### mariadb操作手册

**前言：**

1.蓝色字体表示命令行命令，正式执行时不要复制前面的#号，#号只是提示应该使用root权限操作

2.绿色字体表示注释，有时注释太多就不用绿色表示了

3.注意：本文档的所有操作请先在测试机里进行实践，请不要直接在真实的服务器中操作！

**版权声明**：

本文档以开源的形式发布，所有条款如下：

1. 无担保：作者不保证文档内容的准确无误，亦不承担由于使用此文档所导致的任何后果

2. 自由使用：任何人可以出于任何目的而自由地 阅读/链接/打印/转载/引用/再创作 此文档，无需任何附加条件

若您 阅读/链接/打印/转载/引用/再创作 本文档，则说明接受以上2个条款。

作者：李茂福

更新日期：2023-07-27

**关系型数据库介绍**

RDBMS（Relational DataBase Management System）关系型数据库管理系统

是按照数据结构来组织、存储和管理数据的仓库，RDBMS的特点：

1.数据以表格的形式出现

2.每行为各种记录名称

3.每列为记录名称所对应的数据域

4.若干行和列组成一张表单

5.若干表组成一个database

**章一、安装mariaDB**

# yum install mariadb mariadb-server -y

# systemctl enable mariadb

# systemctl start mariadb

**#初始化设置**

# mysql\_secure\_installation

Enter current password for root (enter for none): #直接回车，初始安装后root用户无密码

Set root password? [Y/n] y #输入y，设置root密码

New password: #输入root用户密码，不回显

Re-enter new password: #再次输入root用户密码，不回显

Password updated successfully!

Remove anonymous users? [Y/n] y #输入y，删除匿名用户

... Success!

Disallow root login remotely? [Y/n] y #禁止root用户从远程登录

... Success!

Remove test database and access to it? [Y/n] y #删除默认的test数据库

Reload privilege tables now? [Y/n] y #重新加载权限表

... Success!

Thanks for using MariaDB!

# ss -ano | grep 3306 # mariadb运行后，默认监听所有ip的3306/tcp端口

# rpm -qc mariadb-server #查看配置文件

**主配置文件有：**

/etc/my.cnf #centos7-->**5.5.68**

/etc/my.cnf.d/mariadb-server.cnf #centos8-->**10.3.27**

本文档以centos7-->mariadb 5.5.68为示例，主配置文件为/etc/my.cnf，当版本为centos8时需要修改主配置文件路径为/etc/my.cnf.d/mariadb-server.cnf

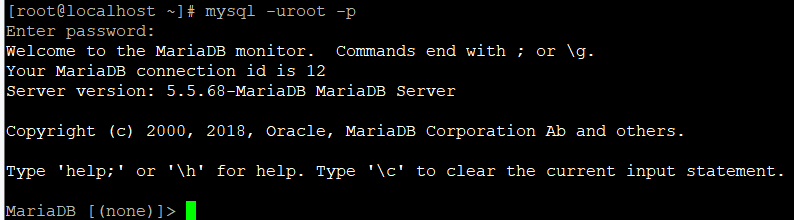
**章二、设置数据库字符集**

**#先登录数据库，查看默认字符集**

# mysql -uroot -p密码 # -p后直接跟密码，回车就进入数据库交互界面

#也可不在 -p后跟密码，防止被人看到，只输入 mysql -uroot -p 就回车，会提示要输入密码，这时输入密码不回显

出现 MariaDB [(none)]> 提示符表示已进入数据库交互界面



MariaDB [(none)]> show variables like "%character%"; #查看字符集设置

+--------------------------+----------------------------+

| Variable\_name | Value |

| character\_set\_client | utf8 |

| character\_set\_connection | utf8 |

| character\_set\_database | latin1 |

| character\_set\_filesystem | binary |

| character\_set\_results | utf8 |

| character\_set\_server | latin1 |

| character\_set\_system | utf8 |

| character\_sets\_dir | /usr/share/mysql/charsets/ |

MariaDB [(none)]> show variables like "%collation%";

