# Knowledge Hub

## Python Library User Guide

**Release R2023.07.00.00**

August 2023

# Contents

# Revision History for This Document

| Version Number | Revised On | Revision Description | Tracker Reference |
|---|---|---|---|
| 1.0 | August 2023 | Created the document for Release R2023.07.00.00. | ADO 228927 |

# Introduction

This document provides information to end users for using the o9's Knowledge Hub Python Library.

## Audience

This document is intended for the users who are interested in using the Knowledge Hub library on tenant and local system to retrieve Knowledge Hub data and implement these external drivers in their forecasting process. Moreover, data scientists can easily use this library to incorporate these external drivers into their ML models.

# Overview

Use the KHub python library to retrieve Knowledge Hub market drivers data as a data frame.

## Business Requirement

The Knowledge Hub Library allows the user to access a wide range of data streams data via Knowledge hub (K-Hub) API's. The user can use this package to not just view the K-Hub catalog, but also download the data. The catalog lists the dimensions and measures stored in the cube. Data downloaded by the library is available to the user as a dataframe.

## Accessing KHub Library

Users can download the python library zip file from this path and follow the steps mentioned in the Installation section.

# Installation

The library can be installed using a code package or directly from the Knowledge Hub repo. This library has package dependencies for openpyxl and azure-storage-blob, as a result the user needs to make sure that these packages are installed on their tenant or local system.

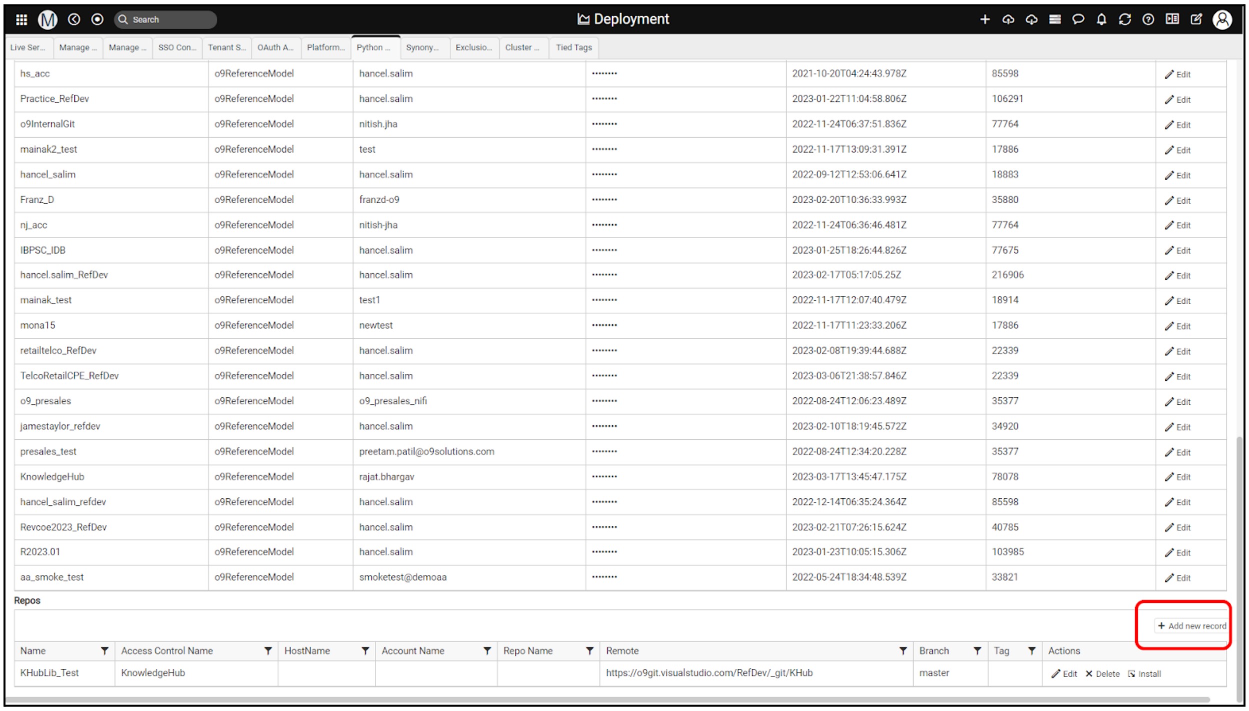
## Using Code Package

The library is fully compatible for use in DSML workspace. Steps to take

- Create a folder in the DSML workspace.
- Copy the library code (init.py file, to be obtained from Knowledge Hub team) into the folder

## Using the Knowledge Hub Git repository

The user needs to add a new record in Repos (Navigate to Deployment > Python Environment > Repos), scroll to the bottom and click on “Add new record”.



| | | | | | | |
|---|---|---|---|---|---|---|
| hs_acc | o9ReferenceModel | hancel.salim | ••••••••• | 2021-10-20T04:24:43.978Z | 85598 | Edit |
| Practice_RefDev | o9ReferenceModel | hancel.salim | ••••••••• | 2023-01-22T11:04:58.806Z | 106291 | Edit |
| o9InternalGit | o9ReferenceModel | nitish.jha | ••••••••• | 2022-11-24T06:37:51.836Z | 77764 | Edit |
| mainak2_test | o9ReferenceModel | test | ••••••••• | 2022-11-17T13:09:31.391Z | 17886 | Edit |
| hancel_salim | o9ReferenceModel | hancel.salim | ••••••••• | 2022-09-12T12:53:06.641Z | 18883 | Edit |
| Franz_D | o9ReferenceModel | franzd-o9 | ••••••••• | 2023-02-20T10:36:33.993Z | 35880 | Edit |
| nj_acc | o9ReferenceModel | nitish-jha | ••••••••• | 2022-11-24T06:36:46.481Z | 77764 | Edit |
| IBPSC_IDB | o9ReferenceModel | hancel.salim | ••••••••• | 2023-01-25T18:26:44.826Z | 77675 | Edit |
| hancel.salim_RefDev | o9ReferenceModel | hancel.salim | ••••••••• | 2023-02-17T05:17:05.25Z | 216906 | Edit |
| mainak_test | o9ReferenceModel | test1 | ••••••••• | 2022-11-17T12:07:40.479Z | 18914 | Edit |
| mona15 | o9ReferenceModel | newtest | ••••••••• | 2022-11-17T11:23:33.206Z | 17886 | Edit |
| retailtelco_RefDev | o9ReferenceModel | hancel.salim | ••••••••• | 2023-02-08T19:39:44.688Z | 22339 | Edit |
| TelcoRetailCPE_RefDev | o9ReferenceModel | hancel.salim | ••••••••• | 2023-03-06T21:38:57.846Z | 22339 | Edit |
| o9_presales | o9ReferenceModel | o9_presales_nifi | ••••••••• | 2022-08-24T12:06:23.489Z | 35377 | Edit |
| jamestaylor_refdev | o9ReferenceModel | hancel.salim | ••••••••• | 2023-02-10T18:19:45.572Z | 34920 | Edit |
| presales_test | o9ReferenceModel | preetam.patil@o9solutions.com | ••••••••• | 2022-08-24T12:34:20.228Z | 35377 | Edit |
| KnowledgeHub | o9ReferenceModel | rajat.bhargav | ••••••••• | 2023-03-17T13:45:47.175Z | 78078 | Edit |
| hancel_salim_refdev | o9ReferenceModel | hancel.salim | ••••••••• | 2022-12-14T06:35:24.364Z | 85598 | Edit |
| Revcoe2023_RefDev | o9ReferenceModel | hancel.salim | ••••••••• | 2023-02-21T07:26:15.624Z | 40785 | Edit |
| R2023.01 | o9ReferenceModel | hancel.salim | ••••••••• | 2023-01-23T10:05:15.306Z | 103985 | Edit |
| aa_smoke_test | o9ReferenceModel | smoketest@demoaa | ••••••••• | 2022-05-24T18:34:48.539Z | 33821 | Edit |

