SESSION WILL BE DIVIDED INTO

1. Subqueries in SQL
2. Drawbacks of subqueries
3. Common Table Expression (CTE)
4. Subqueries Vs CTE
5. CTE Syntax
6. Types of CTEs
7. Learning with Examples

SECTION 1

SUBQUERIES In SQL

What is a SUBQUERY

- In SQL, **subqueries** are one of the **most powerful** and flexible tools for writing **efficient queries**.
- A **subquery** is essentially a query nested within another query, allowing users to **perform operations** that depend on the results of **another query**.
- In simple terms, subqueries are nothing but queries within queries.

SECTION 2

DRAWBACKS

DRAWBACKS OF SUBQUERIES

1. Readability: Not good when reading query

2. Reusability: Subqueries are not reusable, meaning that if the same logic needs to be used in multiple places within a query or across multiple queries, it must be repeated.

3. Complexity: Subqueries can lead to complex SQL statements, which may be challenging to understand and debug.

SECTION 3

Common Table Expression (CTE) In SQL

CTE

- A **Common Table Expression (CTE)**, also referred to as a **WITH clause**, is a temporary named result set that you can reference anywhere in your query.
- In contrast to subqueries, which are inserted exactly where you need them, all CTEs are defined before the main query and are then referenced in the query using the assigned name.

SECTION 4

DIFFERENCE IN ONE TABLE

Topic	Subqueries	CTE
Definition	Nested queries within main query	Temporary result sets with WITH keyword
Reusability	Not reusable, limited to query scope	Reusable, can be referenced multiple times

SECTION 4

DIFFERENCE IN ONE TABLE

Topic	Subqueries	CTE
Definition	Nested queries within main query	Temporary result sets with WITH keyword
Reusability	Not reusable, limited to query scope	Reusable, can be referenced multiple times
Readability	May make queries complex	Enhances readability by modularizing code
Optimization	May lead to poor performance, especially correlated	Can optimize execution plan

SECTION 4

DIFFERENCE IN ONE TABLE

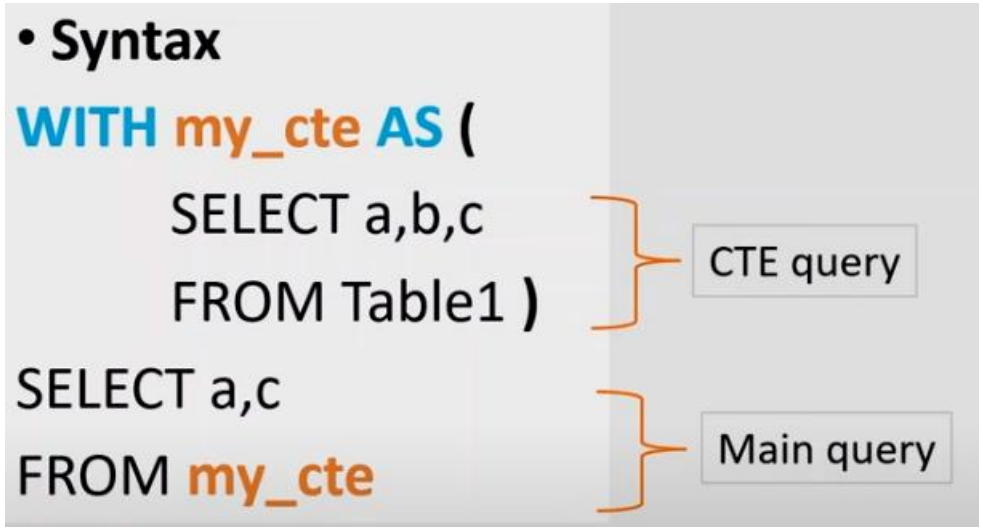
Topic	Subqueries	CTE
Definition	Nested queries within main query	Temporary result sets with WITH keyword
Reusability	Not reusable, limited to query scope	Reusable, can be referenced multiple times
Readability	May make queries complex	Enhances readability by modularizing code
Optimization	May lead to poor performance, especially correlated	Can optimize execution plan
Scope	Limited to query scope	Broader scope, accessible multiple times
Debugging	Challenging due to nested logic and complexity	Facilitates easier debugging by modularity

SECTION 5

SYNTAX

The syntax of a Common Table Expression (CTE) in SQL is as follows:

A **Common Table Expression (CTE)**, is defined by adding a WITH clause.



