

网页设计说明书

|  |  |
| --- | --- |
| 课程 | 软件技术基础 |
| 学 院(系) | 中德学院 |
| 专 业 | 机械工程2017级 |
| 学 号 | 1733327 |
| 学生姓名 | 杨清峰 |
| 指导教师 | 沈斌 |
| 日 期 | 2017年1月04日 |

目录

[基于 JSP 的动态网站说明书 1](#_Toc503349465)

[一、软件环境及工具 1](#_Toc503349466)

[二、网页结构及原理 1](#_Toc503349467)

[1. 体系结构（如图1所示）用户通过浏览器向JSP发出一个请求，JSP根据需要调javaBean，大部分的数据存取，逻辑运算等由javaBean来完成，当javaBean处理完成之后再回转到JSP，JSP显示最后的结果。 1](#_Toc503349468)

[三、功能介绍 2](#_Toc503349469)

[3.1.主要功能 2](#_Toc503349470)

[3.2.查询功能 3](#_Toc503349471)

[3.3.高级查询功能 4](#_Toc503349472)

[3.4.显示详细信息功能 5](#_Toc503349473)

[3.5.添加记录功能 6](#_Toc503349474)

[3.6.管理产品（或零件）记录信息功能 9](#_Toc503349475)

[3.7 BOM表显示功能 11](#_Toc503349476)

[四、程序实例 12](#_Toc503349477)

[4.1首页程序代码 12](#_Toc503349478)

[4.2数据库操作层 16](#_Toc503349479)

[五、操作说明 27](#_Toc503349480)

[六、小结 27](#_Toc503349481)

基于 JSP 的动态网站说明书

# 一、软件环境及工具

JDK 版本：8.0

Tomcat 版本：7.0

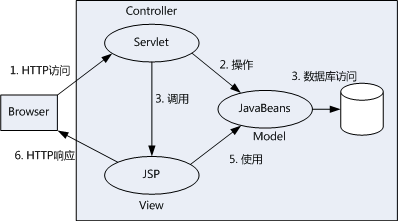
Mysql数据库 版本：5.7.20

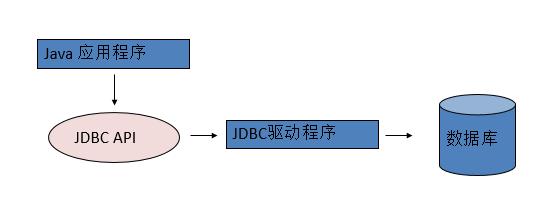
jar包 版本：mysql-connector-java-5.1.45

开发平台 MyEclipse10

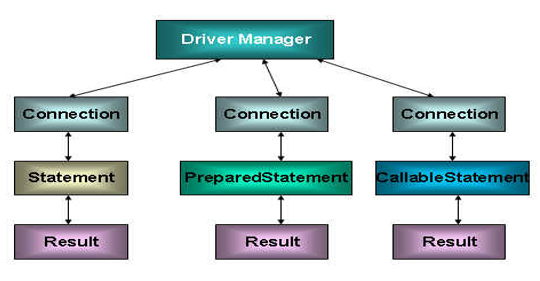
# 二、网页结构及原理

1. 体系结构（如图1所示）用户通过浏览器向JSP发出一个请求，JSP根据需要调javaBean，大部分的数据存取，逻辑运算等由javaBean来完成，当javaBean处理完成之后再回转到JSP，JSP显示最后的结果。





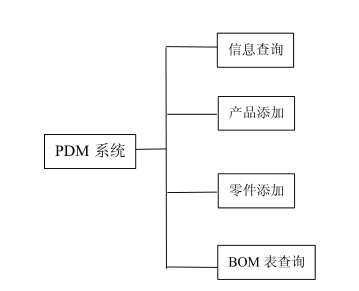
JDBC API 是一系列的接口，它使得应用程序能够进行数据库联接，执行SQL语句，并且得到返回结果。



# 三、功能介绍

## 3.1.主要功能

该系统主要功能模块有：信息查询，产品添加，零件添加，产品BOM表查询。具体结构如下：



通过网页站点对产品及零件数据库进行模糊查询，高级查询，显示产品和零件的详细信息：有添加，编辑修改和删除产品及零件信息的功能；通过BOM表可描述整个数据库的结构总图。

