Microsoft

Microsoft Fabric
COMMUNITY CONFERENCE

Low friction data transformation and data movement using Fabric Dataflows

Benni De Jagere

Microsoft

Benni De Jagere

Senior Program Manager | Fabric Customer Advisory Team (FabricCAT)

Fabric CAT

.be Member

@BenniDeJagere

/bennidejagere

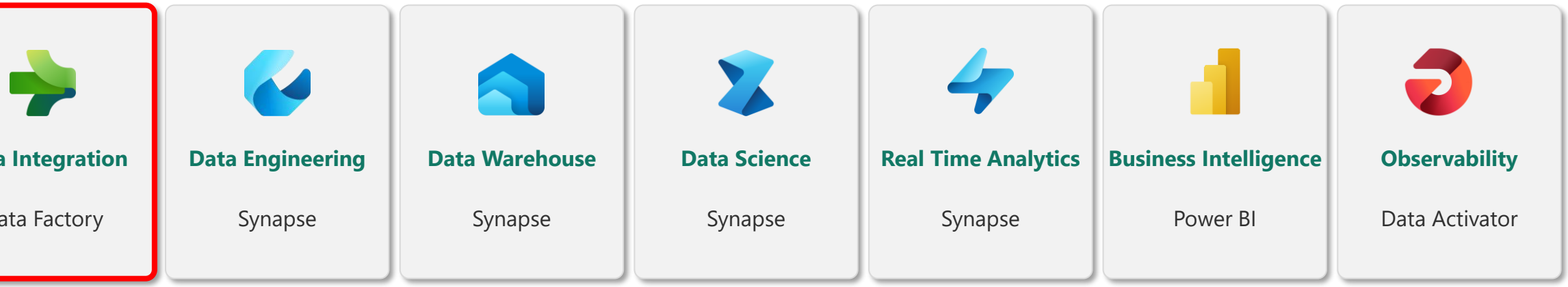
/bennidejagere

/bennidejagere

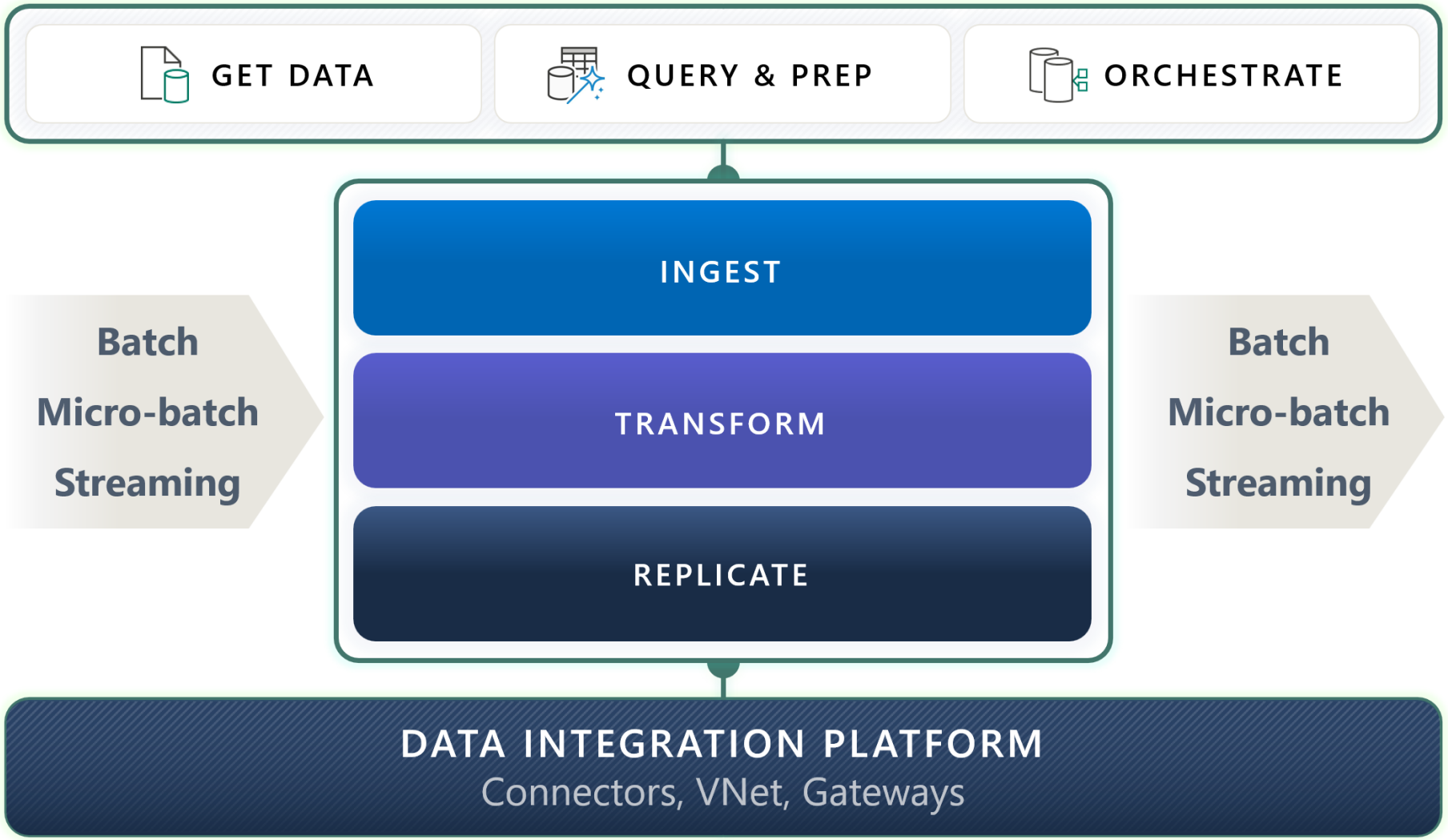
#SayNoToPieCharts

Microsoft Fabric does it all—in a unified solution

An end-to-end analytics platform that brings together all the data and analytics tools that organizations need to go from the data lake to the business user



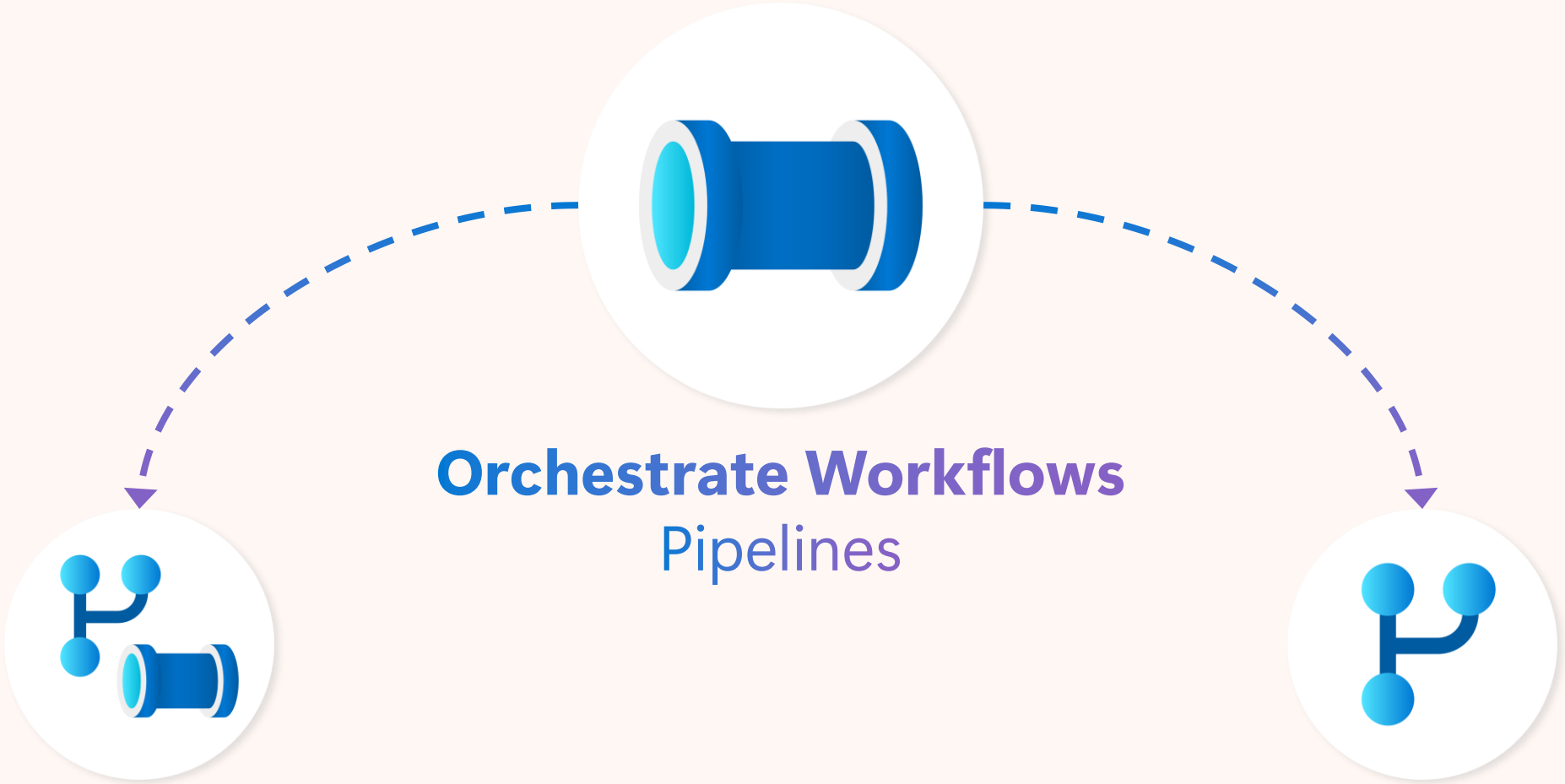
Fabric Data Factory



What is Data Factory in Microsoft Fabric?

No-code / low-code data integration experience with:

- Scale and power of Azure Data Factory
- Ease-of-use of Power Query
- Intelligence of Copilot


Orchestrate Workflows
Pipelines
Ingest Data
Dataflows / Pipelines
Transform Data
Dataflows

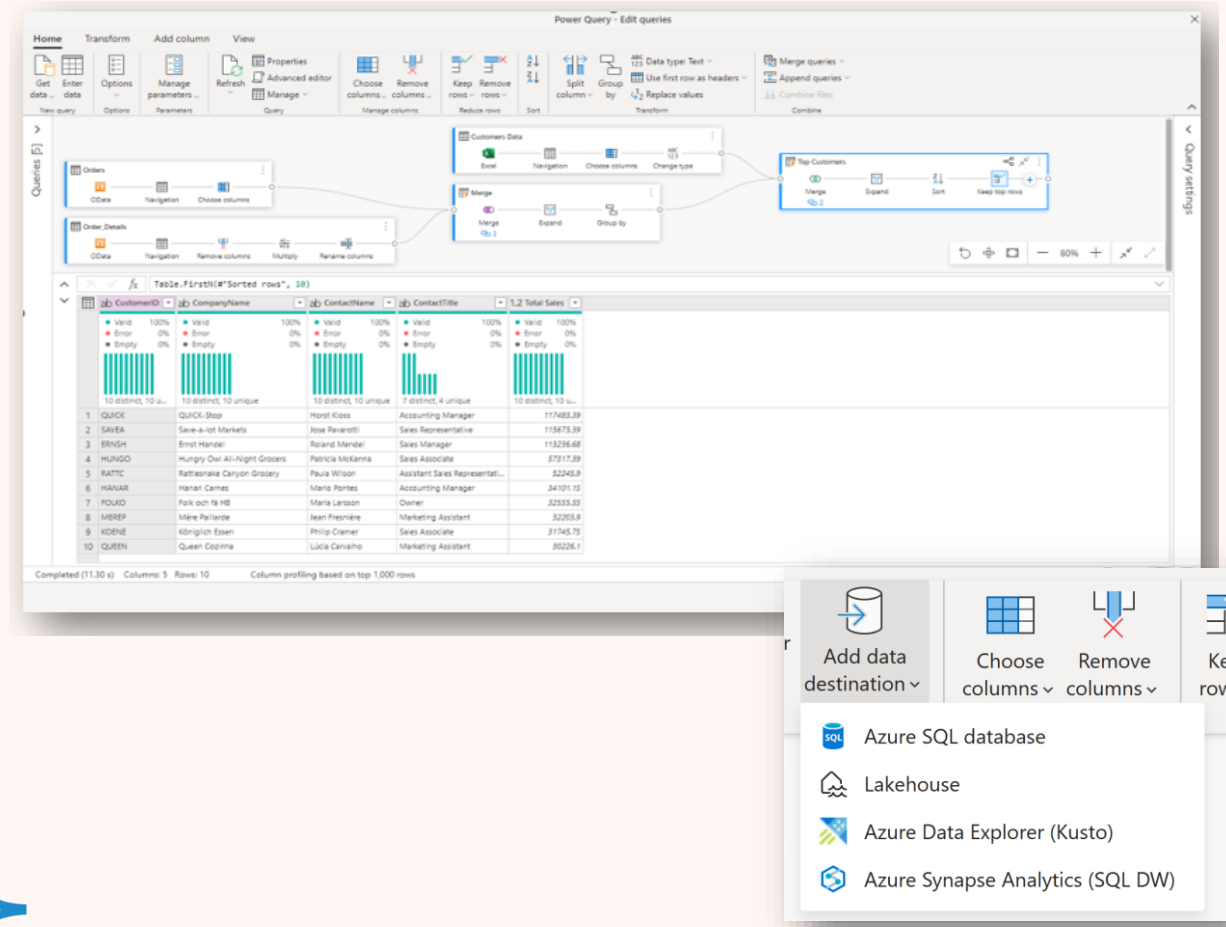
What are Dataflows Gen 2?

