

**分布式人口管理系统分析与设计报告**

|  |  |
| --- | --- |
| **学号/姓名** | 2017200612/陶杰 |
| **院系/专业** | 信息科学与技术/计算机科学与技术 |
| **日 期** | 2017-12-12 |

1. **背景介绍**

**1.1问题由来**

随着我国人口的不断增长，对于人口数据应当釆用快捷的数据联网化的管理。人口管理系统不仅可以整合人口数据资源，还能够实现人口数据的统一管理。人口管理系统的建设也成为人口管理建设的重要工作。但是随着数据量的增大和跨省市的传输数据需求，人口管理系统的成本非常高，如何构建一个成本低廉的人口管理系统是目前急切需要解决的问题。

在互联网、物联网、云计算等浪潮的推动下，业务系统的集成整合、云端迁移成为了信息行业的新的发展趋势，促使很多业务系统也逐步迈入了大数据时代。城市数据、企业数据、医疗数据、网站数据等规模越来越大，计算也越来越复杂、性能、可靠性、可扩展性的需求也越来越强烈，单机版的集中式的数据库已经满足不了大数据存储与处理的需求。如何解决海量数据的存储和访问成为软件系统数据库设计和使用的瓶颈环节。

随着系统规模的扩大，人口管理系统的数据具有了典型的大数据特征，国家安全部门现有的系统多采用MySQL、SQLServer等传统数据库，扩展业务系统的容量而不改变原有业务系统的存储与处理方式是目前大数据时代必须要解决的问题。

由于基于浏览器/服务器模式的系统体系架构是互联网时代的管理信息系统开发的主流方向。本文的主要内容是基于B/S结构的分布式人口管理系统的分析、设计和实现，以有效地管理大量人口信息，实现了对人口的管理和监控。

**1.2技术状况**

分布式数据库系统是数据库技术之一，还有其他多种数据库技术，但是该技术的产生，可以大大的提高使用系统的稳定性以及可用性等多个方面，还可以使得组织上、地理上等相关的多个单位实现信息共享，数据共享。分布式数据库系统有两种形式：第一种就是物理的分布逻辑集成，该形式可以理解为在物理上呈现分布形式，而在逻辑上则是一个整体的统一，这种形式的数据库系统大多适用于用途单一的、专业性较强的部门以及中小企业；第二种则是在物理上还是逻辑上都是呈现分布形式，简称联邦式，这种形式的数据库系统主要用于大量数据的集成，所以这样的数据库系统主要由明显的数据库差距以及用途的不同来组成的。分布式数据库的逻辑集中性主要体现在不管用户在哪个地理位置或者是使用了本局域网里的哪一台电脑，而物理分布性主要体现在数据库中的数据分别储存在不同的电脑上或者是不同的地域里，都可以通过应用程序来对数据进行有效的操作，但是存在着缺陷，用户不会知道这些数据库的具体分布位置，也就是说相同的数据储存在本机里，而这些数据是由本机的数据库管理系统来进行管理操作的。

虽然在云计算时代，传统数据库存在着先天性的弊端，但是现有的大数据领域的数据库又无法将其完全替代，如何规避传统数据库的缺点是目前大数据时代必须要解决的问题。现有的大型数据库的解决方案，如SQLServerCluster或者OracleRAC等，其建设代价较为昂贵，少则几十万，大则上百万乃至更多。在系统采用MySQL数据库的情况下，本文提出以MyCat搭建云服务平台的基础设施，利用MyCat对人口管理系统数据库进行设计与实现。该方案能够低成本地将现有的单机数据库和应用平滑迁移到“云”端，解决系统海量数据存储和业务规模迅速增长情况下的数据瓶颈问题。利用MyCat对系统的数据库进行设计与实现，不仅能扩展了系统的容量，而且能提高了系统的数据处理速度。

随着Internet和Intranet/Extranet的快速增长，Web已经对商业、工业、银行、财政、教育、政府和娱乐及人们的工作生活产生了深远的影响。许多传统的信息和数据库系统正在被移植到互联网上，电子商务迅速增长，早已超出了国界。Web的流行和无所不在，是因为它能提供支持所有类型内容连接的信息发布，容易为最终用户存取。同时，Internet的飞速发展给4GL编写的GUI程序提出了新的挑战。这些程序大都基于C/S结构，这种模式在局域网中将应用一分为二，服务器负责数据管理，客户机完成与用户的交互任务。B/S结构具有如下优点。

1. 操作使用简单，只需安装通用的浏览器；
2. 维护和升级方式简单，客户端零维护，只要管理服务器；
3. 具有分布性特点，可以随时随地处理业务。

B/S结构的软件有着C/S结构软件无法比拟的优势，同时也说明了C/S结构将逐渐被B/S结构取代。

**1.3系统的应用前景**

随着科技与网络的不断发展与更新，计算机和网络技术得到了很好的发展，目前发展与研究的热流是分布式系统，而该系统的管理维护数据中心则是数据库。现今的数据库技术与传统的数据库截然不同，分布式数据库能满足传统数据库达不到的要求，分布式数据库有着维护数据性、数据共享性以及实时性等多个方面的满足需求，而且还可以达到更高的需求。本文提出的分布式人口管理系统以MySQL作为数据库，使用MyCat中间件实现分布式部署，系统后台采用Java EE架构，保证系统的稳定性，系统前端采用Bootstrap框架和Echarts组件进行展示，具有丰富的人机交互界面，确保了系统的可操作性和易用性。本系统可以为人口管理工作减轻负担，维护人员不必担心系统数据量的增加导致系统的可用性降低，管理人员可通过可视化界面简易地操作数据。随着技术和网络的发展，本系统必将为人口管理工作作出巨大贡献。

1. **基本术语定义**

本文基本术语定义如表2-1所示。

**表2-1 基本术语定义表**

|  |  |
| --- | --- |
| **术语** | **定义** |
| 分布式数据库系统 | 包含分布式数据库管理系统和分布式数据库。 |
| MySQL | 是一个关系型数据库管理系统，目前术语Oracle旗下产品。 |
| MyCat | 是由中国数名具有多年丰富经验的DBA和架构师在阿里巴巴的Cobar基础上志愿二次开发的数据库中间件。 |
| B/S | 是Web兴起后的一种网络结构模式，浏览器是客户端最主要的应用软件。 |
| Java EE | 是Sun（2009年4月20日被甲骨文公司收购）公司推出的企业级应用程序版本。 |
| Spring Boot | 是由Pivotal团队提供的全新框架，目的是简化新Spring应用的初始搭建以及开发过程。 |
| MyBatis | 是一个基于Java的持久层框架。 |
| Bootstrap | 来自Twitter，是目前很受欢迎的前端框架，基于HTML、CSS和JavaScript，它简介灵活，使得Web开发更加快捷。 |
| Echarts | 一个纯JavaScript的图表库，可以流畅地运行在PC和移动设备上，兼容当前绝大部分浏览器（IE8/9/10，Chrome，Firefox，Safari等），底层依赖轻量级的Canvas库类ZRender，提供直观、生动、可交互、可高度个性化定制的数据可视化图表。 |

1. **系统总体目标与应用范围**

**3.1 系统目标**

经过对分布式人口管理系统的初步分析，结合开发成本、系统可用性以及现有技术状况几个方面，系统目标主要包括以下几个方面。

1. 充分体现分布式数据库设计思想；
2. 数据量达到千万级别；
3. 良好的人机交互界面；
4. 具有分类查询功能，查询结果合理化展示；
5. 使用合理的算法或模型进行人口增长、住房需求的预测。

**3.2 应用行业**

人口管理及统计分析

**3.3 服务对象**

人口管理及统计分析相关人员

1. **需求捕获**

**4.1 系统角色**

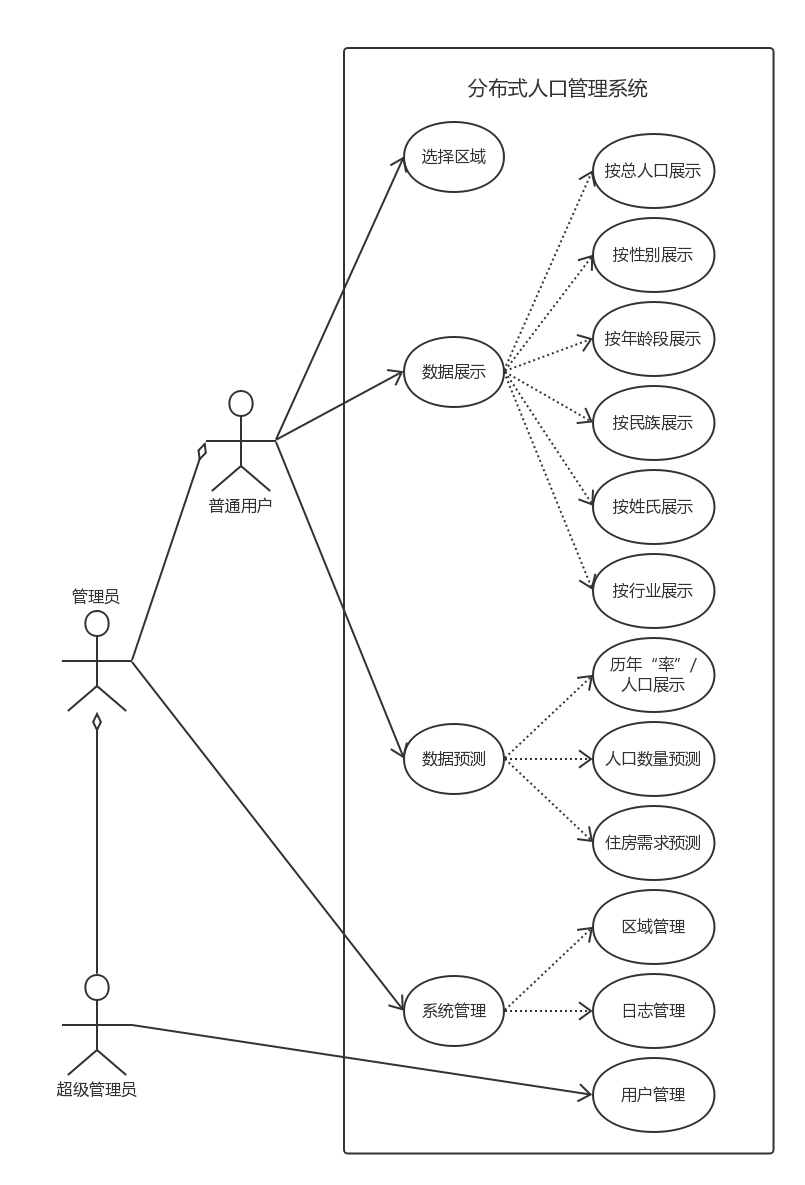
经过对分布式人口管理系统的需求分析，结合系统的可行性，规定表4-1所示系统角色。

**表4-1 系统角色表**

|  |  |
| --- | --- |
| **角色（参与者）** | **角色描述** |
| 用户 | 普通用户可通过前端界面浏览人口统计数据和预测数据等 |
| 管理员 | 管理员在普通用户的基础上增加对区域、日志的管理 |
| 超级管理员 | 超级管理员在管理员的基础上增加对用户的管理 |

