Power BI - Datové modelování

Datové modelování

- Základní proces analýzy dat
- Pomocí diagramu se vizuálně vyjádří vztah mezi jednotlivými daty, která jsou pro analýzu shromažďována.
- Správný datový model zjednodušuje práci s daty a umožňuje dosáhnout správných výsledků
- Zlepšuje se performance reportu
- Je možné se vyvarovat složitým DAX funkcím, které by mohly report zpomalovat nebo vracet nejednoznačné výsledky
- Model je udržitelný a snadno rozšiřitelný o další data (tabulky)
- Často jde o úplně první krok v analýze a stačí nám k němu tužka a papír
- Datový model musí být jednoznačný

Vazby

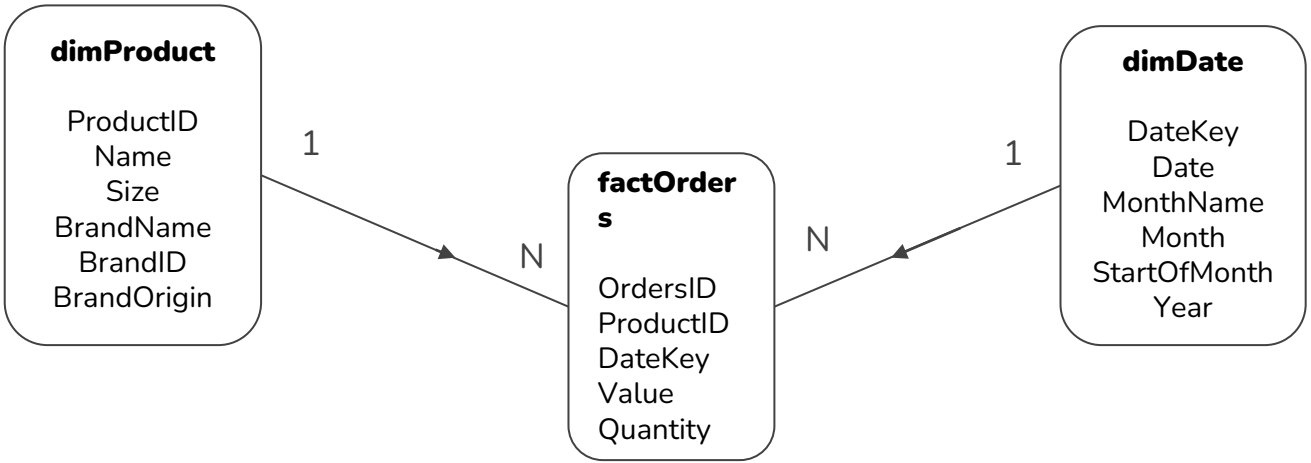
- Rozlišujeme dva typy tabulek
  dimenzní (číselníky), vždy by měly mít unikátní identifikátor pro každý řádek
  faktové
- Typy vazeb
  1:1 (one to one) každá položka je v tabulce právě jednou (produkt, ceník produktů)
  1:N / **1:*** (one to many) každá položka může mít v tabulce více výskytů (produkt, ceník produktů v jednotlivých letech)
  N:N / ***:*** (many to many) Jedna nebo více položek může mít v tabulce více výskytů (tabulka knih v knihovně s autory, která bude mít vazbu na tabulku autorů a jejich knih - autor má více výskytů, protože napsal více knih a zároveň jsou knihy, které může napsat více autorů)
- Směr vazeb
  Jednostranná Jedna tabulka filtruje jednotlivé záznamy v druhé tabulce (typicky dimenzní tabulka filtruje dané záznamy z factové tabulky, druhá tabulka pak ale už nefiltruje tabulku první
  Oboustranná obě tabulky se filtrují navzájem, filtrování vždy probíhá pomocí sloupců přes které je vazba vytvořená

