

Программирование с Grafana и InfluxDB: сборник рецептов

Смирнов Вячеслав



HEISENBUG



**Рассказ о том, как
выделить неделю на
программирование,
и сэкономить недели
на получении
отчетов по нагрузке**

**Ускоряю
Дистанционное
Банковское
Обслуживание
юридических
лиц**



**Писал
отчеты по тестированию
много лет, в разных тулах,
от Word до Confluence;
гордясь их подробностью,
используя лишь руки**

1. Результаты

1.1. Объекты тестирования

1.2. Цели и итоги

1.2.1. Причины

1.2.2. Виды тестирования

1.3. Проверенные гипотезы

1.4. Обнаруженные проблемы

1. Результаты

2. Результаты по видам тестов

2.1. Нагрузочное тестирование

2.2. Тестирование стабильности

2.3. Объёмное тестирование

2.4. Стрессовое тестирование

1. Результаты
2. Результаты по видам тестов
3. **Архитектура и конфигурации системы**
 - 3.1. Структура тестового стенда
 - 3.2. Конфигурация балансировщика
 - 3.3. Конфигурация ...

1. Результаты
2. Результаты по видам тестов
3. Архитектура и конфигурации системы
4. Результаты синтетического тестирования узлов стенда
 - 4.1. ...

1. Результаты
2. Результаты по видам тестов
3. Архитектура и конфигурации системы
4. Результаты синтетического тестирования узлов стенда
5. Рекомендации ...

Неделя работы

Десятки страниц

Один просмотр

Ноль комментариев

Неделя работы

Десятки страниц

Один просмотр

Ноль комментариев

Писал

**отчеты по тестированию
много лет, в разных тулах,
от Word до Confluence;
гордясь их подробностью,
используя лишь руки**

**Автоматизируем
формирование
отчётов с
Grafana и InfluxDB**

Это продолжение доклада



youtu.be/sEcudxQB62M?t=573

Это продолжение доклада



[speakerdeck.com/polarnik/
proghrammirovaniie-s-grafana-i-influxdb](https://speakerdeck.com/polarnik/proghrammirovaniie-s-grafana-i-influxdb)

**В текущем докладе
новая информация**

1. Подход к разработке мониторинга
2. Инструменты для нагрузки и InfluxDB
3. Подготовка окружения разработчика
4. Делаем много баз данных и выбор баз
5. Фильтрация списков тегов
6. Кеш InfluxQL в Variable и отклонения
7. Сложные таблицы в Grafana и % успехов
8. Длительность теста и колонка Time
9. Переход к отчёту по ссылке
10. Демонстрация



1. Подход к разработке мониторинга

2. Инструменты для нагрузки и InfluxDB

3. Подготовка окружения разработчика

4. Делаем много баз данных и выбор баз

5. Фильтрация списков тегов

6. Кеш InfluxQL в Variable и отклонения

7. Сложные таблицы в Grafana и % успехов

8. Длительность теста и колонка Time

9. Переход к отчёту по ссылке

10. Демонстрация



Подход к разработке мониторинга

Если делать задачу с 0-ля:

- а какой отчет нужен?
- какие графики нужны?
- **ГОТОВИМ схему данных**
- **ГОТОВИМ простой отчет**

Если схема данных есть:

- а какой отчет нужен?
- какие графики нужны?
- **ГОТОВИМ СЛОЖНЫЙ ОТЧЕТ С ГОТОВОЙ СХЕМОЙ ДАННЫХ**

Если схема данных есть:

- а какой отчет нужен?
- какие графики нужны?
- ~~ГОТОВИМ СЛОЖНЫЙ ОТЧЕТ С~~
~~ГОТОВОЙ СХЕМОЙ ДАННЫХ~~

Если схема данных есть:

- а какой отчет нужен?
- какие графики нужны?
- **изменяем схему данных**
- **готовим простой отчет**

**Схему данных InfluxDB
можно изменять.**

**Чтобы запросы на
выборку данных были
простыми и быстрыми.**

1. Подход к разработке мониторинга
- 2. Инструменты для нагрузки и InfluxDB**
3. Подготовка окружения разработчика
4. Делаем много баз данных и выбор баз
5. Фильтрация списков тегов
6. Кеш InfluxQL в Variable и отклонения
7. Сложные таблицы в Grafana и % успехов
8. Длительность теста и колонка Time
9. Переход к отчёту по ссылке
10. Демонстрация



Инструменты для нагрузки и InfluxDB

Инструменты для нагрузки



APACHE

JMeter™

Могут сохранять метрики в ...



APACHE

JMeter™





Могут сохранять метрики в InfluxDB




APACHE

JMeter™

InfluxDB 1.8 поддерживается всеми

	Инструмент	Протокол	Приёмник
	Gatling	Graphite	InfluxDB 1.8
	Yandex.Tank	InfluxLine	InfluxDB 1.8
	LoadRunner Enterprise	InfluxLine	InfluxDB 1.8
	Apache.JMeter	Graphite	InfluxDB 1.8
		InfluxLine	InfluxDB 1.8
		InfluxLine 2.0	InfluxDB 2.0



Нагрузочнику важно знать InfluxDB



APACHE

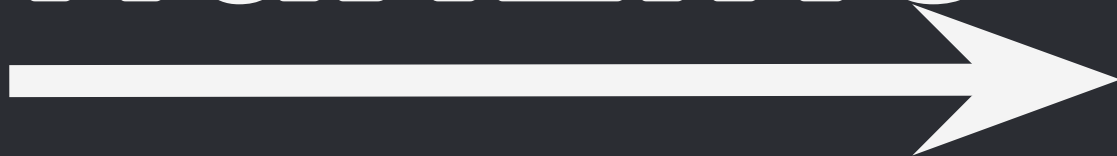
JMeter™

Уметь вставлять данные в InfluxDB



Уметь вставлять данные в InfluxDB

InfluxLine



Graphite

Уметь получать данные из InfluxDB

InfluxQL



flux

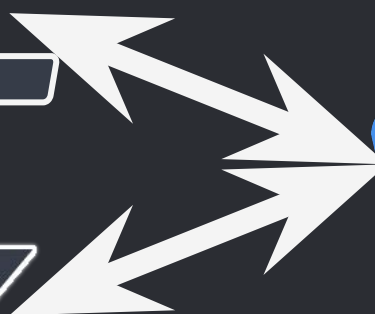
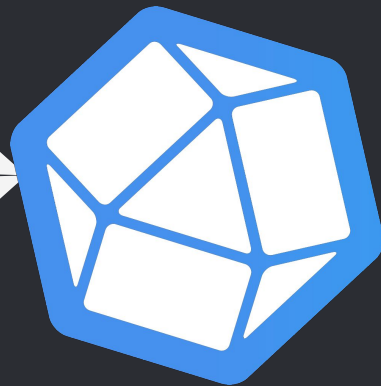


Уметь работать с данными InfluxDB

InfluxQL



flux



Уметь работать с InfluxDB из Grafana

InfluxQL



flux



Используемые версии:

 InfluxDB 1.8 (не 2.0)

 Grafana 7.0

Используемый синтаксис:

- InfluxQL (не Flux)

1. Подход к разработке мониторинга
2. Инструменты для нагрузки и InfluxDB
3. Подготовка окружения разработчика
4. Делаем много баз данных и выбор баз
5. Фильтрация выпадающих списков
6. Кеш InfluxQL в Variable и отклонения
7. Сложные таблицы в Grafana и % успехов
8. Длительность теста и колонка Time
9. Переход к отчёту по ссылке
10. Демонстрация



Подготовка окружения разработчика

**У хорошей Grafana-доски
есть версия и указана
версия окружения: версии
Grafana, Grafana-плагинов,
InfluxDB, ...**



VMware vSphere - Overview by Jorge de la Cruz

DASHBOARD

VMware vSphere Dashboard using the new Telegraf plugin
Last updated: 2 months ago

Downloads: 4801

Reviews: 6



[Add your review!](#)

Overview

Revisions

Reviews

Dashboard Revisions

Revision	Description	Created	
28	VMware vSphere Dashboard using the new Telegraf plugin	January 11th 2020, 4:04 am	Download
27	VMware vSphere Dashboard using the new Telegraf plugin	January 8th 2020, 4:12 pm	Download
26	VMware vSphere Dashboard using the new Telegraf plugin	January 8th 2020, 1:15 pm	Download
25	VMware vSphere Dashboard using the new Telegraf plugin	October 2nd 2019, 12:13 pm	Download
24	VMware vSphere Dashboard using the new Telegraf plugin	October 2nd 2019, 12:06 pm	Download
23	VMware vSphere Dashboard using the new Telegraf plugin	October 1st 2019, 8:53 pm	Download
22	VMware vSphere Dashboard using the new Telegraf plugin	June 27th 2019, 9:28 pm	Download
21	VMware vSphere Dashboard using the new Telegraf plugin	June 25th 2019, 10:09 am	Download
20	VMware vSphere Dashboard using the new Telegraf plugin	June 19th 2019, 10:59 pm	Download
19	VMware vSphere Dashboard using the new Telegraf	May 29th 2019, 1:53 am	Download

Get this dashboard:

8159

[Copy ID to Clipboard](#)

[Download JSON](#)

[How do I import this dashboard?](#)

Dependencies:

GRAFANA 6.5.0

BAR GAUGE

GAUGE

GRAPH

INFLUXDB 1.0.0

SINGLESTAT

У отличной Grafana-доски
есть репозиторий и баг-
трекер: `github`, `gitlab`, ...



Search or jump to...



Pull requests Issues Marketplace Explore



43

jorgedlcruz / vmware-grafana

Watch 6

Star 83

Fork 27

Code

Issues 4

Pull requests 0

Actions

Projects 0

Wiki

Security

Insights

A simple way to retrieve vCenter information and send it to InfluxDB, to present it later with Grafana

40 commits

2 branches

0 packages

0 releases

2 contributors

MIT

Branch: master

New pull request

Create new file

Upload files

Find file

Clone or download



jorgedlcruz Fixed minor tag error

Latest commit 37cf80b 8 days ago



LICENSE

Create LICENSE

10 months ago



README.md

Update README.md

2 months ago



VMware Stats Grafana Dashboard - PowerShell.j...

Updated to 2018 and Telegraf v1.8

17 months ago



VMware vSphere - Hosts.json

Fixed minor tag error

8 days ago

github.com/jorgedlcruz/vmware-grafana



is:open is:issue



Pull requests Issues Marketplace Explore



jorgedlcruz / vmware-grafana

Watch 6

Star 83

Fork 27

Code Issues 4 Pull requests 0 Actions Projects 0 Wiki Security Insights

Filters is:open is:issue

Labels 7

Milestones 0

New issue

Clear current search query, filters, and sorts

4 Open 15 Closed

Author Label Projects Milestones Assignee Sort

vmware-vsphere-overview: VM Disk Performance values issue

#22 opened 6 days ago by BannyRush

VMware vSphere -VMs - show only 20VMs

#21 opened on 19 Jan by lxjmac

2

VMware vSphere -VMs - show only 10VMs

#18 opened on 27 Aug 2019 by tmihaldinec

1

github.com/jorgedlcruz/vmware-grafana

Нам нужно
окружение разработчика,
для создания самых
лучших досок

Окружение храним в git:

- параметры старта Docker
- конфиги Grafana, InfluxDB
- плагины Grafana
- доски Grafana

Параметры старта Docker: InfluxDB

```
docker network create test
```

```
docker pull influxdb:1.8
```

```
DEV=$(pwd)
```

```
docker run --name=influxdb \  
  --network=test -p 8086:8086 -p 2003:2003 -p 2004:2004 \  
  -v $DEV/influxdb.conf:/etc/influxdb/influxdb.conf:ro \  
  -v $DEV/var/lib/influxdb:/var/lib/influxdb \  
  influxdb:1.8 -config /etc/influxdb/influxdb.conf
```

Параметры старта Docker: Grafana

```
docker pull grafana/grafana
```

```
ID=$(id -u)
```

```
docker run --name=grafana \  
  --network=test --user $ID -p 3000:3000 \  
  -v $DEV/grafana.ini:/etc/grafana/grafana.ini \  
  -v $DEV/plugins:/var/lib/grafana/plugins \  
  -v $DEV/provisioning:/etc/grafana/provisioning \  
  -v $DEV/var/lib/grafana:/var/lib/grafana \  
  -v $DEV/usr/share/grafana:/var/usr/share/grafana \  
  -v $DEV/var/log/grafana:/var/log/grafana \  
  grafana/grafana
```


Конфиги храним в git

```
docker pull grafana/grafana
```

```
ID=$(id -u)
```

```
docker run --name=grafana \
```

```
network=test --user $ID -p 3000:3000 \
```

```
-v $DEV/grafana.ini:/etc/grafana/grafana.ini \
```

```
-v $DEV/plugins:/var/lib/grafana/plugins \
```

```
-v $DEV/provisioning:/etc/grafana/provisioning \
```

```
-v $DEV/var/lib/grafana:/var/lib/grafana \
```

```
-v $DEV/usr/share/grafana:/var/usr/share/grafana \
```

```
-v $DEV/var/log/grafana:/var/log/grafana \
```

```
grafana/grafana
```

Плагины храним в git

```
docker pull grafana/grafana
```

```
ID=$(id -u)
```

```
docker run --name=grafana \  
  --network=test --user $ID -p 3000:3000 \  
  -v $DEV/grafana.ini:/etc/grafana/grafana.ini \  
  -v $DEV/plugins:/var/lib/grafana/plugins \  
  -v $DEV/provisioning:/etc/grafana/provisioning \  
  -v $DEV/var/lib/grafana:/var/lib/grafana \  
  -v $DEV/usr/share/grafana:/var/usr/share/grafana \  
  -v $DEV/var/log/grafana:/var/log/grafana \  
  grafana/grafana
```

Плагины из сети качаются с ошибкой

```
docker pull grafana/grafana
```

```
ID=$(id -u)
```

```
docker run --name=grafana \  
  --network=test --user $ID -p 3000:3000 \  
  -v $DEV/grafana_ini:/etc/grafana/grafana.ini \  
  -e "GF_INSTALL_PLUGINS=yesoreyeram-boomtable-panel" \  
  -v $DEV/provisioning:/etc/grafana/provisioning \  
  -v $DEV/var/lib/grafana:/var/lib/grafana \  
  -v $DEV/usr/share/grafana:/var/usr/share/grafana \  
  -v $DEV/var/log/grafana:/var/log/grafana \  
  grafana/grafana
```



Доски храним в git

```
docker pull grafana/grafana
```

```
ID=$(id -u)
```

```
docker run --name=grafana \  
  --network=test --user $ID -p 3000:3000 \  
  -v $DEV/grafana.ini:/etc/grafana/grafana.ini \  
  -v $DEV/plugins:/var/lib/grafana/plugins \  
  -v $DEV/provisioning:/etc/grafana/provisioning \  
  -v $DEV/var/lib/grafana:/var/lib/grafana \  
  -v $DEV/usr/share/grafana:/var/usr/share/grafana \  
  -v $DEV/var/log/grafana:/var/log/grafana \  
  grafana/grafana
```

Базы данных и логи в **.gitignore**

```
docker pull grafana/grafana
```

```
ID=$(id -u)
```

```
docker run --name=grafana \  
  --network=test --user $ID -p 3000:3000 \  
  -v $DEV/grafana.ini:/etc/grafana/grafana.ini \  
  -v $DEV/plugins:/var/lib/grafana/plugins \  
  -v $DEV/provisioning:/etc/grafana/provisioning \  
  -v $DEV/var/lib/grafana:/var/lib/grafana \  
  -v $DEV/usr/share/grafana:/var/usr/share/grafana \  
  -v $DEV/var/log/grafana:/var/log/grafana \  
  grafana/grafana
```

Работа с досками:

1. Процесс разработки

- идёт в `sqlite` (`.gitignore`)

2. Результат разработки

- храним в `provisioning`

Работа с досками:

1. Процесс разработки

- идёт в `sqlite` (`.gitignore`)

2. Результат разработки в `git`

- храним в `provisioning`

Доски хранятся в sqlite (default)

[database]

```
# Either "mysql", "postgres" or "sqlite3", it's your choice  
type = sqlite3
```

```
# For "sqlite3" only, path relative to data_path setting  
path = grafana.db
```

[dashboards]

```
# Number dashboard versions to keep. Default:20  
versions_to_keep = 200
```


Количество версий можно увеличить

```
[database]
```

```
# Either "mysql", "postgres" or "sqlite3", it's your choice
```

```
type = sqlite3
```

```
# For "sqlite3" only, path relative to data_path setting
```

```
path = grafana.db
```

```
[dashboards]
```

```
# Number dashboard versions to keep. Default: 20
```

```
versions_to_keep = 200
```

Работа с досками:

1. Процесс разработки

- идёт в `sqlite` (`.gitignore`)

2. Результат разработки в `git`

- храним в `provisioning`

Grafana может читать доски из папки

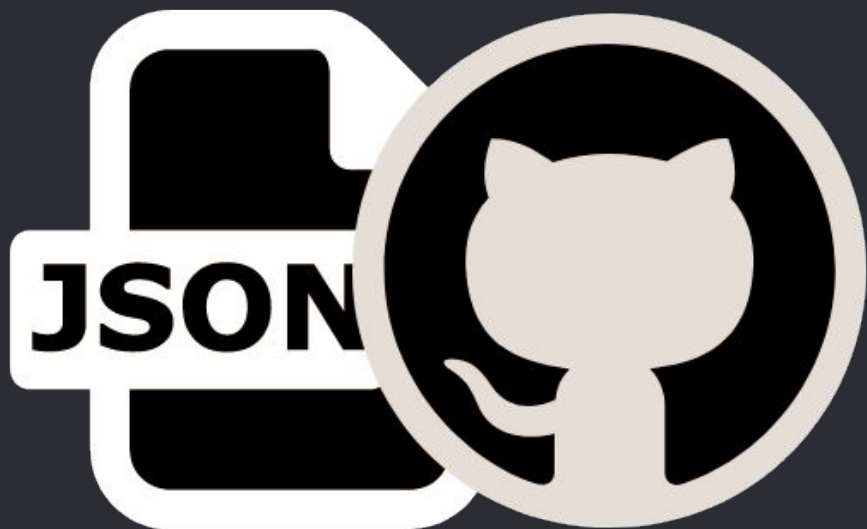


Grafana

```
/etc/grafana/provisioning
```

grafana.com/docs/grafana/latest/administration/provisioning

Docker монтирует папку в контейнер



Grafana
(docker)

`/etc/grafana/provisioning`

grafana.com/docs/grafana/latest/administration/provisioning

GIT версионировует содержимое папки



Grafana
(docker)

```
/etc/grafana/provisioning
```

grafana.com/docs/grafana/latest/administration/provisioning

Только читаем allowUiUpdates: false

```
apiVersion: 1
```

```
providers:
```

```
- name: 'GIT boards'
```

```
  orgId: 1
```

```
  folder: 'GIT'
```

```
  type: file
```

```
  allowUiUpdates: false
```

```
  updateIntervalSeconds: 0
```

```
  options:
```

```
    path: /etc/grafana/provisioning/dashboards/json
```

grafana.com/docs/grafana/latest/administration/provisioning

Вот так не надо allowUiUpdates: true

```
apiVersion: 1
```

```
providers:
```

```
- name: 'GIT boards'
```

```
  orgId: 1
```

```
  folder: 'GIT'
```

```
  type: file
```

```
  allowUiUpdates: true
```

```
  updateIntervalSeconds: 0
```

```
  options:
```

```
    path: /etc/grafana/provisioning/dashboards/json
```

grafana.com/docs/grafana/latest/administration/provisioning

Вот так не надо allowUiUpdates: true

```
apiVersion: 1
```

```
providers:
```

```
- name: 'GIT boards'
```

```
  orgId: 1
```

```
  folder: 'GIT'
```

```
  type: file
```

```
  allowUiUpdates: true
```

```
  updateIntervalSeconds: 0
```

