第三章 SQL语言

授课教师：周逊 教授
哈尔滨工业大学（深圳）
计算机科学与技术学院

第3章　SQL语言

学习内容

1. SQL语言概述
   - ① SQL语言提出和发展
   - ② SQL语言概揽
   - ③ 本章的目标
2. 简单的SQL-DDL/DML: 创建数据库
3. SQL-DML之基本查询SELECT-FROM-WHERE
4. SQL-DML之更新INSERT/UPDATE/DELETE
5. SQL-视图及DDL的进一步介绍

1. SQL语言简述

SQL语言提出和发展

标准的关系
数据库语言

- 1974年，由Boyce和Chamber提出
- 1975-1979年，在System R上首次实现，由IBM的San Jose研究室研制，称为Sequel(Structured English QUEry Language)
- 1986年推出了SQL标准：SQL-86，“数据库语言SQL：Structured Query Language”
- 1989年ANSI / ISO推出了SQL标准: SQL-89, 数据库语言SQL的标准集合
- 1992年进一步推出了SQL标准：SQL-92，也称为SQL2，
  - 是SQL-89的超集
  - 增加了许多新特性，如新数据类型，更丰富数据操作，更强完整性支持等
  - 原SQL-89被称为entry-SQL, 扩展的被称为Intermediate级和Full级

1. SQL语言简述

SQL语言提出和发展(续)

面向对象和对象关系数据库

- 1999年进一步推出了SQL标准： SQL-99，也称为SQL3
  - 对面向对象的一些特征予以支持，支持抽象数据类型，支持行对象和列对象等
  - 对递归、触发等复杂操作也予以规范化定义
  - 有些特征，现有数据库厂商尚不能做到完全支持
  - 废弃了SQL2的分级，但定义了core-SQL及扩展的SQL

数据库应用程序

- SQL 2003； SQL 2006； SQL 2008；…； SQL 2019
- SQL还有一个标准是X/Open标准，主要强调各厂商产品的可移植性，只包含被各厂商广泛认可的操作
- 标准：主要用来衡量一个软件商的产品是否符合共同的约定
- 标准：使用户可以学习“标准”规定的语言，无需关注具体的软件产品
- 注意：实验采用Mysql8.0，支持SQL-99语法集，差异需要阅读文档

1. SQL语言简述

基本（标准）SQL语言

- SQL是集DDL(数据定义)、DML（数据操纵）和DCL（数据控制）于一体的数据库语言
  - ① DDL：例如：CREATE(建立), ALTER (修改), DROP(撤消)
    - 模式的定义和删除，包括定义DATABASE, TABLE, VIEW, INDEX,完整 性约束条件等，也包括定义对象(ROWTYPE行对象, TYPE列对象)
  - ② DML语句引导词：INSERT, UPDATE, DELETE, SELECT…FROM…WHERE…
    - 数据的增、删、改等编辑操作（插入、删除、更新）
    - 数据的查询：6大语句组成的基本结构、连接操作、子查询嵌套等
  - ③ DCL语句引导词：GRANT, REVOKE, COMMIT, ROLLBACK等
    - 安全性控制：授权和撤消各类授权
    - 事务管理、数据库恢复等
    - 主要用于数据库的控制、安全和管理等（DBA职责）

使用形式：交互式SQL->嵌入式SQL->动态SQL，理解查询需求，用SQL正确表达

1. SQL语言简述

扩展SQL语言

- 支持更为复杂的功能
  - 定义存储过程、函数、触发器等功能模块
  - 定义和使用变量；使用循环和条件判断等编程结构
  - 顺序执行命令和标准SQL语句，使用游标(Cursor)对数据表内容进行逐行访问

- 不同数据库厂商有各自的产品和实现方法
  - 微软SQL Server: T-SQL (Transact SQL)
  - 甲骨文Oracle DB: Procedural SQL (PL/SQL)
  - 其他
- 关键字、语法、数据类型等有一定差别，但基本的逻辑大致相同，知识可以迁移。

第3章　SQL语言

学习内容

1. SQL语言概述
2. 简单的SQL-DDL/DML: 创建数据库
   - ① 课堂讲义中的示例数据库
   - ② 创建数据库之CREATE TABLE: 定义表
   - ③ 创建数据库之INSERT：追加表中的元组
3. SQL-DML之查询SELECT
4. SQL-DML之更新INSERT/UPDATE/DELETE
5. SQL-视图及DDL的进一步介绍

