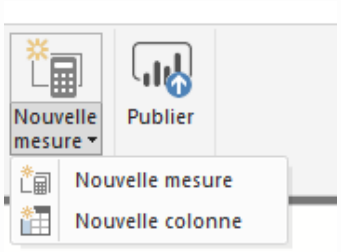
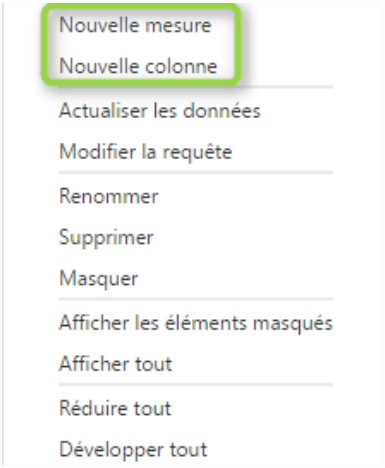
Langage DAX dans POWER BI

KAPEI
Indicateur de performance

Juin 2022

Il est possible de rajouter des informations calculées depuis les vues Rapports ou Données (onglets « Accueil » ou « Modélisation ») ou directement en effectuant un clic droit depuis le panneau des champs.





Ces informations sont :

- Soit des mesures (des informations que l'on peut agréger) ,
  - ⇒ Calculé à la volée lors de leurs utilisations dans les visualisations ,
  - ⇒ Calcul de pourcentages, taux ou agrégations complexes ,
- Soit des colonnes ,
  - ⇒ Font partie de la table dans votre modèle de données ,
  - ⇒ Utiles pour segmenter ou filtrer la valeur ou si un calcul doit se faire pour chaque ligne de la table ,

Optimisation :

- ⇒ Mesures calculées consomment du CPU ,
- ⇒ Colonnes calculées consomment de la mémoire ,

Le langage utilisé pour la création de ces champs est le langage « DAX » (Data Analysis Expression).

DAX est également utilisé pour :

- Power Pivot ,
- SSAS Tabular ,

DAX est un langage fonctionnel, comme les fonctions d'Excel. C'est un ensemble de fonctions, d'opérations et de constantes.

Il est le mix entre les fonctions Excel, le SQL et le MDX.

Exemples :

```
=SUMX(FILTER(InternetSales, InternetSales[SalesTerritoryID]=5),[Freight])
```

```
CA = SUMX(Commande_Details,Commande_Details[Quantite] * RELATED(Produits[CoutUnitaire]) )
```

Les fonctions s'appliquent selon le type de données. Deux types de données importants :

- Numérique :
    - Entier ,
    - Décimal ,
    - Monnaie ,
    - Date ,
    - Booléen ,
- Autres :
    - Texte ,
    - Objets binaire ,

Conversion du type de données implicite dans la formule (« + » va convertir les données en numérique et faire l'addition, « & » concatène les données en texte).

Nom des champs dans les formules DAX :

Si la table contient un espace : 'Table Name'[ColumnName]

Sinon : TableName[ColumnName]

Conseils :

- Ne pas mettre d’espaces dans le nom des tables ,
- Mettre toujours le nom de la table dans les formules ,

Groupe de fonctions	Agrégation
Rôle	Agrège tous les nombres dans une colonne
Syntaxe	SUM(<column>)
Commentaires	Fonctionne qu'avec des données numériques.
Exemple	SUM(Commande[Prix])
Fonctions	• SUM • AVERAGE • MIN • MAX

Groupe de fonctions	« X » Agrégation
Rôle	Retourne l'agrégation d'une expression évaluée pour chaque ligne dans une table
Syntaxe	SUMX(<table>, <expression>) table : table ou expression contenant les lignes pour lesquelles l'expression sera évaluée expression : colonne ou expression à évaluer pour chaque ligne de la table
Commentaires	Fonctionne qu'avec des données numériques. Les espaces ou textes sont ignorés
Exemple	SUMX(Commande, Commande[Prix]*Commande[Quantité])
Fonctions	• SUMX • AVERAGEX • MINX • MAXX

Relation entre les tables :

⇒ Calcul entre des valeurs qui ne sont pas dans la même table.

RELATED : permet de suivre une relation « plusieurs à 1» ,

```
Montant = SUMX(Commande, Commande[Quantité] * RELATED(Produit[Prix unitaire])
```

RELATEDTABLE : permet de suivre une relation « 1 à plusieurs » ,

```
Nombre de vente = COUNTROWS(RELATEDTABLE(Commande)
```

FILTER

Filtrer des données :

- Ajoute une condition ,
- Restreint le nombre ligne dans un table ,
- Retourne une table ,
- Prend une table en entrée ,

```
SUMX(
        FILTER(
                Commande
                Commande[Prix] > 10
        ),
        Commande[Quantity] * Commande[Prix]
)
```

CALCULATE

Evalue une expression dans un contexte modifié par les filtres spécifiés.

⇒ Définition d'une mesure et spécification des filtres.

