Royal Netherlands Meteorological Institute
Ministry of Infrastructure and Water Management

KNMI.nl Dataloader

Quinten Cabo

Radboud University

Finding a Dataset

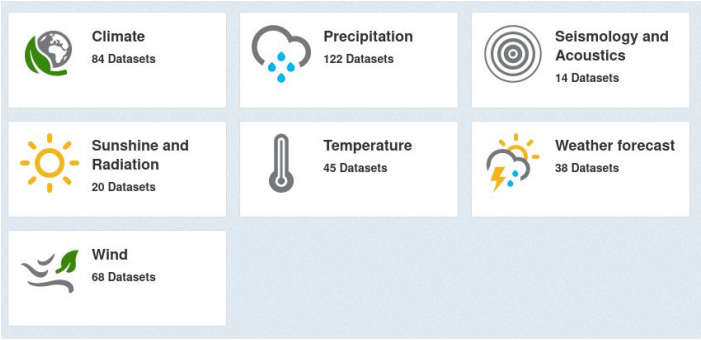
Source of bad datasets?

Finding a dataset

KNMI.nl

https://www.knmi.nl/home

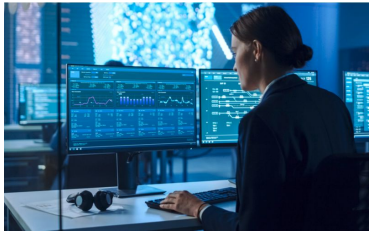
Koninklijk Nederlands Meteorologisch Instituut.

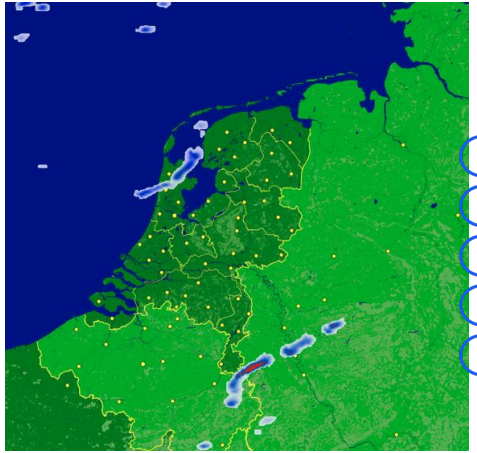


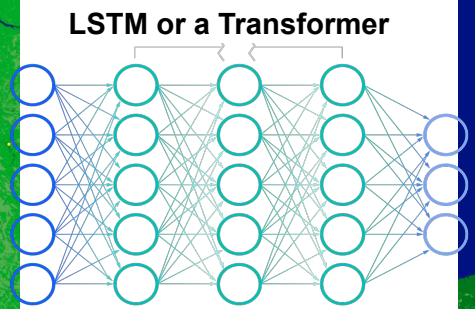
Koninklijk Nederlands
Meteorologisch Instituut
Ministerie van Infrastructuur en Waterstaat

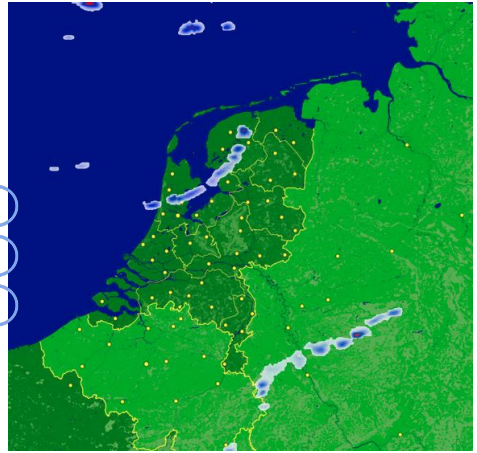
Use case

Weather prediction models









T = 1

T = 2

Finding a dataset

KNMI

https://www.knmi.nl/home

Koninklijk Nederlands Meteorologisch Instituut.

Koninklijk Nederlands

API Catalog

Notification Service

The Notification Service allows you to receive timely updates about events of the KNMI Data Platform.

View Documentation

Request an API key

EDR API (ALPHA)

The *Environmental Data Retrieval* (EDR) REST API. This API allows users to query datasets on specific parameters in a spatio-temporal manner, reducing the need to download entire datasets. The EDR API adheres to *Open Geospatial Consortium* EDR standard, facilitating integration with other geospatial tools and systems.

DISCLAIMER: This pre-release version is still in development. The API may contain bugs, and data integrity is not yet verified.

View Documentation

API Reference (Swagger)

Request an API key

Web Map Service (WMS)

Web service for geospatial data integration in online mapping applications. The web service is compatible with OGC WMS specification versions 1.0.0, 1.1.1, and 1.3.0 (default, unless otherwise requested).

View Documentation

Request an API key

Finding a dataset

KNMI

https://www.knmi.nl/home

Koninklijk Nederlands Meteorologisch Instituut.

