Grafana & Flux

New Flux support in Grafana

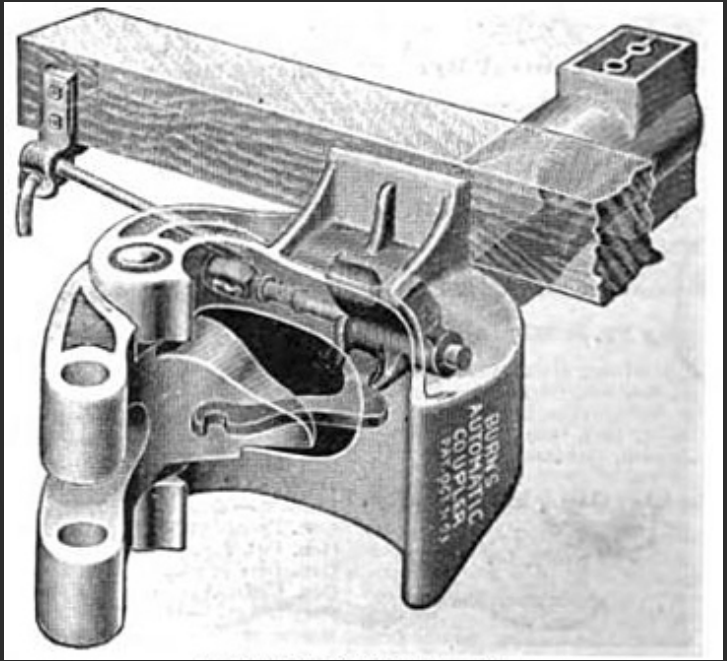
Jacob Lisi
@JacobLisi

TL;DR

- Flux is powerful
- You can start playing with Flux in Grafana today
  - Flux support in Grafana is available via a new datasource plugin
- For now, no automated way to migrate dashboards
  - Transpiler for Influx queries is being worked on
  - You can migrate your dashboards and panels manually

Flux Design Goals

- Decouple language from the execution engine
  - Execution engine takes a DAG of transformations
  - Transpilers: Flux, Influx Query, PromQL
- Decouple database from query computation
  - Iterate faster on query engine
  - Keep database untouched
  - Independently scalable
- Add more functions
  - Chainable: transformation from input table to output table



Grafana Labs

Flux recap

```
from(db: “telegraf”)
  |> filter(fn: (r) => r[“host”] == “myServer”)
  |> range(start: -1h)
```

Flux recap: Select one value from a table

```
from(db: “telegraf”)
  |> filter(fn: (r) => r[“host”] == “myServer”)
  |> range(start: -1h)
  |> max() // Selector
```

Flux primer: Window-chunk and aggregate

```
from(db: “telegraf”)
  |> filter(fn: (r) => r[“_measurement”] == “cpu”)
  |> range(start: -1h)
  |> window(every: 10m)
  |> mean() // Aggregator
  |> filter(fn: (r) => r._value > 1) // Having
```

Grafana Labs

Flux query planning

```
from(db: “telegraf”)
  |> filter(fn: (r) => r[“host”] == “myServer”)
  |> range(start: -1h)
  |> max()

from(db: “telegraf”)
  |> range(start: -1h)
  |> filter(fn: (r) => r[“host”] == “myServer”)
  |> max()
```

- **Same plan DAG**

Flux query planning gotchas

```
from(db: “telegraf”)
  |> filter(fn: (r) => r[“host”] == “myServer”)
  |> range(start: -1h)
  |> max()

from(db: “telegraf”)
  |> filter(fn: (r) => r[“host”] == “myServer”)
  |> max() // Selector function returns 1 record
  |> range(start: -1h)


from(db: “telegraf”)
  |> range(start: -1h)
  |> filter(fn: (r) => r[“_value”] > 1) // Full table scan
```

Grafana Labs

User defined functions

```
select = (db=”telegraf”, m, f) => {
  return from(db:db)
    |> filter(fn: (r) => r._measurement == m
      and r._field == f)
}

select(m: “cpu”, f: “usage_user”)
  |> filter(fn: (r) => r[“host”] == “myServer”)
  |> range(start: -1h)
```

