Why do we need a time-series database?

Our world is changing with a super speed, it requires greater ability to collect and analyse more and more data. For a series questions, such as self-driving Tesla, automated Wall Street trading algorithms, smart home, same-day delivery, tracking daily COVID-19 statistics and neighborhood air quality, we are easy to find that each of these applications needs a special type of data. For example, Self-driving cars need to constantly collect data about the changing environment around them and adjust for weather conditions, potholes and many other variables, automated trading algorithms need to continuously collect relevant data on market changes to optimize short-term and long-term investment returns, smart home needs to monitor indoor conditions to regulate temperature, identify hazards and handle human-computer interaction, the retail industry accurately and efficiently monitors the transportation of every item.

Compared with stock market trends, self-driving cars and predicting the exact time of your next online shopping, there have been many examples of how the collection and analysis of temporal data affects individuals' daily lives in recent years. For the first time, the world's interest in time series data has peaked in a most unexpected way. The COVID-19 pandemic makes billions of people around the world as consumers of time series data and demand accurate and timely information on COVID-19 trends from day to day. In a world where we are so eager to the information, access to series data which is detailed and rich of feature has become one of the most valuable commodities. Business, governments, schools and communities, whenever large or

small , are looking for ways to extract value from analyzing time series data. In the last two years, time series databases have always been the fastest growing databases.

Time series data, just namely time series data, is recorded and indexed sequentially according to the time dimension. Various types of devices in areas such as smart cities, Internet, Internet of vehicles and industrial Internet will generate massive amounts of temporal data. These data will account for more than 90% of the total time series data, it is convenient for us to analyze the data according to the time change. Previously, our view of time series data was static: daily temperature highs and lows, stock market opening and closing prices, even daily or cumulative hospitalizations due to COVID-19. However, we may overlook the potential for subtle differences in these static data. Let's look at some examples. If I give you 10 yuan, the traditional database of the bank will have an expense in my account and an income in your account. And then, if you give me 10 yuan, the same process goes the other way. At the end of the day, our bank balance still looks the same, and as far as banks are concerned, nothing has changed this month. But with a sequential database, the bank could sense that there might be a deeper reason why these two people keep exchanging 10 yuan. If you track this slight discrepancy, your month-end account balance will take on more meaning. For an other example like the average temperature over several days in a place. Over the past few decades, average temperature has been used as the main reference factor for building energy efficiency. In any given week, the average daily temperature in the same place may be only slightly different, but the factors that affect the environment can change dramatically over the same period of time. Conversely, knowing

the hourly temperature changes during the day, coupled with the amount of precipitation, cloud cover and wind speed during that time, can greatly improve the ability to model properties and optimize energy efficiency. As the same, although it is much valuable to know the total number of daily COVID-19 hospitalizations in a community, the number alone does not provide a good situation of the details. For example, a hospital might find that 20 people were admitted on Monday and that the number increased slightly throughout the week, to a total of 23 by Friday. At first glance, there was a 15% increase in hospitalizations this week. But if we count the detailed records (increasing the frequency of collection), we might see a net increase of 3 cases this week, but in fact, 10 people have been discharged, 13 have been added, so in the last 5 days, there has been a 65% increase in new admissions. Tacking various aspects of patient data(such as patient age, admission or discharge, days of recovery) helps us understand how to come up with daily statistics, enabling us to better analyze trends, accurately report totals, and take action that may even influence government policy.

These examples show how different between modern time series data and the used data we know. Time series data analysis is much deeper than pie charts or Excel. The data is not just a measure of time, the point is to help us analyze the data and get valuable information. There is also a wide variety of temporal data, but regardless of scenario or user case, all temporal data sets have three things in common: 1. The written data is almost new data. 2. Data is written in chronological order. 3. Time is a principal axis(intervals can be regular or irregular). In other words, sequential data is often written by appending. While data corrections, processing delays and unordered data may be required after

the fact, these are exceptions, not the norm.

The state of data changes over time. This is different from adding a time column to the data set. The different depends on how the data set tracks the change, it is just update the current item or insert a new one. When collecting new data for a sensor, we can overwrite previous readings or create entirely new data? While both methods will provide the current state of the system, if we insert a new data each time, the change in state can be analyzed. In a word, the sequential data set changes tracks of the system into INSERT rather than UPDATE operations. Each change to the system is logged as a new, different row. This is what makes temporal data powerful, allowing us to measure and analyze changes about what has changed in the past, what is changing now, and what we can predict in the future. In briefly, sequential data is a set of values that represent how the system/process/behavior changes over time. Many times, we have temporal data, but we don’t realize it’s temporal data.

Preserving the inherent timing nature of data allows us to retain valuable information, such as how the data has changed over time. Of course, there's an obvious problem with storing data in this way: we will finally get huge amounts of data, and they grow very fast. So here's the problem: being able to analyze increased temporal data is more valuable than ever, but it's piling up really fast. Large amounts of data can cause a number of problems, from storage to quick queries, which is why people are more inclined to use sequential databases than ever before. The world demands that we make data-driven decisions faster and better. Traditional static data can not solve this problem. To meet the requirements, we need to collect data with the highest quality as

possible as we can, this is what sequential data provides: everything that happens in the system can be stored like a movie, whether it's software, a physical power plant, a game, or a customer in an application.

There are at least two reasons that TSDB become the fastest growing database today: scale and availability. Scale: Temporal data accumulates very quickly, and regular databases are designed to handle this scale (at least not in an automated way). Relational databases perform poorly on very large data sets, while NoSQL databases perform better on scale. In contrast, the benefits that introducing sequential databases (whether relational databases or databases based on nosql) are only possible if we consider time as a primary consideration. These benefits enable them to provide large-scale performance improvements, including higher throughput and faster large-scale queries, as well as better data compression. Availability: TSDB also includes built-in functions and operations commonly used for temporal data analysis, such as data retention policies, continuous queries, flexible time aggregation, and so on. Even if we're just starting to collect this type of data and don't need to worry about scale right now, these features can still provide a better user experience and make the task of analyzing data easier. Using built-in functions and features to analyze trends which are ready to use in the data layer often reveals unexpected value, no matter how large or small the data set is. This is why developer are increasingly using sequential databases and applying them to a variety of scenarios.

Once the sequential data is used to store more information, we still have to choose the sequential database that best fits the business data model, read and write mode, and development technology line. Although NoSQL time series databases have gained popularity as the

preferred storage medium over the past decade, more and more developers see the disadvantages of storing sequential data separately from business data (most sequential databases do not provide good support for relational data). That is one of the main reasons we developed FastData For TSDB, keeping all data in one system can greatly reduce the time of developing a application, then we can make a quick and crucial decision. The point is knowing where temporal data is, and where to store it, will have a huge impact on future developments.

That’s all why we need a time-series database.