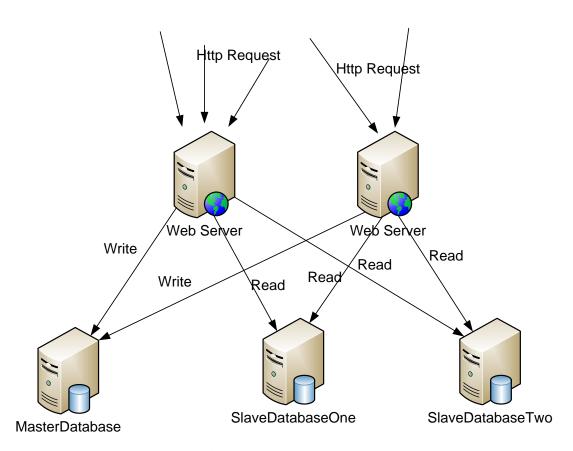
# Routing4DB 使用文档

## 景目

2
2
5
6
8
8
12
14
15
15
16
16
16
17

### 一、Master-Slave 读写分离实现

#### 1.1 写 Master, 读多个 Slaves, 示意图如下:



1.1 读 Master 写 Slave 示意图

Spring 配置如下,详情见 test/resource 目录下 write-master-read-slaves.xml 及 WriteMasterReadSlavesTest.java:

```
<!-- Master数据源 -->
<bean id="masterDataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource"</pre>
destroy-method="close">
      cproperty name="driverClass" value="com.mysql.jdbc.Driver"/>
      property name="jdbcUrl">
<value>jdbc:mysql://localhost:3306/test?useUnicode=true&amp;characterEncodin
g=UTF-8&autoReconnect=true&failOverReadOnly=false</value>
      </property>
      cproperty name="user" value="root" />
      cproperty name="password" value="lovejava" />
      property name="initialPoolSize" value="2" />
      property name="minPoolSize" value="2" />
      property name="maxPoolSize" value="10" />
      property name="acquireIncrement" value="5" />
      property name="maxIdleTime" value="30" />
      property name="maxStatements" value="0" />
```

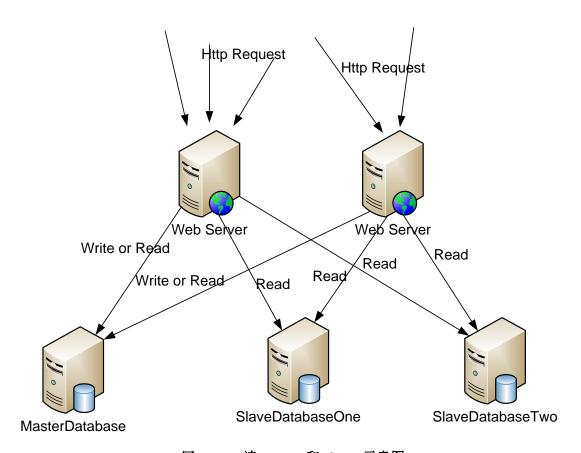
```
<!-- Slave数据源 -->
<bean id="slaveDataSourceOne"</pre>
class="com.mchange.v2.c3p0.ComboPooledDataSource" destroy-method="close">
      cproperty name="driverClass" value="com.mysql.jdbc.Driver"/>
      property name="jdbcUrl">
<value>jdbc:mysql://localhost:3306/test?useUnicode=true&amp;characterEncodin
q=UTF-8& autoReconnect=true& failOverReadOnly=false</value>
      </property>
      property name="user" value="root" />
      property name="password" value="lovejava" />
      cproperty name="initialPoolSize" value="2" />
      property name="minPoolSize" value="2" />
      property name="maxPoolSize" value="10" />
      property name="acquireIncrement" value="5" />
      property name="maxIdleTime" value="30" />
      property name="maxStatements" value="0" />
</bean>
<bean id="slaveDataSourceTwo"</pre>
class="com.mchange.v2.c3p0.ComboPooledDataSource" destroy-method="close">
      property name="driverClass" value="com.mysql.jdbc.Driver"/>
      property name="jdbcUrl">
<value>jdbc:mysql://localhost:3306/test?useUnicode=true&amp;characterEncodin
g=UTF-8&autoReconnect=true&failOverReadOnly=false</value>
      </property>
      property name="user" value="root" />
      property name="password" value="lovejava" />
      property name="initialPoolSize" value="2" />
      property name="minPoolSize" value="2" />
      property name="maxPoolSize" value="10" />
      property name="acquireIncrement" value="5" />
      cproperty name="maxIdleTime" value="30" />
      property name="maxStatements" value="0" />
</bean>
<!-- 配置Routing4DB 数据源 -->
<bean id="routing4DBDataSource"</pre>
class="com.google.code.routing4db.datasource.Routing4DBDataSource">
  property name="targetDataSources">
    <!-- 配置数据源标示符号 -->
    <map>
       <entry key="slaveDataSourceOne" value-ref="slaveDataSourceOne"/>
```