## 3.2.查询功能

系统主界面如下：

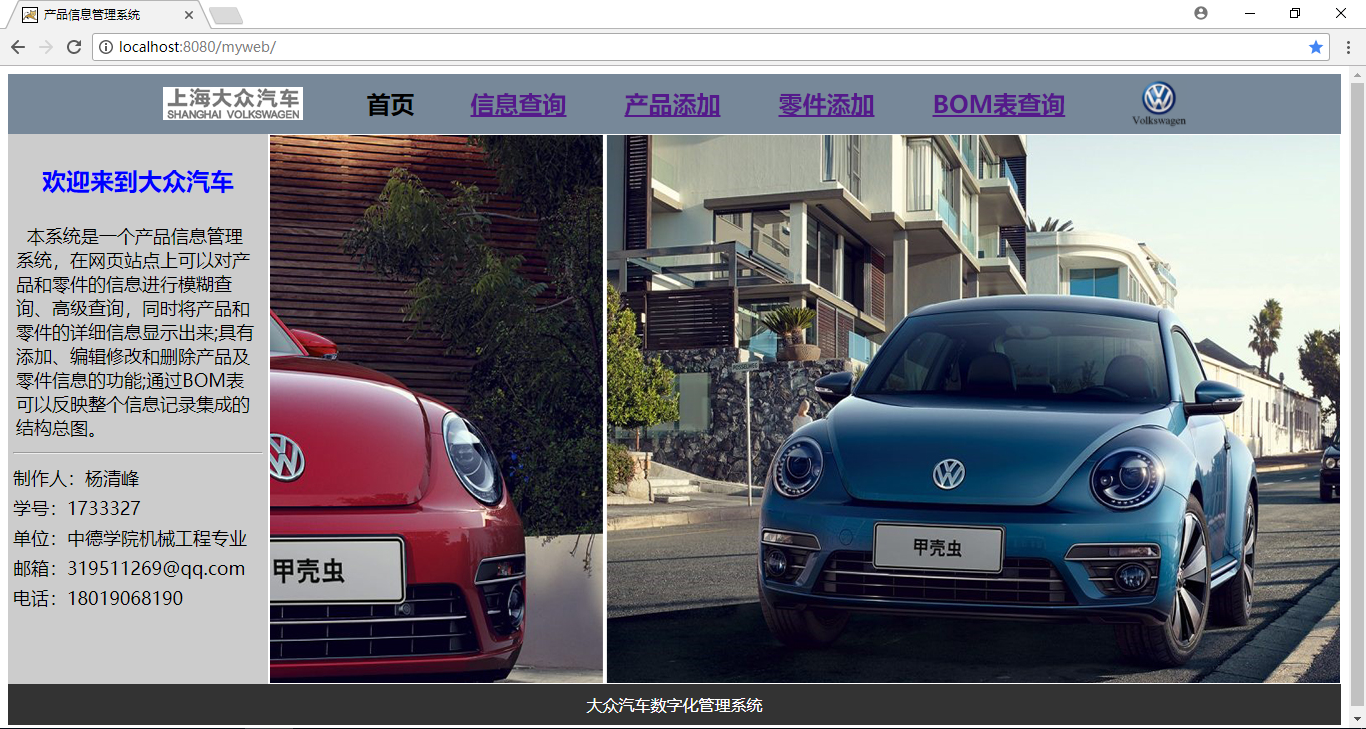


图 3

点击信息查询菜单，进入以下记录信息查询界面：



图 4

例如：在查询产品记录中选择产品编号，则可以进行精确查询。如输入001，点击查询可以得到如下结果：



图 5

## 3.3.高级查询功能

记录信息查询页面，如图（6）所示，在此页面中可以针对产品和零件记录进行高级查询，条件中可以选择查询的字段（型号，名称，制造商，产地和价格等），在产品查询中还有价格的选择表达式（等于，大于和小于），输入关键字查询，还可以选择条件与条件之间的“或者”与“并且”的逻辑关系。

