

# HW 3.2: dbt, Airflow, and Snowflake

Group 5:

- Anushka Rajesh Khadatkar – 018383963
  - Kanika Mamgain – 018319704
  - Samruddhi Suresh Chitnis – 018452122
  - Soroor Ghandali – 018286281
- 

## 1. Set up the development environment:

Successfully installed Python, Apache Airflow, DBT, and the snowflake connector.

```
[simmi@Anushkas-MacBook-Air ~ % source ~/hw3.2_pipeline/venv/bin/activate
(venv) simmi@Anushkas-MacBook-Air ~ % pip list | grep -E "airflow|dbt|snowflake"
apache-airflow                      2.9.3
apache-airflow-providers-common-io    1.3.2
apache-airflow-providers-common-sql   1.14.2
apache-airflow-providers-fab          1.2.2
apache-airflow-providers-ftp          3.10.0
apache-airflow-providers-http         4.12.0
apache-airflow-providers-imap        3.6.1
apache-airflow-providers-smtp        1.7.1
apache-airflow-providers-snowflake   5.6.0
apache-airflow-providers-sqlite       3.8.1
dbt-adapters                         1.9.0
dbt-common                           1.12.0
dbt-core                             1.8.7
dbt-extractor                        0.6.0
dbt-semantic-interfaces              0.5.1
dbt-snowflake                         1.8.3
snowflake-connector-python           3.11.0
snowflake sqlalchemy                  1.6.1
(venv) simmi@Anushkas-MacBook-Air ~ % python -V
[Python 3.11.14
(venv) simmi@Anushkas-MacBook-Air ~ % dbt --version
[Core:
  - installed: 1.8.7
  - latest:    1.10.13 - Update available!

  Your version of dbt-core is out of date!
  You can find instructions for upgrading here:
  https://docs.getdbt.com/docs/installation

Plugins:
  - snowflake: 1.8.3 - Update available!

  At least one plugin is out of date or incompatible with dbt-core.
  You can find instructions for upgrading here:
  https://docs.getdbt.com/docs/installation

(venv) simmi@Anushkas-MacBook-Air ~ % airflow version
[2.9.3
```

## Created a snowflake account.

## 2. Source a Dataset:

Selected Home Depot products dataset with 2,551 products stored in local filesystem.

Dataset link ("products\_prepared.csv") -

<https://drive.google.com/file/d/1cHH6MDRTEG32rh4iLxfz6JOVn-hlvgdW/view?usp=sharing>

Top 10 rows of our dataset -

```
((data_pipeline_venv) simmi@Anushkas-MacBook-Air ~ % cd ~/data_pipeline_project
((data_pipeline_venv) simmi@Anushkas-MacBook-Air data_pipeline_project % ls -lh data/
total 3112
-rw-r--r-- 1 simmi staff 1.5M 6 Nov 15:28 products.csv
((data_pipeline_venv) simmi@Anushkas-MacBook-Air data_pipeline_project % head -10 data/products.csv
id,name,category,price,description,updated_at
310899686,Men's 3X Large Carbon Heather Cotton/Polyester Rain Defender Paxton Heavyweight Hooded Zip-Front Sweatshirt,Carhartt,64.99,"This heavyweight, water-repellent hooded sweatshirt has a zip front for fast layering. ORIGINAL FIT. 13 oz., 75% cotton/25% polyester blend with Rain Defender durable water repellent. Attached, jersey-lined three-piece hood with drawcord closure. Antique-finish brass front zipper. Two front hand-warmer pockets have a hidden security pocket inside. Stretchable, spandex-reinforced rib-knit cuffs and waistband. Locker loop facilitates hanging.",2021-12-14 00:55:53+0000
206724580,Turmode 30 ft. RF TNC Female to RF TNC Male Adapter Cable,Unbranded,71.61,"If you need more length between your existing wireless device and Hi-Gain Antenna, this is the product for you. It's compatible with most Wi-Fi Antennas, so it is easy for you to extend your wireless network. Just replace your existing cable that runs between your wireless device and Antenna and you're ready to use your network with extended range.",2021-12-14 00:55:53+0000
310347105,Large Tapestry Bolster Bed,Carolina Pet Company,166.83,"Polyester cover resembling rich Italian tapestries wraps your pet in comfort and style. 100% recycled polyester high loft MemoryFiber fill keeps pets elevated off cold floors for relief on tired joints and pressure points. A great fit for pets up to 100 lbs. Zippered removable cover is machine washable in cold water only. To dry, use a delicate cycle or gentle setting with very low heat.",2021-12-14 00:55:53+0000
312338711,16-Gauge-Sinks Vessel Sink in White with Faucet,Unbranded,507.63,It features a rectangle shape. This vessel set is designed to be installed as a undermount vessel set. It is constructed with ceramic. This vessel set comes with an enamel glaze finish in White color. It is designed for a deck mount faucet.,2021-12-14 00:55:53+0000
308561619,Men's Crazy Horse 9" Logger Boot - Steel Toe - Black Size 10.5(W),Adtec,103.59,This 9 in. black full grain leather logger boot is Tough. With plain soft toe and rubber outsole it provides dependable traction with a comfortable fit. It is made with Goodyear Welt Construction so it is sturdy and secure.,2021-12-14 00:55:54+0000
312917264,Mariana 6 ft. Multi-Color 3-Panel Screen Divider,HomeRoots,550.24,"With robust structure and sophisticated canvas printing, this 3-panel screen is colorfully designed to be a great way to carve out space in your room or anywhere around the home. Not only will this elegant piece give you the needed exclusivity but it will also add inspiring style and beauty to your decor. This screen creatively features canvas print of a different and complementary images on each side. With its lightweight construction, our screen will definitely fill your space with inspiring visual flair for many Years to come.
3-panel screen finished on both sides Canvas printing.",2021-12-14 00:55:54+0000
316216346,5 gal. #650C-2 Powdery Mist Semi-Gloss Interior Paint,BEHR PRO,118.0,"BEHR PRO i300 Semi-Gloss Interior Paint has a sleek, radiant sheen appearance. This professional quality latex paint has superior hide and coverage, excellent sprayability, spray and back-roll, and superior touch-up. Use on properly prepared and primed drywall, concrete, masonry, wood and metal surfaces in both residential and commercial applications.",2021-12-14 00:55:54+0000
312456240,7/8 in. x 4-1/2 in. x 0.045 in. Metal and Stainless Cutting Wheel (50-Pack),DEWALT,44.97,DEWALT High Performance 0.045 in. Metal Cutting Wheels have a thin 0.045 in. cutting edge design for fast burr free cutting. They have a proprietary aluminum oxide grain combination for aggressive cutting action and the proprietary material mix ensures durable long life wheels. They have 2-full sheets of fiberglass for durability and user safety. These wheels are ideal for high performance cutting in all types of ferrous metals and stainless steel.,2021-12-14 00:55:54+0000
300980025,Ring Gold Bar Cart,Titan Lighting,278.0,This Ring Bar Cart is sure to make a statement in any room. This carts features a metal frame in gold and the two shelves are made of mirror with an antique finish. We offer a wide variety of furniture and decorative accessories to meet all of your decor needs.,2021-12-14 00:55:55+0000
(data_pipeline_venv) simmi@Anushkas-MacBook-Air data_pipeline_project %
```

### 3. Setup Snowflake:

Created a dedicated database and schema in snowflake and also defined staging tables to store raw product data.

- Create a database, and schema for the project -

The screenshot shows the Snowflake SQL editor interface. The top bar has tabs for 'Home' and 'Untitled.sql'. The main area displays the following SQL code:

```
1 -- Create database
2 CREATE DATABASE PRODUCT_CATALOG;
3
4 -- Verify database created
5 SHOW DATABASES LIKE 'PRODUCT_CATALOG';
6
7 -- Use the database
8 USE DATABASE PRODUCT_CATALOG;
9
10 -- Create schemas
11 CREATE SCHEMA STAGING;
12 CREATE SCHEMA CORE;
13 CREATE SCHEMA SNAPSHOT;
```

Below the code, a 'Results (just now)' section shows a table with 5 rows of schema information:

| Table | Column                        | name               | is_default | is_current | database_name   | owner        |
|-------|-------------------------------|--------------------|------------|------------|-----------------|--------------|
| 1     | created_on                    | CORE               | N          | N          | PRODUCT_CATALOG | ACCOUNTADMIN |
| 2     | 2025-11-06 20:49:53.053 -0800 | INFORMATION_SCHEMA | N          | N          | PRODUCT_CATALOG |              |
| 3     | 2025-11-06 20:49:52.096 -0800 | PUBLIC             | N          | N          | PRODUCT_CATALOG | ACCOUNTADMIN |
| 4     | 2025-11-06 20:49:53.289 -0800 | SNAPSHOTS          | N          | Y          | PRODUCT_CATALOG | ACCOUNTADMIN |
| 5     | 2025-11-06 20:49:52.779 -0800 | STAGING            | N          | N          | PRODUCT_CATALOG | ACCOUNTADMIN |

