Java 中连接池的设计与实现

赵勇超，郑　宁，葛瀛龙
（杭州电子工业学院，浙江 杭州 310037）

摘　要：分析了 Java 传统模式访问数据库存在的缺陷，阐述了连接池的概念以及连接池的具体实现。根据试验结果分析了通过连接池与通过传统模式获得相同连接对象花费的时间对比情况，以及创建一个连接对象平均花费的空间情况。连接池可以广泛地应用在基于数据库的 Web 系统中。

关键词：Java；连接池
中图法分类号：TP311.52　　文献标识码：A　　文章编号：1001-3695(2004)06-0219-03

Design and Deploy of the Connection Pool in Java

ZHAO Yong-chao, ZHENG Ning, GE Ying-long
(*Hangzhou College of Electronics Engineering, Hangzhou Zhejiang* 310037, *China*)

Abstract: The paper analyses the disadvantage of traditional method of accessing databases in Java. Also, it explains the conception and implementation of the connection pool. Then, it compares the connection pool with traditional methods in consumption of time to get the same connection objects and analyses that how much average space is spent to create a connection object. Finally, it brings forward that connection pool can apply in Web system based on database extensively.

Key words: Java; Connection Pool

随着 Internet/Intranet 的发展，计算机领域在过去几年中发生了翻天覆地的变化。计算机的应用已经从桌面应用转变到 Web 应用。这些 Web 应用包括：通过 Internet 发送电子邮件；通过 Internet 实现世界各地的信息共享；通过 Internet 实现网上交易等。随着 Internet 的普及，在网上处理的数据成倍增加，而且越来越复杂，迫使科学研究者和应用开发者必须考虑如何快速建立完善有效的新模型以及如何改进和优化一些传统模型。众所周知，在一个基于数据库的 Web 系统中，建立数据库连接的操作是系统中代价最大的操作之一。很多时候，可能网站速度瓶颈就在于此。为此，我们对 Java 中访问数据库传统模型进行了深入研究，并提出了在传统模型中引入连接池技术的方法，从而很好地解决了这个问题。连接池技术是通过重用一组连接对象，使所有用户之间共享一组已经打开的连接。试验证明，我们提出的方法可以节省建立数据库连接的时间，提高访问数据库的速度。

1　Java 应用程序访问数据库基本原理

Java 语言的跨平台性、可移植性以及安全性等特性使其成为开发数据库的一种优秀语言。JDBC(Java DataBase Connectivity)是 Java 应用程序与数据库沟通的桥梁。JDBC 主要提供两种 API，分别是面向开发人员的 JDBC API 和面向底层的 JDBC 驱动程序 API。底层主要通过两种模式，即 JDBC-ODBC 桥驱动和直接的 JDBC 驱动与数据库实现连接。

一般情况下，Java 应用程序访问数据库传统模式基本是按以下步骤：①在主程序中通过 JDBC 建立数据库连接；②进行 SQL 操作，访问数据库，存入、取出数据；③断开数据库连接。

在基于数据库的 Web 系统中，在某一较短的时间段内，只有少数 Web 请求时，传统模式还能很好的工作，但随着请求数不断增加，系统的开销越来越大，响应 Web 请求的速度越来越慢，直到无法响应 Web 请求。造成这种结果的原因是由于传统模式存在下面的一些缺陷：①一次 Web 请求都需要建立一次数据库连接。每建立一次数据库连接就需要花费 0.05s ~ 1s 的时间，还要花费很大的系统开销；②不能控制被创建的连接对象数，系统资源被毫无顾忌的开销，最后导致系统内存溢出，服务器崩溃；③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，最终将不得不重启数据库。

2　在传统模式中引入连接池

2.1　连接池的基本概念

连接池(Connection Pool)顾名思义，就是众多连接对象的“缓冲存储池”，也即是连接对象的集合体。连接池内部提供一种管理机制，能控制连接池内部连接对象的个数，对应用程序提供获取和释放连接的接口。

2.2　连接池的管理模式

定义：允许创建的最大连接对象数为 M(常数)，已创建的连接对象数为 n，连接池中连接对象为 m。初始值 n=0，m=0。

2.3　Java 中通过连接池访问数据库的模式

在引入了连接池以后，绝大部分访问数据库的请求并不需要新建连接，只需到连接池中取出一个空闲连接就可以了。一般情况下，Java 中通过连接池访问数据库需要以下四个步骤：

