

Weather Data Analytics Pipeline Using Snowflake, Airflow, dbt, and Tableau

Deeksha Chauhan, Gayatri Patil, Nikhil Swami, Shweta Shinde

Department Of Applied Data Science, San Jose State University

San Jose, California, USA 95192 - 0250

Abstract:

This report discusses how to implement an end-to-end data analytics pipeline using the theme of weather data analytics. Such a pipeline will use Snowflake as the data warehouse, Apache Airflow for orchestration, dbt for data transformation, and Tableau for visualization. The project shows that all these tools can come together to build a robust analytics solution that is scalable, works on historical weather data, and generates insights into various weather metrics and analyses.

Keywords—

ETL- Extract, Transform, Load
ELT- Extract, Load, Transform
DAG - Directed Acyclic Graph
DBT- Data Build Tool

Problem Statement:

In today's data-driven world, analyzing weather patterns and their impact on various sectors is crucial. However, processing and analyzing vast amounts of weather data presents several challenges:

1. **Data Integration:** Weather data often comes from multiple sources and in various formats, making consolidation difficult.
2. **Scalability:** As historical weather data accumulates, traditional processing methods struggle to keep up.
3. **Data Transformation:** Raw weather data requires significant transformation to derive meaningful metrics and insights.
4. **Automation:** Manual data processing is time-consuming and error-prone, leading to delays in analysis.
5. **Visualization:** Presenting complex weather data in an easily understandable format is challenging but essential for quick interpretation.
6. **Maintainability:** Updating and maintaining data pipelines as requirements evolve can be complex.
7. **Real-time Insights:** There's a growing need for near-real-time weather data processing to support timely decision-making.

To address these challenges, we need an end-to-end data analytics solution that can efficiently ingest, process, transform, and visualize weather data. This project aims to develop such a solution using Snowflake, Apache Airflow, dbt, and Tableau.

I. Introduction

Weather data is crucial for various sectors, including agriculture, energy, transportation, and urban planning. This project aims to create an efficient, scalable data pipeline that can ingest, process, and visualize weather data, providing valuable insights for decision-making.

Objectives:

1. Implement an ETL process to populate raw weather data tables in Snowflake using Airflow
1. Develop an ELT process using dbt to create abstract tables with calculated weather metrics
2. Visualize key weather analytics using Tableau
3. Integrate all components into a cohesive, automated pipeline

The project utilizes Snowflake for data storage, Apache Airflow for orchestration, dbt for data transformation, and Tableau for visualization, leveraging each tool's strengths to create a comprehensive analytics solution.

II. System Architecture

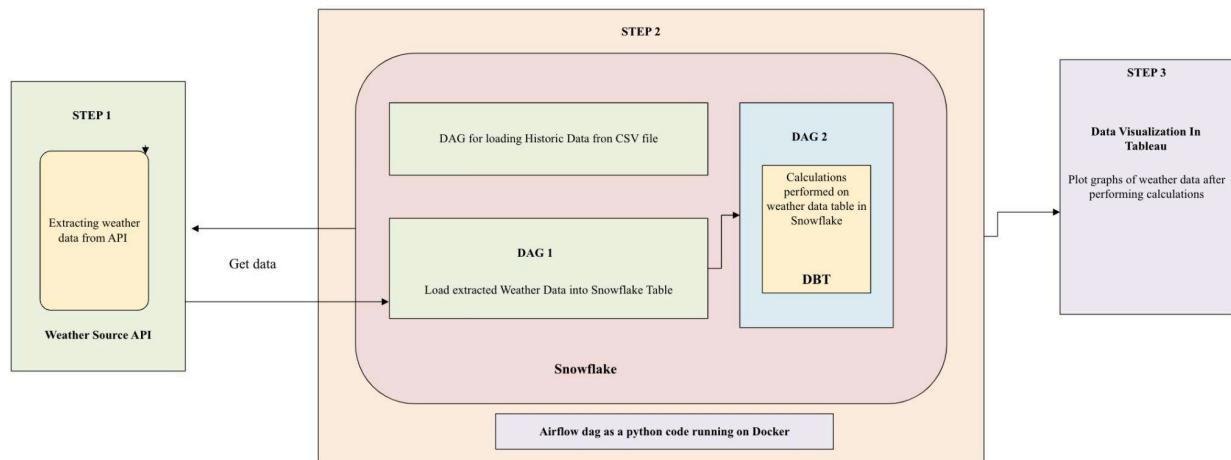


Fig. 1: High Level system diagram

The system architecture consists of four main components:

1. *Snowflake:* Acts as the central data warehouse, storing both raw and transformed data.
2. *Apache Airflow:* Orchestrates the entire pipeline, scheduling and executing both ETL and ELT processes.
3. *dbt:* Performs data transformations within Snowflake, creating abstract tables with calculated metrics.
4. *Tableau:* Connects to Snowflake to visualize the transformed data, providing interactive dashboards for analysis.

Data flows through the system as follows:

1. Raw stock data is ingested into Snowflake using Airflow-scheduled ETL processes.

2. dbt models, triggered by Airflow, transform the raw data into abstract tables within Snowflake.
3. Tableau connects to Snowflake, reading from the abstract tables to create visualizations.

III. ETL Process

The ETL process for our weather data analytics pipeline consists of two main tasks:

- Historical Data Loading
- Recent Data Fetching

Historical Data Loading

We use the `csv_historical_data_etl.py` script to load historical weather data from January 1, 2020, to October 31, 2024, from a CSV file. This process ensures we have a substantial baseline of historical data for analysis.

```

@task
def create_load_incremental(records):
    staging_table = "weather.source.daily_weather_stage"
    target_table = "weather.source.daily_weather"
    conn = return_snowflake_conn()
    try:
        conn.execute(f"""
            CREATE OR REPLACE TABLE {target_table} (
                datetime DATE,
                tempmax FLOAT,
                tempmin FLOAT,
                temp FLOAT,
                feelslike FLOAT,
                humidity FLOAT,
                precip FLOAT,
                windgust FLOAT,
                windspeed FLOAT,
                pressure FLOAT,
                cloudcover FLOAT,
                solarradiation FLOAT,
                severerisk INT,
                sunrise TIME,
                sunset TIME,
                conditions VARCHAR,
                PRIMARY KEY (datetime)
            );
        """)
        ## Create or replace the staging table
        conn.execute(f"""
            CREATE OR REPLACE TABLE {staging_table} (
                datetime DATE,
                tempmax FLOAT,
                tempmin FLOAT,
                temp FLOAT,
                feelslike FLOAT,
                humidity FLOAT,
                precip FLOAT,
                windgust FLOAT,
                windspeed FLOAT,
                pressure FLOAT,
                cloudcover FLOAT,
                solarradiation FLOAT,
                severerisk INT,
                sunrise TIME,
                sunset TIME,
                conditions VARCHAR,
                PRIMARY KEY (datetime)
            );
        """)
    
```

Fig. 2a: Python code to load historical data from csv file

This script reads the CSV file, processes the data, and uploads it to Snowflake, creating a solid foundation for our weather analysis.

```
# Insert records into the staging table
for r in records:
    datetime = r['datetime']
    tempmax = r['tempmax']
    temppin = r['temppin']
    temp = r['temp']
    feelslike = r['feelslike']
    humidity = r['humidity']
    precip = r['precip']
    windgust = r['windgust']
    windspeed = r['windspeed']
    pressure = r['pressure']
    cloudcover = r['cloudcover']
    solarradiation = r['solarradiation']
    severerisk = r['severerisk']
    sunrise = r['sunrise']
    sunset = r['sunset']
    conditions = r['conditions']

    #insert_sql = f"INSERT INTO {staging_table} (datetime, tempmax, temppin, temp, feelslike, humidity, precip, windgust, windspeed, pressure, cloudcover, solarradiation, severerisk, sunrise, sunset, conditions) VALUES ({(datetime)}, {(tempmax)}, {(temppin)}, {(temp)}, {(feelslike)}, {(humidity)}, {(precip)}, {(windgust)}, {(windspeed)}, {(pressure)}, {(cloudcover)}, {(solarradiation)}, {(severerisk)}, {(sunrise)}, {(sunset)}, {(conditions)})"
    #conn.execute(insert_sql) # Execute within the with block

    insert_sql = f"""INSERT INTO {staging_table}
        (datetime, tempmax, temppin, temp, feelslike, humidity, precip, windgust, windspeed, pressure, cloudcover, solarradiation,
        severerisk, sunrise, sunset, conditions)
        VALUES (%s, %s, %s)"""
    conn.execute(insert_sql, (datetime, tempmax, temppin, temp, feelslike, humidity, precip, windgust, windspeed, pressure, cloudcover, solarradiation, severerisk, sunrise, sunset, conditions))

conn.execute("COMMIT")

# perform UPsert
upsert_sql = f"""
    MERGE {target_table} AS target
    USING {staging_table} AS stage
    ON target.datetime = stage.datetime
    WHEN MATCHED THEN
        UPDATE SET
            target.datetime = stage.datetime,
            target.tempmax = stage.tempmax,
            target.temppin = stage.temppin,
            target.temp = stage.temp,
            target.feelslike = stage.feelslike,
            target.humidity = stage.humidity,
            target.precip = stage.precip,
            target.windgust = stage.windgust,
            target.windspeed = stage.windspeed,
            target.pressure = stage.pressure,
            target.cloudcover = stage.cloudcover,
            target.solarradiation = stage.solarradiation,
            target.severerisk = stage.severerisk,
            target.sunrise = stage.sunrise,
            target.sunset = stage.sunset,
            target.conditions = stage.conditions
    WHEN NOT MATCHED THEN
        INSERT ( datetime, tempmax, temppin, temp, feelslike, humidity, precip, windgust, windspeed, pressure, cloudcover, solarradiation, severerisk, sunrise, sunset, conditions)
        VALUES (stage.datetime, stage.tempmax, stage.temppin, stage.temp, stage.feelslike, stage.humidity, stage.precip, stage.windgust, stage.windspeed, stage.pressure, stage.cloudcover, stage.solarradiation, stage.severerisk, stage.sunrise, stage.sunset, stage.conditions)
"""

conn.execute(upsert_sql)
#commit the stage
conn.execute("COMMIT")
print(f"Stage Table '{staging_table}', Target table create '{target_table}' , Data loaded successfully in both the tables using Incremental Load ")
except Exception as e:
    conn.execute("ROLLBACK")
    print(e)
    raise e
```

Fig. 2b: Python code to load historical data from csv file

Github link for Historical data ETL DAG code:

https://github.com/patilgayatri22/DataWarehouse_FinalProject/blob/main/DataWarehouseProject/dags/csv_historical_data_etl.py

Recent Data Fetching:

For more recent data, from November 1, 2024, to the current date, we use the project_etl_1.py script. This script fetches data from the Visual Crossing Weather API and loads it into Snowflake.

