Microsoft

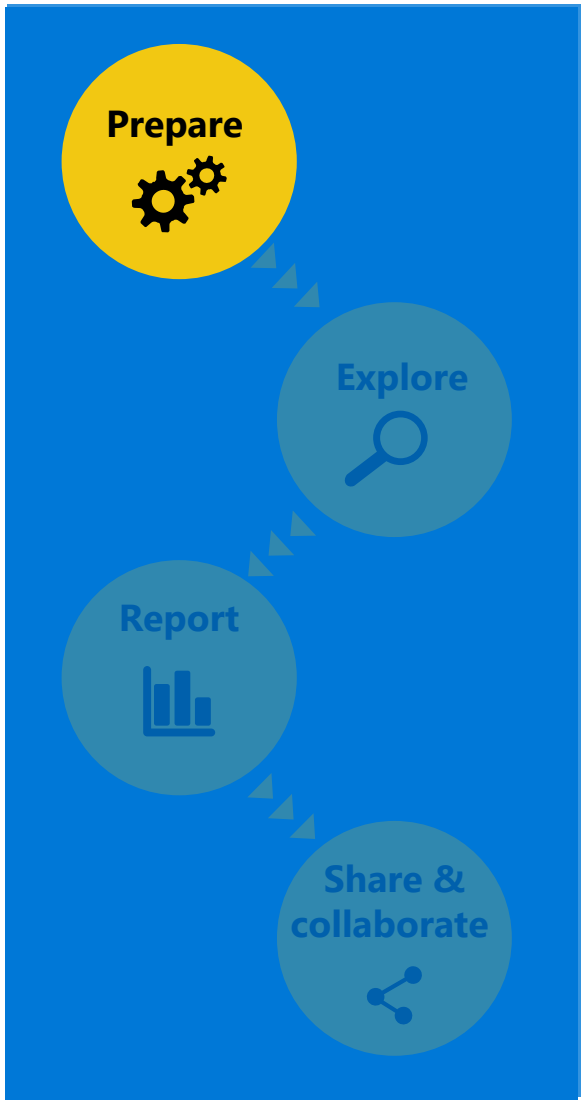
Business Analytics with Power BI

Microsoft Services

Module 1: Power BI Desktop

Lesson 3: Shaping Data

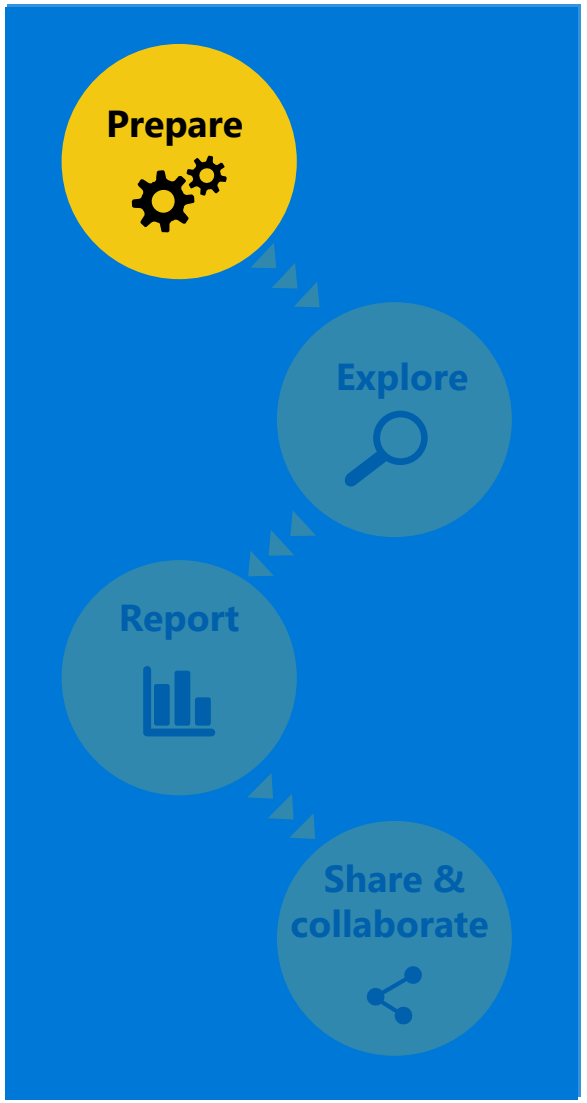
Shaping Data



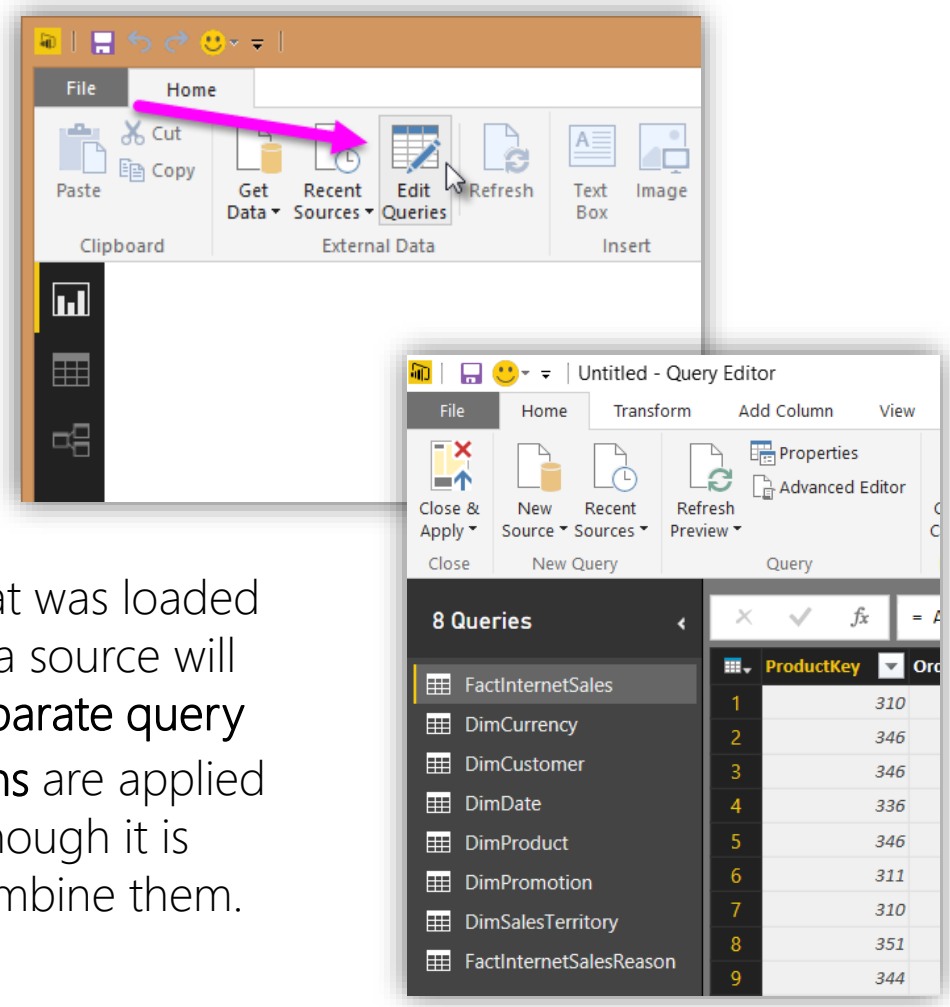
- Shaping is the act of **transforming** the data to meet our **business requirements**
- Apply **data cleaning operations** and correct/remove errors
- **Rename the data** so it is more meaningful
- **Combine data** from different data sources in a single table
- Create **personal** (reusable) **views** of the data
- A set of **out-of-the box transformations** are available to help
- A **programming language** is available for addressing more complex use-cases
- **Source data is not changed**, all the transformations are applied on the Desktop engine

Shaping Data

Query Editor



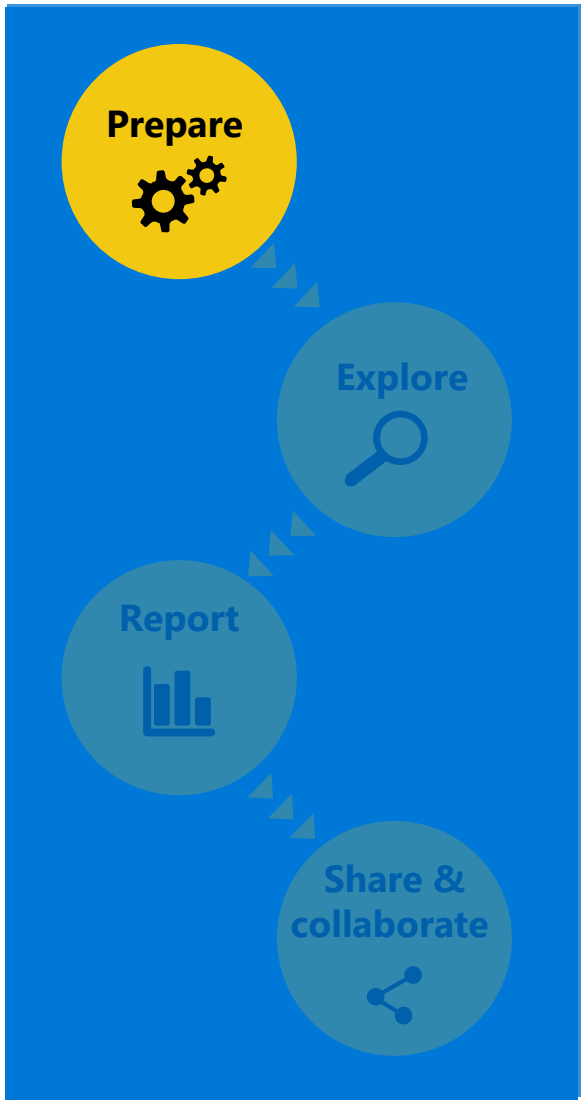
- After the data is loaded or referenced into Power BI Desktop, the **"Edit Queries"** option in the "**Home**" ribbon allows the edition of the imported entities



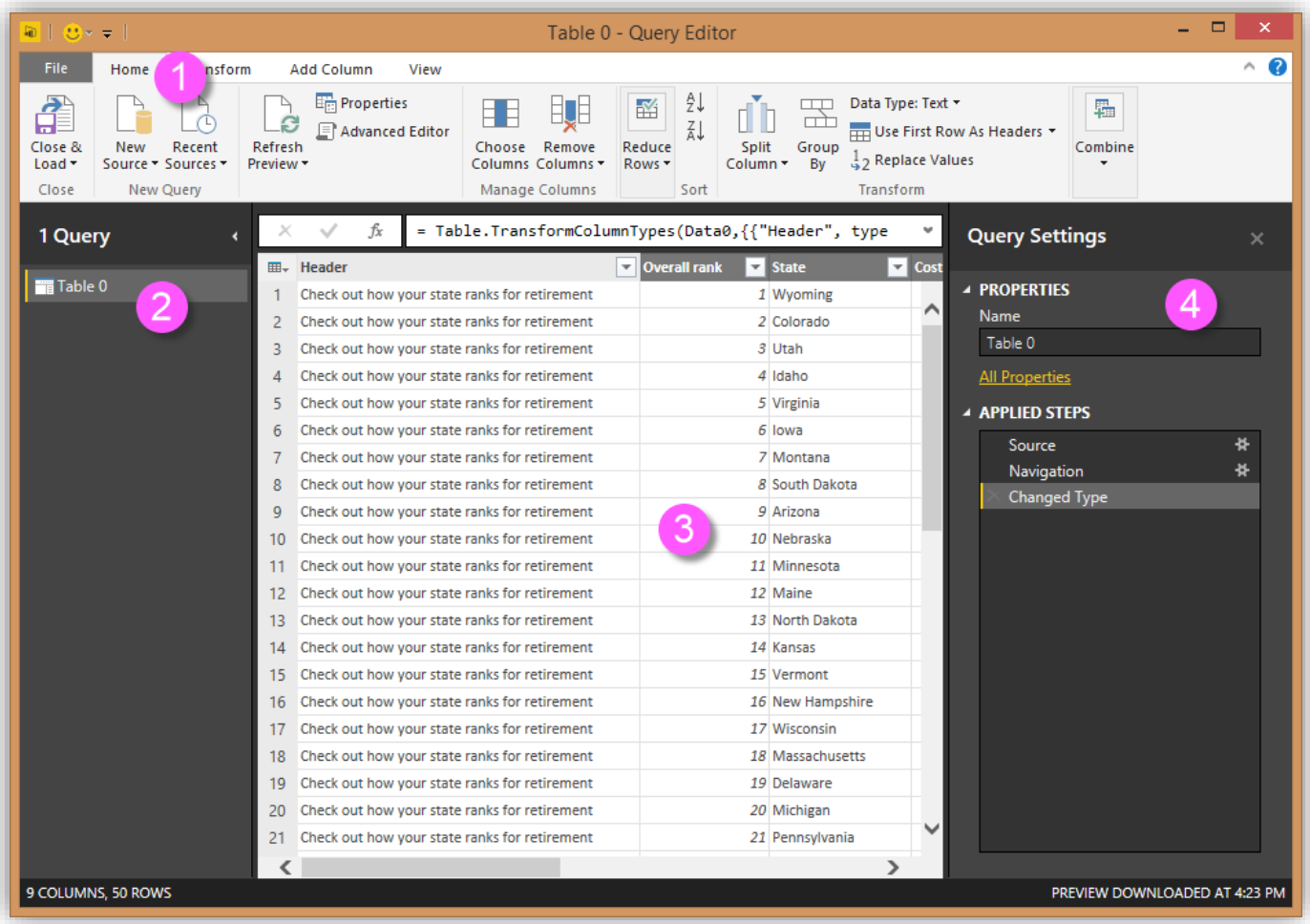
- **Each entity** that was loaded from each data source will generate a **separate query**
- **Transformations** are applied **per-entity**, although it is possible to combine them.

Shaping Data

Query Editor

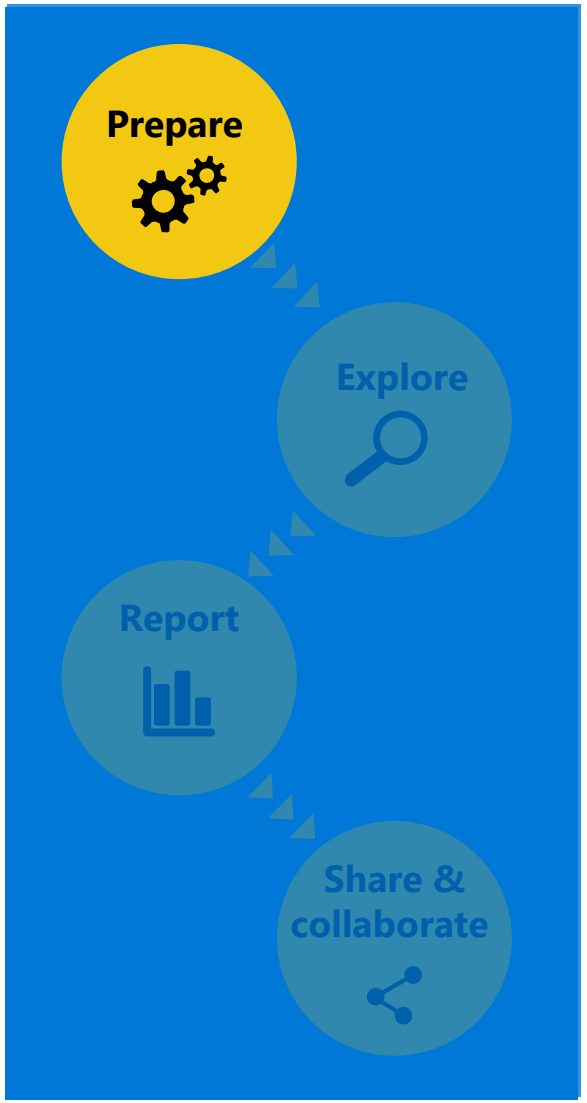


1. Query Ribbon
2. The Left Pane
3. The Center Pane
4. The Query Settings Pane

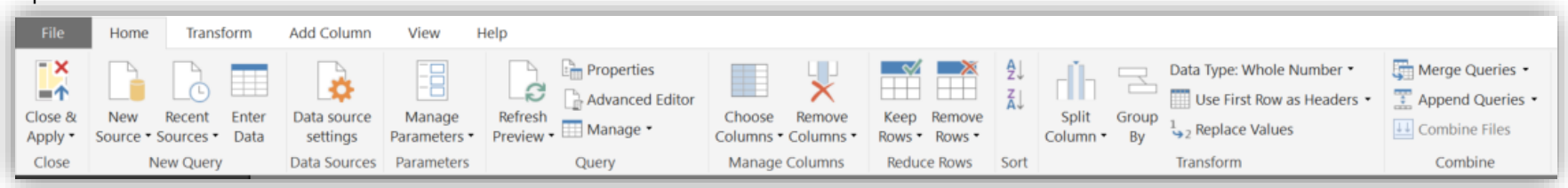


Shaping Data

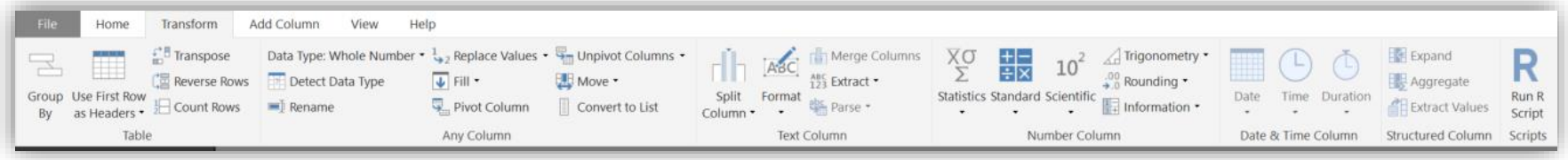
Query Editor – Query Ribbon



- The **"Home"** tab contains the common query tasks including the combination of queries

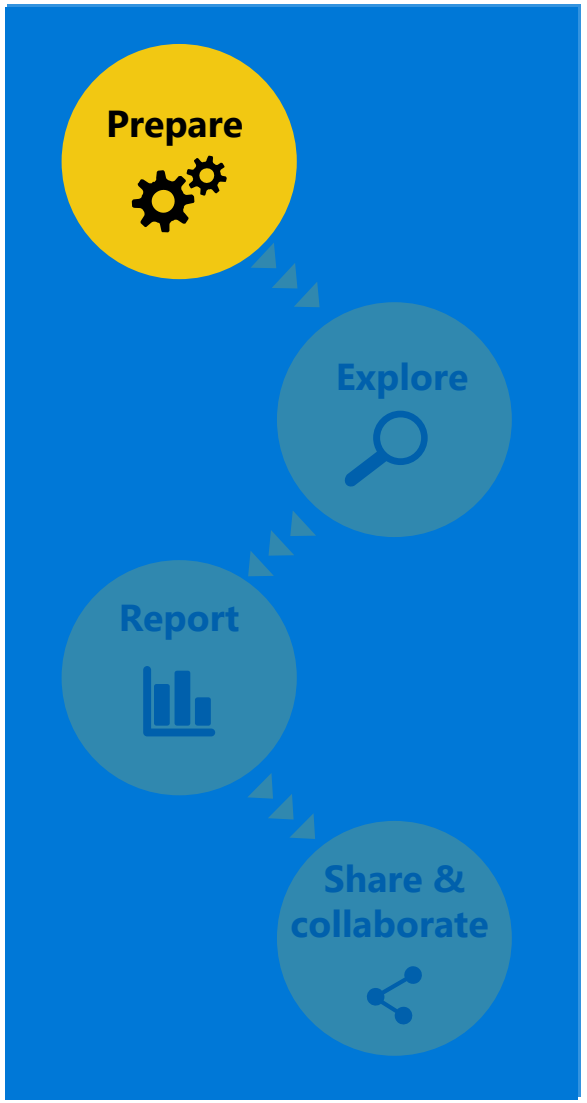


- The **"Transform"** tab provides access to **common data transformation tasks**, such as adding or removing columns, changing data types, splitting columns, and other data-driven tasks