Grafana Labs

Chainable user defined functions

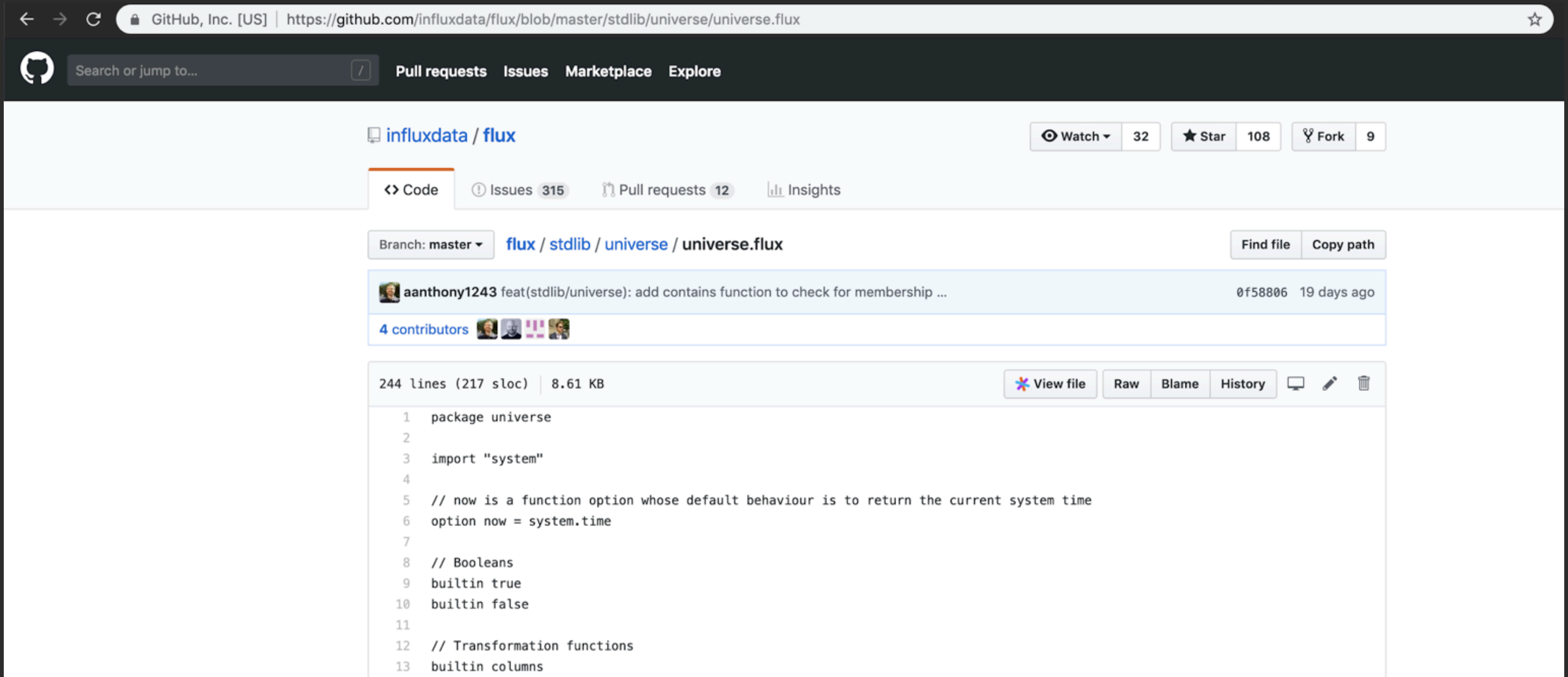
```
myFilter = (m, f, table=<-) => {
  return table
    |> filter(fn: (r) => r._measurement == m
      and r._field == f)
}

from(db: “telegraf”)
  |> myFilter(m: “cpu”, f: “usage_user”)
  |> range(start: -1h)
```

Grafana Labs

Flux StdLib

https://github.com/influxdata/flux/tree/master/stdlib


GitHub, Inc. [US] | https://github.com/influxdata/flux/blob/master/stdlib/universe/universe.flux
Search or jump to...
Pull requests
Issues
Marketplace
Explore
influxdata / flux
Watch 32
Star 108
Fork 9
Code
Issues 315
Pull requests 12
Insights
Branch: master
flux / stdlib / universe / universe.flux
Find file
Copy path
aanthony1243 feat(stdlib/universe): add contains function to check for membership ...
0f58806 19 days ago
4 contributors
244 lines (217 sloc) 8.61 KB
View file
Raw
Blame
History

```
package universe

import "system"

// now is a function option whose default behaviour is to return the current system time
option now = system.time

// Booleans
builtin true
builtin false

// Transformation functions
builtin columns
```

