



Deep dive into Direct Lake

Mathias Halkjaer & Marc Lelijveld

Thank you to our Fabric February Friends!

twoday



bouvet

sopra  steria



DATAmasterminds

 **ADVANCING
ANALYTICS**

 **Evidi**

 **Profisee**
Master Data Management

 **Tabular Editor**

KURANT

 **Fraktal**

 **CluedIn**

 **Dufrain**
THE DATA COMPANY



Mathias Halkjær

Principal Architect, Data & Analytics
Fellowmind Denmark



@halkjaerm

linkedin.com/in/mhalkjaer

Fluxbi.com

FAVORITE STUFF:



Marc Lelijveld

Technical Evangelist | Solution Architect
Macaw Netherlands



 @MarcLelijveld

 linkedin.com/in/MarcLelijveld

 Data-Marc.com

FAVORITE STUFF:





Storage modes

Different types of storage modes

Three familiar storage modes

- **Import** – data cached in the model
- **DirectQuery** – queries are submitted to the back-end data source
- **Dual** – can act in both above storage modes, depending on query context

The screenshot shows the Power BI Desktop interface with a data model diagram and properties pane.

Data Model Diagram:

- Product Subcategory:** English Product Subcategory Name, French Product Subcategory Name, ProductCategoryKey, ProductSubcategoryAltName, ProductSubcategoryKey, Spanish Product Subcategory Name.
- Product:** Arabic Description, Chinese Description, Class, Color, Days To Manufacture, Dealer Price.
- Internet Sales - Agg:** Count, Order Calendar Year, ProductSubcategoryKey, Sales Amount.
- Internet Sales:** Carrier Tracking Number, CurrencyKey, Customer PO Number, CustomerKey, Discount Amount, DueDateKey, Extended Amount, Freight Amount, Order Calendar Year.

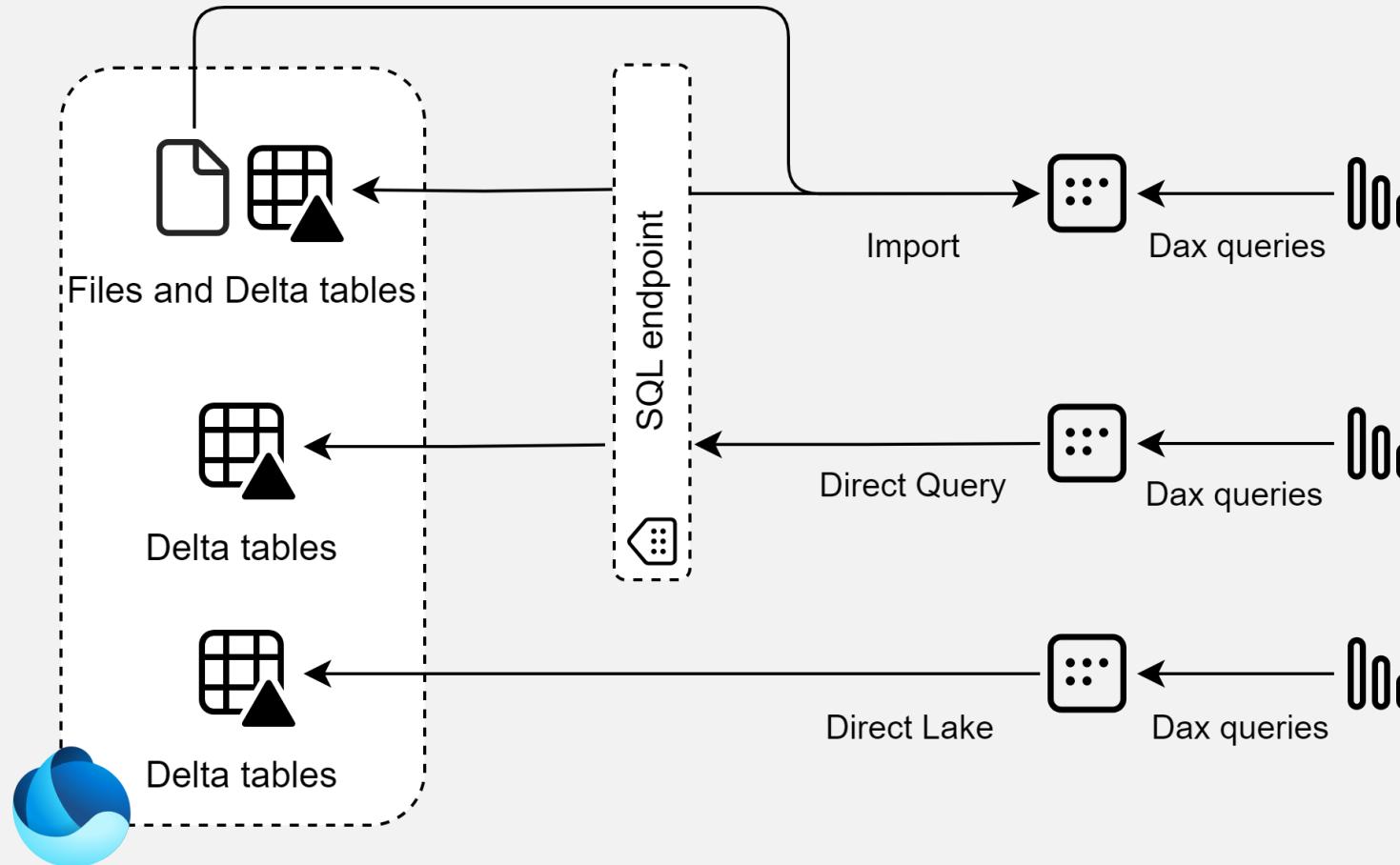
Relationships:

- Product Subcategory (1) — Product (*).
- Product Subcategory (1) — Internet Sales - Agg (*).
- Product (*) — Internet Sales (*).

Properties Pane (Advanced section):

- Name: Internet Sales
- Description: Enter a description
- Synonyms: Enter a comma-separated list of synonyms for Q&A
- Row label: Select a row label
- Key column: Select a column with unique values
- Is hidden: No
- Is featured table: No
- Storage mode:** DirectQuery (highlighted with a red box)

Direct Lake

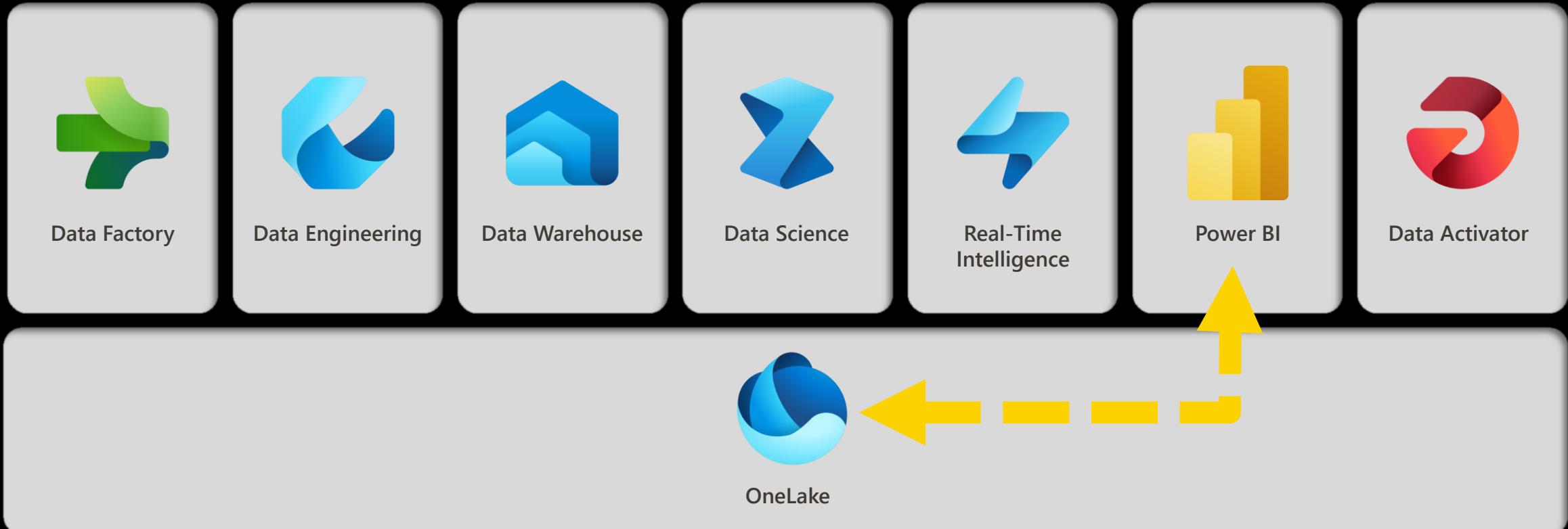


Latent & duplicative but fast

Slow, but real time

Considered being
Best of both worlds

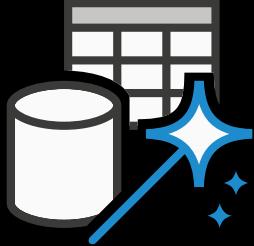
Direct Lake is only applicable to Fabric