Denormalizace a Normalizace

- Denormalizace i Normalizace jsou postupy, které se používají při vytváření různých modelů
- **Denormalizace -** přidávání redundantních (opakujících se dat) do tabulek v modelu pro účely rychlejšího a jasnějšího prohlížení dat
  (v databázích může jít o materializované tabulky, nebo pohledy (views), které usnadňují analytickou práci tak, aby nebylo nutné na sebe pokaždé napojovat různé tabulky).
- Denormalizovaný model by tedy mohl vypadat jako jedna tabulka, to má ale určité nevýhody
- **Normalizace** je pak proces, kdy naopak tvoříme malé tabulky, snižujeme redundaci dat a zlepšujeme jejich integritu a udržitelnost. Využíváme především pokud máme velké objemy faktových dat.
- Většinou používáme oboje a záleží na jednotlivých případech

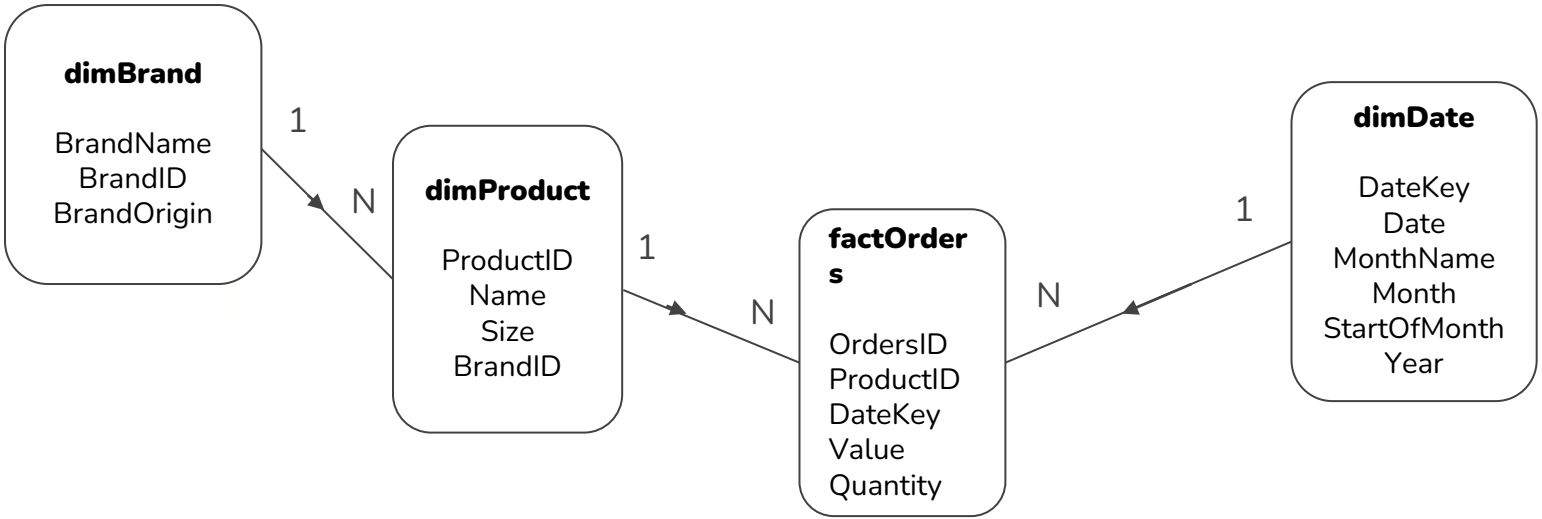
Typy datových modelů - Star Schema

- jde o best practice model, který zajišťuje nejoptimálnější řešení. Pokud to je možné, chceme vždy vytvořit star schema. Model se snadno udržuje a rozšiřuje. Je to model na který bylo Power BI optimalizováno



Typy datových modelů - Snowflake Schema

- Snowflake vzniká normalizací star schema, pokud není nezbytně nutné, tak je vždy na uvážení, jestli je snowflake schéma potřeba
- Kdy je vhodné vytvářet nové dimenze?



