***Sample Power BI Report***

***Design Document Fragment***

***Model***

|  |  |
| --- | --- |
| *Date:* | *2025-05-19* |
| *Version:* | *0.7* |
| *Author:* | *Greg Philps* |

***Table of Contents***

[1. Introduction 3](#_Toc196648538)

[2. Scope of Work 3](#_Toc196648539)

[3. Workflow 3](#_Toc196648540)

[4. Issues 3](#_Toc196648541)

[5. Business Rules 3](#_Toc196648542)

[6. Data 3](#_Toc196648543)

[7. Reports 3](#_Toc196648544)

[8. Validation 3](#_Toc196648545)

[9. Deployment 3](#_Toc196648546)

[10. Model 4](#_Toc196648547)

[10.1. General 4](#_Toc196648548)

[10.1.1. Staging 4](#_Toc196648549)

[10.1.2. INFO VIEW Functions 4](#_Toc196648550)

[10.2. Tables and Columns 5](#_Toc196648551)

[10.2.1. Fact 6](#_Toc196648552)

[10.2.2. Dimension 6](#_Toc196648553)

[10.2.3. Supporting 8](#_Toc196648554)

[10.3. Relationships 9](#_Toc196648555)

[10.4. Measures 10](#_Toc196648556)

# Introduction

*… see sample fragment 01 …*

# Scope of Work

*… see sample fragment 01 …*

# Workflow

*… see sample fragment 02 …*

# Issues

*… see sample fragment 02 …*

# Business Rules

*… see sample fragment 02 …*

# Data

*… see sample fragment 03 …*

# Reports

*… see sample fragment 04 …*

# Validation

*… see sample fragment 05 …*

# Deployment

*… see sample fragment 06 …*

# Model

## General

### Staging

*Data is extracted as-is (i.e., without transformations) into staging tables. Staging tables are clearly named as such by adding a “RAW” prefix. Staging tables are then merged as necessary to produce a model table (e.g., RAW Customers + RAW Addresses 🡪 Customers, etc.).*

*Staging Table: RAW Customers*

| ***ID*** | ***Name*** | ***Type*** | ***Sample*** | ***Notes*** |
| --- | --- | --- | --- | --- |
| *1* | *Customer ID* | *Integer* | *1* | *Primary Key* |
| *2* | *Customer* | *Text* | *Dave* |  |
| *3* | *Address ID* | *Integer* | *1* | *Foreign Key* |

*Staging Table: RAW Addresses*

| ***ID*** | ***Name*** | ***Type*** | ***Sample*** | ***Notes*** |
| --- | --- | --- | --- | --- |
| *1* | *Address ID* | *Integer* | *1* | *Primary Key* |
| *3* | *City* | *Text* | *Ottawa* |  |
| *4* | *Country* | *Text* | *Canada* |  |

*Transformation Operation: Merge RAW Customers and RAW Addresses tables using Address ID (left outer join)*

*Model Table: Customers*

| ***ID*** | ***Name*** | ***Type*** | ***Sample*** | ***Notes*** |
| --- | --- | --- | --- | --- |
| *1* | *Customer ID* | *Integer* | *1* | *Primary Key* |
| *2* | *Customer* | *Text* | *Dave* |  |
| *3* | *City* | *Text* | *Ottawa* |  |
| *4* | *Country* | *Text* | *Canada* |  |

### INFO VIEW Functions

*The DAX INFO VIEW functions are leveraged to extract and consolidate the implemented design.*

*The tables and columns, relationships, and measures as implemented are collated in documentation tables using the following procedure in Power BI Desktop:*

1. *Open DAX Query View and enter the desired query*
2. *Click the “Run” button*
3. *Click the “Copy” button and select “Entire table”*
4. *Select more than the needed number of rows in the desired design document table and use CTRL-V to paste the exported data*
5. *Remove the header row and any blank or duplicate rows*
6. *Edit the “ID” and “Notes” columns as necessary*
7. *Add the data for the “Sample” column as desired*

## Tables and Columns

*List all tables in the model; the following was used in DAX Query View from within Power BI Desktop to retrieve the table data:*

