

Click one below to reach me 🗨️ 😊



[ilham-akbar-3301abc](#)



ilhamakr3301@gmail.com



[+62 895-2258-9852](tel:+6289522589852)

Data Preparation & Management with SQL (PostgreSQL)

Studi Kasus: *Global Superstore Dataset*

By: Ilham Akbar



Pendahuluan



Apa itu data?

Data Adalah kumpulan fakta atau informasi mentah (angka, teks, gambar) yang dapat diolah untuk mendukung keputusan dan ilmu pengetahuan



Apa itu SQL?

Bahasa standar untuk mengakses, mengelola, dan mengolah data pada basis data relasional, mulai dari mengambil informasi, memperbarui, hingga membersihkan data.



Apa itu Data Management?

Praktik mengatur, menyimpan, dan menjaga kualitas data dalam basis data agar dapat digunakan secara konsisten dan andal.

Apa itu data Preparation?

Proses membersihkan, menstandarkan, dan mengorganisir data mentah agar siap digunakan dalam proses analisis.

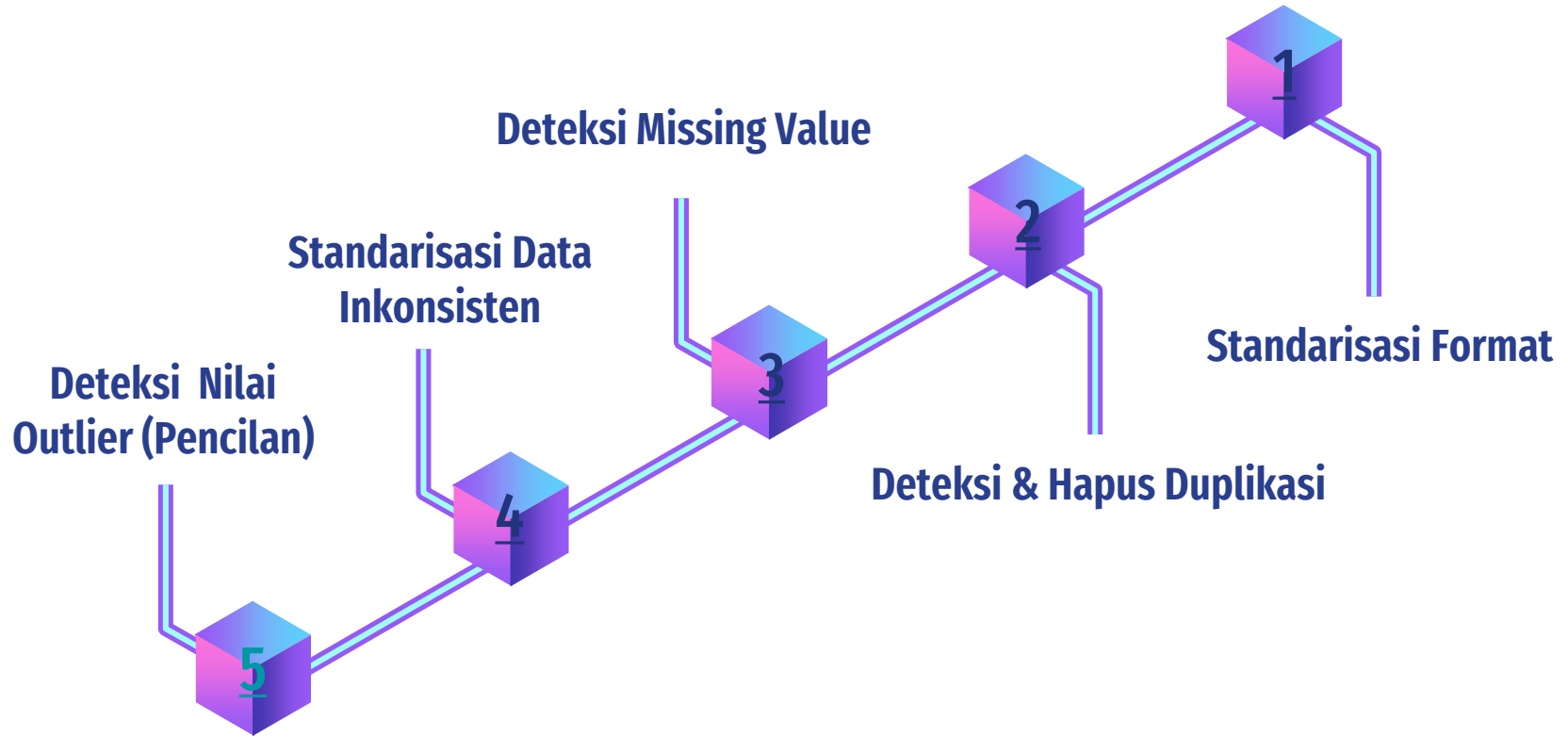




Mengapa Perlu Melakukan Preparation dan Management Data?

