创建表

```
USE studentinfo;#严格来说这一句与后面语句的联系不是特别紧密，是可以进行分开的；
CREATE TABLE IF NOT EXISTS student  (
    StudentID CHAR(10) COMMENT'xuehao',
    StudentName VARCHAR(10) COMMENT'xingming',
    SEX ENUM('nan','nv') DEFAULT'nan',#含有二选一的选择型问题的时候；
    Birthday DATE COMMENT'shengri'#最后一行是没有符号的
);#直到整张表都定义完成之后才会有符号
```

查看表

```
SHOW TABLES;
SHOW COLUMNS from student in studentinfo;#这个表中华所有的设置都会出现；
DESC student;#上面的缩写型式
```

修改表

```
USE studentinfo;
ALTER TABLE student
    ADD ClassID CHAR(10) COMMENT'id';#使用ALTER的方法来修改表的时候，一次性只能进行一次更改操作，于是我们在书写的时候要将不同的修改分开来，各个修改之间是独立的；
ALTER TABLE student
    RENAME COLUMN SEX to Sex;#修改列的名字
ALTER TABLE student
    MODIFY ClassID VARCHAR(10) AFTER Sex; #不仅能够将ClassID的数据类型进行更改，同时在输出的表上ClassID的位置也发生了更改：（change的使用方法也是类似的）
ALTER TABLE student
    DROP ClassID;#删除一整列
ALTER TABLE student
    ALTER COLUMN Sex DROP DEFAULT;
    ALTER COLUMN Sex SET DEFAULT'nan';#删除和甚至出啊是指其实就是使用类似的方法来实现的；
DESC student;
```

删除表

```
DROP TABLE IF EXISTS tb_name;
```

对于表中文件进行约束

- 列级约束：适用于单列主键。也就是说该列数据各行各不相同。此时我们只要设置这个列为键就可以实现约束；
- 表级约束：适用于多列联合主键。一个表的键是由多个列的组合。

不同的约束类型：

- 主键约束：主键的值唯一&所有制非空
- 唯一键约束：值必须唯一&最有仅有一个为NULL&可以有多个键声明为唯一键
- 外键约束：

- 外键：设置外键的是从表，外键关联的是主表的主键；
  - 不妨设主表为年段中的学生信息，从表为班级中的学生信息

    从表中涉及到的学生都在主表中出现；于是说主表删除学生信息的时候要看这个学生信息有没有被从表引用；从表添加学生信息的时候要看这个学生有没有事先在主表中定义。
- 外键约束：根据上面原则进行处理的方法。

不同约束类型的约束方法：

在创建表的时候：

在创建表的最后添加语句：

```
PRIMARY KEY(StudentID)#如果你是有个列的组合


就在后面加上其他的表；
UNIQUE KEY(StudentID)
FOREIGN KEY (ClassID从表外键) REFERENCES class(主表) (ClassID（主表主键名）)
#有时候会对于这个关系本身进行命名【CONSTRAINT FK_student 给外键约束命名为 FK_student】
```

当题目中特别说明要列级的时候：
直接在对应的列定义的后买你添加：

```
PRIMARY KEY
UNIQUE KEY
REFERENCES class主表主键名(主表)
```

使用ALTER对于键进行修改的时候：

```
ALTER TABLE Student
    DROP PRIMARY KEY,
    ADD PRIMARY KEY(StudentID);

ALTER TABLE Student
    MODIFY Birthday DATE UNIQUE KEY;
#由于在同一个表中是允许有多个唯一键，于是说我们需要针对具体某一类型的值添加上唯一键
```

向表中插入记录

```
INSERT INTO Class(CLAssNum,ClassName)
    VALUES('123','emmmm2'),
          ('135','???');
```

其中Class后面的部分是column的名字。

- 当我们填入的是部分的记录的时候，column中要填入的我们这部分信息的列名；（这时候在对应的记录中其他的信息都是NULL）
- 当我们插入记录的格式都是一一对应的时候，可以不用填写Class后面的内容

向表中修改记录

```
UPDATE Class
   SET ClassName='kongbai',ClassNum='114';
   WHERE ClassNum='1';
UPDATE Class
   SET ClassNum=ClassNum+10;
```

SET后面就是要找出你想要进行修改的记录的位置和数值，使用WHERE来确定你要修改的位置（WHERE包含多个数据的时候，多个数据都会产生变化）

当数据是int类型的时候可以对于所有数据进行处理实现所有数据的变化

删除部分数据

```
DETELE FROM Class WHERE Conditions;
```

当WHERE后面没有相关的数据限制的时候，就是对于表中所有的数据进行处理，也就是能够删除表中所有的数据

单表查询数据

```
SELECT [ALL,DISTINCT] column1,column2
    FROM tb_name
    WHERE (CONDITION)
    ORDER BY column1 [ASC|DESC]
    LIMIT num2 OFFSET num1;
```

