

TECHNICAL TEST - DATA ENGINEER

TRIPUTRA AGRO PERSADA

1. Buat akun **Snowflake Free Trial**

2. Download dataset dari Kaggle

<https://www.kaggle.com/datasets/olistbr/brazilian-e-commerce>

Tables yang digunakan:

- o orders.csv
- o order_items.csv
- o customers.csv
- o products.csv
- o payments.csv

|

3. Upload CSV ke Snowflake stage

4. Create raw tables:

- o raw_orders
- o raw_order_items
- o raw_customers
- o raw_products
- o raw_payments

Task 1

Buat:

- o 1 fact table: fact_sales

Jawab : Create Table Fact_sales

The screenshot shows a SQL editor interface with the following details:

- File Path:** My Workspace > test_iftria.sql
- Code Content:**

```
42     b.product_id,
43     b.customer_id,
44     b.order_status,
45     b.order_purchase_timestamp,
46
47     -- metrics
48     b.price,
49     b.freight_value,
50
51     -- check allocated payment
52     (b.price / otp.total_order_price)
53     * op.total_payment_value AS payment_value_allocated
54
55     FROM order_item_check b
56     LEFT JOIN order_total_price otp
57       ON b.order_id = otp.order_id
58     LEFT JOIN order_payment op
59       ON b.order_id = op.order_id
60   ;
61   --select * from OLIST_DB.MART.FACT_SALES
62   --limit 5;
63 
```
- Results (just now):**
 - Table
 - Chart

A status message: 1 Table FACT_SALES successfully created.

limit 5 fact tables

The screenshot shows a database query results window. The SQL query is:

```

00
01 ; select * from OLIST_DB.MART.FACT_SALES
02
03 limit 5;
04
05 results (just now) Add to Chat Explain Quick edit Format

```

The table has the following columns: ORDER_ID, ORDER_ITEM_ID, PRODUCT_ID, CUSTOMER_ID, ORDER_STATUS, and ORDER_PURCHASE_TIMESTAMP.

	ORDER_ID	ORDER_ITEM_ID	PRODUCT_ID	CUSTOMER_ID	ORDER_STATUS	ORDER_PURCHASE_TIMESTAMP
1	00010242fb8c5a6d1ba2dd792cb16214	1	424473e06e7ecb4970a6e2683c13e01	3ce436f183e68e0787b285a838d01a	delivered	2017-09-13 08:59:02.000
2	0001877f203202557190d7144bd3	1	e5f7d52b802189e658865ca93d83a8f	f6dd3ec081db4e3987629fe6b28e5cce	delivered	2017-04-26 10:53:06.000
3	0002299ec398224ef6ca0657a4fc703e	1	c777355d18b72b67abbed9df44fd0fd	6489ae5e433f38e93dfad4372dab8d3	delivered	2018-01-14 14:33:31.000
4	00024abcdf0a6aa9a931b03814c75	1	7834da52a4610f1595efaf21f472fc	d4eb0395c5e0431ee92fc0cd980e5a0e	delivered	2018-08-08 10:00:35.000
5	00042b2bcf59d7ce69dfabb4e55b4fd9	1	ac6c3623068f30de03045885e4e10089	58dbd0b2d70208bf40e62cd34e84d795	delivered	2017-02-04 13:57:51.000

Penjelasan : fact_sale merepresentasikan aktivitas penjualan (sales) pada level order item, di mana 1 baris = 1 produk yang terjual dalam satu order.

Fact table ini menjadi sumber utama untuk analisis revenue, volume penjualan, dan performa customer maupun produk.

Sumber yang saya gunakan meliputi :

- RAW_ORDER_ITEM (pondasi), karena terdapat relasi order terhadap product, terdapat harga per item beserta biaya pengirimannya.
- RAW_ORDER (informasi order), karena terdapat relasi terhadap customer, status order dan waktu pembelian
- RAW Customer (informasi customer), table ini digunakan untuk memberikan detail customer yang melakukan pemesanan dan akan menjadi relasi untuk pembuatan dim_customer
- RAW PAYMENT (informasi finansial), karena payment berada di level order (1 row per order), sedangkan fact_sale berada di level order item (1 row per product per order), dilakukan pendekatan.

$\text{allocated_payment} = (\text{item_price} / \text{total_order_price}) \times \text{total_payment_value}$

o Dimension tables:

§ dim_customers

Jawab :

The screenshot shows a database script creation window. The script is:

```

80
81 CREATE OR REPLACE TABLE OLIST_DB.MART.DIM_CUSTOMER AS
82 SELECT DISTINCT
83   TRIM(UPPER(customer_id)) AS customer_id,
84   customer_unique_id,
85   customer_zip_code_prefix,
86   INITCAP(customer_city) AS customer_city,
87   UPPER(customer_state) AS customer_state
88   FROM OLIST_DB.RAW.RAW_CUSTOMER;
89
90
91

```

The results window shows:

Results (just now)

Table DIM_CUSTOMER successfully created.

