[数据库连接池的作用以及配置](http://blog.csdn.net/xilangyuyun/article/details/52800380)

连接池是创建和管理多个连接的一种技术，这些连接可被需要使用它们的任何线程使用。连接池技术基于下述事实：对于大多数应用程序，当它们正在处理通常需要数毫秒完成的事务时，仅需要能够访问JDBC连接的1个线程。未处理事务时，连接处于闲置状态。使用连接池，允许其他线程使用闲置连接来执行有用的任务。事实上，当某一线程需要用JDBC在[**MySQL**](http://lib.csdn.net/base/mysql)或其他[**数据库**](http://lib.csdn.net/base/mysql)上执行操作时，需要用到由连接池提供的连接。使用连接完成线程后，线程会将连接返回给连接池，以便该连接能够被其他需要使用连接的线程使用。从连接池“借出”连接时，该连接仅供请求它的线程使用。从编程观点看，其效果等同于每次需要JDBC连接时调用DriverManager.getConnection()，但是，采用连接池技术，可通过使用新的或已有的连接结束线程。连接池技术能显著增加[**Java**](http://lib.csdn.net/base/javase)应用程序的性能，同时还能降低资源使用率。

**连接池技术的主要优点包括：**

（1） 缩短了连接创建时间

        创建新的JDBC连接会导致联网操作和一定的JDBC驱动开销，如果这类连接是“循环”使用的，使用该方式，可避免这类不利因素。

（2）简化的编程模型

        使用连接池技术时，每个单独线程能够像创建了自己的JDBC连接那样进行操作，从而允许使用直接的JDBC编程技术。

（3）受控的资源使用

        如果不使用连接池技术，而是在每次需要时为线程创建新的连接，那么应用程序的资源使用将十分浪费，而且在负载较重的情况下会导致无法预期的结果。

注意，与数据库的每个连接均会在客户端和服务器端造成一定的开销（CPU、关联转换等）。每个连接均会对应用程序和数据库服务器的可用资源带来一定的限制。无论连接是否执行任何有用的任务，仍将使用这些资源中的相当一部分。

        连接池能够使性能最大化，同时还能将资源利用控制在一定的水平之下，如果超过该水平，应用程序将崩溃而不仅仅是变慢。

        幸运的是，Sun公司通过JDBC-2.0“可选”接口，完成了JDBC中连接池概念的标准化实施，所有主要应用服务器均实施了能够与MySQL Connector/J一起良好工作的这类API。

通常，你可以在应用服务器的配置文件中配置连接池，并通过Java命名和目录接口（JNDI）访问它。使用连接池时需要牢记的最重要事项是，无论在代码中出现了什么（异常、控制流等），连接以及由连接创建的任何部分（语句、结果集等）均应被关闭，以便能再次使用它们。如不然，它们将纠缠在一起，在最好的情况下，意味着它们所代表的数据库服务器资源（缓冲区、锁定、套接字等）可能会捆绑一段时间，在最坏的情况下，可能会导致永久捆绑。

**连接池的最佳大小是什么？**

        取决于具体情况。尽管最佳大小取决与预期的负载和平均的数据库事务时间，最佳的连接池大小小于你的预期。例如，如果使用的是Sun公司的[**Java**](http://lib.csdn.net/base/java)Petstore Blueprint应用程序，对于包含15～20个连接的连接池，使用MySQL和Tomcat，在可接受的相应时间下，可服务于中等程度的负载（600个并发用户）。要想确定用于应用程序的连接池大小，应使用诸如Apache Jmeter或The Grinder等工具创建负载[**测试**](http://lib.csdn.net/base/softwaretest)脚本，并对应用程序进行负载测试。确定出发点的一种简单方法是，将连接池的最大连接数配置为“无限”，运行负载测试，并测量最大的并发连接数。随后，应进行反向操作，确定出使应用程序具有最佳性能的连接池的最小和最大值。

**连接池与数据源区别？**

        数据库连接池是在应用程序启动时建立足够的数据库连接，并将这些连接组成一个连接池，由应用程序动态地对池中的连接进行申请、使用和释放。对于多于连接池中连接数的并发请求，应在请求队列中排队等待。并且应用程序可根据池中连接的使用率，动态增加或减少池中的连接数。当关闭连接操作时，连接并不真正的关闭，而是返回到连接池中作为空闲连接在后面继续使用，连接池技术解决了数据库连接频繁打开关闭所带来的性能问题。

        有了连接池，我们没必要直接找数据源打交道了，连接池在你的程序所在的机器内存，数据源不一定，并且数据源和连接池会保持一定数量的连接，这样我们访问数据库的时候就不需要找数据源要连接，直接在本地内存中取得连接，可以提高程序的性能。连结池的存在是为了效率，因为实例化一个连接很耗费资源，而连接又有可重用的特征，所以可以把一定数量的连接放在连接池里面以提高效率。

        下面是详细的配置：  
**tomcat中配置数据库连接池**：

**[html]** [view plain](http://blog.csdn.net/xilangyuyun/article/details/52800380) [copy](http://blog.csdn.net/xilangyuyun/article/details/52800380)

[在CODE上查看代码片](https://code.csdn.net/snippets/1925860)

1. **<Resource**
2. name="jdbc/test"
3. auth="Container"
4. factory="org.apache.naming.factory.BeanFactory"
5. type="com.mchange.v2.c3p0.ComboPooledDataSource"
6. description="DB Connection"
7. driverClass="oracle.jdbc.driver.OracleDriver"
8. user="root"
9. password="123456"
10. jdbcUrl="jdbc:oracle:thin:@127.0.0.1:1521:orcl"
11. maxPoolSize="4"
12. minPoolSize="2"
13. acquireIncrement="1"
14. **/>**

1、proxool连接池  
在WEB-INF/classes/下建立文件：Proxool.properties  
文件内容如下：

**[html]** [view plain](http://blog.csdn.net/xilangyuyun/article/details/52800380) [copy](http://blog.csdn.net/xilangyuyun/article/details/52800380)

[在CODE上查看代码片](https://code.csdn.net/snippets/1925860)

1. proxool.alias=test
2. proxool.driver-class=oracle.jdbc.driver.OracleDriver
3. proxool.driver-url=jdbc:oracle:thin:@127.0.0.1:1521:orcl
4. user=root
5. password=123456
6. proxool.maximum-connection-count=20
7. proxool.prototype-count=4
8. proxool.house-keeping-test-sql=select sysdate from DUAL
9. proxool.verbose=true
10. proxool.statistics=10s,1m,1d

在web.xml文件内加入以下内容：

**[html]** [view plain](http://blog.csdn.net/xilangyuyun/article/details/52800380) [copy](http://blog.csdn.net/xilangyuyun/article/details/52800380)

[在CODE上查看代码片](https://code.csdn.net/snippets/1925860)

1. **<servlet>**
2. **<servlet-name>**ServletConfigurator**</servlet-name>**
3. **<servlet-class>**

**[html]** [view plain](http://blog.csdn.net/xilangyuyun/article/details/52800380) [copy](http://blog.csdn.net/xilangyuyun/article/details/52800380)

[在CODE上查看代码片](https://code.csdn.net/snippets/1925860)

1. org.logicalcobwebs.proxool.configuration.ServletConfigurator

**[html]** [view plain](http://blog.csdn.net/xilangyuyun/article/details/52800380) [copy](http://blog.csdn.net/xilangyuyun/article/details/52800380)

[在CODE上查看代码片](https://code.csdn.net/snippets/1925860)

1. **</servlet-class>**
2. **<init-param>**
3. **<param-name>**propertyFile**</param-name>**
4. **<param-value>**WEB-INF/classes/Proxool.properties**</param-value>**
5. **</init-param>**
6. **<load-on-startup>**1**</load-on-startup>**
7. **</servlet>**

2、tomcat默认的连接池

**[html]** [view plain](http://blog.csdn.net/xilangyuyun/article/details/52800380) [copy](http://blog.csdn.net/xilangyuyun/article/details/52800380)

[在CODE上查看代码片](https://code.csdn.net/snippets/1925860)

1. **<Resource**
2. name="jdbc/test"
3. auth="Container"
4. type="javax.sql.DataSource"
5. driverClassName="oracle.jdbc.driver.OracleDriver"
6. url="jdbc:oracle:thin:@127.0.0.1:1521:orcl"
7. username="root"
8. password="123456"
9. maxActive="30"
10. maxIdle="15"
11. maxWait="-1"
12. validationQuery="select 1 from dual"
13. **/>**

[**spring**](http://lib.csdn.net/base/javaee)**中的配置：**  
1 proxool的配置

**[html]** [view plain](http://blog.csdn.net/xilangyuyun/article/details/52800380) [copy](http://blog.csdn.net/xilangyuyun/article/details/52800380)

[在CODE上查看代码片](https://code.csdn.net/snippets/1925860)