<entry key="slaveDataSourceTwo" value-ref="slaveDataSourceTwo"/>

</bean>

```
</map>
  </property>
  </bean>
<!-- JdbcTemplate -->
<bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
   cproperty name="dataSource" ref="routing4DBDataSource"/>
</bean>
<!-- 事务配置, 事务注解@Transactional要放到实现类上, 不支持放到接口上 -->
<bean id="transactionManager"</pre>
class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
 property name="dataSource" ref="routing4DBDataSource" />
</bean>
<tx:annotation-driven transaction-manager="transactionManager"/>
<!-- 配置路由策略 -->
<bean id="masterSlaveStrategy"</pre>
class="com.google.code.routing4db.strategy.impl.MasterSlaveStrategy">
   property name="readMethodPatterns">
      st>
        <value>*get*</value>
        <value>*find*</value>
      </list>
   </property>
   <!-- Master数据源标识符 -->
   <!-- Slaves数据源 从0开始,以此编号 -->
   property name="dataSourceKeyMap">
      <map>
        <entry key="0" value="slaveDataSourceOne"></entry>
        <entry key="1" value="slaveDataSourceTwo"></entry>
      </map>
   </property>
</bean>
<!-- Dao实现 -->
<bean id="userDaoTarget"</pre>
class="com.google.code.routing4db.dao.UserDaoJdbcTemplateImpl"></bean>
<!-- 配置DAO接口代理 ) -->
<bean id="userDao"</pre>
```

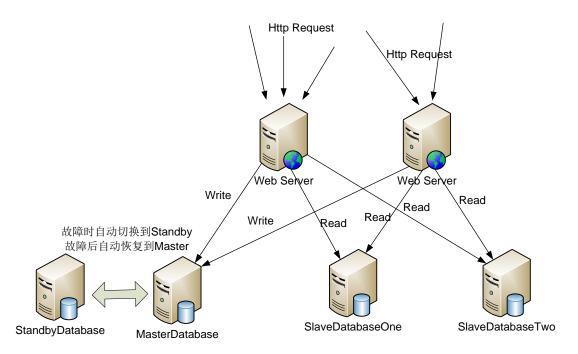
#### 1.2 写 Master, 读 Master 和多个 Slave, 示意图如下



1.2 写 Master 读 Master 和 Slaves 示意图

此方式配置和 1.1 方式相同,仅仅在路由策略 Slaves 节点中加入 Master 即可。不同的配置如下:

#### 1.3 Master-Standby-Slaves 实现,此方式示意图如下:



该方式在 Master 节点出故障时,会自动切换到 Standby 节点。将写入操作转移到 Standby 节点,在 Master 节点故障恢复正常后,自动把写操作迁移到 Master 节点。此方式配置如下:

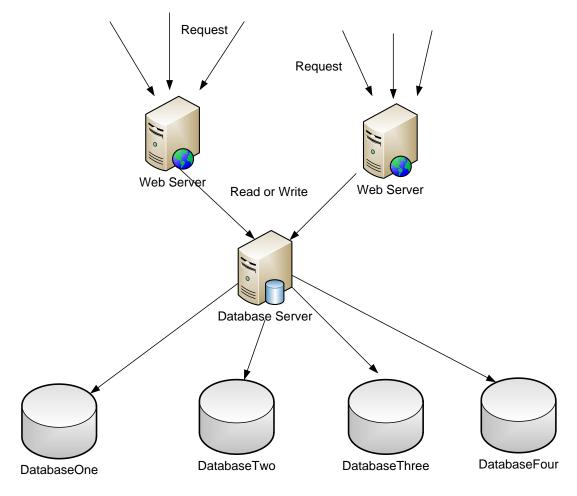
<value>jdbc:mysql://192.168.56.102:3306/test?useUnicode=true&amp;characterEn

```
coding=UTF-8& autoReconnect=true& failOverReadOnly=false</value>
      </property>
      property name="user" value="root" />
      property name="password" value="lovejava" />
      cproperty name="initialPoolSize" value="2" />
      property name="minPoolSize" value="2" />
      property name="maxPoolSize" value="10" />
      property name="acquireIncrement" value="5" />
      property name="maxIdleTime" value="30" />
      cproperty name="maxStatements" value="0" />
      property name="checkoutTimeout" value="2000"/>
      property name="acquireRetryAttempts" value="3">
</bean>
<!-- Standby数据源 -->
<bean id="standbyDataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource"</pre>
destroy-method="close">
      cproperty name="driverClass" value="com.mysql.jdbc.Driver"/>
      property name="jdbcUrl">
<value>jdbc:mysql://localhost:3306/test?useUnicode=true&amp;characterEncodin
q=UTF-8&autoReconnect=true&failOverReadOnly=false</value>
      </property>
      property name="user" value="root" />
      property name="password" value="lovejava" />
      property name="initialPoolSize" value="2" />
      property name="minPoolSize" value="2" />
      property name="maxPoolSize" value="10" />
      property name="acquireIncrement" value="5" />
      cproperty name="maxIdleTime" value="30" />
      property name="maxStatements" value="0" />
      property name="checkoutTimeout" value="2000"/>
      cproperty name="acquireRetryAttempts" value="3">
</bean>
<!-- 配置Master Standby 的数据源 -->
<bean id="masterStandbyDataSource"</pre>
class="com.google.code.routing4db.datasource.MasterStrandbyDataSource">
  cproperty name="standbyDataSource" ref="standbyDataSource" />
  <!-- 配置Routing4DB 数据源 -->
<bean id="routing4DBDataSource"</pre>
class="com.google.code.routing4db.datasource.Routing4DBDataSource">
  property name="targetDataSources">
```

