**数据库连接池基础**

* 没有连接池的数据库连接方式指通过DriverManager和基本实现DataSource进行连接,但它相关连接的建立以及关闭是非常耗时的.
* 如果使用连接池，将有池来管理相关的数据库连接，减少对数据库连接操作.
* 连接池所做的操作，除了管理连接，还有就是对数据库jdbc api的封装，但***jdbc api才是根本***，外面做的都是包装,再花哨都是假的。

**c3p0统一概念**

  checkout :从池中取得可用的连接

   checkoutconnection:被使用的连接

   checkin:把连接放回池中

   checkinconnection:没有被使用的连接

   所有超时设置，相关的连接，是物理连接的关闭，而不是连接返回池中

***管理的是pooledconnection，而不是物理的connection***

   pooledconnection是sun针对连接池的接口，它本身包含connection,和这个connection相关的所有statement,result,一个checkout的connection所作的所有数据库操作，都被pooledconnection所管理.

   statement缓存，主要针对PreparedStatement和CallableStatement,statment缓存主要相对一个connection来说的,不同connection的statment不能通用.

**c3p0行为**

**生成一个connnection**

1. 当池中connection没有到达最大数,当有请求出现,将会产生connection.
2. 成生一个pooledconnection
3. 通过pooledconnection.getConnection()得到连接(得到连接是newProxyConnection,不是物理连接)

**checkin connection**

1. 与pooledconnection脱离关系
2. 关闭与这个connection相关的resultset
3. 关闭所有没有缓冲的statement.
4. checkin所有缓存的statement.
5. 修改pooledconnection相关信息

**checkout connection**

1. 查看池中是否有没有使用的connection,有就返回
2. 没有，如果没有达到最大数，就生成一个，或者就等待

**c3p0常用配置属性**  
**automaticTestTable**

automaticTestTable作为测试connection是否有效的表，如果表存在，但有记录，抛出错误，如果表不存在，则建立，并使用***SELECT \* FROM automaticTestTable*** 作为连接测试语句

如果automaticTestTable没有设置，而preferredTestQuery设置，则使用preferredTestQuery作为连接测试语句

**checkoutTimeout**

从池中拿未使用的连接，超时设置，如果没有设置，就不超时.

**numConnections**

表明池中有多少个连接

**numIdleConnections**

表明池中有多少个空闲连接，它们可以被checkout

**numBusyConnections**

表明池中有多少个被checkout的连接，记住：

***numIdleConnections + numBusyConnections = numConnections***

**numUnclosedOrphanedConnections**

都是checkoutconnection,但他们已经不再池中管理了.当他们checkin时候，将被destory

**connectionCustomizerClassName**

hook方法，在对相关资源做操作的时候，''他所操作的connection是真实的数据库连接，而不是proxy过的connection''

**maxIdleTime**

在checkout一个connection时候，判断这个connection没有被使用的时间是否大于maxIdleTime,来决定是关闭它，还是被checkout

**maxConnectionAge**

设置一个连接在池中最长的时间，如果时间超过，将会从池中清除

**testConnectionOnCheckout**

如果设置为true,每次从池中取一个连接，将做一下测试，使用automaticTestTable 或者 preferredTestQuery,做一条查询语句.看看连接好不好用，不好用，就关闭它，重新从池中拿一个.

**unreturnedConnectionTimeout**

***一个checkout连接的超时设置，一旦一个checkout连接超时，他将物理的关闭，而不是返回池中，主要是防止连接被长期使用不释放，这个设置也是比较危险的***

**idleConnectionTestPeriod**

设置在池中的没有被使用的连接，是否定时做测试，看看这个连接还可以用吗？

**maxStatements,maxStatementsPerConnection**

缓存statement,一个全局的，一个是针对每一个connection,个人觉得效果不是很大,而且也使用了反射机制.

**c3p0 jconsole说明**

* sampleThreadPoolStackTraces:打印出当前c3p0线程池的情况，默认是3个线程,c3p0很多行为异步，放到线程中做的，比如checkout,checkin,close操作，还有内部池重新整理
* sampleThreadPoolStatus:打印出当前c3p0线程池堆栈
* softResetDefaultUser:关闭所有checkinconnection,重新初始化池
* hardReset:关闭所有checkinconnection和checkoutconnection,池这个对象也不要了，全是新的.
* close:关闭所有跟c3p0相关的东西

**源代码分析**

**生成eclipse项目**

* 从sourceforge下载我们目前使用的0.9.1.2版本[http://nchc.dl.sourceforge.net/sourceforge/c3p0/c3p0-0.9.1.2.src.zip 下载]
* ant codegen(因为它有很自动生成代码)
* 导入eclipse(source包括src/classes,build/codegen,缺少juit.jar,log4j.jar,自己解决)

**相关概念**

***首先对datasource的理解，你可以把认为是factory,这样会好理解一点***

* PooledDataSource

  默认情况情况下，PooledDataSource只管理一个连接池(getConnection()的时候),如果你使用getConnection(username,password),而不是默认的username,  
 将会再生产一个连接池针对这个特定的用户,它包含一个ConnectionPoolDataSource实现，连接就是从ConnectionPoolDataSource得到的.

* ConnectionPoolDataSource

  包名是javax.sql,一看就知道是sun定制的接口,表现出一个连接池,是PooledConnection的工厂

* PooledConnection

  包名是javax.sql,也是sun定制的接口.c3p0默认的实现是NewPooledConnection

* Connection,Statement,Result

  操作数据库相关接口，在c3p0中对于NewProxyConnection,NewProxyStatement,NewProxyResultSet,这些东西统一被PooledConnection管理。

**c3p0项目情况**  
c3p0是现在用的最多连接池之一，这么成功的项目却只是一个人开发的。

当目标很明确(连接池要做什么，目标是非常明确的),使用场景很普通的时候,项目能成功，完全求决于程序的架构.

项目在jmx管理和本身死锁监测，做的都比较精彩，但也有它不足支持，jdk1.5提供了很多功能(比如多线程)，它很多都是自己实现了,这就要看作者怎么对待这个项目

目前给我的感觉有点象当年的dom4j感觉，毕竟都需要谋生,创作激情会下降的.