Real-Time Analytics on unstructured

data sets

Background

At Big Wind LLC, we provide timely analytics related to weather data to our

customers.

One of our leading solutions allows our customers to query high-resolution weather

forecast data that is provided by the UK Met Office. This solution provides these core

capabilities to our customers:

•

High availability

•

Quick response

•

Timely availability of new data

Because of the significant financial losses they would incur, our customers cannot

accept any downtime. Our solution provides them with the data they need to plan

and the flexibility to make sure downtime does not occur. For example, airport and

airline operators use our solution to plan their flights or predict delays caused by

weather.

Goal

You are tasked to provide reporting and dashboard access to the forecast data with

Amazon QuickSight so that Data Analysts can query the data and show the data

insights in tabular format.

In this exercise, you will design, implement, and showcase the secured and scalable

data pipelines that ingest, transform, and support visualization of the data in

Amazon QuickSight. All new data should be available to the Data Analysts as fast as

possible.

You can choose any variables in the dataset to support one or more business use

cases of your choice. For example, your dashboard could focus on airport operators

and report the forecast on specific variables used by airline operators (dew point,

air temperature, wind speed) during takeoff.

Data

Your solution should use the MOGREPS-UK dataset and assume that the files are

available in netCDF format every few minutes with 20–30 MBs of data.

A sample dataset is available at the S3 location referenced in link [1] and more

details on the dataset can be found at the location referenced in link [2]. These links

are available in the

Links section.

Each file contains weather forecasts that predict variables such as air temperature,

humidity, and rainfall. Each variable is predicted for these dimensions: latitude,

longitude, altitude (when applicable), pressure (when applicable), and time.

Latitude and longitude values use a

RotatedGeogCS coordinates system. Time is

stored as number of time periods (for example, hours) since epoch.

You may use libraries such as

Iris to explore the file contents, but, make sure to note

the library and dependencies size when you deploy it to production. The

Sample

Code section includes a code sample to help you explore the file contents.

Distributed frameworks also have libraries that support netCDF manipulation.

Compute

Build the solution using AWS services. You can use any services or frameworks that

show our core capabilities, security, and cost effectiveness. The $10 AWS credit code

provided in the email you received should cover the cost of this exercise.

Note

•Stop all EC2 instances, terminate all clusters and disable any

computation triggers after you complete the exercise.

•Delete all data and resources after your recruiting process

concludes.

•We recommend that you tag every resource used in the exercise with

this key/value:

environment/awstakehome

.

•If there are AWS services you want to use that are not yet available in

the region(s) you chose, make sure to indicate that in the architecture

document and use a region where the service is available.

Output

Send us a single PDF document that includes this information:

•IAM User credentials (AccessKey/SecretKey; username/password)

with read access to the console, services, and resources you used to

create the solution. We will use the credentials to review the

implementation.

•A document that details your architecture, thought processes, and

any code samples and Amazon QuickSight dashboard(s). In your

architecture, make sure to highlight how you met our core

capabilities, security, and cost effectiveness.

Sample Code

### To install iris, install conda and run “conda install -y -c conda-forge iris”

import iris

filename='prods\_op\_mogreps-uk\_20140717\_03\_11\_015.nc'

#Load the list of cubes. See [3] in the links section for more details.

listofcubes = iris.load(filename)

print(listofcubes)

#Extract one cube, air\_temperature measure in Kelvin. See [3] in the links section

for more details.

cube=listofcubes[8]

print(cube)

#Number of dimensions for the time coordinate

cube.coord('time').points

#Number of dimensions for the pressure coordinate

cube.coord('pressure').points

#Number of dimensions for the longitude coordinate

cube.coord('grid\_longitude').points.size

#Convert from kelvin to Celsius

cube.convert\_units('celsius')

#Print the cube contents

print(cube.data)

#Get a subset of the data

print(cube.data[0][0][0][0:10])

Links

[1] s3://awsbigdatatakehome/ (region: eu-west-2)

[2]

https://aws.amazon.com/public-datasets/mogreps/

[3]

https://scitools.org.uk/iris/docs/latest/userguide/iris\_cubes.html and

https://scitools.org.uk/iris/docs/latest/userguide/loading\_iris\_cubes.html

**AWS WORK Rough Draft**

--Read about MOGREPS on <https://www.metoffice.gov.uk/research/weather/ensemble-forecasting/mogreps>

--Briefly read about iris and referred examples

<https://conda-forge.org/>

<https://github.com/conda-forge>

<https://scitools.org.uk/iris/docs/latest/userguide/iris_cubes.html>

<https://scitools.org.uk/iris/docs/latest/userguide/loading_iris_cubes.html>

<https://registry.opendata.aws/uk-met-office/>

and then

<https://github.com/MetOffice/aws-earth-examples/blob/master/examples/1.%20Getting%20Started.ipynb>

&

https://github.com/MetOffice/aws-earth-examples/blob/master/examples/2.%20Subscribing%20to%20data.ipynb

read about

<https://www.unidata.ucar.edu/software/netcdf/>

Need to look into:

<https://github.com/conda-forge/iris-feedstock>

Refer:

<https://www.metoffice.gov.uk/services/data/met-office-data-for-reuse/discovery>

Above link talks about:

The following NWP data model sets are available:

* [**Atmospheric Hi-Res Model - 10km deterministic**](https://www.metoffice.gov.uk/binaries/content/assets/metofficegovuk/pdf/data/global-model-parameters-may-2019.pdf)
* [**Met Office UK Atmospheric Hi-Res Model (UKV) - 2km deterministic**](https://www.metoffice.gov.uk/binaries/content/assets/metofficegovuk/pdf/data/ukv-parameters-may-2019.pdf)
* [**Met Office Global and Regional Ensemble Prediction System  - Global Ensemble (MOGREPS-G)**](https://www.metoffice.gov.uk/binaries/content/assets/metofficegovuk/pdf/data/mogreps-g-parameters-may-2019.pdf)
* [**Met Office Global and Regional Ensemble Prediction System - UK Ensemble (MOGREPS-UK)**](https://www.metoffice.gov.uk/binaries/content/assets/metofficegovuk/pdf/data/mogreps-uk-parameters-may-2019.pdf)

Tried out sample code for the key mentioned:

'prods\_op\_mogreps-uk\_20140717\_03\_11\_015.nc'

Setup conda

(base) C:\Users\Win10>conda install -y -c conda-forge iris

Collecting package metadata (current\_repodata.json): done

Solving environment: failed with initial frozen solve. Retrying with flexible solve.

Solving environment: failed with repodata from current\_repodata.json, will retry with next repodata source.

Collecting package metadata (repodata.json): done

Solving environment: done

## Package Plan ##

environment location: C:\Users\Win10\Anaconda3

added / updated specs:

- iris

The following packages will be downloaded:

package | build

---------------------------|-----------------

antlr-python-runtime-4.7.2 | py37\_1001 165 KB conda-forge

cartopy-0.17.0 |py37hfd97135\_1007 1.7 MB conda-forge

certifi-2019.9.11 | py37\_0 147 KB conda-forge

cf-units-2.1.4 | py37hbc2f12b\_0 162 KB conda-forge

cftime-1.2.1 | py37h44b1f71\_0 210 KB conda-forge

conda-4.8.4 | py37hc8dfbb8\_2 3.1 MB conda-forge

expat-2.2.9 | h33f27b4\_2 261 KB conda-forge

geos-3.7.2 | he025d50\_2 915 KB conda-forge

hdf4-4.2.13 | h712560f\_2 1.3 MB

iris-2.4.0 | py37\_0 1.7 MB conda-forge

libnetcdf-4.6.1 | h411e497\_2 494 KB

netcdf4-1.4.2 | py37h812ae01\_0 338 KB

owslib-0.20.0 | py\_0 127 KB conda-forge

proj-6.2.0 | ha7a8c7b\_1 8.5 MB conda-forge

pyepsg-0.4.0 | py\_0 20 KB conda-forge

pykdtree-1.3.1 |py37hbc2f12b\_1003 56 KB conda-forge

pyke-1.1.1 |py37hc8dfbb8\_1002 197 KB conda-forge

pyproj-2.4.1 | py37h77f86b8\_0 269 KB conda-forge

pyshp-2.1.0 | py\_0 31 KB conda-forge

shapely-1.6.4 |py37ha35856d\_1006 383 KB conda-forge

udunits2-2.2.27.6 | h252784a\_1001 126 KB conda-forge

------------------------------------------------------------

Total: 20.1 MB

The following NEW packages will be INSTALLED:

antlr-python-runt~ conda-forge/win-64::antlr-python-runtime-4.7.2-py37\_1001

cartopy conda-forge/win-64::cartopy-0.17.0-py37hfd97135\_1007

cf-units conda-forge/win-64::cf-units-2.1.4-py37hbc2f12b\_0

cftime conda-forge/win-64::cftime-1.2.1-py37h44b1f71\_0

expat conda-forge/win-64::expat-2.2.9-h33f27b4\_2

geos conda-forge/win-64::geos-3.7.2-he025d50\_2

hdf4 pkgs/main/win-64::hdf4-4.2.13-h712560f\_2

iris conda-forge/win-64::iris-2.4.0-py37\_0

libnetcdf pkgs/main/win-64::libnetcdf-4.6.1-h411e497\_2

netcdf4 pkgs/main/win-64::netcdf4-1.4.2-py37h812ae01\_0

owslib conda-forge/noarch::owslib-0.20.0-py\_0

proj conda-forge/win-64::proj-6.2.0-ha7a8c7b\_1

pyepsg conda-forge/noarch::pyepsg-0.4.0-py\_0

pykdtree conda-forge/win-64::pykdtree-1.3.1-py37hbc2f12b\_1003

pyke conda-forge/win-64::pyke-1.1.1-py37hc8dfbb8\_1002

pyproj conda-forge/win-64::pyproj-2.4.1-py37h77f86b8\_0

pyshp conda-forge/noarch::pyshp-2.1.0-py\_0

shapely conda-forge/win-64::shapely-1.6.4-py37ha35856d\_1006

udunits2 conda-forge/win-64::udunits2-2.2.27.6-h252784a\_1001

The following packages will be UPDATED:

conda 4.8.3-py37hc8dfbb8\_1 --> 4.8.4-py37hc8dfbb8\_2

The following packages will be SUPERSEDED by a higher-priority channel:

certifi pkgs/main --> conda-forge

Downloading and Extracting Packages

pykdtree-1.3.1 | 56 KB | ############################################################################ | 100%

pyshp-2.1.0 | 31 KB | ############################################################################ | 100%

netcdf4-1.4.2 | 338 KB | ############################################################################ | 100%

antlr-python-runtime | 165 KB | ############################################################################ | 100%

cftime-1.2.1 | 210 KB | ############################################################################ | 100%

geos-3.7.2 | 915 KB | ############################################################################ | 100%

conda-4.8.4 | 3.1 MB | ############################################################################ | 100%

cf-units-2.1.4 | 162 KB | ############################################################################ | 100%

pyepsg-0.4.0 | 20 KB | ############################################################################ | 100%

libnetcdf-4.6.1 | 494 KB | ############################################################################ | 100%

proj-6.2.0 | 8.5 MB | ############################################################################ | 100%

hdf4-4.2.13 | 1.3 MB | ############################################################################ | 100%

udunits2-2.2.27.6 | 126 KB | ############################################################################ | 100%

shapely-1.6.4 | 383 KB | ############################################################################ | 100%

certifi-2019.9.11 | 147 KB | ############################################################################ | 100%

expat-2.2.9 | 261 KB | ############################################################################ | 100%

owslib-0.20.0 | 127 KB | ############################################################################ | 100%

pyke-1.1.1 | 197 KB | ############################################################################ | 100%

cartopy-0.17.0 | 1.7 MB | ############################################################################ | 100%

pyproj-2.4.1 | 269 KB | ############################################################################ | 100%

iris-2.4.0 | 1.7 MB | ############################################################################ | 100%

