# ============================================================================

# SNOWFLAKE DATAFRAME SHAPE MISMATCH FIX

# ============================================================================

# This code fixes the common pandas error: "Shape of passed values is (X, Y), indices imply (X, Z)"

# The issue occurs when the data being returned from Snowflake has a different

# number of columns than what we're trying to create in the DataFrame.

# First, let's print the result to understand its structure

# This helps us see what we're actually getting from Snowflake

print("Result type:", type(result))

print("Result content:", result)

# From screenshot: Result is a tuple with just 2 values (datetime.date and a number)

# but we're trying to create a DataFrame with 12 columns

# Define the columns we want in our final DataFrame

# These are the target column names that the final DataFrame should contain

columns = [

'PNO', 'PROCESSOR', 'LOAD\_TIMESTAMP', 'TRACKER\_ID',

'h\_system\_id', 'h\_pv\_function', 'h\_units\_of\_work', 'h\_client\_id',

'h\_batchid\_cardnet', 'h\_close\_term\_type', 'h\_delete\_flag',

'key\_block\_trans'

]

try:

# First approach: Try standard DataFrame creation

# This will work if 'result' has the right shape for our columns

data = pd.DataFrame([result], columns=columns)

except ValueError as e:

# This exception is raised when the data shape doesn't match the columns

# In this case, we get "Shape of passed values is (2, 1), indices imply (2, 12)"

print(f"Error: {str(e)}")

# -----------------------------------------------------------------------

# FIX FOR SHAPE MISMATCH WHEN RESULT IS A SINGLE TUPLE WITH FEWER VALUES

# -----------------------------------------------------------------------

import numpy as np

# Create a placeholder row with all values set to None/NaN

# This ensures we have the right number of columns (same length as 'columns')

row\_data = [None] \* len(columns)

# Fill in the available values from the result tuple

# This takes whatever values we do have and puts them in the first positions

# Example: If result is (date, number), then row\_data will have those

# values in positions 0 and 1, with None in positions 2-11

for i, val in enumerate(result):

if i < len(columns):

row\_data[i] = val

# Create a new DataFrame with properly structured data

# We're creating a 1-row DataFrame with the correct number of columns

# Any missing data will be represented as None/NaN

data = pd.DataFrame([row\_data], columns=columns)

print("Fixed DataFrame shape mismatch!")

# Print the resulting DataFrame to verify it looks correct

print(data.head())