



# Power BI for Enterprise Solutions Workshop

Level: Intermediate

January 25, 2024

**Paul Turley**

Director, Principal Consultant,  
3Cloud Solutions  
[paul@IntelligentBiz.net](mailto:paul@IntelligentBiz.net)



# THANK YOU



Platinum



Gold



Lucient



Measure Killer



Silver



Bronze



# Evaluations, evaluations...



[https://evals.datagrillen.com/evals\\_vienna.aspx](https://evals.datagrillen.com/evals_vienna.aspx)

# Paul Turley



## My Values:

- Family
- Community, Mentorship
- Colleagues, Clients, Career

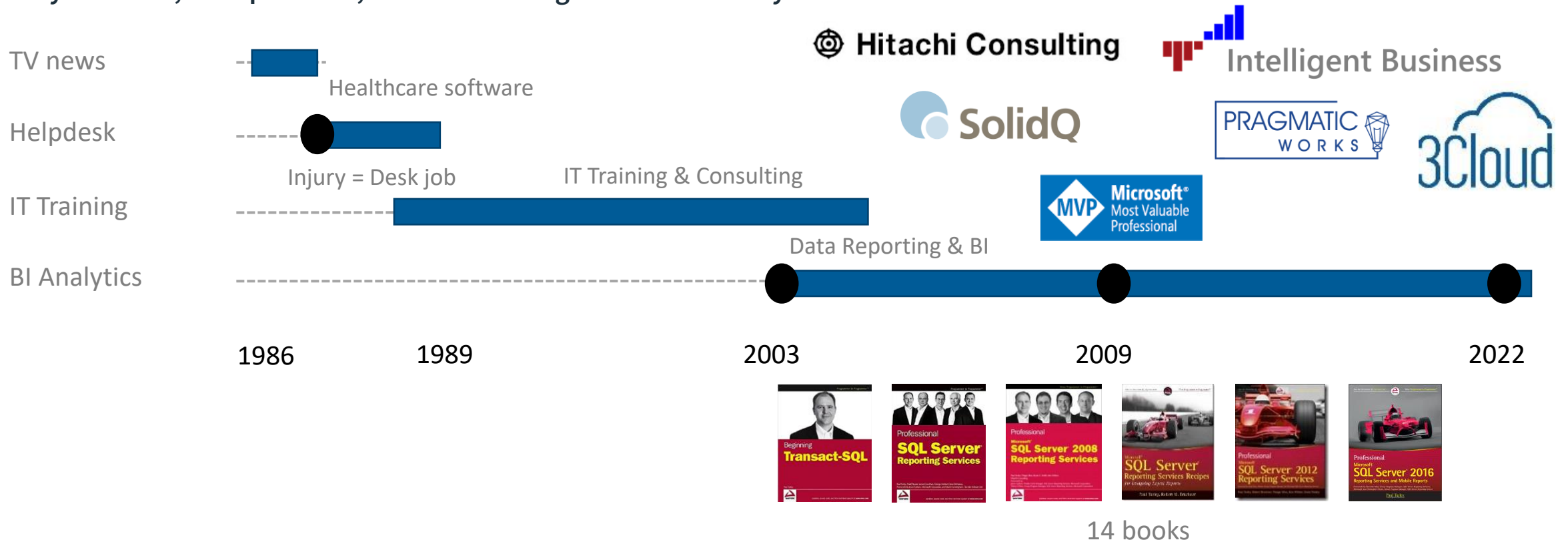


## Conferences & Presentations:

- PASS Summit
- SQL Saturdays
- User Groups
- MBAS, Business Analytics Summit, Live!360
- SqlServerBi.Blog – ~2 million viewers

Principal Consultant, Microsoft Data Platform MVP

~25 years in IT, data platform, Business Intelligence & data analytics



# SOME BACKGROUND

## ABOUT THIS WORKSHOP

- This presentation developed as a full-day preconference
- Expanded to optional 2-day training
- Scaled-down a version to 3-4 hours
- Way too many slides! - Not going to present all of them
- Not going to complete every exercise
- OK to skip ahead and use completed solution files

Personal BI

Team BI

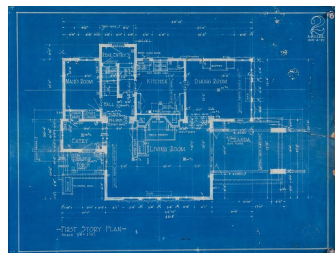
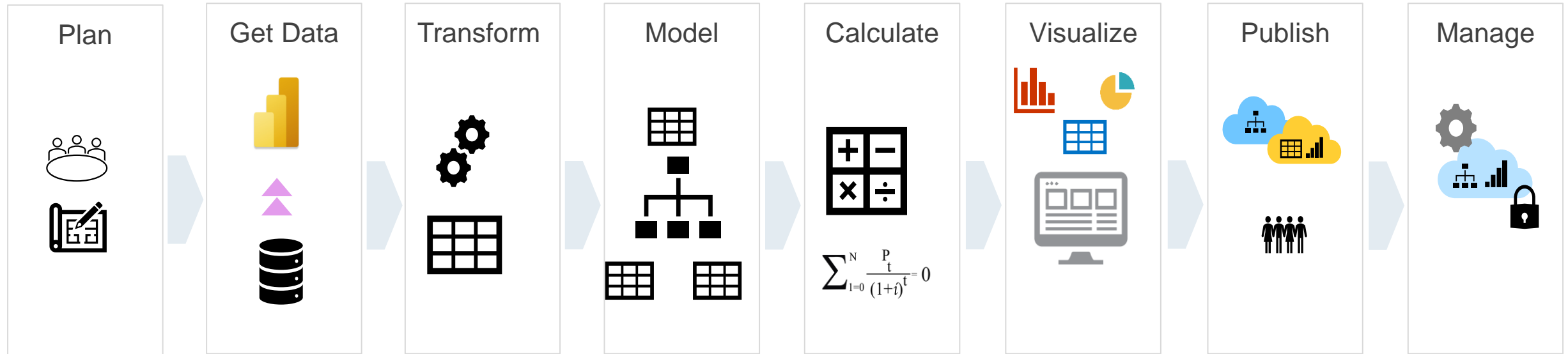
Enterprise BI

**Power BI Premium**

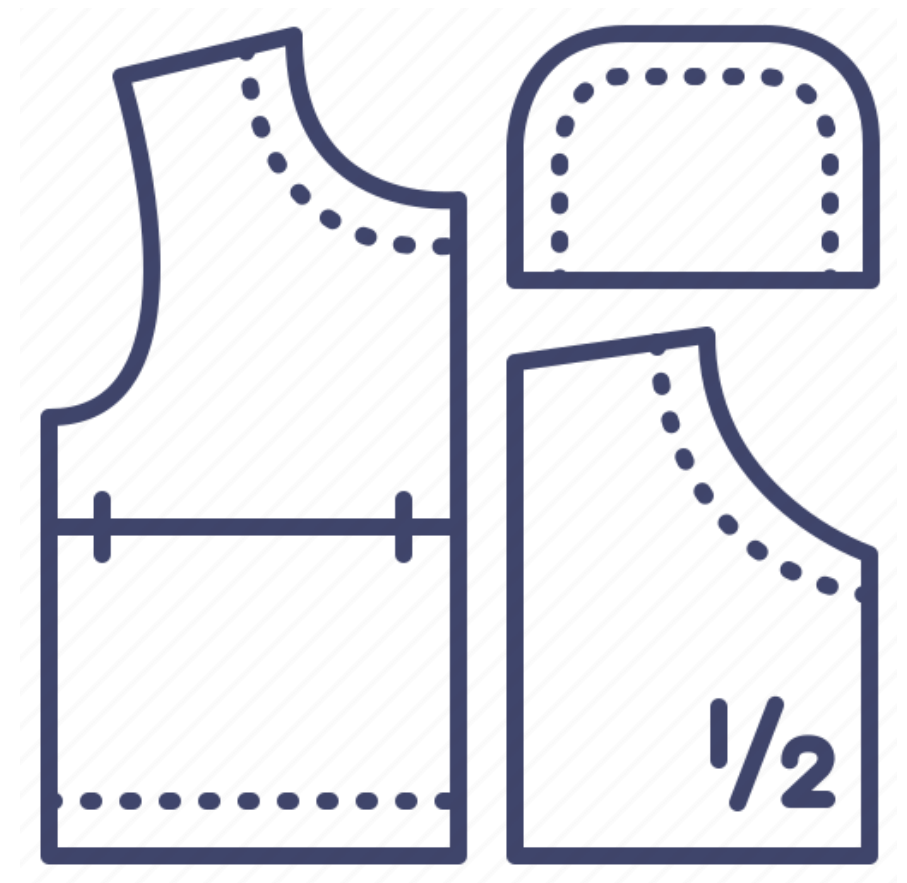
**Microsoft Fabric**

- Futureproof
- Scalable
- Extensible
- Governance
- Team Development
- Versioning
- Deployment Process

# The Business Intelligence Process



# Objectives: Apply enterprise-scale recipes & patterns





Futureproof Solutions

Extensible Design

# Futureproofing Power BI Solutions

*Difficult to see; always in motion, the future is.*



Doing Power BI the Right Way: 1. Futureproofing Power BI solutions | Paul Turley's SQL Server BI Blog  
<https://sqlserverbi.blog/2020/07/29/doing-power-bi-the-right-way-1-futureproofing-power-bi-solutions/>



# Extensible Design

GOING TO EXTREMES AND FINDING BALANCE

One model per Report

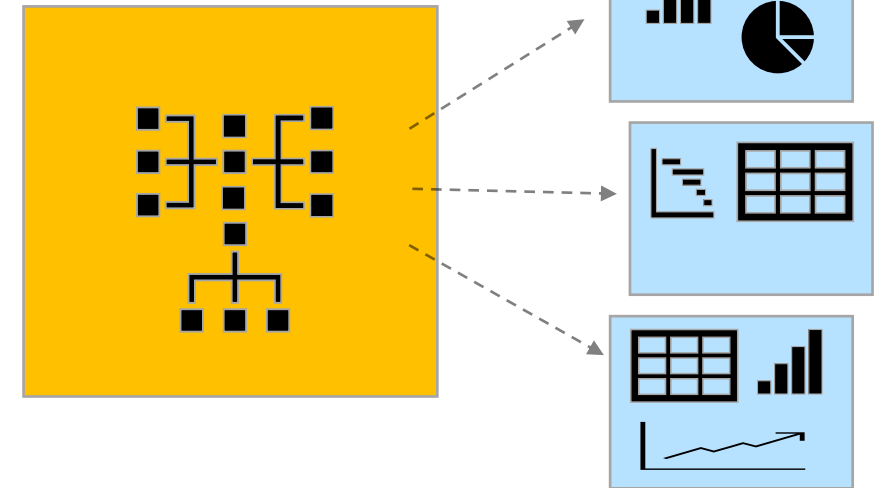


Flexibility

Redundancy & inconsistency



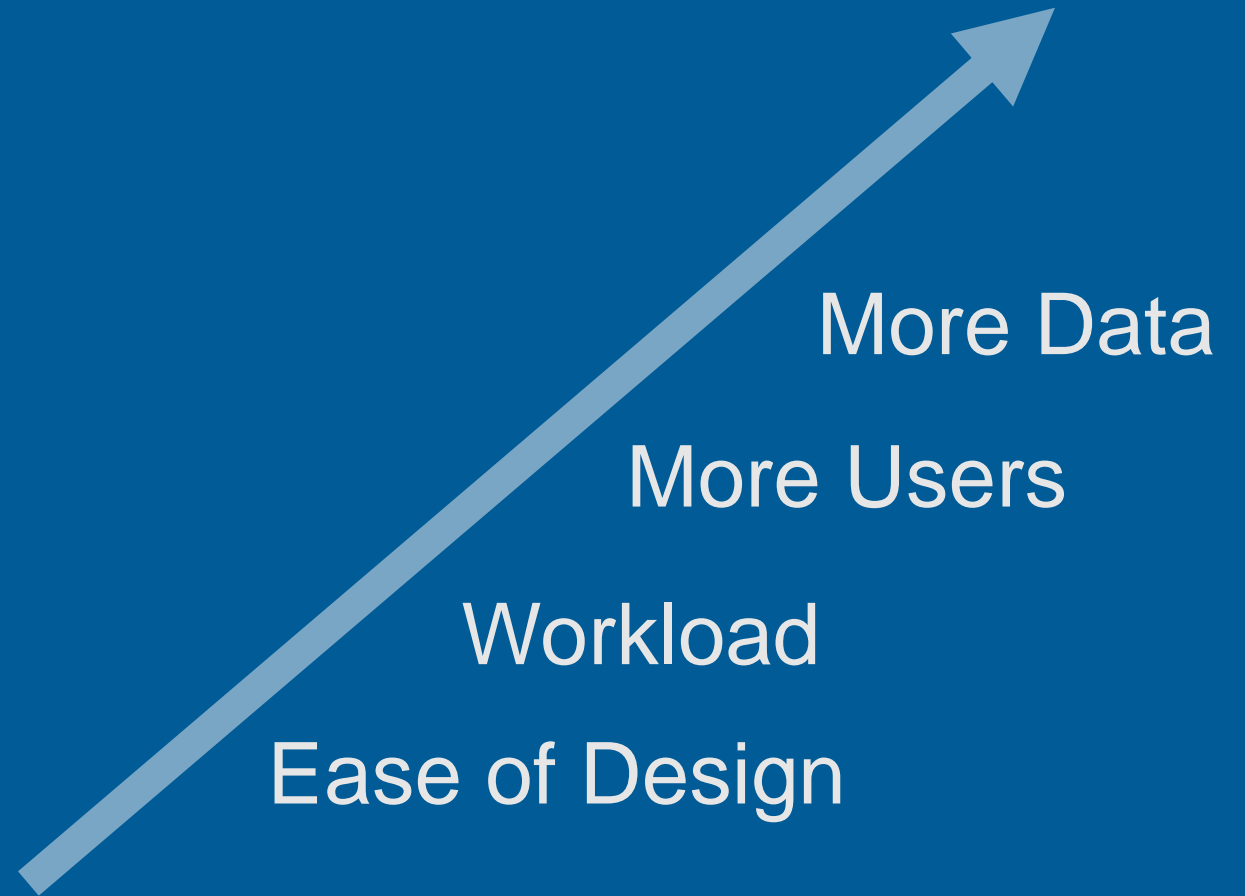
One model with all the answers



Central Control

Complexity & inflexibility

Scalability



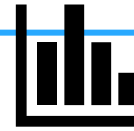
# Self-Service BI Mindset

We Have Some Data



This data looks OK. Let's import it.

We Need a Report



What visuals should be used?

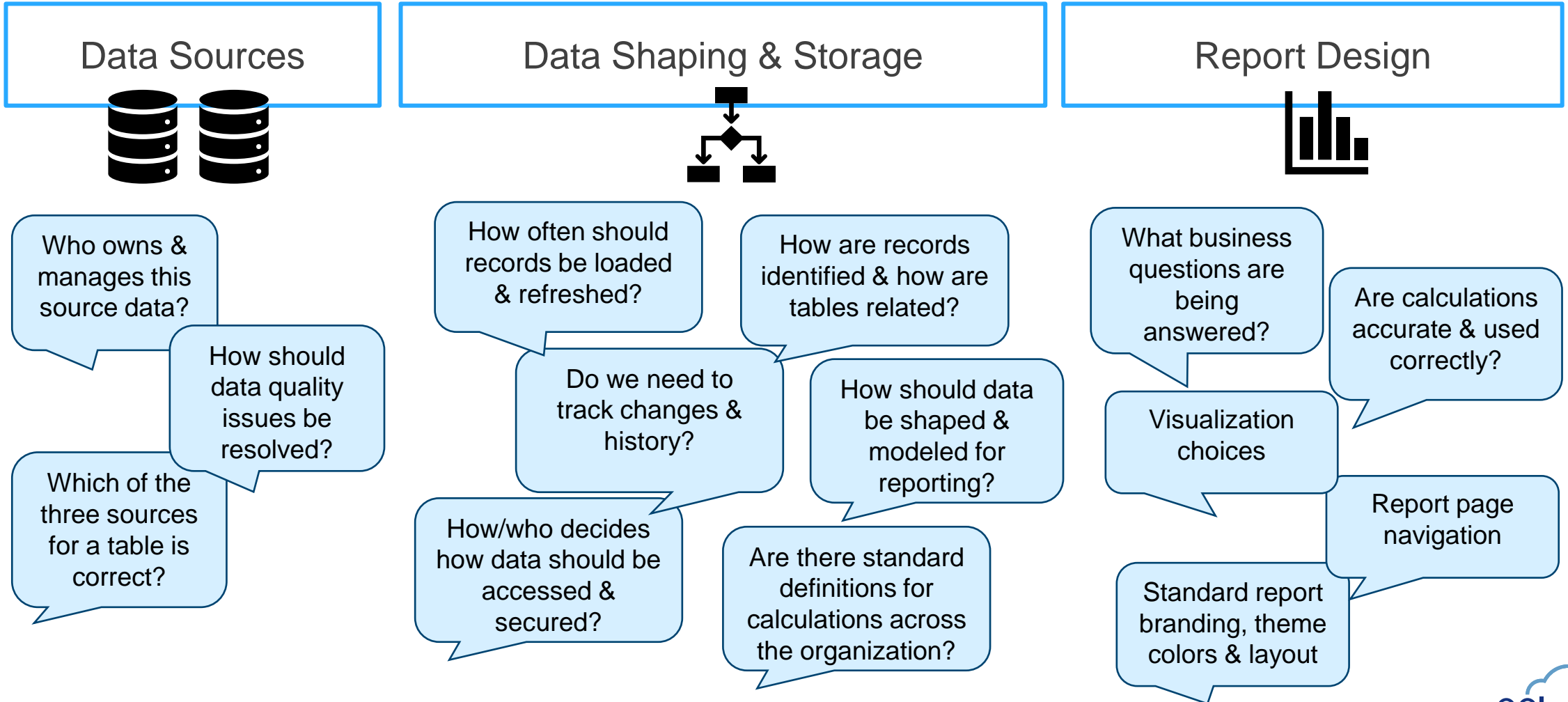
What color should it be?

How to navigate between pages

How can users export the report data?

Can we design the report to look like the old one?

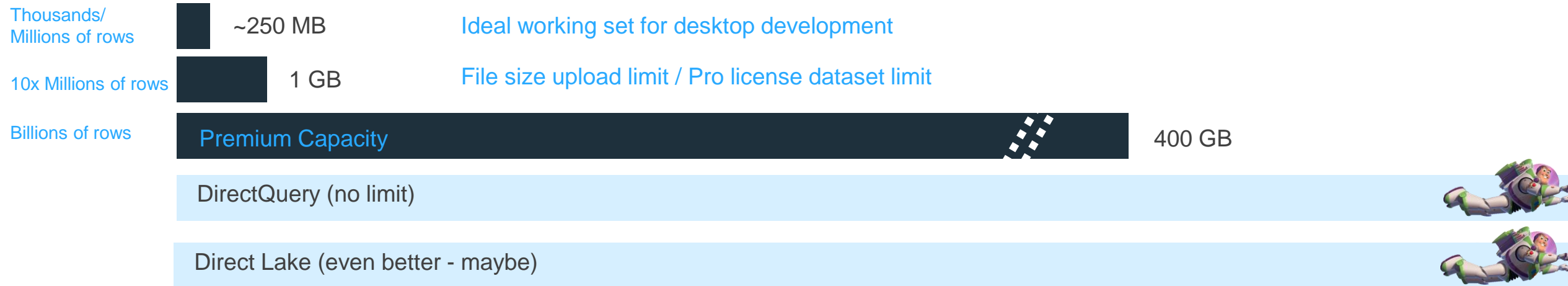
# Enterprise BI Mindset



# How Much Data?

CAN WE MANAGE WITH POWER BI...

- Data models typically require a fraction of the source data storage size (due to column selection & compression)
- Import models run in-memory & are typically very fast.
- DirectQuery enables access to large tables with a performance cost, OK with little grouping and when only light calculations are needed.
- Advanced features enable optimizations over DirectQuery, like aggregations & composite models.



# Team Development Versioning & File Management





# Manage Power BI Desktop Files

- **Store files in a centrally managed network-assessable folder**  
The storage folder should support automatic backup and recovery in the case of storage loss.
- **Report and dataset developers must open files from the Windows file system**  
Files must either reside in or be synchronized with the Windows file system.
- Files containing imported data typically range in size from 100 to 600 MB. Any shared folder synchronization or disaster recovery system should be designed to effectively handle multiple files.

## Options:

- OneDrive For Business (shared by team, with folder synchronization).
- SharePoint or SharePoint Online (with folder synchronization).
- GitHub and/or VSTS with local repository & folder synchronization. If used, Git must be configured for large file storage (LFS) if PBIX files are to be stored in the repository.
- NEW – Power BI Project with Git integration.

Reduce file size to under 100 MB. We'll talk about this a bit later.

