学号 1362810213

年级 2013 级

河 海 大 學

本科毕业论文

时序大数据可视化分析平台研究与设计

专　　业 计算机科学与技术

姓　　名 陶友贤

企业导师 陈跃国

校内导师 牟　艳

评 阅 人

2017 年 6 月

中国　南京

BACHELOR'S DEGREE THESIS OF HOHAI UNIVERSITY

Research and Design of Visualization Analysis Platform on Large Time Series Data

College ： The Internet of things engineering

Subject ： Computer Science and Technology

Name ： Tao Youxian

Directed by ： Mu Yan Professor

NANJING CHINA

学术声明：

郑 重 声 明

本人呈交的毕业设计(论文)，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本设计（论文）的研究成果不包含他人享有著作权的内容。对本设计（论文）所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确的方式标明。本设计（论文）的知识产权归属于培养单位。

本人签名：_____　　日期：_____

河 海 大 学

本科毕业设计（论文）任务书

（理 工 科 类）

Ⅰ、毕业设计（论文）题目：

时序大数据可视化分析平台研究与设计

Ⅱ、毕业设计（论文）工作内容（从综合运用知识、研究方案的设计、研究方法和手段的运用、应用文献资料、数据分析处理、图纸质量、技术或观点创新等方面详细说明）：

本课题来源于广东省重大科技计划项目《高通量大数据实时商业智能系统产业化实现》。如今，可视化分析平台对大数据的处理分析越来越重要。针对时序大数据可视化分析的需求，分析出现有的可视化分析工具操作繁琐、重复、耗时，以及在时序大数据（即随着时间序列变化特征的数据）场景下存储和查询性能下降时的问题。为此，借助开源的 Zenvisage 可视化分析平台，提出了基于 Spark 的时序大数据分析方案，将其集成于 Zenvisage 的可视化分析平台底层中，最终通过系统测试和对比实验评估本文研究内容的有效性和可用性。

本设计根据时序大数据可视化分析的需求，在系统层面上实现对大规模视图的管理，支持用户修改查询条件，调整被分析的属性，进行交互式的可视化分析；根据绘制的可视化趋势，协助用户发现众多视图中趋势相似的，或者趋势明显不同的视图，同时推荐呈现当前正在查看的数据子集中最有趣的趋势，如代表性、异常性；提供用户数据集上传接口，并支持基于可视化集合的查询，从可视化中指定所需的洞察；同时，本时序大数据的可视化分析系统要求实时性强、使用方便，高效、功能完整，有较强的可交互性、可复用性、可扩展性，系统应用数据挖掘、HDFS 分布式文件系统、Parquet 列存储、Spark 查询和 Dygraphs 展示等多种技术开发。

要求设计和实现使用面向对象的软件开发技术，并结合相关设计模式，运用

UML 等相关建模语言建模，完成需求分析，需求建模，软件设计，编码实现，测试等工作，并产生相应文档、数据及程序代码等。通过本设计的实现，将软件开发技术理论的研究与软件开发实践相结合，在实践中培养学习能力、分析和解决问题能力以及独立工作能力。

设计工作完成后，按要求撰写毕业论文，进行毕业答辩。

Ⅲ、进度安排：

2015.12-2016.1 熟悉项目需求、熟悉开发环境、确定数据的需求

2016.2 对系统进行总体设计、确定模块及模块功能、完成数据库设计、对系统进行整体构架

2016.3-2016.4 进行系统的详细设计、完成编码

2016.5 进行测试与调试并进行修改完善

2016.6 资料整理、撰写毕业论文、进行毕业设计答辩

Ⅳ、主要参考资料：

[1] 喻宜，吕志来，齐国印. 分布式海量时序数据管理平台研究[J]. 电力系统保护与控制, 2016, 44(17):165-170.

[2] 杜小勇，陈峻，陈跃国. 大数据探索式搜索研究[J]. 通信学报，2015, 36(12):77-88.

[3] 王囝囝，杨树，毕焘. 大数据时代数据信息可视化的研究[J]. 通讯世界, 2015(14):185-186.

[4] 何贤国. 出租车 GPS 大数据可视化研究[D]. 浙江工业大学, 2014.

[5] Siddiqui T, Kim A, Lee J, et al. Effortless data exploration with zenvisage: an expressive and interactive visual analytics system[J]. Proceedings of the Vldb Endowment, 2016, 10(4):457-468.

[6] Sun Y, Qi J, Zhang R, et al. MapReduce based location selection algorithm for utility maximization with capacity constraints[J]. Computing, 2015, 97(4):403-423.

[7] Liu C, Zhang D, Chen Y. Personalized Knowledge Visualization in

Twitter[M]// Conceptual Modeling. 2015.

[8] Chen J, Chen Y, Du X, et al. SEED: A system for entity exploration and debugging in large-scale knowledge graphs[C]// IEEE, International Conference on Data Engineering. 2016:1350-1353.

[9] Zheng Z, Chen J, Lyu M R. Personalized Web Service Recommendation via Normal Recovery Collaborative Filtering[J]. IEEE Transactions on Services Computing, 2013, 6(4):573-579.

[10] Chen Y, Qin X, Bian H, et al. A Study of SQL-on-Hadoop Systems[M]// Big Data Benchmarks, Performance Optimization, and Emerging Hardware. Springer International Publishing, 2014:154-166.

[11] Parameswaran A, Polyzotis N, Garcia-Molina H. SeeDB: visualizing database queries efficiently[M]. VLDB Endowment, 2013.

指导教师：牟艳，陈跃国 ， 2016 年 12 月 日

学生姓名： 陶友贤 ，专业年级：计算机科学与技术 13 级

系负责人签字： ， 2016 年 12 月 日

摘要

可视化分析平台对大数据的处理分析越来越重要。本文针对时序大数据可视化分析的需求，分析出现有的可视化分析工具操作繁琐、重复、耗时，以及在时序大数据（即随着时间序列变化特征的数据）场景下存储和查询性能下降时的问题。为此，借助开源的 Zenvisage 可视化分析平台，提出了基于 Spark 的时序大数据分析方案，将其集成于 Zenvisage 的可视化分析平台底层中，解决了上述问题。论文主要从以下几个方面展开讨论：

（1）对大规模时序数据进行可视化需求的分析，在系统层面上实现对大规模视图的管理，支持用户修改查询条件，调整被分析的属性，进行交互式的可视化分析；根据绘制的可视化趋势，协助用户发现众多视图中趋势相似的，或者趋势明显不同的视图，同时推荐呈现当前正在查看的数据子集中最有趣的趋势，如代表性、异常性；提供用户数据集上传接口，并支持基于可视化集合的查询，从可视化中指定所需的洞察；

（2）分析了 Zenvisage 平台在大规模时序数据的应用场景下存储和查询性能下降的问题，找到传统关系型数据库在时序大数据分析中的瓶颈；

（3)针对这些瓶颈提出了基于 Spark 的时序大数据分析方案,该方案以 HDFS 进行文件存储、Parquet 列存储数据压缩、分布式内存引擎 Spark 为底层计算框架，以解决 Zenvisage 平台在时序大数据场景下的有效性，提高交互式分析系统的性能和运行效率；

（4）将此分析方案集成于 Zenvisage 的可视化分析平台底层中，并通过系统测试和对比实验评估本文研究内容的有效性和可用性。

本课题主要服务于广东省重大科技计划项目“高通量大数据实时商业智能系统产业化实现”，经测试证明，该系统能够满足用户交互式分析的需求，优化了 Zenvisage 平台在时序大数据场景下的存储和查询性能，运行稳定、功能完整、可扩展性强。

关键词：时序大数据；交互式分析；可视化；Zenvisage；HDFS；列存储；Spark

ABSTRACT

Visual analysis platform for the process and analysis on large data is becoming increasingly important. This paper analyzes the demand for visualization analysis of large time series data(ie, data that changes with time series), and some visual analysis tools sounds cumbersome, repetitive, time-consuming, and time-consuming large data (ie, data with time series variation) problem. For this, with the open source Zenvisage visual analysis platform, we propose a large data analysis scheme based on Spark, which is integrated into the bottom of Zenvisage's visual analysis platform, and solves the above problems. The paper mainly discusses the following aspects:

(1) Analying the visualization requirements of large time series data, the system needs to manage amounts of views, support the user to modify the query conditions, adjust the analysis of the property, make interactive visual analysis; according to the visualization of the trend to help users find the views that are similar in trend, or show a significantly different trend, and recommend presenting the most interesting trends in the subset of data currently being viewed, such as representative visualizations, outliers, providing user with data uploading interfaces, and supporting visualization based collections query, specify the required insight from the visualization.

(2) Analying the Zenvisage platform in the large time data application scenarios storage and query performance degradation, find the bottleneck of traditional relational database in the analysis of large data in timing.

(3) According to these bottlenecks, a large data analysis scheme based on Spark is proposed. The program is stored in HDFS for file storage, Parquet columns store data compression, distributed memory engine Spark as the underlying computing framework to solve effectiveness of the Zenvisage platform in the large time series data scenarios, and improve the effectiveness of interactive analysis system performance and operational efficiency.

(4) This analysis scheme is integrated into the bottom of Zenvisage's visual analysis platform, and the effectiveness and availability of the research contents are

evaluated by systematic testing and comparative experiments.

This project mainly serves the major technology project of Guangdong Province, "high-throughput large-scale real-time business intelligence system to achieve industrialization". It proves that the system can meet the users' needs of interactive analysis, optimize the storage and query performance of Zenvisage platform in the scene of large time series data, and it is stable, functional and extensible.

Key words: Large Time Series Data; Interactive Analysis; Visualization; Zenvisage; HDFS; Column-store; Spark

目 录

图表目录

第1章　绪论

本章主要阐述了大数据下的开源技术和可视化分析的研究背景及意义，重点介绍现有相关研究工作存在的不足，从而总结本课题目前在设计和实现上所面临的难点和挑战，进而提出本文的主要研究内容以及解决目前难点和挑战的方案，最后给出全文的主要贡献和组织结构。

1.1 选题背景和意义

云计算、移动互联网、智慧城市与人工智能已经成为大数据时代下的高新技术，随着人、实物与机器的交叉互联，行为数据、风控数据、日志数据、社会数据等呈指数型增长，大数据时代已全方位到来。随着数据量的不断增大，大数据平台对海量数据的处理分析愈发重要。在大规模时序数据基础上，这里的时序数据指的是随时间序列变化的数据集，借助已有开源领域先进的可视化分析技术通常是从数据集中探索和提取洞察最常用的机制。

1.1.1 交互式可视化分析工具的使用及现况

如今的交互式可视化分析工具，如 Tableau[1]、Spotfire[2]，他们的出现给数据挖掘和数据科学的民主化铺平了道路，并被广泛使用。据调查，去年 Tableau 的收入就在几亿美元，今年预计将达数百亿美元[3]。通过这些工具，标准的数据分析方法一般遵循以下操作：数据分析师首先选择可视化分析工具，再将数据集加载到工具中，通过可视化来检查，分析结果，然后重复该过程，直到找到满足期望的视图。此外，关系数据库一般作为可视化分析系统的计算层，但是也可以通过 SQL 来支持交互式分析，查询语言可以很快地帮助用户得到数据结果，而且数据信息很直观等。

以上现有的交互式分析工具普遍是基于数据层面进行设计和实现的，表面上看是“一劳永逸”、直观的分析方案，充分利用了用户自己的偏好设置，支持用户修改查询条件，却忽略了在充分提高用户体验时加重了用户的检查、试错工作。为了在数据集中找到所需的期望，数据人员可能需要密集型地手工检查集合中的每个可视化，这是一个繁琐、重复、耗时的循环过程[4]。同时，在手工检验下，

像 Tableau 这样的可视化工具只能一次生成并提供一个可视化，适度的数据集也会产生数百以上个可视化，这给数据人员也会造成严重的分析障碍。此外，对于一些复杂的数据探索，如：在单个数据集中找到给定属性的上升趋势的视图，这很难通过单个 SQL 的编写实现，可能需要大量的自定义函数，一定程度上对分析人员的编程能力要求也很高。

针对重复耗时的手工试验和相对复杂的查询这两个对数据本身进行操作时存在的问题，Zenvisage 平台在此背景下应运而生。该平台设计并实现基于可视化集合操作的代数（ZQL），以预定义的方式处理可视化集合，通过对可视化集合进行常见的一些操作，如组合、基于条件过滤、比较可视化、基于条件排序等，从而自动识别相关条件下的可视化期望，一定程度上实现了轻松的数据探索过程[5]。其中，相关条件指的是与特定模式的相似性或不相似性、典型或者异常行为等。

然而，在面临高维且数据量大的数据集时，Zenvisage 平台针对行存储关系型数据库的查询性能有待提高，识别显示相关或期望的趋势视图将会很耗时。大数据时代下如此海量的数据也为交互式数据探索、分析、数据整合带来了巨大的挑战，一个好的存储策略和处理数据的效率就是大数据时代下进行数据分析的生命。

1.1.2 大数据技术在海量数据方面的应用现状

Hadoop 的诞生对于大数据而言具有跨时代的意义。Hadoop 的核心主要是“三驾马车”：HDFS、MapReduce 和 Yarn。HDFS 可用于海量数据存储，是一个具有高容错、高可用、高吞吐量特点的分布式文件系统；MapReduce 可用于数据计算，是针对大数据的分布式计算引擎；Yarn 在 Hadoop 应用中主要负责资源管理和作业调度，从而大大提高了集群利用率和数据共享[6]。

SQL（Structured Query Language，结构化语言）是传统关系型数据库中的查询和程序设计语言，为方便数据人员直接操作 HDFS 中的数据，基于 HDFS 的 SQL 查询系统（SQL on Hadoop）遍应运而生。现在普遍使用的 SQL on Hadoop 系统主要是 Spark 体系中的 Spark SQL、Hadoop 体系中原生的 Hive 和适用于交互式分析的 Presto 等。SQL on Hadoop 系统主要面向两种分析需求：一种是复杂

的分析性查询[7]，另一种是海量数据上的交互式查询。前者要求考虑查询的可扩展性，支持 TB 级上的数据查询；后者特点是是查询相对简单、并发量大，要求数据处理速度较快。

当下大数据主要面向两种存储方案：行存储和列存储。学术界和工业界在开源系统实现中，为兼顾安全、可靠性等而选择列式存储引擎，如对嵌套结构较好的 Parquet[8]和对 ACID 支持较好的 ORC（Optimized Row Columnar）[9]。两者均采用双层压缩模式，第一步是采用特定的编码格式游程、字典、增量、Bit 等进行轻量级压缩，再进一步对编码后的数据使用 zlib、snappy、LZO 等压缩技术进行行压缩[9]。

