|  |
| --- |
| * **CTEs** are great for simplifying queries within a single execution. * **Views** are ideal for reusable and persistent abstractions of complex queries. * **Temporary Tables** are useful for holding intermediate data that should only exist during the current session or transaction. * A very important distinction between Temp Tables and Views is that typically the creation of Views is limited to the Database Administrators. Meaning that you more than likely will not be able to go off and create your own view like you would a Temp Table, giving them yet another great use benefit. * TEMP.TABLE once created, they persist for the duration of the session or until explicitly dropped by the user. Unlike Common Table Expressions (CTEs), temporary tables offer extended reusability and are accessible to any user within the server while they exist. |

DIFFERRENCE BETWEEN CTE, VIEW AND TEMPORARY TABLE



|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
|  | | DIFFERRENCE BETWEEN CTE, VIEW AND TEMPORARY TABLE | | | |  | | |
|  |  | | | | | |  | |
|  | | | * **Common Table Expressions (CTE’s)** | | | | |  |
|  | | | **SYNTAX-**  Simply placing the *WITH*clause, a name for your CTE, adding your query and enclosing it in parentheses completes the CTE. *(Remember, if you are adding multiple CTE’s to add a comma after each one.)* | | | | |  |
|  | | | **DETAIL-** CTE’s are temporary result sets which you can reference within a SELECT, INSERT, UPDATE, or DELETE statement.This is similar to a subquery except the it is separate from the main query and defined by the WITH clause at the beginning of a query.  * CTE’s are short-lived and only exist for the duration of the query itself.   **BENEFITS-**   * CTE’s are beneficial for multiple reasons as they allow you to break down logic into smaller and more manageable parts, improve readability of complex queries, and reference the same CTE multiple times within the same query. * They are useful for making complex code more readable, breaking code into smaller parts and scenarios where you do not need to call on that particular result set repeatedly. | | | | |  |
|  | | | |  |  | | | |

* **VIEW**

**SYNTAX-**

We simply use the CREATE command but replace TABLE with VIEW, give it a name and insert the query creating the table. Similar to that of a Temp Table, Views are virtual tables which are based off of a result set of a SELECT statement query. Similar to Temp Tables, Views do not store the data itself but instead presents the result of a query as if it were a table.

**DETAIL-**

* A view is a stored query that can be treated like a table. It is essentially a virtual table created from a SELECT statement, and you can query it just like a table.
* A view exists permanently in the database schema until it is explicitly dropped.

**BENEFITS-** SAME AS TEMPORARY TABLE-

* **Data Security:** Views can also restrict user access to certain columns or rows. This still allows users to still have access to data in the table which is important to them, while protecting sensitive information that is not.
* **Simplicity:** Simplifying complex code is another benefit views share with Temp Tables and CTE’s.
* **Ease of Use and Shareability:** Once a View is created, anyone within the organization that has access to the database, has access to the View.
* **Data Independence:** Views allow changes to the underlying tables (such as adding or removing columns) without affecting the queries or applications using the View. So changes made to the original tables after the View is created, doesn’t affect the View itself, allowing the original table and data to remain independent and dynamic.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Decorative | |  | |  | | |  |  | |
|  | | TEMPORARY TABLE | | | | | |  | |
|  | |  | |  | | |  |  | |
|  | **SYNTAX-** Temporary Tables are created using the same CREATE TABLE command as any other table with one exception, a # placed in front of the table name. This identifies it and creates the table as a temp table. **DETAIL-**   * Temp tables are tables which are created to temporarily store data. * Temp tables, once created, exist for the duration of the session or until the user specifically drops it. * Temp Tables are stored in the Temp database/Schema of a database and not in tables. * A temporary table is a table that is created and stored in memory or on disk, but only for the duration of the session or transaction. It behaves like a regular table, but with automatic cleanup. | | | | | | | |  |
|  | **BENEFITS-**   * **Data Security:**  Using Temp tables gives you the ability to allow all users access to a table while limiting what they can see and protecting sensitive information.  * **Indexing / Enhanced Performance:**   Temp tables have the ability to be indexed, allowing you to further optimize performance of queries that involve searching or sorting based on specific columns. This can significantly increase data retrieval speed. | | | |  | * **Reduced Typing**:   Using a Temp Table when having to continuously type out the same complex code over and over to get a result is another benefit.   * **Debugging Ease**:   When debugging or analysing complex queries, temp tables can help break them up and store intermediate results at different stages, making the process easier to read and errors easier to identify. | | |  |
|  |  | |