

数据库技术与应用

—— 数据查询

讲师：孙煦雪

浙江传媒学院
COMMUNICATION
UNIVERSITY
OF ZHEJIANG



课程回顾

- 数据库的创建与维护
- 数据库备份与恢复
- 表的创建与维护
- 表中数据的维护

内容提要

- 基本查询
- 嵌套查询
- 连接查询
- 集合查询

<https://www.bilibili.com/video/BV1Ev41167CX/>

数据查询

查询是数据库管理的核心操作，SQL语言提供了
SELECT语句进行单表查询、嵌套查询、多表查询、
集合查询

数据查询——SELECT 语句的一般格式

SELECT

目标列表表达式

FROM

表名/视图名

WHERE

条件表达式

GROUP BY

列名

HAVING

条件表达式

ORDER BY

列名

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分组

ORDER BY

按指定字段值，排序查询结果

示例数据库

学生选课系统

□ 学生表

studinfo(sno, sname, ssex, sage, sdept)

□ 课程表

course(cno, cname, cpno, credit)

□ 学生选课表

sc(sno, cno, score)

内容提要

➤ 单表查询

➤ 基本的 SELECT 语句：投影查询

➤ 基本的 SELECT 语句：选择查询

➤ order by 子句：排序查询

➤ 聚集函数

➤ group by 子句：分组查询

➤ 嵌套查询

➤ 连接查询

➤ 集合查询

投影查询

SELECT 目标列表达式

FROM 表名或视图名

目标列表达式可以是：属性名、算术表达式、字符串常量、函数等

投影

Table 1

查询全体学生的学号、姓名

SELECT sno, sname

FROM studinfo

$\pi_{\text{sno,sname}}(\text{studinfo})$

查询全体学生的详细记录

SELECT *

FROM studinfo

使用函数、运算符及列的别名

- 查询全体学生的姓名、出生年份

SELECT sname, 2022-sage AS BirthYear

FROM studinfo



列别名

- 查询学生学号，课程号，原始成绩，新成绩

SELECT sno, cno, score, sqrt(score)*10 "curve grade"

FROM sc



列别名

内容提要

➤ 单表查询

➤ 基本的 SELECT 语句：投影查询

➤ 基本的 SELECT 语句：选择查询

➤ order by 子句：排序查询

➤ 聚集函数

➤ group by 子句：分组查询

➤ 嵌套查询

➤ 连接查询

➤ 集合查询

选择查询

SELECT **DISTINCT** 目标列表表达式

FROM 表名或视图名

WHERE 条件表达式

选择

Table 1

◆ 取消重复行

查询哪些课程被选修

SELECT **DISTINCT** cno

FROM sc

◆ 查询满足条件的元组：WHERE 子句

查询满足条件的元组

WHERE 子句中用于查询的条件表达式使用的运算符

比较	=、>、<、>=、<=、 <>、!=、!>、!<
确定范围	BETWEEN...AND... ,可加NOT
确定集合	IN ,可加NOT
字符匹配	LIKE , 可加NOT
空值判断	IS NULL, 可加NOT
逻辑运算符	NOT, AND, OR

选择查询举例—比较

比较运算符（=、>、<、>=、<=、<>）

查询CS系所有学生的姓名

```
select  sname  
from    studinfo  
where   sdept='CS'
```

↔ $\pi_{\text{sname}}(\sigma_{\text{sdept} = \text{'CS'}}(\text{studinfo}))$

选择查询举例—比较

- 查询所有年龄在20岁以下的学生的姓名及其年龄

```
select    sname,sage
```

```
from      studinfo
```

```
where     sage<20
```

- 查询成绩不及格的学生号

```
select    distinct    sno
```

```
from      sc
```

```
where     score<60
```

选择查询举例—确定范围BETWEEN AND

BETWEEN...AND...

