

ICT for Building Design: Time series data base for IoT buildings data



**POLITECNICO
DI TORINO**

Lorenzo Bottaccioli

DIST - Interuniversity Department
of Regional and Urban Studies and Planning

What is InfluxDB?

It is a **Time Series Database**!

Time Series Databases, as their name state, are **database systems** specifically designed to handle **time-related data**.

Time series vs Relational databases

- Relational databases are based on tables.
- Tables contain columns and rows, each one of them defining an entry in your table.
- Tables are specifically designed for a purpose : one may be designed to store users, another one for photos and finally for videos.
- Such systems are efficient, scalable and used by plenty of giant companies having million of requests on their servers.

Time series vs Relational databases

Time series databases work differently. Data are still stored in 'collections' but those collections share a common denominator : **they are aggregated over time.**

Classic Relational Databases

Name ▲	Age ◆	Nickname	Employee ▼
Giacomo Guilizzoni Founder & CEO	40	Peldi	●
Marco Botton Tuttofare	38		☑
Mariah MacLachlan Better Half	41	Patata	☐
Valerie Liberty Head Chef	46	Val	☑

Data are multidimensional

Time Series Databases

Sensor Temperature ▲	Time	
39.5	12/04/19 @ 14:12	
41.2	12/04/19 @ 14:13	
12.4	14/04/19 @ 12:15	
18.5	16/04/19 @ 10:05	

Data are aggregated over time



Why do we need time series databases?

Three words : **fast ingestion rate.**

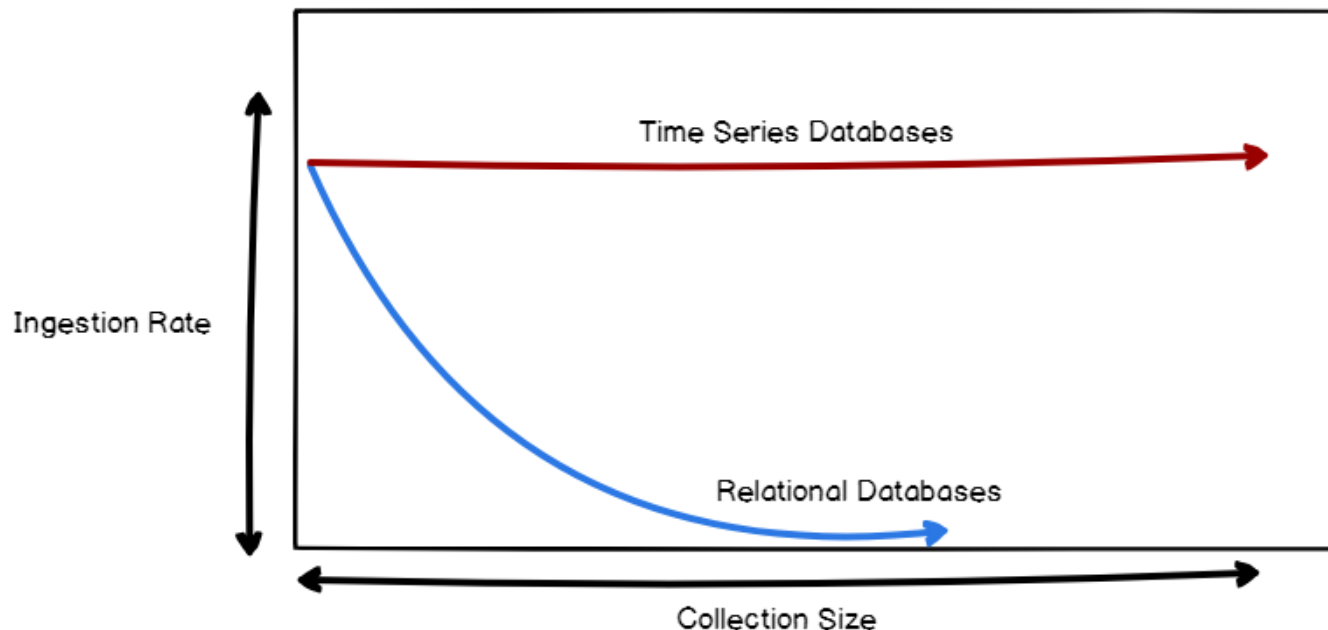
Time series databases systems are built around the predicate that they need to ingest data in a fast and efficient way.