Koninklijk Nederlands

API Catalog

Notification Service

The Notification Service allows you to receive timely updates about events of the KNMI Data Platform.

View Documentation

Request an API key

EDR API (ALPHA)

The *Environmental Data Retrieval* (EDR) REST API. This API allows users to query datasets on specific parameters in a spatio-temporal manner, reducing the need to download entire datasets. The EDR API adheres to *Open Geospatial Consortium* EDR standard, facilitating integration with other geospatial tools and systems.

DISCLAIMER: This pre-release version is still in development. The API may contain bugs, and data integrity is not yet verified.

View Documentation

API Reference (Swagger)

Request an API key

Web Map Service (WMS)

Web service for geospatial data integration in online mapping applications. The web service is compatible with OGC WMS specification versions 1.0.0, 1.1.1, and 1.3.0 (default, unless otherwise requested).

View Documentation

Request an API key

Accessing the data

Requesting access to the mqtt stream

Royal Netherlands Meteorological Institute
Ministry of Infrastructure and Water Management

KNMI Developer Portal

News | FAQ | Documentation | API Catalog | Register | Login

Login

Email address

Enter email

Password

Password

Login

Don't have an account ? **Register** here

Forgot password? **Request password reset**

Accessing the data

Requesting access to the mqtt stream

Royal Netherlands
Meteorological Institute
Ministry of Infrastructure and Water Management

KNMI Developer Portal | News | FAQ | Documentation | API Catalog | Register | Login

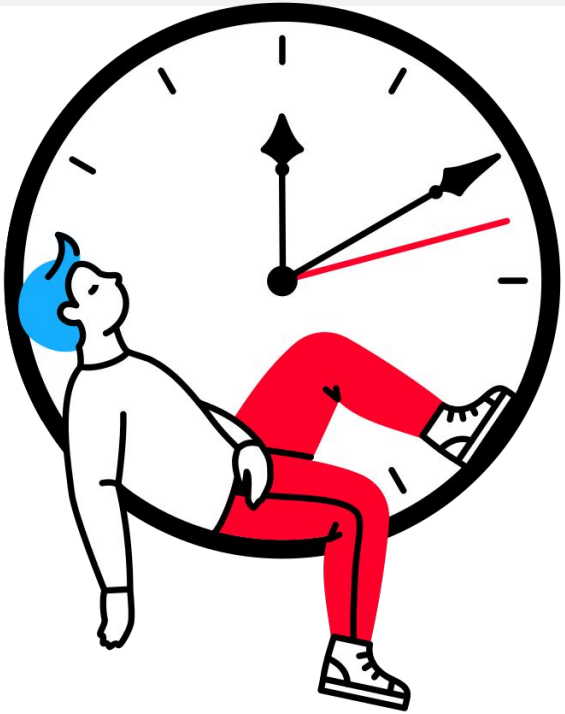
Request a Key

Use case - To help KNMI understand how the API is used.

Request Key

Requesting access

Waiting for a response



Accessing the data

Getting the notification token

Your API Subscriptions

API Key Hash: `8f901bf12c7c456d856dc28f94c1c730`

Accessing the data

Subscribing to the mqtt stream

```python
create database if not exists()
# Version 3.1.1 also supported
PROTOCOL = mqtt client.MQTTv5
logging.basicConfig()
logger = logging.getLogger(__name__)
logger.setLevel("INFO")
def connect mqtt() -> mqtt client:
    def on connect(c: mqtt client, userdata, flags, rc, reason code, props=None):
        logger.info(f"Connected using client ID:{str(c._client_id)}")
        logger.info(f"Session present:{str(flags['session present'])}")
        logger.info(f"Connection result:{str(rc)}")
        # Subscribe here so it is automatically done after disconnect
        subscribe(c, TOPIC)
    client = mqtt_client.Client(paho.mqtt.enums.CallbackAPIVersion.VERSION1,client_id=CLIENT_ID, protocol=PROTOCOL, transport="websockets")
    client.tls set(tls version=ssl.PROTOCOL TLS)
    connect properties = properties.Properties(properties.PacketTypes.CONNECT)
    # Maximum is 3600
    connect properties.SessionExpiryInterval =3600
    # The MQTT username is not used for authentication, only the token
    username = "token"
    client.username pw set(username, TOKEN)
    client.on connect = on connect
    client.connect(host=BROKER_DOMAIN, port=443, keepalive=60, clean_start=False, properties=connect_properties)
    return client
```

```python
def run():
    client = connect mqtt()
    client.enable logger(logger=logger)
    client.loop_forever()
```

