1. 源码安装5.7版本:

```
# 安装依赖:
yum install gcc ncurses-devel libaio bison gcc-c++  git
cmake  ncurses-devel openssl openssl-devel -y

wget
http://nchc.dl.sourceforge.net/project/boost/boost/1.59.0/
boost_1_59_0.tar.gz
tar xf boost_1_59_0.tar.gz
mv boost_1_59_0 /usr/local/boost
# Boost库是为C++语言标准库提供扩展的一些C++程序库的总称，由Boost社
区组织开发、维护

wget http://mirrors.163.com/mysql/Downloads/MySQL-
5.7/mysql-5.7.28.tar.gz
tar xf  mysql-5.7.28.tar.gz
cd mysql-5.7.28

cmake . -DCMAKE_INSTALL_PREFIX=/usr/local/mysql57 \
-DMYSQL_UNIX_ADDR=/data/mysql57/mysql.sock \
-DMYSQL_DATADIR=/data/mysql57 \
-DSYSCONFDIR=/usr/local/mysql57 \
-DMYSQL_USER=mysql \
-DMYSQL_TCP_PORT=3307 \
-DWITH_XTRADB_STORAGE_ENGINE=1 \
-DWITH_INNOBASE_STORAGE_ENGINE=1 \
-DWITH_PARTITION_STORAGE_ENGINE=1 \
-DWITH_BLACKHOLE_STORAGE_ENGINE=1 \
-DWITH_MYISAM_STORAGE_ENGINE=1 \
-DWITH_READLINE=1 \
-DENABLED_LOCAL_INFILE=1 \
-DWITH_EXTRA_CHARSETS=1 \
-DDEFAULT_CHARSET=utf8 \
-DDEFAULT_COLLATION=utf8_general_ci \
-DEXTRA_CHARSETS=all \
-DWITH_BIG_TABLES=1 \
-DWITH_DEBUG=0 \
-DENABLE_DTRACE=0 \
```

```
-DDOWNLOAD_BOOST=1 \
-DWITH_BOOST=/usr/local/boost

make && make install

mkdir -p /data/mysql57
useradd -s /sbin/nologin mysql
chown -R mysql. /data/mysql57

cp support-files/mysql.server /etc/init.d/mysqld57
chmod +x  /etc/init.d/mysqld57

vim /usr/local/mysql57/my.cnf
[mysqld]
basedir=/usr/local/mysql57/
datadir=/data/mysql57/
port=3307
pid-file=/data/mysql57/mysql.pid
socket=/data/mysql57/mysql.sock

[mysqld_safe]
log-error=/data/mysql57/mysql.log

#初始化
/usr/local/mysql57/bin/mysqld --initialize --user=mysql --
datadir=/data/mysql57 \
--basedir=/usr/local/mysql57/

/etc/init.d/mysqld57 start

## 登录后修改密码:
> alter user user() identified by "123";

### 或者跳过权限修改:
跳过权限:
vim /usr/local/mysql57/my.cnf
在[mysqld]字段下添加:
skip-grant-tables

然后重启服务:
/etc/init.d/mysqld restart
```

免密进入数据库：
如果只启动了一个数据库服务，可以直接用下面的命令进入，要是有多个服务启动，会默认进入3306的数据库，到时可以指定IP和端口进入。
/usr/local/mysql57/bin/mysql
更新密码为空：

```
update mysql.user set authentication_string=password('')
where user="root";

# MySQL5.7默认监听ipv6地址，用ipv4地址也是可以正常访问，如果想改为ipv4，可以在[mysqld]字段下配置一下参数，然后重启服务：
bind-address=0.0.0.0


## MySQL 5.7 授权，先创建用户，在授权：
mysql> create user "root"@"192.168.75.124" identified by "123456qaz";
Query OK, 0 rows affected (0.00 sec)

mysql> grant all on *.* to "root"@"192.168.75.124";
Query OK, 0 rows affected (0.00 sec)
```

2. mysql安装目录介绍：

```
# 源码安装mysql 版本：

mysql 主配置目录:/usr/local/mysql55
mysql 数据目录：/data/mysql
mysql 命令目录：/usr/local/mysql55/bin/* 比如：mysql、mysqld等。
mysql 配置文件：/usr/local/mysql55/my.cnf
mysql 启动文件：/usr/local/mysql55/support-files/mysql.server 或者
是/etc/init.d/mysqld
mysql 日志文件：/data/mysql



# yum 安装mariadb程序：

mariadb 主配置目录:/var/lib/mysql
```

```
mariadb 数据目录：/var/lib/mysql
mariadb 命令目录：/usr/bin
mariadb 默认配置文件：/etc/my.cnf
mariadb 启动文件：/usr/bin
mariadb 日志文件：/var/log/mariadb
```

