Gianluca Sartori

Time Series for Relational People

Gianluca Sartori

Founder at Quantumdatis

Data Platform MVP

Works with SQL Server since version 7

DBA @ Scuderia Ferrari

Blog: spaghettidba.com

Twitter: @spaghettidba

Agenda

- What is a Time Series database?
- Options on the market
- InfluxDB
  - Core concepts
  - Installation
  - Working with data in InfluxDB
  - Querying
- Practical uses

What is a Time Series database?

Definitions

What is **time series data**?

> *"Time series data (or time-stamped data) is a collection of observations for a single object (entity) at different time intervals."*

What is a **time series database**?

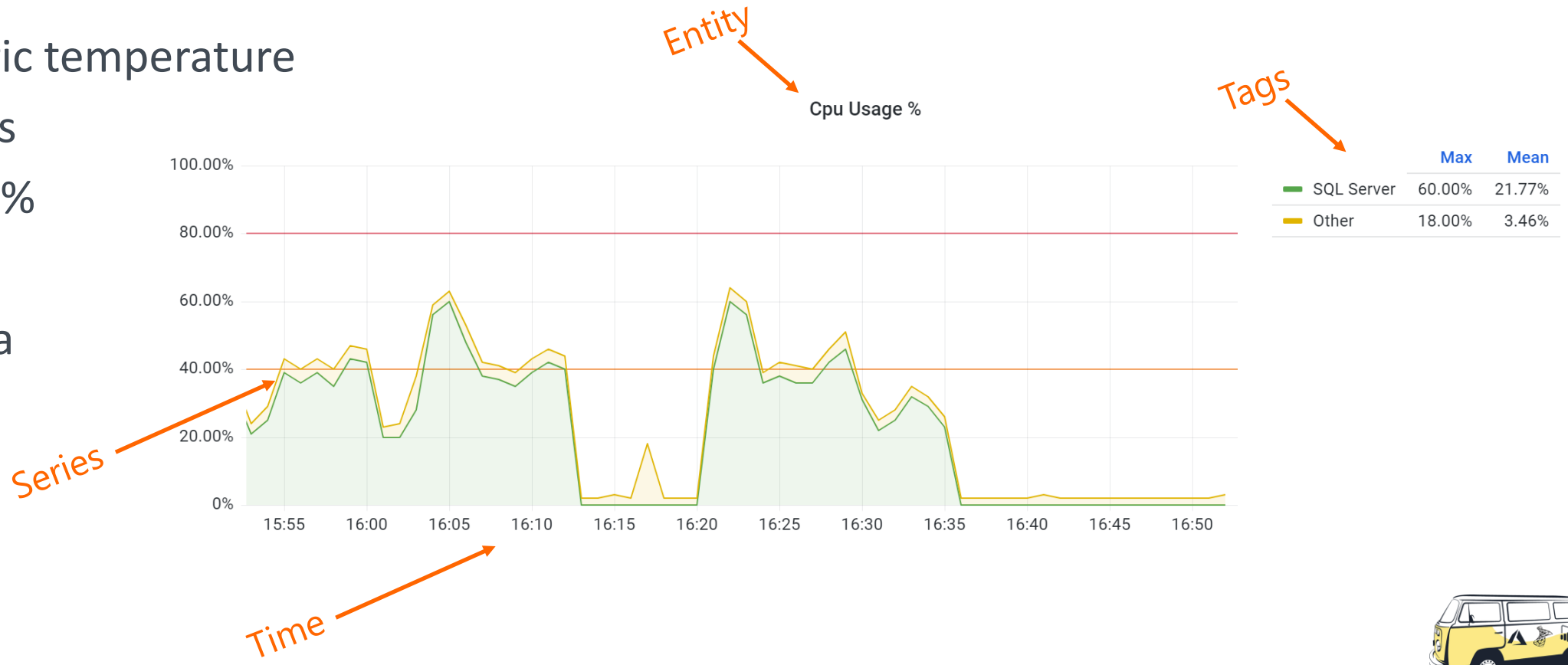
> *"A time series database (TSDB) is a database optimized for **time series data** and for measuring change over time."*

Time Series Data

Time series data is obtained by performing repeated measurements over time

Examples:

- Atmospheric temperature
- Stock prices
- CPU usage %
- Emails/sec
- Sensor data



