MySQL

MySQL?????????????????????????????????MySQL
AB???????2008??????????2009????????????????MySQL??Oracle?????

- [DB Admin Tips](#)
- [SQL](#)
- [Learning MySQL](#)
- [Installation](#)
- [Backup & Restore](#)
- [Partition Table](#)

DB Admin Tips

Initial DB Setup

```
/usr/bin/mysql_secure_installation
```

Change the root's password

```
ALTER USER 'root'@'localhost' IDENTIFIED BY 'MyN3wP4ssw0rd';
flush privileges;
exit;
```

Tablespace Shrink

????????????????????? Tablespace ???????????????????????

```
mysql> OPTIMIZE table foo;
```

??????????

```sh
#!/bin/sh
echo -n "MySQL username: " ; read username
echo -n "MySQL password: " ; stty -echo ; read password ; stty echo ; echo

mysql -u $username -p"$password" -NBe "SHOW DATABASES;" | grep -v 'lost+found' | while read database ; do
mysql -u $username -p"$password" -NBe "SHOW TABLE STATUS;" $database | while read name engine version
rowformat rows avgrowlength datalength maxdatalength indexlength datafree autoincrement createtime
updatetime checktime collation checksum createoptions comment ; do
if [ "$datafree" -gt 0 ] ; then
    fragmentation=$(($datafree * 100 / $datalength))
    echo "$database.$name is $fragmentation% fragmented."
    mysql -u "$username" -p"$password" -NBe "OPTIMIZE TABLE $name;" "$database"
fi
done
done
```

Mariadb Log ????

????

```
$> cat /dev/null > /var/log/mysql.log

$> mysqladmin -uroot -p flush-logs
```

?? logrotate ??????
1. ?? /etc/logrotate.d/mariadb?

```
/var/log/mariadb/*.log {
    create 660 mysql mysql
    notifempty
    daily
    rotate 5
    missingok
    minsize 1M
    maxsize 100M
    compress
    delaycompress
    sharedscripts
    olddir archive/
    createolddir 770 mysql mysql
    postrotate
       # just if mysqld is really running
       if test -x /usr/bin/mysqladmin && \
          /usr/bin/mysqladmin ping &>/dev/null
       then
          /usr/bin/mysqladmin flush-logs
       fi
    endscript
}
```

2. ?? mysql ???????

```
$> vi ~/.my.cnf


[mysqladmin]
user        = root
password      = YourPass


$> chmod 0600 ~/.my.cnf
```

????

```
mysql> show processlist;
+----+------+----------------+--------+---------+------+-------+------------------+
| Id | User | Host           | db     | Command | Time | State | Info             |
+----+------+----------------+--------+---------+------+-------+------------------+
|  3 | root | localhost      | webapp | Query   |    0 | NULL  | show processlist |
|  5 | root | localhost:61704 | webapp | Sleep  |  208 |       | NULL             |
|  6 | root | localhost:61705 | webapp | Sleep  |  208 |       | NULL             |
|  7 | root | localhost:61706 | webapp | Sleep  |  208 |       | NULL             |
+----+------+----------------+--------+---------+------+-------+------------------+
4 rows in set (0.00 sec)

mysql> show status like '%onn%';
+-------------------------+-------+
| Variable_name           | Value |
+-------------------------+-------+
| Aborted_connects        | 0     |
| Connections             | 8     |
| Max_used_connections    | 4     |
| Ssl_client_connects     | 0     |
| Ssl_connect_renegotiates | 0    |
| Ssl_finished_connects   | 0     |
| Threads_connected       | 4     |  << active connections
+-------------------------+-------+
7 rows in set (0.00 sec)
```

Connect with unix socket

```
# Locate the unix socket file
mysql -uroot -p -e "show variables like 'socket'"
Enter password:
+--------------+-------------------------+
| Variable_name | Value                  |
+--------------+-------------------------+
| socket       | /var/lib/mysql/mysql.sock |
+--------------+-------------------------+


# Connect to DB without password via unix socket
```

```
mysql -S /var/lib/mysql/mysql.sock -u root
```

User Management

??????

```
-- List All users
select host,user from mysql.user;


-- Check the privileges for specified user
SHOW GRANTS FOR 'user'@'host';
SHOW GRANTS FOR 'admin'@'%';
SHOW GRANTS FOR 'vivek'@'10.147.164.0/255.255.255.0';
```

????

