```
from google.colab import userdata
vantage_api_key = userdata.get('API_Vantage')
```

```
user_id = userdata.get('snowflake_userid')
password = userdata.get('snowflake_password')
account = userdata.get('snowflake_account')
```

```
symbol = symbol = "MSFT"

url = f'https://www.alphavantage.co/query?function=TIME_SERIES_DAILY&symbol={symbol}&apikey={vantage_api_key }'
```

```
import requests  # Import the requests library
r = requests.get(url)
```

```
data = r.json() # Convert the data into JSON format
```

```
import requests
from datetime import datetime, timedelta

def return_last_90d_price(symbol):
    """
    - return the last 90 days of the stock prices of symbol as a list of json strings
    """
    vantage_api_key = userdata.get('API_Vantage')  # Assuming userdata is already defined
    url = f'https://www.alphavantage.co/query?function=TIME_SERIES_DAILY&symbol={symbol}&apikey={ vantage_api_key}'

    r = requests.get(url)
    data = r.json()

    results = []  # List to hold the last 90 days of stock info
    ninety_days_ago = datetime.today() - timedelta(days=90)  # Get the date 90 days ago

    # Iterate through the daily data and filter for the last 90 days along with date info
    for d in data["Time Series (Daily)"]:
        date_obj = datetime.strptime(d, "%Y-%m-%d")
        if date_obj >= ninety_days_ago:
          # append the data in clear format using table or another result dictionary
          price_data={

              "date":d,
              "open":data["Time Series (Daily)"][d]["1. open"],
              "high":data["Time Series (Daily)"][d]["2. high"],
              "low":data["Time Series (Daily)"][d]["3. low"],
              "close":data["Time Series (Daily)"][d]["4. close"],
              "volume":data["Time Series (Daily)"][d]["5. volume"],
              "symbol":symbol
          }

          results.append(price_data)

    return results
```

```
price_list=return_last_90d_price("MSFT")
```

```
price_list[0]
```

```
{'date': '2024-09-25',
 'open': '429.8300',
 'high': '433.1190',
 'low': '428.5700',
 'close': '432.1100',
 'volume': '13396364',
 'symbol': 'MSFT'}
```

In 2024, U.S. stock markets are open Monday through Friday, excluding weekends and market holidays. From June to September 2024, the major holidays to consider are:

Independence Day: July 4, 2024 (market closed) Labor Day: September 2, 2024 (market closed) Excluding these holidays and weekends, you can calculate the number of trading days over the 90-day period, which would likely fall between 60 and 65 trading days.

Defining ETL functions

```
import requests

def extract(url):
    f = requests.get(url)
    return (f.text)
    print(f.text)


def transform(text):
    lines = text.strip().split("\n")
    records = []
    for l in lines:
      (date,open,high,low,volume) = l.split(",")
      records.append([date,open,high,low,volume])
    # return the records except the first entry
    return records
```

```
pip install snowflake-connector-python
```

```
Collecting snowflake-connector-python
  Downloading snowflake_connector_python-3.12.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (65 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 65.3/65.3 kB 4.8 MB/s eta 0:00:00
Collecting asn1crypto<2.0.0,>0.24.0 (from snowflake-connector-python)
  Downloading asn1crypto-1.5.1-py2.py3-none-any.whl.metadata (13 kB)
Requirement already satisfied: cffi<2.0.0,>=1.9 in /usr/local/lib/python3.10/dist-packages (from snowflake-connector-python)
Requirement already satisfied: cryptography>=3.1.0 in /usr/local/lib/python3.10/dist-packages (from snowflake-connector-pyth
Requirement already satisfied: pyOpenSSL<25.0.0,>=16.2.0 in /usr/local/lib/python3.10/dist-packages (from snowflake-connecto
Requirement already satisfied: pyjwt<3.0.0 in /usr/local/lib/python3.10/dist-packages (from snowflake-connector-python) (2.9
Requirement already satisfied: pytz in /usr/local/lib/python3.10/dist-packages (from snowflake-connector-python) (2024.2)
Requirement already satisfied: requests<3.0.0 in /usr/local/lib/python3.10/dist-packages (from snowflake-connector-python) (
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from snowflake-connector-python) (24.1)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from snowflake-connector
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from snowflake-connector-python) (3.
