CONCAT(first_name, ' ', last_name) NAME FROM emp_record_table WHERE...	3 row(s) returned	0.057 sec / 0.00030...
235	17:16:57	SELECT manager_id, count(*) FROM emp_record_table GROUP BY manager_id ORDE...	7 row(s) returned	0.031 sec / 0.00015...

## 7. Input SQL Query:

```

Select first_name, last_name, dept FROM emp_record_table WHERE dept =
'HEALTHCARE'
UNION
SELECT first_name, last_name, dept FROM emp_record_table WHERE dept =
'FINANCE';
```



Output:

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'employee' schema selected. The main editor shows a SQL script with the following queries:

```

8 OR emp_rating >4
9 OR (emp_rating >2 AND emp_rating <=4);
10 SELECT CONCAT(first_name, ' ', last_name) NAME
11 FROM emp_record_table WHERE dept = 'FINANCE';
12 SELECT manager_id, count(*)
13 FROM emp_record_table
14 GROUP BY manager_id
15 ORDER BY manager_id;
16 Select first_name, last_name, dept FROM emp_record_table WHERE dept = 'HEALTHCARE'
17 UNION
18 SELECT first_name, last_name, dept FROM emp_record_table WHERE dept = 'FINANCE';

```

The 'Result Grid' shows the output of the last query (Result 6):

first_name	last_name	dept
Dianna	Wilson	HEALTHCARE
Dorothy	Wilson	HEALTHCARE
Patrick	Voltz	HEALTHCARE
Chad	Wilson	HEALTHCARE
Eric	Hoffman	FINANCE
Emily	Grove	FINANCE
Steve	Hoffman	FINANCE

The 'Action Output' pane at the bottom shows the execution details for the queries:

	Time	Action	Response	Duration / Fetch Time
233	17:11:54	SELECT emp_id, first_name, last_name, gender, dept, emp_rating FROM emp_record_...	15 row(s) returned	0.051 sec / 0.00033...
234	17:14:21	SELECT CONCAT(first_name, ' ', last_name) NAME FROM emp_record_table WHERE...	3 row(s) returned	0.057 sec / 0.00030...
235	17:16:57	SELECT manager_id, count(*) FROM emp_record_table GROUP BY manager_id ORDE...	7 row(s) returned	0.031 sec / 0.000015...
236	17:17:58	Select first_name, last_name, dept FROM emp_record_table WHERE dept = 'HEALTH...	7 row(s) returned	0.021 sec / 0.000022...

## 8. Input SQL Query:

```

SELECT emp_id, first_name, last_name, role, dept, emp_rating, max(emp_rating)
Over (partition by dept) AS max_emp_rating
FROM emp_record_table;

```

Output:

AdministrationSchemasLesson\_Practice\_File\*Lesson\_Practice\_File\_2CEP Amit Gengaje SQL Script\*new\_procedure - Routine>>

SCHEMAS

Filter objects

COMPANYdatetime\_dbemployeeTablesdata\_science...emp\_record...proj\_tableViewsStored Procedu...Functionsemployee\_manag...employees\_datab...employees\_dblibrary\_dbmath\_functions\_d...

Object InfoSessionSchema: employeee

Limit to 1000 rows

11FROM emp\_record\_table WHERE dept = 'FINANCE';12SELECT manager\_id, count(\*)13FROM emp\_record\_table14GROUP BY manager\_id15ORDER BY manager\_id;16Select first\_name, last\_name, dept FROM emp\_record\_table WHERE dept = 'HEALTHCARE'17UNION18SELECT first\_name, last\_name, dept FROM emp\_record\_table WHERE dept = 'FINANCE';19SELECT emp\_id, first\_name, last\_name, role, dept, emp\_rating, max(emp\_rating)20Over (partition by dept) AS max\_emp\_rating21FROM emp\_record\_table;

100%1:19

Result GridFilter Rows:SearchExport:

emp_id	first_name	last_name	role	dept	emp_rating	max_emp_rati...
E001	Arthur	Black	PRESIDENT	ALL	5	5
E010	William	Butler	LEAD DATA SCIENTIST	AUTOMOTIVE	2	5
E204	Karene	Nowak	SENIOR DATA SCIENTIST	AUTOMOTIVE	5	5
E428	Pete	Allen	MANAGER	AUTOMOTIVE	4	5
E532	Claire	Brennan	ASSOCIATE DATA SCIENTIST	AUTOMOTIVE	1	5
E005	Eric	Hoffman	LEAD DATA SCIENTIST	FINANCE	3	4
E103	Emily	Grove	MANAGER	FINANCE	4	4
E403	Steve	Hoffman	ASSOCIATE DATA SCIENTIST	FINANCE	3	4
E052	Dianna	Wiliam	SENIOR DATA SCIENTIST	HEALTHCARE	5	5

Result 8Read Only

Action Output

	Time	Action	Response	Duration / Fetch Time
235	17:16:57	SELECT manager_id, count(*) FROM emp_record_table GROUP BY manager_id ORDE...	7 row(s) returned	0.031 sec / 0.000015...
236	17:17:58	Select first_name, last_name, dept FROM emp_record_table WHERE dept = 'HEALTH...	7 row(s) returned	0.021 sec / 0.000022...
237	17:20:00	SELECT emp_id, first_name, last_name, role, dept, emp_rating, emp_rating AS max_r...	8 row(s) returned	0.033 sec / 0.000016...
238	17:22:27	SELECT emp_id, first_name, last_name, role, dept, emp_rating, max(emp_rating) Ove...	19 row(s) returned	0.024 sec / 0.000024...



