# Learning Objectives

The learning objectives in the tutorial are:

- Installing a monitoring stack on the cloud server
- Understand how the components of the monitoring stack work together/are interlinked
- Being able to create panels with plots to display data collected by the monitoring stack

## Tools

- InfluxDB, Telegraf
- Grafana
- Rocky Linux 9.6 server
- an editor (vim, nano,...)
- bash
- browser

## Prequisites

- Working ssh access to the frontend node (login)
- Ports 8007 and 8009 open in firewall
- proxy setup for the worker nodes to allow internet access/downloads from them (Exercise 5+)

# Contents

**Installing InfluxDB 1: Tutorial (20 min)** **2**
1.1 Install the packages . . . 2
1.2 Configuring InfluxDB . . . 4
1.3 Change port of InfluxDB . . . 5

**Installing Telegraf 2: Tutorial (20 min)** **6**
2.1 Install package . . . 6
2.2 Configure telegraf . . . 6
2.3 Smoke test for Telegraf-InfluxDB connection . . . 8

**Installing Grafana and setting up a simple dashboard 3: Tutorial (35 min)** **9**
3.1 Installing Grafana package . . . 9
3.2 First admin login . . . 10
3.3 Create datasource . . . 10
3.4 Create simple dashboard . . . 13

**Optional: Explore data and create own queries 4: Tutorial (0 min)** **16**

**Optional: Install another Telegraf Plugin 5: Tutorial (0 min)** **18**

**Optional: Install telegraf on the worker nodes 6: Tutorial (0 min)** **18**

**Optional: Setup Database and Grafana with specific users 7: Tutorial (0 min)** **18**

**Optional: Central setup for telegraf 8: Tutorial (0 min)** **19**

### Data/info requirements

The following information will be needed to be collected and used throughout the different tasks. Whenever this appears in text you may have to replace it in the text. Esp. in configurations, queries etc.

- $< influxip >$ - IP of the InfluxDB server to be used later for connecting nodes to frontend/cluster-manager: not the floating ip
- $< influxport >$ - Port of InfluxDB: the port the InfluxDB will listen to
- $< org >$ - Organization Name - the organization of the database within InfluxDB, Can be anything like "gwdg" or "uni"
- $< bucket >$ - Bucket Name - the name of the database within InfluxDB
- $< influxuser >$ - the main user for the Bucket
- $< influxpassword >$ Password - password of the user for the bucket
- $< token >$ Token - the access token to access the bucket
- $< grafanaadminpass >$ - the password for the admin access of the grafana server (application)
- $< frontendfloatingip >$ - the floating ip of the frontend server/cluster manager, needed to conntect via ssh

The term bucket will be explained a little bit more later.
It should be ensured that the data listed above is stored somewhere - e.g. a pad or in a local editor - for easy access.

The $< frontendfloatingip >$ can be found in the cloud administration tool cloud.gwdg.de

## Installing InfluxDB 1: Tutorial (20 min)

Goal of this task is to set up the InfluxDB v2x, which is the foundation for the rest of the tutorial. After this the InfluxDB should be running and the following information is available for the further tasks:

### 1.1 Install the packages

The database for storing the metrics provided by the different agent on the systems to monitor has to be set up first. This will be done on the **frontend server**.

- Login to the server via ssh to $< frontendfloatingip >$ .
- `$ ifconfig`

- note down the *inet* ip adress from ens3 as $< influxip >$, e.g. 10.254.1.26. Note that this is NOT the floating IP of the server.
- note down "8086" as $< influxport >$ (will be changed later - just in case)

The standard Redhat/Centos package manager **yum** is used to install the InfluxDB. The repository is not yet in the repository list of **yum** and has to be added. Copy the following bash code block to the bash and execute it:

```
cat <<EOF | sudo tee /etc/yum.repos.d/influxdb.repo
[influxdb]
name = InfluxDB Repository - RHEL \$releasever
baseurl = https://repos.influxdata.com/rhel/\$releasever/\$basearch/stable
enabled = 1
gpgcheck = 1
gpgkey = https://repos.influxdata.com/influxdb.key
EOF
```

The InfluxDB repository should have been added. Now the InfluxDB and its according commandline tools can be installed:

- `$ sudo yum install influxdb2 influxdb2-cli --nogpgcheck`
- confirm the installation of the two packages

In order to set the InfluxDB2 up the according service has to be started:
`$ sudo systemctl start influxdb`

This gives no feedback. In order to check if the system is running, the status can be checked via
`$ sudo systemctl status influxdb`

The output should be similar to this and should not contain errors or the info that the service did not start:

```
influxdb.service - InfluxDB is an open-source, distributed, time series database
     Loaded: loaded (/usr/lib/systemd/system/influxdb.service; enabled; preset: disabled)
     Active: active (running) since Tue 2025-10-07 14:26:46 CEST; 8s ago
       Docs: https://docs.influxdata.com/influxdb/
    Process: 67804 ExecStart=/usr/lib/influxdb/scripts/influxd-systemd-start.sh (code=exited,
        ↪ status=0/SUCCESS)
   Main PID: 67805 (influxd)
      Tasks: 14 (limit: 48900)
     Memory: 55.9M
        CPU: 586ms
     CGroup: /system.slice/influxdb.service
             67805 /usr/bin/influxd

Okt 07 14:26:45 cluster-manager.novalocal influxd-systemd-start.sh[67805]: ts=2025-10-07T12:26:45.707359Z
    ↪ lvl=info msg="Open store (start)" log_id=0zRf37b0000 service=storage-engine service=store
    ↪ op_name=tsdb_open op_event=start
Okt 07 14:26:45 cluster-manager.novalocal influxd-systemd-start.sh[67805]: ts=2025-10-07T12:26:45.707440Z
    ↪ lvl=info msg="Open store (end)" log_id=0zRf37b0000 service=storage-engine service=store
    ↪ op_name=tsdb_open op_event=end op_elapsed=0.098ms
Okt 07 14:26:45 cluster-manager.novalocal influxd-systemd-start.sh[67805]: ts=2025-10-07T12:26:45.707494Z
    ↪ lvl=info msg="Starting retention policy enforcement service" log_id=0zRf37b0000 service=retention
    ↪ check_interval=30m
Okt 07 14:26:45 cluster-manager.novalocal influxd-systemd-start.sh[67805]: ts=2025-10-07T12:26:45.707510Z
    ↪ lvl=info msg="Starting precreation service" log_id=0zRf37b0000 service=shard-precreation
    ↪ check_interval=10m advance_period=30m
Okt 07 14:26:45 cluster-manager.novalocal influxd-systemd-start.sh[67805]: ts=2025-10-07T12:26:45.708274Z
    ↪ lvl=info msg="Starting query controller" log_id=0zRf37b0000 service=storage-reads
    ↪ concurrency_quota=1024 initial_memory_bytes_quota_per_query=9223372036854775807
    ↪ memory_bytes_quota_per_query=9223372036854775807 max_memory_bytes=0 queue_size=1024
```

```
Okt 07 14:26:45 cluster-manager.novalocal influxd-systemd-start.sh[67805]: ts=2025-10-07T12:26:45.709792Z
    ↪ lvl=info msg="Configuring InfluxQL statement executor (zeros indicate unlimited)."
    ↪ log_id=0zRf37b0000 max_select_point=0 max_select_series=0 max_select_buckets=0
Okt 07 14:26:45 cluster-manager.novalocal influxd-systemd-start.sh[67805]: ts=2025-10-07T12:26:45.733651Z
    ↪ lvl=info msg=Starting log_id=0zRf37b0000 service=telemetry interval=8h
Okt 07 14:26:45 cluster-manager.novalocal influxd-systemd-start.sh[67805]: ts=2025-10-07T12:26:45.733832Z
    ↪ lvl=info msg=Listening log_id=0zRf37b0000 service=tcp-listener transport=http addr=:8086 port=8086
Okt 07 14:26:46 cluster-manager.novalocal influxd-systemd-start.sh[67804]: InfluxDB started
Okt 07 14:26:46 cluster-manager.novalocal systemd[1]: Started InfluxDB is an open-source, distributed,
    ↪ time series database.
```

