Building Simple Data Stack

Project Brief

This project demonstrate how to implement a modern data stack, build data pipelines, machine learning and reporting capabilities using a variety of tools.

BRIAN GWAYI
Independent Data Lead & Engineer

First Things First !!!

Five Key Questions

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

BRIAN GWAYI
Independent Data Lead & Engineer

Data Stack Architecture Design

Where is our data?

PostgreSQL

MySQL

How will we get it there?

Airflow

+

Python

Where do we consolidate our data?

Amazon Redshift

Transformation

dbt

Raw

Model

Model

How do we analyze it?

tableau

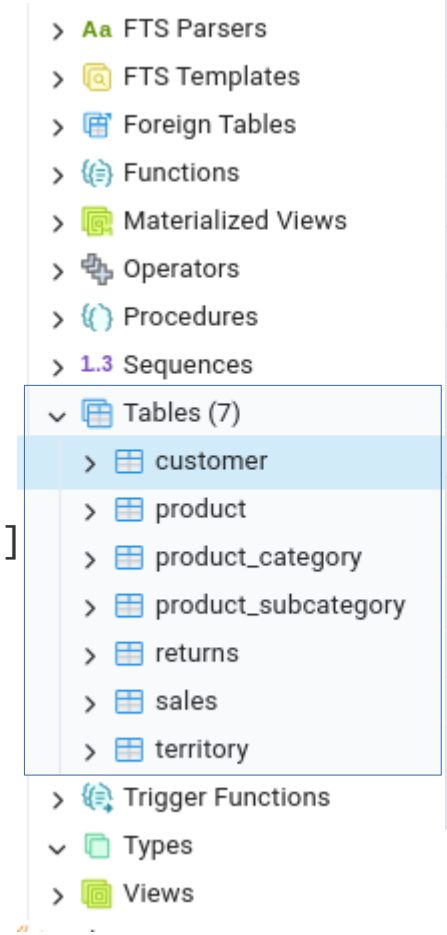
looker

Power BI

Where

is Our Data?

```
Source : PostgreSQL
Schema : Public
Database_Name: adw_db
Tables_Count : 7
Tables : [customer,
          product,
          product_category,
          returns,
          sales,
          territory,
          product_subcategory]
```



Data Output | Messages | Notifications

SQL

	orderdate date	stockdate date	ordernumber character varying (255)	productkey integer	customerkey integer
1	2022-01-01	2021-12-13	SO61285	529	23791
2	2022-01-01	2021-09-24	SO61285	214	23791
3	2022-01-01	2021-09-04	SO61285	540	23791
4	2022-01-01	2021-09-28	SO61301	529	16747
5	2022-01-01	2021-10-21	SO61301	377	16747
6	2022-01-01	2021-10-23	SO61301	540	16747
7	2022-01-01	2021-09-04	SO61269	215	11792
8	2022-01-01	2021-10-21	SO61269	229	11792
9	2022-01-01	2021-10-24	SO61286	528	11530
10	2022-01-01	2021-09-27	SO61286	536	11530
11	2022-01-01	2021-10-23	SO61298	530	18155
12	2022-01-01	2021-12-02	SO61298	214	18155
13	2022-01-01	2021-12-15	SO61298	223	18155
14	2022-01-01	2021-10-01	SO61310	538	13541
15	2022-01-01	2021-11-08	SO61310	584	13541

