Todd's MySQL Blog

Tiny tidbits of trivia from Todd

Exam Cram: MySQL Architecture for Developers (Section 1)

10/17/2013 MySQL certification, MySQL 5.6

The first section in the exam topics for both the MySQL 5.6 Developer and DBA exam deals is titled *MySQL Architecture*, but each has a different emphasis. This blog will focus on those objectives listed for the Developer exam:

> - *Use MySQL client programs to interface with the MySQL Server interactively and in batch*
> - *Describe SQL Modes and their impact on behavior of MySQL*
> - *Identify characteristics which have session scope*

For those of us who have spent time pulling both DBA and developer duties, this section is likely to be relatively straight-forward – perhaps even easy. I'll walk through the exercises and documents I've used to review below to kick off my Exam Cram blog series.

Using MySQL Client Programs

My first step here is to review list of MySQL client programs (particularly mysql, mysqladmin, mysqldump). I should be able to associate programs with their general purpose:

- mysql – SQL shell supporting both interactive and batch access
- mysqladmin – Interface for performing some limited administrative functions
- mysqlcheck – Tool to perform table maintenance operations
- mysqldump – Program to produce logical backups of MySQL instances
- mysqlimport – Imports data from text files (does NOT restore logical backups)
- mysqlshow – Program which wraps SHOW commands (hardly ever use it)
- mysqlslap – Load-generation utility

My next step is to review the options for the mysql client, many of which apply to the other clients as well. Below I've listed those I think are most important with some notes:

- –comments – By default, comments are stripped by the mysql client before sending commands to the server. When running a script which includes comments you might want to

see (in general or slow query logs, PROCESSLIST, etc.), include this option.

- –compress – Useful for slow networks, this causes compression of communication packet data between client and server. This doesn't come for free; there's added overhead on both the server and client side to compress or decompress the data.
- –host – Which host to connect to. Note the special meaning for "localhost" described here. Understand how MySQL Server binds to specific network interfaces and the implications for clients attempting connections.
- All of the options related to option-file handling – Understand how to use options files and how to find options files being used.
- –port – Which port to connect to, when TCP/IP is used. Note that using –port doesn't explicitly trigger TCP/IP connections.
- –force – Make a script continue even when a SQL error is encountered.
- –execute – When specified, the mysql client connects, executes the statement(s) specified and disconnects. Useful for scripting.
- –password – The password, obviously – but make sure you understand that the –password option without any arguments will cause the client programs to prompt you for the password, and won't echo the password to the screen. Passwords can be specified as arguments (e.g., –password=mypass), but this is unsafe and is discouraged.
- –protocol – MySQL supports connections via TCP/IP and Unix sockets, as well as shared memory and named pipes on Windows.
- –socket – Know that this overloaded option represents not only the Unix socket path, but the named pipe for Windows(!).
- –ssl* – All of the SSL options are important to know. If you've never set up SSL connections on MySQL, do this in preparation for the exam.
- –user – Obviously the user, but understand how MySQL Server does authentication and the precedence given to host over user when authenticating users.
- –login-path – This is part of the option file handling section above, but I'm calling this out for special attention here. You'll want to understand this new 5.6 feature, how it assists secure handling of passwords, and it's limitations.

Make sure you know both the long and short variants of options (-p and –password are the same, -P and –port are the same, etc.).

I would also review the options for mysqldump. Understanding the various options which control output (do I want data or just the schema? do I want to load my 5.6 database dump to a 5.1 server? do I want to include replication binary log position information? do I want all databases or select content? events? triggers? stored procedures?) and consistency (do I need to lock tables? use a single transaction?) are important. As a developer, you may have had limited need to perform or restore logical backups. Do that in preparation for the exam.

Most of the use cases for mysqladmin can also be done via the mysql client as normal SQL commands; in fact, many mysqladmin operations map directly to SQL commands. mysqladmin is generally more useful for incorporating into scripts, and I find the following operations most useful to developers:

- ping – check that the server is responsive and you can authenticate
- shutdown – initiate a controlled shutdown
- processlist – show information from PROCESSLIST

The other operations are more typically used by DBA or operations staff, but be familiar that they exist.

Spending time looking at and using mysql_config_editor will serve candidates well.

SQL Modes

The SQL Modes are described here in the manual. Which are most important to know well?

