Data Analysis Expressions (DAX)

1. Data Analysis Expressions – основні поняття
2. Опис та приклади застосування агрегатних функцій
3. Опис та приклади застосування функцій фільтрації

Мова програмування DAX, або Data Analysis Expressions

DAX, або Data Analysis Expressions, є мовою програмування, що розроблена для роботи з даними в Power BI та інших інструментах аналізу даних Microsoft. Вона дозволяє створювати формули та вирази для обчислення значень, агрегування даних, фільтрації результатів, створення умовних виразів та багато іншого. Одна з ключових особливостей DAX полягає в тому, що вона працює з відносною моделлю даних. Замість прямого доступу до бази даних, DAX працює з моделлю даних, яка включає таблиці, стовпці та зв'язки між ними. Це дозволяє звертатися до даних у контексті моделі та виконувати розрахунки, використовуючи інформацію з різних таблиць та стовпців.

DAX (Data Analysis Expressions) — це мова виразів, яка використовується для обчислень у моделях даних. Вона нагадує функції Excel, але надає більше можливостей для роботи з даними, особливо в контексті бізнес-аналізу.
Мова DAX використовується для вирішення різноманітних аналітичних завдань, від простих обчислень, таких як додавання та середні значення, до складного аналізу.
DAX дозволяє створювати нові обчислювані стовпці, міри (метрики) та складні фільтри.
Обчислювані стовпці: DAX дозволяє додавати нові стовпці в таблиці, які базуються на обчисленнях. Наприклад, можна створити стовпець “Загальний продаж", який буде сумою продажів по кожному рядку.
Міри: використовуються для обчислення агрегатних значень: сума, середнє значення, мінімум або максимум. Наприклад, міра "Середній продаж" може розраховувати середнє значення продажів за всіма записами.
Контекст: ключове поняття, яке впливає на обчислення та результати формул. Контекст дає змогу виконувати динамічний аналіз, тобто коли результати формули можуть змінюватися.

Основні функції DAX
Функції DAX можна розділити на кілька груп:
1. функції зв'язків,
2. агрегатні функції,
3. логічні,
4. функції фільтрації,
5. функції дати та часу,
6. фінансові,
7. інформаційні функції,
8. математичні,
9. статистичні та інші функції

Функція DAX завжди посилається на повний стовпець або таблицю. Якщо потрібно використовувати лише певні значення з таблиці або стовпця, до формули можна додати фільтри.

Якщо потрібно настроїти обчислення на основі рядків, daX надає функції, які дають змогу використовувати поточне значення рядка або пов'язане значення як тип аргументу, для виконання обчислень, які залежать від контексту. Ви дізнаєтеся більше про контекст пізніше.

DaX містить багато функцій, які повертають таблицю, а не значення. Таблиця не відображається, але використовується для введення даних для інших функцій. Наприклад, можна отримати таблицю, а потім підрахувати окремі значення в ній або обчислити динамічні суми в фільтрованих таблицях або стовпцях.

DaX містить різноманітні функції часового аналізу. Ці функції дають змогу визначати або вибирати діапазони дат і виконувати динамічні обчислення на їх основі. Наприклад, можна порівняти суми в паралельних періодах.

Іноді важко дізнатися, які функції потрібно використовувати у формулі. Power Pivot та табличний конструктор моделей у засобах sql Server Data Tools включають функцію Вставлення функції – діалогове вікно, яке дає змогу вибирати функції за категоріями та стислий опис для кожної функції.

Функції зв'язків

Використовуються для витягування значень із пов'язаних таблиць, що дозволяє ефективно працювати зі зв'язними даними. Ці функції особливо корисні для роботи з реляційними даними, де таблиці пов'язані між собою через ключі.

RELATED

Повертає значення з пов'язаної таблиці, яка знаходиться у зв'язку "один-до-багатьох".

Синтаксис: RELATED(<ColumnName>)

Приклад: ProductCategory = RELATED(Products[Category])

Функція RELATED отримує значення з колонки Category таблиці Products, яка пов'язана з поточною таблицею через відношення "один-до-багатьох". У цьому прикладі ми припускаємо, що існує зв'язок між таблицею продажів і таблицею продуктів.

RELATEDTABLE

Повертає таблицю, що містить всі рядки з пов'язаної таблиці, які відповідають поточному рядку.

Синтаксис: RELATEDTABLE(<TableName>)

Приклад: TotalSalesForCustomer = SUMX(RELATEDTABLE(Sales), Sales[Amount])

Функція RELATEDTABLE повертає всі рядки з таблиці Sales, які пов'язані з поточним рядком в основній таблиці, наприклад, з таблицею Customers. SUMX у цьому прикладі обчислює загальну суму Amount для всіх продажів, пов'язаних з поточним клієнтом.

Агрегатні функції

Агрегатні функції дозволяють обчислювати сумарні значення на основі певного стовпця або таблиці.

SUM

Повертає суму всіх значень у стовпчику

Синтаксис: SUM(<ColumnName>)

Приклад: TotalRevenue = SUM(Sales[Revenue])

AVERAGE

Повертає середнє значення чисел у стовпчику

Синтаксис: AVERAGE(<ColumnName>)

Приклад: AverageRevenue = AVERAGE(Sales[Revenue])

MIN та MAX:

Обчислюють мінімальне та максимальне значення відповідно.

Синтаксис: MAX(<ColumnName>), MIN(<ColumnName>)

Приклад: MinSale = MIN(Sales[Amount])

MIN повертає найменше значення з колонки Amount.

COUNT

Підраховує кількість рядків із непорожніми значеннями,

Синтаксис: COUNT(<ColumnName>)

Приклад: TotalSales = COUNT(Sales[OrderID])

Функція COUNT повертає кількість значень у стовпчику OrderID.

DISTINCTCOUNT

Повертає кількість унікальних значень у стовпчику

Синтаксис: DISTINCTCOUNT(<ColumnName>)

Приклад: UniqueCustomers = DISTINCTCOUNT(Sales[CustomerID])

Функція DISTINCTCOUNT підраховує кількість унікальних значень у колонці CustomerID.

SUMX

повертає суму виразу, обчисленого для кожного рядка таблиці

Синтаксис: SUMX(<Table>, <Expression>)

Приклад: TotalProfit = SUMX(Sales, Sales[Quantity] * Sales[ProfitPerUnit])

AVERAGEX

Повертає середнє значення виразу, обчисленого для кожного рядка таблиці
Синтаксис: AVERAGEX(<Table>, <Expression>)
Приклад: AverageProfitPerSale = AVERAGEX(Sales, Sales[ProfitPerUnit] * Sales[Quantity])
Функція AVERAGEX обчислює вираз Sales[ProfitPerUnit] * Sales[Quantity] для кожного рядка таблиці Sales, а потім обчислює середнє значення цих результатів.

додавання функції до формули за допомогою функції Insert

У таблиці FactSales прокрутіть список до найбільшого правого стовпця, а потім у заголовку стовпця натисніть кнопку Додати стовпець.

У рядку формул введіть знак рівності =.

Натисніть кнопку Вставити функцію . Вставлення функції Відкриється діалогове вікно Вставлення функції .

У діалоговому вікні Вставлення функції клацніть список Виберіть категорію . За замовчуванням вибрано параметр Усі , а всі функції в категорії Усі перелічено нижче. Це багато функцій, тому вам потрібно буде відфільтрувати функції, щоб спростити пошук потрібного типу функції.