1. **Data USAGE privileges** includes: SELECT, INSERT, UPDATE, DELETE, and FILE
2. **Structure privileges** includes: CREATE, ALTER, INDEX, DROP, CREATE TEMPORARY TABLES, SHOW VIEW, CREATE ROUTINE, ALTER ROUTINE, EXECUTE, CREATE VIEW, EVENT, and TRIGGER
3. **Administration privileges** includes: GRANT, SUPER, PROCESS, RELOAD, SHUTDOWN, SHOW DATABASES, LOCK TABLES, REFERENCES, REPLICATION CLIENT, REPLICATION SLAVE, and CREATE USER
4. **SSL privileges** includes: REQUIRE NONE, REQUIRE SSL, REQUIRE X509
5. ALL PRIVILEGES: Shortcut to grants all privileges to a mysql user account.

???? superuser ??

```
CREATE USER 'admin'@'%' IDENTIFIED BY 'the_secure_password';
-- Or
CREATE USER 'admin'@'10.147.164.0/255.255.255.0' IDENTIFIED BY 'the_secure_password';


GRANT ALL PRIVILEGES ON *.* TO 'admin'@'%';
-- Or
GRANT ALL PRIVILEGES ON *.* TO 'admin'@'10.147.164.0/255.255.255.0';
```

??????? DDL

```
SHOW CREATE USER admin;
SHOW CREATE USER vivek;
```

????

```
REVOKE ALL PRIVILEGES ON *.* FROM 'user'@'%';
```

Table Management

Table's DDL

```
show create table <table_name>;
```

Copy Table without data

```
create table new-table like old-table;
```

SQL

Online Tutorials

- [MySQL Cheat Sheet](#)

WHERE

```sql
SELECT name, wins
FROM golfers
WHERE wins = 1;


-- Comparison
SELECT name
FROM golfers
WHERE name = 'George';
-- <>: tests whether two values are not equal
SELECT name, wins
FROM golfers
WHERE wins <> 1;
--
SELECT name, wins
FROM golfers
WHERE wins < 1;
-- <=: tests whether the first value is less than or equal to the second
SELECT name, wins
FROM golfers
WHERE wins <= 1;
-- >=: tests whether the first value is greater than or equal to the second
SELECT name, wins
FROM golfers
WHERE wins >= 1;


-- Null
SELECT name, rounds_played
FROM golfers
WHERE rounds_played IS NULL;
```

```sql
-- Range
SELECT name, best
FROM golfers
WHERE best BETWEEN 67 AND 73;


-- Membership
SELECT name, best
FROM golfers
WHERE best IN (65, 67, 69, 71);


-- Pattern Match
-- _: an underscore represents a single unknown character
SELECT name, rounds_played
FROM golfers
WHERE rounds_played LIKE '2_';
-- %: a percentage sign represents zero or more unknown characters
SELECT name, rounds_played
FROM golfers
WHERE name LIKE 'G%';


-- Combining Multiple Predicates with AND and OR
SELECT name, best, worst, average
FROM golfers
WHERE best < 70 AND worst < 96;
--
SELECT name, best, worst, average
FROM golfers
WHERE best < 70 OR worst < 96;
--
SELECT name, average, worst, rounds_played
FROM golfers
WHERE average < 85 OR worst < 95 AND rounds_played BETWEEN 19 AND 23;
--
SELECT name, average, worst, rounds_played
FROM golfers
WHERE (average < 85 OR worst < 95) AND rounds_played BETWEEN 19 AND 23;


-- Excluding Results with NOT
SELECT name
```

```sql
FROM golfers
WHERE name NOT LIKE 'R%';
--
SELECT name, rounds_played
FROM golfers
WHERE rounds_played IS NOT NULL;
--
SELECT name, average, best, wins
FROM golfers
WHERE NOT (average < 80 AND best < 70) OR wins = 9;
--
SELECT name, average, best, wins
FROM golfers
WHERE NOT ((average < 80 AND best < 70) OR wins = 9);
--
SELECT name
FROM golfers
WHERE name NOT = 'Grady';
```

Learning MySQL

MySQL Backup

- [How to Backup/Restore MySQL/MariaDB and PostgreSQL Using 'Automysqlbackup' and 'Autopostgresqlbackup' Tools](#)
- [archlinux | MariaDB](#)

Partition Table

- [MySQL ?????? partition ????](#)
- [???? Mysql Partition ????](#)
- [MariaDB Partition ??](#)
- [PARTITION Maintenance in MySQL](#)

Time-Series in MariaDB

- [Using a Staging Table for Efficient MySQL Data Warehouse Ingestion](#)
- [Summary Tables in MySQL](#)

DB Optimization

- [13 Tips for Tuning and Optimizing MySQL and MariaDB](#)

DB Administration

- [20 mysqladmin Commands for MYSQL/MariaDB Database Administration](#)
- [Upgrading Uber's MySQL Fleet to version 8.0 | Uber Blog](#)

