

Select distinct in SQL

26. Display the different position available for employee.

select distinct position from employee;

```
MariaDB [labfour]> select distinct address from employee;
+-----+
| address |
+-----+
| kathmandu |
| pokhara |
| butwal |
| chitwan |
| lalitpur |
+-----+
5 rows in set (0.002 sec)
```

27. List out the unique address available for employee table.

select distinct address from employee;

```
MariaDB [labfour]> select distinct address from employee;
+-----+
| address |
+-----+
| kathmandu |
| pokhara |
| butwal |
| chitwan |
| lalitpur |
+-----+
5 rows in set (0.002 sec)
```

28. List out the employee who have unique first_name and address.

select distinct first_name,address from employee;

```
MariaDB [labfour]> select distinct first_name,address from employee;
+-----+-----+
| first_name | address |
+-----+-----+
| anish | kathmandu |
| roshan | pokhara |
| aakriti | butwal |
| rojina | pokhara |
| keshav | kathmandu |
| roshan | chitwan |
| sita | lalitpur |
| srijana | butwal |
| niraj | kathmandu |
+-----+-----+
```

AS

29. Write a query to get first_name,last_name ,ssf of all employees .ssf is calculated as 31% of salary.

select first_name,last_name, salary*0.31 as ssf from employee;

```
MariaDB [labfour]> select first_name,last_name,salary*0.31 as ssf from employee;
```

first_name	last_name	ssf
anish	sharma	27094.0000
roshan	pokhrel	20320.5000
aakriti	bagale	32174.1250
rojina	karki	28787.3750
keshav	ghimire	22013.8750
roshan	pandey	23707.2500
sita	pokhrel	23029.9000
srijana	bhattra	20997.8500
niraj	acharya	30480.7500

30. Write a query to get the employee _id, name (first_name, last_name), location (address) from employee.

select employee_id ,concat(first_name,' ',last_name) as name ,address as location from employee;

```
MariaDB [labfour]> select employee_id,concat(first_name,' ',last_name)as name, address as location from employee;
```

employee_id	name	location
1	anish sharma	kathmandu
2	roshan pokhrel	pokhara
3	aakriti bagale	butwal
4	rojina karki	pokhara
5	keshav ghimire	kathmandu
6	roshan pandey	chitwan
7	sita pokhrel	lalitpur
8	srijana bhattra	butwal
9	niraj acharya	kathmandu

ORDER BY

31. Display the information of employees in ascending order by address.

select * from employee order by address ; or select * from employee order by address asc;

```
MariaDB [labfour]> select * from employee order by address;
```

employee_id	first_name	last_name	age	address	department	position	salary
3	aakriti	bagale	30	butwal	purchase	manager	103787.5000
8	srijana	bhattra	29	butwal	finance	analyst	67735.0000
6	roshan	pandey	38	chitwan	operations	analyst	76475.0000
1	anish	sharma	26	kathmandu	finance	manager	87400.0000
5	keshav	ghimire	35	kathmandu	purchase	analyst	71012.5000
9	niraj	acharya	40	kathmandu	sales	manager	98325.0000
7	sita	pokhrel	23	lalitpur	marketing	analyst	74290.0000
2	roshan	pokhrel	28	pokhara	sales	analyst	65550.0000
4	rojina	karki	25	pokhara	marketing	manager	92862.5000

9 rows in set (0.001 sec)

32. Display the information of employees in descending order by address.

select * from employee order by address desc;

```
MariaDB [labfour]> select * from employee order by address desc;
```

employee_id	first_name	last_name	age	address	department	position	salary
2	roshan	pokhrel	28	pokhara	sales	analyst	65550.0000
4	rojina	karki	25	pokhara	marketing	manager	92862.5000
7	sita	pokhrel	23	lalitpur	marketing	analyst	74290.0000
1	anish	sharma	26	kathmandu	finance	manager	87400.0000
5	keshav	ghimire	35	kathmandu	purchase	analyst	71012.5000
9	niraj	acharya	40	kathmandu	sales	manager	98325.0000
6	roshan	pandey	38	chitwan	operations	analyst	76475.0000
3	aakriti	bagale	30	butwal	purchase	manager	103787.5000
8	srijana	bhattra	29	butwal	finance	analyst	67735.0000

9 rows in set (0.001 sec)

33. Display the information of employees in ascending order by address and department.

select * from employee order by address,department;

```
MariaDB [labfour]> select * from employee order by address,department;
```

employee_id	first_name	last_name	age	address	department	position	salary
8	srijana	bhattra	29	butwal	finance	analyst	67735.0000
3	aakriti	bagale	30	butwal	purchase	manager	103787.5000
6	roshan	pandey	38	chitwan	operations	analyst	76475.0000
1	anish	sharma	26	kathmandu	finance	manager	87400.0000
5	keshav	ghimire	35	kathmandu	purchase	analyst	71012.5000
9	niraj	acharya	40	kathmandu	sales	manager	98325.0000
7	sita	pokhrel	23	lalitpur	marketing	analyst	74290.0000
4	rojina	karki	25	pokhara	marketing	manager	92862.5000
2	roshan	pokhrel	28	pokhara	sales	analyst	65550.0000

9 rows in set (0.001 sec)

Aggregate functions

34. Count the number of employees.

select count(*) from employee;

```
MariaDB [labfour]> select count(*) from employee;
+-----+
| count(*) |
+-----+
|          9 |
+-----+
1 row in set (0.001 sec)
```

35. Count the number of unique first_name of employees.

select count(distinct first_name) from employee;

```
MariaDB [labfour]> select count(distinct first_name) from employee;
+-----+
| count(distinct first_name) |
+-----+
|                             8 |
+-----+
1 row in set (0.001 sec)
```

36. To get the number of different number of positions available for employees table.

select count(distinct position) from employee;

```
MariaDB [labfour]> select count(distinct position) from employee;
+-----+
| count(distinct position) |
+-----+
|                           2 |
+-----+
1 row in set (0.001 sec)
```

37. To get the total salaries payable to employees.

select sum(salary) from employee;

```
MariaDB [labfour]> select sum(salary) from employee;
+-----+
| sum(salary) |
+-----+
| 737437.5000 |
+-----+
1 row in set (0.001 sec)
```

38. Find the average salary of employess.

select avg(salary) from employee;

```
MariaDB [labfour]> select avg(salary) from employee;
+-----+
| avg(salary) |
+-----+
| 81937.5000000 |
+-----+
1 row in set (0.001 sec)
```

39. Find the minimum salary of employess.

select min(salary) from employee;

```
MariaDB [labfour]> select min(salary) from employee;
+-----+
| min(salary) |
+-----+
| 65550.0000 |
+-----+
1 row in set (0.001 sec)
```

40. Display first_name, last_name of employees with highest salary.

select first_name,last_name from employee where salary=(select max(salary) from employee);

```
MariaDB [labfour]> select first_name,last_name from employee where salary=(select max(salary) from employee);
+-----+-----+
| first_name | last_name |
+-----+-----+
| aakriti    | bagale    |
+-----+-----+
1 row in set (0.002 sec)
```

41. Display first_name,last_name,department,postion whose salary is less than average salary of all employees.

select first_name,last_name,department,position from employee where salary<(select avg(salary) from employee);

```
MariaDB [labfour]> select first_name,last_name,department,position from employee where salary<(select avg(salary)from employee);
+-----+-----+-----+-----+
| first_name | last_name | department | position |
+-----+-----+-----+-----+
| roshan     | pokhrel  | sales      | analyst  |
| keshav     | ghimire  | purchase  | analyst  |
| roshan     | pandey   | operations | analyst  |
| sita       | pokhrel  | marketing  | analyst  |
| srijana    | bhattra  | finance    | analyst  |
+-----+-----+-----+-----+
5 rows in set (0.001 sec)
```

Group by and having clause

42. Find the average salary of employees in each department.

select department,avg(salary) as average_salary from employee group by department;

```
MariaDB [labfour]> select department,avg(salary) as average_salary from employee group by department;
```

department	average_salary
finance	77567.50000000
marketing	83576.25000000
operations	76475.00000000
purchase	87400.00000000
sales	81937.50000000

43.Find the average salary of employees for each position.

select position,avg(salary) as average_salary from employee group by position;

```
MariaDB [labfour]> select department,avg(salary) as average_salary from employee group by position;
```

department	average_salary
sales	71012.50000000
finance	95593.75000000

44.Find the department with their average salary is greater than 60000.

select department ,avg(salary) from employee group by department having avg(salary)>60000;

```
MariaDB [labfour]> select department ,avg(salary)
-> from employee
-> group by department
-> having avg(salary)>60000;
```

department	avg(salary)
finance	77567.50000000
marketing	83576.25000000
operations	76475.00000000
purchase	87400.00000000
sales	81937.50000000

5 rows in set (0.002 sec)

45. Find the position of the employee in which average salary of position is greater than 60000.

select position from employee group by position having avg(salary)>60000;

```
MariaDB [labfour]> select position from employee group by position having avg(salary)>60000;
```

position
analyst
manager

2 rows in set (0.032 sec)

Sub-Queries with SELECT statement

46) Display information of employee whose salary is greater than average salary of all employees.

