**DAX Temel Fonksiyonlar**

DAX Diline Giriş

**Data Analysis Expressions** ifadesinin kısaltmasından oluşan **DAX**, veri analizinde kullanılan önemli fonksiyonları içinde barındıran bir dildir.

DAX , veri modelimizde bulunan verilerden yeni bilgiler elde etmeye yarayan, bir veya daha fazla değeri hesaplayıp döndürmek üzere bir formülde ya da ifadede kullanılan işlevlerin, fonksiyonların bütünüdür. Bu sebepten Power BI ‘da DAX fonksiyonları oldukça sık kullanılır.

Ayrıca farklı BI (İş Zekası) ve görselleştirme araçlarında da DAX fonksiyonları kullanılmaktadır.

DAX yaygın bir modelleme dilidir.

DAX ile Excel Arasındaki Fark Nedir ?

Excel ’de kullanıcılar fonksiyonları kullanarak **hücre** içlerindeki veriler üzerinde istedikleri işlemleri yaparlar. Fakat DAX kodları bu şekilde çalışmaz.

DAX fonksiyonları **tablolara (tables)** dayanarak çalışırlar. Bu tablolar **kolonlardan (columns)** meydana gelir ve kolonlar belirli bir veri tipinden **(data type)** oluşur. (Integer 64 bit, date, string vb.)

DAX fonksiyonları ile bu veri tipine ait veriler üzerinden çeşitli yeni veriler ortaya çıkarılır ve raporlarda bu yeni veriler kullanılır.

DAX Veri Tipleri (Data Types)

**Integer (64 bit) :** (8 Bayt) Ondalık basamak içermeyen sayılardan oluşur. Tamsayılar pozitif veya negatif sayılar olabilir. -9,223,372,036,854,775,808 (-2^63) için 9,223,372,036,854,775,807 (2^63 -1) arasında değerler alabilir.

**Decimal (Floating point-64 bit)):** (8 Bayt) Ondalık basamak içeren sayılardan oluşur. **-1.79E +308 ila 2.23E -308** arasındaki negatif değerler – **Sıfır** – **2.23E -308 ila 1.79E + 308** arası pozitif değerlerini kapsar. Ayrıca anlamlı basamakların sayısı **15** ondalık basamakla sınırlıdır.

**Currency (Money) :** Para birimi veri türü, **dört ondalık** basamak sabit hassasiyete sahip - 922,337,203,685,477,5808 (-2^63) ila 922,337,203,685,477,5807 (2^63 -1) arasındaki değerlere izin verir.

**Date (Date Time) :** Tarih ve saatlerden oluşan veri tipleridir. Geçerli tarih aralığı **1 Ocak 1900** ’den başlayarak devam eder.

1

**True/False (Boolean) :** Sadece **True** (Doğru-1) ve **False** (Yanlış-0) değerlerini içeren veri tipidir. Mantıksal operatörlerle yapılan işlemlerin sonucu bu veri tipinde temsil edilir.

Diğer Veri Türleri

**String** : Unicode karakter dizisindeki karakterlerden oluşur. Metin biçiminde temsil edilen dizeler, sayılar veya tarihler olabilir.

**Blank** : DAX dilindeki Blank değerinin karşılığı SQL ‘ deki NULL ‘ dır. Blank değerini kullanarak boş bir değer atayabilir bu değer ile ilgili ileride ele alacağımız ISBLANK fonksiyonunu kullanarak çeşitli işlemler yapabilirsiniz.

DAX Fonksiyonları

**Toplayıcı (Aggregate) Fonksiyonları**

▪ SUM

▪ AVERAGE (AVERAGEA, AVERAGEX)

▪ MIN

▪ MAX

▪ SUMX (ve diğer X işlevleri)

**Sayma (Count) Fonksiyonları**

▪ COUNT

▪ COUNTA, COUNTAX

▪ COUNTBLANK

▪ COUNTROWS

▪ DISTINCTCOUNT

▪ PRODUCT

**Mantıksal (Logical) Fonksiyonlar**

▪ AND

▪ OR

▪ NOT

▪ IF

▪ IFERROR

▪ SWITCH

▪ IN

**Bilgi (Information) Fonksiyonları**

▪ ISBLANK

▪ ISNONTEXT

▪ ISERROR, ISEMPTY

▪ CONTAINS, CONTAINSROW

▪ HASONEFILTER, HASONEVALUE

▪ ISODD, ISNUMBER, ISTEXT

2

▪ ISFILTERED

▪ SELECTEDMEASURE

**Metin (Text) Fonksiyonları**

▪ CONCATENTATE

▪ SUBSTITUTE

▪ REPLACE

▪ FIXED

▪ FORMAT

▪ EXACT

▪ TRIM, REPT, VALUE

▪ FIND, SEARCH

▪ LEFT, RIGHT

▪ LOWER, UPPER

**Tarih (Date) Fonksiyonları**

▪ DATE

▪ HOUR

▪ NOW

▪ EOMONTH

▪ WEEKDAY

▪ TODAY

▪ DATEADD

▪ ENDOFMONTH, ENFOFQUARTER, ENDOFYEAR

▪ DATESMTD, DATESQTD, DATESYTD

▪ DATESBETWEEN, DATESINPERIOD

▪ TOTALMTD, TOTALYTD, TOTALQTD

▪ PREVIOUSMONTH, PREVIOUSYEAR, PREVIOUSMONTH

▪ SAMEPERIODLASTYEAR

▪ DATESINPERIOD

▪ CALENDAR, CALENDARAUTO

**Power BI’ da DAX İle Calculated Column (Hesaplanmış Kolonlar) ve Measures (Ölçüler) Oluşturma**

▪ Calculated Columns,

▪ Measures,

▪ Power BI ‘ da Oluşturulan Örnek DAX Kodları

**DAX İfadelerinde Değişken Kullanımı**

▪ VAR Kullanımı

**DAX Tablo Fonksiyonları**

▪ FILTER, ALL, ALLEXCEPT, ALLSELECTED

▪ Birden fazla filtre kullanımı

▪ VALUES, DISTINCT, ALLNOBLANKROW, ISEMPTY

3

**DAX ’da Tablolar Arası İlişkiler**

▪ İlişki fonksiyonları, RELATED, RELATEDTABLE,

USERELATIONSHIP

▪ Calculated Tables

**Evaluation Context Kavramı**

▪ Filter Context

▪ Row Context

▪ Context hataları

▪ Context ve tablolar arası ilişkilerin etkileşimi

▪ CALCULATE ve çoklu filtre kullanımı

▪ Filter ve CrossFilter

▪ VAR ve evaluation context etkileşimi

▪ Context Transition

DAX Toplayıcı (Aggregate) Fonksiyonları

**Power BI** ’da en çok kullanılan fonksiyonlardan biri **DAX Toplayıcı (Aggregate) Fonksiyonları’** dır.

Genel olarak **Power BI** raporları hazırlanırken, hazırlanan raporların içeriğine, detayına ve veri türlerine göre **Power BI ’da DAX Fonksiyonları** ‘nın çok karmaşık kullanımları söz konusu olabilir.

DAX Toplayıcı (Aggregate) Fonksiyonları (SUM, AVERAGE, MIN, MAX…) **bir parametre** ile çalışan sütun bazlı fonksiyonlardır. Üzerinde işlem yaptıkları **sütunu** parametre olarak alırlar. **Tek bir sütun üzerinde çalışırlar ve satırı görmezler.**

Fakat **X** ile biten özel toplama işlevleri (**SUMX gibi**) ile birden çok sütunda çalışılabilir. Bu işlevler tablonun tamamında yinelenir (iterator) ve ifadeyi her bir satır için değerlendirir.

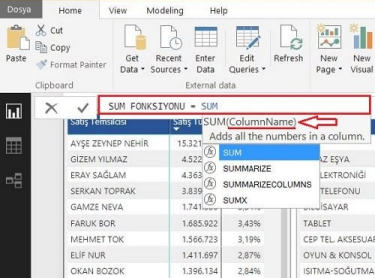
Ayrıca DAX Toplayıcı (Aggregate) Fonksiyonları sadece **Date** ve **Boolean** harici diğer **Sayısal Veri Tiplerinde (Numeric Types)** çalışırlar.

**SUM Fonksiyonu**

[code]Total Amount = SUM (ColumnName) [/code]

**Bu bölümde “**Total Amount**” olarak isimlendirdiğimiz** hesaplanmış bir kolon-sütun (column) **ya da** ölçü (measure) **olabilir.**

4



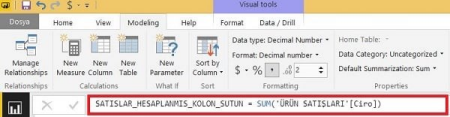
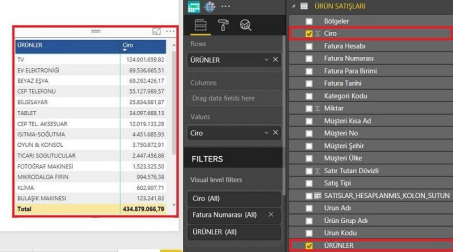
Toplam\_Satis = SUM (‘SIPARISLER'[Birim\_Fiyat] \* ‘SIPARISLER'[Miktar]) //Hata-Error

Şeklinde kullanımda hata verecektir. Hatanın sebebi ise daha önce de belirttiğim gibi **SUM fonksiyonu** satırları parametre olarak almazlar, sadece **tek bir sütunu parametre olarak alabilirler.**

Aşağıdaki örnekte bazı ürünlerin satış rakamlarının yer aldığı veri kümesinden bir görsel paylaştım. Genel olarak bu veri kümesinde bulunan “**Ciro**” alanı bizim satış rakamlarımızı Ürün Grubu bazında kırarak göstermektedir.

