Trigger

A trigger is a special type of stored procedure that automatically runs when an event occurs in the database server. DML triggers run when a user tries to modify data through a data manipulation language (DML) event. DML events are INSERT, UPDATE, or DELETE statements on a table or view. These triggers fire when any valid event fires, whether table rows are affected or not. For
DDL triggers run in response to a variety of data definition language (DDL) events. These events primarily correspond to Transact-SQL CREATE, ALTER, and DROP statements, and certain system stored procedures that perform DDL-like operations.

Logon triggers fire in response to the LOGON event that's raised when a user's session is being established. You can create triggers directly from Transact-SQL statements or from methods of assemblies that are created in the Microsoft .NET Framework common language runtime (CLR) and uploaded to an instance of SQL Server. SQL Server lets you create multiple triggers for any specific statement.

DML Triggers
DML triggers is a special type of stored procedure that automatically takes effect when a data manipulation language (DML) event takes place that affects the table or view defined in the trigger. DML events include INSERT, UPDATE, or DELETE statements. DML triggers can be used to enforce business rules and data integrity, query other tables, and include complex Transact-SQL statements. The trigger and the statement that fires it are treated as a single transaction, which can be rolled back from within the trigger. If a severe error is detected (for example, insufficient disk space), the entire transaction automatically rolls back.

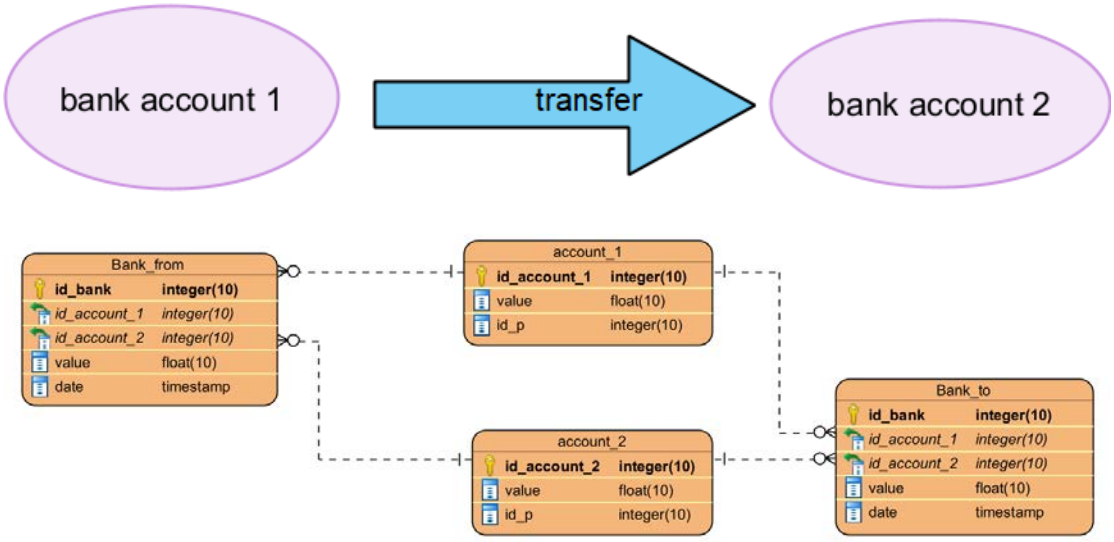
DML Trigger Benefits
DML triggers are similar to constraints in that they can enforce entity integrity or domain integrity. In general, entity integrity should always be enforced at the lowest level by indexes that are part of PRIMARY KEY and UNIQUE constraints or are created independently of constraints. Domain integrity should be enforced through CHECK constraints, and referential integrity (RI) should be enforced through FOREIGN KEY constraints. DML triggers are most useful when the features supported by constraints cannot meet the functional needs of the application.
The following list compares DML triggers with constraints and identifies when DML triggers have benefits over.

- DML triggers can cascade changes through related tables in the database; however, these changes can be executed more efficiently using cascading referential integrity constraints. FOREIGN KEY constraints can validate a column value only with an exact match to a value in another column, unless the REFERENCES clause defines a cascading referential action.
- They can guard against malicious or incorrect INSERT, UPDATE, and DELETE operations and enforce other restrictions that are more complex than those defined with CHECK constraints.
- Unlike CHECK constraints, DML triggers can reference columns in other tables. For example, a trigger can use a SELECT from another table to compare to the inserted or updated data and to perform additional actions, such as modify the data or display a user-defined error message.
- They can evaluate the state of a table before and after a data modification and take actions based on that difference.
- Multiple DML triggers of the same type (INSERT, UPDATE, or DELETE) on a table allow multiple, different actions to take place in response to the same modification statement.
- Constraints can communicate about errors only through standardized system error messages. If your application requires, or can benefit from, customized messages and more complex error handling, you must use a trigger.
- DML triggers can disallow or roll back changes that violate referential integrity, thereby canceling the attempted data modification. Such a trigger might go into effect when you change a foreign key and the new value does not match its primary key. However, FOREIGN KEY constraints are usually used for this purpose.
- If constraints exist on the trigger table, they are checked after the INSTEAD OF trigger execution but prior to the AFTER trigger execution. If the constraints are violated, the INSTEAD OF trigger actions are rolled back and the AFTER trigger is not executed.

Types of DML Triggers
AFTER trigger
AFTER triggers are executed after the action of the INSERT, UPDATE, MERGE, or DELETE statement is performed. AFTER triggers are never executed if a constraint violation occurs; therefore, these triggers cannot be

used for any processing that might prevent constraint violations. For every INSERT, UPDATE, or DELETE action specified in a MERGE statement, the corresponding trigger is fired for each DML operation.

INSTEAD OF trigger
INSTEAD OF triggers override the standard actions of the triggering statement. Therefore, they can be used to perform error or value checking on one or more columns and the perform additional actions before insert, updating or deleting the row or rows. For example, when the value being updated in an hourly wage column in a payroll table exceeds a specified value, a trigger can be defined to either produce an error message and roll back the transaction, or insert a new record into an audit trail before inserting the record into the payroll table. The primary advantage of INSTEAD OF triggers is that they enable views that would not be updatable to support updates. For example, a view based on multiple base tables must use an INSTEAD OF trigger to support inserts, updates, and deletes that reference data in more than one table. Another advantage of INSTEAD OF triggers is that they enable you to code logic that can reject parts of a batch while letting other parts of a batch to succeed.



What are the SQL database functions?

Aggregate functions

Aggregate functions perform a calculation on a set of values and return a single value. They're allowed in the select list or the HAVING clause of a SELECT statement. You can use an aggregation in combination with the GROUP BY clause to calculate the aggregation on categories of rows. Use the OVER clause to calculate the aggregation on a specific range of value. The OVER clause can't follow the GROUPING or GROUPING_ID aggregations.

All aggregate functions are deterministic, which means they always return the same value when they run on the same input values. For more information, see Deterministic and Nondeterministic Functions.

Analytic functions

Analytic functions compute an aggregate value based on a group of rows. However, unlike aggregate functions, analytic functions can return multiple rows for each group. You can use analytic functions to compute moving averages, running totals, percentages, or top-N results within a group.

Bit manipulation functions

Bit manipulation functions allow you to process and store data more efficiently than with individual bits. For more information, see Bit manipulation functions.

Ranking functions

Ranking functions return a ranking value for each row in a partition. Depending on the function that is used, some rows might receive the same value as other rows. Ranking functions are nondeterministic.

Rowset functions

Rowset functions Return an object that can be used like table references in an SQL statement.

Scalar functions

Operate on a single value and then return a single value. Scalar functions can be used wherever an expression is valid.

Categories of scalar functions

Function category	Description
Configuration Functions	Return information about the current configuration.
Conversion Functions	Support data type casting and converting.
Cursor Functions	Return information about cursors.
Date and Time Data Types and Functions	Perform operations on a date and time input values and return string, numeric, or date and time values.
Graph Functions	Perform operations to convert to and from character representations of graph node and edge IDs.
JSON Functions	Validate, query, or change JSON data.
Logical Functions	Perform logical operations.
Mathematical Functions	Perform calculations based on input values provided as parameters to the functions, and return numeric values.
Metadata Functions	Return information about the database and database objects.
Security Functions	Return information about users and roles.
String Functions	Perform operations on a string (**char** or **varchar**) input value and return a string or numeric value.
System Functions	Perform operations and return information about values, objects, and settings in an instance of SQL Server.
System Statistical Functions	Return statistical information about the system.
Text and Image Functions	Perform operations on text or image input values or columns, and return information about the value.

Function determinism

SQL Server built-in functions are either deterministic or nondeterministic. Functions are deterministic when they always return the same result anytime they're called by using a specific set of input values. Functions are nondeterministic when they could return different results every time they're called, even with the same specific set of input values. For more information, see Deterministic and Nondeterministic Functions

Function collation

Functions that take a character string input and return a character string output use the collation of the input string for the output.

Functions that take non-character inputs and return a character string use the default collation of the current database for the output.

Functions that take multiple character string inputs and return a character string use the rules of collation precedence to set the collation of the output string. For more information, see Collation Precedence