Repos

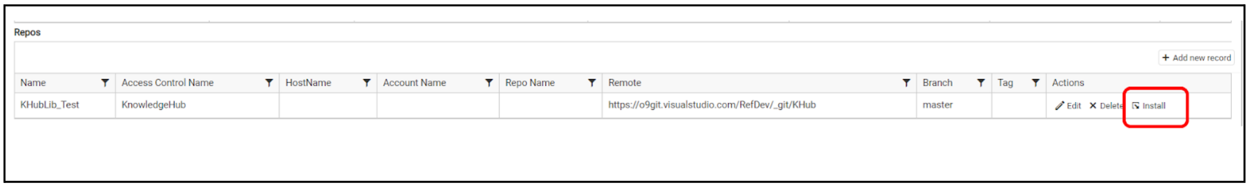
+ Add new record

| Name | Access Control Name | HostName | Account Name | Repo Name | Remote | Branch | Tag | Actions |
|---|---|---|---|---|---|---|---|---|
| KHubLib_Test | KnowledgeHub | | | | https://o9git.visualstudio.com/RefDev/_git/KHub | master | | Edit ✕ Delete Install |

After clicking on “Add new record”, user needs to specify fields as mentioned in the screenshot below

- Name: “User-specific”
- Access Control Name: Select KnowledgeHub from the list
- Remote: https://o9git.visualstudio.com/RefDev/_git/KHub
- Branch: Master

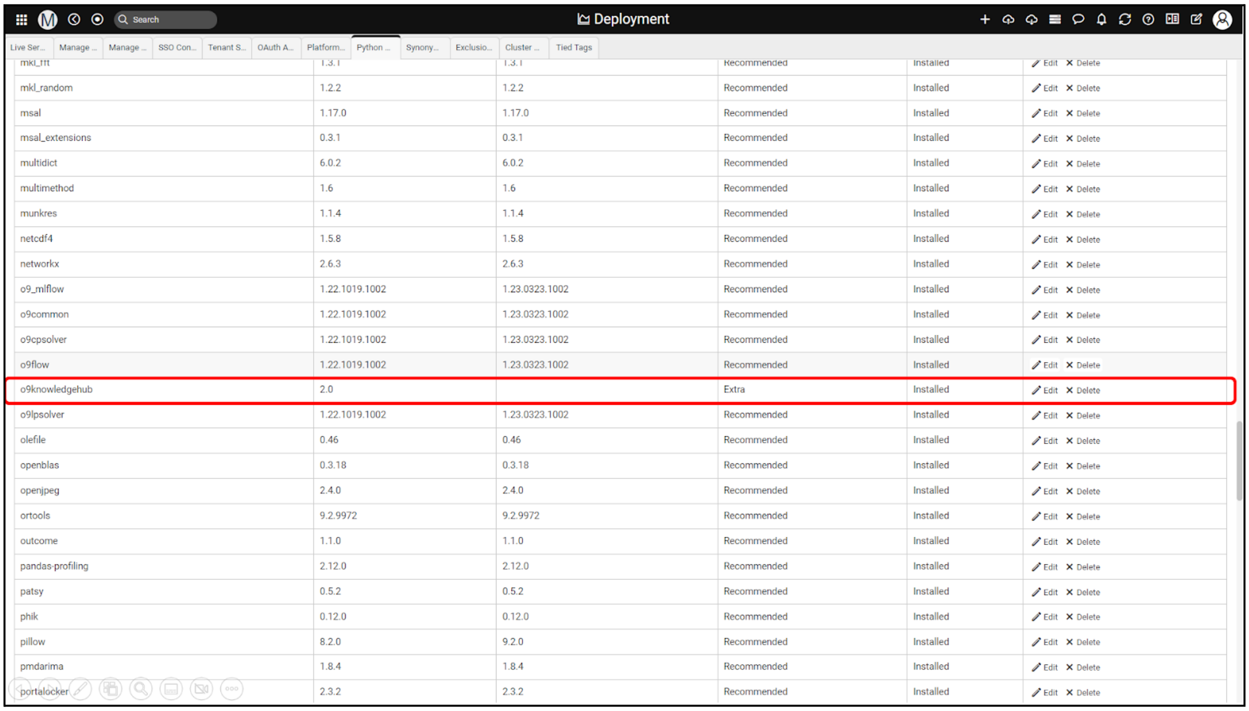
After specifying the details, the user should click “update” which will allow the user to install the Knowledge Hub library package on user’s tenant.


Repos

+ Add new record

| Name | Access Control Name | HostName | Account Name | Repo Name | Remote | Branch | Tag | Actions |
|---|---|---|---|---|---|---|---|---|
| KHubLib_Test | KnowledgeHub | | | | https://o9git.visualstudio.com/RefDev/_git/KHub | master | | Edit ✕ Delete Install |

After the successful Installation, the user can see the o9knowledgehub installed under Additional packages.



