The common table expression (CTE) is a temporary named result set that you can reference within a SELECT, INSERT, UPDATE, or DELETE statement. You can also use a CTE in a CREATE a view, as part of the view’s SELECT query. In addition, as of SQL Server 2008, you can add a CTE to the new MERGE statement.

**Using the CTE –**   
We can define CTEs by adding a WITH clause directly before SELECT, INSERT, UPDATE, DELETE, or MERGE statement. The WITH clause can include one or more CTEs separated by commas. The following syntax can be followed:

[WITH [, ...]]

::=

cte\_name [(column\_name [, ...])]

AS (cte\_query)

After you define your WITH clause with the CTEs, you can then reference the CTEs as you would refer any other table. However, you can refer a CTE only within the execution scope of the statement that immediately follows the WITH clause. After you’ve run your statement, the CTE result set is not available to other statements.

**Creating a Recursive Common Table Expression –**   
A recursive CTE is one that references itself within that CTE. The recursive CTE is useful when working with hierarchical data as the CTE continues to execute until the query returns the entire hierarchy.

A typical example of hierarchical data is a table that includes a list of employees. For every employee, the table provides a reference to that person’s manager. That reference is itself an employee ID within the same table. You can use a recursive CTE to display the hierarchy of employee data.

If a CTE is created incorrectly it can enter an infinite loop. To prevent this, the MAXRECURSION hint can be added in the OPTION clause of the primary SELECT, INSERT, UPDATE, DELETE, or MERGE statement.

Eg:-

with employeeCTE as

(

select \*,row\_number() over (partition by emp\_id order by emp\_id) as rownumber from employee

)

select \* from employeeCTE