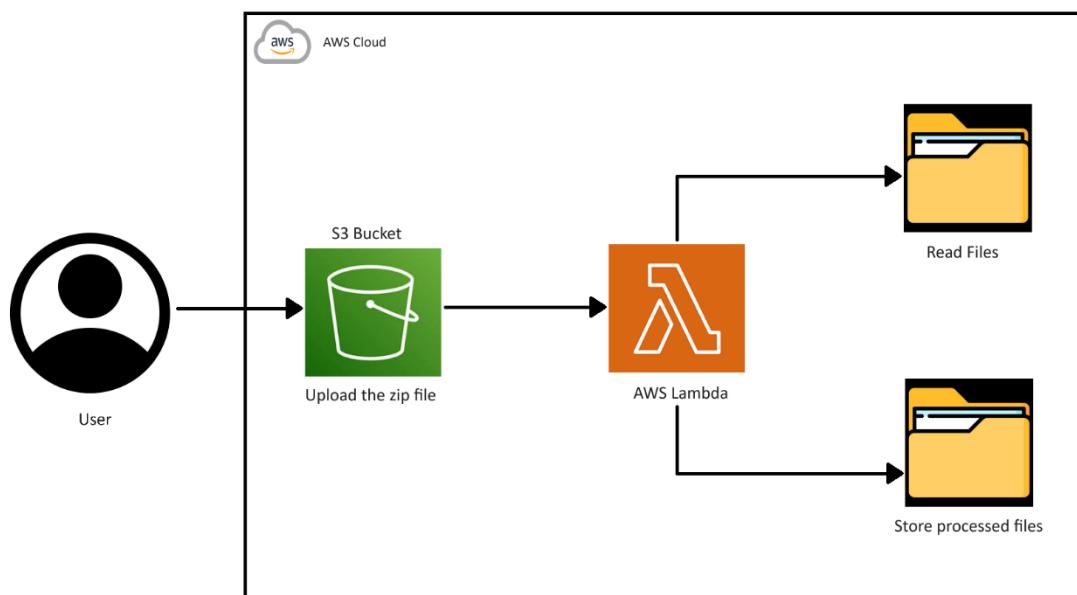


## Task 2 : Python and Lambda

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. While uploading the file to S3, create CloudWatch log file
5. Add header into each rff from RXNORM.xlsx
6. Add CODE\_SET & VERSION\_MONTH column with default values RxNorm and version month from downloaded filename
7. Convert dates into YYYY-MM-DD
8. Save files as txt delimited by comma(,)c
9. Validate row\_count between original and converted files

### Cloud Architecture of this Task



First we create a Lambda Function with following configurations:

Author from scratch Start with a simple Hello World example.

Use a blueprint Build a Lambda application from sample code and configuration presets for common use cases.

Container image Select a container image to deploy for your function.

Function name Enter a name that describes the purpose of your function.  
rrf\_lambdaa

Runtime info Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.  
Python 3.12

Architecture info Choose the instruction set architecture you want for your function code.  
x86\_64

Permissions info By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

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

Create a new role with basic Lambda permissions

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.  
Lambda

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

Advanced settings

Cancel Create function

Secondly, we need to add Layers to our Lambda.

Open our newly created Lambda function, and scroll down. In the bottom of the page, we can see Layers tab. Here, we need to Add a Layer.

Merge order	Name	Layer version	Compatible runtimes	Compatible architectures	Version ARN
There is no data to display.					

Layers Info

Edit Add a layer

Here, select AWSSDKPandas-Python312 and click Add. Also choose Version as 4.

Lambda > Layers > Add layer

## Add layer

### Function runtime settings

Runtime Python 3.12	Architecture x86_64
------------------------	------------------------

#### AWS provided

- AWS-AppConfig-Extension Feature Flags, or other configuration data, for your AWS Lambda functions
- LambdaInsightsExtension CloudWatch Lambda Insights Extension.
- AWS-Parameters-and-Secrets-Lambda-Extension Retrieve parameters from AWS Systems Manager Parameter Store and secrets from AWS Secrets Manager
- AWSSDKPandas-Python312** AWS SDK for pandas Lambda Layer - 3.6.0 (Python 3.12)

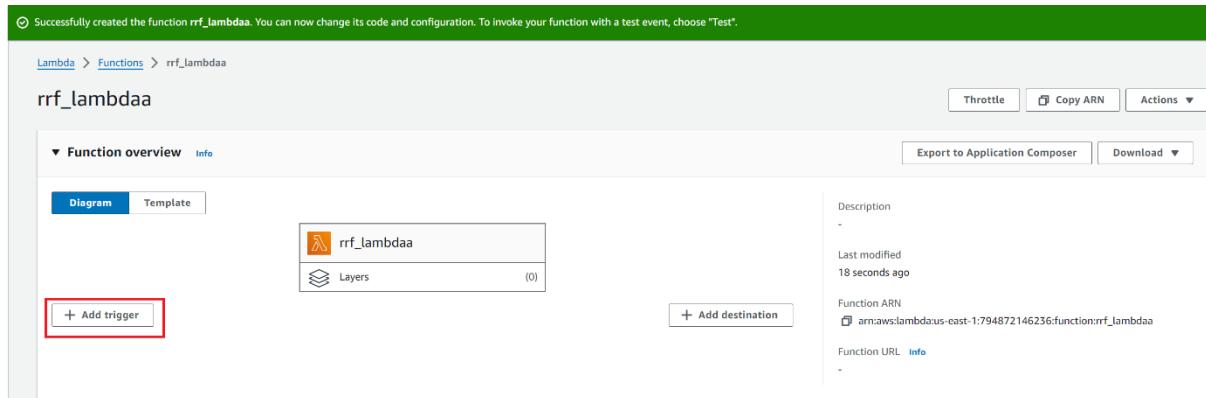
Specify the Amazon Resource Name (ARN) of a layer

Specify an ARN Specify a layer by providing the ARN.

