

Why InfluxDB is building Flux

Paul Dix
@pauldix
paul@influxdata.com






- Query planner
- Query optimizer
- Turing complete language, VM, and query engine
- Multi-language support in Engine
- Multi-data source support
- InfluxDB, CLI, REPL, Go library


```
// 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")
```

Comments



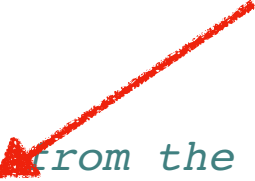
```
// get all data from the telegraf db
from(bucket:"telegraf/autogen")
  // filter that by the last hour
  |> range(start:-1h)
  // filter further by series with a specific measurement and field
  |> filter(fn: (r) => r._measurement == "cpu" and r._field == "usage_system")
```

Named Arguments




```
// 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")
```

String Literals



```
// 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")
```

Buckets, not DBs



```
// get a 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")
```



```
// 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")
```



Duration Literal

```
// get all data from the telegraf db
from(bucket:"telegraf/autogen")
  // filter that by the last hour
  |> range(start:2018-11-07T00:00:00Z)
  // filter further by series with a specific measurement and field
  |> filter(fn: (r) => r._measurement == "cpu" and r._field == "usage_system")
```

← **Time Literal**

```
// 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")
```



Pipe forward operator

```
// 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")
```



Anonymous Function

```
// 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" or r._measurement == "cpu")
                      and r.host == "serverA")
```



Predicate Function

The journey...

The design of IFQL, the New Influx Functional Query Language

A new query language for InfluxDB

**A new scripting & query
language for InfluxDB**

A new **scripting & query**
language ~~for InfluxDB~~

IFQL -> Flux




Search or jump to...



[Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)

 [influxdata](#) / [flux](#)

 Unwatch ▾

32

★ Star

107

 Fork

9

Code

Issues 313

Pull requests 12

Insights

Settings


Flux is a lightweight scripting language for querying databases (like InfluxDB) and working with data. It's part of InfluxDB 1.7 and 2.0, but can be run independently of those. <https://influxdata.com>


Edit

[Manage topics](#)

 865 commits


 48 branches

 22 releases

 23 contributors

 MIT

[Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)

 Unwatch ▾

32


 Star

107

 Fork

9

313


 Pull requests **12**

 Insights

 Settings

Scripting language for querying databases (like InfluxDB) and working with data. It's part of InfluxDB 1.7 and
independently of those. <https://influxdata.com>

Edit

 **48** branches

 **22** releases

 **23** contributors

 MIT

Why We're Building Flux, a New Data Scripting and Query Language

BY PAUL DIX | JUL 19, 2018 | COMMUNITY, DEVELOPER, FLUX, INFLUXDB | 0 COMMENTS

```
from(db:"telegraf")  
  |> range(start:-1h)  
  |> filter(fn: (r) => r._measurement == "foo")  
  |> exponentialMovingAverage(size:-10s)
```



“I don’t want to live in a world where the best language humans could think of for working with data was invented in the 70’s”

–Paul Dix





Hacker News

[new](#) | [threads](#) | [past](#) | [comments](#) | [ask](#) | [show](#) | [jobs](#) | [yc](#) | [submit](#)

pauldix

HATERS



GONNA HATE

“So you're saying you're combining the expressiveness of SQL with the readability of, what, perl?”

–tylerl on HN

“Perl is actually readable compared to that.”

–petre in response to tylerl on HN

“SQL has a very solid ground in research - a lot of - in relational algebra. If you try to make a query language that is a dsl for anything without a really different data model underneath, you will accomplish nothing great.”

-rs86 on HN

“The ultimate fantasy of every programmer is to a) invent a new language and b) force other people to use it. It’s OK, we all get it, it’s fine. But let’s be honest about our motivations...”

–gaius on HN

“Holy shit. The language name alone is a really, really stupid idea.”

–fake-name on HN

HATERS GONNA HATE



“It was a bad idea to focus on Flux vs SQL.”

–lixtra on HN

Paul, don't be a jerk

!zeroSum()

“I don’t want to live in a world where the best language humans could think of for working with data was invented in the 70’s”

–Paul Dix

“I don’t want to live in a world where ~~the best language humans could think of for working with data was invented in the 70’s~~”

–Paul Dix

“I don’t want to live in a world where...”

