A Hybrid Multilevel Storage Architecture for Electric Power Dispatching Big Data

Hu Yan¹, Bibin Huang², Bowen Hong³ and Jing Hu⁴

New Energy and Statistics Department, State Grid Energy Research Institute(SGERI), Beijing, China
Email: ¹yunhu916@126.com, ²huangbibin@sgeri.sgcc.com.cn, ³hongbowen@sgeri.sgcc.com.cn, ⁴hujing@sgeri.sgcc.com.cn

Abstract. Electric power dispatching is the center of the whole power system. In the long run time, the power dispatching center has accumulated a large amount of data. These data are now stored in different power professional systems and form lots of information isolated islands. Integrating these data and do comprehensive analysis can greatly improve the intelligent level of power dispatching. In this paper, a hybrid multilevel storage architecture for electrical power dispatching big data is proposed. It introduces relational database and NoSQL database to establish a power grid panoramic data center, effectively meet power dispatching big data storage needs, including the unified storage of structured and unstructured data, fast access of massive real-time data, data version management and so on. It can be solid foundation for follow-up depth analysis of power dispatching big data.

1. Introduction
In 1969, Edgar Frank Codd published an important paper, first proposed the concept of relational data model[1]. After that, many commercial companies such as IBM, ORACLE, Sybase, Microsoft and so on designed many corresponding relational database products based on the model, and later open source relational database software such as MySQL, Postgresql were developed. All relational database design meets the four characteristics of ACID [2], namely atomicity, consistency, isolation, persistence. Relational database, as a broad general-purpose database, has the following advantages, such as 1) data consistency; 2) standardization, data update cost is small; 2) support JOIN and other complex queries; 3) many mature technologies and applications.

With the development of the Internet, data is growing rapidly. Fast writing and high concurrent access have become an urgent need, while the relational database can’t well satisfy it for considering more consistency than availability. In order to compensate for the lack of relational database, NoSQL database (Not Only SQL) came into being [3,4,5], it makes a single database distribute to a large number of physical and virtual machines and based on BASE rules, that are basically available, soft state, and ultimately the same. There are mainly two types of NoSQL databases: key-value type and document type. Key-value database organizes data in the Key-Value way and access data through a single key, its representative products are open source HBase, Redis, etc[6,7]. Document database is characterized by a single document to store data, allowing users to access the data through the value, its representative products are Mongdb, CloudDB, and other open source software [8,9].

The application scenarios of Relational database and NoSQL database are different. They are not substitute each other, complement each other. Power dispatching big data includes both structured and unstructured data. Traditionally, the structured power dispatching data is stored by relational database, and the unstructured data is stored in the file [10,11], thus all data are decentralized. Paper[12], presented a
distributed key-value pairs based storage method for historical analog data. Paper[13] proposed a version storage method for the grid model. All these methods only solve part storage problems of power dispatching data.

This paper first gives overview of power dispatching data, covering data composition, data structure, data access requirements and so on. Followed a hybrid multilevel data storage architecture for electrical power dispatching big data is proposed, which integrates the advantages of relational database and NoSQL database. Again, the management of the proposed storage system are described, including cluster monitoring, metadata, authority, log. Then, table design examples based on the storage system are given. Finally, the conclusion is given.

2. Power Dispatching Data Overview
There are many kinds of power dispatching systems, for example, Energy Management System(EMS), Operation Management System(OMS), Production Management System(PMS), Geographic Information System(GIS) and so on. Each of them can generate mass of data. The acquisition cycle covers different time scale, from microseconds (such as: high frequency switch), milliseconds (such as: PMU), minutes (such as: power recovery time), day (such as: power generation plan), year (such as: IT asset life), to decade (such as: OT asset life).

The above data can be divided into basic data and derived data according to the source. The basic data includes grid model data, grid analog data, grid management data (such as equipment ledger), location data (such as GIS data), while the derivative data including operational analysis results, statistical data and so on. These data are currently stored in the relational database, accessed by standard SQL but performance requirements are different, including milliseconds, seconds, minutes, hours, etc. Besides, there is another kind of data, that is unstructured data, such as cim / g (<1M), case (30M), grid comprehensive fault reporting and waveform files, teaching plans (10K-30M), these data need to be version management and screen browsing query requires millisecond response.

With the construction of smart grid, the power dispatching data’s sources, scale, type expand rapidly and to be more rich. Because these data structures and coding are not unified, the amount of data is huge, the current lack of a unified real-time (historical) data center, resulting in interfaces great variety between systems, and even the emergence of information islands. Data Integration becomes an urgent need.

3. Power Dispatching Big Data Hybrid Multilevel Storage Architecture

3.1. Design Aim
The design aim is to establish a power grid panoramic data center by integrating power grid operation data from different electric professional systems based on the unified grid model, seen in Figure.1.

The designed storage system should have follow features:
- Support storage of PB level data (PB level), such as WAMS data points storage, historical sampling data periodic writing.
- Support efficient query and statistics of PB level data, many applications need real-time response.
- Disaster recovery, multiple copies for fault tolerance and automatic recovery mechanism.
- Support distributed computing.
- Support structured data and unstructured data integration storage and management.
- Support version management of files.
- Support SQL-based access.
- Support complex JOIN queries between different tables.
3.2. Architecture Design

At present, the main database are relational database and NoSQL database, single database may not meet all the different storage and management needs of power dispatching applications. This paper builds a hybrid multilevel storage architecture for power dispatching big data, as shown in Figure 2. The whole architecture is composed of data source acquisition layer, unified data warehouse layer, real-time data storage layer and interface access layer.

3.2.1 Data source acquisition layer. This layer is responsible for transferring data of external power systems, such as OMS, PMS, water dispatching system, etc. into the unified data warehouse. All kinds of data are transferred by JDBC/ODBC, communication protocol, message queue, message mail, based on data real-time differences.

3.2.2 Unified data warehouse layer. All power dispatching data are stored in the unified data warehouse, building HDFS and Hive. The warehouse does not consider redundancy, more concerned about the clear data structure, including three data marts:

- Synchronized data mart: all the raw data without ETL will be stored in the synchronization data mart first.
- Unified data chart: application-oriented, stored and organized according to application theme.
- Analysis data chart: statistical analysis results.

The three data charts use ETL to synchronize data.
3.2.3 Real-time data storage layer. The layer mainly stores data related to the online applications. In principle, only the daily data is stored, and the storage rules are as follows:

- Grid model data is stored in MySQL database, sampling and other key-value type data is stored in HBase database, and statistical analysis data is stored in the analytical database ADS.
- MySQL database is the prior storage choice, its reliability is relatively high, and response time can reach millisecond level for millions data;
- HBase database only stores data of the most recent period, for example, one week or one month. All the data is stored in Key-Value pattern, querying or writing is by the key and does not support Join queries. The most recent sampling data and the data needed version management are stored here;
- Analytical database ADS is used for large amount of data which needs second level response, complex analysis and son. It updates data in batches, real-time updates are delayed, and does not support the transaction.

3.2.4 Interface access layer. The upper layer application requires a unified interface to access data, blocking interface differences of different database. Therefore, in the selection of the database as much as possible to choose database which supports the standard SQL or SQL. Relational database support standard SQL; ADS support class SQL; Some NoSQL database does not support, such as HBase, but the data warehouse tool Hive provides a class SQL access interface, and can be integrated with HBase. The interface access layer provides SQL Cli, JDBC, ODBC, API, etc. to access data.

The SQL operation is as follows:

- create /drop: create a new table or drop an existed table.
- select/insert/update/delete with bind: do in batch.
- select/insert/update/delete: do in single row.
• alter: add a column or delete a column or revise a column.
  The unstructured data can be access by HDFS command, including put, get, list and so on.

