第1章 多表查询

1.多表查询类型

1.1 笛卡尔乘积

```
1  select * from teacher,course;
2  或者:
3  select * from teacher join course;
4
5  拿着 teacher每行数据和course逐行进行组合，显示
6  两层for循环的实现逻辑。Simple-Nextloop （嵌套循环方式）
7  得出的结果，会有部分数据是无意义的。
```

1.2 内连接 join 取交集

```
1  mysql> select * from teacher join course on teacher.tno=course.tno ;
2  +-----+--------+------+--------+-----+
3  | tno | tname  | cno  | cname  | tno |
4  +-----+--------+------+--------+-----+
5  | 101 | oldboy | 1001 | linux  | 101 |
6  | 102 | hesw   | 1002 | python | 102 |
7  | 103 | oldguo | 1003 | mysql  | 103 |
8  +-----+--------+------+--------+-----+
9  3 rows in set (0.00 sec)
10
11 mysql> select * from teacher,course where teacher.tno=course.tno;
12 +-----+--------+------+--------+-----+
13 | tno | tname  | cno  | cname  | tno |
14 +-----+--------+------+--------+-----+
15 | 101 | oldboy | 1001 | linux  | 101 |
```

```
16  | 102 | hesw   | 1002 | python | 102 |
17  | 103 | oldguo | 1003 | mysql  | 103 |
18  +-----+--------+------+--------+-----+
19  3 rows in set (0.00 sec)
```

1.3 外连接 left join , right join

```
 1  mysql> select * from teacher left join course on
    teacher.tno=course.tno;
 2  +-----+--------+------+--------+-----+
 3  | tno | tname  | cno  | cname  | tno  |
 4  +-----+--------+------+--------+-----+
 5  | 101 | oldboy | 1001 | linux  |  101 |
 6  | 102 | hesw   | 1002 | python |  102 |
 7  | 103 | oldguo | 1003 | mysql  |  103 |
 8  | 104 | oldx   | NULL | NULL   | NULL |
 9  | 105 | oldw   | NULL | NULL   | NULL |
10  +-----+--------+------+--------+-----+
11  5 rows in set (0.00 sec)
12
13  mysql> select * from teacher right join course
    on teacher.tno=course.tno;    +------+--------
14  +------+--------+-----+
15  | tno  | tname  | cno  | cname  | tno |
16  +------+--------+------+--------+-----+
17  |  101 | oldboy | 1001 | linux  | 101 |
18  |  102 | hesw   | 1002 | python | 102 |
19  |  103 | oldguo | 1003 | mysql  | 103 |
20  | NULL | NULL   | 1004 | k8s    | 108 |
21  +------+--------+------+--------+-----+
22  4 rows in set (0.00 sec)
```

2.多表连接语法

2.1 a表 和 b表 有直接的关联关系

```
 1  select a.x,b.y  from a  join b on a.z=b.z  where  group by  having
    order by limit;
 2
 3  select a.x,b.y     #查找的内容
 4  from a
 5  join b              #a关联b
 6  on a.z=b.z          #关联条件
 7  where               #其他条件
 8  group by            #分组依据
 9  having              #分组后判断
10  order by            #排序规则
11  limit;              #显示条目
```

2.2 a表 和 b表 没有直接的关联关系

```
 1  假如：a和c 有关，b和c有关
 2
 3  a join c on a.i = c.j join b on c.x=b.y
 4
 5  a join c
 6  on a.i = c.j join b
 7  on c.x=b.y
 8
```

2.3 套路

```
 1  1.根据题意将所有涉及到的表找出来 a b
 2  2.找到a和b直接或者间接的关联条件
 3  3.用join on 语句把所有表连接到一起
 4  4.罗列其他查询条件
```

3.大量练习

3.1 导入数据

```
1  source /root/school.sql
```

关系图：

3.2 每位老师所教课程名称

```
1  select
2  teacher.tname,course.cname
3  from teacher
4  join course
5  on teacher.tno=course.tno;
```