This script fetches the most recent weather data and uses a MERGE statement to insert or update records in Snowflake, ensuring our dataset remains current.

By combining these two ETL processes, we create a comprehensive weather dataset that spans historical data and includes up-to-date information. This approach allows for both long-term trend analysis and current weather insights, providing a robust foundation for our weather analytics pipeline.

Github link for Recent data ETL DAG code:

https://github.com/patilgayatri22/DataWarehouse_FinalProject/blob/main/DataWarehouseProject/dags/project_ETL_pipeline1.py

IV. ELT Process with dbt

The dbt project structure consists of models for calculating various weather metrics:

dbt code link:

https://github.com/patilgayatri22/DataWarehouse_FinalProject/tree/main/DataWarehouseProject/proj

```
proj/
├── analyses/
└── models/
    ├── input/
    │   └── daily_weather.sql
    └── output/
        ├── cumulative_precipitation.sql
        ├── daily_changes.sql
        ├── daylight_duration.sql
        ├── pressure_drops.sql
        ├── solar_efficiency.sql
        └── temperature_range.sql
    └── snapshots/
        ├── snapshot.daily_changes.sql
        ├── snapshot.daylight_duration.sql
        ├── snapshot.pressure_drops.sql
        ├── snapshot.solar_efficiency.sql
        └── snapshot.temperature_range.sql
└── seeds/
```

Fig. 4: dbt Folder structure (main files)

Key transformations implemented:

1. `temperature_range`:

Tracks daily temperature variations, including min/max ranges and rolling averages.

```

WITH temperature_range AS (
    SELECT
        DATETIME,
        TEMPMAX,
        TEMPMIN,
        (TEMPMAX - TEMPMIN) AS TEMP_RANGE,
        CONDITIONS
    FROM {{ ref('daily_weather') }}
)
SELECT
    DATETIME,
    TEMPMAX,
    TEMPMIN,
    TEMP_RANGE,
    CONDITIONS
FROM temperature_range
ORDER BY DATETIME DESC

```

Figure 5 : Code to calculate temperature range

2. solar_efficiency:

Analyzes solar radiation patterns accounting for cloud cover impact.

```

WITH weather_data AS (
    SELECT
        DATETIME,
        solarradiation,
        cloudcover,
        CASE
            WHEN cloudcover > 0 THEN solarradiation / cloudcover
            ELSE NULL
        END AS solar_efficiency
    FROM {{ ref('daily_weather') }}
)
SELECT
    DATETIME,
    solarradiation,
    cloudcover,
    solar_efficiency
FROM weather_data

```

Figure 6 : Code to calculate solar efficiency

3. Daily_changes:

Monitors day-over-day changes in key weather metrics.

```

WITH weather_data AS (
    SELECT
        datetime,
        temp,
        humidity,
        LAG(temp) OVER (ORDER BY datetime) AS prev_temp,
        LAG(humidity) OVER (ORDER BY datetime) AS prev_humidity
    FROM {{ ref('daily_weather') }}
),
calculated_changes AS (
    SELECT
        datetime,
        temp,
        prev_temp,
        ((temp - prev_temp) / NULLIF(prev_temp, 0)) * 100 AS temp_change,
        humidity,
        prev_humidity,
        ((humidity - prev_humidity) / NULLIF(prev_humidity, 0)) * 100 AS humidity_change
    FROM weather_data
)
SELECT
    datetime,
    temp,
    prev_temp,
    temp_change,
    humidity,
    prev_humidity,
    humidity_change
FROM calculated_changes
ORDER BY datetime

```

Figure 7: Code to calculate daily changes

4. Daylight_duration:

Calculates daily sunlight hours and tracks seasonal changes.

```

WITH daylight_hours_data AS (
    SELECT
        datetime,
        CAST(CONCAT(datetime, ' ', sunrise) AS TIMESTAMP) AS sunrise_ts,
        CAST(CONCAT(datetime, ' ', sunset) AS TIMESTAMP) AS sunset_ts,
        DATEDIFF('second',
            CAST(CONCAT(datetime, ' ', sunrise) AS TIMESTAMP),
            CAST(CONCAT(datetime, ' ', sunset) AS TIMESTAMP)
        ) / 3600.0 AS daylight_hours
    FROM {{ ref('daily_weather') }}
)
SELECT
    datetime,
    sunrise_ts AS sunrise,
    sunset_ts AS sunset,
    daylight_hours
FROM daylight_hours_data
ORDER BY datetime DESC

```

Figure 8: Code to calculate daily duration

5. Cumulative_precipitation:

Maintains running totals and moving averages of precipitation.

```

WITH cumulative_precip_data AS (
    SELECT
        datetime,
        precip,
        SUM(precip) OVER (ORDER BY datetime ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS cumulative_precip
    FROM {{ ref('daily_weather') }}
)
SELECT
    datetime,
    precip,
    cumulative_precip
FROM cumulative_precip_data
ORDER BY datetime DESC

```

Figure 9 : Code to calculate cumulative precipitation

6. Pressure_drops:

Identifies significant weather changes by analyzing temperature, humidity, and wind patterns.

```

WITH weather_data AS (
    SELECT
        datetime,
        pressure,
        LAG(pressure) OVER (ORDER BY datetime) AS prev_pressure,
        (pressure - LAG(pressure) OVER (ORDER BY datetime)) AS pressure_change,
        CASE
            WHEN (pressure - LAG(pressure) OVER (ORDER BY datetime)) < -2 THEN 'Yes'
            ELSE 'No'
        END AS pressure_drop
    FROM {{ ref('daily_weather') }}
)
SELECT
    datetime,
    pressure,
    prev_pressure,
    pressure_change,
    pressure_drop
FROM weather_data

```

Figure 10 : Code to calculate pressure drop

Integration of dbt with Airflow:

```

Follows the standard dbt seed, run, and test pattern.
"""

from pendulum import datetime

from airflow import DAG
from airflow.operators.bash import BashOperator
from airflow.hooks.base import BaseHook

DBT_PROJECT_DIR = "/opt/airflow/dbt"

conn = BaseHook.get_connection('snowflake_conn')
with DAG(
    "BuildELT_dbt",
    start_date=datetime(2024, 10, 14),
    description="A sample Airflow DAG to invoke dbt runs using a BashOperator",
    schedule=None,
    catchup=False,
    default_args={
        "env": {
            "DBT_USER": conn.login,
            "DBT_PASSWORD": conn.password,
            "DBT_ACCOUNT": conn.extra_dejson.get("account"),
            "DBT_SCHEMA": conn.schema,
            "DBT_DATABASE": conn.extra_dejson.get("database"),
            "DBT_ROLE": conn.extra_dejson.get("role"),
            "DBT_WAREHOUSE": conn.extra_dejson.get("warehouse"),
            "DBT_TYPE": "snowflake"
        }
    },
) as dag:
    dbt_run = BashOperator(
        task_id="dbt_run",
        bash_command=f"/home/airflow/.local/bin/dbt run --profiles-dir {DBT_PROJECT_DIR} --project-dir {DBT_PROJECT_DIR}",
    )

    dbt_test = BashOperator(
        task_id="dbt_test",
        bash_command=f"/home/airflow/.local/bin/dbt test --profiles-dir {DBT_PROJECT_DIR} --project-dir {DBT_PROJECT_DIR}",
    )

    dbt_snapshot = BashOperator(
        task_id="dbt_snapshot",
        bash_command=f"/home/airflow/.local/bin/dbt snapshot --profiles-dir {DBT_PROJECT_DIR} --project-dir {DBT_PROJECT_DIR}",
    )

    # print_env_var = BashOperator(
    #     task_id='print_aa_variable',
    #     bash_command='echo "The value of AA is: $DBT_ACCOUNT,$DBT_ROLE,$DBT_DATABASE,$DBT_WAREHOUSE,$DBT_USER,$DBT_TYPE,$DBT_SCHEMA"'
    # )

    dbt_run >> dbt_test >> dbt_snapshot

```

Fig. 11: Python code for ELT DAG

This DAG runs the dbt models daily, ensuring that the abstract tables in Snowflake are updated with the latest calculations.

