Inline Text Wrapping Picture

北京邮电大学

高校大数据治理融合分析与设计

学 号: 2013128226

姓 名: 张伟俭

学 院: 软件学院

专业(领域): 软件工程

研究方向: 企业信息化

导师姓名: 郭文明

攻 读 学 位: 工程硕士

2016年9月21日

**【抽时间参考模板按要求调整封面及格式】**

独创性（或创新性）声明

本人声明所呈交的论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京邮电大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名：日期：

关于论文使用授权的说明

本人完全了解并同意北京邮电大学有关保留、使用学位论文的规定，即：北京邮电大学拥有以下关于学位论文的无偿使用权，具体包括：学校有权保留并向国家有关部门或机构送交学位论文，有权允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，有权允许采用影印、缩印或其它复制手段保存、汇编学位论文，将学位论文全部或者部分内容编入有关数据库进行检索。（保密的学位论文在解密后遵守此规定）

本人签名：日期：

导师签名：日期：

高校大数据治理融合分析与设计

摘要

信息化技术的发展，对教育发展也具有革命性影响，国家政府予以高度重视。全国高校大力发展信息化技术，开发、引入了覆盖教育领域方方面面的信息化系统，但传统的信息化高校管理理念和管理方法无法满足日益增长的数据处理需求，导致发展不均衡难题、方式单调化难题、信息隐形化难题、决策粗放化难题、择校感性化难题、就业盲目化难题等多种问题；如何对校园信息化系统数据进行高效的管理和实现数据共享、融合治理，是当今校园建设所面临的急需解决的问题。

本文通过深入分析体验大数据生态圈主流技术，调研分析高校常见信息化系统数据特征，以学生相关数据(如图书数据、一卡通数据、就业数据、医院数据、上网数据)为融合分析研究对象，构建了高校大数据治理融合平台。平台从下至上分为三层，即各类异构数据源层、大数据治理融合层和大数据专题分析层。大数据治理融合层是核心研究内容，分为系统管理、数据采集、数据存储、数据处理、数据分析、数据呈现多个功能模块。数据采集模块基于Kettle、Sqoop开源技术实现ETL模型，既提高了数据融合的效率又保证数据融合的质量。数据呈现模块研究了FineBI可视化工具，该工具支持多种数据源的可拖拽式零编程数据可视化分析，即可对接高校大数据治理融合平台，亦能对接第三方数据分析平台。

本人参与了开源大数据调研选型、大数据治理融合平台总体架构设计、学生相关数据融合需求分析及实现。由于个人精力及设备资源问题，高校大数据治理平台目前处于研发试验阶段，围绕学生相关数据融合分析研究，初步完成验证了高校大数据治理融合平台理论可行性，初步掌握了高校学生相关数据的特征及业务逻辑；随着对高校大数据的理解和设备资源的投入，将融合更多的高校应用场景，进一步提高高校大数据治理融合平台的性能和应用领域。

**关键词**：高校大数据 治理融合 开源 信息化 FineBI

ANALYSIS AND DESIGN OF Big Data Governance in Colleges and Universities

【Google翻译还未整理】

ABSTRACT

The development of information technology has also had a revolutionary impact on the development of education. The national government has attached great importance to it. National colleges and universities have vigorously developed information technology and developed and introduced information systems that cover all aspects of the education field. However, traditional information technology college management concepts and management methods are unable to meet the ever-increasing demand for data processing, resulting in unbalanced development problems and monotonous ways. Puzzles, information stealth problems, rough decision-making problems, school choice perceptual problems, employment blindness problems and other issues; how to effectively manage campus data and achieve data sharing, integration governance, is the campus today Facing the urgent need to solve the problem.

This article through the in-depth analysis of the experience of big data ecosystem mainstream technology, research and analysis of university information systems data characteristics, build a university big data governance integration platform. The platform is divided into three layers from bottom to top, that is, various heterogeneous data sources, big data governance convergence layer and big data thematic analysis layer. The big data governance fusion layer is the core research content and is divided into system management, metadata management, data collection, data storage, data processing, data analysis, data presentation, and API/SDK eight function modules. Based on the Kettle, Sqoop and Flume open source technologies, the data acquisition module creatively proposed a "three-stage ETL model" that not only improves the efficiency of data fusion but also ensures the quality of data fusion. The data rendering module natively developed a draggable zero-programming data analysis visualization system supporting multiple data sources based on Java, SpringMVC, React, Echarts, D3.js, and microservices. The system uses REST/JDBC protocol and big data. Basic platform communication, with good scalability, can connect to university big data governance integration platform, but also to third-party data analysis platform.

I participated in the selection of open source big data research, the overall architecture design of the big data governance integration platform, the analysis of common application scenarios in colleges and universities, and the related application scenarios and use case realization of the university governance integration platform. Due to the personal energy and equipment resources, the college big data management platform is currently in the experimental stage of R&D. It has initially verified the theoretical feasibility of the university's big data governance integration platform, and has initially grasped the data characteristics and business logic of common university application scenarios; The understanding of college big data and the investment of equipment resources will integrate more application scenarios of colleges and universities to further improve the performance and application fields of the integration platform of big data governance in colleges and universities.

**Keywords**: college big data governance integration open source informationization

目录

[第一章 绪论 7](#_Toc508308962)

[1.1 研究的背景 7](#_Toc508308963)

[1.2 高线大数据治理现状 9](#_Toc508308964)

[1.3 主要研究内容及论文结构 10](#_Toc508308965)

[1.3.1 研究内容 10](#_Toc508308966)

[1.3.4 章节安排 11](#_Toc508308967)

[1.4 本章小结 11](#_Toc508308968)

[第二章 高校大数据治理融合相关方法与技术 12](#_Toc508308969)

[2.1 Hadoop技术生态圈 12](#_Toc508308970)

[2.2高校大数据技术 14](#_Toc508308971)

[2.3 数据采集与建模技术 14](#_Toc508308972)

[2.4 数据存储 16](#_Toc508308973)

[2.4.1 分布式文件存储 16](#_Toc508308974)

[2.4.2 NoSQL数据库 17](#_Toc508308975)

[2.5 数据计算与挖掘 19](#_Toc508308976)

[2.6 可视化技术 20](#_Toc508308977)

[2.7大数据管理系统 20](#_Toc508308978)

[2.8 本章小结 20](#_Toc508308979)

[第三章 高校大数据治理融合总体架构与场景分析 21](#_Toc508308980)

[3.1 高校大数据治理融合平台总体架构 21](#_Toc508308981)

[3.2 高校大数据应用场景分析 22](#_Toc508308982)

[2.2.1 舆情分析 23](#_Toc508308983)

[2.2.2 综合预警 23](#_Toc508308984)

[2.2.2.1 学生失联预警 24](#_Toc508308985)

[2.2.2.2 贫困生预警 24](#_Toc508308986)

[2.2.2.3上网预警 24](#_Toc508308987)

[2.2.2.4 一卡通消费预警 24](#_Toc508308988)

[2.2.2.5 学生沉迷预警 24](#_Toc508308989)

[2.2.2.6 图书借阅预警 24](#_Toc508308990)

[2.2.3 就业招生分析 24](#_Toc508308991)

[2.2.4 科研分析 24](#_Toc508308992)

[2.2.5 教务分析 26](#_Toc508308993)

[2.2.6 资产管理 31](#_Toc508308994)

[2.2.7 教学管理 32](#_Toc508308995)

[2.2.8 战略决策 33](#_Toc508308996)

[2.2.9 教学管理 33](#_Toc508308997)

[2.2.10 学生画像 35](#_Toc508308998)

[2.2.11 财务分析 38](#_Toc508308999)

[2.2.12 助学分析 40](#_Toc508309000)

[2.2.13 人事分析 40](#_Toc508309001)

[2.1.14 学工分析 41](#_Toc508309002)

[2.1.15 校园综合分析 42](#_Toc508309003)

[2.1.16 学校信息化管理分析 42](#_Toc508309004)

[3.3 本章小结 42](#_Toc508309005)

[第四章 高校大数据治理融合平台设计与实现 43](#_Toc508309006)

[4.1 高校大数据治理融合基础平台 43](#_Toc508309007)

[4.1.1 平台物理架构 43](#_Toc508309008)

[4.1.1.1 物理组网 43](#_Toc508309009)

[4.1.1.2 主机配置列表 43](#_Toc508309010)

[4.1.1.3 Host主机名配置 43](#_Toc508309011)

[4.1.1.4 大数据服务配置 44](#_Toc508309012)

[4.1.2 平台管理系统 45](#_Toc508309013)

[4.1.3 平台功能架构 48](#_Toc508309014)

[4.2 平台数据建模 48](#_Toc508309015)

[4.2.1 数据建模工具与流程 48](#_Toc508309016)

[4.2.1.1 Flume拦截器morphline 49](#_Toc508309017)

[4.2.1.2 Sqoop数据清洗(SQL/Kite) 50](#_Toc508309018)

[4.2.1.3 Kettle数据采集与ETL 52](#_Toc508309019)

[4.2.1.4 高校数据采集 53](#_Toc508309020)

[4.2.1.5 数据入库 54](#_Toc508309021)

[4.2.2 数据标准规范 54](#_Toc508309022)

[4.2.3 数据清洗示例 54](#_Toc508309023)

[4.2.4 数据建模 55](#_Toc508309024)

[4.3 平台数据处理 56](#_Toc508309025)

[4.3.1 数据存储 56](#_Toc508309026)

[4.3.1.1 设计思路 56](#_Toc508309027)

[4.3.1.2 分布式文件系统 56](#_Toc508309028)

[4.3.1.3 关系型数据库 57](#_Toc508309029)

[4.3.1.4 非关系型数据库(NoSQL) 57](#_Toc508309030)

[4.3.1.5 数据存储格式 58](#_Toc508309031)

[4.3.1.6 高校大数据治理融合平台存储方案 58](#_Toc508309032)

[4.3.2 数据处理 58](#_Toc508309033)

[4.3.2.1 实时数据计算 59](#_Toc508309034)

[4.3.2.2 离线批量处理 60](#_Toc508309035)

[4.3.2.3 即席查询 60](#_Toc508309036)

[4.4 可视化系统 61](#_Toc508309037)

[4.4.1 可视化系统架构 61](#_Toc508309038)

[4.4.2 数据源管理 62](#_Toc508309039)

[4.4.3 权限管理 63](#_Toc508309040)

[4.4.3.1机构管理 63](#_Toc508309041)

[4.4.3.2用户管理 64](#_Toc508309042)

[4.4.3.3角色管理 66](#_Toc508309043)

[4.4.4 业务前端展示图表 67](#_Toc508309044)

[4.4.5 可视化系统数据库设计 69](#_Toc508309045)

[4.4.5.1 数据源表结构 69](#_Toc508309046)

[4.4.5.2 应用表结构 70](#_Toc508309047)

[4.4.5.3 元数据存储表 70](#_Toc508309048)

[4.4.5.4 仪表盘表结构 71](#_Toc508309049)

[4.4.5.5 图表表结构 71](#_Toc508309050)

[4.6 本章小结 72](#_Toc508309051)

[第五章 总结与展望 73](#_Toc508309052)

[5.1 论文工作与总结 73](#_Toc508309053)

[5.2 进一步研究工作 74](#_Toc508309054)

[参考文献 75](#_Toc508309055)

[致谢 76](#_Toc508309056)

# 第一章 绪论

## 1.1 研究的背景

近几十年来，互联网的快速发展推动了信息化技术如雨后春笋般迅猛发展和普及应用，各行各业的应用系统的规模迅速扩大，给互联网、电子商务、生产制造、交通和物流、金融和保险、医疗卫生、地理信息、政府机构、教育等机构和组织带了极大的便利和价值：实现了信息有效的流通与管理、实现了资源和知识共享、提高工作效率；然而，随着对信息化技术的广泛使用和深入实践，信息化技术也带来一些新问题：信息化系统之间数据对接接口不兼容，数据整合麻烦，形成信息孤岛；数据积累与日俱增，数据存储成本较高；数据产生速度加快、数据量不断增多、数据价值密度越来越低、有效信息或知识获取难度越来越大[1-9]。

特别是近十年来，硬件技术、移动通信、智能设备、传感设备飞速发展有力地带动了电子商务、社交网络、共享经济等新型生活方式的演进，新型生活模式导致数据量猛增，每天都在创造着大量的数据，数据呈爆炸性增长。截止到2012年，数据量已经从TB级别跃升到PB、EB乃至ZB级别。国际数据公司（IDC）的研究结果表明，2008年全球产生的数据量为0.49ZB，2009年的数据量为0.8ZB，2010年增长为1.2ZB，2011年的数量更是高达1.82ZB，相当于全球每人产生200GB以上的数据。而到2012年为止，人类生产的所有印刷材料的数据量是200PB，全人类历史上说过的所有话的数据量大约是5EB。IBM的研究称，整个人类文明所获得的全部数据中，有90%是过去两年内产生的。而到了2020年，全世界所产生的数据规模将达到今天的44倍[10]。海量数据采集、存储及分析能力成为各行各业，特别是数据聚集型行业的核心竞争力。

数据巨头机构或企业，如Google、Yahoo、Facebook等都积极展开大数据技术研究，推动大数据时代变革。2003年-2006年Google相继发表《GFS》、《MapReduce》、《BigTable》三篇论文向全世界介绍了大数据存储、分析及索引的关键理论，2006年Doug Cutting开源了基于Google三篇论文实现的hadoop大数据存储及分析框架，开源Hadoop技术标志着全世界正式进入大数据时代，大数据处理技术将不再是高科技巨头机构的专属，各行各业的机构和组织都可以利用大数据技术解决信息化过程中遇到的问题。大数据将不再只是技术手段，更是思维意识、方法论和国家战略[11-16]。

我国有3000多所高校，许多高校规模较大，人数万人以上的。高校师生及管理人员的活动及社会关系，如招生、教学、医务、学籍、选课、成绩、食堂、上网、体育活动、科研、财务、安全、薪资、住行、图书借阅、设备管理等方面产生大量的数据。传统的管理理念和管理方法无法满足日益增长的数据处理需求，导致发展不均衡难题、方式单调化难题、信息隐形化难题、决策粗放化难题、择校感性化难题、就业盲目化难题等多种问题；如何对校园数据进行高效的管理和实现数据共享，如何有效地分析、挖掘高校积累的海量数据中的教学和科研价值、改善教育领域的授课和学习模式、提高学生的学习能力、生活质量、心理健康、就业水平，助推教育的全方位变革与创新发展，是当今校园建设所面临的急需解决的问题[17]。

现有校园数据分析方法和系统存在以下几个问题：

* 数据来源多元化，数据类型多样化，数据增量大，数据管理及数据质量参差不齐。
* 信息系统种类繁多，系统间数据不能高效共享，不能有效地构建多维度关联分析，形成信息孤岛。
* 大量系统使用文本、关系型数据库存储数据，数据类型各异、数据格式多样、同一高校实体数据分散在不同的系统，数据分割严重、碎片化程度高；导致数据分析技术简单、分析效率低下、数据可挖掘价值低，数据利用不充分。
* 数据展示主要以事务型被动查询为主；无法跨库及跨系统关联查询、不能实时推送、多维分析、不能可视化展示；数据呈现能力弱、时效性低。
* 数据安全与保护意识低，系统间安全技术不统一、安全能力参差不齐，数据的安全性与私密性得不到有效的保障。

本文提出高校大数据治理融合解决方案，旨在研究高校多源异构数据采集、清洗、共享的技术，保证数据质量、消除信息孤岛；研究高校数据分布存储及统一建模分析模型，提高数据融合程度、实现数据海量分析及挖掘技术；研究大数据可视化系统，提升数据表现力，实现实时、多维拖拽式可视化展示技术；分析设计一整套高校大数据治理融合处理平台，提供上层应用API接口，支撑高校对人事、教务、学生、招生、就业、科研、图书、一卡通、医疗、网络、校园管理等信息的统一建模、采集、处理、分析大数据应用的设计与开发。

## 1.2 高线大数据治理现状

信息化技术的发展，对各行各业的产生了深远的影响。对教育发展也具有革命性影响，国家政府予以高度重视。把教育信息化纳入国家信息化发展整体战略，超前部署教育信息网络[18]。尽管，全国高校大力发展信息化技术，开发、引入了覆盖教育领域方方面面的信息化系统，如招生管理系统、教学系统、医务系统、选课系统、上网管理系统、科研系统、财务管理系统、图书系统、设备管理系统等，但传统的信息化高校管理理念和管理方法无法满足日益增长的数据处理需求，导致发展不均衡难题、方式单调化难题、信息隐形化难题、决策粗放化难题、择校感性化难题、就业盲目化难题等多种问题；如何对校园信息化系统数据进行高效的管理和实现数据共享，如何有效地分析、挖掘高校积累的海量数据中的教学和科研价值、改善教育领域的授课和学习模式、提高学生的学习能力、生活质量、心理健康、就业水平，助推教育的全方位变革与创新发展，是当今校园建设所面临的急需解决的问题[19]。因此，必须清醒地认识到，加快推进教育信息化还面临诸多的困难和挑战。对教育信息化重要作用的认识还有待深化和提高；加快推进教育信息化发展的政策环境和体制机制尚未形成；基础设施有待普及和提高；数字教育资源共建共享的有效机制尚未形成，优质教育资源尤其匮乏；教育管理信息化体系有待整合和集成；教育信息化对于教育变革的促进作用有待进一步发挥。推进教育信息化仍然是一项紧迫而艰巨的任务[20].

互联网、物联网和云计算技术的迅猛发展，使得数据充斥着整个世界，大数据应时而生[21]；大数据是信息化技术的发展的高级阶。大数据技术通过采集、集成及融合各种信息化系统产生的异构数据，有效地解决数据孤岛问题；利用低成本的分布式海量数据离线及实时计算对多源数据进行分析、挖掘或机器学习，能够从数据中发现价值。因此，大数据技术这种基于“数据驱动”来解决问题的思路和能力，被广泛各行各业中，如金融、电信、零售、电商等数据聚集领域。国内外，学者专家也相继应用大数据技术来解决高校教育信息化发展中诸多问题[22-24]。

2013年被认为是大数据在教育领域应用研究的元年，国内掀起了基于大数据技术促进教育改革和创新发展相关研究的热潮[25]。国家也相继颁发了系列教育信息化政策文件（如《2014年教育信息化工作要点》、《国家中长期教育改革和发展规划纲要（2010-2020年）》、《国家教育事业发展第十二个五年规划》、《教育信息化十年发展规划2011-2020年》等）有力地推动了教育信息化、教育大数据研究的发展。教育大数据主要研究内容包括对教育数据挖掘、思想政治教育、个性化教育、学习分析、教育方式改善、学习策略探讨、教育管理方式改进、图书管理、财务分析、设备管理、评估方法等方面,教育大数据还尚处于起步阶段，涉猎的内容也比较狭窄、深度也不够，需要进一步加强大数据在教育中的研究力度[26]。教育大数据目前只焦距在教学事务及学生活动相关领域，只有扩展大数据在高校的研究领域、范围才能更好的利用数据，构建高校大数据治理融合平台，为智慧校园提供强大的数据及技术支撑。正如王小波说：“教育正在走向大数据时代，谁能够发现数据，谁就能够赢得未来的生存；谁能够挖掘数据，谁就能够赢得未来的发展；谁能够利用数据，并利用数据提供个性化的服务，谁就能够赢得未来的竞争。三个层次是递进的关系，即发现数据，挖掘数据，利用数据。”[27]

## 1.3 主要研究内容及论文结构

### 1.3.1 研究内容

本文通过深入分析体验大数据生态圈主流技术，并按照成熟度、开源、易用性、社区等标准选取了大数据服务；同时，深入调研分析高校各信息化系统数据特征，构建了高校大数据治理融合平台及高校大数据分析专题包。

高校大数据治理融合平台下至上分为三层，分别为：各类异构数据源、大数据治理融合层和大数据专题分析层。其中，大数据治理融合层是核心研究内容，分为系统管理、数据采集、数据存储、数据处理、数据分析、数据呈现等功能模块。数据采集模块基于Kettle、Sqoop开源技术组合处理ETL、关系型数据库采集和日志记录采集，既提高了数据融合的效率又保证数据融合的质量。数据呈现模块使用FineBI免费商业软件，支持多种数据源的可拖拽式零编程数据分析可视化系统。

大数据专题分析层深入分析了几个典型的高校业务场景，研究了图书管理等业务的数据结构及各个业务的功能，同时，基于高校大数据治理融合平台，提出了合理的解决方案。

### 1.3.4 章节安排

本文的内容结构安排如下：

第一章 绪论，介绍了课题的研究背景，通过对高校大数据应用及治理的国内外现状的综述和分析，确定了本文的研究内容及章节安排。

第二章 本章介绍了探讨了高校大数据技术、Hadoop技术、数据采集建模技术、数据存储、数据计算与挖掘、数据可视化技术及大数据平台管理系统等高校大数据治理融合平台相关的方法与技术，通过对这些方法与技术的深入研究学习，为高校大数据采集、融合和分析提供理论指导和工具准备工作。

第三章 本章介绍了高校大数据治理融合总体架构，深入分析调研高校各信息化系统及数据源，探讨了几个典型的高校大数据应用场景，并结合总体架构提出了数据流程及关键算法。

第四章 本章详细介绍了高校大数据治理融合平台的设计方案与实现方法。使用三台Dell服务器及4台虚拟机搭建了高校大数据治理融合平台试验环境。利用CDH大数据管理系统安装、部署了大数据集群服务，如Hive、Yarn、HDFS、Spark、Kafka、Zookeeper、Flume、Habse等。分析了数据建模相关知识点，即建模使用的工具、流程、规范、规则等，并根据高校大数据的数据内容设计了数据模型。详细展示了数据存储方案及可视化工具的安装与应用。

第五章 总结了论文工作及进一步工作展望

## 1.4 本章小结

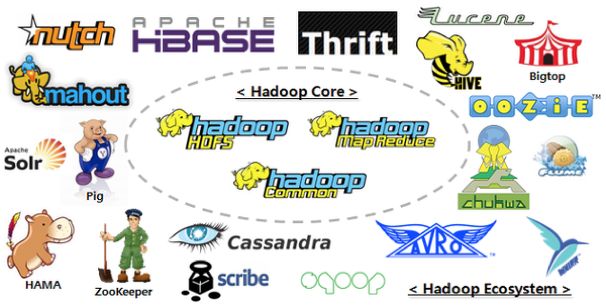
本章介绍了课题的研究背景，通过对高校大数据应用及治理的国内外现状的综述和分析，确定了本文的研究内容及章节安排。

# 第二章 高校大数据治理融合相关方法与技术

## 2.1 Hadoop技术生态圈

Apache Hadoop 是一项由 ASF(Apache Software Foundation，Apache 软件管理基金会)管理的顶级开源项目。Hadoop 不是一个独立实体。最好的定义是一种平台或生态系统，描述了大规模利用被配置成拥有集群计算能力的商用硬件进行分布式数据处理的方法。

下图展示了以Hadoop为核心的大数据生态体系：

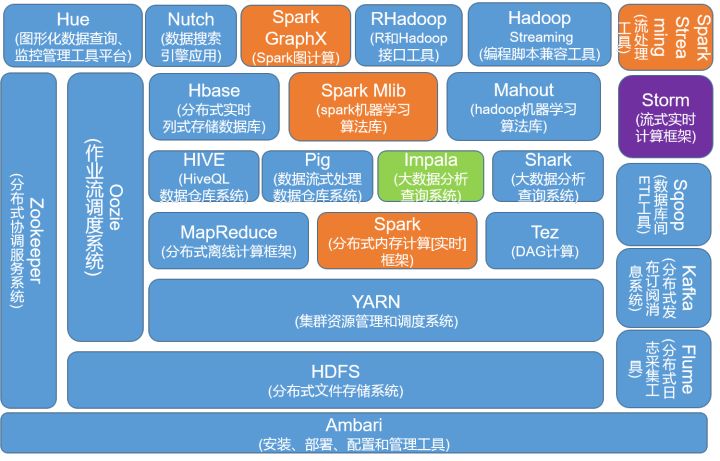


图表 ‑1 Hadoop技术生态图

由于开源技术的繁荣，Hadoop生态圈中的部分组件提供了重复性功能，本文简单介绍高校大数据治理融合平台中用到的组件，至于未介绍的组件，可以到各组件官网查看详细介绍。

1. Hive：Facebook领导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，并提供完整的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。像一些data scientist 就可以直接查询，不需要学习其他编程接口。
2. Zookeeper：一个分布式的，开放源码的分布式应用程序协调服务，是Google的Chubby一个开源的实现。

除了Hadoop为核心的大数据组件，还有其他大数据技术组件，如Spark等基于内存的大数据生态圈，以下这些大数据组件是日常大数据工作中经常会碰到的，每个组件大概的功能：a.蓝色部分，是Hadoop生态系统组件，黄色部分是Spark生态组件，虽然他们是两种不同的大数据处理框架，但它们不是互斥的，Spark与hadoop 中的MapReduce是一种相互共生的关系。Hadoop提供了Spark许多没有的功能，比如分布式文件系统，而Spark 提供了实时内存计算，速度非常快。Spark并不是一定要依附于Hadoop才能生存，除了Hadoop的HDFS，还可以基于其他的云平台。



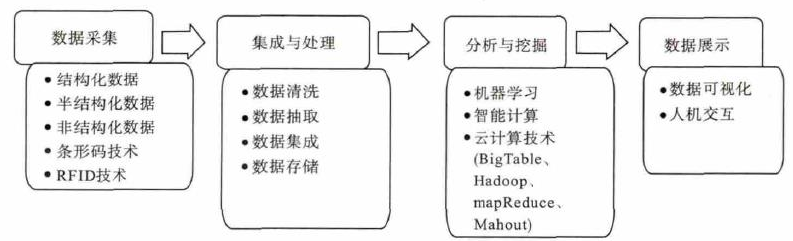
图表 ‑2 大数据技术生态图

Spark在崛起，hadoop和Storm中的一些组件在消退。HSQL未来可能会被Spark SQL替代，现在很多企业都是HIVE SQL和Spark SQL两种工具共存，当Spark SQL逐步成熟的时候，就有可能替换HSQL；MapReduce也有可能被Spark 替换，趋势是这样，但目前Spark还不够成熟稳定，还有比较长的路要走；Hadoop中的算法库Mahout正被Spark中的算法库MLib所替代。

## 2.2高校大数据技术

高校大数据是大数据在高校领域的应用，是教育大数据、智慧校园的集合；教育信息化的高度发展，产生了大量记录高校日常管理和教学工作的庞大数据[28]，智慧校园的急切发展也对高校大数据提出了新要求：研究高校大数据治理关键技术、构建高校大数据治理平台并深入实践高校大数据应用刻不容缓。

大数据的处理流程与一般数据的处理过程类似，可以定义为在合适工具的辅助下对广泛异构的数据源进行抽取和集成，将结果按照一定的标准统一存储，利用合适的数据分析技术对存储的数据进行分析从中提取有益的知识，并利用恰当的方式将结果展现给终端用户。具体来说可以分为数据抽取与集成、数据分析和数据展示[29-31]。



图表 ‑3 大数据处理流程

## 2.3 数据采集与建模技术

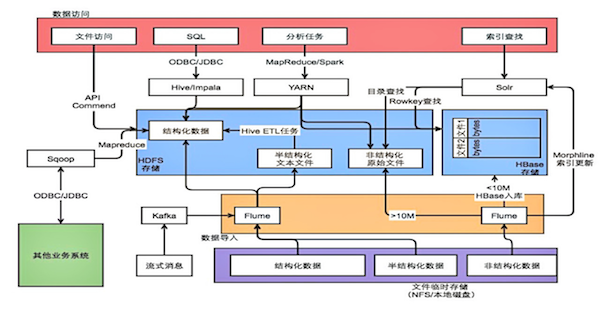
数据采集是大数据流程的第一步，也是最关键的一步。数据采集的质量和数量，直接影响到后续操作的价值。数据采集的挑战也变的尤为突出。这其中包括：

* 数据源多种多样
* 数据量大，变化快
* 如何保证数据采集的可靠性的性能
* 如何避免重复数据
* 如何保证数据的质量

数据采集的数据源通常来自：日志文件、业务系统数据库及Web爬虫，针对这些数据源源特征，国内主要流行如下工具：

Sqoop官网：<http://sqoop.apache.org/> 主要用户采集各高校信息化系统存储到MySQL等关系型数据库中的数据。

高校数据采集主要从高校相关业务系统中采集数据，这些数据主要存储在各业务信息系统的关系型数据库及系统日志文件中，因此实施时，推荐选择Sqoop及Flume工具。另外，根据业务类型对数据进行处理，清洗成规范格式加载到数据仓库，在每个处理的过程进行数据质量处理。提供 ETL 工具，对数据进行抽取与样本分析、数据转换、数据匹配等， 定制表关联规则，包括数据人工处理，以便于进行数据建模，包含制作数据模型、关联模型等。数据采集架构图如下所示：



图表 ‑4 数据采集流程

大数据中的数据建模，可以通俗的理解为对完成数据ETL(提取、转换、加载)任务、格式化、质量监控、清洗及分类组织等工作，并在此基础上建立用于数据分析的数据仓库。

高校大数据建模工作主要指：

1. 对现有的各种高校业务数据进行采集、抽取、清洗、加工和整理，包括清理源数据中的噪声数据和无关数据、处理遗漏数据和清洗脏数据和空缺值、识别删除孤立点、实现数据的汇集，确保数据的质量和高可用性；制定统一、标准规范的数据模型。
2. 提供各类方便的支持大数据量的数据加载、转换、传输工具软件。支持访问不同的数据库和文件系统；数据的清洗、转换和传输必须满足时间要求，能够在规定的时间范围内完成；支持增量加载，只把自上一次加载以来变化的数据加载到数据仓库。

开发大数据仓库，与原有数据交换平台打通，实现综合数据导入；同时存储大数据业务要求的其他非结构化数据，包括网络日志、相关互联网数据等。大数据仓库需要包括各类采集库、统计库与主题库，存储的数据包括本项目建设需要采集的相关结构化、半结构化和非结构化数据。技术上需要采用分布式 Hadoop 系统，以及关系型数据库结合的模式，通过确定主题、确定量度、确定粒度、确定维度和创建事实表等五方面的工作对数据进行备份存储和历史数据归档，并通过标准化数据接口为上层应用提供服务。数据仓库需要能对流式数据、非结构化数据提供周期性的预处理功能，预处理时间周期可根据各业务需求进行灵活设置。

## 2.4 数据存储

数据存储主要指通过选择、设计大数据存储介质(硬盘、内存、磁阵)、存储格式(结构化、非结构化)、存储方式(文件、分布式文件、SQL、NOSQL)等因素，使得对大数据的存储、查询等需求更快速、高效和高可用。

大数据存储和管理发展过程中出现了如下几类大数据存储和管理数据库系统：分布式文件存储、NoSQL数据库。

### 2.4.1 分布式文件存储

分布式文件存储的特点之一是为了解决复杂问题而将大任务分解为多项小任务，通过让多个处理器或多个计算机节点并行计算来提高解决问题的效率。

分布式文件系统能够支持多台主机通过网络同时访问共享文件和存储目录，大部分采用了关系数据模型并且支持SQL语句查询。为了能够并行执行SQL的查询操作，系统中采用了两个关键技术：关系表的水平划分和SQL查询的分区执行。

水平划分的主要思想是根据某种策略将关系表中的元组分布到集群中的不同节点上，由于这些节点上的表结构是一致的，因此便可以对元组并行处理。在分区存储关系表中处理SQL查询需要使用基于分区的执行策略。

分布式文件系统可通过多个节点并行执行数据库任务，提高整个数据库系统的性能和可用性。其主要缺点为缺乏较好的弹性，并且容错性较差。

比较流行的分布式文件系统包括：Hadoop HDFS、MogileFS、FastDFS、GlusterFS、TFS以及GoogleFS。目前，业界用得较多的是Hadoop HDFS.

