

About This Course :

Kamu akan memahami konsep penggunaan fungsi skalar dan fungsi agregat dalam operasi string dan numerik di SQL database. Kamu juga dapat mengerti konsep penggunaan GROUP BY dalam mengelompokkan data dan memahami konsep menggabungkan GROUP BY dengan fungsi aggregate. Selain itu, Kamu juga dapat mengerti penggunaan CASE Statement untuk struktur pengambilan keputusan.

Table of Content :

1. Fungsi di SQL
2. Fungsi Text di SQL
3. Fungsi Aggregate dan Group By
4. Mini Project

START

Module : Fungsi di SQL

Pendahuluan

Buat jadi analis data itu belajarnya banyak banget. Kemarin aku baru saja kelar belajar tentang 'Python', tapi kata Senja, aku masih perlu tahu soal SQL atau Structure Query Language. SQL ini nantinya akan jadi tools yang membantuku dalam menganalisis data perusahaan.

Apalagi, sebentar lagi perusahaan bakal merilis data penjualan tahun lalu. Jadi butuh sekali analis data untuk mengolahnya menjadi data berbentuk agregasi, bukan lagi raw data. Mendengarnya saja aku masih bingung.

Fungsi Scalar vs Fungsi Aggregate

Apa peranan fungsi dalam pengolahan data? Fungsi adalah metode yang digunakan untuk melakukan operasi data di database. Operasi ini bisa berupa kalkulasi numerik seperti sum, count, avg, etc; atau operasi non-numerik seperti string concatenations dan sub string.

SQL Function dapat dibagi dalam 2 kategori, yaitu fungsi scalar dan fungsi aggregate :

1. **Fungsi scalar** dalam SQL digunakan untuk mengembalikan nilai tunggal (single value) dari suatu nilai input yang diberikan
2. **Fungsi agregat** dalam SQL digunakan untuk melakukan perhitungan pada sekelompok nilai dan kemudian mengembalikan nilai tunggal.

Fungsi Skalar Matematika

Oke... kalau begitu **fungsi skalar pertama** yang akan kita bahas adalah fungsi skalar untuk numerik value. Fungsi ini umumnya digunakan jika kita ingin melakukan operasi matematika di SQL secara cepat dan efektif. Di SQL sendiri ada banyak fungsi matematika.

"Memangnya fungsi-fungsi apa saja yang bisa dilakukan di SQL?"

Untuk mengecek fungsi-fungsi apa saja yang bisa dilakukan di SQL, kita bisa membuka dokumentasi fungsi SQL di sini :

1. **postgresql database** : <https://www.postgresql.org/docs/9.5/functions-math.html>
2. **mysql database** : <https://dev.mysql.com/doc/refman/8.0/en/mathematical-functions.html>

Berikut adalah beberapa fungsi matematika umum yang biasa digunakan:

Fungsi	Deskripsi	Contoh	Hasil
<code>abs()</code>	Mengembalikan nilai absolute dari nilai input	<code>abs(-17.4)</code>	17.4
<code>ceiling()</code>	Mengembalikan nilai integer terbesar yang terdekat dengan nilai input	<code>ceiling(-95.3)</code>	-95
<code>floor()</code>	Mengembalikan nilai integer terkecil yang terdekat dengan nilai input	<code>floor(-42.8)</code>	-43
<code>round()</code>	Mengembalikan nilai pembulatan dari suatu nilai decimal	<code>round(42.4382, 2)</code>	42.44
<code>sqrt()</code>	Mengembalikan nilai akar kuadrat dari nilai input	<code>sqrt(2.0)</code>	1.41421356
<code>mod()</code>	Mengembalikan nilai sisa hasil pembagian dari nilai input	<code>mod(9,4)</code>	1
<code>exp()</code>	Mngembalikan nilai ekponensial dari nilai input	<code>exp(1.0)</code>	2.71828183

Untuk lebih memahami fungsi matematika, berikut diberikan tabel dummy yang berisi nilai siswa semester 1 dan 2 di suatu sekolah. Berikut contoh fungsi skalar dengan menggunakan tabel dummy:

Tabel: students

StudentID	FirstName	LastName	Email	Semester1	Semester2	MarkGrowth
1	Jose	Mohit	Jose_Mohit@gmail.com	64.55	72.60	-8.05
2	Lala	Karlina	lala.karlina@yahoo.com	72.85	65.35	7.50
3	Sultan	Hadi	Sultan_Hadi@gmail.com	45.32	50.25	-4.93
4	Jaya	Usman	jaya.usman@yahoo.com	86.73	77.40	9.33
5	Anjali	Wijaya	anjali.wijaya@yahoo.com	92.25	90.75	1.50

Fungsi Skalar Matematika – ABS()

Fungsi ABS() adalah mengembalikan nilai absolute dari nilai input. Berdasarkan data table: students berikut penggunaannya.

