

BAB 6

MENGENAL DAN MENGGUNAKAN FUNGSI

6.1. Tujuan Pembelajaran

1. Mengetahui dan Menggunakan Fungsi Agregat
2. Mengelompokkan dan Mengurutkan Tabel
3. Membatasi hasil Query

6.2. Dasar Teori

Fungsi agregat adalah fungsi yang digunakan untuk mengoleksi sejumlah data yang tersimpan pada suatu field dan menampilkan dalam bentuk nilai tunggal. Fungsi agregat yang ada dalam SQL adalah SUM, MAX, MIN, AVG dan COUNT. Fungsi agregat ini bisa digunakan pada perintah SELECT untuk menampilkan data dengan memunculkan nilai tunggal. Kegunaan masing-masing fungsi agregat antara lain :

- **SUM**, digunakan untuk menghitung total jumlah data yang tersimpan pada sebuah field. Bentuk umum penggunaan fungsi SUM adalah:
SUM(nama_field)
- **MAX**, digunakan untuk menentukan nilai tertinggi dari data yang tersimpan pada sebuah field. Bentuk umum penggunaan fungsi MAX adalah:
MAX(nama_field)
- **MIN**, digunakan untuk menentukan nilai terendah dari data yang tersimpan pada sebuah field. Bentuk umum penggunaan fungsi MAX adalah:
MIN(nama_field)
- **AVG**, digunakan untuk mencari nilai rata-rata dari data yang tersimpan pada sebuah field. Bentuk umum penggunaan fungsi AVG adalah:
AVG(nama_field)
- **COUNT**, digunakan untuk menghitung jumlah record dalam sebuah tabel. Bentuk umum penggunaan fungsi COUNT adalah:
COUNT(nama_field)

Setelah kita menggunakan fungsi, tentu kita bisa jadi bingung bila data yang ditampilkan tidak beraturan. Hal ini bisa kita atasi dengan mengelompokkan dan mengurutkan data yang ada di tabel dengan menggunakan ORDER BY, query nya bisa kita ketikkan seperti ini:

```
SELECT nama_field FROM nama_tabel GROUP BY nama_field
```

Oopss ternyata query yang kita buat menampilkan terlalu banyak data, bagaimana cara kita membatasi hasil query nya? Kita bisa menggunakan LIMIT, query nya bisa kita ketikkan seperti ini: SELECT * FROM Customers LIMIT 3;

6.3. Software

- XAMPP
- MySQL

6.4. Pembelajaran Mengenal dan Menggunakan Fungsi

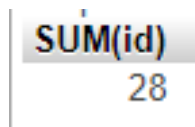
6.4.1. Sum

Fungsi `sum()` digunakan untuk melakukan penjumlahan isi field yang bertipe numerik yang namanya disebutkan pada nama field yang dijadikan parameter pada fungsi `sum()`. Misalkan kita ingin menampilkan jumlah id di tabel penerbit, maka kita bisa mengetikkan query seperti dibawah ini.

Contoh:

```
SELECT SUM(id) FROM penerbit
```

Output:



SUM(id)
28

Gambar 6.1 Hasil Fungsi Sum dari Tabel Penerbit

Perintah diatas digunakan untuk menampilkan jumlah dari id yang ada di tabel penerbit, dimana kita mengisi fungsi SUM dengan parameter id sehingga menampilkan jumlah nya yaitu 28. Bila tidak percaya kita lihat id di tabel penerbit, lalu kita jumlahkan.

id_penerbit	id	penerbit	kota
AOF	1	Andi Offset	Yogyakarta
EMK	2	Elex Media Komputind	Jakarta
IFB	6	Informatika Bandung	Bandung
MDH	3	Media Hidayah	Surakarta
MDK	4	Media Dakwah	Jakarta
MKD	5	Menara Kudus	Yogyakarta
TBY	7	Pustaka At Tibyan	Surakarta

Gambar 6.2 Menampilkan Isi Tabel Penerbit

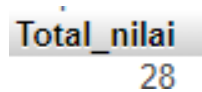
Coba kita hitung id di tabel penerbit yaitu $1 + 2 + 6 + 3 + 4 + 5 + 7 = 28$. Hasilnya samajadi kita tidak perlu lagi menghitung manual, cukup gunakan fungsi yang sudah tersedia di SQL.