Result Grid Zoom:

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

9. Input SQL Query:

```

Select role, MIN(salary) AS min_salary,
MAX(salary) AS max_salary
FROM emp_record_table
GROUP BY ROLE;

```

Output:

The screenshot shows a database management tool interface with the following components:

- Top Bar:** Local Instance O.G., MySQL Model\*, EER Diagram.
- Navigation Panel (Left):** Schemas, Filter objects, COMPANY, datetime\_db, employee (expanded), Tables (data\_science..., emp\_record..., proj\_table), Views, Stored Procedure..., Functions, employee\_manag..., employees\_datab..., employees\_db, library\_db, math\_functions\_d...
- SQL Editor (Center):**

```

15 ORDER BY manager_id;
16 Select first_name, last_name, dept FROM emp_record_table WHERE dept = 'HEALTHCARE';
17 UNION
18 SELECT first_name, last_name, dept FROM emp_record_table WHERE dept = 'FINANCE';
19 SELECT emp_id, first_name, last_name, role, dept, emp_rating, max(emp_rating)
20 Over (partition by dept) AS max_emp_rating
21 FROM emp_record_table;
22 Select role, MIN(salary) AS min_salary,
23 MAX(salary) AS max_salary
24 FROM emp_record_table
25 GROUP BY ROLE;

```
- Result Grid (Bottom):**

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
- Action Output (Bottom):**

	Time	Action	Response	Duration / Fetch Time
236	17:17:58	Select first_name, last_name, dept FROM emp_record_table WHERE dept = 'HEALTH...	7 row(s) returned	0.021 sec / 0.000022...
237	17:20:00	SELECT emp_id, first_name, last_name, role, dept, emp_rating, emp_rating AS max_r...	8 row(s) returned	0.033 sec / 0.000016...
238	17:22:27	SELECT emp_id, first_name, last_name, role, dept, emp_rating, max(emp_rating) Ove...	19 row(s) returned	0.024 sec / 0.000024...
239	17:24:27	Select role, MIN(salary) AS min_salary, MAX(salary) AS max_salary FROM emp_record...	6 row(s) returned	0.038 sec / 0.00021 s...

## 10. Input SQL Query:

```

SELECT emp_id, first_name, last_name, exp, DENSE_RANK() OVER (ORDER BY EXP
DESC) AS experience_rank
FROM emp_record_table;

```

Output:

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'employee' selected. The main editor shows a SQL script with the following content:

```
17 UNION
18 SELECT first_name, last_name, dept FROM emp_record_table WHERE dept = 'FINANCE';
19 SELECT emp_id, first_name, last_name, role, dept, emp_rating, max(emp_rating)
20 Over (partition by dept) AS max_emp_rating
21 FROM emp_record_table;
22 Select role, MIN(salary) AS min_salary,
23 MAX(salary) AS max_salary
24 FROM emp_record_table
25 GROUP BY ROLE;
26 SELECT emp_id, first_name, last_name, exp, DENSE_RANK() OVER (ORDER BY EXP DESC) AS experience_rank
27 FROM emp_record_table;
```

The 'Result Grid' shows the output of the last query (Result 18):

emp_id	first_name	last_name	exp	experience_rank
E001	Arthur	Black	20	1
E083	Patrick	Voltz	15	2
E103	Emily	Grove	14	3
E428	Pete	Allen	14	3
E583	Janet	Hale	14	3
E612	Tracy	Norris	13	4
E010	William	Butler	12	5
E005	Eric	Hoffman	11	6
E057	Dorothy	Wilson	9	7

The 'Action Output' pane at the bottom shows the execution log:

	Time	Action	Response	Duration / Fetch Time
6	17:44:36	Select first_name, last_name, dept FROM emp_record_table WHERE dept = 'HEAL...	7 row(s) returned	0.0027 sec / 0.00000...
7	17:44:38	SELECT emp_id, first_name, last_name, role, dept, emp_rating, max(emp_rating)...	19 row(s) returned	0.0044 sec / 0.00001...
8	17:44:44	Select role, MIN(salary) AS min_salary, MAX(salary) AS max_salary FROM emp_rec...	6 row(s) returned	0.0022 sec / 0.00001...
9	18:14:33	SELECT emp_id, first_name, last_name, exp, DENSE_RANK() OVER (ORDER BY EX...	19 row(s) returned	0.052 sec / 0.000052...

