**What is a transaction log?**

A transaction log is a file – integral part of every SQL Server database. It contains log records produced during the logging process in a SQL Server database. The transaction log is the most important component of a SQL Server database when it comes to the disaster recovery – however, it must be uncorrupted. After each database modification – transaction occurrence, a log record is written to the transaction log. All the changes are written sequentially.

## What does a SQL Server transaction log store?

A transaction log stores every transaction made to a SQL Server database, except some which are minimally logged like BULK IMPORT or SELECT INTO. Internally it is split into the smaller parts called Virtual Log Files (VLFs). When one VLF becomes full logging continue to write into the next available in the transaction log. The transaction log file can be represented as a circular file. When the logging reaches the end of the file it starts again from the beginning, but only if all the requirements has been met and the inactive parts has been truncated. The truncation process is necessary to mark all inactive parts so they can be used again and overwritten

*A log record is no longer needed in the transaction log if all of the following are true:*

* *The transaction of which it is part has committed*
* *The database pages it changed have all been written to disk by a checkpoint*
* *The log record is not needed for a backup (full, differential, or log)*
* *The log record is not needed for any feature that reads the log (such as database mirroring or replication)*

Marks the starting point of an explicit, local transaction. BEGIN TRANSACTION increments @@TRANCOUNT by 1.

Topic link icon [Transact-SQL Syntax Conventions](https://msdn.microsoft.com/en-us/library/ms177563.aspx)

## [Syntax](javascript:void(0))

BEGIN { TRAN | TRANSACTION }

[ { transaction\_name | @tran\_name\_variable }

[ WITH MARK [ 'description' ] ]

]

[ ; ]

## [Arguments](javascript:void(0))

## transaction\_name Is the name assigned to the transaction. transaction\_name must conform to the rules for identifiers, but identifiers longer than 32 characters are not allowed. Use transaction names only on the outermost pair of nested BEGIN...COMMIT or BEGIN...ROLLBACK statements. transaction\_name is always case sensitive, even when the instance of SQL Server is not case sensitive.

* **@tran\_name\_variable**  
  Is the name of a user-defined variable containing a valid transaction name. The variable must be declared with a **char**, **varchar**, **nchar**, or **nvarchar** data type. If more than 32 characters are passed to the variable, only the first 32 characters will be used; the remaining characters will be truncated.
* **WITH MARK [ 'description' ]**  
  Specifies that the transaction is marked in the log. description is a string that describes the mark. A description longer than 128 characters is truncated to 128 characters before being stored in the msdb.dbo.logmarkhistory table.
* If WITH MARK is used, a transaction name must be specified. WITH MARK allows for restoring a transaction log to a named mark.

**COMMIT TRANSACTION**

COMMIT [ { TRAN | TRANSACTION } [ transaction\_name | @tran\_name\_variable ] ] [ WITH ( DELAYED\_DURABILITY = { OFF | ON } ) ]

[ ; ]

# **ROLLBACK TRANSACTION (Transact-SQL)**

**THIS TOPIC APPLIES TO:**yesSQL Server (starting with 2008)yesAzure SQL DatabaseyesAzure SQL Data Warehouse yesParallel Data Warehouse

Rolls back an explicit or implicit transaction to the beginning of the transaction, or to a savepoint inside the transaction. You can use ROLLBACK TRANSACTION to erase all data modifications made from the start of the transaction or to a save point. It also frees resources held by the transaction.

## [Syntax](javascript:void(0))

ROLLBACK { TRAN | TRANSACTION }

[ transaction\_name | @tran\_name\_variable

| savepoint\_name | @savepoint\_variable ]

[ ; ]

## [Arguments](javascript:void(0))

**transaction\_name**  
Is the name assigned to the transaction on BEGIN TRANSACTION. transaction\_name must conform to the rules for identifiers, but only the first 32 characters of the transaction name are used. When nesting transactions, transaction\_name must be the name from the outermost BEGIN TRANSACTION statement. transaction\_name is always case sensitive, even when the instance of SQL Server is not case sensitive.

**@ tran\_name\_variable**  
Is the name of a user-defined variable containing a valid transaction name. The variable must be declared with a **char**, **varchar**, **nchar**, or **nvarchar** data type.

**savepoint\_name**Is savepoint\_name from a SAVE TRANSACTION statement. savepoint\_name must conform to the rules for identifiers. Usesavepoint\_name when a conditional rollback should affect only part of the transaction.

**@ savepoint\_variable**Is name of a user-defined variable containing a valid savepoint name. The variable must be declared with a **char**, **varchar**, **nchar**, or**nvarchar** data type.