Accessing the data

Mqtt stream items

{"specversion": "1.0",
"type": "nl.knmi.dataplatform.file.updated.v1",
"source": "https://dataplatform.knmi.nl", "id":"458081e2-3842-8c0a-b912-85fb93e8c0ed",
"time": "2024-06-05T14:19:36Z", "datacontenttype": "application/json",
"data": {
"datasetName": "Actuele10mindataKNMIstations",
"datasetVersion": "2",
"filename": "KMDS__OPER_P___10M_OBS_L2_202406051410.nc",
"url":"https://api.dataplatform.knmi.nl/open-data/v1/datasets/Actuele10mindataKNMIstations/versions/2/files/KMDS__OPER_P___10M_OBS_L2_202406051410.nc/url"
}
}'

Accessing the data

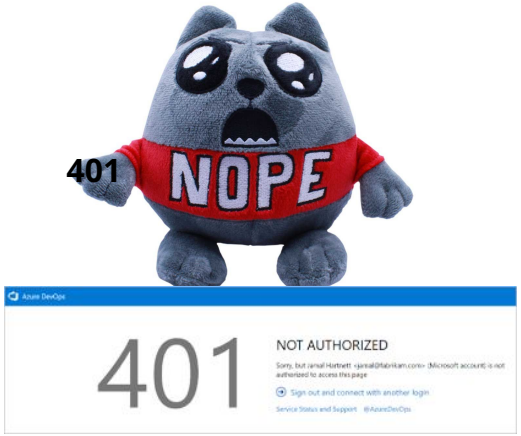
Lets download the file

GET:
https://api.dataplatform.knmi.nl/open-data/v1/datasets/Actuele10mindataKNMIstations/versions/2/files/KMDS__OPER_P___10M_OBS_L2_202406051410.nc/url

Accessing the data

Lets download the file

GET:
https://api.dataplatform.knmi.nl/open-data/v1/datasets/Actuele10mindataKNMIstations/versions/2/files/KMDS__OPER_P___10M_OBS_L2_202406051410.nc/url



Accessing the data

Second KNMI API key

https://www.knmi.nl/home

Koninklijk Nederlands Meteorologisch Instituut.

Koninklijk Nederlands
Meteorologisch Instituut
Ministerie van Infrastructuur en Waterstaat

Open Data API

The Open Data REST API allows you to download files from datasets published on *KNMI Data Platform* - Data Catalog.

View Documentation

API Reference (Swagger)

Request an API key

Accessing the data

Requesting access - logging back in

Royal Netherlands Meteorological Institute
Ministry of Infrastructure and Water Management

KNMI Developer Portal News FAQ Documentation API Catalog Register Login

Login

Email address

Enter email

Password

Password

Login

Don't have an account ? **Register** here

Forgot password? **Request password reset**

Accessing the data

Requesting access again

Royal Netherlands Meteorological Institute
Ministry of Infrastructure and Water Management

KNMI Developer Portal | News | FAQ | Documentation | API Catalog | Register | Login

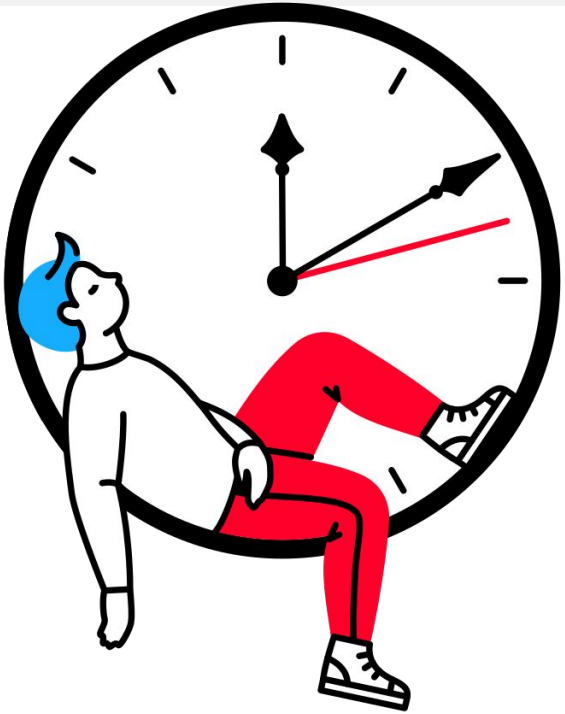
Request a Key

Use case - To help KNMI understand how the API is used.

Request Key

Accessing the data

Waiting for a response again



Accessing the data

Getting the second token

Your API Subscriptions

API Key Hash: `8f901bf12c7c456d856dc28f94c1c730`

Accessing the data

Downloading the data second attempt

GET:
https://api.dataplatform.knmi.nl/open-data/v1/datasets/Actuele10mindataKNMIstations/versions/2/files/KMDS__OPER_P___10M_OBS_L2_202406051410.nc/url

```
headers={"Authorization": f"Bearer {FILE_DOWNLOAD_TOKEN}"}
```

Accessing the data

Downloading the data second attempt

GET:
https://api.dataplatform.knmi.nl/open-data/v1/datasets/Actuele10mindataKNMIstations/versions/2/files/KMDS__OPER_P___10M_OBS_L2_202406051410.nc/url

```
headers={"Authorization": f"Bearer {FILE_DOWNLOAD_TOKEN}"}
```

