Data Mart Creation

Helmy Naufal Aziz
Email | Linkedin | Github

Virtual Internship Experience - Kimia Farma

Dataset

There are 6 tables with the same names and columns. I assume this data is obtained from the data lake from 2 different sources. Therefore, it needs to combine the data first so that it becomes comprehensive data.

Dataset Overview

Dataset Overview

Dataset Extraction

Since the data source format is CSV file, we need to extract the data first into a databases

```
CREATE TABLE pelanggan (
    id_customer VARCHAR(10) PRIMARY KEY,
    levels VARCHAR(10),
    nama VARCHAR(30),
    id_cabang_sales VARCHAR(10),
    cabang_sales VARCHAR(30),
    id_group VARCHAR(10),
    group_type VARCHAR(10)
);

COPY PELANGGAN(ID_CUSTOMER, LEVELS, NAMA, ID_CABANG_SALES, CABANG_SALES, ID_GROUP, GROUP_TYPE)
FROM 'D:/VIX/Kimia Farma/dataset/pelanggan.csv'
DELIMITER ';'
CSV HEADER;
```

Complete query

Set Primary Key

Select the unique column of each table as a primary key.

1. penjualan & penjualan_ds
   There is no unique column in both tables. So, I define the primary key by combining id_invoice with id_barang.

   id_sales = id_invoice_id_barang

2. pelanggan & pelanggan_ds → id_customer
3. barang & barang_ds → kode_barang

Data Mart Design

01

Combining Data

From 6 tables obtained, it will combined into 3 tables:

1. **Sales**
   penjualan and penjualan_ds
2. **Customers**
   pelanggan and pelanggan_ds
3. **Products**
   barang and barang_ds

02

Table Base

- The table base will be created from the 3 tables that have been combined before.
- ‘Id_sales’ column will be the primary key
- Table base will be stored in data warehouse

03

Table Aggregate

- The are several table aggregate will be created from table base.
- table aggregate will be stored in data mart

