AnswerHub

Social Q&A for the Enterprise

½ of the Top 10 StackExchange 1.0 Sites Now Run on **AnswerHub**

Discover Why Now!

DZone Refcardz

CONTENTS INCLUDE:
❯ MySQL 5.5
❯ Configuration
❯ Storage Engines
❯ Data Types
❯ Administration
❯ Hot Tips… and More!

Updated for MySQL 5.5

Essential Admin for MySQL 5.5

By Jason Gilmore

ABOUT THIS REFCARD

MySQL is the world's most popular open-source database, sporting a barrier of entry low enough to attract novice developers yet powerful enough for some of the world's most popular websites, among them Yahoo!, Walmart, Wikipedia, and Craigslist. Now a member of the Oracle family of products, MySQL's future is indeed bright.

Updated to reflect the MySQL 5.5 release, this Refcard will help you quickly navigate some of MySQL's most popular features. Covering topics such as configuration, administration software, backup procedures, SQL features, and user management, this card will serve as a handy desk reference for countless projects to come.

MYSQL 5.5

With the late 2010 release of MySQL 5.5 comes a host of exciting new features largely revolving around the InnoDB storage engine, including its establishment as the default MySQL storage engine on all platforms. Other notable features include:

- The incorporation of InnoDB 1.1 gives users the advantage of enhanced stability and performance integrated into this latest InnoDB release.
- A number of scalability and performance improvements have been incorporated into MySQL's Win32 and Win64 versions.
- The new performance_schema database gives administrators the opportunity to monitor the execution of the MySQL server at a very low level.

CONFIGURATION

MySQL supports over 320 configuration parameters and is capable of controlling behavior regarding memory, logging, error reporting, and much more. While it's possible to tweak these parameters by passing them as flags when starting the MySQL server, you will want to ensure they are always set at server startup, which you can do by adding them to the my.cnf file.

The my.cnf File
The my.cnf file's range of impact on the MySQL server is location dependent. The most commonly used locations are listed here:

File/Option	Description
/etc/my.cnf (C:\my.cnf, WINDIR\my.ini, or INSTALLDIR\my.ini on Windows)	All MySQL database servers installed on this server will first refer to this location.
--defaults-extra-file=NAME	Passing this flag along when starting MySQL will cause the server instance to examine any parameters found within the file and corresponding path identified by NAME.
~/.my.cnf	User-specific scope. This file is located in the user's home directory. This option is not available on the Windows platform.

Hot Tip
On Unix/Linux/OS X, this option file uses the .cnf extension. However, on Windows, both the .cnf and .ini extensions are supported.

my.cnf File Syntax
The my.cnf file is a text file broken into several sections. Each section defines the context of the parameters defined within, the context being specific to a particular MySQL client (see the later section "MySQL's Many Clients") . For example:

Hot Tip
MySQL is bundled with several my.cnf templates, each geared towards a specific purpose and resource availability. Users of the source distribution can find these templates in the support-files directory. Users of the binary distribution can find these templates in the installation directory.

Viewing Configuration Parameters
You can view MySQL's configuration parameters and their current values using one of the following commands:

From the mysqladmin client:

```
%>mysqladmin -u root -p variables
```

From inside the mysql client:

```
mysql>SHOW VARIABLES;
```

You can find a specific parameter using the LIKE operator

```
mysql> show variables like "%key%";
+--------------------------------+------------+
| Variable_name                  | Value      |
+--------------------------------+------------+
| delay_key_write                | ON         |
| foreign_key_checks             | ON         |
| have_rtree_keys                | YES        |
| key_buffer_size                | 26214400   |
| key_cache_age_threshold        | 300        |
| key_cache_block_size           | 1024       |
| key_cache_division_limit       | 100        |
| max_seeks_for_key              | 4294967295 |
| ssl_key                        |            |
+--------------------------------+------------+
9 rows in set (0.02 sec)
```
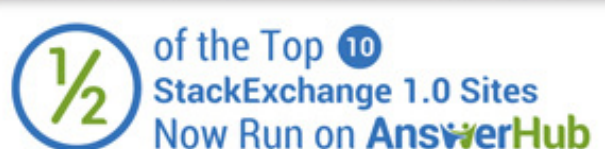
STORAGE ENGINES

MySQL offers a number of different solutions for storing and managing data in the most efficient way.