Saat kita melihat output diatas kok nama field nyaa SUM(id)? Bagaimana cara kita merubah agar kita bisa merubah namanya menjadi lebih spesifik, kita bisa gunakan keyword AS atau alias.

Contoh:

```
SELECT SUM(id) AS Total_nilai FROM penerbit
```

Output:



Total_nilai
28

Gambar 6.3 Menampilkan Penjumlahan Id di Tabel Penerbit

Perintah diatas telah berhasil dilakukan, kita bisa memberi nama yang lebih spesifik tentang output yang kita mau. Maksud query diatas yaitu kita meminta SQL untuk menampilkan penjumlahan id di tabel penerbit dengan nama field yaitu Total_nilai.

6.4.2. Count

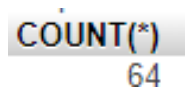
Fungsi ini dimaksudkan untuk mengetahui jumlah record dari suatu tabel, berdasarkan kondisi yang disertakan. Jika kondisi tidak ditulis, maka akan ditampilkan jumlah (semua) record dari tabel. Parameter yang disertakan bisa berupa nama field boleh juga diganti dengan tanda * .

Untuk mengetahui jumlah anggota yang terdaftar, maka kita bisa menggunakan perintah berikut ini:

Contoh:

```
SELECT COUNT(*) FROM anggota
```

Output:



COUNT(*)
64

Gambar 6.4 Menampilkan Jumlah Record Dari Tabel Anggota

Perintah diatas maksudnya yaitu kita meminta SQL untuk menampilkan jumlah record yang ada di tabel anggota, bisa dilihat kalau tabel anggota mempunyai record sebanyak 64.

Kita akan coba lagi menampilkan jumlah record tanpa ada duplikasi di dalamnya, karena tidak jarang dalam mengisi record kita tidak sadar bahwa ada duplikasi di dalamnya. Daripada bingung lebih baik lihat gambar dibawah ini terlebih dahulu.

id_anggota	nama_anggota	tgl_lahir	jklmn
052000	Ana	2001-01-01	2
0540003	Irma M	2001-01-01	2
0540004	Untung Subagyo Al - Kabumaeni	2001-01-01	1
053000	Tashya	2001-01-01	2
054000	Dyaning Utami Putri	2001-01-01	2
053000	Agatha kenang M	2001-01-01	2
0540008	Ririn Rikhul J	2001-01-01	2
0540009	Nurul Madaniyah	2001-01-01	2
0540010	Dyah Kumiawati	2001-01-01	2
0520011	Adelia Rosharyati	2001-01-01	2
0540012	Nur Rahmanita Fitriastuti	2001-01-01	2
0540013	Vina Agustina	2001-01-01	2
054000	Dyaning Utami Putri	2001-01-01	2

Gambar 6.5 Menampilkan Data Yang Ada Dari Tabel Anggota

Bisa dilihat bahwa nama anggota dengan nama Dyaning Utami Putri ada dua, yaitu di baris ke-5 dan di baris ke-13. Terlihat bahwa ada duplikasi di dalam tabel anggota yang kita buat, semisal ada kondisi seperti ini dan kita akan menampilkan data tapi tidak ingin ada yang sama, maka kita bisa menggunakan DISTINCT.

Apa kegunaan dari DISTINCT? DISTINCT digunakan untuk mengevaluasi rumus di setiap rekaman tabel dan menghasilkan tabel satu kolom pada dengan nilai duplikat yang dihapus. Misal kita ingin menampilkan data di tabel anggota dimana akan menghasilkan output **tidak ada nama anggota yang sama**, maka kita bisa mengetikkan query seperti ini.

