COURSE END PROJECT -SIMPLILEARN

SQL Training – Qtech Science Employee Performance Mapping

SUBMITTED BY SREEJITH C SRINIVAS

Tasks 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.
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

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.
2. Write a query to list only those employees who have someone reporting to them. Also, show the number of reporters (including the President).
3. Write a query to list down all the employees from the healthcare and finance departments using union. Take data from the employee record table.
4. 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.
5. Write a query to calculate the minimum and the maximum salary of the employees in each role. Take data from the employee record table.
6. Write a query to assign ranks to each employee based on their experience. Take data from the employee record table.
7. 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.
8. Write a nested query to find employees with experience of more than ten years. Take data from the employee record table.
9. 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.
10. 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'.

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.
2. 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).
3. Write a query to calculate the average salary distribution based on the continent and country. Take data from the employee record table.

Code:

create database ds;

use ds;

create table e\_rec(

emp\_id varchar(4) not null,

First\_name varchar(45) not null,

last\_name varchar(45) not null,

Gender varchar(1) not null,

role\_ varchar(45) not null,

dept varchar(45) not null,

exp int not null,

Country varchar(45) not null,

continent varchar(45) not null,

salary int not null,

emp\_rating int not null,

Manager\_id varchar(10),

Proj\_id varchar(10),

Primary key(emp\_id));

create table pro(

project\_id varchar(25) not null,

project\_name varchar(45) not null,

domain varchar(45) not null,

start\_date date not null,

closure\_date date not null,

dev\_qtr varchar(4) not null,

P\_status varchar(45) not null,

Primary key(project\_id));

create table team(

emp\_id varchar(4) not null,

first\_name varchar(45) not null,

last\_name varchar(45) not null,

Gender varchar(1) not null,

role\_ varchar(45) not null,

dept varchar(45) not null,

exp int not null,

country varchar(45) not null,

continent varchar(45) not null,

primary key(emp\_id));

Select emp\_id, first\_name, last\_name, gender, dept from ds.e\_rec;

Select emp\_id, first\_name, last\_name, gender, dept, emp\_rating from ds.e\_rec

where emp\_rating<2;

Select emp\_id, first\_name, last\_name, gender, dept, emp\_rating from ds.e\_rec

where emp\_rating<4;

Select emp\_id, first\_name, last\_name, gender, dept, emp\_rating from ds.e\_rec

where emp\_rating between 2 AND 4;

Select concat(first\_name, ' ', last\_name) AS NAME from ds.e\_rec;

Select emp\_id, first\_name, last\_name, role\_, dept from ds.e\_rec

where role\_ not in ("Junior Data Scientist", "Associate Data Scientist");

Select count(\*) from ds.e\_rec;

Select first\_name, last\_name, dept from ds.e\_rec

where dept = "Healthcare"

union

Select first\_name, last\_name, dept from ds.e\_rec

where dept = "Finance";

SET sql\_mode = '';

Select emp\_id, first\_name, last\_name, role\_, dept, emp\_rating, max(emp\_rating) as max\_ratings

from ds.e\_rec group by dept;

Select min(salary) as min\_salary, max(salary) as max\_salary from ds.e\_rec;

Select row\_number() over (order by exp desc) as emp\_ranks, emp\_id, first\_name, last\_name, dept from ds.e\_rec;

create view emp\_sal as

select first\_name, last\_name, country from ds.e\_rec where salary>6000;

select first\_name, last\_name from ds.e\_rec

where not exists (select 1 from ds.e\_rec where exp<0) and exp>=10;

delimiter //

create procedure Get\_emp\_details()

begin

select \* from ds.e\_rec where exp>3;

end//

Call Get\_emp\_details()

delimiter //

create function emp\_role(exp int)

returns varchar(50)

deterministic

begin

declare job\_role varchar(50);

if exp <=2 then

set job\_role= 'Junior Data Scientist';

elseif exp<=5 then

set job\_role = 'Associate Data Scientist';

elseif exp<=10 then

set job\_role = 'Senior Data Scientist';

elseif exp<=12 then

set job\_role = 'Lead Data Scientist';

else

set job\_role = 'Manager';

end if;

return (job\_role);

end//

select exp, emp\_role(exp) from ds.team;

create index first\_name\_indx on e\_rec(first\_name(20));

Select \* from e\_rec where first\_name = 'Eric';

Select emp\_id, first\_name, last\_name, Gender, role\_, salary, (salary + 0.05\*salary\*emp\_rating)

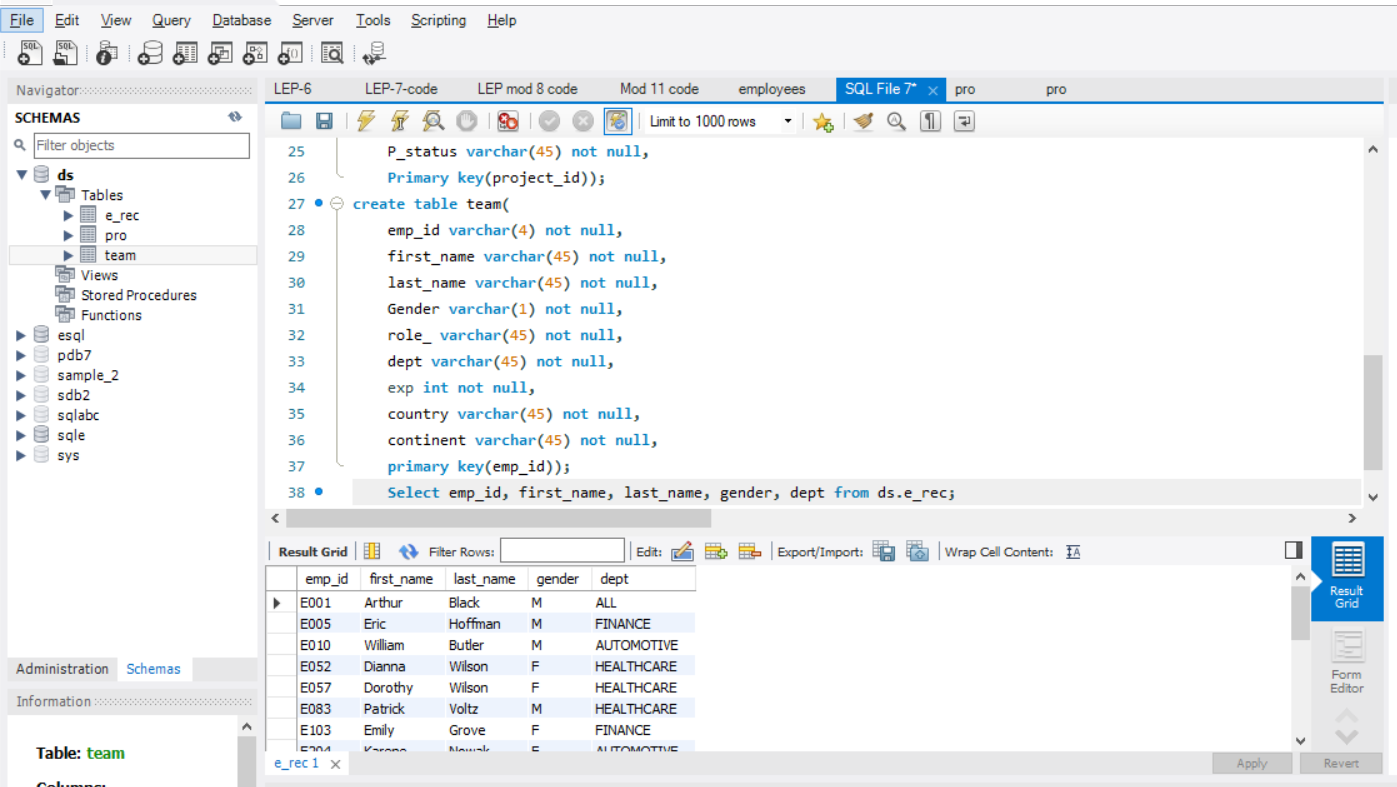
as updated\_salary from ds.e\_rec;

