**第1章-项目介绍&环境搭建**

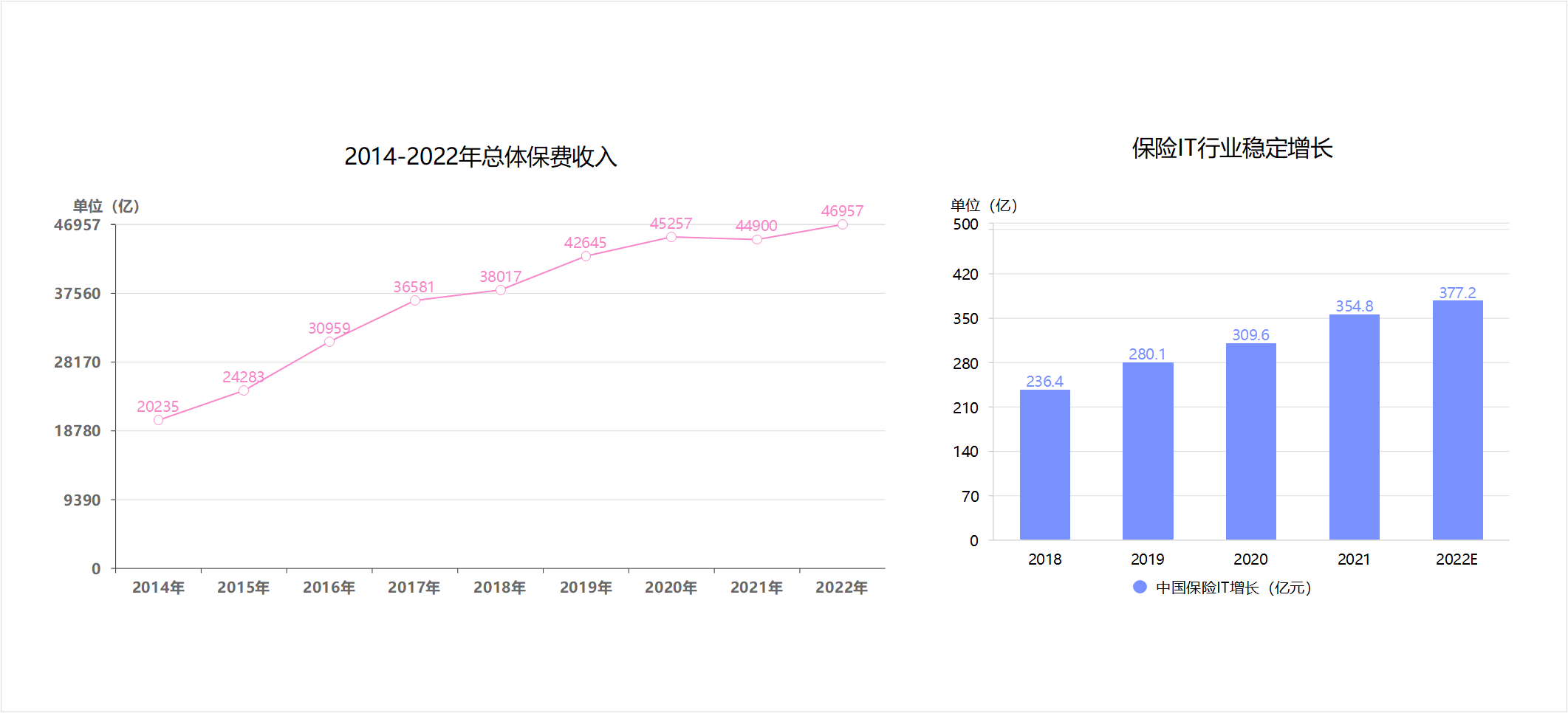
**学习目标**

* 了解保险行业背景与发展趋势前景
* 掌握《四方保险》项目核心架构、项目模板、技术架构、数据库总述
* 快速搭建《四方保险》项目环境并建立git项目管理

**1、背景与趋势**

**1.1、行业背景**

在过去的十年内保险行业的发展如何？IT行业对于保险行业的发展又如何呢？下面我们来看下一组数据：



上述数据可以看出从2014开始保险的总体保费收入在逐年上升，并且中国的保险IT行业呈现稳定增长趋势：

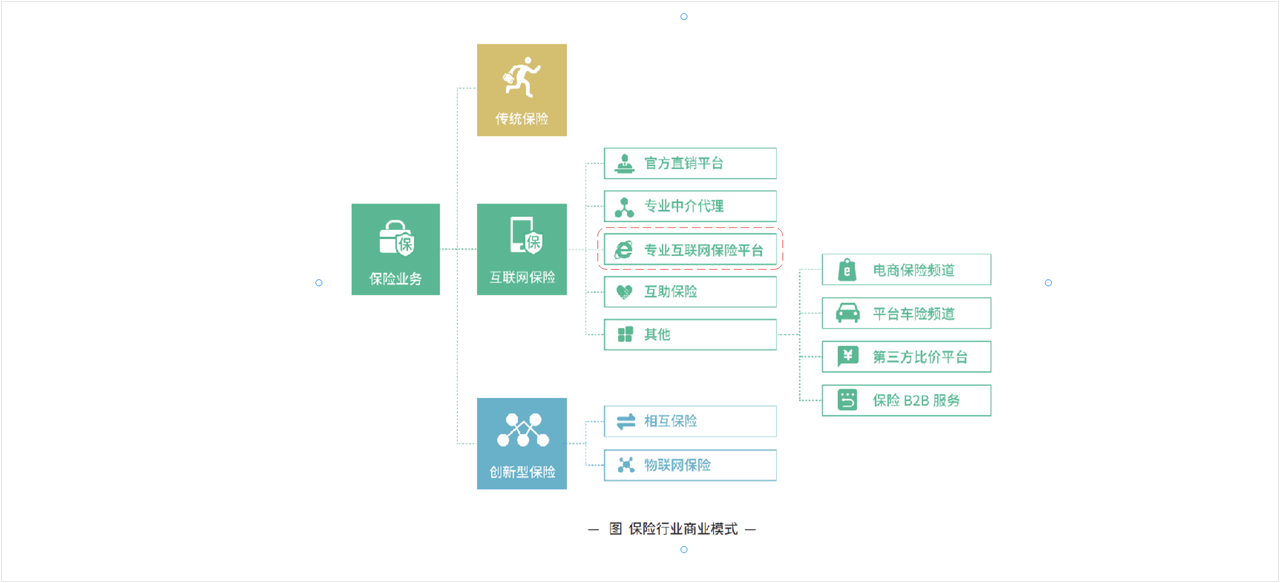
* 数字化转型已成为保险行业的趋势，这为保险IT市场创造了巨大的机会
* IT技术为保险公司提供了创新工具和支持，有助于提高业务效率和客户体验
* 人工智能、大数据分析、规则引擎计算等先进技术的应用使保险公司能够更好地评估风险、优化产品，实现个性化定制
* 从2018年到2021年，保险IT市场投资规模增长了约128.4亿元，这表明行业在资本支持方面表现强劲
* 平均复合增速为12.06%，显示了市场的持续健康发展。示着市场在未来依然有望保持增长。

