武汉纺织大学

Web应用开发课程设计

**基于Highcharts组件的数据可视化系统**

**学 院： 数学与计算机学院**

**班 级： 物联网11801**

**姓 名： 王玉蛟**

**学 号： 180424204**

**指导老师： 聂刚**

**成 绩：**

**完成日期： 2020年12月15日**

目 录

[1 需求分析 1](#_Toc11169)

[2 系统设计 1](#_Toc22967)

[2.1 UML类图（Class Diagram） 1](#_Toc13506)

[2.1.1 ware类 1](#_Toc15151)

[2.1.2countryweather类 1](#_Toc19181)

[2.1.3 Browser类 1](#_Toc13718)

[3 系统实现 2](#_Toc24350)

[3.1 项目结构 2](#_Toc16492)

[3.2 配置文件 2](#_Toc29572)

[3.2.1 jdbc.properties文件 2](#_Toc17186)

[3.2.3 mybatis-config.xml文件 3](#_Toc4033)

[3.3 VO类 3](#_Toc21739)

[3.3.1 Ware.java 3](#_Toc11275)

[3.3.2 Browser.java 3](#_Toc31682)

[3.3.1 CountryWeath.java 3](#_Toc29320)

[3.4 DAO接口类 4](#_Toc27881)

[3.4.1 BrowserDao.java 4](#_Toc28763)

[3.4.2 CountryWeatherDao.java 5](#_Toc8760)

[3.4.3 WareDao.java 6](#_Toc23606)

[3.5 工具包Util 6](#_Toc29418)

[4 系统测试 7](#_Toc1654)

[4.1折线图。 7](#_Toc5067)

[4.2 饼状图 8](#_Toc26380)

[4.3条形图 9](#_Toc16893)

[5 系统总结 10](#_Toc8128)

# 1 需求分析

使用数据库查询到对应的数据表之后，讲数据存放在data中利用highchart设计生成三个图表：条形图，饼状图和折线图。使信息更加直观有条理，浏览者能快速获得自己想要的数据。

# **2 系统设计**

## 2.1 UML类图（Class Diagram）

### 2.1.1 ware类

* Vo实体类ware：与数据库结构进行映射的类。主要由属性，setter, getter方法组成，VO类中的属性与表中的字段相对应，每一个VO类的对象都表示表中的每一条记录。
* DAO接口wareDao：主要定义操作的接口，定义一系列数据库的原子性操作
* service业务逻辑实现类SaleService：对于数据层的原子操作进行整合。
* servlet工厂类SaleServlet：创建wareServlet，

### 2.1.2countryweather类

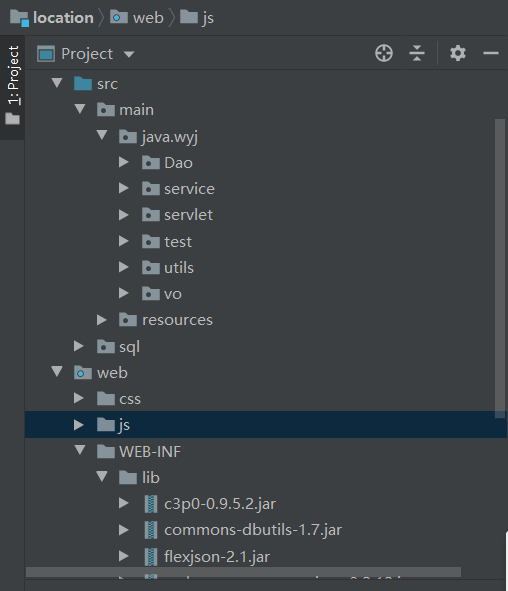
* Vo实体类CountryWeather：与数据库结构进行映射的类。主要由属性，setter, getter方法组成，VO类中的属性与表中的字段相对应，每一个VO类的对象都表示表中的每一条记录。
* DAO接口CountryWeatherDao：主要定义操作的接口，定义一系列数据库的原子性操作。
* service业务逻辑实现类CountryWeatherService：对于数据层的原子操作进行整合
* servlet工厂类CountryWeatherServlet：创建CountryWeatherServlet，

### 2.1.3 Browser类

* Vo实体类browser：与数据库结构进行映射的类。主要由属性，setter, getter方法组成，VO类中的属性与表中的字段相对应，每一个VO类的对象都表示表中的每一条记录。
* DAO接口browserDao：主要定义操作的接口，定义一系列数据库的原子性操作。
* service业务逻辑实现类BrowserService：对于数据层的原子操作进行整合
* servlet工厂类BrowserServlet：创建BrowserServlet，

