



## Quản trị CSDL MySQL

1. Tổng quan MySQL .....	2
2. Cài đặt và cấu hình MySQL.....	5
3. Quản trị căn bản MySQL.....	7
4. Phân quyền MySQL.....	9
5. Phục hồi mật khẩu MySQL root .....	13
6. Câu lệnh SQL.....	13
7. Quản lý cấu hình log .....	18
8. Quản trị MySQL process .....	20
9. Backup .....	21
10. Duplicate .....	23
11. Thực thi SQL từ Bash .....	23
12. Tuning và Troubleshooting.....	26
13. MariaDB .....	28



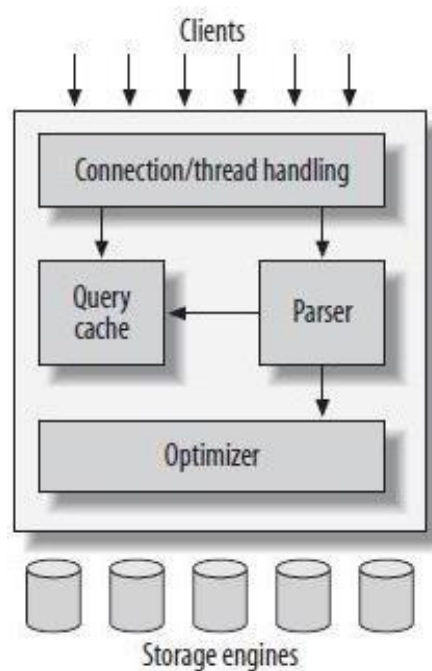
## 1. Tổng quan MySQL

Hầu hết các phần mềm quản lý ứng dụng hiện nay đều thao tác trên CSDL nào đó, do đó việc làm quen với CSDL là một vấn đề thiết yếu và tối quan trọng. Có rất nhiều hệ quản trị CSDL hiện nay, nhưng hầu hết các hệ quản trị CSDL này đều tuân theo chuẩn SQL92 do tổ chức ANSI đề ra và MySQL cũng là một trong các hệ quản trị CSDL đó.

MySQL là Hệ quản trị CSDL gọn nhẹ nhưng hỗ trợ đầy đủ tính năng và có phiên bản miễn phí. MySQL có hai phiên bản:

- MySQL Server Community - phiên bản miễn phí cung cấp đầy đủ tính năng nhưng không được support từ MySQL
- MySQL Server Enterprise - phiên bản thương mại và được sự hỗ trợ từ MySQL. Trong phiên bản này có kèm theo những tool về quản lý MySQL thông qua web, đo hiệu suất của hệ thống...

MySQL hiện hỗ trợ người dùng những công cụ cần thiết cho việc quản lý và phát triển, là một trong các hệ thống CSDL được người dùng trên khắp cộng đồng mã nguồn mở ưa chuộng trong việc chọn lựa và phát triển ứng dụng. Với công cụ MySQL Administrator đã mang đến cho người dùng những thuận lợi trong việc backup và restore dữ liệu, cũng như lên kế hoạch, thay đổi thông số đầu vào và tối ưu hoá các thông số cần thiết nhằm giúp cho hệ quản trị CSDL này vận hành tốt hơn. Bên cạnh đó, MySQL Query Browser cung cấp cho người dùng những tính năng liên quan đến việc quản trị và phát triển sản phẩm của mình. Khai thác những công cụ này, người dùng sẽ tiết kiệm thời gian đáng kể cho việc phát triển cũng như quản trị hệ thống.



Kiến trúc logic CSDL MySQL



## Storage Engines

Điểm khác nhau lớn nhất của MySQL so với các hệ quản trị CSDL khác là MySQL hỗ trợ nhiều các kỹ thuật lưu trữ dữ liệu theo nhiều cách khác nhau, mỗi kiểu lưu trữ dữ liệu được gọi là **Storage Engines**. Ở phiên bản 5.6 thì MySQL đang hỗ trợ hơn mười kỹ thuật lưu trữ khác nhau như *MyISAM, InnoDB, Memory, Merge, CSV, InfoBright...*

MySQL lưu mỗi database (schema) ở trong một thư mục con nằm trong một thư mục dữ liệu ở dưới file hệ thống. Khi tạo một table mới, MySQL sẽ lưu định nghĩa bảng trong file **.FRM** trùng tên bảng.

```
# ls -l /var/lib/mysql/mysql
```

### MyISAM

MyISAM là kỹ thuật lưu trữ mặc định (trước phiên bản 5.5), và được sử dụng ở hầu hết các ứng dụng web, ứng dụng kho dữ liệu và những môi trường ứng dụng khác.

MyISAM lưu mỗi bảng dữ liệu trên 2 file: **.MYD** cho file dữ liệu và **.MYI** cho file chỉ mục. Định dạng MyISAM không phụ thuộc vào nền tảng hệ điều hành, do đó có thể copy file dữ liệu và file index từ hệ điều hành này sang hệ điều hành khác mà hệ thống vẫn chạy được

#### Đặc điểm

- Thiết kế đơn giản nên phổ biến
- Không hỗ trợ khóa ngoại và quản lý giao dịch
- Cho phép đánh chỉ mục toàn cột (Full Text Index), cho tốc độ truy suất đọc và tìm kiếm nhanh nhất trong các Storage Engine
- Hoạt động theo kiểu *Table Level Locking*, khi cập nhật 1 bản ghi trong cùng 1 table thì table đó sẽ bị khóa lại, không cho cập nhật cho đến khi thao tác cập nhật trước đó thực hiện xong.

### InnoDB

InnoDB là kỹ thuật lưu trữ an toàn trong giao dịch như *commit, rollback* và khả năng phục hồi để bảo vệ dữ liệu người dùng. InnoDB cung cấp khóa hàng và tính toàn vẹn dữ liệu bằng các ràng buộc toàn vẹn tham chiếu khóa ngoại

InnoDB lưu dữ liệu trên file: **ibdata1** và log trên các file **ib\_logfile0, ib\_logfile1**

Thuộc tính **innodb\_file\_per\_table** cho phép đặt dữ liệu và chỉ mục cho riêng bảng vào một file riêng thay vì để trong **ibdata1**

#### Đặc điểm

- Hỗ trợ khóa ngoại cho kiểm tra tính toàn vẹn ràng buộc dữ liệu
- Hỗ trợ quản lý các giao dịch
- Với cơ chế COMMIT và ROLLBACK trong kỹ thuật lưu trữ sẽ giúp cho các giao dịch trong thương mại, ngân hàng an toàn và tin cậy hơn.
- Hoạt động theo kiểu *Row Level Locking*, khi cập nhật 1 bản ghi thì chỉ có bản ghi đang thao tác bị khóa, các hoạt động khác trên table này không bị ảnh hưởng.
- InnoDB dùng nhiều không gian lưu trữ hơn MyISAM



