**LAPORAN PRAKTIKUM**

**BASIS DATA RD**

**MODUL 7**

**Oleh :**

**Muhammad Fadhil Zurani (122140146)**



**Program Studi Teknik Informatika**

**Fakultas Teknologi Industri**

**Institut Teknologi Sumatera**

**2024**

# **Daftar Isi**

[**Daftar Isi** 2](#_Toc165490552)

[**1.** **Dasar Teori** 3](#_Toc165490553)

[**2.** **Ulasan** 4](#_Toc165490554)

[**3.** **Hasil dan Jawaban** 5](#_Toc165490555)

[**4.** **Kesimpulan dan Saran** 9](#_Toc165490556)

# 

# **Dasar Teori**

Teori dasar agregasi dalam MySQL melibatkan penggunaan fungsi agregasi seperti SUM, AVG, COUNT, MIN, dan MAX untuk melakukan operasi perhitungan pada data yang ada dalam tabel. Fungsi-fungsi ini digunakan untuk mengumpulkan informasi statistik tentang data yang diproses. Berikut adalah penjelasan lebih lanjut tentang teori dasar agregasi dalam MySQL:

1. Fungsi Agregasi

Fungsi-fungsi agregasi memungkinkan kita untuk melakukan operasi perhitungan seperti menjumlahkan nilai-nilai dalam suatu kolom, menghitung rata-rata, mengambil nilai maksimum atau minimum, dan menghitung jumlah baris dalam suatu kumpulan data.

1. Penggunaan dalam SELECT

Fungsi agregasi dapat digunakan dalam klausa SELECT untuk mengambil nilai-nilai agregat dari suatu kumpulan data. Misalnya, `SELECT SUM(total\_harga) FROM penjualan` akan menghitung total harga dari semua transaksi penjualan.

1. Klausa GROUP BY

Klausa GROUP BY digunakan bersamaan dengan fungsi agregasi untuk mengelompokkan data berdasarkan nilai tertentu, seperti `GROUP BY kategori\_produk` untuk mengelompokkan penjualan berdasarkan kategori produk.

1. Hanya dalam SELECT

Fungsi-fungsi agregasi hanya dapat digunakan di dalam klausa SELECT, kecuali COUNT, yang juga dapat digunakan dalam klausa WHERE atau HAVING untuk menghitung jumlah baris yang memenuhi suatu kondisi.

1. Penanganan Nilai NULL

Fungsi agregasi secara default mengabaikan nilai NULL dalam perhitungan. Namun, kita dapat menggunakan IFNULL atau COALESCE untuk mengatasi nilai NULL jika diperlukan.

1. Klausa HAVING

Klausa HAVING digunakan bersamaan dengan GROUP BY untuk memberikan kondisi filter terhadap hasil agregasi. Misalnya, `HAVING COUNT(\*) > 10` akan memfilter grup yang memiliki lebih dari 10 baris.

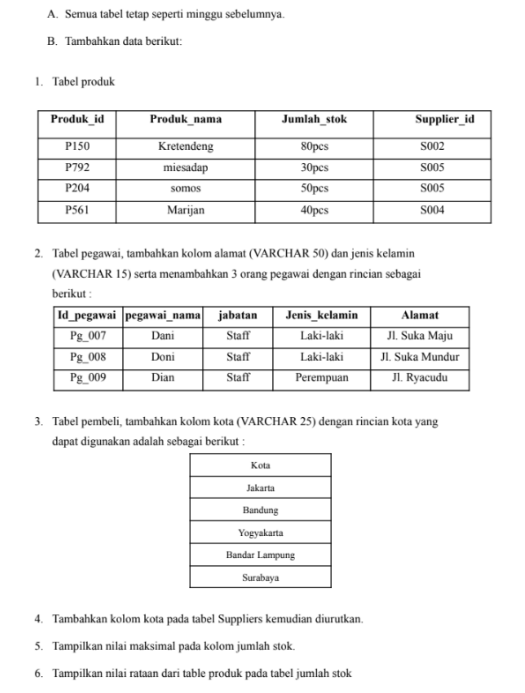
1. Urutan Operasi

Urutan operasi dalam klausa SELECT adalah FROM, WHERE, GROUP BY, HAVING, SELECT, ORDER BY. Ini berarti fungsi agregasi diterapkan setelah klausa WHERE dan sebelum klausa ORDER BY.