现有的 Hadoop 体系中的并行处理框架 M-R 因其架构设计自带的弊端导致大量网络开销和 I/O，现已经逐渐被基于内存计算的分布式引擎 Spark 等计算框架取代。但考虑到 HDFS 分布式文件系统的高可用、高容错特点，在学术界和工业界的大数据存储中仍具有其无法逾越的地位；同时考虑到相比于传统的行式存储引擎，列式存储引擎优秀的压缩性能，更少的 I/O 操作，在面临 GB 甚至 ZB 级数据量挑战时 SQL on Hadoop 系统建议使用列存储作为 HDFS 上的存储方式，尤其在数据集中列属性很多，且每次操作仅针对少许列的场景下，此时列式存储引擎的性价比会更高，查询效率愈发明显[9]。

针对现有的 Zenvisage 平台在时间序列数据（即时序数据）可视化分析处理上的优势，但其在时序大数据的应用场景下存储和查询策略的这两个问题，本方案在面临海量数据可视化分析时设计并实现基于 Spark 的时序大数据分析方案，即通过 HDFS 分布式文件系统实现数据存储以及 Spark SQL 内存计算引擎的查询方案，并根据不同属性数据集设计出适应性读取存储策略，同时通过列存储技术进一步提高存储和查询性能，有效地满足了不同数据集分析时系统底层高吞吐、高并发的查询分析，以达到稳定集成于 Zenvisage 平台的底层设计的目的。

1.2 课题的难点及挑战

关于本课题的研究内容，主要包括一个核心难点和三大挑战，分别介绍如下：

（1）针对 Zenvisage 的应用场景找到传统关系型数据库在大数据分析中的瓶颈，设计适合的时序大数据分析方案是本文的核心难点。

Zenvisage 平台将传统关系数据库用作计算层，且表属性明确，其应用场景都是基于时间序列的数据集（即时序数据）。如此的行存储方式和 SQL 查询在面临大规模数据时，系统的性能和运行效率值得商榷，且造成的 I/O 操作、网络带宽较大。如何针对 Zenvisage 在时序大数据场景下的瓶颈，结合当下已有开源领域先进的存储和查询技术，设计出适合的时序大数据分析方案，且可以扩展性地满足不同规模、多属性等条件下的数据集是本文的核心难点。

（2）针对这些瓶颈设计基于 Spark 的时序大数据分析方案。

尽管现有的开源大数据技术十分火热，但根据上述瓶颈设计出基于 Spark 的时序大数据分析方案，包括在海量数据存储方面利用 HDFS 分布式文件系统和开源列存储引擎 Parquet，以及面对查询时应用 SQL on Hadoop 系统中的 Spark SQL，在技术实现、环境配置调试过程中面临了许多挑战。如对不同数据集需要根据其属性字段设计出相应数据模型的挑战、Parquet 列存储的挑战、Spark SQL 查询时对 Parquet 存储文件和缓存文件信息的读取流程重新设计和实现等难题和挑战。

（3）将设计出的基于 Spark 的时序大数据分析方案集成于 Zenvisage 的可视化分析系统底层中。

在将本文设计出的基于 Spark 的时序大数据分析方案集成于 Zenvisage 的可视化分析系统底层中，两者相辅相成解决了时序大数据可视化分析下的很多问题，但在底层实现过程中也面临了诸多挑战，比如对 Zenvisage 平台中 ZQL 模型理解的挑战、底层查询接口与 Spark SQL 查询接口对接的挑战等。

（4）通过对比实验验证本文设计出的基于 Spark 的时序大数据分析方案的可用性和有效性。

通过对比实验验证本文研究方案在时序大数据场景下存储和查询性能的提升，在实验设计上面临了很多挑战，比如集群配置中的软硬件资源、SQL 查询负载设计以及对比方案条件设置等。

1.3 本文的主要研究内容和贡献

本文的研究内容和贡献主要包括几个方面：针对 Zenvisage 的应用场景找到传统关系型数据库在时序大数据分析中的瓶颈；研究、设计并实现了基于 Spark

的时序大数据分析方案，并将其集成于海量数据的存储和查询；将设计出的基于Spark 的时序大数据分析方案集成于 Zenvisage 的可视化分析系统底层中。具体来说：

（1）本文首先从技术方法这一角度研究现有的可视化分析相关工作，发现在基于数据层面上进行可视化分析一直是一个繁琐、重复、耗时的循环过程，并介绍通过 Zenvisage 这一平台充分利用用户对视图结果的几种特定期望，解决了手动密集型操作、一次显示一个可视化的问题，但其在数据信息存储和查询上还在采用行存储的传统关系型数据库，达不到海量时序数据分析场景下性能和运行的需求。

（2）在此基础上，本文提出基于 Spark 的时序大数据分析这一方案，即通过 HDFS 列存储技术、Spark SQL 查询性能实现海量时序数据存储和查询的策略。该方案具体包括根据不同数据集设计合适的数据模型，在系统已有数据集或用户上传数据集时实现在 HDFS 文件系统上进行 Parquet 列存储，并实现业务上的 Spark SQL 查询接口；同时，将该方案集成于 Zenvisage 的可视化分析系统底层中，理解 Zenvisage 平台的所有业务逻辑，并将底层查询接口与 Spark SQL 查询接口对接，并在应用程序基础上实现查询结果缓存等优化功能。

1.4 本文的组织结构

本文的主要内容可以分为六个章节，各个章节的结构与内容如下：

第一章是绪论，主要介绍了本文的研究背景及意义。本章分析了本课题的现有研究方案的不足和局限性，以此说明了本文的研究方案是在 Zenvisage 平台基础上提出的亟待解决的问题，从而分析了本课题目前在设计和实现上所面临的难点和挑战。此外，对本文的主要研究内容和贡献进行了总结。

第二章为国内外研究现状和相关工作介绍。本章首先阐述了可视化分析技术的一些理论理论方面的知识基础，接着具体介绍了 Zenvisage 平台的主要特点、ZQL 结构、操作简介和视化分析处理步骤等；之后，主要对系统的相关技术工作和采用的架构进行了简单说明。最后本章指出了针对 Zenvisage 等可视化分析平台在大数据处理和可视化分析方案上现有工作的不足，提出了本文在可视化分析研究工作的方向。

第三章为基于 Spark 的时序大数据分析方案。本章是本文的核心之处，重点阐述针对 Zenvisage 的优化设计，是在 Zenvisage 平台基础上针对海量数据存储、计算的策略，详细介绍了本系统底层的列存储和 Spark SQL 查询策略，以及其中的查询结果缓存优化和对 Zenvisage 的相关改进部分。

第四章为时序大数据的可视化分析系统实现。本章在前文 Zenvisage 平台和本文核心内容的提出上进行系统的实现，首先是对开发工具和开发环境做了简单介绍，提出了系统的设计原则。然后，详细介绍了系统总体结构设计和软件结构设计，最后对数据集设计了数据模型设计，并对系统运行成果图进行了展示。

第五章为实验结果分析。本章主要通过设计实验来证明本文研究并实现的可视化分析系统的有效性和实用性。

第六章为总结与与下一步工作展望。本章总结了本文所有的工作，并对本课题的其他不足之处提出了改进方案。

参考文献，列举完成课题设计所参照的期刊、技术论文等。

致谢，对完成本课题“时序大数据可视化分析平台研究与设计”工作提供帮助和一直以来支持笔者、培养笔者的所有人员表示感谢。

附录，翻译对课题设计与实现过程中影响颇大的外文文献。

第2章　国内外研究现状和相关工作介绍

本章首先介绍了可视化分析技术的相关知识，为本文提供理论方面的知识基础；然后，具体介绍了 Zenvisage 平台的主要特点、ZQL 结构、操作简介和视化分析处理步骤等；之后，就本文系统应用的一些技术进行了简单说明。最后本章指出了针对 Zenvisage 等可视化分析平台在大数据处理和可视化分析方案上现有工作的不足，提出了本文在可视化分析研究工作的方向。

2.1 可视化分析技术概述

当前，随着大数据时代的发展，大数据平台对海量数据的处理分析愈发重要。在大规模时序数据基础上，进行数据可视化分析是迄今为止最常用的机制，可以从大数据集中探索和提取洞察，特别是对于数据分析人员来说尤为重要。

接下来，主要从概念、分析步骤和相关工作介绍三个方面展开对可视化分析技术概述的介绍。

2.1.1 可视化分析的概念

可视化分析，主要集成于海量数据的处理、可视化与分析，可辅助人工操作在数据中进行交互式探索和提取洞察，并做出期望的完整的可视化视图或分析图表[11]。它作为对数据分析重要的工具，所涉及到的数据集内容一般较分散、数据结构或字段名称可能不统一，不易形成固定的分析模式或操作流程，这是对于不同数据集源进行统一分析的一个难点之处。

2.1.2 可视化分析的步骤

一般的，可视化分析需要借助功能强大、操作方便的可视化数据交互式分析平台，如 Tableau、Spotfire 等，进一步辅助人工进行数据分析、视图生成，其步骤示意图大体为：

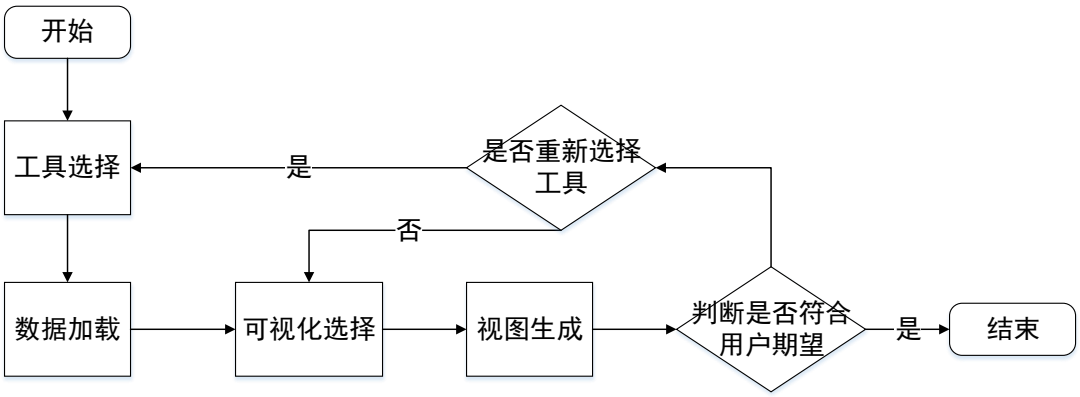


图 2-1 可视化分析步骤示意图

从该示意图中可以看出，在对数据集进行可视化分析操作时首先需要选择一个可视化分析工具或平台，在显示的视图不满足用户期望下会重新识别其他可视化视图或者重新切换依赖的工具，这样就导致用户手动密集型的操作，使得整个可视化分析过程繁琐、复杂、耗时且可能重复。

2.1.3 可视化分析相关工作介绍

一般来说，对于可视化分析需要依赖一些分析工具、平台，主要包括一些工具的用途、与可视化视图相关领域的工作创新等。下面对这些内容进行分类解释介绍：

1）可视化工具。可视化工具包括 Spotfire 和 Tableau 等商业智能软件，方便用户构建具有不同复杂程度的可视化，类似的工具也有 ShowMe[12]、Fusion Tables[13]等等被数据库社区引入，其中一些工具提供了使用启发式来为给定的一组变量建议图表类型的功能。然而，所有这些工具要求用户手动指定可视化，导致繁琐的尝试和错误过程来查找有趣或有用的可视化；同时一次操作一个可视化，用户也无法直接识别所需的模式或需求。

2）推荐可视化。可视化推荐工具主要是 Voyager[14]、SeeDB [错误!未找到引用源。]、Zenvisage 等代表性系统，主要是根据数据差异性给用户自动地识别并推荐一些“有趣性”视图。Voyager 根据可视化的美学特性推荐可视化，而不是查询；SeeDB 建议最佳显示两组数据之间的差异的可视化。Zenvisage 根据交互性、表达性来设计更好地体现了可视化分析系统的优点，且提出的 ZQL 模型更直接对可视化集合进行操作，轻松地从数据集中找到所需的视觉模式。其中，SeeDB 和 Zenvisage

中设计的方案是基于偏差的效用度量模型，且底层的优化策略具有一定的创新价值。

3）可扩展的可视化。为了使得可视化分析尽可能实现可扩展性，主要是提升分析时的高效性，采用内存中缓存，采样和预取，以提高数据库支持的可视化系统的交互性。这样的技术可以用于系统底层设置时以进一步改善响应时间（尽管这些技术中的某些技术，例如内存中缓存，只能用于小数据集）。

2.2 Zenvisage 平台介绍

如 1.1.1 节中所述，在“大数据”时代，每个用户都获得了大量的数据信息，并且正在努力去理解这些数据，并从这些数据中获得价值。非程序员使用的常用办法是将此数据加载到可视化工具中，并重复生成可视化，直到所需的期望被识别为止。这种探索是一个痛苦、乏味和耗时的过程，同时也意味着“单位时间的洞察力”非常低。针对重复耗时的手工试验和相对复杂的查询这两个对数据本身进行操作时存在的问题，Zenvisage 平台在此背景下应运而生。

接下来，主要从平台简介、主要特点、ZQL 结构、操作简介和视化分析处理步骤五大部分来对 Zenvisage 平台进行叙述说明。

2.2.1 Zenvisage 平台简介

Zenvisage 是一个可视化探索系统，基于 SeeDB 并进行了扩展优化，可以自动识别和推荐有趣的可视化，并协助用户发现众多视图中趋势相似的，或者趋势明显不同的视图。值得一提的是，用户也可以自定义所需的复杂的可视化视图，即 Zenvisage Query Language（ZQL），系统将为其进行其他操作，很快速得呈现相关的可视化。

2.2.2 Zenvisage 平台的主要特点

1）交互式视图画板界面，简单易用

在整个可视化分析中，只需要简单的画、拖、拉等操作，就能够完成趋势图的构建，而不需要编码等复杂过程。通过交互式视图面板界面，用户可以直接绘制他们正在寻找的趋势图，然后依靠系统找到适当的匹配：例如，浏览材料属性

数据集的人可能正在寻找显示两个属性之间的具体相关性。用户还可以将趋势图拖放到画布上，然后修改趋势，使用此界面，用户可以指定他们正在寻找的洞察，并期望 Zenvisage 能够找到匹配，就像“可视化搜索引擎”一样。

2）复杂的可视化查询界面，简洁直观

对于更复杂的请求，Zenvisage 支持一种称为 ZQL 的查询语言，全称为 Zenvisage Query Language，这是一种直接操作可视化集合的灵活直观的机制，用于从可视化中指定所需的洞察。使用少量的 ZQL 行，用户可以以任何方式探索期望的趋势、模式和洞察。

