

数据库系统概论

参考: 第三章 关系数据库标准查询语言 P89-P115

本章内容

- ●第一节 SQL概述
- ●第二节 学生-课程数据库
- ●第三节 数据定义
- ●第四节 数据查询
- ●第五节 数据更新
- ●第六节 视图
- ●第七节 小结



本节目标

- ●使用select语句
 - 查询满足一定条件的元组;
 - 查询某些属性的值;
 - 使用表别名和列别名;
 - 利用DISTINCT去掉查询结果中的重复行;
 - 通过在WHERE子句中放入连接条件,进行 多表连接查询;
 - 利用GROUP BY进行分组统计
 - 利用ORDER对查询结果按要求排序;



"数据查询"内容概要

- ❖查询语句概述
 - ■基本语法
 - ■子句功能
 - select语句的含义
- ◆单表查询
- ❖连接查询
- ❖嵌套查询
- ❖集合查询

一、基本语法

❖基本语法

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〉[, 〈列名〉]...

[HAVING〈内部函数表达式〉]]

[ORDER BY 〈列名〉 [ASC | DESC] [, 〈列名〉[ASC | DESC]]...]

二、子句功能

❖子句功能

- SELECT子句与FROM子句是必选子句;
- SELECT ---- 列出查询的结果;
- FROM ---- 指明所访问的对象;
- WHERE ---- 指定查询的条件;
- GROUP BY ---- 将查询结果按指定字段的取值分组;
- HAVING ----筛选出满足指定条件的组;
- ORDER BY ---- 按指定的字段的值,以升序或降序排列查询结果

三、SELECT语句的含义

❖ SELECT语句的含义

- 根据WHERE子句中的条件表达式,从FROM子句中的基本 表或视图中找出满足条件的元组;
- 按SELECT子句中的目标字段,选出元组中的分量形成结果 表;
- GROUP BY子句将结果按字段分组,每个组产生结果表中的一个元组;
- 通常在每组中作用库函数,分组的附加条件用HAVING短语给出;只有满足内部函数表达式的组才予输出;
- 如果有ORDER BY子句,则结果表要根据指定的字段按升序或降序排列。

"数据查询"内容概要

- ❖查询语句概述
- ❖单表查询
 - ■投影查询
 - ■选择查询
 - order by子句
 - ■聚集函数
 - Group by子句
- ❖连接查询
- ❖嵌套查询
- ❖集合查询

单表查询一、投影

SELECT 〈目标列表达式〉 FROM 〈表名或视图名〉 目标表达式可以是:属性名、算术表达式、字符串常量、函数等。

❖属性名

例1 查询全体学生的学号、姓名、所在系

SELECT, Sno, Sname, Sdept

FROM Student;

例2 查询全体学生的详细记录

SELECT *

FROM Student;

❖属性名表达式、常量或函数

例3 查询全体学生的姓名、出生年份

SELECT Sname, 2016 – Sage Birthday FROM Student:

例4 查询全体学生的人数
SELECT count(Sname) 学生人数
FROM Student:

例5 在每个学生的姓名后面显示字符串 2007 SELECT Sname,'2007' FROM student



单表查询二、选择

❖取消重复行

■ 在SELECT子句中使用DISTINCT短语

例6 查询选修了课程的学生的学号

SELECT Sno FROM SC:



SELECT DISTINCT Sno FROM SC:



单表查询二、选择(续)

❖注意 DISTINCT短语的作用范围是所有目标列

例: 查询选修课程的各种成绩 SELECT DISTINCT Cno, Grade FROM SC;

单表查询二、选择(续)

❖查询满足条件的元组 (where子句)

WHERE子句常用的查询条件

表3.4 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

- ❖在WHERE子句的<比较条件>中使用比较运算符
 - =, >, <, >=, <=, != 或 <>, !>, !<,
 - 逻辑运算符NOT + 比较运算符

[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;



SELECT Sname, Sage
FROM Student
WHERE NOT Sage >= 20;

课堂练习

❖练习一

- ■查询性别为女的学生的学号、姓名
- 查询学分为4学分的课程的名字
- 查询成绩在85分以上的学生的学号

2 确定范围

❖使用调词 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;