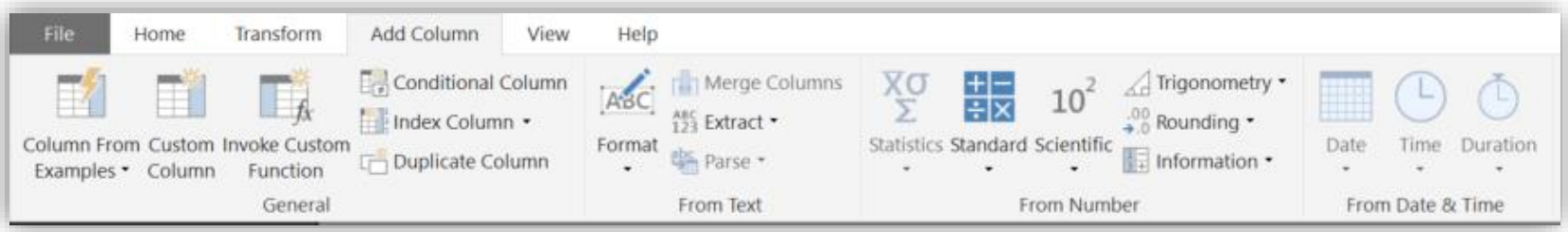


Shaping Data

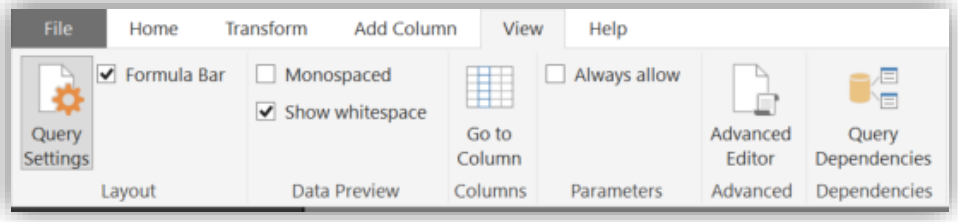
Query Editor – Query Ribbon



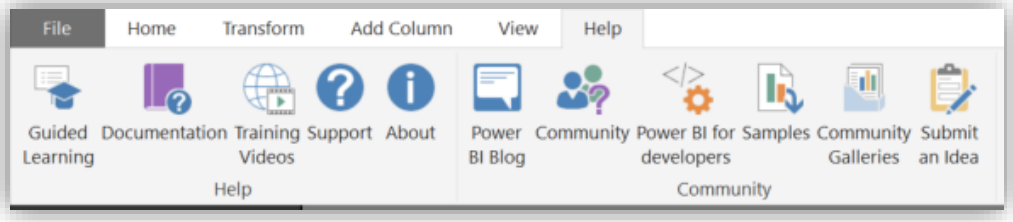
- The **"Add Column"** tab provides additional tasks associated with columns such as formatting column data, adding custom columns or invoking functions



- The **"View"** tab provides access to **query settings** and the **Advanced Editor** where we can develop our own data transformation scripts

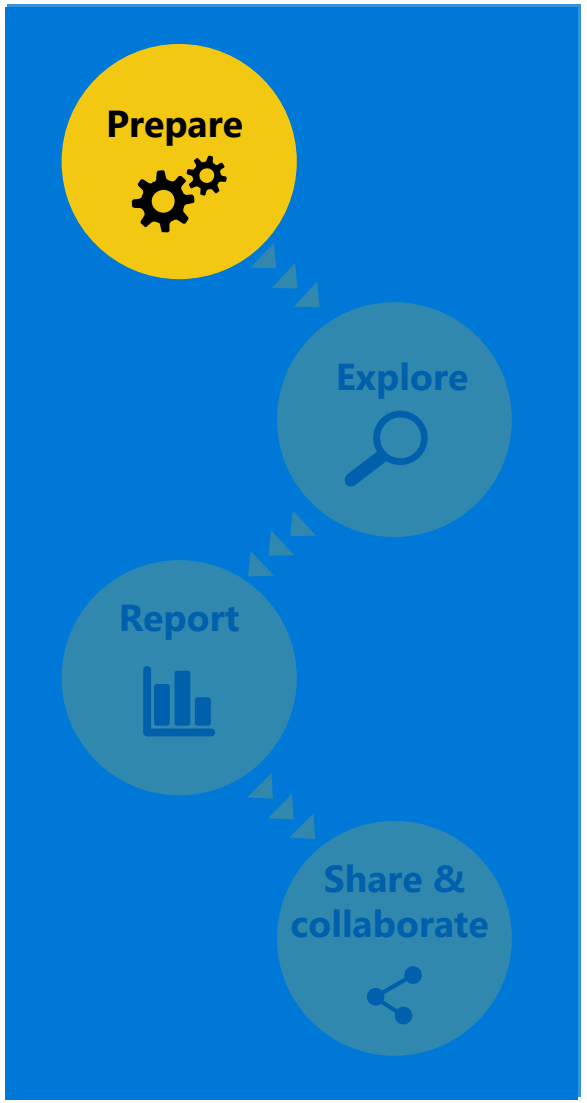


- The **"Help"** tab provides access to some **public resources** and **community**

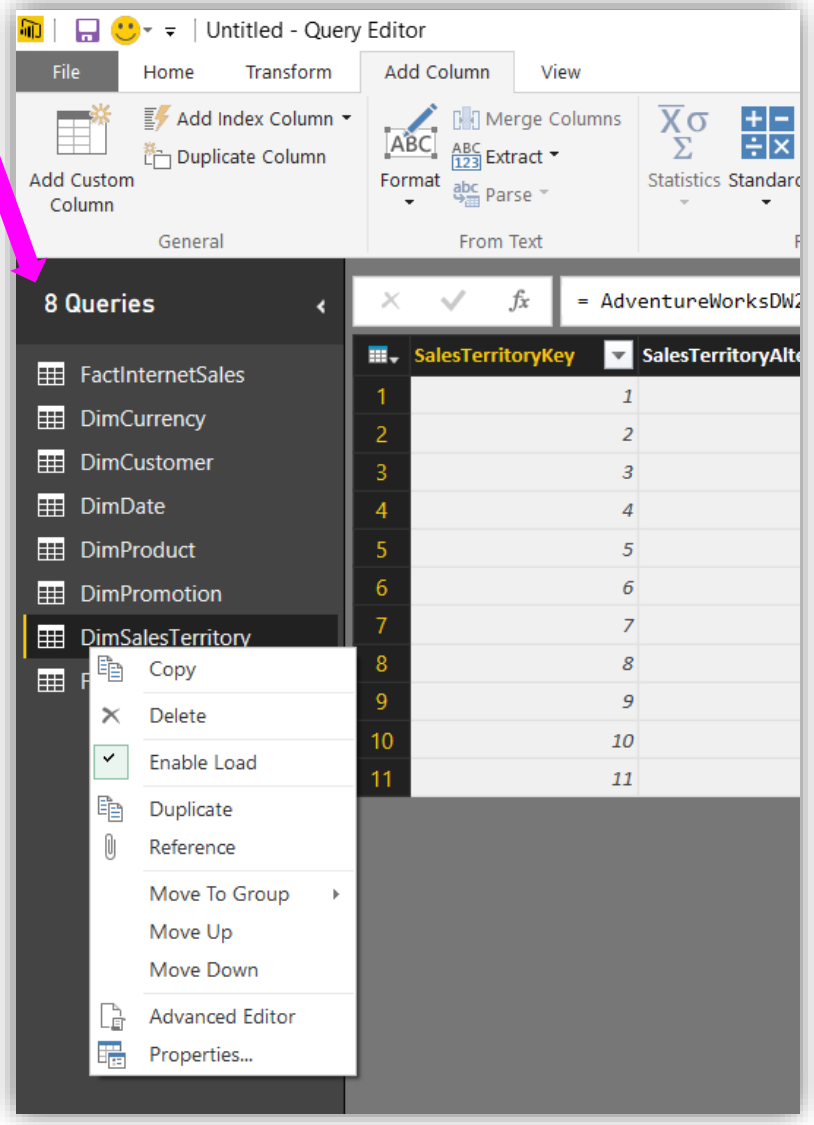


Shaping Data

Query Editor – The Left Pane

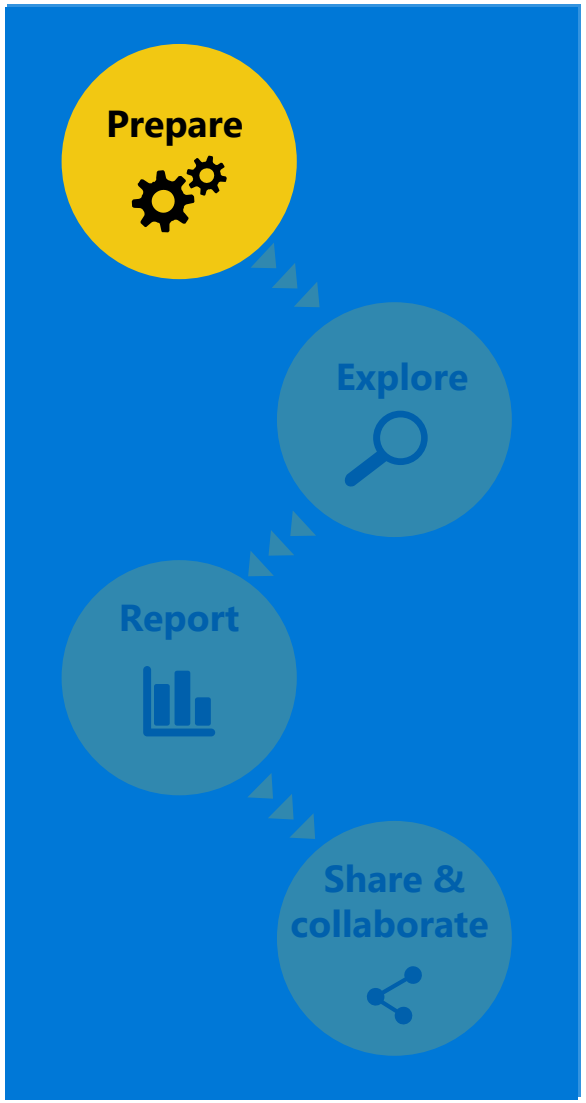


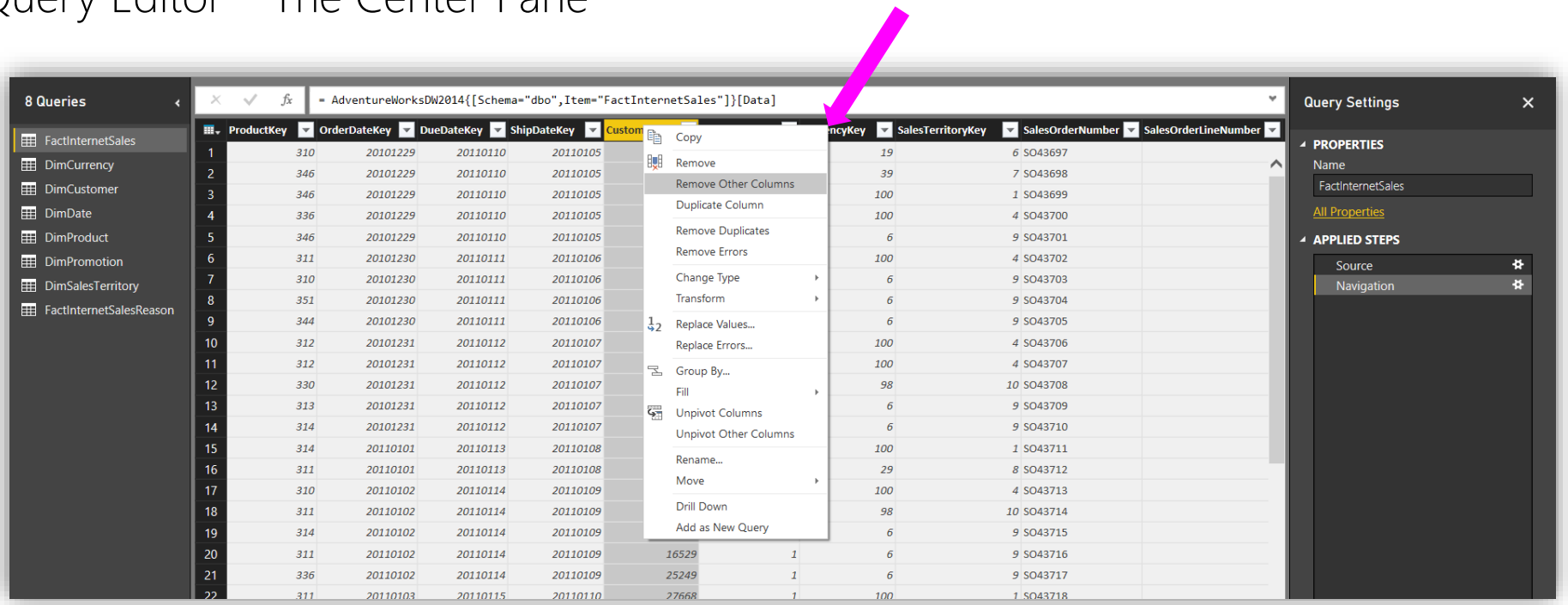
- Displays the **number of active queries**
- The **name** of each query
- Clicking on a query displays the query preview on the Center Pane
- Also allows to **disable the query load**
- **Organizing** the queries into groups
- And also **duplicating** or creating a **reference**



Shaping Data

Query Editor – The Center Pane

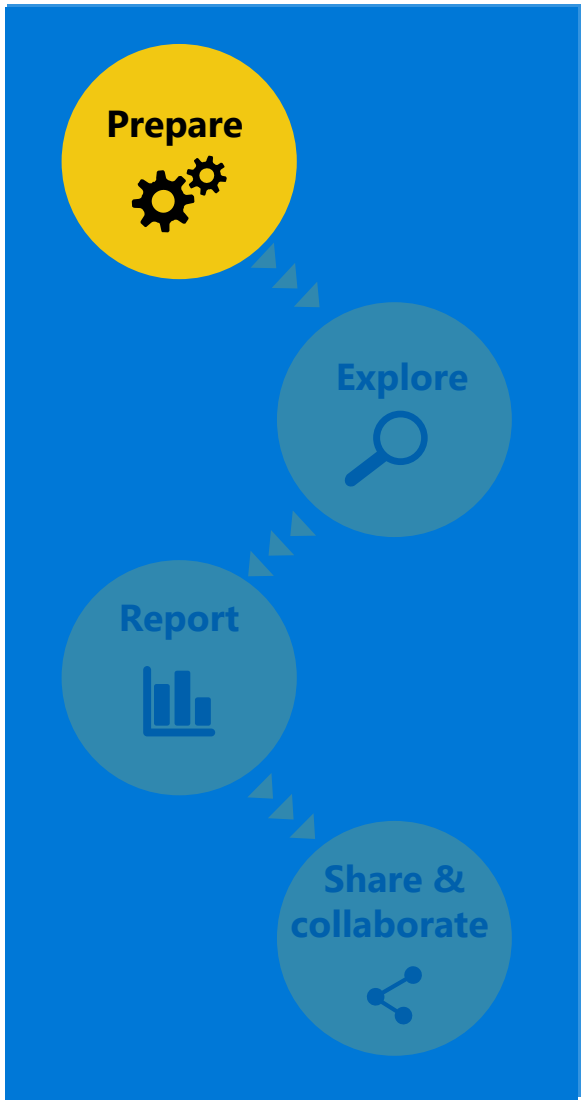


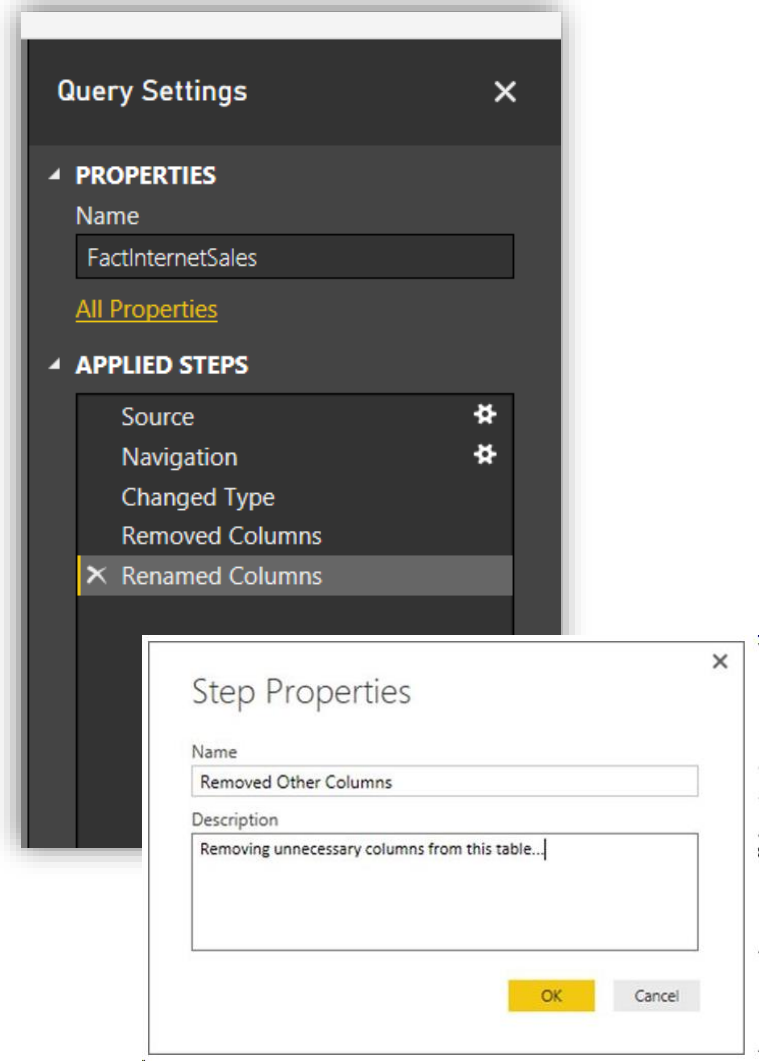


- Displays the **data for the selected query and the current transformation step**
- Many of the transformations can be directly applied here, namely, column related
- It **displays a preview of the data** and might not display the entire dataset