1. Penggunaan Aliases

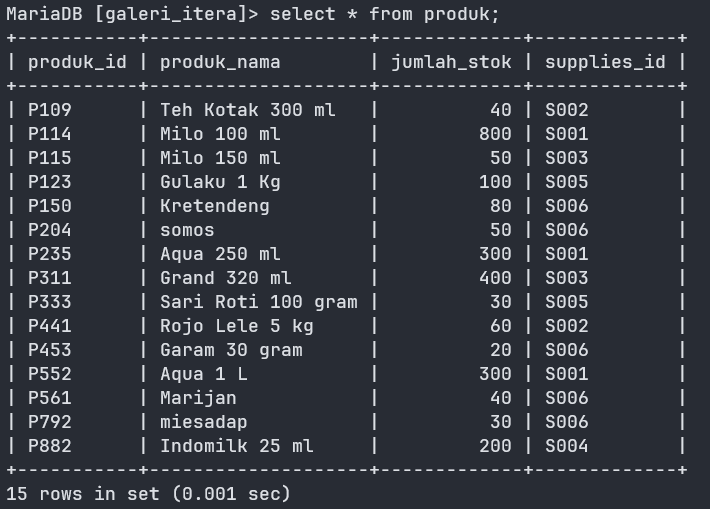
Kita dapat menggunakan aliases untuk memberikan nama yang lebih deskriptif pada hasil fungsi agregasi, misalnya, `SELECT SUM(total\_harga) AS total\_penjualan FROM penjualan`..

