influx days

InfluxDB 101

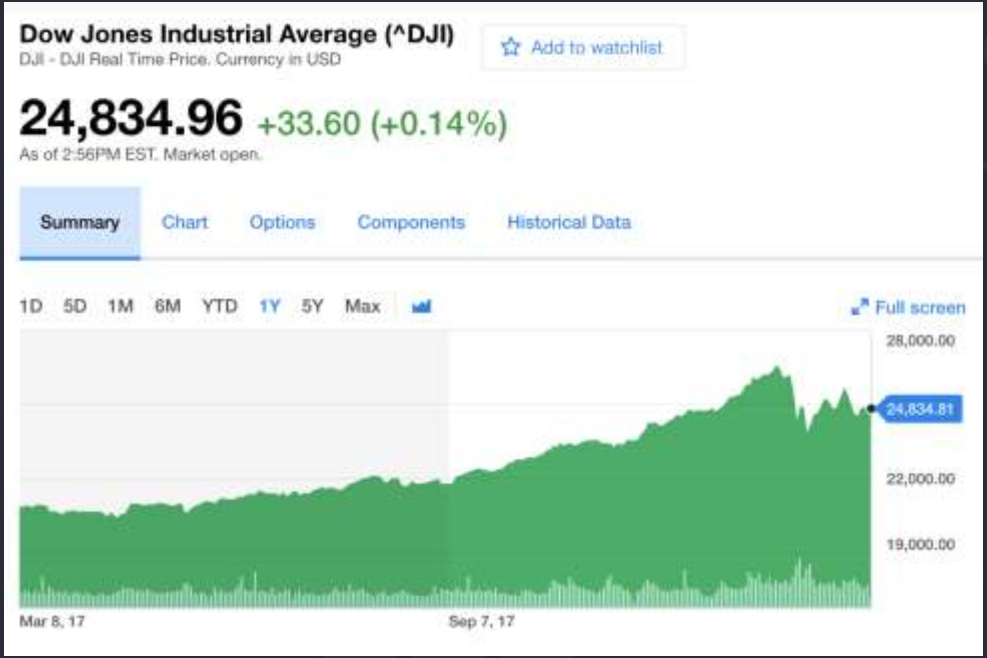
Michael Desa / Software Engineer

Agenda

By the end of this session, users should be able to...

- Define time-series data
- Describe what InfluxDB is and its relationship to InfluxData
- Explain the InfluxDB data model
- Reason about the impact of the schema in an instance

What is time-series data?



Dow Jones Industrial Average (^DJI)

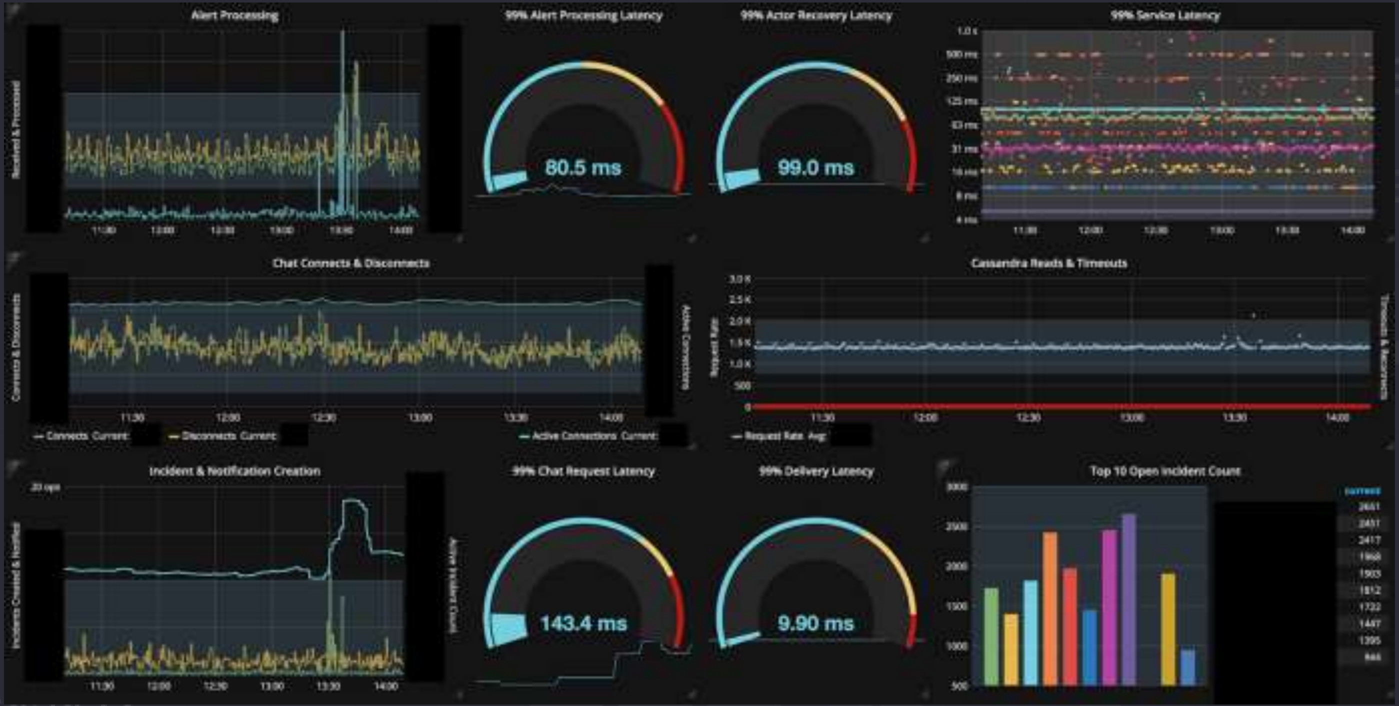
DJI - DJI Real Time Price. Currency in USD

Add to watchlist

24,834.96 +33.60 (+0.14%)

As of 2:56PM EST. Market open.

