Unloading Data

Sujith Nair

Cloud Data Architect

Snowflake Snowpro Certified

# What is data unloading ?

Data unloading is the process of moving data out of snowflake to cloud storage. We use the COPY command to unload data from snowflake.

Why is unloading of data needed ?

- Snowflake is the source of the data and is needed in other apps.
- Data has been computed inside of snowflake and is needed elsewhere.
- Business user wants data in XLS format.

#When unloading data into cloud storage what best practice do you use?

To ensure that the export process is completed quickly and with usage of least amount of credits, I would use the partition clause in the COPY command to ensure that parallelism feature of snowflake is used, and multiple files are generated based on the partition I want.

I would also name of the file to disallow the file to be named generically by snowflake.

```
COPY INTO @MYDB.MYSTAGES.UNLOAD_OUTPUT/CUST
FROM SNOWFLAKE_SAMPLE_DATA.TPCH_SF1.CUSTOMER
PARTITION BY C_MKTSEGMENT
```

BEST QUALITY
BEST QUALITY
BEST QUALITY

# Was there a scenario in your project where you had to unload the data to an internal stage ?

Request for data is frequently received from business users from snowflake. This could be for data analysis or dealing with data quality issues. The way I provide the data is to unload into an internal stage and use the GET command to download the data, I use the SINGLE=TRUE option to ensure that they file does not get split to multiple files.

# Do you prefer unloading data into multiple files or a single file from snowflake ?

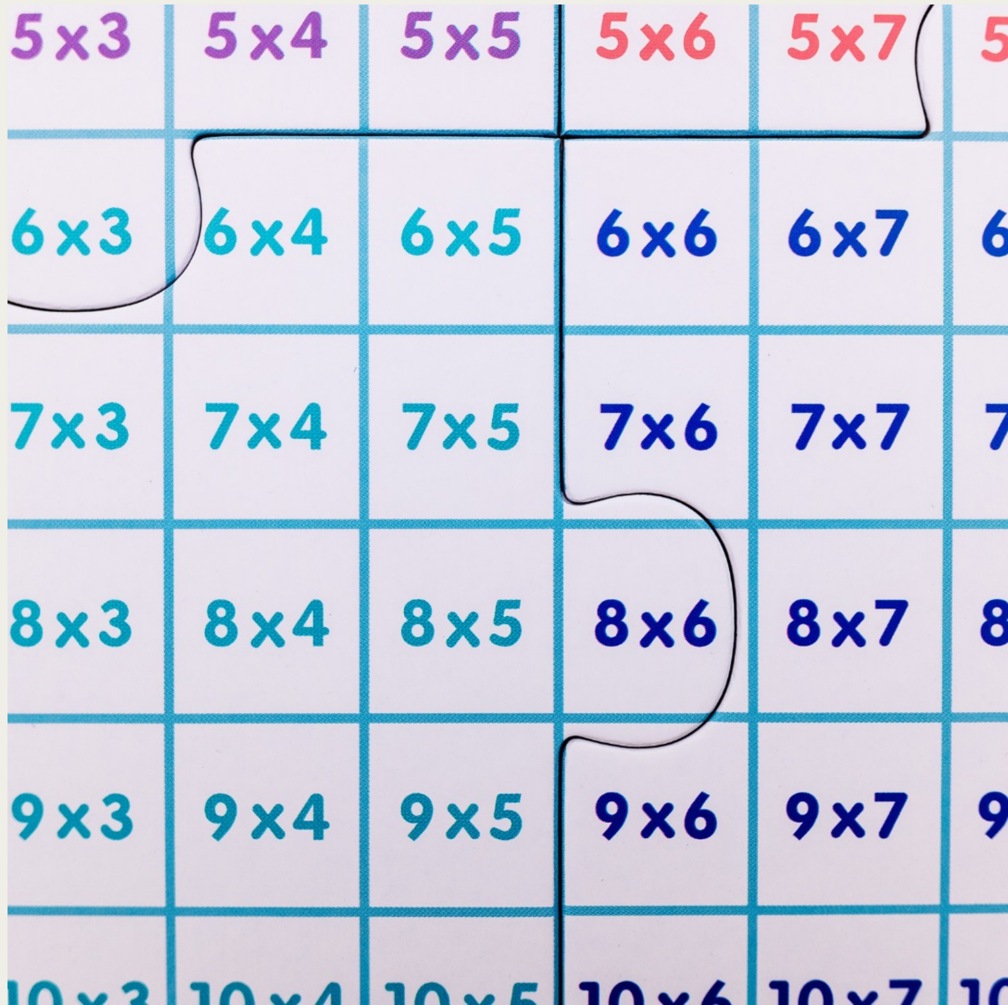
To take advantage of parallelism provided by snowflake and get the files faster I prefer multiple files being generated. This is the default format. There is also a 5GB file limitation on cloud storage and if your file size is bigger then you need to generate multiple files.

To get a single file we need to use the COPY option SINGLE=TRUE

# Can you unload data from multiple tables in a COPY command

COPY statement supports the full syntax of snowflake SQL and hence you can join the tables in the COPY command and get data from more than 1 table.