对于学习保险类项目对于后期的个体发展、职业规划将会有巨大的支持：

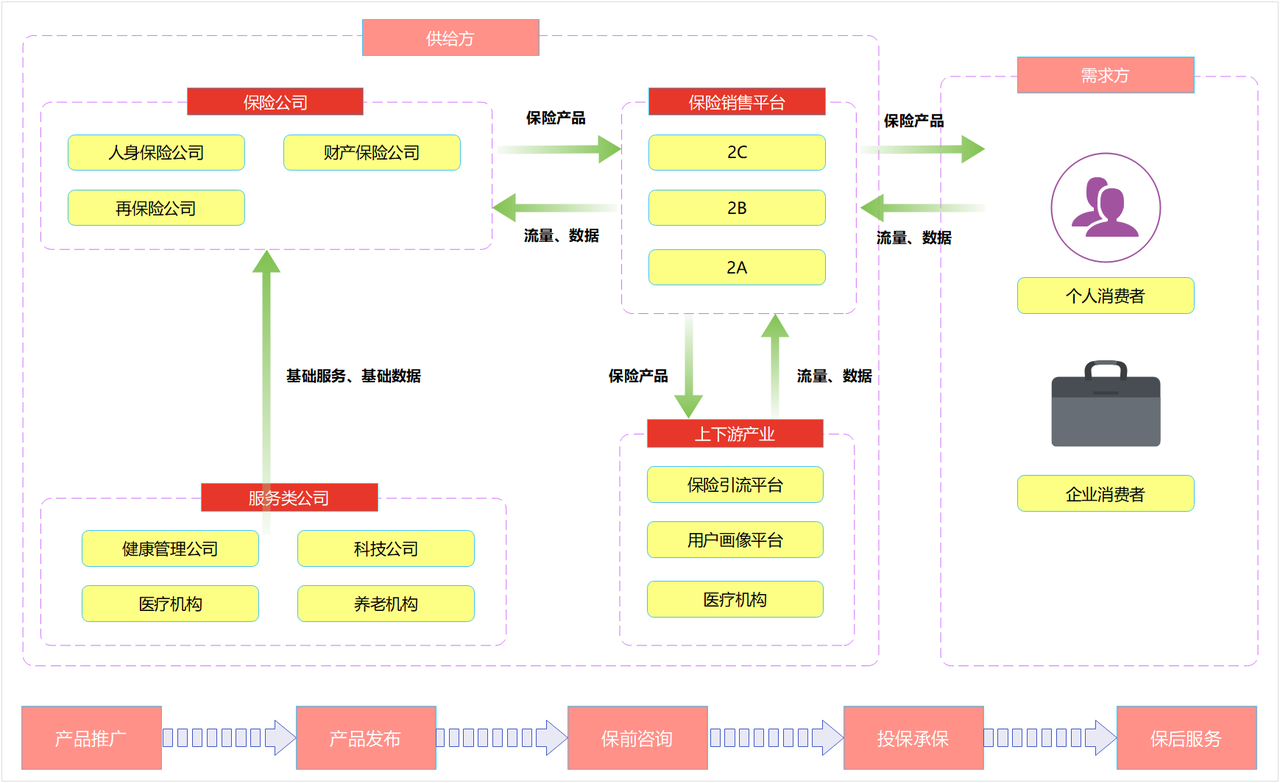
* 拥有特有行业的项目经验，提升个人在就业职业发展竞争力
* 提升个人对于复杂业务系统的设计能力、开发能力、维护能力

**1.2、项目背景**

在整个金融保险行业，保险IT行业是保险行业数字化转型的重要组成部分，涵盖了多个细分领域。其中包括传统保险业务系统，官方直销系统，专业中介代理平台，互联网保险平台，以及其他保险系统。学习《四方保险》项目后各位小伙伴可以向不同方向的保险项目进行个人发展学习。



《四方保险》属于专业互联网保险平台（**保险销售平台**），是一种在线服务平台，通过互联网技术和数据分析等手段，为用户提供方便快捷的保险服务。这些平台通常涵盖了整个保险业务流程，从产品购买、保单管理到理赔服务，主要作用是提高用户体验、降低交易成本、加强风险、可以**对比多家保险公司的保险产品**管理。



|  |
| --- |
| Tips：互联网保险知多点  2C 表示to customer，即产品是直接面向客户；一般保险公司自己网站直接销售保险产品属于这种类型  2B 表示to branch，即产品是直接面向渠道；比如：中介公司、互联网公司都属于渠道  2A 表示to agent，即保险代理人；代理人自行拉客户 |

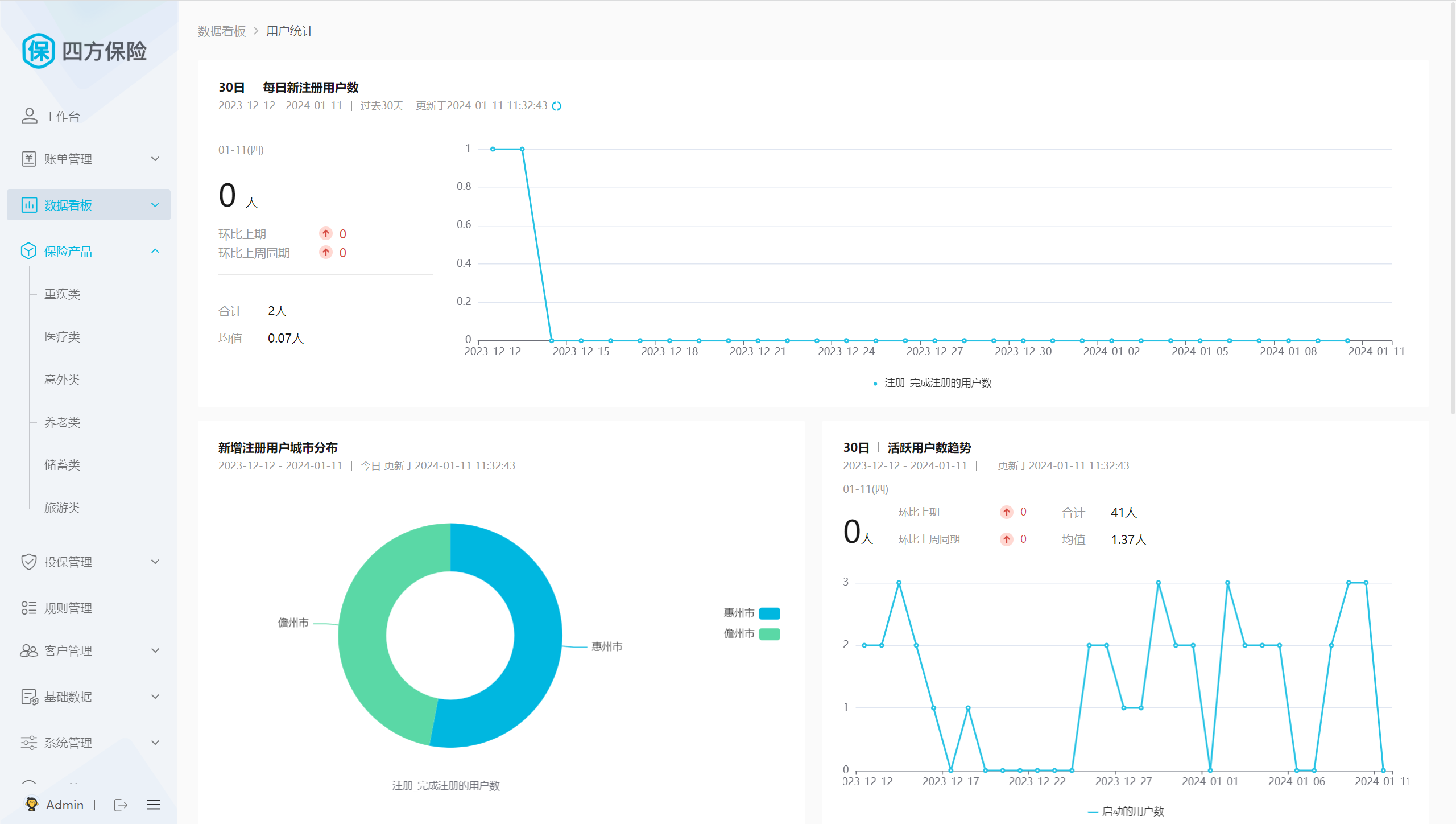
《四方保险》项目中功能介绍：

* 在线购买和投保： 用户可以通过互联网平台方便地浏览、**比较和购买不同保司的各类保险产品**，完成投保流程。
* 个性化定制产品： 平台可能提供根据用户需求定制的保险产品，满足个性化的保险需求。
* 数字化保单管理： 用户可以在平台上轻松管理和查看自己的保单信息，包括保单状态、保费支付等。
* 在线理赔服务： 提供在线理赔服务，用户可以通过平台提交理赔申请、上传必要的文件和信息，加速理赔流程。
* 风险评估和数据分析： 利用大数据和规则引擎技术对用户进行风险评估，提供更精准的保险方案。
* 移动端应用支持： 提供移动端应用，方便用户随时随地进行保险业务的管理和操作。
* 数字化营销和推广： 利用数字化营销手段，通过社交媒体、搜索引擎等渠道推广保险产品，吸引更多用户。
* 合作伙伴关系： 与其他互联网平台、金融机构等建立合作伙伴关系，拓展渠道，提供更全面的保险服务。

