Week 1 Exercises

List of all assignments for first week together at one place. Please find all DDL and all other needed queries for the exercises in following file on GitHub: `Week1/week1_exercises.sql`

03 Data Ingestion workflow

Let's establish all needed objects and integrations needed for the data ingestion into Snowflake. Please use your trial AWS account, where we are going to use S3 as our external stage. During this exercise we are going to create following objects:

- Storage integration to link Snowflake and AWS together in secure way
- Stage object as file location description
- File Format with format definition for our source files

Let's start with creation of S3 bucket in your AWS account. You can use whatever name. My bucket has following name: `oreilly-trainings.`

Now we need to configure the secure access to cloud Storage. It requires multiple configuration steps on AWS side, like creation of IAM policy and role, defining the access control requirements, etc.

Snowflake provide step by step guide for this setup. It is available here:
https://docs.snowflake.com/en/user-guide/data-load-s3-config-storage-integration

Once we have configured the integration between AWS and Snowflake we can proceed with FILE FORMAT creation. File format should have following parameters:

- Csv file as an type
- Comma as an delimiter
- It should skip header
- fields are enclosed by double quotes (")

Next, we need to create a stage object:

- It will use our storage integration
- It will use our file format
- It will use our s3 bucket

Finally we have all the objects in place so we can proceed with target table creation.

Please create a table called `GPT_TWEETS`. DDL script is available on Github in following file:

`Week1/week1_exercises.sql`

Once we have landing table create, let's upload the `chatgpt-tweets-data.csv` into our S3 bucket and we are ready to go with data import. You can download the file from the GitHub repository. It is under following path: `week1/source_files/chatgpt-tweets-data.csv.zip`
Once you downloaded the file, please uncompress it before uploading into S3.

Let's write a copy command to import the data. As source file contains some errors we need to use `ON_ERROR = CONTINUE` option.

As another example, let's try to do some transformations as part of COPY command. We are going to select only a few columns from source file and also do simple casting for timestamp values. Please create a new landing table called `GPT_TWEETS_PROCCESSED`.

Now we can write a new COPY command command which will be selecting only following source columns:

```
ID, DATE, USERNAME, PROCESSED_TWEET
```

- Let's still use `ON_ERROR = CONTINUE` option
- Let's do a casting to TIMESTAMP for DATE value

As a last part of this exercise, let's try to explore a load metadata. Let's start with querying the metadata directly from the staged file along with some columns. We can get following values during loading data into the table:

- `METADATA$FILENAME,`
- `METADATA$FILE_ROW_NUMBER,`
- `METADATA$FILE_CONTENT_KEY,`
- `METADATA$FILE_LAST_MODIFIED,`
- `METADATA$START_SCAN_TIME`

Let's write a select query which will select those attributes along with first 3 columns from the staged file.

As a next example, let's query the load metadata for given landing table from `COPY_HISTORY` table function in `INFORMATION_SCHEMA`. Firstly let's try it for the `GPT_TWEETS_PROCCESSED` table with START_DATE in last hour.

As a last example let's try the same thing but this time for `GPT_TWEETS` table. We should also get an info about error rows.

04 Semi structured data processing

We are going to practice working with semi structured data in this exercise along with using functions INFER_SCHEMA, GENERATE_COLUMN_DESCRIPTION and CREATE TABLE USING TEMPLATE which significantly help to speed up the data ingestion from Parquet or AVRO files.

1. Please download the parquet data from GITHUB. There are two files:
   - `Week1/source_files/gpt_tweets.parwuet_0_0_0.snappy.parquet`
   - `Week1/source_files/gpt_tweets.parwuet_0_0_1.snappy.parquet`

2. Upload the parquet files into your S3 bucket
3. We need to create a landing table for our data. Let's call it GPT_TWEETS_PARQUET
   - You can find all the DDLs and DMLs for this exercise in following file on GitHub: `week1/week1_exercises.sql`

4. For the INFER_SCHEMA function we will need a named file format for the parquet files. Please create `my_parquet_format.`

5. Now it is time to manually ingest data from Parquet format. We need to write a COPY command and specify the data type for each column. Please notice that all the columns are under pseudo column `$1` in the file. You need to use following syntax for each column: `$1:column_name::data_type`

6. Now let's try to automate it and avoid writing the list of attributes manually, same like creating the landing table manually. First, let's try to use `GENERATE_COLUMN_DESCRIPTION` function which will generate a list of columns together with the right data type. It will be based on source parquet file which is used as parameter.

7. Now we can copy the output and paste it into `CREATE` table statement or we can automate it further and create also table automatically by using `CREATE TABLE USING TEMPLATE` function. Let's create table called `GPT_TWEETS_PARQUET_TEMPLATE` by using this TEMPLATE feature.

8. Now we have another landing table in place so let's ingest the data there again. This time we are not going to manually write the body of the `COPY` statement and define each column together with the attribute but we are going to use `INFER_SCHEMA` function which will generate it for us. Then we can copy the output and paste into the `COPY` command.

05 Intro into SnowSQL - CLI interface

Let's start with installation of the CLI client. Please go into https://developers.snowflake.com/snowsql/
This page also contains link into documentation with detailed guide for installation or configuration of the CLI client.

Based on your OS and preferred way of installation download the needed files. Once you have it installed, there should be a configuration file under user home directory: `~/.snowsql/`

Let's open the config file and configure the connection to our Snowflake account and check what are other options which might be configured.

Please find all the commands in `week1/week1_exercises.sql`

06 Views

This part is dedicated to practising working with views and their differences. We will try to create standard view, secure view and materializied view. Thanks to this exercise we will be able to describe their limitations. Please follow the code available in `week1/week1_exercises.sql`