**Why do we need a time series database ?**

**Abstract**: As China has entered the 5G era since the official launch of 5G commercialization, the Internet of Things and industrial production have seen tremendous progress, and with that comes the problem of huge amounts of data storage. This paper introduces the definition of temporal database, the history of temporal database, and the current application of temporal database and its characteristics.

**Keyword**: time series database

**What is time series database**

In the past, in the traditional field, relational database has been widely used due to its simple and easy to understand structure, easy to use and operate, flexible query and easy maintenance. With the official launch of 5G commercialization, China officially enters the 5G era and the Internet of Things (IoT) will also usher in a huge transformation. The high-speed communication brought by 5G technology brings more efficient operation for the data transmission speed of IoT. An important technology in the development of IoT is the temporal database. The large amount of data generated by IoT devices needs to be processed and stored through an effective temporal database. With the rapid development of IoT and related technologies, a large amount of time series data is generated due to the 24\*7 production operation in industrial production, such as electricity, energy consumption and gas.

The data stored in the time series database is the time series data, and the most important feature is that the data is arranged chronologically. Common time series data include CPU usage, stock prices, and readings from various industrial sensors. Time series data is usually not long, in a single piece of data, but the amount of data is large. Time-series data differs significantly from relational data in that.

(1) The most obvious feature is that all time-series data have unique timestamps and are sorted by timestamp size and distinguished by timestamps as unique identifiers, while relational data usually have other fields as identifiers, e.g., student data usually use student numbers as unique identifiers to distinguish.

(2) Temporal data does not care about relationships. In car location, we do not need to know other attributes of the owner of this car, such as age, occupation, etc., and there is no association to the table of car owners.

(3) The data volume of temporal data continues to grow linearly, and new data will be generated at a certain time granularity, which will continue to generate a huge amount of data, so the data volume is huge. The growth of relational data is usually not continuous over time, for example, the amount of student data in a school is relatively stable over a period of time.

(4) Time-series data will rarely be updated, and the measured values at a certain moment will not change, so there is almost no need to update the time-series data. For relational data, updates occur frequently for data that already exists, such as students' personal information, including attributes such as age and height.

Time series data is used to describe information about changes in the state of an object in the historical time dimension. Analysis of time series data is the process of trying to understand and grasp the rules of change. With the development of IoT, Big Data and Artificial Intelligence (AI) technologies, time-series data has experienced an explosion. To better support the storage and analysis of such data, various database products have been created and are available. Such database products were invented to address the shortcomings and drawbacks of traditional relational databases for storing and analyzing time-series data. These products are uniformly classified as time series databases (TSDB). A time series database is a database optimized for accessing time series data. Time series databases are widely used in the Internet, Internet of Things and other fields because of their unique support for time series data, and their application scenarios and scope are still expanding. For the characteristics that the writing frequency of time series data is much higher than the query frequency, the data is less updated, and the queries are mostly based on time periods, most time series databases provide monitoring, downsampling, aggregation and other functions to facilitate the analysis, calculation and visualization of time series data.

**History of Time-Series Database**

Although temporal databases have only entered the public eye in recent years, their development can be traced back to the 1990s, when the need for temporal data storage arose in the surveillance field. The resulting first-generation temporal databases, represented by RRDtool and Whisper, use fixed-size databases that can quickly store numerical data over time, but their read performance is still weak, lacking It is still weak in read performance, lacks special optimization for time, and handles a single data model, which is usually embedded in monitoring systems.

With the development of big data, temporal data has exploded, not only monitoring system, but also other systems have more demands for processing temporal data. In 2011, there are temporal databases based on distributed storage represented by OpenTSDB and KairosDB, which are optimized for time on the basis of inheriting the advantages of general storage. For example, OpenTSDB relies on HBase cluster storage at the bottom, compresses data according to the characteristics of time series to save storage space; reads and writes with TSD, encapsulates common queries for time series data, and provides data aggregation, filtering and other operations. This kind of database dedicated to temporal data has significantly improved the storage and query performance compared with the first generation of temporal database, but this kind of database also has many shortcomings, such as inefficient global UID mechanism, reliance on Hadoop and HBase environment, high deployment and maintenance costs.

With the development of microservices, temporal databases are developing at a high speed, and the shortcomings and deployment complexity of OpenTSDB promote the birth of low-cost vertical temporal databases. IfluxDB has become the mainstream of the temporal database market, with more efficient storage and data processing capabilities and efficient compression algorithms for temporal data. It can reduce redundant storage and improve data compression rate. Compared with OpenTSDB, which requires Java environment and HBase environment, InfluxDB is based on Goland, no dependency, easy to deploy, and uses SQL-like InfluxQL, which is easy to develop.

