MySQL Performance: MyISAM vs InnoDB

Posted on August 6, 2018 by Jason Potter | Updated: March 12, 2021
Category: Series | Tags: Command-line, Data Management, Database, Innodb, MyISAM, MySQL, Performance, Storage Engine

MySQL Performance

Reading Time: 5 minutes

A major factor in database performance is the storage engine used by the database, and

Liquid Web™

What is locking in MySQL?

To protect the integrity of the data stored within databases, MySQL employs locking. Locking, simply put, means protecting data from being accessed. When a lock is applied, the data cannot be modified except by the query that initiated the lock. Locking is a necessary component to ensure the accuracy of the stored information. Each storage engine has a different method of locking used. Depending on your data and query practices, one engine can outperform another. In this series, we will look at the two most common types of locking employed by our two storage engines.

Table locking: The technique of locking an entire table when one or more cells within the table need to be updated or deleted. Table locking is the default method employed by the default storage engine, MyISAM.

Example: MyISAM Table Locking

Column A

Column B

Column C

Query 1 UPDATE

Row 1

Writing

data

data

Query 2 SELECT (Wait)

Row 2

data

data

data

data

Query 4 SELECT (Wait)

Row 4

data

data

data

Query 5 SELECT (Wait)

Row 5

data

data

data

The example illustrates how a single write operation locks the entire table causing other queries to wait for the UPDATE query finish.

Row-level locking: The act of locking an effective range of rows in a table while one or more cells within the range are modified or deleted. Row-level locking is the method used by the InnoDB storage engine and is intended for high-performance databases.

Example: InnoDB Row-Level Locking

Column A

Column A

Column A

Query 1 UPDATE

Row 1

Row 2

Reading

data

data

Query 3 UPDATE

Row 3

data

Writing

data

Query 4 SELECT

Row 4

Reading

Reading

Reading

Query 5 SELECT

Row 5

Reading

data

Reading

The example shows how using row-level locking allows for multiple queries to run on individual rows by locking only the rows being updated instead of the entire table.

Liquid Web™

Intimate knowledge of the database structure and query practices is critical for selecting the best storage engine for your tables.

MyISAM will out-perform InnoDB on large tables that require vastly more read activity versus write activity. MyISAM's readabilities outshine InnoDB because locking the entire table is quicker than figuring out which rows are locked in the table. The more information in the table, the more time it takes InnoDB to figure out which ones are not accessible. If your application relies on huge tables that do not change data frequently, then MyISAM will out-perform InnoDB. Conversely, InnoDB outperforms MyISAM when data within the table changes frequently. Table changes write data more than reading data per second. In these situations, InnoDB can keep up with large amounts of requests easier than locking the entire table for each one.

Should I use InnoDB with WordPress, Magento or Joomla Sites?

The short answer here is yes, in most cases. Liquid Web's Most Helpful Humans in Hosting Support Teams have encountered several table-locking bottlenecks when clients are using some standard web applications of today. Most users of popular third-party applications like WordPress, Magento, and Joomla have limited knowledge of the underlying database components or code involved to make an informed decision on storage engines. Most table-locking bottlenecks from these content management systems (CMS) are generally resolved by changing all the tables for the site over to InnoDB instead of the default MyISAM. If you are hosting many of these types of CMS on your server, it would be beneficial to change the default storage engine in MySQL to use InnoDB for all new tables so that any new table installations start off with InnoDB.

Setting the Default Storage Engine

Set your default storage engine to InnoDB by adding **default_storage_engine=InnoDB** to the **[mysqld]** section of the system config file located at: /etc/my.cnf. Restarting the MySQL service is necessary for the server to detect changes to the file.

```
~ $ cat /etc/my.cnf
[mysqld]
log-error=/var/lib/mysql/mysql.err
```

Unfortunately, MySQL does not inherently have an option to convert tables, leaving each table to be changed individually. Liquid Web's support team has put together an easy-to-follow maintenance plan for this process. The script, which you can run on the necessary server via shell access (SSH) will convert all tables between storage engines.