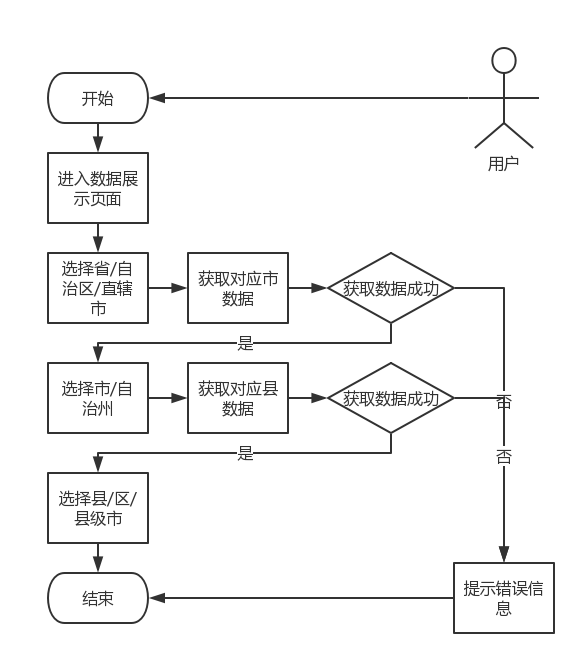
**4.2 系统整体用例图**

系统整体用例图如图4-1所示。



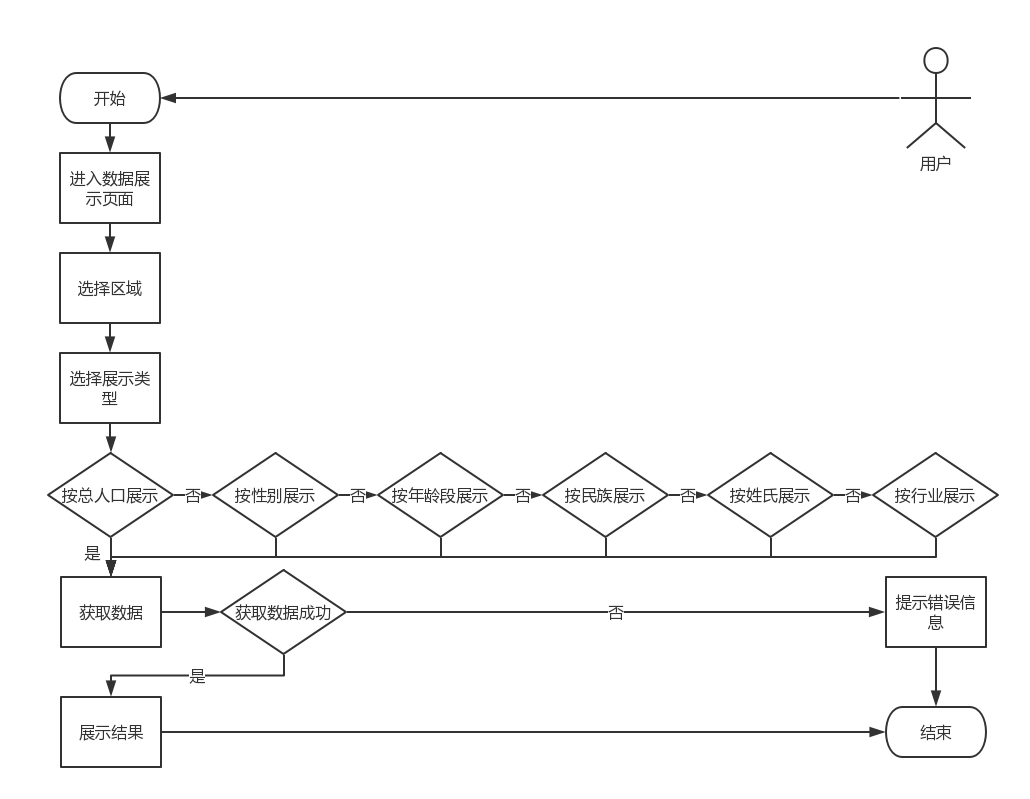
**图4-1 系统整体用例图**

**4.3 选择地区功能流程图**

****

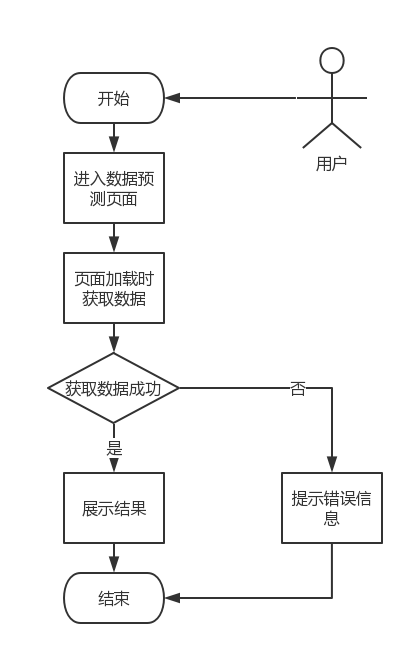
**图4-2 选择地区功能流程图**

**4.4 数据展示功能流程图**

****

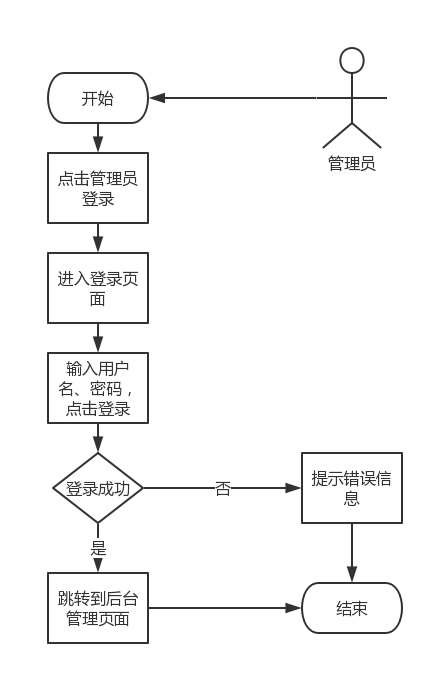
**图4-3 数据展示功能流程图**

**4.5 数据预测功能流程图**

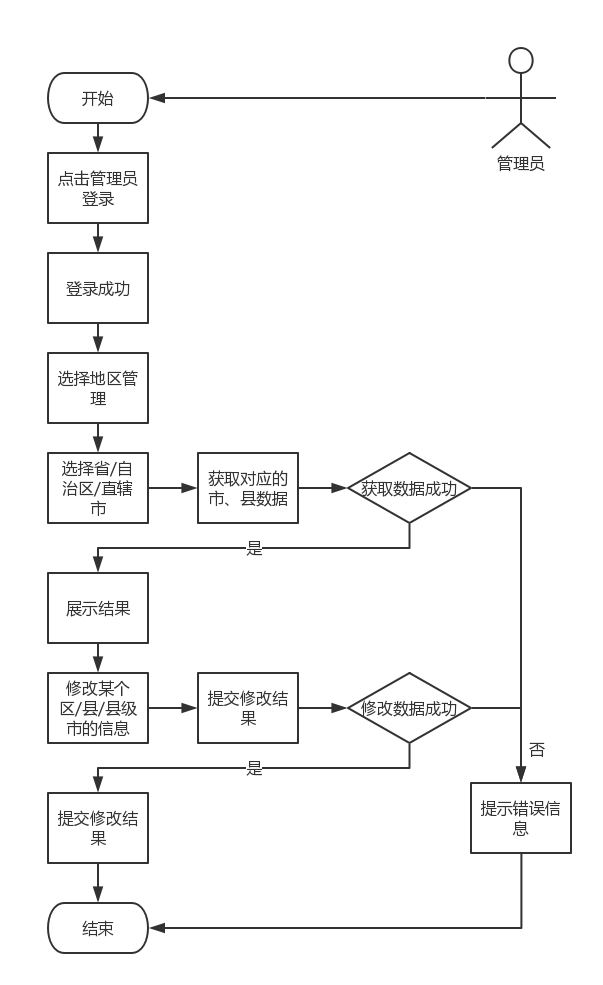
****

**图4-4 数据预测功能流程图**

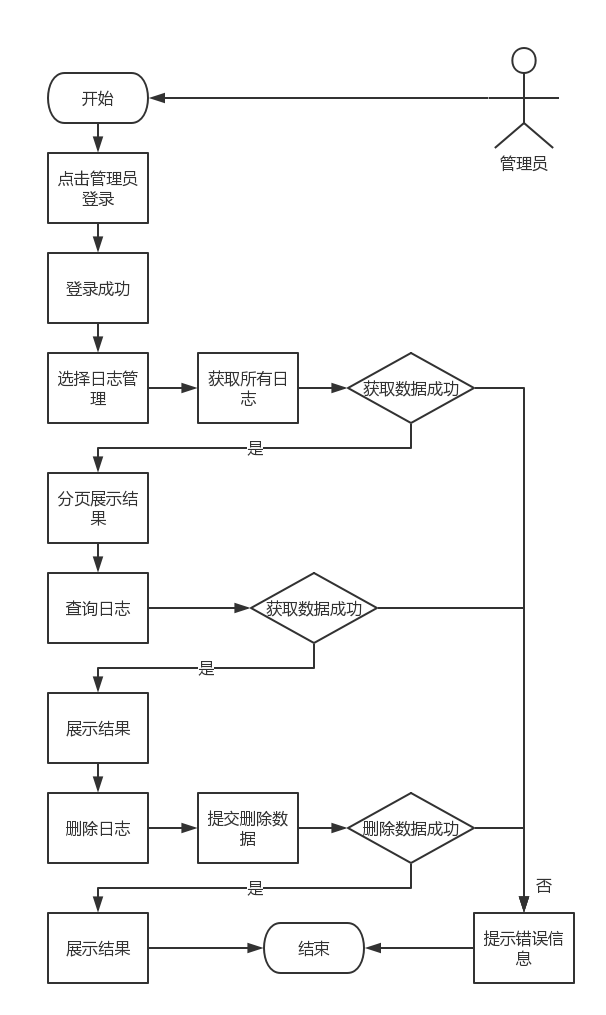
**4.6 系统管理功能流程图**

****

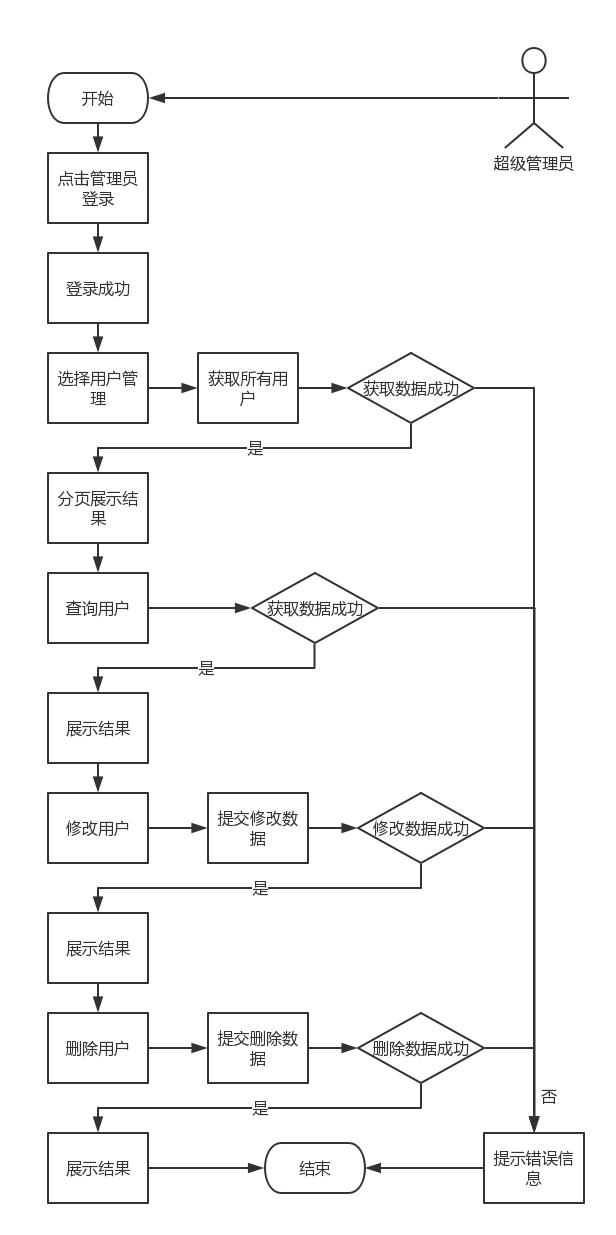
**图4-5 登录流程图**



**图4-6 地区管理流程图**



**图4-7 日志管理流程图**



**图4-8 用户管理流程图**

**4.7 详细用例分析**

**表4-2 选择地区用例分析表**

|  |  |
| --- | --- |
| **用例名称** | 选择地区用例 |
| **用例编号** | 001 |
| **系统角色** | 用户 |
| **简要说明** | 选择地区为展示做准备 |
| **前置条件** | 用户点击数据展示 |
| **后置条件** | 地区信息正确获取 |
| **流程图** | 图4-2 |
| **使用用例频率** | 高 |

**表4-3 数据展示用例分析表**

|  |  |
| --- | --- |
| **用例名称** | 数据展示用例 |
| **用例编号** | 002 |
| **系统角色** | 用户 |
| **简要说明** | 选择按不同的方式进行查询结果的展示 |
| **前置条件** | 选择地区 |
| **后置条件** | 查询结果正确展示 |
| **流程图** | 图4-3 |
| **使用用例频率** | 高 |

**表4-4 数据预测用例分析表**

|  |  |
| --- | --- |
| **用例名称** | 数据预测用例 |
| **用例编号** | 003 |
| **系统角色** | 用户 |
| **简要说明** | 对历年人口的总量、出生率、死亡率和自然增长率进行展示，并且展示从现在到2036年的人口总量及住房需求变化情况 |
| **前置条件** | 用户点击数据预测 |
| **后置条件** | 结果正确展示 |
| **流程图** | 图4-4 |
| **使用用例频率** | 中 |

**表4-5 登录用例分析表**

|  |  |
| --- | --- |
| **用例名称** | 登录用例 |
| **用例编号** | 004 |
| **系统角色** | 管理员 |
| **简要说明** | 管理员登录成功之后进入后台系统管理页面 |
| **前置条件** | 管理员点击管理员登录 |
| **后置条件** | 登录成功进入后台系统管理页面 |
| **流程图** | 图4-5 |
| **使用用例频率** | 低 |

**表4-6 地区管理用例分析表**

|  |  |
| --- | --- |
| **用例名称** | 地区管理用例 |
| **用例编号** | 005 |
| **系统角色** | 管理员 |
| **简要说明** | 对地区的名称、地区码和所属上一级地区进行修改 |
| **前置条件** | 管理员登录成功 |
| **后置条件** | 正确修改地区信息 |
| **流程图** | 图4-6 |
| **使用用例频率** | 低 |

**表4-7 日志管理用例分析表**

|  |  |
| --- | --- |
| **用例名称** | 日志管理用例 |
| **用例编号** | 006 |
| **系统角色** | 管理员 |
| **简要说明** | 日志主要记录用户访问系统的所有请求，管理员可对日志进行查询，删除操作 |
| **前置条件** | 管理员登录成功 |
| **后置条件** | 正确展示操作结果 |
| **流程图** | 图4-7 |
| **使用用例频率** | 低 |

**表4-8 用户管理用例分析表**

|  |  |
| --- | --- |
| **用例名称** | 用户管理用例 |
| **用例编号** | 007 |
| **系统角色** | 超级管理员 |
| **简要说明** | 超级管理员可对管理员信息进行操作 |
| **前置条件** | 超级管理员登录成功 |
| **后置条件** | 正确展示操作结果 |
| **流程图** | 图4-8 |
| **使用用例频率** | 低 |