3）可视化推荐

除了为用户提交的查询返回相关匹配的结果之外，Zenvisage 还运行大量并行查询，以查找用户当前正在查看的数据子集中最有趣的趋势，如代表性、异常性，并将其作为可视化推荐呈现。

2.2.3 Zenvisage Query Language 的结构概述

在当今主流的可视化应用分析中，柱状图、条形图、散点图等被应用相当广泛，如下图 2-2 中展示了一个常见可视化样例的几大组件：（1）X 轴属性(year)；（2）Y 轴属性(sales)；（3）使用的数据子集(product chair)；（4）可视化类型(bar)；（5）Y-X 聚合函数（the sum of sales）[16]。

图 2-2 Product Chair 年销售量条形图

由上通过分析类比可以得到 Zenvisage Query Language（ZQL）的可视化组件部分（Visual Component）：标识符（Name）、X、Y、Z、Viz，其中 Viz 代表上文中的（4）和（5）两大组件，其可视化相应表示如下表 2-1。

表 2-1 Product Chair 年销售量 ZQL 结构表示

标识符	X 轴	Y 轴	Z 轴	Viz
*f1	‘year’	‘sales’	‘product’.’chair’	bar.(y=agg(‘sum’))

为了满足实际中更复杂的可视化查询，ZQL 主要由可视化组件和任务组件组成，其结构如下表 2-2 所示。其中可视化组件除了以上部分外，还包括约束列（Constraints Column），任务组件（Task Component）由处理列（Progress Column）组成。

表 2-2 ZQL 结构表示

Name	X	Y	Z	Viz	Constraints	Process
标识符	可视化组件					任务组件

2.2.4 Zenvisage 平台的操作简介

Zenvisage 提供了简洁直观的用户操作界面，如下图 2-1 主界面（a）和图 2-2 主界面（b）所示，主要分为六大模块：属性选择窗口、可视化推荐区、画布面板窗口、结果匹配区、ZQL 高级探索区、配置参数窗口。

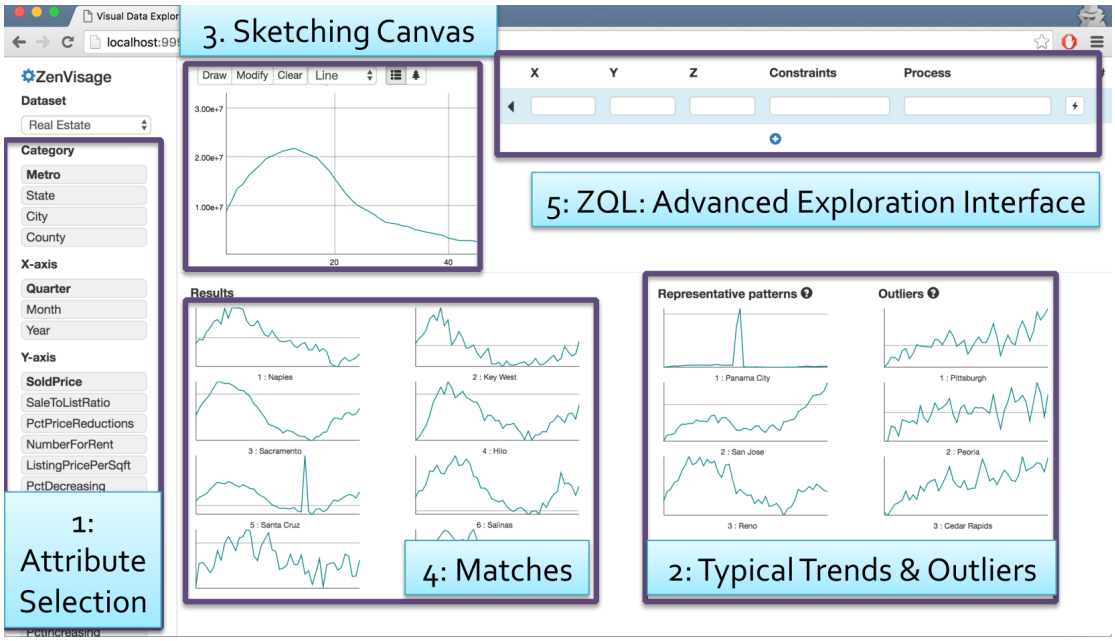


图 2-3 Zenvisage 操作主界面（a）

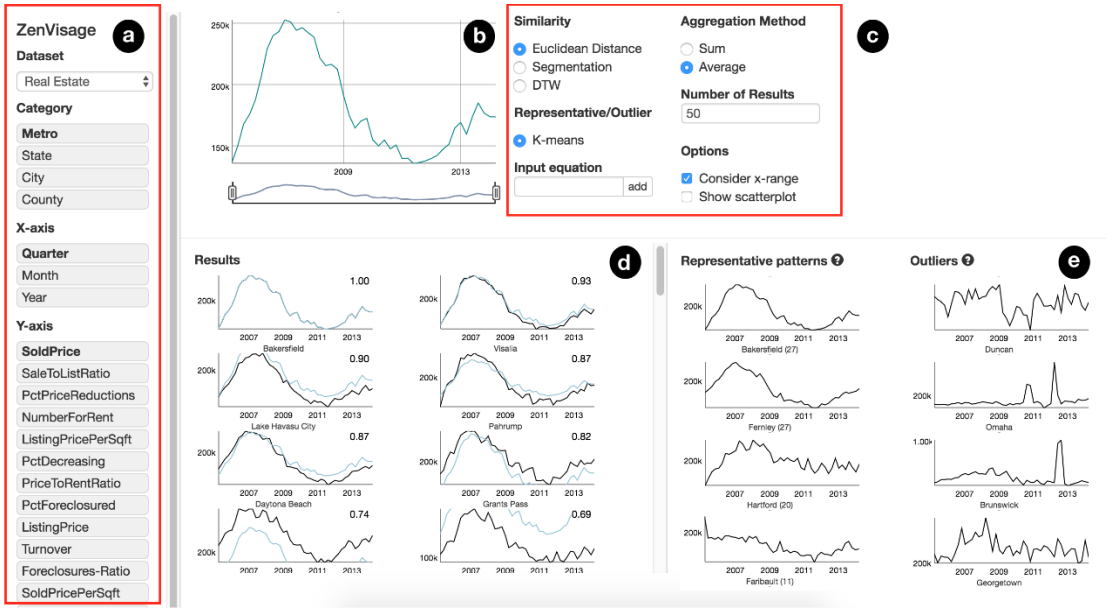


图 2-4 Zenvisage 操作主界面（b）

下面对图 2-3 和图 2-4 中各模块的功能进行解释：

1）属性选择窗口：图 2-2 的“Attribute Selection”区域，用于显示某数据集的属性，便于选择；

2）可视化推荐区：图 2-2 的“Typical Trends & Outliers”区域，用于推荐有趣的可视化，如匹配区中的代表性视图等；

3）画布面板窗口：图 2-2 的“Sketching Canvas”区域，用于自定义可视化趋势、拉伸时间轴；

4）结果匹配区：图 2-2 的“Matches”区域，用于查询条件下的可视化显示，并且根据画布面板中的可视化进行相关性判断；

5）ZQL 高级探索区：图 2-2 的“Advanced Exploration Interface”区域，用于复杂的查询需求，可以通过 ZQL 高级探索区进行设置、提交等操作；

6）配置参数窗口：图 2-3 的“c”区域，用于系统的相关参数配置，如结果显示数目、聚合方法等。

2.2.5 Zenvisage 平台的可视化分析处理步骤

通过 2.2.3 中关于 Zenvisage 平台的操作介绍，在可视化分析处理过程中，用户在构建一个分析方案的过程如下：

1）从数据集源中选择想要进行数据探索的数据集选项；

2）从数据集选项下方的属性选择窗口选择 Category、X 轴、Y 轴属性；

3）根据以上设置，结果匹配区中就会呈现与画布面板窗口的初始化趋势相匹配的可视化，在可视化推荐区也会显示有趣的可视化；

4）根据实际需求，用户也可以进行自己的偏好操作：

（A）在配置参数窗口设置相关参数，如相似性视图算法、聚合方法等；

（B）在画布面板窗口构造自己期望的视图趋势，在结果匹配区呈现出相关的可视化，在可视化推荐区显示有趣的可视化；

（C）用户可以拖拽匹配、推荐区中的可视化至画布面板窗口，或者在拖拽的可视化上进行修改，系统将进行同（B）的操作；

（D）用户如果将进行更加复杂的查询需求，可以通过 ZQL 高级探索区进行设置、提交等操作。

2.3 HDFS 列存储技术

HDFS 文件系统（Hadoop Hadoop Distributed Filesystem）主要为 Apache Hadoop 项目提供底层的数据存储策略，满足项目上层的各种实际需求,属于典型的的“Master/Slave”集群架构，其基本的架构如下图 2-5 所示。

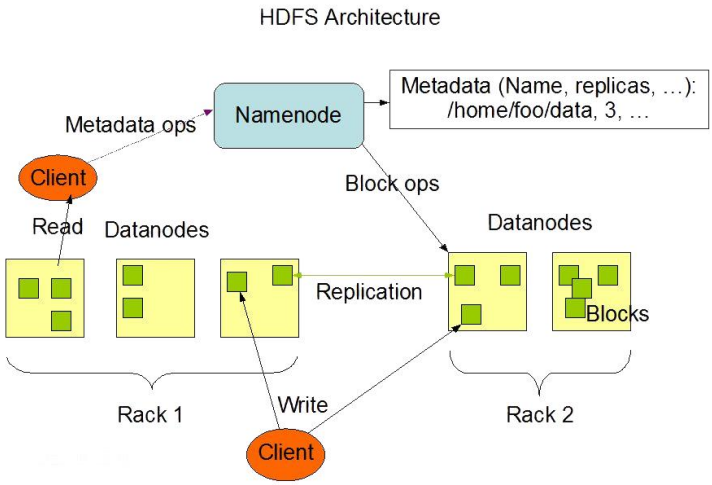


图 2-5 HDFS 文件系统架构图

从该架构图可以看出，HDFS 文件系统由一个 NameNode 和多个 DateNode 组成，只能有一个 NameName，担任 Master 角色，主要是为存储块的元数据提供保存功能；可以有多个 DataNode，担任着 Slave 的角色，单个存储块根据集群环境的配置可以保存在多个 DataNode 副本上，这样的目的是为了保持数据的高可

用性[17]。

如前所述，列式存储引擎具有更加优秀的压缩性能，更少的 I/O 操作，因此在面临 GB 甚至 ZB 级数据量挑战时 SQL on Hadoop 系统建议使用列存储作为 HDFS 上的存储方式，尤其在数据集中列属性很多，且每次操作仅针对少许列的场景下，此时列式存储引擎的性价比会更高，查询效率愈发明显。目前在开源系统中使用度最高的列式存储引擎分别是对嵌套式结构支持较好的 Parquet 和对 ACID 支持较好的 ORC。两者均采用双层压缩模式，第一步是采用特定的编码格式游程、字典、增量、Bit 等进行轻量级压缩，再进一步对编码后的数据使用 zlib、snappy、LZO 等压缩技术进行行压缩。

2.4 SQL on Hadoop 系统

SQL on HadoopSQL 系统是指对存储于 Hadoop 中 HDFS 上的数据进行 OLAP 分析的系统。SQL on Hadoop 近年来发展迅速，并涌现出许多优秀的系统，在这其中比较有代表性的，本文按照分析需求的特点分为两类：

一种是面向复杂的分析性查询的 SQL on Hadoop 系统，分析性查询常需要扫描大量数据，涉及的查询通常也比较复杂，查询所处理的时间长，此种查询在大数据背景下主要考虑的是系统可扩展性，可支持 TB 级以上的数据查询，而对数据查询的响应时间要求不高。此类系统比较有代表性的主要 Hive、Spark SQL 等。

Hive 是 Facebook 推出的 Hadoop 上的类 SQL 查询系统，其最早使用的计算引擎为 Hadoop 体系中 MapReduce，即将 HiveQL 查询经 SQL 解析、逻辑优化并产生物理执行计划后转为 MapReduce 作业，由 MapReduce 读取数据和分析最终产生结果返回。但 MapReduce 执行引擎因其性能不佳已逐渐被淘汰，原因包括执过程中产生的中间结果会持久化到磁盘带来大量磁盘 I/O 开销、在并行计算的各个阶段不能进行有效的数据共享而带来额外的磁盘 I/O 开销和网络、执行策略不支持基于统计数据的优化、未使用基于 Directed Acyclic Graph（DAG）的任务调度机制而带来的磁盘 I/O 开销、每个任务启动一个进程而不是线程导致任务启动和调度额外开销等。由于这些原因从 2.0 版本 Hive 开始相继提供 Apache Tez、Spark Core 等计算引擎[18]。其中 Tez 是一种基于 DAG 的分布式计算引擎，它将提交的 Map/Reduce 计算过程拆分成若干个子过程进行计算处理，并将多个

Map/Reduceap 处理任务组合成一个较大的 DAG 处理任务，从而减少了中间结果本地化到磁盘带来的大量 I/O 开销，同时对这部分子过程进行合理组合，在一定程度上有效地减少了任务的总体运行时间。

Spark SQL 美国加州大学伯克利研发的基于 Spark 计算引擎的分布式 SQL 引擎，是 Spark 软件栈中的核心组成部分。Spark 计算引擎 Spark Core 是一个基于内存的分布式计算引擎，在某种程度上是对 MapReduce 模型的一种扩展，通过 RDD 的设计解决了 MapReduce 无法在并行计算的各个阶段进有效数据共无法的问题。Spark Core 利用 DAG 进行调度规划，可在多任务计算以及迭代计算中大量减少磁盘 I/O 时间，且 Spark Core 对于每一个任务启动的是线程而不是一个进程，相较于 MapReduce 而言大大缩短了任务启动时间[19]。

另一种则是面向大规模数据上交互式查询的 SQL on Hadoop 系统，此类查询具有查询条件相对简单、查询并发量大等特点，且响应时间不能超过用户的可容忍限度，对数据处理速度要求较高。此类系统比较有代表性的主要有 Presto 等。

Presto 是 Facebook 开发的一个基于内存适用于交互式查询的分布式 SQL 查询引擎。其设计和实现目标是解决 Facebook 这种大规模的商业数据仓库交互式分析和处理速度的问题，数据量支持 GB 甚至 ZB 级[20]。Presto 也是基于内存进行并行计算，且使用基于 DAG 的任务调度机制。相较于 Spark SQL 而言 Presto 是通过牺牲容错实现低延时查询，同样是基于内存计算 Spark 为了考虑容错会在查询过程中将部分间结果物化到磁盘中，即 Spark 的 checkpoint 机制，且 Shuffle 过程中在内存存不下的情况 Spark 会将数据 spill 到磁盘中以支持查询的继续执行，但 Presto 是完全没有中间结果的存储而是完全基于内存进行计算，不考虑任何的容错机制，且 Shuffle 过程中内存放不下的情况也会将数据 pill 到磁盘，而是直接反馈给用户查询失败，即 Presto 仅支持内存容量查询情况下的，执行过程中任何一个阶段内存中放不下，则会查询失败。Presto 这样做的益处在于内存容量支持查询性能均较优。