Choose

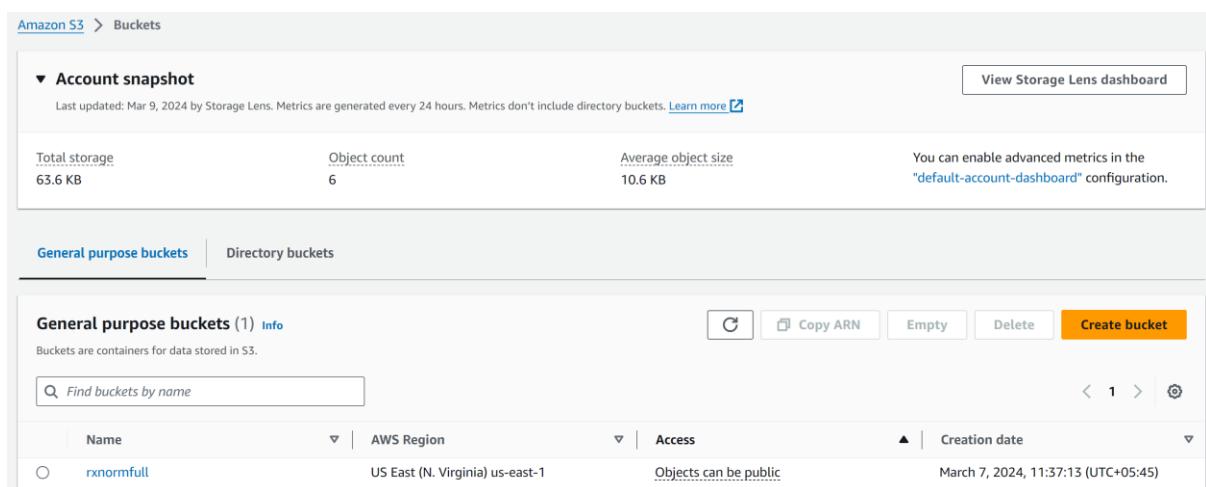
Cancel Add

Next step, is to add a Trigger to our Lambda



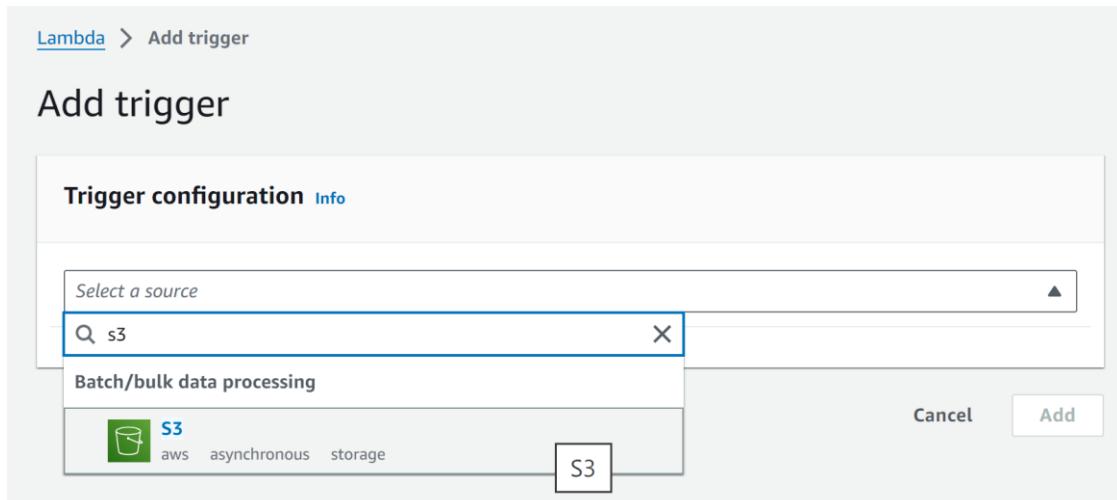
The screenshot shows the AWS Lambda Functions console. A green banner at the top indicates that the function 'rrf\_lambdaaa' has been successfully created. Below the banner, the function name 'rrf\_lambdaaa' is displayed. Under the 'Function overview' section, there is a 'Diagram' tab which is selected, showing a single Lambda function icon labeled 'rrf\_lambdaaa'. To the right of the diagram, there are buttons for 'Throttle', 'Copy ARN', and 'Actions'. Below the diagram, there is a 'Layers' section with '(0)' listed. At the bottom left of the overview section, there is a red box highlighting the '+ Add trigger' button. On the right side of the screen, there are sections for 'Description', 'Last modified', 'Function ARN', and 'Function URL'.

We need to create S3 bucket for this. I have created a bucket.



The screenshot shows the AWS Amazon S3 Buckets page. At the top, it displays an 'Account snapshot' with metrics like Total storage (63.6 KB), Object count (6), and Average object size (10.6 KB). There is a link to 'View Storage Lens dashboard'. Below this, there are tabs for 'General purpose buckets' (selected) and 'Directory buckets'. The 'General purpose buckets' section shows a table with one item: 'rxnormfull'. The table columns include Name, AWS Region, Access, and Creation date. The 'rxnormfull' entry has a status of 'Objects can be public' and was created on March 7, 2024, at 11:37:13 (UTC+05:45). Action buttons for Copy ARN, Empty, Delete, and Create bucket are shown above the table.

Now go back to Lambda and Select S3 in Trigger Configuration



The screenshot shows the 'Add trigger' configuration page. It starts with a breadcrumb navigation 'Lambda > Add trigger'. Below this, the title 'Add trigger' is displayed. Underneath, the 'Trigger configuration' section is shown with a 'Trigger configuration' link. A search bar at the top of the configuration panel contains the text 's3'. A dropdown menu lists 'Batch/bulk data processing' and 'S3'. The 'S3' option is selected and highlighted with a blue border. To the right of the configuration panel, there are 'Cancel' and 'Add' buttons.

[Lambda](#) > Add trigger

## Add trigger

**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.

[X](#) [C](#)

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 [X](#)

**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.

[Cancel](#) [Add](#)

Now this is how our Lambda looks

[Lambda](#) > [Functions](#) > rrf\_lambda

rrf\_lambda

[Throttle](#) [Copy ARN](#) [Actions ▾](#)

[Function overview](#) [Info](#) [Export to Application Composer](#) [Download ▾](#)

[Diagram](#) [Template](#)

Layers	
rrf_lambda	(1)

[+ Add destination](#)

[S3](#) [+ Add trigger](#)

Description  
-

Last modified  
10 hours ago

Function ARN  
[arn:aws:lambda:us-east-1:794872146236:function:rrf\\_lambda](#)

Function URL [Info](#)

In our bucket, two folders i.e, excelfiles/ and zipfiles/ are pre-created whereas transformations/ and unzippedfiles/ are created after the lambda is triggered and files are read

The screenshot shows the Amazon S3 console interface. At the top, the path is "Amazon S3 > Buckets > rxnormfull". Below this, the bucket name "rxnormfull" is shown with a "Info" link. A navigation bar includes tabs for "Objects", "Properties", "Permissions", "Metrics", "Management", and "Access Points". The "Objects" tab is active. A sub-header "Objects (4) Info" is present. A table lists the objects:

Name	Type	Last modified	Size	Storage class
excelfiles/	Folder	-	-	-
transformation/	Folder	-	-	-
unzipped/	Folder	-	-	-
zipfiles/	Folder	-	-	-

We have to

The screenshot shows the AWS Lambda "Edit basic settings" page. The path is "Lambda > Functions > test\_lambda > Edit basic settings". The "Basic settings" tab is selected. The configuration fields include:

- Description - optional:** A text input field.
- Memory**: Set to 10000 MB. Info: Your function is allocated CPU proportional to the memory configured.
- Ephemeral storage**: Set to 10000 MB. Info: You can configure up to 10 GB of ephemeral storage (/tmp) for your function. View pricing.
- SnapStart**: Set to None. 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.
- Timeout**: Set to 8 min 0 sec.
- Execution role**: Choose an existing role. Options: Use an existing role (selected), 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. The dropdown shows "LabRole".

At the bottom right are "Cancel" and "Save" buttons.

We have to import the following:

```

1 import boto3
2 import zipfile
3 import io
4 import os
5 import pandas as pd
6 import json
7 import openpyxl
8 from io import BytesIO
9

```

excel\_headers holds the headers to be assigned to the rrf files

The following function (read\_excel\_from\_s3) reads the excel files and updates the header

```

15 def read_excel_from_s3(bucket):
16     try:
17         folder_path = 'excelfiles/'
18         excel_file_name = 'RxNorm_Header.xlsx'
19         key = folder_path + excel_file_name
20
21         # Download the Excel file to the /tmp directory
22         local_excel_file = '/tmp/RxNorm_Header.xlsx'
23         s3.download_file(bucket, key, local_excel_file)
24
25         # Check if the Excel file exists
26         if os.path.exists(local_excel_file):
27             print(f"Excel file downloaded to: {local_excel_file}")
28
29
30
31         # Read the Excel file into an ExcelFile object
32         excel_file = pd.ExcelFile(local_excel_file)
33
34         # Get the sheet names
35         sheet_names = excel_file.sheet_names
36         print("Sheet names:", sheet_names)
37
38         for sheets in sheet_names:
39             # Read the data from the sheet into a DataFrame
40             sheets_data = excel_file.parse(sheets, header=None)
41             headers_data = sheets_data.iloc[:, 0].tolist()
42             excel_headers[sheets] = headers_data
43
44         print(f"excel_headers dictionary for sheet {sheet_names[0]}: {excel_headers[sheet_names[0]]}")

```

52:9 Python Spaces: 4 ⚡

The following function incorporate the Code Set and Version Mont: The Code Set has been designated as RXNORM, and the Version Month has been derived from the zip file name, specifically RxNorm\_full\_02052024.zip. As a result, the Version Month is identified as 2024-05-02.

