CRM项目

1. 软件开发的生命周期：

1)招标：
投标:----------标书

甲方：
乙方：

2)可行性分析:---------可行性分析报告
技术,经济

3)需求分析:----------需求文档
产品经理,需求调研。设计项目原型，项目原型是没有实现实际功能前端页面，只能看看的。
项目原型:容易确定需求,开发项目时作为jsp网页.

4)分析与设计：

架构设计：----------架构文档

物理架构设计：
应用服务器:tomcat(apache),weblogic(bea-->oracle),websphere(ibm),jboss(redhat),resin(MS)
web javaee:13种协议
servlet,jsp,xml,jdbc
mq ....
数据库服务器：mysql,oracle,DB2,sqlserver,达梦

逻辑架构设计：代码分层.
视图层-->控制层-->业务层-->持久层-->数据库
技术选型：java,.net

项目设计：---------项目设计文档
物理模型设计：哪些表，哪些字段，字段的类型和长度，以及表和表之间的关系。
powerdesigner-----xxxx.pdm
逻辑模型设计：哪些类，哪些属性和方法，方法的参数和返回值，以及类和类之间关系。
rational rose-----.pdl
界面设计：企业级应用 朴素 -----项目原型
互联网应用 炫酷
算法设计：------算法设计文档

5)搭建开发环境：----------技术架构文档
创建项目,添加jar包,添加配置文件,添加静态页面,添加公共类以及其它资源;能够正常启动运行。

6)编码实现：-------注释
7)测试：----------测试用例
8)试运行：---------使用手册
9)上线：----------实施文档
10)运维：----------运维手册

11)文档编纂：

2. CRM项目的核心业务：

1)CRM项目的简介：Customer Relationship Management 客户关系管理系统
企业级应用,传统应用;给销售或者贸易型公司使用,在市场,销售,服务等各个环节中维护客户关系，
CRM项目的宗旨：增加新客户,留住老客户，把已有客户转化为忠诚客户。

2)CRM是一类项目,我们的CRM是给一个大型的进出口贸易公司来使用的，做大宗商品的进出口贸易;商品是受管家管制的。

3)CRM项目的核心业务：

系统管理功能：不是直接处理业务数据，为了保证业务管理的功能正常安全运行而设计的功能。
用户登录,安全退出,登录验证等。跟用户相关的功能是属于系统管理功能。
给超级管理员，开发和运维人员使用。

业务管理功能：处理业务数据
市场活动：市场部，设计市场活动营销活动
线索：销售部(初级销售),增加线索
客户和联系人：销售部(高级销售),有效地区分和跟踪客户和联系人.
交易：销售部(高级销售),更好地区分和统计交易的各个阶段。
售后回访：客服部,妥善安排售后回访。主动提醒。
统计图表：管理层,统计交易表中各个阶段数据量。

所有的项目，无论是企业级应用还是互联网应用，项目的业务都能分成两大块：系统管理功能和业务管理功能。

3.CRM物理模型设计和搭建开发环境：

1.crm的表结构：

```
tbl_user    用户表（用于存放已经注册的用户信息）

tbl_dic_type    数据字典类型表（用于存放项目中所有下拉列表的类型，根据下拉列表的类型确定这个下拉列表中有哪些值）
tbl_dic_value   数据字典值（用于存放项目中所有下拉列表中的数据）
数据字典类型表和数据字典值表是一对多的关系

tbl_activity    市场活动表
tbl_activity_remark   市场活动备注表

tbl_clue        线索表
tbl_clue_remark    线索备注表

tbl_clue_activity_relation   线索和市场活动的关联关系表

tbl_customer    客户表
tbl_customer_remark   客户备注表

tbl_contacts    联系人表
tbl_contacts_remark 联系人备注表

tbl_contacts_activity_relation 联系人和市场活动的关联关系表

tbl_tran        交易表
```

```
tbl_tran_remark   交易备注表
tbl_tran_history   交易历史表

tbl_task    任务表


注：给数据库表起名的规范不是驼峰命名，而是全部小写字母，不同单词之间用_隔开
```

数据字典值表和数据字典类型表是任何的项目中都会有的。

2.创建crm的数据库实例：
把sql脚本导入数据库实例：

3.搭建开发环境：
1)创建项目：crm-project
设置JDK.
创建工程：crm
补全目录结构：
2)添加jar包：添加依赖---参考课件.