Self-serve Data preparation

Familiar experience with **Power Query** + **M Language**

File, relational, multi-dimensional, SaaS data experiences

Embedded into Microsoft citizen experiences – Excel, Power BI, Power Apps & Automate, Dynamics 365



Next Generation of Power BI Dataflows

Shorter authoring experience

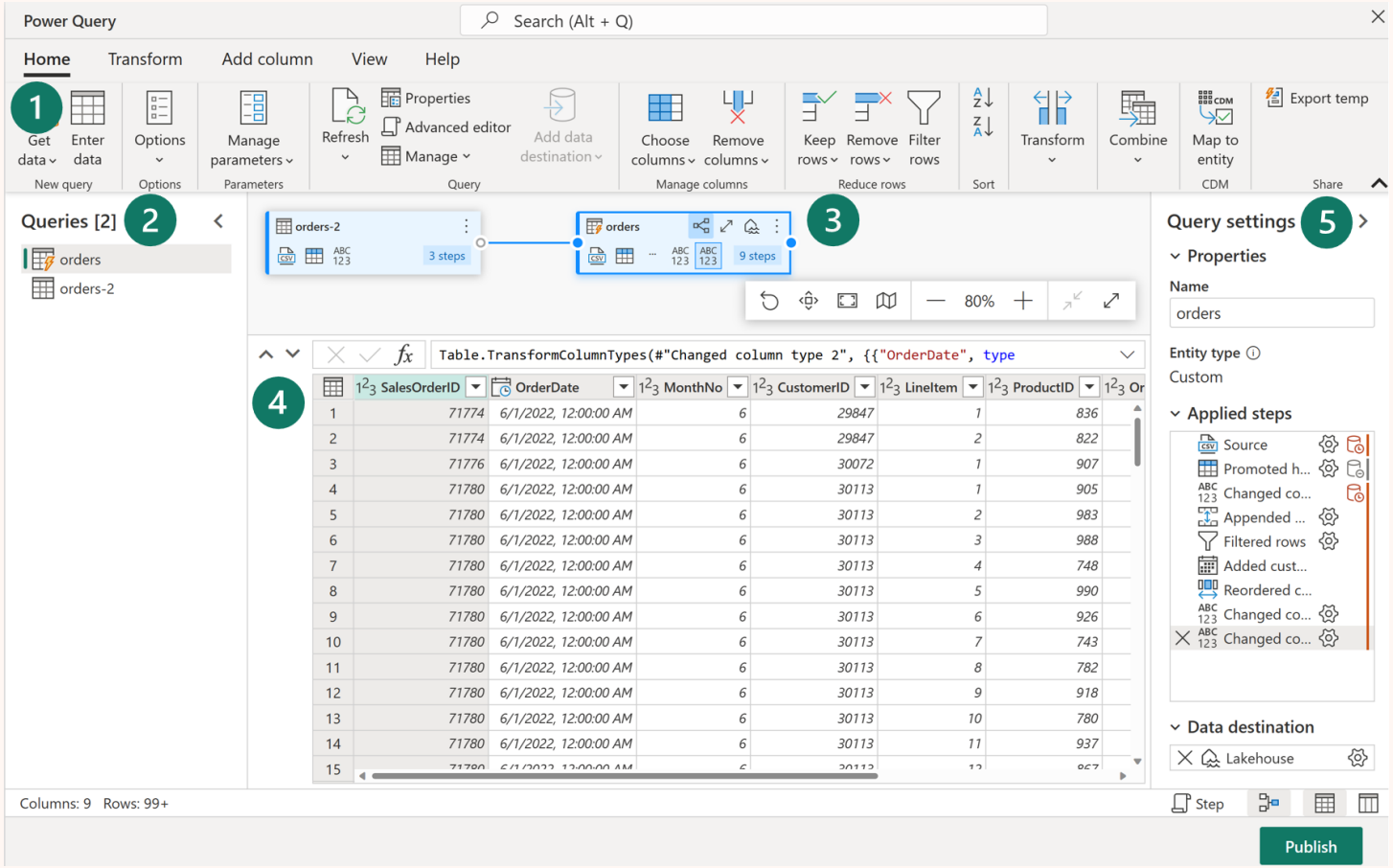
New output **Data Destinations**

New **refresh history** and **monitoring**

Integration with Data pipelines

Components of a Dataflow

1. Get Data (connectors)
2. Queries Pane
3. Diagram View
4. Data Preview pane
5. Query settings pane



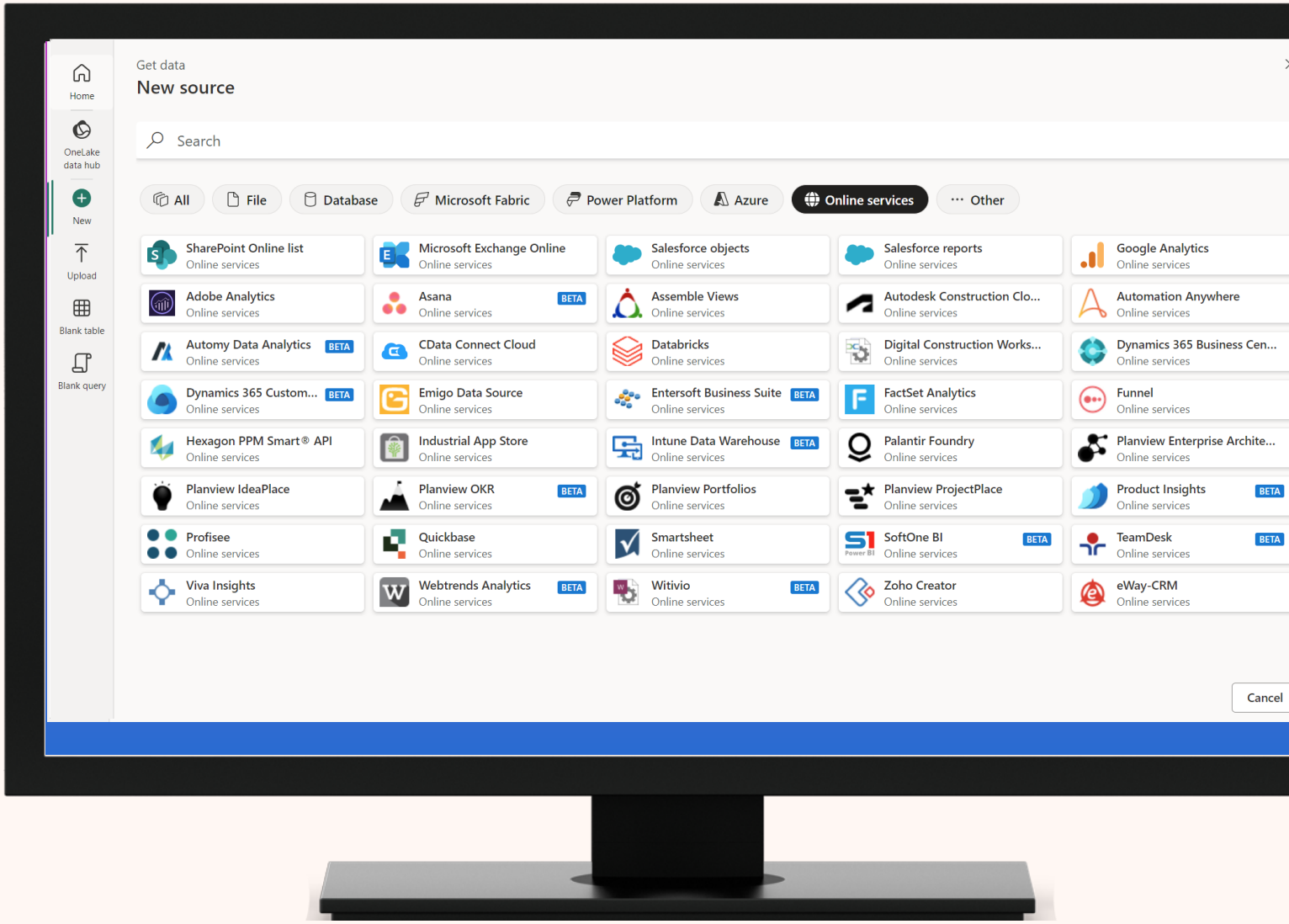
Dataflows Gen 2

- Dataflow Gen 2 in **Microsoft Fabric** is a powerful data preparation technology that allows you to create, transform, and load data into Fabric and Azure destinations. Here's what you can do with it:

- **Create Dataflows:**
  - Dataflows are self-service, cloud-based tools for data preparation.
- **Get Data:**
  - Dataflows enable you to retrieve data from 100s of on-premise and cloud data sources.
- **Apply Transformations:**
  - Once you've connected to your data source, it's time to shape it according to your needs.
  - Use the Power Query editor to apply transformations. For instance:
    - Calculate the total number of orders per customer using the Group By feature, combine, remove columns, etc.
- **Configure Destinations:**
  - Specify a destination to store the results of the query and transformations.
- **Publish Dataflows:**
  - After applying transformations, you can publish your dataflow so that it can start processing data.

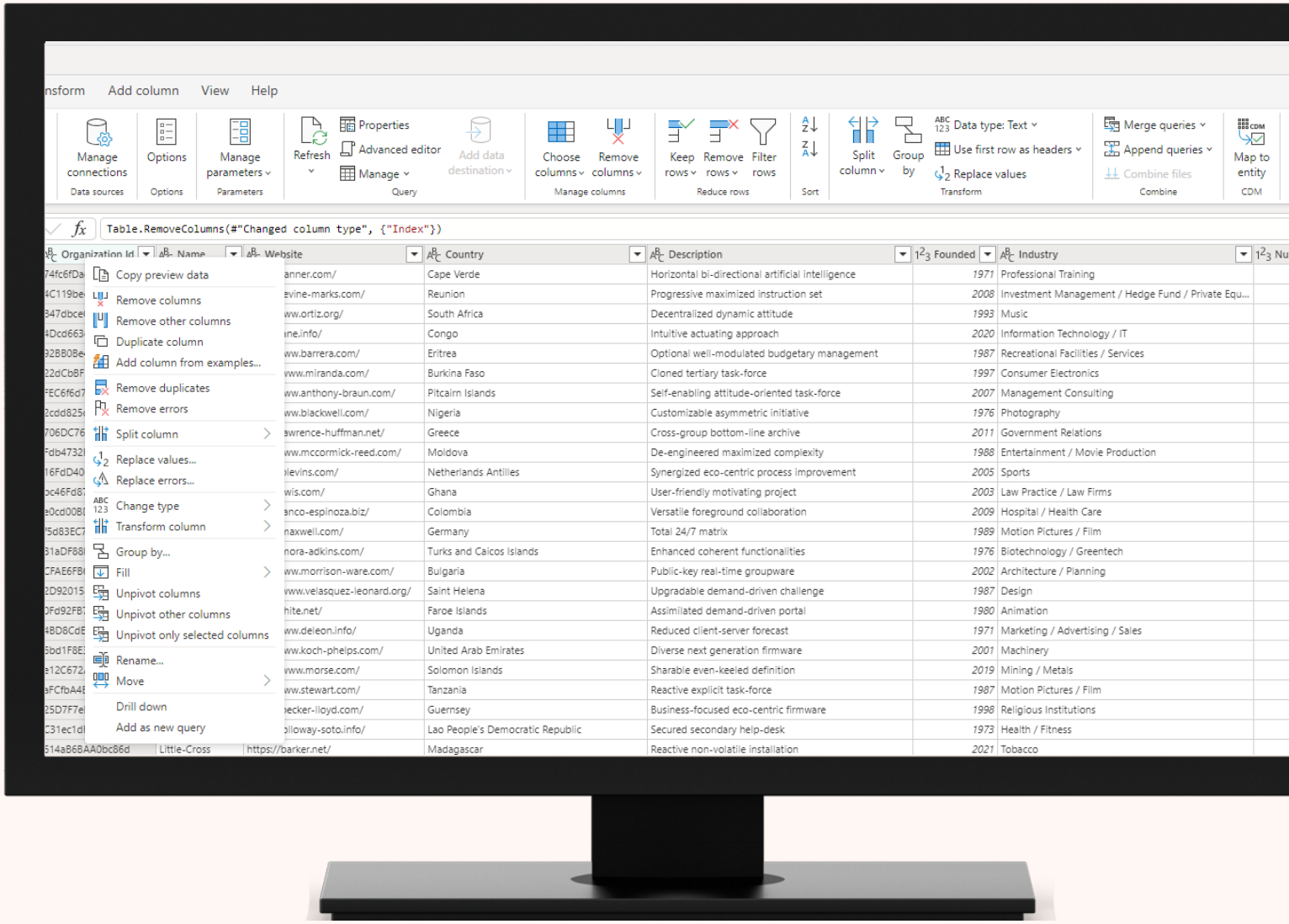
Get Data

- Fabric
- Database
- Azure
- 175+ connectors



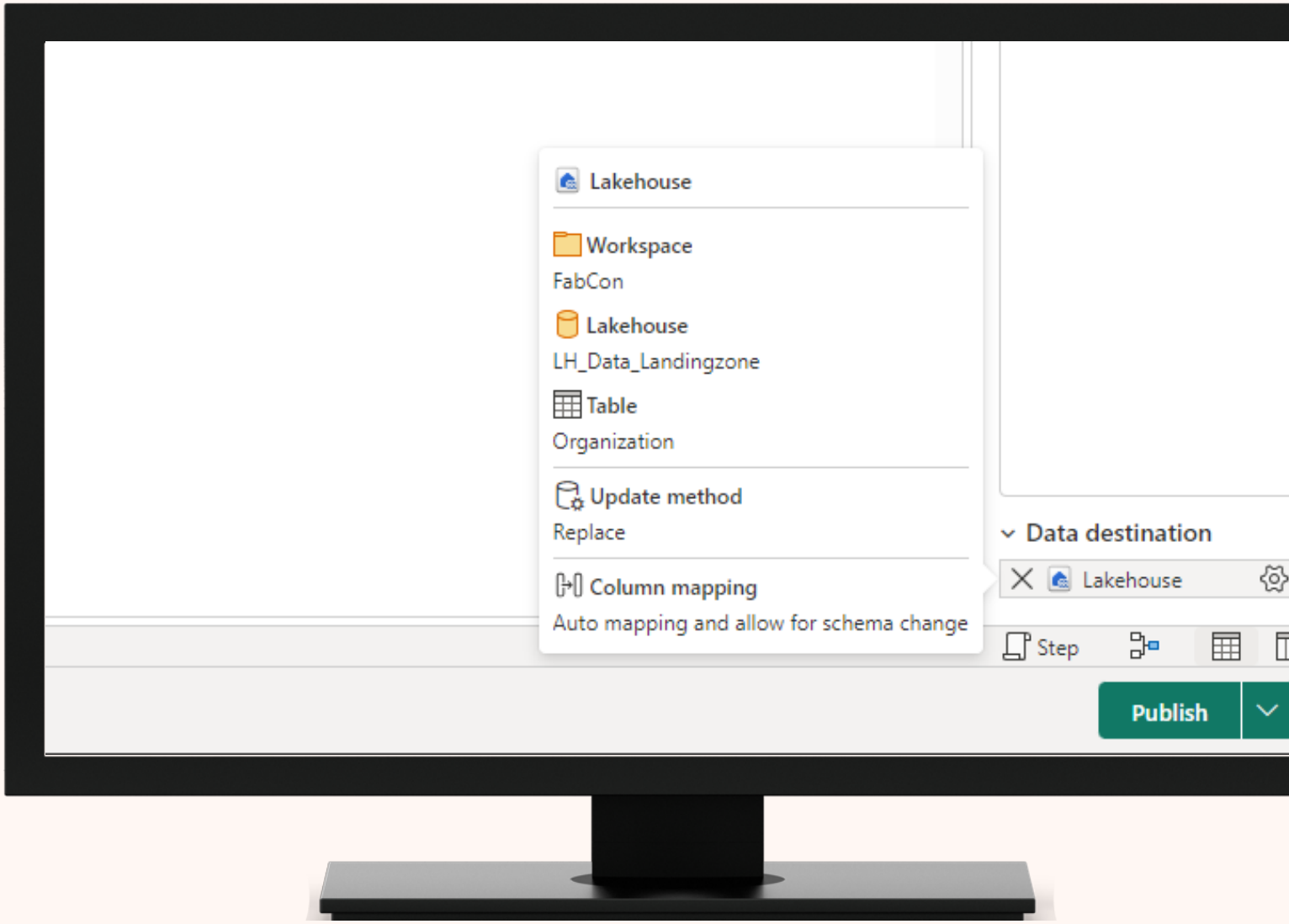
Apply Transformations

- Change data types
- Remove Columns
- Aggregations
- ..



