

day15-MySQL进阶

学习目标

1. 能够使用SQL语句进行排序
2. 能够使用聚合函数
3. 能够使用SQL语句进行分组查询
4. 能够完成数据的备份和恢复
5. 能够说出多表之间的关系及其建表原则
6. 能够理解三大范式

一，可视化软件介绍

- 连接数据库
- 对数据库的操作 关掉有道词典，解决闪退问题
- 对表的操作
- 对数据的操作

```
# 环境准备
CREATE TABLE student(
    sid INT PRIMARY KEY auto_increment,
    sname varchar(40),
    sex varchar(10),
    chinese DOUBLE,
    math DOUBLE,
    english DOUBLE
);

INSERT INTO student VALUES(null,'zs','男',58,100,28);
INSERT INTO student VALUES(null,'ls','男',58,100,88);
INSERT INTO student VALUES(null,'ww','女',98,100,100);
INSERT INTO student VALUES(null,'zl','男',58,100,88);
INSERT INTO student VALUES(null,'wb','男',100,100,68);
INSERT INTO student VALUES(null,'小丽','男',58,100,88);
INSERT INTO student VALUES(null,'翠花','男',18,100,88);
INSERT INTO student VALUES(null,'小红','女',58,80,98);
INSERT INTO student VALUES(null,'小李','男',null,100,58);
```

二,查询记录【重点】

1.排序查询

1.1单列排序

- 只按某一个字段进行排序,单列排序,以..排序order by,升序ascending 降序descending

```
SELECT 字段名 FROM 表名 WHERE 添加 ORDER BY 字段名 [ASC|DESC]; //ASC: 升序, 默认值; DESC: 降序
select * from student order by chinese desc;
```

1.2组合排序

- 同时对多个字段进行排序, 如果第1个字段相等, 则按第2个字段排序, 依次类推

```
SELECT 字段名 FROM 表名 WHERE 字段=值 ORDER BY 字段名1 [ASC|DESC], 字段名2 [ASC|DESC];
select sname from student order by chinese desc,english asc;
```

2.聚合函数

2.1概述 放在select from之间

之前我们做的查询都是横向查询, 它们都是根据条件一行一行的进行判断, 而使用聚合函数查询是纵向查询, 它是对一列的值进行计算, 然后返回一个结果值。聚合函数会忽略空值NULL

聚合函数	作用
max(列名)	求这一列的最大值
min(列名)	求这一列的最小值
avg(列名)	求这一列的平均值
count(列名)	统计这一列有多少条记录
sum(列名)	对这一列求总和

2.2语法

- 语法

```
SELECT 聚合函数(列名) FROM 表名;
select max(chinese) from student ;
select sum(ifnull(chinese,0)) from student;
```

- 聚合函数会忽略空值NULL

我们发现对于NULL的记录不会统计, 建议如果统计个数则不要使用有可能为null的列, 但如果需要把NULL也统计进去呢? 我们可以通过 IFNULL(列名, 默认值) 函数来解决这个问题. 如果列不为空, 返回这列的值。如果为NULL, 则返回默认值。 eg: select sum(ifnull(chinese,0)) from student;

```
-- 2. 聚合查询
-- 查询出语文成绩的最高分
SELECT MAX(chinese) AS '语文最高分' FROM student
-- 查询出英语学科的平均分
SELECT AVG(english) FROM student
-- 统计学生表有多少记录
SELECT COUNT(sid) FROM student
SELECT COUNT(*) FROM student
-- 求出语文的总成绩
SELECT SUM(chinese) FROM student
```

3.分组

3.1概述

分组查询是指使用 GROUP BY语句对查询信息进行分组

GROUP BY怎么分组的？将分组字段结果中相同内容作为一组，如按性别将学生分成两组

GROUP BY将分组字段结果中相同内容作为一组，并且返回每组的第一条数据，所以单独分组没什么用处。分组的目的是为了统计，一般分组会跟聚合函数一起使用

3.2语法

```
SELECT 字段1,字段2... FROM 表名 [where 条件] GROUP BY 列 [HAVING 条件];
-- 直接根据sex分组(单独分组没有实际意义的,把每一组的第一条数据返回了;
-- 分组通常都是为了统计,通常和聚合函数一起用)
select * from student group by sex;
```

