**Fundamental SQL Using FUNCTION and GROUP BY**

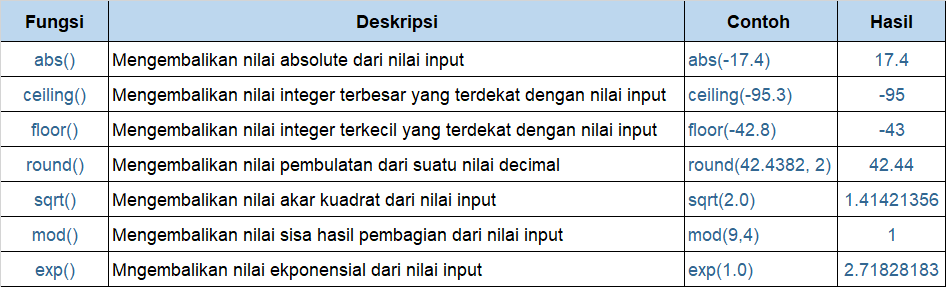
**Fungsi Scalar vs Fungsi Aggregate**

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-strings. SQL Function dapat dibagi ke dalam 2 kategori, yaitufungsi scalar dan fungsi aggregate.

Fungsi skalar dalam SQL digunakan untuk mengembalikan nilai tunggal (single value) dari suatu nilai input yang diberikan, sedangkan fungsi agregat dalam SQL digunakan untuk melakukan perhitungan pada sekelompok nilai dan kemudian mengembalikan nilai tunggal.

**Fungsi Skalar Matematika**

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.



**Fungsi Skalar Matematika - ABS()**

Fungsi **ABS( )**

Syntax:

**SELECT** ABS(*ColumnName*)    
**FROM** *TableName*;

**Fungsi Skalar Matematika - CEILING()**

Fungsi **CEILING()**

Syntax:

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

**Fungsi Skalar Matematika - FLOOR()**

Fungsi **FLOOR()**

Syntax:

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

**Fungsi Skalar Matematika - ROUND()**

Fungsi**ROUND()**

Syntax:

**SELECT** ROUND(ColumnName)    
**FROM** TableName;

**Fungsi Skalar Matematika - SQRT( )**

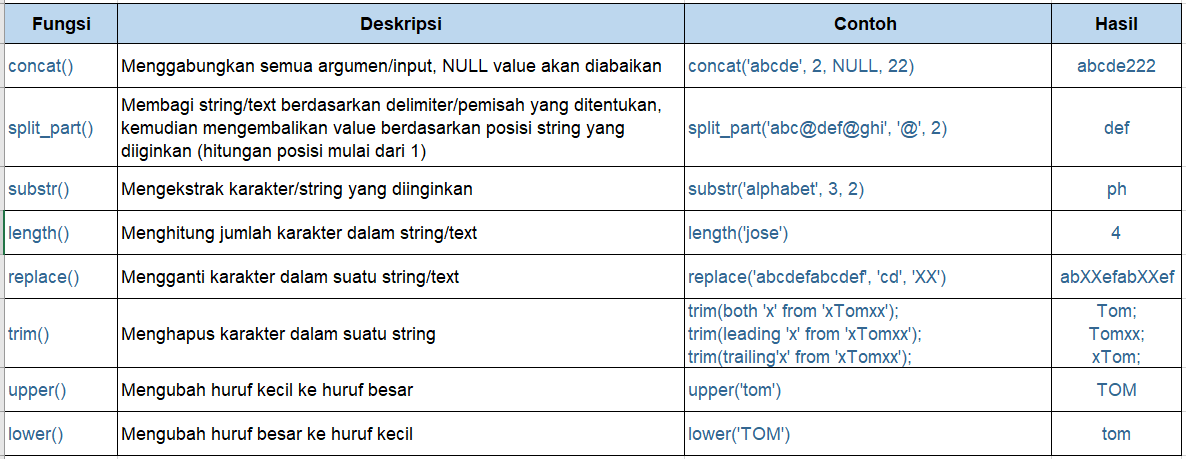
Fungsi **SQRT()**

Syntax:

**SELECT** SQRT(ColumnName)    
**FROM** TableName;

**Fungsi Text**

Sekarang kita akan membahas tentang fungsi skalar untuk text/string value. Fungsi ini digunakan jika kita ingin melakukan operasi pada text atau karakter di SQL, misalnya, mengubah huruf kecil ke huruf besar, menghitung jumlah karakter dari text, dll. Fungsi skalar text di SQL juga cukup banyak.



