# LINUX 安装 mysql

前言：

各软件包下载地址：

Apache下载地址：

<http://httpd.apache.org/download.cgi>

http://archive.apache.org/dist/httpd/

Mysql下载地址：

<https://dev.mysql.com/downloads/mysql/>

<http://mirror.bit.edu.cn/mysql/Downloads/>

http://ftp.jaist.ac.jp/pub/mysql/

Php下载地址：

<http://php.net/downloads.php>

<http://php.net/releases/>

Php扩展下载：

<http://pecl.php.net/>

<http://windows.php.net/downloads/pecl/releases/>

<http://downloads.php.net/pierre/>

依赖包下载地址：<http://download.chinaunix.net/>

Rpm包下载地址：

<http://www.rpmfind.net/linux/>

[https://pkgs.org/download/](https://pkgs.org/download/pcre)

开源软件镜像服务下载地址：<http://mirror.bit.edu.cn/web/>

Linux系统下载地址：

<https://www.linux.org/pages/download/>

<http://archive.kernel.org/centos-vault/>

Libxml2下载地址：

<http://xmlsoft.org/sources/>

## 一，源码安装

测试安装环境： CentOS6.8 64位。

安装位置： /usr/local/

软件包存放位置： /home/

系统支持：gcc，gcc-c++，make编译环境

如无gcc，gcc-c++，make编译环境，则按以下步骤下载安装编译工具包（已安装则忽略）

1. 使用rpm方式安装编译工具包

下载以下软件包（操作系统不同，软件包也不一样）:

ppl-0.10.2-11.el6.x86\_64.rpm

cloog-ppl-0.15.7-1.2.el6.x86\_64.rpm

cpp-4.4.7-17.el6.x86\_64.rpm

mpfr-2.4.1-6.el6.x86\_64.rpm

gcc-4.4.7-17.el6.x86\_64.rpm

gcc-c++-4.4.7-17.el6.x86\_64.rpm

gcc-java-4.4.7-17.el6.x86\_64.rpm

libstdc++-devel-4.4.7-17.el6.x86\_64.rpm

安装顺序：

1，rpm -ivh ppl-0.10.2-11.el6.x86\_64.rpm

2，rpm -ivh cloog-ppl-0.15.7-1.2.el6.x86\_64.rpm

3，rpm -ivh mpfr-2.4.1-6.el6.x86\_64.rpm

4，rpm -ivh cpp-4.4.7-17.el6.x86\_64.rpm

5，rpm -ivh gcc-4.4.7-17.el6.x86\_64.rpm

6，rpm -ivh gcc-c++-4.4.7-17.el6.x86\_64.rpm

7，rpm -ivh libstdc++-devel-4.4.7-17.el6.x86\_64.rpm

8，rpm -ivh gcc-c++-4.4.7-17.el6.x86\_64.rpm

9，rpm -ivh gcc-java-4.4.7-17.el6.x86\_64.rpm(编译java需要，可选)

安装成功后，写一个c语言（hello.c）文件：

内容如下: #include<stdio.h>

main(){

printf("hello word\n");

}

保存退出。

执行gcc hello.c，在查看目录，若出现a.out文件，则运行./a.out，可运行表示安装成功。

B) 使用yum方式安装编译工具包

前提：电脑必须可以联网。

执行命令: yum -y install gcc gcc-c++ make

### 准备安装mysql所需要的软件包

**zlib-1.2.11.tar.gz**

**cmake-3.0.2.tar.gz**

**boost\_1\_59\_0.tar.gz**

**bison-2.4.1-5.el6.x86\_64.rpm**

**bison-devel-2.4.1-5.el6.x86\_64.rpm**

**ncurses-5.7-4.20090207.el6.x86\_64.rpm**

**ncurses-devel-5.7-4.20090207.el6.x86\_64.rpm**

**mysql-5.7.17.tar.gz**

### 安装zlib

**Shell># cd /home/**

**Shell># tar -zxvf zlib-1.2.11.tar.gz**

**Shell># cd zlib-1.2.11.tar.gz**

**Shell># ./configure**

**Shell># make && make install**

注意：

1，若要指定zlib安装路径的话，则在/configure 之后追加 --prefix=安装路径，例如：./configure --prefix=/usr/local/zlib

2，如果系统是64位，需要使用64位元的方法编译安装zlib:

CFLAGS=”-03 -fPIC” ./configusr --prefix=/usr/local/zlib

3，若想知道./configure 有什么参数可以配置? 可以使用：

./configure --help【适用所有源码编译的软件】

4，若要指定安装路径的话，记得先卸载系统自带的zlib库，并且卸载时间必须在安装zlib时已经使用make，但未使用make install 的时候卸载。

5，指定安装好zlib后，在编辑/etc/ld.so.conf 文件，在下面加入一行/usr/local/zlib/lib，保存退出。再使用ldconfig更新缓存即可。

### 安装cmake（跨平台的安装[编译]工具）

简介：mysql-5.5.x版本及以上 需要使用cmake编译扩展。【更先进的configure】

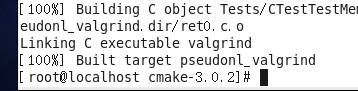
**Shell># cd /home/**

**Shell># tar -zxvf cmake-3.0.2.tar.gz**

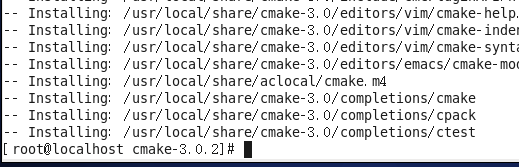
**Shell># cd cmake-3.0.2**

**Shell># ./bootstrap**

**Shell># gmake**



**Shell># gmake install**



注：若检测编译环境失败，可能原因是关联的程序包未安装。解决方法：安装所需的程序包后，将cmake-3.0.2/目录下的CMakeCache.txt删掉，再从./bootstrap开始操作。

或者 重新安装低版本的cmake。

### 安装boost【可选，mysql5.7以上版本必须】

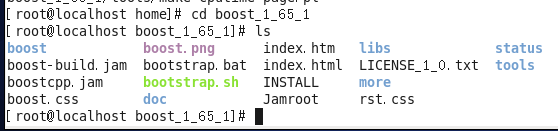
注：若只安装mysql，则只需解压，不必编译。编译之后可能导致mysql安装失败。

**Shell># cd /home/**

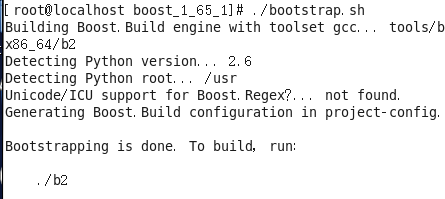
**Shell># tar -zxvf boost\_1\_59\_0.tar.gz -C /usr/local/**

**以下安装命令不是必须执行的：**

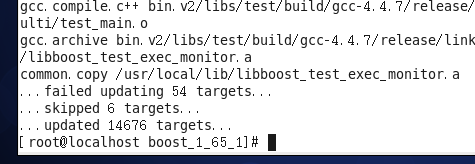
**Shell># cd /usr/local/boost-1\_59\_0/**



**Shell># ./ bootstrap.sh [--prefix=/usr/local/boost]**



**Shell># ./b2 install 【不是必须】**



**安装boost.build**

**Shell># cd /usr/local/boost\_1\_59\_0/tools/build/**

**Shell># ./bootstrap.sh**

**Shell># ./b2 install --prefix=/usr/local/boost**

### 安装ncurses和ncurses-devel 字符终端处理库（已安装则忽略）

**方案一：**

**Shell># cd /home/**

**Shell># rpm -ivh ncurses-5.7-4.20090207.el6.x86\_64.rpm**

**Shell># rpm -ivh ncurses-devel-5.7-4.20090207.el6.x86\_64.rpm**

**方案二：**

**Shell># yum -y install ncurses ncurses-devel**

**方案三：**

**Shell># cd /home/**

**Shell># tar -zxvf ncurses-6.0.tar.gz**

**Shell># cd ncurses-6.0**

**Shell># ./configure --with-shared \**

**> --without-debug \**

**> --without-ada \**

**> --enable-overwrite**

**Shell># make && make install**

### 安装bison 和 bison-devel（可选，C/C++语法分析器）

**方案一：**

**Shell># cd /home/**

**Shell># rpm -ivh bison-2.4.1-5.el6.x86\_64.rpm**

**Shell># rpm -ivh bison-devel-2.4.1-5.el6.x86\_64.rpm**

**方案二：**

**Shell># yum -y install bison bison-devel**

**方案三：**

**Shell># cd /home/**

**Shell># tar -zxvf bison-3.0.4.tar.gz**

**Shell># cd bison-3.0.4**

**Shell># ./configure**

**Shell># make && make install**

### 安装mysql

注：mysql5.7的源码安装包有两种：

1. 没有携带boost的源码包。 如：mysql-5.7.16.tar.gz
2. 携带boost的源码包。 如：mysql-boost-5.7.17.tar.gz