TABLE STRUCTURES

Table - Daily Weather

NAME ↑	TYPE
CLoudCover	# Float
CONDITIONS	Δ Varchar
DATETIME	⌚ Date
FEELSLIKE	# Float
HUMIDITY	# Float
PRECIP	# Float
PRESSURE	# Float
SEVERERISK	# Number
SOLARRADIATION	# Float
SUNRISE	⌚ Time
SUNSET	⌚ Time
TEMP	# Float
TEMPMAX	# Float
TEMPMIN	# Float
WINDGUST	# Float
WINDSPEED	# Float

Table - Temperature Range

NAME ↑	TYPE
CONDITIONS	Δ Varchar
DATETIME	⌚ Date
TEMPMAX	# Float
TEMPMIN	# Float
TEMP_RANGE	# Float

Table - Daily changes

NAME ↑	TYPE
DATETIME	⌚ Date
HUMIDITY	# Float
HUMIDITY_CHANGE	# Float
PREV_HUMIDITY	# Float
PREV_TEMP	# Float
TEMP	# Float
TEMP_CHANGE	# Float

Table- Daylight Duration

NAME ↑	TYPE
DATETIME	⌚ Date
DAYLIGHT_HOURS	# Number
SUNRISE	⌚ Timestamp_NTZ
SUNSET	⌚ Timestamp_NTZ

Table - Solar Efficiency

NAME ↑	TYPE
CLOUDCOVER	# Float
DATETIME	⌚ Date
SOLARRADIATION	# Float
SOLAR_EFFICIENCY	# Float

Table - Pressure Drops

NAME	TYPE ↑
DATETIME	⌚ Date
PRESSURE	# Float
PRESSURE_CHANGE	# Float
PREV_PRESSURE	# Float
PRESSURE_DROP	⚠ Varchar

Table- Cumulative precipitation

NAME ↑	TYPE
CUMULATIVE_PRECIP	# Float
DATETIME	⌚ Date
PRECIP	# Float

V. Airflow Data Pipelines

Airflow is a platform used to automate and manage workflows by organizing tasks into Directed Acyclic Graphs (DAGs). It enables users to schedule tasks, monitor progress, and ensure data is processed in the correct order or in parallel, making it ideal for automating ETL processes in data pipelines.

