java 多数据源

作者：wts
原文链接：https://ld246.com/article/1526544862476
来源网站：链滴

在醒目中需要一个接口方法里操作多个数据库，因此产生多数据源切换问题；

采用spring + springMVC + mybatis，相关测试如下：

首先数数据源配置

```
<bean id="dataSource1" class="com.alibaba.druid.pool.DruidDataSource" init-method="init" destroy-method="close">
  <property name="driverClassName" value="xxx" />
  <property name="url" value="xxx" />
  <property name="username" value="xxx" />
  <property name="password" value="xxx" />
  <property name="initialSize" value="${jdbc.pool.init}" />
  <property name="minIdle" value="1" />
  <property name="maxActive" value="3" />
  <property name="maxWait" value="20" />
  <property name="timeBetweenEvictionRunsMillis" value="60000" />
  <property name="minEvictableIdleTimeMillis" value="300000" />
  <property name="validationQuery" value="SELECT 'x' FROM DUAL" />
  <property name="testWhileIdle" value="true" />
  <property name="testOnBorrow" value="false" />
  <property name="testOnReturn" value="false" />
  <property name="filters" value="stat" />
</bean>
```

再来一个

```
<bean id="dataSource2" class="com.alibaba.druid.pool.DruidDataSource" init-method="init" destroy-method="close">
  <property name="driverClassName" value="xxx" />
  <property name="url" value="xxx" />
  <property name="username" value="xxx" />
  <property name="password" value="xxx" />
  <property name="initialSize" value="xxx" />
  <property name="minIdle" value="1" />
  <property name="maxActive" value="3" />
  <property name="maxWait" value="20" />
  <property name="timeBetweenEvictionRunsMillis" value="60000" />
  <property name="minEvictableIdleTimeMillis" value="300000" />
  <property name="validationQuery" value="SELECT 'x' FROM DUAL" />
  <property name="testWhileIdle" value="true" />
  <property name="testOnBorrow" value="false" />
  <property name="testOnReturn" value="false" />
  <property name="filters" value="stat" />
</bean>
```

接下来是数据源的切换类

```
public class MultipleDataSource extends AbstractRoutingDataSource {
```

```
    public static final ThreadLocal<String> contextHolder = new ThreadLocal<String>();

    public static final String dataSource1 = "dataSource1";
    public static final String dataSource2 = "dataSource2";

/**
 * 设置当前数据源
 * @param dbType
 */
public static void setDbType(String dbType){
    contextHolder.set(dbType);
}

/**
 * 当前数据源
 */
public static String getDbType(){
    return contextHolder.get();
}

@Override
protected Object determineCurrentLookupKey() {
    return contextHolder.get();
}
```

使用AOP来实现动态切换 配置如下

http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/pring-aop-4.2.xsd

```
<context:component-scan base-package="xx.xx.xx" />
<aop:aspectj-autoproxy proxy-target-class="true"/>
```

在配置AOP的时候需要注意的是我们是在SpringMVC上aop监测，那么所有的扫描注入都在SpringMC的配置文件中完成，不要再spring的配置文件中完成，不然在开启代理后，是没有任何作用的。

AOP实现类

```
@Component
@Aspect
public class MultipleDataSourceAspectAdvice {
    //这里根据自己的业务选择切面的路径和数据源的设置
    @Around("execution(* xx.xx.*.*(..))")
    public Object doAround(ProceedingJoinPoint jp) throws Throwable {
        String methodName=jp.getSignature().getName();
        if("save".equals(methodName)){
            MultipleDataSource.setDbType(MultipleDataSource.dataSource1);
            System.out.println("设置dataSource1");
        }else{
            MultipleDataSource.setDbType(MultipleDataSource.dataSource2);
            System.out.println("设置dataSource2");

        }
        return jp.proceed();
    }
```

```
}
```

到这个时候，基本上的配置已经完成了，但是在使用过程中发现问题，已经设置了数据源，可是数据是随机获取的，是什么原因呢？

最后经过查阅资料得知Spring的事务与数据源是绑定的，也就是说如果你开启了事务，那么数据源已绑定了。那么这个时候，你在去切换数据源就无效了。也就是说要想有效，那么就要在事务开启之前把数据源切换好。

在Spring中有一个注解是用来设置加载顺序的@Order(?)，把这个注解加载AOP的实现类上，参数0.12越小代表越先执行。d

加上这个注解以后执行代码，解决问题。