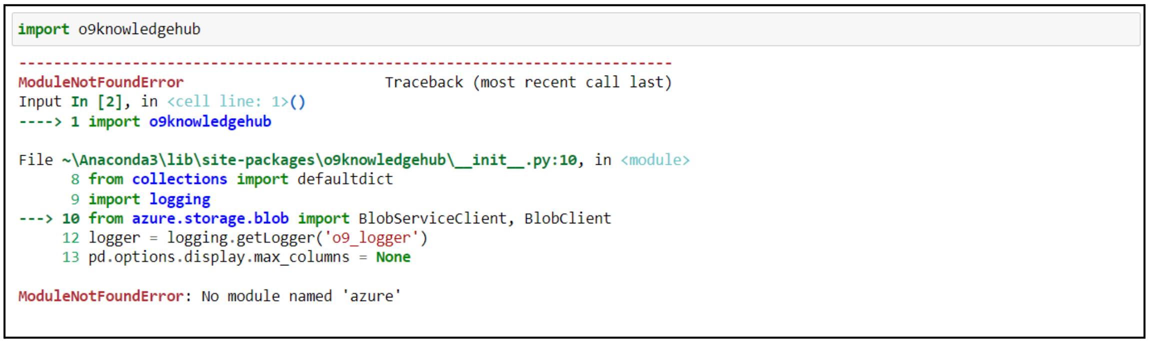
| | | | | | |
|---|---|---|---|---|---|
| mkl_fft | 1.3.1 | 1.3.1 | Recommended | Installed | Edit ✕ Delete |
| mkl_random | 1.2.2 | 1.2.2 | Recommended | Installed | Edit ✕ Delete |
| msal | 1.17.0 | 1.17.0 | Recommended | Installed | Edit ✕ Delete |
| msal_extensions | 0.3.1 | 0.3.1 | Recommended | Installed | Edit ✕ Delete |
| multidict | 6.0.2 | 6.0.2 | Recommended | Installed | Edit ✕ Delete |
| multimethod | 1.6 | 1.6 | Recommended | Installed | Edit ✕ Delete |
| munkres | 1.1.4 | 1.1.4 | Recommended | Installed | Edit ✕ Delete |
| netcdf4 | 1.5.8 | 1.5.8 | Recommended | Installed | Edit ✕ Delete |
| networkx | 2.6.3 | 2.6.3 | Recommended | Installed | Edit ✕ Delete |
| o9_mlflow | 1.22.1019.1002 | 1.23.0323.1002 | Recommended | Installed | Edit ✕ Delete |
| o9common | 1.22.1019.1002 | 1.23.0323.1002 | Recommended | Installed | Edit ✕ Delete |
| o9cpsolver | 1.22.1019.1002 | 1.23.0323.1002 | Recommended | Installed | Edit ✕ Delete |
| o9flow | 1.22.1019.1002 | 1.23.0323.1002 | Recommended | Installed | Edit ✕ Delete |
| o9knowledgehub | 2.0 | | Extra | Installed | Edit ✕ Delete |
| o9lpsolver | 1.22.1019.1002 | 1.23.0323.1002 | Recommended | Installed | Edit ✕ Delete |
| olefile | 0.46 | 0.46 | Recommended | Installed | Edit ✕ Delete |
| openblas | 0.3.18 | 0.3.18 | Recommended | Installed | Edit ✕ Delete |
| openjpeg | 2.4.0 | 2.4.0 | Recommended | Installed | Edit ✕ Delete |
| ortools | 9.2.9972 | 9.2.9972 | Recommended | Installed | Edit ✕ Delete |
| outcome | 1.1.0 | 1.1.0 | Recommended | Installed | Edit ✕ Delete |
| pandas-profiling | 2.12.0 | 2.12.0 | Recommended | Installed | Edit ✕ Delete |
| patsy | 0.5.2 | 0.5.2 | Recommended | Installed | Edit ✕ Delete |
| phik | 0.12.0 | 0.12.0 | Recommended | Installed | Edit ✕ Delete |
| pillow | 8.2.0 | 9.2.0 | Recommended | Installed | Edit ✕ Delete |
| pmdarima | 1.8.4 | 1.8.4 | Recommended | Installed | Edit ✕ Delete |
| portalocker | 2.3.2 | 2.3.2 | Recommended | Installed | Edit ✕ Delete |

For more information, please visit this LINK on Platform Wiki.

## On your local machine

If the user wants to install the library in their local computer, they need to download the KHub library package that can be accessed from this link, and follow the steps mentioned below:

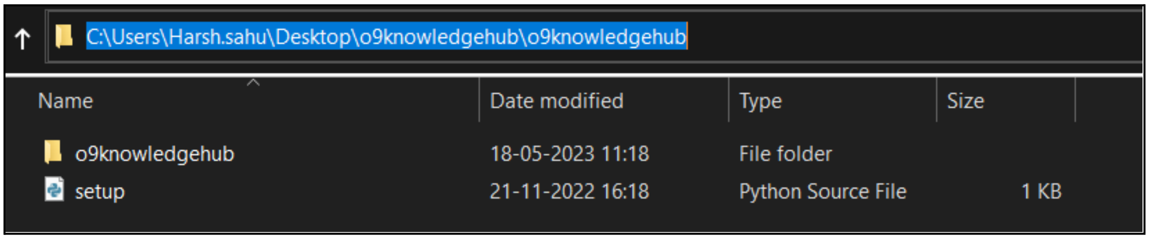
- Knowledge Hub library is bound to package dependencies, hence please install openpyxl and azure-storage-blob in your respective Environment. If the concerned package are not installed, you might face an Error similar to the screenshot below.



```
import o9knowledgehub
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Input In [2], in <cell line: 1>()
----> 1 import o9knowledgehub

File ~\Anaconda3\lib\site-packages\o9knowledgehub\__init__.py:10, in <module>
      8 from collections import defaultdict
      9 import logging
---> 10 from azure.storage.blob import BlobServiceClient, BlobClient
     12 logger = logging.getLogger('o9_logger')
     13 pd.options.display.max_columns = None

ModuleNotFoundError: No module named 'azure'
```

- Extract the zip file named o9knowledgehub, it will create a folder named o9knowledgehub and under that folder, the user can access a setup.py file and o9knowledghub folder (as shown in the screenshot below.



