统一多维数据模型的后关系数据库体系结构

陈松林,王于同
CHEN Song-lin,WANG Yu-tong

杭州电子科技大学 计算机学院,杭州 310018
School of Computer,Hangzhou Dianzi University,Hangzhou 310018,China
E-mail:linzhi_666333@163.com

CHEN Song-lin,WANG Yu-tong.Architecture of post-relational database using unified multidimensional data model. Computer Engineering and Applications,2009,45(8):146-148.

Abstract: Along with great range application of object-oriented programming and Web technology,it is in dire need of high efficient support to these new technologies in the field of database,but the traditional relational database is bad in those aspects for the limit of itself.Aiming at this problem,and combining the developing of object-oriented database and XML database,this paper advanced the conceptual model of a multi-interface mix post-relational database's architecture based on application driving,and proved it in theory.

Key words: multi-interface;post-relational database;multidimensional data structure

摘 要:随着面向对象编程技术的推广和Web技术的广泛应用,迫切需要在数据库领域提供对这些新技术的高效支持,而传统的关系数据库由于其本身的局限性在这些方面表现不佳。针对这一问题,并结合对象数据库及XML数据库的发展现状,提出了一种基于应用驱动的多接口混合型后关系数据库体系结构的概念模型,并在理论上加以验证。

关键词:多接口;后关系数据库;多维数据结构
DOI:10.3778/j.issn.1002-8331.2009.08.045 **文章编号**:1002-8331(2009)08-0146-03 **文献标识码**:A **中图分类号**:TP311.13

1 前言

近年来,面向对象的程序设计成为应用程序开发的前景。于是基于对象编程与关系数据库之间的所谓"阻抗失谐"的呼声日渐高涨。甚至有些学者早在十几年前就把对象数据库作为替代关系数据库的下一代数据库。然而十多年过去了,面向对象的程序设计技术已经在应用程序领域普遍推行,而在数据库领域依然步履维艰。但也可以看到面向对象技术对于数据库领域的影响是深远的。传统的关系数据正在不断地向着面向对象技术进行扩展,由此产生了对象关系数据库。而对象数据库也在工程应用与科学统计、多媒体存储和文献管理等领域发展迅速。

特别地,由于Internet的高速发展和广泛应用及软件系统间交换数据的需要,一种基于XML数据格式的应用日益平凡,关于XML数据的存储与管理问题,又产生了XML本源数据库、XML对象数据库及XML关系数据库。

而基于OLAP的数据仓库和数据挖掘技术也对现有的关系型数据库提出了挑战,从而产生了ROLAP数据库、MOLAP数据库及HOLAP数据库[1]。

通过以上的分析,不难看出,如果说数据本身只是计算机存储器上的一些字符的话,那么数据一旦涉及到应用,就展现出多种模式和丰富复杂的结构。

2 基于应用驱动的思考

关系数据库系统支持一些特定的小数据类型(如整数、日期、字符串),实践已经证明它们足以支持传统应用领域(如管理)的数据处理。这也为关系型数据库供应商积聚了巨大的动力,并占领大量的市场份额。相对于其他数据库系统而言,关系数据库具有简单数据类型、功能强大的查询语言和高保护性。基于目前流行的几种数据库系统性能的比较见表1(由于篇幅所限,这里暂不深入讨论与XML及OLAP有关的内容)。

表1 几种流行数据库系统的性能比较

数据库系统	数据类型	查询性能	事务性能
关系系统	简单数据类型	功能强大的查询语言	高保护性
对象数据库	复杂数据类型	与程序设计语言集成	高性能
对象-关系数据库	复杂数据类型	功能强大的查询语言	高保护性

从表1可以看出,对象关系数据库作为关系与对象之间的一种中间系统兼容了关系数据库功能强大的查询语言和高保护性及对象数据库的复杂数据类型的优点,但是在事务性能上表现不是很理想。