#DataWeekender

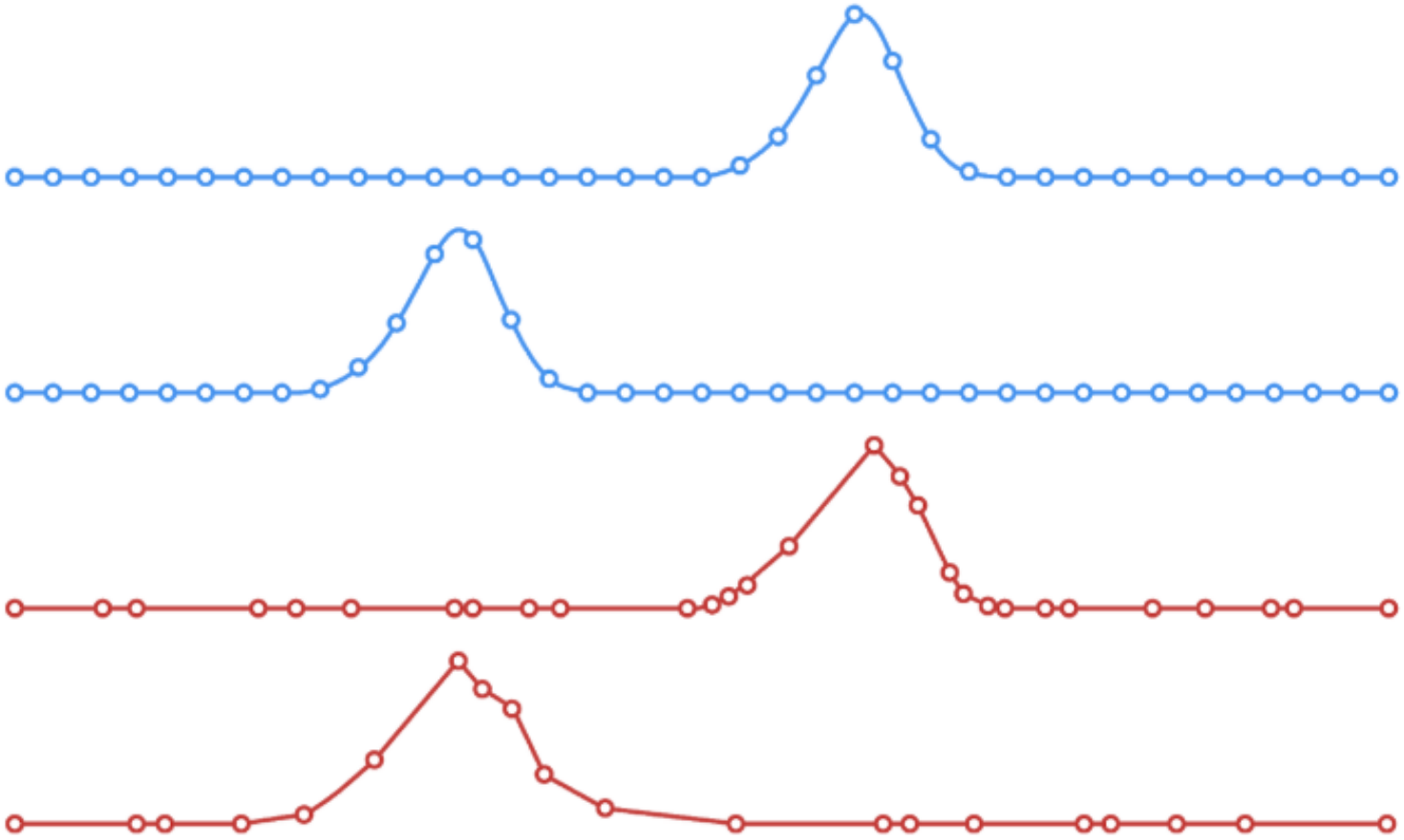
Time Series Data

Metrics (Regular)

Measurements gathered at regular time intervals

Events (Irregular)

Measurements gathered at irregular time intervals



#DataWeekender

Time Series Data

How is this different from relational data?

- Continuous Stream of data in time order
- Bulk uploads of large sets of data
- High volume data
- Data is append-only – no updates
- Delete large volumes of data when it goes out of scope
- Downsampling and aggregating high resolution data to save space

Should I use SQL Server?