物理模型设计思路：

1)主键字段：

在数据库表中，如果有一组字段能够唯一确定一条记录，则可以把它们设计成表的主键字段。
推荐使用一个字段做主键，而且推荐使用没有业务含义的字段做主键,比如：id等。

主键字段的类型和长度由主键值的生成方式来决定。
主键值的生成方式：
1)自增：借助数据库自身主键生成机制 （不推荐）
数值型 长度由数据量来决定

运行效率低
开发效率高

2)assighed(签名)：程序员手动生成主键值,唯一非空,算法.
hi/low算法：数值型 长度由数据量决定
UUID算法：字符串 长度是32位 （最常用）

3)共享主键：由另一张表的类型和长度决定 （不推荐）

tbl_person		tbl_card	
id	name	id	name
1001	zs	1001	card1
1002	ls		

4)联合主键：由多个字段的类型和长度决定 （不推荐）

2)外键字段：

用来确定表和表之间的关系。

1.一对多：

一张表(A)中的一条记录可以对应另一张表(B)中的多条记录;
另一张表(B)中的一条记录只能对应一张表(A)中的一条记录。
一方也叫父表，多方也叫子表

一对多关系表的设计：在多方加外键。

```
tbl_student                    tbl_class
id      name class_id          id     name
1001    zs    111              111    class1
1002    ls    111              222    class2
1003    ww    222
1004    zl
```

添加数据时,先添加父表记录，再添加子表记录;
删除数据时,先删除子表记录，再删除父表记录;
查询数据时,可能会进行关联查询：

```
//查询所有姓张的学生的id,name和所在班级name
select s.id,s.name,c.name as className
from tbl_student s
join tbl_class c on s.class_id=c.id//假如外键不可以为空
where s.name like 'z%'
```

内连接：查询所有符合条件的数据，并且要求结果在两张表中都有相对应的记录
左外连接：查询左侧表中所有符合条件的数据，即使在右侧表中没有相对应的记录

*如果外键不能为空（即设置了非空约束），优先使用内连接;
如果外键可以为空，

```
--假如只需要查询那些在另一张表中有相对应的记录，使用内连接
--假如需要查询左侧表中所有符合条件的记录，使用左外连接.
```

2. 一对一：

一张表(A)中的一条记录只能对应另一张表(B)中的一条记录;
另一张表(B)中的一条记录也只能对应一张表(A)中的一条记录。

```
tbl_person         tbl_card
id     name        id     name
1001   zs          1001    card1
```

一对一关系表的设计：唯一外键（即外键设唯一性约束），是任选其中一张表加外键。

```
tbl_person             tbl_card
id     name            id     name     person_id(唯一性约束)
1001   zs              111    card1    1001
1002   ls              222    card2    1002
1003   ww              333    card3    1003
```

*一对一就是一种特殊的一对多。
*操作跟一对多完全一样。

3. 多对多：

一张表(A)中的一条记录可以对应另一张表(B)中的多条记录;
另一张表(B)中的一条记录也可以对应一张表(A)中的多条记录。

```
tbl_student                    tbl_course
 id     name                   id     name
 1001   zs                     111    java
 1002   ls                     222    mysql


 tbl_student_course_relation (关系表)
 student_id     course_id
 1001            111
 1001            222
 1002            111
 1002            222
```

添加数据时，先添加父表记录(tbl_student,tbl_course),再添加子表(tbl_student_course_relation)记录;

删除数据时，先删除子表记录(tbl_student_course_relation),再删除父表记录(tbl_student,tbl_course)