ETL Dag-

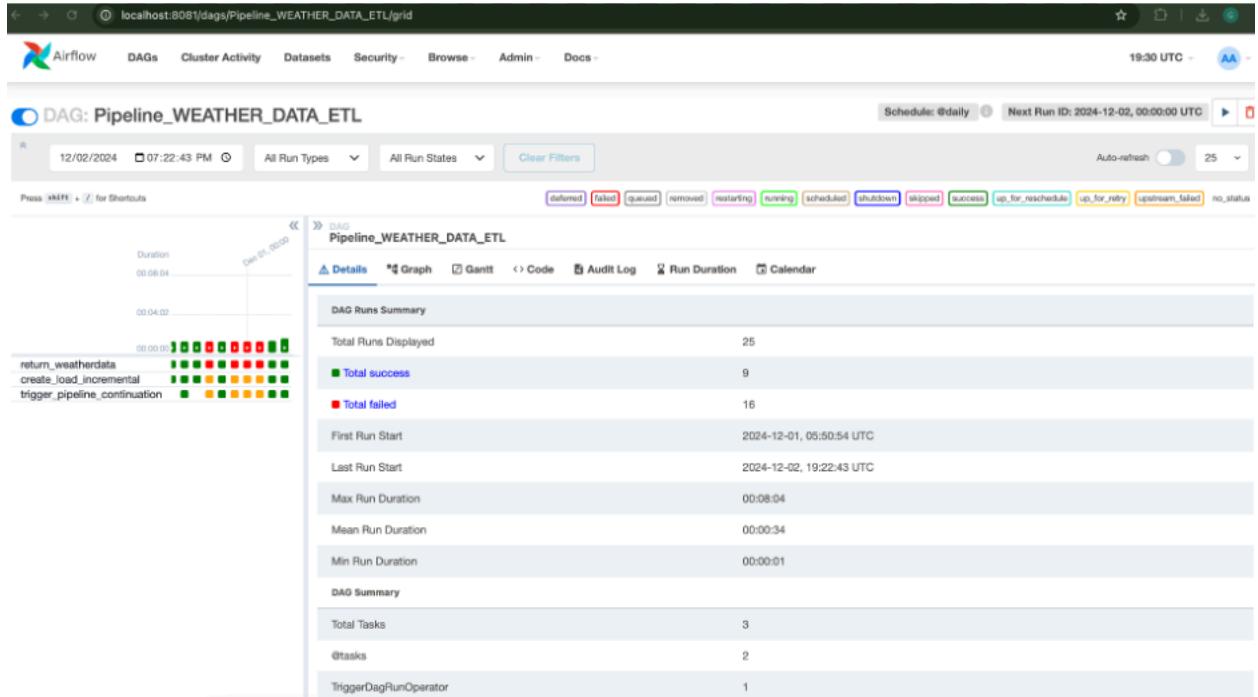


Fig. 12: Airflow ETL DAG

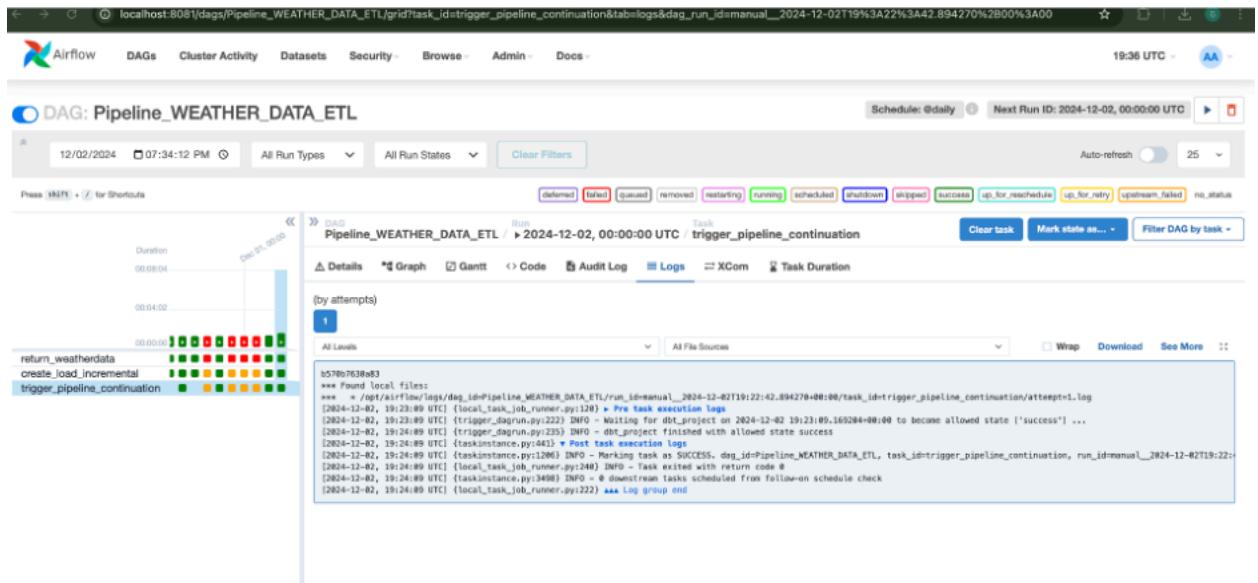


Fig 13 : Airflow ETL Dag with Logs

ELT Dag -

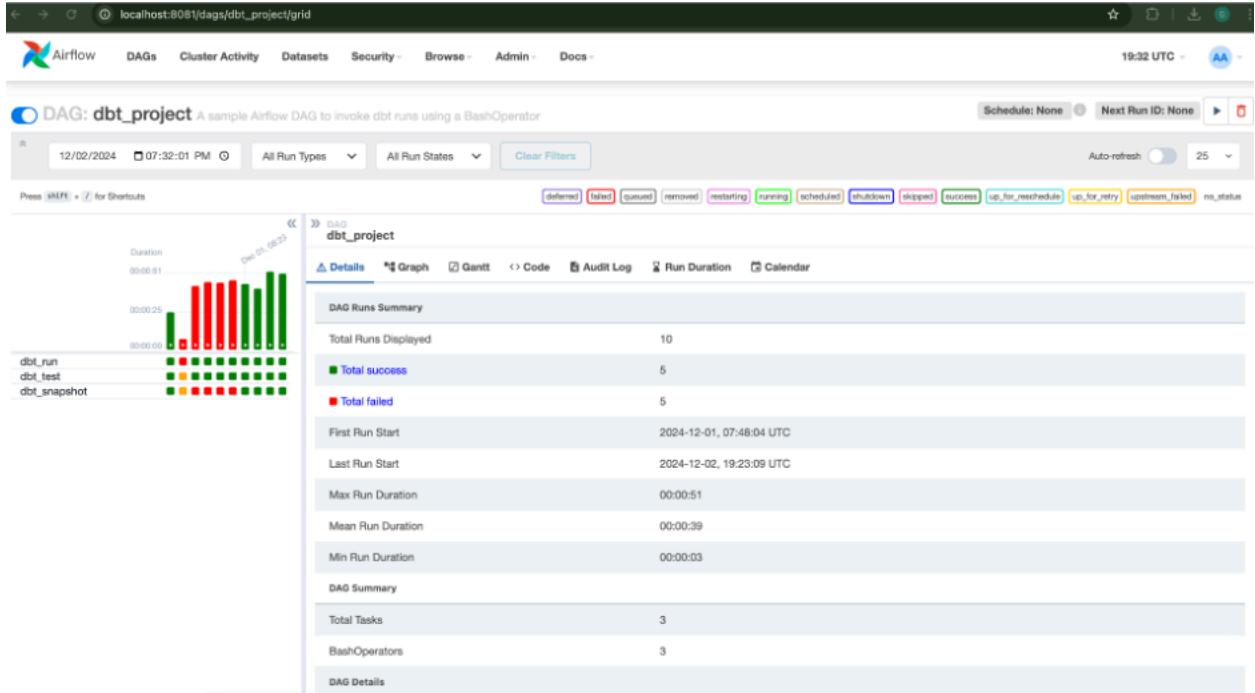


Fig 14 : Airflow ELT Dag

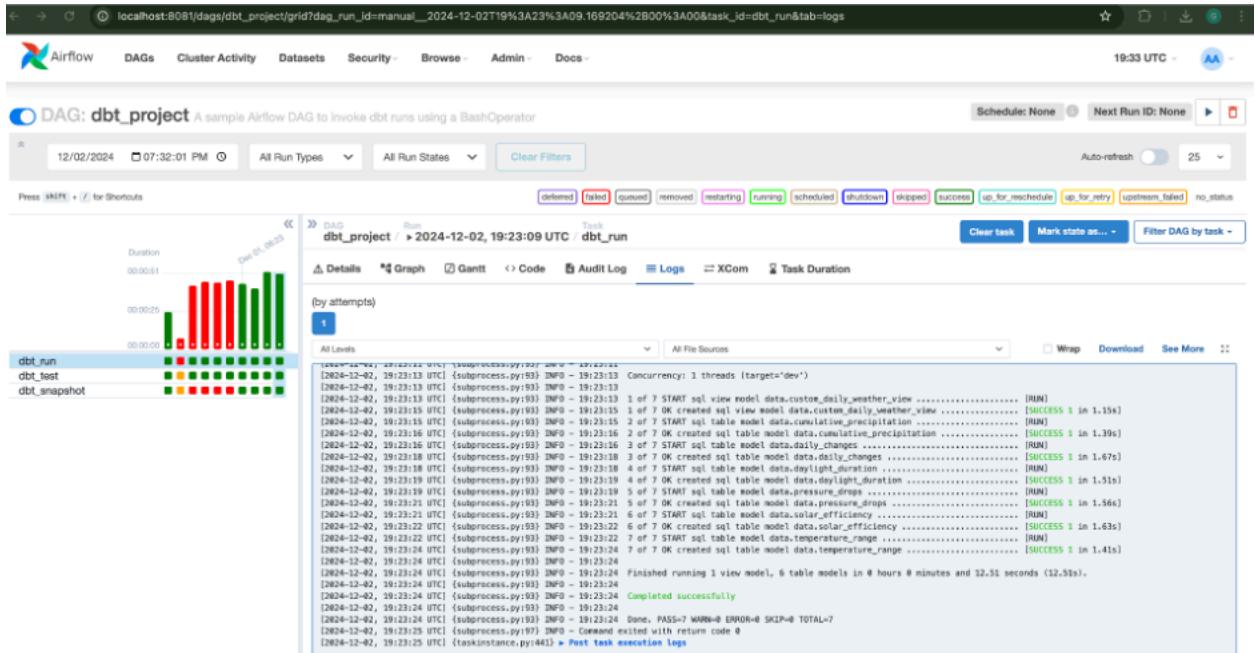


Fig 15a : Airflow ETL Dag with Logs

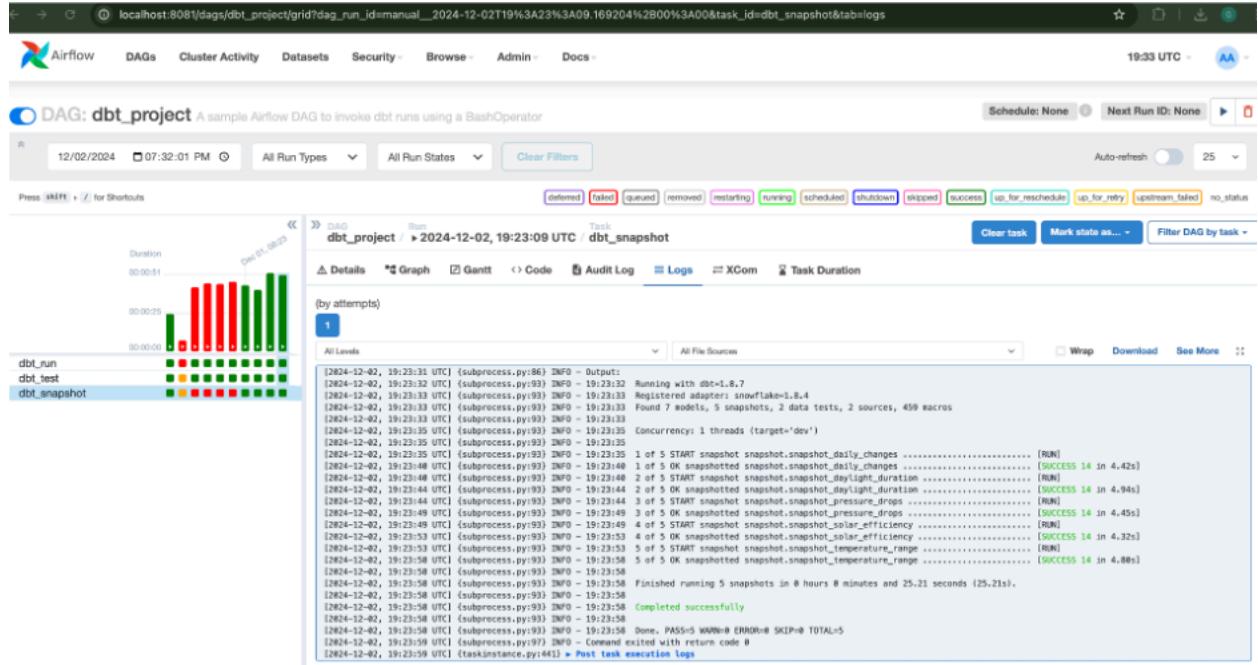


Fig 15b : Airflow ETL Dag with Logs

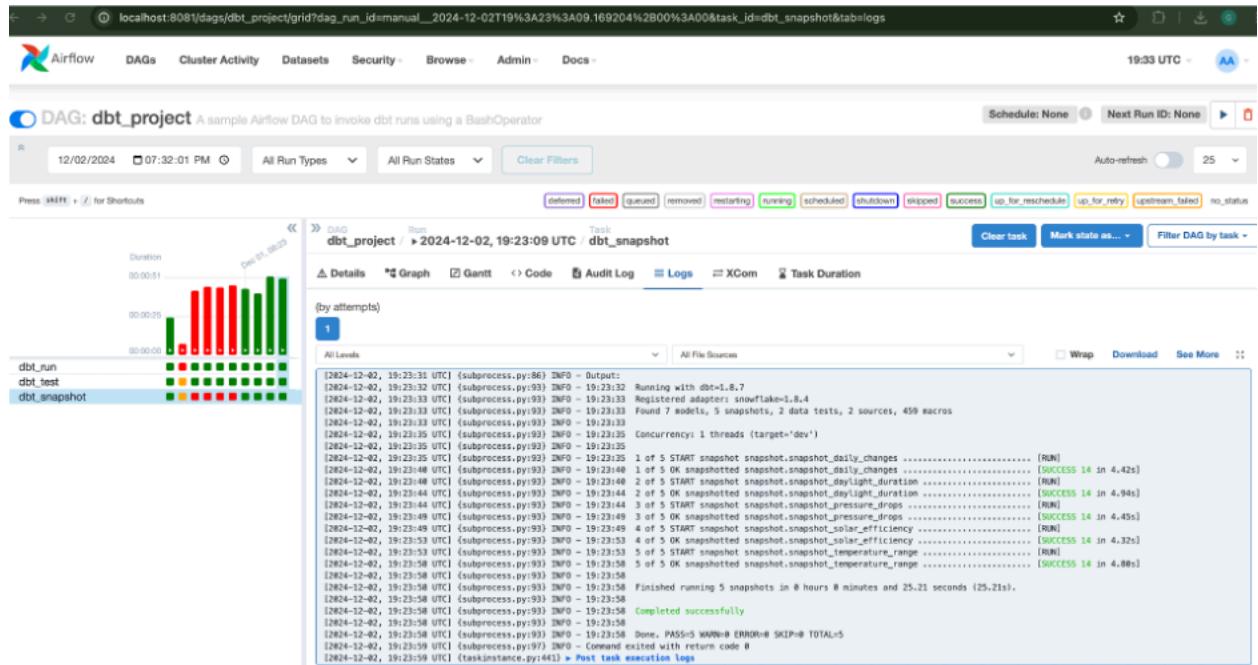


Fig 15c : Airflow ETL Dag with Logs

DBT screenshots from terminal-

```

[gayatripatil@Gayatris-MacBook-Air proj % dbt run
04:51:12  Running with dbt=1.8.7
04:51:13  Registered adapter: snowflake=1.8.4
04:51:13  Found 7 models, 2 data tests, 2 sources, 459 macros
04:51:13
04:51:15  Concurrency: 1 threads (target='dev')
04:51:15
04:51:15  1 of 7 START sql view model data.custom_daily_weather_view ..... [RUN]
04:51:16  1 of 7 OK created sql view model data.custom_daily_weather_view ..... [SUCCESS 1 in 1.50s]
04:51:16  2 of 7 START sql table model data.cumulative_precipitation ..... [RUN]
04:51:18  2 of 7 OK created sql table model data.cumulative_precipitation ..... [SUCCESS 1 in 1.69s]
04:51:18  3 of 7 START sql table model data.daily_changes ..... [RUN]
04:51:19  3 of 7 OK created sql table model data.daily_changes ..... [SUCCESS 1 in 1.76s]
04:51:19  4 of 7 START sql table model data.daylight_duration ..... [RUN]
04:51:22  4 of 7 OK created sql table model data.daylight_duration ..... [SUCCESS 1 in 2.16s]
04:51:22  5 of 7 START sql table model data.pressure_drops ..... [RUN]
04:51:23  5 of 7 OK created sql table model data.pressure_drops ..... [SUCCESS 1 in 1.29s]
04:51:23  6 of 7 START sql table model data.solar_efficiency ..... [RUN]
04:51:25  6 of 7 OK created sql table model data.solar_efficiency ..... [SUCCESS 1 in 2.16s]
04:51:25  7 of 7 START sql table model data.temperature_range ..... [RUN]
04:51:27  7 of 7 OK created sql table model data.temperature_range ..... [SUCCESS 1 in 1.52s]
04:51:27
04:51:27  Finished running 1 view model, 6 table models in 0 hours 0 minutes and 13.45 seconds (13.45s).
04:51:27
04:51:27  Completed successfully
04:51:27
04:51:27  Done. PASS=7 WARN=0 ERROR=0 SKIP=0 TOTAL=7
gayatripatil@Gayatris-MacBook-Air proj %