*EVALUATE*

*VAR \_Result =*

*SELECTCOLUMNS(*

*INFO.VIEW.TABLES(),*

*"ID", [ID],*

*"Name", [Name],*

*"Category", [DataCategory],*

*"Type", IF( ISBLANK([Expression]), "Normal", "Calculated" ),*

*"Expression", [Expression],*

*"Is Hidden", [IsHidden]*

*)*

*RETURN*

*\_Result*

| ***ID*** | ***Name*** | ***Category*** | ***Type*** | ***Expression*** | ***Is Hidden*** |
| --- | --- | --- | --- | --- | --- |
| *1* | *Key Measures* | *Regular* | *Normal* |  | *false* |
| *2* | *Dates* | *Time* | *Normal* |  | *false* |
| *3* | *Invoices* | *Regular* | *Normal* |  | *false* |
| *4* | *Customers* | *Regular* | *Normal* |  | *false* |
| *5* | *Last Refresh* | *Regular* | *Normal* |  | *true* |
| *6* | *Orders* | *Regular* | *Normal* |  | *false* |
| *7* | *Products* | *Regular* | *Normal* |  | *false* |
| *8* | *Regions* | *Regular* | *Normal* |  | *false* |
| *9* | *Aging* | *Regular* | *Calculated* | *SELECTCOLUMNS ( GENERATESERIES ( 1, 6, 1 ), "Aging ID", [Value] )* | *true* |

*List all columns in each table; the following can be used in DAX Query View from within Power BI Desktop to retrieve the column data (adjust the [\_Table] variable to suit):*

*EVALUATE*

*VAR \_Table = "Dates"*

*VAR \_Result =*

*SELECTCOLUMNS(*

*FILTER( INFO.VIEW.COLUMNS(), [Table] = \_Table && [Type] IN {"Data", "Calculated"} ),*

*"ID", [ID],*

*"Name", [Name],*

*"Type", IF( [DataType] = "True/False" , "Boolean", [DataType] ),*

*// "Nulls", IF( [IsNullable] = TRUE(), "Yes", "No" ),*

*"Sample", BLANK(),*

*"Notes", COMBINEVALUES( " | ", IF( [IsKey] = TRUE(), "Primary Key", BLANK() ), IF( OR( [FormatString] = "0", [DataType] = "True/False" ), BLANK(), [FormatString] ), [Expression] )*

*)*

*RETURN*

*\_Result*

### Fact

#### Invoices

| **ID** | **Name** | **Type** | **Sample** | **Notes** |
| --- | --- | --- | --- | --- |
| 1 | Invoice ID | Integer | 2401 | Primary Key |
| 2 | Invoice Date | Date | January 31, 2024 | Long Date |
| 3 | Amount | Number | 371900 |  |
| 4 | Customer ID | Integer | 1 | Foreign Key |
| 5 | Region ID | Integer | 1 | Foreign Key |

#### Orders

| **ID** | **Name** | **Type** | **Sample** | **Notes** |
| --- | --- | --- | --- | --- |
| 1 | Order ID | Integer | 2407 | Primary Key |
| 2 | Order Date | Date | July 17, 2024 | Long Date |
| 3 | Product ID | Integer | 5 | Foreign Key |
| 4 | Quantity | Integer | 16 |  |
| 5 | Customer ID | Integer | 4 | Foreign Key |
| 6 | Region ID | Integer | 6 | Foreign Key |

### Dimension

#### Customers

| **ID** | **Name** | **Type** | **Sample** | **Notes** |
| --- | --- | --- | --- | --- |
| 1 | Customer ID | Integer | 1 | Primary Key |
| 2 | Customer | Text | Dave |  |
| 3 | City | Text | Ottawa |  |
| 4 | Country | Text | Canada |  |

#### Products

| **ID** | **Name** | **Type** | **Sample** | **Notes** |
| --- | --- | --- | --- | --- |
| 1 | Product ID | Integer | 4 | Primary Key |
| 2 | Product | Text | Shoes |  |
| 3 | Unit Price | Integer | 120 |  |
| 4 | Unit Cost | Integer | 110 |  |

#### Regions

| **ID** | **Name** | **Type** | **Sample** | **Notes** |
| --- | --- | --- | --- | --- |
| 1 | Region ID | Integer | 1 | Primary Key |
| 2 | Region | Text | Canada |  |

#### Dates

*A date table will be created using M-code directly in Power BI. The date table should cover the entire time span of interest and with full calendar years and will be initially created for a three-year period (i.e., 2023-01-01 to 2025-12-31).*

*The [Dates] table was generated using Power Query/M code from the Extended Date Table by Enterprise DNA Expert Melissa de Korte, one of the (if not the) best date tables available for Power BI.*

