Todays CT

DDL

```
CREATE TABLE `classroom`(
   `building` varchar(20) NOT NULL,#primary
   `room_num` varchar(20) NOT NULL,
   `capacity` INT,


    PRIMARY KEY(`building`,`room_num`)

);
CREATE TABLE `department`(
   `dept_name` varchar(20) NOT NULL,#primary
   `building` varchar(20) NOT NULL,
   `budget` varchar(20) ,

    PRIMARY KEY(`dept_name`),
    FOREIGN KEY(`building`) REFERENCES classroom(`building`)

);
CREATE TABLE `course`(
   `course_id` varchar(20) NOT NULL,#primary
   `title` varchar(20) ,
   `dept_name` varchar(20) NOT NULL,#foreign
   `credits` float(2,1),

    PRIMARY KEY(`course_id`),
    FOREIGN KEY(`dept_name`) REFERENCES depertment(`dept_name`)

);

CREATE TABLE `instructor`(
   `ID` varchar(20) NOT NULL,#primary
   `name` varchar(20) ,
   `dept_name` varchar(20) NOT NULL,#foreign
   `salary` float(2,1),

    PRIMARY KEY(`ID`)
FOREIGN KEY(`dept_name`) REFERENCES depertment(`dept_name`)
  );
```

DML

```
insert into classroom(building,room_num,capacity)

VALUES('Padma','011A',45)

insert into depertment(dept_name,building,budget)

VALUES('EEE','Meghna','870000.00')

insert into depertment(dept_name,building,budget)

VALUES('EEE','Meghna','870000.00')
```

– Update table name(DDL)

Rename TABLE takes to takings

– Update column name(DDL)

ALTER TABLE advisor CHANGE COLUMN s_ID student_ID Varchar(20);
ALTER TABLE advisor CHANGE COLUMN i_ID Teacher_ID Varchar(20);

– ADD and Drop table

```
CREATE TABLE `prerequisite` (
   `course_id` VARCHAR(20) NOT NULL,
   `title` VARCHAR(20) NOT NULL,
   PRIMARY KEY (`course_id`),
   FOREIGN KEY (`course_id`) REFERENCES course (`course_id`)

);

DROP table prerequisite;
```

– Update Column data

```
UPDATE advisor  SET Teacher_ID = 'NaS'  WHERE Teacher_ID = 'Nas';
```

Primary key and foreign key delete and update

```
ALTER TABLE table_name DROP CONSTRAINT constraint_name;

ALTER TABLE students DROP CONSTRAINT pk_students_id;
```

Searching(ripas)

1. Show all data from countries table.

```
SELECT *
FROM countries;
```

2. Show all data from employees table.

```
SELECT*
FROM employees;
```

3. Show all data from departments table.
4. Show all data from job_history table.

LIKE / NOT LIKE

1. Show those employee details whose first name starts with the letter s.

```
SELECT *
FROM employees
WHERE first_name LIKE 's%';
```

2. Show those employee details whose first name doesn't start with the letter s.

```
SELECT *
FROM employees
WHERE first_name NOT LIKE 's%';
```

3. Show those employee details whose first name ends with the letter a.

```
SELECT*
FROM employees
WHERE first_name LIKE '%a';
```

4. Show those employee details whose first name contains da as substring.

```
SELECT*
```

```
FROM employees
WHERE first_name LIKE 'da%';
```

5. Show those employee details whose first name starts with s and ends with a.

```
SELECT*
FROM employees
WHERE first_name LIKE 's%' AND first_name LIKE'%a';
```

6. Show those employee details whose first name either starts with s or starts with m.

```
SELECT*
FROM employees
WHERE first_name LIKE 's%' OR first_name LIKE 'm%';
```

7. Show those employee details whose first name contains the letter o and a.

```
SELECT*
FROM employees
WHERE first_name LIKE '%o%' AND first_name LIKE'%a%';
```

8. Show those employee details whose first name contains the letter o followed by the letter a.

```
SELECT*
FROM employees
WHERE first_name LIKE '%o_a%';
```

9. Show those employee details whose first name consists of exactly 3 characters.

```
SELECT*
FROM employees
WHERE first_name LIKE '___';
```

10. Show those employee details whose first name consists of minimum 3 characters.

```
SELECT*
FROM employees
WHERE first_name LIKE '%___%';
```

11. Show those employee details whose first name contains the letter a from the second last position.

```
SELECT*
FROM employees
WHERE first_name LIKE '_a%';
```

IN() / NOT IN( ) / OR, ||

1. Show those country details whose country_id is AU/BR/CN/JP.

```
SELECT*
FROM countries
WHERE country_id IN('AU','BR','CN','JP');
```