{“"temporaryDownloadUrl": “**https://amazon.eu-west.supersuperlongurl.temp.kmni.nl**”}

Downloading the data third attempt

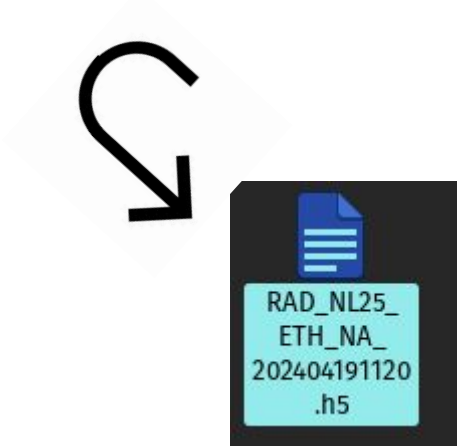
GET: **https://amazon.eu-west.supersuperlongurl.temp.kmni.nl**

(without token)

Downloading the data third attempt

GET: **https://amazon.eu-west.supersuperlongurl.temp.kmni.nl**

(without token)



Recap

Do get data from kmni you only need to

1. Request a token
2. Wait
3. Request another token
4. Wait
5. Listen to the stream with token 1
6. Get link from stream item
7. Request the download link from the stream item link with token 2
8. Download file from this link WITHOUT a token

Then you get an **h5** file…

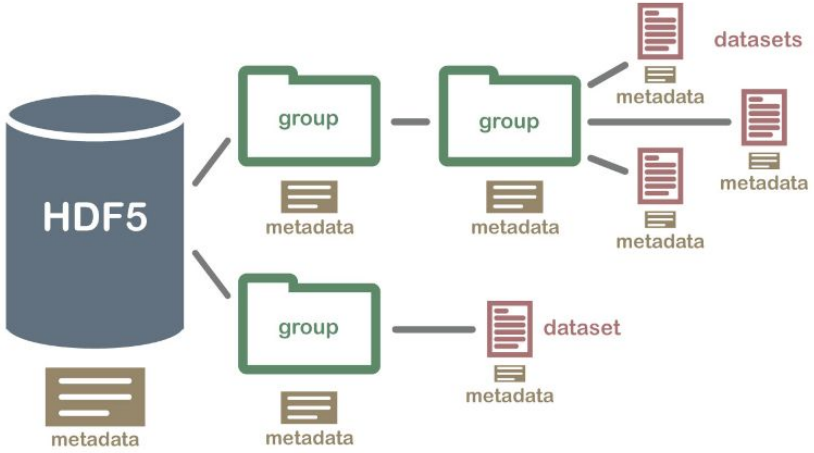
The data itself

Wtf is an h5 file?

The data itself

Wtf is an h5 file?

- Binary zip file with **groups** and **datasets**

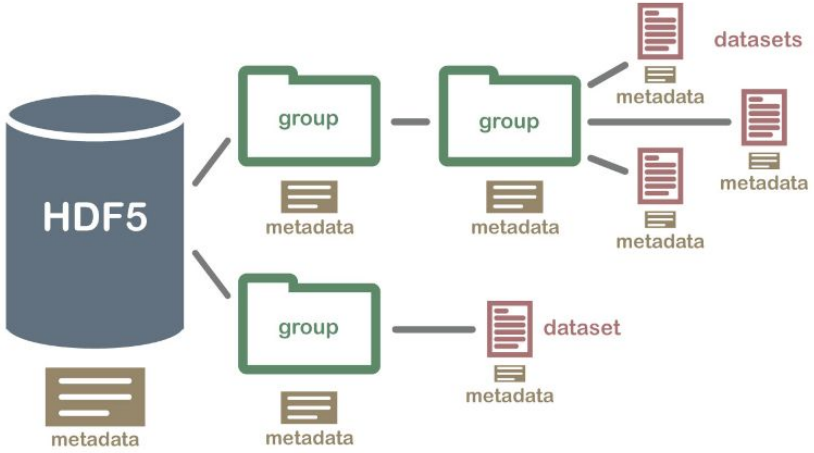


https://www.neonscience.org/resources/learning-hub/tutorials/about-hdf5

The data itself

Wtf is an h5 file?