其它配置与 1.1 及 1.2 相同,可参考 1.1 及 1.2。

#### 二、分库功能

#### 2.1 单机分库功能,示意图如下:



2.1 单机分库示意图

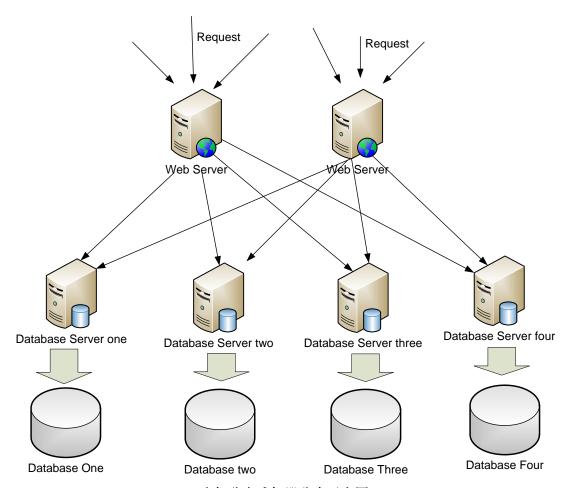
此方式配置如下,具体可参考测试工程中的: ShardDatabaseByModTest.java 及 shard-database-by-mod.xml

```
<value>jdbc:mysql://localhost:3306/test0?useUnicode=true&amp;characterEncodi
ng=UTF-8&autoReconnect=true&failOverReadOnly=false</value>
      </property>
      property name="user" value="root" />
      property name="password" value="lovejava" />
      property name="initialPoolSize" value="2" />
      property name="minPoolSize" value="2" />
      cproperty name="maxPoolSize" value="10" />
      cproperty name="acquireIncrement" value="5" />
      property name="maxIdleTime" value="30" />
      property name="maxStatements" value="0" />
</hean>
<!-- mode two 数据源 -->
<bean id="dataSourceTwo" class="com.mchange.v2.c3p0.ComboPooledDataSource"</pre>
destroy-method="close">
      cproperty name="driverClass" value="com.mysql.jdbc.Driver"/>
      property name="jdbcUrl">
<value>jdbc:mysql://localhost:3306/test1?useUnicode=true&amp;characterEncodi
ng=UTF-8&autoReconnect=true&failOverReadOnly=false</value>
      </property>
      property name="user" value="root" />
      property name="password" value="lovejava" />
      cproperty name="initialPoolSize" value="2" />
      property name="minPoolSize" value="2" />
      cproperty name="maxPoolSize" value="10" />
      property name="acquireIncrement" value="5" />
      property name="maxIdleTime" value="30" />
      property name="maxStatements" value="0" />
</hean>
<!-- mode three 数据源 -->
<bean id="dataSourceThree" class="com.mchange.v2.c3p0.ComboPooledDataSource"</pre>
destroy-method="close">
      cproperty name="driverClass" value="com.mysql.jdbc.Driver"/>
      property name="jdbcUrl">
<value>jdbc:mysql://localhost:3306/test2?useUnicode=true&amp;characterEncodi
ng=UTF-8&autoReconnect=true&failOverReadOnly=false</value>
      </property>
      cproperty name="user" value="root" />
      property name="password" value="lovejava" />
      cproperty name="initialPoolSize" value="2" />
      property name="minPoolSize" value="2" />
      property name="maxPoolSize" value="10" />
      property name="acquireIncrement" value="5" />
```

```
cproperty name="maxIdleTime" value="30" />
      property name="maxStatements" value="0" />
</bean>
<!-- mode four 数据源 -->
<bean id="dataSourceFour" class="com.mchange.v2.c3p0.ComboPooledDataSource"</pre>
destroy-method="close">
      cproperty name="driverClass" value="com.mysql.jdbc.Driver"/>
      property name="jdbcUrl">
<value>jdbc:mysql://localhost:3306/test3?useUnicode=true&amp;characterEncodi
ng=UTF-8&autoReconnect=true&failOverReadOnly=false</value>
      </property>
      property name="user" value="root" />
      property name="password" value="lovejava" />
      property name="initialPoolSize" value="2" />
      property name="minPoolSize" value="2" />
      property name="maxPoolSize" value="10" />
      property name="acquireIncrement" value="5" />
      cproperty name="maxIdleTime" value="30" />
      property name="maxStatements" value="0" />
</bean>
<!-- 配置Routing4DB 数据源 -->
<bean id="routing4DBDataSource"</pre>
class="com.google.code.routing4db.datasource.Routing4DBDataSource">
  property name="targetDataSources">
    <map>
       <entry key="dataSourceOne" value-ref="dataSourceOne"/>
       <entry key="dataSourceTwo" value-ref="dataSourceTwo"/>
       <entry key="dataSourceThree" value-ref="dataSourceThree"/>
       <entry key="dataSourceFour" value-ref="dataSourceFour"/>
    </map>
   </property>
   <!-- 无路由配置时默认的datasource -->
  </bean>
<!-- JdbcTemplate -->
<bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
   property name="dataSource" ref="routing4DBDataSource"/>
</bean>
<!-- 事务配置, 事务注解@Transactional要放到实现类上, 不支持放到接口上 -->
```

```
<bean id="transactionManager"</pre>
class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
 property name="dataSource" ref="routing4DBDataSource" />
</bean>
<tx:annotation-driven transaction-manager="transactionManager"/>
<!-- 配置路由策略 -->
<bean id="modRoutingStrategy"</pre>
class="com.google.code.routing4db.strategy.impl.ModRoutingStrategy">
   <!-- 分4个数据库,采用Id属性进行分库路由 -->
   property name="dataSourceNum" value="4">
    cproperty name="propertyName" value="id"></property>
   <!--取模的数据源 从0开始,以此编号 -->
   property name="dataSourceKeyMap">
      <map>
        <entry key="0" value="dataSourceOne"></entry>
        <entry key="1" value="dataSourceTwo"></entry>
        <entry key="2" value="dataSourceThree"></entry>
        <entry key="3" value="dataSourceFour"></entry>
      </map>
   </property>
</bean>
<!-- Dao实现 -->
<bean id="userDaoTarget"</pre>
class="com.google.code.routing4db.dao.UserDaoJdbcTemplateImpl"></bean>
<!-- 配置DAO接口代理 ) -->
<bean id="userDao"</pre>
class="com.google.code.routing4db.spring.RoutingSpringFactoryBean">
   <!-- 代理接口 -->
   property name="targetInterface"
value="com.google.code.routing4db.dao.UserDao">
   <!-- 代理的DAO实际对象 -->
   <!-- 路由策略 -->
   cproperty name="routingStrategy" ref="modRoutingStrategy">
</bean>
```