Fakat **SUM** fonksiyonunun nasıl kullanıldığını ve aldığı parametreyi veri kümesinde bulunan satırlarda nasıl göstereceği konusunda örnek olması açısından **SUM fonksiyonu** ile bir **Hesaplanmış Kolonun-Sütunun (Column)** oluşturacağız.

5



Aşağıdaki ekran görüntüsünde **SUM fonksiyonu** ile oluşturulan **Hesaplanmış Kolonun Sütunun (Column)** her bir satırda, aynı değeri yani **tüm satışların toplamını** tekrar ederek getirdiğini görebilirsiniz.

SUM fonksiyonunun neden bu şekilde aynı sütunda tekrarladığını daha iyi kavramak için SUM Fonksiyonunun hemen ardından, **SUMX Fonksiyonunun** kullanımını incelemekte fayda var.

6



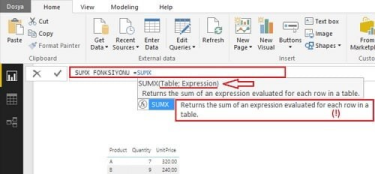
**SUMX Fonksiyonu**

SUMX Fonksiyonu yinelenen yani **iterator** bir fonksiyondur ve **iki parametre** alır. SUMX\_Fonksiyonu = SUMX(TABLE ; EXPRESSION)

Yinelenecek table (tablo), doğrudan bir tablonun adı olabilir veya geriye tablo döndüren bir tablo fonksiyonu da olabilir. Burada önemli olan satır satır SUMX fonksiyonuna yazdığımız komutu çalıştıracak bir tablonun olması.

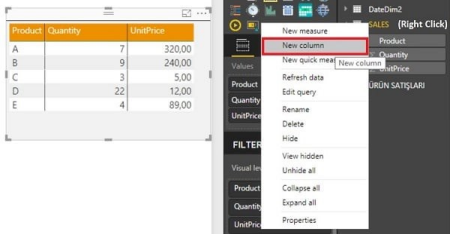
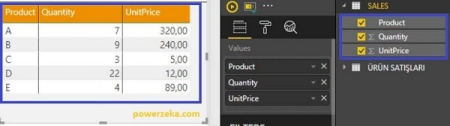
Yineleyen fonksiyonlar parametre olarak verilen tabloda gördüğü tüm satırlara ilk satırdan başlayarak, fonksiyonun ikinci parametresinde belirtilen ifadeyi (Expression) tek tek uygular. Bulduğu sonucu yineleyen fonksiyonu oluşturduğumuz **hesaplanmış bir kolona-sütuna (column)** ya da **ölçüye (measure)** yazar.

7



**SUMX** Fonksiyonunun nasıl kullanıldığını basitçe anlatabilmek amacıyla aşağıdaki örnek veri kümesini **Power BI Desktop** ‘ın **Enter Data** özelliği kullanarak manuel olarak oluşturabilirsiniz. Veri kümesini incelediğimizde Ürünlerin (Product) ve bu ürünlerin satış miktarlarının (Quantity) ve ürünlerin birim fiyatlarının (UnitPrice) olduğunu görüyoruz.

**SUMX** Fonksiyonunu kullanarak her üründen ne kadar satış yapıldığını Hesaplanmış Kolon/Sütun (New Column) ile göstereceğiz.

8



**Hesaplanmış Kolon/Sütun (New Column) Oluşturma**

****SUMX Fonksiyonu-CALCULATE ( ) İle Kullanımı – DAX Toplayıcı (Aggregate) Fonksiyonları

Yazdığımız kodda **CALCULATE** fonksiyonunu kullanılmaktadır. Bunun sebebi SUMX ( ) Fonksiyonu içerisindeki ifadede matematiksel bir işlem/**hesaplama** yaptığımız için CALCULATE fonksiyonunu kullanmamızın gerekliliğindendir.



CALCULATE fonksiyonu genel olarak bir DAX fonksiyonudur. Bir DAX Formülü içerisinde filtreleme istiyorsak **FILTER** fonksiyonunu veya bir matematiksel hesap yapıyorsak CALCULATE( ) fonksiyonunu kullanıyoruz. Eğer CALCULATE fonksiyonu kullanıyorsak veri kümesinde gizli bir filtre işlemi de uygulanıyor demektir.

Şöyle ki; Her satırda **SUMX** fonksiyonuna yazılan işlem uygulandı ve daha sonra **CALCULATE** fonksiyonu Product alanına göre filtreledi ve tüm değerleri ürün bazında kırarak getirmiş oldu.

Bunun dışında yukarıdaki örnekte SUMX Fonksiyonunu CALCULATE fonksiyonu olmadan kullansaydık nasıl bir sonuç dönerdi?

Yukarıda anlatılanlar ışığında SUM Fonksiyonuna benzer bir şekilde ürün bazında kırılım (Product) gerçekleşmeden her satırda tüm işlemlerin toplamı olan **5035,00** rakamını getirdi!

9

**TotalSales = SUMX(SALES;SALES[Quantity]\*SALES[UnitPrice]) **

**AVERAGE Fonksiyonu**

Daha önce Excel gibi programları kullananlar genel olarak bilirler ki **AVERAGE** fonksiyonu belirtilen sütunu bir parametre olarak alır ve o sütundaki değerlerin ortalamasını bulur.

Eğer sütun veri tipi olarak metin içeriyorsa, hiçbir işlem yapılmaz ve boş döner. Sütun mantıksal (logical) değerler veya **boş hücreler** içeriyorsa, bu değerler göz ardı edilerek ortalama hesaplanır. Doğal olarak **sıfır** değerine sahip alanlar hesaplamaya dahil edilir.

Bir sayı kümesine değerlendiren bir ifadenin ortalamasını bulmak istiyorsak, bunun yerine **AVERAGEX** fonksiyonunu kullanmamız gerekir.

QuantityAverage = AVERAGE(SALES[Quantity])

**AVERAGEX () Fonksiyonu**

Averaj Alma – Measure

**OlçümAVGX** =

AVERAGEX(InternetSales;InternetSales[OrderQuantity]\*InternetSales[UnitPrice])

**AVERAGEX** işlevi, bir tablonun her satırı için ifadeleri değerlendirmenizi ve ardından sonuç değer kümesini alıp aritmetik ortalamasını hesaplamanızı sağlar. Bu nedenle işlev ilk bağımsız değişkeni olarak bir tablo ve ikinci bağımsız değişken olarak bir ifadesi alır.

**AVERAGEX** işlevinin diğer kuralları AVERAGE ile aynıdır. Sayısal olmayan veya null hücreler ekleyemezsiniz. Hem tablo hem de ifade bağımsız değişkenleri gereklidir.

10

Toplanacak satır olmadığında işlev boş değer döndürür. Satırlar olmasına rağmen hiçbiri belirtilen ölçütlere uygun değilse işlev 0 döndürür.



**AVERAGEA Fonksiyonu**

Sütundaki değerlerin ortalamasını (aritmetik ortalama) döndürür. Metinleri ve sayısal olmayan değerleri işler.

**AVERAGEA**(<column>)

**AVERAGEA** işlevi bir sütunu alır ve içindeki sayıların ortalamasını hesaplar ancak aynı zamanda sayısal olmayan veri türlerini de aşağıdaki kurallara göre işler:

• TRUE olarak değerlendirilen değerler 1 sayılır.

• FALSE olarak değerlendirilen değerler 0 (sıfır) sayılır.

• Sayısal olmayan metin içeren değerler 0 (sıfır) sayılır.

• Boş metin ("") 0 (sıfır) sayılır.

Hesaplamaya mantıksal değerleri ve başvurulan sayıların metin gösterimlerini dahil etmek istemiyorsanız AVERAGE işlevini kullanın

11



Aşağıdaki örnek, aşağıdaki tabloda verilen başvurulan sütunun içindeki boş olmayan hücrelerin ortalamasını döndürür. AVERAGE işlevini kullandığınızda ortalama 21/2 olurken, AVERAGEA işlevinde sonuç 22/5 olur.



**MIN Fonksiyonu**

****MIN\_FUNCTION = CALCULATE(MIN(ColumnNameOrScalar1;[Scalar2])

**MIN Fonksiyonu**, bir kolondaki (sütundaki) veya iki skaler ifade arasında bulunan en küçük sayısal değeri sonuç olarak döndürür.

Mantıksal (Lojik) değerleri ve metni yok sayar. MIN ( ) Fonksiyonu, **Sayısal Veri Tiplerinde (Numeric Types)** ve boşluklarda (blanks) çalışabilirler. Sayısal bir değer yok ise MIN Fonksiyonu boşlukları döndürür.

12



Ölçü birimlerinde veya sayısal olarak verilerin içerisinden belli bir minimum değeri bir kartta ayrıca göstermek istediğimizde **MIN Fonksiyonu** ile bir **ölçü (measure)** oluşturarak gösterebiliriz.