收稿日期：2003-03-13；修返日期：2003-06-22

①获得对连接池的唯一实例的引用;②从连接池中获得一个连接对象;③使用连接对象访问数据库;④将连接对象放回到连接池。

图 1 是通过连接池访问数据库的流程图。

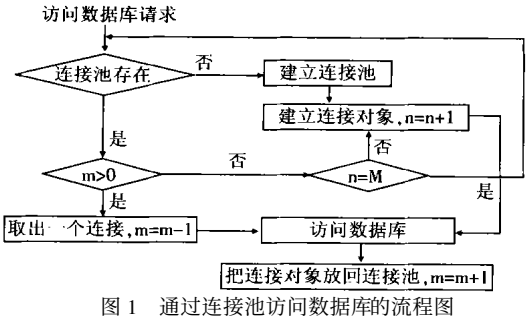

图 1 通过连接池访问数据库的流程图

当传统模式中引入连接池后,访问数据库的模式有了改变。图 2 是通过连接池访问数据库的结构模型图。

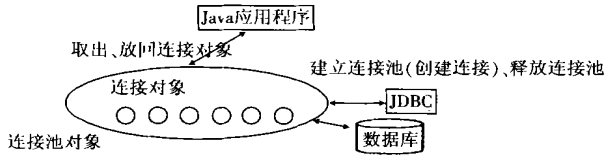

图 2 Java 通过连接池访问数据库结构模型图

3 连接池的实现

3.1 连接池的模型

通过 Connection Pool 类来实现连接池模型,表 1 是 Connection Pool 实现的连接池模型。

表 1 Connection Pool 实现的连接池模型

ConnectionPool	
instance: Connection Pool	连接池的唯一实例
clients: int	表示已创建的连接对象个数
freeConnections: Vector	用来"存储"连接对象
driver: Driver	表示 JDBC 的驱动程序
max: int	表示允许创建的连接对象的最大个数
getInstance: Connection Pool	取出唯一连接池实例
init()	连接池初始化
getConnection: Connection	从连接池中取出一个连接对象
freeConnection: void	把连接对象放到连接池
release: void	释放连接池和其中的所有连接对象

3.2 XML 配置表

为了实现连接池,必须做一些准备工作,首先需要建立一张 XML 表来配置一些所需的参数和相关属性:① connections 创建 Connection 对象所需的参数名,值为 url: user: password。其中,url: 提供一种标志数据库的方法;user: 数据库用户名;password: 数据库用户密码。② drivers 创建 Connection 对象所需的 JDBC 驱动程序的参数名。③max 允许创建的最大连接对象数。

下面是 XML 配置表(config.xml):

```
<?xml version="1.0" standalone="yes"?>
<config
 connections="标志数据库: 数据库用户名: 数据库用户密码"
 drivers="标志 JDBC 驱动程序"
 max="允许创建的最大连接对象数">
</config>
```

3.3 解析 XML 配置表

为了连接池实例和连接对象能得到所需要的参数,必须先构造一个解析方法来解析 XML 配置表。这个方法为 String getConfig(String)。对于这个方法,只要传入 XML 配置表中的 config 元素的一个属性名作为参数,就可以得到对应的属性值。本文是用 JAXP 包中的 DOM 来解析 XML 配置文件的。下面是实现 getConfig 的过程:

(1)建立一个解析器工厂

```
DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
```

(2)当获得一个工厂对象后,使用它的静态方法 newDocumentBuilder()方法可以获得一个 DocumentBuilder 对象。这个对象代表了具体的 DOM 解析器。但具体是哪一种解析器,微软的或者 IBM 的,对于程序而言并不重要。

```
DocumentBuilder db = dbf.newDocumentBuilder();
```

(3)然后,就可利用这个解析器来对 XML 文档进行解析

```
Document doc = db.parse("config.xml");
```

DocumentBuilder 的 parse()方法接收一个 XML 文档名作为输入参数,返回一个 Document 对象。

(4)接着,获得 config 这个元素对象

```
NodeList links =doc.getElementsByTagName("config");
Element link=(Element) links.item(0);
```

使用 Document 对象的 getElementsByTagName()方法,可以得到一个 NodeList 对象,再使用 NodeList 对象的 item()方法得到列表中的每一个 Node 对象,在这里得到第一个对象(因为只有一个对象),然后把 Node 对象造型为 Element 对象。

(5)传入属性名得到相应的属性值

```
属性值=link.getAttribute(属性名)
```

调用 Element 对象的方法 getAttribute 得到属性值。

3.4 连接池中的主要方法实现

已经构造了配置表,解析了配置表,那么准备工作已经完成。下面是连接池的一些主要方法的实现:

(1)获得唯一连接池实例 getInstance

```
if(Instance != null) then{ return Instance; }//返回唯一实例
else{
Instance=new ConnectionPool(); // 创建新连接池
return Instance;
}
```

(2)连接池初始化 init()

```
String drv= getConfig("drivers");
//从配置表中取得 JDBC 驱动程序名
driver = (Driver) Class.forName(drv).newInstance();
//得到 JDBC 驱动程序实例
max= Integer.valueOf(xp.getConfig("max")).intValue();
//从配置表中取得允许创建的连接对象的最大值
```

(3)获得连接对象 getConnectin

```
if (freeConnections.size() > 0) {
//判断连接池中是否有连接对象
con = (Connection) freeConnections.firstElement();
freeConnections.removeElementAt(0);
//从连接池中取得一个连接对象
}else{
if(clients< max){//已创建的连接对象数是否小于允许的最大值
return newConnection; // 返回一个新创建的连接对象
}else{return null;}
}
```

(4)把连接对象放回到连接池 freeConnection

```
freeConnections.addElement(连接对象); // 连接对象放到连接池
```

(5)释放连接池和所有的连接对象 release

```
Enumeration allConnections = freeConnections.elements();
while (allConnections.hasMoreElements()) {
Connection con = (Connection) allConnections.nextElement();
con.close();
} //关闭所有未关闭的连接池
freeConnections.removeAllElements();
//释放连接池中所有连接对象
DriverManager.deregisterDriver(driver); //注销 JDBC 驱动程序
```

4　实现连接池所使用的关键技术

本文的连接池主要是用 Java 语言实现的,它还用到了一些其他关键技术:XML 技术、DOM 技术、JDBC 技术等。

(1)XML 技术。它是一系列规则的集合。它非常易于使用、易于由计算机读取、易于调试,而且易于创建适用于各行业的可扩展标记语言。它和 Java 语言一样也是跨平台的,它们能很好的协调、相互合作。在实现连接池的过程中起的作用主要是:用来实现一个配置表,提供连接池所需要的一些属性。

(2)DOM 技术。DOM 是 Document Object Model 的缩写,即文档对象模型。XML 将数据组织为一棵树,然而,DOM 就是对这棵树的一个对象描述。就是通过解析 XML 文档,为 XML 文档在逻辑上建立一个树模型,树的节点是一个个对象。我们通过存取这些对象就能够存取 XML 文档的内容。在实现连接池的过程中起的作用主要是:用来解析 XML 配置表。

(3)JDBC 技术。为 Java 程序提供一些访问数据库的方法。在实现连接池的过程中起的作用主要是:用来注册数据库驱动程序和创建连接对象。

5　试验结果

5.1　试验环境

(1)硬件环境:CPU PⅢ　450;内存 256MB。

(2)软件环境操作系统 Windows 2000 Professional。

(3)开发平台:JBuilder。

(4)运行环境:JDK1.31。

(5)测试环境:JUnit。

5.2　试验结果

表 2 是传统模型下(无连接池)创建连接对象的试验结果。其中,n 表示创建并返回给用户的连接数;t 表示花费的时间。n 单位:个;t 单位:秒(s)。

表 2　传统模型下创建连接对象的试验结果

n	1	10	100	500	1000	5000
t	0.856	1.356	6.534	28.653	55.665	270.899

创建一个连接对象需要的平均时间大约为

T=(270.899-0.856)/(5000-1)=0.054(s)

表 3 是从连接池中取出连接对象,然后放回连接对象的试验结果。其中,n 表示取出放回的连接对象的个数;t 表示花费的时间。n 单位:个;t 单位:秒(s)。

表 3　取出连接对象,然后放回连接对象的试验结果

n	1	10	100	1000	10000	100000	1000000
m	0.991	1.004	1.041	1.183	1.395	4.567	30.952

取出一个连接并放回这个连接平均花费的时间大约为

T=(30.952-0.991)/1000000=0.00003(s)

表 4 是连接池中创建连接对象的内存耗费情况。其中,n 表示连接对象数;x 表示测试开始时内存耗费值;y 表示测试过程中内存的最大耗费值。n 单位:个;x,y 单位:k 字节。

表 4　连接池中创建连接对象的内存耗费情况

n	100	300	500	700	900	1100
x	131724	131700	131980	131732	133980	131748
y	149000	166360	184960	202984	220864	溢出

从表 4 中试验数据可以计算出创建一个连接对象耗费的内存大约为 M=88kB

传统模式创建连接对象不需要创建连接池实例,其他内存耗费情况和通过连接池创建连接对象的内存耗费情况是一样的。这是由于创建连接对象都是创建一个 JDBC 中的 Connection 对象和注册一个 JDBC 的驱动程序。创建一个连接池对象需要 20kB 左右。图 3 是通过传统模型和通过连接池获得连接对象的花费的时间、空间图。