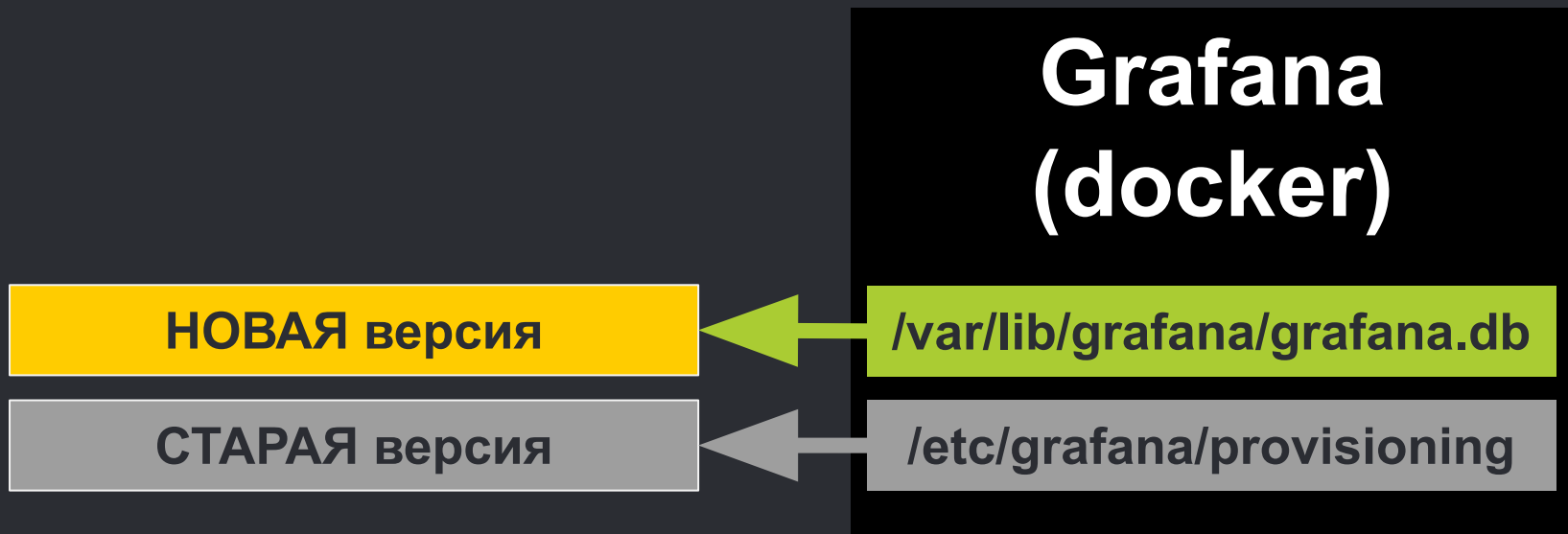
```
  options:
```

```
    path: /etc/grafana/provisioning/dashboards/boards.js
```

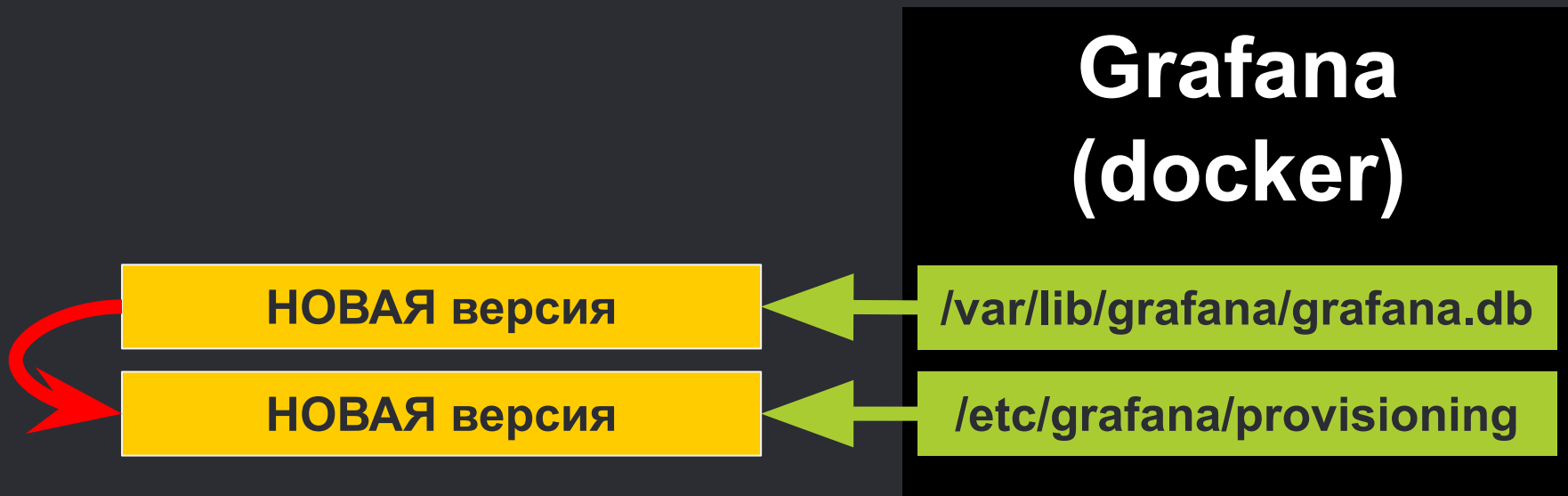
При перечитывании файла с диска правки теряются



Разрабатываем, сохраняя доску в интерфейсе Grafana (кнопка Save)



Периодически выгружаем доску из БД в файл и сохраняем в GIT



0. Получаем **API KEY** из Grafana для работы с HTTP API

API Key Created

Key eyJrIjoiTUIHcHE2ekVoZGNKOVBpS2FwajBac2RJWVNJZjBYTjQiLCJJuljoidmld2VyIiwiaWQiOjF9

You will only be able to view this key here once! It is not stored in this form. So be sure to copy it now.

You can authenticate request using the Authorization HTTP header, example:

```
curl -H "Authorization: Bearer eyJrIjoiTUIHcHE2ekVoZGNKOVBpS2FwajBac2RJWVNJZjBYTjQiLCJJuljoidmld2VyIiwiaWQiOjF9" http://localhost:3000/api/dashboards/home
```



0. Получаем **API KEY** из Grafana для работы с HTTP API

API Key Created

Key eyJrIjoiTUIHcHE2ekVoZGNKOVBpS2FwajBac2RJWVNJZjBYTjQiLCJJuljoidmld2VyIiwiaWQiOjF9

You will only be able to view this key here once! It is not stored in this form. So be sure to copy it now.

You can authenticate request using the Authorization HTTP header, example:

```
curl -H "Authorization: Bearer eyJrIjoiTUIHcHE2ekVoZGNKOVBpS2FwajBac2RJWVNJZjBYTjQiLCJJuljoidmld2VyIiwiaWQiOjF9" http://localhost:3000/api/dashboards/home
```

Add API key



1. Сохраняем файл \$UID.json с помощью **CURL** и **HTTP API**

```
#!/bin/sh -x
```

```
KEY="eyJrIjoiTU1HcHE2ekVoZGNKOVBPST2VyIiwiaWQiaWQiOiJF9"  
UID="gatlingTrend"
```

```
curl -H "Authorization: Bearer $KEY" \  
      "http://localhost:3000/api/dashboards/uid/$UID" \  
-o "/tmp/$UID.json"
```

https://grafana.com/docs/grafana/latest/http_api/dashboard

2. Выделяем из JSON-файла содержимое доски с помощью JQ

```
#!/bin/sh -x
```

```
DIR="./provisioning/dashboards/json/"
```

```
UID="gatlingTrend"
```

```
jq .dashboard "/tmp/$UID.json" > "$DIR/$UID.json"
```

<https://stedolan.github.io/jq/>

3. Добавляем к имени доски суффикс **GIT** с помощью **JQ**

```
#!/bin/sh -x
DIR="./provisioning/dashboards/json/"
UID="gatlingTrend"
TITLE=`jq -r '.title' "$FOLDER/$UID.json`
tmpFile=$(mktemp)
jq --arg a "${TITLE} (GIT)" '.title=$a' $DIR/$UID.json > \
    "$tmpFile"
mv "$tmpFile" "$FOLDER/$UID.json"
```

<https://ru.wikipedia.org/wiki/Sed>

4. Добавляем к UID доски суффикс **GIT** с помощью **JQ**

```
#!/bin/sh -x
DIR="./provisioning/dashboards/json/"
UID="gatlingTrend"

tmpFile=$(mktemp)
jq --arg a "${UID}_GIT" '.uid=$a' $DIR/$UID.json > \
  "$tmpFile"
mv "$tmpFile" "$FOLDER/$UID.json"
```

<https://ru.wikipedia.org/wiki/Sed>

5. Сохраняем JSON-файл в GIT

```
#!/bin/sh -x
```

```
FOLDER="$DEV/provisioning/dashboards/json/"  
UID="gatlingTrend"
```

```
git add "$FOLDER/$UID.json"  
git commit -m "Update $UID"
```

<https://git-scm.com/docs/git-commit>

5. Все вместе

```
#!/bin/sh -x
```

```
# API-KEY Grafana, need change  
KEY="eyJrIjoiR3dsbXA0SzdldEtudVNOdmF5YnM0dD1DeXN1bW9nY3UiLCJuIjoiVmllld2VyiIiwiaWQiOiJF9"  
# Grafana Dashboard UID, need change  
UID="gatlingTrend"  
DIR="./provisioning/dashboards/json/"
```

```
tmpFile=$(mktemp)  
curl -H "Authorization: Bearer $KEY" \  
      "http://localhost:3000/api/dashboards/uid/ $UID" \  
      -o "$tmp"  
jq .dashboard "$tmpFile" > "$DIR/$UID.json"
```

```
TITLE=`jq -r '.title' "$DIR/$UID.json`\  
jq --arg a "${TITLE} (GIT)" '.title = $a' "$DIR/$UID.json" > "$tmpFile"  
mv "$tmpFile" "$DIR/$UID.json"  
jq --arg a "${UID}_GIT" '.uid = $a' "$DIR/$UID.json" > "$tmpFile"  
mv "$tmpFile" "$DIR/$UID.json"
```

```
git add "$DIR/$UID.json"  
git commit -m "Update $UID"
```



Настроенное окружение



[github.com/polarnik/
gatling-grafana-dashboard](https://github.com/polarnik/gatling-grafana-dashboard)

1. Подход к разработке мониторинга
2. Инструменты для нагрузки и InfluxDB
3. Подготовка окружения разработчика
- 4. Делаем много баз данных и выбор баз**
5. Фильтрация списков тегов
6. Кеш InfluxQL в Variable и отклонения
7. Сложные таблицы в Grafana и % успехов
8. Длительность теста и колонка Time
9. Переход к отчёту по ссылке
10. Демонстрация



**Делаем много баз
данных и выбор баз**

Есть две редакции InfluxDB:

- OpenSource без кластера
- Enterprise с кластером

Есть две редакции InfluxDB:

- **OpenSource без кластера**
 - есть опыт работы
- **Enterprise с кластером**

Есть две редакции InfluxDB:

- OpenSource без кластера
 - **удобно использовать несколько серверов**

Есть две редакции InfluxDB:

- OpenSource без кластера
 - **удобно использовать несколько серверов**
 - «масштабирование»

InfluxDB не любит:

- огромных баз данных
- неуникальных тегов
- длинных строк в тегах

InfluxDB не любит:

- **огромных баз данных**
- **неуникальных тегов**
- **длинных строк в тегах**

InfluxDB не любит:

- огромных баз данных
 - удобно использовать несколько баз данных

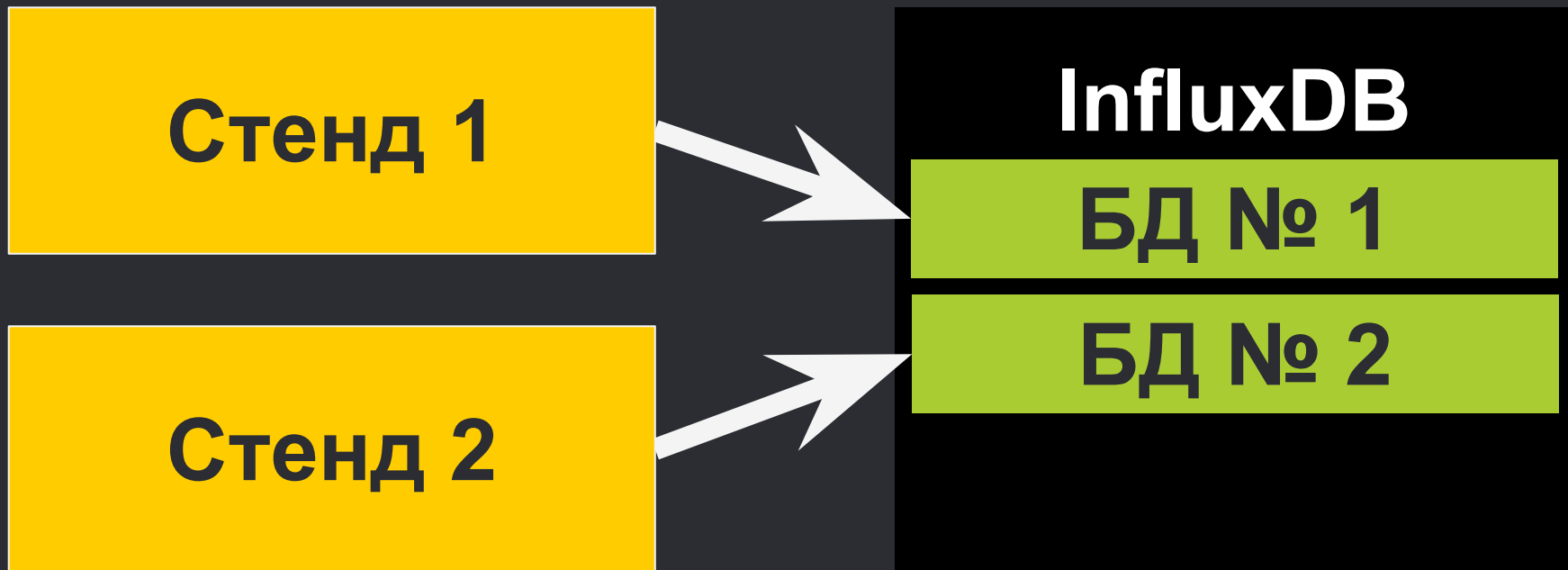
InfluxDB не любит:

- огромных баз данных
 - удобно использовать несколько баз данных
 - «шардирование»

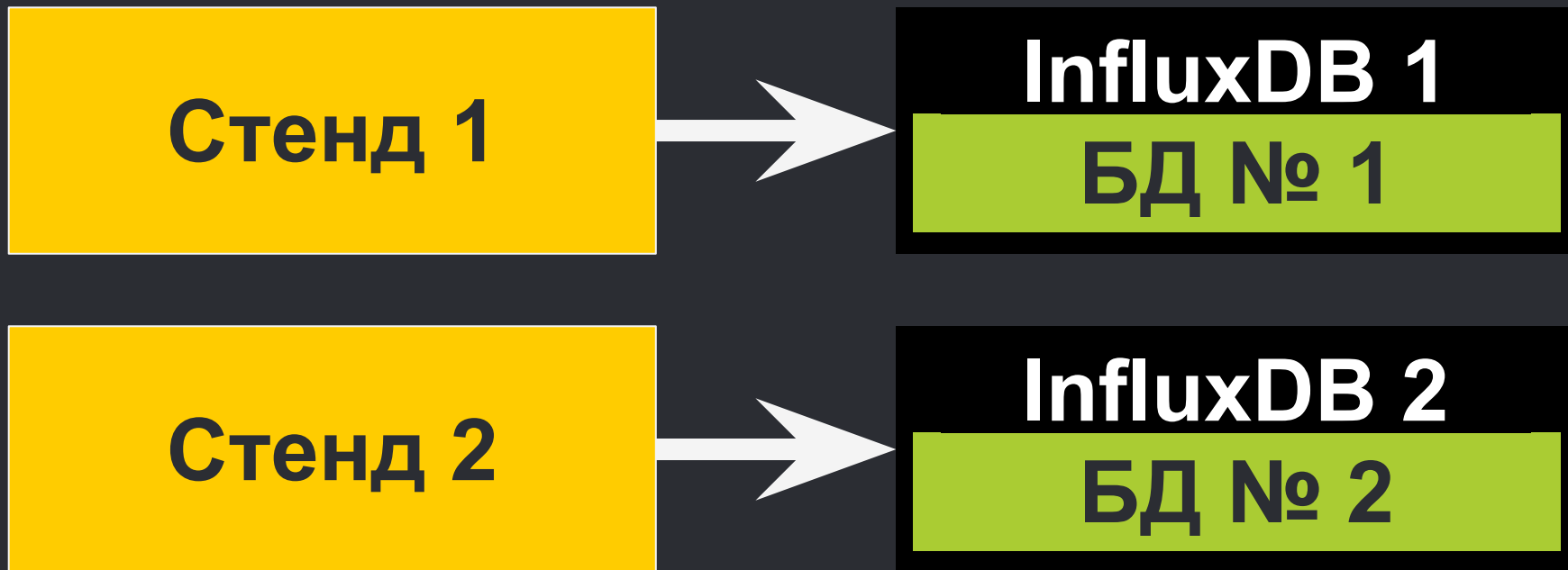
Плохо: все пишут в одну БД



Хорошо: у каждого своя БД



Отлично: у каждого своя БД и сервер



Ускорение InfluxDB OSS:

- несколько серверов
- несколько баз данных
- сложнее поддерживать
- проще использовать

Несколько Data Sources в Grafana

Data Sources

Users

Teams

Plugins

Preferences

API Keys

Search by name or type

Add data source



Gatling_Team1

http://influxdb_team1:8086

INFLUXDB



Gatling_Team2

http://influxdb_team2:8086



Gatling_Team3

http://influxdb_team3:8086



**Grafana-доске нужно
переключение между
Data Sources с InfluxDB**

Data Sources выбираем регуляркой

`/.*gatling.*/i`, `/.*telegraf.*/i`

Name	ds	Type ⓘ	Datasource ▾
Label	data source	Hide	▾

Data source options

Type	InfluxDB ▾
Instance name filter ⓘ	/.*Gatling.*/i



Data Sources выбираем регуляркой

`/. *gatling. */i`, `/. *telegraf. */i`

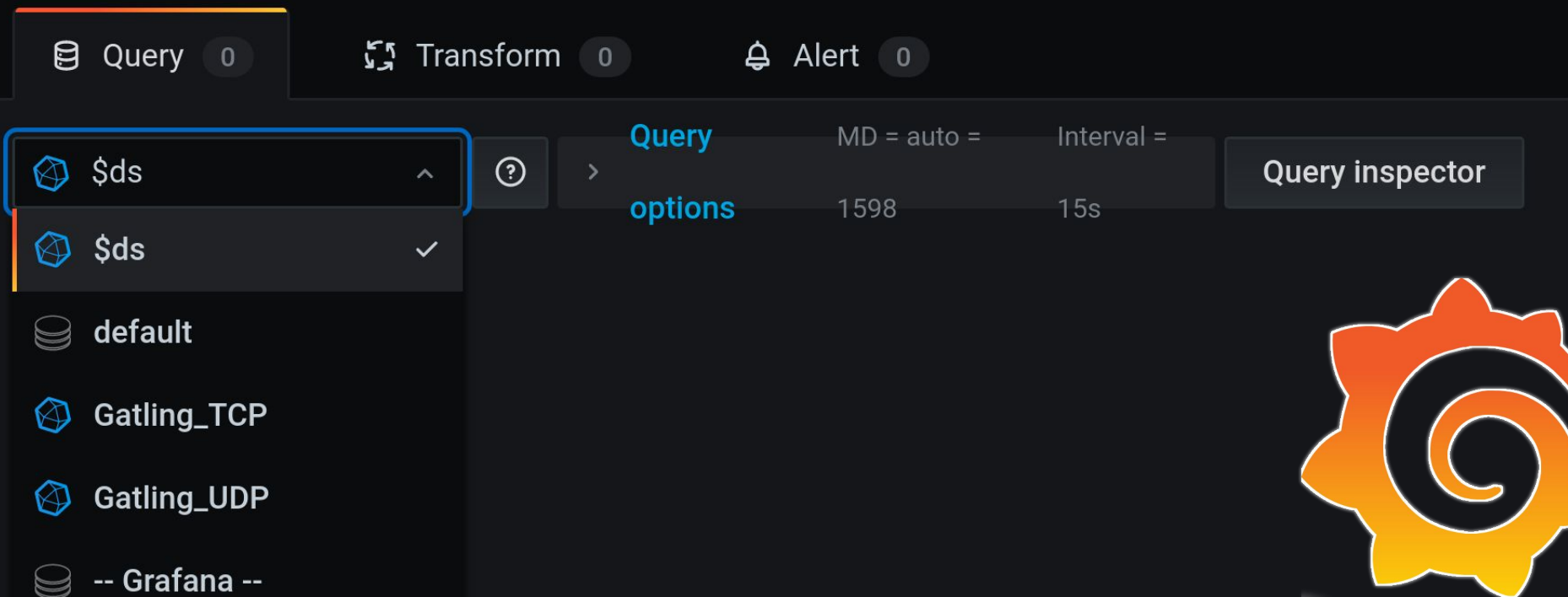
Name	ds	Type	Datasource
Label	data source	Hide	

Data source options

Type	InfluxDB
Instance name filter	<code>/. *Gatling. */i</code>



Data Sources задаем переменной **\$ds** для всех запросов



The screenshot shows the Grafana interface with a dark theme. At the top, there are three tabs: 'Query' (0), 'Transform' (0), and 'Alert' (0). Below the tabs, a dropdown menu is open, showing a list of data sources: '\$ds', '\$ds', 'default', 'Gatling_TCP', 'Gatling_UDP', and '-- Grafana --'. The first '\$ds' option is highlighted with a blue border and a checkmark. To the right of the dropdown, there is a '?' icon and a '>' icon. Below these icons, the text 'Query' is displayed in blue, followed by 'MD = auto =', 'Interval =', and 'options' in blue. The number '1598' is shown below 'options', and '15s' is shown below 'Interval ='. A 'Query inspector' button is visible on the right side of the interface. In the bottom right corner, there is a large, stylized gear icon with a spiral inside, colored in orange and yellow.