Relational databases do not deal very well with time series data

→No built-in storage optimizations

→No easy retention policies enforcement

→No time series specific query capabilities

- Azure SQL Edge is designed for time series data

  See https://docs.microsoft.com/en-us/azure/azure-sql-edge/
- SQL Server 2022 will have time series capabilities from Azure SQL Edge

  See https://techcommunity.microsoft.com/t5/microsoft-mechanics-blog/what-s-new-in-sql-server-2022/ba-p/2922227

Time Series Database Options

RANK	DBMS	SCORE		
APR 2022		APR 2022	24 MOS ▲	12 MOS ▲
1	**InfluxDB**	**30.02**	**+9.10**	**+2.85**
2	Kdb+	8.78	+3.41	+0.52
3	Prometheus	6.31	+2.00	+0.55
4	Graphite	5.36	+1.90	+0.80
5	TimescaleDB	4.56	+2.79	+1.66
6	ApacheDruid	3.18	+1.27	+0.51
7	RRDtool	2.58	+0.06	+0.12
8	OpenTSDB	1.82	0.00	+0.02
9	DolphinDB	1.62	+1.23	+0.72
10	Fauna	1.42	+0.47	-0.07

Source: DB-Engines

39 Systems in Ranking, April 2022

InfluxDB

influxdata®

Most popular time series database

Designed and optimized for time series data

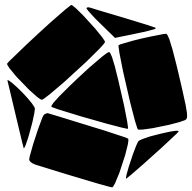
- Uses optimized TSM storage engine with columnar compression
- Built-in retention policies

Open Source with permissive licensing (MIT)

Written in GO

Cross platform: *windows, linux, macOS, docker, kubernetes, raspberryPI*

#DataWeekender



influxdb

Different SKUs

- *Cloud*
  - → Hosted by influxdata
- *OSS 2.X*
  - → Current supported version
  - → More powerful
- *OSS 1.X*
  - → Currently deprecated
  - → **Easier to use for relational people**

InfluxDB design principles

Time series data:

Series are stored in **measurements**

Measurements can have **fields** and **tags**

Series are sets of values of a single field, measured over time

Key of a series = `[measurement name, time, tags, field name]`

Example: CPU Usage

`CPUHistory` ← measurement name

`Time`

`ServerName` ← tag

`CPUPercent` ← field

Series – An Example

SQL Server CPU usage percent from `sys.dm_os_ring_buffers`

time	SQL Server CPU	Other CPU
08:00	10	3
08:01	15	2
08:02	18	6
08:03	73	1

In a relational database this can be saved to a table:

```
CREATE TABLE CPUHistory (
    [time] time PRIMARY KEY,
    SQLServerCPU tinyint,
    OtherCPU tinyint
)
```

Series – An Example

In InfluxDB this is saved in a **measurement** with **two series:**

time	SQL Server CPU	Other CPU
08:00	10	3
08:01	15	2
08:02	18	6
08:03	73	1

[CPUHistory, SQL Server CPU]

time	SQL Server CPU
08:00	10
08:01	15
08:02	18
08:03	73

[CPUHistory, Other CPU]

time	Other CPU
08:00	3
08:01	2
08:02	6
08:03	1

Series – Tags

If we add **tags**, we will get **one series for each tag value:**

time	**ServerName**	SQL Server CPU	Other CPU
08:00	ACCOUNTING	10	3
08:00	CRM	37	7
08:01	ACCOUNTING	15	2
08:01	CRM	48	5
08:02	ACCOUNTING	18	6
08:02	CRM	41	9
08:03	ACCOUNTING	73	1
08:03	CRM	28	7

[CPUHistory, **ACCOUNTING**, SQLServerCPU]

time	SQL Server CPU
08:00	10
08:01	15
08:02	18
08:03	73

[CPUHistory, **ACCOUNTING**, OtherCPU]

time	Other CPU
08:00	3
08:01	2
08:02	6
08:03	1

[CPUHistory, **CRM**, SQLServerCPU]

time	SQL Server CPU
08:00	37
08:01	48
08:02	41
08:03	28

[CPUHistory, **CRM**, OtherCPU]

time	Other CPU
08:00	7
08:01	5
08:02	9
08:03	7

InfluxDB – Storage Engine

- Series are stored independently
- We get one series for each combination of tag values → **cardinality**
- Does this look familiar? [Hint: columnstore indexes in SQL Server]

- Storage engine called **TSM** (Time series Structured Merge tree)
  - derived from **LSM** (Log-Structured Merge tree) → **Cassandra**
  - stores data in columns (series)
  - groups writes in memory and writes to disk in append mode
  - data on disk is always appended, never updated
  - series are compressed at regular intervals

Working with InfluxDB - Installation

- Download the binaries
- Expand the archive
- Check configuration
  - Port
  - Log file location
  - Authentication
- Run the deamon

DEMO!

