World Journal of Innovation and Modern Technology, Vol. 9, Issue 10 (Oct) 2026
ISSN 2682-5910
DOI: 10.66069/ojspub.26821007

Research and Practice of Kettle in the Field of Distributed Data Warehouses

Fenghua Wu, Chenghui Zhou

The 28th Research Institute of China Electronics Technology Group Corporation, Nanjing 210000, China

Abstract: ***To address the diverse and often conflicting data extraction and loading requirements that arise across heterogeneous business systems—ranging from enterprise resource planning (ERP) and manufacturing execution systems (MES) to customer relationship management (CRM) and supply chain platforms—this paper proposes a Kettle-based implementation scheme for the systematic construction of a distributed data warehouse, grounded in simulated data generated by the industry-standard TPC-DS benchmarking tool. Through a rigorous comparative analysis of three distinct data extraction methodologies—namely manual scripting, direct extraction (which pulls data straight from source systems with minimal intermediate processing), and indirect extraction (which employs staging areas and intermediate transformations prior to loading)—the indirect approach is selected as the optimal strategy due to its superior balance of performance, maintainability, and fault tolerance, particularly when handling the complex, multi-source data landscapes typical of modern enterprises***

Keywords: Kettle; ETL; Distributed Data Warehouse.

1. INTRODUCTION

With the rapid development of the mobile Internet and the rapid integration of informatization and industrialization, massive business data has been generated in an exponential growth manner. How to extract useful data from massive data for big data analysis and artificial intelligence applications has become an important issue today [1] . However, due to data isolation and lack of unified management between business information systems in large enterprises, organizations, and government departments, systems from different manufacturers are incompatible, and different systems cannot interconnect or share data, forming "information silos", which makes data integration and sharing between systems difficult. The goal of data integration is to integrate data from different sources, in different formats, and with different characteristics and attributes. Through data integration, users can access and utilize data more efficiently, avoid repeated data collection and processing, and improve the accuracy and credibility of data analysis. However, traditional data integration technologies have problems such as high complexity, high error proneness, low performance, and poor scalability. Therefore, by designing a Kettle-based distributed data warehouse construction implementation scheme, data integration and transformation tasks can be efficiently completed to support enterprise business operations and decision analysis. This indirect method is then operationalized through Kettle's visual ETL (Extract-Transform-Load) pipeline designer, which enables drag-and-drop construction of data flows, support for over 30 mainstream data sources including relational databases, files, APIs, and Hadoop ecosystems, and extensive transformation capabilities such as format conversion, filtering, merging, and calculated field generation. When combined with operating system scheduled tasks—for instance, cron jobs on Linux or Task Scheduler on Windows—the proposed scheme automates the periodic loading of data from source systems into the distributed data warehouse, effectively dismantling the "information silos" that have long plagued cross-system interoperability and data sharing. Empirical validation from real-world deployments underscores the efficacy of this approach: a manufacturing conglomerate implementing Kettle for production-sales-logistics data integration reported a threefold improvement in business analysis efficiency alongside complete elimination of data silos; an e-commerce enterprise successfully consolidated order data scattered across multiple systems into a unified warehouse, cutting the data warehouse construction cycle by nearly half; and a banking institution adopted Kettle for regulatory reporting data integration, achieving automated synchronization across core business systems, third-party credit interfaces, and external customer behavior data. Furthermore, performance optimization techniques—including concurrent job execution, Kettle memory size adjustment, and the use of custom import plugins—have been shown to boost data migration speeds by 5 to 15 times under comparable environments. The TPC-DS benchmark, which simulates an online analytical processing (OLAP) scenario for a large retailer with 99 distinct analytical tasks across 24 tables spanning 17 dimensions and a data volume of 10 TB, provides a robust testing ground for validating the scheme's scalability and analytical performance. Ultimately, this Kettle-driven, schedule-automated distributed data warehouse construction framework not only resolves the interconnection and data-sharing challenges among

disparate business systems but also delivers a cost-effective, low-code, and maintainable solution for enterprise-wide data integration, thereby furnishing reliable, timely, and actionable service support for organizational analysis and strategic decision-making.

