

第5节 SQL语句DQL(二)

主讲老师：华子

人生苦短，我用python

上节课回顾

- 条件查询
 - 比较运算符 $> < =$ 逻辑运算符 `or and not`
 - 范围查询 `between and` 空判断 `null`、`""`
- 模糊查询 `like % _`
- 分页查询 `limit start count`
- 排序查询 `order by asc` 【`dse`】
- 子查询、`distinct`



本节课程内容

01

数据库表常见查询语句-√

02

聚合、外键、组合-本节课

03

多表连接查询

04

子查询

PART TWO


02

聚合、外键、分组

2-1. 聚合查询

聚合函数对一组值执行计算并**返回单一的值**。聚合的目的：为了快速得到统计数据。

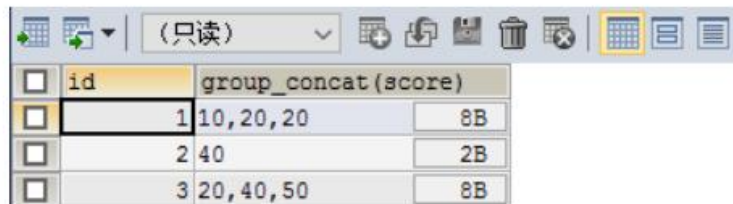
聚合函数	说明
count(*)	表示计算总行数，括号中写*与列名，结果相同
max(列)	表示求此列的最大值
min(列)	表示求此列的最小值
sum(列)	表示此列的和
avg(列)	表示求此列的平均值
group_concat(列)	按组进行来接数据



(只读)

id	score
1	10
1	20
1	20
2	40
3	20
3	40
3	50

```
SELECT id, GROUP_CONCAT(score) FROM testgroup GROUP BY id
```



(只读)

id	group_concat(score)
1	10, 20, 20
2	40
3	20, 40, 50

2-2. 分组查询

分组查询的语法: `select 列1,列2,.....,聚合 from 表名 group by 列1,列2,.....;`

1. 按照字段分组，表示此字段相同的数据会被放到一个组中
2. 可以对分组后的数据进行统计，做聚合运算

	id	name	age
1	2	zs	30
2	1	lisi	20
3	2	zs	30
4	1	lisi	20

2-3. 课堂练习

```
CREATE TABLE `student5`  
( `id`    int(11)    primary key AUTO_INCREMENT,  
  `name`  varchar(50) NOT NULL COMMENT '姓名',  
  `score` int(11)    NOT NULL COMMENT '成绩',  
  `classid` int(11)  NOT NULL COMMENT '班级',);
```

查询每个班级最大分数

2-4. 数据准备——外键(foreign key)FK

###为什么要使用外键?

保证数据的参照完整性。通过定义外键约束，关系数据库可以保证无法插入无效的数据

-- 语法:

foreign key(`classno`) references t_class(`cno`)

User		
user_id	name	type_id
1	张三	1
2	李四	2

一对多

Address			
adr_id	city	detail	user_id
1	北京市	中关村3号	1
2	深圳市	华强北5号	1
3	上海市	陆家嘴1号	2

Students	
id	name
1	zs
2	ls

t_stu_course	
stu_id FK PK	course_id FK PK
1	1
1	2
2	2
2	3

多对多

Courses	
id	name
1	python
2	c
3	java

2-5. 数据准备——外键(foreign key)FK

分析场景

设计数据库表，用来存储学生和班级信息

方案一：普通存储--->将学生信息和班级信息存储到一张表

sno	sname	classno	cname
1	jay	100	浙江省第一中学高三1班
2	lucy	100	浙江省第一中学高三1班
3	king	200	浙江省第一中学高三2班
1234			

缺点：数据冗余，比如cname字段的数据重复太多

方案二：外键形式---->将学生信息和班级信息分开两张表存储

学生表（添加单一外键）

sno(pk)	sname	classno(fk)
1	jack	100
2	lucy	100
3	king	200

班级表

cno(pk)	cname
100	浙江省第一中学高三1班
200	浙江省第一中学高三2班

图中是什么关系？
哪个是主键表，哪个是外键表？

2-6. 数据准备——外键(foreign key)FK

创建新库: bj_school

-- 创建班级表 (主键表)

```
create table grade(  
    id int not null auto_increment primary key,  
    name char(20)  
);  
insert into grade(name) values  
('python开发'),('Java开发'),('Web前端'),('C开发');
```

-- 创建学生表(外键表)

```
create table student(  
    id int not null auto_increment primary key,  
    name char(20),grade_id int,  
    constraint fk_student_grade foreign key(grade_id) references  
grade(id)  
);  
insert into student(name, grade_id) values  
("刘德华", 1),("张惠妹", 2),("张学友", 3),  
("刀郎", 4),("云朵", 4);
```



2-7. 分组查询练习

1. 根据班级id进行分组，汇总各班级人数。
2. 找出班级人数大于1的班级名称。

分组后的数据筛选

需求：展示人数多于1的组信息

#报错

```
select grade_id, count(id) from student group by grade_id where count(id) > 1;
```

使用having

-- 说明：如果对于聚合函数结果进行二次筛选时必须使用having

-- 语法：select 列1,列2,.....,聚合 from 表名 group by 列1,列2,..... having 列1,.....聚合.....;

-- 示例：select grade_id, count(id) from students group by grade_id having count(id) > 1;

having与where区别

1. where是对from后面指定的表进行数据筛选，对于原始数据的筛选
2. having是对group by的结果进行筛选



2-8. 练习

执行: students.sql

- 1. 查询Score表中的最高分的成绩。
- 2. 查询Score表中除了每门课程最高分的学生学号和课程号。
- 3. 查询每门课的平均成绩。
- 4. (最难) 查询Score表中至少有5名学生选修的并以3开头的课程的平均分数。
- 5. 查询分数大于70, 小于90的Sno列。
- 6. 查询所有学生的Sname、Cno和Degree列。 (设计多表查询)

本堂课程已结束

如有疑问, 请咨询学管老师



www.dapengjiaoyu.com