RDBMS ASSIGNMENT

QUESTION 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 thegiven resources.

My command :

```
CREATE DATABASE RDBMS_ASSIGNMENT;

USE RDBMS_ASSIGNMENT;

--  After this, I imported data_science_team.csv, proj_table.csv and emp_record_table.csv into the
  --  RDBMS_ASSIGNMENT database from the given resources using MySQLWorkbench.
```

Ratinder's command:

```
create database employee;

use employee;

   -- Subsequently, He imported data_science_team.csv, proj_table.csv  and  emp_record_table.csv into the
    --  employee database from the given resources using MySQLWorkbench.
```

Difference :

- He used the same approach as mine.

Lav's command :

```
create database employee;

use employee;

  --  Subsequently I imported data_science_team.csv, proj_table.csv and emp_record_table.csv into

  --  the employee database from  the  given resources using MySQLWorkbench.
```

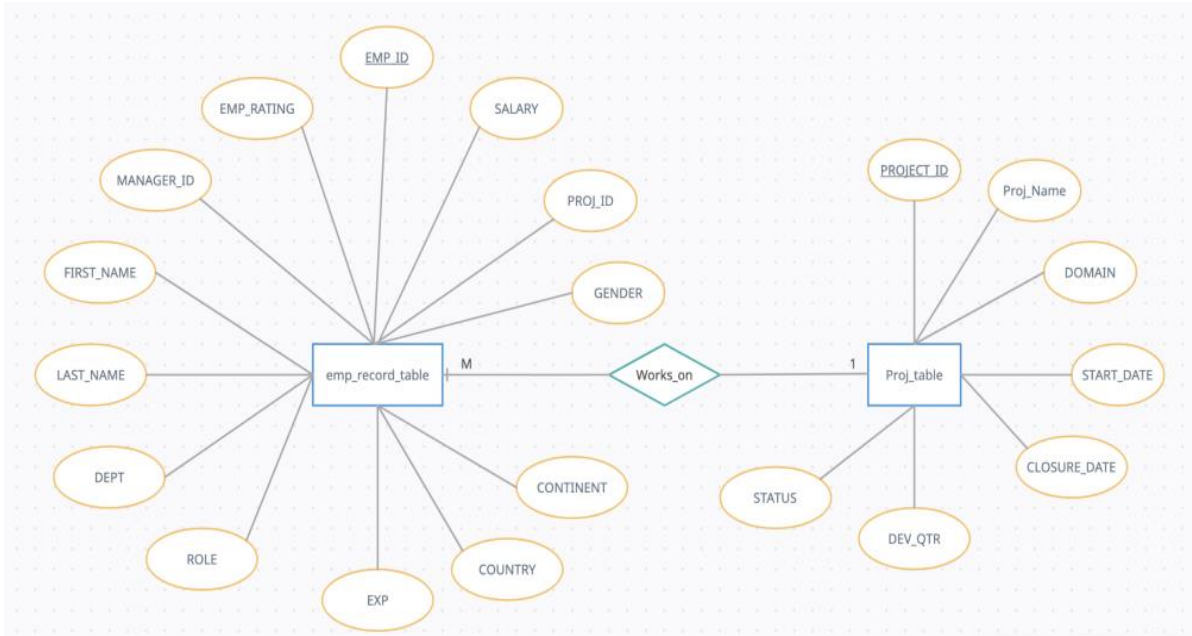
Difference :

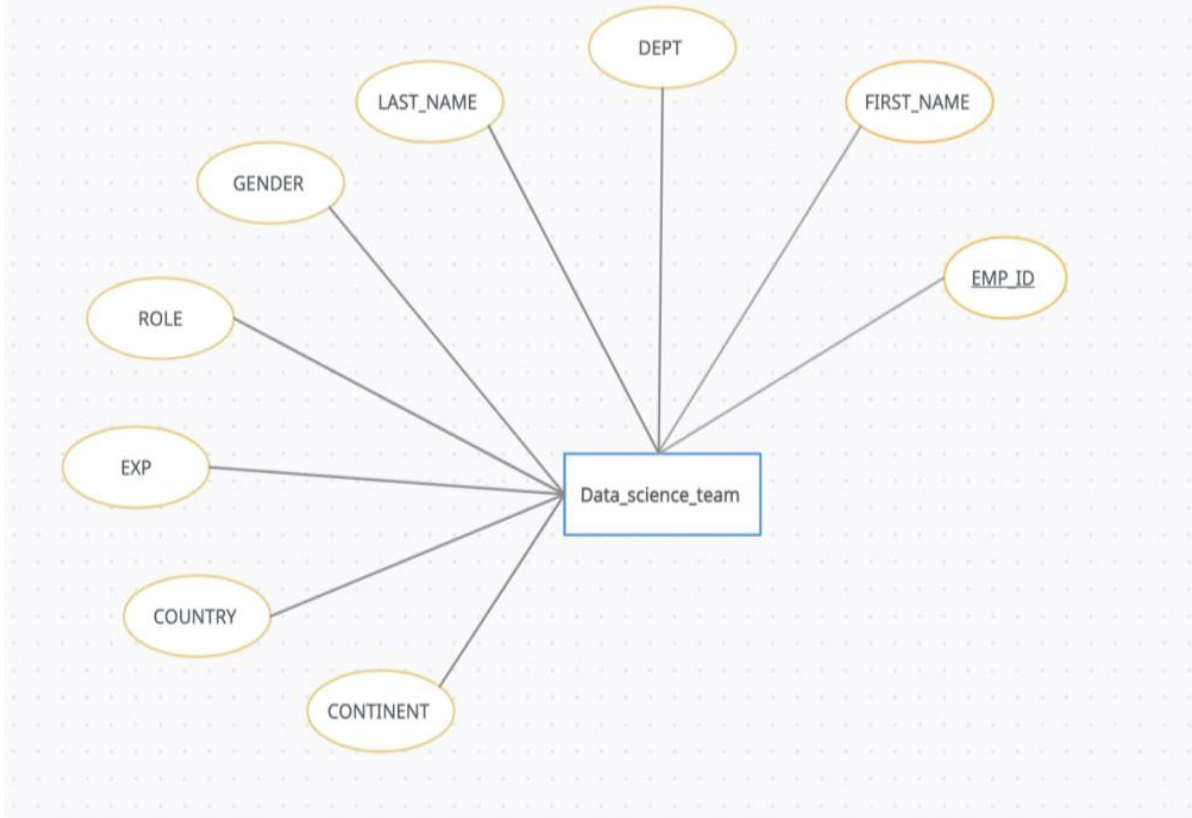
- His approach is same as mine.

.

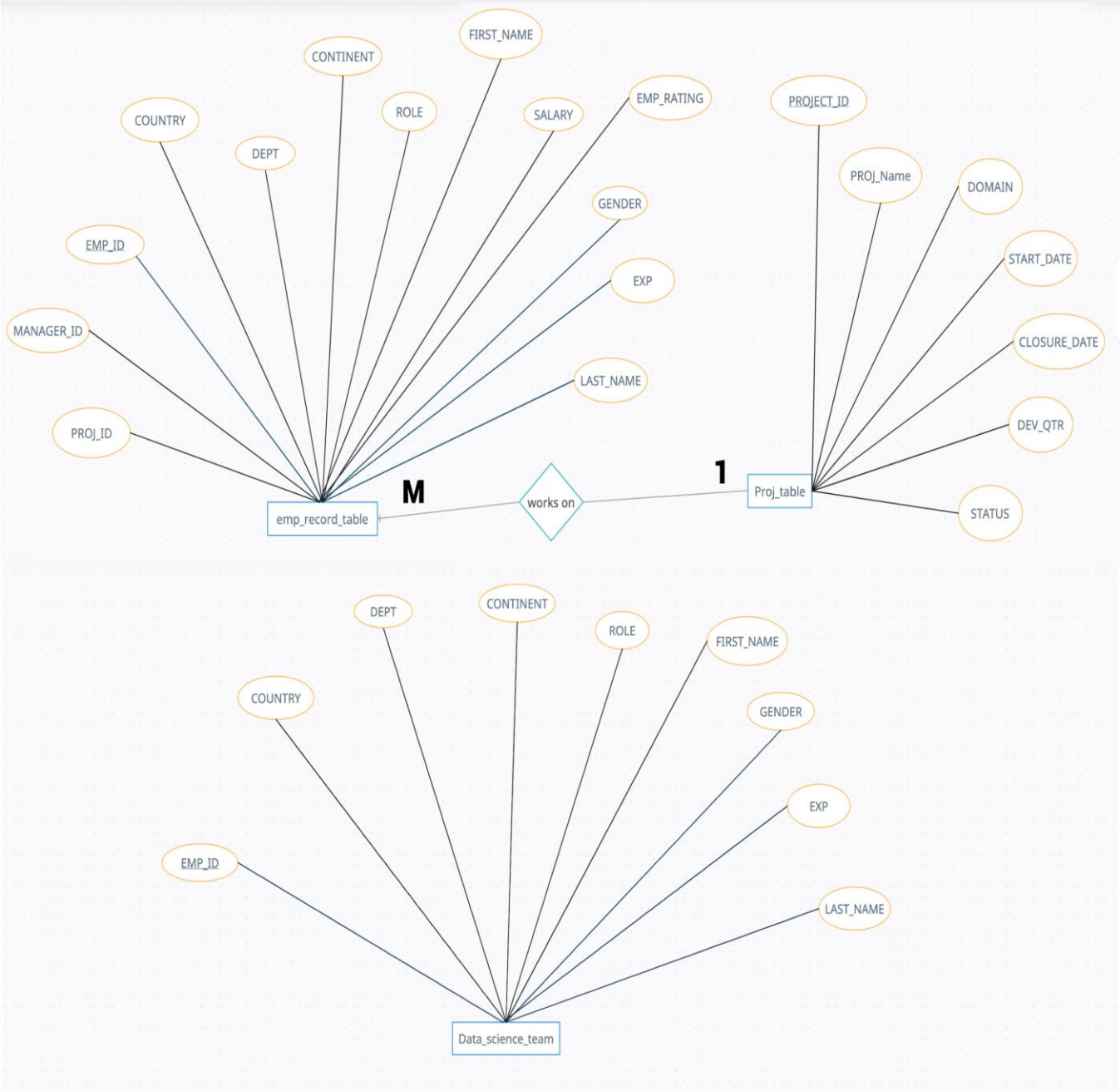
QUESTION 2 :- Create an ER diagram for the given employee database.

My ER Diagram :


EMP_ID
EMP_RATING
SALARY
MANAGER_ID
PROJ_ID
FIRST_NAME
GENDER
LAST_NAME
emp_record_table
M
Works_on
1
Proj_table
PROJECT_ID
Proj_Name
DOMAIN
START_DATE
CLOSURE_DATE
DEPT
CONTINENT
STATUS
DEV_QTR
ROLE
COUNTRY
EXP


DEPT
LAST_NAME
FIRST_NAME
GENDER
EMP_ID
ROLE
EXP
Data_science_team
COUNTRY
CONTINENT

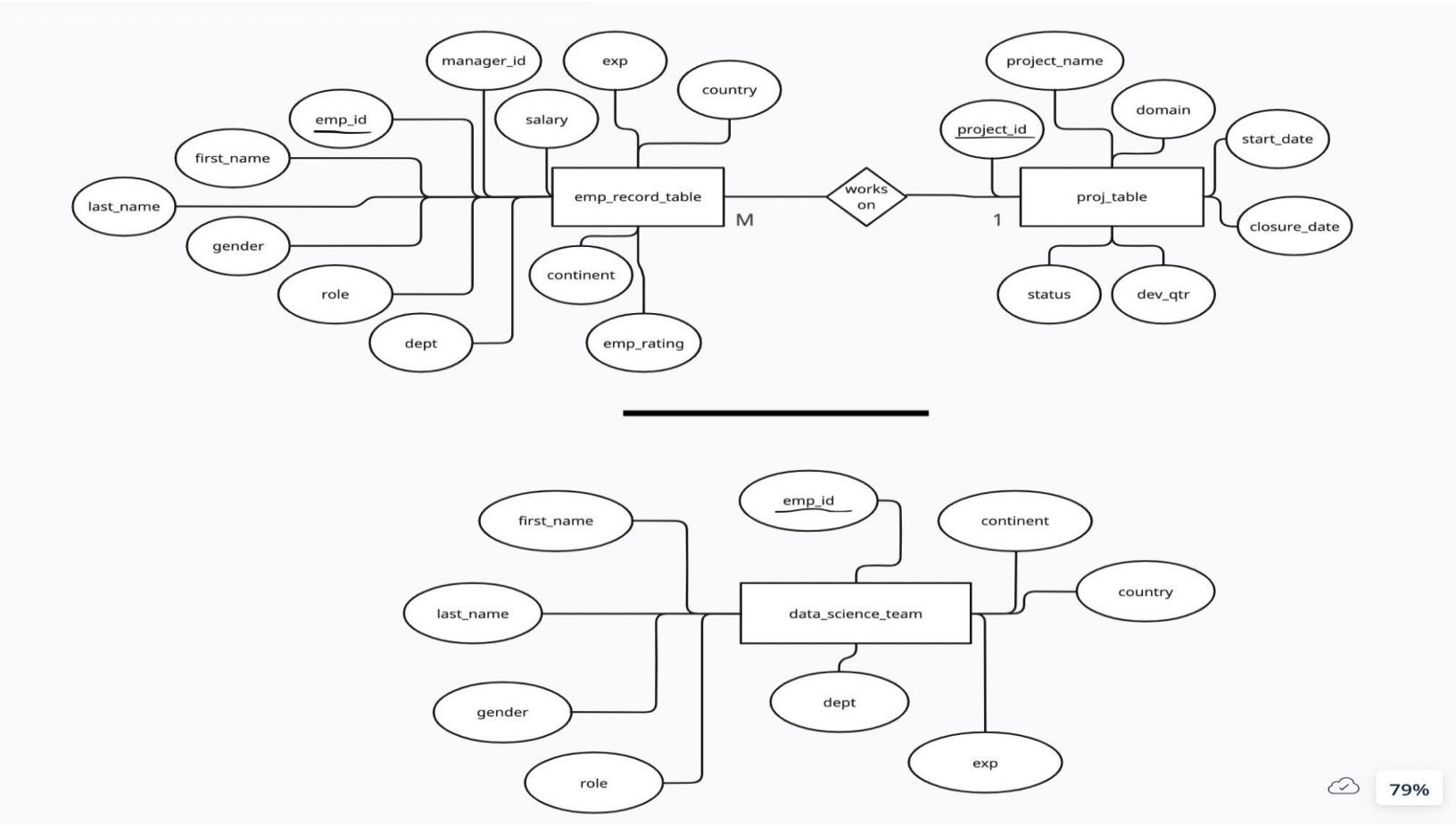
Ratinder’s ER Diagram :



Difference :

- His approach is same as mine.

Lav’s ER Diagram :



Difference :

- His approach is same as mine.

QUESTION 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.

My Query:

```
SELECT EMP_ID, FIRST_NAME,LAST_NAME,GENDER,DEPT
 FROM emp_record_table;
```

Ratinder's Query:

```
SELECT emp_id, first_name, last_name, gender, dept
  FROM emp_record_table;
```

Difference :

- His approach is same as mine.

