LAPORAN PRAKTIKUM BASIS DATA RC

Muhammad Yusuf
122140193

Tugas

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.
2. 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.
3. 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.
4. 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.
5. 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.
6. 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.
7. 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.
8. 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. Tabel produk

Produk_id	Produk_nama	Jumlah_stok	Supplier_id
P150	Kretendeng	80pcs	S002
P792	miesadap	30pcs	S005
P204	somos	50pcs	S005
P561	Marijan	40pcs	S004

```
MariaDB [galeri_itera]> select * from produk;
+-----------+--------------------+-------------+-------------+
| produk_id | produk_nama        | jumlah_stok | supplies_id |
+-----------+--------------------+-------------+-------------+
| P109      | Teh Kotak 300 ml   |          40 | S002        |
| P114      | Milo 100 ml        |         800 | S001        |
| P115      | Milo 150 ml        |          50 | S003        |
| P123      | Gulaku 1 Kg        |         100 | S005        |
| P150      | Kretendeng         |          80 | S006        |
| P204      | somos              |          50 | S006        |
| P235      | Aqua 250 ml        |         300 | S001        |
| P311      | Grand 320 ml       |         400 | S003        |
| P333      | Sari Roti 100 gram |          30 | S005        |
| P441      | Rojo Lele 5 kg     |          60 | S002        |
| P453      | Garam 30 gram      |          20 | S006        |
| P552      | Aqua 1 L           |         300 | S001        |
| P561      | Marijan            |          40 | S006        |
| P792      | miesadap           |          30 | S006        |
| P882      | Indomilk 25 ml     |         200 | S004        |
+-----------+--------------------+-------------+-------------+
15 rows in set (0.001 sec)
```

2. Tabel pegawai

Id_pegawai	pegawai_nama	jabatan	Jenis_kelamin	Alamat
Pg_007	Dani	Staff	Laki-laki	Jl. Suka Maju
Pg_008	Doni	Staff	Laki-laki	Jl. Suka Mundur
Pg_009	Dian	Staff	Perempuan	Jl. Ryacudu

```
MariaDB [galeri_itera]>
MariaDB [galeri_itera]> INSERT INTO pegawai VALUES
    -> ('Pg_007', 'Dani', 'Staff','Jl. Suka Maju', 'Laki-laki' ),
    -> ('Pg_008', 'Doni', 'Staff', 'Jl. Suka Mundur', 'Laki-laki'),
    -> ('Pg_009', 'Dian', 'Staff', 'Jl. Ryacudu', 'Perempuan');
Query OK, 3 rows affected (0.003 sec)
Records: 3  Duplicates: 0  Warnings: 0

MariaDB [galeri_itera]> select * from pegawai;
+------------+--------------+---------+-----------------+---------------+
| id_pegawai | pegawai_nama | jabatan | alamat          | jenis_kelamin |
+------------+--------------+---------+-----------------+---------------+
| pg_001     | Santi        | Cashier | Jl. Suka Maju   | Perempuan     |
| Pg_002     | Siska        | Casier  | Jl. Suka Mundur | Laki-laki     |
| Pg_003     | Nuri         | Casier  | Jl. Ryacudu     | Perempuan     |
| Pg_004     | Jamal        | Casier  | Jl. Suka Maju   | Laki-laki     |
| Pg_007     | Dani         | Staff   | Jl. Suka Maju   | Laki-laki     |
| Pg_008     | Doni         | Staff   | Jl. Suka Mundur | Laki-laki     |
| Pg_009     | Dian         | Staff   | Jl. Ryacudu     | Perempuan     |
| pg_201     | Santo        | Cashier | Jl. Suka Mundur | Perempuan     |
| pg_300     | Yaya         | Manager | Jl. Ryacudu     | Laki-laki     |
+------------+--------------+---------+-----------------+---------------+
9 rows in set (0.001 sec)
```

3. Tabel pembeli, menambahkan kota seperti dibawah ini

Kota
Jakarta
Bandung
Yogyakarta
Bandar Lampung
Surabaya

```sql
UPDATE pembeli
SET kota = 'Jakarta' WHERE id_pembeli = 'C_800';
UPDATE pembeli
SET kota = 'Bandung' WHERE id_pembeli = 'C_810';
UPDATE pembeli
SET kota = 'Yogyakarta' WHERE id_pembeli = 'C_890';
UPDATE pembeli
SET kota = 'Bandar Lampung' WHERE id_pembeli =
'C_901';
UPDATE pembeli
SET kota = 'Surabaya' WHERE id_pembeli = 'C_991';
```

```
MariaDB [galeri_itera]> select * from pembeli;
+------------+-------------+----------------+----------------+
| id_pembeli | pembeli_nama | pembeli_kontak | kota           |
+------------+-------------+----------------+----------------+
| C_800      | Egi          | 0812521221     | Jakarta        |
| C_810      | Ardi         | 0862145121     | Bandung        |
| C_890      | Prassetya    | 08521116464    | Yogyakarta     |
| C_901      | Rudi         | 081231511      | Bandar Lampung |
| C_991      | Andi         | 085212021111   | Surabaya       |
+------------+-------------+----------------+----------------+
5 rows in set (0.003 sec)
```

4. Tambahkan kolom kota pada tabel suppliers kemudian diurutkan

```
UPDATE suppliers
SET kota = 'Jakarta' WHERE suppliers_id = 'S001';
UPDATE suppliers
SET kota = 'Bandung' WHERE suppliers_id = 'S002';
UPDATE suppliers
SET kota = 'Yogyakarta' WHERE suppliers_id = 'S003';
UPDATE suppliers
SET kota = 'Bandar Lampung' WHERE suppliers_id =
'S004';
UPDATE suppliers
SET kota = 'Surabaya' WHERE suppliers_id = 'S005';
UPDATE suppliers
SET kota = 'Surabaya' WHERE suppliers_id = 'S006';
```

```
MariaDB [galeri_itera]> SELECT *
    -> FROM suppliers
    -> ORDER BY kota;
+--------------+--------------+-------------+----------------+
| suppliers_id | company_nama | nama_kontak | kota           |
+--------------+--------------+-------------+----------------+
| S004         | Pelita Baru  | Puspa       | Bandar Lampung |
| S002         | Suka Maju    | Rahmat      | Bandung        |
| S001         | Semua Terang | Ali         | Jakarta        |
| S005         | Surya Kun    | Siti        | Surabaya       |
| S006         | Ceria Kasih  | Topan       | Surabaya       |
| S003         | Maju Terus   | Dayono      | Yogyakarta     |
+--------------+--------------+-------------+----------------+
6 rows in set (0.005 sec)
```

5. Tampilkan nilai maksimal pada kolom jumlah stok

```
MariaDB [galeri_itera]> SELECT MAX(jumlah_stok) AS nilai_maksimum FROM produk;
+----------------+
| nilai_maksimum |
+----------------+
|            800 |
+----------------+
1 row in set (0.000 sec)
```

6. Tampilkan nilai rataan dari tabel produk pada tabel jumlah stok

```
MariaDB [galeri_itera]> SELECT AVG(jumlah_stok) AS nilai_rataan FROM produk;

+--------------+
| nilai_rataan |
+--------------+
|     166.6667 |
+--------------+
1 row in set (0.001 sec)
```

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.