29 March 2025

Brazilian E-commerce Data Warehouse

Pipelines Design and Deployment

DSAI – Group 7

Table of Contents

Project Overview

Project Name

Brazilian E-commerce Data Warehouse Pipelines Design and Deployment.

Objective

To design and deploy a scalable, automated data pipeline that processes and ingest daily transactional CSV data from an e-commerce platform to data warehouse and prepares it for analytical use.

Background

The e-commerce platform generates daily CSV exports covering various domains such as orders, payments, customers, sellers, products, and deliveries. These files are delivered on a scheduled basis and serve as the primary source of truth for business activities. To ensure timely and accurate insights, a reliable data pipeline has been implemented to process these files end-to-end.

Scope

This project focuses on:

- Defining and validating data schemas to align with analytical model requirements.
- Setting up a designated input directory for daily CSV files from the e-commerce platform, with processed files archived to a separate directory.
- Performing preliminary reading and applying basic data quality checks to validate format and content consistency during ingestion into the staging layer.
- Logging issues and handling errors during processing to ensure pipeline stability and continuity.
- Defining data validation and transformation rules using dbt to load clean, verified data into BigQuery.
- Designing the pipeline to be modular and scalable, supporting future analytics, and reporting requirements.

Key Deliverables

- A daily scheduled ingestion pipelines
- A staging and transformation layer in Google BigQuery
- Analytical-ready models (fact and dimension tables)
- Reproducible local development environment
- Project documentation

Stakeholders

- Data Engineering Team
- Analytics and BI Team

Data Sources

The data ingested by the pipeline originates from an internal e-commerce platform, which generates and uploads a daily set of CSV files to a designated ingestion directory on the Ubuntu server. While the data delivery mechanism is managed externally and outside the scope of this project, the pipeline is designed to monitor and process the incoming files reliably.

File Location

Ingestion Directory: project/data/

Archive Directory: project/data_backup/

After successful ingestion, files are renamed with a date-based suffix (`_YYYY_MM_DD`) and moved to the archive directory for traceability and auditing purposes.

Data Format

All files are in **CSV format**, encoded in UTF-8, and use a comma as the delimiter. Each file represents a specific domain entity within the e-commerce ecosystem.

Expected Files (Daily Set)

- `olist_customers_dataset.csv` – Customer information and location data
- `olist_order_items_dataset.csv` – Details of items within each order
- `olist_order_payments_dataset.csv` – Payment transactions and methods
- `olist_orders_dataset.csv` – Order-level information including timestamps and status
- `olist_products_dataset.csv` – Product catalog details
- `olist_sellers_dataset.csv` – Seller profiles and business information

Frequency

- **Daily** at **00:00 (midnight)** server time
- A **cron job** triggers the ingestion process using **dbt**, which reads the available files from the ingestion directory and loads them into the BigQuery data warehouse.

Data Pipeline

The data pipeline is designed to automate the ingestion and transformation of daily CSV files received from the e-commerce platform. It operates on a scheduled basis using a cron job and leverages **dbt** for orchestration and transformation. The pipeline ensures data is ingested into **Google BigQuery**, validated, transformed into analytical models, and made available for downstream analytics and reporting.

Pipeline Flow

1. **Data Delivery**
   The e-commerce platform generates a daily batch of CSV files and uploads them to the ingestion directory on the Ubuntu server at `./data/`. File delivery mechanisms are handled externally and are outside the scope of this project.

2. **Scheduling & Triggering**
   A cron job is configured to run daily at **00:00 (midnight)**. This job triggers **dbt** to initiate ingestion and transformation.

3. **Ingestion Process**

   - dbt reads each CSV file from the `./data/` directory.

   - Preliminary validation checks (e.g., file structure, encoding, basic formatting) are performed.

   - The data is loaded into corresponding **staging tables** in **Google BigQuery**.

4. **Archival**
   Upon successful ingestion, each processed file is renamed using a date-based suffix (`_YYYY_MM_DD.csv`) and moved to the `./data_backup/` directory. This step preserves a historical copy for audit and recovery purposes.

5. **Transformation**
   dbt models are executed to transform staging data into structured, analytical-ready tables, following a **star schema**. These transformed tables reside in BigQuery and serve reporting and analytics use cases.

6. **Analytics and Reporting**
   The transformed data supports dashboards, KPIs, and other reporting tools used by business and analytics teams.

