VIRTAVIS LEARNING

FREE CHEAT SHEET

DAX for Power BI

The formulas, patterns and ideas you need to write measures that give the right answer.

HOW A MEASURE WORKS

Filters `slicers, visuals` › **Measure** `[Total Sales]` › **Result** `£1,245,000`

The same formula gives a new answer for every filter.

GOLDEN RULE

Write small base measures first, then build on them. Never repeat SUM(...) inside other measures.

THE MODEL IN THESE EXAMPLES

One fact table, filtered by its dimensions.

Sales *fact*
SalesID
DateKey
ProductKey
CustomerKey
ChannelKey
Quantity
Revenue
Cost

'Date'
Date · Year · Month

Product
Product Name · Category · Brand

Customer
CustomerKey · Country

Channel
Channel Type

Filters flow from each dimension (one side) to Sales (many side). Refer to columns as Table[Column].

MEASURE OR CALCULATED COLUMN?

Choose before you write the formula.

Measure

Calculated when a visual asks, for the current filters. Not stored. Use for anything you add up or divide.

```
Total Sales =
SUM(Sales[Revenue])
```

Calculated column

Calculated once per row at refresh and stored. Use to slice or group, such as a price band.

```
Band = IF(Sales[Revenue] >
100, "Big", "Small")
```

TWO KINDS OF CONTEXT

Why the same measure shows different numbers.

Filter context

Which rows a measure can see. Set by slicers, visual rows and columns, page filters and CALCULATE.

```
UK + 2025 + Bikes =
£1,245,000
```

Row context

"The current row". Exists in calculated columns and X functions such as SUMX. It doesn't filter anything by itself.

```
Sales[Revenue] -
Sales[Cost]
```

Context transition: CALCULATE, or a measure used inside an X function, turns the current row into a filter. That's why AVERAGEX(VALUES(Customer[CustomerKey]), [Total Sales]) works per customer.

AGGREGATE A COLUMN

The building blocks of every measure.

Total	`SUM(Sales[Revenue])`
Average	`AVERAGE(Sales[Revenue])`
Count rows	`COUNTROWS(Sales)` One row per sale
Count unique	`DISTINCTCOUNT(Sales[CustomerKey])`
First / last	`MIN('Date'[Date])` `MAX('Date'[Date])` In the current filter context

CALCULATE

The most important function: change the filters, then calculate.

```
UK Sales =
CALCULATE(
  [Total Sales],
  Customer[Country] = "UK"
)
```

- Evaluates **[Total Sales]** with Country set to UK.
- A plain filter **replaces** any Country filter from a slicer. Wrap it in KEEPFILTERS to respect the slicer.
- Filter arguments combine with AND.

Two filters (AND)

```
CALCULATE([Total Sales],
  Customer[Country] = "UK",
  Channel[Channel Type] = "Online")
```

Either value (OR)

```
CALCULATE([Total Sales],
  Customer[Country] IN {"UK", "IE"})
```

Remove a filter

```
CALCULATE([Total Sales],
  REMOVEFILTERS(Product))
```

Respect the slicer too

```
CALCULATE([Total Sales],
  KEEPFILTERS(Customer[Country] = "UK"))
```

FILTER FUNCTIONS

Use inside CALCULATE to change filters.

Remove filters	`REMOVEFILTERS(Product)` Only works as a CALCULATE filter
Ignore filters	`ALL(Product[Category])` Like REMOVEFILTERS, but also returns a table
Keep both	`KEEPFILTERS(Product[Category] = "Bikes")` Adds to the slicer instead of replacing it
Filter by a measure	`FILTER(VALUES(Product[Product Name]), [Total Sales] > 10000)` Use FILTER for measure tests. Plain column tests don't need it
Visible values	`VALUES(Product[Category])` Distinct values in the current context
Selected value	`SELECTEDVALUE(Product[Category], "All categories")` Returns the fallback if 0 or 2+ are selected

IF / THEN LOGIC

Return different results by condition.