- Create a staging table to receive raw data from the source -

The screenshot shows the Snowflake SQL editor interface. The top bar has tabs for 'Home' and 'Untitled.sql'. The main area displays the following SQL code:

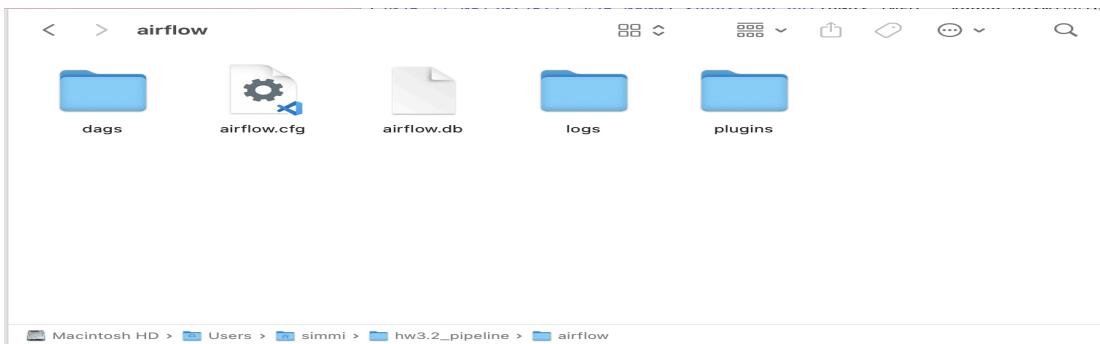
```
33     availability VARCHAR(100),
34     uniq_id VARCHAR(200),
35     scraped_at TIMESTAMP_NTZ,
36     created_at TIMESTAMP_NTZ,
37     updated_at TIMESTAMP_NTZ,
38     loaded_at TIMESTAMP_NTZ DEFAULT CURRENT_TIMESTAMP()
39 );
40
41 SHOW TABLES IN SCHEMA STAGING;
42
43 -- TABLE STRUCTURE
44 DESCRIBE TABLE RAW_PRODUCTS;
```

Below the code, a 'Results (just now)' section shows a table with 7 rows of column metadata:

| Table | Column      | name          | type   | kind | null? | default | primary key | unique key | check      |
|-------|-------------|---------------|--------|------|-------|---------|-------------|------------|------------|
| 1     | INDEX       | NUMBER(38,0)  | COLUMN | Y    | null  | N       | 100.0%      | N          | All values |
| 2     | URL         | VARCHAR(1000) | COLUMN | Y    | null  | N       |             | N          | null       |
| 3     | TITLE       | VARCHAR(1000) | COLUMN | Y    | null  | N       |             | N          | null       |
| 4     | IMAGES      | VARCHAR(5000) | COLUMN | Y    | null  | N       |             | N          | null       |
| 5     | DESCRIPTION | VARCHAR(5000) | COLUMN | Y    | null  | N       |             | N          | null       |
| 6     | PRODUCT_ID  | NUMBER(38,0)  | COLUMN | Y    | null  | N       |             | N          | null       |
| 7     | SKU         | FLOAT         | COLUMN | Y    | null  | N       |             | N          | null       |

#### 4. Implement Airflow DAG:

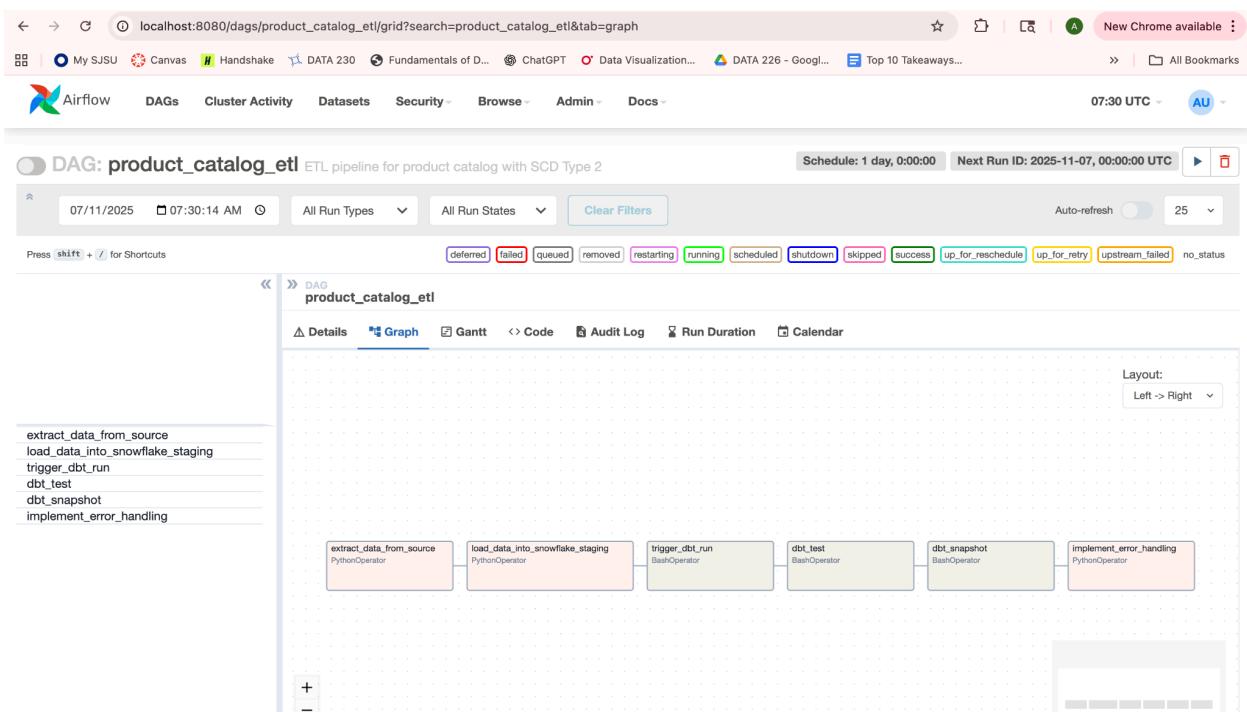
##### a. Create DAG file in the Airflow dags folder -



##### b. Tasks -

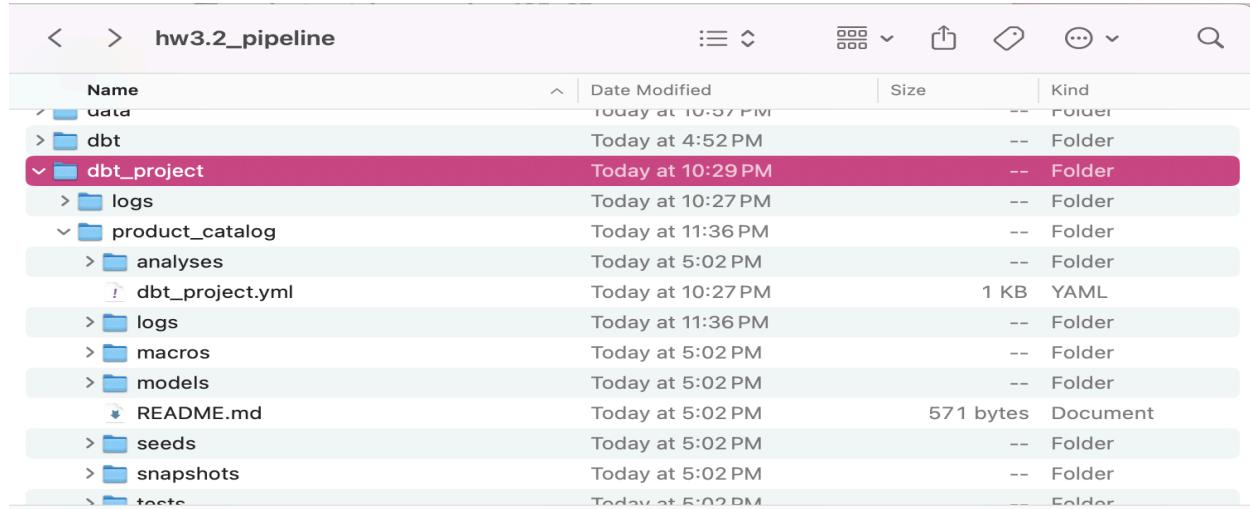
Built an airflow dag names “product\_catalog\_etl”

