数据库原理与技术

浙江農林大学
ZHEJIANG A&F UNIVERSITY

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

浙江农林大学数学与计算机科学学院

第三章 关系数据库标准语言SQL

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基于派生表的查询

3.4.5 Select语句的一般形式

3.4.2 连接查询

- 连接查询：同时涉及两个以上的表的查询
- 连接条件或连接谓词：用来连接两个表的条件

  一般格式：

  - [<表名1>.]<列名1> <比较运算符> [<表名2>.]<列名2>
  - [<表名1>.]<列名1> BETWEEN [<表名2>.]<列名2> AND [<表名2>.]<列名3>
- 连接字段：连接谓词中的列名称
  - 连接条件中的各连接字段类型必须是可比的，但名字不必相同

连接查询（续）

1.等值与非等值连接查询

2.自身连接

3.外连接

4.多表连接

1. 等值与非等值连接查询

- 等值连接：连接运算符为=

[例] 查询每个学生及其选修课程的情况

等值与非等值连接查询（续）

- 一条SQL语句可以同时完成选择和连接查询，这时WHERE子句是由连接谓词和选择谓词组成的复合条件。

[例]查询选修'B3503021'号课程且成绩在90分以上的所有学生的学号和姓名。

- 执行过程:
  - 先从SC中挑选出cnum= 'B3503021'并且score>90的元组形成一个中间关系
  - 再和S中满足连接条件的元组进行连接得到最终的结果关系

连接查询（续）

1.等值与非等值连接查询

2.自身连接

3.外连接

4.多表连接

2. 自身连接

- 自身连接：一个表与其自己进行连接
- 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]列出工资高于杨平的所有教师的姓名、职称和工资

连接查询（续）

1.等值与非等值连接查询

2.自身连接

3.外连接

4.多表连接

3. 外连接

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
- 左外连接(LEFT JOIN)
  - 列出左边关系中所有的元组
- 右外连接(RIGHT JOIN)
  - 列出右边关系中所有的元组
- 全连接(FULL JOIN)
  - 列出左右边关系中所有的元组

