Snowflake Bulk Load Project

Why use it?
You need to load thousands or millions of staged files into a table with efficient partitioning.

What does it do?
It loads every file you specify in the order you set, making best use of micro-partitioning on the table.

How does it work?
After configuration and small scale testing, you schedule tasks running in as many warehouses as you want. All of them call the same stored procedure with control information in a table. The procedure's design expects that several other warehouses will be running the same task in parallel.

There is no practical limit on the number of warehouses you can have running the stored procedure. There are no external dependencies and everything runs inside of Snowflake. The design allows you to start, stop, and restart gracefully. You can even abort a running stored procedure or kill its session with only minor cleanup required (auto-cleanup will follow) by running a single SQL statement.

What are the steps to run it?

1. Import five worksheets into the Snowflake UI or your preferred SQL interface. The project splits the worksheets into five sections for logical flow. If you want, you can combine them into one.
2. The project is hardwired to work on a table in the SNOWFLAKE_SAMPLE_DATABASE. Your account may have a different name for this database. You can modify the scripts slightly as required. If you do not have a sample database at all, you can import it from a share.
3. Run through the project to see how it works with sample data. When ready, modify the scripts to work with your database, stage, and target table.
4. The most important thing to remember is to modify is the body of the *`GetCopyTemplate()`* function. This is where you will put your `COPY INTO` statement for the stored procedure to pick up and use. Make sure your copy into statement is robust and resilient to handle files with formatting errors, Unicode issues, etc. Setting `on_error=continue` is highly recommended.
5. Run the stored procedure unit test. This sets a file limit of 16 files in batches of 8 files at a time and a time limit of one minute. Note that the time limit sets the last time a COPY INTO statement will start. The stored procedure will complete the final copy beyond this time limit, but not run another one after reaching the time limit.
6. Examine the control table, target table, load history, and run relevant SQL in the performance monitoring worksheet. If everything looks good, create multiple warehouses (sized XS – M as advised by a SQL query in the project) and run the same stored procedure unit test, each in separate warehouses. Examine results and start increasing the file and time limits.
7. Once ramp up looks good, you should have the FILES_TO_PROCESS set to the full number of files and FILES_AT_ONCE (used in each COPY INTO) set between 512 and 1000. You'll typically set the max run time to 20-60 minutes per run. Now create a number of events to run the stored procedure, each in its own warehouse. This will ensure that the stored procedure runs without having its session killed by closing or disconnecting a client. Set the timer on the events to one minute. They will wait until their stored procedures complete to restart the timer. Monitor to completion or when the table contains enough data, and then disable the events.