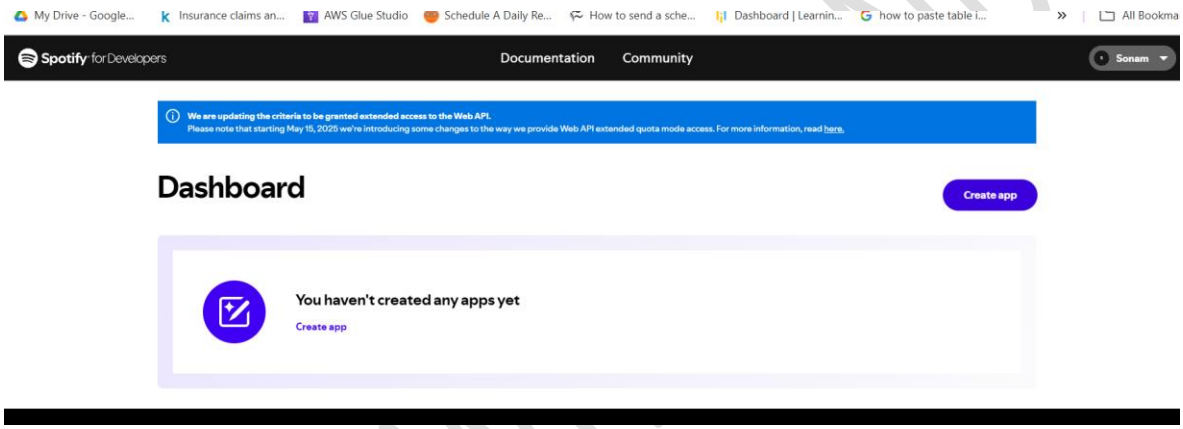
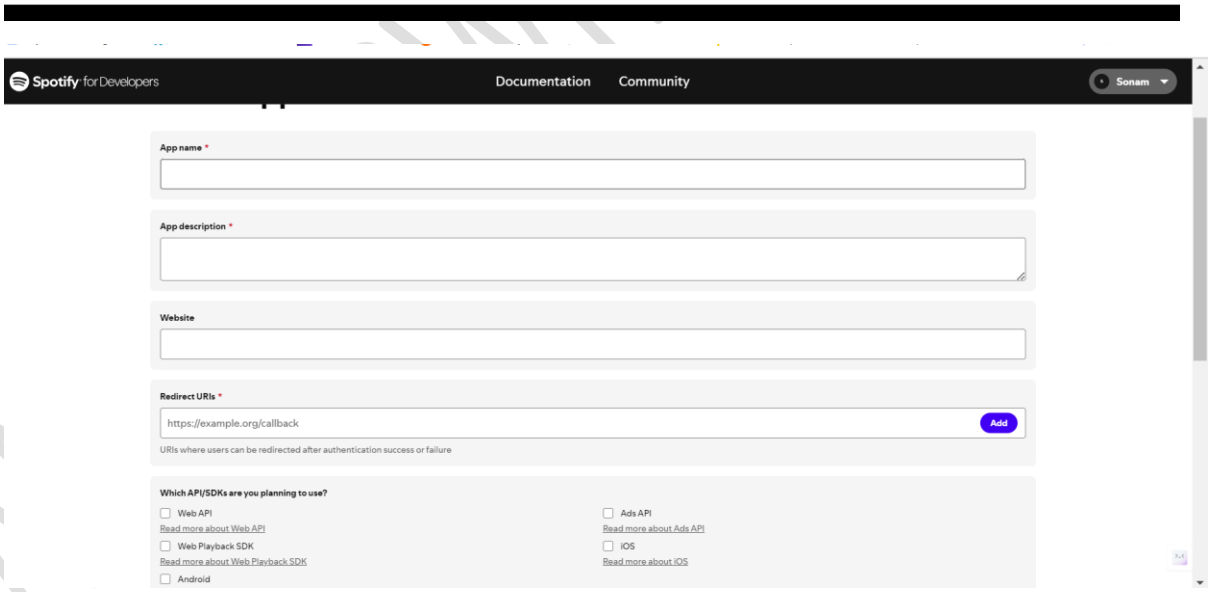
🔹 1. Go to Spotify Developer Dashboard