3 确定集合

- ◆使用谓词 IN <值表>, NOT IN <值表>
 - <值表>: 用逗号分隔的一组取值

[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');

4 字符串匹配

- ❖ [NOT] LIKE '<で配串>' [ESCAPE '<換码字符>']
 - <匹配串>: 指定匹配模板,可以是固定字符串或含通 配符的字符串
 - 当匹配模板为固定字符串时,可以用 = 运算符取代 LIKE 谓词,用!= 或 < >运算符取代 NOT LIKE 谓词
 - ■通配符
 - %(百分号)代表任意长度(长度可以为0)的字符串
 - _(下横线) 代表任意单个字符
 - 当用户要查询的字符串本身就含有%或_时,要使用ESCAPE '<换码字符>' 短语对通配符进行转义

[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__';

[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的课程的详细情况。

SELECT *

FROM Course

WHERE Cname LIKE 'DB_%i_ _' ESCAPE '\';

5 涉及空值的查询

- ❖ 使用谓词 IS NULL 或 IS NOT NULL
 - "IS NULL" 不能用 "= NULL" 代替

[例20] 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;

6 多重条件查询

- ❖用逻辑运算符AND和 OR来联结多个查询条件
 - AND的优先级高于OR,但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 ...

[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;

课堂练习

❖练习2

- 查询课程名以"数"开头的所有课程的课程名、学分
- 查询计算机系所有小于20岁的女生的学号、姓名
- 查询先修课为5或7的课程信息

单表查询三、ORDER BY子句

❖使用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

❖当按多个属性排序时

■ 首先根据第一个属性排序,如果在该属性上有多个相同的值时,则按第二个属性排序,以此类推

[例24] 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;

[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;

单表查询四、使用集函数

❖主要集函数

名称	参数类型 (列名)	结果类型	描述
COUNT	任意 或*	数值	计数
SUM	数值型	数值	计算总和
AVG	数值型	数值	计算平均值
MAX	数值型、字符型	同参数类型一样	求最大值
MIN	数值型、字符型	同参数类型一样	求最小值

- DISTINCT短语: 在计算时要取消指定列中的重复值
- ALL短语: 缺省值,不取消重复值

[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':

注意:下列用法错误:

SELECT sno, MIN(gr

FROM sc

SELECT *

ROM sc

WHERE COUNT(*) > 2

单表查询五、对查询结果分组

- ❖使用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

[例27] 求各个课程号及相应的选课人数。

SELECT Cno 课程号, COUNT(Sno) 人数FROM SCGROUP BY Cno;



- GROUP BY子句的作用对象是查询的中间结果表
- 分组方法:按指定的一列或多列值分组,值相等的为 一组
- 使用GROUP BY子句后,SELECT子句的列名列表中 只能出现分组属性和集函数
- ■可以使用HAVING短语筛选最终输出结果

[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

[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;

WHERE和HAVING子句区别

❖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。



"数据查询"内容概要

- ❖查询语句概述
- ❖单表查询
- ❖连接查询
 - 等值与非等值连接查询
 - ■自身连接
 - 外连接复合条件连接
- ❖嵌套查询
- ❖集合查询

连接查询

❖连接查询

- ■同时涉及多个表的查询称为连接查询
- 用来连接两个表的条件称为连接条件或连接谓词,其一般格式为:

[<表名1>.]<列名1> <**比较运算符**> [<表名2>.]<列名2> **比较运算符:** =、>、<、>=、<=、!=

- 连接字段
 - 连接谓词中的列名称为连接字段
 - 连接条件中的各连接字段类型必须是可比的,但不必是相同的

连接操作执行过程

❖一种可能执行步骤

- 首先在表1中找到第一个元组,然后从头开始扫描表2, 逐一查找满足连接条件的元组,找到后就将表1中的第 一个元组与该元组拼接起来,形成结果表中一个元组。
- 表2全部查找完后,再找表1中第二个元组,然后再从 头开始扫描表2,逐一查找满足连接条件的元组,找到 后就将表1中的第二个元组与该元组拼接起来,形成结 果表中一个元组。
- 重复上述操作,直到表1中的全部元组都处理完毕

连接查询一、等值与非等值连接查询

- ❖若连接运算符为 = 时, 称为等值连接
- ◆使用其他运算符时。 称为非等值连接
- ◆在等值连接中, 去掉目标列中的重复属性则为自然连接

[例33] 查询每个学生及其选修课程的情况。

SELECT Student.*, SC.*

FROM Student, SC

WHERE Student.Sno = SC.Sno;

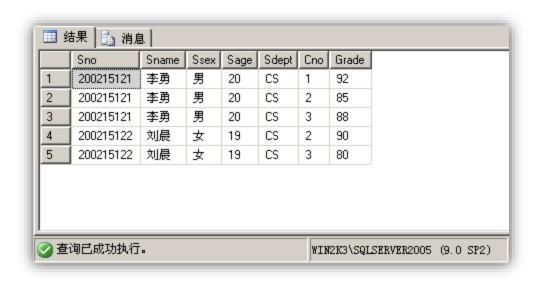
圆绉	Ⅲ 结果 ┃ 消息 ┃											
	Sno	Sname	Ssex	Sage	Sdept	Sno	Cno	Grade				
1	200215121	李勇	男	20	CS	200215121	1	92				
2	200215121	李勇	男	20	CS	200215121	2	85				
3	200215121	李勇	男	20	CS	200215121	3	88				
4	200215122	刘晨	女	19	CS	200215122	2	90				
5	200215122	刘晨	女	19	CS	200215122	3	80				

[例34] 对[例33]用自然连接完成。

SELECT Student.Sno, Sname, Ssex, Sage, Sdept, Cno, Grade

FROM Student, SC

WHERE Student.Sno = SC.Sno;



连接查询二、自身连接

- ❖一个表与其自己进行连接, 称为表的自身连接
 - 需要给表起别名以示区别
 - 由于所有属性名都是同名属性,因此必须使用别名前缀

[例35] 查询每一门课的间接先修课(即先修课的先修课)。

SELECT FIRST.Cno, SECOND.Cpno
FROM Course FIRST, Course SECOND
WHERE FIRST.Cpno = SECOND.Cno;

连接查询三、外连接(Outer Join)

❖外连接与普通连接的区别

- ■普通连接操作只输出满足连接条件的元组
- 外连接操作以指定表为连接主体,将主体表中不满足 连接条件的元组一并输出

[例34] 查询每个学生及其选修课程的情况包括没有选修课程的学生----用外连接操作。

SELECT Student.Sno,Sname,Ssex,Sage,Sdept,Cno,Grade FROM Student LEFT OUTER JOIN SC

ON Student.Sno = **SC.Sno**;

[例35] 用外连接、左连接、右连接完成。

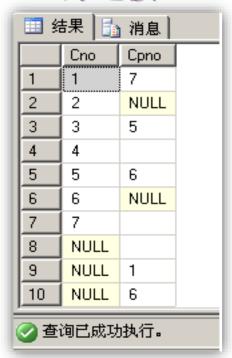
外连接: SELECT FIRST.Cno, SECOND.Cpno
FROM Course FIRST FULL OUTER JOIN Course SECOND

ON FIRST.Cpno = SECOND.Cno;

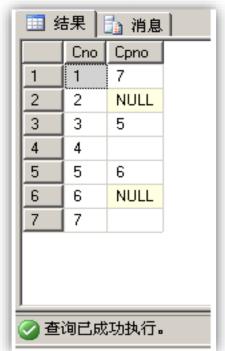
左连接: SELECT FIRST.Cno, SECOND.Cpno
FROM Course FIRST LEFT OUTER JOIN Course SECOND
ON FIRST.Cpno = SECOND.Cno;

右连接: SELECT FIRST.Cno, SECOND.Cpno
FROM Course FIRST RIGHT OUTER JOIN Course SECOND
ON FIRST.Cpno = SECOND.Cno;

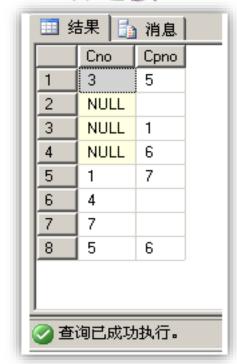
外连接



左连接



右连接



- 在表名后面加外连接操作符指定主体表
- 非主体表有一"万能"的虚行,该行全部由空值组成
- 虚行可以和主体表中所有不满足连接条件的元组进行连接
- 由于虚行各列全部是空值,因此与虚行连接的结果中,来自非主体 表的属性值全部是空值

外连接小结

❖ 左外连接

- 左外连接符为left outer join
- 列出左边关系中所有的元组

❖ 右外连接

- 右外连接符为right outer join
- 列出右边关系中所有的元组

❖ 外连接

- 外连接符为full outer join
- 列出左右两边关系中所有的元组

连接查询四、复合条件连接

WHERE子句中含多个连接条件时,称为复合条件连接

[**例37**] 查询选修2号课程且成绩在90分以上的所有学生的 学号、姓名。

SELECT Student.Sno, student.Sname

FROM Student, SC

WHERE Student.Sno = SC.Sno AND /* 连接谓词*/

SC.Cno='2'AND /* 其他限定条件 */

SC.Grade > 90; /* 其他限定条件 */

[例37] 查询每个学生的学号、姓名、选修的课程名及成绩。

SELECT Student.Sno, Sname, Cname, Grade

FROM Student, SC, Course /*多表连接*/

WHERE Student.Sno = SC.Sno

and SC.Cno = Course.Cno;

"数据查询"内容概要

- ❖查询语句概述
- ◆单表查询
- ❖连接查询
- ❖嵌套查询
 - ■嵌套查询概述
 - ■嵌套查询分类
 - ■嵌套查询求解方法
 - 引出子查询的谓词
- ❖集合查询

嵌套查询一、嵌套查询概述

❖嵌套查询

- 一个SELECT-FROM-WHERE语句称为一个查询块
- 将一个查询块嵌套在另一个查询块的WHERE子句或 HAVING短语的条件中的查询称为<mark>嵌套查询</mark>

例: SELECT Sname

外层查询/父查询

FROM Student

WHERE Sno IN

(SELECT Sno 内层查询/子查询 FROM SC WHERE Cno='2');

嵌套查询一、嵌套查询概述(续)

- 子查询的限制
 - 不能使用ORDER BY子句
- 层层嵌套方式反映了 SQL语言的结构化
- ■有些嵌套查询可以用连接运算替代

嵌套查询二、嵌套查询分类

❖不相关子查询

子查询的查询条件不依赖于父查询

❖相关子查询

子查询的查询条件依赖于父查询

嵌套查询三、嵌套查询求解方法

❖ 不相关子查询

■ 是由里向外逐层处理。即每个子查询在上一级查询处理 之前求解,子查询的结果用于建立其父查询的查找条件。

❖相关子查询

- 首先取外层查询中表的第一个元组,根据它与内层查询相关的属性值处理内层查询,若WHERE子句返回值为真,则取此元组放入结果表;
- 然后再取外层表的下一个元组;
- 重复这一过程,直至外层表全部检查完为止。

嵌套查询四、引出子查询的谓词

- 1. 带有IN谓词的子查询
- 2. 带有比较运算符的子查询
- 3. 带有ANY或ALL谓词的子查询
- 4. 带有EXISTS谓词的子查询

1 带有IN谓词的子查询

[例39] 查询与"刘晨"在同一个系学习的学生。

查询要求可以分步来完成

第一步:确定"刘晨"所在系名

SELECT Sdept

FROM Student

WHERE Sname='刘晨';



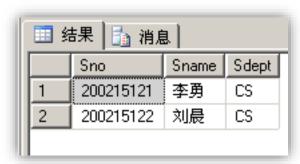
🔢 结果 🛅 消息

第二步: 查找所有在CS系学习的学生。

SELECT Sno, Sname, Sdept

FROM Student

WHERE Sdept= 'CS';



❖构造嵌套查询

■将第一步查询嵌入到第二步查询的条件中

SELECT Sno, Sname, Sdept
FROM Student
WHERE Sdept IN
(SELECT Sdept
FROM Student
WHERE Sname='刘是');



■ 此查询为不相关子查询。DBMS求解该查询时也是分步 去做的。

[例40] 查询选修了课程名为"信息系统"的学生学号和姓名。

SELECT Sno, Sname

FROM Student

WHERE Sno IN

(SELECT Sno

FROM SC

WHERE Cno IN

(SELECT Cno

FROM Course

WHERE Cname='信息系统'));

③ 最后在Student关系中取出Sno和 Sname

② 然后在SC关系中找出选修了3号 课程的学生学号

① 首先在Course关系中找出"信息系统"的课程号,结果为3号

2 带有比较运算符的子查询

- ❖当能确切知道内层查询返回单值时,可用比较运算符(>, <, =, >=, <=, !=或<>)。
- ❖与ANY或ALL谓词配合使用

[例41] 找出每个学生超过他选修课程平均成绩的课程号。

SELECT Sno, Cno

FROM SC x

WHERE Grade >=

(SELECT AVG(Grade)

FROM SC y

WHERE y.Sno = x.Sno);

❖可能的执行过程:

■ S1:从外层查询中取出SC的一个元组x,将元组x的Sno值(200215121)传送给内层查询。

SELECT AVG(Grade) FROM SC y WHERE y.Sno='200215121';

■ S2:执行内层查询,得到值88(近似值),用该值代替内层查询,得到外层查询:

SELECT Sno, Cno FROM SC x WHERE Grade >=88: S3: 执行这个查询,得到

(200215121, 1)

(200215121, 3)

■ S4: 外层查询取出下一个元组重复做上述1至3步骤, 直到外层的SC元组全部处理完毕。结果为:

(200215121, 1)

(200215121, 3)

(200215122, 2)

3 带有ANY或ALL谓词的子查询

❖ 谓词语义

■ ANY: 任意一个值

■ ALL: 所有值

■ 需要配合使用的运算符

■ >ANY 大于子查询结果中的某个值

■ >ALL 大于子查询结果中的所有值

<ANY 小于子查询结果中的某个值</p>

■ <ALL 小于子查询结果中的所有值

- >= ANY 大于等于子查询结果中的某个值

• >= ALL 大于等于子查询结果中的所有值

• <= ANY 小于等于子查询结果中的某个值

<= ALL 小于等于子查询结果中的所有值</p>

■ = ANY 等于子查询结果中的某个值

■ =ALL 等于子查询结果中的所有值(通常没有实际意义)

• !=(或<>)ANY 不等于子查询结果中的某个值

• !=(或<>)ALL 不等于子查询结果中的任何一个值

[**例41**] 查询其他系中比信息系任意一个(其中某一个)学生年龄 小的学生姓名和年龄。

SELECT Sname, Sage

FROM Student

WHERE Sage < ANY

(SELECT Sage

FROM Student

WHERE Sdept= 'IS')

AND Sdept <> ' IS '; /* 注意这是父查询块中的条件 */



[例42] 查询其他系中比计算机科学系所有学生年龄小的学生姓名和年龄。

SELECT Sname, Sage

FROM Student

WHERE Sage < ALL

(SELECT Sage

FROM Student

WHERE Sdept= 'CS')

AND Sdept <> ' CS ';



4 带有EXISTS谓词的子查询

- **❖EXISTS**谓词
- **❖NOT EXISTS**谓词
- ❖不同形式的查询间的替换
- ❖相关子查询的效率
- ❖用EXISTS/NOT EXISTS实现全称量词
- ❖用EXISTS/NOT EXISTS实现逻辑蕴函

❖EXISTS谓词

- 存在量词
- 带有EXISTS谓词的子查询不返回任何数据,只产生逻辑真值"true"或逻辑假值"false"。
 - 若内层查询结果非空,则外层的WHERE子句返回真值
 - 若内层查询结果为空,则外层的WHERE子句返回假值
- 由EXISTS引出的子查询,其目标列表达式通常都用*, 因为带EXISTS的子查询只返回真值或假值,给出列名 无实际意义