Result Grid Zoom:

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

## 11. Input SQL Query:

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

```
SELECT * FROM employee_location;
```

## Outputs:

The screenshot shows the SQL Developer interface with the 'Schemas' tab selected. The 'employee' schema is expanded, showing tables like 'emp\_record\_table' and 'proj\_table'. The SQL script in the editor includes a 'CREATE VIEW' statement and a 'SELECT \*' query. The 'Result Grid' displays the output of the 'SELECT \*' query, showing columns: emp\_id, first\_name, last\_name, country, salary. The 'Action Output' pane shows the execution log with timestamps and messages.

emp_id	first_name	last_name	country	salary
E001	Arthur	Black	USA	16500
E005	Eric	Hoffman	USA	8500
E010	William	Butler	FRANCE	9000
E057	Dorothy	Wilson	USA	7700
E083	Patrick	Voltz	USA	9500
E103	Emily	Grove	CANADA	10500
E204	Karene	Nowak	GERMANY	7500
E245	Nian	Zhen	CHINA	6500
E290	Rou	Online	INDIA	7000

Time	Action	Response	Duration / Fetch Time
10 18:18:05	CREATE VIEW employee_location SELECT emp_id, first_name, last_name, country,...	Error Code: 1064. You have...	0.015 sec
11 18:18:36	CREATE VIEW employee_location	Error Code: 1064. You have...	0.00076 sec
12 18:25:45	CREATE VIEW employee_location AS SELECT emp_id, first_name, last_name, coun...	0 row(s) affected	0.102 sec
13 18:26:12	SELECT * FROM employee_location LIMIT 0, 1000	12 row(s) returned	0.025 sec / 0.00027...

The screenshot shows the 'Views' tab selected in the 'Schemas' pane. The 'employee\_location' view is expanded, showing its columns: emp\_id, first\_name, last\_name, country, salary.

## Result Grid Zoom:

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

## 12. Input SQL Query:

```
SELECT * FROM emp_record_table WHERE emp_id
IN (SELECT emp_id FROM emp_record_table WHERE exp > 10);
```

## Output:

The screenshot shows a database management tool interface. The left sidebar displays a schema tree for the 'employee' schema, including tables like 'data\_science...', 'emp\_record...', 'proj\_table', and views like 'employee\_location'. The main area shows a SQL script with the following queries:

```
22 Select role, MIN(salary) AS min_salary,
23 MAX(salary) AS max_salary
24 FROM emp_record_table
25 GROUP BY ROLE;
26 SELECT emp_id, first_name, last_name, exp, DENSE_RANK() OVER (ORDER BY EXP DESC) AS experience_rank
27 FROM emp_record_table;
28 CREATE VIEW employee_location AS
29 SELECT emp_id, first_name, last_name, country, salary FROM emp_record_table WHERE salary > 6000;
30 SELECT * FROM employee_location;
31 SELECT * FROM emp_record_table WHERE emp_id
32 IN (SELECT emp_id FROM emp_record_table WHERE exp > 10);
```

The 'Result Grid' shows the output of the last query, displaying columns: EMP\_ID, FIRST\_NAME, LAST\_NAME, GENDER, ROLE, DEPT, EXP, COUNTRY, CONTINENT, SALARY, EMP\_RATING, and M. The results are as follows:

EMP_ID	FIRST_NAME	LAST_NAME	GENDER	ROLE	DEPT	EXP	COUNTRY	CONTINENT	SALARY	EMP_RATING	M
E001	Arthur	Black	M	PRESIDENT	ALL	20	USA	NORTH AMERICA	16500	5	
E005	Eric	Hoffman	M	LEAD DATA SCIENTIST	FINANCE	11	USA	NORTH AMERICA	8500	3	E1
E010	William	Butler	M	LEAD DATA SCIENTIST	AUTOMOTIVE	12	FRANCE	EUROPE	9000	2	E4
E083	Patrick	Voltz	M	MANAGER	HEALTHCARE	15	USA	NORTH AMERICA	9500	5	E0
E103	Emily	Grove	F	MANAGER	FINANCE	14	CANADA	NORTH AMERICA	10500	4	E0
E428	Pete	Allen	M	MANAGER	AUTOMOTIVE	14	GERMANY	EUROPE	11000	4	E0
E583	Janet	Hale	F	MANAGER	RETAIL	14	COLOMBIA	SOUTH AMERICA	10000	2	E0
E612	Tracy	Norris	F	MANAGER	RETAIL	13	INDIA	ASIA	8500	4	E0

The 'Action Output' section shows the execution log:

	Time	Action	Response	Duration / Fetch Time
11	18:18:36	CREATE VIEW employee_location	Error Code: 1064. You have...	0.00076 sec
12	18:25:45	CREATE VIEW employee_location AS SELECT emp_id, first_name, last_name, coun...	0 row(s) affected	0.102 sec
13	18:26:12	SELECT * FROM employee_location LIMIT 0, 1000	12 row(s) returned	0.025 sec / 0.00027...
14	18:29:32	SELECT * FROM emp_record_table WHERE emp_id IN (SELECT emp_id FROM emp...	8 row(s) returned	0.055 sec / 0.00036...