**Fungsi Text - CONCAT( )**

Fungsi **CONCAT()**

Syntax:

**SELECT** CONCAT(ColumnName1, ColumnName2, ColumnNameN)    
**FROM** TableName;

**Fungsi Text - SUBSTRING\_INDEX( )**

Fungsi Text **SUBSTRING\_INDEX()**

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.

**Fungsi Text - SUBSTR( )**

Fungsi Text **SUBSTR()**

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.

**Fungsi Text - LENGTH( )**

Fungsi Text **LENGTH()**

Syntax:

**SELECT** LENGTH(ColumnName)  
**FROM** TableName;

**Fungsi Text - REPLACE( )**

Fungsi Text **REPLACE()**

Syntax:

**SELECT** REPLACE(ColumnName, Character/String to be change, New String/Character)  
**FROM** TableName;

Keterangan:

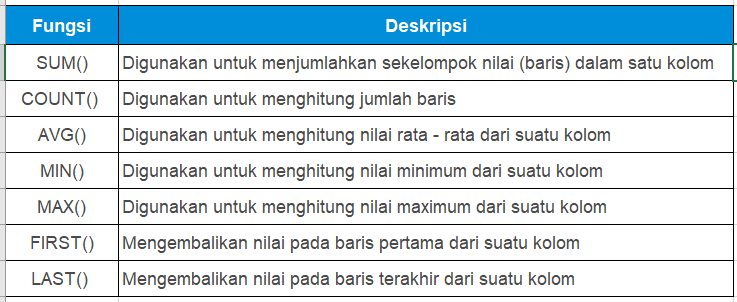
     ColumnName --> nama kolom yang akan diganti isi tiap record/barisnya berdasarkan string/karakter tertentu

     Character/String to be change --> string/karakter yang dimiliki untuk diganti

     New String/Character --> string/karakter baru pengganti string/karakter sebelumnya

**Fungsi Aggregate**

Fungsi aggregate ini digunakan untuk melakukan perhitungan pada sekelompok nilai.



**Fungsi Aggregate - SUM()**

Fungsi Aggregate **SUM()**

Syntax:

**SELECT** SUM(ColumnName)    
**FROM** TableName;

**Fungsi Aggregate - COUNT()**

Fungsi Aggregate **COUNT()**

Syntax:

**SELECT** COUNT(ColumnName)    
**FROM** TableName;

**Fungsi Aggregate - AVG( )**

Fungsi Aggregate **AVG()**

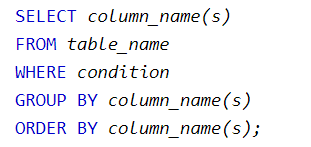
Syntax:

**SELECT** AVG(ColumnName)    
**FROM** TableName;

**Pengenalan 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, dan dengan menggunakan fungsi aggregate seperti (COUNT, MAX, MIN, SUM, AVG) kita bisa melakukan agregasi untuk untuk setiap group atau kelompok yang terbentuk.

Mari simak *syntax* yang diberikan:



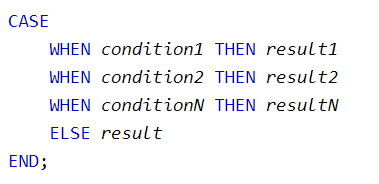
Hal penting yang perlu diperhatikan adalah:

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

Group by bisa dilakukan dengan single column ataupun multiple column.

* **Group by Single Column**, data dikelompokkan menggunakan kriteria dari satu kolom saja. Fungsi Group by **Single Column**memastikan data dapat dikelompokkan menggunakan kriteria dari satu kolom saja, misalnya mengelompokkan data berdasarkan provinsi saja.
* **Group by Multiple Column**, data dikelompokkan menggunakan kriteria dari dua kolom atau lebih.