查询数据时，可能会进行关联查询：

```
//查询所有姓张的学生的id,name,和所选课程的name
select s.id,s.name,c.name as courseName
from tbl_student s
join tbl_student_course_relation scr on s.id=scr.student_id
join tbl_course c on scr.course_id=c.id
 where s.name like 'z%'
```

由于关系表中的外键表示的是关联关系，所以都不能为NULL（即一定设置了非空约束），所以多对多的连接查询，用内连接。

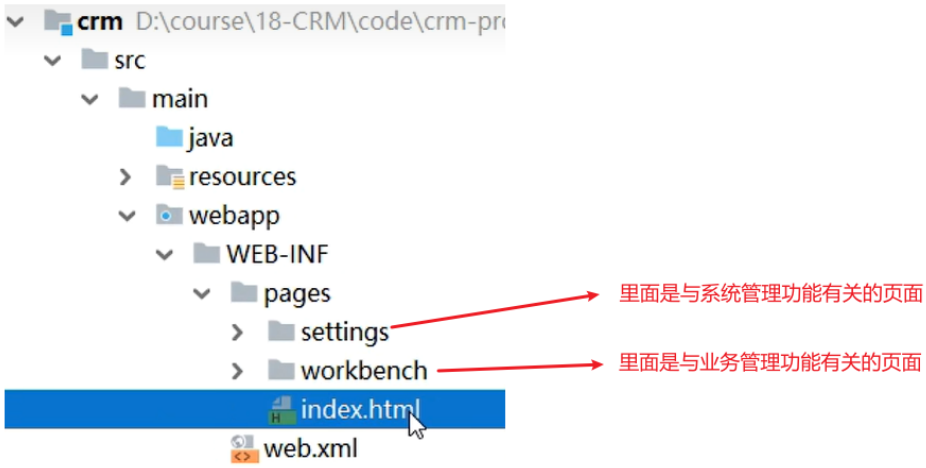
3)关于日期和时间的字段：

都按照字符串处理：也就是在mysql中存日期和时间的时候，用字符串来存储日期和时间，而不要用mysql中的date/time/datetime来存储日期和时间。并且是用定长字符串来存。

char(10) yyyy-MM-dd
char(19) yyyy-MM-dd HH:mm:ss

思考：开发的时候是先实现业务管理功能还是系统管理功能?

答：系统管理功能，因为系统管理功能是为了维护业务管理功能正常安全运行而设计的功能，业务管理功能里面会用到系统管理功能所产生的数据，所以要先实现系统管理功能。



4. 项目首页

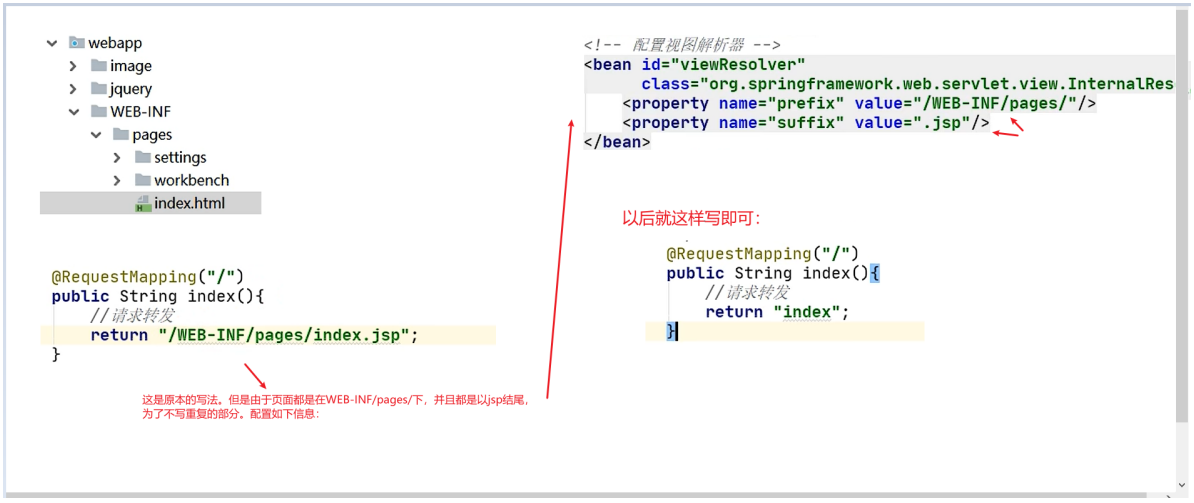
要明确一点：url中的项目名后的任何东西都是一个请求路径。

```
http://127.0.0.1/crm/

crm后面有一个/,这是一个请求路径，一般将/设为项目首页的请求路径
```