Relational databases do have a fast ingestion rate for most of them, from 20k to 100k rows per second. However, the ingestion is not constant over time. Relational databases have one key aspect that make them slow when data tend to grow : **indexes**.

When you add new entries to your relational database, and if your table contains indexes, your database management system will repeatedly re-index your data for it to be accessed in a fast and efficient way. As a consequence, **the performance of your DBMS tend to decrease over time**. The load is also increasing over time, resulting in having difficulties to read your data.

Time series database are optimized for a **fast ingestion rate**. It means that such index systems are optimized to index data that are aggregated over time.

DBMS & TSDB Difference



In general, relational databases are surrounded by applications : web applications, softwares that connect to it to retrieve information or add new entries. Clients connect to a website, that contacts a database in order to retrieve information.

TSDB are built for **client plurality** : you do not have a simple server accessing the database, but a bunch of different sensors (for example) inserting their data at the same time.

InfluxQL is a query language that is very similar to SQL and that allows any user to query its data and filter it. Here's an example of an InfluxQL query :



```
SELECT * FROM cpu_metrics WHERE time < now() - 10m
```

InfluxDB Key Concepts

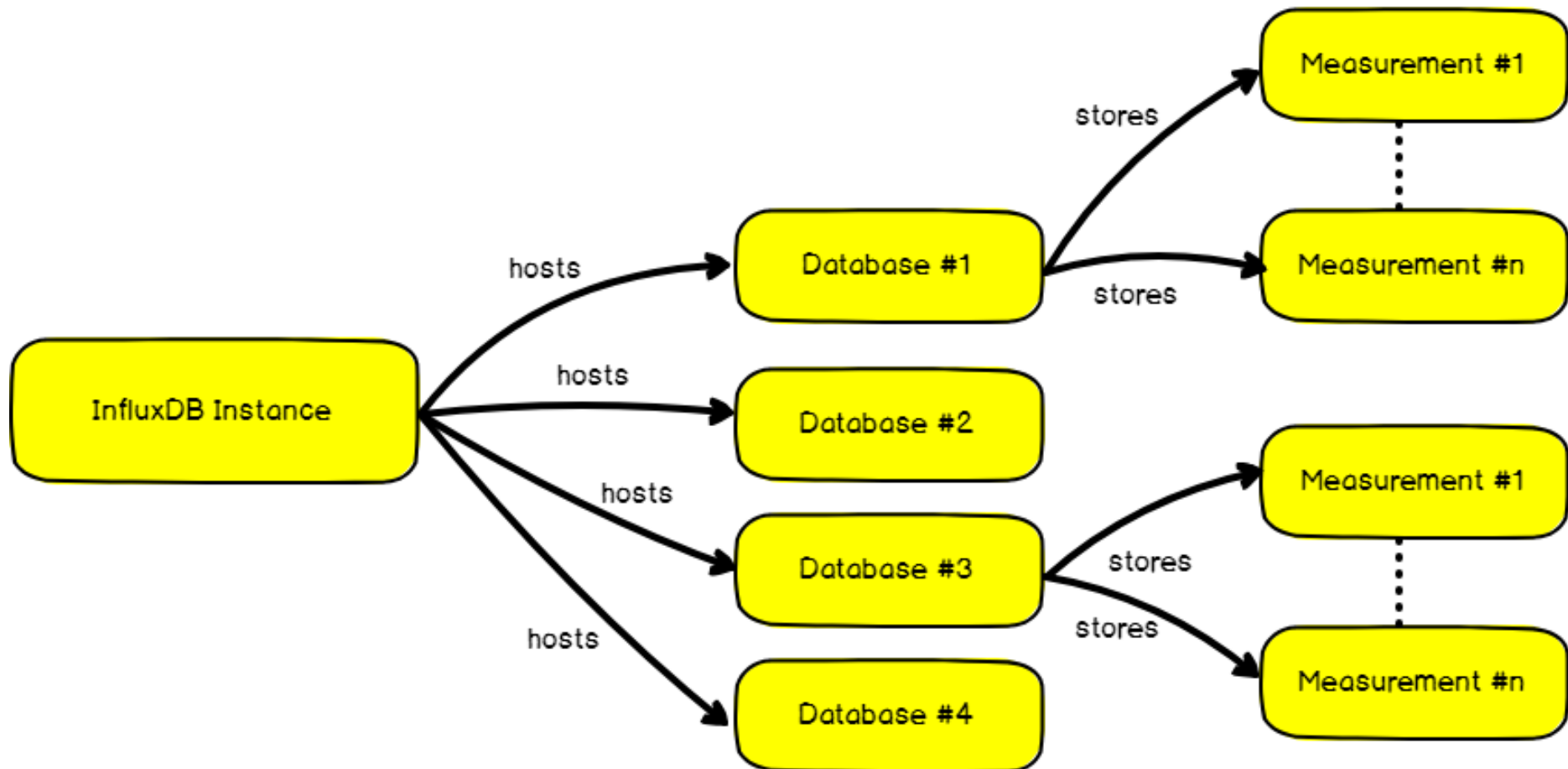
database? tags? point?
fields? timestamp?
measurement? retention policy?