Databases, retention policies, buckets…

SQL Server

DATABASE

SALES

customers
orders
…

PRODUCTION

products
models
…

Table belongs to a single schema

InfluxDB 1.X

DATABASE

retention policies

measurements

	short	long	temp
CPUHistory	●	●	◌
DiskLatency	●	◌	◌
…			

Measurement exists on all retention policies, you can write to whichever you need

InfluxDB 2.X

ORGANIZATION

BUCKET: short

retention period: 15d

CPUHistory
DiskLatency
…

BUCKET: long

retention period: 180d

CPUHistory
WaitStats
…

Measurement belongs to a single bucket

#DataWeekender

Working with InfluxDB – http API

InfluxDB can be queried and controlled with **http endpoints**

Endpoints are documented on the reference page

1.X and 2.X have different sets of APIs

1.X	2.X
v1 API	v2 API
v2 compatibility API	v1 compatibility API

Endpoints:

ping	Check status and version
query	Query using **InfluxQL** or **Flux** (depending on version) Manage users, databases, …
write	Write data

Working with InfluxDB – CLI

InfluxDB ships with a command-line client, **influx.exe**

It is nothing but a client for the http API

Accepts several parameters:

```
-host                    -execute 'command'
-port                    -type 'influxql|flux'
-database                -format 'json|csv|column'
-username                -import
-password                …
```

DEMO!

Query tools

There’s more than the CLI: several query tools available:

- **Chronograf**
  Official query and dashboard tool by Influxdata. Works in the browser
  https://www.influxdata.com/time-series-platform/chronograf/
- **Time Series Admin**
  Open Source query tool
  https://timeseriesadmin.github.io/

Working with InfluxDB – Writing data

- Two ways to write to the database:
  - `/write` endpoint
  - using the CLI and the INSERT command
- Both expect the data to insert in a particular format called **line protocol**

DEMO!

```
CPUHistory,server_name=ACCOUNTING SQLServerCPU=31,OtherCPU=7 1652386888000000000000
```

Working with InfluxDB – Writing data

Probably you don’t want to write individual points → batches

```
CPUHistory,server_name=ACCOUNTING SQLServerCPU=31,OtherCPU=7 1652386888000000000
CPUHistory,server_name=ACCOUNTING SQLServerCPU=48,OtherCPU=9 1652386903000000000
CPUHistory,server_name=ACCOUNTING SQLServerCPU=53,OtherCPU=7 1652386918000000000
CPUHistory,server_name=ACCOUNTING SQLServerCPU=47,OtherCPU=3 1652386933000000000
...
```

Data types (for fields and tags):

numeric, string, integer, boolean

Timestamp is always in epoch format, from second to nanosecond

DEMO!

Working with InfluxDB – Writing data with telegraf

Telegraf is a data collection agent

Works with plugins :

INPUT:
Apache
Cassandra
Docker
Elasticsearch
Kubernetes
Nginx
PostgreSQL
SQL Server
… and more!

OUTPUT:
InfluxDB
Elasticsearch
File
Graphite
Graylog
Kafka
MongoDB
SQL
… and more!

DEMO!

Querying data with InfluxQL

InfluxDB supports 2 query languages:

InfluxQL → similar to SQL, born with version 1, still available in v2

```
SELECT MEAN(SQLServerCPU) AS avg_cpu
FROM CPUHistory
WHERE time > now() -1d
GROUP BY server_name
```

Querying data with Flux

Flux → doesn't look like SQL, default in v2, also available in v1

Supports «advanced» query features like joins…

```
from(bucket:"demo/autogen" )
    |> range(start: -1d)
    |> filter(fn: (r) => r._measurement == "CPUHistory")
    |> filter(fn: (r) => r._field == "SQLServerCPU")
    |> mean()
    |> group(columns: ["server_name"])
    |> yield(name: "avg_cpu")
```

InfluxQL: SELECT syntax

```
SELECT <fields>
    FROM <retention_policy>.<measurement>
    [ INTO <retention_policy>.<measurement> ]
    [ WHERE <search_condition> ]
    [ GROUP BY <tags> [ FILL(<fill_expression>) ] ]
    [ ORDER BY <order_by_expression> ]
    [ LIMIT <number> ]
    [ OFFSET <number> ]
    [ SLIMIT <number> ]
    [ SOFFSET <number> ]
    [ TZ(<timezone_clause>) ]
```

DEMO!

