第一步

配置c3p0数据源

1. **创建一个properties文件**

user=root

password=123456

driverClass=com.mysql.jdbc.Driver

jdbcUrl=jdbc:mysql:///test

initPoolSize=5

maxPoolSize=10

1. **加载到IOC容器中**

<context:property-placeholder location=*"db.properties"* />

1. **配置数据库连接池对象，利用spel**

<bean id=*"dataSource"* class=*"com.mchange.v2.c3p0.ComboPooledDataSource"*>

<property name=*"user"* value=*"${user}"*></property>

<property name=*"password"* value=*"${password}"*></property>

<property name=*"jdbcUrl"* value=*"${jdbcUrl}"*></property>

<property name=*"driverClass"* value=*"${driverClass}"*></property>

<property name=*"initialPoolSize"* value=*"${initPoolSize}"*></property>

<property name=*"maxPoolSize"* value=*"${maxPoolSize}"*></property>

</bean>

第二步

配置JdbcTemplate实例Bean对象

<!-- 配置Spring的jdbcTemplate -->

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

<!-- 有一个属性，数据库连接池 -->

<property name=*"dataSource"* ref=*"dataSource"*></property>

</bean>

第三步

使用JdbcTemplate

**private** ApplicationContext context;

**private** JdbcTemplate jdbcTemplate;

{ //初始化

context = **new** ClassPathXmlApplicationContext("applicationContext.xml");

jdbcTemplate = (JdbcTemplate) context.getBean("jdbcTemplate");

}

1. **更新操作**

直接调用update方法，传入sql语句和占位符参数

String sql = "update user set username=? where id=?";

jdbcTemplate.update(sql, "小绿", 3);

1. **批量更新**

调用batchUpdate方法，传入sql语句和Object[ ]对象的list(一条语句多组属性)

String sql = "insert into user(username,age) value(?,?)";

List<Object[]> list = **new** ArrayList<Object[]>();

list.add(**new** Object[] { "小美", 22 });

list.add(**new** Object[] { "小灰", 21 });

list.add(**new** Object[] { "小紫", 20 });

jdbcTemplate.batchUpdate(sql, list);

1. **从数据库中获取一条记录返回一个bean对象**

调用 queryForObject(String sql, RowMapper<User> rowMapper, Object... args)

RowMapper实现类使用BeanPropertyRowMapper，传入bean.class参数

String sql = "select \* from User where id=?";

RowMapper<User> rowMapper = **new** BeanPropertyRowMapper<>(User.**class**);

User user = jdbcTemplate.queryForObject(sql, rowMapper, 1);

1. **从数据库中查询多条记录，返回一个Bean的list集合**

使用query(String sql, RowMapper<User> rowMapper, Object... args)方法

String sql = "select \* from User where id>?";

RowMapper<User> rowMapper = **new** BeanPropertyRowMapper<>(User.**class**);

List<User> users = jdbcTemplate.query(sql, rowMapper, 3);

1. **从数据中获取单个值，如统计**

使用queryForObject(String sql, Class<Long> requiredType)方法，传入sql语句和获取值类型的.Class

String sql = "select count(1) from user";

Long num = jdbcTemplate.queryForObject(sql, Long.**class**);