+----------------------+-------------------+

| Variable\_name | Value |

| collation\_connection | utf8\_general\_ci |

| collation\_database | latin1\_swedish\_ci |

| collation\_server | latin1\_swedish\_ci |

MariaDB [(none)]> \q #退出数据库交互界面

MariaDB [(none)]> exit #退出数据库交互界面

MariaDB [(none)]> quit #退出数据库交互界面

# mysql -uroot -p -e 'show variables like "%character%";' #一条命令执行

**#设置数据库字符集**

# vi /etc/my.cnf #在[mysqld]下面添加以下5行内容，保存退出

init\_connect='SET collation\_connection = utf8\_unicode\_ci'

init\_connect='SET NAMES utf8'

character-set-server=utf8

collation-server=utf8\_unicode\_ci

skip-character-set-client-handshake

# vi /etc/my.cnf.d/client.cnf #在[client]下添加一行：

default-character-set=utf8



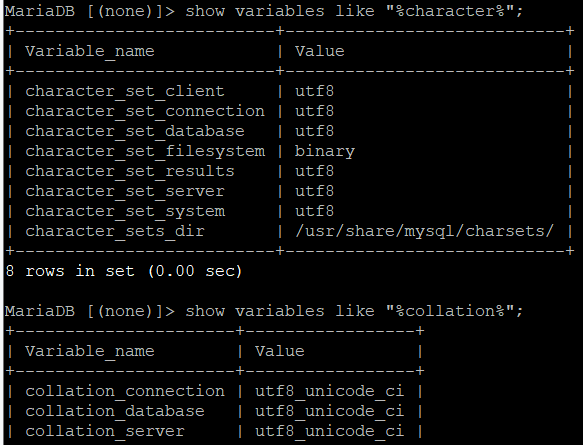
# vi /etc/my.cnf.d/mysql-clients.cnf #在[mysql]下添加一行：

default-character-set=utf8



# systemctl restart mariadb #重启数据库服务

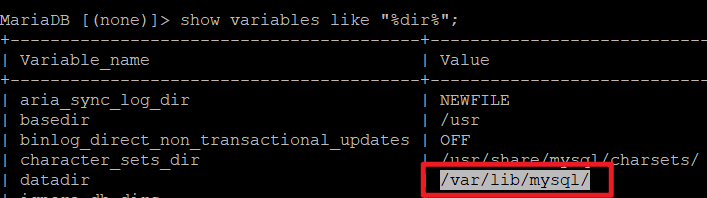
再重新登录，查看字符集设置



**章三、设置数据存储路径**

**#先查看默认数据存储路径**，进入mariadb交互界面后执行以下命令：

MariaDB [(none)]> show variables like "%dir%";



#可见mariadb数据默认是存放在/var/lib/mysql/路径下，一般得单独创建一个数据存放目录，挂载单独的磁盘空间

# mkdir /mariadb\_data #创建数据存放目录

# semanage fcontext -a -t mysqld\_db\_t "/mariadb\_data(/.\*)?"

# systemctl stop mariadb #先停止数据库的运行

# cp -rp /var/lib/mysql/\* /mariadb\_data/ #复制原存储路径下所有文件到新目录下

# chown -R mysql:mysql /mariadb\_data #修改新目录的属主为mysql

# chcon -R -u system\_u /mariadb\_data #修改目录及其下所有文件selinux标签

# chcon -R -t mysqld\_db\_t /mariadb\_data #修改目录及其下所有文件selinux标签

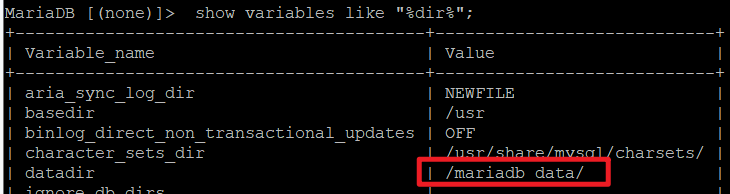
# vi /etc/my.cnf #将[mysqld]下的 datadir值改为新的路径