–Paul Dix

“I don’t want to live in a world where **we can’t try something new**”

–Paul Dix

4GL

Domain Specific Languages

JavaScript?

GUI

Many Data Sources

Optimize for each

Cross compilation

This branch is 7 commits ahead of influxdata:master.

Pull request Compare

juliusv Rename database setup dir and script Latest commit ce4612d 18 hours ago

..

db-setup	Rename database setup dir and script	18 hours ago
README.md	Rename database setup dir and script	18 hours ago
main.go	Better comments, logging, and edge case handling	a day ago
query_helpers.go	Better comments, logging, and edge case handling	a day ago
transpile.go	Better comments, logging, and edge case handling	a day ago

README.md

PromQL -> Flux Transpiler Proof-of-Concept

This takes a PromQL query, transpiles it to Flux, and then runs it against both Prometheus and InfluxDB.

AST = API

Distributed Engine

A Native Go Library for Apache Arrow ∞

Published 22 Mar 2018

By [The Apache Arrow PMC](#) ()

Since launching in early 2016, Apache Arrow has been growing fast. We have made nine major releases through the effort of our contributors. The project's scope has also expanded. We began by focusing on the development of the standardized in-memory columnar format, which now serves as a pillar of the project. Since then, we have been growing into a more general cross-language platform through new additions to the project like the [Plasma shared memory object store](#). A primary goal of the project is to enable fast process and move data fast.

Tables Everywhere

```
from(bucket: "foo")  
  |> range(start: -10m)  
  |> filter(fn: (r) => r._measurement == "cpu")  
  |> group(columns: [ "_measurement" ] )  
  |> sort(columns: [ "_value" ] )
```



Sorting by value!

Group by anything

**Measurements, tags, fields
don't matter**

Beyond Queries

```
option task = {
  name: "email alert digest",
  cron: "0 5 * * 0"
}
import "smtp"

body = ""

from(bucket: "alerts")
  |> range(start: -24h)
  |> filter(fn: (r) => (r.level == "warn" or r.level == "critical") and r._field == "message")
  |> group(columns: ["alert"])
  |> count()
  |> group()
  |> map(fn: (r) => body = body + "Alert {r.alert} triggered {r._value} times\n")

smtp.to(
  config: loadSecret(name: "smtp_digest"),
  to: "alerts@influxdata.com",
  title: "Alert digest for {now()}",
  body: message)
```

```
option task = {  
  name: "email alert digest",  
  cron: "0 5 * * 0"  
}
```



tasks

```
import "smtp"
```

```
body = ""
```

```
from(bucket: "alerts")  
  |> range(start: -24h)  
  |> filter(fn: (r) => (r.level == "warn" or r.level == "critical") and r._field == "message")  
  |> group(columns: ["alert"])  
  |> count()  
  |> group()  
  |> map(fn: (r) => body = body + "Alert {r.alert} triggered {r._value} times\n")
```

```
smtp.to(  
  config: loadSecret(name: "smtp_digest"),  
  to: "alerts@influxdata.com",  
  title: "Alert digest for {now()}",  
  body: message)
```

```
option task = {
  name: "email alert digest",
  cron: "0 5 * * 0"
}
import "smtp"

body = ""

from(bucket: "alerts")
  |> range(start: -24h)
  |> filter(fn: (r) => (r.level == "warn" or r.level == "critical") and r._field == "message")
  |> group(columns: ["alert"])
  |> count()
  |> group()
  |> map(fn: (r) => body = body + "Alert {r.alert} triggered {r._value} times\n")

smtp.to(
  config: loadSecret(name: "smtp_digest"),
  to: "alerts@influxdata.com",
  title: "Alert digest for {now()}",
  body: message)
```