两种源码包的方式安装都一样，只不过第二种源码包则无需操作第4步。

**Shell># cd /home**

**Shell># tar -zxvf mysql-5.7.16.tar.gz**

**Shell># cd mysql-5.7.16**

**Shell># groupadd mysql**

**Shell># useradd -g mysql -s /bin/false mysql**

解释：创建一个mysql用户，并且不允许登陆系统。【可以选择在mysql安装完成后，初始化数据库之前创建】

**Shell># cmake . \**

**> -DCMAKE\_INSTALL\_PREFIX=/usr/local/mysql \**

**> -DMYSQL\_DATADIR=/usr/local/mysql/data \**

**> -DWITH\_BOOST=/usr/local/boost\_1\_59\_0 \**

**> -DSYSCONFDIR=/etc \**

**> -DDEFAULT\_CHARSET=utf8 \**

**> -DDEFAULT\_COLLATION=utf8\_general\_ci \**

解释：

-DCMAKE\_INSTALL\_PREFIX mysql安装位置

-DMYSQL\_DATADIR 数据存放位置

-DWITH\_BOOST boost安装位置

-DSYSCONFDIR my.cnf配置文件目录

-DDEFAULT\_CHARSET 默认字符集

-DDEFAULT\_COLLATION 默认校队字符集

若未安装boost，可增加以下参数，会自动下载：

-DDOWNLOAD\_BOOST=1 开启自动下载boost

-DWITH\_BOOST=/usr/local/boost boost安装位置

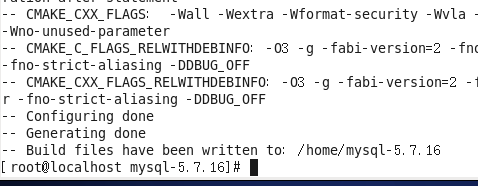
若安装的是自带boost的mysql安装包：

-DWITH\_BOOST=boost boost安装位置

可选配置：

-DWITH\_INNOBASE\_STORAGE\_ENGINE=1 -DWITH\_PARTITION\_STORAGE\_ENGINE=1 -DWITH\_FEDERATED\_STORAGE\_ENGINE=1

-DWITH\_BLACKHOLE\_STORAGE\_ENGINE=1 -DWITH\_MYISAM\_STORAGE\_ENGINE=1



**注意：若在cmake 过程中出现错误的话，需要解决软件依赖问题（确保安装Boost【5.7版本需要】和ncurses-devel）.**

**解决依赖后，删除mysql解压目录下的CMakeCache.txt文件，再cmake..**

**如：**

**Shell># cd /home/mysql-5.7.16/**

**Shell># make clean**

**Shell># rm -rf CMakeCache.txt**

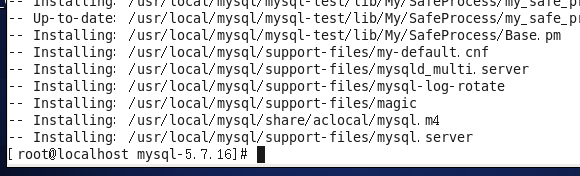
**Shell># rem -rf /etc/my.cnf**

**Shell># make -j `grep processor /proc/cpuinfo | wc -l` 【可选】**

**Shell># make && make install**

解释：

-j参数表示根据CPU核数指定编译时的线程数，可以加快编译速度。默认为1个线程编译，经测试单核CPU，1G的内存，编译完需要将近1个小时。



**Shell># cd /usr/local/**

**Shell> # chmod +x mysql**

**Shell># chown -R mysql.mysql mysql**

**Shell># cd mysql**

解释：设置mysql目录权限

# 注意：MySQL 5.7之前的版本执行这个脚本初始化系统数据库

**Shell> ./bin/mysql\_install\_db \**

**> --user=mysql \**

**> --basedir=/usr/local/mysql \**

**> --datadir=/usr/local/mysql/data**

# 5.7之后版本初始系统数据库脚本（本文使用此方式初始化）

**Shell> ./bin/mysqld \**

**> --initialize-insecure \**

**> --user=mysql \**

**> --basedir=/usr/local/mysql \**

**> --datadir=/usr/local/mysql/data**

解释：

--basedir mysql安装路径

--datadir mysql数据存放路径。该目录下不能有任何文件

--user 执行mysql的用户。若登陆的是mysql用户，可以不用填写。

--initialize 会生成一个随机密码(~/.mysql\_secret)，密码在mysql日志中会显示。并且在登陆mysql后，会要求立即重新修改密码，才可以操作mysql（5.7版本的一个新安全机制）。

命令如下：ALTER USER 'root'@'localhost' IDENTIFIED BY 'new\_password';

--initialize-insecure 不会生成密码，但会产生一条警告，并记录mysql日志中。

注：之前版本mysql\_install\_db是在mysql\_basedir/script下，5.7放在了mysql\_install\_db/bin目录下，且已被废弃。

**Shell> ./bin/mysql\_ssl\_rsa\_setup \**

**> --user=mysql \**

**> --basedir=/usr/local/mysql \**

**> --datadir=/usr/local/mysql/data**

**Shell> chown -R root .**

**Shell> chown -R mysql data**

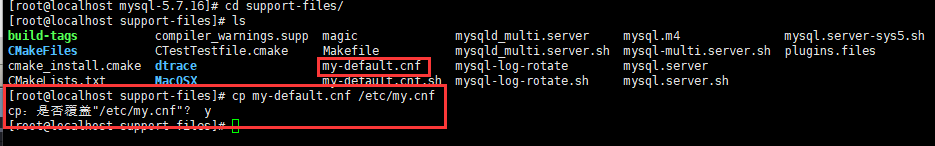
解释：重新更改mysql目录的所有者，预防其他用户误删除文件，导致数据库损坏。

**Shell># cd /home/mysql-5.7.16/support-files/**

**Shell># cp my-default.cnf /etc/my.cnf**

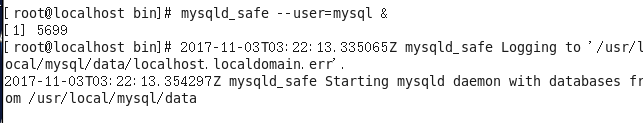
**Shell># y 【若提示是否覆盖，则按 y】**

解释：编译完成，复制配置文件到/etc目录。



### 启动mysql

**Shell># /usr/local/mysql/bin/mysqld\_safe --user=mysql &**

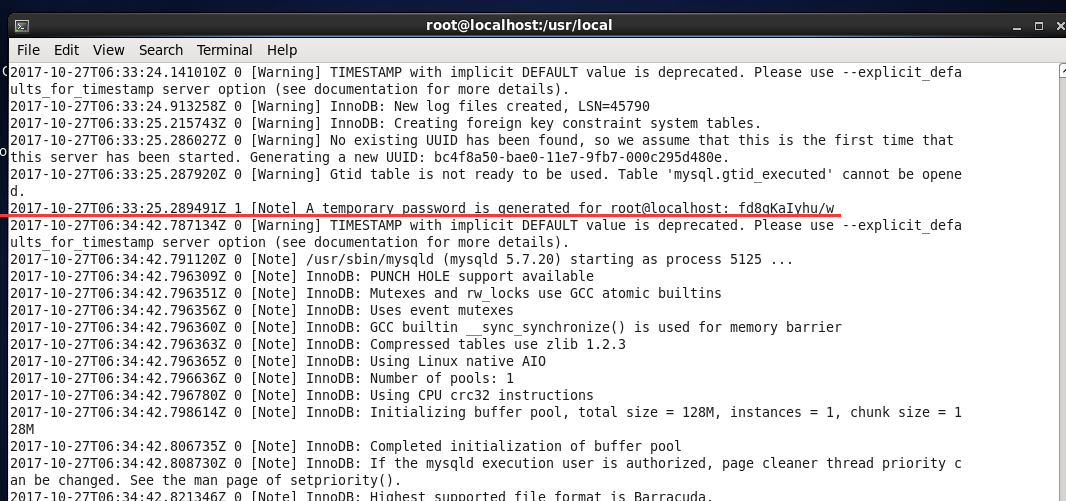


**Shell># netstat -luntp | grep mysql**

**Shell># /usr/local/mysql/bin/mysql -u root -p**

**注：若使用的是--initialize-insecure的话，则无需输入密码。**

**使用的是--initialize的话，需要查询mysqld.log文件，找出密码。**



--initialize初始化的数据库：

会提示立即修改密码，使用如下指令修改密码：

**Mysql> ALTER USER 'root'@'localhost' \**

**> IDENTIFIED BY 'new\_password';**

修改完成之后，才可以操作数据库

以下是mysql修改密码的操作：

**Mysql> use mysql;**

**Mysql> SELECT Hose,User FROM user;**

**Mysql> UPDATE user SET \**

**> authentication\_string \**

**> password=password(‘new\_password’) \**

**> where User=’root’;**

**Mysql> flush privileges;**

**注：可以使用set password = password('xxxxxxxx'); （新密码必须包含大小写字母，特殊字符，数字）命令，设置当前登录用户的密码。**

**修改完成后，记得 flush privileges 刷新权限表，使密码生效。**

### 配置开机启动

**Shell># cp /usr/local/mysql/support-files/mysql.server \**

**> /etc/init.d/mysqld**

**Shell># chmod +x /etc/init.d/mysqld**

**Shell># chkconfig --add mysqld**

**Shell># chkconfig mysqld on**

**Shell># vim /etc/profile**

**找到 export PATH 该行，在上面加入：**

**PATH=/usr/local/mysql/bin:$PATH**

**保存退出。**

**Shell># source /etc/profile**

**Shell># echo $PATH**

**解释：查看环境变量是否配置成功。**

### 卸载mysql

步骤：

1，删除cmake . 指定mysql的安装路径。如:/usr/local/mysql

2，删除mysql解压目录

3，删除mysql的配置文件 my.cnf

**Shell># rm -rf /usr/local/mysql**

**Shell># rm -rf /home/mysql-5.7.16**

**Shell># rm -rf /etc/my.cnf**

### 源码软件安装错误解决

1. 已执行configure操作

》根据正确的参数重新configure即可

2，已经执行configure、make操作

》删除解压后的目录，重新解压、configure、make

3，已经执行configure、make、make install

》删除安装后的文件（有指定安装目录情况/usr/local/httpd24）

》删除解压后的目录

》重新解压，重新configure，重新make，重新make install

### mysql编译错误总结

1，使用cmake 指令发生错误

Shell># make clean # 清除make缓存

Shell>rm -rf CMakeCache.txt # 删除旧配置信息

在重新使用camke

2，make版本过低

出现以下错误代码其中一种：

A)make: Fatal error in reader: Makefile, line 18:

Badly formed macro assignment

B)make: file `Makefile' line 18: Must be a separator (:

C)pthread.h: No such file or directory

则必须将make版本升级 为GNU make。

3，boost版本过低

出现以下错误中的一种：

A) "sql\_yacc.yy", line xxx fatal: default action causes potential...