C:\Users\Harsh.sahu\Desktop\o9knowledgehub\o9knowledgehub

| Name | Date modified | Type | Size |
|---|---|---|---|
| o9knowledgehub | 18-05-2023 11:18 | File folder | |
| setup | 21-11-2022 16:18 | Python Source File | 1 KB |

- Copy these 2 items and place them in a local folder in your system. As shown in the above screenshot, the files are placed at “C:\Users\Harsh.sahu\Desktop\o9knowledgehub\o9knowledgehub”

- Run this command **pip install -U “your_folder_path”** (the path to the folder where Setup.py and o9knowledgehub are placed)

# Classes and methods

After uploading the K-Hub Library in user created folder in Jupyter Notebook, the user can import the library by using command - **import o9knowledgehub**

After the library has been imported, the user can create an object and provide a valid API key as a parameter while creating the object. In the below example we have created an object khub. Example: **khub = o9knowledgehub.o9KnowledgeHub (‘Your_API_KEY')**

The Knowledge Hub library supports multiple methods to download data as listed below:

- get_catalog()
- get_measuregrp()
- get_dim()
- get_translated()
- get_downloaded()
- get_subscription()

Further details about these methods are explained below:

**get_catalog**: This method will return the data catalog and offerings by the Knowledge Hub, wherein the user can filter the list of data streams as per Category and Country.
parameter: filters (optional), caching (optional)
return type: A pandas DataFrame containing catalog data.

**get_measuregrp**: This method will return the data across all measures available in the specified measure group(s) in a dataframe.
parameter: measure group name, filters (optional), caching (optional)
return type: A pandas DataFrame containing measure group data.

**get_dim**: This method allows the user to get the master data across Knowledge Hub dimensions.
parameter: dimension name, filters (optional), caching (optional)
return type: A pandas DataFrame containing dimension data.

**get_translated**: Knowledge Hub offers a wide range of data streams categorized across various measure groups that follows a specific nomenclature which could be difficult for a user to comprehend. Hence, this method allows the user to see the translation of the requested data.
Parameter: dataframe and measure group name

**et_downloaded**: This method allows the user to download the data and metadata in an excel workbook.
Parameter: dataframe and measure group name

**get_subscription**: This method will return the dataframe comprising the list of measures the user has subscribed for.

## Parameters' datatype

Knowledge Hub library uses a list of parameters that can be passed with a method/function to request a specific set of data. Below mentioned parameters can be used.

- filters = dictionary where key is a string and value is a list of strings
- measure group name = string
- dimension name = string
- caching = Boolean

# Filters

The Knowledge Hub library capitalizes on two types of filters:

- Global level filters.
- local level filters.

## Global Filter

Once you apply a global filter, the filter will persist across all the get_catalog and get_measuregrp method calls unless we apply a local filter. It should be noted that after applying the local filter, it will overwrite the global filter and data will be fetched only based on local filters. However, global filters will persist if user puts a request using a new command.

The user can apply a global filter using "add_filter" method:

**add_filter**: This method will allow the user to add Global Filter

The user can use country, category, and time filters to further trim the catalog and view the knowledge hub data offerings. The library allows the user to put global filter based on various aspects, and these filters depend on the grain of the respective measure group:

- ContentCategory
- Geo (Continent, Country, State, etc.)
- Time (Year, Quarter, etc,)

Example:
**khub.add_filter(filters={'Year':['2020','2021'], 'Continent':["Europe","North America"], ‘ContentCategory’: ["Macroeconomic"]})**

In the above example, global filter is used, from now on all the method calls will only download Macroeconomic data from server where the data is from Europe or North America within the time interval of 2020 to 2021.

Now, if the users call the method,
**cat_data= khub.get_catalog()**

They will only download catalog data where ContentCategory is "Macroeconomic" since we have a global filter on K-Hub object.

And, if the users call the below stated method,
**data = khub.get_measuregrp('RetailSalesMonthly')**

they will get data from "RetailSalesMonthly" for the range of 2020 to 2021 and for the countries in Europe and North America.

## Local Filter

Local level filters are filters that you can apply while calling get_catalog, get_measuregrp and get_dim methods. Once we apply a local filter, it will override any global filters that may have been applied before. However, the global filter will persist on the object if the user generates a new request command.

It should be noted that Data Type for filter parameters should be dictionary. The dictionary keys will be the name of the fields on which you want to apply the filter. Moreover, the user should pass filter values in a list for each filter field.

For using time dimension attributes-based filters on any measure group, the user must provide the range in the filter value two values in the filter value list. (i.e., both Start time & End time should be mentioned in the query)

Example:

**Filters = {'filter_field_1’: ['value1', 'value2', 'value3'], 'filter_field_2': [value5', value6']}**
**data = khub.get_measuregrp ('RetailSalesMonthly', filters = {'Year': ['2018','2022'], 'Continent': ["Europe", "North America"]})**

This is a local filter applied to get_measuregrp method. It will filter data from the measure group "RetailSalesMonthly" and return a Dataframe containing data from 2018 to 2022 in Europe and North America.

**data = khub.get_dim ('Geo', filters = {'Continent': ["Asia"]})**
This is a local filter applied to get_dim method and it will return a dataframe with a flattened list of geo dimension hierarchy containing data for the continent of Asia.

## Higher level Filtering

The library allows the user to filter the data at a higher level of granularity. For example, if a measure is grained at City and Day, then for the Geo dimension, Continent & Country attributes can be used for filtering. For time filtering, data can be fetched using Month, Year, and other higher-level attributes.

Example:

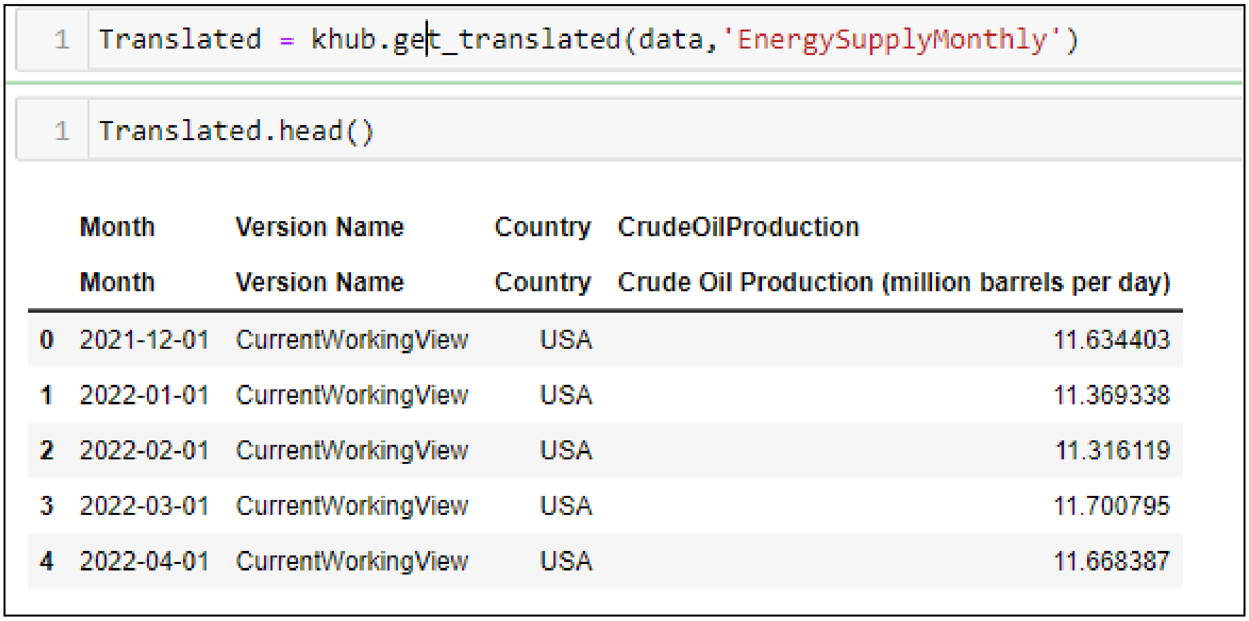
**data = khub.get_measuregrp ('RetailSalesMonthly', filters={'Year’:['2020','2021'] ,'Continent': ["Europe", "North America"]})**

