文章编号：1005-8451 (2012) 02-0055-04

基于Python的SQL Server海量数据转移的研究与实现

张晓博
(西南交通大学 信息科学与技术学院，成都 610031)

摘 要：针对MS SQL Server特定数据库中存放海量数据信息的1张数据表转换成6张数据表来存储数据的问题，考虑到数据被转移的完整性和安全性，采用事务性的存储过程，结合Python语言来解决完成，并建立数据库视图，借助SQL代码测试最终结果。节省了数据库存储资源，实现了一种数据库中存储的海量数据被转移的方法，同时使得用户能够更高效的使用数据库数据。

关键词：存储过程；python；SQL；视图
中图分类号：U29：TP39 **文献标识码**：A

Research and implementation of mass data transferred based on SQL Server of Python

ZHANG Xiao-bo
(School of Information Science & Technology, Southwest Jiaotong University, Chengdu 610031, China)

Abstract: In a certain database of MS SQL Server, there was one table of storing mass data, it was replaced with six tables. In consideration of the integrality and security for those data transferred, this paper used stored procedure with transaction of database and python to solve that problem, also created database view by SQL code to test the result. It could save store resource and implement a method of mass data transferred in database, make use of data of database more effectively.

Key words: stored procedure; python ; SQL; view

本论文是在美国赛门铁克公司安全相应中心的一个项目，其涉及的数据库中[path]表内具有唯一性约束的[path]字段，里面存放在不同服务器上，不同盘符下不同文件夹下面的不同文件的路径及名称数据信息。考虑到相同路径下面的不同文件名称，不同路径下面的相同文件数据信息连接起来被只存放到[path]字段里面所占空间问题，用户对不同文件类型的快速统计需求，更重要的是用户整体数据信息的有效使用，有必要对其进行优化。本文以此为着眼点，结合MS SQL Server 2005和Python2.6.6，实现把存放5千万条不同文件路径信息的1张表转换为6张表，并实现其数据的完整性和安全性转移，最后测试完成结果，提出了一种解决数据库海量数据转移问题的切实可行得有效方法。

收稿日期：2011-05-12
作者简介：张晓博，在读硕士研究生。

1 MS SQL Server数据库设计

1.1 数据库相关知识

MS SQL Server 2005是由Microsoft在MS SQL Server 2000的基础上开发和销售的一款数据库管理系统。通常情况下，一个数据库系统应该具有：数据冗余小、数据完整、数据集成、提供数据共享和安全控制、具备数据独立性等特征[1]。主要设计到的相关数据库概念如下：

(1) 数据库表：数据库中最基本的元素。

(2) 约束：对表中属性的限制，属于静态规则，SQL Server会按照这些多半都是不变或者很少变动的规则强制性地维护数据库的完整性，其中UNIQUE约束确保数据列中没有重复的值。

(3) 索引：主要内容是表中某个属性值与所在记录位置的一一映射，用于提高属性的查找效率。包括简单索引和复杂索引，唯一索引和非唯一索

引，聚集索引和非聚集索引，以及全文索引。其中聚集索引是所有的记录都按顺序聚集在索引结构的叶结点上的索引。而非聚集索引指在叶结点上不包含记录的索引。

(4) 事务：用于将多个语句合并为一个整体并使这个整体具有不可分割和同进同退的特性，具有原子性（Atomicity,即不可分割性）、一致性（Consistency）、隔离性（Isolation）和持久性（Durability）4个属性。

(5) 存储过程：使用SQL语言和SQL扩展并用Transact-SQL编写的一类特殊的批处理，被保存在数据库服务器以提高执行重复任务的性能和一致性。

(6) 视图：一段存储在数据库中的查询语句，可以将其看成是由SELECT查询语句的结果构成的虚拟表。按作用可分为独表的子集和多个表的混合体两类，本文基于第多个表的混合体类。

