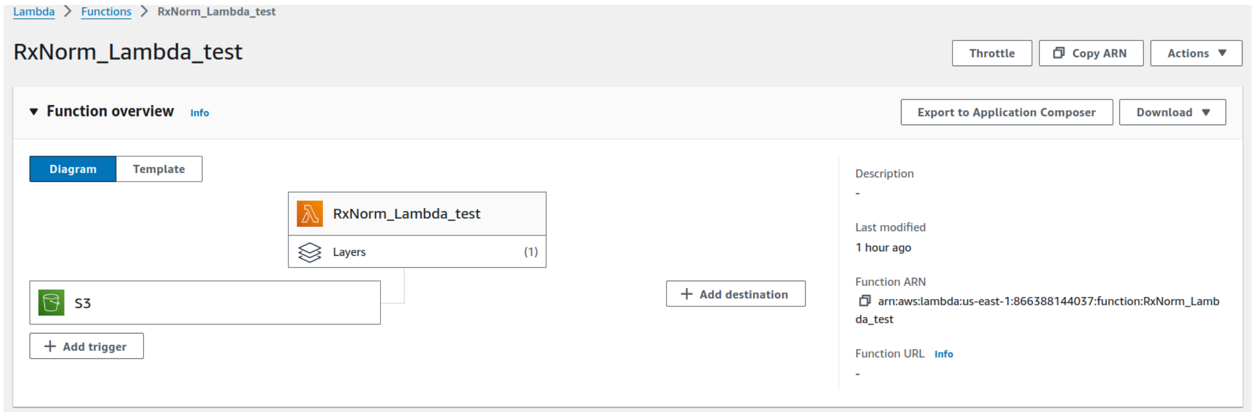
RxNorm Lambda project

General overview

1. **Create the lambda function with s3 bucket trigger**



RxNorm_Lambda_test

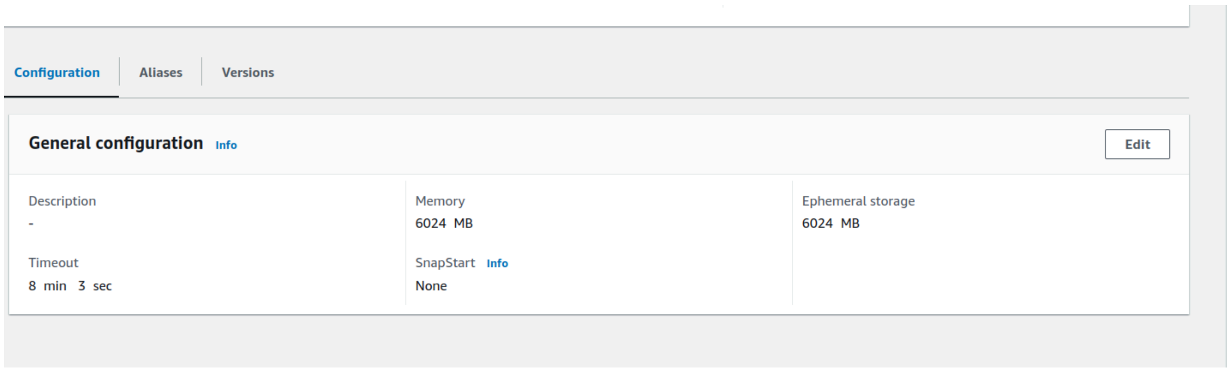
Description
-

Last modified
1 hour ago

Function ARN
arn:aws:lambda:us-east-1:866388144037:function:RxNorm_Lambda_test

Function URL Info
-

2. **Add necessary configuration like memory, timeout and layers for lambda**



General configuration Info

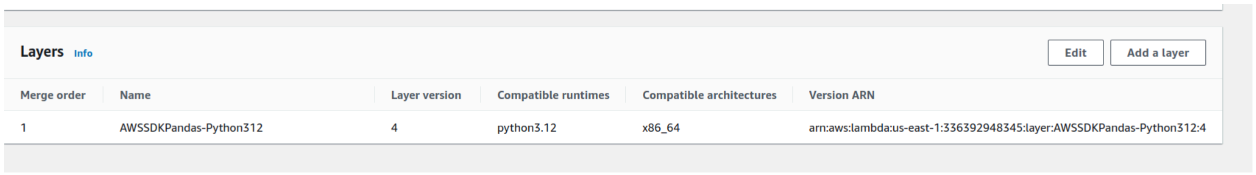
Description
-

Memory
6024 MB

Ephemeral storage
6024 MB

Timeout
8 min 3 sec

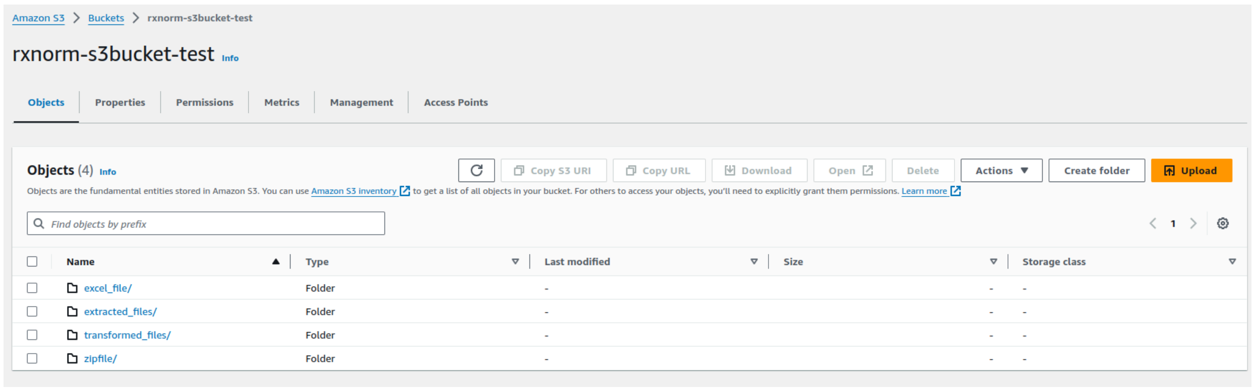
SnapStart Info
None



Layers Info

Merge order	Name	Layer version	Compatible runtimes	Compatible architectures	Version ARN
1	AWSSDKPandas-Python312	4	python3.12	x86_64	arn:aws:lambda:us-east-1:336392948345:layer:AWSSDKPandas-Python312:4

