Creating a Task

In Snowflake, tasks are used to schedule and automate the execution of SQL statements or Python scripts at specified intervals or on a defined schedule. Tasks can run as part of a workflow or in isolation, based on your scheduling requirements.

In the following we’re going to go through creating a task in Snowflake, explaining the required parameters, and giving examples to help you understand how to set up and manage tasks.

**1. What is a Task in Snowflake?**

A task in Snowflake is a background process that runs a SQL statement or Python script on a scheduled basis. This can be used run a SQL query and update a table stored in Snowflake.

**2. Required Parameters for Creating a Task**

To create a task in Snowflake, the basic structure includes the following parameters:

1. **Task Name** (TASK\_NAME):
   * This is a unique identifier for the task.
   * Example: REFRESH\_MATERIALIZED\_VIEW
2. **Warehouse** (WAREHOUSE):
   * This specifies the **virtual warehouse** that will be used to execute the task. If a warehouse is not specified, the task will run without a warehouse.
   * Example: WAREHOUSE = GITBA\_BRONZE
3. **Schedule** (SCHEDULE):
   * The SCHEDULE parameter defines how often the task should run. This uses the cron syntax or the IMMEDIATELY option for one-time tasks.
   * Example: 'USING CRON '0 0 \* \* 1' — to run every Monday at midnight.
   * The schedule parameter is optional, and if not provided, the task can be triggered manually or using a different event.
4. **SQL Statement** (SQL\_STATEMENT):
   * The SQL query or set of queries that the task will execute when triggered.
   * Example: REFRESH MATERIALIZED VIEW my\_view;

Template with Only Required Elements:

CREATE OR REPLACE TASK <database>.<schema>.<task\_name>

WAREHOUSE=<warehouse>

SCHEDULE=‘USING CRON <minute> <hour> <day-of-month> <month> <day-of-week> <time\_zone>’

AS CREATE OR REPLACE TABLE <table\_name> AS <SQL\_statement>

Example:

CREATE OR REPLACE TASK GSC\_ANALYTICS\_DM.GSCA.PLANT\_MASTER\_TASK

WAREHOUSE=GITBA\_BRONZE

SCHEDULE='USING CRON 0 7 \* \* \* America/Los\_Angeles'

AS CREATE OR REPLACE TABLE PLANT\_MASTER AS

**3. Optional Parameters**

1. **Comment** (COMMENT):
   * Optionally, you can add a comment to describe the task's purpose or behavior.
   * Example: COMMENT = 'This task refreshes materialized views weekly'

Specifying a Task Name and Warehouse

Scheduling a Task

Scheduling combined with USING CRON allows us to schedule a task to execute at a desired regular interval. If you’re familiar with Linux, you might be familiar with cron expressions. The fields in a cron expression are similar in Snowflake, but there are a few key differences to keep in mind. Below is an outline of how to schedule with USING CRON.

SCHEDULE='USING CRON <minute> <hour> <day\_of\_month> <month> <day\_of\_week> <time\_zone>'

Let’s break down each field below.

**1. Minute (<minute>)**

* **Range**: 0-59
* This field specifies the minute of the hour when the task will run. A value of 0 means the task will run at the start of the hour.

Examples:

* 0: Run at the start of the hour.
* 15: Run at the 15th minute of the hour.
* \*/15: Run every 15 minutes.

**2. Hour (<hour>)**

* **Range**: 0-23 (24-hour format)
* This field specifies the hour of the day when the task will run. A value of 0 means midnight, and a value of 23 means the last hour of the day.

Examples:

* 0: Run at midnight.
* 12: Run at noon.
* 17: Run at 5 PM.
* \*/2: Run every 2 hours (i.e., 0, 2, 4, 6, ..., 22).

**3. Day of Month (<day\_of\_month>)**

* **Range**: 1-31
* This field specifies the day of the month when the task will run. A value of 1 refers to the first day of the month, and 31 refers to the 31st day, if applicable.

Examples:

* 1: Run on the first day of the month.
* 15: Run on the 15th day of the month.
* \*/5: Run every 5 days (1, 6, 11, 16, 21, 26).

**4. Month (<month>)**

* **Range**: 1-12 (1 = January, 12 = December)
* This field specifies the month when the task will run. A value of 1 refers to January, 12 refers to December, etc.

Examples:

* 1: Run in January.
* 6: Run in June.
* \*/3: Run every 3 months (January, April, July, October).

**5. Day of Week (<day\_of\_week>)**

1. **Range**: 0-6 (0 = Sunday, 6 = Saturday)
2. This field specifies the day of the week when the task will run. A value of 0 refers to Sunday, and 6 refers to Saturday.

Examples:

* 0: Run on Sunday.
* 1: Run on Monday.
* 5: Run on Friday.
* \*/2: Run every 2 days of the week (Sunday, Tuesday, Thursday, Saturday).

**6. Time Zone (<time\_zone>)**

* Time Zone does not accept a range but instead accepts a TZ identifier from the Time Zone Database.

[List of tz database time zones - Wikipedia](https://en.wikipedia.org/wiki/List_of_tz_database_time_zones)

Examples:

* America/Los\_Angeles (Pacific Time)
* Europe/Budapest (CET)
* Europe/Berlin (CET)
* Asia/Singapore (SGT)
* UTC

**Special Characters in Cron Expressions**

Multiple special characters are used in cron expressions in Snowflake. Here’s how they work:

**1. Asterisk (\*)**: This means "every". When placed in a field, it tells Snowflake to run the task every possible value for that field.

Example:

* + \* \* \* \* \*: This expression will run every minute of every hour, day, month, and day of the week (essentially, it will run every minute).

