Time Series databases InfluxDB

Antonio Romero Marin, IT-DB

Time Series Scenario

- Time series data workload assumptions
  - Normally insert or append data workload
    - High ingestion rates in some cases
    - Recent timestamps and the data is added in time ascending order
    - Rare updates and deletes
    - Large deletes to free up disk space
  - Query operations different from RDBMs
    - Individual points are not too important
    - Aggregate data and large data sets
    - Time-centric filtering and calculations

Time Series Databases

- Some popular
  - InfluxDB
    - General purpose time series database
  - Graphite (Carbon + Whisper)
  - Kdb+
    - Propietary
  - OpenTSDB
    - Runs in Hadoop
  - Prometheus
    - Pull architecture
    - Focus on monitoring

InfluxDB overview

- Purpose Built for Time-Series
- Open source (MIT License)
- Native HTTP API
- SQL-like query language
- Schema-less
- Low hardware sizing to handle most of the use cases
  - Compression
  - Downsampling and data retention capabilities
- High availability
  - But clustering only available in Enterprise version (non-free)

InfluxDB main concepts

- Measurement
  - container for tags, fields and timestamp
  - conceptually similar to a table
- Tag
  - The key-value pair that records metadata
  - Tags are optional and they are indexed
- Field
  - Required
  - Not indexed

```
> show measurements
------------------
name
database
httpd
queryExecutor
runtime
```

```
> show tag keys from "database"
name: database
--------------
tagKey
database
hostname
```

```
> show field keys from "database"
name: database
--------------
fieldKey                fieldType
numMeasurements         integer
numSeries               integer
```

InfluxDB main concepts

- Series
  - collection of data that share a measurement and tag set

```
> show series from "database"
--------------
key
database,database=_internal,hostname=db-00000.cern.ch
```

```
> select * from "database" where time > now() - 1m
name: database
--------------
time                  database  hostname           numMeasurements  numSeries
1472657820000000000   _internal db-00000.cern.ch   12               17
1472657830000000000   _internal db-00000.cern.ch   12               17
1472657840000000000   _internal db-00000.cern.ch   12               17
```

InfluxDB interfaces

- HTTP API

Endpoint	Description
/ping	Use /ping to check the status of your InfluxDB instance and your version of InfluxDB
/query	Use /query to query data and manage databases, retention policies, and users
/write	Use /write to write data to a pre-existing database

- Influx CLI
  - interactive shell for the HTTP API

```
$ influx
Connected to https://localhost:8086 version 1.2.0
InfluxDB shell version: 1.2.0
>
>
```

- Multiple API client libs available
  - Go, Python, Java, JavaScript, Perl, .Net…

Writing data

- Syntax

```
measurement[,tag_key1=tag_value1...] field_key=field_value[,field_key2=field_value2] [timestamp]
```

- HTTP write

```
curl -i -XPOST 'http://localhost:8086/write?db=mydb'
--data-binary 'cpu_load_short,host=server01,region=us-west value=0.64 1434055562000000000'
```

- Datatypes
  - Measurements, tag keys, tag values and field keys are strings
  - Field values can be floats, integers, strings or Booleans
  - Timestamps are UNIX timestamps. Precision up to nanoseconds (default)

Querying data

- SQL-like query language

```
SELECT COUNT(water_level)
  FROM h2o_feet
 WHERE time >= '2015-08-19T00:00:00Z'
   AND time <= '2015-08-27T17:00:00Z'
   AND location='coyote_creek'
GROUP BY time(3d)
```

- HTTP API results


```
{
  "results": [
    {
      "series": [
        {
          "name": "cpu_load_short",
          "columns": [
            "time",
            "value"
          ],
          "values": [
            [
              "2015-01-29T21:55:43.702900257Z",
              0.55
            ],
            [
              "2015-01-29T21:55:43.702900257Z",
              23422
            ],
            [
              "2015-06-11T20:46:02Z",
              0.64
            ]
          ]
        }
      ]
    }
  ]
}
```

InfluxQL Functions

Aggregations	Selectors	Transformations	Predictor
COUNT	BOTTOM	CEILING	HOLT_WINTERS
DISTINCT	FIRST	CUMULATIVE_SUM	
INTEGRAL	LAST	DERIVATIVE	
MEAN	MAX	DIFFERENCE	
MEDIAN	MIN	ELAPSED	
MODE	PERCENTILE	FLOOR	
SPREAD	SAMPLE	HISTOGRAM	
STDDEV	TOP	MOVING_AVERAGE	
SUM		NON_NEGATIVE_DERIVATIVE	

Sampling and Data Retention

- Continuous query
  - Query that runs automatically and periodically
  - Require a function in the select clause and must include a “group by time()”
- Retention policy
  - Describes for how long InfluxDB keeps data (duration)

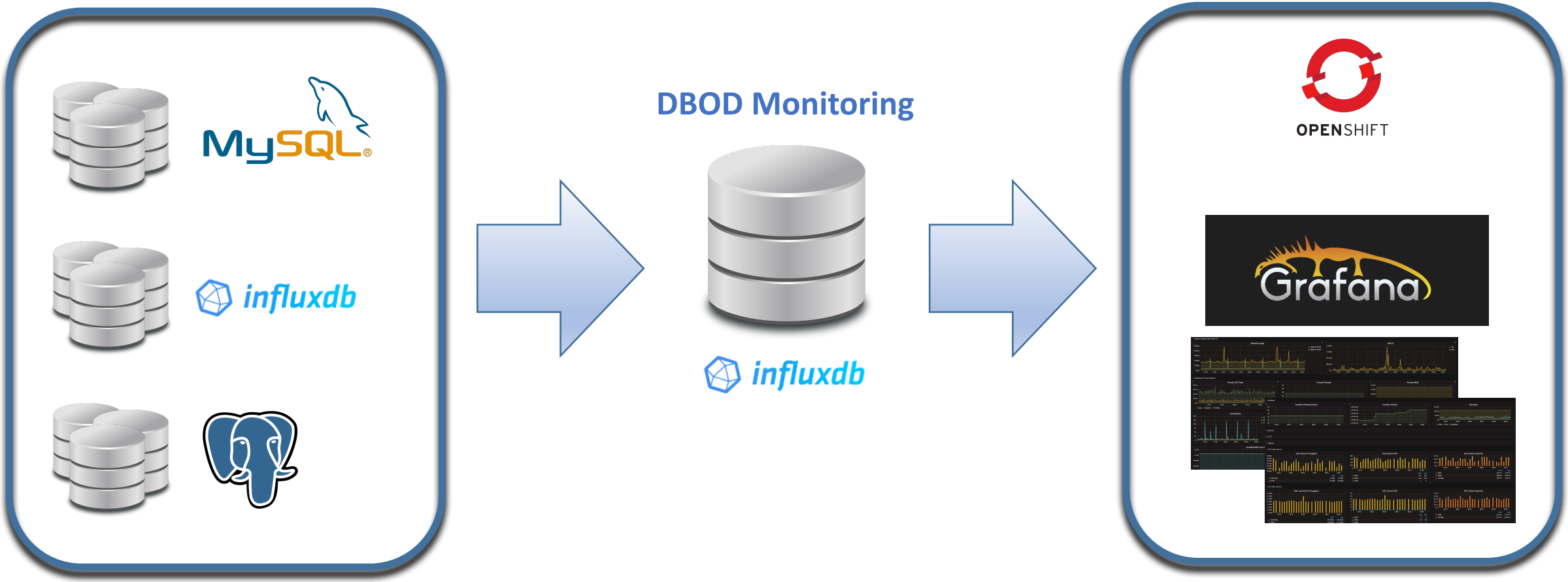
```
CREATE CONTINUOUS QUERY “mycq" ON “mydb"
BEGIN
    SELECT min("temperature")
    INTO "min_temperature"
    FROM "cooling_system"
    GROUP BY time(30m)
END
```