1. **<bean** id="proxoolDataSource" class="org.logicalcobwebs.proxool.ProxoolDataSource"**>**
2. **<property** name="driver" value="${jdbc.connection.driverClassName}"**/>**
3. **<property** name="driverUrl" value="${jdbc.connection.url}"**/>**
4. **<property** name="user" value="${jdbc.connection.username}"**/>**
5. **<property** name="password" value="${jdbc.connection.password}"**/>** <!-- 测试的SQL执行语句 -->
6. **<property** name="houseKeepingTestSql" value="${proxool.houseKeepingTestSql}"**/>** <!-- 最少保持的空闲连接数 默认2个 -->
7. **<property** name="prototypeCount" value="${proxool.prototypeCount}"**/>** <!-- proxool自动侦察各个连接状态的时间间隔(毫秒),侦察到空闲的连接就马上回收,超时的销毁 默认30秒 -->
8. **<property** name="houseKeepingSleepTime" value="${proxool.hourseKeepingSleepTime}"**/>** <!-- 最大活动时间(超过此时间线程将被kill,默认为5分钟) -->
9. **<property** name="maximumActiveTime" value="${proxool.maximumActiveTime}"**/>** <!-- 连接最长时间(默认为4个小时) -->
10. **<property** name="maximumConnectionLifetime" value="${proxool.maximumConnectionLifetime}"**/>** <!-- 最小连接数 默认2个 -->
11. **<property** name="minimumConnectionCount" value="${proxool.minimumConnectionCount}"**/>** <!-- 最大连接数 默认5个 -->
12. **<property** name="maximumConnectionCount" value="${proxool.maximumConnectionCount}"**/>** <!-- -->
13. **<property** name="statistics" value="${proxool.statistics}"**/>** <!-- 别名 -->
14. **<property** name="alias" value="${proxool.alias}"**/>** <!-- -->
15. **<property** name="simultaneousBuildThrottle" value="${proxool.simultaneous-build-throttle}"**/>**
16. **</bean>**

然后注入到sessionFactory中 

**[html]** [view plain](http://blog.csdn.net/xilangyuyun/article/details/52800380) [copy](http://blog.csdn.net/xilangyuyun/article/details/52800380)

[在CODE上查看代码片](https://code.csdn.net/snippets/1925860)

1. **<bean** id="sessionFactory" class="org.springframework.orm.hibernate3.LocalSessionFactoryBean"**>**
2. **<property** name="dataSource" ref="proxoolDataSource"**/>**
3. **</bean>**

   
   
Porxool 配置文件

**[html]** [view plain](http://blog.csdn.net/xilangyuyun/article/details/52800380) [copy](http://blog.csdn.net/xilangyuyun/article/details/52800380)

[在CODE上查看代码片](https://code.csdn.net/snippets/1925860)

1. **<proxool.xml>**
2. **<?xml** version="1.0" encoding="UTF-8"**?>**
3. **<proxool>** **<alias>**orcl**</alias>**
4. **<driver-url>**jdbc:oracle:thin:@127.0.0.1:1521:orcl**</driver-url>**
5. **<driver-class>**oracle.jdbc.driver.OracleDriver**</driver-class>**
6. **<driver-properties>**
7. **<property** name="user" value="root" **/>**
8. **<property** name="password" value="123456" **/>**
9. **</driver-properties>**
10. **<minimum-connection-count>**2**</minimum-connection-count>**
11. **<maximum-connection-count>**40**</maximum-connection-count>**
12. **<simultaneous-build-throttle>**20**</simultaneous-build-throttle>**
13. **<prototype-count>**2**</prototype-count>**
14. **<house-keeping-test-sql>**select CURRENT\_DATE**</house-keeping-test-sql>**
15. **</proxool>**

   
配置说明   
alias -〉数据库连接别名程序中需要使用的名称  
driver-url -〉数据库驱动   
driver-class -〉驱动程序类   
driver-properties -〉联机数据库的用户和密码   
minimum-connection-count -〉最小连接数量建议设置0以上保证第一次连接时间   
maximum-connection-count -〉最大连接数量如果超过最大连接数量则会抛出异常。连接数设置过多服务器CPU和内存性能消耗很 大。   
simultaneous-build-throttle -〉同时最大连接数   
prototype-count -〉一次产生连接的数量。 例如果现在prototype-count设置为4个但是现在已经有2个可以获得的连接那么 将会试图再创建2个连接。 但不能超过最大连接数。  
maximum-active-time -〉连接最大时间活动 默认5分钟   
maximum-connection-lifetime -〉连接最大生命时间 默认4小时