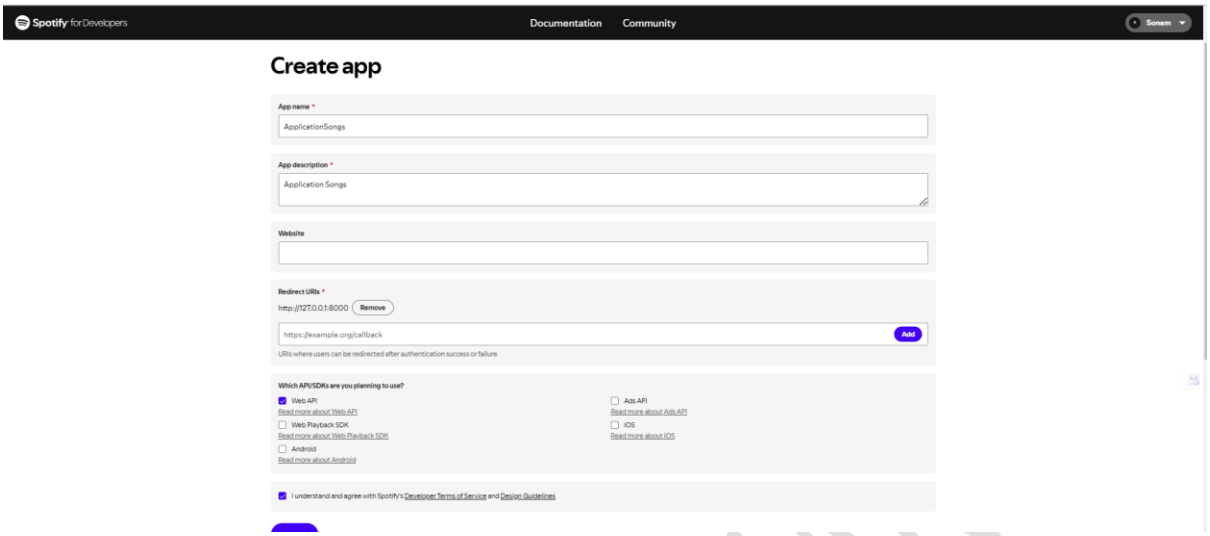
👉 Open this link:
https://developer.spotify.com/dashboard

🔹 2. Log In

- Use your **Spotify Account** (Google, Facebook, or Email).
- It should open the Spotify Developer Console.







🔹 3. Find Your App

You should see your app listed there, for example:
spotify-etl-pipeline (or whatever you named it).

Click on your app.

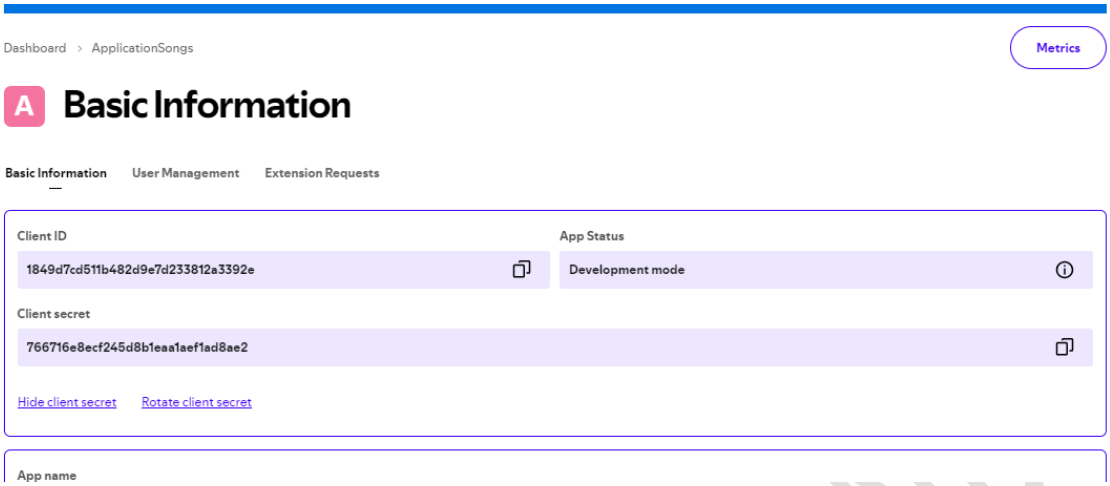
🔹 4. Copy Client ID and Client Secret

Inside your App details page, you will see:

Field	Description
Client ID	It's a long alphanumeric string
Client Secret	It's hidden — you need to click "Show client secret"

Example:

Key	Example Value
Client ID	1849d7cd511b482d9e7d233812a3392e
Client Secret	766716e8ecf245d8b1eaa1aef1ad8ae2



https://open.spotify.com/

Get the URI - 7sMIveCzjMrCLn0cDi91dM

- **Step 5: Create an S3 Bucket**
  1. Go to the AWS Console → Search for **S3**
  2. Click **Create Bucket**
  3. Bucket name: spotify-etl-project-sonam
  4. Region: Choose your closest AWS region
  5. Leave other defaults, click **Create Bucket**

spotify-etl-project-sonam/

├— raw/

| ├— to_be_processed/

| └— processed/

└— transformed/

├— songs_data/

├— album_data/

└— artists_data/

- Python code for first lambda function

```
import json, os, boto3
import spotipy
from datetime import datetime
from spotipy.oauth2 import SpotifyClientCredentials

def lambda_handler(event, context):
    client_id = os.environ['SPOTIFY_CLIENT_ID']
    client_secret = os.environ['SPOTIFY_CLIENT_SECRET']

    credentials = SpotifyClientCredentials(client_id, client_secret)
    sp = spotipy.Spotify(auth_manager=credentials)

    playlist_id = "37i9dQZEVXbLZ52XmnySJg"  # Top 50 - India
    data = sp.playlist_tracks(playlist_id)

    s3 = boto3.client("s3")
    filename = "spotify_raw_" + str(datetime.now()) + ".json"

    s3.put_object(
        Bucket="spotify-etl-project-ameet",
        Key="raw/to_be_processed/" + filename,
        Body=json.dumps(data)
    )
```

🔹 **Step 6: Install Required Packages (Spotipy)**

Since Lambda doesn't include external libraries like spotipy by default, you need to:

Option A: (Recommended) Create a Deployment Package

1. On your computer, make a folder:

   bash

   CopyEdit

   mkdir spotify_lambdas && cd spotify_lambdas
2. Install dependencies inside the folder:

   bash

   CopyEdit

   pip install spotipy -t .

   pip install boto3 -t .
3. Add your Python script:
   - Save the code in a file: lambda_function.py

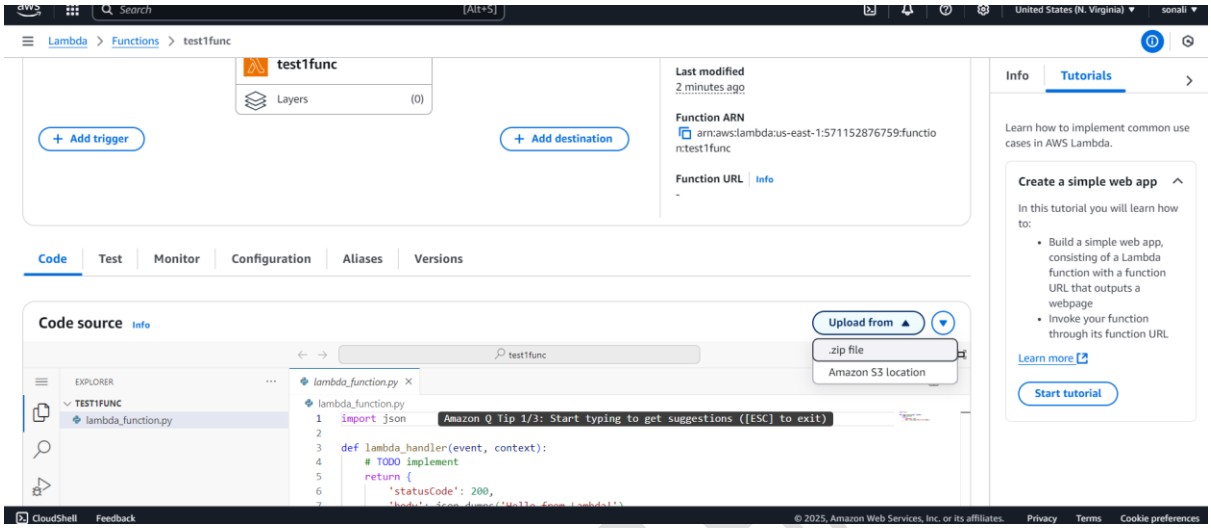
🛠️ **Deployment Steps:**

1. Go to **AWS Lambda** → Click **Create Function**
2. Name: spotify-extract-lambda
3. Runtime: **Python 3.9**
4. Click **Create**