外连接（续）

[例]

```
SELECT S.*,SC.*
FROM  S LEFT OUT JOIN SC ON
(S.SNUM=SC.SNUM);
```

连接查询（续）

1.等值与非等值连接查询

2.自身连接

3.外连接

4.多表连接

4. 多表连接

- 多表连接：两个以上的表进行连接

[例]查询每个学生的学号、姓名、选修的课程名及成绩

[例]查询选修了‘数据库原理与技术’的学生姓名

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基于派生表的查询

3.4.5 Select语句的一般形式

嵌套查询（续）

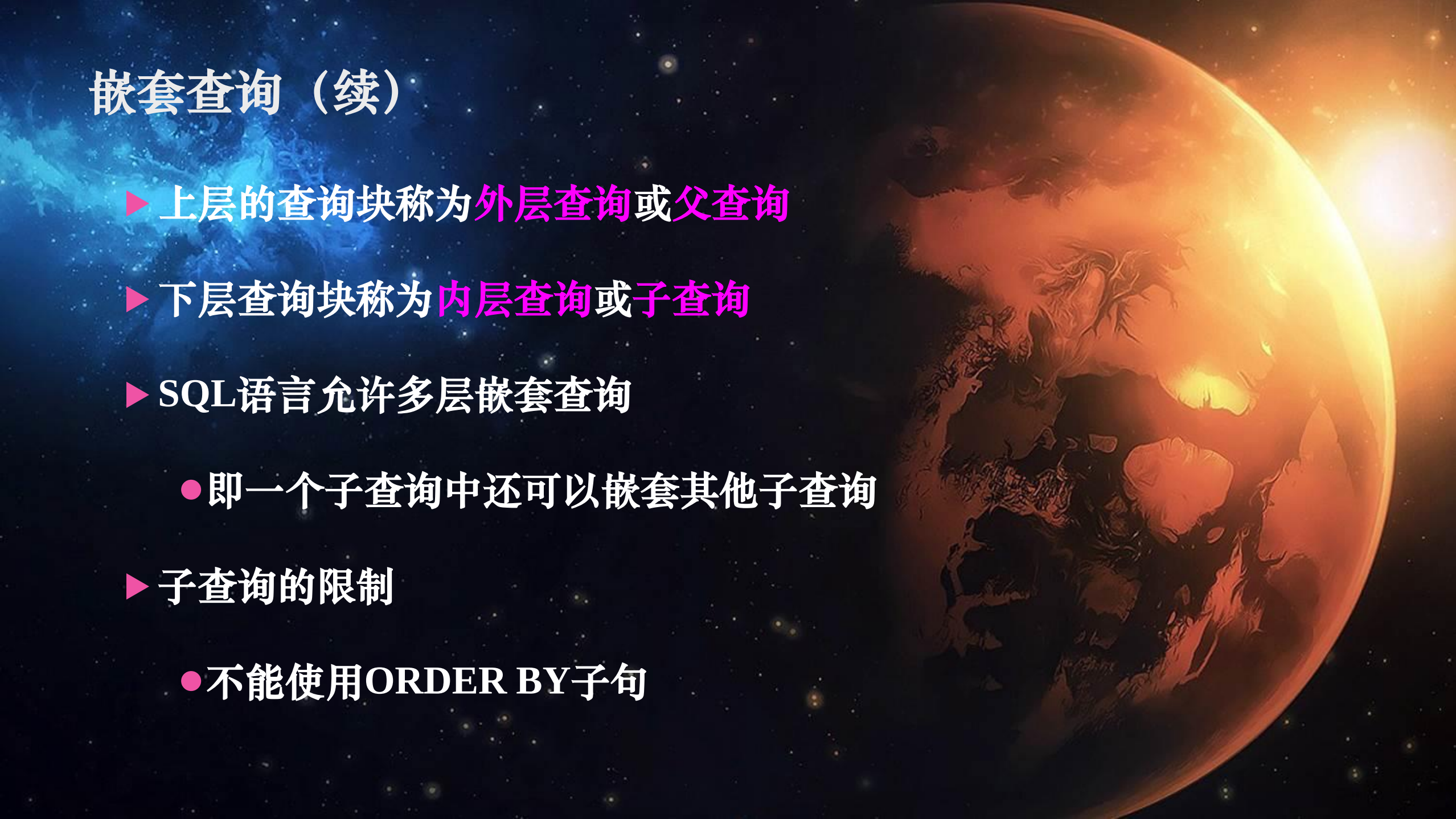
- 嵌套查询概述
  - 一个SELECT-FROM-WHERE语句称为一个查询块
  - 将一个查询块嵌套在另一个查询块的WHERE子句或HAVING短语的条件中的查询称为嵌套查询

```
SELECT Sname                          /*外层查询/父查询*/
FROM S
WHERE Snum IN
            ( SELECT Snum       /*内层查询/子查询*/
              FROM SC
              WHERE Cnum='B3503021');
```

查询选修了‘B3503021’号课程的学生姓名

嵌套查询（续）

- 上层的查询块称为外层查询或父查询
- 下层查询块称为内层查询或子查询
- SQL语言允许多层嵌套查询
  - 即一个子查询中还可以嵌套其他子查询
- 子查询的限制
  - 不能使用ORDER BY子句



嵌套查询求解方法

- 不相关子查询：

  子查询的查询条件不依赖于父查询

  - 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。

嵌套查询求解方法（续）

- 相关子查询：子查询的查询条件依赖于父查询
  - 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

3.4.3 嵌套查询

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（SOME）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] 查询与“张暖”在同一个系学习的学生。