Penjelasan : pada dim_customer, dilakukan data cleansing pada script diatas dan dilakukan pengecekan jumlah terhadap fact_sales untuk

menghindari data yang tidak sesuai

```
73  --> explain check
74  SELECT COUNT(*)
75  FROM MART.FACT_SALES f
76  LEFT JOIN MART.DIM_CUSTOMER c
77  ON f.customer_id = c.customer_id
78  WHERE c.customer_id IS NULL;
79
80
81  -->CREATE OR REPLACE TABLE OLIST_DB.MART.DIM_CUSTOMER AS
82  SELECT DISTINCT
83  TRIM(UPPER(customer_id)) AS customer_id,
84  customer.first_name,
85  customer.zip_code_prefix,
86  INITCAP(customer_city) AS customer_city,
87  INITCAP(customer_state) AS customer_state
88
89
90  Results (just now)
```

Table Chart

COUNT(*)
112650

§ dim_products

```
90
91  CREATE OR REPLACE TABLE OLIST_DB.MART.DIM_PRODUCT AS
92  SELECT DISTINCT
93  TRIM(UPPER(product_id)) AS product_id,
94  INITCAP(product_category_name) AS product_category_name,
95  product_name_length,
96  product_description_length,
97  product_photos_qty,
98  product_weight_g,
99  product_length_cm,
100 product_height_cm,
101 product_width_cm
102 FROM OLIST_DB.RAW.RAW_PRODUCT;
103
```

Results (just now)

Table Chart

A status
Table DIM_PRODUCT successfully created.

Penjelasan : Dilakukan cleansing data dan cross check agar tidak ada duplicate

```
104  SELECT
105  COUNT(*) AS total_rows,
106  COUNT(DISTINCT product_id) AS unique_product
107  FROM OLIST_DB.MART.DIM_PRODUCTS;
108
```

Results (just now)

Table Chart

# TOTAL_ROWS	# UNIQUE_PRODUCT
32851	32951

§ dim_date

Jawab :

```
151  CREATE OR REPLACE TABLE OLIST_DB.MART.DIM_DATE AS
152  SELECT DISTINCT
153  CAST(order_purchase_timestamp AS DATE) AS date_id,
154  DAY(order_purchase_timestamp) AS day,
155  MONTH(order_purchase_timestamp) AS month,
156  INITCAP(TO_CHAR(order_purchase_timestamp,'Month')) AS month_name,
157  '0' || TO_CHAR(order_purchase_timestamp,'0') AS quarter,
158  YEAR(order_purchase_timestamp) AS year,
159  DAYOFWEEKISO(order_purchase_timestamp) AS day_of_week,
160  CASE WHEN DAYOFWEEKISO(order_purchase_timestamp) IN (6,7) THEN TRUE ELSE FALSE END AS is_weekend,
161  INITCAP(TO_CHAR(order_purchase_timestamp,'Day')) AS day_name,
162  WEEKISO(order_purchase_timestamp) AS week_of_year
163  FROM OLIST_DB.RAW.RAW_ORDERS;
164
165  select * from OLIST_DB.MART.dim_date
166  limit 10
```

Results (just now)

