

**《数据库设计与开发》**

**课程设计**

**题 目 企业职工管理系统**

姓 名

学 号

院 系

专 业

**二Ｏ一九 年 月 日**

目 录

1. **概述··················································1**
2. **需求分析··············································1**
   1. 系统功能要求设计······················································1
   2. 系统模块设计·························································1
   3. 数据字典·····························································2
3. **总体设计···············································5**
   1. 逻辑和物理结构设计···················································5
      1. 由系统E—R图转化而得到的关系模··································5
      2. 确定关系模型的存取方法···········································6
      3. 确定数据库的存储结构·············································6
4. **数据库结构示意图·······································6**
5. **系统实现···············································7**
   1. 数据库的实施··························································7
6. **功能测试··············································14**
7. **安全性测试············································15**
8. **总结··················································15**

**一 概述**

1.1目的与要求

随着科技的发展，基本上所有的具有一定数量数据的机构都开始使用计算机数据库来做管理。几乎所有企业也都已经在使用计算机管理数据的机制，大大减少了企业管理的工作量。该课程设计要求设计一个企业员工的数据库管理系统，数据库中要求包含员工的基本信息，通过该课程设计，应该达到把数据库理论知识更加的巩固加深，加强动手能力与实践能力，学以致用，与现实生活中的应用充分的结合起来。

1.2设计环境

① MySQL

② eclipse

**二 需求分析**

2.1 系统功能要求设计

此系统实现如下系统功能：

**（**1）使得企业员工管理工作更加清晰、条理化、自动化。

（2）通过用户名和密码登录系统，添加或查询企业员工基本资料功能。 容易地完成员工信息的查询操作。

（3）设计人机友好界面，功能安排合理，操作使用方便，并且进一步考虑系统在安全性，完整性，并发控制，备份和恢复等方面的功能要求。

2.2 系统模块设计

企业人事管理系统大体可以分成二大模块如， 一是企业员工的基本信息模块，里面应该包含员工的各方面的基本信息；再者便是用户登录注册模块， 在该模块中应该包含有对用户注册登录的校验和存储及查询，如下流程图所示：

用户

登录注册

修改员工基本信息

添加员工基本信息

退出

2.3 数据字典

数据项是数据库的关系中不可再分的数据单位，下表分别列出了数据的名称、数据类型、长度、取值能否为空。利用SQLYOG 建立“企业员工基本信息”数据库，其基本表清单及表结构描述如下：

数据库中用到的表：

|  |  |  |
| --- | --- | --- |
| 数据库表名 | 关系模式名称 | 备注 |
| Employee | 员工 | 员工信息表 |
| rpmanage | 奖罚 | 奖罚信息表 |
| Translate | 调度 | 员工调动信息表 |

Employee基本情况数据表，结构如下：

表3.1 员工基本信息表（employee）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 字段名 | 类型 | 长度 | 主键 | 非空 |
| 员工编号 | EmployeeID | varchar | 6 | yes | yes |
| 姓名 | UserName | varchar | 20 | no | yes |
| 性别 | Sex | varchar | 10 | no | yes |
| 部门 | Branch | varchar | 40 | no | yes |
| 出生日期 | Birthday | date | —— | no | yes |
| 籍贯 | NativePlace | varchar | 20 | no | yes |
| 婚姻 | Marriage | varchar | 20 | no | yes |
| 身份证号 | IdentityID | varchar | 18 | no | yes |
| 政汉面貌 | Politics | varchar | 20 | no | yes |
| 民族 | Folk | varchar | 20 | no | yes |
| 学历 | Education | varchar | 20 | no | no |
| 专业 | Department | varchar | 40 | no | no |
| 毕业时间 | GraduateDate | date | —— | no | no |
| 毕业院校 | University | varchar | 40 | no | no |
| 公积金账号 | AccumulateID | varchar | 20 | no | yes |
| 行政级别 | AdministrationLevel | varchar | 20 | no | yes |
| 职务 | Duty | varchar | 40 | no | yes |
| 职称 | Position | varchar | 20 | no | yes |
| 外语语种 | ForeignLanguage | varchar | 20 | no | no |
| 外语等级 | FLLevel | varchar | 20 | no | no |
| 计算机等级 | ComputerLevel | varchar | 20 | no | no |
| 家庭电话 | Phone | varchar | 20 | no | no |
| 手机号码 | MobilePhone | varchar | 11 | no | no |
| 家庭住址 | Address | varchar | 50 | no | no |
| 在职情况 | Incumbency | varchar | 20 | no | yes |
| 用式形式 | IncumbencyType | varchar | 20 | no | yes |
| 个人简历 | Resume | varchar | 400 | no | no |