- 查询所有的信息，中间使用column的部分使用*来代替
- 默认是ALL，使用DISTINCT保证不会出现重复的目录
- 其中的column部分可以是填入计算式，比如ClassNum-10，这样可以实现输出结果是数值。
- 其中关于WHERE的条件有不同的情况，对应不同的结果

表 5-1　常用的查询条件

查询条件	操作符或关键字
关系运算符	<，<=，=，>，>=，<>，!=，!<，!>，<=>
指定范围	BETWEEN AND，NOT BETWEEN AND
集合	IN，NOT IN
匹配字符	LIKE，NOT LIKE
是否空值	IS NULL，IS NOT NULL
逻辑运算符	NOT 或!，AND 或&&，OR 或||，XOR

  - 关系运算符：就是相关数值的运算
  - 主要是数值的判断和日期的判断
  - 集合：IN ('place1','place2',.....)这样的形式 ，看看是不是在里面

    - LIKE：

      ```
      SELECT * FROM Class WHERE ClassName LIKE 'kongbai';
      ```

      主要就是用来判断字符串的；
    - IS NULL ： 判断是不是空的
- order by: 对于表中的数据进行排序操作
    - ASC：升序
    - DESC：降序
- LIMIT:限制查询的范围
    - 限制查询的范围为num1~num2中间；
    - 其中所有的数值是从0开始的

聚合数据查询

聚合函数

- COUNT(*) 返回数据表中的记录数（包含NULL也计算在内）
- COUNT（[DISTINCT|ALL]column）:记录数
- 其余的MAX、MIN、SUM、AVG就很熟悉了

分组聚合查询

```
SELECT COURSEID,COUNT(*),MAX(Score), MIN(Score)
    FROM tb_name GROUP BY StudentID
    HAVING AVG(Score)>=80
    WITH ROLLUP;
```

- GROUP BY:
    - 表示就是按照某些数据专门继续分组，就像图中的数据统计某个学生的选课门数、最高分、最低分
    - 使用GRUOUP BY能做到的就是将对应COLUMN上面的数据把其中相同的分到一个组中。
    - 在这个组的基础之上我们来继续求解相关的平均值、最大值等东西
- HAVING
    - WHERE只能处理没有分组的数据，HAVING只能处理分组之后的数据
    - 如果前面没有生命GROUP的情况下直接使用HAVING，程序会视所有的数据为一个大组
- WITH ROLLUP
    - 这个主要就是能在表的最后面生成一个总结的一栏，给出分组之后所有数据的平均值

多表连接查询

原因：查询的数据涉及多个表

- 交叉连接：给出两个表所有数据的组合

- 内连接：结果仅仅输出满足结果的行
- 外连接：在左外连接、右外连接：在一个表的基础上连接另外一个表，可能会出现部分数据没有连接上另一个表

一般主要考察时内连接

- 等值连接

```
SELECT column1,column2
    FROM Table1 INNER JOIN Table2 #所有要进行讨论的表都先使用内连接的方式连接在一起
    ON Table1.columna=Table2.columnb #确定什么值相同的连接在一起
    WHERE [Conditions]#进行一个筛选
```

- 不等值连接

  ON的条件不同。ON的条件改成不相同就是等值连接。

- 自然连接

```
SELECT colum1,column2
    FROM tb_name1 NATURAL JOIN tb_name2;
```

这个的前提是两个表中有相同的列名以及对应的数值。自然连接帮你自动将这些值匹配起来

子查询

- 在得出子查询的结果之后的值不会立刻进行输出，而是会传输到父查询中，作为夫查询的值
- 我们需要知道的是，子查询的定义相当之宽广，于是说我们在使用的时候会在很多不同的地方见到子查询的使用：
- 子查询的结果返回一个值的时候，子查询可以作为表达式中的一个项目

```
SELECT (子查询),column1
    FROM __
    WHERE __
```

- 子查询返回的是一组的数据的时候，可以将其用于带IN关键字的查询(也就是看看有没有是在子查询要求下的数据)

```
SELECT * FROM student
    WHERE StudentID in (SELECT StudentID FROM selectcourse WHERE Score<60);
```

- 对于子查询中的数据，IN只能判断被查询表中有没有子查询相关的数据，使用表达式+ANY可以指出所有满足大小关系等关系的（也就是能将所有满足条件的都进行输出）见例5-42
- 同时使用子查询同样能够实现相关插入、删除、修改相关的代码中

只要我们子查询查询出来的数据类型是符合条件的，就能够应用在各个地方