Technologies Used

- **Ubuntu Cron** – For scheduling and automation (to study how install and schedule cron in ubuntu server)

- Example

```
# Everyday 5 am the cron job runs the wrapper script which contains the dbt seed and dbt run
# Only if 'dbt seed' is successful it runs the 'dbt run'
#
# Output of the crontab jobs (including errors) is sent to a logfile
#
# m h  dom mon dow   command
0 5  * * * /bin/bash /home/luser/wrapper_dbt.sh >> /home/luser/cron_dbt.log 2>&1
0 13  * * * /bin/bash /home/luser/order_ingestion_dbt.sh >> /home/luser/cron_dbt.log 2>&1
```

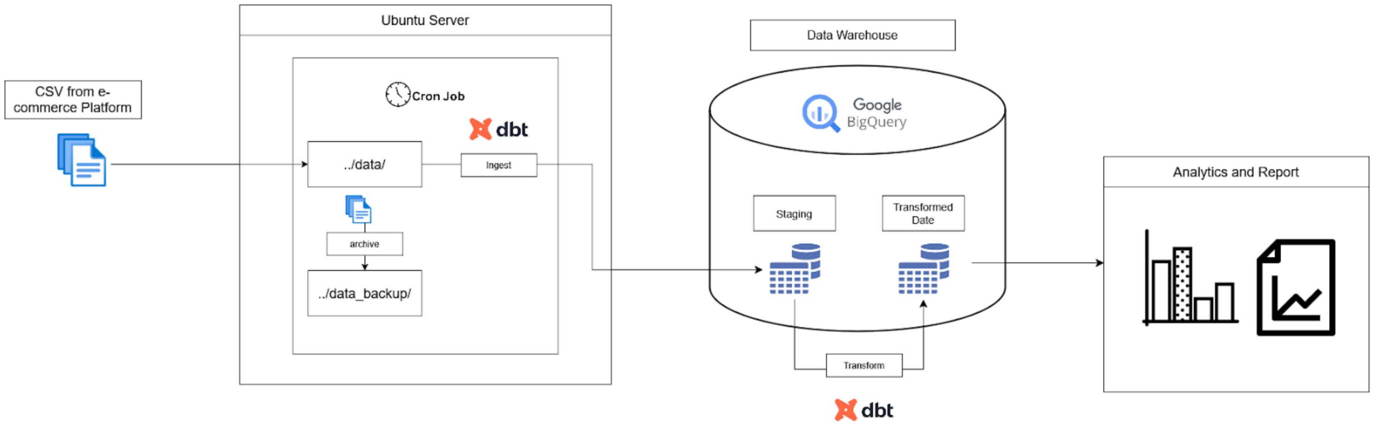
- Crontab guru, this help on define the scheduler
  The link is here: https://crontab.guru/#0_0_*_*_*

```
#!/bin/bash

# Load Conda environment properly
source /home/luser/miniconda3/etc/profile.d/conda.sh
conda activate dwh

# Change to the project directory
cd /home/luser/DS/course/Projects/dsai-module-2-final-project/proj_olist || { echo "Directory not found"; exit 1; }

# Run dbt seed
echo "Running dbt seed..."
dbt seed  > /home/luser/wrapper_dbt.log 2>&1

# Check if dbt seed was successful
if [ $? -eq 0 ]; then
    echo "dbt seed successful. Running dbt run..."
    mv  /home/luser/DS/course/Projects/dsai-module-2-final-project/proj_olist/seeds/*.csv  /home/luser/DS/course/Projects/ds
ai-module-2-final-project/proj_olist/seeds/backup/
    echo " All csv files moved to backup folder"
    dbt run >> /home/luser/wrapper_dbt.log 2>&1
    dbt test >> /home/luser/wrapper_dbt.log 2>&1
    echo "dbt run and test completed."
else
    echo "dbt seed failed. Check dbt_seed.log for details."
    exit 1
fi
```

- **dbt (Data Build Tool)** – For ingestion orchestration and data transformation
- **Google BigQuery** – As the central data warehouse (staging + transformed models)
- **Linux File System** – For organizing raw and archived data (`./data/`, `./data_backup/`)

Pipeline Diagram

The following diagram illustrates the complete flow from data ingestion to reporting:



Models

The data models are designed following a **star schema** approach, it includes two fact tables and four dimension tables, each optimized for querying in BigQuery. Relationships are defined through consistent foreign key references across entities.

Naming Conventions

- Table names follow the pattern: `fact_` or `dim_` prefix depending on their role
- Column names use lowercase with underscores for readability
- Surrogate or primary keys (e.g., `customer_id`, `product_id`) are consistently used for joins

Data dictionary (optional but helpful)

Naming conventions

Fact Tables

fact_orders

Column Name	Data Type
customer_id	object
order_id	object
order_purchase_timestamp	datetime

payment_value	float64
payment_type	object
purchase_month_and_year	object
purchase_year	int64

fact_order_items

Column Name	**Data Type**
customer_id	object
order_id	object
order_item_id	int64
product_id	object
seller_id	object
price	float64
freight_value	float64
order_purchase_timestamp	datetime
order_delivered_customer_date	datetime
order_estimated_delivery_date	datetime