Query 0 Transform 0 Alert 0

\$ds ^ ? > Query MD = auto = Interval =
options 1598 15s Query inspector

\$ds ✓

default

Gatling_TCP

Gatling_UDP

-- Grafana --

Data Sources задаем переменной **\$ds** для всех запросов

The screenshot shows the Grafana interface with a yellow box highlighting the data source selection dropdown. The dropdown lists '\$ds' as the selected source. Below the dropdown, other data sources like 'Gatling_TCP', 'Gatling_UDP', and '-- Grafana --' are visible. The main panel shows a query configuration with 'Query' and 'options' sections, and a 'Query inspector' button.

Query 0 Transform 0 Alert 0

\$ds ^ ?

\$ds ✓

Gatling_TCP


Gatling_UDP

-- Grafana --

Query MD = auto = Interval =

options 1598 15s

Query inspector



+ Переменная **\$ds** избавляет от ошибок настройки досок в будущем



granularity

1s ▾

Sim

Load Station

All ▾

G

Request

All ▾

User

Te



Templating

Template variables could not be initialized: Datasource named DS_GATLING was not found



Templating

Template variables could not be initialized: Datasource named DS_GATLING was not found



Templating

Template variables could not be initialized: Datasource named DS_GATLING was not found



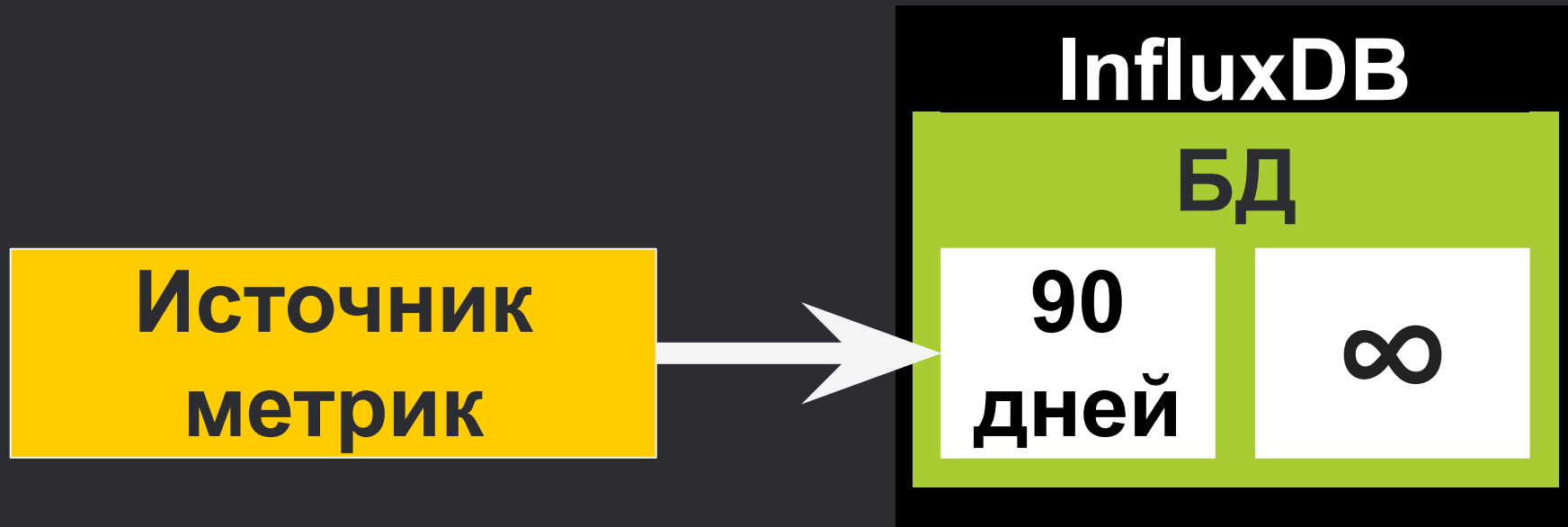
Templating



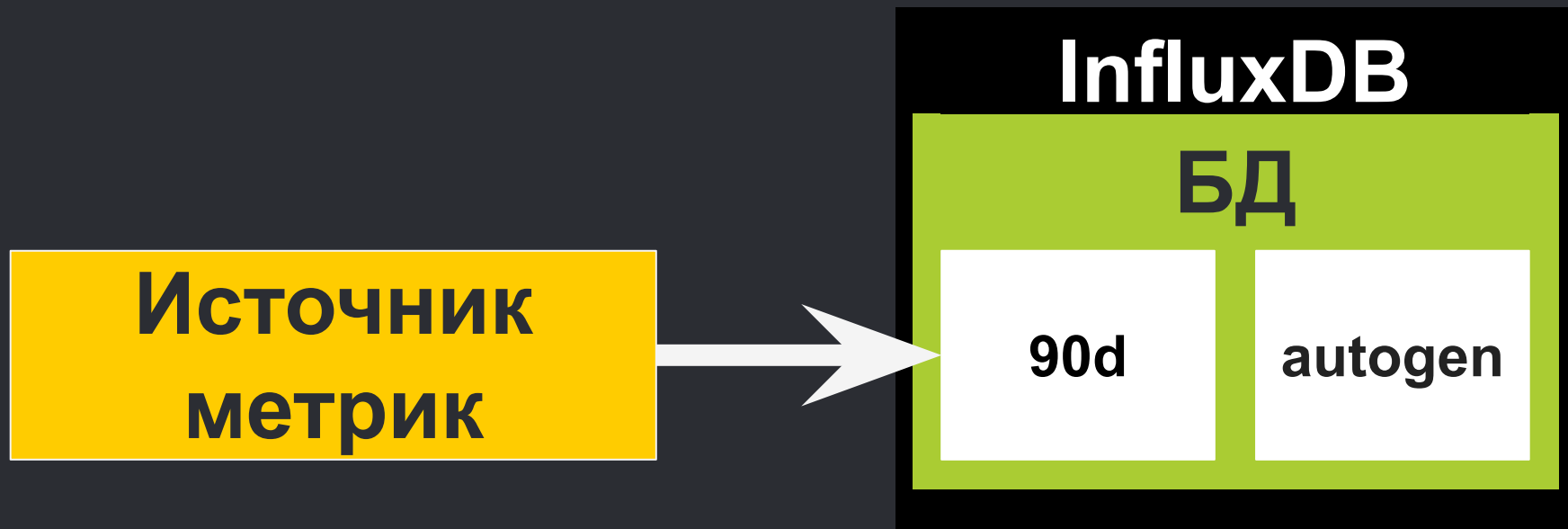
InfluxDB не любит:

- огромных баз данных
 - удобно сменить **retention policy** (default) с ∞ на **90** дней

Новые метрики храним лишь 90 дней
Исторические метрики храним вечно



Новые метрики храним лишь 90 дней
Исторические метрики храним вечно



Создаём «схему» 90d на 90 дней
(политику хранения по умолчанию)

```
CREATE RETENTION POLICY "90d"  
ON "database_name"  
DURATION 90d REPLICATION 1  
SHARD DURATION 1d  
DEFAULT
```



Удобно задать имя, как длительность
(нам это ещё пригодится в Grafana)

```
CREATE RETENTION POLICY "90d"  
ON "database_name"  
DURATION 90d REPLICATION 1  
SHARD DURATION 1d  
DEFAULT
```



В ней данные от `now() - 90d` до `now()`
(нам это ещё пригодится в Grafana)

```
CREATE RETENTION POLICY "90d"  
ON "database_name"  
DURATION 90d REPLICATION 1  
SHARD DURATION 1d  
DEFAULT
```



Заполняем новую «схему» данными
(если в autogen уже были данные)

```
SELECT *  
INTO "90d"."gatling"  
FROM "autogen"."gatling"  
WHERE time >= now() - 90d  
GROUP BY *
```



Заполняем новую «схему» данными
(если в autogen уже были данные)

Для всех измерений

```
SELECT *  
INTO "90d"."gatling.users"  
FROM "autogen"."gatling.users"  
WHERE time >= now() - 90d  
GROUP BY *
```



Теперь две политики хранения

Explore Gatling_TCP Metrics Logs

SHOW RETENTION POLICIES

FORMAT AS

Table

+ Add query

Query history

Table

name	duration	shardGroupDuration	replicaN	default
autogen	0s	168h0m0s	1	false
90d	2160h0m0s	24h0m0s	1	true



Retention Policy в Grafana задаём константой: **autogen** для архива

General

Name	archive	Type	Constant
Label	Retention policy (archive)	Hide	Variable

Constant options

Value	autogen
-------	---------

Preview of values

autogen



Retention Policy можем брать из базы

SHOW RETENTION POLICIES

General

Name	new	Type	Query
Label	Retention policy (new)	Hide	

Query Options

Data source	\$ds	Refresh	On Dashboard Load
Query	SHOW RETENTION POLICIES		
Regex	^(?!\$archive\$).*		
Sort	Alphabetical (asc)		



Выборку можно фильтровать

new != archive: **^(?!\$archive\$).***

General

Name	new	Type	Query
Label	Retention policy (new)	Hide	

Query Options

Data source	\$ds	Refresh	On Dashboard Load
Query	SHOW RETENTION POLICIES		
Regex	^(?!\$archive\$).*		
Sort	Alphabetical (asc)		



Теперь три параметра на доске



Gatling Report Trend



Last 6 hours



data source

Gatling_TCP

Retention policy (archive)

autogen

Retention policy (new)

90d



**Из-за гибкости настроек
источников данных
могут меняться и имена
measurements (измерений)**

Измерение (таблица) для Gatling

```
410
411 [[graphite]]
412 # Determines whether the graphite endpoint is enabled.
413 enabled = true
414 database = "gatling"
415 # retention-policy = ""
416 bind-address = ":2003"
417 protocol = "tcp"
418 # consistency-level = "one"
419 templates = [
420   "gatling.*.users.*.* measurement.simulation.meas
421   "gatling.*.*.*.*.*.*.*.*.* measurement.simulation
422   "gatling.*.*.*.*.*.*.*.*.* measurement.simulation.g
423   "gatling.*.*.*.*.*.*.*.*.* measurement.simulation.gro
424   "gatling.*.*.*.*.*.*.*.*.* measurement.simulation.group1
425   "gatling.*.*.*.*.*.*.*.*.* measurement.simulation.group1.group2 request
```



Измерение (таблица) для Gatling

```
410
411 [[graphite]]
412 # Determines whether the graphite endpoint is enabled.
413 enabled = true
414 database = "gatling"
415 # retention-policy = ""
416 bind-address = ":2003"
417 protocol = "tcp"
418 # consistency-level = "one"
419 templates [
420   gatling.*.users.*.* measurement.simulation.meas
421   gatling.*.*.*.*.*.*.*.*.* measurement.simulation
422   gatling.*.*.*.*.*.*.*.*.* measurement.simulation.g
423   gatling.*.*.*.*.*.*.*.*.* measurement.simulation.gro
424   gatling.*.*.*.*.*.*.*.*.* measurement.simulation.group1
425   gatling.*.*.*.*.*.*.*.*.* measurement.simulation.group1.group2.request
```



Измерение (таблица) для JMeter

File Edit Search Run Options Tools Help



PostProcessors.HTM
 Thread Group
 Backend Listener

Backend Listener

Name: Backend Listener

Comments:

Backend Listener implementation `org.apache.jmeter.visualizers.backend.influxdb`

Async Queue size `5000`

Parameters

Name:	Value
influxdbMetricsSender	org.apache.jmeter.visualizers.backend.influxdb.HttpMetricsSender
influxdbUrl	http://host_to_change:8086/write?db=jmeter
application	application name
measurement	jmeter
summaryOnly	true
samplersRegex	.*
percentiles	90;95;99
testTitle	Test name
eventTags	



Измерение (таблица) для JMeter

File Edit Search Run Options Tools Help



PostProcessors.HTM
 Thread Group
 Backend Listener

Backend Listener

Name: Backend Listener

Comments:

Backend Listener implementation `org.apache.jmeter.visualizers.backend.influxdb`

Async Queue size `5000`

Parameters

Name:	Value
influxdbMetricsSender	org.apache.jmeter.visualizers.backend.influxdb.HttpMetricsSender
influxdbUrl	http://host_to_change:8086/write?db=jmeter
application name	
measurement	jmeter
samplersRegex	.*
percentiles	90;95;99
testTitle	Test name
eventTags	



Из-за гибкости настроек
источников данных
могут меняться и имена
measurements (измерений)
но мы пока это пропустим

**Grafana-доске нужно
переключение между
Data Sources с InfluxDB
выбор Retention Policy
и выбор Measurement**

Grafana-доске нужно
переключение между
Data Sources с InfluxDB
выбор Retention Policy
и выбор Measurement

**ЕСЛИ НУЖНЫ
СКОРОСТЬ
И ГИБКОСТЬ**

1. Подход к разработке мониторинга
2. Инструменты для нагрузки и InfluxDB
3. Подготовка окружения разработчика
4. Делаем много баз данных и выбор баз
- 5. Фильтрация списков тегов**
6. Кеш InfluxQL в Variable и отклонения
7. Сложные таблицы в Grafana и % успехов
8. Длительность теста и колонка Time
9. Переход к отчёту по ссылке
10. Демонстрация



**Фильтрация
выпадающих
списков тегов
(с поддержкой
retention policy)**

Простая выборка тегов из Influx

Фильтрация по тестам (симуляциям)

General

Name	simulation	Type	Query
Label	Simulation	Hide	

Query Options

Data source	\$ds	Refresh	On Dashboard Load
Query	show tag values from "gatling" with key = "simulation"		
Regex	/.*(-.*)-.*/		
Sort	Alphabetical (asc)		



Простая выборка тегов из InfluxDB выдаёт теги



Gatling Report Trend



Last 5 minutes



data source

Gatling_TCP

Retention policy (archive)

autogen

Retention policy (new)

90d

Simulation

closemodel-incrementconcurrentusers

openmodel-atonceusers

openmodel-atonceusers-group

openmodel-rampusers

openmodel-rampusers2

openmodelatonceusers



Простая выборка тегов из InfluxDB выдаёт теги без фильтра по времени



Gatling Report Trend



Last 5 minutes



data source

Gatling_TCP

Retention policy (archive)

autogen

Retention policy (new)

90d

Simulation

closemodel-incrementconcurrentusers

openmodel-atonceusers

openmodel-atonceusers-group

openmodel-rampusers

openmodel-rampusers2

openmodelatonceusers



Refresh: On Time Range Change

нужно для фильтра \$timeFilter

General

Name	simulation	Type	Query
Label	Simulation	Hide	

Query Options

Data source	\$ds	Refresh	On Time Range Chan...
Query	show tag values from "gatling" with key = "simulation" WHERE \$timeFilter		
Regex	/*-(.*)-*/		
Sort	Alphabetical (asc)		

Простая выборка тегов из InfluxDB с фильтром по времени

```
SHOW TAG VALUES  
FROM "gatling"  
WITH KEY = "simulation"  
WHERE $timeFilter
```

Простая выборка тегов из InfluxDB с фильтром по времени

```
SHOW TAG VALUES  
FROM "gatling"  
WITH KEY = "simulation"  
WHERE $timeFilter
```

**НЕТОЧНО
НЕ ФИЛЬТРУЕТ
ПО ВРЕМЕНИ**

Размер шарды в autogen 168 часов

точность фильтрации 168 часов

SHOW RETENTION POLICIES



0.1s



FORMAT AS

Table

+ Add query

🔄 Query history

^ Table

name	duration	shardGroupDuration	replicaN	defa
autogen	0s	168h0m0s	1	fa
90d	2160h0m0s	24h0m0s	1	true



Размер шарды в 90d: 24 часа

точность фильтрации 24 часа

SHOW RETENTION POLICIES



0.1s



FORMAT AS

Table

+ Add query

🔄 Query history

^ Table

name	duration	shardGroupDuration	replicaN	defa
autogen	0s	168h0m0s	1	fal
90d	2160h0m0s	24h0m0s	1	true



Для точной выборки
по времени нужно

Explore data using InfluxQL,
а не **Explore your schema**

Для точной выборки
по времени нужно
выбрать данные
за период времени
и сгруппировать по тегу

GROUP BY "tag" в подзапросе
точно фильтрует теги по времени

```
SELECT "tag_name"  
FROM  
(SELECT last("field_name")  
  FROM "$new"."measurement"  
  GROUP BY "tag_name")  
WHERE $timeFilter
```



GROUP BY "tag" в подзапросе
легко расширяется на несколько
схем

```
SELECT DISTINCT("tag_name")  
FROM  
(SELECT last("field_name")  
FROM "$new"."measurement"  
GROUP BY "tag_name"), ...
```

WHERE \$timeFilter



GROUP BY "tag" в подзапросе точно и гибко фильтрует теги

General

Name	simulation	Type ⓘ	Query ▾
Label	Simulation	Hide	▾

Query Options

Data source	\$ds ▾	Refresh ⓘ	On Time Range Chan... ▾
Query	<pre>SELECT DISTINCT("simulation") FROM (SELECT last("mean") FROM "\$new"."gatling" GROUP BY "simulation"), (SELECT last("mean") FROM "\$archive"."gatling" GROUP BY "simulation") WHERE \$timeFilter</pre>		
Regex ⓘ	/.*-(.*)-.*/		



Дефект!: **\$timeFilter** в Variable Query МОЖНО ИСПОЛЬЗОВАТЬ ТОЛЬКО ОДИН РАЗ

General

Name

simulation

Label

Simulation

Query Options

Data source

\$ds

Refresh

On Time Range Chan...

Query

```
SELECT DISTINCT("simulation") FROM
(SELECT last("mean") FROM "$new"."gatling" WHERE $timeFilter GROUP BY "simulation")
(SELECT last("mean") FROM "$archive"."gatling" WHERE $timeFilter GROUP BY "simulation")
WHERE $timeFilter
```

Regex

/.*-(.*)-.*/



Validation

error parsing query: missing parameter: timeFilter



Templating

Template variables could not be initialized: error parsing query: missing parameter: timeFilter



Поэтому **\$timeFilter** в Variables используется ТОЛЬКО в конце запроса

General

Name	simulation	Type	Query
Label	Simulation	Hide	

Query Options

Data source	\$ds	Refresh	On Time Range Chan...
-------------	------	---------	-----------------------

Query	<pre>SELECT DISTINCT("simulation") FROM (SELECT last("mean") FROM "\$new"."gatling" GROUP BY "simulation"), (SELECT last("mean") FROM "\$archive"."gatling" GROUP BY "simulation") WHERE \$timeFilter</pre>
-------	---

Regex	<pre>/.*(-.*)-.*/</pre>
-------	-------------------------



Теперь за 5 минут нет тегов



Gatling Report Trend



Last 5 minutes



data source

Gatling_TCP

Retention policy (archive)

autogen

Retention policy (new)

90d

Simulation

None



За 90 дней есть три значения



Gatling Report Trend



Last 90 days



data source

Gatling_TCP

Retention policy (archive)

autogen

Retention policy (new)

90d

Simulation

closemodel-incrementconcurrentusers

openmodel-atonceusers

openmodel-atonceusers-group



За 6 месяцев масса значений

☰ ☰ Gatling Report Trend

🕒 Last 6 months ▾

data source

Gatling_TCP ▾

Retention policy (archive)

autogen ▾

Retention policy (new)

90d ▾

Simulation

closemodel-incrementconcurrentusers

openmodel-atonceusers

openmodel-atonceusers-group

openmodel-rampusers










































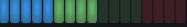






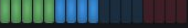















openmodel-rampusers?

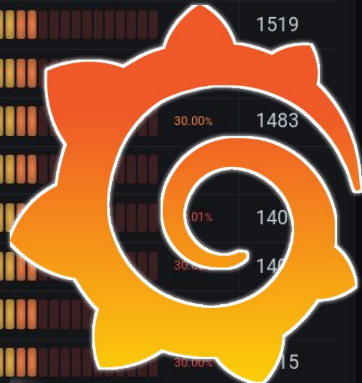


Решим прикладную задачу

Сводный отчет по запускам тестов

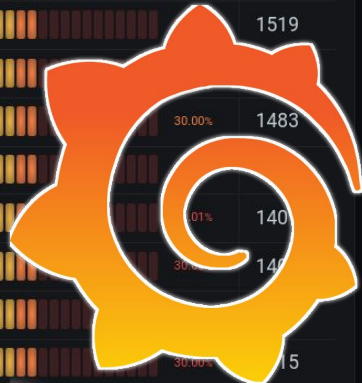
Test Details

run ▾	OK	KO	Total	Mean	Duration	Mean / median(Mean)	Duration / median(Duration)	OK / Total	KO / Total	RPS
2020-02-03_15:50	168024	72012	240036	66	229	 3115%	 297.4%	 70.00%	 30.00%	1048
2020-01-31_21:15	7	3	10	25	1	 1172%	 1.299%	 70.00%	 30.00%	10
2020-01-31_16:20	84014	36006	120020	2.0	79	 96.00%	 102.6%	 70.00%	 30.00%	1519
2020-01-31_15:30	100723	43167	143890	1.9	238	 87.33%	 309.1%	 70.00%	 30.00%	605
2020-01-30_23:16	84014	36006	120020	1.9	79	 87.94%	 102.6%	 70.00%	 30.00%	1519
2020-01-30_16:16	84014	36006	120020	2.2	79	 104.1%	 102.6%	 70.00%	 30.00%	1519
2020-01-29_20:32	84014	36006	120020	1.9	79	 90.87%	 102.6%	 70.00%	 30.00%	1519
2020-01-29_20:09	84014	36006	120020	2	79	 93.80%	 102.6%	 70.00%	 30.00%	1519
2020-01-29_20:05	84014	36006	120020	2	79	 93.80%	 102.6%	 70.00%	 30.00%	1519
2020-01-29_20:02	84014	36006	120020	2.3	79	 105.5%	 102.6%	 70.00%	 30.00%	1483
2020-01-29_19:58	81993	35135	117128	2.6	79	 122.9%	 102.6%	 70.00%	 30.00%	1483
2020-01-29_18:19	81958	35129	117087	2.3	79	 109.0%	 102.6%	 70.00%	 30.00%	1483
2020-01-29_18:16	73525	31523	105048	2.1	75	 100.1%	 97.40%	 69.99%	 30.00%	140
2020-01-29_18:08	73561	31529	105090	2.6	75	 121.9%	 97.40%	 70.00%	 30.00%	140
2020-01-29_18:05	73525	31522	105047	2.3	75	 109.4%	 97.40%	 69.99%	 30.00%	140
2020-01-29_18:03	73500	31500	105000	1.9	65	 90.19%	 84.42%	 70.00%	 30.00%	15



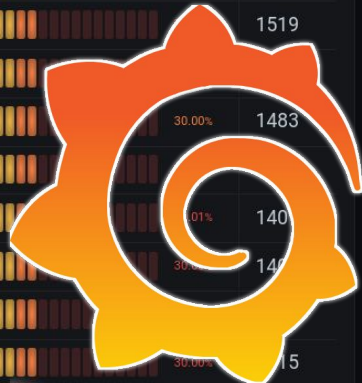
Со сводными результатами по ОК/КО

Test Details											
run	OK	KO	Total	Mean	Duration	Mean / median(Mean)	Duration / median(Duration)	OK / Total	KO / Total	RPS	
2020-02-03_15:50	168024	72012	240036	1.66	229	3115%	297.4%	70.00%	30.00%	1048	
2020-01-31_21:15	7	3	10	1.25	1	1172%	1.299%	70.00%	30.00%	10	
2020-01-31_16:20	84014	36006	120020	1.20	79	96.00%	102.6%	70.00%	30.00%	1519	
2020-01-31_15:30	100723	43167	143890	1.19	238	87.33%	309.1%	70.00%	30.00%	605	
2020-01-30_23:16	84014	36006	120020	1.19	79	87.94%	102.6%	70.00%	30.00%	1519	
2020-01-30_16:16	84014	36006	120020	1.22	79	104.1%	102.6%	70.00%	30.00%	1519	
2020-01-29_20:32	84014	36006	120020	1.19	79	90.87%	102.6%	70.00%	30.00%	1519	
2020-01-29_20:09	84014	36006	120020	1.12	79	93.80%	102.6%	70.00%	30.00%	1519	
2020-01-29_20:05	84014	36006	120020	1.12	79	93.80%	102.6%	70.00%	30.00%	1519	
2020-01-29_20:02	84014	36006	120020	1.23	79	105.5%	102.6%	70.00%	30.00%	1519	
2020-01-29_19:58	81993	35135	117128	1.26	79	122.9%	102.6%	70.00%	30.00%	1483	
2020-01-29_18:19	81958	35129	117087	1.23	79	109.0%	102.6%	70.00%	30.00%	1483	
2020-01-29_18:16	73525	31523	105048	1.21	75	100.1%	97.40%	69.99%	30.00%	140	
2020-01-29_18:08	73561	31529	105090	1.26	75	121.9%	97.40%	70.00%	30.00%	140	
2020-01-29_18:05	73525	31522	105047	1.23	75	109.4%	97.40%	69.99%	30.00%	140	
2020-01-29_18:03	73500	31500	105000	1.19	65	90.19%	84.42%	70.00%	30.00%	15	



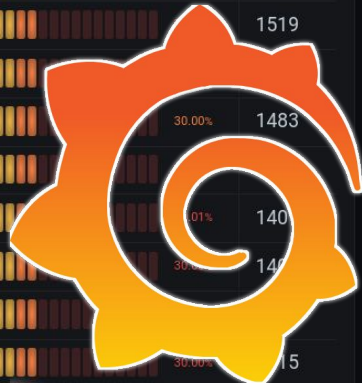
С вычислением длительности теста

Test Details												
run v	OK	KO	Total	Me	Duration	Mean / median(Mean)	Duration / median(Duration)	OK / Total	KO / Total	RPS		
2020-02-03_15:50	168024	72012	240036	6	229	3115%	297.4%	70.00%	30.00%	1048		
2020-01-31_21:15	7	3	10	2	1	1172%	1.299%	70.00%	30.00%	10		
2020-01-31_16:20	84014	36006	120020	2	79	96.00%	102.6%	70.00%	30.00%	1519		
2020-01-31_15:30	100723	43167	143890	1	238	87.33%	309.1%	70.00%	30.00%	605		
2020-01-30_23:16	84014	36006	120020	1	79	87.94%	102.6%	70.00%	30.00%	1519		
2020-01-30_16:16	84014	36006	120020	2	79	104.1%	102.6%	70.00%	30.00%	1519		
2020-01-29_20:32	84014	36006	120020	1	79	90.87%	102.6%	70.00%	30.00%	1519		
2020-01-29_20:09	84014	36006	120020	2	79	93.80%	102.6%	70.00%	30.00%	1519		
2020-01-29_20:05	84014	36006	120020	2	79	93.80%	102.6%	70.00%	30.00%	1519		
2020-01-29_20:02	84014	36006	120020	2	79	105.5%	102.6%	70.00%	30.00%	1483		
2020-01-29_19:58	81993	35135	117128	2	79	122.9%	102.6%	70.00%	30.00%	1483		
2020-01-29_18:19	81958	35129	117087	2	79	109.0%	102.6%	70.00%	30.00%	140		
2020-01-29_18:16	73525	31523	105048	2	75	100.1%	97.40%	69.99%	30.00%	140		
2020-01-29_18:08	73561	31529	105090	2	75	121.9%	97.40%	70.00%	30.00%	140		
2020-01-29_18:05	73525	31522	105047	2	75	109.4%	97.40%	69.99%	30.00%	140		
2020-01-29_18:03	73500	31500	105000	1	65	90.19%	84.42%	70.00%	30.00%	15		



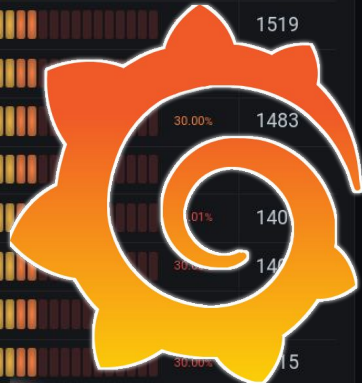
Расчетом отклонения метрик

run v	OK	KO	Total	Mean	Duration	test details		OK / Total	KO / Total	RPS
						Mean / median(Mean)	Duration / median(Duration)			
2020-02-03_15:50	168024	72012	240036	66	229	3115%	297.4%	70.00%	30.00%	1048
2020-01-31_21:15	7	3	10	25	1	1172%	1.299%	70.00%	30.00%	10
2020-01-31_16:20	84014	36006	120020	2.0	79	96.00%	102.6%	70.00%	30.00%	1519
2020-01-31_15:30	100723	43167	143890	1.9	238	87.33%	309.1%	70.00%	30.00%	605
2020-01-30_23:16	84014	36006	120020	1.9	79	87.94%	102.6%	70.00%	30.00%	1519
2020-01-30_16:16	84014	36006	120020	2.2	79	104.1%	102.6%	70.00%	30.00%	1519
2020-01-29_20:32	84014	36006	120020	1.9	79	90.87%	102.6%	70.00%	30.00%	1519
2020-01-29_20:09	84014	36006	120020	2	79	93.80%	102.6%	70.00%	30.00%	1519
2020-01-29_20:05	84014	36006	120020	2	79	93.80%	102.6%	70.00%	30.00%	1519
2020-01-29_20:02	84014	36006	120020	2.3	79	105.5%	102.6%	70.00%	30.00%	1483
2020-01-29_19:58	81993	35135	117128	2.6	79	122.9%	102.6%	70.00%	30.00%	1483
2020-01-29_18:19	81958	35129	117087	2.3	79	109.0%	102.6%	70.00%	30.00%	140
2020-01-29_18:16	73525	31523	105048	2.1	75	100.1%	97.40%	69.99%	30.00%	140
2020-01-29_18:08	73561	31529	105090	2.6	75	121.9%	97.40%	70.00%	30.00%	140
2020-01-29_18:05	73525	31522	105047	2.3	75	109.4%	97.40%	69.99%	30.00%	140
2020-01-29_18:03	73500	31500	105000	1.9	65	90.19%	84.00%	70.00%	30.00%	15



Расчётом соотношений метрик

Test Details										
run	OK	KO	Total	Mean	Duration	Mean / median(Mean)	Duration / median(Duration)	OK / Total	KO / Total	RPS
2020-02-03_15:50	168024	72012	240036	66	229	3115%	297	70.00%	30.00%	1048
2020-01-31_21:15	7	3	10	25	1	1172%	1.29	70.00%	30.00%	10
2020-01-31_16:20	84014	36006	120020	2.0	79	96.00%	102	70.00%	30.00%	1519
2020-01-31_15:30	100723	43167	143890	1.9	238	87.33%	309	70.00%	30.00%	605
2020-01-30_23:16	84014	36006	120020	1.9	79	87.94%	102	70.00%	30.00%	1519
2020-01-30_16:16	84014	36006	120020	2.2	79	104.1%	102	70.00%	30.00%	1519
2020-01-29_20:32	84014	36006	120020	1.9	79	90.87%	102	70.00%	30.00%	1519
2020-01-29_20:09	84014	36006	120020	2	79	93.80%	102	70.00%	30.00%	1519
2020-01-29_20:05	84014	36006	120020	2	79	93.80%	102	70.00%	30.00%	1519
2020-01-29_20:02	84014	36006	120020	2.3	79	105.5%	102	70.00%	30.00%	1519
2020-01-29_19:58	81993	35135	117128	2.6	79	122.9%	102	70.00%	30.00%	1483
2020-01-29_18:19	81958	35129	117087	2.3	79	109.0%	102	70.00%	30.00%	1483
2020-01-29_18:16	73525	31523	105048	2.1	75	100.1%	97.4	69.99%	30.00%	140
2020-01-29_18:08	73561	31529	105090	2.6	75	121.9%	97.4	70.00%	30.00%	140
2020-01-29_18:05	73525	31522	105047	2.3	75	109.4%	97.4	69.99%	30.00%	140
2020-01-29_18:03	73500	31500	105000	1.9	65	90.19%	84.42	70.00%	30.00%	15



1. Подход к разработке мониторинга
2. Инструменты для нагрузки и InfluxDB
3. Подготовка окружения разработчика
4. Делаем много баз данных и выбор баз
5. Фильтрация списков тегов
- 6. Кеш InfluxQL в Variable и отклонения**
7. Сложные таблицы в Grafana и % успехов
8. Длительность теста и колонка Time
9. Переход к отчёту по ссылке
10. Демонстрация



Кеш InfluxQL в Grafana Variable и ОТКЛОНЕНИЯ

Сделаем таблицу средней длительности запросов по запускам

Query 1 Transform 1

\$ds ? > Query options MD = auto = 1956 Interval = 3h Query inspector

A ✎ ↓ ↑ 📄 👁 🗑


FROM \$archive gatling WHERE request = allRequests AND status = ok AND simulation =~ /^\$simulation\$/

+ +

SELECT field (mean) mean () alias (Mean) +

GROUP BY tag (run) +

FORMAT AS Table ▼

A large, stylized logo consisting of two interlocking gears. The top gear is orange and the bottom gear is yellow, both with a white outline. The gears are positioned in the bottom right corner of the interface.

Получаем таблицу, значения в которой нужно сравнивать глазами

run ▾	Mean
2020-03-21_23:00	2.073 s
2020-03-21_22:55	1.890 s
2020-03-21_22:50	738 ms
2020-03-21_22:45	389 ms
2020-03-21_22:10	351 ms
2020-03-21_22:05	1.069 s
2020-03-21_21:50	82 ms



В Grafana есть механизм **Threshold** заливки на основе порогов значений

run ▾	Mean
2020-03-21_23:00	2.073 s
2020-03-21_22:55	1.890 s
2020-03-21_22:50	738 ms
2020-03-21_22:45	389 ms
2020-03-21_22:10	351 ms
2020-03-21_22:05	1.069 s
2020-03-21_21:50	82 ms



В Grafana есть механизм **Threshold** заливки на основе порогов значений

Test Mean Time + Mean analytics...

run ▾	Mean
2020-03-21_23:00	2.073 s
2020-03-21_22:55	1.890 s
2020-03-21_22:50	738 ms
2020-03-21_22:45	389 ms
2020-03-21_22:10	351 ms
2020-03-21_22:05	1.069 s
2020-03-21_21:50	82 ms

Color text, background, show as gauge, etc

Color background

Thresholds

+ Add threshold

● % 80

● % 50

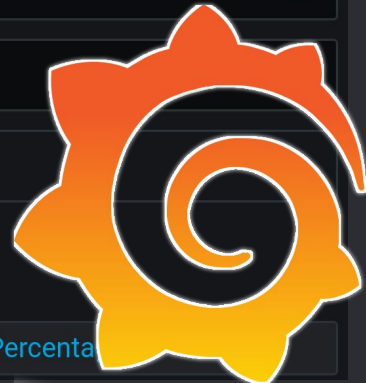
● Base

Thresholds mode

Percentage means thresholds relative to min & max

Absolute

Percentage



Threshold работает хорошо, считая % от максимума по таблице

Test Mean Time + Mean analytics...

run ▾	Mean
2020-03-21_23:00	2.073 s
2020-03-21_22:55	1.890 s
2020-03-21_22:50	738 ms
2020-03-21_22:45	389 ms
2020-03-21_22:10	351 ms
2020-03-21_22:05	1.069 s
2020-03-21_21:50	82 ms

Color text, background, show as gauge, etc

Color background

Thresholds

+ Add threshold

● % 80

● % 50

● Base

Thresholds mode

Percentage means thresholds relative to min & max

Absolute

Percentage



Threshold работает плохо, когда в таблице есть другие колонки

run ▾	OK	KO	Total	Mean	Duration
2020-03-21_23:00	156195	55856	712051	2073	99
2020-03-21_22:55	113237	42006	455243	1890	157
2020-03-21_22:50	142504	27512	570016	738	296
2020-03-21_22:45	46252	38756	185008	389	99
2020-03-21_22:10	38002	14006	152008	351	99
2020-03-21_22:05	16502	49506	66008	1069	13
2020-03-21_21:50	5502	16506	22008	82	54



Нужен способ выбирать данные, с которыми будем сравнивать

run ▾	OK	KO	Total	Mean	Duration
2020-03-21_23:00	156195	555856	712051	2073	99
2020-03-21_22:55	113237	342006	455243	1890	157
2020-03-21_22:50	142504	427512	570016	738	296
2020-03-21_22:45	46252	138756	185008	389	99
2020-03-21_22:10	38002	114006	152008	351	99
2020-03-21_22:05	16502	49506	66008	1069	13
2020-03-21_21:50	5502	16506	22008	82	54



Посчитаем отношение колонки к **max**, **mean** и **median** по колонке(ам)

run ▾	Mean
2020-03-21_23:00	2.073 s
2020-03-21_22:55	1.890 s
2020-03-21_22:50	738 ms
2020-03-21_22:45	389 ms
2020-03-21_22:10	351 ms
2020-03-21_22:05	1.069 s
2020-03-21_21:50	82 ms

Grafana поможет разделить строку на **max**, **mean** и **median** по колонке

run ▾	Mean
2020-03-21_23:00	2.073 s
2020-03-21_22:55	1.890 s
2020-03-21_22:50	738 ms
2020-03-21_22:45	389 ms
2020-03-21_22:10	351 ms
2020-03-21_22:05	1.069 s
2020-03-21_21:50	82 ms



Используем Grafana Variable, как кеш

max, mean и median по колонке

Variable	Definition					
ds	influxdb		↓	📄	🗑️	
archive	SHOW RETENTION POLICIES		↑	↓	📄	🗑️
new	SHOW RETENTION POLICIES		↑	↓	📄	🗑️
simulation	SELECT DISTINCT("simulation") FROM (SELECT last("mean") FROM "\$new"....		↑	↓	📄	🗑️
mean_max	SELECT max("Mean") FROM (SELECT mean("mean") as "Mean" FROM "\$arc...		↑		📄	🗑️
mean_mean	SELECT mean("Mean") FROM (SELECT mean("mean") as "Mean" FROM "\$ar...		↑			🗑️
mean_median	SELECT median("Mean") FROM (SELECT mean("mean") as "Mean" FROM "\$...		↑			🗑️



Максимальное среднее mean_max по запускам тестов

Query Options

Data source	\$ds	Refresh	On Time Range Chan...
Query	<pre>SELECT max("Mean") FROM (SELECT mean("mean") as "Mean" FROM "\$archive"."gatling" WHERE "request" = 'allRequests' AND "status" = 'ok' AND "simulation" =~ /^\${simulation:regex}\$/ AND \$timeFilter GROUP BY "run")</pre>		
Regex	/*-(.*)-*/		
Sort	Disabled		



Среднее среднее mean_mean по запускам тестов

Query Options

Data source	\$ds	Refresh	On Time Range Chan...
Query	<pre>SELECT mean("Mean") FROM (SELECT mean("mean") as "Mean" FROM "\$archive"."gatling" WHERE "request" = 'allRequests' AND "status" = 'ok' AND "simulation" =~ /^\${simulation:regex}\$/ AND \$timeFilter GROUP BY "run")</pre>		
Regex	/*-(.*)-*/		
Sort	Disabled		



Медианное среднее (50 перцентиль) mean_median по запускам тестов

Query Options

Data source	\$ds	Refresh	On Time Range Chan...
Query	<pre>SELECT median("Mean") FROM (SELECT mean("mean") as "Mean" FROM "\$archive"."gatling" WHERE "request" = 'allRequests' AND "status" = 'ok' AND "simulation" =~ /^\${simulation:regex}\$/ AND \$timeFilter GROUP BY "run")</pre>		
Regex	/.*(-.*)-.*/		
Sort	Disabled		



Теперь просто вычислить отношение текущего среднего к общей выборке

\$ds ? > [Query options](#) MD = auto = 932 Interval = 12h Query inspector

▼ A ✎ ↓ ↑ 📄 👁 🗑

FROM	\$archive	gating	WHERE	request	=	allRequests	AND	status	=	ok	AND	simulation	≈	/^\$simulation\$/	+
SELECT	field (mean)	mean ()	alias (Mean)	+											
	field (mean)	mean ()	math (/ \${mean_max})	alias (Mean / max(Mean))	+										
	field (mean)	mean ()	math (/ \${mean_mean})	alias (Mean / mean(Mean))	+										
	field (mean)	mean ()	math (/ \${mean_median})	alias (Mean / median(Mean))	+										
GROUP BY	tag (run)	+													
FORMAT AS	Table	▼													



Теперь просто вычислить отношение текущего среднего к переменной

\$ds ? > Query options MD = auto = 932 Interval = 12h Query inspector

▼ A ✎ ↓ ↑ 📄 👁 🗑

FROM \$archive gatling WHERE request = allRequests AND status = ok AND simulation =~ /^\$simulation\$/ +

SELECT field (mean) mean ()


field (mean) mean (math (/ \${mean_max}) alias (Mean / max(Mean)) +

field (mean) mean (math (/ \${mean_mean}) alias (Mean / mean(Mean)) +

field (mean) mean (math (/ \${mean_median}) alias (Mean / median(Mean)) +

GROUP BY tag (run) +

FORMAT AS Table ▼



И у таких отношений есть ожидаемые значения для **Treshold**

\$ds ? > Query options MD = auto = 932 Interval = 12h Query inspector

▼ A ✎ ↓ ↑ 📄 👁 🗑

FROM \$archive gatling WHERE request = allRequests AND status = ok AND simulation =~ /^\$simulation\$/ +