2. 简单的SQL-DDL/DML: 创建数据库

学生选课数据库SCT（5 个关系）

- 学生： Student (SNo char(8), Sname char(10), Gender char(2), Sage integer, DNo char(2), Sclass char(6))
  学号 姓名 性别 年龄 系号 班号
- 院系： Dept (DNo char(2), Dname char(10), Dean char(10))
  系号 系名 系主任名
- 课程： Course (CNo char(3), Cname char(12), Chours integer, Credit float(1), TNo char(3))
  课号 课名 学时数 学分 授课教师编号
- 教师： Teacher (TNo char(3), Tname char(10), DNo char(2), Salary float(2))
  教师编号 教师姓名 系号 工资
- 选课： SC(SNo char(8), CNo char(3), Score float(1))
  学号 课号 分数

2. 简单的SQL-DDL/DML: 创建数据库

Student

SNo	Sname	Gender	Sage	DNo	Sclass
98030101	张三	男	20	03	980301
98030102	张四	女	20	03	980301
98030103	张五	男	19	03	980301
98040201	王三	男	20	04	980402
98040202	王四	男	21	04	980402
98040203	王五	女	19	04	980402

SC

SNo	CNo	Score
98030101	001	92
98030101	002	85
98030101	003	88
98040202	002	90
98040202	003	80
98040202	001	55
98040203	003	56
98030102	001	54
98030102	002	85
98030102	003	48

Course

CNo	Cname	Chours	Credit	TNo
001	数据库	40	6	001
003	数据结构	40	6	003
004	编译原理	40	6	001
005	C语言	30	4.5	003
002	高等数学	80	12	004

Dept

DNo	Dname	Dean
01	机电	李三
02	能源	李四
03	计算机	李五
04	自动控制	李六

Teacher

TNo	Tname	DNo	Salary
001	赵三	01	1200.00
002	赵四	03	1400.00
003	赵五	03	1000.00
004	赵六	04	1100.00

2. 简单的SQL-DDL/DML: 创建数据库

- 如何创建这样一个数据库?
- 包括两件事：定义数据库和表（DDL，表的格式），向表中添加元组（DML，表的具体数据内容）
- 第一步：创建数据库，并打开（使用）这个数据库
- 第二部：逐一创建5个表，定义其关系模式，并定义其完整性约束条件
- 第三步：为每个表添加数据

2. 简单的SQL-DDL/DML: 创建数据库

- 如何创建这样一个数据库?

- 第一步：创建一个叫SCT的数据库，并打开（使用）这个数据库

```
CREATE DATABASE SCT;
USE SCT;
```

说明：
1. 关键字、表名、列名不区分大小写
2. 两条SQL语句之间用分号隔开。

语法：CREATE DATABASE 数据库名;
USE 数据库名;

2. 简单的SQL-DDL/DML: 创建数据库

- 如何创建这样一个数据库?

- 第二步：逐一创建5个表，并添加约束条件：用 CREATE TABLE 命令。

```
CREATE TABLE Student
(SNo char(8) not null,
 Sname char(10),
 Gender char(2),
 Sage integer,
 DNo char(2),
 Sclass char(6),
 Primary Key (SNo)
);
```

表名，用户定义

属性（列）定义，逗号分隔

列名

列的数据类型

列的约束条件(如有)
例如非空 not null
无重复值 unique 等

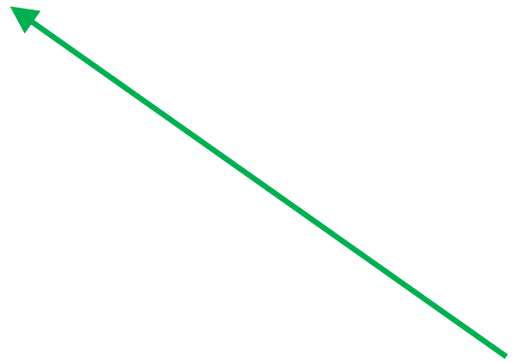
主键约束

2. 简单的SQL-DDL/DML: 创建数据库

- 如何创建这样一个数据库?