2. Show those department details whose manager_id is not 204/100/145.

```
SELECT*
FROM departments
WHERE manager_id NOT IN('204','100','145');
```

3. Show those employee details whose job_id is ST_MAN/IT_PROG.

```
SELECT *
FROM employees
```

```
WHERE job_id  IN('ST_MAN','IT_PROG')
```

4. Show those employee details who does not work in the department_id 100/30/90.

```
SELECT *
FROM employees
WHERE department_id NOT IN(100,30,90);
```

5. Show those location details where the postal_code is either 2901/50090.

```
SELECT *
FROM locations
WHERE postal_code IN(2901,50090);
```

6. Show those location details where the city name is either Roma/Venice/Tokyo.

```
SELECT *
FROM locations
WHERE city IN('Roma','Venice','Tokyo');
```

BETWEEN … AND… / NOT BETWEEN … AND … / AND, &&

1. Show those department details whose location_id is within the range 1000 to 2000 inclusive.

```
SELECT *
FROM departments
WHERE location_id  BETWEEN 1000 AND 2000;
```

2. Show those employee details whose salary is within the range 10000 to 20000 inclusive.

```
SELECT *
FROM employees
WHERE salary BETWEEN 10000 AND 20000;
```

3. Show those employee details whose hire_date is within the range ‘1987-01-01’ to ‘1987-06-30’ inclusive.

```
SELECT *
FROM employees
WHERE  hire_date BETWEEN '1987-01-01' AND '1987-06-30';
```

4. Show those employee details whose department_id is not within the range 50 to 60 inclusive.

```
SELECT *
FROM employees
WHERE  department_id NOT BETWEEN 50 AND 60;
```

5. Show those job details where the difference between max_salary and min _salary is within the range 5000 to
10000 inclusive.

```
SELECT*
FROM jobs
WHERE  max_salary-min_salary BETWEEN 5000 AND 10000;
```

6. Show those job_history details where the end_date is within the range ‘1998-12-01’ to ‘1998-12-31’ inclusive.

```
SELECT*
FROM job_history
WHERE end_date BETWEEN '1998-12-01' AND '1998-12-31';
```

CASE WHEN … WHEN … ELSE … END statement

1. Show all the past employees employee id, start date, job id and his group name from the job_history table:
Determine the group name as below:

Start Date	Group Name
on or before 1989-12-31	C
between 1994-12-31 to 1900-01-01 inclusive	B
after 1995-01-01 inclusive	A

ANS:

```
SELECT employee_id,
       start_date,
       job_id,
      CASE
         WHEN end_date<='1989-12-31' THEN 'C'
         WHEN end_date BETWEEN '1994-12-31' AND '1900-01-01' THEN 'B'
         WHEN end_date>'1995-01-01' THEN 'A'
      END AS group_name
FROM job_history ;
```

2. Show the department id, department name, location id and location group name from the departments table:
Determine the location group name as below:

Location Id	Location Group Name
less than 1200	C
between 2000 to 1200 inclusive	B
greater than 2000	A

ANS:

Numerical and String Functions

1. Show all the employees employee id and their short name in lowercase format.
Short name format: first 3 letters from the first name followed by an underscore and then followed by the first 3
letters of the last name.

```
SELECT employee_id,
      CONCAT(LEFT(LOWER(first_name),3), LEFT(LOWER(last_name),3))
FROM employees;
```

2. Show all those employee details whose name is a palindrome.

```
SELECT *
FROM employees
WHERE first_name= REVERSE(first_name);
```

3. Show all the employees employee id and email (i.e. add ‘@gmail.com’ at the end of each email).

```
SELECT employee_id,
      CONCAT(email, '@gmail.com')
FROM employees;
```

4. Show all the employees first name and phone number.
Phone number format: 515.xxx.xxx7 i.e. only show the first 4 characters and the last character and hide all the
intermediate characters with xxx.xxx

```
SELECT first_name,
      CONCAT(LEFT(phone_number,3), '.xxx.xxx',RIGHT(phone_number,1))
FROM employees;
```

5. Show all the employees employee id, email and full name.
Full name format: first_name<SPACE>last_name
Also show the full name in 20 characters if necessary right pad with necessary no of spaces.

```
SELECT employee_id,
      email,
      CONCAT(first_name,' ',last_name) AS full_name
FROM employees;
```

6. Show those location details from locations table whose postal code consists at most 5 characters and the first
two digits of its postal code is between 50 to 99 inclusive.

```
SELECT *
FROM locations
WHERE (LENGTH(postal_code)<=5 )
AND (LEFT(postal_code,2) BETWEEN 50 AND 99);
```