- Binary zip file with **groups** and **datasets**
- Datasets are homogeneous arrays
- Groups contain groups and data

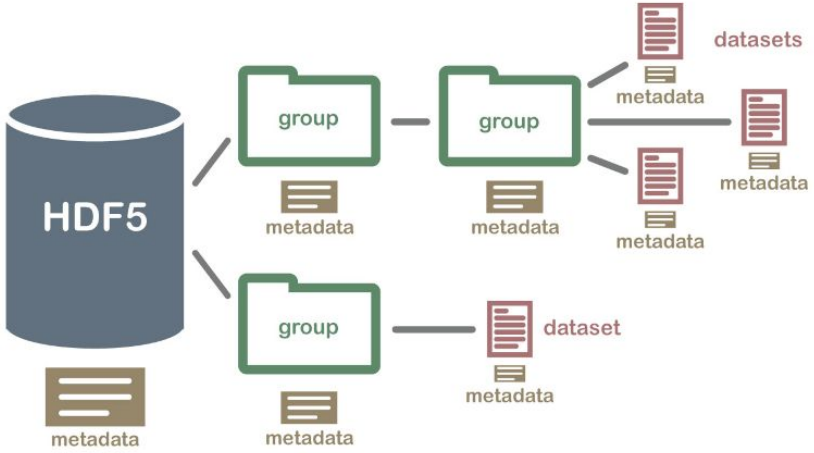


https://www.neonscience.org/resources/learning-hub/tutorials/about-hdf5

The data

Wtf is an h5 file?

- Binary zip file with **groups** and **datasets**
- Datasets are homogeneous arrays
- Groups contain groups and dataset
- Groups and datasets have Metadata



https://www.neonscience.org/resources/learning-hub/tutorials/about-hdf5

The data

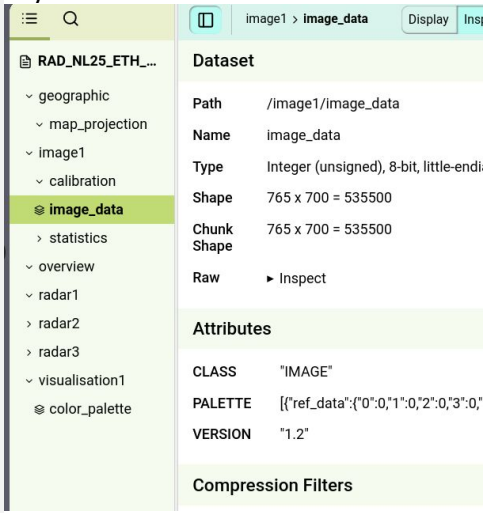
KMNI API update

The data

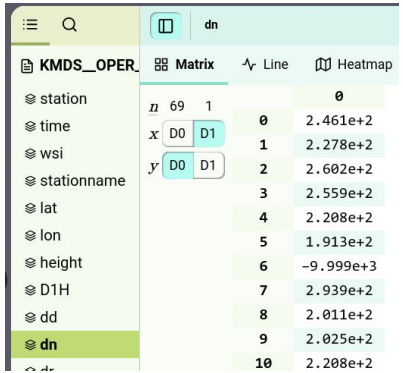
What is an nc file

h5 file:

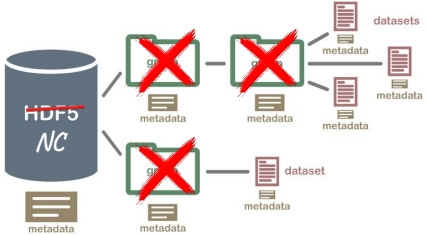
/



nc file instead gives you something like this:



(No groups)



What is IN each nc file?

Each nc file has the measurements of 94 variables for ~69 weather station at a single point in time.

Name	Long name	Shape	Unit	Type
dsd	Wind Direction 10 Min Std Dev with MD	(69, 1)	degree	float64
dx	Wind Direction Sensor 10 Min Maximum with MD	(69, 1)	degree	float64
ff	Wind Speed at 10m 10 Min Average with MD	(69, 1)	m s-1	float64
ffs	Wind Speed Sensor 10 Min Average with MD	(69, 1)	m s-1	float64
fsd	Wind Speed 10 Min Std Dev with MD	(69, 1)	m s-1	float64

Quality

The data quality is actually very good

- Detailed Comments,
- Iso standard ISO 19115
- Clear Units (c, %, ...)
- Missing values (but masked arrays)

Name	Long name	Shape	Unit	Type
dsd	Wind Direction 10 Min Std Dev with MD	(69, 1)	degree	float64
dx	Wind Direction Sensor 10 Min Maximum with MD	(69, 1)	degree	float64
ff	Wind Speed at 10m 10 Min Average with MD	(69, 1)	m s-1	float64
ffs	Wind Speed Sensor 10 Min Average with MD	(69, 1)	m s-1	float64
fsd	Wind Speed 10 Min Std Dev with MD	(69, 1)	m s-1	float64

Quality

The data quality is actually very good

- Detailed Comments,
- Iso standard ISO 19115
- Clear Units (c, %, ...)
- Missing values (but masked arrays)

But

Do get data from kmni you only need to

1. Request a token
2. Wait
3. Request another token
4. Wait
5. Listen to the stream with token 1
6. Get link from stream item
7. Request the download link from the stream item link with token 2
8. Download file from this link WITHOUT a token

Then you get an ~~h5~~ NC file...