1. **数据库概要设计**

**5.1 E-R图**

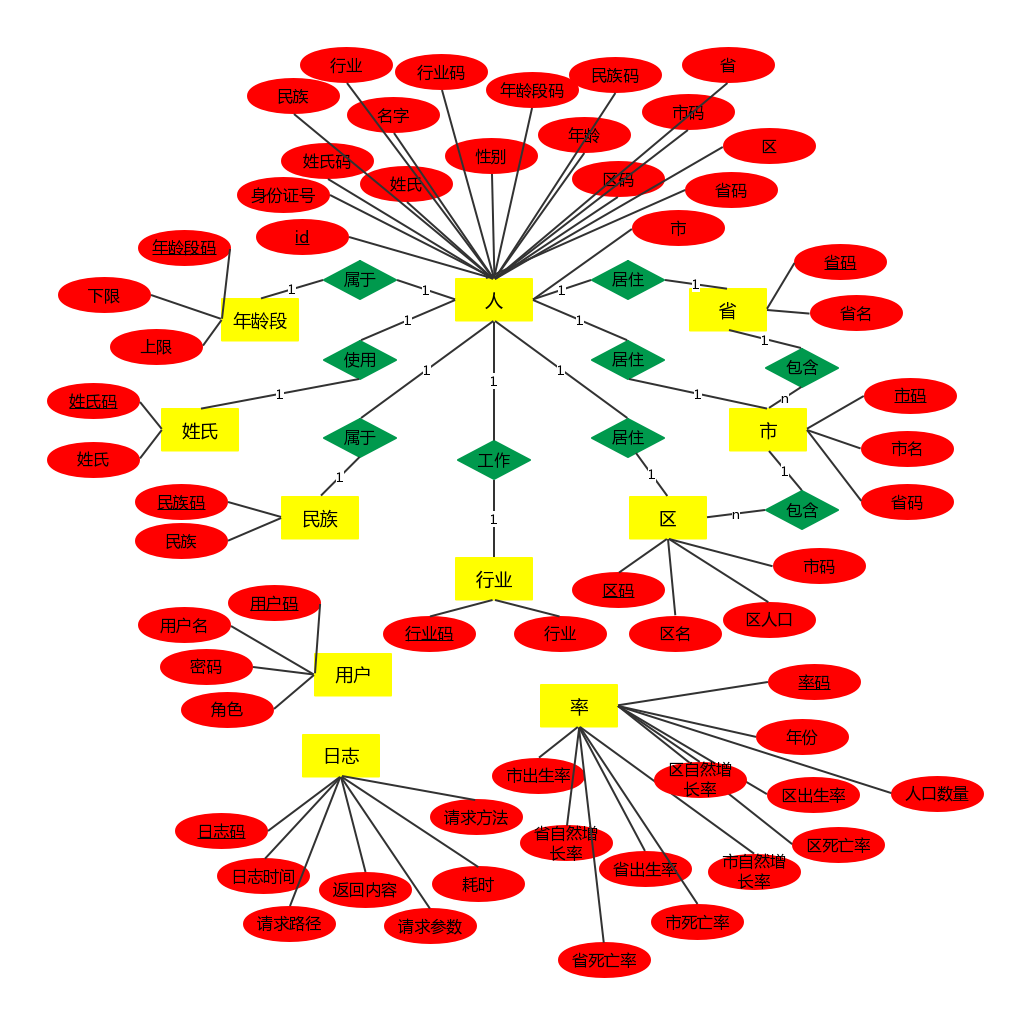


图5-1 E-R图

**5.2 文字说明**

如图5-1所示，本系统的实体如黄色矩形所示：人、省、市、区、行业、民族、姓氏、年龄段、用户、日志、率。实体之间的关系如绿色菱形所示：人与省之间是居住关系（1:1）、人与市之间是居住关系（1:1）、人与区之间是居住关系（1:1）、人与行业之间是工作关系（1:1）、人与民族之间是属于关系（1:1）、人与姓氏之间是使用关系（1:1）、人与年龄段之间是属于关系（1:1）、省与市之间是包含关系（1:n）、市与区之间是包含关系（1:n）。实体的属性如图中红色椭圆形所示。

1. **数据库初始表结构设计**

**6.1 实体基本属性**

由实体（对象）关系图中获取的语义信息得，实体（对象）的基本属性如下：

人（人编号，身份证号码，姓氏码，姓氏，名，性别，年龄段码，年龄，民族码，民族，行业码，行业，省码，省，市码，市，区码，区）

省（省码，省）

市（市码，市，省码）

区（区码，市，市码，人口数量）

行业（行业码，行业）

民族（民族码，民族）

姓氏（姓氏码，姓氏）

年龄段（年龄段码，年龄段下限，年龄段上限）

用户（用户码，用户名，密码，角色）

日志（日志码，日志日期时间，请求路径，请求方法，请求参数，返回内容，耗时）

率（率码，年份，人口数量，出生率-省，死亡率-省，自然增长率-省，出生率-市，死亡率-市，自然增长率-市，出生率-区，死亡率-区，自然增长率-区）

其中直线下划线标注的属性表示的是主键，波浪下划线标注的属性表示的是外键。

**6.2 实体与联系的基本信息**

由实体（对象）关系图中获取的语义信息得，实体（对象）与联系的基本信息如下：

其中人与省之间存在1:1联系：居住（人编号，省码）

其中人与市之间存在1:1联系：居住（人编号，市码）

其中人与区之间存在1:1联系：居住（人编号，区码）

其中人与行业之间存在1:1联系：工作（人编号，行业码）

其中人与民族之间存在1:1联系：属于（人编号，民族码）

其中人与姓氏之间存在1:1联系：使用（人编号，姓氏码）

其中人与年龄段之间存1:1在联系：属于（人编号，年龄段码）

其中省与市之间存在1:n联系：包含（省码，市码）

其中市与区之间存在1:n联系：包含（市码，区码）

1. **数据表设计**

**7.1 人表结构设计**

**表7-1 people2016表结构设计**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **表名** | people2016 | | | | | | | |
| **列名** | **类型** | **长度** | **小数点** | **不是空** | **自动递增** | **主键** | **外键** | **注释** |
| id\_ | int | 11 | 0 | √ |  | √ |  | 主键 |
| id\_number | varchar | 255 | 0 | √ |  |  |  | 身份证号码 |
| last\_name\_code | int | 11 | 0 | √ |  |  | √ | 姓氏码 |
| last\_name | varchar | 255 | 0 | √ |  |  |  | 姓氏 |
| first\_name | varchar | 255 | 0 | √ |  |  |  | 名 |
| gender\_ | tinyint | 1 | 0 | √ |  |  |  | 性别 |
| age\_range\_code | int | 11 | 0 | √ |  |  | √ | 年龄段码 |
| age\_ | int | 11 | 0 | √ |  |  |  | 年龄 |
| nation\_code | int | 11 | 0 | √ |  |  | √ | 民族码 |
| nation\_ | varchar | 255 | 0 | √ |  |  |  | 民族 |
| province\_code | int | 11 | 0 | √ |  |  | √ | 省码 |
| province\_ | varchar | 255 | 0 | √ |  |  |  | 省 |
| city\_code | int | 11 | 0 | √ |  |  | √ | 市码 |
| city\_ | varchar | 255 | 0 | √ |  |  |  | 市 |
| region\_code | int | 11 | 0 | √ |  |  | √ | 区码 |
| region\_ | varchar | 255 | 0 | √ |  |  |  | 区 |
| profession\_code | int | 11 | 0 | √ |  |  | √ | 行业码 |
| profession\_ | varchar | 255 | 0 | √ |  |  |  | 行业 |

**7.2 省表结构设计**

**表7-2 province表结构设计**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **表名** | province | | | | | | | |
| **列名** | **类型** | **长度** | **小数点** | **不是空** | **自动递增** | **主键** | **外键** | **注释** |
| province\_code | int | 11 | 0 | √ |  | √ |  | 省码 |
| province\_ | varchar | 255 | 0 | √ |  |  |  | 省名 |

**7.3 市表结构设计**

**表7-3 city表结构设计**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **表名** | city | | | | | | | |
| **列名** | **类型** | **长度** | **小数点** | **不是空** | **自动递增** | **主键** | **外键** | **注释** |
| city\_code | int | 11 | 0 | √ |  | √ |  | 市码 |
| city\_ | varchar | 255 | 0 | √ |  |  |  | 市名 |
| province\_code | int | 11 | 0 | √ |  |  | √ | 省码 |

**7.4 区表结构设计**

**表7-4 region表结构设计**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **表名** | region | | | | | | | |
| **列名** | **类型** | **长度** | **小数点** | **不是空** | **自动递增** | **主键** | **外键** | **注释** |
| region\_code | int | 11 | 0 | √ |  | √ |  | 市码 |
| region\_ | varchar | 255 | 0 | √ |  |  |  | 市名 |
| city\_code | int | 11 | 0 | √ |  |  | √ | 市码 |
| region\_people\_number | int | 11 | 0 | √ |  |  |  | 人口数量 |

**7.5 行业表结构设计**

**表7-5 profession表结构设计**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **表名** | profession | | | | | | | |
| **列名** | **类型** | **长度** | **小数点** | **不是空** | **自动递增** | **主键** | **外键** | **注释** |
| profession\_code | int | 11 | 0 | √ |  | √ |  | 行业码 |
| profession\_ | varchar | 255 | 0 | √ |  |  |  | 行业名 |

**7.6 民族表结构设计**

**表7-6 nation表结构设计**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **表名** | nation | | | | | | | |
| **列名** | **类型** | **长度** | **小数点** | **不是空** | **自动递增** | **主键** | **外键** | **注释** |
| nation\_code | int | 11 | 0 | √ |  | √ | √ | 民族码 |
| nation\_ | varchar | 255 | 0 | √ |  |  |  | 民族 |

**7.7 姓氏表结构设计**

**表7-7 last\_name表结构设计**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **表名** | last\_name | | | | | | | |
| **列名** | **类型** | **长度** | **小数点** | **不是空** | **自动递增** | **主键** | **外键** | **注释** |
| last\_name\_code | int | 11 | 0 | √ |  | √ |  | 姓氏码 |
| last\_name | varchar | 255 | 0 | √ |  |  |  | 姓氏 |

**7.8 年龄段表结构设计**

**表7-8 age\_range表结构设计**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **表名** | age\_range | | | | | | | |
| **列名** | **类型** | **长度** | **小数点** | **不是空** | **自动递增** | **主键** | **外键** | **注释** |
| age\_range\_code | int | 11 | 0 | √ |  | √ |  | 年龄段码 |
| bottom\_ | int | 11 | 0 | √ |  |  |  | 年龄段下限 |
| top\_ | int | 11 | 0 | √ |  |  |  | 年龄段上限 |

**7.9 用户结构设计**

**表7-9 user表结构设计**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **表名** | user | | | | | | | |
| **列名** | **类型** | **长度** | **小数点** | **不是空** | **自动递增** | **主键** | **外键** | **注释** |
| user\_code | int | 11 | 0 | √ |  | √ |  | 用户码 |
| username\_ | varchar | 255 | 0 | √ |  |  |  | 用户名 |
| password\_ | varchar | 255 | 0 | √ |  |  |  | 密码 |
| role\_ | tinyint | 1 | 0 | √ |  |  |  | 角色 |

**7.10 日志表结构设计**

**表7-10 log表结构设计**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **表名** | log | | | | | | | |
| **列名** | **类型** | **长度** | **小数点** | **不是空** | **自动递增** | **主键** | **外键** | **注释** |
| log\_code | int | 11 | 0 | √ | √ | √ |  | 日志码 |
| log\_date\_time | datetime |  |  | √ |  |  |  | 日志日期和时间 |
| request\_uri | varchar | 255 |  | √ |  |  |  | 请求路径 |
| request\_method | varchar | 255 |  | √ |  |  |  | 请求方法 |
| request\_param | varchar | 255 |  | √ |  |  |  | 请求参数 |
| response\_content | varchar | 255 |  | √ |  |  |  | 返回内容 |
| spend\_time | double | 11 | 2 | √ |  |  |  | 耗时 |

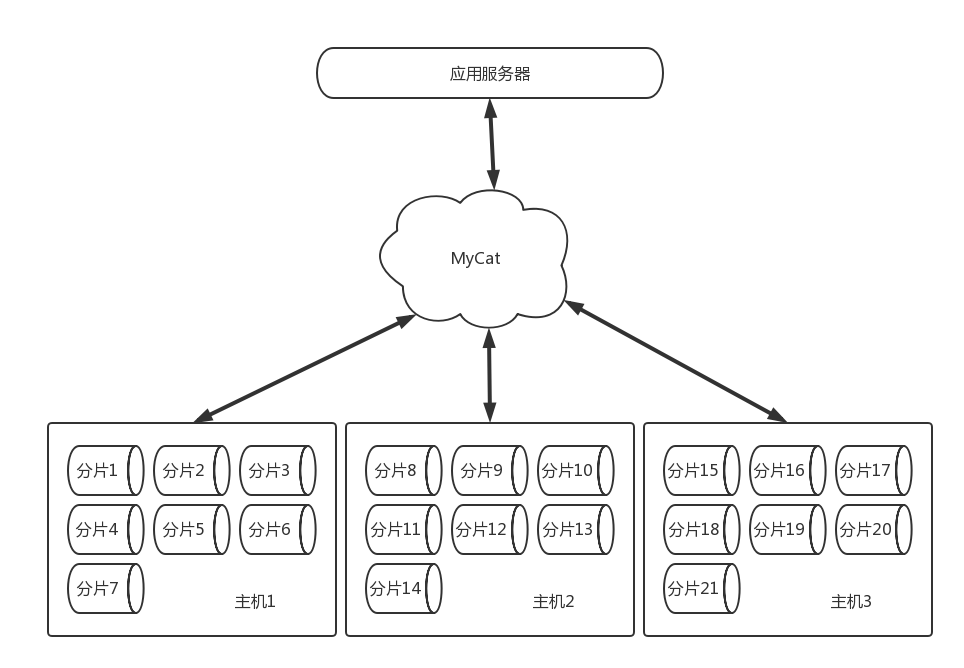
**7.11 率表结构设计**

**表7-11 rate表结构设计**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **表名** | rate | | | | | | | |
| **列名** | **类型** | **长度** | **小数点** | **不是空** | **自动递增** | **主键** | **外键** | **注释** |
| rate\_code | int | 11 | 0 | √ |  | √ |  | 率码 |
| year\_ | int | 11 | 0 | √ |  |  |  | 年份 |
| people\_number | double | 11 | 2 | √ |  |  |  | 人口数量 |
| province\_birth\_rate | double | 11 | 2 | √ |  |  |  | 出生率-省 |
| province\_death\_rate | double | 11 | 2 | √ |  |  |  | 死亡率-省 |
| province\_growth\_rate | double | 11 | 2 | √ |  |  |  | 自然增长率-省 |
| city\_birth\_rate | double | 11 | 2 | √ |  |  |  | 出生率-市 |
| city\_death\_rate | double | 11 | 2 | √ |  |  |  | 死亡率-市 |
| city\_growth\_rate | double | 11 | 2 | √ |  |  |  | 自然增长率-市 |
| region\_birth\_rate | double | 11 | 2 | √ |  |  |  | 出生率-区 |
| region\_death\_rate | double | 11 | 2 | √ |  |  |  | 死亡率-区 |
| region\_growth\_rate | double | 11 | 2 | √ |  |  |  | 自然增长率-区 |