当再从程序设计数据应用层次的角度观察以上三种数据库系统时,可以发现它们处在不同的应用层次上,如图1。

由图1可以看出,对象数据库对于如今流行的面向对象编

作者简介:陈松林(1974-),硕士研究生,主要研究方向:数据库;王于同(1945-),教授,硕士生导师,主要研究方向:数据库,分布式处理,并行计算等。
收稿日期:2008-01-25 **修回日期**:2008-04-24

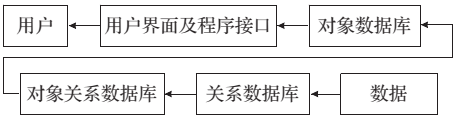

图 1 数据库的数据应用层次

程技术的支持是最直接的,而关系数据库在数据本身的管理上是最直接的,也因此它的查询性能最好。当由于对象关系数据库在操作性能上的表现过于平庸时,我们不得不寻求另一种解决方案。

对于数据用户及数据应用类型可以进行以下分类,如表 2。

表 2 数据用户及数据应用类型分类表

面向对象应用程序用户	传统视图用户	Web 应用	OLAP 应用

3 基于应用驱动的后关系数据库体系结构的提出

鉴于以上的分析及对数据应用用户分类的思考,提出了基于数据应用驱动的后关系数据库体系结构的概念模型如图 2。

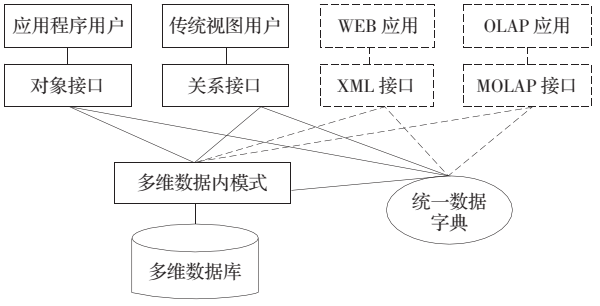

图 2 基于应用驱动的后关系数据库体系结构的概念模型

4 理论依据及性能优势

4.1 单一数据架构的多维存储模式

这个模型的主导思想就是前面分析的面对数据的多种应用需求,创建一个多模式多接口的数据库管理系统。可以利用对象数据模式实现复杂的对象数据类型及用户自定义类型,并达到与面向对象应用的直接交互,而关系模式则可以完成通常的数据分析和报表应用,及对 ODBC 和 JDBC 接口的支持和在必要的时候把对象映射成为关系表。

而实现这一切的核心是采用高效的多维数据引擎,可以在丰富的数据结构支持下,高效而简洁地存储数据。统一数据字典可以定义类和表,并且提供到多维数据结构的映射,这种映射可以自动产生。这样,就给程序员提供了自由地存储和访问数据的空间,可以选择对象、SQL 或者直接访问多维数据结构。无论何时,只要定义了数据库对象类,系统可以通过一种机制自动产生这一数据的关系描述(为 SQL 准备)。同样,如果一个关系数据库的 DDL 定义被导入到数据宝典时, 系统自动产生这一数据的关系描述和对象描述,这样就可以既用 SQL 访问,又用对象访问它。系统自动保持这些描述的协同性,所以只要编辑维护一种数据定义,编程人员可以通过对象或者关系表来编辑和浏览数据字典。系统自动创建关于对象和表是如何存储在多维数据结构中的映射,或者程序员也可以轻易地控制这种映射。

在后关系数据库中,将不再强调关系模式与对象模式的称呼,因为这样好像后关系数据库只是两种数据库模式的简单混合,而实际中把关系模式与对象模式作为两种应用接口,强调它们在存储层的统一性和在应用层上的分工与合作,并且更看重它们的合作与交互,因此以下简称为关系接口与对象接口。