Select emp\_id, first\_name, last\_name, salary, country, continent, avg(salary) over(partition by country) as Country\_Avg\_Sal,

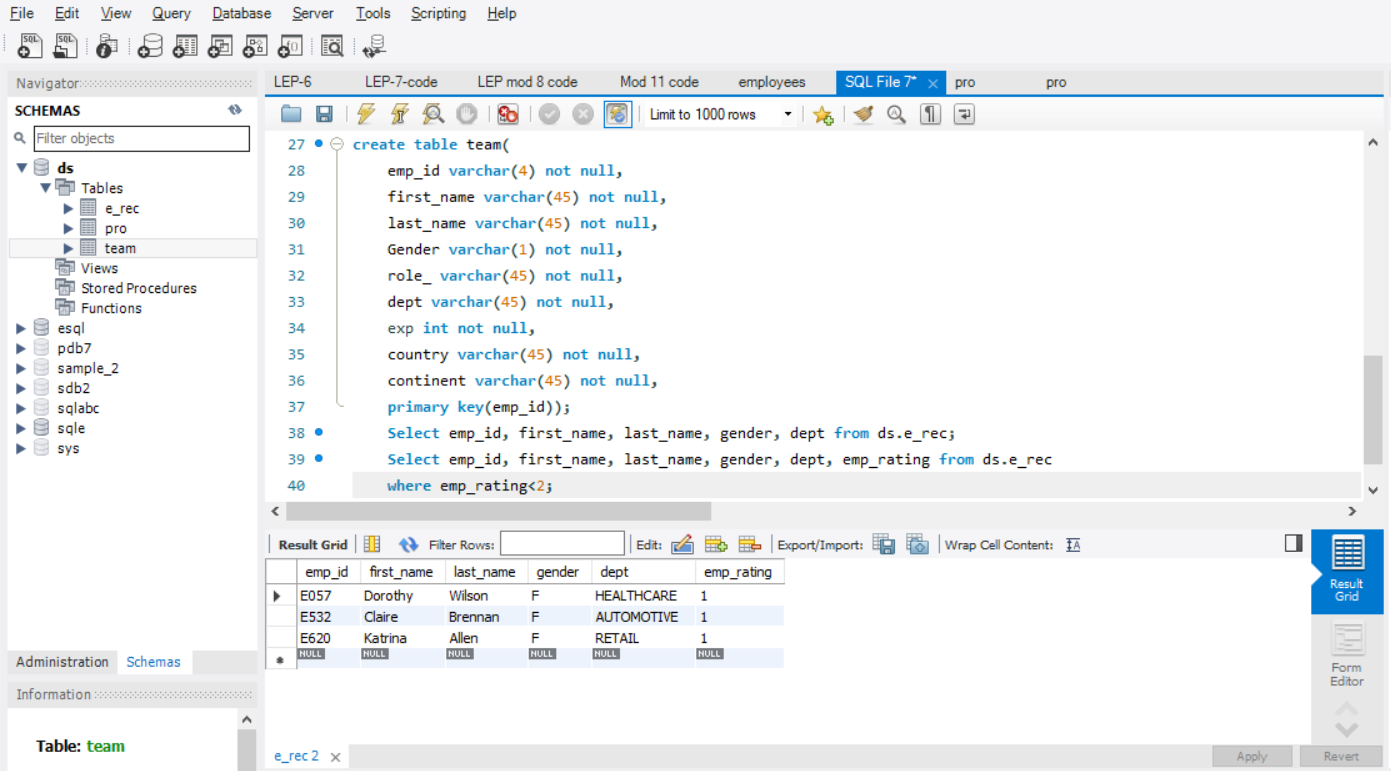
avg(salary) over(partition by continent) as Continent\_Avg\_Sal from ds.e\_rec;