Feature	MyISAM	InnoDB
Storage limits	256TB	64TB
Transactions	No	Yes
Locking	Table	Row
Full-text search indexes	Yes	Yes <i>InnoDB support for FULLTEXT indexes is available in MySQL 5.6.4 and higher</i>
Clustered indexes	No	Yes
Data caches	No	Yes
Index caches	Yes	Yes
Compressed data	Yes <i>Compressed MyISAM tables are supported only when using the compressed row format. Tables using the compressed row format with MyISAM are read only</i>	Yes <i>Compressed InnoDB tables require the InnoDB Barracuda file format</i>
Encrypted data <i>Implemented in the server (via encryption functions), rather than in the storage engine</i>	Yes	Yes
Cluster database support	No	No
Replication support <i>Implemented in the server, rather than in the storage engine</i>	Yes	Yes
Foreign key support	No	Yes
Query cache support	Yes	Yes

✎ Giới hạn dung lượng lưu trữ của bảng cũng phụ thuộc giới hạn của File system mà OS dùng  
**Character set** là một tập hợp các ký tự và các phương thức chuyển mã ký tự (encoding method)  
 MySQL có thể lưu trữ dữ liệu ở nhiều dạng character set khác nhau ở các mức độ khác nhau: server, client, database. Mỗi character set có một collation mặc định.

**Character Set = Characters + Encoding Method**

**Collation** là qui tắc sắp xếp của một *Character set*. Theo qui tắc đặt tên collation của MySQL, *\_ci* là *case insensitive* (collation không phân biệt chữ hoa thường)

```
mysql> show character set;
```

```
mysql> select * from information_schema.schemata;
```

```
mysql> show table status from information_schema;
```



## 2. Cài đặt và cấu hình MySQL

### Cài đặt MySQL 5.5

#### Epel repositories

```
# yum install epel-release -y
```

#### Remi repositories

```
## RHEL/CentOS 5 ##
```

```
# rpm -ivh http://rpms.famillecollet.com/enterprise/remi-release-5.rpm
```

```
## RHEL/CentOS 6 ##
```

```
# rpm -ivh http://rpms.famillecollet.com/enterprise/remi-release-6.rpm
```

✂ *Remi repositories yêu cầu Epel repositories phải được cài đặt trước*

```
# yum --enablerepo=remi install mysql-server -y
```

### Cài đặt MySQL 5.6

**MySQL repositories** <http://dev.mysql.com/downloads/repo>

```
## RHEL/CentOS 5 ##
```

```
# wget http://repo.mysql.com/mysql-community-release-el5-5.noarch.rpm
```

```
# rpm -ivh mysql-community-release-el5-5.noarch.rpm
```

```
## RHEL/CentOS 6 ##
```

```
# wget http://repo.mysql.com/mysql-community-release-el6-5.noarch.rpm
```

```
# rpm -ivh mysql-community-release-el6-5.noarch.rpm
```

Cài đặt MySQL community server

```
# yum install mysql-server -y
```

✂ Khi cài đặt MySQL 5.6 trên CentOS 5.x bằng YUM từ **MySQL repositories**, nếu gặp lỗi cảnh báo “**warning: user mysql does not exist - using root**”, fix lỗi thiếu user *mysql* như sau:

```
# cat /etc/group | grep mysql
```

```
mysql:x:27:
```

```
# vim /etc/passwd
```

```
mysql:x:27:27:daemon:/sbin:/sbin/nologin
```

```
# chown mysql:mysql /var/run/mysqld
```

```
# chown mysql:mysql /var/lib/mysql
```

Cấu hình MySQL

```
# vim /etc/my.cnf
```

```
[mysqld]
```

```
datadir=/var/lib/mysql
```

```
socket=/var/lib/mysql/mysql.sock
```

```
user=mysql port=3306
```

```
max_connections=1000
```





```
bind-address = 127.0.0.1
#skip-networking character-set-
server=utf8 collation-
server=utf8_general_ci
## MyISAM key buffer for caching index data from disk
key_buffer_size=256M
## Set innodb buffer pool size to 50-80% of RAM for both data and indexes
innodb_buffer_pool_size=512M
lower_case_table_names=1
[mysqld_safe]
log-error=/var/log/mysql.log
pid-file=/var/run/mysql/mysql.pid
```

✂ It is recommended to set **innodb\_buffer\_pool\_size** to 50-80% of system memory. This allows MySQL to cache more data (the default is only 8M in MySQL 5.0 or 128M in MySQL 5.5) and can improve performance significantly.

Khi cài đặt MySQL trên Linux, mặc định Linux sẽ bật thông số phân biệt chữ hoa, chữ thường khi tạo thông tin CSDL. Để tắt thông số này thêm **lower\_case\_table\_names=1** vào trong section **[mysqld]**

**Tham khảo:** **/usr/share/mysql/my-large.cnf**, **/usr/share/mysql/my-huge.cnf**

Sau khi sửa đổi xong các thông số, lưu lại thông tin và start dịch vụ MySQL

```
# service mysql start
# netstat -penlt | grep mysql
tcp      0      0 0.0.0.0:3306          0.0.0.0:*            LISTEN   27      60287
21804/mysql
# netstat -penlx | grep mysql
unix 2      [ ACC ]  STREAM  LISTENING  60288 21804/mysql
/var/lib/mysql/mysql.sock
# mysqladmin ping
# mysqladmin status
```

**Uptime: 1886 Threads: 3 Questions: 282 Slow queries: 0 Opens: 41 Flush tables: 1 Open tables: 27 Queries per second avg: 0.149**

**Threads**      Số lượng các kết nối đang mở  
**Questions**    Số lượng các truy vấn đã gửi tới server từ khi bắt đầu  
**Slow queries**   Số các truy vấn thực thi lâu hơn biến hệ thống long\_query\_time system  
**Opens**        Số lượng các bảng đã mở từ khi server bắt đầu  
**Open tables**    Số các bảng đang mở và đang được truy cập  
**Queries per second avg**   Số lượng trung bình các truy vấn trong một giây





Lần đầu login vào MySQL Server bằng user **root** và password **rỗng**

```
# mysql -u root
```

*Your MySQL connection id is 3*

### 3. Quản trị căn bản MySQL

Kết nối máy chủ MySQL

```
# mysql -u root -p
```

```
# mysql --protocol=tcp -P 3306 -u root -p
```

```
# mysql -u root -h 127.0.0.1 -P 3306 -p
```

Lệnh help

```
mysql> help show
```

```
mysql> help create
```

Tạo database **CREATE DATABASE [IF NOT EXISTS] db\_name;**

```
mysql> create database wordpress;
```

Thay đổi cấu trúc database

```
mysql> alter database wordpress default character set utf8 collate utf8_general_ci;
```

```
mysql> show databases;
```