## [Error Handling](javascript:void(0))

A ROLLBACK TRANSACTION statement does not produce any messages to the user. If warnings are needed in stored procedures or triggers, use the RAISERROR or PRINT statements. RAISERROR is the preferred statement for indicating errors.

**DEFAULT PORTS**

TCP 1433

TCP port 1433 is the default port for SQL Server. This port is also the official Internet Assigned Number Authority (IANA) socket number for SQL Server. Client systems use TCP 1433 to connect to the database engine; SQL Server Management Studio (SSMS) uses the port to manage SQL Server instances across the network. You can reconfigure SQL Server to listen on a different port, but 1433 is by far the most common implementation.

TCP 1434

TCP port 1434 is the default port for the Dedicated Admin Connection. You can start the Dedicated Admin Connection through sqlcmd or by typing ADMIN: followed by the server name in the SSMS Connect to Database Engine dialog box.

UDP 1434

UDP port 1434 is used for SQL Server named instances. The SQL Server Browser service listens on this port for incoming connections to a named instance. The service then responds to the client with the TCP port number for the requested named instance.

TCP 2383

TCP port 2383 is the default port for SQL Server Analysis Services.

TCP 2382

TCP port 2382 is used for connection requests to a named instance of Analysis Services. Much like the SQL Server Browser service does for the relational database engine on UDP 1434, the SQL Server Browser listens on TCP 2382 for requests for Analysis Services named instances. Analysis Services then redirects the request to the appropriate port for the named instance.

TCP 135

TCP port 135 has several uses. The Transact-SQL debugger uses the port. TCP 135 is also used to start, stop, and control SQL Server Integration Services, although it is required only if you connect to a remote instance of the service from SSMS.

TCP 80 and 443

TCP ports 80 and 443 are most typically used for report server access. However, they also support URL requests to SQL Server and Analysis Services. TCP 80 is the standard port for HTTP connections that use a URL. TCP 443 is used for HTTPS connections that use secure sockets layer (SSL).

Unofficial TCP Ports

Microsoft uses TCP port 4022 for SQL Server Service Broker examples in SQL Server Books Online. Likewise, BOL Database Mirroring examples use TCP port 7022.

MySQL 5.5 Supported Storage Engines

* [InnoDB](https://dev.mysql.com/doc/refman/5.5/en/innodb-storage-engine.html): The default storage engine as of MySQL 5.5.5. InnoDB is a transaction-safe (ACID compliant) storage engine for MySQL that has commit, rollback, and crash-recovery capabilities to protect user data. InnoDB row-level locking (without escalation to coarser granularity locks) and Oracle-style consistent nonlocking reads increase multi-user concurrency and performance. InnoDB stores user data in clustered indexes to reduce I/O for common queries based on primary keys. To maintain data integrity, InnoDB also supports FOREIGN KEY referential-integrity constraints. For more information about InnoDB, see [Chapter 14, *The InnoDB Storage Engine*](https://dev.mysql.com/doc/refman/5.5/en/innodb-storage-engine.html).
* [MyISAM](https://dev.mysql.com/doc/refman/5.5/en/myisam-storage-engine.html): The MySQL storage engine that is used the most in Web, data warehousing, and other application environments. MyISAM is supported in all MySQL configurations, and is the default storage engine prior to MySQL 5.5.5.
* [Memory](https://dev.mysql.com/doc/refman/5.5/en/memory-storage-engine.html): Stores all data in RAM for extremely fast access in environments that require quick lookups of reference and other like data. This engine was formerly known as the HEAP engine.
* [Merge](https://dev.mysql.com/doc/refman/5.5/en/merge-storage-engine.html): Enables a MySQL DBA or developer to logically group a series of identical MyISAM tables and reference them as one object. Good for VLDB environments such as data warehousing.
* [Archive](https://dev.mysql.com/doc/refman/5.5/en/archive-storage-engine.html): Provides the perfect solution for storing and retrieving large amounts of seldom-referenced historical, archived, or security audit information.
* [Federated](https://dev.mysql.com/doc/refman/5.5/en/federated-storage-engine.html): Offers the ability to link separate MySQL servers to create one logical database from many physical servers. Very good for distributed or data mart environments.
* [NDB](https://dev.mysql.com/doc/refman/5.5/en/mysql-cluster.html) (also known as [NDBCLUSTER](https://dev.mysql.com/doc/refman/5.5/en/mysql-cluster.html))—This clustered database engine is particularly suited for applications that require the highest possible degree of uptime and availability.

