**Window Functions**

Bu bölümde SQL veri analizinde çok önemli ve çok güçlü bir kavram olan pencere fonksiyonunu öğreneceğiz. Analitik fonksiyonlar olarak da bilinirler. Bazı resmi belgeleri kullanarak "window işlevi" tanımıyla başlayalım.

Pencere işlevi, girdi değerlerinin bir SELECT ifadesinin sonuç kümesindeki bir veya daha fazla satırın "penceresinden" alındığı bir SQL işlevidir.

Pencere işlevleri, bir OVER yan tümcesinin varlığıyla diğer SQL işlevlerinden ayrılır. Bir işlevin OVER yan tümcesi varsa, bu bir pencere işlevidir. OVER yan tümcesi yoksa, sıradan bir toplama veya skaler fonksiyondur. window işlevleri, işlev ile OVER yan tümcesi arasında bir FILTER yan tümcesine de sahip olabilir.

Bir pencere işlevi, geçerli satırla bir şekilde ilişkili olan bir dizi tablo satırında bir hesaplama gerçekleştirir. Bu, bir toplama işleviyle yapılabilecek hesaplama türüyle karşılaştırılabilir. Ancak, normal toplama işlevlerinden farklı olarak, bir pencere işlevinin kullanılması, satırların tek bir çıktı satırında gruplandırılmasına neden olmaz - satırlar ayrı kimliklerini korur. Perde arkasında, pencere işlevi, sorgu sonucunun geçerli satırından daha fazlasına erişebilir.

Özetlemek gerekirse, pencere işlevleri bir dizi satırda çalışır ve temel sorgudan her satır için tek bir değer döndürür. Bu, "...satırların ayrı kimliklerini koruduğunu" açıklar. Yukarıdaki tanımdaki ifade. "Pencere" terimi, işlevin üzerinde çalıştığı satır kümesini tanımlar.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bir pencere işlevi kullandığımızda, OVER() yan tümcesini kullanarak pencereyi basitçe tanımlarız. OVER() yan tümcesi, pencere işlevlerini SQL'deki diğer işlevlerden ayırır.

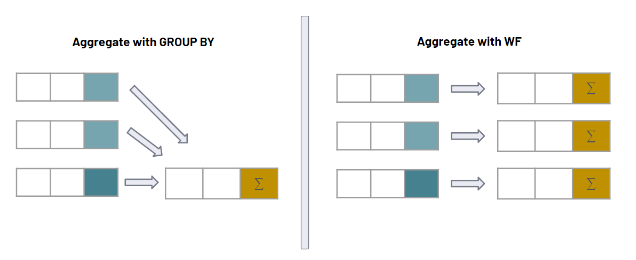
OVER() yan tümcesi, işlevselliğini genişletmek için aşağıdaki yan tümceleri alabilir:

PARTITION BY yan tümcesi: Satır grupları oluşturmak için pencere bölümlerini tanımlar

ORDER BY yan tümcesi: Bir bölüm içindeki satırları sıralar

ROW veya RANGE yan tümcesi: İşlevin kapsamını tanımlar

Aşağıdaki resimde toplama işlevleri ile pencere işlevleri arasındaki farkları görebilirsiniz:



Pencere işlevlerini üç kategoriye ayırabiliriz:

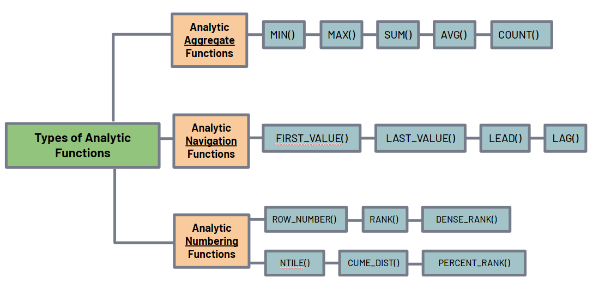
Toplam Pencere İşlevleri

Sıralama Penceresi İşlevleri

Değer Penceresi Fonksiyonları

Bu, sözdizimindeki "pencere\_fonksiyonu" teriminin aşağıdaki herhangi bir işlev olabileceği anlamına gelir.