3.3 分组后筛选 having

- where和having区别【面试题】

子名	作用
where 子句	1) 对查询结果进行分组前，将不符合 where 条件的行去掉，即在 分组之前 过滤数据，即先过滤再分组。 2) where 后面 不可以 使用聚合函数。
having 子句	1) having 子句的作用是筛选满足条件的组，即在 分组之后 过滤数据，即先 分组再过滤 。 2) having 后面 可以 使用聚合函数。

```
-- 统计每一个性别下的学生人数
select sex,count(*)from student group by sex;
select sname,count(sex) from student group by sex having count(sex)>4;
```

- 面试题

id	product	price
1	纸巾	16
2	纸巾	16
3	红牛	5
4	洗衣粉	60
5	苹果	8
6	洗衣粉	60

Orders表数据如下所示，执行如下SQL语句，运行结果是？

```
select product,sum(price) from orders group by product where sum(price) > 30; //报错
```

4.分页查询 limit

4.1概述

LIMIT是限制的意思，所以LIMIT的作用就是限制查询记录的条数. 经常用来做分页查询

4.2 语法

```
select ... from .... limit a ,b.
select * from student limit 0,3;
select * from student limit 3,3;
select * from student limit 6,3;
select * from student limit 9,3;
```

LIMIT a,b;

a: 起始行数，从 0 开始计数，如果省略，默认就是 0

b: 返回的行数

- b: 一页显示的数量(我们自己定义的)
- a: 从哪里开始查询(从0开始计数)

5.查询的语法 先后顺序

```
select .... from .... where... group by.... having....order by.... limit
```

三,多表间的关系

1. 为什么要拆表?

准备工作:

创建一张分类表(类别id,类别名称.备注:类别id为主键并且自动增长)

```
create table student(  
id int primary key auto_increment,  
name varchar(6)  
);
```

创建一张商品表(商品id,商品名称,商品价格,商品数量,类别.备注:商品id为主键并且自动增长)

```
create table product(  
id int primary key auto_increment,  
name varchar(40),  
price double,  
count int,  
style int  
);  
  
insert into product values(null,'iPhone8s',5500,100,1);  
insert into product values(null,'iPhone7',5000,100,1);  
insert into product values(null,'iPhone6s',4500,1000,1);  
insert into product values(null,'iPhone6',3800,200,1);  
insert into product values(null,'iPhone5s',2000,10,1);  
insert into product values(null,'iPhone4s',18000,1,1);  
insert into product values(null,'方便面',4.5,1000,2);  
insert into product values(null,'咖啡',10,100,2);  
insert into product values(null,'矿泉水',2.5,100,2);
```

有些情况下,使用一张表表示数据 数据不好维护,存在数据冗余现象

2.引用完整性【重点】

表和表之间存在一种关系,但是这个关系需要谁来维护和约束?

2.1外键约束

保证引用完整性.表和表之间存在一种关系,但是这个关系是通过外键来维护和约束的

当设置外键后,超出外键范围的数据将无法添加

2.2外键的使用

- 添加外键语法 foreign key references 参考

```
alter table 表 add [CONSTRAINT] [外键名称] foreign key(字段) references 表(字段);  
alter table product add constraint newkey foreign key (style) references student (id);  
  
alter table 表 add foreign key(字段) references 表(字段);  
alter table product add foreign key (style) references student (id);
```

- 删除外键语法

```
ALTER TABLE 表 drop foreign key 外键名称;
删除外键:alter table your_table_name drop foreign key your_foreign_key_id;
这个foreign_key_id 在上边增加外键的时候已经说了,如果你不知你的foreign_key_id也没事,可以用show create
table your_table_name找出来
```

注意:

外键列的类型一定要和参照列(主键)的类型一致

有主键才能有外键, 参照的列必须为主键

- 外键练习

2.3外键的级联

删除一方的时候另外的一方的数据没有任何用途的时候才可以使用 eg: 公司和员工

- 要把类别的id值1, 改成5, 能不能直接修改呢?

```
UPDATE category SET cid=5 WHERE cid=2;
```

不能直接修改:Cannot delete or update a parent row: a foreign key constraint fails 如果副表(商品表)中有引用的数据,不能直接修改主表(类别表)主键