**2. Slash (/)**: This is used to specify a step value, such as "every Nth value". It can be used in any of the fields, and it defines intervals. For example, \*/15 in the minute field would mean "run every 15 minutes".

Example:

* + \*/10 \* \* \* \*: This will run every 10 minutes.

**3. Comma (,)**: This allows you to specify multiple values in a field. When a comma is used, the task will run at the specified values.

Example:

* + 0 12,18 \* \* \*: This will run at 12:00 PM and 6:00 PM every day.

**4. Hyphen (-)**: This is used to define a range of values for a field.

Example:

* + 0 9-17 \* \* \*: This will run every hour from 9 AM to 5 PM (inclusive) every day.

**Useful Examples of Cron Expressions**

1. **Every hour on the hour in Pacific time**:

SCHEDULE='USING CRON 0 \* \* \* \* America/Los\_Angeles'

1. **Every weekday at 7 AM and 4 PM Pacific time**:

SCHEDULE='USING CRON 0 7,16 \* \* 1-5 America/Los\_Angeles'

1. **Every midnight on the last day of every month Pacific time**:

SCHEDULE='USING CRON 0 0 L \* \* America/Los\_Angeles'

1. **Every Friday at 5 PM Pacific time**:

SCHEDULE='USING CRON 0 17 \* \* 5 America/Los\_Angeles'

1. **Close of business on the last day of every 3rd month Pacific time (i.e. end of quarter)**:

SCHEDULE='USING CRON 0 17 L 3,6,9,12 \* America/Los\_Angeles'

Altering a Task

After you create a task, you will likely need to go back at some point and change something about the task, whether it’s updating the refresh schedule, the SQL statement or renaming the task. Below is an outline of some of the key ways you can alter a task.

1. **ALTER TASK <task> RESUME:** Resumes a task after it has been created or after it has been suspended.A note that when tasks are created, they are automatically suspended. After you create a task you’ll need to resume it, otherwise it will never run

Example:

* + ALTER TASK GSC\_ANALYTICS\_DM.GSCA.PLANT\_MASTER\_TASK RESUME;

1. **ALTER TASK <task> SUSPEND:** Prevent a task from running. This is important to do during maintenance or before updating a task

Example:

* + ALTER TASK GSC\_ANALYTICS\_DM.GSCA.PLANT\_MASTER\_TASK SUSPEND;

1. **ALTER TASK <task> SET SCHEDULE = <schedule>:** Changes the run schedule. Used if maybe you need to run it more or less frequently

Example:

* + ALTER TASK GSC\_ANALYTICS\_DM.GSCA.PLANT\_MASTER\_TASK SET SCHEDULE = 'USING CRON 0 \* \* \* \* America/Los\_Angeles';

1. **ALTER TASK <task> RENAME TO <task>:** Rename a task. Update naming to reflect new function/purpose

Example:

* + ALTER TASK PLANT\_MASTER RENAME TO PLANT\_MST\_TASK;

1. **ALTER TASK <task> MODIFY AS <sql\_statement>:** Modify a body SQL statement or Python script. A note that you must suspend a task before using MODIFY AS

Example:

ALTER TASK GSC\_ANALYTICS\_DM.GSCA.PLANT\_MASTER\_TASK SUSPEND;

ALTER TASK GSC\_ANALYTICS\_DM.GSCA.PLANT\_MASTER\_TASK

MODIFY AS

SELECT \*

FROM GSC\_ANALYTICS\_DM.GSCA."Plant\_Master"

LIMIT 15;

ALTER TASK GSC\_ANALYTICS\_DM.GSCA.PLANT\_MASTER\_TASK RESUME;

Dropping a Task

Just like tables or views, tasks can be dropped to remove them when they are no longer needed. Dropping is irreversible — you can’t recover a task after it’s dropped unless you recreate it.

**1. DROP TASK <task>:** Straightforward command that removes the task from your schema

Example:

* + DROP TASK GSC\_ANALYTICS\_DM.GSCA.PLANT\_MASTER\_TASK;

**2. DROP TASK IF EXISTS <task>:** Used for the same purpose as DROP TASK <task>, but prevents an error if the task doesn't exist by adding IF EXISTS.

Example:

* + DROP TASK IF EXISTS GSC\_ANALYTICS\_DM.GSCA.PLANT\_MASTER\_TASK;

**Using TIMESTAMP\_INPUT\_FORMAT**

In Snowflake, the TIMESTAMP\_INPUT\_FORMAT is used to specify the format in which timestamp data is input or parsed. This is particularly useful when loading data from external sources (e.g., CSV files, JSON) that may have date and time representations in different formats.

The format string 'YYYY-MM-DD HH24' defines how Snowflake should interpret the timestamp input when loading or parsing data. Let's break down the components of this format and show you how to adjust it for different date-time representations.

**1. Basic Format: 'YYYY-MM-DD HH24'**

The format 'YYYY-MM-DD HH24' represents the following:

* **YYYY**: Four-digit year (e.g., 2025)
* **MM**: Two-digit month (e.g., 03 for March)
* **DD**: Two-digit day of the month (e.g., 21 for the 21st day)
* **HH24**: Hour in 24-hour format (e.g., 14 for 2:00 PM)

Example:

* '2025-03-21 14' will be interpreted as **March 21st, 2025 at 14:00 (2:00 PM)**.

**2. Adding Minutes: 'YYYY-MM-DD HH24:MI'**

If your input data also includes minutes, you can modify the format to include the minute component. You can use 'MI' for minutes.

Example format:

TIMESTAMP\_INPUT\_FORMAT='YYYY-MM-DD HH24:MI'

Example input: '2025-03-21 14:30'

* This represents **March 21st, 2025 at 14:30 (2:30 PM)**.