Storage Engine	Description
ARCHIVE	The ARCHIVE engine is optimized for managing large amounts of data designated as stored for archived purposes. Data stored using the ARCHIVE engine can only be inserted and selected and cannot be deleted or modified.
BLACKHOLE	The BLACKHOLE storage engine accepts inserted data without error but does not store it. Instead, it deletes it upon acceptance. While seemingly useless, BLACKHOLE can actually serve several practical roles, ranging from troubleshooting data replication processes to assisting in the identification of bottlenecks (due to the ability to use BLACKHOLE to remove the storage engine from the bottleneck candidates).
CSV	Comma-separated values (CSV) format is a common storage solution supported by many applications. MySQL's CSV storage engine manages data in this format, the data files of which can subsequently be read from and written to by applications such as Microsoft Excel.
EXAMPLE	EXAMPLE is a featureless storage engine with the sole purpose of providing a skeleton for writing your own own storage engines. It is incapable of storing data.
FEDERATED	Introduced in MySQL 5.0, the FEDERATED storage engine can pool remote MySQL databases together under the guise of a single logical database by creating pointers to these remote tables.
InnoDB	MySQL's most popular transactional storage solution, InnoDB offers complete commit, rollback, and crash recovery features alongside attractive performance capabilities. InnoDB has long served as MySQL's default storage engine on the Windows platform. It is the default storage engine on all platforms for MySQL 5.5.
MEMORY	The MEMORY storage engine stores data within system memory (RAM), resulting in volatile, although extremely fast, data access.
MERGE	The MERGE storage engine is useful for accessing a group of identical MyISAM tables as if the data resided within a single table structure. Such a configuration might be useful when accessing large amounts of sales data, which has been separately stored by month according to an aptly named table.
MyISAM	Until MySQL 5.5.5, the MyISAM had long been MySQL's default storage engine. Although incapable of supporting transactions, MyISAM is optimized for high traffic environments and is very simple to manage.

DATA TYPES

MySQL supports a rich set of data types capable of representing nearly every conceivable data format, ranging across dates and times, currency, strings, integers, and floats. This section defines each type and its respective range.

Date and Time Types

Type	Description
DATE	The ARCHIVE engine is optimized for managing large amounts of data designated as stored for archived purposes. Data stored using the ARCHIVE engine can only be inserted and selected and cannot be deleted or modified.
DATETIME	The DATETIME type represents values containing both a date and a corresponding time in the format 'YYYY-MM-DD HH:MM:SS'. It has a range of '1000-01-01 00:00:00' to '9999-12-31 23:59:59'.
TIME	The TIME type represents temporal values in the format 'HH:MM:SS', ranging from '-838:59-59' to '838:59:59'.
TIMESTAMP	Like DATETIME, the TIMESTAMP type represents values containing both a date and time and sports a format identical to DATETIME. Its range is '1970-01-01 00:00:01' UTC to '2038-01-09 03:14:07' UTC. The TIMESTAMP differs from other data types in that it can be automatically assigned the current date/time and automatically updated at INSERT and UPDATE time.
YEAR	The YEAR type represents years and supports a two- ('YY') and four-digit format ('YYYY'). The two-digit format supports a range of 70 (1970) to 69 (2069). The four-digit format supports a range of 1901 to 2155.

Numeric Types

Type	Description
BIGINT	The BIGINT data type supports integer values ranging between -9,223,372,036,854,775,808 and 9,223,372,036,854,775,807.
BIT	The BIT data type supports binary values ranging between 1 and 64 bits.
DECIMAL	The DECIMAL type stores exact numeric values and should be used when it is crucial for the data to be stored precisely as provided (currency for instance).
FLOAT	The FLOAT data type stores approximate numeric values. For instance, defining a column as FLOAT(5,3) will store 12.4785 as 12.479, because the defined precision is 3.
INT	The INT data type supports integer values ranging between -2,147,483,648 and 2,147,483,647.
MEDIUMINT	The MEDIUMINT data type supports integer values ranging between -8,388,608 and 8,388,607.
SMALLINT	The SMALLINT data type supports integer values ranging between 32,768 and 32,767.
TINYINT	The TINYINT data type supports integer values ranging between -128 and 127..

