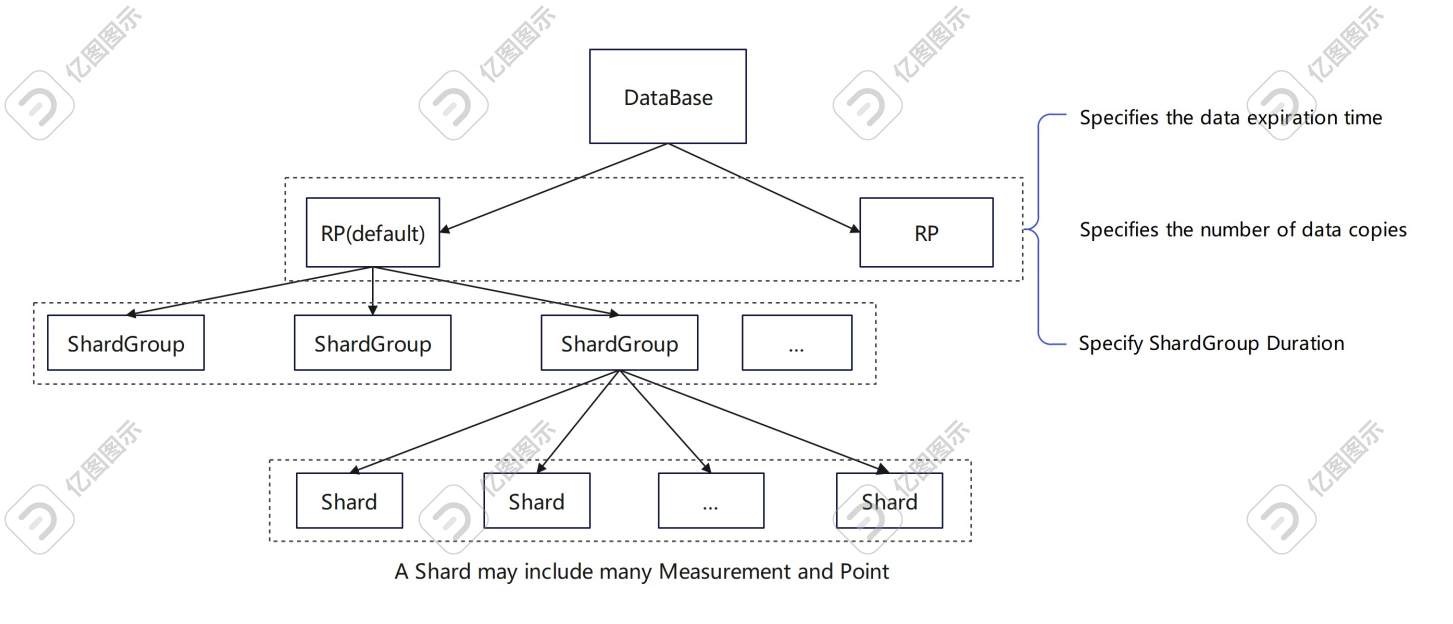
InfluxDB

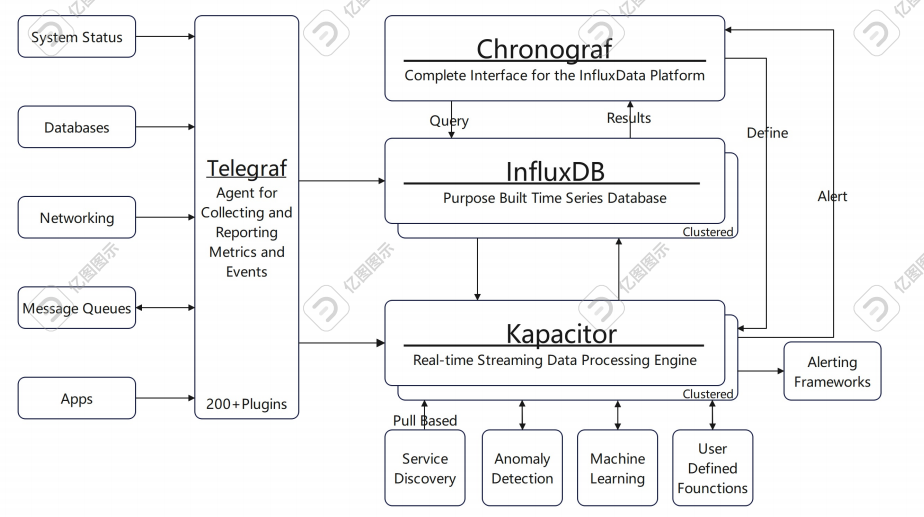
InfluxDB is an open source distributed database of timing, events and metrics written in Go by InfluxData without external dependencies. influxDB focuses on high performance reads, high performance writes, efficient storage and real time analytics of massive amounts of timing data and is designed to be used as a backend storage for any use case designed for large amounts of timestamped data, including DevOps monitoring, application metrics IoT sensor data and real-time analytics.