Preparing transaction: done

Verifying transaction: done

Executing transaction: done

(base) C:\Users\Win10>conda search iris --channel conda-forge

Loading channels: done

# Name Version Build Channel

iris 1.9.2 np110py27\_0 conda-forge

iris 1.9.2 np110py34\_0 conda-forge

iris 1.9.2 np110py35\_0 conda-forge

iris 1.9.2 np111py27\_0 conda-forge

iris 1.9.2 np111py34\_0 conda-forge

iris 1.9.2 np111py35\_0 conda-forge

iris 1.9.2 py27\_1 conda-forge

iris 1.9.2 py34\_1 conda-forge

iris 1.9.2 py35\_1 conda-forge

iris 1.10.0 py27\_0 conda-forge

iris 1.10.0 py34\_0 conda-forge

iris 1.10.0 py35\_0 conda-forge

iris 1.11.0 py27\_0 conda-forge

iris 1.11.0 py34\_0 conda-forge

iris 1.11.0 py35\_0 conda-forge

iris 1.11.0 py36\_0 conda-forge

iris 1.12.0 py27\_0 conda-forge

iris 1.12.0 py27\_1 conda-forge

iris 1.12.0 py27\_2 conda-forge

iris 1.12.0 py27\_3 conda-forge

iris 1.12.0 py34\_0 conda-forge

iris 1.12.0 py35\_0 conda-forge

iris 1.12.0 py35\_1 conda-forge

iris 1.12.0 py35\_2 conda-forge

iris 1.12.0 py35\_3 conda-forge

iris 1.12.0 py36\_0 conda-forge

iris 1.12.0 py36\_1 conda-forge

iris 1.12.0 py36\_2 conda-forge

iris 1.12.0 py36\_3 conda-forge

iris 1.13.0 py27h39e3cac\_2 conda-forge

iris 1.13.0 py35h39e3cac\_2 conda-forge

iris 1.13.0 py36h39e3cac\_2 conda-forge

iris 2.0.0 py27\_0 conda-forge

iris 2.0.0 py27\_1 conda-forge

iris 2.0.0 py27\_2 conda-forge

iris 2.0.0 py35\_0 conda-forge

iris 2.0.0 py35\_1 conda-forge

iris 2.0.0 py35\_2 conda-forge

iris 2.0.0 py36\_0 conda-forge

iris 2.0.0 py36\_1 conda-forge

iris 2.0.0 py36\_2 conda-forge

iris 2.1.0 py27\_1 conda-forge

iris 2.1.0 py27\_2 conda-forge

iris 2.1.0 py27\_3 conda-forge

iris 2.1.0 py27\_4 conda-forge

iris 2.1.0 py35\_1 conda-forge

iris 2.1.0 py35\_2 conda-forge

iris 2.1.0 py35\_3 conda-forge

iris 2.1.0 py36\_1 conda-forge

iris 2.1.0 py36\_2 conda-forge

iris 2.1.0 py36\_3 conda-forge

iris 2.1.0 py36\_4 conda-forge

iris 2.1.0 py37\_4 conda-forge

iris 2.2.0 py27\_0 conda-forge

iris 2.2.0 py27\_1 conda-forge

iris 2.2.0 py27\_1002 conda-forge

iris 2.2.0 py27\_1003 conda-forge

iris 2.2.0 py27\_2 conda-forge

iris 2.2.0 py36\_0 conda-forge

iris 2.2.0 py36\_1 conda-forge

iris 2.2.0 py36\_1002 conda-forge

iris 2.2.0 py36\_1003 conda-forge

iris 2.2.0 py36\_2 conda-forge

iris 2.2.0 py37\_0 conda-forge

iris 2.2.0 py37\_1 conda-forge

iris 2.2.0 py37\_1002 conda-forge

iris 2.2.0 py37\_1003 conda-forge

iris 2.2.0 py37\_2 conda-forge

iris 2.2.1 py27\_0 conda-forge

iris 2.2.1 py36\_0 conda-forge

iris 2.2.1 py37\_0 conda-forge

iris 2.2.1 py38\_0 conda-forge

iris 2.3.0 py27\_0 conda-forge

iris 2.3.0 py36\_0 conda-forge

iris 2.3.0 py37\_0 conda-forge

iris 2.3.0 py38\_0 conda-forge

iris 2.4.0 py27\_0 conda-forge

iris 2.4.0 py36\_0 conda-forge

iris 2.4.0 py37\_0 conda-forge

iris 2.4.0 py38\_0 conda-forge

**Sample Code:**

sample\_sns\_message = {

    'bucket': 'awsbigdatatakehome',

    'key': 'prods\_op\_mogreps-uk\_20140717\_03\_11\_015.nc'

}

import urllib.request

import iris

def download\_data\_object(sns\_message):

    url = "https://s3.eu-west-2.amazonaws.com/" + sns\_message['bucket'] + "/" + sns\_message['key']

    urllib.request.urlretrieve(url, sns\_message['key']) # save in this directory with same name

download\_data\_object(sample\_sns\_message)

filename='prods\_op\_mogreps-uk\_20140717\_03\_11\_015.nc'

#Load the list of cubes. See [3] in the links section for more details.

listofcubes = iris.load(filename)

**print(listofcubes)**

0: wet\_bulb\_potential\_temperature / (K) (time: 3; pressure: 3; grid\_latitude: 548; grid\_longitude: 421)

1: wet\_bulb\_freezing\_level\_altitude / (m) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

2: stratiform\_snowfall\_rate / (kg m-2 s-1) (time: 36; grid\_latitude: 548; grid\_longitude: 421)

3: cloud\_area\_fraction\_assuming\_maximum\_random\_overlap / (1) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

4: cloud\_base\_altitude\_assuming\_only\_consider\_cloud\_area\_fraction\_greater\_than\_2p5\_oktas / (kft) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

5: unknown / (1) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

6: air\_pressure\_at\_sea\_level / (Pa) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

7: air\_temperature / (K) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

8: air\_temperature / (K) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

9: air\_temperature / (K) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

10: air\_temperature / (K) (time: 3; pressure: 2; grid\_latitude: 548; grid\_longitude: 421)

11: dew\_point\_temperature / (K) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

12: fog\_area\_fraction / (1) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

13: geopotential\_height / (m) (time: 3; pressure: 3; grid\_latitude: 548; grid\_longitude: 421)

14: high\_type\_cloud\_area\_fraction / (1) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

15: low\_type\_cloud\_area\_fraction / (1) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

16: medium\_type\_cloud\_area\_fraction / (1) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

17: relative\_humidity / (%) (time: 3; pressure: 2; grid\_latitude: 548; grid\_longitude: 421)

18: relative\_humidity / (%) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

19: specific\_humidity / (1) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

20: stratiform\_rainfall\_amount / (kg m-2) (-- : 3; grid\_latitude: 548; grid\_longitude: 421)

21: stratiform\_rainfall\_rate / (kg m-2 s-1) (time: 36; grid\_latitude: 548; grid\_longitude: 421)

22: stratiform\_snowfall\_amount / (kg m-2) (-- : 3; grid\_latitude: 548; grid\_longitude: 421)

23: surface\_air\_pressure / (Pa) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

24: surface\_downward\_eastward\_stress / (Pa) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

26: surface\_downwelling\_longwave\_flux / (W m-2) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

27: surface\_downwelling\_shortwave\_flux\_in\_air / (W m-2) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

28: visibility\_in\_air / (m) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

29: wind\_speed\_of\_gust / (m s-1) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

30: x\_wind / (m s-1) (time: 3; grid\_latitude: 547; grid\_longitude: 421)

31: x\_wind / (m s-1) (time: 3; pressure: 2; grid\_latitude: 547; grid\_longitude: 421)

32: x\_wind / (m s-1) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

33: y\_wind / (m s-1) (time: 3; pressure: 2; grid\_latitude: 547; grid\_longitude: 421)

35: y\_wind / (m s-1) (time: 3; grid\_latitude: 547; grid\_longitude: 421)

#Extract one cube, air\_temperature measure in Kelvin. See [3] in the links section

for more details.

**cube=listofcubes[8]**

**>>> print(cube)**

air\_temperature / (K) (time: 3; grid\_latitude: 548; grid\_longitude: 421)

Dimension coordinates:

time x - -

grid\_latitude - x -

Auxiliary coordinates:

forecast\_reference\_time: 2014-07-17 03:00:00

height: 1.5 m

Attributes:

Conventions: CF-1.5

STASH: m01s03i236

source: Data from Met Office Unified Model

um\_version: 8.5

#Number of dimensions for the time coordinate

**>>> cube.coord('time').points**

array([390448., 390449., 390450.])

#Number of dimensions for the pressure coordinate

**>>> cube.coord('pressure').points**

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

File "C:\Users\Win10\Anaconda3\lib\site-packages\iris\cube.py", line 1497, in coord

raise iris.exceptions.CoordinateNotFoundError(msg)

iris.exceptions.CoordinateNotFoundError: 'Expected to find exactly 1 pressure coordinate, but found none.'

#Number of dimensions for the longitude coordinate

**>>> cube.coord('grid\_longitude').points.size**

421

#Convert from kelvin to Celsius

**>>>cube.convert\_units('celsius')**

#Print the cube contents

**>>> print(cube.data)**

[[[16.35 16.35 16.35 ... 30.975 31.1 30.975]

[16.35 16.35 16.35 ... 30.975 30.85 30.475]

[16.35 16.35 16.35 ... 30.725 30.6 30.1 ]

...

[14.1 14.1 14.1 ... 15.85 15.85 15.975]

[14.1 14.1 14.1 ... 15.85 15.975 15.975]]

[16.35 16.35 16.35 ... 30.6 30.6 30.1 ]

[16.35 16.35 16.35 ... 30.475 30.225 29.725]

...

[14.225 14.225 14.225 ... 15.975 15.975 16.1 ]

[14.225 14.225 14.225 ... 15.975 16.1 16.1 ]

[14.1 14.1 14.225 ... 16.1 16.1 15.975]]

[16.225 16.225 16.225 ... 29.475 29.35 28.725]

[16.225 16.225 16.225 ... 29.475 29.1 28.475]

...

[14.225 14.225 14.225 ... 15.85 15.975 15.975]

[14.225 14.225 14.225 ... 15.975 15.975 15.975]

[14.225 14.225 14.225 ... 15.975 15.975 15.975]]]

#Get a subset of the data

**#print(cube.data[0][0][0][0:10])**

>>> print(cube.data[0])

[[16.35 16.35 16.35 ... 30.975 31.1 30.975]

[16.35 16.35 16.35 ... 30.975 30.85 30.475]

[16.35 16.35 16.35 ... 30.725 30.6 30.1 ]

...

[14.1 14.1 14.1 ... 15.975 15.975 15.975]

[14.1 14.1 14.1 ... 15.85 15.85 15.975]

[14.1 14.1 14.1 ... 15.85 15.975 15.975]]

**Met Office AWS Earth data - Subscribing to data**

**Create SQS Queue:AjQueue**

arn:aws:sqs:eu-west-2:443602378074:AjQueue

<https://sqs.eu-west-2.amazonaws.com/443602378074/AjQueue>

Subscribe to Amazon SNS topic for our use case:

**Subscribed successfully to topic arn:aws:sns:eu-west-2:021908831235:aws-earth-mo-atmospheric-mogreps-uk-prd.**

Subscription ARN

arn:aws:sns:eu-west-2:021908831235:aws-earth-mo-atmospheric-mogreps-uk-prd:01367e33-d506-4754-a8f5-ae3bd187644e

Topic ARN

arn:aws:sns:eu-west-2:021908831235:aws-earth-mo-atmospheric-mogreps-uk-prd

Create Role for lambda to access sqs

**Rolename:** [**AmazonSQSReadOnlyAccess**](https://console.aws.amazon.com/iam/home?region=eu-west-2#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonSQSReadOnlyAccess)

**arn:aws:iam::aws:policy/AmazonSQSReadOnlyAccess**

Permissions:

{

"Version": "2012-10-17",

"Statement": [

{

"Action": [

"sqs:GetQueueAttributes",

"sqs:GetQueueUrl",

"sqs:ListDeadLetterSourceQueues",

"sqs:ListQueues"

],

"Effect": "Allow",

"Resource": "\*"

}

]

}

**Added inline policy: granting read access to queue**

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "VisualEditor0",

"Effect": "Allow",

"Action": [

"sqs:GetQueueUrl",

"sqs:ListDeadLetterSourceQueues",

"sqs:ReceiveMessage",

"sqs:GetQueueAttributes",

"sqs:ListQueueTags"

],

"Resource": "\*"

}

]

}

**Check in sqs for number of messages: <optional>**

**Now create lambda function**

### **func1lambsqs**

#Looking at complete message

import json

import boto3

sqs = boto3.client('sqs', region\_name='eu-west-2')

def lambda\_handler(event, context):

messages = sqs.receive\_message(

QueueUrl='https://sqs.eu-west-2.amazonaws.com/443602378074/MyQueueNew',

MaxNumberOfMessages=1)

[message] = messages['Messages']

#notification = json.loads(message['Body'])

#s3\_object = json.loads(notification['Message'])

return {

'statusCode': 200,

'body': message

}

**Test Result:**

Response:

{

"statusCode": 200,

"body": {

"MessageId": "984bb8a4-9e3d-4e1f-a5cd-30e82d52f214",

"ReceiptHandle": "AQEBli19TvJdROwxZo/xyAMTy46BHNL+HvvH74dF5K2qICKY9FNByr02q7YlOSiqpHyq43Plaq8hg9rKrUbb851/KJVynE+9LxLZEeyuijai0bG8DTJi421zeiz9HCPXwPCxytd1Wj9KpgF0ZeBxDDeOZxKjHWo3/dIOmHAwSEvfk/7Ql0CREIWVBRZE24BCK2wyGkSMgc057Dflok5olnsBnDKJc5HHv6mfQ5QqB6rocDFFH1qF7RTMWjBIitxV+4CA3Vr0mPGh9mIoHNAxznJf7+K22Fn1EtfmN2RDK7eWFL/rWoCRjgeWNolP3i+8VSMlJPlkI3ydFHdITwiFMDg7LngdMkkZJ2Y24817OUElEYCldEQ1nKRscfQNcQs2KB1doYN7vCuCtiJn5RkSvveKEQ==",

"MD5OfBody": "f13ed5849b535f450cf45c305d29fef7",

"Body": "{\n \"Type\" : \"Notification\",\n \"MessageId\" : \"2447db7a-1880-5598-a66f-8ca1c0669c7f\",\n \"TopicArn\" : \"arn:aws:sns:eu-west-2:021908831235:aws-earth-mo-atmospheric-mogreps-uk-prd\",\n \"Message\" : \"{\\\"model\\\": \\\"mo-atmospheric-mogreps-uk-prd\\\", \\\"object\_size\\\": 369931.0, \\\"forecast\_reference\_time\\\": \\\"2020-09-15T07:00:00Z\\\", \\\"forecast\_period\\\": \\\"132300\\\", \\\"forecast\_period\_units\\\": \\\"seconds\\\", \\\"ttl\\\": 1600852258, \\\"time\\\": \\\"2020-09-16T19:45:00Z\\\", \\\"bucket\\\": \\\"aws-earth-mo-atmospheric-mogreps-uk-prd\\\", \\\"key\\\": \\\"282f5145487119b2a59377f6bf9d7f66d7cee5b8.nc\\\", \\\"created\_time\\\": \\\"2020-09-15T09:03:20Z\\\", \\\"name\\\": \\\"rainfall\_rate\\\", \\\"realization\\\": \\\"6 7 8\\\"}\",\n \"Timestamp\" : \"2020-09-16T09:11:02.490Z\",\n \"SignatureVersion\" : \"1\",\n \"Signature\" : \"laXA80tVVcqotG5VDCKla8DZ45+8+vP2w/OdETO92CNBK2+2rfg58vnlF6LSEA3asL1LgjDCYL81L/GeabRJGOvd8agYssdv79LfhnyvcP5sv8F5A2ysrDjIWv2wJLGMDSpb29BTAXO82uleQ+TnmeCYlfu7l9WmxY1tao39T/b35avl5r5JxdZS4bfCMknZ7v7ieL/zJpl2+dIu0O8lIYW3LsdYfMMlQkEMMKPb6c9Z2DykUk2SKZyFFIpRwpz+7naGbvXvrU4wIuGPLRe/yeLEQfyMJ7KhI2fnOdot+yzdhWbmPvymCdd9+1gDPULHdSjultia4gITs4PAdEU/LA==\",\n \"SigningCertURL\" : \"https://sns.eu-west-2.amazonaws.com/SimpleNotificationService-a86cb10b4e1f29c941702d737128f7b6.pem\",\n \"UnsubscribeURL\" : \"https://sns.eu-west-2.amazonaws.com/?Action=Unsubscribe&SubscriptionArn=arn:aws:sns:eu-west-2:021908831235:aws-earth-mo-atmospheric-mogreps-uk-prd:55e7fc85-e0b5-47bb-ad32-794ee1c02edf\",\n \"MessageAttributes\" : {\n \"forecast\_reference\_time\" : {\"Type\":\"String\",\"Value\":\"2020-09-15T07:00:00Z\"},\n \"name\" : {\"Type\":\"String\",\"Value\":\"rainfall\_rate\"},\n \"model\" : {\"Type\":\"String\",\"Value\":\"mo-atmospheric-mogreps-uk-prd\"}\n }\n}"

}

}

Request ID:

"f545e89e-58ec-42c7-83f1-73f3e1ac8194"

Function logs:

START RequestId: f545e89e-58ec-42c7-83f1-73f3e1ac8194 Version: $LATEST

END RequestId: f545e89e-58ec-42c7-83f1-73f3e1ac8194

REPORT RequestId: f545e89e-58ec-42c7-83f1-73f3e1ac8194 Duration: 152.80 ms Billed Duration: 200 ms Memory Size: 128 MB Max Memory Used: 77 MB

--------------

#Looking at Body in message

import json

import boto3

sqs = boto3.client('sqs', region\_name='eu-west-2')

def lambda\_handler(event, context):

messages = sqs.receive\_message(

QueueUrl='https://sqs.eu-west-2.amazonaws.com/443602378074/MyQueueNew',

MaxNumberOfMessages=1)

[message] = messages['Messages']

notification = json.loads(message['Body'])

#s3\_object = json.loads(notification['Message'])

return {

'statusCode': 200,

'body': notification

}

**Test Result:**

Response:

{

"statusCode": 200,

"body": {

"Type": "Notification",

"MessageId": "e6645315-2f3b-512a-b7af-04cef7416eb3",

"TopicArn": "arn:aws:sns:eu-west-2:021908831235:aws-earth-mo-atmospheric-mogreps-uk-prd",

"Message": "{\"model\": \"mo-atmospheric-mogreps-uk-prd\", \"long\_name\": \"lwe\_graupel\_and\_hail\_fall\_rate\", \"ttl\": 1600852149, \"time\": \"2020-09-16T17:00:00Z\", \"bucket\": \"aws-earth-mo-atmospheric-mogreps-uk-prd\", \"created\_time\": \"2020-09-15T09:03:15Z\", \"name\": \"lwe\_graupel\_and\_hail\_fall\_rate\", \"forecast\_period\_bounds\": \"118800 122400\", \"object\_size\": 153225.0, \"forecast\_period\": \"122400\", \"forecast\_reference\_time\": \"2020-09-15T07:00:00Z\", \"forecast\_period\_units\": \"seconds\", \"cell\_methods\": \"time: maximum\", \"key\": \"a004c03c1a860697c2a683a7fb49c8260a57ecaa.nc\", \"realization\": \"6 7 8\"}",

"Timestamp": "2020-09-16T09:09:14.056Z",

"SignatureVersion": "1",

"Signature": "nrjVndPvOIFOe+E65LGlaS28XTAMClohV2MSw11S+XLeUF4Bksu/IO8qr6HxEWuJbdOYf+jUigxL+5JmrdZqbAvDxni/Ih23xPD4YG/cNxh1qro9DMWfdue7BhzD5HhFM5xN1s+Be5xLbvH7RQijRgp8GuJ1HxP8mBlZGselAg+vo0yKTchXpQq7kbXCk6Baqnyn6kHZlto1zj9lUv6vpts+gL0aSxWU2LHmAXuWdnz/9N/YF/ebMoIlaxCgakNQqIN88Fs89zmXYdWQ1DkGf3hBlaGvpH3O7GgG9fmK21mPI7cST+3899Sg1IJMuQT9etGkWWsqzKqcz3oFN66tDg==",

"SigningCertURL": "https://sns.eu-west-2.amazonaws.com/SimpleNotificationService-a86cb10b4e1f29c941702d737128f7b6.pem",

"UnsubscribeURL": "https://sns.eu-west-2.amazonaws.com/?Action=Unsubscribe&SubscriptionArn=arn:aws:sns:eu-west-2:021908831235:aws-earth-mo-atmospheric-mogreps-uk-prd:55e7fc85-e0b5-47bb-ad32-794ee1c02edf",

"MessageAttributes": {

"forecast\_reference\_time": {

"Type": "String",

"Value": "2020-09-15T07:00:00Z"

},

"name": {

"Type": "String",

"Value": "lwe\_graupel\_and\_hail\_fall\_rate"

},

"model": {

"Type": "String",

"Value": "mo-atmospheric-mogreps-uk-prd"

}

}

}

}

Request ID:

"ca5035fb-20d4-43c8-b6bb-711190f53015"

Function logs:

START RequestId: ca5035fb-20d4-43c8-b6bb-711190f53015 Version: $LATEST

END RequestId: ca5035fb-20d4-43c8-b6bb-711190f53015

REPORT RequestId: ca5035fb-20d4-43c8-b6bb-711190f53015 Duration: 193.74 ms Billed Duration: 200 ms Memory Size: 128 MB Max Memory Used: 76 MB Init Duration: 360.64 ms

#Extracting the message which contains information such as filename,bucketname,type of forecast etc.

import json

import boto3

sqs = boto3.client('sqs', region\_name='eu-west-2')

def lambda\_handler(event, context):

messages = sqs.receive\_message(

QueueUrl='https://sqs.eu-west-2.amazonaws.com/443602378074/MyQueueNew',

MaxNumberOfMessages=1)

[message] = messages['Messages']

notification = json.loads(message['Body'])

s3\_object = json.loads(notification['Message'])

return {

'statusCode': 200,

'body': s3\_object

}

**Test result:**

Response:

{

"statusCode": 200,

"body": {

"model": "mo-atmospheric-mogreps-uk-prd",

"object\_size": 6480768,

"forecast\_reference\_time": "2020-09-15T07:00:00Z",

"forecast\_period": "100800",

"forecast\_period\_units": "seconds",

"ttl": 1600852270,

"time": "2020-09-16T11:00:00Z",

"bucket": "aws-earth-mo-atmospheric-mogreps-uk-prd",

"key": "48e44c008abf875b812c17ca31f840c9bdad8828.nc",

"created\_time": "2020-09-15T09:03:17Z",

"name": "surface\_downwelling\_longwave\_flux\_in\_air",

"realization": "6 7 8"

}

}

Request ID:

"be4c24fd-ede3-40de-836d-7bfdf404f107"

Function logs:

START RequestId: be4c24fd-ede3-40de-836d-7bfdf404f107 Version: $LATEST

END RequestId: be4c24fd-ede3-40de-836d-7bfdf404f107

REPORT RequestId: be4c24fd-ede3-40de-836d-7bfdf404f107 Duration: 184.01 ms Billed Duration: 200 ms Memory Size: 128 MB Max Memory Used: 75 MB Init Duration: 346.34 ms

**#Working on rainfall\_rate data:**

**--Added inline policy to existing role to be able to write to my bucket.**

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "VisualEditor0",

"Effect": "Allow",

"Action": [

"s3:CreateAccessPoint",

"s3:PutAnalyticsConfiguration",

"s3:PutAccelerateConfiguration",

"s3:DeleteObjectVersion",

"s3:RestoreObject",

"s3:DeleteAccessPoint",

"s3:CreateBucket",

"s3:ReplicateObject",

"s3:PutEncryptionConfiguration",

"s3:DeleteBucketWebsite",

"s3:AbortMultipartUpload",

"s3:PutLifecycleConfiguration",

"s3:UpdateJobPriority",

"s3:DeleteObject",

"s3:DeleteBucket",

"s3:PutBucketVersioning",

"s3:PutMetricsConfiguration",

"s3:PutReplicationConfiguration",

"s3:PutObjectLegalHold",

"s3:UpdateJobStatus",

"s3:PutBucketCORS",

"s3:PutInventoryConfiguration",

"s3:PutObject",

"s3:PutBucketNotification",

"s3:PutBucketWebsite",

"s3:PutBucketRequestPayment",

"s3:PutObjectRetention",

"s3:PutBucketLogging",

"s3:PutBucketObjectLockConfiguration",

"s3:ReplicateDelete"

],

"Resource": [

"arn:aws:s3:\*:443602378074:job/\*",

"arn:aws:s3:::\*/\*",

"arn:aws:s3:::mypackmog1",

"arn:aws:s3:\*:443602378074:accesspoint/\*"

]

},

{

"Sid": "VisualEditor1",

"Effect": "Allow",

"Action": "s3:CreateJob",

"Resource": "\*"

}

]

}

**Option 1:** Identifying and copying files to my own bucket for further usage:

import json

import boto3

sqs = boto3.client('sqs', region\_name='eu-west-2')

s3 = boto3.resource('s3',aws\_access\_key\_id='xxxxxxxx',

aws\_secret\_access\_key=xxxxx’)

rainfall\_rate = set()

result = {}

def lambda\_handler(event, context):

messages = sqs.receive\_message(

QueueUrl='https://sqs.eu-west-2.amazonaws.com/443602378074/MyQueueNew',

MaxNumberOfMessages=10)

while(len(rainfall\_rate) <= 5):

for i in messages['Messages']:

x = json.loads(i['Body'])

y = json.loads(x['Message'])

if y['name'] == "rainfall\_rate":

rainfall\_rate.add(y["key"])

result['Bucket'] = y['bucket']

result['Key'] = y['key']

bucket = s3.Bucket('mypackmog1')

bucket.copy(result, result['Key'])

return {

'statusCode': 200,

'body': result

}

**Without credentials:**

import json

import boto3

sqs = boto3.client('sqs', region\_name='eu-west-2')

s3 = boto3.resource('s3',region\_name='eu-west-2')

rainfall\_rate = set()

result = {}

def lambda\_handler(event, context):

messages = sqs.receive\_message(

QueueUrl='https://sqs.eu-west-2.amazonaws.com/443602378074/MyQueueNew',

MaxNumberOfMessages=10)

for i in messages['Messages']:

x = json.loads(i['Body'])

y = json.loads(x['Message'])

if y['name'] == "rainfall\_rate" and len(rainfall\_rate) <= 4:

rainfall\_rate.add(y["key"])

result["Bucket"] = y['bucket']

result["Key"] = y['key']

bucket = s3.Bucket('mypackmog1')

bucket.copy(result, result['Key'])

return {

'statusCode': 200,

'body': rainfall\_rate

}

Deploying code to AWS