## 2.2 多机集群分库,构建分布式数据库,示意图如下:



2.2 多机分布式机器分库示意图

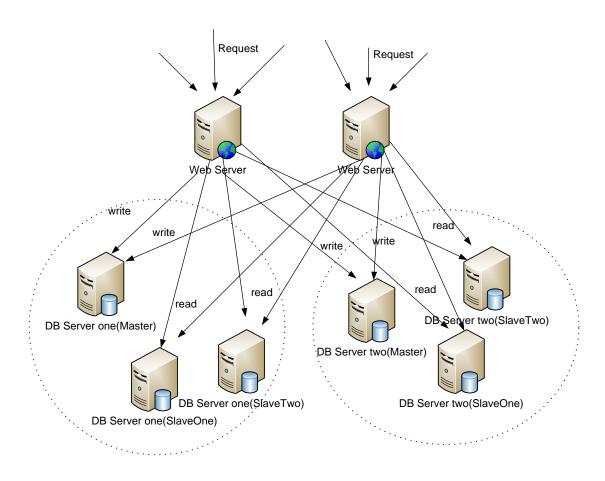
#### 配置如下:

```
<!-- mode one 数据源 -->
<bean id="dataSourceOne" class="com.mchange.v2.c3p0.ComboPooledDataSource"</pre>
destroy-method="close">
      cproperty name="driverClass" value="com.mysql.jdbc.Driver"/>
      property name="jdbcUrl">
<value>jdbc:mysql://192.168.3.25:3306/test?useUnicode=true&amp;characterEnco
ding=UTF-8&autoReconnect=true&failOverReadOnly=false</value>
      </property>
      property name="user" value="root" />
      cproperty name="password" value="lovejava" />
      cproperty name="initialPoolSize" value="2" />
      property name="minPoolSize" value="2" />
      property name="maxPoolSize" value="10" />
      cproperty name="acquireIncrement" value="5" />
      property name="maxIdleTime" value="30" />
      cproperty name="maxStatements" value="0" />
</bean>
```

```
<!-- mode two 数据源 -->
<bean id="dataSourceTwo" class="com.mchange.v2.c3p0.ComboPooledDataSource"</pre>
destroy-method="close">
      property name="driverClass" value="com.mysql.jdbc.Driver"/>
      property name="jdbcUrl">
<value>jdbc:mysql://192.168.3.26:3306/test1?useUnicode=true&amp;characterEnc
oding=UTF-8&autoReconnect=true&failOverReadOnly=false</value>
      </property>
      property name="user" value="root" />
      property name="password" value="lovejava" />
      property name="initialPoolSize" value="2" />
      property name="minPoolSize" value="2" />
      property name="maxPoolSize" value="10" />
      property name="acquireIncrement" value="5" />
      property name="maxIdleTime" value="30" />
      cproperty name="maxStatements" value="0" />
</bean>
<!-- mode three 数据源 -->
<bean id="dataSourceThree" class="com.mchange.v2.c3p0.ComboPooledDataSource"</pre>
destroy-method="close">
      driverClass" value="com.mysql.jdbc.Driver"/>
      property name="jdbcUrl">
<value>jdbc:mysql://192.168.3.27:3306/test2?useUnicode=true&amp;characterEnc
oding=UTF-8&autoReconnect=true&failOverReadOnly=false</value>
      </property>
      property name="user" value="root" />
      property name="password" value="lovejava" />
      cproperty name="initialPoolSize" value="2" />
      property name="minPoolSize" value="2" />
      cproperty name="maxPoolSize" value="10" />
      property name="acquireIncrement" value="5" />
      property name="maxIdleTime" value="30" />
      property name="maxStatements" value="0" />
</bean>
<!-- mode four 数据源 -->
<bean id="dataSourceFour" class="com.mchange.v2.c3p0.ComboPooledDataSource"</pre>
destroy-method="close">
      property name="driverClass" value="com.mysql.jdbc.Driver"/>
      property name="jdbcUrl">
<value>jdbc:mysql://192.168.3.28:3306/test3?useUnicode=true&amp;characterEnc
oding=UTF-8&autoReconnect=true&failOverReadOnly=false</value>
```

