



北京科技大学
University of Science and Technology Beijing

密级： 公开

本科生毕业设计(论文)

题 目： 租房用户行为数据分析
系统的设计与实现

作 者： 邵伟杰

学 号： 41724352

学 院： 计算机与通信工程

专 业： 通信工程

成 绩：

2021 年 06 月

本科生毕业设计(论文)

题 目: 租房用户行为数据分析

系统的设计与实现

英文题目: Design and Realization of Renting

User Behavior Data Analysis

学 院: 计算机与通信工程

班 级: 通信 1704

学 生: 邵伟杰

学 号: 41724352

指导教师: 任超 职称: 讲师

声 明

本人郑重声明：所呈交的论文是本人在指导教师的指导下进行的研究工作及取得研究成果。论文在引用他人已经发表或撰写的研究成果时，已经作了明确的标识；除此之外，论文中不包括其他人已经发表或撰写的研究成果，均为独立完成。其他同志对本文所做的任何贡献均已在论文中做了明确的说明并表达了谢意。

学生签名：_____年__月__日

导师签名：_____年__月__日

摘 要

近几年,随着中国的城市化进程不断加速,城市流动人口规模不断的增加,一二线城市的房价也随之不断攀升,租房已经成为一种社会刚需。各大租房平台也进行了线上线下相结合的尝试,逐渐将运营重心转移到线上。但是伴随着互联网数据规模爆炸式的增长,线上租房平台也面临着挑战,每天都会产生海量的用户数据。如何能从这些用户数据中挖掘用户的行为信息,已成为租房平台的重要目标。这些用户的行为数据如何采集,存储,整理和分析都是大数据系统需要解决的问题。

本文分析了国内外互联网行业的用户行为分析模式,包括相关理论和框架,提出了在目前租房市场快速发展的背景下,设计了一套租房用户行为数据分析系统。本租房用户行为数据分析系统是一个分布式的大数据系统。采用了 Hadoop 分布式大数据框架,利用 Webmagic 框架进行网站数据爬取,通过 kafka 和 Flume 采集日志文件,并使用 Mysql 来进行数据持久化存储。系统后端工程基于 Maven 搭建和统一规范管理,使用了 SpringBoot + SpringMVC + Mybatis 这三个框架进行开发。前端采用了 vue 进行开发,最终实现在 Web 页面上的可视化大数据系统。在系统开发完成后,对系统进行了多项测试,验证了系统的稳定性。论文的主要工作包括以下几个方面:

- (1) 对系统设计背景和意义,系统设计中用到的服务框架、中间件技术进行了介绍;
- (2) 对系统进行了可行分析,功能性分析以及非功能性需求分析;
- (3) 对租房用户行为数据的采集、分析和可视化功能进行了详细设计;
- (4) 实现了系统的主要功能,数据采集、数据仓库和数据可视化功能,提供了最终效果图。
- (5) 在软件测试部分,本文对各个关键模块进行了单元测试,接口测试和功能测试,验证了软件结果数据与预期一致。

关键词: 大数据, 租房, 用户行为, 数据分析

Design and Realization of Renting User Behavior Data Analysis

Abstract

In recent years, as China's urbanization process continues to accelerate and the scale of urban floating population continues to increase, housing prices in first- and second-tier cities have also continued to rise. Renting houses has become a rigid social demand. The major rental platforms have also tried to combine online and offline, gradually shifting the focus of operations to online. However, with the explosive growth of Internet data, online rental platforms are also facing challenges, generating massive amounts of user data every day. How to dig user behavior information from these user data has become an important goal of the rental platform. How to collect, store, organize, and analyze the behavior data of these users is a problem that the big data system needs to solve.

This thesis analyzes the user behavior analysis model of the world Internet industry. It is proposed to design a set of rental user behavior data analysis system under the background of the rapid development of the current rental market. Hadoop distributed big data framework is adopted. Webmagic framework is used for website data crawling, log files are collected through kafka, Flume and Mysql is used for data persistent storage. The back-end project of the system is based on Maven construction and unified standard management, using three frameworks of SpringBoot + SpringMVC + Mybatis for development. The front end adopts vue for development, finally realizes the visual big data system on the web page. After the completion of the system development, a number of tests were performed on the system to verify the stability of the system.

(1) Introduction to the background and significance of system design, service framework and middleware technology used in system design.

(2) Feasibility analysis, functional analysis and non-functional requirement analysis of the system were carried out.

(3) Detailed design of the collection, analysis and visualization functions of renting user behavior data.

(4) Realize the main functions of the system, data collection, data warehouse, data visualization functions and provide the final renderings.

(5) In the software testing part, this thesis carried out unit testing, interface testing and functional testing on each key module, verifying that the software result data is consistent with expectations.

Key Words: Big data, e-commerce, user behavior, data analysis

目 录

摘 要	I
Abstract	III
1 引 言	1
1.1 背景及意义	1
1.2 国内外研究应用现状	1
1.2.1 研究应用范围	1
1.2.2 国内外的应用现状	2
1.3 论文主要内容	3
1.4 论文组织结构	4
2 平台核心组件技术介绍	5
2.1 Hadoop 架构	5
2.1.1 HDFS 架构原理	5
2.1.2 MapReduce 计算框架	7
2.1.3 Yarn 资源调度	8
2.2 Flume 原理	10
2.3 Kafka 原理	11
2.4 SpringBoot、SpringMVC、MyBatis 框架	12
3 系统需求分析与实现	15
3.1 功能需求分析	15
3.2 非功能需求分析	16
4 系统设计与技术选型	17
4.1 系统设计	17
4.1.1 系统功能设计	17
4.1.2 数据采集平台设计	17
4.1.3 数据仓库设计	18
5 系统实现	19
5.1 数据采集模块实现	19
5.1.1 Hadoop 安装部署	20
5.1.2 WebMagic 安装部署	21
5.1.3 Zookeeper 安装部署	21
5.1.4 Kafka 安装部署	22
5.1.5 Flume 安装部署	22
5.2 数据分析模块实现	23

5.3 数据可视化模块实现	24
6 系统测试	26
6.1 单元测试	26
6.2 接口测试	26
6.3 功能测试	27
7 结 论	30
参考文献	31
在学取得成果	34
致 谢	35

1 引 言

1.1 背景及意义

随着互联网技术的发展与普及,人们的生活已经离不开网络,传统行业的运作模式受到巨大的冲击^[1]。许多传统行业开始了线上与线下相融合的运作模式,不仅在线下有人力销售,线上也有专门的网站负责经营销售,租房行业也在其中。在过去,租客想要找到合适的房屋,基本只能靠房屋中介、路边小广告,报纸,到当地寻找和他人推荐等方式,这些方式都会占用大量的人力和时间成本。租房互联网化后,传统的人力管理方式变成了由计算机科学管理,房屋信息管理更加高效和科学。租客能在网上随时随地的选择自己喜欢或者需要的房源,省去了中介费和服务费,更加方便快捷^[2-4]。随着在线租房平台的发展,用户数据的规模越来越大,结构也越来越复杂,所以对数据进行整理和计算,将之转化成更加可视化的和有更高利用价值的数据形式,对租房平台来说极其重要。多年来积累的用户数据包含着在数据量小时无法得到的研究价值,大数据分析能够让租房平台对自身有更全面的认识,平台受欢迎的地方,平台接下来需要改进的地方,这都可以通过数据分析结果表示出来。

在这种背景下,建立一个用户行为分析系统势在必行^[5]。对于租房平台而言,用户行为分析系统能够将用户数据经过计算和研究得出一系列的行为指标,帮助了解平台各方面的运作情况,降低平台运营成本、增强与租客间的联系、及时获得市场反馈,精确地制定推荐商品和投放广告,为平台创造更多的收益价值。对于用户而言,不同的用户有着不同的需求,但是用户的时间非常宝贵,这就需要平台动态地多样化地为用户推荐感兴趣的房屋。用户行为分析系统能对用户的消费数据进行分析,不仅能够通过动态分析了解到用户消费行为取向,也能为企业投入广告制定精准决策,针对用户喜好进行产品推荐,防止市场上人员出现盲目性营销。

1.2 国内外研究应用现状

1.2.1 研究应用范围

随着互联网的发展与进步,以用户数据为基础而开发的各种平台或系统越来越多,用户行为数据分析技术在各个领域都得到了广泛的应用。目前行

业的数据规模逐渐扩大，需要处理的数据结构也变得更加复杂，处理数据的方法也需要不断的进行改进，适应新的行业模式。如何将这些用户行为数据清洗、去重、合并、整理，留下有价值的数据，是目前行业最为关注的问题[6]。

数据分析是指采集大量的数据，并通过合适的统计分析方法处理数据，从处理结果中获取高价值信息的过程，是一种重要的大数据技术。具体来说，数据分析首先要建立模型分析具体的数据结构和内容，通过日志或埋点采集产生的数据存储到操作系统中，再对数据进行清洗，计算，归并，整理等操作，然后将结果数据与计划中的理想数据进行比较，最后查找不同之处，寻找问题的原因，发现新的行业线索。通过数据分析技术，我们可以从散乱的数据中提取和精炼出有价值的信息，从而找出研究对象间的联系[7]。

在众多租房平台的运行过程中，正确的数据分析结对平台的发展来说起到非常重要的作用。例如可以根据对用户租房的历史数据进行分析，得出在本城市中哪些地区房源比较受欢迎，哪些房屋类型有升值价值，或者那些年龄段更有租房实力等等信息。得到这些数据后，平台可根据这些结果数据对网站做出改进，不断丰富平台内容，使平台更加符合用户的使用偏好、提升用户体验，吸引更多用户使用，这对租房平台来说是最重要的核心。因此，数据分析是促进社会发展不可缺少的技术，具有非常优秀的发展和应用前景。

1.2.2 国内外的应用现状

用户行为分析系统需要有数据的采集、存储和计算功能。Hadoop 是一个由 Apache 基金会所开发的分布式系统基础架构，广泛应用于大数据的处理和计算领域[8]。庞双玉使用 Hadoop 框架搭建了一种基于 Hadoop 的用户行为分析系统[9]，张思航使用 Hadoop 对电信大数据进行处理，分析了传统关系型数据库查询海量数据时的不足[10]。Hadoop 核心的设计就是 HDFS (分布式文件系统)和 MapReduce 编程模型。HDFS 是一个分布式的存储方案，MapReduce 提供了大数据分析和计算的功能。除此之外，Hadoop 框架还包含 Pig, Avro, Hive, HBase, ZooKeeper 等众多项目[11]。由于 Hadoop 具有高效性、稳定性、低成本和高扩展性等优点，使得其在国内外许多领域都有着广泛的应用。

在国内，随着互联网和大数据的研究应用日益广泛，各大企业开始逐渐关注大数据技术的应用。2008 年，阿里巴巴便开始研究 Hadoop 平台，尝试使用在电商订单系统里。百度在很早就开始关注 Hadoop 并开始研究和开发，在 2012 年大规模使用 Hadoop 来解决搜索引擎问题，目前数据容量已经超过

了 100 PB，每秒获取数万个请求。Hadoop 也在百度其他团队里提供数据的存储与计算服务。腾讯使用 Hadoop 也比较早，腾讯的大数据平台使用 Hadoop 的生态组件搭建云服务器，还构建了自己的数据仓库系统和开发环境，兼顾了平台稳定性和技术先进性。华为的 Fusion Insight 大数据平台是国内比较领先的企业版 Hadoop，和开源完全兼容，针对开源版 Hadoop 的缺陷，提出自己的解决方案，提高了 Hadoop 的可用性^[12]。华为云和大快搜索的 DKhadoop 面向移动联通运营商和银行客户，做了很多 sql 方面的定制，集成度了 CDH、管理也很方便。