- Modification du contexte de filtres en utilisant les filtres de la fonction ,
- Pas de fonction Calculate imbriquées ,

```
[VENTES CANAUX] = CALCULATE([Ventes], Canaux[Canal] = 1)
```

Où `Ventes = SUM([Ventes]Ventes)`

Ou

```
[VENTES CANAUX] = CALCULATE(SUM([Ventes]Ventes), Canaux[Canal] = 1)
```

ALL

- Ignore les filtres ,
- Prend une table en entrée ,
- Retourne la table en entier ,

```
SUMX(
        ALL(Commande),
        Commande[Quantity] * Commande[Prix]
)
```

```
ALL(Clients[ClientNom])
```

⇒ Retourne une table avec une unique colonne « ClientNom » comprenant toutes les valeurs unique de ce champ.

```
ALLEXCEPT(Client, Clients[ClientNom])
```

⇒ Inverse de la fonction ALL, renvoie toutes les colonnes de Client sauf « ClientNom ».

Variables :

Utile pour remplacer une partie du code qui se répète dans l'expression.

```
Quantité corrigée =
VAR
        QuantitéTotale = SUM(Commande([Quantité])
RETURN
        IF(QuantitéTotale > 1000,
                QuantitéTotale * 0,95,
                QuantitéTotale * 1,25
        )
```

Groupe de fonctions	Logique
Syntaxe / Fonctions	AND => && OR => || NOT => ! IF IFERROR SWITCH FALSE TRUE

Groupe de fonctions	Information
Syntaxe / Fonctions	CONTAINS CUSTOMDATA ISBLANK ISNUMBER ISTEXT ISNONTEXT ISERROR ISEVEN ISLOGICAL ISONORAFTER USERNAME LOOKUPVALUE

Groupe de fonctions	Mathématique
Syntaxe / Fonctions	Exemple : ABS,EXPR, LOG, LOG10, MOD, PI, POWER, QUOTIENT, SIGN, SQRT

Groupe de fonctions	Texte
Syntaxe / Fonctions	BLANK, CODE, CONCATENTATE, CONCATENTATEX, EXACT, FIND, FIXED, FORMAT, LEFT, LEN, LOWER, MID, REPLACE, REPT, RIGHT, SEARCH, SUBSTITUTE, TRIM, UPPER, VALUE, UNICHAR

Groupe de fonctions	Date
Syntaxe / Fonctions	CALENDAR, CALENDARAUTO, DATE, DATEDIFF, DATEVALUE, DAY, EDATE, EOMONTH, HOUR, MINUTE, MONTH, NOW, SECOND, TIME, TIMEVALUE, TODAY, WEEKDAY, WEEKNUM, YEAR, YEARFRAC

Groupe de fonctions	« Time Intelligence »
Commentaires	Nécessite un champ de format date
Syntaxe / Fonctions	CLOSINGBALANCEMONTH, CLOSINGBALANCEQUARTER, CLOSINGBALANCEYEAR, DATEADD, DATESBETWEEN DATESMTD, DATESQTD, DATESYTD, ENDOFMONT, ENDOFQUARTER, ENDOFYEAR, FIRSTDATE, FIRSTNOTBLANK, NEXTDAY, NEXTMONT, NEXTQUARTER, NEXTYEAR, OPENNINGBALANCEMONTH, OPENNINGBALANCEQUARTER, OPENNINGBALANCEYEAR, PARALLELPERIOD, PREVIOUSDAY, PREVIOUSMONTH, PREVIOUSQUARTER, PREVIOUSYEAR, SAMEPERIODLASTYEAR, STARTOFMONTH, STARTOFQUARTER, STARTOFYEAR, TOTALMTD, TOTALQTD, TOTALYTD

Comparaison temps : **SAMEPERIODLASTYEAR**

Retourne une table contenant une colonne de dates sur une la même période l'année précédente :

```
SAMEPERIODLASTYEAR(<dates>)
```

- <dates> = champ au format date d'une table ,

Exemple : Ventes de l'années précédente :

```
[VENTES ANNEE N-1] = CALCULATE( SUM([Ventes]Ventes), SAMEPERIODLASTYEAR (VENTE[DATE])
```

Comparaison temps : **PARALLELPERIOD**

Retourne une table contenant une colonne de dates sur une période parallèle aux dates spécifiées :

```
PARALLELPERIOD(<dates>,<number_of_intervals>,<interval>)
```

- <dates> = champ au format date d'une table ,
- <number_of_intervals> = nombre d'intervalle a ajouter ou soustraire de la date ,
- <interval> = partie de la date à ajouter ou soustraire : « YEAR », « QUARTER », « MONTH » ,

Exemple : Ventes de l'années précédente :

```
[VENTES ANNEE N-1] = CALCULATE( SUM([Ventes]Ventes), PARALLELPERIOD (VENTE[DATE], -1, YEAR)
```

