

Task 2: Python and Lambda

Work with RXNORM file,

- ~~Scrap the latest RXNORM file from NLM webpage.~~
- ~~Download the latest RXNORM file with api_key~~
- ~~Create a log file for the downloaded file.~~
- Add header into each rff from RXNORM.xlsx.
- Add CODE_SET & VERSION_MONTH column with default values RxNorm and version month from downloaded filename.
- Convert dates into YYYY-MM-DD
- Save files as txt delimited by comma (,)
- Validate row_count between original and converted files.

1. Creating Bucket

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.

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](#)

pratikzippedbucket

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.

[Choose bucket](#)

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

Bucket owner enforced

2. Bucket Created Successfully

Amazon S3 > Buckets > praktikzippedbucket

pratikzippedbucket [Info](#)

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

Objects (0) [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](#)

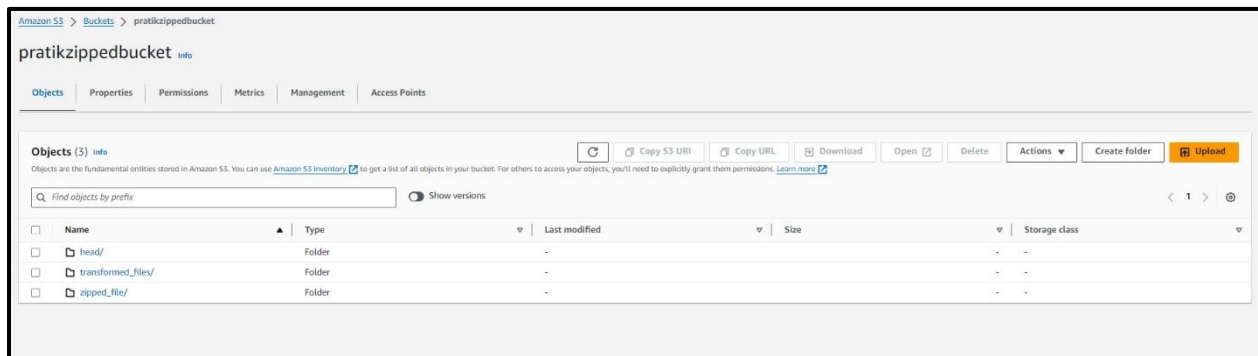
☐ Show versions

[Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

Name	Type	Last modified	Size	Storage class
No objects				
You don't have any objects in this bucket.				
Upload				