雇员信息表 emp(empno, ename, deptno, sal, job, hiredate)

查询工资在[1000,1500]之间的雇员的姓名和薪资

```
SELECT      ename, sal
FROM        emp
WHERE       sal BETWEEN 1000 AND 1500;
```

ENAME	SAL
MARTIN	1250
TURNER	1500
WARD	1250
ADAMS	1100
MILLER	1300

范围的下限 (即低值) 范围的上限 (即高值)

选择查询举例—确定范围BETWEEN AND

查询年龄在[20, 23]岁之间的学生的姓名、
系别和年龄

```
SELECT  sname,sdept,sage
```

```
FROM    studinfo
```

```
WHERE   sage  BETWEEN  20  AND  23
```

选择查询举例—确定集合IN

IN 值表, 或 NOT IN 值表

值表（取值列表）：用逗号分隔的一组取值

查询信息系（IS）、机械系（ME）和计算机科学系（CS）学生的姓名和性别

```
select  sname,ssex
```

```
from    studinfo
```

```
where   sdept  in ('CS', 'IS', 'ME')
```

选择查询举例—字符串匹配LIKE

LIKE 或 NOT LIKE ‘匹配串’ escape 换码字符

□ 通配符

- ◆ % 代表 任意长度（或长度为0）的字符串
- ◆ _ 代表 任意单个字符

```
SELECT  ename
FROM    emp
WHERE   ename LIKE '_A%';
```

ENAME

JAMES

WARD

选择查询举例—字符串匹配LIKE

- ❑ 找出05级学生的姓名和年龄

```
select  sname,sage  from  studinfo  
where  sno  like  '05%'
```

- ❑ 找出姓‘丁’的单名学生的学号、姓名

```
select  sno,sname  from  studinfo  
where  sname  like  '丁_'
```

- ❑ 使用 **ESCAPE 进行转义** 后，可以查找 “%” 或 “_”，
例如找出课程名为“DB_Design”的课号和学分

```
select  cno,credit  from  course  
where  cname  like  'DB\_Design'  escape  '\'
```

选择查询举例—涉及空值

IS NULL 或 IS NOT NULL

- ❑ 找出选修课成绩尚未登记的学生学号及其选修课程号

```
select sno,cno
```

```
from sc
```

```
where score is NULL
```

- ❑ 找出已登记选修课成绩的学生学号及其选修课程号

```
select sno,cno
```

```
from sc
```

```
where score is NOT NULL
```

选择查询举例—多重条件查询

□ 逻辑运算符联结多个查询条件

AND OR

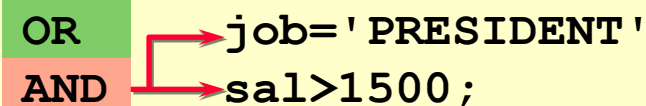
□ 优先顺序

比较运算符 > 逻辑运算符 NOT > 逻辑运算符 AND
> 逻辑运算符 OR

□ 可以用括号改变优先级

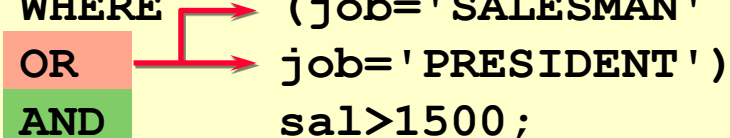
选择查询举例—多重条件查询

```
SELECT  ename, job, sal
FROM    emp
WHERE   job='SALESMAN'
OR      job='PRESIDENT'
AND     sal>1500;
```



ENAME	JOB	SAL
KING	PRESIDENT	5000
MARTIN	SALESMAN	1250
ALLEN	SALESMAN	1600
TURNER	SALESMAN	1500
WARD	SALESMAN	1250

```
SELECT      ename, job, sal
FROM        emp
WHERE       (job='SALESMAN'
OR          job='PRESIDENT')
AND         sal>1500;
```



ENAME	JOB	SAL
KING	PRESIDENT	5000
ALLEN	SALESMAN	1600

内容提要

➤ 单表查询

➤ 基本的 SELECT 语句：投影查询

➤ 基本的 SELECT 语句：选择查询

➤ order by 子句：排序查询

➤ 聚集函数

➤ group by 子句：分组查询

➤ 嵌套查询

➤ 连接查询

➤ 集合查询

排序查询

□ 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

□ 当排序属性列含空值时

- ◆ ASC：排序属性列为空值的元组最后显示
- ◆ DESC：排序属性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

排序查询举例

按结果列的别名排序

```
SELECT  empno, ename, sal*12 annsal  
FROM    emp  
ORDER BY annsal;
```

EMPNO	ENAME	ANNSAL
7369	SMITH	9600
7900	JAMES	11400
7876	ADAMS	13200
7654	MARTIN	15000
7521	WARD	15000
7934	MILLER	15600
7844	TURNER	18000
...		