7. Show all the employees employee id, first name and his salary in “10 thousand 5 hundred and 12 taka only”
format.

```
SELECT first_name
      employee_id,
      CONCAT(floor(salary/1000), 'thousand',
           floor((salary%1000)/100) , 'hundred and',
            floor((salary%100)) ,'taka only') AS 'new salary'
 FROM employees;
```

8. For each job, show the job id, job title and how much greater the max_salary from its min_salary in percentage
format.
Note: Show the output in 2 decimal points

```
%greater=(max_salary-min_salary)*100/min_salary
```

```
SELECT job_id,
       job_title,
       CONCAT(floor(((max_salary-min_salary)*100)/min_salary),'%') AS 'percentage of diff'
       FROM jobs;
```

9. Show all those job details from jobs table whose salary range (i.e. max_salary-min_salary) is greater than 8000
and the job title contains the word ‘Manager’.

```
SELECT *
FROM jobs
WHERE (max_salary- min_salary)>8000 AND
      job_title LIKE '%manager%';
```

10. Show all the employees employee id, and his yearly total gross salary.
Note: Show the floor value of the total salary
    yearly total salary = salary * 12* (1+(commission_pct/100))]

```
SELECT employee_id,
      floor(salary * 12* ( 1+(commission_pct/100))) AS 'Yearly_salary'
      FROM employees;
```

11. Show those department details from departments table whose tens digits of location id is within the range 5 to 9
inclusive.

```
 SELECT*
 FROM departments
 WHERE LENGTH(location_id) BETWEEN '9' AND '5';
```

ORDER BY clause

1. Show all the employees first name, last name, email, hire date, salary in descending order of salary. If multiple
employees receive the same salary then also sort them based on the alphabetical order of their first name.

```
SELECT  first_name,
        last_name,
        email,
        hire_date
        FROM employees
        ORDER BY  salary DESC, first_name ASC;
```

2. Show all the employees employee id and their join date in such a way that the senior most employee comes first.
If multiple employees have the same join date then also sort them based on the descending order of their
department id.

```
SELECT employee_id,
       hire_date
FROM employees
ORDER BY hire_date ASC, employee_id DESC;
```

3. Show all the employees first name, email and phone number. Order the output based on the descending order of
first 3 digits of their phone number.

```
SELECT first_name,
       email,
       phone_number
FROM employees
ORDER BY LEFT(phone_number,3)DESC;
```

4. Show all the employees employee id, email, hire year (only the year portion) and hire month (show the full
month name). Show the output from most recent hired employee to old employees.

```
SELECT
   employee_id,
   email,
   DATE_FORMAT((hire_date),'%Y'),
   DATE_FORMAT((hire_date),'%M')
FROM employees
ORDER BY hire_date DESC;
```

5. Show all the job_history details in such a way that senior most employee data comes first and if multiple
employees have the same start date then also sort them based on the descending order of their end date.

```
SELECT *
FROM job_history
ORDER BY start_date ASC, end_date DESC;
```

6. Show all the jobs from jobs table where the highest salary range (i.e. max_salary-min_salary) job data comes
first.

```
SELECT *
FROM jobs
ORDER BY max_salary-min_salary DESC;
```

DISTINCT clause

1. Show all the distinct manager_ids from employees table.

```
SELECT DISTINCT manager_id
FROM employees;
```

2. Show all the distinct job_ids from the employees table.
3. Show all the distinct country_ids from the locations table.
4. Show all the distinct job_ids and department_ids from employees table.

LIMIT clause

1. Show the highest salary holder employee details from the employees table.

```
SELECT*
FROM employees
ORDER BY salary DESC
LIMIT 0,1;
```

2. Show the top 10 most experienced employee details from the employees table.

```
SELECT*
FROM employees
ORDER BY hire_date ASC
LIMIT 0,10;
```

3. Show the 2nd lowest salary range (i.e. max_salary-min_salary) job details from the jobs table.

```
SELECT*
FROM jobs
ORDER BY max_salary-min_salary ASC
LIMIT 1,1;
```

4. Show the top 3 lowest salary holder employee details from department number 60.

```
SELECT*
FROM employees
WHERE department_id = 60
ORDER BY salary ASC
LIMIT 0,3;
```

5. Among the employees supervised by manager id 108, find out the 2nd highest salary holder employee details.

```
SELECT*
FROM employees
WHERE manager_id = 108
ORDER BY salary DESC
LIMIT 1,1;
```

6. Among the employees whose job type is ‘ST_CLERK’, show the highest experienced employee id from the
job_history table.

```
SELECT employee_id
FROM job_history
WHERE job_id LIKE 'ST_CLERK'
```

ORDER BY start_date ASC
LIMIT 1,1;

Aggregate Operations (GROUP BY, HAVING clauses)

1. Show the total no of employees, their total salary, their average salary, their maximum salary, their minimum
salary from employees table.