将原型中的index.html转成jsp格式。

```
由于idea中的html默认编码格式为iso8859-1，而默认的jsp的编码格式为utf-8，如果直接重命名为index.jsp，则遇到中文时就会出现乱码。一定要先设置index.html的编码格式为utf-8 (即在最上方加上：<% @Page content-Type="text/html;charset=UTF-8" language="java" %>)，再重命名为index.jsp。
```



5. 用户登录：

5.1 同步请求和异步请求的区别：

同步请求：浏览器窗口发出的请求,响应信息返回到浏览器窗口,所以会进行全局刷新。
异步请求：ajax发出的请求,响应信息返回到ajax的回调函数,既可以进行全局刷新，也可以进行局部刷新。

小结：如果需要进行全局刷新，推荐使用同步请求，当然也可以使用异步请求；
如果需要进行局部刷新，只能使用异步请求；
如果既可能进行全局刷新，也可能进行局部刷新，也是只能使用异步请求。

5.2. mybatis逆向工程：

1)简介：根据表生成mapper层三部分代码：实体类，mapper接口，映射文件。
2)使用mybatis逆向工程：
a)为了不污染已有的项目中的代码，单独创建一个工程来使用mybatis逆向工程。创建工程:crm-mybatis-generator
b)添加插件：

```
    <!--myBatis逆向工程插件-->
    <plugin>
    <groupId>org.mybatis.generator</groupId>
    <artifactId>mybatis-generator-maven-plugin</artifactId>
    <version>1.3.2</version>
    <configuration>
        <verbose>true</verbose>
        <overwrite>true</overwrite>
    </configuration>
    </plugin>
```

c)添加配置文件(共2个)：
数据库连接信息
指定代码保存在什么目录
表的信息。注意：每次只能指定一张表，不然会覆盖已有的内容。

generator.properties :

```
<!-- generator.properties -->

jdbc.driverLocation=D:/testDir/Maven/repository_g/mysql/mysql-connector-
java/5.1.43/mysql-connector-java-5.1.43.jar
jdbc.driverClass=com.mysql.jdbc.Driver
jdbc.connectionURL=jdbc:mysql://127.0.0.1:3306/mycrm_db
jdbc.userId=root
jdbc.password=yf123
```

generatorConfig.xml :

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE generatorConfiguration
        PUBLIC "-//mybatis.org//DTD MyBatis Generator Configuration 1.0//EN"
        "http://mybatis.org/dtd/mybatis-generator-config_1_0.dtd">

<generatorConfiguration>
```

```
    <!--指定mysql数据库驱动-->
    <!--<classPathEntry location="E://repository-p2p//mysql//mysql-connector-
java//5.1.43//mysql-connector-java-5.1.43.jar"/>-->

    <!--导入属性配置-->
    <properties resource="generator.properties"></properties>

    <!--指定特定数据库的jdbc驱动jar包的位置-->
    <classPathEntry location="${jdbc.driverLocation}"/>

    <context id="default" targetRuntime="MyBatis3">

        <!-- optional，旨在创建class时，对注释进行控制，false生成注释,true无注释 -->
        <commentGenerator>
            <property name="suppressDate" value="false"/>
            <property name="suppressAllComments" value="false"/>
        </commentGenerator>