3.3 统计每个学员，学习课程的门数

```
1  select student.sname'学生姓名',COUNT(*)'学习门数'
2  from student
3  join sc
4  on student.sno=sc.sno
5  GROUP BY student.sno;
```

3.4 统计每个学员，学习课程的门数和课程名列表

关系图：

```
1  student ----> sc  ----> course ---> teacher
```

语句：

```
1  select
   CONCAT(student.sname,"_",student.sno),COUNT(*),GROUP_CONCAT(course.cname
   )
2  from student
3  join sc
4  on student.sno=sc.sno
5  join course
6  on sc.cno=course.cno
7  group by student.sno
```

3.5 每位老师教的学生数量和学生名列表

关系图：

```
1  student ----> sc  ----> course ---> teacher
```

语句：

```
1   select
    CONCAT(teacher.tname,"_",teacher.tno),COUNT(*),GROUP_CONCAT(student.snam
    e)
2   from teacher
3   join course
4   on teacher.tno=course.tno
5   join sc
6   on course.cno=sc.cno
7   join student
8   on sc.sno=student.sno
9   group by teacher.tno
```

3.6 每位老师教所教课程的平均分

```
1   select
    CONCAT(teacher.tname,"_",teacher.tno,"_",course.cno),AVG(sc.score)
2   from teacher
3   join course
4   on teacher.tno=course.tno
5   join sc
6   on course.cno=sc.cno
7   group by teacher.tno , course.cno
```

3.7 查找学习了hesw但没学习oldguo课程的学生名

case用法：

```
1   select case when 1=1 then "true" end
2
3   USE mysql;
4   SELECT
5   case
6   WHEN USER='root' THEN HOST END,
7   WHEN USER !='root' THEN 2 END
8   FROM mysql.user;
```

```
 9
10  USE mysql;
11  SELECT
12  CASE
13  WHEN USER='root' THEN HOST
14  WHEN USER !='root' THEN 2 END
15  FROM mysql.user;
```

方法1:

```
 1  select a.sname from
 2  a
 3  left join
 4  b
 5  on a.sname=b.sname
 6  where b.sname is null;
 7
 8  select a.sname from
 9  (select student.sname
10  from teacher
11  join course
12  on teacher.tno=course.tno
13  join sc
14  on course.cno=sc.cno
15  join student
16  on sc.sno=student.sno
17  where teacher.tname = 'hesw') as a
18  left join
19  (select student.sname
20  from teacher
21  join course
22  on teacher.tno=course.tno
23  join sc
24  on course.cno=sc.cno
25  join student
```

```
26  on sc.sno=student.sno
27  where teacher.tname = 'oldguo') as b
28  on a.sname=b.sname
29  where b.sname is null
```

方法2:

```
1   SELECT student.`sname`,GROUP_CONCAT(teacher.`tname`)
2   FROM course
3   JOIN sc
4   ON course.cno=sc.cno
5   JOIN student
6   ON sc.sno=student.sno
7   JOIN teacher
8   ON course.tno=teacher.tno
9   GROUP BY student.sname
10  HAVING GROUP_CONCAT(teacher.`tname`) LIKE '%hesw%' AND
       GROUP_CONCAT(teacher.`tname`) NOT LIKE '%oldguo%';
```

3.8 查询出只选修了一门课程的全部学生的学号和姓名

```
1   SELECT student.sname,student.sno,COUNT(sc.cno)
2   FROM sc
3   JOIN student
4   ON sc.sno=student.sno
5   GROUP BY sc.sno
6   HAVING COUNT(sc.cno)=1;
```

3.9 查询各科成绩最高和最低的分: 以如下形式显示: 课程名称, 最高分, 最低分

```
1  SELECT course.cname'课程名称',MAX(sc.`score`)'最高分',MIN(sc.`score`)'最低
   分'
2  FROM sc
3  JOIN course
4  ON sc.cno=course.cno
5  GROUP BY course.cname;
```