例如在产品信息高级查询表单中，输入如图（6）的条件，其中产品编号为001，并且价格选择大于100000，则会显示如图（7）的查询结果。



图 6



图 7

产品记录信息查询页中除了显示产品本身的信息外，还提供了零件信息编辑和删除的链接。

如果没有找到相关产品或零件记录则会在界面上显示如图（8）所示的提示信息。



图 8

## 3.4.显示详细信息功能

完成查询后（如图6和图7），在界面上显示了产品的查询结果。此时可以在产品查询列表中点击“附属零件信息”，查看此产品的附属零件记录，如图（9）所示。

图 9

## 3.5.添加记录功能

3.5.1添加产品记录

点击页面上的“添加产品”的链接，则进入到添加产品记录的页面，如图（10）所示。在这个页面中输入产品记录信息，注意：产品编号，产品型号，产品名称为必填，产品价格必须为数字，否则系统会报错，例如在价格一栏输入“同济大学”，则可得到图（11）。



图 10



图 11

正确输入产品信息完成后，如图12，点击“添加”按钮进入到添加产品记录成功确认界面，如图（13）所示。可单击选择继续添加产品记录，继续添加下一个记录。



图13



图 14

3.5.2添加零件记录

点击页面上的“添加零件”的链接，则进入到添加零件记录的页面，产品型号是已经添加了的产品，拉下拉菜单可选择相应所属产品的类型，如选择 Lavida，则如图（14）所示。在这个页面中输入零件记录信息，注意：零件编号、零件型号、零件名称、所属产品型号为必填项，零件价格必须为数字，否则系统会报错。输入零件信息完成以后，点击“添加”按钮进入到添加零件记录成功确认界面，如 图（15）所示。如果需要继续添加零件记录，可以在确认界面中点击后重新进入 到添加零件记录的页面。



图 14



图 15

## 3.6.管理产品（或零件）记录信息功能

3.6.1更显产品（或零件记录）

前文已提到，完成查询后，在查询结果页中除了显示产品（或零件）本身的 信息以外，还提供了信息编辑的链接(如图 5、图 7 和图 9所示)。在页面中如果在相应记录处点击“编辑”，那么自动进入产品（或零件）记录 更新页面，如图 (16) 所示。在这个页面中可以对产品（或零件）记录的内容进 行编辑修改，修改完成后点击“更新”则页面自动跳到产品（或零件）信息编辑成 功页面（图 17），此时再查看产品记录，发现产品记录已经被更新过了。



图 16



图 17

**3.6.2删除产品（或零件）记录**

与前面所讲的更新产品记录相同，完成查询后，在查询结果页中除了显示产 品本身的信息以外，还提供了信息删除的链接(如图 5、图 7 所示)。然后点击“删 除”按钮，则自动转到删除产品记录确认界面,以防误删，如图（18）所示。确认 后，跳到产品信息删除编辑成功页面（图 19）。





图 18

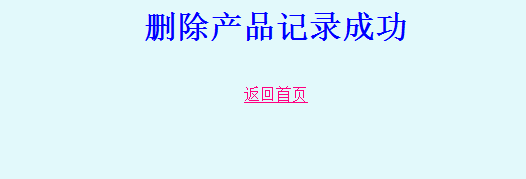


图 19

## 3.7 BOM表显示功能

在界面上点击 BOM 表链接后，则页面跳转到 BOM 表页面。点击“车型及 相关零件”前的“>>>”，可以打开产品目录，点击任意产品前的“>>>”，可以再打开该产品的零件目录。相反分支打开后，点击“----” 具有关闭分支的功能，并且该BOM还具有打开全部分支和关闭所有分支的功能，如图（20）。



图 20

另外，单击产品（或零件）名称可连接到相关产品（或零件），例如单击上海大众，可以得到信息页图（21），如单击发动机，得到图（22）。



图 21



图 22

# 四、程序实例

## 4.1首页程序代码

<%@ page language=*"java"* import=*"java.util.\*"* pageEncoding=*"UTF-8"*%>

<%

String path = request.getContextPath();

String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";

