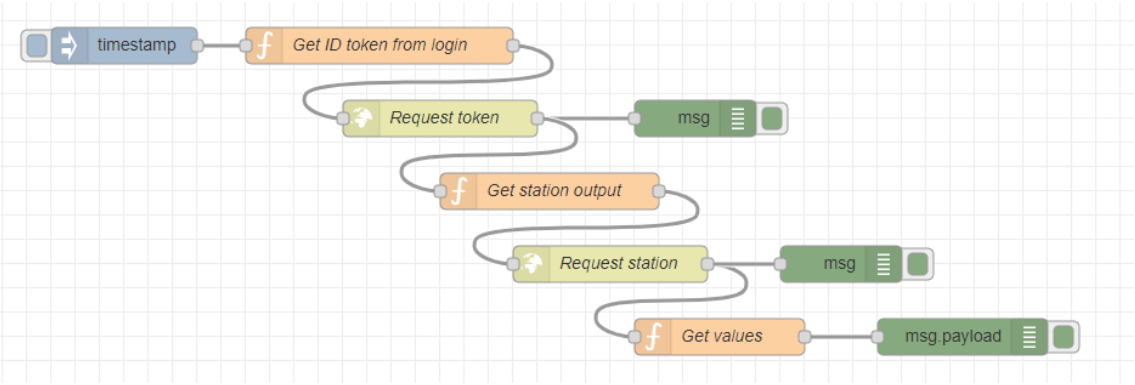
# Lectura Inversor Goodwe desde Node-RED

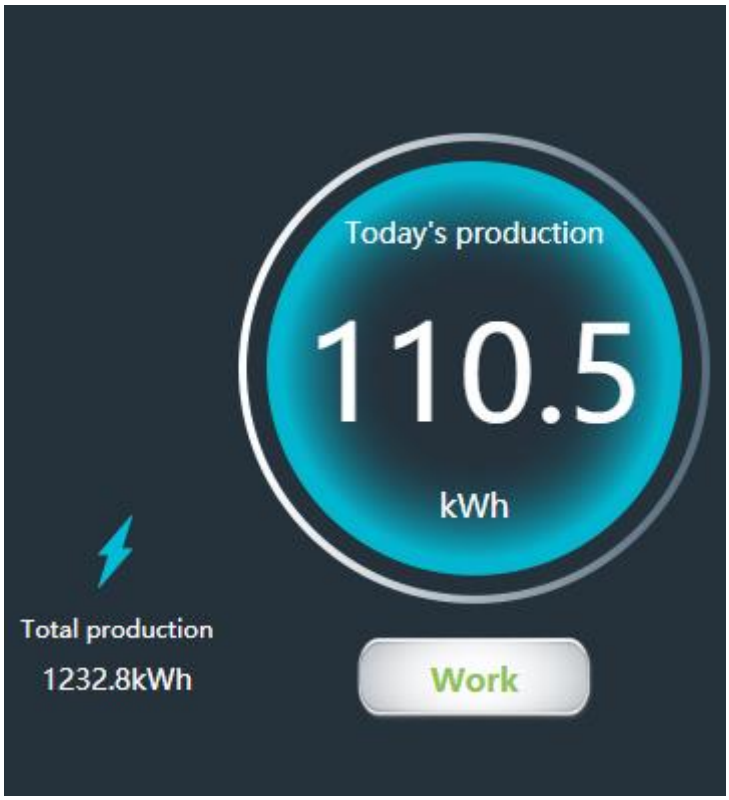
https://flows.nodered.org/flow/dbd6621ce479194571fde919dcb172d1

En la primera función se introduce login y password

En la segunda funcion se introduce el id del inversor (que es es que aparece en el Portal al final de la URL)

Y te da la potencia






```
total_power: 1232.8
day_power: 110.5
```

**Edit function node**

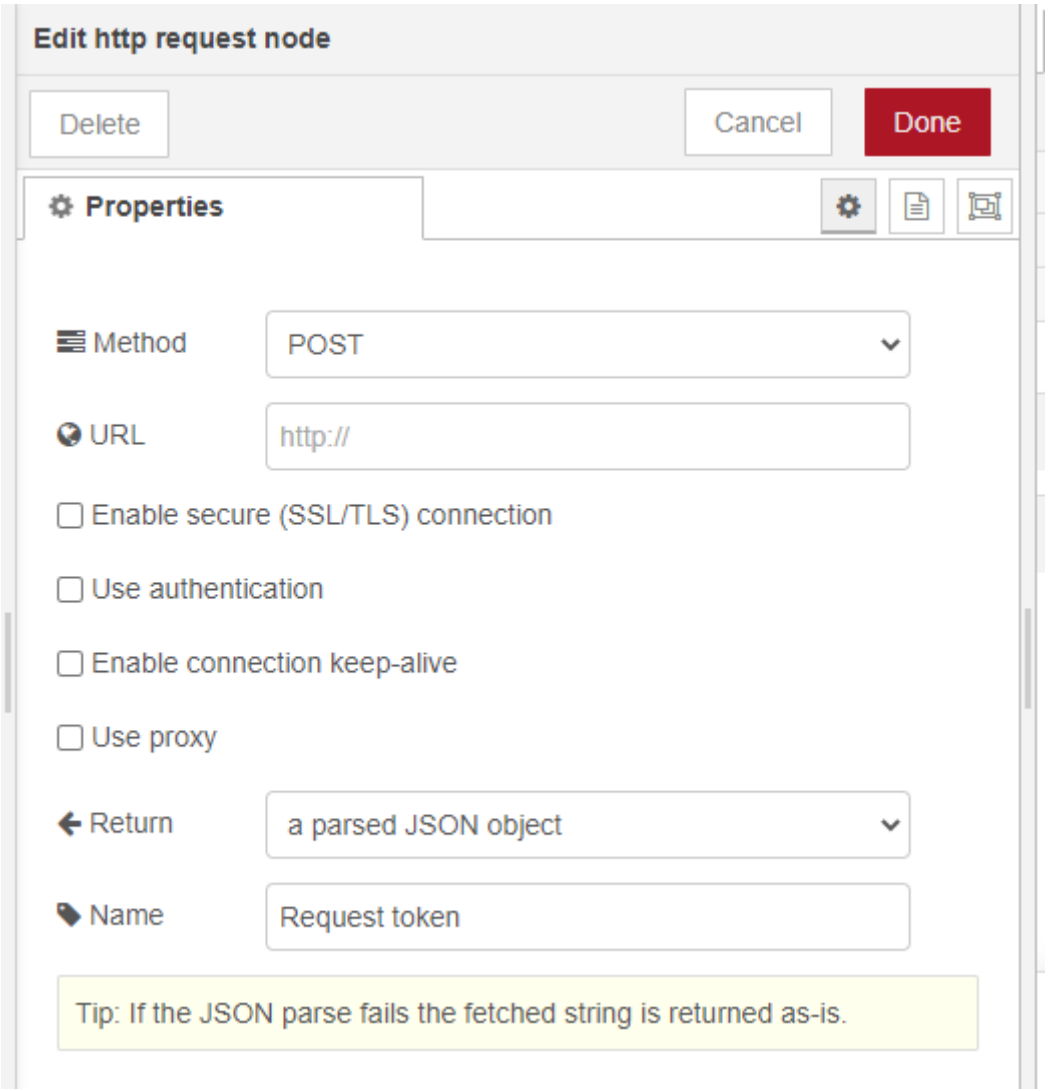
Delete

**Properties**

Name: Get ID token from login

Function

```
// Credentials as used for login to the Semsportal at https://www.semsportal.com/
account = "YOURLOGINHERE";
pwd = "YOURPASSWORDHERE";

// ---------- no changes needed below ----------------------------
loginPayload = { 'account': account, 'pwd': pwd };

// It's a given, likely from the API documentation?
token = '{"client": "ios", "version": "v2.1.0", "language": "en"}';

global_url = 'https://eu.semsportal.com/api/';
headers = { 'token': token };

msg.url = global_url + "v1/Common/CrossLogin";
msg.headers = headers;
msg.payload = loginPayload;

return msg;
```




// Power station ID can be found in the last part of the URL of the SEMSportlal, once logged in

psid = 'YOURPOWERSTATIONIDHERE';