Table Chart

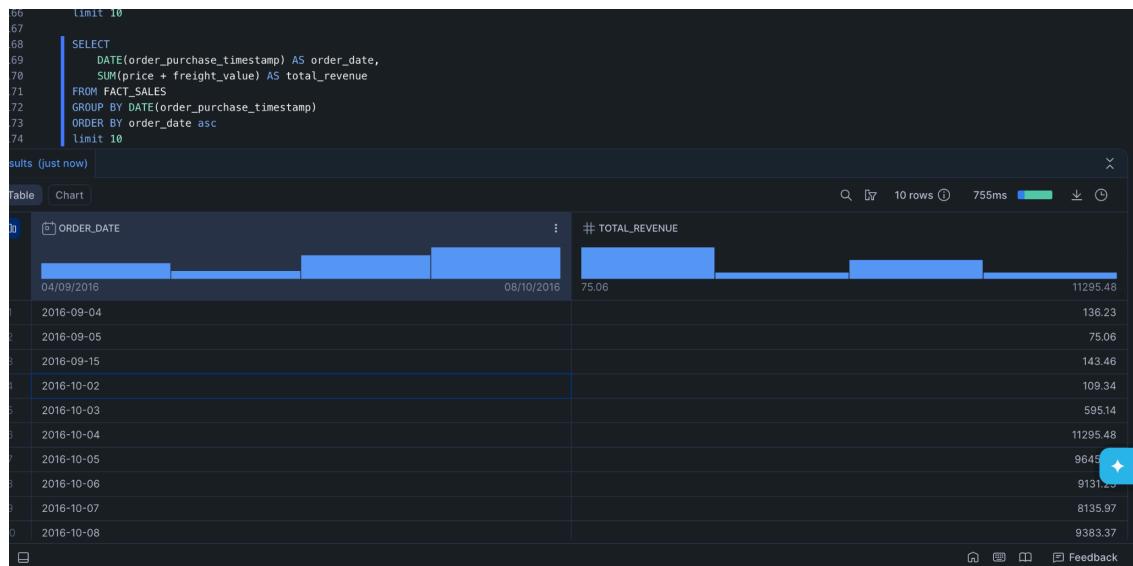
DATE_ID	DAY	MONTH	MONTH_NAME	QUARTER	YEAR	DAY_OF_WEEK	IS_WEEKEND	DAY_NAME	WEEK_OF_YEAR										
13/07/2017, 22/08/2017	1	26	1	Augth	20.0%	Febth	20.0%	100.0%	QQ	2017	2018	1	4	false	100.0%	Day	100.0%	2	34
2017-07-13	13	7	Julth	QQ	2017	2018	4	FALSE	Day	28									
2018-06-07	7	6	Junth	QQ	2018	4	FALSE	Day	23										
2018-03-15	15	3	Marth	QQ	2018	4	FALSE	Day	11										
2018-01-08	8	1	Janth	QQ	2018	1	FALSE	Day	2										

Task 2

Buat SQL query untuk menjawab:

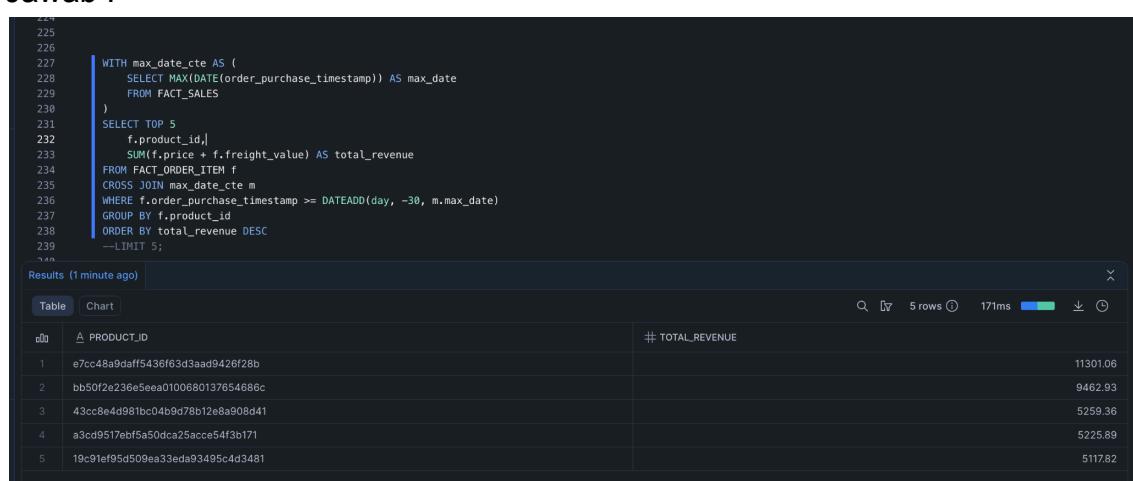
1. Total revenue per day

Jawab :



