ISSN (e): 2250 – 3005 || Volume, 06 || Issue, 05||May – 2016 ||
International Journal of Computational Engineering Research (IJCER)

Database Management in Different Applications of IOT

Shona M
Assistant professor, Department of CSE,SVCE, Bangalore, India

ABSTRACT
In the recent years, the Internet of Things (IoT) is considered as a part of the Internet of future and makes it possible for connecting various smart objects together through the Internet. The use of IoT technology in applications has spurred the increase of real-time data, which makes the information storage and accessing more difficult and challenging. This paper discusses the different Databases used for different applications in IOT.

Keywords : *NoSQL, Hadoop, Time Series Daemon, (TSD), MemSQL, MongoDB*

I. INTRODUCTION

A network comprised of physical objects capable of gathering and sharing electronic information. The Internet of Things includes a wide variety of "smart" devices, from industrial machines that transmit data about the production process to sensors that track information about the human body. Often, these devices use Internet Protocol (IP), the same protocol that identifies computers over the World Wide Web and allows them to communicate with one another.

The connection of physical things to the Internet makes it possible to access remote sensor data and to control the physical world from a distance. All of these things are creating a "perfect storm" for the IoT. It is estimated that by 2020 there will be over 25 billion devices wirelessly connected to the Internet of Things, including embedded and wearable computing devices. At the same time, IOT imposes fewer data quality and integrity constraints. Although IoT sensors generate data rapidly, they do not entail the same kinds of transactions one finds in traditional enterprise business applications.

Relational databases (RDBs) work well for many scenarios, but this is not one of them. RDBs were designed for processing structured, highly uniform data sets, but with IOT, gathered data is nothing close to uniform. With over 50 billion objects predicted to be connected to a single network by 2020, the variety of transmitted data already ranges from simple text to a complex combination of information from different sensors. Not only does information need to be transmitted, but it also must be analyzed and calculated. Basically, necessary requirements can be divided into three categories:

Diversity of information and sensors – More and more heterogeneous data is generated by an exponentially growing number of diverse sensors and devices. In addition, new sources of data are constantly added, and the structure and scale of that data is always different and complex.

Extensive and flexible systems – The systems that are ruling the Internet of Things have to be flexible and agile so there won't be need to rebuild an application when new sensors and devices are being added.

Proficient analytics – Previously, simpler systems communicated using alerts and notifications, where information had to be transmitted between two machines, but in the Internet of Things, analytics is the foundation of the system. And for different types of data, there are different analytical mechanisms.

In the Internet of Things, data analysis will require multiple analytical approaches, and, in some cases, significant value is achieved from real-time data analytics or from historical analysis for predictive maintenance services.As we now know, the IoT generates essentially more data to be stored and processed, and even cloud solutions typically don't meet sufficient capacity. This is happening due to the nature of cloud-based services: they usually store all data on a single server, while IOT requires more distributed databases.

II. REQUIREMENTS OF IOT

The requirements of IOT fall into three general categories and virtually all applications will require that at least two are satisfied by your database platform simultaneously:

1. Continuous machine-scale ingestion, indexing, and storage. A modest data source may generate millions of complex records per second on a continuous basis.

2. Operational ("real-time") queries and analytics. Extracting value from IOT data is all about minimizing the latency from data ingestion to online queries and actionable analytics. For many applications, the value of the data is highly perishable, with an exponential decay on timeframes measured in seconds. IOT queries and analytics are rarely summarizations. Stream processing rarely works, you need to support ad hoc queries in something like SQL.
3. IoT data is all about spatiotemporal relationships and join operations. To support the speed and scale of the first two bullets this means you need at least a true time-series database for very simple uses and a true spatial database for the more general case. Spatiotemporal (or just temporal) must be a fundamental organizing principle of the database internals or it will not scale; you cannot modify a text-and-numbers database with extensions for this purpose.

There are, in practice, two types of databases: relational and non-relational (better known as NoSQL). There are pros and cons to each.

Relational Database

The relational model organizes data into multiple tables and assigns a value to attributes in each row and column, with a unique key for each row. Other tables can use these keys to access the data without reorganizing the table.

- The pros: Relational databases are simple, structured and ¬flexible. They're often used when processing speed is not a factor. They use Structured Query Language (SQL), a commonly understood process for manipulating data. Relational databases are often used in industries such as banking and financial services; because the data is not divisible, data integrity is preserved.
- The cons: Relational databases can be slow. If there are many tables utilizing relationships, the responsiveness of data queries can be delayed. In addition, relational databases scale up well, but do not scale out well, making storage expensive.

