SQL

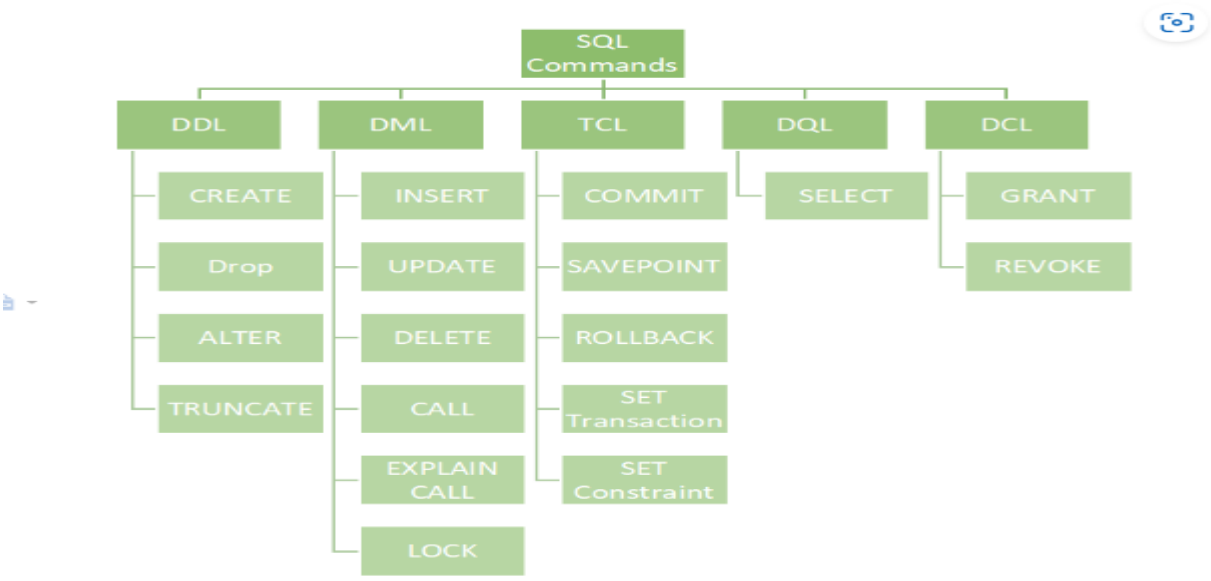
SQL is a language to store the data in organize way , Used to **store, retrieve, manage, and manipulate data** in relational databases.

DBMS it is a software system allow user to create, define, manipulate and manage the data

RDBMS is a program used to maintain relational database system

Database & Table Basics:

- Creating a Database (CREATE DATABASE)
- Deleting a Database (DROP DATABASE)
- Creating Tables (CREATE TABLE)
- Dropping Tables (DROP TABLE)
- Altering Tables (ALTER TABLE)
- Data Types in SQL (INT, VARCHAR, TEXT, DATE, etc.)



A c**omposite primary key** is a primary key made up of two or more columns. Together, these columns must form a unique combination for each row in the table. It's used when a single column isn't sufficient to uniquely identify a record.

Data Types in SQL

- **Numeric**: INT, FLOAT, DECIMAL
- **String**: char ,VARCHAR, Nchar , Nvarchar TEXT

  char: fixed length string
  varchar: variable length string

  nchar: fixed length Unicode string
  nvarchar : variable length Unicode string

- **Date/Time**: DATE, DATETIME, TIMESTAMP
- **Boolean**: BOOLEAN

SQL Operators

- Arithmetic Operators (+, -, *, /, %)
- Comparison Operators (=, !=, <, >, <=, >=)
- Logical Operators (AND, OR, NOT)
- STRING (LIKE, %, _)
- BETWEEN, IN, and NOT IN Operators

Constraints (is used to specify rule for the data in a table)

1. not null : it ensures that column to not accept null values
2. unique key: it ensure all the value in the column are unique
3. primary key : it ensure uniqueness of data and not null vales
4. foreign key : it establish relationship between the data
5. default :it provide default value when no values explicitly provided

6. check :it ensure all values satisfy a specified condition

NOTE: TOP : is used to specify the number of records to return.

Aggregate Functions : Aggregate function can be used to perform calculation on a set of values, which will then return a single value. We can use aggregate function either with GROUP BY clause or without it.

1. SUM
2. MIN
3. MAX
4. AVG
5. COUNT

Note : aggregate function ignores nulls

Distinct takes all null as one/same.

aggregate func ignores null.

Count(*) will not ignores null.

WHERE Clause

- Used to filter **rows before aggregation**.
- Works with **SELECT, UPDATE, DELETE**.
- Can't be used with **aggregate functions** (SUM(), AVG(), COUNT()).
- Faster since it filters data early.

HAVING Clause

- Used to filter **groups after aggregation**.
- Works only with **GROUP BY**.
- Can be used with **aggregate functions**.
- Slower because it filters after grouping.

what is the difference between DISTINCT and GROUP BY

DISTINCT :removes duplicate values from the result set.
Applies To: Columns

Groups rows with the same values and allows applying aggregate functions.
Applies To: Entire Rows

What are the rules to follow when using UNION operator?

- Both queries must return same same no of columns.
- The columns in both the queries must be in same order.
- Data type of all the columns in both the queries must be same.

How can you convert a text into date format? Consider the given text as "31-01-2021".

Microsoft SQL Server (MSSQL):

```
SELECT CAST('31-01-2021' as DATE) as date_value;
```

What is the difference between primary key, unique key and foreign key

primary key:

- Cannot contain **NULL values**.
- Each table can have **only one Primary Key**.

unique key:

- Can contain **NULL values**.
- A table can have **multiple Unique Keys**.

Trigger is a database object which is similar to a stored procedure which will automatically get invoked or executed when the specified event occurs in the database.

Materialized Views

✅ **Definition:** A **precomputed table** that **stores query results** physically.

Feature	View	Materialized View
Definition	A virtual table that executes a query whenever accessed.	A physical table that stores precomputed query results.
Storage	Does **not** store data; only the query definition is saved.	Stores the actual data in the database, consuming storage.

What is the difference between a function and a procedure?

Order of execution :-

- from /join
- where
- group by/aggregation
- having
- select
- distinct
- order by
- Top/limit/offset

Top: It fetches a **specific number of rows** from the result.always used with order by
OFFSET: is used to **skip rows** before returning results.

Delete/drop/truncate

1. delete removes rows from a table with a certain condition
2. truncate remove all rows without deleting the schema
3. drop will remove entire table from schema

1. delete -row label operation
2. truncate - table label operation.
3. drop - structure label operation.

performance -

1. delete - slower, specially for large dataset.
2. truncate is faster for large dataset
3. drop - fast, since it's deleting the table entirely

delete can be used with foreign key.
truncate can't be used if foreign key exist.

drop
when you need to delete specific row, need to maintain foreign key constraints, or need to trigger related actions via triggers.
- when you want remove the entire table(including it's structure from data base).

truncate

when you want to quickly delete from table (cleaning old data) and don't need to worry abot foreign key and trigger.

SQL Functions: IS A SPECIFIC PREDEFINED APPLICATON

- PRE DEFINED
- User DEFINE.

1. Character functions
2. Numeric Functions
3. date time functions

1.Character functions:

1. upper() syntax: select upper('raghu') as name
2. lower() syntax: select lower('RAGHU') as name
3. concat() add two or more string together
   Syntax : select concat('dhibrahm',' ','analytics')

4. charindex() searches for a substring in a string and return the position ; if the substring is not found the function will return 0 Charindex(SUBSTRING,STRING)
   Syntax :SELECT SUBSTRING('RAGHU.GOWDA@GMAIL',1,CHARINDEX('.','RAGHU.GOWDA@GMAIL')-1)AS FIRST_NAME

5. patindex() return the position of a pattern in a string if pattern is not found the function will return 0
6. substring() extracts some character in string ('string',start, length)
   Syntax : SELECT SUBSTRING('RAGHU.GOWDA@GMAIL',1,5)
   Output : Raghu

7. replace() IT Replicase's all occurrences of substring with in string with a new string Note :it is case sensitive
8. trim() removes a space character or other specified character from the start or end of the string

2.Numeric Functions:

1. Addition()
2. subtraction()
3. multiplication()
4. division()
5. Abs () it returns absolute value of number
6. ceiling it returns the smallest integer greater than or =to specified number
7. FLOOR returns the largest integer less than or =to specified number
8. round it rounds the number to the specified number of decimal placese
9. sqrt() returns the square root of a number
10. power()returns the value of a number raised to the power of another number.
11. EXP IT RETUENS EXPONATIAL OF NUMBER
12. LOG it returns natural algorithm with number
13. RAND()
14. Generates a random number between 0 (inclusive) and 1 (exclusive).

3.date time functions:

1. getdate() IT IS USED GET CURRENT DATE
2. Year()
3. Month()
4. Day()
5. DATEPART() returns specified part of a date
6. DATENAME
7. datediff* = DATEDIFF function in SQL is used to calculate the difference between two dates or times, is a sql server function that returns the difference btw two date based on a specified datepart like month ,date,years. Syntax: select datediff(datepart,start date,end date())
8. convert() is sql server fn used to change one data type to another first argument specifies the target data type that you want to convert to , in this case target datatype is

date , which represent date without time part syntax select convert(date,getdate()) when getdate is converted to date the portion is dicarded,and you are left with just date

9. dateadd() it is used to add a specified time interval to date

CASE WHEN : expression goes through conditions and returns a value when the first condition is met (like an if-then-else statement). So, once a condition is true, it will stop reading and return the result. If no conditions are true, it returns the value in the ELSE clause.

NESTED CASE WHEN = CASE INSIDE THE CASE

Joins in SQL

1. INNER JOIN: Returns records that have matching values in both tables.

NOTE: INEER JOINS IGNORE NULL VALUES

2. LEFT JOIN (or LEFT OUTER JOIN): Returns all records from the left table and matching records from the right table.
3. RIGHT JOIN (or RIGHT OUTER JOIN): Returns all records from the right table and matching records from the left table.
4. FULL JOIN (or FULL OUTER JOIN): Returns all records when there is a match in either the left or right table.
5. CROSS JOIN: Returns the Cartesian product of the two tables (every row in the first table joined with every row in the second table).

UNION/ UNION ALL

- Union used to append tables with unique records. Union All Appends the data with duplicate records.
- Union gives result in sorted order but union all not.
- While using union all we have to use order by in the second statement(at last)
- Union all is faster than union.

Rules to union :-

The data structure of both the tables should be same. Order of the column and data type should be same for respective columns. Number of columns should be same.

- In set all nulls are same.
- In "union all" all the nulls will be returned, while in union, intersect,except only one null will come as it removes duplicate.

Precedence - Intersect > union/except

Subqueries (Single-row, Multi-row, Correlated)

- A sub query can be defined as a query embedded within another query.

NOTE: We can't use "order by" inside the subquery without "TOP" or "OFFSET".

scalar/single valued sub queries - sub query returning only one row and one column.

Multivalued sub queries - sub query returning more than one row and columns or sub query returning one column and multiple row.

Co-Related subqueries. The inner query will execute the no of times that the outer query has the no of rows.

- The processing of corelated subquery depends on the value that are returned by the outer query.
- since this outer query in this particular case will be returning n records and the inner query will executed n times for every onetime

What is the difference between IN and EXISTS in SQL?

Answer: i IN checks if a value is within a set of values.

EXISTS checks if a subquery returns any rows. It is typically used for existence checking in correlated subqueries.