**1.3、项目演示**

管理端演示地址：http://sf.mgt.itheima.net/

默认账号：admin@qq.com 密码：pass



用户端演示地址：http://sf.app.itheima.net/

默认账号：15156403088 默认验证码：123456



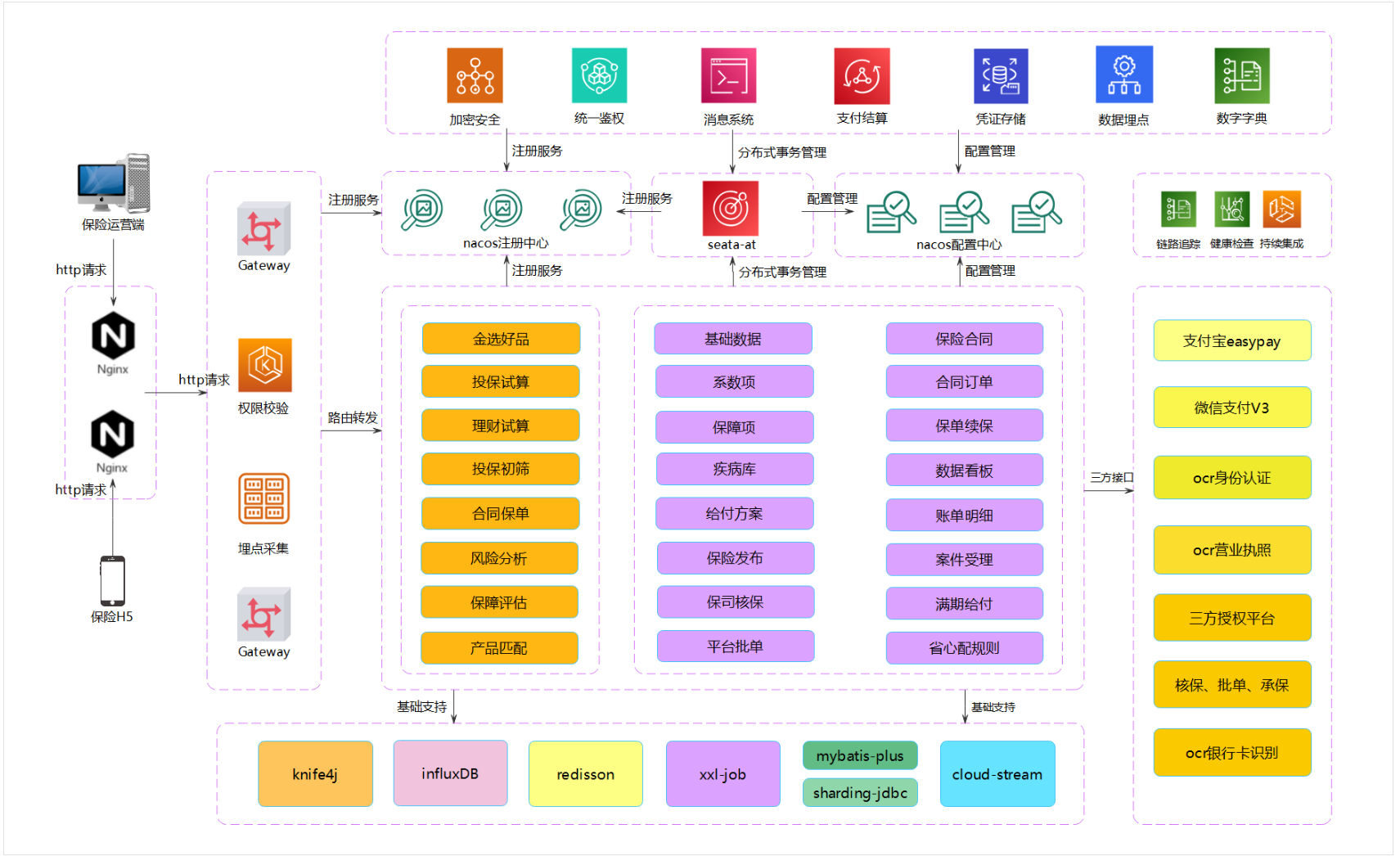
**2、项目介绍**

**2.1、系统架构**

四方保险属于一个保险销售平台；有两个终端：

* 后台管理端
* APP用户端

整体架构如下：



**代理服务：**保险平台和保险销售平台通过的nginx、gateway网关

**路由处理：**gateway网关负责路由通用服务和业务服务，同时做数据埋点和网关权限校验

**服务治理：**gateway网关、通用服务、业务服务、seata等服务注册和配置都放入nacos中统一管理

**通用服务：**抽离出加密安全、统一鉴权、消息系统、支付结算、配置存储、数据埋点、数字字典等基础微服务

**基础支持：**在线文档knife4j、时序数据库influxDB、redisson缓存客户端、xxl-job计划任务、spring-cloud-stream

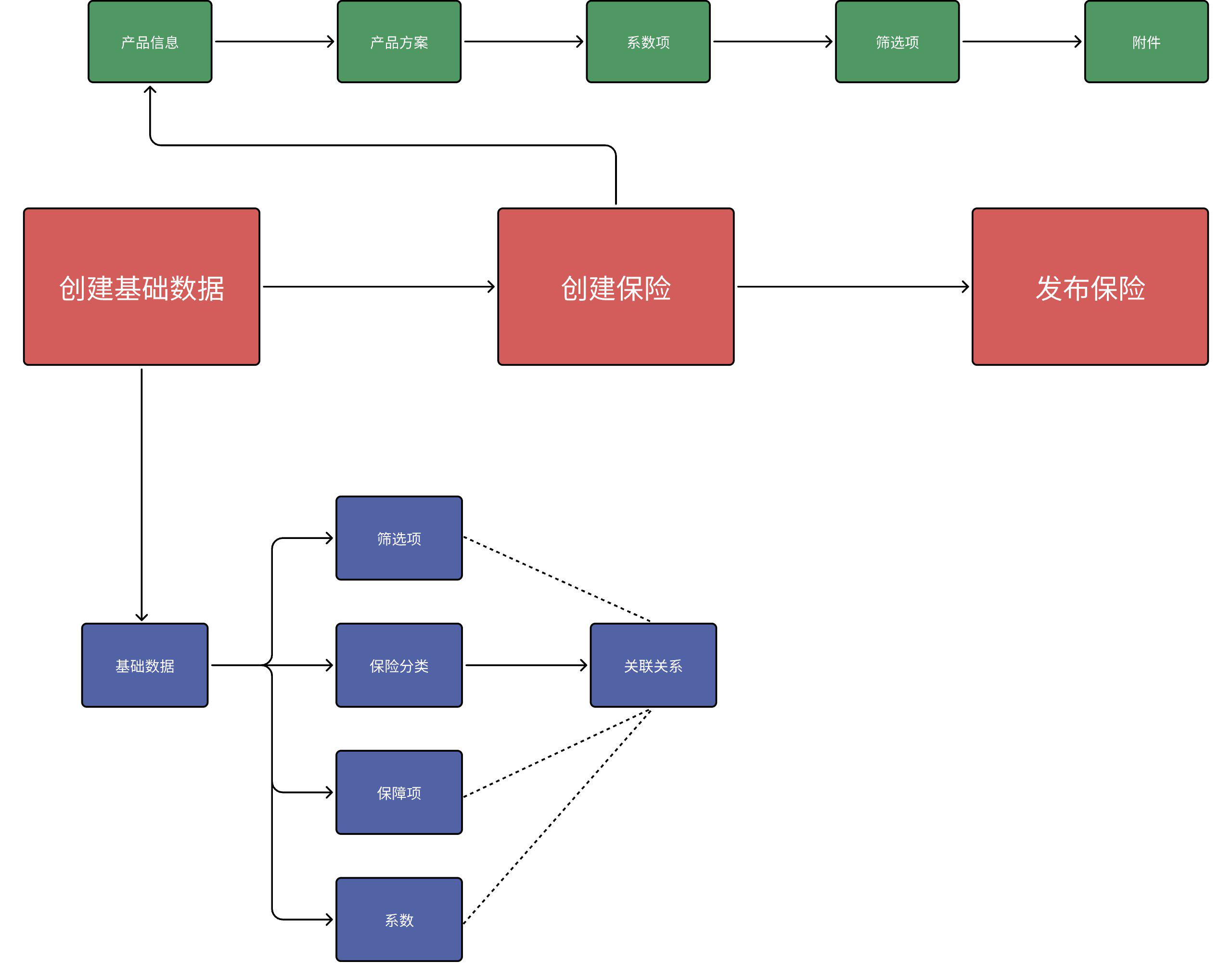
**外部服务：**支付宝、微信支付、ocr、三方核保、批单、承保平台

**2.2、核心业务**

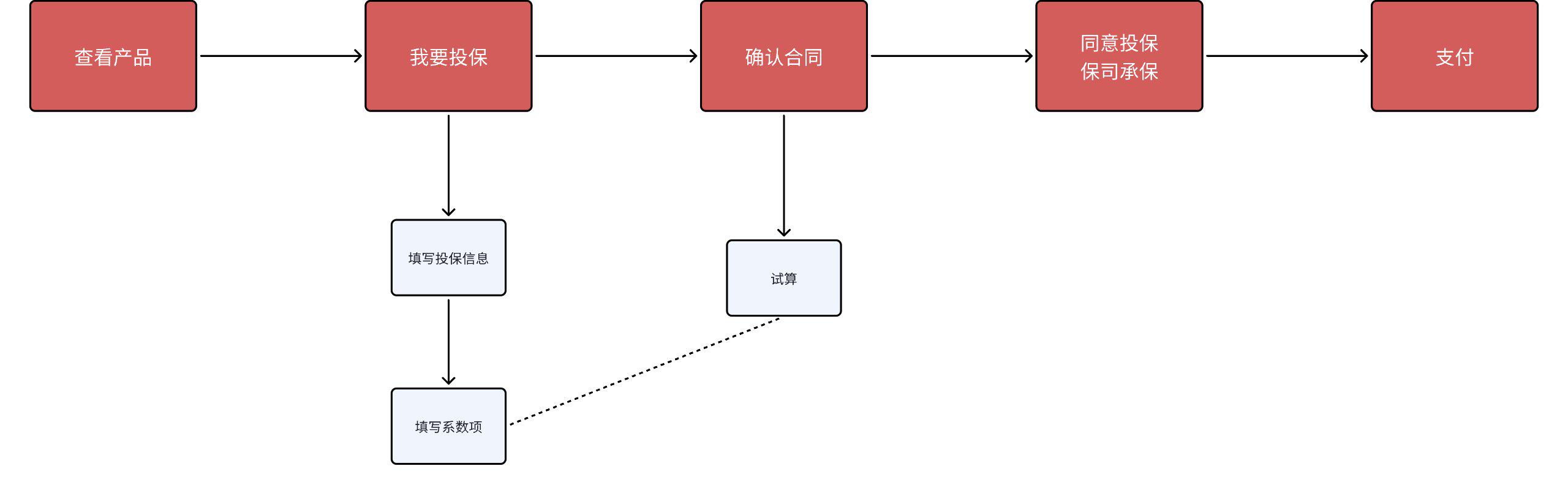
产品原型

* 用户端：https://codesign.qq.com/s/ALwE9VdDxKZX1Dp 密码TBOG
* 管理后台：https://codesign.qq.com/s/Yyg5Zpdwm6Z2lKv 密码YZ4R

管理后端的核心流程：



app端核心流程：



**2.3、课程内容**

* 项目介绍&环境搭建
* 保险基础数据管理-分析与设计
* 保险产品
* 产品附件—对象存储
* 产品详情-性能优化与数据脱敏
* 投保试算-保障类
* 投保试算-理财类
* 投保处理-投保处理
* 支付处理-一次性支付
* 支付处理-周期性扣款
* 数据埋点
* 数据中心——时序数据库
* 短信服务

**3、环境搭建**

在了解了系统架构和核心业务流程后；我们把项目跑起来直接运行感受会更加直接。项目的运行需要使用到系列的组件等环境；步骤如下：

1）导入虚拟机到vmware；验证各个组件启动情况

2）IDEA打开项目并启动；

3）启动前端应用服务器；

4）设置系统hosts文件中IP与域名；

详细的操作请参考：[第0章-环境搭建](https://j1wtmv7ajj.feishu.cn/wiki/FG5AwAXHCivrwskoxTIcQcSfn4e)

**4、修复BUG**

在刚刚进入项目组后，一般不会布置开发任务，而是先熟悉项目代码。为了帮助大家熟悉整个项目，我们预留了个BUG，让大家在修复BUG的过程中熟悉项目代码。

一般修复BUG的过程是这样的:

* 熟悉项目
* 重现bug
* 分析解决
* 测试

因此，解决BUG的过程，就是熟悉项目的过程。

**4.1、熟悉项目**

熟悉项目的第一步是熟悉项目的结构、用到的技术、编码的一些规范等。

**4.1.1、项目结构**

下面我们介绍一下各个模块的主要职能：

├── sfbx-cloud # 项目模块父工程，统一管理JAR版本和插件内容