- **Step 7**

  Go back to Lambda Console:

  - In **Code Source**, click **Upload from → .zip file**
  - Upload spotify_lambda.zip

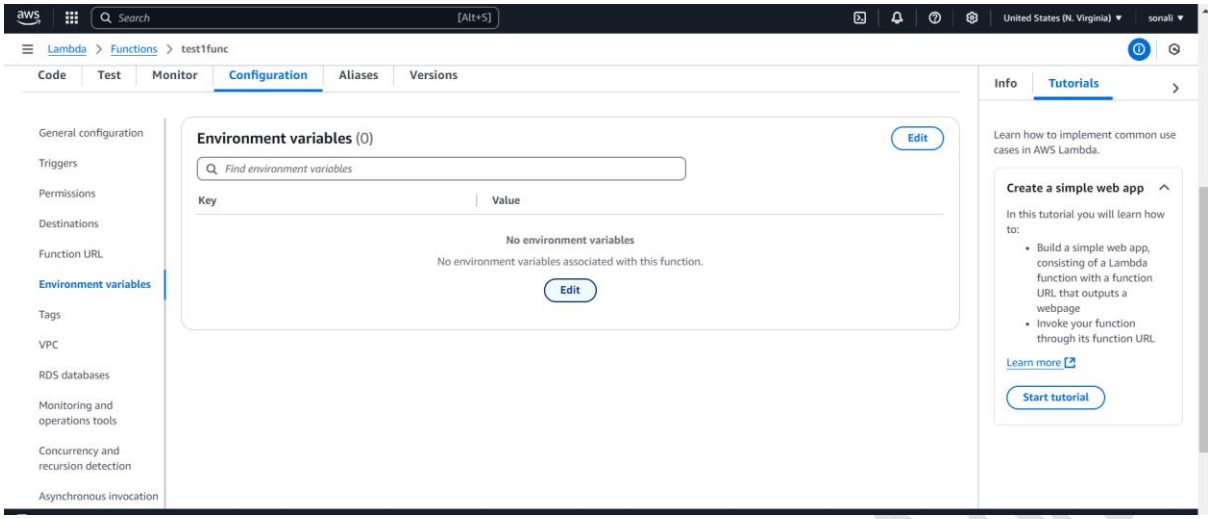


- **Step 8: Add Environment Variables (Client ID & Secret)**

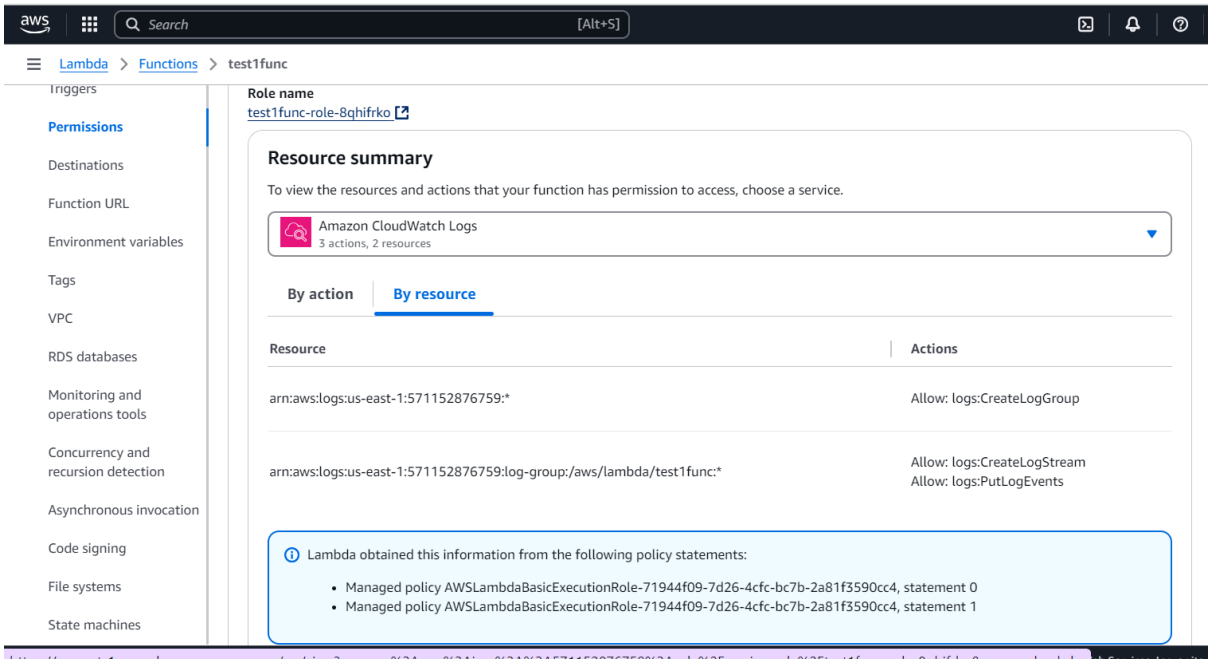
  1. In Lambda Console → Go to the **Configuration** tab
  2. Click on **Environment variables**
  3. Click **Edit**, then **Add environment variable**