SELECT count(employee_id),sum(salary),avg(salary),max(salary),min(salary)
FROM employees;

2. Show the maximum and minimum experienced employees hire dates from employees table
SELECT MAX(CURDATE()-hire_date),MIN(CURDATE()-hire_date)
FROM employees;

3. Show the maximum experienced employee hire date working in department number 50 from employees table.
SELECT MAX(hire_date)
FROM employees
WHERE department_id = 50;

4. Show the number of departments located in location id 1700 from departments table.

SELECT COUNT(department_id)
FROM departments
WHERE location_id=1700;

5. Show the most recent retired employee’s end date working in department number 80 from job history table.

SELECT MAX(end_date)

```
FROM job_history
WHERE department_id= 80;
```

6. Show the maximum and minimum value of min_salary column, maximum and minimum value of max_salary
column from jobs table.

```
SELECT
   MAX(min_salary) AS max_min_salary,
   MIN(min_salary) AS min_min_salary,
   MAX(max_salary) AS max_max_salary,
   MIN(max_salary) AS min_max_salary
FROM
   jobs;
```

7. Count the number of employees managed by manager id 114 from employees table.

```
SELECT
   COUNT(*) AS num_employees_managed
FROM
   employees
WHERE
   manager_id = 114;
```

8. Count the total number of distinct job_ids from employees table

```
 SELECT COUNT(DISTINCT job_id)
 FROM employees ;
```

9. Count the distinct number of countries from locations table.
10. Count the total number of locations located in ‘US’ from locations table.

```
SELECT COUNT(country_id)
```

```
FROM locations
WHERE country_id='US';
```

11. Show the maximum and minimum salary range value (i.e. salary range = max_salary - min_salary) from jobs table.

```
SELECT
    MAX(max_salary - min_salary) AS max_salary_range,
    MIN(max_salary - min_salary) AS min_salary_range
FROM
    jobs;
```

12. Count the number of employees whose employee id is greater than his manager id.

```
SELECT
    COUNT(*) AS num_employees
FROM
    employees
WHERE
    employee_id > manager_id;
```

1. Show each region_id and corresponding no of countries in that region from countries table.

```
SELECT region_id, COUNT(country_id)
FROM countries
GROUP BY region_id;
```

2. Show the location_id and corresponding no of departments in that location from departments table.
3. For each department_id, show the no of employees in that department from employees table.
4. For each manager_id, show the no of employees under his supervision from employees table.

5. For each job_id, show the no of employees in that job type from employees table.
6. For each department_id, show the no of managers from that department using employees table.
7. Count the total number of employees joined in the even month and total number of employees joined in the odd
number months from the employees table.
----
2. Show the location_id and corresponding number of departments in that location from the departments table:

```
SELECT
   location_id,
   COUNT(*) AS num_departments
FROM
   departments
GROUP BY
   location_id;
```

3. For each department_id, show the number of employees in that department from the employees table:

```
SELECT
   department_id,
   COUNT(*) AS num_employees
FROM
   employees
GROUP BY
   department_id;
```

4. For each manager_id, show the number of employees under his supervision from the employees table:

```
SELECT
   manager_id,
   COUNT(*) AS num_employees
FROM
   employees
GROUP BY
```

```
    manager_id;
```

5. For each job_id, show the number of employees in that job type from the employees table:

```
SELECT
    job_id,
    COUNT(*) AS num_employees
FROM
    employees
GROUP BY
    job_id;
```

6. For each department_id, show the number of managers from that department using the employees table:

```
SELECT
    department_id,
    COUNT(*) AS num_managers
FROM
    employees
WHERE
    job_id = 'MANAGER'
GROUP BY
    department_id;
```

7. Count the total number of employees joined in the even month and the total number of employees joined in the odd number months from the employees table:

```
SELECT
    CASE WHEN MONTH(hire_date) % 2 = 0 THEN 'Even' ELSE 'Odd' END AS month_type,
    COUNT(*) AS num_employees
FROM
    employees
GROUP BY
    month_type;
```

----

8. Show the department wise total no of employees, maximum and minimum salary in that department, average
and total salary provided by that department from the employees table.

9. For each year, show the total no of employees who were hired during that year from the employees table.

```
SELECT count(employee_id)
FROM employees
GROUP BY year(hire_date);
```

10. Show the total no of jobs within 0k to 10k, 10k to 20k and so on salary ranges(max_salary-min_salary) groups
from the jobs table.
11. For each country_id, show the total no of locations in that country from the locations table.
12. For each city, show the total no of locations in that city from the locations table.