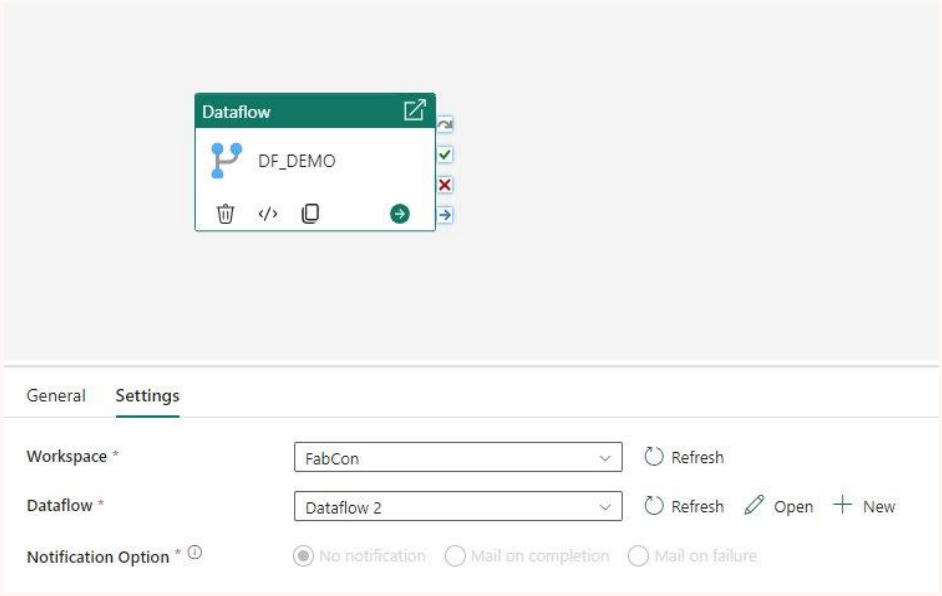
Publish Dataflows

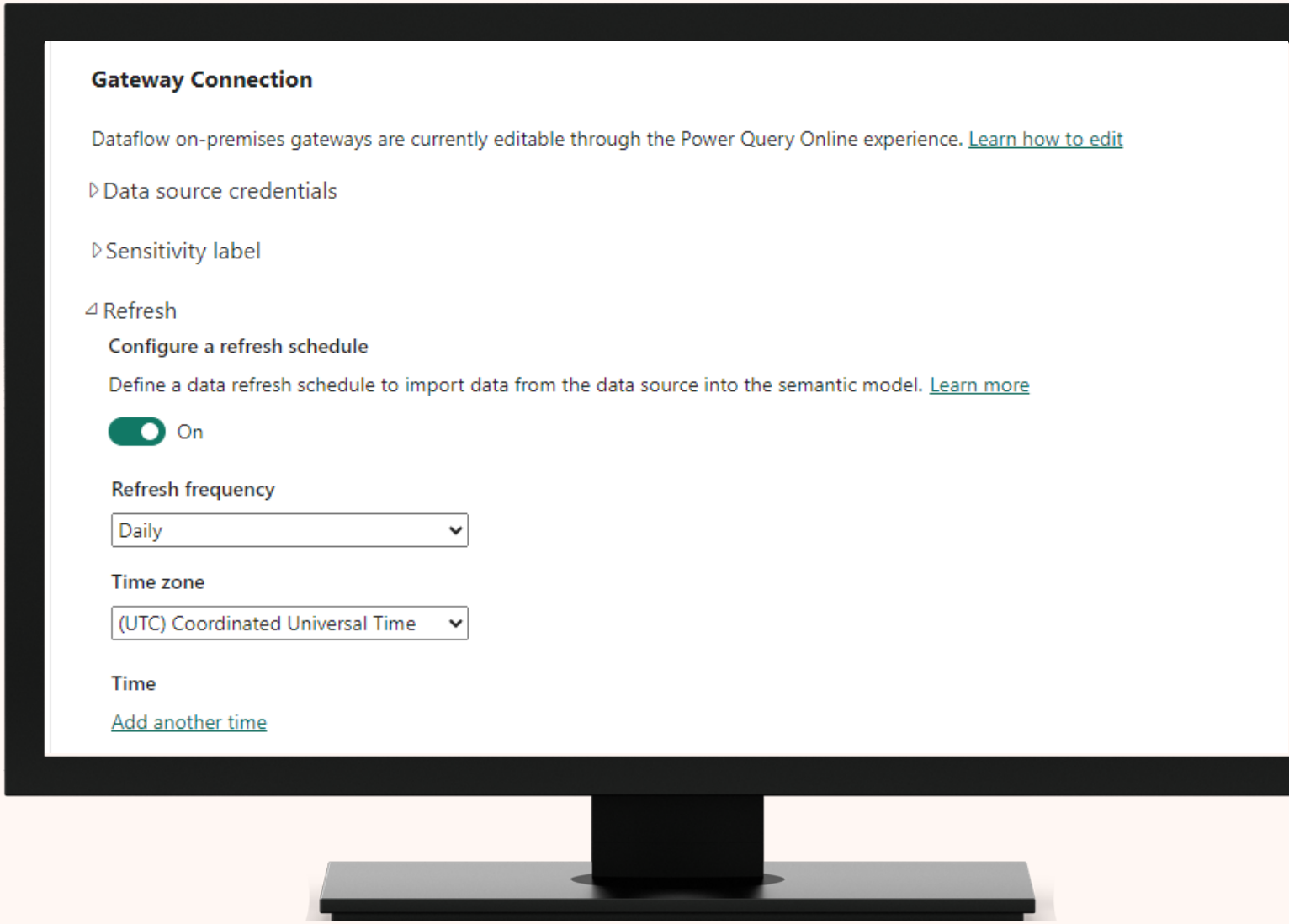
- Select Destination
  - Lakehouse
  - Warehouse
  - Azure Data Explorer
  - Azure SQL Database
    - More to come



Refreshing Dataflows

- Schedule dataflow refreshes for automatic operation
- Execute from a pipeline





Microsoft Fabric
COMMUNITY CONFERENCE

Smart data preparation and deeply integrated experiences

Add column from examples

Transform, combine, extract or enrich your data, in one or more columns within the Power Query Editor tables.

Simply specify a few output values for your new columns and Power Query generates the right column generation logic for you.

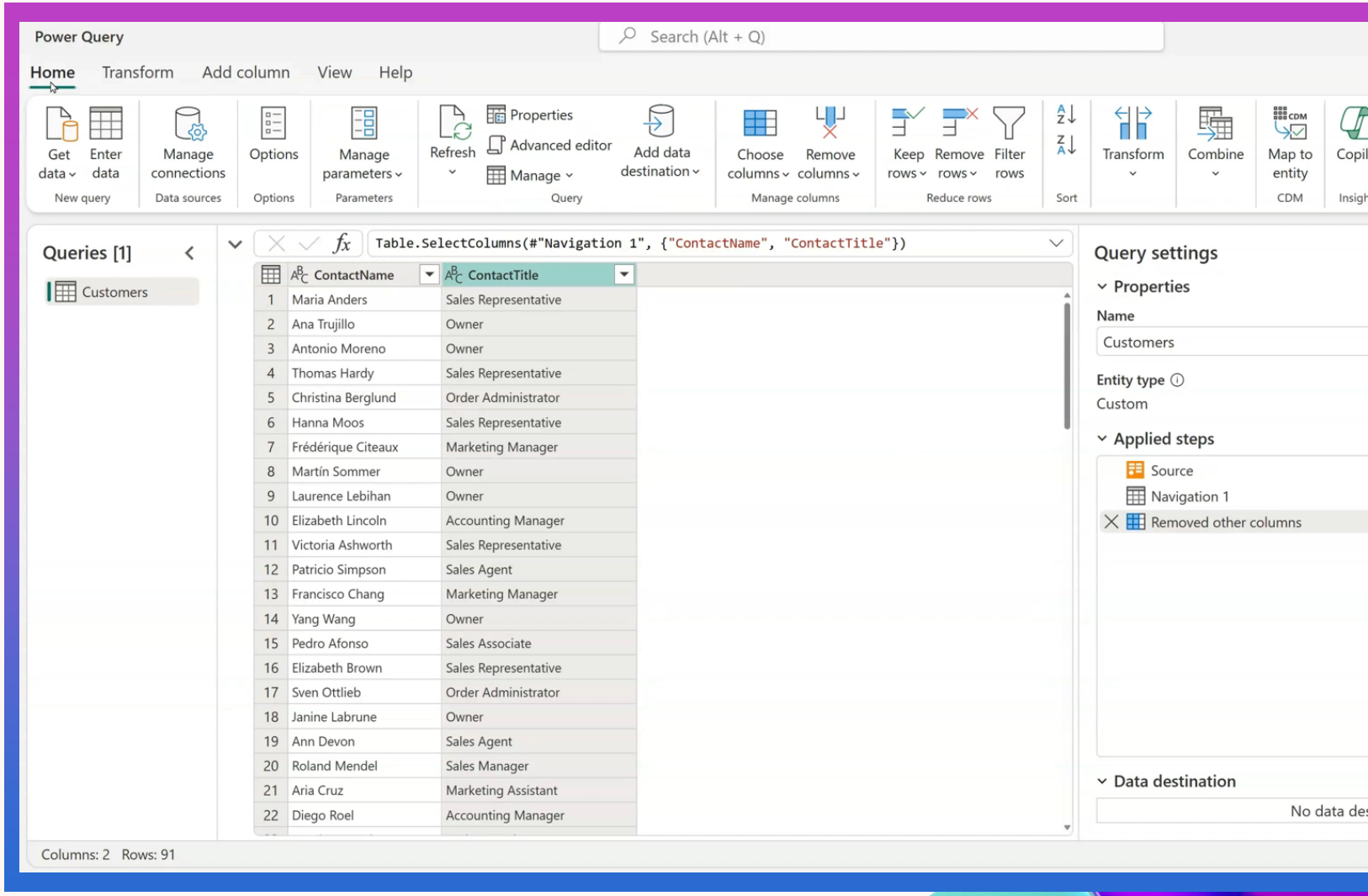


Table by example (Web)

Extract any data from any HTML page in the world. Just preview the source page and specify output values.

Power Query will do the time-consuming work and extract all the appropriate data for you.

