MySQL Engines - MyISAM vs Innodb

Article ID: **681** | Last updated on **May 30, 2013** | Authored by: **Rackspace Support**

MySQL supports several different types of Table Engines also known as "Table Types". A database can have its tables being a mix of different table engine types or all of the same type. Here is more information on each of the different types of table engines that MySQL offers:

http://dev.mysql.com/doc/refman/5.0/en/storage-engines.html

The two most commonly used on most Cloud Sites MySQL servers use **Innodb** and **MyISAM** engines.

The purpose of this document is to briefly cover the two types and identify which ones are more recommended under what circumstances in the Cloud Sites environment. Please note that the purpose of this document is however not to go over a performance comparison of each of the two engine types as far as comparing via running specific sql test benchmarks, which if you are interested are well done on the two links below:

http://www.mysqlperformanceblog.com/2007/01/08/innodb-vs-myisam-vs-falcon-benchmarks-part-1/

http://tag1consulting.com/MySQL_Engines_MyISAM_vs_InnoDB#comment-115

MyISAM is the default table engine type for MySQL 5.0 but the Cloud Sites environment defaults the storage engine to Innodb. In other words Cloud Sites is partial to Innodb if you do not explicity specify your engine type in your table DDL. We have also tuned the database servers to generally perform best with using the Innodb Engine type.

Contents

MyISAM vs Innodb - Quick comparison Table:

MyISAM	Innodb
Not *ACID compliant and non-transactional	*ACID compliant and hence fully transactional with ROLLBACK and COMMIT and support for Foreign Keys
MySQL 5.0 Default Engine	Rackspace Cloud Default Engine
Offers Compression	Offers Compression
Requires full repair/rebuild of indexes/tables	Auto recovery from crash via replay of logs
Changed Db pages written to disk instantly	Dirty pages converted from random to sequential before commit and flush to disk
No ordering in storage of data	Row data stored in pages in PK order
Table level locking	Row level locking

- ACID - Atomicity, Consistency, Isolation, Durability (read more on it here: http://en.wikipedia.org/wiki/ACID)

If you need to see further details on each of the two engine types, please refer to the following MySQL documentations:

- Innodb Storage Engine: http://dev.mysql.com/doc/refman/5.0/en/innodb-storage-engine.html
- MyISAM Storage Engine: http://dev.mysql.com/doc/refman/5.0/en/myisam-storage-engine.html

When MyISAM tables are seen to be mostly useful?

There can be several other reasons that fit your requirement for choosing the MyISAM engine. For example reads can be faster on MyISAM vs Innodb despite what the general claims on the above two links when MyISAM table has fixed (not dynamic) row size i.e. when it uses more CHARs for example versus VARCHARs. Still there could be other reasons besides this why you choose or have chosen MyISAM over Innodb. Another reason why you may have chosen MyISAM over Innodb is perhaps due to the fact that Innodb must perform additional checks owing to its ACID compliant nature - so for example a FK check needs to be checked which could potentially cause an operational overhead. Unless you have benchmarked this to be the case, I would not recommend you believing this to be the case as default as per the links above, you may find out otherwise.

When we have seen conversion of table engine from MyISAM to Innodb as being most beneficial?

1. If you need ACID compliance and need your db to be transactional then choosing Innodb is an obvious choice and you ought to make the necessary conversion including adding any FK constraints, etc if you need them.
2. If you are not disproportionately read-only heavy and are doing a mix of reads (not requiring full text indexing) and writes then we do recommend that you go with Innodb.
3. Most commonly we have observed that MyISAM tables would rather be converted to Innodb when you face frequent table lock escalations for long periods of time.
4. If a read is slow or hasn't completed and a read/write is waiting on the first read to finish then the MyISAM table referenced in the read is held in a locked state till the resultset is made available to the query. This also causes a rise in the load average on the server and slows your site down. During this time no reads or writes can complete ofcourse as MyISAM only has table-level locking.

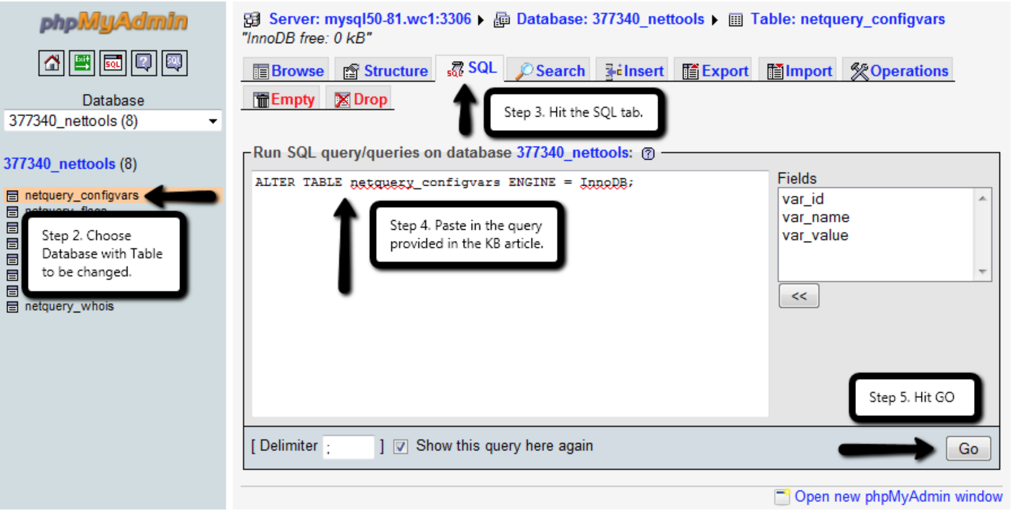
So to summarize, the queries that are victims of lock escalations under heavy but slow reads would do much better as a table converted to Innodb.

How to change your table Engine type from MyISAM to Innodb?

You do so by simply issuing the "ALTER TABLE" DDL statement:

```
ALTER TABLE <table-name> ENGINE=INNODB;
```

Below is a step by step process for altering a table in PHPMyAdmin:



phpMyAdmin
Database
377340_nettools (8)
377340_nettools (8)
netquery_configvars
netquery_whois
Server: mysql50-81.wc1:3306 ▸ Database: 377340_nettools ▸ Table: netquery_configvars
"InnoDB free: 0 kB"
Browse
Structure
SQL
Search
Insert
Export
Import
Operations
Empty
Drop
Run SQL query/queries on database 377340_nettools:
Fields
var_id
var_name
var_value
<<
[Delimiter ;] ☑ Show this query here again
Go
Open new phpMyAdmin window

```
ALTER TABLE netquery_configvars ENGINE = InnoDB;
```

Step 2. Choose Database with Table to be changed.

Step 3. Hit the SQL tab.

Step 4. Paste in the query provided in the KB article.

Step 5. Hit GO

1. Log into the PHPMyAdmin utility through your control panel. If you are unsure how, please see Working with a MySQL database for instruction on how to login to PHPMyAdmin.
2. Select the database which contains the **Table-Name**
3. Click on the SQL tab
4. Paste in the query provided above. Be sure to replace **table-name** with the correct name of your table
5. Click the **GO** button.

Note: A MyISAM table that is using FULL TEXT Indexing can not be converted to an Innodb Table Engine type.

Was this content helpful? Yes No

19 Comments Rackspace Knowledge Center 1 Login

Recommend 2 Share Sort by Newest

Join the discussion…

Jitendra · 13 days ago

Good one ...
Anybody can tell me
Is it possible to create material view based on select statement,
And when there is new data inserted or updated then
It wl refresh the material view in my sql

Thanks in advance

^ | ˅ · Reply · Share ›

Dhanu · 17 days ago

Very good information, thanks

^ | ˅ · Reply · Share ›

joscal · 2 months ago

Excellent info

^ | ˅ · Reply · Share ›

lal · 7 months ago

Very good, thanks for informative

1 ^ | ˅ · Reply · Share ›

Chris · 8 months ago

Very informative.

1 ^ | ˅ · Reply · Share ›

Timoo van Esch · 9 months ago

Yes, very helpful.
Working with large data blocks, we questioned whether to take innoDB or MyISAM. This summary of benefits ánd the fact that you use it as a default for Rackspace, made it very clear to us that innoDB is the way to go.

^ | ˅ · Reply · Share ›

itoctopus · a year ago

Joomla tables are created as MyISAM by default (for performance reasons).