// ---------- Generic below ----------

// https://github.com/DiedB/Homey-SolarPanels/issues/28

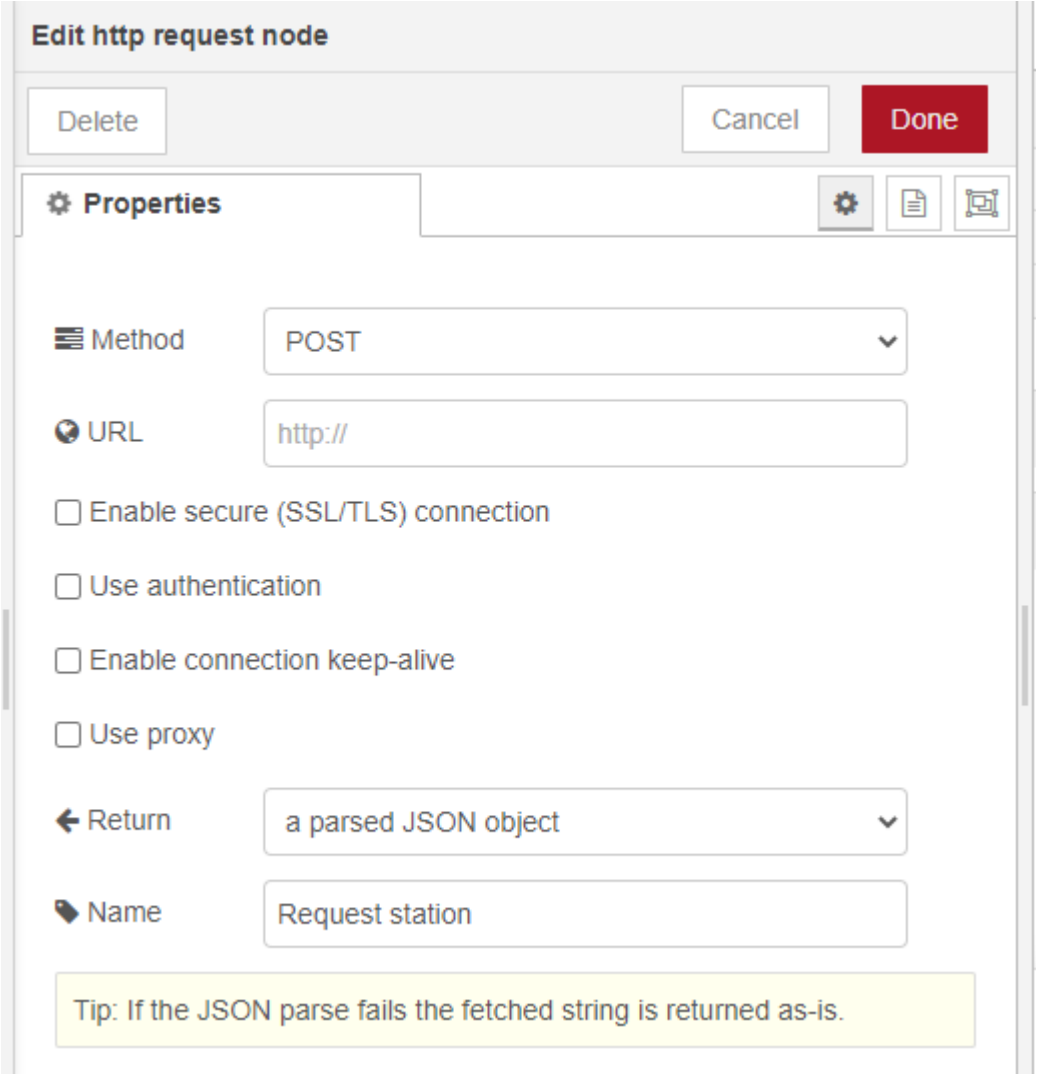
// Get the relevant response from the login call:

var uid = msg.payload.data.uid; // User ID

var id_token = msg.payload.data.token; // Time-bound token from login

var timestamp = msg.payload.data.timestamp;

```
// Warning: The API uses 'token'  with different meanings throughout the communication

token = '{"uid": "'+ uid + '", "timestamp": '+ timestamp+ ', "token": "'+ id_token + '", "client": "ios", "version": "v2.0.4", "language": "en"}';

headers = { 'User-Agent': 'JH', 'Token': token }


global_url = 'https://eu.semsportal.com/api/'

msg.url = global_url + "v1/PowerStation/GetMonitorDetailByPowerstationId";


msg.headers = headers;


payload = {'powerStationId': psid};

msg.payload = payload;



return msg;
```


Edit http request node
Delete
Cancel
Done
Properties
Method
POST
URL
http://
Enable secure (SSL/TLS) connection
Use authentication
Enable connection keep-alive
Use proxy
Return
a parsed JSON object
Name
Request station
Tip: If the JSON parse fails the fetched string is returned as-is.


Edit function node
Delete
Cancel
Done
Properties
Name
Get values
Function
// Get values as shown in the webportal:
// Stationname, top left as displayed in the dropdown (not sure how more stations handle....?)
var station_name = msg.payload.data.info.stationname;
// Fysical address
var address = msg.payload.data.info.address;
// Total power als shown on the webportal left below today's production
var total_power = msg.payload.data.kpi.total_power;
// Today's power als in the circle middle down the webportal main page
var day_power = msg.payload.data.kpi.power;
msg = {};
msg.payload = {'station_name': station_name, 'address': address, 'total_power': total_power, 'day_power': day_power};

https://github.com/DiedB/Homey-SolarPanels/issues/28#issuecomment-521042829

My apologies for not checking in earlier. My solar panels are on a new home, and other jobs still have priority. I'm sure you don't care that my backyard is now a green lawn rather than a wasteland, but my wife did care ;)

I did create a script to see if I can get some output. Here are some details.

You need to following URL endpoints:

# Log in

First, you need to get a authentication token for further requests, if you don't have one.

Post to `https://globalapi.sems.com.cn/api/v1/Common/CrossLogin`
With the headers:
```
'Content-Type': 'application/json'
'Connect': 'keep-alive'
'User-Agent': 'PVMaster/2.1.0 (iPhone; iOS 12.0; Scale/2.00)',
'Accept-Language': 'nl-BE;q=1',
'Token': '{"client":"ios","version":"v2.1.0","language":"en"}',
```
And with the body:
```
{'account': 'email@example.org', 'pwd': '123456'}
```
Note that this 'Token' stuff in the header with client information is required. If it is empty or left out, I did not get a result. Or an error message in Chinese. And I don't know why it is called 'Token' either. It has nothing to do with the authentication token you'll receive in the response.

Data returned looks like:


```
{
  "hasError": false,
  "code": 0,
  "msg": "Success",
  "data": {
    "uid": "c3518cbe-e6b8-4810-c15b-b1f48f32288f",
    "timestamp": 1565735721330,
    "token": "25b81fd55a3e45eb0434301cbfb0ea3d",
    "client": "ios",
    "version": "v2.1.0",
    "language": "en"
  },
  "components": {
    "para": null,
    "langVer": 40,
    "timeSpan": 0,
    "api": "http://eu.semsportal.com:82/api/Auth/GetToken"
  },
  "api": "https://eu.semsportal.com/api/"
}
```

From this, you need to store:

- `data > uid`
- `data > token`
- `data > timestamp`
- `api` (thus not `components > api`)

I have not determined how long the authentication token remains valid, but I get the impression not forever. Based on how often I need to log in on my computer, I expect 12 to 24 hours.

# Get Plant ID

For the first time solar panels are added, you need to get the ID of the 'solar plant'. After this first request, you can store the plant ID.

Post to `{api}PowerStationMonitor/QueryPowerStationMonitorForApp` , with the api URL you got when you logged in.
Likely: `https://eu.semsportal.com/api/PowerStationMonitor/QueryPowerStationMonitorForApp`,
With the headers:

```
'Content-Type': 'application/json'
'Connect': 'keep-alive'
'User-Agent': 'PVMaster/2.1.0 (iPhone; iOS 12.0; Scale/2.00)',
'Accept-Language': 'nl-NL;q=1',
'Token': '{'Token': '{"client": "ios", "version": "v2.1.0", "language": "en",
"timestamp": 1565735721330, "uid": "c3518cbe-e6b8-4810-c15b-b1f48f32288f", "token":
"25b81fd55a3e45eb0434301cbfb0ea3d"}'}',
```

And with the body:

```
{'page_size': '5', 'orderby': '', 'powerstation_status': '', 'key': '', 'page_index':
'1', 'powerstation_id': '', 'powerstation_type': ''}
```

With the Token header updated with the information you got when logging in.

Data returned looks like:


```
{
  "hasError": false,
  "code": 0,
  "msg": "Success",
  "data": [
    {
      "powerstation_id": "e9b465a8-d2fb-4e41-9348-ee9f49501a03",
      "stationname": "Watermunt",
      "first_letter": "",
      "adcode": "011300140010",
      "location": "",
      "status": -1,
      "pac": 0.0,
      "capacity": 4.5,
      "eday": 13.1,
      "emonth": 195.0,
      "eday_income": 604.318,
      "etotal": 2746.9,
      "powerstation_type": "residential",
      "pre_org_id": null,
      "org_id": null,
      "longitude": "4.95537042617798",
      "latitude": "52.1518383717057",
      "pac_kw": 2746.9,
      "to_hour": 2.911111111111111,
      "weather": {
        "HeWeather6": [
          {
            "basic": {
              "cid": "NL2758333",
              "location": "Breukelen",
```

```
              "parent_city": "Breukelen",
              "admin_area": "Utrecht",
              "cnty": "Netherlands",
              "lat": "52.17417145",
              "lon": "5.00138998",
              "tz": "+1.00"
            },
            "update": {
              "loc": "2019-08-12 23:57",
              "utc": "2019-08-12 22:57"
            },
            "status": "ok",
            "now": {
              "cloud": "100",
              "cond_code": "300",
              "cond_txt": "Shower Rain",
              "fl": "14",
              "hum": "100",
              "pcpn": "0.3",
              "pres": "1014",
              "tmp": "14",
              "vis": "16",
              "wind_deg": "90",
              "wind_dir": "E",
              "wind_sc": "1",
              "wind_spd": "4"
            }
          }
        ]
      },
      "currency": "EUR",
      "yield_rate": 0.22,
      "is_stored": false
    }
  ],
  "components": {
    "para": null,
    "langVer": 40,
    "timeSpan": 0,
    "api":
"http://eu.semsportal.com:82/api/PowerStationMonitor/QueryPowerStationMonitorForApp"
  }
}
```


From this, you need to store:

- `data > [item 0] > powerstation_id`

Optionally, you can ask the user which power station to use. Likely very few people will have more than one, but by listing the options and name, it is immediately apparent for the user that log in was successful.