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from snowflake-connector-pytho
Requirement already satisfied: typing-extensions<5,>=4.3 in /usr/local/lib/python3.10/dist-packages (from snowflake-connecto
Requirement already satisfied: filelock<4,>=3.5 in /usr/local/lib/python3.10/dist-packages (from snowflake-connector-python)
Requirement already satisfied: sortedcontainers>=2.4.0 in /usr/local/lib/python3.10/dist-packages (from snowflake-connector-
Requirement already satisfied: platformdirs<5.0.0,>=2.6.0 in /usr/local/lib/python3.10/dist-packages (from snowflake-connect
Collecting tomlkit (from snowflake-connector-python)
  Downloading tomlkit-0.13.2-py3-none-any.whl.metadata (2.7 kB)
Requirement already satisfied: pycparser in /usr/local/lib/python3.10/dist-packages (from cffi<2.0.0,>=1.9->snowflake-connec
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0->snowflake
Downloading snowflake_connector_python-3.12.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (2.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.5/2.5 MB 67.4 MB/s eta 0:00:00
Downloading asn1crypto-1.5.1-py2.py3-none-any.whl (105 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 105.0/105.0 kB 8.3 MB/s eta 0:00:00
Downloading tomlkit-0.13.2-py3-none-any.whl (37 kB)
Installing collected packages: asn1crypto, tomlkit, snowflake-connector-python
Successfully installed asn1crypto-1.5.1 snowflake-connector-python-3.12.2 tomlkit-0.13.2
```

After installing establist connection to Snowflake

```
import snowflake.connector

def return_snowflake_conn():
    # Establish a connection to Snowflake
    conn = snowflake.connector.connect(
        user=user_id,          # Replace with your actual user
        password=password,  # Replace with your actual password
        account=account,    # e.g., 'xyz12345.eu-central-2'
        warehouse='compute_wh',
        database='demoAPI',
        schema='raw_data',
        role='accountadmin',
        ocsp_fail_open=False
    )
 #create a cursor object so that sql knows the starting point of executiom of commands
    return conn.cursor()
```

```
# This function returns a list of price data for the last 90 days
price_list = return_last_90d_price("MSFT")  # This should give you the structured data

def load(con, records):
    # Check if records is empty
    if not records:
        print("No records to load.")
        return  # Exit if there are no records

    # Define the target table for price data
    target_table = "demoAPI.raw_data.stock_price"

    # Create the table if it does not exist
    conn.execute(f"""
    CREATE or Replace TABLE {target_table} (
      date DATE Primary Key,
      symbol VARCHAR,
      open NUMBER,
      high NUMBER,
      low NUMBER,
      close NUMBER,
      volume NUMBER
    )""")

    # Load records into the table
    for r in records:
        date = r['date']
        symbol = r['symbol']
        open_price = r['open']
        high_price = r['high']
        low_price = r['low']
        close_price = r['close']
        volume = r['volume']


        print(f"Inserting data for {date}: Open={open_price}, Symbol='{symbol}', High={high_price}, Low={low_price}, Close={clos

        # Use parameterized INSERT INTO to avoid SQL injection and special character issues
        sql = f"""