 cron scheduling

```
option task = {
  name: "email alert digest",
  cron: "0 5 * * 0"
}
import "smtp"
body = ""

from(bucket: "alerts")
  |> range(start: -24h)
  |> filter(fn: (r) => (r.level == "warn" or r.level == "critical") and r._field == "message")
  |> group(columns: ["alert"])
  |> count()
  |> group()
  |> map(fn: (r) => body = body + "Alert {r.alert} triggered {r._value} times\n")

smtp.to(
  config: loadSecret(name: "smtp_digest"),
  to: "alerts@influxdata.com",
  title: "Alert digest for {now()}",
  body: message)
```

 packages & imports

```

option task = {
  name: "email alert digest",
  cron: "0 5 * * 0"
}
import "smtp"

body = ""

from(bucket: "alerts")
  |> range(start: -24h)
  |> filter(fn: (r) => (r.level == "warn" or r.level == "critical") and r._field == "message")
  |> group(columns: ["alert"])
  |> count()
  |> group()
  |> map(r => body = body + "Alert {r.alert} triggered {r._value} times\n")

smtp.to(
  config: loadSecret(name: "smtp_digest"),
  to: "alerts@influxdata.com",
  title: "Alert digest for {now()}",
  body: message)


```

```
option task = {
  name: "email alert digest",
  cron: "0 5 * * 0"
}
import "smtp"

body = ""

from(bucket: "alerts")
  |> range(start: -24h)
  |> filter(fn: (r) => (r.level == "warn" or r.level == "critical") and r._field == "message")
  |> group(columns: ["alert"])
  |> count()
  |> group()
  |> map(fn: (r) => body = body + "Alert {r.alert} triggered {r._value} times\n")

smtp.to(
  config: loadSecret(name: "smtp_digest"),
  to: "alerts@influxdata.com",
  title: "Alert digest for {now()}",
  body: message)
```



String interpolation

```
option task = {
  name: "email alert digest",
  cron: "0 5 * * 0"
}
import "smtp"

body = ""

from(bucket: "alerts")
  |> range(start: -24h)
  |> filter(fn: (r) => (r.level == "warn" or r.level == "critical") and r._field == "message")
  |> group(columns: ["alert"])
  |> count()
  |> group()
  |> map(fn: (r) => body = body + "Alert {r.alert} triggered {r._value} times\n")

smtp.to(
  config: loadSecret(name: "smtp_digest"),
  to: "alerts@influxdata.com",
  title: "Alert digest for {now()}",
  body: message)
```

 **Ship data elsewhere**


```
option task = {
  name: "email alert digest",
  cron: "0 5 * * 0"
}
import "smtp"

body = ""

from(bucket: "alerts")
  |> range(start: -24h)
  |> filter(fn: (r) => (r.level == "warn" or r.level == "critical") and r._field == "message")
  |> group(columns: ["alert"])
  |> count()
  |> group()
  |> map(fn: (r) => body = body + "Alert {r.alert} triggered {r._value} times\n")


smtp.to(
  config: loadSecret(name: "smtp_digest"),
  to: "alerts@influxdata.com",
  title: "Alert digest for {now()}",
  body: message)
```

**Store secrets in a
store like Vault**

 influxdata / flux

 Code

 Issues 315

 Pull requests 13

 Insights

 Settings









Branch: master ▾

flux / **stdlib** / **testing** /



aanthony1243 fix(stdlib/testing): add locks to make diff threadsafe (#996)

..

 testdata	chore(parser): move strconv from ast to parser
 assert_empty.go	feat(testing): add assertEmpty method and use it with te
 assert_empty_test.go	feat(testing): add assertEmpty method and use it with te
 assert_equals.go	feat(testing): add assertEmpty method and use it with te
 assert_equals_test.go	feat(stdlib): add testing.diff function (#917)
 diff.go	fix(stdlib/testing): add locks to make diff threadsafe (#99
 diff_test.go	feat(testing): add assertEmpty method and use it with te
 flux_gen.go	refactor(testing): use testing.run and testing.inspect as :

Where we are...

1. Make it powerful

2. Make it easy

3. Make it fast

1. Make it powerful

2. Make it easy

3. Make it fast

Flux in InfluxDB 1.7



InfluxDB 2.0



- Multi data source
- Multi data sink
- Turing complete
- Liberally licensed
- Not tied to InfluxDB

Bigger than InfluxDB!



Thank you

Paul Dix

@pauldix

paul@influxdata.com