- 要删除类别为1的类别,能不能直接删除呢?

```
DELETE FROM department WHERE id = 1;
```

不能直接删除:Cannot delete or update a parent row: a foreign key constraint fails 如果副表(商品表)中有引用的数据,不能直接删除主表(类别表)数据

- 什么是级联操作

在修改和删除主表的主键时,同时更新或删除副表的外键值,称为级联操作 `on update cascade` -- 级联更新,主键发生更新时,外键也会更新

`on delete cascade` -- 级联删除,主键发生删除时,外键也会删除

具体操作:

- 删除product表
- 重新创建product表,添加级联更新和级联删除

```
alter table product add foreign key (style) references student(id) on update cascade on delete
cascade;
```

3.多表的关系

3.1 一对多【重点】

- 在多方创建一个字段作为外键，指向一方主键



一对多表的建立：在**多的一方**创建一个字段作为外键指向一的一方的**主键**。 Eg:班主任和学生, 学生和分数, 分类和商品

category: 1		product m				
cid	← cname	pid	name	price	num	cid1
1	手机数码					

```
alter table product add foreign key(cid1) references category(cid)
```

3.2 多对多【重点】

- 新建一张中间表，**至少**包含两个字段，都作为外键，分别指向各自的主键



多对多: 创建一张第三方表, 表里面**至少**包含两个字段作为外键, 分别指向各自的主键。 学生和课程, 订单和商品

student m		course n	
sid	sname	cid	cname
1	张三	1	高数
2	李四	2	C语言
3	王五	3	线性代数
		4	毛概

s_c table	
sno	cno
1	1
1	2
2	3
2	4

```
alter table s_c_table add foreign key(sno) references student(sid);
alter table s_c_table add foreign key(cno) references course(cid);
```

3.3 一对一【了解】

- 先当做一对多，在外键字段添加唯一约束。

3.4 旅游项目的数据库设计

四,数据的备份和还原 【会用】

1.备份的应用场景

MySQL数据库使用命令行备份 | MySQL数据库备份命令

例如：

数据库地址：127.0.0.1

数据库用户名：root

数据库密码：pass

数据库名称：myweb

备份数据库到D盘跟目录

```
mysqldump -h127.0.0.1 -uroot -ppass myweb > d:/backupfile.sql
```

如何使用备份文件恢复我们的数据。

mysql控制台：

先 use 数据库名，然后再执行source命令

```
source c:\datacenter.sql
```

2.使用navicat备份和还原

五,数据库设计三大范式

1.概述

好的数据库设计对数据的存储性能和后期的程序开发，都会产生重要的影响。建立科学的，规范的数据库就需要满足一些规则来优化数据的设计和存储，这些规则就称为范式。

2.三大范式

2.1 1NF

2.1.1概述

数据库表的每一列都是不可分割的原子数据项，不能是集合、数组等非原子数据项。即表中的某个列有多个值时，必须拆分为不同的列。简而言之，第一范式每一列不可再拆分，称为原子性

2.1.2 应用

比如上课时间和下课时间					
姓名	性别	班级名称	教室	考勤时间(开始,结束)	
老王	男	JavaEE	110	18-01-10 8:10, 18-01-10 21:30	
张三	女	UI设计	120	18-01-10 8:11, 18-01-10 21:33	
小王	男	JavaEE	110	18-01-11 8:12, 18-01-11 21:36	
张三	女	UI设计	120	18-01-11 8:12, 18-01-11 21:35	
满足第一范式					
姓名	性别	班级名称	教室	开始考勤时间	结束考勤时间
老王	男	JavaEE	110	2018/1/10 8:10	2018/1/10 21:30
张三	女	UI设计	120	2018/1/10 8:11	2018/1/10 21:33
小王	男	JavaEE	110	2018/1/11 8:12	,18-01-11 21:36
张三	女	UI设计	120	2018/1/11 8:12	2018/1/11 21:35

