Mengapa dibutuhkan Window Function ?

katakan kita memiliki data seperti tabel di bawah:

ID	Name	Score	Kelas
1	Udin	100	A
2	Putu Anom	80	B
3	Gede Gede	90	A
4	Hola	70	B

Data Hasil Ujian Siswa

Misalnya, kita ingin menghasilkan record, yang memiliki nilai tertinggi dari setiap kelas, seperti tabel dibawah ini:

ID	Name	Score	Kelas
1	Udin	100	A
2	Putu Anom	80	B

Ekspektasi

Bagaimana cara kita melakukannya ? 1 cara yang dapat digunakan adalah dengan query berikut:

```
SELECT data_siswa.name, data_siswa.score, data_siswa.kelas, data_siswa.angkatan
FROM data_siswa
INNER JOIN
(SELECT kelas, MAX(score) as score from data_siswa GROUP BY kelas) as sub_tabel
ON data_siswa.kelas = sub_tabel.kelas AND
  data_siswa.score = sub_tabel.score
```

Sedemikian rupa sehingga tabel yang dihasilkan adalah sebagai berikut:

Nah, untuk kasus ini, idenya adalah kita memerlukan subquery di tengah digunakan untuk menyaring skor tertinggi pada masing-masing kelas. Akan tetapi, hanya dengan subquery, kita tidak dapat mengetahui informasi nama dari peraih skor tertinggi di kelas tersebut, sehingga kita harus melakukan join ke query yang melakukan SELECT terhadap kelengkapan data siswa peraih skor tertinggi tersebut, yang sebenarnya hanya mengambil informasi nama user dan angkatan untuk melengkapi tabel.

Untuk melakukan analisa terhadap tabel tersebut, kita menggunakan 2 query dan harus melakukan minimal 1 kali join. Nah, Window Function, hadir untuk membuat proses pembuatan tabel di atas lebih efektif dan efisien.

Berikut adalah query yang menggunakan window function untuk mendapatkan hasil yang sama:

```
SELECT name, score, kelas
FROM (SELECT name, score, kelas, RANK() OVER (PARTITION BY kelas ORDER BY score DESC) as rank)
WHERE rank=1
```

Idenya disini adalah kita memberikan rank ke semua data pada tabel data siswa, sehingga tabel yang dihasilkan adalah sebagai berikut:

ID	Name	Score	Kelas	Ranking
1	Udin	100	A	1
2	Putu Anom	80	B	1
3	Gede Gede	90	A	2
4	Hola	70	B	2

Ranking yang dihasilkan Window Function

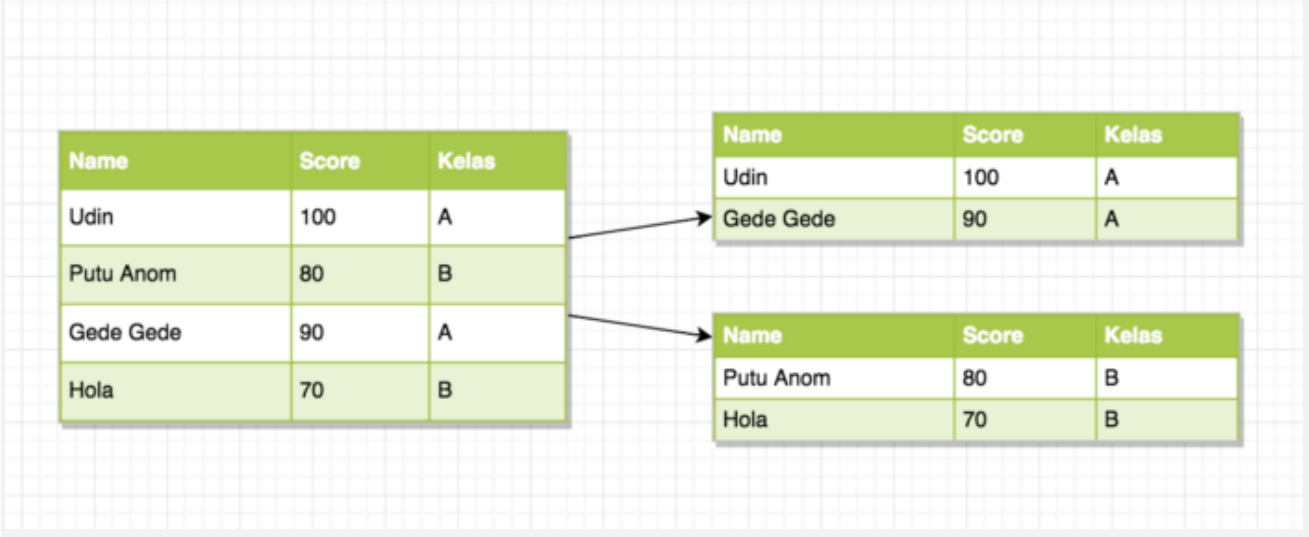
Setelah diberikan ranking menggunakan Window Function, kita melakukan filter terhadap data pada tabel di atas, kita ambil yang ranking 1 sehingga data yang kita hasilkan dengan kueri pertama adalah sama.

