Use case 1 : DWH Commerce

Database operasional :

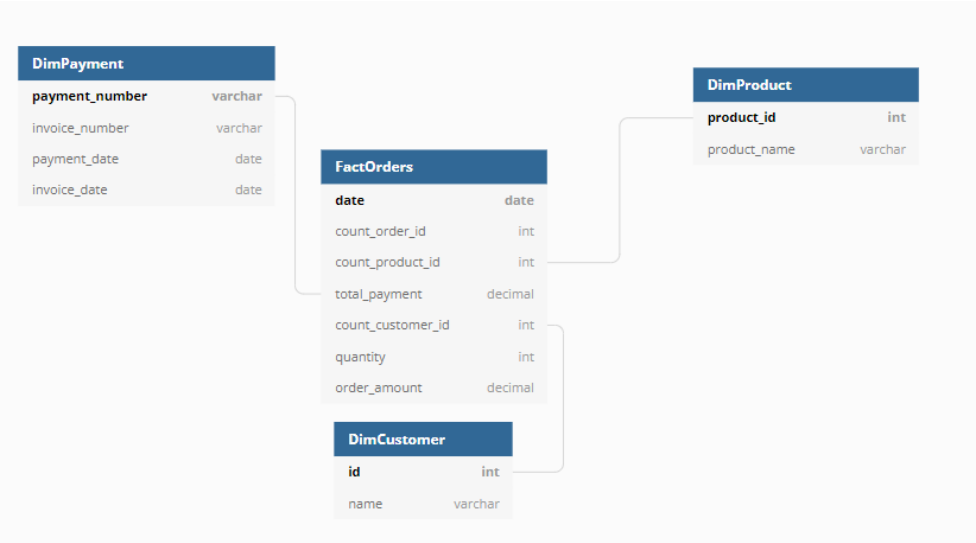
Link : https://dbdiagram.io/d/61d84b1ff8370f0a2ee31957

Database operasional menampung data-data bersifat transaksional yang membutuhkan accessibility dan availability. Beberapa cara untuk meng-improve performa database operasional adalah :

1) Menggunakan tipe data integer sebagai primary key, karena proses select membutuhkan waktu yang lebih banyak untuk karakter yang panjang, juga untuk data alfabet
2) Menggunakan indexing selain id table, bisa juga menggunakan kolom-kolom yang paling sering digunakan sebagai keyword pencarian
3) Melalui custom query, dengan meminimalisir join. Oleh sebab itu, table payment dan invoice dilebur menjadi satu. Pada dasarnya, payment dilakukan berdasarkan invoice, sehingga 2 table ini relatif sama

Database operasional tersebut ditransformasi ke dalam 2 skema Data Warehouse yaitu :

A. Star Schema



Beberapa kelebihan dari skema ini adalah :

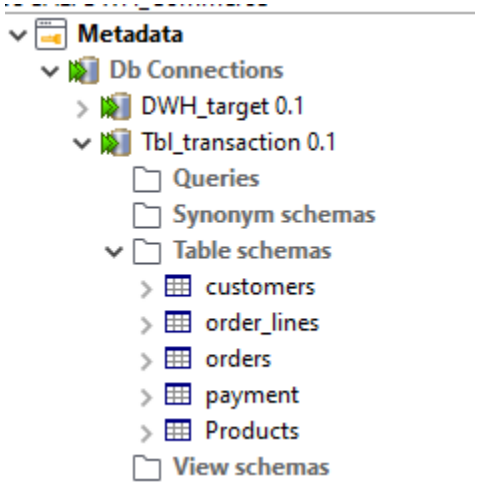
1) Performa tinggi karena bentuk yang partial atau full denormalized
2) Interpretability yang lebih mudah karena setiap variabel dimension berada di 1 table dimensi
3) Development yang sederhana dan disk space yang optimal

Beberapa kelemahan :

1) Struktur denormalisasi yang membuat proses update juga analysis lebih sulit
2) Struktur denormalisasi juga mengakibatkan rendahnya integritas antar data
3) Analysis yang membutuhkan relasi antar tabel dengan level granularity yang tinggi sedikit sulit, karena key-relasi sudah diagregat dan variable dimension sudah di-denormalisasi

Proses ETL sederhana dapat dilakukan menggunakan open source tools, Talend Open Studio :