Syntax:

```
SELECT ABS(ColumnNme)
FROM TableName;
```

Penerapan:

```
1 SELECT StudentID, FirstName, LastName, Semester1, Semester2, ABS(MarkGrowth) AS MarkGrowth
2 FROM students;
```

Hasil:

StudentID	FirstName	LastName	Semester1	Semester2	MarkGrowth
1	Jose	Mohit	64.55	72.6	8.05
2	Lala	Karlina	72.85	65.35	7.5
3	Sultan	Hadi	45.32	50.25	4.93
4	Jaya	Usman	86.73	77.4	9.33
5	Anjali	Wijaya	92.25	90.75	1.5

Fungsi Skalar Matematika – CEILING()

Fungsi CEILING() – Mengembalikan nilai integer terbesar yang terdekat dengan nilai input.

Syntax:

```
SELECT CEILING(ColumnName)
FROM TableName;
```

Penerapan:

```
1 SELECT StudentID, FirstName, LastName, CEILING(Semester1) as Semester1, CEILING(Semester2) as
   Semester2, MarkGrowth
2 FROM students;
```

Hasil:

StudentID	FirstName	LastName	Semester1	Semester2	MarkGrowth
1	Jose	Mohit	65	73	-8.05
2	Lala	Karlina	73	66	7.5
3	Sultan	Hadi	46	51	-4.93
4	Jaya	Usman	87	78	9.33
5	Anjali	Wijaya	93	91	1.5

Fungsi Skalar Matematika – FLOOR()

Fungsi FLOOR() – Mengembalikan nilai integer terkecil yang terdekat dengan nilai input.

Syntax:

```
SELECT FLOOR(ColumnName)
FROM TableName;
```

Penerapan:

```
1 SELECT StudentID, FirstName, LastName, FLOOR(Semester1) AS Semester1, FLOOR(Semester2) AS Semester2,
   MarkGrowth
2 FROM students;
```

Hasil:

StudentID	FirstName	LastName	Semester1	Semester2	MarkGrowth
1	Jose	Mohit	64	72	-8.05
2	Lala	Karlina	72	65	7.5
3	Sultan	Hadi	45	50	-4.93
4	Jaya	Usman	86	77	9.33
5	Anjali	Wijaya	92	90	1.5

Fungsi Skalar Matematika – ROUND()

Fungsi ROUND() – Mengembalikan nilai pembulatan dari suatu nilai decimal.

Syntax:

```
SELECT ROUND(ColumnNane)
FROM TableName;
```

Penerapan:

```
1 SELECT StudentID, FirstName, LastName, ROUND(Semester1,1) AS Semester1, ROUND(Semester2,0) AS
   Semester2, MarkGrowth
2 FROM students;
```

Hasil:

StudentID	FirstName	LastName	Semester1	Semester2	MarkGrowth
1	Jose	Mohit	64.6	73	-8.05
2	Lala	Karlina	72.8	65	7.5
3	Sultan	Hadi	45.3	50	-4.93
4	Jaya	Usman	86.7	77	9.33
5	Anjali	Wijaya	92.2	91	1.5

Fungsi Skalar Matematika – SQRT()

Fungsi SQRT() – Mengembalikan nilai akar kuadrat dari nilai input.

Syntax:

```
SELECT SQRT(ColumnNane)
FROM TableName;
```

Penerapan:

```
1 SELECT StudentID, FirstName, LastName, SQRT(Semester1) AS Semester1, Semester2, MarkGrowth
2 FROM students;
```

Hasil:

StudentID	FirstName	LastName	Semester1	Semester2	MarkGrowth
1	Jose	Mohit	8.034301463101817	72.6	-8.05
2	Lala	Karlina	8.535221145348256	65.35	7.5
3	Sultan	Hadi	6.732013071882734	50.25	-4.93
4	Jaya	Usman	9.31289428695505	77.4	9.33
5	Anjali	Wijaya	9.604686356149273	90.75	1.5

Tugas Praktek

Menggunakan fungsi MOD() untuk menghitung nilai sisa jika nilai Semester1 dibagi 2 dan fungsi EXP() untuk menghitung nilai eksponensial dari nilai MarkGrowth. Gunakan kedua fungsi tersebut dalam satu SELECT Statement.