# **3 系统实现**

## 3.1 项目结构



## 3.2 配置文件

在项目下新建Source Folder，名为resources

### 3.2.1 c3p0-config文件

<?xml version="1.0" encoding="UTF-8"?>  
  
<c3p0-config>  
  
 <!-- 默认配置 -->  
  
 <default-config>  
  
 <!-- 配置数据库连接信息 -->  
  
 <property name="user">root</property>  
  
 <property name="password">wwwyyy</property>  
  
 <property name="jdbcUrl">jdbc:mysql://localhost:8080/test</property>  
  
 <property name="driverClass">com.mysql.jdbc.Driver</property>  
  
 </default-config>  
  
</c3p0-config>

## 3.3 VO类

### 3.3.1 Ware.java

**public class Ware {**

**private int id;**

**private String name;**

**private double amount;**//构造方法、getter,setter，toString方法,自行补充

}

### 3.3.2 Browser.java

**public class Browser {**

**private int id;**

**private String name;**

**private double percent;**

//构造方法、getter,setter，toString方法,自行补充

}

### 3.3.1 CountryWeath.java

**public class CountryWeather {**

**private int id;**

**private String name;**

**private int month;**

**private double temp;**

//构造方法、getter,setter，toString方法,自行补充

}

## 3.4 DAO接口类

### 3.4.1 BrowserDao.java

public class BrowserDao {  
 private final QueryRunner runner = new QueryRunner(DataSourceUtils.*getDataSource*());  
 private Connection con;  
 private PreparedStatement pst;  
  
 public List<Browser> findAllBrowsers(){  
  
 try {  
 List<Browser> list = new ArrayList<>();  
 Jdbc j = new Jdbc();  
 con = j.getConnection();  
  
 list = getAllBrowser();  
  
 return list;  
  
  
 } catch (SQLException e) {  
  
 e.printStackTrace();  
  
 throw new RuntimeException(e);  
  
 }  
  
 }  
  
 public List<Browser> getAllBrowser() throws SQLException {  
 List<Browser> list = new ArrayList<>();  
 String sql = "select \* from browser";  
  
 pst = con.prepareStatement(sql);  
 ResultSet rs = pst.executeQuery();  
  
 while (rs.next()){  
 Browser browser = new Browser();  
  
 browser.setId(rs.getInt(1));  
 browser.setName(rs.getString(2));  
 browser.setPercent(rs.getDouble(3));  
  
 list.add(browser);  
 }  
  
 return list;  
 }  
}

### **3.4.2 CountryWeatherDao.java**

public class CountryWeatherDao {  
 //通过数据类得到数据对象  
  
 private final QueryRunner runner = new QueryRunner(DataSourceUtils.*getDataSource*());  
  
 //查询所有的天气  
  
 public List<CountryWeather> findAllCountryWeathers() {  
  
 try {  
  
 return runner.query("select \* from country\_weather", new BeanListHandler<>(CountryWeather.class));  
  
 } catch (SQLException e) {  
  
 e.printStackTrace();  
  
 throw new RuntimeException(e);  
  
 }  
  
 }  
}

### 3.4.3 WareDao.java

public class WareDao {  
 //数据源的结果传给QueryRunner  
  
 private final QueryRunner runner = new QueryRunner(DataSourceUtils.*getDataSource*());  
  
  
  
 public List<Ware> findAllWares() {  
  
 try {  
  
 return runner.query("select \* from ware", new BeanListHandler<>(Ware.class));  
  
 } catch (SQLException e) {  
  
 e.printStackTrace();  
  
 throw new RuntimeException(e);  
  
 }  
  
 }  
}

## 3.5 工具包Util

DataSourceUtils.java

package main.utils;

import com.mchange.v2.c3p0.ComboPooledDataSource;

import javax.sql.DataSource;

public class DataSourceUtils {

//创建一个连接池

private static final ComboPooledDataSource ds = new ComboPooledDataSource();

public static DataSource getDataSource() {

return ds;

}

}

**static** {

String resourse = "mybatis-config.xml";

**try** {

InputStream inputStream = Resources.*getResourceAsStream*(resourse);