In the method above, for "RetailSalesMonthly" measure group filters are applied at Continent and Year level. Although the measure group is grained at month and country level, the library can manage this and return a dataframe filtered on year and continent.

## Additional Functions

Other than the functions and methods mentioned in the previous sections, knowledge hub library also comprise a few more functions that will be useful for the user.

**get_translated**: Knowledge Hub offers a wide range of data streams categorized across various measure groups that follows a specific nomenclature which could be difficult for a user to comprehend. Hence, this method allows the user to see the translation of the requested data and will add a header of the translated entities.



```
Translated = khub.get_translated(data,'EnergySupplyMonthly')
```

```
Translated.head()
```

|   | Month | Version Name | Country | CrudeOilProduction |
|---|---|---|---|---|
|   | Month | Version Name | Country | Crude Oil Production (million barrels per day) |
| 0 | 2021-12-01 | CurrentWorkingView | USA | 11.634403 |
| 1 | 2022-01-01 | CurrentWorkingView | USA | 11.369338 |
| 2 | 2022-02-01 | CurrentWorkingView | USA | 11.316119 |
| 3 | 2022-03-01 | CurrentWorkingView | USA | 11.700795 |
| 4 | 2022-04-01 | CurrentWorkingView | USA | 11.668387 |

**get_downloaded**: This method allows the user to download the data and metadata in an excel workbook.

User should pass the dataframe and MG:
**khub.get_downloaded(dataframe,measuregroupname)**

**get_subscription**: This method will return the dataframe comprising the list of measures user has subscribed for.

# Data Transformation

Knowledge Hub library also provides the capability to transform the data based on the time dimension attribute it is grained at. The user can also disaggregate and aggregate the data by using below functions:

## Disaggregate Functions

- Month to Week: Allows the user to disaggregate the monthly grained KHub data to week
    - Function: **month_to_week**

Sample Query: month_to_week (df, ‘timekey, 'targetkey','measurename', ’aggregationmethod’)

- Week to Day: Allows the user to disaggregate the weekly grained KHub data to day
  - Function: **convert_to_day**
    Sample Query: convert_to_day(df, ‘timekey, 'targetkey', 'measurename', ’aggregationmethod’)

Parameters used:

- **df**: dataframe obtained from the Knowledge Hub (should include all grains, Time key should be included and contain only one measure)
- **timekey**: to be keep as it is (make sure it is present in the dataframe)
- **targetkey**: desired column name to be provided (converted time dimension column)
- **measurename**: the same measure name to be taken up from **df**
- **aggregationmethod**: ‘SUM’ or ‘AVERAGE’

It should be noted that the dataframe passed as a parameter should include all the grains of the measure and only contain one measure

# Aggregate Functions

- Day to Week: Allows the user to aggregate the daily level KHub data to week.
  - Function: **convert_to_week**
    Sample Query: convert_to week (df, ‘timekey, 'targetkey',’ weekstart’, 'measurename', ’aggregationmethod’)
- Day to Month: Allows the user to aggregate the daily level KHub data to month
  - Function: **convert_to_month**
    Sample Query: convert_to_month(df, ‘timekey, 'targetkey','measurename',’aggregationmethod’)
- Week to Month: Allows the user to aggregate the weekly level KHub data to month
  - Function: **week_to_month**
    Sample Query: week_to_month(df, ‘timekey, 'targetkey','measurename',’aggregationmethod’)

Parameters used:

- **df**: dataframe obtained from the Knowledge Hub (should include all grains, Time key should be included and contain only one measure)
- **timekey**: to be kept as it is (make sure it is present in the dataframe)
- **targetkey**: desired column name to be provided (converted time dimension column)
- **measurename**: the same measure name to be taken up from **df**
- **aggregationmethod**: ‘SUM’ or ‘AVERAGE’
- **weekstart**: start day of the week (please refer to the last )

# Mapping Function

- Week X to Week Y: Allows the user to change the starting day of week from (e.g. changing the starting day from Sunday to Monday)
    - Function: map_week

      Sample query: map_week (df, timekey, targetweekday)

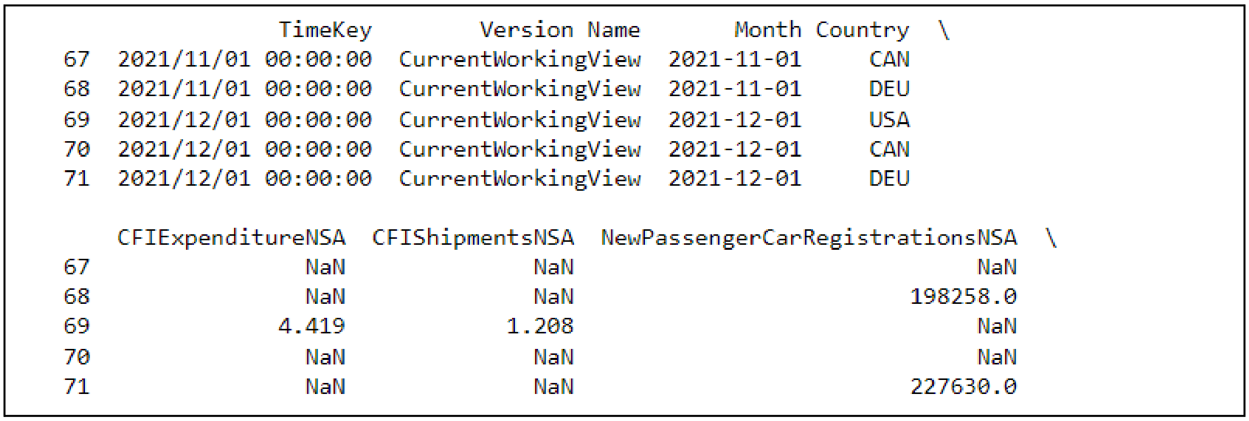
Parameters used:

- **df**: dataframe obtained from the Knowledge Hub (should contain only one measure)
- **timekey**: to be keep as it is (make sure it is present in the dataframe)
- **targetweekday**: desired weekday the user wants to convert.
  “targetweekday” should be chosen from the list; ‘SUN’, ‘MON’, ‘TUE’, ‘WED’, ‘THU’, ‘FRI’, ‘SAT’
  Dataframe should only contain one measure and all the grains including timekey.