Add:

Key	**Value**
SPOTIFY_CLIENT_ID	your actual client ID
SPOTIFY_CLIENT_SECRET	your actual client secret



- **Step 9: Add S3 Access Permissions**
  1. In **Configuration → Permissions**
  2. Under **Execution role**, click the role name
  3. In IAM, click **Add permissions → Attach policies**
  4. Search for and select:
     - AmazonS3FullAccess
     - (Optional: AWSCloudWatchLogsFullAccess for logs)



- **Step 7: Test Your Lambda Function**
  1. Go to the **Test** tab
  2. Click **“Create a new test event”**
     - Name: spotifyTest
     - Keep the event JSON as {}
  3. Click **Test**

Incase of module error ::-

pip install async_timeout -t .

Output of test : Status: Succeeded

Test Event Name: spotifyTestings

Response:

null

Function Logs:

START RequestId: a98ca860-5a7b-4d0f-bb09-b74504ab91ce Version: $LATEST

[WARNING] 2025-04-27T07:44:46.876Z a98ca860-5a7b-4d0f-bb09-b74504ab91ce Couldn't write token to cache at: .cache

[WARNING] 2025-04-27T07:44:47.243Z a98ca860-5a7b-4d0f-bb09-b74504ab91ce Couldn't write token to cache at: .cache

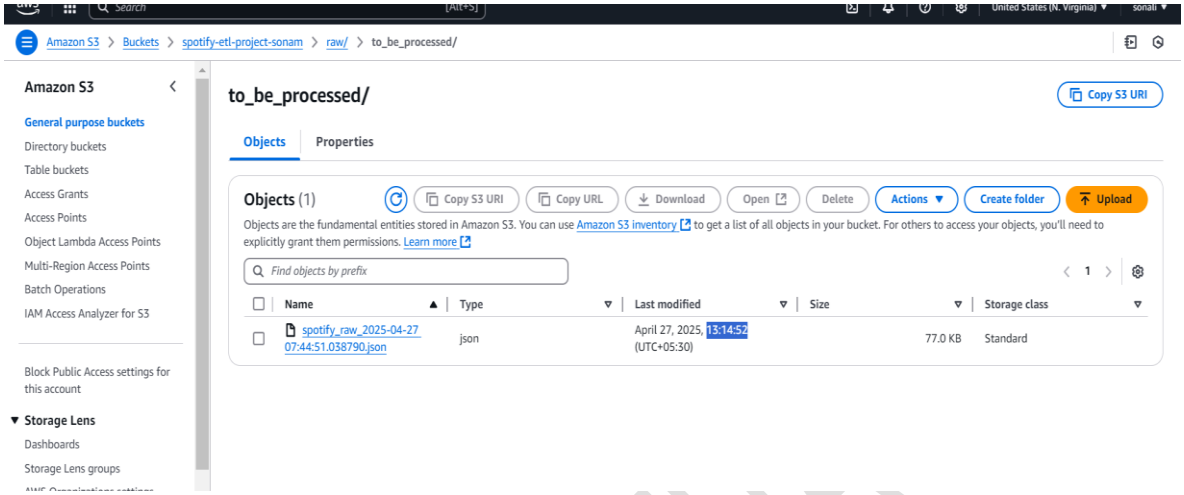
END RequestId: a98ca860-5a7b-4d0f-bb09-b74504ab91ce