- 第二步：逐一创建5个表，并添加约束条件：用 CREATE TABLE 命令。

```
CREATE TABLE Student
( SNo  char(8)  PRIMARY KEY NOT NULL,
  Sname  char(10),
  Gender  char(2),
  Sage  integer,
  DNo  char(2),
  Sclass  char(6)
);
```



也可以将主键约束放在列约束中

2. 简单的SQL-DDL/DML: 创建数据库

创建Table

- 创建Table，需使用CREATE TABLE语句
- CREATE TABLE简单语法形式为：

CREATE TABLE 表名(列名 数据类型 [*PRIMARY KEY*|*UNIQUE*] [*NOT NULL*]

[, 列名 数据类型 [*NOT NULL*] , …][表约束条件])

- “[]”：表示里面的内容可以省略，“|”表示其隔开的两项可以任选其一
- **PRIMARY KEY:** 主键约束，给定的一列或多列，每个表只能创建一个主键约束
- **UNIQUE:** 唯一性约束，候选键，一个表可以有多个UNIQUE 约束
- **Not Null:** 是指该列允许不允许有空值出现，如选择了not null表明该列不允许有空值出现。通常主键是不允许有空值的。

```
CREATE TABLE Student ( SNo char(8) primary key, Sname char(10),
                       Gender char(2), Sage integer, DNo char(2), Sclass char(6) );
```

2. 简单的SQL-DDL/DML: 创建数据库

创建Table(续)(自学)

- 在SQL-92标准中定义的数据类型

  Char (n): 固定长度的字符串

  Varchar (n):可变长字符串

  int: 整数　// 有时不同系统也写作**integer**

  Numeric (p，q): 固定精度数字，一共有p位数字，右边q位

  real: 浮点精度数字 //有时不同系统也写作**float(n)**，小数点后保留n位

  Date: 日期 (2003-09-12)

  time: 时间 (23:15:003)

  …

- 各个商用DBMS的数据类型或有差异，用时注意

2. 简单的SQL-DDL/DML: 创建数据库

Student (SNo char(8), Sname char(10), Gender char(2), Sage integer, DNo char(2), Sclass char(6))

创建Table(续)

- 再例如定义课程表：Course

```
CREATE TABLE Course ( CNo char(3) Primary Key, Cname
        char(12), Chours integer, Credit float(1), TNo char(3) );
```

- 再定义选课：SC（复合主键，外键）

```
CREATE TABLE SC (SNo char(8),
                 CNo char(3),
                 Score float(1),
                 Primary Key(SNo, CNo),
                 Foreign Key (SNo) References Student(SNo),
                 Foreign Key (CNo) References Course(CNo)
                );
```

额外的说明：

1. 定义外键的参照完整性约束时，需要考虑建表的顺序问题。
2. 也可以建好表后再添加外键约束

2. 简单的SQL-DDL/DML: 创建数据库

删除Table

- DROP TABLE 表名

```
DROP TABLE Course;
```

- 把Course表（及其数据）删除。

- 该操作能否成功?
- 涉及到数据一致性问题。后面的课会介绍。

2. 简单的SQL-DDL/DML: 创建数据库

向表中追加元组的值

在表中追加元组的值要使用DML

- DML: Data Manipulation Language
  - ① 向TABLE中追加新的元组：INSERT
  - ② 修改TABLE中某些元组中的某些属性的值: UPDATE
  - ③ 删除TABLE中的某些元组: DELETE
  - ④ 对TABLE中的数据进行各种条件的检索: SELECT, FROM, WHERE…
- DML通常由用户或应用程序员使用，访问经授权的数据库
- 先学习INSERT的简单形式

2. 简单的SQL-DDL/DML: 创建数据库

向表中追加元组的值(续)

- 追加元组，需使用 **INSERT INTO** 语句
- **INSERT INTO** 简单语法形式为：

INSERT INTO 表名[(列名 [,列名]…]
VALUES (值 [,值],…);

- 语法中的VALUES后值的排列，须与INTO子句后面的列名排列一致。
- 若表名后的所有列名省略， 则VALUES后的值的排列，须与该表存储中的列名排列一致。

2. 简单的SQL-DDL/DML: 创建数据库

向表中追加元组的值(续)

```
INSERT INTO  Student ( SNo, Sname, Gender, Sage)
VALUES ( '98030102', '张四', '女', 20);
```

- 例如：追加学生表中的元组

```
INSERT INTO  Student
VALUES ( '98030101', '张三', '男', 20, '03', '980301');

INSERT INTO  Student ( SNo, Sname, Gender, Sage, DNo , Sclass)
VALUES ( '98030102', '张四', '女', 20, '03', '980301');
```

