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

数据查询

（单表查询）

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>]...|(SELECT 语句)

[AS]<别名>

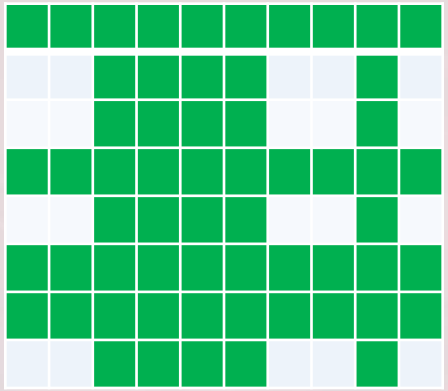
[WHERE <条件表达式>]

[GROUP BY <列名1> [HAVING <条件表达式>]]

[ORDER BY <列名2> [ASC|DESC]];

数据查询

- SELECT子句：指定要显示的属性列
- FROM子句：指定查询对象（基本表或视图）
- WHERE子句：指定查询条件
- GROUP BY子句：对查询结果按指定列的值分组，该属性列值相等的元组为一个组。通常会在每组中作用聚集函数。
- HAVING短语：只有满足指定条件的组才予以输出
- ORDER BY子句：对查询结果表按指定列值的升序或降序排序



SELECT-FROM-WHERE

201215121	李勇	男	20	CS
201215122	刘晨	女	19	CS
201215123	王敏	女	18	MA
201215125	张立	男	19	IS

GROUP BY-HAVING
（常伴有聚集函数）

男	2
女	2

ORDER BY

女	2
男	2

1.选择表中的若干列

❖查询指定列

[例3.16] 查询全体学生的学号与姓名。

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

选择表中的若干列（续）

- 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3.18] 查询全体学生的详细记录

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

或

```
SELECT  *
FROM Student;
```

查询经过计算的值（续）

- “虚”列
  - SELECT子句的<目标列表达式>不仅可以为表中的属性列，也可以是表达式

[例3.19] 查全体学生的姓名及其出生年份。

```
SELECT Sname,2016-Sage
FROM Student;
```

Sname	2014-Sage
李勇	1994
刘晨	1995
王敏	1996
张立	1995

查询经过计算的值（续）

[例3.20] 查询全体学生的姓名、出生年份和所在的院系，要求用小写字母表示系名。

```
SELECT Sname,'Year of Birth: ',2014-Sage,LOWER(Sdept)
FROM Student;
```

输出结果：

Sname	'Year of Birth:'	2014-Sage	LOWER(Sdept)
李勇	Year of Birth:	1994	cs
刘晨	Year of Birth:	1995	cs
王敏	Year of Birth:	1996	ma
张立	Year of Birth:	1995	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,
       2014-Sage  BIRTHDAY,LOWER(Sdept)  DEPARTMENT
FROM Student;
```

输出结果:

NAME	BIRTH	BIRTHDAY	DEPARTMENT
李勇	Year of Birth:	1994	cs
刘晨	Year of Birth:	1995	cs
王敏	Year of Birth:	1996	ma
张立	Year of Birth:	1995	is

2. 选择表中的若干元组

- 消除取值重复的行

  如果没有指定DISTINCT关键词，则缺省为ALL

  [例3.21] 查询选修了课程的学生学号。

  ```
  SELECT Sno   FROM SC;
  ```

  结果为：

Sno
201215121
201215121
201215121
201215122
201215122

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：

Sno
201215121
201215122

（2）查询满足条件的元组

表3.6 常用的查询条件

查 询 条 件	谓　词
比　较	=, >, <, >=, <=, !=, <>, !>, !<; NOT+上述比较运算符
确定范围	BETWEEN AND, NOT BETWEEN AND
确定集合	IN, NOT IN
字符匹配	LIKE, NOT LIKE
空　值	IS NULL, IS NOT NULL
多重条件（逻辑运算）	AND, OR, NOT

① 比较大小

[例3.22] 查询计算机科学系全体学生的名单。

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

[例3.23]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3.24]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3.25] 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;
```

② 确定范围

❖谓词： BETWEEN … AND …
NOT BETWEEN … AND …

[例3.26] 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3.27]查询计算机科学系（CS）、数学系（MA）和信息系（IS）学生的姓名和性别。

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

③ 确定集合

❖谓词：IN <值表>，NOT IN <值表>

[例3.28]查询既不是计算机科学系、数学系，也不是信息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>’]

<匹配串>可以是一个完整的字符串，也可以含有通配符%和 _

- % （百分号） 代表任意长度（长度可以为0）的字符串
  例如a%b表示以a开头，以b结尾的任意长度的字符串
- _ （下横线） 代表任意单个字符。
  例如a_b表示以a开头，以b结尾的长度为3的任意字符串

