**Climate Change Data Analysis Project**

**Climate Change Data Analysis Project**

**Table of Contents**

1. [Introduction](https://d.docs.live.net/579f54d031e0ba34/Documents/Github/Projects/Snowflake/Carbon%20emissions%20and%20Energy%20consumption/Carbon%20Emissions%20and%20Energy%20Consumption%20by%20Country.docx)
2. [Project Objectives](https://chatgpt.com/c/674592e2-2528-800c-a1eb-5781da0561a8#project-objectives)
3. [Data Sources](https://chatgpt.com/c/674592e2-2528-800c-a1eb-5781da0561a8#data-sources)
4. [Data Warehouse Architecture](https://chatgpt.com/c/674592e2-2528-800c-a1eb-5781da0561a8#data-warehouse-architecture)
5. [Data Integration (ETL Process)](https://chatgpt.com/c/674592e2-2528-800c-a1eb-5781da0561a8#data-integration-etl-process)
6. [Data Modeling](https://chatgpt.com/c/674592e2-2528-800c-a1eb-5781da0561a8#data-modeling)
7. [Data Analysis](https://chatgpt.com/c/674592e2-2528-800c-a1eb-5781da0561a8#data-analysis)
8. [Data Visualization](https://chatgpt.com/c/674592e2-2528-800c-a1eb-5781da0561a8#data-visualization)
9. [Conclusion](https://chatgpt.com/c/674592e2-2528-800c-a1eb-5781da0561a8#conclusion)

**1. Introduction**

Climate change is a global crisis impacting weather patterns, sea levels, and ecosystems. Analyzing data from diverse sources helps us understand these impacts and devise strategies for mitigation. This project outlines a comprehensive solution using Snowflake’s data warehouse and visualization tools to gain actionable insights.

**2. Project Objectives**

* Gather and integrate data related to climate change.
* Design a Snowflake data warehouse to store and analyze the data.
* Identify trends and correlations in climate change metrics.
* Visualize the results using BI tools for better decision-making.

**3. Data Sources**

To analyze climate change effectively, data will be gathered from the following sources:

| **Source** | **Data Collected** |
| --- | --- |
| **Weather Stations** | Temperature, precipitation, wind speed |
| **Satellites** | Sea level changes, ice melting rates |
| **Research Institutions** | CO2 emissions, deforestation rates |

**4. Data Warehouse Architecture**

The Snowflake data warehouse architecture consists of three layers:

**4.1. Storage Layer**

* Stores raw, semi-structured, and structured data in a compressed format.
* Ensures scalability and cost-efficiency.

**4.2. Compute Layer**

* Uses Massively Parallel Processing (MPP) clusters for fast query execution.
* Supports multiple concurrent users.

**4.3. Cloud Services Layer**

* Manages metadata, query optimization, and security.
* Enables seamless integration with other tools and platforms.

**5. Data Integration (ETL Process)**

The ETL process ensures seamless integration of data from various sources into Snowflake.

**5.1. Extract**

* Gather raw data from weather stations, satellite feeds, and research institution databases.

**5.2. Transform**

* Clean and validate data to ensure consistency.
* Transform data into a uniform structure (e.g., aligning temperature units, formatting date fields).

**5.3. Load**

* Load transformed data into Snowflake’s storage layer using Snowpipe or the COPY command.

**6. Data Modeling**

Data is organized using a dimensional model with fact and dimension tables.

**6.1. Fact Tables**

* **Fact\_ClimateMetrics:**
  + Stores quantitative data like temperature readings, CO2 levels, and sea level changes.

| **Column** | **Description** |
| --- | --- |
| Fact\_ID | Unique identifier for each record |
| Temperature | Average temperature (°C) |
| CO2\_Emissions | CO2 emissions (Metric Tons) |
| Sea\_Level | Sea level change (mm) |
| Time\_ID (FK) | Links to time dimension |
| Location\_ID (FK) | Links to location dimension |

**6.2. Dimension Tables**

* **Dim\_Location:** Contains descriptive data about locations (e.g., country, region).
* **Dim\_Time:** Contains time-related data (e.g., year, month).

**7. Data Analysis**

SQL queries will help identify trends and correlations in the data.