Secondly, apply\_header\_to\_rrf adds header to rrf files.

```

51 def code_set_and_version_month(zip_filename, rrf_df):
52     try:
53         # Convert the zip data to a string to extract information
54         version_month = os.path.splitext(zip_filename)[0].split('_')[-1]
55
56         # Convert version month to a more readable format
57         version_month = pd.to_datetime(version_month, format='%m%Y').strftime('%Y-%m-%d')
58         print(f"Version month: {version_month}")
59
60         # Add 'Code Set' and 'Version Month' columns to the DataFrame
61         rrf_df['Code Set'] = 'RxNorm'
62         rrf_df['Version Month'] = version_month
63     except Exception as e:
64         print(f"Error occurred while extracting version month: {e}")
65
66     return rrf_df
67
68
69 def apply_header_to_rrf(file_name, rrf_df):
70     # Check if the corresponding Excel sheet exists
71     if file_name in excel_headers:
72         # Get the headers from the Excel sheet
73         excel_headers_list = excel_headers[file_name]
74         excel_headers_list = [header for header in excel_headers_list if header != 'SVER']
75         # Take names from the excel header list up to the length of the split DataFrame
76         excel_headers_list = excel_headers_list[:len(rrf_df.columns)]
77         # Set the correct header for the DataFrame
78         rrf_df.columns = excel_headers_list
79
80     return rrf_df

```

45:5 Python Spaces: 4 ⚡

The following function is employed to transform dates into the specified format. In the case of RXNSAB, the SVER column obtains its value by extracting the year component from the VSTART column, as illustrated below:

```

80 def convert_date_format(value):
81
82     try:
83         # Try to parse the value into datetime format
84         parsed_date = pd.to_datetime(value, format='%Y-%m-%d').date()
85         # Extract only the date part
86         return parsed_date.strftime('%Y-%m-%d')
87     except ValueError:
88         if value == '2020':
89             return '2020-01-01'
90         elif value == '5.0_2024_01_04':
91             # Remove the float value and parse the remaining string
92             return convert_date_format('2024_01_04')
93         elif value == '2020AA':
94             return '2024-01-02'
95         elif value == '20AA_240205F':
96             return '2024-02-05'
97         else:
98             return value
99
100
101

```

The following function handles different date formats of RXNATOMARCHIVE as below:

```

103 def update_nato_date(value):
104     try:
105         # Attempt to parse the value using the first date format
106         parsed_date = pd.to_datetime(value, format='%m/%d/%Y %I:%M:%S %p').date()
107     except ValueError:
108         try:
109             # If the first format fails, attempt to parse using the second date format
110             parsed_date = pd.to_datetime(value, format='%d-%b-%y').date()
111         except ValueError:
112             # If both formats fail, return None or handle the error appropriately
113             return None # Or handle the error appropriately
114         # Check if the parsed_date is NaT
115     if pd.isnull(parsed_date):
116         return '0000-00-00' # Replace NaT with '0000-00-00'
117     else:
118         # Extract only the date part and return it in the desired format
119         return parsed_date.strftime('%Y-%m-%d')
120
121

```

The following function processes date columns

```

121 def process_date_columns(file_name,rrf_df):
122     date_columns = ['VSTART','VEND','CREATED_TIMESTAMP', 'UPDATED_TIMESTAMP', 'LAST_RELEASED']
123     for column in date_columns:
124         if column in rrf_df.columns:
125             if file_name == 'RXNSAB':
126                 # Apply the conversion function to each value in the column
127
128                 rrf_df[column] = rrf_df[column].apply(convert_date_format)
129             # Extract year from the VSTART column after date conversion and save it directly as a string
130             rrf_df['SVER'] = pd.to_datetime(rrf_df['VSTART'], format='%Y-%m-%d').dt.year.astype(str)
131             # Reorder the columns to place 'SVER' before 'VSTART'
132             # Reorder columns
133             # Reorder columns
134             sver_index = rrf_df.columns.get_loc('SVER')
135             vstart_index = rrf_df.columns.get_loc('VSTART')
136             sf_index = rrf_df.columns.get_loc('SF')
137
138             # Remove 'SVER' from its original position
139             column_sver = rrf_df.pop('SVER')
140
141             # Insert 'SVER' after 'SF', before 'VSTART'
142             if sver_index < vstart_index:
143                 rrf_df.insert(vstart_index - 1, 'SVER', column_sver)
144             elif sver_index > vstart_index:
145                 rrf_df.insert(vstart_index, 'SVER', column_sver)
146             if file_name == 'RXNATOMARCHIVE':
147                 rrf_df[column] = rrf_df[column].apply(update_nato_date)
148
149     return rrf_df
150

```

The following code saves the converted file to .txt

```

152
153 def save_as_txt_file(rrf_df, file_name, bucket_name):
154     # Construct the filename for the output text file
155     transformation_folder = 'transformation/'
156
157     # Convert DataFrame to CSV format in memory
158     csv_buffer = io.StringIO()
159     rrf_df.to_csv(csv_buffer, sep='|', index=False)
160
161     # Upload the CSV buffer to S3
162     s3_key = transformation_folder + file_name + '.txt'
163     s3.put_object(Bucket=bucket_name, Key=s3_key, Body=csv_buffer.getvalue())
164
165     print(f"Transformed data saved to: s3://{bucket_name}/{s3_key}") #
166
167
168

```

The primary function where the preceding functions are invoked: Here, zip archives are retrieved from the correct directory, and the final pipe delimiter is disregarded by segmentation.

```

169
170 def read_and_relocate_rrf_files(s3, bucket, key):
171     try:
172         zip_response = s3.get_object(Bucket=bucket, Key=key)
173         zip_data = zip_response['Body'].read()
174         zip_filename = os.path.basename(key)
175
176         # Wrap the zip data in a BytesIO object
177         zip_file = BytesIO(zip_data)
178
179         file_path = 'test_full_02052024/rrf'
180
181         with zipfile.ZipFile(zip_file, 'r') as zip_ref:
182             for file_info in zip_ref.infolist():
183                 if file_info.filename.startswith(file_path) and not file_info.filename.endswith('/'):
184                     filename = os.path.basename(file_info.filename)
185                     print(f"The {filename} is read from zip file.")
186
187                     with zip_ref.open(file_info) as source_file:
188                         file_content = source_file.read().decode('utf-8')
189                         if file_content.endswith('|'):
190                             file_content = file_content[:-1]
191                         file_content_io = io.StringIO(file_content)
192                         rrf_df = pd.read_csv(file_content_io, delimiter='|', header=None)
193                         rrf_df = rrf_df.iloc[:, :-1]
194                         print(f"Row count before transformation: {rrf_df.shape[0]}")
195
196                         file_name = os.path.splitext(filename)[0]
197                         apply_header_to_rrf(file_name, rrf_df)
198                         rrf_df = process_date_columns(file_name, rrf_df)
199                         code_set_and_version_month(zip_filename, rrf_df)
200
201                         print(f"Row count of {file_name} after transformation: {rrf_df.shape[0]}")
202
203                         pd.set_option('display.max_columns', None)
204                         print(rrf_df.head(5))
205
206                         # Save the transformed DataFrame to a text file
207                         save_as_txt_file(rrf_df, file_name, bucket)
208
209                         # Upload the unzipped file to the 'unzipped' folder
210                         unzipped_key = 'unzipped/' + filename
211                         s3.put_object(Bucket=bucket, Key=unzipped_key, Body=file_content)
212                         print(f"Unzipped file saved to: s3://{bucket}/{unzipped_key}")
213
214     except Exception as e:
215         print(f"Error occurred: {e}")
216
217