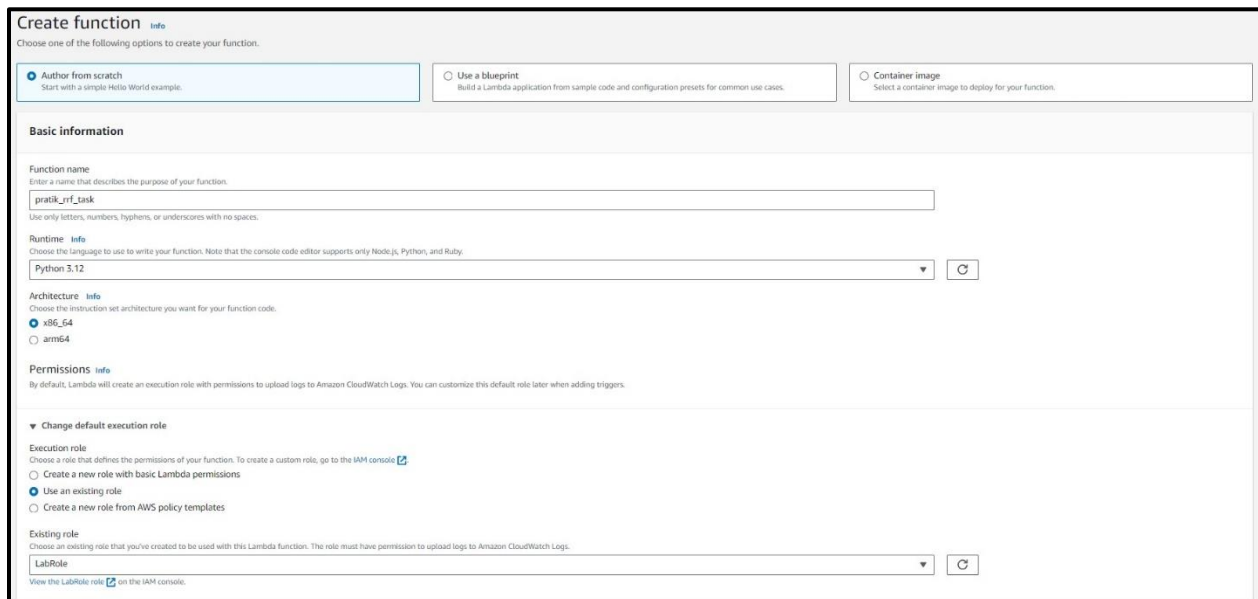
3. Directories Created

Multiple directories are created to store the Zip file, Excel file and the Transformed files.

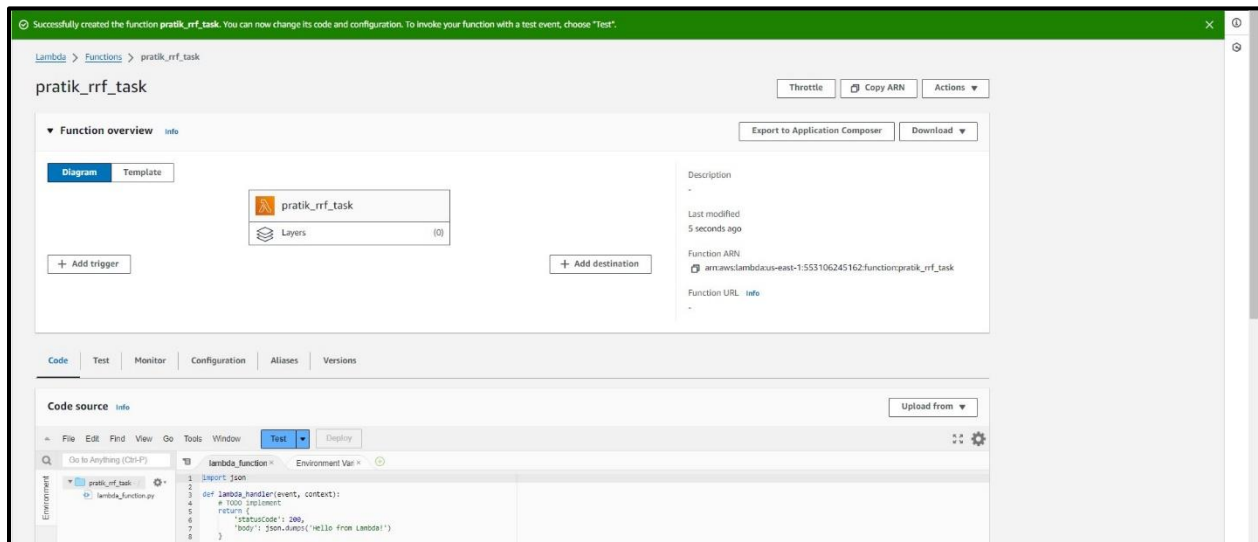


4. AWS Lambda Function

A Lambda Function is created for the processing the files per task requirements. Python runtime is selected and LabRole permission is assigned,