SELECT field (mean) mean ()


field (mean) mean (math (/ \${mean_max}) alias (Mean / max(Mean)) +

field (mean) mean (math (/ \${mean_mean}) alias (Mean / mean(Mean)) +








field (mean) mean (math (/ \${mean_median}) alias (Mean / median(Mean)) +

GROUP BY tag (run) +

FORMAT AS Table ▼










Threshold для значений $< 50\%$ от Max мы уже задавали

run ▼	Mean	Mean / max(Mean)
2020-03-21_23:00	2.073 s	 100%
2020-03-21_22:55	1.890 s	 91.18%
2020-03-21_22:50	738 ms	 35.60%
2020-03-21_22:45	389 ms	 18.76%
2020-03-21_22:10	351 ms	 16.92%
2020-03-21_22:05	1.069 s	 51.56%
2020-03-21_21:50	82 ms	 3.972%







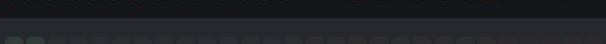
Threshold для значений $< 50\%$ от Max

мы уже задавали: **зеленый цвет**



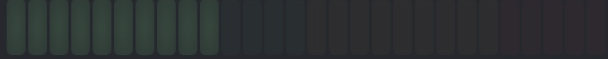




run ▾	Mean	Mean / max(Mean)
2020-03-21_23:00	2.073 s	 100%
2020-03-21_22:55	1.890 s	 91.18%
2020-03-21_22:50	738 ms	 35.60%
2020-03-21_22:45	389 ms	 18.76%
2020-03-21_22:10	351 ms	 16.92%
2020-03-21_22:05	1.069 s	 51.56%
2020-03-21_21:50	82 ms	 3.972%

Threshold от 50% до 80% от Max









тоже задавали: **желтый цвет**

run ▾	Mean	Mean / max(Mean)
2020-03-21_23:00	2.073 s	 100%
2020-03-21_22:55	1.890 s	 91.18%
2020-03-21_22:50	738 ms	 35.60%
2020-03-21_22:45	389 ms	 18.76%
2020-03-21_22:10	351 ms	 16.92%
2020-03-21_22:05	1.069 s	 51.56%
2020-03-21_21:50	82 ms	 3.972%

Threshold если выше 80% от Max прежний: красный цвет

run ▾	Mean	Mean / max(Mean)
2020-03-21_23:00	2.073 s	 100%
2020-03-21_22:55	1.890 s	 91.18%
2020-03-21_22:50	738 ms	 35.60%
2020-03-21_22:45	389 ms	 18.76%
2020-03-21_22:10	351 ms	 16.92%
2020-03-21_22:05	1.069 s	 51.56%
2020-03-21_21:50	82 ms	 3.972%

Интервал для **Mean / max(Mean)** будет от 0 до 1 (Min = 0, Max = 1)

Test Mean Time + Mean analytics		
run ▼	Mean	Mean / max(Mean)
2020-03-21_23:00	2.073 s	 100%
2020-03-21_22:55	1.890 s	 91.18%
2020-03-21_22:50	738 ms	 35.60%
2020-03-21_22:45	389 ms	 18.76%
2020-03-21_22:10	351 ms	 16.92%
2020-03-21_22:05	1.069 s	 51.56%
2020-03-21_21:50	82 ms	 3.972%
2020-02-12_21:50	230 ms	 11.10%

Override 3

Matcher > Filter by field

Set properties for fields matching the name

Mean / max(Mean)

Cell display mode

Color text, background, show as gauge, etc

LCD gauge

Max









Leave empty to calculate based on all values

1

Unit

percent (0.0-1.0)

Интервал для **Mean / max(Mean)** НЕ ЗАВИСИТ ОТ ВСЕХ ЗНАЧЕНИЙ ТАБЛИЦЫ

Test Mean Time + Mean analytics		
run ▼	Mean	Mean / max(Mean)
2020-03-21_23:00	2.073 s	 100%
2020-03-21_22:55	1.890 s	 91.18%
2020-03-21_22:50	738 ms	 35.60%
2020-03-21_22:45	389 ms	 18.76%
2020-03-21_22:10	351 ms	 16.92%
2020-03-21_22:05	1.069 s	 51.56%
2020-03-21_21:50	82 ms	 3.972%
2020-02-12_21:50	230 ms	 11.10%

Override 3

Matcher > Filter by field

Set properties for fields matching the name

Mean / max(Mean)

Cell display mode

Color text, background, show as gauge, etc

LCD gauge

Max

Leave empty to calculate based on all values










1


Unit

percent (0.0-1.0)

Интервал для Mean / mean(Mean) удобнее задать от 0 до 2

Test Mean Time + Mean analytics

run ▾	Mean	Mean / mean(Mean)
2020-03-21_23:00	2.073 s	 316.2%
2020-03-21_22:55	1.890 s	 288.3%
2020-03-21_22:50	738 ms	 112.6%
2020-03-21_22:45	389 ms	 59.30%
2020-03-21_22:10	351 ms	 53.50%
2020-03-21_22:05	1.069 s	 163.0%
2020-03-21_21:50	82 ms	 12.56%
2020-02-12_21:50	230 ms	 35.08%
2020-02-12_16:15	152 ms	 23.14%

 1,5 1 Base

Thresholds mode

Percentage means thresholds relative to min & max

Absolute

Max

Leave empty to calculate base on all values

2

Min

Leave empty to calculate base on all values

0

Threshold значений больше среднего: ЖЕЛТЫЙ ЦВЕТ

Test Mean Time + Mean analytics

run ▾	Mean	Mean / mean(Mean)
2020-03-21_23:00	2.073 s	316.2%
2020-03-21_22:55	1.890 s	288.3%
2020-03-21_22:50	738 ms	112.6%
2020-03-21_22:45	389 ms	59.30%
2020-03-21_22:10	351 ms	53.50%
2020-03-21_22:05	1.069 s	163.0%
2020-03-21_21:50	82 ms	12.56%
2020-02-12_21:50	230 ms	35.08%
2020-02-12_16:15	152 ms	23.14%

● 1,5

● 1

● Base

Thresholds mode

Percentage means thresholds relative to min & max

Absolute

Max

Leave empty to calculate based on all values

2

Min

Leave empty to calculate based on all values

0

Threshold значений больше среднего в полтора раза (x 1,5): **красный цвет**

Test Mean Time + Mean analytics

run ▾	Mean	Mean / mean(Mean)
2020-03-21_23:00	2.073 s	316.2%
2020-03-21_22:55	1.890 s	288.3%
2020-03-21_22:50	738 ms	112.6%
2020-03-21_22:45	389 ms	59.30%
2020-03-21_22:10	351 ms	53.50%
2020-03-21_22:05	1.069 s	163.0%
2020-03-21_21:50	82 ms	12.56%
2020-02-12_21:50	230 ms	35.08%
2020-02-12_16:15	152 ms	23.14%

● 1,5

● 1

● Base

Thresholds mode

Percentage means thresholds relative to min & max

 Absolute

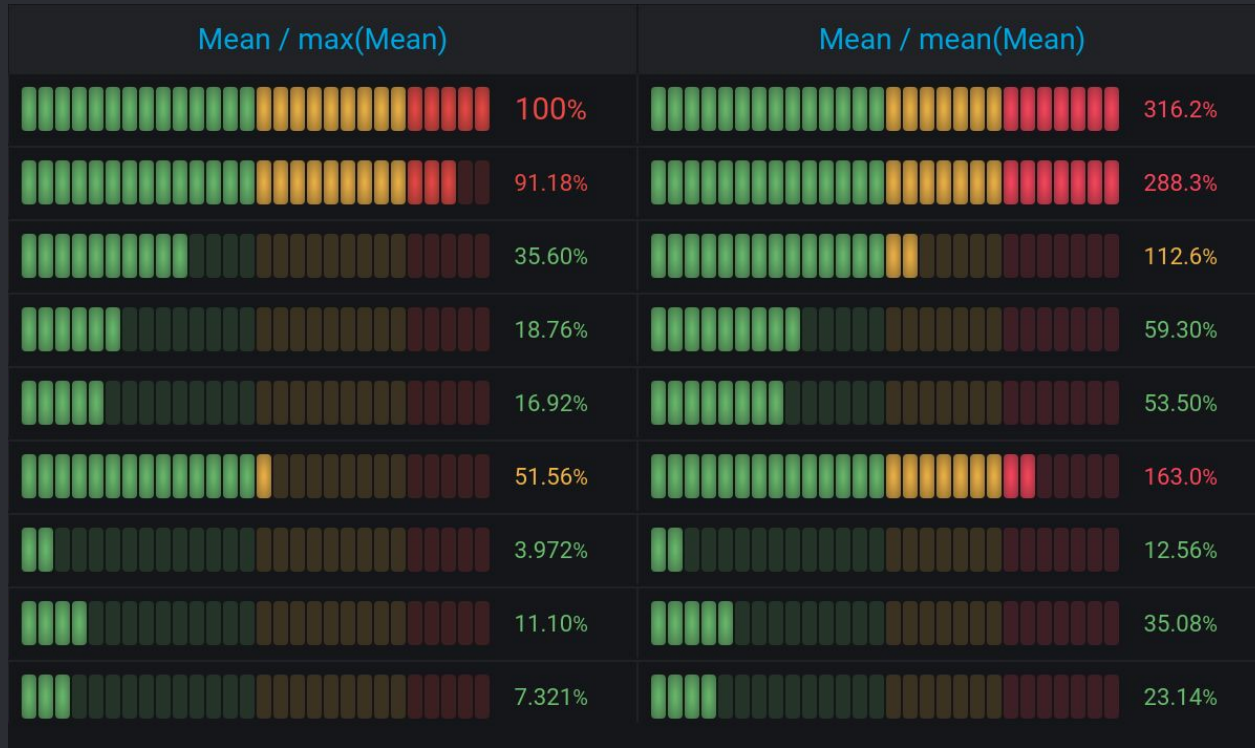
Max

Leave empty to calculate based on all values

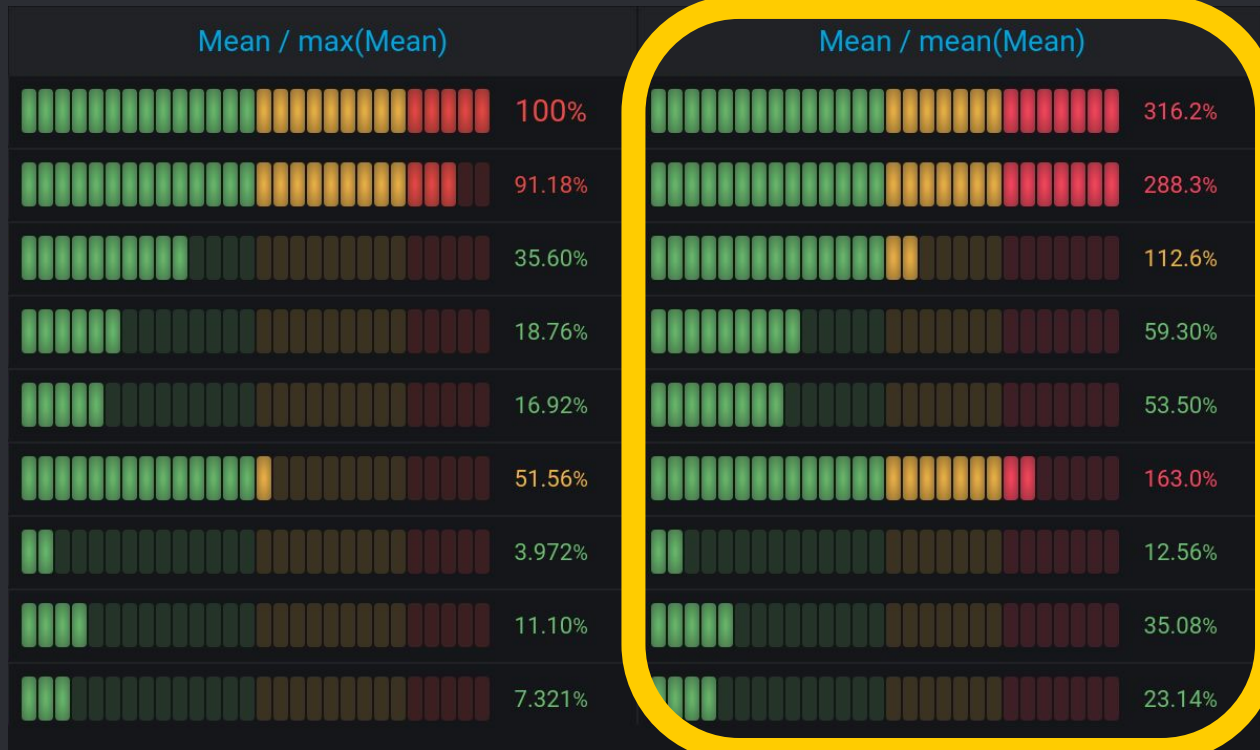
Min

Leave empty to calculate based on all values

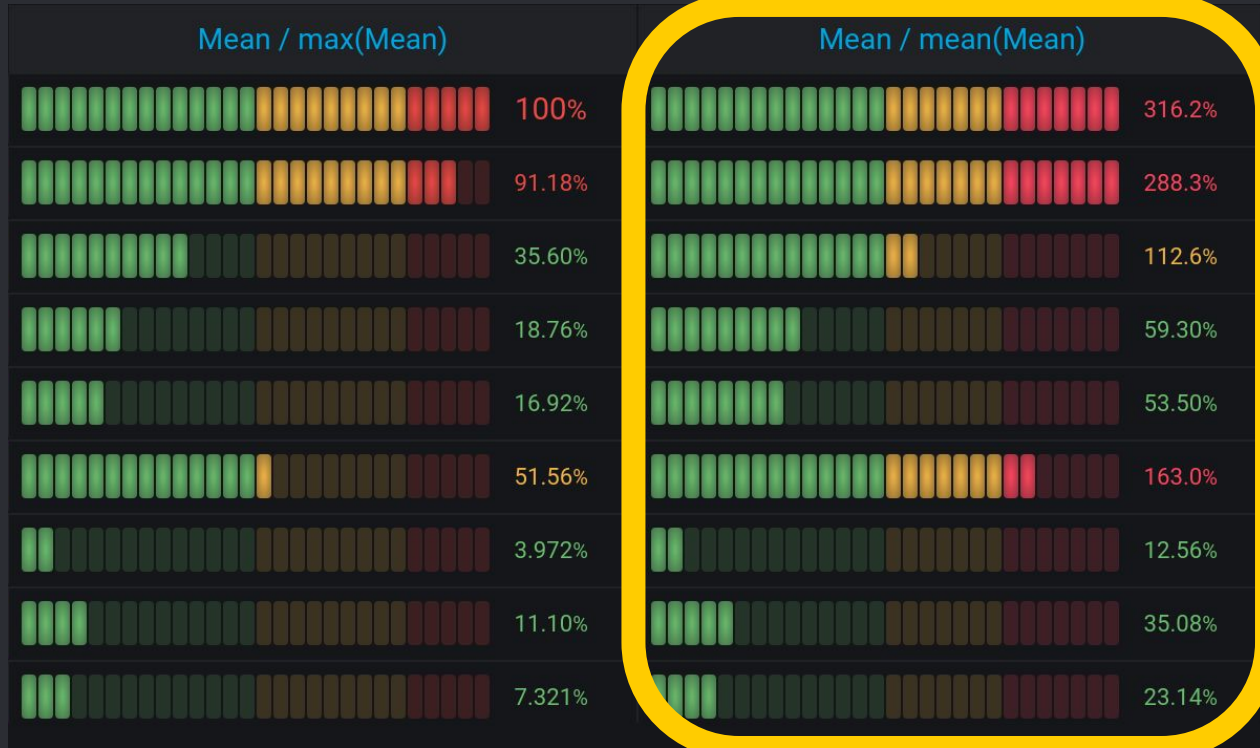
Mean / mean(Mean) нагляднее, чем Mean / max(Mean) для анализа



Mean / mean(Mean) нагляднее, чем Mean / max(Mean) для анализа



Mean / mean(Mean) подвержен влиянию локальных пиков (сбоев)



Mean / median(Mean) не подвержен влиянию локальных пиков (сбоев)

Mean / max(Mean)



Mean / mean(Mean)

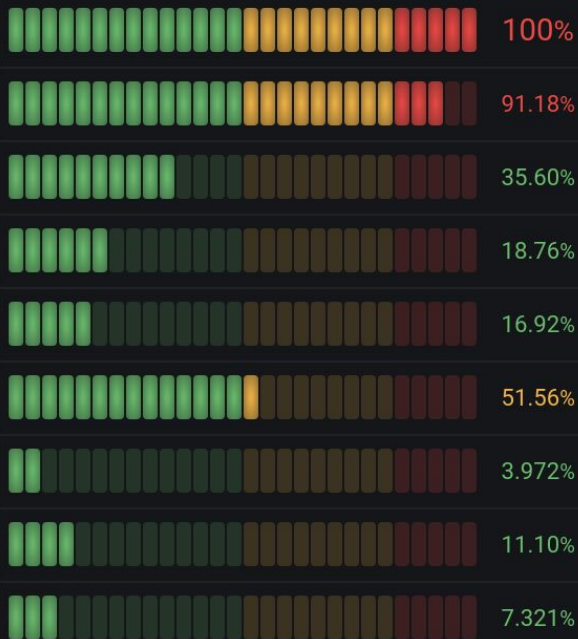


Mean / median(Mean)

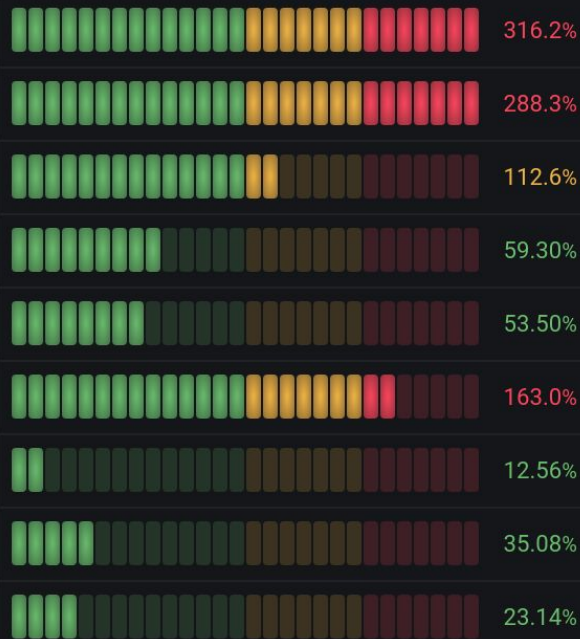


Mean / median(Mean) выявляет отклонения от общей картины

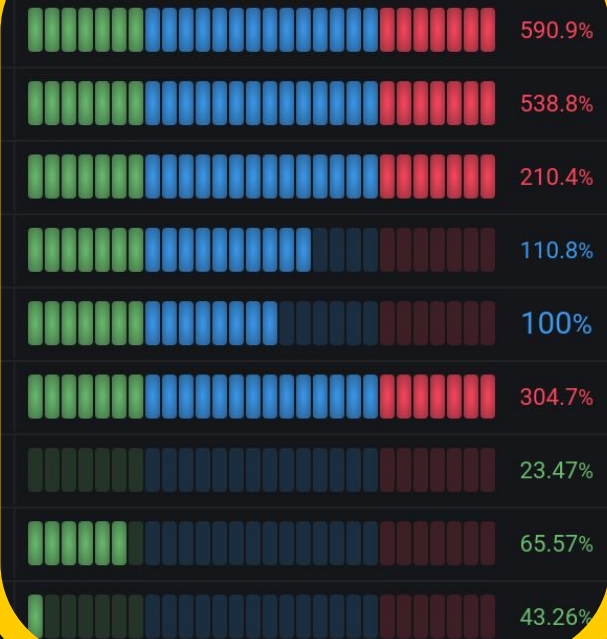
Mean / max(Mean)



Mean / mean(Mean)












Mean / median(Mean)



Mean / median(Mean) измеряет отклонение от центра (от 100%)

Test Mean Time + Mean analytics

run ▾	Mean	Mean / median(Mean)
2020-03-21_23:00	2.073 s	 590.9%
2020-03-21_22:55	1.890 s	 538.8%
2020-03-21_22:50	738 ms	 210.4%
2020-03-21_22:45	389 ms	 110.8%
2020-03-21_22:10	351 ms	 100%
2020-03-21_22:05	1.069 s	 304.7%
2020-03-21_21:50	82 ms	 23.47%
2020-02-12_21:50	230 ms	 65.57%
2020-02-12_16:15	152 ms	 43.26%

● 1,3

● 0,7

● Base

Thresholds mode

Percentage means thresholds relative to min & max

Absolute

Max

Leave empty to calculate based on all values

2

Min

Leave empty to calculate based on all values

0

Threshold для $\pm 30\%$ от 100%: синий (значение застыло, не колеблется)

