**SQL Notes and Practice**

Correct key word order again

1. SELECT 2. FROM 3. JOIN (ON) 4. WHERE 5. GROUP BY 6. HAVING 7. ORDER BY 8. LIMIT

Frankly speaking SQL is something that's impossible to study , i.e it's better to know simple syntax and move forward with practice.

You show know basic syntax like

1) select 2) from 3) where 4) having 5) union and union all 6) group by 7) joins that's it ...

Next use the above understanding and try to do as much as possible.

Spend good time in the thought process and then write the code

* Learn form Interview bit - [javatPoint](https://www.javatpoint.com/sql-tutorial)  - [W3school](https://www.w3schools.com/sql/)  - [Guru99](https://www.guru99.com/sql.html)
* <https://www.youtube.com/watch?v=7S_tz1z_5bA&ab_channel=ProgrammingwithMosh>
* Hacker rank
* <https://sqlzoo.net/wiki/SQL_Tutorial>
* <https://github.com/KangboLu/Learn-SQL-Notes>
* [PRACTICE](https://www.databasestar.com/sql-practice/)- HackerRank, Leetcode, [w3Resource](https://www.w3resource.com/sql/tutorials.php)

Practice sql queries from leetcode

SQL → Features - JOIN - KEYS

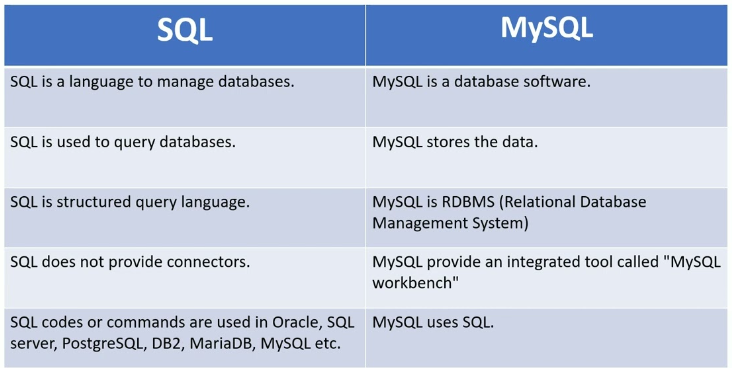
All the keywords - SQL Aggregation - Subqueries

# SQL

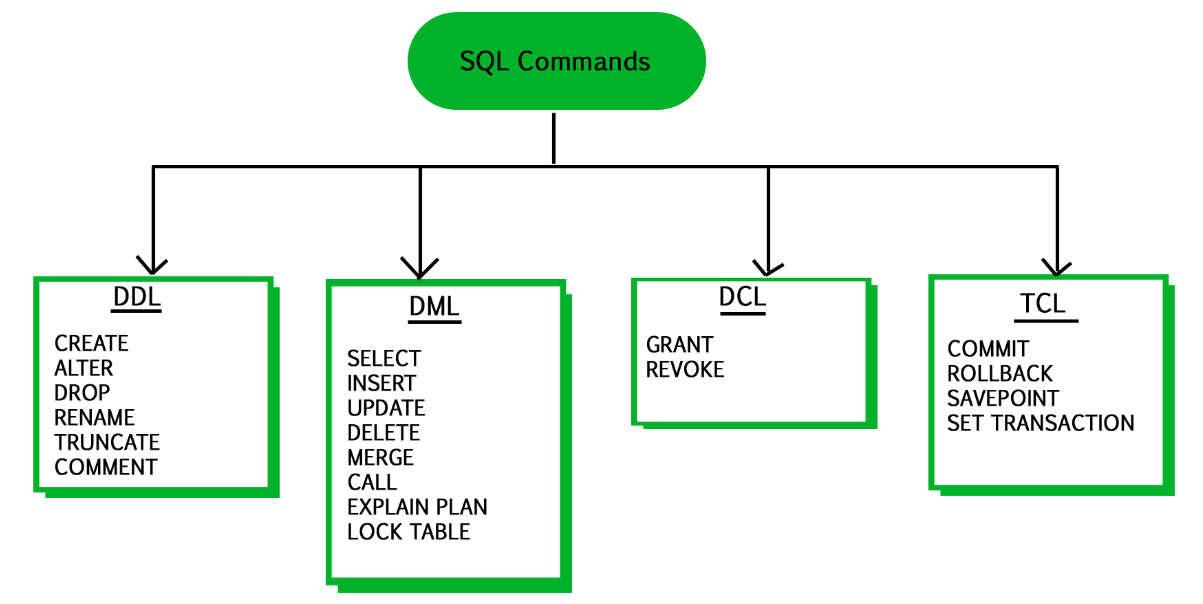
Que- What is SQL ?

SQL stands for Structured Query Language. is a standard language for storing, manipulating and retrieving data in databases.

* Allows users to communicate i.e., access and manipulate the database.
* Allows users to create, update, modify and delete the database



## COMMANDS OF SQL



### **DDL**: DDL stands for Data Definition Language, which deals with database schemas and descriptions of how the data should reside in the database.

● CREATE - to create a database and its objects like (table, index, views, store procedure, function, and triggers)

● ALTER - alters the structure of the existing database

● DROP - delete objects from the database

● TRUNCATE - remove all records from a table, including all spaces allocated for the records are removed

● RENAME - rename an object

● COMMENT – Used to add comments to the data dictionary

### **DML**: Data manipulation language makes the user able to retrieve and manipulate data in a relational database. The DML commands can only perform read-only operations on data.

● SELECT - retrieve data from a database

● INSERT - insert data into a table

● UPDATE - updates existing data within a table

● DELETE - Delete all records from a database table

● MERGE - UPSERT operation (insert or update)

### **DCL**: DCL is short name of Data Control Language which includes commands such as GRANT and mostly concerned with rights, permissions and other controls of the database system.

● GRANT - allow users access privileges to the database

● REVOKE - withdraw users access privileges given by using the GRANT command

### **TCL:** TCL is short name of Transaction Control Language which deals with a transaction within a database.

● COMMIT - commits a Transaction (Conform)

● ROLLBACK - rollback a transaction in case of any error occurs (Revert- Undo)

● SAVEPOINT - to roll back the transaction making points within groups (Checkpoint)

Que - What is Query ?

A query is a request for data or information from a database table or combination of tables.

A database query can be either a select query or an action query.

Que - What is SubQuery?

A subquery is a query within another query, also known as a nested query or inner query. It is used to restrict or enhance the data to be queried by the main query, thus restricting or enhancing the output of the main query respectively.

## (\*) SQL DATABASE

**CREATE DATABASE:** The CREATE DATABASE statement is used to create a new SQL database.

Syntax – ● CREATE DATABASE databasename;

**DROP DATABASE:** The DROP DATABASE statement is used to drop an existing SQL database.

Syntax – ● DROP DATABASE databasename;

**RENAME DATABASE:** the **Rename Database** statement in SQL is used to change the name of the existing database.

**ALTER** **DATABASE** old\_database\_name **MODIFY** **NAME** = new\_database\_name;

**SELECT DATABASE:** Any database user and administrator can easily select the particular database from the current database server using the **USE** statement in SQL.

1. SHOW DATABASES;
2. USE Hospital;

**SELECT:** The SELECT statement is used to select data from a database.

Syntax - ● SELECT column1, column2, ... FROM table\_name;

-Here, column1, column2, ... are the field names of the table you want to select data from. If you want to select all the fields available in the table, use the following

syntax: ● SELECT \* FROM table\_name;

**SELECT DISTINCT:** The SELECT DISTINCT statement is used to return only distinct (different) values. Syntax – SELECT DISTINCT *column1*, *column2, …* FROM *table\_name*;

## (\*)CLAUSE

**1. WHERE:** The WHERE clause is used to filter records.

Syntax – ● SELECT column1, column2, ... FROM table\_name WHERE condition;

The WHERE clause can be combined with AND, OR, and NOT operators.

The AND and OR operators are used to filter records based on more than one condition:

**2. AND: The SQL AND condition is used in SQL query to create two or more conditions to be met,** The AND operator displays a record if all the conditions separated by AND are TRUE.

Syntax- **SELECT** \***FROM** emp **WHERE** Department = "IT" AND Location = "Chennai";

**3.OR** The SQL OR condition is used in SQL query to create a SQL statement where records are returned when any one condition is met.

**Syntax- SELECT columns FROM tables WHERE condition 1 OR condition 2;**

**4.and NOT:** The NOT operator displays a record if the condition(s) is NOT TRUE.

**5. AS:** SQL '**AS'** is used to assign a new name temporarily to a table column or even a table.

**SELECT** Column\_Name1 **AS** New\_Column\_Name, Column\_Name2 **As** New\_Column\_Name **FROM** Table\_Name;

**6. Having:** The HAVING clause places the condition in the groups defined by the GROUP BY clause in the SELECT statement.

This SQL clause is implemented after the 'GROUP BY' clause in the 'SELECT' statement.

**SELECT** column\_Name1, column\_Name2, ....., column\_NameN aggregate\_function\_name(column\_Name) **FROM** table\_name

**GROUP** **BY** column\_Name1

**HAVING** condition;

**GROUP BY:** The GROUP BY statement groups rows that have the same values into summary rows, like "find the number of customers in each country". The GROUP BY statement is often used with aggregate functions (COUNT(), MAX(), MIN(), SUM(), AVG()) to group the result-set by one or more columns. Syntax – ● SELECT column\_name(s) FROM table\_name WHERE condition GROUP BY column\_name(s) ORDER BY column\_name(s);

**HAVING:** The HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions. \*WHERE is given priority over HAVING. Syntax – ● SELECT column\_name(s) FROM table\_name WHERE condition GROUP BY column\_name(s) HAVING condition ORDER BY column\_name(s);

**Operator Description** = Equal > Greater than < Less than >= Greater than or equal <= Less than or equal <> Not equal. Note: In some versions of SQL this operator may be written as !=

Syntax – ● SELECT column1, column2, ... FROM table\_name WHERE condition1 AND condition2 AND condition3 ...;

● SELECT column1, column2, ... FROM table\_name WHERE condition1 OR condition2 OR condition3 ...;

● SELECT column1, column2, ... FROM table\_name WHERE NOT condition;

**ORDER BY:** The ORDER BY keyword is used to sort the result-set in ascending or descending order. The ORDER BY keyword sorts the records in ascending order by default. To sort the records in descending order, use the DESC keyword.

Syntax – SELECT *column1*, *column2, ...*

FROM *table\_name*

ORDER BY *column1, column2, ...* ASC|DESC;

**SELECT column, another\_column, …**

**FROM mytable**

**WHERE *condition(s)***

**ORDER BY column ASC/DESC**

**LIMIT num\_limit OFFSET num\_offset;**

**INSERT INTO:** The INSERT INTO statement is used to insert new records in a table.

Syntax – ● INSERT INTO table\_name (column1, column2, column3, ...) VALUES (value1, value2, value3, ...);

● INSERT INTO table\_name VALUES (value1, value2, value3, ...); \*In the second syntax, make sure the order of the values is in the same order as the columns in the table.

**NULL Value:** It is not possible to test for NULL values with comparison operators, such as =, <, or <>. We will have to use the IS NULL and IS NOT NULL operators instead.

Syntax – ● SELECT column\_names FROM table\_name WHERE column\_name IS NULL;

● SELECT column\_names FROM table\_name WHERE column\_name IS NOT NULL;

**UPDATE:** The UPDATE statement is used to modify the existing records in a table.

Syntax – ● UPDATE table\_name SET column1 = value1, column2 = value2, ... WHERE condition;

**DELETE:** The DELETE statement is used to delete existing records in a table.

Syntax – ● DELETE FROM table\_name WHERE condition;

● DELETE FROM table\_name; In 2nd

syntax, all rows are deleted. The table structure, attributes, and indexes will be intact

**SELECT TOP:** The SELECT TOP clause is used to specify the number of records to return.

Syntax – ● SELECT TOP number|percent column\_name(s) FROM table\_name WHERE condition;

● SELECT column\_name(s) FROM table\_name WHERE condition LIMIT number;

● SELECT column\_name(s) FROM table\_name ORDER BY column\_name(s) FETCH FIRST number ROWS ONLY;

● SELECT column\_name(s) FROM table\_name WHERE ROWNUM <= number; \*In case the interviewer asks other than the TOP, rest are also correct. (Diff. DB Systems)

## (\*)Aggregate Functions:

**MIN():** The MIN() function returns the smallest value of the selected column.

Syntax – ● SELECT MIN(column\_name) FROM table\_name WHERE condition;

**MAX():** The MAX() function returns the largest value of the selected column.

Syntax – ● SELECT MAX(column\_name) FROM table\_name WHERE condition;

**COUNT():** The COUNT() function returns the number of rows that matches a specified criterion.

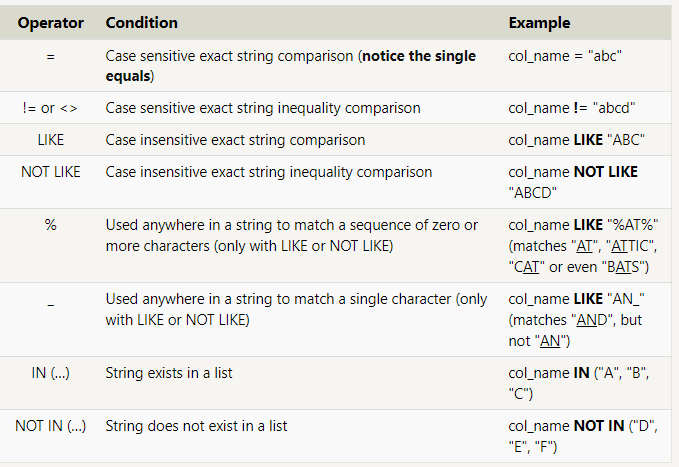
Syntax – ● SELECT COUNT(column\_name) FROM table\_name WHERE condition;

