**COMMON TABLE EXPRESSIONS (CTEs):**

***Objective :***

To simplify complex queries and process hierarchical data using CTEs.

A Common Table Expression (CTE) is the result set of a query which exists temporarily and for use only within the context of a larger query. Much like a derived table, the result of a CTE is not stored and exists only for the duration of the query.

***Queries :***

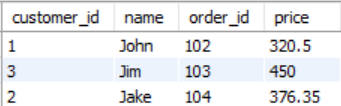
**with cte\_join as**

**(select Customers.customer\_id, Customers.name, Orders.order\_id, Orders.price from Customers inner join Orders on Customers.customer\_id = Orders.customer\_id**

**)**

**select \* from cte\_join having price >= (select avg(price) from orders);**

// The CTE joins the Customers and Orders tables, selecting the customer\_id, name, order\_id, and price for each customer and their order and selects all records from the cte\_join where the price is greater than or equal to the average price of all orders.



**with cte\_employee as**

**(select employee\_id, first\_name, last\_name, department, salary,**

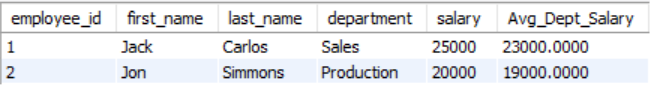
**(select avg(salary) from Employees where department=e.department) as Avg\_Dept\_Salary**

**from Employees e**

**)**

**select \* from cte\_employee where salary > Avg\_Dept\_Salary;**

// The CTE selects employee\_id, first\_name, last\_name, department, salary, and calculates the average salary for each department (Avg\_Dept\_Salary) for each employee and retrieves all records from the cte\_employee where the employee's salary is greater than the calculated Avg\_Dept\_Salary for their department.



**RECURSIVE QUERIES :**

A recursive CTE (Common Table Expression) allows a query to reference itself. It is typically used for hierarchical or tree-like data structures, such as organization charts, bill of materials, or file directory structures.

A recursive CTE consists of two parts:

1) Anchor Member: This is the initial query that provides the base result set.

2) Recursive Member: This is the part of the query that references the CTE itself, forming a loop to repeatedly process data.

**create table Employee\_Details(**

**employee\_id int primary key,**

**name varchar(30),**

**manager\_id int**

**)**

// Employee\_Details Schema

**with recursive employee\_hierarchy as**

**(**

*-- Anchor*

**select \* from Employee\_Details where employee\_id = 7**

**union all**

*-- Recursive member*

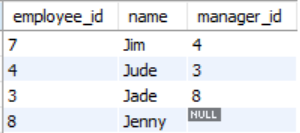
**select e.employee\_id, e.name, e.manager\_id from Employee\_Details e join employee\_hierarchy eh**

**on e.employee\_id = eh.manager\_id**

**)**

**select \* from employee\_hierarchy;**

// This SQL query uses a recursive CTE (Common Table Expression) to find all employees under a specific employee, starting from employee\_id = 7. This is a typical approach for navigating hierarchical data like organizational structures.



**Note:**  A CTE is temporary and exists only during the execution of the query.