2.1.3 总结

如果不遵守第一范式，查询出数据还需要进一步处理（查询不方便）。遵守第一范式，需要什么字段的数据就查询什么数据（方便查询）

2.2 2NF

2.2.2概述

在满足第一范式的前提下，表中的每一个字段都完全依赖于主键。所谓完全依赖是指不能存在仅依赖主键一部分的列。简而言之，第二范式就是在第一范式的基础上所有列完全依赖于主键列。当存在一个复合主键包含多个主键列的时候，才会发生不符合第二范式的情况。比如有一个主键有两个列，不能存在这样的属性，它只依赖于其中一个列，这就是不符合第二范式。

简而言之,第二范式需要满足:

1. 一张表只描述一件事情
2. 表中的每一个字段都依赖于主键

2.2.2 应用

姓名	性别	班级名称	教室	开始考勤时间	结束考勤时间
老王	男	JavaEE	110	2018/1/10 8:10	2018/1/10 21:30
张三	女	UI设计	120	2018/1/10 8:11	2018/1/10 21:33
小王	男	JavaEE	110	2018/1/11 8:12	,18-01-11 21:36
张三	女	UI设计	120	2018/1/11 8:12	2018/1/11 21:35

一张表只描述一件事情，分成3张表

学生表				班级表			
姓名	性别			班级名称	教室		
老王	男			JavaEE	110		
张三	女			UI设计	120		
小王	男						
张三	女						
				考勤时间表			
				开始考勤时间	结束考勤时间		
				2018/1/10 8:10	2018/1/10 21:30		
				2018/1/10 8:11	2018/1/10 21:33		
				2018/1/11 8:12	,18-01-11 21:36		
				2018/1/11 8:12	2018/1/11 21:35		

表中的每一个字段都依赖于主键

学生表			班级表		
学号	姓名	性别	班级号	班级名称	教室
1	老王	男	1	JavaEE	110
2	张三	女	2	UI设计	120
3	小王	男			
4	张三	女			
			考勤时间表		
			考勤号	开始考勤时间	结束考勤时间
			1	2018/1/10 8:10	2018/1/10 21:30
			2	2018/1/10 8:11	2018/1/10 21:33
			3	2018/1/11 8:12	,18-01-11 21:36
			4	2018/1/11 8:12	2018/1/11 21:35

2.3.3总结

如果不遵守第二范式，数据冗余，相同数据无法区分。遵守第二范式减少数据冗余，通过主键区分相同数据。

2.3 3NF

2.2.3概述

在满足第二范式的前提下，表中的每一列都直接依赖于主键，而不是通过其它的列来间接依赖于主键。简而言之，第三范式就是所有列不依赖于其它非主键列，也就是在满足2NF的基础上，任何非主列不得传递依赖于主键。所谓传递依赖，指的是如果存在"A → B → C"的决定关系，则C传递依赖于A。因此，满足第三范式的数据库表应该不存在如下依赖关系：主键列 → 非主键列x → 非主键列y

2.2.4应用

不满足第三范式				
学号	姓名	年龄	所在学院	学院地点
20160101	张三	18	计算机	中粮13A
20160202	李四	19	马克思	中粮12A
学号-->所在学院-->学院地点				
满足足第三范式				
学号	姓名	年龄	学院编号	
20160101	张三	18	1	
20160202	李四	19	2	
学院编号	所在学院	学院地点		
1	计算机	中粮13A		
2	马克思	中粮12A		

2.2.5总结

如果不遵守第三范式，可能会有相同数据无法区分，修改数据的时候多张表都需要修改（不方便修改）。遵守第三范式通过id可以区分相同数据，修改数据的时候只需要修改一张表（方便修改）。

3.总结

范式	特点
1NF	原子性：表中每列不可再拆分。
2NF	不产生局部依赖，一张表只描述一件事情。
3NF	不产生传递依赖，表中每一列都直接依赖于主键。而不是通过其它列间接依赖于主键。

六,总结

```
排序查询  order by asc desc
select * from student order by chinese asc,English desc;
聚合函数
max() min() avg() sum() count()
放在select from 之间
sum(ifnull(chinese,0))忽略空值
```

分组

group by having

分页

写在最后

limit 0,3;

查询的先后顺序

select from where... group by.... having....order by limit

外键

foreign key references 参考

外键级联

on update cascade

on delete cascade;

alter table product add foreign key (style) references student(id) on update cascade on delete cascade;