rpmanage数据表，结构如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 字段名 | 类型 | 长度 | 主键 | 非空 |
|  |  |  |  |  |  |
| 记录编号 | RewardNO | int | —— | yes | yes |
| 员工编号 | EmployeeID | varchar | 6 | no | yes |
| 职称 | Position | varchar | 20 | no | yes |
| 奖/惩 | RewardPunish | varchar | 10 | no | yes |
| 奖惩内容 | Rewardcontent | varchar | 50 | no | yes |
| 原因 | Reason | varchar | 100 | no | yes |
| 批准部门 | Branch | varchar | 40 | no | yes |
| 审核人 | HandleName | varchar | 20 | no | yes |
| 开始时间 | StartDate | date | —— | no | yes |
| 撤销时间 | CancelDate | date | —— | no | no |
| 撤销原因 | CancelReason | varchar | 100 | no | no |
| 备注 | Remark | varchar | 400 | no | no |

translate情况数据表，结构如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 字段名 | 类型 | 长度 | 主键 | 非空 |
| 记录编号 | TranslateNO | int | —— | yes | yes |
| 员工编号 | EmployeeID | varchar | 6 | no | yes |
| 调动时间 | TranslateDate | date | —— | no | yes |
| 前部门 | PriorBranch | varchar | 40 | no | yes |
| 现部门 | NextBranch | varchar | 40 | no | yes |
| 前职务 | PriorDuty | varchar | 40 | no | yes |
| 现职务 | NextDuty | varchar | 40 | no | yes |
| 经办人 | HandleName | varchar | 20 | no | yes |
| 备注 | Remark | varchar | 400 | no | no |

user情况数据表，结构如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 字段名 | 类型 | 长度 | 主键 | 非空 |
| 用户名 | UserName | varchar | 16 | yes | yes |
| 密码 | Password | varchar | 16 | no | yes |

三 总体设计

由需求分析的结果可知，本系统设计的实体包括：

1. 员工基本信息：员工编号，姓名，性别，部门，出生日期，籍贯，婚姻，身份证号，政治面貌，民族，学历，专业，毕业时间，毕业院校，公积金账号，行政级别，职务，职称，外语语种，外语等级，计算机等级，家庭电话，手机号码，家庭住址，在职情况，用工形式，个人简历
2. 用户信息：用户名，密码
3. 员工调动信息：记录编号，员工编号，调动时间，前部门，现部门，前职务，现职务，经办人，备注
4. 员工奖罚信息：记录编号，员工编号，职称，奖/惩，奖惩内容，原因，批准部门，审核人，开始时间，撤销时间，撤销原因，备注

用户

管理

员工基本信息

员工信息

奖惩

调度

3.1逻辑和物理结构设计

3.1.1由系统E—R图转化而得到的关系模式如下：

（1）员工基本信息（员工编号，姓名，性别，部门，出生日期，籍贯，婚姻，身份证号，政治面貌，民族，学历，专业，毕业时间，毕业院校，公积金账号，行政级别，职务，职称，外语语种，外语等级，计算机等级，家庭电话，手机号码，家庭住址，在职情况，用工形式，个人简历），其主关键字为员工编号;