Contoh:

```
SELECT COUNT(DISTINCT nama_anggota) AS Nama_anggota
FROM anggota
```

Output:

Nama_anggota
19

Gambar 6.6 Menampilkan Jumlah Record Dari Tabel Anggota Tanpa Duplikasi

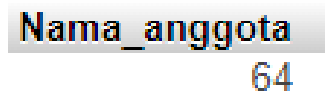
Perintah diatas maksudnya apa sih? Perintah diatas maksudnya adalah kita ingin menampilkan record di tabel anggota dengan nama field Nama_anggota tanpa ada nilai duplikasi. Dapat kita lihat record nya berjumlah 19.

Bagaimana kalau kita tidak menggunakan DISTINCT? Perhatikan perbandingan dibawah ini.

TIDAK MENGGUNAKAN DISTINCT:

```
SELECT COUNT (nama_anggota) AS Nama_anggota
FROM anggota
```

Output:



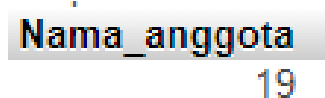
Nama_anggota
64

Gambar 6.7 Menampilkan Jumlah Record Dari Tabel Anggota

MENGGUNAKAN DISTINCT:

```
SELECT COUNT (DISTINCT  
nama_anggota) AS Nama_anggotaFROM anggota
```

Output:



Nama_anggota
19

Gambar 6.8 Menampilkan Jumlah Record Dari Tabel Anggota Tanpa Duplikasi

Dari perbandingan diatas kita bisa simpulkan, kalau menggunakan DISTINCT tidak akan ada data yang duplikat, berbeda dengan kita tanpa menggunakan DISTINCT akan ada data yang duplikat.

6.4.3. Avg

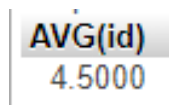
Fungsi ini dipakai untuk memperoleh nilai rata - rata suatu field yang bertipe numerik yang nama fieldnya disebutkan sebagai parameter. Biasanya kita akan menggunakan fungsi ini kalau kita ingin mengetahui rata-rata dari suatu nilai dan jangan lupa untuk memasukkan parameter nya, karena parameter ini penting untuk perhitungan di fungsinya.

Misalnya kita ingin menampilkan id di tabel pengarang, maka kita bisa mengetikkan query seperti dibawah ini:

Contoh:

```
SELECT AVG(id) FROM pengarang
```

Output:



AVG(id)
4.5000

Gambar 6.9 Menampilkan Rata-Rata Id Di Tabel Pengarang

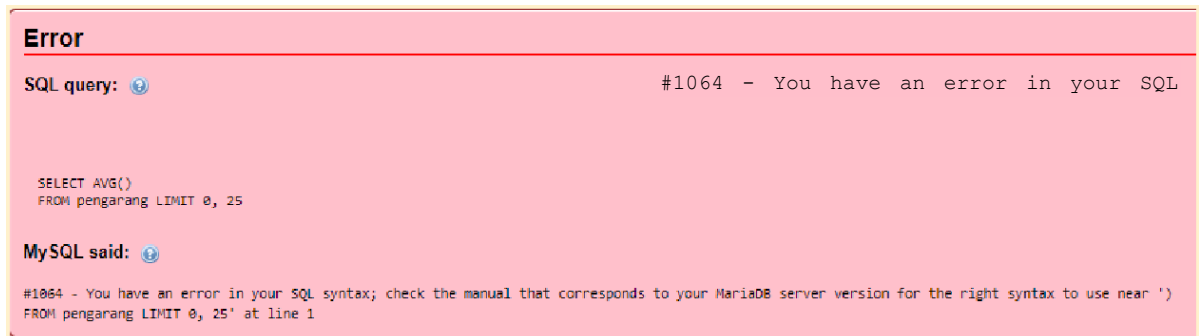
Perintah diatas maksudnya adalah kita ingin melihat rata-rata id di tabel pengarang, jadi ketika kita ingin menggunakan fungsi AVG() maka kita jangan lupa untuk memasukkan parameternya.