Non-Relational Database (NOSQL)

NoSQL[1] was developed in response to the shortcoming of relational databases, and was deigned to be more open source, more ¬flexible, and horizontally scalable. Unlike relational databases, NoSQL databases are not set up to have tables with linked relationships. There are several types of NoSQL databases, each with slightly different attributes defining how the information is stored and displayed to the user.

- The pros: NoSQL databases are generally more scalable than relational ones and performance is generally not an issue. They are designed to expand transparently and horizontally using low-cost hardware.
- The cons: NoSQL databases generally cannot handle the analytic processing of the data or joins, which are common requirements for IoT applications. They employ low-level query languages, and do not accommodate transactions where data integrity needs to be preserved (such as in the banking example above).

The reality is, the IoT requires characteristics of both relational [2] and NoSQL databases; the flexibility of NoSQL, which allows different types of data to be stored, and the agility to adapt the underlying data models to specific business requirements and applications, and the data integrity aspects of the relational approach.The NoSQL database is used to address lightweight data processing requirements. NoSQL database can get as much as 50,000 inserts per second where as SQL databases which supports up to 5,000 inserts per second. Though NoSQL supports real-time data ingest engine which is a significant requirement in Internet of Things data models, but NoSQL databases are not developed in ways to address analytic data processing requirements which are one of the basic requirements to create databases of Internet of Things applications.

A database for IoT applications must be scalable. Ideally, IoT databases are linearly scalable so adding one more server to a 10 node cluster increases throughput by 10%. IoT databases will usually be distributed unless the application collects only a small amount of data that will not grow substantially. Distributed databases [3] can run on commodity hardware and scale by adding new servers instead of swapping out a server for a larger one. Distributed databases are especially well suited for IaaS clouds since it is relatively easy to add and remove servers from the database cluster as needed.

An IoT database should also be fault tolerant and highly available. If a node in the database cluster is down, it should still be able to accept read and write requests. Distributed databases make copies, or replicas, of data and write them to multiple servers. If one of the servers storing a particular data set fails, then one of the other servers storing a replica of the data set can respond to the read query. Write requests can be handled in a couple of ways. If the server that would normally accept a write request is down, another node in the server can accept the write request and forward it to the target server when it is back online.

III. DATABASES IN IOT

The NoSQL database is used to address lightweight data processing requirements. NOSQL database can get as much as 50,000 inserts per second where as SQL databases which supports up to 5,000 inserts per second. Though NoSQL supports real-time data ingest engine which is a significant requirement in Internet of Things data models, but NoSQL databases are not developed in ways to address analytic data processing requirements which are one of the basic requirements to create databases of Internet of Things applications.

A. An approach to ensuring high availability with regards to writes is to use a distributed messaging system such as Apache-Kafka or Amazon Kinesis. These systems can accept writes at high volumes and store them persistently in a publish-and-subscribe system. If a server is down or the volume of writes is too high for the distributed database to ingest in real time, data can be stored in the messaging system until the database processes the backlog of data or additional nodes are added to the database cluster.

B. IoT databases should be as flexible as required by the application NoSQL[4] databases easily accommodate different data types and structures without the need for predefined, fixed schemas. NoSQL databases are good options when an organization has multiple data types and those data types will likely change over time. In other cases, applications that collect a fixed set of data -- such as data on weather conditions -- may benefit from a relational model. In-memory SQL databases, such as MemSQL, offer this benefit. MemSQL concurrently reads and writes data on a distributed system, enabling access to billions of records in seconds.

C. Managing a database for IoT applications in-house : For those organizations choosing to manage their own databases,DataStaxCassandrais a highly scalable distributed database that supports a flexible big table schema and fast writes and scales to large volumes of data.It easily handles modern data formats-structured, semi-structured and unstructured that run through today's web and mobile applications.It also dynamically adapts to the changes in the data structures. The Cassandra query language (CQL)looks and acts just like SQL.It allows to easily add more customers and more data whenever we need.

D. Riak TS is a high performance, highly resilient NoSQL database optimized for time series data. Riak is used to store a variety of data from over 130,000 data sources including satellites, radars, forecast models, users, and weather stations worldwide..RiakTS is a distributed, highly scalable key-value data store which integrates with Apache Spark, a big data analytics platform that enables stream analytic processing.

E. Readying the data centre for the IoT era : OpenTSDB[5] is an open source database capable of running on Hadoop and HBase. The database is made up of command line interfaces and a Time Series Daemon (TSD). TSDs, which are responsible for processing all database requests, run independently of one another.