# **Ulasan**



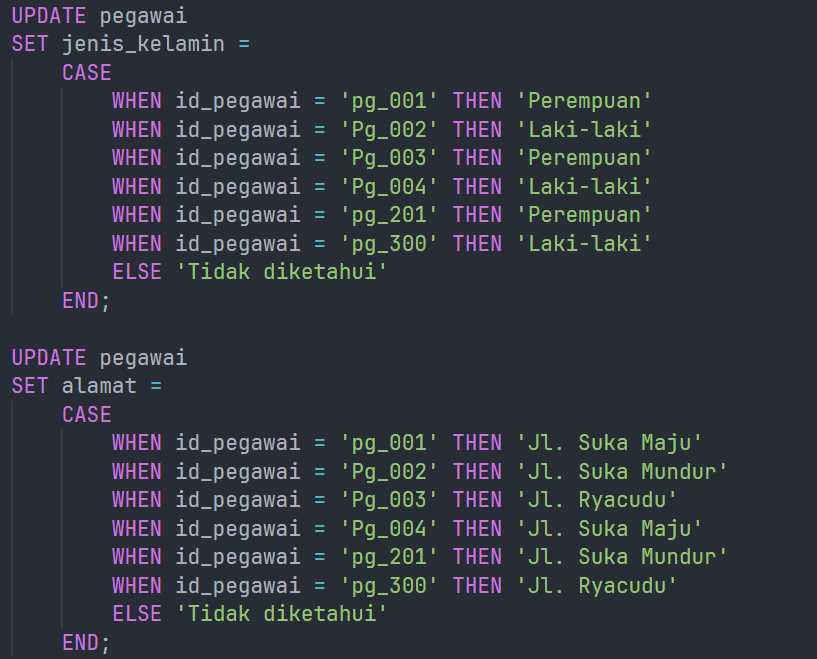
# **Hasil dan Jawaban**

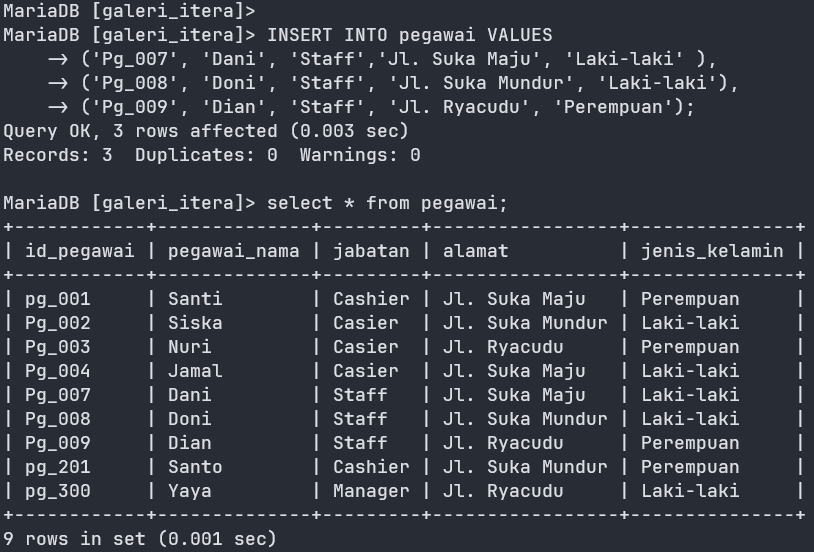
1. Soal 1 Screenshoot hasil dan jawaban dari pengujian



Menambah data yang ada di soal ke tabel produk

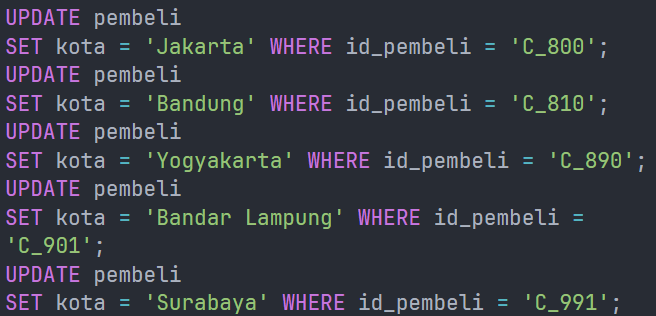
1. Soal 2 Screenshoot hasil dan jawaban dari pengujian

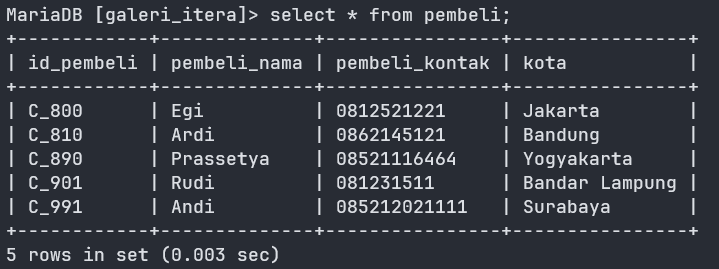




Menambahkan kolom Alamat dan jenis kelamin di tabel pegawai, kita set data lain juga dengan fungsi update

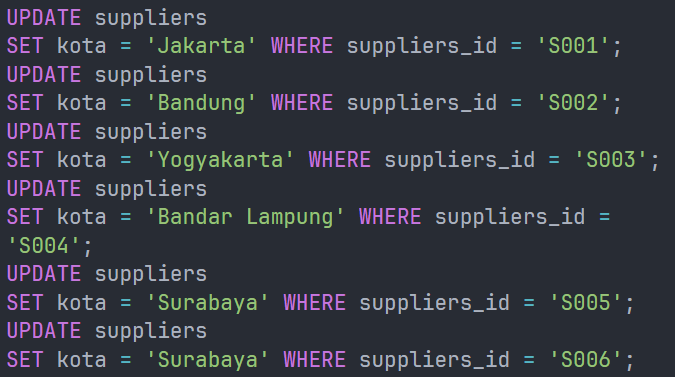
1. Soal 3 Screenshoot hasil dan jawaban dari pengujian