This command may not return to the commandline. Should this be the case the command can be interrupted savely using key-combination "CTRL-C".

To ensure that this service will be started after a boot it needs to be enabled permanentely:
`$ sudo systemctl enable influxdb`

### Hints

- The reason to use the option *--nogpgcheck* is an issue with the gpg-keys that confirm the identity of the binary packages that are installed via yum. Usually these *fingerprints* should be adapted by the package provider when building new versions, but this seems to be not the case. It is usually not recommended to install packages that fail to install because of gpg-fingerprints issue are just installed with this option. The repositories and the situation should be checked before doing so.
- If a service does not start and the status from systemctl is not helpful one should check the log files: e.g. `$ journalctl -eu influxdb`

## 1.2 Configuring InfluxDB

When the InfluxDB service is installed and started it can be configured via the according commandline tools. To set up an initial database a username, the password for this user and a bucket name - the database name - has to be defined. Then the following command can be executed to set up InfluxDB (the values should be replaced by the chosen ones). The general form of the setup command is:
`$ influx setup --username <influx user> --password <influxpassword> --bucket <bucket>`

It has to be ensured that the values are noted down and are accessible for later user.

The setup process will ask for $< org >$ - enter it and write it down. You are free to select a name here. This is just a way to organize databases in InfluxDB. E.g. you have multiple sides collecting the same data. The data is stored in a bucket called "clustermetrics". Instead of having "clustermetrics-side1" and clustermetrics-side2" orgs give the option to create an hierachy like side1 -> clustermetrics, side2 -> clustermetrics. It makes tooling easier and access more consistent.

Select “0” for the retention time.

An example setup command execution:

```
influx setup --username hpcuser --password hpcsa_user --bucket hpcsa
? Please type your primary organization name gwdg
? Please type your retention period in hours, or 0 for infinite 0 0
? Setup with these parameters? --- confirm with 'y'
```

The output should be similar to this:

```
> Welcome to InfluxDB 2.0!
? Please type your primary organization name gwdg
? Please type your retention period in hours, or 0 for infinite 0
? Setup with these parameters?
```

```
  Username: hpcuser
  Organization: gwdg
  Bucket: hpcsa
  Retention Period: infinite
 Yes
User Organization Bucket
hpcuser gwdg  hpcsa
```

Two tasks are performed by this command

- the initial database for the organization with the user to access it is created. The database is called bucket in InfluxDB v2x as it is a combination of the database and the retention policy.
- a profile for accessing the freshly created database is created for the user. This allows easy access w/o the need to provide the access token to the DB manually every time the user wants to utilize the *influx* command to access a database. The file created is *.influxdbv2/configs* in the users home.

Now it is possible to check the authentication information stored in the DB:

```
$ influx auth list
```

```
ID   Description Token            User Name User ID   Permissions
0ac1bf2d2a40b000 hpcuser's Token slgaGixZXXX
```

**Important!**
The `$ tee` command with the *-a* option appends a line to an existing file. If the given command is executed multiple times, e.g. to configure another port for the *http-bind-address* the config file will end up with having multiple lines defining that option. When restarting the InfluxDB service will fail due to a malformed config while giving a non explainatory log output. In that case the wrong *http-bind-address* lines in the config file */etc/influxdb/config.toml* have to be removed.

To check if the InfluxDB has been restarted with the correct port:

```
$ sudo systemctl status influxdb | grep port
```

The output should show that InfluxDB uses port 8009 now:

```
Feb 16 21:58:42 worker.novalocal influxd-systemd-start.sh[11730]: ts=2023-02-16T20:58:42.601641Z lvl=info
    ↪ msg=Listening log\_id=0g30CKUG000 service=tcp-listener transport=http addr=:8009 port=8009
```

The $< influxport >$ value in the notes has to be modified to be "8009", too.

## Installing Telegraf 2: Tutorial (20 min)

In order to fill the database at least one node agent has to run in order provide metrics to the InfluxDB. This will be done in this tutorial.

### 2.1 Install package

The initial Telegraf agent will be installed on the frontend as this is the only machine with access to the internet which is required for an easy download and install of the Telegraf package.

Telegraf is part of the InfluxDB repository and needs the same repository setup as InfluxDB. Therefore, Telegraf can just be installed as InfluxDB repos has already been added to the repository list of the frontend server.
`$ sudo yum install telegraf --nogpgcheck`

The service is now installed but not started yet. Telegraf will not run with the default configuration - it has to be modified first.

### 2.2 Configure telegraf

In order to run Telegraf it has to be configured properly. At least one input and one output plugin has to be configured. The standard input plugin is configured, but the output is not yet done.

The information to be provided in the config in the following steps are the $< influxip >$, $< influxport >$, $< bucket >$, $< org >$ and $< token >$ from the InfluxDB setup. This information is used by the Telegraf agent to contact InfluxDB and access the database (bucket) that has been defined.