Aşağıdaki resim, her kategoride kullanılan işlevleri gösterir:



**Aggregate Window Functions**

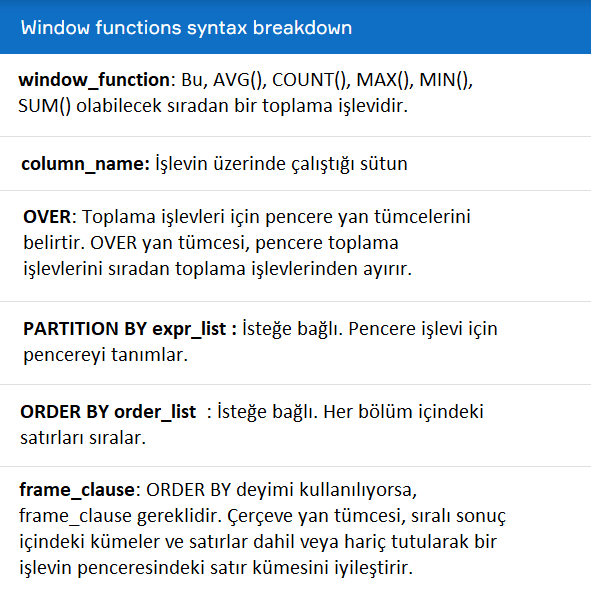
Toplama penceresi işlevi, normal bir toplama işlevine benzer. Ancak aralarındaki temel fark, toplama penceresi işlevinin döndürülen satır sayısını değiştirmemesidir.

Pencere fonksiyonunun genel sözdizimini yeniden yazalım:

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şimdi, aşağıdaki sözdizimi argümanlarını kırabiliriz:







Gördüğünüz gibi, mezuniyet sütunu için tüm tablo kayıtları döndürüldü. Ayrıca tüm tablodaki çalışan sayısı satır başına yazılmıştır. Bir pencere işleviyle oluşturulan yeni sütun, ana tabloda herhangi bir değişiklik yapılmadan sonuca eklendi. Ayrıca, birçok yinelenen satır var.

**💡İpuçları:**

Aşağıdaki sorgu gibi DISTINCT anahtar sözcüğünü kullanırsanız, yinelenen satırlardan kurtulursunuz.



tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Fark ettiğiniz gibi, sonuç için herhangi bir bölümleme, sıralama veya çerçeve koşulu kullanmadık. OVER anahtar kelimesinin yanındaki parantezler boştu ve yukarıdaki sonuçları aldık.

PARTITION BY'yi aşağıdaki gibi DISTINCT anahtar kelimesiyle kullanırsak:



tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Yukarıda gördüğünüz gibi sorgu mezuniyete göre çalışan sayısını döndürdü. Ve sonuç, DISTINCT anahtar sözcüğünü kullanarak yinelenen satırlardan kurtuldu.

**💡İşte mesele şu:**

PARTITION BY, bir pencere işlevinin üzerinde çalıştığı bölümleri belirtir. Pencere işlevi her bölüme ayrı ayrı uygulanır ve her bölüm için hesaplama yeniden başlar. PARTITION BY'yi dahil etmezsek, pencere işlevi tüm sütunda çalışır.