Hadoop HDFS：Hadoop是一个由Apache基金会所开发的分布式系统基础架构。用户可以在不了解分布式底层细节的情况下，开发分布式程序。充分利用集群的威力进行高速运算和存储。Hadoop实现了一个分布式文件系统（Hadoop DistributedFile System），简称HDFS。HDFS有高容错性的特点，并且设计用来部署在低廉的（low-cost）硬件上；而且它提供高吞吐量（high throughput）来访问应用程序的数据，适合那些有着超大数据集（large data set）的应用程序。HDFS放宽了（relax）POSIX的要求，可以以流的形式访问（streaming access）文件系统中的数据。

Hadoop的框架最核心的设计就是：HDFS和MapReduce。HDFS为海量的数据提供了存储，则MapReduce为海量的数据提供了计算。具有如下优点：

高可靠性。Hadoop按位存储和处理数据的能力值得人们信赖。高扩展性。Hadoop是在可用的计算机集簇间分配数据并完成计算任务的，这些集簇可以方便地扩展到数以千计的节点中。

高效性。Hadoop能够在节点之间动态地移动数据，并保证各个节点的动态平衡，因此处理速度非常快。

高容错性。Hadoop能够自动保存数据的多个副本，并且能够自动将失败的任务重新分配。

低成本。与一体机、商用数据仓库以及QlikView等数据集市相比，hadoop是开源的，项目的软件成本因此会大大降低。

Hadoop带有用Java语言编写的框架，因此运行在 Linux生产平台上是非常理想的。Hadoop上的应用程序也可以使用其他语言编写，比如 C++。Hadoop是目前最流行的的分布式文件系统之一，能找到的资料相当丰富。

### 2.4.2 NoSQL数据库

传统关系型数据库在数据密集型应用方面显得力不从心，主要表现在灵活性差、扩展性差、性能差等方面。而NoSQL摒弃了传统关系型数据库管理系统的设计思想，采用了不同的解决方案来满足扩展性方面的需求。由于它没有固定的数据模式并且可以水平扩展，因而能够很好地应对海量数据的挑战。相对于关系型数据库而言，NoSQL最大的不同是不使用SQL作为查询语言。NoSQL数据库主要优势有：避免不必要的复杂性、高吞吐量、高水平扩展能力和低端硬件集群、避免了昂贵的对象-关系映射。

NoSQL纷繁复杂，NoSQL开源技术多达数十种[wiki NoSQL],鉴于作者的精力，完全了解并实验这些数据库不太实际，本文选择国内用得较多、资源文档丰富的Hive作为数据库存储组件。

## 2.5 数据计算与挖掘

鉴于大数据价值密度低、商业价值高的特点，为提高数据质量和可信度，要求从大量的、不完全的、有噪声的、模糊的、随机的实际应用数据中，通过分析数据结构、类型及数据间的关联度，综合运用统计和机器学习，从数据库管理系统的大数据中提取出隐含其中的潜在信息和知识，这就是数据分析与挖掘。

数据分析是在强大的支撑平台上运行分析算法发现隐藏在大数据中潜在价值的过程, 例如隐藏的模式 (pattern) 和未知的相关性[19]。 根据处理时间的需求, 大数据的分析处理可以分为两类：

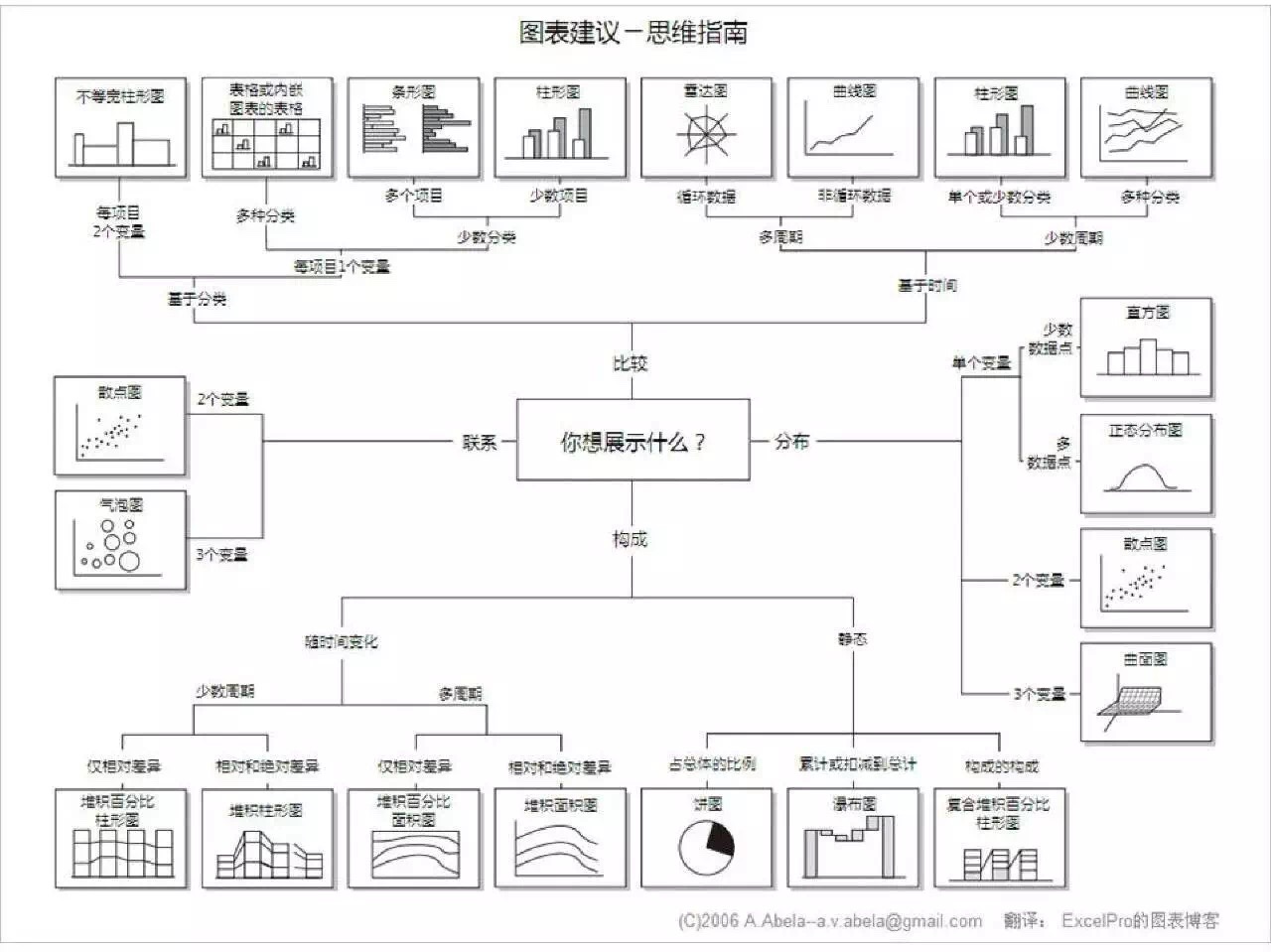
* 1. 流式处理: 流式处理假设数据的潜在价值是数据的新鲜度(freshness) [23], 因此流式处理方式应尽可能快地处理数据并得到结果. 在这种方式下, 数据以流的方式到达. 在数据连续到达的过程中, 由于流携带了大量数据, 只有小部分的流数据被保存在有限的内存中. 流处理理论和技术已研究多年,代表性的开源系统包括Storm, S4 [24] 和 Kafka [25]. 流处理方式用于在线应用, 通常工作在秒或毫秒级别。
  2. 批处理: 在批处理方式中, 数据首先被存储, 随后被分析. Map Reduce 是非常重要的批处理模型.Map Reduce 的核心思想是, 数据首先被分为若干小数据块 chunks, 随后这些数据块被并行处理并以分布的方式产生中间结果, 最后这些中间结果被合并产生最终结果. Map Reduce 分配与数据存储位置距离较近的计算资源, 以避免数据传输的通信开销. 由于简单高效, Map Reduce 被广泛应用于生物信息、web 挖掘和机器学习中。
  3. 实时处理：离线式的批处理框架 MapReduce 已经不能满足业务，大量的场景需要实时的数据处理结果来进行分析，决策。实时计算适用于这种对历史数据依赖不强，短时间内变化较大的数据。实时计算框架SparkStreaming、Storm、Flink、Blink、Duroid。

数据挖掘也叫数据开采，数据采掘等，是按照既定的业务目标从海量数据中提取出潜在、有效并能被人理解的模式的高级处理过程．在较浅的层次上，它利用现有数据库管理系统的查询、检索及报表功能，与多维分析、统计分析方法相结合，进行联机分析处理(OLAP)，从而得出可供决策参考的统计分析数据．在深层次上，则从数据库中发现前所未有的、隐含的知识．OLAP的出现早于数据挖掘，它们都是从数据库中抽取有用信息的方法，就决策支持的需要而言两者是相辅相成的。

OLAP可以看作一种广义的数据挖掘方法，它旨在简化和支持联机分析，而数据挖掘的目的是便这一过程尽可能自动化。

## 2.6 可视化技术

数据呈现即数据可视化，数据可视化是利用图表、动画、色彩等展示数据的方法。数据可视化能够有效提高人们对数据的直观理解、提高数据探索、分析的效率。数据呈现的思维导图如下图所示：



图表 ‑5 数据可视化思维导图

业界有大量的大数据可视化工具，如Tableau、数字冰雹、FineBI、PentahoBI、PowerBI、SuperSet等，本研究主要聚焦于高校大数据治理融合，大数据的可视化采用免费的商业软件FineBI.

## 2.7大数据管理系统

大数据管理系统，通常主要为部署在集群内的服务提供统一的集群管理能力。支持大规模集群的安装部署、性能监控、告警、用户管理、权限管理、审计、服务管理、健康检查、日志采集、升级和补丁等功能。同时，提供元数据管理功能，具备数据源配置、数据字段映射、状态监控、向导式部署、自动升级、数据质量管理等功能。业界提供大数据管理系统的商业软件有Cloudera Distribute Hadoop(CDH)、HDP、MapR、FushionInsight HD。本文选择网络上文档比较丰富的CDH大数据管理平台。

## 2.8 本章小结

本章介绍了探讨了高校大数据技术、Hadoop技术、数据采集建模技术、数据存储、数据计算与挖掘、数据可视化技术及大数据平台管理系统等高校大数据治理融合平台相关的方法与技术，通过对这些方法与技术的深入研究学习，为高校大数据采集、融合和分析提供理论指导和工具准备工作。

# 第三章 高校大数据治理融合需求分析及总体架构

## 3.1 高校大数据治理融合总体需求

#### 3.1.1 平台定位及建设的指导思想

高校中汇聚着大量的数据，从学生角度来看，包括联系方式等基本信息，食堂消费、住宿等生活数据，课后作业、借阅图书、成绩等学习数据，参与的社团、竞赛、讲座等第二课堂数据；从教师角度来看，包含教学任务、课件等教学信息，论文著作、科学研究数据等科研数据；从管理者的角度来看，包含学校的资产信息、师资信息、招生就业信息等。同时随着移动互联网以及物联网等新技术的兴起，学校师生主动产生和由设备自动收集的数据越来越多，如微博、微信等社交数据，各类搜索点击记录数据等。利用大数据治理技术可以高效地治理融合高校大数据。

鉴于数据源及个人精力，本课题拟针对高校最重要的主体-学生为研究对象，利用大数据技术融合分析学生相关大数据，如图书借阅数据、教学数据、医院数据、就业数据、校园网数据、一卡通数据、宿舍数据等，构建学生画像，根据学生的专业、兴趣爱好等实现个性化书籍推荐及学生管理指导。

首先，深入调研分析学生高校活动及相关数据，采用UML统一建模语言描述平台需求及关联流程；其次，明确平台范围边界，确定平台需要实现的具体功能及使用的技术；最后，整理出平台的总体架构图及功能模块图；

### 3.2.1 图书数据融合

高校图书管理涉及实体主要包括：图书馆、纸质图书、学生、教职工、电子资源、工作人员；图书管理信息系统会产生及记录大量实体数据及实体间各种关联活动数据，与学生相关的数据主要由：图书检索与借阅数据、电子资源下载数据、图书馆打卡数据；融合分析学生学校其他活动数据，可以实现：学生学习热度分析、学生读者类型分类、个性化图书推荐及可视化统计分析。

#### 3.2.1.1学生学习热度分析

根据全校学生每月、每学期检索及借阅图书的数量、图书馆打卡时长统计分析，可以识别出全校学生的学习热度。

#### 3.2.1.2 学生读者类型分析

根据学生历史借阅图书及下载电子资源的聚类分析，关联分析学生平时喜欢阅读什么类型的书籍及学习成绩的关系，从而为学生提供图书推荐和学习指导。

#### 3.2.1.3 个性化图书推荐

通过学生检索及借阅记录为学生借阅图书、查询电子资源提供多维度个性化推荐服务，节省读者查找图书资源的时间，提供图书管理系统的服务质量；

纵向推荐算法：根据学生的专业及历史借阅记录关联往届优秀毕业生的专业及历史借阅记录为学生推荐阅读书籍；

【横向推荐算法：根据其他名牌高校同专业学生借阅情况及优秀毕业生借阅情况推荐阅读书籍；】由于数据源问题，该算法可能不能实现。

热门书籍推荐：全国、全国高校、全校、全专业、优秀毕业生阅读榜单；

聚类推荐：相似专业的、相似阅读习惯、相似兴趣爱好的同期及往届优秀毕业生推荐。

#### 3.2.1.3学生图书借阅统计可视化

高校图书借阅总量、院系图书借阅量排行、专业图书借阅量排行、年级图书借阅量排行、班级图书借阅量排行、欠费数额排行榜、欠费次数排行榜、学生图书馆投入时间分析模型、历届优秀毕业生图书馆投入时间聚类关联分析等。

#### 3.2.1.5相关融合数据集

本校学生检索及借阅记录、书籍编目数据、学生查找图书视频录像及活动路径、本校优秀毕业生数据集、全国热门书籍数据、本校院系专业数据、本校学生就业情况数据、本校选修课程数据、本校学生成绩数据等。

### 3.2.2 教学分析

教学活动涉及主体：教师、课程、学生；教学管理系统主要管理教师授课、课程信息、学生选课等信息及相关活动数据。目前高校教学管理系统只是简单的记录学生选课、教师授课的信息，不能很好的反馈教学问题、教学效果、专业设计是否合理等问题。

本课题不研究教学活动中教师及课程相关数据；

教学分析主要研究学生相关数据，如学生选课数据、学生课程成绩、学生出勤情况、作业完成情况、学生课堂互动数据、学生积极性等因素中找出其中的内在联系，分析学生的课程喜好度、课程学习能力、就业相关性。

#### 3.2.2.1 学生课程喜好度分析

根据学生课程课堂互动数据、出勤情况、作业完成情况分析学生课程喜好程度。

纵向分析：对比学生所有选课中课堂互动数据、出勤情况、作业完成情况。

横向分析：对比所有学生同一课程中课堂互动数据、出勤情况、作业完成情况。

#### 3.2.2.2 学生课程学习能力分析

根据学生课程课堂互动数据、出勤情况、作业完成情况分析学生课程喜好程度及其与课程成绩的关系。

纵向分析：对比学生所有选课中课堂互动数据、出勤情况、作业完成情况其与课程成绩的关系。

横向分析：对比所有学生同一课程中课堂互动数据、出勤情况、作业完成情况其与课程成绩的关系。

#### 3.2.2.3 学生课程大数据分析

通过对在读及历届学生日常生活、学习、工作中产生的海量数据进行汇总分析，这些数据可以是课程选择、学习成绩、参加的竞赛及获奖情况、社团活动、实习情况等学习信息，性格特点、消费状况、课外活动，论坛发帖等生活信息，以及就业单位、就业岗位、工作福利待遇、晋升状况等工作信息。通过综合分析这些信息，可以找出在读学生成绩分布、历届学生辍学、就业等行为共同特点。进一步，通过比较在读与历届学生的相关属性，分析发现是否存在面临辍学、需要补助或者心理辅导学生，及时对学生和班主任老师给出预警，以便其在后续的学习和生活中加以关注。同时也可以计算出学生之间（往届和应届）在就业方面相似度，为即将毕业的学生推荐适合的就业单位和岗位，提供个性化的服务。

#### 3.2.2.4相关融合数据集

本校学生图书数据、教学数据、就业数据、一卡通数据、宿舍数据、成绩数据、上网数据、运动数据等。

### 3.2.3 学生画像

通过对学校在校学生的综合分析，包括本科生、研究生、行为轨迹、就业、学生关系、学生预警、招生等内容，为学校学生管理工作提供预警分析和决策支持。

用于辅助学校将学生管理工作者从日常的具体事务性工作中解脱出来，解决运用传统手段难以获取各类统计数据的问题，更重要的是能够从学生管理工作中挖掘出有价值的信息，经过过程性和综合性的分析，找到学生各种行为之间的内在联系，思考背后的逻辑关系，并做出恰当的教学、管理决策，用以提升学校创新创业、学生服务工作的水平。

#### 3.2.3.1 学生成绩-行为分析

以成绩为核心，利用一卡通追踪学生行为轨迹，通过对学生吃饭、上网、出行、消费、借书等行为记录，分析每名学生的学习、生活状态。同时，通过对学生日常学习状态的追踪，还可对学生的期末成绩作出预测。具体分析内容有

* 成绩与上网时长的关系
* 成绩与图书借阅量的关系
* 成绩与学习地点的关系
* 成绩与作息时间的关系

#### 3.2.3.2 沉迷网络学生分析

对网络计费的统计，按照月、周、日时长进行排名，找出上网成瘾的学生名单给班主任和辅导员，辅助其管理；

#### 3.2.3.3 逃课行为分析

根据学生上课信息，比对对应学生的上网、门禁、消费记录等，当两者有交叉的时候说明该学生迟到或者逃课的嫌疑，当学生多次出现这种情况，就需要引起注意。

#### 3.2.3.4 贫困生分析

以往高校习惯于通过学生递交申请、班主任评定和侧面了解等传统方式来确定贫困生，不同校区、不同学院还有不同的评定标准，这导致在实际的工作中，贫困生的贫困程度无法得到统一的评价。通过高校大数据治理融合分析平台，可以分析“一卡通”里的食堂就餐刷卡数据、手机使用状况、网上消费情况，从而评定与识别出真实贫困学生，从而使贫困生的评定工作 得到有序、统一、高效地开展。

#### 3.2.3.5 就业分析

基于高校大数据治理融合分析平台，通过“一卡通”分析到学生的选课情况、专业学习情况以及平日的兴趣爱好，然后计算学生之间的相似度，为即将毕业的学生推荐适合的就业单位和岗位，提供个性化的服务；同时，分析未能及时就业的学生，从中找出其共同的特点，再通过比较在校学生的相关属性，及时对学生给出预警，以便其在后续的学习和生活中加以改进。

#### 3.2.3.6 学生失联预警

利用学生的一卡通消费与上网记录来判断学生的在校情况，甄别出非假期期间长时间不在校学生，可以更迅速及时发现学生失联情况，归类成异常学生名单，提供给学校相关部门，以更好地保障学生安全。

### 3.2.4 医院分析

高校医院相关实体包含：学生、医生、医务人员、医疗用品、医疗设备。高校医务管理系统主要记录这些实体信息及实体间活动关系信息。融合分析医院大量数据可以合理布局医疗用品及设备、降低医疗成本、科学评价及指导医务人员、提高医务效果及质量。

本文主要研究学生相关的医院数据，如学生病例数据、学生医药数据、学生住院数据等，关联分析学生饮食、运动数据；监控学生身心健康。

### 3.2.5 就业分析

根据历届学生毕业就业相关数据(如Officer数量与质量、工作企业知名度、工资水平、职位类型、工作地等)，融合学生其他数据(如成绩、专业课程、学校生活轨迹、图书借阅、教学情况等)数据，分析学生就业影响因素。为在校学生提高就业水平，提供优化指导(如图书推荐、社团活动、公益活动、体育锻炼等)。

## 3.2 高校大数据治理融合平台总体架构

高校大数据治理融合平台总体架构如下图所示：【将图修改简单】

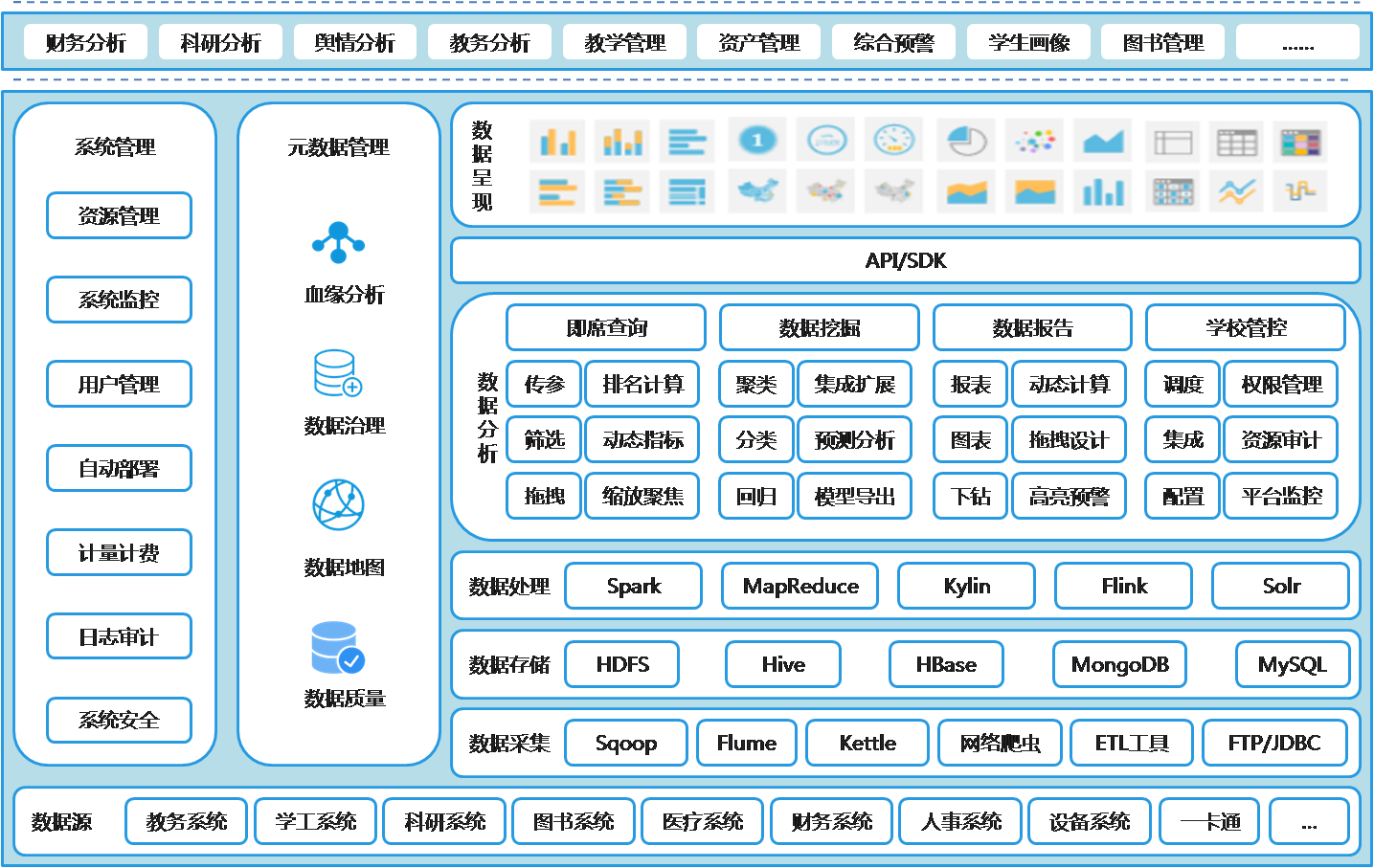


图 3‑‑1 高校大数据治理融合架构图

如图3-1所示，高校大数据治理融合平台从下至上分为三层，分别为：各类异构数据源、大数据治理融合层和大数据专题分析层。

异构数据源层旨在采集高校各类信息化系统中存储的数据，并从网络上通过爬虫采集相关辅助数据，集成融合各种异构的结构化和非结构化的数据，为分析系统提供数据支撑，其中结构化数据主要包括：招生管理系统数据、教学系统数据、医务系统数据、选课系统数据、上网管理系统数据、科研系统数据、财务管理系统数据、图书管理系统数据、设备管理系统数据、教师相关数据与学生基本数据(选课数据、考勤数据)、一卡通数据。非结构化数据主要有：信息化系统日志、微博、微信、校园等日志或媒体数据，同时还有其它和校园数据分析有关系的各类数据。这些异构数据源是构成校园大数据分析平台的数据基础。

大数据治理融合层又分为系统管理、元数据管理、数据采集、数据存储、数据处理、数据分析、数据呈现等功能，各功能说明如下：

系统管理：负责整个高校大数据治理融合平台硬件资源管理、开源大数据组件管理(部署、状态监控、升级)、用户管理、系统安全等功能。

元数据管理：负责数据血缘分析、数据治理、数据质量及数据地图等功能，保证数据的安全、可控、可追溯。

数据采集：为各类异构数据提供采集工具、研制适配接口，从而与校内的各系统对接获取全面的高校数据。高校各系统的结构化数据存储在关系型数据库及文件中，可以通过Sqoop、Kettle、FTP等开源工具提取；高校信息化系统的日志数据可以通过Flume/FTP或原开发商提供的开放接口提取。网络爬虫采集提取微博、微信、校园论坛等网络数据。所有采集的数据通过ETL工具进行适配、转换、数据建模之后存储到数据存储模块中。

数据存储：由Hive、Hbase、MySQL、HDFS等组件构成。其中MySQL主要用于存储元数据、权限管理数据、用户数据、监控数据等平台基础静态数据；HDFS分布式文件系统存储半结构化或非结构化的未经过ETL处理的原始数据；HBase、Hive是位于HDFS之上的数据仓库，其主要功能是提供快速查询、OLAP支持、SQL能力。

数据处理：主要提供分布式数据计算能力，包括离线计算、实时计算和全文检索。离线计算由Hadoop MapReduce、Spark、Kylin支撑；实时计算由SparkStreaming、Flink支撑。

数据分析：封装了即席查询、数据报告、数据挖掘、学校管理等功能及其通用接口/API，使用插件框架为系统提供高扩展性，以满足新增业务功能的分析、查询、处理及展示功能。

数据可视化：使用Java/React/echarts等Web技术，调用API/SDK模块提供的接口实现高校大数据的可视化展示。主要包括智能报表、用户画像、各种图表、用户管理界面、数据建模界面、数据分析界面及平台管理界面等。

大数据分析层基于高校大数据治理融合层提供基础的分析框架封装高校大数据应用专题分析包。分析框架提供基于主流的Hadoop实现的MapReduce编程模型、Spark编程模型、Spark SQL/HSQL模型。定制的专题分析包包括但不限于：财务分析、科研分析、教务分析、教学管理、资产管理、综合预警、学生画像、图书管理、战略决策、招生就业分析、助学分析等高校大数据专题分析。

## 3.3 本章小结

本章介绍了高校大数据治理融合总体架构，深入分析调研高校各信息化系统及数据源，总结分析了16中高校大数据应用场景，并结合总体架构提出了数据流程及关键算法。

# 第四章 高校大数据治理融合平台设计

## 4.1 高校大数据治理融合基础平台

高校大数据融合治理平台至少包含监控、调度、管理、数据采集、数据处理等功能模块，如下图所示：



‑1 高校大数据融合治理平台

为了简化平台的开发及设计工作，系统监管、系统管理及调度系统功能模块由Cloudera Distubtion Hadoop(CDH5.9)开源大数据平台提供支撑，数据采集工作由Sqoop、Flume、Kettle等开源软件实现，数据处理功能CDH5.9平台中集成的impala、Hive、Hbase、Spark、Kafka等开源大数据组件实现。设计完成高校大数据融合治理平台的主要工作包括：