在国外，2004 年，谷歌通过论文介绍了 MapReduce，接着进行了开源，Hadoop 开始初步形成。2005 年，雅虎公司正式引入了 Hadoop，将其集成到 SQL 系统中和多个软件进行数据交互，实现了即插即用的功能，完善了搜索引擎的问题。亚马逊拥有市场上应用最广的 Hadoop 平台，拥有多家合作伙伴并为其提供平台以外的额外服务，例如对数据进行查询、建模、集成和管理。Cloudera 使用开源分布式的 Hadoop 作为其分布式系统的基础，用在 Apache 项目中的很多方面。目前有超过 200 个付费用户稳定的运行在 Cloudera 平台上，有一些用户在其平台上管理 1000 多个节点超过 1PB 的数据。2014 年，IBM 成为了 Hadoop 项目服务在全球最大的供应商，有超过 100 个 Hadoop 部署用户，且很多用户都是 PB 级别的数据。英特尔也利用并优化了 Hadoop 版本，使其能够更好的运行在英特尔的硬件上。微软不仅运行 Hadoop 在 Windows 机器上，而且还把代码提供给开源项目，以促进 Hadoop 的生态系统更广泛的发展。微软将自研的 Hadoop 产品放在公有云 Azure 上，定制“Hadoop 即服务”的产品。微软还有一些其他的项目，包括“Polybase”，能够实现通过熟悉的 SQL 语句查询 Hadoop 的数据的能力。近几年 Facebook、Apple、Google 等公司不断在使用 Hadoop 搭建大数据分析平台，并根据数据结构对自身优化调整，对行业产生了良好的示范效果^[13]。

1.3 论文主要内容

本文主要进行了租房用户行为分析系统的设计与实现。传统数据分析通常在单台机器上运行，当处理大量数据时，机器的性能极大地限制了系统的运行速度。大数据场景下数据分析量大、数据结构复杂难以分析，为解决问题，本文在深入分析大数据分析平台的相关技术下，研究了 Hadoop 的相关技术、消息中间件和后端开发框架的原理，根据现实需求分析设计划分出多个系统模块，实现了数据采集，数据仓库和数据可视化功能。提出了基于 Hadoop 的租房用户行为分析的系统方案，帮助租房平台使用有价值的数

据进行决策分析，更好地发挥社会价值。本系统利用爬虫采集用户在租房网站或 App 上产生的行为数据，并通过消息缓存中间件 kafka 和 flume 存储到多台机器的 HDFS 上，通过 MapReduce 计算处理，使用 Hive 对数据进行数据清洗和分层分表，存储在 MySQL 数据库中，在后端使用 SSH 框架完成接口编写，在前端使用 vue 和 Echarts 进行 web 页面可视化展示。最后进行了测试，验证了系统的可行性。

本系统基于大数据环境搭建，对大数据在实际生产中的应用进行了全景展现。在搭建过程中也对 Hadoop、Hive、Sqoop、Flume、Kafka 等大数据工具进行了大量的调查研究，对租房领域常见的重难点指标进行了实现具体指标包括：房源所在城市和地区，各区域租房平均单价，房屋布局和租金数据，房屋配套设施数据统计，房屋可用面积统计，房屋支付方式等。

1.4 论文组织结构

本文的结构安排如下：

第 1 章主要介绍论文的研究背景和意义，国内外研究现状、研究内容以及论文的组织结构。

第 2 章主要介绍了用户行为分析系统的相关技术，包括 Hadoop 相关的组件和消息缓存中间件，还有系统开发中使用到的 SpringBoot、SpringMVC、MyBatis 框架。

第 3 章对用户行为分析系统进行需求分析，从功能需求方面对系统的数据采集功能、分析功能、可视化功能详细说明，最后从可靠性、扩展性、易用性等方面对系统进行了非功能需求分析。

第 4 章为系统的需求设计部分，阐述了系统的总体功能设计、业务设计和技术选型。业务模块的设计主要包括数据的采集、数据的传输、数据清洗、数据的分析、数据可视化展示等。

第 5 章主要对用户行为分析系统进行分模块搭建，按照前述内容中对系统涉及描述将数据仓库从数据的采集到最后需求的实现都进行了详细实现介绍。

第 6 章主要对用户行为分析系统进行测试，包括单元测试、接口测试和需求展示测试等。

第 7 章对系统的现状进行了总结与展望，并指出其中的不足及需要改善的地方。

2 平台核心组件技术介绍

本章对系统研发过程中用到的核心组件进行简单介绍，本次数据分析系统平台研发主要用到的组件有 Hadoop、Flume、Kafka、MySQL，并对用户行为分析系统搭建过程中用到的 SpringBoot、SpringMVC、MyBatis 框架进行了介绍。

2.1 Hadoop 架构

Hadoop 是由 Apache 基金会开发的一个分布式系统架构，操作者不需要了解底层原理就可以解决大数据场景下各种问题。开发者可以使用简单的编程模型在分布式的计算机环境中高速运算和处理大数据^[14]。

Hadoop 具有开源免费，高稳定性，高容错性，可扩展性等特点。Hadoop 是一个在 GitHub 上的免费框架，不需要像其他的分布式框架那样购买特定的机器，在通常的商用机器上就可以运行。稳定性体现在采用了环形链表的计算方式并且在所有节点上多线程计算对数据进行处理，保障数据稳定完成。Hadoop 中每份文件都会复制 2 份副本，存储在不同的机器上，在任务失败时重新分配任务，配合节点间心跳监控和备份机制，保证了 Hadoop 的容错性。Hadoop 的扩展性很强，节点间以主从的模式进行计算，只需要增加 slave 节点就可以调整集群架构，增加整体的计算能力^[15]。Hadoop 框架的主要运用情形有：大数据分析，搜索算法系统，广告推荐系统，商品推广系统，数据监控系统等^[16]。

对于 Hadoop 来说，其核心组件为 HDFS（数据存储）、MapReduce（数据计算）、Yarn（资源调度）和 Common（辅助工具）共四个部分。

2.1.1 HDFS 架构原理

单台机器理论上可以通过增加磁盘量的方式无限存储数据，但是单机的存储引擎限制了数据读写的速度，同时没有备份，存在数据丢失的风险，在现代的企业环境中并不适用，需要分布式存储。分布式的系统间通过网络设备互联，通过系统间联调对外提供存储服务，但是分布式会存在许多异常问题，分布式场景下数据往往有很多副本，带来了一致性问题。分布式的网络环境很复杂，不只有一台机器，本地的数据库操作依赖第三方系统的结果，带来了分布式事务的问题。这些问题给管理和维护带来了困难，这就需要一

种系统或框架来管理分布式系统的文件来统一管理分布在集群上的文件系统。HDFS 是一种分布式文件管理系统。HDFS 通过数据流的访问模式来存储大量文件，即通过一次导入，多次读取的存储方法。由于主要运用在较大的数据集上，HDFS 适用于存储数据量 TB 级的文件，不适合使用在存储较小的文件上。当存储大型数据文件时，HDFS 将其划分成多个定长块，系统内部按照这些块来组织数据，每个数据块的大小基本相同(大小为 64 或 128MB)，大型文件可以随意存储在多个集群中，不需要担心数据容量的限制^[17]。HDFS 的架构如图 2-1 所示：

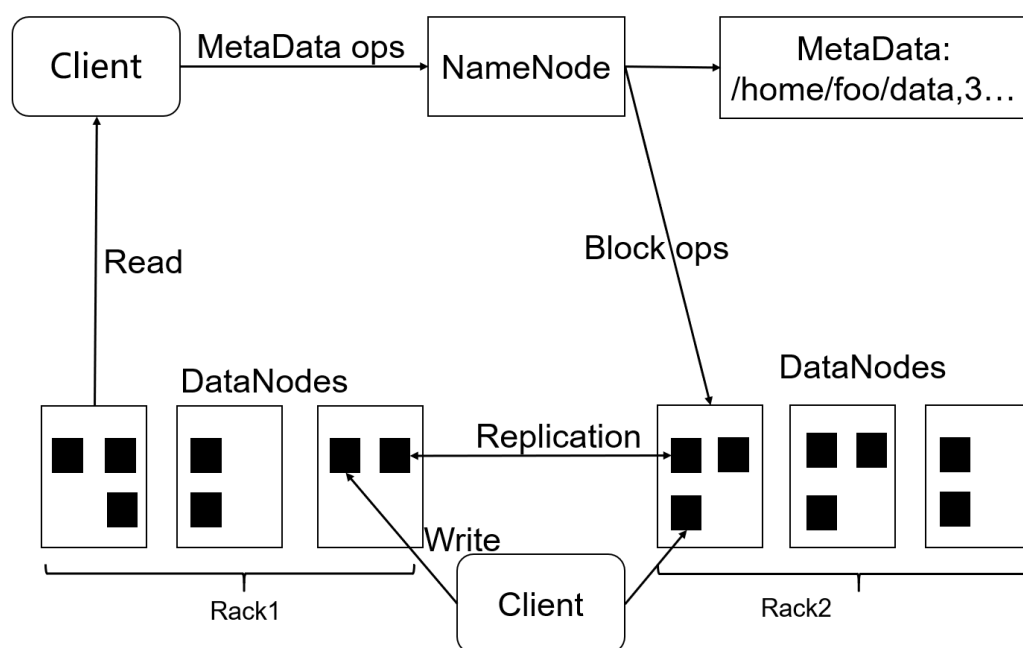


图 2-1 HDFS 架构

为了支持数据流的访问模式，HDFS 有两类节点，以管理者-工作者模式运行。其中，NameNode（命名节点）记录文件中各个块所在的数据节点信息、索引信息等。DataNode（数据节点）负责存放要被存储的数据，把实际数据发送到每个 DataNode 节点上，让 NameNode 负责调用运行。通常集群以一个数据节点 NameNode 与多个数据节点 DataNode 组合的方式运行。NameNode 节点中虽然负责存储数据块的信息，但是这些信息会随时改变，在系统启动时由数据节点会更新数据块的信息发送给 NameNode 节点。NameNode 实现了 ClientProtocol 接口，可以被 DataNode 使用，重复的调用执行 DFS 数据块。当节点和其他节点之间长时间没有通信，NameNode 会将其标记为宕机，此节点的数据不再被集群共享。NameNode 服务于获取块位置的请求，不断检测这些数据块，获取到复制的请求时就会重起复制操作^[18]。

任何数据存储系统，都会有文件缺失和损坏的风险。HDFS 将数据复制多份，通过 NameNode 和 DataNode 来标记存储这些文件。当某块数据出现错误时，系统会根据 NameNode 上的索引读取距离错误文件节点最近的节点上的数据，将其复制并覆盖掉原数据^[19]。

副本存储策略也是分布式系统中的一个关键点。HDFS 采用了一种节点距离感知的策略来优化副本存储的问题。HDFS 默认没有开启节点距离感知策略，为了提升系统的可用性和网络的利用率，用户可以在配置文件中开启策略^[20]。HDFS 中的集群利用这种距离感知的策略，将数据运行在多台机器上。在此策略基础上，NameNode 可以确定每个 DataNode 节点所在机器的唯一 ID，然后将副本暂存在不同的机器上。当某台机器发生错误，数据损坏时，不至于无法复原，保障了数据的完整性。

2.1.2 MapReduce 计算框架

MapReduce 是一个基于 Hadoop 的数据分析计算的核心框架，该组件启发于 Lisp 语言中的 Map 与 Reduce 功能^[21]。MapReduce 将数据处理分为两个阶段：Map（映射函数）和 Reduce（归约函数），每个阶段都有一个核心线程函数。Map 和 Reduce 两者的数据以键值对的形式存储和运行。MapReduce 的工作流程如下图 2-2 所示：

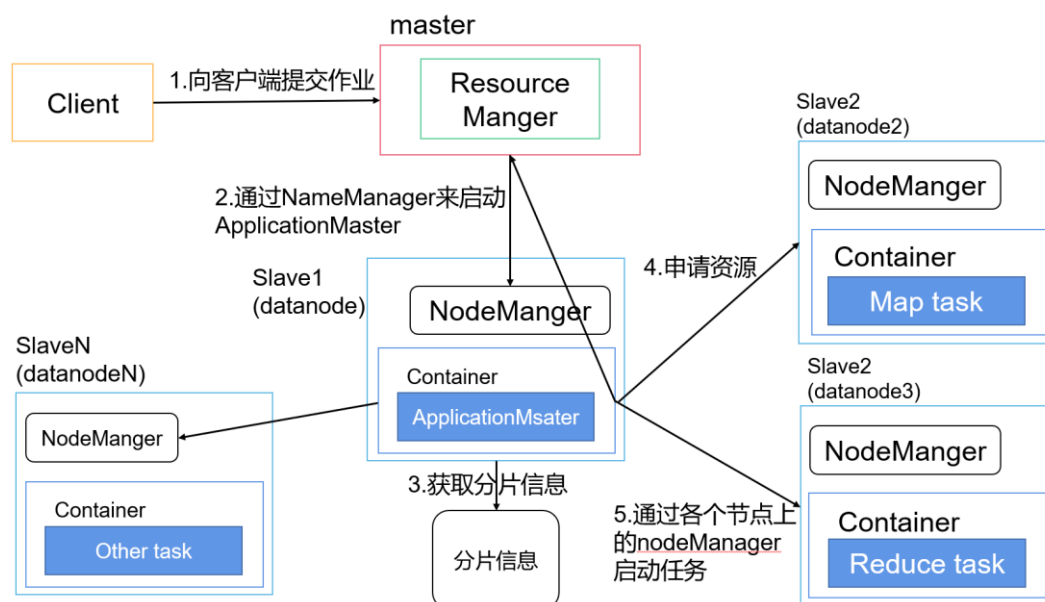


图 2-2 MapReduce 工作流程

MapReduce 处理数据过程主要分成 Map 和 Reduce 两个阶段。MapReduce 框架首先根据数据的存储位置，把数据分成多个分片，每个 Map 在读入各自