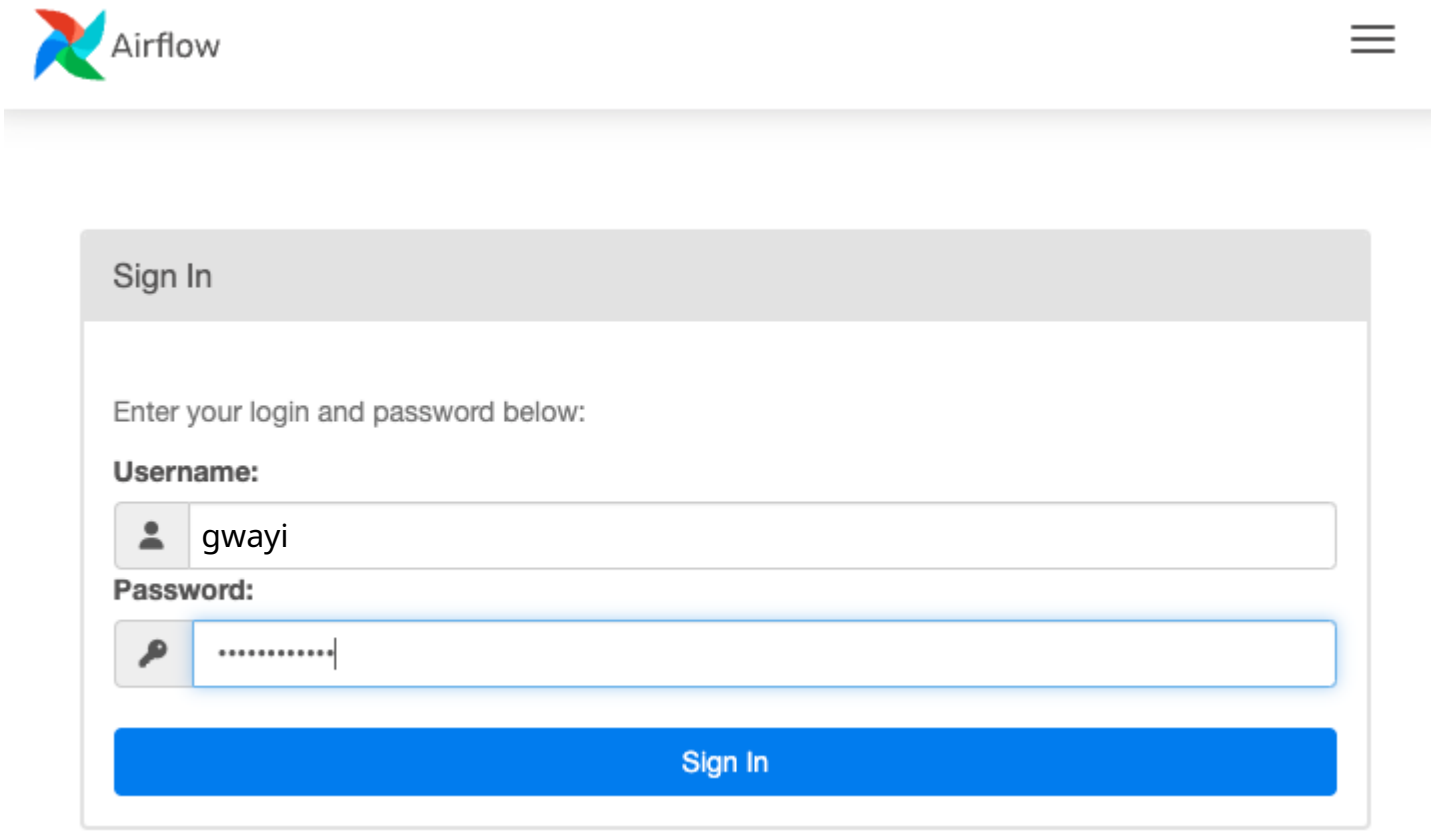
How do we ingest Our Data?

```
Ingestion : Programmatically
Orchestration : Apache Airflow
```

Apache Airflow Setup

```
Terminal

$ python3 -m venv airflow-env
$ source airflow-env/bin/activate
$ export AIRFLOW_HOME=~/airflow
$ pip install apache-airflow
$ airflow db init
$ airflow webserver -p 8080
$ airflow sheduler
```