- Extracting data from the source: Fetch product catalog data from our source (i.e., local filesystem)
- Loading data into Snowflake staging table: loaded extracted data to the staging table in snowflake for processing in 6 batches with 500 rows per batch.
- Triggering dbt run: run dbt models to transform the data into the targeted schema
- Dbt test: runs dbt tests to validate the data and ensure transformations are correct
- Dbt snapshot: capture historical changes in dimension tables
- Error handling: logs or alerts for any task failure



## 5. Implement dbt models:

- Create a dbt project structure - Initialized dbt project names “product\_catalog” with standard folders (core, models, snapshots,tests)



The screenshot shows a file explorer window titled "hw3.2\_pipeline". Inside, there's a folder named "dbt\_project" which is expanded to show its contents: logs, product\_catalog, analyses, dbt\_project.yml, macros, models, README.md, seeds, snapshots, and tests. The "dbt\_project" folder is highlighted with a pink background.

```
product_catalog -- zsh -- 105x37
07:36:05 Using dbt_project.yml file at /Users/simmi/hw3.2_pipeline/dbt_project/product_catalog/dbt_project.yml
07:36:05 adapter type: snowflake
07:36:05 adapter version: 1.8.3
07:36:05 Configuration:
07:36:05   profiles.yml file [OK found and valid]
07:36:05   dbt_project.yml file [OK found and valid]
07:36:05 Required dependencies:
07:36:05   - git [OK found]

07:36:05 Connection:
07:36:05   account: WDCBXVZ-PKB12734
07:36:05   user: Anushkakhadatkar
07:36:05   database: PRODUCT_CATALOG
07:36:05   warehouse: COMPUTE_WH
07:36:05   role: ACCOUNTADMIN
07:36:05   schema: STAGING
07:36:05   authenticator: None
07:36:05   oauth_client_id: None
07:36:05   query_tag: None
07:36:05   client_session_keep_alive: False
07:36:05   host: None
07:36:05   port: None
07:36:05   proxy_host: None
07:36:05   proxy_port: None
07:36:05   protocol: None
07:36:05   connect_retries: 1
07:36:05   connect_timeout: None
07:36:05   retry_on_database_errors: False
07:36:05   retry_all: False
07:36:05   insecure_mode: False
07:36:05   reuse_connections: None
07:36:05 Registered adapter: snowflake=1.8.3
07:36:06   Connection test: [OK connection ok]

07:36:06 All checks passed!
(venv) simmi@Anushka-MacBook-Air product_catalog %
```

- Implement a staging model to clean and prepare the data - Implemented a staging model to clean and standardize raw data before transformation. This ensures consistent field names, dropping of null values.

```

stg_products.sql ×
Users > simmi > hw3.2_pipeline > dbt_project > product_catalog > models > staging > stg_products.sql
1   config(
2     materialized='view',
3     schema='staging'
4   )
5 }
6
7
8 -- Staging model: Clean and prepare raw product data
9 -- This model standardizes data types, handles nulls, and applies business rules
10
11 WITH source_data AS (
12   SELECT * FROM {{ source('staging', 'raw_products') }}
13 ),
14
15 cleaned_data AS (
16   SELECT
17     -- Primary identifiers
18     product_id,
19     uniq_id,
20     sku,
21     gtin13,
22
23     -- Product information
24     TRIM(title) AS title,
25     TRIM(COALESCE(NULLIF(brand, ''), 'Unbranded')) AS brand,
26     CASE
27       WHEN brand IS NULL OR brand = '' THEN 'Unbranded'
28       ELSE TRIM(brand)
29     END AS category,
30     TRIM(description) AS description,
31     TRIM(url) AS url,
32
33     -- Pricing and availability
34     CAST(price AS DECIMAL(10,2)) AS price,
35     UPPER(TRIM(currency)) AS currency,
36     TRIM(availability) AS availability,
37
38     -- Timestamps
39     created_at,
40     updated_at,
41     loaded_at
42
43   FROM source_data
44   WHERE product_id IS NOT NULL
45     AND price IS NOT NULL
46     AND price > 0
47     AND title IS NOT NULL
48
49 )
50   SELECT * FROM cleaned_data

```

```

stg_products.sql ! schema.yml ×
Users > simmi > hw3.2_pipeline > dbt_project > product_catalog > models > staging > ! schema.yml
1 version: 2
2
3 sources:
4   - name: staging
5     description: "Raw staging data from source systems"
6     database: PRODUCT_CATALOG
7     schema: STAGING
8     tables:
9       - name: raw_products
10      description: "Raw product data loaded from CSV"
11      columns:
12        - name: product_id
13          description: "Unique product identifier"
14          tests:
15            - not_null
16        - name: price
17          description: "Product price in local currency"
18          tests:
19            - not_null
20        - name: title
21          description: "Product title/name"
22
23 models:
24   - name: stg_products
25     description: "Cleaned and standardized product data from staging"
26     columns:
27       - name: product_id
28         description: "Unique product identifier"
29         tests:
30           - unique
31           - not_null
32       - name: title
33         description: "Product title"
34         tests:
35           - not_null
36       - name: price
37         description: "Product price"
38         tests:
39           - not_null
40       - name: updated_at
41         description: "Last update timestamp"
42         tests:
43           - not_null
44

```

- c. Create a core model that implements SCD Type 2 for the product dimension - developed this to implement SCD Type 2 logic using LAG and LEAD functions to track historical changes.

```
stg_products.sql      dim_products_scd2.sql X
Users > simmi > hw3.2_pipeline > dbt_project > product_catalog > models > core > dim_products_scd2.sql
40     changed_records AS (
41         -- Identify records that are new or have changed
42         SELECT
43             s.*,
44             e.surrogate_key AS existing_surrogate_key,
45             e.valid_from AS existing_valid_from,
46             CASE
47                 WHEN e.product_id IS NULL THEN 'INSERT' -- New product
48                 WHEN s.title != e.title OR
49                     s.price != e.price OR
50                     s.description != e.description OR
51                     s.category != e.category THEN 'UPDATE' -- Changed attributes
52                 ELSE 'NO_CHANGE'
53             END AS change_type
54         FROM source_data s
55         LEFT JOIN existing_records e
56             ON s.product_id = e.product_id
57     ),
58
59     records_to_close AS (
60         -- Close out the old versions of changed records
61         SELECT
62             e.surrogate_key,
63             e.product_id,
64             e.title,
65             e.category,
66             e.brand,
67             e.price,
68             e.description,
69             e.url,
70             e.availability,
71             e.currency,
72             e.created_at,
73             e.updated_at,
74             e.valid_from,
75             c.updated_at AS valid_to, -- Set end date to new record's timestamp
76             FALSE AS is_current, -- Mark as historical
77             e.uniq_id
78         FROM existing_records e
79         INNER JOIN changed_records c
80             ON e.product_id = c.product_id
81         WHERE c.change_type = 'UPDATE'
82     ),
83
84     new_versions AS (
85         -- Create new versions for changed and new records
86         SELECT
87             {{ dbt_utils.generate_surrogate_key(['product_id', 'updated_at']) }} AS surrogate_key,
88             product_id,
89             title,
90             category
```

```

stg_products.sql      ! schema.yml ×
Users > simmi > hw3.2_pipeline > dbt_project > product_catalog > models > core > ! schema.yml
1   version: 2
2
3   models:
4     - name: dim_products_scd2
5       description: "Product dimension table with SCD Type 2 for tracking historical changes"
6       columns:
7         - name: surrogate_key
8           description: "Surrogate key combining product_id and timestamp"
9           tests:
10          - unique
11          - not_null
12
13         - name: product_id
14           description: "Natural business key for the product"
15           tests:
16             - not_null
17
18         - name: title
19           description: "Product title/name"
20           tests:
21             - not_null
22
23         - name: price
24           description: "Product price"
25           tests:
26             - not_null
27             - dbt_utils.accepted_range:
28               min_value: 0
29               inclusive: false
30
31         - name: valid_from
32           description: "Start date when this version became active"
33           tests:
34             - not_null
35
36         - name: is_current
37           description: "Flag indicating if this is the current active record"
38           tests:
39             - not_null
40             - accepted_values:
41               values: [true, false]
42
43         - name: category
44           description: "Product category"
45           tests:
46             - not_null
47
48         - name: currency
49           description: "Price currency code"
50           tests:

```

Restricted Mode ⌂ ⌂ ⌂ ⌂

- d. Write appropriate tests for your models - added tests to ensure primary key uniqueness, and correct SCD behaviour.

```

Users > simmi > hw3.2_pipeline > dbt_project > product_catalog > tests > test_no_overlapping_dates.sql
1  |-- Test: Ensure no overlapping date ranges for the same product
2  -- Each product should have non-overlapping valid_from/valid_to ranges
3
4  WITH date_ranges AS (
5      SELECT
6          product_id,
7          valid_from,
8          COALESCE(valid_to, '9999-12-31'::TIMESTAMP_NTZ) AS valid_to
9      FROM {{ ref('dim_products_scd2') }}
10 )
11
12 SELECT
13     a.product_id,
14     a.valid_from AS range1_start,
15     a.valid_to AS range1_end,
16     b.valid_from AS range2_start,
17     b.valid_to AS range2_end
18 FROM date_ranges a
19 JOIN date_ranges b
20     ON a.product_id = b.product_id
21     AND a.valid_from < b.valid_from
22 WHERE a.valid_to > b.valid_from

```

```

Users > simmi > hw3.2_pipeline > dbt_project > product_catalog > tests > test_scd2_integrity.sql
1  |-- Test: Ensure each product has exactly one current record
2  -- This test validates SCD Type 2 implementation
3
4  SELECT
5      product_id,
6      COUNT(*) as current_count
7  FROM {{ ref('dim_products_scd2') }}
8  WHERE is_current = TRUE
9  GROUP BY product_id
10 HAVING COUNT(*) > 1

```

## 6. Configure dbt snapshots:

Configured dbt snapshots to capture changes in the data over time.

```

product_snapshot.sql ×
Users > simmi > hw3.2_pipeline > dbt_project > product_catalog > snapshots > product_snapshot.sql
1  {% snapshot product_snapshot %}
2
3  {{{
4      config(
5          target_schema='snapshots',
6          unique_key='product_id',
7          strategy='timestamp',
8          updated_at='updated_at',
9          invalidate_hard_deletes=True,
10     )
11 }}}
12
13  -- Snapshot of product data to track historical changes
14  -- This captures point-in-time product information
15
16  SELECT
17      product_id,
18      title,
19      brand,
20      category,
21      price,
22      currency,
23      description,
24      url,
25      availability,
26      updated_at,
27      uniq_id
28  FROM {{ ref('stg_products') }}
29
30  {% endsnapshot %}

```

## 7. Integrate dbt with Airflow:

Integrated dbt into Airflow by running dbt run, dbt test, and dbt snapshot via BashOperator tasks in the DAG.

### Dbt run -

```
(venv) simmi@Anushkas-MacBook-Air product_catalog % dbt run
08:29:46  Running with dbt=1.8.7
08:29:47  Registered adapter: snowflake=1.8.3
08:29:48  Found 4 models, 24 data tests, 1 snapshot, 1 source, 573 macros
08:29:48
08:29:50  Concurrency: 4 threads (target='dev')
08:29:50
08:29:50  1 of 4 START sql table model STAGING.my_first_dbt_model ..... [RUN]
08:29:50  2 of 4 START sql view model STAGING_staging.stg_products ..... [RUN]
08:29:51  2 of 4 OK created sql view model STAGING_staging.stg_products ..... [SUCCESS 1 in 0.91s]
08:29:51  3 of 4 START sql incremental model STAGING_core.dim_products_scd2 ..... [RUN]
08:29:51  1 of 4 OK created sql table model STAGING.my_first_dbt_model ..... [SUCCESS 1 in 1.34s]
08:29:51  4 of 4 START sql view model STAGING.my_second_dbt_model ..... [RUN]
08:29:53  4 of 4 OK created sql view model STAGING.my_second_dbt_model ..... [SUCCESS 1 in 1.37s]
08:29:55  3 of 4 OK created sql incremental model STAGING_core.dim_products_scd2 ..... [SUCCESS 790 in 4.41s]
08:29:55
08:29:55  Finished running 2 view models, 1 table model, 1 incremental model in 0 hours 0 minutes and 7.52 second
s (7.52s).
08:29:55
08:29:55  Completed successfully
08:29:55
```

### Dbt test -

```
(venv) simmi@Anushkas-MacBook-Air product_catalog % dbt test
08:30:14  Running with dbt=1.8.7
08:30:15  Registered adapter: snowflake=1.8.3
08:30:15  Found 4 models, 24 data tests, 1 snapshot, 1 source, 573 macros
08:30:15
08:30:17  Concurrency: 4 threads (target='dev')
08:30:17
08:30:17  1 of 24 START test accepted_values_dim_products_scd2_currency__USD__EUR__GBP__CAD  [RUN]
08:30:17  3 of 24 START test dbt_utils_accepted_range_dim_products_scd2_price__False__0 .. [RUN]
08:30:17  4 of 24 START test not_null_dim_products_scd2_category ..... [RUN]
08:30:17  2 of 24 START test accepted_values_dim_products_scd2_is_current__True__False .. [RUN]
08:30:18  4 of 24 PASS not_null_dim_products_scd2_category ..... [PASS in 1.48s]
08:30:18  5 of 24 START test not_null_dim_products_scd2_is_current ..... [RUN]
08:30:19  1 of 24 PASS accepted_values_dim_products_scd2_currency__USD__EUR__GBP__CAD .. [PASS in 2.31s]
08:30:19  6 of 24 START test not_null_dim_products_scd2_price ..... [RUN]
08:30:19  2 of 24 PASS accepted_values_dim_products_scd2_is_current__True__False .. [PASS in 2.46s]
08:30:19  3 of 24 PASS dbt_utils_accepted_range_dim_products_scd2_price__False__0 .. [PASS in 2.52s]
08:30:19  7 of 24 START test not_null_dim_products_scd2_product_id ..... [RUN]
08:30:19  5 of 24 PASS not_null_dim_products_scd2_is_current ..... [PASS in 1.05s]
08:30:19  8 of 24 START test not_null_dim_products_scd2_surrogate_key ..... [RUN]
08:30:19  9 of 24 START test not_null_dim_products_scd2_title ..... [RUN]
08:30:20  6 of 24 PASS not_null_dim_products_scd2_price ..... [PASS in 0.95s]
08:30:20  10 of 24 START test not_null_dim_products_scd2_valid_from ..... [RUN]
08:30:20  7 of 24 PASS not_null_dim_products_scd2_product_id ..... [PASS in 0.82s]
08:30:20  11 of 24 START test not_null_my_first_dbt_model_id ..... [RUN]
08:30:20  9 of 24 PASS not_null_dim_products_scd2_title ..... [PASS in 0.83s]
08:30:20  12 of 24 START test not_null_my_second_dbt_model_id .. [RUN]
08:30:21  8 of 24 PASS not_null_dim_products_scd2_surrogate_key ..... [PASS in 1.29s]
08:30:21  13 of 24 START test not_null_stg_products_price ..... [RUN]
08:30:21  10 of 24 PASS not_null_dim_products_scd2_valid_from ..... [PASS in 1.02s]
```

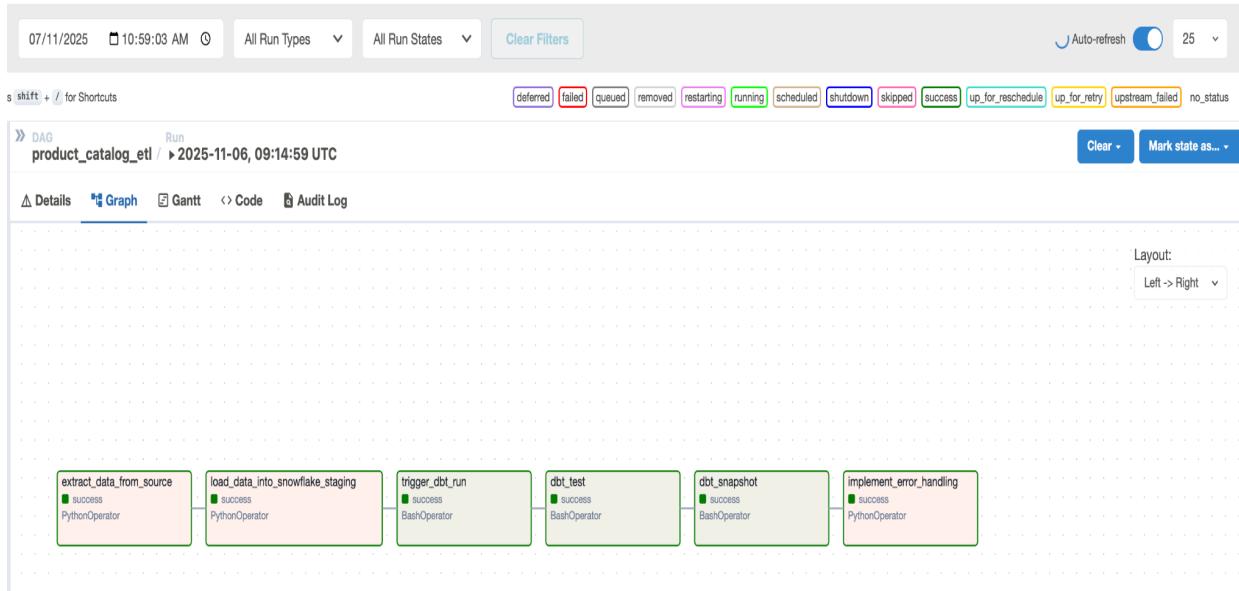
### Dbt snapshot -

```
(venv) simmi@Anushkas-MacBook-Air product_catalog % dbt snapshot
08:31:35  Running with dbt=1.8.7
08:31:36  Registered adapter: snowflake=1.8.3
08:31:37  Found 4 models, 24 data tests, 1 snapshot, 1 source, 573 macros
08:31:37
08:31:38  Concurrency: 4 threads (target='dev')
08:31:38
08:31:38  1 of 1 START snapshot snapshots.product_snapshot ..... [RUN]
08:31:44  1 of 1 OK snapshotted snapshots.product_snapshot ..... [SUCCESS 636 in 5.52s]
08:31:44
08:31:44  Finished running 1 snapshot in 0 hours 0 minutes and 7.44 seconds (7.44s).
08:31:44
08:31:44  Completed successfully
08:31:44
```