3. **This is my final working structure**



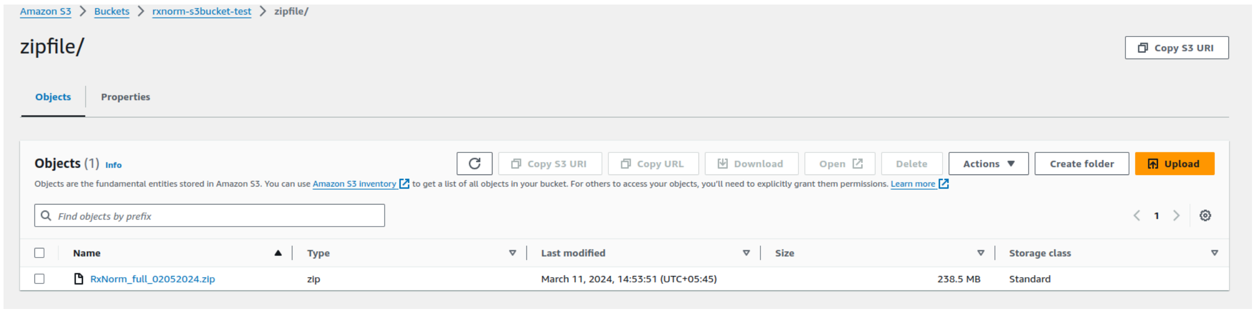
rxnorm-s3bucket-test

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

Name	Type	Last modified	Size	Storage class
excel_file/	Folder	-	-	-
extracted_files/	Folder	-	-	-
transformed_files/	Folder	-	-	-
zipfile/	Folder	-	-	-

4. **First upload the RxNorm_full_02052024.zip in the zip file which will trigger the lambda and executes the specific code**



Amazon S3 > Buckets > rxnorm-s3bucket-test > zipfile/

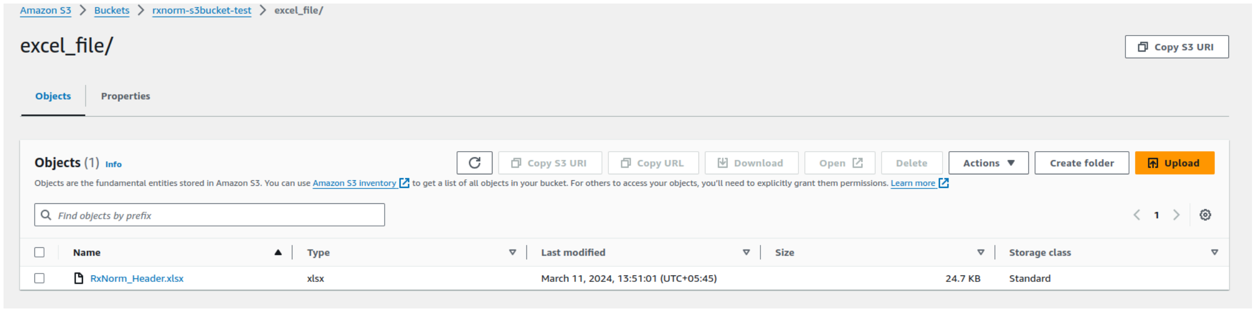
zipfile/

Objects (1)

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
RxNorm_full_02052024.zip	zip	March 11, 2024, 14:53:51 (UTC+05:45)	238.5 MB	Standard

5. **Upload the RxNorm_Header.xlsx in excel_file folder to read the headers**



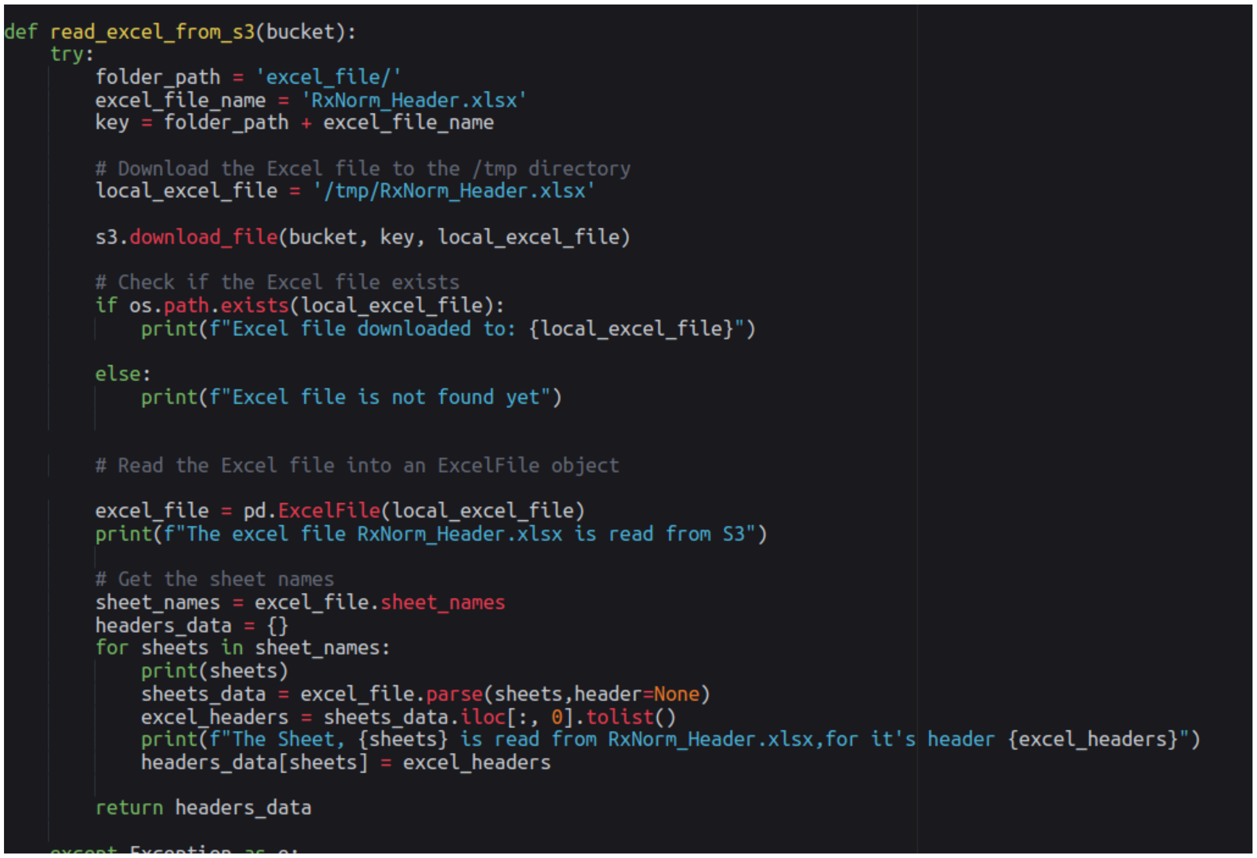
Amazon S3 > Buckets > rxnorm-s3bucket-test > excel_file/

excel_file/

Objects (1)