的数据后，在多个结点上并行处理。接着框架调用 **Map** 函数收集键值对放到内存缓冲区中，调用 **Partitioner** 对这些中间结果按照键值进行排序，排序之后如果有相同键值，则会进行合并，合并的效果可以自定义。中间结果会按照键的范围划分分区，并行发给任务结点，**ReduceTask** 根据分区号，去各个任务节点上取相应的结果数据，再将这些文件进行归并排序，合并成大文件。**Map** 和 **Reduce** 的处理逻辑由用户自定义实现，但要符合 **MapReduce** 框架的约定，输出数据格式写入 **HDFS** 中^[22]。

MapReduce 框架具有开发简单，高扩展性，高容错性和离线处理的特点。**MapReduce** 通过简单地文件配置注入，就可以自动完成生成分布式程序，可以通过脚本部署到机器集群中运行，这使得系统开发变得更加快捷。**MapReduce** 只需要增加机器就可以扩展计算能力，体现了它的高扩展性。**MapReduce** 有很高的容错性，集群中的机器通过类似心跳检测的方式进行连接，当一台机器发生故障时，集群会自动转移计算任务，使得任务可以继续运行，不需要人力维护。**MapReduce** 可以自己离线处理处理数据存储、节点通信、任务调度、容错处理等工作，适合 **PB** 级别以上当数据计算存储。

MapReduce 框架适合处理离线数据，实时计算和流式计算更适合使用 **flink** 框架。**MapReduce** 查询结果需要访问多层节点，无法快速返回结果。**MapReduce** 由于自身设计原因，要求的输入数据必须是静态的，无法处理动态的流式计算数据。**MapReduce** 同时不太适合有向图计算，有向图里各个数据存在依赖关系，输出有对应的下一级输入，在这种情况下，输出结果无法在单个内存中计算，集群间数据调用产生大量逻辑操作，性能十分低下^[23]。

2.1.3 Yarn 资源调度

Yarn 是一个分布式的资源管理和作业调度技术，相当于一个分布式的操作系统平台^[24]。**Yarn** 在 **MapReduce** 的基础上发展而来，负责为集群分配系统资源给软件，调度不同节点上执行的任务。**MapReduce** 是传统的大数据计算框架，存在单点故障、资源利用率低等问题。**Yarn** 将工作追踪模块划分成两个独立的进程：全局的资源管理（**ResourceManager**）和作业管理（**ApplicationMaster**），在单台机器磁盘的粒度上操作资源，有效增加系统的资源利用效率^[25]。

Yarn 采用 **Master/Slave** 的工作方式，整个体系结构主要由 **ResourceManager**、**NodeManager**（节点管理）、**ApplicationMaster** 和 **Container**（容器）等组件构成，如图 2-3 所示。

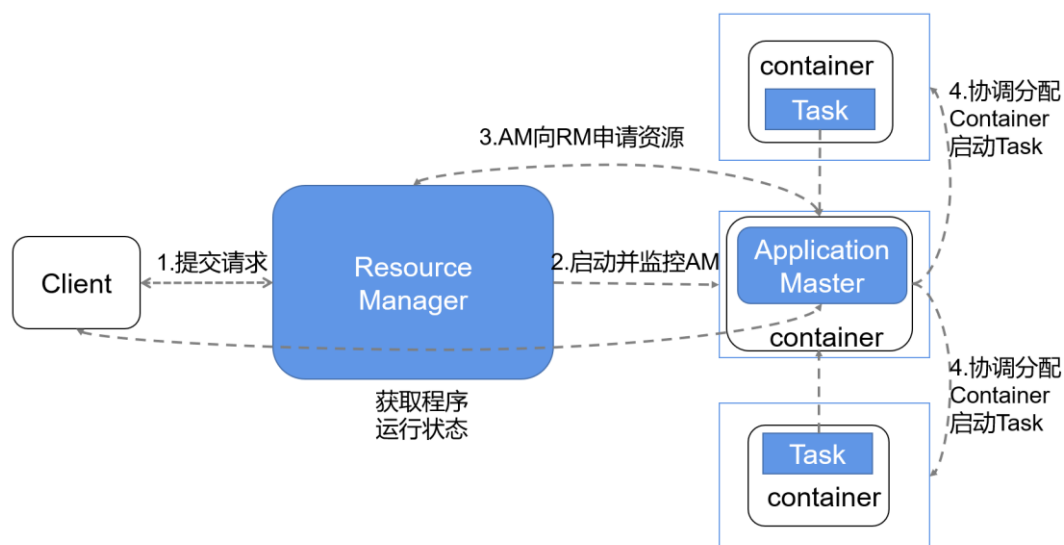


图 2-3 Yarn 组件构成

ResourceManager 负责资源管理和任务调度，只关注资源分配，根据需求合理分配资源。调度器负责分配系统资源，可以根据申请的需要，在指定机器上申请资源。应用管理器负责动态分配资源，每个应用有自己的生命周期管理机制，资源的使用期限、访问权限被记录在内存中，资源利用率更高。

ApplicationMaster 负责单台机器的数据管理和信息监控，同时负责本机器内的容错。节点根据应用的状态来判断根据数据是否应该申请资源，通过 **ApplicationMaster** 申请资源，向 **ResourceManager** 报告数据同步情况和任务进程，同步跟踪任务状态。

NodeManager 是节点和任务的资源管理器，负责管理 Hadoop 集群中的节点资源调度。**NodeManager** 向 **ResourceManager** 注册自己，接受初始化，清理资源等命令。处理 **container** 的工作请求，上传结点健康状况和当前运行状态。

Container 负责封装节点资源，是系统最基本的资源单位。**Container** 可以加载任意语言的程序，一个节点可以有多个 **Container**。**YARN** 会为每个节点动态申请和释放 **Container**。

Yarn 的工作流程主要分为以下步骤：**Yarn** 从 **ResourceManager** 中申请工作任务，检查输入和输出配置，复制数据资源到 **HDFS**。**ResourceManager** 将资源路径递交给任务调度器，为作业分配 **Container**，加载主进程并申请运行 **MRAppmaster**，接着创建监控进程来跟踪作业的进度，将用户的请求初始化为一个 **Task**，如果 **Task**，则直接在同一个 **JVM** 下运行。**NodeManager** 根据 **Task** 分配情况，创建容器 **Container**，从 **HDFS** 上读取任务所需资源，然后复

制资源执行任务。**ResourceManager** 将正在运行的任务提交给 **NodeManager**，每隔一段时间将任务的进度报告给 **ApplicationMaster**。**NodeManager** 根据数据状态启动 **MapTask** 对数据分区排序。**Client** 定时检查整个作业是否完成，程序运行完毕后，**ResourceManager** 会向机器申请注销自己的资源，清空临时文件、目录等。

2.2 Flume 原理

Flume 是一个分布式日志采集和传输系统。数据可以在 **Kafka** 中提取，并且可以使用复杂的算法来处理；**Flume** 能够对数据进行预处理，传输到指定数据流当中。**Flume** 能够提供日志收集功能，读取服务器传来的数据，将数据写入磁盘或下一个数据流^[27]。**Flume** 组成架构如图 2-4 所示：

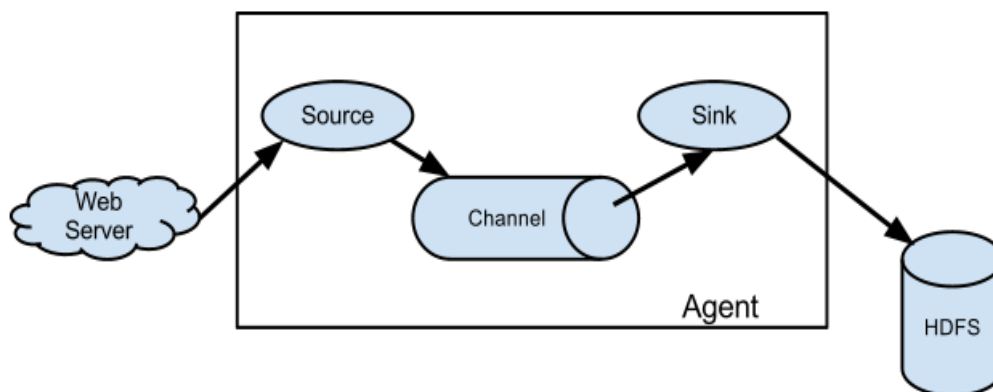


图 2-4 Flume 组成架构

Agent 负责将数据源的数据发送给 collector，实质上是一个 **JVM** 进程，网络服务器的数据通过事件的形式发送到 **HDFS** 中，是 **Flume** 数据传输的基本单元。**Source** 是 **Agent** 的一部分，负责接收 **Avro** 端口产生的数据。**Source** 可以通过修改配置监听不同的端口，只需要更改 **type** 就可以处理不同格式的日志数据。**Channel** 主要是用来缓冲 **Agent** 来不及输出的数据。**Channel** 引入了锁的概念，同一时刻只会有一个 **Sink** 读取数据。**Memory Channel** 是内存队列，在机器重启、程序结束时数据会因为内存重写导致数据丢失，不经常使用。**File Channel** 是磁盘队列，所有的操作都会以日志形式写到磁盘。不会发生丢失数据的情况。**Sink** 是 **Flume** 的尾端接口，从 **Channel** 中取出数据，然后程序读写操作完成后将数据发到下一数据流。**Sink** 支持事务，在数据被推送到下一阶段前数据都是安全的。**Sink** 通过异步通知的形式通知 **Channel**，利用 **Channel** 提交事务。事务提交后就可以从 **Channel** 中删除这个事务。

Flume 中 Event 会随着数据从源头送下一级输入。Event 是数据传输的基本单元，分为响应头和响应体两个部分。响应头以键值对的数据结构存储，包含了 Event 的编码、类型、时间、名称和节点信息等属性，响应体使用数组形式存储，一般用来存放数据的具体内容。

2.3 Kafka 原理

Kafka 是由 LinkedIn 开发的一个分布式的消息系统。Kafka 可以作为消息管理系统，存放接口调用或网页动作生成的行为日志，这些文件可以通过数据流传输到 flink 进行实时监控，或者将日志异步发送到 kafka 集群中，在磁盘中统计分析。Kafka 具有高流量，耐用性，较高的容错能力，这使得它被广泛应用于大数据实时处理领域。现在越来越多的分布式系统如 Cloudera、Spark、Apache Storm 等都支持 Kafka 集成^[28]。Kafka 的基础架构如图 2-5 所示：

