基于大数据的疫情可视化分析平台

1需求分析

1.1引言

1.1.1背景

自从2019年尾2020年初进入疫情时代以来，新冠疫情对人们的生活影响是巨大的，通信行程卡和健康码都成为了出行必备，全国人民也都很关注各地区疫情状况，大数据技术通过追踪移动轨迹、建立个体关系图谱等，在精准定位疫情传播路径，防控疫情扩散方面起到重要作用。利用大数据技术还可以梳理感染者的生活轨迹，追踪人群接触史，成功锁定感染源及密切接触人群，为疫情防控提供宝贵信息。

在位置数据方面，除了航空、铁路、公路、轮渡等交通部门统计的出行数据外，在用户授权的前提下，中国移动、中国联通、中国电信三大运营商基于手机信令能够有效定位用户的手机位置，互联网企业也可以通过APP授权调用用户手机位置数据。此外，地图、打车等APP提供的移动出行服务，电商、外卖平台等APP内的送货地址数据，以及移动支付位置数据等，也可以作为位置数据的有效补充。而关系图谱则可通过各类社交平台、通信网络、通话记录、转账记录等数据搭建。

而将这些个体数据集合形成的群体数据，则能够清晰显示重要疫区的人员流入及流出方向、动态及规模，这些数据有利于定位疫情输出的主要区域、预测地区疫情发展态势、预测地区潜在染病人群，为疾病防控部门及地区政府分类制定春运返程计划、有针对性地出台交通管制措施等提供决策支撑。

1.1.2用户特点

本项目服务的对象是网络真实用户，对多层次的用户都普适。

1.2需求分析

1.2.1总体需求分析

能够可视化地显示当前中国各地区疫情情况，包括新增人数、治愈人数、死亡人数、无症状人数和发展趋势等，并且对接下来的疫情形势作出分析预测。

1.2.2需求描述

1.2.2.1各地区疫情统计

每日新增：当日新增的新冠感染者

每日治愈：当日治愈的新冠感染者

每日死亡：当日死亡的新冠感染者

每日无症状：当日新增的无症状新冠感染者

累计确诊：累计感染的新冠人数

累计治愈：累计治愈的新冠感染者人数

累计死亡：累计死亡的新冠感染者人数

1.2.2.2疫情风险分析

通过折线图展示城市近24天每日新增的感染者人数变化，同时对比昨日与今日确诊、死亡、治愈和无症状人数来对疫情情况进行分析。

2概要设计

2.1引言

2.1.1编写目的

利用比较抽象的语言对整个需求进行概括，确定对系统的物理配置，确定整个系统的处理流程和系统的数据结构、接口设计、人机界面，实现对系统的初步设计。

2.1.2背景

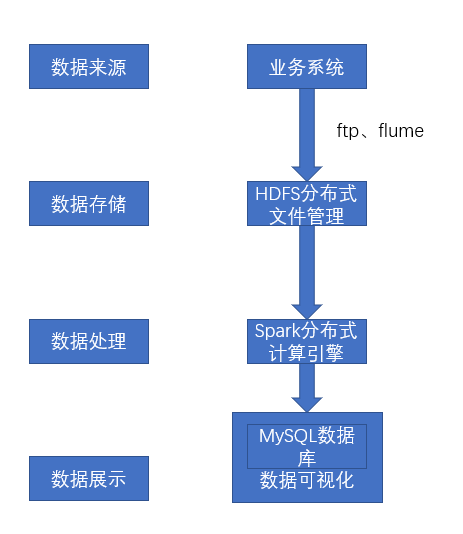
随着疫情的发展人们对各地区疫情情况的发展非常关注，因此本项目利用大数据技术搭建一个疫情可视化分析平台，帮助人们了解各地区疫情情况和对接下来的疫情发展进行分析预测，辅助人们对出行等日常生活进行更好的判断和准备。

2.1.3术语解释

|  |  |
| --- | --- |
| 术语 | 解释 |
| Hadoop | 分布式系统的基础架构 |
| Flume | 一个高可用的，高可靠的，分布式的海量日志采集、聚合和传输的系统 |
| Spark | 基于内存计算的开源的集群计算系统，为速度和通用目标所设计的集群计算平台 |
| ETL | 数据抽取（Extract）、清洗（Cleaning）、转换（Transform）、装载（Load）的过程 |
| 数据可视化 | 用Springboot+mybatis-plus+SpringMVC  +Ajax+Thymeleaf+html5+css3+js+jQuery+  Echarts+MySQL技术，把统计出的数据展示出来 |