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

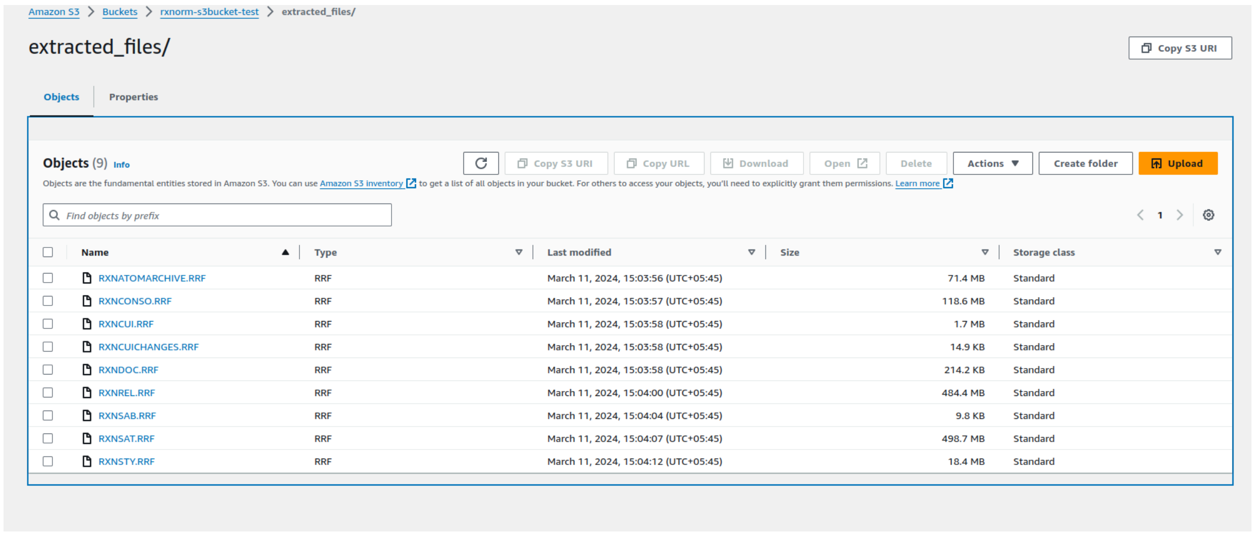
Name	Type	Last modified	Size	Storage class
RxNorm_Header.xlsx	xlsx	March 11, 2024, 13:51:01 (UTC+05:45)	24.7 KB	Standard



```python
def read_excel_from_s3(bucket):
    try:
        folder_path = 'excel_file/'
        excel_file_name = 'RxNorm_Header.xlsx'
        key = folder_path + excel_file_name

        # Download the Excel file to the /tmp directory
        local_excel_file = '/tmp/RxNorm_Header.xlsx'

        s3.download_file(bucket, key, local_excel_file)

        # Check if the Excel file exists
        if os.path.exists(local_excel_file):
            print(f"Excel file downloaded to: {local_excel_file}")

        else:
            print(f"Excel file is not found yet")


        # Read the Excel file into an ExcelFile object

        excel_file = pd.ExcelFile(local_excel_file)
        print(f"The excel file RxNorm_Header.xlsx is read from S3")

        # Get the sheet names
        sheet_names = excel_file.sheet_names
        headers_data = {}
        for sheets in sheet_names:
            print(sheets)
            sheets_data = excel_file.parse(sheets,header=None)
            excel_headers = sheets_data.iloc[:, 0].tolist()
            print(f"The Sheet, {sheets} is read from RxNorm_Header.xlsx,for it's header {excel_headers}")
            headers_data[sheets] = excel_headers

        return headers_data

    except Exception as e:
```

Code that reads the headers for each file when the lambda triggers.

6. **Now when i upload the zip file it will only extract the file in the RRf folder and dumps in the extracted_files folder**