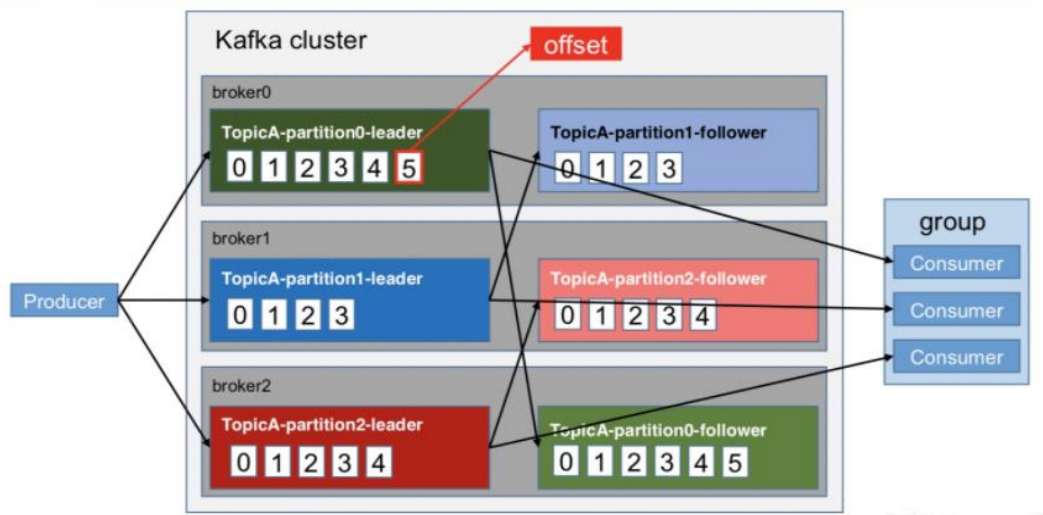


图 2-5 Kafka 基础架构

生产者和消费者（producer 和 consumer）：消息的发送者叫 Producer，消息的使用者和接受者是 Consumer，生产者将数据保存到 Kafka 集群中，消费者从中获取消息进行业务的处理。消费者组由多个 consumer 组成。消费者组内每个消费者负责消费不同分区的数据，一个分区只能由一个消费者消费；消费者组之间互不影响。所有的消费者都属于某个消费者组，即消费者组是逻辑上的一个订阅者。

broker: Kafka 集群中有很多台 Server，其中每一台 Server 都可以存储消息，broker 可以容纳多个 Topic。将每一台 Server 称为一个 kafka 实例，也叫

做 broker。

topic: 一个 topic 里保存的是同一类消息, 相当于对消息的分类, 生产者和消费者面向一个 topic。每个 producer 将消息发送到 kafka 中, 都需要指明要存的 topic 是哪个, 也就是指明这个消息属于哪一类。

Kafka 中数据的生产和消费都是基于 Topic 进行分类的, 在队列消息中具有相同 Topic 组的一组消费者将订阅主题。Kafka 使用分布式提交日志, 数据在大多数情况下都会保存磁盘上, 因此它是持久的。提交日志文件中存储了 producer 生产的数据。新生成的日志数据会被添加到日志文件的末端, 每条数据都有不同的位置偏移量。消费者会定时读取消费的日志数据偏移量, 出现错误时可以快速恢复, 继续消费流程^[29]。

2.4 SpringBoot、SpringMVC、MyBatis 框架

在以前的系统开发中, 无论项目大小, 都需要配置大量的 XML 文件, 大量配置直接影响了开发效率, 无形中也增加了系统的维护难度^[30]。SpringBoot 框架大大的简化了应用开发之中所需要的配置, 它的设计理念在于“预定大约配置”, 使得开发过程中不需要配置大量 XML 配置文件^[31]。SpringBoot 内嵌了 tomcat, 不需要部署程序到 tomcat 中运行, SpringBoot 极大简化了程序的搭建、配置、开发和部署等步骤^[32]。SpringBoot 从开始到事件启动的运行流程图如图 2-6 所示。

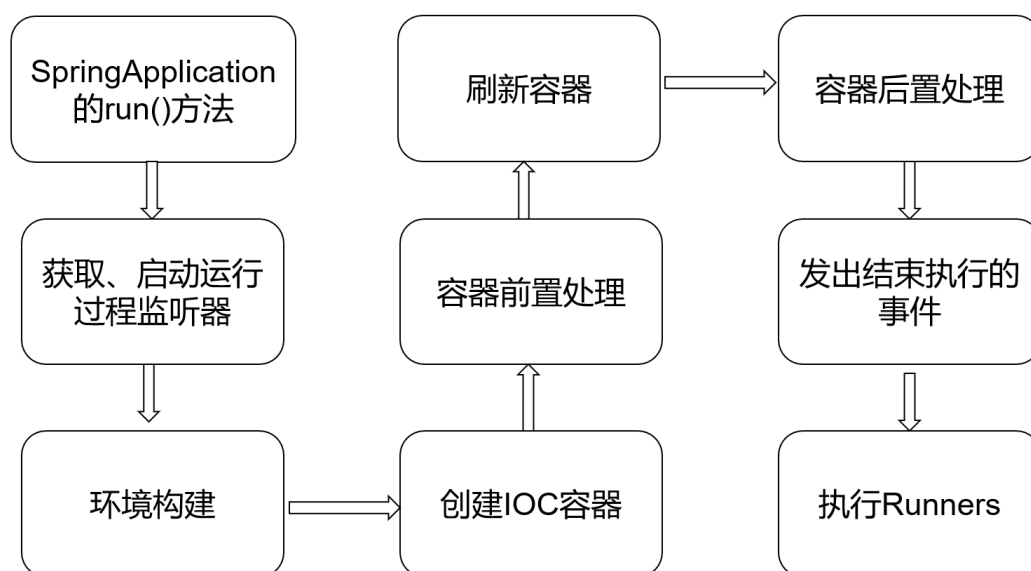


图 2-6 SpringBoot 运行流程图

SpringBoot 通过注解和事件监听的方式完成 bean 的自动配置, 在 Maven

中导入相关依赖后，SpringBoot 只需要运行 application 就能启动整个项目，同时 SpringBoot 集成了很多开发框架，比如 tomcat，maven 等，开发者可以在不配置 xml 文件的情况下开发应用^[33]。

SpringMVC 是 Spring 框架的一部分，是基于 MVC 模式设计实现的表现层框架^[34]。SpringMVC 能接受用户传递的 HTTP 请求，经过处理后，将结果返回前端进行展示。它将视图层（View）、逻辑控制层（Controller）、数据层（Model）三个层进行了分离，降低了耦合度、提升了开发效率。SpringMVC 工作原理图如图 2-7 所示。

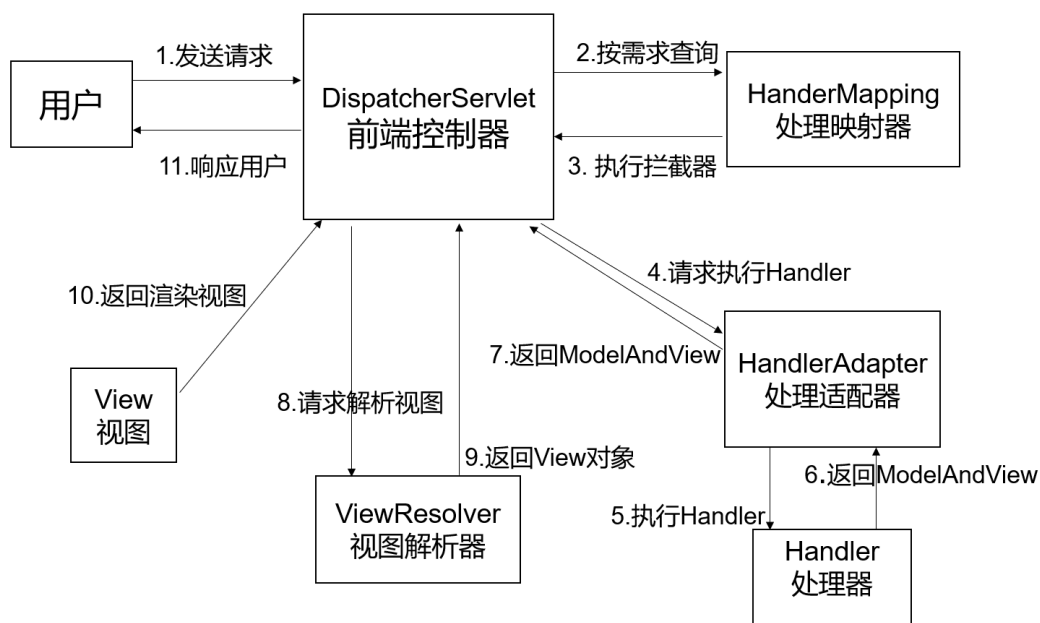


图 2-7 SpringMVC 工作原理

SpringMVC 的核心是 DispatcherServlet（前端控制器），整个框架都围绕它设计，在流程中起到了控制中枢的作用。SpringMVC 通过请求驱动，收到前端发来的 URL 后，将具体内容转发给 HandlerMapping（处理映射器），获取该处理器配置的所有信息，然后返回 HandlerExecutionChain（处理器执行链）^[35]。返回的处理器被前端控制器获得，根据返回的信息匹配到合适的 HandlerAdapter（处理适配器），然后向前端控制器传递 ModelAndView（模型视图）对象^[36]。前端控制器根据返回的对象数据请求 View Resolver（视图解析器）。视图是模型数据的封装对象，通过以 json 或键值对的形式返回，后端将名称解析为视图对象，返回视图渲染前端页面，SpringMVC 返回模型视图，具体的对象处理在前端完成。

MyBatis 是一种商业流行的持久层框架。一般系统的开发主要是在写逻辑的结构基础上对数据库数据进行增删改查的操作，称为持久层的开发。因

为经常访问数据库，随着技术的发展，为了简化持久层开发，对原来的数据库的操作进行了封装，产生了很多持久层框架。Mybatis 框架就是其中一种，基于 SQL 语句编程，开发方便，灵活简单，可以直接操作 SQL 语句实现对数据库的操作，能够与 Spring 很好的集成^[37]。MyBatis 的工作流程图如图 2-8 所示。

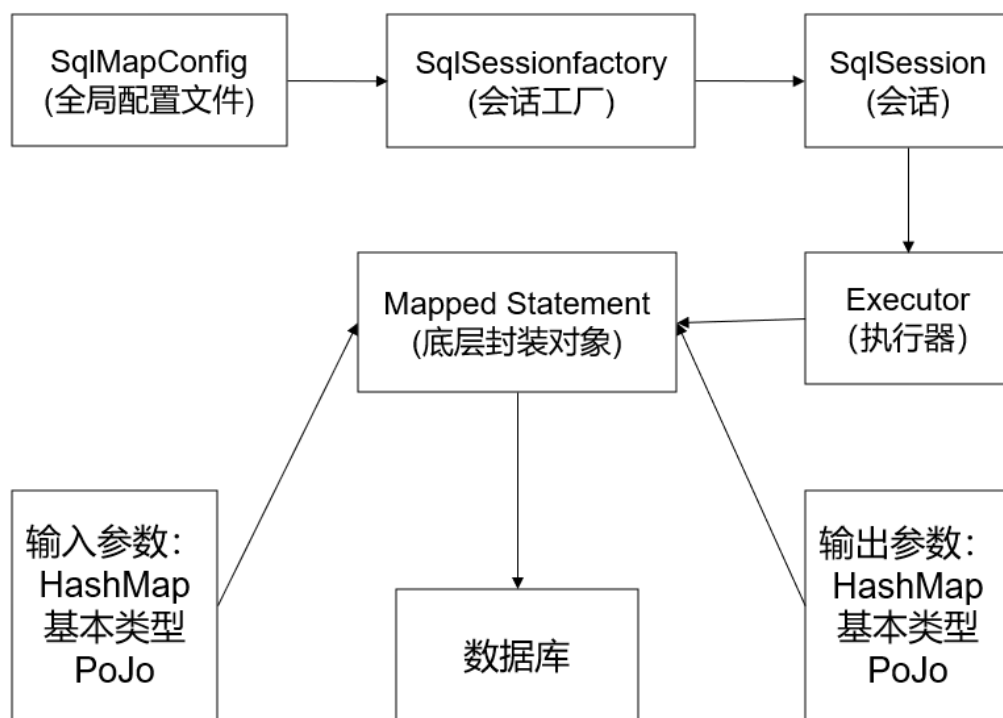


图 2-8 MyBatis 工作流程图

对原生 JDBC 封装后再对数据库进行访问，全局配置文件中配置数据库的信息。会话工厂目的是创建会话，会话可以理解为一个接口，完成对数据库的基本操作，底层 Pojo 对象对参数进行封装。MyBatis 与 SpringBoot 结合使用可以更好地提高开发效率，使开发更加简便快捷。SpringBoot 底层对 Mybatis 进行了封装，在开发过程中，只需配置数据源信息即可^[38]。

3 系统需求分析与实现

本章主要对租房用户行为分析系统进行需求分析。从需求方面对系统的数据采集功能、分析功能、可视化功能三方详细说明，解决了房屋出租系统的功能需求，最后从可靠性、扩展性、易用性和约束对系统进行了非功能需求分析。

3.1 功能需求分析

如图 3-1 所示，租房用户行为分析系统主要分为 3 个功能结构，分别是数据采集平台、数据仓库平台和数据可视化平台。

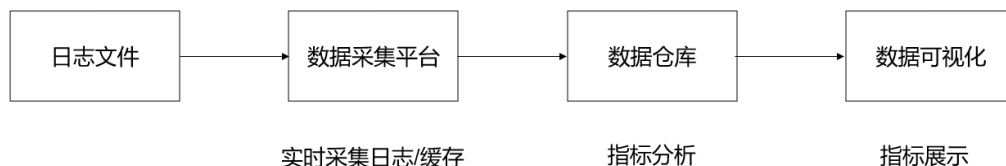


图 3-1 系统架构图

1) 数据采集平台

要求实现用户行为数据采集功能。采集租房网站或 App 上的用户行为数据，然后收集到大数据存储系统 Hadoop 中。在采集日志时要求时效性，同时具有缓存功能。

2) 数据仓库平台

要求实现数据分析功能。整合服务器采集到的数据，对 Hive 中的数据进行分析处理。分别对用户的行为数据进行保存、清洗、合并、拆分、统计等工作。最终要求实现如下需求：

- 统计最新发布房源所在城市、地区
- 各区域房源租金，支付方式
- 各区域平均房屋租金和面积
- 各区域平均租房单价
- 各区域房屋类型

3) 数据可视化平台

要求在前端 web 页面显示统计数据。将最终需求结果数据导入到 MySQL 中，利用后端 spring 和 mybatis 框架启动服务供用户使用并在前端对数据进

行 Web 页面展示。

3.2 非功能需求分析

租房用户行为分析系统需要实现的目标如下：

- 环境搭建完整，技术选型合理，框架服务分配合理；
- 信息流完整，包括数据生成、数据采集、数仓建模；
- 能应对海量数据的分析查询

租房用户行为分析系统包含了各个城市的房源数据，这些数据需要有稳定的服务器和备份机制来防止数据损坏同时随着数据量的扩大，系统随时可能会扩容，这就需要系统有良好的扩展性；系统也要适合租房平台的业务人员，容易上手,减少重复开发.本系统的非功能性能需求主要集中在可靠性、可扩展性与易用性等几个方面。

1) 可靠性

系统在运行的过程中机器可能发生错误和重启，数据可能会因此丢失。所以要求系统具有一定的可靠性，能够复制备份文件信息，将文件分开存储，出现异常时，及时抛出异常，不影响后续程序的运行。错误信息和系统日志要保存到磁盘文件中，用户可以根据时间查询某日的日志数据。

2) 可扩展性

每天都会有数百万用户访问租房平台，这些用户行为数据不断在系统中增加日志，需要不断增加新服务器来存储和计算数据，并且随着系统的不断完善，更多功能也会被添加进来，这些都要求系统具有较高的可扩展性。

3) 易用性

系统应该方便不懂程序当运营人员维护和更新，页面需要有方便的切换按钮和直观的数据显示。在设计系统的使用上，应该有详细当设计手册和接口文档，使得开发人员能够快速了解系统架构，方便后续开发使用。

4) 其他非功能性约束

敏感数据不应该直接展示给用户，需要设置权限管理用户的界面操作。用户每次访问的数据也应该有缓存功能，减少数据库重复查询。在编写代码时，要统一代码风格，添加必要的注释等。

4 系统设计与技术选型

本章主要对租房用户行为数据分析系统进行系统设计，根据需求进行总体功能设计、业务设计和技术选型。

4.1 系统设计

4.1.1 系统功能设计

如图 4-1 所示，该数据仓库系统主要分为 3 个功能结构，分别是数据采集平台、数据仓库平台和数据可视化平台。

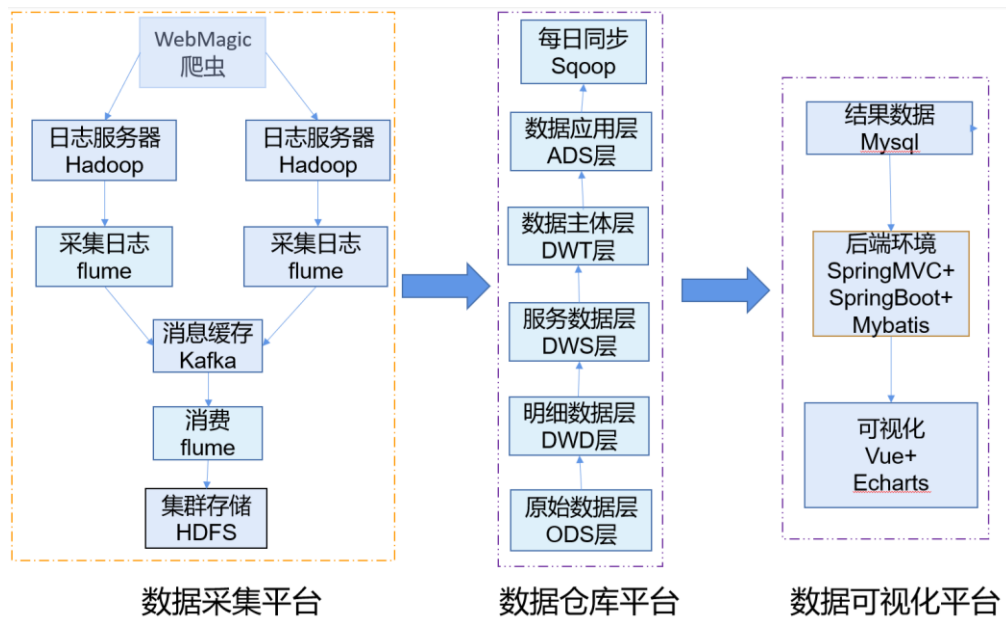


图 4-1 数据仓库架构图

4.1.2 数据采集平台设计

在数据采集方面，在本项目中主要完成三方面需求，将服务器中的日志数据实时采集进大数据存储系统中，防止数据丢失以及数据堵塞，将业务数据库中的数据实时采集进数仓中，同时将需求计算结果导出至关系型数据库方便进行展示。为此我们选用了 Flume、Kafka 和 Sqoop。

Flume 在系统里起到了数据中间存储的功能，能够系统采集、聚集和移动大量的数据并集中存储。Flume 提供了同类型的组件供开发者使用，不同的组件间可以自由组合，只需要修改用户的配置文件即可生效，非常简单，可以满足各种数据采集传输需求。Kafka 在系统里作为消息队列，提供消息

暂存，容错存储的功能。我们可以将它用在应用和处理系统间高实时性和高可靠性的流式数据存储中，也可以实时地为流式应用传送和反馈流式数据。

数据采集模块采用 Flume 采集落盘到服务器文件夹的日志数据，要求可以监控多个日志产生文件夹并能够做到断点续传，可以根据采集到的日志内容对日志进行分类采集落盘，发往不同的 Kafka topic，Kafka 作为一个消息中间件发挥日志缓冲作用，避免同时发生的大量读写请求造成 HDFS 性能下降，并能对 Kafka 的日志生产采集过程进行实时监控，消费层 Flume 避免在落盘 HDFS 过程中产生大量小数据文件，拖低 HDFS 运行性能，并对落盘数据采取适当压缩措施，尽量节省存储空间，降低网络 IO。

4.1.3 数据仓库设计

数据仓库模块主要将采集系统采集到 HDFS 系统的日志文件进行挖掘分析，从中发现内部练习和业务规律，为相关人员提供决策参考。这一过程主要通过数据仓库进行合理分层实现。分析工具选用 Hive，配置 Spark 引擎。数据仓库设计分为 5 层，分别为：

ODS 层：原始数据层，用于存储原始数据。后续可能需要追溯数据源，因此保存了原始的采集日志数据，对原始的数据不做处理。

DWD 层：明细数据层，初步清洗 ODS 层的数据，比如去除非法格式的数据，超出应用范围的数据，过时的旧数据等。数据的结构应该和原始数据层保持一致。

DWS 层：服务中间层，在上一层的基础上对数据聚合操作，聚集到房源城市当日的粒度。以时间为维度生成中间表，算出相应的统计指标，减少重复加工。

DWT 层：主题数据层，按照主题将 DWS 层分为多张小表，然后再拼接成大表对 DWS 层数据进行进一步聚合，构建每个主题的全量宽表，这样就不会出现数据维度少的问题。

ADS 层：数据应用层，为各种统计报表以及可视化提供数据，需要根据具体的需求进行建模。通常这些表都是基于某些维度的数据汇总，需要把结果同步到关系型数据库例如 MySQL 供接口调用或查询。

在 MySQL 中根据 ADS 层结果数据创建对应表 Hourse_info，使用 Sqoop 工具定时将结果数据导出到 MySQL 中，可以使用数据可视化工具 DBeaver 对数据进行离线查看。

5 系统实现

5.1 数据采集模块实现

数据采集模块主要负责将用户浏览日志快速收集并发往 Hadoop。数据采集模块框架结构如图 5-1 所示。

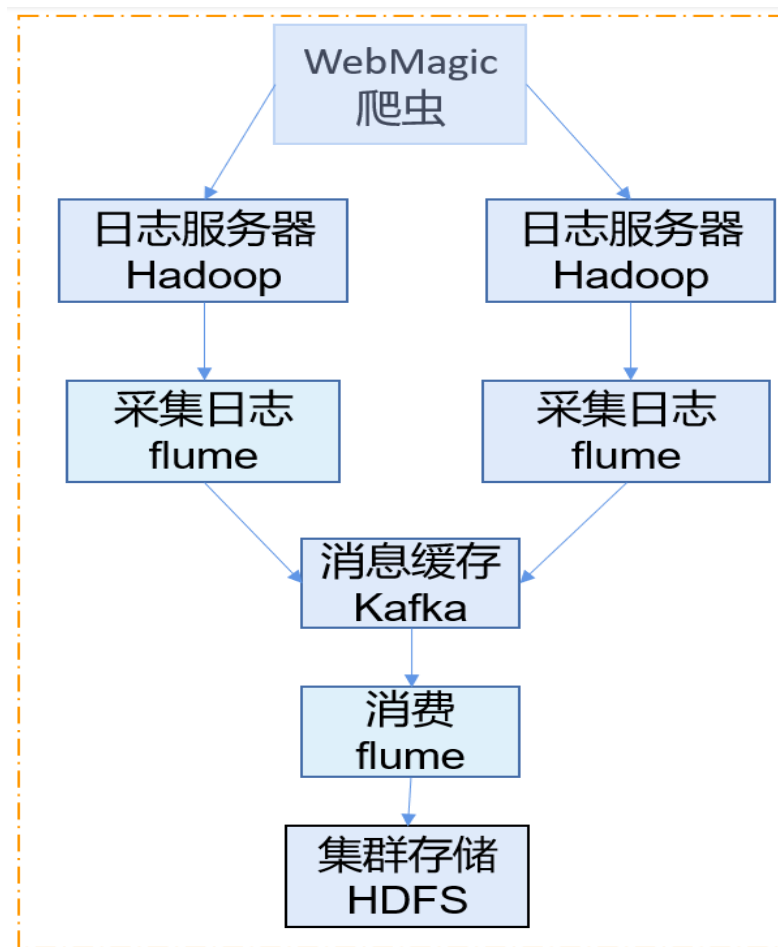


图 5-1 数据采集模块架构

本系统的数据采集和分析模块租用了阿里云的物理机三台，具体配置为：16G 内存、8 核物理 CPU、16 线程、4T HDD 硬盘的戴尔品牌通用云计算机。数据可视化模块使用了本地的计算机，内存 16G，8 核 CPU，1T SSD 硬盘。操作系统为 Linux，CentOS 7（64 位）