Name	Long name	Shape	Unit	Type
dsd	Wind Direction 10 Min Std Dev with MD	(69, 1)	degree	float64
dx	Wind Direction Sensor 10 Min Maximum with MD	(69, 1)	degree	float64
ff	Wind Speed at 10m 10 Min Average with MD	(69, 1)	m s-1	float64
ffs	Wind Speed Sensor 10 Min Average with MD	(69, 1)	m s-1	float64
fsd	Wind Speed 10 Min Std Dev with MD	(69, 1)	m s-1	float64

Easiest path to train test split?

Serving

DX

Easiest path to train test split?

Combining multiple nc files?

data loader

Serving

Dataset interface

•[1]:

```
KMNIdataset(
    root="data",
    train=True,
    download=True,
    after="2024-01-05T01:00:00.000Z",
    until="2024-06-05T16:25:14.948Z",
    stations=("06201", "06203", "06204","06205","06207","06208","06209","06211")
    variables=("fxs", "h", "n2", "p0", "Q1H", "rg")
)
```

Serving

Dataset interface defaults

Just load everything defaults

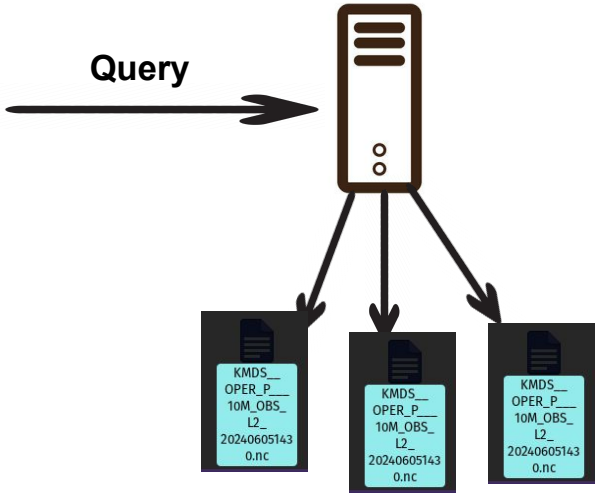
•[1]:

```
KMNIdataset(
    root="data",
    train=True,
    download=True,
)
```

Serving

Querying the nc files directly?

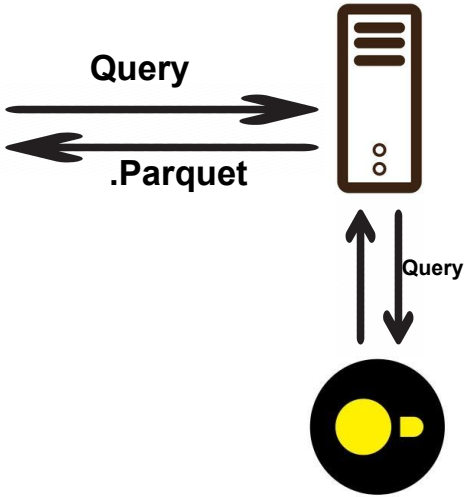
```
▸[1]:

KMNIdataset(
    root="data",
    train=True,
    download=True,
    after="2024-01-05T01:00:00.000Z",
    until="2024-06-05T16:25:14.948Z",
    stations=("06201", "06203", "06204","06205","06207","06208","06209","06211")
    variables=("fxs", "h", "n2", "p0", "Q1H", "rg")
)
```



Querying the data from a database

[1]:

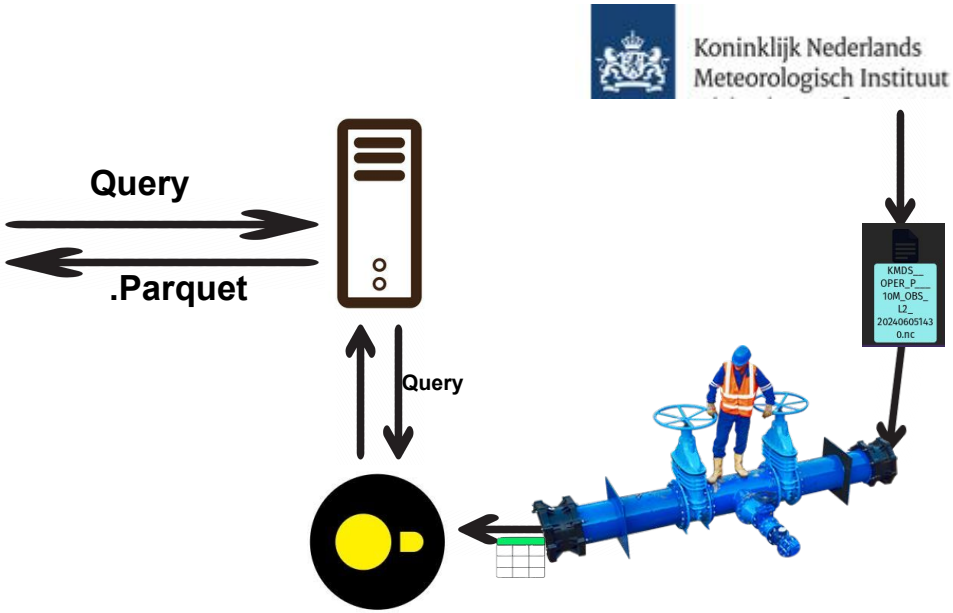
```
KMNIdataset(
    root="data",
    train=True,
    download=True,
    after="2024-01-05T01:00:00.000Z",
    until="2024-06-05T16:25:14.948Z",
    stations=("06201", "06203", "06204","06205","06207","06208","06209","06211")
    variables=("fxs", "h", "n2", "p0", "Q1H", "rg")
)
```



Wrangling

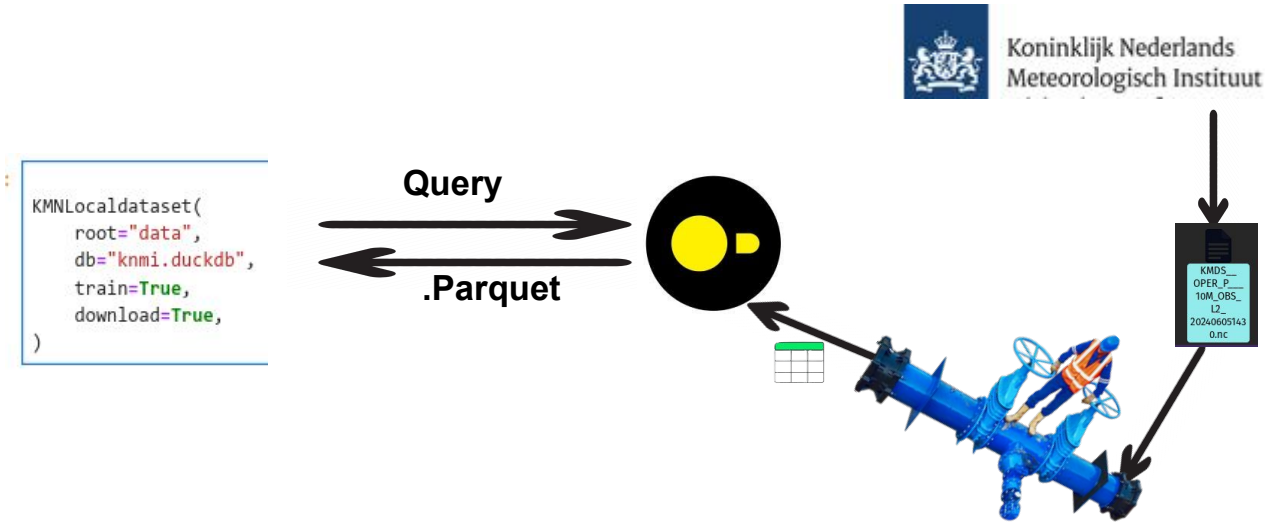
Pipeline to convert nc files into rows

```
▸[1]:

KMNIdataset(
    root="data",
    train=True,
    download=True,
    after="2024-01-05T01:00:00.000Z",
    until="2024-06-05T16:25:14.948Z",
    stations=("06201", "06203", "06204","06205","06207","06208","06209","06211")
    variables=("fxs", "h", "n2", "p0", "Q1H", "rg")
)
```