Ini adalah tampilan ketika kita lupa memasukkan parameter ketiga melakukan pemanggilan fungsi.

Contoh:

```
SELECT AVG()  
FROM pengarang
```

Output:



Gambar 6.10 Error Karena Tidak Ada Parameter Masukan

Apa sih maksud dari error diatas? Kalau ada error diatas kita lebih dulu harus membaca errornya, disini maksudnya ada yang kurang saat kita mengetik query. Tau dari mana???

Coba kita lihat dengan lebih teliti disitu bertulis

syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near ')
FROM pengarang LIMIT 0, 25' at line 1

Kita bisa fokus pada error yang bertulis `syntax to use near ')` berarti disini kita bisa mengetahui kalau kita lupa menuliskan parameter, karena SQL memberitahu sebelum tandakurung tutup harusnya kita masukan parameter untuk dihitungnya.

6.4.4. Max

Dengan menggunakan fungsi max, kita akan mendapatkan nilai terbesar dari field yang bertipe numerik yang nama fieldnya dituliskan dalam parameter. Misalnya kita ingin mengetahui tanggal lahir anggota yang termuda, maka kita bisa menggunakan perintah dibawah ini.

Contoh:

```
SELECT MAX(tgl_lahir) AS Tgllahir_Termuda FROM anggota
```

Output:

```
Tgllahir_Termuda  
2001-01-01
```

Gambar 6.11 Menampilkan Tanggal Lahir Termuda Di Tabel Anggota

Perintah diatas kita gunakan untuk mencari yang termuda, kenapa menggunakan fungsi MAX? Karena semakin muda umurnya maka semakin besar nilainya (kita tidak lihat dari seberapa lama dia hidup), tapi mudahnya seperti ini misal ada yang lahir tahun 2006 dan 1990 secara angka tentu yang 2006 adalah angka maksimalnya.

6.4.5. Min

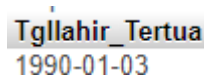
Dengan menggunakan fungsi min, kita akan mendapatkan nilai terkecil dari field yang

bertipe numerik yang nama fieldnya dituliskan dalam parameter. Misalnya kita ingin mengetahui tanggal lahir anggota yang tertua, maka kita bisa menggunakan perintah dibawah ini.

Contoh:

```
SELECT MIN(tgl_lahir) AS Tgllahir_Tertua FROM anggota
```

Output:



Tgllahir_Tertua
1990-01-03

Gambar 6.12 Menampilkan Tanggal Lahir Tertua Di Tabel Anggota

Perintah diatas kita gunakan untuk mencari yang tertua, kenapa menggunakan fungsi MIN? Karena semakin tua umurnya maka semakin kecil nilainya (kita tidak lihat dari seberapa lama dia hidup), tapi analoginya seperti ini misal ada yang lahir tahun 2006 dan 1990 secara angka tentu yang 2006 adalah angka minimalnya.

Mengelompokkan dan Mengurutkan Data

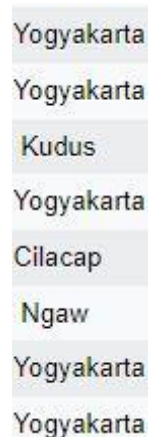
a . Group By

Untuk mengelompokkan data, kita bisa menggunakan perintah group by. Misalnya kita ingin mengetahui anggota dari perpustakaan ini lahinya dimana saja sih? Maka kita bisa menggunakan query seperti dibawah ini.

Contoh:

```
SELECT tmp_lahir from anggota
```

Output:



Yogyakarta
Yogyakarta
Kudus
Yogyakarta
Cilacap
Ngaw
Yogyakarta
Yogyakarta

Gambar 6.13 Menampilkan Tanggal Lahir Tertua Di Tabel Anggota

Tapi semua kota tampil dan ada yang berulang, Nah untuk mengatasinya kita bisa mengetikkan query seperti dibawah ini.