2. OVERVIEW OF KETTLE

Kettle is a free, open-source, visual, easy-to-use, and powerful ETL tool. Written in pure Java, Kettle runs on Windows, Linux, and Unix, enabling simple and efficient data extraction to meet the needs of various scenarios [3]. ETL is a critical for data warehouses to obtain high-quality data; it involves extracting, transforming, cleaning, and loading existing data scattered across various business systems, converting it into useful data required by business intelligence systems [4]. Kettle provides a Java-based visual development environment where ETL solutions can be created by dragging and dropping components and configuring data flows between them. An ETL activity is a quadruple $A = (ID, I, 0, S)$, where ID is the activity identifier, I is the set of input schemas, O is the set of output schemas, and S is one or more extended relational algebra expressions representing the semantics of each output schema [5-6]. KETTLE's ETL activities can be viewed as a directed acyclic graph (DAG), where nodes correspond to individual jobs or transformation steps, and edges represent data supply relationships corresponding to data flow node connections (Hops).

3. TPC-DS OVERVIEW

TPC-DS (Transactional Processing Performance Council Decision Support Benchmark) is a commonly used data warehouse benchmark suite in the big data field, primarily used to measure the performance of data warehouse systems. It simulates the decision support environment of a retail enterprise, including functions such as complex queries, data mining, and business intelligence. When evaluating the performance of production systems, the TPC-DS standard considers data scale, query complexity, number of concurrent users, etc. The TPC-DS benchmark defines 7 fact tables and 17 dimension tables, with an average of 18 columns per dimension table. It uses 99 complex SQL query cases, adhering to SQL99 and SQL2003 standards, with a significant increase in the number and difficulty of SQL queries [7-8]. The benchmark abstracts the diverse operations that real decision support systems may face while retaining essential performance characteristics, enabling both short-term market decision-making for enterprises and predicting potential future issues to provide recommendations [9].

The test data generation process of the TPC-DS tool is as follows:

(1) Tool Download.

Download the required version of the TPC-DS source code directly from the official website.

(2) Compilation and Authorization.

Upload the source code package to the server, extract it, authorize it using operating system commands, then navigate to the tools directory of the source code package for compilation. If compilation errors occur, manually mount the software source, install dependency packages, and then retry the compilation process.

(3) DDL Generation.

After successful compilation, DDL statements for 24 tables will be generated in the tools directory.

(4) Test data generation.

Switch to tools, set data scale parameters and concurrency parameters, execute the dsdgen executable program, and generate a dataset of the corresponding scale.

(5) Simulated data construction.

In the test database MariaDB, use DDL statements to create tables, then import the generated data into the database to simulate the scenario where real business data is stored in a relational database.

4. DATA EXTRACTION AND LOADING

4.1 Manual Data Extraction

Take extracting the TPC-DS dataset stored in the MariaDB database into a distributed data warehouse as an example. Use manual data extraction to complete the construction of the distributed data warehouse. Taking the call_center table as an example, the specific method is as follows:

(1) Data export. Use MariaDB's export command to export data to the local machine.

MariaDB [tpcds]> select * from tpcds.call_ center into outfile '/opt/mariadb_output/call_ center.csv';

(2) Upload files to the distributed file system.

[hdfs@host76 ~]$ hadoop fs -mkdir /tmp/ tpcds/

[hdfs@host76 ~]$ hadoop fs -chmod -R 755 / tmp/tpcds/

[hdfs@host76 ~]$ hadoop fs -put /opt/ mariadb_output/call_center.csv /tmp/tpcds/

(3) Create a table in hive.

Refer to the DDL statements in tpcds.sql generated under the TPC-DS tool tools to complete the table creation.

(4) Import data. Log in to hive, enter the specified database, and execute the import statement.

