MySQL storage engines

A storage engine is a software module that a database management system uses to create, read, update data from a database. There are two types of storage engines in MySQL: transactional and non-transactional.

For MySQL 5.5 and later, the default storage engine is *InnoDB*. The default storage engine for MySQL prior to version 5.5 was *MyISAM*. Choosing the right storage engine is an important strategic decision, which will impact future development. If you are new to MySQL and you are studying the MySQL database management system, then this is not much of a concern. If you are planning a production database, then things become more complicated.

List of storage engines

MySQL supported storage engines:

- InnoDB
- MyISAM
- Memory
- CSV
- Merge
- Archive
- Federated
- Blackhole
- Example

InnoDB is the most widely used storage engine with transaction support. It is an ACID compliant storage engine. It supports row-level locking, crash recovery and multi-version concurrency control. It is the only engine which provides foreign key referential integrity constraint. Oracle recommends using InnoDB for tables except for specialized use cases.

MyISAM is the original storage engine. It is a fast storage engine. It does not support transactions. MyISAM provides table-level locking. It is used mostly in Web and data warehousing.

Memory storage engine creates tables in memory. It is the fastest engine. It provides table-level locking. It does not support transactions. Memory storage engine is ideal for creating temporary tables or quick lookups. The data is lost when the database is restarted.

CSV stores data in CSV files. It provides great flexibility because data in this format is easily integrated into other applications.

Merge operates on underlying MyISAM tables. Merge tables help manage large volumes of data more easily. It logically groups a series of identical MyISAM tables, and references them as one object. Good for data warehousing environments.

Archive storage engine is optimised for high speed inserting. It compresses data as it is inserted. It does not support transactions. It is ideal for storing and retrieving large amounts of seldom referenced historical, archived data.

The *Blackhole* storage engine accepts but does not store data. Retrievals always return an empty set. The functionality can be used in distributed database design where data is automatically replicated, but not stored locally. This storage engine can be used to perform performance tests or other testing.

Federated storage engine offers the ability to separate MySQL servers to create one logical database from many physical servers. Queries on the local server are automatically executed on the remote (federated) tables. No data is stored on the local tables. It is good for distributed environments.

```
mysql> SHOW ENGINES\G
*************************** 1. row ***************************
      Engine: InnoDB
     Support: DEFAULT
     Comment: Supports transactions, row-level locking, and foreign keys
Transactions: YES
          XA: YES
  Savepoints: YES
*************************** 2. row ***************************
      Engine: CSV
     Support: YES
     Comment: CSV storage engine
Transactions: NO
          XA: NO
  Savepoints: NO
...
```

The `SHOW ENGINES` command shows all available engines that the server supports.

Choosing the right engine

No storage engine is ideal for all circumstances. Some perform best under certain conditions and perform worse in other situations. There are tradeoffs than must be considered. A more secure solution takes more resources; it might be slower, take more CPU time, and disk space. MySQL is very flexible in the fact that it provides several different storage engines. Some of them, like the Archive engine, are created to be used in specific situations.

In some cases the answer is clear. Whenever we are dealing with some payment systems, we are obliged to use the most secure solution. We cannot afford to loose such sensitive data. InnoDB is the way to go. If we want full-text search, then we can choose either MyISAM or InnoDB.. Only InnoDB supports foreign key referential integrity constraint and if we plan to use this constraint, then the choice is clear.

Specifying and altering storage engines

The storage engine is specified at the time of the table creation.

```
mysql> CREATE TABLE Cars(Id INTEGER PRIMARY KEY, Name VARCHAR(50),
    -> Cost INTEGER) ENGINE='MyISAM';
```

The `ENGINE` keyword specifies the storage engine used for this particular table.

If we do not specify the storage engine explicitly, then the default storage engine is used. Prior to MySQL 5.5 the default storage engine was MyISAM. For MySQL 5.5 and later, the default storage engine is InnoDB.

It is possible to migrate to a different storage engine. Note that migrating a large table might take a long time. Also we might run into some problems when migrating tables. Some features might not be supported in both tables.

```
mysql> SELECT ENGINE FROM information_schema.TABLES
    -> WHERE TABLE_SCHEMA='mydb'
    -> AND TABLE_NAME='Cars';
+--------+
| ENGINE |
+--------+
| InnoDB |
+--------+
1 row in set (0,05 sec)
```

This SQL statement finds out the storage engine used for a `Cars` table in `mydb` database. We could also use `SELECT CREATE TABLE Cars` SQL statement. The `information_schema` is a table which stores technical information about our tables.

```
mysql> ALTER TABLE Cars ENGINE='MyISAM';
```

This SQL statement changes the storage engine of the `Cars` table to MyISAM.

```
mysql> SELECT ENGINE FROM information_schema.TABLES
    -> WHERE TABLE_SCHEMA='mydb'
    -> AND TABLE_NAME='Cars';
+--------+
| ENGINE |
+--------+
| MyISAM |
+--------+
1 row in set (0,00 sec)
```

Now the storage engine of the table is MyISAM.

In this part of the MySQL tutorial, we have covered storage engines.