2   
<!-- 定义数据源Bean,使用C3P0数据源实现-->  
<bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource" destroy-method="close">  
        <!-- 指定连接数据库的驱动-->  
        <property name="driverClass" value="[**Oracle**](http://lib.csdn.net/base/oracle).jdbc.driver.OracleDriver"/>  
        <!-- 指定连接数据库的URL-->  
        <property name="jdbcUrl" value="jdbc:oracle:thin:@127.0.0.1:1521:orcl"/>  
        <!-- 指定连接数据库的用户名-->  
        <property name="user" value="root"/>  
        <!-- 指定连接数据库的密码-->  
        <property name="password" value="123456"/>  
        <!-- 指定连接池中保留的最大连接数. Default:15-->  
        <property name="maxPoolSize" value="15"/>  
        <!-- 指定连接池中保留的最小连接数-->  
        <property name="minPoolSize" value="10"/>  
        <!-- 指定连接池的初始化连接数  取值应在minPoolSize 与 maxPoolSize 之间.Default:3-->  
        <property name="initialPoolSize" value="5"/>  
        <!-- 最大空闲时间,60秒内未使用则连接被丢弃。若为0则永不丢弃。 Default:0-->  
        <property name="maxIdleTime" value="60"/>  
        <!-- 当连接池中的连接耗尽的时候c3p0一次同时获取的连接数. Default:3-->  
        <property name="acquireIncrement" value="5"/>  
        <!-- JDBC的标准,用以控制数据源内加载的PreparedStatements数量。但由于预缓存的statements属于单个connection而不是整个连接池所以设置这个参数需要考虑到多方面的因数.如果maxStatements与maxStatementsPerConnection均为0,则缓存被关闭。Default:0-->  
        <property name="maxStatements" value="0"/>  
        <!-- 每60秒检查所有连接池中的空闲连接.Default:0 -->  
        <property name="idleConnectionTestPeriod" value="60"/>  
        <!-- 定义在从数据库获取新连接失败后重复尝试的次数。 Default:30 -->  
        <property name="acquireRetryAttempts" value="30"/>  
        <!-- 获取连接失败将会引起所有等待连接池来获取连接的线程抛出异常。但是数据源仍有效保留,并在下次调用getConnection()的时候继续尝试获取连接。如果设为true,那么在尝试获取连接失败后该数据源将申明已断开并永久关闭。Default:false -->  
        <property name="breakAfterAcquireFailure" value="true"/>  
        <!-- 银性能消耗大请只在需要的时候是哟个它。如果设为true,那么在每个connection提交的时候都将校验其有效性。建议使用idleConnectionTestPeriod或automaticTestTable等提升连接测试的性能。 Default:false-->  
        <property name="testConnectionOnCheckout" value="false"/>  
</bean>  
      
[**hibernate**](http://lib.csdn.net/base/javaee)的配置：  
1 只需在hibernate.cfg.xml中加入

**[html]** [view plain](http://blog.csdn.net/xilangyuyun/article/details/52800380) [copy](http://blog.csdn.net/xilangyuyun/article/details/52800380)

[在CODE上查看代码片](https://code.csdn.net/snippets/1925860)

1. **<property** name="connection.username"**>**root**</property>**
2. **<property** name="connection.password"**>**123456**</property>**
3. **<property** name="connection.url"**>**jdbc:oracle:thin:@127.0.0.1:1521:orcl**</property>**
4. **<property** name="connection.driver\_class"**>**oracle.jdbc.driver.OracleDriver**</property>**
5. **<property** name="dialect"**>**org.hibernate.dialect.Oracle9Dialect**</property>**
6. **<property** name="format\_sql"**>**true**</property>**
7. **<property** name="show\_sql"**>**true**</property>**
8. **<property** name="myeclipse.connection.profile"**>**oracle10g\_112**</property>**
9. **<property** name="hibernate.connection.provider\_class"**>**org.hibernate.connection.C3P0ConnectionProvider **</property>**
10. **<property** name="hibernate.c3p0.min\_size"**>**5**</property>** //连接池中数据库连接的最小数目
11. **<property** name="hibernate.c3p0.max\_size"**>**30**</property>**  //连接池中数据库连接的最大数目
12. **<property** name="hibernate.c3p0.time\_out"**>**1800**</property>** //设定数据库连接的过期时间，以秒为单位
13. **<property** name="hibernate.c3p0.max\_statement"**>**50**</property>** //可以被缓存的PreparedStatement实例的最大数目。缓存适量的PreparedStatement实例，能够大大提高Hibernate的性能。
14. **<property** name="hibernate.c3p0.idle\_test\_period"**>**120**</property>**//在使数据库连接自动生效之前处于空闲状态的时间，以秒为单位
15. **<property** name="hibernate.c3p0.acquire\_increment"**>**2**</property>**