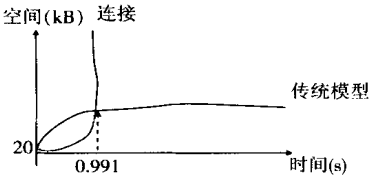

图 3　连接对象的花费的时间、空间图

5.3　试验结果分析

对 5.2 节所述的试验结果进行分析如下:

(1)获得连接对象的个数小于 10 时,传统模式和引入连接池后的模式花费的时间相差不大。

(2)获得的连接对象的个数大于 10 时,传统模式花费的时间远大于引入连接池后的模式花费的时间。

(3)创建一个连接池对象花费空间大约为 20kB,创建一个连接对象花费的空间大约为 88kB。

从上面的分析可以得出下列结论:①如果在同一时间访问数据库的请求超过 10 个,那么就需要引入连接池了。②假定允许创建的最大连接对象数为 M,那么创建这么多连接对象耗费的空间为 88MkB。因此 M 的值应该根据系统资源来决定。如果系统的内存空间为 C(k),操作系统和其他的应用程序用去 D(k),需要预留的空间为 E(k),那么 M 的取值应该小于(C-D-E)/88。其中,C 的值可通过计算机中的内存条知道,D 的值可等到操作系统和其他应用程序启动好后,从操作系统的任务管理器的性能里面获得内存的使用情况。对于 E 的值,只能根据对以后内存使用情况的预测和一些经验来确定了。首先,每次访问数据库都需要创建一个 ConnectionPool 对象、一个 Statement 对象和一个 ResultSet 对象(结果集),通过经验给它们和还有一些其他需要创建的对象留 10000kB~15000kB 的空间,最后,根据经验,当内存还只剩 15000kB 左右的空间时,Java 程序就会出现内存溢出的异常,所以要为此留出 15000kB 左右的空间。这样 E=25000kB~30000kB。

6　结束语

在一个基于数据库的 Web 系统中,建立数据库连接的操作成为了最耗系统资源的操作之一,它越来越成为网站的瓶颈之一。本文提出引入连接池技术的方法很好地解决了传统模式中每次访问数据库都需要建立连接对象这个弊端。连接池就是让所有访问数据库的请求共享一组已经　(下转第 224 页)

关人员修改、删除图像资料，未经授权的人员不能访问中心数据库。

(2)窗口管理模块。控制图像播放窗口的建立、切换和屏幕缩放等。

(3)本地(Client 端)参数设置模块。设备监控终端上每一个播放窗口所对应的视频服务器的 IP 地址和通道号、云台解码器类型及地址、是否进行本地循环录像和设置自动录像时间表等。

(4)视频处理模块。它包括图 4 所示的几个功能。多画面实时监控功能按照 1/4/9/16 画面格式实时播放指定通道的图像；本地录像功能是把指定通道的图像数据以文件形式保存在本地硬盘上，并把该文件的索引信息(如 IP 地址、通道号、录像起止时间等)写入监控系统的中心数据库；本地回放功能可根据索引信息找出相应的图像文件，用 Windows Media Player 进行回放；屏幕抓图功能可抓取当前活动窗口的图像并保存为 BMP 或 JPEG 格式的文件；远程文件下载功能按 IP 地址、通道号、录像起止时间等信息访问指定视频服务器中存储的图像文件，并下载到本地硬盘上，文件的索引信息同时写入中心数据库；远程文件回放功能则直接播放指定的视频服务器中存储的图像文件而不下载。

(5)事件日志模块。以辅助线程的方式保存 Client 端运行异常记录以及 Server 端传来的报警记录。

(6)串口控制模块。通过网络实现 Server 端的串口功能。

(7)数据库管理模块。建立并维护 DVSS 操作人员信息、图像文件的索引信息、视频设备及其监控点信息、远程自动录像计划表、本地自动录像计划表等。

(8)远程参数设置模块。在 Client 端通过网络设置远程视频服务器的全局参数、通道参数和报警参数设置，与 Server 端的相关功能对应。

(9)通信模块。它负责与视频服务器或其他监控终端建立通信链接并进行信息传输。

4.3 DVSS 软件开发环境

MPEG-4 视频服务器运行嵌入式操作系统 VxWorks，监控终端运行 Windows NT 操作系统。Server 端程序代码和配套的软件开发包(Software Development Kit，SDK)均用 C 语言编制。SDK 面向各种应用程序的二次开发人员，包含充足的变量和函数，可以实现对视频服务器的全面控制。Client 端程序以 SDK 为基础，在 Visual C++ 6.0 环境下开发，直接面向最终用户。DVSS 的中心数据库用 SQL Server 2000 建立和维护，Client 端程序通过 ODBC 对它进行访问。