│ ├── sfbx-apache-httpclient # 远程调用通信框架（第三方、保司接口调用）

│ ├── sfbx-dict # 数字字典：其他功能用到的常用字段维护

│ ├── sfbx-file # 对象存储：支持OSS、七牛云的文件存储服务

│ ├── sfbx-framework # 基础支持：缓存、消息中间件、分布式事务、线程池、脱敏、调度等支持

│ ├── sfbx-gateway # 网关路由：销售端、运营端、接口端网关路由

│ ├── sfbx-insurance # 保险业务：四方保险主业务服务包括用户端和管理端

│ ├── sfbx-points # 数据埋点：数据采集、存储、分析

│ ├── sfbx-rule # 规则引擎：规则引擎的UI支持服务

│ ├── sfbx-security # 权限系统：基于oauth2.0的权限管理

│ ├── sfbx-sms # 短信服务：支持阿里云短信、简单短信、腾讯云短信的短信微服务

│ ├── sfbx-task # 计划监听：统一调度响应和监听

│ ├── sfbx-trade # 交易服务：支持支付宝、微信支付的多种场景的支付平台

|  |
| --- |
| 注意：  一般正式使用和发布的系统；如果所有的微服务加起来，数据库中的表大概200+  例如：商城 300 - 500张表； 四方保险类的大概300左右 |

**4.1.2、开发规范**

* 在项目模块中；\*-app的是针对app端的应用服务；\*-mgt的是针对web后台管理的应用服务；
* 项目中带有 \*-interface的模块是feign客户端放置的模块工程；而应用到的 dto 放置在framework-commons模块中，该模块中的实体类以\*VO结尾命名；
* 有需要使用到其它微服务提供的接口的话就引入对应的 \*-interface 即可；
* 一般项目中，如 insurance-mgt 的 feign 包内的相当于当前服务中某些业务对外提供的接口功能（不是接口）；这些对外业务接口处理器的请求路径命名一般为 \*-feign



**4.1.3、配置文件**

四方保险的项目公共配置项目全部都由nacos管理；所以在每个项目中会有一个 bootstrap.yml 的启动配置文件。这个文件几乎不需要修改，只保留了最基础的内容；而对应项目需要使用到的公共组件或配置都在这个配置文件中引入了，要修改的话需要到nacos的配置中找对应的配置项修改即可。



**4.2、复现BUG**

1）访问 http://sf.app.itheima.net/#/pages/login/login 点击 注册

2）注册新账号之后，使用新账号登录 http://sf.app.itheima.net/#/pages/login/login

3）任意查看一个保险产品；如：

|  |  |  |
| --- | --- | --- |
|  |  |  |

在上述的 为谁买 填写了 姓名、身份证号码 之后；点击 保存 。

|  |
| --- |
| 注意；不用点击 我要投保 |

4）返回到首页——》再次找任意一个产品；产品详情页面中发现刚刚填写的 本人 信息还是空。



**4.3、修复BUG**

**4.3.1、分析**

如果是我们自己写的代码，肯定很容易找到业务入口、整个业务线路。但现在我们是接手他人项目，所以只能通过其它途径来梳理业务:

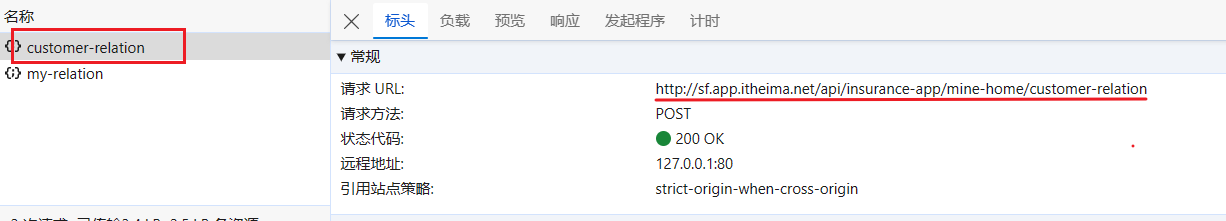
* 1)如果开发业务的同事还在，直接与开发该业务的同事交流
* 2)如果开发者已离职，可以查看相关接口文档
* 3)如果没有文档，也可以查看前端请求，顺藤摸瓜

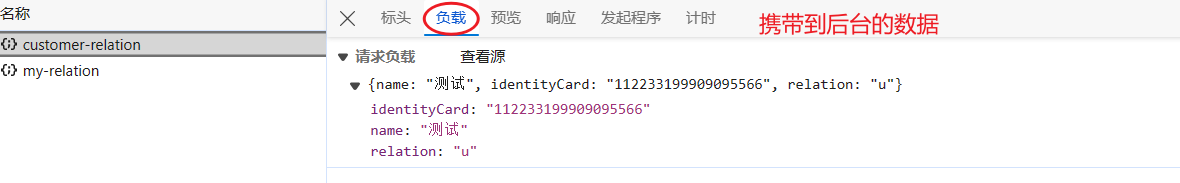
此处由于我们没有人可以交流，只能通过查看前端请求来分析了。

**1）保存**



上述的 保存 的时候；使用浏览器查看其请求地址如下：

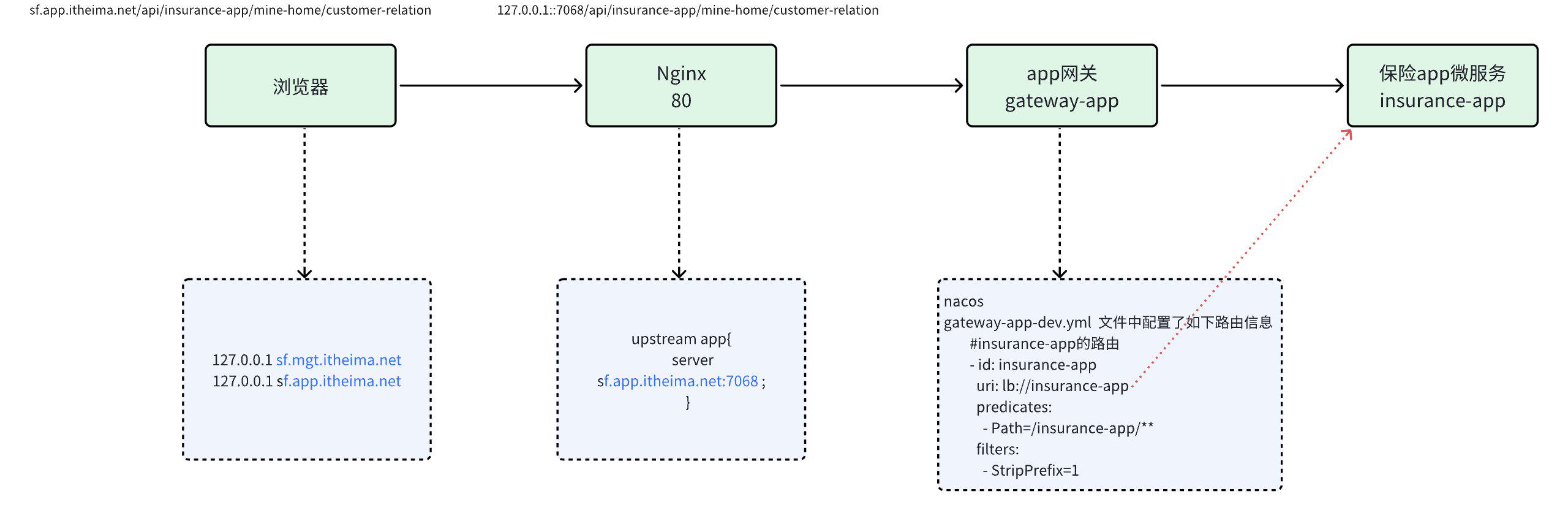




按照之前我们的环境部署方案，sf.app.itheima.net 这个域名会被解析到127.0.0.1这个地址，然后被Nginx反向代理到网关微服务。

而网关则会根据请求路径和路由规则，把请求再路由到具体微服务。这里请求路径以/api/insurance-app开头，对应的微服务是insurance-app，也就是保险app端微服务。

这样，整个请求链路就比较清楚了：



找到了具体的微服务，接下来，我们就进入微服务，查看对应源码，找出问题即可。

请求到达保险app服务后的路径是 /insurance-app/mine-home/customer-relation，对应的controller是：

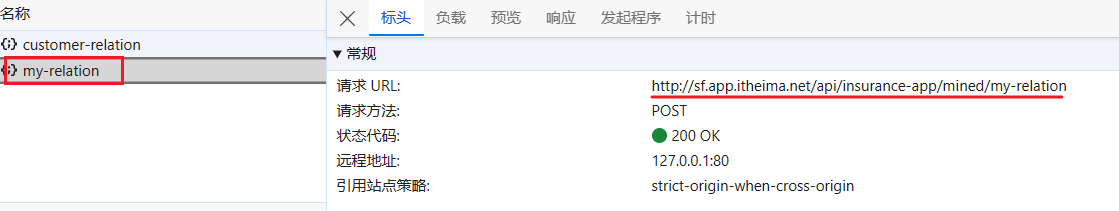


跟入service代码；发现是比较正常的，保存也没有报错；但是为何再次返回详情页面的时候没有查到呢？

接下来；我们看在详情页面的时候；它是怎么查的。

**2）详情页面查询**

还是在app端；点击产品后跳转到的详情页面；发现有如下的请求：



尝试测试之后；只有在返回的数据中 relation 字段的值为 0 的情况下；才会回显。而提交的时候是 u。

说明前端在上述的请求返回数据之后；根据返回的对象的 relation 的值进行了过滤显示。relation属性的值必须要为 0 。

**4.3.2、修复**

已经分析问题：

* 1、在用户注册的时候；携带的状态为 u 保存到了 sfbx-insurance.tab\_customer\_relation数据库表中
* 2、而查询的时候，回显的时候；前端中对 relation的值为 0才会回显；

解决方法：

找到注册时候；保存的业务代码，在前端不配合或者不便于修改的情况下；我们在保存的业务代码中，将 u替换为 0 即可

修改如下：



再次进行测试即可。

**5、数据库建模-Power Designer**

在需求分析之后；需要进行数据库表的创建，如果系统功能模块多，也就是数据库表多；逐个表创建不太现实，而且表之间的关系不那么明显能看出。**通过数据库建模可以更加从整体上查看对于需求的理解是否到位；数据库表设计是否存在问题。**

在更加规范或者正式的单位、机构他们要求项目验收需要对应的数据库建模文档；

* **常见的模型**
* 概念模型CDM
* 物理模型PDM
* **常见的设计工具**
* Power Designer
* pdman

**5.1、概念模型CDM**

* **为什么？**

|  |
| --- |
| 在需求、产品原型评审之后，可以通过图形的方式更加直观的了解是否对需求理解到位。还可以对后续的开发做出引导。  在做需求分析的过程中；常常看到很多要实现的模块，这些模块或者功能基本上都是要处理的实体（名词；领域模型）。例如：在原型或者需求文档中的 xx管理；xx 一般就是实体；系统中这些实体之间是有关系的，开发与其息息相关。；由这些领域模型画出来的图就是概念模型，也被称为ER图  在实际开发中；一开始立项的时候是不知道使用哪款DBMS（Oracle/DB2/Sybase/SQL Server/MySQL）的；所以不能直接基于某一款数据库创建对应的建表语句。可以先创建概念模型，一旦定下来DBMS则可以直接转换。 |