        INSERT INTO {target_table} (date, symbol, open, high, low, close, volume)
        VALUES (TO_DATE('{date}', 'YYYY-MM-DD'),'{symbol}', {open_price}, {high_price}, {low_price}, {close_price}, {volume})
        """

        conn.execute(sql)

# Get the Snowflake connection
conn = return_snowflake_conn()

# Load the data into the Snowflake table
load(conn, price_list)
```

```
Inserting data for 2024-09-25: Open=429.8300, Symbol='MSFT', High=433.1190, Low=428.5700, Close=432.1100, Volume=13396364
Inserting data for 2024-09-24: Open=433.0000, Symbol='MSFT', High=433.3500, Low=426.1000, Close=429.1700, Volume=17015754
Inserting data for 2024-09-23: Open=434.2800, Symbol='MSFT', High=436.4600, Low=430.3889, Close=433.5100, Volume=15128891
Inserting data for 2024-09-20: Open=437.2200, Symbol='MSFT', High=439.2400, Low=434.2200, Close=435.2700, Volume=55167106
Inserting data for 2024-09-19: Open=441.2250, Symbol='MSFT', High=441.5000, Low=436.9000, Close=438.6900, Volume=21706559
Inserting data for 2024-09-18: Open=435.0000, Symbol='MSFT', High=436.0300, Low=430.4100, Close=430.8100, Volume=18898042
Inserting data for 2024-09-17: Open=440.2300, Symbol='MSFT', High=441.8500, Low=432.2700, Close=435.1500, Volume=18874231
Inserting data for 2024-09-16: Open=430.6000, Symbol='MSFT', High=433.5300, Low=428.2200, Close=431.3400, Volume=13834697
Inserting data for 2024-09-13: Open=425.8250, Symbol='MSFT', High=431.8300, Low=425.4600, Close=430.5900, Volume=15874555
Inserting data for 2024-09-12: Open=423.3100, Symbol='MSFT', High=427.3692, Low=419.7500, Close=427.0000, Volume=17418759
Inserting data for 2024-09-11: Open=415.5000, Symbol='MSFT', High=423.9900, Low=409.5800, Close=423.0400, Volume=19266923
Inserting data for 2024-09-10: Open=408.2000, Symbol='MSFT', High=416.3300, Low=407.7000, Close=414.2000, Volume=19594287
Inserting data for 2024-09-09: Open=407.2400, Symbol='MSFT', High=408.6500, Low=402.1500, Close=405.7200, Volume=15295134
Inserting data for 2024-09-06: Open=409.0600, Symbol='MSFT', High=410.6500, Low=400.8000, Close=401.7000, Volume=19609526
Inserting data for 2024-09-05: Open=407.6200, Symbol='MSFT', High=413.1000, Low=406.1300, Close=408.3900, Volume=14195516
Inserting data for 2024-09-04: Open=405.9100, Symbol='MSFT', High=411.2400, Low=404.3700, Close=408.9000, Volume=14886710
Inserting data for 2024-09-03: Open=417.9100, Symbol='MSFT', High=419.8800, Low=407.0300, Close=409.4400, Volume=20313603
Inserting data for 2024-08-30: Open=415.6000, Symbol='MSFT', High=417.4900, Low=412.1300, Close=417.1400, Volume=24308324
Inserting data for 2024-08-29: Open=414.9400, Symbol='MSFT', High=422.0500, Low=410.6000, Close=413.1200, Volume=17045241
Inserting data for 2024-08-28: Open=414.8800, Symbol='MSFT', High=415.0000, Low=407.3100, Close=410.6000, Volume=14882660
Inserting data for 2024-08-27: Open=412.8600, Symbol='MSFT', High=414.3600, Low=410.2500, Close=413.8400, Volume=13492911
Inserting data for 2024-08-26: Open=416.3650, Symbol='MSFT', High=417.2799, Low=411.3400, Close=413.4900, Volume=13152830
Inserting data for 2024-08-23: Open=416.9800, Symbol='MSFT', High=419.2600, Low=412.0900, Close=416.7900, Volume=18493784
Inserting data for 2024-08-22: Open=424.3600, Symbol='MSFT', High=426.7899, Low=414.6100, Close=415.5500, Volume=19361901
Inserting data for 2024-08-21: Open=424.0750, Symbol='MSFT', High=426.4000, Low=421.7200, Close=424.1400, Volume=16067298
Inserting data for 2024-08-20: Open=421.7000, Symbol='MSFT', High=425.8600, Low=421.6400, Close=424.8000, Volume=16387581
Inserting data for 2024-08-19: Open=418.9600, Symbol='MSFT', High=421.7500, Low=416.4600, Close=421.5300, Volume=15233957
Inserting data for 2024-08-16: Open=420.6000, Symbol='MSFT', High=421.3400, Low=417.3000, Close=418.4700, Volume=22775593
Inserting data for 2024-08-15: Open=419.8000, Symbol='MSFT', High=421.1100, Low=417.6600, Close=421.0300, Volume=20752144
Inserting data for 2024-08-14: Open=414.8000, Symbol='MSFT', High=417.7200, Low=412.4456, Close=416.8600, Volume=18266980
Inserting data for 2024-08-13: Open=409.5900, Symbol='MSFT', High=414.9500, Low=409.5700, Close=414.0100, Volume=19414271
Inserting data for 2024-08-12: Open=407.0600, Symbol='MSFT', High=408.7600, Low=404.2434, Close=406.8100, Volume=16762883
Inserting data for 2024-08-09: Open=404.0300, Symbol='MSFT', High=408.0500, Low=402.2624, Close=406.0200, Volume=19276666
Inserting data for 2024-08-08: Open=402.4400, Symbol='MSFT', High=405.8600, Low=399.9407, Close=402.6900, Volume=20203030
Inserting data for 2024-08-07: Open=408.6350, Symbol='MSFT', High=410.0800, Low=397.4700, Close=398.4300, Volume=20650906
Inserting data for 2024-08-06: Open=400.0000, Symbol='MSFT', High=405.6700, Low=398.5000, Close=399.6100, Volume=24946525
Inserting data for 2024-08-05: Open=389.1700, Symbol='MSFT', High=401.0400, Low=385.5800, Close=395.1500, Volume=40709238
Inserting data for 2024-08-02: Open=412.4900, Symbol='MSFT', High=415.0000, Low=404.3400, Close=408.4900, Volume=29437900
Inserting data for 2024-08-01: Open=420.7850, Symbol='MSFT', High=427.4600, Low=413.0901, Close=417.1100, Volume=30296400
Inserting data for 2024-07-31: Open=420.4950, Symbol='MSFT', High=421.7750, Low=412.2100, Close=418.3500, Volume=42891366
Inserting data for 2024-07-30: Open=427.7200, Symbol='MSFT', High=429.0500, Low=417.3600, Close=422.9200, Volume=32687578
Inserting data for 2024-07-29: Open=431.5800, Symbol='MSFT', High=432.1500, Low=424.7042, Close=426.7300, Volume=15125836
Inserting data for 2024-07-26: Open=418.2000, Symbol='MSFT', High=428.9150, Low=417.2700, Close=425.2700, Volume=23583839
Inserting data for 2024-07-25: Open=428.8000, Symbol='MSFT', High=429.8000, Low=417.5100, Close=418.4000, Volume=29943795
Inserting data for 2024-07-24: Open=440.4500, Symbol='MSFT', High=441.4800, Low=427.5850, Close=428.9000, Volume=26805800
Inserting data for 2024-07-23: Open=443.8950, Symbol='MSFT', High=448.3900, Low=443.1000, Close=444.8500, Volume=13107050
Inserting data for 2024-07-22: Open=441.7900, Symbol='MSFT', High=444.6000, Low=438.9125, Close=442.9400, Volume=15808755
Inserting data for 2024-07-19: Open=433.1000, Symbol='MSFT', High=441.1400, Low=432.0000, Close=437.1100, Volume=20940417
Inserting data for 2024-07-18: Open=444.3400, Symbol='MSFT', High=444.6500, Low=434.4000, Close=440.3700, Volume=20794822
Inserting data for 2024-07-17: Open=442.5900, Symbol='MSFT', High=444.8500, Low=439.1800, Close=443.5200, Volume=21754021
Inserting data for 2024-07-16: Open=454.2200, Symbol='MSFT', High=454.3000, Low=446.6600, Close=449.5200, Volume=17175679