```

The following is the lambda\_handler

```

217 def lambda_handler(event, context):
218     bucket = event['Records'][0]['s3']['bucket']['name']
219     key = event['Records'][0]['s3']['object']['key']
220
221
222     # This is the function that relocate the rrf files from zip file
223     read_excel_from_s3(bucket)
224     read_and_relocate_rrf_files(s3,bucket,key)
225

```

Uploading zip file

The screenshot shows the AWS S3 'Uploading' progress dialog. It displays a progress bar at 1% completion, with a note that the transfer rate is 195.1 KB/s. Below the progress bar, there's a summary table:

Destination	Succeeded	Failed
s3://rxnormfull/zipfile/	0 files, 2.0 MB (0.85%)	0 files, 0 B (0%)

At the bottom, there are tabs for 'Files and folders' (selected) and 'Configuration'. The 'Files and folders' section shows a table with 1 total item (238.5 MB), including columns for Name, Folder, Type, Size, Status, and Error.

## Initial rxnorm

The screenshot shows the AWS S3 bucket 'rxnormfull' details page. The 'Objects' tab is selected, displaying 2 objects. The table shows:

Name	Type	Last modified	Size	Storage class
excelfiles/	Folder	-	-	-
zipfiles/	Folder	-	-	-

## After lambda is triggered:

The screenshot shows the AWS S3 bucket 'rxnormfull' details page after a lambda function has triggered. The 'Objects' tab is selected, displaying 4 objects. The table shows:

Name	Type	Last modified	Size	Storage class
excelfiles/	Folder	-	-	-
transformation/	Folder	-	-	-
unzipped/	Folder	-	-	-
zipfiles/	Folder	-	-	-

excelfiles/

Amazon S3 > Buckets > rxnormfull > excelfiles/

### excelfiles/

**Objects** Properties

**Objects (1) 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](#)

Find objects by prefix

Name	Type	Last modified	Size	Storage class
RxNorm_Header.xlsx	xlsx	March 10, 2024, 22:53:55 (UTC+05:45)	27.9 KB	Standard

**Actions** [Copy S3 URI](#) [Upload](#)

### zippedfiles/

Amazon S3 > Buckets > rxnormfull > zipfiles/

### zipfiles/

**Objects** Properties

**Objects (1) 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](#)

Find objects by prefix

Name	Type	Last modified	Size	Storage class
RxNorm_full_02062024.zip	zip	March 11, 2024, 11:25:48 (UTC+05:45)	14.1 MB	Standard

**Actions** [Copy S3 URI](#) [Upload](#)

### Unzippedfiles/

Amazon S3 > Buckets > rxnormfull2 > unzipped/

### unzipped/

**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](#)

Find objects by prefix

Name	Type	Last modified	Size	Storage class
RXNATOMARCHIVE.RRF	RRF	March 11, 2024, 15:15:28 (UTC+05:45)	71.4 MB	Standard
RXNCNSO.RRF	RRF	March 11, 2024, 15:15:44 (UTC+05:45)	118.6 MB	Standard
RXNCUI.RRF	RRF	March 11, 2024, 15:15:45 (UTC+05:45)	1.7 MB	Standard
RXNCUICHANGES.RRF	RRF	March 11, 2024, 15:15:45 (UTC+05:45)	14.9 KB	Standard
RXNDOC.RRF	RRF	March 11, 2024, 15:15:45 (UTC+05:45)	214.2 KB	Standard
RXNREL.RRF	RRF	March 11, 2024, 15:15:09 (UTC+05:45)	484.4 MB	Standard
RXNSAB.RRF	RRF	March 11, 2024, 15:15:15 (UTC+05:45)	9.8 KB	Standard
RXNSAT.RRF	RRF	March 11, 2024, 15:16:06 (UTC+05:45)	498.7 MB	Standard
RXNSTY.RRF	RRF	March 11, 2024, 15:16:13 (UTC+05:45)	18.4 MB	Standard

**Actions** [Copy S3 URI](#) [Upload](#)

### Transformations/

Amazon S3 > Buckets > rxnormfull2 > transformation/

transformation/

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](#)

Name	Type	Last modified	Size	Storage class
RXNATOMARCHIVE.txt	txt	March 11, 2024, 15:15:27 (UTC+05:45)	69.5 MB	Standard
RXNCONSO.txt	txt	March 11, 2024, 15:15:43 (UTC+05:45)	138.1 MB	Standard
RXNCUI.txt	txt	March 11, 2024, 15:15:45 (UTC+05:45)	2.1 MB	Standard
RXNCUICHANGES.txt	txt	March 11, 2024, 15:15:45 (UTC+05:45)	17.8 KB	Standard
RXNDOC.txt	txt	March 11, 2024, 15:15:45 (UTC+05:45)	271.6 KB	Standard
RXNREL.txt	txt	March 11, 2024, 15:15:45 (UTC+05:45)	645.2 MB	Standard
RXNSAB.txt	txt	March 11, 2024, 15:17:08 (UTC+05:45)	10.3 KB	Standard
RXNSAT.txt	txt	March 11, 2024, 15:15:58 (UTC+05:45)	621.9 MB	Standard
RXNSTY.txt	txt	March 11, 2024, 15:16:13 (UTC+05:45)	26.0 MB	Standard

Here is the RXNATOMARCHIVE.rff files after transformation of dates in the txt format

**SQL query**

Add SQL from templates | Run SQL query

Amazon S3 Select supports only the SELECT SQL command. Using the S3 console, you can extract up to 40 MB of records from an object that is up to 128 MB in size. To work with larger files or more records, use the AWS CLI, AWS SDK, or Amazon S3 REST API. For more complex SQL queries, use [Amazon Athena](#)

```
1 /* To create reference point for writing SQL queries, you can display the first 5 records of input data by running the following SQL query:
   SELECT * FROM s3object s LIMIT 5 */
2 SELECT * FROM s3object s LIMIT 100
```

SQL Ln 2, Col 35 Errors: 0 Warnings: 0

**Query results**

Download results

Query results are not available after you choose Close or navigate away. Choose Download results to download a copy of the following query results.