# Get Current Data

Post to `{api}v1/PowerStation/GetMonitorDetailByPowerstationId` , with the api URL you got when you logged in.
Likely: `https://eu.semsportal.com/api/v1/PowerStation/GetMonitorDetailByPowerstationId`,
With the headers:

```
'Content-Type': 'application/json'
'Connect': 'keep-alive'
'User-Agent': 'PVMaster/2.1.0 (iPhone; iOS 12.0; Scale/2.00)',
```

```
'Accept-Language': 'nl-NL;q=1',
'Token': '{'Token': '{"client": "ios", "version": "v2.1.0", "language": "en",
"timestamp": 1565735721330, "uid": "c3518cbe-e6b8-4810-c15b-b1f48f32288f", "token":
"25b81fd55a3e45eb0434301cbfb0ea3d"}'}',
```

And with the body:

```
{'powerStationId': 'e9b465a8-d2fb-4e41-9348-ee9f49501a03'}
```

Now you get the result as shown in https://pastebin.com/JnMJzX6N.

Note that I have two inverters, the first with one circuit, the second with two circuits.
I have not yet studied the output, but it would be great if I could see the power generated by the 3 individual circuits.

~~Note that this output was taken at night, it will most like not contain useful data. I try to run it again during daytime.~~ The linked pastebin contains a full exchange with the API during daytime.

The total current power (all inverters) seems available in data > kpi > pac

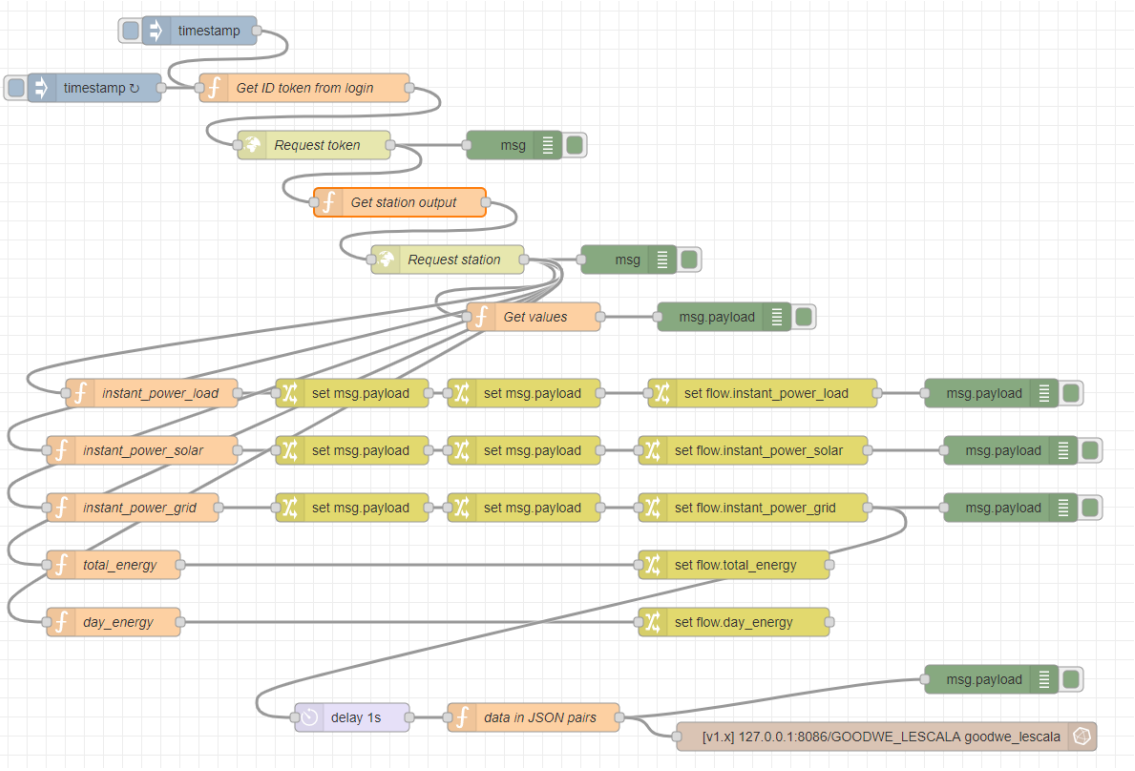
The relevant output seems at data > inverter > [item in list], and then either `d` or `invert_full` (data seems the same), and for individual circuits, multiply voltage and current. Input (DC) voltage is at vpv1, vpv2, vpv3, vpv4 (for up to 4 trackers), and input current at ipv1, ipv2, ipv3, ipv4. Similar, output voltage and current are in vac1, vac2, vac3 and iac1, iac2, iac3 respectively. I'm not sure about the accuracy, but it could even be possible to determine how effective the inverter is :).

# Get Historic Data

To get historic data, use either one of these URLs:

https://eu.semsportal.com/api/PowerStationMonitor/GetPowerStationPowerAndIncomeByDay
https://eu.semsportal.com/api/PowerStationMonitor/GetPowerStationPacByDayForApp

