**How to load API data into Snowflake:**

**Step-by-Step: Simulate API and Load into Snowflake**

**✅ Step 1: Create the CSV File**

1. Open **Notepad** (Windows) or **Text Editor** (Linux/Mac).
2. Paste this content:

csv

order\_id,customer\_name,product,quantity,price,total,date

1001,Anita Sharma,Wireless Mouse,2,750,1500,2025-09-28

1002,Ravi Kumar,Laptop Stand,1,1200,1200,2025-09-28

1003,Meena Joshi,USB-C Hub,3,500,1500,2025-09-29

1004,Suresh Reddy,Keyboard,1,1800,1800,2025-09-29

1005,Divya Rao,Monitor 24",2,8500,17000,2025-09-30

1. Save the file as sales.csv in a folder like:

Code

C:\Pythontestfiles>

**✅ Step 2: Start a Local HTTP Server**

1. Open **Command Prompt**.
2. Navigate to the folder:

bash

1. C:\Pythontestfiles>

Start the server:

bash

**python -m http.server 8000**

1. You’ll see output like:

Code

Serving HTTP on 0.0.0.0 port 8000 ...

**✅ Step 3: Test the URL**

1. Open your browser.
2. Go to:

Code

http://localhost:8000/sales.csv

1. You should see the CSV content displayed—this simulates an API endpoint.

import requests

import pandas as pd

from snowflake.snowpark import Session

import io

# Step 1: Fetch CSV from local server

csv\_url = "http://localhost:8000/sales.csv"

response = requests.get(csv\_url)

df = pd.read\_csv(io.StringIO(response.text))

# 🌟 CORRECTION: Convert column names to uppercase to match Snowflake's default behavior.

# This fixes the "invalid identifier" error for columns like 'order\_id'.

df.columns = df.columns.str.upper()

# Step 2: Define Snowflake connection parameters (Using placeholder values for security)

connection\_parameters = {

# WARNING: Do not hardcode credentials in production code. Use environment variables or secrets management.

"account": "JABGDWO-GY97629",

"user": "prabhakarreddy1433",

"password": "Mysnowflake149$",

"role": "ACCOUNTADMIN",

"warehouse": "COMPUTE\_WH",

"database": "MYSNOW",

"schema": "PUBLIC"

}

# Step 3: Connect to Snowflake

try:

session = Session.builder.configs(connection\_parameters).create()

print("Successfully connected to Snowflake. 🎉")

# Step 4: Load data into Snowflake table

# write\_pandas is now successful because the DataFrame column names are uppercase (e.g., 'ORDER\_ID')

session.write\_pandas(df, "SALES\_DATA", auto\_create\_table=True, overwrite=True)

print(f"Data successfully loaded into table MYSNOW.PUBLIC.SALES\_DATA.")

except Exception as e:

print(f"An error occurred: {e}")

finally:

# Step 5: Close the session

if 'session' in locals() and session:

session.close()

print("Snowflake session closed.")