Synchronize DBs

- [Synchronize databases more easily with open source tools | Opensource.com](#)
- [Apache SeaTunnel | Apache SeaTunnel](#)

Installation

Install on CentOS 7

URL:
https://downloads.mariadb.org/mariadb/repositories/#mirror=mephi&distro=CentOS&distro_release=centos7-amd64--centos7

/etc/yum.repos.d/MariaDB.repo

```
# MariaDB 10.5 CentOS repository list - created 2021-05-08 02:51 UTC
# http://downloads.mariadb.org/mariadb/repositories/
[mariadb]
name = MariaDB
baseurl = http://yum.mariadb.org/10.5/centos7-amd64
gpgkey=https://yum.mariadb.org/RPM-GPG-KEY-MariaDB
gpgcheck=1
```

????

```
yum install MariaDB-server MariaDB-client
```

Starting MariaDB

```
systemctl start mariadb.service
systemctl enable mariadb.service
systemctl status mariadb.service
```

Securing MariaDB

```
mysql_secure_installation
```

OS Optimization

Linux Kernel Settings – IO Scheduler

```
# A temporary change can be done by issuing the following command
echo noop > /sys/block/sda/queue/scheduler


# To make it persistent, you'll need to configure it in GRUB's configuration
# file as shown below in /etc/default/grub , rebuild GRUB, and reboot the system.
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash elevator=noop"
```

Resource Limits – Open Files Limit, Core File Size

```
# /etc/security/limits.conf
mysql soft nofile 65535
mysql hard nofile 65535
mysql soft core unlimited
mysql hard core unlimited
```

Configure Swappiness

```
# /etc/sysctl.conf
vm.swappiness = 1
```

Filesystem Optimizations

```
# /etc/fstab
/dev/sdb /var/lib/mysql ext4 defaults,noatime 0 0
```

DB Optimization

Backup & Restore

Backup

Script#1:

```bash
#!/bin/bash

# 🔲🔲🔲🔲🔲🔲🔲🔲
mysql_user="USER" #MySQL 🔲🔲🔲🔲
mysql_password="PASSWORD" #MySQL 🔲🔲🔲🔲🔲
mysql_host="localhost"
mysql_port="3306"
mysql_charset="utf8" #MySQL 🔲🔲
backup_db_arr=("db1" "db2") #🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲 🔲("db1" "db2" "db3")
backup_location=/var/www/mysql  #🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲"/"🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲
expire_backup_delete="ON" #🔲🔲🔲🔲🔲🔲🔲🔲 ON 🔲🔲🔲 OFF 🔲🔲🔲
expire_days=3 #🔲🔲🔲🔲🔲 🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲 expire_backup_delete 🔲🔲🔲🔲

# 🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲
backup_time=`date +%Y%m%d%H%M`  #🔲🔲🔲🔲🔲🔲🔲
backup_Ymd=`date +%Y-%m-%d` #🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲
backup_3ago=`date -d '3 days ago' +%Y-%m-%d` #3 🔲🔲🔲🔲🔲
backup_dir=$backup_location/$backup_Ymd  #🔲🔲🔲🔲🔲🔲🔲
welcome_msg="Welcome to use MySQL backup tools!" #🔲🔲🔲

# 🔲🔲 MYSQL 🔲🔲🔲🔲,mysql 🔲🔲🔲🔲🔲🔲🔲🔲🔲
mysql_ps=`ps -ef |grep mysql |wc -l`
mysql_listen=`netstat -an |grep LISTEN |grep $mysql_port|wc -l`
if [ [$mysql_ps == 0] -o [$mysql_listen == 0] ]; then
    echo "ERROR:MySQL is not running! backup stop!"
    exit
else
    echo $welcome_msg
fi

# 🔲🔲🔲 mysql 🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲
```

```
mysql -h$mysql_host -P$mysql_port -u$mysql_user -p$mysql_password <<end
use mysql;
select host,user from user where user='root' and host='localhost';
exit
end

flag=`echo $?`
if [ $flag != "0" ]; then
     echo "ERROR:Can't connect mysql server! backup stop!"
     exit
else
     echo "MySQL connect ok! Please wait......"
     # □□□□□□□□□□□□□□□□□□□□□□□□□□□□
     if [ "$backup_db_arr" != "" ];then
          #dbnames=$(cut -d ',' -f1-5 $backup_database)
          #echo "arr is (${backup_db_arr[@]})"
          for dbname in ${backup_db_arr[@]}
          do
               echo "database $dbname backup start..."
               `mkdir -p $backup_dir`
               `mysqldump -h$mysql_host -P$mysql_port -u$mysql_user -p$mysql_password $dbname --default-
character-set=$mysql_charset | gzip > $backup_dir/$dbname-$backup_time.sql.gz`
               flag=`echo $?`
               if [ $flag == "0" ];then
                    echo "database $dbname success backup to $backup_dir/$dbname-$backup_time.sql.gz"
               else
                    echo "database $dbname backup fail!"
               fi

          done
     else
          echo "ERROR:No database to backup! backup stop"
          exit
     fi
     # □□□□□□□□□□□□□□□□□□□□□
     if [ "$expire_backup_delete" == "ON" -a  "$backup_location" != "" ];then
          #`find $backup_location/ -type d -o -type f -ctime +$expire_days -exec rm -rf {} \;`
          `find $backup_location/ -type d -mtime +$expire_days | xargs rm -rf`
          echo "Expired backup data delete complete!"
     fi
```

```
        echo "All database backup success! Thank you!"
        exit
  fi
```