大数据框架：Hadoop 3.1.3

中间件：Flume 1.9.0，Kafka 2.4.1，Sqoop 1.4.7，Hive 3.1.2

服务器通过 WebMagic 爬虫，可以对 58 同城租房数据进行爬取，并发往后台日志服务器。

Hadoop 在该系统中起到了海量数据的存储作用，版本号为 3.1.3，配套的框架还有 Sqoop 1.4.7，Hive 3.1.2。

Kafka 在系统里起到缓冲作用，防止产生数据洪峰，影响系统正常运行，版本号为 2.4.1。

Flume 负责读取日志服务器磁盘中的日志，并发往 Kafka，再读取 Kafka 数据上传到 Hadoop，版本号为 1.9.0。

5.1.1 Hadoop 安装部署

Hadoop 在系统中起到了海量数据的存储作用，是数据采集模块重要的组成部分。本项目的集群搭建在阿里云服务器中，租用了 3 台同样配置的服务器，租用的阿里云服务器默认运行 CentOS 7 操作系统。购买服务器实例后，需要对其网络地址进行配置。本系统里服务器的网关地址为阿里的内网地址，IP 地址、子网掩码均为默认地址，设置三台机器主机名为：Hadoop101，Hadoop102 和 Hadoop103，重启网络服务完成配置。

安装 Hadoop 需要有 Java 运行环境，下载安装 JDK 到服务器上，配置 JDK 的环境变量，使用 `java version` 命令检查是否安装成功。下载解压 Hadoop 安装包，用 SecureCRT 工具导入到相应文件夹下，将 Hadoop 添加到环境变量，测试是否安装成功。成功后结果如图 5-2 所示。

```
$ cd /usr/local/hadoop/bin
$ ./hadoop version
Hadoop 2.7.1
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r 15ecc87ccf4a0228f35
Compiled by jenkins on 2015-06-29T06:04Z
Compiled with protoc 2.5.0
From source with checksum fc0a1a23fc1868e4d5ee7fa2b28a58a
This command was run using /usr/local/hadoop/share/hadoop/common/hadoop-common-2.7.1
```

图 5-2 Hadoop 安装信息

Hadoop 运行需要配置文件，在一台服务器上配置 ssh 生成公钥和私钥来访问其他服务器，配置核心文件 `core-site.xml`，`hdfs-site.xml`，`yarn-site.xml`，`mapred-site.xml`。对 `workers` 文件进行修改，将 3 台节点全部指定为从节点。通过 `scp` 命令将配置文件同步修改到另外两台服务器，每个节点上的 Hadoop 配置都基本相同，所以只需要在一个节点上进行操作，配置完成之后同步到另外两个节点上。用户行为日志文件很大，需要对数据进行分片处理，也要

考虑磁盘容量进行压缩，配置 Hadoop 支持 LZO 压缩。最后在三台服务器上启动 Hadoop 集群。

5.1.2 WebMagic 安装部署

WebMagic 是一个 Java 语言下的爬虫框架，使用方便、配置灵活、方便入门，开发者可以快速开发出一个爬虫。项目中通过 WebMagic 爬取 58 同城租房网站的房源信息。

在服务器上安装 Scrapy in Ubuntu，分析网页的数据结构，根据抓取页面信息的键值对创建实体对象来保存数据，保存 item 的映射。设置爬取页面的 URL，模仿浏览器的请求体，设置爬虫抓取策略为每 30 秒查询一次，15 线程。58 网站有自身的防爬机制，测试后发现爬取超过 10 页数据就会停止访问，必须为爬虫配置代理 IP。本系统租用了快代理网站的匿名 IP，通过多线程的方式匿名访问数据。编写函数完成页面跳转的逻辑，设置定时策略和管道存储，在 Scrapy 下启动爬虫。爬取日志如图 5-3 所示

```
{ 'link': 'http://product.dangdang.com/1259816939.html',
  'num': [u'24'],
  'price': [u'3899'],
  'title': [u'\u8054\u60f3\u2013ThinkPad\u2013u8f7b\u2013u8584\u2013u7cfb\u2013u5217E470(20H1A032CD)14\u2013u82f1\u2013u5bf8\u2013u7b14\u2013u8bb0\u2013u672c\u2013u7535\u2013u8111(i3-6006U 8G 1TB 2G\u2013u72ec\u2013u663e Win10)\u2013u9ed1\u2013u8272']]
2017-04-01 15:45:28 [scrapy.core.scrapers] DEBUG: Scraped from <200 http://product.dangdang.com/1303471271.html>
{ 'link': 'http://product.dangdang.com/1303471271.html',
  'num': [u'80'],
  'price': [u'9888'],
  'title': [u'\u2013u82f9\u2013u679c Apple MacBook 12\u2013u82f1\u2013u5bf8\u2013u7b14\u2013u8bb0\u2013u672c 256G\u2013u95ea\u2013u5b58 MMGL2CH\u2013u73ab\u2013u7470\u2013u91d1 MLHE2CH\u2013u91d1 MLHA2CH\u2013u94f6 MLH72CH\u2013u7079']]
2017-04-01 15:45:28 [scrapy.crawler] INFO: Received SIGINT twice, forcing unclean shutdown
```

图 5-3 爬虫测试

5.1.3 Zookeeper 安装部署

Zookeeper 主要是用来存储 Kafka 运行的相关信息。系统中采用了 Kafka 做数据缓冲。所以需要安装 Zookeeper，保证 Kafka 正常运行。

下载解压 zookeeper3.2.7 的安装包，在安装目录下创建 zkData，修改 dataDir 的路径指向 zkData，用于保存 zookeeper 的相关数据。配置 zoo.cfg 文件，设置心跳监测的时间为 15s，在 zkData 目录下创建一个 my.ini 的文件，用以集群启动时由 Zookeeper 读取此文件。编辑 my.ini 文件，在文件中添加端口号，连接数和另外两台服务器的编号。启动 Zookeeper，测试进程是否启动。

动，结果如图 5-4 所示。

```
Connecting to localhost:2181
.....
Welcome to ZooKeeper!
.....
WATCHER::
WatchedEvent state:SyncConnected type: None path:null
[zk: localhost:2181(CONNECTED) 0]
```

图 5-4 Zookeeper 启动

复制 Zookeeper 的配置文件到其他机器上，在 3 台服务器中分别启动 Zookeeper。由于 Zookeeper 没有提供多台服务器同时启动的脚本，单个节点去执行服务启动命令显然操作繁琐，所以将 Zookeeper 启动命令封装成脚本。

5.1.4 Kafka 安装部署

Kafka 是系统中的一个消息中间件，在该项目用于缓冲数据，在数据高峰时降低 Hadoop 上传的速率，保证采集通道稳定运行。

下载解压 Kafka，修改文件夹名为 Kafka。在安装目录下创建 logs 文件夹，修改 conf 目录下的属性文件，配置网络请求的线程数量为 3，缓冲区大小为 100M，设置运行日志存放的路径为创建的 logs 文件夹，配置连接 Zookeeper 的集群地址。配置环境变量，分发 Kafka 的安装包到另外的服务器上，分别修改配置文件，在 3 台服务器中分别启动 Kafka。同样为了避免多次操作，将 Kafka 启动命令封装成脚本。Kafka 启动结果如图 5-5 所示

```
[root@along ~]# service kafka start
Starting kafka (via systemctl): [ OK ]
[root@along ~]# service kafka status
Running 86018
[root@along ~]# ss -nutl
```

Netid	State	Recv-Q	Send-Q	Local Address:Port	Peer Address:Port
tcp	LISTEN	0	50	:::9092	:::*
tcp	LISTEN	0	50	:::2181	:::*

图 5-5 Kafka 启动

5.1.5 Flume 安装部署

日志采集层 Flume 将日志从落盘文件中采集出来传输给消息中间件 Kafka 集群，这期间保证数据不丢失，程序故障挂掉后可以快速重启，对日

志进行初步分类，分别发往不同的 Kafka Topic，方便后续对日志数据进行分别处理。将日志由 Flume 发送到 Kafka 集群后，接下来需要将日志数据进行落盘存储，这部分依然由 Flume 完成。

下载安装 Flume，为了兼容新版本的 Hadoop，删除 lib 文件夹下的 guava 包。创建日志配置文件，设置端口为 44444，绑定本地主机。在配置文件里 Source 选择 Taildir Source。Taildir Source 可以实现断点续传、多目录监控配置。Channel 采用 Kafka Channel，直接将数据输送到 Kafka 集群中，在大数据量的情况下也能稳定运行。整个过程不需要使用中间件，提高了传输效率，而且因为日志存储在磁盘里，不需要担心数据丢失的问题。Sink 选择 HDFS Sink，因为数据最终要上传到 HDFS。在 Flume 的配置文件中配置拦截器，拦截器用来过滤格式不正确的非法数据，可以对数据进行初步的清洗。将 Flume 的启动停止命令封装成脚本，方便调用执行。在完成所有的日志采集落盘工作后，需要将命令和脚本统一封装成采集通道启动停止脚本，否则一项项去开启采集通道的进程也是非常耗时的。

5.2 数据分析模块实现

大数据分布式系统下数据被存储到了不同的服务器中，并且数据没有经过预处理，需要隔离原始数据，将杂乱的数据替换成高质量的可用数据。这就需要对数据进行分层整理和清洗。数据分析模块组件如图 5-6 所示

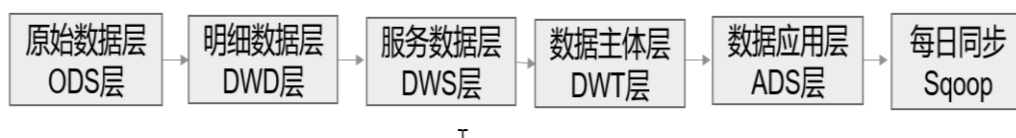


图 5-6 数据分析模块

在 HDFS 里操作数据需要使用 Hive，在本模块里 Hive 既作为存储元数据又负责 SQL 的解析优化，语法是 HQL 语法。下载安装 hive，设置执行引擎为 spark，绑定端口号为 8020，启动 hive 客户端。

ODS 层为原始数据层，用于存储原始数据。导出在 HDFS 里存储的原始数据，使用 Hive 创建数据库 houses_info，存储整个项目的所有数据信息。使用数据库创建日志表 ods_log，创建 json 类型的外部表，按日期进行分区。创建分区表，设置 LZ0 压缩格式，创建索引，提高查询效率。因为展示需要完整的数据源，所以表的内容不做任何修改，只起到备份数据的作用。

DWD 层为明细数据层，主要对 ODS 层数据进行数据清洗，对房源的原

始数据进行分类整理。使用 Hive 新增房屋信息的全量表、增量表和一个错误日志表，全量表用来存储清洗后得到的数据；增量表用于暂存最新的数据，编写脚本执行清洗，将包含 err 字段的日志过滤出来，添加到错误日志表中，每行数据对应一个错误记录。使用 parquet 格式进行存储，保存为 LZO 压缩格式，降低使用存储空间。

DWS 层为汇总数据层，根据指标需求构建公共粒度的汇总表。通过不同的主题，将数据进行汇总和聚合，得到每天每个主题的相关数据。在聚合的过程中，为避免细节数据的丢失，将聚合后的字段使用 concat_ws 函数进行连接，若后期需求开发过程中需要用到这些细节数据，可以通过爆炸函数的使用再次获取到。

DWT 层为数据主体层，把不同的主题做进一步汇总，获得每一个主题的全量数据表。全量表记录了每个维度关联的不同事实表度量值以及首次、末次时间、累积至今的度量值、累积某个时间段的度量值。DWT 层的设备主题宽表将在每日行为表的基础上进行进一步汇总，获得每个房源对应的详情信息，每天将新增加的房源信息增加到主题宽表中，方便后续相关需求。

ADS 层为数据应用层，将主题汇总成多张事实表，最终结果面向用户。对最终指标进行分析，查找最近三周中的房源数据，按照地区、平均单价、租金方式、房屋类型、租房面积进行分组。

用户行为数据分析系统在分析完用户需求得到结果数据之后，需要对结果数据进行可视化展示，在大数据文件系统中进行展示是非常不便的，需要将结果数据导出到关系型数据库中。项目选用 MySQL 作为结果数据保存的关系型数据库。在本地计算机中安装 Mysql，版本为 5.7，搭配搜索引擎 Elasticsearch 5.6.11，使用 Navicat 进行可视化操作，设置用户名和密码，创建对应的数据库和表。开启云服务器和本地服务器的数据库连接，导入数据。