[hdfs@host76 root]$ beeline

0: jdbc:hive2://host76:2181/default> use tpcds;

0: jdbc:hive2://host76:2181/tpcds> load data inpath '/tmp/tpcds/call_center.csv' into table call_center;

The above method steps can extract data from a relational database into a distributed data warehouse, but this method requires manual operation throughout and cannot meet the application scenarios of automated data extraction, especially when the data in the business system changes at any time, which may require periodic data extraction.

4.2 Direct Data Extraction

Considering that the manual data extraction method cannot adapt to the changing requirements of distributed data warehouse construction, the Kettle tool is used to complete the construction of the distributed data warehouse by adopting the direct data extraction method. The data extraction from MariaDB to Hive is shown in Figure 2 below. Taking the call_center table as an example, the specific method is as follows:

(1) Construction of Data Extraction Process

Open the Kettle tool, create a new transformation, drag the table input and table output to the canvas in sequence and connect them to initially complete the construction of the data extraction process.

(2) Relational Database Settings

Double-click the table input operator, fill in the step name information, then click "New". In the pop-up database connection page, select the MariaDB connection type and fill in the connection parameters. Finally, click "Test". After the test passes, click "Confirm" to complete the database connection creation. Return to the table input operator configuration page, click "Get SQL Query Statement", select the tpcds database and call_center table to automatically generate the query SQL, and finally click "Confirm" to complete the configuration of the table input operator.

(3) Distributed Data Warehouse Settings

Double-click the table output operator, fill in the step name information, then click "New". In the pop-up database connection page, select the Hadoop Hive 2 connection type and fill in the connection parameters. Finally, click "Test". After the test passes, click "Confirm" to complete the distributed data warehouse connection creation. Return to the table output operator configuration page, select the target schema tpcds database and target table call_center, check the database fields, click "Get Fields" to complete the field flow mapping configuration, and finally click "Confirm" to complete the configuration of the table output operator. If the table is not pre-created in the distributed data warehouse, you can click "SQL" and then "Execute" to automatically create the table on the destination side.

(4) Operation of Data Extraction Process

Follow the above steps 1-3 to complete the data extraction process configuration for 24 tables, and finally click "Save and Run" to complete the data extraction from MariaDB to Hive.

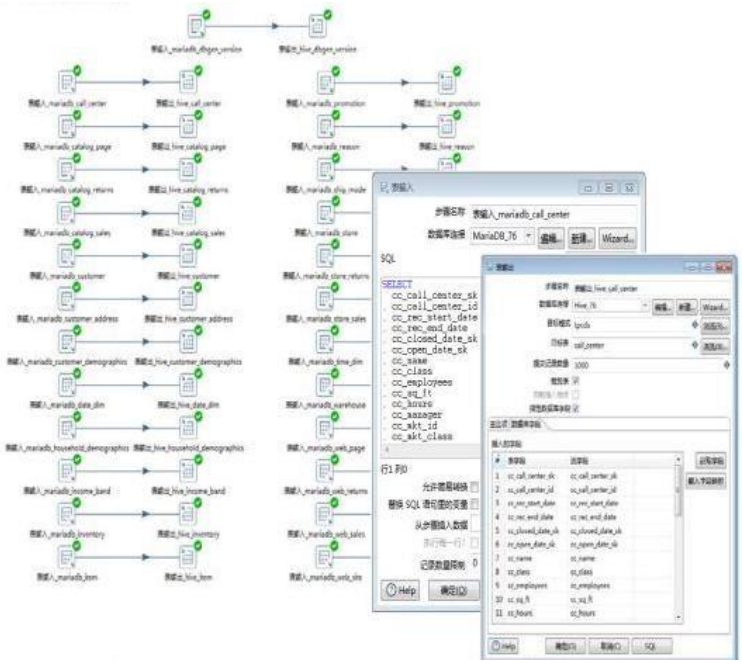


Figure 1: Data Extraction from MariaDB to Hive

The above method steps can extract data from a relational database into a distributed data warehouse. Although this method provides a visual configuration interface, which is convenient, concise and easy to operate, the data extraction process takes a long time to run and has a slow extraction rate, which cannot meet the application scenarios of large-scale datasets.