此查询要求可以分步来完成

① 确定“张暖”所在系编号

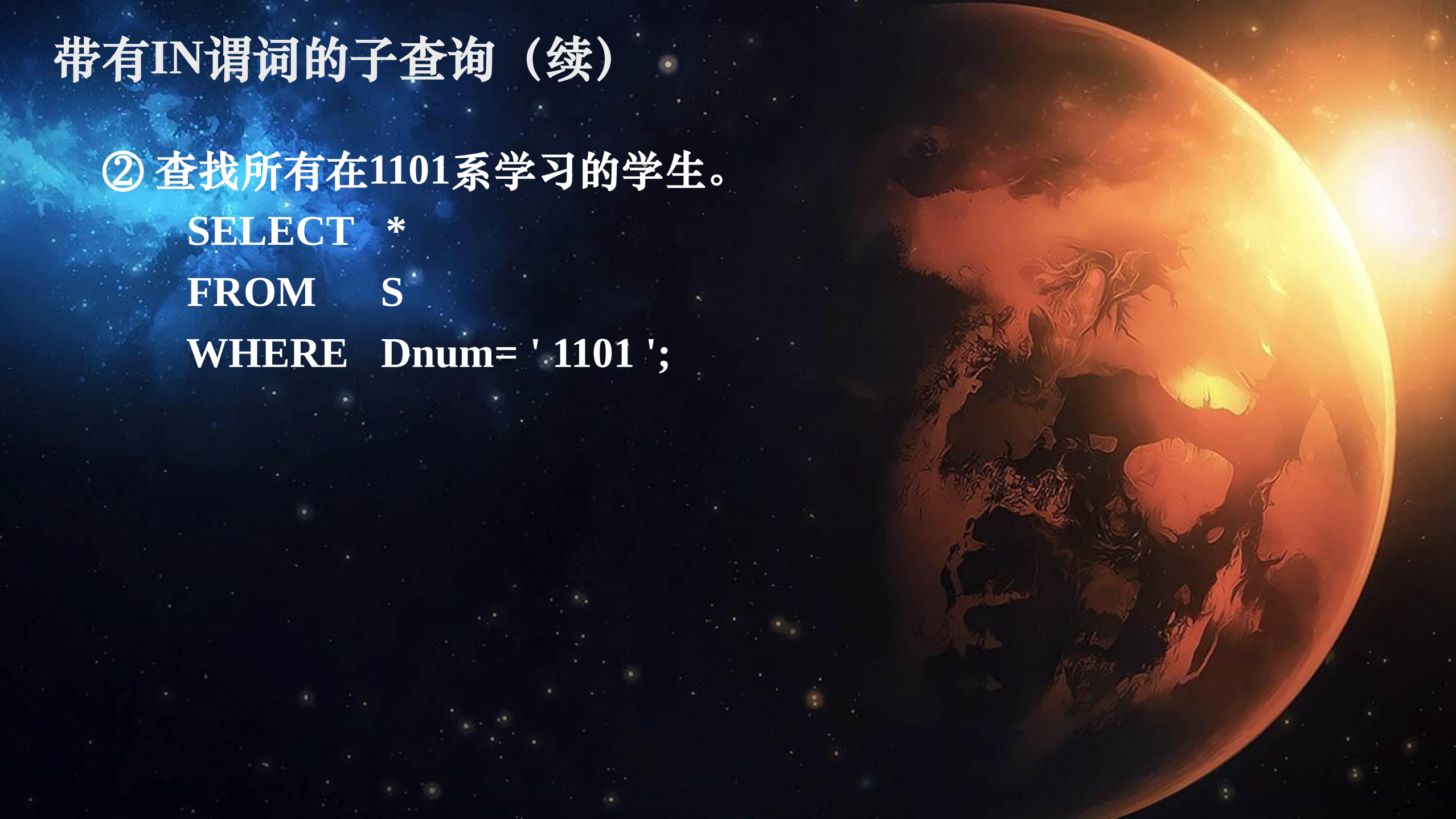
```
SELECT  Snum
FROM    S
WHERE  Sname= ‘ 张暖';
```

结果为： 1101

带有IN谓词的子查询（续）

② 查找所有在1101系学习的学生。

```
SELECT  *
FROM    S
WHERE   Dnum= ' 1101 ';
```



带有IN谓词的子查询（续）

将第一步查询嵌入到第二步查询的条件中

```
SELECT  *
 FROM S
WHERE Dnum IN
        (SELECT Dnum
         FROM D
         WHERE Sname= ‘张暖’);
```

此查询为不相关子查询。

带有IN谓词的子查询（续）

用自身连接完成查询要求:

查询与“张暖”在同一个系学习的学生??

带有IN谓词的子查询（续）

[例]查询选修了课程名为“数据库原理与技术”的学生学姓名

```
SELECT  Sname
FROM    S
WHERE Snum  IN
      (SELECT Snum
       FROM    SC
       WHERE  Cnum IN
            (SELECT Cnum
             FROM C
             WHERE Cname= ‘数据库原理与技术'
        )
      );
```

③ 最后在S关系中取出Sname

② 然后在SC关系中找出选修了该号课程的学生学号

① 首先在C关系中找出“数据库原理与技术”的课程号

3.4.3 嵌套查询

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（SOME）或ALL谓词的子查询

4.带有EXISTS谓词的子查询

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< >）。

[例]查询与“张暖”在同一个系学习的学生

[例]列出工资高于杨平的所有教师的姓名、职称和工资

带有比较运算符的子查询（续）

[例]找出每个学生超过他选修课程平均成绩的课程号。

```
SELECT Snum, Cnum
 FROM   SC  x
WHERE Score >(SELECT AVG（Score）
                FROM  SC y
                    WHERE y.Snum=x.Snum);
```

相关子查询

带有比较运算符的子查询（续）

- 可能的执行过程
  - 从外层查询中取出SC的一个元组x，将元组x的Snum值（200908330413）传送给内层查询。

```
SELECT AVG(Score)
FROM SC y
WHERE y.Snum=' 200908330413‘;
```

带有比较运算符的子查询（续）

- 可能的执行过程（续）
  - 执行内层查询，得到值70用该值代替内层查询，得到外层查询：

```
SELECT Snum,Cnum
FROM   SC x
WHERE  Score >=70;
```

3.4.3 嵌套查询

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（SOME）或ALL谓词的子查询

4.带有EXISTS谓词的子查询

带有ANY或ALL谓词的子查询（续）

谓词语义

- ANY：任意一个值
- ALL：所有值

带有ANY（SOME）或ALL谓词的子查询 （续）

[例]列出工资比'22'学院中最低工资高的其他学院的教师的信息。

[例]列出工资高于‘22’学院中最高工资的其他学院的教师的信息。

能否用聚集函数实现？？？

带有ANY（SOME）或ALL谓词的子查询（续）

ANY（或SOME），ALL谓词与聚集函数、IN谓词的等价转换关系

	=	<>或!=	<	<=	>	>=
ANY	IN	--	<MAX	<=MAX	>MIN	>= MIN
ALL	--	NOT IN	<MIN	<= MIN	>MAX	>= MAX

3.4.3 嵌套查询

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（SOME）或ALL谓词的子查询

4.带有EXISTS谓词的子查询

带有EXISTS谓词的子查询

- 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。

带有EXISTS谓词的子查询（续）

- 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

带有EXISTS谓词的子查询（续）

[例]查询选修了‘B3503021’号课程的学生姓名

能用已经学过的其他方法实现吗？？？

思路分析：

- 本查询涉及S和SC关系
- 在S中依次取每个元组的Snum值，用此值去检查SC表
- 若SC中存在这样的元组，其Snum值等于此S.Snum值，并且其Cnum= ‘B3503021’，则取此S.Sname送入结果表

```
SELECT Sname
FROM S
WHERE EXISTS
    (SELECT *
     FROM SC
     WHERE Snum=S.Snum AND Cnum= ' B3503021 ');
```

带有EXISTS谓词的子查询（续）

[例] 查询没有选修‘B3503021’号课程的学生姓名。

有几种方法？？？

带有EXISTS谓词的子查询（续）

- 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

带有EXISTS谓词的子查询（续）

[例]查询与“张暖”在同一个系学习的学生。

可以用带EXISTS谓词的子查询替换：

有几种方法？？？

带有EXISTS谓词的子查询（续）

- 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：

    （∀x）P≡¬ （∃x（¬P））

带有EXISTS谓词的子查询（续）

[例] 查询选修了全部课程的学生姓名。

```
SELECT Sname
FROM S
WHERE NOT EXISTS
        (SELECT *
         FROM C
         WHERE NOT EXISTS
                 (SELECT *
                  FROM SC
                  WHERE Snum= S.Snum
                      AND Cnum= C.Cnum
                 )
        );
```

带有EXISTS谓词的子查询（续）

- 用EXISTS/NOT EXISTS实现逻辑蕴涵（难点）

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

$$p \rightarrow q \equiv \neg p \vee q$$

带有EXISTS谓词的子查询（续）

[例 3]查询至少选修了学生201215122选修的全部课程的学生号码。

解题思路：

- 用逻辑蕴涵表达：查询学号为x的学生，对所有的课程y，只要201215122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201215122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：（∀y） $p \rightarrow q$