It is important to note that 2 tables, in particular, must be InnoDB: The assets and the sessions table, since these tables tend to crash a lot.

1 ^ | ˅ · Reply · Share ›

pier28 · a year ago

The table is a little misleading on the MyISAM column about flushing changes to disk immediately. This is NOT true. MyISAM relies on the underlying operating system to do the flushes to disk. MyISAM issues a write request to the filesystem buffer cache, and then it is up to the operating system to decide when to flush the cache and actually write the data to disk. This is very dangerous for systems with high volumes of transactions if something goes wrong (power outage, disk failure, etc). InnoDB schedules the I/O on its own, and if properly configured with direct I/O, it will order the data and then instruct the operating system when to do the writes when it's ready and there's no delay from the operating system in performing that task.

3 ^ | ˅ · Reply · Share ›

spuy767 · 2 years ago

Having just migrated a large-ish, 173 tables, a few with millions of rows, database from MyISAM to InnoDB, it's important to note that It appears that full text indices Must be dropped and recreated after the conversion in order to function properly. I could find no indication of this in the MySQL documentation, but my full text indexing didn't work until I did that.

6 ^ | ˅ · Reply · Share ›

> **pier28** ➦ spuy767 · a year ago
>
> You should drop all your indices before a conversion from MyISAM to InnoDB anyway. The conversion will go faster, and there's a fast index creation option for InnoDB that will create the new indices pretty quickly. Also, do an optimize on each table afterwards, and if you're using XFS for your filesystem, run xfs_fsr to defrag the disk after optimizing.
>
> 2 ^ | ˅ · Reply · Share ›

spuy767 ➦ pier28 • a year ago

We run SSDs, so defrag is a No-No, but dropping the indices is a good tip to keep in mind.

2 ^ | ˅ • Reply • Share ›

carterson2 ➦ spuy767 • a year ago

soo, are you glad you switched???

1 ^ | ˅ • Reply • Share ›

spuy767 ➦ carterson2 • a year ago

Absolutely. There are a lot of writes that lock some of the larger tables with MyISAM. After the switch, the mean execution time for requests has decreased by about 40%. Memory utilization is down too, and we are in the financial industry, so transactions improve our bookkeeping.

1 ^ | ˅ • Reply • Share ›

rose.contreras • 2 years ago

Thank you for bringing the issue of InnoDB and Fulltext indices to our attention! We have revised the article accordingly.

2 ^ | ˅ • Reply • Share ›

BlueRaja • 2 years ago

InnoDB has supported Fulltext indices for nearly two years:
http://www.drdobbs.com/databas...

Stick with the times :)

2 ^ | ˅ • Reply • Share ›

Andrija • 2 years ago

Hi,

you have switched the MyISAM and InnoDB in terms of crash recovery options: InnoDB has automatic repair via log replay (not MyISAM), and
MyISAM has to do full table repair (not InnoDB)

Please correct this, as it is a very important thing.

Thanks, Andrija

4 ^ | ˅ • Reply • Share ›

jered.heeschen ➦ Andrija • 2 years ago

Thanks for catching that Andrija. We've swapped those entries in the table to their correct
positions.

3 ^ | ˅ • Reply • Share ›

David Beroff • 2 years ago

Speaking of formatting victims, the article itself may have accidentally fallen victim to some strangeness, as it
currently ends with, "Select the database which contains the". Thanks for your help, Jered!

1 ^ | ˅ • Reply • Share ›

jered.heeschen ➦ David Beroff • 2 years ago

Oof, you're right. I'll see if I can't restore the last part of the article. Thanks for letting us know,
David.

1 ^ | ˅ • Reply • Share ›

✉ Subscribe   Add Disqus to your site   Privacy

PRODUCTS
Public Cloud
Private Cloud
Hybrid Cloud
Managed Hosting
Email Hosting

SUPPORT
Support Home
Knowledge Center
Rackspace Community
API Documentation
Developer Center

CONTROL PANELS
MyRackspace Portal
Cloud Control Panel
Cloud Sites Control Panel
Rackspace Webmail Login
Email Admin Login

ABOUT RACKSPACE
Our Story
Case Studies
Events
Programs
Newsroom
The Rackspace Blog
Developer Blog
Contact Information
Legal
Careers