**Note**

The [NDB](https://dev.mysql.com/doc/refman/5.5/en/mysql-cluster.html) storage engine is not supported in standard MySQL 5.5 releases. Currently supported MySQL Cluster releases include MySQL Cluster NDB 7.0 and MySQL Cluster NDB 7.1, which are based on MySQL 5.1, and MySQL Cluster NDB 7.2, which is based on MySQL 5.5. While based on MySQL Server, these releases also contain support for [NDB](https://dev.mysql.com/doc/refman/5.5/en/mysql-cluster.html).

* [CSV](https://dev.mysql.com/doc/refman/5.5/en/csv-storage-engine.html): The CSV storage engine stores data in text files using comma-separated values format. You can use the CSV engine to easily exchange data between other software and applications that can import and export in CSV format.
* [Blackhole](https://dev.mysql.com/doc/refman/5.5/en/blackhole-storage-engine.html): The Blackhole storage engine accepts but does not store data and retrievals always return an empty set. The functionality can be used in distributed database design where data is automatically replicated, but not stored locally.
* [Example](https://dev.mysql.com/doc/refman/5.5/en/example-storage-engine.html): The Example storage engine is “stub” engine that does nothing. You can create tables with this engine, but no data can be stored in them or retrieved from them. The purpose of this engine is to serve as an example in the MySQL source code that illustrates how to begin writing new storage engines. As such, it is primarily of interest to developers.

It is important to remember that you are not restricted to using the same storage engine for an entire server or schema: you can use a different storage engine for each table in your schema.

***Choosing a Storage Engine***

The various storage engines provided with MySQL are designed with different use cases in mind. To use the pluggable storage architecture effectively, it is good to have an idea of the advantages and disadvantages of the various storage engines. The following table provides an overview of some storage engines provided with MySQL:

**Table 15.1 Storage Engines Feature Summary**

| **Feature** | **MyISAM** | **Memory** | **InnoDB** | **Archive** | **NDB** |
| --- | --- | --- | --- | --- | --- |
| Storage limits | 256TB | RAM | 64TB | None | 384EB |
| Transactions | No | No | Yes | No | Yes |
| Locking granularity | Table | Table | Row | Row | Row |
| MVCC | No | No | Yes | No | No |
| Geospatial data type support | Yes | No | Yes | Yes | Yes |
| Geospatial indexing support | Yes | No | Yes[[a]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html" \l "ftn.idm140094999260160) | No | No |
| B-tree indexes | Yes | Yes | Yes | No | No |
| T-tree indexes | No | No | No | No | Yes |
| Hash indexes | No | Yes | No[[b]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html" \l "ftn.idm140094999252288) | No | Yes |
| Full-text search indexes | Yes | No | Yes[[c]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html" \l "ftn.idm140094999249328) | No | No |
| Clustered indexes | No | No | Yes | No | No |
| Data caches | No | N/A | Yes | No | Yes |
| Index caches | Yes | N/A | Yes | No | Yes |
| Compressed data | Yes[[d]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html" \l "ftn.idm140094999239760) | No | Yes[[e]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html" \l "ftn.idm140094999238448) | Yes | No |
| Encrypted data[[f]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html" \l "ftn.idm140094999236672) | Yes | Yes | Yes | Yes | Yes |
| Cluster database support | No | No | No | No | Yes |
| Replication support[[g]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html" \l "ftn.idm140094999231200) | Yes | Yes | Yes | Yes | Yes |
| Foreign key support | No | No | Yes | No | No |
| Backup / point-in-time recovery[[h]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html" \l "ftn.idm140094999225792) | Yes | Yes | Yes | Yes | Yes |
| Query cache support | Yes | Yes | Yes | Yes | Yes |
| Update statistics for data dictionary | Yes | Yes | Yes | Yes | Yes |
| [[a]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html#idm140094999260160)InnoDB support for geospatial indexing is available in MySQL 5.7.5 and higher.  [[b]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html#idm140094999252288)InnoDB utilizes hash indexes internally for its Adaptive Hash Index feature.  [[c]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html#idm140094999249328)InnoDB support for FULLTEXT indexes is available in MySQL 5.6.4 and higher.  [[d]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html#idm140094999239760)Compressed MyISAM tables are supported only when using the compressed row format. Tables using the compressed row format with MyISAM are read only.  [[e]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html#idm140094999238448)Compressed InnoDB tables require the InnoDB Barracuda file format.  [[f]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html#idm140094999236672)Implemented in the server (via encryption functions). Data-at-rest tablespace encryption is available in MySQL 5.7 and higher.  [[g]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html#idm140094999231200)Implemented in the server, rather than in the storage engine.  [[h]](https://dev.mysql.com/doc/refman/5.5/en/storage-engines.html#idm140094999225792)Implemented in the server, rather than in the storage engine. | | | | | |