第 1 章　CRM 核心业务介绍

1，CRM(Customer Relationship Management)客户关系管理是管理**企业与客户之间关系**的新型管理机制。终极目标是吸引新客户、保留老客户以及将已有客户转变为忠诚客户，以增加市场份额。

它是一个完整的客户关系管理系统，包括市场、销售、服务 3 大环节，产品成熟，操作简单，功能强大。帮您从客户全生命周期的各个阶段获取价值。

2，这是一类软件，很多公司专门做这种 CRM 系统。我们这个系统，是从咱们一个非常有名的专门致力于 CRM 项目开发的公司中拿过来的。

不同行业的 CRM，功能一点都不一样，它是由销售模式来决定的，你比如有分销和直销。

3，CRM 为改善企业与客户之间关系的提供了一种新型管理机制，企业可以规划市场营销活动、增加销售线索、规范客户联系人信息、对不同阶段的交易进行有效区分和统计、妥善安排售后回访、为决策提供支撑等，优化各业务环节，减少各环节客户流失，和公司成本。当然，我们不是全部都做，我们只是拿出来其中的某几个模块来做。

4，业务流程：

市场活动:市场部人员。

线索：销售部人员。(线索购买意愿非常强烈的时候，可以转换)

客户：线索中的公司信息

联系人：线索中的联系人信息

交易：已经促成的交易。促成的交易不一定就肯定成交，只有那些已经有交易意向的客户才创建交易记录。交易中有一些阶段，总共九个阶段，越是往下越是成交的可能性越大，当然也可能最后成交失败。

售后回访：客服人员。成交之后的。

统计报表：销售漏斗。主要体现交易数据各个阶段的数据变化。

第 2 章　搭建 CRM 项目开发环境

2.1 CRM 项目数据库设计

2.1.1　数据库设计原则

我们可以结合项目原型来考虑数据库设计，市场活动，用户，这些都是需要是持久化的，所以都需要设计成表。表和表之间通常还会有一定的关系。看每一个创建表单上都有哪些属性。

1、所有的表来自于需求：

名词，特别是业务相关的名词。理清名词之间的关系。

概念性的名词设计成表、说明性的名词设计成字段。

有些不明显的名词，需要抽象，如数据字典表和数据字典类型表

2、字段：

1>主键:表中有很多字段，如果其中有一组字段能够唯一标识一条记录，则可以把这一

组字段设计成主键。一般来讲，都是一个字段做主键，而且是没有业务语义的字段做主键。当然，也有个别情况下使用业务相关的字段做主键。

主键的生成方式：决定主键字段设计成什么类型和主键字段的长度。

a、自增：采用数据库提供的主键生成机制。

Mysql:auto_increment

Oracle:sequence

> 由于常用的数据库，如 Oracle、DB2、SQLServer、MySql 等，都提供了易用的主键生成机制（Auto-Increase 字段或者 Sequence）。我们可以在数据库提供的主键生成机制上，采用 generator-class=native 的主键生成方式。
>
> 不过值得注意的是，一些数据库提供的主键生成机制在效率上未必最佳，大量并发 insert 数据时可能会引起表之间的互锁。
>
> 数据库提供的主键生成机制，往往是通过在一个内部表中保存当前主键状态（如对于自增型主键而言，此内部表中就维护着当前的最大值和递增量），之后每次插入数据会读取这个最大值，然后加上递增量作为新记录的主键，之后再把这个新的最大值更新回内部表中，这样，一次 Insert 操作可能导致数据库内部多次表读写操作，同时伴随的还有数据的加锁解锁操作，这对性能产生了较大影响。因此，对于并发 Insert 要求较高的系统，推荐采用 uuid.hex 作为主键生成机制。

b、assigned：外部程序员负责生成。

UUID,

hi/lo,

Twitter-Snowflake 算法产生的背景相当简单，为了满足 Twitter 每秒上万条消息的请求，每条消息都必须分配一条唯一的 id，这些 id 还需要一些大致的顺序（方便客户端排序），并且在分布式系统中不同机器产生的 id 必须不同。

