



黑客派

ActiveMQ JDBC 主从集群

作者: [88250](#)

原文链接: <https://hacpai.com/article/1362647148221>

来源网站: [黑客派](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

<p>从 ActiveMQ 4.1 版本开始支持 JDBC 主从集群。</p>
<script async src="https://pagead2.googlesyndication.com/pagead/js/adsbygoogle.js"></script>
<!-- 黑客派PC帖子内嵌-展示 -->
<ins class="adsbygoogle" style="display:block" data-ad-client="ca-pub-5357405790190342" data-ad-slot="8316640078" data-ad-format="auto" data-full-width-responsive="true"></ins>
<script>
 (adsbygoogle = window.adsbygoogle || []).push({});
</script>
<p>如果你正在使用纯 JDBC 以及非高性能日志，那可以认为数据库是一个存在单点故障的地方。</p>
<p>如果你的应用没有高性能需求，那单一数据库的方式是最容易备份与管理的。（译注：可以理解使用 JDBC 主从集群使用单一数据库容易配置）</p>
<h2 id="toc_h2_0">启动</h2>
<p>当你使用 JDBC 作为数据源时，你可以使用主从方式进行 ActiveMQ 集群，运行多个代理如下。</p>
<p>启动时，一个主代理将从数据库获取排它锁——所有其他的从代理将暂停服务，等待获取锁。</p>
<p></p>
<p>客户端应该使用故障转移传输来连接 ActiveMQ 服务代理。例如使用如下形式的 URL：</p>
<pre>failover:(tcp://broker1:61616,tcp://broker2:61616,tcp://broker3:61616)</pre>
<p>主代理必须启动传输连接器后客户端才能连接到，详见文末配置示例。（译注：获取排它锁后，代理才会启动传输连接器）</p>
<h2 id="toc_h2_1">主代理故障</h2>
<p>当主代理丢失数据库连接或丢失排它锁时，主代理将被关闭。当主代理关闭或故障时，其他任一代理将获取到排它锁，拓扑逻辑如下图：</p>
<p></p>
<p>一旦某一从代理获取了数据库排它锁，则它将成为主代理，并启用传输连接器。</p>
<p>客户端丢失对已经停止服务的代理，并使用故障转移传输尝试连接其他代理——唯一可用的就是才新启的主代理。（译注：客户端应该是逐一尝试，因为此时客户端并不知道哪个地址的代理成为了代理）</p>
<h2 id="toc_h2_2">主代理重启</h2>
<p>任何时刻你都可以重启集群中已经挂了的代理，重启后拓扑逻辑如下图：</p>
<p></p>
<h2 id="toc_h2_3">配置 JDBC 主从</h2>
<p>使用<code><jdbcPersistenceAdapter/> </code>避免高性能日志，而使用默认的 JDBC 主从配置。你只需启动多个代理并配置客户端 URLs 来连接该主从集群。</p>
<p>这些代理都会尝试获取数据库表排它锁，并且只有一个代理会获取到从而成为主代理。</p>
<p> </p>
<p>如下的配置示例展示了如何配置 ActiveMQ JDBC 主从集群：</p>
<pre><beans>
<p><!-- Allows us to use system properties as variables in this configuration file --><br

```
<bean class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer"/>
</p>
<p><broker xmlns="http://activemq.apache.org/schema/core" target="_blank" rel="nofollow ugc">http://activemq.apache.org/schema/core</a>"></p>
<pre> <code class="highlight-chroma">&lt;destinationPolicy&gt;
&lt;policyMap&gt;&lt;policyEntries&gt;
```

not found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]</policyEntries
ot found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]>
ot found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]</policyMap
ot found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]>

not found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]</destinationPolicy
ot found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]>

not found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]<persistenceAdapter
ot found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]>

not found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]<jdbcPersistenceAdapter dataDirectory=
ot found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]"\${activemq.base}/activemq-data
ot found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]"/
ot found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]>

not found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]</persistenceAdapter
ot found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]>

not found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]<transportConnectors
ot found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]>

not found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]<transportConnector name=
ot found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]"default
ot found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]" uri=
ot found render function for node [type=NodeHTMLEntity, Tokens=]¬ found render function for node [type=NodeHTMLEntity, Tokens=]>

```
on for node [type=NodeHTMLEntity, Tokens=&]quot;tcp://localhost:61616
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render funct
on for node [type=NodeHTMLEntity, Tokens=&]quot;/
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render funct
on for node [type=NodeHTMLEntity, Tokens=&]gt;
not found render function for node [type=NodeHTMLEntity, Tokens=&]not found render fun
tion for node [type=NodeHTMLEntity, Tokens=&]lt;/transportConnectors
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render funct
on for node [type=NodeHTMLEntity, Tokens=&]gt;
```

