1. Spring JdbcTemplate的基本使用
2. jdbcTemplate概述

JdbcTemplate是Spring框架中提供的一个对象，它对原始繁琐的Jdbc API对象进行简单的封装，使得操作JDBC会更加的容易。Spring框架中集成了很多的操作模板类，如操作关系型数据库的JdbcTemplate和HibernateTemplate，操作nosql数据库的RedisTemplate，操作消息队列的JmsTemplate，这些模板类相当于工具类，我们直接使用即可。

1. JdbcTemplate开发步骤
2. 导入spring-jdbc和spring-tx坐标，两个jar包的版本最好一致吧。
3. 创建数据库表和实体，实体即对象，我们操作数据库是为了将数据从数据库中拿出给对象或者将对象的数据保存到数据库中。
4. 创建JdbcTemplate对象之前首先要创建数据源对象DataSource，采用c3p0或者druid都可以。设置完数据源的驱动，URL，用户名和密码后，将数据源设置到JdbcTemplate对象上，用该对象执行SQL操作。

ComboPooledDataSource dataSource = new ComboPooledDataSource();

dataSource.setDriverClass(“com.mysql.cj.jdbc.Driver”);

dataSource.setJdbcUrl(“jdbc:mysql://localhost:3306/test”);

dataSource.setUser(“root”);

dataSource.setPassword(“root”);

JdbcTemplate jdbcTemplate = new JdbcTemplate();

jdbcTemplate.setDataSource(dataSource);

jdbcTemplate.update(“insert into account values(?,?), “tom”, 500);

1. 执行数据库操作

jdbcTemplate.update(“insert into account values(?,?), “tom”, 500);

1. Spring产生JdbcTemplate对象

要让Spring框架来产生JdbcTemplate对象，就要将JdbcTemplate对象实例为一个Bean对象，同时将DataSource对象实例化为bean对象，注入到JdbcTemplate对象中。

在使用是，用getBean方法获取相应的对象，执行sql语句。

<bean id=”dataSource” class=”com.mchange.v3.c3p0.CombopooledDataSource”>

<property name=”driverClass” value=”com.mysql.cj.jdbc.Driver”/>

<property name=”jdbcUrl” value=”jdbc:mysql:///test”/>

<property name=”user” value=”root”/>

<property name=”password” value=root””/>

</bean>

<bean id=”jdbcTemplate” class=”org.springframework.jdbc.core.JdbcTemplate”>

<property name=”dataSource” ref=”dataSource” />

</bean>

1. 抽取jdbc.properties配置文件

直接将数据库的连接信息硬编码在applicationContext.xml中不太合适，一般将数据库的连接信息抽取到jdbc.properties配置文件中，专门存储数据库的配置信息。那么在applicationContext.xml中就要加载该peroperties文件。

使用context命名空间，首先导入context命名空间，步骤不再赘述。利用context命名空间中的property-placeholder来加载properties文件：

<context:property-placeholder location=”classpath:jdbc.properties”/>

然后将每个property子标签中的value属性值用EL表达式代替，用于抽取properties配置文件中的键值对的值：

<property name=”driverClass” value=”${jdbc.driver}”/>

这样，properties配置文件中的键值对就可以使用了。

1. jdbcTemplate查询多个对象

jdbcTemplate查询数据库时，使用query方法，第一个参数为sql语句，第二个参数为RowMapper接口，这个接口用于对数据库返回的数据进行数据封装的操作，可以直接封装称为一个类对象。我们需要传入一个实现类BeanPropertyRowMapper<>()，尖括号内为要封装的类对象，圆括号内为该类对象的字节码。

List<Account> accountList = jdbcTemplate.query(“select \* from account”, new BeanPropertyRowMapper<Account>(Account.class));

1. jdbcTemplate查询单个对象

使用queryForObject方法查询单个对象：

1. 如果该对象是一个实体对象

传入sql语句，BeanPropertyRowMapper实现类，以及一个Object:

Account tom = jdbcTemplate.queryForObject(“select \* from account where name=?”, new BeanPropertyRowMapper<Account>(Account.class), “tom”);

1. 如果该对象是简单类型

使用queryForObject，只需传入sql语句和简单类型的字节码：

jdbcTemplate.queryForObject(“select count(\*) from account”, Long.class);