2.3系统总体设计描述

根据前面的功能需求分析和系统整体的需求，将整个疫情可视化平台系统划分为以下几层：数据来源层、数据存储层、数据处理层和数据展示层。下图为从数据来源层到数据展示层的结构图：



疫情可视化分析平台系统的总体设计是从网络相关数据平台爬取数据信息进行分析有用的信息，然后以前端界面可视化展现，设计该系统的目标是为人们提供一个疫情数据分析平台，让人们对各地区的疫情情况有更多的了解，方便人们进行出行等活动。

为了最后实现该系统，在总体设计中把握一下重点，然后在详细设计中可以分别对每个细节难点和重点一一解决。

1. 各地区每天产生的疫情相关的数据量是非常庞大的，而且数据种类多，数据类型复杂，进行过滤、清洗是一项很复杂的工作，要实现数据的获取、转换以及加载的过程就需要把所有的可能有关的源数据和它们的数据的格式，以及Flume接收数据的格式，都清晰的整理出来才可以进行接下来的操作，所以Flume组件合理地设计和使用是获取和数据存储的关键。
2. 页面与数据库 MySQL 接口的对接问题。在页面开发人员对数据进行操作 时，Web 要与 MySQL 进行交互，它们两者的接口就需要互相统一，并且要有较 高的效率。所以在编写 MySQL 接口的时候，需要考虑到它们的性能、效率、优 化等问题，让接口符合系统的要求。

2.4各个模块设计描述

2.4.1数据来源模块描述

不管一个系统是多么复杂、多么简单，它们的共同点就是都需要符合要求的 数据，这样才可以接下来的操作。也可以这么说，如果一个系统代码是实现系统 功能的主要部分，那么数据就是核心。因此该系统的数据也是一个值得关注的部分。该项目的数据来源于其他网络平台统计的数据。

2.4.2数据存储模块描述

本模块主要有两部分，第一部分是数据源放入到 Hadoop 的 HDFS 分布式文件系统中，第二部分就是把处理好的数据存储到 MySQL 数据库中。

2.4.3数据处理模块描述

该模块用到的技术是 Hadoop 生态圈中的 Hive数据仓库框架。运用 Hive中的 Hive-sql 对 HDFS 上的文件进行处理，最后把处理好的数据存储到 MySQL 数据库中。

2.4.4数据展示模块描述

通过数据可视化技术将分析结果绘制图表展示到 Web 页面。

3详细设计及实现

3.1引言

本详细设计说明书是针对基于大数据的疫情可视化分析平台系统而编写，目的是对该项目进行详细设计，在概要设计的基础上进一步明确系统结构，详细的介绍各个系统模块，为后面的实现和测试做准备。本详细设计说明书的预期读者为本项目的小组成员以及对该系统感兴趣，在以后想对系统进行扩展和维护的人员。

