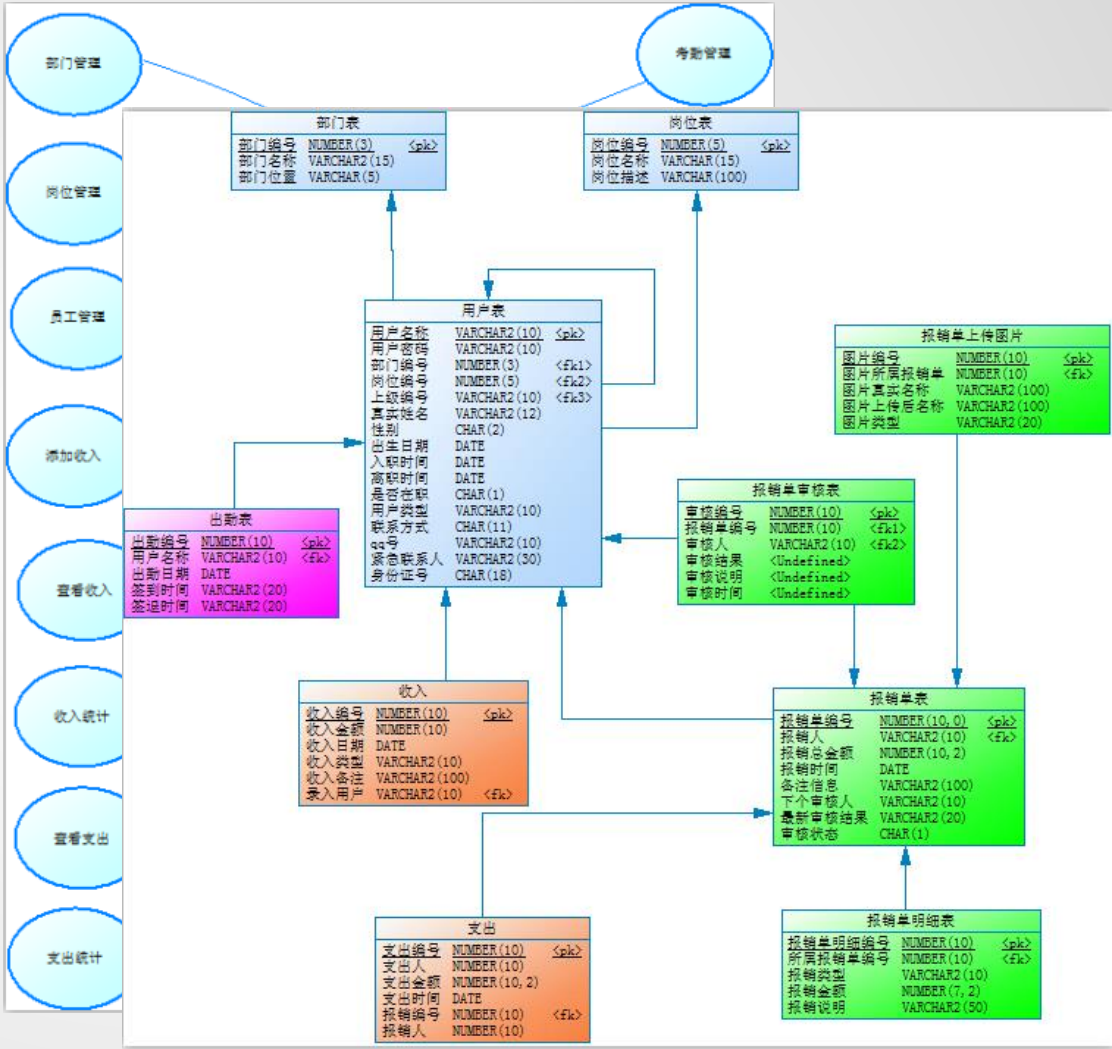
8.贯穿案例：尚学堂OA系统

CONTENTS

练习1——了解项目功能

- 需求说明：
  - 演示项目界面原型
  - 展示用例图并说明
  - 展示数据库模型图并说明
  - 导入数据库表

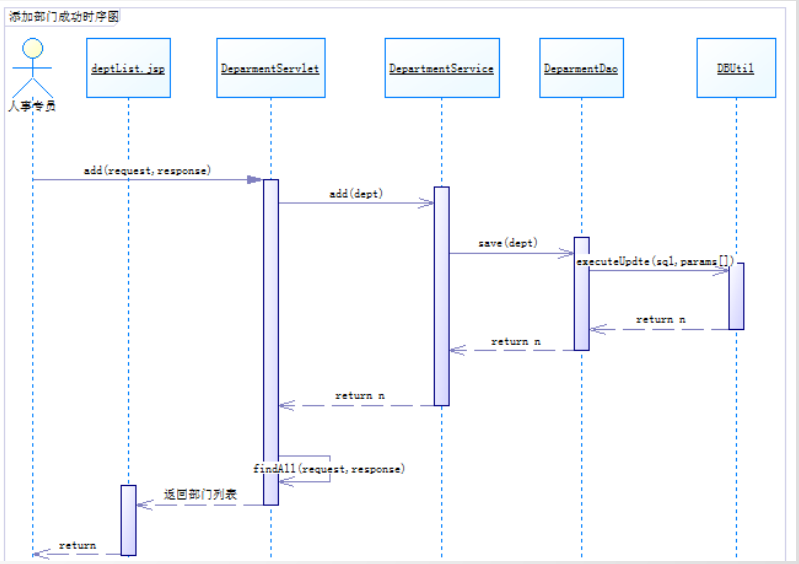


练习2——搭建项目框架

- 需求说明:
  - 创建项目sxtoa（workspace编码统一修改为utf-8）
  - 创建包和文件夹
    - 为测试创建专门的source folder
    - JSP按照模块划分 system 人事 expense 报销 inpay 收支 duty 考勤
  - 加入jar
    - ojdbc6.jar gson-2.2.4.jar junit.jar log4j.jar
  - 加入工具类
    - DBUtil.java BaseServlet
  - 加入过滤器（暂不配置）
  - 加入界面原型 （采用界面原型的jquery版本）
  - 部署项目并测试

练习3——添加部门

- 需求说明：
  - 实体类 Department
  - 数据库访问层 Department Dao Department DaoImpl
  - 业务层： Department Service Department ServiceImpl
  - Junit测试：
  - 控制层： DepartmentServlet extends BaseServlet
  - 视图层：system/deptAdd.jsp
    - 将HTML修改为JSP
- 新技能点：
  - Junit测试：使用JUnit4；更加专业、规范；使用注解实现
  - 将HTML修改为JSP（修改方法、路径的处理）



练习4——查询部门

- 需求说明：
  - 开发数据访问层，实现查询所有部门操作
  - 开发业务层，实现查询所有部门操作
  - 使用Junit测试后台查询所有部门操作
  - 开发控制层，实现查询所有部门操作
  - 开发视图层，实现显示所有部门操作

部门列表

	编号	部门名称	办公地点	操作
☐	1	总裁办	501	修改 删除