Apache Airflow Webserver UI



How do we

ingest Our Data?

```
# install dependencies

pip install google-cloud-bigquery
pip install --upgrade snowflake-connector-python

# importing libraries

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

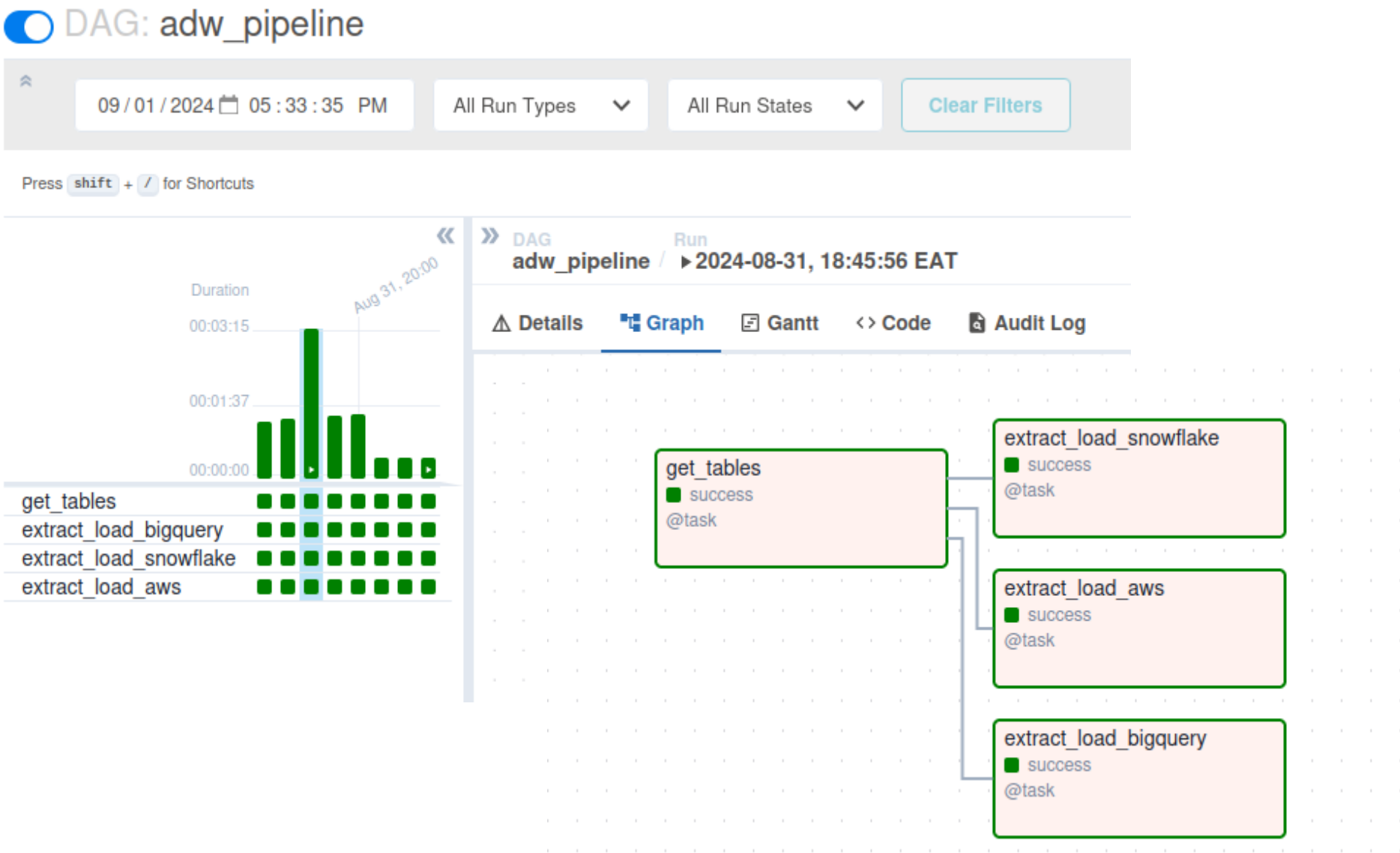
```
# define a DAG

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=['DataOps Team']
    )
```

How do we ingest Our Data?