Shaping Data

Query Editor – The Query Settings Pane



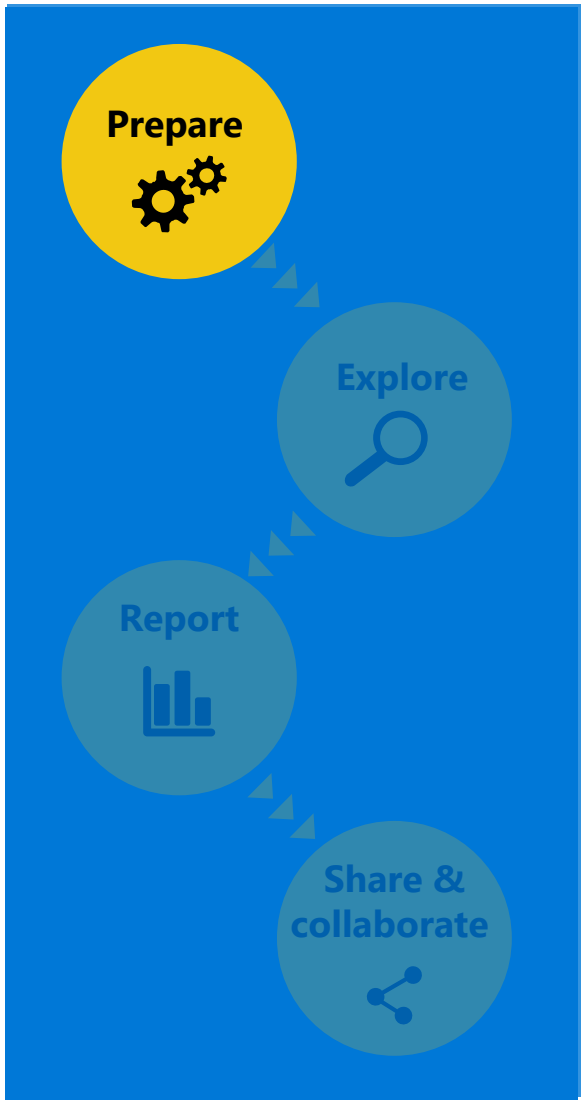


- **Each transformation** that is applied to a query generates a **step**
- Those **steps are applied in order** to generate the desired result
- **Order can be changed** as long as it respects the needs of previous steps
- It is also possible to **edit each step's properties** via this pane
- Possible to **add Description** to Query steps

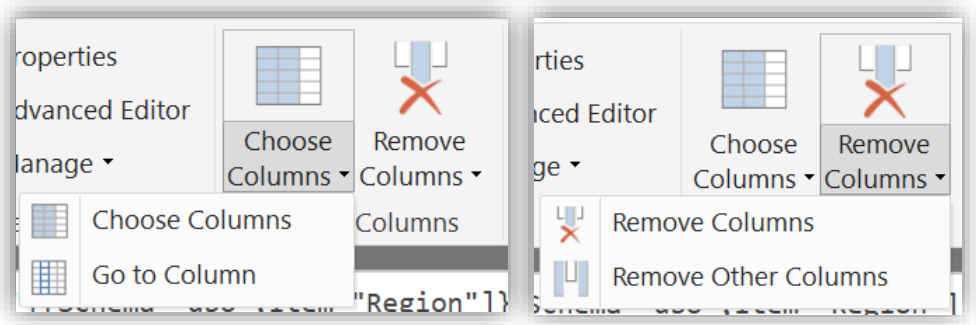
Demonstration
Building a Query

Shaping Data

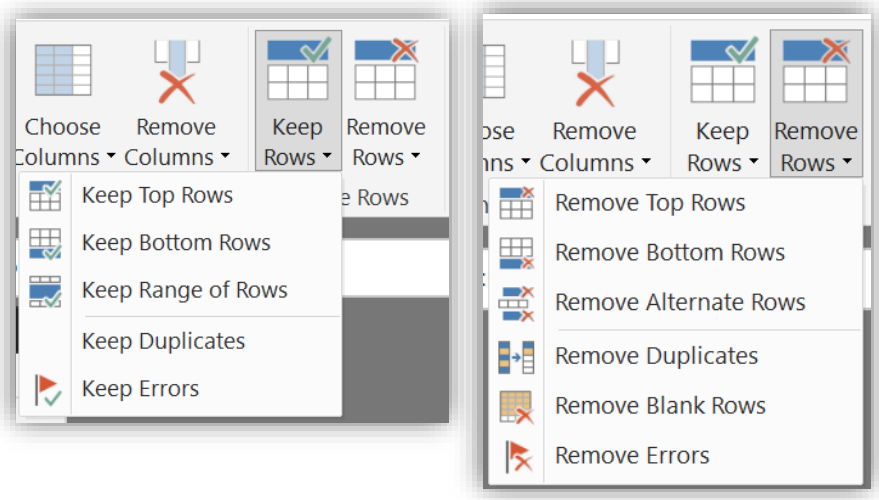
Available Data Transformations:



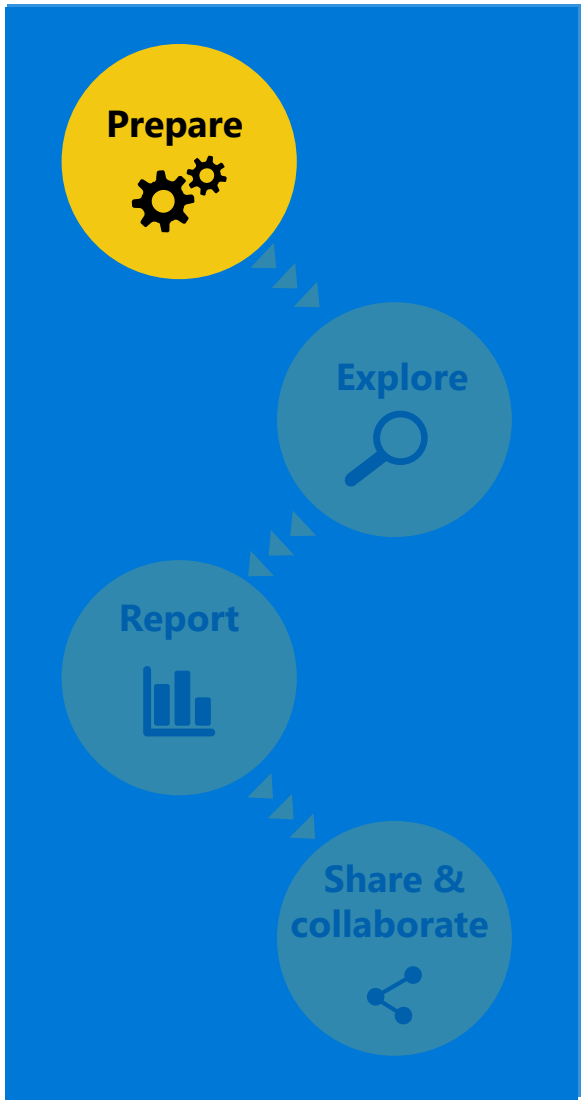
- Manage Columns
  - Choose Columns
  - Remove Columns
  - Remove Other Columns



- Reduce Rows
  - Keep Top/Bottom Rows
  - Keep Range of Rows
  - Remove Top/Bottom Rows
  - Remove Alternate Rows
  - Remove Blank Rows
  - Keep/Remove Duplicates
  - Keep/Remove Errors



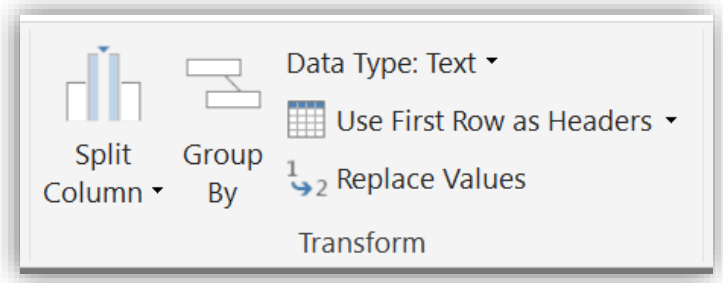
Shaping Data

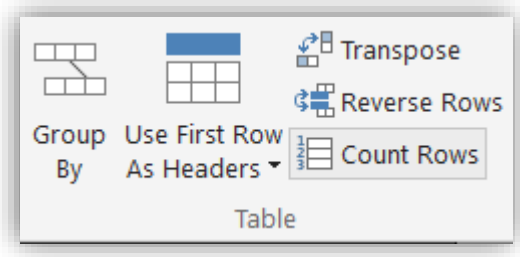


Available Data Transformations

- Sort Columns

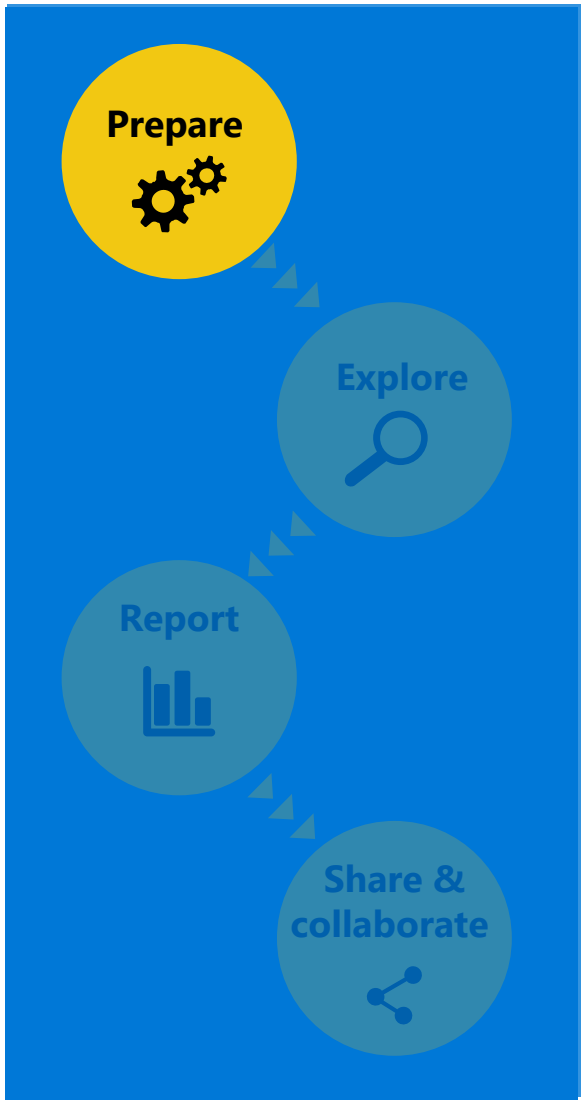
- Transform
  - Merge/Split columns
  - Group rows in a table
  - Aggregate data from a table
  - Use first Row as headers
  - Replace Values
  - Transpose
  - Reverse Rows
  - Count Rows
  - Conditional Columns



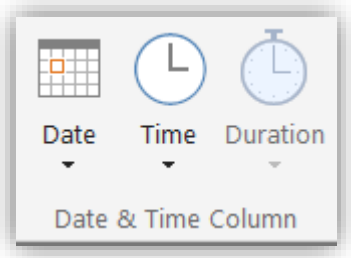


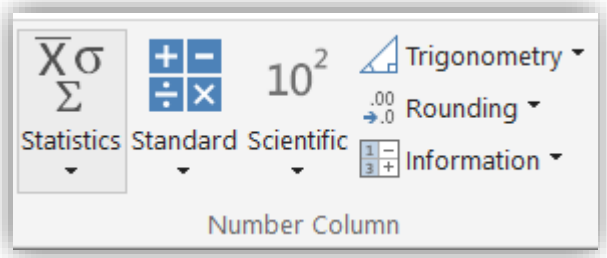
Shaping Data

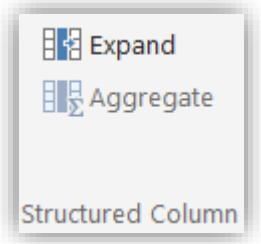
Available Data Transformations

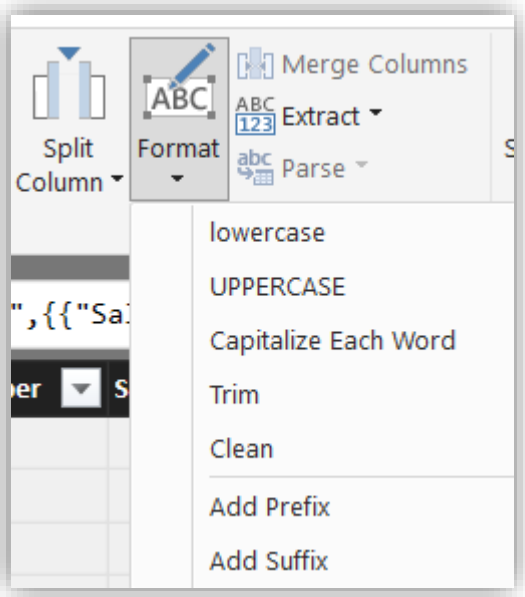


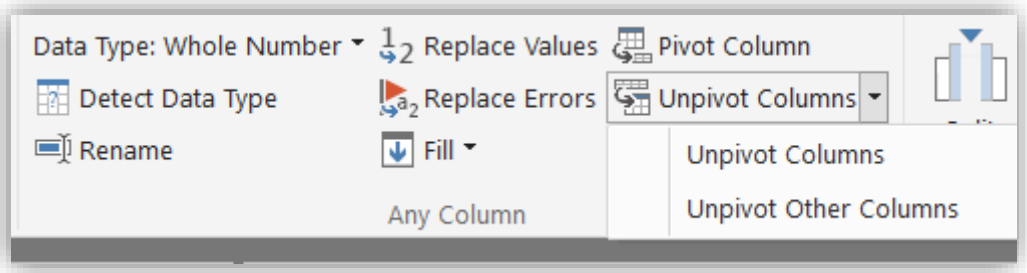
- Transform
  - Filter Table
  - Pivot/Unpivot
  - Change Data Types
  - Text/Number/Date Formatting
  - Extract parts of strings
  - Fill Rows
  - Expand Rows from a related table