Typy Datových modelů

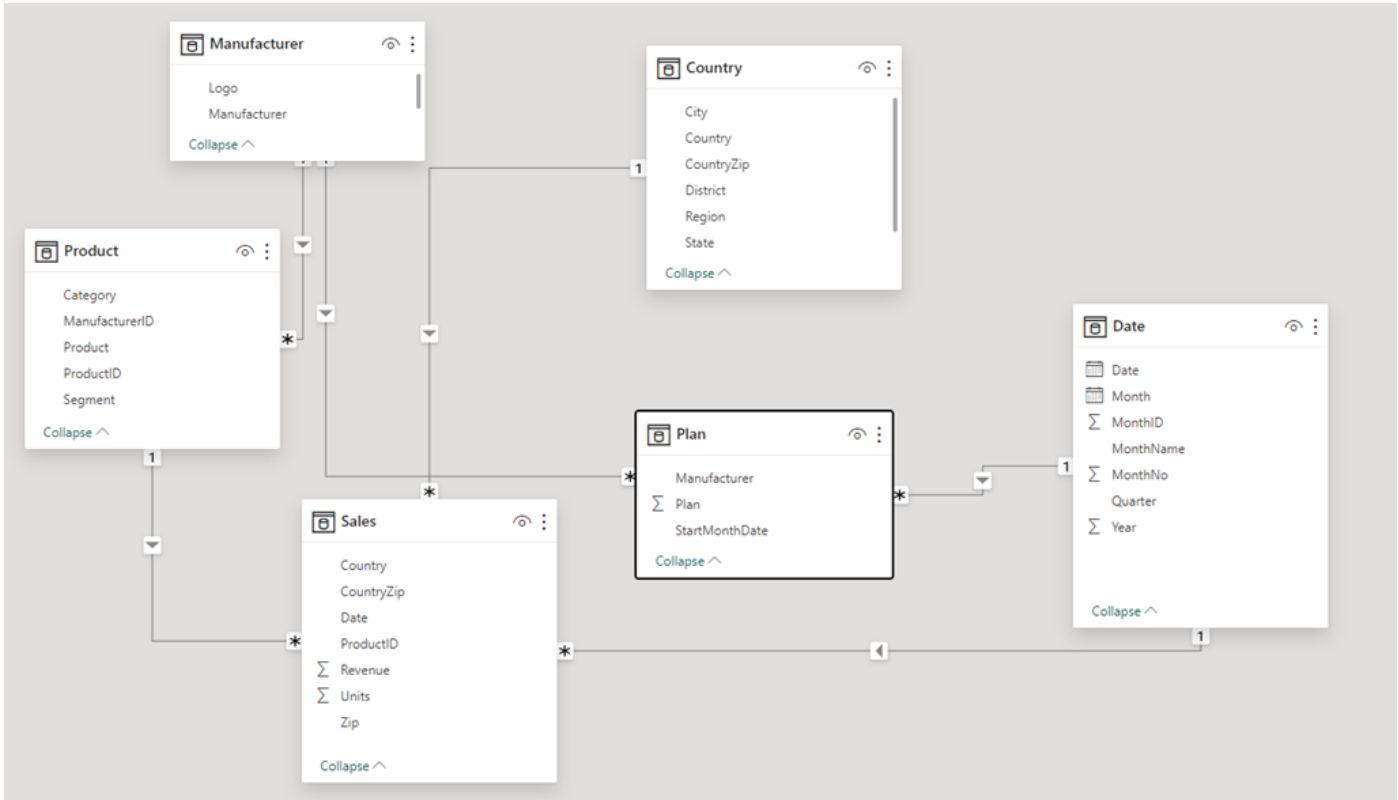
- **Více star schémat**
- **Header - Detail datový model**

- Více vazeb mezi dvěmi tabulkami
  Aktivní a neaktivní vazba
  Vazby se dají vytvářet i dynamicky (pouštět) pomocí DAX funkcí, ale vždy musí existovat

- **Ambiguility** = Mnohoznačný model

- Praktický tip - možnost skrývání sloupců v datovém modelu

Van Arsdel Datový model



- Co jsou dimenze a co fakta?
- Šly by nějaké tabulky normalizovat nebo denormalizovat?
- O jaký typ modelu se jedná?
- Máme v modelu redundantní data?

