spring配置文件

```
spring:
  datasource:
    primary:
      driver-class-name: com.mysql.jdbc.Driver
      url: jdbc:mysql://120.79.183.246:3306/temp1
      username: root
      password: Ajaxdafdajfd8327323jfKDFFDA
    secondary:
      driver-class-name: com.mysql.jdbc.Driver
      url: jdbc:mysql://120.79.183.246:3306/temp2
      username: root
      password: Ajaxdafdajfd8327323jfKDFFDA

  jpa:
    hibernate:
      ddl-auto: update
```

数据源配置类

```
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.boot.autoconfigure.jdbc.DataSourceBuilder;
import org.springframework.boot.context.properties.ConfigurationProperties;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Primary;

import javax.sql.DataSource;
```

```
/**
 *
 * 数据源的配置
 */
@Configuration
public class DataSourceConfig {

    @Bean(name = "primaryDataSource")
    @Qualifier("primaryDataSource")
    @ConfigurationProperties(prefix="spring.datasource.primary")
    public DataSource primaryDataSource() {
        return DataSourceBuilder.create().build();
    }

    @Bean(name = "secondaryDataSource")
    @Qualifier("secondaryDataSource")
    @Primary
    @ConfigurationProperties(prefix="spring.datasource.secondary")
    public DataSource secondaryDataSource() {
        return DataSourceBuilder.create().build();
    }
}
```

配置第一个数据源

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.boot.autoconfigure.orm.jpa.JpaProperties;
import org.springframework.boot.orm.jpa.EntityManagerFactoryBuilder;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
```

```
import org.springframework.context.annotation.Primary;
import
org.springframework.data.jpa.repository.config.EnableJpaRepositories;
import org.springframework.orm.jpa.JpaTransactionManager;
import
org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean;
import org.springframework.orm.jpa.vendor.Database;
import org.springframework.transaction.PlatformTransactionManager;
import
org.springframework.transaction.annotation.EnableTransactionManagement;

import javax.persistence.EntityManager;
import javax.sql.DataSource;
import java.util.Map;

/**
 * 数据源一
 */
@Configuration
@EnableTransactionManagement
@EnableJpaRepositories(
        entityManagerFactoryRef="entityManagerFactoryPrimary",
        transactionManagerRef="transactionManagerPrimary",
        basePackages= { "com.jx.jpa.dao.primary" }) //设置Repository所在位置
public class PrimaryConfig {

    @Autowired
    @Qualifier("primaryDataSource")
    private DataSource primaryDataSource;
    @Primary
    @Bean(name = "entityManagerPrimary")
    public EntityManager entityManager(EntityManagerFactoryBuilder builder)
{
        return
```

```
entityManagerFactoryPrimary(builder).getObject().createEntityManager();
    }
    @Primary
    @Bean(name = "entityManagerFactoryPrimary")
    public LocalContainerEntityManagerFactoryBean
entityManagerFactoryPrimary (EntityManagerFactoryBuilder builder) {
        return builder
                .dataSource(primaryDataSource)
                .properties(getVendorProperties(primaryDataSource))
                .packages("com.jx.jpa.bean.primary") //设置实体类所在位置
                .persistenceUnit("primaryPersistenceUnit")
                .build();
    }
    @Autowired
    private JpaProperties jpaProperties;
    private Map<String,String> getVendorProperties(DataSource dataSource) {
        //如果数据源中存在不同类型的数据库，那么需要配置方言
        //jpaProperties.setDatabase(Database.MYSQL);

//jpaProperties.getProperties().put("hibernate.dialect","org.hibernate.dialect.
MySQL5Dialect");
        return jpaProperties.getHibernateProperties(dataSource);
    }

    @Primary
    @Bean(name = "transactionManagerPrimary")
    public PlatformTransactionManager
transactionManagerPrimary(EntityManagerFactoryBuilder builder) {
        return new
JpaTransactionManager(entityManagerFactoryPrimary(builder).getObject());
    }
}
```

配置第二个数据源

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.boot.autoconfigure.orm.jpa.JpaProperties;
import org.springframework.boot.orm.jpa.EntityManagerFactoryBuilder;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import
org.springframework.data.jpa.repository.config.EnableJpaRepositories;
import org.springframework.orm.jpa.JpaTransactionManager;
import
org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean;
import org.springframework.orm.jpa.vendor.Database;
import org.springframework.transaction.PlatformTransactionManager;
import
org.springframework.transaction.annotation.EnableTransactionManagement;

import javax.persistence.EntityManager;
import javax.sql.DataSource;
import java.util.Map;

/**
 * 数据源2
 */
@Configuration
@EnableTransactionManagement
@EnableJpaRepositories(
        entityManagerFactoryRef="entityManagerFactorySecondary",
        transactionManagerRef="transactionManagerSecondary",
        basePackages= { "com.jx.jpa.dao.secondary" }) //设置Repository所在位
置
public class SecondaryConfig {
```

```
    @Autowired
    @Qualifier("secondaryDataSource")
    private DataSource secondaryDataSource;
    @Bean(name = "entityManagerSecondary")
    public EntityManager entityManager(EntityManagerFactoryBuilder builder)
{
        return
entityManagerFactorySecondary(builder).getObject().createEntityManager();
    }
    @Bean(name = "entityManagerFactorySecondary")
    public LocalContainerEntityManagerFactoryBean
entityManagerFactorySecondary (EntityManagerFactoryBuilder builder) {
        return builder
                .dataSource(secondaryDataSource)
                .properties(getVendorProperties(secondaryDataSource))
                .packages("com.jx.jpa.bean.secondary") //设置实体类所在位置
                .persistenceUnit("primaryPersistenceUnit")
                .build();
    }

    @Autowired
    private JpaProperties jpaProperties;
    private Map<String,String> getVendorProperties(DataSource dataSource) {
        //如果数据源中存在不同类型的数据库，那么需要配置方言
        //jpaProperties.setDatabase(Database.ORACLE);

//jpaProperties.getProperties().put("hibernate.dialect","org.hibernate.dialect.
OracleDialect");
        return jpaProperties.getHibernateProperties(dataSource);
    }

    @Bean(name = "transactionManagerSecondary")
    PlatformTransactionManager
```

```
transactionManagerSecondary(EntityManagerFactoryBuilder builder) {
        return new
JpaTransactionManager(entityManagerFactorySecondary(builder).getObject()
);
    }


}
```