# Clearing DevOps, Versioning & Team Development Blockers

- Power BI file size
  - Keep PBIX files under 100 MB, as a guideline
- Separate data model PBIX from Report PBIX when:
  - Project has matured
  - Parallel data model & report development is feasible
- Treat PBIX file as a binary file
  - Don't compare, split or merge with app dev schema compare tools
  - More DevOps & object-level integration is coming. Be patient
  - Option: manage data model as .BIM (but realize the trade-offs).
- Store BI project files in simple code repository
  - OneDrive for Business
  - Git / Azure DevOps

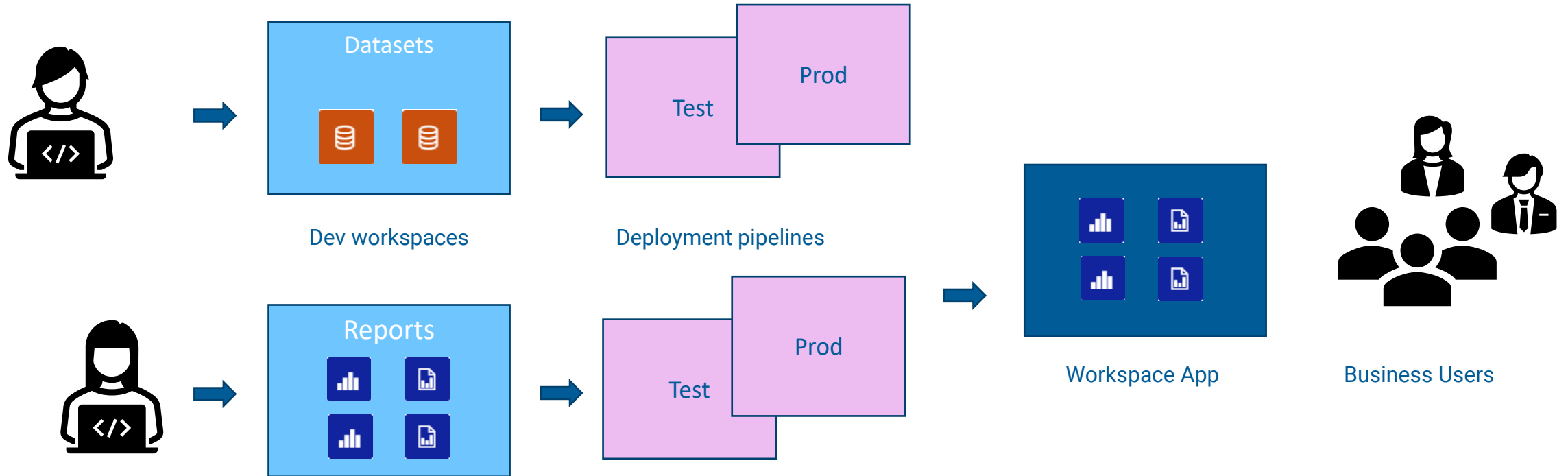
# Data Governance

how it relates to Power BI solutions

# Dataset, Model and App Deployment & Delivery Cycle

REPORT DELIVERY EN MASS

## Certified Reports on Certified Datasets



Developers deploy datasets and reports to workspaces

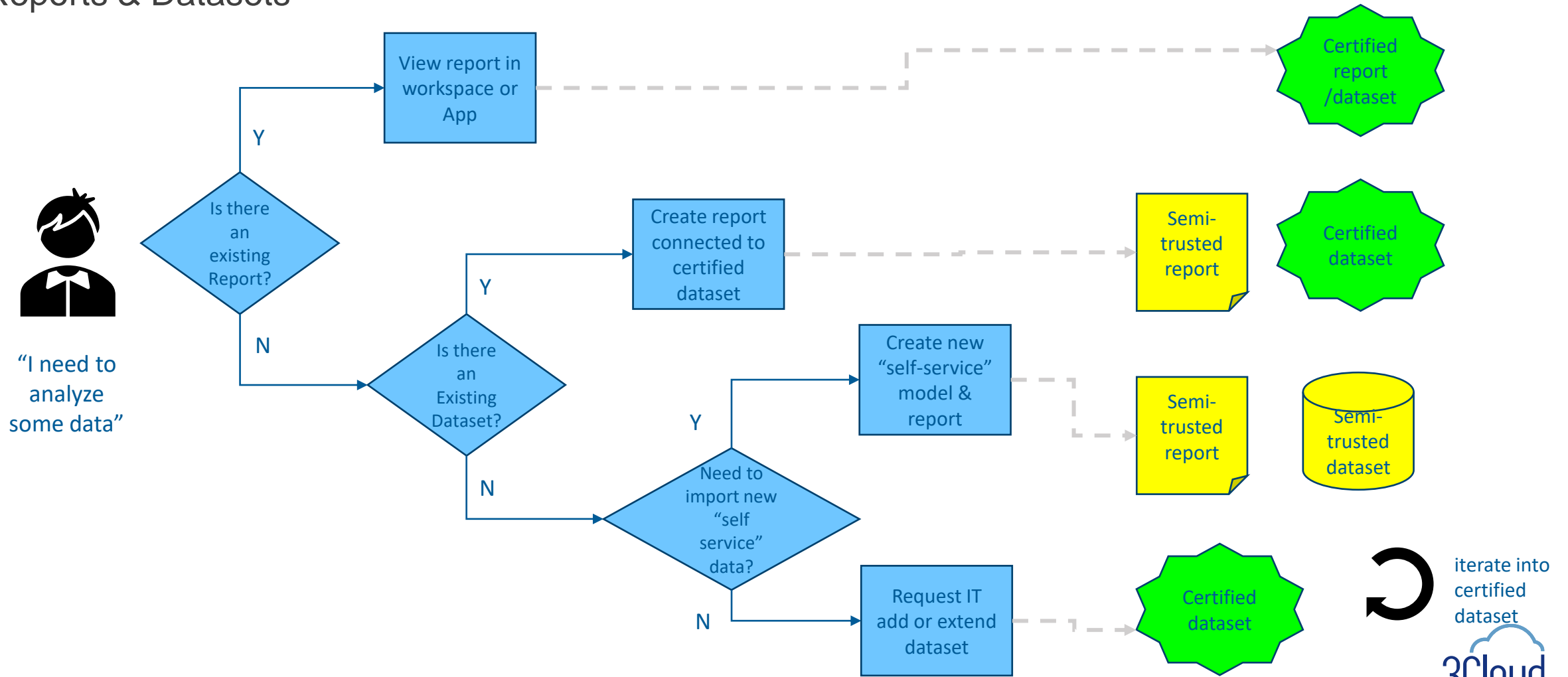
Workspaces are promoted through Deployment Pipelines after test acceptance

Report workspace delivered as an App

# Certified & Self-Service Decision Tree

## REPORTING & ANALYTICS PATHS

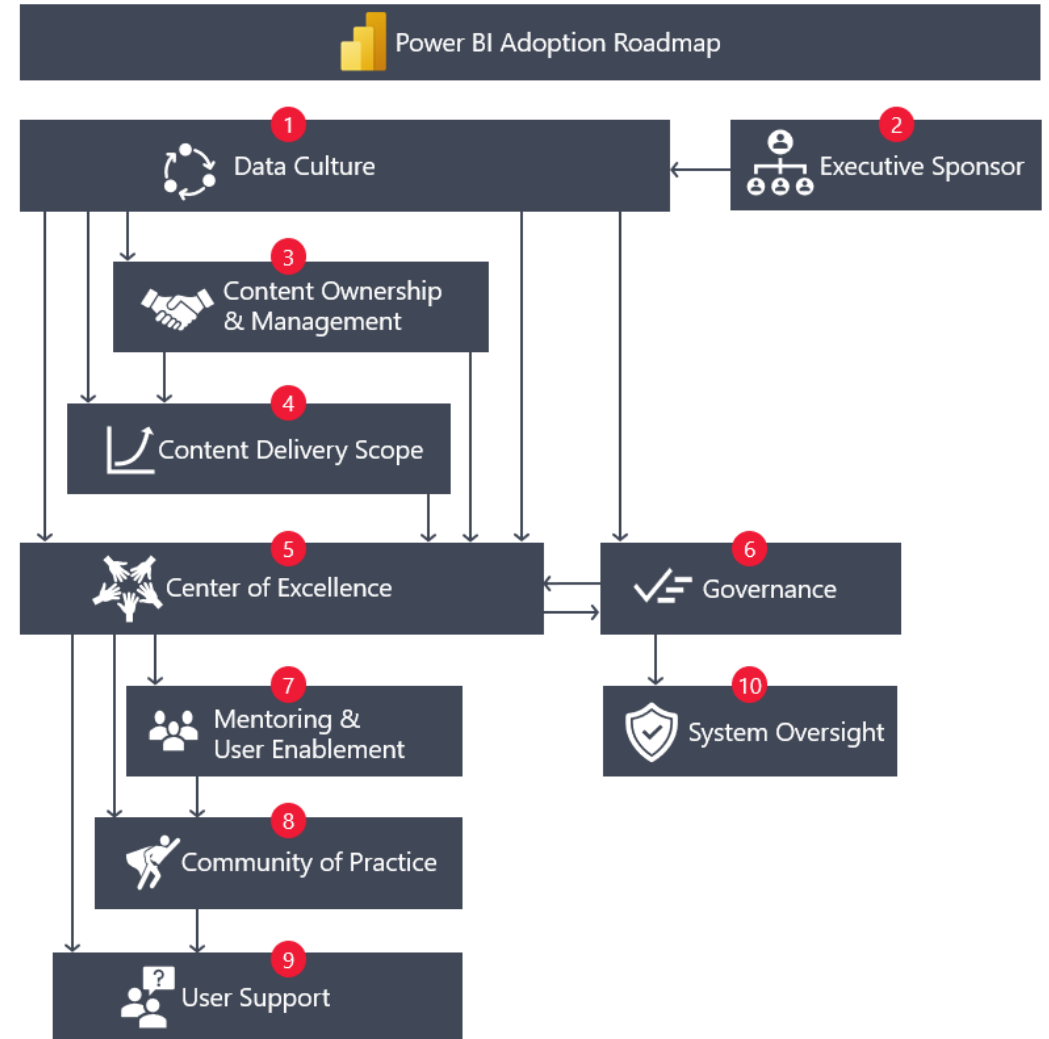
### Reports & Datasets



# Power BI Adoption

Power BI adoption should blend with the organizational data governance strategy

- Data culture
- Sponsorship
- Content ownership (Data Stewardship)
- Content delivery (Certification, self-service)
- Community of Practice / training & knowledge sharing
- Security & system oversight



<https://docs.microsoft.com/en-us/power-bi/guidance/powerbi-adoption-roadmap-overview>

# Hands On Exercises

Lab Exercise Solution Files:

Part 1      Part 2      Part 3      Part 4

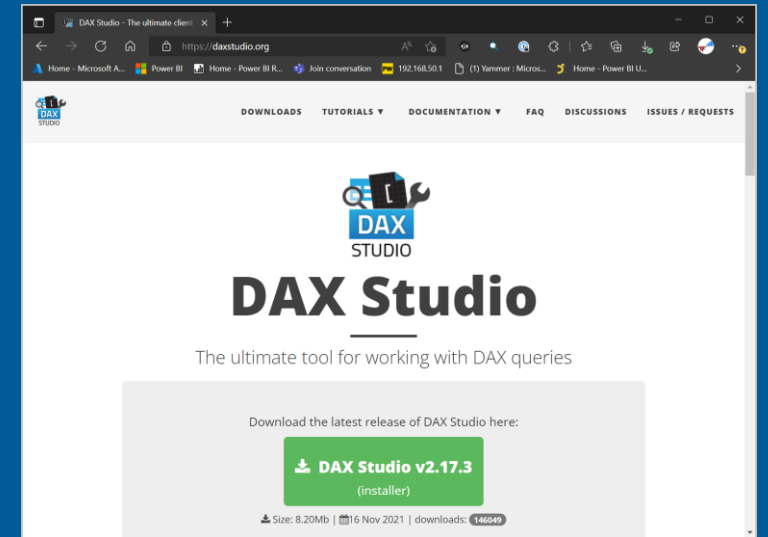
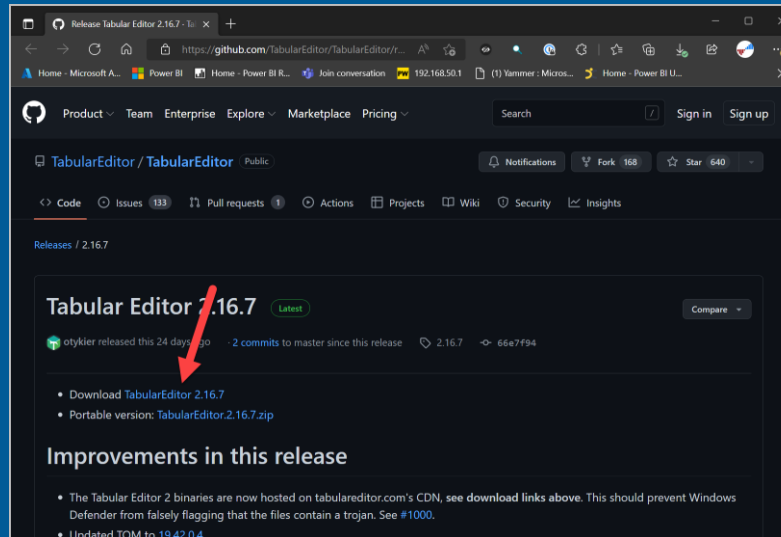
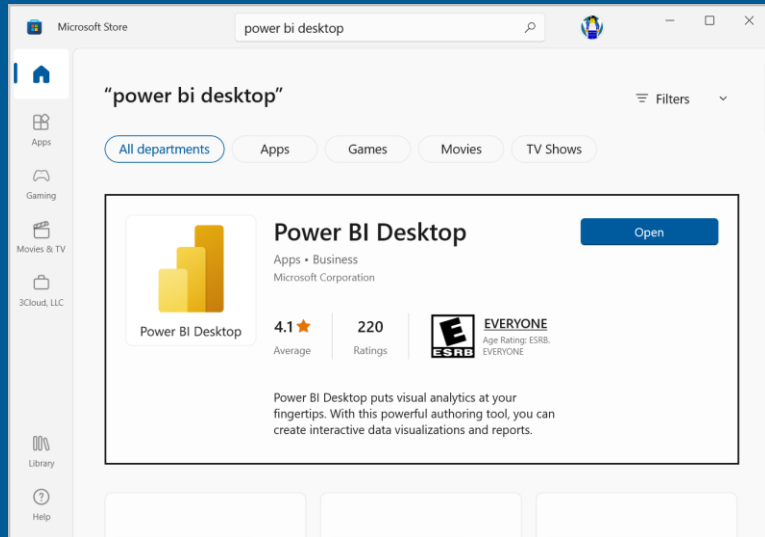


- Prepare source data
- Connect to data sources
- Create parameters
- Filter records

- Transform data
- Import tables
- Create data model
- Create base measures

- Create advanced measures
- Visualize
- Deploy to cloud service
- Deliver to user audience

# Setup



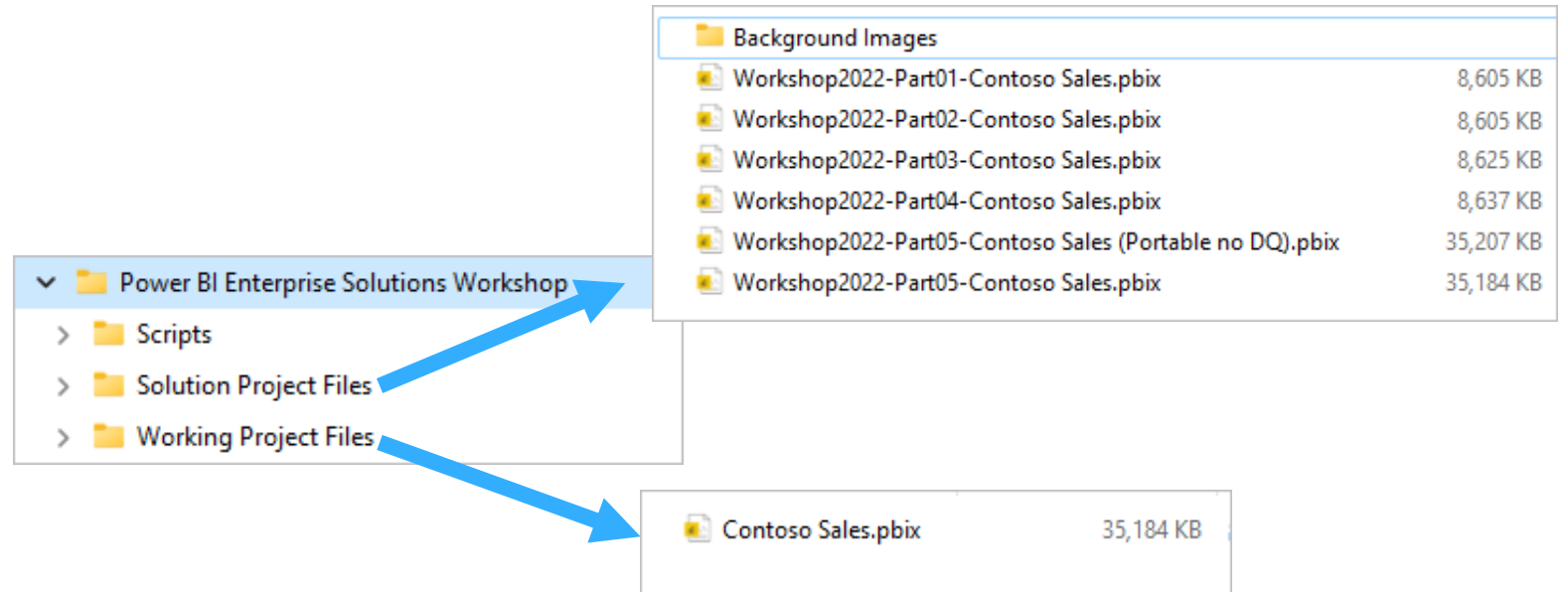


# Lab Project Files & Folders

Using Windows File Explorer, copy the folder **Power BI Enterprise Solutions Workshop** from the USB drive to the C: drive on your computer.

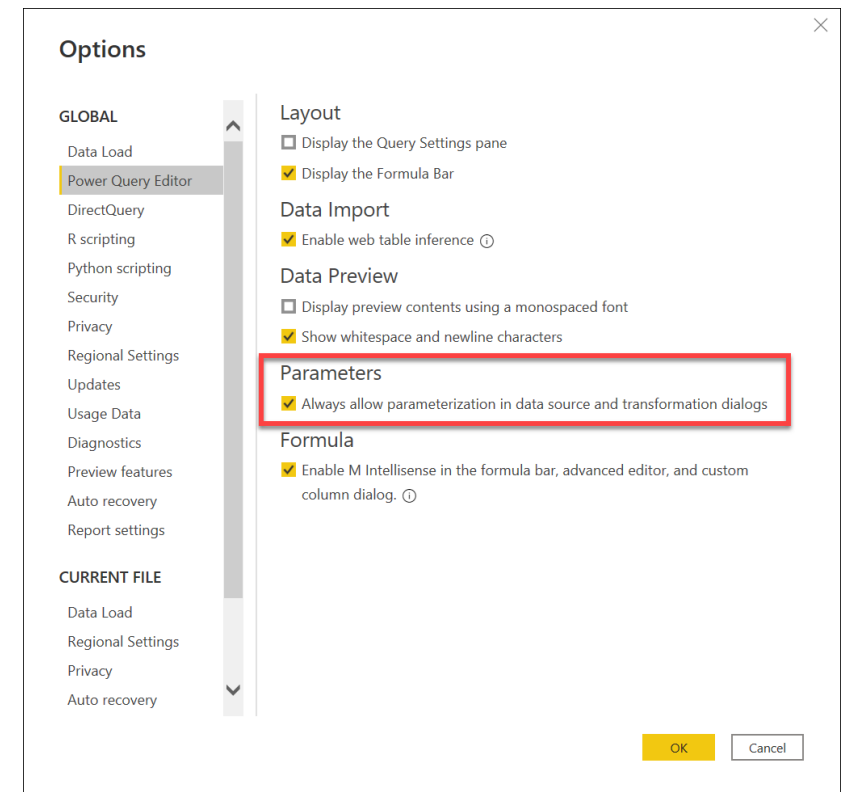
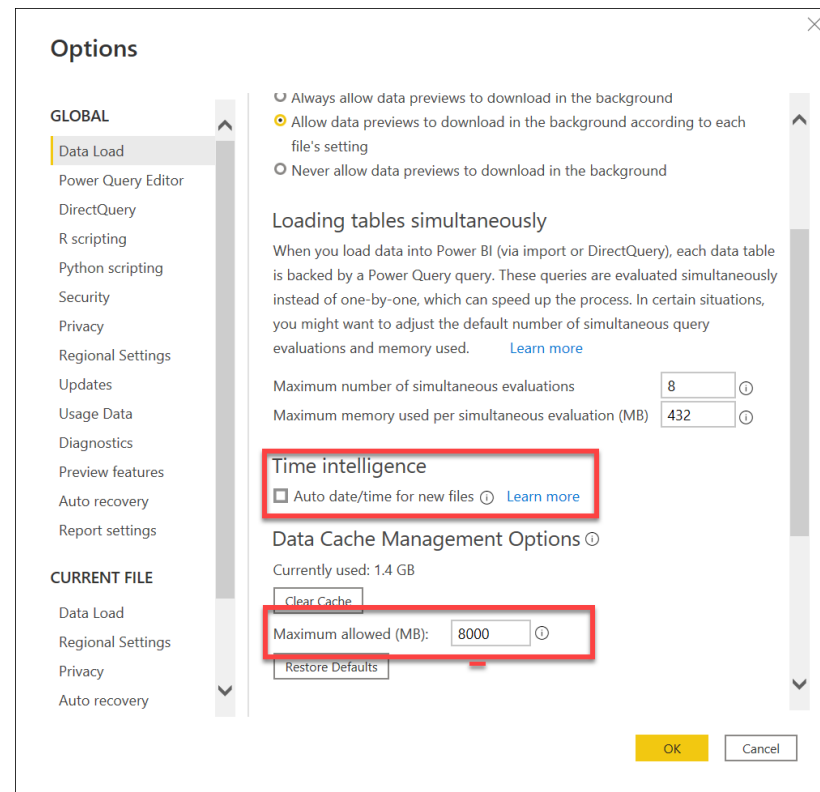
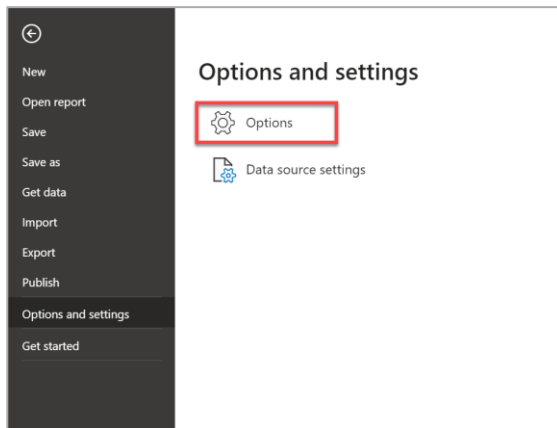
Follow instructions to:

- Work in the **Working Project Files** folder
- Copy solution files from the **Solution Project Files** folder



You will create this file

# Power BI Desktop Options



# Hands On Exercise

## Part 1: Connect & Transform Data

Part 1



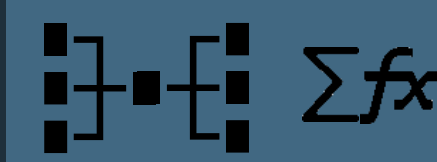
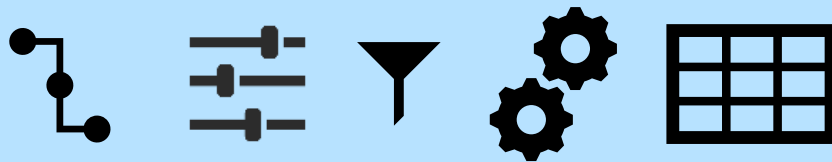
Part 2



Part 3



Part 4



- Prepare source data
- Connect to data sources
- Create parameters
- Filter records

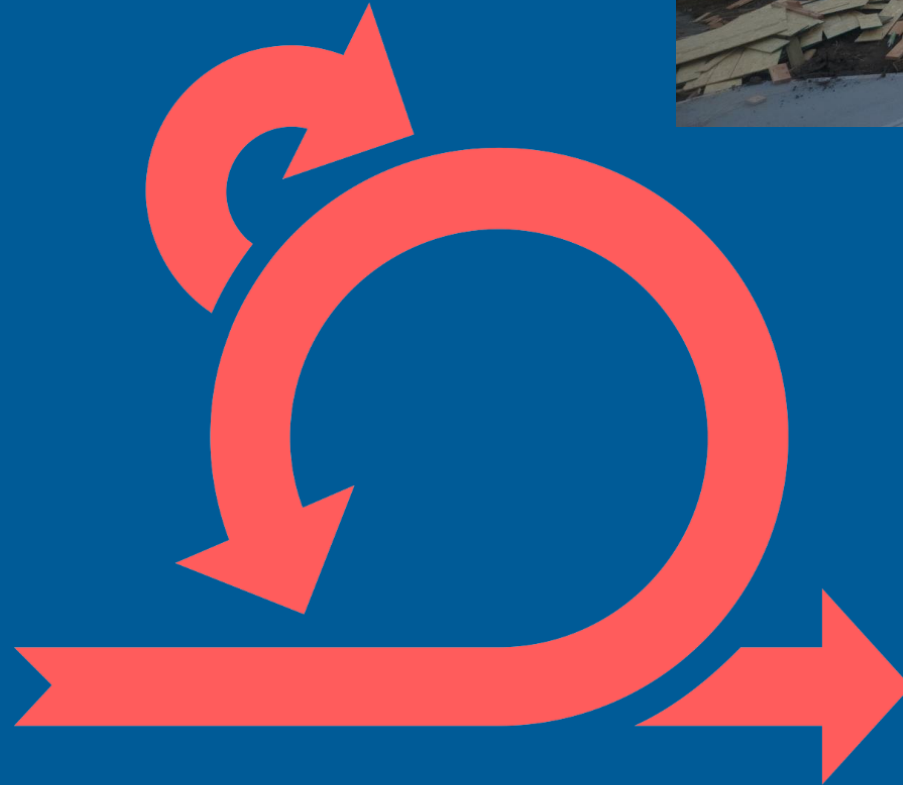
- Transform data
- Import tables
- Create data model
- Create base measures

- Create advanced measures
- Visualize
- Deploy to cloud service
- Deliver to user audience

# Iteration 1

## Fact tables & related dimensions:

- Online Sales
- Store Sales



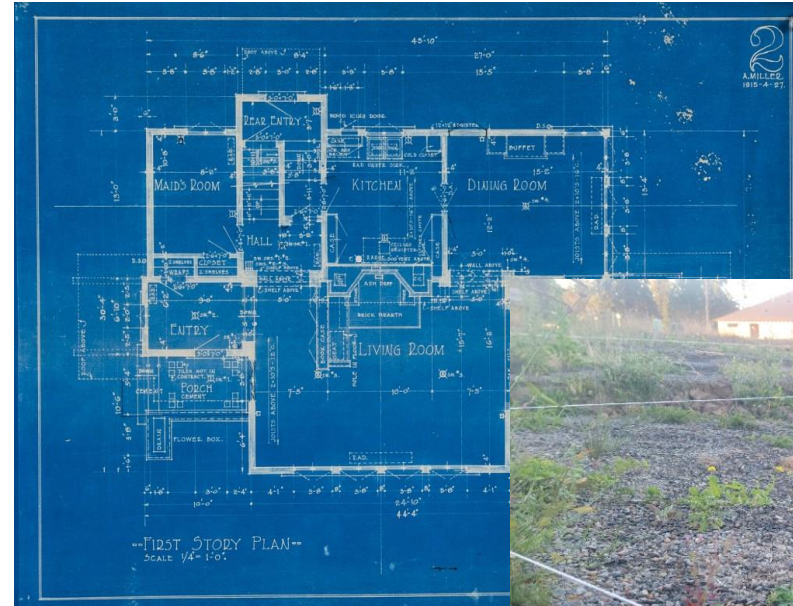
# Iteration 1 Requirements

## FUNCTIONAL REQUIREMENTS

The Sales organization needs...

- Online Sales Qty
- Online Sales Amt
- Store Sales Qty
- Store Sales Amt

Executive Leadership needs...



# Iteration 1 Requirements

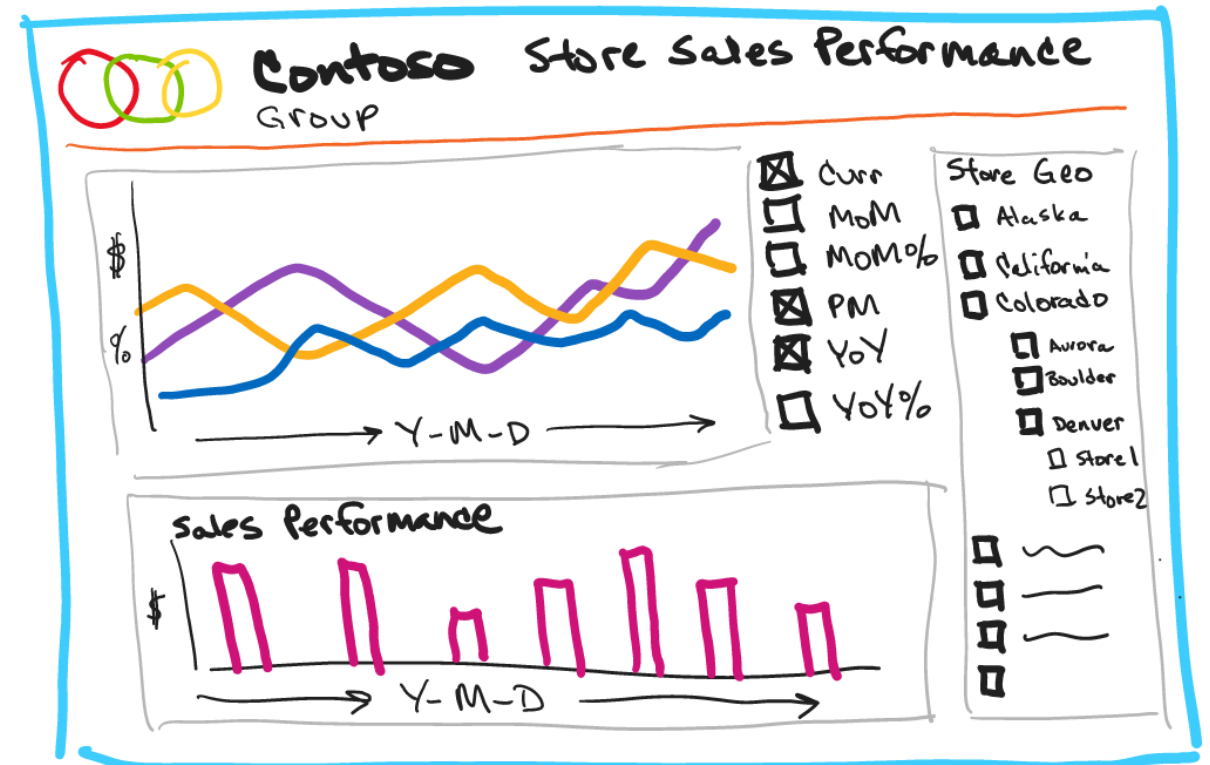
## REPORT CAPABILITIES & WIREFRAME

### Functional Report Requirements

“Back of the napkin report design”

Business questions:

- What is the Sales Amount grouped by Year or Month or Day, for a specific period of time.
- Trend chart should allow a user to select a time series metric (like Month Over Month, Month Over Month % Change, Month To Date, Prior Month, Year To Date) & then show them along the axis of Years, Months or Days.
- All of the data on the report can be filtered by store regions, such as State, City or an individual Store.



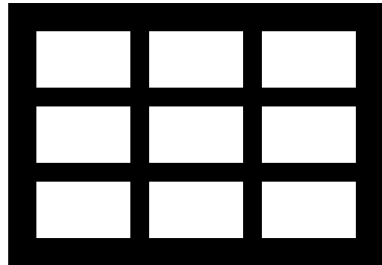
# Iteration 1 Requirements

## DIMENSIONAL MATRIX

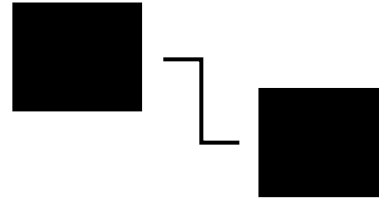
	Date	Account	Customer	Product	Store
Online Sales	X		X	X	
Online Sales Qty					
Online Sales Amt					
Store Sales	X			X	X
Store Sales Qty					
Store Sales Amt					

# Data Modeling Essentials & Best Practices in Power BI

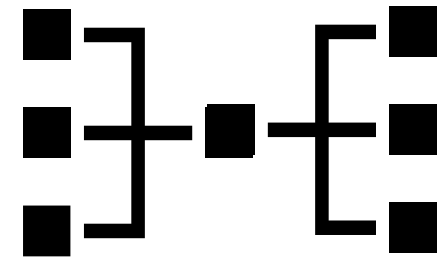
## Flat Model



## Master/Detail



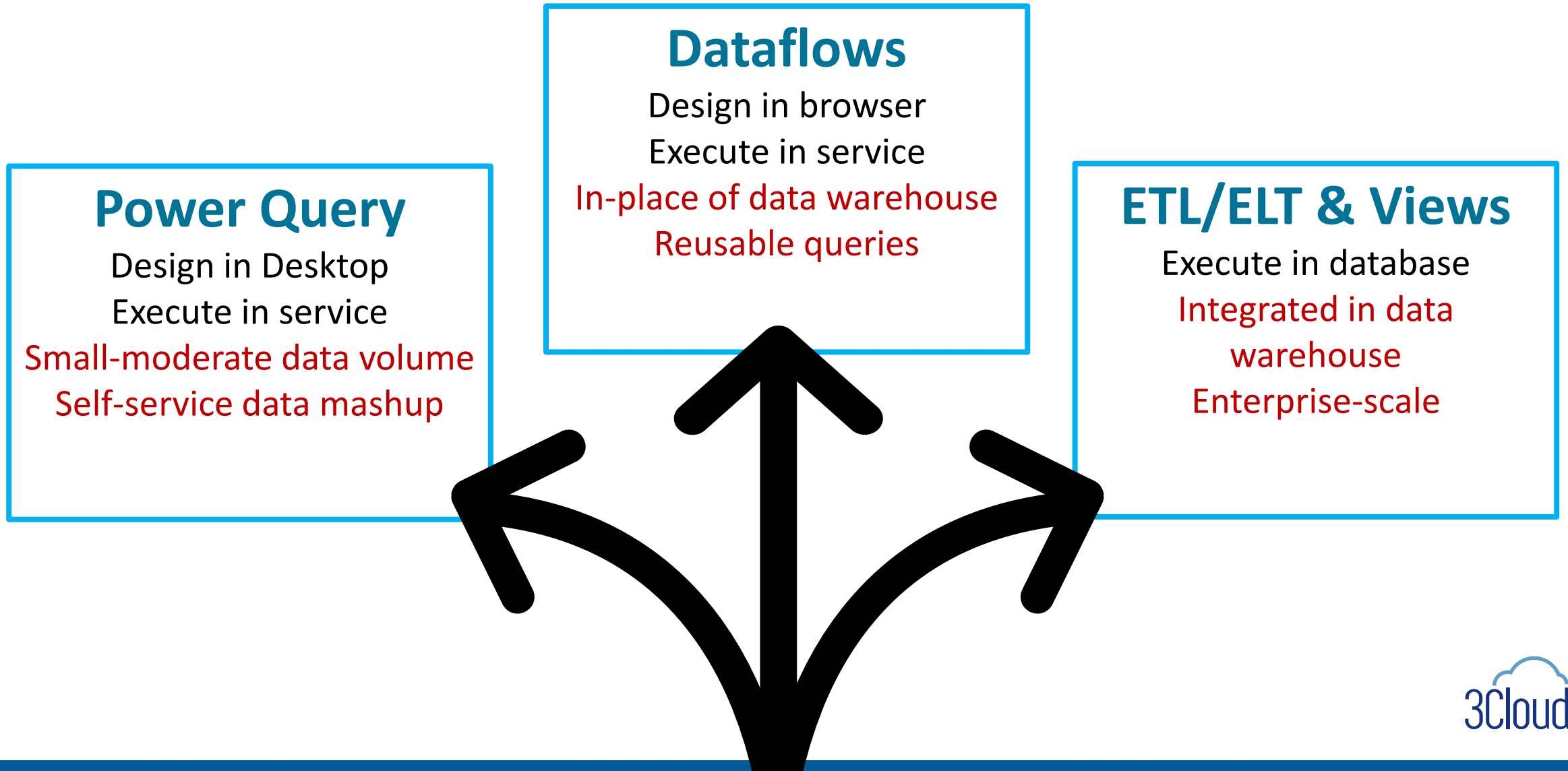
## Dimensional



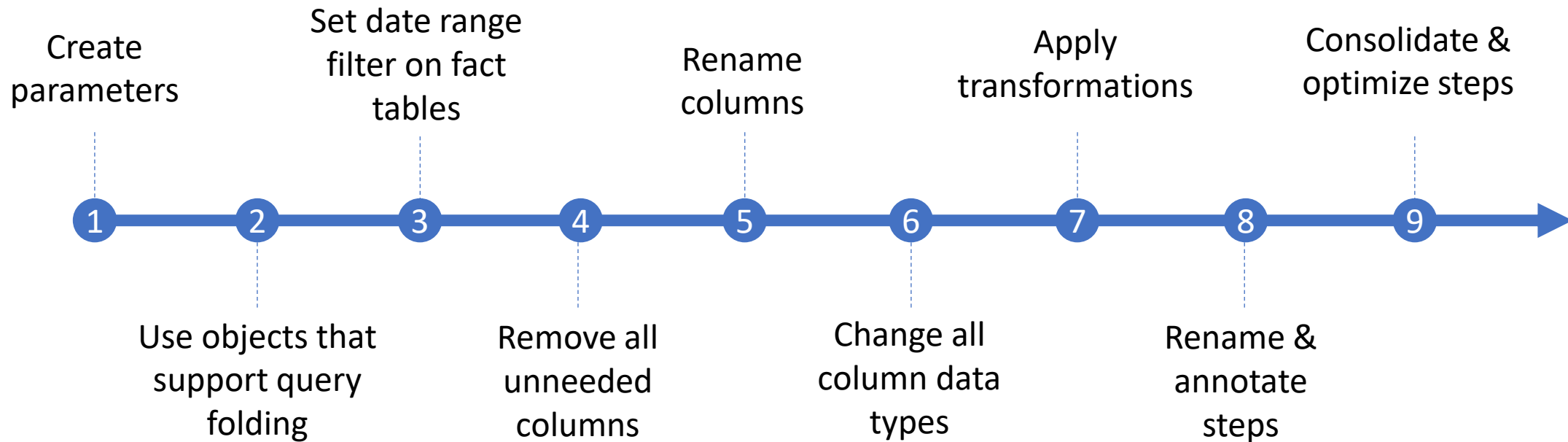


# Transformation Options

TO PREPARE & SHAPE DATA FOR MODELING



# Preparing, shaping & transforming source data using Power Query



[Doing Power BI the Right Way: 2. Preparing, shaping & transforming source data | Paul Turley's SQL Server BI Blog](https://sqlserverbi.blog/2020/08/16/doing-power-bi-the-right-way-2-preparing-source-data/)  
<https://sqlserverbi.blog/2020/08/16/doing-power-bi-the-right-way-2-preparing-source-data/>

# Object Naming Conventions

## Database Developer



- Table, field/column
- Measure names
- Code-friendly na
- Cryptic object na
- Pascal case, Cam
- notation,

## BI Solution Designer



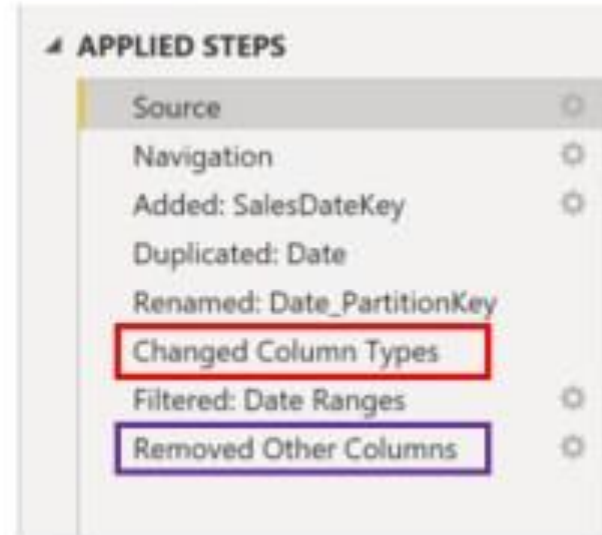
### Balance

- All objects exposed to users should have friendly names
- Hide key columns & utility objects
- Hide numeric columns & create friendly named measures

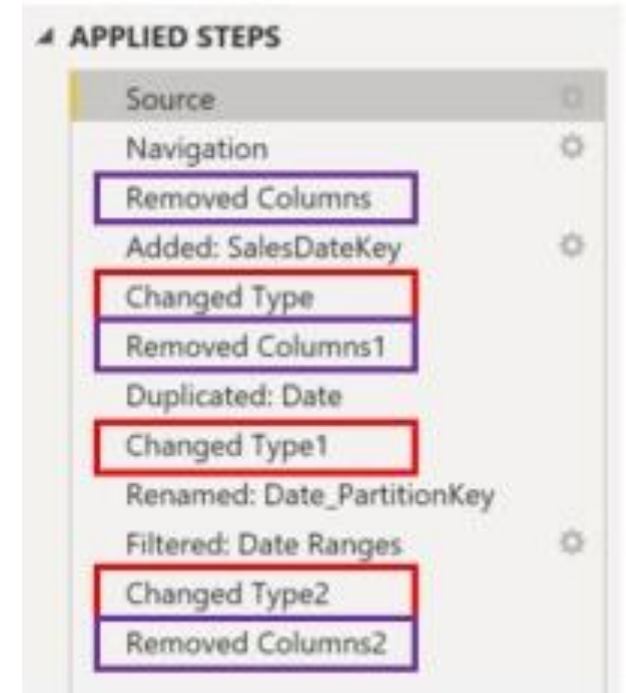
# Optimize Power Query

- Test transformations with large data volumes
- Pivot, unpivot & Transpose actions are costly & may not work effectively with large data volume
- Complex and “creative” transformations might work in Desktop or with small data volumes but not in production
- Web service API calls & nested M functions may not work in the service.