SECTION 6

TYPES OF CTE

CTE Types -

- Simple / Non-Recursive CTE
- Recursive CTE

SECTION 6

TYPES OF CTE

CTE Types -

- Simple / Non-Recursive CTE

SECTION 6

TYPES OF CTE

CTE Types -

- Simple / Non-Recursive CTE
- Recursive CTE
  - **Recursive CTE** is a CTE that references itself in order to perform a recursive operation. It's mainly used to:
    - Traverse hierarchical data (like organization charts, folder structures)
    - Generate sequences (like numbers from 1 to N).
  - A **Recursive CTE** has three elements:
    - **Non-recursive term**: It’s a CTE query definition that forms the base result set of the CTE structure
    - **Recursive Term**: One or more query definitions joined with using UNION or UNION ALL operator
    - **Termination Check**: The condition where the recursion needs to stop.

SECTION 6

TYPES OF CTE

CTE Types -

- Simple / Non-Recursive CTE
- Recursive CTE
  - **UNION**

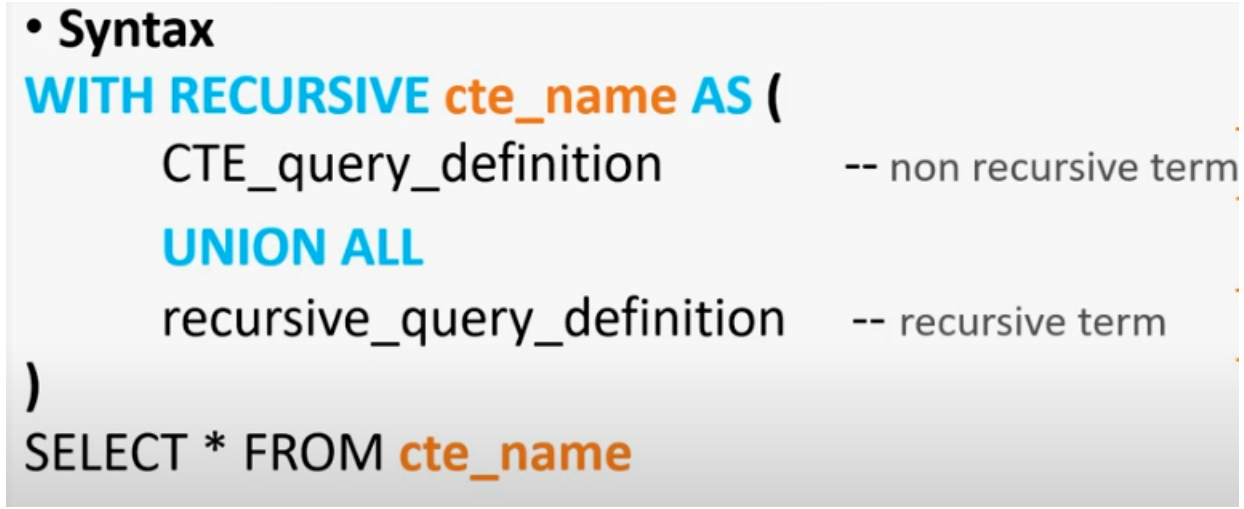
  &
  - **UNION ALL**

SECTION 6

TYPES OF CTE

CTE Types -

- Simple / Non-Recursive CTE
- Recursive CTE
  - SYNTAX TO FOLLOW



SECTION 6

TYPES OF CTE

CTE Types -

- Simple / Non-Recursive CTE
- Recursive CTE
  - SYNTAX TO FOLLOW