```
@task()
def get_tables(conn):
"""extract list of tables
in public schema"""

  try:
    cursor.execute(
         f"""SELECT table_name
         FROM information_schema.tables
         WHERE table_schema = 'public'"""
          )

    records = cursor.fetchall()
    tbls = [x[0] for x in records]

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

  finally:
     conn.close()
```

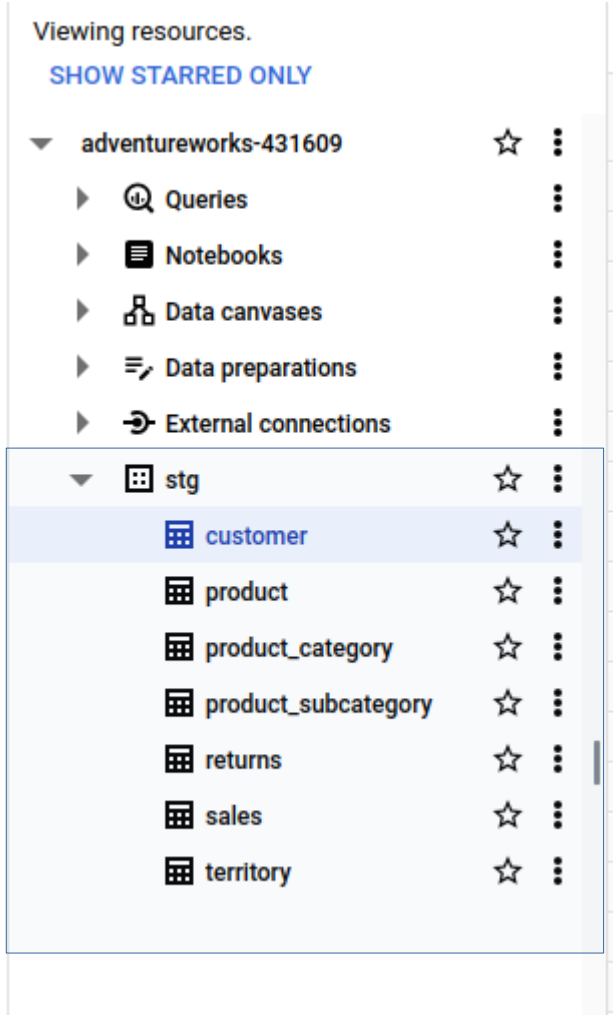
```
@task()
def extract|load_bigquery(tbls, conn):
"""loop through tbls then extract & load"""

     client = bigquery.Client()
     job_config = bigquery.LoadJobConfig(
     write_disposition="WRITE_TRUNCATE")

      for tbl in tbls:
      table_id = f"adventureworks-431609.stg.{tbl}"
      sql = f"SELECT * FROM {tbl} WHERE
      updated_at >= {ds}'"
      df = pd.read_sql(sql, conn)

      job = client.load_table_from_dataframe(
      df, table_id, job_config=job_config)
      job.result()

   get_tables = get_tables()
   extract_load = extract_load(get_tables)
```

Running the Pipeline – Apache Airflow



Where do we consolidate Our Data?

```
Storage : BigQuery
Project : AdventureWorks
Dataset: stg
Tables_Loaded : 7
Tables : [customer,
          product,
          product_category,
          returns,
          sales,
          territory,
          product_subcategory]
```



Query results

JOB INFORMATION | RESULTS | CHART | JSON

Row	customerkey	firstname
18	20259	JENNY
19	20382	JACK
20	21601	JACQUELINE
21	23433	PEDRO
22	24804	TONYA
23	26200	IAN
24	26394	MARTHA
25	28999	EDWARD
26	11005	JULIO
27	11009	SHANNON
28	11057	CARL
29	11074	LEVI
30	11086	RYAN
31	11094	CEDRIC
32	11095	CHAD
33	11104	EDGAR
34	11106	JESSIE
35	11109	RUBEN

How do we

transform Our Data?