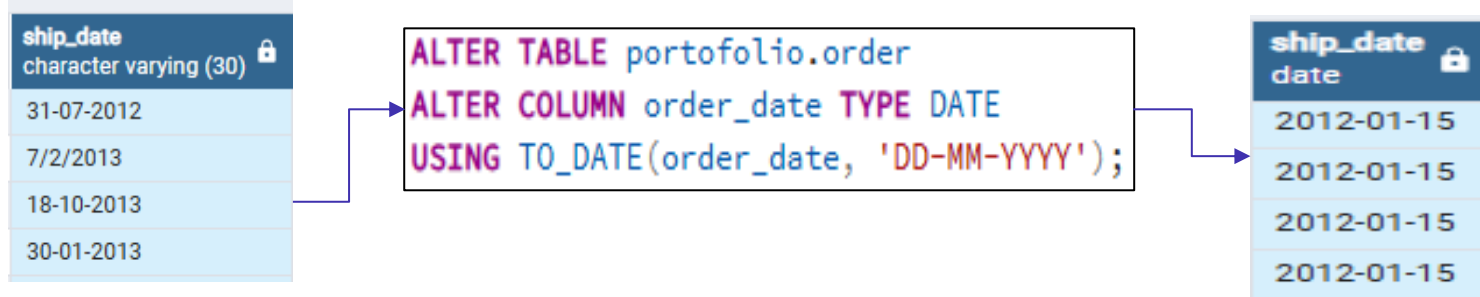
1. Karena kualitas data akan langsung menentukan kualitas analisis yang dihasilkan. Data yang masih kotor, berisi duplikasi, nilai kosong, atau inkonsistensi, berpotensi menghasilkan insight yang salah dan berisiko menyesatkan pengambilan keputusan.
2. Prinsip analisis data selalu berpegang pada konsep ***garbage in, garbage out***: tanpa data yang bersih dan rapi, metode analisis secanggih apapun tidak akan memberi hasil yang akurat.
3. Dalam realita karir seorang data analyst, sebagian besar waktu kerja justru dihabiskan untuk tahap persiapan ini, karena hanya dengan fondasi data yang terkelola baik, analisis bisa berjalan efektif dan memberi nilai nyata bagi bisnis.

Contoh Data Preparation & Management



1. Standarisasi Format

Standarisasi format adalah proses menyamakan bentuk penyajian data (misalnya tanggal, teks, atau angka) agar konsisten dan mudah diproses dalam analisis.



Permasalahan

1. Tanggal tersimpan dalam format teks VARCHAR(30) dengan pola berbeda: 31-07-2012 dan 7/2/2013.
2. PostgreSQL tidak dapat melakukan operasi tanggal (selisih, ekstraksi tahun/bulan) jika kolom bertipe teks.
3. Hal ini menyulitkan analisis tren, perhitungan umur data, dan segmentasi berbasis waktu.

Query

1. **ALTER TABLE:** mengubah struktur tabel.
2. **ALTER COLUMN:** memilih kolom yang diubah (order_date).
3. **TYPE DATE:** Mengatur tipe data kolom menjadi DATE agar dapat dilakukan perhitungan waktu dan query berbasis tanggal.
4. **USING TO_DATE (order_date, 'DD-MM-YYYY'):** Menjelaskan cara mengubah data lama (teks) ke tipe baru (DATE) menggunakan pola hari-bulan-tahun. Tanpa bagian ini, PostgreSQL tidak tahu bagaimana mengonversi isi kolom..

2. Deteksi & Hapus Duplikasi

Deteksi dan penghapusan duplikasi adalah proses mengidentifikasi baris data yang tercatat lebih dari sekali dengan kunci identitas yang sama (misalnya `row_id`), lalu menghapus duplikatnya agar data bersih, konsisten, dan tidak menyesatkan analisis.

```
DELETE FROM portofolio.orders a
USING portofolio.orders b
WHERE a."row_id" = b."row_id"
AND a.ctid < b.ctid;
```

Data Output	Messages	Notifications
DELETE 0		
Query returned successfully in 664 msec.		

Tidak ditemukan baris duplikat
berdasarkan acuan `row_id`

Permasalahan

1. Data transaksi ganda sering muncul akibat kesalahan input manual, bug pada sistem, atau proses ETL (Extract-Transform-Load). Hal ini menyebabkan satu transaksi yang sama tercatat lebih dari sekali.
2. Dampak jika data duplikasi (ganda) tidak dibersihkan, maka berbagai praktik agregasi dan analisis bisnis bisa menyesatkan.
3. Kolom `row_id` dirancang sebagai identitas unik untuk setiap baris (primary key). Namun adanya nilai ganda menunjukkan ketidaksesuaian integritas dan relasi antar dataset.