```

Fig 16 : DBT Run

```

[gayatripatil@Gayatris-MacBook-Air proj % dbt snapshot
06:17:26  Running with dbt=1.8.7
06:17:26  Registered adapter: snowflake=1.8.4
06:17:27  Found 7 models, 2 data tests, 5 snapshots, 2 sources, 459 macros
06:17:27
06:17:30  Concurrency: 1 threads (target='dev')
06:17:30
06:17:30  1 of 5 START snapshot snapshot.snapshot_daily_changes ..... [RUN]
06:17:32  1 of 5 OK snapshotted snapshot.snapshot_daily_changes ..... [SUCCESS 1 in 2.73s]
06:17:32  2 of 5 START snapshot snapshot.snapshot_daylight_duration ..... [RUN]
06:17:35  2 of 5 OK snapshotted snapshot.snapshot_daylight_duration ..... [SUCCESS 1 in 2.30s]
06:17:35  3 of 5 START snapshot snapshot.snapshot_pressure_drops ..... [RUN]
06:17:37  3 of 5 OK snapshotted snapshot.snapshot_pressure_drops ..... [SUCCESS 1 in 1.77s]
06:17:37  4 of 5 START snapshot snapshot.snapshot_solar_efficiency ..... [RUN]
06:17:38  4 of 5 OK snapshotted snapshot.snapshot_solar_efficiency ..... [SUCCESS 1 in 1.92s]
06:17:38  5 of 5 START snapshot snapshot.snapshot_temperature_range ..... [RUN]
06:17:41  5 of 5 OK snapshotted snapshot.snapshot_temperature_range ..... [SUCCESS 1 in 2.44s]
06:17:41
06:17:41  Finished running 5 snapshots in 0 hours 0 minutes and 14.39 seconds (14.39s).
06:17:41
06:17:41  Completed successfully
06:17:41
06:17:41  Done. PASS=5 WARN=0 ERROR=0 SKIP=0 TOTAL=5
gayatripatil@Gayatris-MacBook-Air proj %

```

Fig 16 : DBT Snapshot

```

[gayatripatil@Gayatris-MacBook-Air proj % dbt test
06:23:31  Running with dbt=1.8.7
06:23:31  Registered adapter: snowflake=1.8.4
06:23:31  Found 7 models, 5 snapshots, 2 data tests, 2 sources, 459 macros
06:23:31
06:23:33  Concurrency: 1 threads (target='dev')
06:23:33
06:23:34  1 of 2 START test not_null_temperature_range_datetime ..... [RUN]
06:23:34  1 of 2 PASS not_null_temperature_range_datetime ..... [PASS in 0.76s]
06:23:34  2 of 2 START test unique_temperature_range_datetime ..... [RUN]
06:23:34  2 of 2 PASS unique_temperature_range_datetime ..... [PASS in 0.76s]
06:23:34
06:23:34  Finished running 2 data tests in 0 hours 0 minutes and 3.04 seconds (3.04s).
06:23:34
06:23:34  Completed successfully
06:23:34
06:23:34  Done. PASS=2 WARN=0 ERROR=0 SKIP=0 TOTAL=2
gayatripatil@Gayatris-MacBook-Air proj %

```

Fig 16 : DBT Test

VI. Snowflake screenshots for all tables

The screenshot shows a Snowflake query interface. The query in the editor is:

```

40
41
42 SHOW TABLES IN schema weather.data;
43
44
    
```

The results table has columns: created_on, name, database_name, schema_name, kind, comment, and cluster. The data shows 8 tables created on 2024-12-02:

created_on	name	database_name	schema_name	kind	comment	cluster
2024-12-02 11:23:15.483 -0800	CUMULATIVE_PRECIPITATION	WEATHER	DATA	TRANSIENT		
2024-12-02 11:23:16.979 -0800	DAILY_CHANGES	WEATHER	DATA	TRANSIENT		
2024-12-02 11:22:46.360 -0800	DAILY_WEATHER	WEATHER	DATA	TABLE		
2024-12-02 11:22:46.845 -0800	DAILY_WEATHER_STAGE	WEATHER	DATA	TABLE		
2024-12-02 11:23:18.723 -0800	DAYLIGHT_DURATION	WEATHER	DATA	TRANSIENT		
2024-12-02 11:23:20.123 -0800	PRESSURE_DROPS	WEATHER	DATA	TRANSIENT		
2024-12-02 11:23:21.722 -0800	SOLAR_EFFICIENCY	WEATHER	DATA	TRANSIENT		
2024-12-02 11:23:23.324 -0800	TEMPERATURE_RANGE	WEATHER	DATA	TRANSIENT		

On the right side, there is a "Query Details" panel showing: Query duration 62ms, Rows 8, Query ID 01b8c479-0004-3865-... and a "Show more" button.

Fig 17a: All Tables Created in the Snowflake

The screenshot shows a Snowflake query interface. The query in the editor is:

```

58
59
60 select * from WEATHER_DATA.DAILY_WEATHER;
61
62
    
```

The results table has columns: DATETIME, TEMPMAX, TEMPMIN, TEMP, FEELSlike, HUMIDITY, PRECIP, WINDGUST, WINDSPEED, and PRESSURE. The data shows 16 rows of weather data from November 1st to December 2nd, 2024:

DATETIME	TEMPMAX	TEMPMIN	TEMP	FEELSlike	HUMIDITY	PRECIP	WINDGUST	WINDSPEED	PRESSURE
2024-11-01	18.8	6	12.1	12	77.3	5.965	33.5	21	1016.2
2024-11-02	16.9	11.1	13.5	13.5	79.9	2.213	31.3	16.4	1013.9
2024-11-03	20.9	7.3	13.5	13.2	64.3	0	57.6	40.3	1017.4
2024-11-04	21.2	10	14.2	14.2	50	0	44.3	22	1020.4
2024-11-05	21.3	7.4	14.5	14.5	50.7	0	52.7	37.1	1017.9
2024-11-06	21.7	9	15.2	15.2	31.2	0	63	35.3	1016.3
2024-11-07	21.7	4.7	12	12	48.7	0	13	8.8	1021.1
2024-11-08	21.2	5	11.8	11.8	59.3	0	9.4	6.8	1020.4
2024-11-09	20.8	4.6	11.4	11.3	60.9	0	7.6	5.6	1016.3
2024-11-10	20.8	4.2	12.1	12.1	62.7	0	11.2	10.3	1016.3
2024-11-11	15	8.1	11.2	11.1	77.3	3.139	31.3	12.3	1016.8
2024-11-12	16.6	3.1	9.1	9	82.2	0	14.8	12.3	1022.6
2024-11-13	16.1	4.7	10.6	10.5	79.3	0	45.2	21.9	1020
2024-11-14	16.3	5.7	10.3	10.3	84.4	2.871	16.6	11.9	1016.6
2024-11-15	14.7	4.8	9	8.6	74.5	0	25.9	17.8	1013.9
2024-11-16	15.7	4.3	8.8	7.6	53.9	0	25.9	19.9	1017.7

On the right side, there are three charts: a histogram for DATETIME, a line chart for TEMPMAX, and a line chart for TEMPMIN. There is also a "Query Details" panel showing: Query duration 772ms, Rows 32, Query ID 01b8c47c-0004-386b-0-... and a "Show more" button.

Fig 17b: Table Daily_Weather

Tables Created through DBT run -

48
49 | select * from WEATHER.DATA.TEMPERATURE_RANGE;
50

Results Chart

DATETIME	TEMPMAX	TEMPMIN	TEMP_RANGE	CONDITIONS
1 2024-12-02	19.3	4.5	14.8	Clear
2 2024-12-01	14.5	5.5	9	Partially cloudy
3 2024-11-30	14.2	0.1	14.1	Partially cloudy
4 2024-11-29	15.4	1.4	14	Clear
5 2024-11-28	14.9	1.3	13.6	Clear
6 2024-11-27	15	4	11	Clear
7 2024-11-26	15.8	7	8.8	Rain, Partially cloudy
8 2024-11-25	13.7	8.3	5.4	Rain, Partially cloudy
9 2024-11-24	16.2	3.9	12.3	Partially cloudy
10 2024-11-23	15	6.5	8.5	Rain, Partially cloudy
11 2024-11-22	14.7	10.9	3.8	Rain, Partially cloudy
12 2024-11-21	16.7	10.2	6.5	Rain, Overcast
13 2024-11-20	15.2	10.6	4.6	Rain, Partially cloudy
14 2024-11-19	13.7	1.3	12.4	Clear
15 2024-11-18	15	4.1	10.9	Clear
16 2024-11-17	15.8	2.5	13.3	Rain, Partially cloudy

Query Details ...
Query duration 338ms
Rows 32
Query ID 01b8c47f-0004-3987-0...
Show more

DATETIME
2024-11-01 2024-12-02

TEMPMAX
13.7 21.7

TEMPMIN
0.1 #

Ask Copilot

Fig 17c: Table Temperature Range

51
52
53 | SELECT * FROM WEATHER.DATA.DAYLIGHT_DURATION;
54
55

Results Chart

DATETIME	SUNRISE	SUNSET	DAYLIGHT_HOURS
1 2024-12-02	2024-12-02 07:06:26.000	2024-12-02 16:44:51.000	9.640278
2 2024-12-01	2024-12-01 07:05:29.000	2024-12-01 16:45:00.000	9.658611
3 2024-11-30	2024-11-30 07:04:31.000	2024-11-30 16:45:12.000	9.678056
4 2024-11-29	2024-11-29 07:03:32.000	2024-11-29 16:45:26.000	9.698333
5 2024-11-28	2024-11-28 07:02:32.000	2024-11-28 16:45:42.000	9.719444
6 2024-11-27	2024-11-27 07:01:32.000	2024-11-27 16:46:01.000	9.741389
7 2024-11-26	2024-11-26 07:00:30.000	2024-11-26 16:46:21.000	9.764167
8 2024-11-25	2024-11-25 06:59:29.000	2024-11-25 16:46:43.000	9.787222
9 2024-11-24	2024-11-24 06:58:26.000	2024-11-24 16:47:08.000	9.811667
10 2024-11-23	2024-11-23 06:57:23.000	2024-11-23 16:47:35.000	9.836667
11 2024-11-22	2024-11-22 06:56:19.000	2024-11-22 16:48:03.000	9.862222
12 2024-11-21	2024-11-21 06:55:15.000	2024-11-21 16:48:34.000	9.888611
13 2024-11-20	2024-11-20 06:54:11.000	2024-11-20 16:49:07.000	9.915556
14 2024-11-19	2024-11-19 06:53:06.000	2024-11-19 16:49:41.000	9.943056
15 2024-11-18	2024-11-18 06:52:01.000	2024-11-18 16:50:18.000	9.971389
16 2024-11-17	2024-11-17 06:50:56.000	2024-11-17 16:50:56.000	10.000000