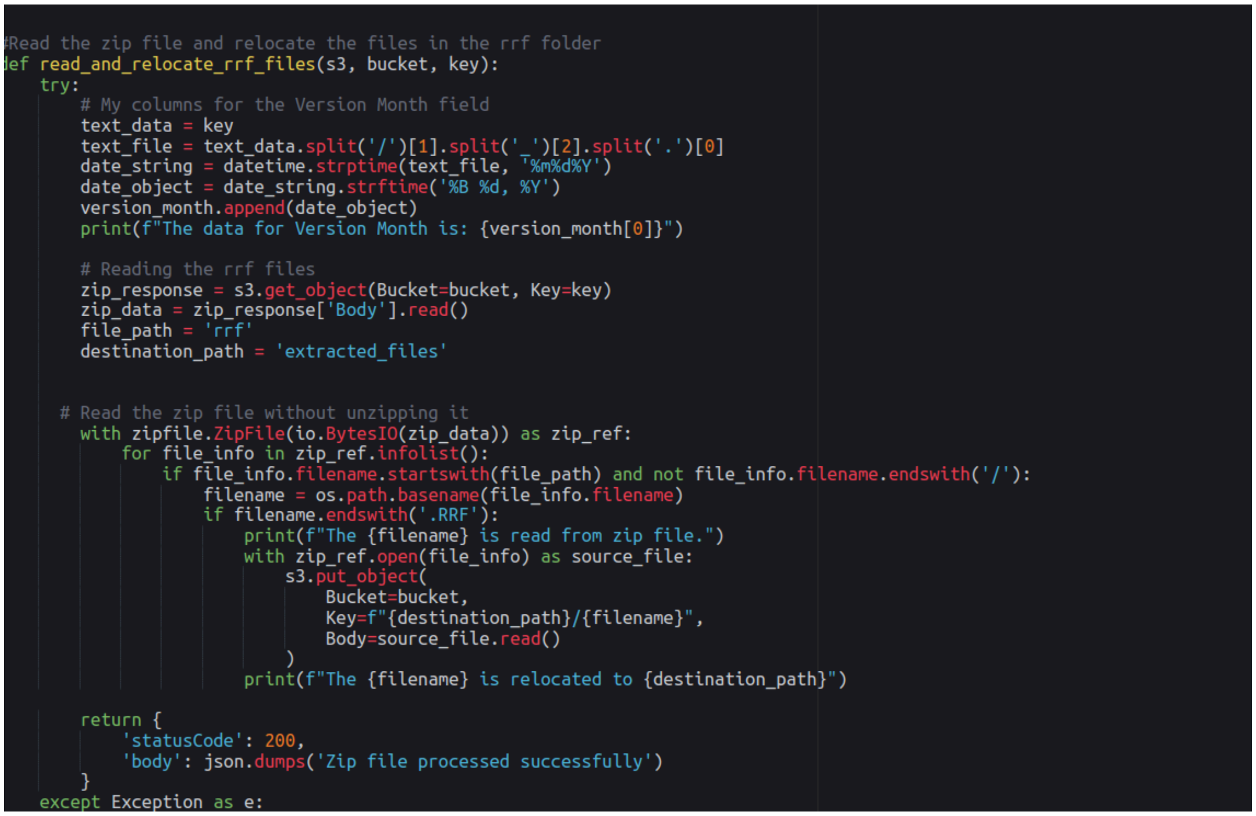

Amazon S3 > Buckets > rxnorm-s3bucket-test > extracted_files/

extracted_files/

Objects (9) Info

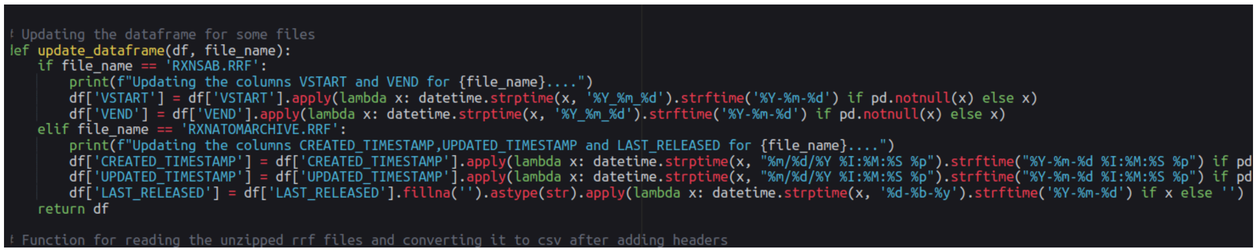
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
RXNATOMARCHIVE.RRF	RRF	March 11, 2024, 15:03:56 (UTC+05:45)	71.4 MB	Standard
RXNCONSO.RRF	RRF	March 11, 2024, 15:03:57 (UTC+05:45)	118.6 MB	Standard
RXNCUI.RRF	RRF	March 11, 2024, 15:03:58 (UTC+05:45)	1.7 MB	Standard
RXNCUICHANGES.RRF	RRF	March 11, 2024, 15:03:58 (UTC+05:45)	14.9 KB	Standard
RXNDOC.RRF	RRF	March 11, 2024, 15:03:58 (UTC+05:45)	214.2 KB	Standard
RXNREL.RRF	RRF	March 11, 2024, 15:04:00 (UTC+05:45)	484.4 MB	Standard
RXNSAB.RRF	RRF	March 11, 2024, 15:04:04 (UTC+05:45)	9.8 KB	Standard
RXNSAT.RRF	RRF	March 11, 2024, 15:04:07 (UTC+05:45)	498.7 MB	Standard
RXNSTY.RRF	RRF	March 11, 2024, 15:04:12 (UTC+05:45)	18.4 MB	Standard


```python
#Read the zip file and relocate the files in the rrf folder
def read_and_relocate_rrf_files(s3, bucket, key):
    try:
        # My columns for the Version Month field
        text_data = key
        text_file = text_data.split('/')[1].split('_')[2].split('.')[0]
        date_string = datetime.strptime(text_file, '%m%d%Y')
        date_object = date_string.strftime('%B %d, %Y')
        version_month.append(date_object)
        print(f"The data for Version Month is: {version_month[0]}")

        # Reading the rrf files
        zip_response = s3.get_object(Bucket=bucket, Key=key)
        zip_data = zip_response['Body'].read()
        file_path = 'rrf'
        destination_path = 'extracted_files'


      # Read the zip file without unzipping it
        with zipfile.ZipFile(io.BytesIO(zip_data)) as zip_ref:
            for file_info in zip_ref.infolist():
                if file_info.filename.startswith(file_path) and not file_info.filename.endswith('/'):
                    filename = os.path.basename(file_info.filename)
                    if filename.endswith('.RRF'):
                        print(f"The {filename} is read from zip file.")
                        with zip_ref.open(file_info) as source_file:
                            s3.put_object(
                                Bucket=bucket,
                                Key=f"{destination_path}/{filename}",
                                Body=source_file.read()
                            )
                        print(f"The {filename} is relocated to {destination_path}")

        return {
            'statusCode': 200,
            'body': json.dumps('Zip file processed successfully')
        }
    except Exception as e:
```

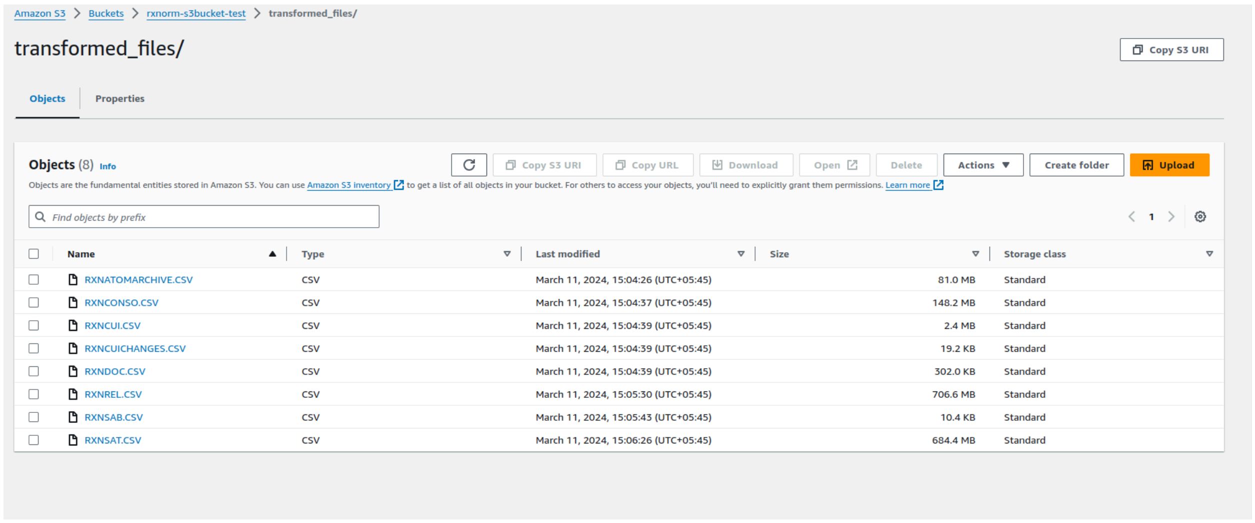
This code will also calculate the data for Version Month column along with relocating the files in the rrf folder to extracted_file

7. **Now the below code will update the columns for the required files**



```
‡ Updating the dataframe for some files
lef update_dataframe(df, file_name):
    if file_name == 'RXNSAB.RRF':
        print(f"Updating the columns VSTART and VEND for {file_name}....")
        df['VSTART'] = df['VSTART'].apply(lambda x: datetime.strptime(x, '%Y_%m_%d').strftime('%Y-%m-%d') if pd.notnull(x) else x)
        df['VEND'] = df['VEND'].apply(lambda x: datetime.strptime(x, '%Y_%m_%d').strftime('%Y-%m-%d') if pd.notnull(x) else x)
    elif file_name == 'RXNATOMARCHIVE.RRF':
        print(f"Updating the columns CREATED_TIMESTAMP,UPDATED_TIMESTAMP and LAST_RELEASED for {file_name}....")
        df['CREATED_TIMESTAMP'] = df['CREATED_TIMESTAMP'].apply(lambda x: datetime.strptime(x, "%m/%d/%Y %I:%M:%S %p").strftime("%Y-%m-%d %I:%M:%S %p") if pd
        df['UPDATED_TIMESTAMP'] = df['UPDATED_TIMESTAMP'].apply(lambda x: datetime.strptime(x, "%m/%d/%Y %I:%M:%S %p").strftime("%Y-%m-%d %I:%M:%S %p") if pd
        df['LAST_RELEASED'] = df['LAST_RELEASED'].fillna('').astype(str).apply(lambda x: datetime.strptime(x, '%d-%b-%y').strftime('%Y-%m-%d') if x else '')
    return df

‡ Function for reading the unzipped rrf files and converting it to csv after adding headers
```