### Result Grid Zoom:

EMP_ID	FIRST_NAME	LAST_NAME	GENDER	ROLE	DEPT	EXP	COUNTRY	CONTINENT	SALARY	EMP_RATING	MANAGER_ID	PROJ_ID
E001	Arthur	Black	M	PRESIDENT	ALL	20	USA	NORTH AMERICA	16500	5	NULL	NULL
E005	Eric	Hoffman	M	LEAD DATA SCIENTIST	FINANCE	11	USA	NORTH AMERICA	8500	3	E103	P105
E010	William	Butler	M	LEAD DATA SCIENTIST	AUTOMOTIVE	12	FRANCE	EUROPE	9000	2	E428	P204
E083	Patrick	Voltz	M	MANAGER	HEALTHCARE	15	USA	NORTH AMERICA	9500	5	E001	NULL
E103	Emily	Grove	F	MANAGER	FINANCE	14	CANADA	NORTH AMERICA	10500	4	E001	NULL
E428	Pete	Allen	M	MANAGER	AUTOMOTIVE	14	GERMANY	EUROPE	11000	4	E001	NULL
E583	Janet	Hale	F	MANAGER	RETAIL	14	COLOMBIA	SOUTH AMERICA	10000	2	E001	NULL
E612	Tracy	Norris	F	MANAGER	RETAIL	13	INDIA	ASIA	8500	4	E001	NULL

### 13. Input SQL Query:

```
CREATE DEFINER='root'@'localhost' PROCEDURE `exp_greater_than_three`()
BEGIN
select * from emp_record_table WHERE exp > 3;
END
```

```
CALL employee.exp_greater_than_three();
```

### Outputs:

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' panel is open, showing a tree view of databases. The 'employee' database is selected, and the 'exp\_greater\_than\_three' stored procedure is highlighted. The main editor window displays the SQL code for creating the procedure:

```
1 CREATE DEFINER='root'@'localhost' PROCEDURE `exp_greater_than_three`()
2 BEGIN
3   select * from emp_record_table WHERE exp > 3;
4 END
```

The status bar at the bottom indicates the current routine is 'exp\_greater\_than\_three' and it is a 'Routine'.



The screenshot shows a database IDE with a dark theme. On the left, a 'SCHEMAS' panel lists a database structure including 'COMPANY', 'datetime\_db', 'employee' (with tables like 'data\_science...', 'emp\_record...', 'proj\_table'), 'Views', 'Stored Procedure...', 'exp\_greater...', 'Functions', 'employee\_manag...', 'employees\_datab...', 'employees\_db', and 'library\_db'. The main editor displays a series of SQL queries:
 

- Line 19: `SELECT emp_id, first_name, last_name, role, dept, emp_rating, max(emp_rating)`
- Line 20: `Over (partition by dept) AS max_emp_rating`
- Line 21: `FROM emp_record_table;`
- Line 22: `Select role, MIN(salary) AS min_salary,`
- Line 23: `MAX(salary) AS max_salary`
- Line 24: `FROM emp_record_table`
- Line 25: `GROUP BY ROLE;`
- Line 26: `SELECT emp_id, first_name, last_name, exp, DENSE_RANK() OVER (ORDER BY EXP DESC) AS experience_rank`
- Line 27: `FROM emp_record_table;`
- Line 28: `CREATE VIEW employee_location AS`
- Line 29: `SELECT emp_id, first_name, last_name, country, salary FROM emp_record_table WHERE salary > 6000;`
- Line 30: `SELECT * FROM employee_location;`
- Line 31: `SELECT * FROM emp_record_table WHERE emp_id`
- Line 32: `IN (SELECT emp_id FROM emp_record_table WHERE exp > 10);`
- Line 33: `CALL employee.exp_greater_than_three();`

 Below the queries, a 'Result Grid' shows data for 'Result 21'. The grid has columns: EMP\_ID, FIRST\_NAME, LAST\_NAME, GENDER, ROLE, DEPT, EXP, COUNTRY, CONTINENT, SALARY, EMP\_RATING. Visible rows include:
 

- Row 1: E001, Arthur, Black, M, PRESIDENT, ALL, 20, USA, NORTH AMERICA, 16500, 5
- Row 2: E005, Eric, Hoffman, M, LEAD DATA SCIENTIST, FINANCE, 11, USA, NORTH AMERICA, 8500, 3
- Row 3: E010, William, Butler, M, LEAD DATA SCIENTIST, AUTOMOTIVE, 12, FRANCE, EUROPE, 9000, 2

 At the bottom, an 'Action Output' panel shows a log of executed actions:
 

- 14: 18:29:32 - SELECT \* FROM emp\_record\_table WHERE emp\_id IN (SELECT emp\_id FROM emp... - 8 row(s) returned - 0.055 sec / 0.00036...
- 15: 18:53:39 - Apply changes to job\_profile
- 16: 18:59:30 - Apply changes to exp\_greater\_than\_three - Changes applied
- 17: 19:00:52 - CALL employee.exp\_greater\_than\_three() - 15 row(s) returned - 0.018 sec / 0.000033...