datadir=/mariadb\_data



# systemctl restart mariadb #重启服务

再登录数据库交互界面查看数据存放路径



**章四、设置监听的ip及端口号**

mariadb默认监听所有ip（0.0.0.0）及3306/tcp端口

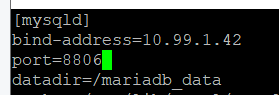


可以修改成只监听某个ip及其他端口号，如8806

# vi /etc/my.cnf #在[mysqld]下面添加2行

bind-address=10.99.1.42

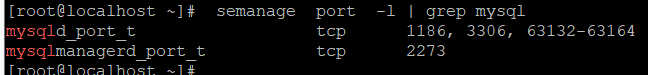
port=8806



如果开启了selinux，得设置8806端口的selinux标签

# yum install policycoreutils-python #先安装此包才有semanage命令

# semanage port -l | grep mysql #可见默认3306端口标签为mysqld\_port\_t

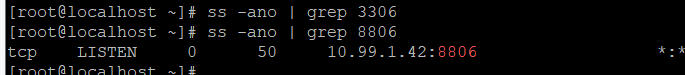


得把8806也打上mysqld\_port\_t这个标签

# semanage port -a -t mysqld\_port\_t -p tcp 8806

最后重启mariadb服务

# systemctl restart mariadb



**★不监听网络端口，只能从本地登录**

# vi /etc/my.cnf #在[mysqld]下面添加1行

skip-networking=1

**章五、创建用户并授权**

第一章操作完成后，默认root用户只可在本地登录，要想让root用户能远程连接数据库，得设置允许它登录的ip地址，一般不让root用户远程登录，防止其他人去试root用户的密码；可创建新的其他用户，分配不同的权限，如只可查询或所有权限等

**①创建用户并指定允许登录的地址**

# mysql -uroot -p #先在本地登录数据库交互界面

MariaDB [(none)]> create user 'cof'@'10.99.1/24' identified by 'mypasswdxx';

#创建一个用户，名为cof，允许从ip网段为10.99.1.0/24的地址远程登录，在mariadb命令这里网段只可写成10.99.1/24，不可写成10.99.1.0/24这种格式；在identified by 后指定用户密码

MariaDB [(none)]> create user 'cof'@'%' identified by 'mypasswdxx';

# @后的'%'表示允许从任意ip地址远程登录

MariaDB [(none)]> create user 'cof'@'localhost' identified by 'mypassyyyy';

# 'localhost'表示只允许本地登录，后面的密码和前2条记录可以不同

创建用户时，同一用户名可有多条记录，针对不同的远程登录地址可以使用不同的密码，例如 只允许本地登录时使用密码mypassyyyy，而远程登录则使用密码mypasswdxx

**②授权**

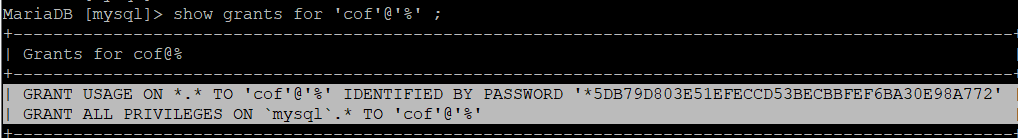
MariaDB [(none)]> grant all privileges on mysql.\* to cof;

#给cof用户授权，可操作mysql库的所有表（具有所有权限）privileges on 后面跟库名.表名，\*.\*表示任意/所有库及所有表

MariaDB [(none)]> grant select,insert,update,delete,create,drop on testdb.\* to userxx;

#授予userxx用户对testdb库.所有表的select,insert,update,delete,create,drop这6个操作权限

MariaDB [mysql]> show grants for 'cof'@'%' ; #查看cof用户的权限



**③更改用户密码**

MariaDB [(none)]> use mysql; #使用mysql这个库，它存放了用户信息

MariaDB [mysql]> update user set password=password("newpasswd") \

where user='cof' and host='%';

#更新cof用户远程登录时的密码，\反斜杠表示断行，即上面2行是当成一行来执行的

MariaDB [mysql]> flush privileges; #重载授权表，使上面授权的操作生效

**④查看数据库所有用户（在mysql库的user表）**

MariaDB [mysql]> select Host,User,Password from mysql.user;

+------------+--------+-------------------------------------------+

| Host | User | Password |

+------------+--------+-------------------------------------------+

| localhost | root | \*D896429FDEDF45DEA93172723EDC0B0CA63C7E19 |

| 127.0.0.1 | root | \*D896429FDEDF45DEA93172723EDC0B0CA63C7E19 |

| ::1 | root | \*D896429FDEDF45DEA93172723EDC0B0CA63C7E19 |

| % | cof | \*5DB79D803E51EFECCD53BECBBFEF6BA30E98A772 |

| 10.99.1/24 | cof | \*5DB79D803E51EFECCD53BECBBFEF6BA30E98A772 |

| localhost | cof | \*87230EC03251F2AA26A5A40680D7F070D95AA266 |

| % | userxx | |

+------------+--------+-------------------------------------------+

**⑤删除用户记录**

MariaDB [(none)]> delete from mysql.user where user='cof' and host='localhost';

MariaDB [(none)]> flush privileges;

**⑥重置root用户密码**

有时忘记了root用户密码，可先停止mariadb服务的运行

# systemctl stop mariadb

# mysqld\_safe --skip-grant-tables & #运行时跳过权限表的加载

# mysql #这时可直接输入mysql命令进入数据库交互界面

[]> update mysql.user set password=password('newpasswdx') where User='root';