#### Steps

1. open the file */etc/telegraf/telegraf.conf* with an editor using sudo (e.g sudo nano)
   - if nano is not installed it must be installed via `$ sudo yum install nano`
2. search for the section [[outputs.influxdb_v2]]

3. uncomment (remove the leading #) and modify following entries according to the data collected during the influxdb config:
   - # [[outputs.influxdb_v2]] ⟶ just uncomment
   - # urls = ["http://127.0.0.1:8086"] ⟶ urls = ["http://127.0.0.1:< *influxport* >"]
   - # token = "" ⟶ token = "< *token* >"
   - # organization = "" ⟶ organization = "< *org* >"
   - # bucket = "" ⟶ bucket = "< *bucket* >"
4. save the file and leave the editor (for nano CTRL-o, Return, CTRL-X)

To check if the configuration is correct:
`$ telegraf`

If there are no errors in the Telegraf output the config is ok:

```
023-02-16T06:07:26Z I! Using config file: /etc/telegraf/telegraf.conf
2023-02-16T06:07:26Z I! Starting Telegraf 1.25.2
2023-02-16T06:07:26Z I! Available plugins: 228 inputs, 9 aggregators, 26 processors, 21 parsers, 57
    ↪ outputs, 2 secret-stores
2023-02-16T06:07:26Z I! Loaded inputs: cpu disk diskio kernel mem processes swap system
2023-02-16T06:07:26Z I! Loaded aggregators:
2023-02-16T06:07:26Z I! Loaded processors:
2023-02-16T06:07:26Z I! Loaded secretstores:
2023-02-16T06:07:26Z I! Loaded outputs: influxdb_v2
2023-02-16T06:07:26Z I! Tags enabled: host=worker2.novalocal
2023-02-16T06:07:26Z I! [agent] Config: Interval:10s, Quiet:false, Hostname:"worker2.novalocal", Flush
    ↪ Interval:10s
^C2023-02-16T06:07:43Z I! [agent] Hang on, flushing any cached metrics before shutdown
```

This output shows which plugins are loaded for input and output. If there are warnings they can be ignored.

To stop telegraf:
`$ CTRL-C`

Now the telegraf service can be started and enabled:
```
$ sudo systemctl start telegraf
$ sudo systemctl enable telegraf
```

Check if telegraf has been started:
`$ sudo systemctl status telegraf`

This should report the service as "Started Telegraf", yet one may see a line like this:

```
Okt 07 14:51:45 cluster-manager.novalocal systemd[1]: Started Telegraf.
Okt 07 14:51:51 cluster-manager.novalocal systemd[1]: /usr/lib/systemd/system/telegraf.service:12: U
```

This just means that an older version of systemd is running and does not know the given option "ImportCredential". Usually this is not a problem, as systemd will not crash a service with unknown options.

## Hints

- the pre-configured standard plugin configuration can be found in the section *[[inputs.cpu]]* of the */etc/telegraf/telegraf.conf* file. More details to the settings/metrics for this plugin can be found on the according ≪CPU Input Plugin≫ webpage.

- the use of 127.0.0.1 here works as the initial installation of the plugin is on the same server as the influxDB - if one installs Telegraf on the compute nodes the config on the nodes has to be changed to use URL:$< influxip >$.

## 2.3 Smoke test for Telegraf-InfluxDB connection

The influx commandline tools can be used to check if data is arriving in the influxdb. On the *frontend* do:

```
$ influx query
```

This opens a query pipeline. The influx tool now waits for a query - it looks like nothing happens. Just copy this query (replace $<$ *bucket* $>$ with your bucketname) to the shell with the query pipeline

```
from(bucket: "<bucket>") |> range(start: -10m)
```

Press Ctrl-d, Ctrl-d to execute the query

This should give some output with table info, system info and data. Depending of the amount of data it will look like:

```
Table: keys: [_start, _stop, _field, _measurement, device, fstype, host, mode, path]
                  _start:time _stop:time _field:string _measurement:string device:string fstype:string
                      ↪ host:string mode:string path:string _time:time _value:int
------------------------------ ------------------------------ ---------------------
    ↪ ---------------------- ---------------------- ---------------------- ----------------------
    ↪ ---------------------- ---------------------- ------------------------------
    ↪ --------------------------
2023-02-16T06:05:22.994420188Z 2023-02-16T06:15:22.994420188Z free disk vda1 ext4 worker2.novalocal rw /
    ↪ 2023-02-16T06:07:30.000000000Z 18264272896
Table: keys: [_start, _stop, _field, _measurement, device, fstype, host, mode, path]
                  _start:time _stop:time _field:string _measurement:string device:string fstype:string
                      ↪ host:string mode:string path:string _time:time _value:int
------------------------------ ------------------------------ ---------------------
    ↪ ---------------------- ---------------------- ---------------------- ----------------------
    ↪ ---------------------- ---------------------- ------------------------------
    ↪ --------------------------
2023-02-16T06:05:22.994420188Z 2023-02-16T06:15:22.994420188Z inodes_free disk vda1 ext4 worker2.novalocal
    ↪ rw / 2023-02-16T06:07:30.000000000Z 1274475
Table: keys: [_start, _stop, _field, _measurement, device, fstype, host, mode, path]
                  _start:time _stop:time _field:string _measurement:string device:string fstype:string
                      ↪ host:string mode:string path:string _time:time : _value:int
------------------------------ ------------------------------ ---------------------
    ↪ ---------------------- ---------------------- ---------------------- ----------------------
    ↪ ---------------------- ---------------------- ------------------------------
    ↪ --------------------------
2023-02-16T06:05:22.994420188Z 2023-02-16T06:15:22.994420188Z inodes_total disk vda1 ext4
    ↪ worker2.novalocal rw / 2023-02-16T06:07:30.000000000Z 1310720
Table: keys: [_start, _stop, _field, _measurement, device, fstype, host, mode, path]
                  _start:time _stop:time _field:string _measurement:string device:string fstype:string
                      ↪ host:string mode:string path:string _time:time _value:int
------------------------------ ------------------------------ ---------------------
    ↪ ---------------------- ---------------------- ---------------------- ----------------------
    ↪ ---------------------- ---------------------- ------------------------------
    ↪ --------------------------
2023-02-16T06:05:22.994420188Z 2023-02-16T06:15:22.994420188Z inodes_used disk vda1 ext4 worker2.novalocal
    ↪ rw / 2023-02-16T06:07:30.000000000Z 36245
Table: keys: [_start, _stop, _field, _measurement, device, fstype, host, mode, path]
                  _start:time _stop:time _field:string _measurement:string device:string fstype:string
                      ↪ host:string mode:string path:string _time:time _value:int
------------------------------ ------------------------------ ---------------------
    ↪ ---------------------- ---------------------- ---------------------- ----------------------
    ↪ ---------------------- ---------------------- ------------------------------
    ↪ --------------------------
```

```
2023-02-16T06:05:22.994420188Z 2023-02-16T06:15:22.994420188Z total disk vda1 ext4 worker2.novalocal rw /
    ↪ 2023-02-16T06:07:30.000000000Z 21046689792
Table: keys: [_start, _stop, _field, _measurement, device, fstype, host, mode, path]
                  _start:time _stop:time _field:string _measurement:string device:string fstype:string
                      ↪ host:string mode:string path:string _time:time _value:int
```

## Installing Grafana and setting up a simple dashboard 3: Tutorial (35 min)

Grafana will be used to display data collected by the Telegraf agent. In the following the Grafana service/server will be setup and configured. At the end a simple dashboard will be created.

