40_性能优化：基于 Druid 连接池进行数据操作，提升系统性能

1、开篇

上节课对 JDBC 的代码做了简单的测试，同时反思 JDBC 编写访问数据库代码时会遇到的问题。例如数据库连接的管理和利用问题，SQL 语句硬编码不利于维护的问题，以及返回数据集需要额外处理的问题。这些问题我们会在接下来的课程中逐一给大家介绍，这节课我们会使用 Druid 连接池进行数据操作，提升系统性能。今天的内容：

- Druid 连接池介绍
- Druid 连接池配置
- Druid 连接池代码应用

2、Druid 连接池介绍

前面几次课在介绍 JDBC 访问数据库时，提到了需要在访问数据库时建立与数据库的连接，在当使用完数据库连接的时候需要做释放连接的操作。由于，建立连接和释放连接需要耗费大量的系统资源，对于访问数据库这个经常使用的功能而言是需要进行优化的。

因此这里需要引入数据库连接池的概念，数据库连接池的基本思想就是为数据库连接建立一个“缓冲池”。预先在缓冲池中放入一定数量的连接，当需要建立数据库连接时，只需从“缓冲池”中取出一个，使用完毕之后再放回去。我们可以通过设定连接池最大连接数来防止系统无尽的与数据库连接。更为重要的是我们可以通过连接池的管理机制监视数据库的连接的数量、使用情况，为系统开发、测试及性能调整提供依据。

市面上优秀的数据库连接池有不少，这里我们选用 Druid 连接池介绍给大家做介绍。

Druid 是 Java 语言中最好的数据库连接池，在功能、性能、扩展性方面，都超过其他数据库连接池，包括 DBCP、C3P0、Proxool、JBoss DataSource。Druid 已经在阿里巴巴部署了超过 600 个应用，经过生产环境大规模部署的严苛考验。

Druid 连接池为监控而生，内置强大的监控功能，监控特性不影响整体性能。功能强大，能防 SQL 注入，内置 Loging 能诊断 Hack 应用行为。同时 Druid 支持所有 JDBC 兼容的数据库，包括 Oracle、MySql、Derby、Postgresql、SQL Server、H2 等等。Druid 针对 Oracle 和 MySql 做了特别优化，比如 Oracle 的 PS Cache 内存占用优化，MySql 的 ping 检测优化。

3、Druid 连接池配置

介绍完了 Druid 连接池，回到项目的代码中，如图 1 所示要使用 Druid 首先需要在 pom.xml 中配置 druid 的依赖信息。

图 1 配置 Druid 的依赖

在配置完依赖以后，在 resources/spring-jdbc.xml 文件中加入 bean 定义。如图 2 所示，其名字叫做 dataSource，其中定义了 Druid 连接池的基本信息。在定义的 dataSource 的 bean 中指出引用的类是“com.alibaba.druid.pool.DruidDataSource”。

同时制定 driverClassName、url、username 以及 password，这些关于数据库驱动，连接串、用户名、密码的信息依旧通过参数的方式从 jdbc.properties 文件中读出。接下来通过定义 initialSize 定义初始化的数据库物理连接数为 5，

minIdle 定义最小连接数为 5，maxActive 最大连接数为 20，最大连接等待时间为 6000 毫秒以及通过配置 filters 制定 SQL 监控页面显示的 SQL 语句。

图 2 Druid 连接池配置 ，修改 spring-jdbc.xml 文件

在配置完了 dataSource 之后在配置 jdbcDruidBaseDao 的 bean 实例，它会去引用 dataSource 的 bean。jdbcDruidBaseDao 作为主要实现 Druid 连接池的基础类，我们在下面会接着介绍。

4、Druid 连接池代码应用

说完了 Druid 数据库连接池的配置以后，再看 jdbcDruidBaseDao 中是如何使用 Druid 进行数据库操作的。如图 3 所示，在 spring/dao 目录下面建立 JdbcDruidBaseDao 类，从上面的配置文件定义可以知道，该类会引用 dataSource（Druid 连接池）。

在 JdbcDruidBaseDao 的构造函数中会引入 dataSource，同时在 getConnection 方法会获取 dataSource 作为 Druid 的数据库连接池，这也是与 JDBC 直接访问数据库的区别。jdbcDruidBaseDao 是通过 Druid 数据库连接池的方式去访问数据库的，也就是说通过 Druid 连接池优化数据库的连接功能，提高访问数据库的效率。

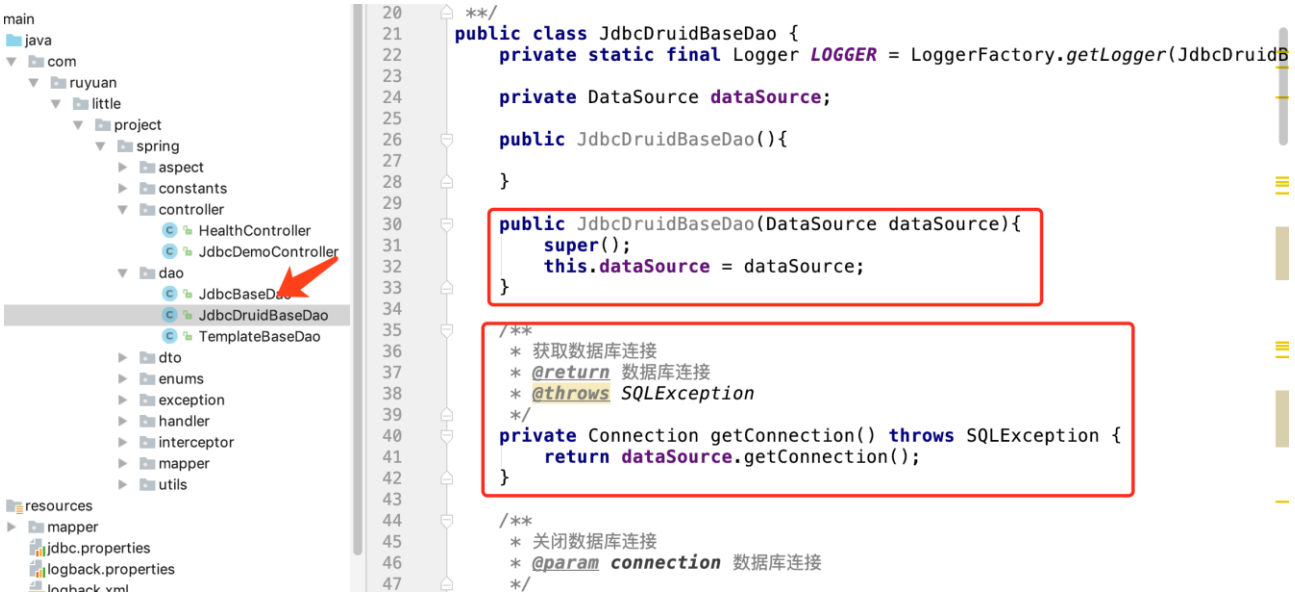

图 3 JdbcDruidBaseDao 类的编写

接着往下看在 JdbcDruidBaseDao 类的方法中会将之前直接连接数据库的方法更换为使用 Druid 连接池的方法。如图 4 所示，queryList 方法中使用 getConnect（）获取连接池资源，同时使用 closeConnection（）方法释放连接池资源，这里也是和 JDBC 直接访问数据库的区别。

```
public <T> List<T> queryList(String sql, Object[] params, Class<T> resu
    List<T> result = new ArrayList<>();
    Connection conn = null;
    PreparedStatement preparedStatement = null;
    ResultSet resultSet = null;
    try {
        conn = getConnection();
        preparedStatement = conn.prepareStatement(sql);
        resultSet = executeQuery(preparedStatement,params);
        while (resultSet.next()) {
            T o = resultClazz.newInstance();
            Field[] fields = resultClazz.getDeclaredFields();
            for (Field field : fields) {
                field.setAccessible(true);
                String fieldName = field.getName();
                try {
                    Object value = resultSet.getObject(fieldName);
                    field.set(o, value);
                }catch (SQLException e){
                    LOGGER.debug("结果集中无 {} 参数", fieldName);
                }
            }
            result.add(o);
        }
    } finally {
        CloseableUtils.close(resultSet);
        CloseableUtils.close(preparedStatement);
        closeConnection(conn);
    }
    return result;
}
```

图 4 在数据库访问方法中修改了数据库连接串的使用方式

最后，来看看在 JdbcDemoController 中访问 Druid 数据库连接池的入口函数，如图 5 所示，JdbcDemoController 中 jdbcDruidGetAll 方法，其内容和其他的 JDBC 方法没有区别也是传入 SQL 语句利用，jdbcDruidBaseDao 查询数据库，并且返回信息。

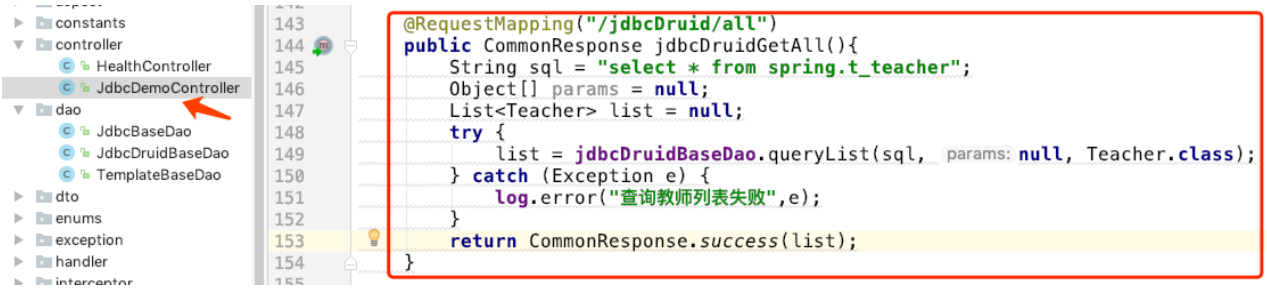

图 5 在 Controller 中 Druid 访问数据库的入口

5、总结

这节课从连接数据库的操作做为切入口，提出了数据库连接池的概念，这个概念的引入帮助提升数据库操作的性能问题。同时在现有的项目中加入了 Druid 连接池配置，并且通过添加 Druid 连接池访问的类，帮助我们使用连接池进行数据库操作。下期我们会测试本期的代码，然后思考一下在数据库操作中还有哪些可以提升的空间。我们下期见，拜拜。