**AVG():** The AVG() function returns the average value of a numeric column.

Syntax – ● SELECT AVG(column\_name) FROM table\_name WHERE condition;

**SUM():** The SUM() function returns the total sum of a numeric column.

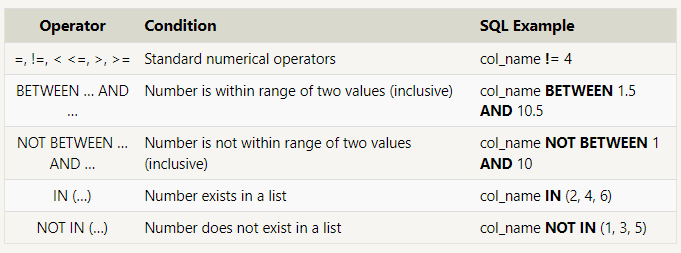
Syntax – ● SELECT SUM(column\_name) FROM table\_name WHERE condition;



**LIKE Operator:** The LIKE operator is used in a WHERE clause to search for a specified pattern in a column. There are two wildcards often used in conjunction with the LIKE operator: ● The percent sign (%) represents zero, one, or multiple characters ● The underscore sign (\_) represents one, single character Syntax – ● SELECT column1, column2, ... FROM table\_name WHERE columnN LIKE pattern;

**IN:** The IN operator allows you to specify multiple values in a WHERE clause. The IN operator is a shorthand for multiple OR conditions. Syntax – ● SELECT column\_name(s) FROM table\_name WHERE column\_name IN (value1, value2, ...); ● SELECT column\_name(s) FROM table\_name WHERE column\_name IN (SELECT STATEMENT); Ex – ● SELECT \* FROM Customers WHERE Country IN ('Germany', 'France', 'UK'); ● SELECT \* FROM Customers LIKE Operator Description WHERE CustomerName LIKE 'a%' Finds any values that start with "a" WHERE CustomerName LIKE '%a' Finds any values that end with "a" WHERE CustomerName LIKE '%or%' Finds any values that have "or" in any position WHERE CustomerName LIKE '\_r%' Finds any values that have "r" in the second position WHERE CustomerName LIKE 'a\_%' Finds any values that start with "a" and are at least 2 characters in length WHERE CustomerName LIKE 'a\_\_%' Finds any values that start with "a" and are at least 3 characters in length WHERE ContactName LIKE 'a%o' Finds any values that start with "a" and ends with "o" WHERE Country IN (SELECT Country FROM Suppliers);

**BETWEEN:** The BETWEEN operator selects values within a given range. The values can be numbers, text, or dates. The BETWEEN operator is inclusive: begin and end values are included. Syntax – ● SELECT column\_name(s) FROM table\_name WHERE column\_name BETWEEN value1 AND value2; Ex – ● SELECT \* FROM Products WHERE Price BETWEEN 10 AND 20;

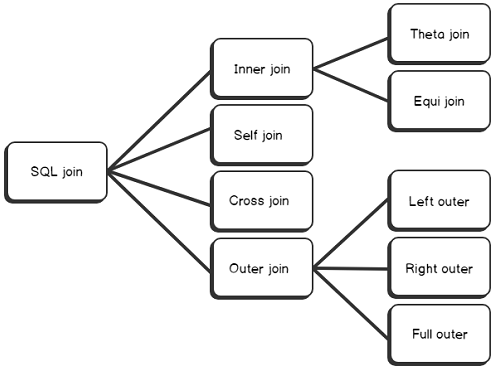
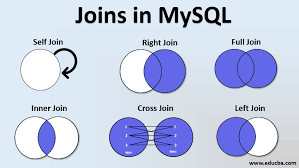


## [(\*) JOINS](https://www.sqlshack.com/sql-join-overview-and-tutorial/)

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

Join:

There are 3 types of join 1. Cross Join 2. Inner Join 3. Outer Join



1. cartesian product, cartesian join:

15\*4= 60

SELECT \*

FROM table1

CROSS JOIN table2;

-----------------------------------------------------

2. inner join

theta:

select ename,dname

from emp e, dept d

where e.deptno=d.deptno and ename like 'A%';

ansii:

select ename,dname

from emp e inner join dept d

on e.deptno=d.deptno

where ename like 'A%';

-----------------------------------------------------

3. outer join

a)left

select ename,dname

from emp e left outer join dept d

on e.deptno=d.deptno;

b)right

select ename,dname

from emp e right outer join dept d

on e.deptno=d.deptno;

c) full outer join

select ename,dname

from emp e left outer join dept d

on e.deptno=d.deptno

union

select ename,dname

from emp e right outer join dept d

on e.deptno=d.deptno;

----------------------------------------------------

4.self join

select w.ename, m.ename

from emp w, emp m

where w.mgr=m.empno and m.ename='BLAKE';

select w.ename, m.ename

from emp w inner join emp m

on w.mgr=m.empno

where m.ename='BLAKE';

**INNER JOIN:** The INNER JOIN keyword selects records that have matching values in both tables.

Syntax – ● SELECT column\_name(s) FROM table1 INNER JOIN table2 ON table1.column\_name = table2.column\_name;

**LEFT (OUTER) JOIN:** The LEFT JOIN keyword returns all records from the left table (table1), and the matching records from the right table (table2). The result is 0 records from the right side, if there is no match. Syntax – ● SELECT column\_name(s) FROM table1 LEFT JOIN table2 ON table1.column\_name = table2.column\_name;

**RIGHT (OUTER) JOIN:** The RIGHT JOIN keyword returns all records from the right table (table2), and the matching records from the left table (table1). The result is 0 records from the left side, if there is no match. Syntax – ● SELECT column\_name(s) FROM table1 RIGHT JOIN table2 ON table1.column\_name = table2.column\_name;

**FULL (OUTER) JOIN:** The FULL OUTER JOIN keyword returns all records when there is a match in left (table1) or right (table2) table records.

Syntax: ● SELECT column\_name(s) FROM table1 FULL OUTER JOIN table2 ON table1.column\_name = table2.column\_name WHERE condition;

**UNION:** The UNION operator is used to combine the result-set of two or more SELECT statements.

● Every SELECT statement within UNION must have the same number of columns

● The columns must also have similar data types

● The columns in every SELECT statement must also be in the same order The UNION operator selects only distinct values by default. To allow duplicate values, use UNION ALL

Syntax – ● SELECT column\_name(s) FROM table1 UNION SELECT column\_name(s) FROM table2;

● SELECT column\_name(s) FROM table1 UNION ALL SELECT column\_name(s) FROM table2;

**(\*) SQL TABLE**

**CREATE TABLE:** The CREATE TABLE statement is used to create a new table in a database.

Syntax – ● CREATE TABLE table\_name ( column1 datatype, column2 datatype, column3 datatype, .... );

**DROP TABLE:** The DROP TABLE statement is used to drop an existing table in a database.

Syntax – ● DROP TABLE table\_name;

**TRUNCATE TABLE:** The TRUNCATE TABLE statement is used to delete the data inside a table, but not the table itself.

Syntax – ● TRUNCATE TABLE table\_name;

**ALTER TABLE**: The ALTER TABLE statement is used to add, delete, or modify columns in an existing table. The ALTER TABLE statement is also used to add and drop various constraints on an existing table. Syntax – ● ALTER TABLE table\_name ADD column\_name datatype; ● ALTER TABLE table\_name DROP COLUMN column\_name; ● ALTER TABLE table\_name MODIFY COLUMN column\_name datatype; Ex – ● ALTER TABLE Customers ADD Email varchar(255); ● ALTER TABLE Customers DROP COLUMN Email; ● ALTER TABLE Persons ALTER COLUMN DateOfBirth year

Que- Diff between TRUNCATE and DELETE command ..