13. Group and count employees based on the first letter of their names. (max 26 groups as 26 alphabets)

```
SELECT first_name,count(employee_id)
FROM employees
GROUP BY SUBSTR(first_name,1,1) ;
```

14. For each job_id and each department, show the total no of employees in that group from the employees table.
15. For each year and each month, show the total no of employees who have left their jobs from the job_history
table.

Show only those department_ids where the total salary expense is more than $100,000.
sql
Copy code

```
SELECT
    department_id
FROM
```

```
    employees
GROUP BY
    department_id
HAVING
    SUM(salary) > 100000;
```
Count the total number of employees for each department, excluding employees with job_id "AD_PRESS" and considering only departments with more than 5 employees.
sql
Copy code
```
SELECT
    department_id,
    COUNT(*) AS total_employees
FROM
    employees
WHERE
    job_id != 'AD_PRESS'
GROUP BY
    department_id
HAVING
    total_employees > 5;
```
Group employees based on the first 3 digits of their phone number, excluding employees from departments 10, 20, and 60, and excluding groups where the total salaries of employees are less than $50,000.
sql
Copy code
```
SELECT
    SUBSTRING(phone_number, 1, 3) AS phone_prefix,
    COUNT(*) AS total_employees,
    SUM(salary) AS total_salary
FROM
    employees
WHERE
    department_id NOT IN (10, 20, 60)
GROUP BY
    phone_prefix
HAVING
    total_salary > 50000;
```

For each year and each month, count the total number of employees joined, excluding years and months where the total number of hired employees is less than 20.
sql
Copy code
```
SELECT
   YEAR(hire_date) AS hire_year,
   MONTH(hire_date) AS hire_month,
   COUNT(*) AS total_employees_joined
FROM
   employees
GROUP BY
   hire_year,
   hire_month
HAVING
   total_employees_joined >= 20;
```
For each country and each city, count the total number of locations, excluding locations from the city 'US' and countries and cities with less than 5 locations.
sql
Copy code
```
SELECT
   country_name,
   city,
   COUNT(*) AS total_locations
FROM
   locations
WHERE
   city != 'US'
GROUP BY
   country_name,
   city
HAVING
   total_locations >= 5;
```

----

1. Show only those manager_ids who handle more than 5 employees.

```
SELECT manager_id
FROM employees
GROUP BY manager_id
HAVING COUNT(employee_id)>5;
```

2. Show only those department_ids where in total salary expense is more than 100000 dollar.
3. Count the total no of employees for each department. Don’t consider employees of job_id “AD_PRESS” and also
consider only those departments where total no of employees is greater than 5.
4. Group employees based on the first 3 digit of their phone number. Avoid employees from department no
10/20/60 and also avoid those groups where total salaries of employees is less than 50000.

5. For each year and each month, count total number of employees joined from employees table. Don’t consider
those year and months where total number of hired employees are less than 20.
6. For each country and each city, count total number of locations from locations table. Don’t consider locations
from city ‘US’ and also don’t consider those country and city having less than 5 locations.

Table Join Operations (JOIN, LEFT JOIN clauses)

1. Show the region_name and corresponding country_name

```
SELECT c.country_name,r.region_name
FROM countries AS c
    JOIN
    regions AS r
    ON c.region_id=r.region_id;
```

2. Show the department_name and corresponding country_name.
3. Show the employee_name and his job place country_name.
4. Show the employee_name and his job_title.

5. Show the employee_name and his manager_name
6. Show the department_name and the manager_name of corresponding department.
7. Show the employee_id, his salary, his manager_id, his manager_name, his manager_salary.

8. Show the employee_id, his join_date, his manager_id, his manager_name, his manager_salary.

```
SELECT e.employee_id,e.hire_date,e.manager_id,m.first_name,m.salary
FROM employees AS e
    JOIN employees AS m
    ON e.employee_id=m.manager_id;
```

9. Show the manger_name and his manager_name (manager of manager).

```
SELECT m.first_name,mm.first_name
FROM employees AS e
    JOIN employees AS m
    ON e.employee_id=m.employee_id
    JOIN employees AS mm
    ON m.employee_id= mm.employee_id;
```

10. Show the employee name and his manager name only for those employees who have joined after this manager.
11. Show the employees name and other employees name who receives higher salary than him
12. Show the employees name and other employees name who is hired after him.
13. For each region, show the region_name and total no of employees in that region.