Let´s take the instant values and put on a database




**Edit function node**

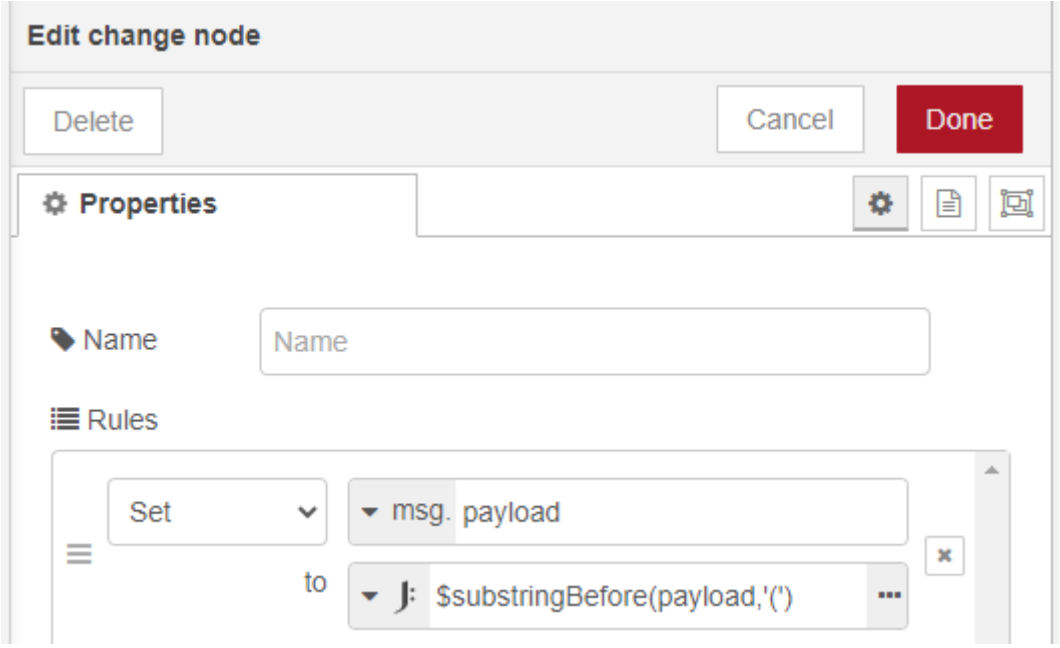
Delete

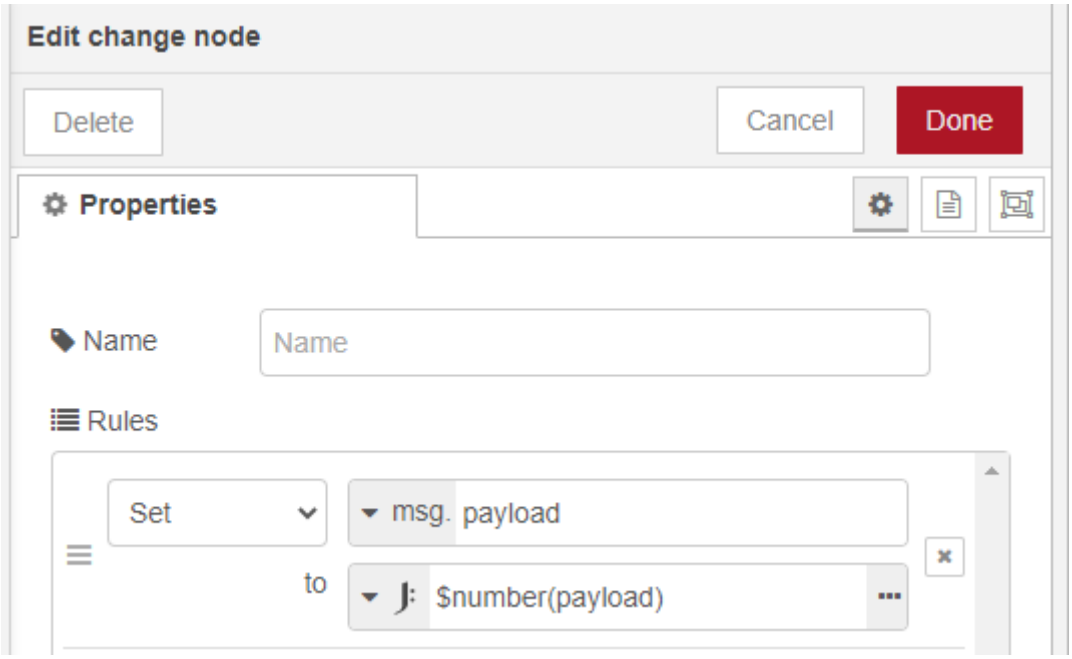
**Properties**

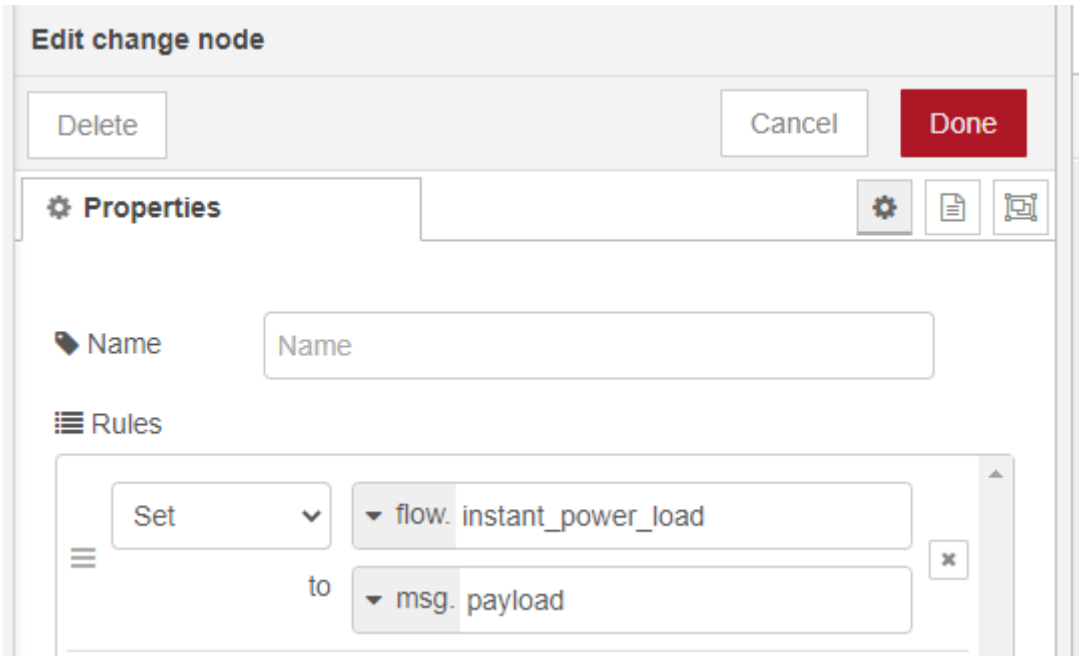
Name: instant_power_load

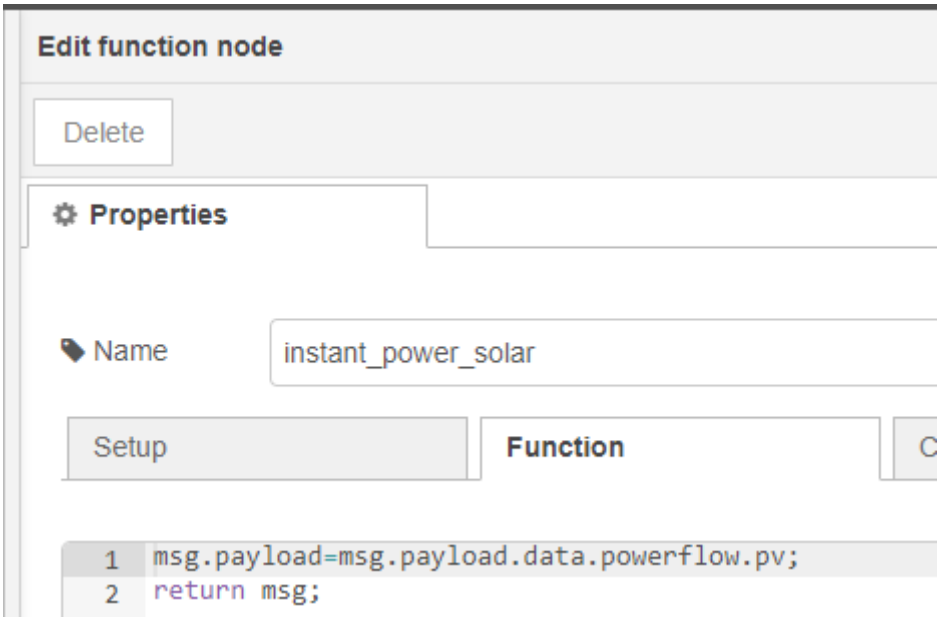
Setup | **Function**

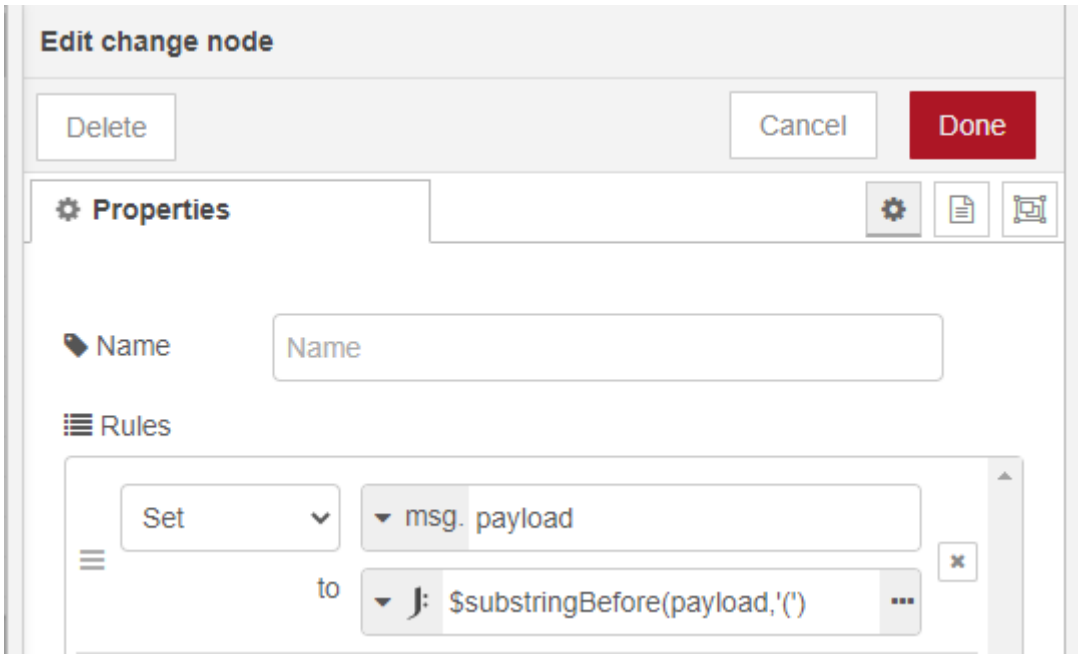
```
1  msg.payload=msg.payload.data.powerflow.load;
2  return msg;
```