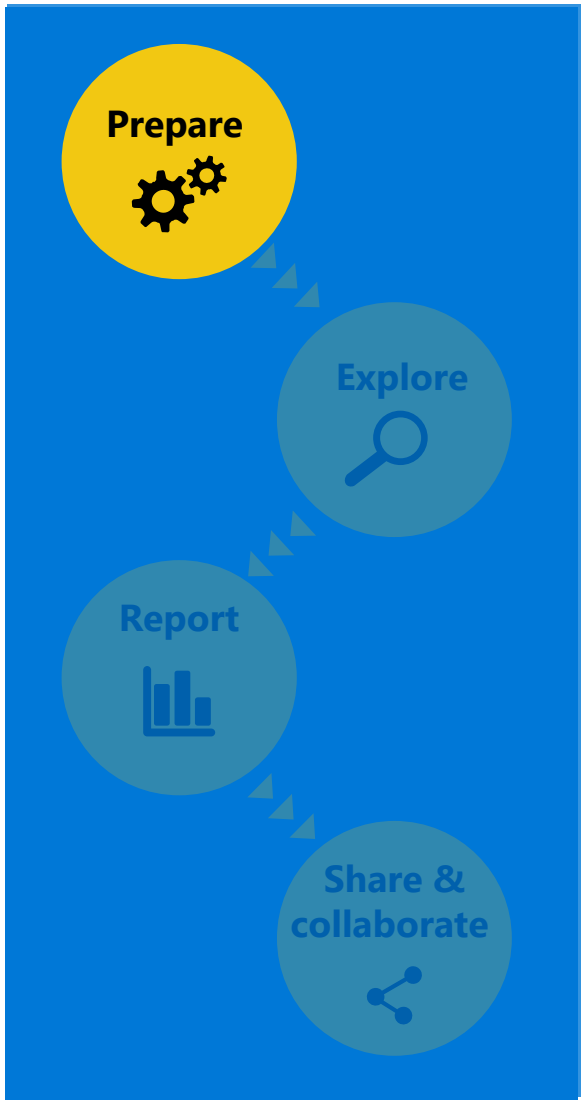




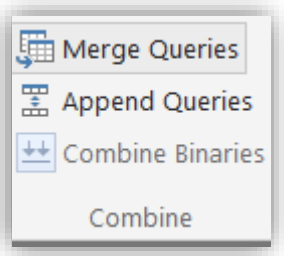


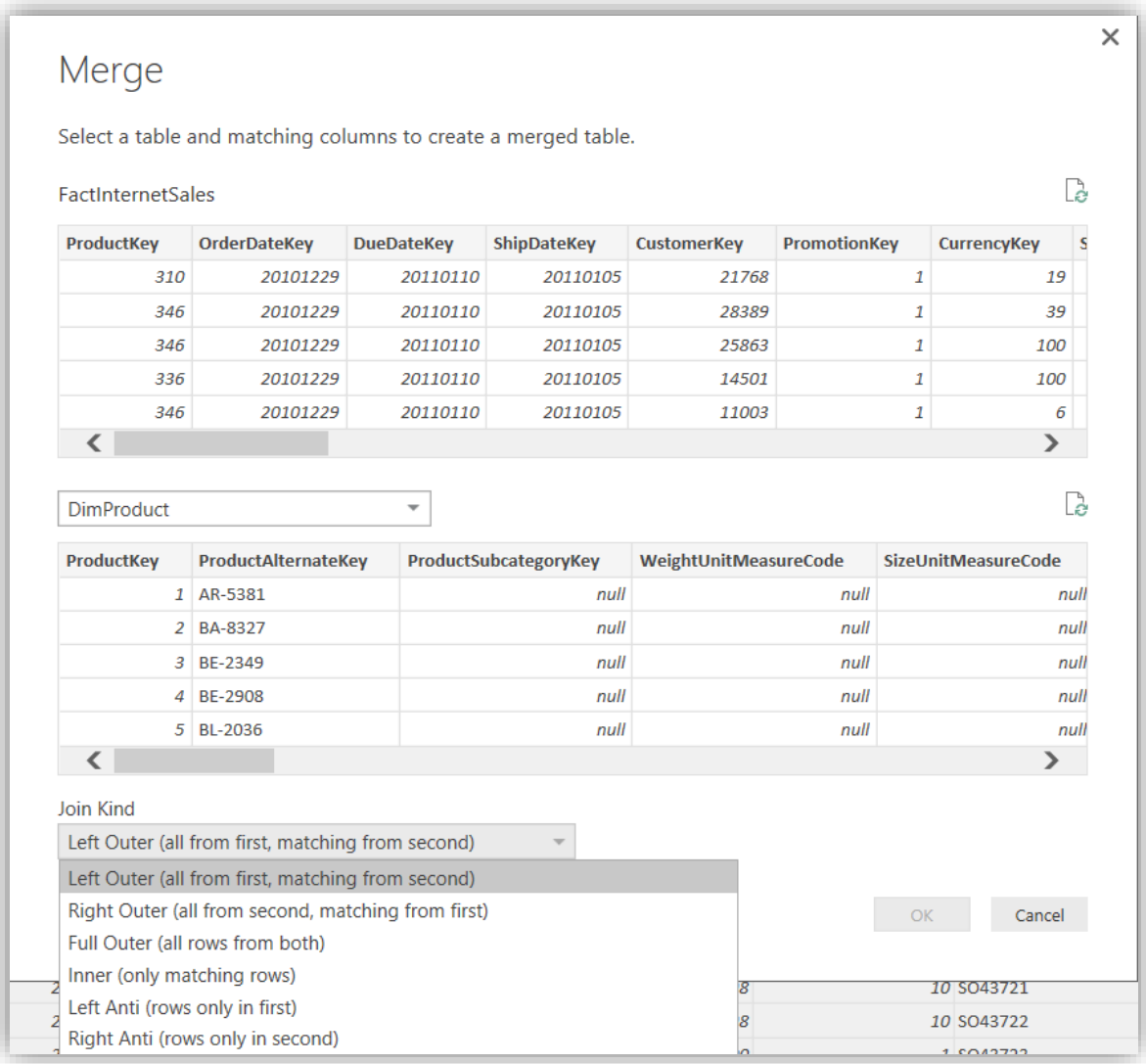
Shaping Data

Available Data Transformations



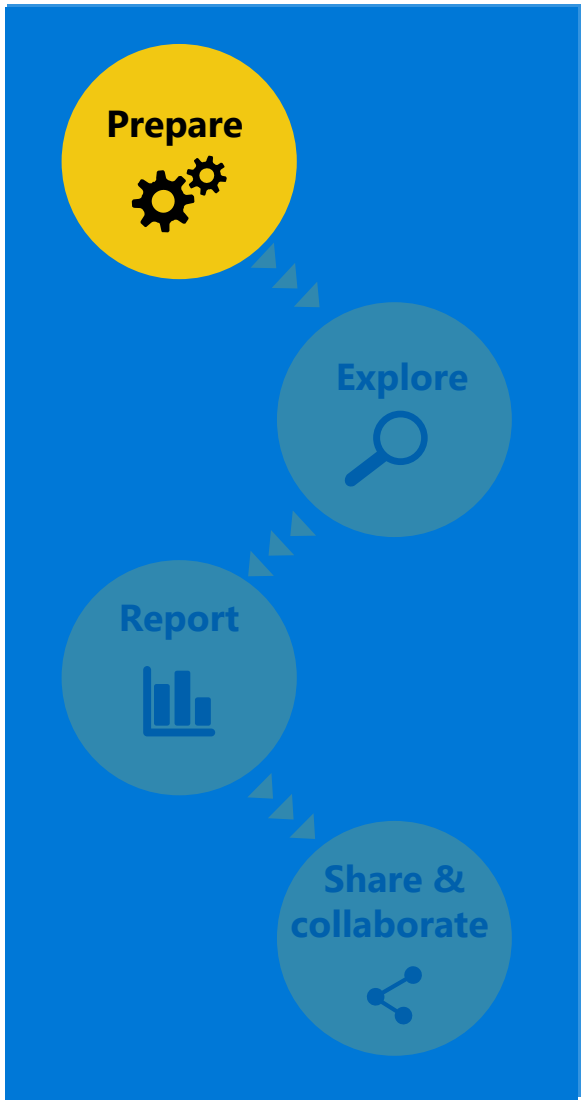
- Combine
  - Merge Queries
  - Append Queries
  - Combine Binaries

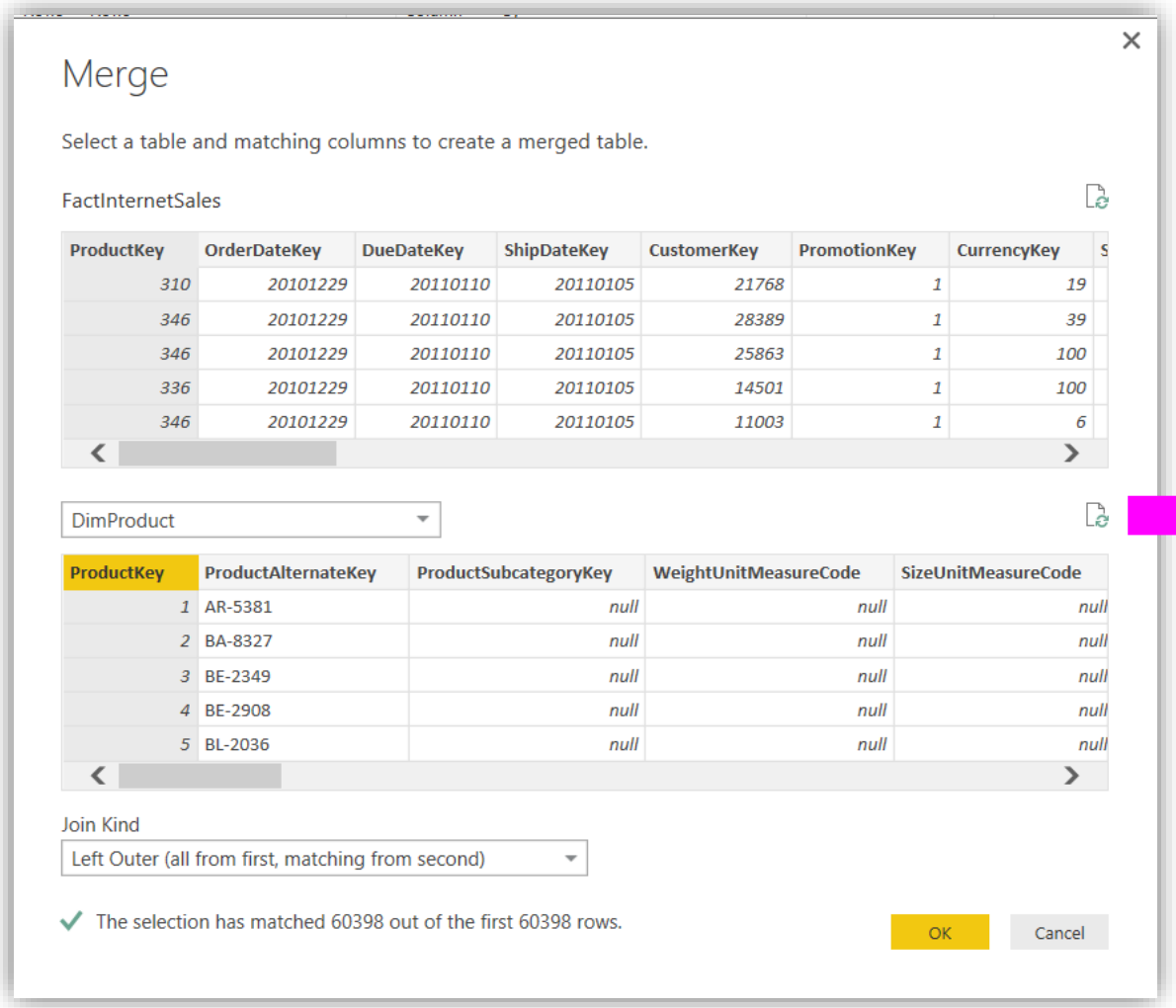


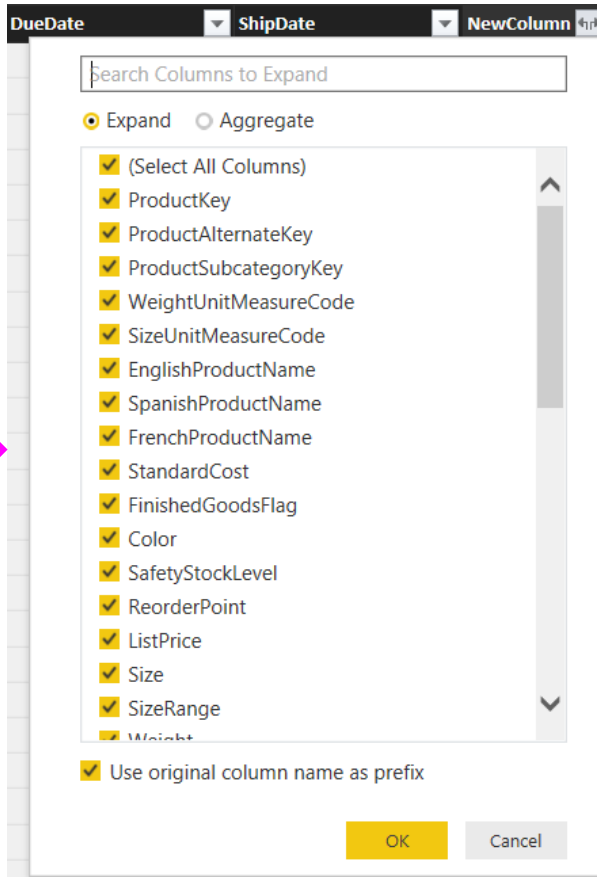


Shaping Data

Available Data Transformations - Merge

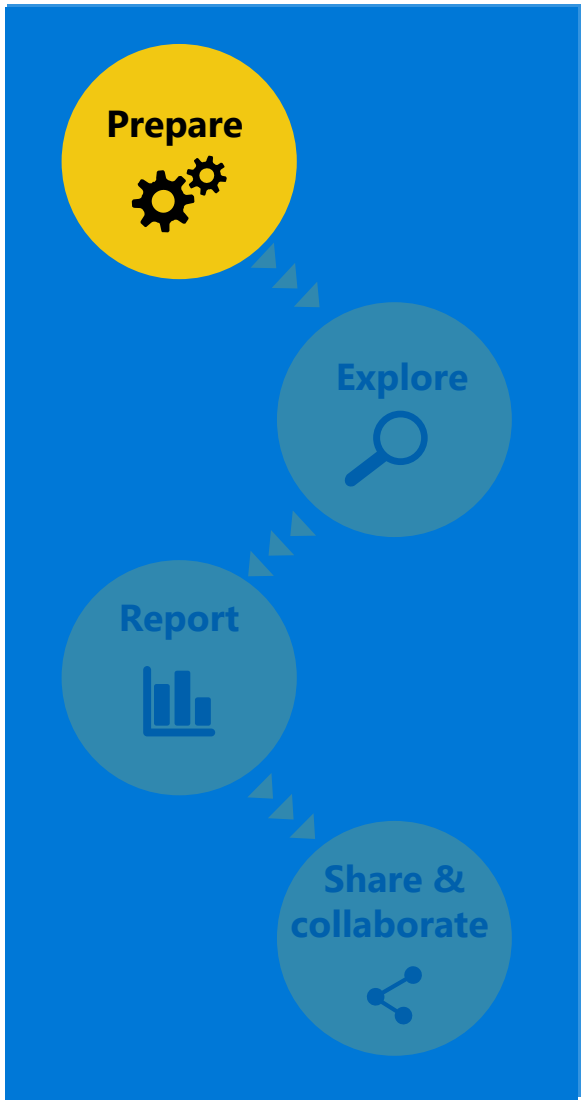




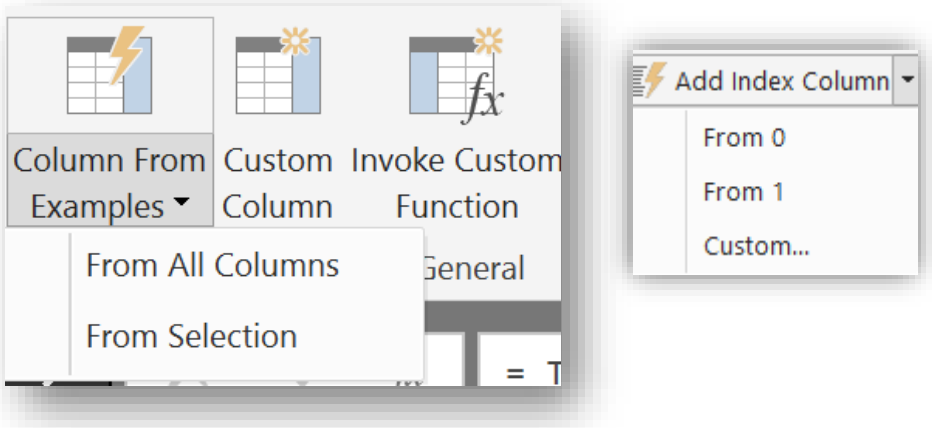


Shaping Data

Available Data Transformations

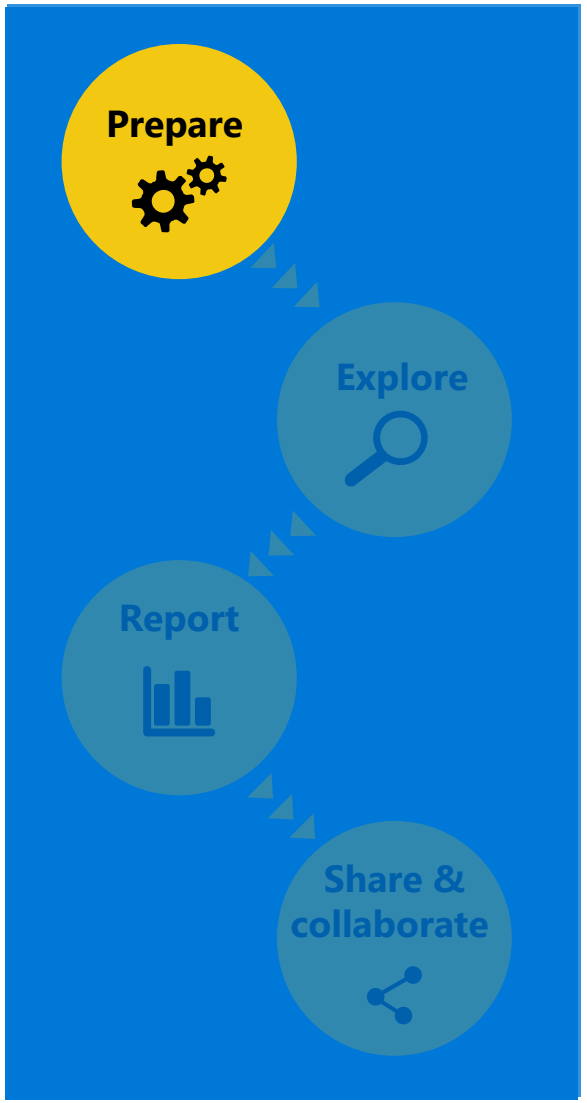


- Add Column
  - Insert Custom Column (also used to invoke functions)
  - Insert Index Column
  - Duplicate Column
  - Column From Examples

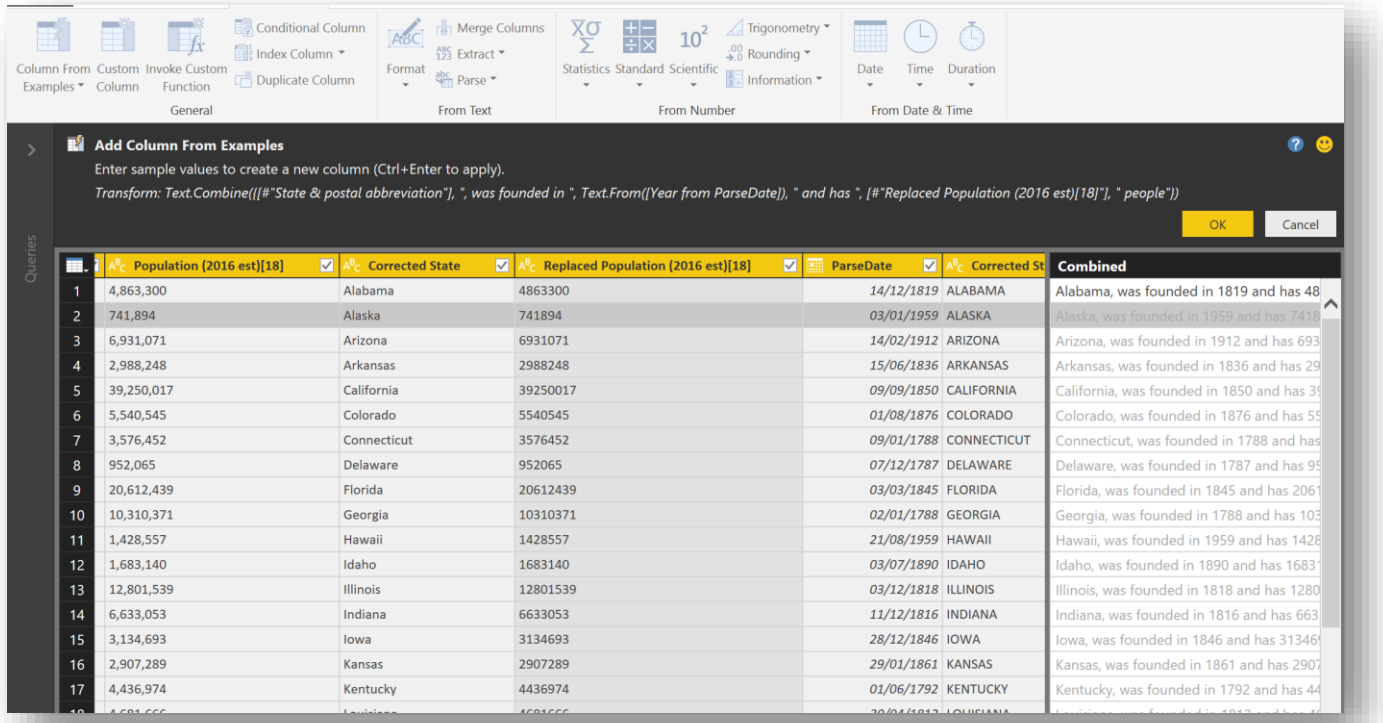


Shaping Data

Available Data Transformations

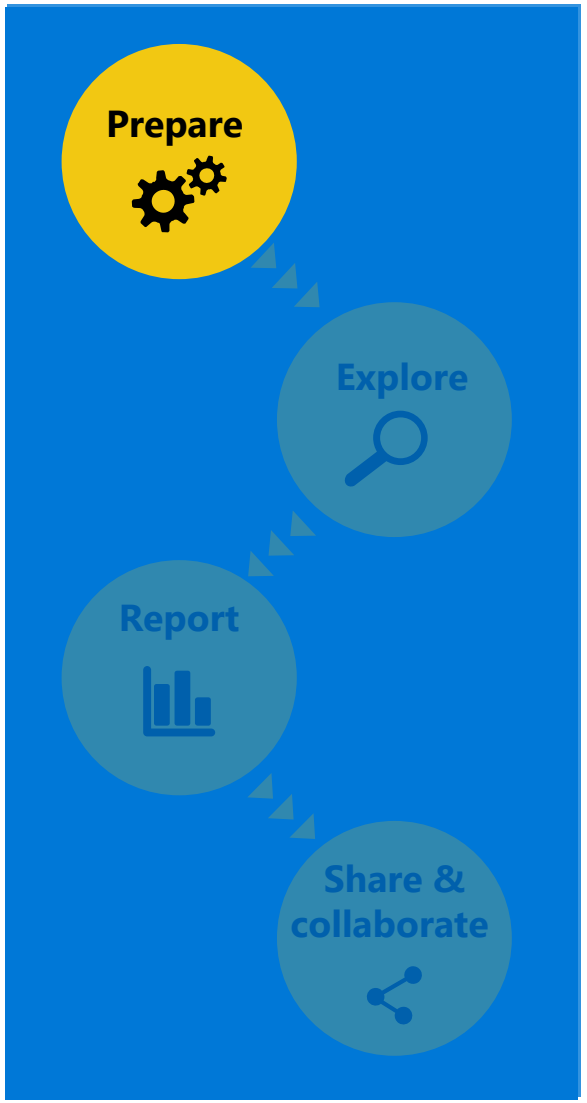


- Column From Examples
  - Useful when the result is known, but not the transformations to apply or where they are
  - Generates an expression automatically (similar to Excel's Flash Fill)

