## 练习课（三）

#### 正则表达式练习

**1、匹配年月日日期 格式2018-12-6**

**^[1-9]\d{0,3}-(1[0-2]|0?[1-9])-(3[01]|[12]\d|0?[1-9])$**

1**.**^[1-9]表示年是以数字1-9开头的,\d{0,3}表示年的位数,^[1-9]\d{0,3}就表示1-9999年之间

2.(1[0-2]|0?[1-9])中|前面的1[0-2]表示从10到12,后面的0?[1-9]表示01-09或者1-9,

(1[0-2]|0?[1-9])表示月,01-12或者1-12

3.(3[01]|[12]\d|0?[1-9])$其中3[01]表示30或31,[12]\d表示从10-29,最后的0?[1-9]表示从

01-09或者是从1-9.整体就表示从01-31或者1-31

**2、长度为8-10位的用户密码 ： 包含数字字母下划线**

**\w{8,10}**

**3、匹配验证码：4位数字字母组成的**

**[\da-zA-Z]{4}或者[0-9a-zA-Z]{4}**

[ ]里面的表示数字,或者a-z或者A-Z,{4}表示4位

**4、从类似**

<a>wahaha</a>  
<b>banana</b>  
<h1>qqxing</h1>

**这样的字符串中，**  
**1）匹配出wahaha，banana，qqxing内容。**

**\w{6}**

**>\w+<**  
**2）匹配出a,b,h1这样的内容**

**<\w+>**

**5、1-2\*((60-30+(-40/5)\*(9-2\*5/3+7/3\*99/4\*2998+10\*568/14))-(-4\*3)/(16-3\*2))**  
**1）从上面算式中匹配出最内层小括号以及小括号内的表达式**

　\([^()]+\)　　\(和\)表示前后位( ),[^()]就表示外面的()里面没有()

**6、从类似9-2\*5/3+7/3\*99/4\*2998+10\*568/14的表达式中匹配出从左到右第一个乘法或除法**

**\d+[\*/]\d+** [*/]前后的\d+表示*或/前面的整数,可能是多位数字,要加+

#### mysql面试题集锦

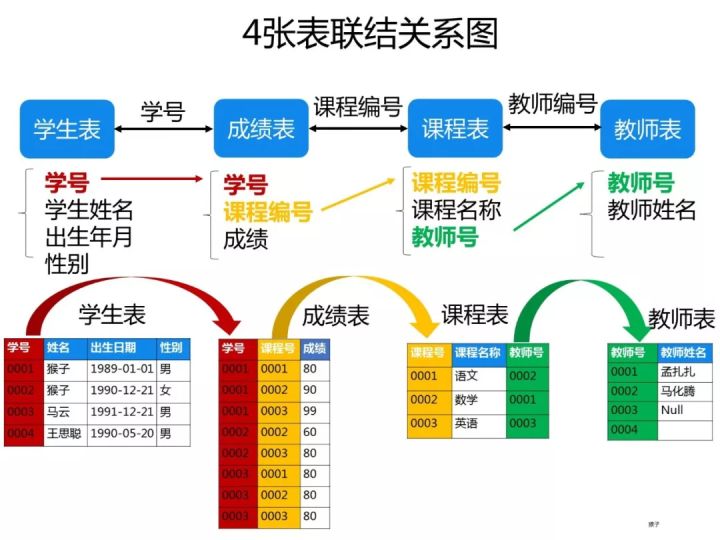
理清SQL语句的执行顺序

创建school数据库

create database school charset=utf8;  
use school;

创建四张表

create table student(  
 s\_id varchar(10),  
 s\_name varchar(20),  
 s\_age date,  
 s\_sex varchar(10)  
);  
  
create table course(  
 c\_id varchar(10),  
 c\_name varchar(20),  
 t\_id varchar(10)  
);  
  
  
create table teacher (  
t\_id varchar(10),  
t\_name varchar(20)  
);  
  
create table score (  
 s\_id varchar(10),  
 c\_id varchar(10),  
 score varchar(10)  
);



