



数据库系统概论

An Introduction to Database System

第三章 关系数据库标准语言 SQL

中国人民大学信息学院

第三章 关系数据库标准语言 SQL



3.1 SQL 概述

3.2 学生 - 课程数据库

3.3 数据定义

3.4 数据查询

3.5 数据更新

3.6 视图

3.7 小结

数据查询



❖ 语句格式

SELECT [ALL|DISTINCT] < 目标列表达式 >

[, < 目标列表达式 >] ...

FROM < 表名或视图名 > [, < 表名或视图名 >] ...

[**WHERE** < 条件表达式 >]

[**GROUP BY** < 列名 1 > [**HAVING** < 条件表达式 >]]

[**ORDER BY** < 列名 2 > [ASC|DESC]] ;

3.4 数据查询



- ❖ 3.4.1 单表查询
- ❖ 3.4.2 连接查询
- ❖ 3.4.3 嵌套查询
- ❖ 3.4.4 集合查询
- ❖ 3.4.5 Select 语句的一般形式

3.4.1 单表查询



❖ 查询仅涉及一个表：

- 一、 选择表中的若干列
- 二、 选择表中的若干元组
- 三、 ORDER BY 子句
- 四、 聚集函数
- 五、 GROUP BY 子句

一、选择表中的若干列



❖ 查询指定列

[例 1] 查询全体学生的学号与姓名。

```
SELECT Sno , Sname  
FROM Student ;
```

[例 2] 查询全体学生的姓名、学号、所在系。

```
SELECT Sname , Sno , Sdept  
FROM Student ;
```

2. 查询全部列



❖ 选出所有属性列：

- 在 **SELECT** 关键字后面列出所有列名
- 将 < 目标列表达式 > 指定为 *

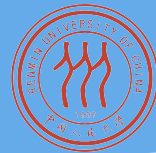
[例 3] 查询全体学生的详细记录。

```
SELECT Sno , Sname , Ssex , Sage , Sdept  
FROM Student ;
```

或

```
SELECT *  
FROM Student ;
```

3. 查询经过计算的值



❖ SELECT 子句的 < 目标列表达式 > 可以为：

- 算术表达式
- 字符串常量
- 函数
- 列别名

查询经过计算的值（续）



[例 4] 查全体学生的姓名及其出生年份。

```
SELECT Sname , 2004-Sage /* 假定当年的年份为 2004 年 */  
FROM Student ;
```

输出结果：

Sname	2004-Sage
-------	-----------

李勇	1984
刘晨	1985
王敏	1986
张立	1985

查询经过计算的值（续）



[例 5] 查询全体学生的姓名、出生年份和所有系，要求用小写字母表示所有系名

```
SELECT Sname , ' Year of Birth: ' , 2004-Sage ,  
        ISLOWER(Sdept)  
FROM Student ;
```

输出结果：

```
Sname 'Year of Birth:' 2004-Sage ISLOWER(Sdept)
```

李勇	Year of Birth:	1984	cs
刘晨	Year of Birth:	1985	is
王敏	Year of Birth:	1986	ma
张立	Year of Birth:	1985	is

查询经过计算的值（续）



❖ 使用列别名改变查询结果的列标题：

```
SELECT Sname NAME , 'Year of Birth: ' BIRTH ,  
       2000-Sage BIRTHDAY , LOWER(Sdept) DEPARTMENT  
FROM Student ;
```

输出结果：

NAME	BIRTH	BIRTHDAY	DEPARTMENT
李勇	Year of Birth:	1984	cs
刘晨	Year of Birth:	1985	is
王敏	Year of Birth:	1986	ma
张立	Year of Birth:	1985	is

3.4.1 单表查询



❖ 查询仅涉及一个表：

- 一、 选择表中的若干列
- 二、 选择表中的若干元组
- 三、 ORDER BY 子句
- 四、 聚集函数
- 五、 GROUP BY 子句

二、选择表中的若干元组



❖ 1. 消除取值重复的行

如果没有指定 **DISTINCT** 关键词，则缺省为 **ALL**

[例 6] 查询选修了课程的学生学号。

```
SELECT Sno FROM SC ;
```

等价于：

```
SELECT ALL Sno FROM SC ;
```

执行上面的 **SELECT** 语句后，结果为：

Sno

200215121

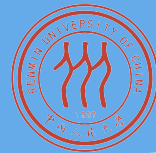
200215121

200215121

200215122

200215122

消除取值重复的行（续）



- ❖ 指定 DISTINCT 关键词，去掉表中重复的行

```
SELECT DISTINCT Sno  
FROM SC ;
```

执行结果：

<u>Sno</u>
200215121
200215122

2. 查询满足条件的元组



表 3.4 常用的查询条件

□ □ □ □	□ □
□ □	= □ > □ < □ >= □ <= □ != □ <> □ !> □ !< □ NOT+ □ □ □ □ □ □ □ □
□ □ □ □	BETWEEN AND □ NOT BETWEEN AND
□ □ □ □	IN □ NOT IN
□ □ □ □	LIKE □ NOT LIKE
□ □	IS NULL □ IS NOT NULL
多重条件（逻辑运算）	AND , OR , NOT

(1) 比较大小



[例 7] 查询计算机科学系全体学生的名单。

```
SELECT Sname  
FROM Student  
WHERE Sdept='CS' ;
```

[例 8] 查询所有年龄在 20 岁以下的学生姓名及其年龄。

```
SELECT Sname , Sage  
FROM Student  
WHERE Sage < 20 ;
```

[例 9] 查询考试成绩有不及格的学生的学号。

```
SELECT DISTINCT Sno  
FROM SC  
WHERE Grade < 60 ;
```


(2) 确定范围



❖ 谓词： BETWEEN ... AND ...

NOT BETWEEN ... AND ...

[例 10] 查询年龄在 20~23 岁（包括 20 岁和 23 岁）之间的学生的姓名、系别和年龄

```
SELECT Sname , Sdept , Sage
FROM Student
WHERE Sage BETWEEN 20 AND 23 ;
```

[例 11] 查询年龄不在 20~23 岁之间的学生姓名、系别和年龄

```
SELECT Sname , Sdept , Sage
FROM Student
WHERE Sage NOT BETWEEN 20 AND 23 ;
```

(3) 确定集合



❖ 谓词： **IN < 值表 >**, **NOT IN < 值表 >**

[例 12] 查询信息系（IS）、数学系（MA）和计算机科学系（CS）学生的姓名和性别。

```
SELECT Sname , Ssex  
FROM Student  
WHERE Sdept IN ( 'IS' , 'MA' , 'CS' );
```

[例 13] 查询既不是信息系、数学系，也不是计算机科学系的学生的姓名和性别。

```
SELECT Sname , Ssex  
FROM Student  
WHERE Sdept NOT IN ( 'IS' , 'MA' , 'CS' );
```

(4) 字符匹配



❖ 谓词： [NOT] LIKE ‘< 匹配串 >’ [ESCAPE ‘< 换码字符 >’]

1) 匹配串为固定字符串

[例 14] 查询学号为 200215121 的学生的详细情况。

```
SELECT *  
FROM Student  
WHERE Sno LIKE '200215121';
```

等价于：

```
SELECT *  
FROM Student  
WHERE Sno = '200215121';
```

字符匹配（续）



2) 匹配串为含通配符的字符串

[例 15] 查询所有姓刘学生的姓名、学号和性别。

```
SELECT Sname , Sno , Ssex  
FROM Student  
WHERE Sname LIKE '刘 %' ;
```

[例 16] 查询姓 " 欧阳 " 且全名为三个汉字的学生的姓名。

```
SELECT Sname  
FROM Student  
WHERE Sname LIKE '欧阳 __' ;
```

字符匹配（续）



[例 17] 查询名字中第 2 个字为 " 阳 " 字的学生的姓名和学号。

```
SELECT Sname , Sno  
FROM Student  
WHERE Sname LIKE '__ 阳 %' ;
```

[例 18] 查询所有不姓刘的学生姓名。

```
SELECT Sname , Sno , Ssex  
FROM Student  
WHERE Sname NOT LIKE '刘 %' ;
```

字符匹配（续）



3) 使用换码字符将通配符转义为普通字符

[例 19] 查询 DB_Design 课程的课程号和学分。

```
SELECT Cno , Ccredit  
FROM Course  
WHERE Cname LIKE 'DB\_Design' ESCAPE '\ ' ;
```

[例 20] 查询以 "DB_" 开头，且倒数第 3 个字符为 i 的课程的具体情况。

```
SELECT *  
FROM Course  
WHERE Cname LIKE 'DB\__%i\__' ESCAPE '\ ' ;
```

ESCAPE ' \ ' 表示 “ \ ” 为换码字符

(5) 涉及空值的查询



- 谓词： IS NULL 或 IS NOT NULL
- “IS” 不能用 “=” 代替

[例 21] 某些学生选修课程后没有参加考试，所以有选课记录，但没有考试成绩。查询缺少成绩的学生的学号和相应的课程号。

```
SELECT Sno , Cno  
FROM SC  
WHERE Grade IS NULL
```

[例 22] 查所有有成绩的学生学号和课程号。

```
SELECT Sno , Cno  
FROM SC  
WHERE Grade IS NOT NULL ;
```

(6) 多重条件查询



❖ 逻辑运算符： AND 和 OR 来联结多个查询条件

- AND 的优先级高于 OR
- 可以用括号改变优先级

❖ 可用来实现多种其他谓词

- [NOT] IN
- [NOT] BETWEEN ... AND ...

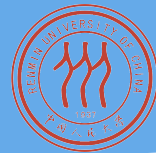
多重条件查询（续）



[例 23] 查询计算机系年龄在 20 岁以下的学生姓名。

```
SELECT Sname  
FROM Student  
WHERE Sdept= 'CS' AND Sage<20 ;
```

多重条件查询（续）



❖ 改写 [例 12]

[例 12] 查询信息系（IS）、数学系（MA）和计算机科学系（CS）学生的姓名和性别。

```
SELECT Sname , Ssex  
FROM Student  
WHERE Sdept IN ( 'IS' , 'MA' , 'CS' )
```

可改写为：

```
SELECT Sname , Ssex  
FROM Student  
WHERE Sdept= ' IS ' OR Sdept= ' MA' OR Sdept= ' CS ' ;
```

3.4.1 单表查询



❖ 查询仅涉及一个表：

- 一、 选择表中的若干列
- 二、 选择表中的若干元组
- 三、 **ORDER BY** 子句
- 四、 聚集函数
- 五、 **GROUP BY** 子句

三、ORDER BY 子句



❖ ORDER BY 子句

- 可以按一个或多个属性列排序
- 升序：ASC；降序：DESC；缺省值为升序

❖ 当排序列含空值时

- ASC：排序列为空值的元组最后显示
- DESC：排序列为空值的元组最先显示

ORDER BY 子句 (续)



[例 24] 查询选修了 3 号课程的学生学号及其成绩，查询结果按分数降序排列。

```
SELECT Sno , Grade
FROM SC
WHERE Cno= ' 3 '
ORDER BY Grade DESC ;
```

[例 25] 查询全体学生情况，查询结果按所在系的系号升序排列，同一系中的学生按年龄降序排列。

```
SELECT *
FROM Student
ORDER BY Sdept , Sage DESC ;
```

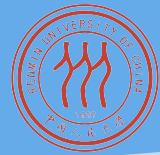
3.4.1 单表查询



❖ 查询仅涉及一个表：

- 一、 选择表中的若干列
- 二、 选择表中的若干元组
- 三、 ORDER BY 子句
- 四、 聚集函数
- 五、 GROUP BY 子句

四、聚集函数



❖ 聚集函数：

- 计数

COUNT ([DISTINCT|ALL] *)

COUNT ([DISTINCT|ALL] < 列名 >)

- 计算总和

SUM ([DISTINCT|ALL] < 列名 >)

- 计算平均值

AVG ([DISTINCT|ALL] < 列名 >)

- 最大最小值

MAX ([DISTINCT|ALL] < 列名 >)

MIN ([DISTINCT|ALL] < 列名 >)

聚集函数（续）



[例 26] 查询学生总人数。

```
SELECT COUNT(*)  
FROM Student ;
```

[例 27] 查询选修了课程的学生人数。

```
SELECT COUNT(DISTINCT Sno)  
FROM SC ;
```

[例 28] 计算 1 号课程的学生平均成绩。

```
SELECT AVG(Grade)  
FROM SC  
WHERE Cno= ' 1 ' ;
```


聚集函数（续）



[例 29] 查询选修 1 号课程的学生最高分数。

```
SELECT MAX(Grade)
FROM SC
WHERE Cno= ' 1 ' ;
```

[例 30] 查询学生 200215012 选修课程的总学分数。

```
SELECT SUM(Ccredit)
FROM SC , Course
WHERE Sno='200215012' AND SC.Cno=Course.Cno;
```

3.4.1 单表查询



❖ 查询仅涉及一个表：

- 一、 选择表中的若干列
- 二、 选择表中的若干元组
- 三、 ORDER BY 子句
- 四、 聚集函数
- 五、 GROUP BY 子句

五、GROUP BY 子句



❖ GROUP BY 子句分组：

细化聚集函数的作用对象

- 未对查询结果分组，聚集函数将作用于整个查询结果
- 对查询结果分组后，聚集函数将分别作用于每个组
- 作用对象是查询的中间结果表
- 按指定的一列或多列值分组，值相等的为一组

GROUP BY 子句（续）



[例 31] 求各个课程号及相应的选课人数。

```
SELECT Cno , COUNT(Sno)
FROM SC
GROUP BY Cno ;
```

查询结果：

Cno	COUNT(Sno)
1	22
2	34
3	44
4	33
5	48

GROUP BY 子句（续）



[例 32] 查询选修了 3 门以上课程的学生学号。

```
SELECT Sno  
FROM SC  
GROUP BY Sno  
HAVING COUNT(*) >3 ;
```

GROUP BY 子句（续）



❖ HAVING 短语与 WHERE 子句的区别：

- 作用对象不同
- WHERE 子句作用于基表或视图，从中选择满足条件的元组
- HAVING 短语作用于组，从中选择满足条件的组。

下课了。。。



追求



休息一会儿。。。



www.hesee.com