第一部分：

在终端连接 MySQL 服务器，MySQL 是作为一个守护进程存在

mysql -h 127.0.0.1 --user=root --password=admin

show databases; //查看有那些数据库

创建数据库

CREATE DATABASE test DEFAULT CHARACTER SET UTF8;

show tables; //查看数据库里的表

//数据类型

1)数值类型

BIT 1～64

TINYINT 表示很小的整数 -128～127 无符号范围是 0～255

BOOL 0 表示假，非 0 表示真

SMALLINT 带符号的范围是-32768～32767；无符号的范围是 0～65535

MEDIUMINT 带符号的范围是-8388608～8388607；无符号的范围是 0～16777215

INT 带符号的范围是-2147483648～2147483647；无符号的范围是 0～4294967295

BIGINT 带符号的范围是-9223372036854775808～9223372036854775807；无符号的范围是 0～18446744073709551615

FLOAT FLOAT(M, D); M 表示总共的位数，D 表示小数点后的位数

DOUBLE DOUBLE(M, D); M 表示总共的位数，D 表示小数点后的位数

2)日期和时间类型

DATE 支持的范围为 '1000-01-01' 到 '9999-12-31'

DATETIME 日期和时间的组合, 支持的范围是 '1000-01-01 00:00:00' 到 '9999-12-31 23:59:59'

TIME 范围是'-838:59:59'到'838:59:59'

YEAR 1901 到 2155

3)字符串类型

CHAR(M) 固定长度字符串 M 表示长度， 存储时不足 M 以空格填充， 检索时删

除尾部空格

VARCHAR(M) 变长字符串，M 是最大长度，M 的范围是 0 到 65,535

TINYBLOB 最大长度为 255(2^8–1)字节 (BLOB 表示二进制大对象，没有字符集属性，以字节为单位)

TINYTEXT 最大长度为 255(2^8–1)字符 (TEXT 有字符集属性，以字符为单位)

BLOB 最大长度为 65,535(2^16–1)字节

TEXT 最大长度为 65,535(2^16–1)字符

MEDIUMBLOB 最大长度为 16,777,215(2^24–1)字节

MEDIUMTEXT 最大长度为 16,777,215(2^24–1)字符

LONGBLOB 最大长度为 4G(2^32–1)字节

LONGTEXT 最大长度为 4G(2^32–1)字符

创建表（SQL 语句不区分大小写）

```
CREATE TABLE major
(
Name            CHAR(255) PRIMARY KEY,
College         CHAR(255),
Comment         TEXT
) ENGINE=INNODB;
```

primary key 约束是指这个字段的所有记录，不能有重复，不能为空

```
DESCRIBE major; //查看 major 表的定义信息

CREATE TABLE student
(
Id              INT PRIMARY KEY AUTO_INCREMENT,
Name        CHAR(255),
Age             TINYINT,
Gender      CHAR(1),
Class           CHAR(16),
Major           CHAR(255) CHARACTOR SET GB2312,
Math            FLOAT(5,1),
C               FLOAT(5,1),
```

```
Chinese    FLOAT(5,1),
Total      FLOAT(6,1),
FOREIGN KEY (major) REFERENCES major (Name)
) ENGINE=INNODB;
```

//INNODB 事务处理引擎，支持 Foreign key，而 Mysql 默认的 ISAM 与 MyISAM 不支持 Foreign key 约束

//如果没有 INNODB， FOREIGN KEY 语句将被忽略

major 表的主键作为 student 表的一个外键,建立起两张表的约束关系

```
DESCRIBE student;
```

//删除表，数据库

```
DROP TABLE 表名称
DROP DATABASE 数据库名称
```

//仅仅除去表内的数据，但并不删除表本身(保留表的定义)

```
TRUNCATE TABLE 表名称
```

//修改数据库或表的定义属性

```
ALTER TABLE student ADD FOREIGN KEY (Major) REFERENCES major (Name);
```

//更改表定义，增加列

```
ALTER TABLE student ADD COLUMN graduate INT;
```

//更改表定义，删除列

```
ALTER TABLE student DROP COLUMN graduate;
```

//更改表定义，更改列类型

```
ALTER TABLE student MODIFY graduate YEAR;
```

//更改表定义，更改列名称和类型

```
ALTER TABLE student CHANGE credit graduate DATE;
```

//

第二部分：

插入记录

```
INSERT INTO 表名称 VALUES (值 1, 值 2,....)
```

或

```
INSERT INTO table_name (列 1, 列 2,...) VALUES (值 1, 值 2,....)
```

向表 major 插入记录

```
INSERT INTO major VALUES ('地理学 ', '理学院 ', '');
```

也可写成 :

```
INSERT INTO major (Name, College) VALUES ('地理学 ', '理学院 '); //Commet 字段为空

INSERT INTO major VALUES ('光学', '理学院', '');

INSERT INTO major VALUES ('应用数学', '理学院', '');

INSERT INTO major VALUES ('应用物理', '理学院', '');

INSERT INTO major VALUES ('生物学', '理学院', '');

INSERT INTO major VALUES ('电子信息工程', '机电学院', '');

INSERT INTO major VALUES ('计算机', '机电学院', '');

INSERT INTO major VALUES ('电气工程', '机电学院', '');

INSERT INTO major VALUES ('机械', '机电学院', '');

INSERT INTO major VALUES ('建筑学', ' 科学与艺术学院', '');
```