In the past 5 years, the development of temporal database is very rapid, and major Internet companies including Google, Ali, Amazon have launched their own temporal database, and DB-Engines has also classified temporal database as an independent directory for statistics since 2014. Since 2015, various temporal databases have sprung up: in 2017, Facebook opened the prototype project beringei of memory-based Gorilla temporal database (released in 2015), TimescaleDB designed based on PostgreSQL was opened, and Ali released the temporal database TSDB; in 2018 Amazon launched the fully managed Timestream; among them, the open source InfluxDB, released in 2013, stands out among all kinds of temporal databases, and until now is the number one temporal database attention and its popularity is still growing, due to its open source features, InfluxDB uses the SQL-like InfluxQL query language Due to its open source feature, the ease of use of InfluxDB using the SQL-like InfluxQL query language, and the lack of environmental dependencies based on Go language, InfluxDB versions are iterating rapidly, with seven releases in 2019 and a cloud service version.

**Features and tasks of a time-series database**

Important features of the chronological database include

1. Data Lifecycle Management: The process of managing the flow of data throughout its lifecycle from collection and introduction to aggregation, processing and expiration.

2. Abstraction: The practice of presenting meaningful data summaries through flexible queries, transformations, visualizations and dashboards.

3. Extensive scanning of many records: Scanning millions of chronological records is a common requirement for many chronological use cases. These types of scans require specialized software, such as temporal databases, that utilize specially built integrated algorithms for compression, indexing, and spatial mapping to enable users to quickly write, query, and visualize millions of points.

These capabilities are designed to facilitate the processing of large volumes of time series data at scale. Common tasks for time series databases include.

1. Writing large volumes of data. Whether you are collecting and writing data with nanosecond precision for high-frequency transactions or collecting data from hundreds of thousands of sensors, the Time Series Database is optimized for high ingestion rates that other databases simply cannot handle.

2. Request data summaries over large time periods. Collecting data summaries over large time periods helps provide insight into the overall behavior of the data. For example, before deciding which city you want to move to, you may want to see the monthly average temperature for each city over many years.

3. Automatically downsample or expire old time series that are no longer useful, or retain high-precision data for short periods of time. For example, monitoring the pressure in a chemical plant pipeline every minute is critical to maintaining safety standards during operation. However, this data does not need to be retained forever with high accuracy. The time series database should allow the user to sample that minute-accurate data as a daily average.

**The use of chronological databases in real-life scenarios**

1. Storing and Accessing IoT Data

Most IoT deployments (e.g., connected water meters, energy meters, and thermometers) involve regular and continuous data collection and reporting. Time series analysis can provide time-stamped data points that can be used to identify patterns, average usage, and inefficiencies.For example, a pH meter connected to a TSDB's connection may tell the technician responsible for maintaining a particular pH that a certain bucket of water is becoming too acidic. IoT endpoints also collect large amounts of data, requiring highly scalable time-series databases.

2.Monitoring Web services, applications and infrastructure

TSDBs can measure the performance of a company's applications and services. For example, the open source monitoring system Prometheus is a time-series database that enables developers to closely monitor performance trends over time. This allows them to easily detect when problems are occurring, so they can plan maintenance and respond quickly to events to maintain an optimal user experience.

Some web and mobile applications store events in the TSDB's application (such as button clicks, playing videos, or sharing certain content). With this data, they can map the user journey, identify frustrations or performance bottlenecks, and enhance the user experience.

3. Understand financial trends

Accurately predicting financial trends using time series data is very difficult. However, TSDB can provide rich contextual data to help analysts. Let's take the stock market as an example; a sudden increase in airline inventory may coincide with holiday travel. Or, an executive purge may spook investors and cause stocks to temporarily plummet. Time-series databases make it easy to cross-reference data to provide a richer, clearer picture.

4. Handling self-driving car data

Self-driving cars typically collect about 4,000 GB of data per day. This is more than a typical relational database can handle. Time-series databases can bring in data and queries faster and compress data more strongly.

As a result, they are ideal for handling large amounts of real-time data that can be used to improve the safety of self-driving cars.

5. Sales forecasting

Retail stores must constantly forecast future sales based on past sales so they can properly replenish their inventory. With time-series databases, retailers can use statistical models that combine with historical data. By cross-referencing with consumer behavior trends, they can predict future patterns and make informed decisions about which products to stock and when to stock them.

For example, retailers are now using forecasting to plan ahead and refill bicycles, which are now experiencing shortages due to the pandemic. Retailers are using data to predict when new products will be available again, what demand will look like, and what alternative modes of transportation consumers are buying instead of bikes, such as tricycles, rollerblades, etc.

**Conclusion**

The development of chronological database in the present time is due to the further deepening of industrialization and the advent of 5G era. Therefore, the temporal database is also in the stage of high speed development and the technology is gradually becoming mature. It is believed that in the near future, temporal database will become a more mature and excellent technology.

**References**

[1]: <https://aiven.io/blog/an-introduction-to-time-series-databases>

[2]: <https://github.com/winfredliu/TSDBHomework/blob/master/Why%20do%20we%20need%20a%20time-series%20database.md>

[3]: <https://baijiahao.baidu.com/s?id=1708500603441764092>