**7.11 分片设计**

分布式人口管理系统预计的数据是四川省2016年的人口,预计数据量达到8000多万，由于四川省包含21个市/自治州，所以预计划分为21个分片，由于物理条件限制，预计划分为3个场地，每个场地包含7个分片。由于市码或区码具有明确的规定，按从小到大排列，所以分片规则是市码或区码的范围。分片结果如图7-1所示。



**图7-1 分片设计图**

**7.12 MyCat配置**

<?xml version="1.0"?>

<!DOCTYPE mycat:schema SYSTEM "schema.dtd">

<mycat:schema xmlns:mycat="http://io.mycat/">

<schema name="DPMS\_DB" checkSQLschema="false">

<table name="people2016" primaryKey="ID" dataNode="chengdu, zigong, panzhihua, luzhou, deyang, mianyang, guangyuan, suining, neijiang, leshan, nanchong, meishan, yibin, guangan, dazhou, yaan, bazhong, ziyang, aba, ganzi, liangshan" rule="auto-sharding-long" />

<table name="age\_range" primaryKey="age\_range\_code" type="global" dataNode="info, chengdu, zigong, panzhihua, luzhou, deyang, mianyang, guangyuan, suining, neijiang, leshan, nanchong, meishan, yibin, guangan, dazhou, yaan, bazhong, ziyang, aba, ganzi, liangshan" />

<table name="profession" primaryKey="profession\_code" dataNode="info" />

<table name="province" primaryKey="province\_code" dataNode="info" />

<table name="city" primaryKey="city\_code" dataNode="info" />

<table name="region" primaryKey="region\_code" dataNode="info" />

<table name="nation" primaryKey="nation\_code" dataNode="info" />

<table name="last\_name" primaryKey="last\_name\_code" dataNode="info" />

<table name="rate" primaryKey="rate\_code" dataNode="info" />

<table name="user" primaryKey="user\_code" dataNode="info" />

<table name="log" primaryKey="log\_code" dataNode="info" />

</schema>

<dataNode name="info" dataHost="127.0.0.1" database="dpms\_info" />

<dataNode name="chengdu" dataHost="127.0.0.1" database="dpms\_chengdu" />

<dataNode name="zigong" dataHost="127.0.0.1" database="dpms\_zigong" />

<dataNode name="panzhihua" dataHost="127.0.0.1" database="dpms\_panzhihua" />

<dataNode name="luzhou" dataHost="127.0.0.1" database="dpms\_luzhou" />

<dataNode name="deyang" dataHost="127.0.0.1" database="dpms\_deyang" />

<dataNode name="mianyang" dataHost="127.0.0.1" database="dpms\_mianyang" />

<dataNode name="guangyuan" dataHost="127.0.0.1" database="dpms\_guangyuan" />

<dataNode name="suining" dataHost="192.168.244.128" database="dpms\_suining" />

<dataNode name="neijiang" dataHost="192.168.244.128" database="dpms\_neijiang" />

<dataNode name="leshan" dataHost="192.168.244.128" database="dpms\_leshan" />

<dataNode name="nanchong" dataHost="192.168.244.128" database="dpms\_nanchong" />

<dataNode name="meishan" dataHost="192.168.244.128" database="dpms\_meishan" />

<dataNode name="yibin" dataHost="192.168.244.128" database="dpms\_yibin" />

<dataNode name="guangan" dataHost="192.168.244.128" database="dpms\_guangan" />

<dataNode name="dazhou" dataHost="192.168.244.129" database="dpms\_dazhou" />

<dataNode name="yaan" dataHost="192.168.244.129" database="dpms\_yaan" />

<dataNode name="bazhong" dataHost="192.168.244.129" database="dpms\_bazhong" />

<dataNode name="ziyang" dataHost="192.168.244.129" database="dpms\_ziyang" />

<dataNode name="aba" dataHost="192.168.244.129" database="dpms\_aba" />

<dataNode name="ganzi" dataHost="192.168.244.129" database="dpms\_ganzi" />

<dataNode name="liangshan" dataHost="192.168.244.129" database="dpms\_liangshan" />

<dataHost name="127.0.0.1" maxCon="1000" minCon="10" balance="0" writeType="0" dbType="mysql" dbDriver="native" switchType="0" slaveThreshold="100">

<heartbeat>select user()</heartbeat>

<writeHost host="127.0.0.1\_host" url="127.0.0.1:3306" user="root" password="yeta" />

</dataHost>

<dataHost name="192.168.244.128" maxCon="1000" minCon="10" balance="0" writeType="0" dbType="mysql" dbDriver="native" switchType="0" slaveThreshold="100">

<heartbeat>select user()</heartbeat>

<writeHost host="192.168.244.128\_host" url="192.168.244.128:3306" user="root" password="yeta" />

</dataHost>

<dataHost name="192.168.244.129" maxCon="1000" minCon="10" balance="0" writeType="0" dbType="mysql" dbDriver="native" switchType="0" slaveThreshold="100">

<heartbeat>select user()</heartbeat>

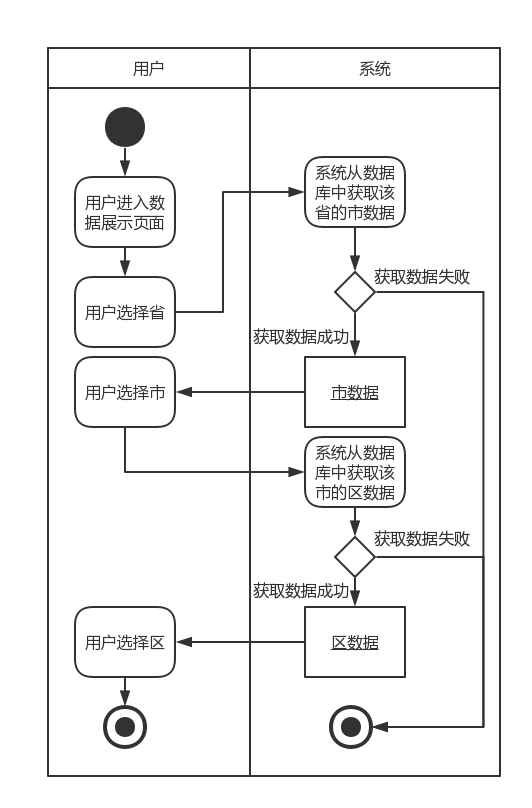
<writeHost host="192.168.244.129\_host" url="192.168.244.129:3306" user="root" password="yeta" />