Tanpa melakukan join terhadap tabel yang sama, kita bisa mendapatkan data skor siswa tertinggi di masing-masing kelas.

Nah, jika kita lihat lebih dekat lagi untuk Window Function yang kita gunakan.

```
RANK() OVER (PARTITION BY kelas ORDER BY score DESC) as ranking
```

Kueri Window Function tersebut, jika di translasikan, berbentuk seperti gambar di bawah ini:



Partition By Kelas Order By Score DESC

Jika ditranslasi ke dalam bahasa indonesia, Query berikut:

```
RANK() OVER (PARTITION BY kelas ORDER BY score DESC) as ranking
```

Akan berbunyi seperti ini:

Aplikasikan fungsi RANK() ke setiap Partisi (Window Data) dimana setiap Partisi (Window Data) dibentuk dari setiap atribut kelas dan diurutkan menurut score yang paling besar.

Nah, window function terdiri dari 2 klausa, klausa pertama adalah Fungsi dan klausa kedua adalah konfigurasi window.

Klausa pertama pada query di bawah adalah RANK().

```
RANK() OVER (PARTITION BY kelas ORDER BY score DESC) as ranking
```

Dimana, RANK() itu akan digunakan untuk memproses Row Partisi yang telah terbagi tadi.

Klausa kedua pada query Window Function di atas adalah

```
PARTITION BY kelas ORDER BY score DESC
```

Dilakukan pemecahan 2 Kelas, akan tetapi masih terjadi dalam tabel yang sama. Sehingga, setelah fungsi RANK() digunakan, maka terbentuklah sebuah atribut baru (ranking) yang memiliki makna yang baru. Partition BY kelas ini menyebabkan, semua kelas yang sama dibuat menjadi 1 Window Data yang baru (Grouping). Bedanya, grouping ini terjadi di dalam Window Function yang kita lakukan, bukan terjadi seperti Query GROUP By, dimana semua data akan di Group sesuai atribut yang kita inginkan.

Secara umum sintax kondisional WHERE di dalam SELECT :

Select Field 1, Field 2, Field 3, Field dst. From [Nama Tabel] Where [Kondisional] Order by Field Index.

Jika semua field akan dimunculkan maka quey diatas bisa dengan cara seperti ini :

Select * From [Nama Tabel] Where [Kondisional] Orer by Field Index

Operator Matematis

Penggunaan WHERE pada sintax diatas dilanjut kondisional. Kondisional merupakan operator matematis yang membandingkan dua atau lebih parameter yang merupakan field tabel. Operator matematis terdiri dari = (sama dengan), .> (lebih besar), < (lebih kecil), >= (lebih besar sama dengan, <= (lebih kecil sama dengan) dan <> (tidak sama dengan).

Contoh penggunaan operator tersebut sebagai berikut :

WHERE Harga Rumah > Harga Tanah artinya data yang akan keluar dari query dengan kondisional ini adalah record yang harga rumahnya lebih besar dari harga tanah

WHERE Jumlah = 100 artinya record yang akan muncul pada output adalah record yang field **Jumlah** bernilai 100, nilai lainnya tidak muncul.

WHERE Jumlah >= 100 artinya record yang akan keluar dari hasil eksekusi query adalah record yang field **Jumlah** bernilai 100 atau di atas 100, seperti 105, 106 dan sebagainya.

WHERE Jenis Kelamin = 'Pria' artinya record yang akan keluar adalah record yang fieldnya bernilai 'pria'.

WHERE Jenis Kelamin <> 'Pria' artinya record yang keluar dari hasil query dengan kondisional tersebut adalah record yang nilainya tidak berisi **'Pria'**, misalnya yang akan keluar adalah record yang berisi Jenis Kelamin **'Wanita'**.