IF	`IF([Total Sales] > 100000, "High", "Low")`
Several bands	`SWITCH(TRUE(), [Total Sales] > 100000, "High", [Total Sales] > 50000, "Medium", "Low")` First true test wins, so biggest first
AND / OR	`IF([Total Sales] > 0 && [Profit] > 0, "Valid", "Check")` && and ||. AND() and OR() take only two tests
Blank check	`IF(ISBLANK([Total Sales]), "No sales")`
Blank to zero	`COALESCE([Total Sales], 0)` Careful: zeros make every empty row appear

ITERATORS (X FUNCTIONS)

Work row by row, then aggregate.

Sum of a calculation	`SUMX(Sales, Sales[Revenue] - Sales[Cost])`
Average per customer	`AVERAGEX(VALUES(Customer[CustomerKey]), [Total Sales])` Average of each customer's total
Count matching rows	`COUNTROWS( FILTER(Sales, Sales[Revenue] > 100))`
Rank	`RANKX(ALL(Product[Product Name]), [Total Sales], , DESC)` ALL so every product is ranked, not just one

VARIABLES

Name each step. Easier to read and debug.

```
Sales Growth % =
VAR CurrentSales = [Total Sales]
VAR LastYear =
    CALCULATE([Total Sales],
        SAMEPERIODLASTYEAR('Date'[Date]))
RETURN
    DIVIDE(CurrentSales - LastYear, LastYear)
```

- Each VAR is calculated **once**, then reused.
- Debug by returning one step: RETURN LastYear.
- DIVIDE without a third argument gives blank, the honest answer when last year had no sales.

TIME INTELLIGENCE

Needs a date table marked as a date table.

Date table	`Date = CALENDAR(DATE(2024, 1, 1), DATE(2026, 12, 31))` Cover every year in the data, no gaps
Year to date	`TOTALYTD([Total Sales], 'Date'[Date])`
Year to 31 March	`TOTALYTD([Total Sales], 'Date'[Date], "31/3")` Year-end as text in your file's locale (US: "3/31")
Month to date	`TOTALMTD([Total Sales], 'Date'[Date])`
Same period last year	`CALCULATE([Total Sales], SAMEPERIODLASTYEAR('Date'[Date]))`
Previous month	`CALCULATE([Total Sales], DATEADD('Date'[Date], -1, MONTH))`
Last 3 months	`CALCULATE([Total Sales], DATESINPERIOD('Date'[Date], MAX('Date'[Date]), -3, MONTH))` Rolling window ending on the last visible date

STARTER MEASURES

Build these first, then reuse them everywhere.

Total Sales =	`SUM(Sales[Revenue])`
Total Cost =	`SUM(Sales[Cost])`
Profit =	`[Total Sales] - [Total Cost]`
Margin % =	`DIVIDE([Profit], [Total Sales])`
Sales Count =	`COUNTROWS(Sales)`
Customers =	`DISTINCTCOUNT(Sales[CustomerKey])`
Avg Sale =	`DIVIDE([Total Sales], [Sales Count])`
% of All Products =	`DIVIDE([Total Sales], CALCULATE([Total Sales], ALL(Product)))`

"% of All Products" ignores product filters and slicers. Use ALLSELECTED(Product) for "% of what's shown".

FIVE CLASSIC MISTAKES

Check these before you publish.

1. **Dividing with /.** Use DIVIDE(a, b). It returns blank, not an error, when b is 0.
2. **Calculated columns for totals.** Use measures for anything you add up. Columns are fixed at refresh.
3. **No date table.** Time intelligence needs a marked date table with every day in it.
4. **Not checking totals.** Compare a measure with the source for one month before you publish.
5. **Repeating logic.** Build base measures once ([Total Sales]) and reuse them.

Check the numbers. Then say what they mean.

By **Sophie Lumpa**, Data and Business Intelligence Specialist. Checked against Microsoft's DAX reference, October 2026. © 2026 Virtavis Learning · Free to share with credit

Start the course › Power BI for Data Analytics

VIRTAVIS
virtavis.com/courses