## Why do we need time-series database

Before we start, what is time-series database, or, what is time-series data?

Time series data is a collection of observations (behavior) for a single subject (entity) at different time intervals (generally equally spaced as in the case of metrics, or unequally spaced as in the case of events).

A time series data example can be any information sequence that was taken at specific time intervals (whether regular or irregular). Common data examples could be anything from heart rate to the unit price of store goods.

Time series data can be classified into two types:

Measurements gathered at regular time intervals (metrics)

Measurements gathered at irregular time intervals (events)

What can a Time-series database be used for?

Time series database(TSDB) is mainly used to process the data with time label (change according to the order of time, that is, time serialization). The data with time label is also called time series data.

Time series data are mainly collected and generated by all kinds of real-time monitoring, inspection and analysis equipment in electric power industry, chemical industry, meteorological industry, geographic information industry and other industries. The industrial data of typical features are: frequency fast (within one second of every monitoring can produce multiple data), which relies heavily on acquisition time (in time only need every data), the large amount of information of measuring point (regular, tens of thousands of monitoring real time monitoring system, monitoring data of every second, and produce dozens of GB of data volume) every day.

Main Application scenarios: **Monitor the business**

The biggest application scenario of TSDB is monitoring services (Sentry). Taking Sentry as an example, sentry will deploy various script clients on the service server to collect server indicator data (IO indicator, CPU indicator, bandwidth and memory indicator, etc.). Business related data (abnormal method call number, delayed response, the JVM GC data, etc.), database related data (read, write, delay delay, etc.), obviously, these data are related to the time sequence, after the client acquisition will be sent to the sentinel server, the sentinel server will these data for storage, and provide the page to the user query.

So that, What is the difference between Time-series database and ordinary database(relational database)?

There are huge differences between Time-series business and ordinary business in many aspects, which can be summarized in the following aspects:

**First**, Continue to produce huge amounts of data, with no peaks and troughs. To take a simple example, a sentry monitoring system that monitors 100 metrics per second on 1W servers would generate 100 WATTS of TPS per second. For example, if 100 million people are wearing a popular sports wristband and each wristband only collects 3 metrcis (heartbeat, pulse and step count) per second, it will also generate 300W TPS per second.

**Second**, Data are all insert operation, basically no update deletion operation. Based on the fact that data generated by sequential services is rarely updated or deleted, there can be considerable simplification in the design of sequential database architectures.

**Third**, In the future, more attention will be paid to streaming processing. Long-lived data are rarely accessed or even discarded. It's easy to understand, sentry systems we tend to care most about the last hour, most about the last three days, very little about the last three days. With the advent of streaming computing, sequential data will inevitably pay more attention to the value of real-time data in the future development, and this part of data is undoubtedly the most valuable. It is a very common and important scenario that the data can be generated and the alarm can be generated according to certain rules. The more timely the alarm is, the better for the business.

**Finally,** Data has labels of multiple dimensions, which often requires multi-dimensional joint query and statistical query. Another very important function of time series data is multi-dimensional aggregated statistical query. For example, the business needs to calculate the click-through rate and total revenue of advertisements published by Google in USA in the last hour, which is a typical multi-dimensional aggregated statistical query demand. This requirement usually has a low requirement for effectiveness, but a high requirement for query aggregation performance.

We have talked about the difference between TSDB and relational database, and then, what are the core technical points that TSDB focuses on?

**High throughput write capability:**

This is tailored to the feature that sequential services continuously generate massive data. Currently, to achieve high throughput write of the system, two basic technical requirements must be met: horizontal scalability of the system and single-node LSM architecture. It is easy to understand the horizontal scalability of the system, which cannot be supported by a single machine. The system must be clustered, and it is easy to add nodes to expand. In the final analysis, it is not aware of the business when expanding. The LSM architecture is used to ensure high throughput write on a single machine. In the LSM architecture, data writing only needs to be written to the memory and written to the log, so there is no need to write data randomly to the disk. Currently, HBase, Kudu, Druid and other systems that require high write performance adopt this architecture.