DELETE=> Basically, it is a [Data Manipulation Language Command (DML)](https://www.geeksforgeeks.org/sql-ddl-dql-dml-dcl-tcl-commands/). It is used to delete one or more tuples of a table. With the help of the “DELETE” command, we can either delete all the rows in one go or can delete rows one by one. i.e., we can use it as per the requirement or the condition using the Where clause. It is comparatively slower than the TRUNCATE command. The TRUNCATE command does not remove the structure of the table.

DROP=> It is a Data Definition Language Command (DDL). It is used to drop the whole table. With the help of the “DROP” command we can drop (delete) the whole structure in one go i.e. it removes the named elements of the schema. By using this command the existence of the whole table is finished or say lost.

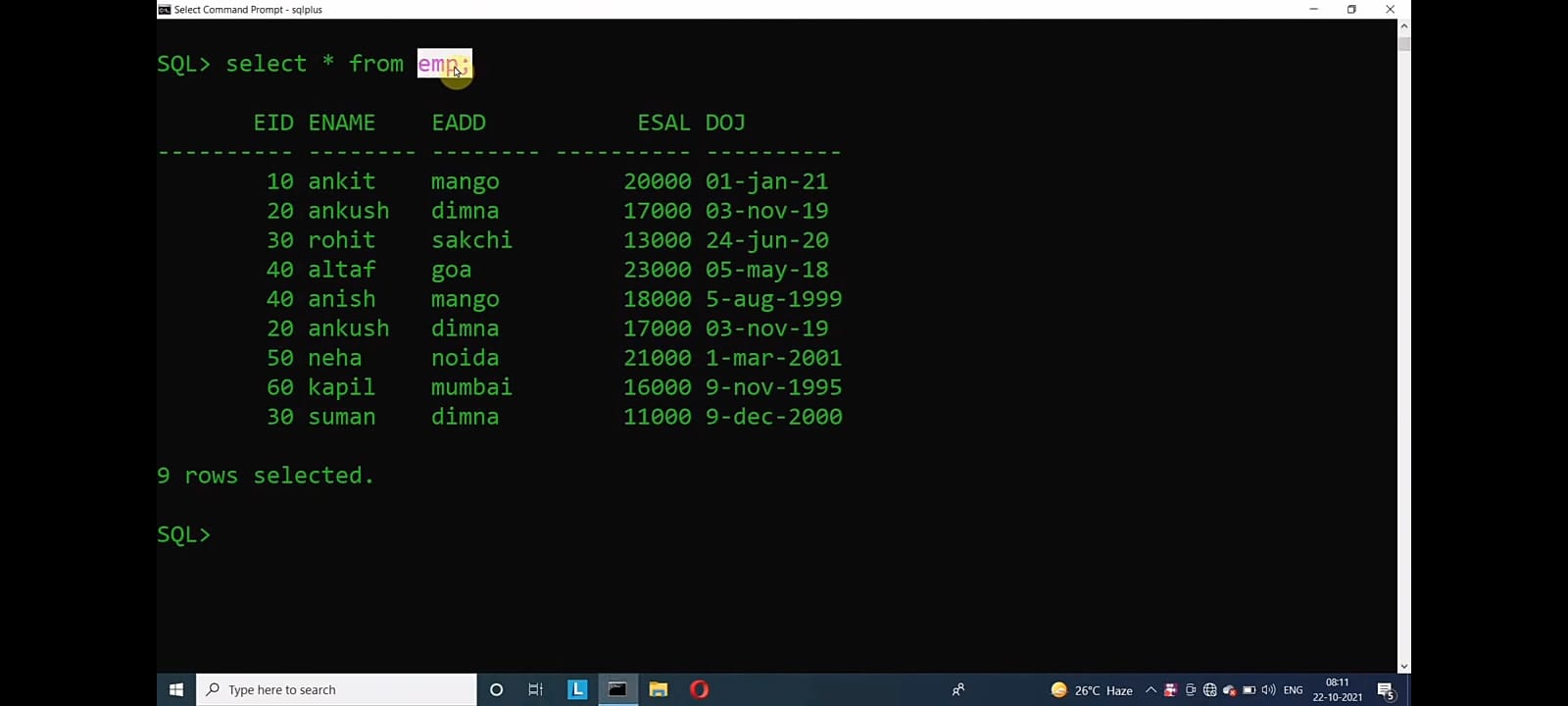
TRUNCATE=> It is also a Data Definition Language Command (DDL). It is used to delete all the rows of a relation (table) in one go. With the help of the “TRUNCATE” command, we can’t delete the single row as here WHERE clause is not used. By using this command the existence of all the rows of the table is lost. It is comparatively faster than the delete command as it deletes all the rows fastly.

### 

## SQL Queries PRACTICE // Interview Practice

<https://youtu.be/niFUzvtGyLs>

Table 1



**Q1-Find maximum salary of employee?**

=> select \* from emp where esal=(select max(esal) from emp);

**Q2-Find the second maximum salary of an employee?**

=> select \* from emp where esal=(select max(esal) from emp

where esal<(select max(esal) from emp));

**Q3-Retrieve all the information of employee whose name begin with 'A'**

=> select \* from emp where ename like ‘a%’;

**Q4-Select all the information of an employee whose salary is 21000,11000 and 13000?**

=> select \* from emp where esal in(23000, 11000, 13000);

**Q5-Create a new table with the help of existing table data and structure?**

=>create table emp2 as select \* from emp;