Query Details ...
Query duration 65ms
Rows 32
Query ID 01b8c484-0004-3864-...
Show more

DATETIME
2024-11-01 2024-12-02

SUNRISE
2024-11-01 2024-12-02

SUNSET
2024-11-01 2024-12-02

Ask Copilot

Fig 17d: Table Daylight Duration

DEV.PUBLIC Settings

```
55 | select * from WEATHER.DATA.DAILY_CHANGES;
```

Results Chart

	DATETIME	TEMP	PREV_TEMP	TEMP_CHANGE	HUMIDITY	PREV_HUMIDITY	HUMIDITY_CHANGE
1	2024-11-01	12.1	null	null	77.3	null	null
2	2024-11-02	13.5	12.1	11.570247934	79.9	77.3	3.363518758
3	2024-11-03	13.5	13.5	0	64.3	79.9	-19.524405507
4	2024-11-04	14.2	13.5	5.185185185	50	64.3	-22.239502333
5	2024-11-05	14.5	14.2	2.112676056	50.7	50	1.4
6	2024-11-06	15.2	14.5	4.827586207	31.2	50.7	-38.461538462
7	2024-11-07	12	15.2	-21.052631579	48.7	31.2	56.08974359
8	2024-11-08	11.8	12	-1.666666667	59.3	48.7	21.765913758
9	2024-11-09	11.4	11.8	-3.389830508	60.9	59.3	2.698145025
10	2024-11-10	12.1	11.4	6.140350877	62.7	60.9	2.9556685025
11	2024-11-11	11.2	12.1	-7.438016529	77.3	62.7	23.285486443
12	2024-11-12	9.1	11.2	-18.75	82.2	77.3	6.338939198
13	2024-11-13	10.6	9.1	16.483516484	79.3	82.2	-3.527980535
14	2024-11-14	10.3	10.6	-2.830188679	84.4	79.3	6.431273644
15	2024-11-15	9	10.3	-12.621359223	74.5	84.4	-11.72985782
16	2024-11-16	8.8	9	-2.222222222	53.9	74.5	-27.651006711

Query Details ...
 Query duration 361ms
 Rows 32
 Query ID 01b8c486-0004-3864-...
 Show more ▼

DATETIME ...

 2024-11-01 2024-12-02

TEMP #

 6.2 15.2

PREV_TEMP #

 6.2

Ask Copilot

Fig 17e: Table Daily Changes

DEV.PUBLIC Settings

```
56
57
58 | select * from WEATHER.DATA.CUMULATIVE_PRECIPITATION;
59
60
```

Results Chart

	DATETIME	PRECIP	CUMULATIVE_PRECIP
1	2024-12-02	0	82.131
2	2024-12-01	0	82.131
3	2024-11-30	0	82.131
4	2024-11-29	0	82.131
5	2024-11-28	0	82.131
6	2024-11-27	0	82.131
7	2024-11-26	2.856	82.131
8	2024-11-25	2.496	79.275
9	2024-11-24	0	76.779
10	2024-11-23	4.569	76.779
11	2024-11-22	43.525	72.21
12	2024-11-21	6.178	28.685
13	2024-11-20	8.087	22.507
14	2024-11-19	0	14.42
15	2024-11-18	0	14.42
16	2024-11-17	0.232	14.42

Query Details ...
 Query duration 224ms
 Rows 32
 Query ID 01b8c487-0004-386b-...
 Show more ▼

DATETIME ...

 2024-11-01 2024-12-02

PRECIP #

 0 43.525

CUMULATIVE_PRECIP #

 5.965

Ask Copilot

Fig 17f: Table Cumulative Precipitation

```

65
66
67 | select * from WEATHER.DATA.PRESSURE_DROPS;
68
69
70

```

Results Chart

	DATETIME	PRESSURE	PREV_PRESSURE	PRESSURE_CHANGE	PRESSURE_DROP
1	2024-11-01	1016.2	null	null	No
2	2024-11-02	1013.9	1016.2	-2.3	Yes
3	2024-11-03	1017.4	1013.9	3.5	No
4	2024-11-04	1020.4	1017.4	3	No
5	2024-11-05	1017.9	1020.4	-2.5	Yes
6	2024-11-06	1016.3	1017.9	-1.6	No
7	2024-11-07	1021.1	1016.3	4.8	No
8	2024-11-08	1020.4	1021.1	-0.7	No
9	2024-11-09	1016.3	1020.4	-4.1	Yes
10	2024-11-10	1016.3	1016.3	0	No
11	2024-11-11	1016.6	1016.3	0.3	No
12	2024-11-12	1022.6	1016.6	6	No
13	2024-11-13	1020	1022.6	-2.6	Yes
14	2024-11-14	1016.6	1020	-3.4	Yes
15	2024-11-15	1013.9	1016.6	-2.7	Yes
16	2024-11-16	1017.7	1013.9	3.8	No

Query Details
...

Query duration
218ms

Rows
32

Query ID
01b8c489-0004-3863-...

Show more

DATETIME
...

PRESSURE
#

PREV_PRESSURE
#

Ask Copilot

Fig 17g: Table Pressure Drops

```

65
66
67 | select * from WEATHER.DATA.SOLAR_EFFICIENCY;
68
69
70
71
72
73

```

Results Chart

	DATETIME	SOLARRADIATION	CLOUDCOVER	SOLAR_EFFICIENCY
1	2024-11-01	15.4	35.9	0.4289693593
2	2024-11-02	11.7	69.4	0.1685878963
3	2024-11-03	16.3	5.6	2.910714286
4	2024-11-04	16.3	1.5	10.8666666667
5	2024-11-05	14.3	10.7	1.336448598
6	2024-11-06	16.3	3.3	4.939393939
7	2024-11-07	15.5	0	null
8	2024-11-08	15	0.6	25
9	2024-11-09	15.1	9.3	1.623655914
10	2024-11-10	14.9	6.5	2.292307692
11	2024-11-11	5.2	48.1	0.1081081081
12	2024-11-12	14.8	10.4	1.423076923
13	2024-11-13	7.1	37.6	0.1888297872
14	2024-11-14	9.5	43.8	0.2168949772
15	2024-11-15	14.4	20.7	0.6956521739
16	2024-11-16	14.5	5.2	2.788461538

Query Details
...

Query duration
445ms

Rows
32

Query ID
01b8c48b-0004-3869-...

Show more

DATETIME
...

SOLARRADIATION
#

CLOUDCOVER
#

Ask Copilot

Fig 17h: Table Solar Efficiency

DBT Snapshots screens -

DEV.PUBLIC Settings

```

75 --snapshots
76
77 | select * from WEATHER.SNAPSHOT.SNAPSHOT_DAILY_CHANGES;
78
79
80

```

↳ Results ⚡ Chart

	DATETIME	TEMP	PREV_TEMP	TEMP_CHANGE	HUMIDITY	PREV_HUMIDITY	HUMIDITY_CHANGE	DBT_SCD_ID
1	2024-12-02	10.0	8.5	17.6470588	77.7	84.4	-7.9383886	818120f3e8e6e0e5c49c22dedc8e
2	2024-12-01	8.9	6.2	43.5483871	74.5	84.2	-11.5201900	5ea9a2b3f39880df4acf69c2a4ed
3	2024-11-30	6.2	null	null	84.2	null	null	274963eda8215ba4f5144f49522c
4	2024-01-01	10.0	null	null	86.6	null	null	21b70839345ea1da608dad68e66
5	2024-01-02	8.8	10.0	-12.0000000	90.6	86.6	4.6189400	e5af3c91a999bb1a729ad045b9e8
6	2024-01-03	9.1	8.8	3.4090900	86.1	90.6	-4.9668900	8299b0af66e13ac86ef22dce83d6
7	2024-01-04	8.5	9.1	-6.5934100	80.4	86.1	-6.6202100	fac289115e6ffe703d8a6bb4601ec
8	2024-01-05	9.9	8.5	16.4705900	71.3	80.4	-11.3184100	4c86026d0d340db57e7e225196b
9	2024-01-06	6.7	9.9	-32.3232300	86.9	71.3	21.8793800	8424a0ba78144de793d13346a05
10	2024-01-07	5.5	6.7	-17.9104500	70.8	86.9	-18.5270400	aa8d6b10843a6cf201aa109fe2e3a
11	2024-01-08	6.6	5.5	20.0000000	64.0	70.8	-9.6045200	c3876fe4ce983cc9bea411acb03a
12	2024-01-09	8.7	6.6	31.8181800	75.9	64.0	18.5937500	1e76308de2899d8cf593a28b98f3
13	2024-01-10	8.5	8.7	-2.2988500	81.4	75.9	7.2463800	e7611d87857495097d202a07214
14	2024-01-11	6.9	8.5	-18.8235300	55.9	81.4	-31.3267800	5b76e70bf0c08b325307108d610
15	2024-01-12	6.8	6.9	-1.4492800	69.6	55.9	24.5080500	94c3dc954d43e4e0c6efa32d1983
16	2024-01-13	8.2	6.8	20.5882400	77.7	69.6	11.6379300	2015049de4a4423939ebd950086

