DAX

DATE TIME DAX FUNCTION

- CALENDARAUTO() Detecta de manera inteligente las fechas que tenga en el modelo y me hará un calendario completo de las fechas correspondientes.

Entrando en la parte de vista de tabla, creo una nueva tabla y pongo este script:

```
1  Tabla = CALENDARAUTO()
```

- Calendar (“fecha_inicio” , “fecha_fin”) Me permite crear un calendario donde yo le indicaré el inicio y fin del mismo.

Entrando en la parte de vista de tabla, creo una nueva tabla y pongo este script:

```
1  Tabla = CALENDAR("1/1/2024" , "28/2/2024")
```

- DATE(Año,Mes,Día) Se puede crear una columna con esta función donde le indiquemos una fecha que queramos utilizar de referencia. El 1 indica que tomaremos día 1.

```
1  Columna = DATE(YEAR(PizzaSales[Date]), Month(PizzaSales[Date]), 1)
```

- DATEDIFF(fecha1, fecha2, intervalo) Nos permite hacer una resta desde una fecha hasta otra donde el intervalo es la unidad de media, días , horas.

```
1  Columna 2 = DATEDIFF(PizzaSales[Columna],PizzaSales[Date],HOUR)
```

- DATEVALUE(DateText) Nos permite, tomando los datos de otras columnas hacer una concatenación de las fechas para mostrarlas en una sola columna.

```
1  Columna 3 = DATEVALUE(Sheet[Day] & "/" & Sheet[Month] & "/" & Sheet[Year])
```

- NOW() and TODAY() Nos proporcionará el momento y la fecha en la que se ejecuta.

```
1  Columna = NOW()
```

```
1  Columna = TODAY()
```

- DAY, MONTH, YEAR(Table[columna_date]) Nos permite indicando los parámetros extraer el día, mes o año de una columna date.

```
1  Columna = MONTH(PizzaSales[Date])
```

- EDATE(StartDate, Months) Hace un conteo de meses a partir de la fecha que se le da como referencia, puede hacerlo de manera positiva como negativa.

```
1  Columna = EDATE(PizzaSales[Date],2)
```

- EOMONTH(StartDate, Months) Retorna el último día del mes indicado donde StarDate es la columna DATE de referencia y si ponemos 0 es para no agregar meses , si se remplaza es para agregarlos.

```
1  Columna = EOMONTH(PizzaSales[Date],0)
```

- NETWORKDAYS(StartDate,EndDate, [weekend], [holidays]) Nos permite calcular el número de días laborables una vez le hemos dado los parámetros necesarios. En Weekend se pone 1 que representa sábado y domingo y holidays deberíamos tener una columna creada con los festivos a considerar.

```
1  Columna = NETWORKDAYS("25/11/2022",PizzaSales[Date],1,Holidays)
```

- WEEKDAY(Date, [ReturnType]) Retorna el número del día pero hay que indicar el ReturnType = 2 para que la semana comience el lunes como el día 1.

```
1  Columna = WEEKDAY(PizzaSales[Date],2)
```

LOGICAL FUNCTIONS IN DAX

- IF(LogicalTest, ResultIfTrue, [ResultIfFalse]) Evalúa la respuesta lógica y da salida si se cumple o no.

```
Columna 2 = if(PizzaSales[Sales] > 100, "Top Seller", "Average Seller")
```

- IFERROR (Value, ValueIfError) Si encuentra un error en la fila que está evaluando muestra el valor que le digamos.

```
Columna 4 = IFERROR(PizzaSales[Columna 3],0)
```

- AND (Logical1, Logical2) Evalúa dos condiciones dando como resultado True o False según sea el caso.

```
Columna 5 = AND(PizzaSales[Pizza_Name]="Deluxe",PizzaSales[Category] = "Non-Veg")
```

Utilizando && nos permite hacer más de una comparación.

```
Columna 5 = IF(PizzaSales[Sales] >=10 && PizzaSales[Columna 2] = "Top Seller" && PizzaSales[Unsold] < 10 , "Star","Budget")
```

- OR (Logical1, Logical2) Evalúa dos condiciones dando como resultado True o False según sea el caso.

```
Columna 5 = OR(PizzaSales[Pizza_Name]="Deluxe",PizzaSales[Category] = "Non-Veg")
```

Utilizando || nos permite hacer más de una comparación.

```
Columna 5 = IF(PizzaSales[Sales] >=10 || PizzaSales[Columna 2] = "Top Seller" || PizzaSales[Unsold] < 10 , "Star","Budget")
```