* **是什么**

用于描述**实体与实体之间的关系的图形**；也被业界称为 ER图

* **怎么画**（9字诀）

**1、列实体**

* 将原型、需求中的那些实体（名词、领域模型）先列出来；在原型中常见的有模块名称、下拉框。
* 例如：产品

**2、填属性**

根据原型、需求文档中对于上述列出的实体设置属性；做法就是结合输入和输出的综合。（输入：新增、添加 等按钮之后的页面；输出；列表、详情页）

* **基本**属性：基本的信息；一般是输入框、文本框、日期时间选择框
* **关联**属性：在页面中的下拉框
* **辅助**属性：状态、创建、更新时间
* 例如：产品（ID，保险代码，名称，类型ID，排序，状态，创建时间...）

**3、画关系**

要将需求中实体之间的关系画出来；常见的关系：

* **一对一**

经常出现在垂直拆表的时候；冷热数据的划分；或者有大字段的表会出现辅助表保存，辅助表与主表的关系就是一对一。如：商品表与商品详情表

* 两个实体共用一个主键
* **一对多（多对一）**

经常出现在有主外键关系的实体之间；页面中出现下拉框（动态变化的话）的时候就是有这种关系。

* 例如：一个保险分类可以有多个保险产品；但是一个产品只属于某个分类 ====》 一对多 (1:n)
* 例如：一个车主可以有多辆车；一辆车只属于某个车主 ===》 1:n
* **多对多**

在业务中，有业务交叉但是相互都有关联的时候；可以思考。可以如下的方式验证：

正推反推都是一对多 =》 双向一对多 =》多对多

例如：

用户与角色

* 一个用户可以有多个角色；——》一对多
* 一个角色可以被多个人使用 ——》一对多

——》多对多

**5.2、物理模型PDM**

* **为什么？**

可以通过物理模型PDM**直接创建整个数据库的表及其关系**。创建数据库及表的时候使用pdm非常方便。如果已经确定使用哪种DBMS之后或者数据库表设计比较特殊，那么也可以直接画PDM而不画CDM。

* **是什么**

是一个**描述表与表之间关系的图形**。可以通过该图形直接获得整个库的sql语句。

* **怎么画**

在确定了客户使用哪种DBMS之后；可以通过CDM直接转换为PDM。也很多公司，开发之初就确定使用哪款DBMS了；所以可以跳过概念模型直接创建物理模型。

**5.3、案例**

**需求**：将人员组织架构的CDM画出来。

人员组织架构：用于管理人员的系统；里面会涉及到：机构、部门、角色、权限、人员

**1、列实体**

* 机构、部门、角色、权限、人员

**2、填属性----》（基础，关联，辅助）**

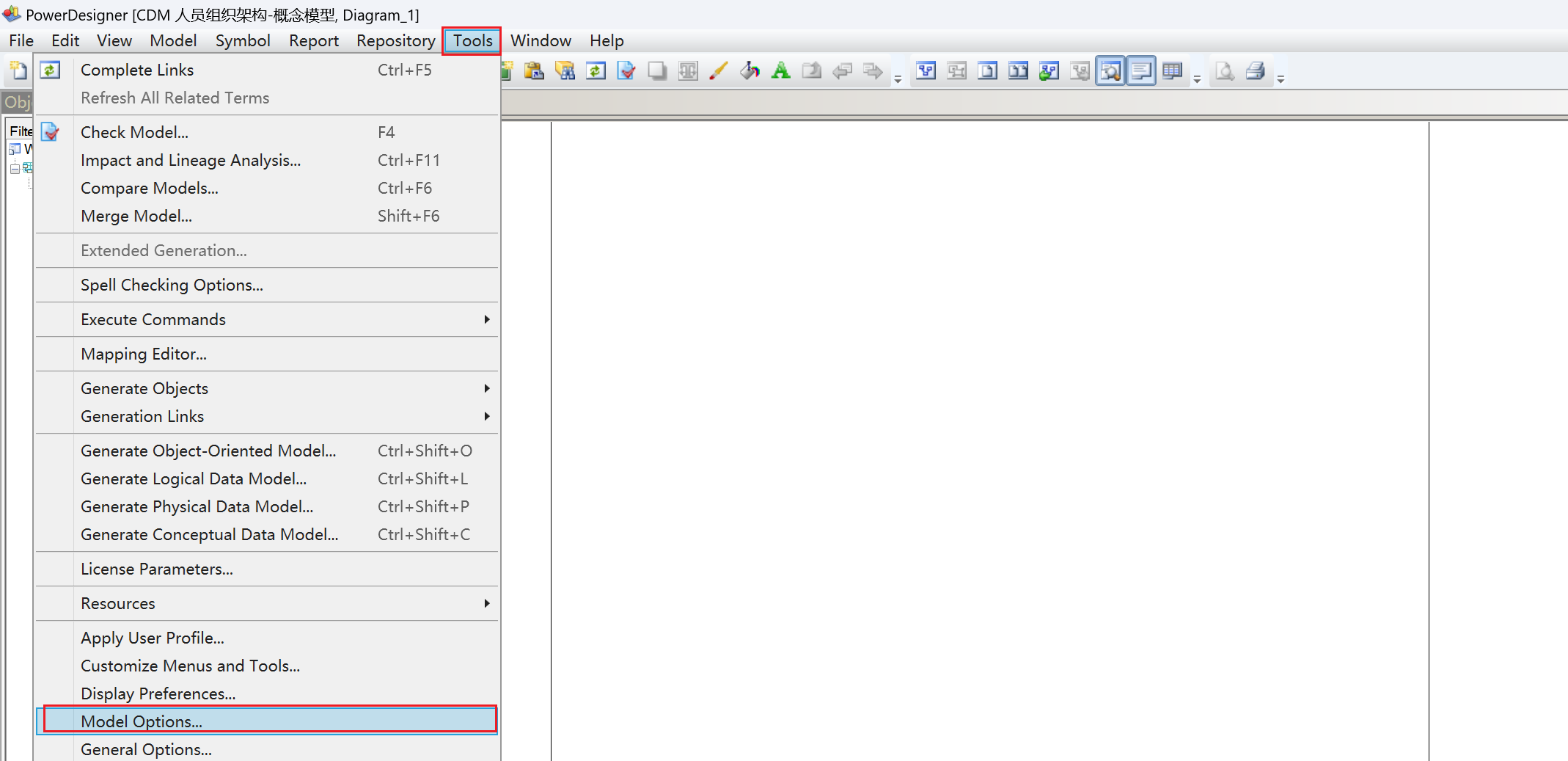
* 机构（机构ID，名称，状态）
* 部门（部门ID，名称，状态）
* 角色（角色ID，名称，状态）
* 权限（权限ID，名称，状态）
* 人员（人员ID，姓名，性别，... 状态）