Power BI - Vizualizace, vzorce a kontexty

DAX

- Power Pivot, Tabular, Power BI
- DAX počíta hodnoty na základe stĺpcov
  - `Sales[Units]`

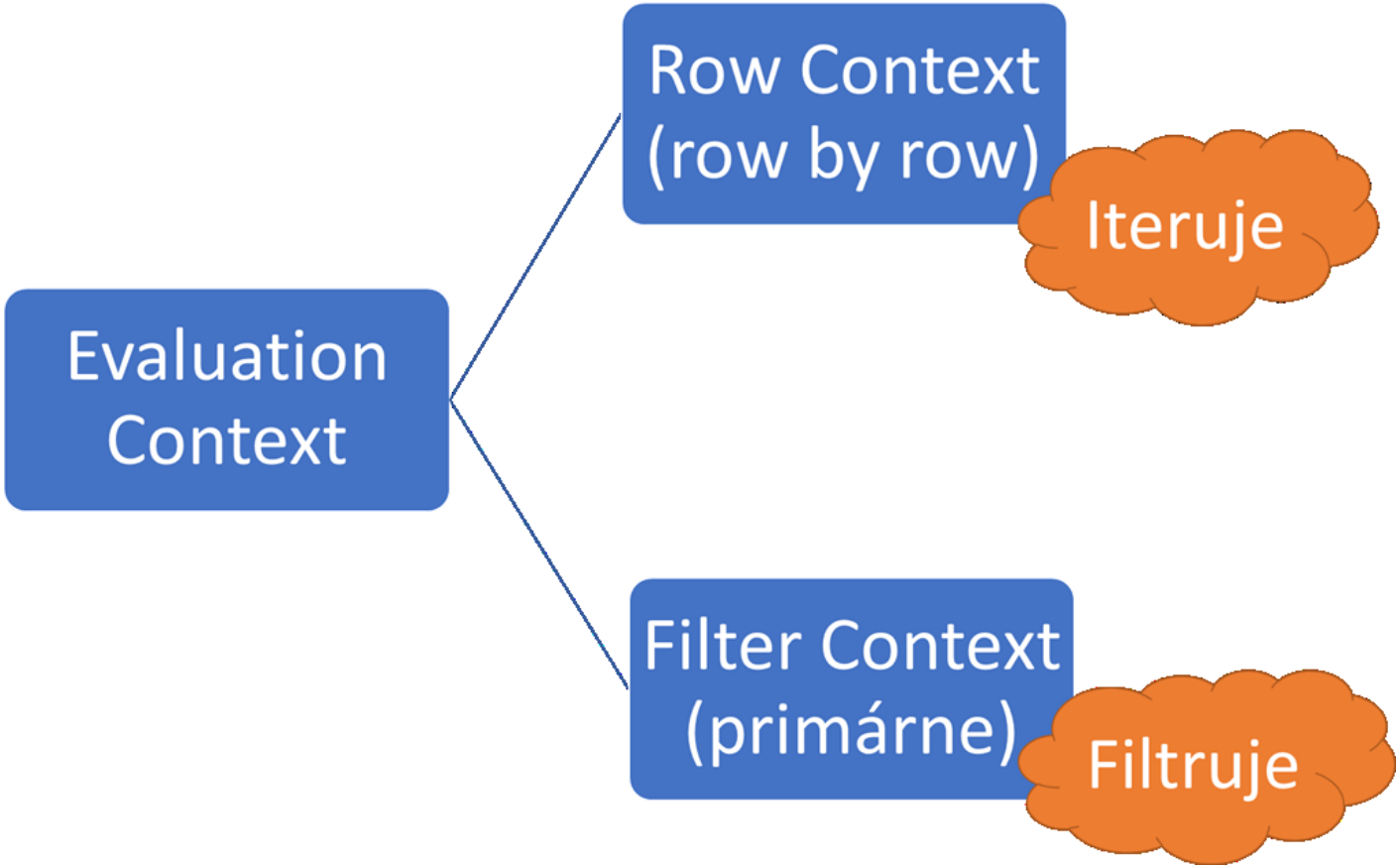
Calculated Columns vs Measures

Calculated Column

- Vyhodnocuje sa v kontexte aktuálneho riadku
- Zaťažuje RAM
- Komplexné nápočty ideálne počítať v jednom kroku
- `Sales[Units] * 2`
- Kategorizácia, Slicer, striktné riadkové výpočty (Price * Quantity)

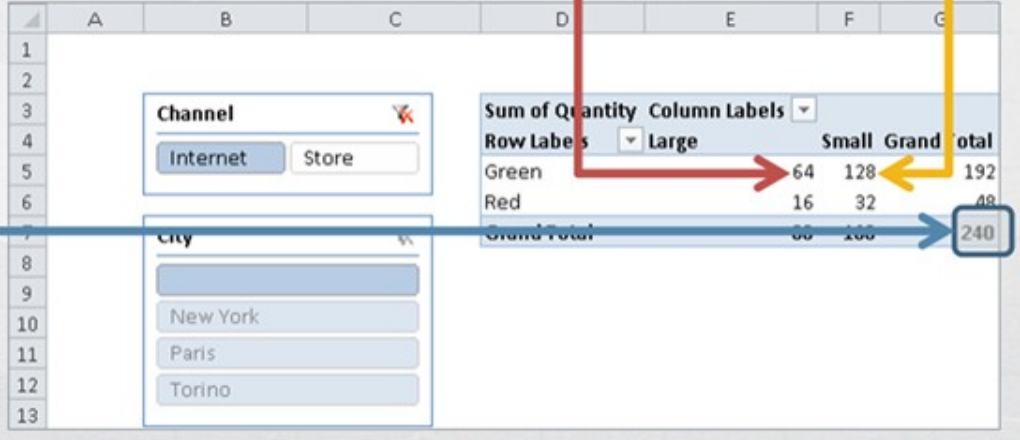
Measure

- Nevyhodnocuje sa v kontexte riadku, ale v rámci agregácie
- Zaťažuje CPU
- Vyhodnocuje sa až vtedy, keď sa použije
- `[Sales Amount] * 2`
- Percentuálne výpočty (marža) alebo ak chceme použiť napr. filter z dvoch rôznych tabuliek


Evaluation Context
Row Context (row by row)
Iteruje
Filter Context (primárne)
Filtruje

