异种数据库中多媒体的查询处理

姚领众　（中国科学院数学研究所　100080）
李文新　（兰州炼油化工总厂　730060）

摘要：本文介绍了异种数据库中多媒体数据的全局查询处理策略。
关键词：异种数据库　HDBUS　多媒体

一、异种数据库联合使用的需求及目的

目前，随着计算机网络技术的发展，特别是 Internet 的兴起，异种数据库技术也得到了前所未有的重视。目前是信息社会，就信息源而论，存在多种数据库服务器是客观现实，由于历史的原因，在一个大企业中，各部门使用不同的数据库也是屡见不鲜。

以一大公司为例，假设该公司的产品、材料、零部件、财务、人事等信息都分别存放在各分公司不同的数据库服务器上。如总公司希望查询某种信息，可以有两种方法。

方法一是逐个访问不同的数据库系统，用每个数据库系统使用的查询手段，最后将查询结果汇总。在概念上应知道各物理数据库的有关知识，而方法二是通过建立全局数据字典，在逻辑上形成全局库。对用户来讲各物理库是透明的，因此就象处理一个本地的物理库一样简单。其中采用的一个重要技术就是设计一种全局查询语言 GSQL，用它对全局库进行各种操作。

基于上述的理由及异种数据库的巨大潜力所在，我们设计了一个异种数据库系统 HDBUS，关于 HDBUS 的其他技术，笔者已在文献[1][2]中作了介绍，这里我们将重点介绍 HDBUS 中多媒体数据的处理策略。

在对 HDBUS 中多媒体的研究，有如下说明：

①全局数据模型：为关系型，多媒体信息有两种数据类型，在全局数据字典中表示为 LONG VARCHAR(大文本)和 LONG BINARY(大二进制)。这种类型支持的最大长度为 2G。

②多媒体数据的存储：HDBUS 中所涉及的各种关系数据库都支持多媒体数据类型。全局关系中的多媒体数据在物理上是存于各局部数据库的多媒体列中的多媒体数据。

③多媒体数据的传播：从远端数据库中查到的多媒体数据，被一次性地传输到客户端，然后显示(播放)而不是边查边显示。

二、PB5 中对多媒体数据的处理方法

由于 HDBUS 的开发工具之一是 PB5(PowerBuilder 5.0)，HDBUS 中对多媒体的查询是全局查询，全局查询最终被分解成针对每个结点局部库的局部查询，该局部查询归根结底是由 PB5 的机制来完成的，因而在实现多媒体的全局查询之前了解开发工具 PB5 对多媒体的处理能力是至关重要的。

由于多媒体数据没有边界，因此处理起来有一定的特殊性。PB5 中对含有多媒体列的表，在这个表上创建数据窗口时不能包含该列。这是因为数据窗口在同一时刻可存放多条数据库记录，而包含了多媒体后最多只能放一条记录。鉴于这一原因，包括 PB5 在内的大多数开发系统都不能在数据窗口中包含多媒体列。这意味着在 PB5 中，数据库最有效的查询和显示工具——数据窗口是无法处理多媒体的。PB5 提供了两条语句专门用于二进制大对象的处理。这两条语句是：

(1)SELECTBLOB 语句

SELECTBLOB 二进制大对象列名 INTO:blob 类型变量 FROM 表名 WHERE 条件

(2)UPDATEBLOB 语句

UPDATEBLOB 表名 SET 二进制大对象列名:blob 类型变量 WHERE 条件

第一条语句用于查询多媒体数据到 blob 类型变量中，第二条语句用于给数据库二进制大对象赋值。用这两条语句可以完成数据库多媒体数据的查询和修改工作。使用这两条语句的要求是，WHERE 子句必须只选出一条记录，换句话说，一次只能播放一个文件。

另外，从上面的语句可以看到，PB5 只能查询和修改数据库中的多媒体数据，不支持插入多媒体数据的 insert 语句。因此，如果想把多媒体数据写入数据库，必须先插入这条记录的其他部分，然后再通过修改记录的方式写入多媒体数据。

GSQL 中两条专门用来处理多媒体的语句 SEBLOB

与 UPBLOB 与上述两条语句关系密切。事实上,多媒体全局查询语句最终是被分解成对应于每个结点数据库的 SELECTBLOB 语句而被执行的。UPBLOB 也是同理。

三、HDBUS 中多媒体的操纵策略

HDBUS 中对多媒体的操纵也是通过 GSQL 语言来实现的,功能主要包括查询(SELECT 和 SEBLOB)、更新(UPBLOB)插入(INSERT)与删除(DELETE)。其中删除与普通类型数据无差别,用来删除一行记录,因此下面不作进一步介绍。