REPORT RequestId: a98ca860-5a7b-4d0f-bb09-b74504ab91ce Duration: 4633.37 ms Billed Duration: 4634 ms Memory Size: 128 MB Max Memory Used: 91 MB Init Duration: 975.14 ms

Request ID: a98ca860-5a7b-4d0f-bb09-b74504ab91ce

If successful:

- You'll see a green success message
- Check your S3 bucket → raw/to_be_processed/ should have a JSON file!



Optional Step 8: Add a CloudWatch Trigger (Run Weekly)

1. Go to **Configuration → Triggers**
2. Click **Add trigger**
3. Choose:
    - **EventBridge (CloudWatch Events)**
    - Create new rule
    - Schedule: cron(0 12 ? * MON *)
4. Click **Add**

First part is finished

1. Create a Second Lambda Function: "spotify-transformer-lambdas"

- Go to AWS Lambda Console
- Click **Create Function**
- Name it: **spotify-transformer-lambdas**
- Runtime: **Python 3.9**
- Permissions: Give it **S3 Full Access** IAM Role (just like extract Lambda)

- **2: Install Required Packages (Spotipy)**

Since Lambda doesn't include external libraries like spotipy by default, you need to:

Option A: (Recommended) Create a Deployment Package

1. On your computer, create a folder

mkdir spotify_transformer_lambdas

cd spotify_transformer_lambdas

- 2. Install pandas into this folder

pip install pandas -t .

pip install boto3 -t .

- **3. Add your Lambda function code**

Create a file lambda_function.py in the same folder and paste your transformation code inside it.

This ZIP file will now have:

- pandas/
- boto3/
- your lambda_function.py

- **5. Upload ZIP to Lambda**

In Lambda console:

- Go to your function
- Click **Upload from → .zip file**
- Upload your spotify_transformer_lambdas
- .zip
- Click **Deploy**
-

Arn used for 2nd lambda function

arn:aws:lambda:us-east-1:336392948345:layer:AWSSDKPandas-Python39:5

🔹 **D. Test Manually**

1. Click **Test**
2. Use test event: {}
3. Run it!

✅ You should see:

- CSVs in transformed/ folders
- JSONs moved to raw/processed/

STEP 5: Set Up Glue Database & Crawlers

🎯 **Goal:**

Convert S3 CSVs into queryable tables

🔹 **A. Create Glue Database**

1. Go to **AWS Glue**
2. Click **Databases → Add Database**
3. Name it: spotify_db

🔹 **A. Create Glue Database**

1. Go to **AWS Glue**
2. Click **Databases → Add Database**
3. Name it: spotify_db

🔹 **B. Create 3 Crawlers**

Repeat 3 times:

Crawler Name	S3 Path
crawler_songs	s3://spotify-etl-project-ameet/transformed/songs_data/
crawler_albums	s3://spotify-etl-project-ameet/transformed/album_data/
crawler_artists	s3://spotify-etl-project-ameet/transformed/artists_data/

Steps:

- Go to **Crawlers → Add Crawler**

- Source: S3
- Target DB: spotify_db
- IAM Role: Create or reuse Glue role with S3 access
- Run crawler

STEP 6: Use Athena to Query the Data

🎯 **Goal:**

Run SQL queries on your Spotify data

🔹 **A. Go to Amazon Athena**

1. Set **query result location**:

arduino

CopyEdit

s3://spotify-etl-project-ameet/athena-results/

2. Choose Database: spotify_db

3. 🔹 B. Example Queries

```
4. sql
5. CopyEdit
6. -- Top 10 popular songs
7. SELECT song_name, popularity
8. FROM songs_data
9. ORDER BY popularity DESC
10.  LIMIT 10;
11.
12.  -- Count albums per artist
13.  SELECT artist_id, COUNT(DISTINCT album_id) AS album_count
14.  FROM songs_data
15.  GROUP BY artist_id;
16.
17.  -- Albums after 2020
18.  SELECT name, release_date
19.  FROM album_data
20.  WHERE release_date > DATE '2020-01-01';
```