</dataHost>

</mycat:schema>

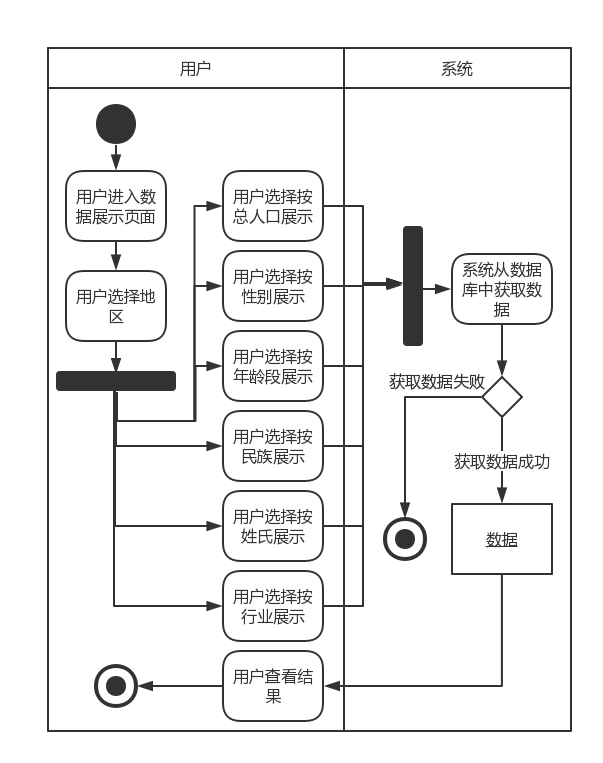
1. **管理事务工作过程描述**

**8.1 选择地区事务活动图**



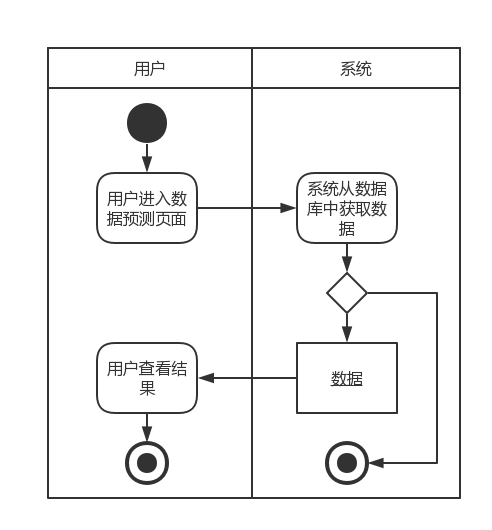
**图8-1 选择地区事务活动图**

**8.2 数据展示事务活动图**

****

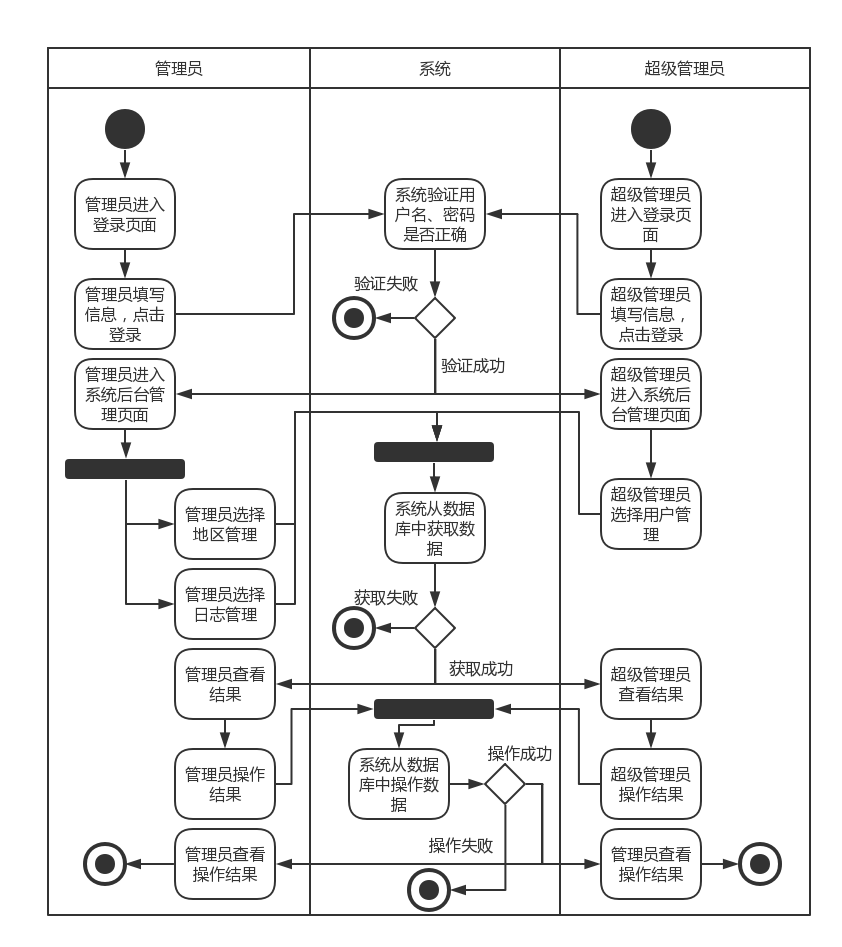
**图8-2 数据展示事务活动图**

**8.3 数据预测事务活动图**

****

**图8-3 数据预测事务活动图**

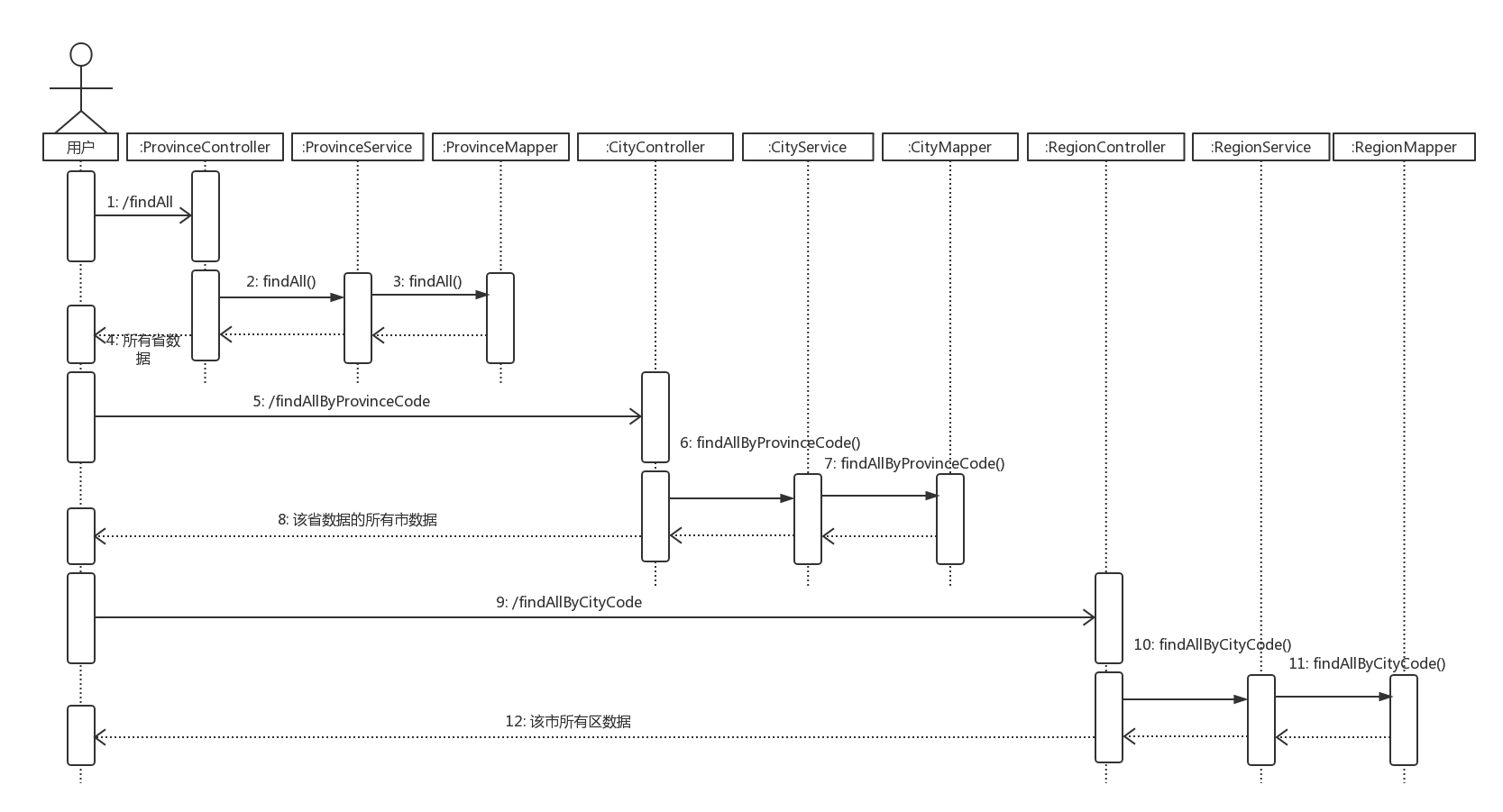
**8.4 系统管理事务活动图**

****

**图8-4 系统管理事务活动图**

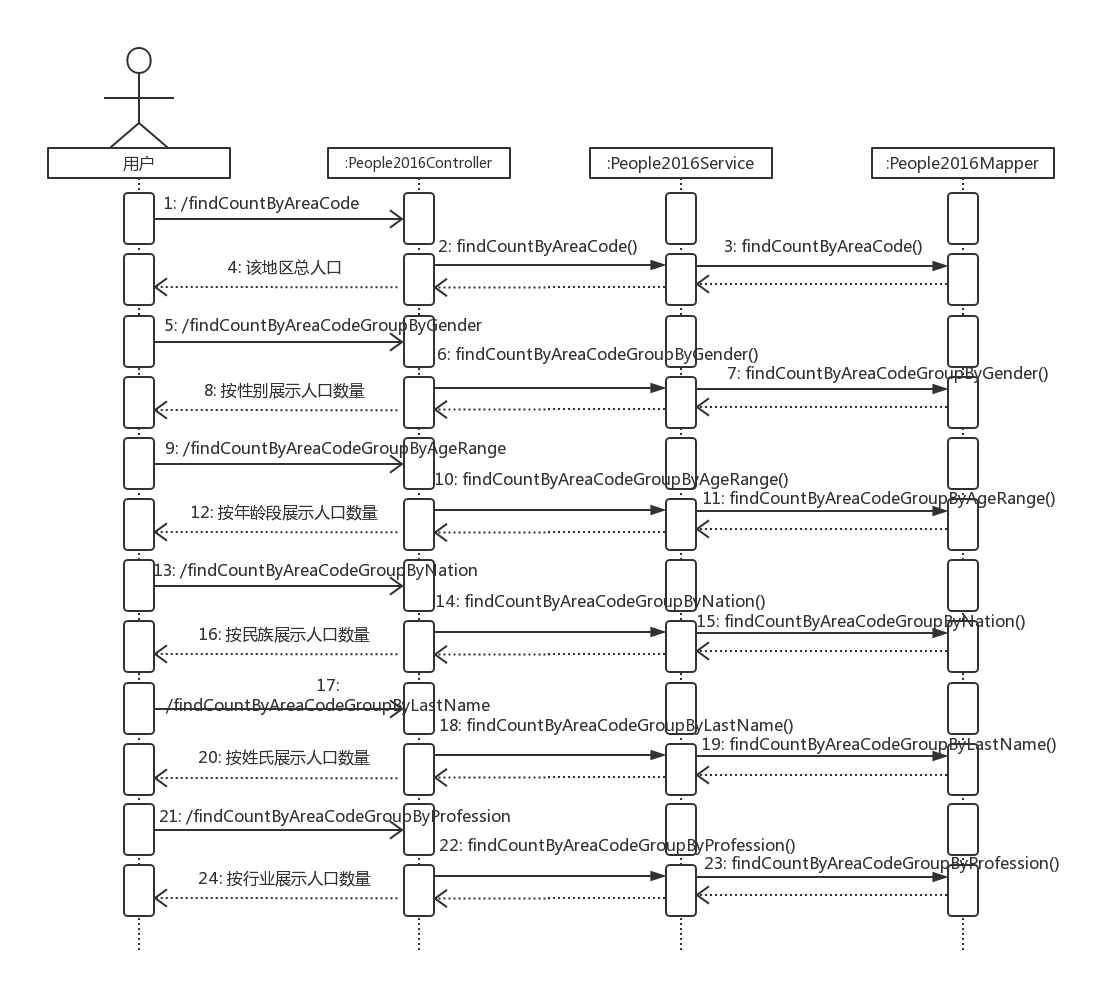
1. **详细设计**

**9.1 地区选择事务时序图**

****

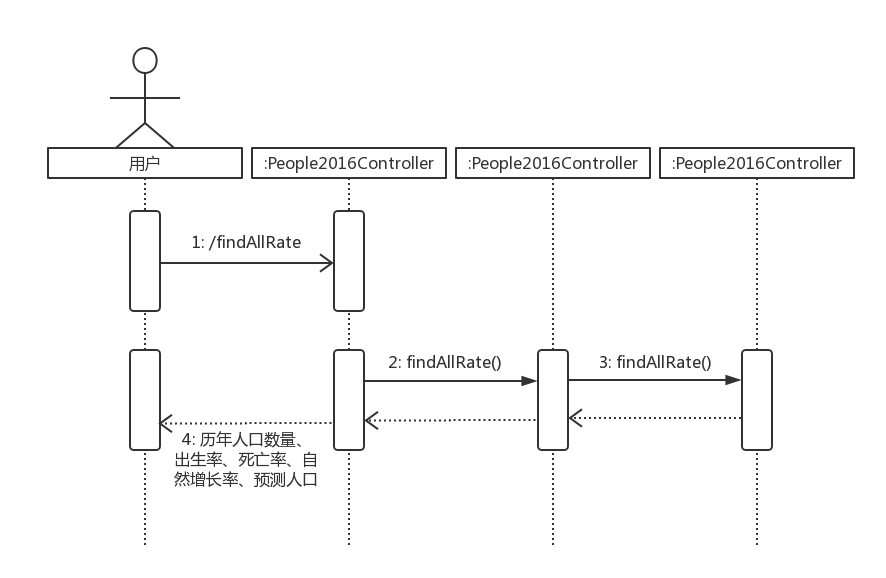
**图9-1 地区选择事务时序图**

**9.2 数据展示事务时序图**

****

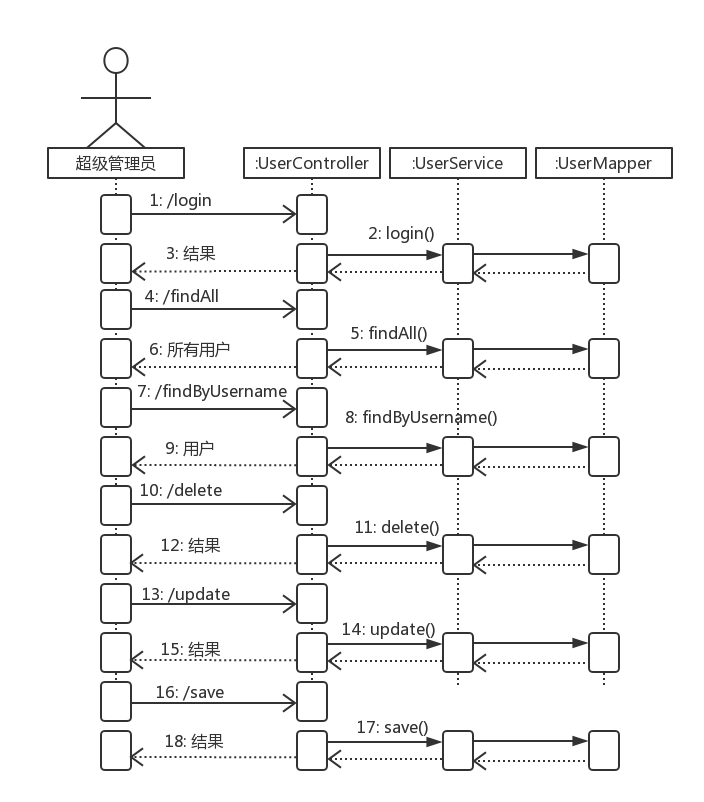
**图9-2 数据展示事务时序图**

**9.3 数据预测事务时序图**

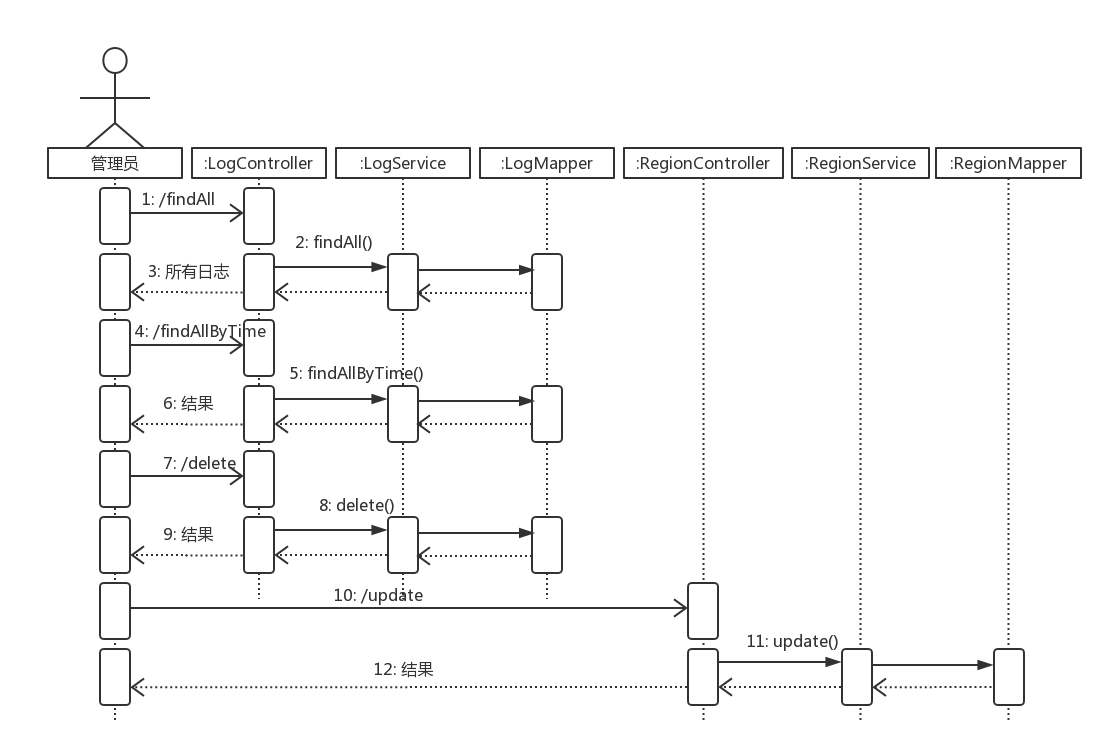
****

**图9-3 数据预测事务时序图**

**9.4 系统管理事务时序图**

****

**图9-4 用户管理事务时序图**

****

**图9-5 日志、地区管理事务时序图**

1. **界面设计与布局**

**10.1 使用技术介绍**

本系统采用B/S结构，界面设计使用到的技术主要有HTML、CSS、JavaScript、jQuery、BootStrap、Echarts等。

HTML也叫做超级文本标记语言，是标准通用标记语言下的一个应用，也是一种规范，一种标准，它通过标记符号来标记要显示的网页中的各个部分。网页文件本身是一种文本文件，通过在文本文件中添加标记符，可以告诉浏览器如何显示其中的内容。浏览器按顺序阅读网页文件，然后根据标记符解释和显示其他标记的内容，对书写错误的标记不指出其错误，且不停止其解释执行过程，编制者只能通过显示效果来分析出错误的原因和出错部位。但需要注意的是，对于不同的浏览器，对同一标记符可能会有不完全相同的解释，因而可能会有不同的显示效果。