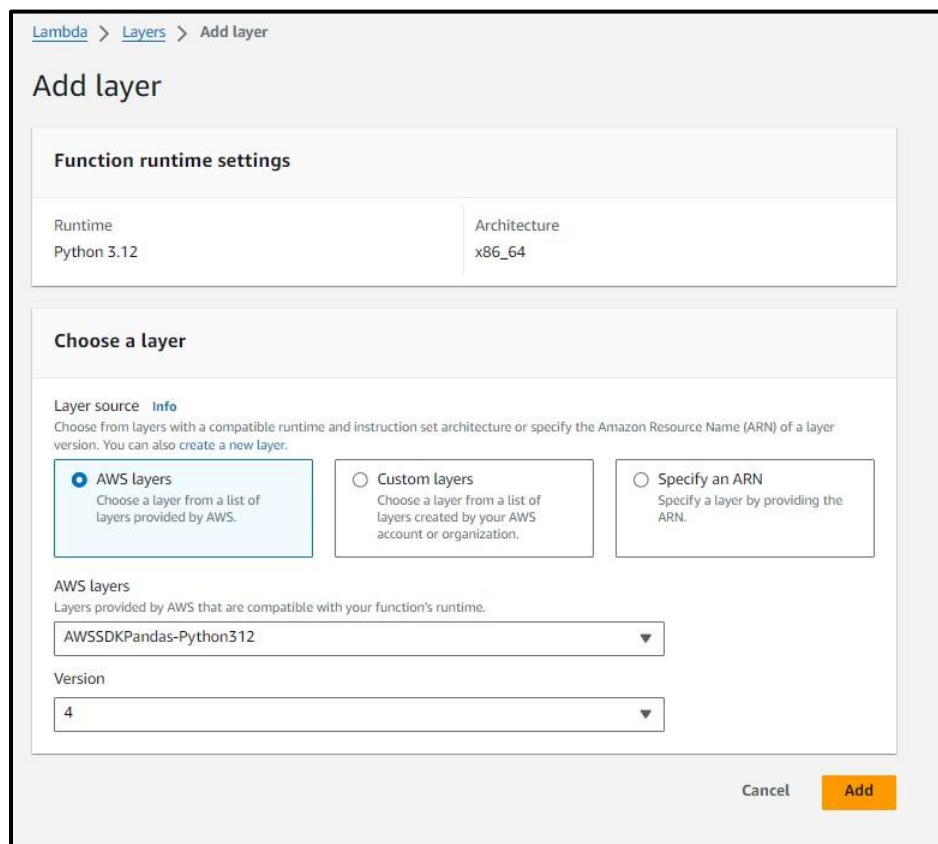


5. AWS Lambda Created



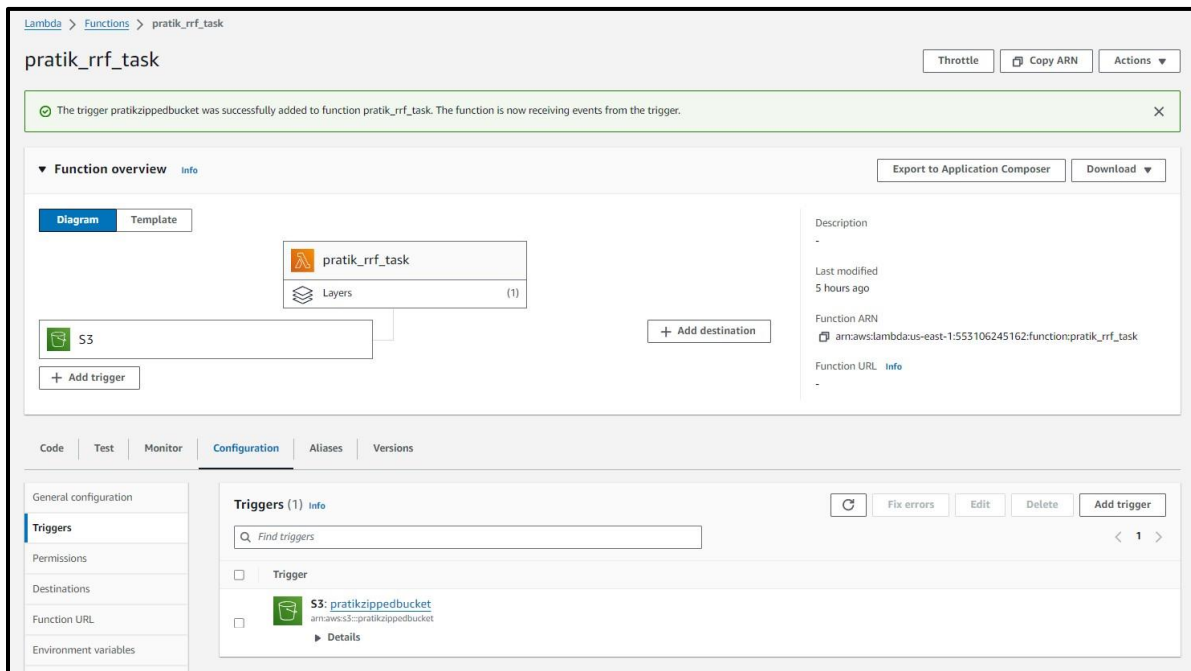
6. Lambda Layers

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



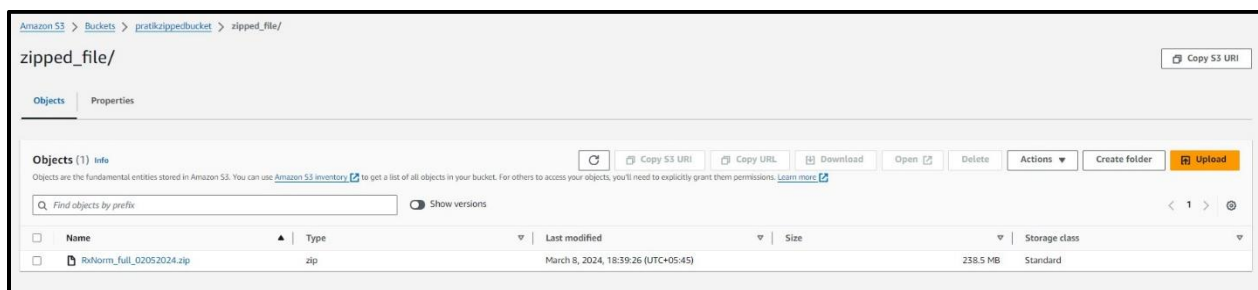
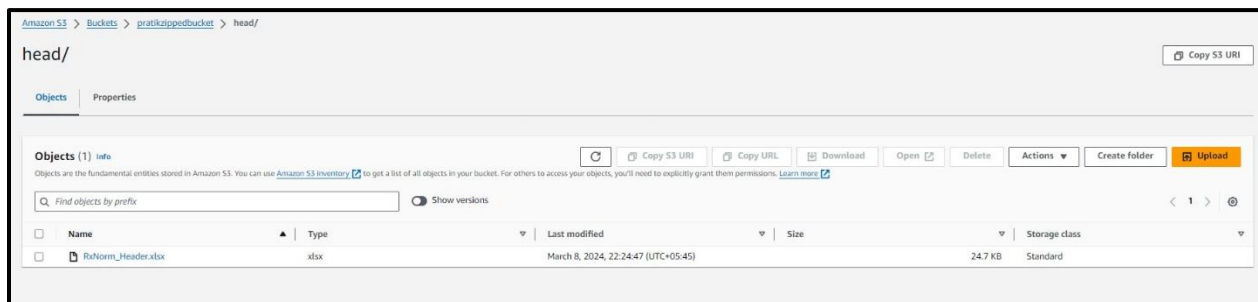
7. Adding Trigger

Previously created S3 bucket is added as trigger to process the files in AWS Lambda



8. Uploading Required Files

All files required are uploaded.



9. 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 function. 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 more](#)

Template - optional

hello-world

Event JSON

Format JSON

```

1 {
2   "key1": "value1",
3   "key2": "value2",
4   "key3": "value3"
5 }

```

Cancel

Invoke

Save

10. Reading Zip File

Code to Read Zip file.

