**1. SQL入门——user list：对SQL有一个初步认识**

本实例的目的，是让读者对SQL有一个初步的认识。

User\_list是一个简单的表格，每条记录只有两个字段，分别是id和name，数据类型是字符串，大小都是16个字节。它保存id和name的对应关系。

**user\_list:**

id

name

建表：建立一张表，记录用户id和name信息。备注：sql语句以分号结束。

|  |
| --- |
| create table user\_list  (  id char(16),  name char(16)  ); |

插入数据：往表中插入4条记录。

在这里，笔者不打算用大量的篇幅讲SQL语法，而是请读者在看到若干个实例之后，很快地学会SQL。基本的学习路线图是，看实例，理解实例，按照新任务要求改实例。

|  |
| --- |
| insert into user\_list values( 's001', 'wang xue');  insert into user\_list values( 's002', 'li gang');  insert into user\_list values( 's003', 'zhang liao');  insert into user\_list values( 's004', 'cao cao');  insert into user\_list values( 's005', 'cao cao'); |

删除数据：删除学号为s004的记录。

|  |
| --- |
| delete from user\_list where id='s004'; |

修改数据：修改学号为s002的记录的name为li gan。

|  |
| --- |
| update user\_list set name='li gan' where id='s002'; |

查询数据：把所有的数据查出来。

|  |
| --- |
| select \* from user\_list; |

备注：\*表示显示所有字段。

把学号为s003的记录查出来。

|  |
| --- |
| select id,name from user\_list where id='s003'; |

把id大于s001的记录找出来

|  |
| --- |
| select id,name from user\_list where id>'s001'; |

聚集查询实例：查询学生的个数。

|  |
| --- |
| select count(\*) from user\_list; |

把重名的学生查出来

|  |
| --- |
| select name, count(\*) as **counter**  from user\_list  group by name  having counter>=2 |

这个语句把表格中的记录按照name进行分组(group by)，也就是把重名的拢到一块，然后数数(count(\*))，最后，找出数量大于等于2的name。

备注：counter是count(\*)的别名。

**2. SQL入门——student & department：单表查询和两表连接查询**

Student表存放学生信息，主要的字段有学号、姓名、性别、出生年月等。

因为要和院系表建立关系，所以学生表还有一个外键字段（这个字段是院系表的主键），即院系id。

**department:**

did

dname

dintro

**student:**

sid

sname

sex

birthday

did

建表：

|  |
| --- |
| create table student  ( sid char(16),  sname char(32),,  sex char(1),  birthday char(8),  did char(16);  );  create table department  ( did char(16),  dname char(64),  dintro char(256)  ); |

备注：MySQL有日期时间数据类型，在student表格的建立中，我们没有用到，而是用字符串表示日期时间，比如用“20180606”表示2018年06月06日。

插入一些种子数据：

|  |
| --- |
| insert into department values('d001', 'information', 'the economics department founded in 1978');  insert into department values('d002', 'economics', 'the economics department founded in 1980');  insert into student values('s001', 'li gang', 'M', '20000101', 'd001');  insert into student values('s002', 'wang tao', 'M', '20000201', 'd001');  insert into student values('s003', 'yang li', 'F', '20000301', 'd002');  insert into student values('s004', 'zhang li', 'F', '20000401', 'd002'); |

查询2000年4月1日以后出生的学生。

|  |
| --- |
| select \*  from student s  where s.birthday >='20000401'; |

备注：这是一个单表查询，由于出生年月字段是字符串，具有天然的字典序，我们可以用s.birthday >='20000401'表达2000年4月1日以后出生的语义。\*表示显示所有字段。

思考：如何编写SQL，查询'20000301'之前出生的学生信息。

查询信息系的学生：

|  |
| --- |
| select s.sid, s.sname  from student s, department d  where s.did = d.did  and d.dname = 'information'; |

备注：s和d分别是student和department的别名。

因为我们的条件在department表格上，而要显示的信息（sid,sname）在学生表上，所以需要连接两张表，s.did = d.did是连接条件。

思考：如何编写SQL，查询'economics'系的学生信息。

按照院系汇总学生人数：

|  |
| --- |
| select d.did,d.dname, count(\*) as counter  from student s, department d  where s.did = d.did  group by d.did,d.dname; |