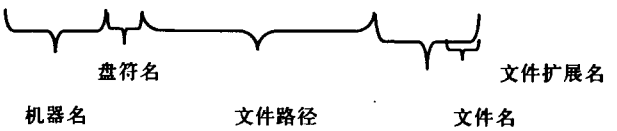
1.2 建立新数据库表

在SQL Server 2005下相应的数据库中是把原来的旧表[path]中[path]的字段内容分成以下5个部分：

机器名；盘符名；文件路径；文件名（带扩展名，方便快速检索）；文件扩展名。

举例说明如下：

VMImage1 C:\WINDOWS\system32\svchost.exe



然后建立6个新表：[machine],[fileExt],[fileName],[dirName],[dirPath]和[pathNew],逐一存放机器名、盘符名、文件路径、文件名和文件扩展名。同时对前表中的[name],[dirPath]中的6个表中的相关字段创建UNIQUE约束；由于数据存储量很大，为满足用户对数据的查询速度，对表中的各个外键添加非聚集索引[2]，表结构及关系如图1。

1.3 建立存储过程

为保证数据被转移的准确性和完整性，利用数据库的事务性特性，给每一个新表创建一个相应的存储过程来插入或更新相应的数据，共6个存储过程，它们的基本算法是一致的，其中[dirPath]表对应的存储过程要调用[machine]和[dirName]表的存储过程，分别插入或更新[machine_id]和[dirName_id],[pathNew]表需要调用[fileExt],[fileName]和[dirPath]表的存储过程，分别插入或更新[fileExt_id],[filename_id]和[dirPath_id]。基本算法流程如图2：

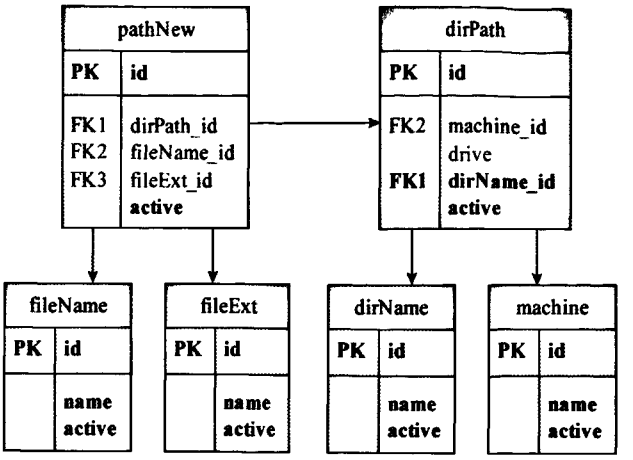


图1 新数据表结构及关系图

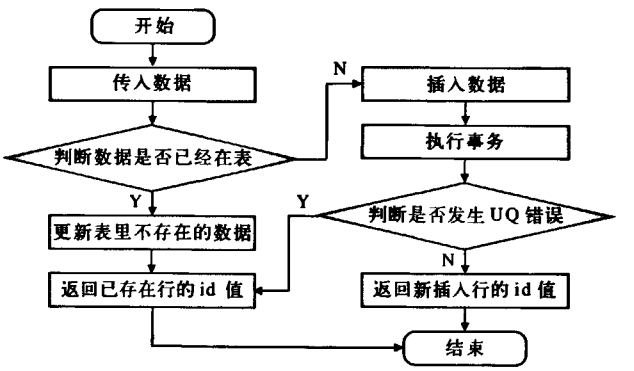


图2 存储过程基本算法流程图

2 Python代码设计

2.1 Python语言技术

Python是一种解释型、面向对象、动态语义、语法优美的脚本语言，自从1989年由Guido Van Rossum设计出来后，经过十余年的发展，已经同Tcl、Perl一起，成为目前应用最广的3种跨平台脚本语言。Python支持现有的各种主流操作系统，如Microsoft Windows、Solaris、Mac OS、Linux等，甚至包括Palm OS这样的嵌入式环境。它的源程序和二进制代码可以免费获得。由于其强大灵活的功能，简洁优美的语法和源代码免费开放，Python被著名国际自由软件项目KDE计划选定为标准系统脚本语言[3]。