（2）用户信息（用户名，密码），其中主关键字为用户名；

（3）奖惩信息（记录编号，员工编号，职称，奖/惩，奖惩内容，原因，批准部门，审核人，开始时间，撤销时间，撤销原因，备注）。

（4）员工调动信息 （记录编号，员工编号，调动时间，前部门，现部门，前职务，现职务，经办人，备注）

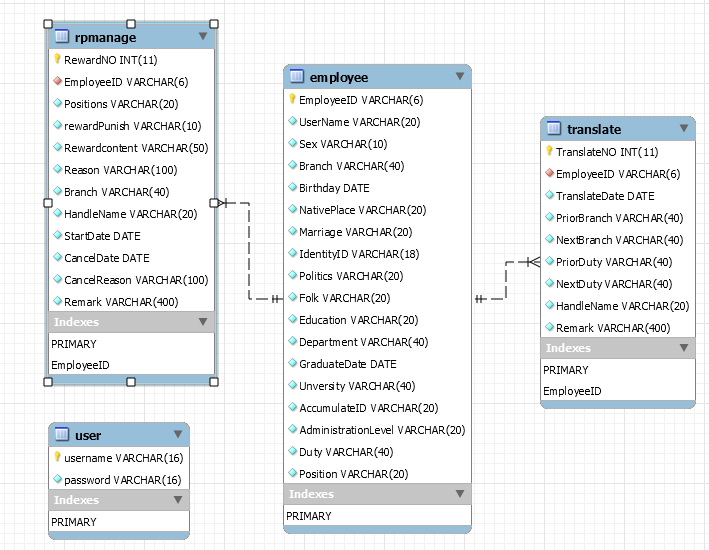
3.1.2 确定关系模型的存取方法

在将概念模型转换成物理模型之后，我们可以对物理模型进行设计，双击物理模型的关系，可以对该关系的名称、注释等信息进行查询。可对该关系的属性列进行设计，可分别设置其名称、码、数据类型以及主码、是否为空等。在实际设计中最常用的存取方法是索引发，使用索引可以大大减少数据的查询时间，在建立索引时应遵循：在经常需要搜索的列上建立索引；  在主关键字上建立索引；在经常用于连接的列上建立索引，即在外键上建立索引；在经常需要根据范围进行搜索的列上创建索引，因为索引已经排序，其指定的范围是连续的等规则。才能充分利用索引的作用避免因索引引起的负面作用。

3.1.3确定数据库的存储结构

确定数据库的存储结构主要指确定数据的存放位置和存储结构，包括确定关系、索引、日志、备份等的存储安排及存储结构，以及确定系统存储参数的配置。因为该成绩管理系统的数据量小，，所以我们只用把数据存储在使用的电脑硬盘上，不用作更多的安排。

**四 数据库结构示意图**



**五 系统实现**

5.1 数据库的实施：

1.c3p0-config.xml



2. 注册类：RegisterServlet.java

package manager.servlet;

import java.io.IOException;

import java.util.HashMap;

import java.util.Map;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import cn.itcast.commons.CommonUtils;

import manager.domian.User;

import manager.service.UserException;

import manager.service.UserService;

/\*\*

\* Servlet implementation class RegisterServlet

\*/

@WebServlet("/RegisterServlet")

public class RegisterServlet extends HttpServlet {

private static final long serialVersionUID = 1L;

/\*\*

\* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

request.setCharacterEncoding("utf-8");

response.setContentType("text/html;charset=utf-8");

UserService userservice = new UserService();

/\*\*

\* 封装表当数据

\*/

User form = CommonUtils.toBean(request.getParameterMap(),User.class);

Map<String,String> errors = new HashMap<String,String>();

//对用户名进行校验

String username = form.getusername();

//String username = request.getParameter("UserName");

if(username==null||username.trim().isEmpty()) {

errors.put("username", "用户名不能为空！");

}else if(username.length()<3||username.length()>15){

errors.put("username", "用户名长度必须在3~15之间!");

}

//对密码进行校验

String password = form.getpassword();

//String password = request.getParameter("Password");

if(password==null||password.trim().isEmpty()) {

errors.put("password", "密码不能为空！");

}else if(password.length()<3||password.length()>15){

errors.put("password", "密码长度必须在3~15之间!");

}

//对验证码进行校验

/\* String sessionVerifyCode = (String) request.getSession().getAttribute("check\_code");

String verifyCode = form.getVerifyCode();

if(verifyCode==null||verifyCode.trim().isEmpty()) {

errors.put("verifyCode", "验证码不能为空！");