Contoh:

```
SELECT tmp_lahir FROM anggota GROUP BY tmp_lahir;
```

Output:

```
tmp_lahir
Kudus
Ngaw
Cilacap
Kebumen
Kudus
Wakai
Yogyakarta
```

Gambar 6.14 Menampilkan tmp_lahir dari Tabel Anggota Menggunakan Grouping

Perintah diatas maksudnya yaitu, kita ingin menampilkan tmp_lahir dari tabel anggota lalu kita lakukan grup untuk menggabungkan record yang sama. Contohnya seperti banyak sekali record yang bertuliskan 'Yogyakarta' yang membuat tidak efektif saat kita ingin membacanya, maka cara yang bisa kita lakukan yaitu menggabungkan record dengan melakukan grouping.

Mengurutkan Data

Order By adalah solusi yang bisa kita gunakan untuk mengurutkan data. Dalam mengurutkan data kita bisa meminta record untuk ditampilkan secara ascending maupun descending, secara default bernilai ascending.

Misalkan kita akan menampilkan data anggotaurut berdasarkan nama maka kita bisa mengetikkan query seperti ini:

Contoh:

```
select nama_anggota , tmp_lahir ,
tgl_lahir from anggota
order by nama_anggota;
```

Output:

nama_anggota	tmp_lahir	tgl_lahir
Ana	Yogyakarta	2001-01-01
Ana	Yogyakarta	2001-01-01
Dyaning Utami Putri	Yogyakarta	1990-01-03
Dyaning Utami Putri	Yogyakarta	1990-01-03
Dyaning Utami Putri	Yogyakarta	1990-01-03
Dyaning Utami Putri	Yogyakarta	1990-01-03
Dyaning Utami Putri	Yogyakarta	1990-01-03
Irma M	Yogyakarta	2001-01-01

Gambar 6.15 Menampilkan Tabel Anggota Diurutkan Berdasarkan Nama Anggota

Untuk mengurutkan berdasarkan tanggal lahir, kita bisa mengetikkan query seperti ini :

Contoh:

```
select nama_anggota , tmp_lahir ,
tgl_lahir from anggota
order by tgl_lahir;
```


Output:

nama_anggota	tmp_lahir	tgl_lahir ▲ 1
Dyaning Utami Putri	Yogyakarta	1990-01-03
Dyaning Utami Putri	Yogyakarta	1990-01-03
Dyaning Utami Putri	Yogyakarta	1990-01-03
Dyaning Utami Putri	Yogyakarta	1990-01-03
Dyaning Utami Putri	Yogyakarta	1990-01-03
Dyah Kumiawati	Yogyakarta	2001-01-01
Adelia Rosharyati	Yogyakarta	2001-01-01
Nur Rahmania Fitriastuti	Yogyakarta	2001-01-01
Vina Agustina	Kudus	2001-01-01
Yossi Dwiyuanna Septiani	Yogyakarta	2001-01-01

Gambar 6.16 Menampilkan Isian Tabel Anggota Diurutkan Berdasarkan Tanggal Lahir

Mengurutkan data anggota berdasarkan dua kriteria , tmp_lahir dan nama kita bisa mengetikkan query seperti ini:

Contoh:

```
select nama_anggota ,  
tmp_lahir  
from anggota  
order by tmp_lahir , nama_anggota;
```

Output:

nama_anggota ▲ 2	tmp_lahir ▲ 1
Vina Agustina	Kudus
Vina Agustina	Kudus
Vina Agustina	Kudus
Vina Agustina	Kudus
Vina Agustina	Kudus
Siti Marfulah	Ngaw
Siti Marfulah	Ngaw
Siti Marfulah	Ngaw
Siti Marfulah	Ngaw

Gambar 6.17 Menampilkan Tabel Anggota Berdasarkan Tanggal Lahir dan Nama Anggota

Untuk mengurutkan secara turun (descending) defaultnyaurut naik (ascending). Kita bisa mengetikkan query seperti ini.