按照成绩由高到低（降序），查询选修了1号课程的学生学号和成绩

```
select      sno,score
from        sc
where       cno='1'
order by    score desc
```

内容提要

➤ 单表查询

➤ 基本的 SELECT 语句：投影查询

➤ 基本的 SELECT 语句：选择查询

➤ order by 子句：排序查询

➤ 聚集函数

➤ group by 子句：分组查询

➤ 嵌套查询

➤ 连接查询

➤ 集合查询

使用聚集函数

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

聚集函数 (**DISTINCT** 或 **ALL** 参数) 在 Select 子句中使用

- ❑ **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

使用聚集函数举例

求20号部门的员工的最高工资
**select max(sal) from EMP
where DEPTNO='20'**

EMP

DEPTNO	SAL
10	2450
10	5000
10	1300
20	800
20	1100
20	3000
20	3000
20	2975
30	1600
30	2850
30	1250
30	950
30	1500
30	1250

MAX (SAL)
3000

使用聚集函数举例

AVG 和 SUM 适用于**数值型**数据

MIN 和 MAX 适用于**不同类型**的数据

```
SELECT MIN(hiredate), MAX(hiredate)
FROM emp;
```

MIN (HIRED	MAX (HIRED
-----	-----
12-JAN-98	28-SEP-13

```
SELECT COUNT (*)
FROM emp
WHERE deptno = 30;
```

COUNT (*)

6

NULL值会被忽略

使用聚集函数举例

□ 求总的学生数

```
SELECT COUNT(*) FROM studinfo
```

□ 求4号课程的平均成绩

```
SELECT AVG(score) FROM sc  
WHERE cno='4'
```


内容提要

➤ 单表查询

➤ 基本的 SELECT 语句：投影查询

➤ 基本的 SELECT 语句：选择查询

➤ order by 子句：排序查询

➤ 聚集函数

➤ group by 子句：分组查询

➤ 嵌套查询

➤ 连接查询

➤ 集合查询

分组查询

- 使用 **GROUP BY** 子句分组
- 分组方法：按指定的一列或多列取值相等分组
- 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

分组查询

- 使用GROUP BY子句后，SELECT子句的列名中只能出现分组属性和聚集函数
- 可以使用HAVING短语筛选最终输出结果

分组查询—按单列分组

雇员信息表 emp(empno, ename, deptno, sal, job, hiredate)

emp表中每一部门的平均工资

```
SELECT      deptno, AVG(sal)
FROM        emp
GROUP BY    deptno
```

EMP

DEPTNO	SAL
10	2450
10	5000
10	1300
20	800
20	1100
20	3000
20	3000
20	2975
30	1600
30	2850
30	1250
30	950
30	1500
30	1250

2916.6667

2175

1566.6667

DEPTNO	AVG (SAL)
10	2916.6667
20	2175
30	1566.6667

分组查询—按多列分组

emp表中不同部门不同职位的
工资总和

EMP

DEPTNO	JOB	SAL
10	MANAGER	2450
10	PRESIDENT	5000
10	CLERK	1300
20	CLERK	800
20	CLERK	1100
20	ANALYST	3000
20	ANALYST	3000
20	MANAGER	2975
30	SALESMAN	1600
30	MANAGER	2850
30	SALESMAN	1250
30	CLERK	950
30	SALESMAN	1500
30	SALESMAN	1250

SELECT
FROM
GROUP BY

deptno, job, sum(sal)
emp
deptno, job

DEPTNO	JOB	SUM(SAL)
10	CLERK	1300
10	MANAGER	2450
10	PRESIDENT	5000
20	ANALYST	6000
20	CLERK	1900
20	MANAGER	2975
30	CLERK	950
30	MANAGER	2850
30	SALESMAN	5600

分组查询—结果筛选

□ 使用 HAVING 短语对分组结果进行筛选

SELECT 目标列表达式, 集函数

FROM 表名

WHERE 条件表达式

GROUP BY 分组条件表达式

HAVING 条件表达式

ORDER BY 列名;

分组查询—结果筛选

□ WHERE 和 HAVING 子句区别

◆ 作用对象

- WHERE 子句：基表或视图，选择满足条件的元组，
不能选择分组
- HAVING 短语：分组，从中选择满足条件的分组

◆ 使用方法

- WHERE 子句：条件表达式不能有集函数
- HAVING 短语：不能单独使用，只能与 GROUP BY 子句配合使用

分组查询—结果筛选

每一部门最高工资大于 2900

```
SELECT
FROM
GROUP BY
HAVING
deptno, max(sal)
emp
deptno
max(sal)>2900
```

EMP

DEPTNO	SAL
10	2450
10	5000
10	1300
20	800
20	1100
20	3000
20	3000
20	2975
30	1600
30	2850
30	1250
30	950
30	1500
30	1250

5000

3000

2850

DEPTNO	MAX (SAL)
10	5000
20	3000

分组查询举例

学生选课系统

□ 求每门课程的选修人数

```
SELECT      cno,COUNT(sno)
FROM        sc
GROUP BY    cno
```

□ 查询选修人数超过2人（不含2人）的课程号

```
SELECT      cno
FROM        sc
GROUP BY    cno
HAVING      COUNT(sno) >2
```

单表查询小结

- 基本的 SELECT 语句：投影查询
- 基本的 SELECT 语句：选择查询
- order by 子句：排序查询
- 聚集函数
- group by 子句：分组查询

内容提要

- 单表查询
- 嵌套查询
 - 带有 **IN** 谓词的子查询
 - 带有**比较运算符**的子查询
 - 带有 **ANY** 或 **ALL** 的子查询
 - 带有 **EXISTS** 谓词的子查询
- 连接查询
- 集合查询

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**嵌套查询**

“谁的工资比 Jones 高?”

主查询



“哪一个雇员的工资比 Jones 高?”

子查询



“Jones 的工资是多少?”

- 允许多层嵌套
- 子查询**不能**使用 **ORDER BY** 子句

内容提要

- 单表查询
- 嵌套查询
 - 带有 **IN** 谓词的子查询
 - 带有比较运算符的子查询
 - 带有 ANY 或 ALL 的子查询
 - 带有 EXISTS 谓词的子查询
- 连接查询
- 集合查询

带有IN谓词的子查询

查询与“李四”在同一个系的学生

```
SELECT  sno,sname,sdept FROM  studinfo
WHERE   sdept IN
        (SELECT  sdept
         FROM    studinfo
         WHERE   sname='李四')
```

第一步：确定“李四”所在系名

```
SELECT sdept
FROM   studinfo
WHERE  sname= '李四'
```

sdept
CS

第二步：查找所有CS系的学生

```
SELECT  sno, sname, sdept
FROM    studinfo
WHERE   sdept = 'CS'
```

sno	sname	sdept
07012	李四	CS
07020	丁五	CS

IN

多层嵌套

查询选修了课程“信息系统”的学生学号和姓名

SELECT sno,sname

③ 在studinfo关系中取出
sno和sname

FROM studinfo

WHERE sno IN

(SELECT sno FROM sc

② 在sc关系中找出选修
了3号课程的学生学号

WHERE cno IN

(SELECT cno FROM course
WHERE cname='信息系统'))

① 在course关系找出
“信息系统”的课程号，
结果为3号

内容提要

- 单表查询
- 嵌套查询
 - 带有 IN 谓词的子查询
 - 带有比较运算符的子查询
 - 带有 ANY 或 ALL 的子查询
 - 带有 EXISTS 谓词的子查询
- 连接查询
- 集合查询

带有比较运算符的子查询

- ❑ 当确切知道内层查询返回的是单值时，可以用比较运算符（>，<，=，>=，<=，!=或<>）
- ❑ 子查询一定要跟在比较符之后
- ❑ 与 ANY 或 ALL谓词配合使用

假设一个学生只可能在一个系学习，并且必须属于一个系，
可以用 = 代替 IN

```
SELECT sno,sname,sdept FROM studinfo
```

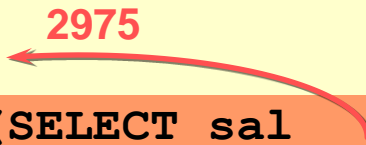
```
WHERE sdept =
```

```
(SELECT sdept
```

```
FROM studinfo
```

```
WHERE sname='李四' )
```

带比较运算符的嵌套查询实现

```
SELECT ename
FROM    emp
WHERE   sal > 
          (SELECT sal
           FROM    emp
           WHERE   empno=7566) ;
```

ENAME

KING

FORD

SCOTT

内容提要

- 单表查询
- 嵌套查询
 - 带有 IN 谓词的子查询
 - 带有比较运算符的子查询
 - 带有 ANY 或 ALL 的子查询
 - 带有 EXISTS 谓词的子查询
- 连接查询
- 集合查询

带有ANY或ALL的子查询

□ 谓词语义

◆ ANY: 任一个值 (某个值)

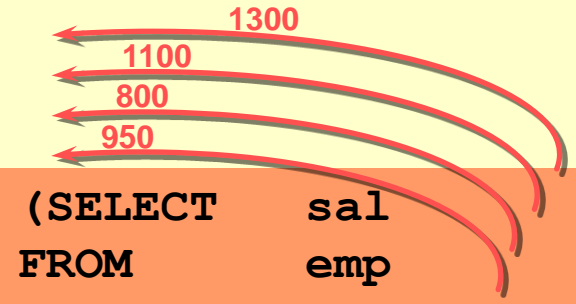
◆ ALL: 所有值

□ 需要配合比较运算符

$>(<)$ ANY	大于(小于)子查询中 某个值
$\geq(\leq)$ ANY	大于等于(小于等于)子查询中 某个值
$=(\neq)$ ANY	等于(不等于)子查询中 某个值
$>(<)$ ALL	大于(小于)子查询中的 所有值
$\geq(\leq)$ ALL	大于等于(小于等于)子查询中的 所有值
$=(\neq)$ ALL	等于(不等于)子查询中的 所有值 (任何值)

使用ANY的嵌套查询实现

```
SELECT empno, ename, job
FROM emp
WHERE sal < ANY
      (SELECT sal
       FROM emp
       WHERE job = 'CLERK')
AND job <> 'CLERK';
```



EMPNO	ENAME	JOB
7654	MARTIN	SALESMAN
7521	WARD	SALESMAN

使用ALL的嵌套查询实现

```
SQL> SELECT empno, ename, job
       FROM emp
       WHERE sal > ALL
              (SELECT avg(sal)
               FROM emp
               GROUP BY deptno);
```

The diagram illustrates the execution of the SQL query. The 'ALL' keyword in the WHERE clause is compared against the results of the subquery. The subquery results are 1566.6667, 2175, and 2916.6667. Red arrows point from the 'ALL' keyword to these values, indicating that the salary of each employee must be greater than all of these values.

EMPNO	ENAME	JOB
7839	KING	PRESIDENT
7566	JONES	MANAGER
7902	FORD	ANALYST
7788	SCOTT	ANALYST

ANY、ALL 与集函数、IN 的等价转换关系

	=	<>	<	<=	>	>=
ANY	IN		<MAX	<= MAX	>MIN	>= MIN
ALL		NOT IN	<MIN	<= MIN	>MAX	>= MAX

< ALL 小于子查询结果中的**所有**值 < MIN

< ANY 小于子查询结果中的**某个**值 < MAX

内容提要

- 单表查询
- 嵌套查询
 - 带有 IN 谓词的子查询
 - 带有比较运算符的子查询
 - 带有 ANY 或 ALL 的子查询
 - 带有 EXISTS 谓词的子查询
- 连接查询
- 集合查询

带有 EXISTS 谓词的子查询

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

带有 NOT EXISTS 谓词的子查询反之

带有EXISTS谓词的子查询

查询所有选修了1号课程的学生姓名

SELECT sname

FROM studinfo

WHERE EXISTS

 (SELECT * FROM sc

 WHERE sno=studinfo.sno AND cno= '1')

相关子查询

带有EXISTS谓词的子查询

参考：

<https://www.bilibili.com/video/BV1Xh4y1r7q9/>

关系代数 除法实现 \longleftrightarrow 双嵌套 Not Exists

```
select distinct R.X from R R1
where not exists
(
  select S.Y from S
  where not exists
  (
    select * from R R2
    where R2.X=R1.X and R2.Y=S.Y
  )
)
```

嵌套查询小结

- 带有 IN 谓词的子查询
- 带有比较运算符的子查询
- 带有 ANY 或 ALL 的子查询
- 带有 EXISTS 谓词的子查询

内容提要

- 单表查询
- 嵌套查询
- 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
- 集合查询

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
- Select 语句实现连接操作

```
SELECT 表名1.目标列, 表名2.目标列  
FROM   表名1, 表名2  
WHERE  表名1.列 连接运算符 表名2.列;
```

连接运算符：=、>、<、>=、<=、!=

```
WHERE 表名1.列 BETWEEN 表名2.列1 AND 表名2.列2
```

连接字段：连接谓词中的列名，其数据类型必须是可比的

内容提要

- 单表查询
- 嵌套查询
- 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
- 集合查询

等值与非等值连接

- 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

查询每个学生及其选修课程的情况

```
SELECT studinfo.*,sc.*  
FROM studinfo, sc  
WHERE studinfo.sno = sc.sno
```


等值连接的实现

EMP

EMPNO	ENAME	DEPTNO
-----	-----	-----
7839	KING	10
7698	BLAKE	30
7782	CLARK	10
7566	JONES	20
7654	MARTIN	30
7499	ALLEN	30
7844	TURNER	30
7900	JAMES	30
7521	WARD	30
7902	FORD	20
7369	SMITH	20
...		
14 rows selected.		

外键

DEPT

DEPTNO	DNAME	LOC
-----	-----	-----
10	ACCOUNTING	NEW YORK
30	SALES	CHICAGO
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
30	SALES	CHICAGO
30	SALES	CHICAGO
30	SALES	CHICAGO
30	SALES	CHICAGO
20	RESEARCH	DALLAS
20	RESEARCH	DALLAS
...		
14 rows selected.		

主键

等值连接可视化

EMP

EMPNO	ENAME	...	DEPTNO
-----	-----	...	-----
7839	KING	...	10
7698	BLAKE	...	30
...			
7934	MILLER	...	10

DEPT

DEPTNO	DNAME	LOC
-----	-----	-----
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON



EMPNO	DEPTNO	LOC
-----	-----	-----
7839	10	NEW YORK
7698	30	CHICAGO
7782	10	NEW YORK
7566	20	DALLAS
7654	30	CHICAGO
7499	30	CHICAGO
...		
14 rows selected.		

```
SELECT  EMP.EMPNO,  
        DEPT.DEPTNO,DEPT.LOC  
FROM    EMP,DEPT  
WHERE  
        EMP.DEPTNO=DEPT.DEPTNO
```

非等值连接

EMP

EMPNO	ENAME	SAL
7839	KING	5000
7698	BLAKE	2850
7782	CLARK	2450
7566	JONES	2975
7654	MARTIN	1250
7499	ALLEN	1600
7844	TURNER	1500
7900	JAMES	950
...		
14 rows selected.		

SALGRADE

GRADE	LOSAL	HISAL
1	700	1200
2	1201	1400
3	1401	2000
4	2001	3000
5	3001	9999

➡ EMP 表中的工资数额在
SALGRADE 表的
LOSAL 和 HISAL 之间

非等值连接的实现

```
SELECT    e.ename, e.sal, s.grade
FROM      emp e, salgrade s
WHERE     e.sal
BETWEEN   s.losal AND s.hisal;
```

ENAME	SAL	GRADE
-----	-----	-----
JAMES	950	1
SMITH	800	1
ADAMS	1100	1
...		

14 rows selected.

内容提要

- 单表查询
- 嵌套查询
- 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
- 集合查询

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

查询每门课的间接先修课（即先修课的先修课）

```
SELECT first.cno,second.cjno  
  
FROM   course first,course second  
  
WHERE  first.cjno=second.cno
```

first. cno	second. cjno
1	2
3	5
5	
6	2

内容提要

- 单表查询
- 嵌套查询
- 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
- 集合查询

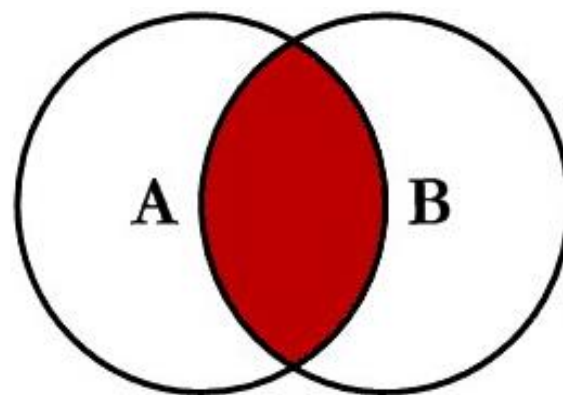
补充：内连接*

查询所有的学生信息及其选修课程情况

```
SELECT studinfo.*,sc.*  
FROM studinfo,sc  
WHERE studinfo.sno=sc.sno
```



```
SELECT studinfo.*,sc.*  
FROM studinfo JOIN sc  
ON studinfo.sno=sc.sno
```



```
SELECT <select_list>  
FROM Table_A A  
INNER JOIN Table_B B  
ON A.Key = B.Key
```


内连接*

关系 Loan

Loanno	Bname	amount
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

关系 Borrow

custna	Loanno
Jones	L-170
Smith	L-230
Mary	L-155

select * from Loan inner join Borrow
on Loan.Loanno=Borrow.Loanno

Loanno	Bname	amount	custna	Loanno
L-170	Downtown	3000	Jones	L-170
L-230	Redwood	4000	Smith	L-230

外连接*

□ 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

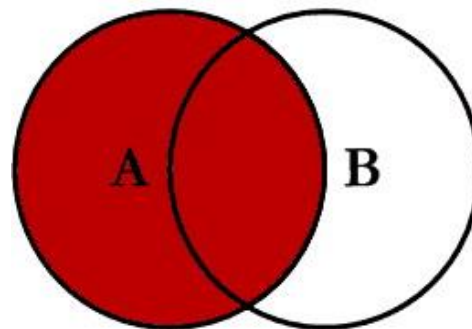
□ 外连接可分为

- ◆ 左连接 (LEFT JOIN) 或左外连接 (LEFT OUTER JOIN)
- ◆ 右连接 (RIGHT JOIN) 或右外连接 (RIGHT OUTER JOIN)
- ◆ 全连接 (FULL JOIN) 或全外连接 (FULL OUTER JOIN)

```
SELECT 表名1.目标列 , 表名2.目标列  
FROM   表1   left outer join  
        或 right outer join  
        或 full outer join 表2  
ON     表名1.列 = 表名2.列
```

左外连接*

```
SELECT <select_list>
FROM Table_A A
LEFT JOIN Table_B B
ON A.Key = B.Key
```



关系 Loan

Loanno	Bname	amount
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

关系 Borrow

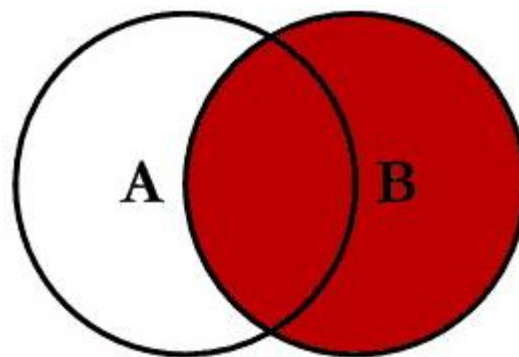
custna	Loanno
Jones	L-170
Smith	L-230
Mary	L-155