2.5 SpringMVC+AngularJS 架构

SpringMVC 是基于 Java 的 MVC 框架，常用来做 WEB 开发。一般来说，MVC 框架将网页的开发逻辑分解成模型层（Model）、视图层（View）和控制器层

（Controller）三方面，有效地解耦了页面逻辑，使得各部分各司其职，从而简化开发。

Spring MVC 基于 Spring IoC，将控制器（Controller）交给 Spring 容器管理，与其他 JAVA WEB 的 MVC 框架相比，该框架可以更好地利用 Spring 的其他组件，如 Spring 的数据库事务管理（Spring transaction）、安全管理（Spring Security）等。

AngularJS 是 Google 开发的优秀前端技术框架，是为了弥补 HTML 在构建网页应用程序上的不足而设计的,且拥有模块化、依赖注入、数据自动化双向绑定、语义化标签等诸多优秀的特性,降低了动态网页应用程序的开发门槛，大大简化了应用程序的开发[21]。此外，其主要偏向于考虑构建 CRUD 应用，因此在本系统大规模可视化视图的构建中显得尤为重要，将数据模型（data-model）结果关联到视图（UI）层面上[22],极大解决了数据集众多属性和大量视图显示时候的加载等问题。

2.6 现有工作的不足和问题提出

本文核心要解决的核心问题是针对 Zenvisage 等可视化分析平台在面临海量数据场景下的存储和查询的瓶颈，设计出适合的时序大数据分析方案，并集成于 Zenvisage 的可视化分析系统底层，现有工作的不足之处有：

1）Apache Zeppelin，提供了 B/S 结构的交互式分析平台，辅助人工进行数据分析和可视化视图显示。此外，该平台底层可以接入大数据下的不同的数据处理引擎。该平台主要实现了数据采集、数据发现、数据分析、数据可视化和协作等用户所需的功能[23]，但此类似于 1.1.1 中所述的 Tableau、Spotfire 等可视化分析软件，虽然一次可以检查多个可视化视图，但无法从可视化集合中自动识别相关可视化，且是手动密集型操作，一定程度上不能满足用户复杂的期望。

2）Zenvisage 平台在一定程度上解决了上面提出的问题，但是其可视化分析系统对于数据集存储策略是面向关系型数据库,数据结构统一、信息相对集中，通过行存储结构形式进行海量数据的存储。这种情况下只针对于小数量级，且需要提前在数据库中对响应的属性字段进行创建，在当下大数据时代，无法满足在性能和运行效率的有效性，且造成的 I/O 操作、网络带宽大。

3）Zenvisage 等现有系统为实现可视化的目的，在底层分析时都是对单个 SQL 操作进行查询，这样在用户不断请求下会对数据库服务器造成很大的压力，并且这些请求很可能是类似的，这在一定程度上也是对 I/O 操作和应用服务器的计算资源的浪费。

2.7 本章小结

本章首先介绍了关于可视化分析技术的相关研究现状，主要是从概念、步骤和相关工作介绍等三方面进行阐述，具体介绍了 Zenvisage 平台的主要特点、ZQL 结构、操作简介和视化分析处理步骤等；之后，就大数据时代下 HDFS 列存储技术和 SQL on Hadoop 系统进行了详细介绍。最后本章指出了针对 Zenvisage 等可视化分析平台在大数据处理和可视化分析方案上现有工作的不足，提出了本文在可视化分析研究工作的方向。

第3章　基于 Spark 的时序大数据分析方案

本章提出了针对于海量数据的基于 Spark 的时序大数据分析方案，并详细介绍了对 Zenvisage 的优化设计，包括 HDFS 列存储和 Spark SQL 查询策略，以及其中的查询结果缓存优化和其他相关改进部分。

3.1 问题定义与分析

如前所述，本章针对的是 Zenvisage 平台在时序大数据场景下的存储和查询处理瓶颈。传统关系型数据库采用行存储结构形式进行数据存储，查询时造成的 I/O 操作、网络带宽大，Zenvisage 平台虽然一定程度上解决了用户在可视化分析时遇到的一部分问题，但是其在时序大数据分析的应用场景下遇到了很大的瓶颈。

在第 2 章中介绍了 HDFS 列存储技术，一定程度上方便了海量数据的存储，同时在提供 Spark SQL 查询时也是节省了时间，提高了效率。基于 Spark 的时序大数据分析方案中的操作流程可大致描述为：

1. 将样例数据集的字段说明文件（*.txt）提交至 HDFS 文件系统上，并对文件内容进行解析；
2. 将样例数据集（*.csv）提交至 HDFS 文件系统上，并根据 1 中的内容解析结果对各个数据集进行 Parquet 列存储转化，可以删除源文件；
3. 选定数据集，且 SQL 查询提交后，通过 Spark SQL 对 2 中的 Parquet 列存储文件进行查询，返回查询结果 List；
4. 根据 List 结果集构造可视化。

如下图 3-1 架构图所示，可以看出此方案在大数据场景下的三大技术点：第一点在于各个数据集的数据模型设计，由数据集文件（*.csv）和数据字段说明文件（*.txt）组成；第二点在于 Parquet 列存储转化；第三点在于 Spark SQL 读取结果集。

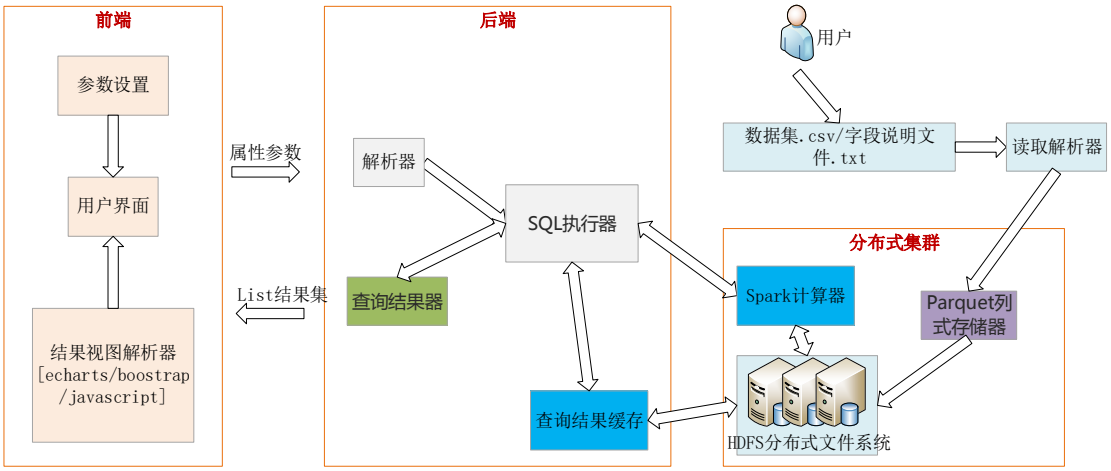


图 3-1 基于 Spark 的时序大数据分析方案架构图

3.2 数据模型设计

针对海量数据，本文在大数据场景下研究的技术点之一在于对各个数据集进行数据模型设计，使得在进行 Parquet 列存储时可以对数据的各字段类型进行标明，便于 Spark SQL 在查询时可以直接仿照关系型数据库的 SQL 查询。

3.2.1 文件结构

1）源数据集

每个数据集含*.txt 和*.csv 两个文件，前者是字段说明文件，后者是样例数据集文件。此处需要对字段说明文件进行统一的文件结构进行设计，以数据集的字段说明文件为例，如下表 3-1 所示。

表 3-1 字段说明文件结构设计表

数据集名称	详细信息				
Weather.txt	location:string ,F,F,T, month:int ,T,F,F dayofyear:int ,T,F,F year:int ,T,F,F temperature:float ,F,T,F,				
字段说明文件	字段名称	字段类型	X 属性	Y 属性	Z 属性
*.txt	name	dataType	xAxisColumns	yAxisColumns	zAxisColumns

由此表 3-1 可知，将关系型数据库中的数据表字段通过文本进行说明设计，

也便于程序理解和用户设计。此处，在字段说明文件结构设计中需要确定各字段的轴属性，在应用程序界面上的“属性选择区”得以支持用户修改查询条件。

2）HDFS 文件目录

在 Hadoop 集群上，关于数据集在 HDFS 文件系统上的存储结构也需要一定的设计，便于程序进行更好地查询和存储操作。其中，本文研究方案中设计的 HDFS 文件系统上的目录结构如下图 3-2 所示。

```
hohai/
├── data/
│   ├── weather.txt
│   └── weather.csv
├── common/
│   ├── visualization_rowobject
│   ├── visualization_metafilelocation
│   └── visualization_metatable
├── cashe/
│   └── cashe.txt
├── sql/
│   ├── *.txt
│   └── datetime.txt
└── parquet/
    ├── ***
    └── weather/
            ├── _SUCCESS
            └── part-*.parquet
```

图 3-2 HDFS 目录结构示意图

由以上 HDFS 文件系统上的目录结构可知，data 文件夹下是上传的源数据集，parquet 文件夹下是各数据集的 Parquet 列存储转化之后的存储位置；此外，common 文件夹是为了便于系统的动态查询而设置，其下的 metafilelocation 保

存了各数据集的*.txt 和*.csv 文件目录信息，metatable 保存了各数据集的字段名称和类型信息，rowobject 保存了各数据集的字段说明文件的字段和字段类型信息。

3.3 HDFS 列存储

HDFS 列存储，即对分布式文件系统 HDFS 上的存储数据集以 Parquet 文件形式存储，使得原本在 HDFS 上的存储空间大大降低，同时在 Spark 等分布式计算框架下的查询效率更高。

3.3.1 HDFS 文件系统的部署

为了能够部署 HDFS 分布式文件系统，搭建 Hadoop 分布式集群至少需要 3 个节点，即 3 台虚拟机，每个虚拟机的运行环境为：CentOs7，单个虚拟机的配置 CPU 核数为 2 个，内存为 2G，单个节点为 20G 的磁盘空间。

各节点的 IP 地址分别为：192.168.160.128（Master），192.168.160.129（Slave1），192.168.160.131（Slave2）。

正如前文 2.3 节中提及的部分内容，部署结构图为：

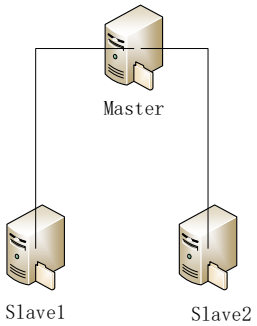


图 3-3 HDFS 分布式文件系统部署图

3.3.2 文件功能

1）文件目录生成

在 HDFS 文件系统上进行文件存储时，需要提供文件目录生成的接口，主要功能是根据文件类型创建存储地址，适用于数据集接收、查询缓存临时文件等场景。

2）文件删除

删除 HDFS 上存储的文件。

3）文件读取

读取 HDFS 上指定地址的文件，如 Spark SQL 读取 parquet 内容、FileSystem 读取 metafilelocation 内容等。

4）目录删除

删除 HDFS 上存储的目录。

5）显示指定目录下的所有文件

根据用户指定的目录，显示出其下所有的文件名。

6）上传内容

在 HDFS 文件系统上特定的文件下上传数据内容。

3.3.2 列存储

Apache Parquet 存储引擎是大数据生态圈下的一种文件列式存储格式，其主要优点是对文件进行了双重压缩，在存储上大大减少了源数据集的存储空间，同时兼容开源大数据下的计算引擎(Mapreduce、Spark 等)，大大降低了查询时间。

文件结构

Parquet 文件以二进制方式存储，不能直接读取、修改；同时，Parquet 文件是自解析的，文件中包含了该文件的数据和元数据。在 Parquet 文件中存在如下几个概念：

- 行组(Row Group)：按照行将数据物理上划分为多个单元，每一个行组包含一定的行数，在一个 HDFS 文件中至少存储一个行组，Parquet 读写的时候会将整个行组缓存在内存中，所以如果每一个行组的大小是由内存大的小决定的。
- 列块(Column Chunk)：在一个行组中每一列保存在一个列块中，行组中的所有列连续的存储在这个行组文件中。不同的列块可能使用不同的算法进行压缩。
- 页(Page)：每一个列块划分为多个页，一个页是最小的编码的单位，在同一个列块的不同页可能使用不同的编码方式。

联系前文 3.3.1 节中对 HDFS 文件系统中的部分概念介绍，通常情况下，在存储 Parquet 数据的时候会按照 HDFS 的 Block 大小设置行组的大小，由于一般情况下每一个 Mapper 任务处理数据的最小单位是一个 Block，这样就可以把每一个行组交给一个 Mapper 任务处理，便于增大任务执行并行度。Parquet 文件的格式如下图 3-4 所示。

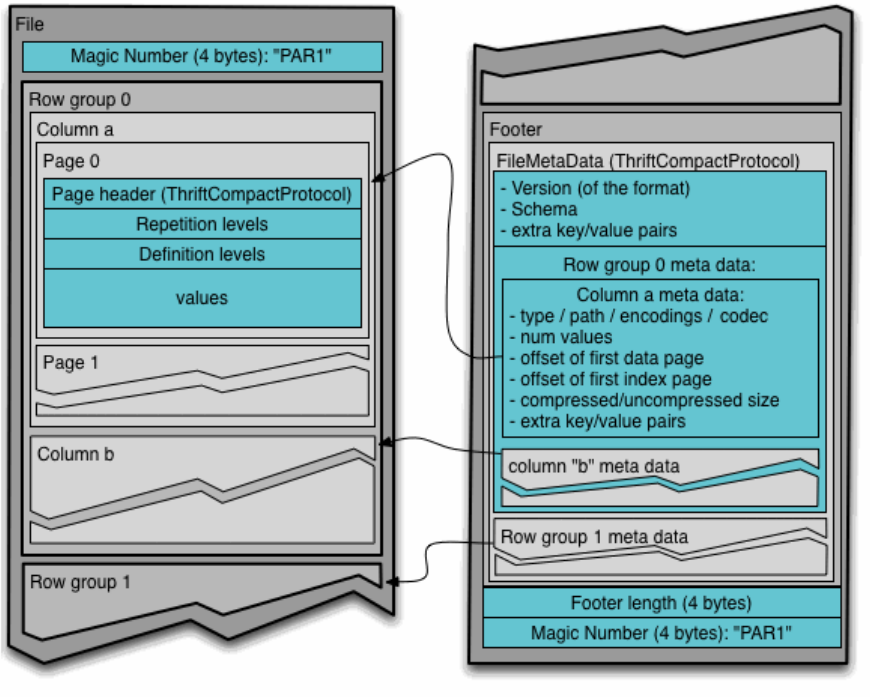


图 3-4 Parquet 文件结构

3.4 Spark SQL

类似于传统关系型数据库中的 SQL 查询语句，Spark SQL 应用的查询语句也是由投影、数据源、过滤等条件组成。在本文 Spark 查询的研究方案中，我们使用 Spark SQL 分支之一的 sqlContext，可支持 SQL 语法解析器，其具体运行过程如下图 3-6 所示：