        <!--jdbc的数据库连接 -->
        <jdbcConnection
                driverClass="${jdbc.driverClass}"
                connectionURL="${jdbc.connectionURL}"
                userId="${jdbc.userId}"
                password="${jdbc.password}">
        </jdbcConnection>


        <!-- 非必需，类型处理器，在数据库类型和java类型之间的转换控制-->
        <javaTypeResolver>
            <property name="forceBigDecimals" value="false"/>
        </javaTypeResolver>


        <!-- Model模型生成器（Model模型就是实体类）,用来生成含有主键key的类，记录类 以及查
询Example类
            targetPackage     指定生成的model生成时所在的包名
            targetProject     指定在该项目下所在的路径|指定生成到的工程名称
        -->
        <javaModelGenerator targetPackage="com.bjpowernode.crm.settings.domain"
                            targetProject="D:/course/18-CRM/code/crm-
project/crm/src/main/java">

            <!-- 是否允许子包，即targetPackage.schemaName.tableName -->
            <property name="enableSubPackages" value="false"/>
            <!-- 是否对model添加 构造函数 true添加，false不添加-->
            <property name="constructorBased" value="false"/>
            <!-- 是否对类CHAR类型的列的数据进行trim操作 -->
            <property name="trimStrings" value="true"/>
            <!-- 建立的Model对象是否 不可改变  即生成的Model对象不会有 setter方法，只有构
造方法 -->
            <property name="immutable" value="false"/>
        </javaModelGenerator>

        <!--指定Mapper.xml映射文件生成后放在哪个目录 为每一个数据库的表生成对应的SqlMap文件
-->
        <sqlMapGenerator targetPackage="com.bjpowernode.crm.settings.mapper"
```

```
                        targetProject="D:/course/18-CRM/code/crm-
project/crm/src/main/java">
            <property name="enableSubPackages" value="false"/>
        </sqlMapGenerator>

        <!-- 客户端代码，生成易于使用的针对Model对象和XML配置文件 的代码
                type="ANNOTATEDMAPPER",生成Java Model 和基于注解的Mapper对象
                type="MIXEDMAPPER",生成基于注解的Java Model 和相应的Mapper对象
                type="XMLMAPPER",生成SQLMap XML文件和独立的Mapper接口
        -->
        <!--指定Mapper接口生成后放在哪个目录-->
       <javaClientGenerator targetPackage="com.bjpowernode.crm.settings.mapper"
                            targetProject="D:/course/18-CRM/code/crm-
project/crm/src/main/java" type="XMLMAPPER">
            <property name="enableSubPackages" value="true"/>
        </javaClientGenerator>

        <!-- 每次只能指定一张表(别的表需要注释掉)-->
        <table tableName="tbl_user" domainObjectName="User"
               enableCountByExample="false" enableUpdateByExample="false"
               enableDeleteByExample="false" enableSelectByExample="false"
               selectByExampleQueryId="false">
        </table>

        <!--
        <table tableName="tbl_transaction" domainObjectName="Transaction"
               enableCountByExample="false" enableUpdateByExample="false"
               enableDeleteByExample="false" enableSelectByExample="false"
               selectByExampleQueryId="false">
        </table>
        <table tableName="tbl_transaction_history"
domainObjectName="TransactionHistory"
               enableCountByExample="false" enableUpdateByExample="false"
               enableDeleteByExample="false" enableSelectByExample="false"
               selectByExampleQueryId="false">
        </table>
        <table tableName="tbl_transaction_remark"
domainObjectName="TransactionRemark"
               enableCountByExample="false" enableUpdateByExample="false"
               enableDeleteByExample="false" enableSelectByExample="false"
               selectByExampleQueryId="false">
        </table>
        -->

    </context>
</generatorConfiguration>
```

d)运行mybatis的逆向工程，根据指定表生成java代码，保存到指定的目录中。

5.3 使用jquery获取指定元素的指定属性的值：

选择器.attr("属性名");//用来获取那些值不是true/false的属性的值.
选择器.prop("属性名");//用来获取值是true/false的属性的值.例如：checked,selected,readonly,disabled等。