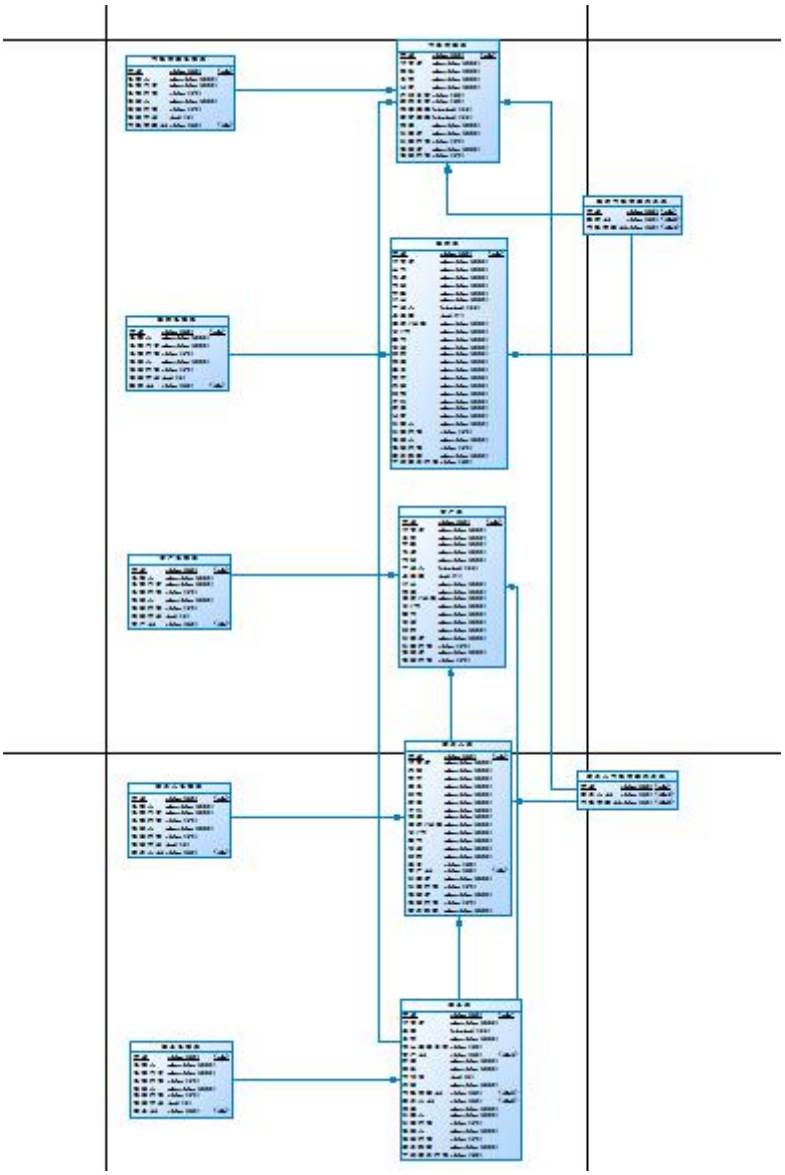
c、共享主键：

d、联合主键：

2>、外键：外键用于表示表和表之间的关系。有的运行时不加外键，就是通过程序来保证数据的完整性。

3>、日期型数据的使用 char 型

2.1.2 数据库结构设计



2.1.3 初始化数据库

导入 init-crm-data-mysql.sql 脚本文件。

2.2 搭建开发环境

2.2.1 创建 crm 项目

创建 Empty Project，作为项目的工作空间

2.2.2 创建 crm 模块

创建 maven 类型的模块，作为开发工程

2.2.3 为项目添加 maven 依赖

（1） mysql 驱动

```
<!-- MySQL 数据库连接驱动 -->
<dependency>
<groupId>mysql</groupId>
<artifactId>mysql-connector-java</artifactId>
<version>5.1.43</version>
</dependency>
```