Parantez içinde yalnızca ORDER BY kullanırsak ne olur? Deneyelim ve görelim.



tablo içeren bir resim

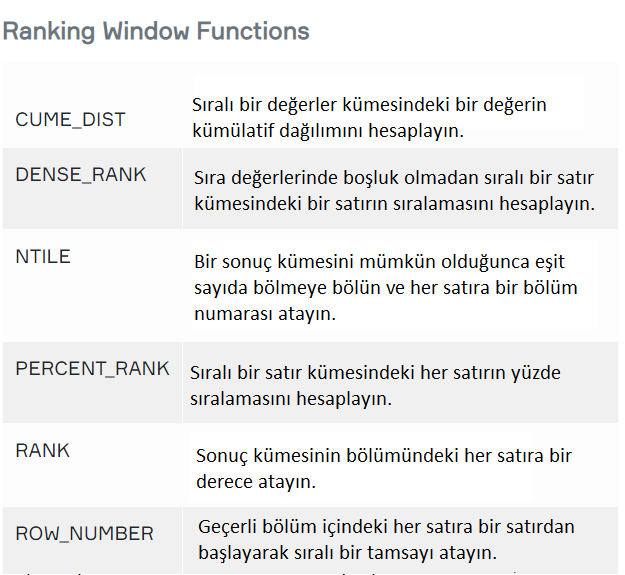
Açıklama otomatik olarak oluşturuldu

Yukarıdaki sonuçla ilgili olarak, eğer sipariş kuralını ASC veya DESC olarak belirtmezseniz, ORDER BY (ORDER BY) varsayılan olarak ASC'yi kabul eder. Dolayısıyla bu örnekte kiralama\_tarihi sütunu artan şekilde sıralanmıştır.

**☝ Not:** ORDER BY'yi dahil etmediğimizde toplamı aldığımızı, ORDER BY'yi dahil ettiğimizde toplam/kümülatif toplam elde ettiğimizi görüyorsunuz. Toplam pencere işlevleriyle ORDER BY kullanmanız gerekmez. Ancak, ORDER BY'ı toplu işlevlerle kullandığınızda ne olduğunu bilmek önemlidir.

**Ranking Window Functions**

Bu bölümde sıralama penceresi fonksiyonlarını öğreneceğiz. Sıralama penceresi işlevleri, bir bölümdeki her satır için bir sıralama değeri döndürür. Sıralama amacıyla kullanılan pencere işlevleri ve açıklamaları buradadır.



Hepsini kursumuzda ele almayacağız. Ancak, bunları kolayca kendi başınıza deneyebilirsiniz.

Size genel pencere işlevi sözdizimini hatırlatmama izin verin.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Çalışanları işe alım tarihlerine göre sıralayalım.





RANK() işlevi, hire\_date değeri aynıysa aynı sıra numarasını atar.

**☝ Not:** RANK() işlevi, sıralama kuralı tarafından oluşturulan listedeki değerlerin satır numaralarını atar. Aynı değerler için en küçük satır numaralarını atar.

Şimdi aynı senaryoyu DENSE\_RANK fonksiyonunu kullanarak uygulayalım.





☝ Not: DENSE\_RANK(), sıralama kuralı tarafından oluşturulan listedeki değerlerin sıra numaralarını döndürür. Aynı değerler için en küçük sıralı tamsayılarını atar.

Başka bir sıralama işleviyle devam edelim, ROW\_NUMBER().

ROW\_NUMBER(), her satıra sıralı bir tamsayı atar. Satır numarası ilk satır için 1 ile başlar.

PARTITION BY ile kullanılırsa, ROW\_NUMBER() bölüm içindeki her satıra sıralı bir tamsayı atar. Her bölümdeki ilk satır için satır numarası 1 ile başlar.

Her kıdem kategorisindeki çalışanlara işe alım tarihlerine göre sıra numarası verelim.



tablo içeren bir resim

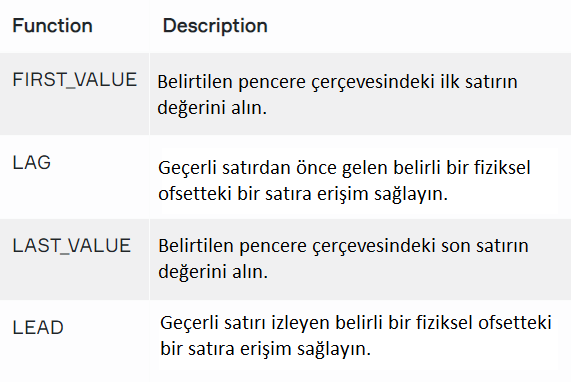
Açıklama otomatik olarak oluşturuldu

☝ Not: Sıralama penceresi işlevleriyle ORDER BY kullanmalıyız.

Pekala, bir sıralama penceresi işlevinin nasıl çalıştığına dair bir mantığa sahipsiniz. Diğer sıralama penceresi işlevlerini kendi başınıza kolayca uygulayabilirsiniz. Pencere işlevinin son kategorisiyle devam edelim: Value Window Functions.

**Value Window Functions**

Bu bölümde, pencere fonksiyonlarının son kategorisi olan değer penceresi fonksiyonlarını öğreneceğiz. Diğer satırlardan değerler eklemenize izin verirler. Değer Penceresi İşlevleri, kendi kendine katılma yapmak zorunda kalmadan önceki satıra erişir. Bazıları bu işlevleri 'ofset işlevleri' olarak da adlandırır. Aşağıdaki tablo değer penceresi fonksiyonlarını ve açıklamalarını göstermektedir.



Hepsini kursumuzda ele almayacağız. Ancak, bunları kolayca kendi başınıza deneyebilirsiniz.

Size genel pencere işlevi sözdizimini hatırlatmama izin verin.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

LAG() ve LEAD() işlevleriyle başlayalım. Bu işlevler, satırları önceki veya sonraki satırlarla karşılaştırmak için kullanışlıdır. LAG, önceki satırlardaki verileri döndürür ve LEAD, sonraki satırlardaki verileri döndürür.

Aşağıda, özellikle LAG ve LEAD işlevinin sözdizimi gösterilmektedir.



ofset: Opsiyonel. Bir değer elde etmek için geçerli satırdan geriye doğru satır sayısını belirtir. Verilmezse, varsayılan 1'dir. Bu durumda, önceki değerin değerini döndürür. Önceki satır yoksa (geçerli satır ilk satırdır), NULL değerini döndürür. Ofset değeri, negatif olmayan bir tam sayı olmalıdır.

default: Uzaklık, bölümün kapsamı dışında olduğunda döndürülecek değer. Varsayılan bir değer belirtilmezse, NULL döndürülür.



tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Aynı örneği LEAD() fonksiyonunu kullanarak yapalım.



tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Geçerli satırdan iki satır geriye erişmek istiyorsanız, offset argümanı 2'yi belirtmeniz gerekir. Aşağıdaki sorgu, geçerli satırdan iki satır geride olan değerleri görüntüler.



tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Örnekleri FIRST\_VALUE() AND LAST\_VALUE() ile yapalım



tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Gördüğünüz gibi, her satır için FIRST\_VALUE() işlevi, tam ad sütunundaki ilk değeri id'ye göre sıralayarak döndürür.

✍ Varsayılan pencere çerçevesi her satır için tüm satırları kapsadığından.



tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu

Yukarıdaki örnekte, her satır için LAST\_VALUE() işlevi, kimliğe göre sıralanmış tam ad sütunundaki son değeri döndürür.

✍ Pencere çerçevesini değiştiriyoruz. Çünkü varsayılan pencere çerçevesi her satır için tüm satırları kapsamadı.

**Window Frames**

Bu bölümde, pencere fonksiyonlarının çok önemli bir parçası olan pencere çerçevelerini öğreneceğiz. Varsayılan olarak, bölümdeki ilk satırdan geçerli satıra kadar tüm satırları kapsayacak şekilde her satır için bir pencere ayarlanır.

Ancak, bu varsayılandır ve pencere çerçevesi yan tümcesi kullanılarak ayarlanabilir. Pencere çerçevesi yan tümcesini kullanan bir pencere işlevi sorgusu aşağıdaki gibi görünür:

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Burada,

{columns}, sorgu için tablolardan alınacak sütunlardır,

{window\_func}, kullanmak istediğiniz pencere işlevidir,

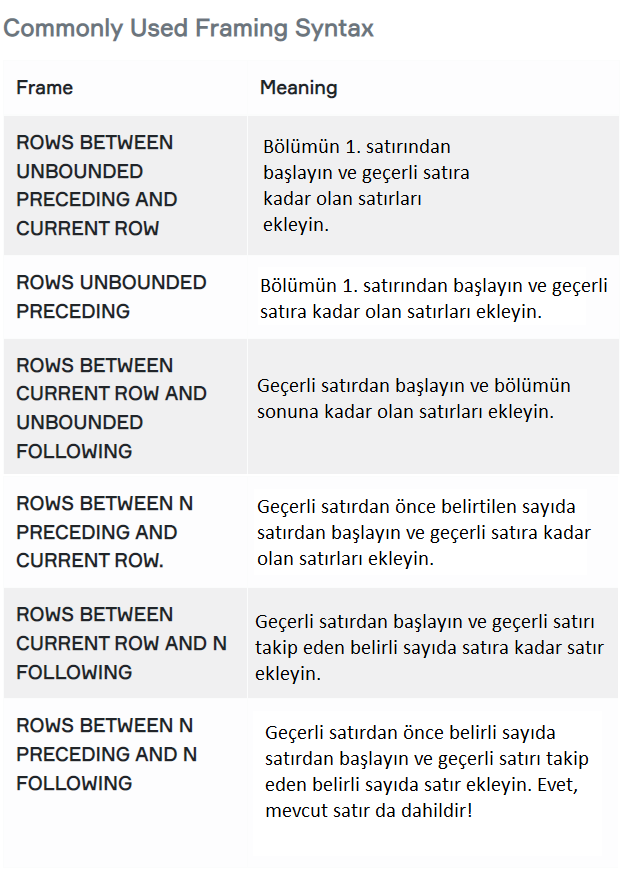
{partition\_key}, bölümlemek istediğiniz sütun veya sütunlardır (bundan sonra bahsedeceğiz),

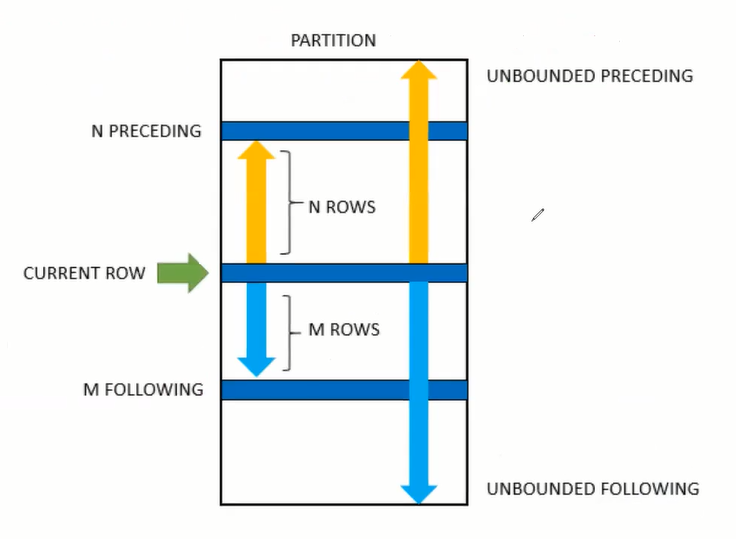
{order\_key}, sıralamak istediğiniz sütun veya sütunlardır,

{rangeorrows}, RANGE anahtar sözcüğü veya ROWS anahtar sözcüğüdür,

{frame\_start}, pencere çerçevesinin nereden başlatılacağını belirten bir anahtar kelimedir,

{frame\_end}, pencere çerçevesinin nerede biteceğini belirten bir anahtar kelimedir ve





tablo içeren bir resim

Açıklama otomatik olarak oluşturuldu