3. 登录数据库服务器：

```
# 通过unix套接字连接
直接通过mysql 或者mysql -uroot -p 登录

# 查看连接状态：
status

MariaDB [(none)]> status;
--------------
mysql  Ver 15.1 Distrib 5.5.64-MariaDB, for Linux (x86_64)
using readline 5.1

# 当前使用的那个数据库，没有选择为空：
Current database:
# 是否使用加密
SSL:            Not in use
Current pager:      stdout
Using outfile:      ''
# 结束符为分号
Using delimiter:    ;
Server:         MariaDB
Server version:     5.5.64-MariaDB MariaDB Server
Protocol version:   10
# 连接方式，本地套接字
Connection:     Localhost via UNIX socket
Server characterset:    latin1
Db     characterset:    latin1
Client characterset:    utf8
Conn.  characterset:    utf8
# 套接字地址
UNIX socket:        /var/lib/mysql/mysql.sock
Uptime:         7 min 21 sec
```

```
# 通过tcp套接字连接
通过mysql  -h127.0.0.1 登录服务器，查看状态：

可以看到连接id不同，套接字也不同，使用的是tcp/ip的套接字通信。
如果有时候遇到无法通过本地套接字连接，可以使用指定服务器ip连接。
```

4. 常用命令操作：

4.1 数据库的操作命令：

```
# 查询数据库：
show  databases:
#  初始化后，默认会有四个数据库：
information_schema：信息数据库。主要保存着关于MySQL服务器所维护的所有其他数据库的信息，如数据库名，数据库的表，表栏的数据类型与访问权限等。通过show databases;查看到数据库信息，也是出自该数据库中得SCHEMATA表。
mysql：  mysql的核心数据库。主要负责存储数据库的用户、权限设置、关键字等mysql自己需要使用的控制和管理信息。
performance_schema：  用于性能优化的数据库。
# 查看数据库的创建语句：
show create database mysql;
mysql为数据库名。

# 查看字符集命令：
show character  set;

# 修改数据库的字符集：
alter database zabbix default character set utf8;

# 创建数据库：
create  database zabbix charset=utf8;
或者：
create database if not exists zabbix charset=gbk;
用上面的这条命令创建数据库，如果数据库已经存在就不会报错了。

# 查看警告：
show warnings;

# 删除数据库：
drop database zabbix;
```

或者：
```
drop database if exists zabbix;
```
用上面这种方式删除数据库，如果数据库不存在就不会报错了。

4.2 创建表命令：

```
create table t1( id  int(10) auto_increment primary key,
name varchar(20), job varchar(10) );
```

4.3 查看表结构相关命令：

查看所有表：
```
use mysql;
show tables;
```
或者
```
show  tables from mysql;
```

查看所有表的详细信息：
```
use mysql;
show table status\G
```
或者
```
show table status from mysql\G
```

查看某张表的详细信息：
```
use mysql;
show table status like "user"\G
```
或者
```
show table status from mysql like "user"\G
```

查看表结构：
```
desc  mysql.user;
```

查看创建表的sql语句：
```
show create table mysql.user\G
```

4.4 修改表结构相关命令：

添加表字段：
```
alter  table t1 add job varchar(20);
```

默认是加在后面，如果想加在第一列，或者某个字段后可以进行指定：

```
#加在第一列
alter  table t1 add job2 varchar(20) first;
#加在name字段后：
alter  table t1 add job3 varchar(20) after name;
```

修改表字段名，需要将字段属性写全：

```
alter table t1 change id age int(5);
id:      原字段
age:     新字段
```

修改表字段的属性或者位置：

```
#修改字段的顺序，把job3放在第一列：
alter table t1 modify job3 varchar(20) first;
```

删除表字段：

```
alter table t1 drop brith1;
```

4.5 增:

```
# 全字段增加数据：
insert into t1 values(1,"xiaoming","it");
或者
insert t1(name) values(2，"xiaohua");
或者
insert t1 set
name="xiaoming",
job="teacher";

# 指定字段增加数据：
insert into t1(name) value("xiaoqiang"),("xiaowang");
```

4.6 删:

```
#物理删除，数据就真没有了
delete from t1 where id=2;

#逻辑删除，需要添加一个字段，默认设置为0：
alter table t1 add isdelete bit default 0;

#将isdelete字段设置为1：
update t1 set isdelete=1 where id=3;

#然后查找isdelete字段为0得数据即可过滤了：
select * from t1 where isdelete=0;
```