#更新用户名为root的用户的所有记录的密码

MariaDB [(none)]> flush privileges; #重新加载权限表

MariaDB [(none)]> exit

# mysqladmin -uroot -p shutdown #停止服务运行，要求输入刚刚修改后的新密码

Enter password:

# systemctl restart mariadb #最后重启mariadb服务

**章六、使用客户端工具远程登录数据库**

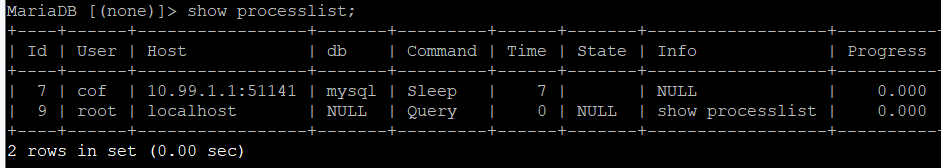
在数据库服务器上可直接使用mysql命令登录，也可在其他linux服务器使用mysql命令去远程登录到目标数据库服务，需要指定目标服务器地址及端口号（前提是要先安装mysql/mariadb）

数据库所在服务器的防火墙要放通相应端口号，如8806/tcp（默认为3306/tcp）

[root@localhost ~]# mysql -ucof -h10.99.1.42 -P8806 -pmypasswdxx

# mysql -u用户名 -h数据库服务器地址 -P端口号 -p密码，前提是此用户允许从执行这条命令的服务器ip地址去远程登录

MariaDB [(none)]> show processlist; #查看当前连接的所有用户

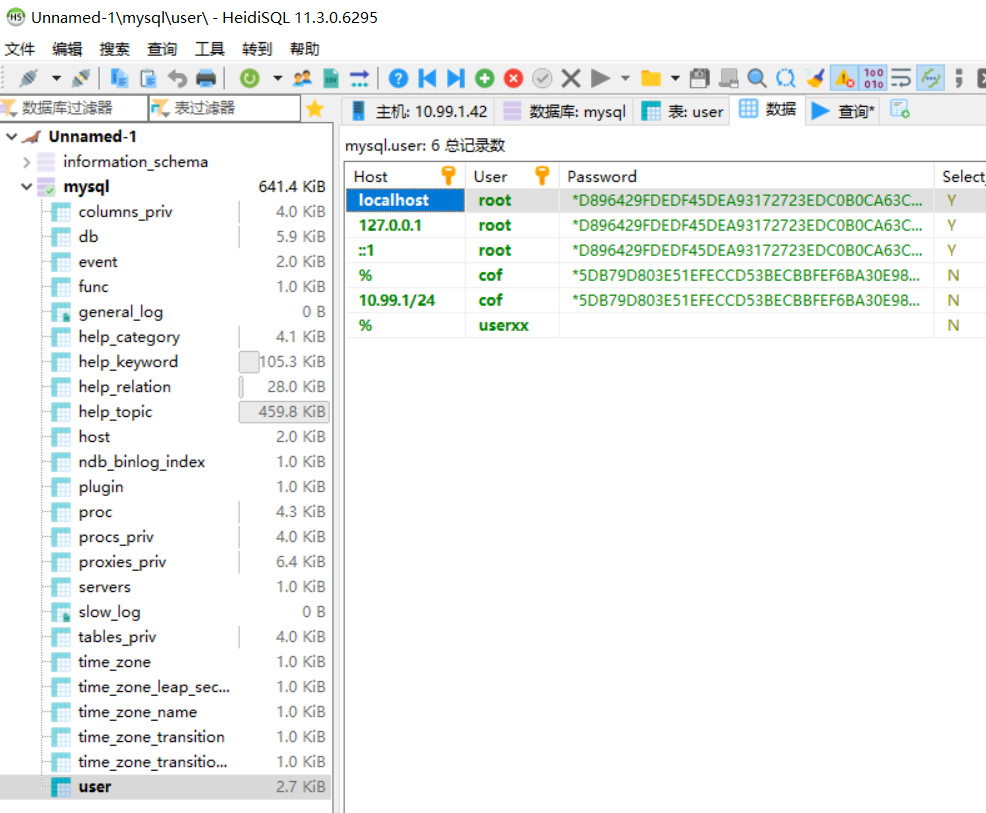


在windows上也可安装mysql/mariadb然后使用命令远程登录，也可使用 可视化的mysql/mariadb客户端工具去远程登录数据库并进行操作

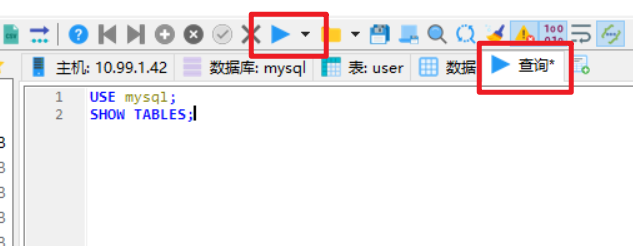
推荐使用开源的 HeidiSQL 客户端软件，下载地址 https://www.heidisql.com/