- NO_ENGINE_SUBSTITUTION – This is the server default in MySQL 5.6, and produces errors when the specified storage engine is not available. Running without this – as was the default in earlier versions of MySQL Server – allows the Server to replace an explicitly-defined storage engine in a CREATE TABLE command with the default storage engine if the requested engine is unavailable.
- STRICT_TRANS_TABLES – When you care about data integrity and want MySQL to reject data manipulation operations which would lose data (for example, trying to insert 1.2345 int a column defined as DECIMAL(5,2)) you want to use this. Note that this only affects transactional storage engines – if you want this to apply to all tables regardless of storage engine, use STRICT_ALL_TABLES instead. This isn't a server default in MySQL 5.6, but it is defined as a default in the configuration files which ship with the server. That has the effect that new installations are likely to see this mode enabled by default.
- NO_AUTO_CREATE_USER – This matters if you are creating user accounts in your application at all; without it, GRANT commands can create new user account without a password if the specified user does not already exist.
- NO_ZERO_IN_DATE – A compliment to the strict modes discussed above, prevents dates with non-zero year components having zeros for the month or day component.
- IGNORE_SPACE – This allows whitespace between function names and parentheses, but causes expansion of reserved words.
- ERROR_FOR_DIVISION_BY_ZERO – Trigger an error instead of returning NULL for INSERT or UPDATE statements which do division by zero.
- ANSI and TRADITIONAL – These are two most commonly-used combination modes – they are simply a collection of SQL modes defined elsewhere bundled together. I won't worry about the other combination modes, but I will make a point to remember which individual SQL modes are contained in these two.

You should understand how you can set the SQL mode, and how the server configuration can be overridden with client (or driver) options or setting dynamically. Know how these SQL modes map to connector behavior (see useJdbcCompliantTruncation for Connector/J here).

If you haven't played with SQL modes before, take some time to experiment. Note that SQL mode is preserved at definition time for certain schema objects like stored routines.

Session scope

Too few developers understand which characteristics are tied to sessions. Most significant:

- Prepared statements
- Temporary tables
- User variables

- Session variables
- Transactions (purposefully excluding XA here)

This has a number of implications for developers – one of the most commonly-held misconceptions from application developers is that they can use driver auto-reconnect features to recover from any communication failure without having to be worried about connection state. This is patently untrue – no connectors can recover all of the above state which further work may depend upon. As an application developer, you ***need*** to know that auto-reconnect won't magically eliminate your need to recover from potentially problematic states.

On the opposite side, this has implications for resource management in persistent connection deployment environments. If you use connection pooling, and each time the connection is reused by a different application thread, it sets new user variables and never resets them, you could be wasting memory. Same with server-side prepared statements or temporary tables (although that's more likely to be disk resources instead of memory).

There's no section specifically dealing with temporary tables, so I'll mention it here: Know that temporary tables with the same name as an existing table can be created, and they obscure the non-temporary table for the connection in which they were created until they are dropped. See the Temporary Tables section of this manual page.

Beyond the list above, candidates should understand that MySQL Server allocates a thread per connection. MySQL provides an API allowing development of plugins that behave differently (one such is offered as part of the MySQL Enterprise subscription for additional scalability). Documentation of thread handling can be found here.

Practice Questions

I've compiled eight practice questions as I found/remembered interesting things in my review. You can download them here. Be aware, I've not seen actual exam questions, so I won't claim these are representative. There are two rules for exam question writing I've intentionally violated in authoring these practice questions – I've written open-ended (pick any number of correct answers from 5 options) and trick (to make points in the answer explanation) questions. Hopefully the result is that these questions are tougher to get right than what we'll experience on the exam.

If you would like to contribute your own practice questions, I'm happy to add them to the collection and give you credit – maybe we can produce a community-sourced practice exam of sorts. You can use the following contact form to submit questions and answers (that way they don't show up in comments as spoilers):

Name (required)

Email (required)

Website

Practice Question (required)

Practice Question Answer

Submit »

I hope that helps preparations – next stop, *General MySQL Syntax*.

Share this:

6 thoughts on "Exam Cram: MySQL Architecture for Developers (Section 1)"

vali10/18/2013 at 4:59 AM

Thanks!

★ Todd Farmer10/18/2013 at 7:09 AM

Glad this is useful to you, vali!

Carlos10/22/2013 at 2:33 PM

Thanks a lot :). keep it up

★ Todd Farmer10/22/2013 at 9:42 PM

Sure thing, Carlos! I'm working on the next section now, and hope to have it ready in the next couple of days. 🙂

Nik11/01/2013 at 9:49 AM

Hi Todd,
tried your first question from the download list. And funny thing: I understand that if you put lines one after another you are right, but if you trying to commit them all together(in Workbench) I didn't have any rows in the table as a result of error on "CALL test.p(-1);"

★ Todd Farmer11/04/2013 at 8:20 AM

Hi Nik,

Thanks for your comment! Indeed, Workbench (by default) cancels execution of scripts on the first observed error. This is configurable behavior, though: In WB 6.0, the seventh button across the top of the SQL Editor pane controls whether script execution is interrupted or continues on error. I think this button could benefit from some more distinguishing iconography, so I opened Bug#70809.