1. 申请服务器，完成平台设备组网；安装部署CDH5.9平台、并熟练掌握使用平台集成的各个大数据组件
2. 梳理、理解高校各信息化系统中的数据，并使用Sqoop、Flume、Kettle等大数据组件将数据统一融合、导入到大数据平台
3. 利用大数据分析技术挖掘高校大数据中存储的数据价值

### 4.1.1 平台物理架构

#### 4.1.1.1 物理组网



#### 4.1.1.2 主机配置列表

高校大数据实验环境中，主机配置列表如下：

|  |  |  |  |
| --- | --- | --- | --- |
| Ip | 类型 | 配置 | 作用 |
| 172.16.18.41 | 物理机 | DELL R730 8核 96G  8\*300G /Redhat 6.4 | CDH管理软件、大数据组件安装包、YUM源、MySQL、MongoDB、Kylin |
| 172.16.18.51 | 物理机 | DELL R730 8核 96G  8\*300G /Redhat 6.4 | Spark集群、HDFS集群、Kafka集群 |
| 172.16.18.52 | 物理机 | DELL R730 8核 96G  8\*300G /Redhat 6.4 | Spark集群、HDFS集群、Kafka集群 |
| 172.16.18.150-4 | 虚拟机 | 2核 32G 600G /Redhat 6.4 | Hive集群、Zookeeper集群、Flume |

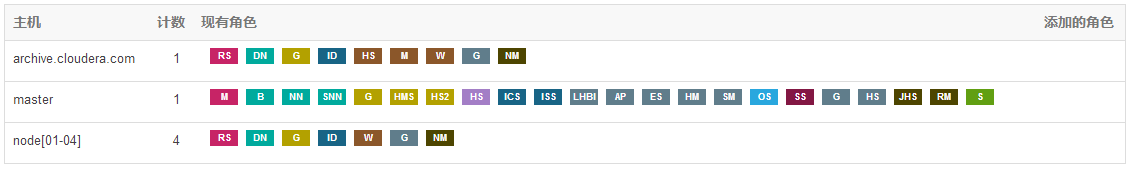
#### 4.1.1.3 Host主机名配置

所有主机中的/etc/hosts文件内容如下：

|  |
| --- |
| 172.16.18.51 master  172.16.18.52 archive.cloudera.com  172.16.18.150 node01  172.16.18.151 node02  172.16.18.152 node03  172.16.18.153 node04  172.16.18.41 cm #Cloudera Manager |

#### 4.1.1.4 大数据服务配置

CDH中查看集群服务器中大数据服务配置列表：



服务说明：

|  |  |
| --- | --- |
| 服务图标 | 服务说明 |
|  | HBase RegionServer |
|  | HBase Master |
|  | HDFS DataNode |
|  | HDFS Balancer |
|  | HDFS NameNode |
|  | HDFS SeconderyNameNode |
|  | Hive Gateway |
|  | Hive Metastore Server |
|  | Hive Server2 |
|  | Impala StateStore |
|  | Impala Deamon |
|  | Impala CatalogServer |
|  | Spark HistoryServer |
|  | Spark Worker |
|  | Spark Gateway |
|  | Spark HistoryServer |
|  | Yarn NodeManager |
|  | Yarn JobHistoryServer |
|  | Yarn ResourceManager |
|  | Hue Server |
|  | Oozie Server |
|  | Solr Server |
|  | Zookeeper Server |

### 4.1.2 平台管理安装

大数据应用技术平台选择使用CDH发行版5.9.0作为基础构建管理平台；安装步骤参考CDH官网[4.1.2官网]及国内网络资料[4.1.2安装参考]

#### 4.1.2.1 安装环境准备[4.1.2安装参考]

安装参考文档中步骤，首先准备安装环境，主要包括安装服务器操作系统、虚拟机创建、域名及IP网络设置、防火墙设置。这一步主要保证服务器之间能够互联互通、服务器能够连接互联网(CDH官网)。从官网下载离线安装包：

Cloudera Manager 5.9 http://archive­primary.cloudera.com/cm5/cm/5/cloudera­manager­el6­cm5.9.0\_x86\_64.tar.gz

CDH5.9 主文件http://archive­primary.cloudera.com/cdh5/parcels/5.9.0.23/CDH­5.9.0­1.cdh5.9.0.p0.23­el6.parcel

CDH5.9 sha文件http://archive­primary.cloudera.com/cdh5/parcels/5.9.0.23/CDH­5.9.0­1.cdh5.9.0.p0.23­el6.parcel.sha1

manifest 文件http://archive­primary.cloudera.com/cdh5/parcels/5.9.0.23/manifest.json

下载后将这些包上传到master机器的/opt/cdh目录下

#### 4.1.2.2 NTP配置

大数据平台的大数据分布式大数据组件要求集群服务器时间几乎同步。NTP时间同步，不一定分秒不差，但要保持在一个合理的范围之内，不然服务运行会报错。时间同步有两种方式，一种是7台均同步单独的时钟服务器，一种是使用master本身作为时钟服务器，其它节点同步它，接下来采用第二种方式。

1. 修改配置文件

打开master机器 /etc/ntp.conf 文件，其它地方可以注释，留下如下关键信息，保存退出。

|  |
| --- |
| driftfile /var/lib/ntp/drift  restrict 127.0.0.1  restrict ‐6 ::1  restrict default nomodify notrap  server 127.127.1.0  fudge 127.127.1.0 stratum 8  includefile /etc/ntp/crypto/pw  keys /etc/ntp/keys |

1. 启动服务

执行如下命令启动NTP服务，如果服务之前己启动，则重启服务。ntp与自身进行同步需要一小段时间，大约需要5分钟。

|  |
| --- |
| service ntpd start  chkconfig ntpd on |

使用命令 ntpstat 检查服务启动后自身是否同步完成，如出现以下信息则完成。

|  |
| --- |
| synchronised to local net at stratum 9  time correct to within 12 ms  polling server every 64 s |

1. 客户端同步

服务己启动，完成其它节点机器同步了，使用命令 ntpdate 172.16.18.150 查看结果，如出现以下信息则同步完成。

|  |
| --- |
| 24 Feb 13:37:35 ntpdate[13675]: adjust time server 172.16.18.150 offset 0.068598 sec |

#### 4.1.2.3 实用shell

后续安装过程，考虑到各个机器间需要拷贝文件，如果到各个机器去执行拷贝命令，需要分别到6台集群机器执行，为了简化操作，使用用脚本的方式批处理。

在master机器root目录下，创建一个nodes 文件，该文件存放集群中所有集群的域名，文件内容如下：

|  |
| --- |
| master  archive.cloudera.com  node01  node02  node03  node04  cm |

在master机器root目录下，编辑一个 scp.sh 文件，文件内容如下，输入需要传送的源文件路径与目标机器的存放路径两个参数，路径都是绝对路径。

|  |
| --- |
| #!/bin/bash cat nodes | while read host do { scp ‐r $1 $host:$2 }&wait done |

#### 4.1.2.4 JAVA环境安装

大数据平台的大部分开源组件使用Java语言编写，需要安装较高版本的java JDK。新安装的Redhat6.0系统自带的OpenJDK版本较低，使用 rpm -qa|grep java查看。如果有，则使用命令 yum -y remove xxxx删除，下载JDK-1.8.0\_112，并将下载好的JDK包解压放至所有服务器的/opt/java 目录下，调置环境变量，编辑文件 /etc/profile 添加如下内容

|  |
| --- |
| export JAVA\_HOME=/opt/java  export PATH=$JAVA\_HOME/bin:$PATH |

拷贝JAVA文件与profile文件至其它节点，注意一点，其它节点不需要己经存在 java 目录，此脚本会自动创建其目录

|  |
| --- |
| ./scp.sh /opt/java/ /opt/java/  ./scp.sh /etc/profile /etc |

设置全局变量并生效，在五台机器上均执行如下命令

|  |
| --- |
| source /etc/profile  echo "JAVA\_HOME=/opt/java" >> /etc/environment |

检查JAVA是否安装成功，执行命令 java ‐version 出现如下结果

|  |
| --- |
| java version "1.8.0\_112"  Java(TM) SE Runtime Environment (build 1.8.0\_112‐b15)  Java HotSpot(TM) 64‐Bit Server VM (build 25.112‐b15, mixed mode) |

#### 4.1.2.5 Mysql安装

使用MySQL作为CDH管理平台的Cloudera Manager的元数据存储，只需要在master机器上安装。使用5.6.24 版本，需要下载如下安装包，然后，依次采用命令 rpm -ivh xxxxx 进行安装，如果中间出现冲突，就先删除再装：

|  |
| --- |
| mysql‐libs‐5.1.66‐2.el6\_3.x86\_64.rpm  MySQL‐shared‐5.6.24‐1.el6.x86\_64.rpm  MySQL‐devel‐5.6.24‐1.el6.x86\_64.rpm  MySQL‐server‐5.6.24‐1.el6.x86\_64.rpm  MySQL‐client‐5.6.24‐1.el6.x86\_64.rpm |

1. 启动MYSQL服务

|  |
| --- |
| chkconfig mysql on  service mysql start |

1. 修改初始密码

MYSQL启动后，默认其实己经生成了默认密码，密码保存在 /root/.mysql\_secret 文件中，为了安全需要改默认密码，示例修改成 123456，执行如下命令。

|  |
| --- |
| # service mysql stop  # mysqld\_safe ‐‐user=mysql ‐‐skip‐grant‐tables ‐‐skip‐networking&  # mysql ‐u root mysql  mysql> use mysql;  mysql> UPDATE user SET Password=PASSWORD('123456')where USER='root';  mysql> FLUSH PRIVILEGES;  mysql> quit  # service mysql restart  # mysql ‐u root –p  Enter password: 123456  mysql>SET PASSWORD=PASSWORD('123456'); （重置密码）  mysql> use mysql; (此DB存放MySQL的各种配置信息)  Database changed  mysql> select host,user from user; (查看是否能够查询) |

1. 设置远程登陆

执行如下命令

|  |
| --- |
| mysql> Grant all privileges on \*.\* to 'root'@'%' identified by '123456' with grant option; （允许远程用户登录mysql）  mysql>flush privileges; |

MYSQL的环境己基本配置完成。

1. 创建Cloudera Manager元数据数据库

执行如下命令，这些库是集群安装时时大数据组件所需要的。

|  |
| --- |
| --hive数据库  create database hive DEFAULT CHARSET utf8 COLLATE utf8\_general\_ci;  --oozie数据库  create database oozie DEFAULT CHARSET utf8 COLLATE utf8\_general\_ci;  --hue数据库  create database hue DEFAULT CHARSET utf8 COLLATE utf8\_general\_ci; |

#### 4.1.2.6 Cloudera推荐设置

在试安装的过程，发现Cloudera给出了一些警告，如下图



设置swap空间

|  |
| --- |
| echo "vm.swappiness = 10" >> /etc/sysctl.conf  ./scp.sh /etc/sysctl.conf /etc/ |

关闭大页面压缩  
试过只设置defrag，但貌似个别节点还是会有警告，干脆全部设置，编辑 /etc/rc.local ，末尾添加如下命令

|  |
| --- |
| echo never > /sys/kernel/mm/transparent\_hugepage/enabled echo never > /sys/kernel/mm/transparent\_hugepage/defrag |

同步到各个节点机器./scp.sh/etc/rc.local/etc/，然后重启 reboot 生效。

#### 4.1.2.7 安装Cloudera Manager

1. 进入 /opt/cdh 目录，将Cloudra Manager解压到/opt目录下，不能解压到其他地方，因为cdh5的源会默认在/opt/cloudera/parcel­repo寻找，而CM可以按照个人喜好安装。

|  |
| --- |
| tar ‐zxvf cloudera‐manager‐el6‐cm5.9.0\_x86\_64.tar.gz ‐C /opt/  mkdir ‐p /opt/program  mv /opt/cm‐5.9.0/ /opt/program/  ln ‐s /opt/program/cm‐5.9.0/ /opt/cm |

1. 将CDH­5.9.0­1.cdh5.9.0.p0.23­el6.parcel 和 CDH­5.9.01.cdh5.9.0.p0.23­el6.parcel.sha1移动到/opt/cloudera/parcel­repo，这样安装时CM就能直接找到了。

|  |
| --- |
| mv CDH‐5.9.0‐1.cdh5.9.0.p0.23‐el6.parcel CDH‐5.9.0‐1.cdh5.9.0.p0.23‐el6.parcel.sha1 /opt/cloudera/parcel‐repo/ |

1. 将CDH­5.9.0­1.cdh5.9.0.p0.23­el6.parcel.sha1重命名为CDH­5.9.0­1.cdh5.9.0.p0.23­el6.parcel.sha(去掉结尾的1)，非常重要。

|  |
| --- |
| mv manifest.json /opt/cloudera/parcel‐repo/ cd /opt/cloudera/parcel‐repo/ mv CDH‐5.9.0‐1.cdh5.9.0.p0.23‐el6.parcel.sha1 CDH‐5.9.0‐1.cdh5.9.0.p0.23‐el6.parcel.sha |

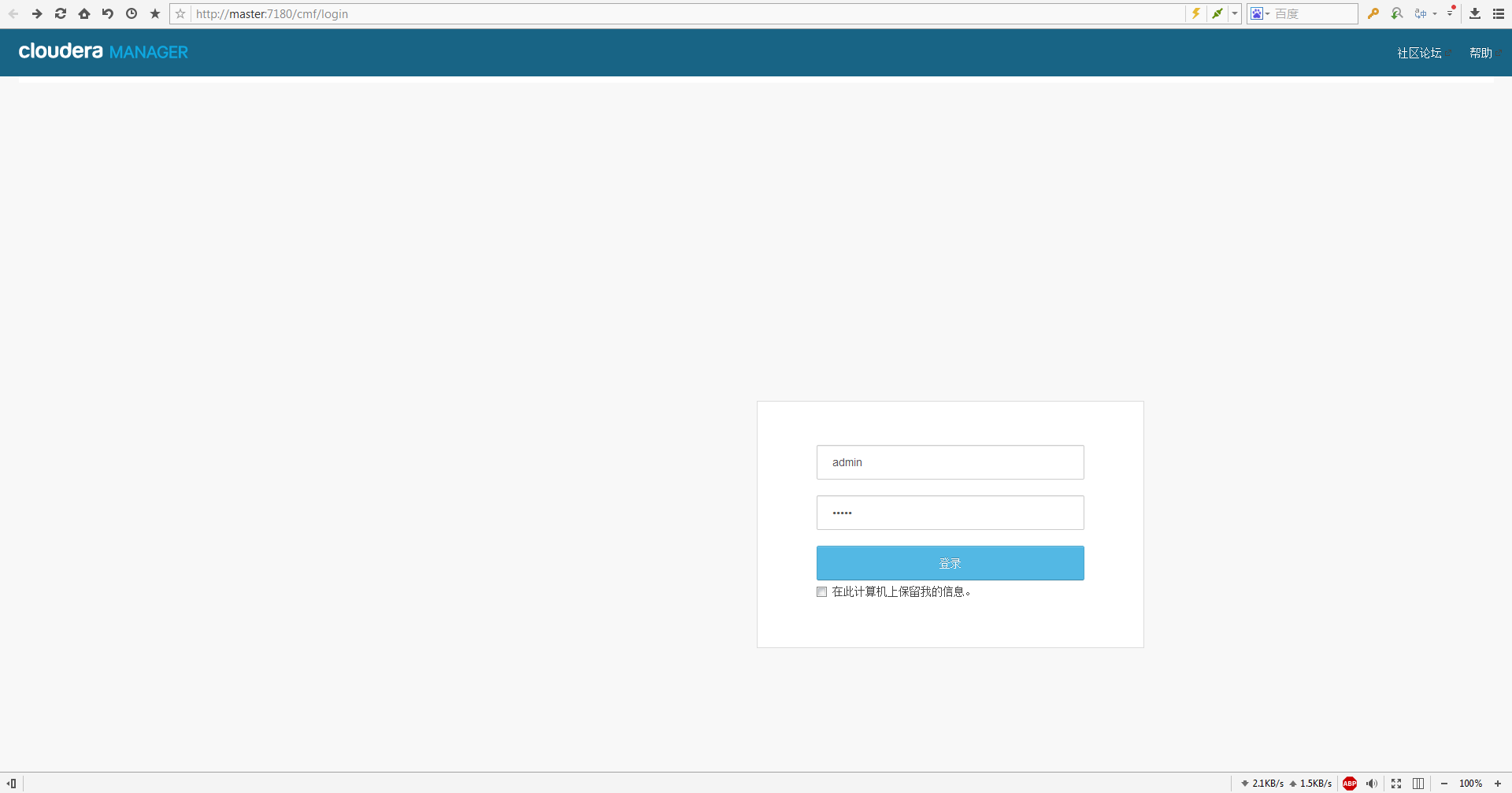
1. 修改配置文件中的server\_host，编辑

|  |
| --- |
| /opt/cm/etc/cloudera‐scm‐agent/config.ini server\_host=master |

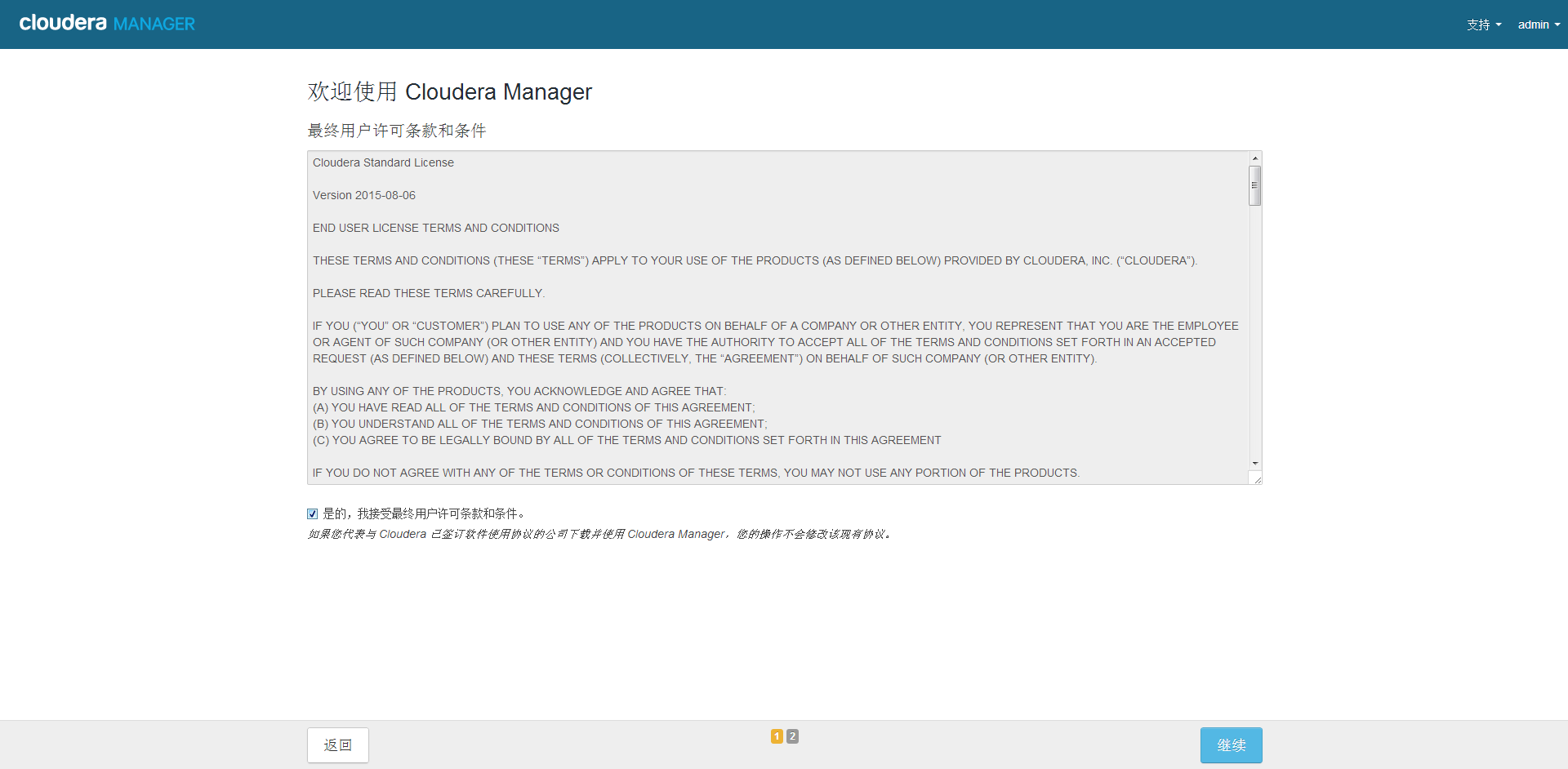
1. 将mysql的JDBC驱动放入CM的lib目录下，下载地址http://dev.mysql.com/downloads/connector/j/。解压获得mysql­connector­java­5.1.40­bin.jar上传到集群。  
   mv mysql‐connector‐java‐5.1.40‐bin.jar /opt/cm/share/cmf/lib/
2. 为CM创建数据库  
   /opt/cm/share/cmf/schema/scm\_prepare\_database.sh mysql cm ‐hlocalhost ‐uroot ‐p123456 ‐‐scm‐host  
   localhost scm scm scm
3. 为每个节点创建cloudera­scm用户，五台机均执行  
   useradd ‐‐system ‐‐home=/opt/cm/run/cloudera‐scm‐server ‐‐no‐create‐home ‐‐shell=/bin/false ‐‐  
   comment "Cloudera SCM User" cloudera‐scm接下来其实可以直接在master上启动服务安装了，但因为其它节点没有CM，最后还是通过远程的yum在线下载再安  
   装，我这设置了内部网络，其它节点是访问不了外网的，所以拷贝CM到其它节点进行完全离线安装。
4. 拷贝CM到每个节点  
   ./scp.sh /opt/program/cm‐5.9.0/ /opt/program/cm‐5.9.0/  
   ln ‐s /opt/program/cm‐5.9.0/ /opt/cm (每台机手动执行)
5. 在master上启动CM的service服务  
   /opt/cm/etc/init.d/cloudera‐scm‐server start
6. 在所有节点上面节点上启动CM的agent服务  
   /opt/cm/etc/init.d/cloudera‐scm‐agent start  
   service服务过程需要启动几分钟，这看机器性能，如果用 netstat ‐apn|grep 7180 能够查看到端口占用，就可以用浏览器打开http://masterIP:7180/进行访问，登录用户名与密码分别为admin到此基本上CM的安装启动己完成，接下来就可以用过WEB完成CDH平台安装。

#### 4.1.2.8 安装Cloudera Distribute Hadoop

1. 使用admin(密码admin)登录



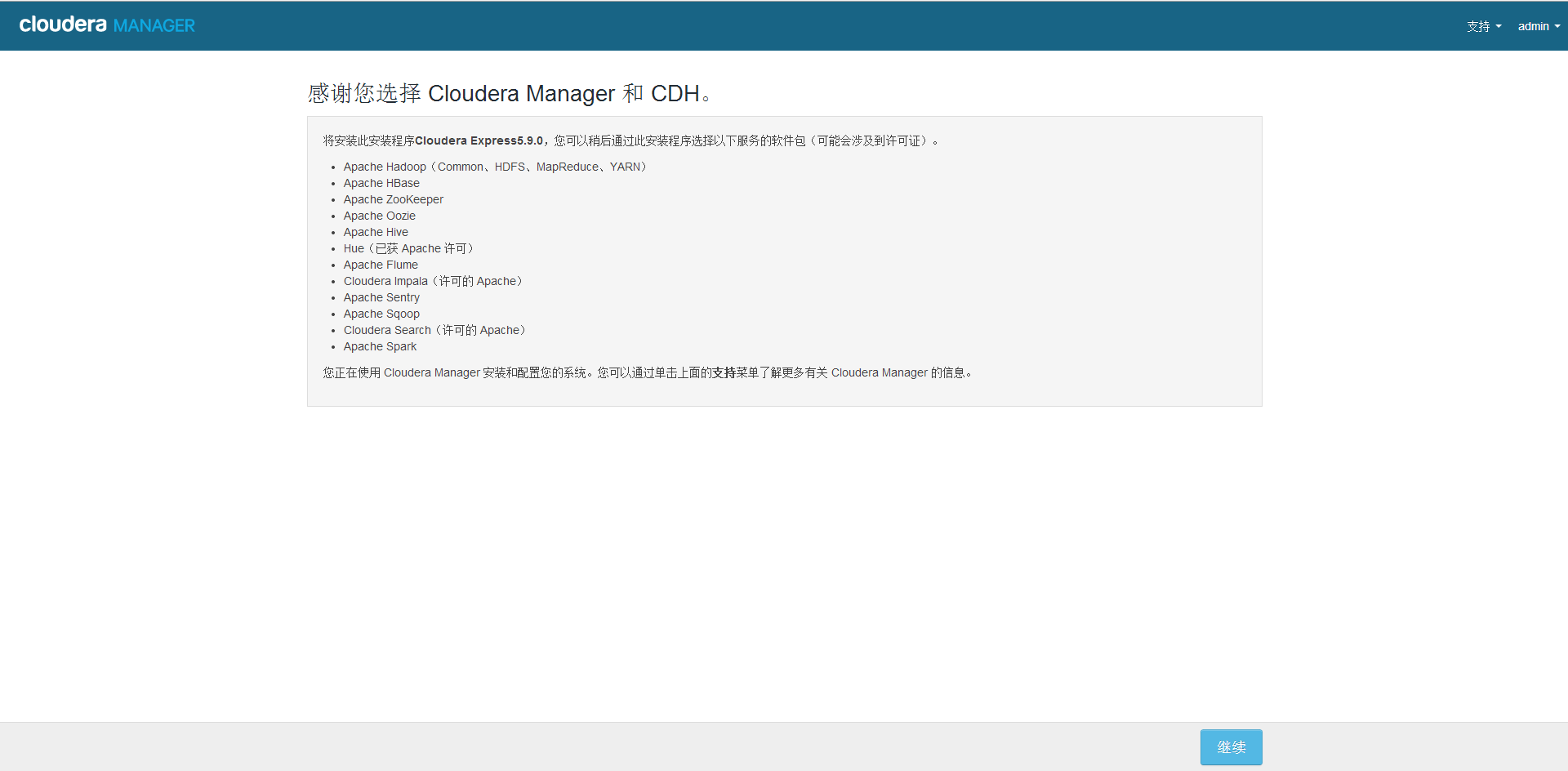
1. 选中“继续”



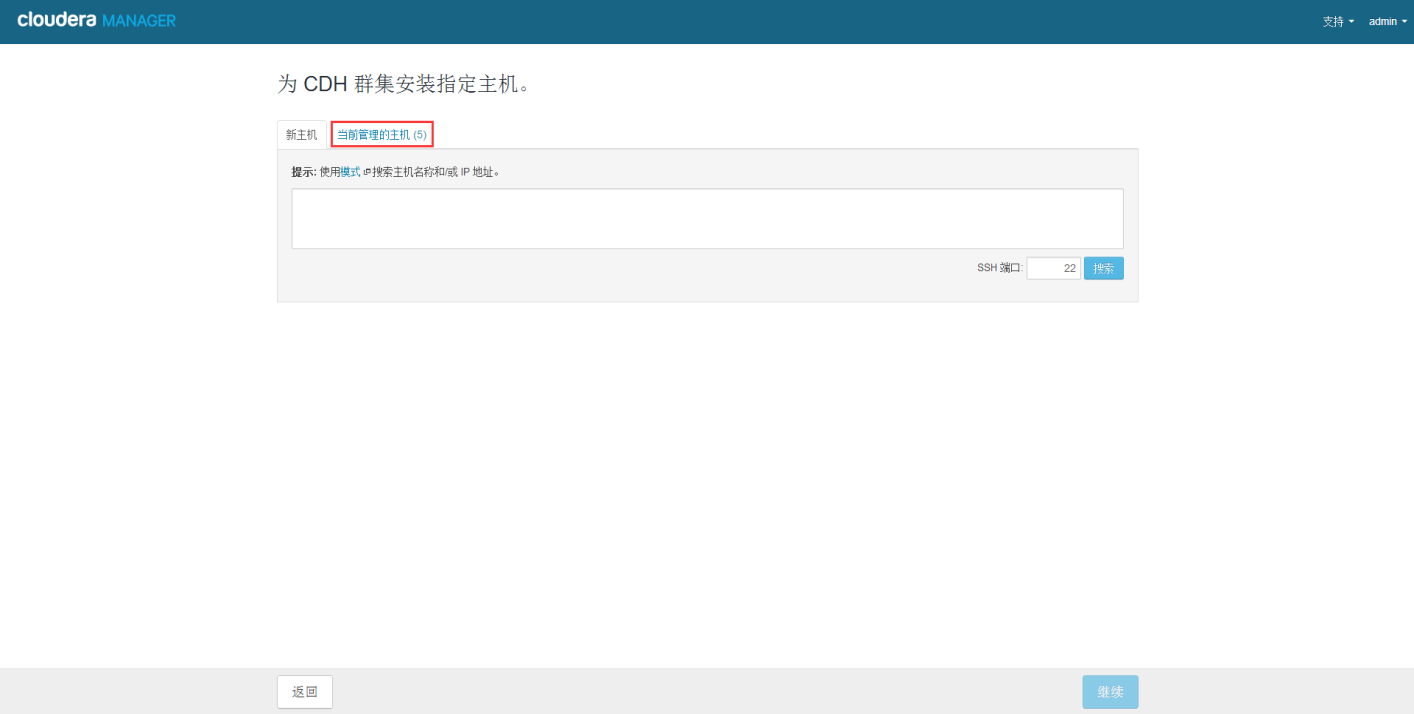
1. 按需选择，选择免费



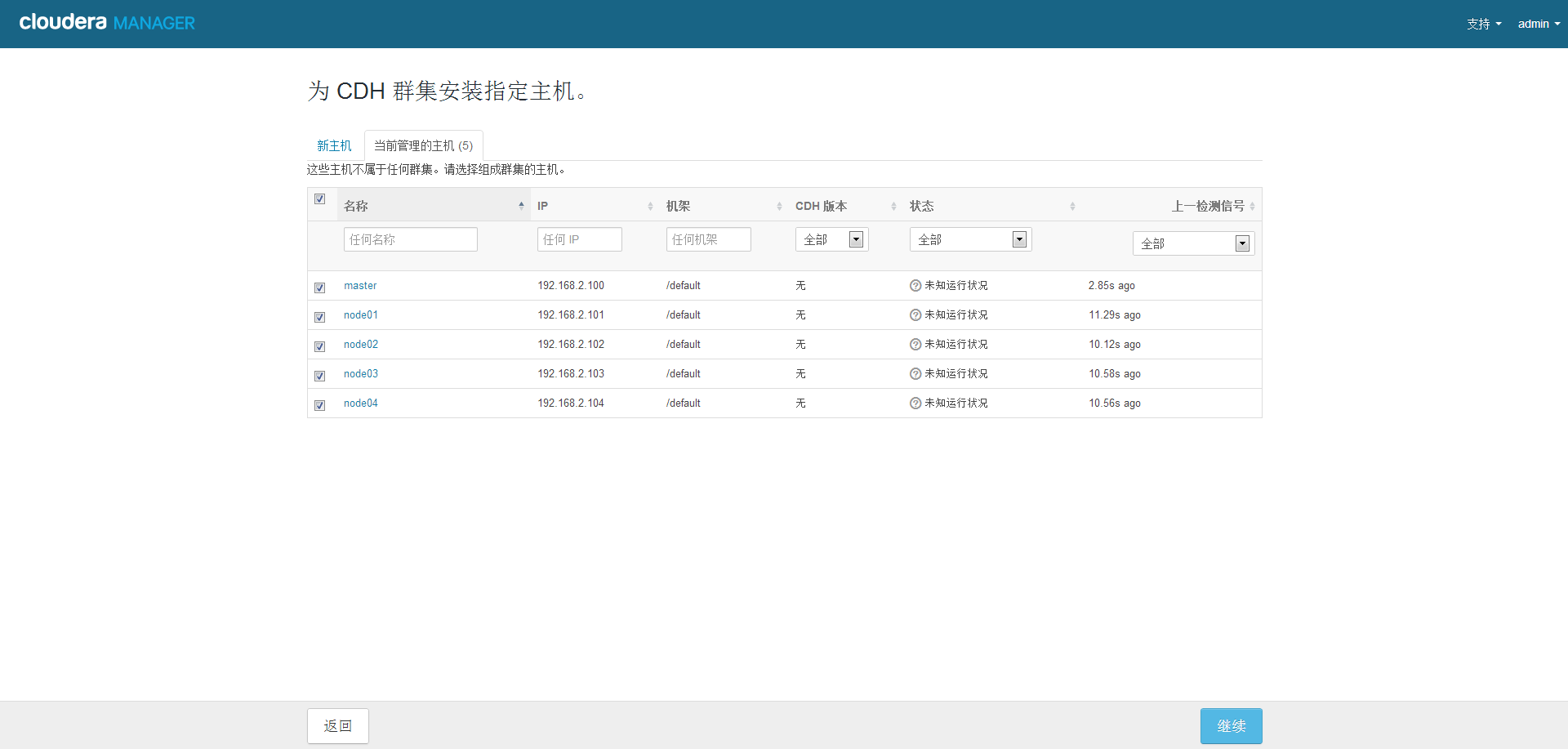
1. 继续



1. 节点上启动了agent，所以直接点“当前管理的主机”。如果节点上没有CM，只有master上有，这边可以在新主机下去搜索，例如172.16.18.[150-154]。