8. **Now read the files from the extracted_file folder and make necessary transformation and dump into the transform_file folder**



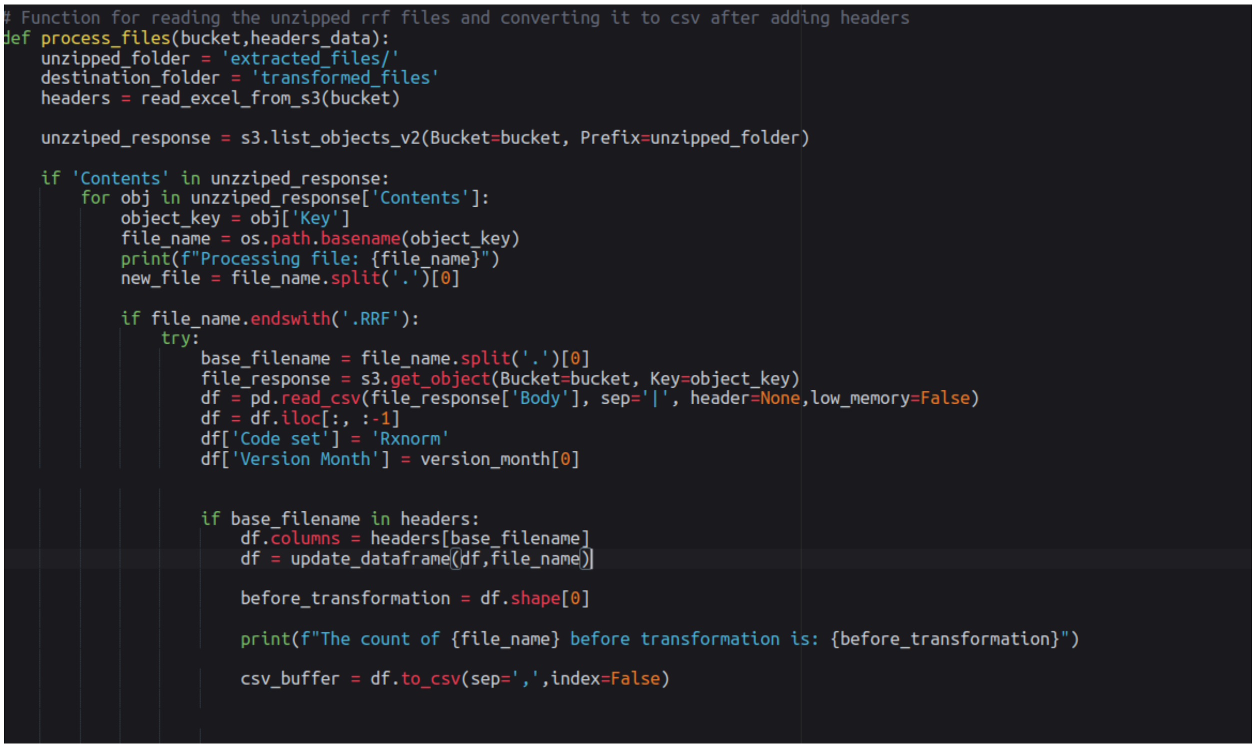
Amazon S3 > Buckets > rxnorm-s3bucket-test > transformed_files/

transformed_files/

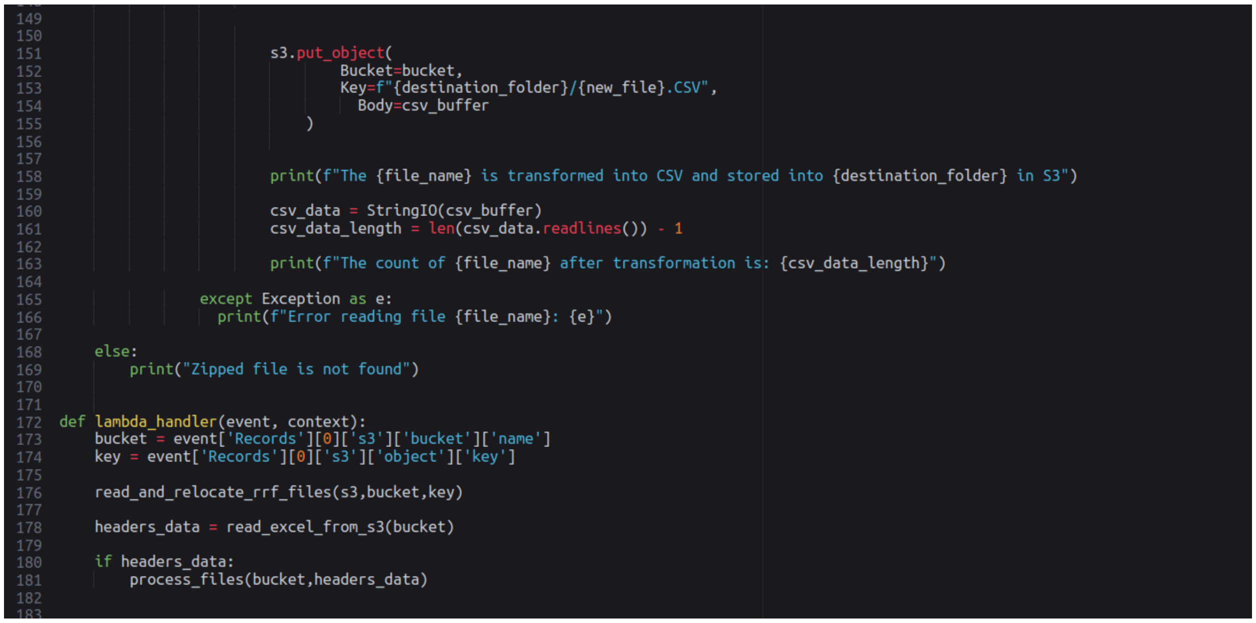
Objects (8)