}else if(verifyCode.length()!=4){

errors.put("verifyCode", "验证码长度必须为4!");

}else if(!verifyCode.equalsIgnoreCase(sessionVerifyCode)){

errors.put("verifyCode", "验证码错误！");

}\*/

// form.setUserName(username);

// form.setPassword(password);

if(errors!=null&&errors.size()>0) {

/\*\*

\* 1、保存errors到request域中

\* 2、保存form到request域中，为了回显

\* 3、转发到regist.jsp

\*/

request.setAttribute("errors", errors);

request.setAttribute("user", form);

request.getRequestDispatcher("/register.jsp").forward(request, response);

return;

}

try {

userservice.regist(form);

// response.getWriter().print(request.getContextPath()+"\*\*\*\*\*\*\*");

// System.out.println(request.getContextPath()+"\*\*\*\*\*\*\*");

response.getWriter().print("<h1>注册成功！</h1><a href='"+request.getContextPath()+"/login.jsp'"+">点击这里</a>");

} catch (UserException e) {

request.setAttribute("msg", e.getMessage());

request.setAttribute("user", form);

request.getRequestDispatcher("/register.jsp").forward(request, response);

}

}

/\*\*

\* 判断Map是否为空，不为空，说明存在错误

\*/

/\*\*

\* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

doGet(request, response);

}

}

3. 登录类LoginServlet.java

**package** manager.servlet;

**import** java.io.IOException;

**import** javax.servlet.ServletException;

**import** javax.servlet.annotation.WebServlet;

**import** javax.servlet.http.HttpServlet;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

**import** cn.itcast.commons.CommonUtils;

**import** manager.domian.User;

**import** manager.service.UserException;

**import** manager.service.UserService;

/\*\*

\* Servlet implementation class LoginServlet

\*/

@WebServlet("/LoginServlet")

**public** **class** LoginServlet **extends** HttpServlet {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

request.setCharacterEncoding("utf-8");

response.setContentType("text/html;charset=utf-8");

UserService userse = **new** UserService();

User form = CommonUtils.*toBean*(request.getParameterMap(),User.**class**);

//String username = request.getParameter("UserName");

//String password = request.getParameter("Password");

// form.setUserName(username);

// form.setPassword(password);

**try** {

User user = userse.login(form);

request.getSession().setAttribute("sessionUser", user);

response.sendRedirect(request.getContextPath()+"/Welcome.jsp");

} **catch** (UserException e) {

request.setAttribute("msg", e.getMessage());

request.setAttribute("user", form);

request.getRequestDispatcher("/login.jsp").forward(request, response);

}

}

/\*\*

\* **@see** HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

**protected** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

doGet(request, response);

}

}

4.员工基本信息类：ExployeeServlet.java

**package** manager.servlet;

**import** java.io.IOException;

**import** java.lang.reflect.InvocationTargetException;

**import** java.util.HashMap;

**import** java.util.Map;

**import** javax.servlet.ServletException;

**import** javax.servlet.annotation.WebServlet;

**import** javax.servlet.http.HttpServlet;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

**import** org.apache.commons.beanutils.BeanUtils;

**import** cn.itcast.commons.CommonUtils;

**import** manager.domian.Employee;

**import** manager.service.EmployeeService;

**import** manager.service.UserException;

**import** manager.service.UserService;

/\*\*

\* Servlet implementation class AddPersonServlet

\*/

@WebServlet("/AddPersonServlet")

**public** **class** AddPersonServlet **extends** HttpServlet {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

/\*\*

\* **@see** HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

**protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

request.setCharacterEncoding("utf-8");

response.setContentType("text/html;charset=utf-8");

// Employee employee = new Employee();

EmployeeService es = **new** EmployeeService();

Employee employee = CommonUtils.*toBean*(request.getParameterMap(),Employee.**class** );

System.***out***.println(employee.getGraduateDate());

System.***out***.println(employee.getComputerLevel());

System.***out***.println(employee.getFolk());

**try** {

es.addMessage(employee);

System.***out***.println("添加成功");

} **catch** (UserException e) {

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

e.printStackTrace();

