

Work with RXNORM file,

1. Scrap the latest RXNORM file from NLM webpage
2. Download the latest RXNORM file with api\_key
3. Create a log file for the downloaded file
4. Add header into each rff from RXNORM.xlsx
5. Add CODE\_SET & VERSION\_MONTH column with default values RxNorm and version month from downloaded filename
6. Convert dates into YYYY-MM-DD
7. Save files as txt delimited by comma(,)c
8. Validate row\_count between original and converted files

Steps Involved:

- 1) A bucket is created to store the zip file containing the (.RFF) files, Excel file and the files created after transformations and headers are applied.

Amazon S3 > Buckets > Create bucket

## Create bucket [Info](#)

Buckets are containers for data stored in S3.

### General configuration

AWS Region

US East (N. Virginia) us-east-1

Bucket type [Info](#)

☒ **General purpose**  
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

☐ **Directory - New**  
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name [Info](#)

myawsbucket

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

Copy settings from existing bucket - optional  
Only the bucket settings in the following configuration are copied.

Format: s3://bucket/prefix

### Object Ownership [Info](#)

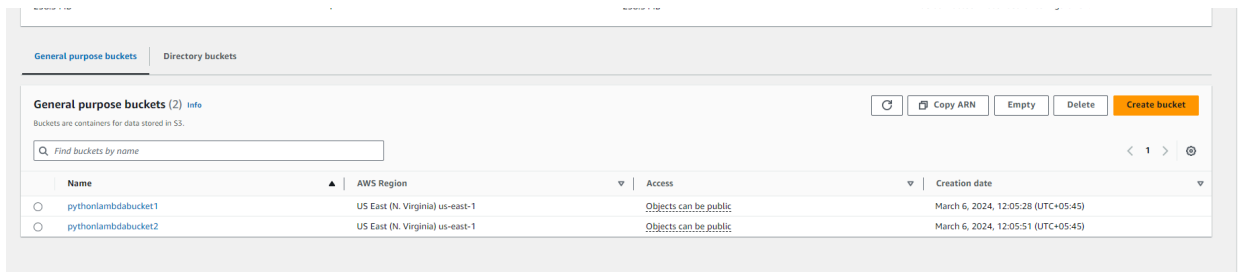
Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☒ **ACLs disabled (recommended)**  
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

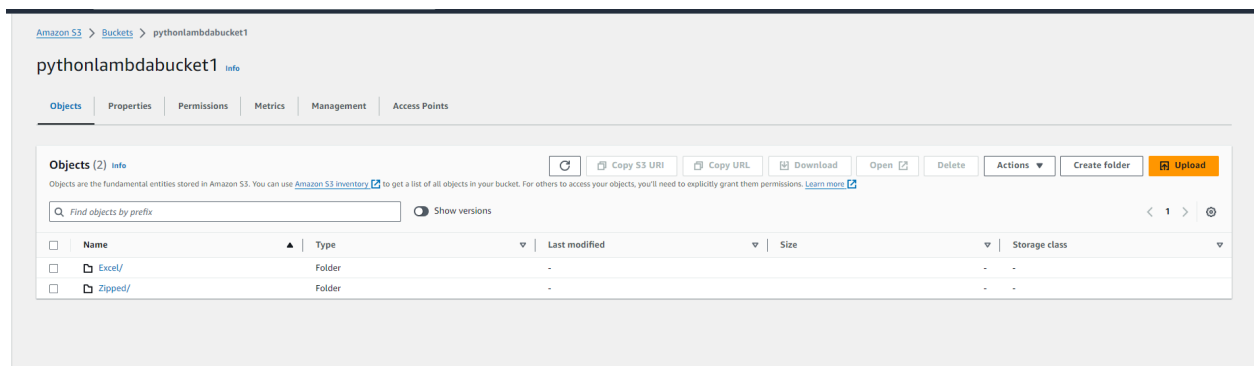
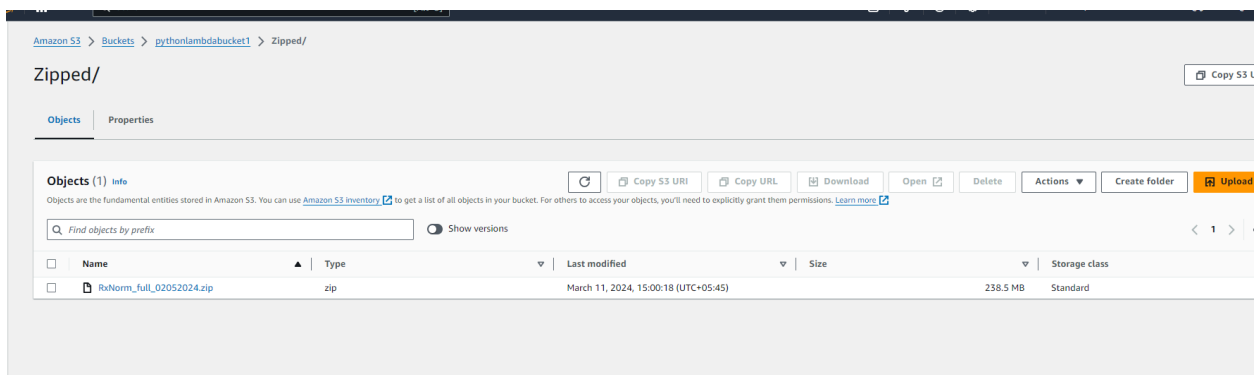
☐ **ACLs enabled**  
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership

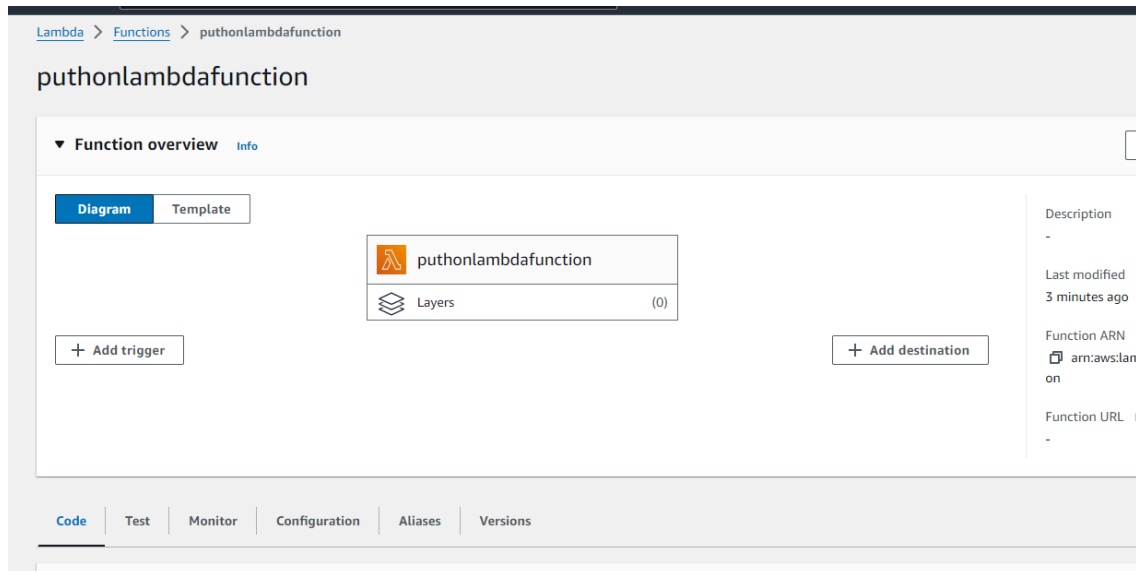
## Buckets created successfully



## 2) Folders for zipped and excel sheets are created and respective file is uploaded in it.

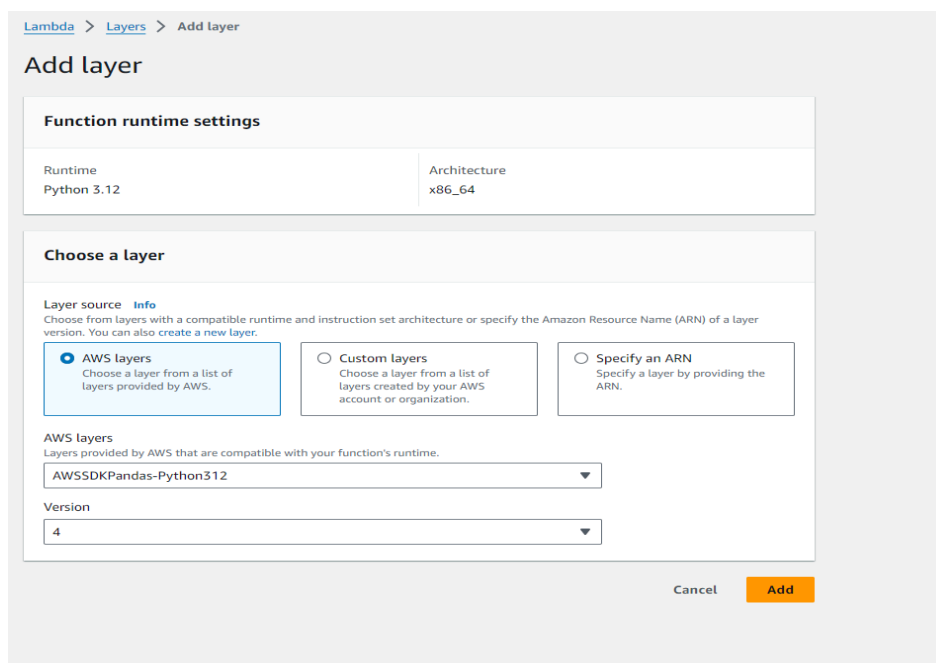


### 3) Creation of AWS Lambda Function



### 4) Adding Lambda Layers

**Pandas Layer is added to the Lambda for processing the files per task requirements**




## 5) Adding Trigger

**Choose Event types – all**

**S3 bucket made previously is added as a trigger to process files.**

**Trigger configuration** [Info](#)

 **S3**  
aws asynchronous storage


**Bucket**  
Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.  
 ✕ ↻  
Bucket region: us-east-1


**Event types**  
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.  

All object create events ✕

**Prefix - optional**  
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters.

**Suffix - optional**  
Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters.

**Recursive invocation**  
If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. [Learn more](#)   
☒ I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.


Lambda will add the necessary permissions for AWS S3 to invoke your Lambda function from this trigger. [Learn more](#)  about the Lambda permissions model.

**Triggered added successfully.**

**Configuration** | Aliases | Versions

**Triggers (1)** [Info](#) ↻ Fix errors Edit Delete Add trigger

☐ Trigger

 **S3: pythonlambdabucket1**  
arn:aws:s3:::pythonlambdabucket1  
[Details](#)

## 6) The complete code is written:

```
io  Tools  Window  Test  Deploy  Changes not deployed

lambda_function x Environment Var x Execution results x +

1 import boto3
2 import zipfile
3 import io
4 import os
5 import pandas as pd
6 from io import BytesIO
7
8 s3 = boto3.client('s3')
9 excel_headers = {}
10
11 def read_excel_from_s3(bucket):
12     try:
13         folder_path = 'excelfiles/'
14         excel_file_name = 'RxNorm_Header.xlsx'
15         key = folder_path + excel_file_name
16
17         # Download the Excel file to the /tmp directory
18         local_excel_file = '/tmp/RxNorm_Header.xlsx'
19         s3.download_file(bucket, key, local_excel_file)
20
21         # Check if the Excel file exists
22         if os.path.exists(local_excel_file):
23             print(f"Excel file downloaded to: {local_excel_file}")
24
25         # Read the Excel file into an ExcelFile object
26         excel_file = pd.ExcelFile(local_excel_file)
27
28         # Get the sheet names
29         sheet_names = excel_file.sheet_names
30         print("Sheet names:", sheet_names)
31
32         for sheets in sheet_names:
33             # Read the data from the sheet into a DataFrame
34             sheets_data = excel_file.parse(sheets, header=None)
35             headers_data = sheets_data.iloc[:, 0].tolist()
36             excel_headers[sheets] = headers_data
37
38         print(f'excel_headers dictionary for sheet {sheet_names[0]}: {excel_headers[sheet_names[0]]}')
39
40     except Exception as e:
41         print(f"Error occurred: {e}")
42
43 def code_set_and_version_month(zip_filename, rrf_df):
44     try:
45         # Extract version month from the filename
46         version_month = os.path.splitext(zip_filename)[0].split('_')[-1]
47
```

```
Go  Tools  Window  Test  Deploy  Changes not deployed

lambda_function x Environment Var x Execution results x +

43 def code_set_and_version_month(zip_filename, rrf_df):
44     try:
45         # Extract version month from the filename
46         version_month = os.path.splitext(zip_filename)[0].split('_')[-1]
47
48         # Convert version month to a more readable format
49         version_month = pd.to_datetime(version_month, format='%m%dY').strftime('%Y-%m-%d')
50         print(f"Version month: {version_month}")