3.10 查询平均成绩大于85的所有学生的学号、姓名和平均成绩

```
1  select sc.sno,student.sname,AVG(sc.score)
2  from sc
3  join student
4  on sc.sno=student.sno
5  group by sc.sno
6  having AVG(sc.score)>85;
```

3.11 统计每门课程:优秀(85分以上),良好(70-85),一般(60-70),不及格(小于60)的学生列表

```
1  select
2  course.cname ,
3  GROUP_CONCAT(case when sc.score>=85 then  student.sname end),
4  GROUP_CONCAT(case when sc.score>=70 and sc.score<85 then  student.sname
   end),
5  GROUP_CONCAT(case when sc.score>=60 and sc.score<70 then  student.sname
   end),
6  GROUP_CONCAT(case when sc.score<60 then  student.sname end)
7  from course
8  join sc
9  on course.cno=sc.cno
10 join student
11 on sc.sno=student.sno
12 group by course.cno
13
14 SELECT course.cname,
```

```
15  GROUP_CONCAT(CASE WHEN sc.score>=85 THEN
    CONCAT(student.sname,":",sc.score) END)'优秀',
16  GROUP_CONCAT(CASE WHEN sc.score>=75 AND sc.`score`<85 THEN
    CONCAT(student.sname,":",sc.score) END)'良好',
17  GROUP_CONCAT(CASE WHEN sc.score>=60 AND sc.`score`<75 THEN
    CONCAT(student.sname,":",sc.score) END)'一般'
18  FROM course
19  JOIN sc
20  ON course.cno=sc.cno
21  JOIN student
22  ON sc.sno=student.sno
23  GROUP BY course.cname;
```

3.12 表别名使用

```
1   select
2   a.cname ,
3   GROUP_CONCAT(case when b.score>=85 then  c.sname end),
4   GROUP_CONCAT(case when b.score>=70 and b.score<85 then  c.sname end),
5   GROUP_CONCAT(case when b.score>=60 and b.score<70 then  c.sname end),
6   GROUP_CONCAT(case when b.score<60 then  c.sname end)
7   from course as a
8   join sc as b
9   on a.cno=b.cno
10  join student as c
11  on b.sno=c.sno
12  group by a.cno
```

3.13 列别名

```sql
1  select
2  a.cname as "课程名称" ,
3  GROUP_CONCAT(case when b.score>=85 then  c.sname end) as "优秀学员",
4  GROUP_CONCAT(case when b.score>=70 and b.score<85 then  c.sname end) as
   "良好学员",
5  GROUP_CONCAT(case when b.score>=60 and b.score<70 then  c.sname end) as
   "一般学员",
6  GROUP_CONCAT(case when b.score<60 then  c.sname end) as "不及格学员"
7  from course as a
8  join sc as b
9  on a.cno=b.cno
10 join student as c
11 on b.sno=c.sno
12 group by a.cno
```

第2章 元数据获取

1.常用show语句

```sql
1  help show;
2  show databases;                   # 查询所有库名
3  show tables;                      # 查询当前库的所有表名
4  show tables from world;           # 查询world库下的所有表名
5  show create database world;       # 查询world建库语句
6  show create table city;           # 当前库下的city表建表语句
7  show create table world.city;     # world库下的建表语句
8  show privileges;                  # 数据库中所有权限
9  show engines;                     # 数据库中支持的存储引擎
10 show grants for root@'localhost'  # 查询某个用户权限
11 show charset;                       # 查询数据库字符集支持
12 show collation;                     # 查询所有校对规则的支持
13 show variables like '%trx%'         # 查询数据库参数
```

```
14   show status like 'com_%'              # 查询数据库的状态
15   show processlist;                     #  查询所有会话信息
16   show engine innodb status             #  查询innodb引擎相关的状态
17   show binary logs                      #  查询二进制日志文件信息
18   show binlog events in 'xxx'           #  查看二进制日志事件
19   show master status ;                  #  当前正在使用的二进制日志信息
20   show slave status\G                   #  查看主从状态相关信息
21   show slave hosts;                     #  查看从库主机信息
```