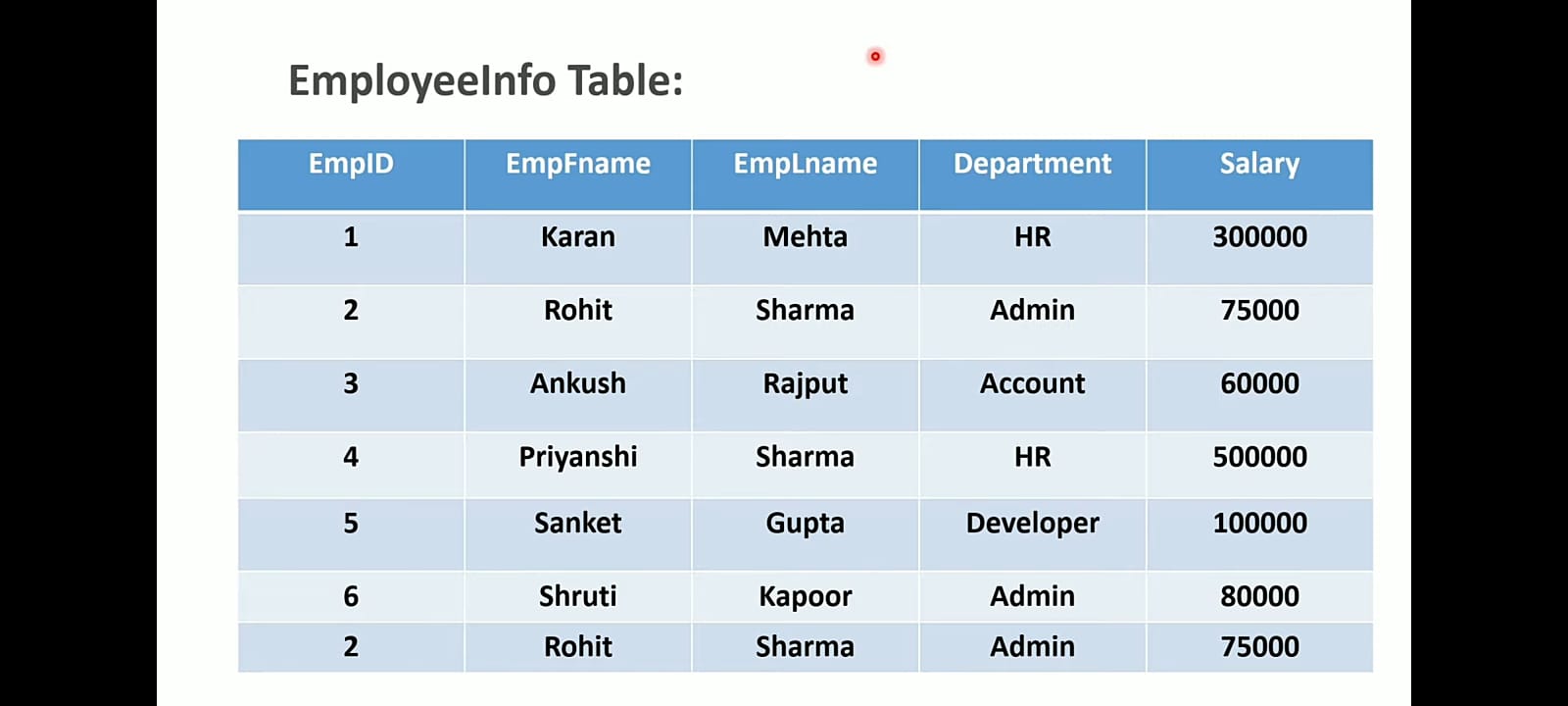
**Q6-Find all employee records whose salary is between 10000 to 15000?**

=>select \* from emp where esal between 10000 and 15000;

**Q7-Write a SQL query to delete more than one record?**

=>delete from emp where eid in(10,20,60);

<https://youtu.be/UlXtfq-kuF4>



**Q-Write a query to find the third highest salary from the EmployeeInfotable.**

=> SELECT salary FROM EmployeeInfo

ORDER BY salary DESC

LIMIT 2,1;

LIMIT 2,1 -> select one row after the second row ie 3rd highest.

**Q-Write a query to find the Nth highest salary from the table without using top and limit keyword**

SELECT salary FROM EmployeeInfo e1

WHERE N-1 = (SELECT COUNT (DISTINCT salary) FROM EmployeeInfo e2 WHERE e2.salary > e1.salary);

**Q-Write a query to calculate the even and odd records from a table**

For even

SELECT \* FROM EmployeeInfo

WHERE MOD(EmpID,2)=0;

For odd

SELECT \* FROM EmployeeInfo

WHERE MOD(EmpID,2)=1;

**Q-Write SQL query to find duplicate rows in a table**

SELECT \*,COUNT(Empid) FROM EmployeeInfo

GROUP BY Empid

HAVING COUNT(Empid)>1;

**Q-Write a query to display the first and the last record from the employee infotable**

For first record

SELECT \* FROM EmployeeInfo

WHERE EmpID= (SELECT MIN(EmpID) FROM EmployeeInfo);

For last record

SELECT \* FROM EmployeeInfo

WHERE EmpID = (SELECT MAX(EmpID)FROM EmployeeInfo);

**Q-How Do you copy all rows of a table using SQL query**

CREATE TABLE EmpDetails AS SELECT \* FROM EmployeeInfo;

CREATE TABLE EmpSalary AS SELECT EmpID, salary FROM EmployeeInfo;

CREATE TABLE EmpDetails AS SELECT\* FROM EmployeeInfo

WHERE 3=4;

**Q-Write a query to retrieve the list of employees working in the same department**

=>SELECT DISTINCT E.EmpId, E.EmpFname, E.Department FROM

EmployeeInfo E, EmployeeInfo E1

WHERE E.Department = E1.Department

AND E.EmpID != E1.EmpID;

**Q-Write a query to retrieve the last three records from the employeeinfotable**

=>SELECT \* FROM (SELECT \* FROM EmployeeInfo

ORDER BY EmpID DESC LIMIT 3)