3.2集群设计

硬件：5 台服务器，1 台主节点，2 台从节点，4 台作业提交节点。

软件：HDFS、MapReduce、Hive、Sqoop、HBase、Zookeeper。

服务器的配置：Intel(R)Xeon(R) CPU E5645 @2.40GHz

内存：100G

硬盘：2T

每天处理的数据量：5GB

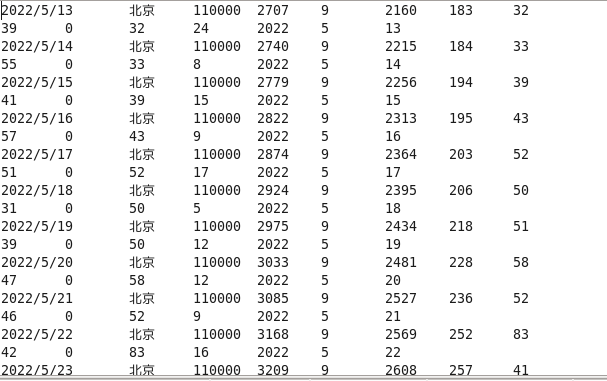
3.3数据来源模块的详细设计

3.3.1数据描述

本数据来源于网络数据爬取，其中一个来自于国家疫情统计公布数据，还有服务商接口数据，数据的各个字段表示如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 描述 |
| 0 | province | Int | 省份 |
| 1 | TimeS | Date | 时间 |
| 2 | 每日新增 | int | 每日新增 |
| 3 | 每日治愈 | int | 每日治愈 |
| 4 | 每日死亡 | int | 每日死亡 |
| 5 | 每日无症状 | Int | 每日无症状 |
| 6 | 累计确诊 | int | 累计确诊 |
| 7 | 累计治愈 | int | 累计治愈 |
| 8 | 累计死亡 | int | 累计死亡 |

3.3.2目标数据描述

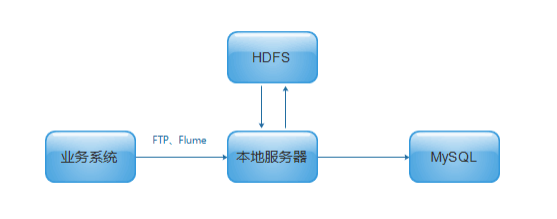


3.4数据存储模块详细设计

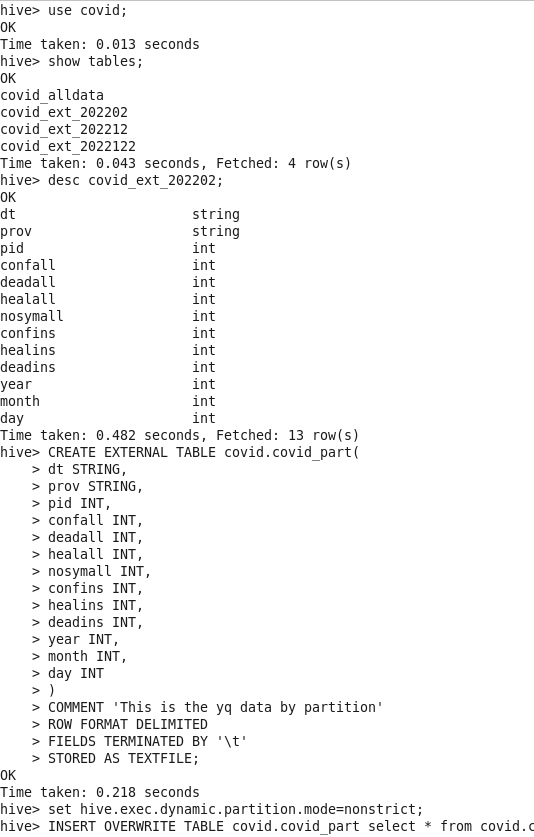
该模块的详细设计，首先选用的是 Python爬虫爬取数据到本地，将本地数据上传到 HDFS并进行处理，将处理完的数据通过hive转为SQL并用sqoop下载到本地，并且将数据导

入到 MySQL。下面对该系统中数据流向进行介绍。图 4-19 展示的是数据流向

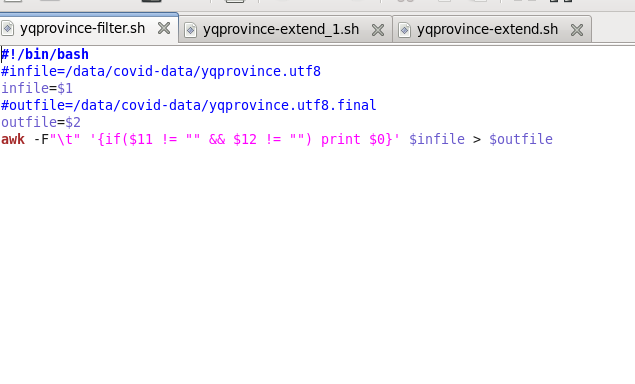
图。



Hive: 是基于Hadoop的一个数据仓库工具，可以将结构化的数据文件映射为一张数据库表，并提供简单的sql查询功能，可以将sql语句转换为MapReduce任务进行运行。 其优点是学习成本低，可以通过类SQL语句快速实现简单的MapReduce统计，不必开发专门的MapReduce应用，十分适合数据仓库的统计分析。