**Data tiered storage:**

This is a technical feature customized for the hot and cold nature of temporal data. Data tiering requires that the latest hour data can be stored in memory, the latest day data can be stored on SSDS, and the older data can be stored on cheaper HDDS or TTL expiration.

**High compression rate**:

There are two reasons to provide high compression rate. One is cost saving, which is easy to understand. If you compress 1 TB data to 100 GB, you can reduce the disk cost of 900 GB, which is a great temptation for businesses. Another aspect is the compressed data can be easier to ensure that stored in the memory, such as the recent data is 1 t 3 hours, I now only 100 gb of memory, if without compression, there will be 900 gigabytes of data are forced into the hard disk, it will query overhead is very large, and will use compression this 1 t data into memory, the query performance will be very good.

**Multi-dimensional query capability:**

Time-series data usually have labels of multiple dimensions to describe a single data, which is the dimension column mentioned above. How to efficiently query according to random dimensions is a problem that must be solved. This problem usually needs to consider bitmap index or inverted index technology

**High efficiency polymerization capacity:**

A common requirement for Time-series services is aggregated statistical report queries, such as the sentinels need to see the total number of exceptions on an interface in the last day, or the maximum elapsed time for an interface to execute. Such aggregation is actually simple count and Max. The problem is how to efficiently query and aggregate the original data that meets the conditions on the basis of such a large amount of data. You should know that the original value of statistics may not be in memory because of a long time ago, so this may be a very time-consuming operation. Currently, the most mature solution in the industry is to use pre-aggregation, which is to complete basic aggregation operations as data is written in.

TSDB provide technology as above, so that TSDB has broad application prospects

What TSDB products are available now?

**InfluxDB**

InfluxDB is a one-stop timing toolkit that includes everything needed for a timing platform: multi-tenant timing databases, UI and dashboard tools, backend processing and monitoring data collectors.

InfluxDB supports dynamic SHCEMA, that is, no schema definition is required before data is written. Therefore, users can add measurement, Tag, and field as many columns as they want

**Prometheus**

Prometheus is an open source monitoring, alarm, and time series database combination originally developed by SoundCloud. As it grew, more and more companies and organizations adopted Prometheus, and the society became so active that it became an open source project in its own right and was run by a company. A similar implementation to their BorgMon monitoring system is Prometheus, as mentioned in Google SRE's book. The most common Kubernetes container management system today is usually monitored with Prometheus.

**[TDengine](https://so.csdn.net/so/search?q=TDengine&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/gaojingsong/article/details/_blank)**

TDengine is a high-performance, distributed, SQL-enabled sequential database, its core code, including clustering functions are all open source (open source license, AGPL V3.0). TDengine can be widely used in Internet of Things, industrial Internet, Internet of vehicles, IT operation and maintenance, finance and other fields. In addition to the core sequential database function, TDengine also provides a series of functions required by the big data platform, such as caching, data subscription, streaming computing, minimizing the complexity of development and operation.

Looking forward to future.

Time-series database is exploding in recent years, and the search index of all search engines is also on the rise.

There is a reason for the rise of Time-series databases. Take unmanned driving as an example, unmanned vehicles need to monitor various states during operation, including coordinates, speed, direction, temperature, humidity and so on, and need to record the monitoring data at every moment for big data analysis. Each car collects nearly 8 terabytes of data every day. If it is just stored without query, it is also ok (although it is already a large cost). But if you need to quickly query the multi-latitude grouping and aggregation query such as "what are the unmanned cars with a speed of more than 60km/h on Beijing Road at 2 o 'clock this afternoon?", then the timing database will be a good choice.

This is time series data, and it's playing a bigger role in our world. At present, time series database (TSDB) has become the fastest growing database category. With the coming of 5G, sequential databases will become even more popular in the future.