2015-06-11T20:46:02Z

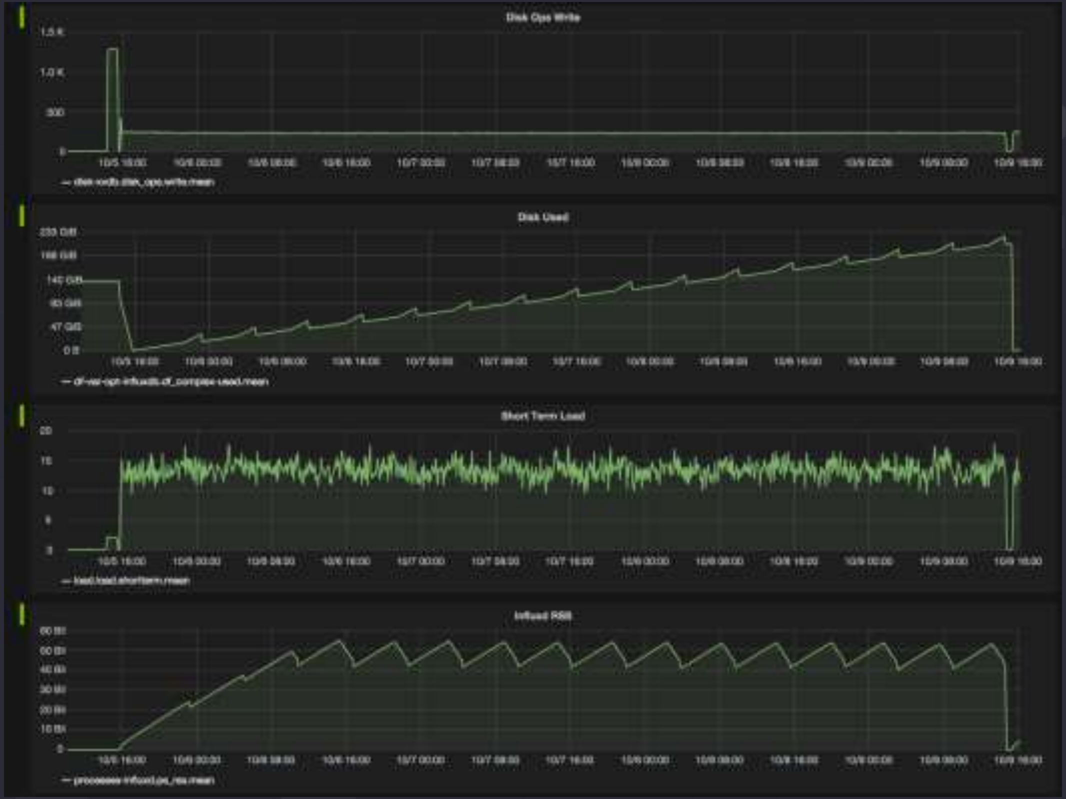


1985-10-25T19:28:07.5

2015-06-11T20:46:02Z

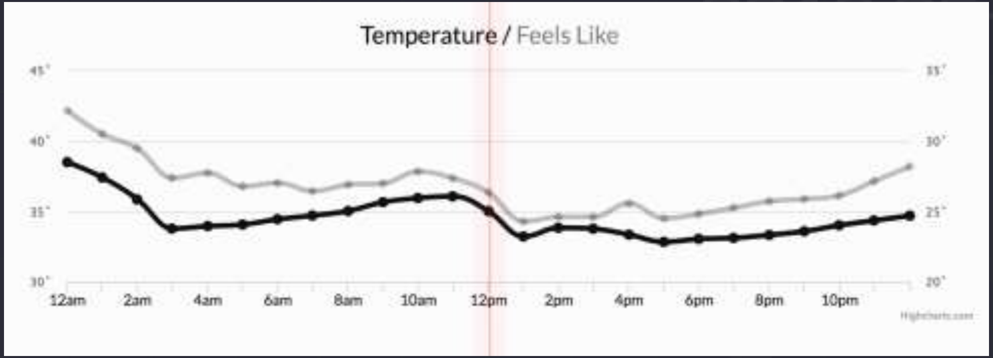
5T19:28:07

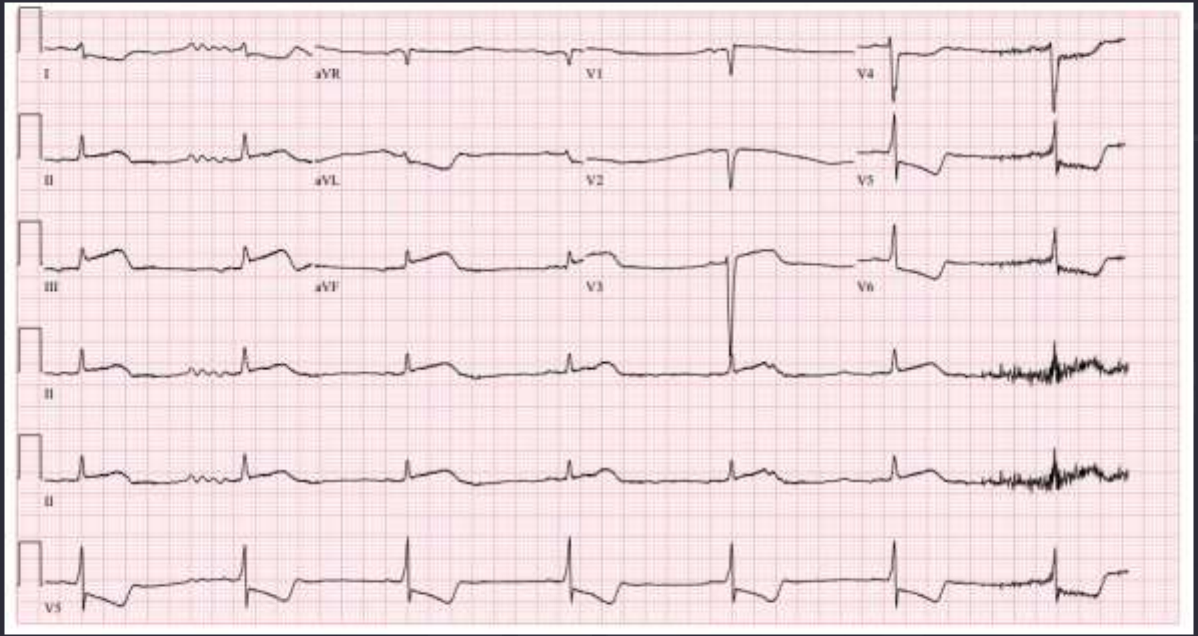
0T23:18:20Z



:28:07.580664347Z

1985-10-25T19:28:07.5

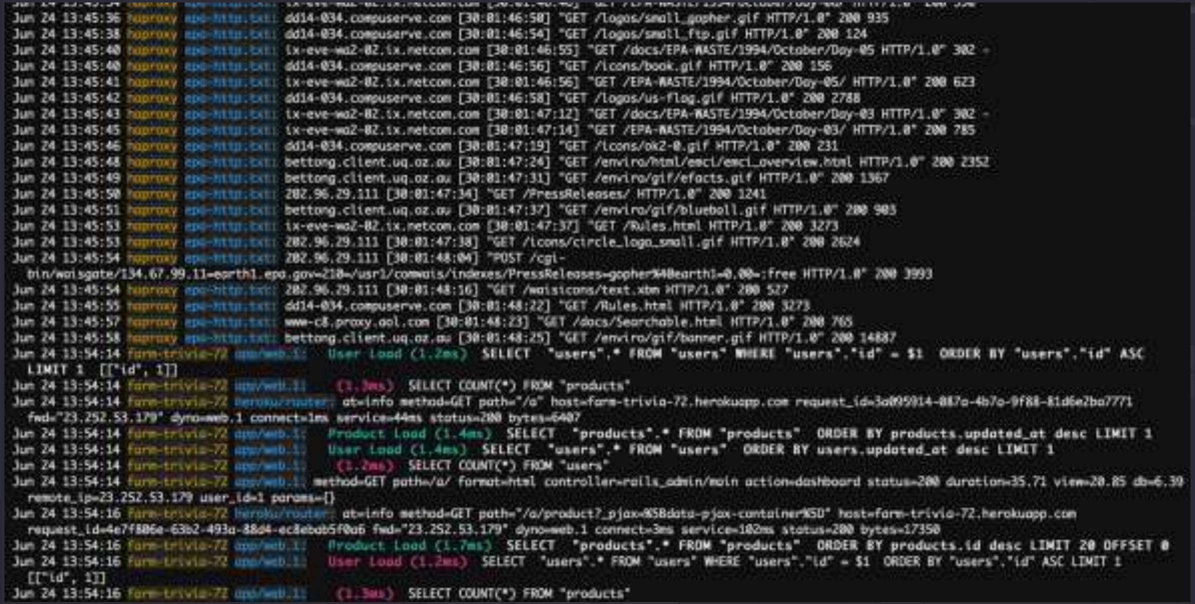

Temperature / Feels Like
45°
40°
35°
30°
35°
30°
25°
20°
12am
2am
4am
6am
8am
10am
12pm
2pm
4pm
6pm
8pm
10pm
Highcharts.com


