ATLANTIS PRESS

Advances in Engineering Research, volume 139

2nd International Conference on Electrical Engineering and Automation (ICEEA 2018)

Application of SQLite Database in Dispatch and Substation Integrated System

Haoqiu Shi*, Xuechun Ji and Hao Li

Nari Group Corporation/State Grid Electric Power Research Institute, Nanjing, China

*Corresponding author

***Abstract*—Substation monitoring and control system generally consists of several unsupervised PC units. Such configuration is not suitable for running large commercial database, so the integration of dispatching system and transformer substation requires a lightweight relational database, and has a strict need for the stability and reliability of data. Taking features of SQLite and system requirements into consideration, the paper introduces the SQLite service currently used at substations, and also describes its components and key mechanism.**

Keywords—SQLite; lightweight; distributed service; dispatch and substation integrated system

I. INTRODUCTION

In accordance with the general strategic plan of SGCC's intelligent substation construction, 6100 intelligent substations will be built during the Twelfth Five-Year period. By using the integrated information platform which connects to all key equipment, the intelligent substation builds an information resource sharing system covering all sub-systems and processes of itself. The dispatch and substation integrated system (DSIS) completes data integration and data standardization at the substation, creating the foundation for local features of the substation. Therefore, the intelligent substation system at the substation side must effectively support the demands of data storage, date query and date modification, to ensure the safety and reliability of the model data and historical data.

In order to achieve the coordination and interaction between the dispatching system and substation system, the design of the substation system is based on the platform of the technical support system for intelligent grid dispatching. In the dispatching system, data is stored in commercial database products, which are large and expensive, but the substation system is more suitable for a few workstations because of its small scale, so it is particularly important to choose a lightweight and stable database.

This article introduces the advantages of SQLite, and analyzes its processes and logic in the operating environment combined with the functions and services encapsulated on SQLite. A lightweight relational database service is described.

II. A BRIEF OF SQLITE

SQLite is a lightweight open source database, which has low resource usage, is easy to configure and manage, and is widely deployed in a variety of lightweight applications. Its database server and client run in the same process, which can reduce the consumption of network access, and can simplify database management, while SQLite also supports transaction processing. The advantages of SQLite in substation systems are as follows:

- Zero-configuration.
- Disk-based storage for easy use.
- SQLite supports Windows, Linux and commercial Unix systems such as Solaris, HPUX, and AIX, while its data files can be used across platforms.
- Database file size up to 2TB.
- Faster than some popular database in most ordinary database operation.
- Provides a simple, easy API for users to use; supports C/C + +.
- SQLite requires no authorization. For commercial use, SQLite does not have any legal limitations.

In summary, SQLite is suitable for the DSIS in many aspects. However, there are also some differences between SQLite and commercial databases currently used in the dispatching system, so it is necessary to design carefully to ensure its stability and efficiency in the substation system.

III. SQLITE SERVICE

In the DSIS, SQLite is mainly used to store data such as grid equipment, parameters, static topology connection, system configuration, alarm and event recording, historical statistics, etc. Relational database services are a set of database management modules which are connected by using service bus of the platform, providing a common set of data access interface, so the applications in DSIS can use the interface to access data in the database Because the DSIS is distributed, relational database services are deployed on several physical servers, and the consistency of data on these servers is greatly required. Database services also are responsible for monitoring SQLite files on their machines, in order to detect anomaly. But as a lightweight relational database, SQLite does not provide any network access interface.

According to the needs of relational database services, the SQLite database service process, also referred to as SQLite Server, is located on each SQLite database server. A set of

interface that can finish communications between user processes and SQLite Server is called SQLite Client. The SQLite service mechanism is formed by SQLite Servers and SQLite Clients. For the DSIS, maintaining the reliability and security of data is also important, so the SQLite service needs to provide complete disaster recovery and troubleshooting capabilities. In order to achieve these requirements, the SQLite service includes the following contents:

- Monitoring the SQLite database status, and providing network access to SQLite database.
- Monitoring the SQLite service status, and dealing with anomaly.
- Deployed on multiple servers, using a master-standby mechanism to achieve service redundancy.
- Providing a synchronization mechanism to ensure data consistency.
- Providing data file backup/dump tools.

A. Database Service Monitoring

Database service monitoring consists of two parts: SQLite Server monitoring and SQLite data file monitoring.

1) SQLite server monitoring: A shell script will monitor the running status of SQLite Server process, and check if ports can be connected, etc. When an exception is found, the shell can restart the stopped service process, and kill and start the abnormal service process. If the SQLite Server has a serious exception, the shell can immediately switch the stand-by SQLite Server with the highest priority to master, and switch the original server to stand-by and mark it as faulty state. This shell script in the Linux system will use inittab to keep running.

2) SQLite data file monitoring: SQLite Server monitors the SQLite data file. It checks if data file is damaged, the I/O of data file, or whether SQLs can be executed and so on. SQLite Server will also provide anomaly switching.

B. Network Accessing

Network accessing is formed by servers and clients. As shown in Figure I, the SQLite Server listens to a certain port. The SQLite client receives the call from the user process and sends the request to the specified port of the specified machine.