B) sql\_yacc.yy:#####: fatal error: maximum table size (32767) exceeded

则必须安装高版本的boost，若安装高版本的boost出现第二种错误，则需要重新安装低一点的版本。

4，找不到c/c++编译器

自行定义C和C++编译器，可以定义 CC和使用CXX环境变量。

环境变量文件：/etc/profile

加入：

CC=gcc

CXX=g++

export CC CXX

### 更多源码编译mysql参考

**<http://www.linuxidc.com/Linux/2016-04/130075.htm>**

**<http://www.cnblogs.com/etangyushan/p/4335241.html>**

**<http://www.cnblogs.com/eagle6688/p/5840030.html>**

**<http://www.cnblogs.com/galengao/p/5755788.html>**

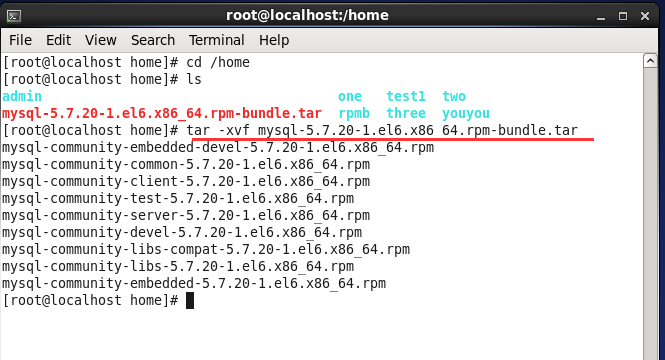
**<http://www.jb51.net/article/103492.htm>**

## 二，rpm包方式安装

### 准备rpm安装包

mysql-5.7.20-1.el6.x86\_64.rpm-bundle.tar

解压：shell># tar -zxvf mysql-5.7.20-1.el6.x86\_64.rpm-bundle.tar



只需要其中的：

mysql-community-common-5.7.20-1.el6.x86\_64.rpm

mysql-community-libs-5.7.20-1.el6.x86\_64.rpm

mysql-community-client-5.7.20-1.el6.x86\_64.rpm

mysql-community-server-5.7.20-1.el6.x86\_64.rpm

perl-DBD-MySQL-4.013-3.el6.x86\_64.rpm

perl-devel-5.10.1-144.el6.x86\_64.rpm

perl-Time-HiRes-1.9721-144.el6.x86\_64.rpm

libaio-0.3.107-10.el6.x86\_64.rpm

numactl-2.0.9-2.el6.x86\_64.rpm

numactl-devel-2.0.9-2.el6.x86\_64.rpm

### 创建mysql用户

**Shell># groupadd mysql**

**Shell># useradd -g mysql -s /bin/nologin mysql**

### 卸载旧版mysql（若无，则忽略）

**Shell># rpm -qa | grep -i mysql**

**若查询到，则删除：**

**第一种方式删除：rpm -e mysql-xxxxxx --nodeps**

**第二种方式删除：yum -y remove mysql-xxxxxxx**

**Shell># rpm -qa | grep -i mysql**

**查看是否删除成功！**

**Shell># find / -name mysql**

**Shell># rm -rf xxxxxx**

**删除相关mysql的目录和文件**

### 安装mysql

安装包顺序如下：

**Shell># rpm -ivh libaio-0.3.107-10.el6.x86\_64.rpm**

**Shell># rpm -ivh numactl-2.0.9-2.el6.x86\_64.rpm**

**Shell># rpm -ivh numactl-devel-2.0.9-2.el6.x86\_64.rpm**

**Shell># rpm -ivh perl-devel-5.10.1-144.el6.x86\_64.rpm**

**Shell># rpm -ivh perl-Time-HiRes-1.9721-144.el6.x86\_64.rpm**

**Shell># rpm -ivh perl-DBD-MySQL-4.013-3.el6.x86\_64.rpm**

**Shell># rpm \**

**> -ivh mysql-community-common-5.7.20-1.el6.x86\_64.rpm**

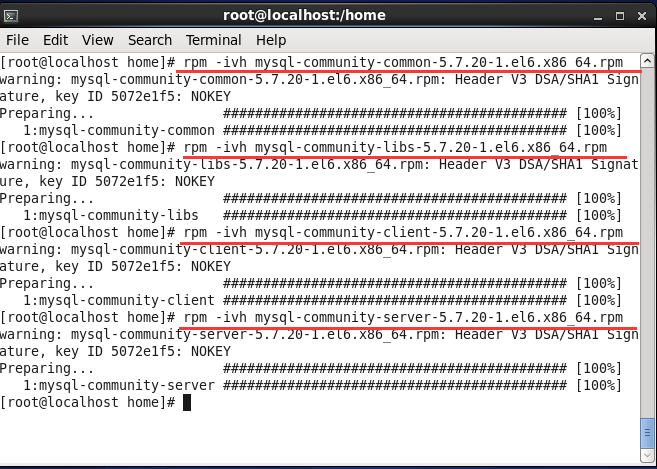
**Shell># rpm -ivh mysql-community-libs-5.7.20-1.el6.x86\_64.rpm**

**Shell># rpm \**

**> -ivh mysql-community-client-5.7.20-1.el6.x86\_64.rpm**

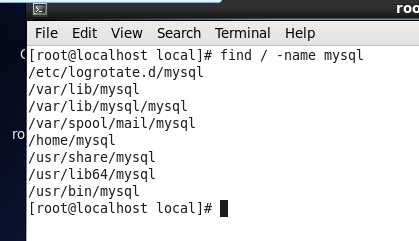
**Shell># rpm \**

**> -ivh mysql-community-server-5.7.20-1.el6.x86\_64.rpm**



### 设置mysql目录权限

**Shell># find / -name mysql**



各目录含义：

/etc/logrotate.d/mysql mysql日志文件配置

/var/lib/mysql 数据库目录（包括mysql启动引导文件）

/var/lib/mysql/mysql mysql的运行程序

/var/spool/mail/mysql mysql的邮箱接收地址

/usr/share/mysql mysql.server命令及配置文件

/usr/lib64/mysql mysql依赖程序

/usr/bin/ mysql相关命令存放地址(mysqldump等)

/etc/my.cnf mysql配置文件

**修改权限：**

**Shell># chown -R mysql.mysql /var/lib/mysql**

**Shell># chown -R mysql /usr/share/mysql**

**Shell># chgrp -R mysql /var/spool/mail/mysql**

**预防后续初始化数据库报错。**

### 初始化mysql数据库

注：新版本的mysql在初始化数据库的时候会生成一个随机密码。

为了保证数据库目录全部文件的所有者为 mysql 登陆用户，如果你是以 root 身份运行 mysql 服务，需要执行下面的命令初始化

mysqld --initialize --user=mysql

如果是以 mysql 身份运行，则可以去掉 --user 选项。

解释：--initialize 选项默认以“安全”模式来初始化，则会为root用户生成一个密码，并将该密码标记为过期，登陆后的第一件事必须更改密码后，才可操作数据库。

而是用 --initialize-insecure命令则不适用安全模式，则不会为root用户生成一个密码。

可选在初始化数据库的时候指定“数据库安装目录”和“数据目录”