与同为脚本语言的Tcl、Perl相比，Python

的特点有：

（1）面向对象：Python提供类,类的继承，类的私有和公有属性，例外处理等完善的对面向对象方法的支持。

（2）虚拟机：像Java一样，Python程序在执行前要先编译成字节码，再通过一个虚拟机解释执行。

（3）高级数据结构：Python内置了对列表，关联数组等常用数据结构的支持。

（4）语法简洁优美：Python的语法非常简单易学，并且采用缩进来表示程序的块层次结构。这样不仅减少不必要的块符号，更重要的是强制程序员用一种清晰统一的风格书写程序，增加了程序的可读性，降低了维护开销。

（5）易于扩展和嵌入：Python语言本身只提供了一个编程语言所需功能的最小内核，其它许多丰富的功能都由扩展模块实现。由于在设计时就考虑到了扩展性，可以很方便地用C或者C++编写Python的扩展模块以添加新的功能，或者把Python解释器自身嵌入到其他程序内部。

2.2 调用存储过程的python语言设计

借助Python语言主要是从数据库获取旧数据表[path]字段的内容；把取出的数据用由Python编写的SplitPath()函数分割成机器名、盘符名、文件路径、文件名和文件扩展名5部分；最后调用已经创建好的存储过程，用那5部分数据作为相对应的参数传入存储过程并执行。其基本算法流程如图3：

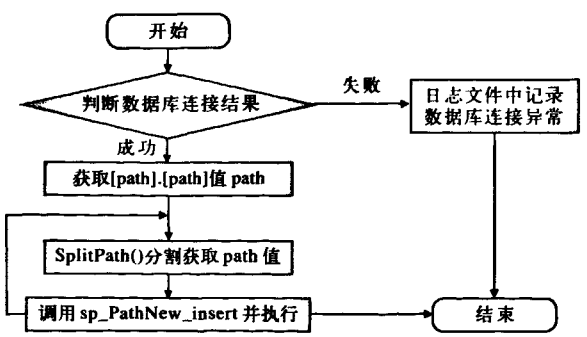


图3　Python代码基本算法流程图

3 海量数据转移的实现与测试

3.1 数据转移的实现

本文所涉及的数据库是在一台CPU为Intel(R)Xeon(R) E5440 @2.83 GHz 2 .83 GHz，RAM是23.9GB DDR并且安装有Python 2.6.6软件的数据库服务器上面，其[path]表存放5千万以上的不同数据，把存储的海量数据转移到新建立的6个表中，用python实现的代码完全自动化执行，大概需要3*24 h的时间，python实现的部分核心代码[4]如下：

```
parametres = splitPath(result1[1])
query = sql_sp_pathDS+ " @pathDSNew_id = ?, @machine = ?, @drive = ?, @dirName = ?, @fileName = ?, @fileExt = ?"
if parametres[2] == '' and parametres[3]!='' :
    if len(parametres[4]) > 250:
            print result1
            print
            m += 1
             params = [result1[0], u'%s' %parametres[0], u'%s' %parametres[1], """",
              u'%s' %parametres[3], '']
    else:
    params = [result1[0], u'%s' %parametres[0], u'%s' %parametres[1], """", u'%s' %parametres[3], u'%s' %parametres[4]]
     db11.execute(query, params)
```

3.2 建立数据库视图并测试转移结果

数据转移完成后，为了测试转移的最终结果，需要把新创建的6个表存储的数据内容与旧表中的数据做比较，就需要创建一个视图（pathNew View）连接6个表，设定与旧表相同的字段（path）[5]。视图的创建代码如下：

```
CREATE VIEW pathNewView
AS
SELECT .id,ISNULL(m.name+' ,'')+ISNULL(d.drive+':','')+ISNULL(dn.name,'')+ISNULL(f.name,'') AS path ,p.active
FROM pathNew p WITH (nolock)
LEFT JOIN fileName f WITH (nolock) ON f.id=p.fileName_id
LEFT JOIN dirPath d WITH (nolock) ON d.id=p.dirPath_id
LEFT JOIN dirName dn WITH (nolock) ON dn.id=d.dirName_id
```