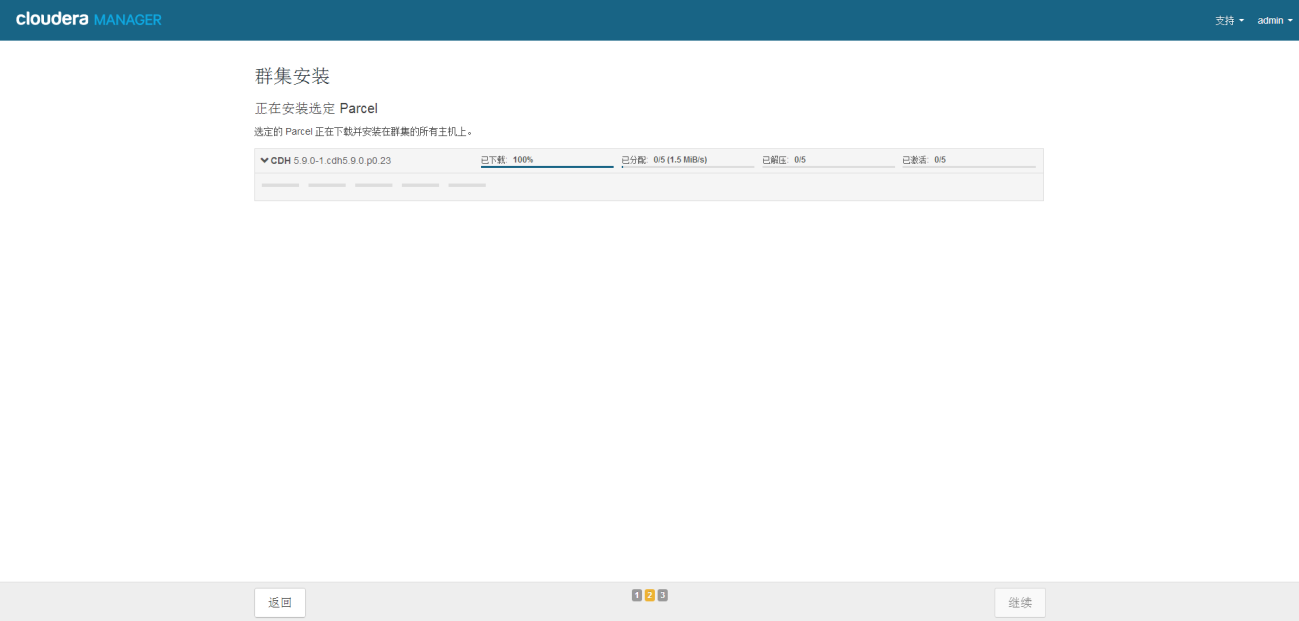


1. 全部勾上，然后继续

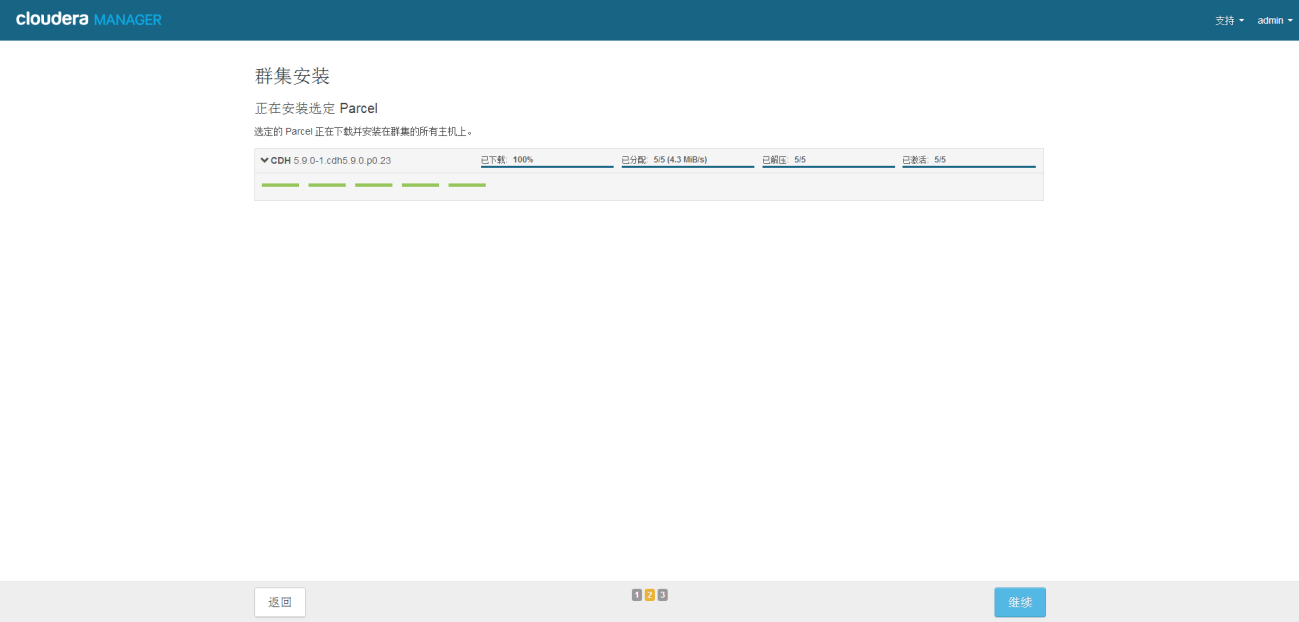


1. 选择版本，然后继续





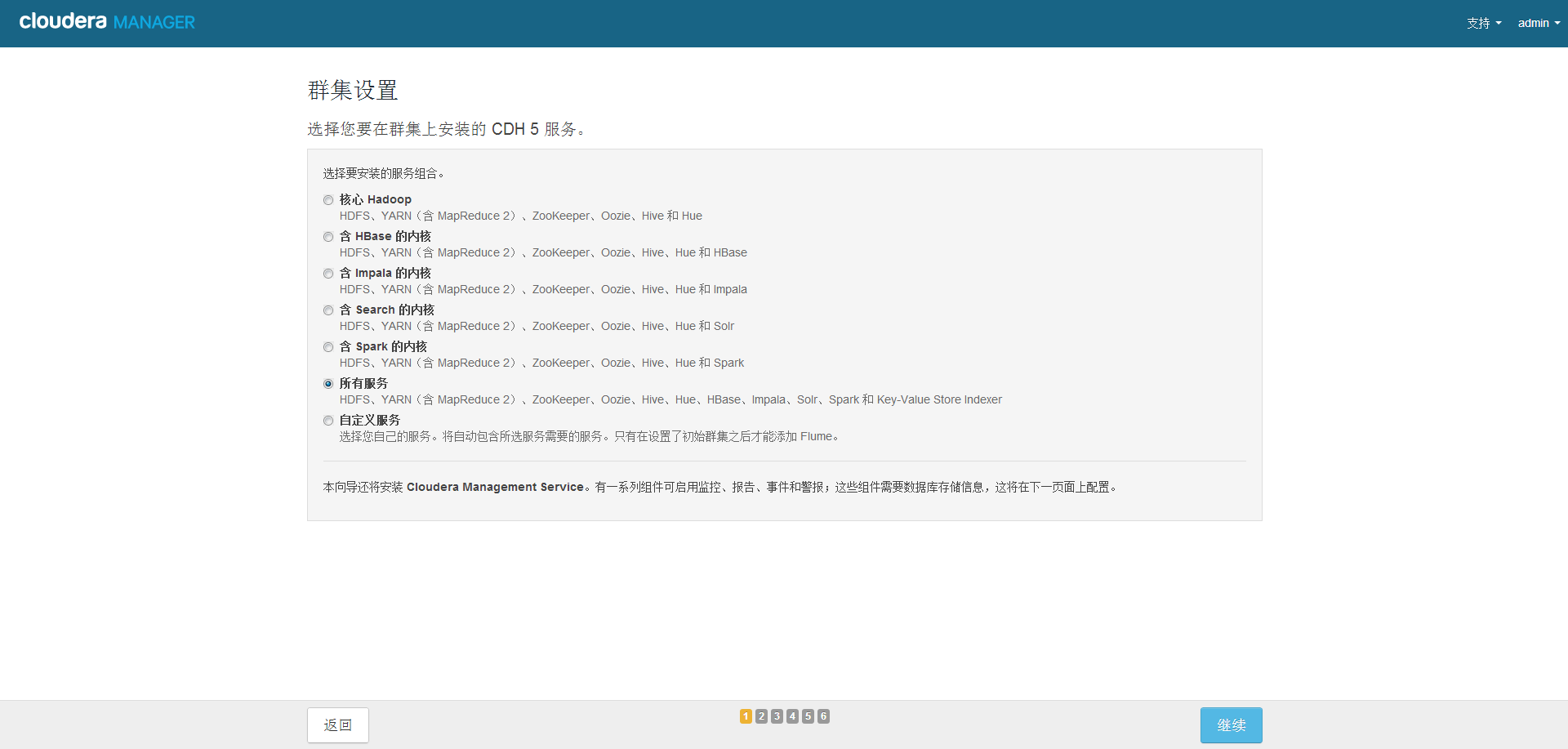
1. 集群分配完成后，继续



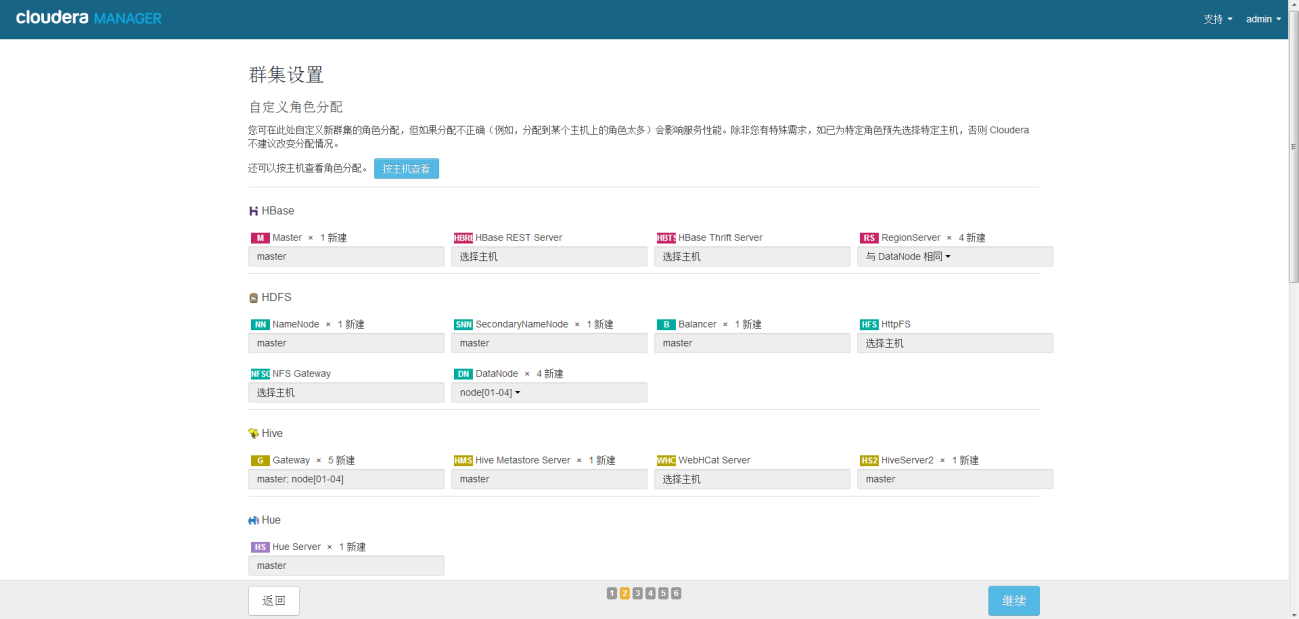
1. 显示检查列表，完成安装！



1. 集群服务分配，按需选择



1. 角色分配，按需分配



1. 创建Mysql数据库并测试(按需创建，比如你没选oozie，就不用创建oozie的数据库)

--hive数据库

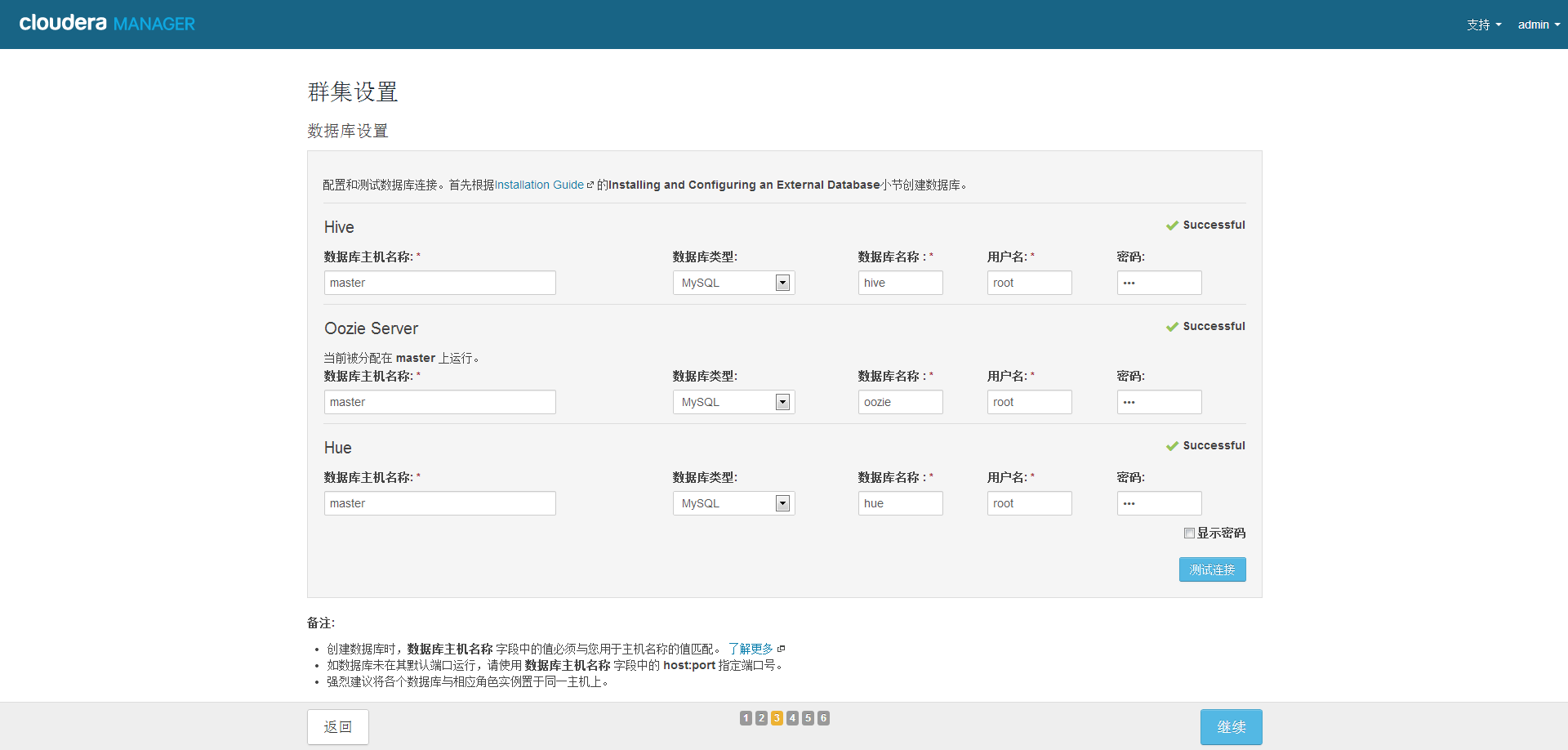
create database hive DEFAULT CHARSET utf8 COLLATE utf8\_general\_ci;

--oozie数据库

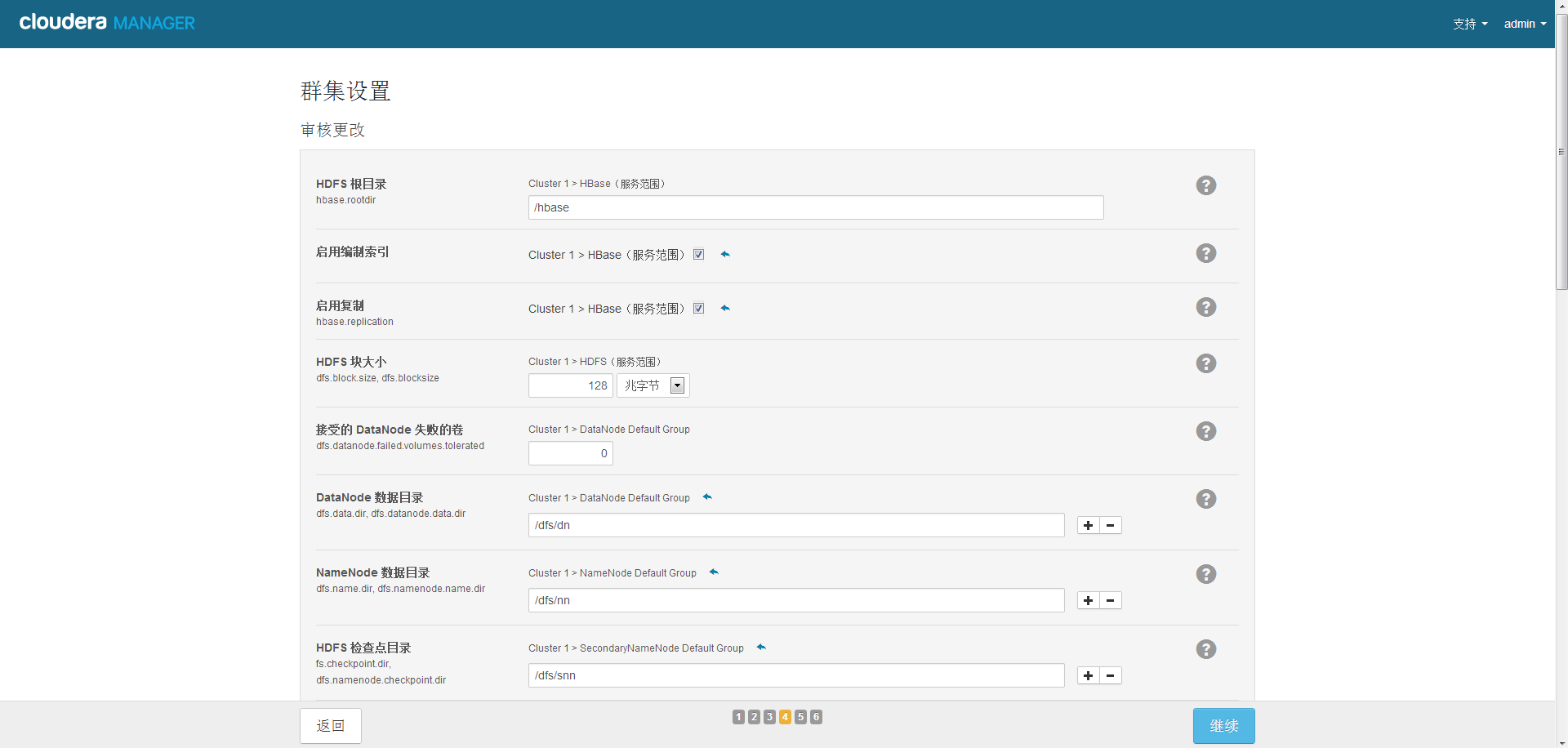
create database oozie DEFAULT CHARSET utf8 COLLATE utf8\_general\_ci;

--hue数据库

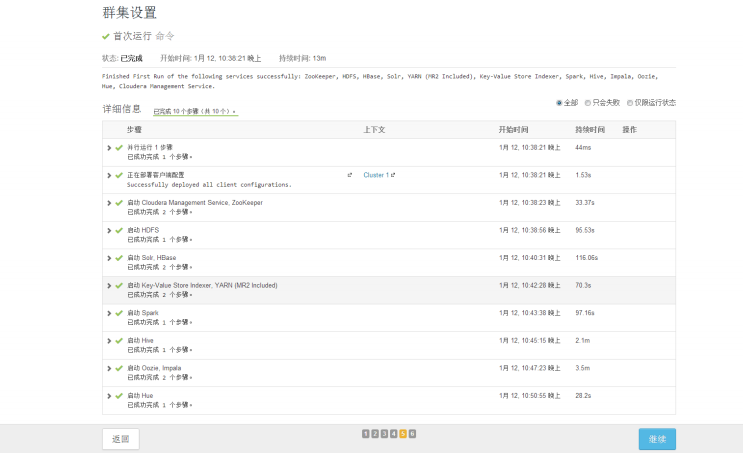
create database hue DEFAULT CHARSET utf8 COLLATE utf8\_general\_ci;