例如：

**Shell># mysqld --initialize --user=mysql \**

**> --basedir=/opt/mysql/mysql**

**> --datadir=/opt/mysql/mysql/data**

或者将相关选项放在配置文件中，并将该文件名称传递给mysqld。

假设相关选项文件名是：

**/opt/mysql/mysql/etc/my.cnf**

**[mysqld]**

**basedir=/opt/mysql/mysql**

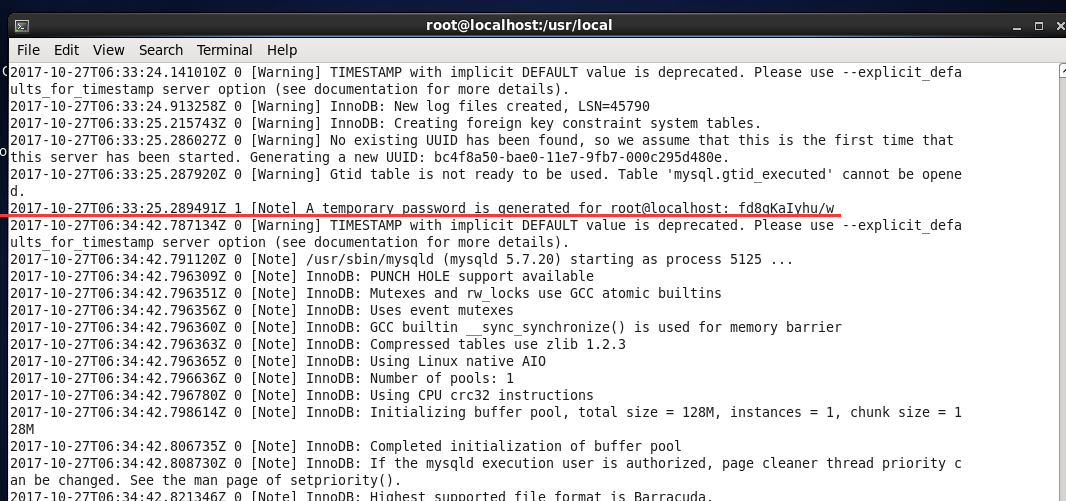
**datadir=/opt/mysql/mysql/data**

初始化命令如下：

**Shell># mysqld --initialize --user=mysql \**

**> --defaults-file=/opt/mysql/mysql/etc/my.cnf**

使用 --initialize初始化的，会生成一个root账户密码，密码存放于mysql日志文件中。日志文件位置：/var/log/mysqld.log



另一种初始化数据库方式：

命令：mysql\_install\_db --datadir=/var/lib/mysql

注意：必须指定datadir,执行后会生成~/.mysql\_secret密码文件

### 重新修改mysql文件权限

**Shell># chown -R root /etc/logrotate.d/mysql**

**mysql日志文件配置**

**Shell># chown -R root /var/lib/mysql/mysql**

**mysql的运行程序**

**Shell># chown -R root /var/spool/mail/mysql**

**mysql的邮箱接收地址**

**Shell># chown -R root /usr/share/mysql**

**mysql.server命令及配置文件**

**Shell># chown -R root /usr/lib64/mysql**

**mysql依赖程序**

**Shell># chown -R root /usr/bin/**

**mysql相关命令存放地址(mysqldump等)**

**Shell># chown -R root /etc/my.cnf**

**mysql配置文件**

### 启动mysql

**启动命令：**

**Shell># service mysqld start**

**Shell># /etc/init.d/mysql start**

**Shell># safe\_mysqld &**

**停止命令：**

**Shell># service mysqld stop**

**Shell># /etc/rc.d/init.d/mysql stop**

**Shell># mysqladmin shutdown**

**重启命令：**

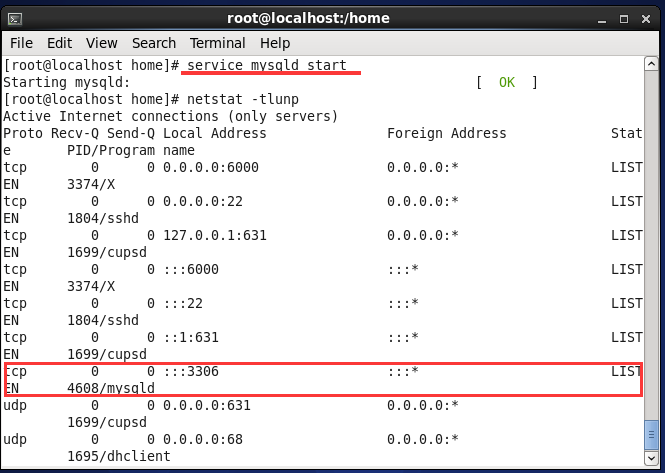
**Shell># service mysqld restart**

**Shell># /etc/rc.d/init.d/mysqld restart**

**查看状态：**

**Shell># service mysqld status**

**Shell># /etc/rc.d/init.d/mysqld statuss**



**链接数据库：**

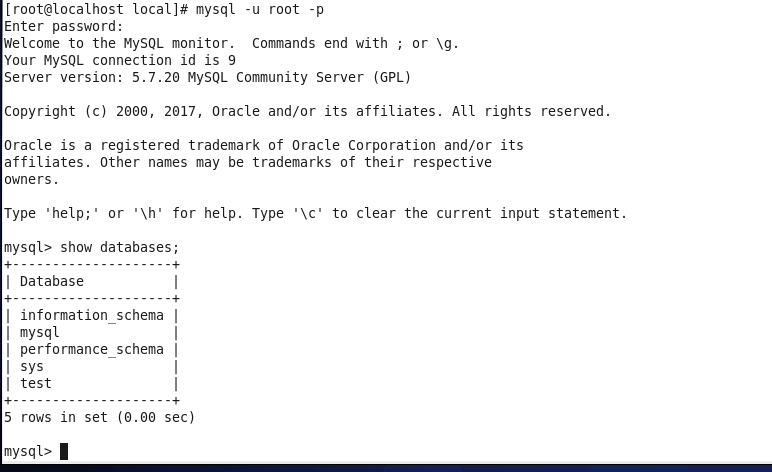
**Shell># mysql –u root –p**

**输入初始化时创建的密码。**

**修改root密码：**

**mysql> ALTER USER ‘root’@’localhost’ IDENTIFIED BY ‘new\_passwd’;**

**修改完成之后就可以操作数据库了。**



小提示：mysql配置文件的存放路径，位于：/etc/my.cnf

若出现问题无法解决，可以去官方文档查找问题的解决方法：

<https://dev.mysql.com/doc/>

### 设置mysql开启启动

**Shell># chkconfig --add mysqld**

**Shell># chkconfig mysql on**

### 卸载rpm方式安装的mysql

**卸载步骤：**

**1，查看已安装mysql软件包**

**Shell># rpm -qa | grep**

**2，卸载已软件包**

**Shell># rpm -e mysql\***

**3，查看mysql安装目录**

**Shell># find / -name mysql**

**4，删除mysql相关文件和目录**

**Shell># rm -rf xxxxxxxx**

### 更多rpm方式安装mysql参考

**<http://blog.csdn.net/superchanon/article/details/8546254/>**

## 三，yum方式安装

### 到官网下载必备包

**Shell># wget \**

**<http://dev.mysql.com/get/mysql-community-release-el7-5.noarch.rpm>**

### 安装mysql-community-release

**Shell># rpm -ivh mysql-community-release-el7-5.noarch.rpm**

### 安装mysql

**Shell># yum install mysql-community-server**

### 启动mysql服务

**Shell># service mysqld restart**

### 进入mysql，修改mysql密码

**Shell># mysql -uroot**

**mysql> set password for 'root'@'localhost' = password('你要修改的密码');**

### 设置远程访问

**Mysql> use mysql;**

**Mysql> select \* from user;**

**Mysql> update user set Host = '%' where User=’root’;**

**Mysql> quit;**

### 设置开启启动

**Shell># chkconfig --add mysqld**

**Shell># chkconfig mysqld on**

### 设置mysql文件权限

**请参考rpm方式安装mysql的步骤7**

### 卸载yum方式安装的mysql

**Shell># yum list install | grep mysql**

**Shell># yum -y remove mysql\***

### 更多yum方式安装mysql参考

**<http://www.cnblogs.com/smbin/p/6946210.html>**

**<http://www.cnblogs.com/007sx/p/7083143.html>**

**<http://jingyan.baidu.com/article/b2c186c8216455c46ef6ff34.html>**

## 四，my.cnf配置文件详解

/etc/my.cnf 参考：

[client] #客户端设置，即客户端默认的连接参数

port = 3306 #默认连接端口号

socket = /tmp/mysql.sock #用于本地连接的socket套接字

default-character-set=utf8mb4 #默认编码

[mysqld] #服务端基本设置

port = 3306 #mysql监听端口

socket = /tmp/mysql.sock #为MySQL客户端程序和服务器之间的本地 通讯指定一个套接字文件

basedir = /usr/local/mysql #mysql安装目录

datadir = /data/mysql #数据文件存放的目录

pid-file = /data/mysql/mysql.pid #pid文件所在目录

user = mysql #MySQL启动用户

character\_set\_server = utf8mb4 　　　　#服务端默认编码（数据库级别）

collation\_server = utf8mb4\_bin 　　　　#服务端默认的比对规则，排序规则

bind-address = 0.0.0.0 #MySQL绑定IP

server-id = 1 #表示是本机的序号为1,一般来讲就是

master的意思

skip-name-resolve

# 禁止MySQL对外部连接进行DNS解析，使用这一选项可以消除MySQL进行DNS解析的时间。但需要注意，如果开启该选项，则所有远程主机连接授权都要使用IP地址方式，否则MySQL将无法正常处理连接请求。

#skip-networking

back\_log = 600

# MySQL能有的连接数量。当主要MySQL线程在一个很短时间内得到非常多的连接请求，这就起作用，然后主线程花些时间(尽管很短)检查连接并且启动一个新线程。back\_log值指出在MySQL暂时停止回答新请求之前的短时间内多少个请求可以被存在堆栈中。

# 如果期望在一个短时间内有很多连接，你需要增加它。也就是说， 如果MySQL的连接数据达到max\_connections时，新来的请求将会被存在堆栈中，以等待某一连接释放资源，该堆栈的数量即back\_log，如果等待连接的数量超过back\_log，将不被授予连接资源。

# 另外，这值（back\_log）限于您的操作系统对到来的TCP/IP连接的侦听队列的大小。你的操作系统在这个队列大小上有它自己的限制（可以检查你的OS文档找出这个变量的最大值），试图设定back\_log高于你的操作系统的限制将是无效的。

max\_connections = 1000

# MySQL的最大连接数，如果服务器的并发连接请求量比较大，建议调高此值，以增加并行连接数量，当然这建立在机器能支撑的情况下，因为如果连接数越多，介于MySQL会为每个连接提供连接缓冲区，就会开销越多的内存，所以要适当调整该值，不能盲目提高设值。可以过'conn%'通配符查看当前状态的连接数量，以定夺该值的大小。

max\_connect\_errors = 6000

# 对于同一主机，如果有超出该参数值个数的中断错误连接，则该主机将被禁止连接。如需对该主机进行解禁，执行：FLUSH HOST。

open\_files\_limit = 65535

# MySQL打开的文件描述符限制，默认最小1024;当open\_files\_limit没有被配置的时候，比较max\_connections\*5和ulimit -n的值，哪个大用哪个，

# 当open\_file\_limit被配置的时候，比较open\_files\_limit和max\_connections\*5的值，哪个大用哪个。

table\_open\_cache = 128

# MySQL每打开一个表，都会读入一些数据到table\_open\_cache缓存中，当MySQL在这个缓存中找不到相应信息时，才会去磁盘上读取。默认值64

# 假定系统有200个并发连接，则需将此参数设置为200\*N(N为每个连接所需的文件描述符数目)；当把table\_open\_cache设置为很大时，如果系统处理不了那么多文件描述符，那么就会出现客户端失效，连接不上

max\_allowed\_packet = 4M

# 接受的数据包大小；增加该变量的值十分安全，这是因为仅当需要时才会分配额外内存。例如，仅当你发出长查询或MySQLd必须返回大的结果行时MySQLd才会分配更多内存。

# 该变量之所以取较小默认值是一种预防措施，以捕获客户端和服务器之间的错误信息包，并确保不会因偶然使用大的信息包而导致内存溢出。

binlog\_cache\_size = 1M

# 一个事务，在没有提交的时候，产生的日志，记录到Cache中；等到事务提交需要提交的时候，则把日志持久化到磁盘。默认binlog\_cache\_size大小32K

max\_heap\_table\_size = 8M

# 定义了用户可以创建的内存表(memory table)的大小。这个值用来计算内存表的最大行数值。这个变量支持动态改变

tmp\_table\_size = 16M

# MySQL的heap（堆积）表缓冲大小。所有联合在一个DML指令内完成，并且大多数联合甚至可以不用临时表即可以完成。

# 大多数临时表是基于内存的(HEAP)表。具有大的记录长度的临时表 (所有列的长度的和)或包含BLOB列的表存储在硬盘上。

# 如果某个内部heap（堆积）表大小超过tmp\_table\_size，MySQL可以根据需要自动将内存中的heap表改为基于硬盘的MyISAM表。还可以通过设置tmp\_table\_size选项来增加临时表的大小。也就是说，如果调高该值，MySQL同时将增加heap表的大小，可达到提高联接查询速度的效果

read\_buffer\_size = 2M

# MySQL读入缓冲区大小。对表进行顺序扫描的请求将分配一个读入缓冲区，MySQL会为它分配一段内存缓冲区。read\_buffer\_size变量控制这一缓冲区的大小。

# 如果对表的顺序扫描请求非常频繁，并且你认为频繁扫描进行得太慢，可以通过增加该变量值以及内存缓冲区大小提高其性能

read\_rnd\_buffer\_size = 8M

# MySQL的随机读缓冲区大小。当按任意顺序读取行时(例如，按照排序顺序)，将分配一个随机读缓存区。进行排序查询时，MySQL会首先扫描一遍该缓冲，以避免磁盘搜索，提高查询速度，如果需要排序大量数据，可适当调高该值。但MySQL会为每个客户连接发放该缓冲空间，所以应尽量适当设置该值，以避免内存开销过大