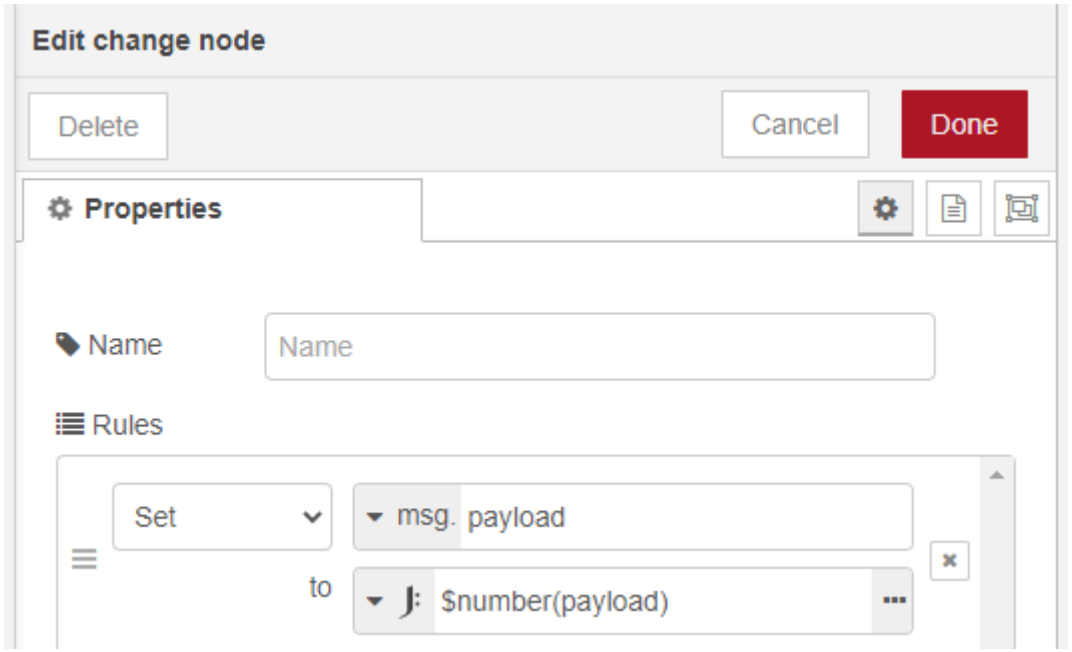

Edit change node
Delete
Cancel
Done
Properties
Name
Name
Rules
Set
msg.
payload
to
$substringBefore(payload,'(')

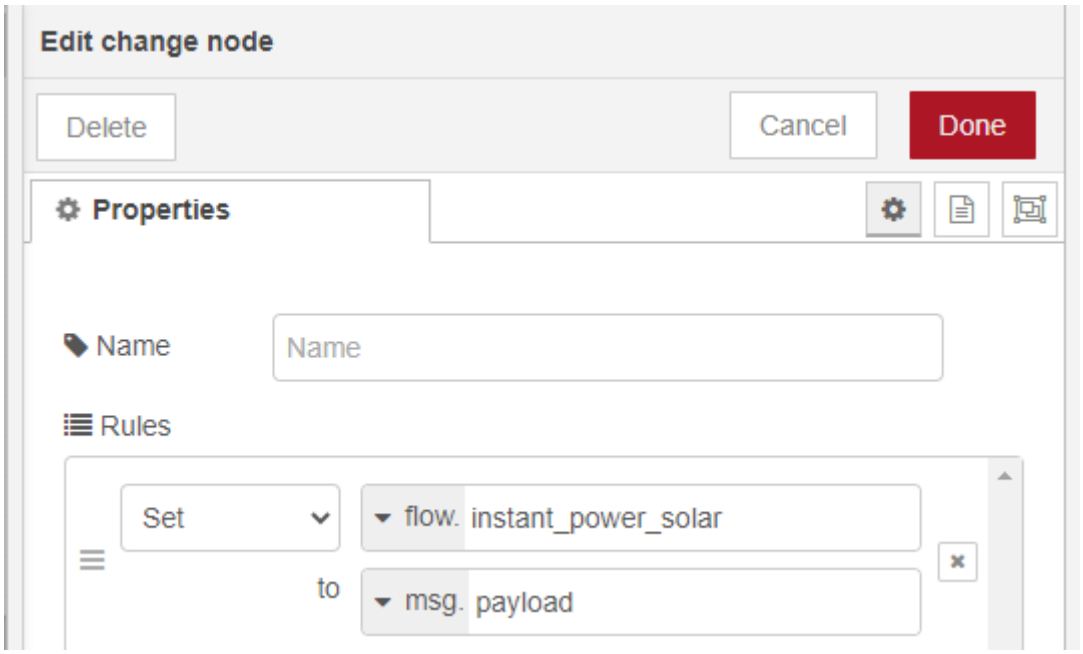

Edit change node
Delete
Cancel
Done
Properties
Name
Name
Rules
Set
msg.
payload
to
$number(payload)

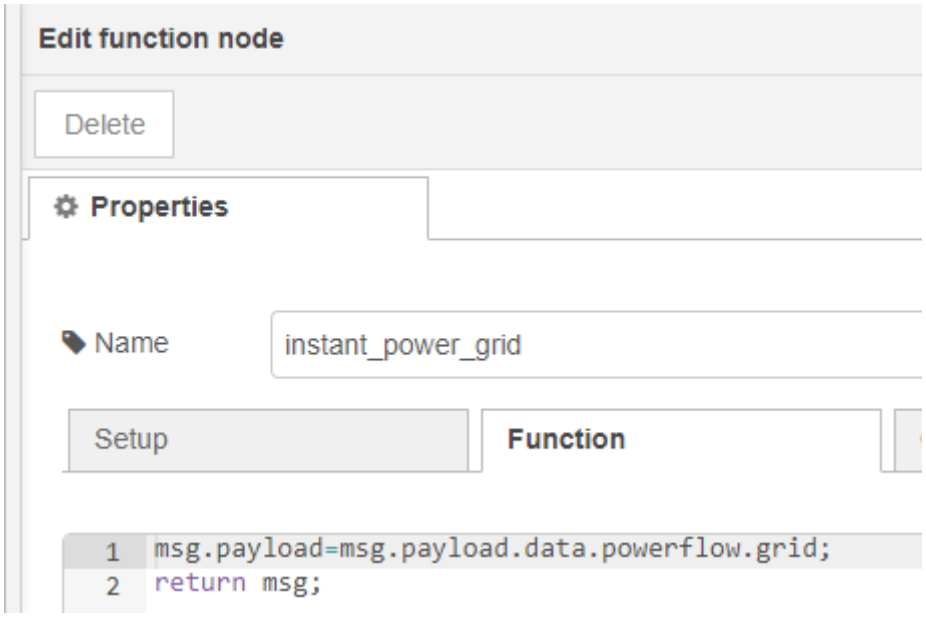

Edit change node
Delete
Cancel
Done
Properties
Name
Name
Rules
Set
flow.
instant_power_load
to
msg.
payload

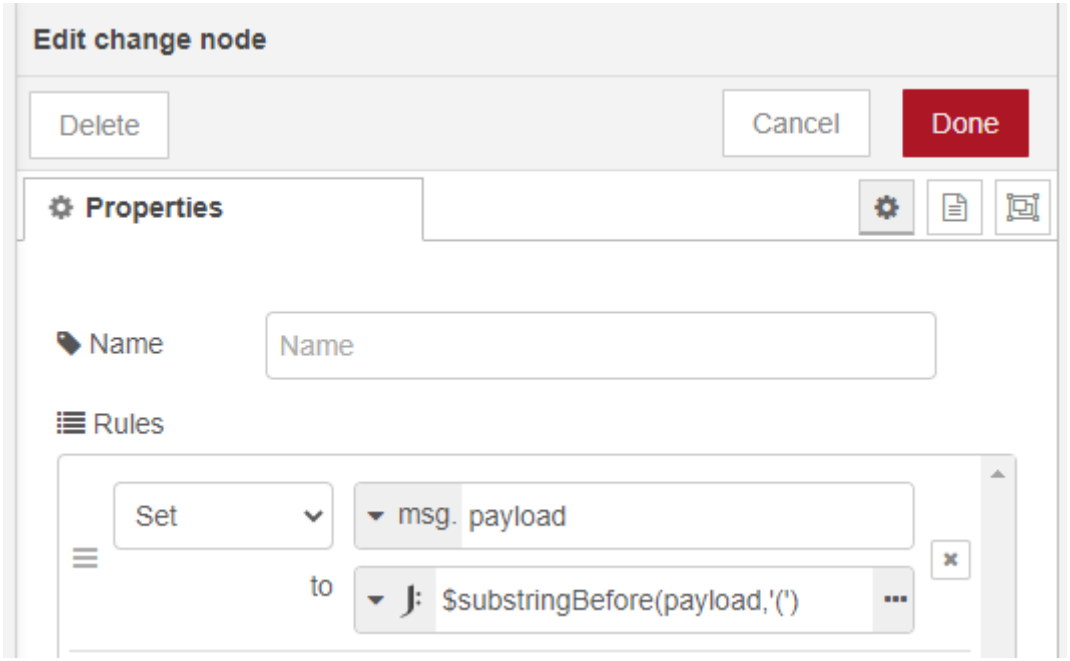

Edit function node
Delete
Properties
Name
instant_power_solar
Setup
Function
1 msg.payload=msg.payload.data.powerflow.pv;
2 return msg;