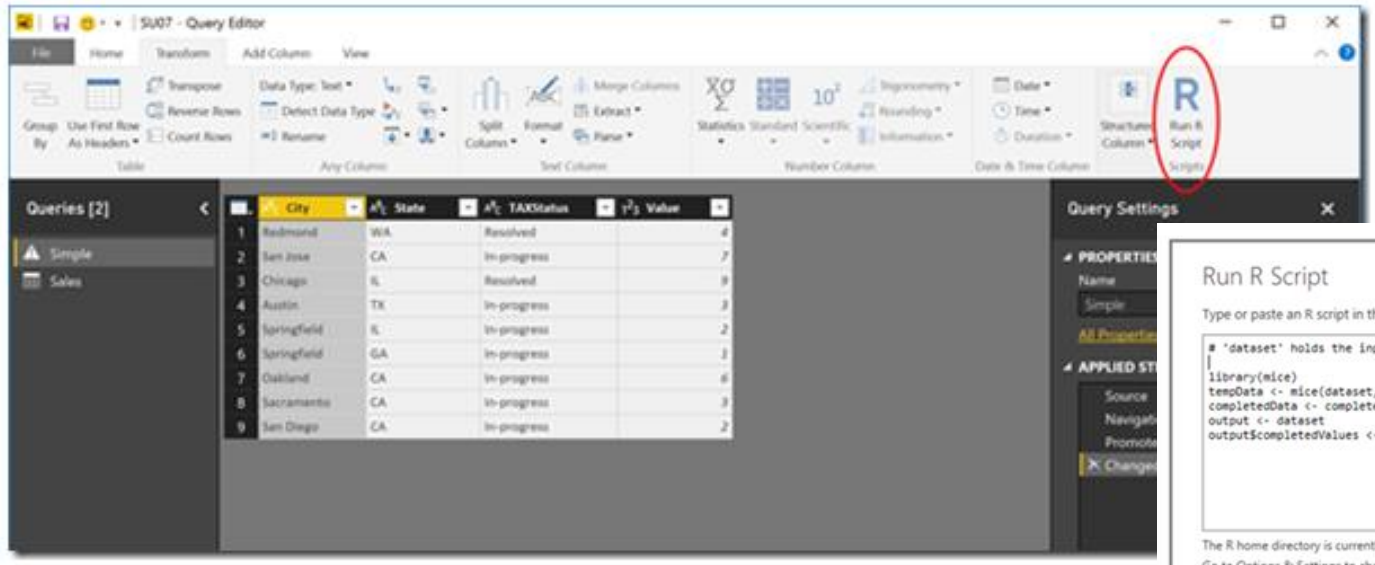


Shaping Data

Available Data Transformations – R Scripts

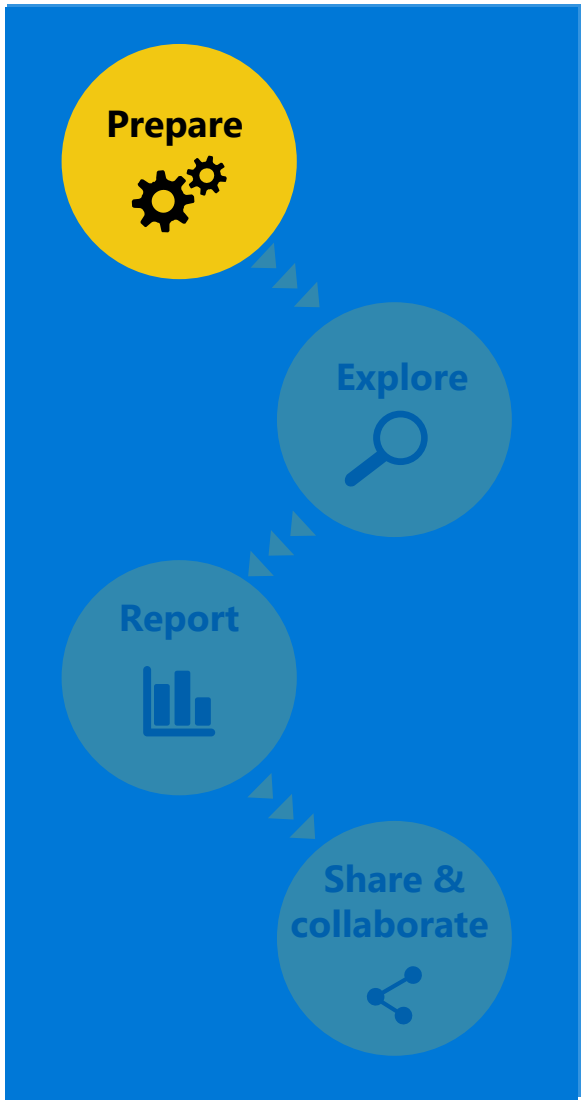


- **Run R Script** from the Query Editor
- Perform **data cleansing, advanced data shaping, and analytics** including completion of missing data, predictions, and clustering etc.
- R should be **installed locally**



Shaping Data

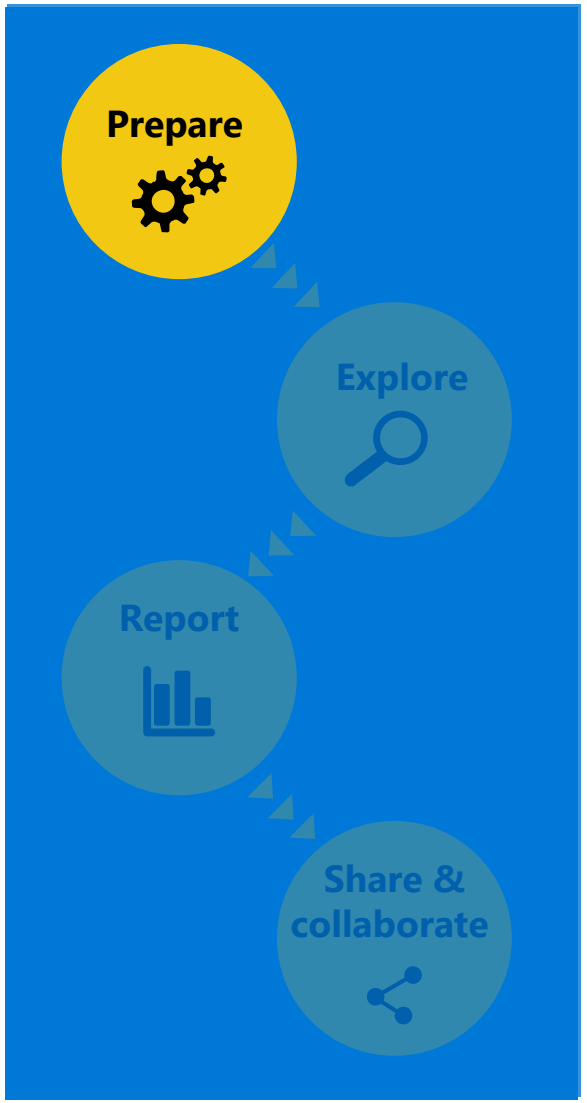
Available Data Transformations – Advanced Editor



- Advanced Editor
  - Every transformation generates code automatically
  - For more advanced scenarios
  - OOB transformations will cover more than 90% of the cases

Shaping Data

An example (how to go from a matrix to a table?)

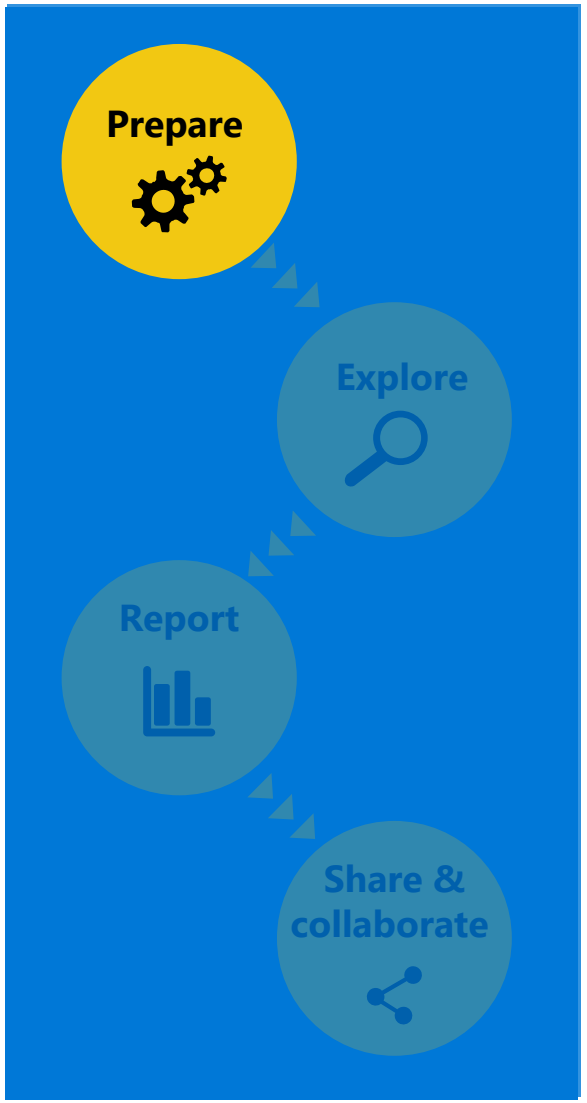


	Column1	Column2	Column3	Column4	Column5
1	*null*	Northland Region	*null*	Auckland Region	*null*
2	*null*	All Ages	Under 20	All Ages	Under 20
3	*1991*	*2475*	*263*	*18347*	*1346*
4	*1992*	*2438*	*240*	*18183*	*1323*

	Region	Age Group	Attribute	Value
1	Northland Region	All Ages	1991	*2475*
2	Northland Region	All Ages	1992	*2438*
3	Northland Region	Under 20	1991	*263*
4	Northland Region	Under 20	1992	*240*
5	Auckland Region	All Ages	1991	*18347*
6	Auckland Region	All Ages	1992	*18183*
7	Auckland Region	Under 20	1991	*1346*
8	Auckland Region	Under 20	1992	*1323*

Shaping Data

An example (how to go from a matrix to a table?)



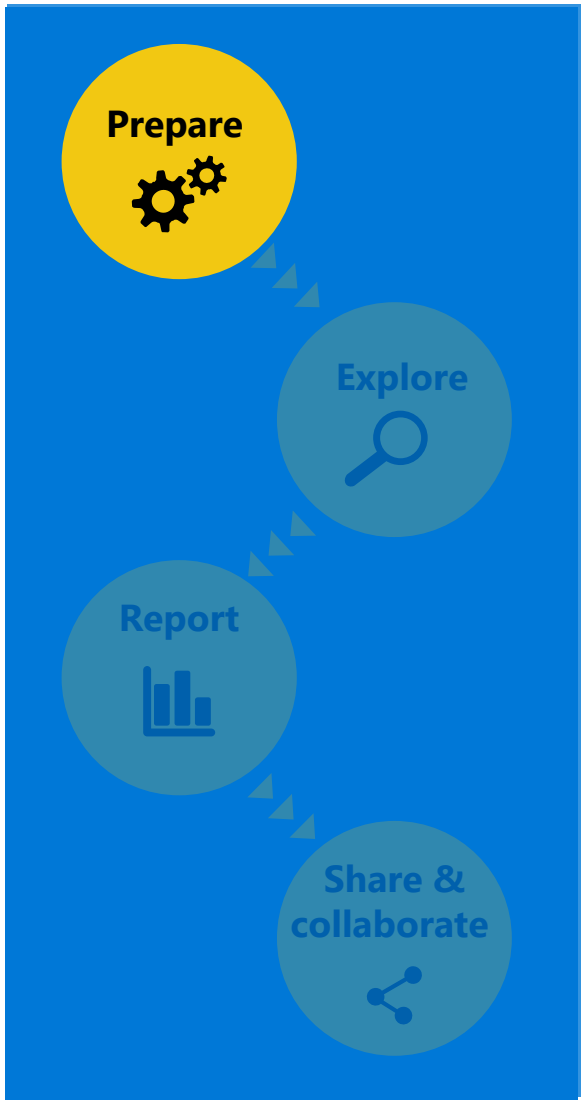
	Column1	Column2	Column3	Column4	Column5
1	*null*	Northland Region	*null*	Auckland Region	*null*
2	*null*	All Ages	Under 20	All Ages	Under 20
3	*1991*	*2475*	*263*	*18347*	*1346*
4	*1992*	*2438*	*240*	*18183*	*1323*

Transpose

	Column1	Column2	Column3	Column4
1	*null*	*null*	*1991*	*1992*
2	Northland Reg	All Ages	*2475*	*2438*
3	*null*	Under 20	*263*	*240*
4	Auckland Regi	All Ages	*18347*	*18183*
5	*null*	Under 20	*1346*	*1323*

Shaping Data

An example (how to go from a matrix to a table?)



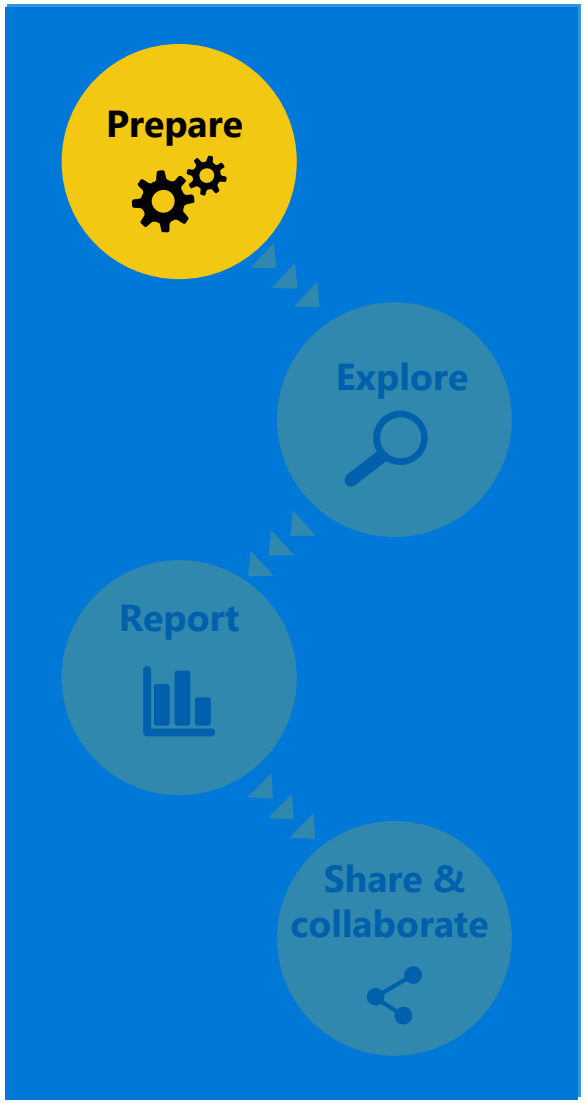
	Column1	Column2	Column3	Column4
1	*null*	*null*	*1991*	*1992*
2	Northland Reg	All Ages	*2475*	*2438*
3	*null*	Under 20	*263*	*240*
4	Auckland Regi	All Ages	*18347*	*18183*
5	*null*	Under 20	*1346*	*1323*

Use First Row as Header

	Column1	Column2	1991	1992
1	Northland Reg	All Ages	*2475*	*2438*
2	*null*	Under 20	*263*	*240*
3	Auckland Regi	All Ages	*18347*	*18183*
4	*null*	Under 20	*1346*	*1323*

Shaping Data

An example (how to go from a matrix to a table?)

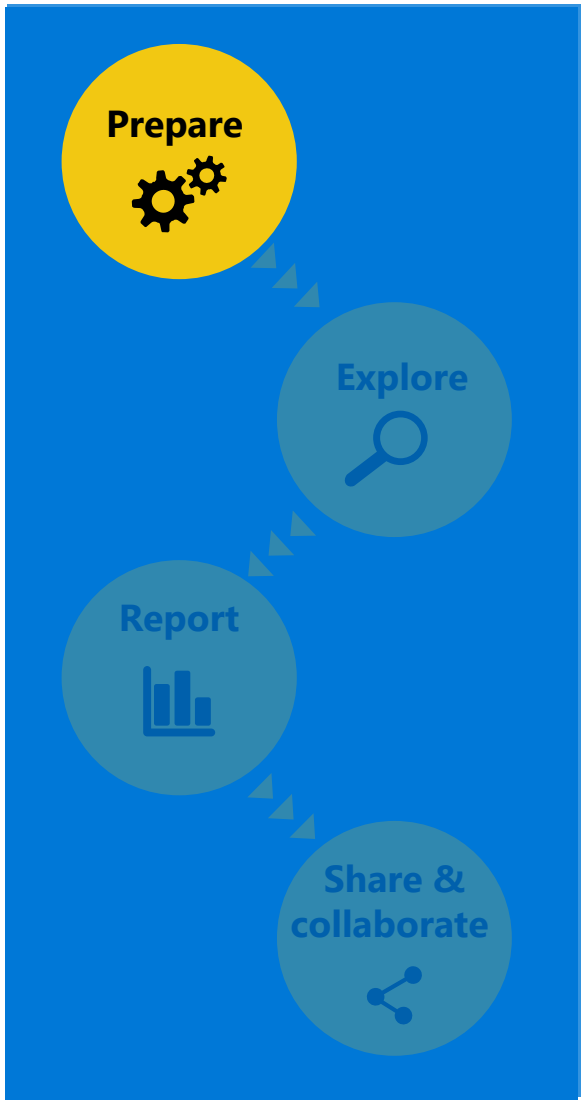


	Column1	Column2	1991	1992
1	Northland Reg	All Ages	*2475*	*2438*
2	*null*	Under 20	*263*	*240*
3	Auckland Regi	All Ages	*18347*	*18183*
4	*null*	Under 20	*1346*	*1323*

Rename Columns

	Region	Age Group	1991	1992
1	Northland Region	All Ages	*2475*	*2438*
2	*null*	Under 20	*263*	*240*
3	Auckland Region	All Ages	*18347*	*18183*
4	*null*	Under 20	*1346*	*1323*

Shaping Data



An example (how to go from a matrix to a table?)

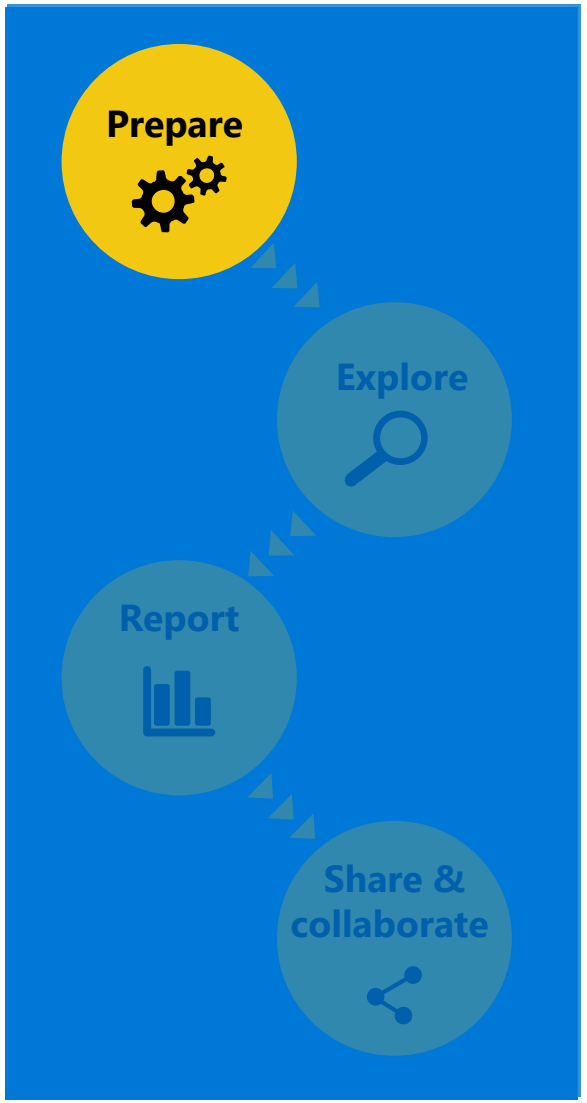
	Region	Age Group	1991	1992
1	Northland Region	All Ages	*2475*	*2438*
2	*null*	Under 20	*263*	*240*
3	Auckland Region	All Ages	*18347*	*18183*
4	*null*	Under 20	*1346*	*1323*

Unpivot “1991” and “1992” and Fill Down

	Region	Age Group	Attribute	Value
1	Northland Region	All Ages	1991	*2475*
2	Northland Region	All Ages	1992	*2438*
3	Northland Region	Under 20	1991	*263*
4	Northland Region	Under 20	1992	*240*
5	Auckland Region	All Ages	1991	*18347*
6	Auckland Region	All Ages	1992	*18183*
7	Auckland Region	Under 20	1991	*1346*
8	Auckland Region	Under 20	1992	*1323*

Shaping Data

An example (how to go from a matrix to a table?)



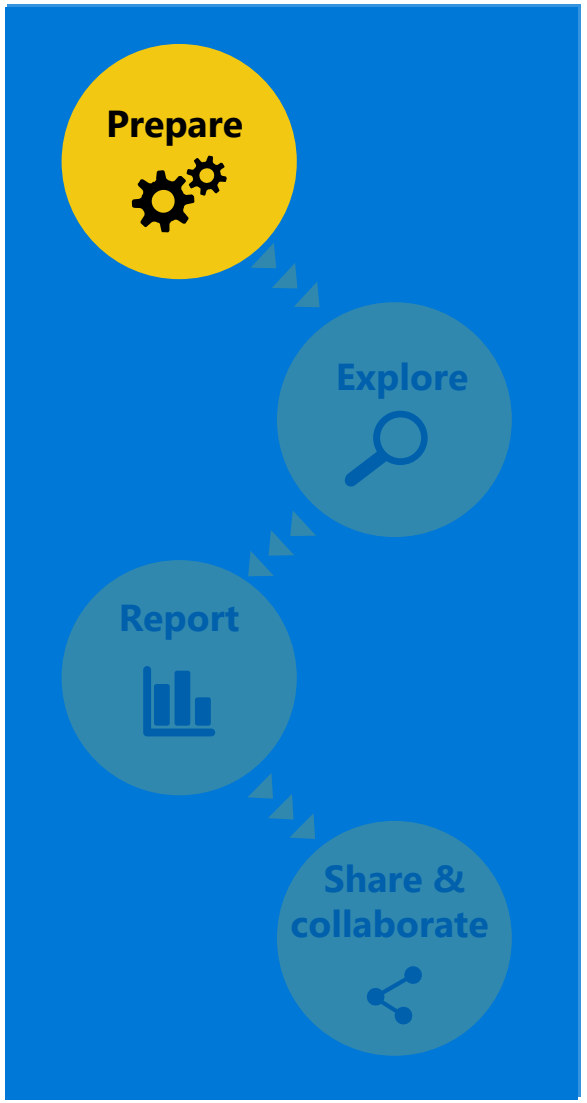
	Region	Age Group	Attribute	Value
1	Northland Region	All Ages	1991	*2475*
2	Northland Region	All Ages	1992	*2438*
3	Northland Region	Under 20	1991	*263*
4	Northland Region	Under 20	1992	*240*
5	Auckland Region	All Ages	1991	*18347*
6	Auckland Region	All Ages	1992	*18183*
7	Auckland Region	Under 20	1991	*1346*
8	Auckland Region	Under 20	1992	*1323*

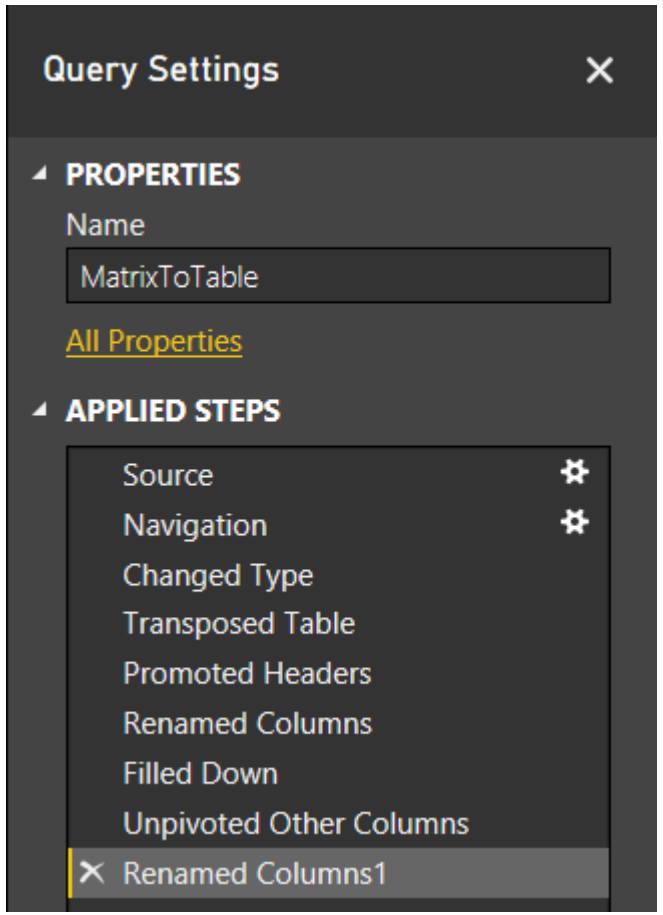
Rename Columns

	Region	Age Group	Year	Value
1	Northland Region	All Ages	1991	*2475*
2	Northland Region	All Ages	1992	*2438*
3	Northland Region	Under 20	1991	*263*
4	Northland Region	Under 20	1992	*240*
5	Auckland Region	All Ages	1991	*18347*
6	Auckland Region	All Ages	1992	*18183*
7	Auckland Region	Under 20	1991	*1346*
8	Auckland Region	Under 20	1992	*1323*

Shaping Data

An example (how to go from a matrix to a table?)



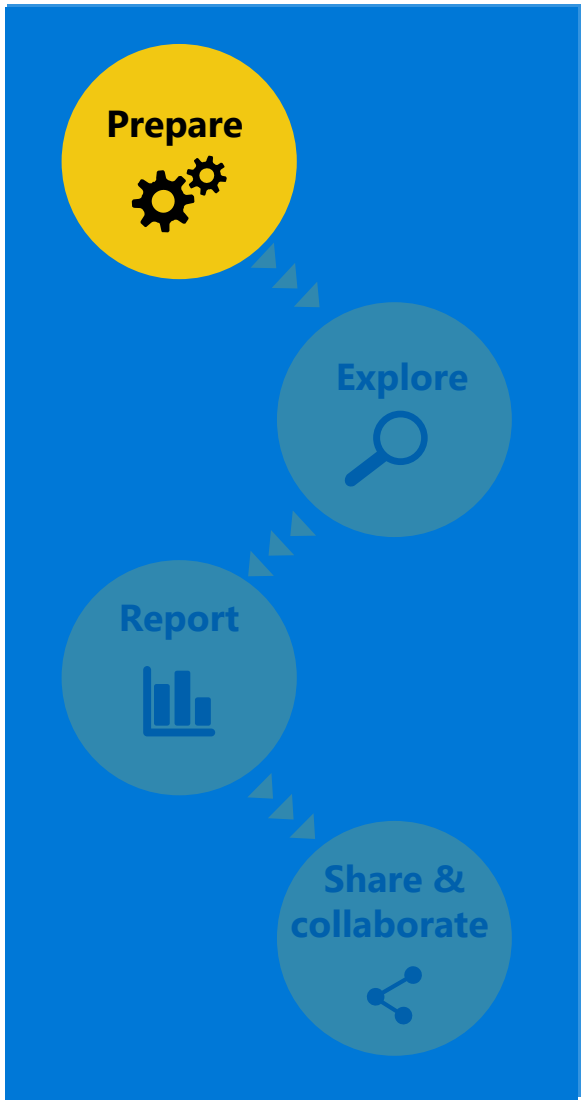


Code Equivalent

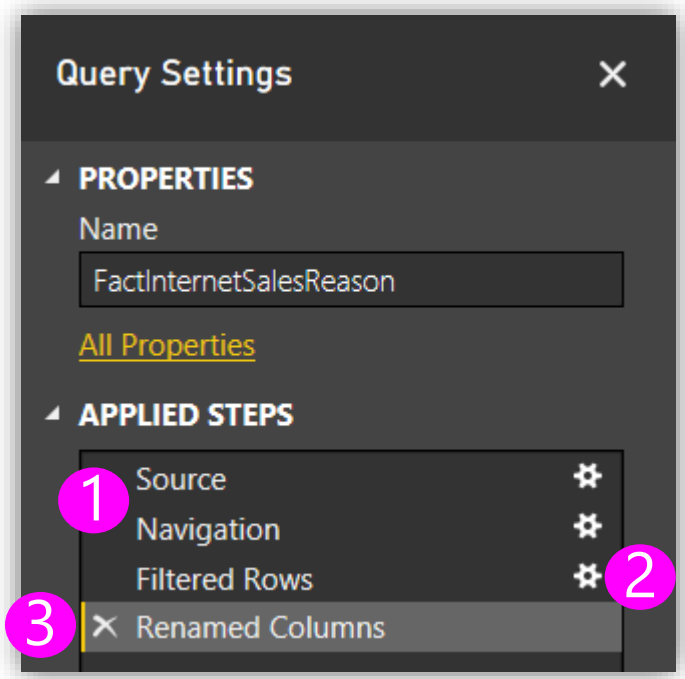
```
let
    Source = Excel.Workbook(File.Contents("C:\Users\femar\Desktop\Business Analytics with Power BI\Empowering the Data Analyst\My Samples\TransposeExample.xlsx"), null, true),
    Table6_Table = Source{[Item="Table6",Kind="Table"]}[Data],
    #"Changed Type" = Table.TransformColumnTypes(Table6_Table,{{"Column1", type any}, {"Column2", type any}, {"Column3", type any}, {"Column4", type any}, {"Column5", type any}}),
    #"Transposed Table" = Table.Transpose(#"Changed Type"),
    #"Promoted Headers" = Table.PromoteHeaders(#"Transposed Table"),
    #"Renamed Columns" = Table.RenameColumns(#"Promoted Headers",{{"Column1", "Region"}, {"Column2", "Age Group"}}),
    #"Filled Down" = Table.FillDown(#"Renamed Columns",{"Region"}),
    #"Unpivoted Other Columns" = Table.UnpivotOtherColumns(#"Filled Down", {"Region", "Age Group"}, "Attribute", "Value"),
    #"Renamed Columns1" = Table.RenameColumns(#"Unpivoted Other Columns",{{"Attribute", "Year"}})
in
    #"Renamed Columns1"
```

Shaping Data

Query Folding

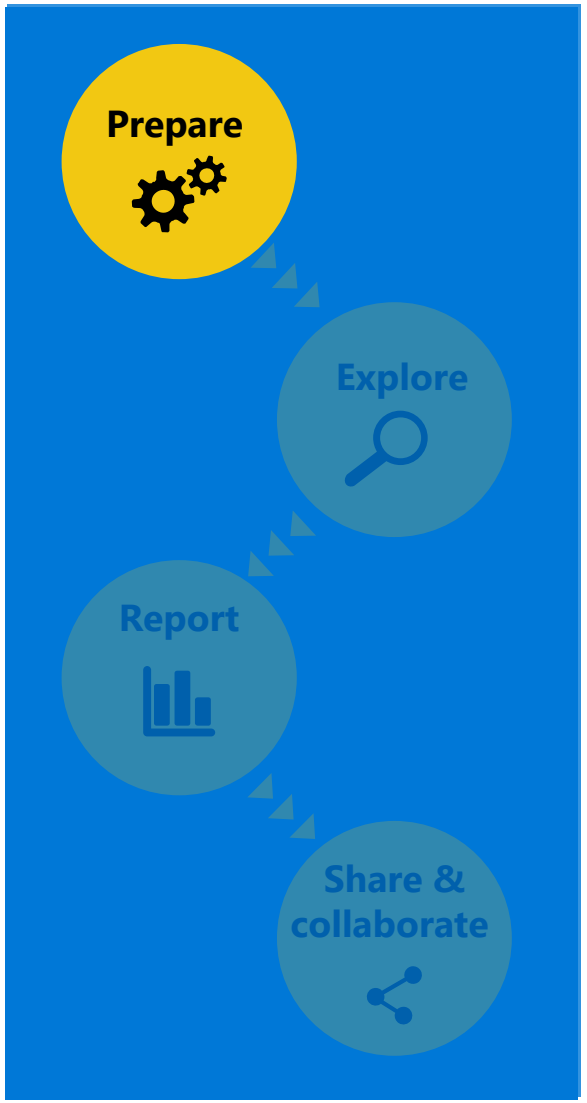


- In some cases, Power BI Desktop will be able to **push the transformation work to the data source**
- This is important from an **optimization perspective**.
- If query folding doesn't happen, all of the data has to be loaded into the tool



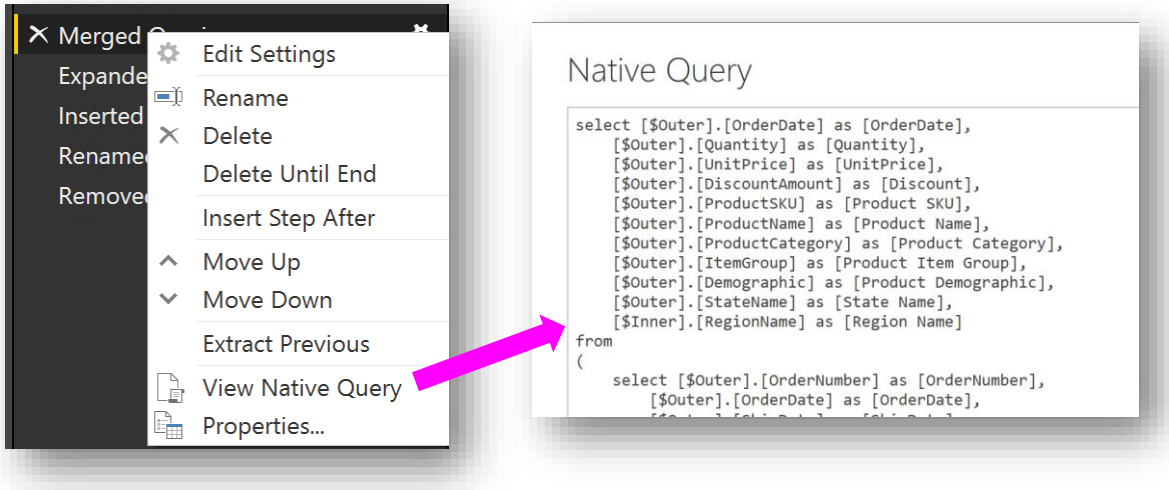

```
SELECT                                                    3
[salesordernumber] as [Sales Order Number],
[salesorderlinenumber] AS
[SalesOrderLineNumber],
[salesreasonkey] AS [SalesReasonKey]
FROM
[dbo].[factinternetsalesreason]  1
WHERE
[salesorderlinenumber] = 2  2
```

Shaping Data



Query Folding

- Supported data sources:
  - Relational Databases
  - Odata
  - Exchange
  - Active Directory



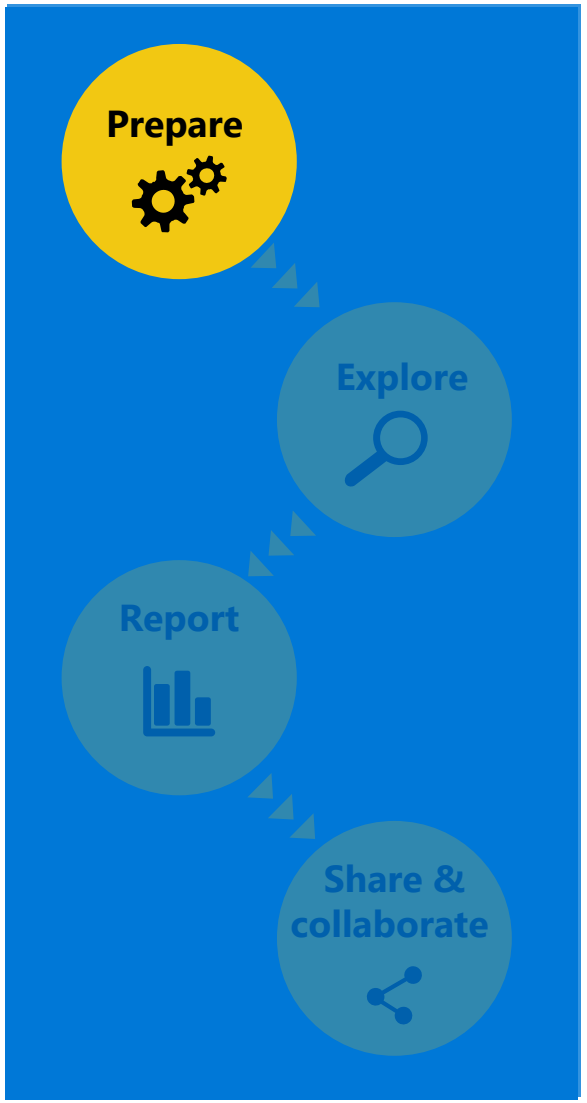
- Not all transformations are supported (for instance, custom SQL prevents folding)

- In order to see what query is being generated use the “View Native Query” option, for a given step.

- As the tool **evolves**, **more transformations and sources will support folding**

Shaping Data

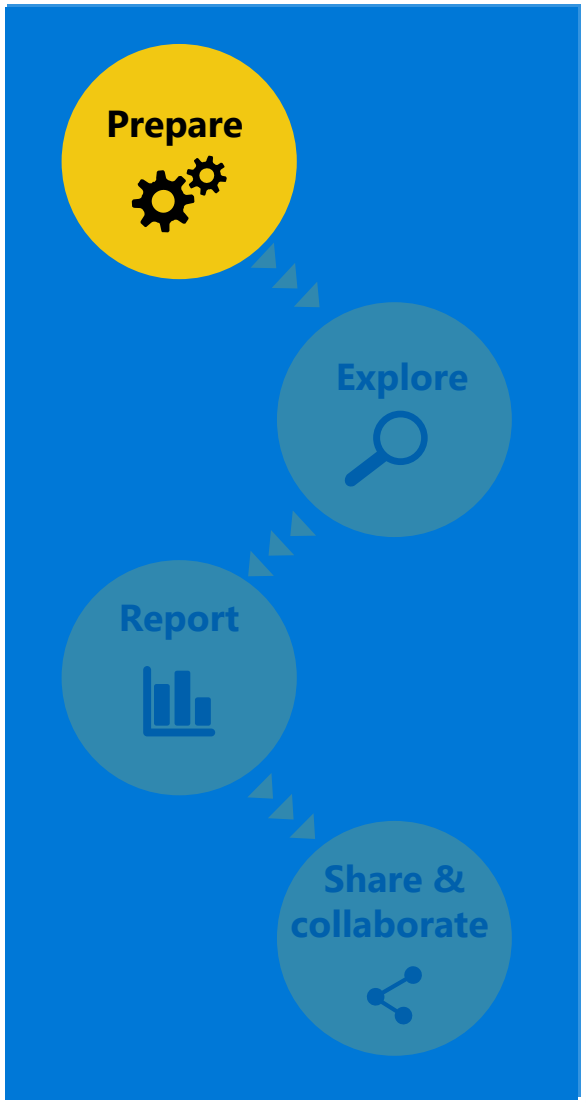
Query Formulas



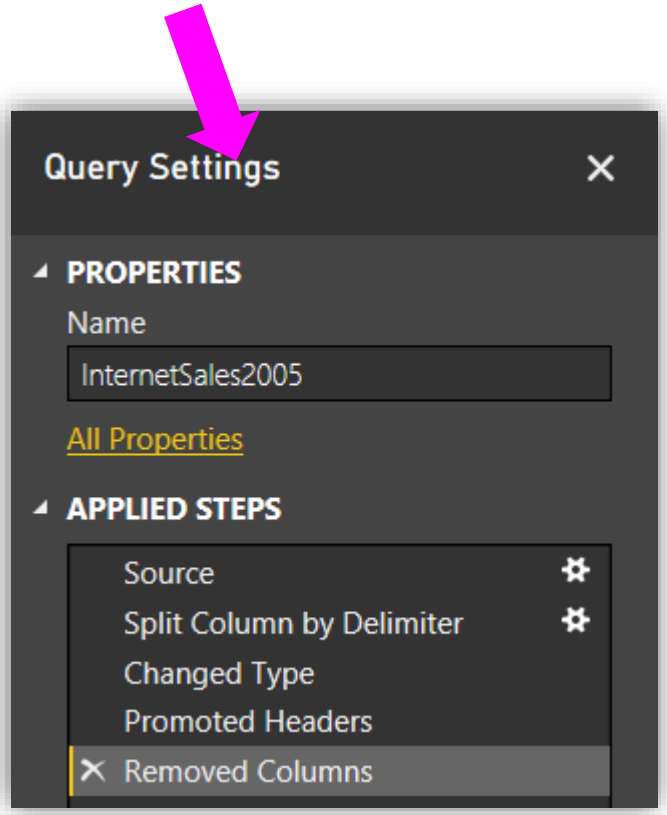
- The language was previously known as **M** or **Power Query Formula Language**
- Has a standard library with built-in functions
  - https://msdn.microsoft.com/library/Mt253322?ui=en-US&rs=en-IN&ad=IN
- Is **case-sensitive** and **strongly typed**
- Every transformation generates code automatically. Each step maps to an instruction.
- Use the **Advanced Editor** to edit the query or **Blank Query** data source to start from an empty query

Shaping Data

Query Formulas – An example



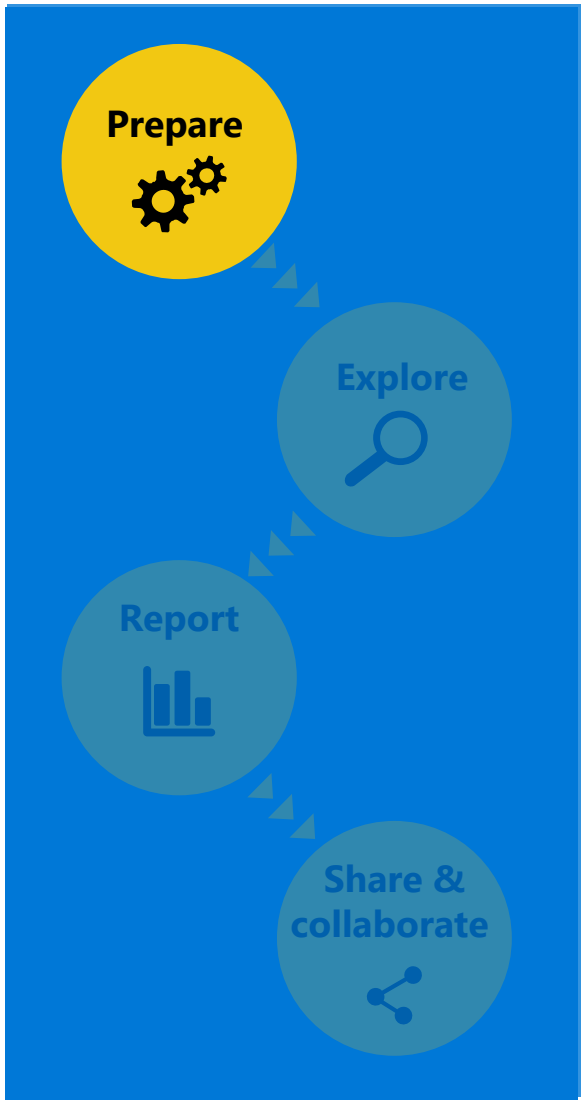

```
let
    Source = Table.FromColumns({Lines.FromBinary(File.Contents("C:\Users\femar\...\InternetSales2005.txt"),null,null,1252)}),
    #"Split Column by Delimiter" = Table.SplitColumn(Source,"Column1",Splitter.SplitTextByDelimiter(",", QuoteStyle.Csv),{"Colur
    #"Changed Type" = Table.TransformColumnTypes(#"Split Column by Delimiter",{{"Column1.1", type text}, {"Column1.2", type text
    #"Promoted Headers" = Table.PromoteHeaders(#"Changed Type"),
    #"Removed Columns" = Table.RemoveColumns(#"Promoted Headers",{"Column27", "Column28", "Column29", "Column30", "Column31", "(
in
    #"Removed Columns"
```



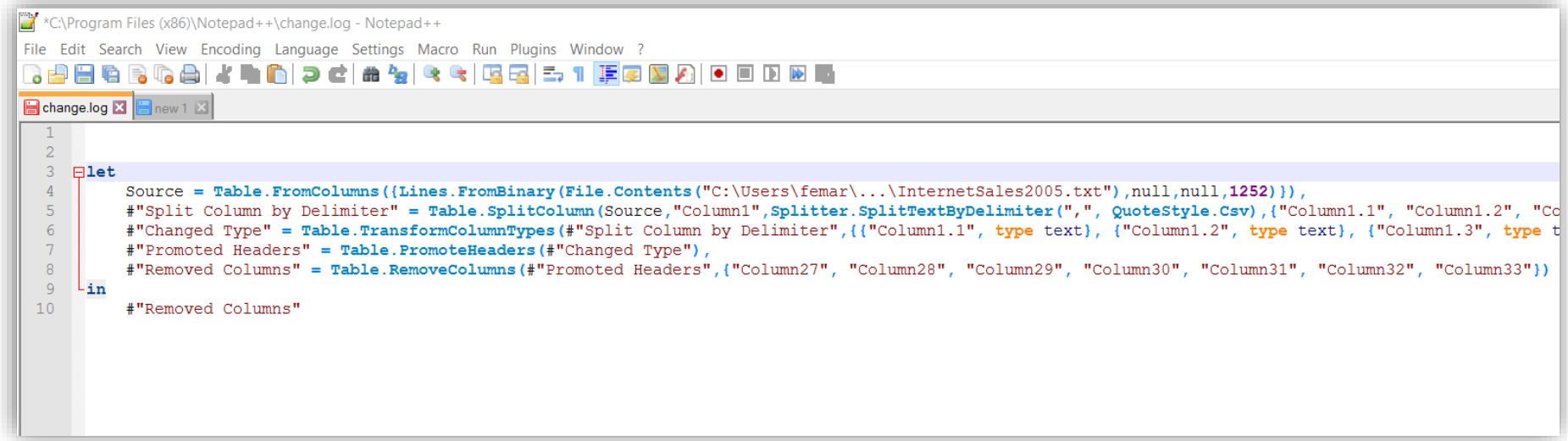
- Every query starts with a **let** instruction and ends with an **in**
- Each step name is prefixed with a **#** if it has spaces and its **name matches** exactly the one in the **Query Settings Pane** and the **order** is also the **same**
- Several built-in functions implement the interface functionality with the **Object.Function(…)** syntax like Table.RemoveColumns

Shaping Data

Query Formulas – Syntax Highlighting

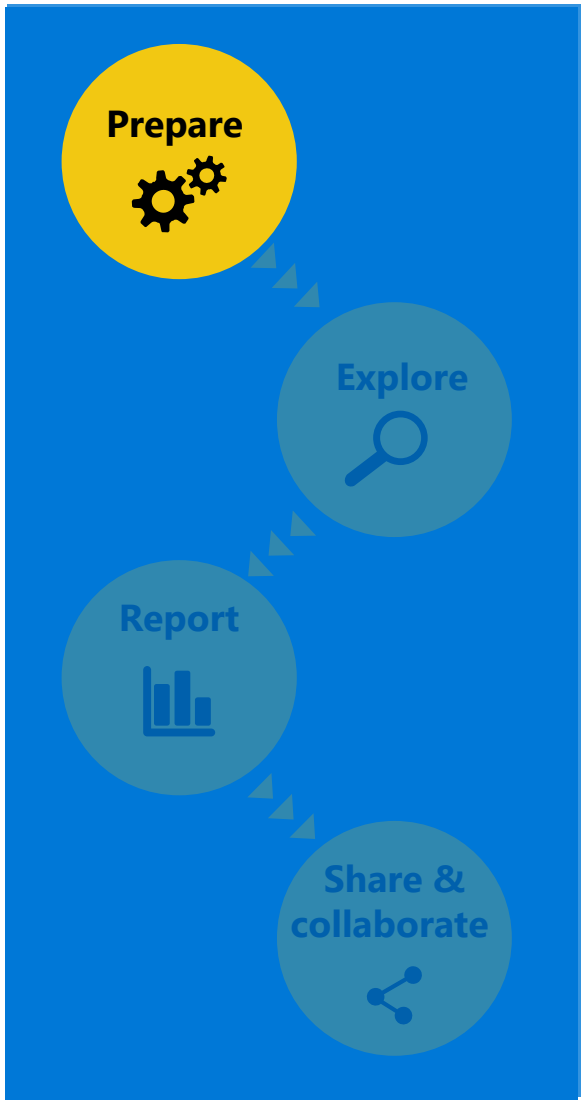


- Power BI Desktop doesn't have syntax highlighting, but there are community alternatives:
  - http://www.mattmasson.com/2014/11/notepad-language-file-for-the-power-query-formula-language-m/



Shaping Data

Query Formulas – User Functions



- Functions allow for **code reuse**. They require a particular syntax:

```
let
ParseInternetSalesFiles=(folder as text,file as text)=>
let
    Source = Table.FromColumns({Lines.FromBinary(File.Contents(folder & file),null,null,1252)}),
    #"Split Column by Delimiter" = Table.SplitColumn(Source,"Column1",Splitter.SplitTextByDelimiter(",", QuoteStyle.Csv),{"
    #"Changed Type" = Table.TransformColumnTypes(#"Split Column by Delimiter",{{"Column1.1", type text}, {"Column1.2", type
    #"Promoted Headers" = Table.PromoteHeaders(#"Changed Type"),
    #"Removed Columns" = Table.RemoveColumns(#"Promoted Headers",{"Column27", "Column28", "Column29", "Column30", "Column31
in
    #"Removed Columns"
in
    ParseInternetSalesFiles
```

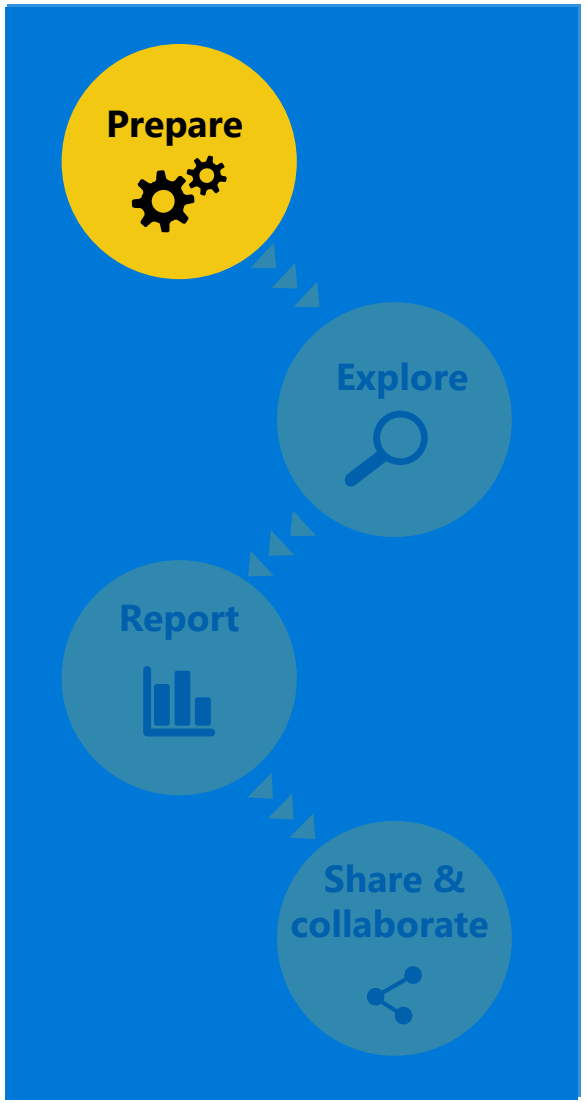
- The procedure has to be enclosed in a new **let** and **in** block
- **Parameters** are declared in the function definition and then they can be used to make the procedure generic:

```
let
ParseInternetSalesFiles=(folder as text,file as text)=>
```

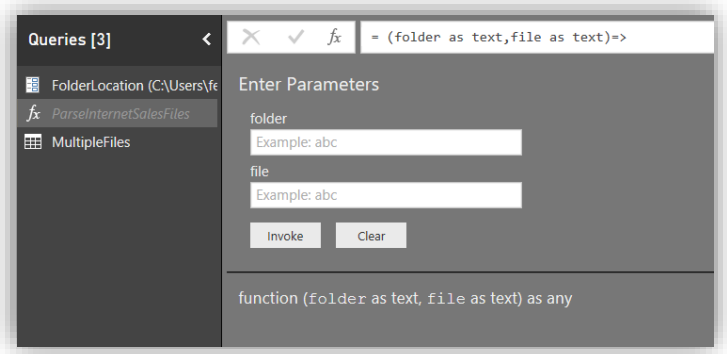
```
ry(File.Contents(folder & file),nul
Column(Source,"Column1",Splitter.Sp
```

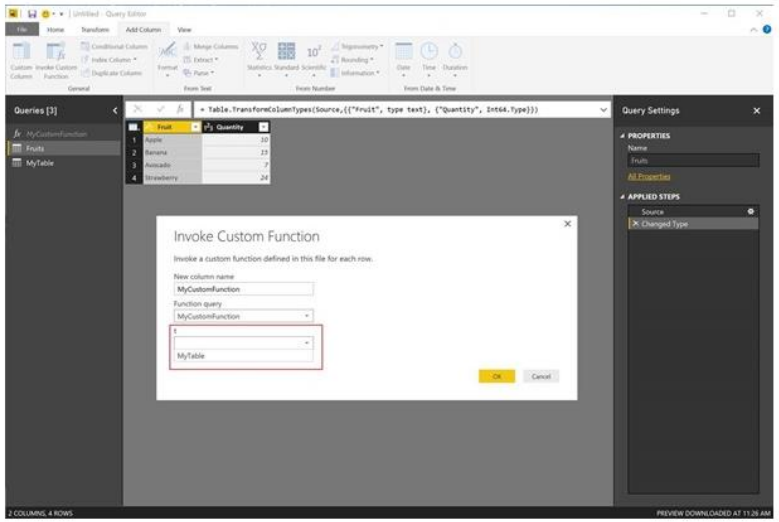
Shaping Data

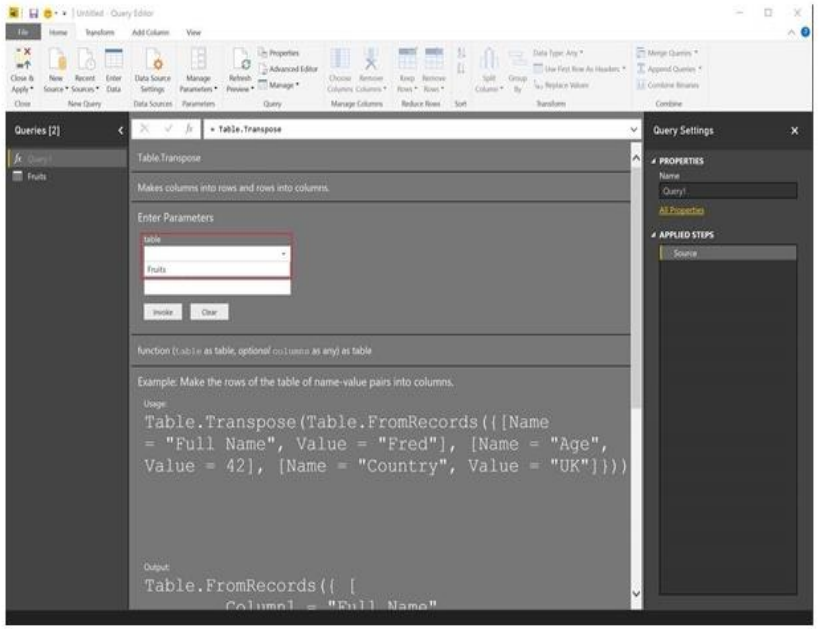
Query Formulas – User Functions



- You can **invoke** them via the Query Editor
- Notice the **fx** icon in the Left Pane that indicates a user function
- Supports parameters of **type "Table"**

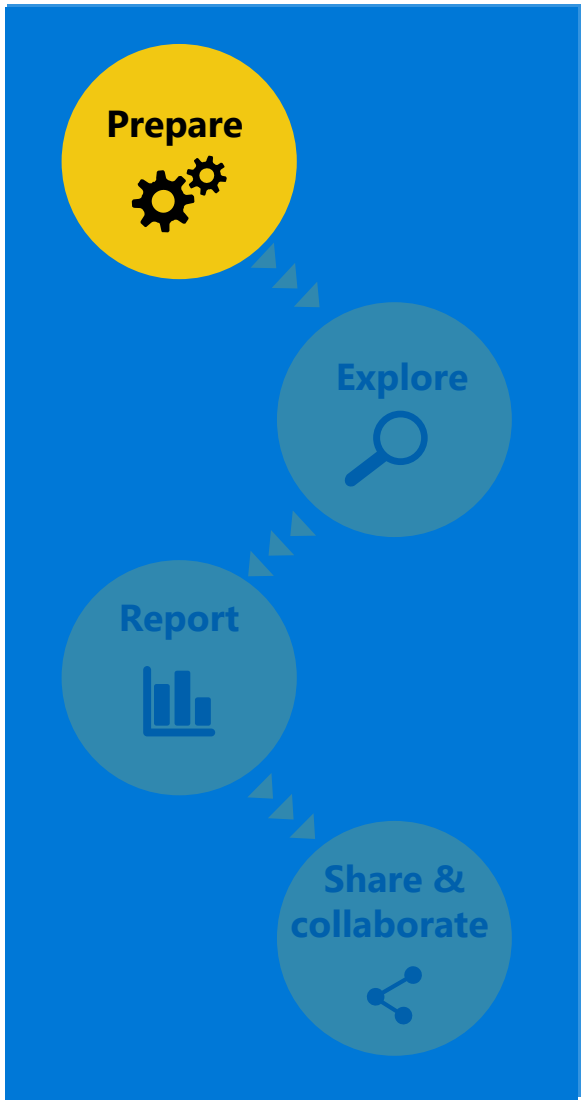




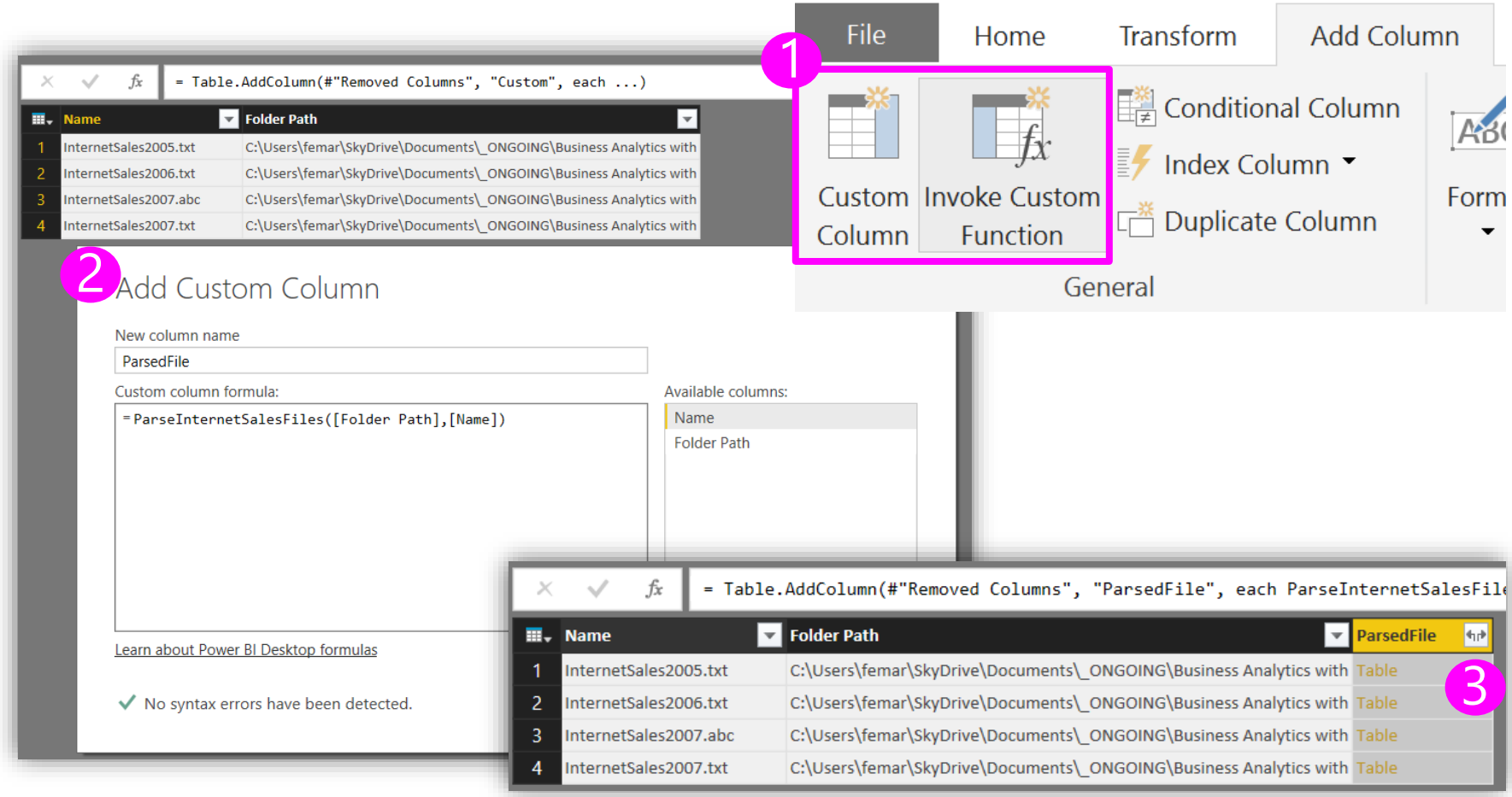


Shaping Data

Query Formulas – User Functions

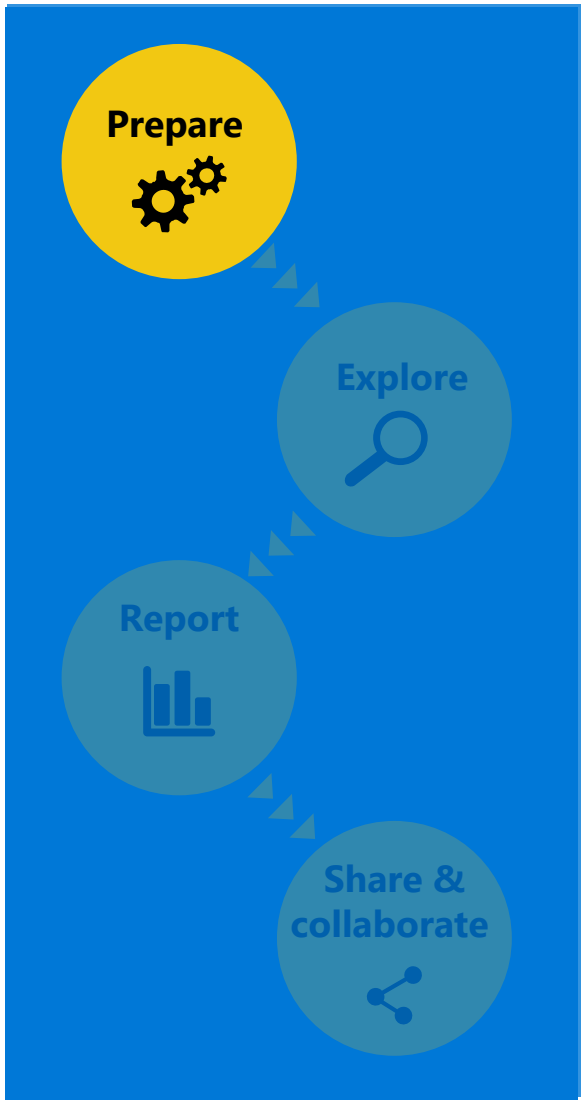


- You can also invoke them with the **Add Custom Column** task or with **Invoke Custom Function**

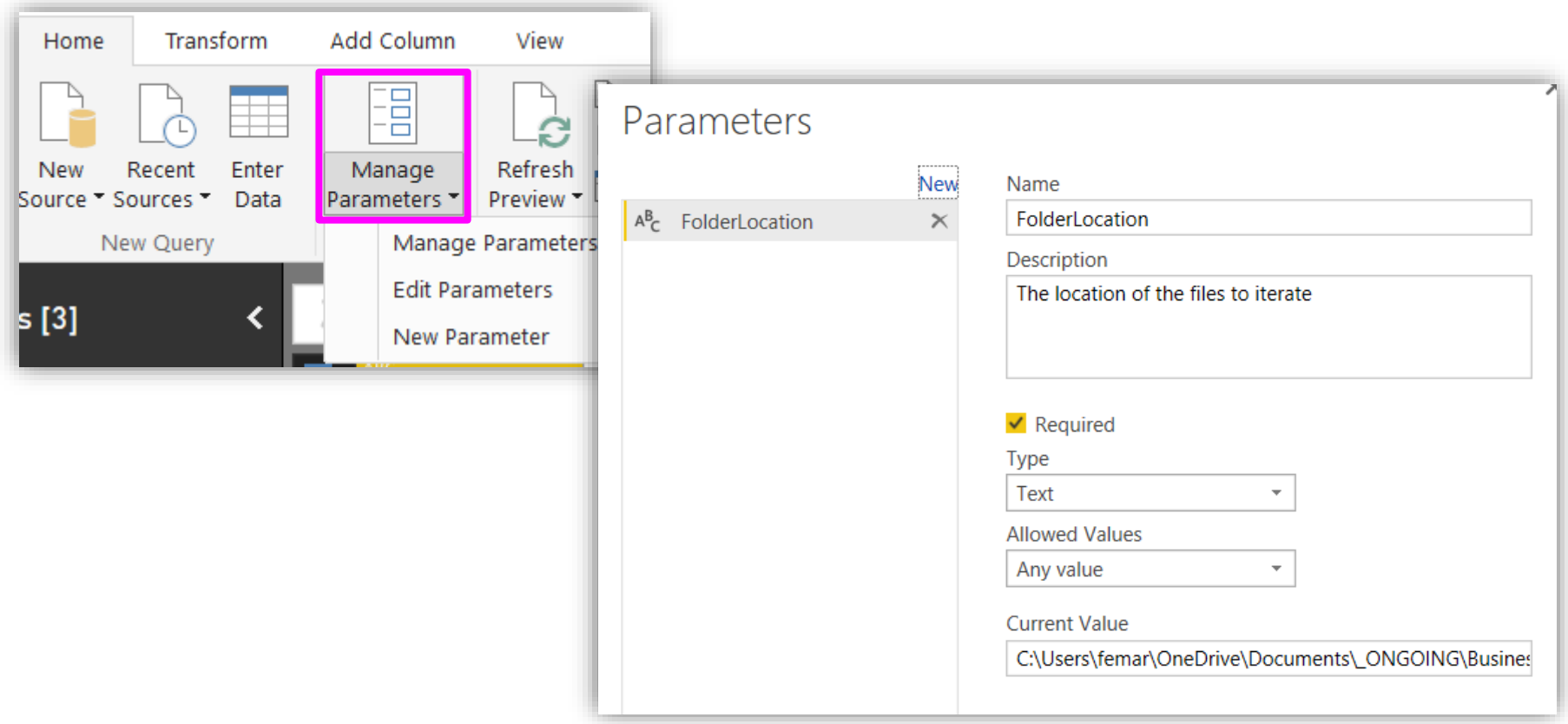


Shaping Data

Query Formulas – Parameters

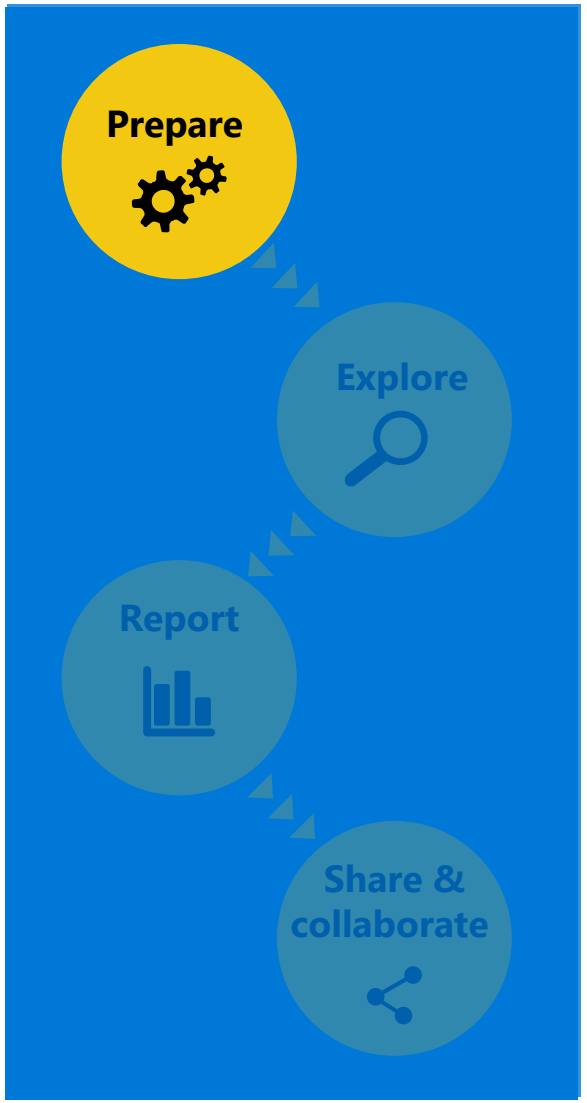


- Queries and other objects can use parameters to get a **dynamic behavior**



Shaping Data

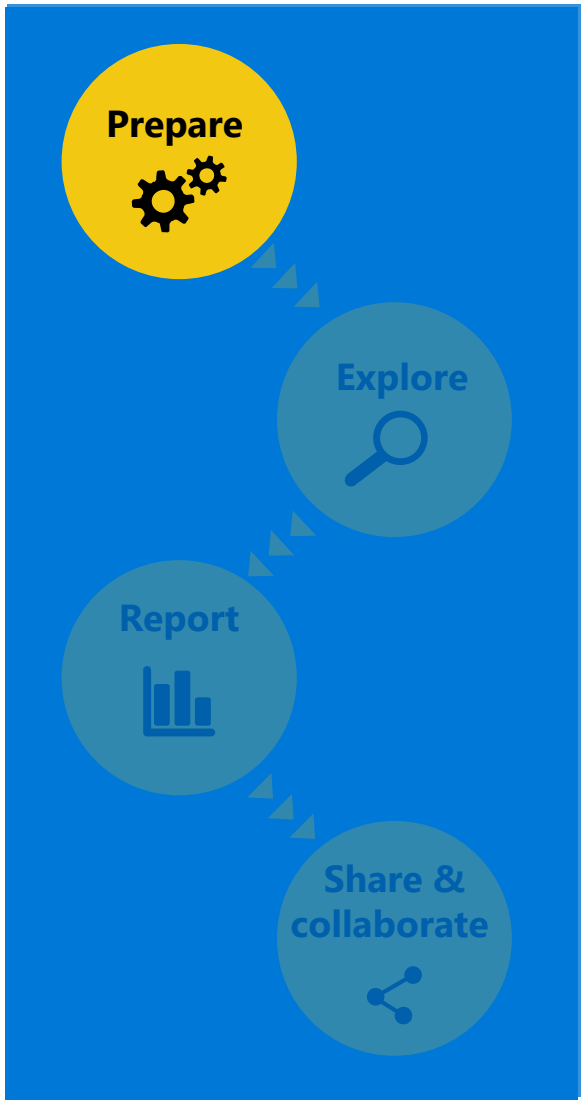
Query Formulas – Parameters



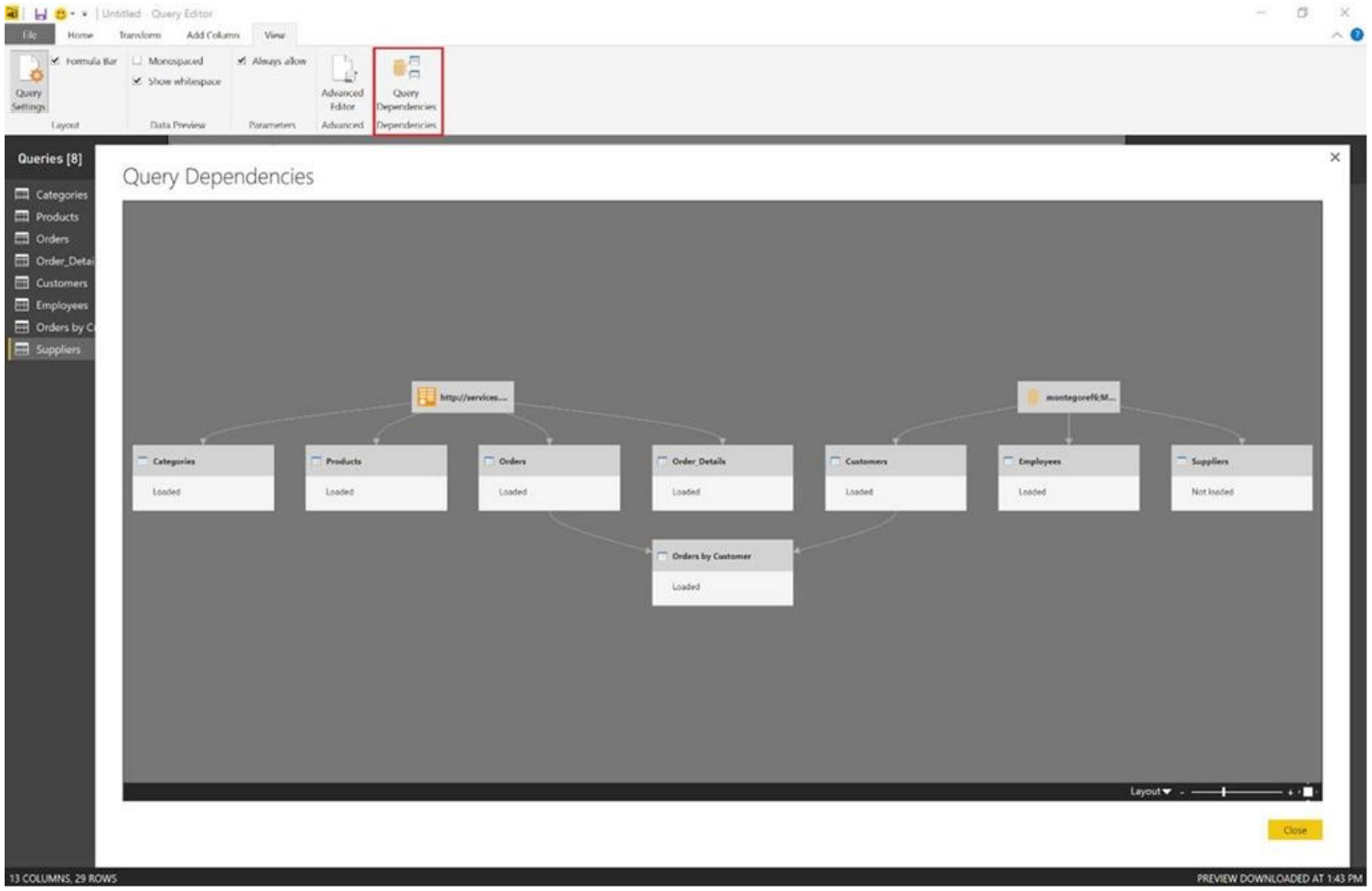
- Each parameter generates **a query**
- **Can be referenced from other queries** and loaded to the data model for use in calculations
- **Support** for parameters is available throughout the tool

Shaping Data

Query Dependencies view



- View dependencies across all queries and data sources



Demonstration: Functions and Iterating Files

Lab 1 Exercise 1: Shaping Data