</code></pre>

```
<p>&lt;/broker&gt;</p>
<p>&lt;!-- This xbean configuration file supports all the standard Spring XML configuration
ptions --&gt;</p>
<p>&lt;!-- Postgres DataSource Sample Setup --&gt;<br>
&lt;!--<br>
&lt;bean id="postgres-ds" class="org.postgresql.ds.PGPoolingDataSource"&gt;<br>
&lt;property name="serverName" value="localhost"/&gt;<br>
&lt;property name="databaseName" value="activemq"/&gt;<br>
&lt;property name="portNumber" value="0"/&gt;<br>
&lt;property name="user" value="activemq"/&gt;<br>
&lt;property name="password" value="activemq"/&gt;<br>
&lt;property name="dataSourceName" value="postgres"/&gt;<br>
&lt;property name="initialConnections" value="1"/&gt;<br>
&lt;property name="maxConnections" value="10"/&gt;<br>
&lt;/bean&gt;<br>
--&gt;</p>
<p>&lt;!-- MySQL DataSource Sample Setup --&gt;<br>
&lt;!--<br>
&lt;bean id="mysql-ds" class="org.apache.commons.dbcp.BasicDataSource" destroy-metho
="close"&gt;<br>
&lt;property name="driverClassName" value="com.mysql.jdbc.Driver"/&gt;<br>
&lt;property name="url" value="jdbc:mysql://localhost/activemq?relaxAutoCommit=true"/&g
;<br>
&lt;property name="username" value="activemq"/&gt;<br>
&lt;property name="password" value="activemq"/&gt;<br>
&lt;property name="poolPreparedStatements" value="true"/&gt;<br>
&lt;/bean&gt;<br>
--&gt;</p>
<p>&lt;!-- Oracle DataSource Sample Setup --&gt;<br>
&lt;!--<br>
&lt;bean id="oracle-ds" class="org.apache.commons.dbcp.BasicDataSource" destroy-metho
="close"&gt;<br>
&lt;property name="driverClassName" value="oracle.jdbc.driver.OracleDriver"/&gt;<br>
&lt;property name="url" value="jdbc:oracle:thin:@localhost:1521:AMQDB"/&gt;<br>
&lt;property name="username" value="scott"/&gt;<br>
&lt;property name="password" value="tiger"/&gt;<br>
&lt;property name="poolPreparedStatements" value="true"/&gt;<br>
&lt;/bean&gt;<br>
--&gt;</p>
<p>&lt;!-- Embedded Derby DataSource Sample Setup --&gt;<br>
&lt;!--<br>
&lt;bean id="derby-ds" class="org.apache.derby.jdbc.EmbeddedDataSource"&gt;<br>
```

```
&lt;property name="databaseName" value="derbydb"/&gt;<br>
&lt;property name="createDatabase" value="create"/&gt;<br>
&lt;/bean&gt;<br>
--&gt;</p>
<p>&lt;/beans&gt;</p></pre>
<script async src="https://pagead2.googlesyndication.com/pagead/js/adsbygoogle.js"></scr
pt>
<!-- 黑客派PC帖子内嵌-展示 -->
<ins class="adsbygoogle" style="display:block" data-ad-client="ca-pub-5357405790190342"
data-ad-slot="8316640078" data-ad-format="auto" data-full-width-responsive="true"></in
>
<script>
  (adsbygoogle = window.adsbygoogle || []).push({});
</script>
<p></p>
<p>&nbsp;</p>
<p>译自: http://activemq.apache.org/jdbc-master-slave.html</p>
```