## 8. Run entire pipeline:

Executed “product\_catalog\_etl” DAG in the Airflow

### a. Run Airflow DAG - all tasks completed successfully



### b. Verify data in Snowflake - initialised query in snowflake to confirm transformed product records were loaded correctly.

The screenshot shows the Snowflake SQL editor with a query named 'Untitled 1.sql'. The code is as follows:

```
39 -- 7. Verify data quality
40 SELECT
41     'Total Records' as metric,
42     COUNT(*) as value
43 FROM PUBLIC_CORE.DIM_PRODUCTS_SCD2
44 UNION ALL
45 SELECT
46     'Current Records',
47     COUNT(*)
48 FROM PUBLIC_CORE.DIM_PRODUCTS_SCD2
49 WHERE IS_CURRENT = TRUE
50 UNION ALL
51 SELECT
52     'Historical Records',
53     COUNT(*)
54 FROM PUBLIC_CORE.DIM_PRODUCTS_SCD2
55 WHERE IS_CURRENT = FALSE
56 UNION ALL
57 SELECT
58     'Unique Products',
59     COUNT(DISTINCT PRODUCT_ID)
60 FROM PUBLIC_CORE.DIM_PRODUCTS_SCD2;
```

Below the code, the 'Results' section shows the output of the query:

| METRIC          | VALUE |
|-----------------|-------|
| Total Records   | 2551  |
| Current Records | 2551  |

Home SQL Untitled.sql SQL Untitled 1.sql +

My Workspace > Untitled 1.sql

```

1 -- Query 1: All table counts
2
3     'STAGING.RAW_PRODUCTS' as table_location,
4     COUNT(*) as record_count
5 FROM STAGING.RAW_PRODUCTS
6 UNION ALL
7     SELECT
8         'PUBLIC_CORE.DIM_PRODUCTS_SCD2 (Total)',
9         COUNT(*)
10    FROM PUBLIC_CORE.DIM_PRODUCTS_SCD2
11 UNION ALL
12     SELECT
13         'PUBLIC_CORE.DIM_PRODUCTS_SCD2 (Current)',
14         COUNT(*)
15    FROM PUBLIC_CORE.DIM_PRODUCTS_SCD2
16 WHERE IS_CURRENT = TRUE
17 UNION ALL
18     SELECT
19         'PUBLIC_CORE.DIM_PRODUCTS_SCD2 (Historical)'

```

Results (just now)

|     | Table                                      | Chart        |
|-----|--|--------------|
| 000 | TABLE_LOCATION                             | RECORD_COUNT |
| 1   | STAGING.RAW_PRODUCTS                       | 2551         |
| 2   | PUBLIC_CORE.DIM_PRODUCTS_SCD2 (Total)      | 2551         |
| 3   | PUBLIC_CORE.DIM_PRODUCTS_SCD2 (Current)    | 2551         |
| 4   | PUBLIC_CORE.DIM_PRODUCTS_SCD2 (Historical) | 0            |

There is zero count for the historical table as we haven't changed our product details yet.

Home SQL Untitled.sql SQL Untitled 1.sql +

My Workspace > Untitled 1.sql

```

1 -- Query 2: View sample product data
2
3     SURROGATE_KEY,
4     PRODUCT_ID,
5     TITLE,
6     PRICE,
7     CATEGORY,
8     VALID_FROM,
9     VALID_TO,
10    IS_CURRENT
11   FROM PUBLIC_CORE.DIM_PRODUCTS_SCD2

```

Results (just now)

|     | Table                            | Chart            |   |        |
|-----|----------------------------------|------------------|---|--------|
| 000 | SURROGATE_KEY                    | PRODUCT_ID       | TITLE   | PRICE  |
| 1   | 001fd61bca49700340181...         | 20564...31734... | #38 High Speed Steel General Purpose Blac...                | 10.0%  |
| 2   | 15565a2a0f84bb5eb76a87...        | 311557356        | 38 in. x 42 in. Octagonal Surface Mount PVC...              | 10.0%  |
| 3   | cb07f6beba028eebe9ae61812777c34  | 315969462        | Lanikai by Colossal Images Canvas Wall Art 18 in. x 24 in   | 49.99  |
| 4   | 46c4151d7db6fb8a3bae4dc00df53720 | 301191569        | Torx Dual Material Screwdriver Set (3-Piece)                | 29.99  |
| 5   | 36376fc9432c2d64ec90a74af507536  | 308416325        | 5/8 in. x 40 in. x 7-1/8 in. Polyurethane Standard Crossh   | 82.00  |
| 6   | e0747cea83057158977e8e4bcaeccc5  | 205773414        | Snow Drift/Shadow White Cordless Day and Night Blackc       | 131.97 |
| 7   | 214c6d74dac27db901cec788af08d3   | 306049965        | 5.5 in. x 36 in. x 36 in. Douglas Fir Traditional Smooth Br | 259.83 |
| 8   | 15565a2a0f84bb5eb76a87fa8a1bbba1 | 317342351        | #38 High Speed Steel General Purpose Black Oxide Twis       | 6.23   |
| 9   | 624e53a5b3e11ff5bf2702a5860d97   | 205641398        | Custom Size Stair Treads Solid Brown 11.5" x 31.5" Indo     | 173.91 |
| 10  | 6ac4adc01b93004f1422ee866ee39e3  | 311557356        | Chelsea Ivory/Camel 2 ft. x 3 ft. Border Area Runn          | 26.58  |

Home SQL Untitled.sql SQL Untitled 1.sql +

My Workspace > Untitled 1.sql

```

1 -- Query 3: Check snapshot (if exists)
2     SELECT COUNT(*) as snapshot_count
3     FROM SNAPSHOTs.PRODUCT_SNAPSHOT;

```

Results (just now)

|     | Table          | Chart |
|-----|----------------|-------|
| 000 | SNAPSHOT_COUNT |       |
| 1   |                | 2551  |

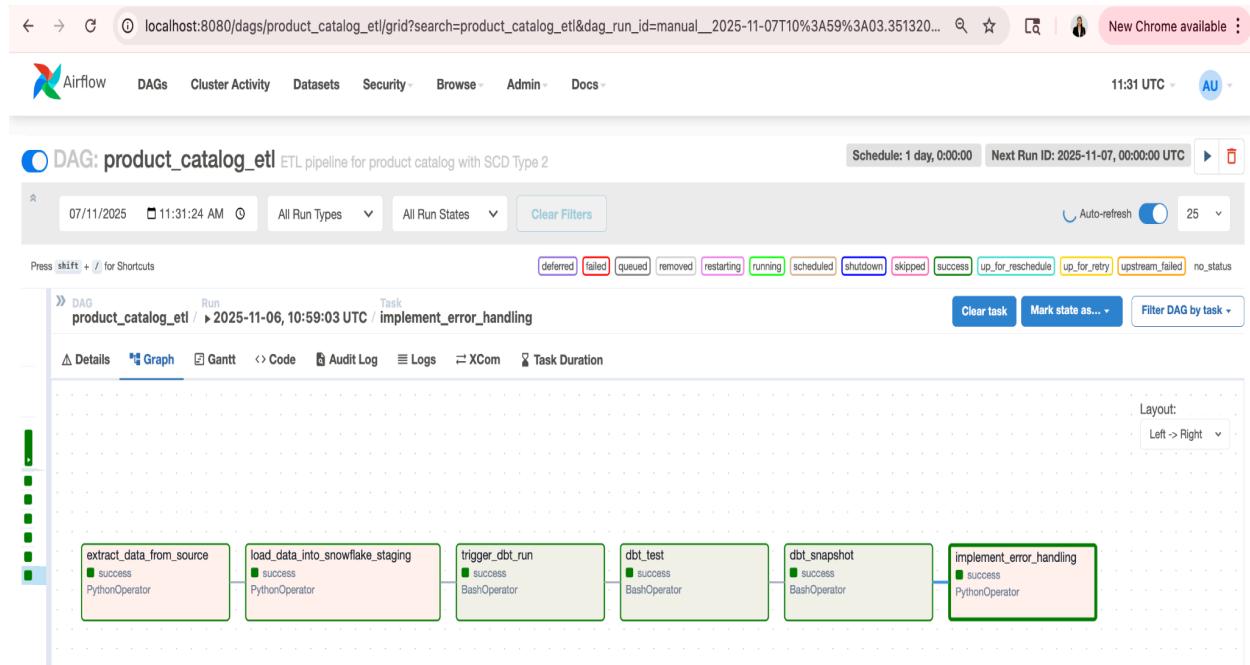
- c. Make changes to the source data and re-run the pipeline to test SCD implementation