String Types

Type	Description
BINARY	The BINARY type stores up to 255 bytes and operates identically to CHAR except that it is used for binary strings.
BLOB/ LONGBLOB/	The BLOB, MEDIUMBLOB, TINYBLOB, and LONGBLOB types are used to store data such as images or binary files and store up to 65,545.
CHAR	The CHAR type stores between 0 and 255 characters. Any column defined as a CHAR will consume all of the allotted space regardless of the stored string size. For instance, any CHAR column defined as CHAR(25) will require the same amount of space (is required to store 25 characters) no matter the size of the stored string.
ENUM	The ENUM type restricts the stored value to one of several predefined strings. Up to 65,535 elements can be predefined. Allowable values include NULL.
SET	A SET type operates like an ENUM, although its predefined number of elements tops out at 64. Further, a SET can store zero, one, or multiple values.
TEXT/ LONGTEXT/ MEDIUMTEXT/ TINYTEXT	The TEXT, LONGTEXT, MEDIUM, and TINYTEXT types store up to 65,534, 4,294,967,295, 16,777,215, and 255 characters, respectively.
VARBINARY	The VARBINARY type stores up to 65,535 bytes and operates identically to VARCHAR except that it's used for binary strings.
VARCHAR	The VARCHAR type stores up to 65,535 characters. Unlike CHAR, each VARCHAR instance requires only the space required to store the provided string, plus one or two additional bytes depending on the string length.

POPULAR ADMINISTRATION SOFTWARE

Web frameworks help you to embrace best practices, simultaneously decreasing errors and eliminating redundant code. If you haven't yet settled upon a framework, consider checking out the following popular solutions.

phpMyAdmin	http://www.phpmyadmin.net/
MySQL Workbench	http://www.mysql.com/products/workbench/
SQLYog	http://www.webyog.com/

MYSQL'S MANY CLIENTS

MySQL is bundled with quite a few clients capable of doing everything from performing backups, managing the MySQL server, converting table formats, and stress-testing the database. This section introduces the most commonly used clients.

Storage Engine	Description
my_print_defaults	Outputs the options defined in the my.cnf files.
myisam_ftdump	Displays information regarding any defined FULLTEXT indexes found in MyISAM-defined tables.
myisamchk	Aids in the review, repair, and optimization of MyISAM-defined tables.
myisamlog	Displays the contents of MyISAM log files.
myisampack	Compresses MyISAM tables, greatly enhancing the read performance.
mysql	The MySQL client used to manage users, databases, tables, and data. It also tweaks MySQL's performance and behavior.
mysql_config	Displays information regarding options you may find useful when compiling MySQL.
mysql_convert_table_format	Converts tables from one storage engine to another.
mysql_fix_extensions	Converts MyISAM table extensions to a standard format, which is useful when migrating MyISAM files from one operating system to another.
mysql_setpermission	A wrapper for setting MySQL user privileges.
mysqlaccess	Aids in the review of user privileges.
mysqladmin	Performs a wide array of administrative tasks pertinent to server operation.
mysqlbinlog	Used for examining the contents of MySQL's binary log.
mysqlcheck	A unified wrapper for the SQL statements CHECK TABLE, REPAIR TABLE, ANALYZE TABLE, and OPTIMIZE TABLE.
mysqldump	Facilitates in-database backup creation. See 'Performing Backups' for more information.
mysqlhotcopy	Facilitates in-database backup creation. Keep in mind that mysqlhotcopy will not work with InnoDB storage engine, which is the default storage engine as of MySQL 5.5. See 'Performing Backups' for more information.
mysqlimport	A wrapper for the LOAD DATA INFILE SQL statement.
mysqlshow	A wrapper useful for learning more about database schemas.
mysqlslap	Tests MySQL's performance by placing an artificial load on the server and reporting the results.
perror	The perror client helps to clarify the often cryptic system error numbers often returned alongside MySQL errors.