（2） JDBC 数据源连接池：Druid

```
<!-- JDBC 数据源连接池 -->
<dependency>
<groupId>com.alibaba</groupId>
<artifactId>druid</artifactId>
<version>1.1.1</version>
</dependency>
```

还有:c3p0 连接池、dbcp 连接池等。

（3） Mybatis 框架依赖

```
<!-- MyBatis 框架依赖 -->
<dependency>
<groupId>org.mybatis</groupId>
<artifactId>mybatis</artifactId>
<version>3.4.1</version>
</dependency>
```

（4） Spring 相关依赖配置

```
<!-- Spring 框架依赖的 JAR 配置 -->
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-context</artifactId>
<version>4.3.9.RELEASE</version>
</dependency>
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-aop</artifactId>
<version>4.3.9.RELEASE</version>
</dependency>
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-core</artifactId>
<version>4.3.9.RELEASE</version>
</dependency>
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-beans</artifactId>
<version>4.3.9.RELEASE</version>
</dependency>
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-jdbc</artifactId>
<version>4.3.9.RELEASE</version>
</dependency>
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-tx</artifactId>
<version>4.3.9.RELEASE</version>
</dependency>
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-web</artifactId>
<version>4.3.9.RELEASE</version>
</dependency>
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-webmvc</artifactId>
<version>4.3.9.RELEASE</version>
</dependency>
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-oxm</artifactId>
<version>4.3.9.RELEASE</version>
</dependency>
```

（5） Spring AOP 依赖

```
<!-- Spring AOP 支持-->
<dependency>
<groupId>org.aspectj</groupId>
<artifactId>aspectjweaver</artifactId>
<version>1.8.9</version>
</dependency>
```

（6） Mybatis 与 Spring 整合依赖

```
<!-- MyBatis 与 Spring 整合依赖 -->
<dependency>
<groupId>org.mybatis</groupId>
<artifactId>mybatis-spring</artifactId>
<version>1.3.0</version>
</dependency>
```

（7） 添加项目对 JSP 的支持

```
<!-- servlet 及 jstl 标签库依赖的 JAR 配置 -->
<dependency>
   <groupId>javax.servlet</groupId>
   <artifactId>javax.servlet-api</artifactId>
   <version>3.1.0</version>
</dependency>
<dependency>
   <groupId>javax.servlet.jsp.jstl</groupId>
   <artifactId>jstl-api</artifactId>
   <version>1.2</version>
</dependency>
<dependency>
   <groupId>org.apache.taglibs</groupId>
   <artifactId>taglibs-standard-spec</artifactId>
   <version>1.2.1</version>
</dependency>
<dependency>
   <groupId>org.apache.taglibs</groupId>
   <artifactId>taglibs-standard-impl</artifactId>
   <version>1.2.1</version>
</dependency>
```

加载 jackson 插件依赖

（8） Jackson 插件依赖

```
<!-- 加载 jackson 插件依赖 -->
<dependency>
   <groupId>com.fasterxml.jackson.core</groupId>
   <artifactId>jackson-core</artifactId>
   <version>2.7.3</version>
</dependency>
<dependency>
   <groupId>com.fasterxml.jackson.core</groupId>
   <artifactId>jackson-databind</artifactId>
   <version>2.7.3</version>
</dependency>
<dependency>
   <groupId>com.fasterxml.jackson.core</groupId>
   <artifactId>jackson-annotations</artifactId>
   <version>2.7.3</version>
</dependency>
```

（9） poi 依赖

```
<!--poi 依赖-->
<dependency>
   <groupId>org.apache.poi</groupId>
   <artifactId>poi</artifactId>
   <version>3.15</version>
</dependency>
```

（10） fileupload 依赖

```
<!-- 文件上传 -->
<dependency>
   <groupId>commons-fileupload</groupId>
   <artifactId>commons-fileupload</artifactId>
   <version>1.3.1</version>
</dependency>
```

（11） Log4j 依赖

```
<!-- Log4j2 依赖的 JAR 配置 -->
<dependency>
<groupId>org.apache.logging.log4j</groupId>
<artifactId>log4j-api</artifactId>
<version>2.3</version>
</dependency>
<dependency>
<groupId>org.apache.logging.log4j</groupId>
<artifactId>log4j-core</artifactId>
<version>2.3</version>
</dependency>
```

2.2.4 添加相关配置

（1） MyBatis 配置

mybatis-config.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration
        PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
        "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
<settings>
    <setting name="logImpl" value="STDOUT_LOGGING"/>
</settings>
    <typeAliases>
        <package name="com.bjpowernode.crm.model"/>
    </typeAliases>
    <mappers>
        <package name="com.bjpowernode.crm.mapper"/>
    </mappers>
</configuration>
```

（2） 配置数据连接和事务

applicationContext-datasource.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:aop="http://www.springframework.org/schema/aop"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:tx="http://www.springframework.org/schema/tx"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
        http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.3.xsd
        http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-4.3.xsd
        http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-4.3.xsd">
    <!-- 配置数据源 -->
     <bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <property name="driverClassName" value="com.mysql.jdbc.Driver"/>
        <property name="username" value="root"/>
        <property name="password" value="123456"/>
        <property name="url"
value="jdbc:mysql://192.168.223.133:3306/crm_db?useSSL=false&amp;useUnicode=true&amp;characterEncoding=UTF-8"/>
    </bean>
    <!-- 配置 SqlSessionFactory -->
    <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
        <!-- 必须注入属性 dataSource -->
        <property name="dataSource" ref="dataSource"/>
        <!-- 如果 mybatis 没有特殊的配置(比如别名等)，configLocation 可以省去 ;否则，不能省略-->
        <property name="configLocation" value="classpath:mybatis-config.xml"/>
    </bean>
    <!-- mapper 注解扫描器配置,扫描@MapperScan 注解,自动生成代码对象 -->
     <bean id="mapperScanner" class="org.mybatis.spring.mapper.MapperScannerConfigurer">
           <property name="basePackage" value="com.bjpowernode.crm.mapper"/>
           <property name="sqlSessionFactoryBeanName" value="sqlSessionFactory"/>
     </bean>

    <!-- 配置事务管理器 -->
    <bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
        <property name="dataSource" ref="dataSource"/>
    </bean>
    <!-- 配置事务 -->
    <aop:config>
        <aop:pointcut expression="execution(* com.bjpowernode.crm..service.*.*(..))" id="allMethodPointcut"/>
        <aop:advisor advice-ref="txAdvice" pointcut-ref="allMethodPointcut"/>
    </aop:config>
    <tx:advice id="txAdvice" transaction-manager="transactionManager">
        <tx:attributes>
            <tx:method name="add*" propagation="REQUIRED" rollback-for="Exception"/>
            <tx:method name="save*" propagation="REQUIRED" rollback-for="Exception"/>
            <tx:method name="edit*" propagation="REQUIRED" rollback-for="Exception"/>
            <tx:method name="update*" propagation="REQUIRED" rollback-for="Exception"/>
            <tx:method name="delete*" propagation="REQUIRED" rollback-for="Exception"/>
            <tx:method name="do*" propagation="REQUIRED" rollback-for="Exception"/>
            <tx:method name="*" propagation="REQUIRED" read-only="true"/>
        </tx:attributes>
    </tx:advice>
</beans>
```