**MIN Fonksiyonunun Ölçü (Measure) İle Kullanılması**

MIN\_FUNCTION\_Measure = CALCULATE(MIN(SALES[UnitPrice])) **MAX Fonksiyonu**

MAX\_FUNCTION = CALCULATE(MAX(ColumnNameOrScalar1;[Scalar2])

Sayısal değerler içeren tek bir sütun değerlendirilirken, sütunda sayı yok ise, **MAX Fonksiyonu** bir boşluk (Blank) döndürür.

Sayı olmayan değerler için **MAXA Fonksiyonu** kullanılır.

**MAX Fonksiyonu** ile iki ifadeyi karşılaştırırken boşluk değerler (blank) sıfır olarak değerlendirilir. Örnek olarak **MAX (3, Blank() ) *3*** değerini döndürürken, **MAX (-2, Blank() ) *sıfır*** değerini döndürür. Eğer MAX) Fonksiyonuna yazılan değerlerin her ikisi de boşluk (blank) ise fonksiyon boşluk (blank) değeri döndürür.

**MINX** - Minimum bulma - Measure

OlçümMINX = MINX(InternetSales;InternetSales[OrderQuantity]\*InternetSales[UnitPrice])

**DISTINCTCOUNT**

Sütundaki farklı değerleri sayar.

13



Bu işlevin izin verilen tek bağımsız değişkeni bir sütundur. Herhangi bir türde veri içeren sütunlar kullanabilirsiniz. İşlev sayılacak satır bulamadığında BLANK döndürür, aksi takdirde farklı değerlerin sayısını döndürür.

ÖlçümDistinct = **DISTINCTCOUNT**('InternetSales'[SalesAmount])

DISTINCTCOUNT işlevi BLANK değerini içerir. BLANK değerini atlamak için **DISTINCTCOUNTNOBLANK** işlevini kullanın.

**DISTINCTCOUNTNOBLANK**

DISTINCTCOUNT işlevinden farklı olarak DISTINCTCOUNTNOBLANK işlevi BLANK değerini dahil etmez.

**COUNT ya da COUNTROWS**

**COUNT** işlevi, sütundaki boş olmayan değer içeren hücrelerin sayısını belirler.

Bu işlevin izin verilen tek bağımsız değişkeni bir sütundur. COUNT işlevi, aşağıdaki değer türlerini içeren satırları sayar:

• Sayılar

• Tarihler

• Dizeler

• İşlev sayacak satır bulamadığında boş değer döndürür.

• Boş değerler atlanır. TRUE/FALSE değerleri desteklenmez.

**COUNTA**

TRUE/FALSE değerlerinden oluşan bir sütunu değerlendirmek isterseniz COUNTA işlevini kullanın.

**COUNTAX**

COUNTA işlevinde olduğu gibi COUNTAX işlevi de diğer ifadeler dahil olmak üzere herhangi bir türde bilgi içeren hücreleri sayar. Örneğin bir sütunda boş dize sonucunu veren bir ifade varsa, COUNTAX işlevi bu sonucu boş olmayan değer olarak değerlendirir. COUNTAX işlevi genellikle boş hücreleri saymaz ancak bu durumda içinde bir formül bulunduğu için hücre sayıma dahil edilir.

Aşağıdaki örnek, Reseller tablosunun [Status] = **Active** ile filtrelenmesi sonucu elde edilen tabloyu kullanarak Phone sütunundaki boş olmayan satırları sayar.

= COUNTAX(FILTER('Reseller',[Status]="Active"),[Phone])

**COUNTX**

Boş olmayan değer veya bir tablo için boş olmayan değer olarak değerlendirilen bir ifadeyi içeren satırları sayar.

14



**COUNTX** işlevi iki bağımsız değişken alır. İlk bağımsız değişken her zaman bir tablo ya da bir tablo döndüren bir ifade olmalıdır. İkinci bağımsız değişken, **COUNTX** tarafından aranan sütun veya ifadedir.

**COUNTX** işlevi yalnızca değerleri, tarihleri veya dizeleri sayar. İşlev sayacak satır bulamadığında boş değer döndürür.

Mantıksal değerleri saymak için **COUNTAX** işlevini kullanın.

**COUNTROWS**

Tablodaki satır sayısını verir, tekrarlanmış veya null olan kayıtları da sayar **Count** Column ile kullanılırsa null olmayan kayıtların adetlerini de getirir **Count** distinct ile kullanılırsa kayıtların sadece birini sayar ve yine null değerler sayılmaz

**COUNTBLANK**

Sütundaki boş hücreleri sayar.

Bir tamsayı döndürür.

Bu işlevin izin verilen tek bağımsız değişkeni bir sütundur. Herhangi bir türde veri içeren sütunları kullanabilirsiniz ancak yalnızca boş hücreler sayılır. Sıfır (0), sayısal bir değer olarak kabul edildiğinden sıfır değerine sahip olan hücreler sayılmaz.

**PRODUCT**

Sütundaki sayıların çarpımını döndürür.

= PRODUCT( Annuity[AdjustedRates] )

**PRODUCTX**

Bir tablodaki her satır için değerlendirilen ifadenin çarpımını döndürür.

**PRODUCTX**(<table>, <expression>)

Bir sütundaki sayıların çarpımını döndürmek için PRODUCT kullanın.

**PRODUCTX** işlevi ilk bağımsız değişken olarak bir tablo veya tablo dönüştüren bir ifade alır. İkinci bağımsız değişken, çarpımını hesaplamak istediğiniz sayıları içeren bir sütun veya sütun sonucu veren bir ifadedir.

Yalnızca sütundaki sayılar sayılır. Boşluklar, mantıksal değerler ve metin yok sayılır.

15



Aşağıdaki formül, bir yatırımın gelecekteki değerini hesaplar:

= [PresentValue] \* PRODUCTX( AnnuityPeriods, 1+[FixedInterestRate] )

Tarih Fonksiyonları (Time Intelligence)

**DATE**

SutunDate = DATE(YEAR(Customer[BirthDate].[Date]);

MONTH(Customer[BirthDate].[Date]); DAY(Customer[BirthDate].[Date])) DATE (Yıl, Ay, Gün)

YIL:

• Yıl bilgisini verecek olan rakamlardan oluşmalıdır.

• 1 ile 4 basamak olabilir.

• 1 Mart 1900 tarihinden itibaren başlar. Başka bir deyişle yıl bilgisine sıfır verirseniz yılınız 1900 olacaktır.

• Negatif bir sayı girildiginde **#VALUE!** hatası alırsınız.

AY:

• AY bilgisini verecek olan rakamlardan oluşmalıdır.

• Genelde “1” ile “12” arasındaki rakamlar kullanılarak ilgili aylar temsil edilir. 1 – Ocak, 2 – Şubat gibi.

• 12’den büyük bir sayı verildiğinde ise sayınızın 12’ye olan mod’u AY olacaktır. Örneğin 18 girildiğinde YIL bilgisi 1 artacak ve AY bilgisi 6 olacaktır. • Sıfır’dan küçük bir rakam girildiğinde de mantık aynı şekildedir. -5 girdiğinizde YIL bilgisi 1 azaltılacak ve AY bilgisi 7 olacaktır.

GÜN:

• GÜN bilgisini verecek olan rakamlardan oluşmalıdır.

• Genelde “1” ile “31” arasındaki rakamlar kullanılarak ilgili günler belirlenir. • Olması gerekenden büyük bir sayı girildiğinde mantık AY ile aynıdır. İlgili sayının mod’u GÜN bilgisine eklenir.

• Negatif bir sayı girildiğinde de mantık değişmemistir, Gün değerine -10 verdiğinizde Ay bilgisi 1 azalır ve bir önceki ayın toplam gün sayısından 10 düşülerek sonuca ulaşılır. • Gün sayısının ondalık kısmı olursa en yakın tam sayıya yuvarlanarak sonuç üretilir.

**Örnekler**

=**DATE** (2017, 1, 21) 2017 yılı, Ocak Ayının 21. günüdür.

=**DATE** (05, 1, 12) 1905 yılı, Ocak Ayının 21. günüdür.

=**DATE** (1899, 1, 21) 3799 yılı, Ocak Ayının 21. günüdür.

16



=**DATE** (2017, 13, 21) 2018 yılı, Ocak Ayının 21. günüdür.

=**DATE** (2017, -2, 21) 2016 yılı, Ekim Ayının 21. günüdür.

=**DATE** (2017, 1, 32) 2017 yılı, Şubat Ayının 1. günüdür.

=**DATE** (2017, 1, -15) 2016 yılı, Aralık Ayının 16. günüdür.

\*\*\*=**DATE** (2017, 1, 0) 2013 versiyon excellerde 2016 yılının son gününü vermesine rağmen Power bi ve 2016 versiyon excellerde hata vermektedir.

**DATEDIFF**

**Sözdizimi**

DATEDIFF (Başlangıç Tarihi, Bitiş Tarihi, Süre)

İki tarih arasındaki farkı, farklı zaman cinsleri değerinden bulan fonksiyondur. **Parametreler**

Başlangıç Tarihi:

Farkları alınmak istenen tarihlerin küçük olanıdır. Saat, dakika gibi zaman değerlerini içerebilir. Bitiş tarihi değerinden büyük olamaz.

Bitiş Tarihi:

Farkları alınmak istenen tarihlerin büyük olanıdır. Saat, dakika gibi zaman değerlerini içerebilir. Başklangıç tarihi değerinden küçük olamaz.

Süre:

Sonucun hesaplanacağı zaman birimi seçimidir. Sonuç tam sayı bir değer olacağından yapılan seçim gayet önemlidir. Örnekleri incelediğinizde daha net anlayacağınızı düşünüyorum. Seçilebilecek paramaetreler aşağıdaki gibidir;

o SECOND (Saniye)

o MINUTE (Dakika)

o HOUR (Saat)

o DAY (Gün)

o WEEK (Hafta)

o MONTH (Ay)

o QUARTER (Çeyrek)

o YEAR (Yıl)

SutunDıffDate = **DATEDIFF**(InternetSales[OrderDate];InternetSales[ShipDate];WEEK)

17



**DATEADD**

Geçerli bağlamda belirtilen sayıda aralık ile zamanda ileri veya geri kaydırılmış olan tarih sütununu içeren bir tablo döndürür.

**Tarihler** bağımsız değişkeni aşağıdakilerden biri olabilir:

• Tarih/saat sütunu başvurusu,

• Tarih/saat değerlerini içeren tek bir sütun döndüren tablo ifadesi,

• Tarih/saat değerlerini içeren tek bir sütun tanımlayan Boole ifadesi.

**number\_of\_intervals** için belirtilen sayı pozitif bir sayıysa **tarihler** içindeki tarihler zamanda ileri kaydırılır, sayı negatifse **tarihler** içindeki tarihler zamanda geri kaydırılır.

**aralık** parametresi, dize kümesi değil sabit listesidir ve bu nedenle değerlerin tırnak içine alınmaması gerekir. Ayrıca year, quarter, month, day değerleri tam olarak yazılmalıdır.

Aşağıdaki formül, geçerli bağlamdaki tarihlerden bir yıl önceki tarihleri hesaplar. = DATEADD(DateTime[DateKey],-1,year)

**NOW**

Bugün Fonksiyonu

**Parametreler**

Herhangi bir parametre kullanilmasina gerek yoktur, guncel tarih degeri uzerinden yapilan islemlerde kullanisli bir fonksiyondur.Güncel gün ve saatini tam olarak verir.

**Döndürülen değer**

Tarih (datetime).

Power BI hizmetinde, NOW işlevinin sonucu her zaman **UTC** saat dilimindendir.

TODAY işlevi de aynı tarihi döndürür ancak tam saati döndürmez, döndürülen saat her zaman 00:00:00 olur ve yalnızca tarih güncelleştirilir.

**Örnekler**

=Now ( ) Bugunun tarih degerini verecektir

=Now ( ) -2 Iki gun oncesinin degerini verecektir.

=Now ( ) +2 Iki gun sonrasinin degerini verecektir.

18



**MONTH**

**Sözdizimi**

MONTH(<TarihZaman>)

Ay bilgisini 1 ile 12 arasinda bir sayi olarak verir.

**Parametreler**

Tarih/Zaman:

• 02/14/2018 veya February,14 2018 seklinde tarih değeri

**Örnekler**

=MONTH (“February 14, 2017 20:51” ) 2 sonucunu doğuracaktır.

Month\_Number = MONTH (Sales [Ship Date].[Date])

**MINUTE**

Dakikayı bilgisini 1 ile 59 arasinda bir sayi olarak verir.

**Parametreler**

Tarih/Zaman:

• 16:48:00 veya 4:48 PM gibi bir datetime değeri.

**Örnekler**

=MINUTE (“January 15, 2017 20:51” ) 51 sonucunu doğuracaktır.

**HOUR**

HOUR(<Zaman>)

Saati 0 (0:00) ile 23 (23:00) arasında bir sayı olarak döndürür.

**Parametreler**

Zaman:

• 16:48:00 veya 4:48 PM gibi bir datetime değeri.

19



**Örnekler**

= HOUR (“January 15, 2017 20:51”) 20 sonucunu doğuracaktır. = HOUR (‘Sipariş'[TarihSaat]) Sipariş tablosundaki ilgili tarihin saat değerini türetir.

**EDATE Fonksiyonu (DAX)**

Başlangıç tarihinden belirtilen sayıda ay önce veya sonra gelen tarihi döndürür. EDATE işlevini kullanarak düzenleme tarihiyle aynı ay gününe denk gelen vade tarihlerini veya son tarihleri hesaplayabilirsiniz.

EDATE(<start\_date>, <months>)

Aşağıdaki örnek, [TransactionDate] sütununda depolanan sipariş tarihinden üç ay sonraki tarihi döndürür.

= EDATE([TransactionDate],3)

**EOMONTH Fonksiyonu (DAX)**

**Sözdizimi**

EOMONTH (<Tarih>, <AY>)

Önceki yahut sonraki ayların son günlerinin tarihini verebilen işlevdir. 1 Mart 1900’den önceki ve 31 Aralık 9999’dan sonraki tarihler geçerli değildir.

**Parametreler**

Tarih:

• Tarih ve saat biçiminde girilebilen öncesi yahut sonrası alınacak tarih bilgisidir. AY:

• Kaç ay öncesi yahut kaç ay sonrasının bulunacağı belirleyen sayıdır. Ondalık biçimde verilirse yuvarlama yapılarak tam sayıya dönüştürülür ve sonrasında hesaplama yapılır.

**Örnekler**

=EOMONTH (“March 3, 2017”, 1.5 ) 2 ay yukarı yuvarlanacağından 31 Mayıs 2017 sonucunu türetecektir.

=EOMONTH (“March 3, 2017”, -1.5 ) 2 ay aşağıya yuvarlanacağından 31 Ocak 2017 sonucunu türetecektir.

20



**DATEVALUE**

FORMAT = DATEVALUE(FORMAT(InternetSales[OrderDate];"dd/mm/yyyy"))

**Sözdizimi**

**DATEVALUE** ()

**Parametreler**

Tarih\_Metni:

• Metin olarak verinizde bulunan tarihsel veridir. Bölgesel ayarlarınıza göre yazılmış olması gerekir.

**Örnekler**

=**DATEVALUE** (“1/12/2017”)

Fonksiyonun sonucu bölgesel ayarlarınız bazında Ocak 12 2017 yahut Aralık 1 2017 olarak sonuc verecektir.

Now(), Today(), EDATE() , WEEKDAY = WEEKDAY(SalesData[ShipDate];2), **WEEKNUM**

Haftanın\_Numarası= WEEKNUM(SalesData[ShipDate];2),

**CALENDAR Fonksiyonu (DAX)**

**Sözdizimi**

CALENDAR (Başlangıç Tarihi, Bitiş Tarihi)

**Örnekler**

Ornek 1 = CALENDAR (DATE (2017, 1, 1 ), DATE ( 2017, 12, 31 ) )

2017 yılının başından sonuna kadar olan tüm tarihlerin listesini getirir.

Ornek 2 = CALENDAR (MINX (‘Ornek 1’, [Date] ), MAXX ( ‘Ornek 1’, [Date] ) )

Ornek 1 tablosundaki tarih alanındaki en küçük tarih ile en büyük tarih arasındaki bütün tarihlerin listesini getirir.

Ornek 3 = ADDCOLUMNS (

**CALENDAR** ( DATE ( 2017, 1, 1 ), DATE ( 2017, 12, 31 ) ),

21



“**DateAsInteger**”, FORMAT ( [Date], “YYYYMMDD” ),

“**Year**”, YEAR ( [Date] ),

“**MonthNo**”, FORMAT ( [Date], “MM” ),

“**YearMonthNo**”, FORMAT ( [Date], “YYYY/MM” ),

“**YearMonth**”, FORMAT ( [Date], “YYYY/mmm” ),

“**MonthShort**”, FORMAT ( [Date], “mmm” ),

“**MonthLong**”, FORMAT ( [Date], “mmmm” ),

“**WeekNo**”, WEEKDAY ( [Date] ),

“**WeekDay**”, FORMAT ( [Date], “dddd” ),

“**WeekDayShort**”, FORMAT ( [Date], “dddd” ),

“**Quarter**”, FORMAT ( [Date], “Q” ),

“**YearQuarter**”, FORMAT ( [Date], “YYYY” ) & “/Q”

& FORMAT ( [Date], “Q” ))

**DateDimension** = CALENDAR(Date(2015;1;1);Date(2017;3;1))

**DateDimension** = Calendar(Date(2015;1;1)

**Datebetween** = Calculate(Sum('Sales'[TotalAmount]);

DATESBETWEEN(Dates[Date];Date(2014;7;1); Date(2014;8;1)))

**DATESBETWEEN**

Belirtilen başlangıç\_tarihi ile başlayıp belirtilen bitiş\_tarihine kadar devam eden tarihleri içeren bir sütuna sahip olan bir tablo döndürür.

Bu işlev, **CALCULATE** işlevine filtre olarak geçirilmeye uygundur. Özel tarih aralığına göre bir ifadeyi filtrelemek için kullanın.

En yaygın kullanım senaryosunda **dates**, işaretli tarih tablosunun tarih sütununa yönelik bir başvurudur.

• başlangıç tarihi BLANK ise başlangıç\_tarihi, tarihler sütunundaki en erken değer olur. • bitiş\_tarihi BLANK ise bitiş\_tarihi, tarihler sütunundaki en geç değer olur. • başlangıç**\_tarihi** ve **bitiş\_tarihi** olarak kullanılan değerler dahildir. Örneğin

**başlangıç\_tarihi** değeri 1 Temmuz 2019 ise bu tarih, döndürülen tabloya dahil edilir. (Tarihin **tarihler** sütununda var olması şartıyla.)

Döndürülen tablo, yalnızca **dates** sütununda depolanan tarihleri içerebilir. Örneğin, **dates** sütunu 1 Temmuz 2017’den başlıyorsa ve **başlangıç\_tarihi** değeri 1 Temmuz 2016’ysa döndürülen tablo 1 Temmuz 2017’den başlar.

Customers LTD =

**CALCULATE**(

**DISTINCTCOUNT**(Sales[CustomerKey]),

**DATESBETWEEN**(

'Date'[Date],

BLANK(),

MAX('Date'[Date])

)

)

22



**DATESINPERIOD**

Belirtilen başlangıç tarihi ile başlayıp belirtilen sayıya ve tarih aralıklarının türüne kadar devam eden bir tablo döndürür.

Tarih değerlerinden oluşan tek bir sütun içeren tablo DÖNDÜRÜR.

Bu işlev, CALCULATE işlevine filtre olarak geçirilmeye uygundur. Bir ifadeyi gün, ay, çeyrek veya yıl gibi standart tarih aralıklarına göre filtrelemek için kullanın.

DATESINPERIOD(<dates>, <start\_date>, <number\_of\_intervals>, <interval>)

Aşağıdaki **Sales** tablosu ölçü tanımında, önceki yıla (ÖY) ait geliri hesaplamak için DATESINPERIOD işlevi kullanılır.

Formülde MAX işlevinin kullanıldığına dikkat edin. Bu işlev, filtre bağlamındaki en ileri tarihi döndürür. Bu nedenle DATESINPERIOD işlevi, son yıla ait en ileri tarihten başlayan tarihlerin tablosunu döndürür.

Revenue PY =

**CALCULATE**(

SUM(Sales[Sales Amount]),

**DATESINPERIOD**(

'Date'[Date],

MAX('Date'[Date]),

-1,

YEAR

)

)

**DATESMTD**

Geçerli bağlamda ay başından bugüne kadar olan tarihleri içeren sütuna sahip olan bir tablo döndürür.

Aşağıdaki örnek formül, İnternet Satışları için "**Ay Başından Bugüne Toplam**" hesabı yapan bir ölçü oluşturur.

**"Ay Başından Bugüne Toplam"** =

CALCULATE(SUM(InternetSales\_USD[SalesAmount\_USD]),

**DATESMTD**(DateTime[DateKey]))

**DATESQTD**

Geçerli bağlamda üç aylık dönemin başından bugüne kadar olan tarihleri içeren sütuna sahip olan bir tablo döndürür.

Aşağıdaki örnek formül, İnternet Satışları için "**Üç Aylık Döneme Ait Değişen Toplam**" hesabı yapan bir ölçü oluşturur.

23



**"Üç Aylık Döneme Ait Değişen Toplam"** =

CALCULATE(SUM(InternetSales\_USD[SalesAmount\_USD]),

**DATESQTD**(DateTime[DateKey]))

**DATESYTD**

Geçerli bağlamda yıl başından bugüne kadar olan tarihleri içeren sütuna sahip olan bir tablo döndürür.

Aşağıdaki örnek formül, İnternet satışları için "**Yılbaşından bugüne Değişen Toplam**" hesabı yapan bir ölçü oluşturur.

**"Yılbaşından bugüne Değişen Toplam"** =

CALCULATE(SUM(InternetSales\_USD[SalesAmount\_USD]),

DATESYTD(DateTime[DateKey]))

**ENDOFMONTH, ENDOFQUARTER, ENDOFYEAR**

Belirtilen tarih sütunu için geçerli bağlamdaki ayın, çeyreğin veya yılın son tarihini döndürür.

**FIRSTDATE, LASTDATE**

Belirtilen tarih sütunu için geçerli bağlamdaki **ilk veya son tarihi** döndürür.

Aşağıdaki örnek formül, geçerli bağlamda İnternet satış kanalında ilk satışın yapıldığı tarihi alan bir ölçü oluşturur.

= **FIRSTDATE**('InternetSales\_USD'[SaleDateKey])

**PARALLELPERIOD**

Geçerli bağlamda, belirtilen **tarihler** sütunundaki tarihlere paralel bir dönemi temsil eden bir tarih sütununu içeren bir tablo döndürür. Bu tarihler, belirtilen aralık kadar zaman içinde ileri veya geri doğru kaydırılmış olur.

**PARALLELPERIOD**(<dates>,<number\_of\_intervals>,<interval>)

Aşağıdaki örnek formül, İnternet satışları için **önceki yılın satışları hesabı yapan** bir ölçü oluşturur.

= CALCULATE(SUM(InternetSales\_USD[SalesAmount\_USD]), **PARALLELPERIOD**(DateTime[DateKey],-1,year))

24



**PREVIOUSDAY**

Geçerli bağlamda **tarihler** sütunundaki ilk tarihten önceki günü temsil eden tüm tarihleri içeren bir sütuna sahip olan tabloyu döndürür.

Aşağıdaki örnek formül, İnternet satışları için **"Önceki günün satışları"** hesabı yapan bir ölçü oluşturur.

= CALCULATE(SUM(InternetSales\_USD[SalesAmount\_USD]), **PREVIOUSDAY**('DateTime'[DateKey]))

**PREVIOUSMONTH**

Geçerli bağlamda **tarihler** sütunundaki ilk tarihi temel alarak önceki ayda yer alan tüm tarihleri içeren bir sütuna sahip olan tabloyu döndürür.

Aşağıdaki örnek formül, İnternet satışları için **"Önceki ayın satışları"** hesabı yapan bir ölçü oluşturur.

= CALCULATE(SUM(InternetSales\_USD[SalesAmount\_USD]), **PREVIOUSMONTH**('DateTime'[DateKey]))

**PREVIOUSQUARTER**

Geçerli bağlamda **tarihler** sütunundaki ilk tarihi temel alarak önceki üç aylık dönemde yer alan tüm tarihleri içeren bir sütuna sahip olan tabloyu döndürür.

Aşağıdaki örnek formül, İnternet satışları için **"önceki üç aylık dönemin satışları"** hesabı yapan bir ölçü oluşturur.

= CALCULATE(SUM(InternetSales\_USD[SalesAmount\_USD]), **PREVIOUSQUARTER**('DateTime'[DateKey]))

**PREVIOUSYEAR**

Geçerli bağlamda **tarihler** sütunundaki son tarihi temel alarak önceki yılda yer alan tüm tarihleri içeren bir sütuna sahip olan tabloyu döndürür.

Aşağıdaki örnek formül, İnternet satışları için **önceki yılın satışları hesabı** yapan bir ölçü oluşturur.

= CALCULATE(SUM(InternetSales\_USD[SalesAmount\_USD]), **PREVIOUSYEAR**('DateTime'[DateKey]))

**SAMEPERIODLASTYEAR**

Geçerli bağlamda belirtilen **tarihler** sütununda yer alan tarihlerden bir yıl geriye kaydırılmış tarihlerin olduğu bir sütun içeren tablo döndürür.

Aşağıdaki örnek formül, Satıcı satışları için **önceki yılın satışlarını** hesaplayan bir ölçü oluşturur.

25



= CALCULATE(SUM(ResellerSales\_USD[SalesAmount\_USD]), **SAMEPERIODLASTYEAR**(DateTime[DateKey]))

**TOTALMTD**

Geçerli bağlamda ay başından bugüne kadar olan tarihler için **ifade** değerini değerlendirir.

Aşağıdaki örnek formül, İnternet satışları için "**Aylık değişen toplam**" veya "**Aylık değişen toplama**" hesabı yapan bir ölçü oluşturur.

=

**TOTALMTD**(SUM(InternetSales\_USD[SalesAmount\_USD]),DateTime[DateKey])

**TOTALQTD**

Geçerli bağlamda üç aylık dönemden bugüne kadar olan tarihler için **ifade** değerini değerlendirir.

Aşağıdaki örnek formül, İnternet satışları için "**Üç aylık değişen toplam**" veya "**Üç aylık değişen toplama**" hesabı yapan bir ölçü oluşturur.

=

**TOTALQTD**(SUM(InternetSales\_USD[SalesAmount\_USD]),DateTime[DateKey])

**TOTALYTD**

Geçerli bağlamda **ifadenin** yıl başından bugüne kadar olan değerini değerlendirir.

Geçerli yıl başından bugüne **tarihleri** için değerlendiren **ifadeyi** temsil eden bir skaler değerdir.

**IF**

Bir koşulu denetler, TRUE olduğunda bir değer döndürür, olmadığında ikinci değeri döndürür.

IF(<logical\_test>, <value\_if\_true>[, <value\_if\_false>])

İlk örnek, **Liste Fiyatı** sütun değerinin 500’den küçük olup olmadığını test eder. Bu koşul true olduğunda, **Low** değeri döndürülür. **value\_if\_false** değeri olmadığından BLANK döndürülür.

Price Group =

**IF**(

'Product'[List Price] < 500,

"Low"

)

26



İkinci örnek aynı testi kullanır, ancak bu kez bir **value\_if\_false** değeri içerir. Bu nedenle, formül her bir ürünü **Low** veya **High** olarak sınıflandırır.

Price Group =

**IF** (

'Product'[List Price] <500,

"Low",

"High"

)

Üçüncü örnek aynı testi kullanır, ancak bu kez ek bir test gerçekleştirmek üzere bir IF işlevini iç içe yerleştirir. Bu nedenle, formül her bir ürünü **Low**, **Medium** veya **High** olarak sınıflandırır.

Price Group =

**IF** (

'Product'[List Price] <500,

"Low",

**IF** (

'Product'[List Price] <1500,

"Medium",

"High"

)

)

**SWITCH**

Bir ifadeyi bir değer listesine göre değerlendirir ve birden çok olası sonuç ifadesinden birini döndürür.

**SWITCH**(<expression>, <value>, <result>[, <value>, <result>]…[, <else>])

SütunDayName =

SWITCH(InternetSales[WeekNumarası];1;"Pazartesi";2;"Salı";3;"Çarşamba";4;"P erşembe";5;"Cuma";6;"Cumartesi";7;"Pazar")

Aşağıdaki örnek, **ay adlarından bir hesaplanmış sütun** oluşturur.

= **SWITCH**([Month], 1, "January", 2, "February", 3, "March", 4, "April" , 5, "May", 6, "June", 7, "July", 8, "August" , 9, "September", 10, "October", 11, "November", 12, "December"

, "Unknown month number”)

**COALESCE**

BLANK olarak değerlendirilmeyen ilk ifadeyi döndürür. Tüm ifadeler BLANK olarak değerlendirilirse BLANK döndürülür.

Deyimlerin birinden gelen skaler bir değer veya tüm ifadeler BLANK olarak değerlendiriliyorsa BLANK.

27



**BLANK olarak değerlendirilmeyen ilk ifade olan 10** ifadesini döndürür. EVALUATE {**COALESCE**(BLANK(), 10, DATE(2008, 3, 3)) }

FactInternetSales tablosunun SalesAmount sütunundaki tüm değerlerin toplamını veya 0 döndürür.

= **COALESCE**(SUM(FactInternetSales[SalesAmount]), 0)

**OR**

TRUE değerini döndürmek için bağımsız değişkenlerden birinin TRUE olup olmadığını denetler. Her iki bağımsız değişken de FALSE ise işlev FALSE döndürür.

**OR**(<logical1>,<logical2>)

DAX içindeki **OR** işlevi yalnızca iki (2) bağımsız değişken kabul eder. Birden fazla ifadede OR işlemi gerçekleştirmeniz gerekiyorsa hesaplama dizisi oluşturabilir veya daha da iyisi OR işlecini (**||**) kullanarak tümünü birleştirip daha basit bir ifade elde edebilirsiniz.

İşlev, ilk TRUE bağımsız değişkenine kadar olan bağımsız değişkenleri değerlendirir ve ardından TRUE döndürür.

**IFERROR**

Bir ifadeyi değerlendirir ve ifade bir hata döndürürse belirtilen değeri döndürür; aksi takdirde ifade değerini döndürür.

IFERROR(value, value\_if\_error)

• IFERROR işlevini kullanarak bir ifadedeki hataları yakalayabilir ve düzeltebilirsiniz. • **Değer** veya **hata\_değeri** boş bir hücreyse IFERROR bunu boş dize değeri ("") olarak kabul eder.

• IFERROR işlevi IF işlevini temel alır ve aynı hata iletilerini kullanır ancak daha az bağımsız değişkene sahiptir. IFERROR işlevi ile IF işlevi arasındaki ilişki şu şekildedir:

**IFERROR**(A,B) := IF(ISERROR(A), B, A)

A ve B için döndürülen değerlerin aynı veri türüne sahip olması gerekir. Bu nedenle **değer** için kullanılan sütun veya ifade ile **hata\_değeri** için döndürülen değer aynı veri türüne sahip olmalıdır.

Aşağıdaki örnek, 25/0 ifadesinin hata döndürmesi durumunda 9999 sayısını döndürür. = **IFERROR**(25/0,9999)

**NOT**

28



FALSE değerini TRUE değerine veya TRUE değerini FALSE değerine dönüştürür.

**AND**

İki bağımsız değişkenin de TRUE olup olmadığını denetler ve iki bağımsız değişken de TRUE ise TRUE döndürür. Aksi halde false döndürür.

DAX içindeki **AND** işlevi yalnızca iki (2) bağımsız değişken kabul eder. Birden fazla ifadede AND işlemi gerçekleştirmeniz gerekiyorsa hesaplama dizisi oluşturabilir veya daha da iyisi AND işlecini ( **&&** ) kullanarak tümünü birleştirip daha basit bir ifade elde edebilirsiniz.

**FILTER**

Başka bir tablonun veya ifadenin alt kümesini temsil eden bir tablo döndürür. **FILTER**(<table>,<filter>)

TabloSatışOzet = **FILTER** ('Product'; 'Product'[Color]="NA" && 'Product'[ProductAlternateKey]="AR-5381")

TabloSatışOzet = **FILTER** ('Product'; 'Product'[Color]="NA" || 'Product'[ProductAlternateKey]="AR-5381")

FILTER işlevini kullanarak çalıştığınız tablodaki satır sayısını azaltabilir veya hesaplamalara yalnızca belirli verileri dahil edebilirsiniz. FILTER, tek başına değil bağımsız değişken olarak tablo gerektiren diğer işlevlere eklenmiş bir işlev şeklinde kullanılır.

Aşağıdaki örnek, ABD'deki satışları filtreleyen bir ölçü kullanarak ABD dışındaki İnternet satışları raporunu oluşturur

FILTER('InternetSales\_USD',

RELATED('SalesTerritory'[SalesTerritoryCountry])<>"United States")

**RELATED**

N- 1 ; Satış tablosundan Product Tablosuna

RelatedSutun = RELATED('Product'[ModelName])

**Ürün başı toplam satış**

RelatedTableSutun = SUMX(RELATEDTABLE(InternetSales);InternetSales[SalesAmount]) **Farklı Tablodan Veri Çekerek Ölçü oluşturma**

OlçümSumX =

SumX(InternetSales;RELATED('Product'[DealerPrice])\*InternetSales[UnitPrice]) OlçümSumX = SumX(InternetSales;InternetSales[OrderQuantity]\*InternetSales[UnitPrice])

**RELATEDTABLE**

29



Bir tablo ifadesini verilen filtreler tarafından değiştirilen bağlamda değerlendirir. **RELATEDTABLE**(<tableName>)

**RELATEDTETABLE** işlevi, verilerin filtrelendiği bağlamı değiştirir ve ifadeyi belirttiğiniz yeni bağlamda değerlendirir.

Mantıksal ifade içermeyen bu işlev, CALCULATETABLE işlevinin kısayoludur. = SUMX(RELATEDTABLE('InternetSales\_USD'), [SalesAmount\_USD])

**USERELATIONSHIP**

Belirtilen hesaplamada kullanılacak ilişkinin columnName1 ile columnName2 arasındaki ilişki olmasını belirtir.

**USERELATIONSHIP**(<columnName1>,<columnName2>)

**USERELATIONSHIP**, modelde mevcut ilişkileri kullanarak ilişkileri bitiş noktası sütunlarına göre tanımlamayı sağlar.

**USERELATIONSHIP** işlevinde bir ilişkinin durumu önemli değildir. Başka bir deyişle ilişkinin etkin olup olmaması, işlevin kullanımını etkilemez. İlişki devre dışı olsa bile kullanılır ve modelde mevcut olabilecek ancak işlev bağımsız değişkenlerinde belirtilmeyen diğer etkin ilişkileri geçersiz kılar.

Aşağıdaki örnekte InternetSales ile DateTime tabloları arasındaki varsayılan etkin ilişkinin nasıl geçersiz kılınacağı gösterilmiştir. Varsayılan ilişki InternetSales tablosunun OrderDate sütunu ile DateTime tablosunun Date sütunu arasındadır.

İnternet satışlarının toplamını hesaplamak ve geleneksel OrderDate yerine ShippingDate ile dilimlemeye izin vermek için şu ifadeyi kullanarak [InternetSales by ShippingDate] adlı bir ölçü oluşturun:

= CALCULATE(SUM(InternetSales[SalesAmount]),

USERELATIONSHIP(InternetSales[ShippingDate], DateTime[Date]))

InternetSales[ShipmentDate] ile DateTime[Date] arasında ilişki bulunması gerekir ve bu ilişkinin etkin olmasına gerek yoktur. Ayrıca InternetSales[OrderDate] ve DateTime[Date] arasında da ilişki bulunması ve bunun etkin olması gerekir.

**SUMMARIZE- ÖZET TABLO**

= SUMMARIZE ('InternetSales’; InternetSales[SalesAmount];'Product'[ModelName])

OzetTablo = SUMMARIZE('InternetSales';InternetSales[TotalProductCost]; "Ürün Toplam Satışları"; Sum(InternetSales[SalesAmount]); "Ürün Ortalama Maliyet"; SUM(InternetSales[UnitPrice]))

30



OzetTablom = SUMMARIZE ('Sales';Sales[Product Name];Sales[Customer Name];"Total Sales"; SUM(Sales[Total Revenue]))

**SUMMARIZECOLUMNS**

SUMMARIZECOLUMNS( <groupBy\_columnName> [, < groupBy\_columnName >]…, [<filterTable>]…[, <name>, <expression>]…)

Bir grup kümesinden özet tablo döndürür.

**SUMMARIZECOLUMNS** (

'Sales Territory'[Category],

FILTER('Customer', 'Customer' [First Name] = "Alicia") )

**SELECTCOLUMNS**

Hesaplanan sütunları verilen tabloya veya tablo ifadesine ekler.

SELECTCOLUMNS, ADDCOLUMNS ile aynı imzaya sahip ve aynı davranışa sahip ancak <SELECTCOLUMNS, belirtilen tabloyla değil sütun eklemeden önce boş bir tabloyla > başlar.

**SELECTCOLUMNS** (Info, "StateCountry", [State]&", "&[Country])

31

**IF, IF ISBLANK**

****

Sütun = IF(myTable[Product]= ""; "Ürün adı girilmemiş"; [Product])

**CONTAINS**

Başvurulan tüm sütunların değeri mevcutsa veya bu sütunlarda yer alıyorsa true, aksi takdirde false döndürür.

Contains = **CONTAINS** ('Product'; 'Product'[ModelName]; "Product 1")

Aşağıdaki örnekte 214 numaralı ürünün 11185 numaralı müşteriye İnternet üzerinden satılıp satılmadığını belirten bir ölçüm oluşturulur.

= **CONTAINS** (InternetSales, [ProductKey], 214, [CustomerKey], 11185)

**CONTAINSROW**

Bir tablo içinde değer satırı varsa veya mevcutsa TRUE, yoksa FALSE döndürür.

EVALUATE FILTER(ALL(DimProduct[Color]), CONTAINSROW({ "Red", "Yellow", "Blue" }, [Color]))

ORDER BY [Color]

**ALL**

Uygulanmış olan filtreleri yok sayarak bir tablodaki tüm satırları veya bir sütundaki tüm değerleri döndürür. Bu işlev, filtreleri temizlemek ve bir tablodaki satırların tümünü kapsayan hesaplamalar gerçekleştirmek için kullanışlıdır.

32



Kategori Satışlarının Toplam Satışlara oranını hesaplama

= SUMX(ResellerSales\_USD,

ResellerSales\_USD[SalesAmount\_USD])/SUMX(**ALL**(ResellerSales\_USD), ResellerSales\_USD[SalesAmount\_USD])

**SELECTEDVALUE**

columnName bağlamı yalnızca bir ayrı değeri içerecek şekilde filtrelendiğinde ilgili değeri döndürür. Aksi takdirde alternateResult döndürür.

Yalnızca Avustralyalı müşterilere yapılan satışlar için satış vergisi tutarını hesaplar.

Australian Sales Tax =

IF(

**SELECTEDVALUE**(Customer[Country-Region]) = "Australia", [Sales] \* 0.10)

**LOOKUPVALUE**

En az bir arama koşulu tarafından belirtilen tüm ölçütleri karşılayan satır değerini döndürür.

**LOOKUPVALUE** (

<result\_columnName>,

<search\_columnName>,

33



<search\_value>

[, <search2\_columnName>, <search2\_value>]…

[, <alternateResult>]

)

**sonuç\_sütunu** değerinin **arama\_sütunu** ve **arama\_değeri** ile tam olarak eşleşen satırları.

Tüm arama değerlerini karşılayan bir eşleşme yoksa BLANK veya **alternateResult** (sağlanırsa) döndürülür. Başka bir deyişle ölçütlerin yalnızca bir bölümünün eşleşmesi durumunda işlev bir arama değeri döndürmez.

Sonuç ve arama tabloları arasında bir ilişki varsa çoğu durumda LOOKUPVALUE yerine **RELATED** işlevinin kullanılması daha verimli olur ve daha iyi performans sağlar.

**Sales** tablosunda belirtilen aşağıdaki hesaplanmış sütun, **Sales Order** tablosundan kanal değerlerini döndürmek için LOOKUPVALUE işlevini kullanır.

CHANNEL = **LOOKUPVALUE**('Sales Order'[Channel],'Sales Order'[SalesOrderLineKey],[SalesOrderLineKey])

Ancak bu durumda, **Sales Order** ve **Saler** tabloları arasında ilişki olduğundan, **RELATED** işlevini kullanmak daha verimlidir.

CHANNEL = **RELATED** ('Sales Order'[Channel])

**REMOVEFILTERS**

Belirtilen tablo veya sütunlardan filtreleri temizleyin.

**REMOVEFILTERS**([<table> | <column>[, <column>[, <column>[,…]]]]) **REMOVEFILTERS** yalnızca filtreleri temizlemek için kullanılabilir, tablo döndürmez.

**EVALUATE**

Bir veya daha fazla sütuna sahip olan bir tablo döndürür.

**EVALUATE** {1, 2, 3}

**EVALUATE** {(1), (2), (3)}

Tek sütundan oluşan şu tabloyu döndürür:

34



**EVALUATE**

{



(1.5, DATE(2017, 1, 1), CURRENCY(199.99), "A"), (2.5, DATE(2017, 1, 2), CURRENCY(249.99), "B"), (3.5, DATE(2017, 1, 3), CURRENCY(299.99), "C") }

Şunu döndürür:

**EVALUATE** {1, DATE(2017, 1, 1), TRUE, "A" }



**TOPN**

Belirtilen tablonun ilk N satırını döndürür.

• n\_değeri 0 (sıfır) veya daha küçükse TOPN işlevi boş tablo döndürür.

35



• TOPN işlevi, sonuçlar için herhangi bir sıralama düzeni garantisi vermez. Aşağıdaki örnek, en çok satan 10 ürünün satış rakamlarını içeren bir ölçü oluşturur.

= SUMX (**TOPN**(10, SUMMARIZE(Product, [ProductKey], "TotalSales", SUMX(RELATED(InternetSales\_USD[SalesAmount\_USD]),

InternetSales\_USD[SalesAmount\_USD]) +

SUMX(RELATED(ResellerSales\_USD[SalesAmount\_USD]),

ResellerSales\_USD[SalesAmount\_USD]))

**VALUES**

Giriş parametresi bir sütun adı olduğunda, tek sütunlu bir tablo veya belirtilen sütundaki birbirinden farklı değerleri içeren bir tablo döndürür. Yinelenen değerler kaldırılır ve yalnızca benzersiz değerler döndürülür. BLANK değer eklenebilir. Giriş parametresi bir tablo adı olduğunda, belirtilen tablodaki satırları döndürür. Yinelenen satırlar korunur. BLANK satır eklenebilir.

Bu işlev, değerleri bir çalışma sayfasındaki bir hücreye veya sütuna döndürmek için kullanılamaz. Bunun yerine bir formül içine ekleyerek ara işlev olarak kullanabilir, sayılabilecek veya diğer değerleri filtrelemek ya da toplamak için kullanılabilecek benzersiz değerlerden oluşan bir liste elde edebilirsiniz.

DISTINCT işlevi, bağımsız değişken olarak sütun adı veya geçerli bir tablo ifadesi alır ancak VALUES işlevi, bağımsız değişken olarak yalnızca sütun adı veya tablo adı alır.

Aşağıdaki formül, Ürün Kategorisi Adlarını içeren bir raporda kullanıldığında benzersiz faturaları (satış siparişlerini) sayar ve aşağıdaki sonuçları verir:

= **COUNTROWS**(**VALUES**('InternetSales\_USD'[SalesOrderNumber])) **FORMAT**

Değeri, belirtilen biçime göre metne dönüştürür.

= FORMAT (12345.67, "General Number")

= FORMAT (12345.67, "Currency")

= FORMAT (12345.67, "Fixed")

= FORMAT (12345.67, "Standard")

= FORMAT (12345.67, "Percent")

= FORMAT (12345.67, "Scientific")

Şunu döndürür:

**12345,67** "General Sayı", sayıyı biçimlendirme olmadan görüntüler.

**$12.345,67** "Para Birimi", sayıyı para birimi yerel ayarınıza göre biçimlendirir. Buradaki örnekte varsayılan ABD para birimi biçimlendirmesi gösterilmiştir.

36



**12345,67** "Sabit", ondalık ayırıcının sol tarafında en az bir basamak ve ondalık ayırıcının sağında iki basamak görüntüler.

**12.345,67** "Standart", ondalık ayırıcının sol tarafında en az bir basamak ve ondalık ayırıcının sağında iki basamak görüntüler ve binlik ayırıcıları kullanır. Buradaki örnekte varsayılan ABD sayı biçimlendirmesi gösterilmiştir.

**1.234.567,00 %** "Yüzde", sayıyı yüzde değeri (100 ile çarpılmış) olarak biçimlendirir ve sayının sağ tarafında bir boşluk ve yüzde işareti gösterir.

**1,23E+04** "Bilimsel", sayıyı iki basamaklı bilimsel gösterimle biçimlendirir.

Metin İşlevleri

**CONCATENATE**

İki metin dizesini tek bir metin dizesinde birleştirir.