Do this:



...not this:

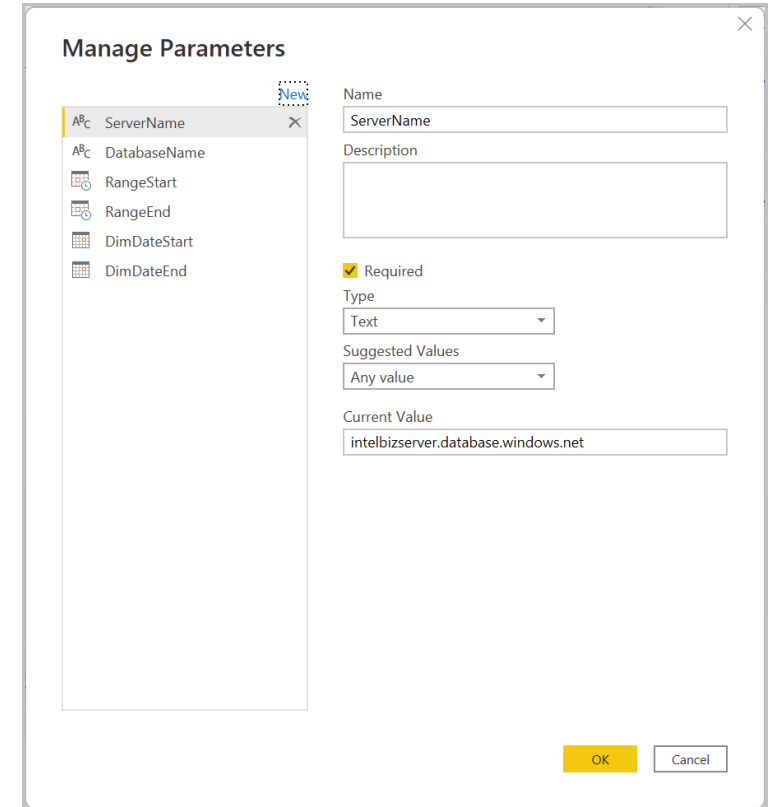
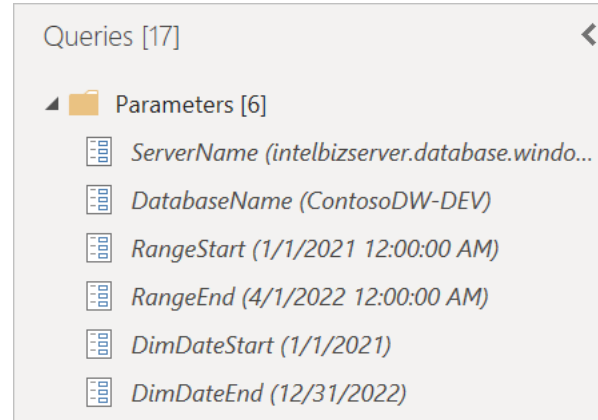
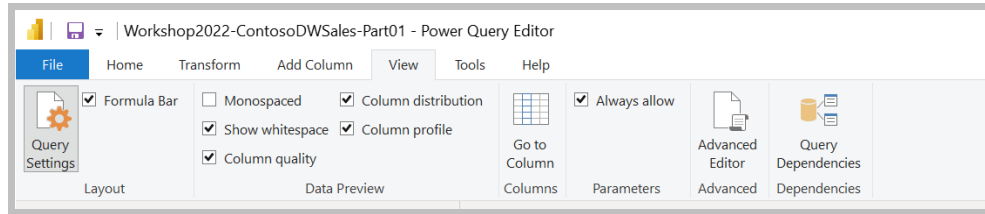


This query, with redundant, dependent steps; takes three times longer to load records

# Using Parameters Importing Dimensions Importing Facts



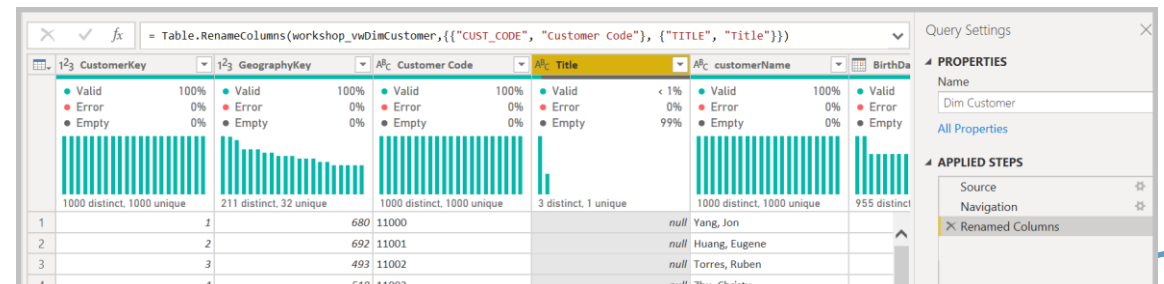
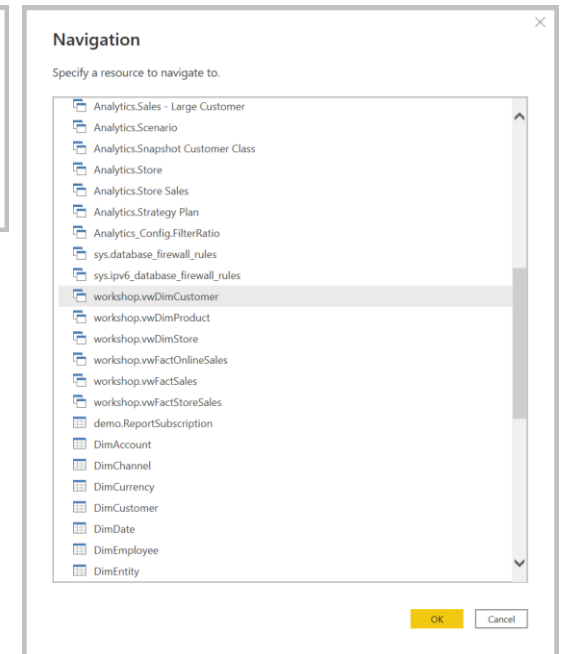
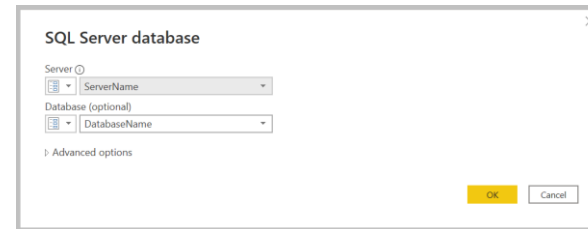
# Add Query Parameters



# Importing Dimension Data

- Dim Customer
- Dim Product
- Dim Store
- Dim Date

- Parameterize connection information
- Import from tables or views, not using in-line SQL statements
- Remove unneeded columns
- Rename columns to apply friendly names



# Power Query Optimization

- Power Query steps generate M code
- Formatted code can be easier to read and debug
- Generate steps to establish code pattern & then enhance code
- Code comments create tooltips in the designer

The screenshot displays the Power Query Advanced Editor window. The main pane shows the following M code:

```
let
    Source = Sql.Database(ServerName, DatabaseName),
    workshop_vwDimCustomer = Source[Schema="workshop",Item="vwDimCustomer"] [Data],
    // Rename all necessary columns, using friendly names
    #"Renamed Columns" = Table.RenameColumns(workshop_vwDimCustomer,
    {
        {"TITLE", "Title"},
        {"CUST_CODE", "Customer Code"},
        {"CustomerName", "Customer Name"},
        {"BirthDate", "Birth Date"},
        {"MaritalStatus", "Marital Status"},
        {"CustomerGender", "Customer Gender"},
        {"YearlyIncome", "Yearly Income"},
        {"TotalChildren", "Total Children"},
        {"NumberChildrenAtHome", "Number Children At Home"},
        {"CityName", "City Name"},
        {"StateProvinceName", "State Province Name"},
        {"RegionCountryName", "Region Country Name"},
        {"StreetAddress", "Street Address"},
        {"DateFirstPurchase", "Date First Purchase"},
        {"CustomerType", "Customer Type"},
        {"CompanyName", "Company Name"}
    })
in
    #"Renamed Columns"
```

The status bar at the bottom of the editor indicates "No syntax errors have been detected." and "955 distinct".

The Query Settings pane on the right shows the following configuration:

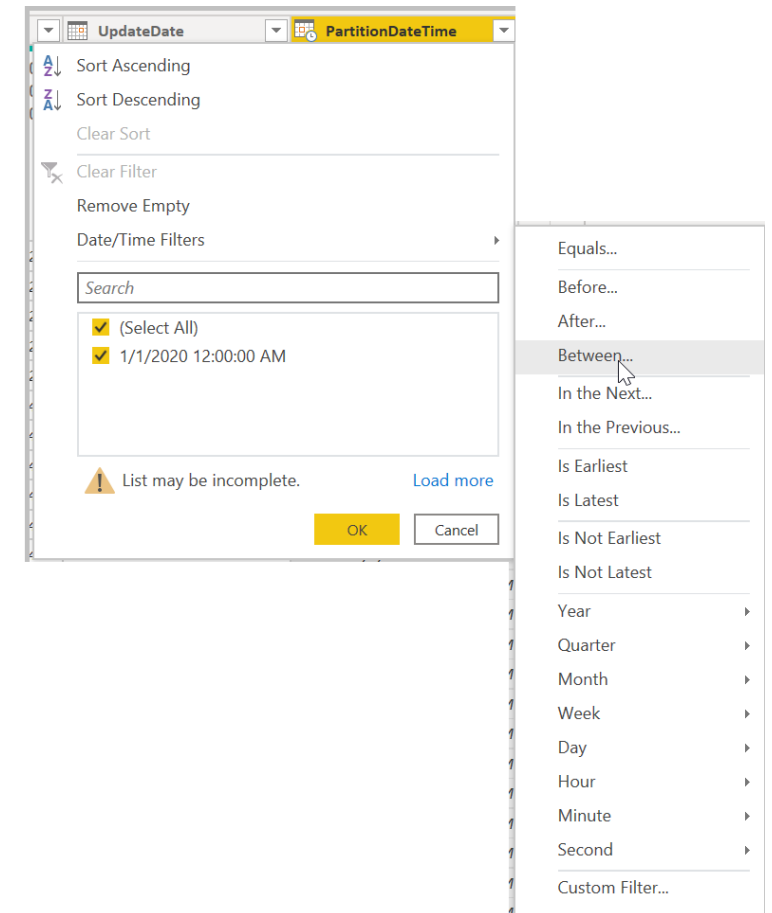
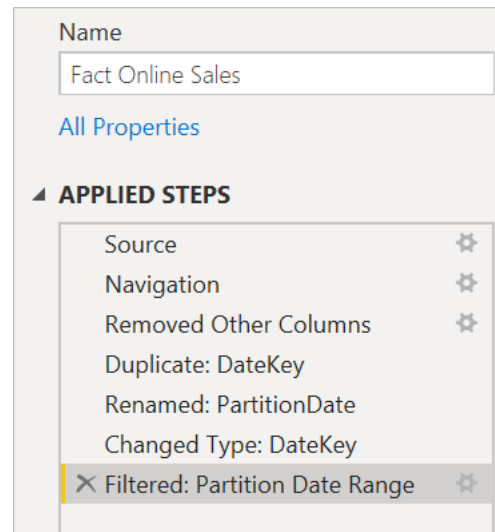
- PROPERTIES**
  - Name: Dim Customer
- APPLIED STEPS**
  - Source
  - Navigation
  - Renamed Columns** (selected)

A tooltip is visible over the 'Renamed Columns' step, containing the text: "Rename all necessary columns, using friendly names".



# Importing Fact Tables

- Tables contain tens of millions of rows
- Use date range parameters to reduce local working set and keep PBIX file small.
- PartitionDateTime:
  - Row count reduction
  - Incremental Refresh
- UpdateDate:
  - Detect changes
- No need to rename fact table columns



# Where to Perform Calculations

## Store Sales Amt

### If calculated at the row-level:

1. Perform calculation in the upstream ETL process & store value
2. Calculate at the source, in a view
3. Custom field in Power Query
4. Calculated column in DAX

**Store Sales Amt = SalesQuantity \* UnitPrice**  
(on a single row)

### If calculated outside of a single row context, or if the calculation on a single row is affected by filter context:

- Perform calculation in DAX as a measure

**Custom Column**

Add a column that is computed from the other columns.

New column name  
SalesAmount

Custom column formula ⓘ  
= [SalesQuantity] \* [UnitPrice]

Available columns  
DateKey  
StoreKey  
ProductKey  
PartitionDateTime  
SalesQuantity  
UnitPrice  
UnitCost  
DiscountQuantity

<< Insert

Learn about Power Query formulas

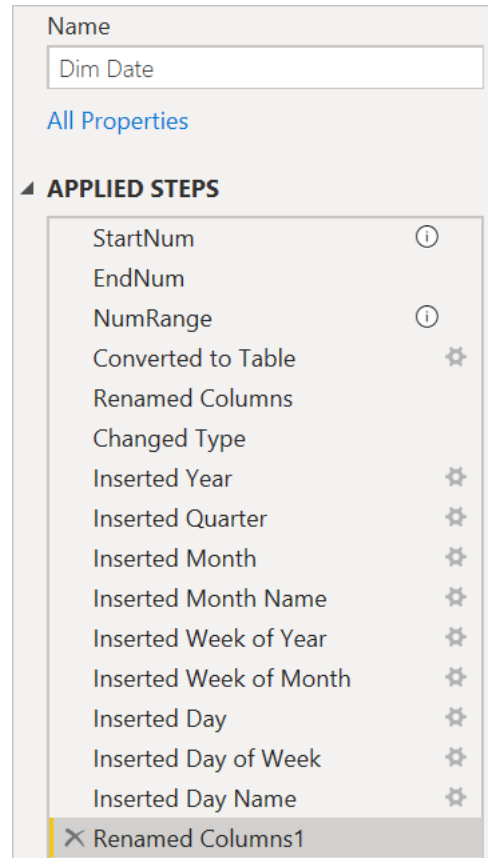
✓ No syntax errors have been detected.

OK Cancel

# Date Dimension

## Generating a Date dimension table options:

- ETL pipeline
- SQL script in the source database
- Power Query/M
- Calculated table using DAX



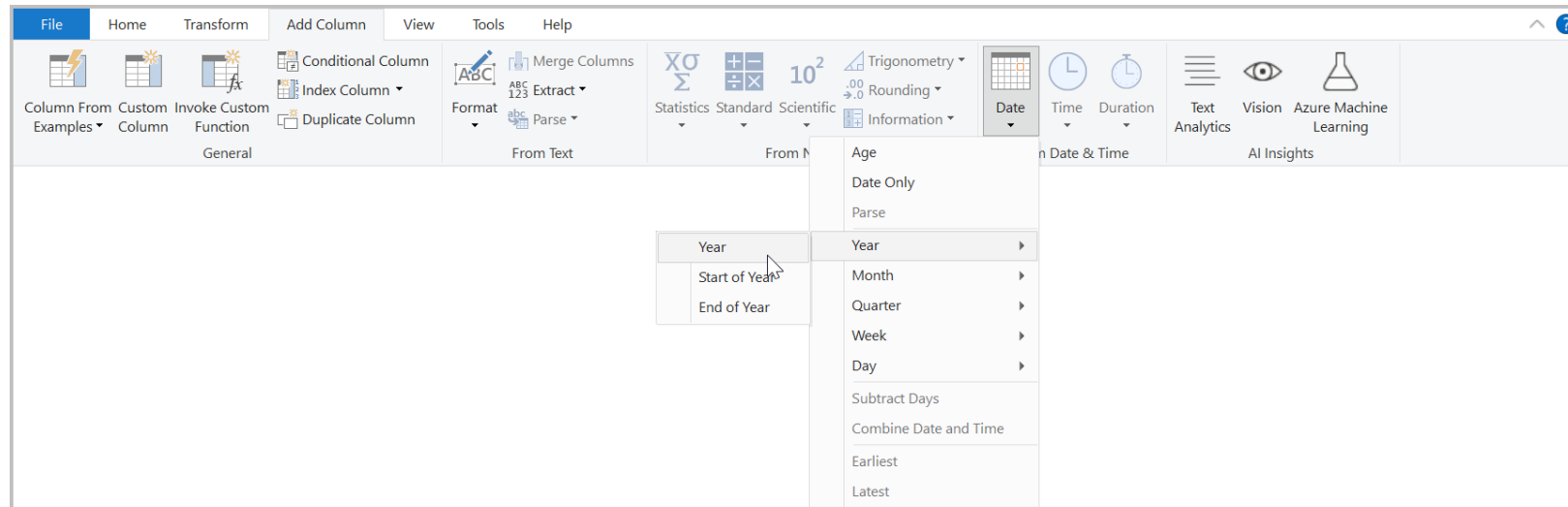
## Convert date values to numbers

```
// Convert dates to numbers
StartNum    = Number.From( DimDateStart ),
EndNum      = Number.From( DimDateEnd )
```

## Create a range of numbers using the variable values

```
// Generate number range
NumRange    = { StartNum..EndNum }
```

# Adding Date Part Columns to the Dim Date Table



- Quarter > Quarter of Year
- Month > Month
- Month > Name of Month
- Week > Week of Year
- Week > Week of Month
- Day > Day
- Day > Name of Day

# Fact Table Transformations

## GENERAL GUIDANCE

- Remove all unnecessary columns
- Ideally only contains keys and numeric measure base columns
- Utility columns for partitioning & change tracking
- Table will be hidden in the data model
- No need to rename columns with friendly names

- **Fact Online Sales**
- **Fact Store Sales**

	DiscountQuantity	DiscountAmount	TotalCost	UnitPrice	UpdateDate
1	0.00	1	2.59	12.95	null
2	0.00	1	2.59	12.95	null
3	0.00	1	2.59	12.95	null
4	0.00	1	2.59	12.95	null
5	0.00	1	2.59	12.95	null
6	0.00	1	1.00	4.98	null
7	0.00	1	1.00	4.98	null
8	0.00	1	1.00	4.98	null
9	0.00	1	1.00	4.98	null
10	0.00	1	1.00	4.98	null
11	0.00	1	1.00	4.98	null
12	0.00	1	1.00	4.98	null

# Fact Table Transformations

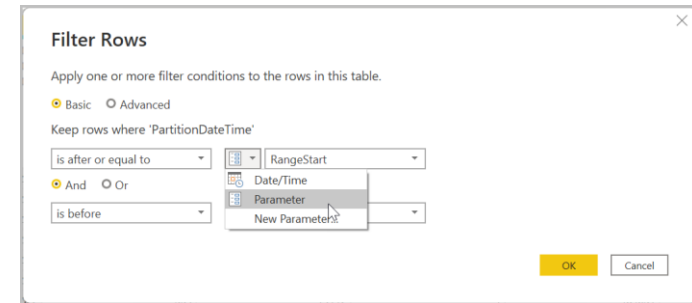
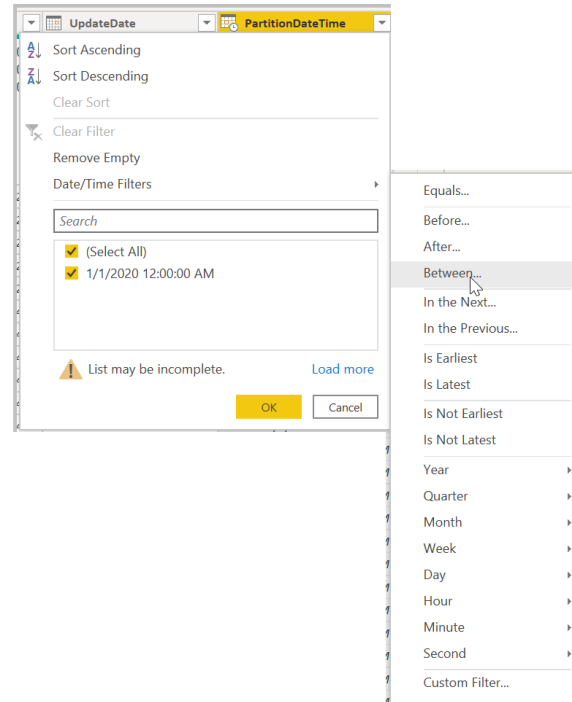
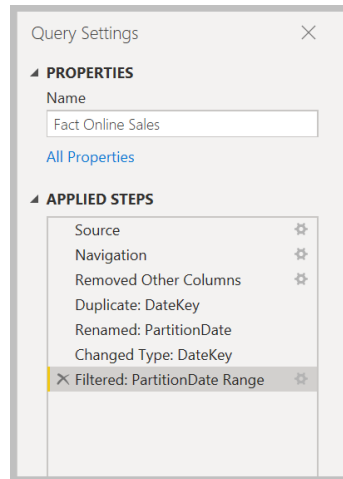
## MANAGING DATA VOLUME

- Add Partition key date/time type column
- Set date range filter to use **RangeStart** & **RangeEnd** parameters



The FactOnlineSales table contains about 21 million rows.

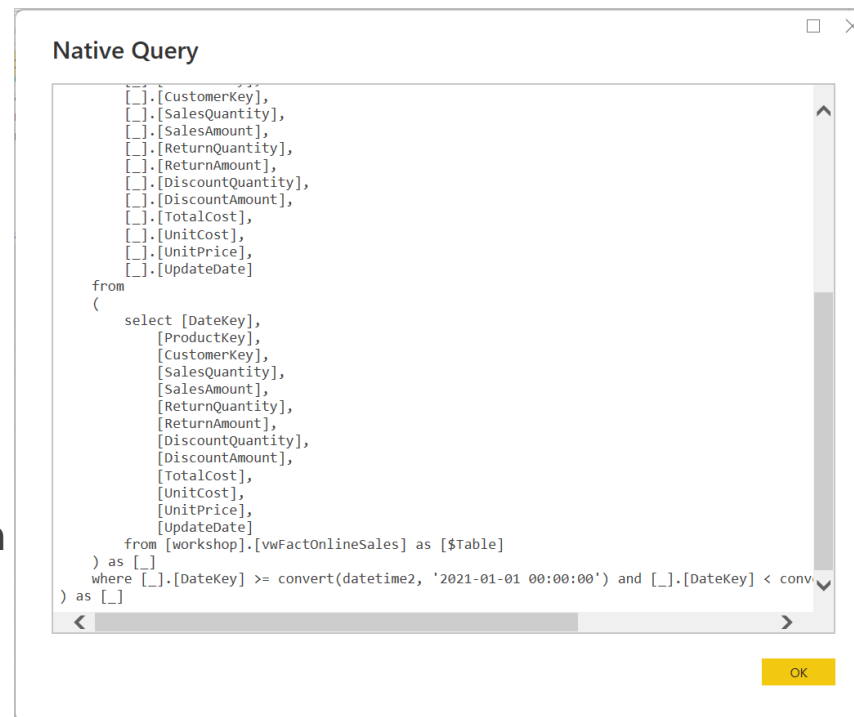
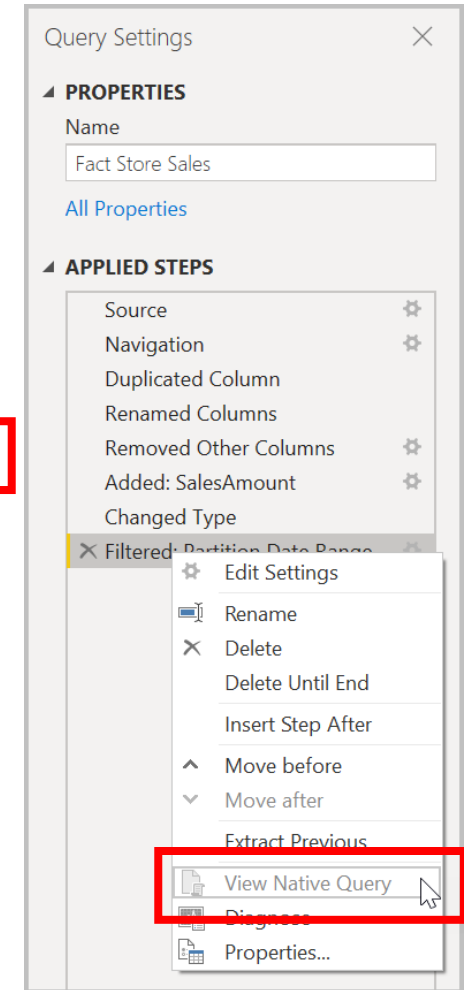
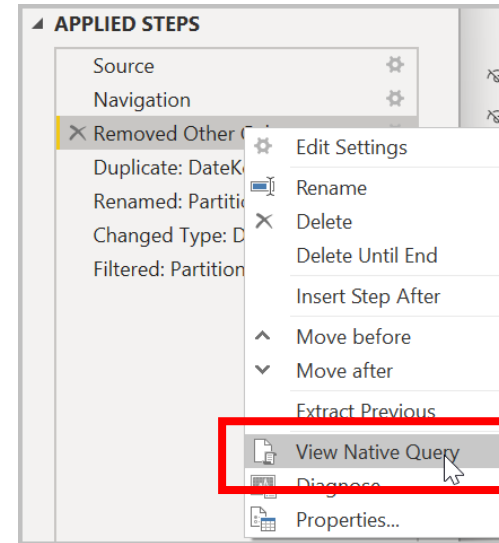
Load only data needed for development into the local copy of the model.



# You Gotta Know When to Fold 'Em

## UNDERSTANDING QUERY FOLDING

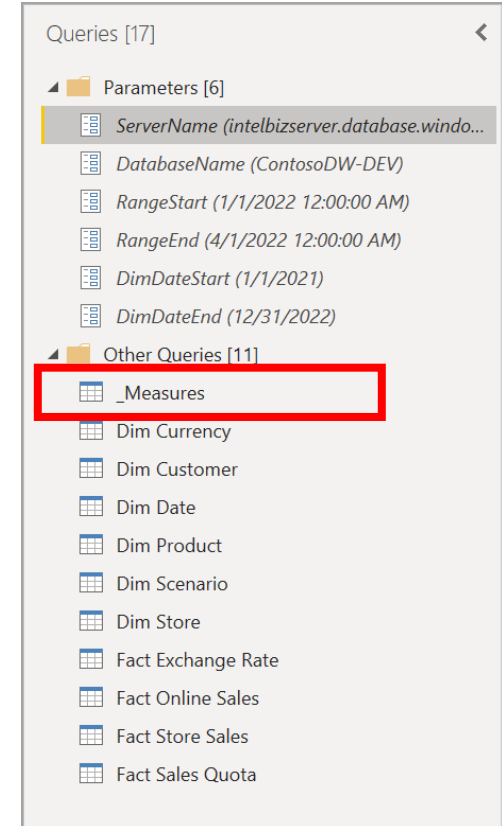
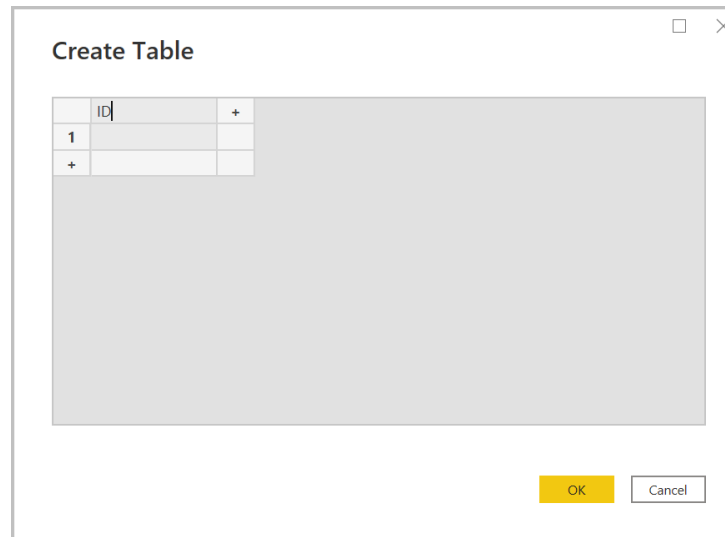
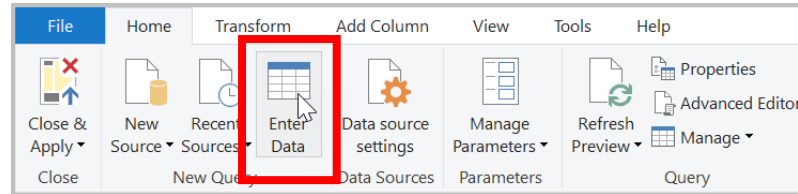
- Query folding produces a query, in the native language of the data source.
- Many query steps can be folded
- Some steps cannot
- Perform the most critical steps first:
  - Filtering
  - Grouping
  - Remove columns
  - Change data type
- 1. Try to get entire query to fold
- 2. Perform less impactful steps later, if folding is broken



# Create a Measure Container Table

A PLACE TO PUT MY MEASURES

- Create a single measure group/container table for all measures
- Fact tables in the data model will be hidden
- Some measures are based on values from multiple fact tables

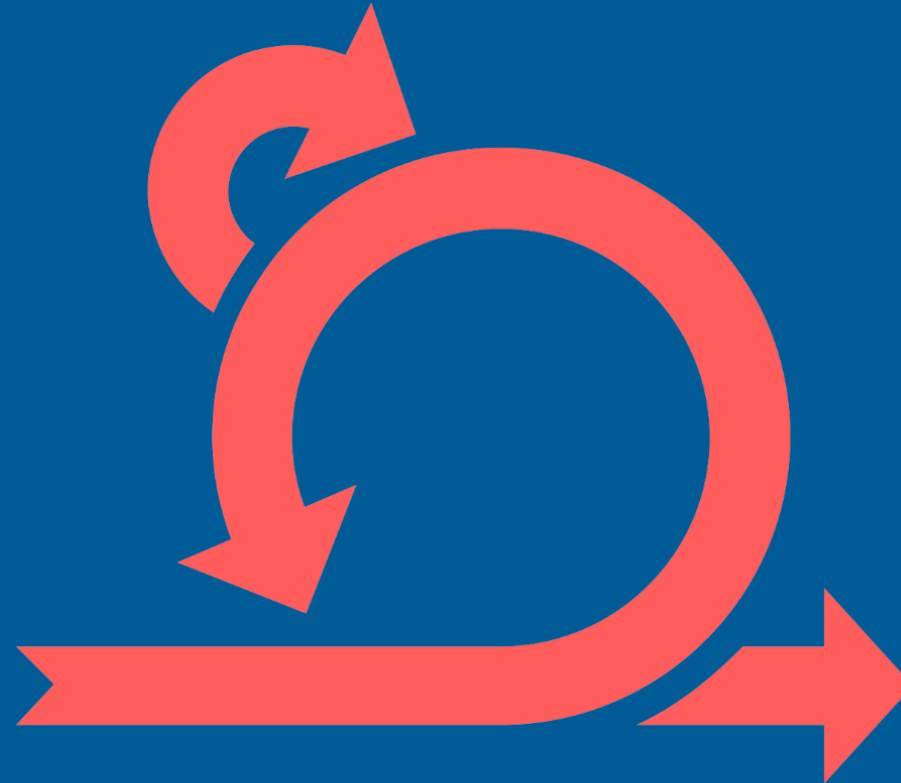




# Iteration 2

Introduce new fact tables  
& related dimensions:

- Scenario Plan
- Exchange Rate



# Iteration 2 Requirements

## CHANGE REQUEST

Our business stakeholder has requested that an addition be made to the current project due to business conditions that have changed since the beginning of the project. Our **Project Manager** has met with the **CFO** and the **CIO**, suggesting that the current work be completed and delivered before making additions. However, the additional request is deemed to be a critical need, and the business have asked that the project scope be expanded to include these additional tables and report functionality. **A change request has been written and approved** to include these new requirements and the **project budget and schedule have been adjusted** to reflect this new request.

Financial Planners need:

- Sales Quota table with Actual Sales & Budget

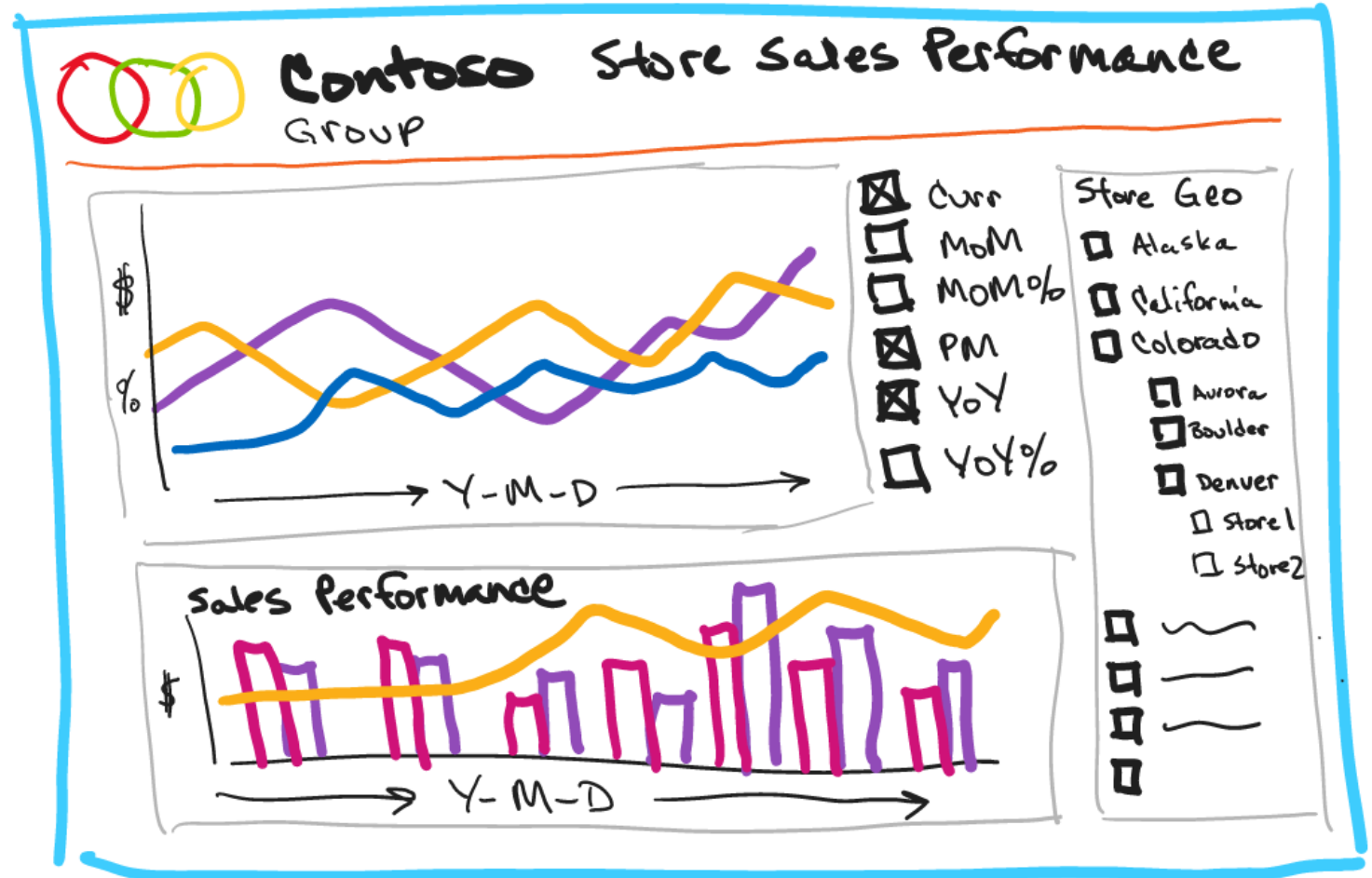
Sales Organization needs:

- Exchange Rates
- Currency Rates change during the day and reports with currency converted measures must include **real-time** updates as the source data changes

# Iteration 2 Requirements

## REPORT CAPABILITIES & WIREFRAME

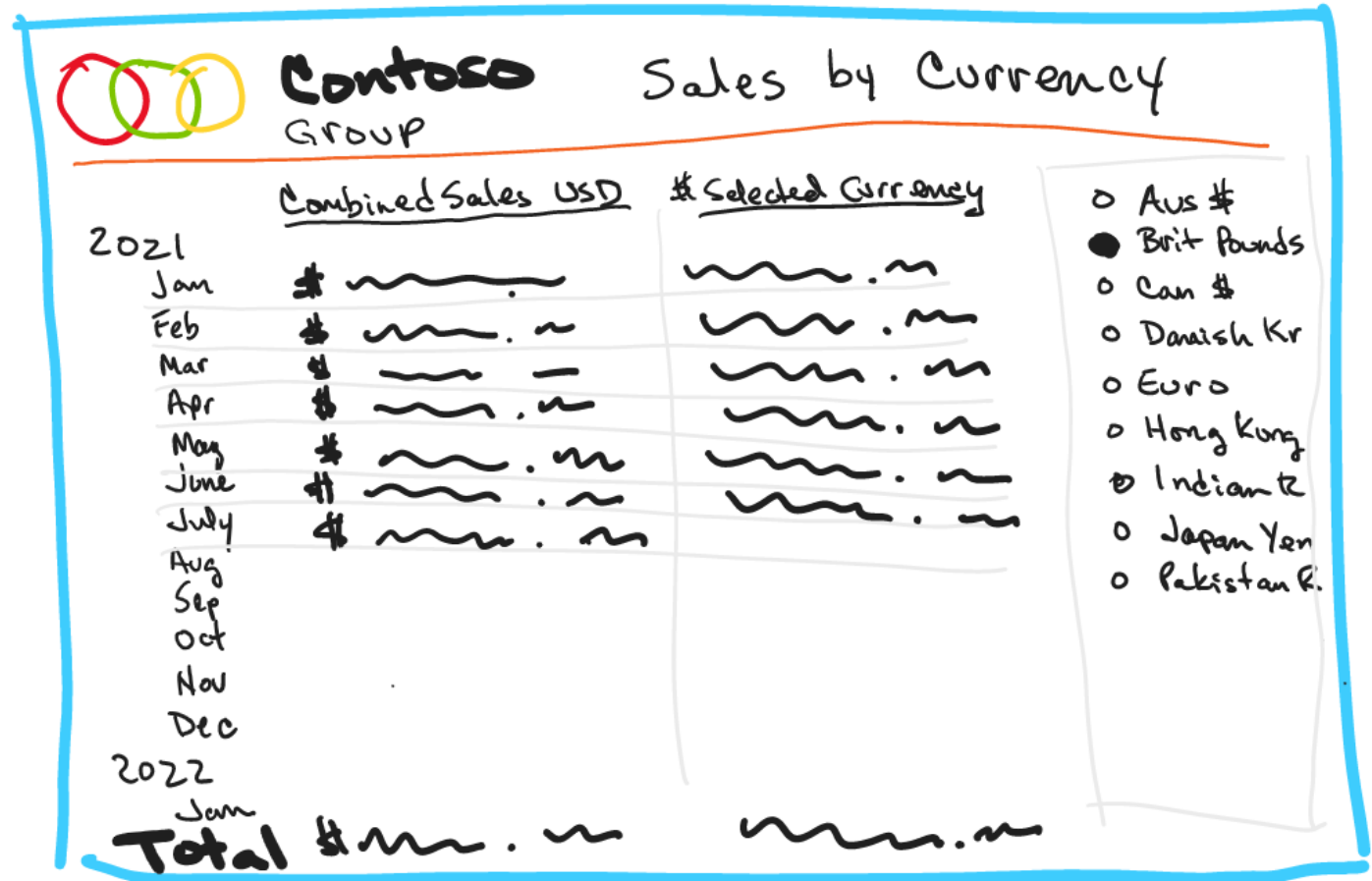
- The bottom visual should be enhanced to include **Actual Sales Amount** and **Budget Sales Amount** from the Sales Quota system
- Calculated difference between Actual Sales and Budget



# Iteration 2 Requirements

## REPORT CAPABILITIES & WIREFRAME

- A second report page will show **Combined Sales Amount** values in US dollars alongside the sales value converted to any selected foreign currency.
- Currency conversion is based on the **most current** exchange rate, using real-time source data.

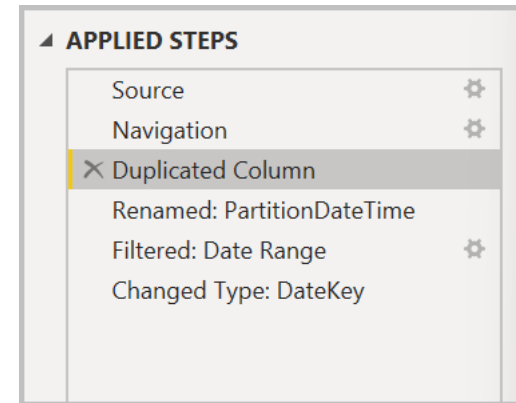


# Iteration 2 Requirements

	Date	Customer	Product	Store	Scenario	Currency
<b>Online Sales</b>	X	X	X			
Online Sales Amt						
Online Sales Qty						
<b>Store Sales</b>	X		X	X		
Store Sales Qty						
Store Sales Amt						
<b>Sales Quota</b>	X		X		X	
Quota Qty						
Quota Amt						
<b>Exchange Rate</b>	X					
End of Day Rate						X

# New Fact Table Queries

- Fact Sales Quota
  - Simple query, import from **vwFactSalesQuota**
  - Apply the same pattern as previous fact table queries except no UpdateDate column
  
- Fact Exchange Rate
  - Based on **vwFactExchangeRate** view
  - Connect using **DirectQuery**



# Hands On Exercise

## Part 2:

### Data Modeling for Scale

Part 1



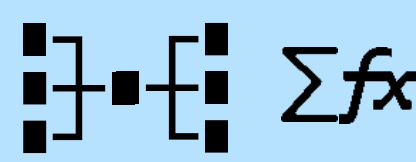
Part 2



Part 3



Part 4



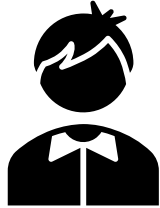
- Prepare source data
- Connect to data sources
- Create parameters
- Filter records

- Transform data
- Import tables
- Create data model
- Create base measures

- Create advanced measures
- Visualize
- Deploy to cloud service
- Deliver to user audience

# Model Design

## Database Developer



- Flattened results
- Transform in SQL
- Wide tables
- Pre-aggregated re
- Import summary

## BI Solution Designer



### Balance

- Build dimensional star schema
- Avoid wide tables
- Remove long text fields
- Remove unused fields
- Tall tables are OK if they are optimized for storage & analytics

er Query



# Model Design



# Date Dimension

- Mark table as Date table, select Date type key column
- Create date part hierarchies to support report drill-down navigation

### Mark as date table

Select a column to be used for the date. The column must be of the data type 'date' and must contain only unique values. [Learn more](#)

Date column

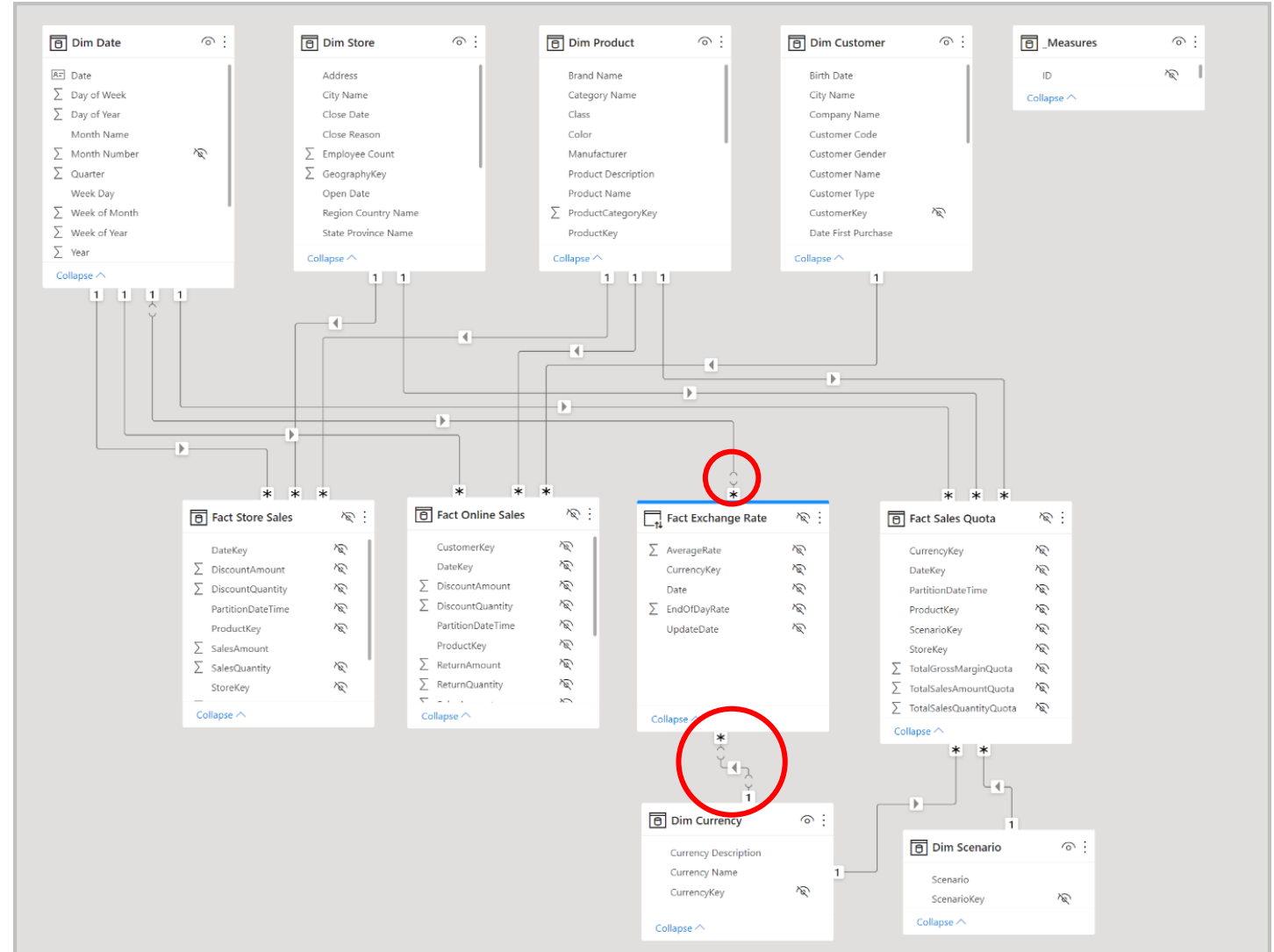
Date

When you mark this as a date table, the built-in date tables that were associated with this table are removed. Visuals or DAX expressions referring to them may break. [Learn how to fix visuals and DAX expressions](#)

OK Cancel

# Build the Data Model

- Create relationships
- Arrange tables
- Create multiple layouts for large models
- Proper query and transformation design simplifies data model design
- Adding DirectQuery tables to an import model creates a “composite model”
- DirectQuery tables are displayed with different colored headings
- A composite model contains “soft” relationships



# Build the Data Model

- True dimensional design makes relationship mapping simple
- Avoid bi-directional relationships, but use them when necessary (typically for many-to-many)
- Manage user expectations with dimension>dimension cross filtering
- High cardinality dimensions affect performance at scale
- Consistent key naming simplifies model and helps avoid mistakes
- Some exceptions are OK (for example: DateKey > Date)
- Be careful with relationship Autodetect (which is on by default)
- Relationship MUST be on the same data type. watch for:
  - Date <> DateTime
  - WholeNumber <> Decimal

Manage relationships

Active	From: Table (Column)	To: Table (Column)
<input checked="" type="checkbox"/>	Fact Exchange Rate (CurrencyKey)	Dim Currency (CurrencyKey)
<input checked="" type="checkbox"/>	Fact Exchange Rate (Date)	Dim Date (Date)
<input checked="" type="checkbox"/>	Fact Online Sales (CustomerKey)	Dim Customer (CustomerKey)
<input checked="" type="checkbox"/>	Fact Online Sales (DateKey)	Dim Date (Date)
<input checked="" type="checkbox"/>	Fact Online Sales (ProductKey)	Dim Product (ProductKey)
<input checked="" type="checkbox"/>	Fact Sales Quota (CurrencyKey)	Dim Currency (CurrencyKey)
<input checked="" type="checkbox"/>	Fact Sales Quota (DateKey)	Dim Date (Date)
<input checked="" type="checkbox"/>	Fact Sales Quota (ProductKey)	Dim Product (ProductKey)
<input checked="" type="checkbox"/>	Fact Sales Quota (ScenarioKey)	Dim Scenario (ScenarioKey)
<input checked="" type="checkbox"/>	Fact Sales Quota (StoreKey)	Dim Store (StoreKey)
<input checked="" type="checkbox"/>	Fact Store Sales (DateKey)	Dim Date (Date)
<input checked="" type="checkbox"/>	Fact Store Sales (ProductKey)	Dim Product (ProductKey)
<input checked="" type="checkbox"/>	Fact Store Sales (StoreKey)	Dim Store (StoreKey)

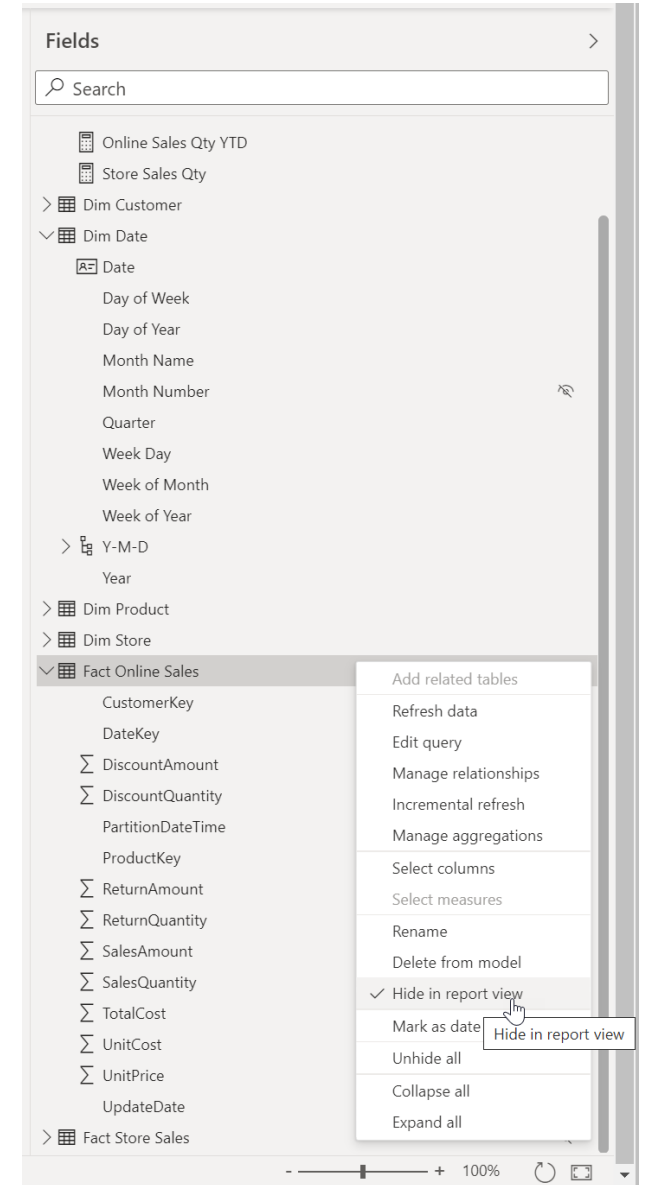
New... Autodetect... Edit... Delete

Close

# Quick... Hide the Facts

## HIDE ALL FACT TABLES

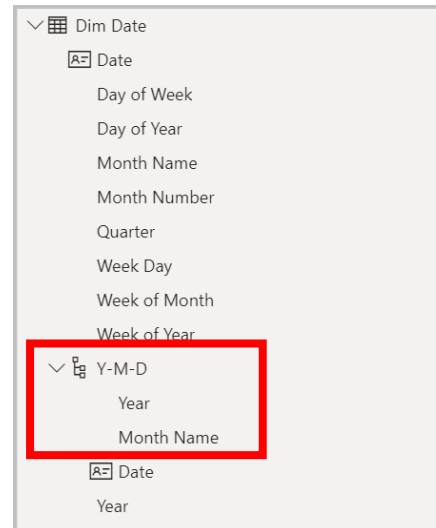
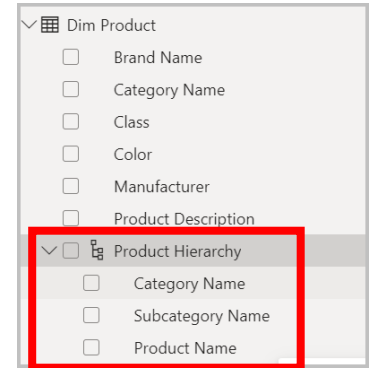
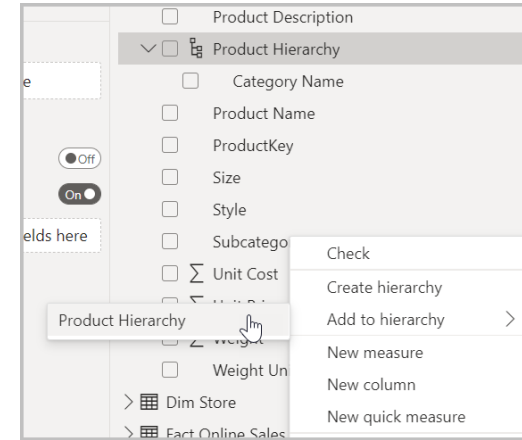
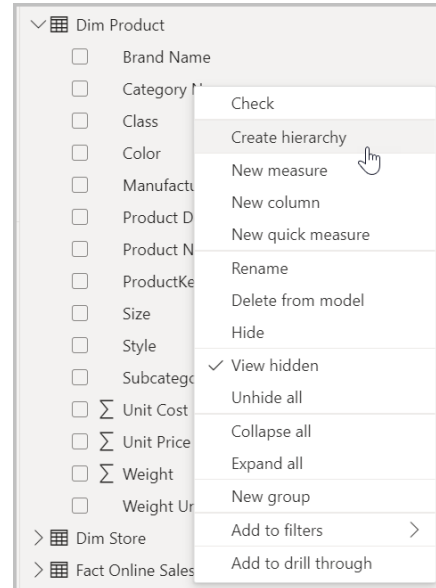
- Implicit & Explicit Measures
  - Ideally, expose only measures and hide all summable numeric columns
  - Some client tools don't support implicit measures (like Excel)
  - Explicit measures, although a little more work, provide more control and flexibility
- Hide key & utility fields
- Hiding a table effectively hides the individual columns



# Hierarchies

## SUPPORT DRILL-DOWN NAVIGATION

- Product hierarchy
- Date hierarchies
- Create multiple hierarchies for different reporting needs
- Parent-child hierarchies (chart of accounts or employee org chart) are more complicated.

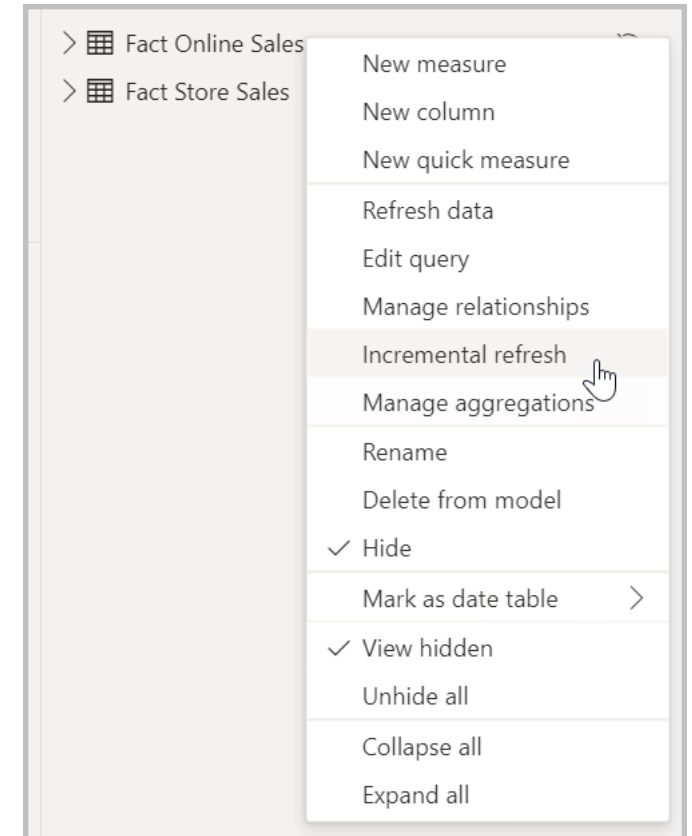


# Table Partitioning & Incremental Refresh

# Date Range Filtering & Incremental Refresh

SOLVES MULTIPLE PROBLEMS

- In a data model without Incremental Refresh, data volume can be managed in the service using RangeStart & RangeEnd parameters
- In a data model with Incremental Refresh, date range partitioning is automatically managed in the service & the parameters are removed from view
- Using the RangeStart & RangeEnd parameter filtering approach provide a consistent best practice design pattern





# Configure Incremental Refresh

FOR EACH FACT TABLE

- Archive periods:  
Generates one static partition per specified period
- Incremental refresh periods:  
Generates one partition per specified period
- Detect data changes:  
Reprocesses any partition with a date newer than the last refresh date.
- Get the latest data in real time with DirectQuery:  
Create a hybrid model with a DirectQuery partition newly inserted records

**Incremental refresh and real-time data**

1. Select table  
Fact Online Sales

2. Set import and refresh ranges  
 Incrementally refresh this table  
Archive data starting 3 Years before refresh date  
Data imported from 1/1/2019 to 1/31/2022 (inclusive)  
Incrementally refresh data starting 6 Months before refresh date  
Data will be incrementally refreshed from 2/1/2022 to 7/31/2022 (inclusive)

3. Choose optional settings  
 Get the latest data in real time with DirectQuery (Premium only) [Learn more](#)  
 Only refresh complete months [Learn more](#)  
 Detect data changes [Learn more](#)  
Only refresh data in the last 6 months if the maximum value of this datetime column changes:  
UpdateDate

4. Review and apply  
Archived Incremental Refresh  
3 years before refresh date 6 months before refresh Refresh date  
Apply Cancel

# Incremental Refresh & Partition Demonstration

FOR EACH FACT TABLE

- First data refresh process takes time to create all partitions



# Hands On Exercise

## Part 3: Measure Development in the Enterprise

Part 1



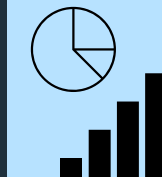
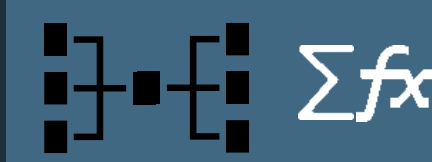
Part 2



Part 3



Part 4



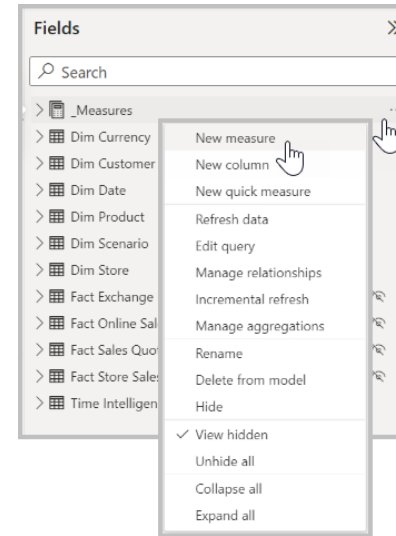
- Prepare source data
- Connect to data sources
- Create parameters
- Filter records

- Transform data
- Import tables
- Create data model
- Create base measures

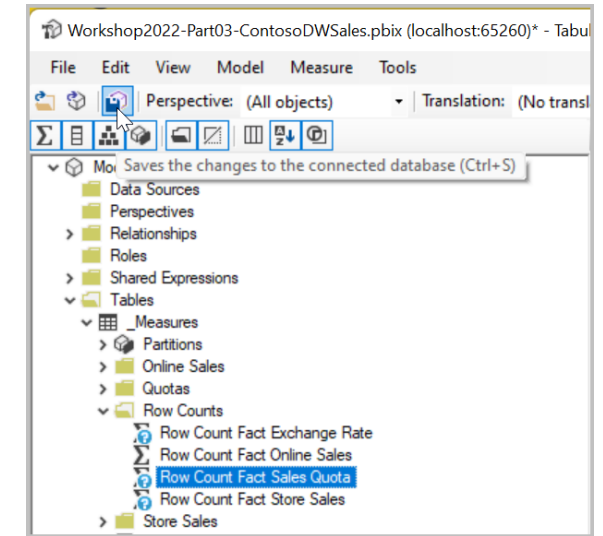
- Create advanced measures
- Visualize
- Deploy to cloud service
- Deliver to user audience

# Row Count & Validation Measures

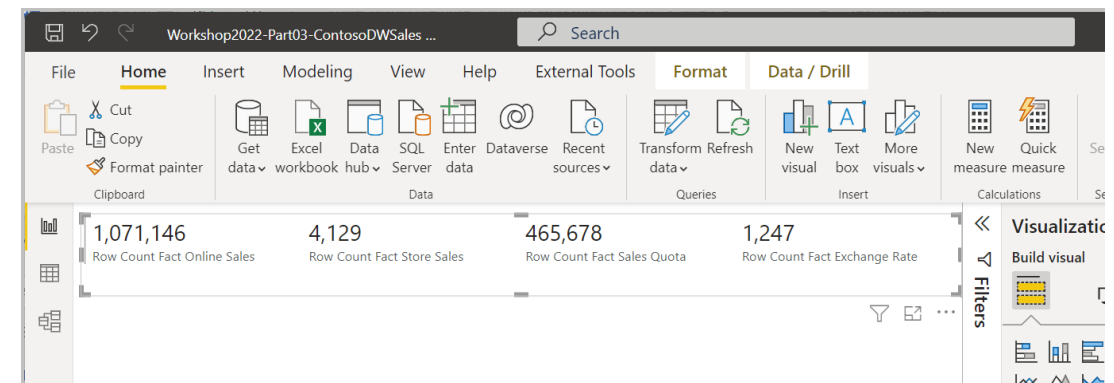
- Table row counts are a convenient initial data validation test
- Ensures data is loaded as expected
- Use Tabular Editor to duplicate existing measures & add to display folders



Tabular Editor:



Row Count Fact Online Sales =  
`COUNTROWS ( 'Fact Online Sales' )`

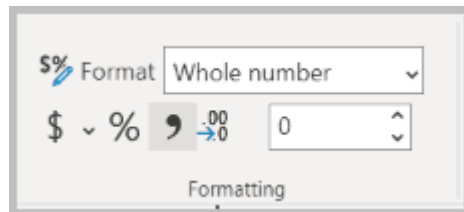
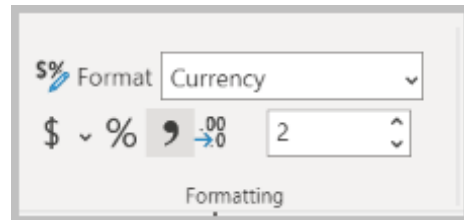


# Enhancing the Model with Measures



# Base Measures

- Replace “implicit measures” / numeric fact table columns
- Basis for additional measures



Measure Name	Expression
Online Sales Amt	= SUM( 'Fact Online Sales'[SalesAmount] )
Online Sales Qty	= SUM( 'Fact Online Sales'[SalesQuantity] )
Store Sales Amt	= SUM( 'Fact Store Sales'[SalesAmount] )
Store Sales Qty	= SUM( 'Fact Store Sales'[SalesQuantity] )
Combined Sales Amt	= [Online Sales Qty] + [Store Sales Qty]
Quota Sales Amt	= SUM( 'Fact Sales Quota'[TotalSalesAmountQuota] )
Quota Sales Qty	= SUM( 'Fact Sales Quota'[TotalSalesQuantityQuota] )

# Display Folders

## TO ORGANIZE MEASURES

- Display folder is a measure property (rather than an actual “folder”)
- Set Display Folder property for a measure to “create” a folder
- Use Tabular Editor to drag measures into existing folders

### Tabular Editor:

The screenshot shows the Tabular Editor interface. On the left, a tree view displays the data model structure, including tables and measures. The 'Combined Sales' folder is expanded, showing measures like 'Combined Sales Amt', 'Combined Sales Qty', and 'Combined Unit Cost'. The main area shows the 'Basic' properties for a selected measure. The 'Display Folder' property is highlighted with a red box and set to 'Combined Sales'. Below the properties, a description for the 'Display Folder' property is provided: 'Defines the display folder for the Measure, for use by clients.'

### Power BI Desktop – Model view:

The screenshot shows the Power BI Desktop Model view. The 'Properties' pane on the left is open to the 'General' tab, and the 'Display folder' property is highlighted with a red box and set to 'Combined Sales'. The 'Fields' pane on the right shows the data model structure, with the 'Combined Sales' folder expanded and its measures listed.

# Calculation Groups

Reduce measure count & development effort

Implement time intelligence

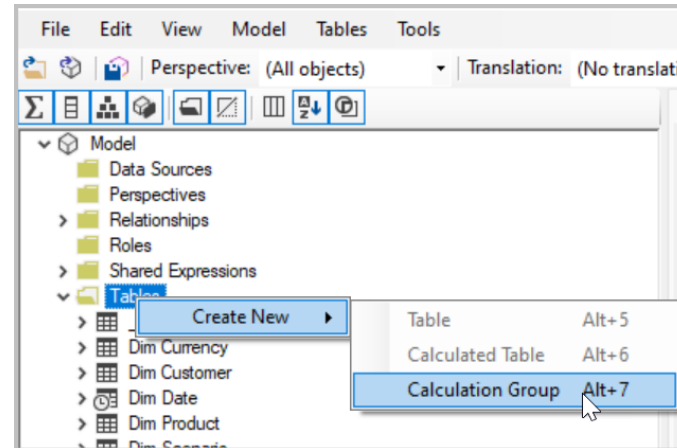
Layer complex calculation logic over any selected measure



# Creating multiple measures: There has got to be a better way!

USING TABULAR EDITOR TO CREATE A CALCULATION GROUP

- Enables variation logic to be added to a selected measure
- Can be used to implement calculation enhancements like time intelligence
- Many advanced & creative possibilities
- Specialized DAX functions are used to pass the selected measure into a calculation group expression



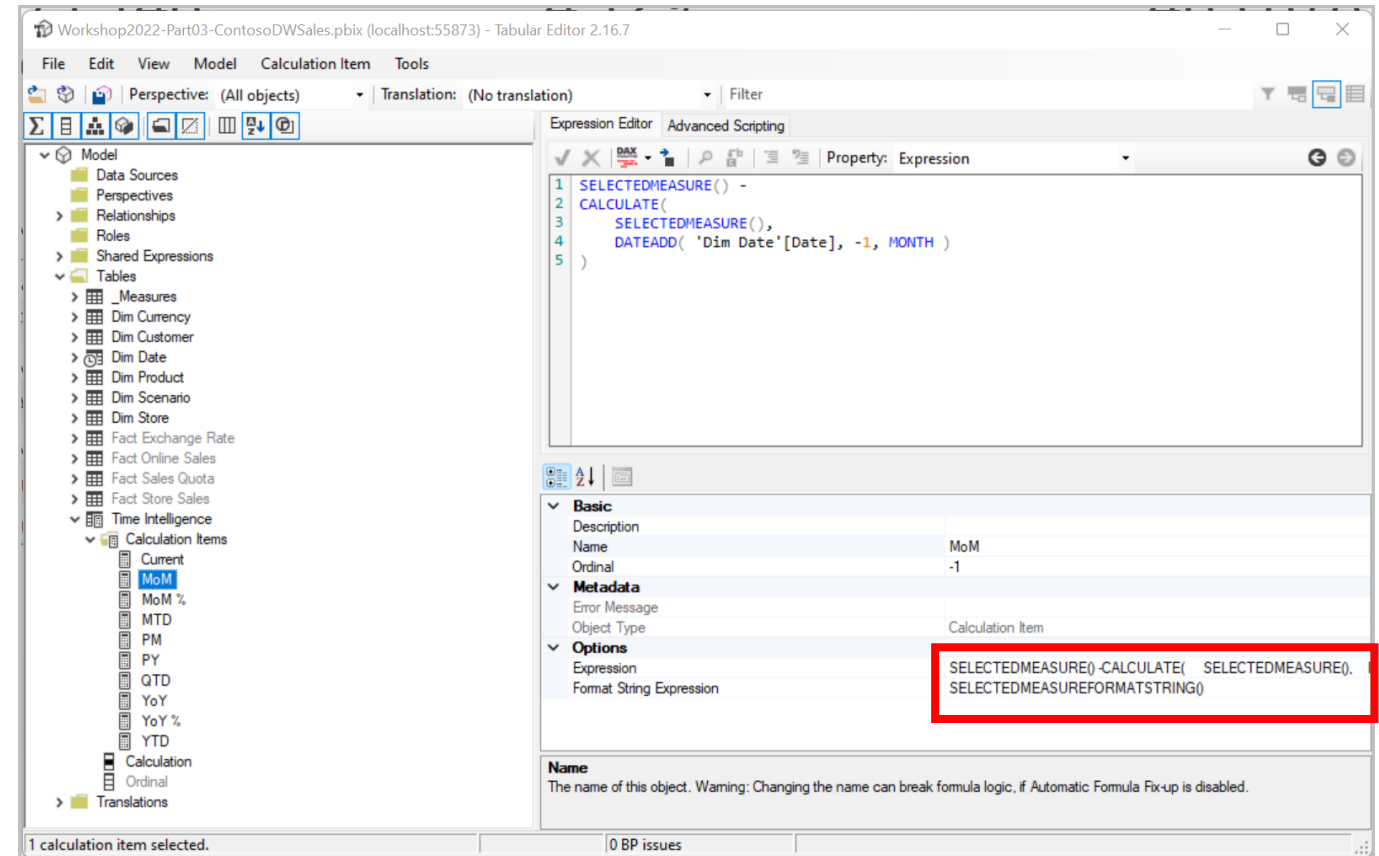
DAX Function	Purpose
<b>SELECTEDMEASURE()</b>	Passes the currently selected measure into a calculation group expression.
<b>SELECTEDMEASUREFORMATSTRING()</b>	Returns the format string property of the selected measure.
<b>SELECTEDMEASURENAME()</b>	Returns the name of the selected measure.

# Calculation Group to Implement Time Intelligence

USE TABULAR EDITOR TO CREATE A CALCULATION GROUP

Time Intelligence calculation items:

- Current
- MoM
- MoM %
- MTD
- PM
- PY
- QTD
- YoY
- YoY %
- YTD



Completed expressions for all items are in: **Time Intelligence Calculation Group Items.txt**

# Hands On Exercise

## Part 4: Visualize & Analyze

Part 1



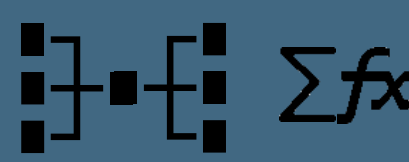
Part 2



Part 3



Part 4



- Prepare source data
- Connect to data sources
- Create parameters
- Filter records

- Transform data
- Import tables
- Create data model
- Create base measures

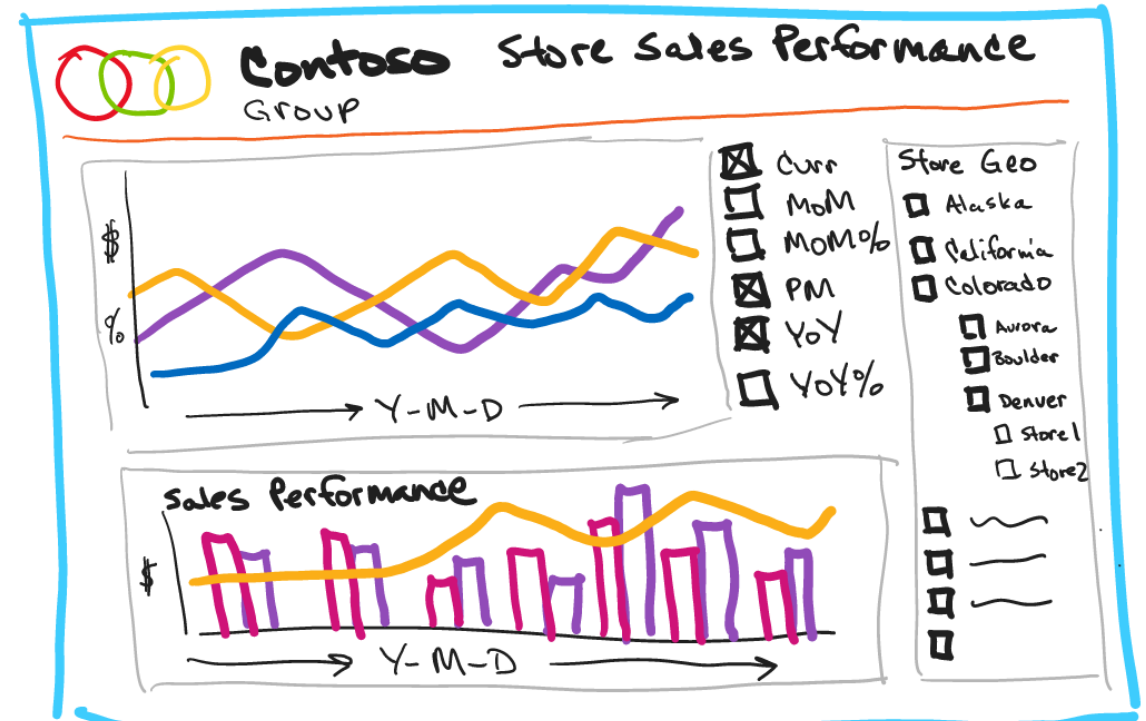
- Create advanced measures
- Visualize
- Deploy to cloud service
- Deliver to user audience

# Report Requirements Review

## REPORT CAPABILITIES & WIREFRAME

Business questions:

- What is the **Sales Amount** grouped by **Year** or **Month** or **Day**, for a specific period of time.
- Trend chart should allow a user to select a **time series metric** (like Month Over Month, Month Over Month % Change, Month To Date, Prior Month, Year To Date) & then show them along the axis of Years, Months or Days.
- All of the data on the report can be filtered by store regions, such as **State**, **City** or an individual **Store**.
- The bottom visual should be enhanced to include **Actual Sales Amount** and **Budget Sales Amount** from the Sales Quota system
- Calculated difference between **Actual Sales** and **Budget**



# Report Requirements Review

## ITERATION 2 REPORT CAPABILITIES & WIREFRAME

- A second report page will show **Combined Sales Amount** values in US dollars alongside the sales value converted to any selected foreign currency.
- Currency conversion is based on the **most current** exchange rate, using real-time source data.

**Contoso Group** Sales by Currency

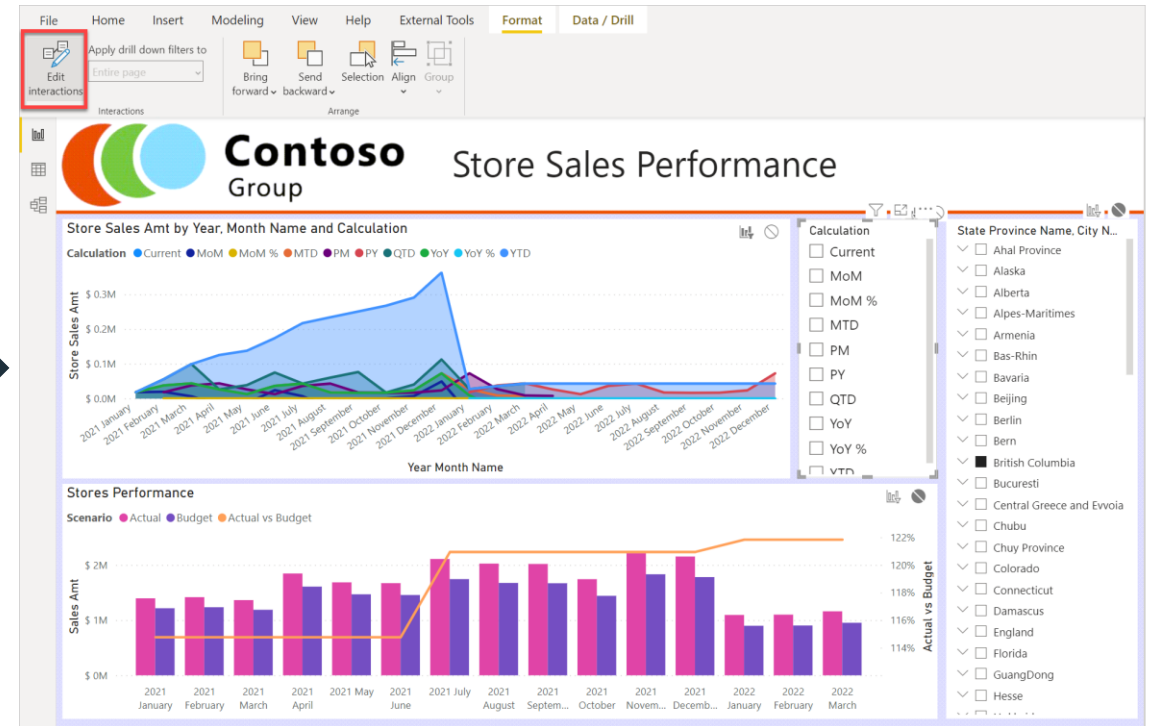
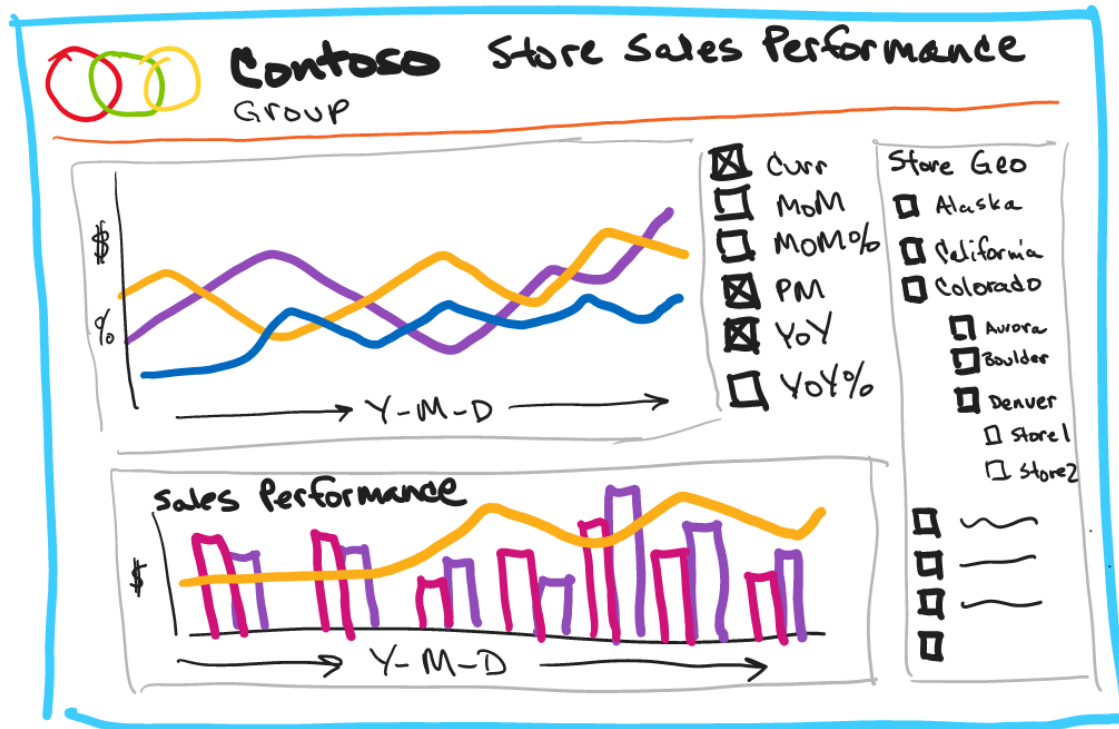
	Combined Sales USD	\$ Selected Currency
2021		
Jan	\$ ~~~~~	~~~~~.
Feb	\$ ~~~~~.	~~~~~.
Mar	\$ ~~~~~	~~~~~.
Apr	\$ ~~~~~.	~~~~~.
May	\$ ~~~~~.	~~~~~.
June	\$ ~~~~~.	~~~~~.
July	\$ ~~~~~.	~~~~~.
Aug		
Sep		
Oct		
Nov		
Dec		
2022		
Jan		
<b>Total</b>	\$ ~~~~~.	~~~~~.

- Aus \$
- Brit Pounds
- Can \$
- Danish Kr
- Euro
- Hong Kong
- Indian R
- Japan Yen
- Pakistan R.

# Visual Report Design

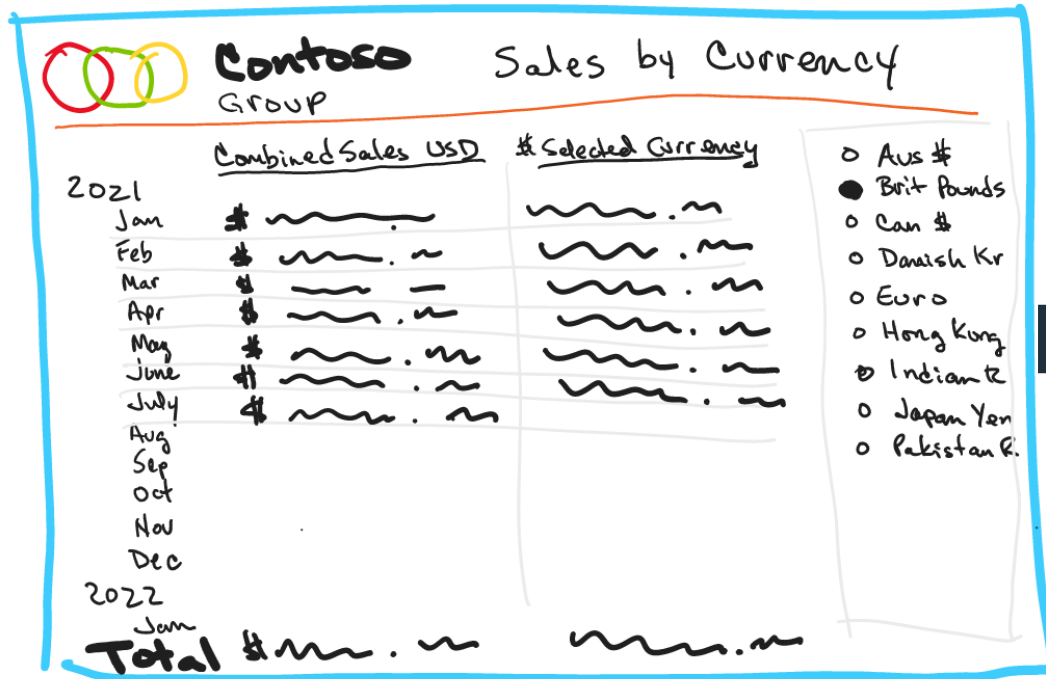


# Functional Design to Physical Design



# Functional Design to Physical Design

## ITERATION 2 REQUIREMENTS



This page is provided in the Part 5 solution



# Hands On Exercise

## Part 5: Deploy, Test & Deliver

Part 1



Part 2



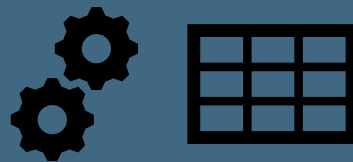
Part 3



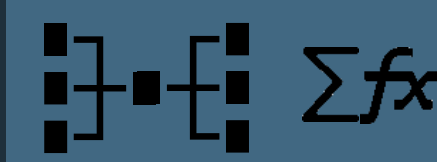
Part 4



- Prepare source data
- Connect to data sources
- Create parameters
- Filter records



- Transform data
- Import tables
- Create data model
- Create base measures



- Create advanced measures
- Visualize
- Deploy to cloud service
- Deliver to user audience

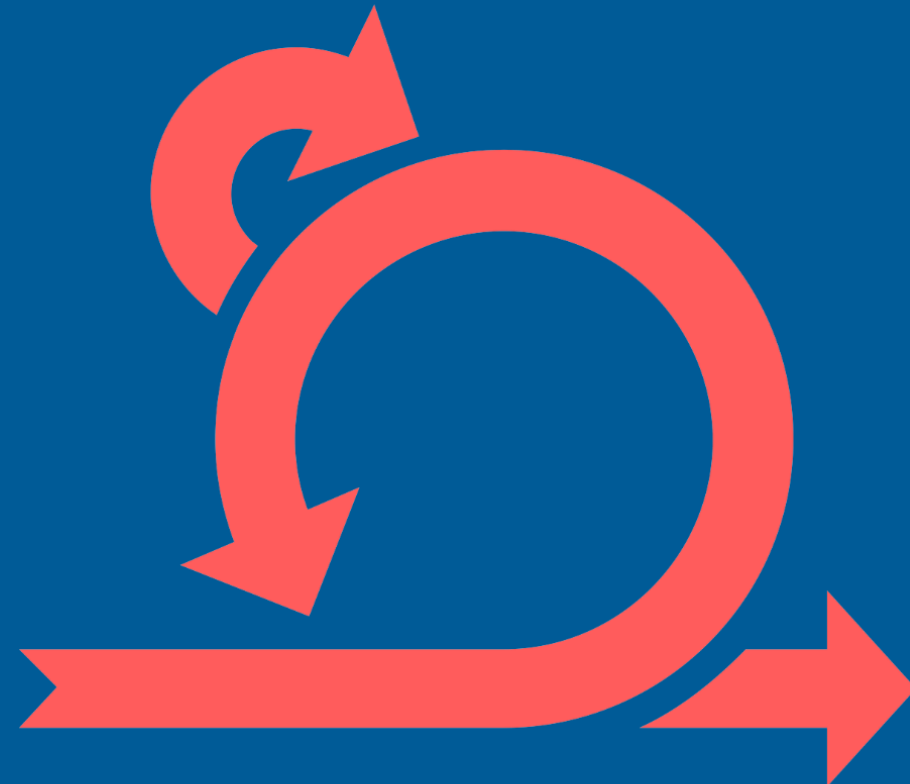


# Managed Deployment & Delivery

# Continuous Integration & Continuous Delivery (CI/CD) for Power BI

A quickly-evolving discipline:

- Manual techniques
- Report & dataset separation
- Deployment pipeline
- Decompose PBIX file objects with Tabular Editor
- Next-generation:
- Desktop capability
- Azure DevOps integration



# Using a Deployment Pipeline to Manage Deployment Stages

The screenshot shows the Power BI Deployment Pipelines interface. On the left, a sidebar lists workspaces: 'Contoso BI Analytics DEV', 'Contoso BI Analytics TEST', and 'Contoso BI Analytics'. On the right, a deployment pipeline named 'Enterprise Workshop Pipeline' is shown with a rocket icon. Three callout boxes are overlaid on the image:

- Deploy here**: Points to the 'Contoso BI Analytics DEV' workspace.
- Promote to Test**: Points to the 'Contoso BI Analytics TEST' workspace.
- Promote to Release & Convert workspace to App**: Points to the 'Contoso BI Analytics' workspace.

A fourth callout box on the right states: **Deployment pipeline contains 3 workspaces: DEV, TEST & Prod**, with a red arrow pointing to the 'Enterprise Workshop Pipeline' card.

# Using a Deployment Pipeline to Manage Deployment Stages

The screenshot displays the Microsoft Power BI Deployment Pipelines interface. At the top, the browser address bar shows the URL <https://app.powerbi.com/pipelines/66d8b75e-ec09-41f7-a5eb-3018c8d826d9>. The page title is "Power BI Deployment pipelines".

The main content area shows a pipeline named "Enterprise Workshop Pipeline" with three stages: "Development", "Test", and "Production". Each stage is represented by a card with a "Learn more" link and a "Deploy" button. The "Development" stage is labeled "Enterprise Workshop DEV" and shows 0 Dataflows, 0 Datamarts, and 1 Dataset. The "Test" stage is labeled "Enterprise Work..." and shows 0 Dataflows, 0 Datamarts, and 1 Dataset, with a deployment date of 31/07/2022. The "Production" stage is labeled "Enterprise Work..." and shows 0 Dataflows, 0 Datamarts, and 1 Dataset, with a deployment date of 29/07/2022. A "Compare" button is visible between the Test and Production stages.

At the bottom of the interface, a large blue banner with white text reads "DEV TEST PROD", indicating the deployment stages.

# Dataset Settings

APPLIES TO DEPLOYED DATASET

**Development** [Learn more](#)

Design, review, and revise your content in a development workspace. When it's ready to test and preview, deploy the content to the test stage.

**Enterprise Workshop DEV**

0 Dataflows 0 Datamarts (Preview)

Settings for ContosoDW Sales

[View dataset](#)

This dataset has been configured by [Paul@intelligentbiz.net](#).

Last refresh succeeded: Sun Jul 31 2022 08:07:54 GMT-0700 (Pacific) [Refresh history](#)

Dataset description

Describe the contents of this dataset.

Apply Discard

Gateway connection

Data source credentials

Parameters

DatabaseName  
ContosoDW-DEV

ServerName  
intelbizserver.database.windows.net

Apply Discard

**Test** [Learn more](#)

Test and verify your content in a preproduction workspace. When it's ready to distribute, deploy the content to the production stage.

**Enterprise Workshop TEST**

0 Dataflows 0 Datamarts (Preview)

Settings for ContosoDW Sales

[View dataset](#)

This dataset has been configured by [Paul@intelligentbiz.net](#).

[Refresh history](#)

Dataset description

Describe the contents of this dataset.

Apply Discard

Gateway connection

Data source credentials

Parameters

DatabaseName  
ContosoDW

ServerName  
intelbizserver.database.windows.net

Apply Discard

**Production** [Learn more](#)

Your content has been tested and is ready to distribute to your consumers as an app or by access to the production workspace.

**Enterprise Workshop** Deployed: 7/29/2022, 8:59:03 PM

0 Dataflows 0 Datamarts (Preview) 1 Datasets 1 Reports

ContosoDW Sales

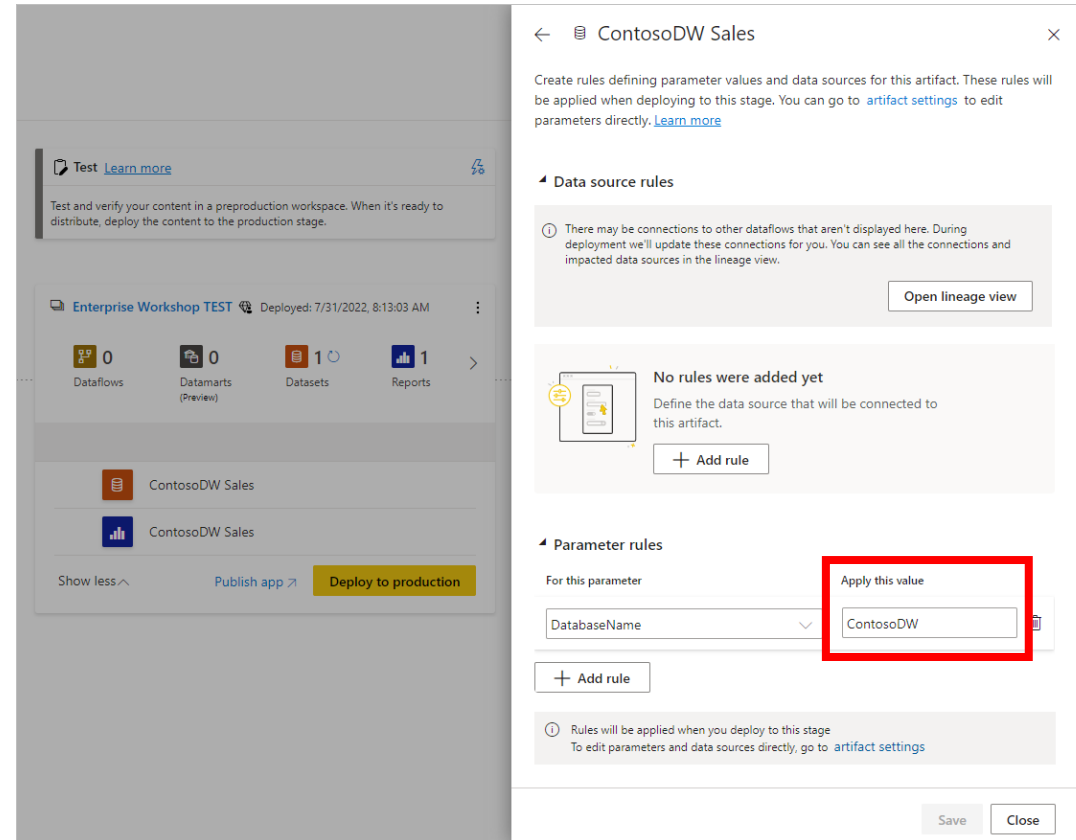
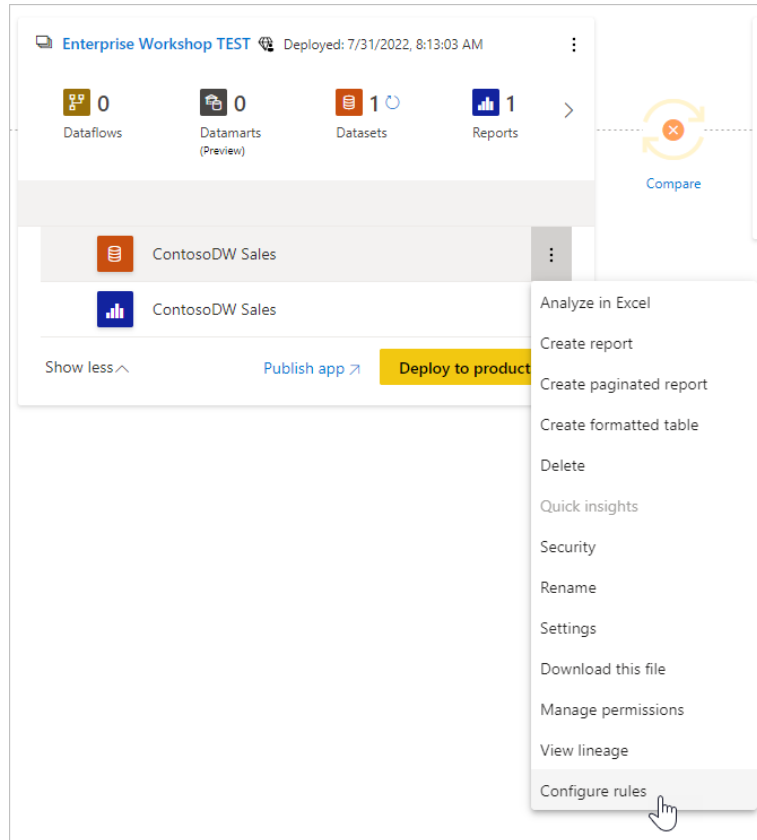
ContosoDW Sales

Show less ^

[Publish app](#) [Update app](#)

# Deployment Rules

APPLIES TO EACH VERSION OF A DATASET IN THE WORKSPACE



# Pipeline Comparison

COMPARES VERSIONS OF EACH OBJECT IN PIPELINE

The screenshot displays three stages of a pipeline: Development, Test, and Production. Each stage shows a summary of objects (Dataflows, Datamarts, Datasets, Reports) and their status. In the Development stage, a green circular icon with a checkmark is circled in red. In the Test stage, a yellow circular icon with a red 'X' and the word 'Compare' is circled in red. In the Production stage, a yellow circular icon with a red 'X' and the number '1' is circled in red. Below the Test and Production stages, yellow labels indicate 'Different' for the ContosoDW Sales objects. The Test stage also has a 'Deploy to production' button, and the Production stage has an 'Update app' button.

DEV & TEST versions are the same

TEST & PROD versions are Different PROD is older.



# Automate Deployment Pipelines Using APIs and Azure DevOps

REST methods enable automation with:

- Azure DevOps actions
- PowerShell
- Custom code of choice

Assign Workspace	Get Pipeline Operation	Selective Deploy
Create Pipeline	Get Pipeline Operations	Unassign Workspace
Delete Pipeline	Get Pipelines	Update Pipeline
Delete Pipeline User	Get Pipeline Stage Artifacts	Update Pipeline User
Deploy All	Get Pipeline Stages	
Get Pipeline	Get Pipeline Users	

3. **Deploy** - Here you perform the deployment.

```
PowerShell Copy  
  
$url = "pipelines/{0}/Deploy" -f "Insert you pipeline ID here"  
$deployResult = Invoke-WebRequest -Uri $url -Method Post -Body $body | ConvertFrom-Json
```

4. (Optional) **Deployment completion notification** - As the deployment API is asynchronous, you can program the script to notify you when the deployment is complete.

```
PowerShell Copy  
  
$url = "pipelines/{0}/Operations/{1}" -f "Insert you pipeline ID here",$deployResult.id  
$operation = Invoke-WebRequest -Uri $url -Method Get | ConvertFrom-Json  
while($operation.Status -eq "NotStarted" -or $operation.Status -eq "Executing")  
{  
    # Sleep for 5 seconds  
    Start-Sleep -s 5  
    $operation = Invoke-WebRequest -Uri $url -Method Get | ConvertFrom-Json  
}
```

# Certified & Governed Data Models

# Endorsing Certified Datasets & Reports

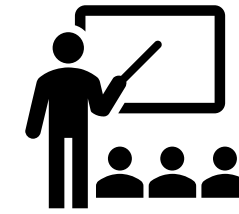
- Part of the organization's governance & adoption plan
- Governance policy sets criteria for validation & endorsement
- Data steward owns source data & validates dataset trustworthiness

The screenshot displays the Power BI Enterprise Workshop interface. At the top, it shows 'Power BI Enterprise Workshop' and 'Production' environment. A navigation bar includes options like '+ New', 'View in pipeline', 'View', 'Filters', 'Settings', 'Access', and 'Search'. Below this, there are tabs for 'All', 'Content', 'Datasets + dataflows', and 'Datamarts (Preview)'. The main area features a table with the following columns: Name, Type, Owner, Refreshed, Next refresh, Endorsement, and Sensitivity. Two rows are visible, both owned by 'Enterprise Workshop' and refreshed on '8/8/22, 9:57:45 AM'. The first row is a 'Report' named 'Contoso Sales' with a 'Promoted' endorsement status. The second row is a 'Dataset' named 'Contoso Sales' with a 'Certified' endorsement status. A yellow 'Update app' button is located in the top right corner.

Name	Type	Owner	Refreshed	Next refresh	Endorsement	Sensitivity
Contoso Sales	Report	Enterprise Workshop	8/8/22, 9:57:45 AM	—	Promoted	—
Contoso Sales	Dataset	Enterprise Workshop	8/8/22, 9:57:45 AM	N/A	Certified	—

# Separating Data Models & Reports

# Planning for Separation – data models & reports



## The Thick and Thin of Reports

Separate reports and data models can be:

- Versioned
- Deployed & managed separately
- Central “Certified” dataset
- Use Live Connect when creating new reports



[Doing Power BI the Right Way: 7.](#)

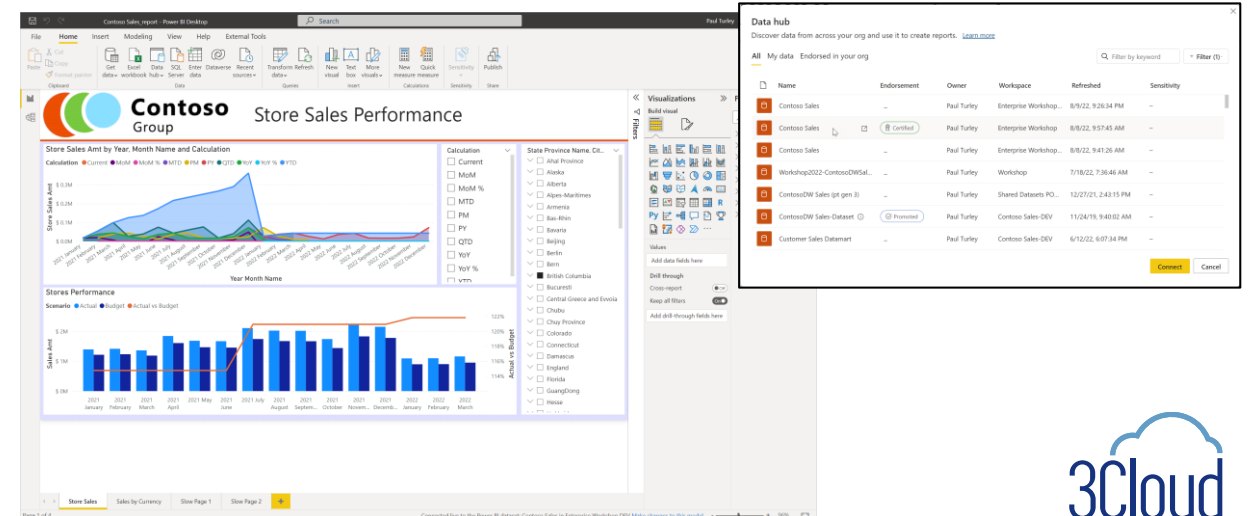
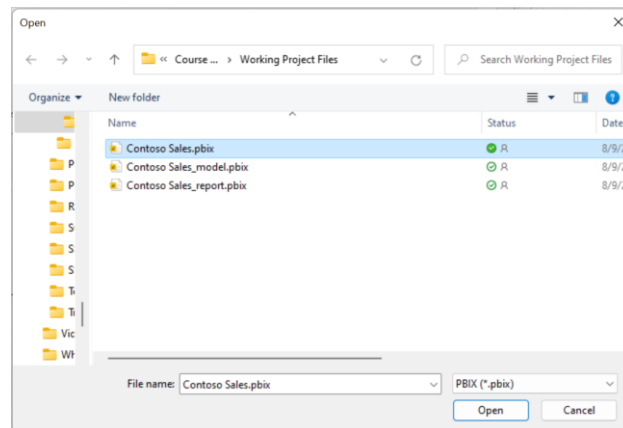
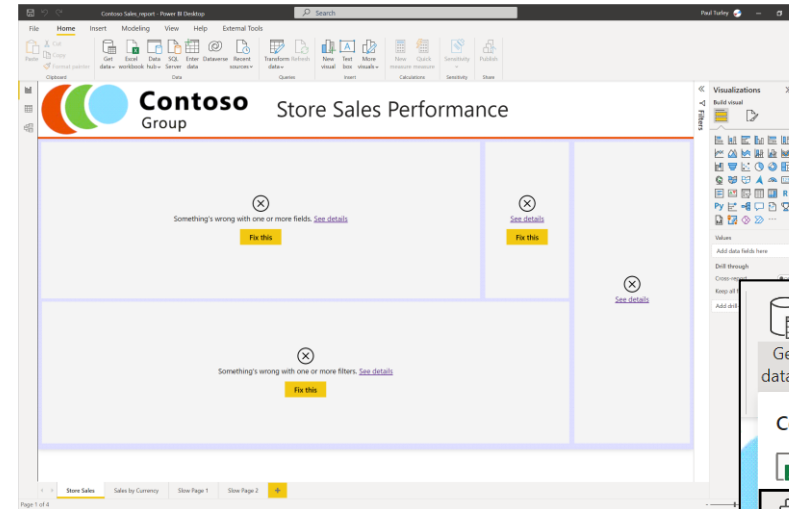
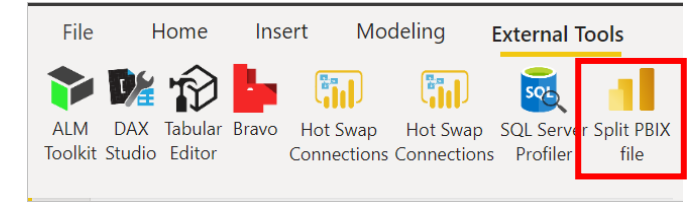
[Planning for separation – data models and reports | Paul Turley's SQL Server BI Blog](#)

[5 Tips for Separating Power BI Datasets and Reports — Coates Data Strategies](#)

# Separating an Existing Model & Report

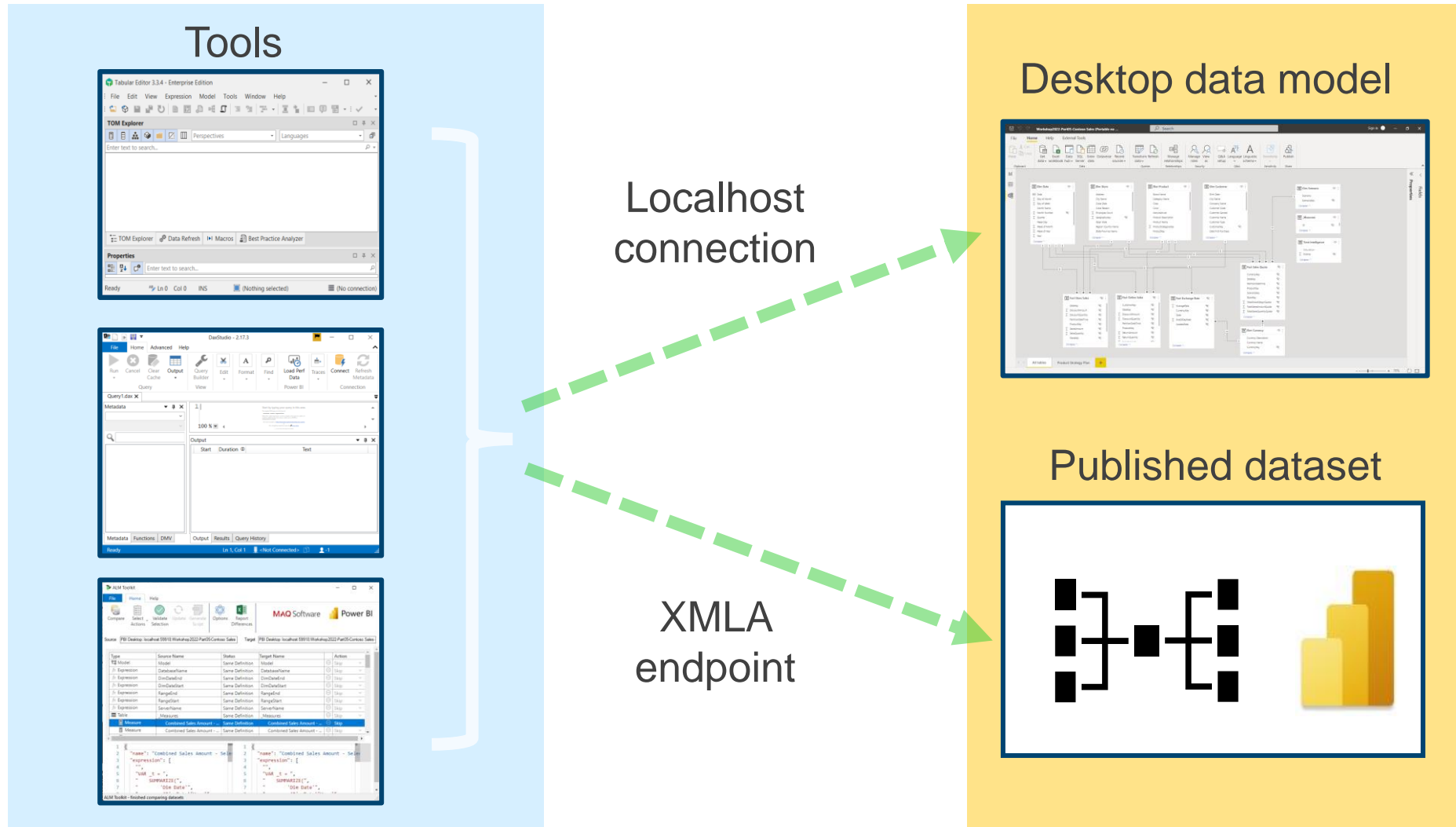
SPLIT PBIX FILE EXTERNAL TOOL

- Split PBIX & Hot Swap Connections developed by Steve Campbell
- Installs with Business Ops – free tools from PowerBI.tips
- Removes data model & connection from new report file
- Reconnect report to published dataset



# Analyzing & Tuning Data Models

## CONNECTING TO LOCAL AND REMOTE DATA MODELS



# Using Performance Analyzer

...

Name	Duration (ms)
Refreshed visual	-
Combined Sales Amt by Date	223
Combined Sales Amt by Subcategory Name	92
Table	348
Cross-highlighted	-
Combined Sales Amt by Date	218
Table	362
DAX query	206
Visual display	50
Other	105
Copy query	-

```
// DAX Query
DEFINE
    VAR __DS0FilterTable =
        TREATAS({"Refrigerators"}, 'Dim Product'[Subcategory Name])

    VAR __DS0Core =
        SUMMARIZECOLUMNS (
            ROLLUPADDISSUBTOTAL('Dim Customer'[Company Name],
                "IsGrandTotalRowTotal"),
            __DS0FilterTable,
            "Online_Sales_Performance_Summary", '_Measures'[Online Sales
                Performance Summary],
            "v_Online_Sales_Performance_Summary_FormatString",
            IGNORE(' _Measures'[_Online Sales Performance Summary FormatString])
        )

    VAR __DS0PrimaryShowAll =
        ADDMISSINGITEMS (
            'Dim Customer'[Company Name],
            __DS0Core,
            ROLLUPISSUBTOTAL('Dim Customer'[Company Name],
                [IsGrandTotalRowTotal]),
            __DS0FilterTable
        )

    VAR __DS0PrimaryWindowed =
        TOPN(502, __DS0PrimaryShowAll, [IsGrandTotalRowTotal], 0, 'Dim
            Customer'[Company Name], 1)

EVALUATE
    __DS0PrimaryWindowed

ORDER BY
    [IsGrandTotalRowTotal] DESC, 'Dim Customer'[Company Name]
```



# Performing Analysis with DAX Studio

## CLEAR CACHE, QUERY PLAN & SERVER TIMINGS

```
// DAX Query
DEFINE
  VAR __DS0FilterTable =
    TREATAS({"Refrigerators"}, 'Dim Product'[Subcategory Name])

  VAR __DS0Core =
    SUMMARIZECOLUMNS (
      ROLLUPADDISSUBTOTAL('Dim Customer'[Company Name],
        "IsGrandTotalRowTotal"),
      __DS0FilterTable,
      "Online_Sales_Performance_Summary", '_Measures'[Online
        Sales Performance Summary],
      "v_Online_Sales_Performance_Summary_FormatString",
      IGNORE ('_Measures'[_Online Sales Performance Summary
        FormatString])
    )

  VAR __DS0PrimaryShowAll =
    ADDMISSINGITEMS (
      'Dim Customer'[Company Name],
      __DS0Core,
      ROLLUPISSUBTOTAL('Dim Customer'[Company Name],
        [IsGrandTotalRowTotal]),
      __DS0FilterTable
    )

  VAR __DS0PrimaryWindowed =
    TOPN(502, __DS0PrimaryShowAll, [IsGrandTotalRowTotal], 0,
      'Dim Customer'[Company Name], 1)

EVALUATE
  __DS0PrimaryWindowed

ORDER BY
  [IsGrandTotalRowTotal] DESC, 'Dim Customer'[Company Name]
```

The screenshot shows the DAX Studio application interface. A 'Connect' dialog box is open at the top, showing the 'PBI / SSDT Model' selected with the data source 'Workshop2022-Part05-Contoso Sales (Portable)'. Below the dialog is the main application window with a menu bar (File, Home, Advanced, Help, Layout) and a toolbar containing buttons for Run, Cancel, Clear Cache, Output, Query Builder, Edit, Format, Find, Load Perf Data, All Queries, Query Plan, Server Timings, Connect, and Refresh Metadata. The main window displays a query plan for 'Query2.dax' with a tree view on the left showing folders for \_Measures, Dim Currency, Dim Customer, Dim Date, Dim Product, Dim Scenario, Dim Store, Fact Exchange Rate, Fact Online Sales, Fact Sales Quota, Fact Store Sales, and Time Intelligence. The right pane shows the query text and a 'Server Timings' table.

Total	SE CPU	Line	Subclass	Duration	CPU	Par.	Rows	KB	Query
259 ms	63 ms	2	Scan	1	0		388		4 SELECT 'Dim Custo
	x3.7	4	Scan	2	0		388		4 SELECT 'Dim Custo
		6	Scan	1	0		733		6 SELECT 'Dim Date'
		8	Scan	2	0		733		12 WITH \$Expr0 := (P
242 ms	17 ms	10	Scan	11	63	x5.7	1,487		18 SELECT 'Dim Custo
93.4%	6.6%	12	Scan	0	0		733		12 WITH \$Expr0 := (P
<b>SE Queries</b>									
6									
<b>SE Cache</b>									
1									
16.7%									

# Evaluations, evaluations...



[https://evals.datagrillen.com/evals\\_vienna.aspx](https://evals.datagrillen.com/evals_vienna.aspx)

# Resources

# Model Design Guidelines

- Dimensional design concepts haven't changed in 20 years & are as true as ever
- Dimensional modeling “rules” should be followed but can be relaxed for Power BI in certain cases, such as:
  - Leaving some dimensional attributes in fact tables
  - Use natural keys rather than generating surrogate keys
- The art of dimensional modeling ranges from simple to complex. Start with the basics.
- Flattened “spreadsheet” models are OK for small, informal projects but have significant limitations
- As models grow in size & complexity, data quality challenges will surface that can be solved by implementing proper governance controls

The Kimball Method: <https://www.kimballgroup.com/data-warehouse-business-intelligence-resources/kimball-techniques/dimensional-modeling-techniques>

Lawrence Corr, Model Storming Agile method: <https://modelstorming.com/hierarchy-map>

# Enterprise Scale Options

In many ways, Power BI has now surpassed the capabilities of SQL Server Analysis Services. Microsoft are investing in the enterprise capabilities of the Power BI platform by enhancing Power BI Premium Capacity, adding Paginated Report and features to support massive scale specialized use cases. Consider the present and planned capabilities of the Power BI platform; before, choosing another data modeling tool such as SSAS.

## Resources:

<https://sqlserverbi.blog/2018/07/27/power-bi-for-grownups>

<https://sqlserverbi.blog/2018/12/13/data-model-options-for-power-bi-solutions>