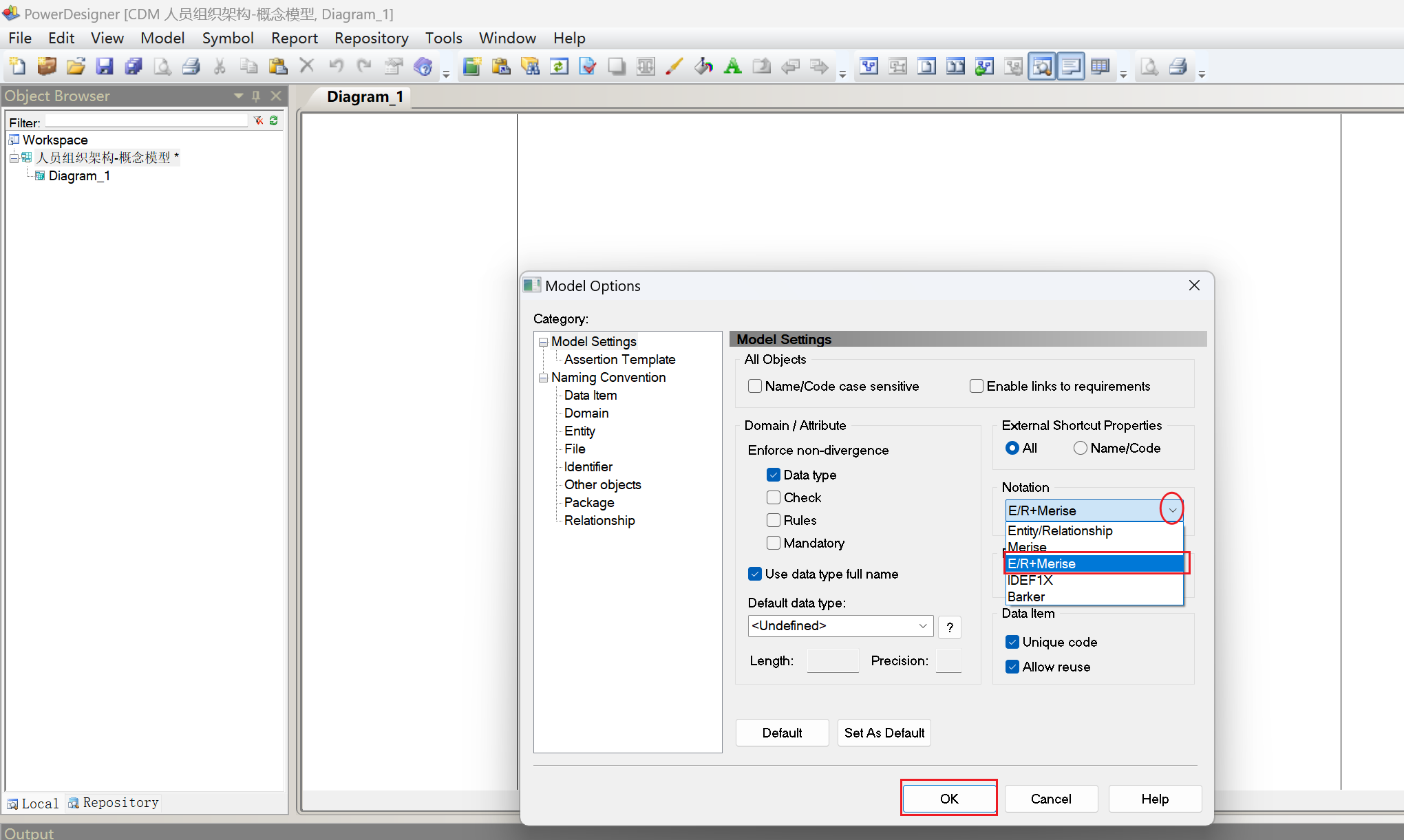
**3、画关系**

* 一个机构可以有多个部门；——》一对多
* 一个部门下可以有多个人员；——》一对多
* 一个角色可以有多个权限；一个权限可以被多个角色使用 ——》多对多
* 一个人可以有多个角色；一个角色可以被多个人使用 ——》多对多