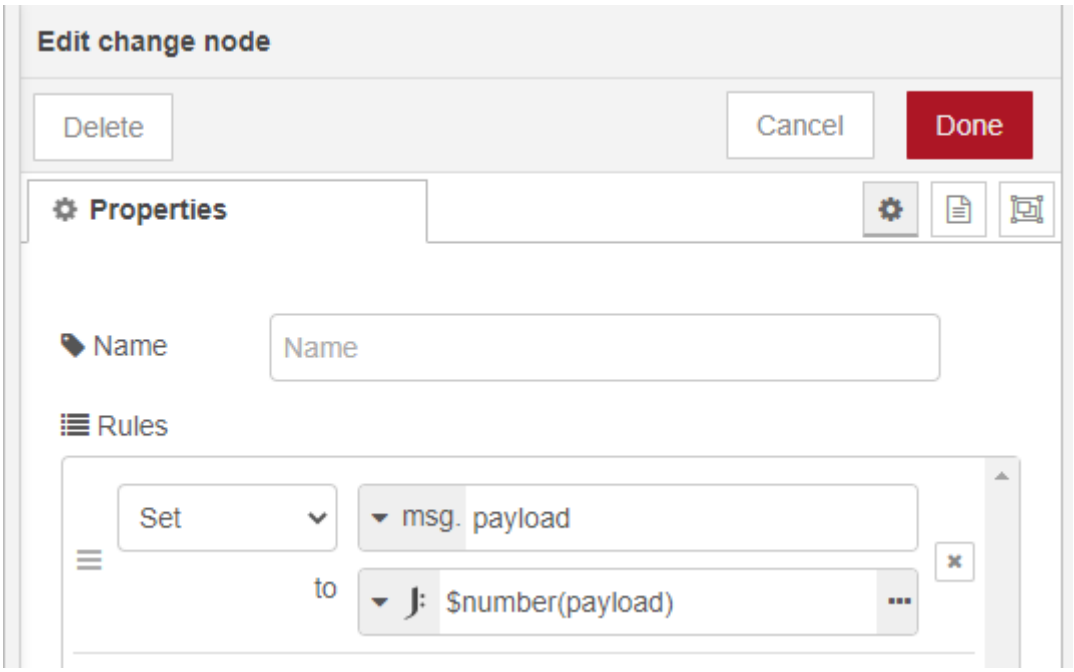

Edit change node
Delete
Cancel
Done
Properties
Name
Name
Rules
Set
msg.
payload
to
$substringBefore(payload,'(')

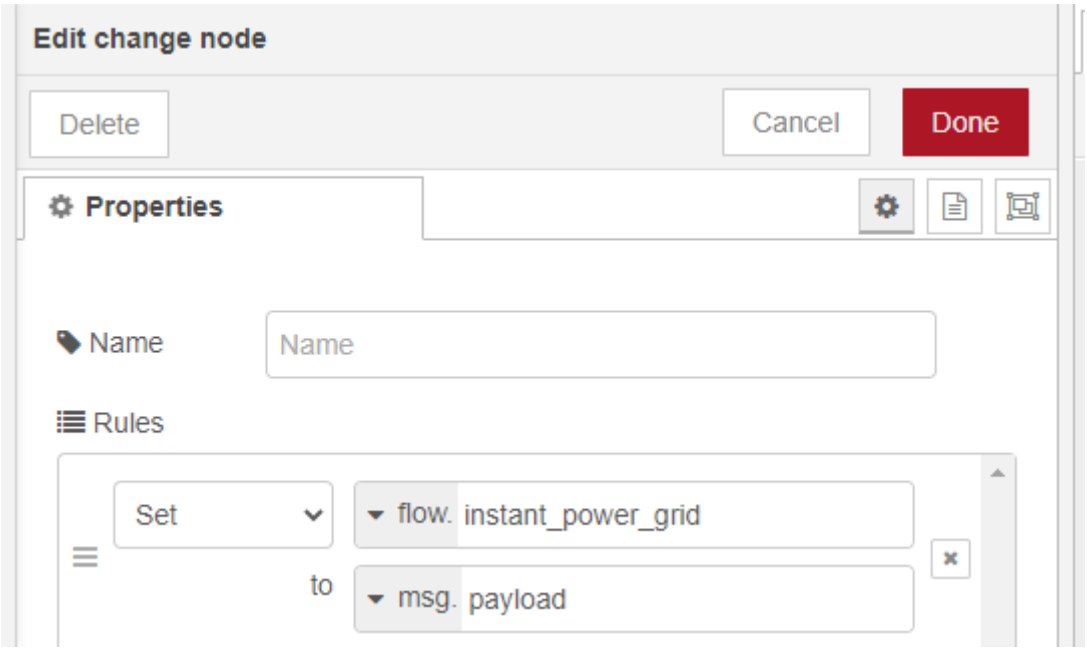

Edit change node
Delete
Cancel
Done
Properties
Name
Name
Rules
Set
msg.
payload
to
$number(payload)

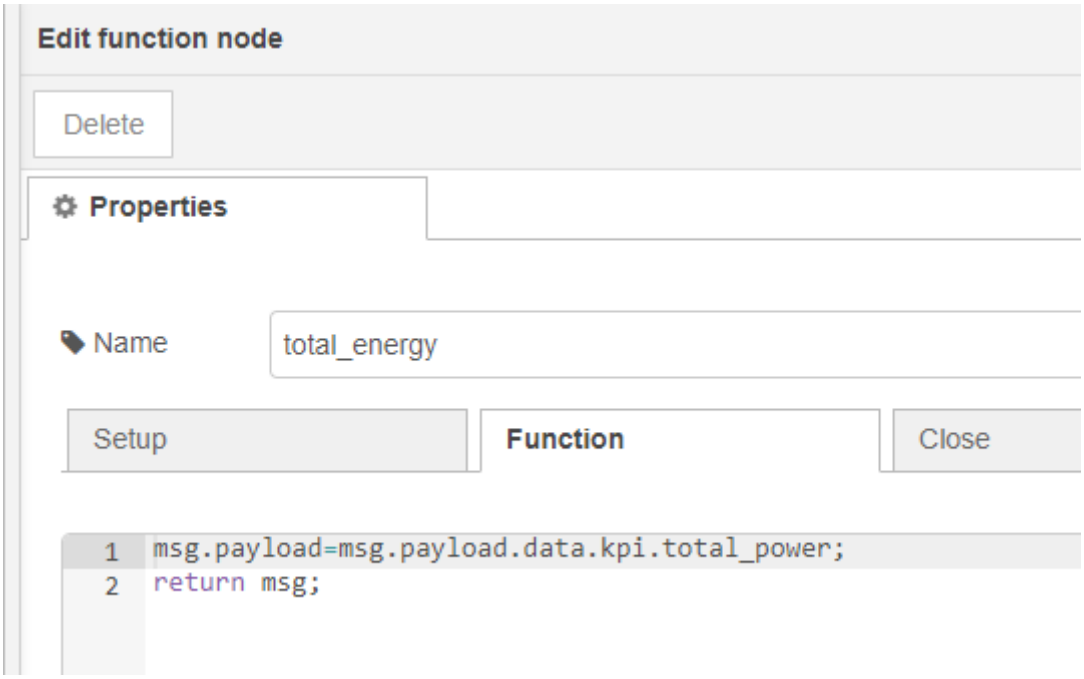

Edit change node
Delete
Cancel
Done
Properties
Name
Name
Rules
Set
flow.
instant_power_solar
to
msg.
payload

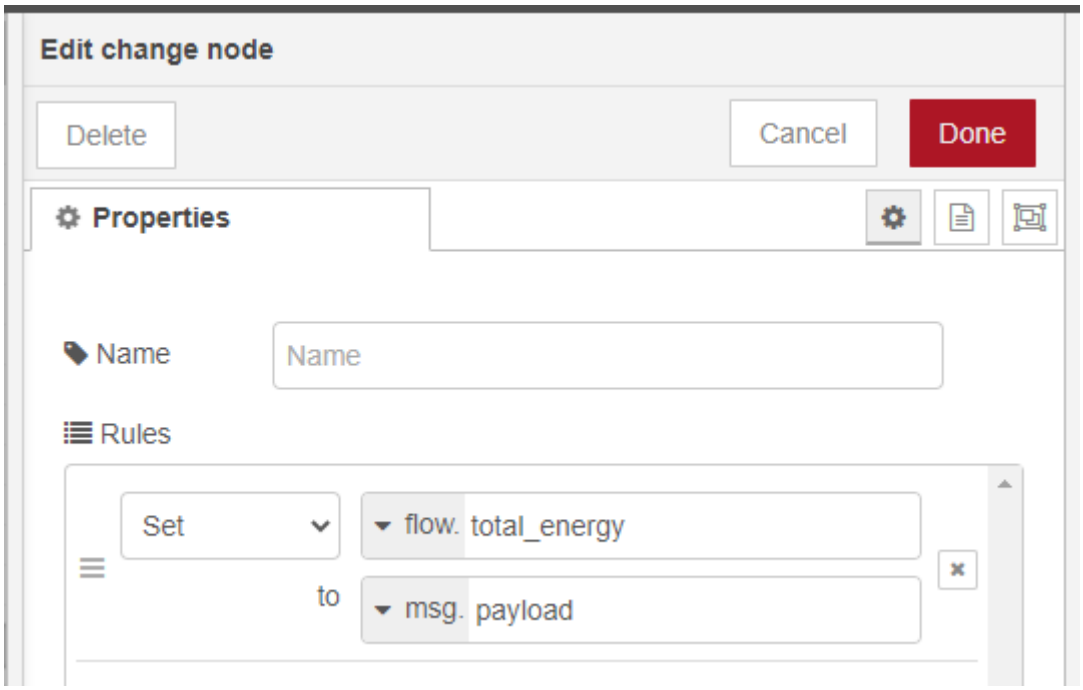

Edit function node
Delete
Properties
Name
instant_power_grid
Setup
Function
1 msg.payload=msg.payload.data.powerflow.grid;
2 return msg;