3.5数据处理模块详细设计

3.6数据展示模块详细设计

3.6.1数据可视化

SpringBoot:Spring的快速开发框架，其思想是“约定大于配置”，因此SpringBoot可以快速建立Web项目并且进行项目搭建。本项目主要利用SpringBoot构建项目，利用容器管理各种对象。

Mybatis-plus：国内开发的对Mybaitis这种半自动orm框架进行增强的全自动orm框架，可以快速对mapper层接口进行常用方法实现，配合Mybatis-X插件可以进行快速开发，编写业务层代码。同时也可以配合xml文件进行复杂sql编写。

Thymeleaf:前端引擎，可以构建动态html界面，将后端数据回显过来，支持el表达式，只需要在html标签添加上xmlns:th=<http://www.thymeleaf.org>属性即可使用，类似jsp技术中的c标签和el表达式。

JavaScript:一种脚本语言，是一种动态型、弱型、基于原型的语言，内置支

持类型。它的解释器被叫做 JavaScript 引擎，是浏览器的一部分，非常广泛用于

客户端的脚本语言，用来给 HTML 网页增加动态功能，增强页面的友好度。js

语言和他语言一样，有它自己的基本数据类型，表达式和算术运算符及程序的基

本程序框架。js 提供了四种基本的数据类型和两种特殊数据类型用来处理数据和

文字。变量提供存放信息的地方，表达式则可以完成较复杂的信息处理

jQuery：对javascript进行封装，简化javascripe开发和ajax请求编写，非常方便，代码非常简洁。

echart: 一个用 javascript 写的图表库，能非常流畅的运行在 PC 和移动设备

上，兼容性非常好，底层依赖轻量级的 Canvas 类库 ZRender，提供直观，生动，

交互性高，可高度个性化定制的数据可视化图表。

mvc 框架: 英文名字是 Model View Controller，是模型(model)－视图(view)