（3） springmvc 配置

applicationContext-mvc.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"
xmlns:p="http://www.springframework.org/schema/p"
xmlns:util="http://www.springframework.org/schema/util"
xmlns:aop="http://www.springframework.org/schema/aop"
xmlns:tx="http://www.springframework.org/schema/tx"
xmlns:mvc="http://www.springframework.org/schema/mvc"
xsi:schemaLocation="
http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context.xsd
http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx.xsd
http://www.springframework.org/schema/aop
```

（4） spring 总配置文件

applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"
xmlns:p="http://www.springframework.org/schema/p"
xmlns:aop="http://www.springframework.org/schema/aop"
xmlns:tx="http://www.springframework.org/schema/tx"
xmlns:task="http://www.springframework.org/schema/task"
xsi:schemaLocation="
http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context.xsd
http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx.xsd
http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop.xsd">
<!-- 加载系统配置文件
<context:property-placeholder location="classpath:*.properties" />-->
<!-- 扫描注解 -->
<context:component-scan base-package="com.bjpowernode.crm.service" />
<!-- 导入数据相关配置 -->
<import resource="applicationContext-datasource.xml" />
</beans>
```

（5） web.xml 配置

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xmlns="http://java.sun.com/xml/ns/javaee"
         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
         id="dataservice" version="3.0">
  <display-name>dataservice application</display-name>
  <!-- spring 监听器加载 applicationContext.xml 配置文件 -->
  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath:applicationContext.xml</param-value>
  </context-param>
  <listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
  </listener>
  <!-- spring 字符过滤器 -->
  <filter>
    <filter-name>encodingFilter</filter-name>
    <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
    <init-param>
      <param-name>encoding</param-name>
      <param-value>UTF-8</param-value>
    </init-param>
   </filter>
  <filter-mapping>
    <filter-name>encodingFilter</filter-name>
    <url-pattern>/*</url-pattern>
  </filter-mapping>
  <!-- Spring mvc 分发 servlet -->
  <servlet>
    <servlet-name>dispatcher</servlet-name>
     <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
    <init-param>
      <param-name>contextConfigLocation</param-name>
      <param-value>classpath:applicationContext-mvc.xml</param-value>
    </init-param>
      <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>dispatcher</servlet-name>
    <url-pattern>/</url-pattern>
   </servlet-mapping>
  <servlet-mapping>
      <servlet-name>dispatcher</servlet-name>
    <url-pattern>*.do</url-pattern>
  </servlet-mapping>
  <!-- 欢迎页，默认进入 index controller -->
```

（6） 设置 maven 对配置文件的编译选项

pom.xml

```
<resources>
		<resource>
			<directory>src/main/java</directory>
			<includes>
				<include>**/*.xml</include>
			</includes>
		</resource>
		<resource>
			<directory>src/main/resources</directory>
			<includes>
				<include>**/*.*</include>
			</includes>
		</resource>
</resources>
```

2.2.5 添加页面及静态资源

commons/
css/
images/
img/
js/
*.jsp

2.2.6 创建数据持久层接口及映射文件

采用 mybatis 的逆向工程。

2.3 创建 mybatis 的逆向工程

2.3.1 创建 maven 版的 java 工程

2.3.2 添加 mybatis 逆向工程插件