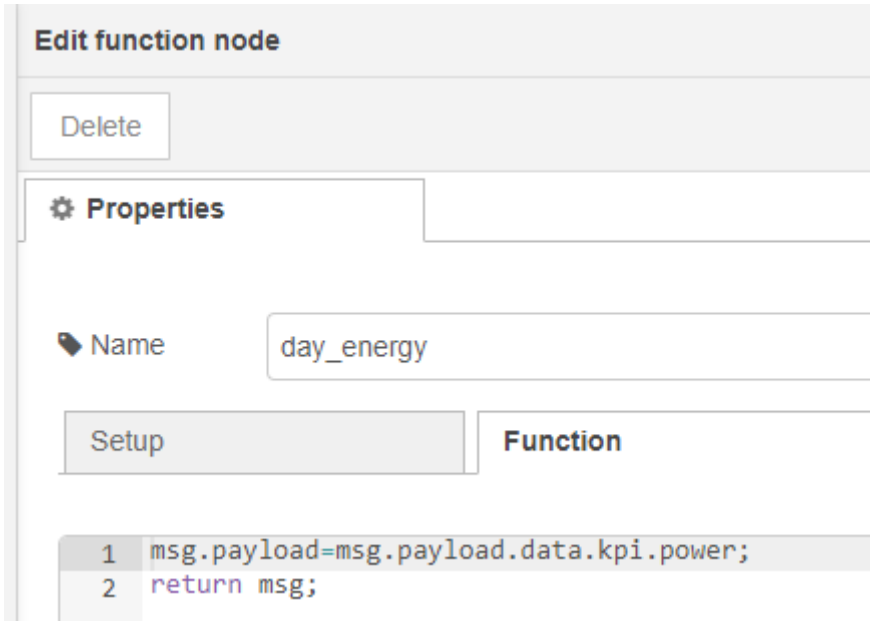

Edit change node
Delete
Cancel
Done
Properties
Name
Name
Rules
Set
msg.
payload
to
$substringBefore(payload,'(')

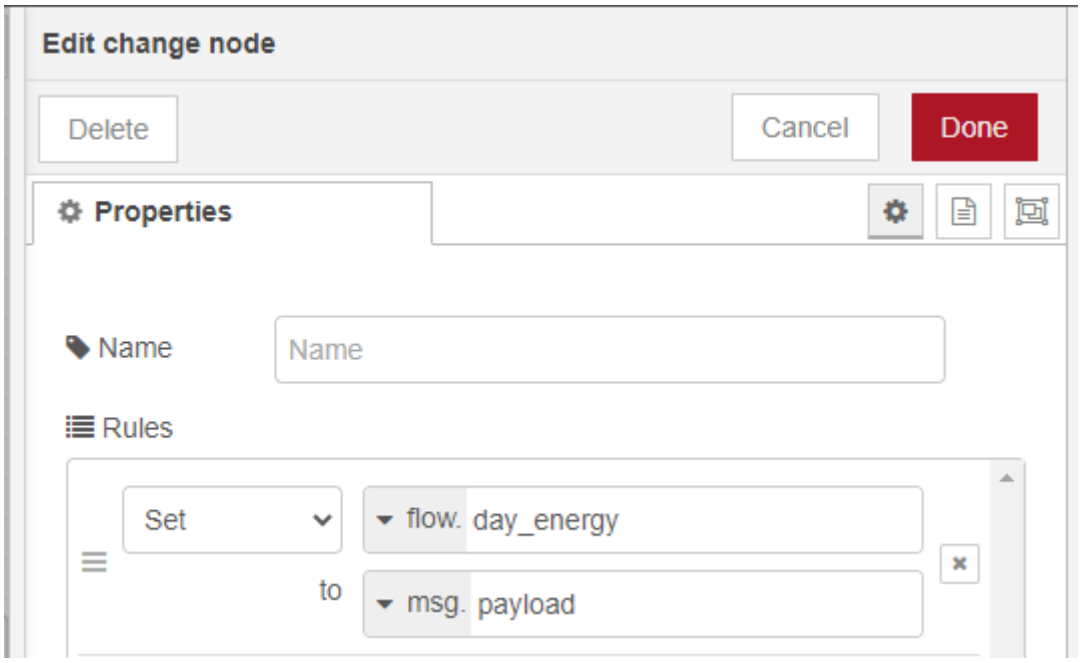

Edit change node
Delete
Cancel
Done
Properties
Name
Name
Rules
Set
msg.
payload
to
$number(payload)

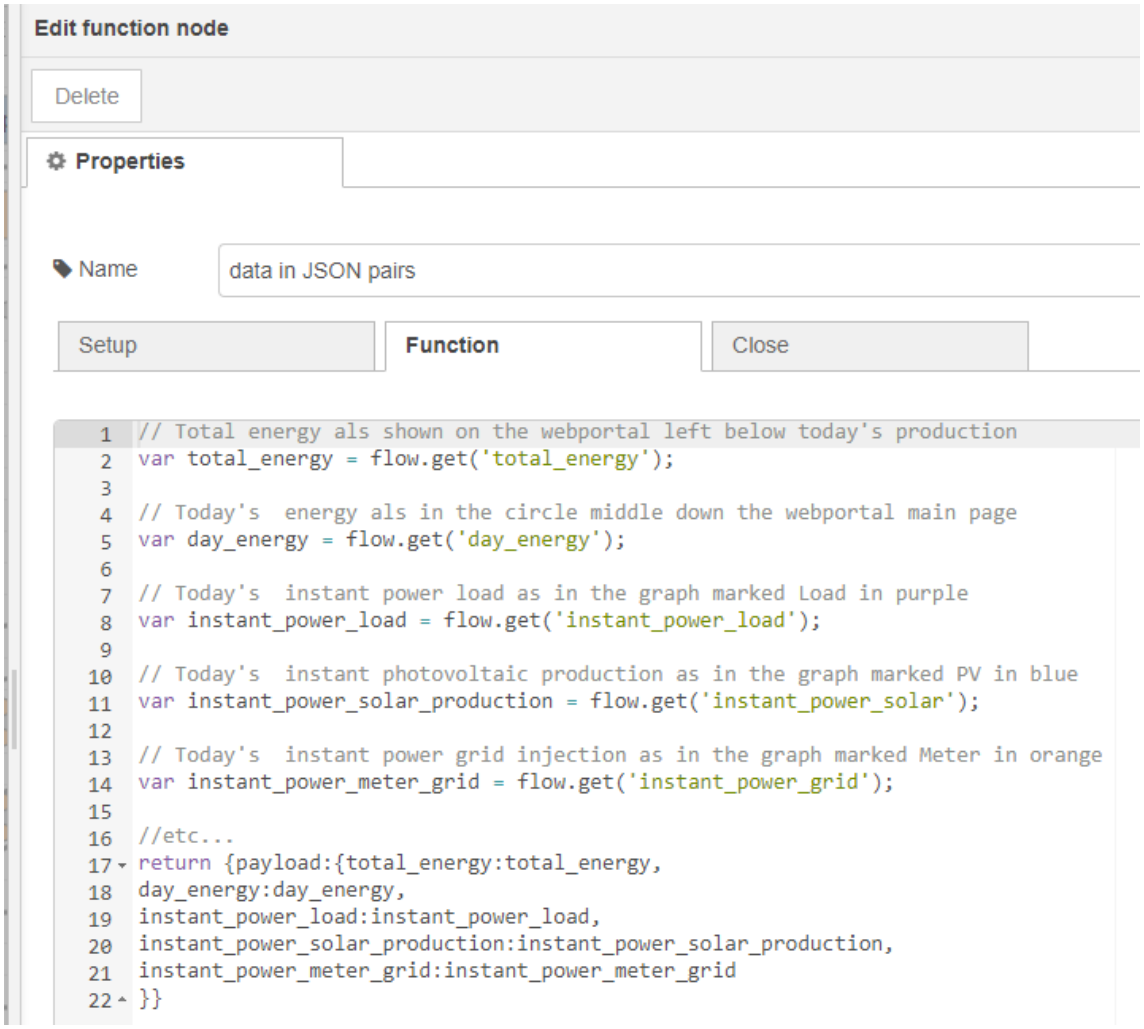

Edit change node
Delete
Cancel
Done
Properties
Name
Name
Rules
Set
flow.
instant_power_grid
to
msg.
payload