安装后双击图标打开，填写连接信息





可以在查询框里进行sql语句操作



也有其他商业的 可视化数据库客户端连接工具，这里不一一列出

**章七、数据库及表基础操作**

MariaDB [(none)]> show databases; #列出所有数据库

+--------------------+

| Database |

+--------------------+

| information\_schema |

| mysql |

| performance\_schema |

+--------------------+

3 rows in set (0.00 sec)

上面查询结果可见一共有3个库

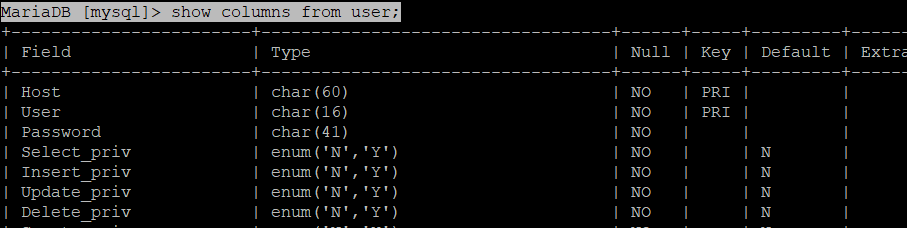
MariaDB [(none)]> use mysql; #使用名为mysql的库

MariaDB [mysql]> #然后提示符[]括号里由none变成mysql库名了

MariaDB [mysql]> show tables; #在mysql库名下列出所有的 表

MariaDB [mysql]> show columns from user; #查看user表的所有列（字段属性）

MariaDB [mysql]> show columns from mysql.user; #同上一句



MariaDB [(none)]> show tables from mysql; #也可直接查询指定库的所有表

MariaDB [db\_testy]> show create table db\_testy.tnamex \G ; #查看表的完整结构

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Table: tnamex

Create Table: CREATE TABLE `tnamex` (

`id` int(11) DEFAULT NULL,

`name` char(30) DEFAULT NULL,

`age` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8

1 row in set (0.00 sec)

**#库 操作**

MariaDB [mysql]> create database db\_testx; #创建名为db\_testx的数据库

MariaDB [mysql]> drop database db\_testx; #删除名为db\_testx的数据库

#也可在数据库服务器的linux命令行去操作

[root@localhost ~]# mysqladmin -uroot -p create db\_testx #创建名为db\_testx的数据库

Enter password: #要求输入密码

[root@localhost ~]#

[root@localhost ~]# mysqladmin -uroot -p drop db\_testx #删除名为db\_testx的数据库

Enter password: #要求输入密码

Dropping the database is potentially a very bad thing to do.

Any data stored in the database will be destroyed.

Do you really want to drop the 'db\_testx' database [y/N] y #输入y，回车，表示确定要删除

Database "db\_testx" dropped

**#表 操作**

MariaDB [mysql]> create database db\_testy; #创建名为db\_testy的数据库

MariaDB [(none)]> use db\_testy; #使用db\_testy库

MariaDB [db\_testy]> create table tb\_namex ( id INT, name CHAR(30), age INT ) \

ENGINE=InnoDB DEFAULT CHARSET=utf8; #在db\_testy库下创建一个表，表名为tb\_namex

MariaDB [db\_testy]> create table db\_testy.tb\_namey ( id INT, name CHAR(30), age INT ) \

ENGINE=InnoDB DEFAULT CHARSET=utf8; #在指定数据库下创建表tb\_namey

MariaDB [db\_testy]> drop table db\_testy.tb\_namey; #删除db\_testy库的tb\_namey表

MariaDB [(none)]> alter table db\_testy.tb\_namex rename to db\_testy.tb\_namex2; #修改表名称

**#复制表**

MariaDB [db\_testy]> create table db\_testy.newtable like db\_testy.tb\_namex2;

