```
================================================================================
CATEGORY 3: TEXT FUNCTIONS (8 Functions)
================================================================================
```

2️⃣3️⃣ CONCATENATE() or & - JOIN TEXT

```
SYNTAX:
  CONCATENATE(<text1>, <text2>, ...)
  <text1> & <text2> & ...

PARAMETERS:
  • <text1>, <text2>: Text values to join

EXAMPLE FROM YOUR DATASET:

Full Address (Calculated Column):
  Full Address = SalesData[City] & ", " & SalesData[State]

  Result:
    • "Mumbai, Maharashtra"
    • "Delhi, Delhi"
    • "Bangalore, Karnataka"

Customer City-State (Calculated Column):
  Location = CustomerMaster[CustomerName] & " - " & CustomerMaster[City]

  Result:
    • "Rajesh Kumar - Mumbai"
    • "Priya Singh - Delhi"

Order Summary (Calculated Column):
  Order Summary = "Order " & SalesData[OrderID] & " for ₹" &
SalesData[FinalAmount]

  Result: "Order 10001 for ₹45,230"

REAL USAGE:
  • Create full addresses
  • Combine names
  • Generate descriptions
  • Create labels
```

2️⃣4️⃣ LEN() - TEXT LENGTH

```
SYNTAX:
  LEN(<text>)

PARAMETERS:
  • <text>: Text to measure

EXAMPLE FROM YOUR DATASET:

Customer Name Length (Calculated Column):
  Name Length = LEN(CustomerMaster[CustomerName])

  Result:
    • "Rajesh Kumar" = 12 characters
    • "Priya Singh" = 11 characters

Email Length (Calculated Column):
```

```
  Email Length = LEN(CustomerMaster[Email])

Quality Check (Calculated Column):
  Valid Email = IF(LEN(CustomerMaster[Email]) > 5, "Valid", "Too Short")

REAL USAGE:
  • Data quality validation
  • Text length constraints
  • Format verification
```

2️⃣5️⃣ UPPER() - CONVERT TO UPPERCASE

```
SYNTAX:
  UPPER(<text>)

PARAMETERS:
  • <text>: Text to convert

EXAMPLE FROM YOUR DATASET:

City in Uppercase (Calculated Column):
  City Upper = UPPER(SalesData[City])

  Result:
    • "Mumbai" → "MUMBAI"
    • "Delhi" → "DELHI"
    • "Bangalore" → "BANGALORE"

Category in Caps (Calculated Column):
  Category Upper = UPPER(SalesData[Category])

  Result:
    • "Electronics" → "ELECTRONICS"
    • "Clothing" → "CLOTHING"

REAL USAGE:
  • Standardize text display
  • Data consistency
  • Report formatting
```

2️⃣6️⃣ LOWER() - CONVERT TO LOWERCASE

```
SYNTAX:
  LOWER(<text>)

PARAMETERS:
  • <text>: Text to convert

EXAMPLE FROM YOUR DATASET:

Email Format (Calculated Column):
  Email Lower = LOWER(CustomerMaster[Email])

  Result: Standardizes email to lowercase for consistency

Product Name Lower (Calculated Column):
  Product Lower = LOWER(SalesData[Product])
```

```
  Result:
    • "Laptop" → "laptop"
    • "T-Shirt" → "t-shirt"

REAL USAGE:
  • Standardize text
  • Email formatting
  • Data consistency
```

2️⃣7️⃣ TRIM() - REMOVE EXTRA SPACES

```
SYNTAX:
  TRIM(<text>)

PARAMETERS:
  • <text>: Text to clean

EXAMPLE FROM YOUR DATASET:

Clean City Name (Calculated Column):
  City Clean = TRIM(SalesData[City])

  Result: Removes leading/trailing spaces
    • "  Mumbai  " → "Mumbai"
    • " Delhi " → "Delhi"

Clean Customer Name (Calculated Column):
  Name Clean = TRIM(CustomerMaster[CustomerName])

REAL USAGE:
  • Data cleaning
  • Remove extra spaces
  • Data quality
```

2️⃣8️⃣ SUBSTITUTE() - REPLACE TEXT

```
SYNTAX:
  SUBSTITUTE(<text>, <old_text>, <new_text>, [instance])

PARAMETERS:
  • <text>: Original text
  • <old_text>: Text to find
  • <new_text>: Text to replace with
  • <instance>: Which occurrence (optional)

EXAMPLE FROM YOUR DATASET:

Replace State Abbreviation (Calculated Column):
  State Full = SUBSTITUTE(SalesData[State], "MH", "Maharashtra")

  Result: "MH" → "Maharashtra"

Category Correction (Calculated Column):
  Category Corrected = SUBSTITUTE(SalesData[Category], "Electronic",
"Electronics")

Region Format (Calculated Column):
  Region Clean = SUBSTITUTE(SalesData[Region], "North", "North India")
```

REAL USAGE:
  • Data standardization
  • Correct misspellings
  • Format conversion

2️⃣9️⃣ SEARCH() / FIND() - FIND TEXT POSITION

SYNTAX:
  SEARCH(<find_text>, <within_text>, [start_num])
  FIND(<find_text>, <within_text>, [start_num])

PARAMETERS:
  • <find_text>: Text to find
  • <within_text>: Text to search in
  • <start_num>: Starting position (optional)

EXAMPLE FROM YOUR DATASET:

Find "@" Position in Email (Calculated Column):
  At Position = SEARCH("@", CustomerMaster[Email])

  Result: Position of @ symbol
    • "rajesh.kumar@gmail.com" → 13

Check if City Contains "Delhi" (Calculated Column):
  Has Delhi = IF(ISERROR(SEARCH("Delhi", SalesData[City])), "No", "Yes")

REAL USAGE:
  • Text pattern finding
  • Email validation
  • Data parsing

3️⃣0️⃣ LEFT() / RIGHT() / MID() - EXTRACT TEXT

SYNTAX:
  LEFT(<text>, <num_chars>)   – Get first N characters
  RIGHT(<text>, <num_chars>)   – Get last N characters
  MID(<text>, <start>, <count>) – Get middle characters

PARAMETERS:
  • <text>: Original text
  • <num_chars>: Number of characters
  • <start>: Starting position
  • <count>: Number of characters to extract

EXAMPLE FROM YOUR DATASET:

Extract Year from OrderDate (Calculated Column):
  Order Year = LEFT(SalesData[OrderDate], 4)

  Result: "2023-01-15" → "2023"

Extract Month from OrderDate (Calculated Column):
  Order Month = MID(SalesData[OrderDate], 6, 2)

  Result: "2023-01-15" → "01"

Get Domain from Email (Calculated Column):
  Email Domain = RIGHT(CustomerMaster[Email], 10)

  Result: "rajesh.kumar@gmail.com" → "gmail.com"

Extract Last 5 Characters of OrderID (Calculated Column):
  Order Suffix = RIGHT(SalesData[OrderID], 5)

REAL USAGE:
  • Parse dates
  • Extract domains
  • Split text
  • Code extraction