Building
Modern Data Stacks

# Project Brief

## Adventure Works

Adventure works is a bicycle manufacturing company. This project demonstrated how to build data pipelines for an e-commerce, implement machine learning models, and develop business intelligence reporting solutions.

BRIAN GWAYI

Building
Modern Data Stacks

# First Things First !!!

## Four Key Questions

I. Where do we consolidate our data ? > Storage
II. How will we get it there ? > Ingestion
III. How will we clean it up? > Transformation
IV. How will we analyze it? > Reporting

BRIAN GWAYI

The
Big Choice

# Data Stack

## Popular Options

Storage > Snowflake, BigQuery, s3, Redshift
Ingestion > Airbyte, Airflow, Fivetran, dagster
Transformation > dbt
Reporting > Tableau, Power BI, Looker, Superset

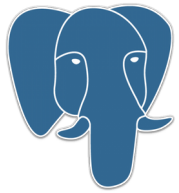
N/B This is not an exhaustive list.

BRIAN GWAYI

# Data Stack Architecture Design

Source

PostgreSQL



Ingestion

Airflow

+

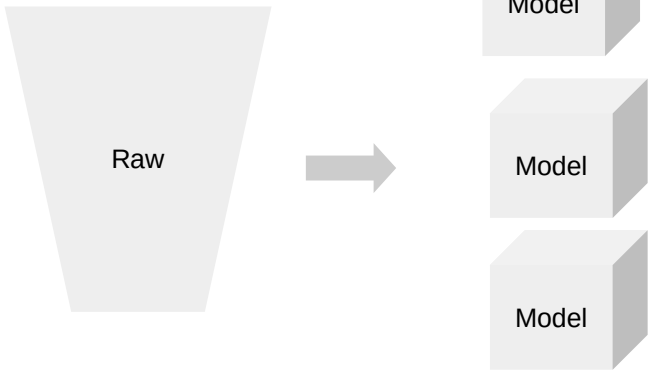
Python

Storage
BigQuery

Amazon **Redshift**

Transformation
dbt

dbt




Reporting

Looker

looker

# End Goal

**Put data to use - make decisions**

“Data is like garbage. You’d better know what you are going to do with it before you collect it.”

~ Mark Twain

# PROJECTS

01

**Storage/Database/Data Warehouse**
Google BigQuery
Snowflake
AWS Redshift

02

**Ingestion**
Apache Airflow
Airbyte
Dagster

03

**Transformation**
Setting up dbt
Building Models

04

**Reporting**
Looker
Tableau
Power BI

# 02

## INGESTION
## BUILDING ELT DATA PIPELINE
## – Apache Airflow

Setting up Apache Airflow - Documentation

Python ELT (Extract Load Transform) script

```
# importing libraries

from airflow.decorators import dag, task
from datetime import datetime, timedelta
import requests
from google.cloud import bigquery
import pandas as pd
import psycopg2
from io import StringIO
```

# 02

## INGESTION

Setting up Apache Airflow

Defining a DAG - Directed Acyclic Graph

```
args{
    ”owner”:”gwayi”,
    "retries": 1,
    "retry_delay":timedelta(minutes=5)
    }


@dag(
    default_arguments = args
    schedule=timedelta(minutes=30),
    start_date=datetime(2024, 7, 29),
    catchup=False,
    tags=['Team B']
    )
```

# 02

## INGESTION

Setting up Apache Airflow

Extract Task Group – Source PostgreSQL Database

```
@task()
def extract():
  try:
     src_cursor.execute(sql)
     tables = cursor.fetchall()

     output = {}

     for table in tables:
       cursor.execute(f"SELECT *
              FROM {table[0]}")
rows = cursor.fetchall()

output.update({table[0]: rows})
       return output

    except Exception as e:
        print("extract error:" +
str(e))

    finally:
        connection.close()
output = extract()
```

Load
Strategy

# Incremental Load

## Data Change Capture

**Timestamp Approach** – extracts all rows modified Since the last execution date {{ds}}.

```
SELECT *
FROM {table[0]}
WHERE last_updated >= '{{ ds }}';
```