### Result Grid Zoom:

EMP_ID	FIRST_NAME	LAST_NAME	GENDER	ROLE	DEPT	EXP	COUNTRY	CONTINENT	SALARY	EMP_RATING	MANAGER_ID	PROJ_ID
E001	Arthur	Black	M	PRESIDENT	ALL	20	USA	NORTH AMERICA	16500	5	NULL	NULL
E005	Eric	Hoffman	M	LEAD DATA SCIENTIST	FINANCE	11	USA	NORTH AMERICA	8500	3	E103	P105
E010	William	Butler	M	LEAD DATA SCIENTIST	AUTOMOTIVE	12	FRANCE	EUROPE	9000	2	E428	P204
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
E103	Emily	Grove	F	MANAGER	FINANCE	14	CANADA	NORTH AMERICA	10500	4	E001	NULL
E204	Karene	Nowak	F	SENIOR DATA SCIENTIST	AUTOMOTIVE	8	GERMANY	EUROPE	7500	5	E428	P204
E245	Nian	Zhen	M	SENIOR DATA SCIENTIST	RETAIL	6	CHINA	ASIA	6500	2	E583	P109
E260	Roy	Collins	M	SENIOR DATA SCIENTIST	RETAIL	7	INDIA	ASIA	7000	3	E583	NA
E403	Steve	Hoffman	M	ASSOCIATE DATA SCIENTIST	FINANCE	4	USA	NORTH AMERICA	5000	3	E103	P105
E428	Pete	Allen	M	MANAGER	AUTOMOTIVE	14	GERMANY	EUROPE	11000	4	E001	NULL
E505	Chad	Wilson	M	ASSOCIATE DATA SCIENTIST	HEALTHCARE	5	CANADA	NORTH AMERICA	5000	2	E083	P103
E583	Janet	Hale	F	MANAGER	RETAIL	14	COLOMBIA	SOUTH AMERICA	10000	2	E001	NULL
E612	Tracy	Norris	F	MANAGER	RETAIL	13	INDIA	ASIA	8500	4	E001	NULL

### 14. Input SQL Query:

```

CREATE DEFINER=`root`@`localhost` PROCEDURE `job_profile`()
BEGIN
select * , case
WHEN exp<=2 then 'JUNIOR DATA SCIENTIST'
WHEN exp>2 and exp<=5 then 'ASSOCIATE DATA SCIENTIST'
WHEN exp>5 and exp<=10 then 'SENIOR DATA SCIENTIST'
WHEN exp>10 and exp<=12 then 'LEAD DATA SCIENTIST'
WHEN exp>12 and exp<=16 then 'MANAGER'
end as job_prof_match FROM emp_record_table;
END

call employee.job_profile();

```

## Outputs:

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane is open, showing a tree view of databases. The 'employee' database is selected, and the 'job\_profile' stored procedure is highlighted. The main editor displays the SQL code for creating the procedure:

```
1 CREATE DEFINER='root'@'localhost' PROCEDURE `job_profile`()
2 BEGIN
3   select *, case
4     WHEN exp<=2 then 'JUNIOR DATA SCIENTIST'
5     WHEN exp>2 and exp<=5 then 'ASSOCIATE DATA SCIENTIST'
6     WHEN exp>5 and exp<=10 then 'SENIOR DATA SCIENTIST'
7     WHEN exp>10 and exp<=12 then 'LEAD DATA SCIENTIST'
8     WHEN exp>12 and exp<=16 then 'MANAGER'
9   end as job_prof_match FROM emp_record_table;
10 END
```

The status bar at the bottom indicates the routine is a 'Routine' and provides 'Apply' and 'Revert' buttons.

The screenshot shows the MySQL Workbench interface with a SQL script being executed. The script includes several queries and a call to the 'job\_profile' procedure. The 'Result Grid' is displayed at the bottom, showing the output of the queries.

**Script Content:**

```
20 Over (partition by dept) AS max_emp_rating
21 FROM emp_record_table;
22 Select role, MIN(salary) AS min_salary,
23 MAX(salary) AS max_salary
24 FROM emp_record_table
25 GROUP BY ROLE;
26 Select emp_id, first_name, last_name, exp, DENSE_RANK() OVER (ORDER BY EXP DESC) AS experience_rank
27 FROM emp_record_table;
28 CREATE VIEW employee_location AS
29 SELECT emp_id, first_name, last_name, country, salary FROM emp_record_table WHERE salary > 6000;
30 SELECT * FROM employee_location;
31 SELECT * FROM emp_record_table WHERE emp_id
32 IN (SELECT emp_id FROM emp_record_table WHERE exp > 10);
33 CALL employee.exp_greater_than_three();
34 call employee.job_profile();
```

**Result Grid:**

EMP_ID	FIRST_NAME	LAST_NAME	GENDER	ROLE	DEPT	EXP	COUNTRY	CONTINENT	SALARY	EMP_RATING
E001	Arthur	Black	M	PRESIDENT	ALL	20	USA	NORTH AMERICA	16500	5
E005	Eric	Hoffman	M	LEAD DATA SCIENTIST	FINANCE	11	USA	NORTH AMERICA	8500	3
E010	William	Butler	M	LEAD DATA SCIENTIST	AUTOMOTIVE	12	FRANCE	EUROPE	9000	2

**Action Output:**

Time	Action	Response	Duration / Fetch Time
17 19:00:52	CALL employee.exp_greater_than_three()	15 row(s) returned	0.018 sec / 0.000033...
18 19:04:40	Apply changes to job_profile	Changes applied	
19 19:05:11	Apply changes to job_profile	Changes applied	
20 19:06:05	call employee.job_profile()	19 row(s) returned	0.0095 sec / 0.00003...