Dimension Tables

dim_customer

Column Name	**Data Type**
customer_id	object

customer_unique_id	object
customer_zip_code_prefix	int64
customer_city	object
customer_state	object

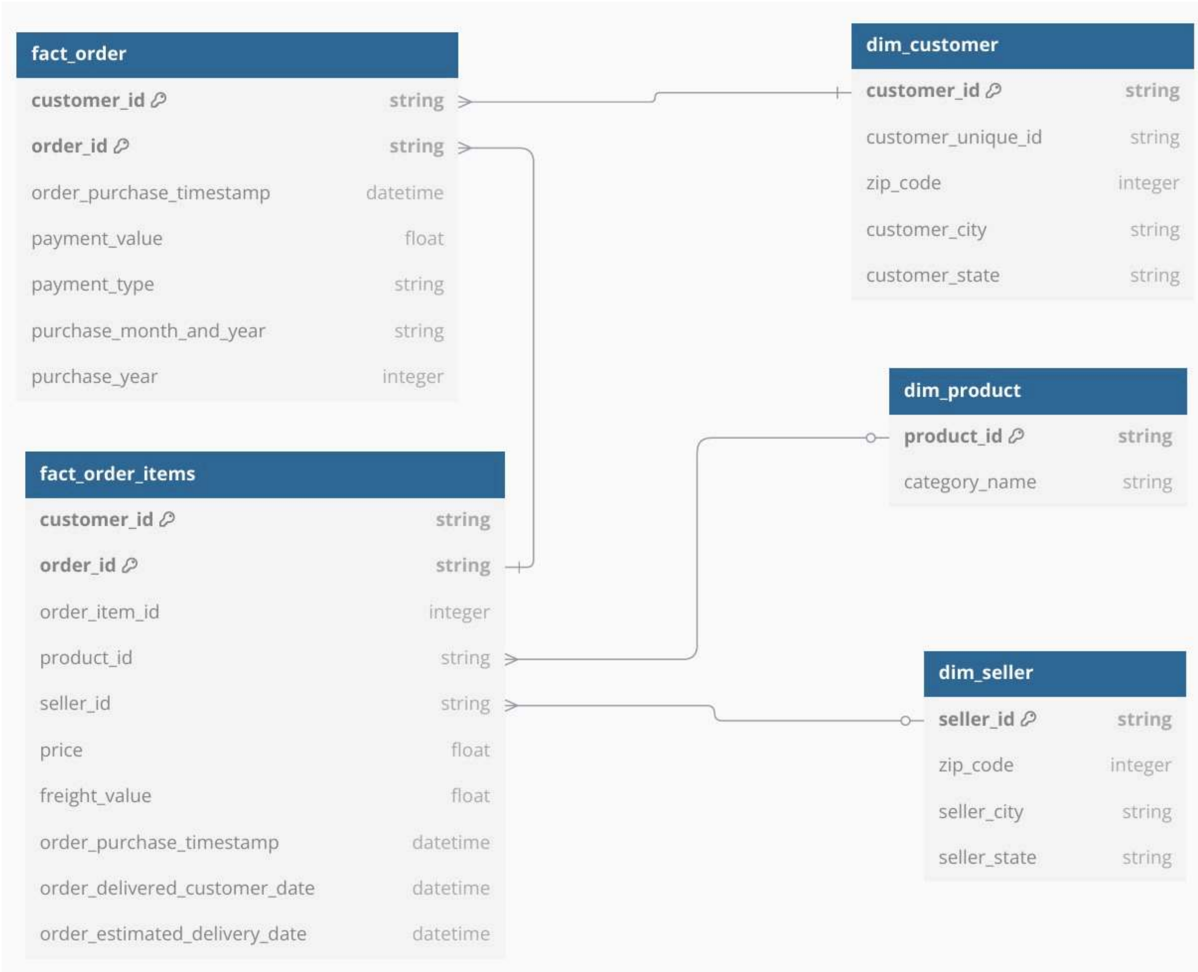
dim_product

Column Name	**Data Type**
product_id	object
product_category_name	object

dim_seller

Column Name	**Data Type**
seller_id	object
seller_zip_code_prefix	int64
seller_city	object
seller_state	object

Schema overview (tables, columns, key relationships)



Handling Slowly changing dimensions

example: dim_product tables

- incremental model for order
- scd2 for product

Slowly Changing Dimensions (Type 2)

The `dim_product` table supports **SCD Type 2** to track changes in product category names over time. Historical records are preserved by using `start_date` and `end_date` fields, as shown below:

For example:

- A change from `utilidades_domesticas` to `utilidades_domesticas_newname` is reflected by closing the old record with a valid `end_date` and inserting a new one with a fresh `start_date`.