第一个DAO

```
package com.jx.jpa.dao.primary;

import org.springframework.data.jpa.repository.JpaRepository;

import com.jx.jpa.bean.primary.Temp1;

/**
 *
 */
public interface Temp1Repository extends JpaRepository<Temp1, Long>{

}
```

第二个DAO

```
package com.jx.jpa.dao.secondary;

import org.springframework.data.jpa.repository.JpaRepository;
```

```
import com.jx.jpa.bean.secondary.Temp2;

/**
 *
 */
public interface Temp2Repository extends JpaRepository<Temp2, Long>{

}
```

Service

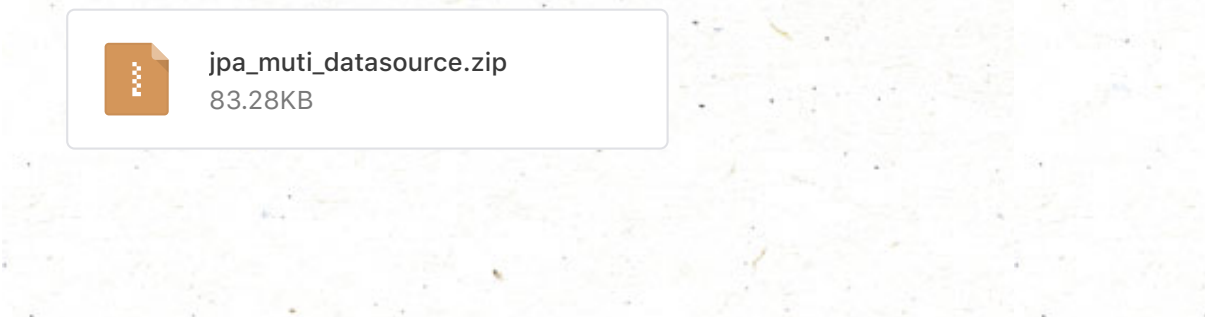
```
import javax.transaction.Transactional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.jx.jpa.bean.primary.Temp1;
import com.jx.jpa.bean.secondary.Temp2;
import com.jx.jpa.dao.primary.Temp1Repository;
import com.jx.jpa.dao.secondary.Temp2Repository;

@Service
public class TempService {

    @Autowired
    private Temp1Repository temp1Repository;

    @Autowired
    private Temp2Repository temp2Repository;

    public void saveTemp1(Temp1 temp1) {
        temp1Repository.save(temp1);
```

```
    }

    public void saveTemp2(Temp2 temp2) {
        temp2Repository.save(temp2);
    }

    @Transactional
    public void saveTemp() {
        Temp1 t1 = new Temp1();
        t1.setId(1);
        t1.setName("temp1");
        temp1Repository.save(t1);
        int i = 1/0;
        Temp2 t2 = new Temp2();
        t2.setId(1);
        t2.setName("temp2");
        temp2Repository.save(t2);
    }
}
```

controller

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;

import com.jx.jpa.bean.primary.Temp1;
import com.jx.jpa.bean.secondary.Temp2;
import com.jx.jpa.service.TempService;

/**
```

```
 * @author qiyang
 * 2018年4月3日 下午2:43:02
 * phone 13554046541
 * q348002671@qq.com
 */
@Controller
@ResponseBody
public class TempController {

	@Autowired
	private TempService tempService;

	@RequestMapping("save.do")
	public void saveTemp() {
		Temp1 t1 = new Temp1();
		t1.setId(3);
		t1.setName("temp1");

		tempService.saveTemp1(t1);

		Temp2 t2 = new Temp2();
		t2.setId(6);
		t2.setName("temp2");

		tempService.saveTemp2(t2);
	}

	@RequestMapping("save2.do")
	public String save2Temp() {
		tempService.saveTemp();
		return "ok";
	}
}
```


jpa_muti_datasource.zip
83.28KB