}

}

/\*\*

\* **@see** HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

**protected** **void** doPost(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {

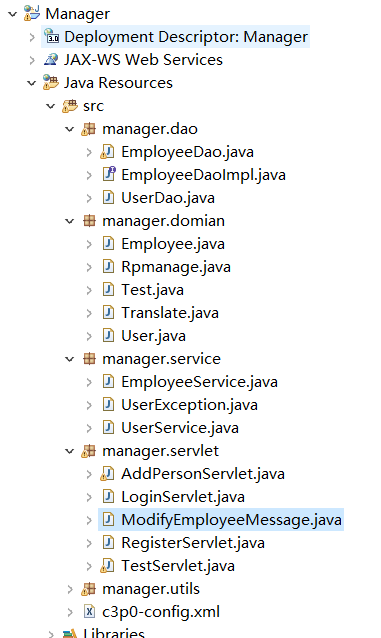
// **TODO** Auto-generated method stub

doGet(request, response);

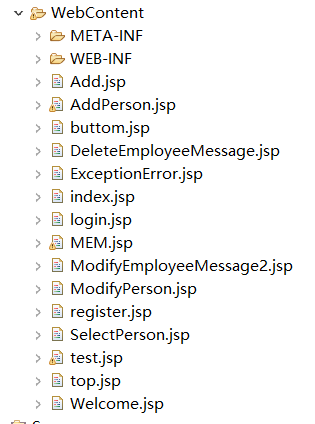
}

}

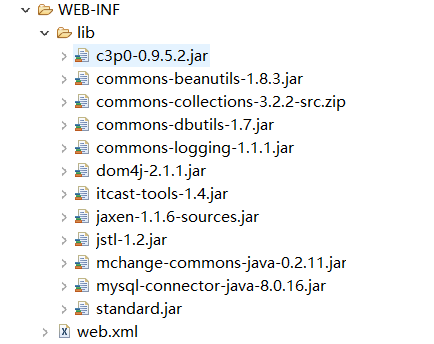
下图显示了创建的包和类



下图显示了JSP页面：

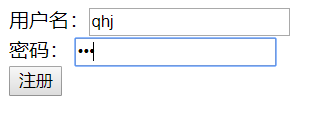


下图显示了所是由的jar包：



**六 功能测试**

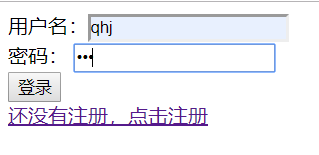
下图显示了用户注册界面：



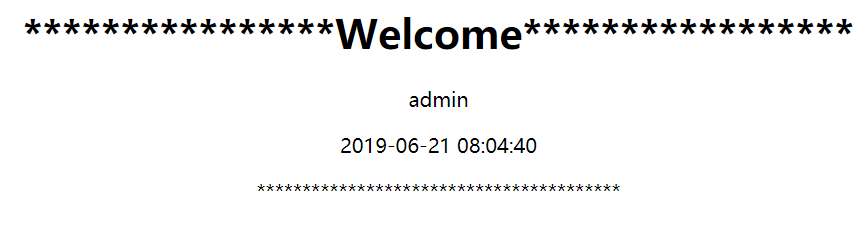
下图显示了用户注册成功页面：



下图显示了用户登录页面：



下图显示了登录成功页面：



下图显示了进行添加员工信息时界面：





**七 安全测试**

数据表中的用户表的密码采取的是明文存储的方案，一旦数据库管理密码泄露，很容易造成用户密码泄露的情况产生。攻击者可以利用这些已知密码去“撞库”来达到获取更大利益的可能。改进方法是将数据库用户表中的密码采取计算hash值存储，这样可以一定程度减小因为数据库管理密码泄露而造成的损失。

**八 总结**

进一步深入理解数据库设计的整个过程，该企业员工基本信息管理的需求分析、数据库概念结构设计有关内容等。掌握使用前台程序设计技术eclipse和MySQL结合设计

MVC结构的管理信息系统。综合练习了MySQL的使用和数据库得管理技术。综合熟悉并练习了SQL语言在编程中的使用。 综合训练编程能力。认识系统，取得系统需求分析，更重要得是在编程方面提高了实现目标代码得能力及调试代码得能力