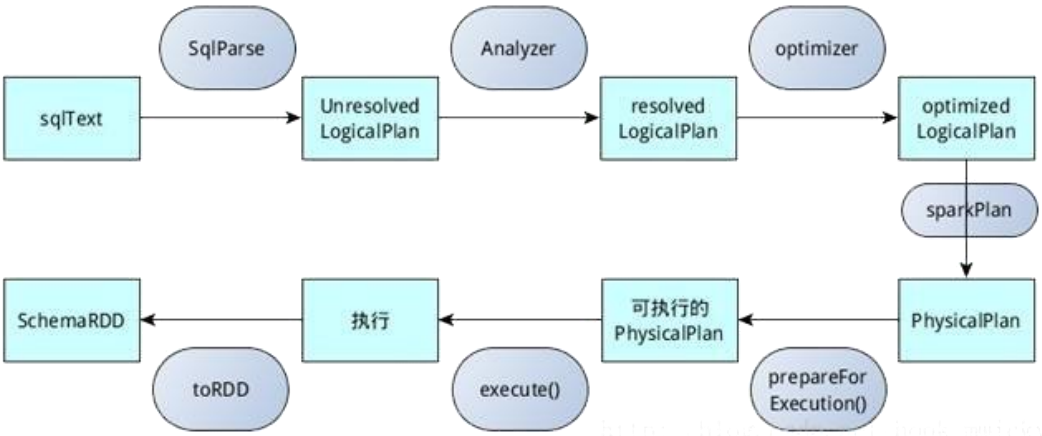


图 3-5 sqlContext 的运行过程示意图

在 sqlContext 的整个运行过程中，同样会涉及到 SqlParse、Analyzer 等多个 SparkSQL 的组件[24]。此外，Spark 应用程序可以通过 RDD、JSON 格式数据等数据源创建 DataFrames，再将 DataFrame 注册为临时表后，就可以对该 DataFrame 执行类似 SQL 查询的操作，从而获取应用的结果集。

3.4.1 RDD 机制

如上文 sqlContext 的运行过程示意图 3-6 所示，Spark SQL 会将执行结果转化成 RDD 对象（Resilient Distributed Datasets，即弹性分布式数据集），运行于内存中。以 Spark 官方提供数据为例，在某些场景下，RDD 的计算效率可以达到 Hadoop 的 20 倍，可见 RDD 效率之高[25]，其内部的实现机制介绍如下：

1. RDD 数据是只读的，不可修改[25]。如果需要修改数据，必须从父 RDD 转换（transformation）到子 RDD。所以，在容错策略中，RDD 没有数据冗余，而是通过 RDD 父子依赖（血缘）关系进行重算实现容错。

2. RDD 数据存储在内存中，使得数据不用落地到磁盘上[25]，从而提供了低延迟性，避免不必要的 I/O 操作。

3. RDD 存放的数据可以是 java 对象，可以避免不必要的对象序列化和反序列化[25]。

总的来看，RDD 高效的主要因素在于尽量避免了不必要的操作和牺牲数据的操作精度，以此提高计算效率。由此，本文底层计算方法也是通过 Spark SQL 使用 RDD 机制实现 sql 查询。

3.4.2 查询方案

Spark SQL 查询方案允许用户使用 SQL 语句或丰富的 DataFrame API 来查询 Spark 程序中的结构化数据，由于海量数据都是有一定结构的，因此符合应用场景要求。其中，根据数据源类型，Spark 官方提供了三种方案。

1）通过 Jdbc 方式操作数据库

此方式是最基本的从数据库中读写数据的方法，Spark SQL 通过将读取的数据转化为单个 DataFrame，然后借助 Spark 的 API 来对数据进行其他操作。

2）对源数据集（csv、json 等格式）进行访问

对 csv 等源数据集直接访问是 Spark SQL 中最常见的方式，利用 SparkSeesion 进行创建，一般是 sparkSeesion 读取该数据集的内容，接着同 1）来进行操作。

3）对 Parquet 文件访问

Spark SQL 支持对 Parquet 文件的读取和写入，并且自动保留初始数据的数据类型。Parquet 模式通过 Data Frame API，Spark SQL 可以识别这些数据文件，并同（1）中所述将数据加载到单个 DataFrame 中，也可以将其注册为临时表，从而直接进行 SQL 查询，性能大幅度提升。

对于以上三种数据源的操作方案，在后文 5.2.2 节中对查询性能进行了实验设计，本文考虑到存储和查询优化选择了方案 3，系统稳定，对于 Zenvisage 在时序大数据应用场景下的瓶颈有一定的助力。

3.4.3 查询结果缓存

如前文中 3.2.1 节中 HDFS 目录结构示意图所示，cashe、sql 文件夹主要是为了实现用户的查询结果缓存。其中，cashe 文件夹下的 cashe.txt 主要记录了某时刻下的 SQL 查询语句，相当于 Map，再将所有查询记录构成一个 Json 字符串，在系统启动时读取 metacashe 内容相当于本地缓存，系统结束时销毁。sql 文件夹下的*.txt 文件的名称对应 Map 中的 Key 值，datetime.txt 文件中存储的内容为相应 Key 值对应的 SQL 查询结果集。由此，从对于用户的 SQL 查询结果访问流程可以如下图 3-5 所示。

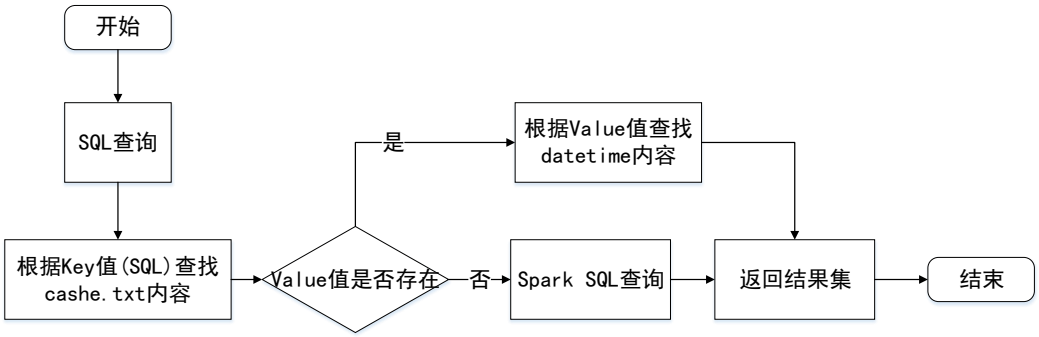


图 3-6 SQL 查询结果访问流程图

3.5 效果展示

以上详细介绍了对 Zenvisage 平台的底层设计的优化部分，是本文研究的核心内容，本节对基于 Spark 的时序大数据分析方案作一个简单的基于 Spark 的视图控制平台效果演示，如下为实际效果图。

图 3-7 时序大数据分析方案的效果演示图

其中，选择 Zenvisage 平台下的 weather 数据集为数据源，控制面板中进行 SQL 语句、Spark SQL 查询方案的设置，视图预览窗口通过 Echarts 开源库显示

具体查询的可视化，后台消息窗口主要是显示该查询方案的消耗时间。此外，考虑到实际效果，关于 HDFS 和 Parquet 操作部分是在后台预先进行，这里不做介绍。

3.6 本章小结

本章主要阐述了针对 Zenvisage 平台在时序大数据场景下的瓶颈设计并实现的基于 Spark 的时序大数据分析方案，首先分析了此方案的流程和重要技术点，并详细介绍了优化设计部分，包括 HDFS 列存储和 Spark SQL 查询策略，以及其中的查询结果缓存优化和其他相关改进部分。

第4章　时序大数据的可视化分析系统实现

本可视化分析系统由SpringMVC进行底层架构设计，前端采用支持数据自动化双向绑定、依赖注入等特性的AngularJS优秀框架，弥补了HTML在构建网页应用程序上的弊端，并且应用Dygraph这一开源的可交互式、可缩放的曲线表，使用户可以实现视图拖、拉、拽等功能。系统界面模块中的UI直接采用Bootstrap框架主题样式。

接下来主要是对于系统实现中的开发工具及开发环境、设计原则、系统总体结构设计等几大模块进行介绍。

4.1 开发工具及开发环境

本文开发的时序数据可视化分析系统主要是以HDFS（Hadoop Hadoop Distributed Filesystem）进行文件存储、分布式内存引擎Spark为底层计算框架，需要搭建分布式集群开发环境，服务器端通过SpringMVC来集成系统。主要开发工具有：系统开发工具为IntelliJ IDEA 2017、Apache Maven3.0.5及以上，系统运行环境为Linux、Java Platform（JDK）>=8、Jetty服务器、Spark+Hadoop集群>=2.7.1。

4.1.1 系统开发工具IntelliJ IDEA 2017

IntelliJ IDEA，是集成了Java语言开发环境以及开放源代码的可扩展平台，提供丰富的导航模式和历史记录功能，便于开发人员更快地查找自己的程序，在程序调试时直接通过“光标+按键”就可以被IDEA理解，从而迅速得到结果。同时还集成了git、svn等常见版本控制工具的插件，便于直接在IDEA中完成代码的提交、检出等工作。

4.1.2 Web服务器

本系统应用开源的servlet容器Jetty作为底层的web服务器，此服务器的优点是开发人员或系统运行人员不需要安装Apache Tomcat等Java的web容器，可以通过Jetty这个服务器更快地为系统应用提供网络和web连接。

Jetty服务器相对于Apache Tomcat来说更加轻量级，灵活性更高。由于

Apache Tomcat 在遵循 Java Servlet 规范之外，其自身还扩展了满足企业级应用需求的 JavaEE 特性，因此使用 Jetty 更加能够为应用服务器节省内存资源，且 Jetty 作为一个优秀的组件，其设计目的并不是为了需要修改，而是容易嵌入到应用程序中。

4.1.3 Hadoop+Spark 集群

与现有的关系型数据库相比较，Apache Hadoop 框架设计的意思主要是为了海量的数据提供存储，通过流的形式访问写入大型文件，其中的 HDFS 支持 Parquet 列存储格式的优化，对存储文件进行压缩，目前 Hadoop 技术在互联网领域已经得到了广泛的运用,例如:Facebook 使用 1000 个节点的集群实现 Hadoop 应用程序的运行,实现对于大规模日志数据的存储,并支持数据分析、机器学习；百度使用 Hadoop 处理每周高达 200TB 的数据，从而进行搜索日志分析、网页数据挖掘以及其他方面的研究工作[26]。

Apache Spark 与 Map/Reduce 一样都是分布式集群计算框架，但是 Spark 在负载方面通过内存计算来进行交互式查询，其内部集成了文本处理、SQL 查询、机器学习等优秀的组件，可以在 HDFS 文件系统基础上进行低延迟、大型的数据分析应用。

4.2 设计原则

(1) 可靠性

可靠性是指数据人员在使用本系统时所查询到的数据信息真实有效,视图可视化展示完整，和 Hadoop 集群上存储的 Parquet 数据集文件内容一致，不会出现数据缺失、不完整等情况。

(2) 可维护性

可维护性是指对系统理解、更新的难度。系统维护是指系统在开发完成后投入使用至被淘汰的这个时间段内,为了改正其中的错误或为了满足用户新的的功能性需求而对系统本身进行更新的操作[错误!未找到引用源。]。

(3) 可复用性

可复用性指的是系统本身可以复用到其他新的领域中,对于本文设计的分析

方案使用现有的开源列存储引擎 Parquet 和 SQL on Hadoop 系统 Spark 中，由于在开发思想上开源引擎的原理相似,因此也可以应用到列存储引擎 ORC 和 SQL on Hadoop 系统 Hive 或 Presto 中，这样可以使系统底层更具有普遍应用性。

4.3 系统总体结构设计

在设计本系统时，主要考虑将基于 Spark 的时序大数据分析方案集成于 Zenvisage 的可视化数据探索系统底层，其在功能上主要分为四个模块内容，分别为数据上传、后台数据读取、后台数据转化、Zenvisage 可视化数据探索。

其中，数据上传模块主要负责将系统数据集样例和用户上传数据集提交至 HDFS 文件系统上，并进行临时目录建立、文件更新等操作；后台数据读取模块主要负责根据前台提交的 SQL 查询读取所需相关联的数据；后台数据转换模块主要负责对后台读取到的数据进行相关的转换工作，以便 Zenvisage 可视化数据探索系统模块底层的调用。Zenvisage 可视化数据探索系统模块主要负责将转换完毕的数据进行视图呈现、交互式分析。

系统结构图如下图 4-1 所示。

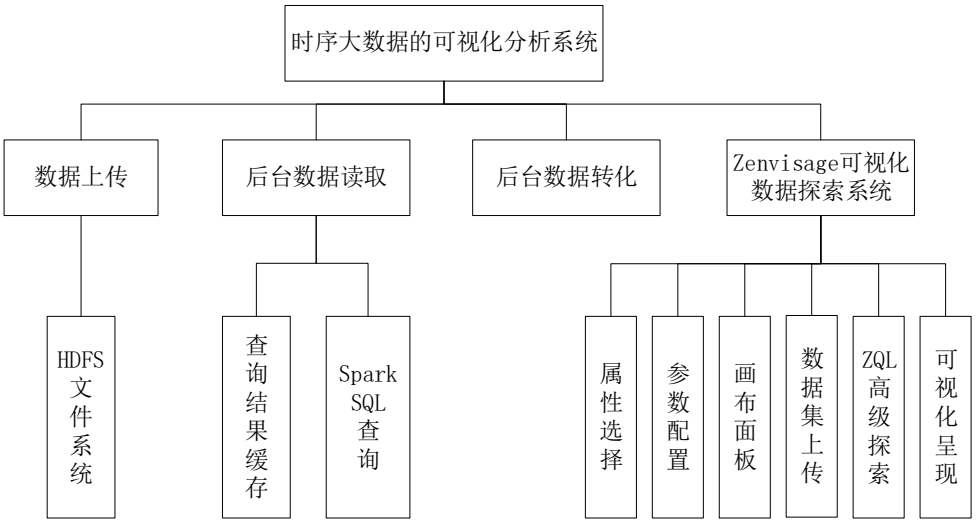


图 4-1 时序大数据的可视化分析系统的系统结构图

4.4 软件结构设计

本系统在功能实现上主要分为四个模块内容，分别为数据上传、后台数据读取、后台数据转化、Zenvisage 可视化数据探索。下面分别进行介绍。

4.4.1 数据上传

本系统的数据来源主要是两方面：一方面来自系统启动时本地的数据集，另一方面来自用户用于分析提交的数据集。在系统启动时，本模块需要先将本地数据集上传至HDFS文件系统上，并进行数据块的副本复制和文件Parquet转化，经过适当的处理工作后（如：检查 HDFS 文件系统上是否已经有完整目录、缓存数据是否存在等），再为系统后期需求建立临时目录和数据缓存等操作。