Restore

How to Restore Large MySQL Database

> " When we restore a large database, usually we come across two errors as
> shown below: "MySQL server has gone away" & "Lost connection to MySQL
> server during query"

Here, we consider restoring a **16GB** database:

When restoring huge database, we need to set the values as specified below:

- connect_timeout = **3600**
- max_allowed_packet = **1024**M
- net_buffer_length = **1000000**
- wait_timeout = **86400** (which means 24 hours)
- interactive_timeout = **86400** (which means 24 hours)
- net_write_timeout = **3600**
- net_read_timeout = **3600**

```
mysql -u root -p -max_allowed_packet=1024M \
 -connect_timeout=3600 \
 -net_buffer_length=1000000 \
 -wait_timeout=86400 \
 -interactive_timeout=86400 \
 -net_write_timeout=3600 \
 -net_read_timeout=3600 \
 database-name < database.sql
```

Partition Table

Check if partition is supported

```
> show plugins;
```

Create partition table

```
create table mm_tx_new (
  start_time    timestamp(6) not null default '0000-00-00 00:00:00.000000',
  end_time      timestamp(6),
  hostname      varchar(20),
  servername    varchar(20) not null default 'noservername',
  tx_id         varchar(15),
  user_opi      varchar(20),
  tx_resp       integer,
  primary key(start_time,servername)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
PARTITION BY RANGE  ( FLOOR(UNIX_TIMESTAMP(start_time)) )
(PARTITION p202105 VALUES LESS THAN ( FLOOR(UNIX_TIMESTAMP('2021-06-01 00:00:00.000000')) ) ENGINE =
InnoDB,
 PARTITION p202106 VALUES LESS THAN ( FLOOR(UNIX_TIMESTAMP('2021-07-01 00:00:00.000000')) ) ENGINE =
InnoDB,
 PARTITION p202107 VALUES LESS THAN ( FLOOR(UNIX_TIMESTAMP('2021-08-01 00:00:00.000000')) ) ENGINE =
InnoDB,
 PARTITION p202108 VALUES LESS THAN ( FLOOR(UNIX_TIMESTAMP('2021-09-01 00:00:00.000000')) ) ENGINE =
InnoDB,
 PARTITION p202109 VALUES LESS THAN ( FLOOR(UNIX_TIMESTAMP('2021-10-01 00:00:00.000000')) ) ENGINE =
InnoDB,
 PARTITION p202110 VALUES LESS THAN ( FLOOR(UNIX_TIMESTAMP('2021-11-01 00:00:00.000000')) ) ENGINE =
InnoDB,
 PARTITION p202111 VALUES LESS THAN ( FLOOR(UNIX_TIMESTAMP('2021-12-01 00:00:00.000000')) ) ENGINE =
InnoDB,
 PARTITION p202112 VALUES LESS THAN ( FLOOR(UNIX_TIMESTAMP('2022-01-01 00:00:00.000000')) ) ENGINE =
InnoDB,
 PARTITION p202201 VALUES LESS THAN ( FLOOR(UNIX_TIMESTAMP('2022-02-01 00:00:00.000000')) ) ENGINE =
```

```
InnoDB)
;
```

Check partition table

```
> show create table mm_tx_new;
```

Add new partition for specified table

```
ALTER TABLE `mm_tx_new` REORGANIZE PARTITION pMax INTO (
    PARTITION p202501 VALUES LESS THAN ( FLOOR(UNIX_TIMESTAMP('2025-02-01 00:00:00.000000')) ) ENGINE
= InnoDB,
    PARTITION pMax VALUES LESS THAN MAXVALUE
);
```

> " ??: pMax ????????? partition ???????? pMax
> ????????????????????????

Delete the partition

Purge the data in the specified partition

```
ALTER TABLE <schema.table> TRUNCATE PARTITION <partition-name>
```