Status: Successfully returned 100 records in 702 ms

Bytes returned: 13929 B

Raw | Formatted

```
5 2192 |A16791816|Etoposide|Aceto[2020-04-27]2006-03-10|2020-04-27|99 |ENG| 2020-04-06 |RXNORM_19AB_200406F|99|RXNORM|P2H|98|RxNorm|2024-02-05
6 2265 |A10345231|5-hydroxytriptophan[2020-04-27]2005-03-11-06|94 |ENG| 2020-04-06 |RXNORM_19AB_200406F|94|RXNORM|IN|94|RxNorm|2024-02-05
7 2311 |A16793037|Ticlopidine Hydrochloride[2020-04-27]2005-03-10|2020-04-27|97 |ENG| 2020-04-06 |RXNORM_19AB_200406F|97|RXNORM|P2N|97|RxNorm|2024-02-05
8 2322 |A10345333|6-Aminocaproic Acid[2020-04-27]2005-03-10|2020-04-27|99 |ENG| 2020-04-06 |RXNORM_19AB_200406F|99|RXNORM|IN|99|RxNorm|2024-02-05
9 2453 |A10345343|Benzylpenicillinase[2020-04-27]2005-03-10|2020-04-27|103 |ENG| 2019-10-04 |RXNORM_19AB_200406F|103|RXNORM|IN|103|RxNorm|2024-02-05
10 2454 |A10345344|Benzylpenicilloyl[2020-04-27]2005-03-10|2020-04-27|104 |ENG| 2019-10-04 |RXNORM_19AB_200406F|104|RXNORM|IN|104|RxNorm|2024-02-05
11 4330 |A10345359|Arctobetol[2020-04-27]2005-03-10|2020-04-27|149 |ENG| 2020-04-06 |RXNORM_19AB_200406F|149|RXNORM|IN|149|RxNorm|2024-02-05
12 4414 |A10345440|Adenosuccinic[2020-04-27]2005-03-10|2020-04-27|154 |ENG| 2020-04-06 |RXNORM_19AB_200406F|154|RXNORM|IN|154|RxNorm|2024-02-05
13 4458 |A10345452|Acepromazine[2020-04-27]2005-03-10|2020-04-27|155 |ENG| 2020-04-06 |RXNORM_19AB_200406F|155|RXNORM|IN|155|RxNorm|2024-02-05
14 4565 |A10345453|Acetaminilide[2020-04-27]2005-03-10|2020-04-27|156 |ENG| 2020-04-06 |RXNORM_06AC_061812|162|RXNORM|IN|162|RxNorm|2024-02-05
15 4807 |A10345454|Acetyl Acetone[2020-04-27]2005-03-10|2020-04-27|157 |ENG| 2020-04-06 |RXNORM_19AB_200406F|157|RXNORM|IN|157|RxNorm|2024-02-05
16 4714 |A10345457|Acetic Acid[2020-04-27]2005-03-10|2020-04-27|168 |ENG| 2020-04-06 |RXNORM_19AB_200406F|168|RXNORM|IN|168|RxNorm|2024-02-05
17 4855 |A10345466|Acetohexamide[2020-04-27]2005-03-10|2020-04-27|173 |ENG| 2020-04-06 |RXNORM_19AB_200406F|173|RXNORM|IN|173|RxNorm|2024-02-05
18 4998 |A10345467|Acetone[2020-04-27]2005-03-10|2020-04-27|178 |ENG| 2020-04-06 |RXNORM_19AB_200406F|178|RXNORM|IN|178|RxNorm|2024-02-05
19 5411 |A10345468|Acetylacetone[2020-04-27]2005-03-10|2020-04-27|179 |ENG| 2020-04-06 |RXNORM_19AB_200406F|179|RXNORM|IN|179|RxNorm|2024-02-05
20 5475 |A10345469|Acetylglucosamine[2020-04-27]2005-03-10|2020-04-27|180 |ENG| 2020-04-06 |RXNORM_19AB_200406F|180|RXNORM|IN|180|RxNorm|2024-02-05
21 5475 |A10345459|Acetylglutathione[2020-04-27]2005-03-10|2020-04-27|187 |ENG| 2020-04-06 |RXNORM_19AB_200406F|187|RXNORM|IN|187|RxNorm|2024-02-05
22 5588 |A10345452|AcetylIdoxigoxine[2020-04-27]2005-03-10|2020-04-27|199 |ENG| 2020-04-06 |RXNORM_19AB_200406F|199|RXNORM|IN|199|RxNorm|2024-02-05
23 5608 |A10345453|Acetylene[2020-04-27]2005-03-10|2020-04-27|209 |ENG| 2020-04-06 |RXNORM_19AB_200406F|209|RXNORM|IN|209|RxNorm|2024-02-05
24 6433 |A10345455|Acidulated Phenylalanide[2020-04-27]2005-03-10|2020-04-27|236 |ENG| 2020-04-06 |RXNORM_19AB_200406F|236|RXNORM|IN|236|RxNorm|2024-02-05
25 6458 |A10345456|Acidulase[2020-04-27]2005-03-10|2020-04-27|237 |ENG| 2020-04-06 |RXNORM_19AB_200406F|237|RXNORM|IN|237|RxNorm|2024-02-05
26 6658 |A10345458|Acotile[2020-04-27]2005-03-10|2020-04-27|242 |ENG| 2020-04-06 |RXNORM_19AB_200406F|242|RXNORM|IN|242|RxNorm|2024-02-05
27 7012 |A10345459|Acriflavine[2020-04-27]2005-03-10|2020-04-27|249 |ENG| 2020-04-06 |RXNORM_19AB_200406F|249|RXNORM|IN|249|RxNorm|2024-02-05
28 7746 |A10345460|Spectinomycin[2020-04-27]2005-03-10|2020-04-27|278 |ENG| 2020-04-06 |RXNORM_19AB_200406F|278|RXNORM|IN|278|RxNorm|2024-02-05
29 7824 |A10345461|Acetylcholinesterase Cholinesterase[2020-04-27]2005-03-10|2020-04-27|272 |ENG| 2020-04-06 |RXNORM_19AB_200406F|272|RXNORM|IN|272|RxNorm|2024-02-05
30 8019 |A10345464|AcetylLorazepam[2020-04-27]2005-03-10|2020-04-27|281 |ENG| 2020-04-06 |RXNORM_19AB_200406F|281|RXNORM|IN|281|RxNorm|2024-02-05
31 8416 |A10345466|Adenine[2020-04-27]2005-03-10|2020-04-27|290 |ENG| 2020-04-06 |RXNORM_19AB_200406F|290|RXNORM|IN|290|RxNorm|2024-02-05
32 8498 |A10345467|Adenine Nucleotide[2006-10-11]2005-03-10|2006-10-11|292 |ENG| 0000-00-00 |RXNORM_06AC_069901|292|RXNORM|IN|292|RxNorm|2024-02-05
33 8755 |A1034568|Adenosine[2020-04-27]2005-03-10|2020-04-27|296 |ENG| 2020-04-06 |RXNORM_19AB_200406F|296|RXNORM|IN|296|RxNorm|2024-02-05
```

Here is the RXNSAB.rff files transformed after adding SVER column and changing the date format.

## Query results

Query results are not available after you choose **Close** or navigate away. Choose **Download results** to download a copy of the following query results.

 Download results

## Status

SuccessFully returned 14 records in 2173 ms

Bytes returned: 10506 B

[Raw](#) | [Formatted](#)

Finally, we validate the CloudWatch logs:

## 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](#)

Filter events		Clear
▶	Timestamp	Message
There are older events to load. <a href="#">Load more</a> .		
▶	2024-03-11T15:15:23.511+05:45	1 2024-02-05
▶	2024-03-11T15:15:23.511+05:45	2 2024-02-05
▶	2024-03-11T15:15:23.511+05:45	3 2024-02-05
▶	2024-03-11T15:15:23.511+05:45	4 2024-02-05
▶	2024-03-11T15:15:27.321+05:45	Transformed data saved to: s3://rxnormfull2/transformation/RXNATOMARCHIVE.txt
▶	2024-03-11T15:15:28.005+05:45	Unzipped file saved to: s3://rxnormfull2/unzipped/RXNATOMARCHIVE.RRF
▶	2024-03-11T15:15:28.005+05:45	The RXNCONSO.RRF is read from zip file.
▶	2024-03-11T15:15:33.125+05:45	Row count before transformation: 1133065
▶	2024-03-11T15:15:33.125+05:45	Version month: 2024-02-05
▶	2024-03-11T15:15:33.137+05:45	Row count of RXNCONSO after transformation: 1133065
▶	2024-03-11T15:15:33.141+05:45	RXCUI LAT TS LUI STT SUI ISPREF RXAUI SAUI SCUI SDUI \
▶	2024-03-11T15:15:33.141+05:45	0 3 ENG NaN NaN NaN NaN 8717795 NaN 58488005 NaN
▶	2024-03-11T15:15:33.141+05:45	1 3 ENG NaN NaN NaN NaN 8717796 NaN 58488005 NaN
▶	2024-03-11T15:15:33.141+05:45	2 3 ENG NaN NaN NaN NaN 8717808 NaN 58488005 NaN
▶	2024-03-11T15:15:33.141+05:45	3 3 ENG NaN NaN NaN NaN 8718164 NaN 58488005 NaN
▶	2024-03-11T15:15:33.141+05:45	4 19 ENG NaN NaN NaN NaN 10794494 NaN 112116001 NaN