Show the employee name and his manager name only for those employees who have joined after their manager.
sql
Copy code
```
SELECT
    e1.employee_name AS employee_name,
    e2.employee_name AS manager_name
FROM
    employees e1
JOIN
    employees e2 ON e1.manager_id = e2.employee_id
WHERE
    e1.join_date > e2.join_date;
```
Show the employees name and other employees name who receive a higher salary than them.
sql
Copy code
```
SELECT
    e1.employee_name AS employee_name,
    e2.employee_name AS other_employee_name
FROM
    employees e1
JOIN
    employees e2 ON e1.salary < e2.salary;
```
Show the employees name and other employees name who were hired after them.
sql
Copy code
```
SELECT
    e1.employee_name AS employee_name,
    e2.employee_name AS other_employee_name
FROM
    employees e1
JOIN
    employees e2 ON e1.join_date < e2.join_date;
```
For each region, show the region_name and total number of employees in that region.
sql
Copy code
```
SELECT
```

```
    r.region_name,
    COUNT(*) AS total_employees
FROM
    employees e
JOIN
    departments d ON e.department_id = d.department_id
JOIN
    locations l ON d.location_id = l.location_id
JOIN
    regions r ON l.region_id = r.region_id
GROUP BY
    r.region_name;
```
----

14. For each job, show the job_title and total no of employees.

```
SELECT j.job_title,COUNT(e.employee_id)
FROM employees AS e
     JOIN jobs AS j
     ON e.job_id=j.job_id
GROUP BY j.job_title;
```

15. For each country, show the total no of departments in that country.
16. For each department, show the department_name and corresponding total no of ex-employees (job_history
table) from that department.
17. For each manager, show the manager_name and total no of employees under his supervision.
18. For each manager, show the manager_name and total no of employees under his supervision who receives
higher salary than him.
----
15. For each country, show the total number of departments in that country.
sql
Copy code

```
SELECT l.country_name, COUNT(d.department_id) AS total_departments
FROM locations l
JOIN departments d ON l.location_id = d.location_id
GROUP BY l.country_name;
```

16. For each department, show the department_name and corresponding total number of ex-employees (job_history table) from that department.

sql

Copy code

```
SELECT d.department_name, COUNT(jh.employee_id) AS total_ex_employees
FROM departments d
LEFT JOIN job_history jh ON d.department_id = jh.department_id
GROUP BY d.department_name;
```

17. For each manager, show the manager_name and total number of employees under his supervision.

sql

Copy code

```
SELECT CONCAT(m.first_name, ' ', m.last_name) AS manager_name, COUNT(e.employee_id)
AS total_employees
FROM employees e
JOIN employees m ON e.manager_id = m.employee_id
GROUP BY m.employee_id;
```

18. For each manager, show the manager_name and total number of employees under his supervision who receive a higher salary than him.

sql

Copy code

```
SELECT CONCAT(m.first_name, ' ', m.last_name) AS manager_name,
      COUNT(e.employee_id) AS total_higher_salary_employees
FROM employees e
JOIN employees m ON e.manager_id = m.employee_id
WHERE e.salary > m.salary
GROUP BY m.employee_id;
```

```
SELECT m.first_name, count(e.employee_id)
FROM employees AS e
    JOIN employees AS m
    ON e.employee_id=m.manager_id
GROUP BY m.manager_id
```

HAVING e.salary>m.salary;

19. Show the employee name and no of employees who receives lower salary than him.
20. Show the employee name and no of employees who is hired before him.
------
19. Show the employee name and number of employees who receive a lower salary than him.
sql
Copy code
```
SELECT CONCAT(e1.first_name, ' ', e1.last_name) AS employee_name,
      COUNT(e2.employee_id) AS num_employees_lower_salary
FROM employees e1
JOIN employees e2 ON e1.salary < e2.salary
GROUP BY e1.employee_id;
```
20. Show the employee name and number of employees who were hired before him.
sql
Copy code
```
SELECT CONCAT(e1.first_name, ' ', e1.last_name) AS employee_name,
      COUNT(e2.employee_id) AS num_employees_hired_before
FROM employees e1
JOIN employees e2 ON e1.hire_date > e2.hire_date
GROUP BY e1.employee_id;
```

Subquery

Retrieve the first name, last name, and email of all employees.

```
SELECT first_name, last_name, email FROM employees;
```

List the job titles and salaries of all jobs.

```
SELECT job_title, salary FROM jobs;
```

Display the country names and their corresponding region names.

```
SELECT country_name, region_name FROM countries
JOIN regions ON countries.region_id = regions.region_id;
```

Find the number of employees in each department.

```
SELECT department_id, COUNT(employee_id) AS num_employees FROM employees
GROUP BY department_id;
```

Show the street address, city, and postal code of all locations.

```
SELECT street_address, city, postal_code
 FROM locations;
```

Get the total number of employees in the company.

```
SELECT COUNT(employee_id) AS total_employees
FROM employees;
```

List all employees hired after January 1, 2020.

```
SELECT * FROM employees
WHERE hire_date > '2020-01-01';
```