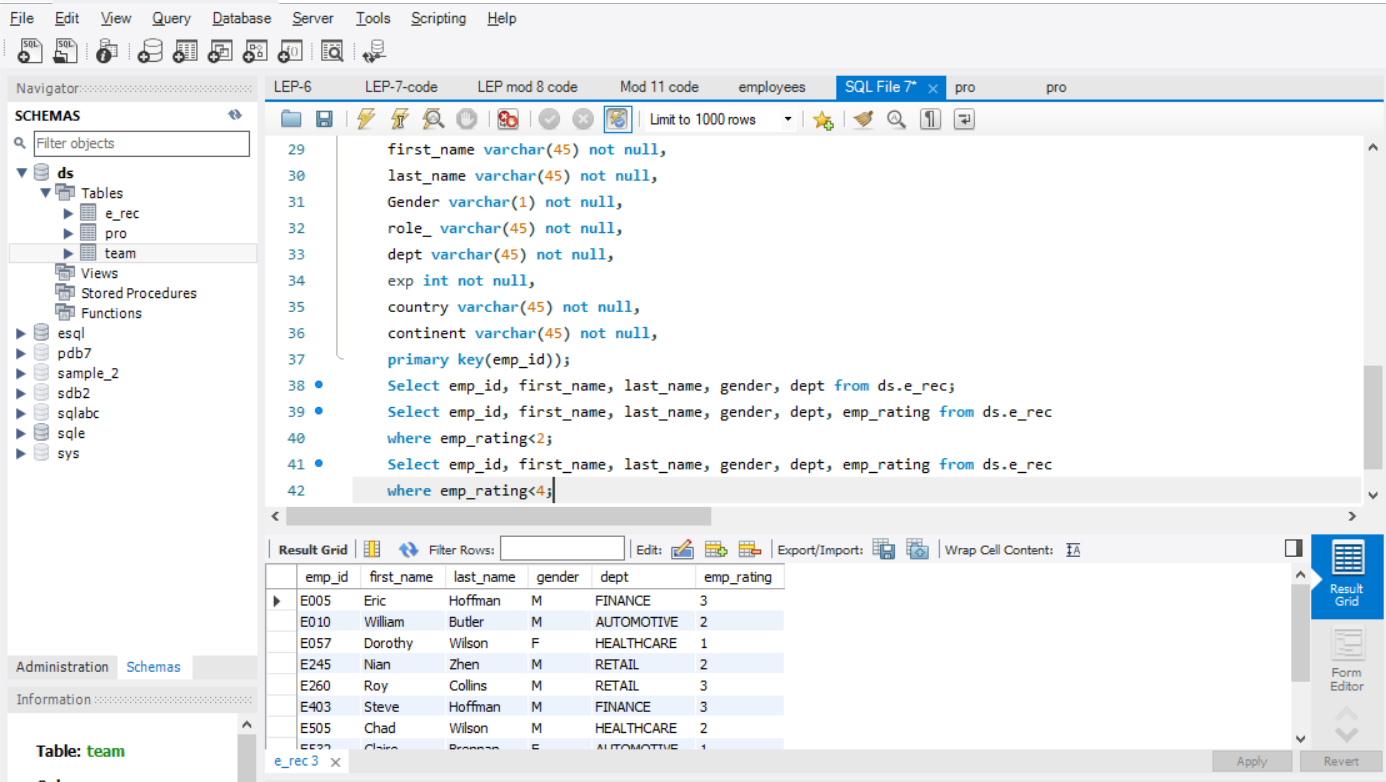
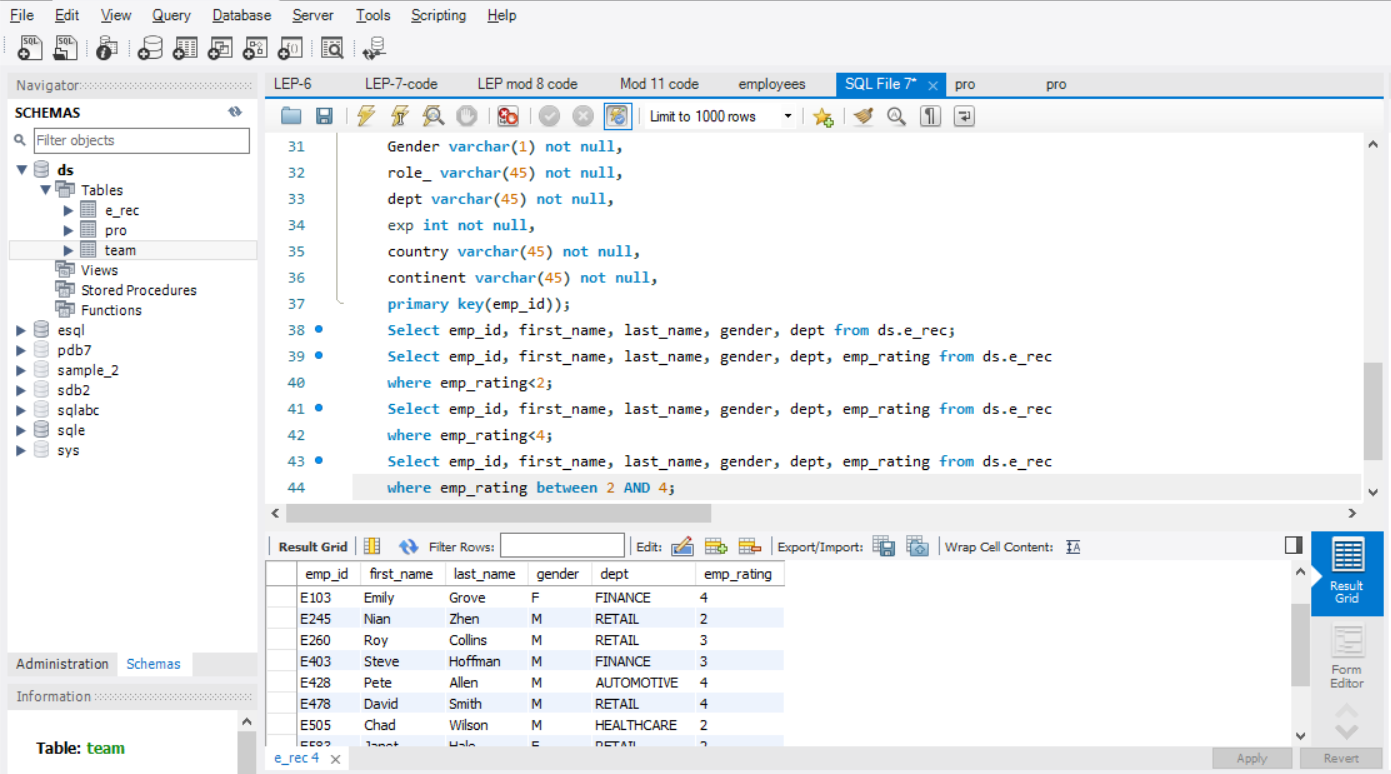
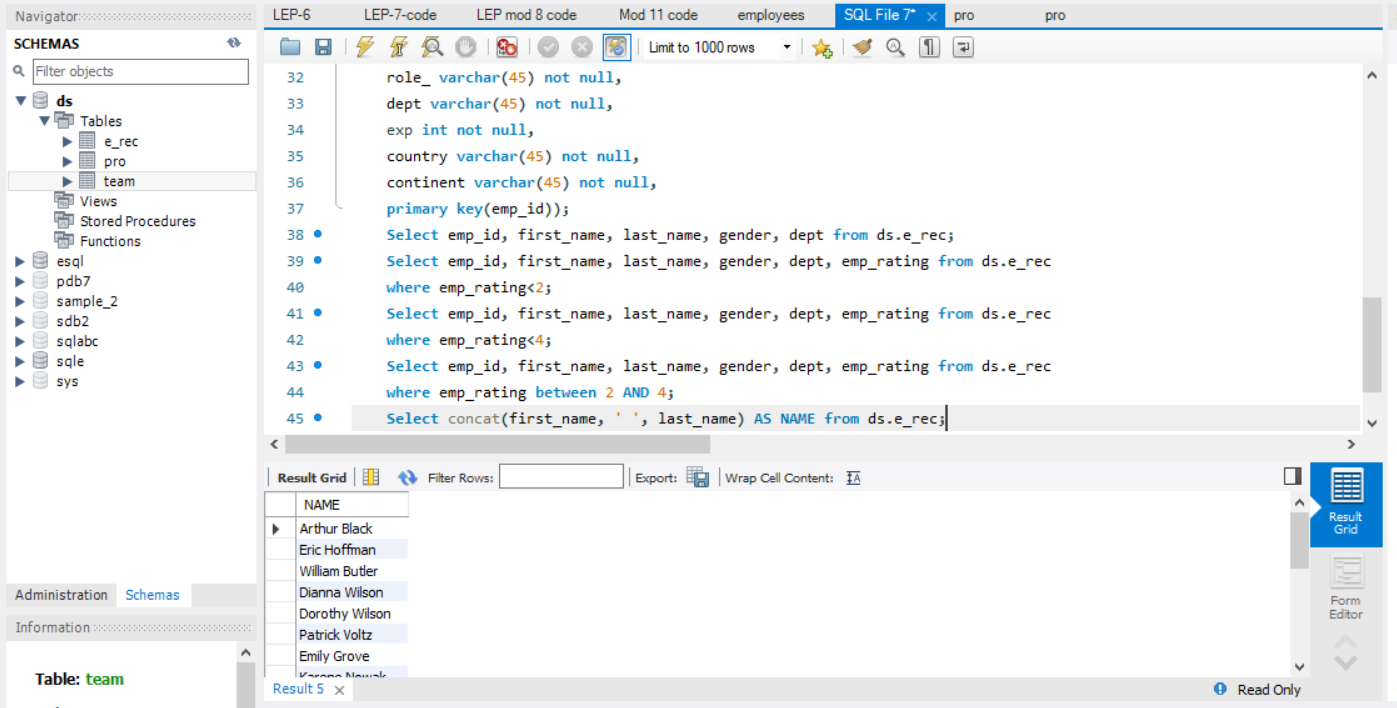
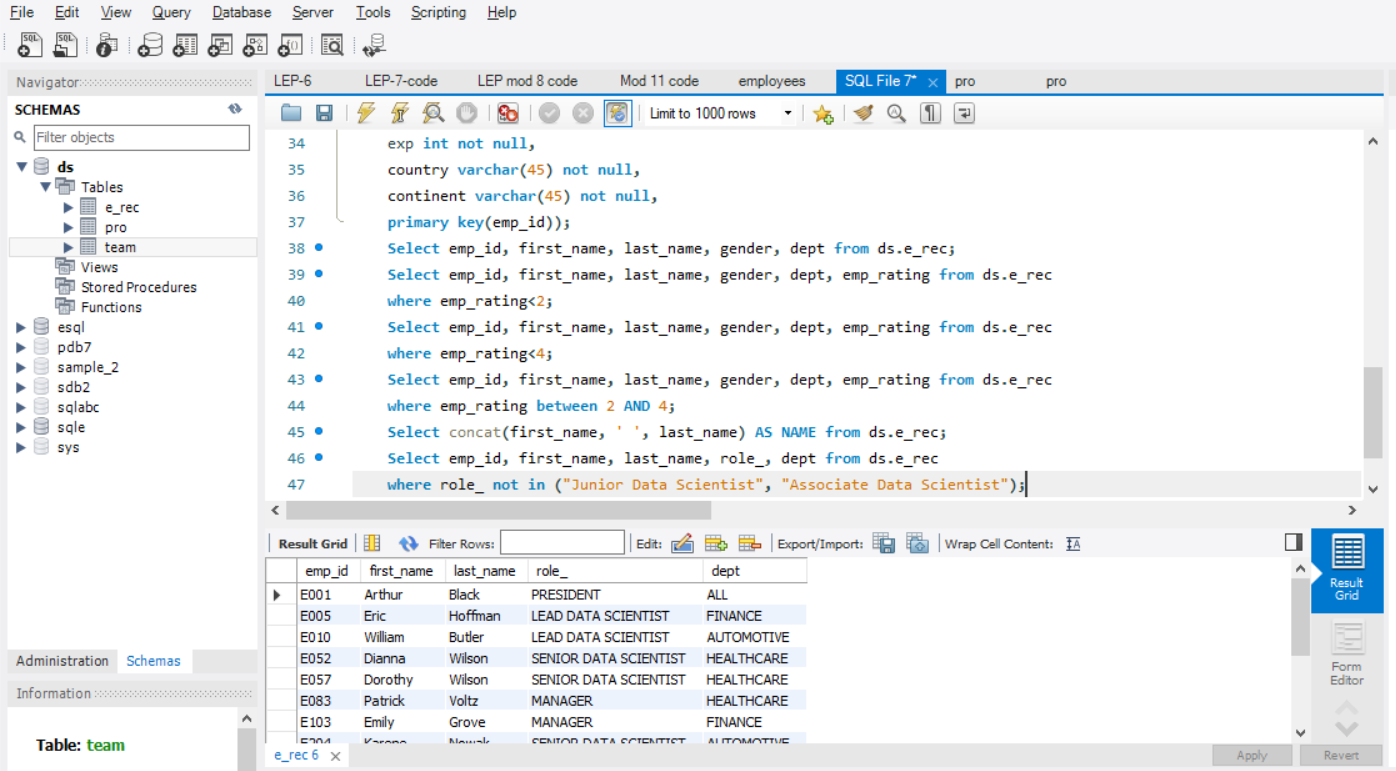
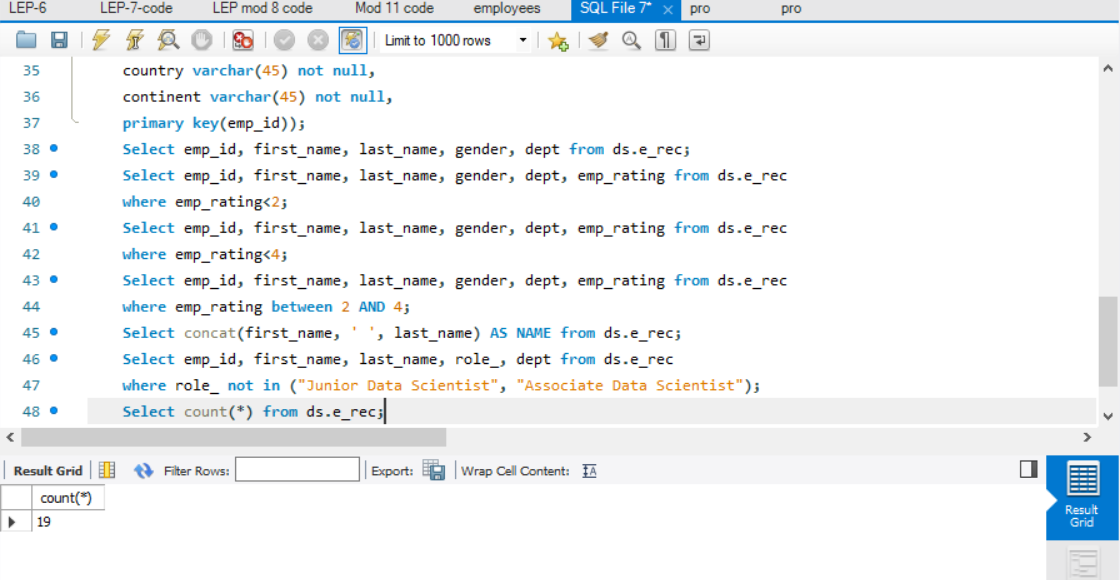
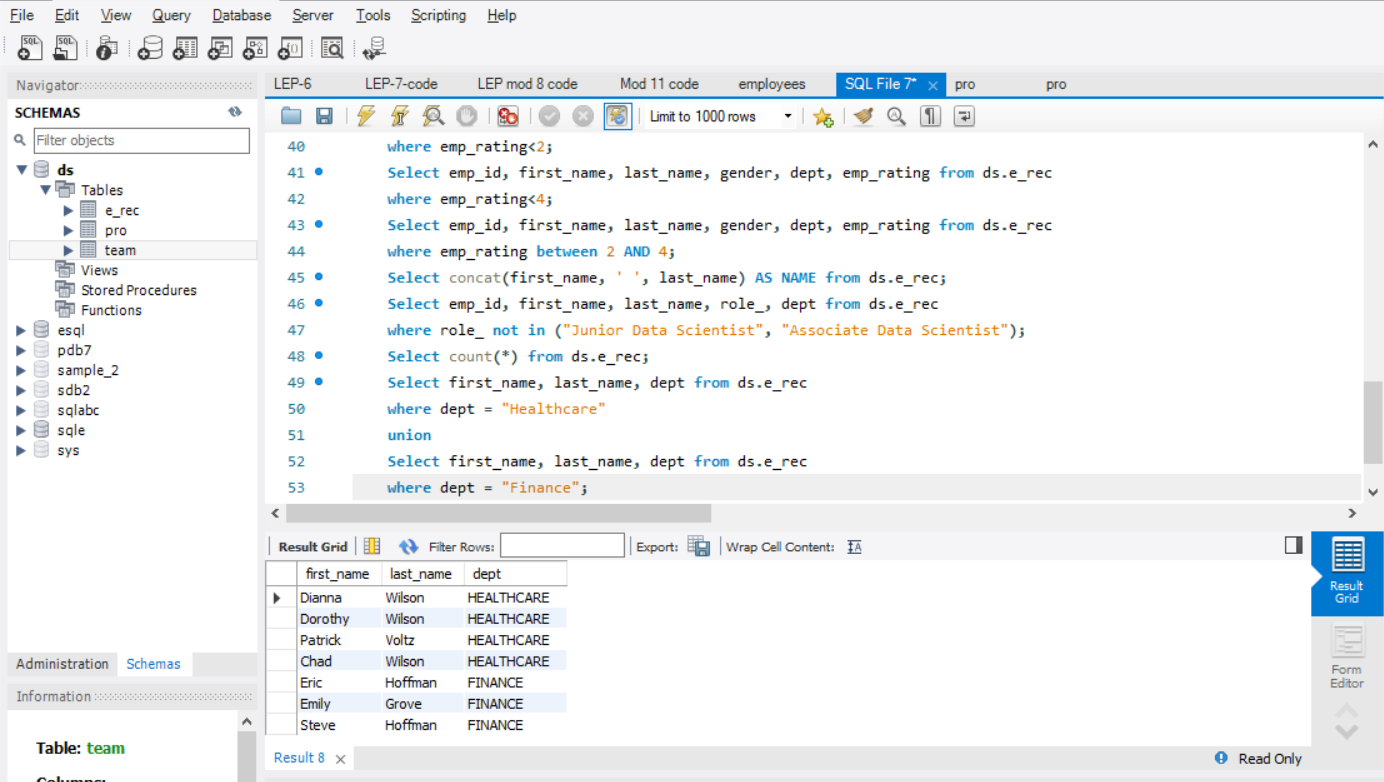
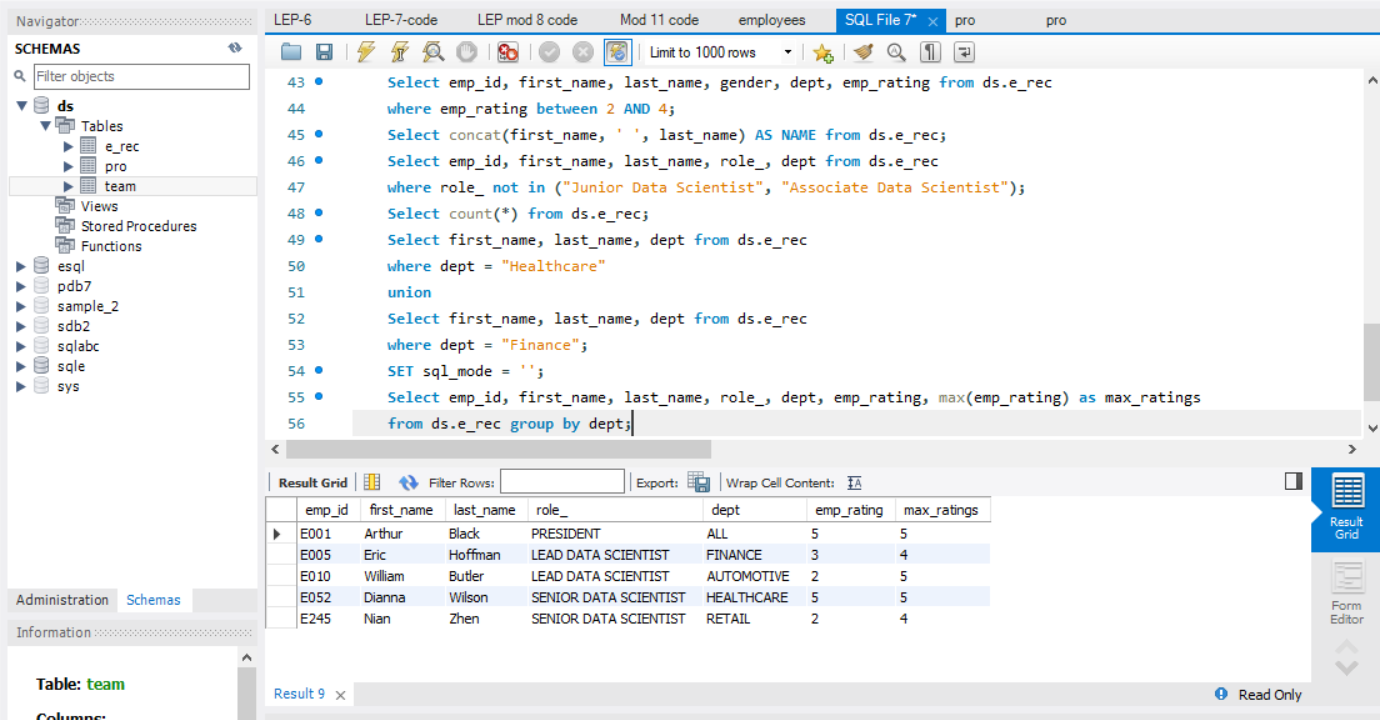
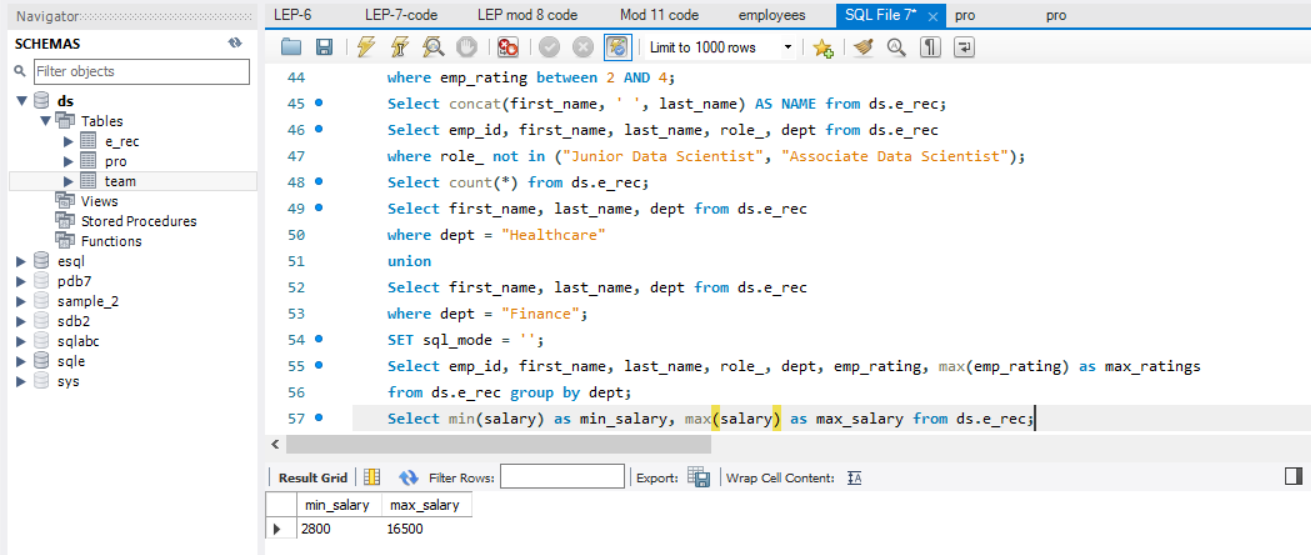
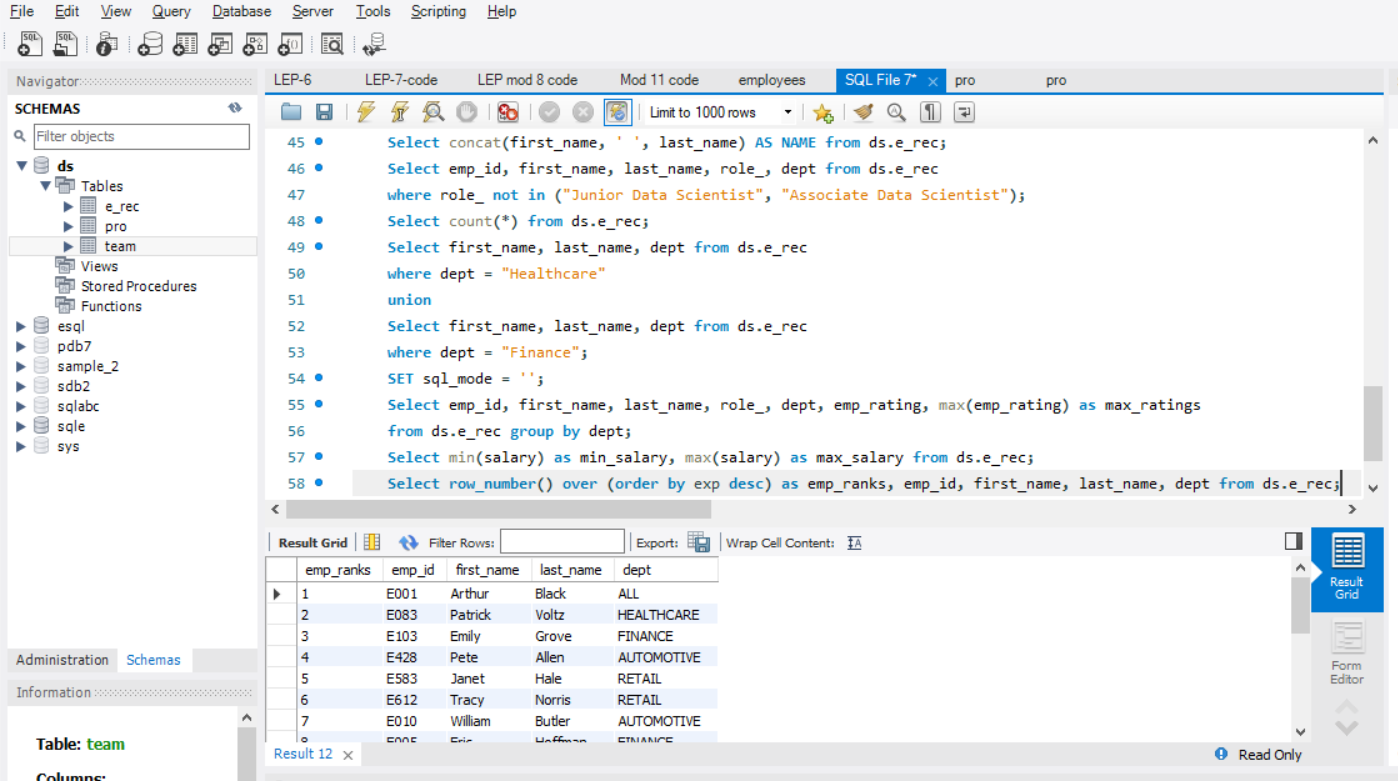
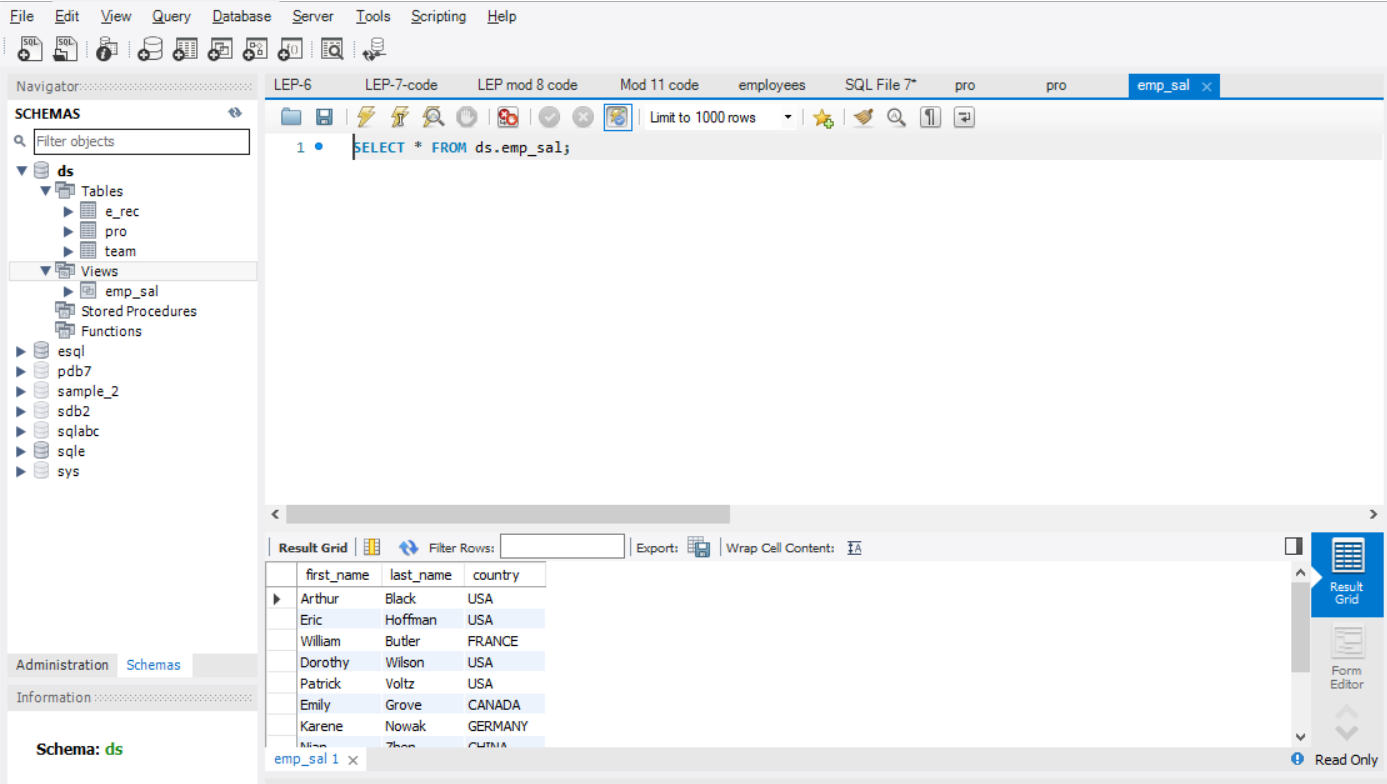
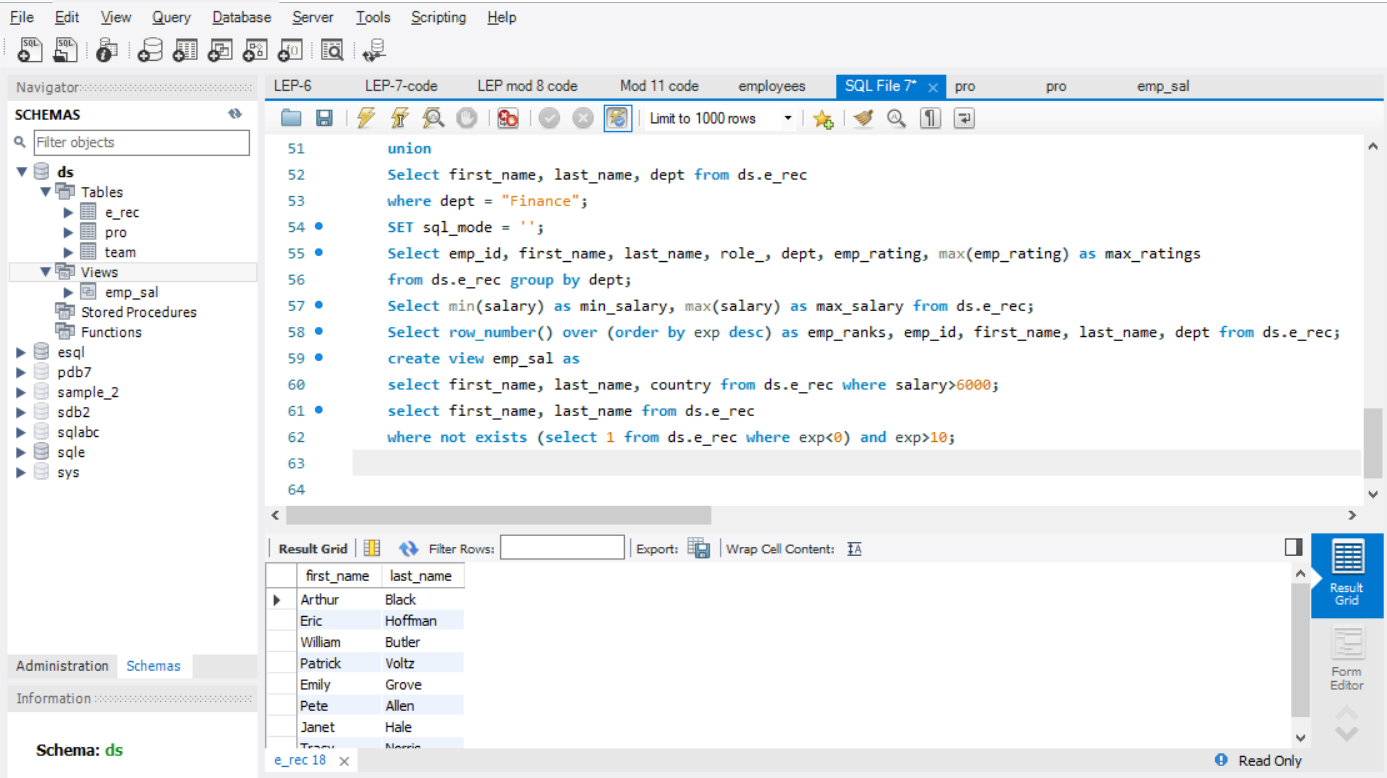
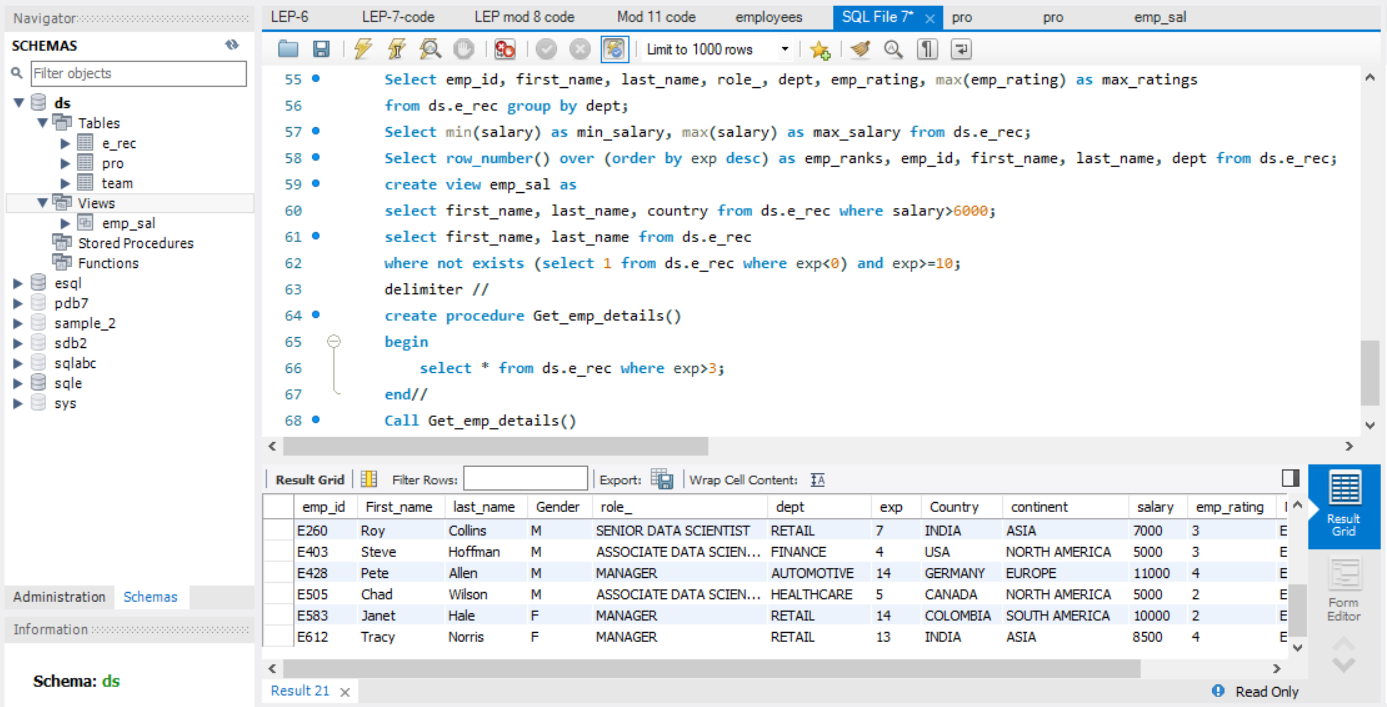
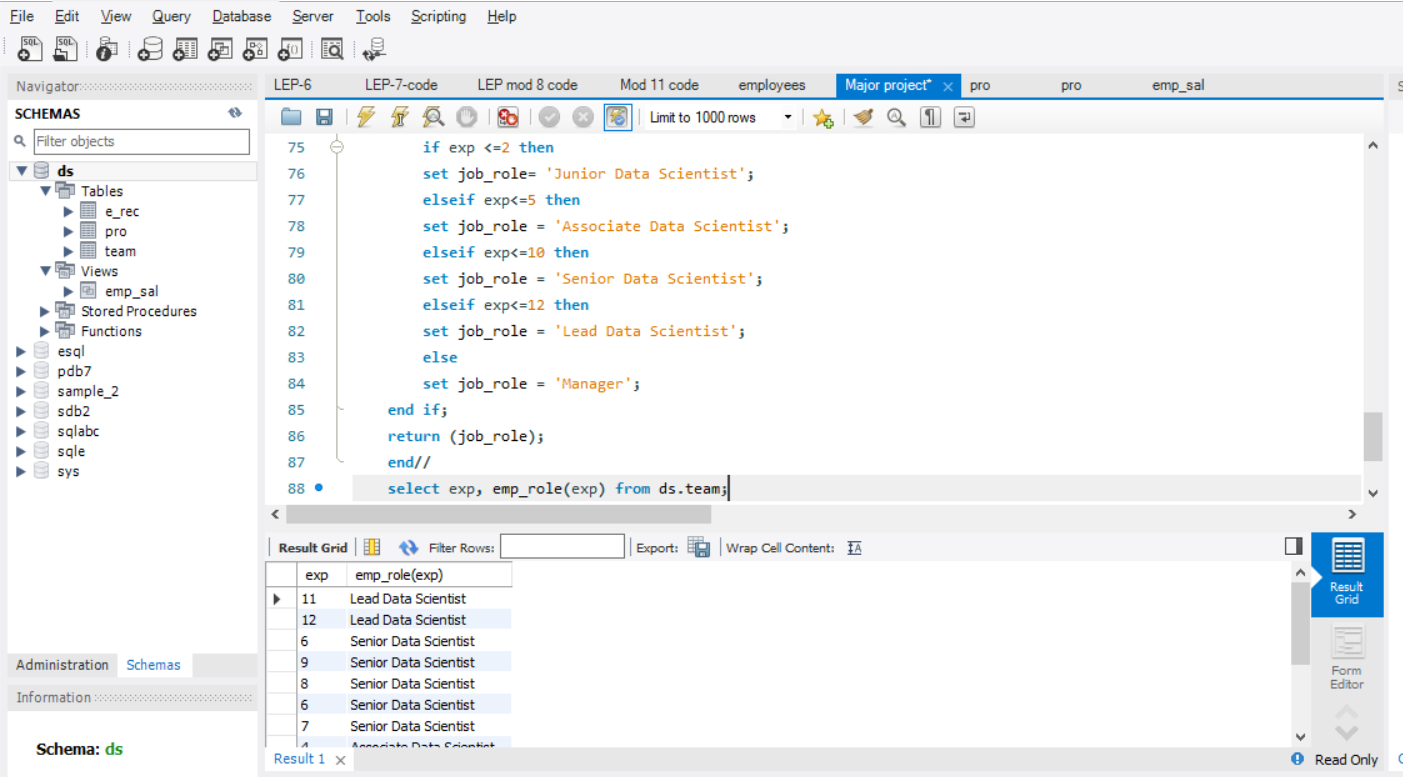