（下转P61）

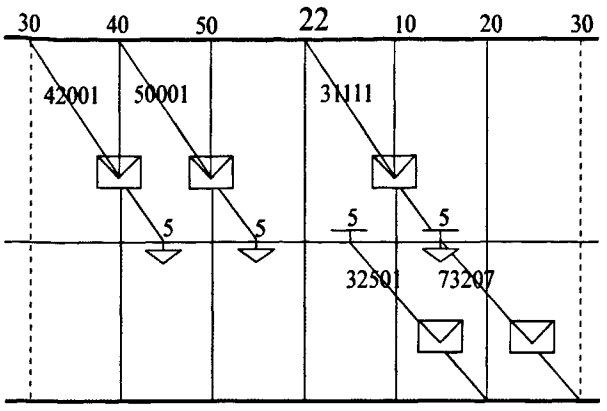


图1　系统动态绘制的运行图

4　结束语

传统的使用图片方式绘图运行图，需要事先绘制，不能随动态数据变化，图片需要从服务器端下载到客户端才能显示，有一定的传输时延，用户的体验性不好，服务器的负担也较大。而采用VML绘制运行图，充分利用了客户浏览器本身的计算功能，绘图工作在客户端完成，服务器压力较小，而且动态绘图能够适应信息的即时变化，方便快捷，实用性较高。

参考文献：

[1] 张雅净，王鹤鸣. 铁路行车调度[M]. 北京：中国铁道出版社，2010.

[2] 欧国明，黄云峰，黄　键，等. 用ASP+VML实现气象图表的快速绘制[J]. 广东气象，2010，32（5）.

责任编辑　方　圆

（上接P57）

LEFT JOIN machine m WITH (nolock) ON m.id=d.machine_id

在上述视图建立好的基础上，然后在Microsoft SQL Server Management Studio下面用SQL语言编写测试代码[6]，具体如下：

```
DECLARE @num_before BIGINT,@num_old BIGINT
SELECT @num_before=count(p.id) FROM pathDS
SELECT @num_old=count(p.id) FROM pathDS p JOIN pathDSView pv ON
pv.id=p.id WHERE p.[path] = pv.[path]
IF @num_before==@num_old PRINT '数据转移成功！'
ELSE PRINT '数据转移失败！'
```

上述SQL代码执行后，如果输出‘数据转移成功！’，证明结果正确；如果输出‘数据转移失败！’，说明结果有误。本文所研究的海量数据转移在实际项目当中结果完全正确。

4　结束语

论文针对数据库优化过程中，数据表结构发生变化的时候，海量数据转移过程中遇到的数据完整性和安全性问题，提出了一种基于Python语言，结合数据库事务特性和存储过程，快速稳定地实现数据库的合理优化，最后建立数据库视图，测试最终结果。因此，这样一种处理数据转移的方法，不仅满足本文项目本身的需要，也对普遍存在的两个不同数据库之间的数据交互处理问题起到一定作用，有助于用户更高效的使用数据库数据资源。

参考文献：

[1] 龙　帅. 深入浅出SQL Server数据库开发[M]. 北京：中国青年电子出版社，2006.

[2]（美）Dusan Petkovic.Microsoft SQL Server 2005初学者指南[M]. 冯　飞，薛　莹，译. 北京：清华大学出版社，2007，66-242.

[3] Martin C.Brown. python技术参考大全[M]. 康　博，译. 北京：清华大学出版社，2002.

[4] Martelli, A.;Alex Martelli. Python in a nutshellPYTHON技术手册[M]，南京：东南大学出版，2006.

[5] 吴溥峰，张玉清. 数据库安全综述[J]. 计算机工程，2006（32）：85-88.

[6] Stefan Brass, Christian Goldberg, Proving the Safety of SQL Queries, Proceedings of the Fifth International Conference on Quality Software (QSIC' 05), 2005.

责任编辑　徐侃春