5.3 数据可视化模块实现

可视化模块使用前后端分离的架构模式，数据可视化模块组件如图 5-7 所示。



图 5-7 数据可视化模块

后端使用 IDEA2020 软件进行开发, 采用了 MyBatis 作为持久层框架与 MySQL 对接以方便开发。项目以及自动构建工具使用 Maven-3.5.6。在 pom.xml 配置文件中添加 MyBatis 类和数据库的依赖, 然后在配置文件中添加 com.mysql.jdbc.Driver 驱动, JDBC 的 URL, 数据库的用户名和密码。编写查询语句, 用注释引入配置文件的位置和对应 java 对象的位置。编写 pojo 对象, 设置构造函数、get 和 set 方法。

后端同时使用了 SpringBoot 和 SpringMVC 框架来进行业务逻辑接口的开发。在 pom.xml 配置文件中配置 Spring 的依赖, 根据结构编写对应的实体类 domain, 持久层 dao, 服务层 service, 资源层 resource, 为编写查询函数, 使用 Get 方法返回查询数据, 结合 MySQL 查询结果编写返回体的构造函数, 分层调用方法实现接口方法。

前端使用 Visual Studio Code 软件进行开发, 用 vue 作为开发框架来实现 web 页面显示。在网站上寻找图片作为 web 页面的静态图片放在 static 文件夹下, 引入 Echarts 组件, 绑定类型为 javascript, 指定图表的配置项和数据。在 json 文件上编写脚本方法, ajax 异步调用后端的数据查询接口, 拼接 URL 到函数中, 编写 data 对应返回的数据结构, function 设计点击按钮实现查询和切换功能, 完成数据可视化展示。最终的结果如图 5-8 所示。



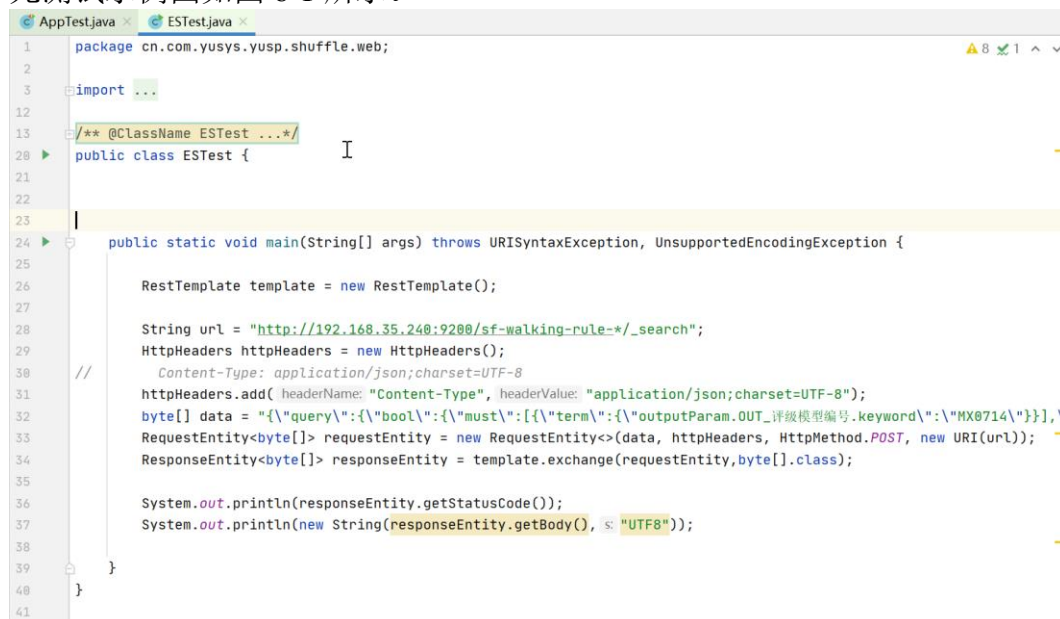
图 5-8 可视化结果图

6 系统测试

在系统开发阶段过程中系统的测试非常重要，测试能发现系统存在的各种问题，并在发现问题后不断提出改进，系统存在的问题越早发现，解决该问题所花的成本也将越低。测试的结果也对系统的开发质量和用户体验产生直接性影响。测试的作用是在系统正式上线之前，发现系统存在的隐藏的 bug，并及时加以修复。本章将主要对租房用户行为数据分析系统进行测试，从单元测试、接口测试、功能性测试进行说明。

6.1 单元测试

系统最基本的测试是单元测试，单元测试主要验证函数是否符合预期，面向的是一个函数或者一个模块，以保证接口的基本需求。单元测试将系统隔离分开，然后进行单独的测试。在软件的开发过程中都会进行单元测试，单元测试通常用来验证函数的执行结果，还能让代码质量更高^[39]。在 SpringBoot 开发的过程中，单元测试比较简单，首先在工程的 pom 文件中引入 test 相关依赖，然后在需要测试的类上加上相关注解就可以进行测试。单元测试示例图如图 6-1 所示。



```
1 package cn.com.yusys.yusp.shuffle.web;
2
3 import ...
4
12
13 /** @ClassName EStest ... */
14 public class EStest {
15
16
17
18
19
20
21
22
23
24 public static void main(String[] args) throws URISyntaxException, UnsupportedEncodingException {
25
26     RestTemplate template = new RestTemplate();
27
28     String url = "http://192.168.35.240:9200/sf-walking-rule-*/_search";
29     HttpHeaders httpHeaders = new HttpHeaders();
30     // Content-Type: application/json;charset=UTF-8
31     httpHeaders.add( headerName: "Content-Type", headerValue: "application/json;charset=UTF-8");
32     byte[] data = "{\"query\":{\"bool\":{\"must\":{\"term\":{\"outputParam.OUT_评级模型编号.keyword\":\"MX0714\"}}},\"
33     RequestEntity<byte[]> requestEntity = new RequestEntity<>(data, httpHeaders, HttpMethod.POST, new URI(url));
34     ResponseEntity<byte[]> responseEntity = template.exchange(requestEntity, byte[].class);
35
36     System.out.println(responseEntity.getStatusCode());
37     System.out.println(new String(responseEntity.getBody(), s: "UTF8"));
38
39 }
40
41 }
```

图 6-1 单元测试

6.2 接口测试

在后端接口开发完成后需要对每个接口进行测试，接口测试满足要求后，才能与前端进行对接。接口分为程序内部的接口和系统对外的接口，由于项

目前后端调用主要是基于 HTTP 协议的接口,这里测试的是系统对外的接口,主要为后端接口。接口测试的工具很多,比如 Postman、Poster、Swagger、SoapUI 等^[40]。本系统采用的接口测试方式是用 Swagger 工具,Swagger 测试结果如图 6-2 所示。Swagger 工具能够满足该系统开发的各种请求。

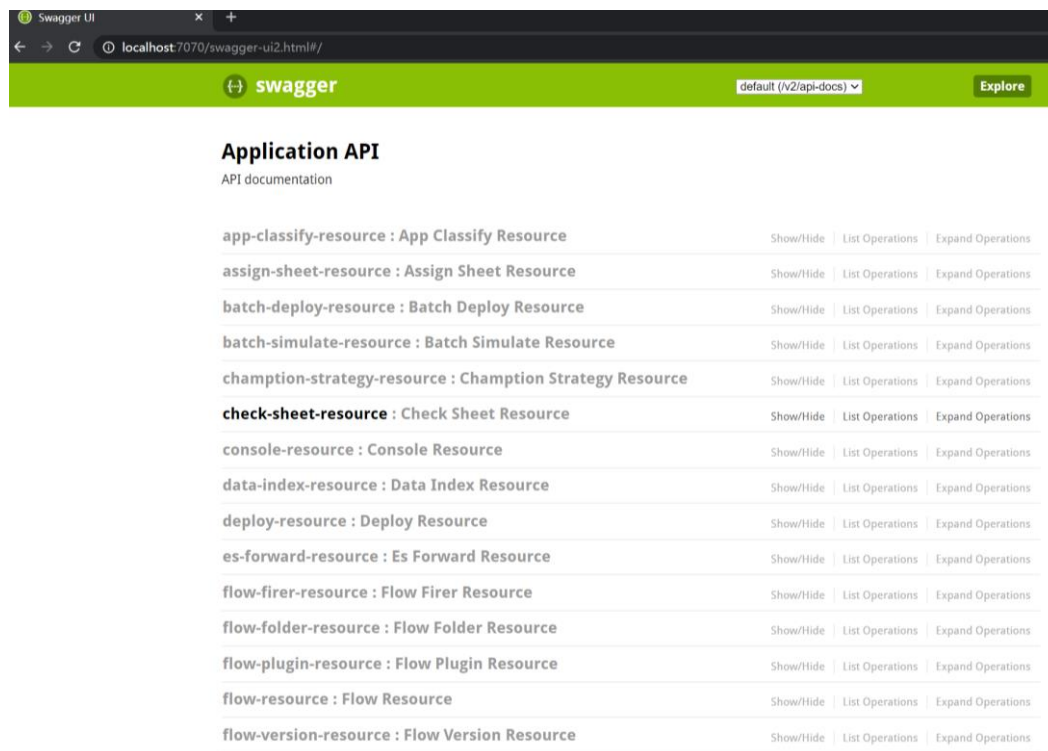


图 6-2 swagger 接口测试

6.3 功能测试

经过单元测试、接口测试,功能整合完成之后,还需要对系统功能进行端到端测试。功能测试是一种黑盒测试,对系统的各项功能进行验证,检查功能是否符合预期的需求。按照数据流向顺序,首先进行数据采集测试,测试数据是否能正常从服务器落盘文件夹中采集,是否能正常落盘到 HDFS 文件系统中,然后进行数据分析测试,主要测试通过数据分析脚本,是否能争取得到结果数据,最后进行数据可视化测试,主要测试前端图标是否能完全显示数据。数据采集模块,数据分析模块,数据可视化模块是系统的三大模块,本节将对这三个模块进行用例测试。

1) 数据采集模块测试

编辑用户行为数据采集通道启动停止脚本,在脚本中依次开启 Hadoop 集群、Zookeeper 集群、Kafka 集群、Flume 数据传输通道,给脚本增加执行权限,启动脚本,然后启动数据模拟程序,去 HDFS 文件系统 web 端查

看数据落盘情况，如图 6-3 所示即说明用户行为数据采集成功。

```

: Spider hf.58.com started!
: downloading page success https://hf.58.com/chuzu/?PGTID=0d100000-0034-58df-ad90-0088ee8cbf22&ClickID=4
: downloading page success https://hf.58.com/hezu/45384051920792x.shtml?houseId=1887116961618954&shangquan=tongli
: downloading page success https://hf.58.com/hezu/45395576243215x.shtml?houseId=1888592075006981&shangquan=binhus
: downloading page success https://hf.58.com/hezu/43476336672900x.shtml?houseId=1642929409991695&shangquan=modian
: downloading page success https://hf.58.com/zufang/45509181548978x.shtml?houseId=1903133553990670&shangquan=hfht
: downloading page success https://hf.58.com/hezu/45497797142275x.shtml?houseId=1901676350219279&shangquan=hqwhf&
: downloading page success https://hf.58.com/hezu/45308087607840x.shtml?houseId=1877393529675777&shangquan=hqwhf&
: downloading page success https://hf.58.com/zufang/45476301105816x.shtml?houseId=1898924857171983&shangquan=hccb
: downloading page success https://hf.58.com/hezu/45506314491021x.shtml?houseId=1902766570725385&shangquan=nanyh&
: downloading page success https://hf.58.com/hezu/43250215312911x.shtml?houseId=1613985875778571&shangquan=ninggu
: downloading page success https://hf.58.com/hezu/45505785492260x.shtml?houseId=1902698858569742&shangquan=sanlia