Display the department names and the number of employees in each department.

```
SELECT departments.department_name, COUNT(employees.employee_id) AS num_employees
FROM departments
LEFT JOIN employees ON departments.department_id = employees.department_id
GROUP BY departments.department_id;
```

Find the average salary of all employees.

```
SELECT AVG(salary) AS average_salary FROM employees;
```

Show the job titles and the maximum salary for each job.

```
SELECT job_title, MAX(salary) AS max_salary FROM jobs
GROUP BY job_title;
```

Retrieve the employee ID, first name, last name, and salary of employees earning more than $5000.

```sql
SELECT employee_id, first_name, last_name, salary FROM employees
WHERE salary > 5000;
```

List the department names and their respective manager IDs.

```sql
SELECT department_name, manager_id FROM departments;
```

Get the employee IDs, first names, and last names of employees with no manager.

```sql
SELECT employee_id, first_name, last_name FROM employees
WHERE manager_id IS NULL;
```

Show the job titles and their corresponding minimum salary.

```sql
SELECT job_title, MIN(salary) AS min_salary FROM jobs
GROUP BY job_title;
```

Find the employee with the highest salary.

```sql
SELECT * FROM employees
WHERE salary = (SELECT MAX(salary) FROM employees);
```

Retrieve the employee IDs, first names, last names, and commission percentages of employees with a commission percentage.

```
SELECT employee_id, first_name, last_name, commission_pct FROM employees
WHERE commission_pct IS NOT NULL;
```

List the department IDs and names where the department has no manager.

```
SELECT department_id, department_name FROM departments
WHERE manager_id IS NULL;
```

Display the job titles and the difference between the maximum and minimum salaries for each job.

```
SELECT job_title, (MAX(salary) - MIN(salary)) AS salary_difference FROM jobs
GROUP BY job_title;
```

Find the employee with the longest tenure in the company.

```
SELECT * FROM employees
WHERE hire_date = (SELECT MIN(hire_date) FROM employees);
```

Show the job IDs and titles of jobs with no minimum or maximum salary specified.

```
SELECT job_id, job_title FROM jobs
WHERE min_salary IS NULL OR max_salary IS NULL;
```

List the department IDs and names along with the city and country of their corresponding locations.

```
SELECT departments.department_id, departments.department_name, locations.city,
countries.country_name
FROM departments
JOIN locations ON departments.location_id = locations.location_id
JOIN countries ON locations.country_id = countries.country_id;
```

Retrieve the employee IDs and names of employees who do not belong to any department.

```
SELECT employee_id, first_name, last_name FROM employees
WHERE department_id IS NULL;
```

Show the job IDs and titles of jobs where the title contains 'Manager'.

```
SELECT job_id, job_title FROM jobs
WHERE job_title LIKE '%Manager%';
```

Find the department with the highest average salary.

```
SELECT department_id, AVG(salary) AS avg_salary FROM employees
GROUP BY department_id
ORDER BY avg_salary DESC
LIMIT 1;
```

Display the employee IDs, first names, last names, and hire dates of employees hired in 2022.

```
SELECT employee_id, first_name, last_name, hire_date FROM employees
WHERE hire_date BETWEEN '2022-01-01' AND '2022-12-31';
```

List the job titles and the number of employees in each job.

```
SELECT job_title, COUNT(employee_id) AS num_employees FROM employees
GROUP BY job_title;
```

Get the employee IDs and names of employees who started after their managers.

```
SELECT e.employee_id, e.first_name, e.last_name
FROM employees e
JOIN employees m ON e.manager_id = m.employee_id
WHERE e.hire_date > m.hire_date;
```

Show the department names and the total salary of employees in each department.

```
SELECT departments.department_name, SUM(employees.salary) AS total_salary
FROM departments
JOIN employees ON departments.department_id = employees.department_id
GROUP BY departments.department_name;
```

Retrieve the employee IDs and names of employees whose first name starts with 'J'.

```
SELECT employee_id, first_name, last_name FROM employees
WHERE first_name LIKE 'J%';
```

Display the job IDs and titles of jobs where the minimum salary is less than the average salary of all jobs.

```
SELECT job_id, job_title FROM jobs
WHERE min_salary < (SELECT AVG(salary) FROM jobs);
```

Find the department with the lowest total salary.

```
SELECT department_id, SUM(salary) AS total_salary FROM employees
GROUP BY department_id
ORDER BY total_salary ASC
LIMIT 1;
```

List the job IDs and titles of jobs where the title ends with 'Analyst'.

```
SELECT job_id, job_title FROM jobs
WHERE job_title LIKE '%Analyst';
```

