在多层数据库结构中利用事务处理保证数据的一致性

Utilizing Transaction to Realize Data Consistency in Multi－tier Database Architecture

李德军 （广东商学院教育技术中心 510320）

摘要：本文先分析了数据库管理系统中事务的概念，到 Delphi 中事务处理在多层结构中的实现方法，并且以客户端显式和中间层隐式的事务处理方法作讨论，阐述了在 Delphi 中如何利用事务处理保证基于多层数据库的数据一致性问题。

关键词：Delphi 事务 数据库 一致性

目前，针对数据库的开发应用系统中，多层数据库结构的应用较多，在基于多层数据库的开发应用中，如何做到保证数据的一致性是十分重要的，针对这一问题，本人就 Delphi 如何利用在应用程序中控制服务器端的行为（即通过事务处理）来实现保证数据的一致性问题。

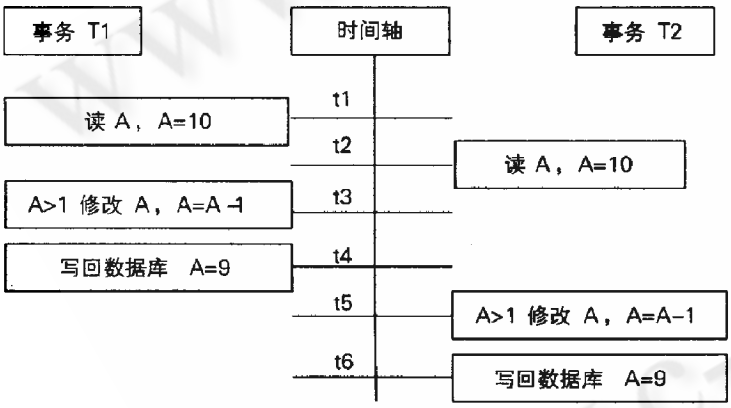

图 1

1 事务的概念

所谓事务就是一种机制，是数据库完整性的最小单位；事务是数据库最小的逻辑工作单元。一个事务的执行是把数据库从一个一致的状态转换成另一个一致的状态。如果多个事务交错地并发执行，就可能相互干扰，造成数据状态的不一致。如果多个并发的事务不加控制去修改数据库，就会产生错误，造成数据的不完整，会造成一个事务的修改覆盖另一个事务的修改。

在基于多层结构的数据库管理中，经常多个用户同时对数据库中同一基表进行操作，现假设某售票系统有甲乙两个售票点在同时销售车票，即有两个事务 T1、T2 在同时对数据 A 进行操作。售票时先查询 A 的值，如 A >0，则说明有票，就卖出一张票，同时 A 的值减 1。先假设有余票 10 张（A = 10），事务 T1 和事务 T2 并发修改了 A 的值，执行顺序如图 1 所示。

事务 T1 与 T2 并发地修改数据库内 A 的值，事务 T1 卖票 1 张，事务 T2 卖票 1 张，当 T1 和 T2 两个事务完成后，A 的值应为 8，如果按上图顺序执行，则最后剩余票数为 9，显然不对。

上述操作中，数据库的更新涉及多条纪录、多个环节，尤期是当这些记录处于不同的数据集时，就要用到事务。利用事务可以解决数据库数据的一致性，同时也可避免很多在设计阶段不可预知的问题。

2 多层数据库结构体系

多层结构的典型是三层结构，其基本思想是把用户界面与应用服务器分离。整体结构如图 2 所示。

客户端主要是提供用户接口，它不包含或包含少许部分不重要的企业应用逻辑。企业应用逻辑主要包含在处于中间层的应用程序服务器上。应用程序服务器是应用主体，它掌握数据集定义的全部细节，和远程数据库服务器进行通信。远程数据库服务器是 DBMS，负责管理对数据的读写和维护。

3 多层数据库结构在 Delphi 中的实现方法

在 Delphi 中多层数据库结构的基础是 MIDAS

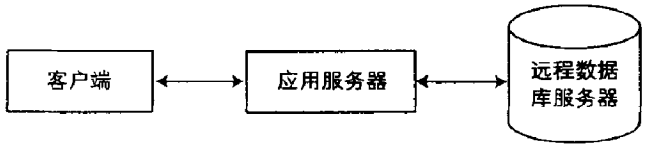

图 2　三层结构模型

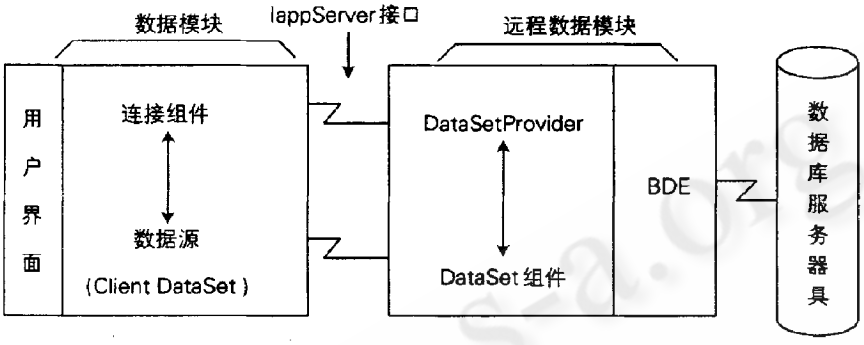

图 3

(Multi－tier Distributed Application Service Suite)技术。图 3MIDAS 的数据库应用程序需要一些特殊组件,这些组件分类如下:

(1) 远程数据模块。位于服务器端,作为 COM 服务器或 CORBA 服务器让客户端应用程序访问它的接口。

(2) DataSetProvider 组件。位于服务器端,提供 IAppServer 接口,客户端应用程序通过 IAppServer 接口获得数据。

(3) 连接组件。位于客户端,为客户端应用程序定位应用程序服务器和 IAppServer 接口。

(4) ClientDataSet 组件。位于客户端,访问服务器端的 IAppServer 接口。客户端应用程序通过 IAppServer 接口与应用程序服务器通讯。

在远程数据模块上,BDE 数据集组件通过 BDE 与远程数据服务器连接访问数据库,DataSetProvider 组件输出 IAppServer 接口,DataSetProvider 组件通过 DataSet 属性与数据集组件相连,这样客户端通过 IAppServer 接口就可以存取数据。

4　Delphi 事务处理的实现

4.1　在客户端利用显式事务处理保证数据一致

参考上面图 3,客户端的 TclientDataSet 组件通过 IAppServer 接口就可以存取数据。在这种结构中,可以在 TclientDataSet 组件的 commandtext 属性中的 SQL 语句里显式进行事务处理。步骤如下:

(1) 客户端的 TclientDataSet 组件的 commandtext 属性中 SQL 语句里含有事务定义。

(2) 该 SQL 语句通过 IAppServer 接口传给应用程序服务器中的数据集组件(如 TQuery 组件)。

(3) 数据集组件通过 BDE 访问远程数据库服务器。并将含有事务处理的 SQL 语句传给该远程数据库服务器。这样就可以直接利用远程数据库服务器的事务处理能力。

以显式使用事务中需要用到 TDataBase 控件。TDataBase 控件有几个重要属性和方法。

- StartTranscation: 开始一个事务
- Commit: 数据提交,更新服务器数据库
- RollBack: 取消该事务对数据库所做的修改,将数据库恢复到事务前状态
- InTransaction: 判断当前进程中是否存在数据库事务

TDataBase 控件实现数据库事务的一般过程是首先调用方法 StartTranscation 开始一个事务,然后对数据库进行读写操作,此时对数据库的操作都与该事务有关。操作完毕调用 Commit 方法确认提交;如果发生错误,调用方法 RollBack 取消该操作,将数据库恢复事务开始前的状态。

基本程序如下:

```
DataBase1. StartTransaction; //开始一个事务
Try
  Table1. open;
  Table1. edit;
    … //记录操作过程
  Table1. post;
  DataBase1. Commit; //提交数据
Except
  DataBase1. RollBack; //提交失败,回滚数据库
  Raise; //如果遇到相同错误,再次引发异常
```

```
End
```

4.2 在中间层隐含数据库事务处理保证数据一致性

在 Delphi 中基于 BDE 体系的三层结构本身提供了隐含事务处理功能。当客户端应用程序向应用程序服务器申请更新数据时，处于中间层上的 Datasetprovider 组件会自动将更新数据的例程加上一层事务的外套——处于事务控制下。如果出错记录没有超过客户端设定的参数值，就向远程数据库服务器正式提交事务，否则滚回。因为这种隐式事务处理功能已经包含事务处理能力，所以不需在客户端或中间层进行显式事务定义。

这里以一个例子说明这种中间层的隐式事务处理。例如在客户端通过数据控制组件编辑、修改好数据，然后调用客户端的 Tclientdataset 组件的 Applyupdates (maxerrors:integer) 函数，则该更新数据的例程就处在应用程序服务器上的 Datasetprovider 组件所加的隐式事务控制下。若更新成功，则所有包含在事务中的操作全都随着 COMMIT 存入数据库服务器上。若更新失败，则所有操作都会随着 ROLLBACK 撤销。

Delphi 中采用隐含事务，基本程序如下：

```
With Table1 do
  Begin
    Open; //打开数据集
    Edit; //数据集编辑方式
    … //记录操作过程
    Post; //数据提交
    End
```

5 结束语

数据库应用系统的好坏取决于其事务处理能力的强弱。本文运用事务处理，针对客户端和中间层的控制方法，可以较好的解决多层数据库结构中的数据一致性和完整性的问题。

参考文献

1 张玉珍，浅析多层结构及其 Delphi 中的实现[J]，计算机工程与设计，2002，23(6)：35－37。

2 张玉珍，在多层应用中利用事务处理中的触发器实现数据完整性[J]，工业控制计算机，2002。

3 Mike Gunderloy Mary Chipman. SQL SERVER 7 轻松进阶[M]，邱仲潘等译，北京 电子工业出版社，1999。

4 新智工作室，Delphi 数据库编程[M]，北京 电子工业出版社，2001。