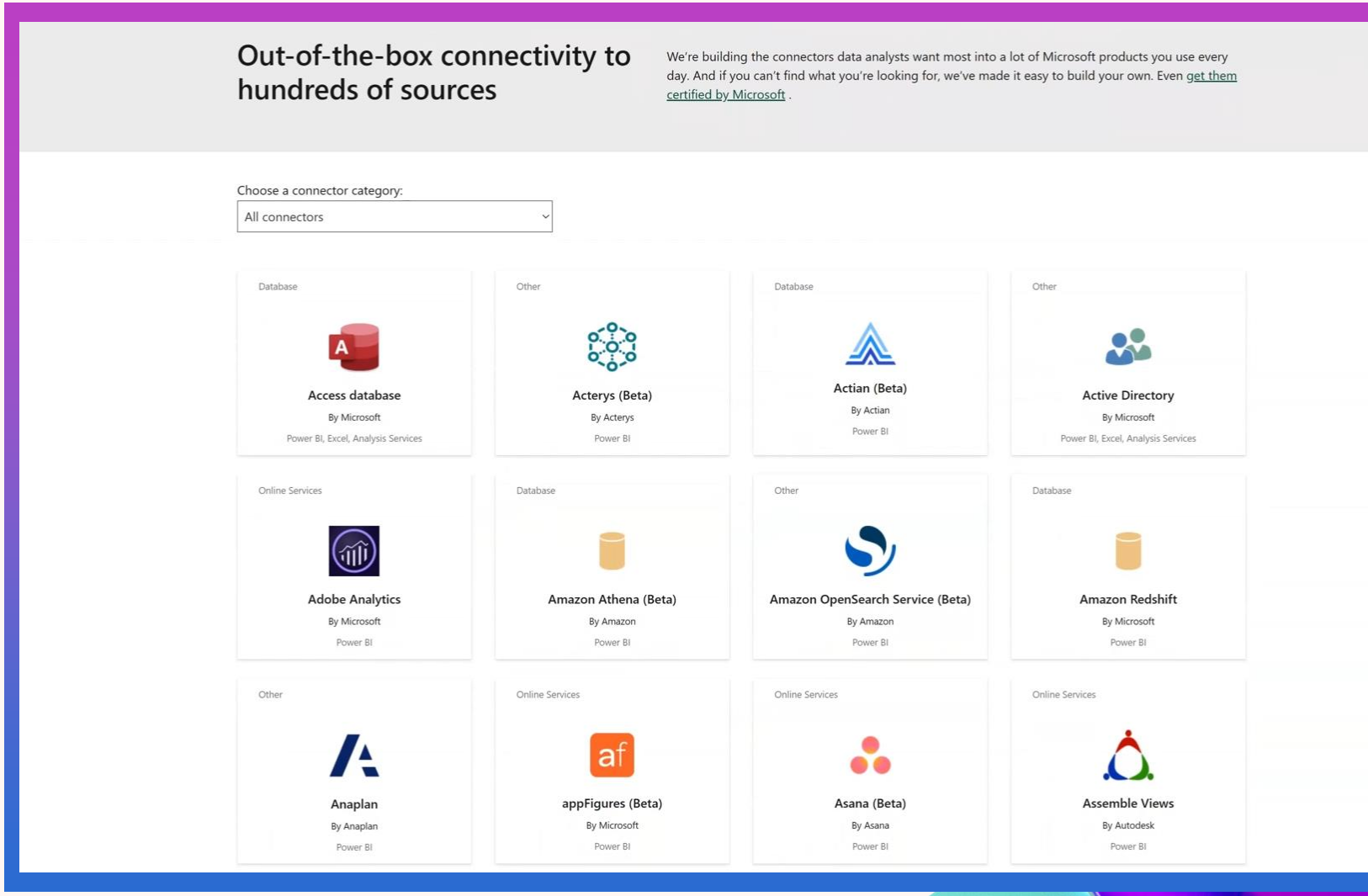
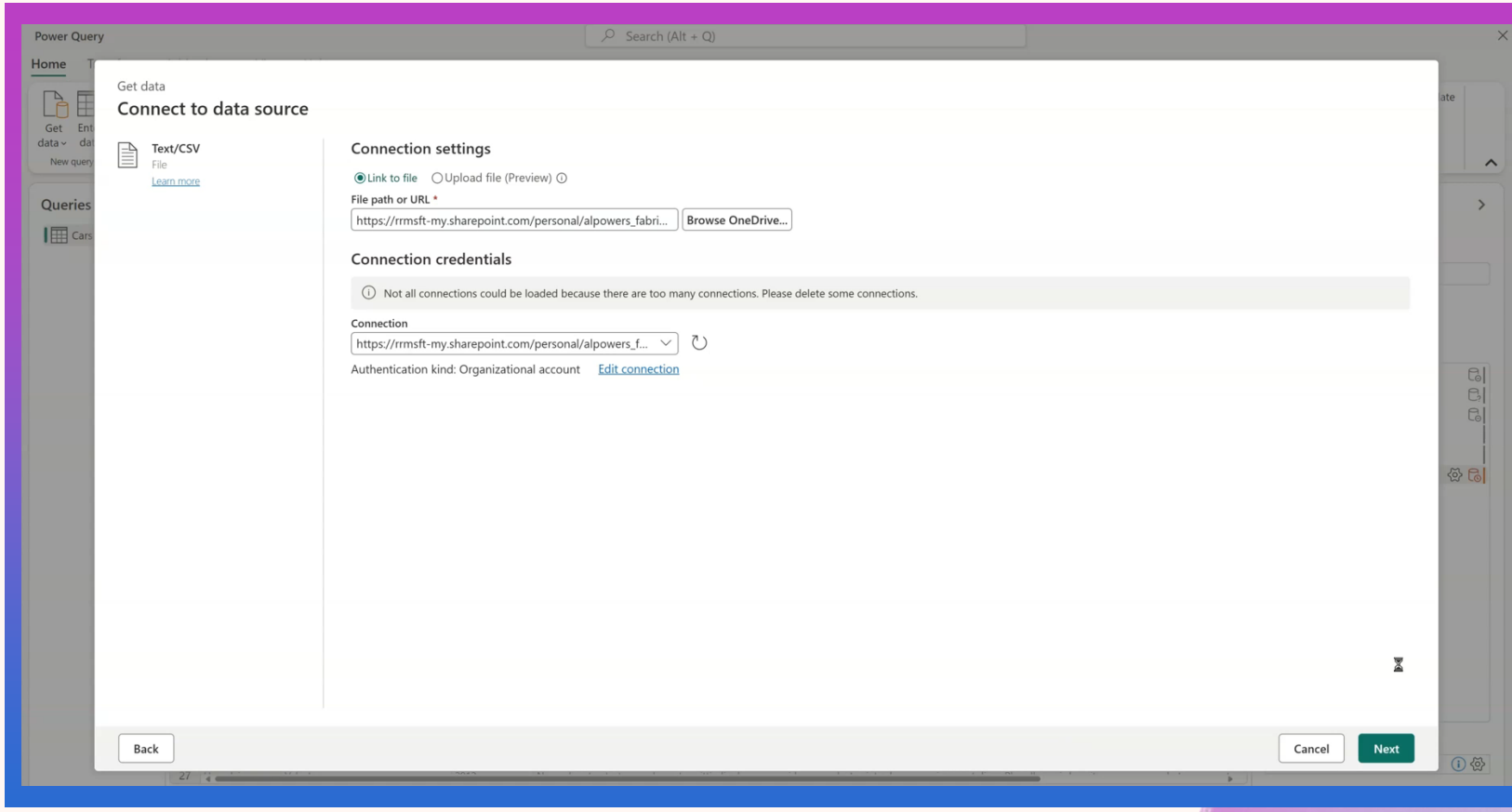


Table by example (Text/CSV)

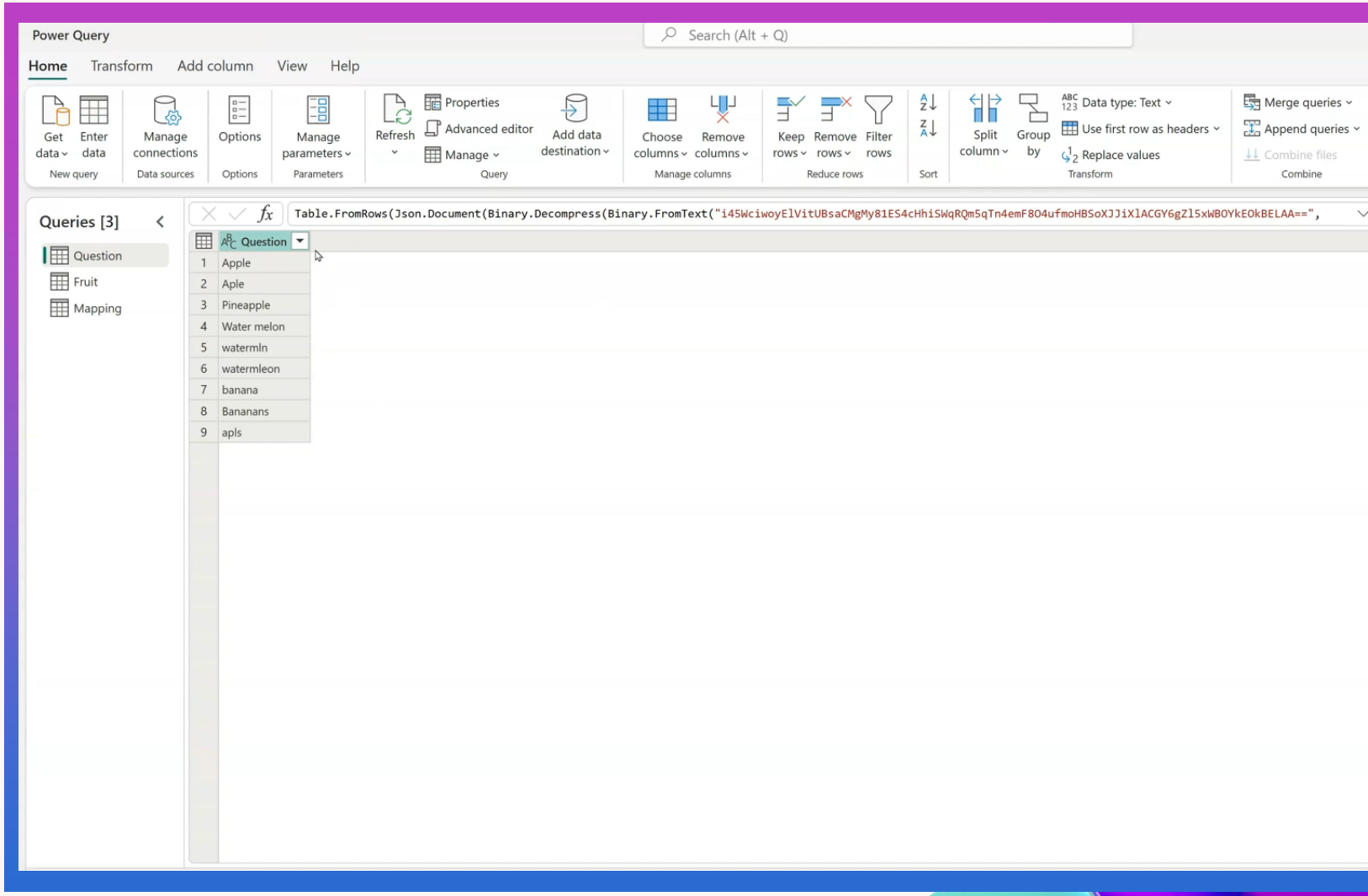
Extract data from a Text/CSV file. Just specify sample output values from the source.

Power Query will do the time-consuming work and extract all the appropriate data for you.



Fuzzy merge

Power Query's built-in Fuzzy Matching algorithm lets you to merge multiple tables using an approximate match to correlate things like slightly different versions of product names, customer names or address information—to name a few examples.



Data profiling

- Inline Column Quality bars

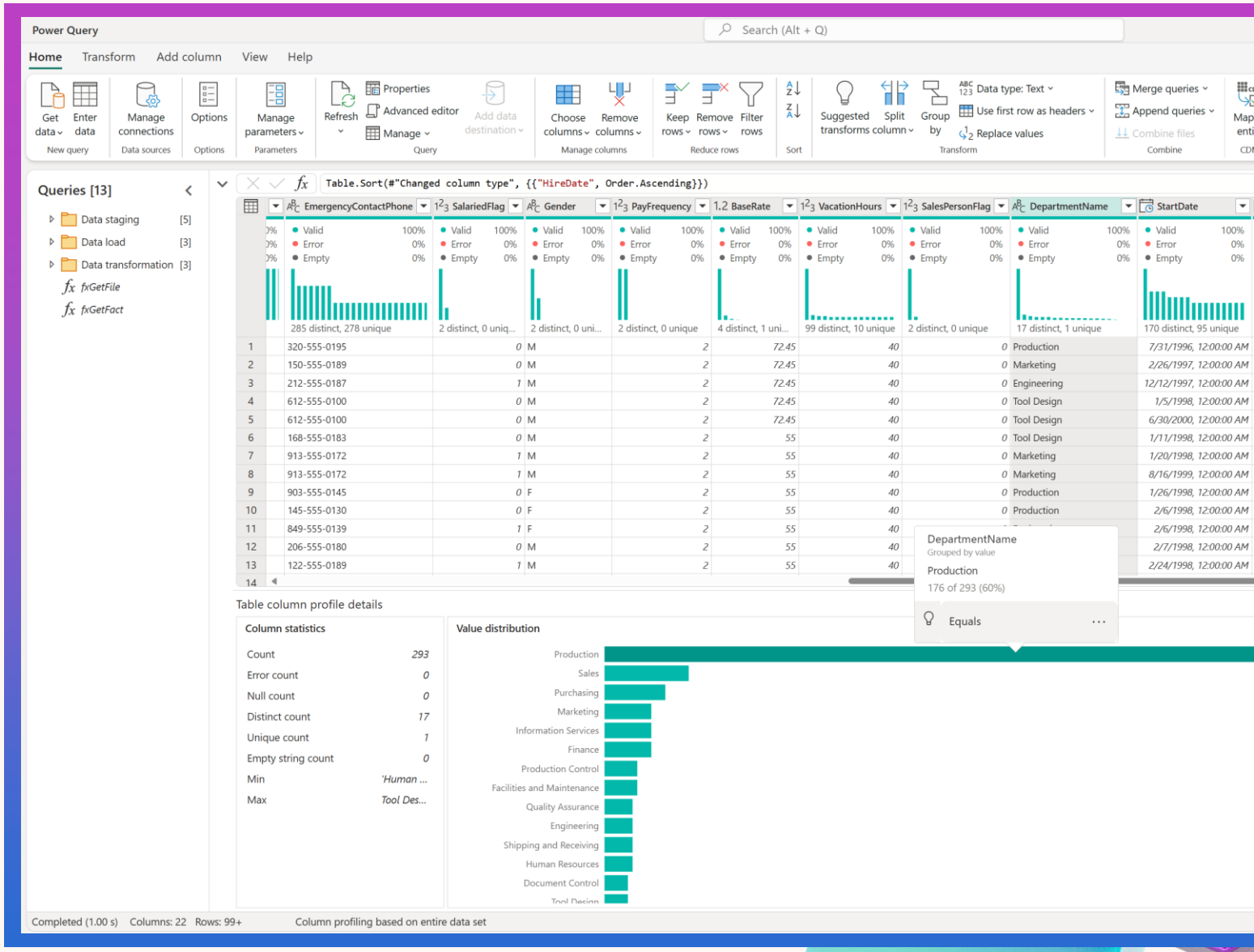
  Quickly spot erroneous or empty values across all your columns.

- Inline Value Distribution histograms

  Understand the number of different values, which are unique, and most/least common.

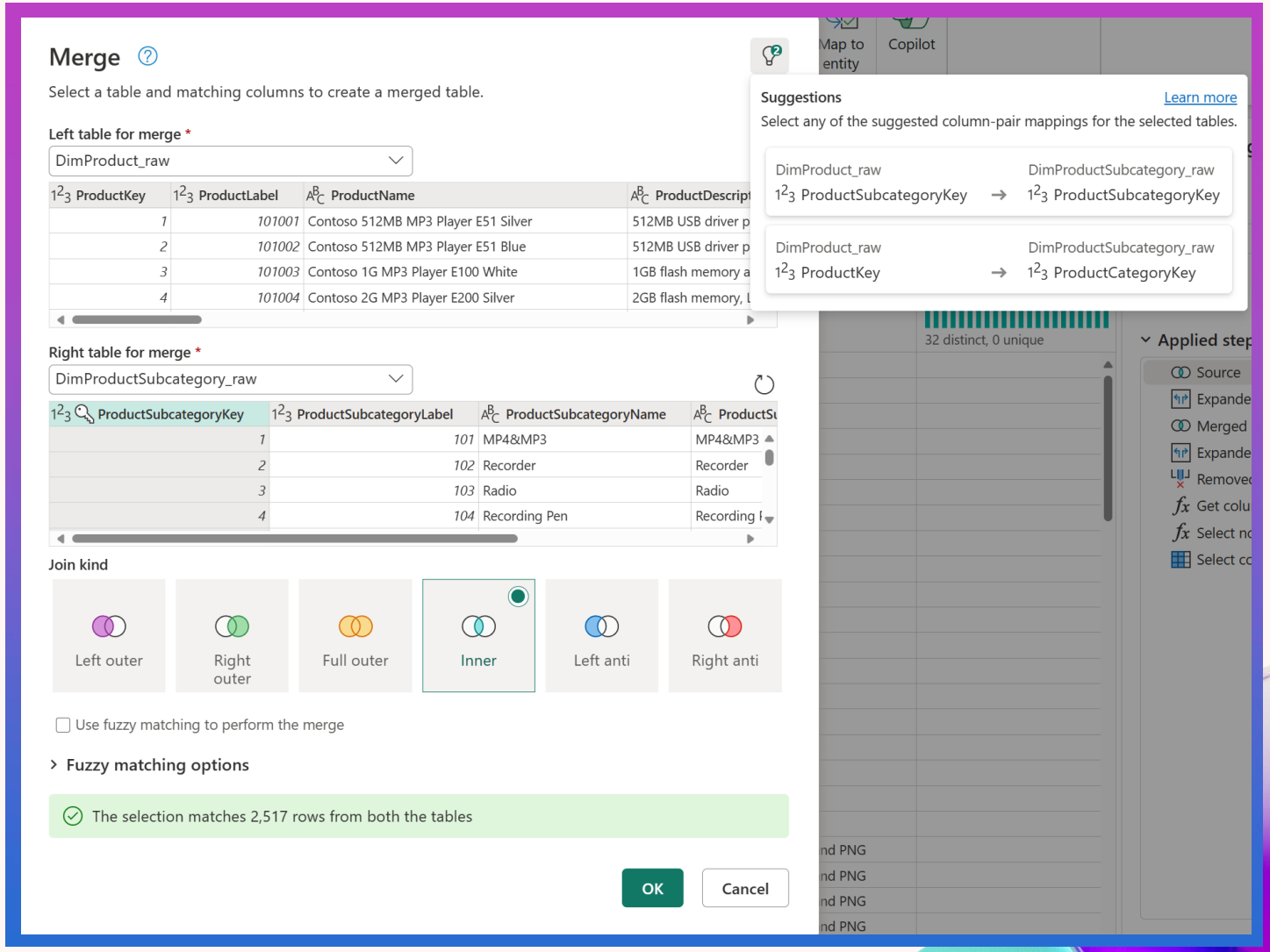
- Detailed Column Profiles pane

  Select a specific column and dig deeper into a profile to fully understand what's in the data.



Column-pair suggestions

- The Merge dialog now intelligently detects matching columns from both the left and right tables, displayed in a convenient lightbulb icon in the top right corner.
  - Precise or approximate column title matches trigger these helpful suggestions.



Microsoft Fabric
COMMUNITY CONFERENCE

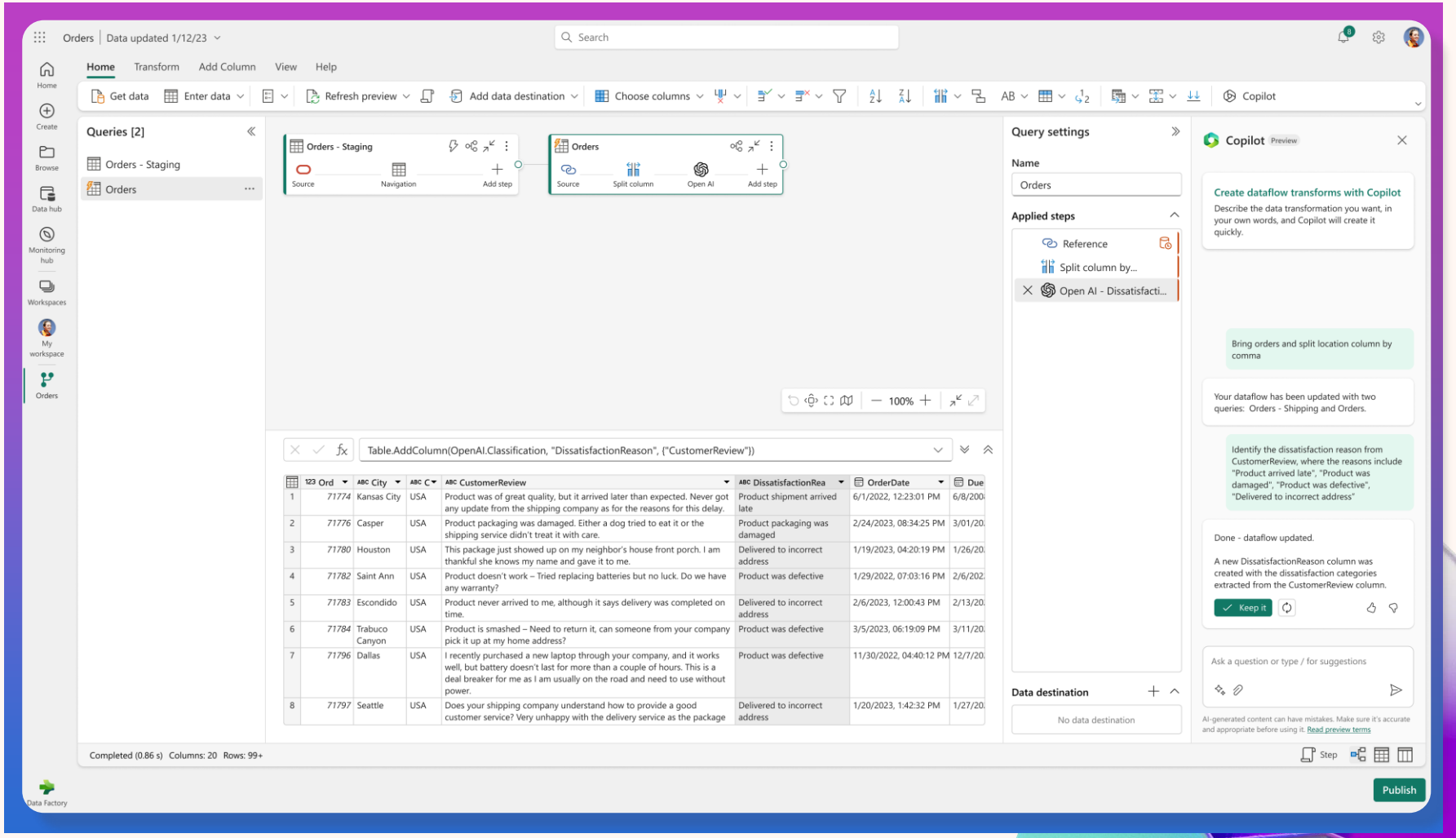
Copilot for Data Factory

Copilot for Data Factory

Easily integrate generative AI into your **dataflows** using Copilot

Chat with **Copilot** to describe data transformations in natural language

Tap into generative AI capabilities from **Azure Open AI** as data transformation steps



Copilot skills

Create new transformations

- Add a single transformation step to an existing query
- Chain multiple transformation steps to an existing query

Create a new query

- Create a new query from scratch or by referencing existing data

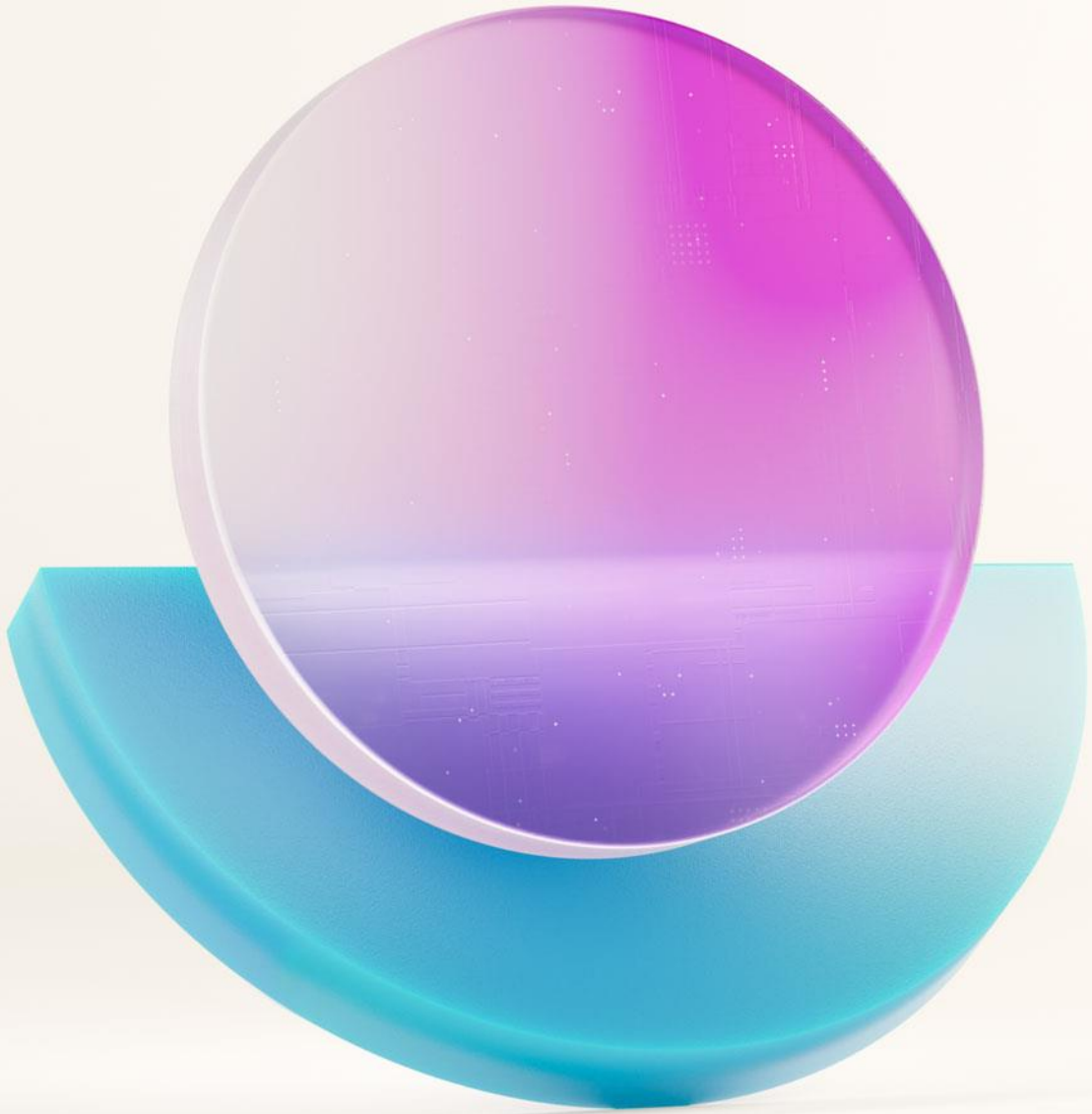
Explain a single step

- Generates a description of a transformation step

Explain the full query

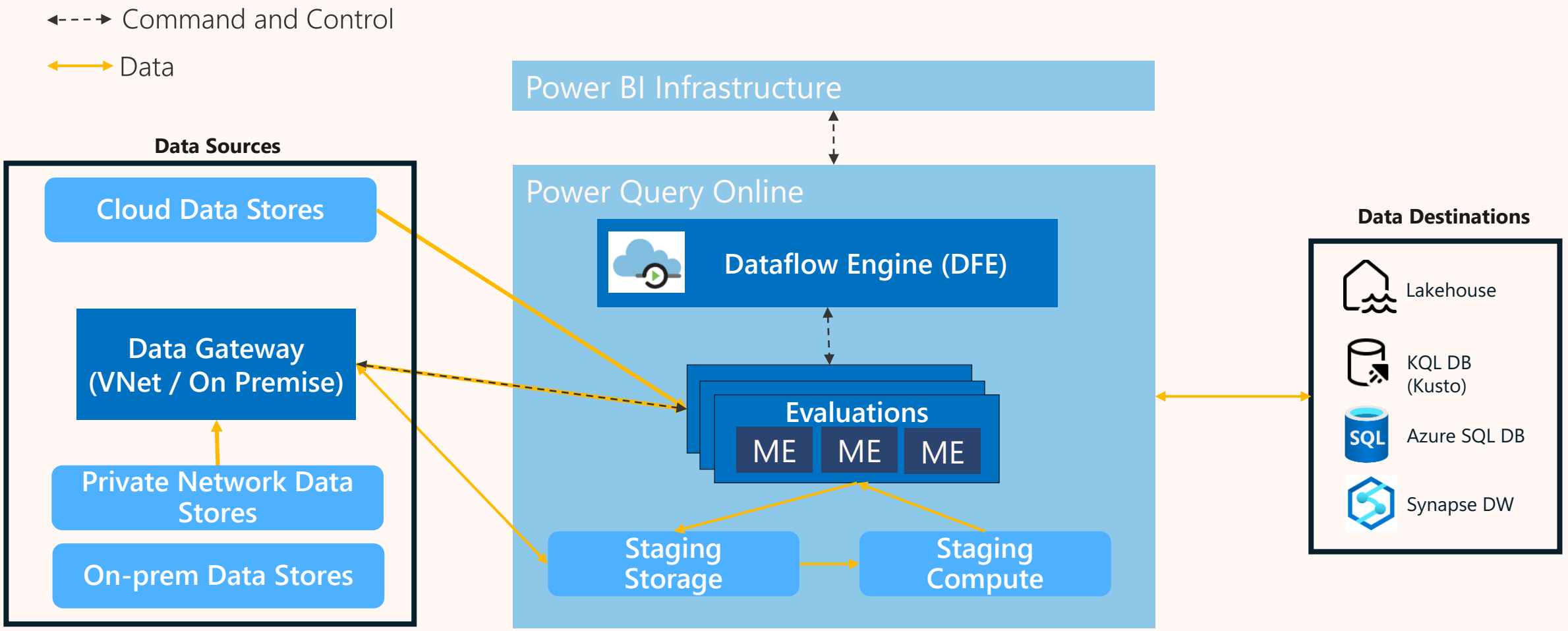
- Generates a description of a complete query and its steps