▶	2024-03-11T15:15:33.141+05:45	SAB TTY CODE STR \\
▶	2024-03-11T15:15:33.141+05:45	0 SNOMEDCT_US PT 58488005 1,4-alpha-Glucan branching enzyme
▶	2024-03-11T15:15:33.141+05:45	1 SNOMEDCT_US FN 58488005 1,4-alpha-Glucan branching enzyme (substance)
▶	2024-03-11T15:15:33.141+05:45	2 SNOMEDCT_US SY 58488005 Amylo-(1,4,6)-transglycosylase
▶	2024-03-11T15:15:33.141+05:45	3 SNOMEDCT_US SY 58488005 Branching enzyme
▶	2024-03-11T15:15:33.141+05:45	4 SNOMEDCT_US SY 112116001 17-hydrocorticosteroid
▶	2024-03-11T15:15:33.141+05:45	SRL SUPPRESS CVF Code Set Version Month
▶	2024-03-11T15:15:33.141+05:45	0 NaN N NaN RxNorm 2024-02-05
▶	2024-03-11T15:15:33.141+05:45	1 NaN N NaN RxNorm 2024-02-05
▶	2024-03-11T15:15:33.141+05:45	2 NaN N NaN RxNorm 2024-02-05
▶	2024-03-11T15:15:33.141+05:45	3 NaN N NaN RxNorm 2024-02-05
▶	2024-03-11T15:15:33.141+05:45	4 NaN N NaN RxNorm 2024-02-05
▶	2024-03-11T15:15:42.936+05:45	Transformed data saved to: s3://rxnormfull2/transformation/RXNCONSO.txt
▶	2024-03-11T15:15:43.948+05:45	Unzipped file saved to: s3://rxnormfull2/unzipped/RXNCONSO.RRF
▶	2024-03-11T15:15:43.948+05:45	The RXNCUICHANGES.RRF is read from zip file.
▶	2024-03-11T15:15:44.041+05:45	Row count before transformation: 153
▶	2024-03-11T15:15:44.042+05:45	Version month: 2024-02-05
▶	2024-03-11T15:15:44.042+05:45	Row count of RXNCUICHANGES after transformation: 153
▶	2024-03-11T15:15:44.044+05:45	RXAUI CODE SAB TTY STR OLD_RXCUI NEW_RXCUI Code Set Version Month
▶	2024-03-11T15:15:44.044+05:45	0 NaN NaN RXNORM NaN NaN 1653118 2672916 RxNorm 2024-02-05
▶	2024-03-11T15:15:44.044+05:45	1 NaN NaN RXNORM NaN NaN 1653119 2672917 RxNorm 2024-02-05
▶	2024-03-11T15:15:44.044+05:45	2 NaN NaN RXNORM NaN NaN 1806685 2672352 RxNorm 2024-02-05

▶ 2024-03-11T15:15:44.044+05:45 3 NaN NaN RXNORM NaN NaN 1806938 2673304 RxNorm 2024-02-05

▶ 2024-03-11T15:15:44.044+05:45 4 NaN NaN RXNORM NaN NaN 2647628 2672353 RxNorm 2024-02-05

▶ 2024-03-11T15:15:44.081+05:45 Transformed data saved to: s3://rxnormfull2/transformation/RXNCUICHANGES.txt

▶ 2024-03-11T15:15:44.133+05:45 Unzipped file saved to: s3://rxnormfull2/unzipped/RXNCUICHANGES.RRF

▶ 2024-03-11T15:15:44.133+05:45 The RXNCUI.RRF is read from zip file.

▶ 2024-03-11T15:15:44.159+05:45 Row count before transformation: 30046

▶ 2024-03-11T15:15:44.160+05:45 Version month: 2024-02-05

▶ 2024-03-11T15:15:44.160+05:45 Row count of RXNCUI after transformation: 30046

▶ 2024-03-11T15:15:44.162+05:45 CUI1 VER\_START VER\_END Cardinality CUI2 \

▶ 2024-03-11T15:15:44.162+05:45 0 89904 RXNORM\_17AA\_170905F RXNORM\_20AA\_221003F 1 89904

▶ 2024-03-11T15:15:44.162+05:45 1 91669 RXNORM\_17AA\_170905F RXNORM\_18AB\_181203F 1 91669

▶ 2024-03-11T15:15:44.162+05:45 2 91714 RXNORM\_10AA\_100802F RXNORM\_10AA\_100802F 1 1000992

▶ 2024-03-11T15:15:44.162+05:45 3 91811 RXNORM\_04AC\_050210F RXNORM\_09AA\_090706F 1 857082

▶ 2024-03-11T15:15:44.162+05:45 4 91812 RXNORM\_04AC\_050210F RXNORM\_09AA\_090706F 1 857087

▶ 2024-03-11T15:15:44.162+05:45 Code Set Version Month

▶ 2024-03-11T15:15:44.162+05:45 0 RxNorm 2024-02-05

▶ 2024-03-11T15:15:44.162+05:45 1 RxNorm 2024-02-05

▶ 2024-03-11T15:15:44.162+05:45 2 RxNorm 2024-02-05

▶ 2024-03-11T15:15:44.162+05:45 3 RxNorm 2024-02-05

▶ 2024-03-11T15:15:44.162+05:45 4 RxNorm 2024-02-05

▶ 2024-03-11T15:15:44.363+05:45 Transformed data saved to: s3://rxnormfull2/transformation/RXNCUI.txt

▶ 2024-03-11T15:15:44.479+05:45 Unzipped file saved to: s3://rxnormfull2/unzipped/RXNCUI.RRF

- ▶ 2024-03-11T15:15:44.479+05:45 The RXNDOC.RRF is read from zip file.
- ▶ 2024-03-11T15:15:44.486+05:45 Row count before transformation: 3445
- ▶ 2024-03-11T15:15:44.487+05:45 Version month: 2024-02-05
- ▶ 2024-03-11T15:15:44.487+05:45 Row count of RXNDOC after transformation: 3445
- ▶ 2024-03-11T15:15:44.489+05:45 KEY VALUE TYPE \  
  0 ATN AAL\_TERM expanded\_form
- ▶ 2024-03-11T15:15:44.489+05:45 1 ATN ACCEPTABILITYID expanded\_form
- ▶ 2024-03-11T15:15:44.489+05:45 2 ATN ACCEPTED\_THERAPEUTIC\_USE\_FOR expanded\_form
- ▶ 2024-03-11T15:15:44.489+05:45 3 ATN ACTIVE expanded\_form
- ▶ 2024-03-11T15:15:44.489+05:45 4 ATN ADDED\_MEANING expanded\_form
- ▶ 2024-03-11T15:15:44.489+05:45 EXPL Code Set Version Month
- ▶ 2024-03-11T15:15:44.489+05:45 0 AAL term RxNorm 2024-02-05
- ▶ 2024-03-11T15:15:44.489+05:45 1 Acceptability Id RxNorm 2024-02-05
- ▶ 2024-03-11T15:15:44.489+05:45 2 Accepted therapeutic use for RxNorm 2024-02-05
- ▶ 2024-03-11T15:15:44.489+05:45 3 Active RxNorm 2024-02-05
- ▶ 2024-03-11T15:15:44.489+05:45 4 Additional descriptive information RxNorm 2024-02-05
- ▶ 2024-03-11T15:15:44.587+05:45 Transformed data saved to: s3://rxnormfull12/transformation/RXNDOC.txt
- ▶ 2024-03-11T15:15:44.652+05:45 Unzipped file saved to: s3://rxnormfull12/unzipped/RXNDOC.RRF
- ▶ 2024-03-11T15:15:44.652+05:45 The RXNREL.RRF is read from zip file.
- ▶ 2024-03-11T15:16:01.421+05:45 Row count before transformation: 7154306
- ▶ 2024-03-11T15:16:01.422+05:45 Version month: 2024-02-05
- ▶ 2024-03-11T15:16:01.512+05:45 Row count of RXNREL after transformation: 7154306

