**Load data from local storage to a database via Snowsql**

### **Bulk loading from a local file system using COPY**

1. Open a command line window.
2. Check SnowSQL installation by writing a command snowsql -v
3. Then login to snowsql: snowsql -a <account\_identifier> -u <user\_name>

## Create Snowflake objects:

During this step you create the following Snowflake objects:

* A database (sf\_tuts) and a table (emp\_basic). You load sample data into this table.
* A [virtual warehouse](https://docs.snowflake.com/en/user-guide/warehouses-overview) (sf\_tuts\_wh). This warehouse provides the compute resources needed to load data into the table and query the table. For this tutorial, you create an X-Small warehouse.

1. Create a database: CREATE OR REPLACE DATABASE sf\_tuts;

In this tutorial, you use the default schema (public) available for each database, rather than creating a new schema.

Note that the database and schema you just created are now in use for your current session, as reflected in the SnowSQL command prompt. You can also use the context functions to get this information. Run below cmd:

SELECT CURRENT\_DATABASE(), CURRENT\_SCHEMA();

1. **Create a table:**

**CREATE OR REPLACE TABLE emp\_basic (**

**first\_name STRING ,**

**last\_name STRING ,**

**email STRING ,**

**streetaddress STRING ,**

**city STRING ,**

**start\_date DATE**

**);**

1. Create a virtual warehouse

CREATE OR REPLACE WAREHOUSE sf\_tuts\_wh WITH

WAREHOUSE\_SIZE='X-SMALL'

AUTO\_SUSPEND = 180

AUTO\_RESUME = TRUE

INITIALLY\_SUSPENDED=TRUE;

The sf\_tuts\_wh warehouse is initially suspended, but the DML statement also sets AUTO\_RESUME = true. The AUTO\_RESUME setting causes a warehouse to automatically start when SQL statements that require compute resources are executed.

After you create the warehouse, it’s now in use for your current session. This information is displayed in your SnowSQL command prompt. You can also retrieve the name of the warehouse by using the following context function:

1. SELECT CURRENT\_WAREHOUSE();
2. Stage data files

A Snowflake stage is a location in cloud storage that you use to load and unload data from a table. Snowflake supports the following types of stages:

* **Internal stages**—Used to store data files internally within Snowflake. Each user and table in Snowflake gets an internal stage by default for staging data files.
* **External stages**—Used to store data files externally in Amazon S3, Google Cloud Storage, or Microsoft Azure. If your data is already stored in these cloud storage services, you can use an external stage to load data in Snowflake tables.

Download sample data files

For this tutorial you download sample employee data files in CSV format that Snowflake provides.

To download and unzip the sample data files:

1. Download the set of sample data files. Right-click the name of the archive file, [getting-started.zip](https://docs.snowflake.com/en/_downloads/34f4a66f56d00340f8f7a92acaccd977/getting-started.zip), and save the link/file to your local file system.
2. Unzip the sample files. unpack files into one of the following directories:

**Linux/macOS:** /tmp

**Windows: C:**\\temp

1. Linux or macOS users, type command in the terminal of Snowsql:

PUT file:///tmp/employees0\*.csv @sf\_tuts.public.%emp\_basic;

Windows users, type command in the terminal of Snowsql:

Let’s take a closer look at the command:

* file://<file-path>[/]employees0\*.csv specifies the full directory path and names of the files on your local machine to stage. Note that file system wildcards are allowed, and if multiple files fit the pattern they are all displayed.
* **@<namespace>.%<table\_name>** indicates to use the stage for the specified table, in this case the emp\_basic table.

1. Listing the staged files by using below command:

LIST @sf\_tuts.public.%emp\_basic;

1. Copy data into target tables: To load your staged data into the target table, execute COPY INTO <table>.

The COPY INTO <table> command uses the virtual warehouse you created.

COPY INTO emp\_basic

FROM @%emp\_basic

FILE\_FORMAT = (type = csv field\_optionally\_enclosed\_by='"')

PATTERN = '.\*employees0[1-5].csv.gz'

ON\_ERROR = 'skip\_file';

Where:

* The FROM clause specifies the location containing the data files (the internal stage for the table).
* The FILE\_FORMAT clause specifies the file type as CSV, and specifies the double-quote character (") as the character used to enclose strings. Snowflake supports diverse file types and options. These are described in [CREATE FILE FORMAT](https://docs.snowflake.com/en/sql-reference/sql/create-file-format).
* The PATTERN clause specifies that the command should load data from the filenames matching this regular expression (.\*employees0[1-5].csv.gz).
* The ON\_ERROR clause specifies what to do when the COPY command encounters errors in the files. By default, the command stops loading data when the first error is encountered. This example skips any file containing an error and moves on to loading the next file. (None of the files in this tutorial contain errors; this is included for illustration purposes.)

+--------------------+--------+-------------+-------------+-------------+-------------+-------------+------------------+-----------------------+-------------------------+

| file | status | rows\_parsed | rows\_loaded | error\_limit | errors\_seen | first\_error | first\_error\_line | first\_error\_character | first\_error\_column\_name |

|--------------------+--------+-------------+-------------+-------------+-------------+-------------+------------------+-----------------------+-------------------------|

| employees02.csv.gz | LOADED | 5 | 5 | 1 | 0 | NULL | NULL | NULL | NULL |

| employees04.csv.gz | LOADED | 5 | 5 | 1 | 0 | NULL | NULL | NULL | NULL |

| employees05.csv.gz | LOADED | 5 | 5 | 1 | 0 | NULL | NULL | NULL | NULL |

| employees03.csv.gz | LOADED | 5 | 5 | 1 | 0 | NULL | NULL | NULL | NULL |

| employees01.csv.gz | LOADED | 5 | 5 | 1 | 0 | NULL | NULL | NULL | NULL |

+--------------------+--------+-------------+-------------+-------------+-------------+-------------+------------------+-----------------------+-------------------------+

1. Query loaded data

You can query the data loaded in the emp\_basic table using standard [SQL](https://docs.snowflake.com/en/sql-reference/constructs) and any supported [functions](https://docs.snowflake.com/en/sql-reference-functions) and [operators](https://docs.snowflake.com/en/sql-reference/operators). You can also manipulate the data, such as updating the loaded data or inserting more data, using standard [DML commands](https://docs.snowflake.com/en/sql-reference/sql-dml).

### Retrieve all data

Return all rows and columns from the table: SELECT \* FROM emp\_basic;

### Insert additional data rows

INSERT INTO emp\_basic VALUES

('Clementine','Adamou','cadamou@sf\_tuts.com','10510 Sachs Road','Klenak','2017-9-22') ,

('Marlowe','De Anesy','madamouc@sf\_tuts.co.uk','36768 Northfield Plaza','Fangshan','2017-1-26');

### Query rows based on email address

SELECT email FROM emp\_basic WHERE email LIKE '%.uk';

### Query rows based on start date

SELECT first\_name, last\_name, DATEADD('day',90,start\_date) FROM emp\_basic WHERE start\_date <= '2017-01-01';

1. Cleanup

DROP DATABASE IF EXISTS sf\_tuts;

DROP WAREHOUSE IF EXISTS sf\_tuts\_wh;

1. Exit the connection

To exit a connection, use the !exit command for SnowSQL (or its alias, !disconnect).

Exit drops the current connection and quits SnowSQL if it is the last connection.

Happy Cloud computing

Regards

Saransh Jain