InfluxDB has the following features: 1. custom high performance data store written specifically for time series data. tsm engine with high performance writing and data compression. 2. tailored SQL-like query language for time series data, easy to query aggregated data. 3. tags can be indexed serialized to provide fast and efficient queries. 4. supports multiple protocols, in addition to HTTP InfluxDB supports a variety of protocols, in addition to native protocols such as HTTP, UDP and so on, it is also compatible with the communication protocols of CollectD, Graphite, OpenTSDB, Prometheus and other components. 5. The interface is friendly and easy to use. 6.It has rich aggregation operation and sampling capability, and provides flexible data storage strategy. In addition, InfluxDB also has time series-based, supporting time-related correlation functions (such as maximum, minimum, summation); metric, which means it can calculate a large amount of data in real time; event-based,as it supports arbitrary event data and other functional features. At the same time, InfluxDB has a number of design concepts. For example, multiple writes to the same data at the same point in time are considered duplicate writes; data is rarely deleted, and deletions are basically cleanups of expired data; existing data is rarely updated and there are no contentious updates, and time series data is always new; most writes are to the most recent timestamped data, and data is added in ascending time order; being able to write and query data is more important than strong consistency. The size of the data will be very large and it must be able to handle a large number of read and write operations, etc.



(InfluxDB structure)

InfluxDB, Telegraf, Chronograf and Kapacitor are the four components of TICK, an open source timing console that inherits the functions of acquisition, storage, analysis and visualization. Its various modules work together and complement each other, focusing on DevOps monitoring, IoT monitoring, real-time analytics and other scenarios. Among them, InfluxDB focuses on DevOps monitoring, IoT monitoring and other scenarios. A set of software designed for high-performance read and write, real-time operations, temporal storage and high availability, designed and developed from scratch, InfluxDB can efficiently collect, store, query, visualise and perform predefined operations in real-time by implementing a highly scalable data reception and storage engine. It does this through sampling and data retention policies to support keeping high-value, high-precision data in memory and low-value data to disk. As a well-designed, architecturally superior, dedicated system, InfluxDB offers significant performance and cost advantages over OpenTSDB, MongoDB, Graphite, Cassandra and others. According to statistics, InfluxDB's write performance is 5 times that of OpenTSDB, 2.4 times that of MongoDB, 12 times that of Graphite, 4.5 times that of Cassandra, storage efficiency is 16.5 times that of OpenTSDB, 20 times that of MongoDB, 6.3 times that of Graphite, 2.1 times that of Cassandra. Query is also 3.65 times more efficient than OpenTSDB, 5.7 times more efficient than MongoDB, 9 times more efficient than Graphite, and 45 times more efficient than Cassandra. This shows that InfluxDB has a great advantage over other major database software in all aspects, whether in terms of read/write, storage or query, InfluxDB is a level above other mainstream software performance.



(TICK structure)

In InfluxDB, a virtual key for data consists of database, retention policy, measurement, tag sets, field name, timestamp, where database and retention policy are usually specified in the corresponding fields of the http. The database and retention policy are usually specified in the corresponding fields of the http request when inserting the data. Multiple databases can be created in InfluxDB, with data files in different databases being stored in isolated, non-disabled directories on disk. Retention policy automatically creates a default storage policy autogen for each database, with a permanent data retention time and a number of copies in the cluster of 1, after which the user can set their own settings (view, create, modify, delete), such as keeping the most recent data. Afterwards, users can set their own settings (view, create, modify, delete), e.g. retaining the last 2 hours of data. InfluxDB periodically purges expired data. Measurement is a container for tags, fields and time columns, and is a description of the data stored in the relevant field. Tags are sorted in InfluxDB by dictionary order, no matter tag key or time, if they are inconsistent, they belong to two keys. Field sets is the set of field keys and tag values,which is the required fields. Timestamp is the timestamp specified for each piece of data, which presents the UTC date and time associated with a particular piece of data in RFC3339 format. date and time, with the timestamp being the primary key. These parts combine to uniquely identify a record, a Point, which is the equivalent of a tuple in a relational database.

Point and Series are unique concepts in InfluxDB. Point consists of time, tags and fields. It is equivalent to a tuple in a relational database; all the data in the database needs to be displayed through a chart, and series represents the data in this table, which can be drawn as several lines on the chart: it is calculated by arranging and combining the tags. It is important to note that InfluxDB does not require the creation of various tables like a traditional relational database; its tables are mainly created automatically when the data is inserted for the first time, which means the InfluxQL can be written directly to operate on the table, even if the table does not exist.

In addition to the design advantages, InfluxDB also has significant functional advantages. InfluxDB has a unique language InfluxQL, which provides a continuous query approach, which are a set of statements that are automatically started at regular intervals in the database. It must contain keywords such as SELECT. InfluxDB places the query results in a specified data table. Using continuous queries is the best way to reduce the sample rate, and using continuous queries in conjunction with a storage strategy will significantly reduce the system footprint of InfluxDB. Also, when using continuous queries, the data is stored in the specified data table, which makes it easier to count data of different precision in the future. On the other hand, when data is deleted after the time specified in the retention policy, but you don't want it to be completely deleted, in this time you can use continuous queries to store the data in aggregate.