%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>

<head>

<base href=*"*<%=basePath%>*"*>

<title>产品信息管理系统</title>

<meta http-equiv=*"pragma"* content=*"no-cache"*>

<meta http-equiv=*"cache-control"* content=*"no-cache"*>

<meta http-equiv=*"expires"* content=*"0"*>

<meta http-equiv=*"keywords"* content=*"keyword1,keyword2,keyword3"*>

<meta http-equiv=*"description"* content=*"This is my page"*>

<style>

*#header* {

background-color:*#778899*;

color:*white*;

text-align:*center*;

padding:*2px*;

}

*#nav* {

line-height:*30px*;

background-color:*#CCCCCC*;

height:*540px*;

width:*250px*;

float:*left*;

padding:*5px*;

}

*#section* {

background-color:*#FFFFFF*;

width:*1070px*;

float:*right*;

padding:*1px*;

}

*#footer* {

background-color:*#333333*;

color:*white*;

clear:*both*;

text-align:*center*;

padding:*10px*;

}

</style>

</head>

<body>

<div id=*"header"*>

<table align=*"center"*>

<tr>

<td width=*"200"*><a href=*"http://cdhk.tongji.edu.cn/zh-hans/"* target=*"\_blank"*><img src=*"image/shanghai.jpg"*></a></td>

<td width=*"100"*><font size=*5*><b>首页</b></font></td>

<td width=*"150"*><a href=*"searchInfo.jsp"*><font size=*5*><b>信息查询</b></font></a></td>

<td width=*"150"*><a href=*"addProduct.jsp"*><font size=*5*><b>产品添加</b></font></a></td>

<td width=*"150"*><a href=*"addPart.jsp"*><font size=*5*><b>零件添加</b></font></a></td>

<td width=*"150"*><a href=*"BOM.jsp"*><font size=*5*><b>BOM表查询</b></font></a></td>

<td width=*"100"* align=*"right"*><a href=*"http://www.vw.com.cn/cn.html"*><img src=*"image/logo.jpg"* height=*"50"*></a></td>

</tr>

</table>

</div>

<div id=*"nav"*>

<table>

<tr>

<td width=*"600"* align=*"center"* height=*"80"*><font size=*"5"* face=*"仿宋\_GB2312"* color=*"#0000ff"*><strong>欢迎来到大众汽车</strong></font></td>

</tr>

<tr>

<td width=*"800"*><font size=*"4"* face=*"仿宋\_GB2312"*>&nbsp;&nbsp;本系统是一个产品信息管理系统，在网页站点上可以对产品和零件的信息进行模糊查询、高级查询，同时将产品和零件的详细信息显示出来;具有添加、编辑修改和删除产品及零件信息的功能;通过BOM表可以反映整个信息记录集成的结构总图。</font></td>

</tr>

</table>

<font face=*"楷体\_GB2312"* size=*"4"*>

<hr/>

制作人：杨清峰<br>

学号：1733327<br>

单位：中德学院机械工程专业<br>

邮箱：319511269@qq.com<br>

电话：18019068190</font>

</div>

<div id=*"section"*>

<marquee align=*"absmiddle"* direction=*"right"* behavior=*"scroll"* width=*"1070"* height=*"548"* loop=*"-1"* scrollamount=*"20"* onMouseOut="this.start()" onMouseOver="this.stop()">

<img src=*"image/Car00.jpg"*/>

<a href=*"http://127.0.0.1:8080/myweb/productInfo.jsp?id=7"*><img src=*"image/Car01.jpg"*/></a>

<img src=*"image/Car02.jpg"*/>

<img src=*"image/Car03.jpg"*/>

<img src=*"image/Car04.jpg"*/>

<img src=*"image/Car05.jpg"*/>

<img src=*"image/Car06.jpg"*/>

<a href=*"http://127.0.0.1:8080/myweb/productInfo.jsp?id=9"*><img src=*"image/Car07.jpg"*/></a>

</marquee>

</div>

<div id=*"footer"*>

大众汽车数字化管理系统

</div>

</body>

## 4.2数据库操作层

**package** com.dao;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**public** **class** ConnectionDao

{

**public** **static** Connection getConn()

{

Connection conn=**null**;