## Result Grid Zoom:

EMP_ID	FIRST_NAME	LAST_NAME	GENDER	ROLE	DEPT	EXP	COUNTRY	CONTINENT	SALARY	EMP_RATING	MANAGER_ID	PROJ_ID	job_prof_match
E001	Arthur	Black	M	PRESIDENT	ALL	20	USA	NORTH AMERICA	16500	5	NULL	NULL	NULL
E005	Eric	Hoffman	M	LEAD DATA SCIENTIST	FINANCE	11	USA	NORTH AMERICA	8500	3	E103	P105	LEAD DATA SCIENTIST
E010	William	Butler	M	LEAD DATA SCIENTIST	AUTOMOTIVE	12	FRANCE	EUROPE	9000	2	E428	P204	LEAD DATA SCIENTIST
E052	Dianna	Wilson	F	SENIOR DATA SCIENTIST	HEALTHCARE	6	CANADA	NORTH AMERICA	5500	5	E083	P103	SENIOR DATA SCIENTIST
E057	Dorothy	Wilson	F	SENIOR DATA SCIENTIST	HEALTHCARE	9	USA	NORTH AMERICA	7700	1	E083	P302	SENIOR DATA SCIENTIST
E083	Patrick	Voltz	M	MANAGER	HEALTHCARE	15	USA	NORTH AMERICA	9500	5	E001	NULL	MANAGER
E103	Emily	Grove	F	MANAGER	FINANCE	14	CANADA	NORTH AMERICA	10500	4	E001	NULL	MANAGER
E204	Karene	Nowak	F	SENIOR DATA SCIENTIST	AUTOMOTIVE	8	GERMANY	EUROPE	7500	5	E428	P204	SENIOR DATA SCIENTIST
E245	Nian	Zhen	M	SENIOR DATA SCIENTIST	RETAIL	6	CHINA	ASIA	6500	2	E583	P109	SENIOR DATA SCIENTIST
E260	Roy	Collins	M	SENIOR DATA SCIENTIST	RETAIL	7	INDIA	ASIA	7000	3	E583	NA	SENIOR DATA SCIENTIST
E403	Steve	Hoffman	M	ASSOCIATE DATA SCIENTIST	FINANCE	4	USA	NORTH AMERICA	5000	3	E103	P105	ASSOCIATE DATA SCIENTIST
E428	Pete	Allen	M	MANAGER	AUTOMOTIVE	14	GERMANY	EUROPE	11000	4	E001	NULL	MANAGER
E478	David	Smith	M	ASSOCIATE DATA SCIENTIST	RETAIL	3	COLOMBIA	SOUTH AMERICA	4000	4	E583	P109	ASSOCIATE DATA SCIENTIST
E505	Chad	Wilson	M	ASSOCIATE DATA SCIENTIST	HEALTHCARE	5	CANADA	NORTH AMERICA	5000	2	E083	P103	ASSOCIATE DATA SCIENTIST
E532	Claire	Brennan	F	ASSOCIATE DATA SCIENTIST	AUTOMOTIVE	3	GERMANY	EUROPE	4300	1	E428	P204	ASSOCIATE DATA SCIENTIST
E583	Janet	Hale	F	MANAGER	RETAIL	14	COLOMBIA	SOUTH AMERICA	10000	2	E001	NULL	MANAGER
E612	Tracy	Norris	F	MANAGER	RETAIL	13	INDIA	ASIA	8500	4	E001	NULL	MANAGER
E620	Katrina	Allen	F	JUNIOR DATA SCIENTIST	RETAIL	2	INDIA	ASIA	3000	1	E612	P406	JUNIOR DATA SCIENTIST
E640	Jenifer	Jhones	F	JUNIOR DATA SCIENTIST	RETAIL	1	COLOMBIA	SOUTH AMERICA	2800	4	E612	P406	JUNIOR DATA SCIENTIST

## 15. Input SQL Query:

```

SELECT * FROM emp_record_table ORDER BY first_name;
DESCRIBE emp_record_table;
ALTER TABLE emp_record_table modify first_name VARCHAR(50);
CREATE INDEX fname_index ON emp_record_table(first_name);
SHOW INDEXES FROM emp_record_table;
SELECT * from emp_record_table WHERE first_name = "Eric";

```

## Output:

The screenshot shows the MySQL Workbench interface with the following components:

- Navigation Panel (Left):** Shows the database schema structure including COMPANIES, datettime\_db, employee (Tables), data\_science..., emp\_record..., proj\_table, Views, Stored Procedure..., exp\_greater..., job\_profile, Functions, employee\_manag..., employees\_datab..., and employees\_db.
- SQL Editor (Center):** Contains the following SQL queries:
 

```

26 SELECT emp_id, first_name, last_name, exp, DENSE_RANK() OVER (ORDER BY EXP DESC) AS experience_rank
27 FROM emp_record_table;
28 CREATE VIEW employee_location AS
29 SELECT emp_id, first_name, last_name, country, salary FROM emp_record_table WHERE salary > 6000;
30 SELECT * FROM employee_location;
31 SELECT * FROM emp_record_table WHERE emp_id
32 IN (SELECT emp_id FROM emp_record_table WHERE exp > 10);
33 CALL employee.exp_greater_than_three();
34 call employee.job_profile();
35 SELECT * FROM emp_record_table ORDER BY first_name;
36 DESCRIBE emp_record_table;
37 ALTER TABLE emp_record_table modify first_name VARCHAR(50);
38 CREATE INDEX fname_index ON emp_record_table(first_name);
39 SHOW INDEXES FROM emp_record_table;
40 SELECT * from emp_record_table WHERE first_name = "Eric";
41