4.4.2 后台数据读取

本模块通过 SQL 查询读取的数据来源也主要是两方面：一方面来自 HDFS 缓存文件数据，另一方面来自 HDFS 文件系统上的 Parquet 数据集文件，这种查询缓存的设计在一定程度上减少了后台接口的访问次数和 I/O 操作。其中，具体介绍内容如前文 3.2.1 节和 3.4.3 节中的 HDFS 目录结构、查询结果访问流程图所述，其结构图如 4-2 所示。

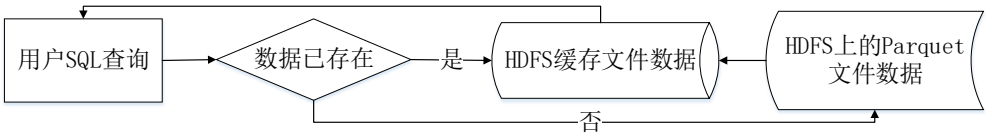


图 4-2 后台数据读取功能结构图

4.4.3 后台数据转化

后台数据转换模块的主要功能是负责将后台数据读取模块读取的数据文件（Json 格式文件或 Parquet 文件等）根据底层接口的设计要求对文件内容进行转换。其中，一部分属于直接读取数据操作（如 Json 格式文件），一部分属于 Spark SQL 进行加载数据集操作，具体可分以下几个步骤实现：

（1）打开 HDFS 文件，进行 Laod 操作读取文件内容；

（2）如果接口请求内容是 JSON 格式文件，则读取内容为 JSON 字符串，可以将其转换为 JSON 对象[27]，然后进行步骤 3；如果接口请求内容是 Parquet 文件，则读取结果为 DataFrame，进行步骤 4；

（3）对转换过得 JSON 对象进行数据过滤、分类等转换工作；

（4）如 3.4.2 节中查询方案中的操作步骤所述，将单个 DataFrame 注册为

临时表，从而直接进行 SQL 语句查询，并将查询结果转化成 Json 对象。

（5）将步骤 3、4 中转换过得 JSON 结果提交给 Zenvisage 底层处理，最后将处理结果在前端进行显示输出。

4.4.4 Zenvisage 可视化数据探索系统

由于本文研究的基础是 Zenvisage 平台，前文已经具体介绍了其特点、操作、可视化分析处理步骤等相关工作，本文最终实现的时序大数据的可视化分析系统沿用了其大部分功能，并负责对后台数据转化模块传递的数据进行接收，从而满足用户或者数据人员交互式分析的需求。

同时，本模块也是负责将前端界面的指令（如 ZQL 高级探索、属性选择等操作）提交至后端，也是与后台数据读取模块不可分割。

4.5 系统架构设计

前文 3.1 节中介绍了基于 Spark 的时序大数据分析方案架构图，考虑到本文在 Zenvisage 平台基础上研究并实现的可视化分析系统由 SpringMVC、AngularJS 等众多优秀的前后台框架组成，同时加上 Hadoop 集群上的 HDFS 目录设计结构，我们还需要在之前的架构图上添加系统其他功能的机制。

因此，我们将系统架构抽象成如下图 4-3 所示：

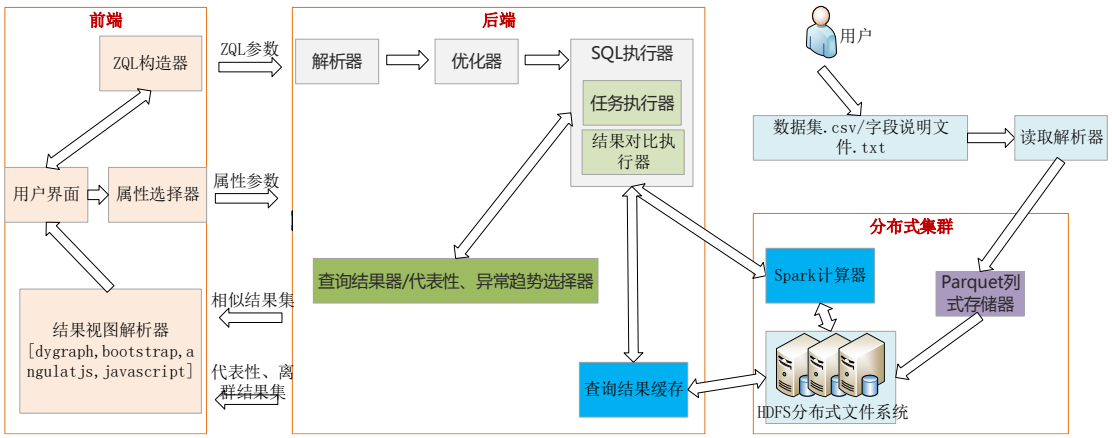
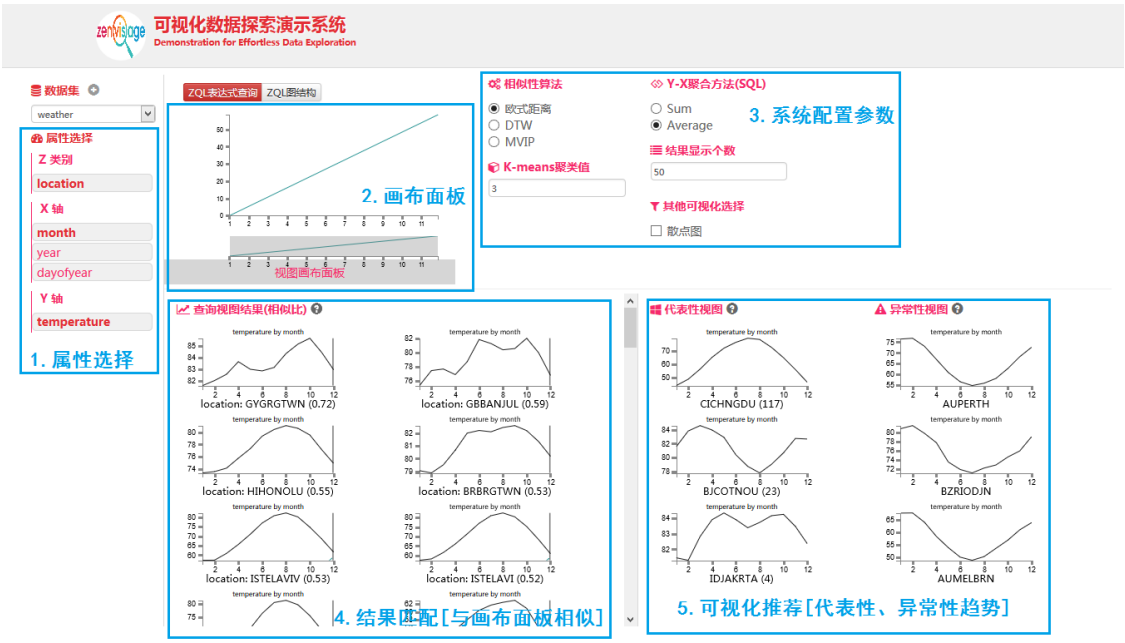


图 4-3 时序大数据的可视化分析系统的系统架构示意图

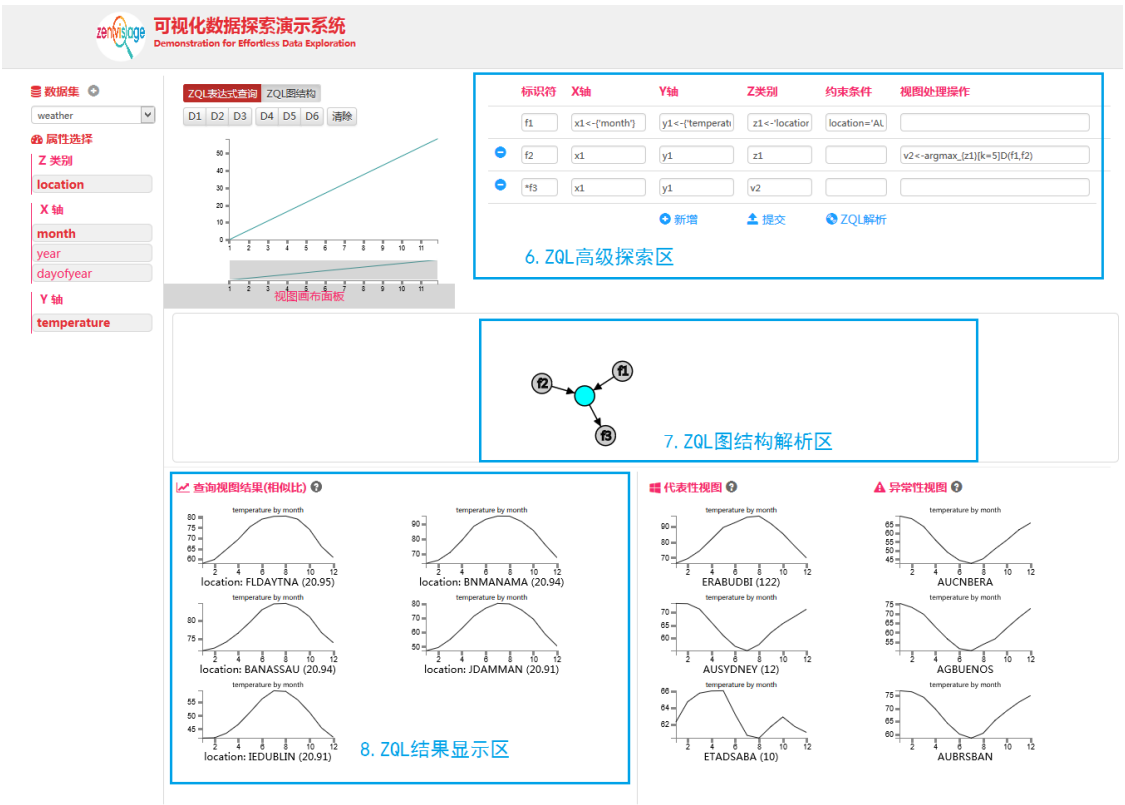
4.6 系统运行成果图

本文将基于 Spark 的时序大数据分析方案集成于 Zenvisage 的可视化数据探索系统的底层，并对 Zenvisage 平台进行功能上的调整，本节将对系统功能的

实现效果做一个界面化的演示。如下图 4-4 为系统在 PC 端的运行成果图：（a）系统主面板区域；（b）ZQL 高级探索区；（c）数据源上传区。



（a）主面板区域



（b）ZQL 高级探索区

（c）数据源上传区

图 4-4 时序大数据的可视化分析系统的界面展示

由上图 4-4 中所知，在前文 2.2.2 节中已经介绍了 Zenvisage 平台的几部分功能，在此处为了系统的完善添加了（7）ZQL 图结构解析区，（8）数据源上传区。其作用分别为：

7）ZQL 图结构解析区： 每个有效的 ZQL 查询都可以通过解析查询计划转化成一个有向无环图（DAG）的形式。 DAG 包含集合节点（c-node），用于表示 ZQL 查询中的可视化集合，以及进程节点（p-code），用于表示操作列中的集合操作过程。

8）数据源上传区：对于每个数据集的探索过程的前提是需要字段说明文件和样例数据集文件，正如前文 3.2 节中数据模型设计所述，此部分的功能即提供一个系统的上传接口，便于数据集的更新、添加等操作。

4.7 本章小结

本章在前文 Zenvisage 平台和本文核心内容的提出上进行系统的实现，将基于 Spark 的时序大数据分析方案集成于 Zenvisage 平台底层。本章首先是对开发工具和开发环境做了简单介绍，提出了系统的设计原则。然后，详细介绍了系统总体结构设计和软件结构设计，并对系统运行成果图进行了展示。

第5章　实验结果分析

本章主要通过实验测试出本文提出并实现的基于 Spark 的时序大数据分析方案的查询效率和列存储的有效性。本文首先给出了实验设计，包括集群配置、数据集以及对比方案等。然后从数据存储的大小、数据查询性能测试实验等两个方面来评估本文研究内容的有效性。

5.1 实验设计

本文提出的基于 Spark 的时序大数据分析方案是针对海量时序数据的存储和查询的分析方案，由于海量数据的特点是不确定性数据集大小和字段数量，因此本文选择 Zenvisage 平台应用场景下有代表性的数据集;同时,考虑 Zenvisage 平台中的 jdbc 查询是在单节点的环境下，所以 SQL on Hadoop 系统 Spark 也是通过单节点来执行。基于 Spark 的时序大数据分析方案其效率实验设计将分别从集群配置中的软硬件资源、SQL 查询负载设计以及对比方案等方面介绍。

本实验的时序大数据分析方案中的数据存储部署于拥有 3 个节点的 Hadoop 分布式集群上，其中 1 个节点作为主节点，其他 2 个节点作为从节点[28]。每个节点 CPU 核数为 2 个，内存为 2G，操作系统使用当前最新版 CentOs7。每个节点拥有 20G 的磁盘空间，且都是在虚拟机中搭建；同时，Zenvisage 分析方案中的数据存储部署于 Postgresql 中，虚拟机环境一致。此外，Java 版本为 1.8，Postgresql 版本为 9.5，Hadoop 版本为 2.7.3，与 Hadoop 集群配套的 Spark 版本为 2.1.1。

在数据集选择上，考虑到 Zenvisage 平台的应用场景的有效性，我们选择其在 Postgresql 数据库中的 weather 数据表，目前在 Github 中的数据集大小为 24.5MB。为了扩展 weather 数据集的大小，通过 Python 程序将数据规模扩展大小为 485.2MB、1.02GB、3.02GB、5.02GB、8.02GB。

在 SQL 查询负载设计上,我们考虑到 Zenvisage 平台中后台使用的 SQL 查询语句，主要在于 Group By 分组、where 条件等操作，同时参考 HAIL[29]的实验设计中对 UserVisits 表的查询设计，将实验查询从查询字段类型、数据规模和查询效率三个维度上对 weather 数据集设计如下表 5-1 所示：

表 5-1 实验 SQL 查询设计表

名称	SQL 语句
Q1	SELECT location, month, avg(temperature) FROM weather where location = 'BRBRGTWN' GROUP BY location, month ORDER BY month
Q2	SELECT location, month, avg(temperature) FROM weather GROUP BY location, month ORDER BY month
Q3	SELECT location, year, avg(temperature) FROM weather where location = 'BRBRGTWN' GROUP BY location, year ORDER BY year
Q4	SELECT location, year, avg(temperature) FROM weather GROUP BY location, year ORDER BY year

其中，Q1 和 Q3、Q2 和 Q4 分别属于一类查询，前者包含 where 条件，属于对单个 location 的细粒度查询；后者对所有 location 的结果进行查询。

Q1~Q4 中 每一个查询在 24.5MB->8.02GB 的数据规模上均进行了相应实验，以更好地提现基于 Spark 的时序大数据分析方案的优化性能。

在对比实验方案中，我们使用相同的数据集、相同的 SQL 查询负载，通过数据集存储大小进行存储方案的对比，以及 Zenvisage 中的 jdbc 方式查询、本文研究方案查询以及 Postgresql 管理器查询等三方面进行查询效率对比；同时，针对现有的 Spark SQL 中不同查询方案下的时间性能对比。为了保证查询结果的准确性，每次查询执行都是在执行结果稳定时记录，且每个查询的查询执行时间都是执行三次后计算平均时间而得来。

5.2 对比方案

由上述 5.1 节中实验设计可知，本文的实验对比方案主要从存储和查询两方面来进行。

5.2.1 存储方案

本文提出的存储方案是先将数据集提交至 HDFS 文件系统上，然后进行 Parquet 列存储转化，减少了数据集在 HDFS 上的存储空间，对比 postgresql 数据库和实际数据集大小进行实验对比。

5.2.2 查询方案

对于数据集的结果查询性能是针对相同的 SQL 在不同查询方案下的效率对比，本文将设计两个实验，实验一是通过 Postgresql 管理器、jdbc 和 Spark SQL 三方面进行实验，此实验不仅可以测试出三种方式在 SQL 查询时的效率，还可以证明随着数据规模的变化来验证本文采取的大数据分析方案的有效性；实验二是针对现有的 Spark SQL 中不同查询方案下的时间性能对比，查询 SQL 查询为 Q1，选择的数据集规模分别为 24.5MB 和 485.22MB，继而验证选择 Spark SQL 对于 Parquet 列存储文件进行查询时能够带来查询性能的优化。

5.3 结果与分析

1）存储方案实验结果

本文提出的存储方案将 HDFS 文件系统上的数据集进行 Parquet 列存储转化，对比 postgresql 数据库和实际数据集大小的实验对比如下图 5-1 所示。

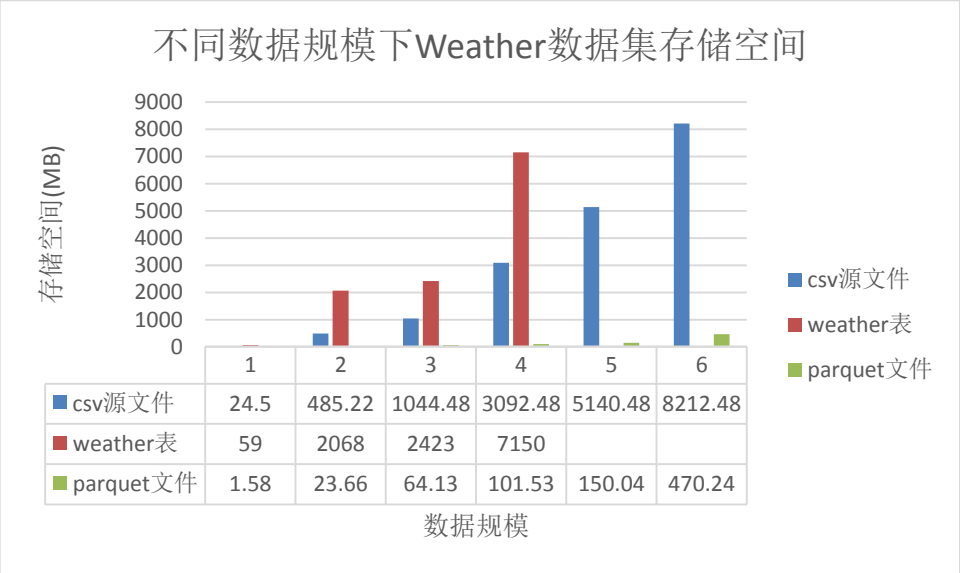
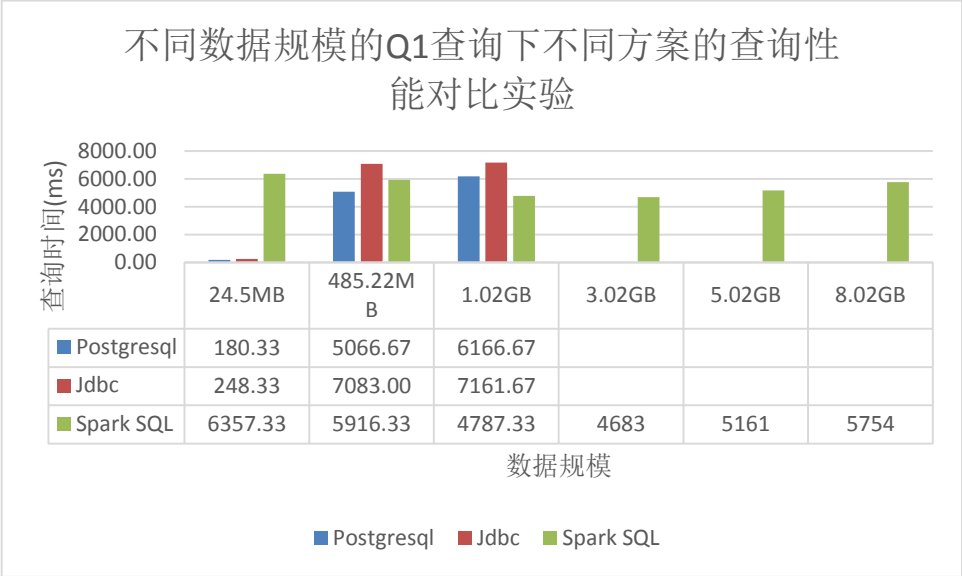


图 5-1 不同规模下 Weather 数据集存储空间对比实验

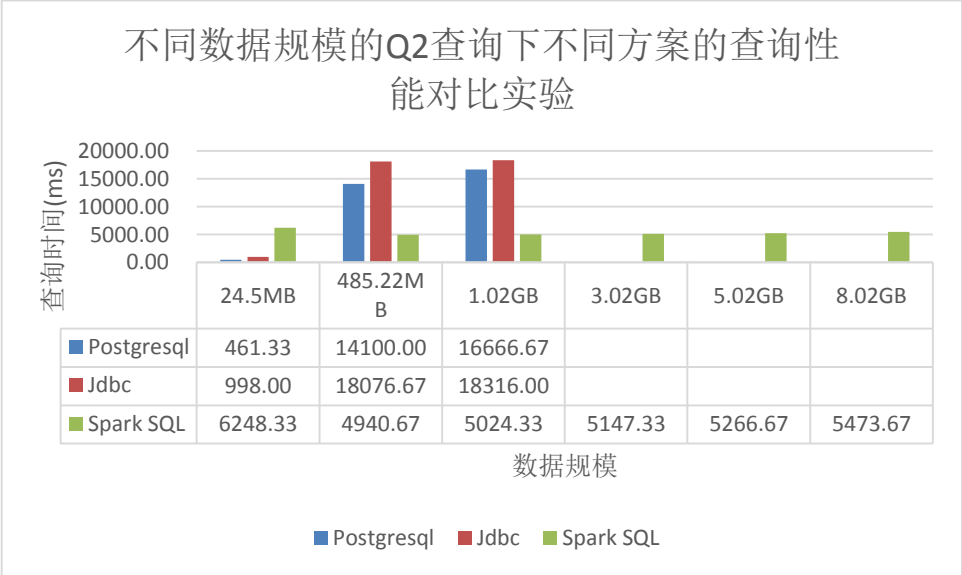
由上图 5-1 可知，在 5.02GB、8.02GB 数据规模的源数据集中，导入 Postgresql 数据库在实验中已经造成了很大的困难（这里为略），且关系数据库行存储结构占用了很大表空间；此外，HDFS 文件系统不会影响数据集大小，但在本文方案中将源数据集转成 Parquet 格式，对数据集的存储性能均有大幅度提升。

2）查询方案实验结果

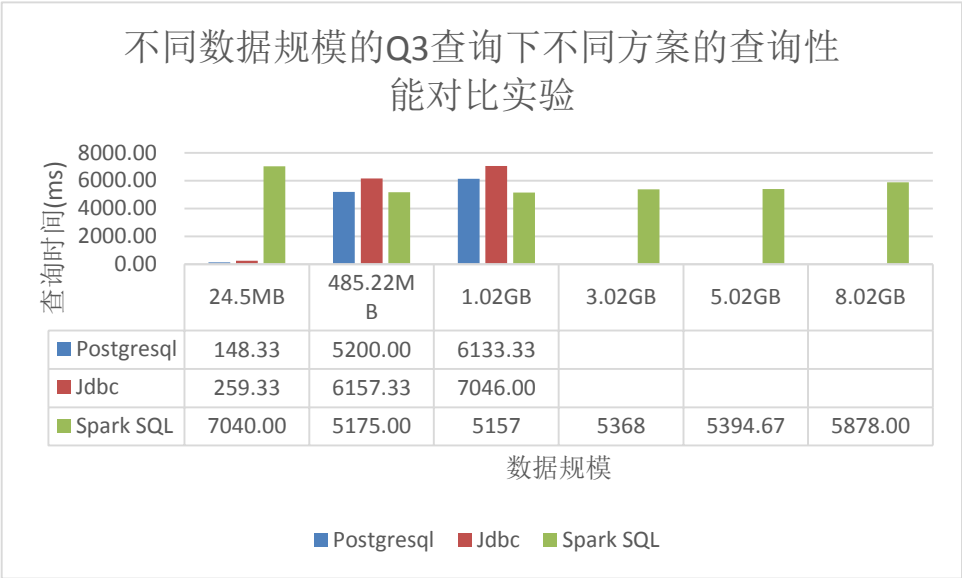
实验一是通过 Postgresql 管理器、jdbc 和 Spark SQL 三方面进行实验。实验设计如下，实验包括 Q1~Q4 这 4 条 SQL 查询语句，通过对 24.5GB~8.02GB 等不同数据规模条件下的查询性能对比实验。按照此设计实验并进行测试，实验一的测试结果如下图 5-2 所示。



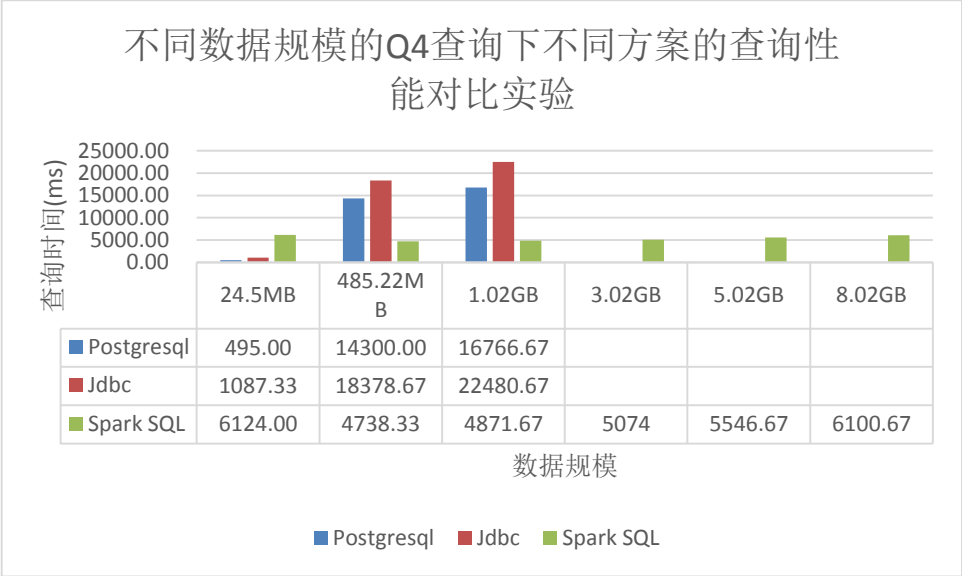
（Q1）



（Q2）



（Q3）



（Q4）

图 5-2 不同数据规模的 Q1~Q4 查询下不同方案的查询性能对比实验

如图 5-2 中所示的在 Q1~Q4 查询条件下，我们对不同数据规模下的 Weather 数据集查询结果在不同方案下进行了对比实验。在查询时间上，随着数据集的增加，各方案中的查询时间基本呈上升趋势；此外，在 24.5MB~8.02GB 的数据集上，Spark SQL 的数据查询方案在数据集更大的情况下明显优越，其中，在数据集 GB 级上时根据 5.2.1 方案中的存储无法进行查询（此处为略，按照理想推断其查询性能仍不如 Spark SQL 方案）。

实验二是针对现有的 Spark SQL 中不同查询方案下的时间性能对比，继而验

证选择 Spark SQL 对于 Parquet 列存储文件进行查询时能够带来查询性能的优化。目前，Spark SQL 在对数据集进行查询时候有三种方案，通俗来说是：（a）直接针对 Postgresql 操作，这种情况下系统需要安装数据库；（b）对 HDFS 文件系统上的源数据集进行操作；（c）对 Parquet 数据集文件进行操作。按照此设计实验并进行测试，实验二的测试结果如下图 5-3 所示。