Tips

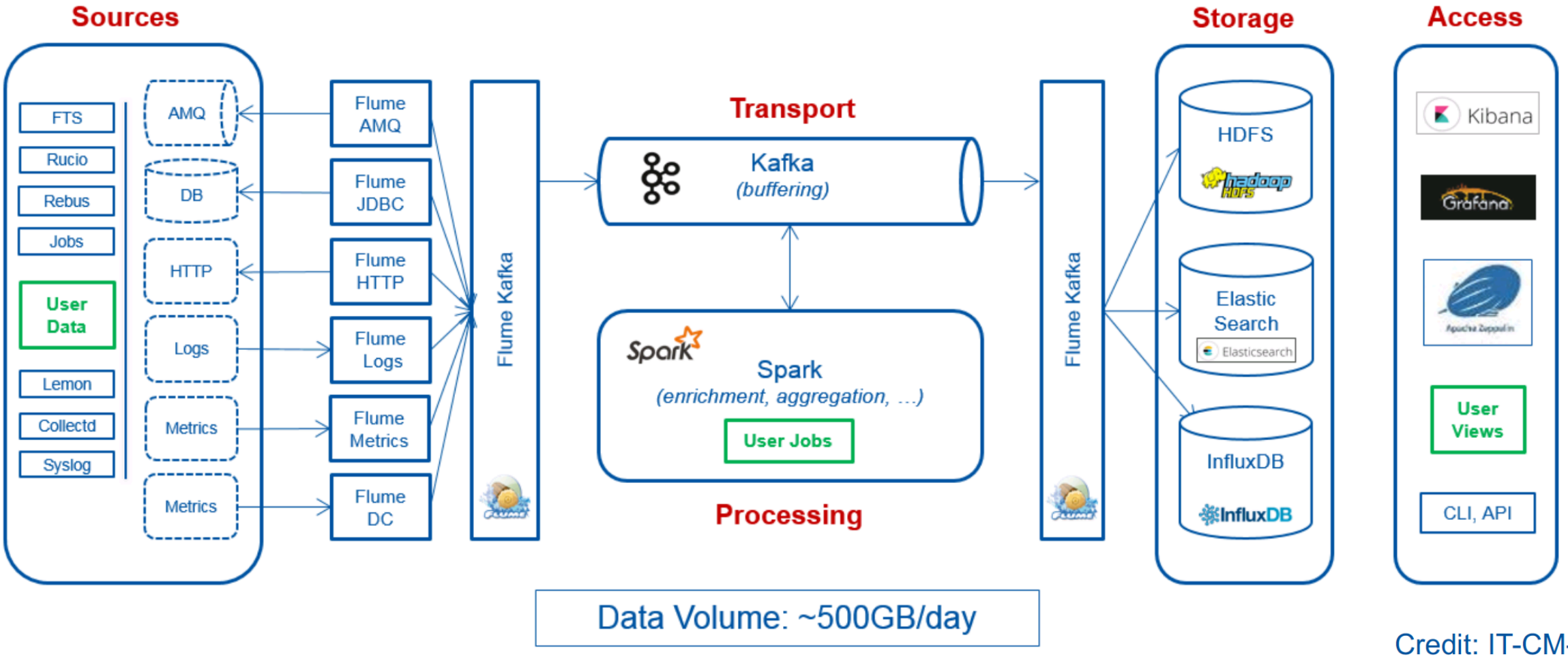
- Series cardinality is key
- Always batch points
- Downsample your data
- Condense similar data, separate un-similar data
- Be precise and choose the time precision you need

- More info: https://www.influxdata.com/resources/influxdb-performance-tuning-tips/

Use Case: DB on Demand



Use Case: IT Monitoring



Credit: IT-CM-MM

Additional notes

- Project is very active and evolving fast
  - Performance improvements
  - Administration capabilities
  - Subqueries
- Most use cases work fine with single instance
- Available in DB on Demand (currently +30 instances)
- More technical details in future ASDF meeting

www.cern.ch