51
52         # Add 'Code Set' and 'Version Month' columns to the DataFrame
53         rrf_df['Code Set'] = 'RxNorm'
54         rrf_df['Version Month'] = version_month
55
56     except Exception as e:
57         print(f"Error occurred while extracting version month: {e}")
58
59     return rrf_df
60
61 def apply_header_to_rrf(file_name, rrf_df):
62     # Check if the corresponding Excel sheet exists
63     if file_name in excel_headers:
64         # Get the headers from the Excel sheet
65         excel_headers_list = excel_headers[file_name]
66         excel_headers_list = [header for header in excel_headers_list if header != 'SVER']
67
68         # Take names from the excel header list up to the length of the split DataFrame
69         excel_headers_list = excel_headers_list[:len(rrf_df.columns)]
70
71         # Set the correct header for the DataFrame
72         rrf_df.columns = excel_headers_list
73
74     return rrf_df
75
76 def convert_date_format(value):
77     try:
78         # Try to parse the value into datetime format
79         parsed_date = pd.to_datetime(value, format='%Y-%m-%d').date()
80
81         # Extract only the date part
82         return parsed_date.strftime('%Y-%m-%d')
83     except ValueError:
84         if value == '2020':
85             return '2020-01-01'
86         elif value == '%s 0 2024 01 04':
87             # Remove the float value and parse the remaining string
88             return convert_date_format('2024_01_04')
89         elif value == '2020AA':
```

```
View Go Tools Window Test Deploy Changes not deployed
-P) lambda_function Environment Var Execution results
nct ion.py
85     return '2020-01-01'
86     elif value == 's.d.2024.01.04':
87         # Remove the float value and parse the remaining string
88         return convert_date_format('2024.01.04')
89     elif value == '202004':
90         return '2024-01-02'
91     elif value == '20AA.240205':
92         return '2024-02-05'
93     else:
94         return value
95
96 def update_nato_date(value):
97     try:
98         # Attempt to parse the value using the first date format
99         parsed_date = pd.to_datetime(value, format='%m/%d/%Y %I:%M:%S %p').date()
100     except ValueError:
101         try:
102             # If the first format fails, attempt to parse using the second date format
103             parsed_date = pd.to_datetime(value, format='%d-%b-%y').date()
104         except ValueError:
105             # If both formats fail, return None or handle the error appropriately
106             return None # Or handle the error appropriately
107
108 # Check if the parsed_date is NaT
109 if pd.isnull(parsed_date):
110     return '0000-00-00' # Replace NaT with '0000-00-00'
111 else:
112     # Extract only the date part and return it in the desired format
113     return parsed_date.strftime('%Y-%m-%d')
114
115 def process_date_columns(file_name, rrf_df):
116     date_columns = ['VSTART', 'VEND', 'CREATED_TIMESTAMP', 'UPDATED_TIMESTAMP', 'LAST_RELEASED']
117
118     for column in date_columns:
119         if column in rrf_df.columns:
120             if file_name == 'XONGAB':
121                 # Apply the conversion function to each value in the column
122                 rrf_df[column] = rrf_df[column].apply(convert_date_format)
123
124             # Extract year from the VSTART column after date conversion and save it directly as a string
125             rrf_df['SVER'] = pd.to_datetime(rrf_df['VSTART'], format='%Y-%m-%d').dt.year.astype(str)
126
127             # Reorder the columns to place 'SVER' before 'VSTART'
128             sver_index = rrf_df.columns.get_loc('SVER')
129             vstart_index = rrf_df.columns.get_loc('VSTART')
130
131             # Remove 'SVER' from its original position
```