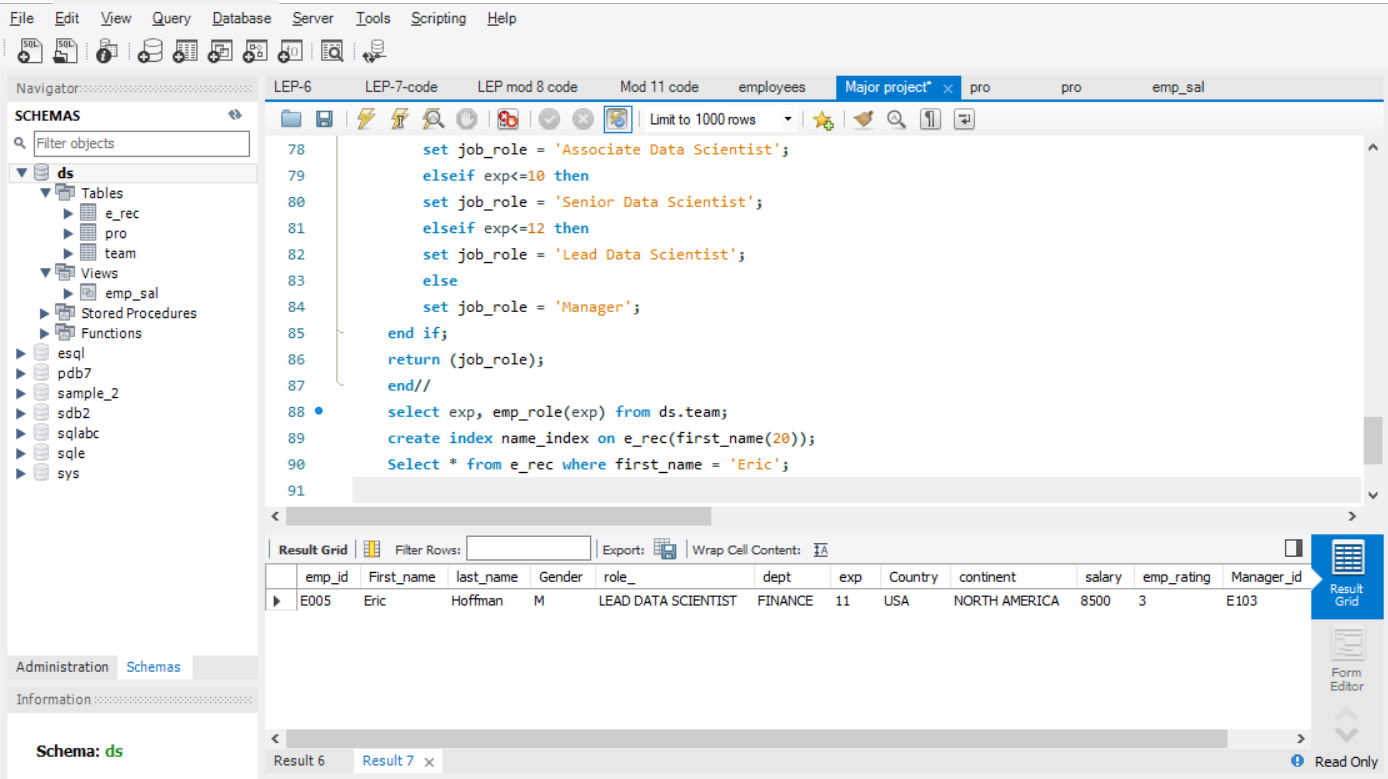
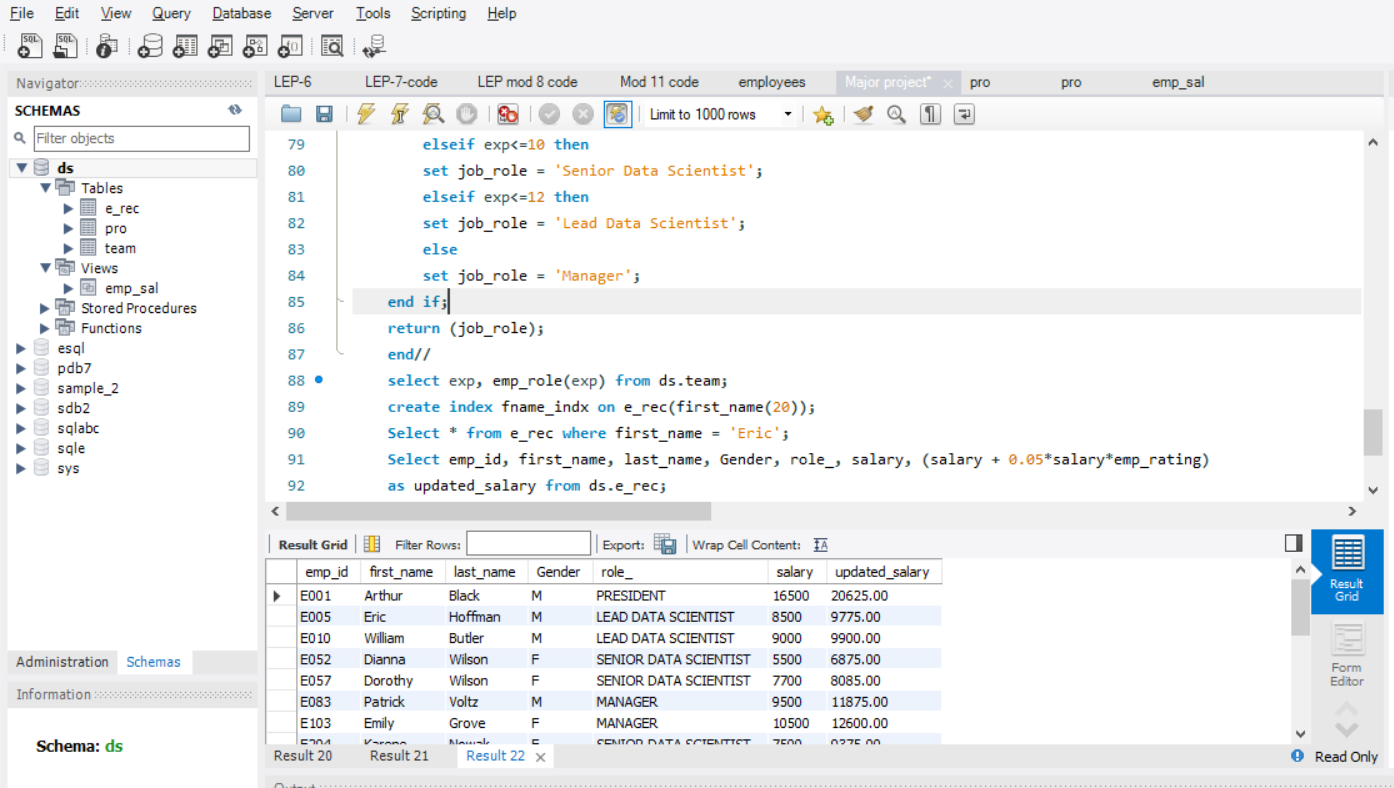
Outputs:

1. Retreving Data from table



1. Emp\_rating < 2:



1. Emp Rating < 4
2. Emp rating between 2 & 4
3. Concatenate names: 
4. Employees who have someone reporting to them: 
5. Count of reporters: 
6. Employees in healthcare and finance using UNION: 
7. Group employees by dept and determine max rating in each department: 
8. Calculate minimum and maximum salary: 
9. Ranking employees based on their experience: 
10. Employee with Salary>6000 from various countries using a view: 
11. Exp>10 years using nested query: 
12. Stored procedure for emp with exp>3: 
13. 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.
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: 
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. 
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). 
17. Write a query to calculate the average salary distribution based on the continent and country. Take data from the employee record table. 