Overview of data loading

This topic provides an overview of the main options available to load data into Snowflake.

Supported file locations

Snowflake refers to the location of data files in cloud storage as a *stage*. The COPY INTO <table> command used for both bulk and continuous data loads (Snowpipe) supports cloud storage accounts managed by your business entity (*external stages*) as well as cloud storage contained in your Snowflake account (*internal stages*).

External stages

Loading data from any of the following cloud storage services is supported regardless of the cloud platform that hosts your Snowflake account:

- Amazon S3
- Google Cloud Storage
- Microsoft Azure

You cannot access data held in archival cloud storage classes that requires restoration before it can be retrieved. These archival storage classes include, for example, the Amazon S3 Glacier Flexible Retrieval or Glacier Deep Archive storage class, or Microsoft Azure Archive Storage.

Upload (i.e. *stage*) files to your cloud storage account using the tools provided by the cloud storage service.

A named external stage is a database object created in a schema. This object stores the URL to files in cloud storage, the settings used to access the cloud storage account, and convenience settings such as the options that describe the format of staged files. Create stages using the CREATE STAGE command.

information, see Understanding data transfer cost.

Internal stages

Snowflake maintains the following stage types in your account:

User	A user stage is allocated to each user for storing files. This stage type is designed to store files that are staged and managed by a single user but can be loaded into multiple tables. User stages cannot be altered or dropped.
Table	A table stage is available for each table created in Snowflake. This stage type is designed to store files that are staged and managed by one or more users but only loaded into a single table. Table stages cannot be altered or dropped. Note that a table stage is not a separate database object; rather, it is an implicit stage tied to the table itself. A table stage has no grantable privileges of its own. To stage files to a table stage, list the files, query them on the stage, or drop them, you must be the table owner (have the role with the OWNERSHIP privilege on the table).
Named	A named internal stage is a database object created in a schema. This stage type can store files that are staged and managed by one or more users and loaded into one or more tables. Because named stages are database objects, the ability to create, modify, use, or drop them can be controlled using security access control privileges. Create stages using the CREATE STAGE command.

Upload files to any of the internal stage types from your local file system using the PUT command.

Bulk vs continuous loading

Snowflake provides the following main solutions for data loading. The best solution may depend upon the volume of data to load and the frequency of loading.

Bulk loading using the COPY command

This option enables loading batches of data from files already available in cloud storage, or copying

Compute resources

Bulk loading relies on user-provided virtual warehouses, which are specified in the COPY statement. Users are required to size the warehouse appropriately to accommodate expected loads.

Simple transformations during a load

Snowflake supports transforming data while loading it into a table using the COPY command. Options include:

- Column reordering
- Column omission
- Casts
- Truncating text strings that exceed the target column length

There is no requirement for your data files to have the same number and ordering of columns as your target table.

Continuous loading using Snowpipe

This option is designed to load small volumes of data (i.e. micro-batches) and incrementally make them available for analysis. Snowpipe loads data within minutes after files are added to a stage and submitted for ingestion. This ensures users have the latest results, as soon as the raw data is available.

Compute resources

Snowpipe uses compute resources provided by Snowflake (i.e. a serverless compute model). These Snowflake-provided resources are automatically resized and scaled up or down as required, and are charged and itemized using per-second billing. Data ingestion is charged based upon the actual workloads.

Simple transformations during a load

The COPY statement in a pipe definition supports the same COPY transformation options as when bulk loading data.

In addition, data pipelines can leverage Snowpipe to continuously load micro-batches of data into staging tables for transformation and optimization using automated tasks and the change data capture (CDC) information in streams.

Continuous loading using Snowpipe Streaming

The Snowpipe Streaming API writes rows of data directly to Snowflake tables without the requirement of staging files. This architecture results in lower load latencies with corresponding lower costs for loading any volume of data, which makes it a powerful tool for handling near real-time data streams.

Snowpipe Streaming is also available for the Snowflake Connector for Kafka, which offers an easy upgrade path to take advantage of the lower latency and lower cost loads.

For more information, refer to Snowpipe Streaming.

Loading data from Apache Kafka topics

The Snowflake Connector for Kafka enables users to connect to an Apache Kafka server, read data from one or more topics, and load that data into Snowflake tables.

Schema detection of column definitions from staged semi-structured data files

Semi-structured data can include thousands of columns. Snowflake provides robust solutions for handling this data. Options include referencing the data directly in cloud storage using external tables, loading the data into a single column of type VARIANT, or transforming and loading the data

A different solution involves automatically detecting the schema in a set of staged semi-structured data files and retrieving the column definitions. The column definitions include the names, data types, and ordering of columns in the files. Generate syntax in a format suitable for creating Snowflake standard tables, external tables, or views.

Note
This feature supports Apache Parquet, Apache Avro, ORC, JSON, and CSV files.

This support is implemented through the following SQL functions:

INFER_SCHEMA

Detects the column definitions in a set of staged data files and retrieves the metadata in a format suitable for creating Snowflake objects.

GENERATE_COLUMN_DESCRIPTION

Generates a list of columns from a set of staged files using the INFER_SCHEMA function output.

These SQL functions support both internal and external stages.

Create tables or external tables with the column definitions derived from a set of staged files using the CREATE TABLE … USING TEMPLATE or CREATE EXTERNAL TABLE … USING TEMPLATE syntax. The USING TEMPLATE clause accepts an expression that calls the INFER_SCHEMA SQL function to detect the column definitions in the files. After the table is created, you can then use a COPY statement with the `MATCH_BY_COLUMN_NAME` option to load files directly into the structured table.

Note
Creating an Iceberg table with the USING TEMPLATE clause (and column definitions derived from INFER_SCHEMA output) isn't supported.

Schema detection can also be used in conjunction with table schema evolution, where the structure of tables evolves automatically to support the structure of new data received from the data sources.

You can use the following option to query your data in cloud storage without loading it into Snowflake tables.

External tables (data lake)

External tables enable querying existing data stored in external cloud storage for analysis without first loading it into Snowflake. The source of truth for the data remains in the external cloud storage. Data sets materialized in Snowflake via materialized views are read-only.

This solution is especially beneficial to accounts that have a large amount of data stored in external cloud storage and only want to query a portion of the data; for example, the most recent data. Users can create materialized views on subsets of this data for improved query performance.

Working with Amazon S3-compatible storage

You can create external stages and tables in Snowflake to access storage in an application or device that is Amazon S3-compatible. This feature lets you manage, govern, and analyze your data, regardless of where the data is stored. For information, see Working with Amazon S3-compatible storage.