4. Management of the Hybrid Multilevel Mstorage System

4.1. Cluster Monitoring
The proposed data storage architecture involves multiple clusters, requiring centralized monitoring of the cluster status, monitoring cluster load (including CPU, disk capacity, network status, etc.), node status. Ganglia\cite{13,14} is introduced.

Ganglia is an open source cluster surveillance project initiated by UC Berkeley, which includes gmond, gmetad and a Web front end, primarily for monitoring system performance such as cpu, mem, hard disk utilization, I/O load, network traffic, etc. Though Ganglia software, it is easy to see the working status of each node through the curve and can be used to adjust and allocate system resources, playing an important role in improving the overall performance of the system.

4.2. Metadata Management
Metadata management mainly manage table structure information, the relationship among different table structures, and the table version information. All data structure information is maintained by the metadata management module, which can be modified online and the database table structure is traceable.

The unified data warehouse stores all the power dispatching data, so the metadata management module here is only responsible for managing the data in the unified data warehouse. When a database table is created and modified, the metadata management module is automatically triggered, and the metadata management module manages table structure and table structure version information. Metadata is stored in MySQL.

4.3. Rights Management
HBase authority control is introduced. The scope of authority control includes database level, table level, column cluster level and column level, where EXEC is currently only defined in the code but not used; CREATE and ADMIN only relate to table level and database level, column cluster level and column level only relate to READ and WRITE permissions.

4.4. Log Management
All the database operation and access will have logs, on the one hand, it can be used to find a solution to the problem in the future, on the other hand, it can be as evidence data of security control.

5. Table Design

5.1. Table Design in HBASE
HBase database is very suitable for storing time-scale data. In the EMS, there are two tables named Manalog_SCADA and Mpoint-SCADA which contains import electrical operation information. Every minute millions of data are wrote into the two tables. These data have the feature of structure simple, large volume, query by id and son. In HBase database, the two table can be designed as follows:

| Table 1 Manalog_SCADA Table in HBASE |
|--------------------------------------|
| **Column Name** | **Type** | **Other**            |
| Rowkey          | bigint   | Id+date              |
| Timestamp       | time     | Year:month:hour:minute |
| CF:value        | float    |                      |
| CF:quality      | int      | Stands for the value status |
| CF:code         | string   | Code of electric device |
| CF:name         | string   | Name of electric device |
Table 2 Mpoint_Scada Table In HBASE

| Column Name | Type        | Other                        |
|-------------|-------------|------------------------------|
| Rowkey      | bigint      | Id+date                      |
| Timestamp   | time        | Year:month:hour:minute:second|
| CF:value    | unsigned int| 0 or 1                       |
| CF:quality  | int         | Stands for the value status  |
| CF:code     | string      | Code of electric device      |
| CF:name     | string      | Name of electric device      |

5.2. Table Design in HIVE

Grid model has a lot of structured equipment tables. Table information in a period of time will often change, so there is a need for version management. Hive table has a good ability to support for structured data, and can establish partition on the date, achieving version management. For example, the following BUSBARSECTION table, you can add a column to store time, the other column structure unchanged, each line of data on behalf of the device value at some time. As the date partition, you can quickly find a version of the data.

Table 3 Busbarsection Table In HIVE

| Column Name | Data format | Other         |
|-------------|-------------|---------------|
| DEV_ID      | 117375280623910913 | As old column |
| Time(with date) | 2015-11-11 21:11 | New column   |
| V           | 509.561     | As old column |
| PB          | 20.0        | As old column |
| QB          | 30          | As old column |
| VB          |             | As old column |
| ANG         | 0.0         | As old column |

6. Conclusion

This paper presents and implements a hybrid multilevel data storage architecture for electrical power dispatching big data, which can not only meet the unified storage of structured data and unstructured data in the field of electric power regulation, solving the information island, but also can meet different access performance requirements of real-time or non-real-time applications, effectively response to large data rapid growth of access performance bottlenecks, in addition, the entire storage architecture has the characteristics of flexibility, high scalability, strong robustness. In the future, based on the power dispatching unified storage, more study of power dispatching data applications will be done, achieving from power dispatching data to value conversion.