```
<!--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>
```

2.3.3 提供 mybatis 逆向工程配置文件

generatorConfig.xml

2.3.4 运行插件，生成实体层和持久层代码

第 3 章 业务功能开发

3.1 首页功能

用户访问项目首页，首先进入登录页面。

3.2 用户登录

用户在登录页面,输入用户名和密码,点击"登录"按钮或者回车,完成用户登录的功能.

*用户名和密码不能为空

*用户名或者密码错误,用户已过期,用户状态被锁定,ip 受限 都不能登录成功

*登录成功之后,所有业务页面显示当前用户的名称

*实现 10 天记住密码

*登录成功之后,跳转到业务主页面

*登录失败,页面不跳转,提示信息

3.3 安全退出

用户在任意的业务页面,点击"退出"按钮,弹出确认退出的模态窗口;用户在确认退出的模态窗口,点击"确定"按钮,完成安全退出的功能.

*安全退出,清空 cookie,销毁 session

*退出完成之后,跳转到首页

3.4 登录验证

登录验证.

用户访问任何业务资源,都需要进行登录验证.

*只有登录成功的用户才能访问业务资源

*没有登录成功的用户访问业务资源,跳转到登录页面

3.5 数据字典类型维护

用户打开”系统设置”-->”数据字典表”，显示字典类型列表，完成数据字典类型的增删改查。

1、 查询数据字典类型

用户在业务主页面，点击”系统设置”，跳转到系统设置主页面；

用户在系统设置主页面，点击”数据字典表”，跳转到数据字典维护主页面；

在数据字典维护主页面，默认在工作区中显示数据字典类型列表。

2、创建数据字典类型：

用户在数据字典类型主页面，点击”创建”按钮，跳转到创建页面；

用户在创建页面，填写表单，点击”保存”按钮，完成创建数据字典类型的功能.

*编码不能为空、不能重复

*创建成功之后，跳转数据字典类型主页面；

*创建失败，提示信息，页面不跳转。

3、修改数据字典类型

用户在数据字典类型主页面，选择要修改的记录，点击”编辑”按钮，跳转到修改记录的页面；

用户在修改记录的页面，填写表单，点击”更新”按钮，完成修改数据字典类型的功能。

*每次必须修改一条记录,而且只能修改一条记录

*数据字典类型的编码不能够修改

*修改成功之后，跳转到数据字典类型的主页面

*修改失败，提示信息，页面不跳转

4、删除数据字典类型

用户在数据字典类型主页面，选择要删除的记录，点击”删除”按钮，弹出确认删除对话框，用户点击”确定”，完成删除数据字典类型的功能。

*每次至少删除一条

*可以批量删除

*删除成功之后，刷新数据字典类型列表

*删除失败，提示信息，列表不刷新

3.6数据字典值维护

用户打开”系统设置”-->”数据字典表”-->”字典值”，显示字典值列表，完成数据字典值的增删改查。

1、 查询数据字典值

用户在数据字典主页面，点击”字典值”菜单，在工作区中显示数据字典值主页面；

在数据字典值主页面显示所有数据字典值的记录。

2、 创建数据字典值

用户在数据字典值主页面，点击”创建”按钮，跳转到创建数据字典值的页面；

用户在创建数据字典值的页面填写表单，点击”保存”按钮，完成创建数据字典值的功能。

*字典类型编码来自于数据库，并且不能为空

*字典值也不能为空

*创建成功之后，跳转到数据字典值主页面

*创建失败，提示信息，页面不跳转

3、修改数据字典值

用户在数据字典值主页面，选择要修改的记录，点击”编辑”按钮，跳转到修改页面；

用户在修改数据字典值页面，填写表单，点击”更新”按钮，完成修改数据字典值的功能。

*每次只能修改一条记录，而且必须修改一条

*”所属字典类型”字段不能修改

*”字典值”不能为空

*修改成功之后，跳转到数据字典值主页面

*修改失败，提示信息，页面不跳转

4、删除数据字典值

用户在数据字典值主页面，选择要删除的记录，点击”删除”按钮，弹出确认删除对话框,用户点击”确定”，完成删除数据字典值的功能。

*每次至少删除一条记录

*删除成功之后，跳转到数据字典值主页面

*删除失败，提示信息，页面不跳转

3.7创建市场活动

创建市场活动.

用户在市场活动主页面,点击"创建"按钮,弹出创建市场活动的模态窗口;

用户在创建市场活动的模态窗口填写表单,点击"保存"按钮,完成创建市场活动的功能.

*所有者是动态的(//在现实市场活动主页面时，就从数据库中查询出所有用户并且显示在创建的模态窗口中)

*所有者和名称不能为空

*如果开始日期和结束日期都不为空,则结束日期不能比开始日期小

*成本只能为非负整数

*创建成功之后,关闭模态窗口,刷新市场活动列，显示第一页数据，保持每页显示条数不变

*创建失败,提示信息创建失败,模态窗口不关闭,市场活动列表也不刷新

3.8查询市场活动

当市场活动主页面加载完成之后,显示所有数据的第一页;

用户在市场活动主页面填写查询条件,点击"查询"按钮,显示所有符合条件的数据的第一页，保持每页显示条数不变

实现翻页功能.

*在市场活动主页面,显示市场活动列表和记录的总条数

*默认每页显示条数:10

3.9修改市场活动

用户在市场活动主页面,选择要修改的市场活动,点击"修改"按钮,弹出修改市场活动的模态窗口;

用户在修改市场活动的模态窗口填写表单,点击"更新"按钮,完成修改市场活动的功能.

*每次能且只能修改一条市场活动

*所有者 动态的

*表单验证(同创建)

*修改成功之后,关闭模态窗口,刷新市场活动列表,保持页号和每页显示条数都不变

*修改失败,提示信息,模态窗口不关闭,列表也不刷新

3.10 删除市场活动

用户在市场活动主页面,选择要删除的市场活动,点击"删除"按钮,弹出确认窗口;

用户点击"确定"按钮,完成删除市场活动的功能.

*每次至少删除一条市场活动

*可以批量删除市场活动

*删除成功之后,刷新市场活动列表,显示第一页数据,保持每页显示条数不变

*删除失败,提示信息,列表不刷新

3.11 批量导出市场活动

用户在市场活动主页面,点击"批量导出"按钮,把所有市场活动生成一个 excel 文件,弹出文件下载的对话框;

用户选择要保存的目录,完成导出市场活动的功能.

*导出成功之后,页面不刷新

3.12 选择导出市场活动

用户在市场活动主页面,选择要导出的市场活动,点击"选择导出"按钮,把所有选择的数据生成一个 excel 文件,弹出文件下载的对话框;

用户选择要保存的目录,完成选择导出市场活动的功能.

*每次至少选择导出一条记录

*导出成功之后,页面不刷新

3.13 导入市场活动

用户在市场活动主页面,点击"导入"按钮,弹出导入市场活动的模态窗口;

用户在导入市场活动的模态窗口选择要上传的文件,点击"导入"按钮,完成导入市场活动的功能.

*只支持.xls

*文件大小不超过 5MB

*导入成功之后,提示成功导入记录条数,关闭模态窗口,刷新市场活动列表,显示第一页数据,保持每页显示条数不变

*导入失败,提示信息,模态窗口不关闭,列表也不刷新

3.14 查看市场活动明细

用户在市场活动主页面,点击市场活动名称超级链接,跳转到明细页面,完成查看市场活动明细的功能.

*在市场活动明细页面,展示:

-市场活动的基本信息

-该市场活动下所有的备注信息

3.15 添加市场活动备注

用户在市场活动明细页面,输入备注内容,点击"保存"按钮,完成添加市场活动备注的功能.

*备注内容不能为空

*添加成功之后,清空输入框,刷新备注列表

*添加失败,提示信息,输入框不清空,列表也不刷新

3.16 删除市场活动备注

用户在市场活动明细页面,点击"删除"市场活动备注的图标,完成删除市场活动备注的功能.

*删除成功之后,刷新备注列表

*删除失败,提示信息,备注列表不刷新

3.17 修改市场活动备注

用户在市场活动明细页面,点击"修改"市场活动备注的图标,弹出修改市场活动备注的模态窗口;

用户在修改市场活动备注的模态窗口,填写表单,点击"更新"按钮,完成修改市场活动备注的功能.

*备注内容不能为空

*修改成功之后,关闭模态窗口,刷新备注列表