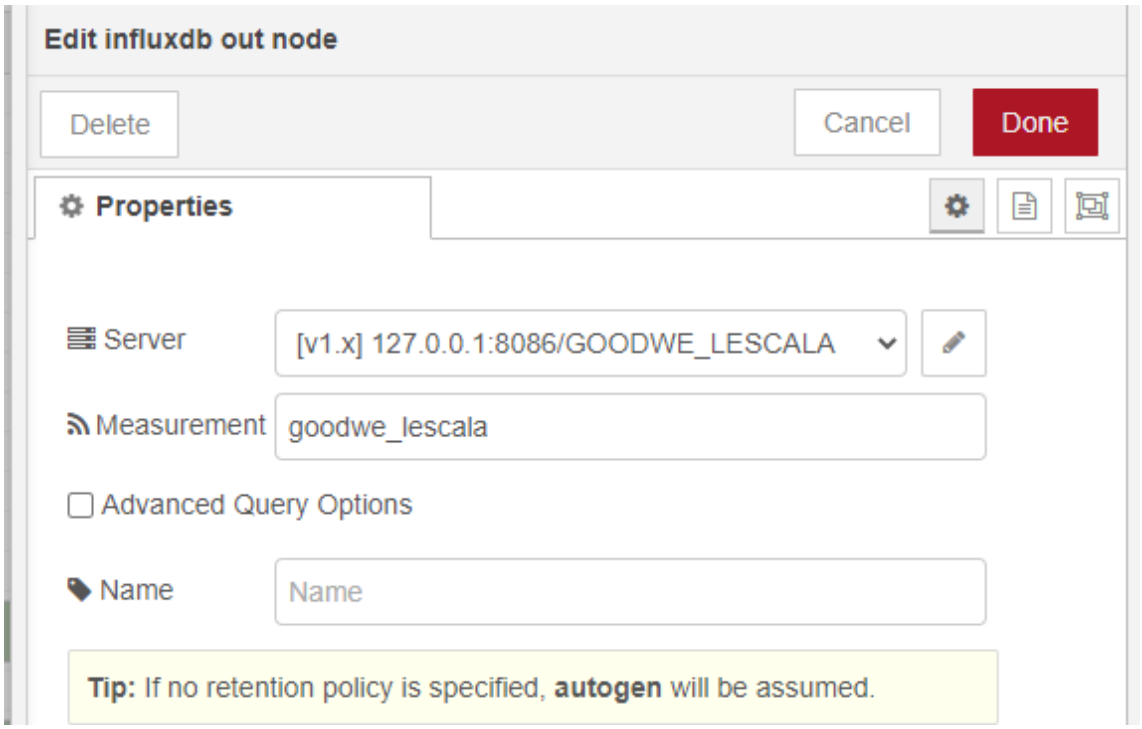

Edit function node
Delete
Properties
Name
total_energy
Setup
Function
Close
1
2
msg.payload=msg.payload.data.kpi.total_power;
return msg;

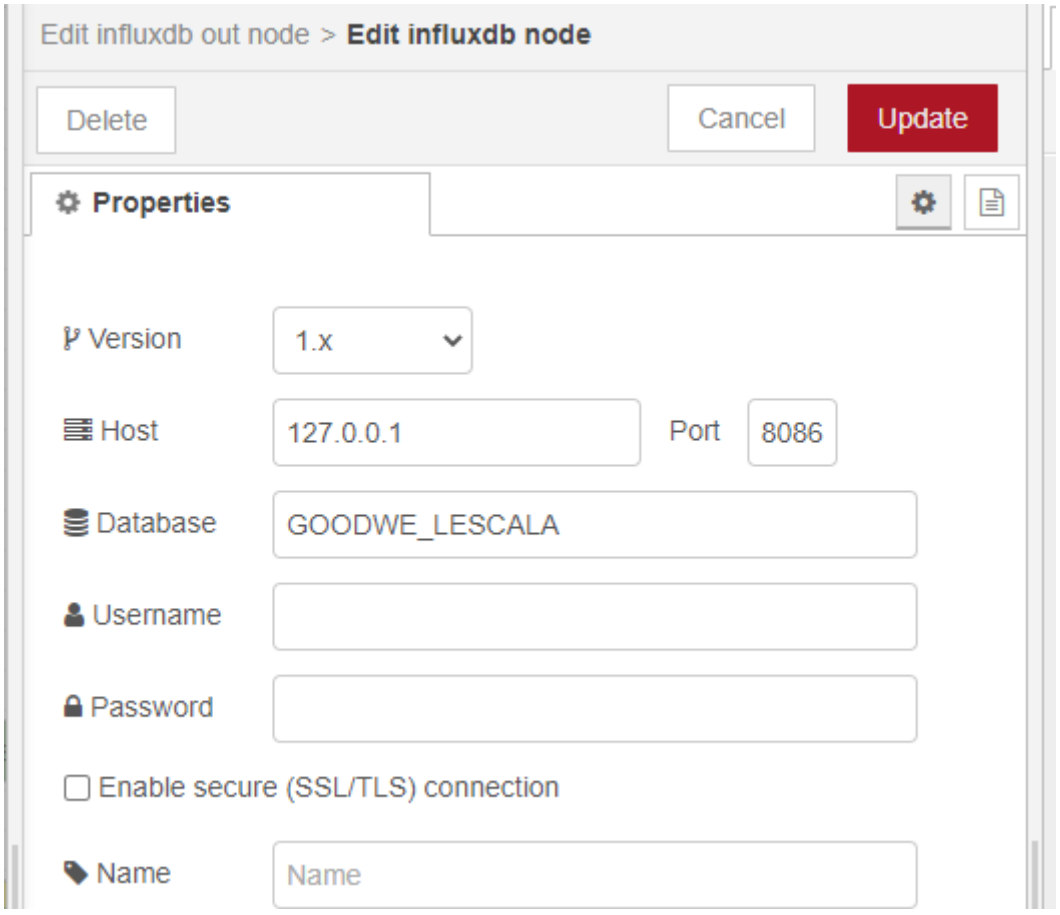

Edit change node
Delete
Cancel
Done
Properties
Name
Name
Rules
Set
flow.
total_energy
to
msg.
payload


Edit function node
Delete
Properties
Name
day_energy
Setup
Function
1
msg.payload=msg.payload.data.kpi.power;
2
return msg;


Edit change node
Delete
Cancel
Done
Properties
Name
Name
Rules
Set
flow.
day_energy
to
msg.
payload


Edit function node
Delete
Properties
Name
data in JSON pairs
Setup
Function
Close
// Total energy als shown on the webportal left below today's production
var total_energy = flow.get('total_energy');
// Today's energy als in the circle middle down the webportal main page
var day_energy = flow.get('day_energy');
// Today's instant power load as in the graph marked Load in purple
var instant_power_load = flow.get('instant_power_load');
// Today's instant photovoltaic production as in the graph marked PV in blue
var instant_power_solar_production = flow.get('instant_power_solar');
// Today's instant power grid injection as in the graph marked Meter in orange
var instant_power_meter_grid = flow.get('instant_power_grid');
//etc...
return {payload:{total_energy:total_energy,
day_energy:day_energy,
instant_power_load:instant_power_load,
instant_power_solar_production:instant_power_solar_production,
instant_power_meter_grid:instant_power_meter_grid
}}


Edit influxdb out node
Delete
Cancel
Done
Properties
Server
[v1.x] 127.0.0.1:8086/GOODWE_LESCALA
Measurement
goodwe_lescala
Advanced Query Options
Name
Name
Tip: If no retention policy is specified, autogen will be assumed.


Edit influxdb out node > Edit influxdb node
Delete
Cancel
Update
Properties
Version
1.x
Host
127.0.0.1
Port
8086
Database
GOODWE_LESCALA
Username
Password
Enable secure (SSL/TLS) connection
Name
Name

```
Linux KBXNEXDOM01 4.19.0-12-amd64 #1 SMP Debian 4.19.152-1 (2020-10-18) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Apr 12 16:31:31 2021 from 192.168.201.17
$ influx
Connected to http://localhost:8086 version 1.6.4
InfluxDB shell version: 1.6.4
> create database GOODWE_LESCALA
> SHOW DATABASES
name: databases
name
----
_internal
DECENTLAB
GOSPI
JORDI
CEMC31
PORT
DECENTLAB_2021
POWER_LESCALA
PRUEBA
FORMACION_SESION3
GOODWE_LESCALA
>
> USE GOODWE_LESCALA
Using database GOODWE_LESCALA
> select * from goodwe_lescala
name: goodwe_lescala
time                day_energy instant_power_load instant_power_meter_grid instant_power_solar_production total_energy
----                ---------- ------------------ ------------------------ ------------------------------ ------------
1618589838810150616 85.1       23912              22342                    1570                           1554
1618589858558744282 85.1       23735              22342                    1393                           1554
>
```