BRIAN GWAYI

# 02

## INGESTION

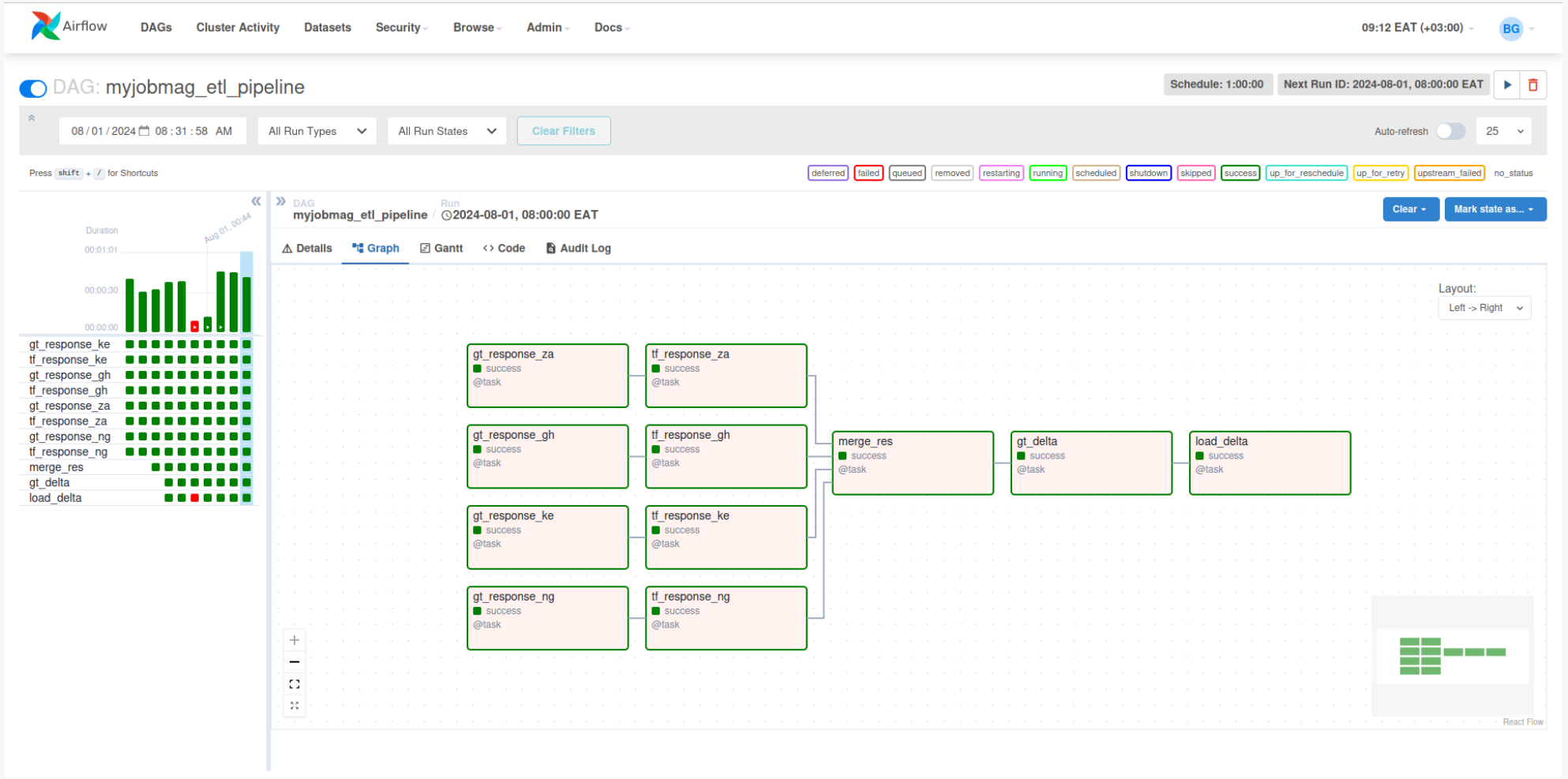
Setting up Apache Airflow

Load Task Group – Destination BigQuery

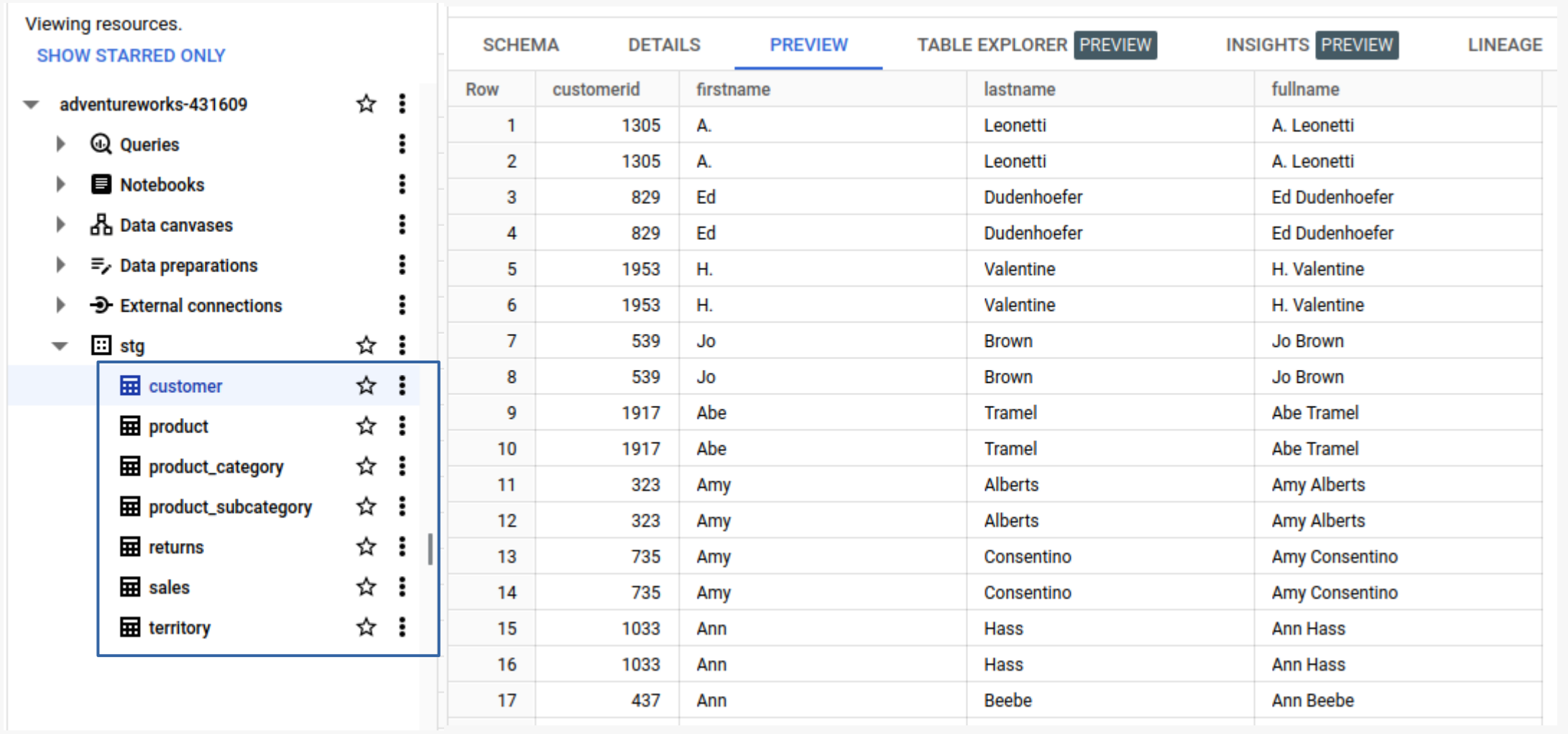
```
task()
@def load(dict):
    pandas_gbq.to_gbq(
        df[{table}],
        project_id=project_id,
        if_exists=append,
load(data)
```

```
Set dependencies

extract = extract()
load = load(extract)

extract >> load
```

# Orchestrating Data Pipeline - Airflow



# Google BigQuery




| Row | customerid | firstname | lastname | fullname |
|---|---|---|---|---|
| 1 | 1305 | A. | Leonetti | A. Leonetti |
| 2 | 1305 | A. | Leonetti | A. Leonetti |
| 3 | 829 | Ed | Dudenhoefer | Ed Dudenhoefer |
| 4 | 829 | Ed | Dudenhoefer | Ed Dudenhoefer |
| 5 | 1953 | H. | Valentine | H. Valentine |
| 6 | 1953 | H. | Valentine | H. Valentine |
| 7 | 539 | Jo | Brown | Jo Brown |
| 8 | 539 | Jo | Brown | Jo Brown |
| 9 | 1917 | Abe | Tramel | Abe Tramel |
| 10 | 1917 | Abe | Tramel | Abe Tramel |
| 11 | 323 | Amy | Alberts | Amy Alberts |
| 12 | 323 | Amy | Alberts | Amy Alberts |
| 13 | 735 | Amy | Consentino | Amy Consentino |
| 14 | 735 | Amy | Consentino | Amy Consentino |
| 15 | 1033 | Ann | Hass | Ann Hass |
| 16 | 1033 | Ann | Hass | Ann Hass |
| 17 | 437 | Ann | Beebe | Ann Beebe |

# Transformation

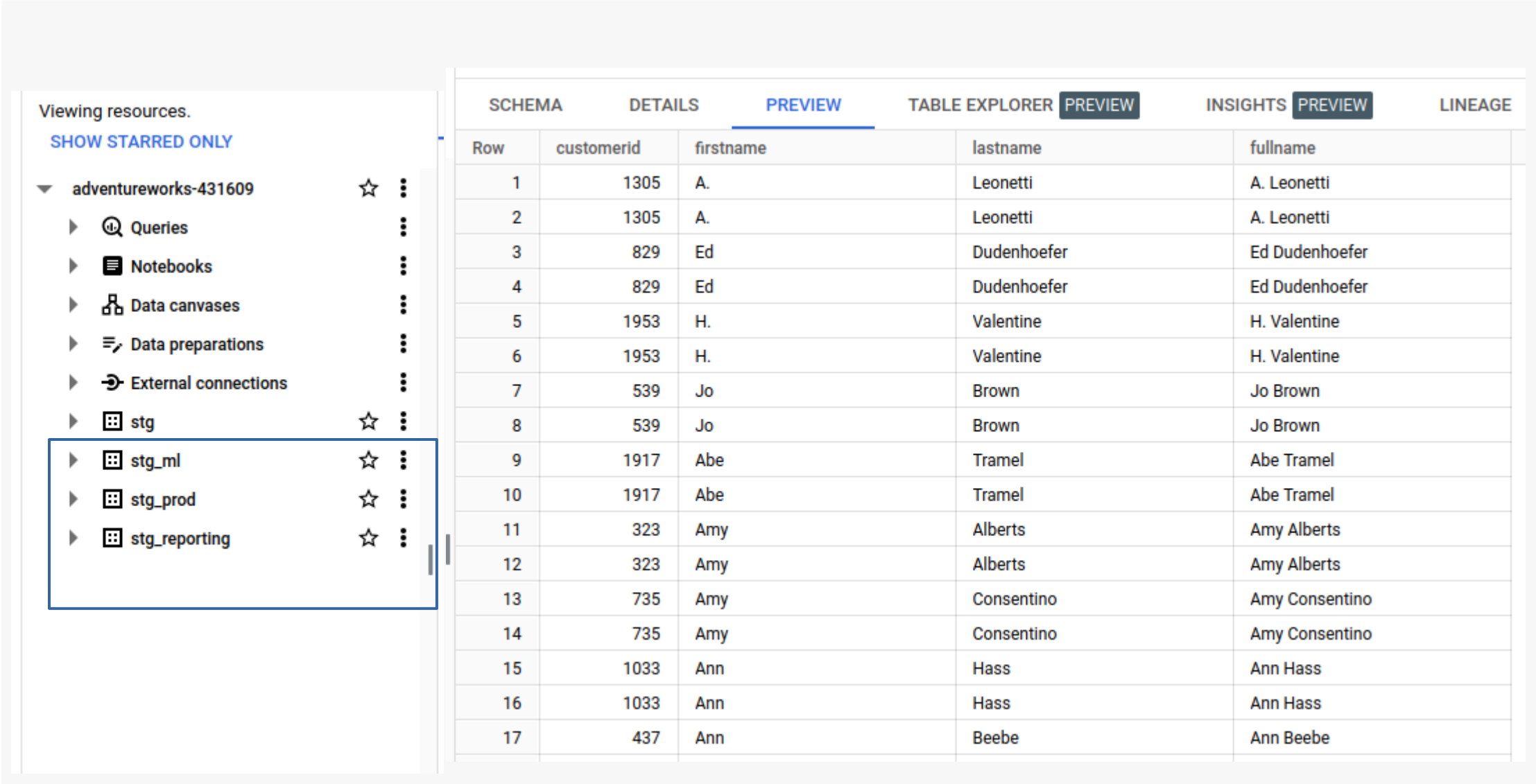
## Business Modeling

**Using  dbt build three models ;**

**ML Model –** for Machine Learning Model
**Production –** for production
**Reporting –** for reporting

BRIAN GWAYI

# Google BigQuery




| Row | customerid | firstname | lastname | fullname |
|---|---|---|---|---|
| 1 | 1305 | A. | Leonetti | A. Leonetti |
| 2 | 1305 | A. | Leonetti | A. Leonetti |
| 3 | 829 | Ed | Dudenhoefer | Ed Dudenhoefer |
| 4 | 829 | Ed | Dudenhoefer | Ed Dudenhoefer |
| 5 | 1953 | H. | Valentine | H. Valentine |
| 6 | 1953 | H. | Valentine | H. Valentine |
| 7 | 539 | Jo | Brown | Jo Brown |
| 8 | 539 | Jo | Brown | Jo Brown |
| 9 | 1917 | Abe | Tramel | Abe Tramel |
| 10 | 1917 | Abe | Tramel | Abe Tramel |
| 11 | 323 | Amy | Alberts | Amy Alberts |
| 12 | 323 | Amy | Alberts | Amy Alberts |
| 13 | 735 | Amy | Consentino | Amy Consentino |
| 14 | 735 | Amy | Consentino | Amy Consentino |
| 15 | 1033 | Ann | Hass | Ann Hass |
| 16 | 1033 | Ann | Hass | Ann Hass |
| 17 | 437 | Ann | Beebe | Ann Beebe |