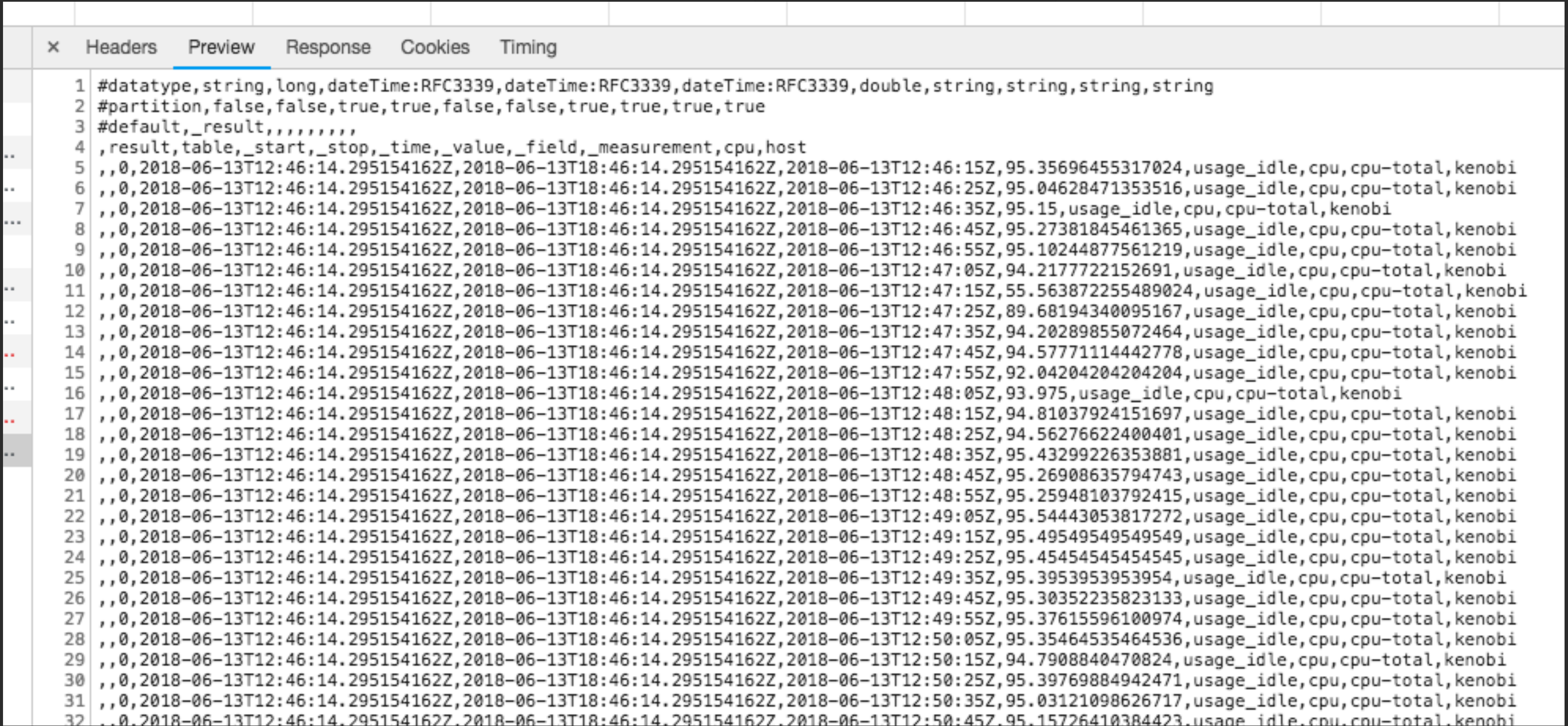
Math on tables

```
cpu = from(db)...

CpuRequests = from(db)...

join(
  tables: {cpu: cpu, req: CpuRequests},
  fn: (t) => t.cpu._value / t.req._value
) // Implicit join on time
```