*修改失败,提示信息,模态窗口不关闭,列表也不刷新

3.18 创建线索

用户在线索主页面，点击”创建”按钮，弹出创建线索的模态窗口；
用户在创建线索的模态窗口，填写表单，点击”保存”按钮，完成创建线索的功能。
*所有者、称呼、线索状态、线索来源 是动态
*表单验证
*创建成功之后，关闭模态窗口，刷新线索列表，显示第一页数据，保持每页显示条数不变
*创建失败，提示信息，模态窗口不关闭，列表也不刷新。

3.19 查询线索

当线索主页面加载完成之后,显示所有数据的第一页;
用户在线索主页面填写查询条件,点击"查询"按钮,显示所有符合条件的数据的第一页;
实现翻页功能.

*在线索主页面,显示市场活动列表和记录的总条数

*默认每页显示条数:10

3.20 查看线索明细

用户在线索主页面,点击线索名称(fullname 和 appellition)超级链接,跳转到线索明细页面,完成查看线索明细的功能.

*在线索明细页面,展示:

-线索的基本信息

-线索的备注信息

-跟该线索相关联的市场活动信息

3.21 线索关联市场活动

用户在线索明细页面,点击"关联市场活动"按钮,弹出线索关联市场活动的模态窗口;
用户在线索关联市场活动的模态窗口,输入搜索条件,每次键盘弹起,根据名称模糊查询市场活动,把所有符合条件的市场活动显示到列表中;用户选择要关联的市场活动,点击"关联"按钮,完成线索关联市场活动的功能.

*每次至少关联一个市场活动

*同一个市场活动只能跟同一个线索关联一次

*关联成功之后,关闭模态窗口,刷新已经关联过的市场活动列表

*关联失败,提示信息,模态窗口不关闭,已经关联过的市场活动列表也不刷新

3.22 解除线索关联市场活动

用户在线索明细页面,点击某一个"解除关联"按钮,弹出确认解除的窗口;
用户点击"确定"按钮,完成解除线索关联市场活动的功能.

*解除成功之后,刷新已经关联的市场活动列表

*解除失败,提示信息,列表也不刷新

3.23 线索转换

用户在线索明细页面,点击"转换"按钮,跳转到线索转换页面;

用户在线索转换页面,如果需要创建创建交易,则填写交易表单数据,点击"转换"按钮,完成线索转换的功能.

*在线索转换页面,展示:fullName,appellation,company,owner

*市场活动源是可搜索的

*数据转换:

把线索中有关公司的信息转换到客户表中

把线索中有关个人的信息转换到联系人表中

把线索的备注信息转换到客户备注表中一份

把线索的备注信息转换到联系人备注表中一份

把线索和市场活动的关联关系转换到联系人和市场活动的关联关系表中

如果需要创建交易,还要往交易表中添加一条记录
如果需要创建交易,还要把线索的备注信息转换到交易备注表中一份
删除线索的备注
删除线索和市场活动的关联关系
删除线索

在一同个事务中完成.

*转换成功之后,跳转到线索主页面
*转换失败,提示信息,页面不跳转

3.24 创建交易

用户在交易主页面，点击”创建”按钮，跳转到创建交易的页面；
用户在创建交易的页面填写表单，点击”保存”按钮，完成创建交易的功能。
*所有者、阶段、类型、来源 都是动态的
*市场活动源是可搜索的
*联系人也是可搜索的
*可能性是可配置的
*客户名称支持自动补全
*表单验证
*保存成功之后，跳转到交易主页面
*保存失败，提示信息，页面不跳转

3.25 查看交易明细

用户在交易主页面，点击交易名称超级链接，跳转到交易明细页面，完成查看交易明细的功能。
*显示交易的基本信息
*显示交易的备注信息
*显示交易的历史信息
*显示交易的阶段图标信息

3.26 修改交易阶段

用户在交易明细页面，点击交易阶段的图标，把交易当前的阶段修改为指定的阶段，完成修改交易阶段的功能。
*已经成交的交易不能修改阶段
*修改成功之后，更新：
--交易的图标信息
--交易的基本信息
--交易的历史信息
*修改失败，提示信息，页面不更新

3.27 交易统计图表

用户点击”交易统计图表”菜单，显示交易统计图表页面，以销售漏斗图的形式显示交易表中各个阶段的记录数量，完成查看交易统计图表的功能。