Query

1. **DELETE FROM portofolio.order a:** Perintah untuk menghapus baris data dari tabel order, dengan alias a agar lebih mudah dibedakan saat dibandingkan dengan alias lain.
2. **USING portofolio.order b:** Menggunakan tabel yang sama (self-join) dengan alias b untuk menemukan pasangan baris yang duplikat.
3. **WHERE a.row_id = b.row_id:** Acuan utama untuk mendeteksi baris dengan `row_id` yang identik, menandai duplikasi pada kolom tersebut.
4. **AND a.ctid < b.ctid:** PostgreSQL menyimpan baris dengan identitas internal bernama **ctid**. Kondisi ini memastikan baris (yang lebih kecil `ctid`-nya) yang dihapus, sehingga menyisakan satu baris unik.

3. Deteksi Missing Value

Deteksi missing value adalah proses mengidentifikasi data yang kosong atau hilang pada kolom penting. Data yang tidak lengkap dapat mengganggu akurasi analisis dan menyebabkan bias dalam pengambilan keputusan.

```
SELECT
SUM(CASE WHEN profit IS NULL THEN 1 ELSE 0 END) AS p0
, SUM(CASE WHEN sales IS NULL THEN 1 ELSE 0 END) AS s0
, SUM(CASE WHEN customer_name IS NULL THEN 1 ELSE 0 END) AS cn0
FROM portofolio.order o
JOIN portofolio.customer c ON o.row_id = c.row_id;
```



	p0 bigint	s0 bigint	cn0 bigint
1	0	0	0

Tidak ditemukan Missing Value (nilai kosong) pada kolom krusial

Permasalahan

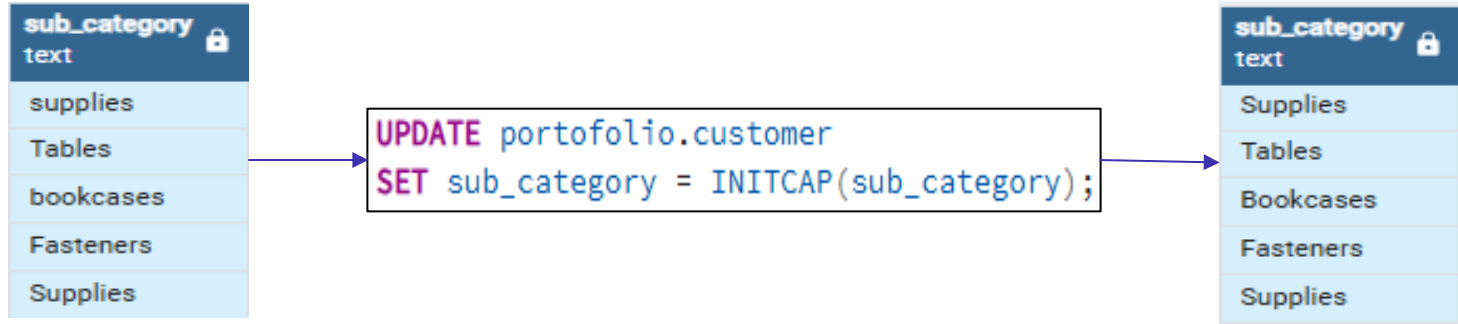
1. Nilai kosong pada kolom krusial seperti profit, sales, atau customer_name menyebabkan informasi transaksi tidak utuh.
2. Analisis penjualan bisa salah karena baris dengan nilai kosong tetap ikut dihitung.
3. Jika tidak ditangani, maka hasil agregasi (total, rata-rata, tren) bisa menyesatkan karena sebagian data tidak terisi.

Query

1. **SUM(CASE WHEN (profit/sales/customer_name) IS NULL ... END):** Membuat logika pengecekan bahwa jika profit, sales, dan customer_name kosong, maka hitung 1, jika tidak, maka hitung 0. Hasil per baris dijumlahkan dengan SUM, sehingga kita mendapatkan total baris yang tidak memiliki salah satu dari ketiga nilai tersebut
2. **JOIN portofolio.customer:** Menggabungkan tabel order (yang berisi data transaksi) dengan tabel customer (yang berisi informasi pelanggan) melalui row_id. Tujuannya agar pengecekan nilai kosong bisa dilakukan pada dua sumber data penting sekaligus, sehingga lebih komprehensif.