Example:
In the below query, we are trying to get the data from ‘TransportMonthly’ MG and stored it in DF1 object. Notice, ‘timekey’ argument has been set to True and it should be included while creating a dataframe to perform data transformation.

**DF1 = KHub.get_measuregrp (‘TransportMonthly’, timekey = True)**

The output will look something like this.



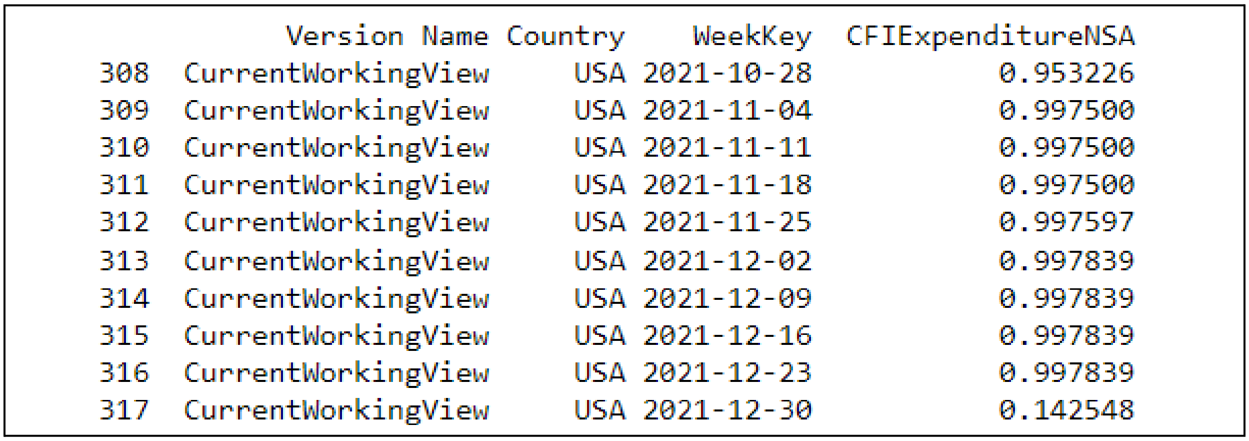
```
               TimeKey        Version Name       Month Country  \
67  2021/11/01 00:00:00  CurrentWorkingView  2021-11-01     CAN
68  2021/11/01 00:00:00  CurrentWorkingView  2021-11-01     DEU
69  2021/12/01 00:00:00  CurrentWorkingView  2021-12-01     USA
70  2021/12/01 00:00:00  CurrentWorkingView  2021-12-01     CAN
71  2021/12/01 00:00:00  CurrentWorkingView  2021-12-01     DEU

    CFIExpenditureNSA  CFIShipmentsNSA  NewPassengerCarRegistrationsNSA  \
67                NaN              NaN                              NaN
68                NaN              NaN                         198258.0
69              4.419            1.208                              NaN
70                NaN              NaN                              NaN
71                NaN              NaN                         227630.0
```

Herein, the MG has multiple measures, however, to perform time transformation, only a single measure must be mentioned. Please note the below query for time conversion wherein a single measure df is being stored in an object DF2.

**DF2 = DF1[['Version Name', 'Country','TimeKey', ‘CFIExpenditureNSA']]**

F1Conversion=KHub.month_to_week (DF2, 'TimeKey', 'WeekKey', 'THU', 'CFIExpenditureNSA', 'SUM')

Kindly notice the DF2 object has only one measure, and the same measure is used as a target measure as an argument. The bold ‘TimeKey’ should be kept same in params as well as the object dataframe. Please refer to the below screenshot of the output, wherein monthly data is disaggregated at week level.



```
         Version Name Country    WeekKey  CFIExpenditureNSA
308  CurrentWorkingView     USA 2021-10-28           0.953226
309  CurrentWorkingView     USA 2021-11-04           0.997500
310  CurrentWorkingView     USA 2021-11-11           0.997500
311  CurrentWorkingView     USA 2021-11-18           0.997500
312  CurrentWorkingView     USA 2021-11-25           0.997597
313  CurrentWorkingView     USA 2021-12-02           0.997839
314  CurrentWorkingView     USA 2021-12-09           0.997839
315  CurrentWorkingView     USA 2021-12-16           0.997839
316  CurrentWorkingView     USA 2021-12-23           0.997839
317  CurrentWorkingView     USA 2021-12-30           0.142548
```

It should be noted that to execute this transformation, user must include timekey as a column in a dataframe. This can be done by passing the parameter **timekey = True.**

# Cache

Knowledge hub library allows the user to keep a cache file for every request and returns result from the cache file if the same request is made within 24 hours.

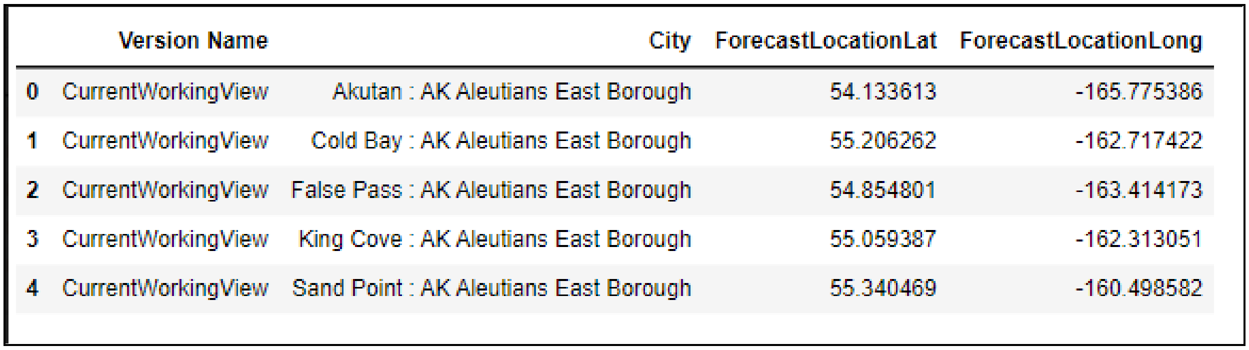
If the users want, they can switch off the cache by passing parameter caching = False in the methods
Example: data = khub.get_catalog(caching=False)

# Test Case

Plot a graph to show Weather Forecast data for Detroit, Michigan
Here we will be requesting the Weather forecast data for Detroit Michigan using filters on '**WeatherForecastDaily**' measure group.