Query results

Save results | Open in

Job information | Results | Chart | JSON | Execution details | Execution graph

Row	product_id	product_category_name	start_date	end_date
1	null	null	null	null
2	96bd76ec8810374ed1b65e291...	esporte_lazer	2017-05-01 00:00:00 UTC	9999-12-31 00:00:00 UTC
3	9dc1a7de274444849c219cff1...	utilidades_domesticas	2017-05-01 00:00:00 UTC	9999-12-31 00:00:00 UTC
4	cef67bcfe19066a932b7673e2...	bebes	2017-05-01 00:00:00 UTC	9999-12-31 00:00:00 UTC
5	3aa071139cb16b67ca9e5dea6...	artes	2017-05-01 00:00:00 UTC	9999-12-31 00:00:00 UTC
6	1e9e8ef04dbcff4541ed26657e...	perfumaria	2017-05-01 00:00:00 UTC	9999-12-31 00:00:00 UTC

Query results

Save results | Open in

Job information | Results | Chart | JSON | Execution details | Execution graph

Row	product_id	product_category_name	start_date	end_date
1	null	null	null	null
2	3aa071139cb16b67ca9e5dea6...	artes	2017-05-01 00:00:00 UTC	2020-03-31 00:00:00 UTC
3	9dc1a7de274444849c219cff1...	utilidades_domesticas	2017-05-01 00:00:00 UTC	2019-11-01 00:00:00 UTC
4	cef67bcfe19066a932b7673e2...	bebes	2017-05-01 00:00:00 UTC	9999-12-31 00:00:00 UTC
5	1e9e8ef04dbcff4541ed26657e...	perfumaria	2017-05-01 00:00:00 UTC	9999-12-31 00:00:00 UTC
6	96bd76ec8810374ed1b65e291...	esporte_lazer	2017-05-01 00:00:00 UTC	9999-12-31 00:00:00 UTC
7	9dc1a7de274444849c219cff1...	utilidades_domesticas_newna...	2019-11-02 00:00:00 UTC	9999-12-31 00:00:00 UTC
8	3aa071139cb16b67ca9e5dea6...	artes_newname	2020-04-01 00:00:00 UTC	9999-12-31 00:00:00 UTC
9	w516mse071139cb16b67ca9e...	latestproduct	2023-06-01 00:00:00 UTC	9999-12-31 00:00:00 UTC

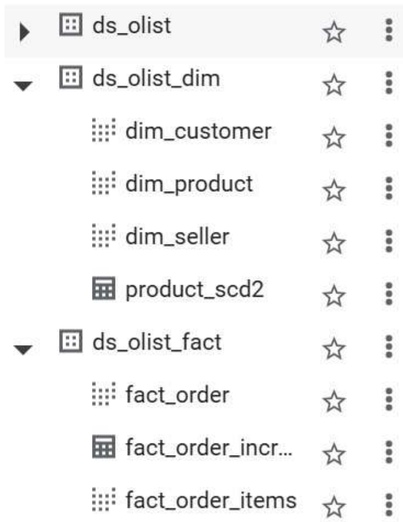
Environments & Infrastructure

suggest content:

Overview of environments (dev/test/prod)

Infrastructure used (e.g., GCP, AWS, on-prem)
Google Cloud Platform's BigQuery is used as the data warehouse. The project is named as `"final-project-57294"`

- ds_olist stores the datasets
- ds_olist_fact stores the fact tables
- ds_olist_dim stores the dimension tables



Access management basics

Monitoring & Logging

suggest content:

What is being monitored
Test cases on the ingested dataset:

Table	***column name/test***
fact_order	*customer_id*
	not null
	relationships
dim_customer	*customer_id*
	not null
	unique

fact_order	*order_purchase_timestamp*
	datetime

fact_order	*purchase_year*
	integer

fact_order	*order_id*
	not null
fact_order_items	*order_id*
	not null
	relationships

fact_order_items	*product_id*
	not null
dim_product	*product_id*
	not null
	unique
	relationships

fact_order_items	*seller_id*
	not null
	relationships
dim_product	*seller_id*
	not null
	unique

fact_order_items	*order_purchase_timestamp*
	not null
	datetime
fact_order_items	*order_delivered_customer_date*
	not null
	datetime
fact_order_items	*order_estimated_delivery_date*

	not null
	datetime
fact_order_items	*customer_id*
	not null

Tools used

Note: run dbt deps to ensure versions are updated.
dbt version 1.9.3, plugins: bigquery 1.9.1

dbt_expectations 0.10.8

macros: is_datetime.sql, is_integer.sql

Reference on validated dbt run/test cases*:*

$ dbt run

14:59:16 Running with dbt=1.9.3

14:59:17 Registered adapter: bigquery=1.9.1

14:59:18 Found 6 models, 6 seeds, 23 data tests, 876 macros

14:59:18

14:59:18 Concurrency: 1 threads (target='dev')

