Indexing in Power BI

Indexing in Power BI provides a layer of abstraction that can be advantageous when dealing with changing data or evolving business requirements.

Advantages

1. **Flexibility in Data Sources**
   By establishing relationships and indexing, you can create a more flexible data model. This allows you to switch data sources or update the underlying data without affecting the report visuals. When column names or data structures change in the source, you often only need to update the data transformation steps in Power Query rather than modifying every report and visualization.

2. **Centralized Management**
   Indexing and relationships promote centralized data management. If category names or other attributes change, you can adjust them in one place within the data model, and the changes are automatically propagated to all relevant visuals and reports.

3. **Data Consistency**
   Relationships ensure that data remains consistent throughout the report. For instance, if you change a category name in one location, it will automatically reflect across all related visuals and calculations. This consistency minimizes the risk of errors due to data inconsistencies.

4. **Specific situations**

Filtering	•Indexing is especially useful when you need to filter or slice data by specific columns frequently. It accelerates filtering operations, allowing users to interact with reports more efficiently.
Sorting	•If you require sorted data for tables or visuals, indexing can expedite the sorting process, leading to quicker report rendering.
Aggregations	•In cases where you perform aggregation operations (e.g., SUM, AVERAGE), indexing can enhance the speed of these calculations.
Join Operations	•When working with multiple tables and performing join operations, indexing on columns used for joining can significantly improve performance.
Lookup Tables	•Indexing is particularly valuable for lookup tables that are frequently used to filter, categorize, or categorize data in reports.
Time Intelligence Functions	•If your data involves time-based analysis and you use time intelligence functions (e.g., TOTALYTD, SAMEPERIODLASTYEAR), indexing can optimize these calculations.
Distinct Count Measures	•When calculating distinct counts of values, indexing can speed up these calculations for large datasets.

Disadvantages

1. **Model Complexity**: Creating many relationships and indexes can lead to increased model complexity. This can make your data model harder to understand, maintain, and troubleshoot, particularly for complex datasets.

2. **Increased Memory Usage**: Indexes consume memory in the Power BI model. Creating too many indexes on large tables can lead to high memory usage, potentially causing performance issues and limiting the report's scalability.

3. **DAX Complexity**: Using relationships and indexed columns in DAX calculations may introduce complexity to your measures, which could be challenging to manage.

4. **Dependency on Data Structure**: While relationships provide flexibility, they also make reports dependent on the underlying data structure. Changes to data structures may necessitate adjustments in the Power Query transformation steps or in the data model.

Conclusion

To mitigate these disadvantages, it's crucial to strike a balance between optimizing for performance and maintaining a manageable, well-structured data model.

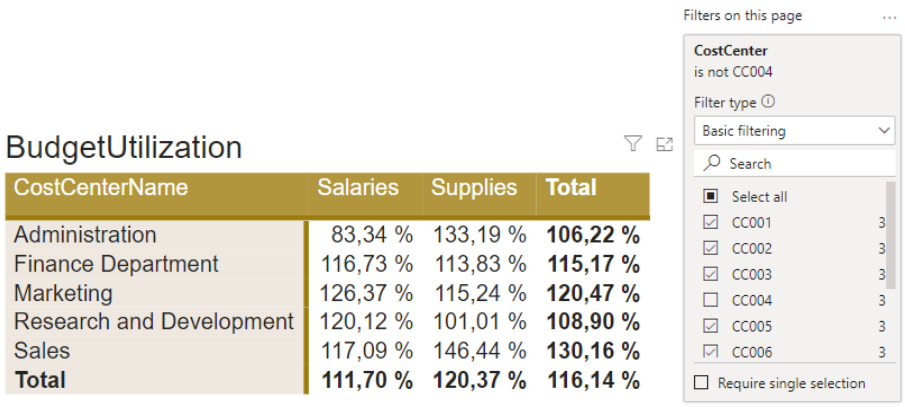
Showcase

a. Manual index

We have a manually created index for cost centers and cost units

CostCenter	CostCenterName	CostUnit	CostUnitName
CC001	Administration	CU001	Administrative Team
CC001	Administration	CU002	HR Department
CC001	Administration	CU003	Legal Department
CC002	Sales	CU004	Sales Team North
CC002	Sales	CU005	Sales Team South
CC002	Sales	CU006	Sales Support
CC003	Research and Development	CU007	R&D Team 1
CC003	Research and Development	CU008	R&D Team 2
CC003	Research and Development	CU009	Innovation Lab
CC004	IT Department	CU010	IT Support
CC004	IT Department	CU011	Systems Administration
CC004	IT Department	CU012	Software Development
CC005	Marketing	CU013	Marketing Campaigns
CC005	Marketing	CU014	Digital Marketing
CC005	Marketing	CU015	Branding and Design
CC006	Finance Department	CU016	Accounting
CC006	Finance Department	CU017	Financial Planning
CC006	Finance Department	CU018	Treasury

In this example, IT Department shouldn't be included in the visual "BudgetsUtilization".
To achieve this in a dynamic way, we filter the index of the entry. Even if the name changes in the future, it wouldn't have any impact on this report.



b. Automatic index (Power Query)

Another way to use indexes is to insert a column in Power Query with the function Table.AddIndexColumn.

= Table.AddIndexColumn(#"Changed Type", "Index", 0, 1, Int64.Type)

	Date	CostCenter	CostUnit	ExpenseCategory	ActualAmount	Period	Index
	Valid 100 % Error 0 % Empty 0 % 6 distinct, 0 unique	Valid 100 % Error 0 % Empty 0 % 6 distinct, 0 unique	Valid 100 % Error 0 % Empty 0 % 18 distinct, 0 unique	Valid 100 % Error 0 % Empty 0 % 2 distinct, 0 unique	Valid 100 % Error 0 % Empty 0 % 276 distinct, 265 unique	Valid 100 % Error 0 % Empty 0 % 6 distinct, 0 unique	Valid 100 % Error 0 % Empty 0 % 288 distinct, 288 unique
1	01.01.2023	CC001	CU001	Salaries	890	2023-1	0
2	01.01.2023	CC001	CU001	Supplies	2231	2023-1	1
3	01.01.2023	CC001	CU002	Salaries	827	2023-1	2
4	01.01.2023	CC001	CU002	Supplies	6013	2023-1	3
5	01.01.2023	CC001	CU003	Salaries	3171	2023-1	4
6	01.01.2023	CC001	CU003	Supplies	5796	2023-1	5
7	01.01.2023	CC001	CU001	Salaries	2857	2023-1	6
8	01.01.2023	CC001	CU001	Supplies	1703	2023-1	7
9	01.02.2023	CC001	CU002	Salaries	5133	2023-2	8
10	01.02.2023	CC001	CU002	Supplies	5132	2023-2	9
11	01.02.2023	CC001	CU003	Salaries	1404	2023-2	10

In this example, we have a dataset of revenues by date. In a DAX measure, we want to filter the latest entry. We can achieve this with a simple calculate function with help of the index:

```
LastRevenue =

VAR MaxIndex =

    MAX ( 'FactActuals'[Index] )

RETURN

    CALCULATE (

        SUM ( 'FactActuals' [ActualRevenue] ),

        ( 'FactActuals' [Index] ) = MaxIndex

    )
```