4.2 关系接口存取及 SQL 的对象扩展

关系接口的关系表在存储模式上也就是一些二维数组,可以直接存储在多维数组中。SQL 的对象扩展使得可以将存储在多维数组上的对象数据很方便地转换为关系表[2]。

4.2.1 结构类型

结构类型允许直接表示复合属性,例如可以定义如下结构类型表示一个复合属性 address:

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

类似地,下面的结构类型可以用于表示一个复合属性 name:

```
create type Name as(firstname varchar(20),lastname var-
char(20))final
```

现在可以使用这些结构类型在关系里创建复合属性,只需简单地声明一个属性具有一个这样的复合类型。例如,可以创建如下的一个 customer 表:

```
create table customer(name Name,address Address,date-
OfBirth date)
```

4.2.2 继承

假定有如下关于人的类型定义:

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

希望在数据库中对那些是学生和教师的人分别存储一些额外的信息,由于学生和教师也同样是人,就可以在 SQL 中使用继承来定义学生和教师类型:

```
create type Student under Person(degree varchar(20),de-
partment varchar(20))
create type Teacher under Person(salary integer,department
varchar(20))
```

4.2.3 嵌套和解除嵌套

先创建一个包含数组和多重集合值属性的图书记录的例子:

```
create type Publisher as(name varchar(20),branch var-
char(20))
create type Book as(title varchar(20),author_array var-
char (20) array [10],pub_date date,publisher Publisher,key-
word_set varchar(20) multiset)
create table books of Book
```

将一个嵌套关系转换成具有更少(或没有)以关系为值的属性的形式的过程被称为解除嵌套(unnesting)。可以使用以下查询来完成这个任务:

```
select title,A.author,publisher.name as pub_name,pub-
lisher.branch as pub_branch,K.keyword from book as B,unnest
(B.author_array)as A(author),unnest(B.keyword_set) as k(key-
word)
```

4.2.4 对象标识和 SQL 中的引用类型

面向对象的程序设计语言提供了引用对象的能力,类型的一个属性可以是对一个指定类型的对象的引用。例如,在 SQL 中可以定义一个 Department 类型,它有一个 name 域和一个引用到对象 Person 类型的 head 域,然后定义一个 Department 类

型的表 department,如下所示:

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

create table departments of Department

在后关系数据库中,把对象数据转换成关系表只需使用以上 SQL 扩展的部分功能, 基于对象特有的一些特性留在对象接口中完成。

4.3 关系对象功能及联系类在对象接口中的使用

4.3.1 对象标识与行标识

为了达到关系接口与对象接口的数据交互,参考文献[3]中 S-子类的概念,将对象以类区间的形式存储,并确定关系接口中的行标识(RID)和对象接口中的类实例(CID)标识由系统自动产生,并统一为一种标识(CID)。即当关系接口中产生一个元组时,系统自动产生一个关系表内唯一的行标识 CID,在对象接口中可以当作类标识使用;而当以对象模式产生一个类实例时,系统自动产生一个类实例标识 CID,并在关系接口中使用。而对象标识 OID 只在对象接口中产生,它可以等同于一个多重继承对象类的行标识, 并与多个 S-子类的类实例标识发生联系,形成如表 3 所示的对象标识表:

表 3 对象标识表

对象名	对象类	S-子类 1	S-子类 2	…
对象 A	AOID	ACID1	ACID2	…
对象 B	BOID	BCID1		
对象 C	COID			

4.3.2 联系类

对象间的联系及类间联系一般通过对象引用及类实例引用来完成,这一特性在处理一些复杂对象间的关系时,非常迅速和有效,这也是对象数据库的特点之一。而这实际上只是对象或类间的一种结构联系,对象或类间的数据互操作往往通过对象的方法接口与消息传递来完成,这在增加程序员的灵活性的同时,也给数据管理系统对数据本身的保护性带来了不可忽视的威胁。而关系数据库通过联系表来完成关系实体间的联系和联系属性,这大大提高了数据查询的速度和联系关系的清晰性,并对数据提供了高保护性。