Penerapan:

```
1 |SELECT StudentID, FirstName, LastName, MOD(Semester1,2) AS Semester1, Semester2, EXP(MarkGrowth)
2 |FROM students;
```

Hasil:

StudentID	FirstName	LastName	Semester1	Semester2	EXP(MarkGrowth)
1	Jose	Mohit	0.5499999999999972	72.6	0.00031910192248120326
2	Lala	Karlina	0.8499999999999943	65.35	1808.0424144560632
3	Sultan	Hadi	1.3200000000000003	50.25	0.0072265032813764625
4	Jaya	Usman	0.7300000000000004	77.4	11271.131485524471
5	Anjali	Wijaya	0.25	90.75	4.4816890703380645

Module : Fungsi Text di SQL

Fungsi Text

Fungsi skalar kedua untuk text/string value digunakan jika kita ingin melakukan operasi pada text atau karakter di SQL. Misalnya mengubah huruf kecil ke besar, menghitung jumlah karakter dari text, dll.

Fungsi skalar text secara lengkap dapat dilihat di dokumentasi berikut:

1. **PostgreSQL**: <https://www.postgresql.org/docs/9.1/functions-string.html>
2. **MySQL**: <https://dev.mysql.com/doc/refman/8.0/en/string-functions.html>

Berikut adalah beberapa fungsi yang secara umum sering digunakan:

Fungsi	Deskripsi	Contoh	Hasil
<code>concat()</code>	Menggabungkan semua argumen/input, NULL value akan diabaikan	<code>concat('abcde', 2, NULL, 22)</code>	abcde222
<code>split_part()</code>	Membagi string/text berdasarkan delimiter/pemisah yang ditentukan, kemudian mengembalikan value berdasarkan posisi string yang diinginkan (hitungan posisi mulai dari 1)	<code>split_part('abc@def@ghi', '@', 2)</code>	def
<code>substr()</code>	Mengekstrak karakter/string yang diinginkan	<code>substr('alphabet', 3, 2)</code>	ph
<code>length()</code>	Menghitung jumlah karakter dalam string/text	<code>length('jose')</code>	4
<code>replace()</code>	Mengganti karakter dalam suatu string/text	<code>replace('abcdefabcdef', 'cd', 'XX')</code>	abXXefabXXef
<code>trim()</code>	Menghapus karakter dalam suatu string	<code>trim(both 'x' from 'xTomxx');</code> <code>trim(leading 'x' from 'xTomxx');</code> <code>trim(trailing 'x' from 'xTomxx');</code>	Tom; Tomxx; xTom;
<code>upper()</code>	Mengubah huruf kecil ke huruf besar	<code>upper('tom')</code>	TOM
<code>lower()</code>	Mengubah huruf besar ke huruf kecil	<code>lower('TOM')</code>	tom

Tabel: students

StudentID	FirstName	LastName	Email	Semester1	Semester2	MarkGrowth
1	Jose	Mohit	Jose_Mohit@gmail.com	64.55	72.60	-8.05
2	Lala	Karlina	lala.karlina@yahoo.com	72.85	65.35	7.50
3	Sultan	Hadi	Sultan_Hadi@gmail.com	45.32	50.25	-4.93
4	Jaya	Usman	jaya.usman@yahoo.com	86.73	77.40	9.33
5	Anjali	Wijaya	anjali.wijaya@yahoo.com	92.25	90.75	1.50

Fungsi Text – CONCAT()

Menggabungkan semua argument/input, NULL value akan diabaikan.

Penerapan:

```
1 SELECT StudentID, CONCAT(FirstName, LastName) AS Name, Semester1, Semester2, MarkGrowth
2 FROM students;
```

Contoh:

StudentID	Name	Semester1	Semester2	MarkGrowth
1	JoseMohit	64.55	72.6	-8.05
2	LalaKarlina	72.85	65.35	7.5
3	SultanHadi	45.32	50.25	-4.93
4	JayaUsman	86.73	77.4	9.33
5	AnjaliWijaya	92.25	90.75	1.5

Fungsi Text – SUBSTRING_INDEX()

Membagi string/text berdasarkan delimiter/pemisah yang ditentukan, kemudian mengembalikan value berdasarkan posisi string yang diinginkan.

Syntax:

```
SELECT SUBSTRING_INDEX(column, delimiter, index to return)
FROM TableName;
```

Keterangan:

column --> merupakan nama kolom yang akan dipecah text-nya,

delimiter --> karakter atau gabungan beberapa karakter untuk pemecah text pada kolom bersangkutan,

index_to_return --> indeks dari pecahan text yang akan diambil.

Penerapan:

```
1 SELECT StudentID, Email, SUBSTRING_INDEX(Email, '@', 1) AS Name
2 FROM students;
```

Hasil:

Dapat di perhatikan bagaimana pengaruh dari delimiter dan index to return.

StudentID	Email	Name
1	Jose_Mohit@gmail.com	Jose_Mohit
2	lala_karlina@yahoo.com	lala_karlina
3	Sultan_Hadi@gmail.com	Sultan_Hadi
4	jaya_usman@yahoo.com	jaya_usman
5	anjali_wijaya@yahoo.com	anjali_wijaya

Fungsi Text – SUBSTR()

Mengekstrak karakter/string yang diinginkan.

Syntax:

```
SELECT SUBSTR(columnName, Start Index, Number of string to be extract)
FROM TableName;
```

Keterangan:

columnName --> nama kolom yang akan dicari substring-nya

Start Index --> indeks dari text yang dimiliki (dimulai dari 1)

Number of string to be extract --> jumlah karakter atau beberapa karakter yang akan diambil.

Penerapan:

```
1 SELECT StudentID, FirstName, SUBSTR(FirstName, 2, 3) AS Initial
2 FROM students;
```

Hasil:

StudentID	FirstName	Initial
1	Jose	ose
2	Lala	ala
3	Sultan	ult
4	Jaya	aya
5	Anjali	nja

Fungsi Text – LENGTH()

Menghitung jumlah karakter dalam string.

Syntax:

```
SELECT LENGTH(Column Name)
FROM TableName;
```

Penerapan:

```
1 SELECT StudentID, FirstName, LENGTH(FirstName) AS Total_Char
2 FROM students;
```

Hasil:

StudentID	FirstName	Total_Char
1	Jose	4
2	Lala	4
3	Sultan	6
4	Jaya	4
5	Anjali	6

Fungsi Text – REPLACE()

Mengganti karakter pada suatu string/text.

Syntax:

```
SELECT REPLACE(ColumnNme, Character/String to be change, New String/Character)
FROM TableName;
```

Penerapan:

```
1 SELECT StudentID, Email, REPLACE(Email, 'yahoo', 'gmail') AS New_Email
2 FROM students;
```

Hasil:

StudentID	Email	New_Email
1	Jose_Mohit@gmail.com	Jose_Mohit@gmail.com
2	lala_karlina@yahoo.com	lala_karlina@gmail.com
3	Sultan_Hadi@gmail.com	Sultan_Hadi@gmail.com
4	jaya_usman@yahoo.com	jaya_usman@gmail.com
5	anjali_wijaya@yahoo.com	anjali_wijaya@gmail.com

Tugas Praktek

Penggunaan UPPER() dan LOWER() untuk mengubah karakter menjadi huruf kapital ataupun non-kapital.

Penerapan:

```
1 SELECT StudentID, UPPER(FirstName) AS FirstName, LOWER(LastName) AS LastName
2 FROM students;
```

Hasil:

StudentID	FirstName	LastName
1	JOSE	mohit
2	LALA	karlina
3	SULTAN	hadi
4	JAYA	usman
5	ANJALI	wijaya

Module : Fungsi Aggregate dan Group By

Memahami fungsi aggregate dan group by di SQL untuk mengagregasi data dan mengelompokkan data berdasarkan kriteria tertentu.

Fungsi Aggregate

Fungsi aggregate digunakan untuk melakukan perhitungan pada sekelompok nilai.

Berikut fungsi – fungsi aggregate yang umumnya digunakan.

