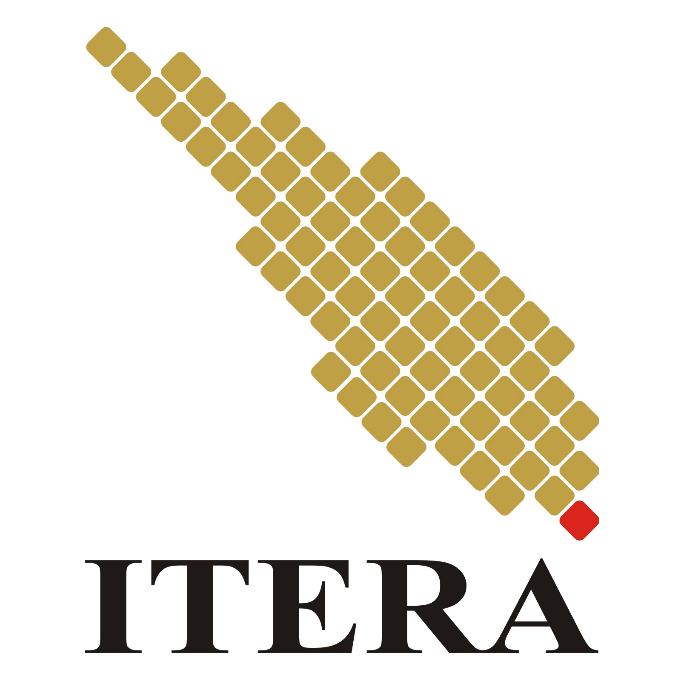
**LAPORAN PRAKTIKUM BASIS DATA RC**

**Muhammad Yusuf  
122140193**

**Latihan**

****

**Teori Dasar**

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.

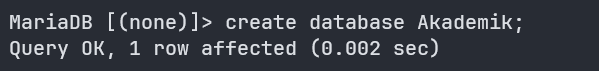
1. Penggunaan Aliases

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

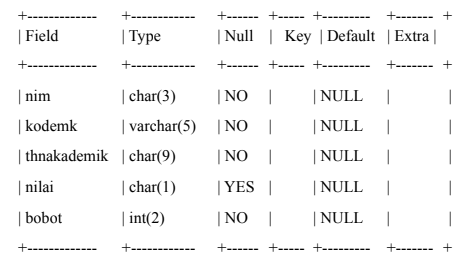
Dengan memahami teori dasar agregasi dalam MySQL, kita dapat membuat query yang kompleks untuk menganalisis data secara efektif dan mendapatkan informasi yang dibutuhkan dari database. Namun, perlu diingat untuk memperhatikan kinerja query agar tetap efisien, terutama ketika menggabungkan fungsi agregasi dengan subquery atau operasi lainnya.

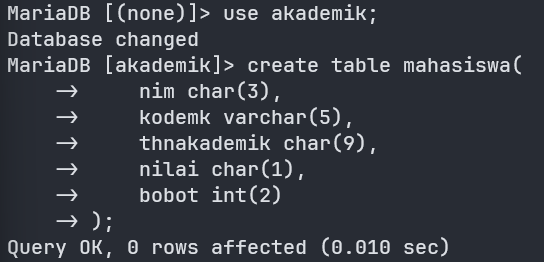
**Pembahasan**

1. Buatlah database dengan nama ”Akademik”

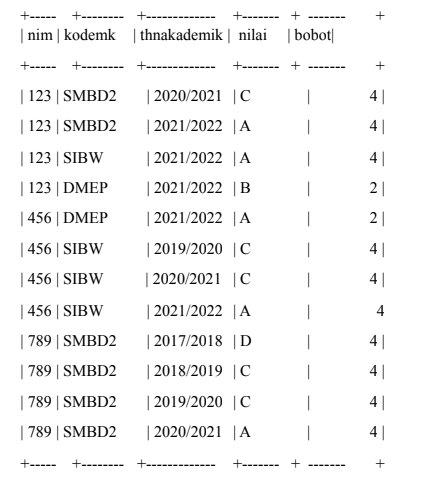


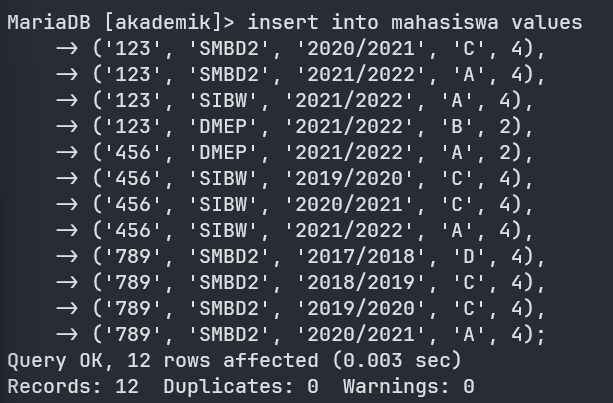
1. Buatlah tabel dengan nama ”mahasiswa” dengan struktur tabel sebagai berikut