KEY ADMINISTRATION TASKS

Logging into the MySQL server
To log in to the MySQL server using the mysql client, you'll typically provide your MySQL username and password:

```
%>mysql –u USERNAME –p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.1.22-rc-community MySQL Community Server (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

Once logged in, you can select a database or begin carrying out other administrative tasks. To save some time you can pass the desired database along on the command line when logging in:

```
%>mysql –u USERNAME –p DATABASE
```

If you're connecting to a remote database, pass the hostname or IP address along using the –h option:

```
%>mysql –h HOSTNAME –u USERNAME –p DATABASE
```

To log out of the MySQL server, use quit or the \q flag:

```
mysql>quit
Bye
%>
```

Modifying the mysql Prompt
MySQL's default prompt clearly indicates that you are currently logged into MySQL rather than into an operating system shell. However, like most shells, you can modify MySQL's prompt to your liking. For instance, when logged into the mysql client, execute the following command to change your prompt to mysql (user@host)>:

```
mysql>prompt mysql (\U)>
mysql (root@localhost)>
```

Common mysql prompt sequences

\c	A counter that tracks the total number of issued session commands
\d	The current database
\D	The current date
\h	The server host
\u	Your username
\U	Your username@hostname

Databases

Creating a Database	Once logged into the MySQL server, you can create a new database using the CREATE DATABASE command: mysql>CREATE DATABASE dzone;	You can also create a new database without logging into the server using the mysqladmin client: %>mysqladmin -u root -p create dzone
Switching to a Database	You can begin using a specific database by specifying it on the command-line when logging into the MySQL server (see 'Logging into the MySQL server') or by using the USE command: %>mysqladmin -u root -p create dzone	
Deleting a Database	To delete a database, use the DROP DATABASE command: mysql>DROP DATABASE dzone;	

Tables

Creating a Table	To create a table, pass the desired table name to the CREATE TABLE structure along with any column definitions:
	
	CREATE TABLE table_name (
	  column1 definition,
	  column2 definition,
	  ...
	  columnN definition
	);
		For instance:
		
		CREATE TABLE authors (
		 id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
		 name VARCHAR(255) NOT NULL,
		 email VARCHAR(255) NOT NULL
		);

Displaying a Table Structure	mysql>DESCRIBE table_name;	
Listing All Tables	To view a list of all tables in a database, execute the SHOW TABLES command: mysql>SHOW TABLES;	To view a list of tables residing in a database other than the one you're currently in, use: mysql>SHOW TABLES FROM database_name;

Altering a Table Structure	You can add, delete, and modify table columns using the ALTER TABLE command.
	
	- To add a column to an existing table:
	mysql>ALTER TABLE table_name ADD COLUMN column_name
	  ->column_type column_type_attributes;
	
	- To add a column to the end of the previously created authors table:
	mysql>ALTER TABLE authors ADD COLUMN telephone
	  ->VARCHAR(20) NOT NULL;
	
	- To add a column at a specific location, use the AFTER clause:
	mysql>ALTER TABLE authors ADD COLUMN telephone
	  ->VARCHAR(20) NOT NULL AFTER name;
	
	- To delete a column:
	mysql>ALTER TABLE table_name DROP COLUMN column_name;
	
	- To modify an existing column:
	mysql>ALTER TABLE table_name CHANGE COLUMN column_name
	  ->column_name column_type column_type_attributes;

Deleting a Table	To delete a table, use the DROP TABLE command: mysql>DROP TABLE table_name;
Renaming a Table	To rename a table, use the ALTER TABLE command with the RENAME clause: mysql>ALTER TABLE table_name RENAME new_table_name;

MANAGING USERS

MySQL offers a powerful security model capable of controlling practically every conceivable action. These actions range from which databases, tables, and even columns a user is allowed to access to which commands a user can execute and even how many queries a user can execute in an hour. This model works in a two-step sequence:

Step 1. Authentication	The user'is provided host, username, and password are examined. If a match is made within MySQL's privilege tables, the user is authorized. Otherwise, the user's connection attempt is denied.
Step 2. Authorization	Once authenticated, the user's submitted command is examined and compared against the user's defined privileges, also found in MySQL's privilege tables. If the user has sufficient privileges, the command is executed, otherwise it is denied.

Although covering the nuances surrounding MySQL's privilege tables is beyond the scope of this document, the remainder of this section should give you ample reminders regarding commonplace tasks. You are, however, encouraged to carefully review the privilege table documentation found in the MySQL manual (http://dev.mysql.com/doc/), as it's easy to make a mistake when using this powerful feature.

