文章编号:1005-8451(2005)02-0048-03

数据库连接池技术及其在Web系统开发中的应用

崔莹峰,王 洪
(北京交通大学 计算机与信息技术学院,北京 100044)

摘 要:使用传统的数据库连接模式开发Web系统,建立和关闭数据库连接的过程将会严重增加系统开销,成为造成Web系统速度瓶颈的重要原因。使用连接池技术,数据库连接得到了高效、安全的复用,避免了其频繁建立、关闭的开销。介绍数据库连接池的基本原理、工作机制,详细讨论了如何用Java实现数据库连接池,并对连接池在Web系统开发中的作用做出评价。

关键词:数据库连接池;Web系统;管理策略;Java
中图分类号:TP392 **文献标识码**:B

Technology of database connection pool and application of it to developing Web System

CUI Ying-feng, WANG Hong
(School of Computer Science & Information Technology of Beijing Jiaotong University, Beijing 100044, China)

Abstract: Using the traditional database connection mode to develop Web system, the course of opening and closing database connections would increase the systematic expenses seriously, and cause Web systematic tempo bottleneck. With the connection pool, the database connection could be reuse efficiently and safely, and it also could avoid frequently opening and closing. It was introduced basic principle, working mechanism of the connection pool, and was discussed in detail how to implement the connection pool with Java, additionally it also was appraised the impact of the connection pool in developing Web system.

Key words: database connection pool; Web System; management policy; Java

随着企业信息化进程的深入,数据库应用已成为企业信息处理的主要组成部分,当前的任何一个稍具规模的网站,其后台必然涉及到对大型数据库的访问,换句话说,基于对数据库的应用已经成为当前Web系统的主要特征。

一般情况下,在开发基于数据库的Web系统时,传统的模式基本是按以下步骤:

a.在主程序中建立数据库连接;

b.进行SQL操作,对数据库中的对象进行查询、修改和删除等操作;

c.断开数据库连接。

使用这种模式开发,存在很多问题。

(1)要为每一次Web请求(例如查看某一篇文章的内容)建立一次数据库连接并在完成操作之后关闭连接,对于一次或几次操作来讲,或许不怎么影响系统开销。但是,对于影响Web系统来讲,即使在某一较短的时间段内,其操作请求数也远远不是一两次,而是上百次,在这种情况下,系统开销是相当大的。事实上,在一个基于数据库的Web系统中,建立和关闭数据库连接的操作将是系统中的操作之一,这种频繁的建立和关闭连接的过程将会成为Web系统速度瓶颈的重要原因。

(2)使用传统的模式,必须管理每一个连接,确保它们能被正确关闭,如果出现程序异常而导致某些连接未能关闭,将导致数据库系统中的内存泄露,最终我们将不得不重启数据库。

可以发现,之所以会出现以上问题,根本原因是因为传统模式对数据库低效地管理造成的。而数据库连接池恰恰能够为解决这种模式带来的诸种弊端提供一种可靠、有效的管理策略。

使用数据库连接池的优势就在于它可以对数据库连接进行管理,一方面应用程序通过连接池获得连接,即通过与数据库交互直接获得并释放连接,为多个用户提供共享的数据库连接,当一个用户不再使用某个数据库连接时,将连接返回到连接池中。

收稿日期:2004-07-12
作者简介:崔莹峰,在读硕士研究生;王 洪,教授。

1　数据库连接池技术

1.1　基本原理

连接池最简单的思想也就是预先建立一些连接，放置在内存对象中以备使用：当 Jsp、Javabean 和 Servlet 等应用程序需要使用连接时，只需从内存中取一个来用而不需要重新建一个连接；同样，使用完毕后，只需将此连接放回到内存中即可，而连接的建立和断开都由连接池自身来管理。基本原理如图 1 所示。

