[Power BI DAX functions] [Extended-cheatsheet]

1. Data Aggregation

- Sum of a column: `SUM('Table'[Column])`
- Average of a column: `AVERAGE('Table'[Column])`
- Minimum value in a column: `MIN('Table'[Column])`
- Maximum value in a column: `MAX('Table'[Column])`
- Count of rows in a table: `COUNT('Table'[Column])`
- Count of distinct values in a column: `DISTINCTCOUNT('Table'[Column])`
- Weighted average of a column: `SUMX('Table', 'Table'[Column] * 'Table'[Weight]) / SUMX('Table', 'Table'[Weight])`

2. Filtering and Slicing

- Filter a table based on a condition: `FILTER('Table', 'Table'[Column] > 100)`
- Calculate a measure based on a filter: `CALCULATE([Measure], 'Table'[Column] = "Value")`
- Filter a table based on the current context: `CALCULATETABLE('Table', 'Table'[Column] = "Value")`
- Remove filters from a table: `ALL('Table')`
- Remove filters except for specific columns: `ALLEXCEPT('Table', 'Table'[Column1], 'Table'[Column2])`
- Get the top N rows based on a measure: `TOPN(10, 'Table', [Measure])`
- Get the bottom N rows based on a measure: `BOTTOMN(10, 'Table', [Measure])`

3. Time Intelligence

- Get the year-to-date value of a measure: `TOTALYTD([Measure], 'Date'[Date])`
- Get the quarter-to-date value of a measure: `TOTALQTD([Measure], 'Date'[Date])`
- Get the month-to-date value of a measure: `TOTALMTD([Measure], 'Date'[Date])`
- Get the value of a measure in the previous year: `CALCULATE([Measure], SAMEPERIODLASTYEAR('Date'[Date]))`
- Get the value of a measure in the previous quarter: `CALCULATE([Measure], PREVIOUSQUARTER('Date'[Date]))`

By: Waleed Mousa

- Get the value of a measure in the previous month: `CALCULATE([Measure], PREVIOUSMONTH('Date'[Date]))`
- Get the year-over-year growth of a measure: `([Measure] - CALCULATE([Measure], SAMEPERIODLASTYEAR('Date'[Date]))) / CALCULATE([Measure], SAMEPERIODLASTYEAR('Date'[Date]))`

4. Conditional Calculations

- Conditional statement using IF: `IF([Condition], [Value if True], [Value if False])`
- Multiple conditions using SWITCH: `SWITCH([Expression], [Value1], [Result1], [Value2], [Result2], ...)`
- Check if a condition is true: `AND([Condition1], [Condition2], ...)`
- Check if any condition is true: `OR([Condition1], [Condition2], ...)`
- Check if a value exists in a table: `CONTAINS('Table', 'Table'[Column], "Value")`
- Check if a table is empty: `ISEMPTY('Table')`

5. Text Manipulation

- Concatenate text values: `CONCATENATE('Table'[Column1], " ", 'Table'[Column2])`
- Extract a substring from a text value: `MID('Table'[Column], 1, 5)`
- Find the position of a substring within a text value: `FIND("Substring", 'Table'[Column])`
- Replace a substring within a text value: `SUBSTITUTE('Table'[Column], "Old", "New")`
- Convert text to uppercase: `UPPER('Table'[Column])`
- Convert text to lowercase: `LOWER('Table'[Column])`
- Trim whitespace from text: `TRIM('Table'[Column])`

6. Date and Time Functions

- Get the current date: `TODAY()`
- Get the current time: `NOW()`
- Extract the year from a date: `YEAR('Table'[Date])`
- Extract the month from a date: `MONTH('Table'[Date])`
- Extract the day from a date: `DAY('Table'[Date])`
- Calculate the difference between two dates in days: `DATEDIFF('Table'[Date1], 'Table'[Date2], DAY)`

By: Waleed Mousa

- Add or subtract days from a date: `DATEADD('Table'[Date], 7, DAY)`

7. Ranking and Sorting

- Rank values within a group: `RANKX('Table', 'Table'[Measure], ,,,ASC)`
- Rank values across the entire table: `RANK.EQ('Table'[Measure],,,ASC)`
- Sort a table by a measure: `TOPN(10, 'Table', [Measure],,ASC)`
- Sort a table by multiple columns: `TOPN(10, 'Table', 'Table'[Column1], 'Table'[Column2],,ASC)`

8. Iterative Calculations

- Calculate a running total: `CALCULATE([Measure], FILTER(ALL('Table'), 'Table'[Column] <= MAX('Table'[Column])))`
- Calculate a moving average: `AVERAGEX(FILTER('Table', 'Table'[Column] <= MAX('Table'[Column]) && 'Table'[Column] > MAX('Table'[Column]) - 30), [Measure])`
- Calculate a cumulative sum: `CALCULATE([Measure], FILTER(ALL('Table'), 'Table'[Column] <= EARLIER('Table'[Column])))`
- Calculate a percentage of total: `DIVIDE([Measure], CALCULATE([Measure], ALL('Table')))`

9. Hierarchical Data

- Calculate the parent value in a hierarchy: `RELATED('ParentTable'[Column])`
- Calculate the child value in a hierarchy: `RELATEDTABLE('ChildTable')`
- Calculate the ancestor value at a specific level: `ANCESTOR('HierarchyTable'[Column], 2)`
- Calculate the descendant values at a specific level: `DESCENDANTS('HierarchyTable'[Column], 2)`

10. Advanced Calculations

- Calculate the distinct count of a column with a condition: `DISTINCTCOUNT(FILTER('Table', [Condition]))`
- Calculate the median of a column: `MEDIANX('Table', 'Table'[Column])`
- Calculate the mode of a column: `MAXX(GROUPBY('Table', 'Table'[Column], "Count", COUNTX(CURRENTGROUP())), [Count])`
- Calculate the correlation between two columns: `CORRELATIONX('Table', 'Table'[Column1], 'Table'[Column2])`

By: Waleed Mousa

- Calculate the covariance between two columns: `COVARIANCEX.P('Table', 'Table'[Column1], 'Table'[Column2])`

11. Error Handling

- Handle division by zero: `IFERROR([Measure] / [Denominator], 0)`
- Handle missing values: `IFBLANK('Table'[Column], 0)`
- Handle invalid values: `IFNA([Measure], 0)`
- Handle errors in a calculation: `ISERROR([Measure])`

12. Table Manipulation

- Create a new table from an existing table: `SELECTCOLUMNS('Table', "NewColumn1", [Expression1], "NewColumn2", [Expression2])`
- Filter a table based on multiple conditions: `FILTER('Table', [Condition1] && [Condition2])`
- Join two tables based on a common column: `NATURALINNERJOIN('Table1', 'Table2')`
- Append rows from one table to another: `UNION('Table1', 'Table2')`
- Group a table by a column and aggregate values: `SUMMARIZECOLUMNS('Table'[GroupColumn], "AggregatedValue", [Expression])`
- Pivot a table based on a column: `EVALUATE TOPN(10, ADDCOLUMNS(SUMMARIZE('Table', 'Table'[PivotColumn]), "Value", [Measure]), [Measure], , DESC)`

13. Performance Optimization

- Use variables to store intermediate results: `VAR Result = [Calculation] RETURN Result`
- Avoid using FILTER inside iterative functions: `SUMX('Table', IF('Table'[Column] = "Value", [Measure], 0))`
- Use DISTINCTCOUNT instead of COUNT(DISTINCT()): `DISTINCTCOUNT('Table'[Column])`
- Avoid using CALCULATE inside iterative functions: `SUMX(FILTER('Table', [Condition]), [Measure])`
- Use CALCULATETABLE instead of FILTER for complex filters: `CALCULATETABLE('Table', [Condition1], [Condition2])`

14. Statistical Functions

- Calculate the standard deviation of a column: `STDEV.P('Table'[Column])`

By: Waleed Mousa

- Calculate the variance of a column: `VAR.P('Table'[Column])`
- Calculate the skewness of a column: `SKEW.P('Table'[Column])`
- Calculate the kurtosis of a column: `KURTOSIS.P('Table'[Column])`
- Calculate the correlation coefficient between two columns: `CORRELATION('Table'[Column1], 'Table'[Column2])`

15. Miscellaneous Functions

- Get the current user's name: `USERNAME()`
- Get the current user's email: `USERPRINCIPALNAME()`
- Get the current workspace name: `WORKSPACE()`
- Get the current report name: `REPORTNAME()`
- Get the current page name: `PAGENAME()`
- Get the current filter context: `FILTERS('Table')`
- Get the current row context: `ROW("Column1", [Expression1], "Column2", [Expression2])`

16. Geographical Analysis

- Calculate the distance between two points: `DISTANCE('Table'[Latitude1], 'Table'[Longitude1], 'Table'[Latitude2], 'Table'[Longitude2])`
- Calculate the area of a polygon: `GEOPOLYGONAREA('Table'[Geometry])`
- Calculate the length of a line: `GEOLINELENGTH('Table'[Geometry])`
- Check if a point is within a polygon: `GEOPOINTINPOLYGON('Table'[PointGeometry], 'Table'[PolygonGeometry])`
- Find the intersection of two geometries: `GEOINTERSECTS('Table'[Geometry1], 'Table'[Geometry2])`