Fungsi	Deskripsi
SUM()	Digunakan untuk menjumlahkan sekelompok nilai (baris) dalam satu kolom
COUNT()	Digunakan untuk menghitung jumlah baris
AVG()	Digunakan untuk menghitung nilai rata - rata dari suatu kolom
MIN()	Digunakan untuk menghitung nilai minimum dari suatu kolom
MAX()	Digunakan untuk menghitung nilai maximum dari suatu kolom
FIRST()	Mengembalikan nilai pada baris pertama dari suatu kolom
LAST()	Mengembalikan nilai pada baris terakhir dari suatu kolom

Tabel: students

StudentID	FirstName	LastName	Email	Semester1	Semester2	MarkGrowth
1	Jose	Mohit	Jose_Mohit@gmail.com	64.55	72.60	-8.05
2	Lala	Karlina	lala.karlina@yahoo.com	72.85	65.35	7.50
3	Sultan	Hadi	Sultan_Hadi@gmail.com	45.32	50.25	-4.93
4	Jaya	Usman	jaya.usman@yahoo.com	86.73	77.40	9.33
5	Anjali	Wijaya	anjali.wijaya@yahoo.com	92.25	90.75	1.50

Penerapan SUM, COUNT, AVG dalam satu statement:

```
1 SELECT COUNT(FirstName) AS Jumlah_baris,  
2 SUM(Semester1) AS Sum_1, SUM(Semester2) AS Sum_2,  
3 AVG(Semester1) AS AVG_1, AVG(Semester2) AS AVG_2  
4 FROM students;
```

Hasil:

Hasil ini tidak memiliki pengertian hanya contoh penggunaan fungsi saja. Karena tidak mungkin jumlah baris dalam satu row dengan sum dan rata-rata.

```
+-----+-----+-----+-----+-----+  
| Jumlah_baris | Sum_1 | Sum_2 | AVG_1 | AVG_2 |  
+-----+-----+-----+-----+-----+  
|          5 | 361.7 | 356.35 | 72.34 | 71.270000000000001 |  
+-----+-----+-----+-----+-----+
```

Penerapan MAX() dan MIN():

```
1 SELECT MIN(Semester1) AS Min1, MAX(Semester1) AS Max1, MIN(Semester2) AS Min2, MAX(Semester2) AS  
   Max2  
2 FROM students;
```

Hasil:

```
+-----+-----+-----+-----+  
| Min1 | Max1 | Min2 | Max2 |  
+-----+-----+-----+-----+  
| 45.32 | 92.25 | 50.25 | 90.75 |  
+-----+-----+-----+-----+
```

Fungsi GROUP BY

Untuk mengelompokkan data di SQL kita menggunakan **GROUP BY statement**. GROUP BY statement akan mengelompokkan data yang bernilai sama ke dalam satu group, kemudian dengan menggunakan fungsi aggregate seperti (count, max, min, sum, avg) kita bisa melakukan agregasi untuk setiap group atau kelompok yang terbentuk.

Contohnya, mengolah data penjualan tahun sebelumnya, lalu mengelompokkan dan menghitung berdasarkan penjualan setiap provinsi maupun dikelompokkan per bulan.

Berikut Syntax yang digunakan:

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
ORDER BY column_name(s);
```

Hal Penting:

- GROUP BY digunakan dengan SELECT, artinya kolom yang digunakan di GROUP BY statement, juga perlu ditempatkan di SELECT.
- GROUP BY ditempatkan setelah WHERE, tetapi jika tidak menggunakan WHERE maka langsung ditempatkan setelah FROM.
- Jika menggunakan ORDER BY, maka GROUP BY ditempatkan sebelum ORDER BY.

Grouping dapat dilakukan dengan cara:

- Group by Single Column, data dikelompokkan menggunakan kriteria dari satu kolom saja, misalnya mengelompokkan data berdasarkan province saja.
- Group by Multiple Column, data dikelompokkan menggunakan kriteria dari dua kolom atau lebih, misalnya mengelompokkan data berdasarkan province dan brand.

Group by Single Column

Fungsi ini memastikan data dapat dikelompokkan menggunakan kriteria dari satu kolom saja, misalnya mengelompokkan data berdasarkan provinsi saja.

Penerapan:

Pengelompokan data `total_order` dari banyaknya `order_id` dan `total_price` dari jumlah `item_price` untuk setiap provinsi.

Fungsi `DISTINCT` untuk menghilangkan duplikasi.

```
1 SELECT province,  
2 COUNT(DISTINCT order_id) as total_order,  
3 SUM(item_price) as total_price  
4 FROM sales_retail_2019  
5 GROUP BY province;
```

Hasil:

province	total_order	total_price
Aceh	16	575100000
Bali	454	12555567000
Bangka Belitung	8	378494000
Banten	596	14925141000
DKI Jakarta	8332	183872878000
Jambi	55	3030991000
Jawa Barat	2831	62982148000
Jawa Tengah	1475	40508136000
Jawa Timur	1406	28809529000
Kalimantan Barat	18	423139000
Kalimantan Selatan	26	1195590000
Kalimantan Tengah	275	13057098000
Lampung	37	1226267000
NTB	5	146872000
Riau	91	2138244000
Sulawesi Selatan	384	12822560000
Sulawesi Tengah	53	2326860000
Sulawesi Utara	41	1278392000
Sumatra Barat	92	1852032000
Sumatra Selatan	121	5224917000
Sumatra Utara	145	3710699000
unknown	57	8256525000
Yogyakarta	1318	27181441000

Group by Multiple Column

Dengan fungsi ini data dapat dikelompokkan menggunakan kriteria dari dua kolom atau lebih, misalnya mengelompokkan data berdasarkan province dan brand.

Penerapan:

Pengelompokan data total_order dari banyaknya order_id dan total_price dari jumlah item_price untuk setiap provinsi dan semua brand.

```
1 SELECT province, brand,  
2 COUNT(DISTINCT order_id) as total_order,  
3 SUM(item_price) as total_price  
4 FROM sales_retail_2019  
5 GROUP BY province, brand;
```

Hasil:

province	brand	total_order	total_price
Aceh	BRAND_A	9	13172000
Aceh	BRAND_B	10	56795000
Aceh	BRAND_C	9	42598000
Aceh	BRAND_D	7	34080000
Aceh	BRAND_E	1	8380000
Aceh	BRAND_F	6	11596000
Aceh	BRAND_G	9	17435000
Aceh	BRAND_H	2	2965000
Aceh	BRAND_I	2	6311000
Aceh	BRAND_J	7	10225000
Aceh	BRAND_K	4	3318000
Aceh	BRAND_L	4	5492000
Aceh	BRAND_M	9	15870000
Aceh	BRAND_O	2	3835000
Aceh	BRAND_P	16	204486000
Aceh	BRAND_R	11	63277000
Aceh	BRAND_S	12	49185000
Aceh	BRAND_T	6	5864000
Aceh	BRAND_V	3	10797000

Penerapan:

Menggunakan filter untuk menampilkan data dari semua provinsi dan BRAND_A saja.

```
1 SELECT province, brand,  
2 COUNT(DISTINCT order_id) as total_order,  
3 SUM(item_price) as total_price  
4 FROM sales_retail_2019  
5 WHERE brand = "BRAND_A"  
6 GROUP BY province, brand;
```

Hasil:

province	brand	total_order	total_price
Aceh	BRAND_A	9	13172000
Bali	BRAND_A	221	501458000
Bangka Belitung	BRAND_A	3	22060000
Banten	BRAND_A	333	798540000
DKI Jakarta	BRAND_A	4460	9856196000
Jambi	BRAND_A	29	36928000
Jawa Barat	BRAND_A	1476	3424649000
Jawa Tengah	BRAND_A	672	1994833000
Jawa Timur	BRAND_A	726	1780805000
Kalimantan Barat	BRAND_A	10	42739000
Kalimantan Selatan	BRAND_A	16	41716000
Kalimantan Tengah	BRAND_A	175	567703000
Lampung	BRAND_A	14	39496000
NTB	BRAND_A	3	2812000
Riau	BRAND_A	39	65245000
Sulawesi Selatan	BRAND_A	189	454875000
Sulawesi Tengah	BRAND_A	31	106156000
Sulawesi Utara	BRAND_A	10	13995000
Sumatra Barat	BRAND_A	41	96511000
Sumatra Selatan	BRAND_A	76	177359000
Sumatra Utara	BRAND_A	73	190182000
unknown	BRAND_A	34	126872000
Yogyakarta	BRAND_A	720	1571458000

Penggunaan Case ... When ...

Bagaimana jika ingin menambahkan kolom rekomendasi atau remark dari hasil agregasi data? Misalnya, untuk menambahkan remark 'Target Achieved' pada penjualan bulan Maret 2019 lebih dari 30M.

Untuk melakukan remark maka menggunakan Syntax berikut:

```
SELECT ColumnName1, ColumnName2,  
CASE  
    WHEN condition1 THEN result1  
    WHEN condition2 THEN result2  
    WHEN conditionN THEN resultN  
    ELSE result  
END as alias  
FROM TableName;
```

CASE – statement: akan mengevaluasi kondisi yang sudah ditentukan, dimulai condition1, akan menampilkan result1, jika kondisi terpenuhi (TRUE). Jika tidak, maka akan turun ke condition2, dst. Jika tidak ada kondisi yang terpenuhi maka result pada bagian ELSE akan dikembalikan.

Tugas Praktek:

Dengan menggunakan data sales_retail_2019, buatlah syntax query yang menggunakan fungsi skalar MONTH() untuk mengubah order_date dari tanggal ke bulan, fungsi aggregate SUM() untuk menjumlahkan kolom item_price.

Tambahkan kolom remark menggunakan CASE... WHEN... statement. Jika sum(item_price) >= 30.000.000.000, maka remark-nya 'Target Achieved'; Jika sum(item_price) <= 25.000.000.000 maka remark-nya 'Less performed'; Selain itu, beri remark 'Follow Up'.

Penerapan:

```
1 SELECT MONTH(order_date) AS order_month, SUM(item_price)  
   AS total_price,  
2 CASE  
3     WHEN SUM(item_price) >= 30000000000 THEN 'Target  
   Achieved'  
4     WHEN SUM(item_price) <= 25000000000 THEN 'Less  
   performed'  
5     ELSE 'Follow Up'  
6 END as remark  
7 FROM sales_retail_2019  
8 GROUP BY order_month;
```

Hasil:

order_month	total_price	remark
1	21872088000	Less performed
2	23872553000	Less performed
3	24729248000	Less performed
4	33436592000	Target Achieved
5	29464257000	Follow Up
6	32428828000	Target Achieved
7	31704144000	Target Achieved
8	28859392000	Follow Up
9	27425513000	Follow Up
10	49548111000	Target Achieved
11	54841176000	Target Achieved
12	70296718000	Target Achieved

Module : Mini Project

Studi Kasus untuk memahami kasus sederhana untuk mengukur pemahaman modul fundamental SQL Using Function dan GROUP BY.

Kasus: Saya minta tolong agar kamu melakukan analisis penjualan di suatu store. Adapun laporan yang diminta sebagai berikut:

1. Total jumlah seluruh penjualan (Total/revenue)
2. Total quantity seluruh produk yang terjual
3. Total quantity dan total revenue untuk setiap kode produk
4. Rata-rata total belanja per kode pelanggan
5. Selain itu, jangan lupa menambahkan kolom baru dengan nama “kategori” yang mengkategorikan total/revenue ke dalam 3 kategori: High > 300K; Medium:100K-300K; Low:<100K

Data Table:

kode_transaksi	kode_pelanggan	no_urut	kode_produk	nama_produk	qty	total
tr-001	dqlabcust07	1	prod-01	Kotak Pensil DQLab	5	312500
tr-001	dqlabcust07	2	prod-03	Flash disk DQLab 32 GB	1	100000
tr-001	dqlabcust07	3	prod-09	Buku Planner Agenda DQLab	3	276000
tr-001	dqlabcust07	4	prod-04	Flashdisk DQLab 32 GB	3	120000
tr-002	dqlabcust01	1	prod-03	Gift Voucher DQLab 100rb	2	200000
tr-002	dqlabcust01	2	prod-10	Sticky Notes DQLab 500 sheets	4	220000
tr-002	dqlabcust01	3	prod-07	Tas Travel Organizer DQLab	1	48000
tr-003	dqlabcust03	1	prod-02	Flashdisk DQLab 64 GB	2	110000
tr-004	dqlabcust03	1	prod-10	Sticky Notes DQLab 500 sheets	5	275000
tr-004	dqlabcust03	2	prod-04	Flashdisk DQLab 32 GB	4	160000
tr-005	dqlabcust05	1	prod-09	Buku Planner Agenda DQLab	3	276000
tr-005	dqlabcust05	2	prod-01	Kotak Pensil DQLab	1	62500
tr-005	dqlabcust05	3	prod-04	Flashdisk DQLab 32 GB	2	80000
tr-006	dqlabcust02	1	prod-05	Gift Voucher DQLab 250rb	4	1000000
tr-006	dqlabcust02	2	prod-08	Gantungan Kunci DQLab	2	31600

Data Penjualan Part 1

Laporan 1 – 3:

```
1 ## 1. Total jumlah seluruh penjualan (total/revenue).
2 SELECT SUM(total) as total
3 FROM tr_penjualan;
4 ## 2. Total quantity seluruh produk yang terjual.
5 SELECT SUM(qty) as qty
6 FROM tr_penjualan;
7 ## 3. Total quantity dan total revenue untuk setiap kode
   produk.
8 SELECT kode_produk, SUM(qty) as qty, SUM(total) as total
9 FROM tr_penjualan
10 GROUP BY kode_produk;
```

Hasil :

```
+-----+
| total  |
+-----+
| 3271600 |
+-----+

+-----+
| qty   |
+-----+
| 42   |
+-----+

+-----+-----+-----+
| kode_produk | qty | total  |
+-----+-----+-----+
| prod-01    | 6  | 375000 |
| prod-02    | 2  | 110000 |
| prod-03    | 3  | 300000 |
| prod-04    | 9  | 360000 |
| prod-05    | 4  | 1000000 |
| prod-07    | 1  | 48000  |
| prod-08    | 2  | 31600  |
| prod-09    | 6  | 552000 |
| prod-10    | 9  | 495000 |
+-----+-----+-----+
```

Laporan 4 – 5:

```
1 ## 4. Rata - Rata total belanja per kode pelanggan.
2 SELECT kode_pelanggan, AVG(total) as avg_total
3 FROM tr_penjualan
4 GROUP BY kode_pelanggan;
5 ## 5. Selain itu, jangan lupa untuk menambahkan kolom baru dengan nama 'kategori' yang
   mengkategorikan total/revenue ke dalam 3 kategori: High: > 300K; Medium: 100K - 300K; Low: <100K.
6 SELECT kode_transaksi,kode_pelanggan,no_urut,kode_produk, nama_produk, qty, total,
7 CASE
8     WHEN total > 300000 THEN 'High'
9     WHEN total < 100000 THEN 'Low'
10    ELSE 'Medium'
11 END as kategori
12 FROM tr_penjualan;
```

Hasil :

kode_pelanggan	avg_total
dqlabcust01	156000.0000
dqlabcust02	515000.0000
dqlabcust03	181666.6667
dqlabcust05	139500.0000
dqlabcust07	202125.0000

kode_transaksi	kode_pelanggan	no_urut	kode_produk	nama_produk	qty	total	kategori
tr-001	dqlabcust07	1	prod-01	Kotak Pensil DQLab	5	312500	High
tr-001	dqlabcust07	2	prod-03	Flash disk DQLab 32 GB	1	100000	Medium
tr-001	dqlabcust07	3	prod-09	Buku Planner Agenda DQLab	3	276000	Medium
tr-001	dqlabcust07	4	prod-04	Flashdisk DQLab 32 GB	3	120000	Medium
tr-002	dqlabcust01	1	prod-03	Gift Voucher DQLab 100rb	2	200000	Medium
tr-002	dqlabcust01	2	prod-10	Sticky Notes DQLab 500 sheets	4	220000	Medium
tr-002	dqlabcust01	3	prod-07	Tas Travel Organizer DQLab	1	48000	Low
tr-003	dqlabcust03	1	prod-02	Flashdisk DQLab 64 GB	2	110000	Medium
tr-004	dqlabcust03	1	prod-10	Sticky Notes DQLab 500 sheets	5	275000	Medium
tr-004	dqlabcust03	2	prod-04	Flashdisk DQLab 32 GB	4	160000	Medium
tr-005	dqlabcust05	1	prod-09	Buku Planner Agenda DQLab	3	276000	Medium
tr-005	dqlabcust05	2	prod-01	Kotak Pensil DQLab	1	62500	Low
tr-005	dqlabcust05	3	prod-04	Flashdisk DQLab 32 GB	2	80000	Low
tr-006	dqlabcust02	1	prod-05	Gift Voucher DQLab 250rb	4	1000000	High
tr-006	dqlabcust02	2	prod-08	Gantungan Kunci DQLab	2	31600	Low

Hasil BelajarKu

Aku bangga dengan diriku sendiri! Dari modul Fundamental SQL Using FUNCTION and GROUP BY yang aku pelajari, aku telah memahami dan mampu mempraktikkan:

1. Penggunaan fungsi skalar dan fungsi aggregate dalam operasi string dan numerik di SQL database
2. Penggunaan GROUP BY dalam mengelompokkan data dan memahami konsep menggabungkan GROUP BY dengan fungsi aggregate
3. Penggunaan CASE Statement untuk struktur pengambilan keputusan.
4. Mengerjakan mini project yang merupakan integrasi keseluruhan materi dan tentunya materi-materi pada modul-modul sebelumnya untuk menyelesaikan persoalan bisnis.

Dengan kemampuan ini, aku lebih pede untuk mengolah data dengan SQL. Keterampilan ini sendiri adalah 60% aktivitas awal yang akan dilakukan seorang analis. **Keep Fighting!**