1) Membuat koneksi ke data source, lalu menarik skemanya



2) Memuat data-data yang akan ditransformasi dari existing table,

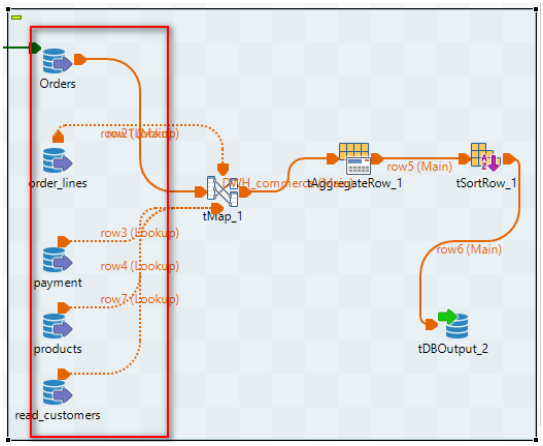
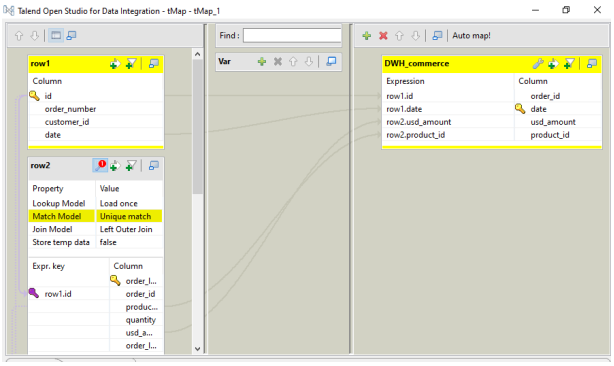


Table order dijadikan sebagai main table, karena menjadi fokus analisis yang akan dimuat ke data warehouse

3) Memapping kolom tabel input ke table final yang akan menjadi Fact Table di data warehouse :

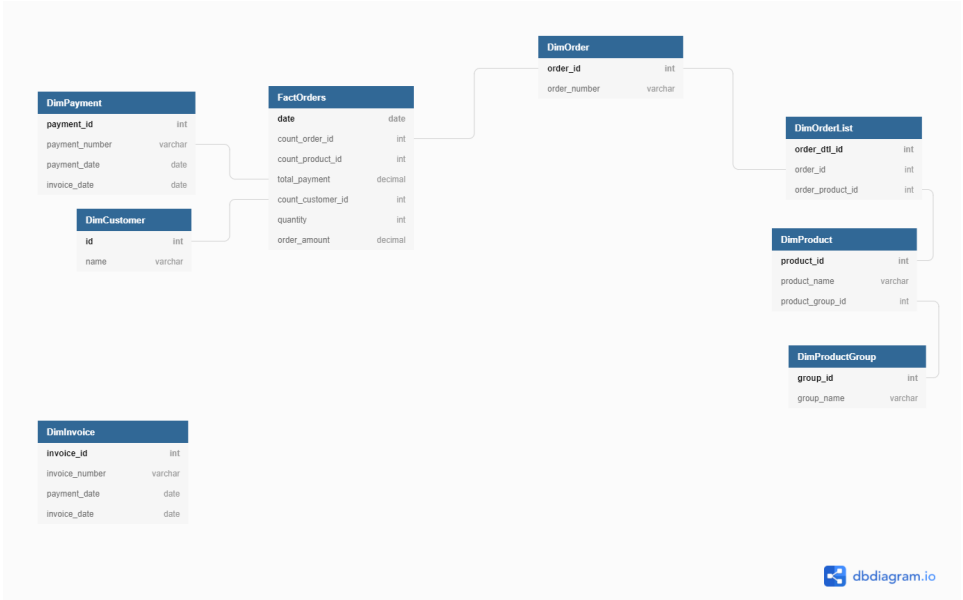


4) Agregat data, lalu dump ke Fact Table simulasi yang dibuat di dalam PostgreSql

count_order	order_date	total_amount	count_product
1	2021-01-23	123.2	1
1	2020-12-04	3.6	1
1	2020-11-16	70.4	1
1	2020-10-09	73.5	1
1	2020-09-10	10.8	1
1	2020-08-16	10.8	1
1	2020-07-13	94.5	1
1	2020-06-01	42	1
1	2020-03-02	31.5	1
1	2020-02-25	12	1

Link file .sql : https://drive.google.com/drive/folders/1EmvNGZOB0lP0kj2VgU9hMbrhYQ1Sjp9w?usp=sharing

B. Snowflake Scheme



Beberapa kelebihan dari skema ini :

1) Struktur yang dinormalisasi membuat analisis lebih fleksibel, bahkan bisa berkembang di luar objektif yang ditentukan di awal
2) Data reference integrity terjaga
3) Mudah menambahkan dimension baru tanpa merubah dimension yang sudah ada

Beberapa kekurangan :

1) Performance lebih rendah akibat join table
2) Data hirarki akan menjadi bias jika ditempatkan di dimension table yang berbeda

Use Case 2 : Data Warehouse Design Architecture

Requirements :
Design EDW ditentukan oleh objektif development, dalam hal ini menggunakan goal-driven requirements. Use case yang ingin dikerjakan yaitu Market Expansion, sehingga requirement yang ingin dijawab yaitu :

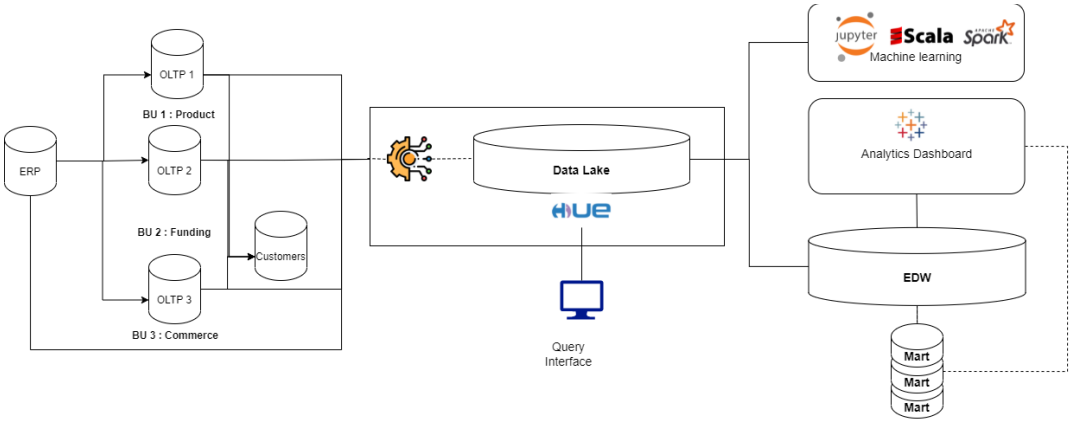
Business Goal
Produk apa yang paling dibutuhkan
Di provinsi mana budidaya memiliki potensi berkembang
Apa saja hasil budidaya per provinsi/ wilayah
Daerah mana yang bisa menjadi target market baru

Dari requirements tersebut, karakter data yang digunakan kurang lebih sebagai berikut :

Data	Description	Owner	Update Frequency
Order	Transaksi harian produk-produk smart feeder, pakan, etc.	B2B, B2C	Daily/hourly
Order details	List produk yang ada di dalam setiap purchase order	B2B, B2C	Daily/hourly
Product	Master produk yang dijual	Product	Quarterly
Customer	Pelanggan yang dimiliki	B2B, B2C	Daily
Campaign	Campaign yang pernah dilakukan	Marketing	Weekly
Master jenis budidaya	Jenis budidaya masyarakat	Product	Yearly
Master Campaign	Master outbound campaign : event, newsletter, dll.	Marketing	Quarterly
Provinsi	Data master provinsi di Indonesia	B2B, B2C	Yearly
Invoice	Invoice-invoice yang diterbitkan berdasarkan term of payment setiap order	Finance	Daily
Lokasi	Kabupaten hingga provinsi	B2B, B2C	Yearly

Untuk detail data requirement bisa dilihat di link berikut :
https://docs.google.com/spreadsheets/d/1NnXZR2W8NlqQ6Udf2_4G4GPgaCVw8RVgIjW2d2SyJMY/edit#gid=0

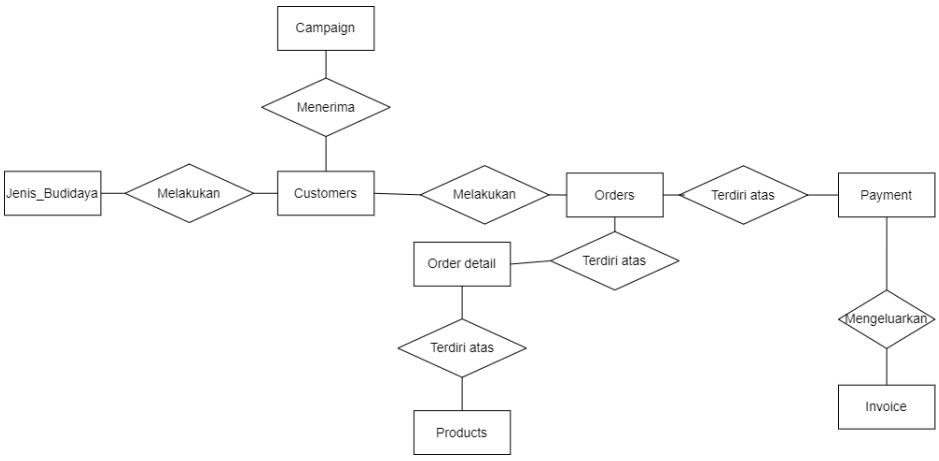
Data pipeline berikut diasumsikan untuk EDW yang diupdate harian, sehingga tidak membutuhkan arsitektur yang melakukan real-time processing.



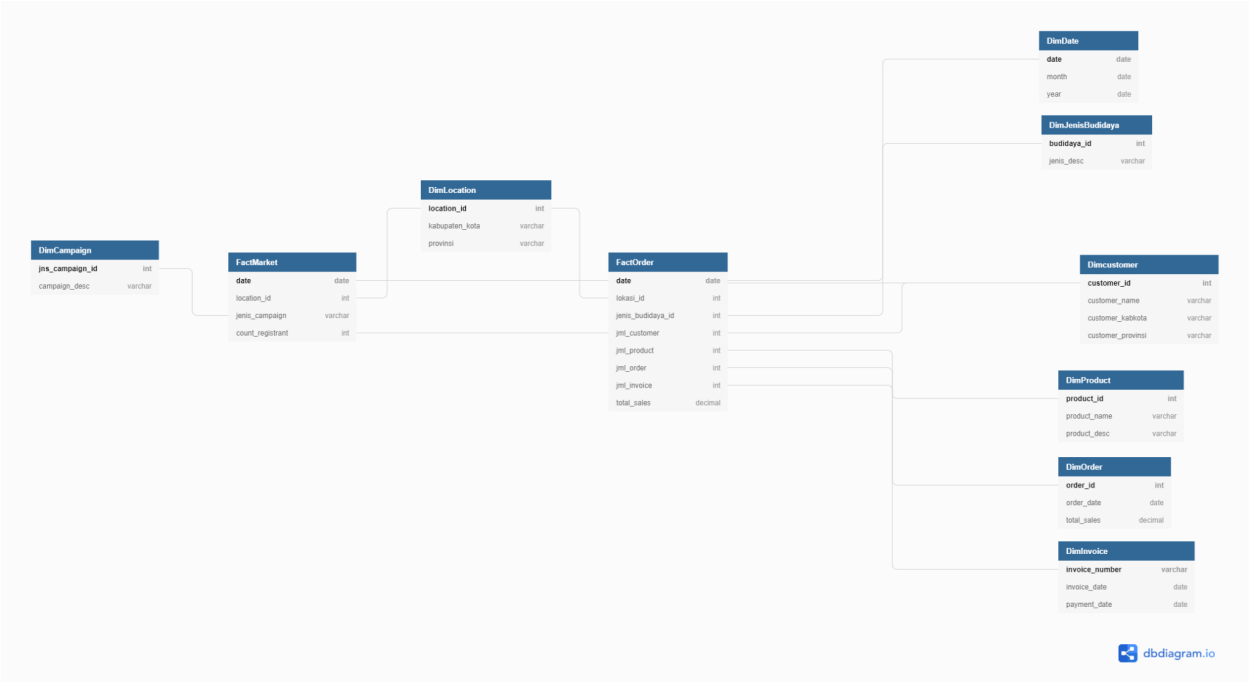
Data dari OLTP setiap business unit,ERP dan customer database di-pull ke dalam Data Lake. Di sini data melalui proses Extract Transform Load, yaitu :

- Extract : membuat “routine” data ekstraksi dari sumbernya, dengan prosedur incremental (hanya perubahan data) atau full snapshot (truncate existing dan extract seluruh data)
- Transform : melakukan data type checking/ assigning, data validation, redundancy and uniqueness checking, integrity validation, timestamp field creation, column mapping, dll.
- Load : load data yang sudah diagregasi berdasarkan objektif yang sudah didefinisikan di awal, ke data warehouse

ERD operational database :



Arsitektur EDW menggunakan galaxy schema :



EDW memiliki 2 fact table dengan alasan user requirement menggunakan 2 data dengan measurement yang sedikit berbeda, yaitu transaksi dan marketing. Selain itu, use case ini masih dalam tahap uji coba, sehingga bentuk partial normalized tersebut, akan memudahkan pengembangan jika dibutuhkan.