CSS也叫层叠样式表，是一种用来表现[HTML](https://baike.baidu.com/item/HTML" \t "https://baike.baidu.com/item/CSS/_blank)或[XML](https://baike.baidu.com/item/XML" \t "https://baike.baidu.com/item/CSS/_blank)等文件样式的计算机语言。CSS不仅可以静态地修饰网页，还可以配合各种脚本语言动态地对网页各元素进行格式化。CSS 能够对网页中元素位置的排版进行像素级精确控制，支持几乎所有的字体字号样式，拥有对网页对象和模型样式编辑的能力。

JavaScript一种直译式脚本语言，是一种动态类型、弱类型、基于原型的语言，内置支持类型。它的解释器被称为JavaScript引擎，为浏览器的一部分，广泛用于客户端的脚本语言，最早是在[HTML](https://baike.baidu.com/item/HTML" \t "https://baike.baidu.com/item/_blank)网页上使用，用来给HTML网页增加动态功能。

jQuery是一个快速、简洁的JavaScript框架，是继Prototype之后又一个优秀的JavaScript代码库（*或JavaScript框架*）。jQuery设计的宗旨是“write Less，Do More”，即倡导写更少的代码，做更多的事情。它封装JavaScript常用的功能代码，提供一种简便的JavaScript设计模式，优化HTML文档操作、事件处理、动画设计和Ajax交互。jQuery的核心特性可以总结为：具有独特的链式语法和短小清晰的多功能接口；具有高效灵活的css选择器，并且可对CSS选择器进行扩展；拥有便捷的插件扩展机制和丰富的插件。jQuery兼容各种主流浏览器，如IE 6.0+、FF 1.5+、Safari 2.0+、Opera 9.0+等。

Bootstrap 是基于 HTML、CSS、JavaScript 的，它简洁灵活，使得 Web 开发更加快捷。它由Twitter的设计师Mark Otto和Jacob Thornton合作开发，是一个CSS/HTML框架。Bootstrap提供了优雅的HTML和CSS规范，它即是由动态CSS语言[Less](https://baike.baidu.com/item/Less" \t "https://baike.baidu.com/item/Bootstrap/_blank)写成。Bootstrap一经推出后颇受欢迎，一直是[GitHub](https://baike.baidu.com/item/GitHub" \t "https://baike.baidu.com/item/Bootstrap/_blank)上的热门开源项目，包括[NASA](https://baike.baidu.com/item/NASA" \t "https://baike.baidu.com/item/Bootstrap/_blank)的MSNBC（微软全国广播公司）的Breaking News都使用了该项目。国内一些移动开发者较为熟悉的框架，如[WeX5](https://baike.baidu.com/item/WeX5" \t "https://baike.baidu.com/item/Bootstrap/_blank)前端开源框架等，也是基于Bootstrap源码进行性能优化而来。

ECharts，一个纯 Javascript 的图表库，可以流畅的运行在 PC 和移动设备上，兼容当前绝大部分浏览器（IE8/9/10/11，Chrome，Firefox，Safari等），底层依赖轻量级的 Canvas 类库 [ZRender](https://github.com/ecomfe/zrender" \t "http://echarts.baidu.com/_blank)，提供直观，生动，可交互，可高度个性化定制的数据可视化图表。ECharts 3 中更是加入了更多丰富的交互功能以及更多的可视化效果，并且对移动端做了深度的优化。ECharts 提供了常规的[折线图](http://echarts.baidu.com/option.html" \l "series-line" \t "http://echarts.baidu.com/_blank)，[柱状图](http://echarts.baidu.com/option.html" \l "series-line" \t "http://echarts.baidu.com/_blank)，[散点图](http://echarts.baidu.com/option.html" \l "series-scatter" \t "http://echarts.baidu.com/_blank)，[饼图](http://echarts.baidu.com/option.html" \l "series-pie" \t "http://echarts.baidu.com/_blank)，[K线图](http://echarts.baidu.com/option.html" \l "series-candlestick" \t "http://echarts.baidu.com/_blank)，用于统计的[盒形图](http://echarts.baidu.com/option.html" \l "series-boxplot" \t "http://echarts.baidu.com/_blank)，用于地理数据可视化的[地图](http://echarts.baidu.com/option.html" \l "series-map" \t "http://echarts.baidu.com/_blank)，[热力图](http://echarts.baidu.com/option.html" \l "series-heatmap" \t "http://echarts.baidu.com/_blank)，[线图](http://echarts.baidu.com/option.html" \l "series-lines" \t "http://echarts.baidu.com/_blank)，用于关系数据可视化的[关系图](http://echarts.baidu.com/option.html" \l "series-graph" \t "http://echarts.baidu.com/_blank)，[treemap](http://echarts.baidu.com/option.html" \l "series-treemap" \t "http://echarts.baidu.com/_blank)，多维数据可视化的[平行坐标](http://echarts.baidu.com/option.html" \l "series-parallel" \t "http://echarts.baidu.com/_blank)，还有用于BI的[漏斗图](http://echarts.baidu.com/option.html" \l "series-funnel" \t "http://echarts.baidu.com/_blank)，[仪表盘](http://echarts.baidu.com/option.html" \l "series-gauge" \t "http://echarts.baidu.com/_blank)，并且支持图与图之间的混搭。

**10.2 界面设计与布局分析**

分布式人口管理系统首页如图10-1所示。页面顶部是一个导航栏，导航栏左边是系统图标和系统名称，右边是其他页面的跳转超链接。页面中部是一个图片轮播，图片轮播表达的内容是中国的人口数量庞大，图片轮播的下面分四列显示了四组文字数据，从数据能得知2016年的人口情况。页面底部是系统的版权声明。

分布式人口管理系统数据展示页面如图10-2所示。顶部和底部与首页一致，分别是导航栏和版权申明。页面中部一个地区（省、市、区）的三级联动选择，选择了省才会出现市的选择，选择了市才会出现区的选择。在这个选择区下方左边是查询结果展示方式的选择，包括按总人口展示、按性别展示、按年龄段展示、按民族展示、按姓氏展示和按行业展示。这个选择区右边则是结果展示区，数据获取成功则在该区域以图表的方式展示结果。

分布式人口管理系统数据预测页面如图10-3-1和10-3-2所示。顶部和底部与首页一致，分别是导航栏和版权申明。页面中部是数据展示区。

分布式人口管理系统管理员登录页面如图10-4所示。

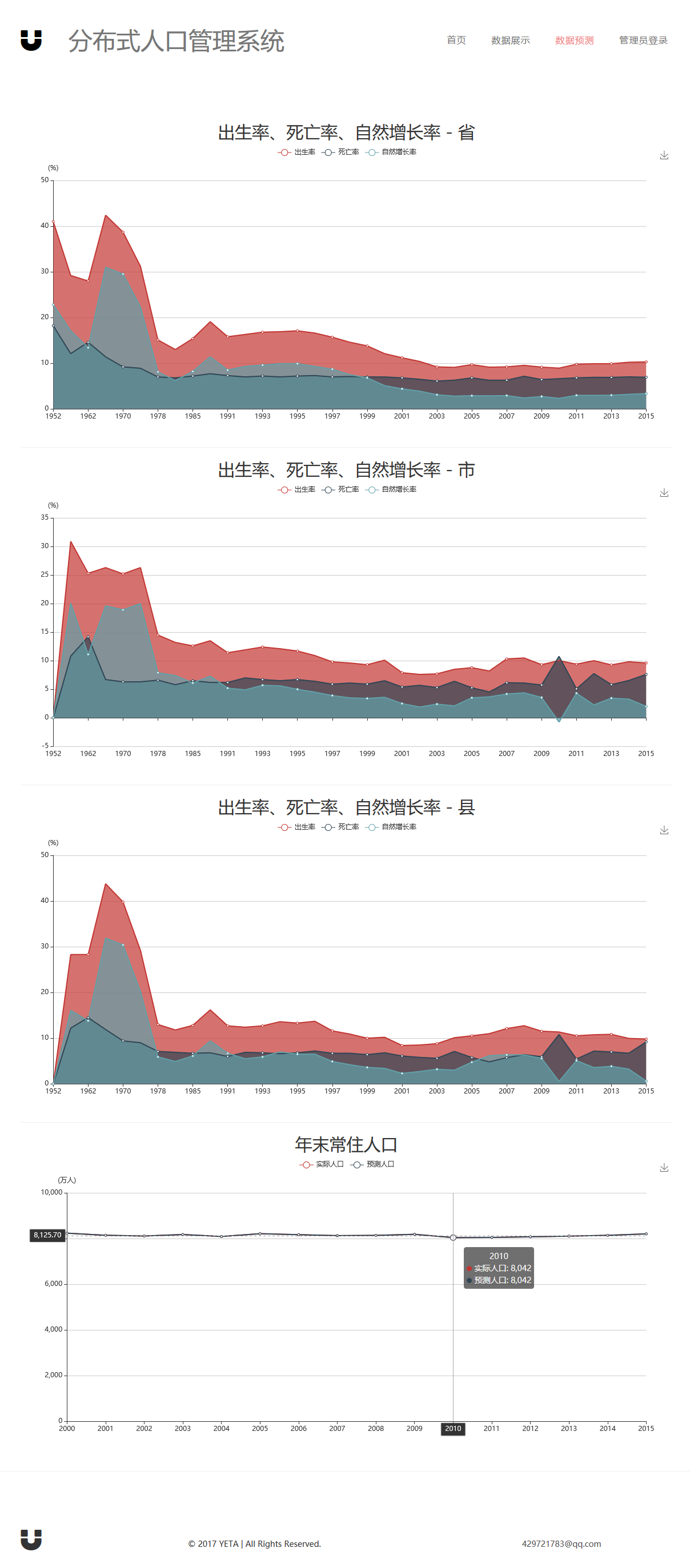
分布式人口管理系统后台管理页面如图10-5所示，顶部和底部与首页一致，分别是导航栏和版权申明。页面中部是数据管理区。