☐	2	教学部	302	修改 删除
☐	3	咨询部	204	修改 删除
☐	4	教务部	303	修改 删除

练习5——删除部门

- 需求说明：
  - 开发视图层，实现删除部门超链接
  - 开发控制层，实现删除部门功能
  - 开发业务层，实现删除部门功能
  - 开发数据访问层，实现删除部门功能

练习6——更新部门

- 需求说明：
  - 使用MVC模式查询并显示指定部门信息
  - 使用MVC模式更新指定部门信息

总结： MVC架构

- MVC架构
  - MVC它主要分模型、视图、控制器三层。
  - 实现了显示模块与功能模块的分离
- MVC架构的好处
  - 降低耦合性
  - 分工协作
  - 组件重用
- MVC架构的缺点
  - 增加了系统结构和实现的复杂性

<table>
<tr><td>视图层View
JSP</td><td>视图层V
JSP + Ajax
+JavaScript</td></tr>
<tr><td>控制层Control
Servlet</td><td>控制层
Servlet</td></tr>
<tr><td rowspan="2">模型层Model
JavaBean</td><td>业务层
JavaBean</td></tr>
<tr><td>数据访问层
JDBC+DAO</td></tr>
</table>

总结： PowerDesigner面向对象建模

- PowerDesigner面向对象建模
  - 用例图：(User Case Diagram) 通常用来定义系统的高层次草图，它从用户角度描述了应用的系统功能，指出了各个功能的外部操作者。包括用例、执行者、关联三个建模要素。
  - 序列图：(Sequence Diagram):描述系统如何实际完成在User Case图中定义的功能。可以画出对象(类的实例)之间交互时产生的时序关系。主要包括执行者、对象、生命线、激活、消息、返回消息等建模要素
  - 类图(Class Diagram):描述类与类之间的静态关系。其中包括一系列的包，类，接口和它们之间的关系。类图是定义其他图的基础。

作业

- 完成岗位管理功能
  - 添加岗位
  - 查询岗位
  - 删除岗位
  - 更新岗位
  - 画出四个操作的时序图