Для цієї формули потрібно повернути деякі дані, які вже існують в іншій таблиці. Для цього потрібно використовувати функцію в категорії Фільтр. Перейдіть вперед і виберіть категорію Фільтр , а потім у розділі Виберіть функцію прокрутіть униз і двічі клацніть функцію RELATED. Натисніть кнопку OK , щоб закрити діалогове вікно Вставлення функції .

Скористайтеся функцією IntelliSense, щоб знайти та вибрати стовпець DimChannel[ChannelName].

Закрийте формулу та натисніть клавішу Enter.

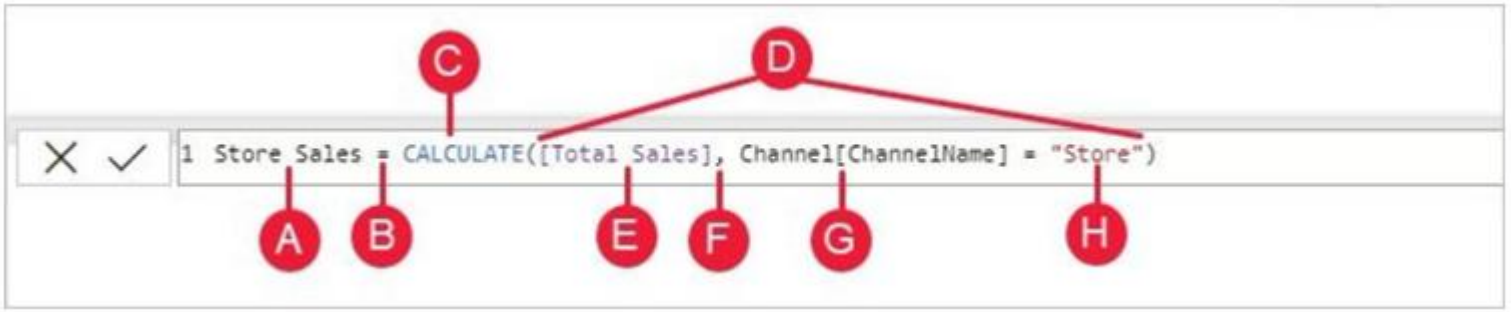
Коли ви натиснете клавішу Enter, щоб завершити формулу, слово Обчислення з'явиться в рядку стану в нижній частині вікна Power Pivot. Тепер ви побачите, що щойно створили новий стовпець у таблиці FactSales із відомостями про канал із таблиці DimChannel.

Перейменуйте стовпець Канал.

Формула має виглядати так: =RELATED(DimChannel[ChannelName])

Синтаксис

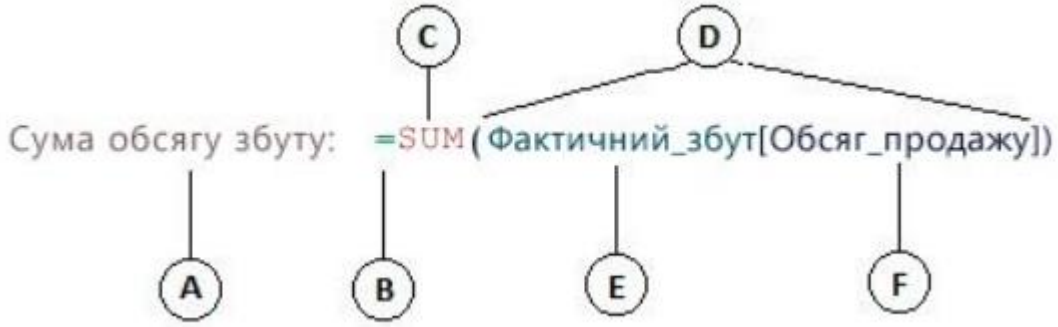
Перш ніж створювати власні формули, розгляньмо синтаксис формули DAX. Синтаксис включає різні елементи, які складають формулу, або, простіше кажучи, спосіб написання формули. Наприклад, розгляньмо просту формулу DAX, яка використовується для створення нових даних (значень) для кожного рядка в обчислюваному стовпці з іменем Margin у таблиці FactSales: (кольори тексту формули призначені лише для ілюстрації)



Синтаксис цієї формули містить такі елементи:

Оператор знака рівності (=) позначає початок формули, і коли ця формула обчислюється, вона повертає результат або значення. Усі формули, які обчислюють значення, починаються зі знака рівності.
Стовпець [SalesAmount] містить значення, від які потрібно відняти. Посилання на стовпець у формулі завжди оточено квадратними дужками []. На відміну від формул Excel, які посилаються на клітинку, формула DAX завжди посилається на стовпець.
Математичний оператор віднімання (-).
Стовпець [Загальна вартість] містить значення, які потрібно відняти від значень у стовпці [SalesAmount].
Коли ви намагаєтеся зрозуміти, як читати формулу DAX, часто корисно розбивати кожен елемент на мову, про які ви думаєте, і говорити щодня.
Наприклад, цю формулу можна прочитати так:
У таблиці FactSales для кожного рядка в обчислюваному стовпці Margin обчислює (=) значення, віднімаючи (-) значення в стовпці [Загальна вартість] від значень у стовпці [SalesAmount].

Розгляньмо інший тип формули, який використовується в мірі:



Ця формула містить такі синтаксичні елементи:
Ім'я міри Сума обсягу збуту. Формули для мір можуть містити ім'я міри, а потім двокрапку, а потім формулу обчислення.
Оператор знака рівності (=) позначає початок формули обчислення. Під час обчислення повертається результат.
Функція SUM додає всі числа в стовпці [SalesAmount]. Ви дізнаєтеся більше про функції пізніше.
Дужки () оточують один або кілька аргументів. Для всіх функцій потрібен принаймні один аргумент. Аргумент передає значення функції.
Таблиця FactSales, на яку посилаються.
Стовпець [SalesAmount] у таблиці FactSales. За допомогою цього аргументу функція SUM знає, у якому стовпці потрібно об'єднати функцію SUM.
Цю формулу можна прочитати так:
Для міри "Сума обсягу збуту" обчислити (=) суму значень у стовпці [SalesAmount] таблиці FactSales .

У разі розміщення в зоні розкривного списку значень у списку полів зведеної таблиці ця міра обчислює та повертає значення, визначені кожною клітинкою зведеної таблиці, наприклад "Мобільні телефони" в США.

Зверніть увагу, що ця формула відрізняється від формули, яка використовується для обчислюваного стовпця "Поле". Зокрема, ми ввели функцію SUM. Функції – це попередньо написані формули, які спрощують виконання складних обчислень і маніпуляцій із числами, датами, часом, текстом тощо. Ви дізнаєтеся більше про функції пізніше.

На відміну від обчислюваного стовпця Margin раніше, перед стовпцем [SalesAmount] передувала таблиця FactSales, до якої належить стовпець. Це повне ім'я стовпця, яке містить ім'я стовпця, яке передує імені таблиці. Для стовпців, на які посилається та сама таблиця, ім'я таблиці не обов'язково включати у формулу. Це може зробити довгі формули, які посилаються на багато стовпців коротшим і легше читати. Проте радимо завжди включати ім'я таблиці у формули мір, навіть якщо вони містяться в одній таблиці.

Примітка.: Якщо ім'я таблиці містить пробіли, зарезервовані ключові слова або заборонені символи, ім'я таблиці потрібно брати в одинарні лапки. Імена таблиць також слід брати в лапки, якщо ім'я містить будь-які символи за межами діапазону букв і цифр ANSI незалежно від того, чи підтримує локалізація набір символів.