**try**

{

//加载驱动程序

Class.*forName*("com.mysql.jdbc.Driver");

Class.*forName*("com.mysql.jdbc.Driver").newInstance();

//获得数据库的连接

String URL = "jdbc:mysql://localhost:3306/product?user=root&password=520524&useUnicode=true&characterEncoding=UTF-8";

conn = DriverManager.*getConnection*(URL);

}

**catch** (Exception e)

{

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**return** conn;

}

}

**package** com.dao;

**import** java.sql.Connection;

**import** java.sql.PreparedStatement;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.util.ArrayList;

**import** java.util.List;

**import** com.entity.Part;

**import** com.entity.Product;

//数据层处理类，实现增删改查的功能

**public** **class** CommonDao

{

Connection conn;

//添加产品

**public** **int** insertProduct(Product product)

{

PreparedStatement pstmt = **null**;

**int** rs = 0;

String values = "'"+product.getProductId()+"','"+product.getProductType()+"','"+product.getProductName()+"','"+product.getProductManufacture()+"','"+product.getProductPlace()+"',"+product.getProductPrice();

String sql ="insert into ProductTable(productID,productType,productName,productManufacture,productPlace,productPrice) values ("+values+")";

**try**

{

//获得数据连接

conn = ConnectionDao.*getConn*();

//加载mysql驱动程序

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeUpdate();

pstmt.close();

conn.close();

}

**catch** (SQLException e)

{

e.printStackTrace();

}

**return** rs;

}

//更改产品信息

**public** **int** updateProduct(Product product)

{

PreparedStatement pstmt = **null**;

**int** rs = 0;

String set = "productID='" +product.getProductId()+ "',productType='"+product.getProductType()+"',productName='"+product.getProductName()+"',productManufacture='"+product.getProductManufacture()+"',productPlace='"+product.getProductPlace()+"',productPrice="+product.getProductPrice()+"";

String sql = "update ProductTable set "+set+" where ID="+product.getId();

**try**

{

conn = ConnectionDao.*getConn*();

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeUpdate();

pstmt.close();

conn.close();

}

**catch** (SQLException e)

{

e.printStackTrace();

}

**return** rs;

}

//删除产品

**public** **int** deleteProduct(Product product)

{

PreparedStatement pstmt = **null**;

**int** rs = 0;

String sql = "delete from ProductTable where ID="+product.getId();

**try**

{

//获得数据连接

conn = ConnectionDao.*getConn*();

//加载mysql驱动程序

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeUpdate();

deletePartByType(product.getProductType());

pstmt.close();

conn.close();

}

**catch** (SQLException e)

{

e.printStackTrace();

}

**return** rs;

}

//通过类型删除零件

**public** **int** deletePartByType(String type)

{

PreparedStatement pstmt = **null**;

**int** rs = 0;

String sql = "delete from PartTable where ProductType='"+type+"'";

**try**

{

conn = ConnectionDao.*getConn*();

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeUpdate();

pstmt.close();

conn.close();

}

**catch** (SQLException e)

{

e.printStackTrace();

}

**return** rs;

}

//增加零件

**public** **int** insertPart(Part part)

{

PreparedStatement pstmt = **null**;

**int** rs = 0;

String values = "'"+part.getPartId()+"','"+part.getPartName()+"','"+part.getProductType()+"','"+part.getPartManufacture()+"','"+part.getPartPlace()+"',"+part.getPartPrice();

String sql ="insert into PartTable(partID,partname,producttype,partManufacture,partPlace,partprice) values ("+values+")";

**try**

{

conn = ConnectionDao.*getConn*();

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeUpdate();

pstmt.close();

conn.close();

}

**catch** (SQLException e)

{

e.printStackTrace();

}

**return** rs;

}

//修改零件信息

**public** **int** updatePart(Part part)

{

PreparedStatement pstmt = **null**;

**int** rs = 0;

String set = "partID='"+part.getPartId()+"',partname='"+part.getPartName()+"',producttype='"+part.getProductType()+"',partManufacture='"+part.getPartManufacture()+"',partPlace='"+part.getPartPlace()+"',partprice="+part.getPartPrice();