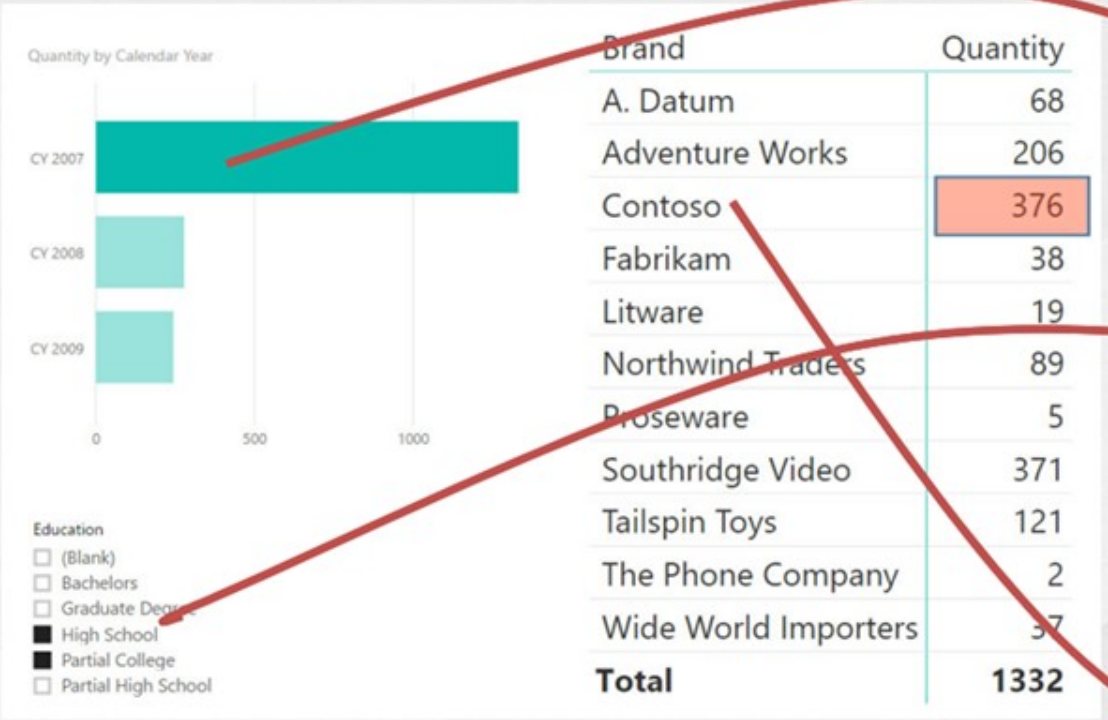
Example of a filter context

City	Channel	Color	Size	Quantity	Price
Paris	Store	Red	Large	1	15
Paris	Store	Red	Small	2	13
Torino	Store	Green	Large	4	11
New York	Store	Green	Small	8	9
	Internet	Red	Large	16	7
	Internet	Red	Small	32	5
	Internet	Green	Large	64	3
	Internet	Green	Small	128	1

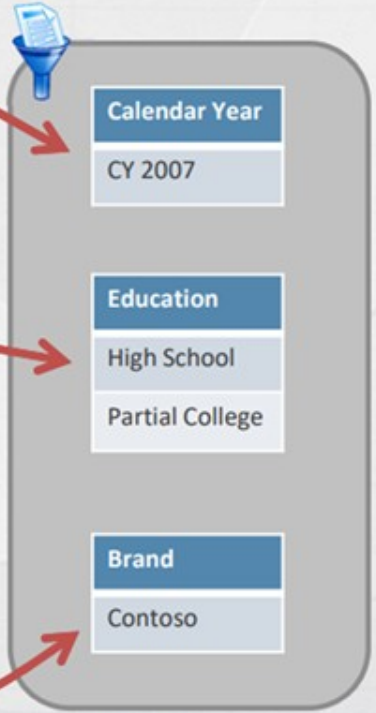


sqlbi

Filter context in Power BI



Brand	Quantity
A. Datum	68
Adventure Works	206
Contoso	376
Fabrikam	38
Litware	19
Northwind Traders	89
Proseware	5
Southridge Video	371
Tailspin Toys	121
The Phone Company	2
Wide World Importers	37
Total	**1332**



sqlbi

ROW Kontext

- Iteruje
- Automaticky u počítaných stĺpcov
- Všetky X funkcie = SUMX, AVERAGEX

- 1. Najskôr sa použije filter context (vizuál, slicer...)
- 2. Potom sa aplikuje row context

Calculated Column

- Napr. `COUNTROWS(filter(Sales,Sales[Units]>50))`
- Jeden row context existuje z dôvodu, že ide o calculated column (automaticky, neviditeľný)
- Druhý row context vytvára iterátor FILTER (viditeľný)
- Najskôr filtujem počet produktov (vnútorný kontext) a potom spočítam, koľko riadkov zodpovedá tomuto kritériu (vonkajší kontext)
- Výsledok bude rovnaké číslo na každom riadku

SUMX a FILTER

```
SUMX(FILTER(Sales,Sales[Units]>50),Sales[Revenue]/Sales[Units]
)
```

Najskôr sa použije filter z vizuálov, prípadne tabuľky, potom FILTER a až potom funkcia SUMX

FILTER nemení filter kontext, ale pridá ďalšie filtre

FILTER je tabuľková funkcia, tzn. Vezme tabuľku, prefiltruje a vráti tabuľku

SUMX nemení filter kontext, ten je stále použitý, pokiaľ ho explicitne nezmeníme.