New response format: CSV



```
#datatype,string,long,dateTime:RFC3339,dateTime:RFC3339,dateTime:RFC3339,double,string,string,string,string
#partition,false,false,true,true,false,false,true,true,true,true
#default,_result,,,,,,,,,
,result,table,_start,_stop,_time,_value,_field,_measurement,cpu,host
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:46:15Z,95.35696455317024,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:46:25Z,95.04628471353516,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:46:35Z,95.15,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:46:45Z,95.27381845461365,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:46:55Z,95.10244877561219,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:47:05Z,94.2177722152691,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:47:15Z,55.563872255489024,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:47:25Z,89.68194340095167,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:47:35Z,94.20289855072464,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:47:45Z,94.57771114442778,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:47:55Z,92.04204204204204,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:48:05Z,93.975,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:48:15Z,94.81037924151697,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:48:25Z,94.56276622400401,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:48:35Z,95.43299226353881,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:48:45Z,95.26908635794743,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:48:55Z,95.25948103792415,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:49:05Z,95.54443053817272,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:49:15Z,95.49549549549549,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:49:25Z,95.45454545454545,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:49:35Z,95.3953953953954,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:49:45Z,95.30352235823133,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:49:55Z,95.37615596100974,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:50:05Z,95.35464535464536,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:50:15Z,94.7908840470824,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:50:25Z,95.39769884942471,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:50:35Z,95.03121098626717,usage_idle,cpu,cpu-total,kenobi
,,0,2018-06-13T12:46:14.295154162Z,2018-06-13T18:46:14.295154162Z,2018-06-13T12:50:45Z,95.15726410384423,usage_idle,cpu,cpu-total,kenobi
```

Getting Started With Flux

- Run latest influxd
  - https://portal.influxdata.com/downloads
  - Update your influxdb.conf to include

```
# ...
[http]
  # ...
  flux-enabled = true
  # ...
```

- Generate data
  - Telegraf

Grafana Labs

Get started: Grafana datasource

- Get Grafana 5.3+
- Install Flux datasource plugin
  - https://github.com/grafana/influxdb-flux-datasource
  - Clone into your grafanas data/plugins
  - Restart Grafana
- Add your Flux datasource
- Add a dashboard
- Add a panel

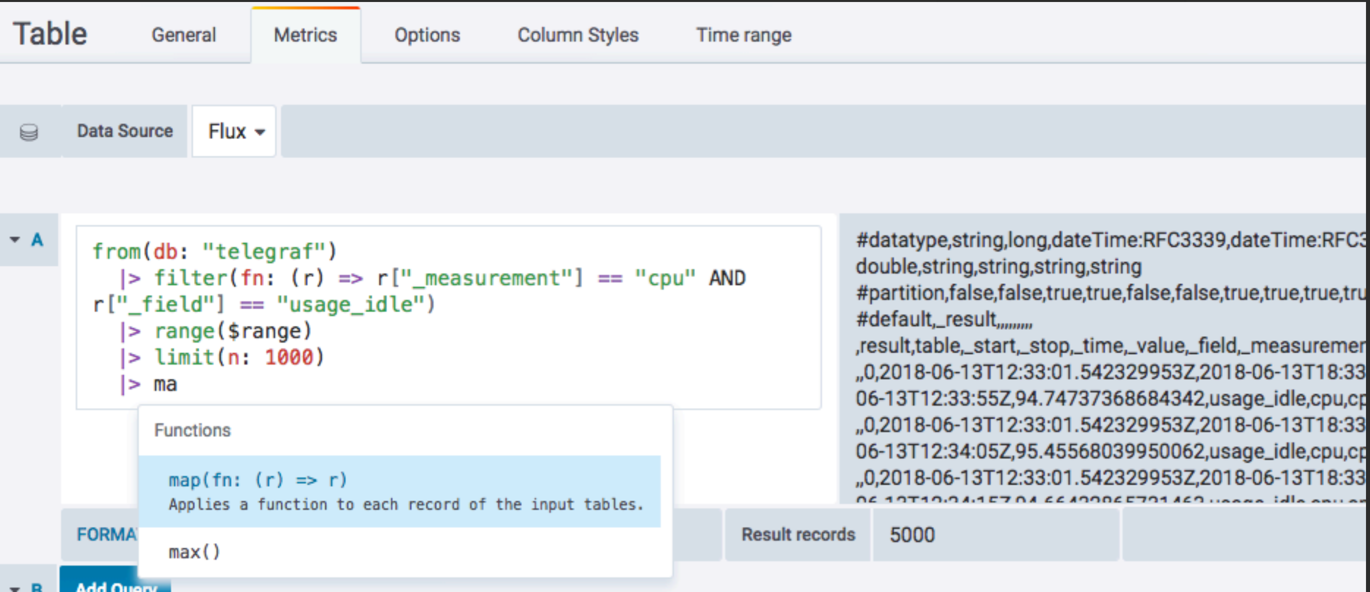
Grafana Labs

Demo

https://github.com/jtlisi/grafana_flux_demo

Datasource feature summary

- Syntax highlighting, tab completion, raw table preview
- Inline function documentation
- $range variable



```
from(db: "telegraf")
  |> filter(fn: (r) => r["_measurement"] == "cpu" AND
r["_field"] == "usage_idle")
  |> range($range)
  |> limit(n: 1000)
  |> ma
```