5 结束语

DVSS 充分利用了嵌入式体系结构和 Internet 技术，具有系统精简、运行可靠、响应速度快和监控范围广的优点。MPEG-4 数字视频服务器的运行故障率远低于 PC 机加视频卡的集成式现场装置。在相同的网络环境下监视相同数量的画面，DVSS 视频主机上图像的显示时间只比实际发生时间滞后 0.5s，而采用集成式现场装置时，显示时间滞后 2.5s。

DVSS 的另一特点是具有较强的可扩展性，MPEG-4 视频服务器提供了视频、音频、DI/DO、RS-232、RS-485 和 RJ45 等多种信号接口和通信端口，可连接各种报警装置、传感器和执行机构。DVSS 的软件体系具有开放性，兼顾了门禁控制、环境变量监测、组合报警和安防联动的需求，为各种安防系统的融合打下了基础。

参考文献：

[1] 芮雨，余松煜. 基于 VxWorks 的视频监控系统[J]. 电视技术，2000，(12)：69-71.

[2] 钟玉琢. 基于对象的多媒体数据压缩编码国际标准 MPEG-4 及其校验模型[M]. 北京：科学出版社，2000.

[3] 周小四，王淑华，杨杰. 数字图像网络报警系统设计[J]. 计算机工程，2002，28(4)：61-64.

[4] 孔祥营，柏桂枝. 嵌入式实时操作系统 VxWorks 及其开发环境 Tornado[M]. 北京：中国电力出版社，2001.

[5] Lixue. Video Multicast over the Internet[J]. IEEE Network，1999，13(2)：46-60.

[6] Myung-Ki Shin，Jae-Yong Lee. The RTMW Application：Bring Multicast Audio/Video to the Web[J]. Computer Networks and ISDN System，1998，(30)：685-687.

[7] Uytterhoven. Digital Video Processing[J]. Journal of Computational and Applied Mathematics，1996，66(1-2)：N5-N6.

[8] W Holfoder. Interactive Remote Recording and Playback of Multicast Video Conference[J]. Computer Communication，1998，21(15)：1285-1294.

[9] Liu Ke，Mu Zhichun，Wang Zhong. High Performance Speech Compression System[J]. Journal of University of Science and Technology Beijing(English Edition)，2001，8(3)：229-234.

作者简介：

刘灿(1979-)，女，山东邹县人，硕士研究生，主要研究方向为数字视频监控系统、计算机网络、面向对象程序设计；刘克(1965-)，男，河南郑州人，教授，博士生导师，主要研究方向为离散系统和混杂系统的建模与分析、智能信息处理、计算机网络应用等；刘春瑞(1970-)，女，山西清徐人，硕士研究生，主要研究方向为网络数字视频监控系统的开发与应用；孙志刚(1978-)，男，黑龙江集贤县人，硕士研究生，主要研究方向为计算机网络应用与通信技术。

(上接第 221 页)打开的连接对象，从而使几乎所有的访问数据库的请求都不需要花费创建连接对象的时间，这样就可节省创建连接对象时所需要的时间，加快了访问数据库的速度。另外，通过设定允许创建的最大连接对象数的值来控制系统开销，合理地利用系统资源，解决了传统模型中不能控制系统开销这个弊端。笔者在参与基于 Web 的全国性考试考务管理系统开发中引入并运用连接池的技术，大大提高了访问数据库的速度。因此，连接池可广泛应用在基于数据库的 Web 系统中。

参考文献：

[1] SUN Microsystems Inc. Document Object Model[EB/OL]. http://java.sun.com/webservices/docs/1.0/tutorial/doc/JAXPDOM.html，2001-2002.

[2] 伊晓强. J2EE 全实例教程[M]. 北京：北京希望电子出版社，2002.

[3] Danny Ayers，et al. Java 服务器高级编程[M]. 北京：机械工业出版社，2001.

[4] Heather Williamson. XML 技术大全[M]. 北京：机械工业出版社，2002.

作者简介：

赵勇超(1980-)，男，硕士，研究生，主要研究方向为电子商务；郑宁，男，研究员，教授，硕士生导师，主要研究方向为 ICCAD、电信网管、电子商务；葛瀛龙，男，研究生，主要研究方向为电子商务、电信网管。