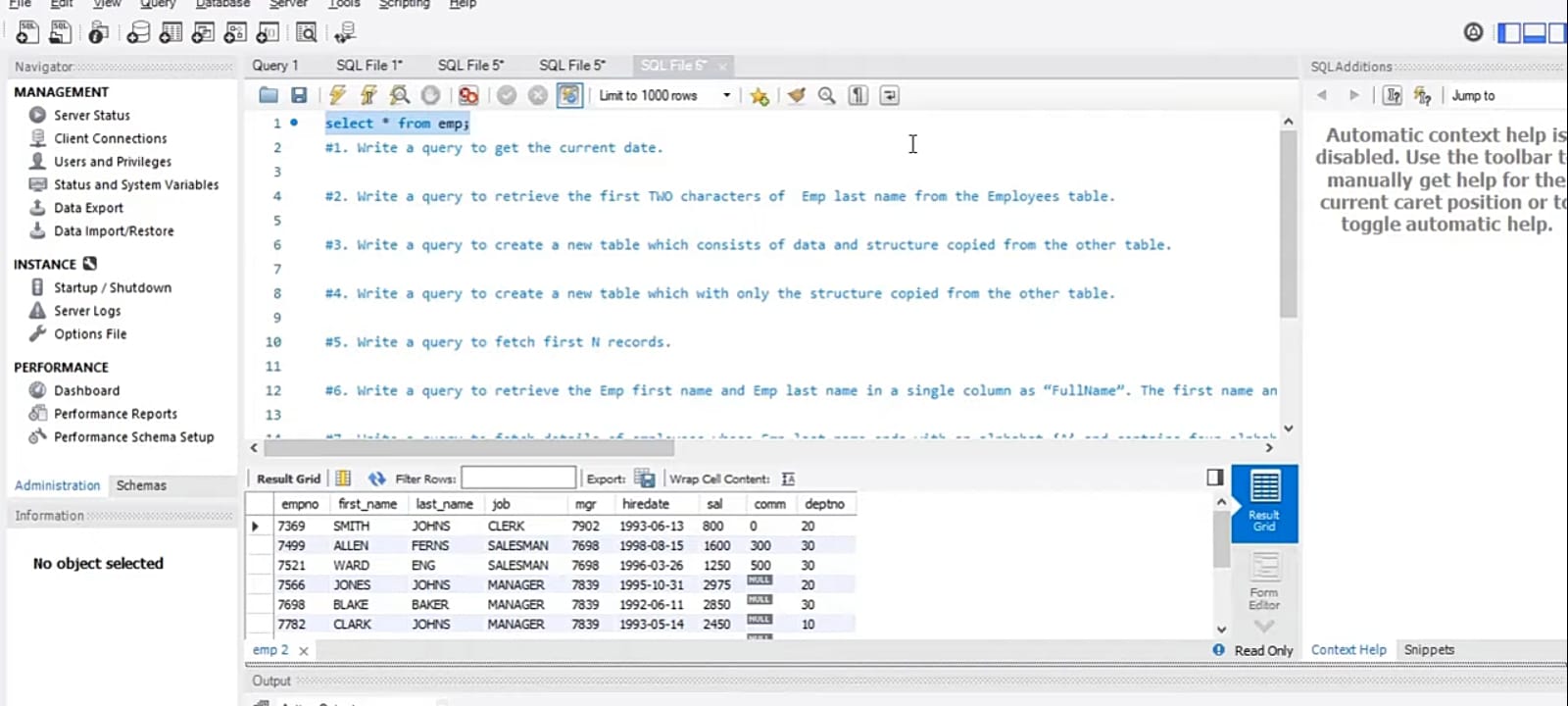
temp ORDER BY EmpID ASC;

SELECT \* FROM temp ORDER BY EmpID ASC;

**Q-Write a query to fetch details of employees whose EmpLname ends with an alphabet 'A' and contains 5 alphabet.**

=> SELECT \* FROM EmployeeInfo WHERE EmpLname LIKE '\_\_\_A';

<https://youtu.be/zfN6CguL8Xo>



**Q-1 Write a query to get the current date.**

SELECT sysdate(); // SELECT GETDATE();

**Q-2 Write a query to retrieve the first TWO characters of Emp last name from the Employees table.**

SELECT substr(last\_name, 1,2), last\_name from emp

**Q-3 Write a query to create a new table which consists of data and structure copied from the other table.**

CREATE table temp AS select \* from emp

SELECT \* From temp

**Q-4. Write a query to create a new table which with only the structure copied from the other table.**

CREATE table temp2 AS SELECT \* FROM emp WHERE 1=0

SELECT \* From temp2

**Q-5. Write a query to fetch first N records.**

SELECT \* FROM emp LIMIT 5

**Q-6. Write a query to retrieve the Emp first name and Exp last name in a single column as "Fullname". The first name and last name is separated with space.**

SELECT CONCAT(First\_Name,’ ‘,Last\_Name) As ‘FullName’ FROM emp

**Q-7 Write a query to fetch details of employees whose Emp last name ends with an alphabet 'S' and contains 5 alphabet**

SELECT \* FROM Emp WHERE Last\_Name LIKE ‘\_\_\_\_S’;

**Q8. Write a query to fetch details of all employees excluding the employees with first names, "SMITH" and "BLAKE" fr**

Select \* FROM Emp WHERE First\_name IN(‘SMITH’, ‘BLAKE‘)

**Q9. Write a query to fetch the department-wise count of employees sorted by department's count in ascending order.**

SELECT COUNT(EmpNO) C, JOB FROM Emp

Group By JOB

Order By C DECS;

**Q10. Write a query to retrieve duplicate records from a table.**

SELECT \*,COUNT(EmpNO) FROM emp

Group By EmpNO

Having COUNT(EmpNO)>1;

<https://youtu.be/sB_39iTDCjo>

<https://youtu.be/eQFhgF_fjFs>

**Q- Find the number of employees working in each department and the number of employee should be more than 5**

SELECT COUNT(\*) as count,d.dept\_name

FROM employee e

INNER JOIN

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

GROUP BY e.dept\_id

HAVING count>5;