**5.4、pd画图**

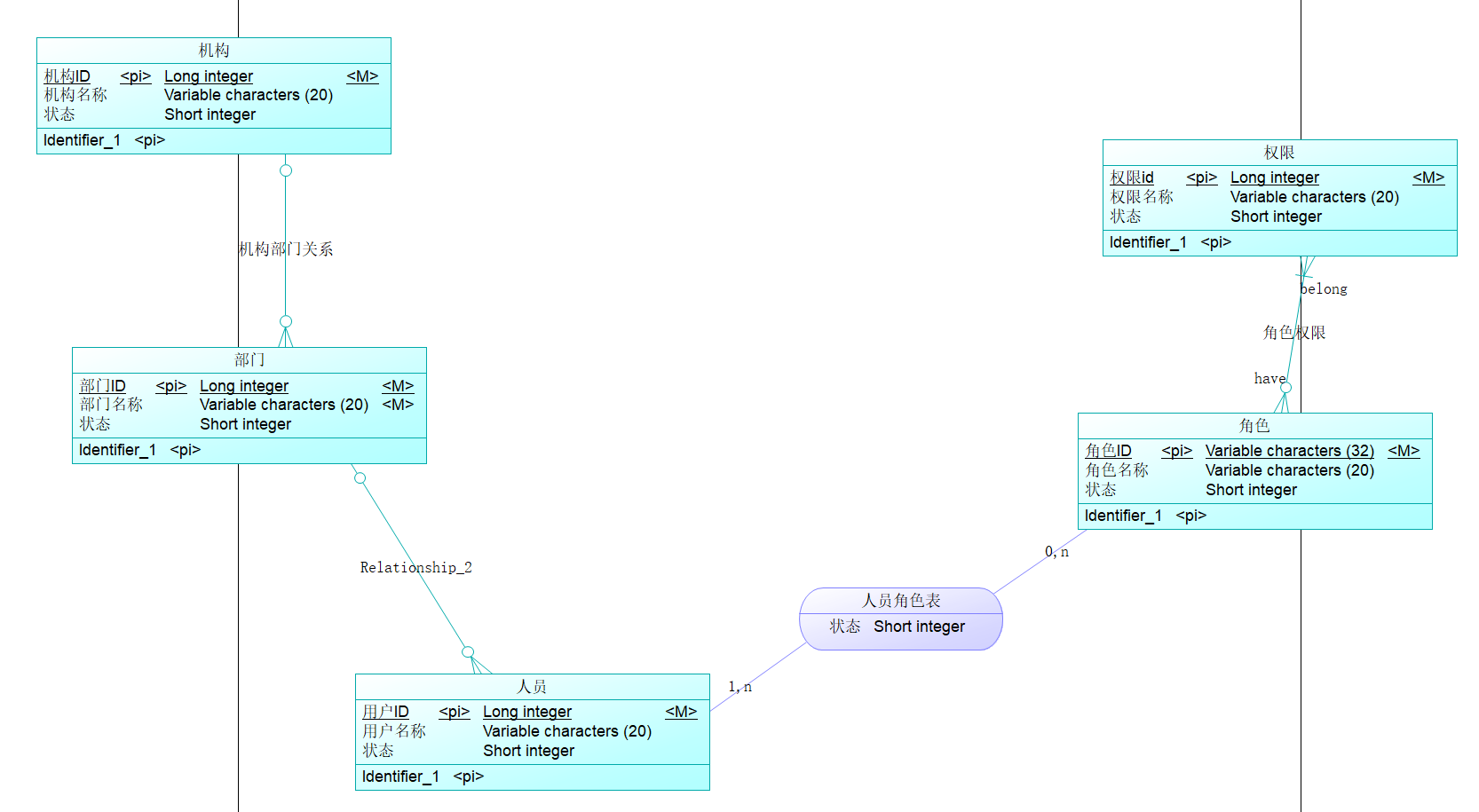
**5.4.1、pd设置**



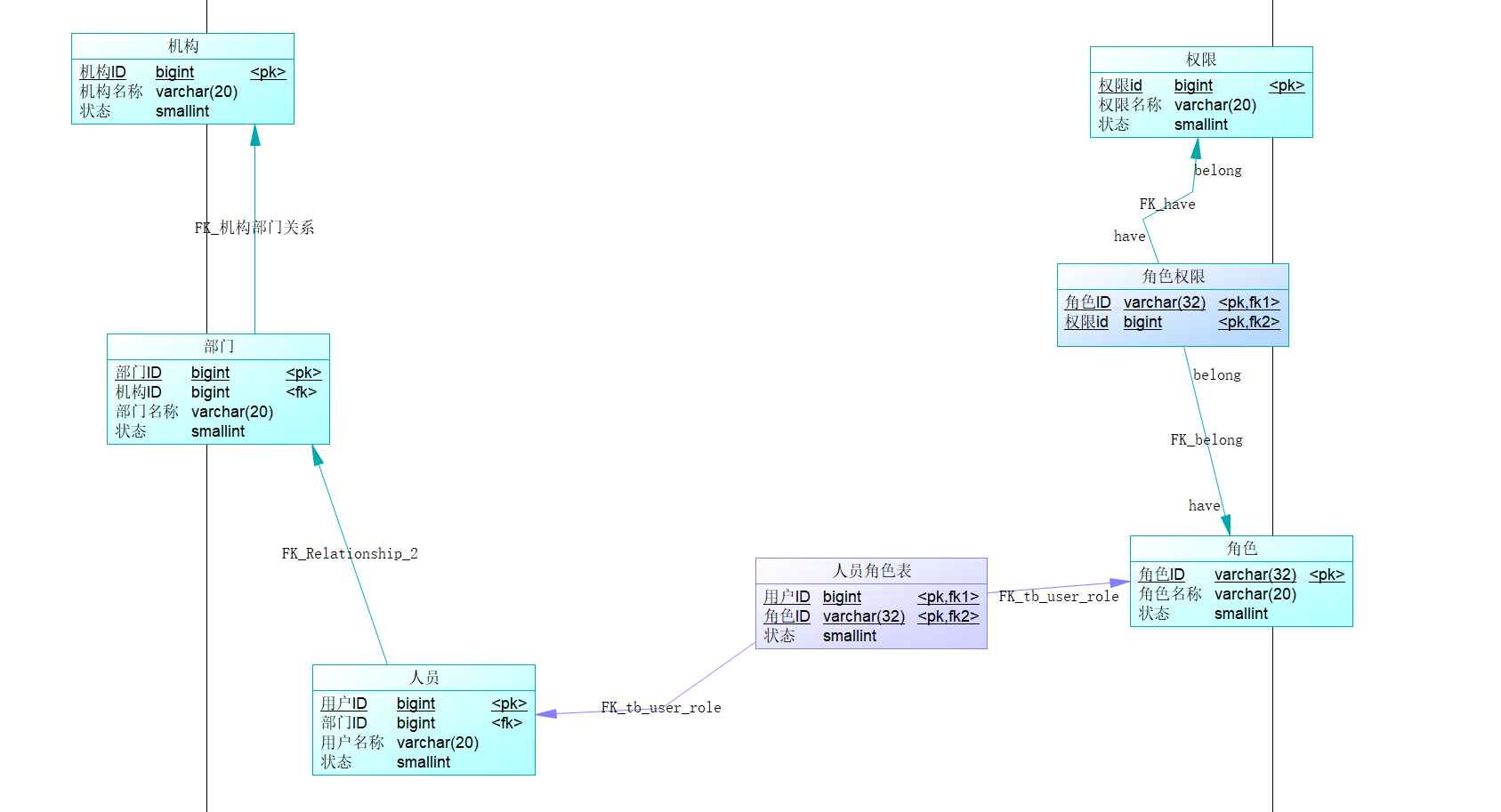


**5.4.2、模型**

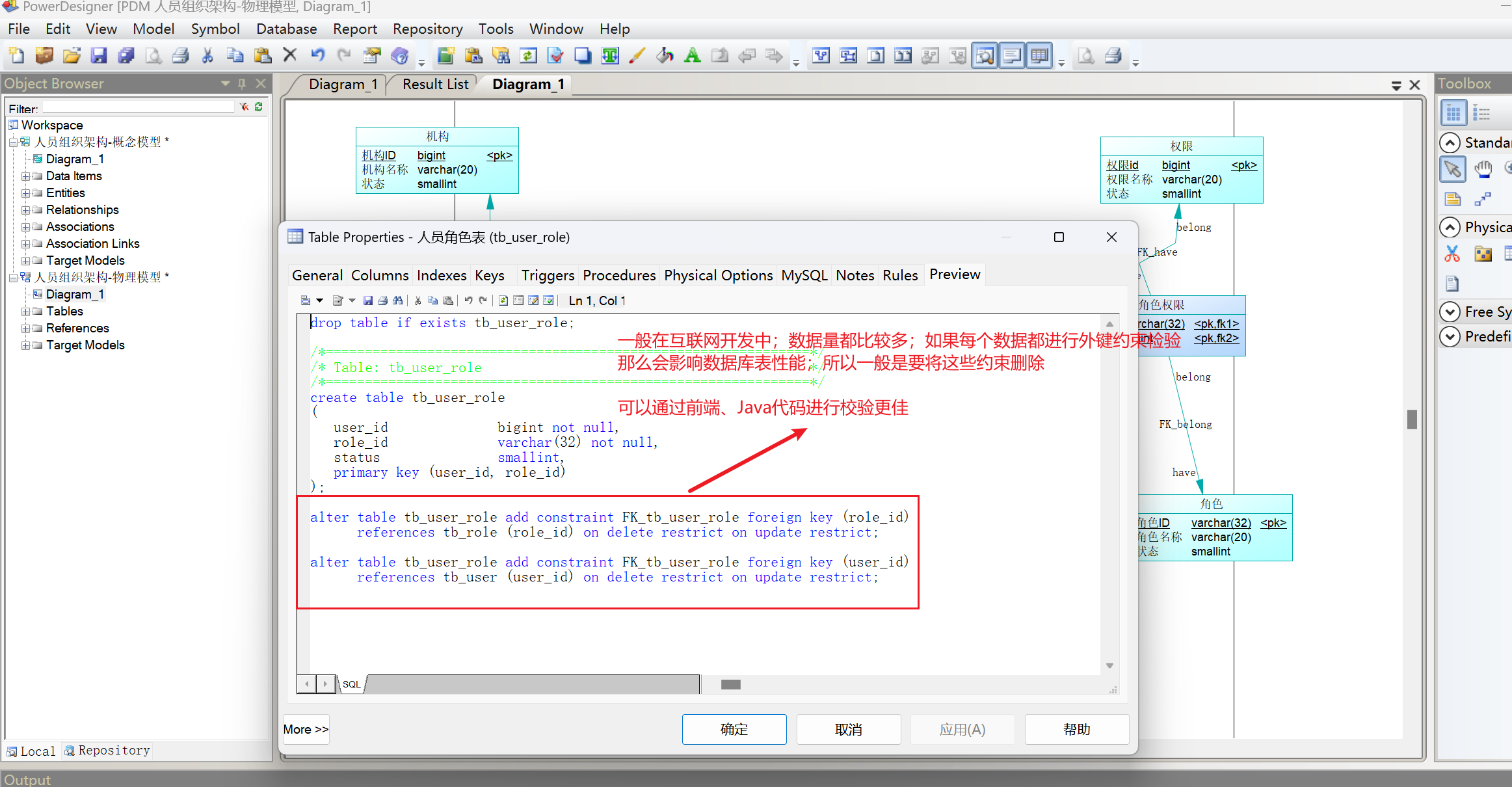
概念模型



物理模型



**5.5、误区**



|  |
| --- |
| **思考**：数据库表范式  并不是范式越高性能越好的；而且一般反范式反而数据库表查询性能好 |

**6、作业**

访问 http://sf.mgt.itheima.net/#/base/type

分析：筛选、保障项、系数、分类这四类数据的关系。

保险-基础数据。通过基础数据功能的学习我们主要解决下列问题：

* 什么是基础数据，包含哪些，分表有什么作用？
* 基础数据的表结构如何设计进行设计？
* 分类项与保障项、系数项、筛选项有什么关系？



上图是用户端功能截图，从中我们提炼并定义下列4个名词：

* **保障项：**通常是指在购买保险时所包含的具体风险或事件，以及在这些风险或事件发生时，保险公司愿意提供的经济赔偿或保障。不同类型的保险产品涵盖不同的保障项
* **系数项：**通常是指一系列用于计算保险费率的因子或系数。这些因子可以包括被保险人的年龄、性别、健康状况、驾驶记录、保险类型、地理位置等，它们会影响最终的保险费率计算。
* **筛选项：**通常是指一系列因素和选项，需要根据你的具体需求和情况来考虑，来选择自己需要的保险服务。
* **分类项：**通常是指保险根据不同的标准和覆盖范围进行的分类，不同的分类涵盖的方面不同，此处我们主要讨论下列几类保险：医疗、重疾、意外、养老金、旅游等分类

|  |
| --- |
| Tips:  结合app端的界面分析；从分类开始分析起 |

**7、午间问题**

* 你们与保险公司有什么关系
* 为什么到你们平台来购买保险？直接到保险公司对应的app上买不是更好吗？
* 你们常卖的保险有哪些类型
* 描述一下系统的技术架构
* 说一下你们开发团队情况
* 介绍一下你们项目的集群部署情况？有几个微服务；你开发的话要启动几个微服务