字符匹配（续）

- 匹配串为固定字符串

[例3.29] 查询学号为201215121的学生的详细情况。

```
SELECT *
FROM  Student
WHERE  Sno LIKE '201215121';
```

等价于：

```
SELECT  *
FROM  Student
WHERE Sno = ' 201215121 ';
```

字符匹配（续）

- 匹配串为含通配符的字符串

[例3.30] 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%';
```

字符匹配（续）

- 匹配串为含通配符的字符串

[例3.31] 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3.32] 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3.33] 查询所有不姓刘的学生姓名、学号和性别。

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

字符匹配（续）

- 使用换码字符将通配符转义为普通字符

[例3.34] 查询DB_Design课程的课程号和学分。

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

字符匹配（续）

- 使用换码字符将通配符转义为普通字符

[例3.35] 查询以"DB_"开头，且倒数第3个字符为 i的课程的详细情况。

```
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3.36] 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
```

⑤ 涉及空值的查询

❖谓词： IS NULL 或 IS NOT NULL

■ “IS” 不能用 “=” 代替

[例3.37] 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来连接多个查询条件

- AND的优先级高于OR
- 可以用括号改变优先级

[例3.38] 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3.27]

[例3.27] 查询计算机科学系（CS）、数学系（MA）和信息系（IS）学生的姓名和性别。

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

可改写为：

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

3.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
- 对于空值，排序时显示的次序由具体系统实现来决定

ORDER BY子句（续）

[例3.39]查询选修了3号课程的学生的学号及其成绩，查询结果按分数降序排列。

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

ORDER BY子句（续）

[例3.40]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;
```

数据查询

（单表查询2 ）

数据查询

- SELECT子句：指定要显示的属性列
- FROM子句：指定查询对象（基本表或视图）
- WHERE子句：指定查询条件
- GROUP BY子句：对查询结果按指定列的值分组，该属性列值相等的元组为一个组。通常会在每组中作用聚集函数。
- HAVING短语：只有满足指定条件的组才予以输出
- ORDER BY子句：对查询结果表按指定列值的升序或降序排序

3.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
- 对于空值，排序时显示的次序由具体系统实现来决定

ORDER BY子句（续）

[例3.39]查询选修了3号课程的学生的学号及其成绩，查询结果按分数降序排列。

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

ORDER BY子句（续）

[例3.40]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;
```

4. 聚集函数

❖ 聚集函数：

- 统计元组个数

  COUNT(*)

- 统计一列中值的个数

  COUNT([DISTINCT|ALL] <列名>)

- 计算一列值的总和（此列必须为数值型）

  SUM([DISTINCT|ALL] <列名>)

- 计算一列值的平均值（此列必须为数值型）

  AVG([DISTINCT|ALL] <列名>)

- 求一列中的最大值和最小值

  MAX([DISTINCT|ALL] <列名>)

  MIN([DISTINCT|ALL] <列名>)

聚集函数（续）

[例3.41] 查询学生总人数。

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

[例3.42] 查询选修了课程的学生人数。

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

[例3.43] 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3.44] 查询选修1号课程的学生最高分数。

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

[例3.45] 查询学生201215012选修课程的总学分数。

```
SELECT SUM(Ccredit)
FROM  SC,Course
WHERE Sno='201215012' AND SC.Cno=Course.Cno;
```

5. GROUP BY子句

❖ **GROUP BY子句分组：**

细化聚集函数的作用对象

- 如果未对查询结果分组，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
- 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3.46] 求各个课程号及相应的选课人数。

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

查询结果可能为：

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

GROUP BY子句（续）

[例3.47] 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）

[例3.48]查询平均成绩大于等于90分的学生学号和平均成绩

下面的语句是不对的：

```
SELECT Sno, AVG(Grade)
FROM  SC
WHERE AVG(Grade)>=90
GROUP BY Sno;
```

因为WHERE子句中是不能用聚集函数作为条件表达式

正确的查询语句应该是：

```
SELECT  Sno, AVG(Grade)
FROM  SC
GROUP BY Sno
HAVING AVG(Grade)>=90;
```

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。

6 综合练习

[练习1] 列出计算机系姓刘的同学的信息，按照学号大小排序

```
SELECT *
FROM  Student
WHERE Sdept='CS' AND Sname LIKE  '刘%'
 ORDER BY Sno;
```

6 综合练习

[练习2] 按系并区分男女统计各系学生的人数、并按照人数降序排序

```
SELECT Sdept, Ssex,COUNT(Sno)
FROM  Student
GROUP BY Sdept,Ssex
 ORDER BY COUNT(Sno) DESC;
```