**Q-Find the number or count of employees working in each department**

SELECT COUNT(\*) as count,d.dept\_name

FROM employee e

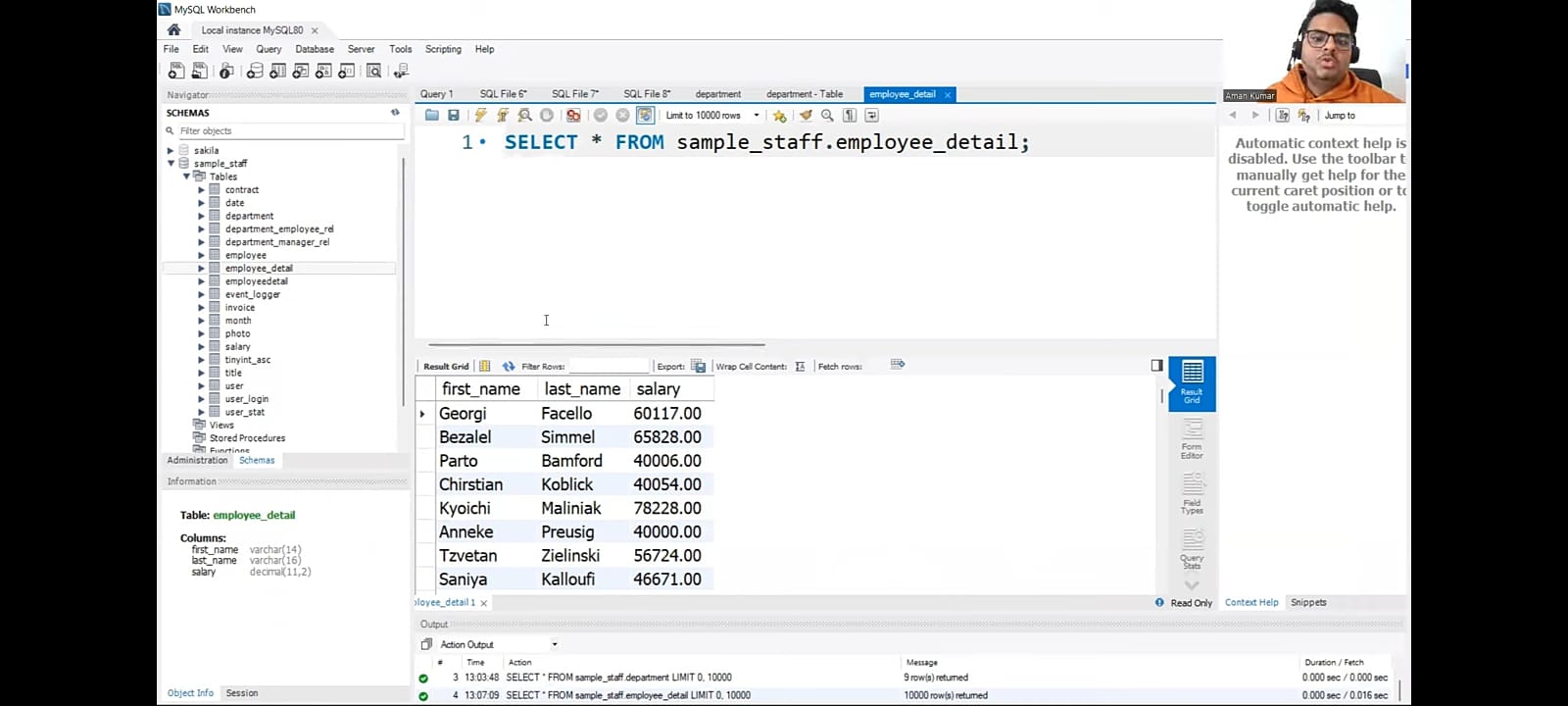
INNER JOIN

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

GROUP BY e.dept\_id;

<https://youtu.be/Iv9qBz-cyVA>

<https://youtu.be/dewfkPNDwK4>

****

**Q- find duplicate rows in a table**

SELECT first\_name, last\_name, salary, count(\*)

FROM sample\_staff.employee\_detail

group by first\_name, last\_name, salary

Having Count(\*)>1;

**Q- find first and last record from a table**

select \* from sample\_staff.department

where id= (select min(id) from sample\_staff.department)

select \* from sample\_staff.department

where id= (select max(id) from sample\_staff.department)

**Q- find last 3 records from a table**

select \* from sample\_staff.department order by id desc limit 3

**Q- find the 2nd highest salary from employee table.**

SELECT first\_name, last\_name, salary

FROM sample\_staff. employee\_detail

order by salary desc

limit 1,1;

**Q- select \* from sample\_staff. employee\_detail**

order by salary desc;

<https://youtu.be/_trMsDNGFR8>

**Q1- Write a query to fetch the number of employees working in the department 'HR'**

SELECT COUNT(\*) FROM Employee WHERE Department='HR'

**Q2- Write a query to get the current date.**

SELECT GETDATE();

**Q3- Write a query to retrieve the first 4 characters of last name from the Employee table.**

SELECT SUBSTRING(LastName,1,4)FROM Employee;

**Q4- Write a Query to create a new table which consists of data and structure copied from the other table.**

SELECT \* INTO NewTable FROM Employee WHERE 1=1;

**Q5- Write a query to find all the employees who salary is between 500 to 1000**

SELECT \* FROM Employee WHERE Salary BETWEEN 500 AND 1000

**Q6- WRITE A QUERY TO FIND THE NAMES OF EMPLOYEES THAT BEGIN WITH 'S'.**

SELECT \* FROM Employee WHERE Name LIKE 'S%';

**Q7- Write a query to fetch top five records having the highest salary.**

SELECT TOP 5 \* FROM Employee ORDER BY Salary DESC;

**Q8- Write a query to retrieve departments who have less than two employees working in it**

SELECT Department, COUNT(EmpID) as 'EmpNo' FROM Employee

GROUP BY Department HAVING COUNT(EmpID)<2;

**Q9- Write a Query to receive the first name and last name in a single column as full name the first name and the last name must be separated by space**

SELECT CONCAT(FirstName, ' ' , LastName) As 'FullName' FROM Employee

**Q10- Write a query to fetch the first name from the employee table in uppercase.**

SELECT UPPER(FirstName) AS EmpName FROM Employee

**Q11- Write a Query to find the number of employees whose dob is between 01/01/1990 to 31/12/2021 and are grouped According to Department.**

SELECT Department, COUNT(\*) FROM Employee WHERE dob BETWEEN '01/01/1990' AND '31/12/2021' GROUP BY Department;

**Q12- Write Query to fetch all the records from the employee table ordered by the first name in descending order and department in the ascending order.**

SELECT \* FROM Employee ORDER BY FirstName desc, Department asc;

**Q13- Write a query to fetch details of employees whose first name ends with an alphabet A and contain 5 alphabets**

SELECT \* FROM Employee WHERE LastName LIKE '\_ \_ \_ \_a';

**Q14- Write a query to fetch details of all employees excluding the employees with first name, "IQBees" and "Test" from the Employee table**

SELECT \* FROM Employee WHERE FirstName NOT IN ('IQBees', 'Test');

**Q15- Write a query to fetch details of employees having 'ABC' in their first name.**

SELECT \* FROM Employee WHERE FirstName LIKE '%ABC%';

**Q16- WRITE A QUERY TO RETRIEVE DUPLICATE RECORDS FROM A TABLE**

SELECT EmpID, Name, Department, COUNT(\*) FROM Employee GROUP BY EmpID, Name, Department HAVING COUNT(\*)> 1;

**Q17- WRITE A QUERY TO FIND THE THIRD HIGHEST SALARY FROM THE Employee TABLE.**

SELECT TOP 1 Salary FROM(SELECT TOP 3 Salary FROM Employee ORDER BY Salary DESC) AS emp ORDER BY Salary ASC;

**Q18- Write a query to fetch all the employees having valid email**

SELECT EmailAddress AS ValidEmail, \* FROM Employee WHERE EmailAddress LIKE '%@\_%' AND PATINDEX('%[^a-z, 0-9, @,.,\-]%, EmailAddress)=0

<https://youtu.be/d-SJmsgoUrw>

**Q-1. SQL query to find Nth highest salary**

**Q-2. SQL query to delete duplicate rows from a table**

**Q-3. Join in SQL**

**Q-4. constraints in SQL**

**Q-5. aggregate function in SQL**

**Q-6. Keys in DBMS**

**Q-7. what is normalization and why is it needed**

**Q-8. different type of SQL commands**

**Q-9. difference between delete and truncate and drop**

**Q-10. difference between where and having**

**Q-11. difference between group by and order by**

**Q-12. difference between union and join**

**Q-13. trains between in and exits**

**Q-14. difference between char and varchar**

**Q-15. how to create empty table with the same structure as another table**

SELECT \* INTO students\_copy

FROM student WHERE 1=2;

**Q-16. what is pattern matching in SQL**

% wildcard + LIkE operator

**SELECT \* FROM student**

WHERE name LIKE '\_k%';

**Q-17. What does union minus intersect?**

SELECT name FROM student

UNION

SELECT name FROM

contacts;

Q-18. **difference between union and union all?**

Q-19. **What is the view ?**

Q-20. **character manipulation function in SQL**

<https://youtu.be/HQUUBOS4S9Q>



**Q.1- Write a query to delete the Duplicate record from employee\_info table.**

DELETE E1

FROM employee\_info E1, employee\_info E2

WHERE E1.email = E2.email

AND E1.Id> E2.Id;