*sqlSessionFactory* = **new** SqlSessionFactoryBuilder().build(inputStream);

} **catch** (IOException e) {

e.printStackTrace();

}

}

**public** **static** SqlSession getSqlSession() {

SqlSession sqlSession = *sqlSessionFactory*.openSession();

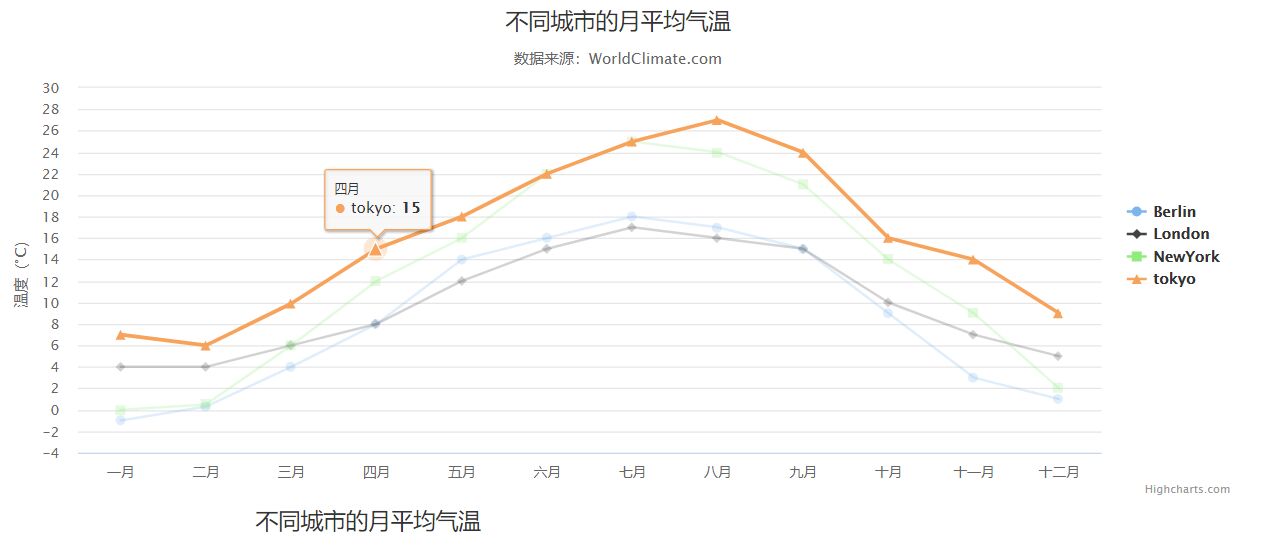
**return** sqlSession;