Filter = Sales[Units]>50 => vezme tabuľku, vyberie záznamy, kde platí, že cena je vyššia, než 50 a vráti tabuľku, nad ktorou sa uskutoční výpočet Revenue/ Units

Funkcia ALL v SUMX

- Funkcia ALL ignoruje filter context = vráti vždy celú tabuľku, takže SUMX iteruje celú tabuľku

- !!!Pozor!!! Filter kontext samotný sa nemení, celá “kalkulácia“ má stále ten istý kontext, ako na začiatku

CALCULATE

- Môže zahŕňať akúkoľvek DAX funkciu ako prvý argument
- V druhom a ďalších argumentoch môžeme pomocou filtrovacích podmienok zmeniť filter context (kde bude funkcia vyhodnotená)
- POZOR!!! Každý argument po prvom argumente je vlastne tabuľka
- Môže existujúci filter modifikovať, pridať, nahradiť alebo odstrániť existujúce filtre vo filter contexte

Calculate

```
CALCULATE(SUM(Sales[Revenue]), Sales[Units] > 50)
```

je ekvivalent

```
CALCULATE(sum(Sales[Revenue]),
Filter(all(Sales[Units]),Sales[Units] > 50))
```

- Aj keď sme nenapísali FILTER(ALL()), použije sa vždy!!!

ALL vo funkcií Calculate

- ALL v druhom argumente sa stáva z tabuľkovej funkcie filtrom
- Ak do ALL nedáme žiadny stĺpec (len tabuľku), tak zrušíme filter zo všetkých stĺpcov danej tabuľky
- Ak chceme zrušiť filter len u dvoch stĺpcov, musíme oba napísať
- Môžeme použiť ALL viacnásobne, ale nedá sa odkazovať na stĺpce z dvoch rôznych tabuliek

ALLSELECTED

- Vracia kompletný výber mimo vizuál
- Vracia hodnoty všetkých riadkov tabuľky alebo stĺpca, pričom ignoruje akýkoľvek filter aplikovaný vnútri query ale zachováva filtre zvonka
- V prípade, že nechcem percento z celkových predajov, ale z predajov danej kategórie

```
CALCULATE(SUM(Sales[Revenue]),ALL(Sales))
```

vs

```
CALCULATE(sum(Sales[Revenue]),ALLSELECTED(Sales))
```

Time Intelligence Funkcie

- Funkcie pracujúce s dátumom
- Pre správne fungovanie potrebujú separátnu dátumovú tabuľku
- V prípade, že nemáme separtánu dátumovú tabuľku, dá sa použiť funkcia CALENDAR alebo CALENDARAUTO

CALENDAR a CALENDARAUTO

- U funkcie CALENDAR je potrebné stanoviť hornú a spodnú hranicu dátumov

- ```
  Calendar = CALENDAR(min(Sales[Date]), max(Sales[Date]))
  ```
- ```
  Calendar = CALENDAR(date(year(min(Sales[Date])),1,1),
  Date(YEAR(max(Sales[Date])),12,31))
  ```

- CALENDARAUTO skenuje všetky stĺpce s dátumami (mimo calculated columns)
- Treba si dať pozor, aby sme neťahali veľa dát
- Generuje dátumy od 1.1. prvého roku po 31.12. posledného roku automaticky

- ```
  Calendar = CALENDARAUTO()
  ```
  ```

Time Intelligence Funkcie

- `Revenue YTD = TOTALYTD(sum(Sales[Revenue]),'Date'[Date])`

- ```
  Revenue Same Period LY =
  Calculate(sum(Sales[Revenue]),SAMEPERIODLASTYEAR('Date'[Date]))
  ```

- ```
  Revenue DateAdd LY =
  Calculate(sum(Sales[Revenue]),DATEADD('Date'[Date],-1,YEAR))
  ```