Microsoft Fabric
COMMUNITY CONFERENCE

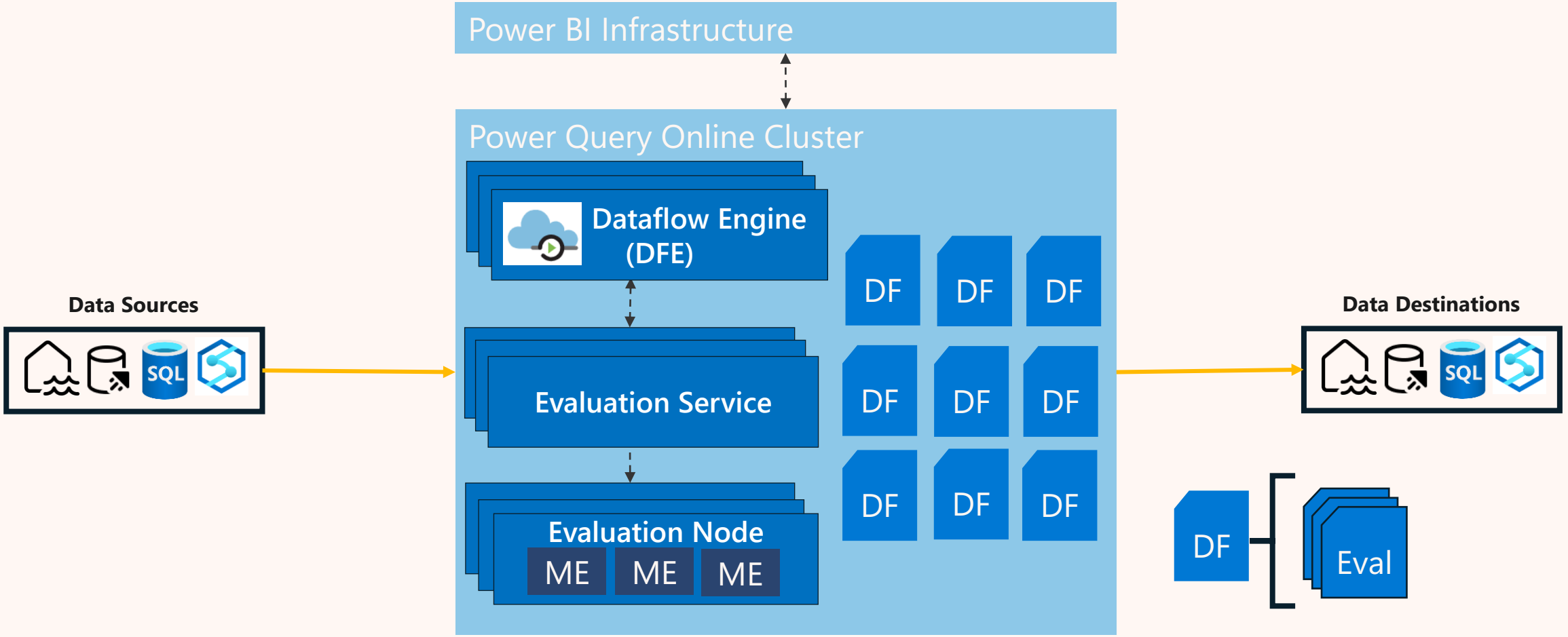


Dataflows In-Depth

Fabric Dataflows (aka Gen2)

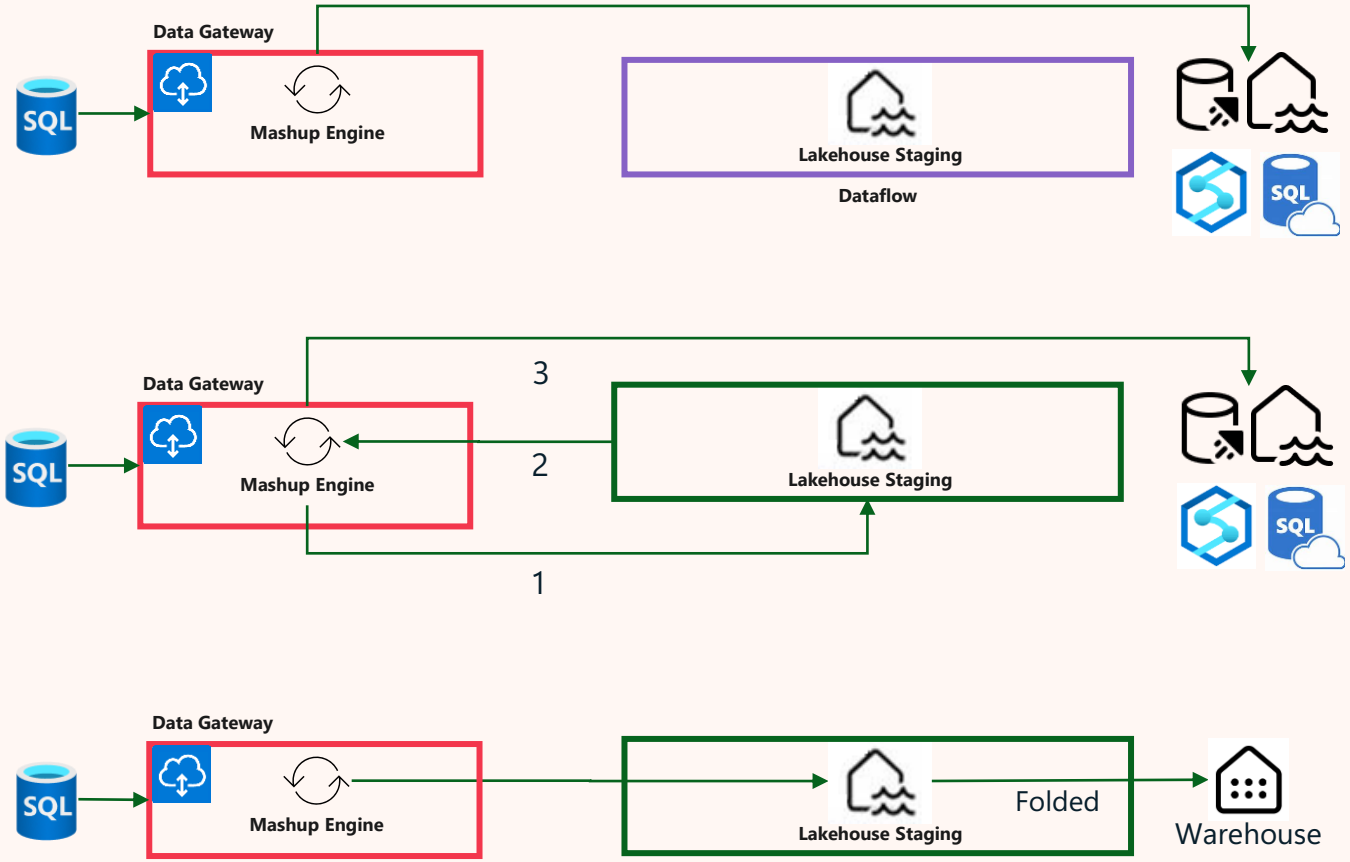


Fabric Dataflows Scale Out



Data Gateway (VNET / On Premise)

- Capabilities
  - Enables access to data sources without exposing them to the internet
  - Data is processed locally
- When to use it
  - Data source is inside your private network
  - Move compute closer to source
  - Control where evaluations happen
- When to skip it
  - Entities that stage data and write to destination (excluding WH)
    - Split Ingest and Destination into separate dataflows (consider where to Transform)
    - Don’t stage



Dataflows Performance Principles

Four Principles

- Delegate to the most capable resource
- Do the expensive work first
- Divide and conquer
- Do as little work as possible

Dataflows Performance Capabilities

Principle	Capability	High-Level Approach
Delegate	Query Folding	Send as much of the query as possible to the underlying data source
Expensive work first	Staging	For large volumes of data reused by multiple query, first land the shared data in staging
Divide & conquer	Fast Copy	Leverage the Copy Activity for parallelized, high-scale data movement prior to transformation (ELT)
	Partitioning	Divide expensive queries into sub-queries that can be executed in parallel
Be lazy	Incremental Refresh	During refreshes, process only the data that has changed
	Lazy Evaluation	Leverage the feature of the “M” language that minimizes the amount of data traversed

Query Folding – What it is

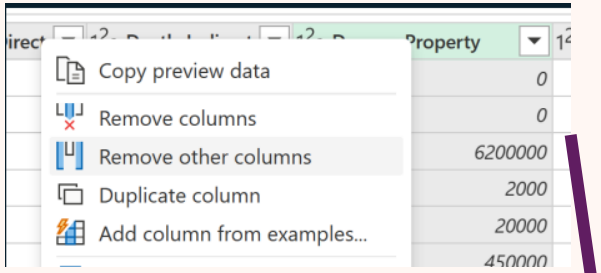
- Terms that are synonymous to Query Folding
  - Query Delegation
  - Query Push Down
  - Remote/Distributed Query Evaluation

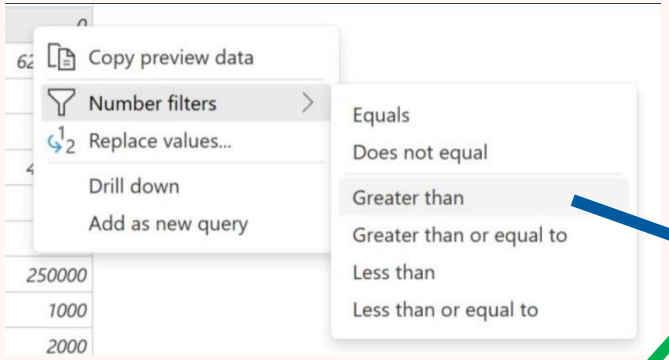
- Wherever possible, the script in the Power Query Editor (“M”) is translated to a “native query”

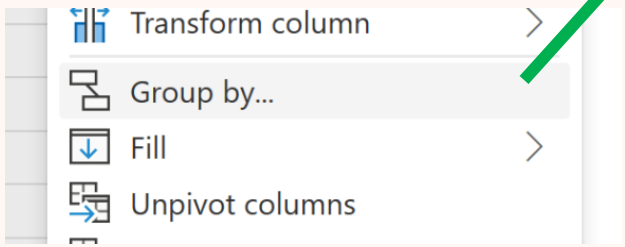
- The native query is then executed by the underlying data source.

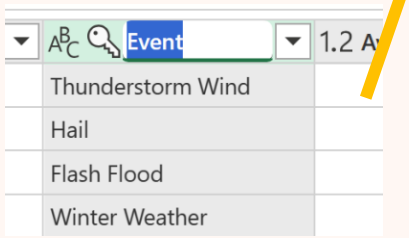
https://learn.microsoft.com/en-us/power-bi/guidance/power-query-folding

Query Folding - Example










```
let
  Source = Sql.Databases("server"),
  Navigation1 = Source{[Name = "Samples"]}[Data],
  Navigation2 = Navigation1{[Name = "StormEvents"]}[Data],
  SelectColumns = Table.SelectColumns(Navigation2, {"State", "EventType", "DamageProperty"}),
  Filter = Table.SelectRows(SelectColumns, each [DamageProperty] > 0),
  Group = Table.Group(Filter,{"State","EventType"},{{"AvgDmg",each List.Average([DamageProperty])}}),
  Rename = Table.RenameColumns(Group, {{"EventType", "Event"}})
in
  Rename
```

Query Folding – SQL native queries

The Dataflow runtime translates the M query to the data source's native query language, pushing down as much work as possible to the backend. In this example, all transformations steps can be folded to SQL.

```
let
  Source = Sql.Databases("server"),
  Navigation1 = Source{[Name = "Samples"]}[Data],
  Navigation2 = Navigation1{[Name = "StormEvents"]}[Data],
  SelectColumns = Table.SelectColumns(Navigation2, {"State", "EventType", "DamageProperty"}),
  Filter = Table.SelectRows(SelectColumns, each [DamageProperty] > 0),
  Group = Table.Group(Filter,{"State","EventType"},{{"AvgDmg",each List.Average([DamageProperty])}}),
  Rename = Table.RenameColumns(Group, {{"EventType", "Event"}})
in
  Rename
```

```
SELECT      [State], [EventType] as "Event", AVG([DamageProperty]) as AvgDmg
FROM        [StormEvents]
WHERE       [DamageProperty] > 0
GROUP BY    [State], [Event]
```

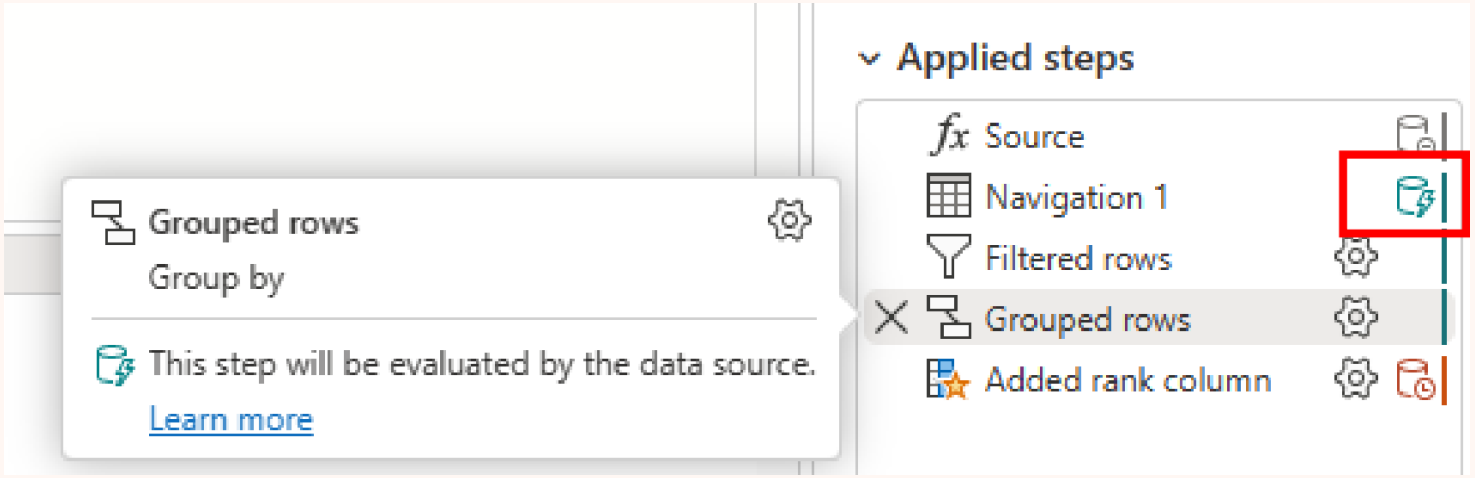
Query Folding – Indicators

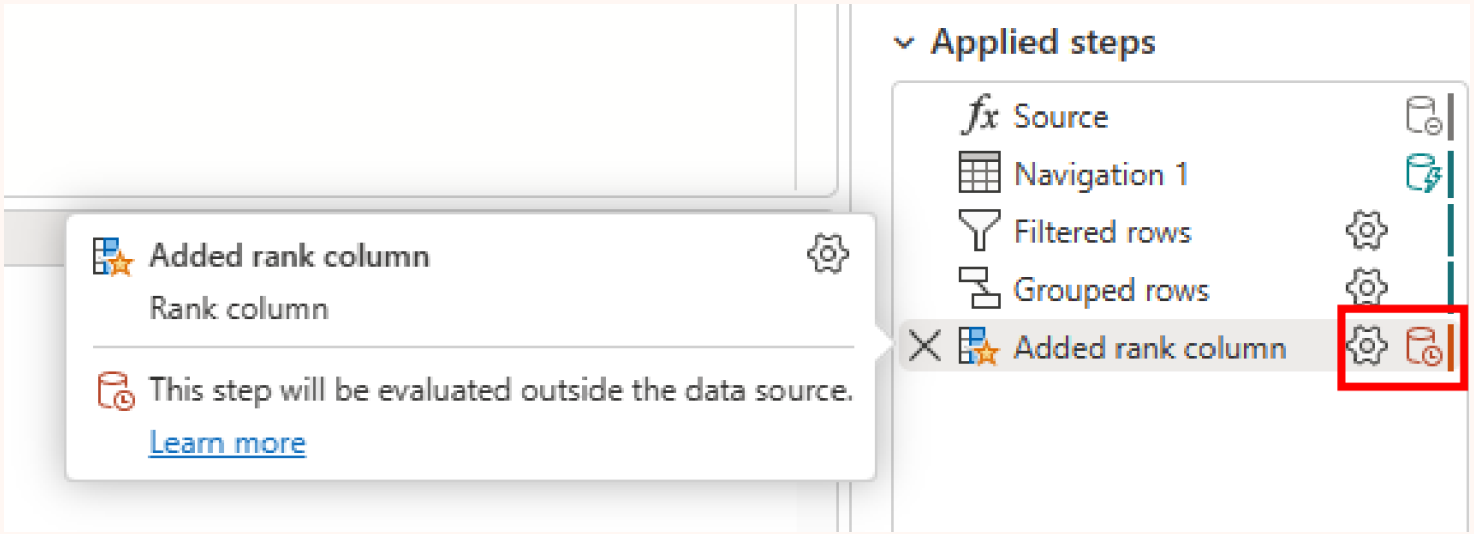
Green “lightning bolt” indicator for folded steps