4. Standarisasi Data Inkonsisten

Standarisasi data inkonsisten adalah proses menyamakan penulisan atau format data agar seragam. Tujuannya untuk menghindari perbedaan representasi yang sebenarnya merujuk pada entitas yang sama.



Permasalahan

1. Nilai pada kolom `sub_category` ditulis dengan format kapital yang berbeda (misalnya: `Supplies` dan `supplies`).
2. Perbedaan penulisan menyebabkan data dianggap sebagai kategori yang berbeda padahal maksudnya sama.
3. Hal ini membuat hasil agregasi, perhitungan jumlah, atau analisis kategori menjadi tidak akurat.


Query

1. **UPDATE portofolio.customer:** Memperbarui data pada tabel `customer`.
2. **SET sub_category:** Menentukan kolom yang nilainya akan diperbarui.
3. **INITCAP(sub_category):** Fungsi PostgreSQL untuk mengubah teks menjadi format huruf besar di awal kata dan huruf kecil di sisa kata. Hasilnya semua nilai pada `sub_category` konsisten dengan gaya penulisan yang sama, sehingga mudah dipakai dalam analisis.

5. Standarisasi Format

Outlier adalah nilai ekstrem yang jauh berbeda dari pola mayoritas data. Identifikasi outlier penting untuk menemukan kesalahan input, anomali transaksi, atau kasus khusus yang dapat memengaruhi hasil analisis.

```
SELECT row_id, discount, profit
FROM portofolio.order
WHERE discount >= 1 OR profit < -10000;
```



row_id	discount	profit
integer	numeric (15,2)	numeric (15,2)

Tidak ditemukan nilai Outlier (pencilan) pada kolom discount dan profit

Permasalahan

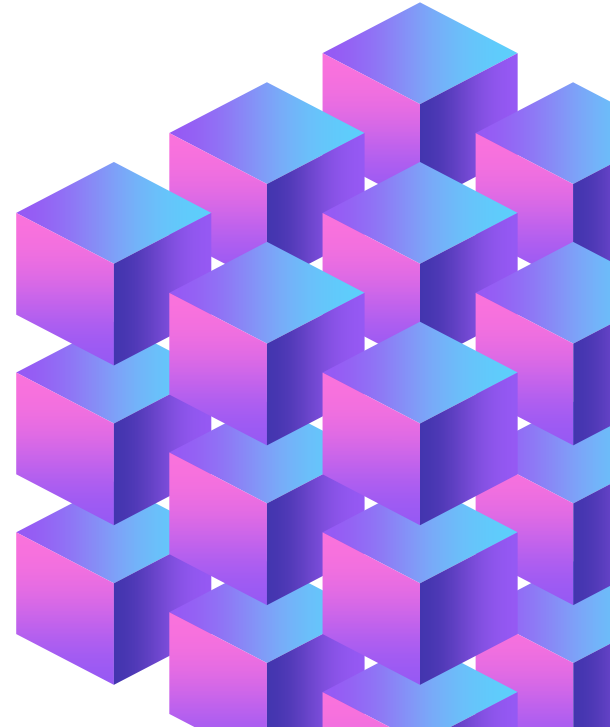
1. Kolom discount kadang berisi nilai ≥ 1 (100%), padahal logisnya diskon berada pada rentang 0–0,9.
2. Kolom profit berpotensi bernilai sangat negatif (misalnya < -10.000), kemungkinan akibat kesalahan pencatatan atau kasus transaksi tidak wajar.
3. Nilai ekstrem ini bisa mendistorsi hasil analisis rata-rata, total profit, maupun model prediksi bisnis.

Query

1. **SELECT row_id, discount, profit:** Menampilkan kolom penting untuk menelusuri baris yang bermasalah.
2. **FROM portofolio.order:** Mengambil data dari tabel transaksi order.
3. **WHERE discount ≥ 1 OR profit < -10000 :** Menemukan baris dengan diskon $\geq 100\%$ ataupun kerugian ekstrim, yang mengindikasikan data yang perlu dicek ulang validitasnya.

Kesimpulan

1. **Data preparation & management adalah fondasi analisis data.**
2. **Data kotor menghasilkan insight yang salah dan berisiko menyesatkan keputusan bisnis.**
3. **SQL berperan penting dalam membersihkan, menstandarkan, dan menjaga integritas data.**
4. **Tahap ini menyita porsi waktu terbesar seorang data analyst di dunia kerja nyata.**
5. **Investasi pada kualitas data mempercepat analisis dan meningkatkan keandalan hasil.**



Thanx!

for your attention

I am fully open to suggestions, feedback, and collaboration.
Reach me through the contacts below 🙌 😊



ilham-akbar-3301abc



+62 895-2258-9852



ilhamakr3301@gmail.com