**CONCATENATE**(<text1>, <text2>)

CONCATENATE işlevi, iki metin dizesini tek bir metin dizesinde birleştirir. Birleştirilen öğeler metin, sayılar veya metin olarak gösterilen Boole değerleri veya bunların birleşimi olabilir. Sütun uygun değerleri içeriyorsa sütun başvurusu da kullanabilirsiniz.

DAX'taki CONCATENATE işlevi yalnızca iki bağımsız değişken kabul eder ancak Excel'deki CONCATENATE işlevine en fazla 255 bağımsız değişken eklenebilir. Birden fazla sütunu birleştirmeniz gerekiyorsa hesaplama dizisi oluşturabilir veya daha da iyisi birleştirme işlecini ( **&** ) kullanarak tümünü birleştirip daha basit bir ifade elde edebilirsiniz.

= **CONCATENATE** ("Hello ", "World")

**FIND**

Başka bir metin dizesinin içinde yer alan bir metin dizesinin başlangıç konumunu döndürür. FIND büyük/küçük harfe duyarlıdır.

FIND işlevi joker karakterleri desteklemez. Joker karakter kullanmak için SEARCH işlevini kullanın

Aşağıdaki formül, ürün açıklamasını içeren dizede ürüne ait BMX ifadesinin ilk harfinin konumunu bulur.

= **FIND** ("BMX","line of BMX racing goods")

**SEARCH**

Soldan sağa okuyarak belirli bir karakter veya metin dizesinin ilk bulunduğu karakter sayısını döndürür. **SEARCH** işlevi büyük/küçük harfe duyarlı değildir ancak aksana duyarlıdır.

37



Bu işlevi kullanarak bir metin dizesini ikinci bir metin dizesinin içinde bulabilir ve ilk dizenin başladığı konumu döndürebilirsiniz.

Aşağıdaki formül, "n" harfinin "printer" sözcüğü içindeki konumunu bulur. = **SEARCH**("n","printer")

"n" harfi, "printer" sözcüğünün dördüncü karakteri olduğu için formül 4 değerini döndürür.

Aşağıdaki formül, "-" karakterinin konumunu sütun içinde arar ve dize bulunamazsa -1 değerini döndürür.

= **IFERROR**(SEARCH("-",[PostalCode]),-1)

**EXACT**

İki metin dizesini karşılaştırır ve tamamen aynı olmaları durumunda TRUE, aksi takdirde FALSE döndürür. EXACT, büyük/küçük harfe duyarlıdır ancak biçimlendirme farklarını yoksayar. Bir belgeye girilen metinleri sınamak için EXACT işlevini kullanabilirsiniz.

**EXACT**(<text1>,<text2>)

Aşağıdaki formül, geçerli satırdaki Sütun1 ve Sütun2 değerlerini denetleyip aynı olmaları halinde TRUE, farklı olmaları halinde ise FALSe döndürür.

= **EXACT**([Column1],[Column2])

**REPT**

Metni verilen sayıda yineler. Hücreyi bir metin dizesinin bir dizi örneğiyle doldurmak için REPT kullanın.

Aşağıdaki örnek, [MyText] sütunundaki dizeyi [MyNumber] sütununda belirtilen sayıda yineleyerek döndürür. Formül sütunun tamamını kapsadığından elde edilen dize, her bir satırdaki metne ve sayı değerine göre değişir.

= **REPT**([MyText],[MyNumber])

**LOWER, UPPER**

Metin dizesindeki tüm harfleri küçük ve büyük harfe dönüştürür.

Harf olmayan karakterler değişmez. Örneğin, formül = **LOWER**("123ABC”) = LOWER("123ABC")

Aşağıdaki formül, [ProductCode] sütunundaki her satırı alır ve değeri tümü küçük harfe dönüştürür. Sütundaki sayılar etkilenmez.

= LOWER ('New Products'[ProductCode])

38



Aşağıdaki formül, [ProductCode] sütunundaki dizeyi tamamı büyük harfe dönüştürür. Alfabe dışı karakterler etkilenmez.

= **UPPER** (['New Products'[Product Code])

**VALUE**

Bir sayıyı temsil eden metin dizesini sayıya dönüştürür.

Altyapı, metni gerektiğinde örtük olarak sayıya dönüştürdüğünden genellikle bir formülde VALUE işlevini kullanmanız gerekmez.

**metin** parametresi olarak geçirilen değer, kullanmakta olduğunuz uygulama veya hizmetler tarafından tanınan sabit, sayı, tarih veya saat biçimlerinden herhangi birinde olabilir. **metin** bu biçimlerden birinde değilse bir hata döndürülür.

Aşağıdaki formül "3" metin dizesini 3 sayısal değerine dönüştürür.

= **VALUE** ("3")

**REPLACE**

**REPLACE**, bir metin dizesinin belirttiğiniz sayıda karaktere sahip olan bölümünü başka bir metin dizesiyle değiştirir.

Aşağıdaki formül, [ProductCode] sütununda yer alan ürün kodunun ilk iki karakterini yeni bir iki harfli kodla (OB) değiştiren yeni bir hesaplanmış sütun oluşturur.

= **REPLACE** ('New Products'[Product Code],1,2,"OB")

**SUBSTITUTE**

Varolan metni, bir metin dizesindeki yeni metinle değiştirir.

Bir metin dizesindeki belirli bir metni değiştirmek istediğinizde SUBSTITUTE işlevini, bir metin dizesinin belirli bir konumunda yer alan ve farklı uzunluklara sahip olan metinleri değiştirmek için ise **REPLACE** işlevini kullanın.

**SUBSTITUTE** işlevi büyük/küçük harfe duyarlıdır. **metin** ile **eski\_metin** içindeki büyük/küçük harf durumları aynı değilse SUBSTITUTE işlevi metni değiştirmez.

Aşağıdaki formül sütunda geçen eski **PA** ürün kodu örneklerini yeni **NW** ürün koduyla değiştirerek [Product Code] sütununun bir kopyasını oluşturur.

= **SUBSTITUTE**([Product Code], "NW", "PA")

**TRIM**

Kelimeler arasındaki tek boşluklar hariç olmak üzere bir metindeki tüm boşlukları kaldırır.

39



Aşağıdaki formül, sonunda boşluk bulunmayan yeni bir dize oluşturur.

= **TRIM** ("A column with trailing spaces. ")

**LEFT, RIGHT**

Metnin sağından ve solunda istenen karakter adedinde kesme uygular.

**LEFT** ([Product Code],2)

Bilgi İşlevleri

**ISFILTERED**

columnName doğrudan filtrelendiğinde TRUE döndürür. Sütunda filtre yoksa veya filtrelemenin nedeni, aynı tablodaki veya ilgili tablodaki farklı bir sütunun filtrelenmesi ise işlev FALSE döndürür.

**ISFILTERED**(<columnName>)

**ISEMPTY**

Bir tablonun boş olup olmadığını denetler.

Tablo boşsa (satır yoksa) true, değilse false olur.

**HASONEVALUE**

columnName bağlamı yalnızca bir ayrı değeri içerecek şekilde filtrelendiğinde TRUE döndürür. Aksi takdirde FALSE döndürür.

**HASONEVALUE** () ile eşdeğer olan ifade COUNTROWS(VALUES(<columnName>)) = 1 ifadesidir.

Aşağıdaki örnekte önceden tanımlanmış bir senaryoda yüzde değeri tahmininde bulunmak için bağlamın bir değerle dilimlenip dilimlenmediğini doğrulayan bir ölçüm formülü yer alır. Bu örnekte Bayi Satışlarını 2007 yılındaki satışlarla karşılaştırmak ve bağlamın tek bir yıl ile filtrelenip filtrelenmediğini öğrenmek istiyorsunuz. Ayrıca karşılaştırmanın anlamsız olması durumunda BLANK döndürmek istiyorsunuz.

= IF (**HASONEVALUE**(DateTime

[CalendarYear]),SUM(ResellerSales\_USD[SalesAmount\_USD])/CALCULATE(SU M(ResellerSales\_USD[SalesAmount\_USD]),DateTime[CalendarYear]=2007),B LANK())

40



**HASONEFILTER**

columnName üzerinde doğrudan filtrelenmiş değerlerin sayısı bir olduğunda TRUE döndürür; aksi takdirde FALSE döndürür.

*columnName* üzerindeki doğrudan filtrelenen değerlerin sayısı bir ise **TRUE**, değilse **FALSE**.

Bu işlev, **HASONEVALUE** () ile benzerdir. Ancak **HASONEVALUE** (), çapraz filtrelerle çalışırken **HASONEFILTER**() doğrudan filtre ile çalışır.

**ISODD**

Sayı tek ise TRUE, çift ise FALSE değerini döndürür.

**ISTEXT**

Değerin metin olup olmadığını denetler ve TRUE veya FALSE değerini döndürür.

**ISNUMBER**

Değerin bir sayı olup olmadığını denetler ve TRUE veya FALSE değerini döndürür.

**ISERROR**

Değerin bir hata olup olmadığını denetler ve TRUE veya FALSE değerini döndürür.

**SELECTEDMEASURE**

İfadeler tarafından hesaplama öğelerinin bağlam içindeki ölçüye başvurmasını sağlamak için kullanılır.

Aşağıdaki hesaplama öğesi ifadesi, bağlam içindeki ölçü için yıldan bugüne hesabı gerçekleştirir.

CALCULATE(SELECTEDMEASURE(), DATESYTD(DimDate[Date]))

41