Inserting data for 2024-07-15: Open=453.3000, Symbol='MSFT', High=457.2600, Low=451.4300, Close=453.9600, Volume=14429447
Inserting data for 2024-07-12: Open=454.3250, Symbol='MSFT', High=456.3600, Low=450.6450, Close=453.5500, Volume=16324274
Inserting data for 2024-07-11: Open=462.9800, Symbol='MSFT', High=464.7800, Low=451.5500, Close=454.7000, Volume=23111175
Inserting data for 2024-07-10: Open=461.2200, Symbol='MSFT', High=466.4600, Low=458.8550, Close=466.2500, Volume=18196100
Inserting data for 2024-07-09: Open=467.0000, Symbol='MSFT', High=467.3300, Low=458.0000, Close=459.5400, Volume=17228507
Inserting data for 2024-07-08: Open=466.5500, Symbol='MSFT', High=467.7000, Low=464.4600, Close=466.2400, Volume=12962321
Inserting data for 2024-07-05: Open=459.6100, Symbol='MSFT', High=468.3500, Low=458.9650, Close=467.5600, Volume=16000290
```

Using try/except in below block

```
# This function returns a list of price data for the last 90 days
price_list = return_last_90d_price("MSFT")  # This should give you the structured data

def load(con, records):
    # Check if records is empty
    if not records:
        print("No records to load.")
        return  # Exit if there are no records

    # Define the target table for price data
    target_table = "demoAPI.raw_data.stock_price"

    try:
        # Start the transaction
        con.execute("BEGIN;")

        # Create or replace the table
        con.execute(f"""
        CREATE OR REPLACE TABLE {target_table} (
          date DATE Primary Key,
          symbol VARCHAR,
          open NUMBER,
          high NUMBER,
          low NUMBER,
          close NUMBER,
          volume NUMBER
        )
        """)

        # Load records into the table
        for r in records:
            date = r['date']
            open_price = r['open']
            high_price = r['high']
            low_price = r['low']
            close_price = r['close']
            volume = r['volume']

            print(f"Inserting data for {date}: Open={open_price}, High={high_price}, Low={low_price}, Close={close_price}, Volum

            # Use parameterized INSERT INTO to avoid SQL injection and special character issues
            sql = f"""
            INSERT INTO {target_table} (date, symbol, open, high, low, close, volume)
            VALUES (TO_DATE('{date}', 'YYYY-MM-DD'),'{symbol}', {open_price}, {high_price}, {low_price}, {close_price}, {volume}
            """
            con.execute(sql)

        # Commit the transaction
        con.execute("COMMIT;")

    except Exception as e:
        # Rollback the transaction in case of an error
        con.execute("ROLLBACK;")
        print("An error occurred while loading data:", e)
        raise e  # Re-raise the exception for further handling if needed

# Get the Snowflake connection
conn = return_snowflake_conn()

# Load the data into the Snowflake table
load(conn, price_list)
```

```
Inserting data for 2024-09-25: Open=429.8300, High=433.1190, Low=428.5700, Close=432.1100, Volume=13396364
Inserting data for 2024-09-24: Open=433.0000, High=433.3500, Low=426.1000, Close=429.1700, Volume=17015754
Inserting data for 2024-09-23: Open=434.2800, High=436.4600, Low=430.3889, Close=433.5100, Volume=15128891
Inserting data for 2024-09-20: Open=437.2200, High=439.2400, Low=434.2200, Close=435.2700, Volume=55167106
Inserting data for 2024-09-19: Open=441.2250, High=441.5000, Low=436.9000, Close=438.6900, Volume=21706559
Inserting data for 2024-09-18: Open=435.0000, High=436.0300, Low=430.4100, Close=430.8100, Volume=18898042
Inserting data for 2024-09-17: Open=440.2300, High=441.8500, Low=432.2700, Close=435.1500, Volume=18874231
Inserting data for 2024-09-16: Open=430.6000, High=433.5300, Low=428.2200, Close=431.3400, Volume=13834697
Inserting data for 2024-09-13: Open=425.8250, High=431.8300, Low=425.4600, Close=430.5900, Volume=15874555
Inserting data for 2024-09-12: Open=423.3100, High=427.3692, Low=419.7500, Close=427.0000, Volume=17418759
Inserting data for 2024-09-11: Open=415.5000, High=423.9900, Low=409.5800, Close=423.0400, Volume=19266923
Inserting data for 2024-09-10: Open=408.2000, High=416.3300, Low=407.7000, Close=414.2000, Volume=19594287
Inserting data for 2024-09-09: Open=407.2400, High=408.6500, Low=402.1500, Close=405.7200, Volume=15295134
Inserting data for 2024-09-06: Open=409.0600, High=410.6500, Low=400.8000, Close=401.7000, Volume=19609526
Inserting data for 2024-09-05: Open=407.6200, High=413.1000, Low=406.1300, Close=408.3900, Volume=14195516
Inserting data for 2024-09-04: Open=405.9100, High=411.2400, Low=404.3700, Close=408.9000, Volume=14886710
Inserting data for 2024-09-03: Open=417.9100, High=419.8800, Low=407.0300, Close=409.4400, Volume=20313603
Inserting data for 2024-08-30: Open=415.6000, High=417.4900, Low=412.1300, Close=417.1400, Volume=24308324
Inserting data for 2024-08-29: Open=414.9400, High=422.0500, Low=410.6000, Close=413.1200, Volume=17045241
Inserting data for 2024-08-28: Open=414.8800, High=415.0000, Low=407.3100, Close=410.6000, Volume=14882660
Inserting data for 2024-08-27: Open=412.8600, High=414.3600, Low=410.2500, Close=413.8400, Volume=13492911
Inserting data for 2024-08-26: Open=416.3650, High=417.2799, Low=411.3400, Close=413.4900, Volume=13152830
Inserting data for 2024-08-23: Open=416.9800, High=419.2600, Low=412.0900, Close=416.7900, Volume=18493784
Inserting data for 2024-08-22: Open=424.3600, High=426.7899, Low=414.6100, Close=415.5500, Volume=19361901
Inserting data for 2024-08-21: Open=424.0750, High=426.4000, Low=421.7200, Close=424.1400, Volume=16067298
Inserting data for 2024-08-20: Open=421.7000, High=425.8600, Low=421.6400, Close=424.8000, Volume=16387581
Inserting data for 2024-08-19: Open=418.9600, High=421.7500, Low=416.4600, Close=421.5300, Volume=15233957
Inserting data for 2024-08-16: Open=420.6000, High=421.3400, Low=417.3000, Close=418.4700, Volume=22775593
Inserting data for 2024-08-15: Open=419.8000, High=421.1100, Low=417.6600, Close=421.0300, Volume=20752144
Inserting data for 2024-08-14: Open=414.8000, High=417.7200, Low=412.4456, Close=416.8600, Volume=18266980
Inserting data for 2024-08-13: Open=409.5900, High=414.9500, Low=409.5700, Close=414.0100, Volume=19414271
Inserting data for 2024-08-12: Open=407.0600, High=408.7600, Low=404.2434, Close=406.8100, Volume=16762883
Inserting data for 2024-08-09: Open=404.0300, High=408.0500, Low=402.2624, Close=406.0200, Volume=19276666
Inserting data for 2024-08-08: Open=402.4400, High=405.8600, Low=399.9407, Close=402.6900, Volume=20203030
Inserting data for 2024-08-07: Open=408.6350, High=410.0800, Low=397.4700, Close=398.4300, Volume=20650906