Indexing in SQL

An index is a database object used to improve the speed of data retrieval operations on a table.

clustered index determines the physical order of rows in a table.

- There can only be one clustered index per table because the rows themselves are stored in this order.

A non-clustered index is a separate structure from the actual data. It contains pointers (row locators) to the data stored in the table.

- A table can have multiple non-clustered indexes.

Optimisation technique:

1. Index frequently used columns for filtering and sorting.
2. Apply filtering (WHERE) as early as possible.
3. Use GROUP BY instead of DISTINCT when possible.
4. Prefer UNION ALL over UNION when duplicates are acceptable.
5. Use JOIN for large datasets and subqueries for small tables (<10%).
6. Avoid unnecessary ORDER BY for better performance.
7. Use CTEs (WITH queries) for code reuse and readability.
8. Avoid SELECT *; fetch only needed columns.
9. Prevent implicit conversions in queries.
10. Properly define table structure with optimized data types.

window functions:

Window functions perform calculations across a set of table rows that are related to the current row without collapsing result set

Types:

1. Aggregate Window Functions (SUM(), AVG(), MIN(), MAX(), COUNT())
2. Ranking Window Functions (RANK(), DENSE_RANK(), ROW_NUMBER())
3. Value-based Window Functions (LAG(), LEAD(), FIRST_VALUE(), LAST_VALUE())

Aggregate Window Functions

These functions compute aggregate values **without grouping the result set**.

a) SUM() - Calculates cumulative sum

WITH Clause in SQL

The `WITH` clause, also known as **Common Table Expression (CTE)**, is used to create a temporary result set that can be referenced within a `SELECT`, `INSERT`, `UPDATE`, or `DELETE` statement.

Advantages of WITH Clause

- Improves readability and simplifies complex queries.
- Helps avoid writing **subqueries** multiple times.
- Makes queries **modular** and **maintainable**.

```
SELECT customer_id, order_id, amount,
    SUM(amount) OVER(PARTITION BY customer_id ORDER BY order_id) AS running_total
FROM orders;
```

Ranking Window Functions: These functions **assign a rank** to each row based on a specified order.

RANK() - Assigns rank with gaps for duplicates
DENSE_RANK() - Assigns consecutive ranks (no gaps)
ROW_NUMBER() - Assigns unique row numbers
NTILE(n) - Divides rows **into n equal buckets** (e.g., quartiles).

Value Window Functions :

LAG() Gets previous row's value.
LEAD() Gets next row's value.

FIRST_VALUE() Returns first value in partition.
LAST_VALUE() Returns last value in partition.

View:

A view is a virtual table in SQL that consists of a stored query and does not store data itself.

Use a View

- If you frequently run a **complicated query**, you can create a **view**
- Instead of writing **long queries**, you can simply call the view.
- . Enhances Security by **hiding sensitive data** while allowing access to only necessary columns.
- Reduces Storage Cost

stored procedure in SQL?

A stored procedure is a precompiled collection of one or more SQL statements that can be executed as a single unit. It can accept input parameters, perform operations, and return output.

Benefits of Stored Procedures

✅ **Faster Execution** – Precompiled and optimized for performance.
✅ **Code Reusability** – Can be reused across different applications.
✅ **Security** – Prevents SQL injection by encapsulating queries.
✅ **Easier Maintenance** – Reduces redundancy and simplifies debugging.

Advanced Filtering & Subqueries

- Using HAVING Clause with Aggregate Functions
- Subqueries (Single-row, Multi-row, Correlated)
- EXISTS and NOT EXISTS
- Using CASE for Conditional Logic

ACID ensures reliable **transactions** in SQL databases.

Atomicity → All or nothing. If one part of the transaction fails, everything is rolled back.
Consistency → The database must stay valid before and after a transaction.
Isolation → Transactions don't interfere with each other.
Durability → Once committed, data is **permanently saved**, even after a system