**Que – 1. Problem Statement:**

Show the details of the employees who have the 5th highest salary in each job category.

* Return the columns '**employee\_id**', '**first\_name**', '**job\_id**'.

**Note:**

1. Select MySQL 8.0 in the drop-down.
2. Use the employees table.

**Ans – 1.**

SELECT \* from

(SELECT employee\_id,

first\_name,

job\_id,

DENSE\_RANK()OVER(PARTITION BY job\_id ORDER BY salary desc)as salary\_rank

FROM employees)as s

WHERE s.salary\_rank=5

**Que – 2. Problem Statement:**

Find the **average** **salary** of the employees for **each** department and **order** the departments by **department\_id**in ascending order. Save the average salary as ‘Average\_salary’.

* Return the columns '**department\_id**', '**department\_name**', '**Average\_salary**'.
* Use the tables **employees** and **departments**.

**Ans – 2.**

SELECT e.department\_id,

d.department\_name,

avg(e.salary)over(partition by e.department\_id) AS average\_salary

FROM employees AS e

LEFT JOIN departments AS d

ON e.department\_id=d.department\_id

**Que – 3. Problem Statement:**

Write a Query to find the **first day of the first job** of every employee and return it as ‘first\_day\_job’.

* Return the columns '**first\_name**', '**first\_day\_job**'.

**Ans – 3.**

SELECT e.first\_name,

jh.first\_date

FROM employees AS e left join

(SELECT DISTINCT employee\_id,

min(start\_date)over(partition by employee\_id) AS first\_date

FROM job\_history) AS jh

ON e.employee\_id=jh.employee\_id

**Que – 4. Problem Statement:**

Write a query to find the starting **maximum salary of the first job** that every employee held and return it as ‘first\_job\_sal’.

* Return the columns '**first\_name**', '**last\_name**', '**first\_job\_sal**' **sorted by first\_name**.

**Ans – 4.**

select t.first\_name,

t.last\_name,

t.first\_job\_id,

j.max\_salary

from(select e.first\_name,e.last\_name,

case when jh.job\_id is null

then e.job\_id

else jh.job\_id

end as first\_job\_id

from employees as e

left join(SELECT DISTINCT employee\_id,job\_id,

min(start\_date)over(partition by employee\_id) AS first\_date

FROM job\_history) as jh

on e.employee\_id=jh.employee\_id)as t

left join jobs as j

on t.first\_job\_id = j.job\_id

**Que – 5. Problem Statement:**

Display the details of the employees who earn the **highest** **salary** in their departments.

* Return the columns '**department\_name**', '**first\_name**', '**last\_name**', and '**salary**'.

**Ans – 5.**

select d.department\_name,

e.first\_name,

e.last\_name,

max(e.salary)over(partition by e.department\_id)as salary

from employees as e

left join departments as d

on e.department\_id = d.department\_id

**Que – 6. Problem Statement:**

Write a Query to find the **first day of the most recent job** of every employee and return it as the ‘recent\_job’.

* Return the columns '**first\_name**', and '**recent\_job**'.

**Ans – 6.**

SELECT e.first\_name,

jh.recent\_date

FROM employees AS e left join

(SELECT DISTINCT employee\_id,

max(start\_date)over(partition by employee\_id) AS recent\_date

FROM job\_history) AS jh

ON e.employee\_id=jh.employee\_id

**Que – 7. Problem Statement:**

Display the employee’s details along with the **current job** and the **previous job** as ‘previous\_job’ of all the employees in the company.

* Return the columns '**first\_name**', '**last\_name**', '**job\_title**', '**previous\_job**'.

**Ans – 7.**

SELECT e.first\_name,

e.last\_name,

e.job\_id,

t.prev\_job\_id,

j.job\_title

FROM employees AS e left join

(select \* from(SELECT employee\_id as prev\_emp\_id,

job\_id as prev\_job\_id,

count(employee\_id)over(partition by employee\_id

ORDER BY end\_date desc)as count

FROM job\_history)as t

where t.count>=2)as t

ON e.employee\_id=t.prev\_emp\_id

left join jobs as j

on e.job\_id=j.job\_id

**Que – 8. Problem Statement:**

Display the employee’s details along with the **previous job** as ‘first\_job’ and the **next promoted job** as ‘promoted\_to’ for all the employees in the company.

* Return the columns '**employee\_id**', '**first\_name**', '**last\_name**', '**first\_job**', '**promoted\_to**' .

**Ans – 8.**

SELECT e.employee\_id,

e.first\_name,

e.last\_name,

t.prev\_job\_id as first\_job\_id,

e.job\_id as current\_job\_id

FROM employees AS e left join

(select \* from(SELECT employee\_id as prev\_emp\_id,

job\_id as prev\_job\_id,

count(employee\_id)over(partition by employee\_id

ORDER BY end\_date desc)as count

FROM job\_history)as t

where t.count>=2)as t

ON e.employee\_id=t.prev\_emp\_id

**Que – 10. Problem Statement:**

Write a query to calculate **row number** and save as ‘emp\_row\_no’, **rank** and save as ‘emp\_rank’, and the **dense** **rank** of employees as ‘emp\_dense\_rank’ based on the **salary** column in **descending order** **within each department** using the employee’s table.

* Return the columns '**full\_name**' (first\_name and last\_name separated by space), '**department\_id**', '**salary**', '**emp\_row\_no**', '**emp\_rank**', and '**emp\_dense\_rank**'.

**Ans – 10.**

select concat(first\_name," ",last\_name) as full\_name,

department\_id,

salary,

row\_number()over(partition by department\_id order by salary desc) as emp\_row\_no,

rank()over(partition by department\_id order by salary desc) as emp\_rank,

dense\_rank()over(partition by department\_id order by salary desc) as emp\_dense\_rank

from employees

**Que – 11.**