Chọn database làm việc: **USE db\_name;**

```
mysql> use wordpress;
```

Tạo bảng

**CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl\_name**

**[(create\_definition,...)]**

**[table\_options]**

**[select\_statement]**

**CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl\_name [( LIKE old\_tbl\_name  
D)];**

Tạo bảng **test**

```
mysql> create table test (id int not null primary key auto_increment, val int);
```

Thay đổi cấu trúc bảng

```
mysql> alter table test engine = myisam;
```

Xem thông tin về bảng

```
mysql> show tables;
```

```
mysql> show table status;
```

Xem thông tin về cột trong bảng

```
mysql> desc test;
```

Chèn dữ liệu vào bảng

```
mysql> insert into test(val) values (1),(2),(3),(4),(5),(6),(7),(8),(9),(10);
```

```
mysql> select * from test;
```

Từ khóa **TEMPORARY** để tạo bảng tạm thời, bảng này chỉ tồn tại trong connection và sau đó sẽ bị xóa khi disconnect





```
mysql> create temporary table mysql.user1 select * from mysql.user;
mysql> show global status like 'created_tmp%';
```

```
+-----+-----+
| Variable_name | Value |
+-----+-----+
| Created_tmp_disk_tables | 9 |
| Created_tmp_files | 6 |
| Created_tmp_tables | 63 |
+-----+-----+
```

```
mysql> show variables like '%tmp%';
```

```
+-----+-----+
| Variable_name | Value |
+-----+-----+
| max_tmp_tables | 32 |
| slave_load_tmpdir | /tmp |
| tmp_table_size | 16777216 |
| tmpdir | /tmp |
+-----+-----+
```

```
# ls /tmp
```

```
mapping-root #sqlf72_a_0.frm
```

Từ khóa **IF NOT EXISTS** dùng để tránh phát sinh lỗi khi bảng đã tồn tại

Bạn có thể tạo bảng từ 1 câu lệnh **SELECT** với cú pháp **CREATE TABLE tbl\_name SELECT...**, tuy nhiên phải chú ý đến tên trường của câu lệnh **SELECT**

Để tạo 1 bảng mới với cấu trúc giống với một bảng đã có, có thể sử dụng **CREATE TABLE ... LIKE** (lệnh này chỉ sao chép cấu trúc, không sao chép dữ liệu)

Xóa database

```
DROP DATABASE [IF EXISTS] db_name;
```

```
mysql> select version();
```


```
mysql> select database();
```

```
mysql> select now();
```

```
mysql> select current_user();
```

```
mysql> select user();
```


Tạo user quản trị từ xa

 user quản trị không có quyền **grant**

```
mysql> create user root@'%' identified by '123456';
```

```
mysql> grant all privileges on *.* to root@'%';
```

```
mysql> grant all privileges on *.* to admin1@'%' identified by '123456';
```


 user quản trị có quyền **grant**







```
mysql> grant all privileges on *.* to admin2@%' identified by '123456' with grant option;  
mysql> SELECT * FROM information_schema.user_privileges WHERE PRIVILEGE_TYPE =  
'SUPER';  
mysql> select user,host from mysql.user where Super_priv = 'Y';  
Xóa user khỏi hệ thống  
mysql> drop user admin1 @'%';  
Kiểm tra lại các tham số hệ thống: character_set, autocommit... bằng lệnh show variables;  
mysql> show variables like 'character_set%';  
mysql> show variables like 'autocommit';  
mysql> show variables like 'innodb%';  
innodb_data_file_path ibdata1:12M:autoextend  
innodb_file_per_table ON  
mysql> select * from information_schema.global_variables;  
mysql> select * from information_schema.session_variables;
```

 DB **INFORMATION\_SCHEMA** chứa siêu dữ liệu (*metadata*) về các DB trong MySQL

Kiểm tra Storage Engines MySQL support

```
mysql> show engines;
```

```
mysql> exit
```

Khai báo user và password client

```
# vim ~/.my.cnf
```

```
[client]
```

```
user=root
```

```
password=123456
```

Kết nối máy chủ MySQL không cần user và password

```
# mysql
```

Xem lịch sử lệnh MySQL

```
# more ~/.mysql_history
```

## 4. Phân quyền MySQL

### Nguyên tắc bảo mật

- Không cho người dùng nào ngoài **root** có quyền truy cập database **mysql**
- Không để password người dùng trống
- Quản lý phân quyền bằng lệnh **GRANT & REVOKE**
- Không lưu **PASSWORD** dưới dạng *clear text* (Sử dụng các hàm **MD5**, **SHA** để mã hóa mật khẩu)

**GRANT & REVOKE** cho phép tạo mới người dùng, cấp và hủy bỏ quyền của người dùng

Quyền trong MySQL được chia làm 4 level





- **Global level:** Áp dụng cho tất cả các CSDL và bảng **GRANT/REVOKE ALL ON \*.\*;**
- **Database level:** Áp dụng tất cả các bảng trong 1 CSDL **GRANT/REVOKE ALL ON db.\*;**
- **Table level:** Áp dụng cho tất cả các cột trong 1 bảng xác định **GRANT/REVOKE ALL ON db.table;**
- **Column level:** Áp dụng cho 1 cột xác định trong bảng

### Tài khoản & mật khẩu

Các tài khoản và mật khẩu của MySQL độc lập với tài khoản và mật khẩu của Linux

Tài khoản username trong MySQL dài tối đa 16 ký tự

Mật khẩu trong MySQL được mã hóa bằng hàm **PASSWORD** và giải mã bằng hàm **ENCRYPT**

Tài khoản & mật khẩu người dùng trong MySQL được tạo, cấp và rút quyền bằng lệnh **GRANT** & **REVOKE**

Mật khẩu người dùng được lưu trong bảng user dưới dạng mã hóa

Để kết nối đến máy chủ MySQL, cần cung cấp username & password. Nếu kết nối máy chủ từ xa, phải xác định hostname **mysql -h hostname -u username -p**

Thay đổi mật khẩu user bằng lệnh **mysqladmin** hoặc **set password**

**mysqladmin -u root password NEWPASSWORD**

**mysqladmin -u root -pOLDPASSWORD password NEWPASSWORD**

**# mysqladmin -u root password 123456**

**# mysql -u root -p**

**mysql> set password = password('123456');**

**mysql> exit**

### Các quyền truy cập database

- **ALL** hay **ALL PRIVILEGES** Tất cả các đặc quyền ngoại trừ **GRANT OPTION**
- **CREATE TEMPORARY TABLES** Cho phép tạo các bảng temp
- **EXECUTE** Quyền thực thi các Functions, Procedures
- **FILE** Cho phép load data infile
- **GRANT OPTION** Grant quyền cho các user
- **LOCK TABLES** Cho phép lock các tables
- **PROCESS** View các thông tin về các thread thực thi trên host
- **RELOAD** Gán quyền Reload tới các tables và flush log hay cache database
- **REPLICATION SLAVE** Cho phép đọc binary log events từ master
- **SHOW DATABASES** View các database trên hệ thống
- **SHUTDOWN** Shut down server
- **SUPER** Kill thread, SET GLOBAL, CHANGE MASTER,...
- **ALTER** Thay đổi cấu trúc table và index
- **CREATE** Tạo database và table
- **DELETE** Delete các bản ghi từ table
- **DROP** Drop (remove) database và table
- **INDEX** Create hay drop index