备注：学生在学生表，院系在院系表，为了统计各个院系的学生人数，需要连接两张表。连接条件是s.did = d.did。

这是一个分组统计，分组的条件是d.did, d.dname，也就是按照did和dname对各个院系关联的学生进行分组，统计一下计数（count(\*)）。

**3. SQL入门——student & course & student\_course：多表连接查询**

为了展示多表查询，我们增加两个表，分别是课程表和选课表。

**student:**

sid

sname

sex

birthday

did

**department:**

did

dname

dintro

**course:**

cid

cname

cintro

**student\_course:**

sid

cid

grade

课程表保存一系列的课程，包括课程号、课程名、课程介绍等字段。

选课表保存了学生的选课记录，选课表是一个特殊的表，它表达了学生对课程的选修这样的**关系**。一个学生可以选修多门课程，一个课程可以被多个学生选修。它包含学号、课程号、成绩等字段。

注意：student\_course表的sid和cid分别指向student表、和course表，是外键，但是它们两者结合在一起，构成了student\_course表的主键，也就是两者**唯一标识了**学生对课程的选修。

建表：

|  |
| --- |
| student和department的建表语句，已经在上文介绍了。  create table course  ( cid char(16),  cname char(32),  cintro char(128)  );  create table student\_course  (sid char(16),  cid char(16),  grade decimal(16,2)  ); |

插入一些种子数据：

|  |
| --- |
| insert into course values('c001', 'data science', 'data science is a new course');  insert into course values('c002', 'database', 'this is the database course');  insert into student\_course values('s001', 'c001', 88.50);  insert into student\_course values('s002', 'c001', 95.50);  insert into student\_course values('s003', 'c002', 78.50);  insert into student\_course values('s004', 'c002'', 87.00); |

查询wang tao选修的课程及其成绩：从学生表入手

|  |
| --- |
| select c.cname, sc.grade  from student s, course c, student\_course sc  where s.sname='wang tao'  and s.sid = sc.sid and c.cid=sc.cid; |

因为要显示的课程名在课程表，查询条件wang tao又在学生表，要显示的成绩在选课表，所以需要三表连接，s.sid = sc.sid and c.cid=sc.cid是连接条件。

思考：如何编写SQL，查询'yang li'选修的课程及其成绩。

查询数据库课程有哪些同学选修：从课程表入手

|  |
| --- |
| select s.sid, s.sname  from student s, course c, student\_course sc  where c.cname='database'  and s.sid = sc.sid and c.cid=sc.cid; |

因为要显示的学生信息在学生表，课程名在课程表，学生选课信息在选课表，所以需要三表连接，连接条件和上一个语句是一样的。

思考：如何编写SQL，查询'data science'课程有哪些同学选修。

查询选择数据库的同学的平均分：从课程表入手

|  |
| --- |
| select avg(sc.grade)  from cource c, student\_course sc  where c.cname ='database'  and c.cid = sc.cid |

这是一个聚集查询，求平均分。

由于课程信息在课程表里，选课信息在选课表里，所以需要两表连接。

首先把选数据库这门课的选课记录拿出来，然后对其grade字段求average。

思考：如何查询选择数据库的同学的最高分？提示：把avg改成max即可。

思考：如何编写SQL，查询wang tao选修的课程的平均分。提示：student和student\_course表的连接查询。

查询选修人数最多的课程：从课程表入手

|  |
| --- |
| select c.cid, c.cname, count(\*) as counter  from course c, student\_course sc  where c.cid = sc.cid  group by c.cid, c.cname  order by counter desc |

这是一个分组汇总语句，把course表格和student\_course表格连接起来，把每门课的选课记录按照cid和cname拢在一起，然后进行计数（count(\*)），别名是counter。

接着按照counter进行降序排列(desc)，排在最开始的那条记录就是被最多的同学选修的课程。

思考：如何编写SQL，查询选修最多门课程的同学。

提示：照着这个实例，把course和student\_course的连接操作，改成student和student\_course的连接操作。其它部分适当改改，造成每门课的选课记录，按照sid、sname进行聚拢，然后进行计数(count(\*))。

4.Hints：如何学好SQL

　　看实例，抄过来，改一改，就能很快学会了。

熟读唐诗300首，不会作诗也会吟。

熟悉SQL了以后，最后自己也就能够写出来了。

**特别提示：为了编写表间连接查询，请随时参考上文中绘制的ER图，即实体关系图，特别关注每个表格的主键和外键。**

５.扩展阅读

MySQL支持各种数据类型，包括整数、实数、字符串、日期时间等。

SQL语句不仅可以进行简单查询，还可以进行聚集查询（汇总查询）以及连接查询（join）。