Loan left outer join Borrow on Loan.Loanno=Borrow.Loanno

Loanno	Bname	amount	custna	Loanno
L-170	Downtown	3000	Jones	L-170
L-230	Redwood	4000	Smith	L-230
L-260	Perryridge	1700	NULL	NULL

右外连接*

```
SELECT <select_list>
FROM Table_A A
RIGHT JOIN Table_B B
ON A.Key = B.Key
```



关系 Loan

Loanno	Bname	amount
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

关系 Borrow

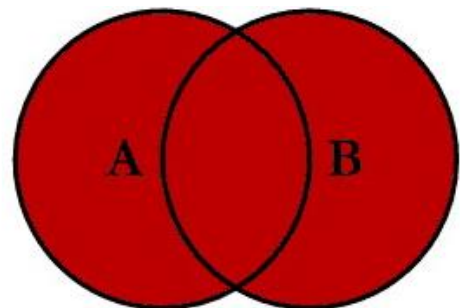
custna	Loanno
Jones	L-170
Smith	L-230
Mary	L-155

Loan right outer join Borrow on Loan.Loanno=Borrow.Loanno

Loanno	Bname	amount	custna	Loanno
L-170	Downtown	3000	Jones	L-170
L-230	Redwood	4000	Smith	L-230
L-155	NULL	NULL	Mary	L-155

全外连接*

```
SELECT <select_list>
FROM Table_A A
FULL OUTER JOIN Table_B B
ON A.Key = B.Key
```



关系 Loan

Loanno	Bname	amount
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

关系 Borrow

custna	Loanno
Jones	L-170
Smith	L-230
Mary	L-155

Loan full outer join Borrow on Loan.Loanno=Borrow.Loanno

Loanno	Bname	amount	custna
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-260	Perryridge	1700	NULL
L-155	NULL	NULL	Mary

内容提要

- 单表查询
- 嵌套查询
- 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
- 集合查询

复合条件连接

当 WHERE 子句中出现多个条件的连接操作时，称为复合条件连接

查询每个学生的学号、姓名、选修的课程名及成绩

```
SELECT studinfo.sno,sname,course.cname,sc.score
```

```
FROM studinfo,sc,course
```

--多表连接

```
WHERE studinfo.sno=sc.sno
```

```
AND sc.cno=course.cno
```

内容提要

- 单表查询
- 嵌套查询
- 连接查询
- 集合查询

集合查询

- ❑ Select 语句的查询结果是元组的集合，可以进行集合操作
- ❑ 标准 SQL 直接支持的集合操作种类
 - ◆ 并操作(UNION)
- ❑ 一般商用数据库支持的集合操作种类
 - ◆ 并操作(UNION)
 - ◆ 交操作(INTERSECT)
 - ◆ 差操作(MINUS)

并操作

查询块

UNION

查询块

- UNION：将多个查询结果合并起来时，系统自动去掉重复元组
- 参加 UNION 操作的各结果表必须
 - ◆ 列数相同
 - ◆ 对应项数据类型相同

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，只能出现在最后**
- ❑ 对集合操作结果排序时，ORDER BY子句中可以用数字指定排序属性

?

```
SELECT      *  
FROM        studinfo  
WHERE       sdept= 'CS'  
ORDER BY   Sdept  
UNION  
SELECT      *  
FROM        studinfo  
WHERE       sage<=19  
ORDER BY    Sdept
```

1

连接和集合查询小结

□ 连接查询

- 等值与非等值连接查询
- 自身连接
- 外连接：左外连接，右外连接，全外连接
- 复合条件连接

□ 集合查询

课程小结

➤ 单表查询

投影查询，选择查询，order by 子句的排序查询，聚集函数和 group by 子句的分组查询

➤ 嵌套查询

带 IN 谓词的子查询、带比较运算符的子查询、带 ANY 或 ALL 的子查询、带 EXISTS 谓词的子查询

➤ 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

➤ 集合查询

Q & A