```

图 6-3 数据采集测试

2) 数据分析模块测试

编写数据仓库数据分析通道 ADS 层数据导入脚本，在脚本中依次执行 ODS 层数据导入、DWD 层数据导入、DWS 层数据导入、DWT 层数据导入、ADS 层数据导入，给脚本执行权限，执行该脚本，然后去 ADS 层对应的 HDFS 文件目录下查看数据生成情况，如图 6-4 所示即说明数据导入成功。

```

Hibernate: select houseinfo0_.id as id1_0_, houseinfo0_.address as address2_0_, houseinfo0_.agent_name as agent_na3_0_, houseinfo0_.area
Hibernate: select houseinfo0_.id as id1_0_, houseinfo0_.address as address2_0_, houseinfo0_.agent_name as agent_na3_0_, houseinfo0_.area
Hibernate: select houseinfo0_.id as id1_0_, houseinfo0_.address as address2_0_, houseinfo0_.agent_name as agent_na3_0_, houseinfo0_.area
Hibernate: select houseinfo0_.id as id1_0_, houseinfo0_.address as address2_0_, houseinfo0_.agent_name as agent_na3_0_, houseinfo0_.area
Hibernate: select houseinfo0_.id as id1_0_, houseinfo0_.address as address2_0_, houseinfo0_.agent_name as agent_na3_0_, houseinfo0_.area
Hibernate: select houseinfo0_.id as id1_0_, houseinfo0_.address as address2_0_, houseinfo0_.agent_name as agent_na3_0_, houseinfo0_.area
Hibernate: select houseinfo0_.id as id1_0_, houseinfo0_.address as address2_0_, houseinfo0_.agent_name as agent_na3_0_, houseinfo0_.area
Hibernate: select houseinfo0_.id as id1_0_, houseinfo0_.address as address2_0_, houseinfo0_.agent_name as agent_na3_0_, houseinfo0_.area
Hibernate: insert into house_info (address, agent_name, area, floor, floor_height, house_area, house_decora, house_desc, house_disposal,
Hibernate: insert into house_info (address, agent_name, area, floor, floor_height, house_area, house_decora, house_desc, house_disposal,
Hibernate: insert into house_info (address, agent_name, area, floor, floor_height, house_area, house_decora, house_desc, house_disposal,
Hibernate: insert into house_info (address, agent_name, area, floor, floor_height, house_area, house_decora, house_desc, house_disposal,
Hibernate: insert into house_info (address, agent_name, area, floor, floor_height, house_area, house_decora, house_desc, house_disposal,
Hibernate: insert into house_info (address, agent_name, area, floor, floor_height, house_area, house_decora, house_desc, house_disposal,
Hibernate: insert into house_info (address, agent_name, area, floor, floor_height, house_area, house_decora, house_desc, house_disposal,
Hibernate: insert into house_info (address, agent_name, area, floor, floor_height, house_area, house_decora, house_desc, house_disposal,

```

图 6-4 数据分析测试

3) 数据可视化模块测试

启动前端 IDE，配置框架和组件，后端配置任务流和时间参数，对接结果数据存储 MySQL，制作仪表盘，配置数据库信息，制作图表，获得可视化数据，如图 6-5 所示。



图 6-5 数据可视化测试

在整个测试的过程中，没有测试系统的高并发性能和安全性，在后面会随着条件的改善，会加入性能测试和安全测试来保障系统的稳定运行。经过单元测试、接口测试和功能测试后，结果显示系统的各个模块和功能运行正常。系统已经达到房屋用户行为分析系统的基础功能，各个模块功能满足系统需求且工作正常。

7 结 论

本文主要完成了以下几个方面的工作：

首先对系统所涉及大数据理论进行了研究。包括 Hadoop 分布式框架和其他的组件，大数据中间件 Flume、Kafka，后端框架 SpringBoot、SpringMVC、MyBatis 的研究。

然后根据需求分析了租房用户行为分析系统的功能需求和非功能需求，将实现功能划分为数据收集模块、数据仓库模块、数据可视化模块。数据采集功能的实现具体对采集的事件进行分析，对采集的流程、Flume、Kafka 的配置进行了详细的说明。数据分析模块则是对数据分层处理。详细分析了：租房平均单价、房屋地区数据、房屋面积与租金数据、房屋类型数据、房屋租金支付方式、租房面积数据数据展示采用 Vue 和 Echarts 对不同的指标分别采用：柱状图、饼状图、折线图、和散点图进行可视化。

最后搭建了整套集群环境，对系统进行了单元测试，接口测试和功能测试，验证了系统的可靠性。租房用户行为分析系统解决了传统数据分析的局限性，将数据转化为知识，发掘用户行为数据的商业价值。可视化的结果数据能帮助租房平台，了解用户心理行为，分析租房市场的现状，制定相应的商务决策与营销策略。

本文主要完成了以上几个方面内容，考虑到数据分析比较简陋，原始数据没有得到完全的利用，为了更好地完善系统的功能，未来会添加的功能如下：

- 1) 由于时间原因只针对用户行为数据进行了分析，后续需要增加业务数据的分析；
- 2) 针对当前流行的元数据管理和质量监控方面需要引入到该项目中。增加项目的完善程度；
- 3) 考虑增加查询方面的功能，查询数月或数周前的数据，有效利用旧的数据。

参考文献

- [1] 罗珉, 李亮宇. 互联网时代的商业模式创新:价值创造视角[J]. 中国工业经济, 2015, 28(01):95-107.
- [2] 张光昇. 基于 Elasticsearch 的房源搜索系统的设计与实现[D]. 武汉:华中科技大学,2019.
- [3] 刘洋瑀. 某在线租房系统的设计与实现[D]. 武汉:华中科技大学, 2019.
- [4] 张帆. 基于 Android 的智能找房系统设计与实现[D]. 北京:中国地质大学, 2017.
- [5] 李大洲. 基于大数据的用户行为日志系统设计与实现[D]. 南京:南京邮电大学, 2020.
- [6] 白建森. 大数据的应用现状与未来展望[J]. 电脑迷, 2018(09):137.
- [7] 许长福. 日志数据分析系统的设计与实现[D]. 北京:北京交通大学,2017.
- [8] Vanita Jain, Arun Kumar Dubey, Achin Jain et al. Crime pattern recognition in Chicago city using Hadoop multimode cluster[J]. Journal of Information and Optimization Sciences:2019, 359:605-609.
- [9] 庞双玉. 基于 Hadoop 大数据平台的金融产品购买行为分析[J]. 电子技术与软件工程, 2019(04):182+232.
- [10] 张思航. 基于 Hadoop 的电信大数据处理的研究及应用[D]. 北京:华北电力大学, 2017.
- [11] 王彦明. 近年来 Hadoop 国内研究进展[J]. 现代情报, 2014, 34(08):14-19.
- [12] 袁丹. 基于大数据平台的电信用户行为日志分析研究[D]. 成都:成都理工大学, 2017.
- [13] 曹茜茜. 基于 Hadoop 的电信大数据分析的设计与实现[D]. 西安:西安科技大学, 2015.
- [14] Cong Jin. The research for storage scheme based on Hadoop[C]. IEEE Beijing Section, Sichuan Institute of Electronics. Proceedings of 2015 IEEE International Conference on Computer and Communications (ICCC 2015). IEEE Beijing Section, Sichuan Institute of Electronics. 2015. pp.271-276.
- [15] 李元亨,邹学玉. Hadoop 综述[J]. 电脑知识与技术,2018,14(09):8-9.
- [16] Li Tao, He Shuibing, Chen Ping et al. Application and storage-aware data

- placement and job scheduling for Hadoop clusters[J]. Journal of Circuits, systems and Computers, 2020, 29(16):124.
- [17] 郝树魁. Hadoop HDFS 和 MapReduce 架构浅析[J]. 邮电设计技术, 2012, 53(07):37-42.
- [18] Xun Cai. An optimization strategy of massive small files storage based on HDFS[A]. 重庆环球联合科学技术研究院 2018:6.
- [19] 陈忠义. 基于 Hadoop 的分布式文件系统[J]. 电子技术与软件工程, 2017, 5(09):175.
- [20] Zhipeng Gao, Yinghao Qin, Kun Niu. An effective merge strategy based hierarchy for improving small file problem on HDFS[C]. IEEE Beijing Section, 2016, pp50-68.
- [21] Mininath Bendre and Ramchandra Manthalkar. Time series decomposition and predictive analytics using MapReduce framework[J]. Expert Systems With Applications, 2018, 28(116):108-120.
- [22] V. Subramaniaswamy, R. Logesh et al. Unstructured data analysis on big data using MapReduce[J]. Procedia Computer Science, 2015, 50(06): 456-465.
- [23] 付伟. 基于 Hadoop 的 Web 日志的分析平台的设计与实现[D]. 北京:北京邮电大学, 2015.
- [24] Vinod Kumar Vavilapalli et al. Apache Hadoop Yarn[C]. Cloud Computing, 2013, PP 1-16.
- [25] 刘阳. 基于 YARN 的高响应性 Hadoop 计算资源调度器的研究与实现[D]. 黑龙江:哈尔滨工业大学,2016.
- [26] 王珊珊. 基于 Hadoop 集群作业调度性能优化技术的研究与实现[D]. 辽宁:沈阳工业大学,2020.
- [27] 方中纯, 赵江鹏. 基于 Flume 和 HDFS 的大数据采集系统的研究与实现 [J]. 内蒙古科技大学学报, 2018,37(03):255-259.
- [28] 牛牧. 基于 Kafka 的大规模流数据分布式缓存与分析平台[D]. 吉林:吉林大学, 2016.
- [29] Bunrong Leang, Ean Sokchomrern, Ryu Ga-Ae et al. Improvement of Kafka streaming using partition and multi-threading in big data environment[J]. Sensors, 2019, 19(1):134-138.
- [30] 吕宇琛. Spring Boot 框架在 web 应用开发中的探讨[J]. 科技创新导报, 2018, 14(8): 168-173.
- [31] 江雁. 浅谈 SpringBoot 框架下如何快速进行后台开发[J]. 海峡科技与

- 产业, 2019, 24(2): 53-56.
- [32] Walls C. SpringBoot in action[M]. London: Manning Publications, 2016.
- [33] Reddy K S P. Web applications with SpringBoot[M]. Chicago: Beginning SpringBoot 2. Apress, Berkeley, CA, 2017: 107-132.
- [34] Deck P. Spring MVC: a tutorial[M]. London: Brainy Software Inc, 2016.
- [35] 戴克. Spring MVC 学习指南[M]. 北京:人民邮电出版社, 2017.
- [36] 孙浩. 基于 SpringMVC 及 MyBatis 框架的在线教育平台的设计与实现[D]. 辽宁:沈阳师范大学, 2019.
- [37] Yao Zhang Li, Sheng Gao, Jing Pan et al. Research and application of template engine for web back-end based on MyBatis-Plus[J]. Procedia Computer Science, 2020, 11(166): 206-212.
- [38] 熊凯. 基于 Spring Cloud 的培训商务系统的设计与实现[D]. 北京:北京邮电大学, 2019.
- [39] 林斌. 单元测试在软件测试中的应用分析[J]. 电脑编程技巧与维护, 2020, 26(02):20-22.
- [40] 江屿. 基于 Junit 的接口测试框架的设计与实现[D]. 江苏: 东南大学, 2015.

在学取得成果

一、 在学期间所获的奖励

专业英文词汇赛本科组三等奖，
全国大学生计算机应用能力与信息素养大赛组委会，
2019.6

二、 在学期间发表的论文

三、 在学期间取得的科技成果

致 谢

春来秋去，花开花落，转眼间就到了毕业季。意味着我的校园生活即将画上圆满的句号。回顾起这四年的本科生活，报名入学的场景仿佛如同还在昨日。四年的时光里笑过也惆怅过，感触颇多，也收获了许多。在此对学习和生活中给予过我帮助的人表示衷心的感谢！

首先最深的谢意先给我的导师任超。任老师一丝不苟的工作态度、丰富的专业知识、长远的学术眼界使我受益终身。在生活上任老师对待学生宽容体贴，亦师亦友。感谢老师对我学习和生活的指导和关怀，不仅教授我知识，还带我参加许多学术会议，鼓励我积极寻找工作参与社会实践，使我学以致用。论文写作方面任老师给予了我很大的帮助，在选题方面给了我很大的自由，为我分析了答辩和写作的技巧，帮助我进行架构分析和论文后期的整理。论文的顺利完成离不开任老师的辛勤指导。

感谢我的父母对我的关爱和培养，数年来是你们无私且伟大的爱使我得以长大成人，在我前进的道路上你们给了我坚定的支持和鼓舞，感谢你们对我学习生涯的支持和奉献；感谢我的舍友，一起生活了 4 年有很多的不舍，时常会让你下楼顺路带饭或者取快递回来，有时也会借你的饭卡洗澡和打水，有一些生活上的问题也经常与你们讨论；同时我还要感谢学校的各位专业老师，感谢你们的细心教导，为我讲解了很多专业知识，让我的学习生活不再迷茫。

最后，感谢各位参与评审和答辩工作的评委老师和专家们，谢谢你们给我的论文提出宝贵意见和建议。