(typically, faster)

Red “clock” indicator for inmemory steps evaluated outside the data source

(typically, slower)





Query Folding – Why it's Faster

- Reduced data movement
  - In-memory evaluation engine would have to first transfer the data from the source
  - A filter that selects 10 rows from a 10M row table, can avoid transferring 10M rows

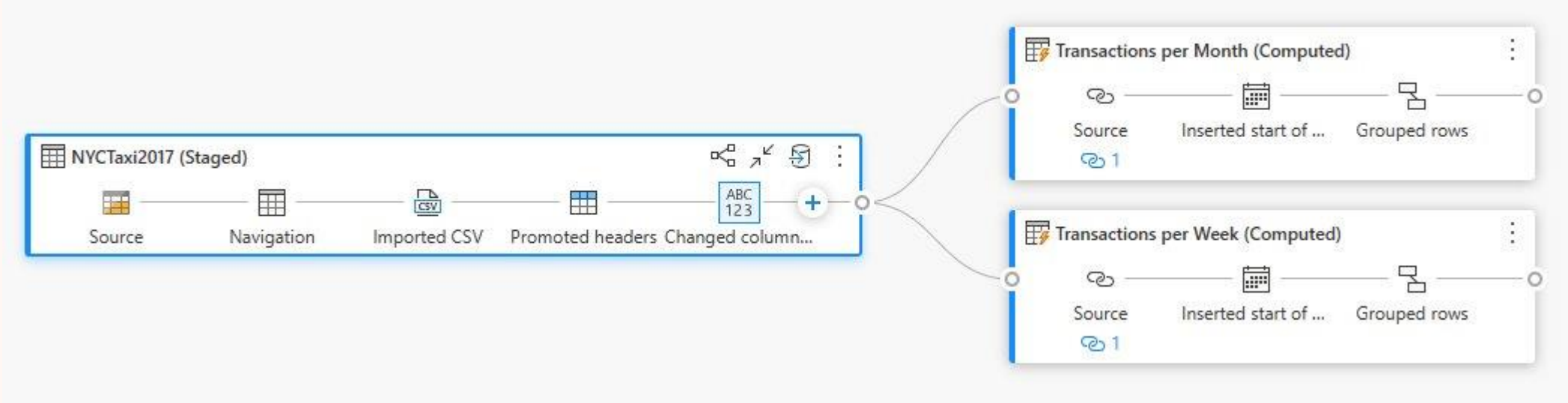
- Many data sources have highly efficient "query processors"
  - Optimized query plans
  - Ability to leverage indices, keys, etc.

Tips

1. Perform foldable operations early on
2. Do data type conversions last as they frequently break folding
3. Use the Table.StopFolding function to force local evaluation of subsequent steps

Staging – What it is

- Loading data into Fabric storage (Staging Lakehouse) as a first step
- The staged data can then be referenced by downstream queries that benefit from SQL Compute over the staged data



Staging - Tips

- **Data sources without rich query folding (e.g. files) are good candidates for staging**, especially when the files are large

- Data sources with rich query folding (e.g. databases) often do not require staging

- For a faster authoring experience create **separate dataflows for staging and transformation**
  - The transformation dataflow references the staging dataflow's data via the Dataflows connector

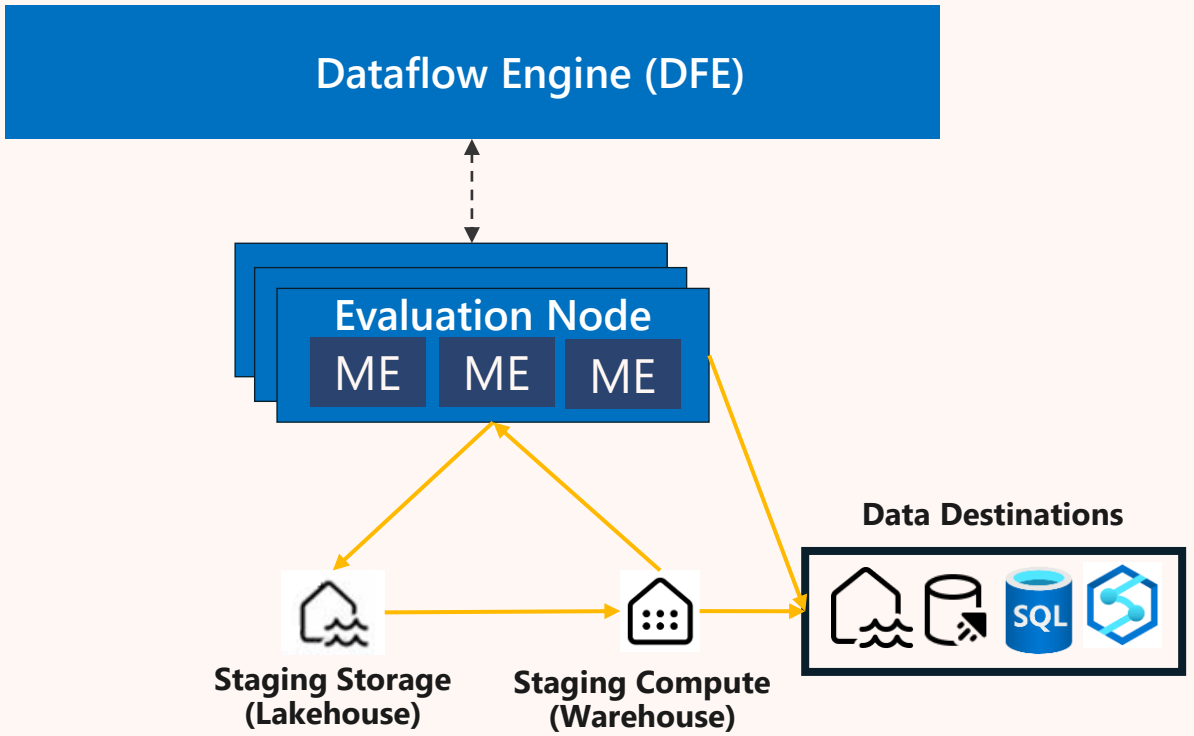
https://blog.crossjoin.co.uk/2023/09/03/fabric-dataflows-gen2-to-stage-or-not-to-stage/

Staging – Why it's Faster

- In Fabric (Gen 2) Dataflows, staged data is stored in a Staging Lakehouse
- Queries against the Staging Lakehouse benefit from a SQL analytics endpoint
- SQL analytics endpoint queries can be orders of magnitude faster than directly querying "slow sources" like files (or SharePoint..)

File	Size	Direct Grouping Query	Staged Grouping Query
2020 Yellow Taxi Trip Data (CSV)	2.2 GB	4 mins 12 secs	2.8 secs
2017 Yellow Taxi Trip Data (CSV)	9.8 GB	Timeout (> 10 mins)	4.6 secs

Staging - Architecture

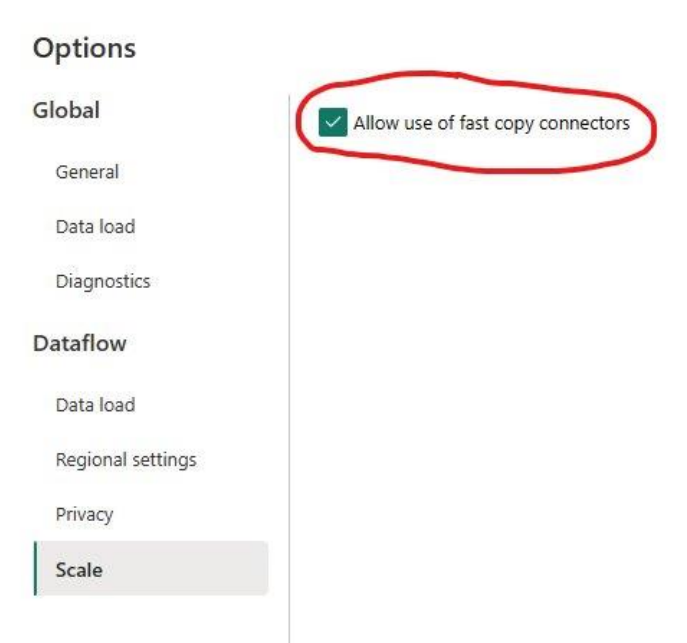


Fast Copy – What it is

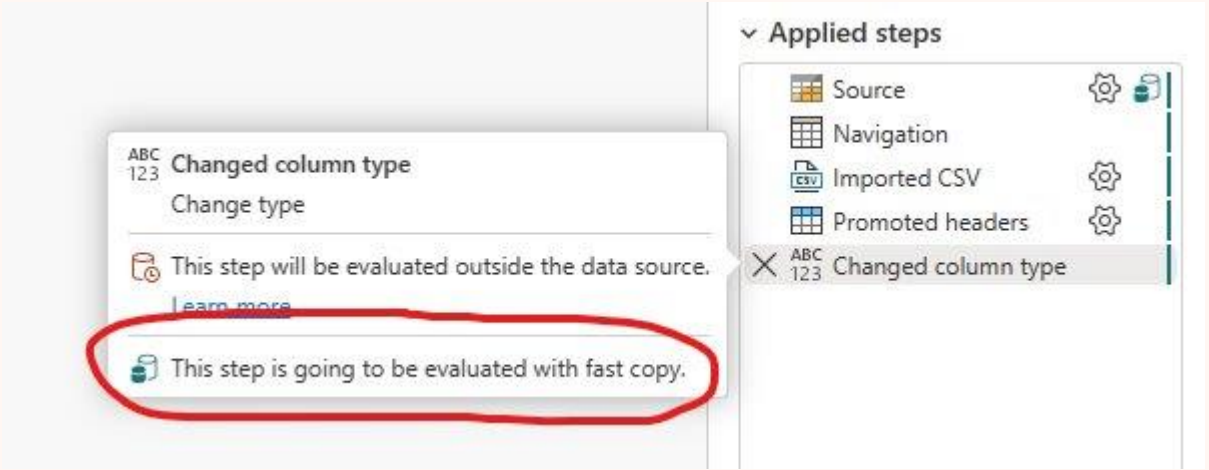
Highly parallelized data movement

- Powered by Azure Data Factory Copy Tasks
- Standard Power Query experience

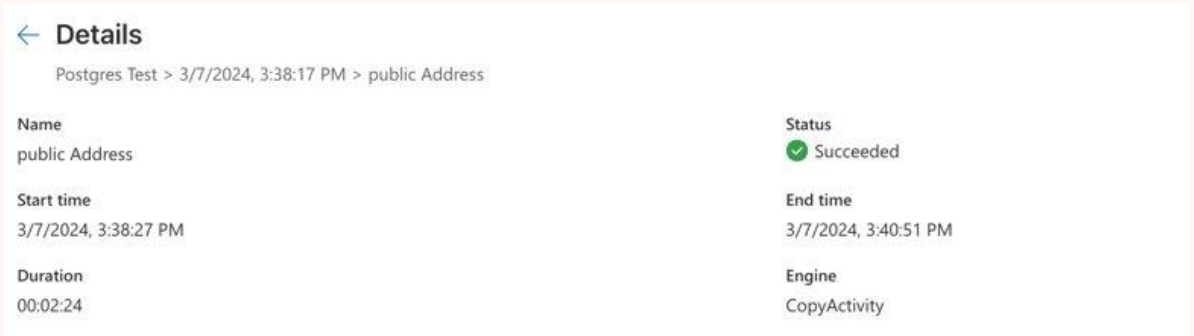
Step 1: Enable Fast Copy



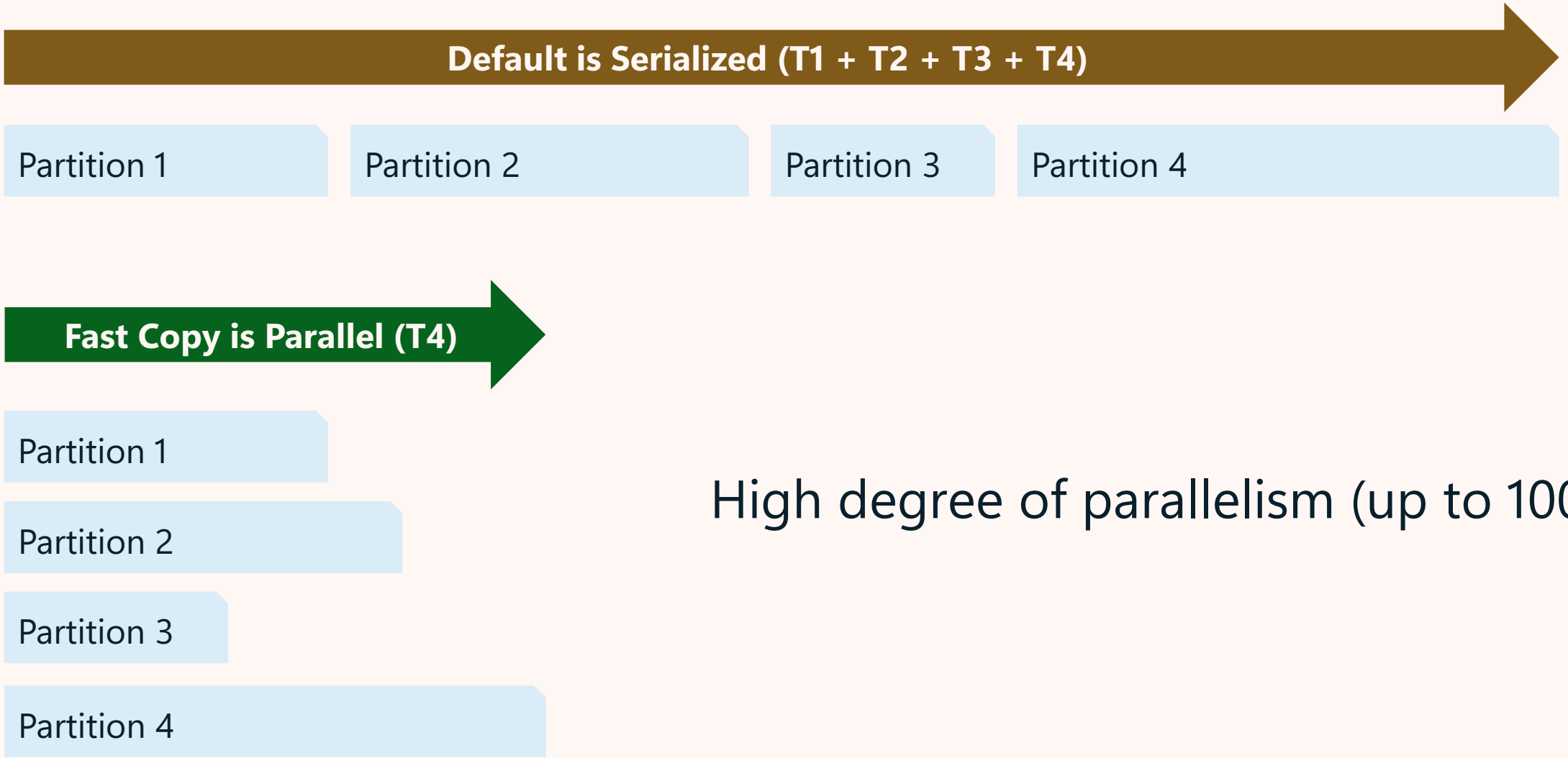
Step 2: Use Indicators in Steps pane to verify use of Fast Copy