I
aVR
V1
V4
II
aVL
V2
V5
III
aVF
V3
V6
II
II
V5

2015-06-11T20:46:02Z

T19:28:07

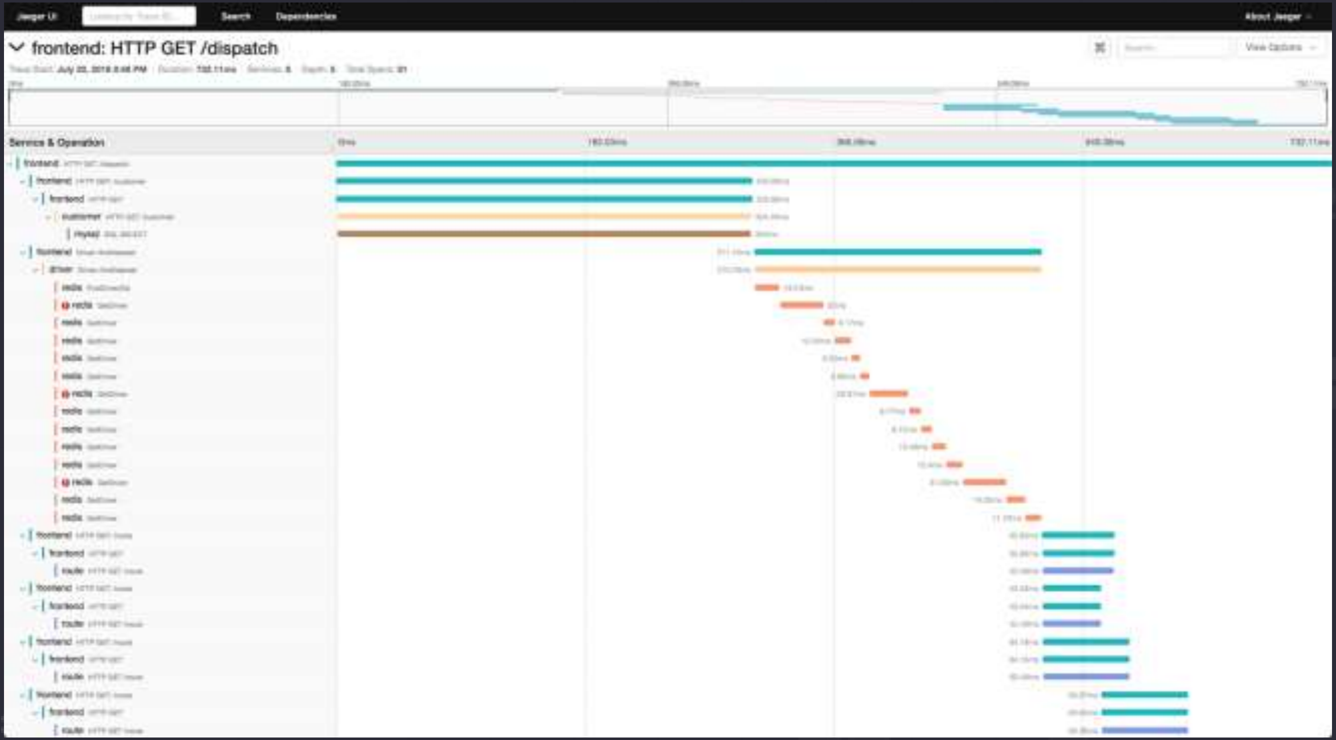
T23:18:20Z



```
Jun 24 13:45:36 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:46:50] "GET /logos/small_gopher.gif HTTP/1.0" 200 935
Jun 24 13:45:38 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:46:54] "GET /logos/small_ftp.gif HTTP/1.0" 200 124
Jun 24 13:45:40 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:46:55] "GET /docs/EPA-WASTE/1994/October/Day-05 HTTP/1.0" 302 -
Jun 24 13:45:40 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:46:56] "GET /icons/book.gif HTTP/1.0" 200 156
Jun 24 13:45:41 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:46:56] "GET /EPA-WASTE/1994/October/Day-05/ HTTP/1.0" 200 623
Jun 24 13:45:42 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:46:58] "GET /logos/us-flag.gif HTTP/1.0" 200 2788
Jun 24 13:45:43 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:47:12] "GET /docs/EPA-WASTE/1994/October/Day-03 HTTP/1.0" 302 -
Jun 24 13:45:45 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:47:14] "GET /EPA-WASTE/1994/October/Day-03/ HTTP/1.0" 200 785
Jun 24 13:45:46 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:47:19] "GET /icons/ok2-0.gif HTTP/1.0" 200 231
Jun 24 13:45:48 haproxy epa-http.txt: bettong.client.uq.oz.au [30:01:47:24] "GET /enviro/html/emci/emci_overview.html HTTP/1.0" 200 2352
Jun 24 13:45:49 haproxy epa-http.txt: bettong.client.uq.oz.au [30:01:47:31] "GET /enviro/gif/efacts.gif HTTP/1.0" 200 1367
Jun 24 13:45:50 haproxy epa-http.txt: 202.96.29.111 [30:01:47:34] "GET /PressReleases/ HTTP/1.0" 200 1241
Jun 24 13:45:51 haproxy epa-http.txt: bettong.client.uq.oz.au [30:01:47:37] "GET /enviro/gif/blueball.gif HTTP/1.0" 200 903
Jun 24 13:45:53 haproxy epa-http.txt: ix-eve-wa2-02.ix.netcom.com [30:01:47:37] "GET /Rules.html HTTP/1.0" 200 3273
Jun 24 13:45:53 haproxy epa-http.txt: 202.96.29.111 [30:01:47:38] "GET /icons/circle_logo_small.gif HTTP/1.0" 200 2624
Jun 24 13:45:54 haproxy epa-http.txt: 202.96.29.111 [30:01:48:04] "POST /cgi-
  bin/waisgate/134.67.99.11=earth1.epa.gov=210=/usr1/comwais/indexes/PressReleases=gopher%40earth1=0.00=:free HTTP/1.0" 200 3993
Jun 24 13:45:54 haproxy epa-http.txt: 202.96.29.111 [30:01:48:16] "GET /waisicons/text.xbm HTTP/1.0" 200 527
Jun 24 13:45:55 haproxy epa-http.txt: dd14-034.compuserve.com [30:01:48:22] "GET /Rules.html HTTP/1.0" 200 3273
Jun 24 13:45:57 haproxy epa-http.txt: www-c8.proxy.aol.com [30:01:48:23] "GET /docs/Searchable.html HTTP/1.0" 200 765
Jun 24 13:45:58 haproxy epa-http.txt: bettong.client.uq.oz.au [30:01:48:25] "GET /enviro/gif/banner.gif HTTP/1.0" 200 14887
Jun 24 13:54:14 form-trivia-72 app/web.1:   User Load (1.2ms)  SELECT  "users".* FROM "users" WHERE "users"."id" = $1  ORDER BY "users"."id" ASC
  LIMIT 1  [["id", 1]]
Jun 24 13:54:14 form-trivia-72 app/web.1:    (1.3ms)  SELECT COUNT(*) FROM "products"
Jun 24 13:54:14 form-trivia-72 heroku/router:  at=info method=GET path="/a" host=form-trivia-72.herokuapp.com request_id=3a095914-087a-4b7a-9f88-81d6e2ba7771
  fwd="23.252.53.179" dyno=web.1 connect=1ms service=44ms status=200 bytes=6407
Jun 24 13:54:14 form-trivia-72 app/web.1:   Product Load (1.4ms)  SELECT  "products".* FROM "products"  ORDER BY products.updated_at desc LIMIT 1
Jun 24 13:54:14 form-trivia-72 app/web.1:   User Load (1.4ms)  SELECT  "users".* FROM "users"  ORDER BY users.updated_at desc LIMIT 1
Jun 24 13:54:14 form-trivia-72 app/web.1:    (1.2ms)  SELECT COUNT(*) FROM "users"
Jun 24 13:54:14 form-trivia-72 app/web.1: method=GET path=/a/ format=html controller=rails_admin/main action=dashboard status=200 duration=35.71 view=20.85 db=6.39
  remote_ip=23.252.53.179 user_id=1 params={}
Jun 24 13:54:16 form-trivia-72 heroku/router:  at=info method=GET path="/a/product?_pjax=%5Bdata-pjax-container%5D" host=form-trivia-72.herokuapp.com
  request_id=4e7f806e-63b2-493a-88d4-ec8ebab5f0a6 fwd="23.252.53.179" dyno=web.1 connect=3ms service=102ms status=200 bytes=17350
Jun 24 13:54:16 form-trivia-72 app/web.1:   Product Load (1.7ms)  SELECT  "products".* FROM "products"  ORDER BY products.id desc LIMIT 20 OFFSET 0
Jun 24 13:54:16 form-trivia-72 app/web.1:   User Load (1.2ms)  SELECT  "users".* FROM "users" WHERE "users"."id" = $1  ORDER BY "users"."id" ASC LIMIT 1
  [["id", 1]]
Jun 24 13:54:16 form-trivia-72 app/web.1:    (1.3ms)  SELECT COUNT(*) FROM "products"
```

:28:07.580664347Z

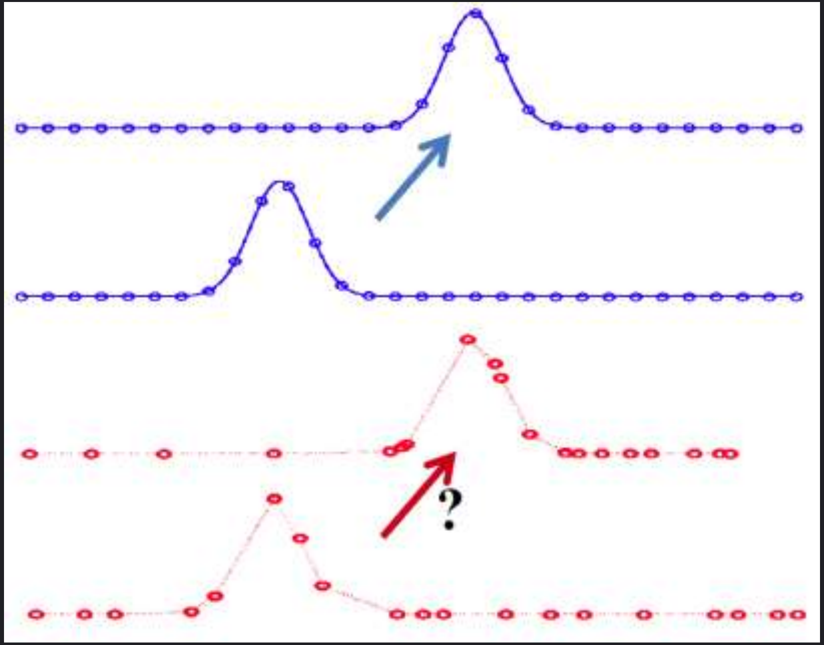
1985-10-25T19:28:07.5



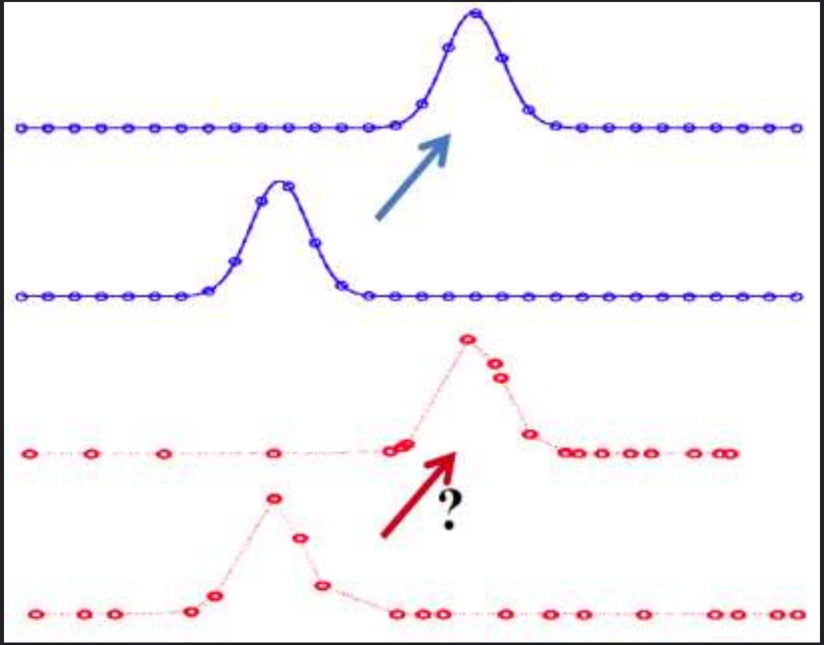
2015-06-11T20:46:02Z

1985-10-25T19:28:07.5

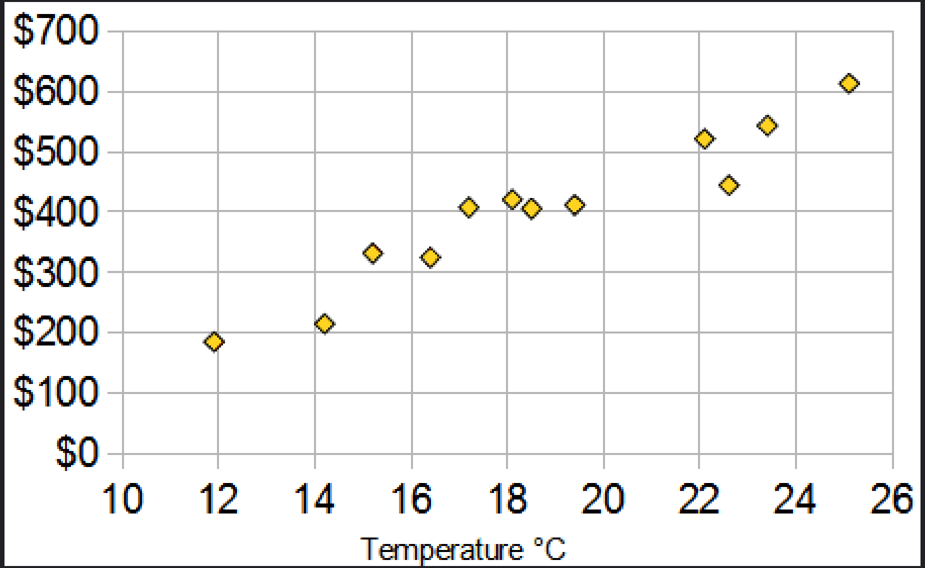
Regular vs Irregular Time-Series



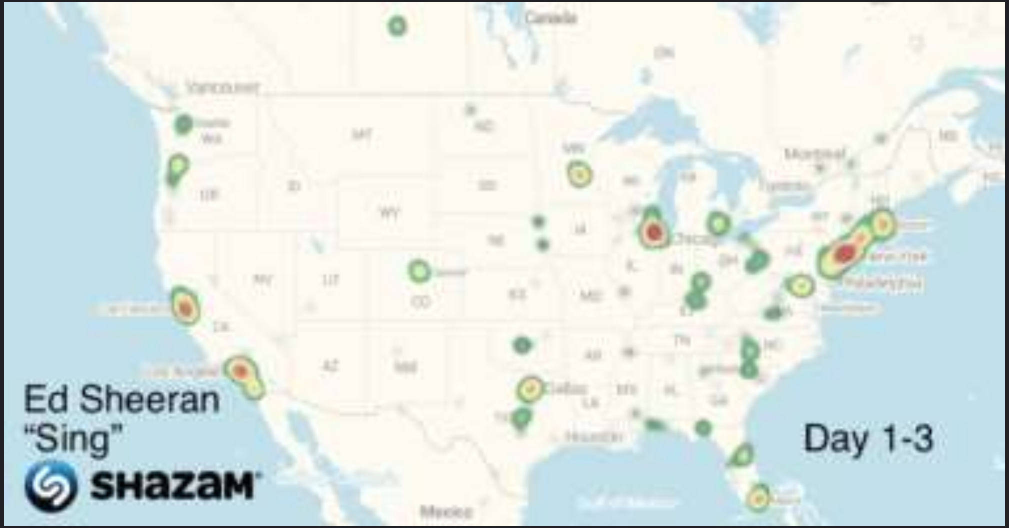
Metrics vs Events



Question: Time-Series?



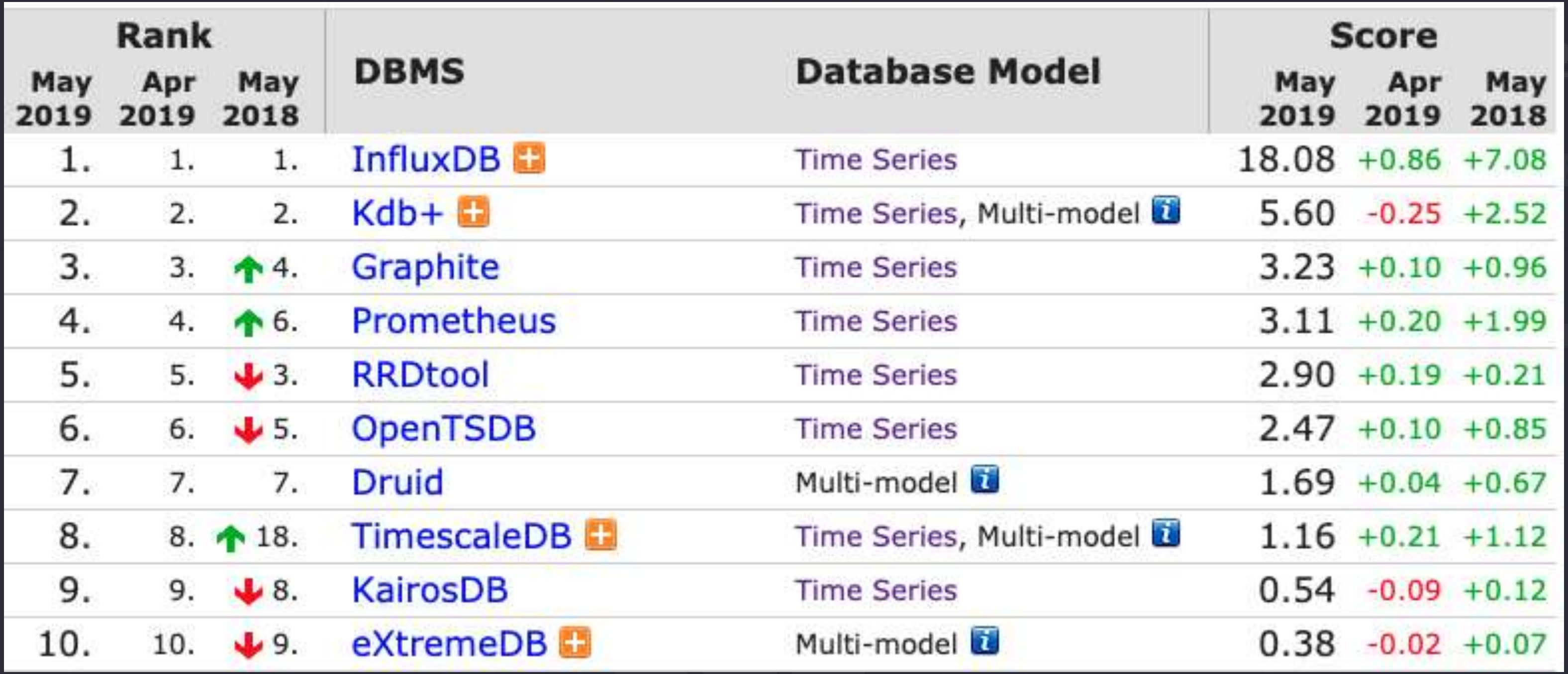
Question: Time-Series?



What is a time-series database (tsdb)?

Time-series Database

- A database where you manage and store time-series data
- Efficiently handles time-series data
- Supports time based queries



Rank			DBMS	Database Model	Score		
May 2019	Apr 2019	May 2018			May 2019	Apr 2019	May 2018
1.	1.	1.	InfluxDB	Time Series	18.08	+0.86	+7.08
2.	2.	2.	Kdb+	Time Series, Multi-model	5.60	-0.25	+2.52
3.	3.	↑ 4.	Graphite	Time Series	3.23	+0.10	+0.96
4.	4.	↑ 6.	Prometheus	Time Series	3.11	+0.20	+1.99
5.	5.	↓ 3.	RRDtool	Time Series	2.90	+0.19	+0.21
6.	6.	↓ 5.	OpenTSDB	Time Series	2.47	+0.10	+0.85
7.	7.	7.	Druid	Multi-model	1.69	+0.04	+0.67
8.	8.	↑ 18.	TimescaleDB	Time Series, Multi-model	1.16	+0.21	+1.12
9.	9.	↓ 8.	KairosDB	Time Series	0.54	-0.09	+0.12
10.	10.	↓ 9.	eXtremeDB	Multi-model	0.38	-0.02	+0.07

Why couldn’t I just use [insert db]?

But, time-series is not just a database problem

Time-series problems

- Visualizing your data
- Alerting you data
- Processing your data
- Taking action based on your data

What is InfluxDB/InfluxData?

InfluxData History

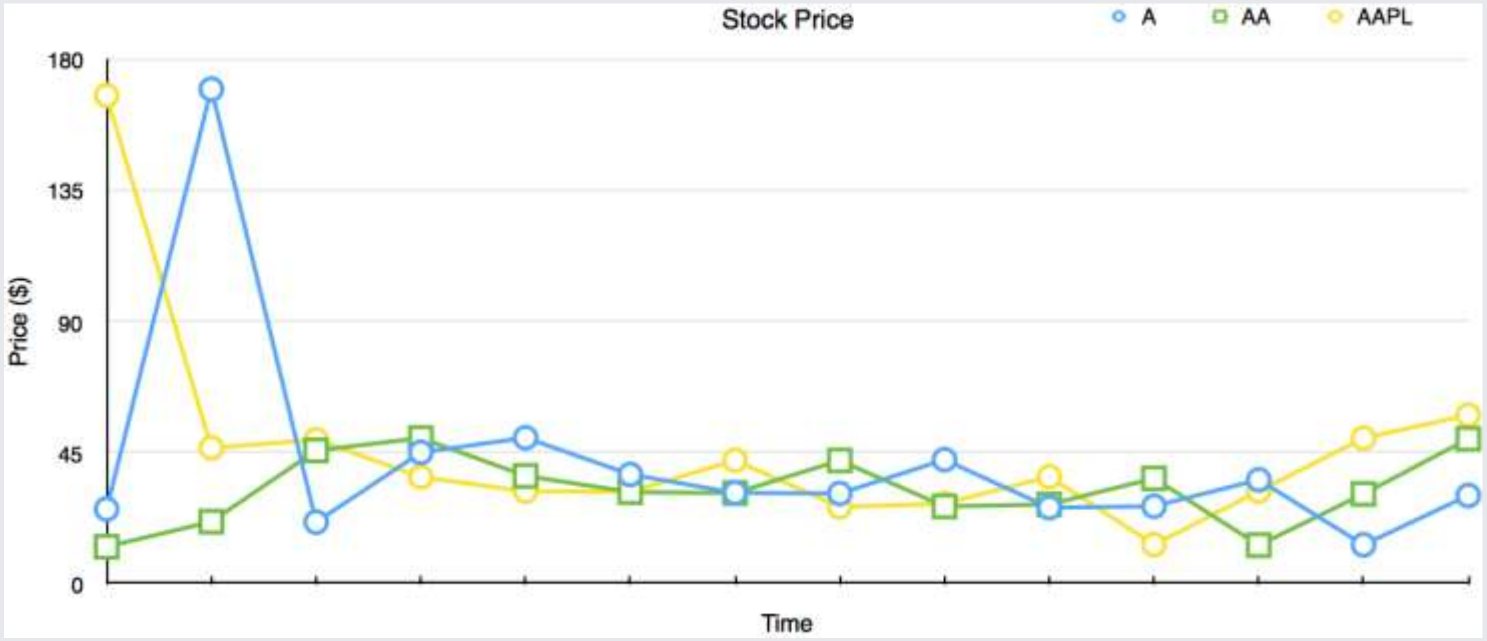
- 2012 - Errplane
- 2014 - InfluxDB is born
- 2015 - Transition to InfluxData
  - A platform for time-series data
- 2018 - 2.0 of InfluxData

Why InfluxData

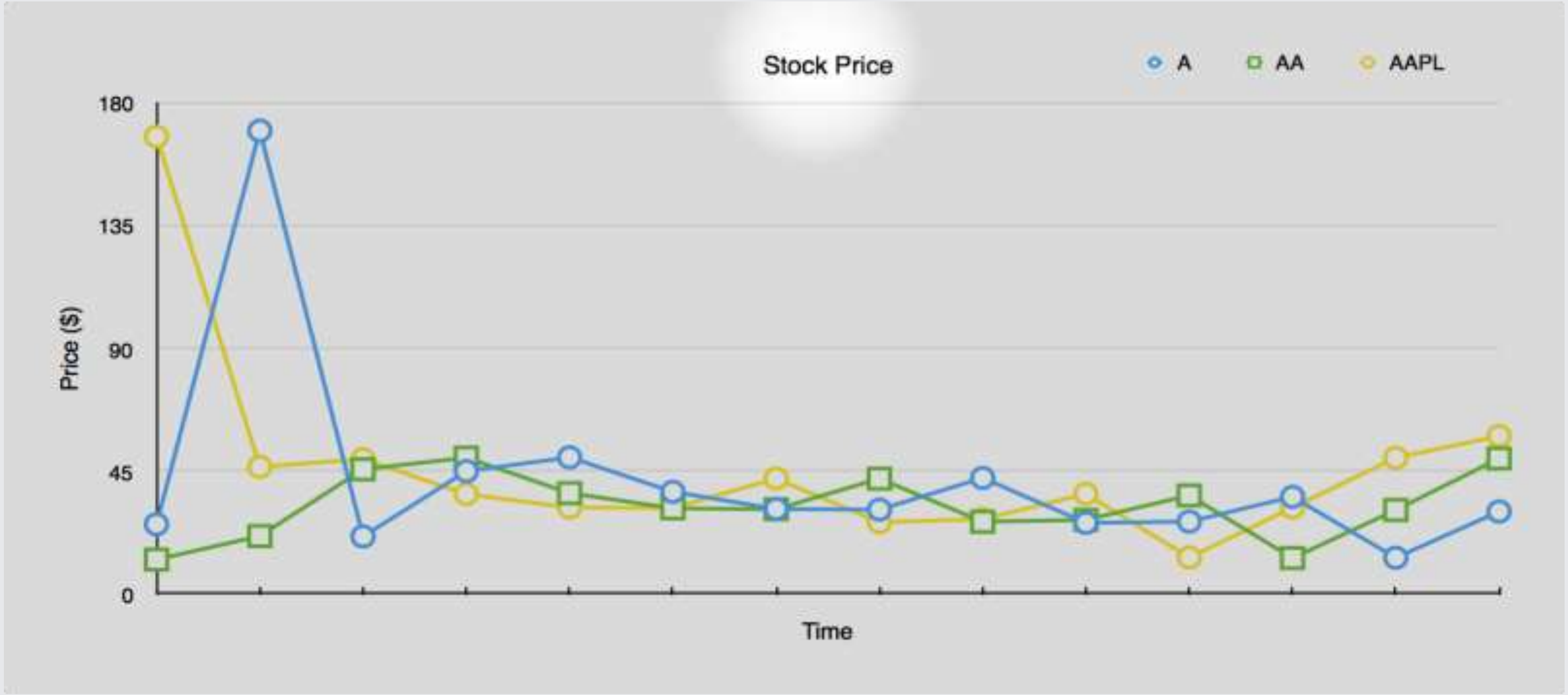
- Easy to get started with
- Aims to solve the entire time-series problem
- Scales well
    - Both horizontally and vertically

InfluxDB Data Model

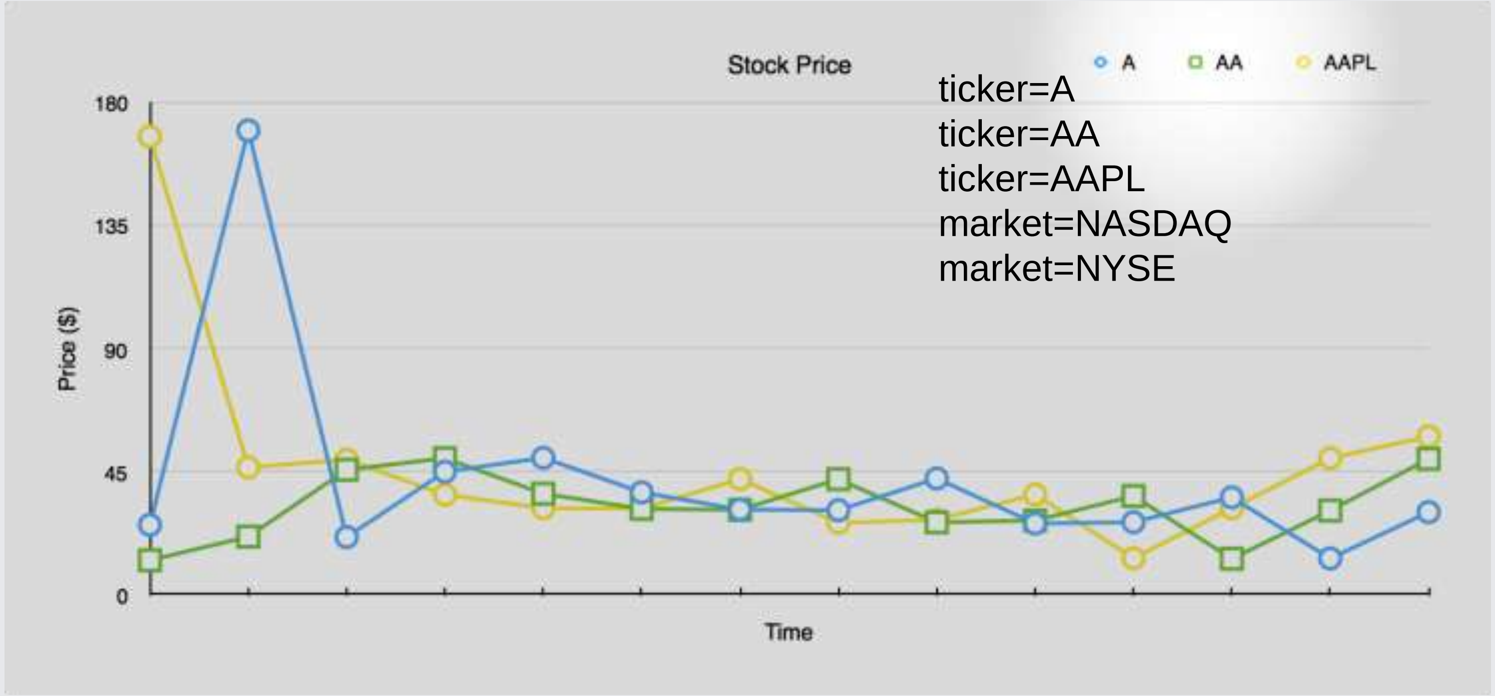
Canonical Time-Series Line Graph



The Label (measurement)

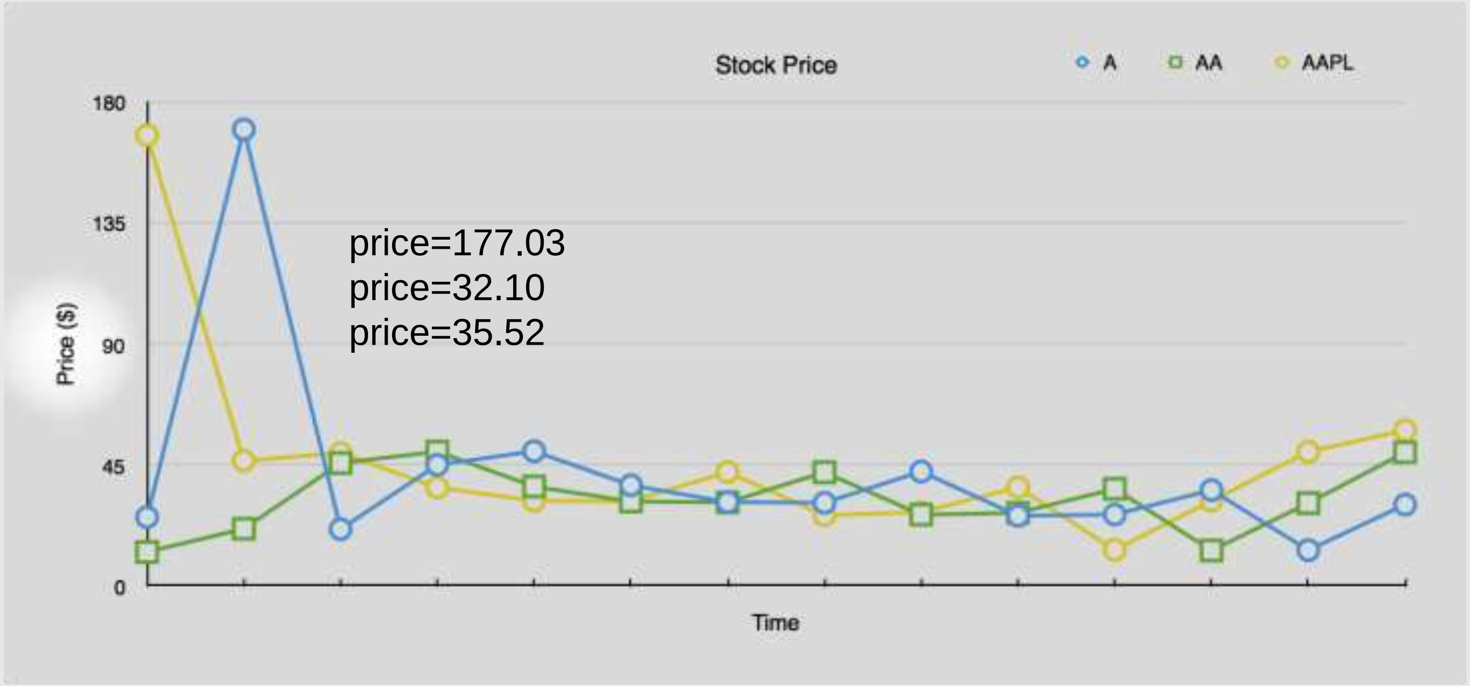


The Legend (tags)



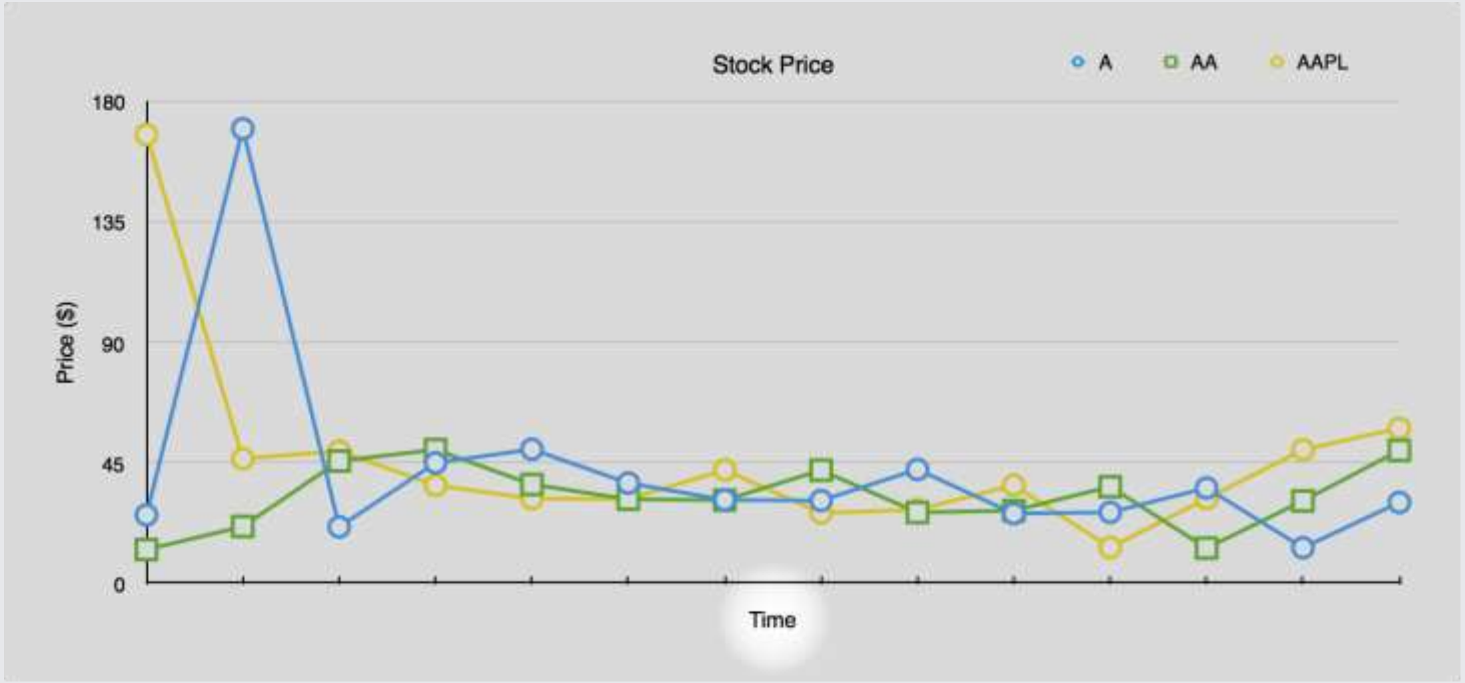
ticker=A
ticker=AA
ticker=AAPL
market=NASDAQ
market=NYSE

Y-Axis Values

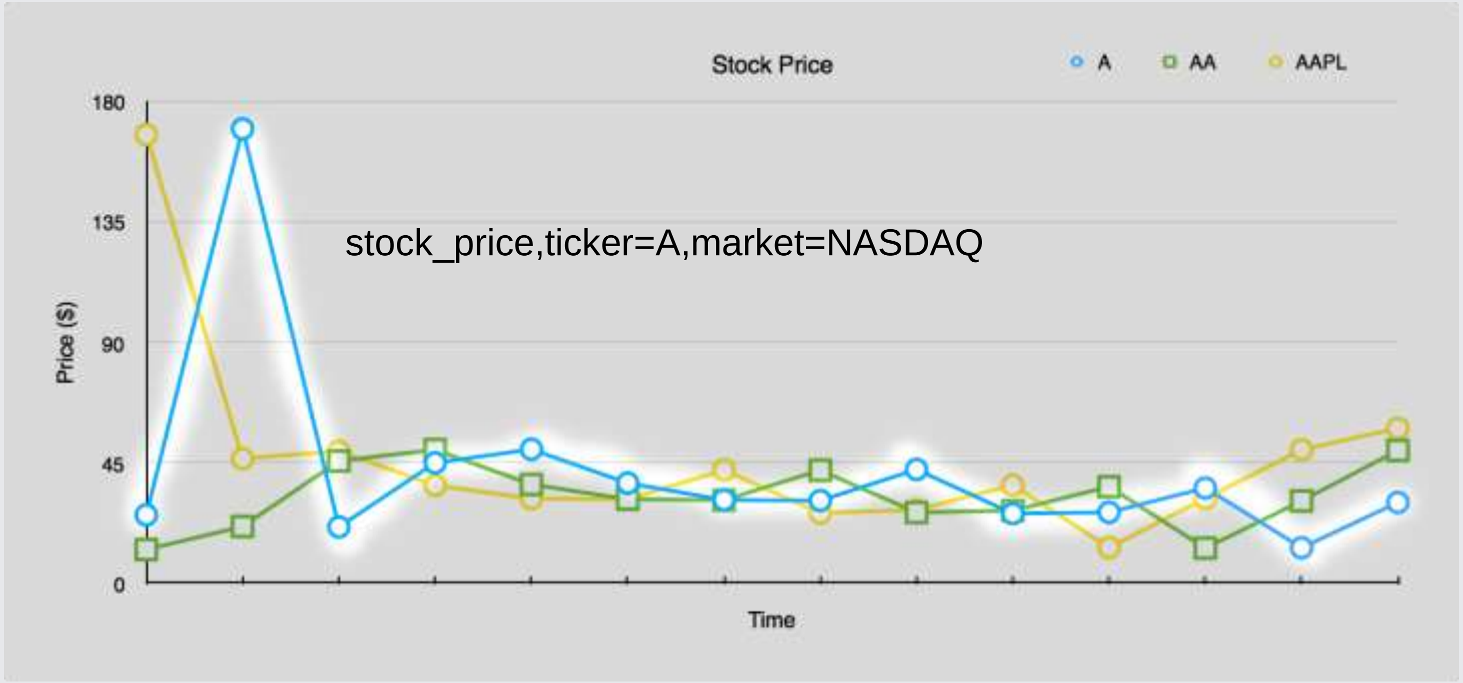


price=177.03
price=32.10
price=35.52

X-Axis Values



Series



stock_price,ticker=A,market=NASDAQ

Data Model

- Measurement
  - High level grouping of data
- Tags
  - Indexed key-value pairs
- Fields
  - Key-value pairs of actual data
- Timestamp
  - Time of the data
- Series
  - A unique combination of measurement+tags

Line Protocol

```
cpu,host=serverA,num=1,region=west idle=1.667,system=2342.2 1492214400000000000
```

↑

Measurement

Line Protocol

```
cpu,host=serverA,num=1,region=west idle=1.667,system=2342.2 1492214400000000000
```

Tag
s

Line Protocol

```
cpu,host=serverA,num=1,region=west idle=1.667,system=2342.2 1492214400000000000
```

↑

Fields

Line Protocol

```
cpu,host=serverA,num=1,region=west idle=1.667,system=2342.2 1492214400000000000
```

$\uparrow$

timestamp

Querying Data

InfluxData Languages

- InfluxQL
  - SQL-like query language

- TICKscript
  - Time-series data processing language

- Flux
  - Next generation functional data scripting language

InfluxQL

```
> SELECT index, id FROM h2o_quality WHERE time > now() - 1w GROUP BY location
name: h2o_quality
tags: location = coyote_creek
time                  index  id
----                  -----  ---
2015-08-18T00:00:00Z  41      1
2015-08-18T00:00:00Z  41      1

name: h2o_quality
tags: location = santa_monica
time                  index  id
----                  -----  ---
2015-08-18T00:00:00Z  99     2
2015-08-18T00:06:00Z  56     2
```

TICKscript

```
var measurement = 'requests'

var data = stream
    |from()
        .measurement(measurement)
    |where(lambda: "is_up" == TRUE)
    |where(lambda: "my_field" > 10)
    |window()
         .period(5m)
         .every(5m)

// Count number of points in window
data
     |count('value')
       .as('the_count')

// Compute mean of data window
data
     |mean('value')
      .as('the_average')
```

Flux

```
// get all data from the telegraf db
from(bucket:"telegraf/autogen")
  // filter that by the last hour
  |> range(start:-1h)
  // filter further by series with a specific measurement and field
  |> filter(fn: (r) => r._measurement == "cpu" and r._field == "usage_system")
```

Why Flux?

- Composabile
  - Users should be able to take pieces of different scripts and combine them into a single one to solve their own problem
- Extensible
  - Adding new functions and capabilities to flux should be easy
- Shareable
  - Users should be able to create libraries and packages to solve specific problems
- Flexible
  - Users should be able to use the language for arbitrary data processing

Schema Design

What not to do

Don’t Encode Data into Measurement/Tags

Bad:

```
cpu.server-5.us-west value=2 1444234982000000000

cpu.server-6.us-west value=4 1444234982000000000

mem-free.server-6.us-west value=2500 1444234982000000000
```

Good:

```
cpu,host=server-5,region=us-west value=2 1444234982000000000

cpu,host=server-6,region=us-west value=4 1444234982000000000

mem-free,host=server-6,region=us-west value=2500 1444234982000000
```

Don’t Encode Data into Measurement/Tags

Bad:

```
cpu,server=localhost.us-west value=2 1444234982000000000

cpu,server=localhost.us-east value=3 1444234982000000000
```

Good:

```
cpu,host=localhost,region=us-west value=2 1444234982000000000

cpu,host=localhost,region=us-east value=3 1444234982000000000
```

Don’t Use Tags with Very High Variability

Bad:

```
response_time,session_id=33254331,request_id=3424347 value=340 14442349820000
```

Good-ish:

```
response_time session_id=33254331,request_id=3424347,value=340 14442349820000
```

Don’t Use Too Few Tags

Bad:

```
cpu,region=us-west host="server1",value=0.5 1444234986000
cpu,region=us-west host="server2",value=4 1444234982000
cpu,region=us-west host="server2",value=1 1444234982000
```

Good-ish:

```
cpu,region=us-west,host=server1 value=0.5 1444234986000
cpu,region=us-west,host=server2 value=4 1444234982000
cpu,region=us-west,host=server2 value=1 1444234982000
```

What should I do then?

Designing a Schema

- What dashboards do I need?
- What alerts do I need?
- What kind of reports do I want to generate?
- What type of information do I need readily available when there’s an incident?

Schema Example

- I operate a SaaS application
- There are ~1000 different services
- I want to know the request and error rates for each service
- I want to trigger an alert if the error rate for each service
- I want to see the services with the highest average request duration

Data Available

`app`	Service name, e.g. user_service, auth_service…
`container_id`	Container ID of the container running the service
`path`	HTTP request path
`method`	HTTP method, e.g. GET, POST, DELETE…
`src`	Hostname of client making request
`dest`	Hostname of server being contacted
`status`	HTTP status code associated with the request
`request_id`	Unique request identifier
`duration`	Duration of request
`bytes_tx`	Number of bytes transmitted
`bytes_rx`	Number of bytes received

Question

Why would it be a bad idea to make `container_id` or `request_id` a tag?

Answer

Why would it be a bad idea to make `container_id` or `request_id` a tag?

`request_id` and `container_id` both have a high cardinality and could result in an large number of series, which impacts memory utilization.

`request_id` is substantially worse than `container_id`. In the next few releases we hope to allow for indexing on `container_id`.

Question

How should we organize our data?

Data Available

`app`	Service name, e.g. user_service, auth_service…
`container_id`	Container ID of the container running the service
`path`	HTTP request path
`method`	HTTP method, e.g. GET, POST, DELETE…
`src`	Hostname of client making request
`dest`	Hostname of server being contacted
`status`	HTTP status code associated with the request
`request_id`	Unique request identifier
`duration`	Duration of request
`bytes_tx`	Number of bytes transmitted
`bytes_rx`	Number of bytes received

Schema

measurement:

`latency`

tags:

`app container_id  path  method  src  dst   status`

fields:

`request_id  duration  bytes_tx   bytes_rx`

Request/Error Rate per Service

Top 10 average request duration

```
> SELECT top(avg_duration, app, 10) FROM (
       SELECT mean(duration) AS avg_dur
              FROM latency
              WHERE time > now() - 1h
              GROUP BY time(1m), *
  )
```

Request Rate Per Service

```
> SELECT count(duration)
  FROM latency
  WHERE time > now() - 10m
  GROUP BY app, time(1s) fill(none)
```

Error Rate Per Service

```
> SELECT count(duration)
  FROM latency
  WHERE time > now() - 10m AND status != ‘200’
  GROUP BY app, time(1s) fill(none)
```

Thank You!

influx|days

influx|days