往表里插值

insert into student (s\_id, s\_name, s\_age, s\_sex)  
values ('01' , '赵雷' , '1990-01-01' , '男'),  
 ('02' , '钱电' , '1990-12-21' , '男'),  
 ('03' , '孙风' , '1990-05-20' , '男'),  
 ('04' , '李云' , '1990-08-06' , '男'),  
 ('05' , '周梅' , '1991-12-01' , '女'),  
 ('06' , '吴兰' , '1992-03-01' , '女'),  
 ('07' , '郑竹' , '1989-07-01' , '女'),  
 ('08' , '王菊' , '1990-01-20' , '女');  
  
insert into course (c\_id, c\_name, t\_id)  
values ('01' , '语文' , '02'),  
 ('02' , '数学' , '01'),  
 ('03' , '英语' , '03');  
  
insert into teacher (t\_id, t\_name)  
values ('01' , '张三'),  
 ('02' , '李四'),  
 ('03' , '王五');  
  
insert into score (s\_id, c\_id, score)  
values ('01' , '01' , 80),  
 ('01' , '02' , 90),  
 ('01' , '03' , 99),  
 ('02' , '01' , 70),  
 ('02' , '02' , 60),  
 ('02' , '03' , 80),  
 ('03' , '01' , 80),  
 ('03' , '02' , 80),  
 ('03' , '03' , 80),  
 ('04' , '01' , 50),  
 ('04' , '02' , 30),  
 ('04' , '03' , 20),  
 ('05' , '01' , 76),  
 ('05' , '02' , 87),  
 ('06' , '01' , 31),  
 ('06' , '03' , 34),  
 ('07' , '02' , 89),  
 ('07' , '03' , 98);

创建一张总总表

create table total(  
select a.s\_id as s\_id,a.s\_name as s\_name,a.s\_age as s\_age,a.s\_sex as s\_sex,  
b.c\_id as c\_id,b.score as score,c.t\_id as t\_id,d.t\_name as t\_name  
from student a  
left join  
score b on a.s\_id=b.s\_id  
left join  
course c on b.c\_id=c.c\_id  
left join  
teacher d on c.t\_id=d.t\_id  
);  
select \* from total;

# 1、查询"01"课程比"02"课程成绩高的学生的信息及课程分数

select a.s\_id as s\_id,score1,score2 from  
(select s\_id, score as score1 from score where c\_id='01') a  
inner join  
(select s\_id, score as score2 from score where c\_id='02') b  
on a.s\_id=b.s\_id  
where score1>score2;

# 2、查询"01"课程比"02"课程成绩低的学生的信息及课程分数

select a.s\_id as s\_id,score1,score2 from  
(select s\_id, score as score1 from score where c\_id='01') a  
inner join  
(select s\_id, score as score2 from score where c\_id='02') b  
on a.s\_id=b.s\_id  
where score1<score2;

# 3、查询平均成绩大于等于60分的同学的学生编号和学生姓名和平均成绩

select student.s\_id as s\_id,student.s\_name as s\_name,b.avg\_score as avg\_score from student   
right join   
(select s\_id,avg(score) as avg\_score from score  
group by s\_id having avg\_score>60) b  
on student.s\_id=b.s\_id;

# 4、查询所有同学的学生编号、学生姓名、选课总数、所有课程的总成绩

select s\_id, s\_name, count(c\_id) as c\_num, sum(score) as total\_score  
from total  
group by s\_id ;

# 5、查询"李"姓老师的数量

select count(t\_name) from teacher  
where t\_name like '李%';

# 6、查询学过"张三"老师授课的同学的信息

select distinct s\_id,s\_name,s\_age,s\_sex  
from total  
where t\_name='张三';

# 7、查询学过编号为"01"并且也学过编号为"02"的课程的同学的信息

select \* from student  
where s\_id in  
(select s\_id from score where c\_id='01')  
and s\_id in  
(select s\_id from score where c\_id='02');

# 8、查询没有学全所有课程的同学的信息

select s.s\_id,s.s\_name,s.s\_age,s.s\_sex from student as s inner join (select s\_id from total group by s\_id having count(c\_id) <3) as a on s.s\_id = a.s\_id;

# 9、查询至少有一门课与学号为"01"的同学所学相同的同学的信息

思路：先找出‘01’同学学过的c*id，再找出学过任一门的s*id，再根据s\_id在student找学生信息。

select \* from student  
where s\_id in  
(select distinct s\_id from score  
where c\_id in  
(select c\_id from score where s\_id='01'));

# 10、查询没学过"张三"老师讲授的任一门课程的学生姓名

select s\_id,s\_name from student  
where s\_id not in  
(select distinct s\_id from total  
where t\_name='张三');

# 11、查询两门及其以上不及格课程的同学的学号，姓名及其平均成绩

思路：先找不及格超过两门的s\_id，为表a，再根据表a连接学生信息表student和平均分表b。

select a.s\_id,student.s\_name,b.avg\_score from  
(select s\_id from score  
where score<60  
group by s\_id having count(\*)>=2) a  
left join  
student on a.s\_id=student.s\_id  
left join  
(select s\_id,avg(score) as avg\_score  
from score  
group by s\_id) b  
on a.s\_id=b.s\_id;