1. 多媒体全局查询

GSQL 语言中实现多媒体全局查询的语句有两种:SEBLOB 和 SELECT。

(1)用 SEBLOB 语句实现多媒体数据的全局查询。SEBLOB 语句的语法在第三章中已作了描述,格式如下:

SEBLOB　列名 FROM 表名 WHERE　条件;

该语句专为查询某一行某列的数据而设计,因此 WHERE 后的条件必须保证结果为一行,而非多行。SEBLOB 的工作过程如下:

①利用浏览器或全局数据管理器提供的编辑功能建立 SEBLOB 语句;

②语法判断。若正确,则执行下一步,否则报错;

③分解翻译。将 GSQL 语句分解翻译为对应每个局部数据库的 LSQL;

④全局数据管理器将各 LSQL 分别传递给各并行桥;

⑤各并行桥向数据库服务器发出服务申请;

⑥数据库服务器作查询处理;

⑦数据库服务器通知并行桥查询结束;

⑧数据库服务器将查询结果送全局数据管理器并作为临时文件保存;

⑨浏览器(或全局事务管理器的用户界面)显示(或播放)多媒体。

SEBLOB 语句的查询处理过程与 GSQL 中的 SELECT 语句的处理过程略有不同,表现在:①无查询汇总。因为 SEBLOB 语句的 WHERE 子句规定必须只选出一条记录,因此不再需要汇总。②查询到的结果以一个临时文件保存,该文件的格式视所查列的结果不同而不同,有 .avi、.bmp、.gif 等多种形式。在 SELECT 中普通数据类型的结果则是一个 WATCOM 格式的临时表。

(2)用 SELECT 语句实现多媒体数据的全局查询。SQL-92 中没有提供支持多媒体数据的语句。虽然象 ORACLE7, SYBASE 10 等都有专门处理多媒体数据的工具,然而它们所支持的 SELECT、UPDATE 等 DML 语句仍然不具有处理多媒体数据的能力。

我们在 HDBUS 的 SELECT 语句中扩充了支持多媒体的能力,所采用的技术要点如下:

①组成全局库的局部库是存放多媒体信息的物理库。

②局部库中多媒体数据类型的列,在全局库中的对应列其类型定义为字符型列,这一信息被登录在全局数据字典中。

③用 SELECT 语句查询带有多媒体数据类型的表时,查询和显示方式都是按普通类型列来处理的,因为在全局表中,多媒体类型被定义为字符类型。其显示的数据值只是一些标志,如"VOICE"表示所在列为声音信息,"PICT"则为图片,"AVI"是电视剪辑等。这正如同 dBASE, FoxBASE 及 FoxPro 中的备注型字段,当用 BROW 查看表内容时,其表面值为"MEMO"一样。

④多媒体数据的查询是通过二次查询来完成的,当用 SELECT 语句发出查询后,实际上并不涉及多媒体数据,查询含有或不含有多媒体数据的全局库,返回的都是格式化的数据。如果不需查看多媒体数据,则此表格数据便是最后的结果,若想查看某行某列的多媒体数据,用户只需在对应的多媒体列值上双击鼠标。此时便为触发一个多媒体查询事件,这一事件便是二次查询。二次查询主要完成如下工作:

·根据全局数据字典中该全局表的组成结点,构造对应于每个结点局部库的多媒体局部查询语句,其中 WHERE 后的条件由对应行的其他普通型列值决定;

·查询列的结果以相应格式的文件保存;

·对于浏览器界面,只需给浏览器传送多媒体文件的路径。对于全局数据管理器界面,则是通过 OLE(Object Linking and Embedding)技术来实现的。

2. 多媒体数据的更新

GSQL 中对多媒体数据的更新是由 UPBLOB 语句来完成的。UPBLOB 语句的语法如下:

UPBLOB 表名 SET 列名 = [路径] 文件名 WHERE 条件

受多媒体数据特殊性的制约,目前 UPBLOB 也要求 WHERE 子句必须只选出一条记录。UPBLOB 语句的执行过程与 GSQL 中的语句 UPDATE 十分相似。也要经过如下的过程:输入 - 语句检查 - 分解翻译 - LSQL - 数据库服务处理。其中分解翻译要比 UPDATE 语句复杂。因为 UPDATE 在 HDBUS 所支持的各种数据库中其形式都基本上遵循了 SQL-92 标准。而 UPBLOB 则是非标准语句。最后要分解成为 PB5 支持的 UPDATEBLOB