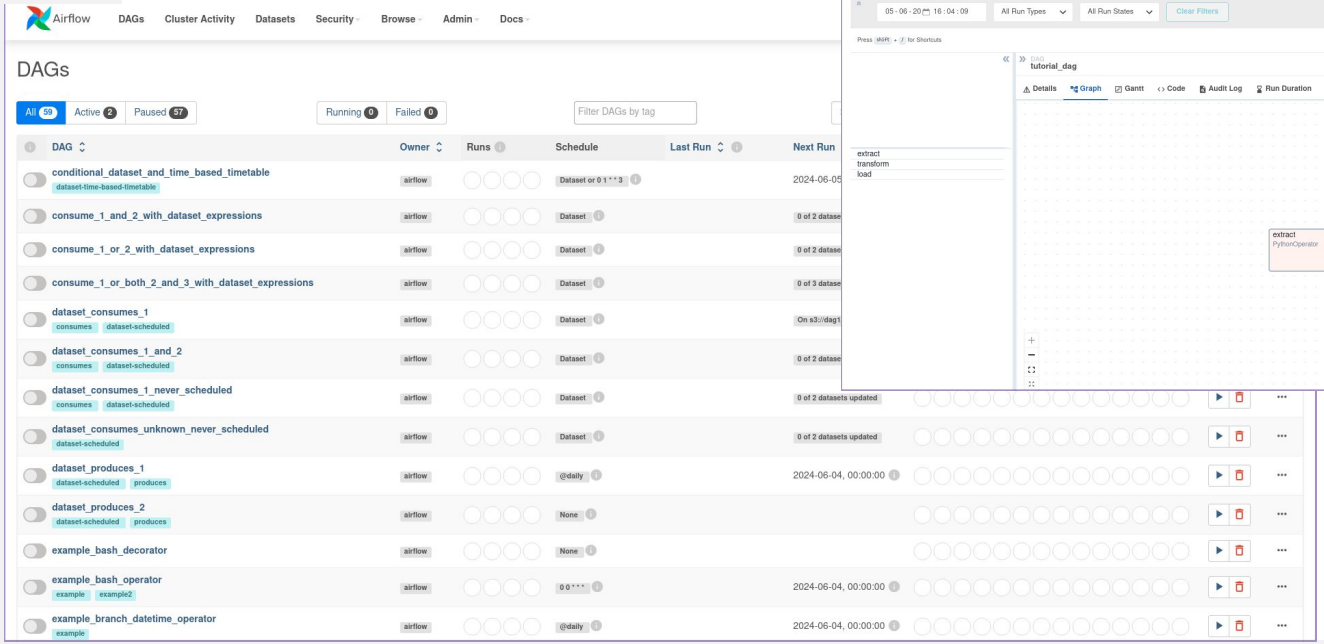
Serving

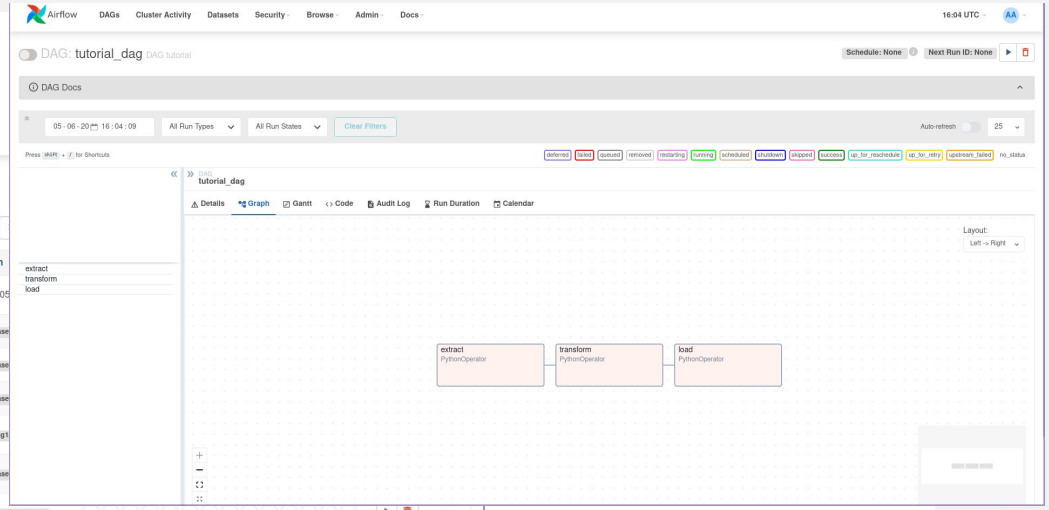
Pipeline to convert nc files into rows LOCALLY



Airflow

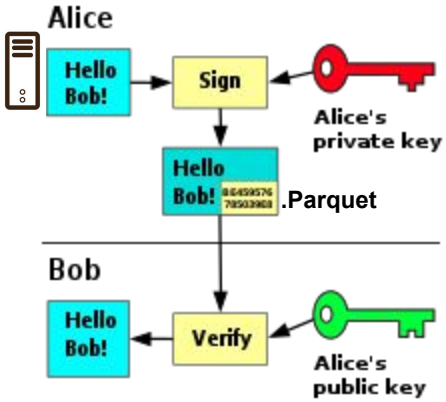
Cool but too complex





One more thing

Integrity

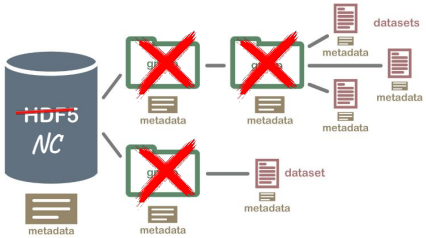


Questions

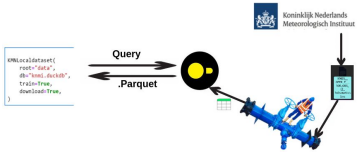
Do get data from kmni you only need to

1. Request a token
2. Wait
3. Request another token
4. Wait
5. Listen to the stream with token 1
6. Get link from stream item
7. Request the download link from the stream item link with token 2
8. Download file from this link WITHOUT a token

Then you get an nc file...



Koninklijk Nederlands Meteorologisch Instituut
Ministerie van Infrastructuur en Waterstaat



data loader

Wrangling
Pipeline to convert nc files into rows