# 12、检索"01"课程分数小于60，按分数降序排列的学生信息

select a.s\_id,student.s\_name,student.s\_age,student.s\_sex,a.score from  
(select s\_id,score from score  
where c\_id='01' and score<60  
order by score desc) a  
left join student on a.s\_id=student.s\_id;

# 13、查询不同老师所教不同课程平均分从高到低显示

select t\_id,t\_name,c\_id,avg(score) as avg\_score   
from total  
group by t\_id,c\_id  
order by avg\_score desc;

# 14、查询每门课程被选修的学生数

select c\_id,count(s\_id) as '选修人数'   
from score group by c\_id;

# 15、查询出只有两门课程的全部学生的学号和姓名

select student.\* from  
(select s\_id from score  
group by s\_id having count(c\_id)=2) a  
left join student on a.s\_id=student.s\_id;

# 16、查询男生、女生人数

select s\_sex as '性别',count(1) as '人数'  
from student group by s\_sex;

# 17、查询名字中含有"风"字的学生信息

select \* from student  
where s\_name like '%风%';

# 18、查询同名同姓学生名单，并统计同名人数

select distinct s\_name,num as '同名人数' from student,(select count(s\_id) as num from student group by s\_name) a;

# 19、查询1990年出生的学生名单(注：Student表中Sage列的类型是datetime)

select s\_name from student where year(s\_age)='1990';

# 20、查询每门课程的平均成绩，结果按平均成绩降序排列，平均成绩相同时，按课程编号

select c\_id,avg(score) as '平均成绩'  
from score group by c\_id  
order by 平均成绩 desc,c\_id;

# 21、查询平均成绩大于等于85的所有学生的学号、姓名和平均成绩

select a.s\_id,s\_name,avg\_score from   
(select s\_id,avg(score) as avg\_score from score  
group by s\_id having avg(score)>=85) a  
left join student on a.s\_id=student.s\_id;

# 22、查询课程名称为"数学"，且分数低于60的学生姓名和分数

select s\_name,c\_name,score from total  
where c\_name='数学' and score<60;

# 23、查询所有学生的课程及分数情况

select s\_id,  
sum(case when c\_id='01' then score else 0 end) as '语文',  
sum(case when c\_id='02' then score else 0 end) as '数学',  
sum(case when c\_id='03' then score else 0 end) as '英语'  
from total  
group by s\_id;

# 24、查询任何一门课程成绩在70分以上的姓名、课程名称和分数

select s\_name,c\_name,score  
from total where score>70;

# 25、查询不及格的课程

select score.c\_id,course.c\_name,score  
from score left join course  
on score.c\_id=course.c\_id  
where score<60;

# 26、查询课程编号为01且课程成绩在80分以上的学生的学号和姓名

select student.s\_id,s\_name from student  
right join score on student.s\_id=score.s\_id  
where c\_id='01' and score>80;

因为‘01’课程最高分为80，所以查询结果为空。

# 27、求每门课程的学生人数

select c\_id,count(1) as '选课人数'  
from score group by c\_id;

# 28、查询选修"张三"老师所授课程的学生中，成绩最高的学生信息及其成绩

select student.\*,a.score from  
(select s\_id,score  
from total where t\_name='张三'  
order by score desc limit 1) a  
left join student on a.s\_id=student.s\_id;

# 29、查询不同课程成绩相同的学生的学生编号、课程编号、学生成绩

select a.s\_id,a.c\_id,a.score  
from score a,score b  
where a.c\_id=b.c\_id and a.s\_id!=b.s\_id and a.score=b.score;

# 30、查询每门功成绩最好的前两名

(select c\_id,s\_id from score where c\_id='01' order by score limit 2)  
union  
(select c\_id,s\_id from score where c\_id='02' order by score limit 2)  
union  
(select c\_id,s\_id from score where c\_id='03' order by score limit 2);

# 31、统计每门课程的学生选修人数（超过5人的课程才统计）。要求输出课程号和选修人数，查询结果按人数降序排列，若人数相同，按课程号升序排列

select c\_id,count(s\_id) as 选修人数 from score  
group by c\_id having 选修人数>5  
order by 选修人数 desc,c\_id;

# 32、检索至少选修两门课程的学生学号

select s\_id from score group by s\_id having count(c\_id)>=2;

# 33、查询选修了全部课程的学生信息

select \* from student  
where s\_id in  
(select s\_id from score  
group by s\_id having count(c\_id)=(select count(\*) from course));

#### 中期项目设计

### 复习内容

* 网络编程
* 进程线程
* 网络并发模型
* http协议
* 聊天室,文件服务器