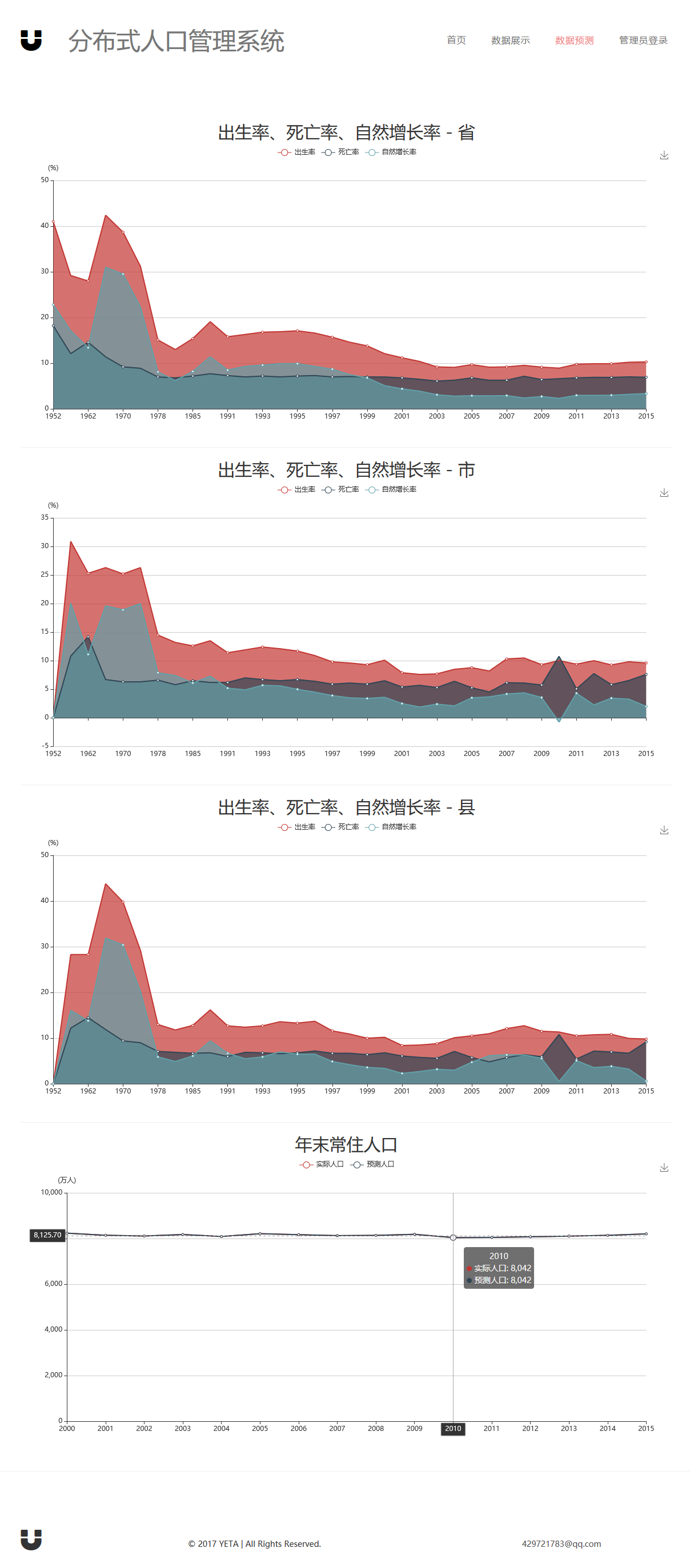


**图10-1 系统首页设计图**



**图10-2 数据展示页面设计图**

**图10-3-1 数据预测页面设计图**



**图10-3-2 数据预测页面设计图**



**图10-4 管理员登录页面设计图**

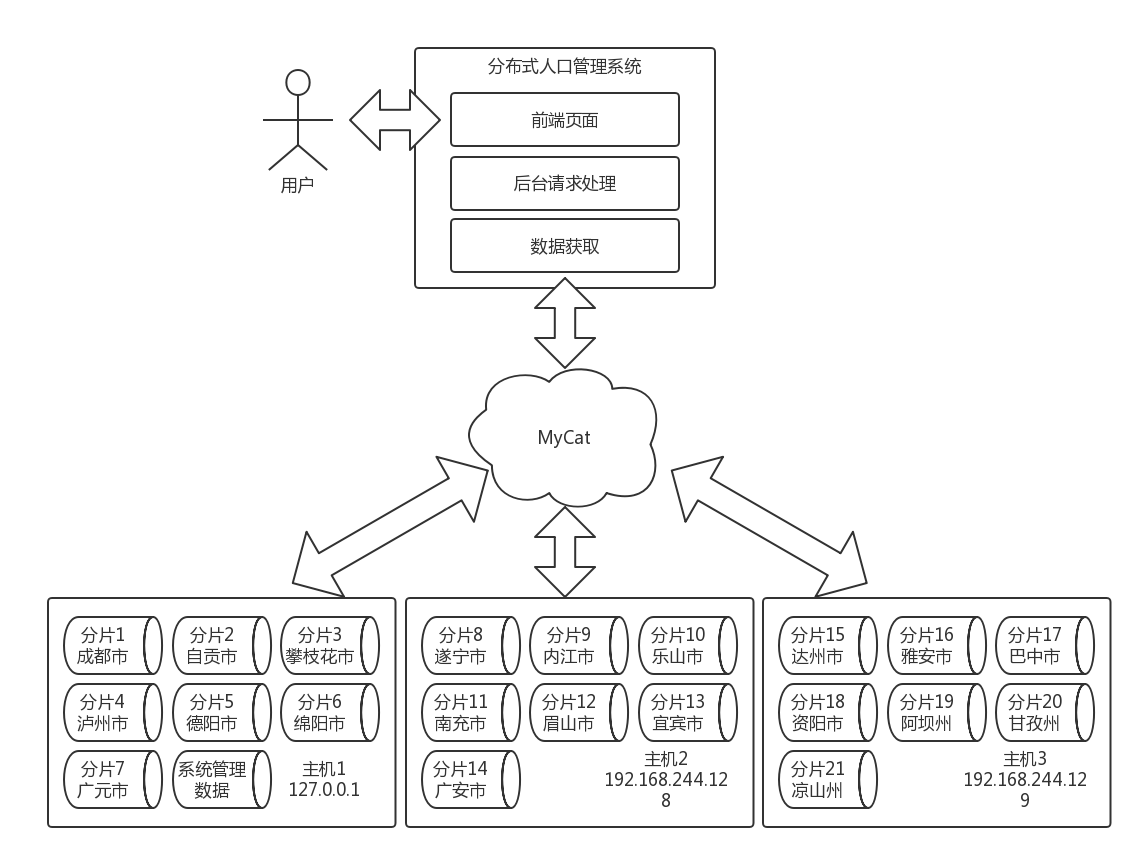


**图10-5 后台管理页面设计图**

1. **分布查询举例**

**11.1 分布查询过程**

系统分布查询过程如图11-1所示，用户通过前端页面发起查询请求，系统后台程序接收请求，通过MyCat中间件查询数据库，MyCat配置的时候已经配置好分片规则，所以它能自动的根据用户想获取的数据去对应的场地，对应的分片上进行查询。

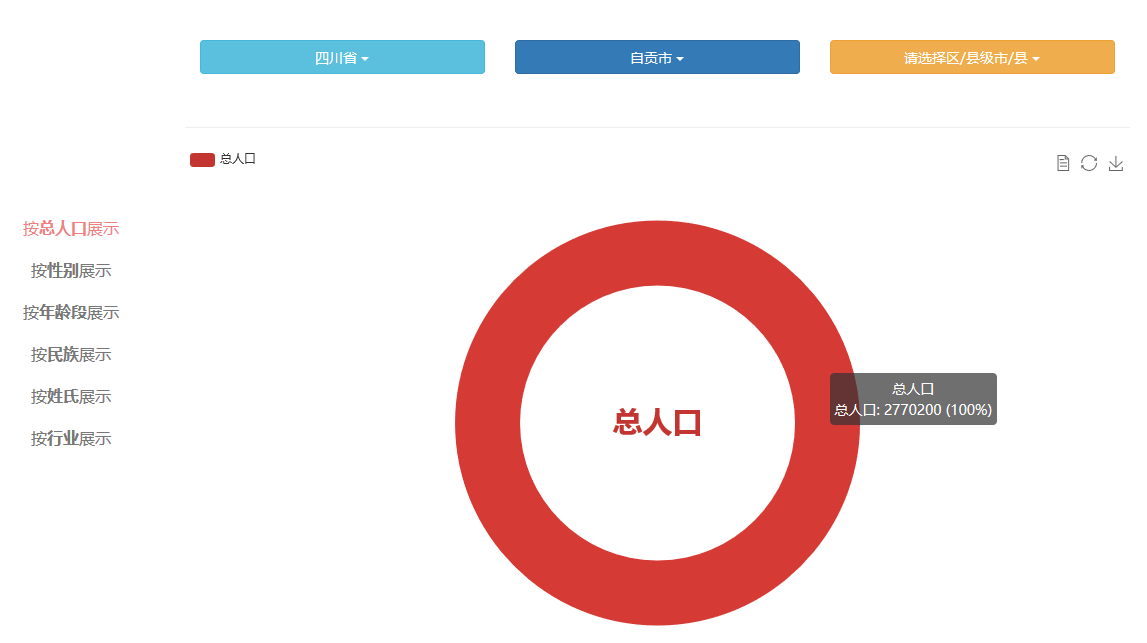
****

**图11-1 分布式查询过程图**

**11.2 分布查询举例**

1. 查询四川省自贡市总人口

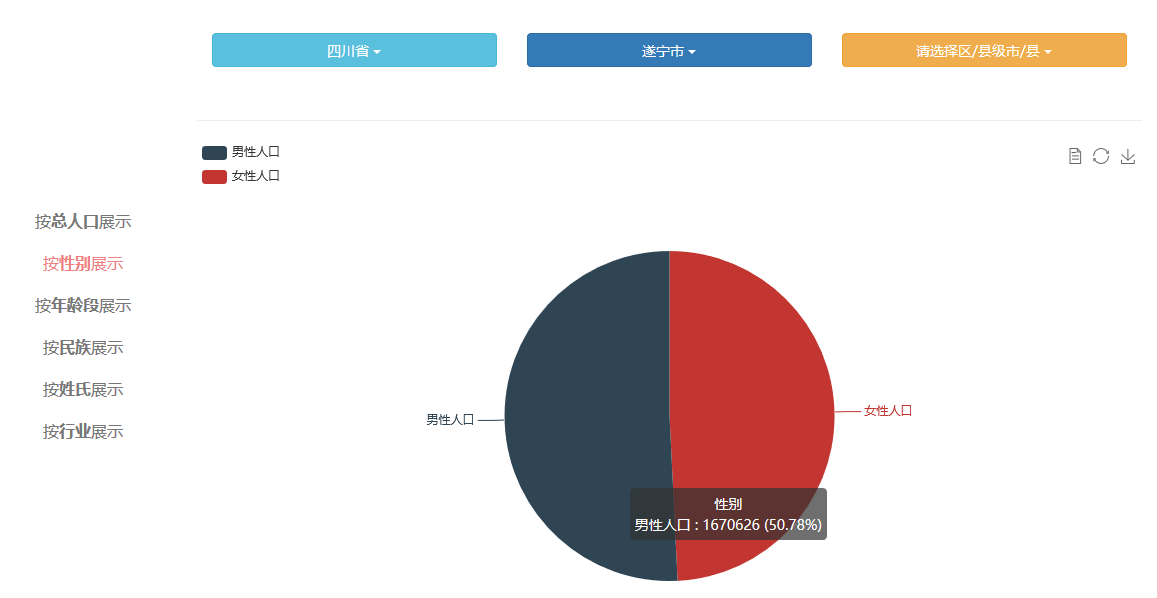
SELECT COUNT(\*) FROM people2016 WHERE city\_code = #{cityCode}



**图11-2 查询四川省自贡市总人口结果图**

1. 查询四川省遂宁市各性别人口

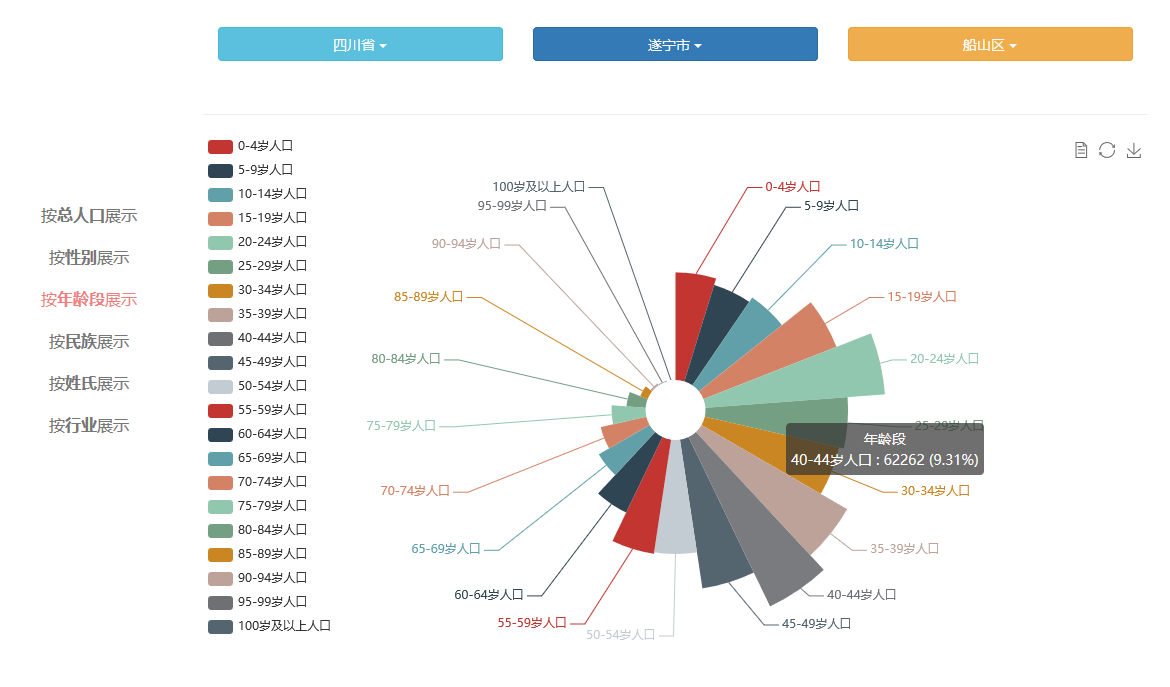
SELECT gender\_, COUNT(gender\_) AS count\_ FROM people2016 WHERE city\_code = #{cityCode} GROUP BY gender\_



**图11-3 查询四川省遂宁市各性别人口结果图**

1. 查询四川省遂宁市船山区各年龄段人口

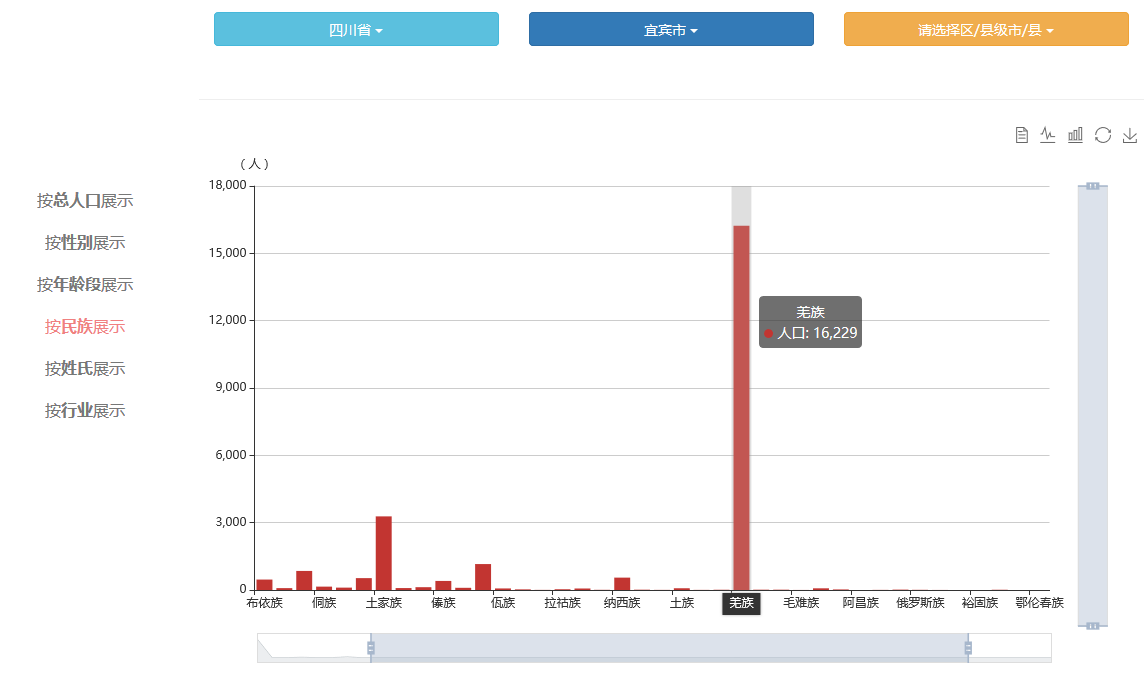
SELECT p.age\_range\_code, a.bottom\_, a.top\_, COUNT(p.age\_range\_code) AS count\_ FROM people2016 AS p INNER JOIN age\_range AS a ON p.age\_range\_code = a.age\_range\_code WHERE p.region\_code = #{regionCode} GROUP BY p.age\_range\_code ORDER BY p.age\_range\_code ASC