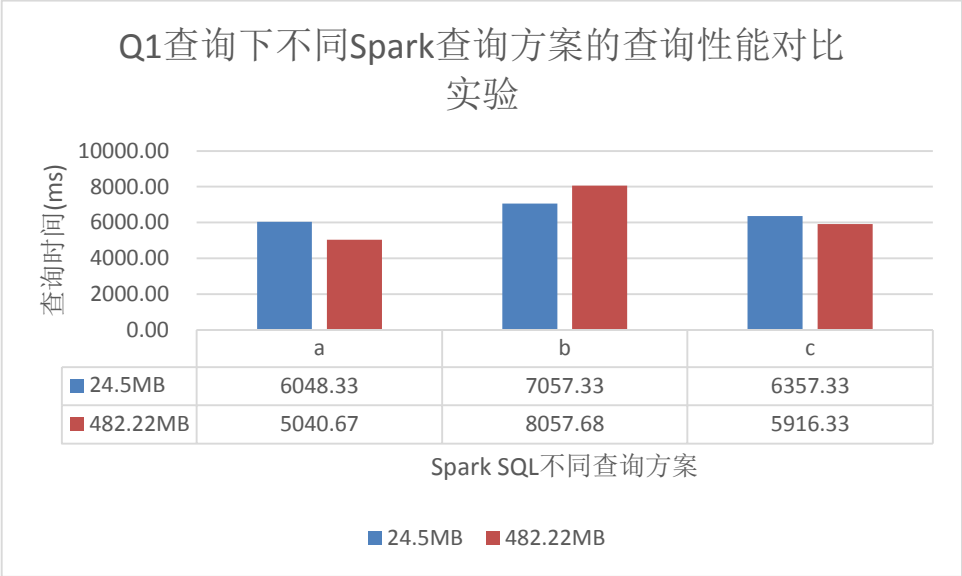


图 5-3 Q1 查询下不同 Spark 查询方案的查询性能对比实验

前文 3.4.2 节中详细介绍了 Spark SQL 在时序大数据场景下的三种查询方案的步骤与特性，对于（a）方案，尽管在小数据下查询效率高，在数据量增加时和（c）方案在 Q1 下查询时间差不多，但是需要提前安装 Postgresql 软件，不符合大数据下的应用场景。对于（b）方案操作下，（c）方案在不同数据规模下查询性能更高，（b）使用了 Parquet 列存储转化功能，在数据存储上有一定优越性。

5.4 本章小结

本章主要通过设计实验来对比本文提出并实现的基于 Spark 的时序大数据分析方案的有效性，同时测试针对不确定属性数据集设计的 Spark SQL 与 jdbc 进行数据查询时的性能。本文首先给出了实验设计，包括集群配置、数据集以及对比方案等。然后从数据存储的大小、数据查询性能测试实验等两个方面来评估本文研究内容的有效性。

第6章　总结与展望

本章总结了针对 Zenvisage 平台在其应用场景下进行大数据分析的瓶颈，设计出的基于 Spark 的时序大数据分析方案，并将其集成于 Zenvisage 的可视化分析系统底层中；同时基于目前研究工作中存在的不足，提出本课题下一步的改进方案和工作展望。

6.1 工作总结

本课题是“时序大数据可视化分析平台研究与设计”，主要研究了时序大数据可视化分析的需求，分析了现有相关工作的不足，总结出现有的可视化分析工具操作繁琐、重复、耗时，指定可视化趋势困难，以及在时序大数据场景下存储和查询性能下降时的问题。为此，我们借助开源的 Zenvisage 可视化分析平台，提出了基于 Spark 的时序大数据分析方案，并将其集成于 Zenvisage 的可视化分析平台底层中，通过系统测试和实验结果分析验证了基于 Spark 的时序大数据分析这一方案在时序大数据场景下存储和查询性能的优化，以及其在 Zenvisage 平台下的可用性和有效性。本文的主要研究工作包括以下几个方面：

（1）针对时序大数据可视化分析的需求，分析出现有的可视化分析工具操作繁琐、重复、耗时，指定可视化趋势困难。

本文首先从技术方法这一角度研究现有的可视化分析相关工作，发现现有可视化分析工具尽管利用了用户的偏好设置，支持用户修改查询条件，却忽略了在充分提高用户体验时加重了用户的检查、试错工作。为了在数据集中找到所需的期望，数据人员可能需要密集型地手工检查集合中的每个可视化，这是一个繁琐、重复、耗时的循环过程；同时，在指定可视化趋势时底层需要复杂的查询，相当困难。

（2）分析了 Zenvisage 平台在大规模时序数据的应用场景下存储和查询性能下降的问题，找到传统关系型数据库在大数据分析中的瓶颈。

在 Zenvisage 对时序数据的分析场景下，对于数据存储和查询这两方面它还在采用行存储的传统关系型数据库，达不到海量时序数据分析场景下性能和运行效率的需求，且造成的 I/O 操作、网络带宽较大。

（3）针对这些瓶颈设计并实现基于 Spark 的时序大数据分析方案。

本文提出基于Spark的时序大数据分析这一方案,即通过HDFS列存储技术、Spark SQL 查询性能实现海量时序数据存储和查询的策略。该方案具体包括根据不同数据集设计合适的数据模型,在系统已有数据集或用户上传数据集时实现在HDFS 文件系统上进行 Parquet 列存储，并实现业务上的 Spark SQL 查询接口。

（4）将设计出的基于 Spark 的时序大数据分析方案集成于 Zenvisage 的可视化分析系统底层中。

在将本文设计出的基于 Spark 的时序大数据分析方案集成于 Zenvisage 的可视化分析系统底层中，理解 Zenvisage 平台的所有业务逻辑，并将底层查询接口与 Spark SQL 查询接口对接，并在应用程序基础上实现查询结果缓存等优化功能。

（5）通过对比实验验证本文设计出的基于 Spark 的时序大数据分析方案的可用性和有效性。

本文通过对比实验验从数据存储和查询这两方面验证本文研究方案同传统关系型数据库的性能对比,从实验结果可以看出这两方面性能的大大提升。同时，对于 Spark SQL 在时序大数据场景下的三种查询方案进行实验对比，得出本文选取的查询方案的优越性，并分析了选取方案优越的原因及其影响因素。

此外,本系统中基于 Spark 的时序大数据分析方案的设计原理还可应用其他HDFS 列存储引擎中如 ORC 中，且上层的 SQL on Hadoop 系统也可随意更换（如 Presto、Hive 等均可），具有良好的可移植性。

6.2 未来展望

在长时间的研究和实验工作中，我们发现目前可视化分析系统上的“有价值性”视图仍有许多可以思考的地方。通过对比其他交互式的可视化分析工具和大量的实验结果分析，下一步的改进方案主要从以下几个方面：

1. 本文借助 Zenvisage 平台，其可视化分析系统设计方案对单个数据集视图进行查询支持较好，但不支持数据集本身的连接、更新进行操作，导致数据集需要重新导入或数据模型的设计，针对此问题后期通过增加接口，在存储数据前提供用户对数据集属性上的筛选和对不同数据集的连接操作,一方面可以减少存

储时间和文件大小，另一个方面支持更多的数据集操作；同时，在视图代数的表达范围上设计更多的功能，以便于符合更多的实际应用场景。

2. 本文通过相似性度量作为视图展示，对于可视化分析上还有更多视图特征需要提取，也希望在后期研究中能够对这一方面做更加精细的规划和学习；同时，对于系统底层基于 Spark 的时序大数据分析上的设计还需要进一步进行优化，以便在后期使用上达到更好的使用效果。

3. 通过视图趋势的相似性度量是本文进行可视化分析的一个比较“有趣”的地方，通过视图相似或不相似等进行视图的排序，找到适合用户的数据分析期望。本系统底层通过欧氏距离、时间规整算法（DTW）[30]等进行设计，后期考虑其他相似性度量算法进行视图分析，以更好地实现用户不同度量方式的系统功能完整性。

参考文献

[1] Tableau public (www.tableaupublic.com/). [Online; accessed 3-March-2014].

[2] Spotfire, http://spotfire.com. [Online; accessed 17-Aug-2015].

[3] Tableau valuation. http://www.sramanamitra.com/2015/01/12/billiondollar-unicorns-tableaus-valuation-increases-to-6-billion/.

[4] Vartak M, Rahman S, Madden S, et al. SeeDB: Efficient Data-Driven Visualization Recommendations to Support Visual Analytics[C]// Proceedings VLDB Endowment, 2015:2182.

[5] Siddiqui T, Kim A, Lee J, et al. Effortless data exploration with zenvisage: an expressive and interactive visual analytics system[J]. Proceedings of the Vldb Endowment, 2016, 10(4):457-468.

[6] Vavilapalli V K, Murthy A C, Douglas C, et al. Apache Hadoop YARN: yet another resource negotiator[C]// Symposium on Cloud Computing. ACM, 2013:5.

[7] 杜小勇，陈跃国，覃雄派. 大数据与 OLAP 系统[J]. 大数据，2015(1):48-60.

[8] Parquet:http://parquet.apache.org/

[9] ORC:http://orc.apache.org/

[10] Abadi D J, Madden S R, Hachem N. Column-stores vs. row-stores:how different are they really?[C]// ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, Bc, Canada, June. DBLP, 2008:967-980.

[11] 陈明. 大数据可视化分析[J]. 计算机教育，2015(5):94-97.

[12] Mackinlay J, Hanrahan P, Stolte C. Show me: automatic presentation for visual analysis.[J]. IEEE Transactions on Visualization & Computer Graphics, 2007, 13(6):1137.

[13] Gonzalez H, Halevy A Y, Jensen C S, et al. Google fusion tables: web-centered data management and collaboration[C]// ACM SIGMOD International Conference on Management of Data. ACM, 2010:1061-1066.

[14] Wongsuphasawat K, Moritz D, Anand A, et al. Voyager: Exploratory Analysis via Faceted Browsing of Visualization Recommendations[J]. IEEE Transactions on Visualization & Computer Graphics, 2016, 22(1):649-58.

[15] Parameswaran A, Polyzotis N, Garcia-Molina H. SeeDB: visualizing database queries efficiently[M]. VLDB Endowment, 2013.

[16] Siddiqui T, Kim A, Lee J, et al. zenvisage: Effortless Visual Data Exploration[J]. Proceedings of the Vldb Endowment, 2016, 10(4):457-468.

[17] Shvachko K, Kuang H, Radia S, et al. The Hadoop Distributed File System[C]// MASS Storage Systems and Technologies. IEEE, 2010:1-10.

[18] Hive: https://hive.apache.org/

[19] Spark SQL: http://spark.apache.org/sql/

[20] Presto: https://prestodb.io/

[21] AngularJS: https://angularjs.org/

[22] Darwin P B, Kozlowski P. AngularJS web application development[J]. 2013.

[23] Doan D H. "Apache Zeppelin, the missing GUI for your Big Data back-end"[J]. 2014.

[24] Spark SQL: http://spark.apache.org/docs/latest/sql-programming-guide.html.

[25] RDD: http://www.infoq.com/cn/articles/spark-core-rdd/.

[26] Borthakur, Dhruba, et al. "Apache hadoop goes realtime at Facebook."*ACM SIGMOD International Conference on Management of Data, SIGMOD 2011, Athens, Greece, June* DBLP, 2011:1071-1080.

[27] 郭庆燕，张敏，杨贤栋. JQuery Ajax 异步处理 JSON 数据实现气象图片的显示[J]. 计算机应用与软件，2016，33(6):20-22.

[28] 何龙，陈晋川，杜小勇. 一种面向 HDFS 的多层索引技术[J]. 软件学报，2017, 28(3):502-513.

[29] Richter S, Quian, Ruiz J A, et al. Towards zero-overhead static and

adaptive indexing in Hadoop[J]. The VLDB Journal, 2014, 23(3):469-494.

[30] 侯靖勇，谢磊，杨鹏, 等. 基于 DTW 的语音关键词检出[J]. 清华大学学报(自然科学版)，2017(1):18-23.

致谢

光阴似箭，时光飞逝。转眼间，在河海大学物联网工程学院的四年本科生活也将面临尾声了。毕业临近，回想起 2013 年第一次以学生的身份走入河海园，第一次和同学们走入厚德楼一起上课，第一次和导师在实验室见面，第一次……这些种只属于河海园的第一次仿佛就在昨天，此刻坐在实验室写着这篇“致谢”的我，心中被这最真切的感受满溢——感恩。

首先我要感谢我的导师牟艳副教授，她深厚的专业知识，严谨的科研态度，幽默风趣的人格魅力和诲人不倦的高尚师德对我影响深远。我这四年之中最大的收获是，进入了河海大学体育信息实验室，接触了大型体育运动会赛事、信息系统开发与实施、流媒体技术开发等，这些我指明了未来的方向，这些收获都是在牟老师的带领和指导下得来的。感谢牟老师根据我的兴趣和对未来的规划为我提供了对我最有助益的研究方向，把我引向不断学习、不断科研的道路；感谢牟老师为我提供参加科研项目比赛和参与实际项目开发的锻炼机会，使本人的系统开发能力和科研都得到了大幅度提升；感谢牟老师在我校外保研的过程中提供建设性地建议，并支持我进行校外毕设；感谢牟老师在我科研碰到困难，失去信心的时候予以及鼓励和帮助，使我的科研工作得以顺利地进行下去。牟老师能成为我的导师，是我此生莫大的荣幸。

同时，我要感谢实验室的陈鹏老师，他是我们实验室的动力航帆。他开阔的视野，独到的科研视角，不论是在帮助同学搞科研方面还是在做企业项目开发上都兢兢业业；感谢实验室丁波老师在运动会项目实施上给予的帮助，印象最深的是在第一届全国青年运动会三明乒乓球场馆中，丁老师认真仔细的赛场裁判精神让我敬佩；感谢实验室中缪刚、庄亚军、张文等众师兄师姐的陪伴，与你们一起科研和编码的时光，成为我一生难忘的回忆；感谢陈慧萍老师在本科四年中作为班主任热情的帮助，以及在计算机专业技能上的教学，同时在各项国家级项目比赛和此次的校外毕设上，陈老师也是耐心指导和安排工作。希望，所有的老师在未来工作和生活中，一切顺利。

当然，在此次校外毕设中也要感谢企业导师陈跃国副教授，他是我今后在中国人民大学研究生阶段的导师。此次校外毕设在人大的生活和学习过程中，感谢

陈老师对我的关系和在科研上的指导，给我在科研上提供建设性建议，并把我引入大数据开发的道路。感谢师兄卞昊穹、金国栋、马登豪、李博放、毛文祥等对我毕设工作上无私的帮助，感谢师姐张香玲、赵丽萍等对我在论文撰写上耐心的指导。一起校外毕设的未来队友们陈成、邵明锐、韩涵、黄文韬、程一舰，感谢与你们一起快乐的时光。感谢人大直博生赵衍衍师兄，是你在人大提供我住宿，并对我的毕设系统研发进行建议，让我在外校也体会到了温暖。

感谢潘翰廷、朱琦、吴龙影等我的朋友们，是你让在科研之外体会了更多精彩生活。感谢邓子越、赵守月等我本科的学长学姐们，是你们为我指明了一条学习的道路，并在专业学习上进行帮助。感谢我的室友们，四年中我们一起欢笑、一起流泪、一起成长，你们早已为我人生中不可或缺的一部分。

感谢我的家人，没有你们就没有我如今的一切，你们是我最大的财富和温暖的归处。我爱你们。

最后感谢母校，感谢河海园成为我学生生涯的不可或缺的一部分。我会继续践行河海精神，努力成为于国于社会有用的人才。