<https://forum.enterprisedna.co/t/extended-date-table-power-query-m-function/6390>

| ***ID*** | ***Name*** | ***Type*** | ***Sample*** | ***Notes*** |
| --- | --- | --- | --- | --- |
| *1* | *Date* | *Date* | *05-Feb-2025* | *Primary Key*  *dd-mmm-yyyy* |
| *2* | *Year* | *Integer* | *2025* |  |
| *3* | *Quarter Number* | *Integer* | *1* |  |
| *4* | *Quarter* | *Text* | *Q1* |  |
| *5* | *Month* | *Integer* | *2* |  |
| *6* | *Month & Year* | *Text* | *Feb 2025* |  |
| *7* | *MonthnYear* | *Integer* | *202502* |  |
| *8* | *Month Name* | *Text* | *February* |  |
| *9* | *Month Short* | *Text* | *Feb* |  |
| *10* | *Month Initial* | *Text* | *F* |  |
| *11* | *Day of Month* | *Integer* | *5* |  |
| *12* | *Day of Week Number* | *Integer* | *2* |  |
| *13* | *Day of Week Name* | *Text* | *Wednesday* |  |
| *14* | *Day of Week Initial* | *Text* | *W* |  |
| *15* | *DateInt* | *Integer* | *20250205* |  |
| *16* | *IsAfterToday* | *Boolean* | *True* |  |
| *17* | *IsWeekDay* | *Boolean* | *True* |  |
| *18* | *Day Type* | *Text* | *Weekday* |  |
| *19* | *Fiscal Year* | *Text* | *FY2025* |  |
| *20* | *Fiscal Quarter* | *Text* | *FQ4 2025* |  |
| *21* | *FQuarternYear* | *Integer* | *20254* |  |
| *22* | *Fiscal Period* | *Text* | *FP11 2025* |  |
| *23* | *FPeriodnYear* | *Integer* | *202511* |  |
| *24* | *IsCurrentFY* | *Boolean* | *True* |  |
| *25* | *IsCurrentFQ* | *Boolean* | *True* |  |
| *26* | *IsCurrentFP* | *Boolean* | *False* |  |
| *27* | *IsPYTD* | *Boolean* | *False* |  |
| *28* | *IsPFYTD* | *Boolean* | *False* |  |

### Supporting

#### Aging

*This is a calculated table and reflects the invoice aging groups used by the organization.*

*Aging = SELECTCOLUMNS ( GENERATESERIES ( 1, 6, 1 ), "Aging ID", [Value] )*

| ***ID*** | ***Name*** | ***Type*** | ***DAX Expression*** | ***Notes*** |
| --- | --- | --- | --- | --- |
| *1* | *Aging ID* | *Integer* | *(see table creation expression above)* | *Primary Key* |
| *2* | *Aging* | *Text* | *SWITCH(TRUE(),*  *[Aging ID] = 1, "0-30 days",*  *[Aging ID] = 2, "31-60 days",*  *[Aging ID] = 3, "61-90 days",*  *[Aging ID] = 4, "91-120 days",*  *[Aging ID] = 5, "121-180 days",*  *[Aging ID] = 6, "over 180 days",*  *"ERROR")* |  |
| *3* | *Min* | *Integer* | *SWITCH(TRUE(),*  *[Aging ID] = 1, 0,*  *[Aging ID] = 2, 31,*  *[Aging ID] = 3, 61,*  *[Aging ID] = 4, 91,*  *[Aging ID] = 5, 121,*  *[Aging ID] = 6, 181,*  *-1)* |  |
| *4* | *Max* | *Integer* | *SWITCH(TRUE(),*  *[Aging ID] = 1, 30,*  *[Aging ID] = 2, 60,*  *[Aging ID] = 3, 90,*  *[Aging ID] = 4, 120,*  *[Aging ID] = 5, 180,*  *[Aging ID] = 6, 9999,*  *-1)* |  |

#### Tiers

| ***ID*** | ***Name*** | ***Type*** | ***Sample*** | ***Notes*** |
| --- | --- | --- | --- | --- |
| *1* | *Tier ID* | *Integer* | *1* | *Primary Key* |
| *2* | *Tier* | *Text* | *Gold* |  |
| *3* | *Lower Bound* | *Integer* | *10,000,000* |  |
| *4* | *Upper Bound* | *Integer* | *9,999,999,999* |  |

#### Last Refresh

*There is no data source for this table, rather, it was generated using Power Query/M code by Enterprise DNA Expert Melissa de Korte:*

[*https://forum.enterprisedna.co/t/adding-a-last-refresh-date-to-your-report/6485*](https://forum.enterprisedna.co/t/adding-a-last-refresh-date-to-your-report/6485)

This table is a supporting table used by the [Last Refresh] measure.

let

Source = DateTime.FixedLocalNow(),

#"Converted to Table" = #table(1, {{Source}}),

#"Renamed Columns" = Table.RenameColumns(#"Converted to Table", {{"Column1", "Last Refresh"}}),

#"Changed Type" = Table.TransformColumnTypes(#"Renamed Columns", {{"Last Refresh", type datetime}}),

#"Inserted Date" = Table.AddColumn(#"Changed Type", "Date", each DateTime.Date([Last Refresh]), type date),

#"Insert Time" = Table.AddColumn(#"Inserted Date", "Time", each DateTime.Time([Last Refresh]), type time)

in

#"Insert Time"

| ***ID*** | ***Name*** | ***Type*** | ***Sample*** | ***Notes*** |
| --- | --- | --- | --- | --- |
| *1* | *Last Refresh* | *Date* | *2024-12-05 1:19.22* | *General Date* |
| *2* | *Date* | *Date* | *December 5, 2024* | *Long Date* |
| *3* | *Time* | *Date* | *1:19.22 PM* | *Long Time* |

## Relationships

*List all relationships in the model; the following can be used in DAX Query View from within Power BI to retrieve the relationship details:*

*EVALUATE*

*VAR \_Result =*

*SELECTCOLUMNS(*

*INFO.VIEW.RELATIONSHIPS(),*

*"ID", [ID],*

*// "Relationship", [Relationship],*

*"From", [FromTable] & "[" & [FromColumn] & "]",*

*"To", [ToTable] & "[" & [ToColumn] & "]",*

*"Cardinality", [FromCardinality] & "-to-" & [ToCardinality],*

*"Cross-filter Direction", SWITCH( TRUE(),*

*[CrossFilteringBehavior] = "OneDirection", "Single",*

*[CrossFilteringBehavior] = "BothDirections", "Both",*

*[CrossFilteringBehavior] ),*

*"Status", IF( [IsActive] = TRUE(), "Active", "Inactive" )*

*)*

*RETURN*

*\_Result*

| ***ID*** | ***From*** | ***To*** | ***Cardinality*** | ***Cross-Filter Direction*** | ***Status*** |
| --- | --- | --- | --- | --- | --- |
| *1* | *Invoices[Customer ID]* | *Customers[Customer ID]* | *Many-to-One* | *Single* | *Active* |
| *2* | *Invoices[Invoice Date]* | *Dates[Date]* | *Many-to-One* | *Single* | *Active* |
| *3* | *Orders[Order Date]* | *Dates[Date]* | *Many-to-One* | *Single* | *Active* |
| *4* | *Orders[Product ID]* | *Products[Product ID]* | *Many-to-One* | *Single* | *Active* |
| *5* | *Orders[Region ID]* | *Regions[Region ID]* | *Many-to-One* | *Single* | *Active* |
| *6* | *Orders[Customer ID]* | *Customers[Customer ID]* | *Many-to-One* | *Single* | *Active* |

## Measures

*A manually-entered* ***Key Measures*** *table was added and all measures were located therein.*

*To format all measures in a Power BI model:*

1. *Open [Power BI Desktop] and the PBIX file of interest*
2. *Open the [Tabular Editor V2] (Version 2.13 or higher) external tool*
3. *Select the [C# Script] tab*
4. *Enter the below in the C# Script window, then click the [Run] button*

*bool shortFormat = false;*

*bool skipSpaceAfterFunctionName = true;*

*FormatDax(Model.AllMeasures, shortFormat, skipSpaceAfterFunctionName);*

*Method:*

*List all measures in the model; the following can be used in DAX Query View from within Power BI to retrieve the measure names and expressions:*

*EVALUATE*

*VAR \_Result =*

*SELECTCOLUMNS(*

*INFO.VIEW.MEASURES(),*

*"Documentation", [Name] & " = " & [Expression]*

*)*

*RETURN*

*\_Result*

*Alternate Method 1 (DAX Studio): To list all the measures in a Power BI model:*

1. *Open [Power BI Desktop] and the PBIX file of interest*
2. *Open the [DAX Studio] external tool*
3. *Right-click on any table in the [Metadata] tab of the [Query] pane and select [Define All Measures (All Tables)*

*Alternate Method 2 (TMDL): To list all the measures in a Power BI model (including format strings):*

1. *Open [Power BI Desktop] and the PBIX file of interest*
2. *In the left sidebar, select [TMDL View]*
3. *Drag the desired table (containing the measures) into the TMDL window*
4. *Remove unwanted lines (createOrReplace, table, lineageTag, partition, annotation)*

*Total Sales = SUMX( Orders, Orders[Quantity] \* RELATED( Products[Unit Price] ) )*

*Total Costs = SUMX( Orders, Orders[Quantity] \* RELATED( Products[Unit Cost] ) )*

*Total Profit = [Total Sales] - [Total Costs]*

*Profit % = DIVIDE( [Total Profit], [Total Sales], 0 )*

*-- end of fragment*