> **Note**
>
> Plan accordingly when performing batch operations of this nature just in case downtime occurs. Best practice is to back up all your MySQL Databases before implementing a change of this magnitude, doing so provides an easy recovery point to prevent any data loss.

Step 1: Prep

Plan to start at a time of day when downtime would have minimal consequences. This process itself does not require any downtime, however, downtime may be necessary to recover from unforeseen circumstances.

Step 2: Backup All Databases To A File

The command below creates a single file backup of all databases named **all-databases-backup.sqld** and can be deleted once the conversion has succeeded and there are no apparent problems.

```
mysqldump --all-databases > all-databases-backup.sql
```

Step 3: Record Existing Table Engines To A File

Run the following script to record the existing table engines to a file named **table-engine-backup.sql**. You can then "import" or "run" this file later to convert back to their original engines if necessary.

```
mysql -Bse 'SELECT CONCAT("ALTER TABLE ",table_schema,".",table_name,"
ENGINE=",Engine,";") FROM information_schema.tables WHERE table_schema NOT
IN("mysql","information_schema","performance_schema");' | tee table-engine-
backup.sql
```

If you need to revert the table engines back for any reason, run:

```
ENGINE=InnoDB;") FROM information_schema.tables WHERE table_schema NOT IN
("mysql","information_schema","performance_schema") AND Engine = "MyISAM";'
| while read -r i; do echo $i; mysql -e "$i"; done | tee convert-to-
innodb.log
```

Step 4b: Convert All InnoDB Tables To MyISAM

This command will proceed even if a table fails and lets you know which tables failed to convert. The output is also saved to the file named **convert-to-myisam.log** for later review.

```
mysql -Bse 'SELECT CONCAT("ALTER TABLE ",table_schema,".",table_name,"
ENGINE=MyISAM;") FROM information_schema.tables WHERE table_schema NOT IN
("mysql","information_schema","performance_schema") AND Engine = "InnoDB";'
| while read -r i; do echo $i; mysql -e "$i"; done | tee convert-to-
myisam.log
```

Converting A Single Table Between MyISAM and InnoDB

The following commands illustrate how converting a single table is accomplished.

> **Note**
>
> Replace database_name with the proper database name and table_name with the correct table name. Make sure you have a valid backup of the table in question before proceeding.

Backup A Single Table To A File

```
mysqldump database_name table_name > backup-table_name.sql
```

Convert A Single Table To InnoDB

```
mysql -Bse ‘ALTER TABLE database_name.table_name ENGINE=InnoDB;’
```

Convert A Single Table To MyISAM:

Check out our other articles in this series, MySQL Performance: Identifying Long Queries, to pinpoint slow queries within your database. Stay tuned for our next article where we will cover caching and optimization.

Series Navigation

<< Previous Article

Next Article >>

Related Articles:

- MySQL Performance: How To Leverage MySQL Database Indexing
- MySQL Performance: Identifying Long Queries
- Apache Performance Tuning: Swap Memory
- Create a MySQL User on Linux via Command Line
- Grant Permissions to a MySQL User on Linux via Command Line
- Use SFTP To Connect To WordPress Optimized Template

About the Author: Jason Potter

A veteran of the IT Support field, I have more than a decade of experience in systems administration, web hosting, and cPanel servers. I enjoy writing and providing complicated technical concepts in layman terms. On my free time, I enjoy playing several types video games, automation scripting and just living life with my wife and two kids.

Categories

Series	62
Technical Support	409
Tutorials	796

Have Some Questions?

Our Sales and Support teams are available 24 hours by phone or e-mail to assist.

1.800.580.4985
1.517.322.0434

Latest Articles

OTHER

Types of VMware Disks

Read Article

TUTORIALS

Liquid Web™

COMMON FIXES

Five Steps to Create a Robots.txt File for Your Website

Read Article

PRODUCTS

Premium Business Email Pricing FAQ

Read Article

SECURITY BULLETINS

Microsoft Exchange Server Security Update

Read Article

Want More Great Content Sent to Your Inbox?