Lav's Query:

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

  FROM emp_record_table;
```

Difference :

- His approach is same as mine.

QUESTION 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**

My Queries:

- SELECT EMP_ID, FIRST_NAME,LAST_NAME,GENDER,DEPT ,EMP_RATINGFROM emp_record_table WHERE EMP_RATING<2;
- SELECT EMP_ID, FIRST_NAME,LAST_NAME,GENDER,DEPT ,EMP_RATINGFROM emp_record_table WHERE EMP_RATING>4;
- SELECT EMP_ID, FIRST_NAME,LAST_NAME,GENDER,DEPT ,EMP_RATINGFROM emp_record_table WHERE EMP_RATING BETWEEN 2 AND 4;

Ratinder's Queries:

- SELECT emp_id, first_name, last_name, gender, dept, emp_rating
FROM emp_record_table
WHERE emp_rating<2;
- SELECT emp_id, first_name, last_name, gender, dept, emp_rating
FROM emp_record_table
WHERE emp_rating>4;
- SELECT emp_id, first_name, last_name, gender, dept, emp_rating
FROM emp_record_table
WHERE emp_rating BETWEEN 2 AND 4;

Difference :

- His approaches are same as mine.

Lav's Queries:

- SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
  FROM emp_record_table
  WHERE EMP_RATING<2;

- SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
  FROM emp_record_table
  WHERE EMP_RATING>4;

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

Difference :

- His approaches are same as mine.

QUESTION 5 :-

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

MY Query:

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

Ratinder's Query:

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

Difference :

- His approach is same as mine.

Lav's Query:

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

Difference :

- His approach is same as mine.

QUESTION 6 :-

Write a query to list only those employees who have someone reporting to them.Also, show the number of reporters (including the President).

My Query:

```
SELECT  M.EMP_ID , M.FIRST_NAME ,M.LAST_NAME , COUNT(*) AS CNT
FROM emp_record_table  E INNER JOIN emp_record_table MON
E.MANAGER_ID=M.EMP_ID
GROUP BY M.EMP_ID;
```

Ratinder's Query:

```
SELECT mgr.emp_id, mgr.first_name, mgr.last_name, COUNT(e.emp_id)
FROM emp_record_table AS e INNER JOIN emp_record_table AS mgr
ON e.manager_id = mgr.emp_id
GROUP BY mgr.emp_id, mgr.first_name, mgr.last_name;
```

Difference :

- His approach is same as mine.

Lav's Query:

```
SELECT CONCAT(emp.FIRST_NAME, ' ', EMP.LAST_NAME) AS NAME, CNT AS count
FROM (SELECT  COUNT(W.MANAGER_ID) CNT, MANAGER_ID
FROM emp_record_table W
GROUP BY W.MANAGER_ID
HAVING CNT<>0) A
join emp_record_table emp on emp.emp_id=a.manager_id;
```

Difference :

- He is finding the employee_id, count (no of employees working under that employee) by using sunqueries.
- He is joining the subqueries and emp table and finding the appropriate result by comparing their emp_id.

QUESTION 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.

My Query:

```
SELECT *
FROM emp_record_table WHERE
DEPT='HEALTHCARE'UNION
SELECT *
 FROM emp_record_table WHERE
 DEPT ='FINANCE';
```

Ratinder's Query:

```
  SELECT emp_id, first_name, last_name, dept
FROM emp_record_table
WHERE dept='Healthcare'
UNION
SELECT emp_id, first_name, last_name, dept
FROM emp_record_table
WHERE dept='Finance';
```

Difference :

- His approach is same as mine.

Lav's Query:

```
SELECT CONCAT(FIRST_NAME, ' ', LAST_NAME)
FROM emp_record_table
WHERE DEPT='HEALTHCARE'
UNION
SELECT CONCAT(FIRST_NAME, ' ', LAST_NAME)
FROM emp_record_table
WHERE DEPT='FINANCE';
```

Difference :

- His approach is same as mine.

QUESTION 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 thedepartment.

My Query:

```
SELECT EMP_ID,FIRST_NAME,LAST_NAME,ROLE,DEPT,EMP_RATING,
MAX(EMP_RATING)  OVER(PARTITION BY DEPT) AS MAX_RATING_BY_DEPT
 FROM emp_record_table;
```

Ratinder's Query:

```
SELECT emp_id, first_name, last_name, role, dept, emp_rating,
max(emp_rating) OVER(PARTITION BY dept) as max_rating_by_dept
FROM emp_record_table;
```

Difference :

- His approach is same as mine.

Lav's Query:

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, EMP_RATING,
MAX(EMP_RATING) OVER(PARTITION BY DEPT) AS MAX_RATING
FROM emp_record_table;
```

Difference :

- His approach is same as mine.