2. Top 5 products by revenue in last 30 days

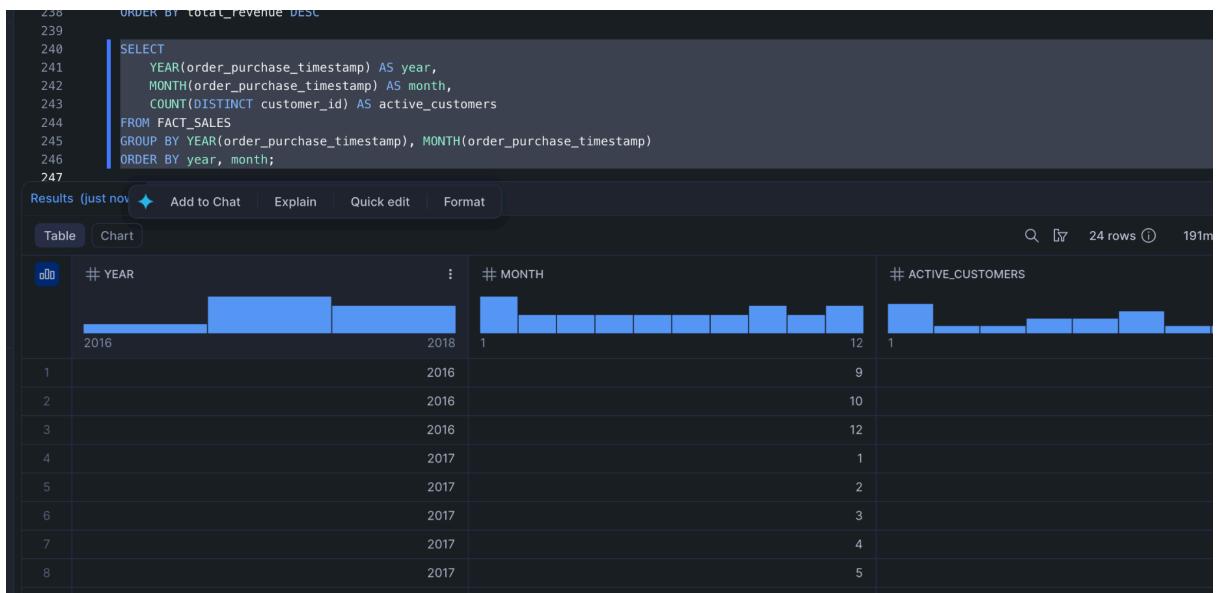
Jawab :



Penjelasan : disini untuk revenue 30 hari terhari, menggunakan 30 hari terakhir dari Maxdate