1. 集群设置

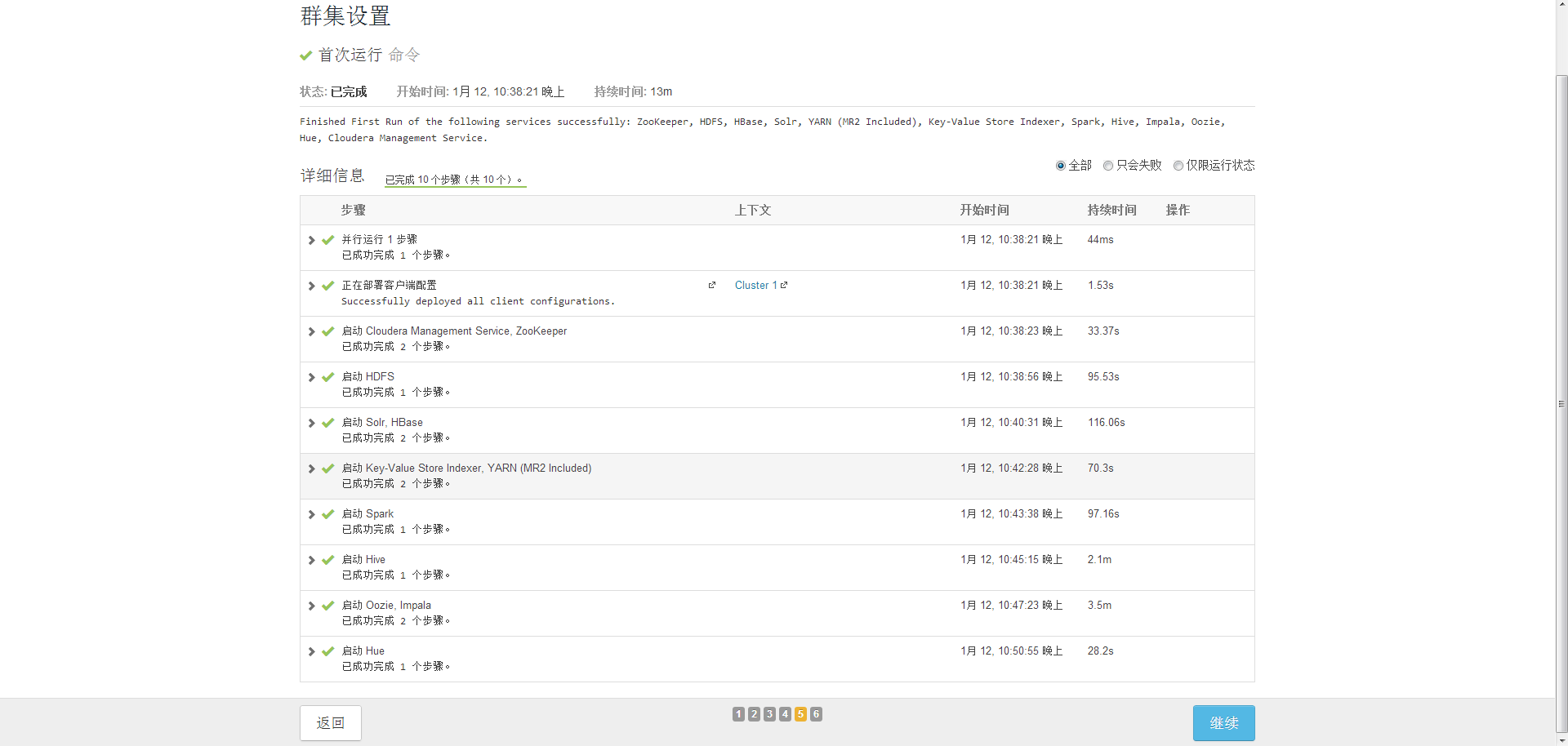


1. 启动集群

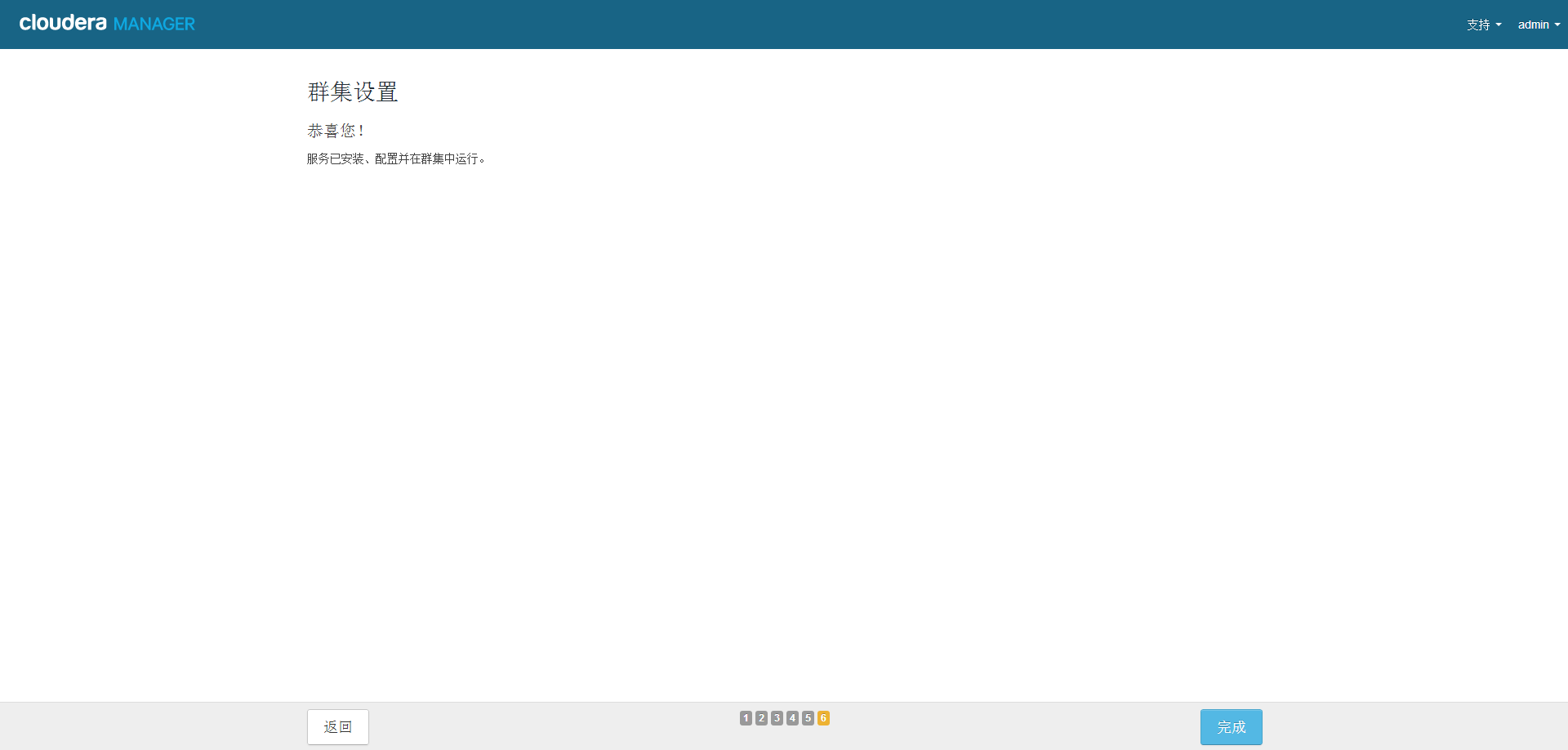


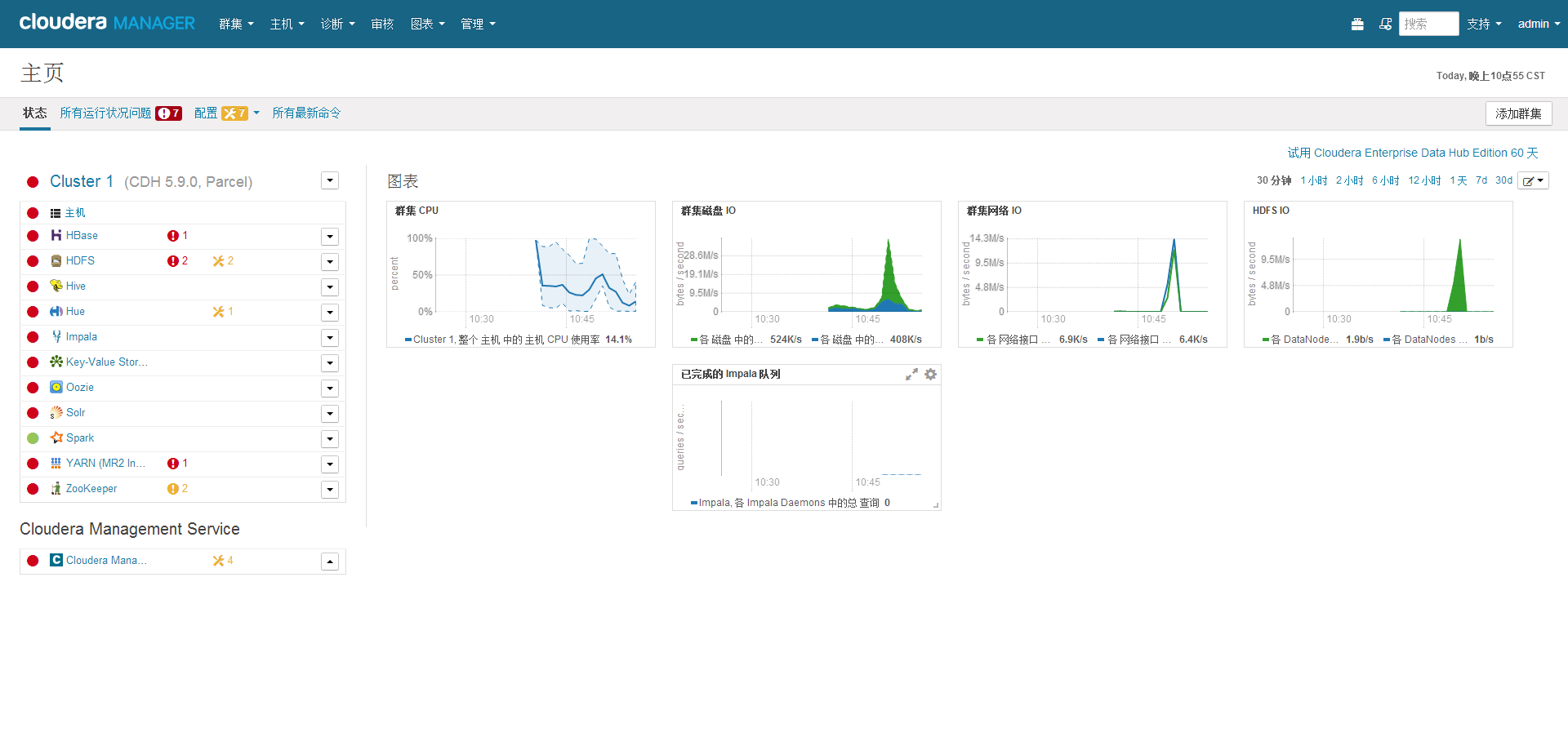
1. 大数据平台界面





20、终于装好了0.0





## 4.2 平台数据建模

### 4.2.1 数据建模工具与流程

数据采集模块主要功能是**连接**外部数据(应用程序日志文件、实时流数据或关系型数据库数据)或本地磁盘数据(本地文件系统或分布式文件系统)，并将这些各种数据源的各种格式的数据**按照一定的方式清洗、整理或预处理**后，**传输**给大数据实时计算模块或批处理模块分析处理，进一步通过可视化模块查看或展示处理结果。本文研究的高校数据主要为融合治理高校各信息化系统中存储到MySQL关系型数据库中的数据，因此使用开源Sqoop工具将数据导入到Hive分布式数据库中。

#### 4.2.2.1 import将MySQL表数据导入Hive

将MySQL数据库表的数据导入到hive中，如果在hive中没有对应的表，则自动生成与数据库表名相同的表。

sqoop import -connect jdbc:mysql://localhost:3306/hive -username root -password 123456 -table user -split-by id -hive-import

-split-by指定数据库表中的主键字段名，在这里为id。

|  |  |
| --- | --- |
| 参数 | 说明 |
| -append | 将数据追加到hdfs中已经存在的dataset中。使用该参数，sqoop将把数据先导入到一个临时目录中，然后重新给文件命名到一个正式的目录中，以避免和该目录中已存在的文件重名。 |
| -as-avrodatafile | 将数据导入到一个Avro数据文件中 |
| -as-sequencefile | 将数据导入到一个sequence文件中 |
| -as-textfile | 将数据导入到一个普通文本文件中，生成该文本文件后，可以在hive中通过sql语句查询出结果。 |
| -boundary-query <statement> | 边界查询，也就是在导入前先通过SQL查询得到一个结果集，然后导入的数据就是该结果集内的数据，格式如：–boundary-query ‘select id,creationdate from person where id = 3’，表示导入的数据为id=3的记录，或者select min(<split-by>), max(<split-by>) from <table name>，注意查询的字段中不能有数据类型为字符串的字段，否则会报错：java.sql.SQLException: Invalid value for getLong() |
| -columns<col,col,col…> | 指定要导入的字段值，格式如：–columns id,username |
| -direct | 直接导入模式，使用的是关系数据库自带的导入导出工具。官网上是说这样导入会更快 |
| -direct-split-size | 在使用上面direct直接导入的基础上，对导入的流按字节数分块，特别是使用直连模式从PostgreSQL导入数据的时候，可以将一个到达设定大小的文件分为几个独立的文件。 |
| -inline-lob-limit | 设定大对象数据类型的最大值 |
| -m,-num-mappers | 启动N个map来并行导入数据，默认是4个，最好不要将数字设置为高于集群的节点数 |
| -query，-e<statement> | 从查询结果中导入数据，该参数使用时必须指定–target-dir、–hive-table，在查询语句中一定要有where条件且在where条件中需要包含$CONDITIONS，示例：–query ‘select \* from person where $CONDITIONS ‘ –target-dir /user/hive/warehouse/person –hive-table person |
| -split-by<column-name> | 表的列名，用来切分工作单元，一般后面跟主键ID |
| -table <table-name> | 关系数据库表名，数据从该表中获取 |
| -target-dir <dir> | 指定hdfs路径 |
| -warehouse-dir <dir> | 与–target-dir不能同时使用，指定数据导入的存放目录，适用于hdfs导入，不适合导入hive目录 |
| -where | 从关系数据库导入数据时的查询条件，示例：–where ‘id = 2′ |
| -z,-compress | 压缩参数，默认情况下数据是没被压缩的，通过该参数可以使用gzip压缩算法对数据进行压缩，适用于SequenceFile, text文本文件, 和Avro文件 |
| -compression-codec | Hadoop压缩编码，默认是gzip |
| -null-string <null-string> | 可选参数，如果没有指定，则字符串null将被使用 |
| -null-non-string<null-string> | 可选参数，如果没有指定，则字符串null将被使用 |

增量导入

|  |  |
| --- | --- |
| 参数 | 说明 |
| -check-column (col) | 用来作为判断的列名，如id |
| -incremental (mode) | append：追加，比如对大于last-value指定的值之后的记录进行追加导入。lastmodified：最后的修改时间，追加last-value指定的日期之后的记录 |
| -last-value (value) | 指定自从上次导入后列的最大值（大于该指定的值），也可以自己设定某一值 |

对incremental参数，如果是以日期作为追加导入的依据，则使用lastmodified，否则就使用append值。

### 4.2.2 高校数据采集

高校大数据采集主要工作为按照数据清洗规则将高校各种信息化系统中的数据提取、转换、加载到大数据治理融合平台的数据仓库中。

根据3.2节需求，本文主要采集如下类别数据：

|  |  |  |  |
| --- | --- | --- | --- |
| 数据类别 | 数据用途 | 数据源 | 数据表 |
| 学生数据 |  |  |  |
| 宿舍数据 |  |  |  |
| 一卡通数据 |  |  |  |
| 上网数据 |  |  |  |
| 考核数据 |  |  |  |
| 课程数据 |  |  |  |
| 机构数据 |  |  |  |
| 教师数据 |  |  |  |
| 图书数据 | 分析学生借阅记录；图书采购指导、图书推荐、统计分析等 | 图书馆管理系统 |  |
| 医疗数据 | 学生健康监控、流感预测、医药用品采购 | 医院管理系统 |  |
| 就业数据 | 就业统计、聚类分析、辅助识别优秀毕业生、辅助图书推荐及学生行为分析 | 就业管理系统 |  |

### 4.2.3 数据清洗

数据清洗的主要目的

|  |  |  |
| --- | --- | --- |
| 名称 | 描述 | 不规范数据 |
| 确定性 | 数据具有一个一致统一的业务定义和计算规则 | 在高校内部，不同信息化系统使用不同的算法和源数据计算投入、收益及资源使用 |
| 有效性 | 数据与它的定义(数据类型、格式、内容)相符合 | 学生记录包含数字或乱码，无法使用 |
| 唯一性 | 数据时独一无二的—没有冗余 | 两个学生具有相同的学号 |
| 完整性 | 数据时不可缺的或与另一个数据关联的 | 学生记录缺少学号  婚否(Y/N)填写的是“空”值 |
| 一致性 | 数据元素不会因为有了另一个数据而产生变化或自相矛盾 | 学生的毕业日期比入学日期早 |
| 及时性 | 数据代表谋业务活动的产出物的最当前信息 | 学生家庭联系地址和方式是旧的，联系电话是空号，没有及时更新 |
| 准确性 | 恰当地指派了数据值 | 学生年级分班具有一个不准确或无效的值 |
| 精确性 | 数据只能按它被界定的定义去使用，即数据特征被理解和准确地应用的程度 | 实验室设备被用于不同记录的不同产品类型 |

高校数据清洗举例：

* **清洗规则**：门禁系统，把没有卡ID的数据过滤，并把每个卡ID与用户ID做关联，记录到门禁记录中
* **一卡通系统**：管理学生ID，增加学生姓名与学号
* **教务系统**：过滤成绩为0的记录，过滤重复的课程，通过门禁系统、增加宿舍位置等全方位完整信息
* **学生信息**：过滤无效的地址信息，通过身份证号码，适配学生身份、城市信息等

### 4.2.4 数据建模

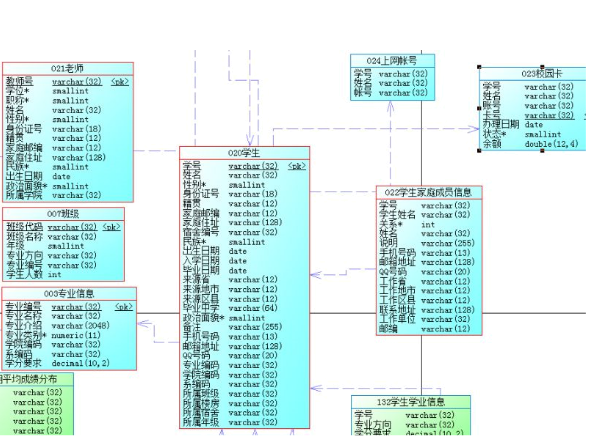
[//5.1日补充高校数据表]

按照第三章的需求分析，作者调研了 教育信息标准化表格、学校调研、论文查阅，整理了XXX类XXX张表，如下清单所示：

类别|表名|表描述|表字段|

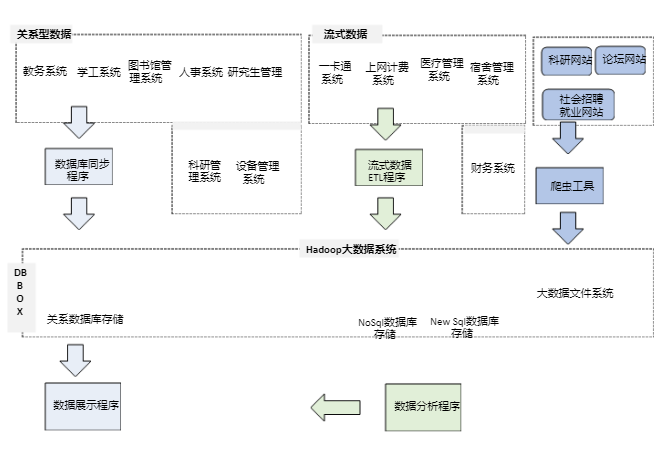
建表语句见附件

表的关联关系，E-R逻辑模型如下图：



### 4.2.5数据导入Hive

[//5.1日完成数据导入实例]



//【删除该图】

## 4.3 高校大数据应用场景实现

### 4.3.1 图书馆大数据分析

大数据存储包含多种数据存储方式，具体为分布式文件系统（DFS）、非关系型数据库（NOSQL）及关系型数据库（RDBMS），满足数据连接、预处理、加工与呈现不同场景下业务能力的需求。

#### 4.3.1.1 设计思路

数据存储已经有成熟的解决方案，主要使用开源分布式文件系统HDFS及基于分布式文件系统的NoSQL数据库HBase等。数据存储主要工作体现在设备组网、HDFS及HBase配置优化等内容，相对其它模块而言，工作量较小。

#### 4.3.1.2

### 4.3.1.5

#### 4.3.1.6 高校大数据治理融合平台存储方案

根据3.1章节介绍的高校大数据架构设计，高校大数据治理融合平台存储层由Hive、HDFS、Hbase、MySQL、MongoDB等分布式文件系统或数据仓库构成。数据存储层的访问接口与业务数据关系，如下图所示：



图‑2 数据存储及业务关系图

### 4.3.2 数据处理

根据数据计算的实时性：分为实时数据计算和离线批处理：

* **实时数据计算** 往往要求系统在分钟或数秒内返回上亿行数据的分析，从而才能达到不影响用户体验的目的。
* **离线批处理** 对大多数反馈时间要求不高的应用，比如离线统计分析、机器学习等，应采用离线分析的方式，通过数据采集工具将日志数据导入专门的分析平台进行分析。

系统主要以离线数据分析为主，采用目前在互联网业界流行的Hadoop体系结构及Spark技术栈对大批量的数据进行运算，采用集群的方式对大数据进行运算。

数据运算平台以调度为主线，作为运算平台的核心控制系统，对运算平台的各个环节进行控制，且对运算过程中的步骤依赖关系进行控制，同时对各个环节进行监控，通过监控异常报警来提高系统的稳定性和异常响应速度。数据计算模块如下图所示：

图4‑‑3 数据处理流程

#### 4.3.2.1 实时数据计算

大数据的应用场景：数据源是实时的不间断的，数据量大且无法或没有必要预算，但要求响应时间是实时的。为了解决这类场景，出现了大数据实时数据计算技术，常见的实时数据计算技术有Storm、SparkStreaming、Druid、Pinot，Kylin，其对比如下表所示：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **对比项** | **Storm** | **SparkSteaming** | **Druid** | **Pinot** | **Kylin** |
| 技术 | 流式计算 | 切片处理，内存计算 | lamda架构、自研引擎 | 类Druid架构  存内存计算 | MapReduce预计算 |
| 性能 | 性能高，亚秒 | 性能较高，秒 | 毫秒及亚毫秒级速度， | 毫秒及亚毫秒级速度， | 性能一般：分钟级别 |
| 成熟度 | 起步早，技术成熟 | 起步晚，技术较成熟 | 技术成熟 | 初步阶段 | 初级研究阶段 |
| 支持语言 | JVM语言/Python/RubyPerl | Java/Scala/Python | java | Java | Java |
| 市场普及率 | 生产环境占有率，高 | 较高 | 较高 | 一般 | 无 |
| 使用难度 | 一般 | 一般 | 较难 | 较难 | 无参考 |
| 文档及社区 | 文档丰富、社区活跃 | 文档丰富、社区活跃 | 文档较少 | 文档少，活跃 | 几乎没有 |
| 代码活跃度 | 一般 | 高 | 高 | 高 | 高 |

本设计方案支持基于Spark上的Spark Streaming，满足秒级或毫秒级的实时计算场景需求，如实时推荐、用户行为分析等。**考虑支持**Storm流式任务作业引擎，覆盖实时要求极高的流式作业场景。实时数据计算流程图，如下图所示：

图 13 实时计算流程图

#### 4.3.2.2 离线批量处理

离线批处理计算已经成熟，业界常见的方案有两种：MapReduce任务及Spark。本设计方案根据技术栈一致性及性能指标出发，主要以Spark作为离线批处理的主要选择，当数据量超过内存配置时或时效性要求不高的非重要任务辅助使用MapReduce任务处理。

图 14 离线批处理流程图

## 4.4 可视化系统

### 4.4.1 可视化系统架构



**Web层**：主要用于展示数据，显示、增加、修改、删除数据，websocket用于实时更新页面数据。管理用户的请求和响应提供一个控制起来将将调用委托业务逻辑和其他上游处理组装可以在视图中表现的模型对象执行UI校验实现拖拽。

**服务层**，实现各个业务功能，采用微服务架构。通过分解巨大单体式应用为多个服务方法解决复杂性问题，在功能不变的情况下，应用被分解为多个可管理的分支或服务。可以把每个服务独立部署。避免服务之间的影响。

**DAO层**，采用JPA提供更加简单的的编程模型，定义了Java ORM及实体操作API的标准。除了提供面向对象的ORM框架外，还提供原始的JDBC操作数据的方式，使平台更加灵活。

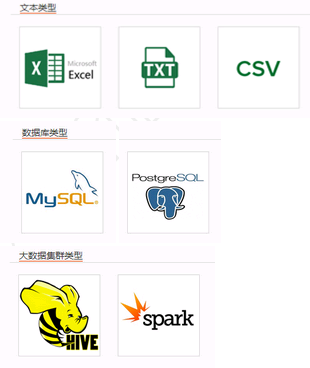
### 4.4.2 数据源管理

高校大数据治理融合平台的数据源支持对多种常见数据库的连接，包括MySql、PostgreSQL、Excel等数据源。数据源依附应用而存在，若依附的应用被删除，该应用中包含的数据源也会被删除。

* 文本类型，支持上传CVS/Excel文件。只支持已经格式化的文件。并且大小限制500MB。
* 数据库类型，MySQL、PostgreSQL。
* 大数据集群类型，Hive、Spark等。



图表 ‑4 数据源对接流程图



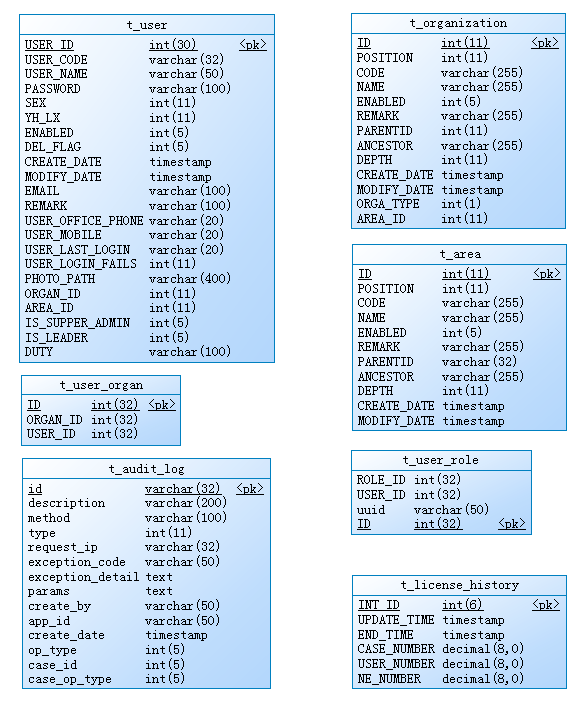
图表 ‑5 数据源对接界面

### 4.4.3

### 4.4.4 业务前端展示图表

业务前端利用echarts部分可视化图表，包括柱状图、饼图、折线图、热力图、地图、雷达图、仪表盘、预警信号、网络图等。

### 4.4.5 可视化系统数据库设计



#### 4.4.5.1 数据源表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | cbi\_datasource |  |  |
| 字段ID | 字段名称 | 字段类型 | 备注 |
| Id |  | Bigint | 主键 |
| config | 连接参数 | Text | json |
| db\_type | 数据库类型 | Int |  |
| Is\_copy | 是否复制 | Int |  |
| brief | 概要 | text |  |
| is\_success | 是否连接成功 | Int | 0:成功；1：失败 |

#### 4.4.5.2 应用表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | cbi\_object |  |  |
| 字段ID | 字段名称 | 字段类型 | 备注 |
| Id |  | Bigint | 主键 |
| Code | 编码 | Text | json |
| Data\_attr\_type | 数据类型 | Int |  |
| Create\_time | 创建时间 | Datetime |  |
| Modified\_time | 修改时间 | Datetime |  |
| Is\_del | 是否删除 | Int | 0:成功；1：失败 |
| Levels | 等级 | Int |  |
| Name | 名称 | Longtext |  |
| Order | 排序 | Int |  |
| Owner | 拥有者 | bigInt |  |
| Parent\_id | 父级ID | Bigint |  |
| Project\_id |  | Bigint |  |
| Status | 状态 | Int |  |
| Sub\_type |  | Int |  |
| type |  | int |  |

#### 4.4.5.3 元数据存储表

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | cbi\_colmun |  |  |
| 字段ID | 字段名称 | 字段类型 | 备注 |
| Id |  |  | 主键 |
| Code |  | text |  |
| Data\_type | 数据类型 | Int | 1：维度；2：度量 |
| Expr |  | Text |  |
| Foreign\_key\_tableid | 外键表ID | Bigint |  |
| Is\_del | 是否删除 | Int |  |
| Is\_formula | 是否公式 | Int |  |
| Is\_partition | 是否分割 | Int |  |
| Is\_primary | 是否主键 | Int |  |
| Mapping\_type | 映射类型 | Int |  |
| Name | 名称 | Text |  |
| Object\_name | 对象名称 | Text |  |
| Order | 排序 | Int |  |
| Talbe\_id | 表ID | Bigint |  |
| label |  | Text |  |
| Original\_name | 原始名称 | Text |  |

#### 4.4.5.4 仪表盘表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | cbi\_board |  |  |
| 字段ID | 字段名称 | 字段类型 | 备注 |
| Id |  |  | 主键 |
| Auto\_update | 是否自动更新 | int |  |
| Config | 配置信息 | Text | Json |
| Descp | 描述信息 | Text |  |
| Is\_booked | 是否 |  |  |
| Is\_public | 是否公开 |  |  |
| Need\_report | 是否需要统计 |  |  |

#### 4.4.5.5 图表表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | cbi\_chart |  |  |
| 字段ID | 字段名称 | 字段类型 | 备注 |
| Id |  |  | 主键 |
| Board\_id | 仪表盘ID | int |  |
| C\_type | 数据类型 | Int |  |
| Config | 配置信息 | Text | Json |
| Is\_update | 是否更新 | Int |  |
| Last\_update\_time | 最后更新时间 | Datetime |  |
| Update\_interval | 更新间隔 | Bigint |  |

#### 4.4.6 可视化分析实例

## 4.6 本章小结

本章详细介绍了高校大数据治理融合平台的设计方案与实现方法。使用三台Dell服务器及4台虚拟机搭建了高校大数据治理融合平台试验环境。利用CDH大数据管理系统安装、部署了大数据集群服务，如Hive、Yarn、HDFS、Spark、Kafka、Zookeeper、Flume、Habse等。分析了数据建模相关知识点，即建模使用的工具、流程、规范、规则等，并根据高校大数据的数据内容设计了数据模型。详细展示了数据存储方案及可视化系统设计方案与实现方法。

# 第五章 高校大数据治理融合平台实现

## 4.1 高校大数据治理融合基础平台

高校大数据融合治理平台至少包含监控、调度、管理、数据采集、数据处理等功能模块，如下图所示：



‑1 高校大数据融合治理平台

为了简化平台的开发及设计工作，系统监管、系统管理及调度系统功能模块由Cloudera Distubtion Hadoop(CDH5.9)开源大数据平台提供支撑，数据采集工作由Sqoop、Flume、Kettle等开源软件实现，数据处理功能CDH5.9平台中集成的impala、Hive、Hbase、Spark、Kafka等开源大数据组件实现。设计完成高校大数据融合治理平台的主要工作包括：

1. 申请服务器，完成平台设备组网；安装部署CDH5.9平台、并熟练掌握使用平台集成的各个大数据组件
2. 梳理、理解高校各信息化系统中的数据，并使用Sqoop、Flume、Kettle等大数据组件将数据统一融合、导入到大数据平台
3. 利用大数据分析技术挖掘高校大数据中存储的数据价值

### 4.1.1 平台物理架构

#### 4.1.1.1 物理组网



#### 4.1.1.2 主机配置列表

高校大数据实验环境中，主机配置列表如下：

|  |  |  |  |
| --- | --- | --- | --- |
| Ip | 类型 | 配置 | 作用 |
| 172.16.18.41 | 物理机 | DELL R730 8核 96G  8\*300G /Redhat 6.4 | CDH管理软件、大数据组件安装包、YUM源、MySQL、MongoDB、Kylin |
| 172.16.18.51 | 物理机 | DELL R730 8核 96G  8\*300G /Redhat 6.4 | Spark集群、HDFS集群、Kafka集群 |
| 172.16.18.52 | 物理机 | DELL R730 8核 96G  8\*300G /Redhat 6.4 | Spark集群、HDFS集群、Kafka集群 |
| 172.16.18.150-4 | 虚拟机 | 2核 32G 600G /Redhat 6.4 | Hive集群、Zookeeper集群、Flume |

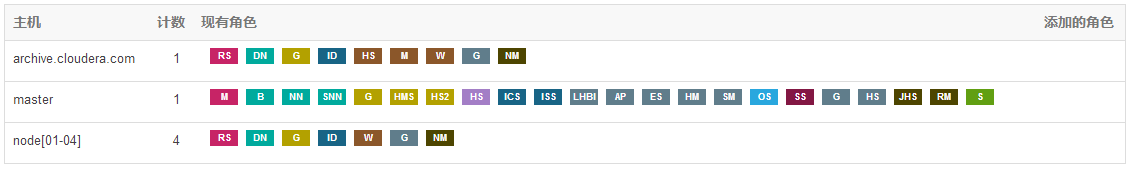
#### 4.1.1.3 Host主机名配置

所有主机中的/etc/hosts文件内容如下：

|  |
| --- |
| 172.16.18.51 master  172.16.18.52 archive.cloudera.com  172.16.18.150 node01  172.16.18.151 node02  172.16.18.152 node03  172.16.18.153 node04  172.16.18.41 cm #Cloudera Manager |

#### 4.1.1.4 大数据服务配置

CDH中查看集群服务器中大数据服务配置列表：



服务说明：

|  |  |
| --- | --- |
| 服务图标 | 服务说明 |
|  | HBase RegionServer |
|  | HBase Master |
|  | HDFS DataNode |
|  | HDFS Balancer |
|  | HDFS NameNode |
|  | HDFS SeconderyNameNode |
|  | Hive Gateway |
|  | Hive Metastore Server |
|  | Hive Server2 |
|  | Impala StateStore |
|  | Impala Deamon |
|  | Impala CatalogServer |
|  | Spark HistoryServer |
|  | Spark Worker |
|  | Spark Gateway |
|  | Spark HistoryServer |
|  | Yarn NodeManager |
|  | Yarn JobHistoryServer |
|  | Yarn ResourceManager |
|  | Hue Server |
|  | Oozie Server |
|  | Solr Server |
|  | Zookeeper Server |

### 4.1.2 平台管理安装

大数据应用技术平台选择使用CDH发行版5.9.0作为基础构建管理平台；安装步骤参考CDH官网[4.1.2官网]及国内网络资料[4.1.2安装参考]

#### 4.1.2.1 安装环境准备[4.1.2安装参考]

安装参考文档中步骤，首先准备安装环境，主要包括安装服务器操作系统、虚拟机创建、域名及IP网络设置、防火墙设置。这一步主要保证服务器之间能够互联互通、服务器能够连接互联网(CDH官网)。从官网下载离线安装包：

Cloudera Manager 5.9 http://archive­primary.cloudera.com/cm5/cm/5/cloudera­manager­el6­cm5.9.0\_x86\_64.tar.gz

CDH5.9 主文件http://archive­primary.cloudera.com/cdh5/parcels/5.9.0.23/CDH­5.9.0­1.cdh5.9.0.p0.23­el6.parcel

CDH5.9 sha文件http://archive­primary.cloudera.com/cdh5/parcels/5.9.0.23/CDH­5.9.0­1.cdh5.9.0.p0.23­el6.parcel.sha1

manifest 文件http://archive­primary.cloudera.com/cdh5/parcels/5.9.0.23/manifest.json

下载后将这些包上传到master机器的/opt/cdh目录下

#### 4.1.2.2 NTP配置

大数据平台的大数据分布式大数据组件要求集群服务器时间几乎同步。NTP时间同步，不一定分秒不差，但要保持在一个合理的范围之内，不然服务运行会报错。时间同步有两种方式，一种是7台均同步单独的时钟服务器，一种是使用master本身作为时钟服务器，其它节点同步它，接下来采用第二种方式。

1. 修改配置文件

打开master机器 /etc/ntp.conf 文件，其它地方可以注释，留下如下关键信息，保存退出。

|  |
| --- |
| driftfile /var/lib/ntp/drift  restrict 127.0.0.1  restrict ‐6 ::1  restrict default nomodify notrap  server 127.127.1.0  fudge 127.127.1.0 stratum 8  includefile /etc/ntp/crypto/pw  keys /etc/ntp/keys |

1. 启动服务

执行如下命令启动NTP服务，如果服务之前己启动，则重启服务。ntp与自身进行同步需要一小段时间，大约需要5分钟。

|  |
| --- |
| service ntpd start  chkconfig ntpd on |

使用命令 ntpstat 检查服务启动后自身是否同步完成，如出现以下信息则完成。

|  |
| --- |
| synchronised to local net at stratum 9  time correct to within 12 ms  polling server every 64 s |

1. 客户端同步

服务己启动，完成其它节点机器同步了，使用命令 ntpdate 172.16.18.150 查看结果，如出现以下信息则同步完成。

|  |
| --- |
| 24 Feb 13:37:35 ntpdate[13675]: adjust time server 172.16.18.150 offset 0.068598 sec |

#### 4.1.2.3 实用shell

后续安装过程，考虑到各个机器间需要拷贝文件，如果到各个机器去执行拷贝命令，需要分别到6台集群机器执行，为了简化操作，使用用脚本的方式批处理。

在master机器root目录下，创建一个nodes 文件，该文件存放集群中所有集群的域名，文件内容如下：

|  |
| --- |
| master  archive.cloudera.com  node01  node02  node03  node04  cm |

在master机器root目录下，编辑一个 scp.sh 文件，文件内容如下，输入需要传送的源文件路径与目标机器的存放路径两个参数，路径都是绝对路径。

|  |
| --- |
| #!/bin/bash cat nodes | while read host do { scp ‐r $1 $host:$2 }&wait done |

#### 4.1.2.4 JAVA环境安装

大数据平台的大部分开源组件使用Java语言编写，需要安装较高版本的java JDK。新安装的Redhat6.0系统自带的OpenJDK版本较低，使用 rpm -qa|grep java查看。如果有，则使用命令 yum -y remove xxxx删除，下载JDK-1.8.0\_112，并将下载好的JDK包解压放至所有服务器的/opt/java 目录下，调置环境变量，编辑文件 /etc/profile 添加如下内容

|  |
| --- |
| export JAVA\_HOME=/opt/java  export PATH=$JAVA\_HOME/bin:$PATH |

拷贝JAVA文件与profile文件至其它节点，注意一点，其它节点不需要己经存在 java 目录，此脚本会自动创建其目录

|  |
| --- |
| ./scp.sh /opt/java/ /opt/java/  ./scp.sh /etc/profile /etc |

设置全局变量并生效，在五台机器上均执行如下命令

|  |
| --- |
| source /etc/profile  echo "JAVA\_HOME=/opt/java" >> /etc/environment |

检查JAVA是否安装成功，执行命令 java ‐version 出现如下结果

|  |
| --- |
| java version "1.8.0\_112"  Java(TM) SE Runtime Environment (build 1.8.0\_112‐b15)  Java HotSpot(TM) 64‐Bit Server VM (build 25.112‐b15, mixed mode) |

#### 4.1.2.5 Mysql安装

使用MySQL作为CDH管理平台的Cloudera Manager的元数据存储，只需要在master机器上安装。使用5.6.24 版本，需要下载如下安装包，然后，依次采用命令 rpm -ivh xxxxx 进行安装，如果中间出现冲突，就先删除再装：

|  |
| --- |
| mysql‐libs‐5.1.66‐2.el6\_3.x86\_64.rpm  MySQL‐shared‐5.6.24‐1.el6.x86\_64.rpm  MySQL‐devel‐5.6.24‐1.el6.x86\_64.rpm  MySQL‐server‐5.6.24‐1.el6.x86\_64.rpm  MySQL‐client‐5.6.24‐1.el6.x86\_64.rpm |

1. 启动MYSQL服务

|  |
| --- |
| chkconfig mysql on  service mysql start |

1. 修改初始密码

MYSQL启动后，默认其实己经生成了默认密码，密码保存在 /root/.mysql\_secret 文件中，为了安全需要改默认密码，示例修改成 123456，执行如下命令。

|  |
| --- |
| # service mysql stop  # mysqld\_safe ‐‐user=mysql ‐‐skip‐grant‐tables ‐‐skip‐networking&  # mysql ‐u root mysql  mysql> use mysql;  mysql> UPDATE user SET Password=PASSWORD('123456')where USER='root';  mysql> FLUSH PRIVILEGES;  mysql> quit  # service mysql restart  # mysql ‐u root –p  Enter password: 123456  mysql>SET PASSWORD=PASSWORD('123456'); （重置密码）  mysql> use mysql; (此DB存放MySQL的各种配置信息)  Database changed  mysql> select host,user from user; (查看是否能够查询) |

1. 设置远程登陆

执行如下命令

|  |
| --- |
| mysql> Grant all privileges on \*.\* to 'root'@'%' identified by '123456' with grant option; （允许远程用户登录mysql）  mysql>flush privileges; |

MYSQL的环境己基本配置完成。

1. 创建Cloudera Manager元数据数据库

执行如下命令，这些库是集群安装时时大数据组件所需要的。

|  |
| --- |
| --hive数据库  create database hive DEFAULT CHARSET utf8 COLLATE utf8\_general\_ci;  --oozie数据库  create database oozie DEFAULT CHARSET utf8 COLLATE utf8\_general\_ci;  --hue数据库  create database hue DEFAULT CHARSET utf8 COLLATE utf8\_general\_ci; |

#### 4.1.2.6 Cloudera推荐设置

在试安装的过程，发现Cloudera给出了一些警告，如下图



设置swap空间

|  |
| --- |
| echo "vm.swappiness = 10" >> /etc/sysctl.conf  ./scp.sh /etc/sysctl.conf /etc/ |

关闭大页面压缩  
试过只设置defrag，但貌似个别节点还是会有警告，干脆全部设置，编辑 /etc/rc.local ，末尾添加如下命令

|  |
| --- |
| echo never > /sys/kernel/mm/transparent\_hugepage/enabled echo never > /sys/kernel/mm/transparent\_hugepage/defrag |

同步到各个节点机器./scp.sh/etc/rc.local/etc/，然后重启 reboot 生效。

#### 4.1.2.7 安装Cloudera Manager

1. 进入 /opt/cdh 目录，将Cloudra Manager解压到/opt目录下，不能解压到其他地方，因为cdh5的源会默认在/opt/cloudera/parcel­repo寻找，而CM可以按照个人喜好安装。

|  |
| --- |
| tar ‐zxvf cloudera‐manager‐el6‐cm5.9.0\_x86\_64.tar.gz ‐C /opt/  mkdir ‐p /opt/program  mv /opt/cm‐5.9.0/ /opt/program/  ln ‐s /opt/program/cm‐5.9.0/ /opt/cm |

1. 将CDH­5.9.0­1.cdh5.9.0.p0.23­el6.parcel 和 CDH­5.9.01.cdh5.9.0.p0.23­el6.parcel.sha1移动到/opt/cloudera/parcel­repo，这样安装时CM就能直接找到了。

|  |
| --- |
| mv CDH‐5.9.0‐1.cdh5.9.0.p0.23‐el6.parcel CDH‐5.9.0‐1.cdh5.9.0.p0.23‐el6.parcel.sha1 /opt/cloudera/parcel‐repo/ |

1. 将CDH­5.9.0­1.cdh5.9.0.p0.23­el6.parcel.sha1重命名为CDH­5.9.0­1.cdh5.9.0.p0.23­el6.parcel.sha(去掉结尾的1)，非常重要。

|  |
| --- |
| mv manifest.json /opt/cloudera/parcel‐repo/ cd /opt/cloudera/parcel‐repo/ mv CDH‐5.9.0‐1.cdh5.9.0.p0.23‐el6.parcel.sha1 CDH‐5.9.0‐1.cdh5.9.0.p0.23‐el6.parcel.sha |

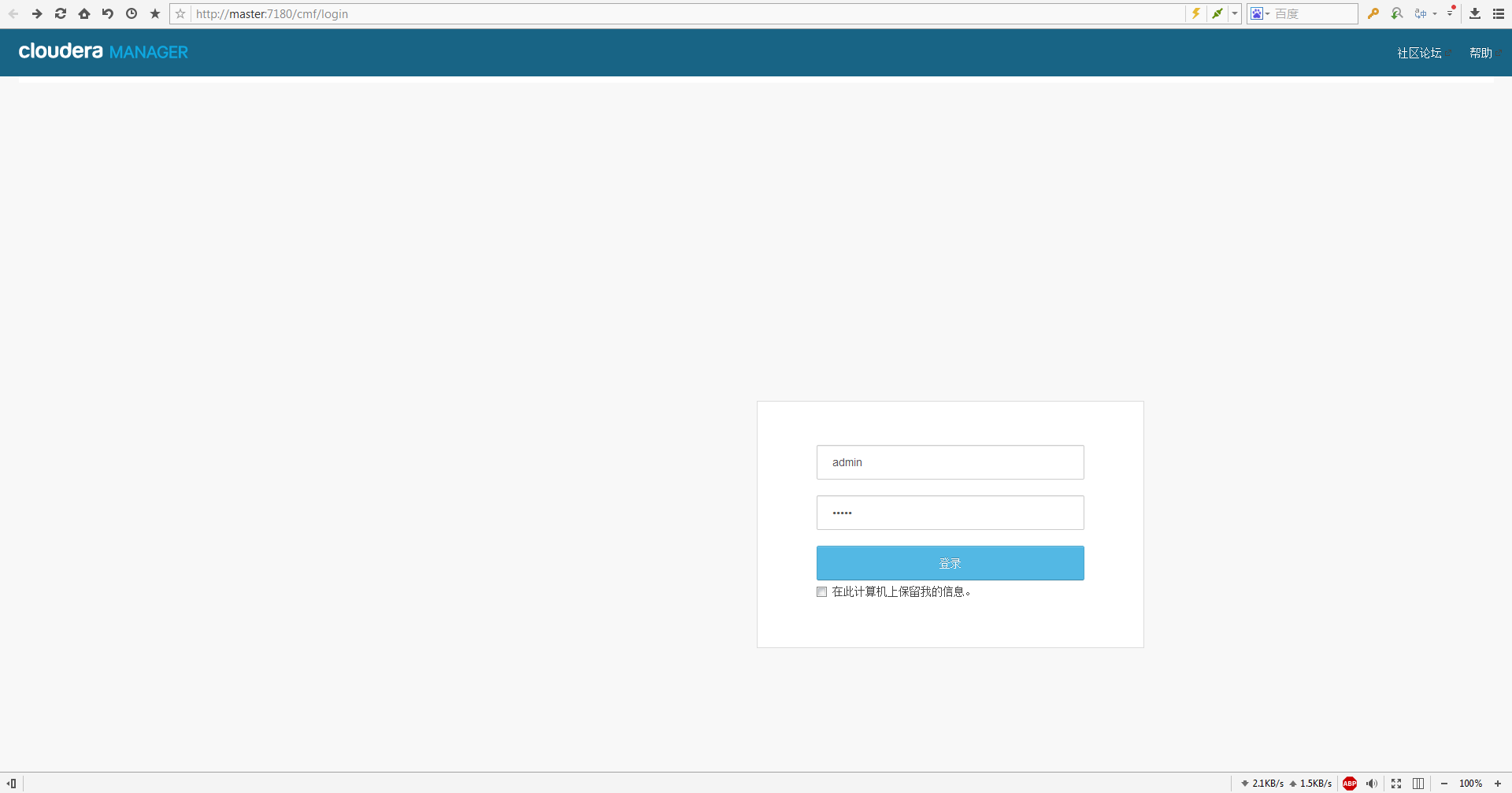
1. 修改配置文件中的server\_host，编辑

|  |
| --- |
| /opt/cm/etc/cloudera‐scm‐agent/config.ini server\_host=master |

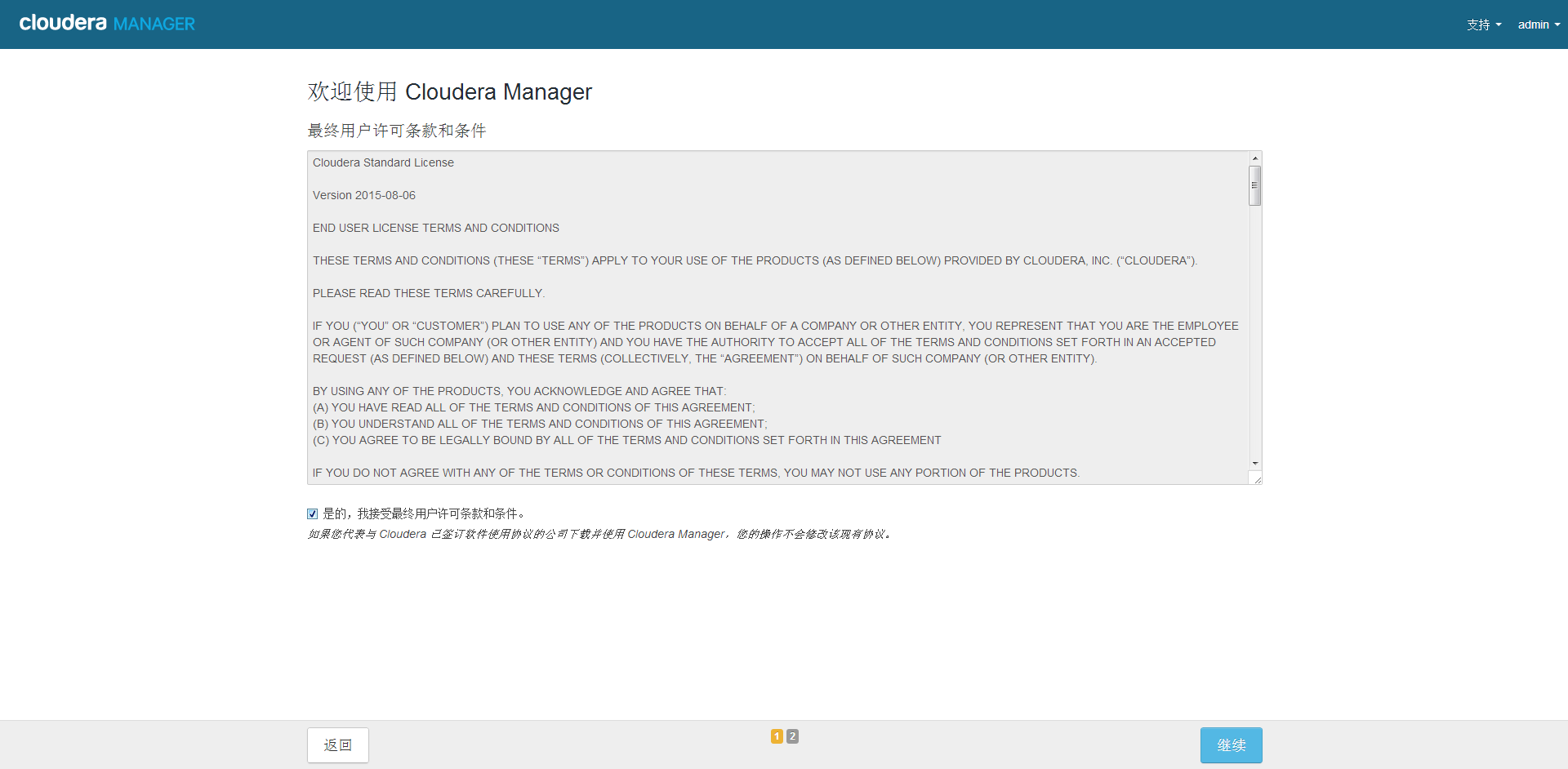
1. 将mysql的JDBC驱动放入CM的lib目录下，下载地址http://dev.mysql.com/downloads/connector/j/。解压获得mysql­connector­java­5.1.40­bin.jar上传到集群。  
   mv mysql‐connector‐java‐5.1.40‐bin.jar /opt/cm/share/cmf/lib/
2. 为CM创建数据库  
   /opt/cm/share/cmf/schema/scm\_prepare\_database.sh mysql cm ‐hlocalhost ‐uroot ‐p123456 ‐‐scm‐host  
   localhost scm scm scm
3. 为每个节点创建cloudera­scm用户，五台机均执行  
   useradd ‐‐system ‐‐home=/opt/cm/run/cloudera‐scm‐server ‐‐no‐create‐home ‐‐shell=/bin/false ‐‐  
   comment "Cloudera SCM User" cloudera‐scm接下来其实可以直接在master上启动服务安装了，但因为其它节点没有CM，最后还是通过远程的yum在线下载再安  
   装，我这设置了内部网络，其它节点是访问不了外网的，所以拷贝CM到其它节点进行完全离线安装。
4. 拷贝CM到每个节点  
   ./scp.sh /opt/program/cm‐5.9.0/ /opt/program/cm‐5.9.0/  
   ln ‐s /opt/program/cm‐5.9.0/ /opt/cm (每台机手动执行)
5. 在master上启动CM的service服务  
   /opt/cm/etc/init.d/cloudera‐scm‐server start
6. 在所有节点上面节点上启动CM的agent服务  
   /opt/cm/etc/init.d/cloudera‐scm‐agent start  
   service服务过程需要启动几分钟，这看机器性能，如果用 netstat ‐apn|grep 7180 能够查看到端口占用，就可以用浏览器打开http://masterIP:7180/进行访问，登录用户名与密码分别为admin到此基本上CM的安装启动己完成，接下来就可以用过WEB完成CDH平台安装。

#### 4.1.2.8 安装Cloudera Distribute Hadoop

1. 使用admin(密码admin)登录



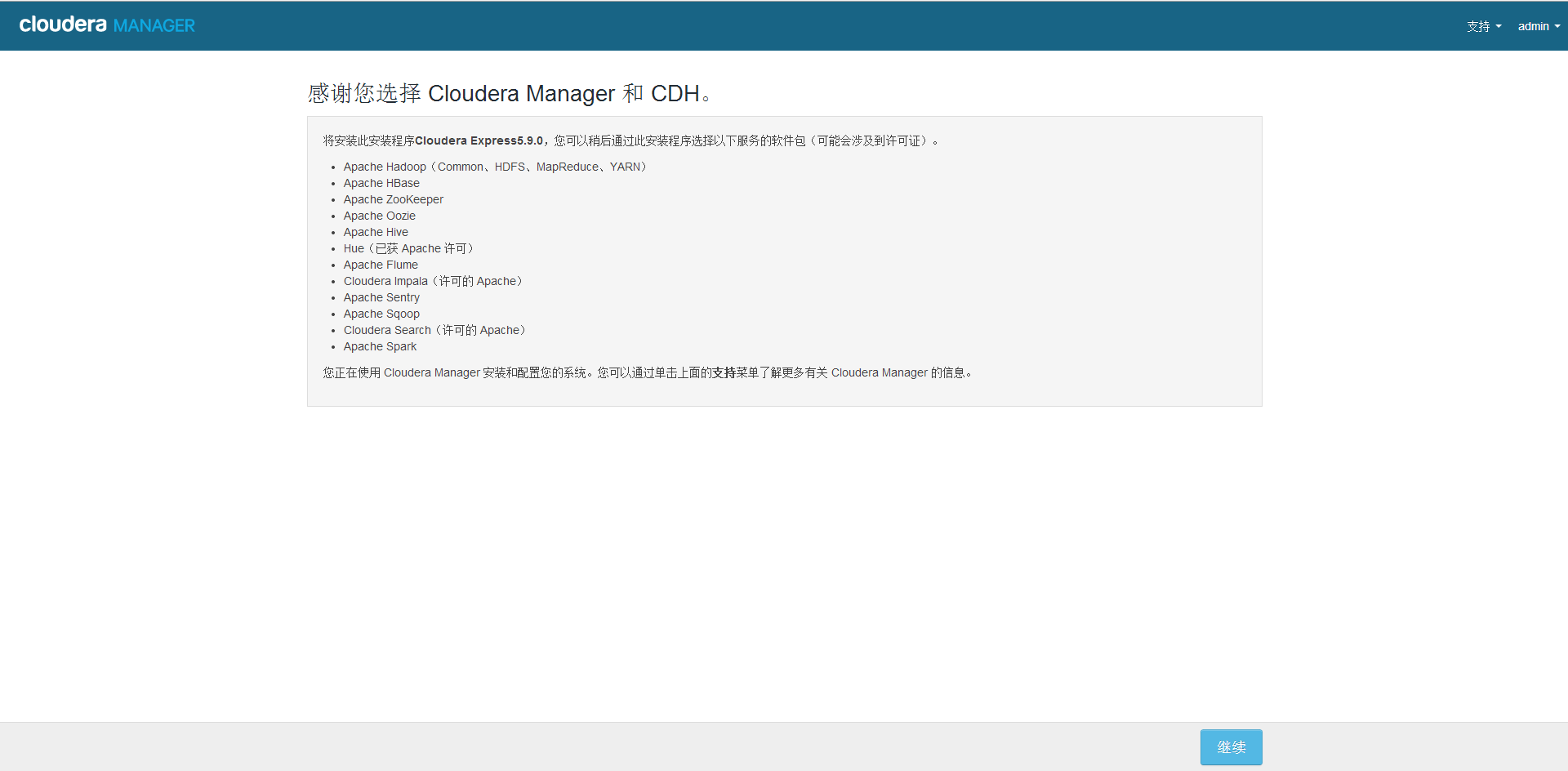
1. 选中“继续”



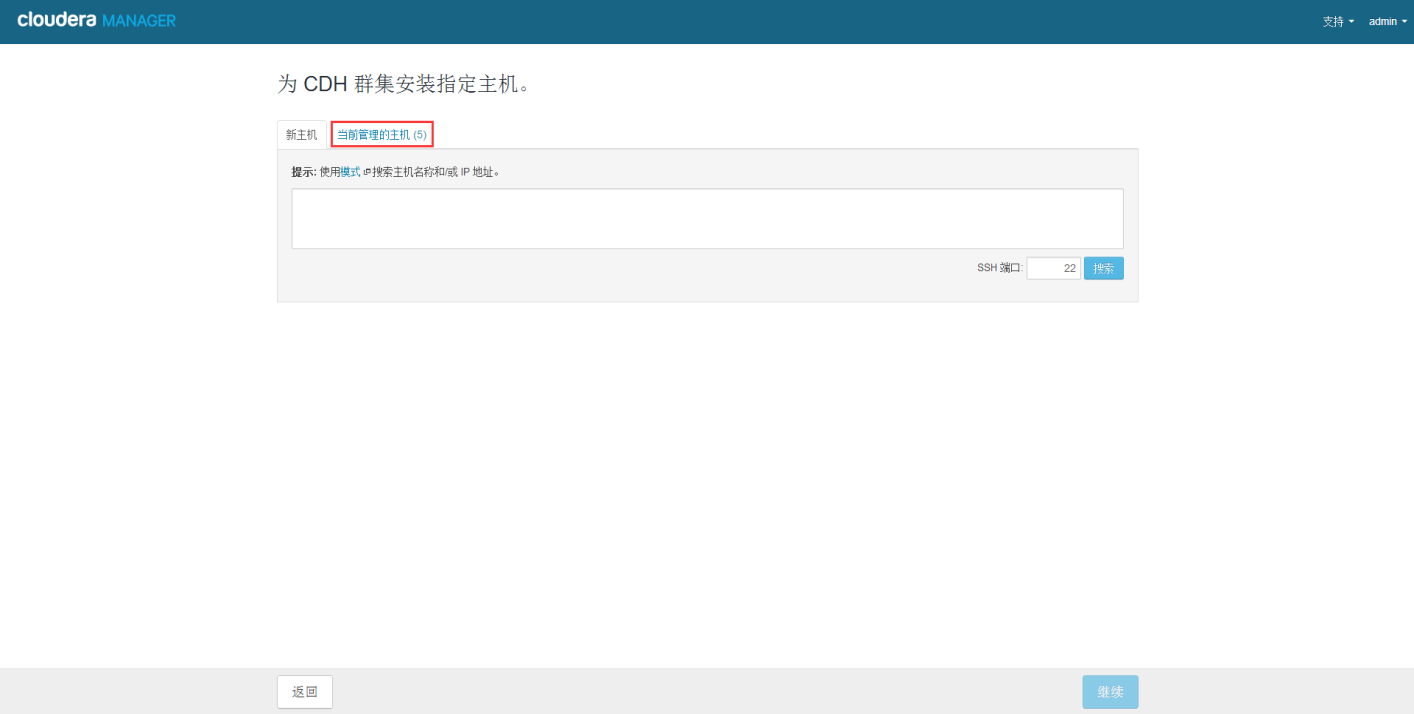
1. 按需选择，选择免费



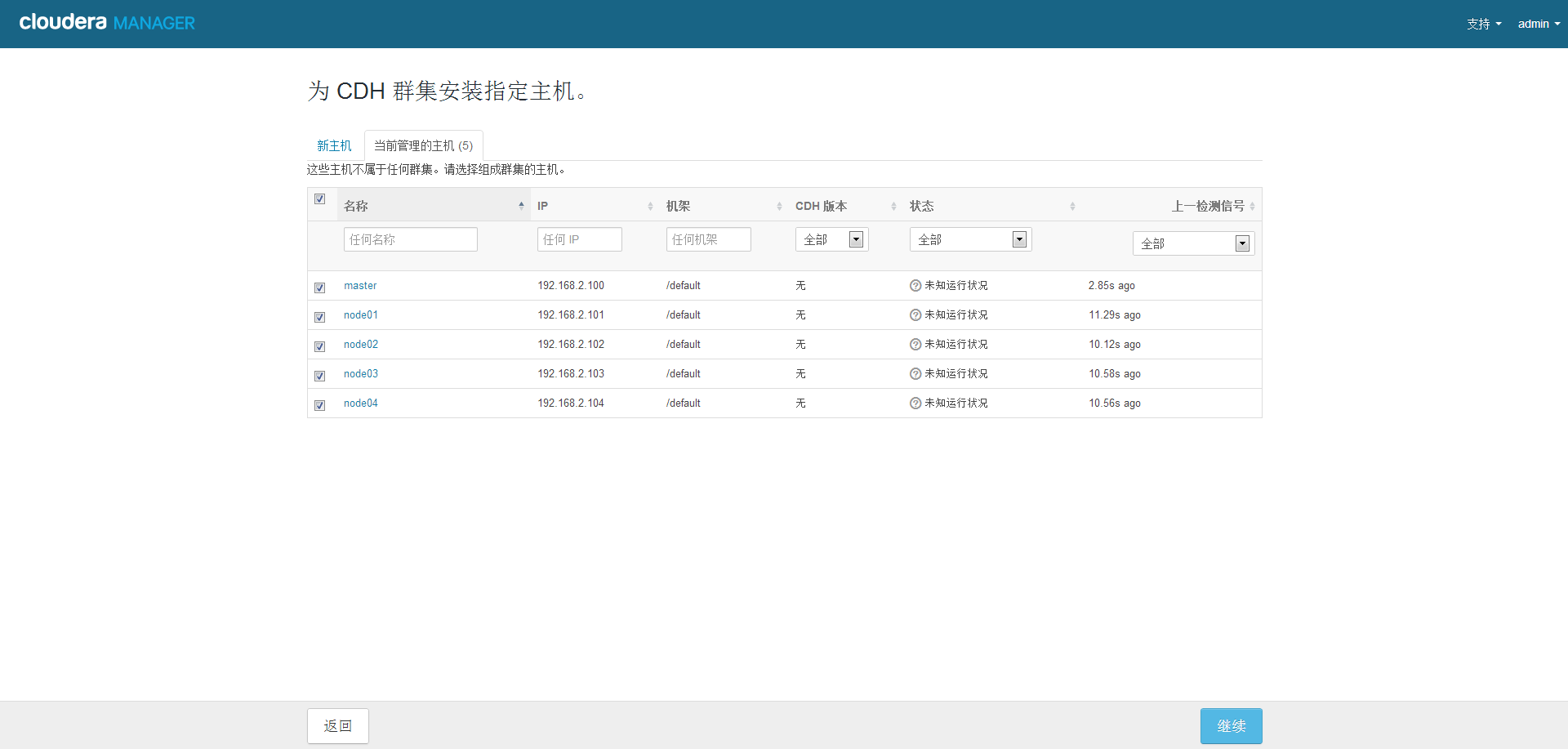
1. 继续



1. 节点上启动了agent，所以直接点“当前管理的主机”。如果节点上没有CM，只有master上有，这边可以在新主机下去搜索，例如172.16.18.[150-154]。

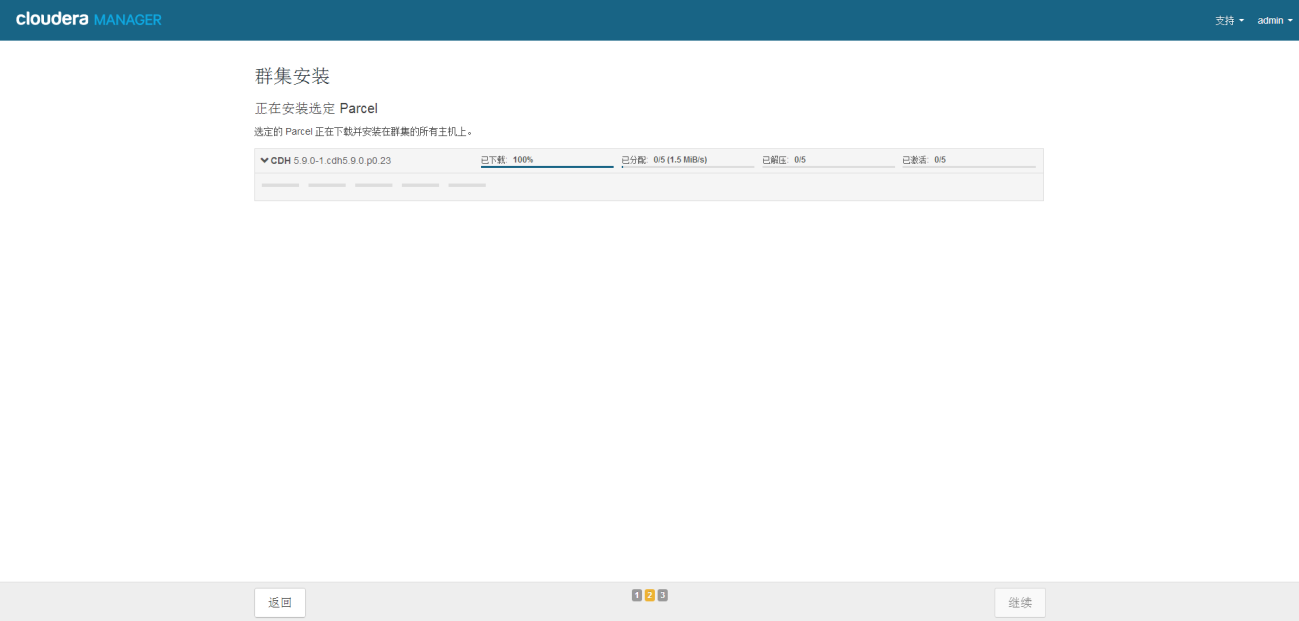


1. 全部勾上，然后继续

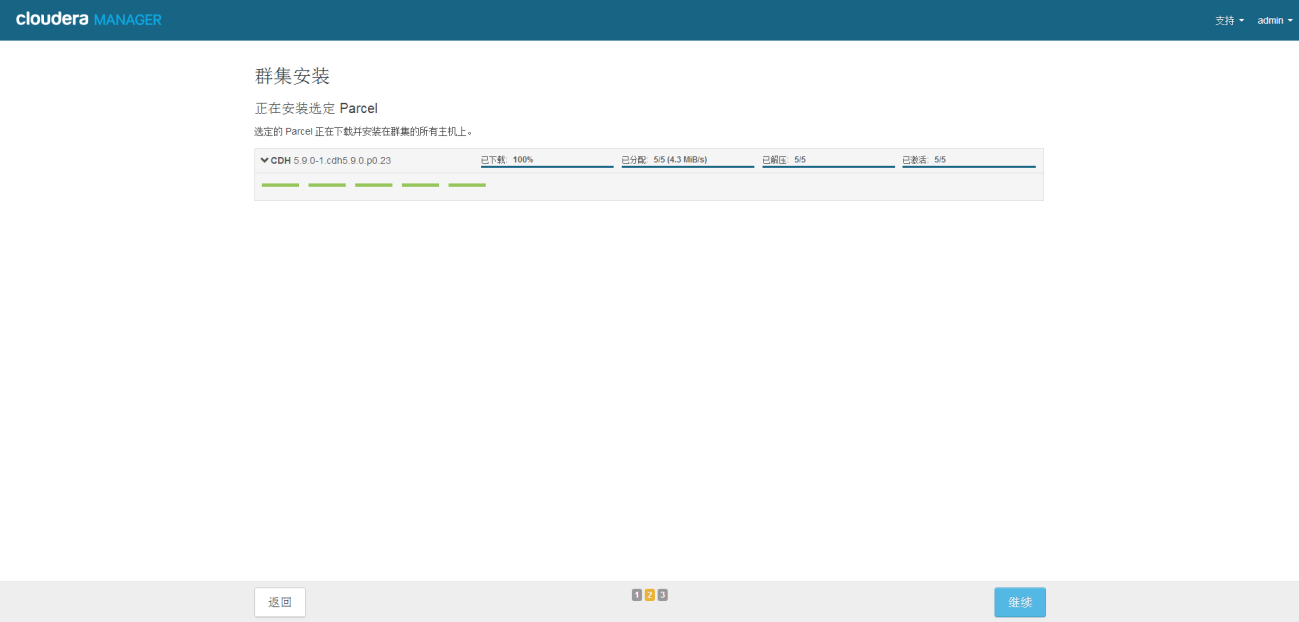


1. 选择版本，然后继续





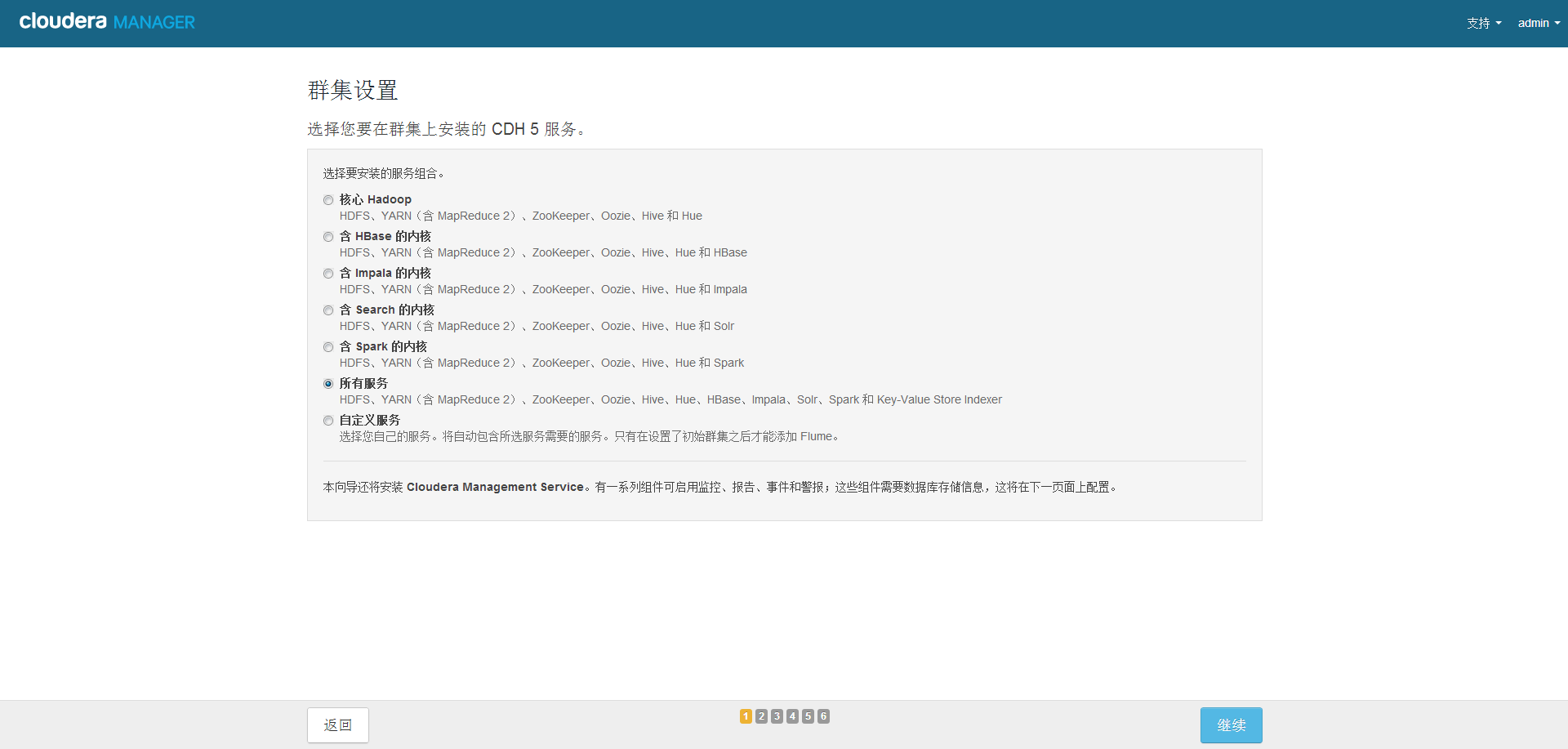
1. 集群分配完成后，继续



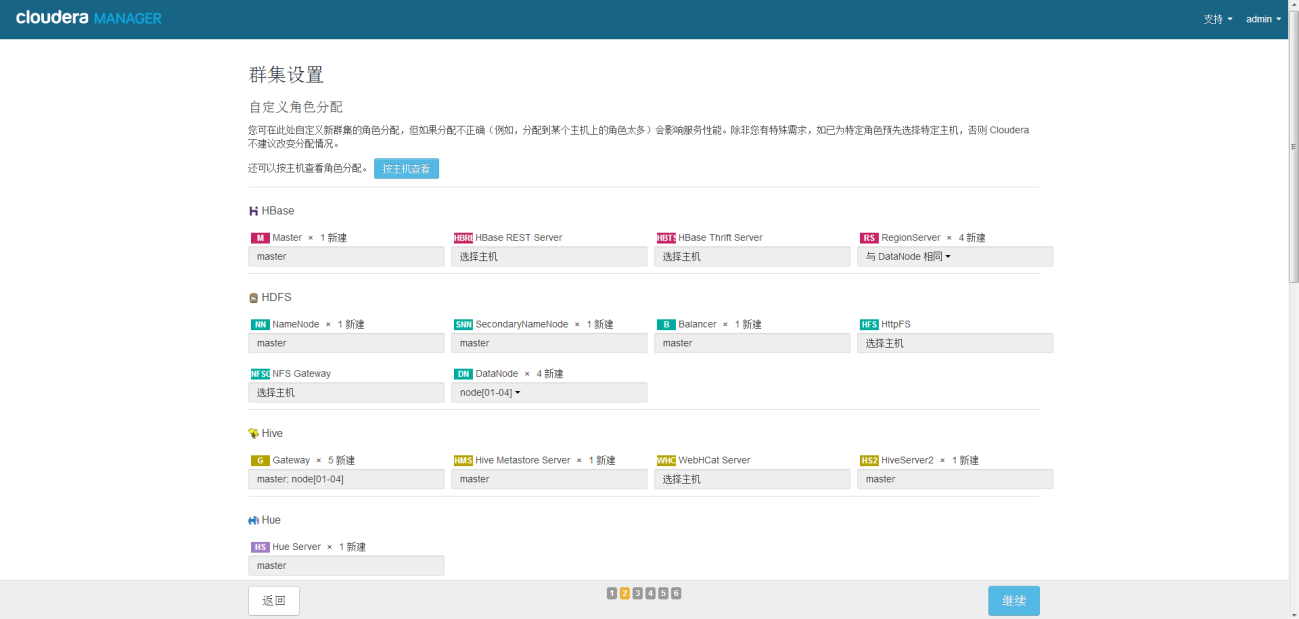
1. 显示检查列表，完成安装！



1. 集群服务分配，按需选择



1. 角色分配，按需分配



1. 创建Mysql数据库并测试(按需创建，比如你没选oozie，就不用创建oozie的数据库)

--hive数据库

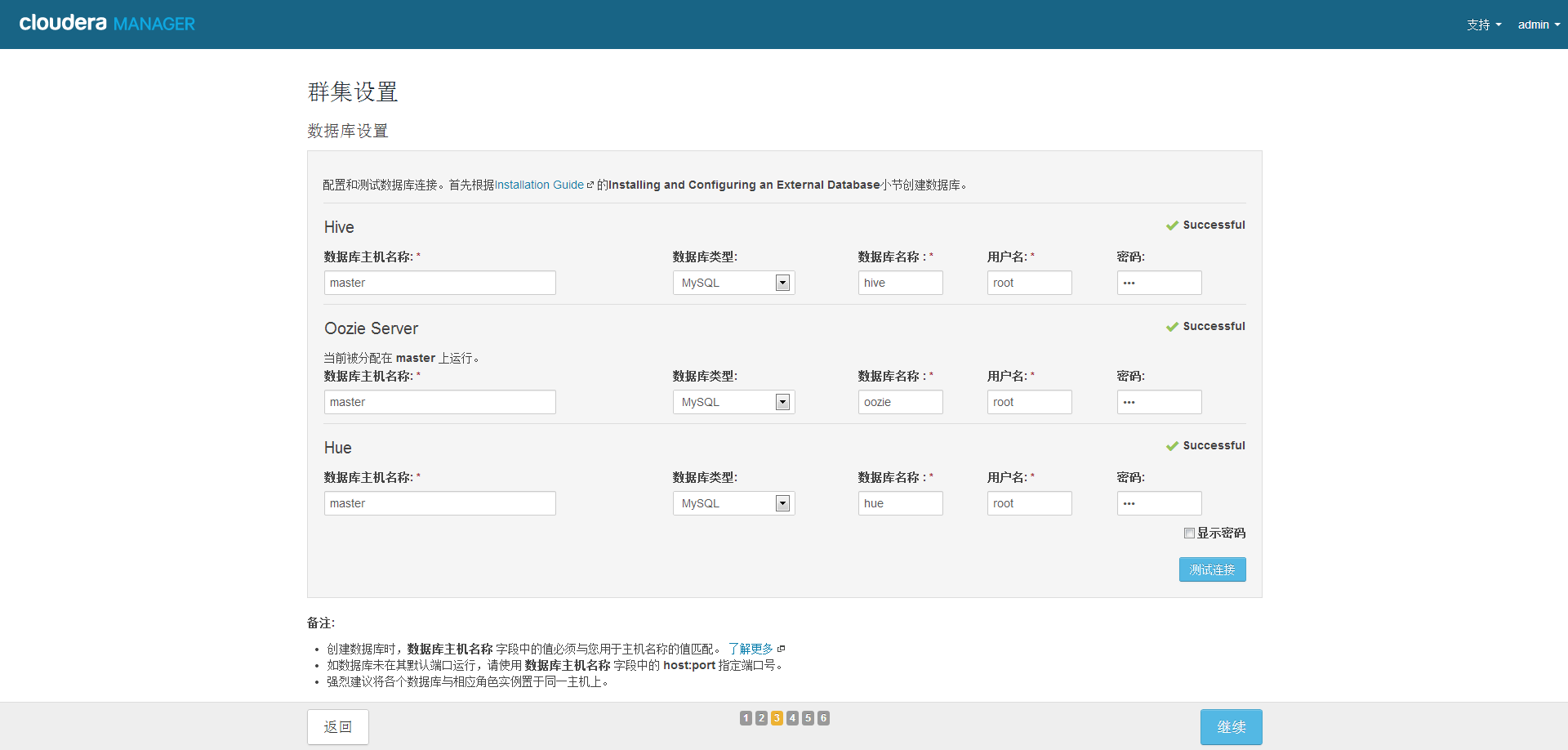
create database hive DEFAULT CHARSET utf8 COLLATE utf8\_general\_ci;

--oozie数据库

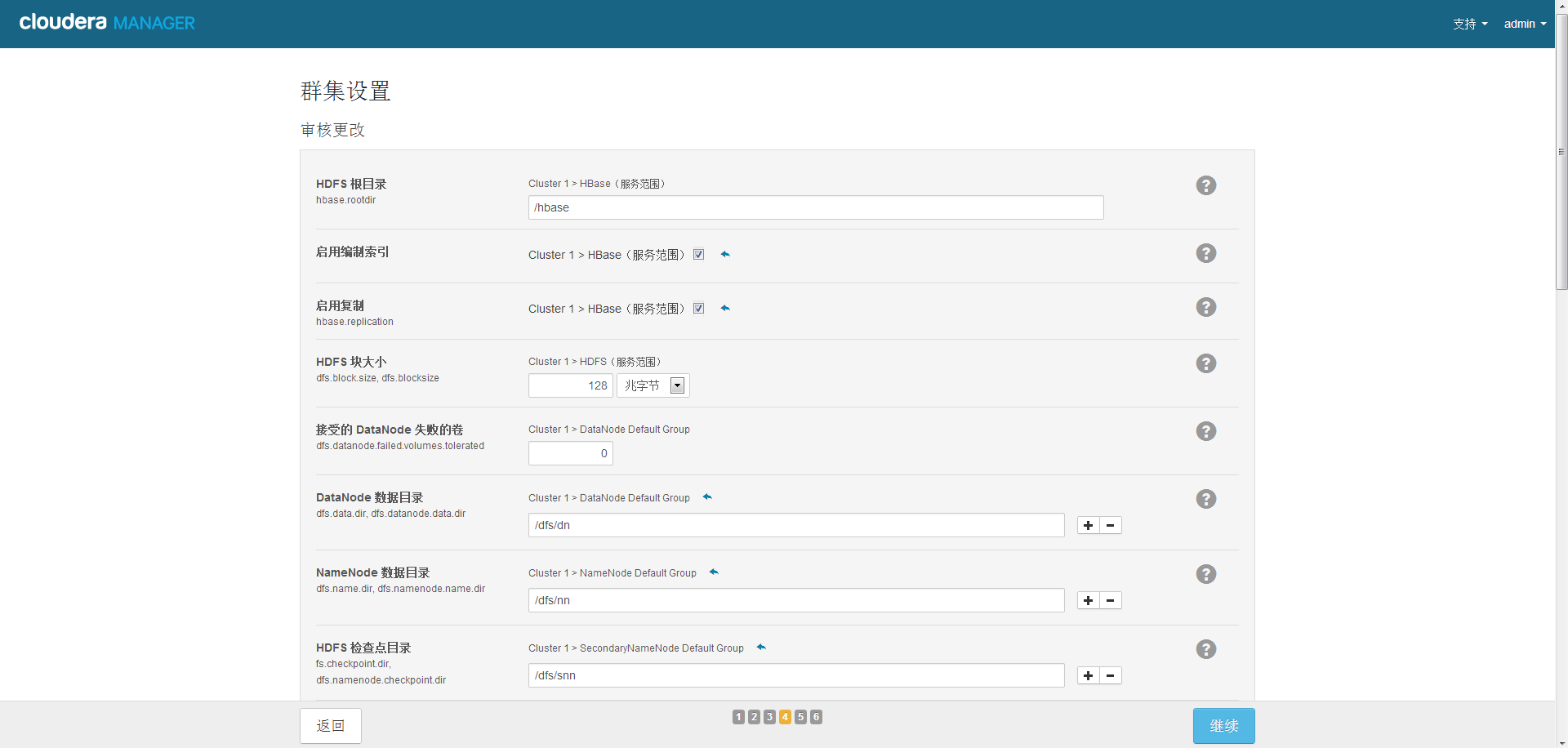
create database oozie DEFAULT CHARSET utf8 COLLATE utf8\_general\_ci;

--hue数据库

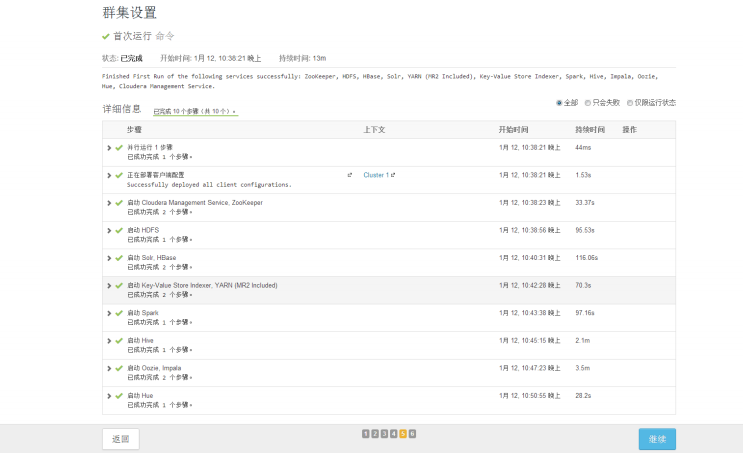
create database hue DEFAULT CHARSET utf8 COLLATE utf8\_general\_ci;



1. 集群设置

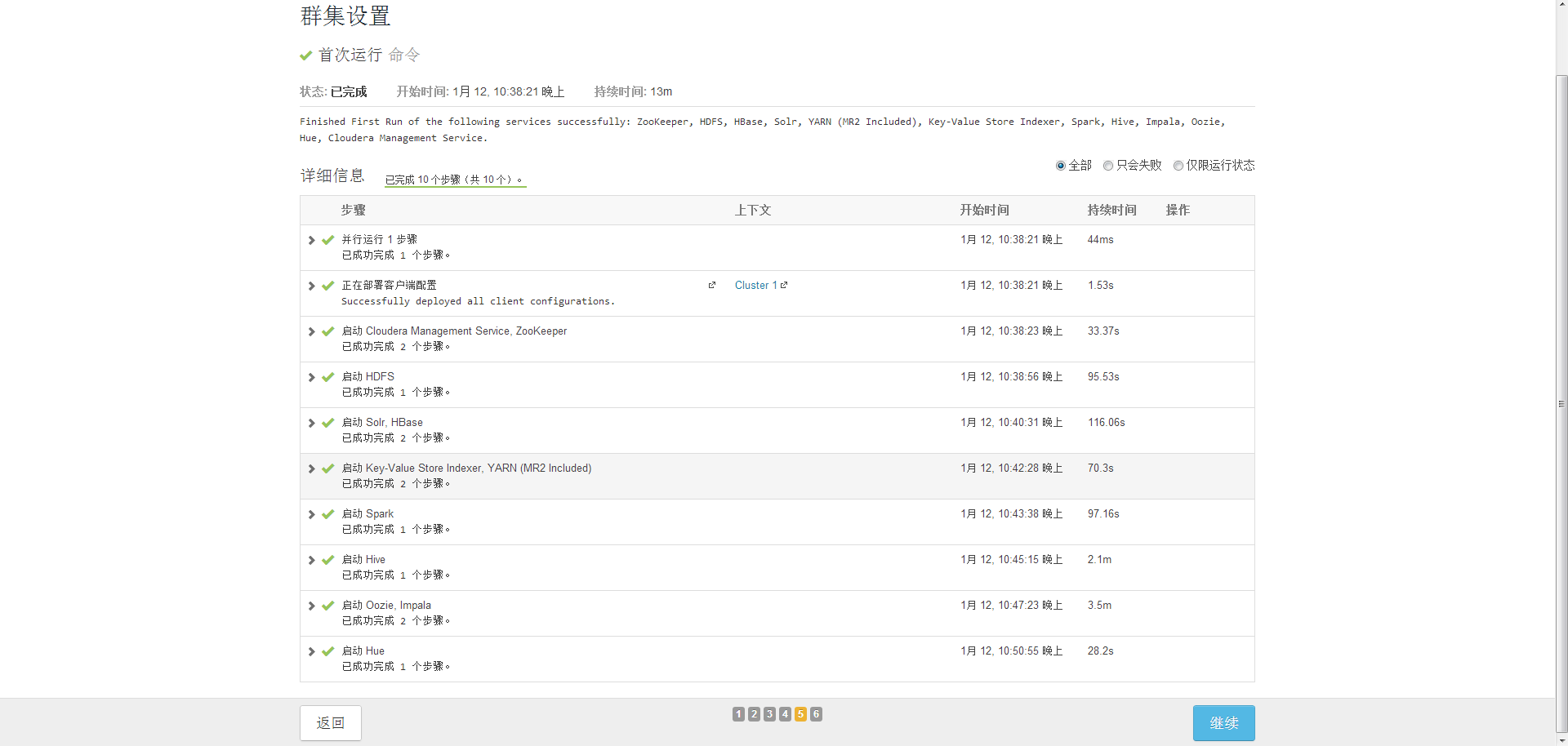


1. 启动集群

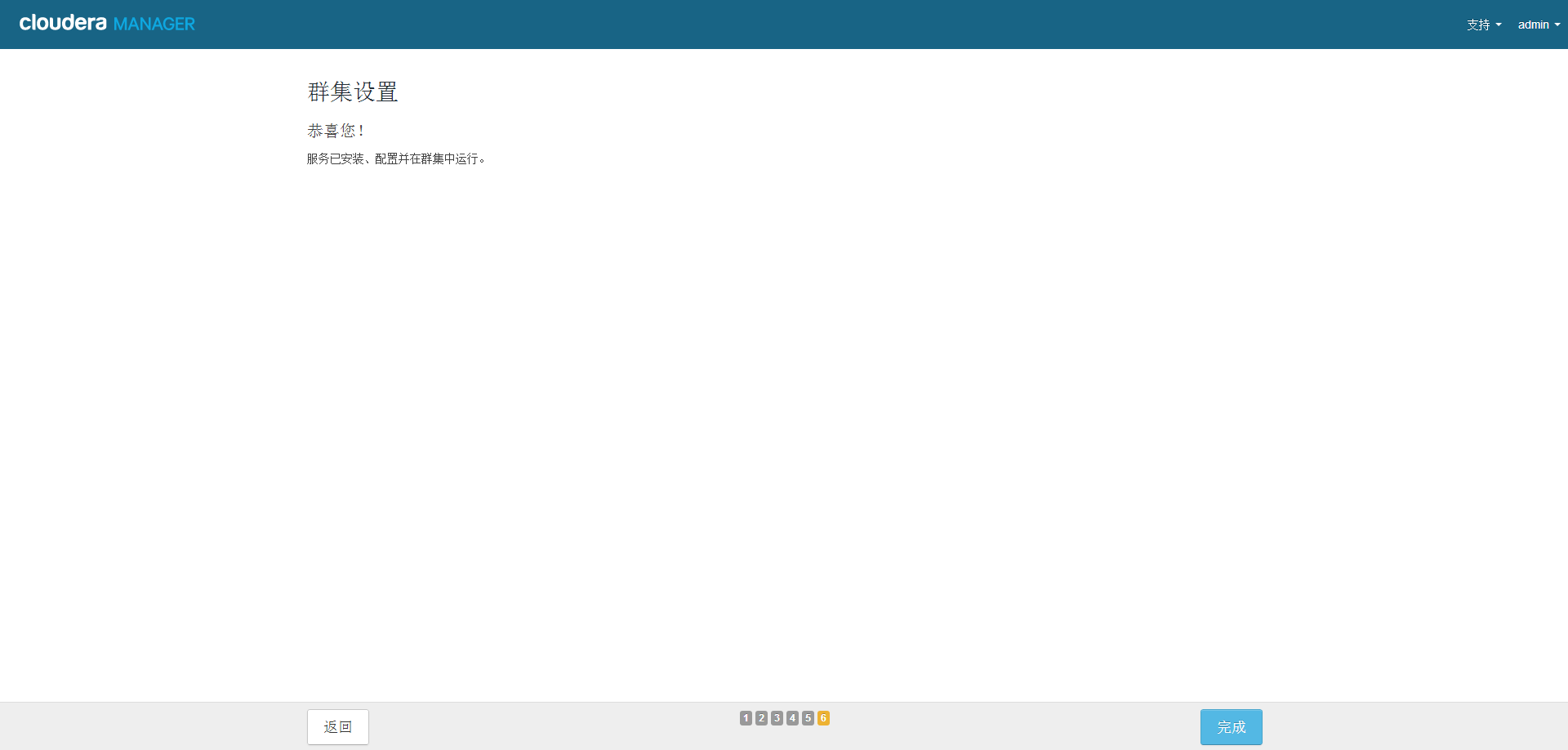


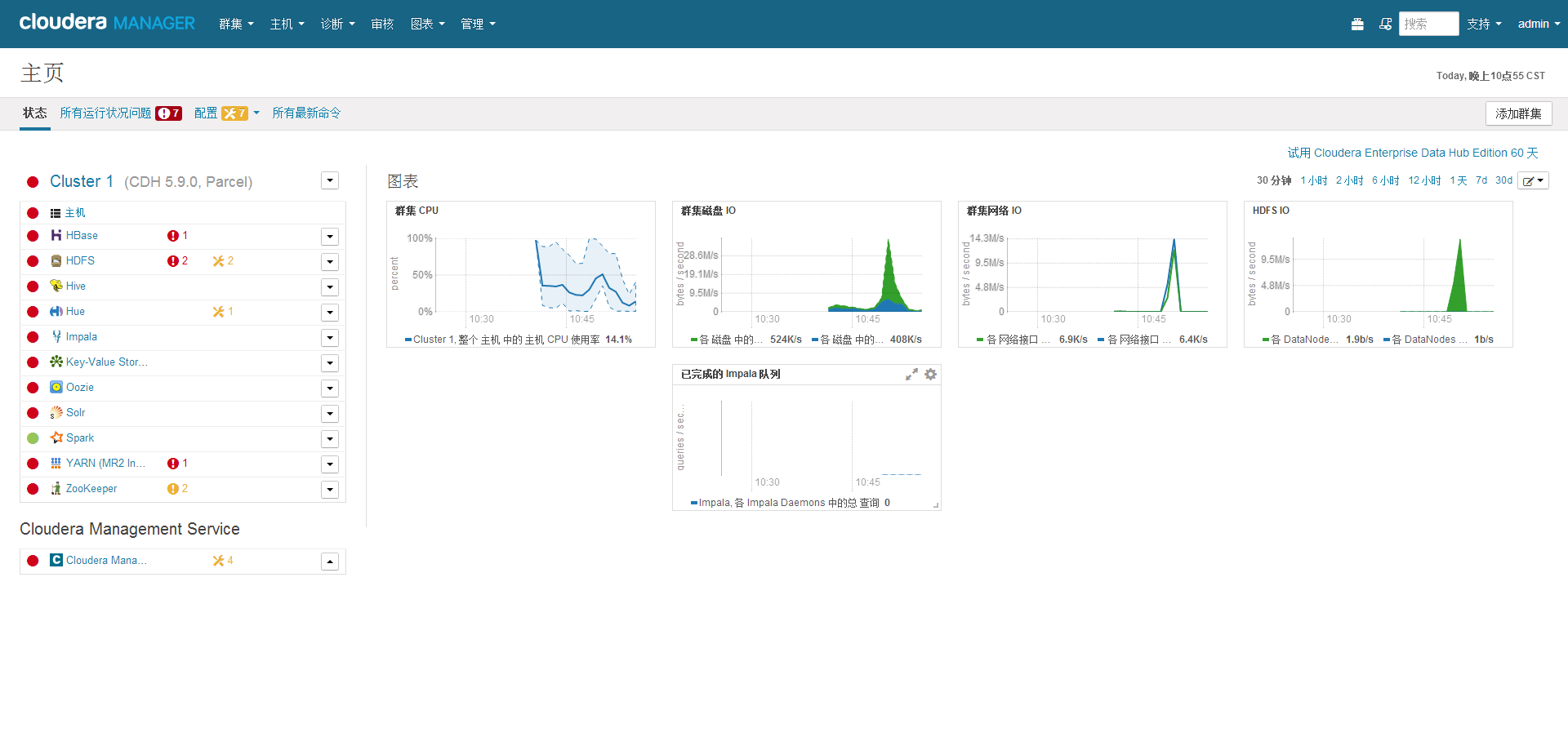
1. 大数据平台界面





20、终于装好了0.0





## 4.2 平台数据建模

### 4.2.1 数据建模工具与流程

数据采集模块主要功能是**连接**外部数据(应用程序日志文件、实时流数据或关系型数据库数据)或本地磁盘数据(本地文件系统或分布式文件系统)，并将这些各种数据源的各种格式的数据**按照一定的方式清洗、整理或预处理**后，**传输**给大数据实时计算模块或批处理模块分析处理，进一步通过可视化模块查看或展示处理结果。本文研究的高校数据主要为融合治理高校各信息化系统中存储到MySQL关系型数据库中的数据，因此使用开源Sqoop工具将数据导入到Hive分布式数据库中。

#### 4.2.2.1 import将MySQL表数据导入Hive

将MySQL数据库表的数据导入到hive中，如果在hive中没有对应的表，则自动生成与数据库表名相同的表。

sqoop import -connect jdbc:mysql://localhost:3306/hive -username root -password 123456 -table user -split-by id -hive-import

-split-by指定数据库表中的主键字段名，在这里为id。

|  |  |
| --- | --- |
| 参数 | 说明 |
| -append | 将数据追加到hdfs中已经存在的dataset中。使用该参数，sqoop将把数据先导入到一个临时目录中，然后重新给文件命名到一个正式的目录中，以避免和该目录中已存在的文件重名。 |
| -as-avrodatafile | 将数据导入到一个Avro数据文件中 |
| -as-sequencefile | 将数据导入到一个sequence文件中 |
| -as-textfile | 将数据导入到一个普通文本文件中，生成该文本文件后，可以在hive中通过sql语句查询出结果。 |
| -boundary-query <statement> | 边界查询，也就是在导入前先通过SQL查询得到一个结果集，然后导入的数据就是该结果集内的数据，格式如：–boundary-query ‘select id,creationdate from person where id = 3’，表示导入的数据为id=3的记录，或者select min(<split-by>), max(<split-by>) from <table name>，注意查询的字段中不能有数据类型为字符串的字段，否则会报错：java.sql.SQLException: Invalid value for getLong() |
| -columns<col,col,col…> | 指定要导入的字段值，格式如：–columns id,username |
| -direct | 直接导入模式，使用的是关系数据库自带的导入导出工具。官网上是说这样导入会更快 |
| -direct-split-size | 在使用上面direct直接导入的基础上，对导入的流按字节数分块，特别是使用直连模式从PostgreSQL导入数据的时候，可以将一个到达设定大小的文件分为几个独立的文件。 |
| -inline-lob-limit | 设定大对象数据类型的最大值 |
| -m,-num-mappers | 启动N个map来并行导入数据，默认是4个，最好不要将数字设置为高于集群的节点数 |
| -query，-e<statement> | 从查询结果中导入数据，该参数使用时必须指定–target-dir、–hive-table，在查询语句中一定要有where条件且在where条件中需要包含$CONDITIONS，示例：–query ‘select \* from person where $CONDITIONS ‘ –target-dir /user/hive/warehouse/person –hive-table person |
| -split-by<column-name> | 表的列名，用来切分工作单元，一般后面跟主键ID |
| -table <table-name> | 关系数据库表名，数据从该表中获取 |
| -target-dir <dir> | 指定hdfs路径 |
| -warehouse-dir <dir> | 与–target-dir不能同时使用，指定数据导入的存放目录，适用于hdfs导入，不适合导入hive目录 |
| -where | 从关系数据库导入数据时的查询条件，示例：–where ‘id = 2′ |
| -z,-compress | 压缩参数，默认情况下数据是没被压缩的，通过该参数可以使用gzip压缩算法对数据进行压缩，适用于SequenceFile, text文本文件, 和Avro文件 |
| -compression-codec | Hadoop压缩编码，默认是gzip |
| -null-string <null-string> | 可选参数，如果没有指定，则字符串null将被使用 |
| -null-non-string<null-string> | 可选参数，如果没有指定，则字符串null将被使用 |

增量导入

|  |  |
| --- | --- |
| 参数 | 说明 |
| -check-column (col) | 用来作为判断的列名，如id |
| -incremental (mode) | append：追加，比如对大于last-value指定的值之后的记录进行追加导入。lastmodified：最后的修改时间，追加last-value指定的日期之后的记录 |
| -last-value (value) | 指定自从上次导入后列的最大值（大于该指定的值），也可以自己设定某一值 |

对incremental参数，如果是以日期作为追加导入的依据，则使用lastmodified，否则就使用append值。

### 4.2.2 高校数据采集

高校大数据采集主要工作为按照数据清洗规则将高校各种信息化系统中的数据提取、转换、加载到大数据治理融合平台的数据仓库中。

根据3.2节需求，本文主要采集如下类别数据：

|  |  |  |  |
| --- | --- | --- | --- |
| 数据类别 | 数据用途 | 数据源 | 数据表 |
| 学生数据 |  |  |  |
| 宿舍数据 |  |  |  |
| 一卡通数据 |  |  |  |
| 上网数据 |  |  |  |
| 考核数据 |  |  |  |
| 课程数据 |  |  |  |
| 机构数据 |  |  |  |
| 教师数据 |  |  |  |
| 图书数据 | 分析学生借阅记录；图书采购指导、图书推荐、统计分析等 | 图书馆管理系统 |  |
| 医疗数据 | 学生健康监控、流感预测、医药用品采购 | 医院管理系统 |  |
| 就业数据 | 就业统计、聚类分析、辅助识别优秀毕业生、辅助图书推荐及学生行为分析 | 就业管理系统 |  |

### 4.2.3 数据清洗

数据清洗的主要目的

|  |  |  |
| --- | --- | --- |
| 名称 | 描述 | 不规范数据 |
| 确定性 | 数据具有一个一致统一的业务定义和计算规则 | 在高校内部，不同信息化系统使用不同的算法和源数据计算投入、收益及资源使用 |
| 有效性 | 数据与它的定义(数据类型、格式、内容)相符合 | 学生记录包含数字或乱码，无法使用 |
| 唯一性 | 数据时独一无二的—没有冗余 | 两个学生具有相同的学号 |
| 完整性 | 数据时不可缺的或与另一个数据关联的 | 学生记录缺少学号  婚否(Y/N)填写的是“空”值 |
| 一致性 | 数据元素不会因为有了另一个数据而产生变化或自相矛盾 | 学生的毕业日期比入学日期早 |
| 及时性 | 数据代表谋业务活动的产出物的最当前信息 | 学生家庭联系地址和方式是旧的，联系电话是空号，没有及时更新 |
| 准确性 | 恰当地指派了数据值 | 学生年级分班具有一个不准确或无效的值 |
| 精确性 | 数据只能按它被界定的定义去使用，即数据特征被理解和准确地应用的程度 | 实验室设备被用于不同记录的不同产品类型 |

高校数据清洗举例：

* **清洗规则**：门禁系统，把没有卡ID的数据过滤，并把每个卡ID与用户ID做关联，记录到门禁记录中
* **一卡通系统**：管理学生ID，增加学生姓名与学号
* **教务系统**：过滤成绩为0的记录，过滤重复的课程，通过门禁系统、增加宿舍位置等全方位完整信息
* **学生信息**：过滤无效的地址信息，通过身份证号码，适配学生身份、城市信息等

### 4.2.4 数据建模

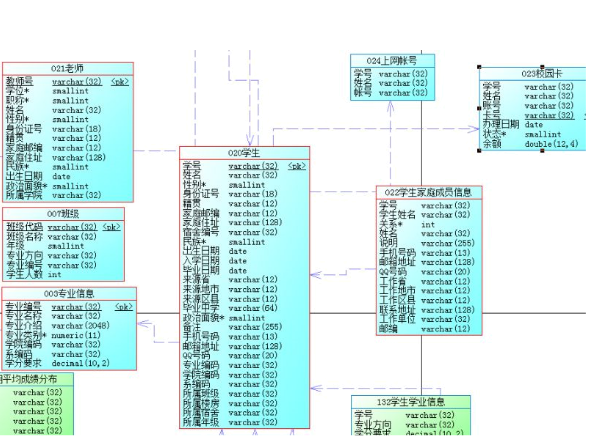
[//5.1日补充高校数据表]

按照第三章的需求分析，作者调研了 教育信息标准化表格、学校调研、论文查阅，整理了XXX类XXX张表，如下清单所示：

类别|表名|表描述|表字段|

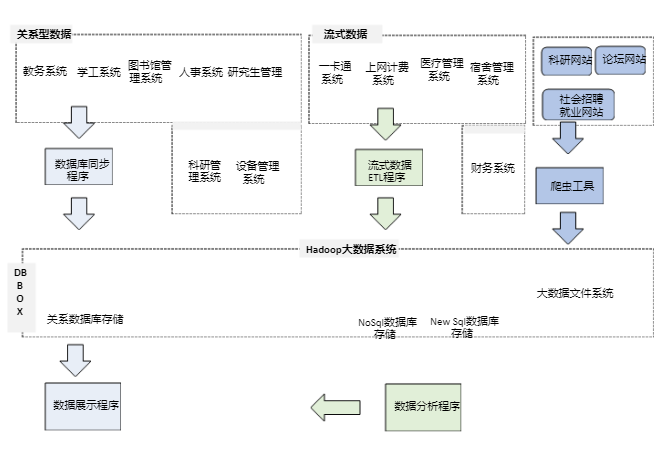
建表语句见附件

表的关联关系，E-R逻辑模型如下图：



### 4.2.5数据导入Hive

[//5.1日完成数据导入实例]



//【删除该图】

## 4.3 高校大数据应用场景实现

### 4.3.1 图书馆大数据分析

大数据存储包含多种数据存储方式，具体为分布式文件系统（DFS）、非关系型数据库（NOSQL）及关系型数据库（RDBMS），满足数据连接、预处理、加工与呈现不同场景下业务能力的需求。

#### 4.3.1.1 设计思路

数据存储已经有成熟的解决方案，主要使用开源分布式文件系统HDFS及基于分布式文件系统的NoSQL数据库HBase等。数据存储主要工作体现在设备组网、HDFS及HBase配置优化等内容，相对其它模块而言，工作量较小。

#### 4.3.1.2

### 4.3.1.5

#### 4.3.1.6 高校大数据治理融合平台存储方案

根据3.1章节介绍的高校大数据架构设计，高校大数据治理融合平台存储层由Hive、HDFS、Hbase、MySQL、MongoDB等分布式文件系统或数据仓库构成。数据存储层的访问接口与业务数据关系，如下图所示：



图‑2 数据存储及业务关系图

### 4.3.2 数据处理

根据数据计算的实时性：分为实时数据计算和离线批处理：

* **实时数据计算** 往往要求系统在分钟或数秒内返回上亿行数据的分析，从而才能达到不影响用户体验的目的。
* **离线批处理** 对大多数反馈时间要求不高的应用，比如离线统计分析、机器学习等，应采用离线分析的方式，通过数据采集工具将日志数据导入专门的分析平台进行分析。

系统主要以离线数据分析为主，采用目前在互联网业界流行的Hadoop体系结构及Spark技术栈对大批量的数据进行运算，采用集群的方式对大数据进行运算。

数据运算平台以调度为主线，作为运算平台的核心控制系统，对运算平台的各个环节进行控制，且对运算过程中的步骤依赖关系进行控制，同时对各个环节进行监控，通过监控异常报警来提高系统的稳定性和异常响应速度。数据计算模块如下图所示：

图4‑‑3 数据处理流程

#### 4.3.2.1 实时数据计算

大数据的应用场景：数据源是实时的不间断的，数据量大且无法或没有必要预算，但要求响应时间是实时的。为了解决这类场景，出现了大数据实时数据计算技术，常见的实时数据计算技术有Storm、SparkStreaming、Druid、Pinot，Kylin，其对比如下表所示：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **对比项** | **Storm** | **SparkSteaming** | **Druid** | **Pinot** | **Kylin** |
| 技术 | 流式计算 | 切片处理，内存计算 | lamda架构、自研引擎 | 类Druid架构  存内存计算 | MapReduce预计算 |
| 性能 | 性能高，亚秒 | 性能较高，秒 | 毫秒及亚毫秒级速度， | 毫秒及亚毫秒级速度， | 性能一般：分钟级别 |
| 成熟度 | 起步早，技术成熟 | 起步晚，技术较成熟 | 技术成熟 | 初步阶段 | 初级研究阶段 |
| 支持语言 | JVM语言/Python/RubyPerl | Java/Scala/Python | java | Java | Java |
| 市场普及率 | 生产环境占有率，高 | 较高 | 较高 | 一般 | 无 |
| 使用难度 | 一般 | 一般 | 较难 | 较难 | 无参考 |
| 文档及社区 | 文档丰富、社区活跃 | 文档丰富、社区活跃 | 文档较少 | 文档少，活跃 | 几乎没有 |
| 代码活跃度 | 一般 | 高 | 高 | 高 | 高 |

本设计方案支持基于Spark上的Spark Streaming，满足秒级或毫秒级的实时计算场景需求，如实时推荐、用户行为分析等。**考虑支持**Storm流式任务作业引擎，覆盖实时要求极高的流式作业场景。实时数据计算流程图，如下图所示：

图 13 实时计算流程图

#### 4.3.2.2 离线批量处理

离线批处理计算已经成熟，业界常见的方案有两种：MapReduce任务及Spark。本设计方案根据技术栈一致性及性能指标出发，主要以Spark作为离线批处理的主要选择，当数据量超过内存配置时或时效性要求不高的非重要任务辅助使用MapReduce任务处理。

图 14 离线批处理流程图

## 4.4 可视化系统

### 4.4.1 可视化系统架构



**Web层**：主要用于展示数据，显示、增加、修改、删除数据，websocket用于实时更新页面数据。管理用户的请求和响应提供一个控制起来将将调用委托业务逻辑和其他上游处理组装可以在视图中表现的模型对象执行UI校验实现拖拽。

**服务层**，实现各个业务功能，采用微服务架构。通过分解巨大单体式应用为多个服务方法解决复杂性问题，在功能不变的情况下，应用被分解为多个可管理的分支或服务。可以把每个服务独立部署。避免服务之间的影响。

**DAO层**，采用JPA提供更加简单的的编程模型，定义了Java ORM及实体操作API的标准。除了提供面向对象的ORM框架外，还提供原始的JDBC操作数据的方式，使平台更加灵活。

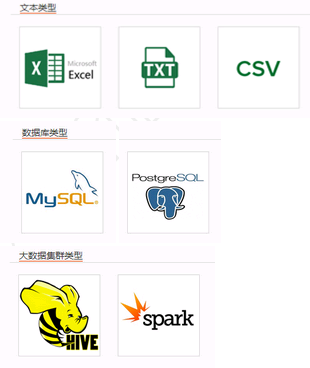
### 4.4.2 数据源管理

高校大数据治理融合平台的数据源支持对多种常见数据库的连接，包括MySql、PostgreSQL、Excel等数据源。数据源依附应用而存在，若依附的应用被删除，该应用中包含的数据源也会被删除。

* 文本类型，支持上传CVS/Excel文件。只支持已经格式化的文件。并且大小限制500MB。
* 数据库类型，MySQL、PostgreSQL。
* 大数据集群类型，Hive、Spark等。



图表 ‑4 数据源对接流程图



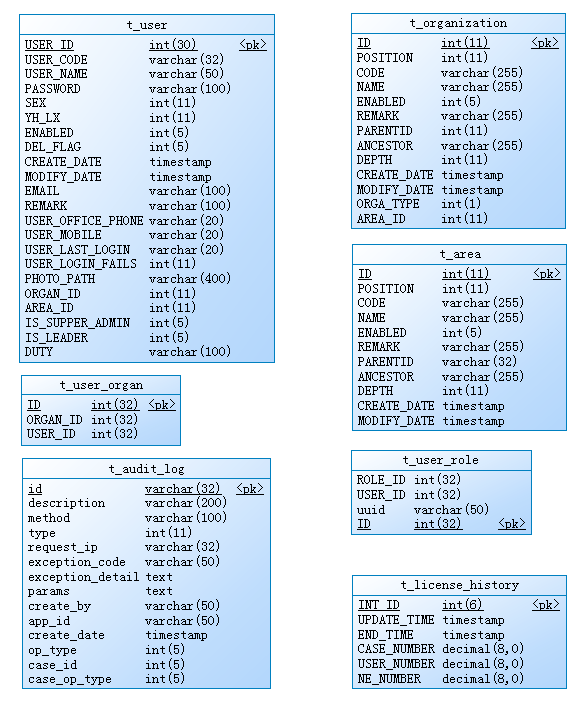
图表 ‑5 数据源对接界面

### 4.4.3

### 4.4.4 业务前端展示图表

业务前端利用echarts部分可视化图表，包括柱状图、饼图、折线图、热力图、地图、雷达图、仪表盘、预警信号、网络图等。

### 4.4.5 可视化系统数据库设计



#### 4.4.5.1 数据源表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | cbi\_datasource |  |  |
| 字段ID | 字段名称 | 字段类型 | 备注 |
| Id |  | Bigint | 主键 |
| config | 连接参数 | Text | json |
| db\_type | 数据库类型 | Int |  |
| Is\_copy | 是否复制 | Int |  |
| brief | 概要 | text |  |
| is\_success | 是否连接成功 | Int | 0:成功；1：失败 |

#### 4.4.5.2 应用表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | cbi\_object |  |  |
| 字段ID | 字段名称 | 字段类型 | 备注 |
| Id |  | Bigint | 主键 |
| Code | 编码 | Text | json |
| Data\_attr\_type | 数据类型 | Int |  |
| Create\_time | 创建时间 | Datetime |  |
| Modified\_time | 修改时间 | Datetime |  |
| Is\_del | 是否删除 | Int | 0:成功；1：失败 |
| Levels | 等级 | Int |  |
| Name | 名称 | Longtext |  |
| Order | 排序 | Int |  |
| Owner | 拥有者 | bigInt |  |
| Parent\_id | 父级ID | Bigint |  |
| Project\_id |  | Bigint |  |
| Status | 状态 | Int |  |
| Sub\_type |  | Int |  |
| type |  | int |  |

#### 4.4.5.3 元数据存储表

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | cbi\_colmun |  |  |
| 字段ID | 字段名称 | 字段类型 | 备注 |
| Id |  |  | 主键 |
| Code |  | text |  |
| Data\_type | 数据类型 | Int | 1：维度；2：度量 |
| Expr |  | Text |  |
| Foreign\_key\_tableid | 外键表ID | Bigint |  |
| Is\_del | 是否删除 | Int |  |
| Is\_formula | 是否公式 | Int |  |
| Is\_partition | 是否分割 | Int |  |
| Is\_primary | 是否主键 | Int |  |
| Mapping\_type | 映射类型 | Int |  |
| Name | 名称 | Text |  |
| Object\_name | 对象名称 | Text |  |
| Order | 排序 | Int |  |
| Talbe\_id | 表ID | Bigint |  |
| label |  | Text |  |
| Original\_name | 原始名称 | Text |  |

#### 4.4.5.4 仪表盘表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | cbi\_board |  |  |
| 字段ID | 字段名称 | 字段类型 | 备注 |
| Id |  |  | 主键 |
| Auto\_update | 是否自动更新 | int |  |
| Config | 配置信息 | Text | Json |
| Descp | 描述信息 | Text |  |
| Is\_booked | 是否 |  |  |
| Is\_public | 是否公开 |  |  |
| Need\_report | 是否需要统计 |  |  |

#### 4.4.5.5 图表表结构

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | cbi\_chart |  |  |
| 字段ID | 字段名称 | 字段类型 | 备注 |
| Id |  |  | 主键 |
| Board\_id | 仪表盘ID | int |  |
| C\_type | 数据类型 | Int |  |
| Config | 配置信息 | Text | Json |
| Is\_update | 是否更新 | Int |  |
| Last\_update\_time | 最后更新时间 | Datetime |  |
| Update\_interval | 更新间隔 | Bigint |  |

#### 4.4.6 可视化分析实例

## 4.6 本章小结

本章详细介绍了高校大数据治理融合平台的设计方案与实现方法。使用三台Dell服务器及4台虚拟机搭建了高校大数据治理融合平台试验环境。利用CDH大数据管理系统安装、部署了大数据集群服务，如Hive、Yarn、HDFS、Spark、Kafka、Zookeeper、Flume、Habse等。分析了数据建模相关知识点，即建模使用的工具、流程、规范、规则等，并根据高校大数据的数据内容设计了数据模型。详细展示了数据存储方案及可视化系统设计方案与实现方法。

# 第六章 总结与展望

## 5.1 论文工作与总结

本文的核心工作包括但不限于：

1) 大数据应用平台技术选型：体验分析数十种大数据服务组件，根据组件服务的成熟度流行度、技术平台能提供API或基于源码开发、企业实践验证情况及社区支持等方面，选取了高校大数据治理融合平台的基础平台及组件。

2) 大数据应用平台架构设计：通过对高校各信息系统数据的深入调研分析，利用开源组件或对开源组件的二次开发，搭建了高校大数据融合平台，并设计实现了简单、可扩展的分层高校大数据融合治理架构。

3) 数据清洗模型研究：组合扩展Kettle、Sqoop、Flume等开源ETL工具，创造性的开发了三阶段ETL模型，并针对高校大数据特点，制定了数据清洗标准规范，建立了高校大数据模型。

4) 数据存储方案研究：利用MongoDB、Hive、HDFS、Kafka、Hbase等开源NoSQL或者分布式文件系统搭建了支持实时计算及离线计算的海量数据存储方案。

5) 数据计算模型研究：设计实现了离线计算与实时计算融合的混合计算引擎，利用kylin实现即席查询，简化了模型建立及OLAP实现工作，提高了查询的效率。

6) 可视化系统研发：利用Java/spring MVC/echarts/D3.js/微服务等技术，从零开发了高校大数据治理融合可视化系统，该系统与大数据基础平台采用REST/JDBC方式通信，降低了平台间的耦合性。支持关系型数据源、文本CSV格式数据源、Spark、Hive数据源对接分析，实现了饼图、柱状图、雷达图、折线图、热点图、地图、关系图等多种数据图表。

本文研究形成高校大数据治理融合总体架构，系统总体架构主要包括：各类异构数据源、大数据治理融合层和大数据专题分析层。系统为高校各类数据研制接口，实现了对数据的采集，利用分布式文件系统对高校数据进行存储和管理，研究形成数据采集框架。深入分析了高校大数据特征，研究形成了高校大数据应用专题分析包，如财务分析、科研分析、舆情分析、教务分析、教学管理、资产管理、综合预警、学生画像、图书管理、战略决策、招生就业分析、助学分析等高校大数据专题分析。

通过学习大数据技术理论，调研Hadoop开源及商业大数据技术栈，深入分析各种高校信息化系统数据模型及相关实体，本文分析并设计了高校大数据治理融合的技术平台，用于解决高校数据孤岛问题，为高校信息系统提供数据整合及建模的方法；提供多源异构海量数据分布式存储、数据分析、数据挖掘及可视化探索的能力；通过高校大数据治理融合平台，发现数据背后的各种关联与趋势，为学校教务管理、科研分析、制定决策及教学改革提供分析工具及数据支持。

## 5.2 进一步研究工作

作者研发的高校大数据融合治理平台，较好的满足智慧校园工程下对高校数据实现信息的高效管理和共享，有效地解决高校部分信息化系统的信息孤岛问题。分层设计及REST/JDBC接口调用，具有很好的可扩展性，但作者认为还需要投入更多精力和时间对以下几个方面进行深入研究及相关改进：

可视化界面：可视化是大数据的最后一公里，目前实现的可视化界面参考了魔镜大数据产品，功能还比较简单、只能通过拖拽实现部分图表的展示。丰富图表、数据分析、权限管理、机器学习等功能可以进一步加强。

**硬件规模：**由于实验环境条件所限，现有的硬件资源及数据规模较小，尚未达到TB或PB级别，大数据技术的优势还未能够充分的体现出来，系统需要扩大所采集的数据量，针对更多的数据信息进行分析，进行更大数据规模的测试和验证测试结果。在多个高校环境中进行测试，验证分析结果的准确度。在多种测试环境中，对系统进行改进和优化，提高系统分析算法的精度。

**大数据融合范围：**现在很多高校都有自己的信息化系统用于日常的工作和日常管理，如何将高校大数据治理融合分析平台应用在高校的各个信息化系统，并与其他各高校的信息化系统所产生的数据整合，也是未来高校大数据治理融合分析平台发展的一个主要的方向。

**机器学习方面：**当前平台预测分析功能还处于较初级的阶段，需要引入更好的算法模型和更大范围的数据集对系统进行测试训练。改进高校业务中需要预测的功能的预测精确度。同时，可以考虑引进TensorFlow等高级神经网络模型。

**智慧校园方面：**随着物联网技术成熟、5G通信技术的全面实施，智慧城市、智慧校园等应用将进入全面发展的阶段，高校大数据治理融合平台需要进一步研究对传感器数据的采集能力的支持；高校数据模型应该由现在的“以人为本”的理念转向“以物为主”的观念上来，构建一个“万物互联”的数据分析模型。基于“数据驱动”的模式，使高校大数据治理融合平台更好的支持智慧校园。

# 参考文献

[1] 刘智慧，张泉灵 大数据技术研究综述[J], 浙江大学学报(工学版) 第48卷第6期2014年6月

[2] 桑庆兵 大数据在高校的应用与思考[J],南通纺织职业技术学院学报(综合版) 第13卷第2期2013年6月

[3] 查永军 大数据与高校院系治理[J] 中国电化教育 总第372期，2018.1

[4] 刘凤娟 大数据的教育应用研究综述[J] SDJYIS Vol.24 No.8 2014

[5] 吴 玮 国内外教育大数据的应用研究综述[J] 中国校外教育下旬刊 2016年2月

[6] 陶雪娇，胡晓峰，刘洋 大数据研究综述[J] 系统仿真学报 第25卷 增刊 2013年8月

[10] Victor Mayer - Schonberger，Kenneth Cukier．大数据时代［M］． 杭州: 浙江人民出版社，2013: 193－232．

[11] 刘智慧，张泉灵 大数据技术研究综述[J], 浙江大学学报(工学版) 第48卷第6期2014年6月

[12] 桑庆兵 大数据在高校的应用与思考[J],南通纺织职业技术学院学报(综合版) 第13卷第2期2013年6月

[13] 查永军 大数据与高校院系治理[J] 中国电化教育 总第372期，2018.1

[14] 刘凤娟 大数据的教育应用研究综述[J] SDJYIS Vol.24 No.8 2014

[15] 吴 玮 国内外教育大数据的应用研究综述[J] 中国校外教育下旬刊 2016年2月

[16] 陶雪娇，胡晓峰，刘洋 大数据研究综述[J] 系统仿真学报 第25卷 增刊 2013年8月

[17] 基于技术的高校学生行为分析系统研究与实现-北邮

[18] 国家中长期教育改革和发展规划纲要（2010-2020年）

[19] 国家教育事业发展第十二个五年规划

[20] 教育信息化十年发展规划2011-2020年.doc。

[21] 刘智慧，张泉灵 大数据技术研究综述[J], 浙江大学学报(工学版) 第48卷第6期2014年6月

[22] 桑庆兵 大数据在高校的应用与思考[J],南通纺织职业技术学院学报(综合版) 第13卷第2期2013年6月

[23] 查永军 大数据与高校院系治理[J] 中国电化教育 总第372期，2018.1

[24] 刘凤娟 大数据的教育应用研究综述[J] SDJYIS Vol.24 No.8 2014

[25] 吴 玮 国内外教育大数据的应用研究综述[J] 中国校外教育下旬刊 2016年2月

[26] 于长虹 智慧校园智慧服务和运维平台构建研究[J] 中国电化教育 总第343期2015.8

[27]王晓波 大数据促进教育变革与创新——专访中央电化教育馆王晓波副馆长[J].中小学信息技术教育,2013, (10):10-11.

[28] 构建高校大数据分析平台的探索

[29] 吴亚坤，郭海旭，王晓明 大数据技术研究综述[J] 辽宁大学学报自然科学版 第42卷 第3期 2015年

[30] 宫夏屹，李伯虎，柴旭东，谷牧 大数据平台技术综述[J] 系统仿真学报 第26卷第3期 2014年3月

[31] 李学龙, 龚海刚 大数据系统综述[J] 中国科学：信息科学 2015年 第45卷 第1期

[4.1.2官网] https://www.cloudera.com/documentation/enterprise/latest.html

[4.1.2安装参考] <https://www.cnblogs.com/pojishou/p/6267616.html>

# 致谢