4.7 改:

```
#修改表中得数据，不增加行,insert into会增加行：
update t1 set name="xiaoxiao" where id=6;
t1是表名，
name是字段名，
where后面是条件语句，如果没有，就是对整个表得修改了，要慎重!

#修改多个字段用逗号分隔：
update t1 set name="xiaoqiang",job="engineer" where id=3;
```

4.8 查:

```
全字段查找，不建议：
select * from t2;

查找指定字段：
select name,job from t2;
```

根据运算符查找:

```
>    大于
>=   大于等于
=    等于
<    小于
<=   小于等于
!=   不等于

查找id大于等于4的用户id,用户名及工作：
```

```
select id,name,job from t2 where id >= 4;

and      多个条件同时满足
or       几个条件，满足其一即可

查找id大于等于3，并且没有标记删除的数据
select id,name,job,isdelete  from t2 where id >= 3 and
isdelete=0;
```

模糊查找:

```
like         模糊查找
%        匹配任意多个字符
_        匹配单个字符

查找名字中含有xiao字符的数据：
select id,name,job from t2 where name like "xiao%";

插入一条数据，查找名字中含有xiao并且后面跟一个单字符的数据
insert t2(name) value("xiaom");
select id,name,job from t2 where name like "xiao_";
```

范围查找:

```
in                    表示非连续的范围
between A and B       表示一个连续的范围内
not                   不在某个条件内

查找id等于2，4或者6的数据
select id,name,job from t2 where id in (2,4,6);

查找id在3到6之间数据（包含3和6）：
select id,name,job from t2 where id between 3 and 6;

查找id不等于3，6的数据：
select id,name,job from t2 where id not in (3,6);
```

查找空值:

```
is null      查找空值

select id,name,job from t2 where job is null;
```

聚合：

```
select max(age) from t2;
select min(age) from t2;
select count(*) from t2 where isdelete=0;
```

排序：

```
order by  字段  asc        根据“列”从小到大排序
order by  字段  desc       根据“列”从大到小排序

根据id，从大到小进行排序：
select * from t2 order by id desc;
```

分组：

```
group by 字段

#根据名字分组，统计同名的个数：
select count(*),name  from t2 group by name;
```

限制：

```
limit n;    显示前n行

#显示前三行
select * from t2 limit 3;
#显示从第3行开始后的3行；
select * from t2 limit 3,3;
#显示从第3行开始后的4行
select * from t2 limit 4 offset 3;
```

5. 视图的相关操作：

视图并不是真实存在的表，主要是将常用的到字段或者数据整合成一个“表”。

5.1 创建视图：

```
# 先创建表：
create table t1 ( id int(10) not null auto_increment
primary key, name varchar(20), job varchar(10) );
# 插入数据：
insert t1 set name="xiaoming", job="it";
insert t1 set name="xiaowang", job="it";
insert t1 set name="xiaohong", job="it";
# 创建视图：
create view v1 as select name,job from t1;
create view v2 as select * from t1 where id >= 2;
```

5.2 查看视图：

```
# 查看所有视图：
select * from information_schema.views where
table_schema="test"\G
# 查看某个视图结构：
desc v2;
# 查看视图内容：
select * from v2;
```

5.3 删除视图：

```
drop   view v2;
或者
drop view  if exists v2;
```

5.4 修改视图：

```
alter view v1 as select id,job from t1;
```

6. 修改密码：

6.1 密码为空或者已知当前密码：

方法一：用SET PASSWORD命令
首先登录MySQL。
格式：mysql> set password for 用户名@localhost = password('新密码');
范例：
```
mysql> set password for root@localhost = password('123');
```

方法二：在shell终端执行：

```
mysql -uroot -p -e "set password for root@localhost=password('123');"
```

ps:注意双引号和单引号！！！

方法三：在shell终端用mysqladmin

格式：mysqladmin -u用户名 -p旧密码 password 新密码

范例：
```
mysqladmin -uroot -p123456 password 123
```

方法四：用UPDATE直接编辑user表
首先登录MySQL。
```
mysql> use mysql;
mysql> update user set password=password('123') where user='root' and host='localhost';
mysql> flush privileges;
```

6.2 忘记密码：

```
# yum安装的mysql

可执行使用/usr/bin/mysqld_safe --user=mysql --skip-grant-tables &

然后用mysql命令进入修改密码：
update mysql.user set password=password('') where user="root" and host="localhost";
flush privileges;
```