InfluxQL: what about INSERT, UPDATE and DELETE?

INSERT does not exist

The write endpoint takes care of inserts

UPDATE does not exist.

To update a measurement, upload the data again.

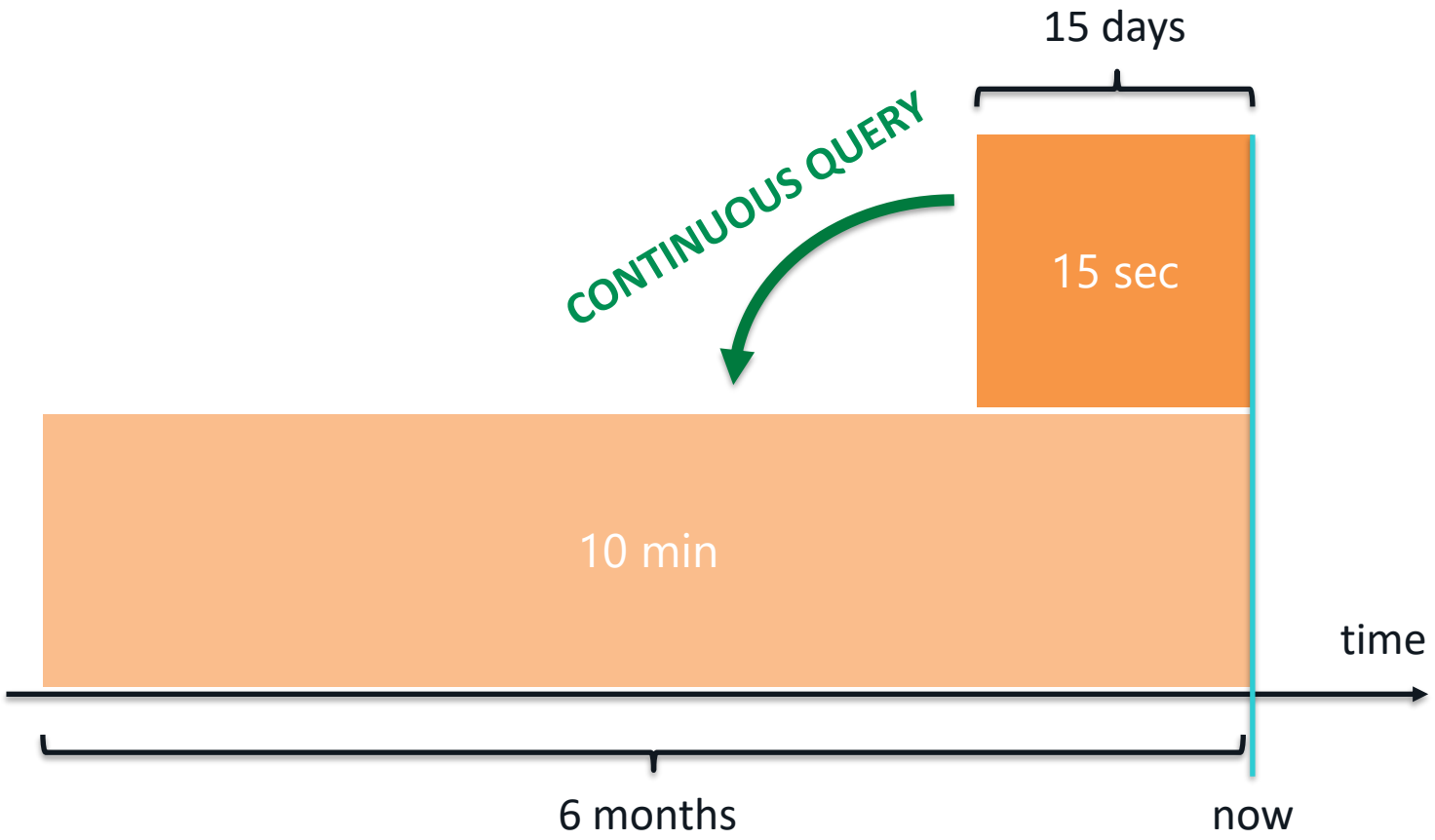
Points that match the key [time, tags] will be updated

DELETE works pretty much like SQL

DEMO!