**Penggunaan CASE … WHEN…**



CASE - Statement ditempatkan sesudah SELECT, berikut syntax-nya secara lengkap :

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

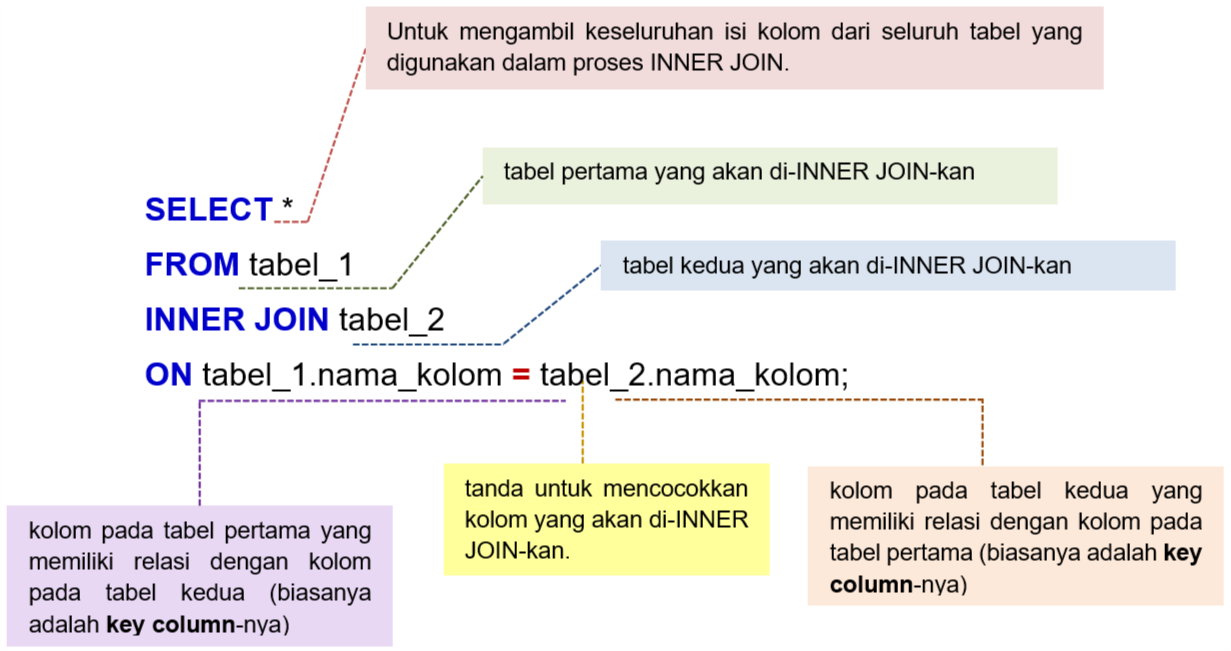
CASE-statement akan mengevaluasi kondisi yang sudah ditentukan, dimulai dari *condition1*, dan akan mengembalikan hasil (*result1*), jika *condition1* terpenuhi (TRUE). Jika tidak, maka *condition2* akan dievaluasi, dan akan mengembalikan *result2* jika*condition2* terpenuhi, dst. Apabila tidak ada kondisi yang terpenuhi, maka *result* pada bagian ELSE yang akan dikembalikan.

**Fundamental SQL Using INNER JOIN and UNION**

Syntax INNER JOIN

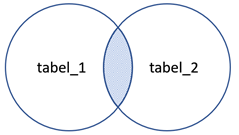
Tujuan JOIN adalah menggabungkan dua atau lebih tabel yang berbeda di dalam database. Nah, JOIN ini hanya bisa dilakukan apabila tabel - tabel tersebut memiliki key kolom yang sama. Di SQL, terdapat beberapa tipe JOIN yaitu : INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN, dan SELF JOIN. Setiap tipe join memiliki tujuan dan hasil query yang berbeda. Pada metode INNER JOIN, baris dari kedua atau lebih tabel akan dibandingkan untuk mengecek baris - baris mana saja yang cocok satu sama lain berdasarkan kondisi JOIN yang ditentukan.

Syntax INNER JOIN:



Penggunaan prefix pada tabel\_1.nama\_kolom dan tabel\_2.nama\_kolom ditujukan agar terdapat kejelasan (clarity) kolom pada tabel mana yang dijadikan acuan dalam proses INNER JOIN. Tentunya, penggunaan prefix ini ini sangat bermanfaat jika ingin menggabungkan beberapa tabel.