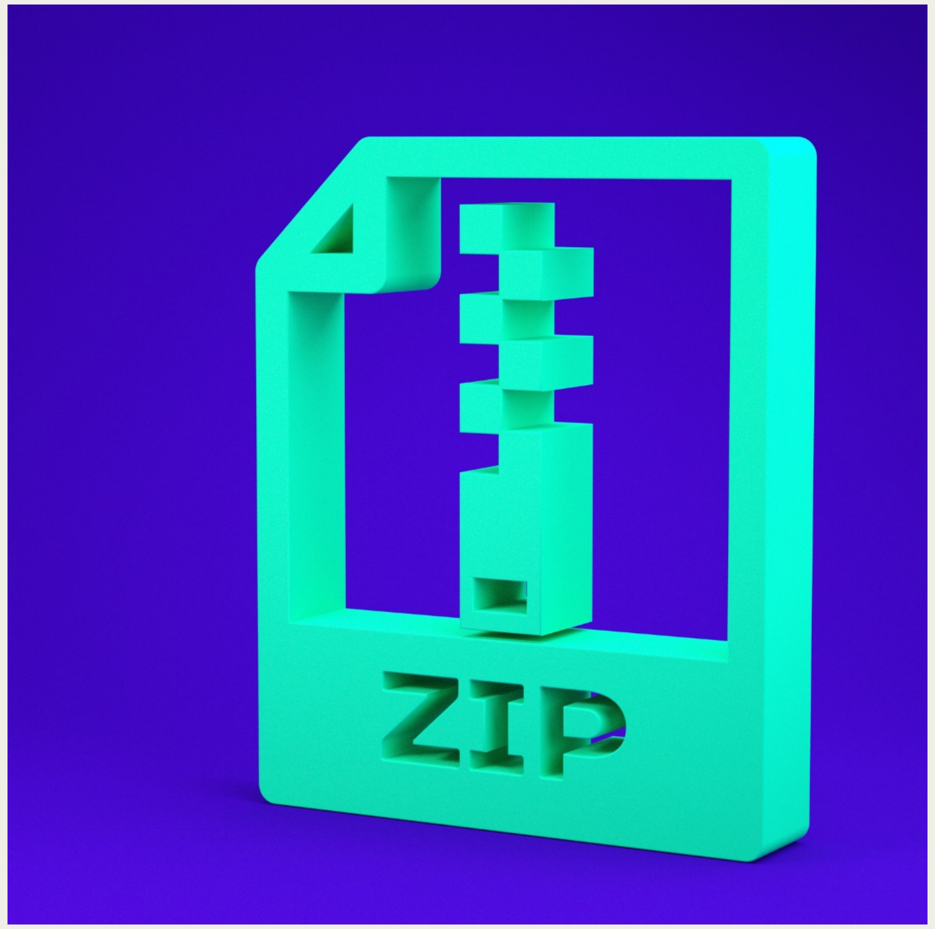
```
COPY INTO @MYDB.MYSTAGES.UNLOAD_OUTPUT FROM
(SELECT C.* FROM
SNOWFLAKE_SAMPLE_DATA.TPCH_SF1.CUSTOMER
C,SNOWFLAKE_SAMPLE_DATA.TPCH_SF1.NATION N
                                        WHERE
C.C_NATIONKEY=N.N_NATIONKEY
)
```



How are parquet files zipped when being unloaded?

The default compression when unloading data to snowflake in parquet format is snappy. We can also use LZO if we wish. However we need to explicitly provide the compression type if we don't want the default.

For CSV and JSON files the default compression type is GZIP.
The other supported compression types are bzip2, Brotli,Zstandard



# How do we modify column datatypes when unloading data?

When using the COPY command we can use the CAST function to modify data types when unloading data in parquet format.

```
    COPY INTO @UNLOAD_OUTPUT_PARQUET
    FROM (SELECT
     CAST(C_CUSTKEY AS STRING),
     CAST(C_NATIONKEY AS STRING)
    FROM
    SNOWFLAKE_SAMPLE_DATA.TPCH_SF1.CUSTOMER
          )
```

# What problems have you encountered when unloading data?

Load failures due to files already being present in the folder is a challenge we have encountered, we resolved that by creating a lambda function(or Azure functions) that copies the file to a different folder and adds the timestamp to the file.

What problems have you encountered when unloading data?

When unloading data in CSV format we observed that snowflake truncates data in decimal columns to 15,9.

We overcame this problem by casting the data to string and unloading the data.

# How can you limit the size of a file generated by unloading data from snowflake.

We need to use the MAX_FILE_SIZE parameter and limit the size of the files. Files generated are generally around 16 MB in size , you can make the files smaller by using the MAX_FILE_SIZE parameter.

```
COPY INTO @MYDB.MYSTAGES.UNLOAD_OUTPUT
FROM SNOWFLAKE_SAMPLE_DATA.TPCH_SF1.CUSTOMER
MAX_FILE_SIZE=10000000
```

Why have smaller files instead of 1 big file ?

- Faster processing
- Able to take advantage of parallel processing
- Easier to consume by other applications which may have size limits.

Thank you!

Learn2CloudData Solutions