Contoh:

```
select nama_anggota, tmp_lahir ,  
tgl_lahir  
from anggota order by  
tgl_lahir desc;
```

Output:

nama_anggota	tmp_lahir	tgl_lahir
Ana	Yogyakarta	2001-01-01
Ririn Rikhul J	Yogyakarta	2001-01-01
Nurul Madaniyah	Yogyakarta	2001-01-01
Dyah Kumiawati	Yogyakarta	2001-01-01
Adelia Rosharyati	Yogyakarta	2001-01-01
Nur Rahmania Fitriastuti	Yogyakarta	2001-01-01
Vina Agustina	Kudus	2001-01-01
Yossi Dwiyuanna Septiani	Yogyakarta	2001-01-01
Dwian Soffa Ardafit	Cilacap	2001-01-01
Siti Marfulah	Ngaw	2001-01-01
Ana	Yogyakarta	2001-01-01

Gambar 6.18 Menampilkan Isian Tabel Anggota Diurutkan Berdasarkan Tanggal Lahir Tertua

Menggabungkan Antara Fungsi Agregat , Pengelompokan dan Pengurutan

Menampilkan data jumlah anggota tiap - tiap kota tempat lahir urut dari jumlah terkecil sampai terbesar dan urut berdasarkan nama kota tempat lahir. Kita bisa mengetikkan query seperti ini:

Contoh:

```
SELECT tmp_lahir, COUNT(nama_anggota) as  
jumlah  
FROM anggota  
GROUP BY tmp_lahir  
ORDER BY jumlah,  
tmp_lahir;
```

Output:

tmp_lahir	jumlah
Kebumen	1
Wakai	1
Kudus	3
Ngaw	4
Cilacap	4
Kudus	5
Yogyakarta	46

Gambar 6.19 Menampilkan Isian Tabel Anggota Diurutkan Berdasarkan Tempat Lahir dan Jumlah

Menambahkan parameter limit

Menampilkan 5 data pertama, kita bisa mengetikkan query seperti ini.

Contoh:

```
select id_anggota,  
nama_anggota  
from anggota  
limit 5;
```

Output:

id_anggota	nama_anggota
052000	Ana
0540003	Irma M
0540004	Untung Subagyo Al - Kabumaeni
053000	Tashya
054000	Dyaning Utami Putri

Gambar 6.20 Menampilkan Id dan Nama Anggota dengan Limit Sebanyak 5

Menampilkan data mulai data ke 12 (setelah data ke 11) sampai data ke 17 (sebanyak 5 data) kita bisa mengetikkan query seperti ini.

Contoh:

```
select id_anggota,  
nama_anggota from anggota  
limit 11,5;
```

Output:

id_anggota	nama_anggota
0540013	Vina Agustina
054000	Dyaning Utami Putri
053000	Agatha kenang M
0540008	Ririn Rikhul J
0540009	Nurul Madaniyah

Gambar 6.21 Menampilkan id dan nama anggota dengan Limit 5 dimulai dari Index ke 11

Coba jalankan perintah berikut:

```
select nama_anggota , tgl_lahir from  
anggota order by tgl_lahir limit 1;  
select nama_anggota , tgl_lahir  
from anggota order by tgl_lahir desc limit 1 ;
```

Output:

nama_anggota	tgl_lahir
Dyaning Utami Putri	1990-01-03

Gambar 6.22 Menampilkan Satu Anggota Dengan Tanggal Lahir Tertua

6.5. Latihan

1. Bagaimana Cara Untuk Menghilangkan Duplikasi Nilai Saat Menggunakan SUM()!
2. Jelaskan Perbedaan SUM() dan COUNT()!