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.CSV	csv	March 11, 2024, 15:04:26 (UTC+05:45)	81.0 MB	Standard
RXNCONSO.CSV	csv	March 11, 2024, 15:04:37 (UTC+05:45)	148.2 MB	Standard
RXNCUI.CSV	csv	March 11, 2024, 15:04:39 (UTC+05:45)	2.4 MB	Standard
RXNCUICHANGES.CSV	csv	March 11, 2024, 15:04:39 (UTC+05:45)	19.2 KB	Standard
RXNDOC.CSV	csv	March 11, 2024, 15:04:39 (UTC+05:45)	302.0 KB	Standard
RXNREL.CSV	csv	March 11, 2024, 15:05:30 (UTC+05:45)	706.6 MB	Standard
RXNSAB.CSV	csv	March 11, 2024, 15:05:43 (UTC+05:45)	10.4 KB	Standard
RXNSAT.CSV	csv	March 11, 2024, 15:06:26 (UTC+05:45)	684.4 MB	Standard

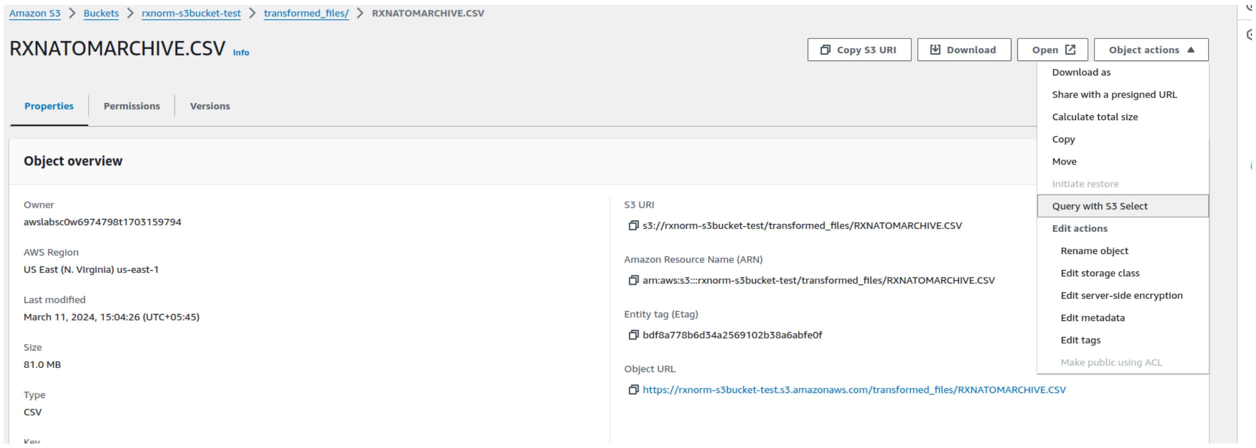


```
‡ Function for reading the unzipped rrf files and converting it to csv after adding headers
lef process_files(bucket,headers_data):
    unzipped_folder = 'extracted_files/'
    destination_folder = 'transformed_files'
    headers = read_excel_from_s3(bucket)

    unzziped_response = s3.list_objects_v2(Bucket=bucket, Prefix=unzipped_folder)

    if 'Contents' in unzziped_response:
        for obj in unzziped_response['Contents']:
            object_key = obj['Key']
            file_name = os.path.basename(object_key)
            print(f"Processing file: {file_name}")
            new_file = file_name.split('.')[0]

            if file_name.endswith('.RRF'):
                try:
                    base_filename = file_name.split('.')[0]
                    file_response = s3.get_object(Bucket=bucket, Key=object_key)
                    df = pd.read_csv(file_response['Body'], sep='|', header=None,low_memory=False)
                    df = df.iloc[:, :-1]
                    df['Code set'] = 'Rxnorm'
                    df['Version Month'] = version_month[0]


                    if base_filename in headers:
                        df.columns = headers[base_filename]
                        df = update_dataframe(df,file_name)

                        before_transformation = df.shape[0]

                        print(f"The count of {file_name} before transformation is: {before_transformation}")

                        csv_buffer = df.to_csv(sep=',',index=False)
```



```
                        s3.put_object(
                                Bucket=bucket,
                                Key=f"{destination_folder}/{new_file}.CSV",
                                  Body=csv_buffer
                            )


                        print(f"The {file_name} is transformed into CSV and stored into {destination_folder} in S3")

                        csv_data = StringIO(csv_buffer)
                        csv_data_length = len(csv_data.readlines()) - 1

                        print(f"The count of {file_name} after transformation is: {csv_data_length}")

                except Exception as e:
                  print(f"Error reading file {file_name}: {e}")

    else:
        print("Zipped file is not found")


def lambda_handler(event, context):
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = event['Records'][0]['s3']['object']['key']

    read_and_relocate_rrf_files(s3,bucket,key)

    headers_data = read_excel_from_s3(bucket)

    if headers_data:
        process_files(bucket,headers_data)
```

9. Now open the transformed file and check the data



RXNATOMARCHIVE.CSV Info

Object overview

Owner
awslabsc0w6974798t1703159794

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

Last modified
March 11, 2024, 15:04:26 (UTC+05:45)

Size
81.0 MB

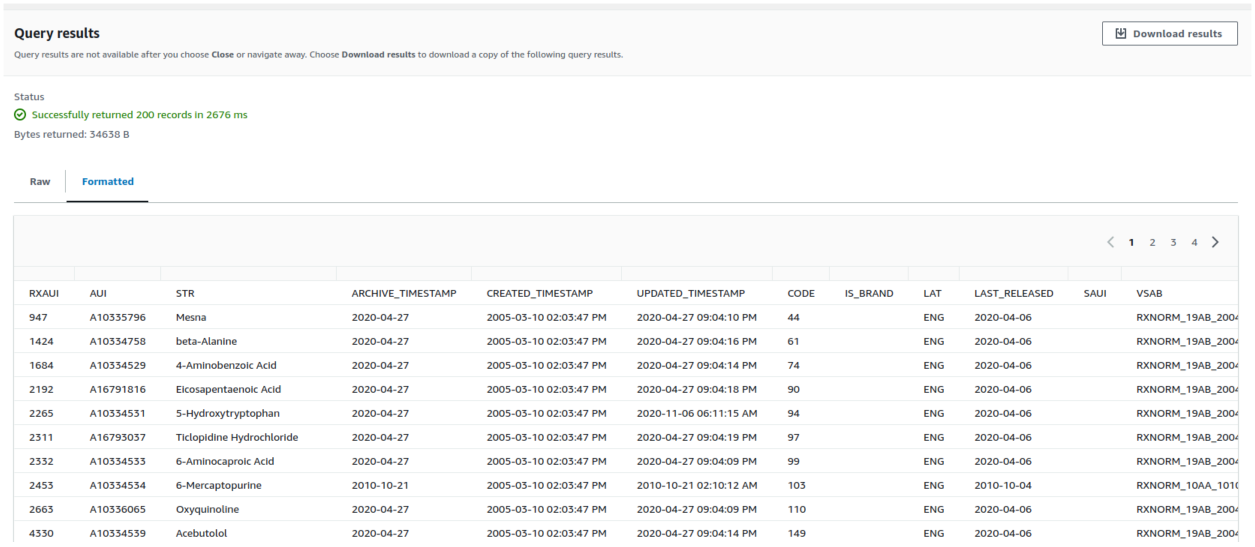
Type
CSV

S3 URI
s3://rxnorm-s3bucket-test/transformed_files/RXNATOMARCHIVE.CSV

Amazon Resource Name (ARN)
arn:aws:s3:::rxnorm-s3bucket-test/transformed_files/RXNATOMARCHIVE.CSV