Test Mean Time + Mean analytics

run ▾	Mean	Mean / median(Mean)
2020-03-21_23:00	2.073 s	590.9%
2020-03-21_22:55	1.890 s	538.8%
2020-03-21_22:50	738 ms	210.4%
2020-03-21_22:45	389 ms	110.8%
2020-03-21_22:10	351 ms	100%
2020-03-21_22:05	1.069 s	304.7%
2020-03-21_21:50	82 ms	23.47%
2020-02-12_21:50	230 ms	65.57%
2020-02-12_16:15	152 ms	43.26%

● 1,3

● 0,7

● Base

Thresholds mode

Percentage means thresholds relative to min & max

Absolute

Max

Leave empty to calculate based on all values

2

Min

Leave empty to calculate based on all values

0

Threshold для +130%: красный

(значение превысило норму, плохо)

Test Mean Time + Mean analytics

run ▾	Mean	Mean / median(Mean)
2020-03-21_23:00	2.073 s	590.9%
2020-03-21_22:55	1.890 s	538.8%
2020-03-21_22:50	738 ms	210.4%
2020-03-21_22:45	389 ms	110.8%
2020-03-21_22:10	351 ms	100%
2020-03-21_22:05	1.069 s	304.7%
2020-03-21_21:50	82 ms	23.47%
2020-02-12_21:50	230 ms	65.57%
2020-02-12_16:15	152 ms	43.26%

● 1,3

● 0,7

● Base

Thresholds mode

Percentage means thresholds relative to min & max

Absolute

Max

Leave empty to calculate based on all values

2

Min

Leave empty to calculate based on all values

0

Threshold для 0..70%: зеленый

(значение ниже нормы, это хорошо)

Test Mean Time + Mean analytics

run ▾	Mean	Mean / median(Mean)
2020-03-21_23:00	2.073 s	590.9%
2020-03-21_22:55	1.890 s	538.8%
2020-03-21_22:50	738 ms	210.4%
2020-03-21_22:45	389 ms	110.8%
2020-03-21_22:10	351 ms	100%
2020-03-21_22:05	1.069 s	304.7%
2020-03-21_21:50	82 ms	23.47%
2020-02-12_21:50	230 ms	65.57%
2020-02-12_16:15	152 ms	43.26%

● 1,3

● 0,7

● Base

Thresholds mode

Percentage means thresholds relative to min & max

Absolute

Max

Leave empty to calculate based on all values

2

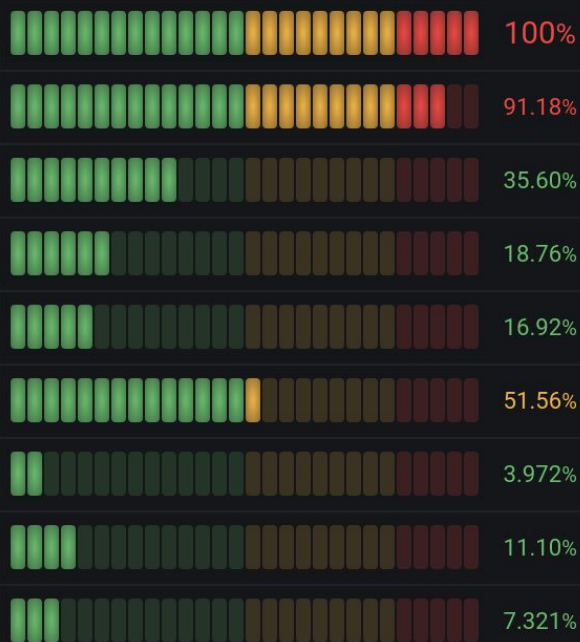
Min

Leave empty to calculate based on all values

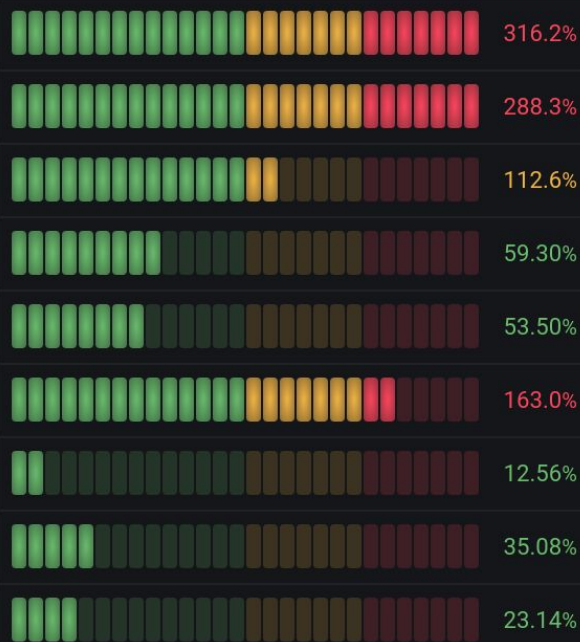
0

Выбрал **Mean / median(Mean)** для поиска отклонения от общей картины

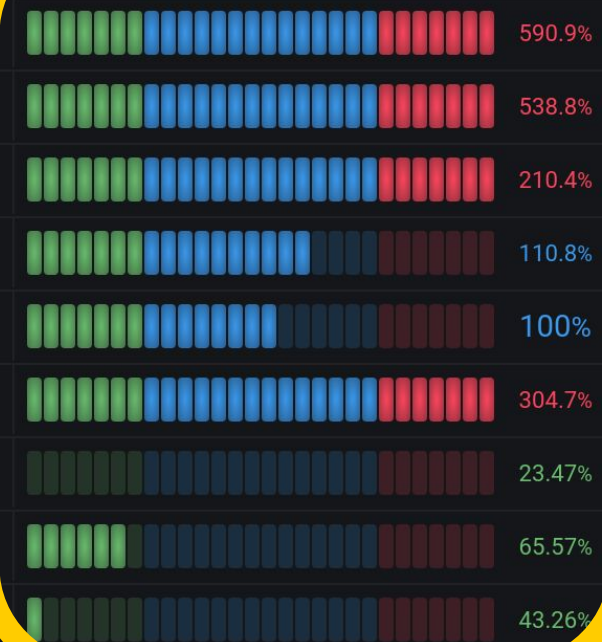
Mean / max(Mean)



Mean / mean(Mean)



Mean / median(Mean)



1. Подход к разработке мониторинга
2. Инструменты для нагрузки и InfluxDB
3. Подготовка окружения разработчика
4. Делаем много баз данных и выбор баз
5. Фильтрация списков тегов
6. Кеш InfluxQL в Variable и отклонения
- 7. Сложные таблицы в Grafana и % успехов**
8. Длительность теста и колонка Time
9. Переход к отчёту по ссылке
10. Демонстрация



Сложные таблицы в Grafana и % успехов

Есть ли JOIN в InfluxQL?

(InfluxDB 1.8 + InfluxQL + Grafana)

InfluxDB Cloud
The elastic time series
platform as a service.

Sign up now!

About the project

Release notes

Contribute to InfluxDB

Contributor license agreement

InfluxDB license

How do I query data across measurements?

Currently, there is no way to perform cross-measurement math or grouping. All data must be under a single measurement to query it together. InfluxDB is not a relational database and mapping data across measurements is not currently a recommended [schema](#). See [GitHub Issue #3552](#) for a discussion of implementing JOIN in InfluxDB.

Does the order of the timestamps matter?

No. Our tests indicate that there is a only a negligible difference between the times it takes InfluxDB to complete the following queries:

```
SELECT ... FROM ... WHERE time > 'timestamp1' AND time < 'timestamp2'
```

Нет, в InfluxQL нет JOIN-ов (InfluxDB 1.8 + InfluxQL + Grafana)

InfluxDB Cloud
The elastic time series
platform as a service.

[Sign up now!](#)

How do I query data across measurements?

Currently, there is no way to perform cross-measurement math or grouping. All data must be under a single measurement to query it together. InfluxDB is not a relational database and mapping data across measurements is not currently a recommended [schema](#). See [GitHub Issue #3552](#) for a discussion of implementing JOIN in InfluxDB.

docs.influxdata.com/influxdb/v1.8/troubleshooting/frequently-asked-questions/#how-do-i-query-data-across-measurements

[Contribute to InfluxDB](#)

[Contributor license agreement](#)

[InfluxDB license](#)

takes InfluxDB to complete the following queries:

```
SELECT ... FROM ... WHERE time > 'timestamp1' AND time < 'timestamp2'
```


Есть ли JOIN во Flux?

(InfluxDB 1.8 + Kapacitor + Flux + ...)

Unlock powerful insights that help
you delight your customers.

Try InfluxDB Enterprise

JoinNode

About the project

Contributing

CLA

License

Release Notes/Changelog

Introduction

Working with Kapacitor

The `join` node joins data from any number of nodes. As each data point is received from a parent node it is paired with the next data points from the other parent nodes with a matching timestamp. Each parent node contributes at most one point to each joined point. A tolerance can be supplied to join points that do not have perfectly aligned timestamps. Any points that fall within the tolerance are joined on the timestamp. If multiple points fall within the same tolerance window than they are joined in the order they arrive.

Aliases are used to prefix all fields from the respective nodes.

The join can be an inner or outer join, see the `JoinNode.Fill` property.

Example: Joining two measurements

In the example below, the `errors` and `requests` streams are joined and transformed to calculate a

Да, JOIN по time давно есть во Flux (InfluxDB 1.8 + Kapacitor + Flux + ...)

Unlock powerful insights that help
you delight your customers.

Try InfluxDB Enterprise

JoinNode

About the project

Contributing

CLA

License

Release Notes/Changelog

Introduction

The `join` node joins data from any number of nodes. As each data point is received from a parent node it is paired with the next data points from the other parent nodes with a matching timestamp. Each parent node contributes at most one point to each joined point. A tolerance can be supplied to join points that do not have perfectly aligned timestamps. Any points that fall within the tolerance are joined on the timestamp. If multiple points fall within the same tolerance window than they are joined in the order they arrive.

Aliases are used to prefix all fields from the respective nodes.

The join can be an inner or outer join, see the `JoinNode.Fill` property.

docs.influxdata.com/kapacitor/v1.5/nodes/join_node/

In the example below, the `errors` and `requests` streams are joined and transformed to calculate a

Да, JOIN по time давно есть во Flux (InfluxDB 1.8 + Караситор + Flux + ...)

Unlock powerful insights that help
you delight your customers.

Try InfluxDB Enterprise

JoinNode

About the project

Contributing

CLA

License

Release Notes/Changelog

Introduction

The `join` node joins data from any number of nodes. As each data point is received from a parent node it is paired with the next data points from the other parent nodes with a matching timestamp. Each parent node contributes at most one point to each joined point. A tolerance can be supplied to join points that do not have perfectly aligned timestamps. Any points that fall within the tolerance are joined on the earliest timestamp. If multiple points fall within the same tolerance window than they are joined when they arrive.

Aliases are used to prefix all fields from the respective nodes.

The join can be an inner or outer join, see the `JoinNode.Fill` property.

docs.influxdata.com/kapacitor/v1.5/nodes/join_node/

Working with Kapacitor

In the example below, the `errors` and `requests` streams are joined and transformed to calculate a

Но Караситор
редко
используют

Есть ли JOIN во Flux без Capacitor? (InfluxDB 1.8 + Flux + Grafana)

Unlock powerful insights
that help you delight your
customers.

Try InfluxDB
Enterprise

Chronograf 1.8 documentation

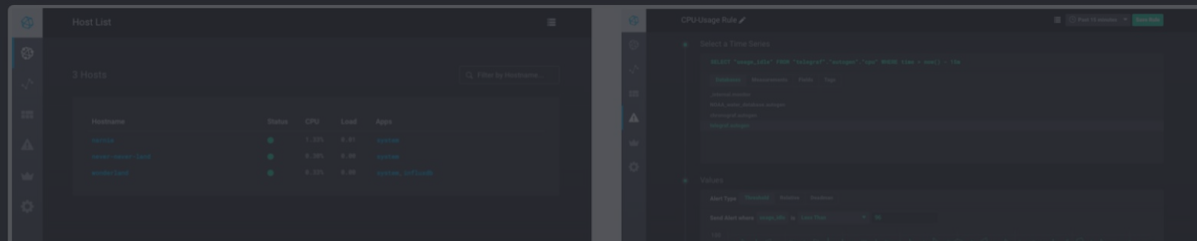
Chronograf is InfluxData's open source web application. Use Chronograf with the other components of the **TICK stack** to visualize your monitoring data and easily create alerting and automation rules.

About the project

Release notes

Contributing

InfluxData Contributor License
Agreement (CLA)



Да, есть JOIN во Flux без Capacitor (InfluxDB 1.8 + Flux + ~~Grafana~~)

Unlock powerful insights
that help you delight your
customers.

Try InfluxDB
Enterprise

Chronograf 1.8 documentation

Chronograf is InfluxData's open source web application. Use Chronograf with the other components of the **TICK stack** to visualize your monitoring data and easily create alerting and automation rules.

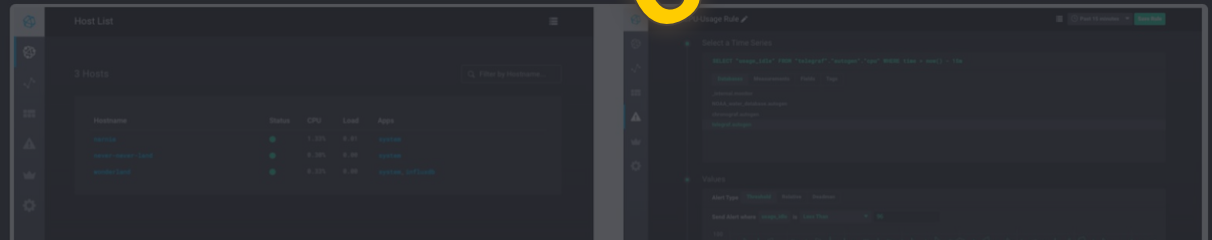
**ЕСТЬ, НО НЕ
С Grafana**

About the project

Release notes

Contributing

InfluxData Contributor License
Agreement (CLA)



Да, есть JOIN во Flux без Kapacitor (InfluxDB 1.8 + Flux + Chronograf)

Unlock powerful insights
that help you delight your
customers.

Try InfluxDB
Enterprise

Chronograf 1.8 documentation

Chronograf is InfluxData's open source web application. Use Chronograf with the other components of the **TICK stack** to visualize your monitoring data and easily create alerting and automation rules.

About the project

Release notes

Contributing

InfluxData Contributor License
Agreement (CLA)

docs.influxdata.com/chronograf/v1.8/



Да, есть JOIN во Flux без Capacitor (InfluxDB 1.8 + Flux + Chronograf)

Unlock powerful insights
that help you delight your
customers.

Try InfluxDB
Enterprise

Chronograf 1.8 documentation

Chronograf is InfluxData's open source web application. Use Chronograf and the other components of the **TICK stack** to visualize your monitoring data and easily create alerting and automation rules.

About the project

Release notes

Contributing

InfluxData Contributor License
Agreement (CLA)

docs.influxdata.com/chronograf/v1.8/

Но Chronograf
редко
используют



Есть ли JOIN во Flux без Chronograf? (InfluxDB + Flux + Grafana)

The image shows a screenshot of the InfluxDB 2.0 documentation website. The top navigation bar includes the InfluxData logo, the text "InfluxDB Docs", a version selector set to "v2.0", and icons for settings and a dark mode toggle. A search bar is located on the left side of the page. Below the search bar is a sidebar menu with the following items: "Get started", "Write data", "Query data", "Process data", "Visualize data", "Monitor & alert", and "Back up & restore data (OSS)". The main content area is a dark blue banner with the text "Get started with InfluxDB 2.0". Below this banner are two buttons: "Start with InfluxDB Cloud" and "Start with InfluxDB OSS (beta)". A red callout box with a white border and a close button (X) is positioned over the "Start with InfluxDB OSS (beta)" button, containing the text "New! Cloud or OSS?".

influxdata | InfluxDB Docs

v2.0

Search v2.0

- Get started
- Write data
- Query data
- Process data
- Visualize data
- Monitor & alert
- Back up & restore data (OSS)

Get started with InfluxDB 2.0

Start with InfluxDB Cloud

Start with InfluxDB OSS (beta)

New! Cloud or OSS?

Да, есть JOIN во Flux без Chronograf (InfluxDB 2.0 + Flux + Grafana)

 *influxdata* | *InfluxDB Docs*

v2.0 ▾



New! Cloud or OSS? ✕

Search v2.0



▪ [Get started](#)

+ [Write data](#)

+ [Query data](#)

+ [Process data](#)

+ [Visualize data](#)

+ [Monitor & alert](#)

+ [Back up & restore data](#)

(oss)

Get started with InfluxDB 2.0

Start with InfluxDB Cloud

v2.docs.influxdata.com/v2.0/ beta

Да, есть JOIN во Flux без Chronograf (InfluxDB 2.0 + Flux + Grafana)

 influxdata | InfluxDB Docs

Search v2.0

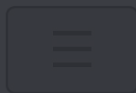
- Get started
- Write data
- Query data
- Process data
- Visualize data
- Monitor & alert
- Back up & restore data

v2.docs.influxdata.com/v2.0/ beta

Но InfluxDB 2.0
редко
используют

Есть ли JOIN для InfluxDB не 2.0? (как-нибудь ещё, по другому)

[Blog](#) [Case Studies](#) [Community](#) [Documentation](#) [GrafanaCon 2020](#) [Tutorials](#)

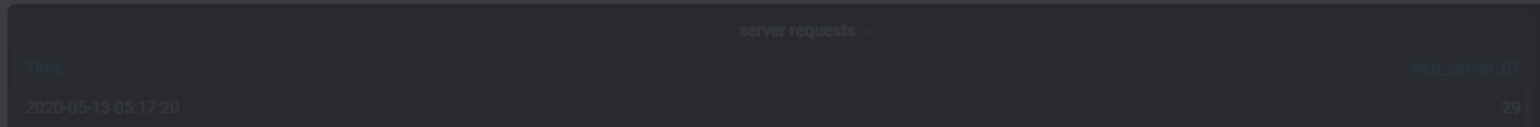


Join by field (outer join)

Use this transformation to join multiple time series from a result set by field.

This transformation is especially useful if you want to combine queries so that you can calculate results from the fields.

In the example below, I have a template query displaying time series data from multiple servers in a table visualization. I can only view the results of one query at a time.



Да, появился JOIN в Grafana 7.0

(InfluxDB 1.8 + InfluxQL + Grafana 7.0)

[Blog](#) [Case Studies](#) [Community](#) [Documentation](#) [GrafanaCon 2020](#) [Tutorials](#)



Join by field (outer join)

Use this transformation to join multiple time series from a result set by field.

This transformation is especially useful if you want to combine queries so that you can calculate results from the fields.