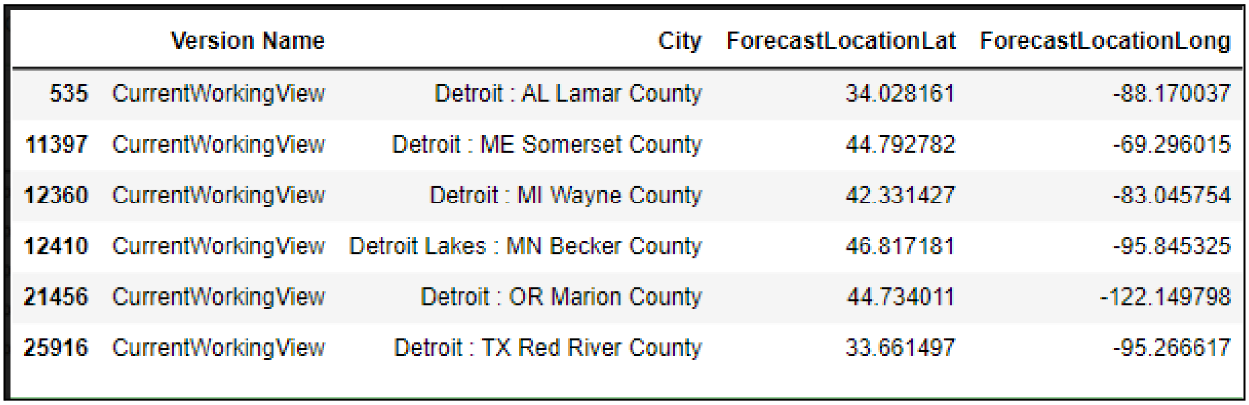
The first step is to identify the stored city name for Detroit situated in Michigan State is stored in that measure group, we need to find out city names in the measure group 'ForecastLocationsList'.

Add a global filter on K-Hub object for country "USA" and Year 2021 to 2022 and calling get_measuregrp to fetch data from "ForecastLocationsList" with global filters.
**FcstCity = khub.get_measuregrp('ForecastLocationsList')**
FcstCity is a DataFrame containing Forecast city information of cities within "USA”.



| | Version Name | City | ForecastLocationLat | ForecastLocationLong |
|---|---|---|---|---|
| **0** | CurrentWorkingView | Akutan : AK Aleutians East Borough | 54.133613 | -165.775386 |
| **1** | CurrentWorkingView | Cold Bay : AK Aleutians East Borough | 55.206262 | -162.717422 |
| **2** | CurrentWorkingView | False Pass : AK Aleutians East Borough | 54.854801 | -163.414173 |
| **3** | CurrentWorkingView | King Cove : AK Aleutians East Borough | 55.059387 | -162.313051 |
| **4** | CurrentWorkingView | Sand Point : AK Aleutians East Borough | 55.340469 | -160.498582 |

We can further filter the Dataframe to look for cities whose name starts with Detroit.
**FcstCity.loc[FcstCity['City'].str.startswith('Detroit', na=False)]**

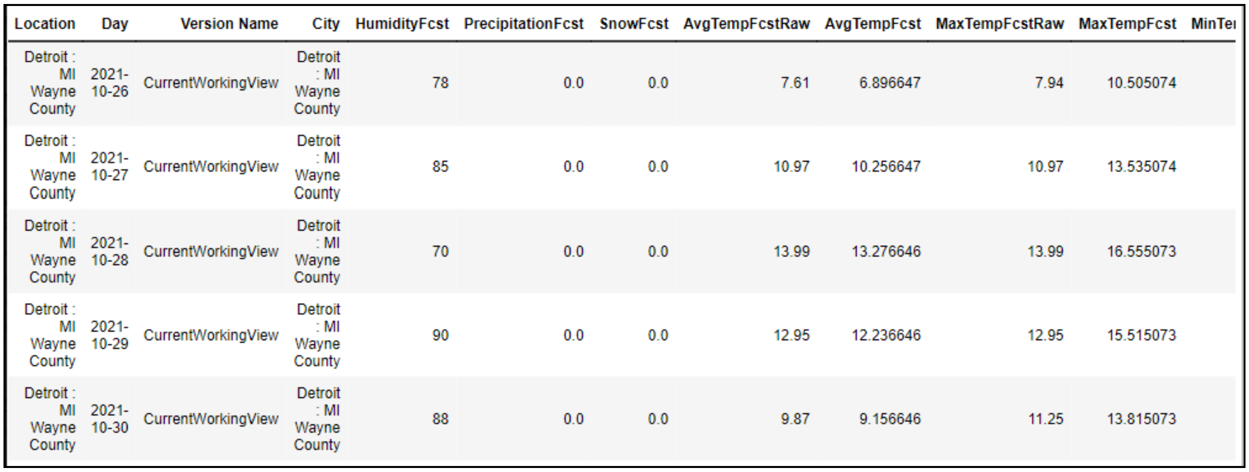


| | Version Name | City | ForecastLocationLat | ForecastLocationLong |
|---|---|---|---|---|
| **535** | CurrentWorkingView | Detroit : AL Lamar County | 34.028161 | -88.170037 |
| **11397** | CurrentWorkingView | Detroit : ME Somerset County | 44.792782 | -69.296015 |
| **12360** | CurrentWorkingView | Detroit : MI Wayne County | 42.331427 | -83.045754 |
| **12410** | CurrentWorkingView | Detroit Lakes : MN Becker County | 46.817181 | -95.845325 |
| **21456** | CurrentWorkingView | Detroit : OR Marion County | 44.734011 | -122.149798 |
| **25916** | CurrentWorkingView | Detroit : TX Red River County | 33.661497 | -95.266617 |

This shows all the cities in USA whose name starts with Detroit from the above DataFrame we can figure out that Detroit, Michigan = Detroit: MI Wayne County

Now, we can bring data from 'WeatherForecastDaily' and apply filter city = Detroit : MI Wayne County