**7.1. Trend Analysis**

Identify long-term trends in global temperatures:

SELECT year, AVG(temperature) AS avg\_temp

FROM Fact\_ClimateMetrics

JOIN Dim\_Time ON Fact\_ClimateMetrics.Time\_ID = Dim\_Time.Time\_ID

GROUP BY year

ORDER BY year;

**7.2. Correlation Analysis**

Analyze the relationship between CO2 emissions and temperature changes:

SELECT CO2\_Emissions, AVG(temperature) AS avg\_temp

FROM Fact\_ClimateMetrics

GROUP BY CO2\_Emissions

ORDER BY CO2\_Emissions;

**8. Data Visualization**

We’ll use Power BI to create interactive dashboards and charts.

**8.1. Dashboards**

* **Global Trends:** A dashboard showing trends in global temperature, CO2 emissions, and sea level rise.

**8.2. Charts**

* **Line Chart:** Display temperature trends over time.
* **Bar Chart:** Compare CO2 emissions across countries.

**9. Conclusion**

By building a Snowflake data warehouse and visualizing the data, I’ve created a scalable solution for climate change analysis. The insights gained from this project can help researchers and policymakers develop effective mitigation strategies.

**Absolutely! Let's apply the structured approach with multiple layers and implement Slowly Changing Dimension (SCD) Type 2 with merge upsert logic for the climate change data problem using Snowflake.**

Step-by-Step Implementation

1. Data Layers in Snowflake

1.1 Integration Layer

This layer contains raw data ingested from various sources.

1.2 Staging Layer

This layer stores intermediate data after initial transformations and cleaning.

1.3 Dimensional Layer

This layer holds dimension and fact tables designed for analytical queries. This is where SCD Type 2 logic is implemented.

1.4 Dimensional Key Map Layer

This layer maintains mappings between source keys and dimension keys, facilitating SCD management and ETL processes.

1.5 Base Layer

This layer includes aggregated or derived data used for reporting and analysis.

2. Setting Up Layers

2.1 Integration Layer

Create tables to store raw data from sources like weather stations, satellites, and research institutions.

CREATE TABLE integration.weather\_raw (

station\_id STRING,

date DATE,

temperature FLOAT,

precipitation FLOAT,

wind\_speed FLOAT

);

CREATE TABLE integration.satellite\_raw (

satellite\_id STRING,

date DATE,

sea\_level FLOAT,

ice\_melt\_rate FLOAT

);

CREATE TABLE integration.emissions\_raw (

country\_id STRING,

date DATE,

co2\_emission FLOAT,

deforestation\_rate FLOAT

);

2.2 Staging Layer

Create tables for initial transformations and cleaning.

CREATE TABLE staging.weather\_stg AS

SELECT \* FROM integration.weather\_raw;

CREATE TABLE staging.satellite\_stg AS

SELECT \* FROM integration.satellite\_raw;

CREATE TABLE staging.emissions\_stg AS

SELECT \* FROM integration.emissions\_raw;

2.3 Dimensional Layer

Create tables for dimensions and facts. Implement SCD Type 2 for the weather station dimension.

CREATE TABLE dimensional.weather\_fact (

station\_id STRING,

date DATE,

temperature FLOAT,

precipitation FLOAT,

wind\_speed FLOAT,

load\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE dimensional.weather\_station\_dim (

station\_sk NUMBER AUTOINCREMENT,

station\_id STRING,

location STRING,

effective\_start\_date DATE,

effective\_end\_date DATE,

current\_flag BOOLEAN DEFAULT TRUE,

load\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE dimensional.satellite\_fact (

satellite\_id STRING,

date DATE,

sea\_level FLOAT,

ice\_melt\_rate FLOAT,

load\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE dimensional.emissions\_fact (

country\_id STRING,

date DATE,

co2\_emission FLOAT,

deforestation\_rate FLOAT,

load\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

2.4 Dimensional Key Map Layer

Create a mapping table for dimension keys.

CREATE TABLE dimensional\_key\_map.station\_key\_map (

station\_id STRING,

station\_sk NUMBER,

load\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

2.5 Base Layer

Create tables for aggregated data.