## 7) Configuring Test Event

A simple test event is made to check for correct functioning of the code.

### Configure test event

A test event is a JSON object that mocks the structure of requests emitted by AWS services to invoke a Lambda. Use it to see the function's invocation result.

To invoke your function without saving an event, configure the JSON event, then choose Test.

Test event action

☒ Create new event

☐ Edit saved event

Event name

MyEventName

Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Event sharing settings

☒ Private

This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)

☐ Shareable

This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn](#)

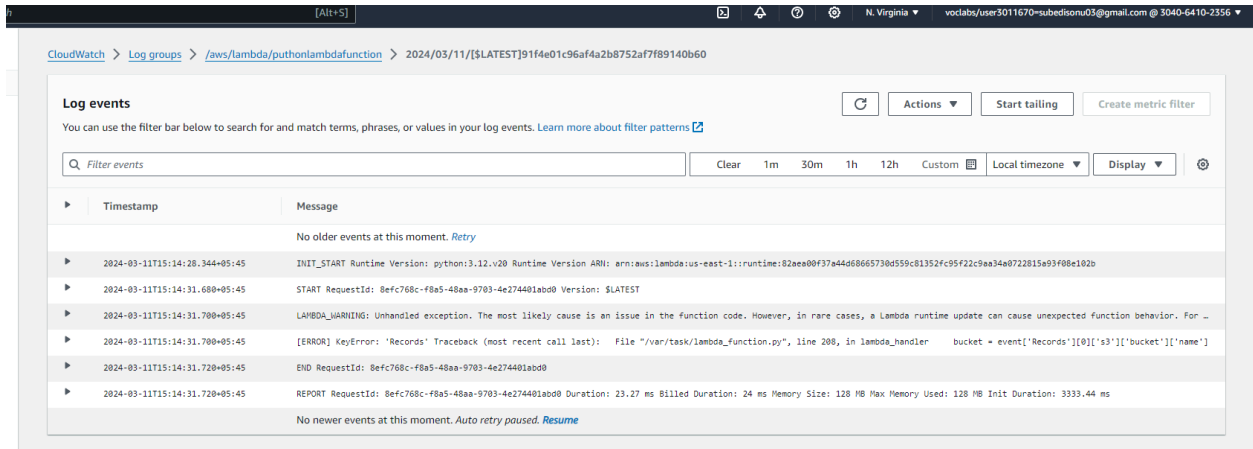
Template - optional

hello-world

Event JSON

Format

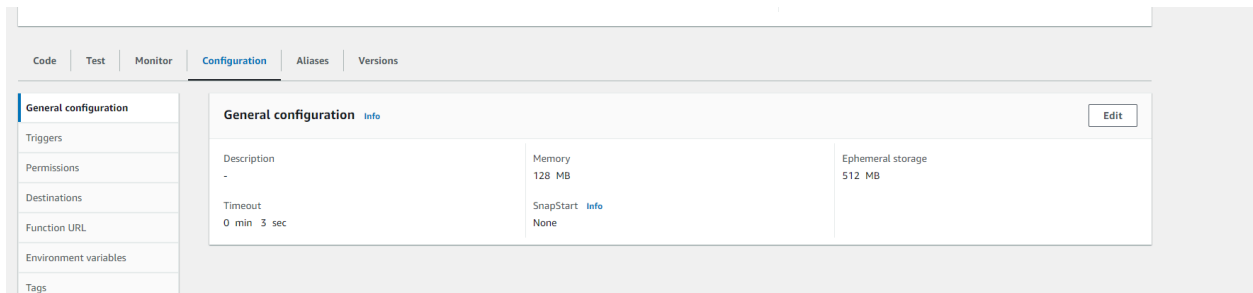
## 8) Checking log events.. The issues persist and are solved accordingly.



The screenshot shows the AWS CloudWatch console interface for a Lambda function's logs. The breadcrumb navigation indicates the path: CloudWatch > Log groups > /aws/lambda/pythonlambdafunction > 2024/03/11/[LATEST]91f4e01c96af4a2b8752af7f89140b60. The 'Log events' section is active, displaying a table of log events. The table has two columns: 'Timestamp' and 'Message'. The events show the function's execution, including an 'INIT\_START' event, a 'START' event, a 'LAMBDA\_WARNING' event, an '[ERROR] KeyError' event, an 'END' event, and a 'REPORT' event. The 'REPORT' event provides details about the function's execution, such as duration (23.27 ms), billed duration (24 ms), memory size (128 MB), and max memory used (128 MB). The interface includes a search bar, a filter bar, and buttons for 'Start tailing' and 'Create metric filter'.

Timestamp	Message
	No older events at this moment. <a href="#">Retry</a>
2024-03-11T15:14:28.344+05:45	INIT_START Runtime Version: python:3.12.v20 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:82aea09f37a44d68665730d559c81352fc95f2c9ea34e072815a93f08e102b
2024-03-11T15:14:31.680+05:45	START RequestId: 8efc768c-f8a5-48aa-9703-4e274401abd0 Version: \$LATEST
2024-03-11T15:14:31.700+05:45	LAMBDA_WARNING: Unhandled exception. The most likely cause is an issue in the function code. However, in rare cases, a Lambda runtime update can cause unexpected function behavior. For ..
2024-03-11T15:14:31.700+05:45	[ERROR] KeyError: 'Records' Traceback (most recent call last): File "/var/task/lambda_function.py", line 208, in lambda_handler bucket = event['Records'][0]['s3']['bucket']['name']
2024-03-11T15:14:31.720+05:45	END RequestId: 8efc768c-f8a5-48aa-9703-4e274401abd0
2024-03-11T15:14:31.720+05:45	REPORT RequestId: 8efc768c-f8a5-48aa-9703-4e274401abd0 Duration: 23.27 ms Billed Duration: 24 ms Memory Size: 128 MB Max Memory Used: 128 MB Init Duration: 3333.44 ms
	No newer events at this moment. Auto retry paused. <a href="#">Resume</a>

## 9) Go to the General Configuration of lambda function and change its basic settings.



The screenshot shows the AWS Lambda console interface for the 'General configuration' of a Lambda function. The breadcrumb navigation indicates the path: Code > Test > Monitor > Configuration > Aliases > Versions. The 'General configuration' tab is active, displaying a table with the function's basic settings. The table has three columns: 'Description', 'Memory', and 'Ephemeral storage'. The 'Description' column shows a hyphen (-). The 'Memory' column shows '128 MB'. The 'Ephemeral storage' column shows '512 MB'. The 'Timeout' is set to '0 min 3 sec'. The 'SnapStart' is set to 'None'. The 'Info' link is visible next to the 'SnapStart' value. The interface includes a search bar, a filter bar, and buttons for 'Edit' and 'Info'.

Description	Memory	Ephemeral storage
-	128 MB	512 MB
Timeout 0 min 3 sec	SnapStart <a href="#">Info</a> None	

Changing Basic Settings. Edit Memory configuration and Timeout as required.

[Lambda](#) > [Functions](#) > [pythonlambdafunction](#) > Edit basic settings

## Edit basic settings

**Basic settings** [Info](#)

Description - *optional*

**Memory** [Info](#)

Your function is allocated CPU proportional to the memory configured.

MB

Set memory to between 128 MB and 10240 MB

**Ephemeral storage** [Info](#)

You can configure up to 10 GB of ephemeral storage (/tmp) for your function. [View pricing](#)

MB

Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.

**SnapStart** [Info](#)

Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the [SnapStart compatibility considerations](#).

None ▼

Supported runtimes: Java 11, Java 17, Java 21.

**Timeout**

min  sec

**Execution role**

Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

☒ Use an existing role

☐ Create a new role from AWS policy templates

**Existing role**

Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

▼

View the [LabRole](#) role on the IAM console.

Cancel

Save



Some of the errors seen in logs of CloudWatch:

CloudWatch > Log group > /aws/lambda/puthonlambdafunction > 2024/03/11/[SLATEST]2:c6f3af96bb04a9f9d239c6e54579bfb

Log events

⌂

Actions

Start tailing

Create metric f

You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

Q Filter events

Clear1m30m1h12hCustomLocal timezoneDisplay

Timestamp	Message
No older events at this moment. <a href="#">Retry</a>	
2024-03-11T16:16:34.983+05:45	INIT_START Runtime Version: python:3.12.v20 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:82aea0ff37a44d686657380559c81352fc95f22c9a834e0722815e893f08e182D
2024-03-11T16:16:37.075+05:45	START RequestId: 17598d2b-3470-4444-939a-6795a3a607a2 Version: SLATEST
2024-03-11T16:16:37.877+05:45	s3 event structure is not as expected. Missing 'Records' key.
2024-03-11T16:16:37.882+05:45	END RequestId: 17598d2b-3470-4444-939a-6795a3a607a2
2024-03-11T16:16:37.882+05:45	REPORT RequestId: 17598d2b-3470-4444-939a-6795a3a607a2 Duration: 4.84 ms Billed Duration: 5 ms Memory Size: 10240 MB Max Memory Used: 190 MB Init Duration: 2889.69 ms
No newer events at this moment. <a href="#">Auto retry paused. Resume</a>	

Log events

⌂

Actions

Start tailing

Create metric f

You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

Q Filter events

Clear1m30m1h12hCustomLocal timezoneDisplay

Timestamp	Message
No older events at this moment. <a href="#">Retry</a>	
2024-03-11T16:16:34.983+05:45	INIT_START Runtime Version: python:3.12.v20 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:82aea0ff37a44d686657380559c81352fc95f22c9a834e0722815e893f08e182D
2024-03-11T16:16:37.075+05:45	START RequestId: 17598d2b-3470-4444-939a-6795a3a607a2 Version: SLATEST
2024-03-11T16:16:37.877+05:45	s3 event structure is not as expected. Missing 'Records' key.
2024-03-11T16:16:37.882+05:45	END RequestId: 17598d2b-3470-4444-939a-6795a3a607a2
2024-03-11T16:16:37.882+05:45	REPORT RequestId: 17598d2b-3470-4444-939a-6795a3a607a2 Duration: 4.84 ms Billed Duration: 5 ms Memory Size: 10240 MB Max Memory Used: 190 MB Init Duration: 2889.69 ms
No newer events at this moment. <a href="#">Auto retry paused. Resume</a>	

10) Resolving errors and testing the code.

Code | Test | Monitor | Configuration | Aliases | Versions

Code source info

Upload from

File Edit Find View Go Tools Window

TestDeploy

Go to Anything (Ctrl-P)

lambda\_function x Environment Var x Execution result x

Environment

puthonlambdafunct

lambda\_function.py

Execution results

Loading...

Execution started: 16:27:35 (1 minute ago)

## 11) Test done successfully.

File Edit Find View Go Tools Window **Test** Deploy

Go to Anything (Ctrl-P)

lambda\_function x Environment Var x Execution result x

▼ Execution results Status: Succeeded Max memory used

Test Event Name  
lamdatest

Response  
null

Function Logs

```
RXN_CDC_DF
3 SCDF,DPG,SBD,PIN,THSY,BN,SBOC,MIN,SBDG,PSN,SCD...
4 FN,SY,PT,PTGB,SYGB
LAT CENC CURVER SABIN SSN \
0 DCSA,DDF,DDFA,DMDC,DPC,DRT,DRTA,DST,NDC,TYPE ENG UTF-8 Y Y
1 DCSA,DDF,DESI_DESC,DRT,DST,LABELER,WWW_RXO,NDC... ENG UTF-8 Y Y
2 DCSA,DDF,DESI_DESC,DRT,DST,LABELER,WWW_RXO,NDC... NDC ENG UTF-8 Y Y
3 AHBIGUTY_FLAG,NDC,ORIG_CODE,ORIG_SOURCE,RXN_A... ENG UTF-8 Y Y
4 NaN ENG UTF-8 Y Y
SCIT \
0 Multum
1 Micromedex
2 FDB MedKnowledge
3 RxNorm work done by the National Library of Me...
4 US Edition of SNOMED CT
Code set Code Set Version Month
0 ;;;Medisource Lexicon;;;January 01, 2024;Denv... RxNorm 2024-02-05
1 ;;;Micromedex RED BOOK;;;January 02, 2024;;;... RxNorm 2024-02-05
2 ;;;FDB MedKnowledge (formerly NDDF Plus);;;Ja... RxNorm 2024-02-05
3 ;;;RxNorm;;;HETA2020AA Full Update 2024_02_05... RxNorm 2024-02-05
4 ;;;International Health Terminology Standards D... RxNorm 2024-02-05
Transformed data saved to: s3://pythonlambdabucket1/transformation/RXNSAB.csv
Unzipped file saved to: s3://pythonlambdabucket1/unzipped/RXNSAB.RRF
The RXNSAB.RRF is read from zip file.
/var/task/lambda_function.py:183: DtypeWarning: Columns (6) have mixed types. Specify dtype option on import or set low_memory=False.
rrf_df = pd.read_csv(file_content_io, delimiter='|', header=None)
```

.\*? aA 1 lambda

Replace With

## 12) Logs Events in Cloudwatch

CloudWatch > Log groups > /aws/lambda/pythonlambdabunction > 2024/03/11/[LATEST]53652df33fd4fac017ace859c4e4d

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

Q Filter events Clear 1m 30m 1h 12h Custom Local timezone Display

Timestamp Message

There are older events to load. [Load more](#)

2024-03-11T16:29:38.524+05:45 The RXNSAB.RRF is read from zip file.

2024-03-11T16:29:42.274+05:45 /var/task/lambda\_function.py:183: DtypeWarning: Columns (6) have mixed types. Specify dtype option on import or set low\_memory=False.

2024-03-11T16:29:42.274+05:45 rrf\_df = pd.read\_csv(file\_content\_io, delimiter='|', header=None)

2024-03-11T16:29:43.111+05:45 Row count before transformation: 1133065

2024-03-11T16:29:43.111+05:45 Version month: 2024-02-05

2024-03-11T16:29:43.124+05:45 Row count of RXNSAB after transformation: 1133065

2024-03-11T16:29:43.129+05:45 RXNUI LAT TS LUI STT SUZ ISPREF RXNUI SAUI SCUI SUI \

2024-03-11T16:29:43.129+05:45 0 3 ENG NaN NaN NaN NaN NaN 8717795 NaN 58488005 NaN

2024-03-11T16:29:43.129+05:45 1 3 ENG NaN NaN NaN NaN NaN 8717796 NaN 58488005 NaN

2024-03-11T16:29:43.129+05:45 2 3 ENG NaN NaN NaN NaN NaN 8717808 NaN 58488005 NaN

2024-03-11T16:29:43.129+05:45 3 3 ENG NaN NaN NaN NaN NaN 8738164 NaN 58488005 NaN

2024-03-11T16:29:43.129+05:45 4 10 ENG NaN NaN NaN NaN NaN 18794404 NaN 112116001 NaN

2024-03-11T16:29:43.129+05:45 SAB TTY CODE STR \

2024-03-11T16:29:43.129+05:45 0 SNOMEDCT\_US PT 58488005 1,4-alpha-Glucan branching enzyme

2024-03-11T16:29:43.129+05:45 1 SNOMEDCT\_US PN 58488005 1,4-alpha-Glucan branching enzyme (substance)

2024-03-11T16:29:43.129+05:45 2 SNOMEDCT\_US SY 58488005 Anyo-(1,4,6)-transglycosylase

2024-03-11T16:29:43.129+05:45 3 SNOMEDCT\_US SY 58488005 Branching enzyme

2024-03-11T16:29:43.129+05:45 4 SNOMEDCT\_US SY 112116001 17-hydrocorticosteroid

### 13) The two files transformed and unzipped created successfully.

pythonlambdabucket1 [Info](#)

[Objects](#) | [Properties](#) | [Permissions](#) | [Metrics](#) | [Management](#) | [Access Points](#)

**Objects (4)** [Info](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

☐ Show versions < 1 > ⚙

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	excelfiles/	Folder	-	-	-
<input type="checkbox"/>	transformation/	Folder	-	-	-
<input type="checkbox"/>	unzipped/	Folder	-	-	-
<input type="checkbox"/>	zipfiles/	Folder	-	-	-

### 14) Transformation

After execution, headers are added, delimiter is changed to comma(,), and date format is changed.

[Amazon S3](#) > [Buckets](#) > [pythonlambdabucket1](#) > transformation/

transformation/ [Copy S3 URI](#)

[Objects](#) | [Properties](#)

**Objects (9)** [Info](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

☐ Show versions < 1 > ⚙

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	<a href="#">RXNATOMARCHIVE.csv</a>	csv	March 11, 2024, 16:29:38 (UTC+05:45)	69.6 MB	Standard
<input type="checkbox"/>	<a href="#">RXNCONSO.csv</a>	csv	March 11, 2024, 16:29:54 (UTC+05:45)	138.5 MB	Standard
<input type="checkbox"/>	<a href="#">RXNCUI.csv</a>	csv	March 11, 2024, 16:29:57 (UTC+05:45)	2.1 MB	Standard
<input type="checkbox"/>	<a href="#">RXNCUICHANGES.csv</a>	csv	March 11, 2024, 16:29:57 (UTC+05:45)	17.9 KB	Standard
<input type="checkbox"/>	<a href="#">RXNDOC.csv</a>	csv	March 11, 2024, 16:29:57 (UTC+05:45)	271.7 KB	Standard
<input type="checkbox"/>	<a href="#">RXNREL.csv</a>	csv	March 11, 2024, 16:31:15 (UTC+05:45)	645.2 MB	Standard
<input type="checkbox"/>	<a href="#">RXNSAB.csv</a>	csv	March 11, 2024, 16:31:28 (UTC+05:45)	10.3 KB	Standard
<input type="checkbox"/>	<a href="#">RXNSAT.csv</a>	csv	March 11, 2024, 16:32:29 (UTC+05:45)	622.4 MB	Standard
<input type="checkbox"/>	<a href="#">RXNSTY.csv</a>	csv	March 11, 2024, 16:32:45 (UTC+05:45)	26.1 MB	Standard

# 15) Unzipped

The zipped file is unzipped successfully.

Amazon S3 > Buckets > pythonlambdabucket1 > unzipped/

unzipped/

Copy S3 URI










Objects | Properties

Objects (9) Info

Refresh Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix Show versions

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	 <a href="#">RXNATOMARCHIVE.RRF</a>	RRF	March 11, 2024, 16:29:39 (UTC+05:45)	71.4 MB	Standard
<input type="checkbox"/>	 <a href="#">RXNCONSO.RRF</a>	RRF	March 11, 2024, 16:29:56 (UTC+05:45)	118.6 MB	Standard
<input type="checkbox"/>	 <a href="#">RXNCUL.RRF</a>	RRF	March 11, 2024, 16:29:57 (UTC+05:45)	1.7 MB	Standard
<input type="checkbox"/>	 <a href="#">RXNCUICHANGES.RRF</a>	RRF	March 11, 2024, 16:29:57 (UTC+05:45)	14.9 KB	Standard
<input type="checkbox"/>	 <a href="#">RXNDOC.RRF</a>	RRF	March 11, 2024, 16:29:57 (UTC+05:45)	214.2 KB	Standard
<input type="checkbox"/>	 <a href="#">RXNREL.RRF</a>	RRF	March 11, 2024, 16:31:23 (UTC+05:45)	484.4 MB	Standard
<input type="checkbox"/>	 <a href="#">RXNSAB.RRF</a>	RRF	March 11, 2024, 16:31:28 (UTC+05:45)	9.8 KB	Standard
<input type="checkbox"/>	 <a href="#">RXNSAT.RRF</a>	RRF	March 11, 2024, 16:32:37 (UTC+05:45)	498.7 MB	Standard
<input type="checkbox"/>	 <a href="#">RXNSTY.RRF</a>	RRF	March 11, 2024, 16:32:45 (UTC+05:45)	18.4 MB	Standard