String sql = "update PartTable set "+set+" where ID="+part.getId();

**try**

{

conn = ConnectionDao.*getConn*();

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeUpdate();

pstmt.close();

conn.close();

}

**catch** (SQLException e)

{

e.printStackTrace();

}

**return** rs;

}

//删除零件

**public** **int** deletePart(String id)

{

PreparedStatement pstmt = **null**;

**int** rs = 0;

String sql = "delete from PartTable where ID="+id;

**try**

{

conn = ConnectionDao.*getConn*();

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeUpdate();

pstmt.close();

conn.close();

}

**catch** (SQLException e)

{

e.printStackTrace();

}

**return** rs;

}

//通过产品类型查看零件

**public** List<Part> getPartByType(String type)

{

PreparedStatement pstmt = **null**;

List<Part> list=**new** ArrayList<Part>();

ResultSet rs = **null**;

String sql = "select \* from PartTable where ProductType = '"+type+"'";

**try**

{

conn = ConnectionDao.*getConn*();

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

Part part = **null**;

**while**(rs.next())

{

part = **new** Part();

part.setId(rs.getInt("id"));

part.setPartId(rs.getString("partId"));

part.setProductType(rs.getString("productType"));

part.setPartName(rs.getString("partName"));

part.setPartManufacture(rs.getString("partManufacture"));

part.setPartPlace(rs.getString("partPlace"));

part.setPartPrice(rs.getInt("partPrice"));

list.add(part);

}

rs.close();

pstmt.close();

conn.close();

} **catch** (SQLException e)

{

e.printStackTrace();

}

**return** list;

}

// 通过编号查询产品

**public** Product getProductById(**int** id)

{

PreparedStatement pstmt = **null**;

Product product = **null**;

ResultSet rs = **null**;

String sql = "select \* from ProductTable where Id=" + id;

**try**

{

conn = ConnectionDao.*getConn*();

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

**while**(rs.next())

{

product = **new** Product();

product.setId(rs.getInt("id"));

product.setProductId(rs.getString("productId"));

product.setProductType(rs.getString("productType"));

product.setProductName(rs.getString("productName"));

product.setProductManufacture(rs.getString("productManufacture"));

product.setProductPlace(rs.getString("productPlace"));

product.setProductPrice(rs.getInt("productPrice"));

}

rs.close();

pstmt.close();

conn.close();

}

**catch** (SQLException e)

{

e.printStackTrace();

}

**return** product;

}

// 通过编号查询零件

**public** Part getPartById(**int** id)

{

PreparedStatement pstmt = **null**;

Part part = **null**;

ResultSet rs = **null**;

String sql = "select \* from PartTable where Id = "+id;

**try**

{

conn = ConnectionDao.*getConn*();

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

**while**(rs.next()){

part = **new** Part();

part.setId(rs.getInt("id"));

part.setPartId(rs.getString("partId"));

part.setProductType(rs.getString("productType"));

part.setPartName(rs.getString("partName"));

part.setPartManufacture(rs.getString("partManufacture"));

part.setPartPlace(rs.getString("partPlace"));

part.setPartPrice(rs.getInt("partPrice"));

}

rs.close();

pstmt.close();

conn.close();

}

**catch** (SQLException e)

{

e.printStackTrace();

}

**return** part;

}

//根据条件查询产品信息

**public** List<Product> getProductByWhere(String where)

{

List<Product> list = **new** ArrayList<Product>();

PreparedStatement pstmt = **null**;

ResultSet rs = **null**;

String sql = "select \* from ProductTable " + where;

**try**

{

conn = ConnectionDao.*getConn*();

pstmt = conn.prepareStatement(sql);

Product product = **null**;

rs = pstmt.executeQuery();

**while**(rs.next()){

product = **new** Product();

product.setId(rs.getInt("id"));

product.setProductId(rs.getString("productId"));

product.setProductType(rs.getString("productType"));

product.setProductName(rs.getString("productName"));

product.setProductManufacture(rs.getString("productManufacture"));

product.setProductPlace(rs.getString("productPlace"));

product.setProductPrice(rs.getInt("productPrice"));

list.add(product);

}

rs.close();

pstmt.close();

conn.close();

}

**catch** (SQLException e)

{

e.printStackTrace();

}

**return** list;

}

// 根据条件查询零件信息

**public** List<Part> getPartByWhere(String where)

{

List<Part> list = **new** ArrayList<Part>();

PreparedStatement pstmt = **null**;

Part part = **null**;

ResultSet rs = **null**;

String sql = "select \* from PartTable " + where;

**try**

{

conn = ConnectionDao.*getConn*();

pstmt = conn.prepareStatement(sql);

rs = pstmt.executeQuery();

**while**(rs.next())

{

part = **new** Part();

part.setId(rs.getInt("id"));

part.setPartId(rs.getString("partId"));

part.setProductType(rs.getString("productType"));

part.setPartName(rs.getString("partName"));

part.setPartManufacture(rs.getString("partManufacture"));

part.setPartPlace(rs.getString("partPlace"));

part.setPartPrice(rs.getInt("partPrice"));

list.add(part);

}

rs.close();

pstmt.close();

conn.close();

}

**catch** (SQLException e)

{

e.printStackTrace();

}

**return** list;

}

//BOM表查询，所有产品及零件信息

**public** List<Product> getAllProduct()

{

List<Product> list = **new** ArrayList<Product>();

PreparedStatement pstmt = **null**;

ResultSet rs = **null**;

String sql = "select \* from ProductTable ";

**try**

{

conn = ConnectionDao.*getConn*();

pstmt = conn.prepareStatement(sql);

Product product = **null**;

rs = pstmt.executeQuery();

**while**(rs.next())

{

product = **new** Product();

product.setId(rs.getInt("id"));

product.setProductId(rs.getString("productId"));

product.setProductType(rs.getString("productType"));

product.setProductName(rs.getString("productName"));

product.setProductManufacture(rs.getString("productManufacture"));

product.setProductPlace(rs.getString("productPlace"));

product.setProductPrice(rs.getInt("productPrice"));

list.add(product);

}

rs.close();

pstmt.close();

conn.close();

}

**catch** (SQLException e)

{

e.printStackTrace();

}

**return** list;

}

}

# 五、操作说明

1. 确认开启Mysql。
2. 开启Navicat for MySQ或者MySQL workbench，使 MySQL数据库可视化。
3. 根据JDBC实体层com.entity中属性和字符类型，创建表格。
4. 打开myeclipse10，连接数据库，连接成功后可以看到数据库中已创建的表格。
5. 以上步骤无误完成后，点开Sercers面板，对tomcat7.0添加路径点击绿色开始按钮。
6. 点击开始按钮之后，myeclipse跳转到Console窗口。当出现信息: Server startup in 4525 ms（开启时间），说明已经成功开启。
7. 在浏览器地址框中输入<http://localhost:8080/>（数据库名称），即可进入网页。

# 六、小结

整个网站制作周期投入了大量时间和精力，收获也颇丰。从前期什么都不懂，到自己查阅相关资料到慢慢熟悉java语言以及html标签，这个过程中都付出了艰辛的努力。总的来说可以归结如下：一个网站如何开发，在前期都需要周密规划，规划如何设计，如何布局。然后才根据需要进行代码编写，在编写的过程中，碰到bug是常态，那么就要不停地检查问题出在哪，修改，校验，修改，校验。。。直到解决问题。在这解决bug的过程中尤其能学到东西以及经验积累，当碰到实在不能解决的bug，我就向经验丰富的同学请教，同学也学学别人是如何找到bug，并予以解决。编写完成后，还要不断地完善，反反复复直到达到自己满意的效果，同时，我也体会到了精益求精的工作态度，也学到了开发系统的知识以及培养了良好的自学和钻研的能力。

在此，我要感谢杨远时同学向我推荐的学习java及网页制作的网课，让我快速抓住要点和解决问题，在他的帮助下，我才能在这短时间内完成这个大作业。

由于水平有限，此网站还存在一定的缺陷和不足，希望老师和同学能够多多指教，提出更多宝贵的建议，进一步完善网站，让我在网站开发上更加全面。