- **INSERT** Insert các bản ghi mới vào bảng
- **REFERENCES** Unused
- **SELECT** Truy vấn nội dung từ bảng
- **UPDATE** Thay đổi nội dung bảng
- **USAGE** Quyền sử dụng các database


Xem danh sách quyền của người dùng

```
mysql> show privileges;
```

Tạo và phân quyền cho user với lệnh **GRANT**

Tạo **user1** có quyền **select** bảng **user** trong database **mysql**

```
mysql> grant usage on *.* to user1 @'localhost' identified by '123456';
```

 *usage can be specified to create a user that has no privileges*

```
mysql> grant select on mysql.user to user1 @'localhost';
```

```
mysql> show grants for root@localhost;
```

```
+-----+
/ Grants for root@localhost                                     /
+-----+
/ GRANT ALL PRIVILEGES ON *.* TO 'root'@'localhost' IDENTIFIED BY PASSWORD
'565491d704013245' WITH GRANT OPTION /
+-----+
1 row in set (0.00 sec)
```



Copy user mới bằng câu lệnh **SHOW GRANTS FOR**

Chú ý quyền trong cột **Grant\_priv** nếu copy từ user **root@localhost**

**IDENTIFIED BY PASSWORD** copy mật khẩu sang user mới

```
mysql> grant all privileges on *.* to 'root'@'192.168.15.139' identified by password
```

```
'565491d704013245' with grant option;
```

```
mysql> show grants for root@192.168.15.139;
```

```
+-----+
--
/ Grants for root@192.168.15.139                               /
+-----+
/ GRANT ALL PRIVILEGES ON *.* TO 'root'@'192.168.15.139' IDENTIFIED BY PASSWORD
'565491d704013245' WITH GRANT OPTION /
+-----+
```

Tạo user quản trị có quyền trên tất cả database và chỉ có thể kết nối MySQL server từ localhost

```
mysql> grant all privileges on *.* to admin1 @localhost identified by '123456';
```

```
mysql> select user,host,password from mysql.user;
```

Nếu trong câu lệnh **GRANT** không có **IDENTIFIED BY** thì user sẽ được tạo với mật khẩu rỗng





```
mysql> grant all privileges on *.* to admin2@localhost;
```

Nếu đổi mật khẩu cho 1 user mà không muốn thay đổi lại các quyền hiện có của user đó, MySQL cung cấp lệnh **SET PASSWORD**

```
mysql> set password for admin2@localhost = password('123456');
```

Tạo user *wordpress* có quyền thao tác database *wordpress* và chỉ có thể kết nối từ localhost

```
mysql> grant create,alter,drop,select,insert,update,delete,index, execute on wordpress.* to wordpress@localhost identified by '123456';
```

```
mysql> select * from mysql.user where user='wordpress'\G
```

### Giới hạn tài nguyên người dùng

Để đảm bảo an toàn, nên giới hạn tài nguyên mà người dùng được phép sử dụng

- Số lượng query trong 1 giờ
- Số lượng update trong 1 giờ
- Số lượng connection được cấp trong 1 giờ

#### **GRANT ... WITH**

**MAX\_QUERIES\_PER\_HOUR N1**

**MAX\_UPDATES\_PER\_HOUR N2**

**MAX\_CONNECTIONS\_PER\_HOUR N3;**

Tạo user *wordpress* có quyền thao tác database *wordpress* và chỉ có thể kết nối từ máy 192.168.1.2 với tham số tùy chọn giới hạn tài nguyên

```
mysql> grant create,alter,drop,select,insert,update,delete,index on wordpress.* to wordpress@'192.168.1.2' identified by '123456' with max_queries_per_hour 20 max_updates_per_hour 10 max_connections_per_hour 5;
```

Xóa quyền của user với lệnh **REVOKE**

```
mysql> revoke all privileges on database_name.* from admin1@localhost;
```



Không thể **revoke all privileges on \*.\***

Thông tin về quyền của tài khoản được MySQL lưu trữ trong các bảng: *user*, *db*, *tables\_priv*, *columns\_priv*, *procs\_priv* trong database **mysql**

- **user**: chứa những tài khoản, quyền cục bộ và những cột không phân quyền
- **db**: chứa những quyền database theo cấp độ
- **tables\_priv**: chứa những quyền table theo cấp độ
- **columns\_priv**: chứa những quyền column theo cấp độ
- **procs\_priv**: chứa những quyền của hàm và thủ tục

### Script copy MySQL user và Database Priveleges

```
CREATE TABLE mysql.user_copy SELECT * FROM mysql.user;
DELETE FROM mysql.user_copy WHERE Host NOT LIKE 'OLD_HOST_NAME';
UPDATE mysql.user_copy SET Host = 'NEW_HOST_NAME';
INSERT INTO mysql.user SELECT * FROM mysql.user_copy;
DROP TABLE mysql.user_copy;
```





```
CREATE TABLE mysql.db_copy SELECT * FROM mysql.db;
DELETE FROM mysql.db_copy WHERE Host NOT LIKE 'OLD_HOST_NAME';
UPDATE mysql.db_copy SET Host = 'NEW_HOST_NAME';
INSERT INTO mysql.db SELECT * FROM mysql.db_copy;
DROP TABLE mysql.db_copy;
```

```
FLUSH PRIVILEGES;
```

## 5. Phục hồi mật khẩu MySQL root

Nếu quên mật khẩu *root* của MySQL, thực hiện những bước sau để khôi phục mật khẩu:

Stop MySQL Server

```
# service mysqld stop
```

Khởi động MySQL Server ở chế độ safe mode

```
# mysqld_safe --skip-grant-tables &
```

Kết nối MySQL server không cần mật khẩu

```
# mysql -u root
```

Đặt mật khẩu mới cho user root của MySQL

```
mysql> use mysql;
```

```
mysql> update user set password=password('123456') where user='root';
```

```
mysql> exit
```

Khởi động MySQL server và login

```
# service mysqld restart
```

```
# mysql -u root -p
```

## 6. Câu lệnh SQL

Tạo database **example1**

```
mysql> create database example1;
```

```
mysql> show databases;
```

Tạo bảng **test** thuộc database **example1**

```
mysql> use example1; create table test (id int not null primary key auto_increment, val int)
```

```
engine = myisam; mysql>
```

```
show tables; mysql>
```

```
show table status;
```

```
mysql> select table_name, engine from information_schema.tables where table_schema =
'example1';
```

```
mysql> show create table test;
```

**Stored procedure**





Tạo thủ tục:

```
create procedure tên-thủ-tục([các-tham-số])  
begin  
    các-câu-lệnh;  
end
```

