拓展：

`IP`：

计算机的唯一标识，类似于人的身份证。一个`ip`代表一台计算机。`localhost` 的`IP`地址为127.0.0.1，表示当前计算机本身。

端口：

代表的是计算机中某一个程序，端口本质是一个数据接口。

`mysql`是一个程序，默认占用一个端口 3306 。连接到数据库后，执行`show global variables like 'port';`即可查看端口号。

```
# 时间类型举例
create table a(id1 date, id2 time,id3 datetime, id4 timestamp,id5 year);

# desc查看表格格式
+-----------+----------+---------------------+---------------------+------------
--+
| Field | Type      | Null | Key | Default             | Extra
  |
+-----------+----------+---------------------+---------------------+------------
--+
| id1   | date      | YES  |     | NULL                |
  |
| id2   | time      | YES  |     | NULL                |
  |
| id3   | datetime  | YES  |     | NULL                |
  |
| id4   | timestamp | NO   |     | CURRENT_TIMESTAMP | on update
CURRENT_TIMESTAMP |
| id5   | year(4)   | YES  |     | NULL                |
  |
+-----------+----------+---------------------+---------------------+------------
--+

# now() 内置的日期和时间函数，用于获取当前的日期和时间。
# now() 它返回一个表示当前日期和时间的日期时间值。
insert into a values(now(),now(),now(),now(),now());
# 查询表
select * from a;
+------------+----------+---------------------+---------------------+------+
| id1        | id2      | id3                 | id4                 | id5  |
+------------+----------+---------------------+---------------------+------+
| 2023-04-29 | 20:33:15 | 2023-04-29 20:33:15 | 2023-04-29 20:33:15 | 2023 |
+------------+----------+---------------------+---------------------+------+
```

MySQL不同条件查询

判断是否为空

```
is null  #为空
is not null    # 不为空
#语法：
select 字段名 from 表格名 where 字段名 is null
select 字段名 from 表格名 where 字段名 is not null
# mysql中的 null 和 python中的 None 都是空的意思

create table t1(id int, name varchar(10));    # 新建一张表格拥有id和name字段
insert into t1(id) values(1),(2);      # 往t1表中插入id字段的两个数据，而name字段不插入
insert into t1(id,name) values(3,'tongyao');
insert into t1 values(4,'tangsan');
# 当要往表格中插入所有字段数据的时候，表格后面可以不声明要新增数据的字段

# select 查询   * 所有  from 从...中    where 当...
# 查询t1表格中姓名字段为空的数据
select * from t1 where name is null;
# 查询t1表格中姓名字段不为空的数据
select * from t1 where name is not null;
```

逻辑判断

```
and  or  not
# and  并且、与（左右两边条件都需要满足）
# or   或者、或（左右两边条件只需要满足一个）
# not  取反、非
# 结合 where 进行使用，where类似于Python中if

# 语法：
select * from 表格名 where 条件表达式 and 条件表达式;
select * from 表格名 where 条件表达式 or 条件表达式;
select * from 表格名 where not 条件表达式;

# 查询学员信息表：
select * from students;

# 练习1：查询年龄在16岁~21岁的学员信息
select * from students where age>=16 and age<=21;

# 练习2：查询年龄在16岁以下 或者 年龄在21岁以上的学员信息
select * from students where age<16 or age>21;

# 练习3：查询年龄不是 16岁的所有学员
select * from students where not age=16;
```

排序

```
order by 排序列名      # 根据...排序，默认正序
order by 排序列名 desc        # 根据...排序，倒叙排列
# 语法：
select 字段名 from 表格名 order by 字段名;  # 正序
select 字段名 from 表格名 order by 字段名 desc;   # 倒叙

#正序  练习1：在查询学员信息表过程中，按照年龄从小到大的顺序排列
select * from students order by age;
#倒序  练习2：在查询学员信息表过程中，按照年龄从大到小的顺序排列
select * from students order by age desc;
```

限制

```
limit  # 关键字，限制从数据库检索数据行数
limit x    # 展示 x 行数据（从第一条数据开始，包括第一条）
limit x,y    # 展示 x 行（从0开始数）开始的 y 条数据
# 语法：
select 字段名 from 表格名 limit x;   # 查询某个表格中数据，只查询x行
select 字段名 from 表格名 limit x,y;  # 查询某个表格中数据，查询从x行（从0开始数）开始的y条数据

# 练习1：在查询学员信息表过程中，只查询5条数据
select * from students  limit 5;
# 练习2：在查询学员信息表过程中，查询从学员Tonny开始的3条数据
select * from students limit 7,3;
```

去重

```
distinct   #去重关键字，写在select之后，被查询内容之前
# 语法：
select distinct 字段名 from 表格名;

# 练习1：在查询学员信息表过程中，查询出学员分别是什么年龄（进行去重）
select distinct age from students;
# 练习2：查询成绩表中收录了哪些科目的成绩？
select distinct subject_number from grades;
```

模糊查询

```
like   # 关键词
%      # 匹配任意多个
_      # 匹配1个字符

# 语法：
select 字段名 from 表格名 where 字段名 like 模糊查询条件;

# 练习1：在查询学员信息表过程中，只查询名字以J开头的学员信息
select * from students;
select * from students where name like 'J%';

# 练习2：在查询学员信息表过程中，只查询名字以St开头，并后面跟4个字母的学员信息
select * from students where name like 'St____';
```

```
# 在like 后面，模糊查询的时候，用到% 其实是告诉mysql 这个地方 % 可以匹配任意多个数据。 一个下划线代表匹配一个数据。
```

范围查询

```
in      #在...（范围）里面
between ... and ...     #在...与...之间

# 语法：
select 字段名 from 表格名  where  字段 in (条件);
select 字段名 from 表格名  where  between 条件1 and 条件2

#练习1：在查询学员信息表过程中，只查询年龄是 16岁的，18岁，20岁，22岁，24岁的学员信息。
select * from students;
select * from students where age in (16,18,20,22,24);
#练习2：查询年龄在16岁~21岁的学员信息
select * from students where age between 16 and 22;
```

聚合函数

```
count(列名)     #统计所有行数
sum(列名)           #求和
avg(列名)       #平均值
max(列名)           #最大值
min(列名)           #最小值

# 语法:
select 聚合函数(字段) from 表格;

# 练习1：统计学员人数？
select count(*) from students;
# 练习2：统计学员年龄总和？
select sum(age) from students;
# 练习3：求学员平均多少岁？
select avg(age) from students;
# 练习5：求最大岁数的学员？
select max(age) from students;
# 练习6：求最小岁数的学员？
select min(age) from students;
```

分组

```
group by 字段名      # 根据...字段进行分组
having       # 用于在进行完分组后对查询数据进行再次筛选，类似与where

#语法:
select 字段,行统计 from 表格名 group by 字段名;
select 字段,行统计 from 表格名 group by 字段名 having 筛选条件;


# 练习1：请对成绩表格数据，按照成绩科目进行分组
```

```sql
select * from grades;
select subject_number,count(*) from grades group by subject_number;

# 练习2：请对成绩表格数据，按照成绩科目和分数进行分组，并且在分组后筛选出成绩>80的所有组
# 按照成绩科目和分数进行分组
select subject_number,grade,count(*) from grades group by subject_number,grade;
# 筛选出成绩>80的所有组
select subject_number,grade,count(*) from grades group by subject_number,grade
having grade>80;

# 分组查询某些情况下看起来像去重，但是其实他没有去重功能，只是数据折叠了，没有给我们显示而已
```

子查询(嵌套查询)

```sql
# 子查询语法：
select 字段名 from (select 字段名 from 表格 where 条件) as 临时表格名 where 条件;

# 练习思考：  查询年龄在16岁~21岁的学员信息,并按照正序进行显示。
# 1、查询年龄在16岁~21岁的学员信息
select * from students where age between 16 and 22;
# 2、按照正序进行显示
select * from (select * from students where age between 16 and 22) as x order by
age;

# 其他案例：
# 需求1：查询出，学员信息表格中，姓名是以 St 开头的学员信息 --》 需要用到like 模糊查询语句
select * from students where name like 'St%';
# 需求2，在查询出来的 以 St开头姓名的学员信息中，进行筛选，筛选出年龄 >= 20 岁的学员
select * from (select * from students where name like 'St%') as x where age>=20;
```

作业

1.首先把上节课笔记最后的sql作业，全部敲到mysql中去，并将课堂笔记中的案例代码自己练习一遍。（不用交作业的）

2.统计出 grades 表中 0001和0002分别有多少人 （需要截图交作业的）
3.统计出 students 中age 大于18岁的人数（需要截图交作业的）