Database

In a SQL environment, a database would host a collection of tables, and even schemas and would represent one instance on its own.

In InfluxDB, **a database host a collection of measurements**. However, a single InfluxDB instance can host multiple databases. This is where it differs from traditional database systems.

InfluxDB Internals



The most common ways to interact with databases are either **creating a database** or by **navigating into a database** in order to see collections (you have to be “in a database” in order to query collections, otherwise it won’t work).

Most used IFQL database queries



```
CREATE DATABASE "devconnected"
```

} Creating a database



```
USE devconnected;
```

} Navigating into a database

Measurement

You could think of a measurement as a SQL table. **It stores data, and even meta data, over time.** Data that are meant to coexist together should be stored in the same measurement.

Measurement #1

cpu_metrics

Sensor	Temperature	time
Sensor 1	40	12/14 @ 15:16 pm
Sensor 1	38	13/14 @ 11:16 am
Sensor 1	41	12/11 @ 11:16 am
Sensor 1	38	11/11 @ 12:14 am

Data are aggregated over time!



```
SELECT * FROM cpu_metrics WHERE temperature='40'
```

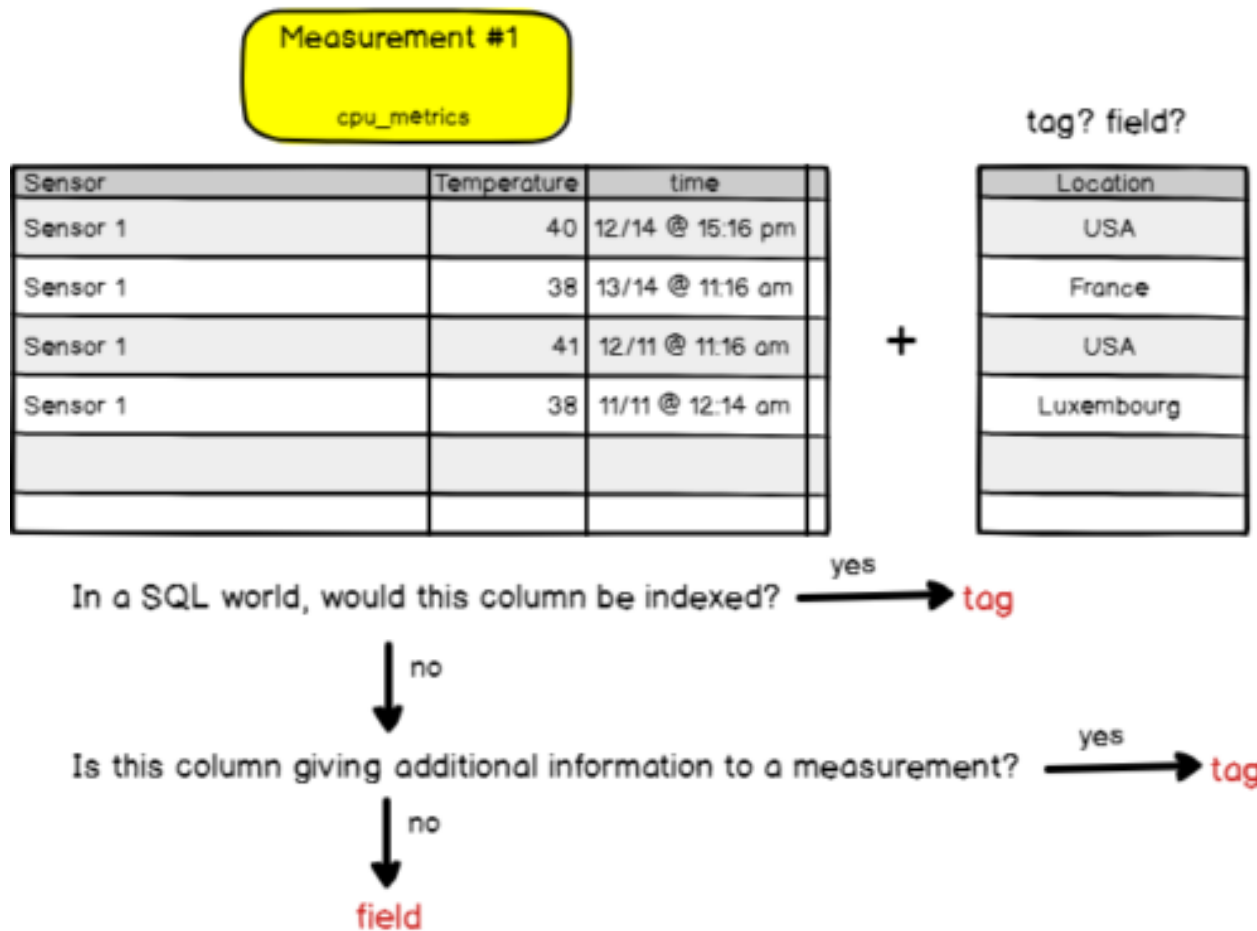
Tags & Fields

In a SQL world, data are stored in columns, but in InfluxDB we have two other terms : **tags & fields**.

When defining a new 'column' in InfluxDB, you have the choice to either declare it **as a tag or as a value** and it makes a very big difference.

Difference between the two is **that tags are indexed and values are not.**

Tags can be seen as **metadata** defining our data in the measurement. They are hints giving additional information about data, but not data itself.



In our case, it would be added as a.. **tag**! We definitely want the location 'column' to be **indexed** and taken into account when performing a query over the location.

A set of tags is called a “**tag-set**”. The ‘column name’ of a tag is called a “**tag key**”. Values of a tag are called “**tag values**”. The same taxonomy repeats for fields.

Measurement #1

cpu_metrics

tag set

Sensor	Location	Temperature	time
Sensor 1	USA	40	12/14 @ 15:16 pm
Sensor 1	France	38	13/14 @ 11:16 am
Sensor 1	USA	41	12/11 @ 11:16 am
Sensor 1	Luxembourg	38	11/11 @ 12:14 am

tag key

tag value

Timestamp

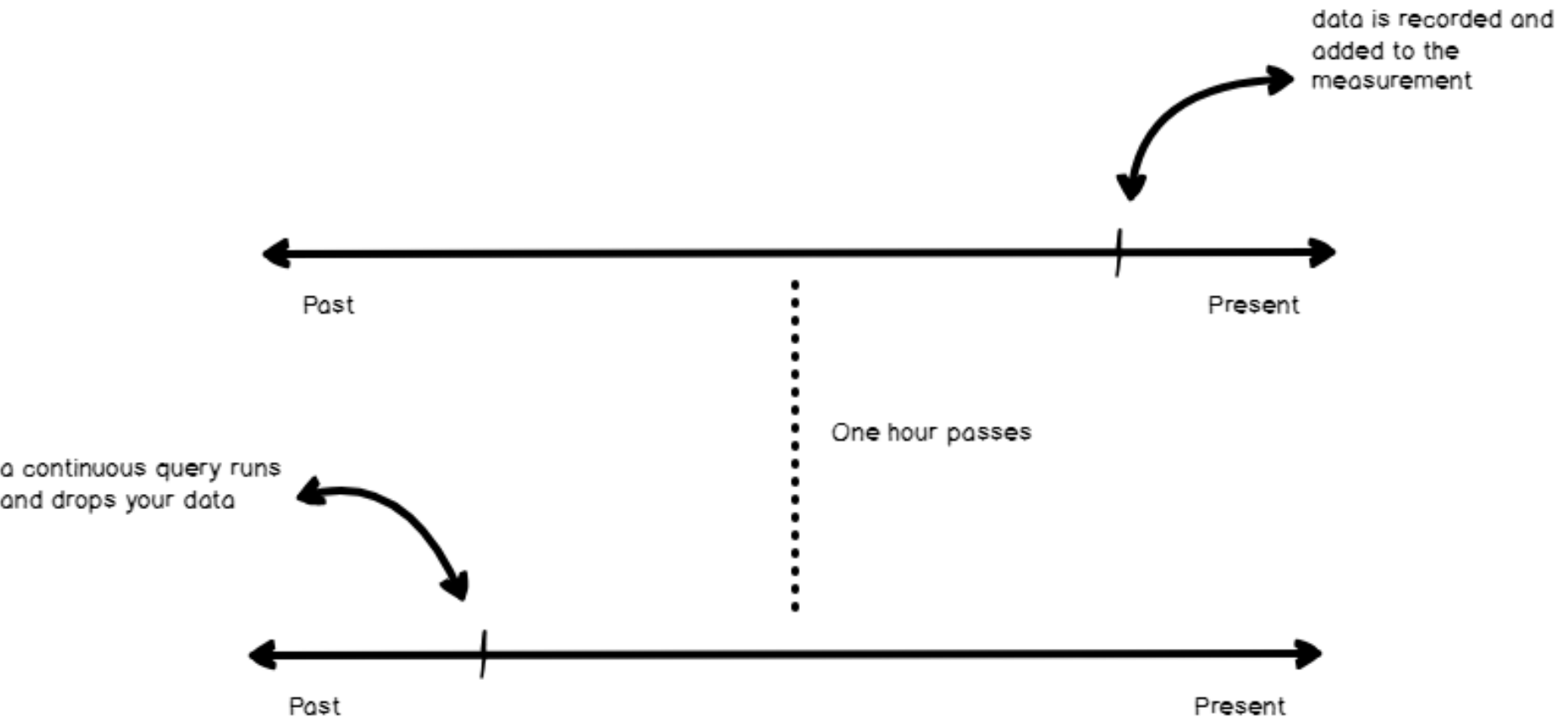
Probably the simplest keyword to define. A timestamp in InfluxDB is **a date and a time defined in RFC3339 format**. When using InfluxDB, it is very common to define your time column as a timestamp in Unix time expressed in nano seconds.

Retention policy

A retention policy defines how long you are going to keep your data. Retention policies are defined per database and of course you can have multiple of them. By default, the retention policy is '**autogen**' and will basically keep your data forever. In general, databases have multiple retention policies that are used for **different purposes**.

Retention policy

Case : retention policy = 1 hour



Point

Finally, an easy one to end this chapter about InfluxDB terms. **A point is simply a set of fields that has the same timestamp.** In a SQL world, it would be seen as a row or as a unique entry in a table. Nothing special here.