> **Hot Tip**
> MySQL 5.5.7 added pluggable authentication, allowing administrators to create and install plugins, which can authenticate users using a wide variety of new methods and repositories.

Creating a New User Account

New user accounts can be created in a variety of ways. However, the easiest and most error-proof way is through the GRANT command. The general structure looks like this:

mysql>GRANT privilege1, privilege2, privilegeN ON database_name.*
  ->TO 'username'@'host' IDENTIFIED BY 'password';

The following command will create a new user named jason, granting SELECT, INSERT, and UPDATE privileges to all tables found in the dzone database when connecting from 192.168.1.145 and when providing the password secret:

mysql>GRANT SELECT, INSERT, UPDATE ON dzone.*
     ->TO 'jason'@'192.168.1.145' IDENTIFIED BY 'secret';

Although hardly recommended due to security purposes, it is possible to grant a user all privileges on all databases. Use *.* in place of the database name.

> **Hot Tip**
> Be sure to specify the database name and include the IDENTIFIED BY clause when creating new users! Neglecting to include this information will grant this user global privileges and create a user lacking a password, respectively!

Deleting a User Account

You can delete a user using two methods. The method you choose is dependent on the context of 'delete'. To remove all user privileges but not entirely remove the user from the system (for instance, if you needed to temporarily disable account access), use the REVOKE command:

mysql>REVOKE ALL PRIVILEGES FROM 'jason'@'192.168.1.145';

To completely remove a user from the system, revoking all privileges and erasing the user account, use the DROP USER command:

mysql>DROP USER 'jason'@'192.168.1.145';

Changing a Password

To change a user's password, use the SET PASSWORD command. For instance, to change the password of the previously created jason account:

mysql>SET PASSWORD FOR 'jason'@'192.168.1.145' = PASSWORD('supersecret');

Granting Privileges

To grant additional privileges, use the GRANT command in precisely the same manner as it was used to create a user; MySQL will recognize the user's existence and just modify the user's privileges accordingly. For instance, to add the DELETE privilege to the previously created user jason@192.168.1.145:

mysql>GRANT DELETE ON dzone.* TO 'jason'@'192.168.1.145';

Revoking Privileges

To revoke privileges, use the REVOKE command. For instance, to remove the DELETE and UPDATE privileges from the previously created user jason@192.168.1.145:

mysql>REVOKE DELETE, UPDATE FROM 'jason'@'192.168.1.145';

Granting Table- and Column-specific Privileges

MySQL administrators can also control user privileges at the table and column level using the GRANT and REVOKE commands. For instance, to grant user jason@192.168.1.145 INSERT and SELECT privileges on the dzone database's timesheets table:

mysql>GRANT INSERT ON dzone.timesheets TO 'jason'@'192.168.1.145';

However, particularly security-minded administrators can prevent users from potentially modifying or selecting any column but the hours column found in the timesheets table:

mysql>GRANT INSERT (hours), SELECT (hours) ON dzone.timesheets
  ->TO 'jason'@'192.168.1.145';

Renaming Users

To rename an existing user, use the RENAME USER command:

mysql>RENAME USER 'jason'@'192.168.1.145' TO 'wjg'@'192.168.1.145';

KEY SQL TASKS

While executing standard SQL statements is likely old hat for most users, it may be more difficult to recall the syntax pertinent to some of MySQL's relatively new SQL features, namely Stored Routines, Views, and Triggers. This section serves as a refresher for these features' basic syntax.

Stored Routines
MySQL collectively refers to stored procedures and stored functions as stored routines. Stored procedures are executed using the CALL statement and can return values as MySQL variables, whereas stored functions can be called directly from within a MySQL like any other standard MySQL function.

In this section, a brief refresher is provided regarding managing what is arguably the more useful of the two, namely stored functions.

Creating a Stored Function

A stored function is created using the CREATE FUNCTION command. A simple example follows:

```
mysql>DELIMITER $$
mysql>CREATE FUNCTION calculate_bonus
   ->(employee_id INTEGER) RETURNS DECIMAL(5,2)
   ->BEGIN
   ->DECLARE article_count INTEGER;
   ->DECLARE bonus DECIMAL(10,2);
   ->SELECT count(id) AS article_count FROM articles
   ->WHERE author_id = employee_id;
   ->SET bonus = article_count * 10;
   ->RETURN bonus;
   ->END;
   ->$$
mysql>DELIMITER ;
```

Once created, you can call calculate_bonus() from within a query:

```
mysql>SELECT name, phone, calculate_bonus(id) FROM authors;
```

Hot Tip
Stored procedures and functions support complex logical syntax features, including conditionals and looping statements.

Altering a Stored Function

To modify an existing function, use the ALTER FUNCTION command:

```
mysql>DELIMITER $$
mysql>ALTER FUNCTION calculate_bonus
   ->MODIFIED FUNCTION BODY...
   ->$$
mysql>DELIMITER $$
```

Deleting a Stored Function

To delete a stored function, use the DROP FUNCTION command:

```
mysql>DROP FUNCTION calculate_bonus;
```

Views

Views can greatly simplify the execution and management of an otherwise complex query by assigning an alias to it, allowing the developers to execute the query by its alias rather than repeatedly entering the query in its entirety.

Creating a View

Views are created using the CREATE VIEW command. For instance:

```
mysql>CREATE VIEW author_view AS
   ->SELECT name, e-mail, phone FROM authors ORDER BY email ASC;
```

You can then execute the view like so:

```
mysql>SELECT * FROM author_view;
```

Passing Query Parameters

You can pass parameters to a view like you would any typical query. For instance, to retrieve only information about the author with the e-mail address jason@example.com:

```
mysql>SELECT * FROM author_view WHERE email = 'jason@example.com';
```

Viewing a View

You can examine the columns retrieved by the view using the DESCRIBE statement:

```
mysql>DESCRIBE author_view;
```

To view the view syntax, use SHOW CREATE VIEW:

```
mysql>SHOW CREATE VIEW author_view;
```

Modifying a View

To modify a view, use the ALTER VIEW statement:

```
mysql>ALTER VIEW author_view AS
   ->SELECT name, phone FROM authors ORDER BY phone;
```

Deleting a View

To delete a view, use the DROP VIEW statement:

```
mysql>DROP VIEW author_view;
```

Triggers

Triggers are automatically activated in accordance with a specific table-related event. They're particularly useful for automating table updates which should occur when another table is modified in some way.

Creating a Trigger

To create a trigger, use the CREATE TRIGGER command, passing the trigger actions into the command body. For instance, the following trigger will increment category's article counter each time a new article of that specific category is added to the database:

```
mysql>DELIMITER $$
mysql>CREATE TRIGGER article_counter
   ->AFTER INSERT ON articles
   ->FOR EACH ROW BEGIN
   ->UPDATE categories SET counter = counter + 1 WHERE id = NEW.category_id;
   ->END;
   ->$$
mysql>DELIMITER ;
```

Modifying a Trigger

You currently cannot modify an existing trigger from within the mysql client. Instead, you should delete the existing trigger and create it anew with the desired changes incorporated.

Deleting a Trigger

To delete a trigger, execute the DROP TRIGGER command:

```
mysql>DROP TRIGGER pay_author;
```

PERFORMING BACKUPS

Performing regular backups is an essential part of even the smallest database project. Fortunately, MySQL makes this very easy by offering several backup solutions.

Copying Files

If your tables use the MyISAM storage engine, you can back up the database simply by copying the files used to store the tables and data. To do so consistently, you'll either want to stop the MySQL server before copying the files or first execute the LOCK TABLES command (only a read lock is required), followed by FLUSH TABLES. Once executed, copy the files; and when the copy is complete, execute UNLOCK TABLES.

Creating Delimited Backups

To back up the table data in delimited format, use the SELECT INTO OUTFILE command. For instance, to back up the authors table used in previous examples, execute:

```
mysql>SELECT * INTO OUTFILE 'authors090308.sql' FROM authors
```

Using mysqldump

The mysqldump client is particularly convenient because it can backup databases of all storage engine types, InnoDB included, not to mention that it automatically takes care of important details such as locking the tables during the backup.

The **mysqldump** client supports an enormous number of options, and it is recommended you take some time to review them in the MySQL manual. However, this section will give you enough to at least remind you of what's required to perform a variety of different backups.

Backing up a Specific Database

To back up a single database, just pass the database name to the mysqldump client, piping the output to a text file:

```
%>mysqldump [options] database_name > backup0903.sql
```

Of course, you'll require proper permissions to execute mysqldump in conjunction with a specific database (namely the SELECT and LOCK privileges). Therefore, you'll typically also need to pass along your username and password. In this case, this command typically looks similar to:

```
%>mysqldump -u root -p database_name > backup0903.sql
```

Backing Up Specific Tables

To back up specific tables, you'll need to identify the database, followed by each specific table name you'd like to back up:

```
%>mysqldump [options] database_name table_name [table_name2...] > backupfile.sql
```

Backing Up All Databases

To back up all databases, pass the --all-databases option:

```
%>mysqldump [options] --all-databases > backupfile.sql
```

Hot Tip

MySQL's replication features make it possible to maintain a consistently synchronized version of the live database. Replication is out of the scope of this reference card, but be sure to visit the MySQL documentation (http://dev.mysql.com/doc/) if replication is more suitable to your needs.

Using mysqlhotcopy

If all of your backup tables use the MyISAM storage engine and you're able to log into the server where the tables are stored, the **mysqlhotcopy** might be the ideal solution due to its speed advantages.

To back up the dzone database to a directory located at /home/jason/backups using mysqlhotcopy, execute:

```
%>mysqlhotcopy -u root -p dzone /home/jason/backups
```

To copy multiple databases, just string each database name together:

```
%>mysqlhotcopy -u root -p dzone wjgilmore /home/jason/backups
```

Like **mysqldump**, **mysqlhotcopy** offers an enormous number of options, so be sure to review the MySQL manual (http://dev.mysql.com/doc/) to learn all that's available.

MySQL 5.6 Features

At the time this refcard was revised, the MySQL team was hard at work on MySQL 5.6. Notable 5.6 features include performance and support enhancements to the InnoDB storage engine (version 1.2 is supported in MySQL 5.6), which became the default storage engine as of MySQL 5.5.5. Notably, a wealth of new information regarding InnoDB optimization and performance statistics is made available via both the INFORMATION_ SCHEMA and server logs.

Another interesting addition is a memcached API, which can directly access the native InnoDB API. This opens up the possibility of using high-performance NoSQL access methods for simple queries, bypassing the intensive overhead otherwise required for tasks such as query parsing. If you're interested in testing out version 5.6, head over to http://labs.mysql. com/ and download the latest version.

ABOUT THE AUTHOR

Jason Gilmore is founder of W.J. Gilmore, LLC, a Columbus, Ohio-based publishing, training, and consulting firm. Jason is a prolific contributor to a number of leading publications such as Developer. com, PHPBuilder, and TechTarget, with more than 200 articles to his credit. He's cofounder of the CodeMash conference (http://www.codemash.org/), a non-profit organization charged with organizing the annual namesake event.

RECOMMENDED BOOK
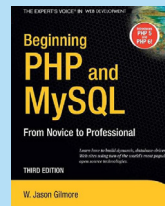
Beginning PHP and MySQL, Fourth Edition is the definitive book on the PHP language and MySQL database. Essentially three books in one, this book provides readers with comprehensivge introductions of both technologies in addition to in-depth instruction regarding using these two powerful technologies in unison to build dynamic web sites. See http://www. wjgilmore.com for more information!

BUY HERE

Browse our collection of over 150 Free Cheat Sheets

Android Application Development
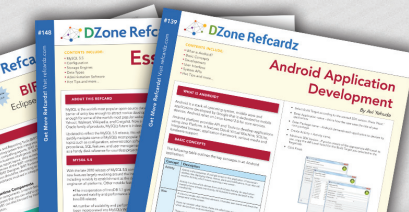
Free PDF

Upcoming Refcardz

HTTP
PHP 5.4
Modularity Patterns
MongoDB

DZone

DZone communities deliver over 6 million pages each month to more than 3.3 million software developers, architects and decision makers. DZone offers something for everyone, including news, tutorials, cheat sheets, blogs, feature articles, source code and more. **'"DZone is a developer's dream",'** says PC Magazine.

DZone, Inc.
150 Preston Executive Dr.
Suite 201
Cary, NC 27513

888.678.0399
919.678.0300

Refcardz Feedback Welcome
refcardz@dzone.com

Sponsorship Opportunities
sales@dzone.com

Version 1.0