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

从 ActiveMQ 4.1 版本开始支持 JDBC 主从集群。

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。
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）
<h2 id="toc_h2_0">启动</h2>
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。
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。

客户端应该使用故障移传输来连接 ActiveMQ 服务代理。例如使用如下形式的 URL：
<pre>failover:(tcp://broker1:61616,tcp://broker2:61616,tcp://broker3:61616)</pre>
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）
<h2 id="toc_h2_1">主代理故障</h2>
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：

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。
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）
<h2 id="toc_h2_2">主代理重启</h2>
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：

<h2 id="toc_h2_3">配置 JDBC 主从</h2>
使用 <jdbcPersistenceAdapter/> 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。
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。

如下的配置示例展示了如何配置 ActiveMQ JDBC 主从集群：
<pre><beans>
<!-- Allows us to use system properties as variables in this configuration file -->

</pre>

<bean class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer"/&t;
<broker xmlns="http://activemq.apache.org/schema/core">
<pre><code class="highlight-chroma"><destinationPolicy>
  <policyMap><policyEntries>

not found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]lt;/policyEntries
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]gt;
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]lt;/policyMap
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]gt;

not found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]lt;/destinationPolicy
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]gt;

not found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]lt;persistenceAdapter
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]gt;

not found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]lt;jdbcPersistenceAdapter dataDirectory=
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]quot;${activemq.base}/activemq-data
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]quot;/
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]gt;

not found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]lt;/persistenceAdapter
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]gt;

not found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]lt;transportConnectors
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]gt;

not found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]lt;transportConnector name=
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]quot;default
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render function for node [type=NodeHTMLEntity, Tokens=&]quot; uri=
ot found render function for node [type=NodeHTMLEntity, Tokens=&]not found render funct

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>

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

<property name="databaseName" value="derbydb"/>
<property name="createDatabase" value="create"/>
</bean>
-->
</beans> </pre>
<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>

译自：http://activemq.apache.org/jdbc-master-slave.html