Contents

[**Snowflake Assessment Project** 2](#_Toc152103801)

[**Data Preparation** 2](#_Toc152103802)

[**Data loading** 2](#_Toc152103803)

[**Data Architecture** 3](#_Toc152103804)

[**Dashboards Created** 4](#_Toc152103805)

# **Snowflake Assessment Project**

This document explains Data preparation, Data loading and Data Architecture steps involved in the Snowflake assessment project.

Project Git Repo : <https://github.com/smvinodkumar910/snowflake_learning.git>

## **Data Preparation**

1. We are getting Weather data from Rest API: <https://rapidapi.com/weatherapi/api/weatherapi-com/>
2. Created account with Rapidapi.com and subscribed weather data provided by weatherapi.com. Got the Api keys to access Api data.
3. Written the python code to access Rest Api to get the weather data. <https://github.com/smvinodkumar910/snowflake_learning/blob/main/source_generate/api_to_csv.py>

Above code does the following operation:

* Gets Weather data from Rest Api and convert that into 3 csv files as below:

LOCATION\_DTL\_YYYYMMDDHHMISS.csv

WEATHER\_DTL\_ YYYYMMDDHHMISS.csv

CURRENT\_CONDITION\_DTL\_ YYYYMMDDHHMISS.csv

* Organize the files in folder structure like **YYYY/MM/DD/HH/File.csv** and Uploads the files to AWS S3 Bucket staging area.

## **Data loading**

1. In Snowflake **STORATE INTEGRATION** object created to access the staging area in S3 bucket.

<https://github.com/smvinodkumar910/snowflake_learning/blob/main/snowflake_sql/aws_weather_project/integration/AWS_S3_INTEG_VM.sql>

1. From AWS console provided required permissions to the AWS IAM Service account for snowflake account, as mentioned in the document –

<https://docs.snowflake.com/en/user-guide/data-load-s3-config-storage-integration>

1. Created 3 SNOWPIPE objects for 3 files with **auto\_ingest=true** option.

<https://github.com/smvinodkumar910/snowflake_learning/tree/main/snowflake_sql/aws_weather_project/pipe>

1. The auto\_ingest=true parameter specifies to read event notifications sent from an S3 bucket to an SQS (Simple Queue Service) queue when new data is ready to load.
2. Using the SHOW PIPES command get the SQS ARN details, using which configure AWS S3 with event notification.
3. Once the above configuration is done, SNOWPIPE object will be notified whenever new file gets uploaded to the staging path in the S3 bucket.
4. Created DATABASE and SCHEMA in snowflake as required. Create 3 RAW tables and 3 DM tables for each file generated from API.
5. Created 3 Procedures to load data from STG table to DM table, and a PLP Procedure to load the final Reporting Fact Table. <https://github.com/smvinodkumar910/snowflake_learning/tree/main/snowflake_sql/aws_weather_project/stored_procedures>
6. Created 3 APPEND-ONLY STREAM objects on RAW tables. <https://github.com/smvinodkumar910/snowflake_learning/tree/main/snowflake_sql/aws_weather_project/streams>
7. Created 4 TASKS to EXECUTE the procedures. Used the **WHEN** option with **SYSTEM$STREAM\_HAS\_DATA** to configure the TASK to run only if there are new records inserted into the RAW layer. Scheduled one ROOT task and setup all other tasks as dependent of that ROOT TASK using the **AFTER** option. <https://github.com/smvinodkumar910/snowflake_learning/tree/main/snowflake_sql/aws_weather_project/tasks>
8. Run the ALTER TASK RESUME command to enable all the tasks.
9. Now, whenever files uploaded to the S3 Bucket, SNOWPIP gets notified and loads data to RAW Layer. ROOT TASKS runs the procedure as the STREAM has data, and subsequently, other tasks also get triggered.
10. In the SIL/PLP Procedures we are using the STREAM in the DML statement. Hence the STREAM will be offset to the current time when the DML statement succeeds.

## 

## **Data Architecture**

We are loading 4 DM tables –

1. DM\_LOCATION\_DTL

This dimension table stores weather location details like CITY, COUNTRY, LATITUDE, LONGITUDE, TIMEZONE etc. CITY is the primary key.

1. DM\_WEATHER\_DTL

This Fact table stores weather metric information like temperature, cloud levels, humidity details for each city in DM\_LOCATION\_DTL table along with LAST\_UPDATED time which represents last weather updated time for the city. This table having composite primary key consists of (CITY, LAST\_UPDATED) (Minute Level)