Inserting data for 2024-08-06: Open=400.0000, High=405.6700, Low=398.5000, Close=399.6100, Volume=24946525
Inserting data for 2024-08-05: Open=389.1700, High=401.0400, Low=385.5800, Close=395.1500, Volume=40709238
Inserting data for 2024-08-02: Open=412.4900, High=415.0000, Low=404.3400, Close=408.4900, Volume=29437900
Inserting data for 2024-08-01: Open=420.7850, High=427.4600, Low=413.0901, Close=417.1100, Volume=30296400
Inserting data for 2024-07-31: Open=420.4950, High=421.7750, Low=412.2100, Close=418.3500, Volume=42891366
Inserting data for 2024-07-30: Open=427.7200, High=429.0500, Low=417.3600, Close=422.9200, Volume=32687578
Inserting data for 2024-07-29: Open=431.5800, High=432.1500, Low=424.7042, Close=426.7300, Volume=15125836
Inserting data for 2024-07-26: Open=418.2000, High=428.9150, Low=417.2700, Close=425.2700, Volume=23583839
Inserting data for 2024-07-25: Open=428.8000, High=429.8000, Low=417.5100, Close=418.4000, Volume=29943795
Inserting data for 2024-07-24: Open=440.4500, High=441.4800, Low=427.5850, Close=428.9000, Volume=26805800
Inserting data for 2024-07-23: Open=443.8950, High=448.3900, Low=443.1000, Close=444.8500, Volume=13107050
Inserting data for 2024-07-22: Open=441.7900, High=444.6000, Low=438.9125, Close=442.9400, Volume=15808755
Inserting data for 2024-07-19: Open=433.1000, High=441.1400, Low=432.0000, Close=437.1100, Volume=20940417
Inserting data for 2024-07-18: Open=444.3400, High=444.6500, Low=434.4000, Close=440.3700, Volume=20794822
Inserting data for 2024-07-17: Open=442.5900, High=444.8500, Low=439.1800, Close=443.5200, Volume=21754021
Inserting data for 2024-07-16: Open=454.2200, High=454.3000, Low=446.6600, Close=449.5200, Volume=17175679
Inserting data for 2024-07-15: Open=453.3000, High=457.2600, Low=451.4300, Close=453.9600, Volume=14429447
Inserting data for 2024-07-12: Open=454.3250, High=456.3600, Low=450.6450, Close=453.5500, Volume=16324274
Inserting data for 2024-07-11: Open=462.9800, High=464.7800, Low=451.5500, Close=454.7000, Volume=23111175
Inserting data for 2024-07-10: Open=461.2200, High=466.4600, Low=458.8550, Close=466.2500, Volume=18196100
Inserting data for 2024-07-09: Open=467.0000, High=467.3300, Low=458.0000, Close=459.5400, Volume=17228507
Inserting data for 2024-07-08: Open=466.5500, High=467.7000, Low=464.4600, Close=466.2400, Volume=12962321
Inserting data for 2024-07-05: Open=459.6100, High=468.3500, Low=458.9650, Close=467.5600, Volume=16000290
```

Ensuring Idempotency by running the load function twice:

```
def check_record_count(con):
    target_table = "demoAPI.raw_data.stock_price"
    result = con.execute(f"SELECT COUNT(*) FROM {target_table}").fetchone()
    return result[0]


# Get Snowflake connection
conn = return_snowflake_conn()

# Check initial record count
initial_count = check_record_count(conn)
print(f"Initial record count: {initial_count}")

# Load data twice
load(conn, price_list)
first_load_count = check_record_count(conn)
print(f"Record count after first load: {first_load_count}")

load(conn, price_list)
second_load_count = check_record_count(conn)
print(f"Record count after second load: {second_load_count}")

# Validate idempotency
if first_load_count == second_load_count:
    print("Idempotent: record counts are consistent.")
else:
    print("Not idempotent: record counts differ.")