Student					
SNo	Sname	Gender	Sage	DNo	Sclass
98030101	张三	男	20	03	980301
98030102	张四	女	20	Null	Null

- 再例如：追加课程表中的元组

```
INSERT INTO  Course   /*所有列名省略，须与定义或存储的列名顺序一致
VALUES ( '001', '数据库', 40, 6, '001'); /*如列名未省略，须与语句中列名的顺序一致

INSERT INTO  Course(Cname, CNo, Credit, Chours, TNo)
VALUES ('数据库', '001', 6, 20, '001');
```

学习内容

1. SQL语言概述
2. 简单的SQL-DDL/DML: 创建数据库
3. SQL-DML之查询SELECT
   - ① ---- 基本的检索操作　　---- 多表联合检索
   - ② ---- 子查询　　---- 结果计算与聚集函数
   - ③ ---- 分组查询与分组过滤　---- 并、交、差的处理
   - ④ ---- 空值处理　　---- 内连接、外连接
   - ⑤ ---- SQL的完整语法
4. SQL-DML之更新INSERT/UPDATE/DELETE
5. SQL-视图及DDL的进一步介绍

3. SQL-DML之查询SELECT

基本的检索操作

- SQL提供了结构形式一致但, 功能多样化的检索语句:

  SELECT 列名 [[, 列名] …]

  FROM 表名

  [*WHERE* 检索条件];

最基本的

- 语义: 从表名所给出的表中，查询出满足检索条件的元组，并按给定的列名及顺序进行投影显示，相当于$\pi_{列名,\cdots,列名}(\sigma_{检索条件}(表名))$
- SELECT语句中的SELECT … , FROM… , WHERE…, 等被称为子句
- 另外三个子句为: GROUP BY… , HAVING…, ORDER BY…，后面介绍
- 通过六个子句的组合，扩展以及关键字的配合，可以实现很多复杂查询。

3. SQL-DML之查询Select

Student

SNo	Sname	Gender	Sage	DNo	Sclass
98030101	张三	男	20	03	980301
98030102	张四	女	20	03	980301
98030103	张五	男	19	03	980301
98040201	王三	男	20	04	980402
98040202	王四	男	21	04	980402
98040203	王五	女	19	04	980402

基本的检索操作

- 例如：查询学生表中所有学生的信息

```
SELECT SNo, Sname, Gender, Sage, Sclass, DNo
FROM   Student;

SELECT *   FROM  Student;  //如投影所有列，则可以用*来简写
```

- 再如：查询学生表中所有学生的姓名及年龄

```
SELECT  Sname, Sage                    //投影出某些列
 FROM  Student;
```

Sname	Sage
张三	20
张四	20
张五	19
王三	20
王四	21
王五	19

- 再如：查询学生表中所有年龄不大于19岁的学生的年龄及姓名

```
SELECT  Sage, Sname                    //投影的列可以重新排定顺序
  FROM  Student
WHERE  Sage <= 19;
```

Sage	Sname
19	张五
19	王五

3. SQL-DML之查询SELECT DISTINCT

基本的检索操作

- 结果唯一性问题。尽管关系模型要求无重复元组出现在数据库中，但现实DBMS操作中，是允许检索结果出现重复元组的，但也允许无重复元组。
- 在关系(存储的Table)中要求无重复元组是通过定义主键或UNIQUE来保证的，在检索结果中要求无重复元组，是通过DISTINCT保留字的使用来实现的。
- 例如：在选课表中，检索成绩大于80分的所有学号

```
SELECT SNo FROM SC
WHERE Score>80; //有重复元组出现，比如一个同学两门以上课程大于80
```

SNo
98030101
98030101
98030101
98040202

```
SELECT DISTINCT SNo
FROM SC
WHERE Score>80; //重复元组被DISTINCT过滤掉，只保留一份
```

SNo
98030101
98040202

3. SQL-DML之查询WHERE

基本的检索操作: 用WHERE筛选符合条件的行

- 检索条件的书写，与选择运算con(R)的条件con书写是一样的，只是其逻辑运算符用AND, OR, NOT 来表示, 同时也要注意运算符的优先次序及括弧的使用。更要注意对汉语检索条件的正确理解。

检索教师表中所有工资少于1500元或者工资大于2000元并且是03系的教师姓名

- 如何翻译下面的查询?

```
SELECT    Tname
 FROM  Teacher
 WHERE    Salary < 1500  OR  Salary  > 2000  AND  DNo = '03';

SELECT    Tname
 FROM Teacher
 WHERE    (Salary < 1500  OR  Salary  > 2000)  AND  DNo = '03';
```