Retention policies and downsampling

- Default **retention policy** → ∞
- Multiple retention policies, different retention periods
- **Continuous queries** aggregate and downsample the data to long term retention policy



#DataWeekender

Putting it all together

- Upload metrics using **Telegraf**
- Create a database in **InfluxDB**
- Display the data using **Grafana**

} **TIG** stack

DEMO!

- Open source analytics/dashboarding platform
- Native InfluxDB support
- Powerful visualizations
- Easy to use

Conclusion

- Time series databases are great for storing and analyzing time series data
- Similarities with relational databases, but they are different beasts
- **InfluxDB** is the market leader and offers a great product with:
  - Optimized storage
  - Retention policies
  - Time series query capabilities
  - Continuous queries
- Great tool for IOT and telemetry observability solutions

Conclusion

- Time series databases can help you store, process and analyze huge volumes of time series data

Example:

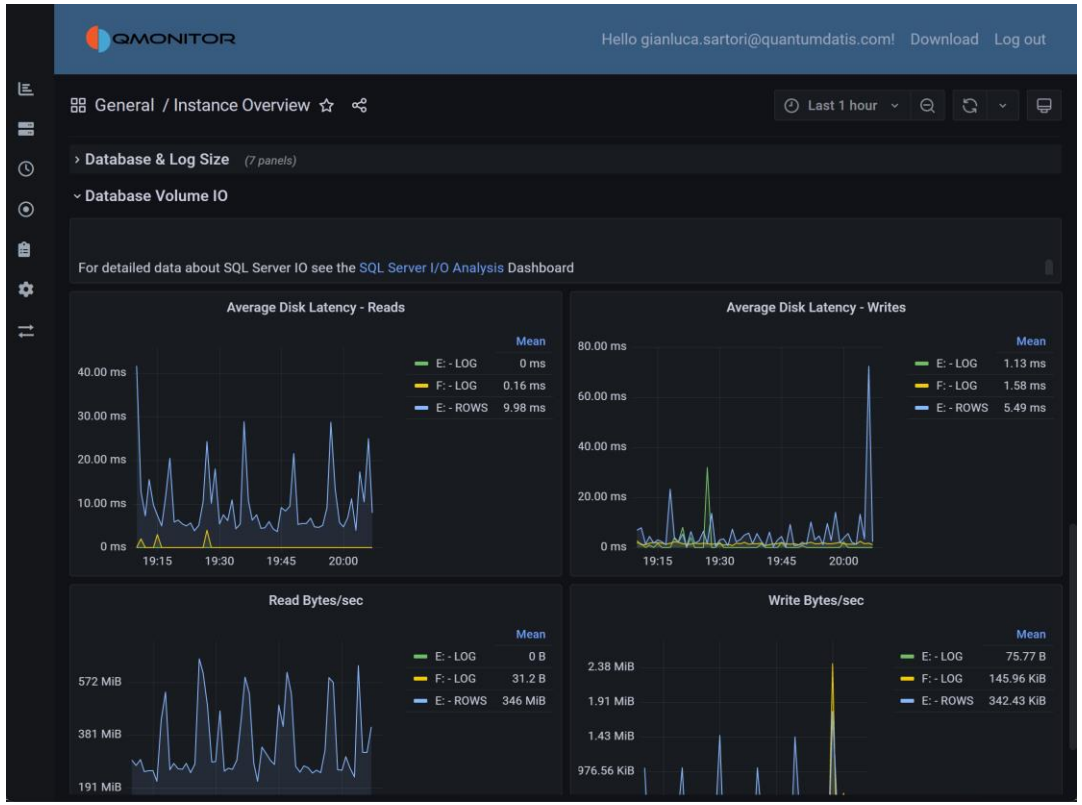
QMONITOR

It’s a complete SQL Server monitoring solution built on InfluxDB and Grafana

http://qmonitor.quantumdatis.com

User: demo@quantumdatis.com

Pass: qmonitor



#DataWeekender

Thank you!

Questions? Ask me!

spaghettidba@sqlconsulting.it