QUESTION 9 :-

Write a query to calculate the minimum and the maximum salary of the employees ineach role. Take data from the employee record table.

My Query:

```
SELECT DISTINCT ROLE, MAX(SALARY) OVER(PARTITION BY ROLE) ASMAX_SALARY
,MIN(SALARY)   OVER(PARTITION BY ROLE)
AS MIN_SALARY  FROM emp_record_table;
```

Ratinder's Query:

```
SELECT MIN(salary) as min_sal_by_role, MAX(salary) as max_sal_by_role
FROM emp_record_table
GROUP BY role;
```

Difference :

- He is using GROUP BY function and I am using window function.
- He is not printing the name of role .

Lav's Query:

```
SELECT DISTINCT ROLE, MIN(SALARY) OVER(PARTITION BY ROLE) AS MIN_SAL,
MIN(SALARY) OVER(PARTITION BY ROLE) AS MAX_SAL
FROM emp_record_table;
```

Difference :

- His approach is same as mine.

QUESTION 10 :-

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

My Query:

```
SELECT EMP_ID ,EXP, DENSE_RANK() OVER(ORDER BY EXP DESC) ASRANK_BY_EXP
FROM emp_record_table ;
```

Ratinder's Query:

```
SELECT DISTINCT emp_id, exp, DENSE_RANK() OVER(ORDER BY exp DESC) AS rank_by_exp
FROM emp_record_table;
```

Difference :

- His approach is same as mine.

Lav's Query:

```
SELECT *, row_NUMBER() OVER(ORDER BY EXP DESC) AS RANKING
FROM emp_record_table;
```

Difference :

- He has used ROW_NUMBER instead of DENSE_RANK.

QUESTION 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 recordtable.

QUERY:

```
CREATE OR REPLACE VIEW SALAR_GREATER_THEN_SIX_THOUSAND AS
SELECT EMP_ID,COUNTRY,SALARY FROM emp_record_tableWHERE SALARY>6000;

SELECT * FROM SALAR_GREATER_THEN_SIX_THOUSAND;
```

Ratinder's Query:

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

SELECT * FROM sal_greater_six_K;
```

Difference :

- His approach is same as mine.

Lav's Query:

```
CREATE VIEW EMP_RECORD AS
SELECT FIRST_NAME, COUNTRY
FROM emp_record_table
WHERE SALARY>6000;

SELECT * FROM VIEW_EMP;
```

Difference :

- His approach is same as mine.

QUESTION 12 :-

Write a nested query to find employees with experience of more than tenyears. Take data from the

employee record table.

My Query:

```
SELECT *
FROM emp_record_table
WHERE EMP_ID  IN
(SELECT EMP_ID
 FROM emp_record_table
WHERE EXP>10);
```

Ratinder's Query:

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

Difference :

- His approach is same as mine.

Lav's Query:

```
SELECT EMP_ID, FIRST_NAME, ROLE, EXP
FROM (SELECT *
FROM emp_record_table
where exp>10) A;
```

Difference :

- His approach is same as mine.

QUESTION 13 :-

Write a query to create a stored procedure to retrieve the details of the employees whose experience is more than threeyears. Take data from the employee record table.

My Query:

```
DELIMITER $$
CREATE PROCEDURE Emp_Details()
BEGIN
SELECT *
FROM emp_record_tableWHERE
EXP>3;
END
$$ DELIMITER ;

CALL Emp_Details();
```

Ratinder's Query:

```
delimiter $$
create procedure emp_details()
begin
select * from emp_record_table
where exp>3;
end$$
delimiter ;

call emp_details();
```

Difference :

- His approach is same as mine.

Lav's Query:

```
DELIMITER $$
CREATE PROCEDURE EMP_DET()
BEGIN
SELECT * FROM emp_record_table
WHERE EXP>3;
END $$
DELIMITER ;

CALL EMP_DET();
```

Difference :

- His approach is same as mine.

QUESTION 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'.

QUERY:

```
DELIMITER $$
CREATE FUNCTION ScientistExperiance()RETURNS
TINYINT(1)
DETERMINISTIC
BEGIN
DECLARE RES INT DEFAULT 1;DECLARE
RL VARCHAR(30);
DECLARE SETEND INTEGER DEFAULT 0;DECLARE EP
INT;
DECLARE CURNAME CURSOR FOR
SELECT ROLE,EXP FROM emp_record_table; DECLARE CONTINUE
HANDLER FOR NOT FOUNDSET SETEND=1;

OPEN CURNAME;
MY_LOOP:LOOP
FETCH CURNAME INTO RL,EP;IF
SETEND=1 THEN
LEAVE MY_LOOP;
END IF;

IF(RL='JUNIOR DATA SCIENTIST' AND EP>2) THENSET RES=0;
LEAVE MY_LOOP;
ELSEIF(RL='ASSOCIATE DATA SCIENTIST' AND (EP<=2 OR EP>5)) THENSET RES=0;
LEAVE MY_LOOP;
ELSEIF(RL='SENIOR DATA SCIENTIST' AND (EP<=5 OR EP>10)) THENSET RES=0;
LEAVE MY_LOOP;
ELSEIF(RL='LEAD DATA SCIENTIST' AND (EP<=10 OR EP>12)) THENSET RES=0;
LEAVE MY_LOOP;
```

```
ELSEIF(RL='MANAGER' AND (EP<=12 OR EP>16)) THENSET RES=0;
LEAVE MY_LOOP;
END IF;
END LOOP MY_LOOP;CLOSE
CURNAME; IF(RES=1) THEN
RETURN TRUE;
ELSE
RETURN FALSE;
END IF;
END
$$ DELIMITER ;
```

```
DELIMITER $$
CREATE PROCEDURE Helper_Procedure()BEGIN
IF ScientistExperiance() THEN
SELECT 'THE PROFILE ASSIGNED TO EACH EMPLOYEE IN THE DATA SCIENCETEAM MATCHES THE
ORGANIZATION SET STANDARD.' AS MESSAGE;
ELSE
SELECT 'THE PROFILE ASSIGNED TO EACH EMPLOYEE IN THE DATA SCIENCE TEAM DOES NOT MATCH
THE ORGANIZATION SET STANDARD.' AS MESSAGE;END IF;
END $$
DELIMITER ;

CALL Helper_Procedure();
```

Ratinder's Query:

```
delimiter $$

create function emp_details() returns tinyint(1) deterministic
begin
declare v_exp int default 0;
declare v_role varchar(50) default "";
declare finished int default 0;
declare dummy_cursor cursor for
select exp, role from emp_record_table;
declare continue handler for not found
set finished=1;
open dummy_cursor;
check_role: loop
fetch dummy_cursor into v_exp, v_role;
if finished = 1 then
leave check_role;
end if;
if (v_exp<=2 and v_role!='JUNIOR DATA SCIENTIST') then
return false;
elseif (v_exp>2 and v_exp<=5 and v_role!='ASSOCIATE DATA SCIENTIST') then
return false;
elseif (v_exp>5 and v_exp<=10 and v_role!='SENIOR DATA SCIENTIST') then
return false;
elseif (v_exp>10 and v_exp<=12 and v_role!='LEAD DATA SCIENTIST') then
return false;
elseif (v_exp>12 and v_exp<=16 and v_role!='MANAGER') then
return false;
end if;
```

```
end loop check_role;

close dummy_cursor;

return true;

end$$

delimiter ;

delimiter $$
create procedure helper_procedure()
begin

if emp_details() then
select 'The job profile assigned to each employee
in the data science team matches the organization’s set standard.' as message;

else
select 'The job profile assigned to each employee
in the data science team does not match the organization’s set standard.' as message;
end if;
end$$
delimiter ;
call helper_procedure();
```

;

Difference :

- His approach is same as mine.

Lav’s Query:

```
DELIMITER $$
        CREATE FUNCTION get_job_profile(experience INT, role VARCHAR(30))
        RETURNS VARCHAR(50)
        DETERMINISTIC

        BEGIN
```

```
 DECLARE job_profile varchar(50);
  DECLARE flag varchar(30);

 IF experience<=2 THEN
SET job_profile='JUNIOR DATA SCIENTIST';
 ELSEIF experience between 2 AND 5 THEN
SET job_profile='ASSOCIATE DATA SCIENTIST';
 ELSEIF experience between 5 AND 10 THEN
SET job_profile='SENIOR DATA SCIENTIST';
 ELSEIF experience between 10 AND 12 THEN
SET job_profile='LEAD DATA SCIENTIST';
 ELSEIF experience between 12 AND 16 THEN
SET job_profile='MANAGER';
 END IF;

 IF job_profile=role THEN
 SET flag='YES';
ELSE
SET flag='NO';
 END IF;

 RETURN flag;
END $$
DELIMITER ;


SELECT EMP_ID, FIRST_NAME, EXP, ROLE, get_job_profile(EXP, ROLE) AS CHECK_PROFILE
FROM data_science_team;
```

Difference :

- He is creating a function and checking all the condition and returning YES or NO.
- He is finding all the details of the emplyees and adding a new column where he is storing ‘YES’ or ‘NO’ as result.

QUESTION 15 :-

Create an index to improve the cost and performance of the query to findthe employee whose FIRST_NAME is ‘Eric’ in the employee table after checking the execution plan.

My Query:

```
CREATE INDEX Eric_Index
ON emp_record_table(FIRST_NAME);

SELECT FIRST_NAME
FROM emp_record_table WHERE
FIRST_NAME='Eric';
```

Ratinder’s Query:

```
create index ename_index
on emp_record_table(first_name);

select *
from emp_record_table
where first_name = 'Eric';
```

Difference :

- His approach is same as mine.

Lav’s Query:

```
CREATE INDEX idx_first_name
ON emp_record_table(FIRST_NAME(20));

SELECT * FROM emp_record_table
WHERE FIRST_NAME='Eric';
```

Difference :

- His approach is same as mine.

QUESTION 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).

My Query:

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

Ratinder's Query:

```
select emp_id, emp_rating, salary, (0.05*salary*emp_rating) as bonus
from emp_record_table;
```

Difference :

- His approach is same as mine.

Lav's Query:

```
SELECT EMP_ID, CONCAT(FIRST_NAME, ' ', LAST_NAME) AS NAME, SALARY,
 ROUND(((SALARY*5)/100)*EMP_RATING) AS BONUS
 FROM emp_record_table;
```

Difference :

- His approach is same as mine.

QUESTION 17 :-

Write a query to calculate the average salary distributionbased on the continent and country.Take data from the employee record table.

My Query:

```
SELECT DISTINCT CONTINENT,COUNTRY , AVG(SALARY)
OVER(PARTITION BY CONTINENT) AS
AVG_SALARY_BY_CONTINENT,
AVG(SALARY) OVER(PARTITION BY COUNTRY) ASAVG_SALARY_BY_COUNTRY
FROM emp_record_table;
```

Ratinder's Query:

```
select distinct continent,
avg(salary) over(partition by continent) as avg_sal_by_continent,
    country, avg(salary) over(partition by country) as avg_sal_by_country
    from emp_record_table;
```

Difference :

- His approach is same as mine.

Lav's Query:

```
 SELECT  DISTINCT COUNTRY,
AVG(SALARY) OVER(PARTITION BY COUNTRY) AS COUNTRY_AVG,
CONTINENT, AVG(SALARY) OVER(PARTITION BY CONTINENT) AS CONTINENT_AVG
FROM emp_record_table;
```

Difference :

- His approach is same as mine.