```

```
Initial record count: 61
Inserting data for 2024-09-25: Open=429.8300, Symbol='MSFT', High=433.1190, Low=428.5700, Close=432.1100, Volume=13396364
Inserting data for 2024-09-24: Open=433.0000, Symbol='MSFT', High=433.3500, Low=426.1000, Close=429.1700, Volume=17015754
Inserting data for 2024-09-23: Open=434.2800, Symbol='MSFT', High=436.4600, Low=430.3889, Close=433.5100, Volume=15128891
Inserting data for 2024-09-20: Open=437.2200, Symbol='MSFT', High=439.2400, Low=434.2200, Close=435.2700, Volume=55167106
Inserting data for 2024-09-19: Open=441.2250, Symbol='MSFT', High=441.5000, Low=436.9000, Close=438.6900, Volume=21706559
Inserting data for 2024-09-18: Open=435.0000, Symbol='MSFT', High=436.0300, Low=430.4100, Close=430.8100, Volume=18898042
Inserting data for 2024-09-17: Open=440.2300, Symbol='MSFT', High=441.8500, Low=432.2700, Close=435.1500, Volume=18874231
Inserting data for 2024-09-16: Open=430.6000, Symbol='MSFT', High=433.5300, Low=428.2200, Close=431.3400, Volume=13834697
Inserting data for 2024-09-13: Open=425.8250, Symbol='MSFT', High=431.8300, Low=425.4600, Close=430.5900, Volume=15874555
Inserting data for 2024-09-12: Open=423.3100, Symbol='MSFT', High=427.3692, Low=419.7500, Close=427.0000, Volume=17418759
Inserting data for 2024-09-11: Open=415.5000, Symbol='MSFT', High=423.9900, Low=409.5800, Close=423.0400, Volume=19266923
Inserting data for 2024-09-10: Open=408.2000, Symbol='MSFT', High=416.3300, Low=407.7000, Close=414.2000, Volume=19594287
Inserting data for 2024-09-09: Open=407.2400, Symbol='MSFT', High=408.6500, Low=402.1500, Close=405.7200, Volume=15295134
Inserting data for 2024-09-06: Open=409.0600, Symbol='MSFT', High=410.6500, Low=400.8000, Close=401.7000, Volume=19609526
Inserting data for 2024-09-05: Open=407.6200, Symbol='MSFT', High=413.1000, Low=406.1300, Close=408.3900, Volume=14195516
Inserting data for 2024-09-04: Open=405.9100, Symbol='MSFT', High=411.2400, Low=404.3700, Close=408.9000, Volume=14886710
Inserting data for 2024-09-03: Open=417.9100, Symbol='MSFT', High=419.8800, Low=407.0300, Close=409.4400, Volume=20313603
Inserting data for 2024-08-30: Open=415.6000, Symbol='MSFT', High=417.4900, Low=412.1300, Close=417.1400, Volume=24308324
Inserting data for 2024-08-29: Open=414.9400, Symbol='MSFT', High=422.0500, Low=410.6000, Close=413.1200, Volume=17045241
Inserting data for 2024-08-28: Open=414.8800, Symbol='MSFT', High=415.0000, Low=407.3100, Close=410.6000, Volume=14882660
Inserting data for 2024-08-27: Open=412.8600, Symbol='MSFT', High=414.3600, Low=410.2500, Close=413.8400, Volume=13492911
Inserting data for 2024-08-26: Open=416.3650, Symbol='MSFT', High=417.2799, Low=411.3400, Close=413.4900, Volume=13152830
Inserting data for 2024-08-23: Open=416.9800, Symbol='MSFT', High=419.2600, Low=412.0900, Close=416.7900, Volume=18493784
Inserting data for 2024-08-22: Open=424.3600, Symbol='MSFT', High=426.7899, Low=414.6100, Close=415.5500, Volume=19361901
Inserting data for 2024-08-21: Open=424.0750, Symbol='MSFT', High=426.4000, Low=421.7200, Close=424.1400, Volume=16067298
Inserting data for 2024-08-20: Open=421.7000, Symbol='MSFT', High=425.8600, Low=421.6400, Close=424.8000, Volume=16387581
Inserting data for 2024-08-19: Open=418.9600, Symbol='MSFT', High=421.7500, Low=416.4600, Close=421.5300, Volume=15233957
```

Start coding or generate with AI.