#创建名为newtable的新表，其结构复制tb\_namex2的

MariaDB [db\_testy]> insert into db\_testy.newtable select \* from db\_testy.tb\_namex2;

#复制所有数据

[db\_testy]> insert into db\_testy.newtable (id, name) select id,name from db\_testy.tb\_namex2;

#复制指定列的数据

**章八、表 字段操作**

MariaDB [(none)]> show create table db\_testy.tb\_namex2 \G ; #首先查看表的结构

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Table: tb\_namex2

Create Table: CREATE TABLE `tb\_namex2` (

`id` int(11) DEFAULT NULL,

`name` char(30) DEFAULT NULL,

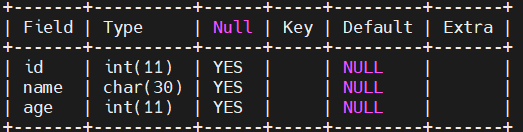
`age` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8

1 row in set (0.00 sec)

再进行其他关于表的字段（列）的操作

MariaDB [(none)]> show columns from db\_testy.tb\_namex2; #查看tb\_namex2表的所有列（字段属性）



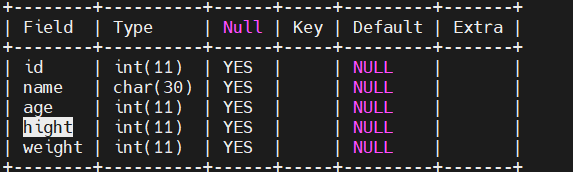
①增加字段

MariaDB [(none)]> alter table db\_testy.tb\_namex2 add weight INT ;

#给指定的表添加一个字段，名为weight，类型为INT（默认新添加的列是追加在原字段结构最后面）

MariaDB [(none)]> alter table db\_testy.tb\_namex2 add hight INT AFTER age;

#新增一个字段（名为hight），并用AFTER指定在age列的后面



MariaDB [(none)]> alter table db\_testy.tb\_namex2 add newid INT FIRST;

#新增一个字段（名为newid），排在最前面

②修改字段顺序

MariaDB [(none)]> alter table db\_testy.tb\_namex2 modify newid INT AFTER id ;

#调整字段顺序，将newid这字段放到id字段的后面

③删除字段

MariaDB [(none)]> alter table db\_testy.tb\_namex2 drop newid; #删除名为newid的字段

④仅修改字段类型

MariaDB [(none)]> alter table db\_testy.tb\_namex2 modify name char(32) NOT NULL ;

⑤修改字段名称[及类型]

[(none)]> alter table db\_testy.tb\_namex2 change name username char(32) NOT NULL ;

#把字段name改名为username

⑥修改字段默认值

MariaDB [(none)]> alter table db\_testy.tb\_namex2 alter age set DEFAULT 18; #默认值为18

⑦删除字段的默认值

MariaDB [(none)]> alter table db\_testy.tb\_namex2 alter age drop DEFAULT ;

**章九、表 数据操作**

MariaDB [(none)]> show create table db\_testy.tb\_namex2 \G ; #首先查看表的结构

Create Table: CREATE TABLE `tb\_namex2` (

`id` int(11) DEFAULT NULL,

`username` char(32) NOT NULL,

`age` int(11) DEFAULT '18',

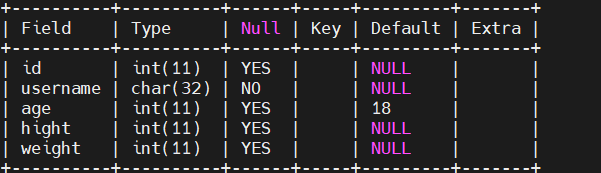
`hight` int(11) DEFAULT NULL,

`weight` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8

1 row in set (0.00 sec)

MariaDB [(none)]> show columns from db\_testy.tb\_namex2; #查看tb\_namex2表的所有列（字段属性）



**①插入数据**

> insert into db\_testy.tb\_namex2 (id,username,age,hight) values (12,"wangx",23,177) ;

> insert into db\_testy.tb\_namex2 (id,username,age,hight) values (9,"coflee",22,178) ;

> insert into db\_testy.tb\_namex2 (id,username,age,hight) values (14,"wxfds",23,177) ;

**②删除数据**

MariaDB [(none)]> delete from db\_testy.tb\_namex2 where id=17 ;

#删除id=17的所有记录（行）

**③更新数据**

MariaDB [(none)]> update db\_testy.tb\_namex2 set id=9,age=28 where username='coflee';

MariaDB [(none)]> update db\_testy.tb\_namex2 set weight=140; #所有记录的weight都设置为140