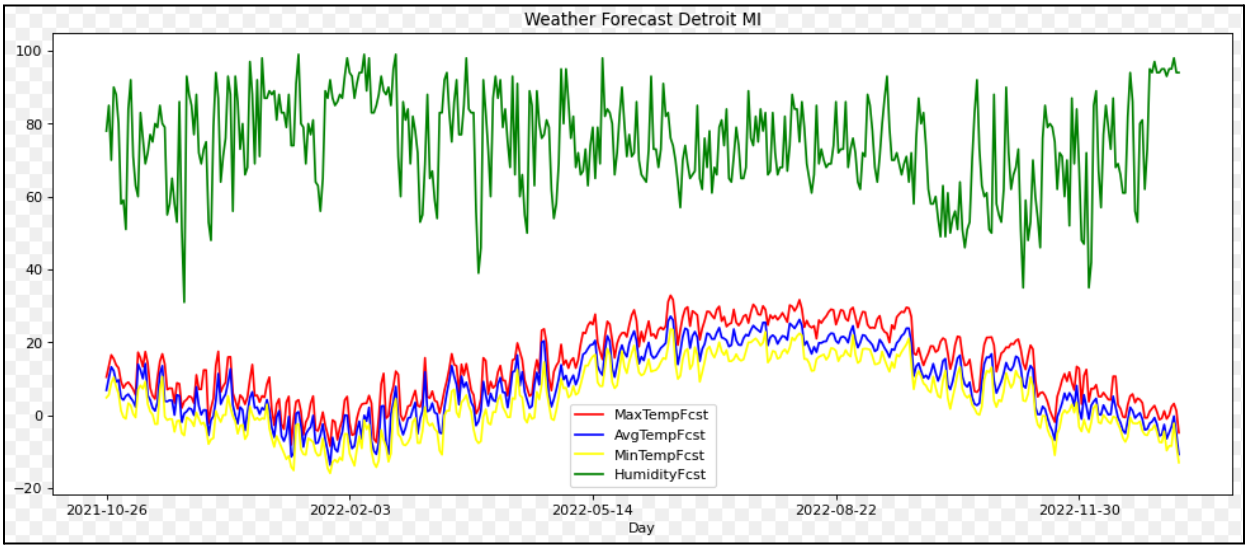
```
data = khub.get_measuregrp('WeatherForecastDaily’, filters= {'City':["Detroit : MI Wayne
County"]})
```



| Location | Day | Version Name | City | HumidityFcst | PrecipitationFcst | SnowFcst | AvgTempFcstRaw | AvgTempFcst | MaxTempFcstRaw | MaxTempFcst | MinTe |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Detroit : MI Wayne County | 2021-10-26 | CurrentWorkingView | Detroit : MI Wayne County | 78 | 0.0 | 0.0 | 7.61 | 6.896647 | 7.94 | 10.505074 | |
| Detroit : MI Wayne County | 2021-10-27 | CurrentWorkingView | Detroit : MI Wayne County | 85 | 0.0 | 0.0 | 10.97 | 10.256647 | 10.97 | 13.535074 | |
| Detroit : MI Wayne County | 2021-10-28 | CurrentWorkingView | Detroit : MI Wayne County | 70 | 0.0 | 0.0 | 13.99 | 13.276646 | 13.99 | 16.555073 | |
| Detroit : MI Wayne County | 2021-10-29 | CurrentWorkingView | Detroit : MI Wayne County | 90 | 0.0 | 0.0 | 12.95 | 12.236646 | 12.95 | 15.515073 | |
| Detroit : MI Wayne County | 2021-10-30 | CurrentWorkingView | Detroit : MI Wayne County | 88 | 0.0 | 0.0 | 9.87 | 9.156646 | 11.25 | 13.815073 | |

We can plot this data on a graph using the following code snippet:

```
import matplotlib.pyplot as plt
from matplotlib.pyplot import figure
figure(figsize=(15, 6), dpi=80)
ax = plt.gca()

data.plot(kind = 'line',x = 'Day', y = 'MaxTempFcst',color = 'red',ax = ax)
data.plot(kind = 'line',x = 'Day',y = 'AvgTempFcst',color = 'blue',ax = ax)
data.plot(kind = 'line',x = 'Day', y = 'MinTempFcst',color = 'yellow',ax = ax)
data.plot(kind = 'line',x = 'Day',y = 'HumidityFcst',color = 'green',ax = ax)
plt.title('Weather Forecast Detroit MI')
plt.show()
```


Weather Forecast Detroit MI
100
80
60
40
20
0
−20
2021-10-26
2022-02-03
2022-05-14
2022-08-22
2022-11-30
Day
MaxTempFcst
AvgTempFcst
MinTempFcst
HumidityFcst

# References

For More information on Knowledge Hub and its tools visit this LINK.
Knowledge Hub APIs: https://o9-knowledge.github.io/Hub/

# FAQs

Q. How can I get the API key for Knowledge Hub?
A. The user should reach out to Knowledge Hub team for the API Key (mk-pm@o9solutions.com)

Q. Can I use this Library in local machine?
A. Yes, you can use the Knowledge Hub library in your local machine, please refer to the Installation section.

Q. How can I use the Knowledge Hub data?
A. Knowledge Hub data can be used to improve demand and supply forecasting models. These data streams are the external market drivers.

Q. What is Knowledge Hub Subscription?
A. Knowledge Hub is a pool of data streams wherein the entire offerings can be browsed using get_catalog() function. Since all these data streams are not suitable for every user, K-Hub maintains a table wherein the relevant series as per country, category, etc. are managed and only those series will be made available as per user.

Q. How can I subscribe to other Knowledge Hub data streams?
A. Work-in-progress (Reach out to Knowledge Hub team (mk-pm@o9solutions.com)

Q. What is * when I see my subscription?
A. If you see * on your subscription list, it implies that you have access to all the data across that field. For example, if you see a star on Country column for a measure, it means you can access data for all countries for that respective measure.

Please look at the Knowledge Hub microsite for other FAQs

# Contact Us

You can use the following resources to contact us.

## For Support

Log in to our support site at https://support.o9solutions.com/hc/en-us/ or email us at support@o9solutions.com

## To Send Your Comments

If you have any questions or comments about this documentation, email us at pmmdocumentation@o9solutions.com or visit us at our website: https://o9solutions.com/.

## To Join Our Community

Click on the following links:

- LinkedIn: https://www.linkedin.com/company/o9solutions
- Twitter: https://twitter.com/o9solutions
- Instagram: https://www.instagram.com/o9solutions

1501 Lyndon B. Johnson
Freeway, Suite 140
Dallas, Texas 75234
(214) 838-3125
https://o9solutions.com