

# A Few Words about Time Series Databases

University of Fribourg  
Department of Informatics  
Software Engineering Group



October 15, 2020

# Introduction
















- Time series are collections of data points associated with a timestamp.
- Time series databases (TSDB) are databases specialized in managing time series. Their requirements are:
  - High transaction volume
  - Advanced, efficient aggregators
- Typically, TSDBs require data to be structured in a fixed way.
  - <name> <time> <value>

- Is an open-source, in-memory key-value store, written in C.
- While not originally a TSDB, Redis can handle time series using streams (append-only data structures).
- Advantages are speed and ease of use, with a very small learning curve.
- It has built-in pub-sub features, so you can use Redis as a synchronization primitive.
- Nodejs client: [https://github.com/NodeRedis/node\\_redis](https://github.com/NodeRedis/node_redis)
- AsyncIO client: <https://github.com/aio-libs/aioredis>

- Is an open-source DB, written in GO
- Uses SQL-like queries
- Has an HTTP API as well as multiple client libraries, including NodeJS and Python (asyncio)
  - <https://github.com/node-influx/node-influx>
  - <https://github.com/gusutabopb/aioinflux>
- Can automatically manage data over long-term periods (data downsampling or expiring + summarizing)

# InfluxDB

## Time Series DB Ranking

Rank			DBMS	Database Model	Score		
Oct 2020	Sep 2020	Oct 2019			Oct 2020	Sep 2020	Oct 2019
1.	1.	1.	InfluxDB 	Time Series	24.15	+0.81	+4.52
2.	2.	2.	Kdb+ 	Time Series, Multi-model 	7.66	+0.24	+2.23
3.	3.	3.	Prometheus	Time Series	5.33	-0.36	+1.73
4.	4.	4.	Graphite	Time Series	4.36	+0.06	+1.02
5.	5.	5.	RRDtool	Time Series	3.19	+0.14	+0.48
6.	6.	 8.	TimescaleDB 	Time Series, Multi-model 	2.91	+0.18	+1.40
7.	 8.	7.	Apache Druid	Multi-model 	2.39	+0.10	+0.54
8.	 7.	 6.	OpenTSDB	Time Series	2.29	-0.01	+0.37
9.	9.	 11.	FaunaDB 	Multi-model 	1.79	-0.07	+1.31
10.	10.	10.	GridDB 	Time Series, Multi-model 	0.83	+0.09	+0.30

source: <https://db-engines.com/en/ranking/time+series+dbms>

- A very useful feature is the notion of continuous query, which allows you to downsample data to reduce memory usage

```
CREATE CONTINUOUS QUERY "cq_basic" ON "transportation"  
BEGIN  
  SELECT mean("passengers") INTO "average_passengers" FROM "bus_data" GROUP BY time(1h)  
END
```

- The query executes each unit of time (defined by the “group by” clause).
  - If the interval is one hour, the query is executed at the beginning of each hour, not when the query is created.

- Retention Policies (RP) allow you to specify how long the data should be kept.

```
CREATE RETENTION POLICY "two_hours" ON "food_data" DURATION 2h REPLICATION 1 DEFAULT  
CREATE RETENTION POLICY "a_year" ON "food_data" DURATION 52w REPLICATION 1
```

- Set a default policy, and additional custom ones.
- You can then store downsampled data using longer RPs, thus having a good trade-off on memory vs precision
  - Full example:  
[https://docs.influxdata.com/influxdb/v1.8/guides/downsample\\_and\\_retain/](https://docs.influxdata.com/influxdb/v1.8/guides/downsample_and_retain/)

# Your Project

- Using a TSDB is a good idea for your project, because of the time-related sensor data of the Thingy:91.
  - If you save all sensor data several times per minute or second, you may want to use downsampling and RP features.