**[html]** [view plain](http://blog.csdn.net/xilangyuyun/article/details/52800380) [copy](http://blog.csdn.net/xilangyuyun/article/details/52800380)

[在CODE上查看代码片](https://code.csdn.net/snippets/1925860)



还有在classespath中加入c3p0-0.8.4.5.jar

2 dbcp(hibernate3.0 已经不支持dbcp)      
在hibernate.cfg.xml中加入

**[html]** [view plain](http://blog.csdn.net/xilangyuyun/article/details/52800380) [copy](http://blog.csdn.net/xilangyuyun/article/details/52800380)

[在CODE上查看代码片](https://code.csdn.net/snippets/1925860)

1. **<property** name="connection.driver\_class"**>**oracle.jdbc.driver.OracleDriver**</property>**
2. **<property** name="connection.url"**>** jdbc:oracle:thin:@127.0.0.1:1521:orcl**</property>**
3. **<property** name="connection.username"**>**root**</property>**
4. **<property** name="connection.password"**>**123456**</property>**
5. **<property** name="dbcp.maxActive"**>**100**</property>**
6. **<property** name="dbcp.whenExhaustedAction"**>**1**</property>**
7. **<property** name="dbcp.maxWait"**>**60000**</property>**
8. **<property** name="dbcp.maxIdle"**>**10**</property>**
9. **<property** name="dbcp.ps.maxActive"**>**100**</property>**
10. **<property** name="dbcp.ps.whenExhaustedAction"**>**1**</property>**
11. **<property** name="dbcp.ps.maxWait"**>**60000**</property>**
12. **<property** name="dbcp.ps.maxIdle"**>**10**</property>**

 还有在classespath中加入commons-pool-1.2.jar 和commons-dbcp-1.2.1.jar.    
   
3 proxool (有问题，暂时不可用)    
在hibernate.cfg.xml中加入:   
<property name="dialect">org.hibernate.dialect.Oracle9Dialect </property>   
<property name="format\_sql">true</property>  
<property name="show\_sql">true</property>   
<property name="myeclipse.connection.profile">oracle10g\_112 </property> <!-- 配置proxool数据库连接池 --> <!-- 连接池的别名 -->  
<property name="hibernate.proxool.pool\_alias">Proxool\_DB\_Pool</property>  <!-- 向Hibernate声明连接池的配置文件位置(和hibernate.cfg.xml放在一个目录下) -->  
<property name="hibernate.proxool.xml">ProxoolConf.xml</property>  <!-- 定义Hibernate的连接加载类，这里Proxool连接池是用这个，不同的连接池有不同的加载类 -->  
<property name="hibernate.connection.provider\_class"> org.hibernate.connection.ProxoolConnectionProvider </property> <!--为false,当hibernate开始被调用时，就会初始化proxool连接池，进行数据库连接等操作  -->  
<property name="hibernate.proxool.existing\_pool">true</property>

在和hibernate.cfg.xml同一个目录下，加一个ProxoolConf.xml文件，  
内容为<?xml version="1.0" encoding="UTF-8"?>   
<something-else-entirely>    
<proxool>     
<!-- 连接池别名 -->  
<alias>Proxool\_DB\_Pool</alias>      
<!--proxool只能管理由自己产生的连接-->   
<driver-url>jdbc:oracle:thin:@10.142.50.112:1521:omds</driver-url>      
<driver-class>oracle.jdbc.driver.OracleDriver</driver-class>  
<driver-properties>      
  <property name="user" value="omdssd\_admin"/>  
  <property name="password" value="omdssd\_hskj"/>   
</driver-properties>      
<!-- proxool自动侦察各个连接状态的时间间隔(毫秒),侦察到空闲的连接就马上回收,超时的销毁-->      
<house-keeping-sleep-time>90000</house-keeping-sleep-time>      
<!-- 指因未有空闲连接可以分配而在队列中等候的最大请求数,超过这个请求数的用户连接就不会被接受-->      
<maximum-new-connections>20</maximum-new-connections>  
<!-- 最少保持的空闲连接数-->      
<prototype-count>5</prototype-count>      
<!-- 允许最大连接数,超过了这个连接，再有请求时，就排在队列中等候，最大的等待请求数由maximum-new-connections决定-->      
<maximum-connection-count>100</maximum-connection-count>    
<!-- 最小连接数-->      
<minimum-connection-count>10</minimum-connection-count>     
</proxool>    
</something-else-entirely>      
并在classespath中加入proxool-0.8.3.jar