Data Mart Design

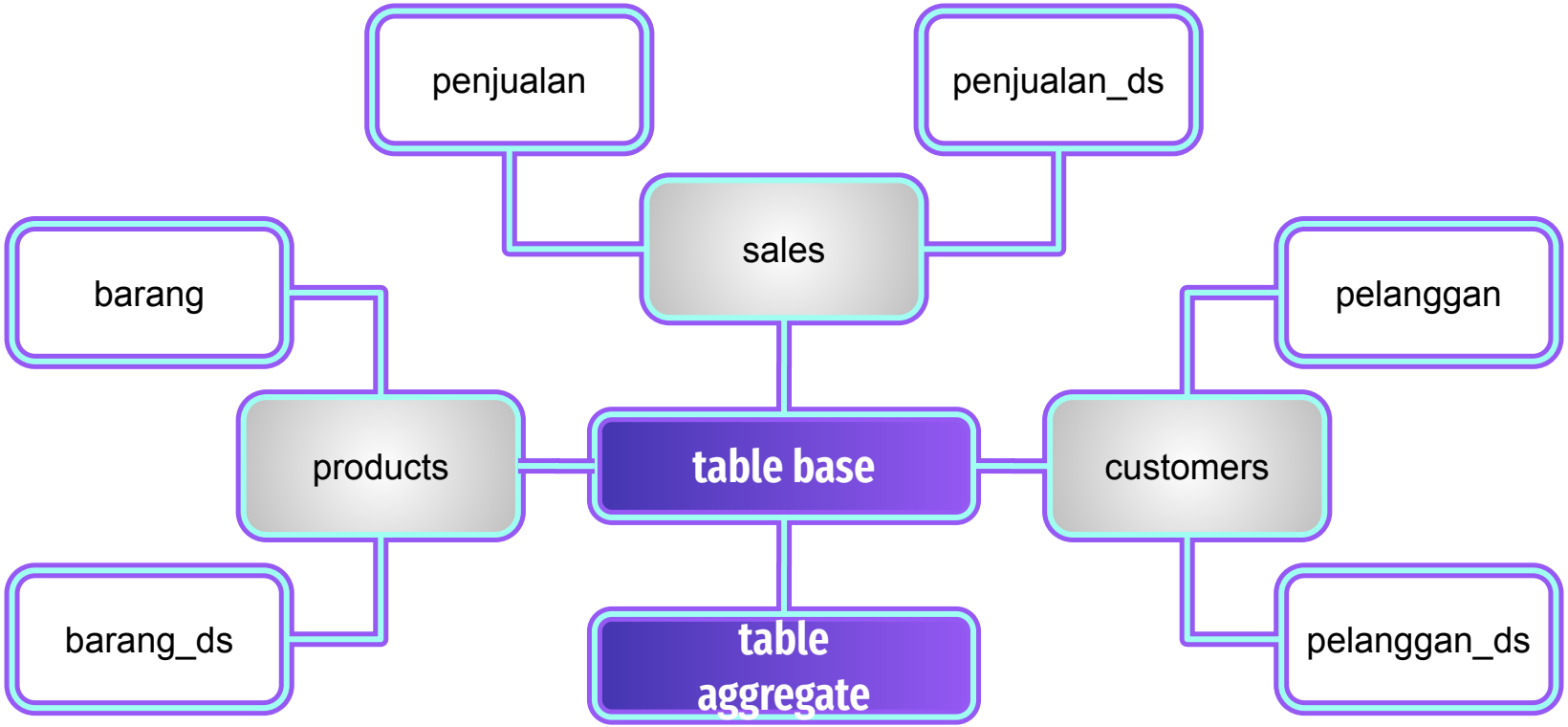


Table Base (sales_report)

column	data type	description	transformation
id_sales	varchar	nomor penjualan	Id_invoice, id_barang
id_invoice	varchar	nomor pemesanan pelanggan	
tanggal	date	tanggal pemesanan	
id_customer	varchar	nomor pelanggan	
nama	varchar	nama tempat penjualan	
cabang	varchar	kota tempat penjualan	
nama_barang	varchar	produk yang dipesan	

Table Base (sales_report)

column	data type	description	transformation
harga	int	harga produk	
jumlah_barang	int	jumlah pemesanan produk	
kemasan	varchar	jenis kemasan produk	
total_harga	int	total harga	harga * jumlah_barang
brand	varchar	brand produk	
distributor	varchar	nama distributor	
group_type	varchar	tipe tempat penjualan	

Table Base (sales_report)

```
CREATE TEMPORARY TABLE sales AS
   (SELECT id_sales, id_invoice, tanggal, id_customer, id_barang, jumlah_barang
    FROM penjualan);

MERGE INTO sales s USING penjualan_ds pds ON s.id_sales = pds.id_sales
WHEN matched THEN UPDATE
SET id_sales = s.id_sales, id_invoice = s.id_invoice, tanggal = s.tanggal, id_customer = s.id_customer,
    id_barang = s.id_barang, jumlah_barang = s.jumlah_barang
WHEN NOT matched THEN
   INSERT (id_sales, id_invoice, tanggal, id_customer, id_barang,jumlah_barang)
   VALUES (pds.id_sales, pds.id_invoice, pds.tanggal, pds.id_customer, pds.id_barang, pds.jumlah_barang);

CREATE TABLE sales_report AS
   (WITH customers AS
          (SELECT p.id_customer, p.nama, p.cabang_sales AS cabang,
                  pds.id_distributor AS distributor, p.group_type
           FROM pelanggan p JOIN pelanggan_ds pds ON p.id_customer = pds.id_customer),
         products AS
          (SELECT b.kode_barang, b.nama_barang, b.kemasan, bds.brand, bds.harga
           FROM barang b JOIN barang_ds bds ON b.kode_barang = bds.kode_barang)
   SELECT s.id_sales, s.id_invoice, s.tanggal, c.id_customer, c.nama, c.cabang, p.nama_barang, p.harga,
          s.jumlah_barang, p.kemasan, (p.harga * s.jumlah_barang) AS total_harga,
          p.brand, c.distributor, c.group_type
   FROM sales s
   JOIN customers c ON s.id_customer = c.id_customer
   JOIN products p ON s.id_barang = p.kode_barang);
```

Complete query

Data Cleaning

Several rows have the same value on id_invoice but different values on other columns, like the following example:

id_sales character varying (30)	id_invoice character varying (10)	tanggal date	id_customer character varying (10)	nama character varying (30)	cabang character varying (30)
IN6113_BRG0001	IN6113	2022-01-27	CUST55400	APOTEK SINAR JAYA	Bekasi
IN6113_BRG0002	IN6113	2022-02-01	CUST55382	KLINIK GM	Jakarta

I assume each order with the same id_invoice will have the same order date (tanggal) and id_customer. Therefore I decided to change id_invoice on one of the rows using the following query:

```
UPDATE sales_report
SET id_invoice = replace(replace(replace(replace(id_invoice, 'IN6028', 'IN6328'),
                                                             'IN6064', 'IN6329'),
                                                             'IN6113', 'IN6330'),
                                                             'IN6131', 'IN6331')
WHERE id_sales in ('IN6028_BRG0001','IN6064_BRG0006','IN6113_BRG0001','IN6131_BRG0009');

UPDATE sales_report
SET id_sales = concat(id_invoice, '_', split_part(id_sales, '_',2));
```

Table Aggregate (Monthly Revenue)

column	data type	description	transformation
month	text	bulan pemesanan	tanggal
revenue	int	penghasilan	harga * jumlah_barang

Table Aggregate (Monthly Revenue)

```sql
-- Monthly Revenue
SELECT to_char(tanggal, 'Month') AS MONTH, sum(total_harga) AS revenue
FROM sales_report
GROUP BY 1
ORDER BY min(tanggal);
```

Complete query

Table Aggregate (Total Transaction)

column	data type	description	transformation
month	text	bulan pemesanan	tanggal
transaction	int	total transaksi	id_invoice

Table Aggregate (Total Transaction)

```sql
-- Total Transaction
SELECT to_char(tanggal, 'Month') AS MONTH, count(DISTINCT id_invoice) AS TRANSACTION
FROM sales_report
GROUP BY 1
ORDER BY min(tanggal);
```

Complete query

Table Aggregate (Average Product Sold)

column	data type	description	transformation
month	text	bulan pemesanan	tanggal
avg_product	numeric	rata-rata penjualan produk	jumlah_barang

Table Aggregate (Average Product Sold)

```sql
-- Average Product Sold
SELECT to_char(tanggal, 'Month') AS MONTH,
       round(avg(jumlah_barang),1) AS avg_product
FROM sales_report
GROUP BY 1
ORDER BY min(tanggal);
```

Complete query

Table Aggregate (Seller Type)

column	data type	description	transformation
seller_type	varchar	jenis tempat penjualan	group_type
total	int	jumlah masing-masing jenis	id_invoice

Table Aggregate (Seller Type)

```sql
-- Seller Type
SELECT group_type as seller_type, count(DISTINCT id_invoice) AS total
FROM sales_report
GROUP BY 1;
```

Complete query

Table Aggregate (Top Product)

column	data type	description	transformation
nama_barang	varchar	nama produk	
avg_product	numeric	rata-rata penjualan produk	jumlah_barang

Table Aggregate (Top Product)

```sql
-- Top Product
SELECT nama_barang, round(avg(jumlah_barang), 1) AS avg_product
FROM sales_report
GROUP BY 1
ORDER BY 2 DESC;
```

Complete query

Table Aggregate (Top Seller)

column	data type	description	transformation
nama	text	nama tempat penjualan	
transaction	int	Total transaksi	
avg_product	numeric	rata-rata penjualan produk	jumlah_barang

Table Aggregate (Top Seller)

```sql
-- Top Seller
WITH trx AS
      (SELECT nama, count(DISTINCT id_invoice) AS transaction
       FROM sales_report
       GROUP BY 1),
     sell AS
      (SELECT nama, round(avg(jumlah_barang), 2) AS avg_product
       FROM sales_report
       GROUP BY 1)
SELECT t.nama, t.transaction, s.avg_product
FROM trx t
JOIN sell s ON t.nama = s.nama
ORDER BY 3 DESC;
```

Complete query

Table Aggregate (Region Sales)

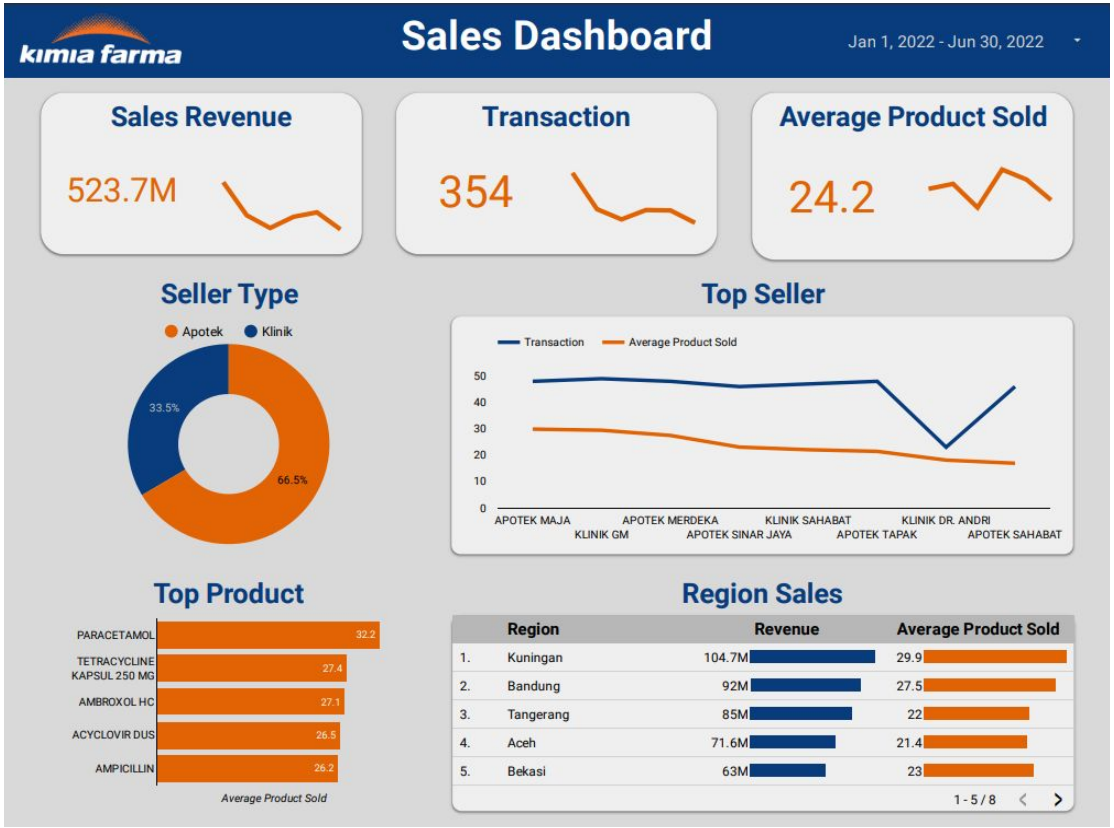
column	data type	description	transformation
nama	text	nama tempat penjualan	
revenue	int	penghasilan	harga * jumlah_barang
avg_product	numeric	rata-rata penjualan produk	jumlah_barang

Table Aggregate (Region Sales)

```sql
-- Region Sales
WITH rev AS
      (SELECT cabang, sum(total_harga) AS revenue
       FROM sales_report
       GROUP BY 1),
     sell AS
      (SELECT cabang, round(avg(jumlah_barang), 1) AS avg_product
       FROM sales_report
       GROUP BY 1)
SELECT r.cabang, r.revenue, s.avg_product
FROM rev r
JOIN sell s ON r.cabang = s.cabang
ORDER BY 2 DESC;
```

Complete query

Visualization



Looker Studio