17. Financial Calculations

- Calculate the net present value (NPV) of a series of cash flows: `NPV([Discount Rate], 'Table'[Cash Flow])`
- Calculate the internal rate of return (IRR) of a series of cash flows: `IRR('Table'[Cash Flow])`
- Calculate the payment amount for a loan or annuity: `PMT([Interest Rate], [Number of Periods], [Loan Amount])`
- Calculate the future value of an investment: `FV([Interest Rate], [Number of Periods], [Payment], [Present Value])`
- Calculate the present value of an investment: `PV([Interest Rate], [Number of Periods], [Payment], [Future Value])`

By: Waleed Mousa

18. Image and Hyperlink Functions

- Display an image in a table: `IMAGE([Image URL])`
- Create a hyperlink to a web page: `HYPERLINK([URL], [Display Text])`
- Create a hyperlink to another Power BI report: `DRILLTHROUGH([Report Name], [Table])`
- Create a hyperlink to a bookmark in the current report: `BOOKMARKLINK([Bookmark Name], [Display Text])`

19. Security and Access Control

- Check if the current user has permission to a table: `USERPERMISSION('Table')`
- Check if the current user belongs to a specific role: `USERINROLE([Role Name])`
- Check if the current user has permission to a specific column: `COLUMNPERMISSION('Table'[Column])`
- Filter a table based on the user's permissions: `CALCULATETABLE('Table', USERPERMISSION('Table'))`

20. Dynamic and Parameterized Calculations

- Create a dynamic measure based on a selected value: `SWITCH(SELECTEDVALUE('Table'[Column]), [Value1], [Measure1], [Value2], [Measure2])`
- Create a parameterized calculation based on a slicer selection: `CALCULATE([Measure], TREATAS(VALUES('Slicer'[Column]), 'Table'[Column]))`
- Create a dynamic filter based on a parameter: `CALCULATE([Measure], FILTER('Table', 'Table'[Column] = [Parameter]))`
- Create a dynamic top N filter based on a parameter: `TOPN([Parameter], 'Table', [Measure])`

21. Data Quality and Validation

- Check if a value is valid: `ISVALUE('Table'[Column])`
- Check if a value is a number: `ISNUMBER('Table'[Column])`
- Check if a value is a date: `ISDATE('Table'[Column])`
- Check if a value is text: `ISTEXT('Table'[Column])`
- Check if a value is blank: `ISBLANK('Table'[Column])`

By: Waleed Mousa

- Count the number of invalid values in a column: `COUNTX(FILTER('Table', NOT(ISVALUE('Table'[Column]))), 1)`

22. Advanced Table Manipulation

- Unpivot a table: `SELECTCOLUMNS(UNPIVOT('Table', "Attribute", "Value"), "Attribute", [Attribute], "Value", [Value])`
- Create a running total with partitioning: `CALCULATE([Measure], FILTER(ALL('Table'), 'Table'[Partition] = EARLIER('Table'[Partition]) && 'Table'[Date] <= EARLIER('Table'[Date])))`
- Create a running average with partitioning: `AVERAGEX(FILTER(ALL('Table'), 'Table'[Partition] = EARLIER('Table'[Partition]) && 'Table'[Date] <= EARLIER('Table'[Date])), [Measure])`
- Create a cumulative distribution: `COUNTROWS(FILTER('Table', 'Table'[Column] <= EARLIER('Table'[Column]))) / COUNTROWS('Table')`

23. Custom Formatting

- Format a measure as currency: `FORMAT([Measure], "$#,##0.00")`
- Format a measure as percentage: `FORMAT([Measure], "0.00%")`
- Format a measure as date: `FORMAT([Measure], "mm/dd/yyyy")`
- Format a measure as time: `FORMAT([Measure], "hh:mm:ss")`
- Format a measure with a custom format string: `FORMAT([Measure], "Custom Format String")`

24. Data Modeling and Relationships

- Create a many-to-one relationship: `'Table1'[ForeignKey] = 'Table2'[PrimaryKey]`
- Create a many-to-many relationship: `'FactTable'[ForeignKey1] = 'DimensionTable1'[PrimaryKey] && 'FactTable'[ForeignKey2] = 'DimensionTable2'[PrimaryKey]`
- Create an active relationship: `USERELATIONSHIP('Table1'[ForeignKey], 'Table2'[PrimaryKey])`
- Create an inactive relationship: `CROSSFILTER('Table1'[ForeignKey], 'Table2'[PrimaryKey], NONE)`

25. Data Importing and Transformation

- Import data from a CSV file: `Csv.Document([File Path])`

By: Waleed Mousa

- **Import data from an Excel file:** `Excel.Workbook([File Path])`
- **Import data from a SQL database:** `Sql.Database([Server], [Database], [Query])`
- **Import data from a web service:** `Web.Contents([URL])`
- **Transform data using Power Query:** `Table.TransformColumns('Table', {"Column1", each _ * 2})`