F. For Location-based IoT applications: MemSQL[6] is a relational database tuned for real-time data streaming. With MemSQL, streamed data, transactions and historical data can be kept within the same database. The database also has the capacity to work well with geospatial data out of the box, which could be useful for location-based IoT applications. MemSQL supports integration with Hadoop Distributed File System and Apache Spark, as well as other data warehousing solutions.

G. For smart home applications: VoltDB's distributed database infrastructure and ability to perform streaming analytics with transactions on live streams of data support smart home security application requirements. VoltDB is elastically scalable, able to keep pace with variable workloads, is cloud-based, and supports real-time decision making, which relies on both streaming analytics and transactions. VoltDB's familiar relational SQL model and ability to support the highly interactive security services, while managing state and executing per-event transactions, provide an ideal solution for providers of smart home security systems.

H. Hadoop[7] is another database structure chosen for creating low-cost data storage. This database structure also allows addressing and performing analytic processing of large volume of device data. But despite its capacity to address large volume of data requirements, there is one deficit: the Hadoop database batch process takes 6 to 48 hours to analyse which is a sheer challenge in processing real-time data processing. However, database vendors are working on alleviating this problem and reduce time-consuming impact by adopting processes such as in-memory systems and incremental batching methods. But, on the other hand, these procedures take in significant expenses and time. Hence, Hadoop serves as low-cost high-volume Internet of Things database but, it does not support fast-paced data processing and analysis.

I. For ad hoc and real-time data processing: ScaleDB is another database structure which provides analytical processing and transactional data processing. ScaleDB can be used for ad hoc and real-time data processing in IoT environment. It combines SQL-based database structure with high-velocity data analysis performance.

J. MongoDB is also one of the open-source document database, and leading NoSQL database. MongoDB is written in C++.Organizations are using MongoDB for IoT because it lets them store any kind of data, analyze it in real time, and change the schema as they go. Its features are:

- New Devices and Data. MongoDB's document model enables you to store and process data of any structure: events, time series data, geospatial coordinates, text and binary data, and anything else. You can adapt the structure of a document's schema just by adding new fields, making it simple to handle the rapidly changing data generated by IoT applications.
- Horizontal Scalability. MongoDB's automatic sharing distributes data across fleets of commodity servers, with complete application transparency. With multiple options for scaling – including range-based, hash-based and location-aware sharing – MongoDB can support thousands of nodes, petabytes of data and hundreds of thousands of ops per second, without requiring you to build custom partitioning and caching layers.
- MapReduce, MongoDB can run complex ad-hoc or reporting analytics in-place against sensor data.
- Security. Robust authentication, authorization, auditing and encryption controls protect valuable sensor data and the analytics delivered from it.

IV. ACKNOWLEDGMENT

This paper considers the challenges brought by the need to manage vast quantities of data across heterogeneous systems. In particular, it considers the factors to keep in mind when choosing a database for an IoT application. Different applications, by nature, have different requirements. This is why choosing the appropriate database type is a vital component for ensuring success. There are many relatively mature time-series databases, or platforms that can be used as a time-series database, a number of which are open source. There are too many to enumerate (for open source you have Open TSDB, Cassandra, etc) and a number of fast relational databases can similarly support basic time-series applications (ParStream, MemSQL, etc).

REFERENCES

[1]. Cattell R. Scalable SQL and NoSQL data stores. ACM SIGMOD Rec. 2010;39:12–27
[2]. NishthaJatana, SahilPuri, Mehak Ahuja, IshitaKathuria, DishantGosain, "A Survey and Comparison of Relational and Non-Relational Database," International Journal of Engineering Research & Technology (IJERT), vol. I, no. 6, 2012.
[3]. Ozsu M.T., Valduriez P. Principles of Distributed Database Systems. 3rd ed. Springer; New York, NY, USA: 2011.
[4]. C. Nance and T. Losser, "NOSQL VS RDBMS -WHY THERE IS ROOM FOR BOTH," inProceedings of the Southern Association forInformation Systems Conference., Savannah, GA,USA, 2013
[5]. StumbleUpon, OpenTSDB: Open Time Series Database, http://opentsdb.net
[6]. Harris, Derrick (2012). "Ex-Facebookers Launch MemSQL to make your Database Fly" (published 2012-06-18). Retrieved 2012-10-07.
[7]. A Review paper on Big Data and Hadoop-published at "International journal of Scientific and Research Publications(IJSRP),volume 4, Issue 10,October 2014