CREATE TABLE base.temperature\_trend AS

SELECT

date,

AVG(temperature) AS avg\_temperature

FROM dimensional.weather\_fact

GROUP BY date;

CREATE TABLE base.co2\_emission\_summary AS

SELECT

country\_id,

COUNT(\*) AS total\_emissions,

SUM(co2\_emission) AS total\_co2,

MIN(date) AS first\_record\_date,

MAX(date) AS last\_record\_date

FROM dimensional.emissions\_fact

GROUP BY country\_id;

3. Implementing SCD Type 2 and Delta Load

3.1 SCD Type 2 with Merge Upsert Logic

Use Snowflake's MERGE statement to implement SCD Type 2 for the weather station dimension.

MERGE INTO dimensional.weather\_station\_dim AS target

USING staging.weather\_stg AS source

ON target.station\_id = source.station\_id

WHEN MATCHED AND target.current\_flag = TRUE AND (

target.location <> source.location

)

THEN

UPDATE SET

target.current\_flag = FALSE,

target.effective\_end\_date = CURRENT\_DATE

WHEN NOT MATCHED THEN

INSERT (

station\_id,

location,

effective\_start\_date,

effective\_end\_date,

current\_flag,

load\_date

)

VALUES (

source.station\_id,

source.location,

CURRENT\_DATE,

NULL,

TRUE,

CURRENT\_TIMESTAMP

);

3.2 Delta Load for Fact Tables

Use the MERGE statement to handle delta loads for the fact tables.

MERGE INTO dimensional.weather\_fact AS target

USING staging.weather\_stg AS source

ON target.station\_id = source.station\_id AND target.date = source.date

WHEN MATCHED THEN

UPDATE SET

target.temperature = source.temperature,

target.precipitation = source.precipitation,

target.wind\_speed = source.wind\_speed,

target.load\_date = CURRENT\_TIMESTAMP

WHEN NOT MATCHED THEN

INSERT (

station\_id,

date,

temperature,

precipitation,

wind\_speed,

load\_date

)

VALUES (

source.station\_id,

source.date,

source.temperature,

source.precipitation,

source.wind\_speed,

CURRENT\_TIMESTAMP

);

MERGE INTO dimensional.satellite\_fact AS target

USING staging.satellite\_stg AS source

ON target.satellite\_id = source.satellite\_id AND target.date = source.date

WHEN MATCHED THEN

UPDATE SET

target.sea\_level = source.sea\_level,

target.ice\_melt\_rate = source.ice\_melt\_rate,

target.load\_date = CURRENT\_TIMESTAMP

WHEN NOT MATCHED THEN

INSERT (

satellite\_id,

date,

sea\_level,

ice\_melt\_rate,

load\_date

)

VALUES (

source.satellite\_id,

source.date,

source.sea\_level,

source.ice\_melt\_rate,

CURRENT\_TIMESTAMP

);

MERGE INTO dimensional.emissions\_fact AS target

USING staging.emissions\_stg AS source

ON target.country\_id = source.country\_id AND target.date = source.date

WHEN MATCHED THEN

UPDATE SET

target.co2\_emission = source.co2\_emission,

target.deforestation\_rate = source.deforestation\_rate,

target.load\_date = CURRENT\_TIMESTAMP

WHEN NOT MATCHED THEN

INSERT (

country\_id,

date,

co2\_emission,

deforestation\_rate,

load\_date

)

VALUES (

source.country\_id,

source.date,

source.co2\_emission,

source.deforestation\_rate,

CURRENT\_TIMESTAMP

);

4. Visualization Using BI Tools

Connect your BI tool (e.g., Power BI) to Snowflake and create visualizations:

- Dashboards: Show trends in temperature, CO2 emissions, and sea level changes.

- Reports: Detailed reports on climate indicators and their changes over time.

Summary

By structuring the data warehouse into multiple layers and implementing SCD Type 2 with merge upsert logic, we ensure robust, scalable, and efficient data processing. This solution provides a comprehensive approach to analyzing climate change data and generating actionable insights through BI tools.

Feel free to tweak the steps and code to fit your specific requirements. If you have any questions or need further assistance, feel free to ask!