



特种数据库（基于对象的数据库）

单 位：重庆大学计算机学院

主要学习目标

- 了解对象数据库的基本概念
- 理解对象数据库上的SQL操作



思考问题

- 对象如何在关系数据库中映射？
- 一对一，一对多，多对一，多对多？

一 复杂数据类型

讨论1. 数据库是否需要支持对复杂数据的存储?

应用示例:

在图书馆应用中, 希望存储如下信息:

书名

作者列表

出版商

关键字集合

如果对上述信息定义一个关系, 则很多属性域都是非原子的。

【图1】

1. 复杂数据类型

返回

1) 常存在哪些复杂数据类型, 如何存储?

title	author	position
Compilers	Smith	1
Compilers	Jones	2
Networks	Jones	1
Networks	Frick	2

authors

title	keyword
Compilers	parsing
Compilers	analysis
Networks	Internet
Networks	Web

keywords

可用多个关系存储

title	pub-name	pub-branch
Compilers	McGraw-Hill	New York
Networks	Oxford	London

books4

图22-2 具有4NF的关系数据表 - books

可否直接存储?
哪一种模型更易理解和使用?

结构描述:

存储内容:

title	author-set	publisher (name, branch)	keyword-set
Compilers	{Smith, Jones}	(McGraw-Hill, New York)	{parsing, analysis}
Networks	{Jones, Frick}	(Oxford, London)	{Internet, Web}

多值属性(子关系)

记录结构属性

多值属性(子关系)

图22-1 非1NF的数据文件 - books

2. 面向对象的数据库

2) 什么是对象-关系数据库和面向对象的数据库?

对象-关系数据库系统:

(在关系数据库基础上, 扩展支持对复杂数据对象的存储能力)

对象-关系数据模型 (object-relational data model) 通过提供更加丰富的类型系统扩展了关系数据模型, 它包括了复杂数据类型和面向对象。关系查询语言, 特别是 SQL, 需要相应的扩展以处理更丰富的类型系统。这种扩展试图在扩展建模能力的同时保持关系的基础, 特别是对数据的声明式访问。对象-关系数据库系统 (object-relational database system), 也就是基于对象-关系模型的数据库系统, 为想要使用面向对象特性的关系数据库用户提供了一个方便的移植途径。

面向对象的数据库系统:

(在对象程序设计语言上, 扩展支持对复杂数据对象的存储能力)

接着, 我们说明对于面向对象程序设计语言的本地类型系统中的数据的持久化的支持问题。实践中使用了两种方法:

1. 建立面向对象的数据库系统 (object-oriented database system), 即一个以本地方式支持面向对象类型系统, 而且允许面向对象编程语言使用本地语言的类型系统直接访问数据的数据库系统。

二 SQL支持的复杂数据类型

讨论2. SQL标准中, 支持哪些复杂数据类型?

```
create type Person as (  
  name varchar(20),  
  address varchar(20) );
```

```
create type Student 类型继承  
under Person (  
  degree varchar(20),  
  department varchar(20) );
```

```
create type Teacher 类型继承  
under Person (  
  salary integer,  
  department varchar(20) );
```

```
create type Name as (  
  firstname varchar(20),  
  Lastname varchar(20) )  
Final; 结构类型
```

```
create type Address as (  
  street varchar(20),  
  city varchar(20),  
  zipcode varchar(9) )  
not final; 结构类型
```

```
create type PersonType1 as (  
  name Name,  
  address Address,  
  dateOfBirth date)  
not final;
```

Yes

2) 可用已有复杂数据类型定义新类型, 支持继承?

数组类型

```
create type Publisher as (  
  name varchar(20),  
  branch varchar(20) );
```

```
create type Book as (  
  title varchar(20),  
  author_array varchar(20) array[10],  
  pub_date date,  
  publisher Publisher,  
  keyword_set varvhar(20) multiset);
```

多重集类型

返回

1) 什么是结构类型, 数组类型和多重集类型, SQL中如何定义?

3) 复杂数据类型上可以定义方法吗?

Yes

```
create type PersonType2 as (  
  name Name,  
  address Address,  
  dateOfBirth date)  
not final  
method ageOnDate (onDate date)  
returns interval year;
```

但需单独创建方法体:

```
create instance method  
ageOnDdate(onDate Date)  
Return interval year  
for PersonType2  
begin  
  return (onDate - self.dateOfBirth);  
end
```

三 在SQL中复杂关系表的定义

讨论3. SQL标准中,
如何定义带复杂数
据类型的表?

```
create table person1 (  
  name Name,  
  Address Address,  
  dateOfBirth date);  
  
create table person2 of PersonType;  
  
create table person3 (  
  name row ( firstname varchar(20),  
  lastname varchar(20) ),  
  address row ( street varchar(20),  
  city varchar(20),  
  zipcode varchar(9) ),  
  dateOfBirth date );
```

【图2】

name	address	dataOfBirth
(firsname,Lastname)	(street,city,zipcode)	
(David, Smith)	(23th, Newyork, 0001)	20-03-2001

复杂类型定义

```
create table books of Book;
```

【图3】

复杂表:books

```
create table people of Person;  
  
create table students of Student  
under people;  
  
create table teachers of Teacher  
under people;
```

【图4】

1) 图2中定义的三个
关系表具有什么样的
复杂数据结构?

2) 图3中定义的
复杂表什么样?

3) 复杂表的定义
也支持继承?

← Yes

子表对应于E-R图中特殊化
子表的类型必须是父表类型

复杂表person1
Person2
person3

四 在SQL复杂关系表上插入数据

```
create function Name(firstname varchar(20), lastname varchar(20))
returns Name
begin
    set self.firstname = firstname;
    set self.lastname = lastname;
end
```

```
insert into Person1
values ( new Name('John', 'Smith'),
new Address('20 Main St', 'New York', '11001'),
date '1960-8-22');
```

```
insert into books
values ('Compilers', array['Simth', 'Jones'],
new Publisher('McGraw-Hill', 'New York'),
multiset[ 'parsing', 'analysis']);
```

讨论4. SQL标准中,
如何在复杂关系表
上插入数据?

1) 什么是构造器
函数, 如何生成
复杂数据项?

2) 如何将复杂数
据记录插入复杂
数据关系表?

复杂数据类型的赋值需特殊处理: 构造器函数!
数组类型可以采用array构造器;
多重集类型可采用multiset构造器;
结构类型须采用显示定义的构造器(如Name)。

五 在SQL复杂关系表上查询

```
select name.lastname, address.city  
from person;
```

或者:

```
select name->lastname, address->city  
from person;
```

【图5】

```
select name.lastname, ageOnDate(current_date)  
from person;
```

【图6】

```
select author_array[1], author_array[2], author_array[3]  
from books  
where title = 'Database System Concepts';
```

【图7】

```
select title  
from books  
where 'database' in (unnest(keyword_set));
```

【图8】

讨论5. SQL标准中,
如何在复杂关系表
上查询数据?

1) 如何查询结构
类型中数据?

2) 图6中的查询
结果是什么?

ageOnDate()表示调用
复杂数据类型上的方法

3) 如何查询数
组类型中数据?

unnest用于解套数组中元素
(但元素无序)

五 在SQL复杂关系表上查询(续)

```
select B.title, A.author
```

```
from books as B, unnest(B.author_array) as A(author);
```

unnest解套出的元素集合形成一个(无序)子表

4) 如何理解图9中的
unnest...子句?

```
select title, A.author, A.position
```

```
from books as B, unnest(B.author_array) with ordinality as A(author, position);
```

unnest解套出的元素集合形成一个‘有序’子表(得到前面的**authors**关系)

```
select title, A.author, publisher.name as pub_name,
```

```
publisher.branch as pub_branch, K.keyword
```

```
from books as B, unnest(B.author_array) as A(author),
```

```
unnest(B.keyword_set) as K(keyword);
```

由原复杂关系表形成一个平面关系表

【图9】

```
select title, collect(author) as author_set,
```

```
Publisher(pub_name, pub_branch) as publisher,
```

```
collect(keyword) as keyword_set
```

```
from flat_books
```

由平面关系表形成一个复杂关系表

```
group by title, publisher;
```

【图10】

5) 图10中的查询
结果什么样?

title	author-set	publisher	keyword-set
		(name, branch)	
Compilers	{Smith, Jones}	(McGraw-Hill, New York)	{parsing, analysis}
Networks	{Jones, Frick}	(Oxford, London)	{Internet, Web}

flat_books:

title	author	pub_name	pub_branch	keyword
Compilers	Smith	McGraw-Hill	New York	parsing
Compilers	Jones	McGraw-Hill	New York	parsing
Compilers	Smith	McGraw-Hill	New York	analysis
Compilers	Jones	McGraw-Hill	New York	analysis
Networks	Jones	Oxford	London	Internet
Networks	Frick	Oxford	London	Internet
Networks	Jones	Oxford	London	Web
Networks	Frick	Oxford	London	Web

六 SQL中的对象标识和引用类型

```
create type Department(  
  name varchar(20),  
  head ref(Person) scope people);
```

```
create table departments of Dpartment;  
或者:
```

```
create type Department(  
  name varchar(20),  
  head ref(Person) );
```

```
create table departments of Dpartment  
  (head with options scope people);
```

【图11】

```
create table people of Person  
  ref is person_id system generated;
```

```
insert into departments  
  values ('CS', null);
```

```
update deparments  
  set head = (select p.person_id  
    from people as p  
    where name = 'John')  
  where name = 'CS';
```

【图12】

通过ref和scope标识符

讨论6. SQL标准中, 如何支持对象标识和引用?

1) 如何定义'引用类型属性(解释图11)?

2) 如何指定对象的唯一标识?

1) 系统自动生成唯一标识

3) 具有引用类型属性的关系上, 如何插入元组?

首先插入元组, 引用类型属性置空, 然后对引用类型属性赋值唯一标识

六 SQL中的对象标识和引用类型(续)

```
create type Person (  
  name varchar(20),  
  address varchar(20),  
  ref using varchar(20) );
```

```
create table people of Person  
  ref is person_id user generated;
```

```
insert into people(person_id, name, address)  
  values ('01284567', 'John', '23 Coyote Run');
```

```
insert into departments  
  values('CS', '01284567');
```

【图13】

```
create type Person (  
  name varchar(20) primary key,  
  address varchar(20) ),  
  ref from(name);
```

```
create table people of Person  
  ref is person_id derived;
```

```
insert into departments  
  values('CS', 'John');
```

【图14】

```
select deref(head).name  
from departments;
```

【图15】

2) 还允许用户生成唯一标识

插入元组时，需人为指定唯一标识

3) 还允许指定主码为唯一标识

插入元组时，输入主码为唯一标识

插入元组时，输入主码为唯一标识

4) 还有哪些方式可指定对象唯一标识?

5) 插入元组时唯一标识如何赋值?

6) 如何查看引用类型属性中的属性内容?



课堂思考小问题

- 面向对象相对于关系型数据库开发，有什么区别？优点是什么？

课堂小结和作业安排

- 基本知识：
 - 对象数据库的基本概念
 - SQL如何操作
- 扩展学习：
 - 哪些数据库支持面向对象？
- 作业
 - 第章习题： 22. 1, 22. 3