语句,而且两者在语法上相距甚远。

UPBLOB 在分解翻译时的步骤如下:

(1)第一遍扫描,分离出表名;

(2)根据表名,通过查询全局字典组成该表的结点名;

(3)产生相应结点的 LSQL 字符串(其初值就是 GSQL 语句);

(4)第二遍扫描,将各 LSQL 语句中的全局表名、列名替换为对应结点组成该全局表的局部表表名和列名;

(5)第三遍扫描,分离出包含多媒体信息的文件名(包括路径);

(6)打开并读取上述文件,将其读入一个 BLOB 类型的变量 media 中;

(7)将(5)中的文件及其路径替换为 INTO :media;

(8)保留字替换。将 UPBLOB 替换为 UPDATEBLOB。

3.多媒体数据的插入

目前,象 PB 尚未提供将多媒体数据直接插入到数据库的语句。只能通过间接的办法来实现,采取的办法是:通过 INSERT 语句先插入这条记录的其他部分,然后再通过用 UPBLOB 修改记录的方式写入多媒体数据。但是,这种数据输入方法用起来显然很麻烦。

HDBUS 提供了直接实现插入的语句,实现的思路是:将 GSQL 中的 INSERT 扩充,使其也支持多媒体数据的插入。扩充后的 INSERT 语句,在插入多媒体数据时,格式稍有不同。即如果某列为多媒体型数据,则 VALVES 后括号中对应的常数值为存放该多媒体信息的路径及其文件名。

实现方法步骤如下:

(1)首遍扫描 INSERT 语句,提取全局表表名及其语句中出现的列名;

(2)判断列中有无多媒体列,无则按普通 INSERT 语句分解翻译,有则转(3);

(3)提取出 INSERT 语句中的多媒体列及其要插入的路径和文件名字符串,并从语句中消去;

(4)消去了多媒体列的 INSERT 语句已是普通的 INSERT 语句,按普通 INSERT 分解翻译;

(5)对提取的多媒体列列名及其含有插入信息的多媒体文件名及其路径按如下步骤构造 UPDATEBLOB 语句(其中一个多媒体对应一条 UPDATEBLOB):

①打开多媒体文件,将其值读入 BLOB 型变量比如 media 中;

②构造型如

UPDATEBLOB 局部表名 SET 多媒体列 = :media WHERE 条件;

的语句,其中 WHERE 后的子句由前面 INSERT 语句的阐述。

例:全局表 employee 含有 4 列:emp _ no、name、voice、photo,其中 voice 和 photo 分别用来存放 .wav 和 .bmp 信息,组成全局表的各局部表的表名及列名分别与全局表的表名、列名相同。现要在 SQL SERVER 结点(节点名为 server6)上插入一条记录,GSQL 语句如下:

```
INSERT INTO server6. employee(emp _ no, name, voice, photo)
        VALUES(1000,'Wang Tao','D:/WAV/WANG.WAV',
  'D:/BMP/WANG.BMP');
```

设经上面第(5)步后'D:/WAV/WANG.WAV'与'D:/BMP/WANG.BMP'分别被读入到 BLOB 变量 m _ voice 与 m _ photo 中,则经过分解翻译后产生的基于 SQL SERVER 结点的 LSQL 语句组如下:

```
INSERT INTO employee(emp _ no, name)
  VALUES(1000,'Wang Tao');

UPDATEBLOB employee
  SET voice: = m _ voice
  WHERE emp _ no = 1000 AND name = 'Wang Tao';

UPDATEBLOB employee
  SET photo: = m _ photo
  WHERE emp _ no = 1000 AND name = 'Wang Tao';
```

四、结论

本文介绍了我们设计实现的异种数据库系统 HDBUS 中多媒体数据的处理策略,主要是通过在 GSQL 增加专门用于多媒体数据查询、更新和插入的操纵语句来实现。整个过程对用户来讲都是透明的,而且 GSQL 中对多媒体数据的操作其语法形式与非多媒体数据极为相似,克服了传统方法的不足。

参考文献

[1] 宋瀚涛,赵敬中,姚领众。异种数据库与多媒体联合使用技术。计算机世界,1998 年 9 月 14 日

[2] 姚领众,宋瀚涛,梁允荣。UUHDB 的体系结构及功能实现。计算机世界,1998 年 9 月 14 日

(来稿时间:1999 年 3 月)