Step 3: Win

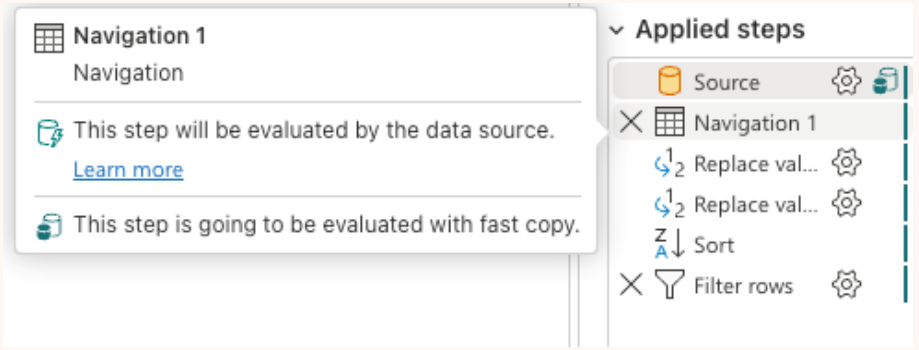


Fast Copy – Why it’s Faster

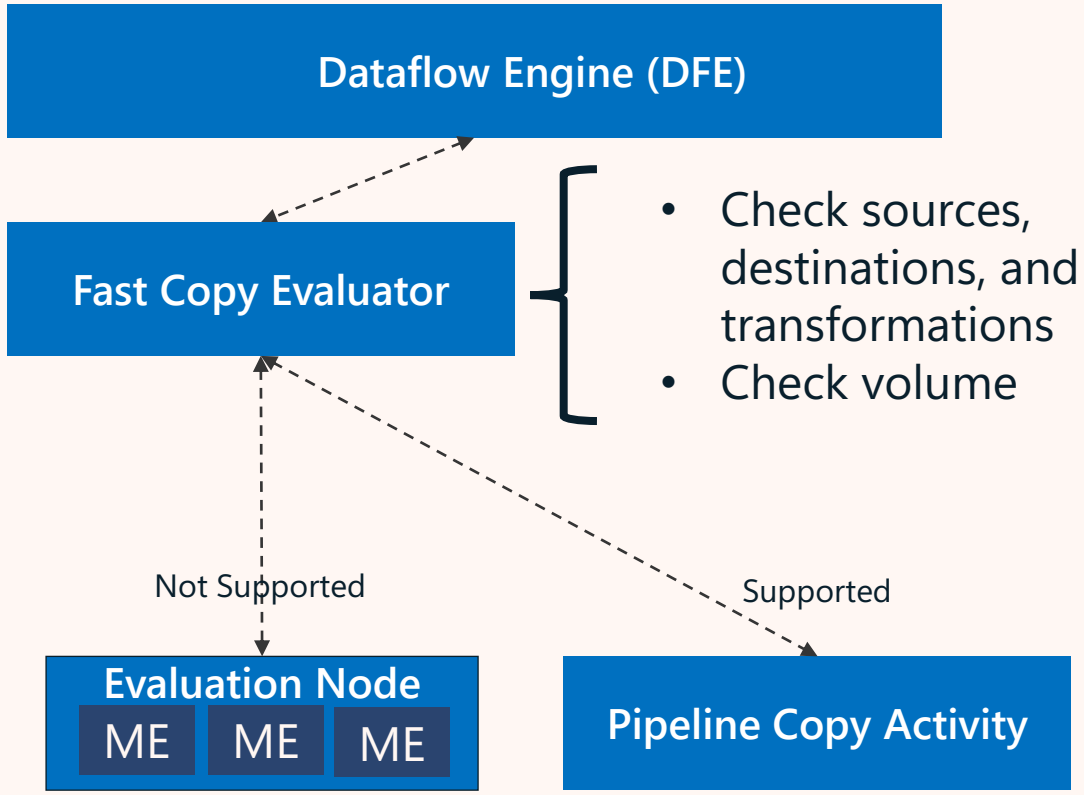


High degree of parallelism (up to 100)

Fast Copy - Architecture



- Capabilities
  - Leverages Pipeline Copy Activity for large performance boost in ingest
  - Automatically used based on pattern matching and volume
  - Transparent (no pipeline to manage)
- When to use it
  - Whenever possible
    - Defer transformations to post ingest if they affect Fast Copy use
  - Enable in Options..Scale..Allow use of fast copy connectors
- When to skip it
  - Don't – if it's an option, use it
  - Mark as "Require fast copy" to enforce



Partitioning – What it is

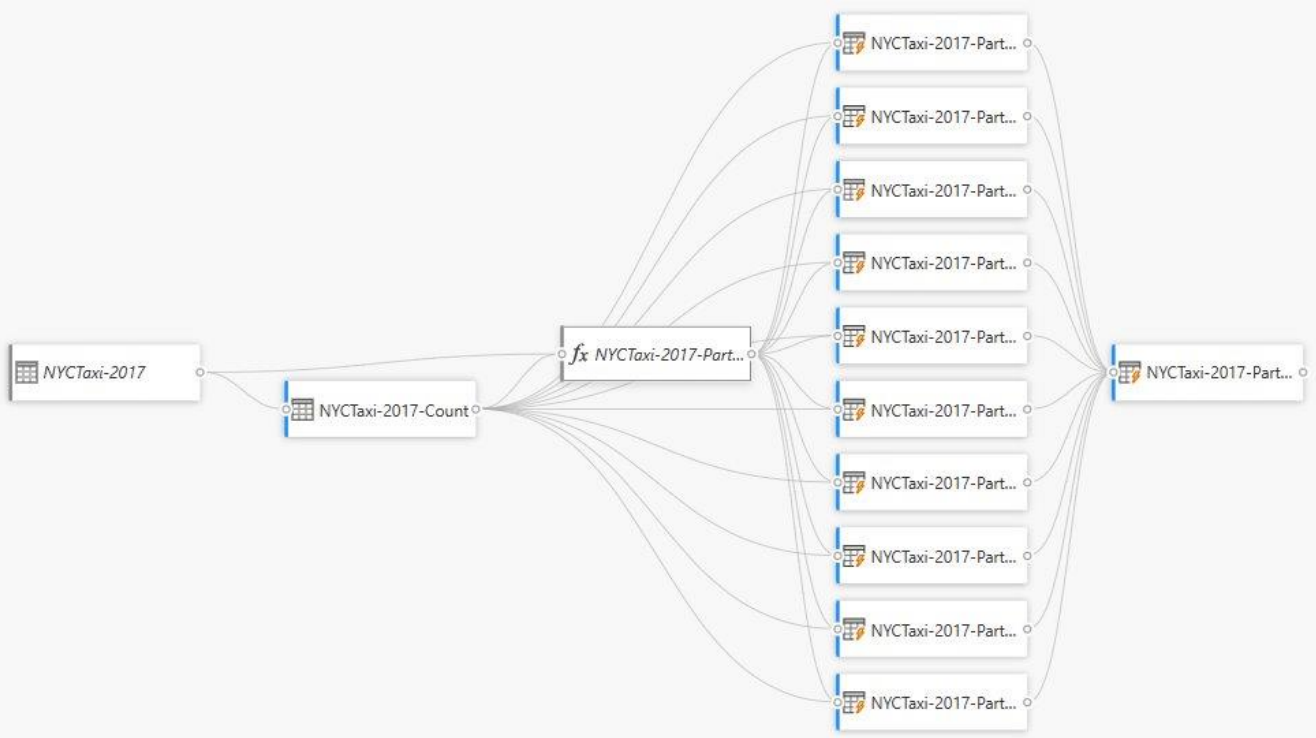
- Breaking long-running queries into smaller queries that can be run in parallel
- Explicit specification of what Fast Copy does implicitly
- Faster for the same reason as Fast Copy – parallelism

Partitioning – Sample

2017 Yellow Taxi Trip Data (9.8GB)

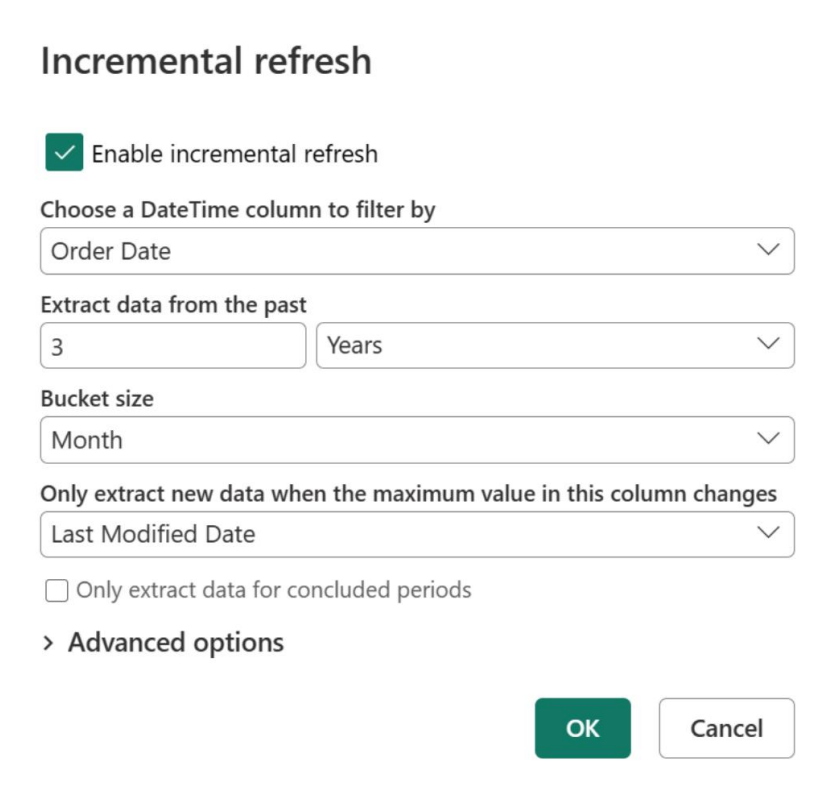
1. Calculate count (~113M rows)
2. Divide into 10 dynamic buckets using Table.Range (~11.3M rows per bucket)
3. Union the buckets

7x faster load time!



Incremental Refresh – What it is

- Specifying a timeframe
  - Example: Past 3 years
- Breaking a query into time-based buckets
  - Example: Month-based buckets based on “Order Date”
  - Bucket types: Days, Weeks, Months, Quarters, Years
- Specifying a “watermark” column
  - Example: Refresh a bucket only if “Last Modified Date” has increased since the last refresh
- Public Preview in first half of 2024!



https://learn.microsoft.com/en-us/fabric/data-factory/tutorial-setup-incremental-refresh-with-dataflows-gen2

Incremental Refresh – Why it's Faster

- Primarily about "doing less work", but also embodies many of the other principles:

- Dividing and Conquering (partitions)
  - Queries are broken into sub-buckets (partitions) that are processed in parallel
  - Even though the initial load processes all data, it benefits from parallelization

- Doing Less Work
  - If a bucket has not changed (based on the "high watermark check"), it is not processed

- Delegating (query-folding)
  - The check to see whether a bucket has changed is a filter that is folded to the underlying data source

Leveraging the Architecture

- Carefully consider ETL vs ELT when laying out dataflows
- Know why you are using staging
  - Accelerates some operations, but adds no value to others
- Avoid “double-hops” with Data Gateways
- Take advantage of Fast Copy wherever possible
  - As patterns are added, dataflows can automatically benefit
  - Enable “use fast copy connectors”
  - Ensure that queries fully fold to maximize fast copy usage
- Build for parallel processing
  - Evaluations, dataflows
  - Incremental refresh when available

New Features for DI at #FabCon

GENERAL AVAILABILITY

VNET Data Gateway support with Private Links for Dataflows Gen 2 in Fabric

PREVIEW

Data Pipelines access on-premises data using "On Premises Data Gateway" (OPDG)

Fast Copy for Dataflows

40 to 80 activity limit in Data Pipelines

Semantic Model Refresh

CI/CD in Data Pipelines

Cancel Dataflow Refresh

SPN support for VNET Data Gateway

Modern Get Data – browse Azure Connections

Dataflow output destinations – Support for schema changes for Lakehouse & Azure SQL DB

SNEAK PEAK

Incremental Refresh for Dataflows

Resources

- https://blog.fabric.microsoft.com/en-us/blog/data-factory-spotlight-dataflows-gen2
- https://blog.fabric.microsoft.com/en-us/blog/modern-get-data-in-dataflows
- https://blog.fabric.microsoft.com/en-us/blog/service-principal-support-to-connect-to-data-in-dataflow-datamart-dataset-and-dataflow-gen-2
- https://blog.fabric.microsoft.com/en-us/blog/dataflows-gen-2-data-destinations-and-managed-settings
- https://blog.fabric.microsoft.com/en-us/blog/updates-to-default-data-destination-behavior-dataflow-gen-2
- https://blog.fabric.microsoft.com/en-us/blog/fast-copy-in-dataflows-gen-2

Slides

https://github.com/BenniDeJagere/Presentations/{Year}/{YYYYMMDD}_{Event}