Xoá thủ tục: **drop procedure tên-thủ-tục;**

Gọi thủ tục: **call tên-thủ-tục**([các-tham-số]);

Hàm tương tự như thủ tục nhưng có giá trị trả về

Tạo thủ tục thêm 1000000 bản ghi vào bảng test

```
-- change delimiter to //  
delimiter //  
create procedure add_data()  
begin  
    declare i int default 1;  
    while i <= 1000000 do  
        insert into test (val) values (i);  
        set i = i + 1;  
    end while;  
end //  
-- change delimiter to ;  
delimiter ;
```



**delimiter //** Đổi ký hiệu phân tách câu lệnh (để dùng dấu “;” bên trong thủ tục con)

**delimiter ;** Đổi lại ký hiệu phân tách thành dấu “;”

Show thủ tục

```
mysql> show create procedure add_data;  
mysql> show procedure status;  
mysql> select type,db,name,body from mysql.proc;
```



Grant quyền cho user khác view procedure

```
GRANT SELECT ON mysql.proc TO 'wordpress'@'localhost';
```

Gọi thủ tục

```
mysql> call add_data();  
mysql> show open tables from example1;  
mysql> show open tables where in_use > 0;  
mysql> flush tables with read lock;  
mysql> show open tables;  
mysql> unlock tables;
```





```
mysql> show status like 'table%';
```



**SHOW OPEN TABLES [FROM database] [LIKE 'pattern'|WHERE expression]**

**OPTIMIZE TABLE *tbl\_name* [, *tbl\_name*] ...**

Giống Compact của Access hay Defragment của Windows

Xóa các khoảng trống hình thành do xóa dòng, thay đổi dữ liệu các trường *VARCHAR*, *BLOB*, *TEXT*

```
mysql> select count(*) from test;
```

```
+-----+
| count(*) |
+-----+
| 4100000 |
+-----+
```

1 row in set (0.00 sec)

```
mysql> delete from test where id <= 100000;
```

```
# ls -lh /var/lib/mysql/example1/
```

**total 76M**

```
-rw-rw---- 1 mysql mysql 65 Sep  4 16:25 db.opt
-rw-rw---- 1 mysql mysql 8.4K Sep  4 16:25 test.frm
-rw-rw---- 1 mysql mysql 36M Sep  4 16:31 test.MYD
-rw-rw---- 1 mysql mysql 41M Sep  4 16:31 test.MYI
```

```
mysql> optimize table test;
```

```
+-----+-----+-----+-----+
| Table      | Op      | Msg_type | Msg_text |
+-----+-----+-----+-----+
| example1.test | optimize | status   | OK       |
+-----+-----+-----+-----+
```

1 row in set (0.04 sec)

```
# ls -lh /var/lib/mysql/example1/
```

**total 74M**

```
-rw-rw---- 1 mysql mysql 65 Sep  4 16:25 db.opt
-rw-rw---- 1 mysql mysql 8.4K Sep  4 16:25 test.frm
-rw-rw---- 1 mysql mysql 35M Sep  4 16:31 test.MYD
-rw-rw---- 1 mysql mysql 39M Sep  4 16:31 test.MYI
```

**Câu lệnh truy vấn trong MySQL**

**SELECT ... FROM *tbl\_name* WHERE {CONDICTIONS}**

**GROUP BY ... HAVING ... ORDER BY ...LIMIT START, TOTAL;**







Câu lệnh SELECT là 1 trong những phát biểu yêu cầu MySQL truy vấn dữ liệu trên cơ sở dữ liệu chỉ định.

Mệnh đề FROM chỉ ra tên 1 bảng hay những bảng có liên quan cần truy vấn thông tin

Mệnh đề WHERE để tạo nên điều kiện cần lọc mẫu tin theo tiêu chuẩn được định nghĩa .

Các phép toán so sánh

*WHERE id > 10;*

*WHERE id >= 10;*

*WHERE id = 10;*

*WHERE id < 10;*

*WHERE id <= 10;*

*WHERE id != 10;*

*WHERE id <> 10;*

Các phép toán logic

*WHERE id=1 AND name='admin';*

*WHERE id=1 OR name='admin';*

*WHERE name LIKE '%admin';*

*WHERE name NOT LIKE '%admin';*

*WHERE id IN ('10','20');*

*WHERE id NOT IN( '10','20');*

*WHERE id BETWEEN 10 And 20;*

*WHERE password is not NULL;*

Thông thường trong khi truy vấn, kết quả hiển thị sắp xếp theo chiều tăng hay giảm dựa trên ký tự ALPHABET.

Cú pháp cho mệnh đề ORDER BY cùng với trạng thái tăng (ASC), giảm dần (DESC)

*ORDER BY col\_name {ASC/DESC}*

Hầu hết câu lệnh của ANSI SQL đều tương thích trong MySQL, tuy nhiên câu lệnh **select ... from ... where ... limit start, total** linh hoạt hơn **select top N** của MS SQL Server (T-SQL)

Trong đó

**start**: record đầu tiên  $0 \leq start < count(*)$

**total** : tổng số record được hiển thị

*mysql> select \* from example1.test limit 20;*

Câu lệnh **select ... from ... where ... limit start, total** trong MySQL là cơ sở để thực hiện việc phân trang trong PHP

## Hàm trong MySQL

Các hàm tổng hợp dữ liệu: *avg, count, sum, min, max*

Các hàm riêng của MySQL: *first, last, ucase, lcase, mid, now, len, round, format*

Cách dùng các hàm: *select function() as col\_name*

*mysql> select current\_date, date(now()), date\_format(sysdate(), '%d/%m/%Y') as sysdate;*

```
+-----+-----+-----+
| current_date | date(now()) | sysdate |
+-----+-----+-----+
| 2014-03-26 | 2014-03-26 | 26/03/2014 |
+-----+-----+-----+
1 row in set (0.00 sec)
```

Get size of database







**data\_length:** store the real data.

**index\_length:** store the table index.

Get size of all database

```
SELECT table_schema "Data Base Name", SUM( data_length + index_length) / 1024 / 1024  
"Data Base Size in MB" FROM information_schema.TABLES GROUP BY table_schema ;
```

Get size of all database tables

```
SELECT TABLE_SCHEMA AS 'Database', TABLE_NAME AS 'Table',  
CONCAT(ROUND(((DATA_LENGTH + INDEX_LENGTH - DATA_FREE) / 1024 / 1024),2),"  
Mb") AS Size FROM INFORMATION_SCHEMA.TABLES;
```

### Tạo Index

Index (chỉ mục) được tạo ra nhằm truy xuất dữ liệu nhanh và hiệu quả hơn. Index có thể được tạo trên một hoặc nhiều cột của bảng, và mỗi Index được đặt một tên. Người dùng không thấy được các Index này, chúng chỉ được dùng để tăng tốc cho CSDL.

