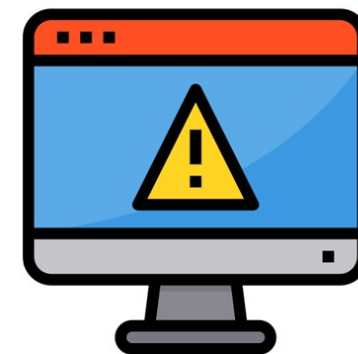




# HAVENS

CONSULTING



## Performance, Optimization, and Error Handling in Power Query



# Presenter Introduction

- **Reid Havens**

- Founder | BI Evangelist | Consultant
- Microsoft MVP
- PBI User Group Co-Organizer – Redmond, WA
- Nickname: “The Viz Wiz”
- Specializes in teaching, consulting, and design

<https://www.havensconsulting.net>

[reid@havensconsulting.net](mailto:reid@havensconsulting.net)

**HAVENS**  
CONSULTING





# Performance, Optimization, and Error Handling in Power Query

Reducing Query Data

Optimizing Query Performance

Leveraging Query Folding

Handling Conversion Errors



# Performance, Optimization, and Error Handling in Power Query

Reducing Query Data

Optimizing Query Performance

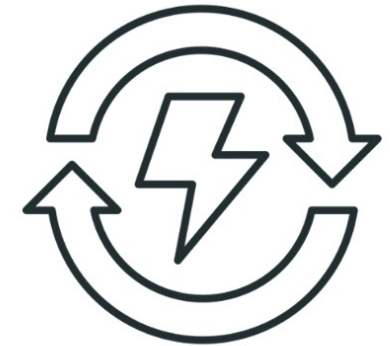
Leveraging Query Folding

Handling Conversion Errors

# Reducing Query Data

## Common Transformations

- **Common reduction transformations are:**
  - Column modification
  - Reducing rows



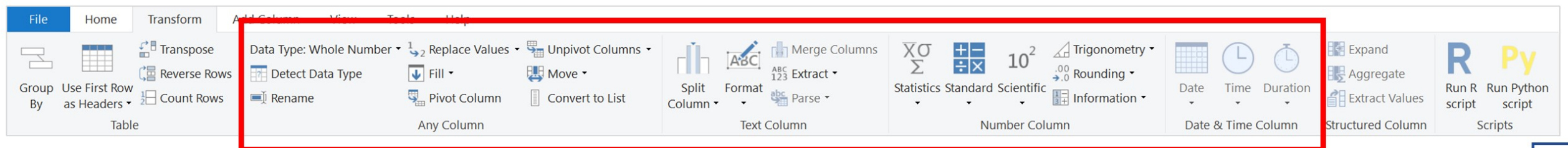
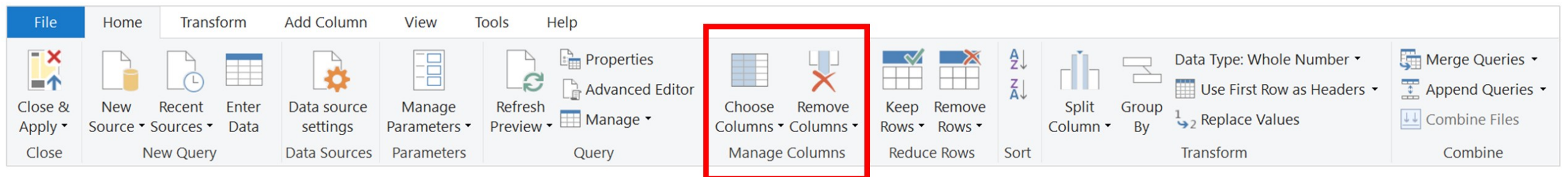
# Reducing Query Data

## Common Transformations ► Column Modification

- **Types of column modifications:**

- *Choosing or removing* columns
- *Modifying* columns

Most column modifications can be added without any coding required



# Reducing Query Data

## Common Transformations ► Reducing Rows

- Types of row reduction:

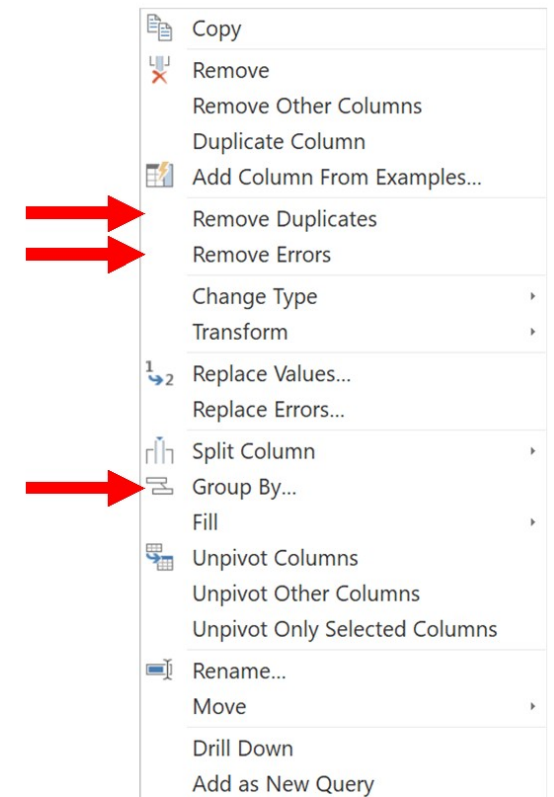
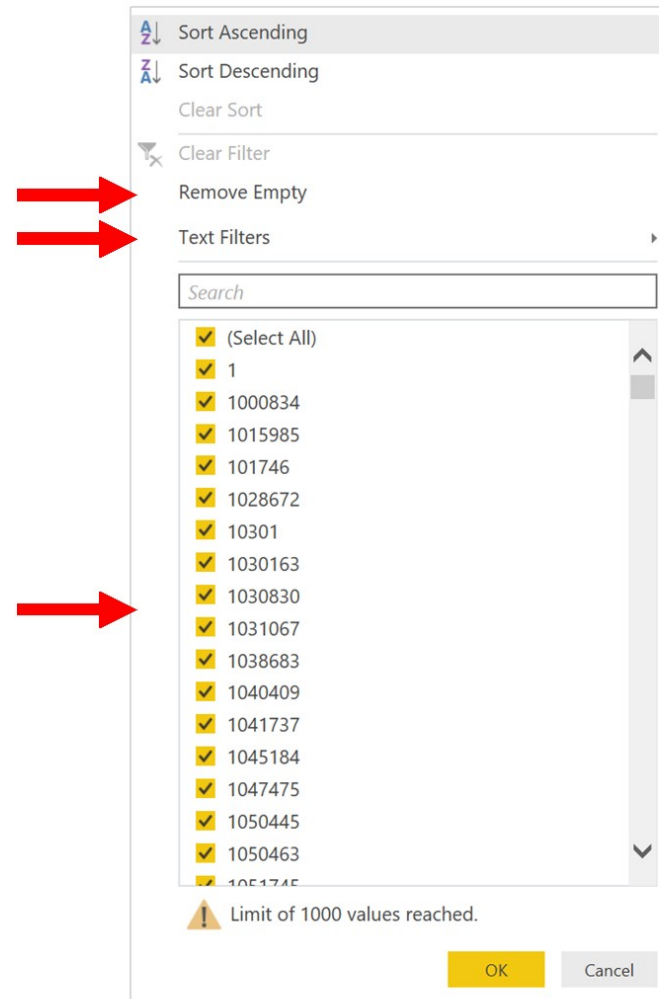
- Filtering rows:**

- Filtering by *category* or *date*<sup>1</sup>
- Removing *empty rows*
- Removing *duplicates*
- Removing *errors*

- Group by column(s)**<sup>2</sup>

**1:** Depending on data type

**2:** The **Group By** function aggregates rows up to a level higher than the original granularity of the table





# Demonstration





# Recommended Practices

- Query data reduction practices:
  - **Select columns:**
    - Select only columns that meet the current report requirements
  - **Filter rows:**
    - Filter by date (history)
    - Filter by category (subset of source data)
  - **Reduce granularity:**
    - Remove row level details outside the scope of report requirements
  - **Change data type:**
    - Set columns to appropriate data types



# References

- Power Query Docs
  - <https://docs.microsoft.com/en-us/power-query/>
- Common Power Query Tasks
  - <https://docs.microsoft.com/en-us/power-bi/desktop-common-query-tasks>
- HC – Power Query Videos Playlist
  - <https://www.youtube.com/playlist?list=PLzN99cpDw6oBuidLOD20RVv83RusrBQ3o>



Questions?



# Performance, Optimization, and Error Handling in Power Query

Reducing Query Data

Optimizing Query Performance

Leveraging Query Folding

Handling Conversion Errors



# Demonstration



# Recommended Practices

- Query optimization practices:
  - **Row and column selection first:**
    - Choose columns and filter rows before transformations to reduce the amount of information processed by the mashup engine
  - **Avoid duplicate applied steps:**
    - Try to avoid having multiple applied steps for the same types of transformations. Common transformations where this occurs are: change type, rename columns, or choose columns.
  - **Remove duplicates:**
    - When remove duplicates from a column – avoid using the Group By command, unless an aggregation is required. Otherwise use the remove duplicates command
  - **Add columns using left joins:**
    - Left joins with other tables typically calculate faster than adding a new column using the conditional column command



# Recommended Practices

- Query optimization practices:
  - **Use list commands for column calculations:**
    - Whenever a calculation against a column is needed, such as min/max or sort commands. It is better to treat the column as a list using Statistical commands or the Group By and aggregate commands.



# References

- The Biccountant – Speed and Performance Aspects in Power Query
  - <https://www.thebiccountant.com/speedperformance-aspects/>
- HC – Power Query Videos Playlist
  - <https://www.youtube.com/playlist?list=PLzN99cpDw6oBuidLOD20RVv83RusrBQ3o>





Questions?



# Performance, Optimization, and Error Handling in Power Query

Reducing Query Data

Optimizing Query Performance

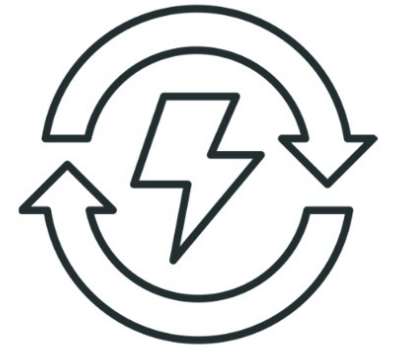
Leveraging Query Folding

Handling Conversion Errors

# Query Folding

## Introducing Query Folding

- Query folding is the ability for a Power Query query to generate a **single query statement** to retrieve and transform data *at the source*
  - Query folding is the **most efficient path** to connect a Power BI model table to its *underlying data source*
- Query folding **may occur** for an *entire Power Query query*, or for a *subset of its steps*
  - When query folding cannot be achieved for an applied step – the transformation is imported and processed by Power Query<sup>1</sup>

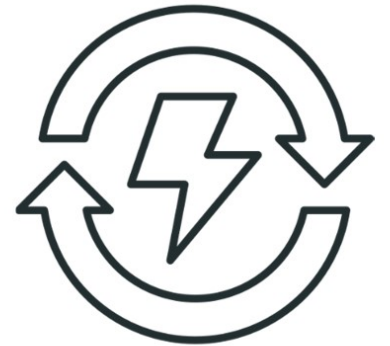


**1:** For large datasets this can be very resource intensive and slow

# Query Folding

## Compatible Data Sources

- Most data sources that have the **concept of a query language** *support query folding* – these can include:
  - Relational databases
  - OData feeds (including SharePoint lists), Exchange, and Active Directory.
- Data sources that are **not supported**:
  - Flat files
  - Blob
  - Web

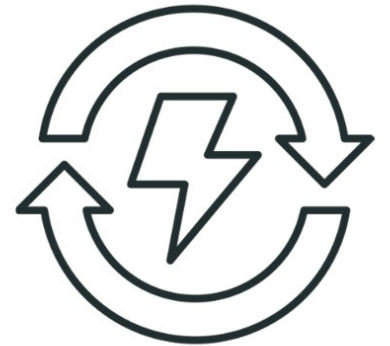


# Query Folding

## Compatible Applied Steps

### Applied steps that **can** be query folded:

- Removing columns
- Renaming columns
- Changing a column data type
- Filtering rows with static values or Power Query parameters
- Grouping and summarizing
- Expanding record columns
- Non-fuzzy merging of fold-able queries from the same source
- Appending fold-able queries from the same source
- Adding custom columns with simple logic
- Pivoting and unpivoting



# Query Folding

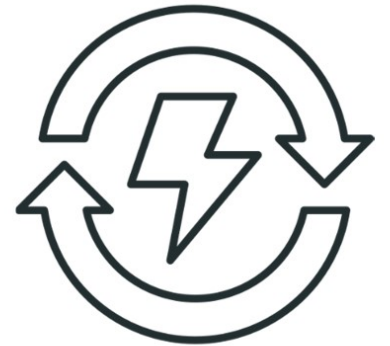
## Incompatible Applied Steps

### Applied steps that **cannot** be query folded:

- Using a custom SQL statement<sup>1</sup>
- Merging queries based on different sources
- Appending queries based on different sources
- Adding custom columns with complex logic.
- Adding index columns
- Using the Buffer command
- Using any command to remove errors
- Transformations that use a List function<sup>2</sup>

**1:** Compared to connecting to a **native SQL query** that support folding

**2:** Unless used as an aggregation function inside a **Group By** command

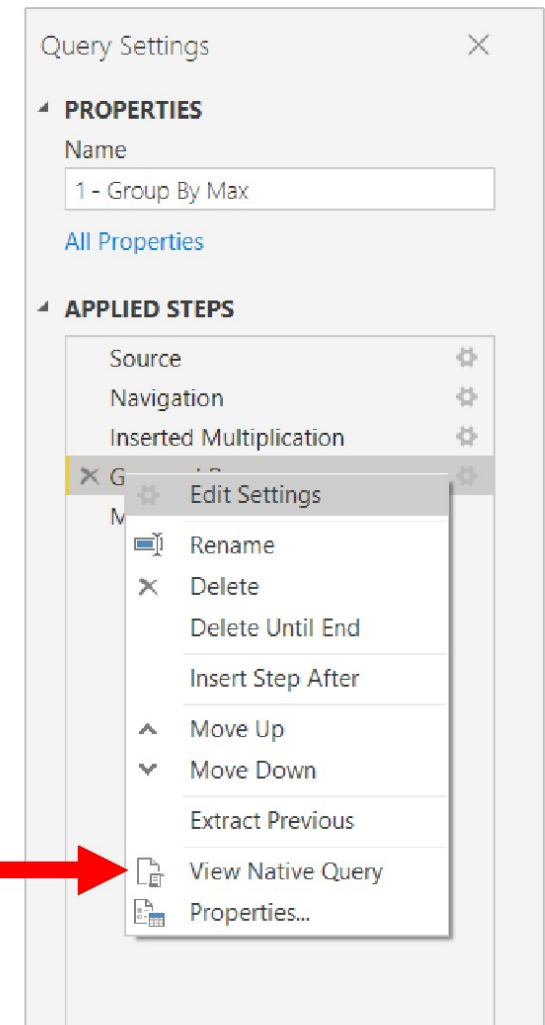


# Query Folding

## Identifying Query Folding

- In the Power Query Editor window, it is possible to determine if any **applied step** *can be folded*
- In the **Query Settings** pane, when you right-click an applied step, if the **View Native Query** option is enabled (not greyed out) – then the query can be folded

Foldable →



Native query  
sent to data  
source



Native Query

```
select max([rows].[Multiplication]) as [Max Sales Amount]
from
(
  select case
    when [__].[OrderQty] is null and [__].[UnitPrice] is null
    then null
    else (case
      when [__].[OrderQty] is null
      then 1
      else [__].[OrderQty]
    end) * (case
      when [__].[UnitPrice] is null
      then 1
      else [__].[UnitPrice]
    end)
    end as [Multiplication]
  from
  (
    select [OrderQty],
      [UnitPrice]
    from [SalesLT].[SalesOrderDetail] as [$Table]
  ) as [__]
) as [rows]
```

OK





# Demonstration



# Recommended Practices

- Delegate as much processing to the data source as possible
  - When all steps of a Power Query query cannot be folded, discover the step that prevents query folding
  - When possible, move subsequent steps earlier in sequence so they may be factored into the query folding<sup>1</sup>
- Prepare and transformation data in the source
  - If certain Power Query query steps cannot be folded, it may be possible to apply the transformations in the data source
  - This could be achieved by writing a database view that logically transforms source data, or by physically preparing and materializing data

**1:** Note that the Power Query mashup engine may be smart enough to reorder your query steps when it generates the source query



# References

- Importance of Power Query Folding
  - <https://docs.microsoft.com/en-us/power-bi/guidance/power-query-folding>
- Handling Query Folding
  - <https://docs.microsoft.com/en-us/power-query/handlingqueryfolding>



Questions?



# Performance, Optimization, and Error Handling in Power Query

Reducing Query Data

Optimizing Query Performance

Leveraging Query Folding

Handling Conversion Errors



# Demonstration



# Recommended Practices

- To build a reliable data model in Power BI, you need to account for and handle errors carefully
- Consider the pros and cons of *replacing values*, *replacing errors*, or letting the refresh *fail upon error*
  - If you use the replace errors command consider outputting an errors table in the report for review and analysis when errors occur



# References

- Exception Report: Error Rows
  - <https://radacad.com/exception-reporting-in-power-bi-catch-the-error-rows-in-power-query>
- Errors – Power Query
  - <https://docs.microsoft.com/en-us/powerquery-m/errors>
- Error Handling for Power Query Connectors
  - <https://docs.microsoft.com/en-us/power-query/handlingerrors>





Questions?

# Online Resources



Presentation PDF

- <http://www.havensconsulting.net/speaking-events>



Consulting Services

- <http://www.havensconsulting.net/consulting-services>



Files & Templates

- <http://www.havensconsulting.net/files-and-templates>



30% Discount Code

- PBIDAYS\_AUG\_2020



YouTube Channel

- <https://www.youtube.com/c/HavensConsulting>



LinkedIn Page

- <https://www.linkedin.com/in/reidhavens>



# Company Website



# Blog Signup Raffle