向 student 表插入记录

```
INSERT INTO student VALUES ('', 'Tom', '21', 'M', '1', '地理学',  '79.5', '60', '60.5',
Math + C + Chinese);
```

学号字段是主键，如果为空，数据库会以递增的方式自动填充， Total 字段可以写成 Math + C + Chinese 让数据库计算它的值

也可以写成

```
INSERT INTO student
(Name, Age, Gender, Class, Major, Math, C, Chinese, Total)
VALUES
('Tom', '21', 'M', '1', '地理学',  '79.5', '60', '60.5',  Math + C + Chinese)
```

这里学号没有指定

```
INSERT INTO student VALUES ('', 'Bob', '22', 'M', '1', '应用物理', '29.5', '50', '66.5',
Math + C + Chinese);

INSERT INTO student VALUES ('', 'Maline', '18', 'F', '2', '光学', '39.5', '70', '66.5', Math
+ C + Chinese);

INSERT INTO student VALUES ('', 'Paul', '29', 'M', '2', '生物学', '69.5', '80', '63.5',
Math + C + Chinese);

INSERT INTO student VALUES ('', 'Gates', '26', 'M', '1', '计算机', '59.5', '85', '35', Math
+ C + Chinese);

INSERT INTO student VALUES ('', 'Bree', '24', 'F', '2', '建筑学', '89.5', '95', '75', Math +
C + Chinese);

INSERT INTO student VALUES ('', 'Susan', '19', 'F', '3', '电气工程', '29.5', '67', '88',
Math + C + Chinese);

INSERT INTO student VALUES ('', 'Cluse', '20', 'M', '1', '应用数学', '49.5', '77', '82',
Math + C + Chinese);

INSERT INTO student VALUES ('', 'Matt', '30', 'M', '3', '电子信息工程', '28.5', '63', '38',
Math + C + Chinese);

INSERT INTO student VALUES ('', 'Blues', '28', 'M', '1', '机械', '28.5', '63', '38', Math +
C + Chinese);

INSERT INTO student VALUES ('', 'Albert', '23', 'F', '1', '生物学', '99.5', '150', '69',
Math + C + Chinese);
```

此时如果插入这一句，由于“理论物理”在 major 表中不存在，就没法在 student 表中插入专业为“理论物理”的记录

```
INSERT INTO student VALUES ('', 'Duke', '21', 'M', '1', '理论物理', '79.5', '90', '60.5',
Math + C + Chinese);
```

//查询记录
SELECT 字段名 FROM 表名 [WHERE 条件表达式]
//将结果按升序或降序显示
SELECT * FROM student ORDER BY Math ASC/DESC;
//使用 AND 和 OR
SELECT * FROM student WHERE Major = '应用物理' AND Class = '1';
SELECT * FROM student WHERE Major = '应用物理' OR Age > '23';
练习:
查询年龄大于 25 岁的记录且数学成绩大于 60 分的记录
查询所有女同学的 Major 和 各科成绩

//更新记录
//修改学生各科成绩
UPDATE student SET Math = '63' WHERE Name = 'Cluse';
UPDATE student SET Total = Math + C + Chinese WHERE Name = 'Cluse';
练习:
更新一个学生的三门课的成绩，和总成绩

//删除一条记录
DELETE FROM student WHERE Id = '1001';
//此时执行删除 major 表中 "电气工程" 专业的操作也会报错，因为这也会违反 Foreign key 约束——与之关联 student 表中有专业为 “电气工程“的记录
DELETE FROM major WHERE Name = '电气工程';
练习:
删除数学成绩小于 60 分的记录

//在 Id 字段上建立索引降序 / 升序，索引可以加快查询(按建立索引的字段查询)
CREATE INDEX StudentIndex ON student (Id DESC/ASC);
//查看表中的索引
SHOW INDEX FROM student;
//删除索引

ALTER TABLE student DROP INDEX StudentIndex;

练习:

在总分(Total)字段建立降序索引，并删除索引

//

第三部分：

//导入导出

//导出数据库 test 为 sql 文件

mysqldump --user=root --password=admin test > test.sql

//只导出表的内容

mysqldump -u root -p test --no-create-db=TRUE --no-create-info=TRUE --add-drop-table=FALSE student>student.sql

//导入

source /home/linux/student.sql;

练习:

导出 student 表，然后删除 student 表内容但不删除表定义，然后导入 student 表