sort\_buffer\_size = 8M

# MySQL执行排序使用的缓冲大小。如果想要增加ORDER BY的速度，首先看是否可以让MySQL使用索引而不是额外的排序阶段。

# 如果不能，可以尝试增加sort\_buffer\_size变量的大小

join\_buffer\_size = 8M

# 联合查询操作所能使用的缓冲区大小，和sort\_buffer\_size一样，该参数对应的分配内存也是每连接独享

thread\_cache\_size = 8

# 这个值（默认8）表示可以重新利用保存在缓存中线程的数量，当断开连接时如果缓存中还有空间，那么客户端的线程将被放到缓存中，

# 如果线程重新被请求，那么请求将从缓存中读取,如果缓存中是空的或者是新的请求，那么这个线程将被重新创建。

# 如果有很多新的线程，增加这个值可以改善系统性能.通过比较Connections和Threads\_created状态的变量，可以看到这个变量的作用。(–>表示要调整的值)

# 根据物理内存设置规则如下：

# 1G —> 8

# 2G —> 16

# 3G —> 32

# 大于3G —> 64

query\_cache\_size = 8M

#MySQL的查询缓冲大小（从4.0.1开始，MySQL提供了查询缓冲机制）使用查询缓冲，MySQL将SELECT语句和查询结果存放在缓冲区中，

# 今后对于同样的SELECT语句（区分大小写），将直接从缓冲区中读取结果。根据MySQL用户手册，使用查询缓冲最多可以达到238%的效率。

# 通过检查状态值'Qcache\_%'，可以知道query\_cache\_size设置是否合理：如果Qcache\_lowmem\_prunes的值非常大，则表明经常出现缓冲不够的情况，

# 如果Qcache\_hits的值也非常大，则表明查询缓冲使用非常频繁，此时需要增加缓冲大小；如果Qcache\_hits的值不大，则表明你的查询重复率很低，

# 这种情况下使用查询缓冲反而会影响效率，那么可以考虑不用查询缓冲。此外，在SELECT语句中加入SQL\_NO\_CACHE可以明确表示不使用查询缓冲

query\_cache\_limit = 2M

#指定单个查询能够使用的缓冲区大小，默认1M

key\_buffer\_size = 4M

#指定用于索引的缓冲区大小，增加它可得到更好处理的索引(对所有读和多重写)，到你能负担得起那样多。如果你使它太大，系统将开始换页并且真的变慢了。对于内存在4GB左右的服务器该参数可设置为384M或512M。通过检查状态值Key\_read\_requests和Key\_reads，可以知道key\_buffer\_size设置是否合理。比例key\_reads/key\_read\_requests应该尽可能的低，至少是1:100，1:1000更好(上述状态值可以使用SHOW STATUS LIKE 'key\_read%'获得)。注意：该参数值设置的过大反而会是服务器整体效率降低

ft\_min\_word\_len = 4

# 分词词汇最小长度，默认4

transaction\_isolation = REPEATABLE-READ

# MySQL支持4种事务隔离级别，他们分别是：

# READ-UNCOMMITTED, READ-COMMITTED, REPEATABLE-READ, SERIALIZABLE.

# 如没有指定，MySQL默认采用的是REPEATABLE-READ，ORACLE默认的是READ-COMMITTED

log\_bin = mysql-bin

binlog\_format = mixed

# 在复制方面的改进就是引进了新的复制技术：基于行的复制。简言之，这种新技术就是关注表中发生变化的记录，而非以前的照抄 binlog 模式。

　　# 从 MySQL 5.1.12 开始，可以用以下三种模式来实现：基于SQL语句的复制(statement-based replication, SBR)，基于行的复制(row-based replication, RBR)，混合模式复制(mixed-based replication, MBR)。相应地，binlog的格式也有三种：STATEMENT，ROW，MIXED。MBR 模式中，SBR 模式是默认的。

expire\_logs\_days = 30 #超过30天的binlog删除

log\_error = /data/mysql/mysql-error.log #错误日志路径

slow\_query\_log = 1 #开启慢查询

long\_query\_time = 1 #慢查询时间 超过1秒则为慢查询

slow\_query\_log\_file = /data/mysql/mysql-slow.log

# 慢查询日志存放位置

performance\_schema = 0 # 性能优化的引擎，默认关闭

explicit\_defaults\_for\_timestamp # 明确时间戳默认null方式

#lower\_case\_table\_names = 1 #不区分大小写

skip-external-locking

# 跳过外部锁定;External-locking用于多进程条件下为MyISAM数据表进行锁定

# MySQL选项以避免外部锁定。该选项默认开启

default-storage-engine = InnoDB #默认存储引擎

innodb\_file\_per\_table = 1

# InnoDB为独立表空间模式，每个数据库的每个表都会生成一个数据空间

# 独立表空间优点：

# 1．每个表都有自已独立的表空间。

# 2．每个表的数据和索引都会存在自已的表空间中。

# 3．可以实现单表在不同的数据库中移动。

# 4．空间可以回收（除drop table操作处，表空不能自已回收）

# 缺点：

# 单表增加过大，如超过100G

# 结论：

# 共享表空间在Insert操作上少有优势。其它都没独立表空间表现好。当启用独立表空间时，请合理调整：innodb\_open\_files

innodb\_open\_files = 500

# 限制Innodb能打开的表的数据，如果库里的表特别多的情况，请增加这个。这个值默认是300

innodb\_buffer\_pool\_size = 64M

# InnoDB使用一个缓冲池来保存索引和原始数据, 不像MyISAM.

# 这里你设置越大,你在存取表里面数据时所需要的磁盘I/O越少.

# 在一个独立使用的数据库服务器上,你可以设置这个变量到服务器物理内存大小的80%

# 不要设置过大,否则,由于物理内存的竞争可能导致操作系统的换页颠簸.

# 注意在32位系统上你每个进程可能被限制在 2-3.5G 用户层面内存限制,

# 所以不要设置的太高.

innodb\_write\_io\_threads = 4

innodb\_read\_io\_threads = 4

# innodb使用后台线程处理数据页上的读写 I/O(输入输出)请求,根据你的 CPU 核数来更改,默认是4

#对于多核的CPU机器，可以修改innodb\_read\_io\_threads和innodb\_write\_io\_threads来增加IO线程，来充分利用多核的性能

# 注:这两个参数不支持动态改变,需要把该参数加入到my.cnf里，修改完后重启MySQL服务,允许值的范围从 1-64

innodb\_thread\_concurrency = 0

# 默认设置为 0,表示不限制并发数，这里推荐设置为0，更好去发挥CPU多核处理能力，提高并发量

innodb\_purge\_threads = 1

# InnoDB中的清除操作是一类定期回收无用数据的操作。在之前的几个版本中，清除操作是主线程的一部分，这意味着运行时它可能会堵塞其它的数据库操作。

# 从MySQL5.5.X版本开始，该操作运行于独立的线程中,并支持更多的并发数。用户可通过设置innodb\_purge\_threads配置参数来选择清除操作是否使用单独线程,默认情况下参数设置为0(不使用单独线程),设置为 1 时表示使用单独的清除线程。建议为1

innodb\_flush\_log\_at\_trx\_commit = 2

# 0：如果innodb\_flush\_log\_at\_trx\_commit的值为0,log buffer每秒就会被刷写日志文件到磁盘，提交事务的时候不做任何操作（执行是由mysql的master thread线程来执行的。

# 主线程中每秒会将重做日志缓冲写入磁盘的重做日志文件(REDO LOG)中。不论事务是否已经提交）默认的日志文件是ib\_logfile0,ib\_logfile1

# 1：当设为默认值1的时候，每次提交事务的时候，都会将log buffer刷写到日志。

# 2：如果设为2,每次提交事务都会写日志，但并不会执行刷的操作。每秒定时会刷到日志文件。要注意的是，并不能保证100%每秒一定都会刷到磁盘，这要取决于进程的调度。

# 每次事务提交的时候将数据写入事务日志，而这里的写入仅是调用了文件系统的写入操作，而文件系统是有 缓存的，所以这个写入并不能保证数据已经写入到物理磁盘

# 默认值1是为了保证完整的ACID。当然，你可以将这个配置项设为1以外的值来换取更高的性能，但是在系统崩溃的时候，你将会丢失1秒的数据。

# 设为0的话，mysqld进程崩溃的时候，就会丢失最后1秒的事务。设为2,只有在操作系统崩溃或者断电的时候才会丢失最后1秒的数据。InnoDB在做恢复的时候会忽略这个值。

# 总结：

# 设为1当然是最安全的，但性能也是最差的（相对其他两个参数而言，但不是不能接受）。如果对数据一致性和完整性要求不高，完全可以设为2，如果只最求性能，例如高并发写的日志服务器，设为0来获得更高性能

innodb\_log\_buffer\_size = 2M

# 此参数确定些日志文件所用的内存大小，以M为单位。缓冲区更大能提高性能，但意外的故障将会丢失数据。MySQL开发人员建议设置为1－8M之间

innodb\_log\_file\_size = 32M

# 此参数确定数据日志文件的大小，更大的设置可以提高性能，但也会增加恢复故障数据库所需的时间

innodb\_log\_files\_in\_group = 3

# 为提高性能，MySQL可以以循环方式将日志文件写到多个文件。推荐设置为3

innodb\_max\_dirty\_pages\_pct = 90

# innodb主线程刷新缓存池中的数据，使脏数据比例小于90%

# innodb\_max\_dirty\_pages\_pct作用：控制Innodb的脏页在缓冲中在那个百分比之下，值在范围1-100,默认为90.这个参数的另一个用处：当Innodb的内存分配过大，致使swap占用严重时，可以适当的减小调整这个值，使达到swap空间释放出来。建义：这个值最大在90%，最小在15%。太大，缓存中每次更新需要致换数据页太多，太小，放的数据页太小，更新操作太慢。

innodb\_lock\_wait\_timeout = 120

# InnoDB事务在被回滚之前可以等待一个锁定的超时秒数。InnoDB在它自己的锁定表中自动检测事务死锁并且回滚事务。InnoDB用LOCK TABLES语句注意到锁定设置。默认值是50秒

bulk\_insert\_buffer\_size = 8M

# 批量插入缓存大小， 这个参数是针对MyISAM存储引擎来说的。适用于在一次性插入100-1000+条记录时， 提高效率。默认值是8M。可以针对数据量的大小，翻倍增加。

myisam\_sort\_buffer\_size = 8M

# MyISAM设置恢复表之时使用的缓冲区的尺寸，当在REPAIR TABLE或用CREATE INDEX创建索引或ALTER TABLE过程中排序 MyISAM索引分配的缓冲区

myisam\_max\_sort\_file\_size = 10G

# 如果临时文件会变得超过索引，不要使用快速排序索引方法来创建一个索引。注释：这个参数以字节的形式给出

myisam\_repair\_threads = 1

# 如果该值大于1，在Repair by sorting过程中并行创建MyISAM表索引(每个索引在自己的线程内)

interactive\_timeout = 28800

# 服务器关闭交互式连接前等待活动的秒数。交互式客户端定义为在mysql\_real\_connect()中使用CLIENT\_INTERACTIVE选项的客户端。默认值：28800秒（8小时）

wait\_timeout = 28800

# 服务器关闭非交互连接之前等待活动的秒数。在线程启动时，根据全局wait\_timeout值或全局interactive\_timeout值初始化会话wait\_timeout值，

# 取决于客户端类型(由mysql\_real\_connect()的连接选项CLIENT\_INTERACTIVE定义)。参数默认值：28800秒（8小时）

# MySQL服务器所支持的最大连接数是有上限的，因为每个连接的建立都会消耗内存，因此我们希望客户端在连接到MySQL Server处理完相应的操作后，

# 应该断开连接并释放占用的内存。如果你的MySQL Server有大量的闲置连接，他们不仅会白白消耗内存，而且如果连接一直在累加而不断开，最终肯定会达到MySQL Server的连接上限数，这会报'too many connections'的错误。对于wait\_timeout的值设定，应该根据系统的运行情况来判断。