[例44] 查询所有选修了1号课程的学生姓名。

思路分析:

- 本查询涉及Student和SC关系。
- 在Student中依次取每个元组的Sno值,用此值去检查SC关系。
- 若SC中存在这样的元组,其Sno值等于此Student.Sno值,并且其Cno='1',则取此Student.Sname送入结果关系。

```
SELECT Sname
FROM Student
WHERE EXISTS

(SELECT *
FROM SC /*相关子查询*/
WHERE Sno=Student. Sno AND Cno='1');
```

❖NOT EXISTS谓词

- ■若内层查询结果非空,则外层的WHERE子句返回假值
- 若内层查询结果为空,则外层的WHERE子句返回真值

[例45] 查询没有选修了1号课程的学生姓名。

❖不同形式的查询间的替换

- 一些带EXISTS或NOT EXISTS谓词的子查询不能被其 他形式的子查询等价替换
- 所有带IN谓词、比较运算符、ANY和ALL谓词的子查 询都能用带EXISTS谓词的子查询等价替换。

[例39] 查询与"刘晨"在同一个系学习的学生。

```
SELECT Sno, Sname, Sdept
FROM Student
WHERE Sdept IN
(SELECT Sdept
FROM Student
WHERE Sname='刘晨');
```

```
SELECT Sno, Sname, Sdept
FROM Student S1
WHERE EXISTS
(SELECT *
FROM Student S2
WHERE S2. Sdept = S1. Sdept AND S2. Sname = ' 刘晨';
```

- ❖用EXISTS/NOT EXISTS实现全称量词(难点)
 - SQL语言中没有全称量词∀ (For all)
 - 可以把带有全称量词的谓词转换为等价的带有 存在量词的谓词:

$$(\forall x) P \equiv \neg (\exists x (\neg P))$$

[例45] 查询选修了全部课程的学生姓名。

```
SELECT Sname
FROM Student
WHERE NOT EXISTS
      SELECT *
      FROM Course
      WHERE NOT EXISTS
            (SELECT *
             FROM SC
             WHERE Sno= Student. Sno AND
                   Cno= Course. Cno);
```

❖用EXISTS/NOT EXISTS实现逻辑蕴函(难点)

- SQL语言中没有蕴函(Implication)逻辑运算
- 可以利用谓词演算将逻辑蕴函谓词等价转换为:

$$p \rightarrow q \equiv \neg p \lor q$$

[例47] 查询至少选修了学生95002选修的全部课程的学生号码。

解题思路:

- ■用逻辑蕴函表达:查询学号为x的学生,对所有的课程y,只要95002学生选修了课程y,则x也选修了y。
- ■形式化表示:

用P表示谓词 "学生95002选修了课程y" 用q表示谓词 "学生x选修了课程y" 则上述查询为: $(\forall y)$ $p \rightarrow q$

■ 等价变换:

$$(\forall y)p \rightarrow q \equiv \neg (\exists y (\neg (p \rightarrow q)))$$

$$\equiv \neg (\exists y (\neg (\neg p \lor q))$$

$$\equiv \neg \exists y (p \land \neg q)$$

- 变换后语义:不存在这样的课程y,学生95002选修了y,而学生x没有选。
- 用NOT EXISTS谓词表示:

SELECT DISTINCT Sno

FROM SC SCX

WHERE NOT EXISTS

(SELECT *

FROM SC SCY

WHERE SCY.Sno = '95002 ' AND

NOT EXISTS

(SELECT *

FROM SC SCZ

WHERE SCZ.Sno=SCX.Sno AND

SCZ.Cno=SCY.Cno));

"数据查询"内容概要

- ◆查询语句概述
- ❖单表查询
- ❖连接查询
- ❖嵌套查询
- ❖集合查询
 - 并操作(UNION)
 - 交操作(INTERSECT)
 - 差操作(MINUS)

集合查询一并操作(UNION)

❖形式

<查询块> UNION <查询块>

- 参加UNION操作的各结果表的列数必须相同;对应项的数据类型 也必须相同
- UNION: 将多个查询结果合并起来时,系统自动去掉重复元组。
- UNION ALL: 将多个查询结果合并起来时,保留重复元组

[例42] 查询计算机科学系的学生及年龄不大于19岁的学生。

方法一:

方法二:

SELECT *

FROM Student

WHERE Sdept= 'CS'

UNION

SELECT *

FROM Student

WHERE Sage<=19;

SELECT DISTINCT *

FROM Student

WHERE Sdept= 'CS' OR Sage<=19;

Ⅲ 结果 ┃ 請 消息 ┃					
	Sno	Sname	Ssex	Sage	Sdept
1	200215121	李勇	男	20	CS
2	200215122	刘晨	女	19	CS
3	200215123	王敏	女	18	MA
4	200515125	张立	男	19	IS

集合查询一交操作

[例50] 查询计算机科学系的学生与年龄不大于19岁的学生的交集(INTERSECT)。

SELECT *

FROM Student

WHERE Sdept='CS'

INTERSECT

SELECT *

FROM Student

WHERE Sage<=19

请使用连接查询写出等价脚本

SELECT *

FROM Student

WHERE Sdept= 'CS' AND

Sage<=19;

集合查询一差操作

[例52] 查询计算机科学系的学生与年龄不大于19岁的学生的差集。

SELECT *

FROM Student

WHERE Sdept='CS'

EXCEPT

SELECT *

FROM Student

WHERE Sage <=19;

请使用连接查询写出等价脚本

SELECT *

FROM Student

WHERE Sdept= 'CS' AND

Sage>19;

对集合操作结果的排序

- ❖ ORDER BY子句只能用于对最终查询结果排序,不能对中 间结果排序
- ❖ 任何情况下,ORDER BY子句只能出现在最后
- ❖ 对集合操作结果排序时,ORDER BY子旬中用数字指定排 序属性

SELECT *
FROM Student
WHERE Sdept= 'CS'
ORDER BY Sno
UNION
SELECT *
FROM Student
WHERE Sage<=19
ORDER BY Sno;

SELECT *
FROM Student
WHERE Sdept= 'CS'
UNION
SELECT *
FROM Student
WHERE Sage<=19
ORDER BY Sno;



SELECT小结

```
SELECT [ALL|DISTINCT]
 <目标列表达式>[别名][, <目标列表达式>[别名]]...
FROM <表名或视图名>[别名]
     [, <表名或视图名>[别名]]...
[WHERE <条件表达式>]
[GROUP BY <列名1>[, <列名1'>] ...
[HAVING <条件表达式>]]
[ORDER BY <列名2> [ASC|DESC]
     [, <列名2'>[ASC|DESC]]...];
```

❖ 整条语句的含义:

- 根据WHERE子句的条件表达式,从FROM子句指定的基本表或 视图中找出满足条件的元组,再按SELECT子句中的目标列表达 式,选出元组中的属性值形成结果表。
- 如果有GROUP子句,则将结果按<列名1>的值进行分组,该属性列值相等的元组为一个组,每个组产生结果表中的一条记录,通常会在每组中使用集函数。如果GROUP子句带HAVING短语,则只有满足指定条件的组才输出。如果有ORDER子句,则结果表还要按<列名2>的值的升序或降序排列。

Q & A

- ❖如何给列起别名,如何写计 算列?
- ❖如何去掉重复行?
- ◆多个字段排序的顺序是怎样的?
- ❖Where和group by都是选择 语句,他们的区别是什么?



这次课我们学到了…

❖ 使用SELECT语句:

- 使用表别名和列别名;
- 查询满足一定条件的元组;
- 查询某些属性的值;
- 通过在WHERE子句中放入连接条件,进行多表连接查询;
- 利用DISTINCT去掉查询结果中的重复行;
- 利用GROUP BY进行分组统计
- 利用ORDER对查询结果按要求排序;

❖ 复杂查询

- 嵌套查询
- 集合查询

休息…



不贤而 『见贤思齐馬 肉 自省也。 见