7. References

[1] Codd EF. “A relational model of data for large shared data banks”, Commun ACM[J], vol.13, pp. 377–387,1970.

[2] Carlo Zaniolo. “Analysis and design of relational schemata for database systems”[C], international symposium on physical design, University of California, Los Angeles,1976, pp.226-230.

[3] Jaroslav Pokorny. “NoSQL databases: a step to database scalability in web environment”[J], international Journal of Web Information Systems, vol.9, pp. 69–82,2013.

[4] Chang F, Dean J, Ghemawat S, et al. Bigtable: a distributed storage system for structured data[C], ACM Transactions on Computer Systems (TOCS). New York, USA: ACM, 2008:4.

[5] Borthakur D. HDFS architecture guide[J]. Hadoop Apache Project. http://hadoop. apache.org/common/docs/current/hdfs_design. pdf, 2008.

[6] Facebook. Hbase: Bigtable-like structured storage for Hadoop
[7] Baron C. “NoSQL Key-Value DBs Riak and Redis”, Database Systems Journal, 2015, vol.5, issue 4.
[8] Harley Vera Olivera, Maristela Holanda, Valeria Guimarães, et al. “Data Modeling for NoSQL Document-Oriented Databases”, CEUR Workshop Proceedings, 2015, vol. 1478, pp.129-135
[9] Kyle Banker. “MongoDB in Action”, Geography (2011) p. 520.
[10] XIE Shujing, WANG Hao, ZHANG Zhiwei. Discussions about Power Network Modeling Based on CIM [J]. Power System and Clean Energy, 2011, 27(2) :37–41.
[11] YAN Hu, ZHOU Wei, LI Lixin, et al. “Database Platform of EMS System Based on Distributed Key-Value Pair Storage Technology”[J]. Power System Technology, 2012, 36(9) :162–166.
[12] YAN Hu, LI Lixin, YUAN Rongchang, et al. “Study on Multi-dimensional grid model of integrated storage and management”[J]. Automation of Electric Power System, 2014, 38(16) :94–98.
[13] M. L. Massie, B. N. Chun, and D. E. Culler. The Ganglia Distributed Monitoring System: Design, Implementation, and Experience. Parallel Computing, 30(7):817–840, 2004.
[14] Ganglia Monitoring System, http://ganglia.info.
[15] Solon Barocas, Andrew D Selbst. “Big Data's Disparate Impact”[J], California Law Review, 2016, 104(3) :671.
[16] Kaile Zhou, Chao Fu, Shanlin Yang. “Big data driven smart energy management: From big data to big insights”[J], Renewable & Sustainable Energy Reviews, 2016, 56:215-225.
[17] Yong Chen, Hong Chen, Anjee Gorkhali Kaile Zhou, et al. “Big data analytics and big data science: a survey”[J], Journal of Management Analytics, 2016, 3:1-42.
[18] Paolo Atzeni, Francesca Bugiotti, Luca Cabibbo, et al. “Data Modeling in the NoSQL World”[J], Computer Standards & Interfaces, 2016.
[19] Diego Pasqualin, Giovanni Souza, Eduardo Luis Buratti, et al. “A Case Study of the Aggregation Query Model in Read-Mostly NoSQL Document Stores”[C], international database engineering and applications symposium, Montreal, Canada, 2016, pp.224-229.
[20] Manar Abourezq, Abdellah Idrissi. “Database-as-a-Service for Big Data: An Overview”[J], International Journal of Advanced Computer Science and Applications, 2016, 7(1) :151-157.