In the example below, I have a template query displaying time series data from multiple servers in a table visualization. I can only view the results of one query at a time.

grafana.com/docs/grafana/latest/panels/transformations/#join-by-field-outer-join

time
2020-05-13 05:17:20

server requests

web server 01

29

Merge есть в **Grafana 5.2+**,
но он требует совпадения:

- тегов из group by (**просто**)
- значения **time** (**сложно**)

grafana.com/docs/grafana/v5.2/features/panels/table_panel/#merge-multiple-queries-per-table

Merge есть в Grafana 5.2+,
но он требует совпадения:

- тегов из group by (просто)
- значения **time** (сложно)

для 2, 3, 4, ... колонок

grafana.com/docs/grafana/v5.2/features/panels/table_panel/#merge-multiple-queries-per-table

MERGE в Grafana 5.2+:

- сохраняет первый time
- 2, 3, ... time уже ключевые

grafana.com/docs/grafana/v5.2/features/panels/table_panel/#merge-multiple-queries-per-table

MERGE в Grafana 5.2+:

- сохраняет первый time
- 2, 3, ... time уже ключевые
 - их удобно сбросить в 0

grafana.com/docs/grafana/v5.2/features/panels/table_panel/#merge-multiple-queries-per-table

Сброс колонки **time** в **0**:

```
SELECT
```

```
  last(A) + first(A) - last(A)
```

```
FROM
```

```
  (SELECT A FROM ... GROUP BY B)
```

```
WHERE $timeFilter
```

```
GROUP BY B
```

Сброс колонки **time** в 0:

```
SELECT
```

```
  last(A) + first(A) - last(A)
```

```
FROM
```

```
  (SELECT A FROM ... GROUP BY B)
```

```
WHERE $timeFilter
```

```
GROUP BY B
```

MERGE позволяет создавать красивые таблицы (Table)

STATISTICS ▾														
Group (level 1)	Group (level 2) ▾	Requests	Total	OK	KO	% KO	Min	50th pct	75th pct	95th pct	99th pct	Max	Mean	Std Dev
-	-	ubuntu-logo-png	12002	0	12002	10%	1	1	2	6	16	89	1	2
-	-	Global Information	120020	84014	36006	30%	1	1	3	9	65	161	2	3
-	-	/(GET)	12002	12002	0	0%	1	2	4	14	132	153	4	6
images	-	/image3-png	12002	12002	0	0%	1	1	3	10	34	74	2	4
images	-	/image2-png	12002	12002	0	0%	1	1	3	9	27	161	2	3
images	-	/image1-png	12002	12002	0	0%	1	1	3	9	26	55	2	3
errorPages	-	/50x-html_(GET)	12002	0	12002	10%	1	1	1	6	26	115	1	2
errorPages	-	/40x-html_(GET)	12002	0	12002	10%	1	1	1	6	14	59	1	2
images	bigImages	/image4-png	12002	12002	0	0%	1	1	3	9	26	66	2	4

Протестируем новый
OUTER JOIN по полю
из **Grafana 7.0**
на примере нашего отчета

Выберем сумму **OK**, KO и Total с группировкой по запускам (run)

▼ A ✎ ↓ ↑ 📄 👁 🗑

FROM	\$archive	gatling	WHERE	request	=	allRequests	AND	status	=	ok	AND	simulation	=~	/^\$simulation\$/	+
SELECT	field (count)	sum ()	alias (OK)	+											
GROUP BY	tag (run)	+													
FORMAT AS	Table	▼													

Выберем сумму ОК, **KO** и Total с группировкой по запускам (run)

▼ D ✎ ↓ ↑ 📄 👁 🗑

FROM	\$archive	gatling	WHERE	request	=	allRequests	AND	status	=	ko	AND	simulation	=~	/^\$simulation\$/	+
SELECT	field (count)	sum ()	alias (KO)	+											
GROUP BY	tag (run)	+													
FORMAT AS	Table	▼													

Выберем сумму ОК, КО и **Total** с группировкой по запускам (run)

▼ E ✎ ↓ ↑ 📄 🗨 🗑

FROM	\$archive	gatling	WHERE	request	=	allRequests	AND	status	=	all	AND	simulation	=~	/^\$simulation\$/	+
SELECT	field (count)	sum ()	alias (Total)	+											
GROUP BY	tag (run)	+													
FORMAT AS	Table	▼													

Выберем сумму ОК, КО и Total тремя разными запросами

FROM	\$archive	gatling	WHERE	request	=	allRequests	AND	status	=	ok	AND	simulation	=~	/^\$simulation\$/ +
SELECT	field (count)	sum ()	alias (OK)	+										
GROUP BY	tag (run)	+												
FORMAT AS	Table	▼												

FROM	\$archive	gatling	WHERE	request	=	allRequests	AND	status	=	ko	AND	simulation	=~	/^\$simulation\$/ +
SELECT	field (count)	sum ()	alias (KO)	+										
GROUP BY	tag (run)	+												
FORMAT AS	Table	▼												

FROM	\$archive	gatling	WHERE	request	=	allRequests	AND	status	=	all	AND	simulation	=~	/^\$simulation\$/ +
SELECT	field (count)	sum ()	alias (Total)	+										
GROUP BY	tag (run)	+												
FORMAT AS	Table	▼												

И объединим их по тегу **run**,
просто перейдя на вкладку Transform

Query 3

Transform 4

Outer join



Field name

run



OUTER JOIN по полю run объединяет OK, KO и Total

run ▾	Time	OK	Time	KO	Time	Total
2020-02-03_15:50	2019-06-16 12:14:59	168024	2019-06-16 12:14:59	72012	2019-06-16 12:14:59	240036
2020-01-31_21:15	2019-06-16 12:14:59	7	2019-06-16 12:14:59	3	2019-06-16 12:14:59	10
2020-01-31_16:20	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-31_15:30	2019-06-16 12:14:59	100723	2019-06-16 12:14:59	43167	2019-06-16 12:14:59	143890
2020-01-30_23:16	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-30_16:16	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:32	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:09	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:05	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:00	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020

OUTER JOIN по полю run объединяет OK, KO и Total

run ▾	Time	OK	Time	KO	Time	Total
2020-02-03_15:50	2019-06-16 12:14:59	168024	2019-06-16 12:14:59	72012	2019-06-16 12:14:59	240036
2020-01-31_21:15	2019-06-16 12:14:59	7	2019-06-16 12:14:59	3	2019-06-16 12:14:59	10
2020-01-31_16:20	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-31_15:30	2019-06-16 12:14:59	100723	2019-06-16 12:14:59	43167	2019-06-16 12:14:59	143890
2020-01-30_23:16	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-30_16:16	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:32	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:09	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:05	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:00	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020

OUTER JOIN по полю run объединяет OK, KO и Total

run ▾	Time	OK	Time	KO	Time	Total
2020-02-03_15:50	2019-06-16 12:14:59	168024	2019-06-16 12:14:59	72012	2019-06-16 12:14:59	240036
2020-01-31_21:15	2019-06-16 12:14:59	7	2019-06-16 12:14:59	3	2019-06-16 12:14:59	10
2020-01-31_16:20	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-31_15:30	2019-06-16 12:14:59	100723	2019-06-16 12:14:59	43167	2019-06-16 12:14:59	143890
2020-01-30_23:16	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-30_16:16	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:32	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:09	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:05	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:00	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020

OUTER JOIN по полю run объединяет OK, KO и Total

run ▾	Time	OK	Time	KO	Time	Total
2020-02-03_15:50	2019-06-16 12:14:59	168024	2019-06-16 12:14:59	72012	2019-06-16 12:14:59	240036
2020-01-31_21:15	2019-06-16 12:14:59	7	2019-06-16 12:14:59	3	2019-06-16 12:14:59	10
2020-01-31_16:20	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-31_15:30	2019-06-16 12:14:59	100723	2019-06-16 12:14:59	43167	2019-06-16 12:14:59	143890
2020-01-30_23:16	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-30_16:16	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:32	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:09	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:05	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:00	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020

OUTER JOIN по полю run объединяет OK, KO и Total: Time сброшен в now()

run ▾	Time	OK	Time	KO	Time	Total
2020-02-03_15:50	2019-06-16 12:14:59	168024	2019-06-16 12:14:59	72012	2019-06-16 12:14:59	240036
2020-01-31_21:15	2019-06-16 12:14:59	7	2019-06-16 12:14:59	3	2019-06-16 12:14:59	10
2020-01-31_16:20	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-31_15:30	2019-06-16 12:14:59	100723	2019-06-16 12:14:59	43167	2019-06-16 12:14:59	143890
2020-01-30_23:16	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-30_16:16	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:32	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:09	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:05	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020
2020-01-29_20:00	2019-06-16 12:14:59	84014	2019-06-16 12:14:59	36006	2019-06-16 12:14:59	120020

Можно скрыть колонку **Time** на вкладке Transform / Organaze fields

The screenshot shows the 'Transform' tab (4) in a data tool. The 'Organize fields' section is active, displaying a list of fields. The 'Time' field is highlighted with a yellow box, and a yellow arrow points to its visibility icon (an eye with a slash through it), which is also highlighted with a yellow box. This indicates that the 'Time' column is being hidden.

Query 3

Transform 4

Outer join

Field name run

Organize fields

Time

run

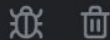
OK

И использовать новую фичу: Transform / **Add field from calculation**

Query 3

Transform 4

▼ Add field from calculation



Mode

Binary operation



Operation

OK



/



Total



Alias

OK / Total

Replace all fields

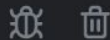


Рассчитаем % успехов с Transform / **Add field from calculation**

Query 3

Transform 4

▼ Add field from calculation



Mode

Binary operation



Operation

OK



/



Total



Alias

OK / Total

Replace all fields

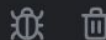


Рассчитаем % провалов с Transform / **Add field from calculation**

Query 3

Transform 4

Add field from calculation



Mode

Binary operation



Operation

KO



/



Total



Alias

KO / Total

Replace all fields



Проценты удобно визуализировать в виде **LCD gauge** выставив **Threshold**

run ▾	OK	KO	Total	OK / Total	KO / Total
2020-02-03_15:50	168024	72012	240036	70.00%	30.00%
2020-01-31_21:15	7	3	10	70.00%	30.00%
2020-01-31_16:20	84014	36006	120020	70.00%	30.00%
2020-01-31_15:30	100723	43167	143890	70.00%	30.00%
2020-01-30_23:16	84014	36006	120020	70.00%	30.00%
2020-01-30_16:16	84014	36006	120020	70.00%	30.00%
2020-01-29_20:32	84014	36006	120020	70.00%	30.00%
2020-01-29_20:09	84014	36006	120020	70.00%	30.00%
2020-01-29_20:05	84014	36006	120020	70.00%	30.00%
2020-01-29_20:02	84014	36006	120020	70.00%	30.00%
2020-01-29 19:58	81993	35135	117128	70.00%	30.00%

Проценты удобно визуализировать в виде **LCD gauge** выставив **Threshold**

run ▾	OK	KO	Total	OK / Total	KO / Total
2020-03-21_23:00	156195	555856	712051	21.94%	78.06%
2020-03-21_22:55	113237	342006	455243	24.87%	75.13%
2020-03-21_22:50	142504	427512	570016	25.00%	75.00%
2020-03-21_22:45	46252	138756	185008	25.00%	75.00%
2020-03-21_22:10	38002	114006	152008	25.00%	75.00%
2020-03-21_22:05	16502	49506	66008	25.00%	75.00%
2020-03-21_21:50	5502	16506	22008	25.00%	75.00%
2020-02-12_21:50	101	303	404	25.00%	75.00%
2020-02-12_16:15	202	606	808	25.00%	75.00%
2020-02-12_16:5	101	303	404	25.00%	75.00%
	101	303	404	25.00%	75.00%

Мы попробовали
OUTER JOIN по полю
из **Grafana 7.0**
и узнали его ограничения:
он меняет все Time на now()

1. Подход к разработке мониторинга
2. Инструменты для нагрузки и InfluxDB
3. Подготовка окружения разработчика
4. Делаем много баз данных и выбор баз
5. Фильтрация списков тегов
6. Кеш InfluxQL в Variable и отклонения
7. Сложные таблицы в Grafana и % успехов
8. Длительность теста и колонка Time
9. Переход к отчёту по ссылке
10. Демонстрация



Длительность теста и колонка Time

В отчете важно видеть:

- **как долго длился тест?**
- **не прервался ли он?**

В отчете важно видеть:

- как долго длился тест?
 - **чтобы посчитать RPS:**
 - Total / Duration
- не прервался ли он?

Длительность: сумма дельт времени

```
SELECT SUM("e") as "Duration" FROM
  (SELECT elapsed("count", 1s) as "e"
    FROM "$archive"."gatling"
    WHERE ("request" = 'allRequests' AND
           "simulation" =~ /^$simulation$/ AND
           "status" = 'all'
          ) AND $timeFilter
    GROUP BY "run")
GROUP BY "run"
```

docs.influxdata.com/influxdb/v1.8/query_language/functions/#elapsed

Запрос не составить в конструкторе Нужна правка текста запроса

Query 1

Transform 1



\$ds



> Query options

MD = auto = 2468

Interval = 3h

Query inspector

▼ B



```
SELECT SUM("e") as "Duration" FROM
(SELECT elapsed("count", 1s) as "e" FROM "$archive"."gatling" WHERE ("request" = 'allRequests' AND "simulation"
=~ /^^$simulation$/ AND "status" = 'all') AND $timeFilter GROUP BY "run")
GROUP BY "run"
```

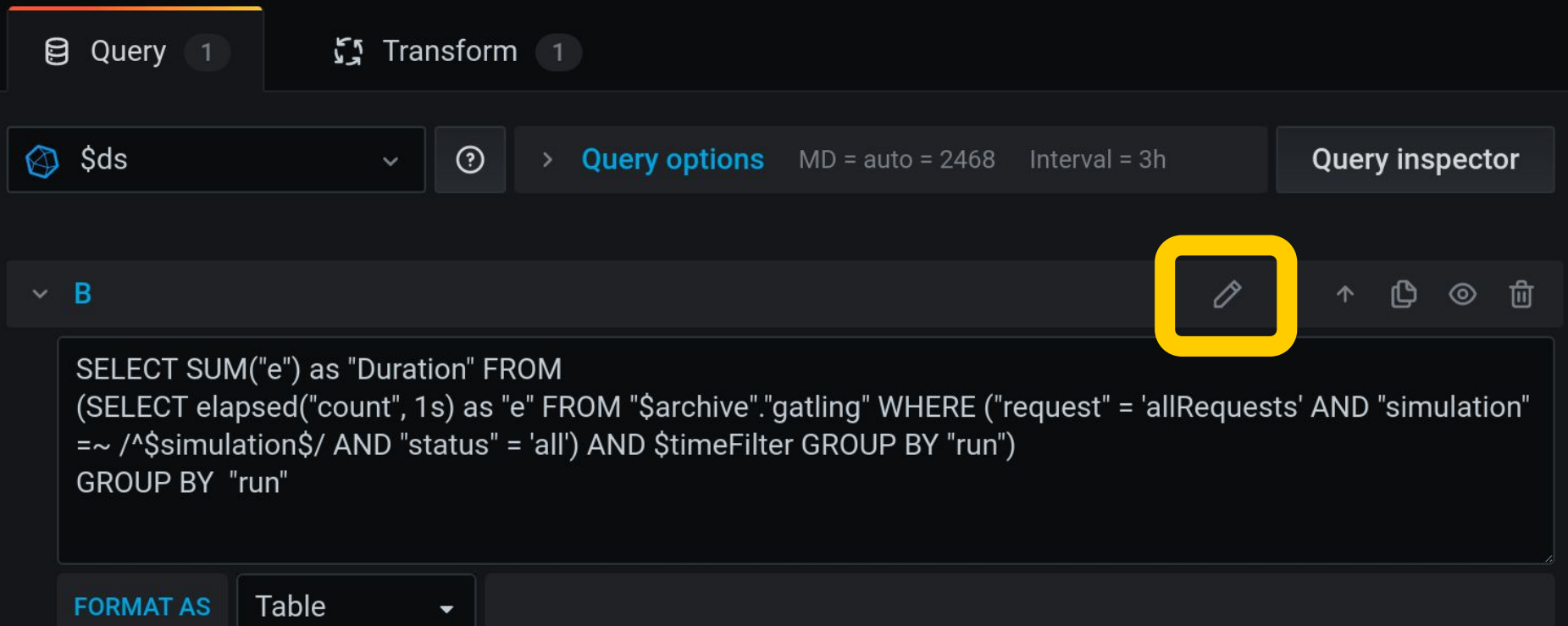
FORMAT AS

Table



Запрос не составить в конструкторе

Нужна **правка** текста запроса



The screenshot shows a query editor interface. At the top, there are tabs for "Query 1" and "Transform 1". Below the tabs, there is a dropdown menu showing "\$ds" and a "Query options" section with "MD = auto = 2468" and "Interval = 3h". A "Query inspector" button is visible on the right. The main area displays a SQL query:

```
SELECT SUM("e") as "Duration" FROM
(SELECT elapsed("count", 1s) as "e" FROM "$archive"."gatling" WHERE ("request" = 'allRequests' AND "simulation"
=~ /^$simulation$/ AND "status" = 'all') AND $timeFilter GROUP BY "run")
GROUP BY "run"
```

At the bottom left, there is a "FORMAT AS" dropdown menu set to "Table". A yellow square highlights the edit icon (a pencil) in the top right corner of the query editor area.

Функция **SUM** возвращает результат в **1970-01-01** с поправкой на пояс

run ▾	Duration	Time
2020-03-21_23:00	99	1970-01-01 03:00:00
2020-03-21_22:55	157	1970-01-01 03:00:00
2020-03-21_22:50	296	1970-01-01 03:00:00
2020-03-21_22:45	99	1970-01-01 03:00:00
2020-03-21_22:10	99	1970-01-01 03:00:00
2020-03-21_22:05	13	1970-01-01 03:00:00
2020-03-21_21:50	54	1970-01-01 03:00:00
2020-02-12_21:50	4	1970-01-01 03:00:00
2020-02-12_16:15	75	1970-01-01 03:00:00

Функция **SUM** возвращает результат в **1970-01-01** с поправкой на пояс

run ▾	Duration	Time
2020-03-21_23:00	99	1970-01-01 03:00:00
2020-03-21_22:55	157	1970-01-01 03:00:00
2020-03-21_22:50	296	1970-01-01 03:00:00
2020-03-21_22:45	99	1970-01-01 03:00:00
2020-03-21_22:10	99	1970-01-01 03:00:00
2020-03-21_22:05	13	1970-01-01 03:00:00
2020-03-21_21:50	54	1970-01-01 03:00:00
2020-02-12_21:50	4	1970-01-01 03:00:00
2020-02-12_16:15	75	1970-01-01 03:00:00

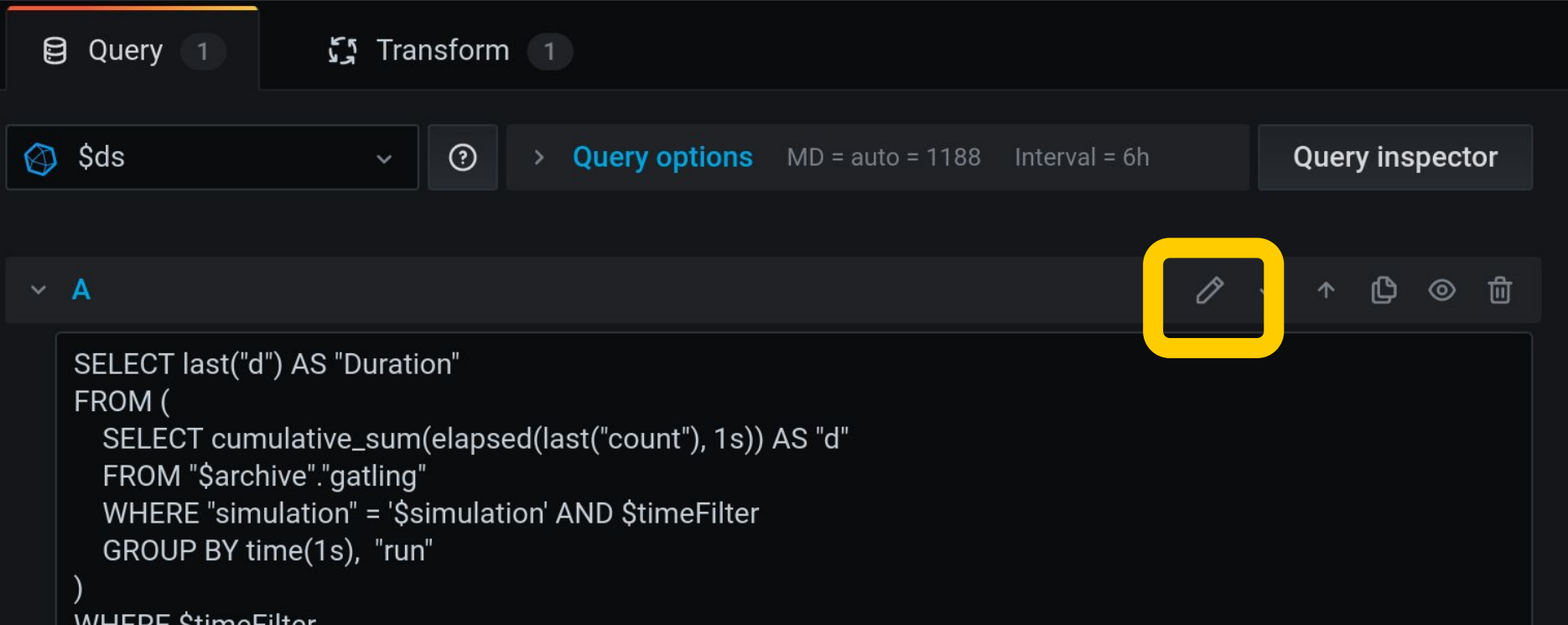
Длительность в конце теста дает не **sum**, а **cumulative_sum**

```
SELECT last("d") AS "Duration"
FROM (
  SELECT cumulative_sum(elapsed(last("count"), 1s)) AS "d"
  FROM "$archive"."gatling"
  WHERE "simulation" = '$simulation' AND $timeFilter
  GROUP BY time(1s), "run"
)
WHERE $timeFilter
GROUP BY "run"
```

[docs.influxdata.com/influxdb/v1.8/query_language/functions/
#cumulative-sum](https://docs.influxdata.com/influxdb/v1.8/query_language/functions/#cumulative-sum)

Запрос не составить в конструкторе

Также нужна правка текста запроса



The screenshot shows a query editor interface with a dark theme. At the top, there are tabs for "Query" (1) and "Transform" (1). Below the tabs, there is a toolbar with a dropdown menu showing "\$ds", a help icon (?), and "Query options" (MD = auto = 1188, Interval = 6h). To the right of the toolbar is a "Query inspector" button. Below the toolbar, there is a dropdown menu showing "A" and a toolbar with an edit icon (pencil), an up arrow, a copy icon, an eye icon, and a trash icon. The edit icon is highlighted with a yellow square. Below the toolbar, there is a text area containing a SQL query:

```
SELECT last("d") AS "Duration"
FROM (
  SELECT cumulative_sum(elapsed(last("count"), 1s)) AS "d"
  FROM "$archive"."gatling"
  WHERE "simulation" = '$simulation' AND $timeFilter
  GROUP BY time(1s), "run"
)
WHERE $timeFilter
```