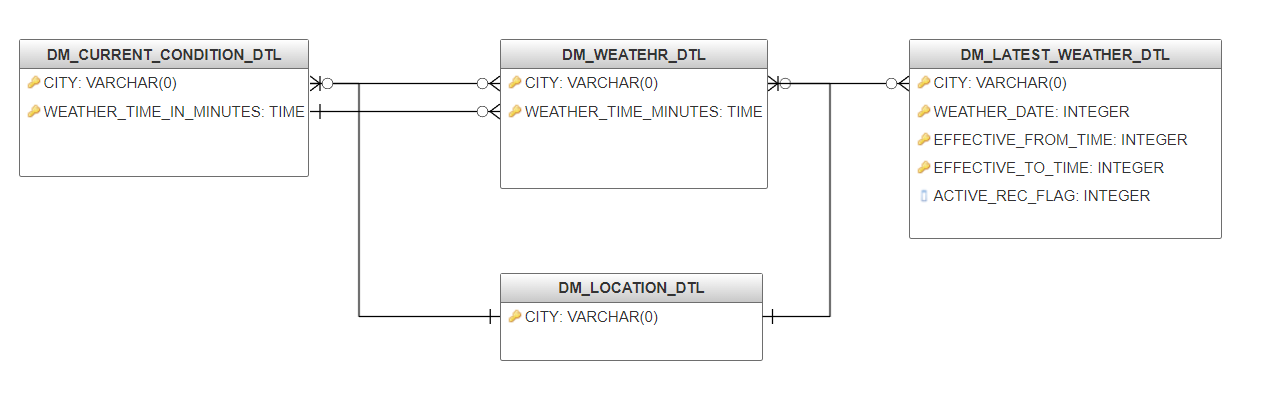
1. DM\_CURRENT\_CONDITION\_DTL

This Dimension table stores Weather descriptive details and related image representation.

i.e. Mist, Expected rain. This table also having same composite primary key (CITY, LAST\_UPDATED) (Minute Level)

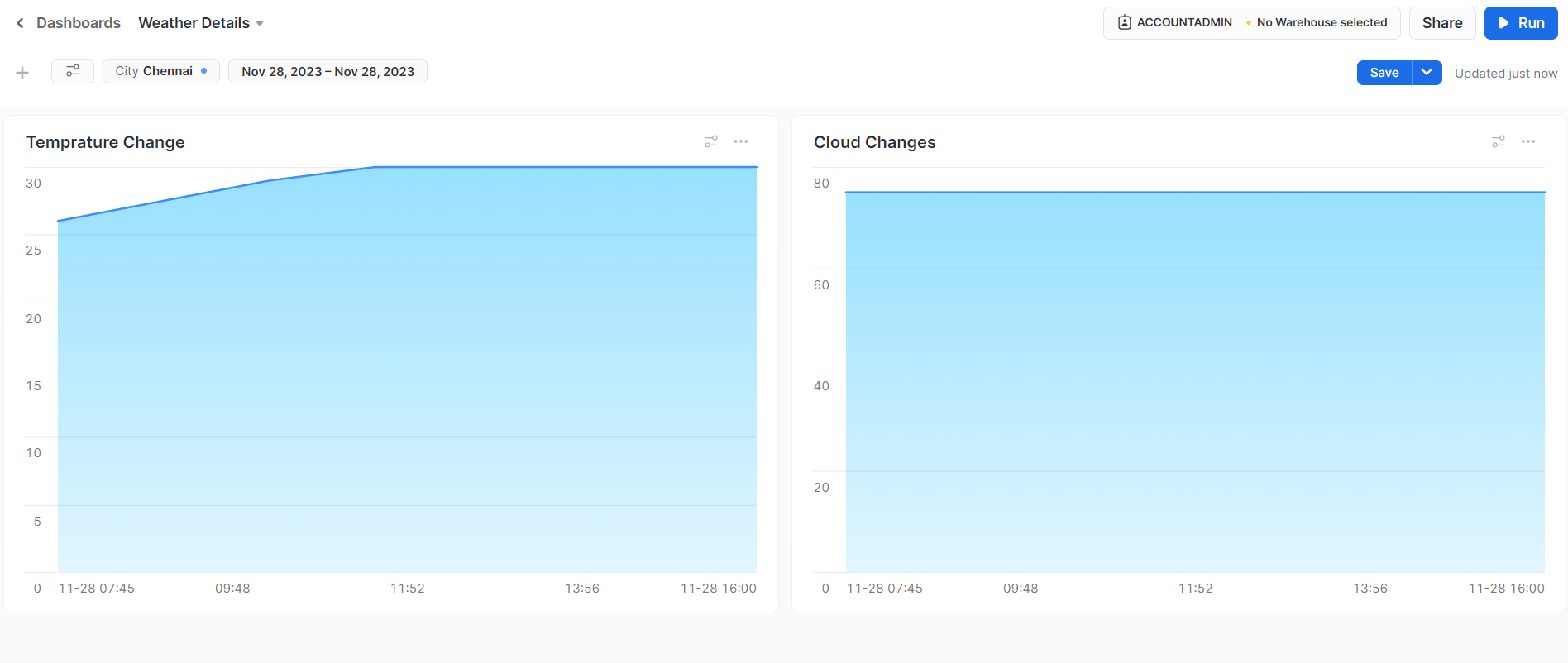
1. DM\_LATEST\_WEATHER\_DTL (SCD TYPE 2)

This is like a DATAMART table, loaded with a subset of data from DM\_WEATHER\_DTL table and DM\_CURRENT\_CONDITION\_DTL table. SCD2 implemented in this table with EFFECTIVE\_FROM, EFFECTIVE\_TO , ACTIVE\_REC\_FLAG columns to identify weather changes in minute level, also to get the latest weather for the day.



## **Dashboards Created**

1. Temparature/Cloud changes for the selected City on a selected day



1. City with Latest Temperature recorded:

