Membangun Conceptual Schema Data Pada Industri Asuransi

Disusun Oleh:

Kelompok 4
Data Warehouse RA

Evan Aprianto	121450024
Kiwit Novitasari	121450126
Meira Listyaningrum	122450011
Salwa Farhanatussaidah	122450055
Ibrahim Al-Kahfi	122450100

PROGRAM STUDI SAINS DATA

FAKULTAS SAINS

INSTITUT TEKNOLOGI SUMATERA

2025

Latar Belakang

Data warehouse merupakan sistem penyimpanan data yang dirancang untuk mendukung proses analisis dan pengambilan keputusan. Dalam industri asuransi, data warehouse memainkan peran krusial dalam mengintegrasikan data dari berbagai sumber seperti data polis, klaim, pelanggan, dan pembayaran premi menjadi satu kesatuan yang tersusun secara strategis. Analisis data yang dihasilkan dapat membantu perusahaan asuransi dalam mengidentifikasi risiko, mengevaluasi produk, dan meningkatkan layanan pelanggan.

Pada laporan ini, kami menganalisis desain data warehouse untuk industri asuransi dengan fokus pada tiga tahap utama: desain konseptual, logikal, dan fisikal. Kami juga memberikan rekomendasi tambahan untuk optimasi dan pengembangan lebih lanjut.

1. Desain Konseptual

1.1 Design and implement dimension tables

Tabel dimensi dirancang untuk menyimpan informasi deskriptif yang melengkapi data kuantitatif dalam tabel fakta. Berikut adalah tabel dimensi yang telah dirancang:

Nama tabel	Atribut utama	deskripsi
dim_waktu	ID_Tanggal, Tahun, Kuartal, Bulan, Tanggal	Memungkinkan analisis temporal dengan hierarki Tahun → Kuartal → Bulan → Tanggal.
dim_pelanggan	ID_Pelanggan, Usia, Jenis_Kelamin, Status_Perokok	Berisi atribut demografis dan kebiasaan pelanggan.
dim_wilayah	ID_Wilayah, Nama_Wilayah	Mencakup informasi geografis pelanggan.
dimBMI	ID_BMI, Kategori_BMI, Range_BMI	Khusus untuk kategori indeks massa tubuh.

dimChildren	ID_Children, Jumlah_Anak	Untuk jumlah anak tanggungan.

- **Tabel Dimensi Pelanggan**

  Tabel ini mencatat data penting mengenai setiap pelanggan, seperti umur, jenis kelamin, jumlah anak yang menjadi tanggungan, status merokok, serta indeks massa tubuh (BMI). Informasi ini berguna dalam mengelompokkan pelanggan berdasarkan risiko kesehatan dan kebutuhan asuransi mereka.

Kolom	Tipe Data	Deskripsi
id_pelanggan	INT (PK)	Kunci utama pelanggan
umur	INT	Umur pelanggan
jenis_kelamin	VARCHAR(10)	Kategori Laki-laki / Perempuan
bmi	FLOAT	Indeks massa tubuh (Body Mass Index
jumlah_anak	INT	Jumlah anak tanggungan
status _perokok	VARCHAR(10)	Kategori Perokok / Tidak

- **Tabel Dimensi Waktu**

  Tabel dimensi waktu digunakan untuk merekam kapan suatu klaim terjadi, baik dari sisi tanggal, bulan, maupun tahun. Struktur ini memudahkan analisis tren berdasarkan waktu, seperti peningkatan jumlah klaim bulanan atau fluktuasi tahunan.

Kolom	Tipe Data	Deskripsi
id_waktu	INT(PK)	ID waktu
tanggal	DATE	Tanggal klaim
bulan	INT	Bulan klaim
tahun	INT	Tahun klaim

- **Tabel Dimensi Wilayah**

Tabel ini berisi data wilayah tempat pelanggan berasal, seperti nama kota atau provinsi. Dimensi ini sangat berguna dalam menganalisis distribusi klaim menurut lokasi geografis

Kolom	Tipe Data	Deskripsi
id_wilayah	INT(PK)	ID unik wilayah
nama_wilayah	VARCHAR	Nama wilayah

- **Tabel Dimensi BMI**

Tabel ini berisi informasi tentang kategori indeks massa tubuh (BMI) pelanggan. Dimensi ini membantu dalam menganalisis klaim asuransi berdasarkan status kesehatan atau berat badan pelanggan, yang dapat berpengaruh terhadap besarnya biaya klaim.

Kolom	Tipe Data	Deskripsi
id_bmi	INT	Kunci utama untuk setiap kategori BMI
kategori_bmi	VARCHAR(20)	Kategori BMI seperti

		"Underweight", "Normal", "Overweight", atau "Obese"
range_bmi	VARCHAR(20)	Rentang nilai BMI untuk kategori tertentu, misalnya "18.5–24.9"

- **Tabel Dimensi Children**

  Tabel ini menyimpan informasi tentang jumlah tanggungan atau anak yang dimiliki oleh pelanggan. Dimensi ini penting untuk menganalisis pengaruh jumlah tanggungan terhadap besarnya biaya klaim asuransi.

Kolom	Tipe Data	Deskripsi
id_children	INT	Kunci utama untuk entitas jumlah anak
jumlah_anak	INT	Jumlah anak tanggungan yang dimiliki oleh pelanggaan

Query :

```
-- Tabel dim_pelanggan
CREATE TABLE dim_pelanggan (
    id_pelanggan INT PRIMARY KEY,
    umur INT,
    jenis_kelamin VARCHAR(10),
    bmi FLOAT,
    jumlah_anak INT,
```

```
    status_perokok VARCHAR(10)
);

-- Tabel dim_waktu
CREATE TABLE dim_waktu (
    id_waktu INT PRIMARY KEY,
    tanggal DATE,
    bulan INT,
    tahun INT
);

-- Tabel dim_wilayah
CREATE TABLE dim_wilayah (
    id_wilayah INT PRIMARY KEY,
    nama_wilayah VARCHAR(50)
);

-- Tabel dimBMI
CREATE TABLE dim_bmi (
    id_bmi INT PRIMARY KEY,
    kategori_bmi VARCHAR(20),
    range_bmi VARCHAR(20)
);

-- Tabel dim_children
CREATE TABLE dim_children (
    id_children INT PRIMARY KEY,
    jumlah_anak INT
);
```

1.2 Design and implement fact tables

Tabel fakta ini menjadi pusat analisis dalam data warehouse karena menyimpan data transaksi klaim yang bersifat kuantitatif, seperti total biaya klaim. Setiap entri di dalam tabel ini mewakili satu klaim yang dilakukan pelanggan, dan terhubung langsung ke dimensi waktu, pelanggan, serta wilayah. Berikut desain tabel fakta dan implementasinya:

Kolom	Tipe Data	Deskripsi
id_fakta	INT (PK)	Kunci utama baris fakta
id_pelanggan	INT (FK)	ID pelanggan referensi ke dim_pelanggan
id_waktu	INT (FK)	ID waktu referensi ke dim_waktu
id_wilayah	INT (FK)	ID wilayah referensi ke dim_wilayah
biaya_klaim	FLOAT	Biaya medis individual

Query :

```
-- Tabel fakta
CREATE TABLE fact_klaim (
    id_fakta INT PRIMARY KEY,
    id_pelanggan INT,
    id_waktu INT,
    id_wilayah INT,
    biaya_klaim FLOAT,
    FOREIGN KEY (id_pelanggan) REFERENCES dim_pelanggan(id_pelanggan),
    FOREIGN KEY (id_waktu) REFERENCES dim_waktu(id_waktu),
    FOREIGN KEY (id_wilayah) REFERENCES dim_wilayah(id_wilayah)
```

```
);
```

1.3 Skema yang Digunakan

Kelompok kami memilih Star Schema dengan alasan sebagai berikut:

- Sederhana dan Mudah Dipahami: Satu tabel fakta di pusat yang terhubung langsung ke beberapa tabel dimensi.
- Optimal untuk Query Analitik: Memudahkan analisis agregat seperti total klaim per wilayah atau tren klaim per bulan.
- Kemudahan Maintenance: Struktur yang jelas memudahkan pemeliharaan dan pengembangan.

2. Desain Logikal & Fisikal

2.1 Design and implement indexes

Untuk optimasi performa query, berikut desain indeks yang kami gunakan:

Jenis indeks	Contoh implementasi	Tujuan
Primary key index	Pada semua kolom ID di setiap tabel dimensi.	Memastikan keunikan dan akses cepat.
Foreign key index	Pada kolom kunci asing di tabel fakta.	Mempercepat operasi JOIN.
Bitmap index	Pada kolom seperti status_perokok, jenis_kelamin	Efisien untuk kolom dengan kardinalitas rendah.
Composite index	Kombinasi id_pelanggan dan id_tanggal	Optimal untuk query yang sering menggunakan kombinasi kolom.

Query:

```
-- Primary key sudah otomatis index
-- Foreign Key Index
CREATE INDEX idx_fk_pelanggan ON fact_klaim(id_pelanggan);
CREATE INDEX idx_fk_waktu ON fact_klaim(id_waktu);
CREATE INDEX idx_fk_wilayah ON fact_klaim(id_wilayah);

-- Di PostgreSQL bisa gunakan B-tree karena tidak mendukung bitmap secara eksplisit
CREATE INDEX idx_perokok ON dim_pelanggan(status_perokok);
CREATE INDEX idx_gender ON dim_pelanggan(jenis_kelamin);

-- Composite Index
CREATE INDEX idx_composite_pelanggan_waktu ON fact_klaim(id_pelanggan, id_waktu);
```

2.2 Desain storage

Staging Layer

Staging layer merupakan lapisan awal tempat data mentah dari sumber eksternal dimuat. Dalam proyek ini, data dikumpulkan dari file insurance_with_random_dates.csv dan dimasukkan ke dalam tabel staging di SQL Server. Tabel bersifat flat dan disimpan menggunakan row-based storage. Kolom-kolom bersifat fleksibel untuk menerima berbagai tipe data dari file CSV.

Data Warehouse Layer

Lapisan ini merupakan inti dari sistem gudang data. Setelah data diproses dan dibersihkan dari staging, data dipindahkan ke tabel fakta dan tabel dimensi dengan struktur yang telah dinormalisasi mengikuti skema bintang (star schema). Tabel disimpan menggunakan default row-store SQL Server.

Presentation Layer

Lapisan ini digunakan untuk menampilkan hasil analisis dalam bentuk visual interaktif menggunakan Power BI Desktop. Laporan visualisasi disimpan dalam format .pbix tujuannya untuk Menyajikan laporan dan dashboard interaktif berbasis data dari data warehouse

2.3 Design and implement partitioned tables and views

Partisi tabel dan view digunakan untuk meningkatkan efisiensi dalam pengelolaan data historis dan mempercepat proses query analitik, desain fisikal data warehouse ini menerapkan strategi partisi pada tabel utama (tabel fakta dan dimensi). Partisi dilakukan berdasarkan atribut yang sering digunakan dalam filter atau agregasi data, seperti tahun klaim dan wilayah pelanggan. Selain itu, untuk mempercepat akses data yang sering dianalisis, beberapa materialized view juga dirancang agar hasil perhitungan agregat tertentu dapat disimpan secara permanen dan diperbarui secara berkala. Rincian desain tabel terpartisi dan view yang digunakan disajikan pada tabel berikut:

Desain Tabel Terpartisi

Nama Tabel	Jenis Partisi	Kolom Partisi	Kriteria Partisi	Tujuan
fact_klaim	Range Partitioning	tahun	P2022 (<2023), P2023(<2024), p2024 (<2025), pmax	Memisahkan data historis tahunan untuk efisiensi query
dim_wilayah	List Partitioning	nama_wilayah	Jakarta, Bandung, Surabaya, Medan, Padang, dll.	Optimasi analisis geografis berdasarkan wilayah tertentu

Tabel Desain Materialized View

Untuk mendukung performa analisis dan visualisasi data secara cepat, digunakan materialized view, yaitu objek database yang menyimpan hasil dari query agregat tertentu. Materialized view sangat berguna untuk laporan yang bersifat rutin dan sering diakses, seperti rekap klaim per bulan atau berdasarkan kelompok pelanggan tertentu. Dengan menyimpan hasil agregasi secara terpisah, beban komputasi pada tabel fakta utama dapat dikurangi, sekaligus mempercepat waktu respon query dalam dashboard atau laporan berkala. Berikut tabel desain materialized view yang digunakan:

Nama View	Agregasi	Dimensi yang Digunakan	Tujuan
mv_total_klaim_wilayah_bulanan	Total & Rata-rata klaim	dim_waktu, dim_wilayah	Memonitor klaim per wilayah dan waktu secara efisien
mv_klaim_per_status_perokok	Jumlah & Rata-rata klaim	dim_pelanggan (status_perokok)	Menganalisis dampak kebiasaan merokok terhadap biaya klaim
mv_klaim_per_kategori_bmi	Jumlah & Rata-rata klaim per BMI	dim_bmi	Menganalisis klaim berdasarkan status kesehatan pelanggan

Query:

```
-- Tabel partisi berdasarkan kolom tahun
CREATE TABLE fact_klaim (
    id_fakta INT,
    id_pelanggan INT,
    id_waktu INT,
    id_wilayah INT,
    tahun INT NOT NULL,
    biaya_klaim FLOAT
)
```

```
PARTITION BY RANGE (tahun) (
    PARTITION p2022 VALUES LESS THAN (2023),
    PARTITION p2023 VALUES LESS THAN (2024),
    PARTITION p2024 VALUES LESS THAN (2025),
    PARTITION pmax VALUES LESS THAN MAXVALUE
);
```

3. Tabel Agregat

Implementasi tabel agregat digunakan untuk menyimpan metrik rata-rata klaim per wilayah per bulan, yang dapat mempercepat query analitik dan mengurangi beban komputasi pada tabel fakta utama.

3.1 Struktur Tabel Agregat

Untuk mendukung efidiensi dalam pemrosesan dat analitik, tabel agregat dirancang untuk menyimpan ribgkasan data klaim berdasarkan wilayah dan waktu. Tabel ini berisi merik seperti total klain dan rata-rata biaya klaim per bulan, yang berh=guna untuk mempercepat proses pelaporan rutin dan mengurangi beban pemrosesan pada tabel fakta utama. Struktur lengkap tabel agregat disajikan pada tabel berikut.

Nama Kolom	Tipe Data	Deskripsi	Contoh Nilai
ID_Agregat	INT	Primary key,out-increment.	1,2,3
ID_Wilayah	VARCHAR(10)	Foreign key ke tabel Dim_Wilayah	“W001”, “W002”
Tahun	SMALLINT	Tahun transaksi.	2023,2024
Bulan	TINYINT	Bulan transaksi (1-12)	1(jan), 12 (Des)
Jumalh_Klaim	INT	Total klaim dalam periode tersebut	150,320

Rata_Rata_Klaim	DECIMAL(10,2)	Rata-rata klaim per transaksi	1250.50, 980.75
Last_Updated	TIMESTAMP	Waktu terakhir data diperbarui	2024-05-20 14:30:00

Query:

```
CREATE TABLE agregat_klaim_wilayah_bulanan (
    id_agregat INT AUTO_INCREMENT PRIMARY KEY,
    id_wilayah VARCHAR(10),
    tahun SMALLINT,
    bulan TINYINT,
    jumlah_klaim INT,
    rata_rata_klaim DECIMAL(10,2),
    last_updated TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
);
```

3.2 Proses Pengisian Data (ETL)

Data dapat diisi melalui stored procedure atau job ETL yang dijalankan secara berkala:

```
INSERT INTO agregat_klaim_wilayah_bulanan (
    id_wilayah, tahun, bulan, jumlah_klaim, rata_rata_klaim
)
SELECT
    fk.id_wilayah,
    dw.tahun,
    dw.bulan,
    COUNT(fk.biaya_klaim) AS jumlah_klaim,
    AVG(fk.biaya_klaim) AS rata_rata_klaim
```

```
FROM fact_klaim fk
JOIN dim_waktu dw ON fk.id_waktu = dw.id_waktu
GROUP BY fk.id_wilayah, dw.tahun, dw.bulan;
```

4. Kesimpulan

Desain data warehouse untuk industri asuransi menggunakan star schema untuk mengidentifikasi tabel fakta dan dimensi yang relevan, serta menyesuaikan dengan kebutuhan bisnis dari berbagai pemangku kepentingan. Untuk pengembangan selanjutnya, performa dapat ditingkatkan melalui penerapan teknik indexing agar query berjalan lebih cepat, penggunaan partitioning untuk pengelolaan data historis yang lebih efisien, serta strategi penyimpanan yang optimal seperti columnar storage dan kompresi data. Pendekatan yang terstruktur ini memungkinkan perusahaan asuransi untuk mengoptimalkan penggunaan data warehouse dalam analisis risiko, evaluasi produk, dan peningkatan kualitas layanan pelanggan secara lebih terukur dan efektif.