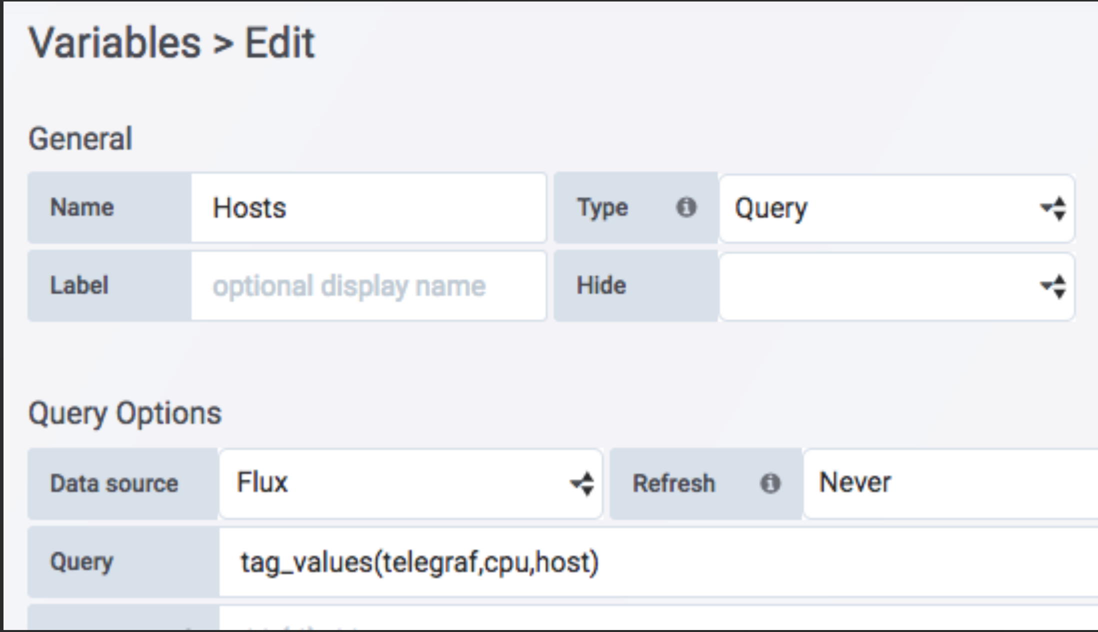
```
map(fn: (r) => r)
Applies a function to each record of the input tables.
max()
```

```
#datatype,string,long,dateTime:RFC3339,dateTime:RFC3
double,string,string,string,string
#partition,false,false,true,true,false,false,true,true,true,tru
#default,_result,,,,,,,,
,result,table,_start,_stop,_time,_value,_field,_measuremen
,,0,2018-06-13T12:33:01.542329953Z,2018-06-13T18:33
06-13T12:33:55Z,94.74737368684342,usage_idle,cpu,cp
,,0,2018-06-13T12:33:01.542329953Z,2018-06-13T18:33
06-13T12:34:05Z,95.45568039950062,usage_idle,cpu,cp
,,0,2018-06-13T12:33:01.542329953Z,2018-06-13T18:33
```

Grafana Labs

Datasource feature summary

- Shortcodes
- Template variables with helper functions
  - measurements()
  - field_keys()
  - tags()
  - tag_values()
- Annotations



Variables > Edit

General

Name	Hosts	Type	Query
Label	optional display name	Hide	

Query Options

Data source	Flux	Refresh	Never
Query	tag_values(telegraf,cpu,host)		

Grafana Labs

Roadmap

- Alerting
- UI improvements
- Improve Query Shortcuts
- Transitioning to a default plugin
- Dashboard Migrations?

Thanks for listening!
Questions?

@JacobLisi

Grafana Labs