Sau khi bảng tạo Index, việc cập nhật thêm dòng mới vào bảng sẽ mất nhiều thời gian hơn là đối với bảng không đánh Index. Vì thế, chỉ nên tạo Index cho các cột thường xuyên dùng trong các tác vụ tìm kiếm.

**Unique Index:** Chỉ mục đơn nhất sẽ bắt buộc hai dòng bất kỳ của bảng sẽ không được phép mang cùng giá trị ở cột được tạo chỉ mục.

```
CREATE UNIQUE INDEX index_name ON tbl_name (col_name);
```

**Simple Index:** Không dùng từ khoá UNIQUE trong câu lệnh tạo chỉ mục, các giá trị trùng nhau trong cột sẽ được phép.

```
CREATE INDEX index_name ON tbl_name (col_name);
```

Tạo bảng USER trong database **example1**

```
mysql> use example1;
```

```
mysql> create table user(id int not null primary key, name varchar(30), password varchar(25))  
engine=myisam;
```

Tạo một chỉ mục đơn **idx\_name** trên cột **name** của bảng USER

```
mysql> create index idx_name on user (name);
```

Show index

```
mysql> show index from user;
```

```
mysql> show index from user where key_name = "PRIMARY";
```

Xoá chỉ mục đã tạo bằng lệnh DROP

```
DROP INDEX index_name ON tbl_name;
```

```
mysql> drop index idx_name on user;
```

### Phân tích câu lệnh SELECT

```
mysql> explain select * from example1.test where id = 1;
```





```
mysql> explain select * from example1.test where val = 1;
```

### Performance/Load Testing

**mysqlslap** là công cụ trong mysql-client được thiết kế để giả lập các client nhằm kiểm tra tải cho máy chủ MySQL, đồng thời báo cáo thời gian máy chủ MySQL trả về kết quả truy vấn cho client

```
# mysqlslap --auto-generate-sql --concurrency=10 --number-of-queries=1000
# mysqlslap --create-schema=mysql --user=root -p --query="SELECT * FROM user" --
concurrency=50 --iterations=200
```

Trong ví dụ này, công cụ *mysqlslap* truy vấn dữ liệu trong bảng *user* của CSDL *mysql* với số client kết nối đến là 50 và được lặp lại 200 lần

Để đo thời gian thực hiện của 1 hàm, có thể sử dụng lệnh **BENCHMARK**

## the example below shows that the divide takes 10 times longer to run than the addition

```
mysql> select benchmark(10000000, 1+1);
```

```
mysql> select benchmark(10000000, 1/1);
```

## the example below extracts the date from a datetime, surprisingly the left function is faster

```
mysql> select benchmark(10000000, left('2012-05-14 00:00:00',10));
```

```
mysql> select benchmark(10000000, date('2012-05-14 00:00:00'));
```

### Kiểm tra hiệu năng MySQL bằng sysbench

```
# yum install sysbench -y
```

Chuẩn bị dữ liệu test

```
# sysbench --test=oltp --oltp-table-size=1000000 --db-driver=mysql --mysql-db=test --mysql-
user=root --mysql-password=123456 prepare
```

Thực hiện test

```
# sysbench --test=oltp --oltp-table-size=1000000 --db-driver=mysql --mysql-db=test --mysql-
user=root --mysql-password=123456 --max-time=60 --oltp-read-only=on --max-requests=0 --
num-threads=8 run
```

```
mysql> show processlist;
```

Xóa dữ liệu test

```
# sysbench --test=oltp --db-driver=mysql --mysql-db=test --mysql-user=root --mysql-
password=123456 cleanup
```

## 7. Quản lý cấu hình log

### Slow query log

Slow query log trong MySQL để log lại những query chậm, chiếm tài nguyên hệ thống

Cấu hình log lại các slow query không cần khởi động lại MySQL

```
# touch /var/log/mysql-slow-query.log
```

```
# chown mysql:mysql /var/log/mysql-slow-query.log
```

```
# mysql -u root -p
```

```
mysql> set global slow_query_log = 'ON';
```

```
mysql> set global slow_query_log_file = '/var/log/mysql-slow-query.log';
```





```
mysql> set long_query_time = 5;
```

 *long\_query\_time* default is 10 seconds

- *slow\_query\_log* kích hoạt tính năng slow query log
- *slow\_query\_log\_file* khai báo đường dẫn slow query log file
- *long\_query\_time* thiết lập thời gian cho query được tính là slow query. Đơn vị tính theo giây

```
mysql> show variables like '%slow%';
```

```
mysql> show variables like '%long_query%';
```

Cấu hình log lại các query không dùng index

```
mysql> set global log_queries_not_using_indexes = 'ON';
```

```
# mysql -e 'SELECT SLEEP(12);'
```

```
# tail -f /var/log/mysql-slow-query.log
```

```
# Time: 140710 17:57:38
```

```
# User@Host: root[root] @ localhost []
```

```
# Query_time: 12.001264 Lock_time: 0.000000 Rows_sent: 1 Rows_examined: 0
```

```
SET timestamp=1404989858;
```

```
SELECT SLEEP(12);
```

```
# mysql -e "SELECT BENCHMARK(1000000000, CONCAT('a','b'));"
```

 **Định dạng output slow query log**

- Time: thời điểm câu lệnh slow query được log, thời gian lấy theo máy chủ với định dạng **YYMMDD H:M:S**

- User@Host: câu lệnh được thực hiện từ user@host

- Query\_time, Lock\_time, Rows\_sent, Rows\_examined

- SET timestamp=UNIXTIME; thời điểm câu lệnh query chạy

- Toàn bộ câu lệnh slow query

Dùng lệnh **date -d@** để đọc timestamp

```
# date -d @1404989858
```

```
Thu Jul 10 17:57:38 ICT 2014
```

Cấu hình slow query log

```
# vim /etc/my.cnf
```

```
[mysqld]
```

```
slow_query_log=1
```

```
slow_query_log_file="/var/log/mysql-slow-query.log"
```

```
long_query_time=5
```

```
# service mysqld restart
```

**General log**

```
# mysql -u root -p
```

```
mysql> SHOW VARIABLES LIKE 'general_log%';
```

```
mysql> SET GLOBAL general_log = 'ON';
```





```
mysql> SET GLOBAL general_log_file ='/var/log/mysqld.log';
```

## 8. Quản trị MySQL process

Lệnh **show process list** liệt kê danh sách process đang chạy trên máy chủ cơ sở dữ liệu MySQL

```
# mysql -u root -p
```

```
mysql> show processlist;
```

```
+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+
| Id | User | Host   | db   | Command | Time | State | Info                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+
|  4 | root | localhost | example1 | Query   |  0 | query end | insert into test (val) values (
NAME_CONST('i',33812)) |
|  6 | root | localhost | NULL | Query   |  0 | NULL | show processlist
|
+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+
```

2 rows in set (0.00 sec)