带有EXISTS谓词的子查询（续）

- 等价变换：

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

- 变换后语义：不存在这样的课程y，学生201215122选修了y，而学生x没有选。

带有EXISTS谓词的子查询（续）

- 用NOT EXISTS谓词表示：

```
SELECT DISTINCT Sno
FROM SC SCX
WHERE NOT EXISTS
        (SELECT *
         FROM SC SCY
         WHERE SCY.Sno = ' 201215122 '  AND
                NOT EXISTS
                (SELECT *
                 FROM SC SCZ
                 WHERE SCZ.Sno=SCX.Sno AND
                        SCZ.Cno=SCY.Cno));
```

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基于派生表的查询

3.4.5 Select语句的一般形式

3.4.4 集合查询

- 集合操作的种类
  - 并操作UNION

    MySQL不支持
  - 交操作INTERSECT
  - 差操作EXCEPT
- 参加集合操作的各查询结果的列数必须相同;对应项的数据类型也必须相同

集合查询（并）

[例]查询选修了课程c01或者选修了课程c02的学生学号

[例]查询学校中所有师生的姓名

- 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

集合查询（交）

[例]查询选修课程c01的学生集合与选修课程c02的学生集合的交集

本例实际上是查询既选修了课程c01又选修了课程c02的学生

[例]查询学生姓名与教师姓名的交集

本例实际上是查询学校中与教师同名的学生姓名

集合查询（差）

[例]查询选修课程c01的学生集合与选修课程c02的学生集合的差集

本例实际上是查询选修了课程c01没有选修课程c02的学生

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基于派生表的查询

3.4.6 Select语句的一般形式

3.4.5 基于派生表的查询

- 子查询不仅可以出现在WHERE子句中，还可以出现在FROM子句中，这时子查询生成的临时派生表（Derived Table）成为主查询的查询对象

[例]找出每个学生超过他自己选修课程平均成绩的课程号??
(你能想到几种方法？？)

```
SELECT Snum, Cnum
FROM SC, (SELECT Snum, Avg(Score)
                FROM SC
          GROUP BY Snum)
                AS   Avg_sc(avg_snum,avg_score)
WHERE SC.Snum = Avg_sc.avg_snum
  and SC.Score >Avg_sc.avg_score
```

基于派生表的查询（续）

- 如果子查询中没有聚集函数，派生表可以不指定属性列，子查询SELECT子句后面的列名为其缺省属性。

[例3.60]查询所有选修了‘B3503021’号课程的学生姓名，可以用如下查询完成：

```
SELECT Sname
FROM    S,
            (SELECT Snum FROM SC WHERE Cnum=' B3503021 ')
AS SC1
WHERE  S.Snum=SC1.Snum;
```

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基于派生表的查询

3.4.6 SELECT语句的一般形式

3.4.6 SELECT语句的一般格式

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]] …

|(<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]];

1. 目标列表达式的可选格式

- 目标列表达式格式

  （1） *

  （2） <表名>.*

  （3） COUNT([DISTINCT|ALL]*)

  （4） [<表名>.]<属性列名表达式>[,<表名>.]<属性列名表达式>]…

其中<属性列名表达式>可以是由属性列、作用于属性列的聚集函数和常量的任意算术运算（+，-，*，/）组成的运算公式

2. 聚集函数的一般格式

COUNT

SUM

AVG （[DISTINCT|ALL] <列名>）

MAX

MIN

3. WHERE子句的条件表达式的可选格式

(1)

<属性列名>

<属性列名> θ <常量>

[ANY|ALL] （SELECT语句）

(2)

<属性列名> <属性列名>

<属性列名> [NOT] BETWEEN <常量> AND <常量>

（SELECT语句） （SELECT语句）

WHERE子句的条件表达式格式（续）

（3） <属性列名> [NOT] IN （<值1>[，<值2>] …）

（SELECT语句）

（4） <属性列名> [NOT] LIKE <匹配串>

（5） <属性列名> IS [NOT] NULL

（6） [NOT] EXISTS （SELECT语句）

WHERE子句的条件表达式格式（续）

（7）

<条件表达式> { AND / OR } <条件表达式> { AND / OR } <条件表达> …

数据库原理与技术

浙江農林大學
ZHEJIANG A&F UNIVERSITY

数计 刘丽娟