MariaDB [(none)]> update db\_testy.tb\_namex2 set username=replace(username,"w","nn");

#将username列的值中含有w的字符都替换为nn

**④查询记录**

MariaDB [(none)]> select \* from db\_testy.tb\_namex2; #查询指定表的所有记录（行）

| id | username | age | hight | weight |

+------+----------+------+-------+--------+

| 12 | nnangx | 23 | 177 | 140 |

| 9 | coflee | 28 | 178 | 140 |

| 14 | nnxfds | 23 | 177 | 140 |

3 rows in set (0.00 sec)

MariaDB [(none)]> select count(\*) from db\_testy.tb\_namex2; #统计查询出的行数

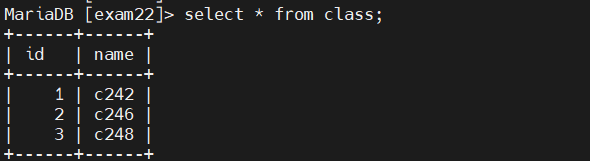
| count(\*) |

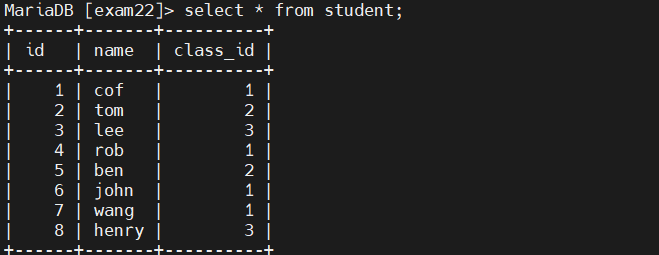
+----------+

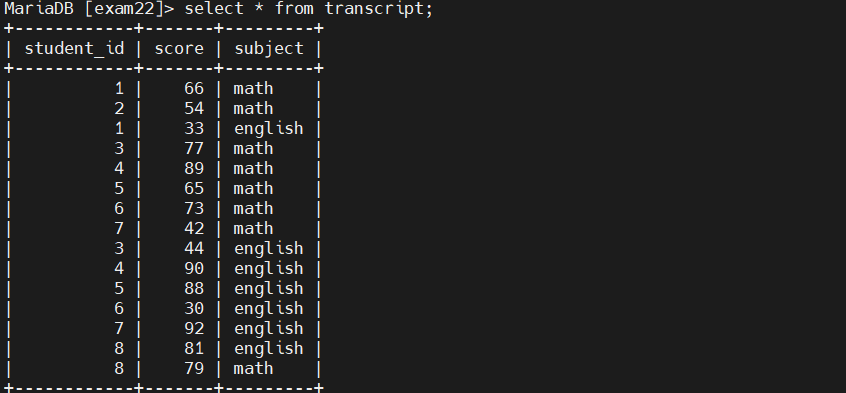
| 3 |

**⑤多表联查**

有三张表：class、student、transcript







要求查找c242班级的数学分数大于60分的同学，命令如下

> select \* from student,class,transcript where \

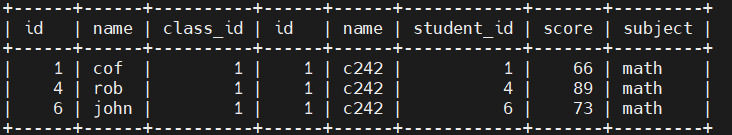
student.class\_id=class.id and \

student.id=transcript.student\_id and \

transcript.subject='math' and \

transcript.score > 60 and \

class.name='c242';

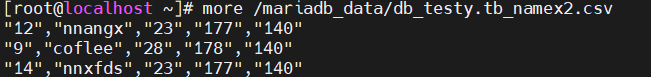


**章十、数据导入导出**

①导出表为.csv文件

> select \* from db\_testy.tb\_namex2 into outfile '/mariadb\_data/db\_testy.tb\_namex2.csv' \

fields terminated by ',' enclosed by '"' lines terminated by '\n';



②导入csv文件到指定表

MariaDB [(none)]> load data local infile 'dump.csv' INTO TABLE db\_testy.tb\_namex2 \

fields terminated by ',' enclosed by '"' lines terminated by '\n';