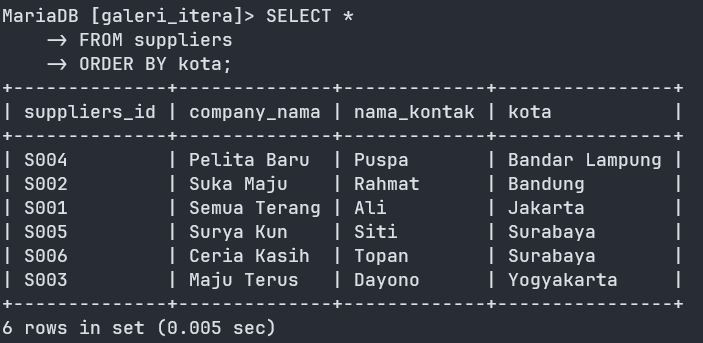




Menambahkan data kota di tabel pembeli

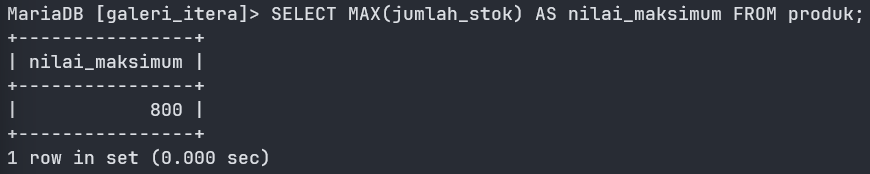
1. Soal 4 Screenshoot hasil dan jawaban dari pengujian





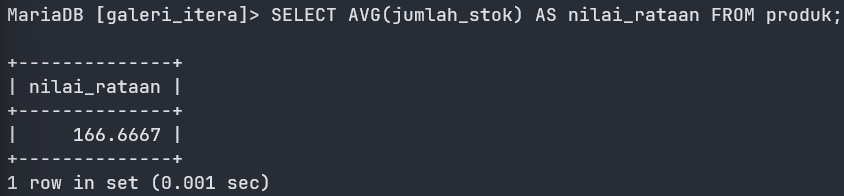
Mengurutkan kota yang ada pada suppliers

1. Soal 5 Screenshoot hasil dan jawaban dari pengujian



Menampilkan nilai maksimal pada kolom jumlah stok

1. Soal 6 Screenshoot hasil dan jawaban dari pengujian



Menampilkan nilai rata-rata pada tabel produk kolom jumlah stok

# **Kesimpulan dan Saran**

Setelah menjalani praktikum mengenai agregasi di MySQL, saya mendapatkan pemahaman yang kuat tentang bagaimana melakukan perhitungan statistik dan menganalisis data dengan efisien. Penggunaan fungsi agregasi seperti SUM, AVG, COUNT, dan lainnya sangat membantu dalam menghitung total nilai, rata-rata, atau mengelompokkan data berdasarkan kriteria tertentu. Selain itu, penggunaan klausa GROUP BY dan HAVING memberikan fleksibilitas yang besar dalam mengatur hasil agregasi berdasarkan kelompok data yang berbeda, sehingga memungkinkan analisis yang lebih mendalam.

Namun, saya juga memperhatikan pentingnya optimasi kinerja saat menggunakan fungsi agregasi, terutama dalam database yang besar. Pemilihan indeks yang tepat, desain query yang efisien, dan menghindari penggunaan subquery yang berlebihan sangat penting untuk meningkatkan kinerja query secara keseluruhan. Praktikum ini tidak hanya meningkatkan pemahaman saya tentang agregasi di MySQL, tetapi juga memberikan wawasan penting tentang desain query yang efisien untuk mengoptimalkan analisis data.