Assignment: Building a Modern Data Pipeline with dbt, Snowflake, Great Expectations, and Airflow

This assignment demonstrates an end-to-end ELT pipeline using the Bank Marketing Dataset from Kaggle, which contains ~45,000 rows of Portuguese banking campaign data with features like age, job, marital status, and loan subscription outcomes. The pipeline loads raw data into Snowflake, transforms it with dbt for analytics-ready models, validates quality with Great Expectations, and orchestrates everything via Airflow DAGs. This setup aligns with production data engineering practices for cloud data platforms.

Dataset Overview

The Bank Marketing Dataset includes 17 columns such as `age`, `job`, `marital`, `education`, `default`, `housing`, `loan`, `contact`, `month`, `day_of_week`, `duration`, `campaign`, `pdays`, `previous`, `poutcome`, `emp_var_rate`, `cons_price_idx`, and `y` (target: 'yes'/'no' for term deposit subscription). Key goals include cleaning categorical variables, handling missing values, and creating features for campaign performance analysis. Download `bank-additional-full.csv` from the Kaggle link for this pipeline.

Architecture Overview

```
Raw Data (S3/Kaggle) → Airflow (Load to Snowflake) → dbt (Transform) → Great Expectati
```

- **Snowflake**: Serves as the data warehouse with stages for raw data and schemas for transformed models.
- **dbt**: Handles SQL-based transformations, testing, and documentation.
- **Great Expectations**: Enforces data quality checks post-transformation.
- **Airflow**: Orchestrates the sequence with task dependencies, retries, and scheduling.

Step 1: Environment Setup

Create Snowflake resources via SQL worksheets:

```
CREATE DATABASE BANK_MARKETING;
CREATE SCHEMA RAW, ANALYTICS;
CREATE STAGE RAW_BANK_DATA FILE_FORMAT = (TYPE = CSV FIELD_DELIMITER=',' SKIP_HEADER=1);
CREATE WAREHOUSE BANK_WH WITH WAREHOUSE_SIZE = 'XSMALL';
GRANT USAGE ON DATABASE BANK_MARKETING TO ROLE ACCOUNTADMIN;
```

Set up Airflow connections: `snowflake_default` (Snowflake), `aws_default` (S3 for data upload). Install packages: `dbt-snowflake`, `apache-airflow-providers-snowflake`, `great-expectations`, `astronomer-cosmos` for dbt-Airflow integration.

Step 2: Data Ingestion with Airflow

Create `dags/bank_marketing_dag.py`:

```
from airflow import DAG
from airflow.operators.python import PythonOperator
from airflow.providers.snowflake.hooks.snowflake import SnowflakeHook
from airflow.providers.apache.cosmos.operators.dbt import DbtCloudRunJobOperator
from great_expectations_provider.operators.great_expectations import GreatExpectationsOpe
from datetime import datetime

def upload_and_load():
    hook = SnowflakeHook(snowflake_conn_id='snowflake_default')
    # Assume CSV uploaded to S3; copy to stage
    hook.run("PUT file://bank-additional-full.csv @RAW_BANK_DATA AUTO_COMPRESSION=ON;")
    hook.run("COPY INTO RAW.bank_data FROM @RAW_BANK_DATA FILE_FORMAT = (TYPE = CSV);")

dag = DAG('bank_marketing_pipeline', start_date=datetime(2026,1,1), schedule='@daily')

load_task = PythonOperator(task_id='load_raw', python_callable=upload_and_load, dag=dag)
```

This task stages and copies the CSV into `RAW.bank_data` table.

Step 3: dbt Transformations

Initialize dbt project: `dbt init bank_marketing --adapter snowflake`. Configure `profiles.yml`:

```
bank_marketing:
  target: dev
  outputs:
    dev:
      type: snowflake
      account: your_account
      user: your_user
      password: your_pass
      role: ACCOUNTADMIN
      database: BANK_MARKETING
      warehouse: BANK_WH
      schema: ANALYTICS
```

models/staging/stg_bank_data.sql (basic cleaning):

```
{{ config(materialized='table') }}
SELECT
  $1:age::INT as age,
  $1:job::STRING as job,
  $1:y::BOOLEAN as subscribed,
```

```
  -- Parse other columns similarly
FROM {{ source('raw', 'bank_data') }}
```

models/marts/campaign_performance.sql (aggregated model):

```
{{ config(materialized='table') }}
SELECT
  job,
  COUNT(*) as total_contacts,
  AVG(CASE WHEN subscribed THEN 1 ELSE 0 END) as conversion_rate,
  COUNT(CASE WHEN subscribed THEN 1 END) as successful_campaigns
FROM {{ ref('stg_bank_data') }}
GROUP BY job
```

Run: `dbt run --models stg_bank_data+` followed by `dbt test` (add schema.yml tests for not_null, accepted_values).

Step 4: Great Expectations Validation

Initialize GX: `great_expectations init`. Create `great_expectations/expectations/bank_suite.yml`:

```
name: bank_marketing_suite
expectations:
  - expect_column_values_to_not_be_null: {column: age}
  - expect_column_values_to_be_between: {column: age, min_value: 18, max_value: 100}
  - expect_column_values_to_be_in_set: {column: job, value_set: ["admin", "blue-collar",
```

In Airflow DAG, add:

```
gx_task = GreatExpectationsOperator(
    task_id='validate_data',
    expectation_suite_name='bank_marketing_suite',
    data_context_root='/path/to/gx/',
    conn_id='snowflake_default',
    dag=dag
)
```

This runs post-dbt checks on `ANALYTICS.campaign_performance`.

Step 5: Full Airflow DAG and Execution

Complete DAG sequence:

```
load_task >> dbt_run = DbtCloudRunJobOperator(
    task_id='dbt_transform', dbt_cloud_conn_id='dbt_cloud',
    project_id=your_project_id, job_id=your_job_id, dag=dag
) >> gx_task
```

Deploy via `docker-compose` or Kubernetes. Schedule daily; monitor via Airflow UI. Expected output: Validated mart table queryable in Snowflake for BI tools like Tableau.

Testing and Best Practices

- **dbt Tests**: Unique keys on `age+job`, freshness on source.
- **GX Checkpoints**: Multi-batch validation for incremental loads.
- **Monitoring**: Airflow SLAs, Snowflake query profiles for cost.
- **CI/CD**: GitHub Actions for dbt deps/test, Airflow Docker builds.
  This pipeline scales to production, handling data quality at every layer while leveraging your expertise in Snowflake, dbt, and Airflow.[1] [2] [3] [4]

⁂

1. https://www.astronomer.io/blog/improved-data-quality-checks-in-airflow-with-great-expectations-operator/
2. https://github.com/jonathangosling/startups_dbt
3. https://www.clearpeaks.com/orchestrating-dbt-on-snowflake-using-airflow-and-astro/
4. https://www.kaggle.com/datasets/janiobachmann/bank-marketing-dataset
5. https://www.kaggle.com/datasets/marehmanforkaggle/orders-dataset-dbt-airflow-snowflake
6. https://www.linkedin.com/posts/kaustubh-gupta_dataengineering-dbt-airflow-activity-7383094365444956160-_6BA
7. https://github.com/Ali-jalil88/Mlflow-Bank-Marketing
8. https://github.com/indrayantom/Bank_Marketing_Predictive
9. https://www.reddit.com/r/dataengineering/comments/10im7hb/how_to_integrate_great_expectations_and_dbt_with/
10. https://www.youtube.com/watch?v=kIZPfU06Lbo