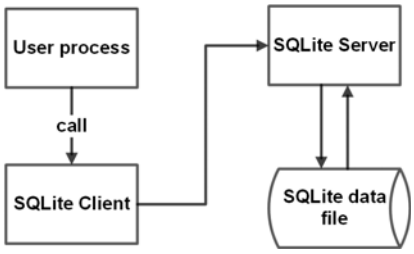

FIGURE I. PROCESS OF NETWORK ACCESSING

A complete process of database access request is shown below:

1) The SQLite client sends a request to the SQLite server.

The SQLite client first obtains the host name where the master SQLite Server is in the system, and then serializes the calling parameters to form a fixed format network transmission message. After this, the client sends the request message to the port of the host where the master SQLite Server is located.

2) SQLite Server deals with the request message.

SQLite Server gets the request message and deserializes the calling parameters, such as SQLs. Then the server executes these SQLs on the database on this server, to collect execution results and error messages (if any). Finally, SQLite Server serializes the results and error messages and sends it back to the client host.

3) SQLite Server returns calling result.

The SQLite client gets the messages returned by the SQLite Server, analyzes the execution result and the error messages, and feeds it back to the user process.

C. Redundancy and Synchronization

As shown in Figure II, a set of SQLite Servers, which is in the form of one master and multiple standbys, implements the redundant mechanism, and the master and standby services follow different logic.

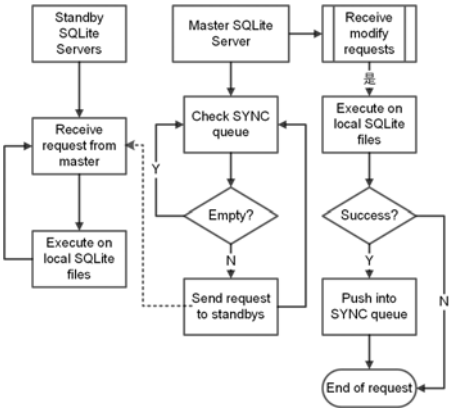

FIGURE II. PROCESS OF MASTER-TO-STANDBY DATA REPLICATION

When receiving a database modification request, the master SQLite Server creates a separate thread specifically responsible for processing the request. First it checks the validity of the request, then, if passed, the SQLs will be executed on database and get a result. If any exception or error occurred, the error message will be returned to the user process; if there is no error, the request will be pushed into a queue for synchronizing. Meanwhile, a thread in SQLite Server is responsible for checking the queue, popping the element (a request) and sending the request to all standby SQLite Servers.

All standby SQLite Servers receive the synchronization request sent by master SQLite Server and execute it in the SQLite databases on their hosts. This is to ensure the validation of a database modification request, but also to ensure that the databases on master and standby SQLite Servers are strictly the same.

SQLite Server also provides data file dumping, used for recovering damaged data files. When database service

monitoring finds that the SQLite data file is corrupted, the data file is dumped from the current master SQLite Server machine to local host, replacing the old file. The data file dumping is formed by full dumping and incremental dumping. The former just copies the entire SQLite data file, while the latter will first do comparing, and only dump the differences, to reduce the network traffic and time used for copying files. The dumping works not only in anomaly, but also in timing mode: checking each SQLite data file regularly, finding differences between the most reliable data files in the system, replacing other files, and backing up.

The redundancy and synchronization meet, to the greatest extent, both the needs of database redundancy and the consistency and security of data in a multi-database model.

IV. CONCLUSION

This paper introduces the application of SQLite database in DSIS, analyzes the features of SQLite, and describes the implementation of many key mechanisms. SQLite database services has been actually used in DSIS by this time: it has a good performance in resource occupancy, and its stability and reliability has also been practically tested in the substation, providing a solid guarantee for database access.

In the gradual deployment of DSIS, many new features of SQLite service are still being discussed, such as triggers in Oracle, logic backup and recovering tools, and so on. These are the key features of current commercial database products, and have reference value and guidance significance to improve our SQLite service. To make it happen, we must study and explore SQLite at a much deeper level.

ACKNOWLEDGMENTS

This work is supported by the SGCC Science and Technology Program “Research on the Key Technologies of the Distributed Data Management in the Dispatching and Control System Platform of Physical Distribution and Logical Integration”.

REFERENCES

[1] SQLite. http://www.sqlite.org/.

[2] Mike Owens, “The Definitive Guide to SQLite,” Apress, 2006

[3] CHUNYUE BI. “Research and Application of SQLite Embedded Database Technology”, WSEAS TRANSACTIONS on COMPUTERS, volume 8, January 2009

[4] Lv Junyan, Xu Shiguo, and Li Yijie, “Application Research of Embedded Database SQLite”, Information Technology and Applications, 2009. IFITA '09. International Forum on

[5] CHEN Xiao-gang, ZHEN Yi-jun, “Application of SQLite embedded database in PMU”, Journal of Mechanical & Electrical Engineering JidianGongcheng, volumn 3, 2008

[6] Song Jinyu, Wei Shanshan, and Yue Kang, “Research and Improvement of Locking Mechanism in Embedded Database Based on SQLite”, Journal of Computer Research and Development, 47, 2010

[7] Z Zhuang, Y Xue, and R Lian, “Application of Embedded Database SQLite for Remote Monitoring System”, Modern Electronics Technique, 2007-08

[8] WANG Bo, GE Zhao-qiang, and ZHU Sheng, “Encapsulation and Utilization of Dispatching System Application Functions in East China Grid”, East China Electric Power, vol.39, no.8, 2011