



Transaction Control Language (TCS)

.NET

Software used to maintain relational databases is called a Relational Database Management System (RDBMS).

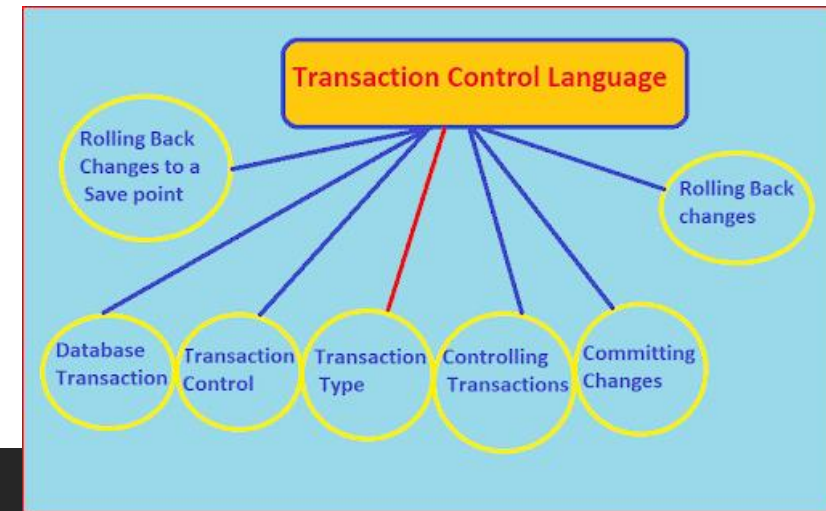
Many relational database systems use Structured Query Language (SQL) for querying and maintaining databases.

[HTTPS://DOCS.MICROSOFT.COM/EN-US/SQL/T-SQL/LANGUAGE-REFERENCE?](https://docs.microsoft.com/en-us/sql/t-sql/language-reference?)

Transaction Control Language (TCL)

<https://www.databasestart.com/dml-ddl-tcl-commands/>
<https://www.geeksforgeeks.org/sql-ddl-dml-tcl-dcl/>

Transaction Control Language commands are used to manage transactions in the database. These are used to manage the changes made by DML-statements. It also allows statements to be grouped together into logical transactions.



Commit

<https://www.geeksforgeeks.org/sql-ddl-dml-tcl-dcl/>

<https://docs.microsoft.com/en-us/sql/t-sql/language-elements/commit-transaction-transact-sql?view=sql-server-ver15>

The ***Commit*** command is used to permanently save any transaction into the database.

Commit marks the end of a successful ***implicit*** or ***explicit*** transaction. If **@@TRANCOUNT** is 1, **COMMIT TRANSACTION** makes all data modifications since the start of the transaction a permanent part of the database, it then frees the transaction's resources and decrements **@@TRANCOUNT** to 0. When **@@TRANCOUNT** is greater than 1, **COMMIT TRANSACTION** decrements **@@TRANCOUNT** only by 1 and the transaction stays active.

```
COMMIT [ { TRAN | TRANSACTION } [ transaction_name | @tran_name_variable ] ] [ WITH (
  DELAYED_DURABILITY = { OFF | ON } ) ]
[ ; ]
```

Rollback

<https://www.geeksforgeeks.org/sql-ddl-dml-tcl-dcl/>

<https://docs.microsoft.com/en-us/sql/t-sql/language-elements/commit-transaction-transact-sql?view=sql-server-ver15>

- **Rollback** restores the database to the last **committed** state. **Rollback** is also used **savepoint** to jump to a specific **savepoint** in a transaction.
- **Rollback** rolls back an explicit or implicit transaction to the beginning of the transaction, or to a **savepoint** inside the transaction.
- **ROLLBACK TRANSACTION** erases all data modifications made from the start of the transaction or to a **Savepoint**. It also frees resources held by the transaction.
- This does not include changes made to local variables or table variables. These are not erased by this statement.

```
ROLLBACK { TRAN | TRANSACTION }  
    [ transaction_name | @tran_name_variable  
    | savepoint_name | @savepoint_variable ]  
[ ; ]
```


Save

<https://www.geeksforgeeks.org/sql-ddl-dml-tcl-dcl/>

<https://docs.microsoft.com/en-us/sql/t-sql/language-elements/save-transaction-transact-sql?view=sql-server-ver15>

- **Save** is used to temporarily save a transaction so that you can **rollback** to that point whenever necessary.
- The **savepoint** defines a location to which a **transaction** can return if part of the **transaction** is conditionally canceled. You can even set a **savepoint** within a **transaction**.
- If a **transaction** is rolled back to a **savepoint**, it must proceed to completion with more **Transact-SQL statements** if needed and a **COMMIT TRANSACTION** statement, or it must be canceled altogether by rolling the **transaction** back to its beginning.
- To cancel an entire **transaction**, use **ROLLBACK TRANSACTION transaction_name** to undo all the **statements** or **procedures** of the **transaction**.
- Duplicate **savepoint** names are allowed in a **transaction**, but **ROLLBACK TRANSACTION** will only roll the **transaction** back to the most recent **SAVE TRANSACTION** with that name.

```
SAVE { TRAN | TRANSACTION } { savepoint_name | @savepoint_variable }  
[ ; ]
```