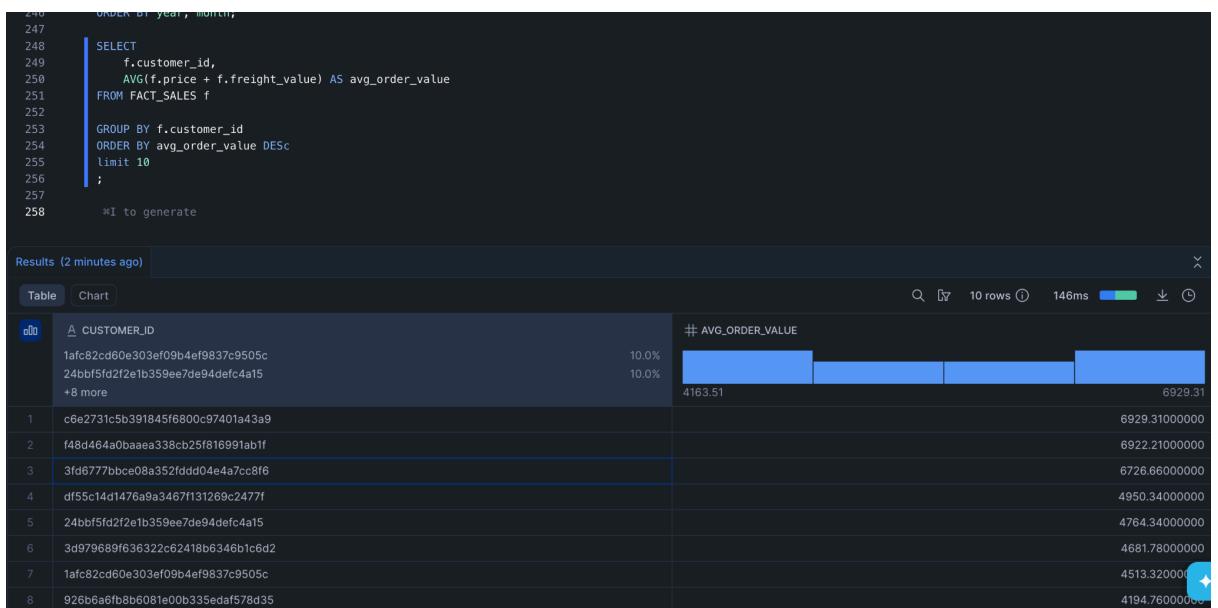
3. Monthly active customers

Jawab



4. Average order value per customer

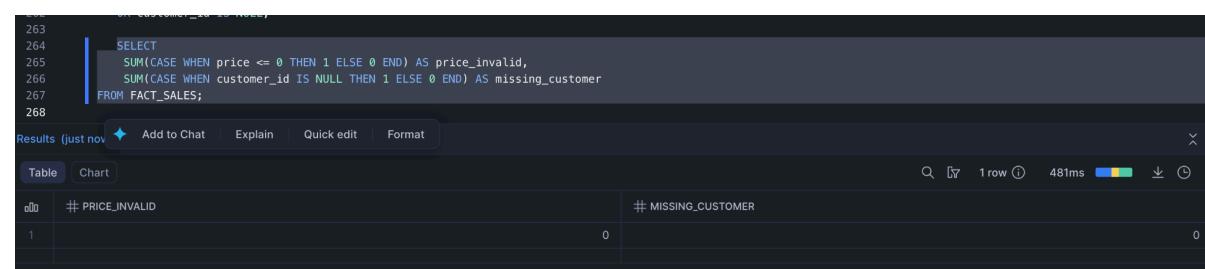
Jawab :



5. Data anomaly:

- price <= 0
- quantity <= 0
- missing customer_id

Jawab :



Task 3

Buat dbt project dengan struktur:
models/
staging/
 stg_orders.sql

 stg_customers.sql
 stg_products.sql
marts/
 dim_customers.sql
 dim_products.sql
 fact_sales.sql

jawab

The screenshot shows the Snowflake Database Explorer interface. On the left, there's a sidebar with various navigation options like 'Work with data', 'Horizon Catalog', and 'Manage'. The 'Catalog' option is currently selected. The main area displays the 'Database Explorer' for the 'HORIZON CATALOG'. Under the 'Databases' tab, it shows the 'OLIST_DB' database structure. The 'INFORMATION_SCHEMA' and 'MART' schemas are expanded, showing tables like 'DIM_CUSTOMER', 'DIM_DATE', 'DIM_PRODUCT', and 'FACT_SALES'. The 'RAW' schema is also expanded, showing tables like 'RAW_CUSTOMER', 'RAW_ORDERS', 'RAW_ORDER_ITEM', 'RAW_PAYMENT', and 'RAW_PRODUCT'. The 'Views' section is also visible. At the bottom right, there's a 'Microsoft' watermark.

penjelasan: untuk dbt saya gagal dalam koneksi dilocal, sehingga saya coba bikin alternatif disnowflake

Task 4

Buat file schema.yml berisi tests berikut:

- not_null pada primary key
- unique pada order_id
- accepted_values pada order_status
- relationships:
 - fact_sales.customer_id → dim_customers.customer_id
 - fact_sales.product_id → dim_products.product_id

Jawab:

models:

- name: FACT_SALES
 - columns:
 - name: sales_id
 - tests:
 - not_null
 - unique
 - name: order_id
 - tests:
 - not_null
 - unique
 - name: customer_id
 - tests:
 - not_null
 - relationships:
 - to: ref('DIM_CUSTOMER')
 - field: customer_id
 - name: product_id
 - tests:
 - not_null
 - relationships:
 - to: ref('DIM_PRODUCT')
 - field: product_id

```
- name: order_status
  tests:
    - not_null
    - accepted_values:
        values: ['pending', 'completed', 'canceled', 'returned']

- name: DIM_CUSTOMER
  columns:
    - name: customer_id
      tests:
        - not_null
        - unique

- name: DIM_PRODUCT
  columns:
    - name: product_id
      tests:
        - not_null
        - unique
```

The screenshot shows a Snowflake workspace interface. On the left, there's a sidebar with icons for file operations like search, add new, and database management. Below it is a 'Worksheets (Legacy)' section with a search bar and two entries: 'Load sample data from AWS S3 with ...' and '2026-02-10 6:40pm'. Under 'Database Explorer', there are tabs for 'Objects' and 'Data Products', with a search bar and a filter icon. The main area displays a Jupyter Notebook titled 'Untitled.ipynb' containing a Markdown cell with a JSON schema definition:

```
version: 2

models:
  - name: FACT_SALES
    columns:
      - name: sales_id
        tests:
          - not_null
          - unique

      - name: order_id
        tests:
          - not_null
          - unique

      - name: customer_id
        tests:
          - not_null
          - relationships:
              to: ref('DIM_CUSTOMER')
              field: customer_id

      - name: product_id
        tests:
          - not_null
          - relationships:
```

Below the notebook is a terminal window showing the command '47281e63384a0224\$'.

penjelasan : lanjutan dari task 3, saya masih gagal dalam membuat koneksi

DELIVERABLE

GitHub repository berisi:

1. SQL scripts
2. dbt project
3. Screenshot :
 - o dbt run
 - o dbt test
 - o Snowflake tables