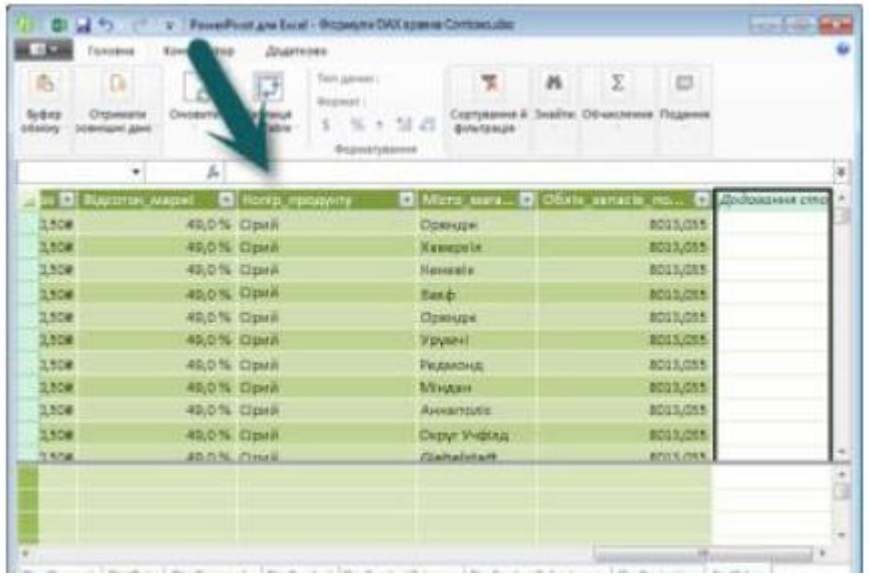
Завдання: створення простої формули для обчислюваного стовпця

Якщо вікно Power Pivot ще не відкрито, у програмі Excel на стрічці Power Pivot натисніть кнопку Power Pivot Вікно.

У вікні Power Pivot виберіть таблицю FactSales (вкладка).

Прокрутіть до найбільшого стовпця праворуч, а потім у заголовку стовпця натисніть кнопку Додати стовпець.

Клацніть у рядку формул у верхній частині вікна конструктора моделі.



1.Коли курсор активний у рядку формул, ці три кнопки стають активними. Крайня ліва кнопка (**X**) – це просто кнопка скасування. Перейдіть вперед і клацніть його. Курсор більше не відображатиметься в рядку формул, а кнопка скасувати та позначка більше не відображатимуться. Перейдіть вперед і знову клацніть у рядку формул. Кнопка "Скасувати" та кнопка "Позначка" знову з'являться. Це означає, що ви готові почати вводити формулу.

2.Кнопка "Позначити" – це кнопка "Перевірити формулу". Це не робить багато, доки ви не ввели формулу. Ми трохи повернемося до неї.

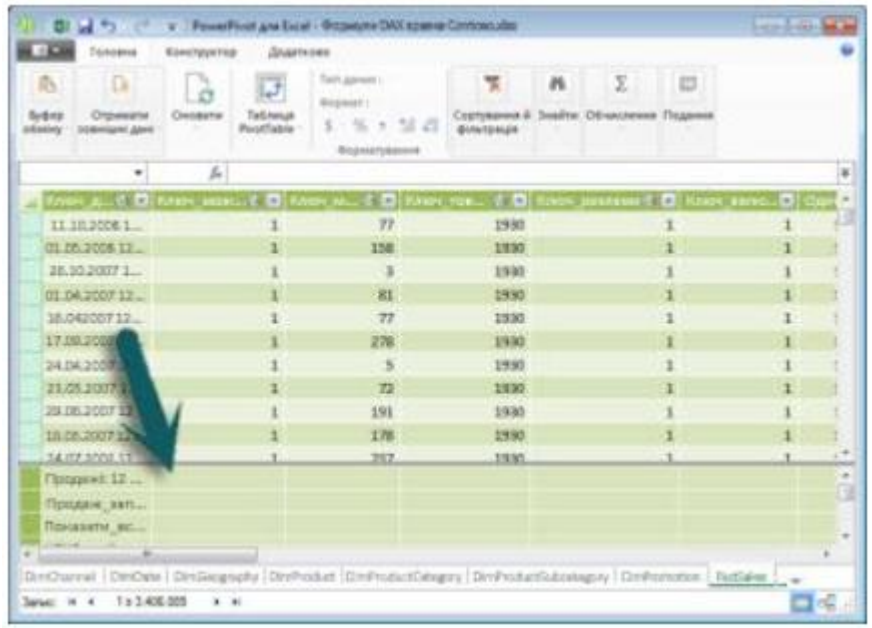
3.Натисніть кнопку **Fx** . З'явиться нове діалогове вікно; діалогове вікно Вставлення функції. Діалогове вікно Вставлення функції – це найпростіший спосіб почати вводити формулу DAX. Ми додамо функцію до формули, коли трохи пізніше створимо міру, але поки що додавати функцію до формули обчислюваного стовпця не потрібно. Закрийте діалогове вікно Вставлення функції.

Аналіз результату

Щойно створили просту, але дуже потужну формулу DAX. Для кожного рядка в таблиці FactSales формула NetSales обчислює значення, віднімаючи значення в стовпці [ReturnQuantity] від значення в стовпці [Обсяг продажів]. Зверніть увагу, як ми щойно сказали "Для кожного рядка". Це уявлення про ще одну дуже важливу концепцію в DAX; контексту рядка. Докладніше про контекст рядків можна дізнатися пізніше.
Щось дуже важливе, що потрібно зрозуміти, вводячи оператор у формулу DAX, – це тип даних в аргументах, які ви використовуєте. Наприклад, якщо ввести таку формулу, = 1 & 2, повернуте значення буде текстовим значенням "12". Це відбувається тому, що оператор амперсанд (&) призначений для об'єднання тексту. DaX інтерпретує цю формулу, щоб прочитати: обчислити результат, взявши значення 1 як текст і додайте значення 2 як текст. Тепер, якщо ви ввели = 1 + 2, DAX читає цю формулу як: Обчислити результат, взявши числове значення 1 і додавши числове значення 2. Результат, звичайно, "3", числове значення. DaX обчислює результувані значення залежно від оператора у формулі, а не на основі типу даних стовпців, які використовуються в аргументі. Типи даних у DAX дуже важливі, але за межами області цього короткого посібника.

Завдання: створення формули міри

У таблиці FactSales клацніть будь-яку пусту клітинку в області обчислення. Це область пустих клітинок під таблицею у вікні Power Pivot.



У рядку формул введіть ім'я Попередній квартал Продаж:.
Введіть знак рівності = для початку формули обчислення.
Введіть кілька перших букв CAL, а потім двічі клацніть потрібну функцію. У цій формулі потрібно використовувати функцію CALCULATE .
Введіть відкривну дужку (щоб почати передавати аргументи функції CALCULATE.
Зверніть увагу: якщо ввести відкривну дужку, функція IntelliSense відобразить аргументи, необхідні для функції CALCULATE. Ви дізнаєтеся про аргументи трохи.
Введіть кілька перших букв таблиці FactSales , а потім у розкривному списку двічі клацніть FactSales[Sales].
Введіть кому (,), щоб указати перший фільтр, а потім введіть ПРЕ, а потім двічі клацніть функцію PREVIOUSQUARTER .
Якщо вибрати функцію PREVIOUSQUARTER, з'явиться ще одна відкривна дужка, яка вказує на те, що потрібен інший аргумент; на цей раз для функції PREVIOUSQUARTER.
Введіть кілька перших букв Dim і двічі клацніть DimDate[DateKey].
Закрийте обидва аргументи, які передаються функції PREVIOUSQUARTER, і функцію CALCULATE, ввівши дві закривні дужки)).
Тепер формула має виглядати так:
Попередній квартал продажів:=CALCULATE(FactSales[Продажі], PREVIOUSQUARTER(DimDate[DateKey]))
Натисніть кнопку "Перевірити формулу" в рядку формул, щоб перевірити формулу. Якщо з'явиться повідомлення про помилку, перевірте кожен елемент синтаксису.

Аналіз завдання

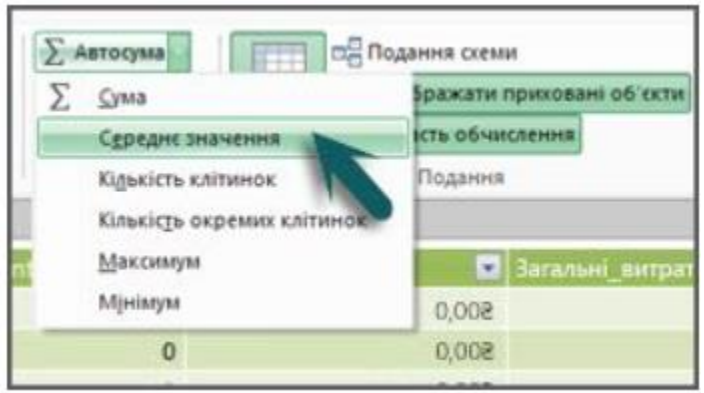
Ця формула обчислює загальний обсяг збуту за попередній квартал залежно від фільтрів, застосованих у зведеній таблиці або зведеній діаграмі.

Вас щойно ознайомили з кількома важливими аспектами формул DAX. По-перше, ця формула включала дві функції. Зверніть увагу, що функція PREVIOUSQUARTER вкладено як аргумент, що передається функції CALCULATE . Формули DAX можуть містити до 64 вкладених функцій. Навряд чи формула коли-небудь міститиме стільки вкладених функцій. Насправді така формула була б дуже важко створити і налагодити, і це, ймовірно, не було б дуже швидко або.

У цій формулі також використовуються фільтри. Фільтри звужують обчислювані елементи. У цьому випадку ви вибрали один фільтр як аргумент, який фактично є іншою функцією. Ви дізнаєтеся більше про фільтри пізніше.

створення формули міри за допомогою функції "Автосума"

У таблиці FactSales прокрутіть до стовпця ReturnQuantity і клацніть заголовок стовпця, щоб виділити весь стовпець. На вкладці Основне на стрічці в групі Обчислення натисніть кнопку Автосума .



Клацніть стрілку вниз поруч із кнопкою **Автосума**, а потім виберіть **пункт Середнє** (зверніть увагу на інші стандартні функції агрегації, які також можна використовувати).

Одразу створюється нова міра з іменем Average of ReturnQuantity: і формулою =AVERAGE([ReturnQuantity]).

Функції фільтрів і значень у DAX є одними з найскладніших і найпотужніших значно відрізняються від функцій Excel.
Функції пошуку працюють за допомогою таблиць і відносини, як база даних. Функції фільтрації дозволяють маніпулювати контекстом даних створювати динамічні розрахунки.

```
DAX

ALL( [<table> | <column>[, <column>[, <column>[,…]]]] )
```

Аргументом функції ALL має бути або посилання на базову таблицю, або а посилання на базовий стовпець. Ви не можете використовувати табличні вирази або вирази стовпців з функцією ALL.
Результат: Таблиця або стовпець із видаленими фільтрами
Ця функція не використовується сама по собі, а служить проміжною функцією, яка може використовувати для зміни набору результатів, над якими виконується інший обчислення виконується. Нормальною поведінкою виразів DAX, що містять функцію ALL(), є будь-яка застосовані фільтри ігноруватимуться. Однак є сценарії, коли це не так випадок через auto-exist, технологію DAX, яка оптимізує фільтрацію в порядку щоб зменшити обсяг обробки, необхідний для певних запитів DAX.

Приклад 1:

обчислити співвідношення продажів категорії до загального обсягу продажів Припустімо, що ви хочете знайти суму продажів для поточної комірки у зведеній таблиці, поділений на загальний обсяг продажів для всіх посередників. Щоб переконатися, що знаменник однаковий незалежно від того, як користувач PivotTable може фільтрувати чи групувати дані, визначаєте ви формула, яка використовує ALL для створення правильного загального підсумку. У наведеній нижче таблиці наведено результати, отримані за допомогою нового показника «Коефіцієнт продажів усіх торгових посередників». створено за формулою, наведеною в розділі коду. Щоб побачити, як це працює, додайте поле CalendarYear до області «Мітки рядків» зведеної таблиці та додайте поле, ProductCategoryName, до області «Мітки стовпців». Потім перетягніть міру, All Reseller Коефіцієнт продажів до області значень зведеної таблиці. Щоб переглянути результати у відсотках, використовуйте функції форматування Excel для застосування форматування чисел у відсотках до клітинок що містить міру

Row Labels	Accessories	Bikes	Clothing	Components	Grand Total
2005	0.02%	9.10%	0.04%	0.75%	9.91%
2006	0.11%	24.71%	0.60%	4.48%	29.90%
2007	0.36%	31.71%	1.07%	6.79%	39.93%
2008	0.20%	16.95%	0.48%	2.63%	20.26%
Grand Total	0.70%	82.47%	2.18%	14.65%	100.00%

```
= SUMX(ResellerSales_USD,
ResellerSales_USD[SalesAmount_USD]
)/SUMX(ALL(ResellerSales_USD),
ResellerSales_USD[SalesAmount_USD]
)
```

Формула будується наступним чином:

1. Чисельник SUMX(ResellerSales_USD, ResellerSales_USD[SalesAmount_USD]) дорівнює сума значень у ResellerSales_USD[SalesAmount_USD] для поточної клітинки у зведеній таблиці з контекстними фільтрами, застосованими до CalendarYear і ProductCategoryName.
2. Для знаменника ви починаєте із зазначення таблиці ResellerSales_USD і використовуєте функція ALL, щоб видалити всі контекстні фільтри в таблиці.
3. Потім ви використовуєте функцію SUMX, щоб підсумувати значення в Стовпець ResellerSales_USD[SalesAmount_USD].

Іншими словами, ви отримуєте суму ResellerSales_USD[SalesAmount_USD] для всіх продажів торговельних посередників.

Приклад 2:

Обчислити відношення продажів продукції до загального обсягу продажів протягом поточного року Припустімо, що ви хочете створити таблицю, яка б відображала відсоток продажів у порівнянні роки для кожної категорії продукту (ProductCategoryName). Щоб отримати відсоток для кожного року на кожне значення ProductCategoryName потрібно розділити суму продажі за конкретний рік і категорію продукту за сумою продажів за той самий рік категорії продукту за всі роки. Іншими словами, ви хочете залишити фільтр увімкненим ProductCategoryName, але видаліть фільтр на рік під час обчислення знаменник відсотка. У наведеній нижче таблиці наведено результати за нової міри «Рік продажів торговельного посередника». створено за формулою, наведеною в розділі коду. Щоб побачити, як це працює, додайте поле CalendarYear до області «Мітки рядків» зведеної таблиці та додайте поле, ProductCategoryName, до області «Мітки стовпців». Щоб переглянути результати у відсотках, скористайтеся функціями форматування Excel, щоб застосувати формат чисел у відсотках до клітинок містить показник «Рік продажів торговельного посередника».

Row labels	Accessories	Bikes	Clothing	Components	Grand Total
2005	3.48%	11.03%	1.91%	5.12%	9.91%
2006	16.21%	29.96%	27.29%	30.59%	29.90%
2007	51.62%	38.45%	48.86%	46.36%	39.93%
2008	28.69%	20.56%	21.95%	17.92%	20.26%
Grand Total	100.00%	100.00%	100.00%	100.00%	100.00%

= SUMX(ResellerSales_USD, ResellerSales_USD[SalesAmount_USD])/CALCULATE(SUM(ResellerSales_USD[SalesAmount_USD]), ALL(DateTime[CalendarYear]))

Формула будується наступним чином:

Чисельник SUMX(ResellerSales_USD, ResellerSales_USD[SalesAmount_USD]) дорівнює сума значень у ResellerSales_USD[SalesAmount_USD] для поточної клітинки у зведеній таблиці з контекстними фільтрами, застосованими до стовпців CalendarYear і ProductCategoryName.

Для знаменника, ви видаляєте наявний фільтр на CalendarYear за допомогою Функція ALL(Column). Це обчислює суму для решти рядків на Таблиця ResellerSales_USD після застосування наявних контекстних фільтрів зі стовпця етикетки. Чистий ефект полягає в тому, що для знаменника сума обчислюється за вибране ProductCategoryName (приблизний контекстний фільтр) і для всіх значень у рік

Приклад 3.

Обчисліть внесок категорій продуктів у загальний обсяг продажів за рік Припустімо, що ви хочете створити таблицю, яка показуватиме відсоток продажів для кожного категорію продукту на основі рік за роком. Щоб отримати відсоток для кожного товару категорії в певний рік, вам потрібно обчислити суму продажів для цього конкретного року категорію продукту (ProductCategoryName) у році n, а потім розділіть отримане значення за сумою продажів за рік n по всіх товарних категоріях. Іншими словами, ви хочете щоб залишити фільтр на рік, але видалити фільтр на ProductCategoryName, коли обчислення знаменника відсотка. У наведеній нижче таблиці показано результати нового показника «Продажі торговельного посередника». CategoryName створюється за допомогою формули, наведеної в розділі коду. Щоб побачити, як це працює, додайте поле CalendarYear до області «Мітки рядків» зведеної таблиці та додайте поле ProductCategoryName в область «Мітки стовпців». Потім додайте нову міру до область значень зведеної таблиці. Щоб переглянути результати у відсотках, використовуйте Excel функції форматування для застосування формату чисел у відсотках до клітинок, які містять новий показник, Reseller Sales CategoryName.

Row Labels	Accessories	Bikes	Clothing	Components	Grand Total
2005	0.25%	91.76%	0.42%	7.57%	100.00%
2006	0.38%	82.64%	1.99%	14.99%	100.00%
2007	0.90%	79.42%	2.67%	17.01%	100.00%
2008	0.99%	83.69%	2.37%	12.96%	100.00%
Grand Total	0.70%	82.47%	2.18%	14.65%	100.00%

```
= SUMX(ResellerSales_USD, ResellerSales_USD[SalesAmount_USD])/CALCULATE(
SUM( ResellerSales_USD[SalesAmount_USD]),
ALL(ProductCategory[ProductCategoryName]))
```

Формула будується наступним чином: 1. Чисельник SUMX(ResellerSales_USD, ResellerSales_USD[SalesAmount_USD]) дорівнює сума значень у ResellerSales_USD[SalesAmount_USD] для поточної клітинки у зведеній таблиці з контекстними фільтрами, застосованими до полів CalendarYear і ProductCategoryName. 2. Для знаменника ви використовуєте функцію ALL(Column), щоб видалити фільтр на ProductCategoryName та обчисліть суму для решти рядків у Таблиця ResellerSales_USD після застосування наявних контекстних фільтрів із рядка етикетки. Чистий ефект полягає в тому, що для знаменника сума обчислюється за обраний Рік (приблизний контекстний фільтр) і для всіх значень ProductCategoryName.

Calculate - Обчислює вираз у зміненому контексті фільтра.

```
DAX

CALCULATE(<expression>[, <filter1> [, <filter2> [, …]]])
```

Вираз, який використовується як перший параметр, по суті, такий самий, як міра.
Фільтри можуть бути:

1. Логічні вирази фільтрів
2. Таблиця фільтрів виразів
3. Функції модифікації фільтра

Логічні вирази фільтрів Фільтр логічного виразу – це вираз, який має значення TRUE або FALSE. Є кілька правил, яких вони повинні дотримуватися: Вони можуть посилатися на стовпці з однієї таблиці. Вони не можуть посилатися на заходи. Вони не можуть використовувати вкладену функцію CALCULATE.

Total sales on the last selected date =
CALCULATE (SUM (Sales[Sales Amount]),
'Sales'[OrderDateKey] = MAX (
'Sales'[OrderDateKey]))

Таблиця фільтрів виразів застосовує об’єкт таблиці як фільтр. Це може бути посилання на а модель таблиці, але, швидше за все, це функція, яка повертає об’єкт таблиці. Ви можете використовувати Функція FILTER для застосування складних умов фільтрації, включно з тими, яких неможливо визначається логічним виразом фільтра.

Якщо надано вирази фільтра, функція CALCULATE змінює фільтр контекст для оцінки виразу. Для кожного виразу фільтра є два можливі стандартні результати, коли вираз фільтра не загорнутий у Функція KEEPFILTERS: Якщо стовпців (або таблиць) немає в контексті фільтра, нові фільтри будуть додано до контексту фільтра для оцінки виразу. Якщо стовпці (або таблиці) уже знаходяться в контексті фільтра, наявні фільтри будуть бути перезаписано новими фільтрами для обчислення виразу CALCULATE. Функція CALCULATE, яка використовується без фільтрів, відповідає певній вимогі. Це перетворює контекст рядка на контекст фільтра. Він потрібен, коли вираз (не а вимірювання моделі), який узагальнює дані моделі, потрібно оцінити в рядку контекст. Цей сценарій може статися у формулі обчислюваного стовпця або коли обчислюється вираз у функції ітератора. Зауважте, що коли міра моделі є використовується в контексті рядка, перехід контексту відбувається автоматично. Ця функція не підтримується для використання в режимі DirectQuery, коли використовується в обчислювані стовпці або правила безпеки на рівні рядків (RLS).

Приклад 1. визначення показника таблиці продажів створює результат доходу, але лише для продукти синього кольору.

Blue Revenue = CALCULATE(SUM(Sales[Sales Amount]), 'Product'[Color] = "Blue")

Category	Sales Amount	Blue Revenue
Accessories	$1,272,057.89	$165,406.62
Bikes	$94,620,526.21	$8,374,313.88
Clothing	$2,117,613.45	$259,488.37
Components	$11,799,076.66	$803,642.10
Total	**$109,809,274.20**	**$9,602,850.97**

Функція CALCULATE обчислює суму стовпця Sales Amount таблиці Sales у а змінений контекст фільтра. Новий фільтр додається до стовпця «Колір» таблиці «Товар» або filter перезаписує будь-який фільтр, який уже застосовано до стовпця.

Наведене нижче визначення показника таблиці продажів дає співвідношення продажів над продажами для всіх канали збуту

Channel	Sales Amount	Revenue % Total Channel
Internet	$29,358,677.22	26.74%
Reseller	$80,450,596.98	73.26%
Total	**$109,809,274.20**	**100.00%**

Функція DIVIDE ділить вираз, який підсумовує суму продажів таблиці Sales значення стовпця (у контексті фільтра) тим самим виразом у зміненому контексті фільтра. Це функція CALCULATE, яка змінює контекст фільтра за допомогою Функція REMOVEFILTERS, яка є функцією модифікатора фільтра. Він видаляє фільтри з Стовпець каналів таблиці замовлень на продаж

```
Revenue % Total Channel = DIVIDE( SUM(Sales[Sales Amount]), CALCULATE( SUM(Sales[Sales Amount]), REMOVEFILTERS('Sales Order'[Channel]) ) )
```

Наведене нижче визначення обчислюваного стовпця таблиці клієнтів класифікує клієнтів на а клас лояльності. Це дуже простий сценарій: коли дохід, створений клієнтом, є менше 2500 доларів, вони класифікуються як низькі; інакше вони високі.

```
Customer Segment = IF( CALCULATE(SUM(Sales[Sales Amount]), ALLEXCEPT(Customer, Customer[CustomerKey])) < 2500, "Low", "High" )
```

FILTER - ФІЛЬТР, щоб зменшити кількість рядків у таблиці, з якою ви працюєте і використовувати в розрахунках лише конкретні дані. FILTER самостійно не використовується, але як функція, яка вбудована в інші функції, які потребують таблиці як аргумент.

```
DAX

FILTER(<table>,<filter>)
```

У наступному прикладі створюється звіт про продажі в Інтернеті за межами Сполучених Штатів використовуючи показник, який відфільтровує продажі в Сполучених Штатах, а потім нарізає за календарем рік і категорії продукції. Щоб створити цей показник, відфільтруйте таблицю Інтернет-продажі доларів США, використовуючи територію продажу, а потім використовуйте відфільтровану таблицю у функції SUMX.

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

Повертає таблицю, яка є підмножиною Інтернет-продажів мінус усі рядки, які належать United Територія продажу штатів. Функція RELATED – це те, що пов’язує ключ Territory в Інтернеті Таблиця Sales для SalesTerritoryCountry у таблиці SalesTerritory. У наведеній нижче таблиці демонструється підтвердження концепції заходу, НЕ США Інтернет-продажі, формула для яких наведена в розділі коду нижче. стіл порівнює всі продажі в Інтернеті з продажами в Інтернеті за межами США, щоб показати, що фільтр вираз працює, виключаючи продажі в США з обчислення. Щоб повторно створити цю таблицю, додайте поле SalesTerritoryCountry до області «Мітки рядків» звіт або зведену таблицю.

EARLIER корисний для вкладених обчислень, де потрібно використовувати певне значення як вхідні дані та робити розрахунки на основі цих вхідних даних. Такі обчислення можна робити лише в Microsoft Excel в контексті поточного рядка; однак у DAX можна зберігати значення введення та потім зробіть розрахунок, використовуючи дані з усієї таблиці

```
DAX

EARLIER(<column>, <number>)
```

EARLIER виконується успішно, якщо перед початком сканування таблиці існує контекст рядка. Інакше він повертає помилку. Продуктивність EARLIER може бути повільною, оскільки теоретично вона може працювати кількість операцій, яка наближається до загальної кількості рядків (у стовпці), помноженої на однакове число (залежно від синтаксису виразу). Наприклад, якщо у вас 10 рядків у колонці може знадобитися приблизно 100 операцій; якщо у вас 100 рядків, тоді може бути виконано близько 10 000 операцій. Ця функція не підтримується для використання в режимі DirectQuery, коли використовується в обчисленому стовпці або правила безпеки на рівні рядків (RLS).

Приклад 1. Щоб проілюструвати використання EARLIER, необхідно побудувати сценарій, який обчислює значення рангу та потім використовує це значення рангу в інших обчисленнях. Наступний приклад базується на цій простій таблиці ProductSubcategory, яка показує загальну суму продажів для кожної підкатегорії продукту. Тут показано підсумкову таблицю, включаючи стовпець рейтингу

ProductSubcategoryKey	EnglishProductSubcategoryName	TotalSubcategorySales	SubcategoryRanking
18	Bib-Shorts	$156,167.88	18
26	Bike Racks	$220,720.70	14
27	Bike Stands	$35,628.69	30
28	Bottles and Cages	$59,342.43	24
5	Bottom Brackets	$48,643.47	27
6	Brakes	$62,113.16	23
19	Caps	$47,934.54	28
7	Chains	$8,847.08	35

Один із способів отримати значення рангу для даного значення в рядку полягає в тому, щоб підрахувати кількість рядків у та сама таблиця, яка має значення більше (або менше), ніж та, що порівнюється. Це Метод повертає порожнє або нульове значення для найвищого значення в таблиці, тоді як рівні значення матиме однакове значення рангу, а наступне значення (після рівних значень) матиме непослідовне значення значення рангу.

Новий обчислюваний стовпець SubCategorySalesRanking створюється за допомогою такої формули

```
= COUNTROWS(FILTER(ProductSubcategory, EARLIER(ProductSubcategory[TotalSubcategorySales])
```

Наступні кроки описують метод розрахунку більш детально.

1. Функція EARLIER отримує значення TotalSubcategorySales для поточного рядка в стіл. У цьому випадку, оскільки процес починається, це перший рядок у таблиці
2. EARLIER([TotalSubcategorySales]) оцінюється як $156 167,88, поточний рядок у зовнішньому петля.
3. Тепер функція FILTER повертає таблицю, у якій усі рядки мають значення TotalSubcategorySales перевищує $156 167,88 (це поточне значення для EARLIER).
4. Функція COUNTROWS підраховує рядки відфільтрованої таблиці та присвоює це значення новий обчислюваний стовпець у поточному рядку плюс 1. Додавання 1 необхідно, щоб запобігти вершині ранжоване значення з стає порожнім. 5. Формула обчисленого стовпця переходить до наступного рядка та повторює кроки 1–4. Ці кроки повторюються, доки не буде досягнуто кінця таблиці. Функція EARLIER завжди отримуватиме значення стовпця перед поточною таблицею операція. Якщо перед цим потрібно отримати значення з циклу, встановіть другий аргумент рівним 2.

INDEX - Повертає рядок в абсолютній позиції, визначеній параметром position, у межах вказаний розділ, відсортований за вказаним порядком. Якщо поточний розділ не можна вивести до одного розділу можна повернути декілька рядків.

```
DAX

INDEX(<position>[, <relation> or <axis>][, <orderBy>][, <blanks>][,
<partitionBy>][, <matchBy>][, <reset>] )
```

Кожен стовпець <partitionBy> і <matchBy> повинен мати відповідне зовнішнє значення допомогти визначити «поточний розділ», на якому потрібно працювати, з такою поведінкою: Якщо існує рівно один відповідний зовнішній стовпець, використовується його значення. Якщо відповідного зовнішнього стовпця немає: INDEX спочатку визначить усі стовпці <partitionBy> і <matchBy>, які мають немає відповідного зовнішнього стовпця. Для кожної комбінації наявних значень для цих стовпців у батьківському INDEX контексті, обчислюється INDEX і повертається рядок. Остаточним результатом INDEX є об'єднання цих рядків. Якщо існує більше ніж один відповідний зовнішній стовпець, повертається помилка. Якщо <matchBy> присутній, INDEX спробує використати стовпці <matchBy> і <partitionBy> для визначити рядок.
Якщо <matchBy> відсутній і стовпці, указані в <orderBy> і <partitionBy> не може однозначно ідентифікувати кожен рядок у <relation>: INDEX намагатиметься однозначно знайти найменшу кількість додаткових стовпців визначити кожен рядок. Якщо такі стовпці знайдено, INDEX автоматично додасть ці нові стовпці до <orderBy>, і кожен розділ сортується за допомогою цього нового набору стовпців OrderBy. Якщо такі стовпці не знайдено, повертається помилка. Порожня таблиця повертається, якщо: Відповідне зовнішнє значення стовпця PartitionBy не існує всередині <відношення>. Значення <position> посилається на позицію, яка не існує в розділі. Якщо INDEX використовується в обчислюваному стовпці, визначеному в тій самій таблиці, що й <relation> і <orderBy> пропущено, повертається помилка. <reset> можна використовувати лише у візуальних обчисленнях і не можна використовувати в поєднанні з <orderBy> або <partitionBy>. Якщо <reset> присутній, <axis> можна вказати, але <relation> не може.

Приклад 1. EVALUATE SUMMARIZECOLUMNS (FactInternetSales[ProductKey], DimDate[MonthNumberOfYear], FILTER (VALUES(FactInternetSales[ProductKey]), [ProductKey] < 222), "CurrentSales", SUM(FactInternetSales[SalesAmount]), "LastMonthSales", CALCULATE (SUM(FactInternetSales[SalesAmount]), INDEX(-1, ORDERBY(DimDate[MonthNumberOfYear])))) ORDER BY [ProductKey], [MonthNumberOfYear]

FactInternetSales[ProductKey]	DimDate[MonthNumberOfYear]	[CurrentSales]	[LastMonthSales]
214	1	5423.45	8047.7
214	2	4968.58	8047.7
214	3	5598.4	8047.7
214	4	5073.55	8047.7
214	5	5248.5	8047.7
214	6	7487.86	8047.7
214	7	7382.89	8047.7
214	8	6543.13	8047.7
214	9	6788.06	8047.7
214	10	6858.04	8047.7
214	11	8607.54	8047.7
214	12	8047.7	8047.7
217	1	5353.47	7767.78
217	2	4268.78	7767.78
217	3	5773.35	7767.78

Приклад 2. SalesComparedToBeginningOfYear = [SalesAmount] - CALCULATE(SUM([SalesAmount]), INDEX(1, ROWS, HIGHESTPARENT)) SalesComparedToBeginningOfQuarter = [SalesAmount] - CALCULATE(SUM([SalesAmount]), INDEX(1, , -1))

Покращте таблицю, щоб вона містила для кожного місяця: - загальна сума продажів; - різниця до першого місяця відповідного року; - та різниця до першого місяця відповідного кварталу.

```
1 SalesComparedToBeginningOfYear = [SalesAmount] - CALCULATE(SUM([SalesAmount]), INDEX(1, ROWS, HIGHESTPARENT))
```

CalendarYear	CalendarQuarter	MonthNumberOfYear	SalesAmount	SalesComparedToBeginningOfYear	SalesComparedToBeginningOfQuarter
2011	1	1	$323,743.2056	0.00	0.00
		2	$437,372.4476	113,629.24	113,629.24
		3	$514,918.2284	191,175.02	191,175.02
		Total	**$1,276,033.8816**	**0.00**	**0.00**
	2	4	$523,950.0258	200,206.82	0.00
		5	$539,469.0004	215,725.79	15,518.97
		6	$647,524.214	323,781.01	123,574.19

OFFSET - Повертає один рядок, який розташований перед або після поточного рядка в межах ту саму таблицю за заданим зсувом. Якщо поточний рядок не може бути виведений до одного рядка, може бути повернуто кілька рядків.

```
OFFSET ( <delta>[, <relation> or <axis>][, <orderBy>][, <blanks>][,
<partitionBy>][, <matchBy>][, <reset>] )
```

За винятком стовпців, доданих табличними функціями DAX, кожен стовпець у <relation>, коли <matchBy> відсутній або кожен стовпець у <matchBy> і <partitionBy>, коли <matchBy> присутній, має мати відповідне зовнішнє значення, щоб допомогти визначити поточний рядок, з яким потрібно працювати, з такою поведінкою: Якщо існує рівно один відповідний зовнішній стовпець, використовується його значення. Якщо відповідного зовнішнього стовпця немає, то: OFFSET спочатку визначить усі стовпці, які не мають відповідних зовнішніх колонка.
Для кожної комбінації наявних значень для цих стовпців у батьківському елементі OFFSET контексті, OFFSET обчислюється і повертається рядок. Остаточним результатом OFFSET є об’єднання цих рядків. Якщо існує більше ніж один відповідний зовнішній стовпець, повертається помилка. Якщо всі стовпці <relation> були додані табличними функціями DAX, повертається помилка

```
OFFSET ( <delta>[, <relation> or <axis>][, <orderBy>][, <blanks>][,
<partitionBy>][, <matchBy>][, <reset>] )
```

Якщо <matchBy> присутній, OFFSET спробує використати стовпці <matchBy> і <partitionBy> щоб визначити рядок. Якщо <matchBy> відсутній і стовпці, указані в <orderBy> і <partitionBy> не може однозначно ідентифікувати кожен рядок у <relation>, тоді: OFFSET намагатиметься однозначно знайти найменшу кількість додаткових стовпців визначити кожен рядок. Якщо такі стовпці можна знайти, OFFSET автоматично додасть ці нові стовпців до <orderBy>, і кожен розділ сортується за допомогою цього нового набору OrderBy колонки. Якщо такі стовпці не знайдено, повертається помилка. Порожня таблиця повертається, якщо: Відповідне зовнішнє значення стовпця OrderBy або PartitionBy не існує у межах <відношення>. Значення <delta> викликає зсув до рядка, якого немає в розділі. Якщо OFFSET використовується в обчислюваному стовпці, визначеному в тій же таблиці, що й <relation>, і <orderBy> пропущено, повертається помилка. <reset> можна використовувати лише у візуальних обчисленнях і не можна використовувати в поєднанні з <orderBy> або <partitionBy>. Якщо <reset> присутній, <axis> можна вказати, але <relation> не може.

Приклад

```
DEFINE VAR vRelation = SUMMARIZECOLUMNS ( DimProductCategory[EnglishProductCategoryName], DimDate[CalendarYear], "CurrentYearSales", SUM(FactInternetSales[SalesAmount]) ) EVALUATE ADDCOLUMNS ( vRelation, "PreviousYearSales", SELECTCOLUMNS( OFFSET ( -1, vRelation, ORDERBY([CalendarYear]), PARTITIONBY([EnglishProductCategoryName])
```

Повертає таблицю, яка підсумовує загальні продажі для кожної категорії продукту та календарного року, а також загальні продажі для цієї категорії за попередній рік. Наступний запит DAX: DAX використовує OFFSET() у вимірюванні, щоб повернути таблицю, яка підсумовує загальні продажі за кожен календарний рік і загальні продажі за попередній рік. Наступний запит DAX: DAX), [CurrentYearSales]))

Приклад

```
DEFINE MEASURE DimProduct[CurrentYearSales] = SUM(FactInternetSales[SalesAmount]) MEASURE DimProduct[PreviousYearSales] = CALCULATE(SUM(FactInternetSales[SalesAmount]), OFFSET(-1, , ORDERBY(DimDate[CalendarYear]))) EVALUATE SUMMARIZECOLUMNS ( DimDate[CalendarYear], "CurrentYearSales", DimProduct[CurrentYearSales], "PreviousYearSales", DimProduct[PreviousYearSales] )
```

Використовує OFFSET() у вимірюванні, щоб повернути таблицю, яка підсумовує загальні продажі для кожного календарний рік і загальний обсяг продажів за попередній рік.

```
EVALUATE
ADDCOLUMNS (
 FactInternetSales,
 "Previous Sales Amount",
 SELECTCOLUMNS (
 OFFSET (
 -1,
 FactInternetSales,
 ORDERBY ( FactInternetSales[SalesAmount], DESC ),
 PARTITIONBY ( FactInternetSales[ProductKey] ),
 MATCHBY( FactInternetSales[SalesOrderNumber],
FactInternetSales[SalesOrderLineNumber] ) ), FactInternetSales[SalesAmount] ) )
```

Повертає таблицю FactInternetSales із додаванням стовпця, у якому для кожного продажу вказано його сума попереднього продажу того самого продукту в порядку зменшення суми продажу, с поточний продаж ідентифікується за його SalesOrderNumber і SalesOrderLineNumber. Без MATCHBY запит повернув би помилку, оскільки в ньому немає ключових стовпців Таблиця FactInternetSales.

RANK - Повертає рейтинг для поточного контексту в указаному розділі, відсортований за визначений порядок. Якщо відповідність не знайдено, то рейтинг буде порожнім.

```
DAX

RANK ( [<ties>][, <relation> or <axis>][, <orderBy>][, <blanks>][,
<partitionBy>][, <matchBy>][, <reset>] )
```

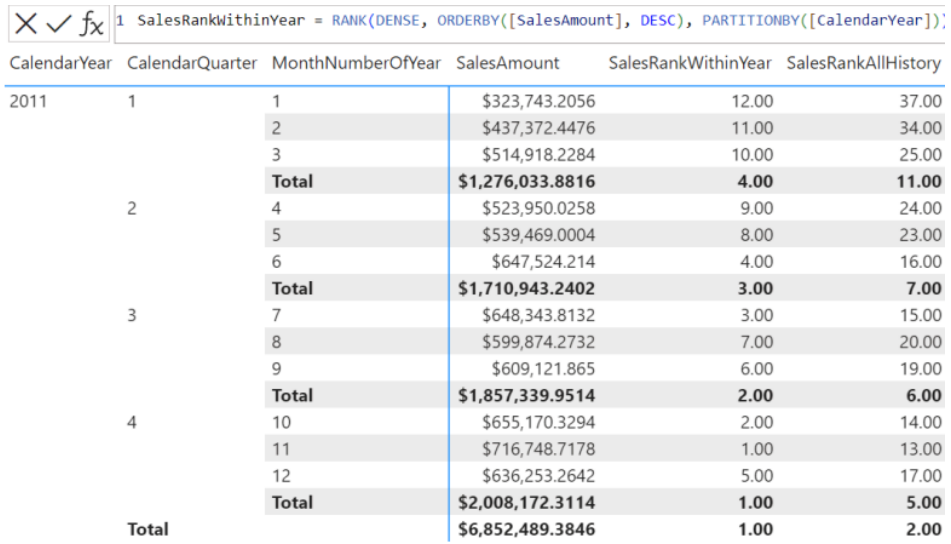
Кожен стовпець orderBy, partitionBy і matchBy повинен мати відповідний зовнішній значення, щоб допомогти визначити поточний рядок, з яким потрібно працювати, з наступним поведінка:
Якщо існує рівно один відповідний зовнішній стовпець, використовується його значення.
Повернене значення
Зауваження
Якщо відповідного зовнішнього стовпця немає, то: RANK спочатку визначить усі стовпці orderBy, partitionBy і matchBy
які не мають відповідного зовнішнього стовпця.
Для кожної комбінації наявних значень для цих стовпців у батьківському RANK контексті, оцінюється RANK і повертається рядок.
Остаточним результатом RANK є число рангу.
Якщо matchBy присутній, тоді RANK спробує використати стовпці в matchBy і
partitionBy для ідентифікації поточного рядка.
Якщо стовпці, указані в orderBy та partitionBy, не можуть однозначно ідентифікувати
кожен рядок у відношенні , то два або більше рядків можуть мати однаковий рейтинг і
рейтинг визначатиметься параметром рівних.
RANK повертає порожнє значення для підсумкових рядків. Рекомендується перевірити свій
вираз ретельно.
RANK не порівнюється з RANKX, оскільки SUM порівнюється з SUMX.
reset можна використовувати лише у візуальних обчисленнях і не можна використовувати разом
з orderBy або partitionBy . Якщо присутній скидання, можна вказати вісь, але
відношення canno

Приклад 1. EVALUATE ADDCOLUMNS('DimGeography', "Rank", RANK(DENSE, 'DimGeography', ORDERBY('DimGeography'[StateProvinceName], desc, 'DimGeography'[City], asc), LAST, PARTITIONBY('DimGeography'[EnglishCountryRegionName]))) ORDER BY [EnglishCountryRegionName] asc, [StateProvinceName] desc, [City] asc

Повертає таблицю, яка ранжує кожну географію з однаковою назвою EnglishCountryRegionName, за назвою штату та містом. Порожні значення стовпця orderBy сортуються за кінець

Приклад2. SalesRankWithinYear = RANK(DENSE, ORDERBY([SalesAmount], DESC), PARTITIONBY([CalendarYear])) SalesRankAllHistory = RANK(DENSE, ORDERBY([SalesAmount], DESC))

Створіть два стовпці, які класифікують кожен місяць за загальним обсягом продажів протягом кожного року та всю історію



```
1 SalesRankWithinYear = RANK(DENSE, ORDERBY([SalesAmount], DESC), PARTITIONBY([CalendarYear]))
```

CalendarYear	CalendarQuarter	MonthNumberOfYear	SalesAmount	SalesRankWithinYear	SalesRankAllHistory
2011	1	1	$323,743.2056	12.00	37.00
		2	$437,372.4476	11.00	34.00
		3	$514,918.2284	10.00	25.00
		Total	**$1,276,033.8816**	**4.00**	**11.00**
	2	4	$523,950.0258	9.00	24.00
		5	$539,469.0004	8.00	23.00
		6	$647,524.214	4.00	16.00
		Total	**$1,710,943.2402**	**3.00**	**7.00**
	3	7	$648,343.8132	3.00	15.00
		8	$599,874.2732	7.00	20.00
		9	$609,121.865	6.00	19.00
		Total	**$1,857,339.9514**	**2.00**	**6.00**
	4	10	$655,170.3294	2.00	14.00
		11	$716,748.7178	1.00	13.00
		12	$636,253.2642	5.00	17.00
		Total	**$2,008,172.3114**	**1.00**	**5.00**
Total			**$6,852,489.3846**	**1.00**	**2.00**