Tools

Window

Test

Display

lambda_function

```

1 import json
2 import boto3
3 from zipfile import ZipFile
4 import pandas as pd
5 from io import BytesIO
6 from datetime import datetime
7 import os
8
9 s3_client = boto3.client('s3')
10
11 def lambda_handler(event, context):
12     try:
13         # Extract bucket and key information from the S3 event
14         bucket = 'pratikzipbucket'
15         zip_key = 'zipfile/low.zip'
16         # bucket = event['Records'][0]['s3']['bucket']['name']
17         # key = event['Records'][0]['s3']['object']['key']
18
19         # Read the zip file directly from S3
20         zip_file_obj = s3_client.get_object(Bucket=bucket, Key=zip_key)
21         zip_file_content = BytesIO(zip_file_obj['Body'].read())
22
23         print("Zipper")
24         # Perform RxNorm transformation
25         rxnorm_transformation(zip_file_content, bucket, key)
26
27     except Exception as e:
28         print(f"An error occurred: {e}")
29         return {
30             'statusCode': 500,
31             'body': json.dumps('Error occurred during transformation')
32         }

```

Execution results

Environment Var

Status: Succeeded

Max memory used: 128 MB

Time: 880.02 ms

Test Event Name

CheckingCode

Response

null

Function Logs

```

START RequestID: 3a054f06-8502-4a7f-8294-06695516a7c0 Version: SLATEST
zipper
END RequestID: 3a054f06-8502-4a7f-8294-06695516a7c0
REPORT RequestID: 3a054f06-8502-4a7f-8294-06695516a7c0 Duration: 888.02 ms Billed Duration: 861 ms Memory Size: 128 MB Max Memory Used: 12
Request ID
3a054f06-8502-4a7f-8294-06695516a7c0

```

11. Complete Code

Code is Completed and tested. It shows the execution result on the right.

The screenshot shows the AWS Lambda console interface. The 'Code source' tab is active, displaying a Python script for a Lambda function. The script includes a function to validate row counts and another to find Excel files in an S3 bucket. The 'Execution result' tab on the right shows the output of the function, indicating that various transformed files were successfully read and their row counts were validated.

```

157 row_count_validation(bucket)
158
159 except Exception as e:
160     print(f"An error occurred during transformation: {e}")
161     raise e
162
163
164 # Function For Finding Excel File In The Bucket
165 def find_xlsx_in_bucket(bucket, file_name):
166     try:
167         response = s3_client.list_objects_v2(Bucket=bucket,
168         for obj in response.get('Contents', []):
169             if obj['Key'] == file_name:
170                 print(obj['Key']) #Printing Object Key
171                 return obj['Key']
172
173     return None
174 except Exception as e:
175     print(f"An error occurred while finding the rrf file")
176     raise e
177
  
```

Execution results:

```

transformed_files/RXNCUI.txt file found
Successfully Read txt File: transformed_files/RXNCUI.txt
Row Count: 38847
transformed_files/RXNCUI.txt Row Count: 38847
transformed_files/RXNCUICHANGES.txt file found
Successfully Read txt File: transformed_files/RXNCUICHANGES.txt
Row Count: 154
transformed_files/RXNCUICHANGES.txt Row Count: 154
transformed_files/RXNDOC.txt file found
Successfully Read txt File: transformed_files/RXNDOC.txt
Row Count: 3446
transformed_files/RXNDOC.txt Row Count: 3446
transformed_files/RXNREL.txt file found
Successfully Read txt File: transformed_files/RXNREL.txt
Row Count: 7154387
transformed_files/RXNREL.txt Row Count: 7154387
transformed_files/RXNSAB.txt file found
Successfully Read txt File: transformed_files/RXNSAB.txt
Row Count: 14
transformed_files/RXNSAB.txt Row Count: 14
transformed_files/RXNSAT.txt file found
Successfully Read txt File: transformed_files/RXNSAT.txt
Row Count: 7222485
transformed_files/RXNSAT.txt Row Count: 7222485
transformed_files/RXNSTY.txt file found
Successfully Read txt File: transformed_files/RXNSTY.txt
Row Count: 461875
transformed_files/RXNSTY.txt Row Count: 461875
transformed_files/RXNSTY.txt N/A
txt File Row Count: ('transformed_files/RXNATOWHARCHIVE.txt': 371369, 'transformed_files/RXNCON
END RequestId: cc2b5c91-a1d9-42d3-ad6e-9b25d9825f6a Duration: 251593.49 ms Billed Duration:
REPORT RequestId: cc2b5c91-a1d9-42d3-ad6e-9b25d9825f6a
  