3. SQL-DML之查询WHERE

基本的检索操作

- 例子：求或者学过001号课程, 或者学过002号课程的学生的学号

```
SELECT SNo FROM SC
WHERE CNo = '001' OR CNo='002';
```

- 练习：求所有上过001或002课程，且至少有一门达到90分及以上的同学的学号。

```
SELECT SNo
FROM SC
WHERE (CNo = '001' OR CNo='002' ) AND Score>=90;
```

表（Table）：SC

SNo	CNo	Score
98030101	001	92
98030101	002	85
98030101	003	88
98040202	002	90
98040202	003	80
98040202	001	55
98040203	003	56

3. SQL-DML之查询WHERE

基本的检索操作

- WHERE中可对字符型数据进行查询。比如检索姓张的学生，检索张某某
- 首先：字符串的值在查询中加单引号，区分大小写！
- 其次：字符串间可以比较是否相等、大小（字母序），也可以模糊匹配
- 使用含有*LIKE*运算符的表达式进行模糊匹配

列名[*NOT*] *LIKE* “目标模式”

找出匹配目标字符串的数据。其中目标字符串中可以出现%, _等匹配符。

- 匹配规则：
  - “%”：匹配零个或多个字符
  - “_”：匹配任意单个字符
  - “\”：转义字符，用于去掉一些特殊字符的特定含义
  - 注：通字符看待,如用 “\%”去匹配字符%，用_ 去匹配字符_

3. SQL-DML之查询WHERE

基本的检索操作

- 例如：检索所有叫张五的学生学号及姓名

  SELECT SNo, Sname FROM Student WHERE Sname =‘张五’ ;

- 例如：检索所有姓张的学生学号及姓名

  SELECT SNo, Sname FROM Student WHERE Sname LIKE ‘张%’;

- 再如：检索名字为张某某的所有同学姓名

  SELECT Sname FROM Student WHERE Sname LIKE ‘张__’;

- 再如：检索名字不姓张的所有同学姓名

  SELECT Sname FROM Student WHERE Sname NOT LIKE ‘张%’;

3. SQL-DML之查询WHERE

- 基本的检索操作：IN (NOT IN) 关键字

判断某属性的值是否属于一个给定的集合

- 例如：列出03, 04, 05系同学的所有信息

```
SELECT * FROM  Student
WHERE Dno  IN  ('03', '04' , '05');
```

注意数据类型要匹配。IN后面的集合也可以用另一个查询得到（叫做子查询，后面会讲）

- 上述示例相当于

```
SELECT * FROM Student
WHERE  Sno = '03' OR Sno = '04' OR Sno= '05';
```

3. SQL-DML之查询ORDER BY

基本的检索操作

- 结果排序问题。DBMS可以对检索结果进行排序，可以升序排列，也可以降序排列。
- SELECT语句中结果排序是通过增加*ORDER BY*子句实现的

ORDER BY 列名 [*ASC* | *DESC*]

意义为结果按指定列名进行排序，若后跟asc或省略，则为升序；若后跟 desc, 则为降序。

- 例如：按学号由小到大的顺序显示出所有学生的学号及姓名

```
SELECT SNo, Sname FROM Student ORDER BY SNo ASC
```

（升序可省略）

- 再如：检索002号课大于80分的所有同学学号并按成绩由高到低顺序显示

```
SELECT SNo FROM SC WHERE CNo = '002' and Score > 80 ORDER BY
Score DESC
```

3. SQL-DML之基本查询-综合练习

- 练习：找到所有名字中包含‘乐’字，且年龄在19-22之间（两端包含）的男同学。输出这些同学的姓名和年龄，并按年龄从大到小排序。

```
SELECT  Sname, Sage
FROM    Student
WHERE  Sname  LIKE ‘%乐%’ AND Sage >=19 AND Sage <= 22 AND Gender= ‘男’
ORDER BY Sage DESC;
```

- 练习：找到所有姓名中只包含1个‘乐’字的同学。输出这些同学的姓名和年龄。

```
SELECT  Sname, Sage
FROM    Student
WHERE  Sname  LIKE ‘%乐%’ AND Sname  NOT  LIKE ‘%乐%乐%’;
```

下一个部分

- 多表联合查询
- 输入，即FROM子句中需要用到多个表

3. SQL-DML之查询SELECT

多表联合检索

- 多表联合检索可以通过连接运算来完成，而连接运算又可以通过广义笛卡尔积后再进行选择运算来实现。
- SELECT 的多表联合检索语句如下：

  SELECT 列名 [[, 列名] …] *FROM* 表名1, 表名2, … *WHERE* 检索条件；

  相当于 $\pi_{列名,\cdots,列名}(\sigma_{检索条件}(表名1 \times 表名2 \times \cdots))$

- 检索条件中要包含连接条件，通过不同的连接条件可以实现等值连接、不等值连接及各种连接。

3. SQL-DML之查询SELECT

多表联合检索

- θ-连接之等值连接示例

例如：按“001”号课成绩由高到低的顺序显示出所有学生的姓名(二表连接)

```
SELECT Sname FROM  Student, SC
WHERE Student.SNo = SC.SNo  AND SC.CNo = '001'
ORDER BY  Score  DESC;
```

- 当多表连接时，如果两个表的属性名相同，则需采用表名.属性名方式来 限定该属性是属于哪一个表

再如：按‘数据库’课程成绩由高到低顺序显示所有同学姓名(三表连接)

```
SELECT    Sname  FROM  Student, SC, Course
WHERE Student.SNo =  SC.SNo AND SC.CNo = Course.CNo and Cname =  ‘数据库’
ORDER BY  Score  DESC;
```

3. SQL-DML之查询SELECT

多表联合检索

- 连接运算涉及到重名的问题，如两个表中的属性重名，连接的两个表重名(同一表的自连接)等，因此需要使用别名以便区分

  SELECT中采用别名的方式:

  SELECT 列名 AS 列别名 [[, 列名 AS 列别名] …]

  FROM 表名1 AS 表别名1, 表名2 AS 表别名2, …

  WHERE 检索条件；

- 上述定义中的 AS 可以省略
- 当定义了别名后，在检索条件中可以使用别名来限定属性

3. SQL-DML之查询SELECT

多表联合检索

θ-连接示例

① 例如：找到所有存在薪水差额的两位教师组成的教师对

```
SELECT T1.Tname AS Teacher1, T2.Tname  AS Teacher2
FROM  Teacher  T1, Teacher T2
WHERE  T1.Salary > T2.Salary ;
```

② 例如：找到所有存在年龄差的两位同学的组合

```
SELECT S1.Sname AS Stud1,  S2.Sname  AS Stud2
FROM  Student  S1,  Student  S2
WHERE  S1.Sage > S2.Sage ;
```

- 有时表名很长时，为书写条件简便，也定义表别名，以简化书写

3. SQL-DML之多表联合查询

多表联合检索

- 再如：求既学过“001”号课又学过 “002”号课的所有学生的学号(二表连接)

```
SELECT S1.SNo FROM SC S1, SC S2 WHERE
S1.SNo = S2.SNo AND S1.CNo='001'
        AND S2.CNo='002';
```

- 再如：求“001”号课成绩比“002”号课成绩高的所有学生的学号(二表连接)

```
SELECT S1.SNo FROM SC S1, SC S2 WHERE
        S1.SNo = S2.SNo AND S1.CNo='001'
        AND S2.CNo='002 AND S1.Score > S2.Score;
```

3. SQL-DML之多表联合查询

多表联合检索

- 连接时也可以用JOIN… ON…关键词

```
SELECT S1.SNo
FROM  SC  S1 JOIN  SC  S2  ON S1.SNo = S2.SNo
WHERE  AND  S1.CNo='001' AND   S2.CNo='002 ;
```

- 再如：求“001”号课成绩比“002”号课成绩高的所有学生的学号(二表连接)

```
SELECT S1.SNo
FROM  SC S1 JOIN SC S2 ON S1.SNo = S2.SNo
WHERE S1.CNo='001' AND  S2.CNo='002  AND  S1.Score > S2.Score;
```

3. SQL-DML之多表联合查询

综合练习

① 找到所有来自同一个系的两位不同教师组成的组合。注意不能有重复组合。教师(A, B)的组合与(B, A)的组合属于重复，只保留一个

② 修改以上查询，找到所有工资和少于2500的两位不同教师的组合，仍然不能有重复组合。

Teacher

TNo	Tname	DNo	Salary
001	赵三	01	1200.00
002	赵四	03	1400.00
003	赵五	03	1000.00
004	赵六	04	1100.00