4.3 Indirect Data Extraction

Considering the problems faced by using manual data extraction or direct data extraction for distributed data warehouse construction, an indirect data extraction method is proposed, which does not require excessive manual intervention, has no slow data extraction efficiency, and can meet the changing requirements of distributed data warehouse construction. Using the Kettle tool, the direct data extraction method is adopted to complete the data migration from MariaDB to HDFS, and then an external table is created on the distributed data warehouse, and the location of the table is pointed to the HDFS storage path to complete the association between data files and the table, thereby realizing the construction of the distributed data warehouse. The data migration from MariaDB to HDFS is shown in Figure 3 below. Taking the call_center table as an example, the specific method is as follows:

(1) Construction of Data Extraction Process

Open the Kettle tool, create a new transformation, drag and drop "Table Input" and "Hadoop File Output" onto the canvas in sequence and connect them to initially complete the construction of the data migration process.

(2) Relational Database Settings
Configure the Table Input operator by referring to Step 2 of direct data extraction.

(3) Distributed File System Settings

Double-click the Hadoop File Output operator, fill in the step name information, then click "New". In the pop-up Hadoop Cluster configuration page, fill in parameters such as HDFS, JobTracker, and Zookeeper. Finally, click "Test"; after the test passes, click "Confirm" to complete the Hadoop connection creation. Return to the Hadoop File Output operator configuration page, select the file storage path, then fill in the corresponding delimiter under the "Content" tab, click "Get Fields" under the "Fields" tab to complete the field flow mapping configuration, and finally click "Confirm" to finish configuring the Hadoop File Output operator.

(4) Data Extraction Process Execution

Follow Steps 1-3 above to configure the data migration process for 24 tables, click "Save and Run" to complete the data migration from MariaDB to HDFS.

(5) Data Binding

In the above steps, the data has been migrated to the /user/Administraor/call_center directory. Refer to the DDL statements in tpcds.sql generated under the tools directory of the TPC-DS tool to create a Hive table and point the storage path to this directory. The specific method is as follows:

0: jdbc:hive2://host76:2181/tpcds>create table catalog_sales (..., cs_item_sk, cs_order_ number, ...) primary key (cs_item_sk, cs_order_ number) row format delimited fields terminated by '|' location 'hdfs://host76:9020/user/Administraor/call_center';

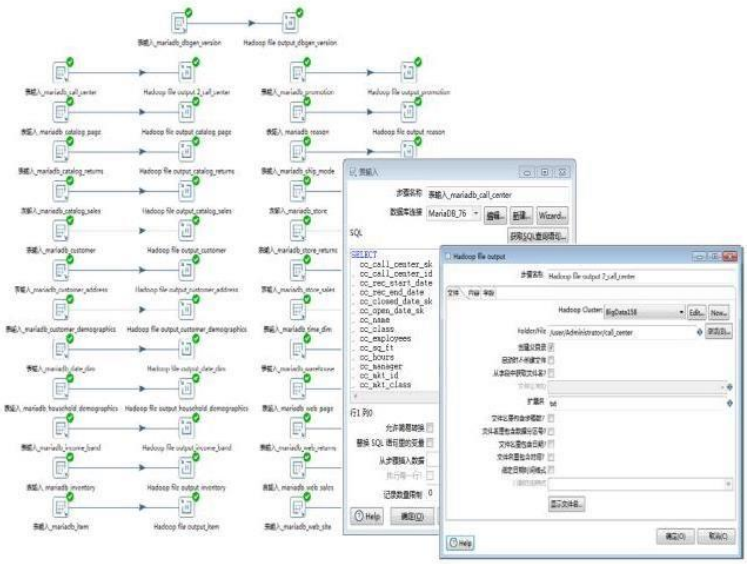


Figure 2: Data Migration from MariaDB to HDFS

4.4 Data Extraction Implementation Plan

Considering practical scenarios, the implementation plan for building a distributed data warehouse based on indirect data extraction still needs to address data extraction in scenarios where business system data changes in real time. For data extraction requirements where business system data changes in real time, the following methods can be used to implement periodic execution of data extraction and transformation tasks.