－控制器(controller)的缩写，用一种业务逻辑、数据、界面 显示分离的方法组织

代码，将业务逻辑聚集到一个部件里面，在改进和个性化定制界面及用户交互的

同时，不需要重新编写业务逻辑。MVC 用于映射传统的输入、处理和输出功能

在一个逻辑的图形化用户界面的结构中。

Ajax：即“Asynchronous Javascript And XML”（异步 JavaScript 和

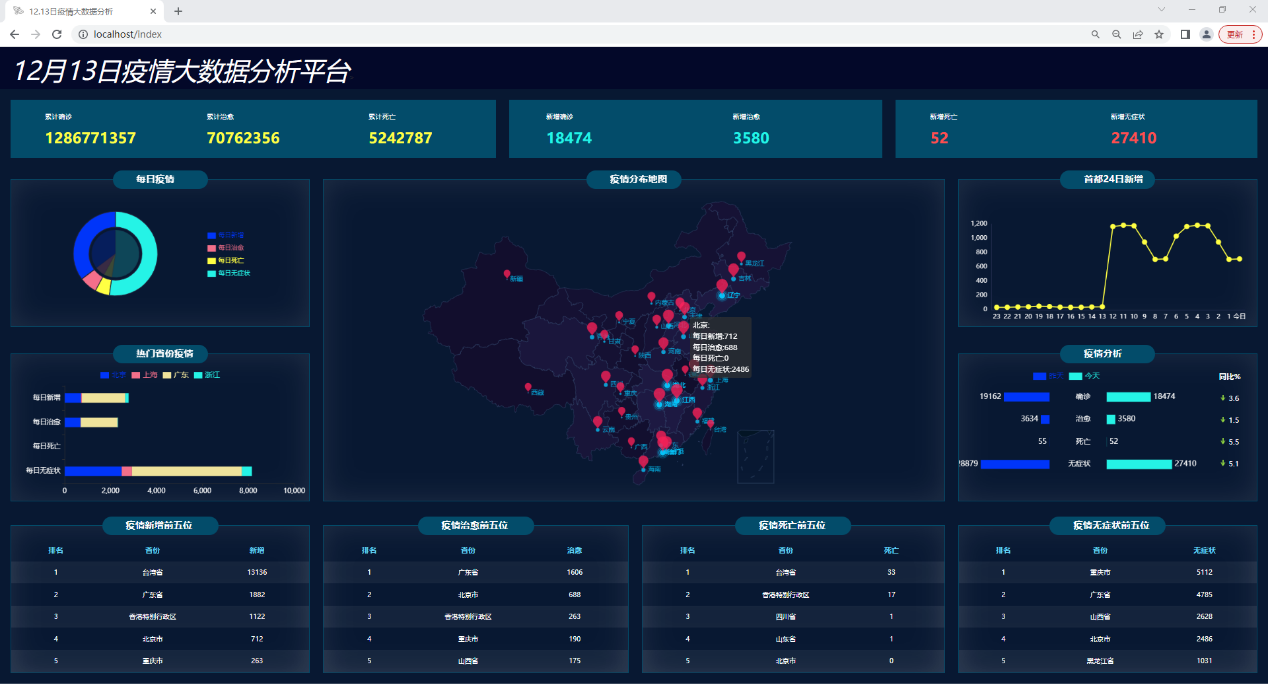
XML），是指一种创建交互式网页应用的网页开发技术。AJAX 是一种用于创建快

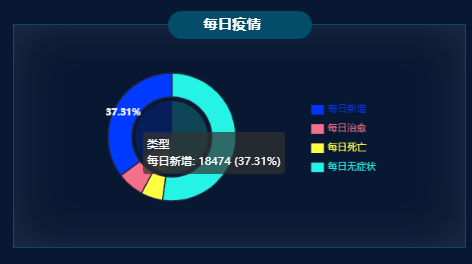
速动态网页的技术。通过在后台与服务器进行少量数据交换，AJAX 可以使网页

实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分

进行更新

3.6.2界面设计













4编码文档

4.1数据分析

4.1.1Sql脚本解释

无sql脚本

4.2数据可视化

4.2.1 Thymeleaf

前端引擎框架，这里专门处理非ajax的同步请求，将后端数据回显到前端html界面中的标签内，支持动态绑定参数。



例如：

图示

中度可信度描述已自动生成

4.2.2包

屏幕的截图

描述已自动生成

4.2.3类

文本

描述已自动生成文本

描述已自动生成文本

描述已自动生成图形用户界面, 文本, 应用程序, 聊天或短信

描述已自动生成

4.2.4资源

图形用户界面, 文本, 应用程序

描述已自动生成电脑萤幕画面

描述已自动生成图形用户界面

低可信度描述已自动生成文本

描述已自动生成

4.2.5配置文件内容



文本

描述已自动生成



<?xml version="1.0" encoding="UTF-8" ?>  
<configuration>  
 <!--定义日志文件的存储地址-->  
 <property name="LOG\_HOME" value="${catalina.base}/logs/"/>  
  
 <!-- 控制台输出 -->  
 <appender name="Stdout" class="ch.qos.logback.core.ConsoleAppender">  
 <!-- 日志输出编码 -->  
 <layout class="ch.qos.logback.classic.PatternLayout">  
 <!--格式化输出：%d表示日期，%thread表示线程名，%-5level：级别从左显示5个字符宽度%msg：日志消息，%n是换行符-->  
 <pattern>%d{MM-dd HH:mm:ss.SSS} [%thread] %-5level %logger{50} - %msg%n  
 </pattern>  
 </layout>  
 </appender>  
  
 <!-- 按照每天生成日志文件 -->  
 <appender name="RollingFile" class="ch.qos.logback.core.rolling.RollingFileAppender">  
 <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">  
 <!--日志文件输出的文件名-->  
 <FileNamePattern>${LOG\_HOME}/server.%d{yy99-MM-dd}.log</FileNamePattern>  
 <MaxHistory>30</MaxHistory>  
 </rollingPolicy>  
 <layout class="ch.qos.logback.classic.PatternLayout">  
 <!--格式化输出：%d表示时间，%thread表示线程名，%-5level：级别从左显示5个字符宽度%msg：日志消息，%n是换行符-->  
 <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{50} - %msg%n  
 </pattern>  
 </layout>  
 <!--日志文件最大的大小-->  
 <triggeringPolicy class="ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy">  
 <MaxFileSize>10MB</MaxFileSize>  
 </triggeringPolicy>  
 </appender>  
  
 <!-- 日志输出级别 -->  
 <root level="info">  
 <appender-ref ref="Stdout"/>  
 <appender-ref ref="RollingFile"/>  
 </root>  
</configuration>