Kill process với **id = 4**

```
mysql> kill 4;
```

- **Id**: Khi thực hiện một câu lệnh hoặc một công việc thì sẽ sinh ra một process. Mỗi một process bắt đầu được đánh bằng một số ID duy nhất để nhận diện. Giá trị các Process ID được đánh số theo thứ tự tăng dần. Bắt đầu từ 0 và tăng lên cho tới khi gặp giá trị maximum. Giá trị maximum của Process ID là có thể cấu hình được tùy vào từng hệ thống.
- **User**: Tài khoản người dùng.
- **Host**: Tên máy kết nối đến máy chủ CSDL
- **db**: Cơ sở dữ liệu mà quá trình đang được chạy. NULL sẽ được hiển thị nếu không có cơ sở dữ liệu được chọn.
- **Command**: Lệnh đang được thực thi
- **Time**: Thời gian tính theo giây kể từ lúc lệnh cuối cùng được thực hiện
- **State**: Trạng thái của process. Nếu như là NULL thì sẽ hiển thị “SHOW PROCESSLIST” trong cột Info.
- **Info**: Thông tin về các process

```
# mysql -u root -p
```

```
mysql> select connection_id();
```

```
mysql> prompt MySQL ID 2>;
```

```
mysql> show status like 'max_used_connections';
```

```
mysql> show variables like 'max_connections';
```

```
mysql> set global max_connections = 2;
```





```
mysql> show processlist;
mysql> show status like 'conn%';
```

```
+-----+-----+
/ Variable_name / Value /
+-----+-----+
/ Connections  / 12  /
+-----+-----+
1 row in set (0.00 sec)
```

🔗 **Connections:** số connections được mở tính từ thời điểm MySQL start

Monitor MySQL Process List in Realtime

```
# watch -n 1 mysqladmin processlist
# watch -n 1 mysqladmin -u root -p123456 processlist
```

🔗 Từ phiên bản **MySQL 5.1+**, có thể truy vấn bảng **processlist** trong database

### INFORMATION\_SCHEMA

```
mysql> select * from information_schema.processlist;
-- display number of connections for each host
mysql> select host,count(*) from information_schema.processlist group by host;
```

Cài đặt công cụ giám sát MySQL process

```
# yum install mytop -y
```

Cấu hình các thông số kết nối MySQL

```
# vim ~/.mytop
```

```
user=root
pass=
host=localhost
db=mysql
delay=2
port=3306
socket=
```

```
# mytop
```

## 9. Backup

```
mysql> create database ex2; use ex2;
mysql> create table test (id int not null primary key) engine = myisam;
mysql> insert into test values (1),(2),(3),(4),(5),(6),(7),(8),(9),(10);
mysql> select * from test;
Backup dữ liệu table: select into outfile from
mysql> select * into outfile '/tmp/test.sql' from test;
```

Xoá dữ liệu trong table





```
mysql> delete from test;
```

Phục hồi dữ liệu table: *load data infile replace into table*

```
mysql> load data infile '/tmp/test.sql' replace into table test;
```

```
mysql> select * from test;
```

**Load data infile** cần được grant với quyền **file** như sau:


```
mysql> grant file on *.* to user@hostname;
```

Backup database MyISAM

```
# mkdir -p /backup/mysql
```

```
# mysql -u root -p
```

```
# cp -rp /var/lib/mysql/ex2/ /backup/mysql
```

 *Lệnh **cp** option **-p** cho phép copy các thuộc tính của file*

```
# ls -l /backup/mysql
```

Xóa database

```
# mysql -u root -p
```

```
mysql> drop database ex2;
```

```
mysql> show databases;
```

Phục hồi database

```
# cp -rp /backup/mysql/ex2 /var/lib/mysql
```

```
# mysql -u root -p
```

```
mysql> show databases;
```

Check table

```
mysql> check table ex2.test;
```

Repair table nếu gặp lỗi: *Database failed to execute query (query) 1016: Can't open file: 'sometable.MYI'. (errno: 145) Error Msg: 1034: Incorrect key file for table:...*

```
mysql> repair table ex2.test;
```

**Backup với mysqldump**

```
# mysqldump -help
```

Backup một database bao gồm cả stored procedures, functions và events

```
mysqldump -u username -p --routines --events db_name > db_name.sql
```

Backup cấu trúc một database không bao gồm data

```
mysqldump -u username -p --routines --events --no-data db_name > db_name.sql
```

```
# mysqldump -uroot -p --routines --events wordpress > /backup/mysql/wordpress.sql
```

Restore một database

```
mysql -u username -p db_name < db_name.sql
```

```
# mysql -u root -p
```

```
mysql> drop database wordpress;
```

```
mysql> create database wordpress;
```

```
mysql > use wordpress;
```





```
mysql >source /backup/mysql/wordpress.sql
```

```
mysql > exit;
```

Hoặc

```
# mysql -uroot -p wordpress < /backup/mysql/wordpress.sql
```

Backup một table

```
mysqldump -u username -p db_name table_name > table_name.sql
```

Restore một table

```
mysql -u username -p new_db_name < table_name.sql
```

## 10. Duplicate

### Duplicate database

Cú pháp:

```
mysqldump -h [server] -u [user] -p[password] original_database | mysql -h [server] -u [user] -p[password] duplicate_database
```

Duplicate CSDL db1 sang CSDL mới db2

Tạo CSDL gốc db1

```
# mysql -u root -p
```

```
mysql> create database db1; use db1;
```

```
mysql> create table test (id int not null primary key);
```

```
mysql> insert into test values (1),(2),(3),(4),(5);
```

```
mysql> select * from test;
```

Tạo và duplicate CSDL db2

```
mysql> create database db2;
```

```
# mysqldump -uroot -p123456 db1 | mysql -uroot -p123456 db2
```

```
# mysql -u root -p
```

```
mysql> use db2;
```

```
mysql> select * from test;
```

### Duplicate table

Cú pháp:

```
CREATE TABLE IF NOT EXISTS new_table
```

```
SELECT * FROM existing_table
```

```
WHERE conditions
```

```
# mysql -u root -p
```

```
mysql> use db1;
```

```
mysql> create table if not exists new_test select * from test;
```

## 11. Thực thi SQL từ Bash

Thực thi câu lệnh SQL từ Bash Shell

Cú pháp:

```
mysql -h [ip] -u [user] -p[pass] -e "[mysql commands]"
```

```
mysql -D [db name] -u [user] -p[pass] -e "[mysql commands]"
```





```
# mysql -u root -p123456 -D mysql -e "select host,user from user"
# mysql -u root -p123456 -D information_schema -e "select * from processlist"
# mysql -u root -p123456 --execute="SELECT user,host FROM mysql.user"
```

Thực thi câu lệnh SQL từ Bash script dùng EOF

Cú pháp:

```
mysql -u [user] -p[pass] << EOF
```

```
[mysql commands]
```

```
EOF
```

```
#!/bin/bash
```

```
mysql -u root -p123456 << EOF
```

```
use mysql;
```

```
show tables;
```

```
EOF
```