14:59:18

14:59:21 1 of 6 START sql view model ds_olist_dim.dim_customer .... [RUN]

14:59:24 1 of 6 OK created sql view model ds_olist_dim.dim_customer .... [CREATE VIEW (0 processed) in 3.94s]

14:59:24 2 of 6 START sql view model ds_olist_dim.dim_product .... [RUN]

14:59:26 2 of 6 OK created sql view model ds_olist_dim.dim_product .... [CREATE VIEW (0 processed) in 1.70s]

14:59:26 3 of 6 START sql view model ds_olist_dim.dim_seller .... [RUN]

14:59:27 3 of 6 OK created sql view model ds_olist_dim.dim_seller .... [CREATE VIEW (0 processed) in 1.28s]

14:59:27 4 of 6 START sql view model ds_olist_fact.fact_order .... [RUN]

14:59:29 4 of 6 OK created sql view model ds_olist_fact.fact_order .... [CREATE VIEW (0 processed) in 1.31s]

14:59:29 5 of 6 START sql incremental model ds_olist_fact.fact_order_incremental .... [RUN]

14:59:32 5 of 6 OK created sql incremental model ds_olist_fact.fact_order_incremental ... [MERGE (0.0 rows, 12.6 MiB processed) in 3.47s]

14:59:32 6 of 6 START sql view model ds_olist_fact.fact_order_items .... [RUN]

14:59:34 6 of 6 OK created sql view model ds_olist_fact.fact_order_items .... [CREATE VIEW (0 processed) in 1.33s]

14:59:34

14:59:34 Finished running 1 incremental model, 5 view models in 0 hours 0 minutes and 15.61 seconds (15.61s).

14:59:34

14:59:34 Completed successfully

14:59:34

14:59:34 Done. PASS=6 WARN=0 ERROR=0 SKIP=0 TOTAL=6

~~~~

$ dbt test

14:59:39 Running with dbt=1.9.3

14:59:40 Registered adapter: bigquery=1.9.1

14:59:41 Found 6 models, 6 seeds, 23 data tests, 876 macros

14:59:41

14:59:41 Concurrency: 1 threads (target='dev')

14:59:41

14:59:42 1 of 23 START test is_datetime_fact_order_items_order_delivered_customer_date .. [RUN]

14:59:44 1 of 23 PASS is_datetime_fact_order_items_order_delivered_customer_date .... [PASS in 2.36s]

14:59:44 2 of 23 START test is_datetime_fact_order_items_order_estimated_delivery_date .. [RUN]

14:59:46 2 of 23 PASS is_datetime_fact_order_items_order_estimated_delivery_date .... [PASS in 2.26s]

14:59:46 3 of 23 START test is_datetime_fact_order_items_order_purchase_timestamp .... [RUN]

14:59:49 3 of 23 PASS is_datetime_fact_order_items_order_purchase_timestamp .... [PASS in 2.30s]

14:59:49 4 of 23 START test is_datetime_fact_order_order_purchase_timestamp .... [RUN]

14:59:51 4 of 23 PASS is_datetime_fact_order_order_purchase_timestamp .... [PASS in 2.07s]

14:59:51 5 of 23 START test is_integer_fact_order_items_order_item_id .... [RUN]

14:59:53 5 of 23 PASS is_integer_fact_order_items_order_item_id .... [PASS in 2.25s]

14:59:53 6 of 23 START test is_integer_fact_order_purchase_year .... [RUN]

14:59:55 6 of 23 PASS is_integer_fact_order_purchase_year .... [PASS in 2.03s]

14:59:55 7 of 23 START test not_null_dim_customer_customer_id .... [RUN]

15:00:00 7 of 23 PASS not_null_dim_customer_customer_id .... [PASS in 4.65s]

15:00:00 8 of 23 START test not_null_dim_product_product_id .... [RUN]

15:00:02 8 of 23 PASS not_null_dim_product_product_id .... [PASS in 2.39s]

15:00:02 9 of 23 START test not_null_dim_seller_seller_id .... [RUN]

15:00:04 9 of 23 PASS not_null_dim_seller_seller_id .... [PASS in 2.03s]

15:00:04 10 of 23 START test not_null_fact_order_customer_id .... [RUN]

15:00:07 10 of 23 PASS not_null_fact_order_customer_id .... [PASS in 2.70s]

15:00:07 11 of 23 START test not_null_fact_order_items_customer_id .... [RUN]

15:00:09 11 of 23 PASS not_null_fact_order_items_customer_id .... [PASS in 2.53s]

15:00:09 12 of 23 START test not_null_fact_order_items_order_id .... [RUN]

15:00:13 12 of 23 PASS not_null_fact_order_items_order_id .... [PASS in 3.28s]

15:00:13 13 of 23 START test not_null_fact_order_items_order_item_id .... [RUN]

15:00:15 13 of 23 PASS not_null_fact_order_items_order_item_id .... [PASS in 2.39s]

15:00:15 14 of 23 START test not_null_fact_order_items_product_id .... [RUN]

15:00:17 14 of 23 PASS not_null_fact_order_items_product_id .... [PASS in 2.35s]

15:00:17 15 of 23 START test not_null_fact_order_items_seller_id .... [RUN]

15:00:20 15 of 23 PASS not_null_fact_order_items_seller_id .... [PASS in 2.83s]

15:00:20 16 of 23 START test not_null_fact_order_order_id .... [RUN]

15:00:23 16 of 23 PASS not_null_fact_order_order_id .... [PASS in 2.55s]

15:00:23 17 of 23 START test relationships_dim_product_product_id__product_id__ref_fact_order_items_ [RUN]

15:00:26 17 of 23 PASS relationships_dim_product_product_id__product_id__ref_fact_order_items_ [PASS in 3.31s]

15:00:26 18 of 23 START test relationships_fact_order_customer_id__customer_id__ref_dim_customer_ [RUN]

15:00:29 18 of 23 PASS relationships_fact_order_customer_id__customer_id__ref_dim_customer_ [PASS in 2.86s]

15:00:29 19 of 23 START test relationships_fact_order_items_order_id__order_id__ref_fact_order_items_ [RUN]

15:00:35 19 of 23 PASS relationships_fact_order_items_order_id__order_id__ref_fact_order_items_ [PASS in 5.89s]

15:00:35 20 of 23 START test relationships_fact_order_items_seller_id__seller_id__ref_dim_seller_ [RUN]

15:00:38 20 of 23 PASS relationships_fact_order_items_seller_id__seller_id__ref_dim_seller_ [PASS in 3.43s]

15:00:38 21 of 23 START test unique_dim_customer_customer_id .... [RUN]

15:00:41 21 of 23 PASS unique_dim_customer_customer_id .... [PASS in 2.43s]

15:00:41 22 of 23 START test unique_dim_product_product_id .... [RUN]

15:00:43 22 of 23 PASS unique_dim_product_product_id .... [PASS in 2.62s]

15:00:43 23 of 23 START test unique_dim_seller_seller_id .... [RUN]

15:00:46 23 of 23 PASS unique_dim_seller_seller_id .... [PASS in 2.22s]

15:00:46

15:00:46 Finished running 23 data tests in 0 hours 1 minutes and 5.16 seconds (65.16s).

15:00:46

15:00:46 Completed successfully

15:00:46

15:00:46 Done. PASS=23 WARN=0 ERROR=0 SKIP=0 TOTAL=23

Alerting mechanism

```
08:45:02  Finished running 25 data tests in 0 hours 1 minutes and 3.93 seconds (63.93s).
08:45:02
08:45:02  Completed with 2 errors, 0 partial successes, and 0 warnings:
08:45:02
08:45:02  Failure in test dbt_expectations_expect_column_pair_values_A_to_be_greater_than_B_fact_order_items_order_delivered_customer_date__order_purchase_timestamp__True (models/fact/schema.yml)
08:45:02    Got 69 results, configured to fail if != 0
08:45:02
08:45:02    compiled code at target/compiled/proj_olist/models/fact/schema.yml/dbt_expectations_expect_column_377a304b66edb18e2a93856fa2719be2.sql
08:45:02
08:45:02  Failure in test dbt_expectations_expect_column_pair_values_A_to_be_greater_than_B_fact_order_items_order_estimated_delivery_date__order_purchase_timestamp__True (models/fact/schema.yml)
08:45:02    Got 9 results, configured to fail if != 0
```

Date engineer checked : Ordered date is later than delivery date.This may due to the scalability of system and data ingestion failed and can't backdate when ingested.Example following:

Job information | Results | Chart | JSON | Execution details | Execution graph

Row	product_id	seller_id	price	freight_value	order_purchase_timestamp	order_delivered_customer_date	order_estimated_delivery_date
1	f60e19fc3b0ee38735e6a6fefe...	6481e96574816ead57975da2c...	7.9	7.87	2018-04-24T19:10:37	2018-04-24T18:47:46	2018-05-10T00:00:00
2	d2f5484cbffe4ca766301b21ab...	36a968b544695394e4e9d757...	12.88	7.87	2018-04-24T18:25:30	2018-04-24T15:28:41	2018-05-04T00:00:00
3	a027d4d9a0ffc6b3c2cf45636b...	2ff97219cb8622eaf3cd89b7d9...	15.9	7.87	2018-04-24T19:25:17	2018-04-24T19:18:51	2018-05-10T00:00:00
4	cce679660c66e6fbd5c8091df...	d2374cbcbb3ca4ab108653410...	17.9	12.79	2018-07-28T23:30:58	2018-07-27T12:03:15	2018-08-09T00:00:00
5	c985e917daf44dfe277983320...	95ec4458365c4d11f452ccf53...	18.9	7.39	2018-07-27T23:31:10	2018-07-25T14:58:47	2018-08-01T00:00:00
6	c985e917daf44dfe277983320...	95ec4458365c4d11f452ccf53...	18.9	7.39	2018-07-27T23:31:10	2018-07-25T14:58:47	2018-08-01T00:00:00
7	e8a1dffbef0392ef084cca4165...	411f3b52d857390502ee4e4d5...	19.8	8.3	2018-07-28T23:31:06	2018-07-26T17:03:43	2018-07-31T00:00:00
8	04262a7ed71aa34edfde0f842...	ea8482cd71df3c1969d7b9473...	19.9	8.3	2018-07-26T23:32:08	2018-07-26T16:58:30	2018-08-01T00:00:00
9	5099f7000472b634fea830444...	138dbe45fc62f1e244378131a...	19.9	8.3	2018-07-05T16:08:11	2018-07-04T15:32:20	2018-07-13T00:00:00
10	ab1f9387c0627dd24000bfbc5...	8b321bb669392f5163d04c59e...	19.9	7.4	2018-07-28T23:31:01	2018-07-25T17:35:30	2018-07-31T00:00:00
11	47376be1404bbe927766e8a9...	fcb5ace8bcc92f75707dc0f01a...	22.35	7.39	2018-04-24T18:59:19	2018-04-24T18:17:07	2018-05-08T00:00:00
12	52e5fdcb5e51164483d584c75...	92eb0f42c21942b6552362b9b...	25.77	7.39	2018-04-24T19:06:45	2018-04-24T15:32:18	2018-05-07T00:00:00
13	1613b819ab5dae53aead2dbb4...	16090f2ca825584b5a147ab24...	27.9	7.46	2018-07-26T23:31:53	2018-07-25T23:58:19	2018-07-31T00:00:00