Query Details ...
 Query duration 343ms
 Rows 337
 Query ID 01b8c48d-0004-386b-...
 Show more

DATETIME
TEMP
PREV_TEMP

Ask Copilot

Fig 18a: Snapshot Daily Changes

```

88
89 | select * from WEATHER.SNAPSHOT.SNAPSHOT_TEMPERATURE_RANGE;
90
91
92

```

↳ Results ⚡ Chart

	DATETIME	TEMPMAX	TEMPMIN	TEMP_RANGE	CONDITIONS	DBT_SCD_ID	DBT_UPDATED_AT
1	2024-12-02	16.2	4.5	11.7	Clear	818120f3e8e6e0e5c49c22dedc8e316c	2024-12-02
2	2024-11-30	14.2	0.1	14.1	Partially cloudy	274963eda8215ba4f5144f49522c2c42	2024-11-30
3	2024-12-01	15.0	4.5	10.5	Partially cloudy	5ea9a2b3f39880df4acf69c2a4edb4a	2024-12-01
4	2024-11-15	14.7	4.9	9.8	Partially cloudy	340157159b701e52f67020b2f23ea308	2024-11-15
5	2024-11-14	16.3	5.7	10.6	Rain, Partially cloudy	715261695f64cea9b2c78aeb4f4d336f	2024-11-14
6	2024-11-13	16.0	4.6	11.4	Rain, Partially cloudy	3f757dabe00f3d593d6c20966c96a798	2024-11-13
7	2024-11-12	16.6	3.1	13.5	Clear	aec08faf167800c5fe986b6d1fa281df	2024-11-12
8	2024-11-11	14.9	8.1	6.8	Rain, Partially cloudy	7a22cb53f8c68230fe7e8c3305919b9	2024-11-11
9	2024-11-10	20.8	4.2	16.6	Clear	034bf1ff9447a9512f697e5c35f13665	2024-11-10
10	2024-11-09	20.8	4.6	16.2	Clear	fc95321d8fc13af5c24098032482a178	2024-11-09
11	2024-11-08	21.2	5.0	16.2	Clear	3a64227717e55c5df8954a39b3b51a0	2024-11-08
12	2024-11-07	21.7	4.7	17.0	Clear	a4c0ee96cd4f8e07c03ebad502ba5b92	2024-11-07
13	2024-11-06	21.7	9.0	12.7	Clear	2c980c692b8ad447a40e1119eeb936a2	2024-11-06
14	2024-11-05	21.3	7.4	13.9	Clear	aac4ad3c54cd29db5002b7fe40a02750	2024-11-05
15	2024-11-04	21.2	10.0	11.2	Clear	b38424a882639d3d42ab43bc33adef7b	2024-11-04
16	2024-11-03	20.9	7.3	13.6	Clear	e1b76a5b1118abbe9aeed7f4fca26518	2024-11-03

Query Details ...
 Query duration 1ms
 Rows 16
 Query ID 01b8c48d-0004-386b-...
 Show more

DATETIME
TEMPMAX
TEMPMIN

Fig 18b: Snapshot Temperature Range

```

79 | select * from WEATHER.SNAPSHOT.SNAPSHOT_DAYLIGHT_DURATION;
80
81
82
83

```

Results Chart

DATETIME	SUNRISE	SUNSET	DAYLIGHT_HOURS	DBT_SCD_ID	DBT
1 2024-12-02	2024-12-02 07:06:26.000	2024-12-02 16:44:51.000	9.640278	818120f3e8e6e0e5c49c22dedc8e316c	202
2 2024-11-30	2024-11-30 07:04:31.000	2024-11-30 16:45:12.000	9.678056	274963eda8215ba4f5144f49522c2c42	202
3 2024-12-01	2024-12-01 07:05:29.000	2024-12-01 16:45:00.000	9.658611	5ea9a2b3f39880df4acf69c2a4edb4a	202
4 2024-11-15	2024-11-15 06:48:44.000	2024-11-15 16:52:19.000	10.059722	340157159b701e52f67020b2f23ea308	202
5 2024-11-14	2024-11-14 06:47:39.000	2024-11-14 16:53:03.000	10.090000	715261695f64cea9b2c79aebe4fd336f	202
6 2024-11-13	2024-11-13 06:46:33.000	2024-11-13 16:53:49.000	10.121111	3f757dabe00f3d593d6c20966c96a798	202
7 2024-11-12	2024-11-12 06:45:26.000	2024-11-12 16:54:37.000	10.153056	aec08af167800c5fe986b6d1fa281df	202
8 2024-11-11	2024-11-11 06:44:20.000	2024-11-11 16:55:27.000	10.185278	7a22cb53f8c68230fe77e8c3305919b9	202
9 2024-11-10	2024-11-10 06:43:14.000	2024-11-10 16:56:18.000	10.217778	034bf1ff9447a9512f697e5c35f13665	202
10 2024-11-09	2024-11-09 06:42:08.000	2024-11-09 16:57:11.000	10.250833	fc95321d8fc13af5c24098032482a178	202
11 2024-11-08	2024-11-08 06:41:02.000	2024-11-08 16:58:05.000	10.284167	3a64227717e55c5dfe9854a39b3b51a0	202
12 2024-11-07	2024-11-07 06:39:56.000	2024-11-07 16:59:02.000	10.318333	a4c0ee96cd4f8e07c03ebad502ba5b92	202
13 2024-11-06	2024-11-06 06:38:50.000	2024-11-06 17:00:00.000	10.352778	2c980c692b8ad447a40e1119eeb936a2	202
14 2024-11-05	2024-11-05 06:37:44.000	2024-11-05 17:00:59.000	10.387500	aac4ad3c54cd29db5002b7fe40a02750	202
15 2024-11-04	2024-11-04 06:36:39.000	2024-11-04 17:02:00.000	10.422500	b38424a882639d3d42ab43bc33adef7b	202
16 2024-11-03	2024-11-03 06:35:33.000	2024-11-03 17:03:03.000	10.458333	e1b76a5b1118abbe9aeed7f4fc26518	202

Query Details
Query duration 317ms
Rows 332
Query ID 01b8c490-0004-3987-...
Show more

DATETIME

SUNRISE

SUNSET

Ask Copilot

Fig 18c: Snapshot Daylight Duration

```

84
85
86 | select * from WEATHER.SNAPSHOT.SNAPSHOT_SOLAR_EFFICIENCY;
87

```

Results Chart

DATETIME	SOLARRADIATION	CLOUDCOVER	SOLAR EFFICIENCY	DBT_SCD_ID	DBT_UPDATED_AT	DB
1 2024-12-02	88.0	4.2	20.9523810	818120f3e8e6e0e5c49c22dedc8e316c	2024-12-02	202
2 2024-12-01	70.5	40.0	1.7625000	5ea9a2b3f39880df4acf69c2a4edb4a	2024-12-01	202
3 2024-11-30	12.5	26.0	0.4807692	274963eda8215ba4f5144f49522c2c42	2024-11-30	202
4 2024-01-01	89.3	42.1	2.1211401	21b70839345ea1da806dad68e66202a4	2024-01-01	202
5 2024-01-02	38.4	70.5	0.5446809	e5af3c91a999bb1a729ad045b9e87371	2024-01-02	202
6 2024-01-03	99.4	80.8	1.6348684	8299b0af6e13ac86ef22dc83d6d167	2024-01-03	202
7 2024-01-04	104.1	16.7	6.2335329	fac289115e6ffe703d8a6bb4601ed47b	2024-01-04	202
8 2024-01-05	112.3	15.5	7.2451613	4c86026dd0d340db57e7e225196b3c171	2024-01-05	202
9 2024-01-06	13.3	44.5	0.2988764	8424a0ba78144de793d13346a056ce71	2024-01-06	202
10 2024-01-07	113.8	4.9	23.2244898	aa8d6b10843a6cf201aa109fe2e3acb2	2024-01-07	202
11 2024-01-08	100.8	33.7	2.9910979	c3876fe4ce983cc9bea411acb03a9462	2024-01-08	202
12 2024-01-09	39.5	94.5	0.4179894	1e76308de2899d8cf593a28b98f31fa9	2024-01-09	202
13 2024-01-10	45.0	76.9	0.5851756	e7611d87857495097d202a072146169f	2024-01-10	202
14 2024-01-11	119.2	6.1	19.5409836	5b76e70bf0c08b325307108d610fc3eb	2024-01-11	202
15 2024-01-12	67.6	45.2	1.4955752	94c3dc954d43e4e0c6efa32d19831808	2024-01-12	202
16 2024-01-13	22.9	99.9	0.2292292	2015049de4a4423939ebd9500866f0cc	2024-01-13	202

Query Details
Query duration 10ms
Rows 20
Query ID 01b8c492-0004-386...
Show more

DATETIME

SOLARRADIATION

CLOUDCOVER

Ask Copilot

Fig 18d: Snapshot Solar Efficiency

81 | select * from WEATHER.SNAPSHOT.SNAPSHOT_PRESSURE_DROPS;