```

12. Transformed Files

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

The screenshot shows the Amazon S3 console interface for a bucket named 'pratikkipedbucket'. The 'transformed_files/' prefix is selected, displaying a list of 9 objects. Each object is a text file with a specific name, type, last modified date, size, and storage class.

Name	Type	Last modified	Size	Storage class
RXNATOWHARCHIVE.txt	txt	March 9, 2024, 22:23:47 (UTC+05:45)	78.9 MB	Standard
RXNCONSO.txt	txt	March 9, 2024, 22:23:56 (UTC+05:45)	129.8 MB	Standard
RXNCUI.txt	txt	March 9, 2024, 22:23:57 (UTC+05:45)	1.9 MB	Standard
RXNCUICHANGES.txt	txt	March 9, 2024, 22:23:57 (UTC+05:45)	16.7 KB	Standard
RXNDOC.txt	txt	March 9, 2024, 22:23:57 (UTC+05:45)	244.9 KB	Standard
RXNREL.txt	txt	March 9, 2024, 22:24:48 (UTC+05:45)	590.6 MB	Standard
RXNSAB.txt	txt	March 9, 2024, 22:24:53 (UTC+05:45)	10.4 KB	Standard
RXNSAT.txt	txt	March 9, 2024, 22:25:31 (UTC+05:45)	567.3 MB	Standard
RXNSTY.txt	txt	March 9, 2024, 22:25:38 (UTC+05:45)	22.5 MB	Standard

13. CloudWatch Logs

CloudWatch Logs for the execution. Below is screenshots for the start and end of the logs. The logs itself is uploaded in a different file.

CloudWatch > Log groups > /aws/lambda/pratik_crf_task > 2024/03/09/[SLATEST]b89bd6339f2d4e4695e67b34cc336798

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
	No older events at this moment. <i>Retry</i>
2024-03-09T22:12:22.860+05:45	INIT_START Runtime Version: python:3.12.v20 Runtime Version ARN: arn:aws:lambda:us-east-1:runtime:82ae0bf3744d66665730d559c813d2fc9f22c9a34a0722815a93f08e102b
2024-03-09T22:12:29.382+05:45	START RequestID: cc2b5c91-81d9-42d3-8d6e-9025d9825f6a Version: SLATEST
2024-03-09T22:12:29.524+05:45	Zipper
2024-03-09T22:12:29.524+05:45	Transformation Start
2024-03-09T22:12:29.535+05:45	Sorted Files: ['RXNATOWHARCHIVE.RRF', 'RXNCONSO.RRF', 'RXNUII.RRF', 'RXNUICHANGES.RRF', 'RXNDOC.RRF', 'RXNREL.RRF', 'RXNSAB.RRF', 'RXNSAT.RRF', 'RXNSTV.RRF']
2024-03-09T22:12:29.543+05:45	head/RXNORM_header.xlsx
2024-03-09T22:12:29.572+05:45	Reading RRF file: RXNATOWHARCHIVE.RRF
2024-03-09T22:12:31.467+05:45	Read RRF As CSV Successfully
2024-03-09T22:12:31.467+05:45	Reading RRF file: RXNCONSO.RRF
2024-03-09T22:12:35.956+05:45	Read RRF As CSV Successfully
2024-03-09T22:12:35.968+05:45	Reading RRF file: RXNUII.RRF
2024-03-09T22:12:36.003+05:45	Read RRF As CSV Successfully
2024-03-09T22:12:36.069+05:45	Reading RRF file: RXNUICHANGES.RRF
2024-03-09T22:12:36.071+05:45	Read RRF As CSV Successfully
2024-03-09T22:12:36.071+05:45	Reading RRF file: RXNDOC.RRF
2024-03-09T22:12:36.077+05:45	Read RRF As CSV Successfully
2024-03-09T22:12:36.077+05:45	Reading RRF file: RXNREL.RRF
2024-03-09T22:12:49.439+05:45	Read RRF As CSV Successfully
2024-03-09T22:12:49.439+05:45	Reading RRF file: RXNSAB.RRF

2024-03-09T22:15:47.203+05:45	Row Count: 154
2024-03-09T22:15:47.203+05:45	transformed_files/RXNUICHANGES.txt Row Count: 154
2024-03-09T22:15:47.203+05:45	transformed_files/RXNDOC.txt file found
2024-03-09T22:15:47.303+05:45	Successfully Read txt file: transformed_files/RXNDOC.txt
2024-03-09T22:15:47.303+05:45	Row Count: 3446
2024-03-09T22:15:47.303+05:45	transformed_files/RXNDOC.txt Row Count: 3446
2024-03-09T22:15:47.303+05:45	transformed_files/RXNREL.txt file found
2024-03-09T22:16:15.692+05:45	Successfully Read txt file: transformed_files/RXNREL.txt
2024-03-09T22:16:15.692+05:45	Row Count: 7154307
2024-03-09T22:16:15.692+05:45	transformed_files/RXNREL.txt Row Count: 7154307
2024-03-09T22:16:15.692+05:45	transformed_files/RXNSAB.txt file found
2024-03-09T22:16:16.279+05:45	Successfully Read txt file: transformed_files/RXNSAB.txt
2024-03-09T22:16:16.279+05:45	Row Count: 14
2024-03-09T22:16:16.279+05:45	transformed_files/RXNSAB.txt Row Count: 14
2024-03-09T22:16:16.279+05:45	transformed_files/RXNSAT.txt file found
2024-03-09T22:16:35.370+05:45	Successfully Read txt file: transformed_files/RXNSAT.txt
2024-03-09T22:16:35.370+05:45	Row Count: 7222485
2024-03-09T22:16:35.370+05:45	transformed_files/RXNSAT.txt Row Count: 7222485
2024-03-09T22:16:35.370+05:45	transformed_files/RXNSTV.txt file found
2024-03-09T22:16:36.475+05:45	Successfully Read txt file: transformed_files/RXNSTV.txt
2024-03-09T22:16:36.475+05:45	Row Count: 461875
2024-03-09T22:16:36.475+05:45	transformed_files/RXNSTV.txt Row Count: 461875
2024-03-09T22:16:36.475+05:45	transformed_files/RXNSTV.txt N/A
2024-03-09T22:16:36.475+05:45	txt file Row Count: ['transformed_files/RXNATOWHARCHIVE.txt': 371389, 'transformed_files/RXNCONSO.txt': 1133866, 'transformed_files/RXNUII.txt': 38047, 'transformed_files/RXNUICHANGES.txt': 154, 'transformed_files/R...
2024-03-09T22:16:36.977+05:45	END RequestID: cc2b5c91-81d9-42d3-8d6e-9025d9825f6a
2024-03-09T22:16:36.977+05:45	REPORT RequestID: cc2b5c91-81d9-42d3-8d6e-9025d9825f6a Duration: 251593.49 ms Billed Duration: 251594 ms Memory Size: 8280 MB Max Memory Used: 8053 MB Init Duration: 2528.26 ms

No newer events at this moment. *Auto retry paused. Resume*

Back to top

14. Architecture