Entity tag (Etag)
bdf8a778b6d34a2569102b38a6abfe0f

Object URL
https://rxnorm-s3bucket-test.s3.amazonaws.com/transformed_files/RXNATOMARCHIVE.CSV

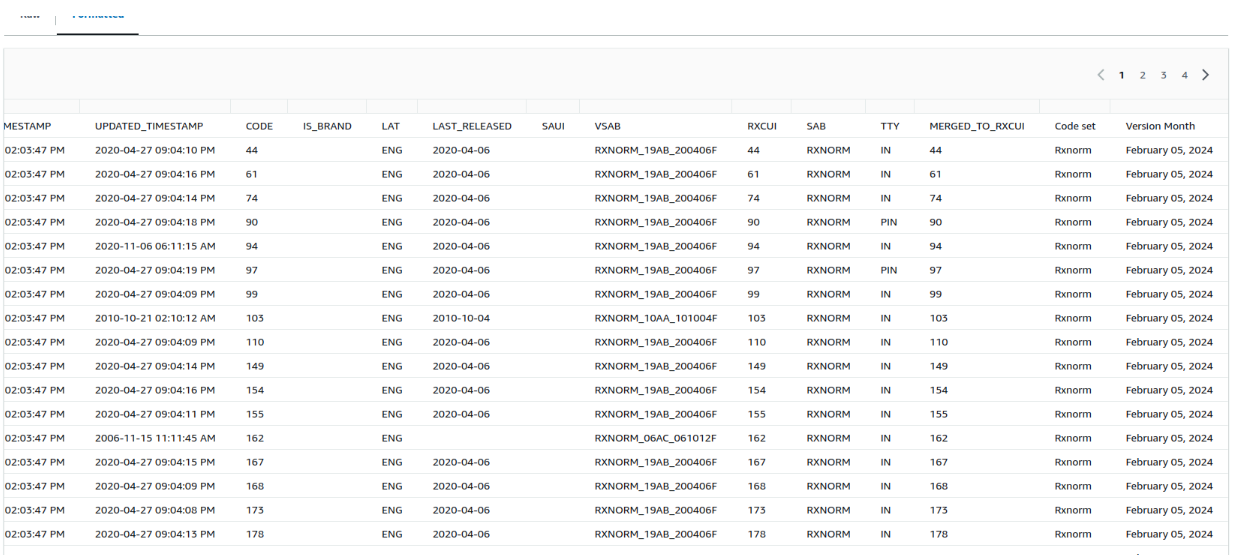


Query results

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

Status
Successfully returned 200 records in 2676 ms
Bytes returned: 34638 B

RXAUI	AUI	STR	ARCHIVE_TIMESTAMP	CREATED_TIMESTAMP	UPDATED_TIMESTAMP	CODE	IS_BRAND	LAT	LAST_RELEASED	SAUI	VSAB
947	A10335796	Mesna	2020-04-27	2005-03-10 02:03:47 PM	2020-04-27 09:04:10 PM	44		ENG	2020-04-06		RXNORM_19AB_2004
1424	A10334758	beta-Alanine	2020-04-27	2005-03-10 02:03:47 PM	2020-04-27 09:04:16 PM	61		ENG	2020-04-06		RXNORM_19AB_2004
1684	A10334529	4-Aminobenzoic Acid	2020-04-27	2005-03-10 02:03:47 PM	2020-04-27 09:04:14 PM	74		ENG	2020-04-06		RXNORM_19AB_2004
2192	A16791816	Eicosapentaenoic Acid	2020-04-27	2005-03-10 02:03:47 PM	2020-04-27 09:04:18 PM	90		ENG	2020-04-06		RXNORM_19AB_2004
2265	A10334531	5-Hydroxytryptophan	2020-04-27	2005-03-10 02:03:47 PM	2020-11-06 06:11:15 AM	94		ENG	2020-04-06		RXNORM_19AB_2004
2311	A16793037	Ticlopidine Hydrochloride	2020-04-27	2005-03-10 02:03:47 PM	2020-04-27 09:04:19 PM	97		ENG	2020-04-06		RXNORM_19AB_2004
2332	A10334533	6-Aminocaproic Acid	2020-04-27	2005-03-10 02:03:47 PM	2020-04-27 09:04:09 PM	99		ENG	2020-04-06		RXNORM_19AB_2004
2453	A10334534	6-Mercaptopurine	2010-10-21	2005-03-10 02:03:47 PM	2010-10-21 02:10:12 AM	103		ENG	2010-10-04		RXNORM_10AA_101
2663	A10336065	Oxyquinoline	2020-04-27	2005-03-10 02:03:47 PM	2020-04-27 09:04:09 PM	110		ENG	2020-04-06		RXNORM_19AB_2004
4330	A10334539	Acebutolol	2020-04-27	2005-03-10 02:03:47 PM	2020-04-27 09:04:14 PM	149		ENG	2020-04-06		RXNORM_19AB_2004

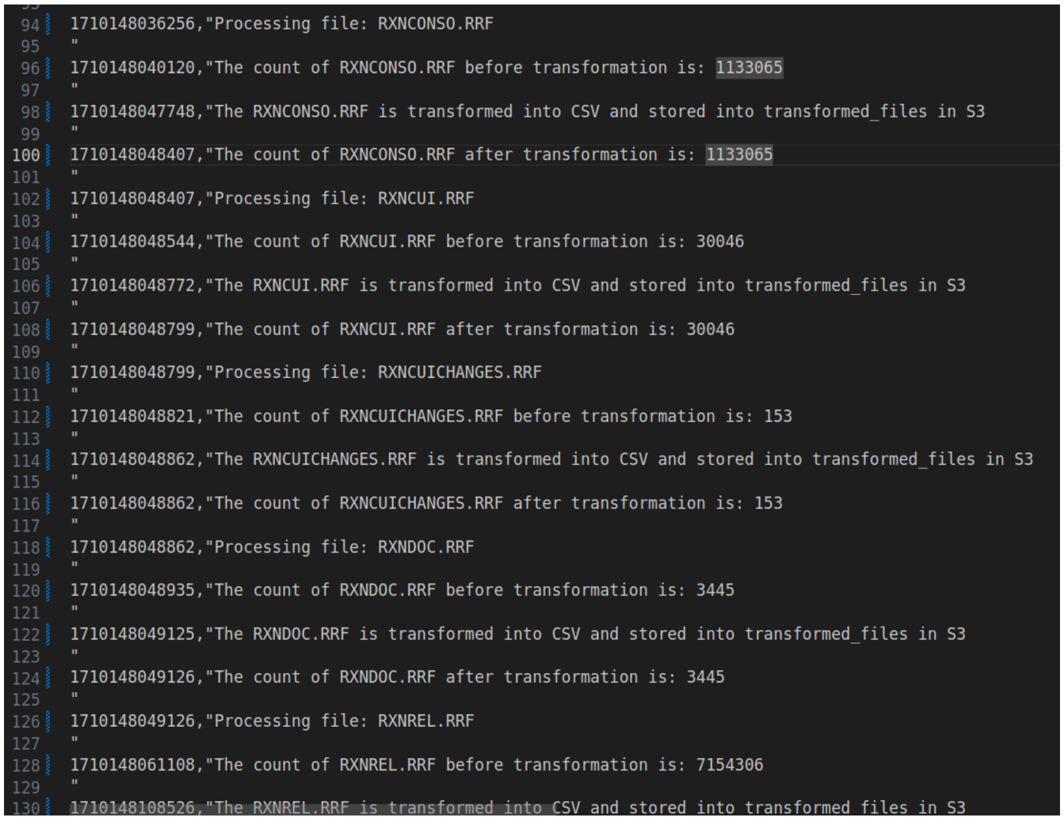


1 2 3 4

MESTAMP	UPDATED_TIMESTAMP	CODE	IS_BRAND	LAT	LAST_RELEASED	SAUI	VSAB	RXCUI	SAB	TTY	MERGED_TO_RXCUI	Code set	Version Month
02:03:47 PM	2020-04-27 09:04:10 PM	44		ENG	2020-04-06		RXNORM_19AB_200406F	44	RXNORM	IN	44	Rxnorm	February 05, 2024
02:03:47 PM	2020-04-27 09:04:16 PM	61		ENG	2020-04-06		RXNORM_19AB_200406F	61	RXNORM	IN	61	Rxnorm	February 05, 2024
02:03:47 PM	2020-04-27 09:04:14 PM	74		ENG	2020-04-06		RXNORM_19AB_200406F	74	RXNORM	IN	74	Rxnorm	February 05, 2024
02:03:47 PM	2020-04-27 09:04:18 PM	90		ENG	2020-04-06		RXNORM_19AB_200406F	90	RXNORM	PIN	90	Rxnorm	February 05, 2024
02:03:47 PM	2020-11-06 06:11:15 AM	94		ENG	2020-04-06		RXNORM_19AB_200406F	94	RXNORM	IN	94	Rxnorm	February 05, 2024
02:03:47 PM	2020-04-27 09:04:19 PM	97		ENG	2020-04-06		RXNORM_19AB_200406F	97	RXNORM	PIN	97	Rxnorm	February 05, 2024
02:03:47 PM	2020-04-27 09:04:09 PM	99		ENG	2020-04-06		RXNORM_19AB_200406F	99	RXNORM	IN	99	Rxnorm	February 05, 2024
02:03:47 PM	2010-10-21 02:10:12 AM	103		ENG	2010-10-04		RXNORM_10AA_101004F	103	RXNORM	IN	103	Rxnorm	February 05, 2024
02:03:47 PM	2020-04-27 09:04:09 PM	110		ENG	2020-04-06		RXNORM_19AB_200406F	110	RXNORM	IN	110	Rxnorm	February 05, 2024
02:03:47 PM	2020-04-27 09:04:14 PM	149		ENG	2020-04-06		RXNORM_19AB_200406F	149	RXNORM	IN	149	Rxnorm	February 05, 2024
02:03:47 PM	2020-04-27 09:04:16 PM	154		ENG	2020-04-06		RXNORM_19AB_200406F	154	RXNORM	IN	154	Rxnorm	February 05, 2024
02:03:47 PM	2020-04-27 09:04:11 PM	155		ENG	2020-04-06		RXNORM_19AB_200406F	155	RXNORM	IN	155	Rxnorm	February 05, 2024
02:03:47 PM	2006-11-15 11:11:45 AM	162		ENG			RXNORM_06AC_061012F	162	RXNORM	IN	162	Rxnorm	February 05, 2024
02:03:47 PM	2020-04-27 09:04:15 PM	167		ENG	2020-04-06		RXNORM_19AB_200406F	167	RXNORM	IN	167	Rxnorm	February 05, 2024
02:03:47 PM	2020-04-27 09:04:09 PM	168		ENG	2020-04-06		RXNORM_19AB_200406F	168	RXNORM	IN	168	Rxnorm	February 05, 2024
02:03:47 PM	2020-04-27 09:04:08 PM	173		ENG	2020-04-06		RXNORM_19AB_200406F	173	RXNORM	IN	173	Rxnorm	February 05, 2024
02:03:47 PM	2020-04-27 09:04:13 PM	178		ENG	2020-04-06		RXNORM_19AB_200406F	178	RXNORM	IN	178	Rxnorm	February 05, 2024

We can check all the data like wise .

10. **For the verification of row counts before and after the transformation we can check the log file which have logs for each file for the entire processing As shown below**



```
1710148036256,"Processing file: RXNCONSO.RRF
"
1710148040120,"The count of RXNCONSO.RRF before transformation is: 1133065
"
1710148047748,"The RXNCONSO.RRF is transformed into CSV and stored into transformed_files in S3
"
1710148048407,"The count of RXNCONSO.RRF after transformation is: 1133065
"
1710148048407,"Processing file: RXNCUI.RRF
"
1710148048544,"The count of RXNCUI.RRF before transformation is: 30046
"
1710148048772,"The RXNCUI.RRF is transformed into CSV and stored into transformed_files in S3
"
1710148048799,"The count of RXNCUI.RRF after transformation is: 30046
"
1710148048799,"Processing file: RXNCUICHANGES.RRF
"
1710148048821,"The count of RXNCUICHANGES.RRF before transformation is: 153
"
1710148048862,"The RXNCUICHANGES.RRF is transformed into CSV and stored into transformed_files in S3
"
1710148048862,"The count of RXNCUICHANGES.RRF after transformation is: 153
"
1710148048862,"Processing file: RXNDOC.RRF
"
1710148048935,"The count of RXNDOC.RRF before transformation is: 3445
"
1710148049125,"The RXNDOC.RRF is transformed into CSV and stored into transformed_files in S3
"
1710148049126,"The count of RXNDOC.RRF after transformation is: 3445
"
1710148049126,"Processing file: RXNREL.RRF
"
1710148061108,"The count of RXNREL.RRF before transformation is: 7154306
"
1710148108526,"The RXNREL.RRF is transformed into CSV and stored into transformed_files in S3
```