### 3.1 Installing Grafana package

Grafana will be the outward facing user interface that will be used to display plots from time-series data. Therefor it needs to be installed on the *frontend*:
`$ sudo yum install grafana`

This will install multiple packages.

**Important!**
FYI: If there is need to adjust the port for Grafana - it is 3000 by default - follow the next steps. **It should not be necessary for 2026 HPCSA**. Continue with starting grafana below after "Skip End")

— Skip

Before starting Grafana the port has to be adjusted to avoid conflicts:

- open /etc/grafana/grafana.ini in an editor with sudo right (e.g sudo nano - install it if not installed)
- search for the option *;http_port*
- remove the semicolon at the beginning of the line
- set the value to 8007
- save the file and exit

— Skip End

Now the Grafana server can be started:
`$ sudo systemctl start grafana-server`

The status should be checked as before with InfluxDB and Telegraf.
`$ sudo systemctl status grafana-server`

If the server is running it should be checked if the port is set correctly to “3000” (or the port you selected):
`$ sudo systemctl status grafana-server | grep address`

The output should show "Active: active (running) and the correct port should to be used by grafana-server:

```
Okt 08 07:23:49 cluster-manager.novalocal grafana[79303]: logger=http.server
    ↪ t=2025-10-08T07:23:49.158700958+02:00 level=info msg="HTTP Server Listen" address=[::]:3000
    ↪ protocol=http subUrl= socket=
```

If everything is ok the service can be enabled permanetly:
`$ sudo systemctl enable grafana-server`

## 3.2 First admin login

Setting up Grafana is done via the web-interface. Grafana provides its own web server. First the "admin" account has to be setup. This happens automatically when trying to login to Grafana for the first time.

### Steps

1. on you local computer: forward the port 3000 from the local to remote machine:
   `$ ssh -L 127.0.0.1:3000:127.0.0.1:3000 -i <YOUR KEY> -o ServerAliveInterval=60 cloud@<FLOATING IP>`
2. open a browser
3. URL to use: localhost:3000
4. enter "admin" as user name
5. enter "admin" as password
6. Grafana ask for a new password and the according confirmation - enter an arbitrary password
7. the password should be noted down as $< grafanaadminpass >$ - just in case

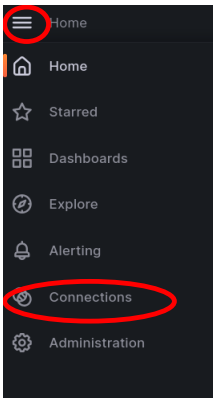
### Hints

- If you loose the Grafana admin password it can be reset on the frontend:
  `$ sudo grafana-cli -homepath "/usr/share/grafana" admin reset-admin-password <new password>`

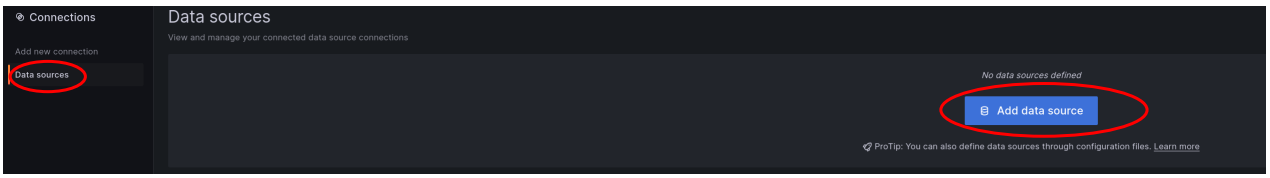
## 3.3 Create datasource

In order to display data in Grafana the application requires the information where and how to retrieve the data from. This is information a "Datasource". It is possible to define multiple datasources, but in this case only one is created.

As "admin" on the Grafana server select "Configuration – Data Sources" using the cogwheel symbol on the toolbar on the bottom left:

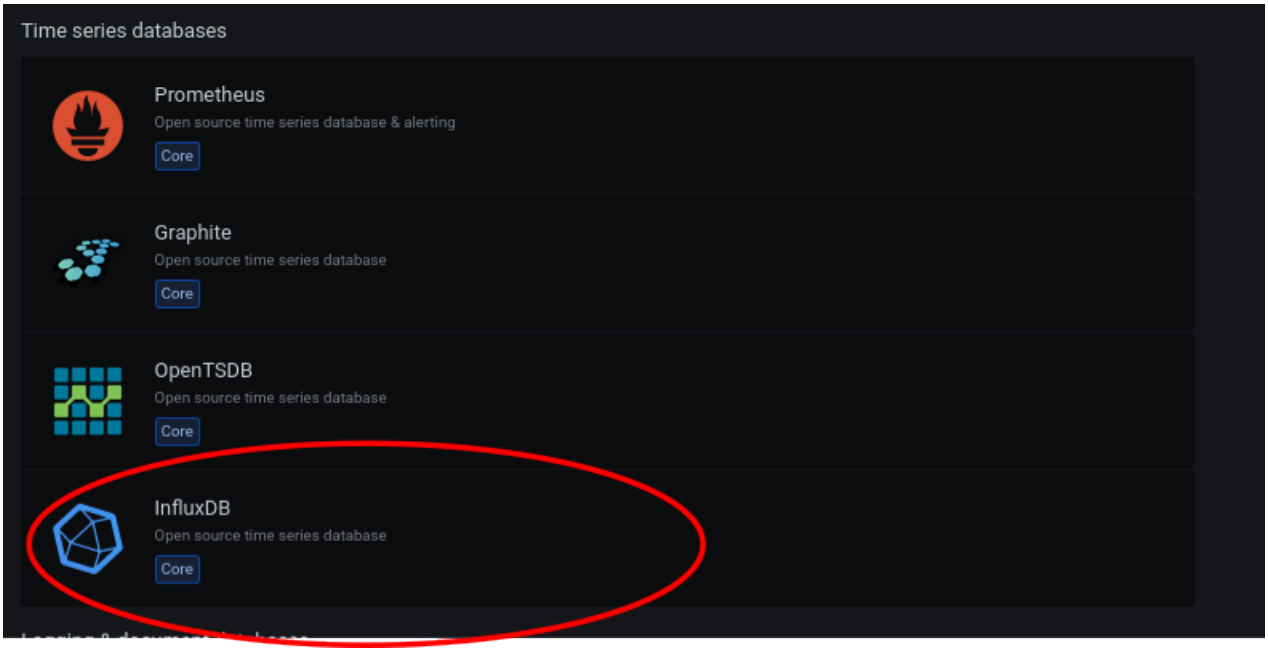



To add a datasource select the "Add data source" button:

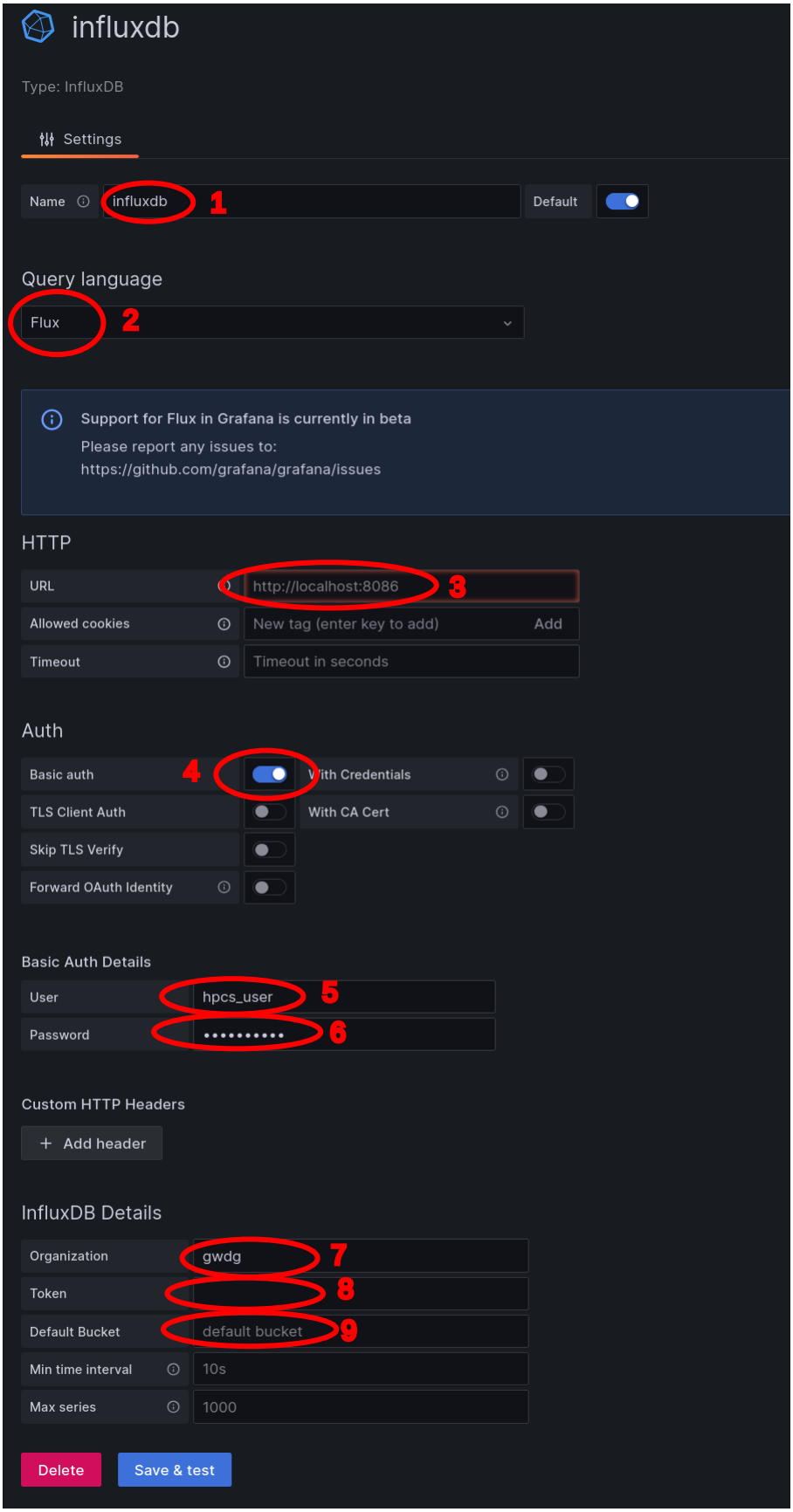


Now the data gathered before has to be provided in the form:

Select the option “InfluxDB” in the section time series db:




Fill out the given form with the collected information, check the image and the text below for hints:

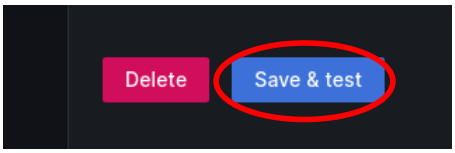


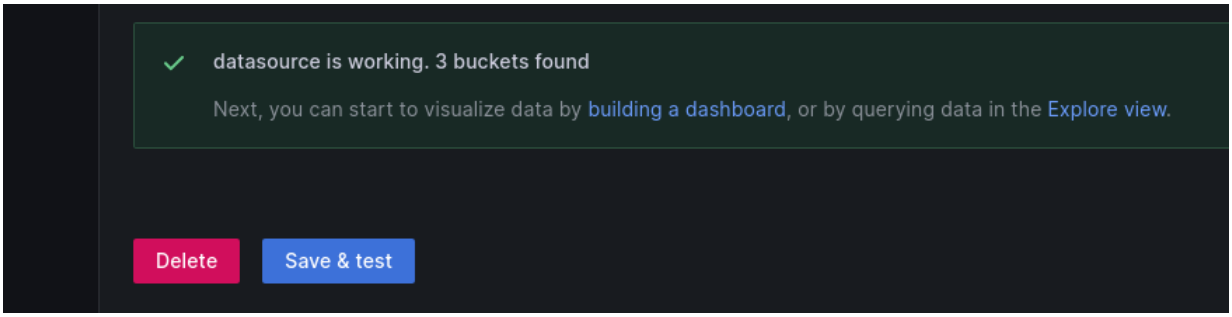

Data to change:

1. Name — select any name
2. Query Language — select “Flux”
3. URL — http://*localhost* :< *influxport* >

4. Enable basic authentication
5. User is $< influxuser >$
6. password for the influx user should be $< influxpassword >$
7. Organization — $< org >$
8. Token — $< token >$
9. Default Bucket — $< bucket >$

When done press the "Save and test" button at the bottom of the form. If everything is setup correclty there will be green feedback:



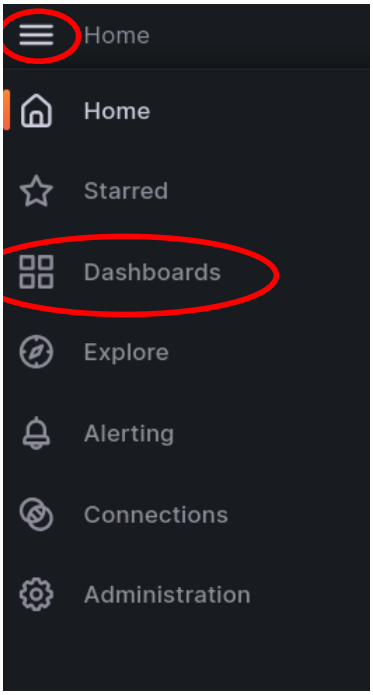

Grafana is now able to connect to the given database and retrieve metrics to display

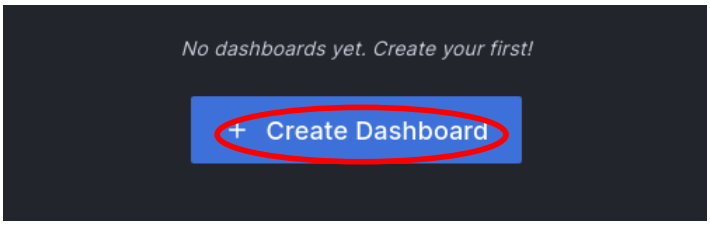
### 3.4 Create simple dashboard

As the datasource is setup it is possible to create panels showing plots from metrics of this source. In the following a simple dashboard is created using the Flux query language. Flux is a topic on its own, so the according query will be provided.

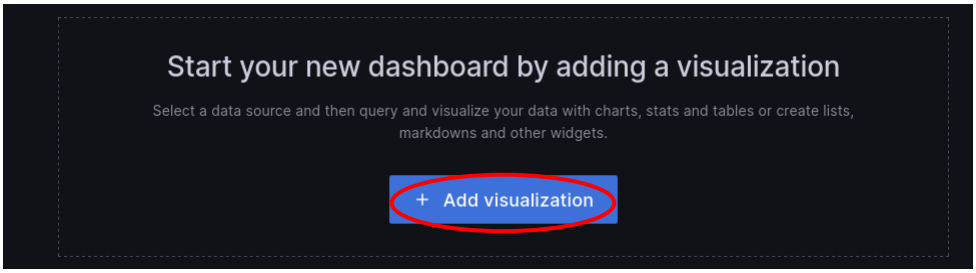
This can be done as user "admin" in Grafana as no other user is created yet.

Create a new dashboard by entering the dashboard menu and pressing pressing "Create dashboard" button:

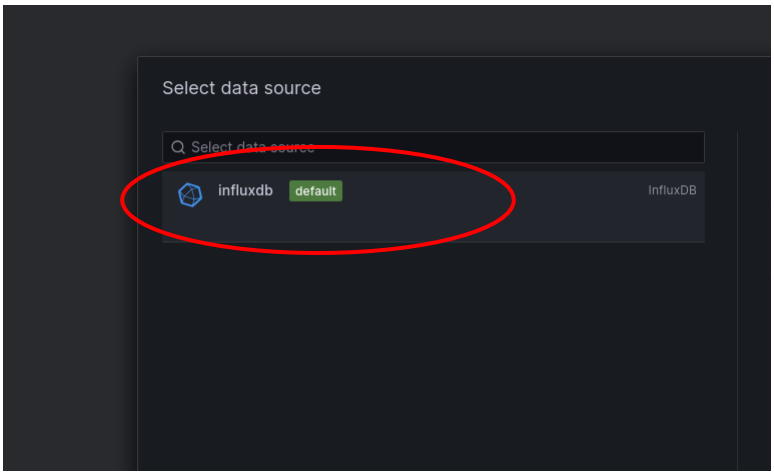


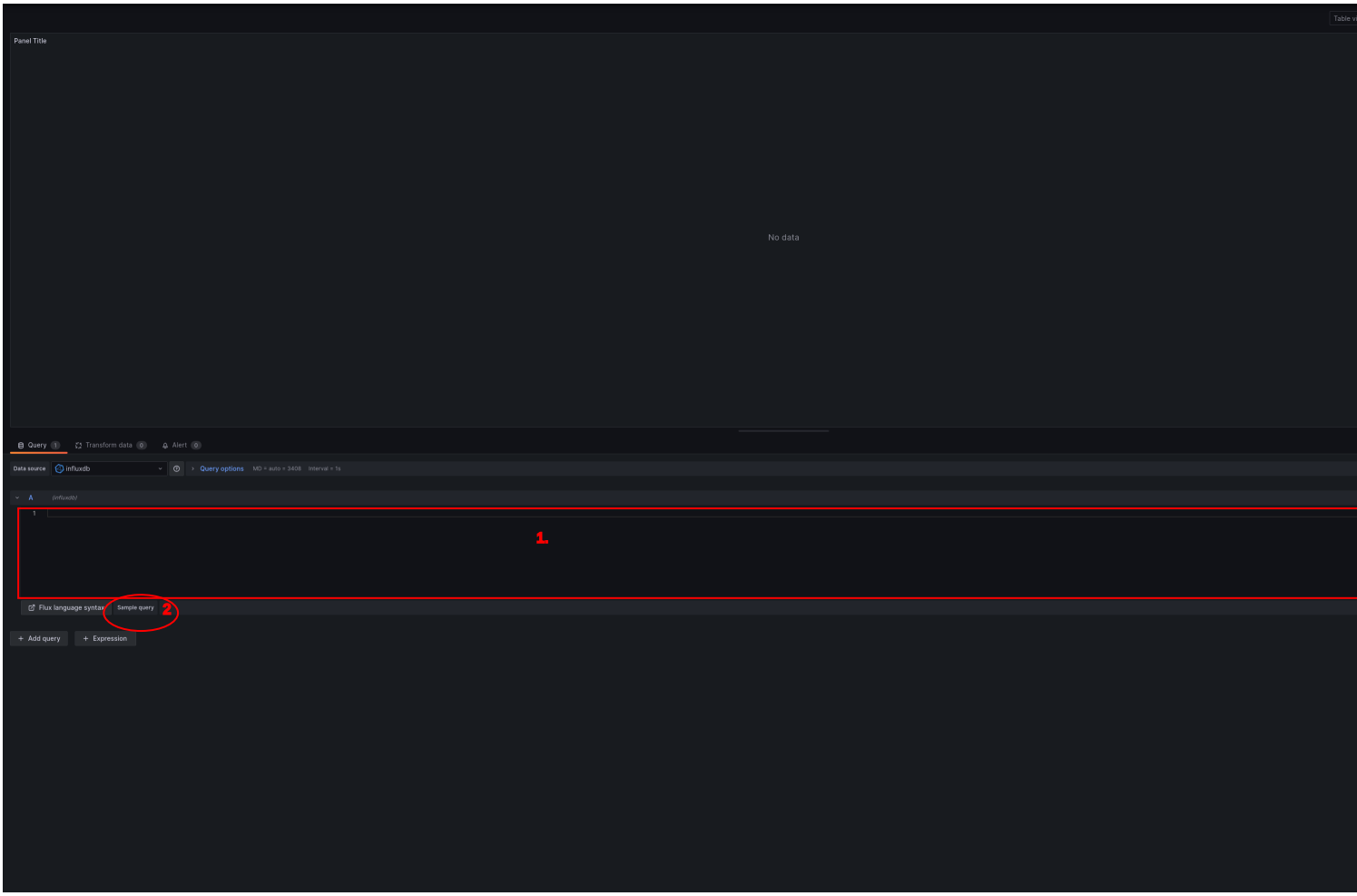

The dashboard creation interface shows up. Select the "Add visualization" button:




You will be asked for the data source to use - select the one that you created before:




The panel editor is now presented. It is already setup to the correct datasource as it is the only one created yet.




First query:

- Copy and paste this Flux query into the Query editor (1.) and modify the bucket to match < *bucket* > from the InfluxDB config:

```
from(bucket: "<bucket>")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "cpu")
  |> filter(fn: (r) => r["_field"] == "usage_user")
  |> filter(fn: (r) => r["cpu"] == "cpu-total" or r["cpu"] == "cpu0" or r["cpu"] == "cpu1")
  |> yield(name: "mean")
```

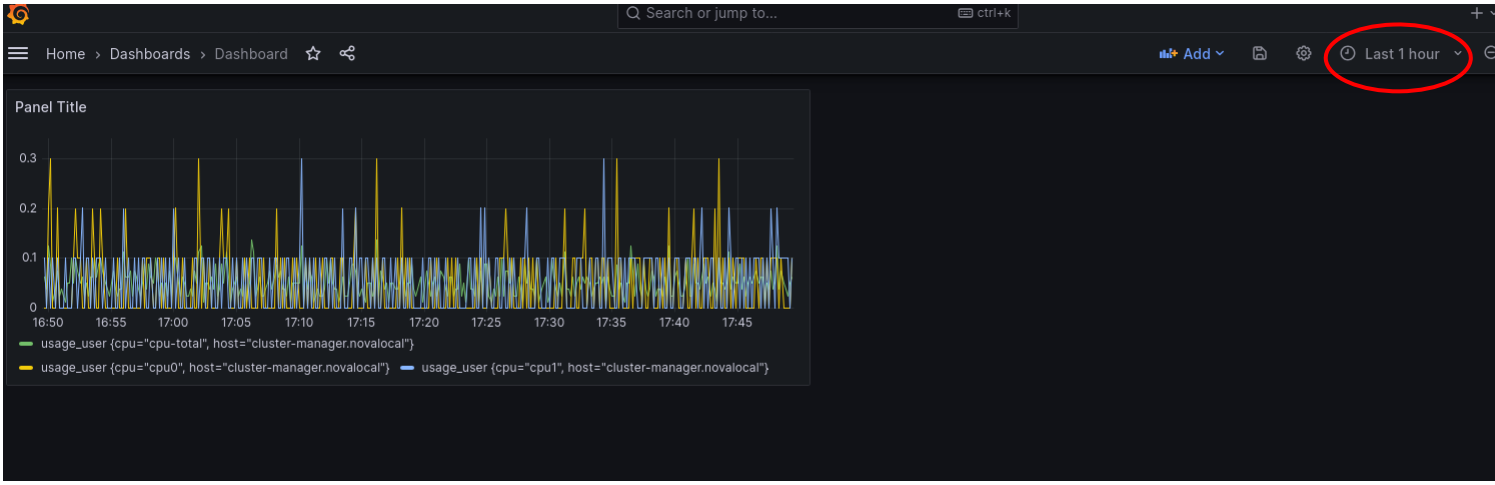
- Modify the < *bucket* > variable to fit your name.
- Press 2. "Sample Query"

This query request data from the given bucket with in the time-range given (this is a dynamic timerange). The InfluxDB will first select all data in the given timespan, and they apply the filters on it. In this case the query filters for the general CPU usage of the users on the systems which provide the according measurements.

If you want you can give the Panel a name (Panel options on the right side)

Press "Save" in the top right corner of the editor page and save the dashboard with a meaningfull name.

You now have finalized your first dashboard. You can play with the timeframe to display to zoom in and out:




## Optional: Explore data and create own queries 4: Tutorial (0 min)

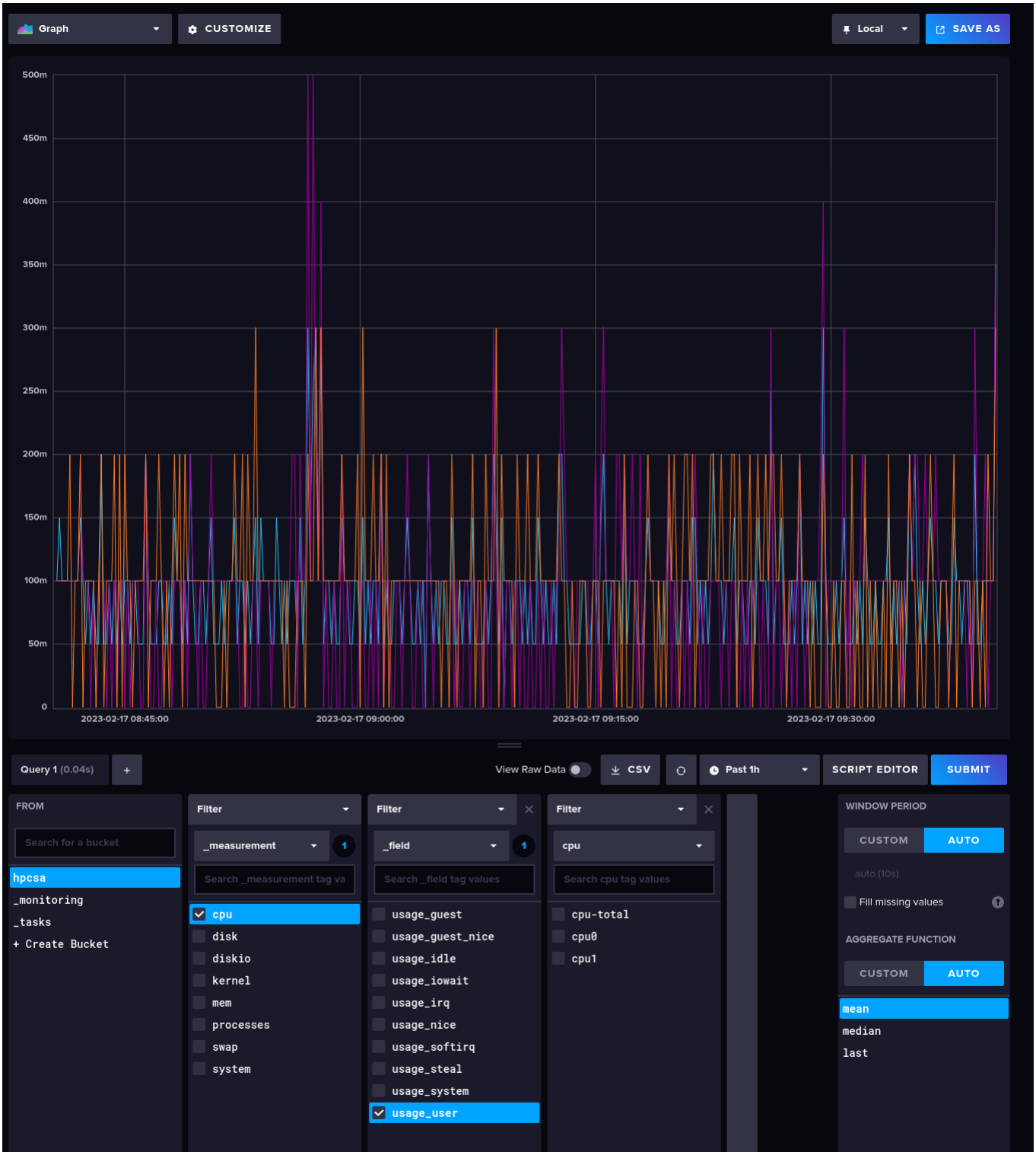
The dashboard created is quite simple. The issue with creating more interesting panel is to know the available metrics and the syntax. InfluxDB provides a web interface for data exploration to make this easier for the users.

This web interface can be reached via web browser on http://localhost:$< influxport >$ again via port forwarding. Execute the same forwarding ssh command as for Grafana but with the correct port (likely 8009)

Login is $< influxuser >$ and $< influxpassword >$.

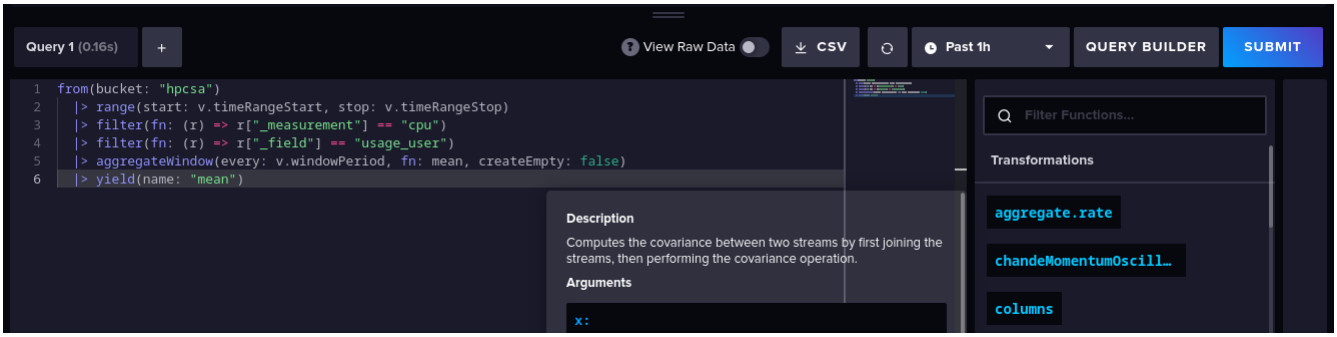
Select the dataexplorer from the toolbar on the left:

This will open the dataexplorer. At the bottom are columns letting users select, the bucket, measurements, according metrics - with this it is easy to dig down through the data in InfluxDB.




To execute a query press the “Submit” button. This will build the graph accordingly.

In order to get the script for this query to use in other tools, e.g. Grafana, the script editor has to be openened by pressing “Script Editor”.




To use the script displayed there in Grafana just mark the script and copy/paste it to the query editor in the panel creation in Grafana.

## Optional: Install another Telegraf Plugin 5: Tutorial (0 min)

This requires a working TIG Stack as it has been installed in the previous tasks.

Check the available plugins on the according webpage https://docs.influxdata.com/telegraf/v1.20/plugins/ , select one of them, install and configure it, according to documentation.

### Hints

- After updating the Telegraf config the agent has to be restarted to utilize the new config.

## Optional: Install telegraf on the worker nodes 6: Tutorial (0 min)

Requires at least the PXE booting worker nodes setup before.

Two worker nodes have been set up during previous sessions. Install telegraf on those machines via warewulf images and integrate them into a dashboard.

### Hints

- the telegraf config for the influxdb output plugin has to be modified to use utilize $< influxip >$ instead of 127.0.0.1 on the nodes.

## Optional: Setup Database and Grafana with specific users 7: Tutorial (0 min)

Requires a basic TIG stack running as setup in the first 3 tasks.

Recap the previous installation. Just one main user has been used to access the database and the Grafana dashboard. Usually this is considered bad practice in regards to security. Setup users for the InfluxDB and Grafana and modify all parts of the stack accordingly.

### Hints

- Think about the following: Wow many users would be useful for the InfluxDB and for Grafana? What rights should they have? Why?
- Do not forget to change parts the telegraf configuration.

## Optional: Central setup for telegraf 8: Tutorial (0 min)

Requires the working TIG stack from this tutorial and telegraf already integrated on the worker nodes.

In a previous lecture slurm has been setup using a central installation and configuration. Is this possible for telegraf, too? Or has something to be considered and adapted to telegraf. Implement you solution.

### Hints

- HInt for tutor: you may have different hardware installed which utilize different plugins. Therefor at least the configuration has to be local for the specific machine types/hardware.

### Further Reading

- https://grafana.com/docs/grafana/latest/getting-started/get-started-grafana-influxdb/
- https://grafana.com/docs/grafana/latest/introduction/
- https://docs.influxdata.com/influxdb/v2.6/install/
- https://docs.influxdata.com/telegraf/v1.20/plugins/