在后关系数据库中定义对象接口的类关键字(Ckey)来与关系接口中的主关键字(Primary Key)相对应,并参照文献[4]在对象接口中当类间或对象间的引用关系为多对多或带有联系属性时建立联系类,来达到与关系接口的交互。类关键字将基于类中的某些属性,与关系表中的关键字相对应,而 CID 和 OID 由系统自动产生和维护,属于后关系数据库系统管理的全局标识(CID 需要与类或表名联系在一起才能成为全局标识)。

4.4 高效的多维数据引擎

关系数据库一般只对单个索引值建立索引,当对多个关键字进行查找时,通过对每个关键字的单一索引来分别查询并在结果集中求交来完成。这在基于主码的联系应用中表现优良。而对象大多通过直接引用来完成,在查找及索引性能上表现低劣,这将使得大型软件开发的程序员不得不花费大量时间和精力来整理和查找自己定义的无数对象实例及其概念模型。后关系数据库的多维数据引擎将改变这一现实,提供优良的多维数据查询性能,并可以通过关系表来实现对对象的查询和更新。

虽然,目前的多维数据模型大多应用在 OLAP 领域,当在后关系数据库中使用参考文献[5]提出的多维类型结构(MTS)来对后关系数据进行聚集存储时,也可以达到非常高的查询性能。

一个多维类型结构 MTS 的四维结构的例子,如图 3:

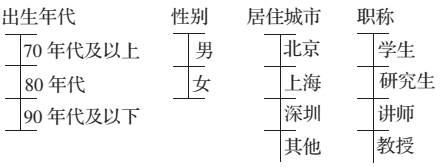

图 3 四维度的多维类型结构

图 3 中四个维中维值的每个组合构成了一个逻辑“单元”,这个单元在物理上组织成页的块,而一块由磁盘上一系列连续页组成[2]。例如可以找到用维值<80,男,北京,研究生>标识的逻辑单元的几个数据块 9、23、70。

维块索引在每个维之上创建。每个维块索引和传统的 B 树索引以相同的方式构造,不同的是,在叶结点层次,码指向块标识符(BID)而不是记录标识符(RID)。一个切片,或者说包含所有在一个维中都有相同值的记录的页面的块集合,在相关联的维块索引中由一个该码值的 BID 链表来表示。如图 4:

图 4 地址域值“北京”的维块索引记录

因此可以建立一个四维数组来对以上类型数据进行聚集存储和查找,也可以用这种 MTS 结构对对象进行聚集存储和查询。

5 实例

由于后关系数据库仍处于开发与研究阶段, 借用由美国 InterSystems 公司开发的 Caché 后关系型移动嵌入式数据库作为实验模板来观察后关系数据库所具有的优良性能[6]。

Caché 数据库是新一代的超高性能数据库技术,它基本整合了后关系数据库所具有的各种新特性,还包括一个应用服务器,几种内嵌脚本语言及为基于网页应用程序提供的丰富的集成环境。目前主要应用于医疗、电信、政府、金融部门及旅游行业。

这里采用一家医疗系统正在使用的流行的关系数据库与 Caché 数据库之间进行测试,测试使用的是历史病人的数据(7 张表,超过 650 万条记录),模拟负载 30、60、90 和 120 个并发用户的情况下运行 8 个查询。测试结果如表 4:

表 4 医疗系统使用的关系数据库与 Caché 数据库间的性能测试

#of concurrent users	Average response time (over all eight queries)		Relative performance Caché:关系型数据库
	关系型数据库/ms	Caché/ms	
30	375.125	59.125	6.3:1
60	637.25	137.75	4.6:1
90	915.625	206.875	4.2:1
120	1 146.375	290.125	3.9:1

6 总结

本文没有过多讨论后关系数据库对于 XML 数据及 OLAP

(下转 152 页)