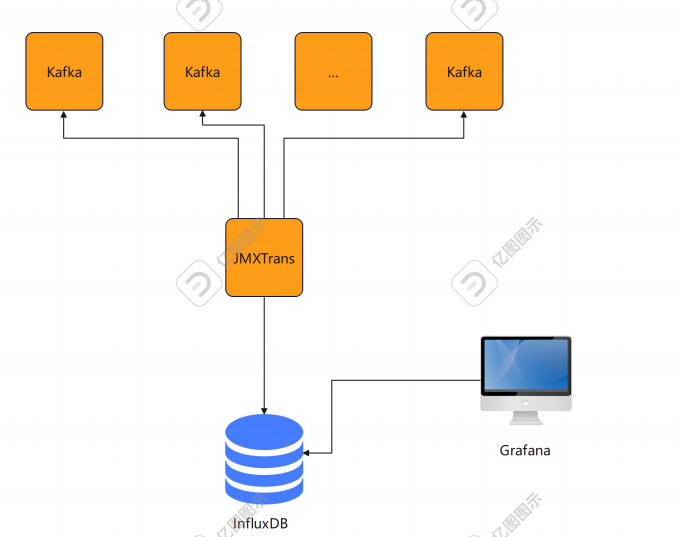
In addition to the InfluxQL language, InfluxDB extends Flux syntax queries. Flux is InfluxData's functional data scripting language designed to query, analyse and manipulate data, and it is a replacement for InfluxQL and other SQL-like query languages. Each Flux query needs to contain the following parts: 1) the data source, using bucket to identify the name of the database. 2) a time range, either a specific time (UTC time) or a timestamp, or a relative time range,for example,’-1h’ for data within the last hour (relative to the current time), with optional units such as seconds, minutes, hours, days, months, years. 3) a data filter, the range of data passed to the filter () function to narrow the results according to data attributes or columns. The filter() function has one argument FN, it requires an anonymous function that has logic for filtering data based on columns or attributes. Multiple filters can be concatenated with “and” or “or” .In another way, a separate filter().

Grafana is a cross-platform open source metric analysis and visualization tool that can query collected data and then show it. Grafana supports many different data sources, the following data sources are officially supported: Graphite, InfluxDB, OpenTSDB, Prometheus, Elasticsearch, CloudWatch and KairosDB. InfluxDB and Grafana can be used to quickly build a system for storing and displaying time-series data.

Telegraf is an open source data collection agent written in Go, based on a plugin-driven approach. Telegraf is the "T" in the TICK technology stack, InfluxData's time-series platform, and is used to collect time-series data, such as server CPU metrics, memory metrics, data generated by various IoT devices, and so on. Telegraf collects data by executing input plug-ins at regular intervals, and the data is processed by processing plug-ins and aggregation plug-ins and exported in bulk to data storage. InfluxDB is used for data storage, Telegraf for data collection and reporting metrics, and Grafana for data presentation, enabling lightweight system monitoring of Windows and Linux systems, server performance monitoring, etc.

Kafka is a distributed, partitioned, replica enabled, zookeeper-based orchestrated distributed messaging system, whose greatest feature is its ability to process large amounts of data in real time to meet various demand scenarios. It has features such as high throughput, low latency, scalability, persistence and high concurrency.

JMXTrans is an open source JMX metrics collection tool, which is easy to use, no need to write code, just configuration files can easily collect a large number of applications JMX metrics data.JMX is monitoring the Java application system running state information, through the analysis of JMX information, can be used to monitor the application running state, optimize the program, troubleshooting problems. JMXTrans + InfluxDB + Grafana can use JMXTrans through JMX to collect kafka monitoring metrics written to InfluxDB, then Grafana read InfluxDB plot and do display method to kafka monitoring and representation, so as to achieve the monitoring of streaming media framework. To implement operations such as queries and calculations on streaming points, InfluxDB uses Tasks to replace InfluxDB v1.x continuous queries.An InfluxDB task is a scheduled Flux script that takes a stream of An InfluxDB task is a scheduled Flux script that takes a stream of input data, modifies or analyzes it in some way, then writes the modified data back to InfluxDB or performs other actions.



(JMXTrans + InfluxDB + Grafana Framework)

Overall, InfluxDB has a great advantage among the mainstream temporal databases today, both in terms of design and functionality. It is not only better than other software in terms of performance, but also has its own unique design concept, which is very convenient for both users and developers. At the same time, InfluxDB has strong compatibility and can fit well in different usage scenarios; at the same time, InfluxDB has strong openness in terms of extensibility and can match various types of frameworks and provide corresponding interfaces, such as InfluxDB+JMXTrans, InfluxDB+Telegraf, InfluxDB+Grafana, etc.