Show the employee IDs, first names, last names, and hire dates of employees hired before their managers.

```
SELECT e.employee_id, e.first_name, e.last_name, e.hire_date
```

```
FROM employees e
JOIN employees m ON e.manager_id = m.employee_id
WHERE e.hire_date < m.hire_date;
```

Retrieve the employee IDs, first names, last names, and hire dates of employees hired in 2020 or 2021.

```
SELECT employee_id, first_name, last_name, hire_date FROM employees
WHERE hire_date BETWEEN '2020-01-01' AND '2021-12-31';
```

Display the job IDs and titles of jobs where the maximum salary is greater than the average salary of all jobs.

```
SELECT job_id, job_title FROM jobs
WHERE max_salary > (SELECT AVG(salary) FROM jobs);
```

Find the department with the highest number of employees.

```
SELECT department_id, COUNT(employee_id) AS num_employees FROM employees
GROUP BY department_id
ORDER BY num_employees DESC
LIMIT 1;
```

List the job IDs and titles of jobs where the minimum salary is not equal to the maximum salary.

```
SELECT job_id, job_title FROM jobs
WHERE min_salary != max_salary;
```

Show the department names and the total number of employees in each department where the number of employees is greater than 5.

```
SELECT departments.department_name, COUNT(employees.employee_id) AS num_employees
FROM departments
LEFT JOIN employees ON departments.department_id = employees.department_id
GROUP BY departments.department_id
HAVING num_employees > 5;
```

Retrieve the employee IDs, first names, last names, and salaries of employees whose salary is not within the range $3000 to $6000.

```
SELECT employee_id, first_name, last_name, salary FROM employees
WHERE salary NOT BETWEEN 3000 AND 6000;
```

Display the job IDs and titles of jobs where the difference between the minimum and maximum salary is greater than $5000.

```
SELECT job_id, job_title FROM jobs
WHERE (max_salary - min_salary) > 5000;
```

Find the department with the highest total commission.

```
SELECT department_id, SUM(commission_pct) AS total_commission FROM employees
GROUP BY department_id
```

```
ORDER BY total_commission DESC
LIMIT 1;
```

List the job IDs and titles of jobs where the title contains either 'Engineer' or 'Developer'.

```
SELECT job_id, job_title FROM jobs
WHERE job_title LIKE '%Engineer%' OR job_title LIKE '%Developer%';
```

Show the department names and the average salary of employees in each department where the average salary is less than $7000.

```
SELECT departments.department_name, AVG(employees.salary) AS avg_salary
FROM departments
JOIN employees ON departments.department_id = employees.department_id
GROUP BY departments.department_id
HAVING avg_salary < 7000;
```

Retrieve the employee IDs, first names, last names, and salaries of employees whose salary is the maximum salary in their department.

```
SELECT e.employee_id, e.first_name, e.last_name, e.salary
FROM employees e
JOIN (
 SELECT department_id, MAX(salary) AS max_salary
 FROM employees
 GROUP BY department_id
) max_salaries ON e.department_id = max_salaries.department_id AND e.salary =
max_salaries.max_salary;
```

Display the job IDs and titles of jobs where the maximum salary is less than the minimum salary of any other job.

```
SELECT job_id, job_title FROM jobs
WHERE max_salary < ANY (SELECT min_salary FROM jobs WHERE min_salary IS NOT NULL);
```

Find the department with the highest number of managers.

```
SELECT department_id, COUNT(manager_id) AS num_managers FROM employees
WHERE manager_id IS NOT NULL
GROUP BY department_id
ORDER BY num_managers DESC
LIMIT 1;
```

List the job IDs and titles of jobs where the minimum salary is not specified but the maximum salary is.

```
SELECT job_id, job_title FROM jobs
WHERE min_salary IS NULL AND max_salary IS NOT NULL;
```

Show the department names and the average commission percentage of employees in each department where the average commission percentage is greater than 0.

```
SELECT departments.department_name, AVG(employees.commission_pct) AS
avg_commission
FROM departments
JOIN employees ON departments.department_id = employees.department_id
```

```
GROUP BY departments.department_id
HAVING avg_commission > 0;
```

Retrieve the employee IDs, first names, last names, and hire dates of employees who were hired after their managers but before January 1, 2023.

```
SELECT e.employee_id, e.first_name, e.last_name, e.hire_date
FROM employees e
JOIN employees m ON e.manager_id = m.employee_id
WHERE e.hire_date > m.hire_date AND e.hire_date < '2023-01-01';
```

Display the job IDs and titles of jobs where the minimum salary is greater than the maximum salary of any other job.

```
SELECT job_id, job_title FROM jobs j
WHERE min_salary > ALL (SELECT max_salary FROM jobs WHERE max_salary IS NOT NULL);
```

View