▶ 2024-03-11T15:16:01.517+05:45 RXCUI1 RXAUI1 STYPE1 REL RXCUI2 RXAUI2 STYPE2 \\

▶ 2024-03-11T15:16:01.517+05:45 0 NaN 828.0 AUI RO NaN 788773.0 AUI

▶ 2024-03-11T15:16:01.517+05:45 1 NaN 828.0 AUI RO NaN 790716.0 AUI

▶ 2024-03-11T15:16:01.517+05:45 2 NaN 829.0 AUI SY NaN 828.0 AUI

▶ 2024-03-11T15:16:01.517+05:45 3 NaN 945.0 AUI RO NaN 2369805.0 AUI

▶ 2024-03-11T15:16:01.517+05:45 4 NaN 950.0 AUI RO NaN 9251787.0 AUI

▶ 2024-03-11T15:16:01.517+05:45 RELA RUI SRUI SAB SL RG \\

▶ 2024-03-11T15:16:01.517+05:45 0 has\_ingredient 5066189.0 NaN MMSL NaN NaN

▶ 2024-03-11T15:16:01.517+05:45 1 has\_ingredient 3823969.0 NaN MMSL NaN NaN

▶ 2024-03-11T15:16:01.517+05:45 2 NaN NaN NaN RXNORM NaN NaN

▶ 2024-03-11T15:16:01.517+05:45 3 has\_ingredient 4306227.0 NaN MMSL NaN NaN

▶ 2024-03-11T15:16:01.517+05:45 4 has\_basis\_of\_strength\_substance 116141227.0 NaN SNOMEDCT\_US NaN NaN

▶ 2024-03-11T15:16:01.517+05:45 DIR SUPPRESS CVF Code Set Version Month

▶ 2024-03-11T15:16:01.517+05:45 0 NaN NaN NaN RxNorm 2024-02-05

▶ 2024-03-11T15:16:01.517+05:45 1 NaN NaN NaN RxNorm 2024-02-05

▶ 2024-03-11T15:16:01.517+05:45 2 NaN NaN NaN RxNorm 2024-02-05

▶ 2024-03-11T15:16:01.517+05:45 3 NaN NaN NaN RxNorm 2024-02-05

▶ 2024-03-11T15:16:01.517+05:45 4 NaN NaN NaN RxNorm 2024-02-05

▶ 2024-03-11T15:17:01.034+05:45 Transformed data saved to: s3://rxnormfull2/transformation/RXNREL.txt

▶ 2024-03-11T15:17:07.625+05:45 Unzipped file saved to: s3://rxnormfull2/unzipped/RXNREL.RRF

▶ 2024-03-11T15:17:07.625+05:45 The RXNSAB.RRF is read from zip file.

▶	2024-03-11T15:17:07.854+05:45	Row count before transformation: 13
▶	2024-03-11T15:17:07.861+05:45	Version month: 2024-02-05
▶	2024-03-11T15:17:07.861+05:45	Row count of RXNSAB after transformation: 13
▶	2024-03-11T15:17:07.866+05:45	VCUI RCUI VSAB RSAB \
▶	2024-03-11T15:17:07.866+05:45	0 C5233827 C1140218 MMSL_2024_01_01 MMSL
▶	2024-03-11T15:17:07.866+05:45	1 C5233830 C1140182 MMX_2024_01_02 MMX
▶	2024-03-11T15:17:07.866+05:45	2 C5233828 C1140261 NDDF_2024_01_03 NDDF
▶	2024-03-11T15:17:07.866+05:45	3 C5233835 C1140284 RXNORM_20AA_240205F RXNORM
▶	2024-03-11T15:17:07.866+05:45	4 C3531723 C2720507 SNOMEDCT_US_2023_06_30 SNOMEDCT_US
▶	2024-03-11T15:17:07.866+05:45	SON SF SVER VSTART VEND \
▶	2024-03-11T15:17:07.866+05:45	0 Multum MediSource Lexicon MMSL 2024 2024-01-01 NaN
▶	2024-03-11T15:17:07.866+05:45	1 Micromedex RED BOOK MMX 2024 2024-01-02 NaN
▶	2024-03-11T15:17:07.866+05:45	2 FDB MedKnowledge (formerly NDDF Plus) NDDF 2024 2024-01-03 NaN
▶	2024-03-11T15:17:07.866+05:45	3 RxNorm Vocabulary RXNORM 2024 2024-02-05 NaN
▶	2024-03-11T15:17:07.866+05:45	4 US Edition of SNOMED CT SNOMEDCT 2023 2023-06-30 NaN
▶	2024-03-11T15:17:07.866+05:45	IMETA RMETA SLC SCC \
▶	2024-03-11T15:17:07.866+05:45	0 NaN 2020AA NaN ;;Multum Information Services;3200 Cherry Cree...
▶	2024-03-11T15:17:07.866+05:45	1 NaN 2020AA NaN ;;Micromedex;6200 South Syracuse Way, Suite 30...
▶	2024-03-11T15:17:07.866+05:45	2 NaN 2020AA NaN ;;First Databank Customer Support;701 Gateway ...
▶	2024-03-11T15:17:07.866+05:45	3 NaN 2020AA NaN RxNorm Customer Service;;U.S. National Library...
▶	2024-03-11T15:17:07.866+05:45	4 NaN 2020AA NaN National Library Of Medicine;NLM is a Charter ...
▶	2024-03-11T15:17:07.866+05:45	SRL TFR CFR CXTY TTYL \

▶	2024-03-11T15:17:07.866+05:45	0 ;;Multum Information Services;3200 Cherry Cree... 1 NaN NaN NaN
▶	2024-03-11T15:17:07.866+05:45	1 NaN 3 NaN NaN NaN
▶	2024-03-11T15:17:07.866+05:45	2 ;;First Databank Customer Support;701 Gateway ... 3 NaN NaN NaN
▶	2024-03-11T15:17:07.866+05:45	3 RxNorm Customer Service;;U.S. National Library... 0 NaN NaN NaN
▶	2024-03-11T15:17:07.866+05:45	4 National Library Of Medicine;NLM is a Charter ... 9 NaN NaN NaN
▶	2024-03-11T15:17:07.866+05:45	ATNL \
▶	2024-03-11T15:17:07.866+05:45	0 GN,MTH_RXN_BD,CD,IN,MS,SC,BD,BN
▶	2024-03-11T15:17:07.866+05:45	1 MTH_RXN_CD,BD,CD,MTH_RXN_BD
▶	2024-03-11T15:17:07.866+05:45	2 CDC,CDA,CDD,IN,MTH_RXN_CDC,DF
▶	2024-03-11T15:17:07.866+05:45	3 SCDF,DFG,SBD,PIN,TMSY,BN,SBDC,MIN,SBDG,PSN,SCD...
▶	2024-03-11T15:17:07.866+05:45	4 FN,SY,PT,PTGB,SYGB
▶	2024-03-11T15:17:07.866+05:45	LAT CENC CURVER SABIN SSN \
▶	2024-03-11T15:17:07.866+05:45	0 DCSA,DDF,DDFA,DHJC,DPC,DRT,DRTA,DST,NDC,TYPE ENG UTF-8 Y Y
▶	2024-03-11T15:17:07.866+05:45	1 DCSA,DDF,DESI_DESC,DRT,DST,LABELER,MMX_RXO,NDC... ENG UTF-8 Y Y
▶	2024-03-11T15:17:07.866+05:45	2 NDC ENG UTF-8 Y Y
▶	2024-03-11T15:17:07.866+05:45	3 AMBIGUITY_FLAG,NDC,ORIG_CODE,ORIG_SOURCE,RXN_A... ENG UTF-8 Y Y
▶	2024-03-11T15:17:07.866+05:45	4 NaN ENG UTF-8 Y Y
▶	2024-03-11T15:17:07.866+05:45	SCIT \
▶	2024-03-11T15:17:07.866+05:45	0 Multum
▶	2024-03-11T15:17:07.866+05:45	1 Micromedex
▶	2024-03-11T15:17:07.866+05:45	2 FDB MedKnowledge
▶	2024-03-11T15:17:07.866+05:45	3 RxNorm work done by the National Library of Me...

▶	2024-03-11T15:17:07.866+05:45	4 US Edition of SNOMED CT
▶	2024-03-11T15:17:07.866+05:45	Code set Code Set Version Month
▶	2024-03-11T15:17:07.866+05:45	0 ;;;Medisource Lexicon;;January 01, 2024;Denv... RxNorm 2024-02-05
▶	2024-03-11T15:17:07.866+05:45	1 ;;;Micromedex RED BOOK;;January 02, 2024;;;; RxNorm 2024-02-05
▶	2024-03-11T15:17:07.866+05:45	2 ;;;FDB MedKnowledge (formerly NDDF Plus);;Ja... RxNorm 2024-02-05
▶	2024-03-11T15:17:07.866+05:45	3 ;;;RxNorm;;META2020AA Full Update 2024_02_05... RxNorm 2024-02-05
▶	2024-03-11T15:17:07.866+05:45	4 ;;International Health Terminology Standards D... RxNorm 2024-02-05
▶	2024-03-11T15:17:07.895+05:45	Transformed data saved to: s3://rxnormfull2/transformation/RXNSAB.txt
▶	2024-03-11T15:17:07.948+05:45	Unzipped file saved to: s3://rxnormfull2/unzipped/RXNSAB.RRF
▶	2024-03-11T15:17:07.948+05:45	The RXNSAT.RRF is read from zip file.
▶	2024-03-11T15:17:24.932+05:45	Row count before transformation: 7222404
▶	2024-03-11T15:17:24.932+05:45	Version month: 2024-02-05
▶	2024-03-11T15:17:25.023+05:45	Row count of RXNSAT after transformation: 7222404
▶	2024-03-11T15:17:25.026+05:45	RXCUI LUI SUI RXAUI STYPE CODE ATUI SATUI ATN \
▶	2024-03-11T15:17:25.026+05:45	0 38 NaN NaN 829 AUI 38 NaN NaN RXN_BN_CARDINALITY
▶	2024-03-11T15:17:25.026+05:45	1 44 NaN NaN 955 AUI d01411 NaN NaN DPC
▶	2024-03-11T15:17:25.026+05:45	2 44 NaN NaN 2798745 AUI NR701405Q9 NaN NaN SPL_SET_ID
▶	2024-03-11T15:17:25.026+05:45	3 44 NaN NaN 2982613 AUI NR701405Q9 NaN NaN SPL_SET_ID
▶	2024-03-11T15:17:25.026+05:45	4 44 NaN NaN 2982613 AUI NR701405Q9 NaN NaN SPL_SET_ID
▶	2024-03-11T15:17:25.026+05:45	SAB ATV SUPPRESS CVF Code Set \
▶	2024-03-11T15:17:25.026+05:45	0 RXNORM single N 4096.0 RxNorm
▶	2024-03-11T15:17:25.026+05:45	1 MMSL N N NaN RxNorm

▶	2024-03-11T15:17:25.026+05:45	2 MTHSPL 2a2a526f-636b-7920-4261-6c626f612a2a N 4096.0 RxNorm
▶	2024-03-11T15:17:25.026+05:45	3 MTHSPL 2e8eebc3-ea75-4c57-bfe1-2c5dc2c37806 N 4096.0 RxNorm
▶	2024-03-11T15:17:25.026+05:45	4 MTHSPL 49fdfbc3-69f8-4706-9d11-bf60e4e7811b N 4096.0 RxNorm
▶	2024-03-11T15:17:25.026+05:45	Version Month
▶	2024-03-11T15:17:25.026+05:45	0 2024-02-05
▶	2024-03-11T15:17:25.026+05:45	1 2024-02-05
▶	2024-03-11T15:17:25.026+05:45	2 2024-02-05
▶	2024-03-11T15:17:25.026+05:45	3 2024-02-05
▶	2024-03-11T15:17:25.026+05:45	4 2024-02-05
▶	2024-03-11T15:18:11.032+05:45	Transformed data saved to: s3://rxnormfull2/transformation/RXNSAT.txt
▶	2024-03-11T15:18:17.738+05:45	Unzipped file saved to: s3://rxnormfull2/unzipped/RXNSAT.RRF
▶	2024-03-11T15:18:17.738+05:45	The RXNSTY.RRF is read from zip file.
▶	2024-03-11T15:18:18.462+05:45	Row count before transformation: 461874
▶	2024-03-11T15:18:18.462+05:45	Version month: 2024-02-05
▶	2024-03-11T15:18:18.467+05:45	Row count of RXNSTY after transformation: 461874
▶	2024-03-11T15:18:18.470+05:45	RXCUI TUI STN STY ATUI CVF \
▶	2024-03-11T15:18:18.470+05:45	0 3 T126 A1.4.1.1.3.3 Enzyme NaN NaN
▶	2024-03-11T15:18:18.470+05:45	1 3 T116 A1.4.1.2.1.7 Amino Acid, Peptide, or Protein NaN NaN
▶	2024-03-11T15:18:18.470+05:45	2 19 T125 A1.4.1.1.3.2 Hormone NaN NaN
▶	2024-03-11T15:18:18.470+05:45	3 19 T109 A1.4.1.2.1 Organic Chemical NaN NaN
▶	2024-03-11T15:18:18.470+05:45	4 21 T125 A1.4.1.1.3.2 Hormone NaN NaN
▶	2024-03-11T15:18:18.470+05:45	Code Set Version Month
▶	2024-03-11T15:18:18.470+05:45	0 RxNorm 2024-02-05
▶	2024-03-11T15:18:18.470+05:45	1 RxNorm 2024-02-05
▶	2024-03-11T15:18:18.470+05:45	2 RxNorm 2024-02-05
▶	2024-03-11T15:18:18.470+05:45	3 RxNorm 2024-02-05
▶	2024-03-11T15:18:18.470+05:45	4 RxNorm 2024-02-05
▶	2024-03-11T15:18:20.072+05:45	Transformed data saved to: s3://rxnormfull2/transformation/RXNSTY.txt
▶	2024-03-11T15:18:20.391+05:45	Unzipped file saved to: s3://rxnormfull2/unzipped/RXNSTY.RRF
▶	2024-03-11T15:18:20.414+05:45	END RequestId: 69ed8773-2ad3-4253-8698-c202a72f4ed9
▶	2024-03-11T15:18:20.414+05:45	REPORT RequestId: 69ed8773-2ad3-4253-8698-c202a72f4ed9 Duration: 289681.64 ms Billed Duration: 289682 ms Memory Size: 10000 MB Max Memory Used: 7672 MB Init Duration: 2877.71 ms
No newer events at this moment. Auto retry paused. <a href="#">Resume</a>		<a href="#">Back</a>