```
- Result Grid (Bottom):** Displays the results of the SQL queries. The first query (SELECT \* FROM emp\_record\_table ORDER BY first\_name) shows a single row for Eric Hoffman (E005) as a Lead Data Scientist in Finance, USA, North America, with a salary of 8500 and an employee rating of 3. The second query (DESCRIBE emp\_record\_table) shows the table structure with columns: EMP\_ID, first\_name, LAST\_NAME, GENDER, ROLE, DEPT, EXP, COUNTRY, CONTINENT, SALARY, EMP\_RATING, and MANAGER\_ID.
- Action Output (Bottom):** Shows the execution details of the queries, including the time taken and the response received. The last query (SELECT \* from emp\_record\_table WHERE first\_name = "Eric") returned 1 row(s) in 0.0036 seconds.

## 16. Input SQL Query:

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

## Output:

The screenshot displays the MySQL Workbench interface. The left sidebar shows the 'SCHEMAS' tree with 'employee' expanded, showing 'Tables' and 'Views'. The main editor shows a SQL script with the following queries:

```
35 SELECT * FROM emp_record_table ORDER BY first_name;
36 DESCRIBE emp_record_table;
37 ALTER TABLE emp_record_table modify first_name VARCHAR(50);
38 CREATE INDEX fname_index ON emp_record_table(first_name);
39 SHOW INDEXES FROM emp_record_table;
40 SELECT * from emp_record_table WHERE first_name = "Eric";
41 SELECT
42     EMP_ID,
43     FIRST_NAME,
44     LAST_NAME,
45     SALARY,
46     EMP_RATING,
47     (0.05 * SALARY) * (EMP_RATING) AS BONUS
48 FROM
49     emp_record_table;
```

The 'Result Grid' shows the output of the last query (Result 28). It contains 19 rows of data with columns: EMP\_ID, FIRST\_NAME, LAST\_NAME, SALARY, EMP\_RATING, and BONUS.

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
ERIC9	Pinna	Willena	5500	6	1975.00

The 'Action Output' pane at the bottom shows the execution log for the last three queries:

	Time	Action	Response	Duration / Fetch Time
25	19:10:36	CREATE INDEX fname_index ON emp_record_table(first_name)	0 row(s) affected Records:...	0.040 sec
26	19:10:45	SHOW INDEXES FROM emp_record_table	1 row(s) returned	0.0077 sec / 0.00002...
27	19:11:12	SELECT * from emp_record_table WHERE first_name = "Eric" LIMIT 0, 1000	1 row(s) returned	0.0036 sec / 0.00001...
28	19:12:49	SELECT EMP_ID, FIRST_NAME, LAST_NAME, SALARY, EMP_RATING,...	19 row(s) returned	0.0045 sec / 0.0000...

### Result Grid Zoom:

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
E083	Patrick	Voltz	9500	5	2375.00
E103	Emily	Grove	10500	4	2100.00
E204	Karene	Nowak	7500	5	1875.00
E245	Nian	Zhen	6500	2	650.00
E260	Roy	Collins	7000	3	1050.00
E403	Steve	Hoffman	5000	3	750.00
E428	Pete	Allen	11000	4	2200.00
E478	David	Smith	4000	4	800.00
E505	Chad	Wilson	5000	2	500.00
E532	Claire	Brennan	4300	1	215.00
E583	Janet	Hale	10000	2	1000.00
E612	Tracy	Norris	8500	4	1700.00
E620	Katrina	Allen	3000	1	150.00
E640	Jenifer	Jhones	2800	4	560.00

### 17. Input SQL Query:

```
SELECT * , avg(salary) over (partition by country) AS country_wise,  
avg(salary) over (partition by continent) AS continent_wise  
FROM emp_record_table;
```



Output:

Local Instance O.G.

MySQL Model\*

EER Diagram

AdministrationSchemasLesson\_Practice\_File\*Lesson\_Practice\_File\_2CEP Amit Gengaje SQL Script\*job\_profile - Routineexp\_greater\_than\_three - Routine

SCHEMAS

Filter objects

COMPANY

datetime\_db

employee

Tables

data\_science...

emp\_record\_...

proj\_table

Views

Stored Procedu...

exp\_greater\_...

job\_profile

Functions

employee\_manag...

employees\_datab...

employees\_db

38CREATE INDEX fname\_index ON emp\_record\_table(first\_name);

39SHOW INDEXES FROM emp\_record\_table;

40SELECT \* from emp\_record\_table WHERE first\_name = "Eric";

41SELECT

42EMP\_ID,

43FIRST\_NAME,

44LAST\_NAME,

45SALARY,

46EMP\_RATING,

47(0.05 \* SALARY) \* (EMP\_RATING) AS BONUS

48FROM

49emp\_record\_table;

50SELECT \* , avg(salary) over (partition by country) AS country\_wise,

51avg(salary) over (partition by continent) AS continent\_wise

52FROM emp\_record\_table;

53

100%

1:50

Result Grid

Filter Rows: Search

Export:

Object Info

Session

Procedure: job\_profile

EMP_ID	first_name	LAST_NAME	GENDER	ROLE	DEPT	EXP	COUNTRY	CONTINENT	SALARY	EMP_RATING
E245	Nian	Zhen	M	SENIOR DATA SCIENTIST	RETAIL	6	CHINA	ASIA	6500	2
E260	Roy	Collins	M	SENIOR DATA SCIENTIST	RETAIL	7	INDIA	ASIA	7000	3
E612	Tracy	Norris	F	MANAGER	RETAIL	13	INDIA	ASIA	8500	4
E620	Katrina	Allen	F	JUNIOR DATA SCIENTIST	RETAIL	2	INDIA	ASIA	3000	1

Result 29

Read Only

Action Output

	Time	Action	Response	Duration / Fetch Time
26	19:10:45	SHOW INDEXES FROM emp_record_table	1 row(s) returned	0.0077 sec / 0.00002...
27	19:11:12	SELECT * from emp_record_table WHERE first_name = "Eric" LIMIT 0, 1000	1 row(s) returned	0.0036 sec / 0.00001...
28	19:12:49	SELECT EMP_ID, FIRST_NAME, LAST_NAME, SALARY, EMP_RATING,...	19 row(s) returned	0.0045 sec / 0.0000...
29	19:15:07	SELECT * , avg(salary) over (partition by country) AS country_wise, avg(salary) ov...	19 row(s) returned	0.026 sec / 0.000038...

Result Grid Zoom:

EMP_ID	first_name	LAST_NAME	GENDER	ROLE	DEPT	EXP	COUNTRY	CONTINENT	SALARY	EMP_RATING	MANAGER_ID	PROJ_ID	country_wise	continent_wise
E245	Nian	Zhen	M	SENIOR DATA SCIENTIST	RETAIL	6	CHINA	ASIA	6500	2	E583	P109	6500.0000	6250.0000
E260	Roy	Collins	M	SENIOR DATA SCIENTIST	RETAIL	7	INDIA	ASIA	7000	3	E583	NA	6166.6667	6250.0000
E612	Tracy	Norris	F	MANAGER	RETAIL	13	INDIA	ASIA	8500	4	E001	NULL	6166.6667	6250.0000
E620	Katrina	Allen	F	JUNIOR DATA SCIENTIST	RETAIL	2	INDIA	ASIA	3000	1	E612	P406	6166.6667	6250.0000
E010	William	Butler	M	LEAD DATA SCIENTIST	AUTOMOTIVE	12	FRANCE	EUROPE	9000	2	E428	P204	9000.0000	7950.0000
E204	Karene	Nowak	F	SENIOR DATA SCIENTIST	AUTOMOTIVE	8	GERMANY	EUROPE	7500	5	E428	P204	7600.0000	7950.0000
E428	Pete	Allen	M	MANAGER	AUTOMOTIVE	14	GERMANY	EUROPE	11000	4	E001	NULL	7600.0000	7950.0000
E532	Claire	Brennan	F	ASSOCIATE DATA SCIENTIST	AUTOMOTIVE	3	GERMANY	EUROPE	4300	1	E428	P204	7600.0000	7950.0000
E052	Dianna	Wilson	F	SENIOR DATA SCIENTIST	HEALTHCARE	6	CANADA	NORTH AMERICA	5500	5	E083	P103	7000.0000	8525.0000
E103	Emily	Grove	F	MANAGER	FINANCE	14	CANADA	NORTH AMERICA	10500	4	E001	NULL	7000.0000	8525.0000
E505	Chad	Wilson	M	ASSOCIATE DATA SCIENTIST	HEALTHCARE	5	CANADA	NORTH AMERICA	5000	2	E083	P103	7000.0000	8525.0000
E001	Arthur	Black	M	PRESIDENT	ALL	20	USA	NORTH AMERICA	16500	5	NULL	NULL	9440.0000	8525.0000
E005	Eric	Hoffman	M	LEAD DATA SCIENTIST	FINANCE	11	USA	NORTH AMERICA	8500	3	E103	P105	9440.0000	8525.0000
E057	Dorothy	Wilson	F	SENIOR DATA SCIENTIST	HEALTHCARE	9	USA	NORTH AMERICA	7700	1	E083	P302	9440.0000	8525.0000
E083	Patrick	Voltz	M	MANAGER	HEALTHCARE	15	USA	NORTH AMERICA	9500	5	E001	NULL	9440.0000	8525.0000
E403	Steve	Hoffman	M	ASSOCIATE DATA SCIENTIST	FINANCE	4	USA	NORTH AMERICA	5000	3	E103	P105	9440.0000	8525.0000
E478	David	Smith	M	ASSOCIATE DATA SCIENTIST	RETAIL	3	COLOMBIA	SOUTH AMERICA	4000	4	E583	P109	5600.0000	5600.0000
E583	Janet	Hale	F	MANAGER	RETAIL	14	COLOMBIA	SOUTH AMERICA	10000	2	E001	NULL	5600.0000	5600.0000
E640	Jenifer	Jhones	F	JUNIOR DATA SCIENTIST	RETAIL	1	COLOMBIA	SOUTH AMERICA	2800	4	E612	P406	5600.0000	5600.0000