82

83

↳ Results ↵ Chart

Query Details

Query duration

Rows

Query ID 01b8c491

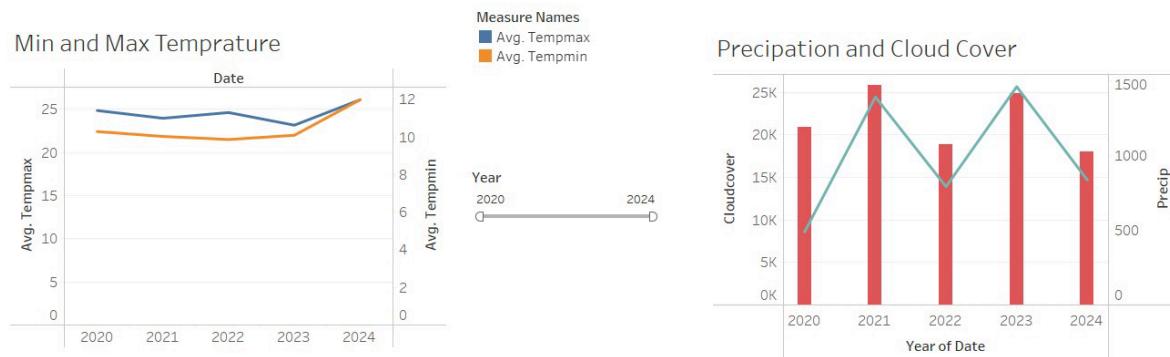
Show more ▾

	DATETIME	SEALEVELPRESSURE	PREV_PRESSURE	PRESSURE_CHANGE	PRESSURE_DROP	DBT_SCD_ID
1	2024-12-02	null	1021.5	0.2	No	818120f3e8e6e0e5c49c22dedc8e316c
2	2024-12-01	null	1021.4	-0.2	No	5ea9a2b3f39880df4acfe69c2a4edb4a
3	2024-11-30	null	null	null	No	274963eda8215ba4f5144f49522c2c42
4	2024-01-01	1021.4	null	null	No	21b70839345ea1da606dad68e66202a
5	2024-01-02	1020.8	1021.4	-0.6	No	e5af3c91a999bb1a729ad045b9e87371
6	2024-01-03	1020.7	1020.8	-0.1	No	8299b0af66e13ac86ef22dce83d6d167
7	2024-01-04	1024.7	1020.7	4.0	No	fac289115e6ffe703d8a6bb4601ed47b
8	2024-01-05	1025.3	1024.7	0.6	No	4c86026d0d340db57e7e225196b3c17
9	2024-01-06	1018.8	1025.3	-6.5	Yes	8424a0ba78144de793d13346a056ce7
10	2024-01-07	1019.6	1018.8	0.8	No	aa8d6b10843a6cf201aa109fe2e3acb2
11	2024-01-08	1026.2	1019.6	6.6	No	c3876fe4ce983cc9bea411acb03a9462
12	2024-01-09	1024.7	1026.2	-1.5	No	1e76308de2899d8cf593a28b98f31fa9
13	2024-01-10	1016.1	1024.7	-8.6	Yes	e7611d87857495097d202a072146168
14	2024-01-11	1020.5	1016.1	4.4	No	5b76e70bf0c08b325307108d610fc3et
15	2024-01-12	1024.3	1020.5	3.8	No	94c3dc954d43e4e0c6efa32d19831808
16	2024-01-13	1019.7	1024.3	-4.6	Yes	2015049de4a4423939ebd9500866f0c

Fig 18e: Snapshot Pressure Drops

VII. Data Visualization

California Weather Analysis for 2020 to 2024



Summary of the Day

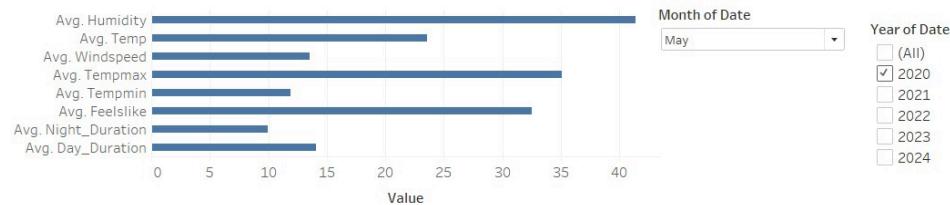


Figure 19: Historical data analysis dashboard

The dashboard contains three distinct visualizations of California's weather data:

1. Min and Max Temperature Chart

- Shows a gradual convergence of maximum and minimum temperatures from 2020 to 2024, with both lines meeting around 2024 at approximately 25°C
- Maximum temperatures remain relatively stable around 25°C while minimum temperatures show an upward trend from about 10°C to 12°C

2. Precipitation and Cloud Cover Chart

- Displays a cyclical pattern with peak precipitation levels in 2021 and 2023, shown by red bars reaching approximately 25K units
- Cloud cover (blue line) correlates with precipitation patterns, showing highest coverage during the same peak years and declining in 2024

3. Summary of the Day Chart

- Presents eight key weather metrics for May 2020, with Average Humidity showing the highest value at around 40 units
- Shows the relationship between day/night duration and temperature metrics, with average temperature sitting at approximately 25 units while day duration is about 15 units

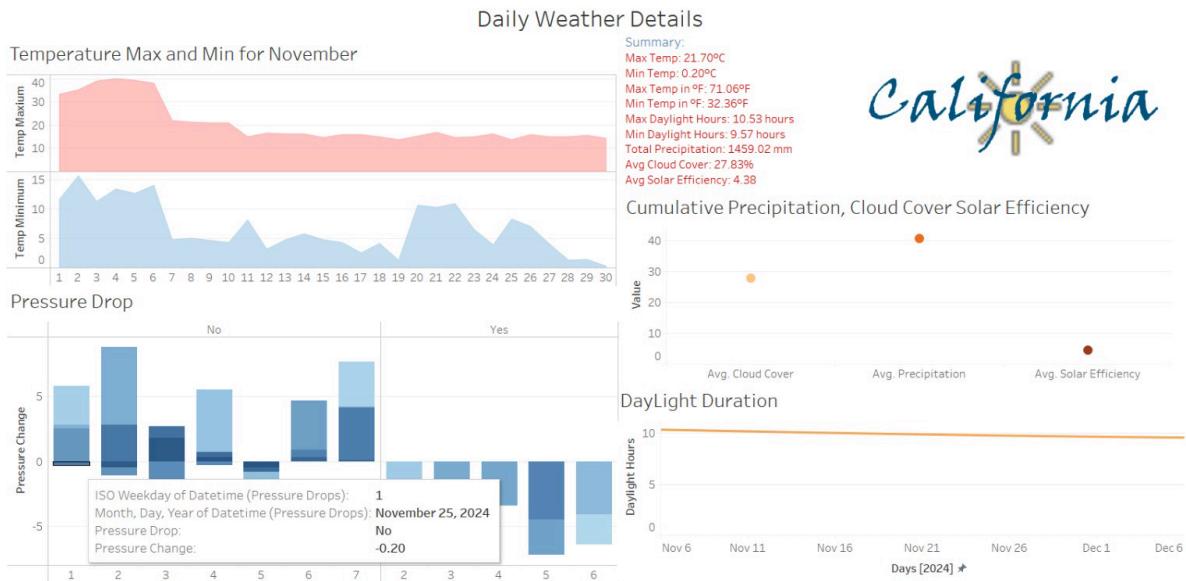


Figure 20: Recent data analysis dashboard

The dashboard displays current weather data for California with four key visualizations:

1. Temperature Max and Min for November

- Shows daily temperature variations with maximum temperatures (pink area) ranging from 40°C to 15°C, with a notable drop after day 7

- Minimum temperatures (blue area) fluctuate between 0-15°C, showing several peaks around days 3-5 and 20-22
2. Pressure Drop Analysis
 - Displays pressure changes categorized as "Yes" and "No", with the largest negative pressure change of -0.20 recorded on November 25, 2024
 - Shows more frequent pressure variations in the first week of the month, with smaller fluctuations in subsequent weeks
 3. Cumulative Precipitation, Cloud Cover & Solar Efficiency
 - Reveals current measurements with cloud cover at 27.83%, total precipitation at 1459.02mm, and solar efficiency at 4.38
 - Demonstrates the relationship between these metrics with precipitation showing the highest value among the three parameters
 4. Daylight Duration Trend
 - Shows a gradual decrease in daylight hours from 10.53 (maximum) to 9.57 (minimum) hours from early November to early December
 - Displays a consistent downward trend in daylight hours, typical of the approaching winter season

VIII. Results and Discussion

The implemented pipeline successfully processes daily weather data, calculates key metrics, and presents them in an easily digestible format. The use of Airflow ensures reliable scheduling and execution of both ETL and ELT processes, while dbt enables version-controlled, modular data transformations.

Scalability Considerations:

- The pipeline can easily accommodate additional locations by adding them to the ETL process.
- Snowflake's scalability allows for efficient processing of large volumes of historical weather data.

Limitations and future improvements:

- The current implementation focuses on daily data. Incorporating hourly could provide more granular insights.
- Additional weather metrics and predictive models could be incorporated to provide more comprehensive analysis and forecasting capabilities.

IX. GitHub repository link

https://github.com/patilgayatri22/DataWarehouse_FinalProject/tree/main/DataWarehouseProject

X. Conclusion

This project demonstrates the successful integration of modern data tools to create an end-to-end analytics pipeline for weather data. By leveraging Snowflake, Airflow, dbt, and Tableau, we've created a scalable, maintainable solution that can provide valuable insights for weather analysis and its impact on various sectors.

The modular nature of the pipeline allows for easy expansion and modification, making it adaptable to changing requirements and additional data sources. This approach to data analytics can be applied to various domains beyond weather analysis, showcasing the versatility of the chosen tools and architecture.

Acknowledgement

The Data Warehouse project of building an end-to-end Weather Data Analytics using Snowflake, Airflow, dbt, Tableau was an enriching experience because of the guidance and support of professor Keeyong Hun and TA Revanth Kumar Bondada. We would like to extend our gratitude towards them for their insights and expertise which turned out to be instrumental in helping us navigate the complexities of the project. Thank you for fostering a collaborative and engaging learning environment that encouraged us to explore and innovate. We greatly appreciate your dedication and encouragement.

References

- [1] Professor Keeyong Hun, DATA226 Lecture Notes Week 8, Week 9, Week 10, Week 11
- [2] Visual Crossing Weather Data Services API as data source,
<https://www.visualcrossing.com/weather/weather-data-services>