1. % Growth in Sales Compared to Last Year
DAX

```
Sales Growth % YoY =
VAR CurrentSales = SUM(Sales[SalesAmount])
VAR LastYearSales = CALCULATE(SUM(Sales[SalesAmount]), SAMEPERIODLASTYEAR(Date[Date]))
RETURN
DIVIDE(CurrentSales - LastYearSales, LastYearSales)
```

✅ 2. Difference Between Current Month and Previous Month Sales
DAX

```
Sales MoM Difference =
VAR CurrentMonthSales = SUM(Sales[SalesAmount])
VAR PrevMonthSales = CALCULATE(SUM(Sales[SalesAmount]), PREVIOUSMONTH(Date[Date]))
RETURN
CurrentMonthSales - PrevMonthSales
```

✅ 3. Total Boxes Shipped and Average Monthly Boxes in One Measure (Using VAR)
DAX

```
Total and Avg Boxes =
VAR TotalBoxes = SUM(Shipments[Boxes])
VAR MonthsWithData = CALCULATE(DISTINCTCOUNT(Date[Month]), REMOVEFILTERS(Date[Date]))
VAR AvgMonthly = DIVIDE(TotalBoxes, MonthsWithData)
RETURN
"Total: " & FORMAT(TotalBoxes, "#,##0") & " | Avg: " & FORMAT(AvgMonthly, "#,##0.0")
```

✅ 4. Only Return Average Monthly Boxes (Still Using VAR)
DAX

```
Average Monthly Boxes =
VAR TotalBoxes = SUM(Shipments[Boxes])
VAR MonthsWithData = CALCULATE(DISTINCTCOUNT(Date[Month]), REMOVEFILTERS(Date[Date]))
RETURN DIVIDE(TotalBoxes, MonthsWithData)
```

✅ 5. % Growth from Last Month
DAX

```
Sales Growth % MoM =
VAR CurrentMonthSales = SUM(Sales[SalesAmount])
VAR PrevMonthSales = CALCULATE(SUM(Sales[SalesAmount]), PREVIOUSMONTH(Date[Date]))
RETURN DIVIDE(CurrentMonthSales - PrevMonthSales, PrevMonthSales)
```

✅ 6. 3-Month Moving Average of Sales
DAX

```
Sales 3-Month MA =
CALCULATE(
   AVERAGEX(
      DATESINPERIOD(Date[Date], MAX(Date[Date]), -3, MONTH),
      CALCULATE(SUM(Sales[SalesAmount]))
   )
)
```

✅ 7. Dynamic Card Message Based on Sales Rank and YoY Performance
DAX

```
Sales Performance Message =
VAR Product = SELECTEDVALUE(Products[ProductName])
VAR SalesThisYear = CALCULATE(SUM(Sales[SalesAmount]), YEAR(Date[Date]) = YEAR(TODAY()))
VAR SalesLastYear = CALCULATE(SUM(Sales[SalesAmount]), SAMEPERIODLASTYEAR(Date[Date]))
VAR YoYGrowth = DIVIDE(SalesThisYear - SalesLastYear, SalesLastYear)
VAR ProductRank =
  RANKX(
    ALLSELECTED(Products[ProductName]),
    CALCULATE(SUM(Sales[SalesAmount])),
    ,
    DESC
  )
RETURN
SWITCH(TRUE(),
  ProductRank <= 3 && YoYGrowth > 0.1, "Top Performer - Sales up by " & FORMAT(YoYGrowth,
"0.0%"),
  YoYGrowth >= -0.05 && YoYGrowth <= 0.05, "Consistent Performer",
  "Needs Improvement"
)
```

✅ 8. Top 5 Tips to Optimize DAX Queries

Avoid Repeated Calculations
Use VAR to store expensive expressions like totals or ranks to prevent recalculating.

Filter Minimally
Use REMOVEFILTERS or KEEPFILTERS wisely to reduce filter context overhead.

Prefer CALCULATE with Filter Arguments Over FILTER() When Possible
FILTER creates row context; simple expressions are better for performance.

Minimize Use of Iterators
Prefer aggregated functions (e.g., SUMX) only when necessary. Use base aggregations like SUM when possible.

Optimize Row Context Transition
Avoid nesting CALCULATE inside X functions like SUMX unless necessary, to reduce context switching.

✅ 9. Benefits of Tools like DAX Studio, Performance Analyzer, Tabular Editor

DAX Studio
Analyze query plans, durations, and server timings. Helps you find bottlenecks in formulas.

Performance Analyzer
Shows visual rendering vs. DAX query time, helping to separate UI and model optimization.

Tabular Editor
Great for writing and managing DAX measures in bulk, validating syntax, and using Best Practice Analyzer for model validation.

These tools provide visibility into the behind-the-scenes behavior of your queries and data models—something manual optimization alone can’t give.

✅ 10. Create a Top 5 Flag Using RANKX Without Recalculating Rank
DAX

```
Top 5 Product Flag =
VAR ProductRank =
   RANKX(
      ALL(Products[ProductName]),
      CALCULATE(SUM(Sales[SalesAmount])),
      ,
      DESC
   )
RETURN IF(ProductRank <= 5, "Yes", "No")
```