`cumulative_sum` вернёт результат уже на момент записи последней точки

run ▾	Duration	Test Finish Time
2020-03-21_23:00	99	2020-03-21 23:05:03
2020-03-21_22:55	157	2020-03-21 23:01:22
2020-03-21_22:50	296	2020-03-21 22:55:33
2020-03-21_22:45	99	2020-03-21 22:49:04
2020-03-21_22:10	99	2020-03-21 22:12:33
2020-03-21_22:05	13	2020-03-21 22:08:49
2020-03-21_21:50	54	2020-03-21 21:51:37
2020-02-12_21:50	4	2020-02-12 21:50:42
2020-02-12_16:15	75	2020-02-12 16:18:15

cumulative_sum вернёт результат уже
на момент записи последней точки

run ▾	Duration	Test Finish Time
2020-03-21_23:00	99	2020-03-21 23:05:03
2020-03-21_22:55	157	2020-03-21 23:01:22
2020-03-21_22:50	296	2020-03-21 22:55:33
2020-03-21_22:45	99	2020-03-21 22:49:04
2020-03-21_22:10	99	2020-03-21 22:12:33
2020-03-21_22:05	13	2020-03-21 22:08:49
2020-03-21_21:50	54	2020-03-21 21:51:37
2020-02-12_21:50	4	2020-02-12 21:50:42
2020-02-12_16:15		

`cumulative_sum` вернёт результат уже на момент записи последней точки

run ▾	Duration	Test Finish Time
2020-03-21_23:00	99	2020-03-21 23:05:03
2020-03-21_22:55	157	2020-03-21 23:01:22
2020-03-21_22:50	296	2020-03-21 22:55:33
2020-03-21_22:45	99	2020-03-21 22:49:04
2020-03-21_22:10	99	2020-03-21 22:12:33
2020-03-21_22:05	13	2020-03-21 22:08:49
2020-02-12_21:50	4	2020-02-12 21:50:42

Группировать **cumulative_sum** по каждой секунде накладно

```
SELECT last("d") AS "Duration"
FROM (
  SELECT cumulative_sum(elapsed(last("count"), 1s)) AS "d"
  FROM "$archive"."gatling"
  WHERE "simulation" = '$simulation' AND $timeFilter
  GROUP BY time(1s), "run"
)
WHERE $timeFilter
GROUP BY "run"
```

docs.influxdata.com/influxdb/v1.8/query_language/functions/#cumulative-sum

Можно считать **cumulative_sum** с ускорением и потерей точности

```
SELECT last("d") AS "Duration"
FROM (
  SELECT cumulative_sum(elapsed(last("count"), 1s)) AS "d"
  FROM "$archive"."gatling"
  WHERE "simulation" = '$simulation' AND $timeFilter
  GROUP BY time(10s), "run"
)
WHERE $timeFilter
GROUP BY "run"
```

[docs.influxdata.com/influxdb/v1.8/query_language/functions/
#cumulative-sum](https://docs.influxdata.com/influxdb/v1.8/query_language/functions/#cumulative-sum)

Мы получим неточные результаты, но значительно быстрее

Test Duration with Finish Time		
run ▾	Duration	Test Finish Time
2020-03-21_23:00	100	2020-03-21 23:05:00
2020-03-21_22:55	160	2020-03-21 23:01:20
2020-03-21_22:50	300	2020-03-21 22:55:30
2020-03-21_22:45	100	2020-03-21 22:49:00
2020-03-21_22:10	100	2020-03-21 22:12:30
2020-03-21_22:05	10	2020-03-21 22:08:40
2020-03-21_21:50	50	2020-03-21 21:51:30
2020-02-12_21:50	10	2020-02-12 21:50:40
2020-02-12_16:15	70	2020-02-12 16:18:10

Получим **округленные** результаты, округленные до размера группы

run	Duration	Finish Time
2020-03-21_23:00	100	20-03-21 23:05:00
2020-03-21_22:55	160	20-03-21 23:01:20
2020-03-21_22:50	300	20-03-21 22:55:30
2020-03-21_22:45	100	20-03-21 22:49:00
2020-03-21_22:10	100	20-03-21 22:12:30
2020-03-21_22:05	10	20-03-21 22:08:40
2020-03-21_21:50	50	20-03-21 21:51:30
2020-02-12_21:50	10	20-02-12 21:50:40
2020-02-12_16:15	70	20-02-12 16:18:10

Получим результаты с потерей тестов короче, чем размер группы

Test Duration with Finish Time (precise)		
run ▾	Duration	Test Finish Time
2020-03-21_23:00	99	2020-03-21 23:05:03
2020-03-21_22:55	157	2020-03-21 23:01:22
2020-03-21_22:50	296	2020-03-21 22:55:33
2020-03-21_22:45	99	2020-03-21 22:49:04
2020-03-21_22:10	99	2020-03-21 22:12:33
2020-03-21_22:05	13	2020-03-21 22:08:49
2020-03-21_21:50	54	2020-03-21 21:51:37
2020-02-12_21:50	4	2020-02-12 21:50:42
2020-02-12_16:15	75	2020-02-12 16:18:15
2020-02-12_16:5	4	2020-02-12 16:06:57
	4	2020-02-12 16:01:16

Test Duration with Finish Time		
run ▾	Duration	Test Finish Time
2020-03-21_23:00	100	2020-03-21 23:05:00
2020-03-21_22:55	160	2020-03-21 23:01:20
2020-03-21_22:50	300	2020-03-21 22:55:30
2020-03-21_22:45	100	2020-03-21 22:49:00
2020-03-21_22:10	100	2020-03-21 22:12:30
2020-03-21_22:05	10	2020-03-21 22:08:40
2020-03-21_21:50	50	2020-03-21 21:51:30
2020-02-12_21:50	10	2020-02-12 21:50:40
2020-02-12_16:15	70	2020-02-12 16:18:10


Но в момент завершения теста

Test Duration with Finish Time (precise)		
run ▾	Duration	Test Finish Time
2020-03-21_23:00	99	2020-03-21 23:05:03
2020-03-21_22:55	157	2020-03-21 23:01:22
2020-03-21_22:50	296	2020-03-21 22:55:33
2020-03-21_22:45	99	2020-03-21 22:49:04
2020-03-21_22:10	99	2020-03-21 22:12:33
2020-03-21_22:05	13	2020-03-21 22:08:49
2020-03-21_21:50	54	2020-03-21 21:51:37
2020-02-12_21:50	4	2020-02-12 21:50:42
2020-02-12_16:15	75	2020-02-12 16:18:15
2020-02-12_16:5	4	2020-02-12 16:06:57
	4	2020-02-12 16:01:16

Test Duration with Finish Time		
run ▾	Duration	Test Finish Time
2020-03-21_23:00	100	2020-03-21 23:05:00
2020-03-21_22:55	160	2020-03-21 23:01:20
2020-03-21_22:50	300	2020-03-21 22:55:30
2020-03-21_22:45	100	2020-03-21 22:49:00
2020-03-21_22:10	100	2020-03-21 22:12:30
2020-03-21_22:05	10	2020-03-21 22:08:40
2020-03-21_21:50	50	2020-03-21 21:51:30
2020-02-12_21:50	10	2020-02-12 21:50:40
2020-02-12_16:15	70	2020-02-12 16:18:10

При OUTER JOIN колонок Duration и Total потеряется колонка Time

Test Duration with Finish Time join Total

run 	Time	Duration	Time	Total
2020-03-21_23:00	2019-06-16 13:46:45	99	2019-06-16 13:46:45	712051
2020-03-21_22:55	2019-06-16 13:46:45	157	2019-06-16 13:46:45	455243
2020-03-21_22:50	2019-06-16 13:46:45	296	2019-06-16 13:46:45	570016
2020-03-21_22:45	2019-06-16 13:46:45	99	2019-06-16 13:46:45	185008
2020-03-21_22:10	2019-06-16 13:46:45	99	2019-06-16 13:46:45	152008
2020-03-21_22:05	2019-06-16 13:46:45	13	2019-06-16 13:46:45	66008
2020-03-21_21:50	2019-06-16 13:46:45	54	2019-06-16 13:46:45	22008

Но можно будет легко посчитать среднее значение **RPS** по тесту

Query 5

Transform 5

▼ Add field from calculation

Mode

Binary operation



Operation

Total



/



Duration






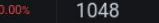







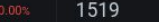




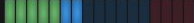


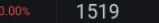







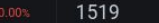







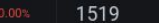







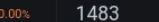




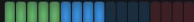







Alias

RPS

Replace all fields



И получить таблицу для выбора проблемных тестов

run ▾	OK	KO	Total	Mean	Duration	Mean / median(Mean)	Duration / median(Duration)	OK / Total	KO / Total	RPS
2020-02-03_15:50	168024	72012	240036	66	229	 3115%	 297.4%	 70.00%	 30.00%	1048
2020-01-31_21:15	7	3	10	25	1	 1172%	 1.299%	 70.00%	 30.00%	10
2020-01-31_16:20	84014	36006	120020	2.0	79	 96.00%	 102.6%	 70.00%	 30.00%	1519
2020-01-31_15:30	100723	43167	143890	1.9	238	 87.33%	 309.1%	 70.00%	 30.00%	605
2020-01-30_23:16	84014	36006	120020	1.9	79	 87.94%	 102.6%	 70.00%	 30.00%	1519
2020-01-30_16:16	84014	36006	120020	2.2	79	 104.1%	 102.6%	 70.00%	 30.00%	1519
2020-01-29_20:32	84014	36006	120020	1.9	79	 90.87%	 102.6%	 70.00%	 30.00%	1519
2020-01-29_20:09	84014	36006	120020	2	79	 93.80%	 102.6%	 70.00%	 30.00%	1519
2020-01-29_20:05	84014	36006	120020	2	79	 93.80%	 102.6%	 70.00%	 30.00%	1519
2020-01-29_20:02	84014	36006	120020	2.3	79	 105.5%	 102.6%	 70.00%	 30.00%	1519
2020-01-29_19:58	81993	35135	117128	2.6	79	 122.9%	 102.6%	 70.00%	 30.00%	1483
2020-01-29_18:19	81958	35129	117087	2.3	79	 109.0%	 102.6%	 70.00%	 30.00%	1482
2020-01-29_18:16	73525	31523	105048	2.1	75	 100.1%	 97.40%	 69.99%	 30.01%	1401
2020-01-29_18:08	73561	31529	105090	2.6	75	 121.9%	 97.40%	 70.00%	 30.00%	1401

В которой хотелось бы иметь ссылки для перехода к детальным отчетам

run	OK	KO	Total	Mean	Duration	Mean / median(Mean)	Duration / median(Duration)	OK / Total	KO / Total	RPS
2020-02-03_15:50	168024	72012	240036	66	229	3115%	297.4%	70.00%	30.00%	1048
2020-01-31_21:15	7	3	10	25	1	1172%	1.299%	70.00%	30.00%	10
2020-01-31_16:20	84014	36006	120020	2.0	79	96.00%	102.6%	70.00%	30.00%	1519
2020-01-31_15:30	100723	43167	143890	1.9	238	87.33%	309.1%	70.00%	30.00%	605
2020-01-30_23:16	84014	36006	120020	1.9	79	87.94%	102.6%	70.00%	30.00%	1519
2020-01-30_16:16	84014	36006	120020	2.2	79	104.1%	102.6%	70.00%	30.00%	1519
2020-01-29_20:32	84014	36006	120020	1.9	79	90.87%	102.6%	70.00%	30.00%	1519
2020-01-29_20:09	84014	36006	120020	2	79	93.80%	102.6%	70.00%	30.00%	1519
2020-01-29_20:05	84014	36006	120020	2	79	93.80%	102.6%	70.00%	30.00%	1519
2020-01-29_20:02	84014	36006	120020	2.3	79	105.5%	102.6%	70.00%	30.00%	1519
2020-01-29_19:58	81993	35135	117128	2.6	79	122.9%	102.6%	70.00%	30.00%	1483
2020-01-29_18:19	81958	35129	117087	2.3	79	109.0%	102.6%	70.00%	30.00%	1482
2020-01-29_18:16	73525	31523	105048	2.1	75	100.1%	97.40%	69.99%	30.01%	1401
2020-01-29_18:08	73561	31529	105090	2.6	75	121.9%	97.40%	70.00%	30.00%	1401

1. Подход к разработке мониторинга
2. Инструменты для нагрузки и InfluxDB
3. Подготовка окружения разработчика
4. Делаем много баз данных и выбор баз
5. Фильтрация списков тегов
6. Кеш InfluxQL в Variable и отклонения
7. Сложные таблицы в Grafana и % успехов
8. Длительность теста и колонка Time
9. **Переход к отчёту по ссылке**
10. Демонстрация



**Переход к
детальному отчёту
по ссылке**

В Grafana можно осуществлять переход на указанное время **from** / **to**

Controlling time range using URL

Time range of a dashboard can be controlled by providing following query parameters in dashboard URL:

- `from` - defines lower limit of the time range, specified in ms epoch
- `to` - defines upper limit of the time range, specified in ms epoch
- `time` and `time.window` - defines a time range from `time-time.window/2` to `time+time.window/2`.

Both params should be specified in ms. For example `?time=1500000000000&time.window=10000` will result in 10s time range from 1499999995000 to 1500000005000

grafana.com/docs/grafana/latest/reference/timerange/#controlling-time-range-using-url

В Grafana можно осуществлять переход на указанное время from / to

Controlling time range using URL

Time range of a dashboard can be controlled by providing following query parameters in dashboard URL.

- `from` - defines lower limit of the time range, specified in ms epoch
- `to` - defines upper limit of the time range, specified in ms epoch
- `time` and `time.window` - defines a time range from `time-time.window/2` to `time+time.window/2`.

Both params should be specified in ms. For example `?time=1500000000000&time.window=10000` will result in 10s time range from 1499999995000 to 1500000005000

Мы знаем
только: to

grafana.com/docs/grafana/latest/reference/timerange/#controlling-time-range-using-url

В Grafana можно осуществлять переход на время $time \pm time.window/2$

Controlling time range using URL

Time range of a dashboard can be controlled by providing following query parameters in dashboard URL:

- `from` - defines lower limit of the time range, specified in ms epoch
- `to` - defines upper limit of the time range, specified in ms epoch
- `time` and `time.window` - defines a time range from `time-time.window/2` to `time+time.window/2`.

Both params should be specified in ms. For example `?time=1500000000000&time.window=10000` will result in 10s time range from 1499999995000 to 1500000005000

grafana.com/docs/grafana/latest/reference/timerange/#controlling-time-range-using-url

В Grafana можно осуществлять переход на время $time \pm time.window/2$

Controlling time range using URL

Time range of a dashboard can be controlled by providing following query parameters in dashboard URL:

- `from` - defines lower limit of the time range, specified in ms epoch
- `to` - defines upper limit of the time range, specified in ms epoch
- `time` and `time.window` - defines a time range from `time-time.window/2` to `time+time.window/2`

Both params should be specified in ms. For example `?time=1500000000000&time.window=10000` will

result in 10s time range from 1499999995000 to 1500000005000

Мы
знаем
Всё!!!

grafana.com/docs/grafana/latest/reference/timerange/#controlling-time-range-using-url

Нужна **Duration*2** (для **time.window/2**) в конце теста (для **time**)

Controlling time range using URL

Time range of a dashboard can be controlled by providing following query parameters in dashboard URL:

- `from` - defines lower limit of the time range, specified in ms epoch

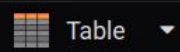
• `time` and `time.window` - defines a time range from `time-time.window/2` to `time+time.window/2`.

Both params should be specified in ms. For example `?time=1500000000000&time.window=10000` will result in 10s time range from 1499999995000 to 1500000005000

grafana.com/docs/grafana/latest/reference/timerange/#controlling-time-range-using-url

И механизм превращения ячеек таблицы в гиперссылки на отчеты

Visualization



Table

Column Styles

Options

Apply to columns named

Time

Column Header

Start Time Link

Render value as link



Type

Type

String

Sanitize HTML



Preserve Formatting



Value Mappings

Type

Value to text

+

Link

Url

`/d/gatling/gatling-report?time=${_cell}&time.window=${_cell_2}000&var-g=${_cell_3}s&var-duration_1=${duration_1}&var...`

Tooltip



Open in new tab



И механизм превращения ячеек таблицы в гиперссылки на отчеты

Panel

Column Styles

Options

Name pattern

Time

Column Header

Time Link

Render value as link



Link

Url



/d/gatling/gatling-report?time=\${__cell}&time.window=\${__cell_2}000...

Tooltip



Open in new tab



ЕСТЬ В
Grafana
7.0

И механизм превращения ячеек таблицы в гиперссылки на отчеты

Panel

Column Styles

Options

Name pattern

Time

Column Header

Time Link

Render value as link



Link

Url

Tooltip

Open in new tab



Сломан в
Grafana
7.0

MERGE в Grafana 5.2+:

- **сохраняет первый time**
- 2, 3, ... time уже ключевые
 - их удобно сбросить в 0

grafana.com/docs/grafana/v5.2/features/panels/table_panel/#merge-multiple-queries-per-table

Но в старых версиях Grafana можно составить ссылку для ячейки

```
/d/gatling/gatling-report?  
time=${__cell}&  
time.window=${__cell_2}&  
var-run=${__cell_1}&  
var-simulation=${simulation}&  
var-loadstation=All&
```

...

А новой версии Grafana 7.0 можно открывать доски созданные ранее

```
#!/bin/sh -x
```

```
docker pull grafana/grafana:6.7.2
```

```
DIR="$(pwd)"
```

```
ID=$(id -u)
```

```
docker run --name=grafana672 \  
  --network=test --user $ID -p 3672:3000 \  
  -v $DIR/grafana.ini:/etc/grafana/grafana.ini \  
  -v $DIR/provisioning:/etc/grafana/provisioning \  
  grafana/grafana:6.7.2
```

у нас же
Docker!!

А новой версии Grafana 7.0 можно открывать доски созданные ранее

```
#!/bin/sh -x
```

```
docker pull grafana/grafana:6.7.2
```

```
DIR="$(pwd)"
```

```
ID=$(id -u)
```

```
docker run --name=grafana672 \  
  --network=test --user $ID -p 3672:3000 \  
  -v $DIR/grafana.ini:/etc/grafana/grafana.ini \  
  -v $DIR/provisioning:/etc/grafana/provisioning \  
  grafana/grafana:6.7.2
```

Используем
Grafana
6.7.2

Обход потери функциональности за счёт обратной совместимости

1. Запустим в Docker **Grafana 6.7.2**
2. Создадим таблицу в **Grafana 6.7.2**
3. Выгрузим доску в JSON
4. Запустим в Docker **Grafana 7.0**
5. Импортируем доску в **Grafana 7.0**
6. Доработаем доску не меняя старое

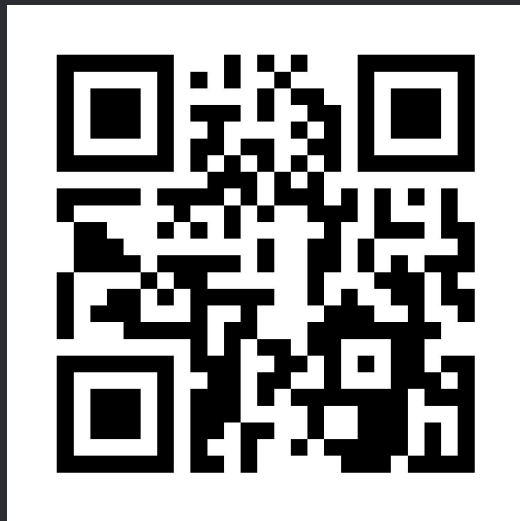
1. Подход к разработке мониторинга
2. Инструменты для нагрузки и InfluxDB
3. Подготовка окружения разработчика
4. Делаем много баз данных и выбор баз
5. Фильтрация списков тегов
6. Кеш InfluxQL в Variable и отклонения
7. Сложные таблицы в Grafana и % успехов
8. Длительность теста и колонка Time
9. Переход к отчёту по ссылке

10. Демонстрация



Демонстрация

Демо-стенд проекта



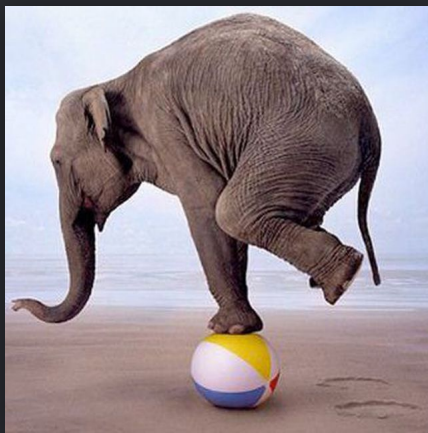
<http://84.201.161.113:3000>

Исходные коды проекта



[github.com/polarnik/
gatling-grafana-dashboard](https://github.com/polarnik/gatling-grafana-dashboard)

Сообщество @qa_load



Смирнов Вячеслав

owasp@yandex.ru

 [@polarnik](#)

 [@qa_load](#)