此方式与单机分库没什么区别,唯一区别就是把数据源的 IP 修改一下,其余配置可参考 2.1 单机版分库。

#### 2.3 高可用多级分布式集群,示意图如下:



#### 2.3 高可用分布式集群

#### 部分配置如下:

```
<entry key="dataSourceOneMaster" value-ref="dataSourceOneMaster"/>
        <entry key="dataSourceOneSlaveOne"</pre>
value-ref="dataSourceOneSlaveOne"/>
        <entry key="dataSourceOneSlaveTwo"</pre>
value-ref="dataSourceOneSlaveTwo"/>
        <entry key="dataSourceTwoMaster" value-ref="dataSourceTwoMaster"/>
        <entry key="dataSourceSlaveOne" value-ref="dataSourceSlaveOne"/>
        <entry key="dataSourceSlaveTwo" value-ref="dataSourceSlaveTwo"/>
    </map>
   </property>
   <!-- 默认的datasource -->
  </bean>
<!-- 配置路由策略 -->
<bean id="modMasterSlaveRoutingStrategy"</pre>
class="com.google.code.routing4db.strategy.impl.ModMasterSlaveRoutingStrateg
   <!-- 分2集群组,采用Id属性进行分库路由 -->
   property name="dataSourceNum" value="2">
    cproperty name="propertyName" value="id"></property>
   <!--取模的数据源 从0开始,以此编号 -->
   property name="dataSourceKeyMap">
       <map>
        <entry key="0"</pre>
value="dataSourceOneMaster,dataSourceOneSlaveOne,dataSourceOneSlaveTwo"></en
try>
        <entry key="1"</pre>
value="dataSourceTwoMaster,dataSourceTwoSlaveOne,dataSourceTwoSlaveTwo"></en
try>
      </map>
   </property>
</bean>
```

其它配置与 2.1 或 2.2 配置相同,具体可参考 mod-master-slaves-example.xml 及 ModMasterSlavesTest.java

## 三、负载均衡示例

负责均衡,可参考 Master-Slaves 模式的实现。也可利用数据库驱动自带的功能实现,如 Mysql 的 Connector-J,本身驱动就实现了对应的功能。

### 四、自定义数据源路由策略

通过扩展 RoutingStrategy 接口,实现 route 方法,来实现自定义的数据源路由策略。 在 route 方法可根据 DAO 对象,方法,以及方法参数选择需要的数据源,将选择的数据源 的 key 设置在 RoutingHolder 中。在自定义扩展数据源路由策略时可参考: MasterSlaveStrategy, ModRoutingStrategy 及 ModMasterSlaveRoutingStrategy 代码。route 方法参数说明如下:

#### 五、如何指定数据源

你可以通过 RoutingHolder 的 setCurrentDataSourceKey(String Key)方法手动指定数据源。其中 key 为你注册到 Routing4DBDataSource 中的 key,设置 key 为 null,则采用默认的数据源。示例如下:

采用 dataSourceOne 数据源:

```
RoutingHolder.setCurrentDataSourceKey("dataSourceOne");
采用默认数据源:
```

RoutingHolder.setCurrentDataSourceKey(null);

#### 六、事务的处理

Routing4DB 采用接口代理的方式实现数据源的路由,可实现单数据源事务支持,不支持多数据源事务。事务处理方式和普通处理方式没有什么差别;如果需要事务支持,只需在对应接口实现类的方法上,只需加入@Transactional或 AOP的方式实现。示例如下:

```
@Transactional()
public void insertWithTransaction(User user) {
    String sql ="insert sql";
    jdbcTemplate.execute(sql);
}
```

注意:要把@Transactional 放到实现类,而非接口上。原因是 Routing4DB 采用基于接口代理的方式实现数据源路由,若@Transactional 放到接口上,Spring FactoryBean会在创建接口代理后,针对该代理创建事务处理拦截,事务处理在代理执行(路由策略执行)之前执行,导致未知异常。

## 七、针对 Mybatis 的增强功能

Mybatis 3.0 提供了基于工厂代理创建 DAO 的方式,您只需书写接口,无需实现类。 Routing4DB 针对 Mybatis 这一功能提供了增强; 你只需实现简单修改即可实现数据源的路由。 示例如下:

原 Mybatis 配置:

```
<!-- (此配置方式不支持事务,详情见6事务说明) --> <bean id="userMapper" class="org.mybatis.spring.mapper.MapperFactoryBean">
```

### 八、BUG 反馈及交流

若你有 BUG 反馈,若你想参与其中,改进 Routing4DB 项目。请与我联系,Routing4DB 期待您的参与。作者联系方式:

email: efurture@gmail.com 或 gubaojian@163.com

**QQ:** 787277208