**SUBSTRING() function extracts a substring from a string (starting at any position).**

EXAMPLE-

Name = Codeera

SUBSTRING(name, 1,4)

Output-Code

SUBSTRING(name, 5,3) Output-era

# Extract a substring from the text (start at position 1, extract 4 characters)

# The first letter starts from position 1 not 0

**Q-2. Query to retrieve the first four characters of Employee name from the employee\_info table.**

**If we pass date in YYYY-MM-DD format**

SELECT COUNT(\*), Gender

FROM employee\_info

WHERE DOB

BETWEEN '1995-01-01' AND '1998-06-30'

GROUP BY Gender;

SELECT COUNT(\*), Gender FROM employee\_info

WHERE DOB

BETWEEN

STR\_TO\_DATE('01/01/1995', '%d/%m/%Y')

AND

STR\_TO\_DATE('30/06/1998', '%d/%m/%Y')

GROUP BY Gender;

**Q-3. Query find number of employees whose DOB is between 01/06/1995 to 30/06/1998 and are grouped according to gender**

**STR\_TO\_DATE() function returns a date based on a string and a format.**

EXAMPLE-

STR\_TO\_DATE ('01/01/1995', '%d/%m/%Y') Output-1995-01-01

STR\_TO\_DATE ('August 10 2017', '%M %d %Y') Output-2017-08-10

**Q-4. Query to fetch all the records from the employee\_info table ordered by Department in ascending order and Salary in the descending order.**

SELECT \*

FROM employee\_info

ORDER BY department ASC, salary DESC;

**Q-5. Query to fetch details of all employees excluding the employees who are "HR" and "Admin" ✓ Query to fetch 50%** records from the employee\_info table.

SELECT \* FROM employee\_info WHERE department NOT IN ('HR','Admin');

**Q- Write a query to fetch 50% records from the employee\_info table.**

# Works fine when ID is in proper sequence i.e. no record deleted

SELECT \*

FROM employee\_info WHERE Id <= (SELECT COUNT(Id)/2

FROM employee\_info );

# When some records are deleted.

SET @count = (SELECT COUNT(Id)/2 FROM employee\_info); PREPARE STMT FROM 'SELECT \* FROM employee\_info LIMIT?'; EXECUTE STMT USING @count;

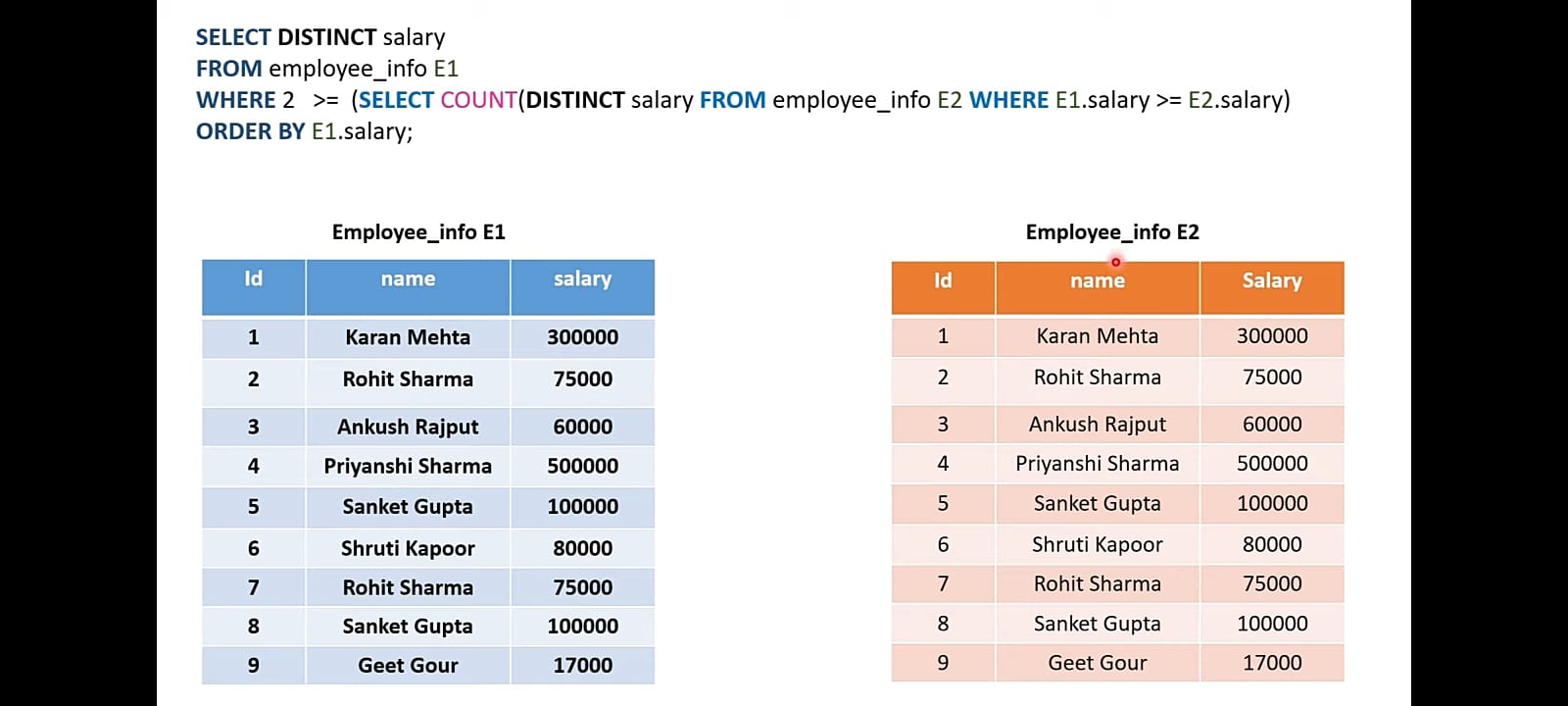
'SELECT \* FROM employee\_info LIMIT 4.5';

'SELECT \* FROM employee\_info LIMIT 5';

**Q-6. Query to display the total salary of each employee after adding 10% increment in the Salary.**

SELECT Id,name, salary+(salary/10) as TotalSalary FROM employee\_info;

**Q7. Query to retrieve two minimum and maximum salaries**



**For minimum Salary-**

SELECT DISTINCT salary

FROM employee\_info E1

WHERE 2 >= (SELECT COUNT(DISTINCT salary) FROM employee\_info E2 WHERE E1.salary >= E2.salary)

ORDER BY E1.salary;

**For maximum Salary-**

SELECT DISTINCT salary

FROM employee\_info E1

WHERE 2 >= (SELECT COUNT(DISTINCT salary)

FROM employee\_info E2

WHERE E1.salary <= E2.salary)

ORDER BY E1.salary DESC;

**Q8. Query to fetch the employee name and replace the space with '-'**

SELECT REPLACE (name,’ ’,’-’) FROM employee\_info;

### 

### 

### 

### 

### \* **Cheat sheet**