}

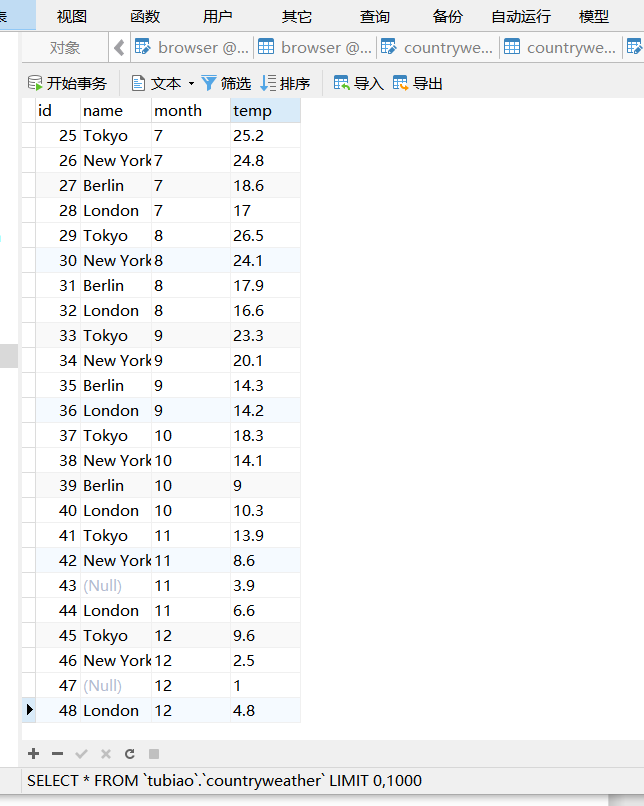
}

# **4 系统测试**

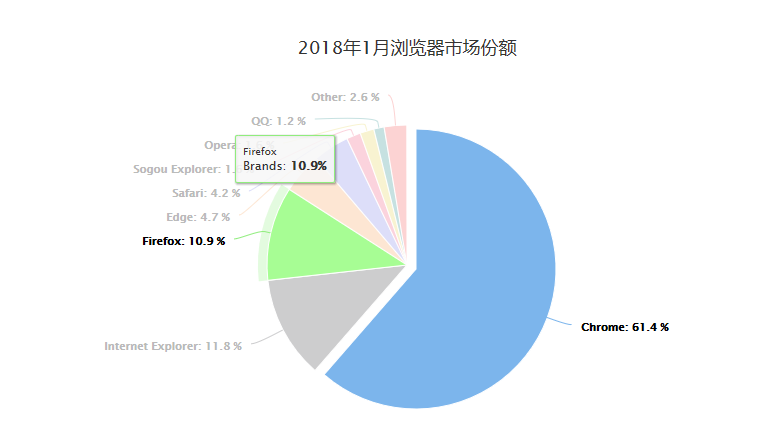
## 4.1折线图。



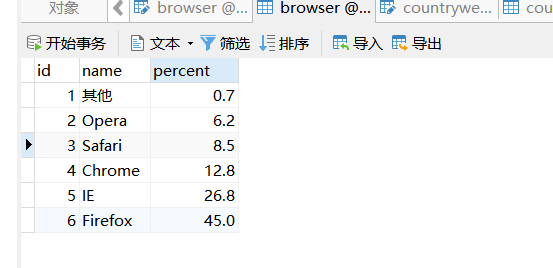
使用的数据库如图

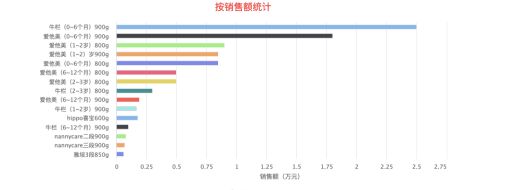
## 4.2 饼状图



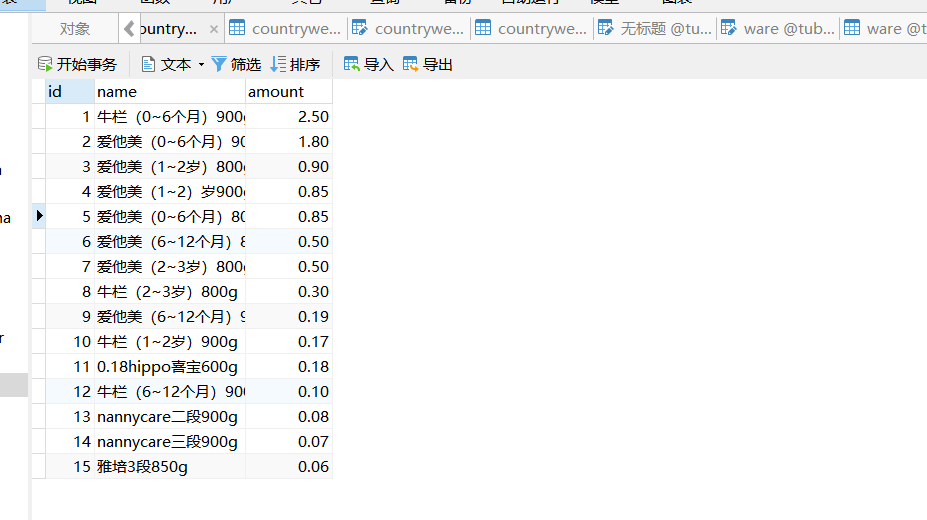
数据库内容如图



## 4.3条形图



使用的数据库内容如图



# **5 系统总结**

这次的实验涉及到了自己从来没接触过的领域，从highchart官网学习到了生成图表的构建，但是没有关于如何将数据库的数据处理成所需要的图表的格式，后来通过哔哩哔哩的网站中一步步学习利用c3p0连接池进行数据库的链接，然后使用sevlet处理数据，再将数据处理成图表所需要的格式，最后通过highchart组件来生成图表。但还是存在着很多不足，比如图表生成不够多变，目前只学会了这三个图表的生成方式。之后我也会学习web的其他知识，多看视频跟着视频实际的去实现例子修改例子，变成自己的东西。努力跟上web知识的更新速度，活到老学到老。