```
Transformation : dbt
Orchestration : Apache Airflow

Models 3 :

  production.sql,
  machine_learning.sql,
  business_reporting.sql
```

Set up dbt

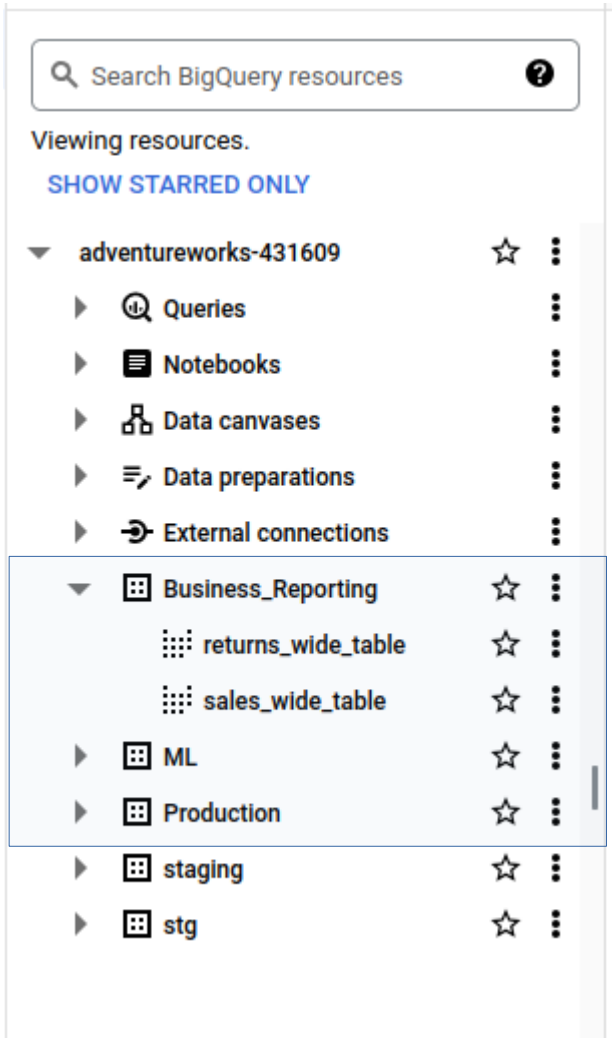
```
pip install dbt-biquery
dbt –version
dbt init dbt_adw

Key Commands
dbt debug, dbt run, dbt run –full-refresh,
dbt seed, dbt test, dbt docs generate
```

profiles.yml Set up

```
dbt_adw:

  Outputs:

    dev:

      dataset: stg
      job_execution_timeout_seconds: 300
      job_retries: 1
      keyfile: <json_key_path>
      location: US
      method: service-account
      priority: interactive
      project: adventureworks-431609
      threads: 10
      type: bigquery

  target: dev
```

How do we transform Our Data?

```
# run models
$ dbt run

# generate documentation
$ dbt docs generate

# run seed
$ dbt seed

# run validation
$ dbt test
```



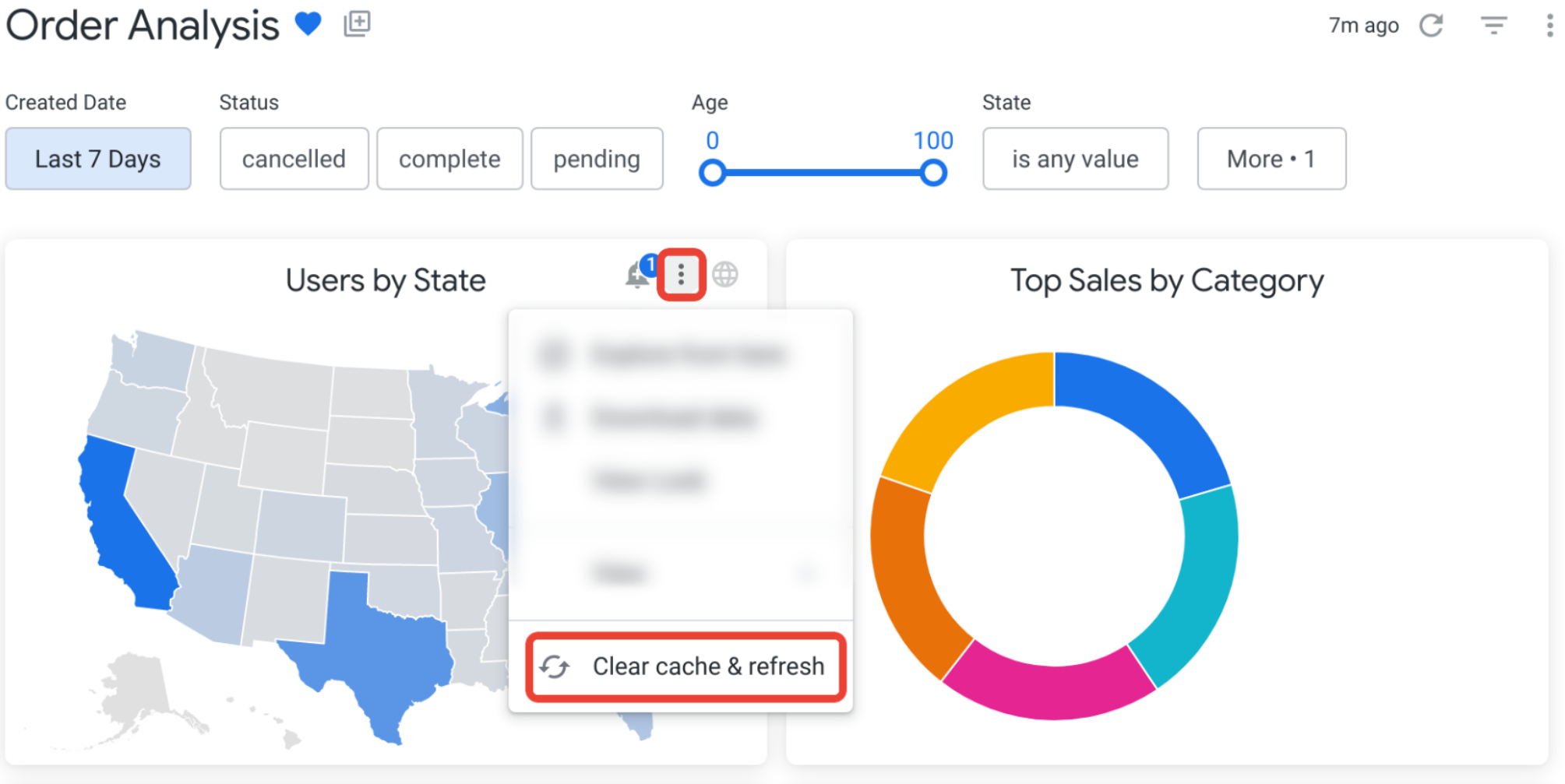
Query results

JOB INFORMATION | RESULTS | CHART | JSON

Row	customerkey	firstname
18	20259	JENNY
19	20382	JACK
20	21601	JACQUELINE
21	23433	PEDRO
22	24804	TONYA
23	26200	IAN
24	26394	MARTHA
25	28999	EDWARD
26	11005	JULIO
27	11009	SHANNON
28	11057	CARL
29	11074	LEVI
30	11086	RYAN
31	11094	CEDRIC
32	11095	CHAD
33	11104	EDGAR
34	11106	JESSIE
35	11109	RUBEN

How do we analyze & Report Our Data?

Google Looker Studio



Ultimate End Goal

Data + Insights + Action
= Actionable Insights

Data	Insight	Action
What happened/ will happen?	**Why** did it happened/ will it happen?	**What** do we do?

Improvements
Connection pooling

```
import psycopg2
from psycopg2 import pool

# Set up connection pooling
db_pool = psycopg2.pool.SimpleConnectionPool(
    1, 20,  # min and max connections
    dbname='your_db',
    user='your_user',
    password='your_password',
    host='localhost',
    port='5432'
)

def extract_data_with_pool(query):
    conn = None
    try:
        conn = db_pool.getconn()
        cursor = conn.cursor()
        cursor.execute(query)
        rows = cursor.fetchall()
```