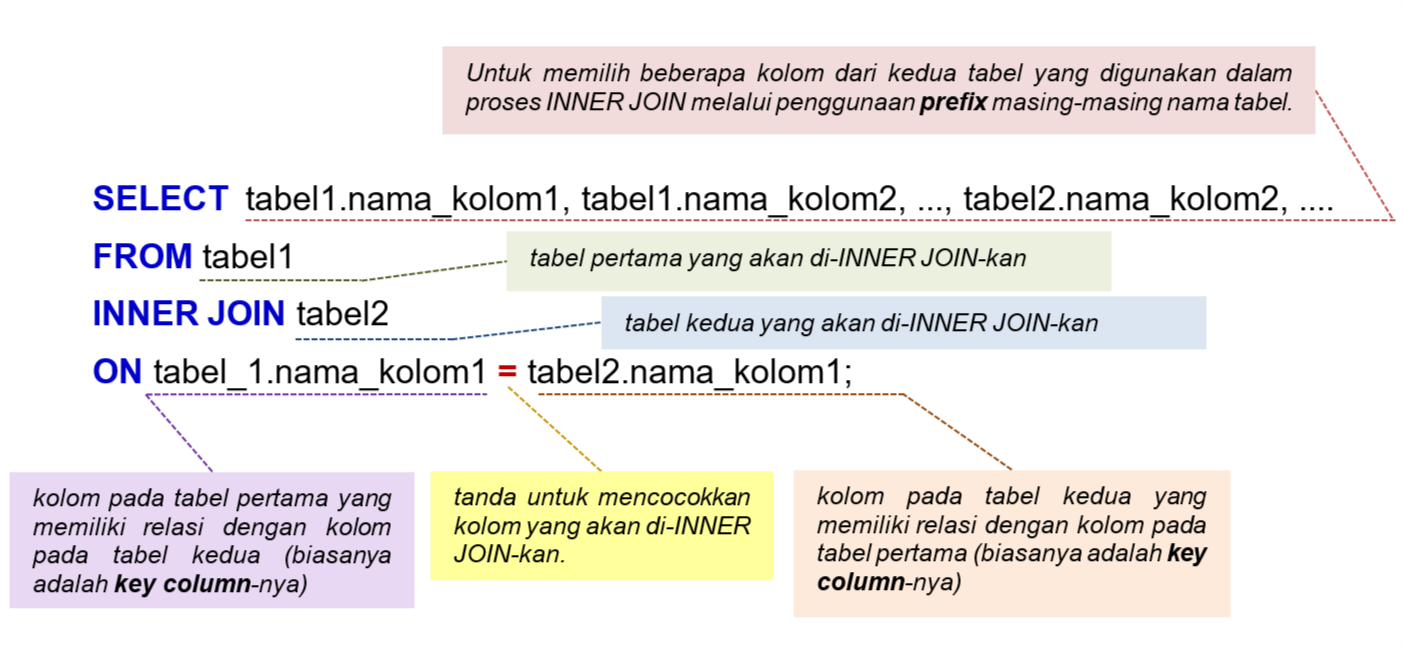
Jika masing-masing tabel tersebut dianalogikan sebagai dua himpunan maka proses INNER JOIN ekivalen dengan INTERSECTION (IRISAN) antara dua himpunan.



Proses INNER JOIN memiliki konsep penggabungan untuk records pada kedua tabel yang bernilai sama.

**Syntax INNER JOIN – Memilih Beberapa Kolom Untuk Ditampilkan**

Sementara itu, jika kita ingin menampilkan beberapa kolom saja dari tabel hasil penggabungan, maka syntax SQL dari INNER JOIN dinyatakan sebagai berikut ini.

****

**Penggabungan hasil SELECT secara “Vertikal”.**

**UNION** adalah operator SQL yang digunakan untuk menggabungkan hasil dari 2 atau lebih **SELECT** - statement secara “Vertikal”, dengan catatan,

Setiap hasil dari SELECT statement yang akan digabungkan (UNION) memiliki jumlah kolom yang sama.

Kolom tersebut juga harus memiliki tipe data yang sama, dan Kolom tersebut memiliki urutan posisi yang sama.

Berikut format syntax-nya:

