Architecture of Influx DB

- **Key concepts:**
  - Opensource (TICK stack: Telegraph, InfluxDB, Chronograf, Kapacitor)

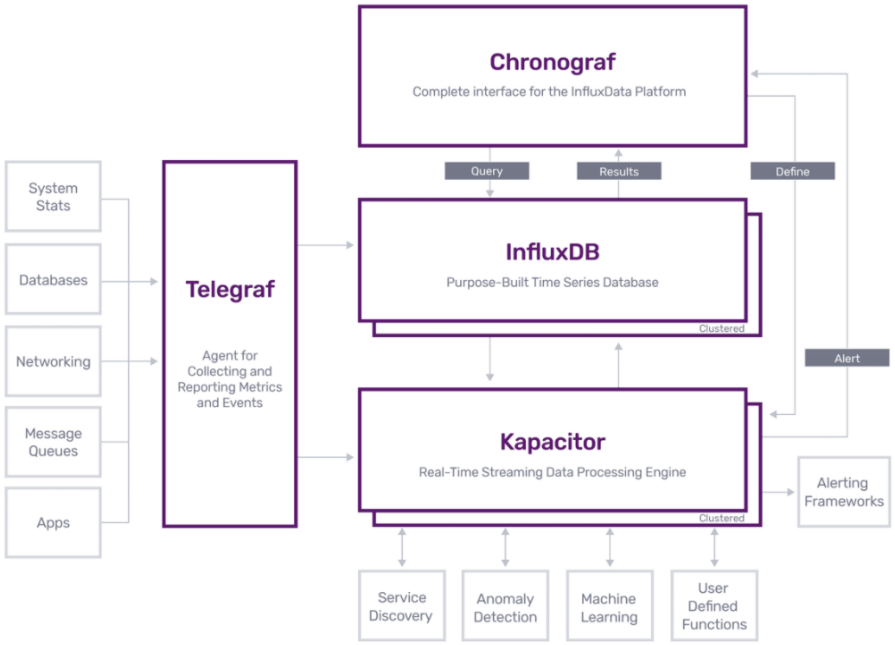

Image from https://www.influxdata.com/time-series-platform/

  - Schemaless
  - Time series database
  - Written in Go programming language
  - Provides query language similar to SQL.
  - Data is **time-indexed/time-stamped**

Commented [1]: agent for collecting and reporting metrics

Commented [2]: a TSDB build to handle high query and write loads using time-indexed data

Commented [3]: administrative user interface and visualization engine.

Commented [4]: native data processing engine. can process both batch and stream data

name=passengers				
time	minors	adults	location	driver
2015-08-18T00:00:00Z	1	2	1	doe
2015-08-18T00:00:00Z	2	2	1	jones
2015-08-18T00:06:00Z	1	1	1	doe
2015-08-18T00:06:00Z	0	1	1	jones
2015-08-18T05:54:00Z	0	2	2	doe
2015-08-18T06:30:00Z	2	2	2	doe
2015-08-18T06:06:00Z	3	1	2	jones
2015-08-18T06:30:00Z	0	4	2	jones

Table 1: Sample time series dataset.

- Columns after the time are "**fields**". Field is a key value pair that records metadata and actual data value. Values are always associated the time stamp.
    - **Why field/tag**: Field sets are not indexed and so they will not require another data structure. Tags will require a separate data structure but in the long run, if you are frequently performing queries on the tag (example, if you frequently select rows where driver is 'jones', the query performance will be greatly increased.
    - **Why not make everything tags:** tags are always interpreted as strings, so you will not be able to perform many InfluxQL functions on them like MEAN(), MIN(), SPREAD() etc
- Columns after fields are "**tags**". Tags are indexed, so queries on tags are faster than queries on fields, but they are optional pieces of data.
    - Can be thought of as metadata
    - Always interpreted as strings
- Field values are **measurements** (describes the data stored in the fields). A single measurement can belong to multiple retention policies.
- **Retention policy(RP)**: how long InfluxDB stores data and how many copies (**replication factor**) of the data is stored in the cluster.
    - Autogen: infinite duration and replication of 1
- **Series:** a collection of data with common retention policy.
    - Made up of field set, measurement, tag set (
- **Point:** field set in the same series with a specific time stamp.

- **Sharding:**
    - Horizontal partitioning of data in the database. Each partition is a shard.
    - InfluxDB stores data in shard groups, organized by retention policy (RP) and time interval (which also depends on the RP).

Commented [5]: "minors, adults" is the field set
minors is the key
adults is the value

Commented [6]: string

Commented [7]: any data type

Commented [8]: location, driver

- **Shard duration:** how much time each shard spans
- **Shard group:** logical containers of shards separated by time and retention policy
- Google database sharding

- **Storage engine:**
  - InfluxDB uses an time structured merge tree (TSM tree)
    - How does this relate to fields/tags
  - TSM tree is similar to log structured merge tree (LSM tree)
    - LSM tree is basically

Commented [9]: explained here: https://priyankvex.wordpress.com/2019/04/28/introduction-to-lsm-trees-may-the-logs-be-with-you/

Resources:
https://docs.influxdata.com/influxdb/v1.7/concepts/glossary/#tag

https://www.alibabacloud.com/blog/a-comprehensive-analysis-of-open-source-time-series-databases-3_594732#:~:text=Data%20sharding%3A%20Data%20is%20divided,being%20hashed%20to%20multiple%20shards

https://blog.yugabyte.com/how-data-sharding-works-in-a-distributed-sql-database/#:~:text=Sharding%20is%20the%20process%20of,portion%20of%20the%20overall%20workload