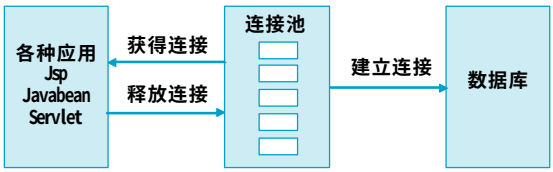

图 1　连接池基本原理

1.2　工作机制

连接池主要由 3 部分组成：连接池的建立、连接池中连接的使用管理、连接池的关闭。

1.2.1　连接池的建立

程序中要建立的其实是一种静态连接池，所谓静态连接池，是指连接池在系统初始化时就已经建立成功，而且不能随意关闭连接。连接池建立时，根据相应的配置，可以一次性建立预定数目的连接对象，这些连接对象作为系统可自由分配的资源，当程序需要使用连接时，直接从连接池里获得，避免了随意建立、释放连接所带来的系统资源。

1.2.2　连接池中连接的使用管理

连接池管理策略是连接池机制的核心。当应用程序需要访问数据库时，不是直接同数据库建立连接，而是向连接池申请一个连接。同样，当应用程序访问数据库完毕，释放连接时，并不是直接关闭连接，而是向连接池释放连接。

该策略定义如下：当用户向连接池请求连接时，先查看池中是否有没被分配的空闲连接，如果存在，则把空闲连接分配给用户，并作相应处理；如果池中没有空闲连接，则等待直到有空闲连接分配给用户，此时该连接被多个用户所复用。当用户释放连接时，唤醒所有等待连接的用户线程并做相应的处理。如果连接释放后没有等待连接的用户线程，则把它重新放回连接池中，并不关闭连接。

1.2.3　连接池的关闭

当应用程序退出时，应关闭连接池，此时应把在连接池建立时申请的连接对象统一归还给数据库（即关闭所有数据库连接），这与连接池的建立正好是一个相反过程。

2　具体实现

很多 Web 服务器提供了数据库连接池技术，如有名的 TOMCAT 和 WEBLOGIC 服务器，这些服务器通过简单的配置，可以实现功能强大，管理方便的连接池。但是对于一般的项目，采用这些服务器配置的连接池固然好，自己开发实现的连接池也能够完全胜任项目要求。（1）因为项目作为一个产品的话，肯定不是只部署在 TOMCAT 或者是 WEBLOGIC 上，从功能上来说，自己写的连接池完全够用；（2）根据不同项目的特点，有针对性实现的连接池应该比 Web 应用服务器提供的通用连接池更适合；（3）使用自己开发的连接池，对于项目小组本身来说，可以提高开发效率，节省开发费用和时间。

通过以上分析和项目的具体需求，我们以 Java 为工具，实现了一个连接池对象对数据库连接的建立、释放进行管理，并把它应用于项目的开发之中。

2.1　描述与实现

这个连接池对象最主要的成员就是 ConnectionPool 类和 ConnectionManager 类。在一个特定的连接池中，所有连接都将连接到同一个 JDBC URI，即同一个主机、数据库实例及登陆 ID。在连接池中单个线程发出数据库连接请求，通过连接池管理程序得到这个连接，这时其他线程就不会得到这个数据库连接，此线程使用结束后，该线程将连接交还给连接池管理程序，以分配给其他等待连接的请求线程。这里，连接池充分利用 JAVA 的线程同步机理，使当前服务线程处于等待状态，直至有空闲的连接出现。当 Java 服务线程对数据库访问完毕后，向连接池释放连接，而不是关闭连接。

ConnectionPool 类的最基本的属性如下：

maxConn：连接池中连接数量；

timeout：等待分配空闲连接的最大时间。

连接池中，最多可同时维持 maxConn 个连接，当 Java 线程访问数据库时，要首先从连接池获得连接，如果没有即时获得，则等待 timeout 时间后，再次向连接池发出请求。

ConnectionPool 类包含的成员方法如下：

getconnection()：获取一个连接；

getconnection(timeout):等待timeout获取一个连接;

freeconnection():释放一个连接;

initialize():连接池初始化;

release():销毁连接池。

ConnectionManager用于管理多个连接池对象,它提供以下功能:

(1)装载和注册JDBC驱动程序;

(2)根据在属性文件中定义的属性创建连接池对象;

(3)实现连接池名字与其实例之间的映射。

2.2 配置文件

配置文件db.properties是用来记录连接数据库的基本信息,当所连接的数据库服务器有改动时,通过重新设置配置文件,可以连接各种数据库服务器.包括Oracle和Sql server.连接池初始化之前.首先读入配置文件.本文以连接Sql server 2000,连接池名称为demo为例,配置文件如下:

```
demo.url=jdbc:microsoft:sqlserver://localhost:1433;
DatabaseName=textbook;//连接的数据库服务器及
数据库
demo.user=sa;//连接数据库服务器用户名
demo.password=sa;//密码
demo.maxcon=10;//连接池中最大连接数量
drivers=com.microsoft.jdbc.sqlserver.SQL
ServerDriver;//数据库服务器JDBC驱动程序
```

2.3 数据库操作

由此将对数据库的基本操作分为5个部分,分别为查询、增加、修改、删除和调用存储过程,对应用Java生成的类分别为SelectBean.InsertBean,UpdateBean.DeleteBean和ExecuteBean.因为建立连接和释放数据库连接的操作在这些基本类中已经实现,当对数据库进行具体操作时,所做的只是找到特定操作所属的基本类,继承就可以了。

2.4 事务处理

所谓事务是用户定义的一个数据库操作序列,这些操作要么全做,要么全不做,是一个不可分割的工作单位。例如,一个事务可以是一条Sql语句、一组Sql语句或整个程序。对于一般的数据库连接,采用上述连接复用的策略会起到良好效果。但是对于具体的一个事务来说,直接使用上述连接就会出现问题,因为它不能控制属于同一个事务的多个数据库操作,有可能这些操作将在不同的连接上实现,而这些连接又会被其它非事务操作所复用共享。因此,连接池必须提供相应的事务连接机制,才可以真正安全高效地实现连接复用。Connection本身提供了对于事务的支持,可以通过设置它的AutoCommit属性为false,显式的调用commit或者rollback方法来实现。采用每一个事务独占一个连接,既可以大大降低对于事务处理的复杂性,又不会妨碍连接的复用,因为隶属于该事务的所有数据库操作都是通过这一个连接完成的。

因此,需要指定事务何时开始,何时结束,并且在事务结束时提交事务。在UpdateBean中使用一个新方法,即excutebatch方法,该方法主要是用来进行事务管理的。当一个事务开始时,继承UpdateBean的对象调用excutebatch方法.获得一个新的数据库连接,并将连接的AutoCommit设定为False,随后的所有SQL语句都是用同一个连接,如果执行SQL语句的过程中发生了异常,程序会自动发出一个回滚申请,以恢复程序对数据库所作的改变;只有所有的SQL语句都正常执行通过,程序才会发出提交申请,接着将连接的AutoCommit设定为True,并将连接返回到连接池中。对于通过继承UpdateBean生成新的类来进行事务管理的开发人员来说,不需要担心在出现异常后处理回滚或关闭连接的工作。

3 结束语

本文主要介绍了数据库连接池的基本原理、工作机制及其实现,另外也谈到了连接池技术在Web开发项目中的应用。采用数据库连接池大大提高了连接被复用的效能,并且提高了项目开发效率。随着Internet的高速发展,与数据库相关的Web应用将会越来越广泛,我们有必要不断提高和改善数据库连接池技术,使其能够更加适应需要,满足要求。

参考文献:

[1] 张 聪,王福川. 连接池模式和Java连接池[M]. 计算机应用,2001.08.)

[2] 飞思科技产品研发中心. Java 2应用开发指南[M]. 北京:电子工业出版社,2002.

[3] 段林海,易小山. 数据库连接池化技术及其JAVABEAN实现[M]. 电脑开发与应用,2002.11.)