UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA
DIPARTIMENTO DI INGEGNERIA «ENZO FERRARI»

Corso di Laurea in Ingegneria Informatica

Tecnologie per l'archiviazione dei dati generati dall'Internet of Things

RELATRICE
Prof.ssa Sonia Bergamaschi

CANDIDATO
Rosario Lissandrello

ANNO ACCADEMICO 2018 - 2019

IOT: INTERNET OF THINGS

- L'Internet of Things (IoT o Internet delle cose) è un neologismo che indica l' estensione di internet al mondo delle "cose", aumentando la capacità di raccolta e di utilizzo dei dati da una moltitudine di sorgenti (prodotti industriali, sistemi di fabbrica, veicoli di trasporto, etc….) a vantaggio di una maggiore digitalizzazione e automazione dei processi.

Alcuni dei principali campi applicativi interessati dallo sviluppo del IoT sono:

- Agricoltura
- Domotica
- Reti wireless di sensori
- Smart city
- Industria

TIME-SERIES DATA

- I time-Series data sono sequenze di dati che rappresentano come un sistema, un processo o un evento si evolve con il passare del tempo.
- Consistono solitamente in misurazioni successive applicate alla stessa sorgente su un intervallo di tempo.

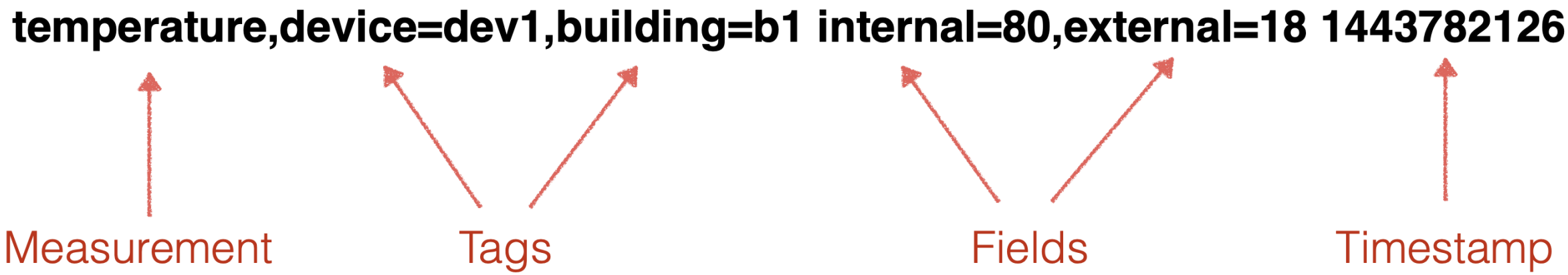
RANK	DBMS	SCORE		
NOV 2019		NOV 2019	24 MOS ▲	12 MOS ▲
1	**InfluxDB**	**19.93**	**+10.59**	**+6.29**
2	Kdb+	5.29	+3.44	+0.45
3	Prometheus	3.64	+2.83	+1.69
4	Graphite	3.32	+0.46	+0.47
5	RRDtool	2.90	-0.29	+0.17
6	OpenTSDB	2.13	+0.42	+0.11
7	Druid	1.79	+0.81	+0.43
8	TimescaleDB	1.73	+1.73	+1.19
9	FaunaDB	0.61	+0.44	+0.40
10	GridDB	0.57	+0.47	+0.40

Source: DB-Engines

23 Systems in Ranking, November 2019

- I Time-Series DBMS sono database server ottimizzati per collezionare, immagazzinare, recuperare e processare i time-Series data.

- DBMS time-series
- Open Source
- NoSQL
- Scritto in Go



MODELLO DEI DATI

VANTAGGI E SVANTAGGI DI INFLUXDB

Vantaggi

- IndexFile in-memory
- Struttura e funzioni ad hoc per la gestione dei dati time-series
- Politiche di conservazione
- Compressione on-disk migliore dei DBMS relazionali

Svantaggi

- Impossibilità di effettuare JOIN
- Linguaggio NoSQL
- Query sui campi valori molto lente
- Impossibilità di eliminare/modificare i singoli valori
- Difficoltà di migrazione dei dati da un database relazionale

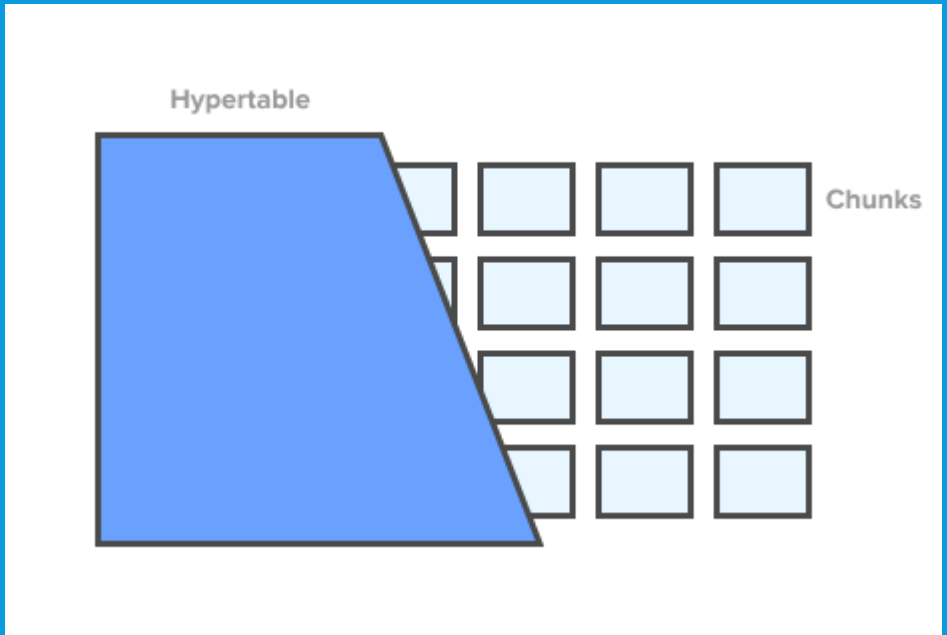
TIMESCALEDB

- Time-Series, Relational DBMS
- Estensione di PostgreSQL

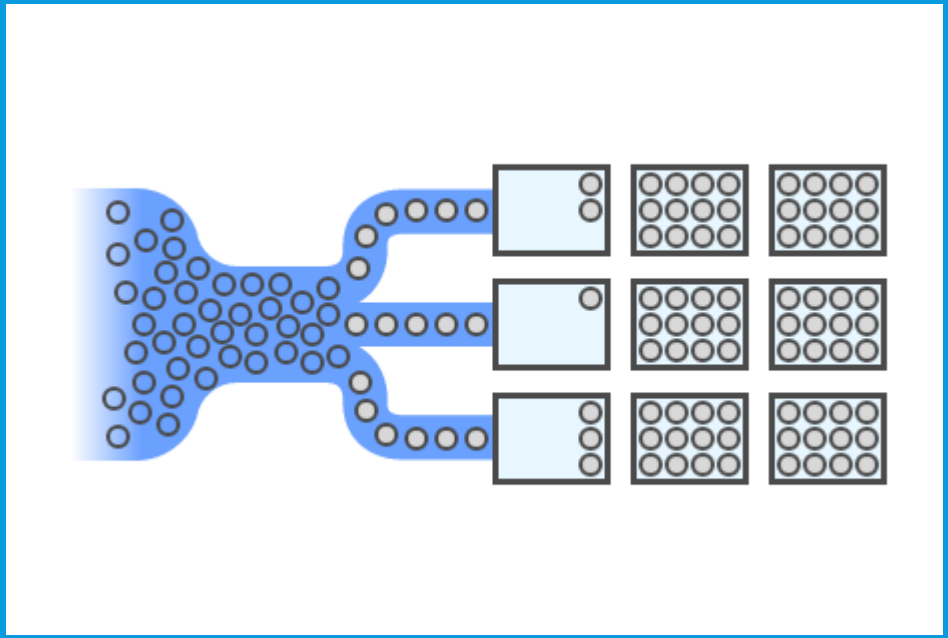
Timescale

ARCHITETTURA

Hypertable



Chunks



TIMESCALEDB VS POSTGRESQL

Un ingest-rate molto più elevato, soprattutto nel caso di database molto grandi.

Prestazioni delle query simili o superiori in alcuni casi.

(Es. query basate sull'ordinamento temporale)

Caratteristiche "time-oriented".

(ES. politiche di conservazione)

VANTAGGI E SVANTAGGI DI TIMESCALEDB

Vantaggi

- Utilizzo del linguaggio SQL
- Utilizzo di un solo database per dati relazionali e time-series
- Possibilità di effettuare join tra tabelle postgres e hypertable
- Supporto nativo per i dati geospaziali

Svantaggi

- In alcune query risulta più lento di InfluxDB

INFLUXDB VS TIMESCALEDB

InfluxDB

- IndexFile in-memory
- Tag indexati
- Struttura e funzioni ad hoc per il trattamento di dati time-series
- Compressione sul disco migliore

TimescaleDB

- Maggiore scalabilità
- Possibilità di utilizzare molti tipi di dati differenti
- Possibilità di lavorare con dati relazionali oltre che time-series nello stesso database
- Possibilità di migrare i dati Influx grazie ad Outflux

WHERE

Database:	InfluxDB	TimescaleDB
Query:	SELECT * FROM "h20_quality" WHERE location = 'santa_monica'	SELECT * FROM h20_qualityTS WHERE location like 'santa_monica'
Execution query runtime:	65 ms	90 ms

Database:	InfluxDB	TimescaleDB
Query:	SELECT * FROM "h20_quality" WHERE “index” > 40	SELECT * FROM h20_qualityts WHERE index > 40
Execution query runtime:	105 ms	90 ms

GROUP BY

Database:	InfluxDB	TimescaleDB
Query:	SELECT MEAN("water_level") FROM "h2o_feet" GROUP BY "location"	SELECT AVG(water_level) FROM h2o_feetTS GROUP BY location
Execution query runtime:	25 ms	60 ms

Database:	InfluxDB	TimescaleDB
Query:	SELECT MEAN("water_level") FROM "h2o_feet" WHERE "location"='coyote_creek' AND time >= '2015-08-18T00:06:00Z' AND time <= '2015-08-18T00:54:00Z' GROUP BY time(18m)	SELECT TIMESTAMP WITH TIME ZONE 'epoch' + INTERVAL '1 second' * round(extract('epoch' from time) / 1080) * 1080,AVG(water_level) FROM h2o_feetTS WHERE location='coyote_creek' AND time >= '2019-08-18T00:06:00Z' AND time <= '2019-08-18T00:54:00Z' GROUP BY round(extract('epoch' from time) / 1080)
Execution query runtime:	3 ms	60 ms

ORDER BY

Database:	InfluxDB	TimescaleDB
Query:	SELECT "water_level" FROM "h2o_feet" WHERE "location" = 'santa_monica' ORDER BY time DESC	SELECT water_level FROM h2o_feetTS WHERE location = 'santa_monica' ORDER BY time DESC
Execution query runtime:	80 ms	90 ms

UTILIZZO DI QUERY GEOSPAZIALI

```
SELECT sensor_channel_id,time
FROM prova_ts2
WHERE ST_Distance(geom,  ST_Transform  (ST_SetSRID
(ST_MakePoint(44.493804,11.342798),4326),2163)) < 40000
```

dbtesi/postgres@tesi

Query Editor

```
1
2   select sensor_channel_id, time
3   from prova_ts2
4   WHERE ST_Distance(geom, ST_Transform(ST_SetSRID(ST_MakePoint(44.493804,11.342798),4326),2163)) < 40000
5
6
7
8
9
10
11
12
13
14
```

Data Output | Explain | Messages | Notifications

	sensor_channel_id integer	time [PK] timestamp with time zone
1	129	2018-02-20 02:23:01+01
2	149	2018-02-21 02:39:45+01
3	140	2018-02-21 02:40:04+01
4	261	2018-02-21 02:40:13+01
5	70	2018-02-21 02:40:32+01
6	119	2018-02-21 02:40:41+01
7	119	2018-02-21 02:40:51+01
8	69	2018-02-21 02:41:56+01
9	149	2018-02-21 02:42:15+01
10	258	2018-02-21 02:42:52+01

INFLUXDB VS TIMESCALEDB

- Che tipi di dato bisogna raccogliere?
- Quanto è importante la scalabilità del sistema?
- Quanto è importante la velocità del sistema?
- Bisogna creare il sistema da zero o bisogna partire da un database relazionale?