We modified 20 product records and re-ran the pipeline to get historical changes.

```
product_catalog -- zsh -- 80x24
Product 304590104: $39.47 → $47.72 (+20.9%)
Product 317017153: $30.74 → $37.44 (+21.8%)
Product 309275017: $40.73 → $44.86 (+10.1%)
Product 311568667: $31.67 → $39.26 (+24.0%)
Product 318858902: $165.14 → $186.79 (+13.1%)
Product 202760200: $37.52 → $43.80 (+16.7%)
Product 205257192: $44.75 → $50.09 (+11.9%)
Product 303337016: $36.09 → $44.06 (+22.1%)
Product 307341453: $228.77 → $285.04 (+24.6%)
Product 100650630: $440.16 → $569.84 (+29.5%)
Product 312732949: $244.45 → $295.88 (+21.0%)
Product 204955163: $44.03 → $53.88 (+22.4%)
Product 308687923: $514.56 → $625.43 (+21.5%)
Product 314920993: $131.03 → $145.33 (+10.9%)
Product 204909099: $231.58 → $268.14 (+15.8%)
Product 317797313: $132.98 → $152.47 (+14.7%)
Product 314372149: $413.70 → $478.07 (+15.6%)
Product 309600308: $64.27 → $75.39 (+17.3%)
Product 205237692: $251.22 → $286.87 (+14.2%)

=====
✓ Successfully modified 20 products!
✓ Updated file: /Users/simmi/hw3.2_pipeline/data/data/products_prepared.csv
```

All tasks in the “product\_catalog\_etl” for SCD Type 2 ran successfully.



Verified the historical entries per product ID using snowflake, validating SCD Type 2 behaviour.

Home Untitled.sql Untitled 1.sql +

My Workspace > Untitled 1.sql

Share ...

```

1 -- Query 1: Find products with multiple versions (history)
2   SELECT
3     PRODUCT_ID,
4     COUNT(*) AS version_count
5   FROM PUBLIC_CORE.DIM_PRODUCTS_SCD2
6   GROUP BY PRODUCT_ID
7   HAVING COUNT(*) > 1
8   ORDER BY version_count DESC
9   LIMIT 10;
10
11

```

Results (just now)

Table Chart

Query History

| # | PRODUCT_ID | VERSION_COUNT |
|---|------------|---------------|
| 1 | 100650630  | 2             |
| 2 | 309600308  | 2             |
| 3 | 311568667  | 2             |
| 4 | 204955163  | 2             |
| 5 | 312732949  | 2             |
| 6 | 314372149  | 2             |
| 7 | 202760200  | 2             |
| 8 | 307341453  | 2             |

Home Untitled.sql Untitled 1.sql +

My Workspace > Untitled 1.sql

Share ...

```

1 -- Query 2: View complete history for changed products
2   WITH products_with_history AS (
3     SELECT PRODUCT_ID
4     FROM PUBLIC_CORE.DIM_PRODUCTS_SCD2
5     GROUP BY PRODUCT_ID
6     HAVING COUNT(*) > 1
7     LIMIT 5
8   )
9   SELECT
10    p.PRODUCT_ID,
11    p.TITLE,

```

Results (just now)

Table Chart

Query History

| # | PRODUCT_ID       | TITLE  | PRICE  | VALID_FROM              | VALID_TO              |                       |
|---|------------------|--|--|-------------------------|-----------------------|-----------------------|
| 1 | 30333...31273... | Cucina Daisy Ivory 4 ft. x 4 ft. Kitchen Mat<br>8.5 in. Antique White Daisy the Muggly Face St...<br>+7 more | 20.0%<br>8.5 in. Antique White Daisy the Muggly Face Statu...<br>+7 more | 29...51...              | 13/11/2021 06/11/2025 | 2025-11-01 2025-11-01 |
| 2 | 303337016        | 8.5 in. Antique White Daisy the Muggly Face Statue Planter   | 29.56  | 2021-11-14 00:57:10.693 | 2025-11-01            |                       |
| 3 | 303337016        | Updated - 8.5 in. Antique White Daisy the Muggly Face Statue Planter   | 36.09  | 2025-11-07 02:26:11.625 | null                  |                       |
| 4 | 307341453        | Kontur Wood 72 in. Single Traverse Rod Set in Wenge with Brackets  | 183.61   | 2021-11-14 01:04:43.173 | 2025-11-01            |                       |
| 5 | 308687923        | Updated - Kontur Wood 72 in. Single Traverse Rod Set in Wenge with Brackets                                  | 228.77   | 2025-11-07 02:26:11.625 | null                  |                       |
| 6 | 308687923        | Margie Tribal Fringe Black 8 ft. x 10 ft. Area Rug   | 423.34   | 2021-11-14 00:57:16.234 | 2025-11-01            |                       |
| 7 | 309600308        | Updated - Margie Tribal Fringe Black 8 ft. x 10 ft. Area Rug   | 514.56   | 2025-11-07 02:26:11.626 | null                  |                       |
| 8 | 309600308        | Cucina Daisy Ivory 4 ft. x 4 ft. Kitchen Mat   | 54.79  | 2021-11-14 00:56:20.492 | 2025-11-01            |                       |
|   |                  | Cucina Daisy Ivory 4 ft. x 4 ft. Kitchen Mat   | 64.27  | 2025-11-07 02:26:11.626 | null                  |                       |

Home Untitled.sql Untitled 1.sql + ⚙

My Workspace > Untitled 1.sql

Play ⏪ ⏹ Share ...

```

1 -- Query 3: Show price changes over time
2 WITH ranked_prices AS (
3     SELECT
4         PRODUCT_ID,
5         TITLE,
6         PRICE,
7         VALID_FROM,
8         IS_CURRENT,
9         LAG(PRICE) OVER (PARTITION BY PRODUCT_ID ORDER BY VALID_FROM) as previous_price
10    FROM PUBLIC_CORE.DIM_PRODUCTS_SCD2
11 )

```

Results (just now)

Table Chart

Chart type Bar chart ▾

X-axis ▲ TITLE ▾

Sort None ▾

Y-axis # PRODUCT\_ID ▾

+ Add column

Aggregate Sum ▾

Group by None ▾

Sum of PRODUCT\_ID

| TITLE                               | Sum of PRODUCT_ID |
|-------------------------------------|-------------------|
| 13/16 in. x 1 in. High Speed St...  | 200,000,000       |
| Cucina Daisy Ivory 4 ft. x 4 ft...  | 200,000,000       |
| Galvan 19 in. White LED Task ...    | 200,000,000       |
| Two Inch Series 8 ft. x 8 ft. x ... | ~10,000,000       |
| Updated - 1 gal. #730B-7 Eng...     | 200,000,000       |
| Updated - 12.25 in. Brown Me...     | 200,000,000       |
| Updated - 8.5 in. Antique Whi...    | 200,000,000       |
| Updated - Kontur Wood 72 in....     | 200,000,000       |
| Updated - Large Rectangle W...      | 200,000,000       |
| Updated - Margie Tribal Fring...    | 200,000,000       |

Query History

Home Untitled.sql Untitled 1.sql + ⚙

My Workspace > Untitled 1.sql

Play ⏪ ⏹ Share ...

```

1 -- Query 4: Audit trail - see all changes
2 SELECT
3     PRODUCT_ID,
4     TITLE,
5     PRICE,
6     CATEGORY,
7     VALID_FROM,
8     VALID_TO,
9     IS_CURRENT,
10    DATEDIFF('day', VALID_FROM, COALESCE(VALID_TO, CURRENT_TIMESTAMP())) as days_active
11   FROM PUBLIC_CORE.DIM_PRODUCTS_SCD2

```

Results (just now)

Table Chart

6 rows 109ms

| # PRODUCT_ID | A TITLE  | # PRICE | A CATEGORY     | VALID_FROM          |
|--------------|--|---------|----------------|---------------------|
| 1 205237692  | 5 gal. #GR-W12 Confident White Semi-Gloss Enamel Exterior Paint                | 220.00  | BEHR ULTRA     | 2021-11-14 01:04:00 |
| 2 205237692  | 5 gal. #GR-W12 Confident White Semi-Gloss Enamel Exterior Paint                | 251.22  | BEHR ULTRA     | 2025-11-07 02:26:00 |
| 3 304590104  | 13/16 in. x 1 in. High Speed Steel Annular Cutter with 3/4 in. TCT Tip         | 32.65   | Drill America  | 2021-11-14 00:56:00 |
| 4 304590104  | 13/16 in. x 1 in. High Speed Steel Annular Cutter with 3/4 in. TCT Tip         | 39.47   | Drill America  | 2025-11-07 02:26:00 |
| 5 314372149  | Textile Classic 15 in. 3-Light Brushed Nickel Close-to-Ceiling Light           | 358.00  | Justice Design | 2021-11-14 01:04:00 |
| 6 314372149  | Updated - Textile Classic 15 in. 3-Light Brushed Nickel Close-to-Ceiling Light | 413.70  | Justice Design | 2025-11-07 02:26:00 |

## 9. Leverage Snowflake Zero-Copy Cloning

### a. Development Environment Cloning

- Create clone database - Used Snowflake's Zero-Copy cloning to create a cloned development database "dev\_product\_catalog" from production to test changes without affecting original data.

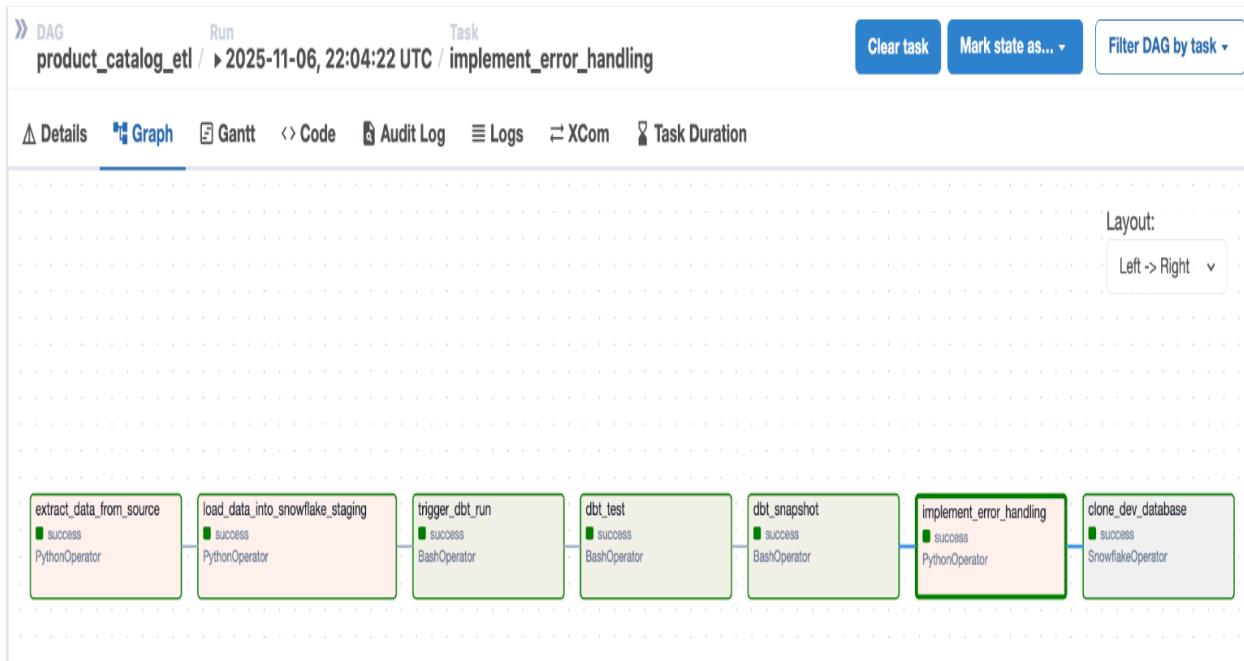
The screenshot shows a Snowflake SQL interface with two tabs: 'Untitled.sql' and 'Untitled 1.sql'. The 'Untitled 1.sql' tab contains the following SQL code:

```
1 -- Create development environment using zero-copy clone
2 CREATE DATABASE DEV_PRODUCT_CATALOG CLONE PRODUCT_CATALOG;
3
4 -- Verify the clone
5 SHOW DATABASES LIKE 'DEV%';
6
7 -- Check that all schemas were cloned
8 USE DATABASE DEV_PRODUCT_CATALOG;
9 SHOW SCHEMAS;
10
11 -- Verify data in cloned database
12 SELECT COUNT(*) AS dev_record_count
13 FROM DEV_PRODUCT_CATALOG.PUBLIC_CORE.DIM_PRODUCTS_SCD2;
14
15 -- Compare with production
16 SELECT
17     'PRODUCTION' AS environment,
18     COUNT(*) AS record_count
19     FROM PRODUCT_CATALOG.PUBLIC_CORE.DIM_PRODUCTS_SCD2
20 UNION ALL
21     SELECT
22         'DEVELOPMENT',
```

The 'Results (just now)' section shows a table with the following data:

| ENVIRONMENT | RECORD_COUNT |
|-------------|--------------|
| PRODUCTION  | 2571         |
| DEVELOPMENT | 2571         |

- Modify Airflow DAG that creates this clone



- Implement a dbt model in the development environment to test a new feature-

Home Untitled.sql Untitled 1.sql +

My Workspace > Untitled 1.sql

Share ...

```
1 -- Use dev database
2 USE DATABASE DEV_PRODUCT_CATALOG;
3 USE SCHEMA PUBLIC_CORE;
4
5 -- Add a new calculated column for testing
6 CREATE OR REPLACE TABLE DIM_PRODUCTS_WITH_RATING AS
7 SELECT
8     *,
9     CASE
10        WHEN PRICE < 50 THEN 'Budget'
11        WHEN PRICE BETWEEN 50 AND 200 THEN 'Mid-Range'
12        ELSE 'Premium'
13    END as price_category,
14    ROUND(UNIFORM(3.0, 5.0, RANDOM()), 1) as customer_rating
15 FROM DIM_PRODUCTS_SCD2
16 WHERE IS_CURRENT = TRUE;
17
18 -- Verify new table
19 SELECT
20     price_category,
21     COUNT(*) as product_count,
22     AVG(price) as avg_price,
```

Results (just now)

Table Chart

| # | PRICE_CATEGORY | PRODUCT_COUNT | Avg_Price    | Avg_Rating |
|---|----------------|---------------|--------------|------------|
| 1 | Budget         | 1057          | 25.64920530  | 4.000946   |
| 2 | Mid-Range      | 869           | 111.48805524 | 3.981588   |
| 3 | Premium        | 625           | 711.27564800 | 4.004800   |

Query History

- b. Point-in-Time Analysis - Created database clones at specific timestamps to perform historical comparisons and year-end analysis.

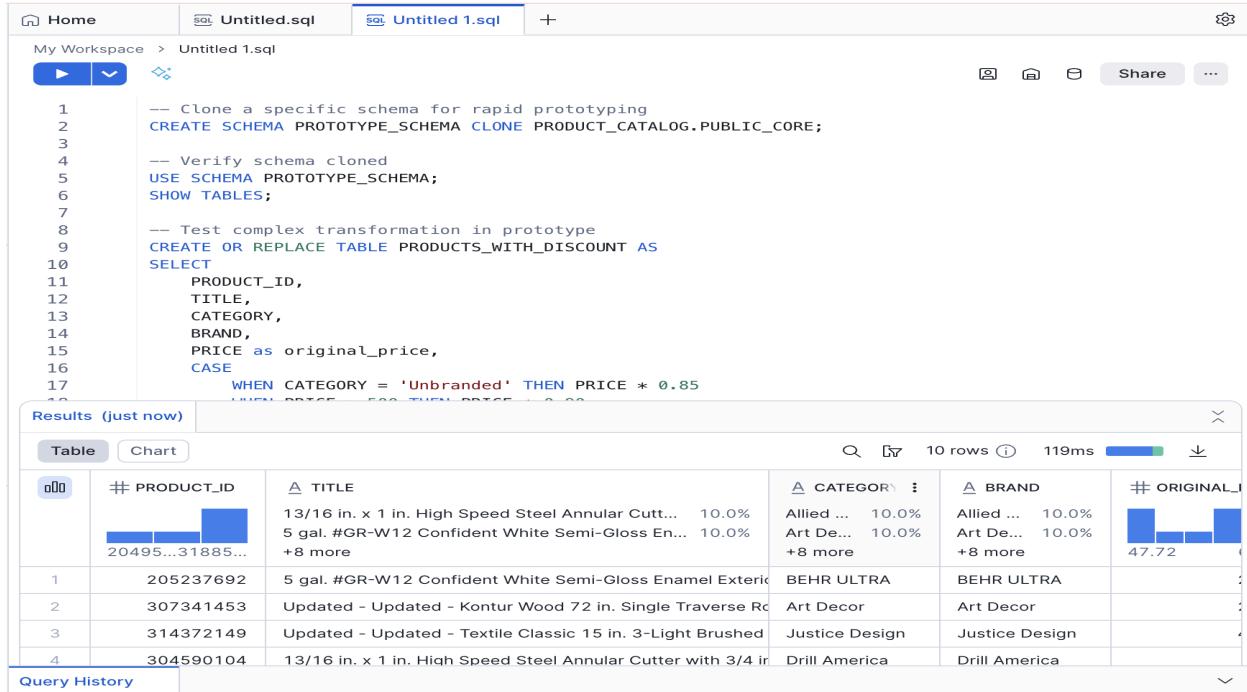
Home Untitled.sql Untitled 1.sql +

My Workspace > Untitled 1.sql

Share ...

```
1 CREATE DATABASE PRODUCT_CATALOG_EOY_2024 CLONE PRODUCT_CATALOG;
2
3 SHOW DATABASES LIKE '%EOY%';
4
5 USE DATABASE PRODUCT_CATALOG_EOY_2024;
6
7 SELECT COUNT(*) as eoy_record_count
8 FROM PUBLIC_CORE.DIM_PRODUCTS_SCD2;
9
10
11 SELECT
12     CATEGORY,
13     COUNT(*) as product_count,
14     AVG(PRICE) as avg_price,
15     MIN(PRICE) as min_price,
16     MAX(PRICE) as max_price
17 FROM PUBLIC_CORE.DIM_PRODUCTS_SCD2
18 WHERE IS_CURRENT = TRUE
19 GROUP BY CATEGORY;
```

- c. Rapid Prototyping with Cloning - Cloned a specific schema "prototype\_schema" to test new transformations without hindering the existing data.



The screenshot shows a Snowflake SQL workspace with two tabs: "Untitled.sql" and "Untitled 1.sql". The "Untitled 1.sql" tab contains the following SQL code:

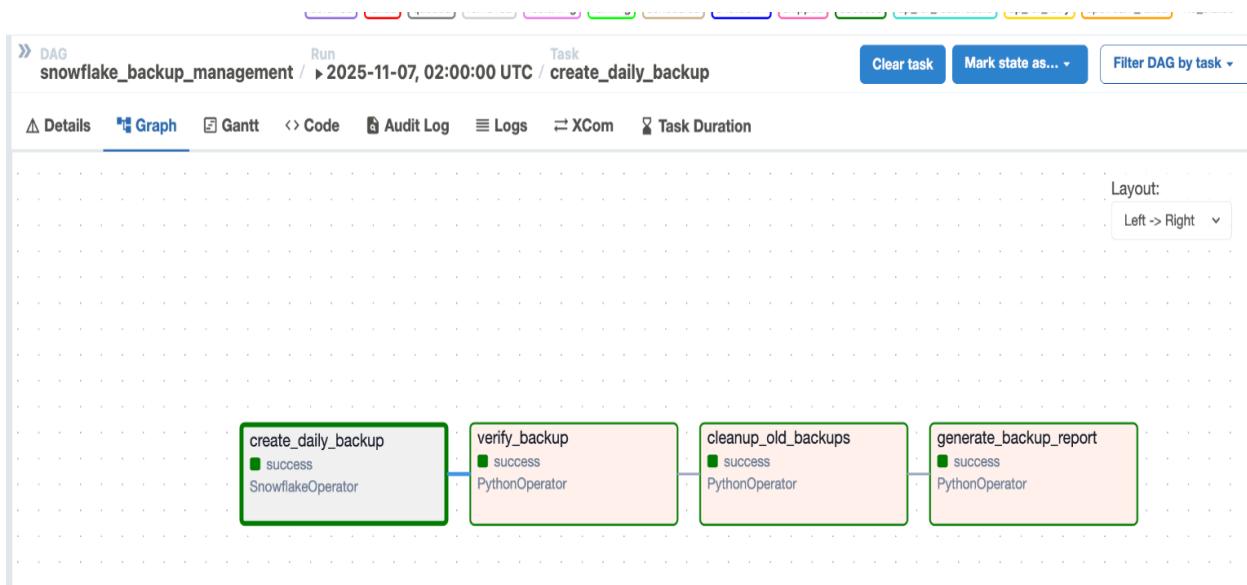
```

1 -- Clone a specific schema for rapid prototyping
2 CREATE SCHEMA PROTOTYPE_SCHEMA CLONE PRODUCT_CATALOG.PUBLIC_CORE;
3
4 -- Verify schema cloned
5 USE SCHEMA PROTOTYPE_SCHEMA;
6 SHOW TABLES;
7
8 -- Test complex transformation in prototype
9 CREATE OR REPLACE TABLE PRODUCTS_WITH_DISCOUNT AS
10 SELECT
11     PRODUCT_ID,
12     TITLE,
13     CATEGORY,
14     BRAND,
15     PRICE as original_price,
16     CASE
17         WHEN CATEGORY = 'Unbranded' THEN PRICE * 0.85
18         ELSE PRICE
19     END as discounted_price
20

```

The "Results (just now)" section displays a table with 10 rows of data. The columns are PRODUCT\_ID, TITLE, CATEGORY, BRAND, and ORIGINAL\_PRICE. The data includes various products like "13/16 in. x 1 in. High Speed Steel Annular Cutter" and "5 gal. #GR-W12 Confident White Semi-Gloss Enamel Exterior Paint". The "ORIGINAL\_PRICE" column shows values like 10.00% and 47.72.

- d. Backup and Restore Strategy - Implemented daily backups using snowflake cloning and configured airflow DAG "snowflake\_backup\_management"



```

My Workspace > Untitled 1.sql
Share ...
1   -- Create daily backup using zero-copy clone
2   CREATE OR REPLACE DATABASE PRODUCT_CATALOG_BACKUP_20250106
3   CLONE PRODUCT_CATALOG;
4
5   -- Verify backup
6   SHOW DATABASES LIKE '%BACKUP%';
7
8   -- Check backup data
9   SELECT
10      'PRODUCTION' AS environment,
11      COUNT(*) AS record_count
12  FROM PRODUCT_CATALOG.PUBLIC_CORE.DIM_PRODUCTS_SCD2
13 UNION ALL
14  SELECT
15      'BACKUP',
16      COUNT(*)
17  FROM PRODUCT_CATALOG_BACKUP_20250106.PUBLIC_CORE.DIM_PRODUCTS_SCD2;

```

| Results (just now) |                               |                                 |            |            |        |
|--------------------|-------------------------------|---------------------------------|------------|------------|--------|
|                    | created_on                    | name                            | is_default | is_current | origin |
| 1                  | 2025-11-07 13:48:41.590 -0800 | PRODUCT_CATALOG_BACKUP_20250104 | N          | Y          |        |
| 2                  | 2025-11-07 13:48:37.668 -0800 | PRODUCT_CATALOG_BACKUP_20250105 | N          | N          |        |
| 3                  | 2025-11-07 13:48:32.618 -0800 | PRODUCT_CATALOG_BACKUP_20250106 | N          | N          |        |

#### 10. Experiment with various options and describe 5 key takeaways.

- Airflow provides robust scheduling and dependency management for ETL orchestration.
- dbt simplifies SQL transformations and enables modular, test-driven data engineering.
- Zero-copy cloning in Snowflake allows for instant environment creation with no additional storage cost.
- SCD Type 2 implementation ensures full product history tracking for analytical accuracy.
- Integrating dbt with Airflow builds an automated end-to-end data pipeline.

#### 11. If given a chance, how will you extend this exercise to make the learning experience better?

1. Deploy the project to a dockerized Airflow environment for portability
2. Integrate CI/CD automation for dbt using GitHub Actions

#### 12. Did you find or come across solutions to similar problems by using Generative AI or other sources?

1. A problem when one Airflow webserver port was occupied and many PID processes were running in the background had been fixed using ChatGPT; GPT-5 instructed me how to find out and kill those using terminal commands.
2. I tried to troubleshoot Airflow's Snowflake connection using ChatGPT (GPT-5) by learning how to correctly configure the snowflake\_conn\_id in Airflow Connections UI and then how to test it from the command line. This guidance helped resolve the issue where the authentication for SnowflakeOperator failed.