(1) Kettle Tool Job Process.

Open the Kettle tool, create a new job, drag and drop "Start", "Transformation", and "Success" operators onto the canvas in sequence and connect them. Click the "Start" operator to select periodic configurations with intervals of months, weeks, days, minutes, or seconds. Click the "Transformation" operator, browse and select the indirect data extraction transformation task to complete the construction of the data job process.

(2) Operating System Scheduled Tasks.

Considering the inconvenience of the Kettle tool in periodic execution, periodic execution of data extraction and transformation tasks can be achieved by writing automated execution scripts and scheduling them through the operating system's scheduled tasks.

If the Kettle tool is deployed on a Linux machine, use the operating system's crontab to schedule periodic execution of the script. The start.sh script is as follows.

```
[root@host76 kettle]# cat start.sh
#!/bin/sh
/home/data-integration/kitchen.sh -file=/ home/kettle/mysql_to_hdfs.ktr -level=Detailed
-logfile=/home/kettle/mysql_to_hdfs.log
[root@host76 kettle]# crontab -l
*/1 * * * * * sh /home/kettle/start.sh >> / home/kettle/mysql_to_hdfs_crontab.log
```

If the Kettle tool is deployed on a Windows machine, use the task scheduler自带的 (built-in) in the Windows operating system to schedule the periodic execution of the script. The start.bat script is shown as follows.

```
D: \kettle>type start.bat
D: \data-integration \pan.bat /file D: \kettle \mysql_to_hdfs.ktr /level:Basic > D: \kettle \mysql_to_hdfs.log
```

5. CONCLUSION

Kettle has the advantages of being easy to get started and simple to develop in multi-source heterogeneous data integration and data extraction and loading. Aiming at the application requirements in the field of distributed data warehouses, an implementation scheme for building a distributed data warehouse based on Kettle is designed. Relying on Kettle and TPC-DS tools, this scheme realizes the automatic construction of a distributed data warehouse by analyzing and comparing the differences between manual data extraction, direct data extraction, and indirect data extraction, combined with the operating system's scheduled tasks. It breaks down the information barriers between business systems and can provide service support for enterprise analysis and decision-making. In the future, we will continue to improve the data transformation and cleaning process, enhance data quality, optimize the ETL design process, and improve data update efficiency, laying a data foundation for enterprises' further business intelligence analysis, OLAP analysis, and knowledge discovery.

REFERENCES

[1] Wang Xuesong, Zhang Liangjun. ETL Data Integration and Processing (Kettle) [M]. Beijing: Posts & Telecom Press, 2021, 1-2.
[2] Cui Jidong. Research and Implementation of Data Integration Platform Based on Kettle and Quartz [D]. Zhengzhou University, 2019, 12-13.
[3] Chen Rongxin, Fu Yonggang, Chen Weibin. Business Intelligence System Based on Pentaho [J]. Computer Engineering and Design (9): 263-265.
[4] Zhong Hua, Feng Wenlan, Tan Hongxing, et al. Design and Implementation of ETL System for Data Integration [J]. Computer Science, 2004, 31(9): 87-89.
[5] Wu Yuanhong. Research on Optimization of ETL Execution Process [J]. Computer Science, 2007, 34(1): 81-83.
[6] Cui Youwen, Zhou Jinhai. Research on Business Intelligence of Chinese Herbal Decoction Pieces Enterprises Based on Pentaho[J]. Electronic Design Engineering, 2014, 22(7):12-15.

[7] He Lei. Research and Development of Testing System Based on TPC-DS[D]. Shanghai: Fudan University, 2013.
[8] Niu Yijie, Deng Wu. Design and Implementation of Decision Support Evaluation System[J]. Computer Era, 2007(3):32-33.
[9] Liu Baoxing. Design and Implementation of a Performance Testing Tool Based on TPC-DS [D]. Dalian University of Technology, 2018.