select * from employee where salary>(select avg(salary) from employee);

```
MariaDB [labfour]> select *
-> from employee
-> where salary>(select avg(salary) from employee);
```

employee_id	first_name	last_name	age	address	department	position	salary
1	anish	sharma	26	kathmandu	finance	manager	87400.0000
3	aakriti	bagale	30	butwal	purchase	manager	103787.5000
4	rojina	karki	25	pokhara	marketing	manager	92862.5000
9	niraj	acharya	40	kathmandu	sales	manager	98325.0000

4 rows in set (0.002 sec)

47) Display information of employee whose salary is greater than at least one employee of Finance department.

select * from employee where salary>some(select salary from employee where department ='finance');

```
MariaDB [labfour]> select * from employee
-> where salary>some(select salary from employee where department ='finance');
```

employee_id	first_name	last_name	age	address	department	position	salary
1	anish	sharma	26	kathmandu	finance	manager	87400.0000
3	aakriti	bagale	30	butwal	purchase	manager	103787.5000
4	rojina	karki	25	pokhara	marketing	manager	92862.5000
5	keshav	ghimire	35	kathmandu	purchase	analyst	71012.5000
6	roshan	pandey	38	chitwan	operations	analyst	76475.0000
7	sita	pokhrel	23	lalitpur	marketing	analyst	74290.0000
9	niraj	acharya	40	kathmandu	sales	manager	98325.0000

7 rows in set (0.073 sec)

Subqueries with UPDATE statement

48) Increase salary of employees by 10% whose age is greater than 28.

update employee set salary=salary*1.1 where age in (select age from employee where age>28);

```
MariaDB [labfour]> update employee
-> set salary=salary*1.1
-> where age in (select age from employee where age>28);
Query OK, 5 rows affected (0.162 sec)
Rows matched: 5 Changed: 5 Warnings: 0

MariaDB [labfour]> select * from employee;
```

employee_id	first_name	last_name	age	address	department	position	salary
1	anish	sharma	26	kathmandu	finance	manager	87400.0000
2	roshan	pokhrel	28	pokhara	sales	analyst	65550.0000
3	aakriti	bagale	30	butwal	purchase	manager	114166.2500
4	rojina	karki	25	pokhara	marketing	manager	92862.5000
5	keshav	ghimire	35	kathmandu	purchase	analyst	78113.7500
6	roshan	pandey	38	chitwan	operations	analyst	84122.5000
7	sita	pokhrel	23	lalitpur	marketing	analyst	74290.0000
8	srijana	bhattarai	29	butwal	finance	analyst	74508.5000
9	niraj	acharya	40	kathmandu	sales	manager	108157.5000

```
9 rows in set (0.001 sec)
```

Subqueries with DELETE statement

49) Delete the information of employees whose age is less than 35.

delete from employee where age in (select age from employee where age<35);

```
MariaDB [labfour]> delete from employee where age in (select age from employee where age<35);
Query OK, 6 rows affected (0.100 sec)

MariaDB [labfour]> select * from employee;
```

employee_id	first_name	last_name	age	address	department	position	salary
5	keshav	ghimire	35	kathmandu	purchase	analyst	78113.7500
6	roshan	pandey	38	chitwan	operations	analyst	84122.5000
9	niraj	acharya	40	kathmandu	sales	manager	108157.5000

• DISCUSSION:

In this lab, we explored essential components of SQL, including DISTINCT, AS, ORDER BY, Aggregate Functions, GROUP BY, HAVING clause, and Sub-Queries.

Distinct allowed us to retrieve unique values from a column, eliminating duplicates and streamlining data output. AS clause enabled us to rename columns or tables, enhancing readability and simplifying complex queries. 'Order By' facilitated sorting data either in ascending or descending order, providing control over result presentation. Functions like SUM, AVG, MIN, MAX, and COUNT helped in performing calculations on groups of rows, summarizing data efficiently. 'GROUP BY' allowed us to group rows based on a common attribute, aiding in analyzing data in subsets. 'HAVING clause' worked in conjunction with GROUP BY, applying conditions to grouped data, filtering results effectively. By nesting queries within queries, we gained the ability to perform complex operations and retrieve specific data subsets.

- **Conclusion:**

Hence it is concluded that the different operator that we have used in DBMS LAB-4 like DISTINCT, AS, ORDER BY, Aggregate Functions, GROUP BY, HAVING clause, and Sub-Queries offer powerful tools to manipulate and extract meaningful information from databases. Their combined use allows for precise data retrieval, organization, and analysis, significantly enhancing the efficiency and accuracy of database queries and operations.