Comparaison temps : **DATEADD**

Retourne une table contenant une colonne de dates sur une période parallèle aux dates spécifiées :

```
DATEADD(<dates>,<number_of_intervals>,<interval>)
```

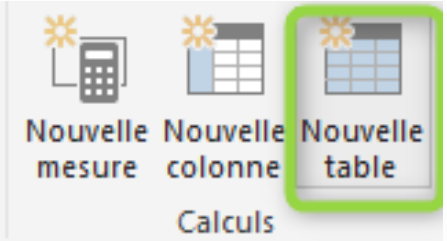
- <dates> = champ au format date d'une table ,
- <number_of_intervals> = nombre d'intervalle a ajouter ou soustraire de la date ,
- <interval> = partie de la date à ajouter ou soustraire : « YEAR », « QUARTER », « MONTH », « DAY » ,

Exemple : Ventes de l'années précédente :

```
[VENTES ANNEE N-1] = CALCULATE( SUM([Ventes]Ventes), DATEADD(VENTE[DATE], -365, DAY)
```

Il est possible de rajouter une table avec le langage DAX :

- Cette table est un sous ensemble d'une table précédemment chargée ,
- Elle appartient au modèle et peut donc avoir des relations avec les autres tables ,



Utilités :

- Filtrer une table ,
- Tables d'agrégats ,
- Dimension temps ,
- Table passerelle pour une relation plusieurs à plusieurs ,
- …

Exemple : Utilisation de la fonction « UNION » pour ajouter des nouvelles données

```
SEGMENTS_UNIONROWS =
UNION (
    ROW ( "PRICE RANGE", "LOW", "MIN PRICE", CURRENCY ( 0 ), "MAX PRICE", CURRENCY ( 10 ) ),
    ROW ( "PRICE RANGE", "MEDIUM", "MIN PRICE", 10, "MAX PRICE", 100 ),
    ROW ( "PRICE RANGE", "HIGH", "MIN PRICE", 100, "MAX PRICE", 9999999 )
)
```

Exemple : Utilisation de la fonction « DATATABLE » pour ajouter des nouvelles données

```
Segments_Datatable =
DATATABLE (
    "Price Range", STRING,
    "Min Price", CURRENCY,
    "Max Price", CURRENCY,
    {
        { "Low", 0, 10 },
        { "Medium", 10, 100 },
        { "High", 100, 9999999 }
    }
)
```

- Exemple : copie d'une table

```
[HISTORIQUE DES VENTES] = VENTES
```

- Exemple : union de 2 tables

```
[HISTORIQUE DES VENTES] = UNION(VENTES_2017, VENTES_2016)
```

- Exemple : valeurs distincts d'une table

```
[VENDEUR] = DISTINCT('VENTES'[VENDEUR])
```

- Exemple : Regroupement d'informations

SUMMARIZE(<table>, <grouBy Column>, <name>, <expression>)

- Table : nom de la table où doit se faire le group by
- groupBy Column : nom de la colomme à grouper
- name : nom de la colonne
- expression : l'agrégation à effectuer

```
[Product Sales = SUMMARIZE(Sales,'Product'[ProductKey],"Total Sales« ,Sum('Product'[SalesAmount]))
```

- Exemple : n premières valeurs

```
[TOP 10 PRODUITS] = TOPN(10, 'VENTES', 'VENTES'[TOTAL VENTES], DESC)
```

Génération d'un calendrier :

⇒ Initie l'ensemble des dates et ajout des autres champs via l'ajout de colonnes dans la modélisation :

- CALENDAR(Date_début , Date_Fin) : retourne tous jour compris entre la Date_Début et la Date_Fin.
- CALENDARAUTO() : retourne tous jour compris entre la date min et max du modèle.

⇒ Tout en DAX :

```
Date =
VAR BaseCalendar = CALENDARAUTO()
RETURN ADDCOLUMNS (
    BaseCalendar,
    "Year", YEAR ( [Date] ),
    "Month Number", MONTH ( [Date] ),
    "Month", FORMAT ( [Date], "mmmm" ),
    "Year Month Number", YEAR ( [Date] ) * 12 + MONTH ( [Date] ) - 1,
    "Year Month", FORMAT ( [Date], "yyyy mmm" ))

Date =
VAR BaseCalendar = CALENDARAUTO()
RETURN GENERATE (
        BaseCalendar,
        VAR BaseDate = [Date]
        VAR YearDate = YEAR ( BaseDate )
        VAR MonthNumber = MONTH ( BaseDate )
        VAR YearMonthNumber = YearDate * 12 + MonthNumber - 1
        RETURN ROW (
            "Day", BaseDate,
            "Year" , YearDate,
            "Month Number" , MonthNumber,
            "Month" , FORMAT ( BaseDate, "mmmm" ),
            "Year Month Number" , FORMAT ( BaseDate, « yyyy mm" ),
            "Year Month" , FORMAT ( BaseDate, « yyyy mmm" )
        )
    )
```