14	e0cf79767c5b016251fe13991...	da8622b14eb17ae2831f4ac5b...	29.9	7.79	2018-07-05T16:07:28	2018-07-04T22:12:51	2018-07-16T00:00:00
15	de92134fd940e9302d27c31af...	6a8a889bde935bafa76d78487...	30.49	11.98	2018-04-24T19:23:54	2018-04-24T19:18:35	2018-05-10T00:00:00
16	de92134fd940e9302d27c31af...	6a8a889bde935bafa76d78487...	30.49	11.98	2018-04-24T19:23:54	2018-04-24T19:18:35	2018-05-10T00:00:00
17	cd322544c79e58e64b103bdce...	d566c37fa119d5e66c4e9052e...	30.9	7.39	2018-04-24T18:41:20	2018-04-24T14:03:28	2018-04-30T00:00:00
18	70e0c10acc3dd72e17f2136dd...	ce27a3cc3c8cc1ea79d11e561...	31.99	15.32	2018-08-20T15:59:54	2018-08-17T20:51:25	2018-08-23T00:00:00
19	3af6d5f9fdb78f106c003ce49d...	42b729f859728f5079499127a...	32.9	7.49	2018-07-05T16:17:20	2018-07-05T12:48:37	2018-07-16T00:00:00
20	3bd016185be10ea4b4b2baa6e...	ff4e2d38692ce827b1a4f4b819...	35.0	16.44	2018-07-05T16:25:45	2018-07-05T12:58:37	2018-07-30T00:00:00

Results per page: 50 1 – 50 of 69

Deployment & Version Control

suggest content:

Code repository location

Deployment process (manual/CI-CD)

Branching strategy (if any)

Known Issues / TODOs

suggest content:

Current limitations

1.Process issues : Scalability Issue: Orders are not ingested when a corresponding sales record is encountered, causing the `order_purchase_date` to be later than the `delivery_date`.

2. Order Fulfillment Compliance Gaps

Issue: Critical delivery date fields exhibit systemic problems:

- ***`shipping_limit_date` not enforced in all orders***
- ***`order_delivered_customer_date` missing for some completed shipments***

3. The existing `product_category` classification proves insufficient for strategic decision-making due to:

- ***Overly broad groupings masking top-performing SKUs***
- ***Inability to identify true drivers within categories***

Planned improvements

1. Current State Analysis

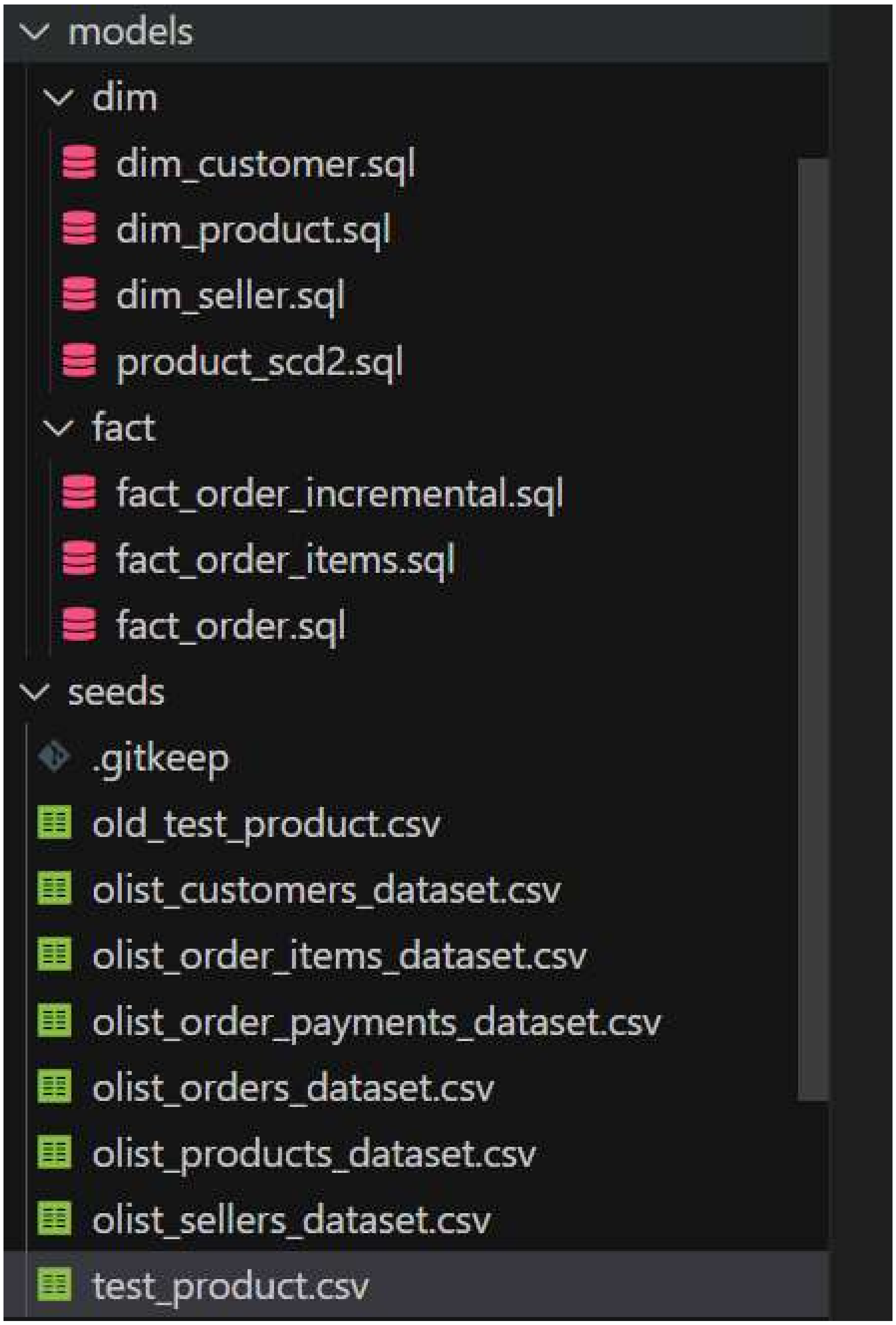
- *Conducted comprehensive evaluation of existing technical infrastructure*
- *Identified critical gaps in data integration across sales channels*
- *Assessed limitations in customer behavior tracking capabilities*

2. Business Value Mapping

- *Quantified operational inefficiencies and revenue leakage points*
- *Established clear correlations between technical constraints and business outcomes*
- *Prioritized pain points based on financial impact and strategic importance*

3. Future-State Planning

- *Designed integrated architecture to unify multi-channel sales data*
- *Developed behavioral analytics framework to track cross-platform customer journeys*
- *Proposed phased implementation roadmap aligned with business growth targets*


models
dim
dim_customer.sql
dim_product.sql
dim_seller.sql
product_scd2.sql
fact
fact_order_incremental.sql
fact_order_items.sql
fact_order.sql
seeds
.gitkeep
old_test_product.csv
olist_customers_dataset.csv
olist_order_items_dataset.csv
olist_order_payments_dataset.csv
olist_orders_dataset.csv
olist_products_dataset.csv
olist_sellers_dataset.csv
test_product.csv