**图11-4 查询四川省遂宁市船山区各年龄段人口结果图**

1. 查询四川省宜宾市各民族人口

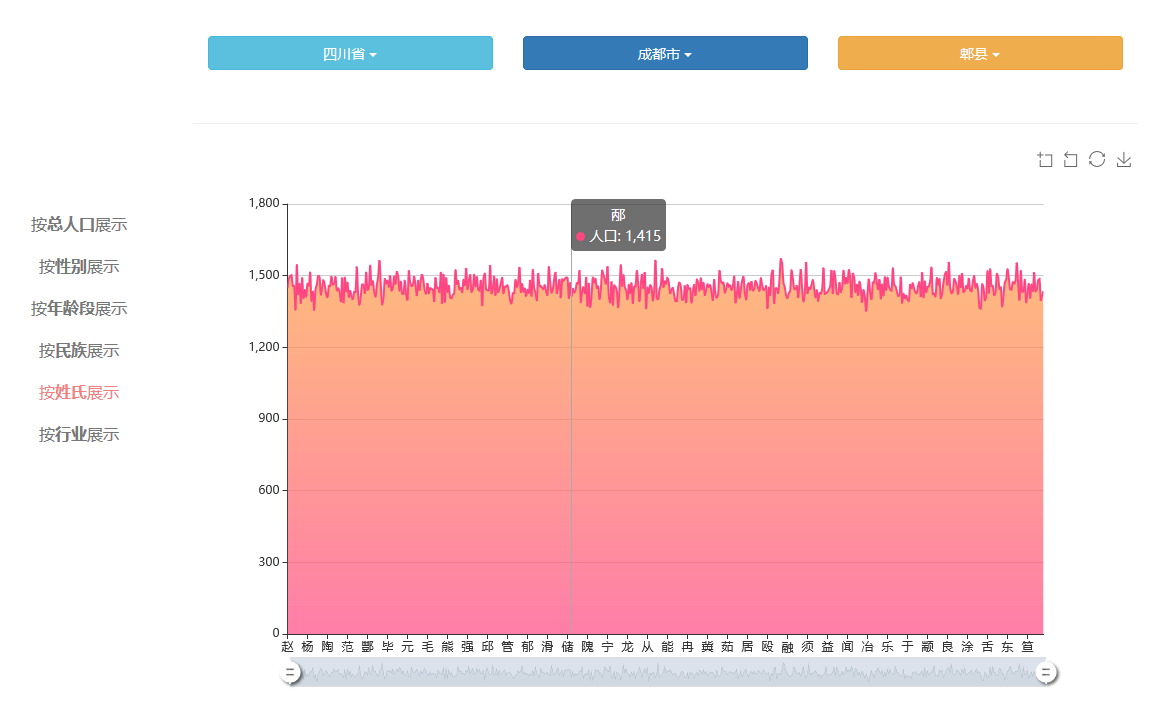
SELECT p.nation\_code, p.nation\_, COUNT(p.nation\_code) AS count\_ FROM people2016 AS p WHERE p.city\_code = #{cityCode} GROUP BY p.nation\_code ORDER BY p.nation\_code ASC



**图11-5 查询四川省宜宾市各民族人口结果图**

1. 查询四川省成都市郫县各姓氏人口

SELECT p.last\_name\_code, p.last\_name, COUNT(p.last\_name\_code) AS count\_ FROM people2016 AS p WHERE p.region\_code = #{regionCode} GROUP BY p.last\_name\_code ORDER BY p.last\_name\_code ASC



**图11-6 查询四川省成都市郫县各姓氏人口结果图**

1. 查询四川省各行业人口

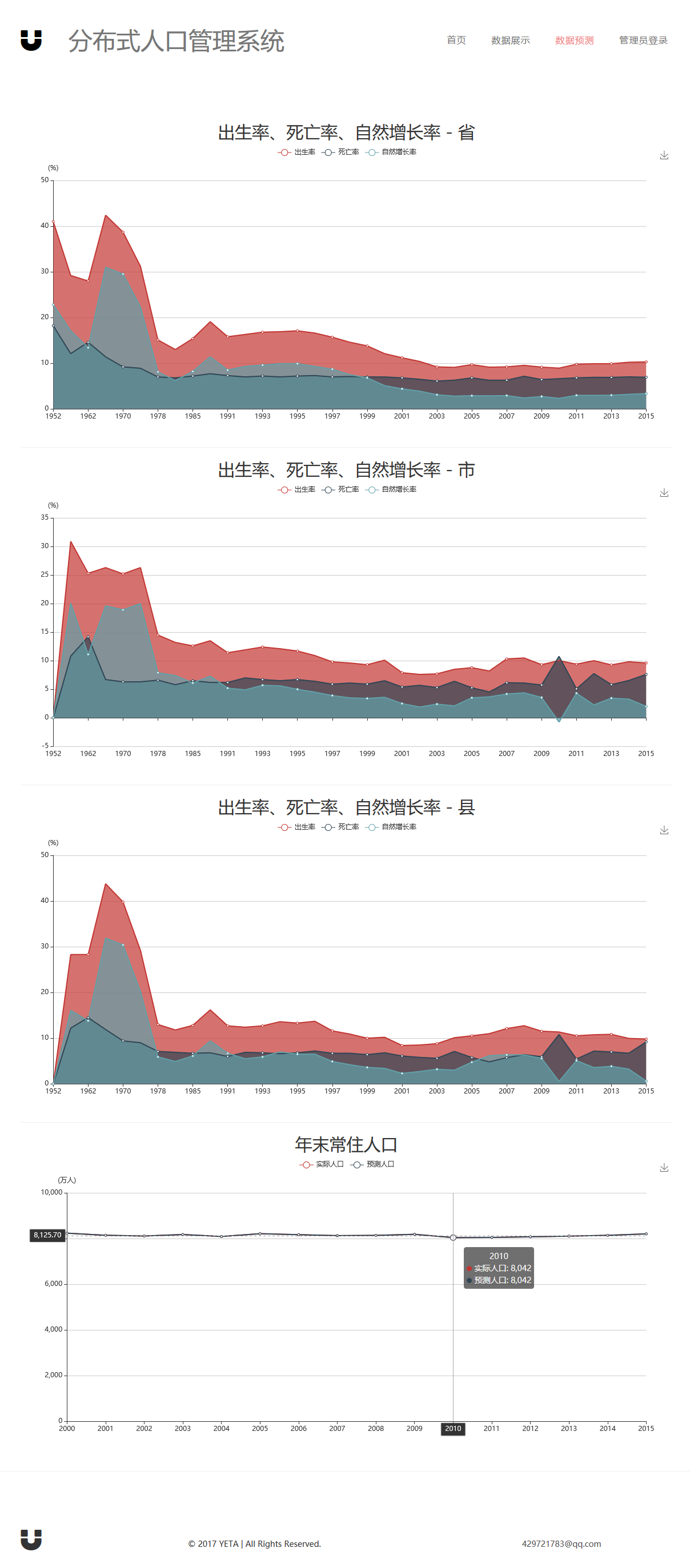
SELECT p.profession\_code, p.profession\_, COUNT(p.profession\_code) AS count\_ FROM people2016 AS p WHERE p.province\_code = #{provinceCode} GROUP BY p.profession\_code



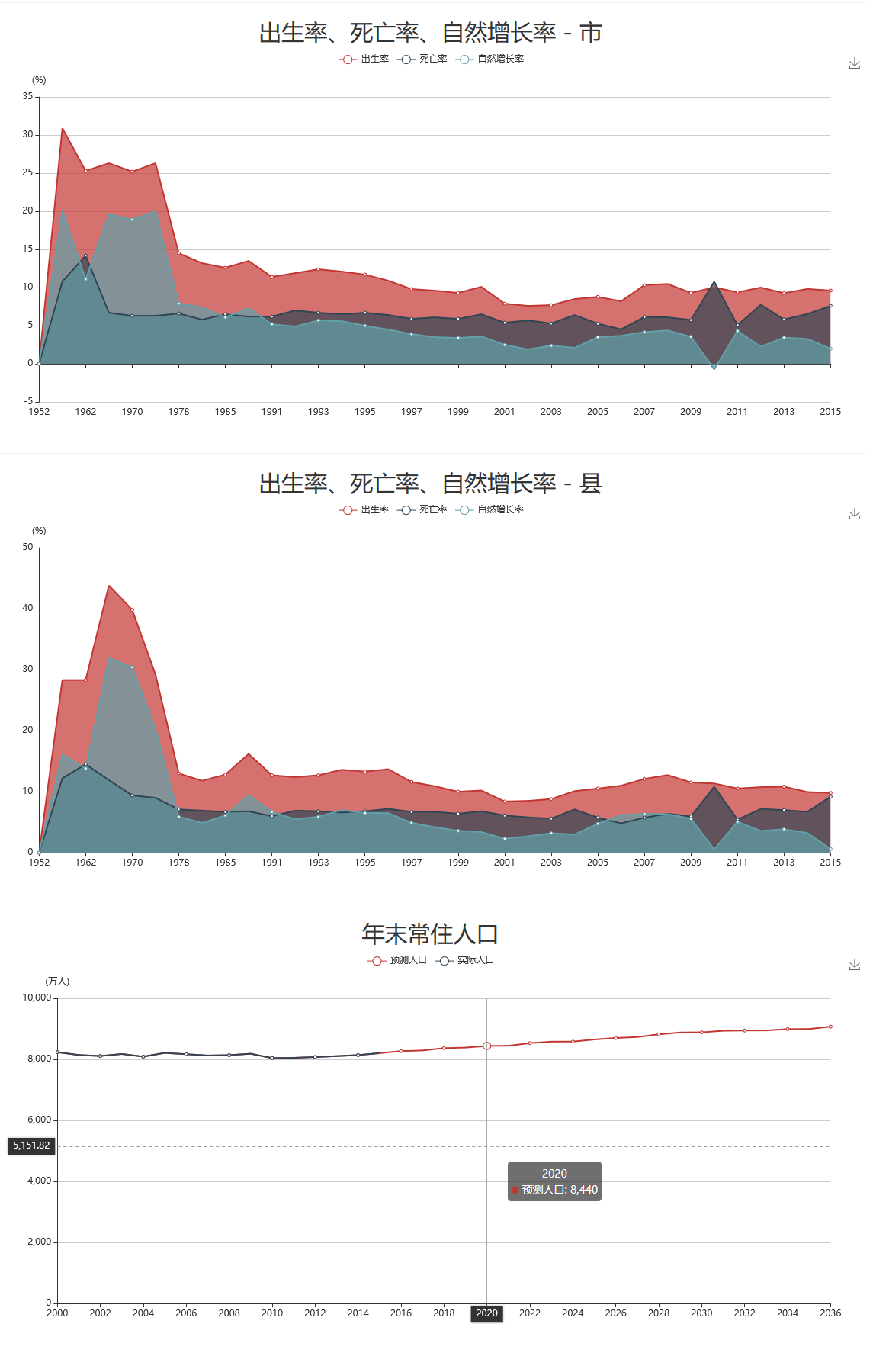
**图11-7 查询四川省各行业人口结果图**

1. 查询四川省历年常住人口、出生率、死亡率、自然增长率、预测人口

SELECT r.rate\_code, r.year\_, r.people\_number, r.province\_birth\_rate, r.province\_death\_rate, r.province\_growth\_rate, r.city\_birth\_rate, r.city\_death\_rate, r.city\_growth\_rate, r.region\_birth\_rate, r.region\_death\_rate, r.region\_growth\_rate FROM rate AS r



**图11-8-1 查询结果图**



**图11-8-2 查询结果图**

1. **系统主要操作与功能说明**

**表12-1 系统主要操作与功能说明表**

|  |  |  |
| --- | --- | --- |
| **序号** | **操作/功能** | **说明** |
| 1 | 首页数据展示 | 让用户知道中国人口现状 |
| 2 | 2016年四川省人口统计 | 从地区、总人口、性别、民族、姓氏、行业、年龄段等几方面让用户能全面了解人口数量 |
| 3 | 数据预测 | 让用户了解四川省历年常住人口、人口出生率、死亡率、自然增长率、以及预测人口 |
| 4 | 登录 | 管理员登录之后对系统进行管理 |
| 5 | 地区管理 | 由于存在地区信息变更的情况，例如简阳县改为简阳市，之前属于资阳市，现在属于成都市 |
| 6 | 日志管理 | 日志主要记录用户访问系统的每个请求，包括请求路径、请求方法、请求参数、返回内容、耗时等，便于系统的维护，出现错误可以及时发现，也可能请求耗时对系统性能做出相应改进 |
| 7 | 用户管理 | 适用于超级管理员对管理员用户进行管理 |

1. **参考文献**
2. 马东波.分布式数据库系统的设计与实现[J].产业与科技论坛,2017,16(15):82-83.
3. 陈华.浅谈分布式数据库系统的设计与优化[J].电脑知识与技术,2017,13(06):1+5.
4. 万勇.数据库在分布式管理系统中的应用[J].电子技术与软件工程,2017,(01):167.
5. 马东波.分布式数据库系统的安全机制[J].产业与科技论坛,2017,16(16):45-46.
6. 靳晟,李永可,李悦,陈燕红.基于分布式的农业信息检索系统的设计与实现[J].电脑知识与技术,2017,13(01):237-238.
7. 黄冬英.基于分布式数据库的高校学生档案管理系统设计与实现[D].苏州大学,2016.
8. 闫兵.基于分布式数据库的图书馆自动管理系统设计[J].西南师范大学学报(自然科学版),2016,41(02):147-153.
9. 王普刚.基于HBase的工业日志系统设计与实现[D].大连理工大学,2016.
10. 王葱.基于MyCAT的分布式数据存储研究与应用[D].东华大学,2016.
11. 姚丽丽,万玉建,朱峰.基于MyCat的建筑能耗监管系统数据库设计与实现[J].电脑编程技巧与维护,2015,(24):5-7+14.
12. 鞠剑勋.基于HBase的人口管理系统的设计与实现[D].南京大学,2013.
13. 陈忠昊.实有人口管理系统的开发与运用[D].贵州大学,2009.