Tham khảo:

```
# vim check_processlist.sh
```

```
#!/bin/bash
```

```
# Author: Nguyen Truong Giang
```

```
#####
```

```
# Nagios plugin to check processlist          #
```

```
#####
```

```
HOST="localhost"
```

```
PORT=3306
```

```
## Nagios return codes
```

```
STATE_OK=0
```

```
STATE_WARNING=1
```

```
STATE_CRITICAL=2
```

```
STATE_UNKNOWN=3
```

```
WARNING=5
```

```
CRITICAL=10
```

```
usage()
```

```
{
```

```
    echo "Usage: $0 -h [Host/IP] -p [Port Number] -w [limit] -c [limit]"
```

```
    echo "Options:"
```







```
    echo "-w=INTEGER"
    echo "  Exit with WARNING status if more than INTEGER"
    echo "-c=INTEGER"
    echo "  Exit with CRITICAL status if more than INTEGER"
    exit $STATE_UNKNOWN
}

## Parse commandline arguments with getopt
while getopt "h:p:w:c:u" args; do
    case $args in
        h) HOST=$OPTARG
            ;;
        p) PORT=$OPTARG
            ;;
        w) WARNING=$OPTARG
            ;;
        c) CRITICAL=$OPTARG
            ;;
        u) usage
            ;;
    esac
done

if [[ $CRITICAL -lt $WARNING ]];then
    echo "UNKNOWN: Warning threshold must be lower than Critical threshold"
    exit $STATE_UNKNOWN
fi

TELNET=$(which telnet 2>/dev/null)
if [ "$?" -eq "0" ]; then
    ## Test MySQL connect
    echo quit | $TELNET $HOST $PORT 2>/dev/null | grep Connected >/dev/null
    ## if last command return value !=0
    if [ "$?" -ne "0" ]; then
        echo "MySQL CRITICAL - Can't connect to MySQL server"
        exit $STATE_CRITICAL
    fi
fi
```





```
MYSQL=$(which mysql)
DB="information_schema"
QUERY_TIME=0
```

```
## Create user monitor with command below
## GRANT PROCESS ON *.* to monitor@'localhost' identified by 'sa3tHJ3';
USER="monitor"
PASSWD="sa3tHJ3"
```

```
SQL=$(($MYSQL -h $HOST -P $PORT -u $USER -p$PASSWD -D $DB -e "select concat('id:',
id), concat('user:', user,'@', host), concat('time:', time), concat('info:', info) from processlist
where command = 'query' and time > 0 order by time desc limit 1" 2>/dev/null | grep select)
```

```
## get length of string
if [ ${#SQL} == 0 ]; then
    echo "MySQL OK - No Slow query"
    exit $STATE_OK
else
    QUERY_TIME=$(echo $SQL | cut -d: -f4 | awk '{ print $1 }')
    if [ $QUERY_TIME -ge $CRITICAL ];then
        echo "MySQL CRITICAL - The slowest query is running: $SQL"
        exit $STATE_CRITICAL
    elif [ $QUERY_TIME -ge $WARNING ]; then
        echo "MySQL WARNING - The slowest query is running: $SQL"
        exit $STATE_WARNING
    else
        echo "MySQL OK - The slowest query is running: $SQL"
        exit $STATE_OK
    fi
fi
```

fi

## 12. Tuning và Troubleshooting

```
# cat /proc/sys/vm/swappiness
60
# cat /proc/sys/fs/file-max
45396
# cat /proc/sys/net/ipv4/ip_local_port_range
32768 61000
# vim /etc/sysctl.conf
```





```
vm.swappiness = 10
# Increase system file descriptor limit to
fs.file-max = 65536
# Increase system IP port limits
net.ipv4.ip_local_port_range = 10000 65000
# sysctl -p
# vim /etc/security/limits.conf
*          soft  nofile      10000
*          hard  nofile      10000
# ulimit -n
Kiểm tra giới hạn Max open files của mysql process
# service mysqld status
mysqld (pid 7727) is running...
# cat /proc/7727/limits
```

Limit	Soft Limit	Hard Limit	Units
Max open files	10000	10000	files

List open files của mysql process

```
# ls /proc/7727/fd -l
# lsof -p 7727
# lsof -u mysql
```

🗑 Từ Linux kernel 2.6 có 1 tham số mới là swappiness. Tham số này có giá trị từ 0 đến 100. Giá trị cao có nghĩa là nhiều paged được swapped, giá trị thấp có nghĩa là nhiều ứng dụng được giữ trong bộ nhớ RAM ngay cả khi chúng không làm gì cả (idle). Giá trị mặc định swappiness = 60

Download sử dụng script tuning

```
# ./mysqltuner.pl
# ./tuning-primer.sh
```

Troubleshooting **Waiting for table metadata lock**

## **Connection #1**

```
mysql> use test;
mysql> create table t1 (id int);
mysql> set @@autocommit=0;
mysql> select @@autocommit;
mysql> select @@tx_isolation;
mysql> select * from t1;
```

## **Connection #2**

```
mysql> use test;
```





```
mysql> alter table t1 rename to t2;
```

```
## Connection #1
```

```
mysql> show processlist;
```

Id	User	Host	db	Command	Time	State	Info
10	root	localhost	test	Query	14	Waiting for table metadata lock	alter table t1 rename to t2

```
mysql> select * from information_schema.innodb_trx\G;
```

```
mysql> commit;
```

```
mysql> select * from information_schema.innodb_trx\G;
```

Generated by Percona Configuration Wizard <http://tools.percona.com/>

### 13. MariaDB

Link download MariaDB

<https://mariadb.com/kb/en/installing-mariadb-with-yum/>

<http://downloads.mariadb.org/mariadb/repositories/>

Thêm MariaDB repo

```
## CentOS 5 32-Bit ##
```

```
# vim /etc/yum.repos.d/MariaDB.repo
```

```
[mariadb]
```

```
name = MariaDB
```

```
baseurl = http://yum.mariadb.org/5.5/centos5-x86
```

```
gpgkey=http://yum.mariadb.org/RPM-GPG-KEY-MariaDB
```

```
gpgcheck=1
```

Cài đặt MariaDB

```
# yum install MariaDB-server MariaDB-client -y
```





Nguyễn Thế Đức – Học viện CNTT Bkacad

```
# rpm -qa | grep MariaDB
```

```
MariaDB-common-5.5.39-1
```

```
MariaDB-client-5.5.39-1
```

```
MariaDB-server-5.5.39-1
```

```
# service mysql start
```

```
# netstat -penlt | grep :3306
```

```
tcp      0      0 :::3306          :::*              LISTEN      101      13240      4022/mysqld
```

Đặt mật khẩu user root

```
# mysqladmin -u root password 123456
```

```
# mysql -u root -p123456
```