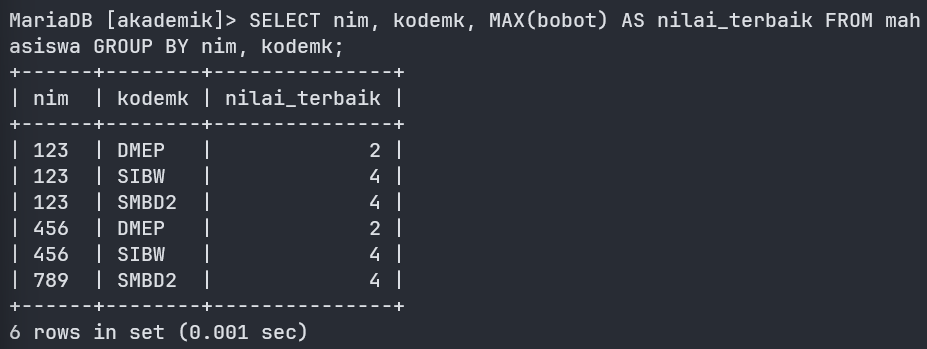


1. Masukan data berikut ke dalam tabel ”mahasiswa”

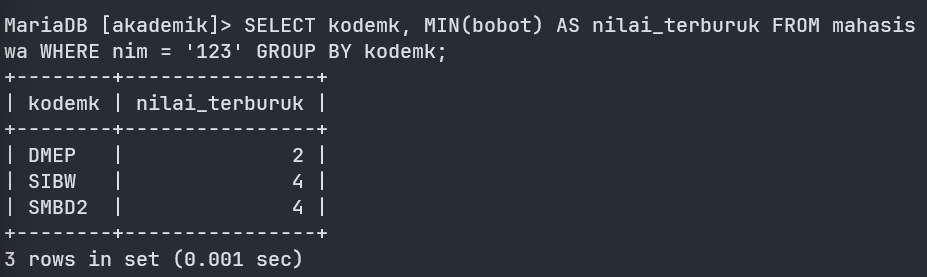




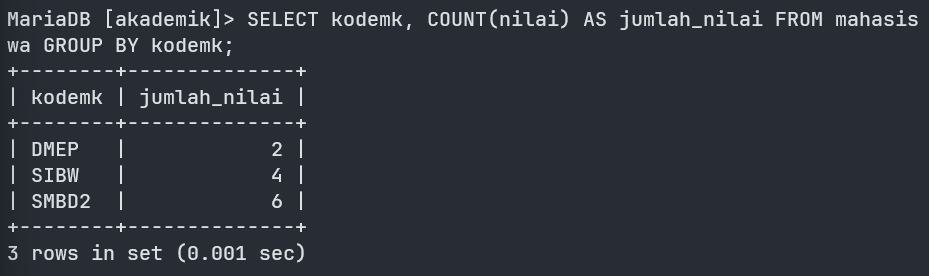
1. Tampilkan nilai terbaik yang didapatkan oleh seorang mahasiswa mata kuliah tertentu.



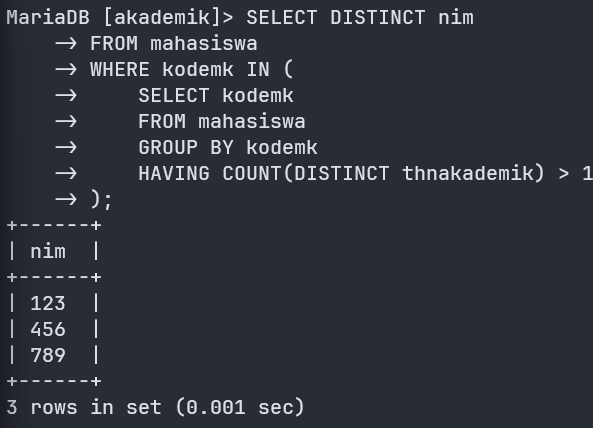
1. Tampilkan mata kuliah berserta nilai yang terburuk yang pernah didapatkan oleh mahasiswa dengan nim 123



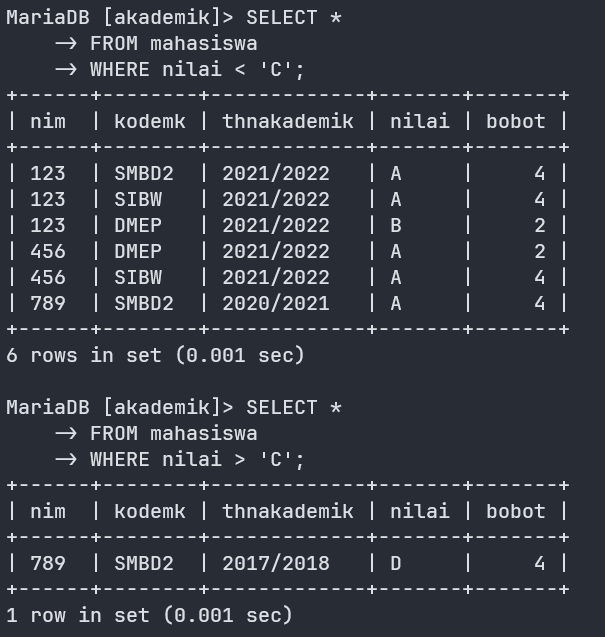
1. Tampilkan jumlah cacah nilai yang pernah diberikan untuk matakuliah tertentu



1. Tampilkan seluruh mahasiswa yang pernah mengulang sebuah mata kuliah



1. Tampilkan semua tabel dimana fieldnya terdiri dari nim, kodemk, thnakademik, nilai dimana nilai lebih kecil dari C.



**Analisis & Kesimpulan**

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

Namun, pentingnya juga untuk memperhatikan pengoptimalan kinerja saat menggunakan fungsi agregasi, terutama dalam konteks database besar. Memilih indeks yang tepat, merancang struktur query yang efisien, dan menghindari penggunaan subquery yang berlebihan dapat membantu meningkatkan kinerja query secara keseluruhan. Dengan demikian, praktikum ini tidak hanya meningkatkan pemahaman tentang agregasi dalam MySQL, tetapi juga memberikan wawasan tentang pentingnya desain query yang efisien untuk mengoptimalkan analisis data.