# 在系统运行一段时间后，可以通过show processlist命令查看当前系统的连接状态，如果发现有大量的sleep状态的连接进程，则说明该参数设置的过大，可以进行适当的调整小些。要同时设置interactive\_timeout和wait\_timeout才会生效。

[mysqldump]

quick

#它强制 mysqldump 从服务器查询取得记录直接输出而不是取得所有记录后将它们缓存到内存中

max\_allowed\_packet = 16M

#限制server接受的数据包大小;指代mysql服务器端和客户端在一次传送数据包的过程当中数据包的大小

net\_buffer\_length = 16384

#TCP/IP和套接字通信缓冲区大小,创建长度达net\_buffer\_length的行

[mysql]

auto-rehash　　　　　　　　　　　　　　 #auto-rehash是自动补全的意思

[isamchk] 　　　　　　　　　　　　 #isamchk数据检测恢复工具

key\_buffer = 256M

sort\_buffer\_size = 256M

read\_buffer = 2M

write\_buffer = 2M

[myisamchk] #使用myisamchk实用程序来获得有关你 的数据库桌表的信息、检查和修复他们或 优化他们

key\_buffer\_size = 8M

sort\_buffer\_size = 8M

read\_buffer = 4M

write\_buffer = 4M

[mysqlhotcopy]

interactive-timeout

#mysqlhotcopy使用lock tables、flush tables和cp或scp来快速备份数据库.它是备份数据库或单个表最快的途径,完全属于物理备份,但只能用于备份MyISAM存储引擎和运行在数据库目录所在的机器上.

#与mysqldump备份不同,mysqldump属于逻辑备份,备份时是执行的sql语句.使用mysqlhotcopy命令前需要要安装相应的软件依赖包.

## 五，mysql 源配置详解

有关CMake支持的选项的信息，请在顶级源目录中运行以下命令之一：

**shell># cmake . -LH**

**shell># ccmake .**

**常用源配置：**

**PREFIX 代表 mysql 的安装路径**

-DCMAKE\_INSTALL\_PREFIX=dir\_name

安装基本目录。默认：/usr/local/mysql

可以使用该--basedir选项在服务器启动时设置此值 。

-DINSTALL\_MYSQLTESTDIR=dir\_name

在哪里安装mysql-test 目录。默认：PREFIX

从MySQL 5.7.2开始，要禁止安装此目录，请将该选项显式设置为空值（-DINSTALL\_MYSQLTESTDIR=）。

-DINSTALL\_DOCDIR=dir\_name

哪里安装文档。默认：PREFIX/docs

-DINSTALL\_DOCREADMEDIR=dir\_name

哪里安装README文件。默认：PREFIX

-DINSTALL\_LIBDIR=dir\_name

哪里安装库文件。默认：PREFIX/lib

-DINSTALL\_MANDIR=dir\_name

在哪里安装手册页。默认：PREFIX/man

-DINSTALL\_PLUGINDIR=dir\_name

插件目录的位置。默认：PREFIX/lib/plugin

可以使用该--plugin\_dir选项在服务器启动时设置此值 。

-DINSTALL\_SBINDIR=dir\_name

服务器可执行文件

在哪里安装mysqld服务器。默认：PREFIX/bin

-DINSTALL\_SCRIPTDIR=dir\_name

脚本目录

在哪里安装mysql\_instll\_db。默认：PREFIX/scripts

-DINSTALL\_SQLBENCHDIR=dir\_name

在哪里安装sql-bench目录。默认：PREFIX

要禁止安装此目录，请将该选项显式设置为空值（-DINSTALL\_SQLBENCHDIR=）

从MySQL 5.7.8开始，该sql-bench 目录不再包含在MYSQL发行版中，所以该 INSTALL\_SQLBENCHDIR=选项也被删除。

-DMYSQL\_DATADIR=dir\_name

MySQL数据目录的位置。

可以使用该--datadir选项在服务器启动时设置此值 。

-DSYSCONFDIR=dir\_name

默认my.cnf选项文件目录。

在服务器启动时无法设置此位置，但您可以使用 选项启动具有给定选项文件的服务器 ，其中是文件的完整路径名。 --defaults-file=file\_name

-DSYSTEMD\_PID\_DIR=dir\_name

当MySQL由systemd管理时，创建PID文件的目录的名称。默认值为 /var/run/mysqld; 这可能会根据INSTALL\_LAYOUT价值隐含地改变 。

忽略此选项，除非 WITH\_SYSTEMD已启用。它被添加到MySQL 5.7.6中。

-DSYSTEMD\_SERVICE\_NAME=name

当MySQL由systemd管理时使用的MySQL服务的名称。默认值为mysqld; 这可能会根据INSTALL\_LAYOUT价值隐含地改变 。

忽略此选项，除非 WITH\_SYSTEMD已启用。它被添加到MySQL 5.7.6中。

-DTMPDIR=dir\_name

用于tmpdir系统变量的默认位置 。如果未指定，则该值默认为P\_tmpdir in <stdio.h>。此选项已添加到MySQL 5.7.4中。

-DDEFAULT\_CHARSET=charset\_name

服务器字符集。默认情况下，MySQL使用 latin1（cp1252西欧）字符集。

charset\_name可以是一种 binary，armscii8， ascii，big5， cp1250，cp1251， cp1256，cp1257， cp850，cp852， cp866，cp932， dec8，eucjpms， euckr，gb2312， gbk，geostd8， greek，hebrew， hp8，keybcs2， koi8r，koi8u， latin1，latin2， latin5，latin7， macce，macroman， sjis，swe7， tis620，ucs2， ujis，utf8， utf8mb4，utf16， utf16le，utf32。cmake/character\_sets.cmake文件中列出了允许的字符集 作为值CHARSETS\_AVAILABLE。

可以使用该--character\_set\_server选项在服务器启动时设置此值 。

-DDEFAULT\_COLLATION=collation\_name

服务器排序规则。默认情况下，MySQL使用 latin1\_swedish\_ci。使用该 SHOW COLLATION语句来确定哪些归类可用于每个字符集。

可以使用该--collation\_server选项在服务器启动时设置此值 。

-DDISABLE\_PSI\_COND=bool

是否排除性能模式条件检测。默认值为OFF（包括）。此选项已添加到MySQL 5.7.3中。

-DDISABLE\_PSI\_FILE=bool

是否排除Performance Schema文件检测。默认值为OFF （包括）。此选项已添加到MySQL 5.7.3中。

-DDISABLE\_PSI\_IDLE=bool

是否排除Performance Schema空闲工具。默认值为OFF （包括）。此选项已添加到MySQL 5.7.3中。

-DDISABLE\_PSI\_MEMORY=bool

是否排除性能模式存储器检测。默认值为OFF （包括）。此选项已添加到MySQL 5.7.3中。

-DDISABLE\_PSI\_METADATA=bool

是否排除Performance Schema元数据工具。默认值为OFF （包括）。此选项已添加到MySQL 5.7.3中。

-DDISABLE\_PSI\_MUTEX=bool

是否排除性能模式互斥检测。默认值为OFF （包括）。此选项已添加到MySQL 5.7.3中。

-DDISABLE\_PSI\_RWLOCK=bool

是否排除Performance Schema rwlock工具。默认值为OFF （包括）。此选项已添加到MySQL 5.7.3中。

-DDISABLE\_PSI\_SOCKET=bool

是否排除Performance Schema套接字工具。默认值为OFF （包括）。此选项已添加到MySQL 5.7.3中。

-DDISABLE\_PSI\_SP=bool

是否排除Performance Schema存储的程序工具。默认值为OFF （包括）。此选项已添加到MySQL 5.7.3中。

-DDISABLE\_PSI\_STAGE=bool

是否排除Performance Schema阶段工具。默认值为OFF （包括）。此选项已添加到MySQL 5.7.3中。

-DDISABLE\_PSI\_STATEMENT=bool

是否排除性能模式语句工具。默认值为OFF （包括）。此选项已添加到MySQL 5.7.3中。

-DDISABLE\_PSI\_STATEMENT\_DIGEST=bool

是否排除Performance Schema statement\_digest工具。默认值为OFF （包括）。此选项已添加到MySQL 5.7.3中。

-DDISABLE\_PSI\_TABLE=bool

是否排除Performance Schema表的工具。默认值为OFF （包括）。此选项已添加到MySQL 5.7.3中。

-DENABLE\_DOWNLOADS=bool

是否下载可选文件。例如，启用此选项后，CMake会下载测试套件使用的Google Test发行版，以运行单元测试。

-DENABLE\_DTRACE=bool

是否包括对DTrace探针的支持。

此选项不推荐使用，因为在MySQL 5.7中不支持DTrace，在MySQL 8.0中已被删除。

-DENABLE\_GCOV=bool

是否包含gcov支持（仅限Linux）。

-DENABLE\_GPROF=bool

是否启用gprof（仅优化Linux版本）。

-DFORCE\_UNSUPPORTED\_COMPILER=bool

默认情况下，CMake检查支持的编译器的最低版本：Visual Studio 2013（Windows）; GCC 4.4或Clang 3.3（Linux）; Developer Studio 12.5（Solaris服务器）; Developer Studio 12.2或GCC 4.4（Solaris客户端库）; Clang 3.3（macOS），Clang 3.3（FreeBSD）。要禁用此检查，请使用 -DFORCE\_UNSUPPORTED\_COMPILER=ON。此选项已添加到MySQL 5.7.5中。

-DIGNORE\_AIO\_CHECK=bool

如果-DBUILD\_CONFIG=mysql\_release 在Linux上给出了该 选项，那么该libaio 库必须默认链接。如果没有 libaio或不想安装，可以通过指定来禁止检查 -DIGNORE\_AIO\_CHECK=1。

-DMAX\_INDEXES=num

每个表的最大索引数。默认值为64.最大值为255.小于64的值将被忽略，默认值为64。

-DMYSQL\_MAINTAINER\_MODE=bool

是否启用MySQL维护者特定的开发环境。如果启用，此选项将导致编译器警告成为错误。

-DMYSQL\_TCP\_PORT=port\_num

服务器侦听TCP / IP连接的端口号。默认值为3306。

可以使用该--port选项在服务器启动时设置此值 。

-DMYSQL\_UNIX\_ADDR=file\_name

服务器侦听套接字连接的Unix套接字文件路径。这必须是绝对路径名。默认是/tmp/mysql.sock。

可以使用该--socket选项在服务器启动时设置此值 。

-DWITH\_BOOST=path\_name

Boost库源的位置。

从MySQL 5.7.5开始，需要Boost库构建MySQL。这些CMake选项启用对库源位置的控制，以及是否自动下载它：

-DWITH\_BOOST=path\_name 指定Boost库目录位置。也可以通过设置BOOST\_ROOT或 WITH\_BOOST环境变量来指定Boost位置 。

从MySQL 5.7.11开始， -DWITH\_BOOST=system允许并指示在标准位置的编译主机上安装了正确版本的Boost。在这种情况下，使用Boost的安装版本，而不是使用MySQL源代码发行版中包含的任何版本。

-DDOWNLOAD\_BOOST=bool

指定是否下载Boost源（如果它不在指定位置）。默认是 OFF。

-DDOWNLOAD\_BOOST\_TIMEOUT=seconds

下载Boost库的超时时间（以秒为单位）。默认值为600秒。

例如，如果您通常构建MySQL将对象输出放在bldMySQL源代码树的子目录中，则可以使用Boost构建：

Shell># mkdir bld

Shell># cd bld

Shell># cmake .. -DDOWNLOAD\_BOOST=ON -DWITH\_BOOST=$HOME/my\_boost

这会导致Boost被下载到 my\_boost您的主目录下的目录中。如果所需的Boost版本已经存在，则不进行下载。如果所需的Boost版本更改，则下载较新版本。

如果Boost已经在本地安装，并且您的编译器自行找到Boost头文件，则可能无需指定前面的CMake选项。但是，如果MySQL所需的Boost版本更改并且本地安装的版本尚未升级，则可能会有构建问题。使用CMake 选项可以让您成功构建。

使用允许Boost下载到指定位置的上述设置，当需要的Boost版本更改时，您需要删除该bld文件夹，重新创建它，然后再次执行cmake步骤。否则，新的Boost版本可能无法下载，编译可能会失败。

-DWITH\_EXTRA\_CHARSETS=name

哪些额外的字符集包括：

all：所有字符集。这是默认值。

complex：复杂的字符集。

none：没有额外的字符集。

-DWITH\_INNODB\_MEMCACHED=bool

是否生成memcached共享库（libmemcached.so和 innodb\_engine.so）。

-DWITH\_SYSTEMD=bool

是否启用systemd支持文件的安装。默认情况下，此选项被禁用。启用系统支持文件后， 将不安装mysqld\_safe和System V初始化脚本等脚本。在systemd不可用的平台上，启用CMakeWITH\_SYSTEMD 导致错误。

该部分还包括有关指定先前在[mysqld\_safe]选项组中指定的 选项的信息。因为 在使用systemd时没有安装mysqld\_safe，所以这个选项必须以另一种方式指定。

此选项已添加到MySQL 5.7.6中。

-DWITH\_ZLIB=zlib\_type

某些功能要求使用压缩库支持来构建服务器，例如 客户机/服务器协议的功能COMPRESS()和 UNCOMPRESS()功能以及压缩。这 WITH\_ZLIB表明zlib支持的来源：

bundled：使用zlib与发行版捆绑在一起的 库。这是默认值。

system：使用系统 zlib库。

-DWITHOUT\_SERVER=bool

不要建立服务器。默认是 OFF，它构建服务器。

更详细的mysql源配置请参考：

<https://dev.mysql.com/doc/refman/5.7/en/source-configuration-options.html#cmake-general-options>

BUILD\_CONFIG 使用与官方版本相同的构建选项

CMAKE\_BUILD\_TYPE 生产类型 RelWithDebInfo

CMAKE\_CXX\_FLAGS C ++编译器的标志

CMAKE\_C\_FLAGS C编译器的标志

CMAKE\_INSTALL\_PREFIX 安装基地目录 /usr/local/mysql

COMPILATION\_COMMENT 关于编译环境的评论

CPACK\_MONOLITHIC\_INSTALL 包构建是否生成单个文件 OFF

DEFAULT\_CHARSET 默认服务器字符集 latin1

DEFAULT\_COLLATION 默认服务器排序规则 latin1\_swedish\_ci

DISABLE\_PSI\_COND 排除性能模式条件仪器 OFF 5.7.3

DISABLE\_PSI\_FILE 排除性能模式文件检测 OFF 5.7.3

DISABLE\_PSI\_IDLE 排除性能模式空闲测试 OFF 5.7.3

DISABLE\_PSI\_MEMORY 排除性能模式存储器测试 OFF 5.7.3

DISABLE\_PSI\_METADATA 排除性能模式元数据测试 OFF 5.7.3

DISABLE\_PSI\_MUTEX 排除性能模式互斥检测 OFF 5.7.3

DISABLE\_PSI\_RWLOCK 排除性能模式rwlock仪器 OFF 5.7.3

DISABLE\_PSI\_SOCKET 排除性能模式套接字工具 OFF 5.7.3

DISABLE\_PSI\_SP 排除性能模式存储的程序工具 OFF 5.7.3

DISABLE\_PSI\_STAGE 排除性能模式阶段仪器 OFF 5.7.3

DISABLE\_PSI\_STATEMENT 排除性能模式语句工具 OFF 5.7.3

DISABLE\_PSI\_STATEMENT\_DIGEST 排除性能模式statement\_digest工具 OFF 5.7.3

DISABLE\_PSI\_TABLE 排除性能模式表工具 OFF 5.7.3

DOWNLOAD\_BOOST 是否下载Boost库 OFF 5.7.5

DOWNLOAD\_BOOST\_TIMEOUT 下载Boost库超时（秒） 600 5.7.6

ENABLED\_LOCAL\_INFILE 是否启用LOCAL用于LOAD DATA INFILE OFF

ENABLED\_PROFILING 是否启用查询分析代码 ON

ENABLE\_DEBUG\_SYNC 是否启用Debug Sync支持 ON 5.7.1

ENABLE\_DOWNLOADS 是否下载可选文件 OFF

ENABLE\_DTRACE 是否包括DTrace支持

ENABLE\_GCOV 是否包括gcov支持

ENABLE\_GPROF 启用gprof（仅优化Linux版本） OFF

FORCE\_UNSUPPORTED\_COMPILER 是否允许不支持的编译器 OFF 5.7.5

IGNORE\_AIO\_CHECK 用-DBUILD\_CONFIG = mysql\_release，忽略libaio检查 OFF

INNODB\_PAGE\_ATOMIC\_REF\_COUNT 启用或禁用原子页面引用计数 ON 5.7.4 5.7.5

INSTALL\_BINDIR 用户可执行文件目录 PREFIX/bin

INSTALL\_DOCDIR 文档目录 PREFIX/docs

INSTALL\_DOCREADMEDIR README文件目录 PREFIX

INSTALL\_INCLUDEDIR 头文件目录 PREFIX/include

INSTALL\_INFODIR Info文件目录 PREFIX/docs

INSTALL\_LAYOUT 选择预定义的安装布局 STANDALONE

INSTALL\_LIBDIR 库文件目录 PREFIX/lib

INSTALL\_MANDIR 手动页面目录 PREFIX/man

INSTALL\_MYSQLKEYRINGDIR keyring\_file插件数据文件的目录 platform specific 5.7.11

INSTALL\_MYSQLSHAREDIR 共享数据目录 PREFIX/share

INSTALL\_MYSQLTESTDIR mysql-test目录 PREFIX/mysql-test

INSTALL\_PKGCONFIGDIR mysqlclient.pc pkg-config文件的目录 INSTALL\_LIBDIR/pkgconfig 5.7.9

INSTALL\_PLUGINDIR 插件目录 PREFIX/lib/plugin

INSTALL\_SBINDIR 服务器可执行文件 PREFIX/bin

INSTALL\_SCRIPTDIR 脚本目录 PREFIX/scripts

INSTALL\_SECURE\_FILE\_PRIVDIR secure\_file\_priv默认值 platform specific 5.7.6

INSTALL\_SECURE\_FILE\_PRIV\_EMBEDDEDDIR libmysqld的secure\_file\_priv默认值 5.7.8

INSTALL\_SHAREDIR aclocal / mysql.m4安装目录 PREFIX/share

INSTALL\_SQLBENCHDIR sql-bench目录 PREFIX 5.7.8

INSTALL\_SUPPORTFILESDIR 额外的支持文件目录 PREFIX/support-files

MAX\_INDEXES 每个表的最大索引数 64 5.7.1

MUTEX\_TYPE InnoDB互斥体类型 event 5.7.2

MYSQLX\_TCP\_PORT X插件使用的TCP / IP端口号 33060 5.7.17

MYSQLX\_UNIX\_ADDR X插件使用的Unix套接字文件 /tmp/mysqlx.sock 5.7.15

MYSQL\_DATADIR 数据目录

MYSQL\_MAINTAINER\_MODE 是否启用MySQL维护者特定的开发环境 OFF

MYSQL\_PROJECT\_NAME Windows / OS X项目名称 MySQL

MYSQL\_TCP\_PORT TCP / IP端口号 3306

MYSQL\_UNIX\_ADDR Unix套接字文件 /tmp/mysql.sock

ODBC\_INCLUDES ODBC包含目录

ODBC\_LIB\_DIR ODBC库目录

OPTIMIZER\_TRACE 是否支持优化器跟踪

SUNPRO\_CXX\_LIBRARY Solaris 10+上的客户端链接库 5.7.5

SYSCONFDIR 选项文件目录

SYSTEMD\_PID\_DIR 系统下PID文件的目录 /var/run/mysqld 5.7.6

SYSTEMD\_SERVICE\_NAME systemd下的MySQL服务名称 mysqld 5.7.6

TMPDIR tmpdir默认值 5.7.4

WIN\_DEBUG\_NO\_INLINE 是否禁用功能内联 OFF 5.7.6

WITHOUT\_SERVER 不要建立服务器 OFF

WITHOUT\_xxx\_STORAGE\_ENGINE 从构建中排除存储引擎xxx

WITH\_ASAN 启用AddressSanitizer OFF 5.7.3

WITH\_AUTHENTICATION\_LDAP 是否报告错误，如果LDAP认证插件无法构建 OFF 19年7月5日

WITH\_AUTHENTICATION\_PAM 构建PAM验证插件 OFF

WITH\_BOOST Boost库源的位置 5.7.5

WITH\_CLIENT\_PROTOCOL\_TRACING 构建客户端协议跟踪框架 ON 5.7.2

WITH\_CURL 卷曲库的位置 19年7月5日

WITH\_DEBUG 是否包括调试支持 OFF

WITH\_DEFAULT\_COMPILER\_OPTIONS 是否使用默认编译器选项 ON

WITH\_DEFAULT\_FEATURE\_SET 是否使用默认功能集 ON

WITH\_EDITLINE 哪个libedit / editline库要使用 bundled 5.7.2

WITH\_EMBEDDED\_SERVER 是否构建嵌入式服务器 OFF

WITH\_EMBEDDED\_SHARED\_LIBRARY 是否构建共享嵌入式服务器库 OFF 5.7.4

WITH\_EXTRA\_CHARSETS 要包括哪些额外的字符集 all

WITH\_INNODB\_EXTRA\_DEBUG 是否包含对InnoDB的额外调试支持。 OFF 5.7.2

WITH\_INNODB\_MEMCACHED 是否生成memcached共享库。 OFF

WITH\_KEYRING\_TEST 构建密钥环测试程序 OFF 5.7.11

WITH\_LIBEVENT 哪个libevent库要使用 bundled

WITH\_LIBWRAP 是否包含libwrap（TCP包装）支持 OFF

WITH\_LZ4 LZ4支持类型 bundled 5.7.14

WITH\_MECAB 编译MeCab 5.7.6

WITH\_MSAN 启用MemorySanitizer OFF 5.7.4

WITH\_MSCRT\_DEBUG 启用Visual Studio CRT内存泄漏跟踪 OFF 5.7.6

WITH\_NDBCLUSTER 构建NDB存储引擎; WITH\_NDBCLUSTER\_STORAGE\_ENGINE的别名 ON

WITH\_NDBCLUSTER\_STORAGE\_ENGINE 构建NDB存储引擎 ON

WITH\_NDB\_TEST 包括NDB API测试程序。 OFF

WITH\_NUMA 设置NUMA内存分配策略 5.7.17

WITH\_PROTOBUF 哪个协议缓冲区包使用 bundled 5.7.12

WITH\_RAPID 是否构建快速的开发周期插件 ON 5.7.12

WITH\_SSL SSL支持类型 bundled

WITH\_SYSTEMD 启用systemd支持文件的安装 OFF 5.7.6

WITH\_TEST\_TRACE\_PLUGIN 构建测试协议跟踪插件 OFF 5.7.2

WITH\_UBSAN 启用未定义的行为消毒剂 OFF 5.7.6

WITH\_UNIT\_TESTS 使用单元测试编译MySQL ON

WITH\_UNIXODBC 启用unixODBC支持 OFF

WITH\_VALGRIND 是否在Valgrind头文件中编译 OFF

WITH\_ZLIB zlib支持类型 bundled

WITH\_xxx\_STORAGE\_ENGINE 将存储引擎xxx静态编译到服务器中

## 六，mysql远程登陆设置

**Mysql> Grant all privileges on \*.\* to \**

**> [‘root’@’192.168.234.2’](mailto:‘root’@’192.168.234.2’) identified by ‘password’;**

**Mysql> flush privileges;**

解释：以上语句，代表创建一个用户为root，可远程登陆mysql客户端的主机ip为192.168.234.2,密码为password。

@后面是访问mysql的客户端ip地址(或 主机名)，%代表任意客户端，如果填写 localhost 为本地访问（那此用户就不能远程访问该mysql数据库了）。

**同时，也可以为现有的用户设置是否具有远程访问权限。如：**

**Mysql> Use mysql;**

**Mysql> update user set host = '%' where user = 'test';**

**注：如果写成 host=localhost那此用户就不具有远程访问权限**

**Mysql> grant all privileges on \*.\* to test@'%';**

**Mysql> flush privileges;**

**最后需要设置防火墙：**

**方案一：关闭防火墙**

**Shell># service iptables stop #关闭防火墙**

**Shell># chkconfig iptables off #禁止开机启动**

**方案二：开放mysql客户端(3306)端口**

**Shell># vim /etc/sysconfig/iptables #编辑防火墙文件**

**添加一条规则：**

**-A INPUT -p tcp -m state --state NEW -m tcp --dport 3306 -j ACCEPT**

**Shell># service iptables restart**

## 七，mysql修改密码的方式

前言：在设置mysql密码的情况下，建议拒绝所有外部链接，关闭mysql端口、关闭apache服务后，在确保只有自己一人在操作数据库的时候设置。已确保数据的安全性

1. **update方式修改**

**mysql5.7及以上版本：**

**Mysql> UPDATE user SET authentication\_string \**

**= password('new-password') WHERE User = 'root' ;**

**mysql5.7以下版本：**

**Mysql> UPDATE user SET Password \**

**= password('new-password') WHERE User = 'root' ;**

1. **Set password方式修改**

**Mysql> Set password for ‘root’@’localhost’ \**

**=password(‘new\_password’);**

**或者**

**Mysql> Set password = password(‘new\_password’); 【设置当前用户密码】**

1. **Alter 方式修改**

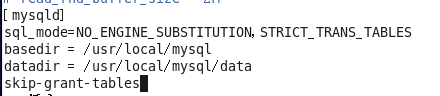
**Mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'new\_password';**

**忘记密码的话：**

**1，修改mysql的登陆设置**

**Shell># vim /etc/my.cnf**

**在[mysqld]的段中加上一句：skip-grant-tables**

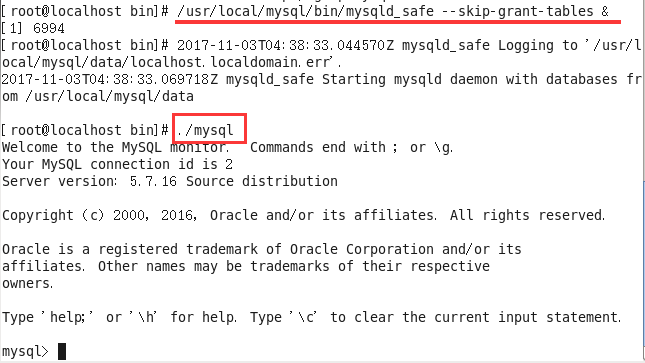
****

**保存退出。**

**或者**

**使用安全模式启动mysql**

**Shell># /usr/local/mysql/bin/mysqld\_safe --skip-grant-tables &**

****

**注：使用该方法的话，可以省略第2步。**

**2，重新启动mysqld**

**Shell># /etc/init.d/mysqld restart**

**或者**

**Shell># service mysqld restart**

**3，登陆并修改mysql的root密码**

**Shell># /usr/local/mysql/bin/mysql**

**Mysql> 修改密码...**

**mysql> flush privileges;**

**Mysql> quit;**

**4，将配置文件my.cnf中的 skip-grant-tables 注释掉，重启mysql**

**Shell># /etc/init.d/mysqld restart**

**或者**

**Shell># service mysqld restart**

**5，再次登陆mysql，并重新更改密码**

**Shell># /usr/local/mysql/bin/mysql -u root -p 第3步设置的密码**

**Mysql> set password = password('xxxxxxxx');**

**注：新密码必须包含大小写字母，特殊字符，数字**

**Mysql> flush privileges;**

**Mysql> quit;**

## 八，mysql grant语法详解

Grant语法命令：grant 权限 on 数据库对象 to 用户

一、grant 普通数据用户，查询、插入、更新、删除 数据库中所有表数据的权利。

grant select on testdb.\* to common\_user@'%'

grant insert on testdb.\* to common\_user@'%'

grant update on testdb.\* to common\_user@'%'

grant delete on testdb.\* to common\_user@'%'

或者，用一条 MySQL 命令来替代：

grant select, insert, update, delete on testdb.\* to common\_user@'%'

二、grant 数据库开发人员，创建表、索引、视图、存储过程、函数。。。等权限。

grant 创建、修改、删除 MySQL 数据表结构权限。

grant create on testdb.\* to developer@'192.168.0.%';

grant alter on testdb.\* to developer@'192.168.0.%';

grant drop on testdb.\* to developer@'192.168.0.%';

MySQLgrant 操作 MySQL 外键权限。

grant references on testdb.\* to developer@'192.168.0.%';

grant 操作 MySQL 临时表权限。

grant create temporary tables on testdb.\* to

grant 操作 MySQL 索引权限。

grant index on testdb.\* to

grant 操作 MySQL 视图、查看视图源代码 权限。

1，grant create view on testdb.\* to developer@'192.168.0.%';

2，grant show view on testdb.\* to developer@'192.168.0.%';

grant 操作 MySQL 存储过程、函数 权限。

1，grant create routine on testdb.\* to developer@'192.168.0.%';

2，grant alter routine on testdb.\* to developer@'192.168.0.%';

3，grant execute on testdb.\* to developer@'192.168.0.%';

三、grant 普通 DBA 管理某个 MySQL 数据库的权限。

MySQL> grant all privileges on testdb to

其中，关键字 “privileges” 可以省略。

grant execute on procedure testdb.pr\_add to 'dba'@'localhost'

grant execute on function testdb.fn\_add to 'dba'@'localhost'

grant all on \*.\* to dba@'localhost'

五、MySQL grant 权限，分别可以作用在多个层次上。

1. grant 作用在整个 MySQL 服务器上：

grant select on \*.\* to dba@localhost; -- dba 可以查询 MySQL 中所有数据库中的表。

grant all on \*.\* to dba@localhost; -- dba 可以管理 MySQL 中的所有数据库

2. grant 作用在单个数据库上：

grant select on testdb.\* to dba@localhost; -- dba 可以查询 testdb 中的表。

3. grant 作用在单个数据表上：

grant select, insert, update, delete on testdb.orders to dba@localhost;

4. grant 作用在表中的列上：

grant select(id, se, rank) on testdb.apache\_log to dba@localhost;

5.MySQL grant 作用在存储过程、函数上：

grant execute on procedure testdb.pr\_add to 'dba'@'localhost'

grant execute on function testdb.fn\_add to 'dba'@'localhost'

六、查看 MySQL 用户权限

1，查看当前用户（自己）权限：

show grants;

2，查看其他 MySQL 用户权限：

show grants for dba@localhost;

七、撤销已经赋予给 MySQL 用户权限的权限。

revoke 跟 grant 的语法差不多，只需要把关键字 “to” 换成 “from” 即可：

grant all on \*.\* to dba@localhost;

revoke all on \*.\* from dba@localhost;

八、MySQL grant、revoke 用户权限注意事项

1. grant, revoke 用户权限后，该用户只有重新连接 MySQL 数据库，权限才能生效。

2. 如果想让授权的用户，也可以将这些权限 MySQL grant 给其他用户，需要选项 “grant option“

grant select on testdb.\* to dba@localhost with grant option;

这个特性一般用不到。实际中，数据库权限最好由 DBA 来统一管理

## 九，mysql常见问题解决参考

<http://www.linuxidc.com/Linux/2013-03/81331.htm>

<http://www.linuxidc.com/Linux/2015-03/114803.htm>

<http://www.cnblogs.com/daizhuacai/archive/2013/01/17/2865138.html>