②导出指定库的指定表到sql文件

# mysqldump -uroot -p db\_testy tb\_namex2 > db\_testy-tb\_namex2.dump.sql

#导出的文件内容是创建tb\_namex2这个表以及插入这个表的所有数据的sql语句；要求输入密码

③导出指定库的所有表到sql文件

# mysqldump -uroot -p db\_testy > db\_testy-all.dump.sql

③导出所有数据库 数据到sql文件

# mysqldump -uroot -p --all-databases > all-database.dump.sql

# mysqldump -h 10.99.1.42 -P 8806 -uroot -p db\_testy > db\_testy-all.sql

#导出远程服务器上的指定库到本地的db\_testy-all.sql文件

**⑤导入sql数据到指定数据库下**

# mysql -uroot -p dbxx < /tmp/db\_testy-tb\_namex2.dump.sql

#dbxx库已创建，但里面没有tb\_namex2这个表，把这个表导入到dbxx库里（如果dbxx库里已有tb\_namex2表，则会先删除此表，再新建tb\_namex2表，再插入数据）

★导入整个数据库的dump.sql文件到某数据库下也是一样的原理

**章十一、主从同步**

①在主服务器上设置server-id和log-bin二进制日志

# vi /etc/my.cnf #在[mysqld]下面添加以下3行

server-id=1

log-bin=mysql-bin

slave-skip-errors=1062 #跳过重复字段错误

# systemctl restart mariadb #重启服务，使配置生效

②在主服务器上创建一个用于同步的用户并赋予复制权限

# mysql -uroot -p #在本地登录数据库交互界面

> grant replication slave on \*.\* to 'replicator'@'%' identified by 'passwdxxx';

> flush privileges;

> show master status; #记下File及Position的值，在备份服务器上要用

| File | Position | Binlog\_Do\_DB | Binlog\_Ignore\_DB |

+------------------+----------+--------------+------------------+

| mysql-bin.000001 | 245 | | |

+------------------+----------+--------------+------------------+

1 row in set (0.00 sec)

③主备服务器要放开防火墙相应端口

④在备份服务器上设置server-id

# vi /etc/my.cnf #在[mysqld]下面添加以下3行

server-id=2

log-bin=mysql-bin

slave-skip-errors=1062 #跳过重复字段错误

# systemctl restart mariadb #重启服务，使配置生效

⑤在备份服务器上设置主服务器连接信息

备服务器> change master to master\_host='主服务器ip', \

master\_user='replicator',master\_password='passwdxxx', \

master\_log\_file='mysql-bin.000001',master\_log\_pos=245 ;

#在主服务器上查看，MASTER\_LOG\_FILE的值对应File, MASTER\_LOG\_POS对应Position

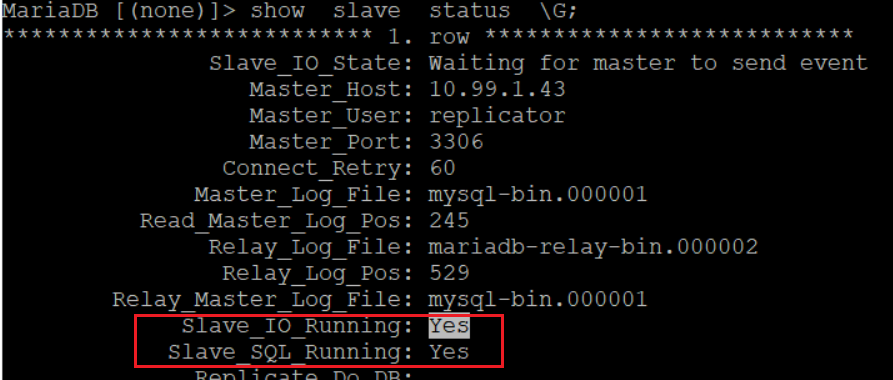
> start slave ; #在备份服务器上启动复制进程

> show slave status \G; #查看复制进程状态

**#出现以下2行说明配置同步成功**

Slave\_IO\_Running: Yes

Slave\_SQL\_Running: Yes



同理，如果主服务器要去同步备服务器的数据，则在备服务器上也创建一个用于同步的账号，并查看master\_log\_file及master\_log\_pos，最后在主服务器上配置要同步的信息即可：

主服务器> change master to master\_host='备服务器ip', \

master\_user='replicator',master\_password='passwdxxx', \

master\_log\_file='*mysql-bin.00000xx*',master\_log\_pos=*xxx* ;

> start slave ; #在主服务器上启动复制进程

> show slave status \G; #查看复制进程状态

**★关闭同步：**

> stop slave; #停止复制进程

> reset slave all; #清空主从信息

★备份服务器设置为只读数据库

> set global read\_only=ON;

> set global super\_read\_only=ON;

要退出只读则设置为off即可