Build your data model benefitting Direct Lake

Data transformations



No Power Query or other data transformation capabilities*

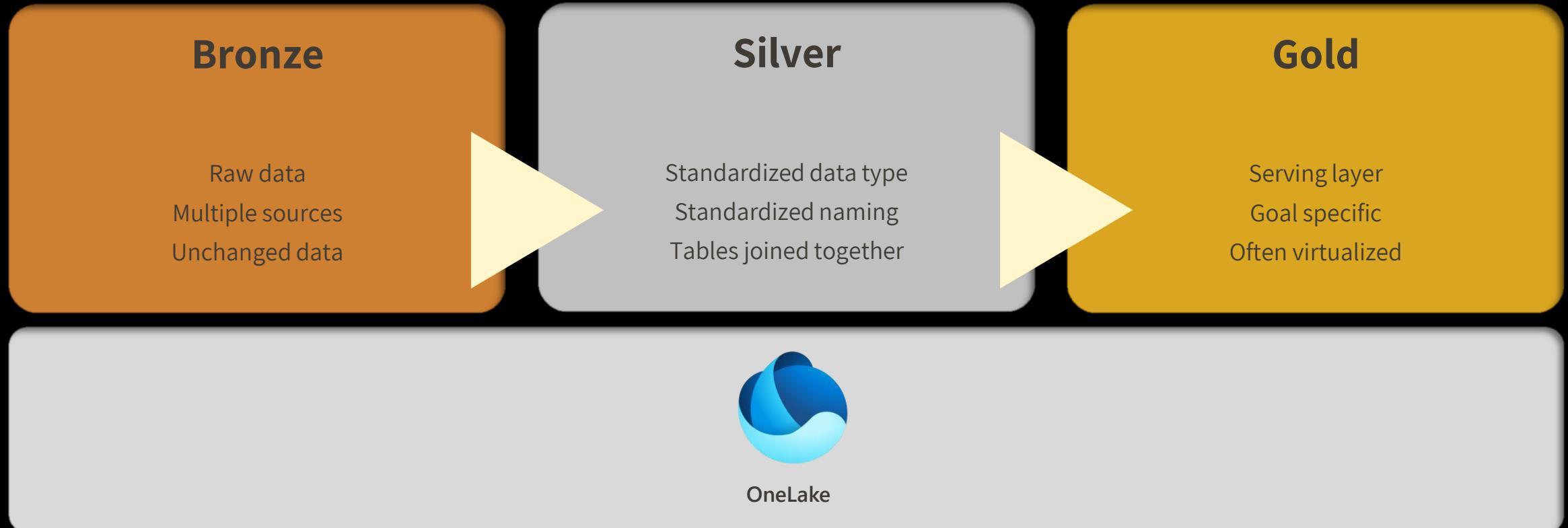


Data transformations should be done as far upstream as possible

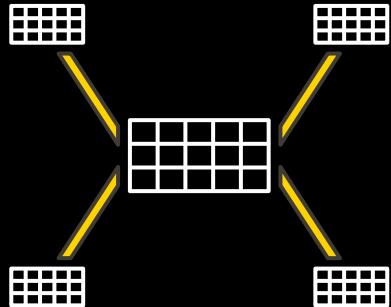


Data transformation directly in the Lakehouse unlocks “new” possibilities

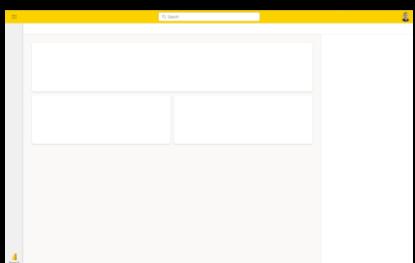
Medallion architecture



Data modeling



General best practice to have a star schema still applies



Web (browser) experience only to develop new data models

You can edit from Desktop – though desktop acts like a browser

Demo





Internals & performance

Delta (Parquet)



Parquet (File format)

- Column-store
- Open industry standard
- Compressed & Encoded
- Parallelism

Enables fast bulk operations of large data volumes

Delta (Storage layer management)

- Also, an open industry standard
- ACID transactions & schema enforcement
- Delete, update, merge
- Time-travel
- Optimized for querying, skipping and pruning

Brings warehouse reliability to the data lake

From Z-order to V-order

Yellow taxi (3 Billion rows)



416 GB



164 GB



V-order
60GB

x3.2
Less I/O for all ★
workloads

Data saving

The Analysis Services column-oriented storage using Delta Lake/Parquet open standard for Direct Lake

SSAS, AAS, Power BI large models

C:\Program Files\Microsoft SQL Server\MSAS15.MSSQLSERVER\OLAP\Data\AdventureWorksTabular.0.db\DimCustomer (10).tbl\238.prt

Name	Date modified	Type	Size
1.DimCustomer (10).AddressLine1 (78).0.idf	1/29/2020 6:36 PM	IDF File	37 KB
1.DimCustomer (10).AddressLine1 (78).0.idfmeta	1/29/2020 6:36 PM	IDFMETA File	1 KB
1.DimCustomer (10).AddressLine2 (79).0.idf	1/29/2020 6:36 PM		
1.DimCustomer (10).AddressLine2 (79).0.idfmeta	1/29/2020 6:36 PM		
1.DimCustomer (10).BirthDate (66).0.idf	1/29/2020 6:36 PM		
1.DimCustomer (10).BirthDate (66).0.idfmeta	1/29/2020 6:36 PM		
1.DimCustomer (10).CommuteDistance (82).0.idf	1/29/2020 6:36 PM		
1.DimCustomer (10).CommuteDistance (82).0.idfmeta	1/29/2020 6:36 PM		

Finance

Lake view Table view

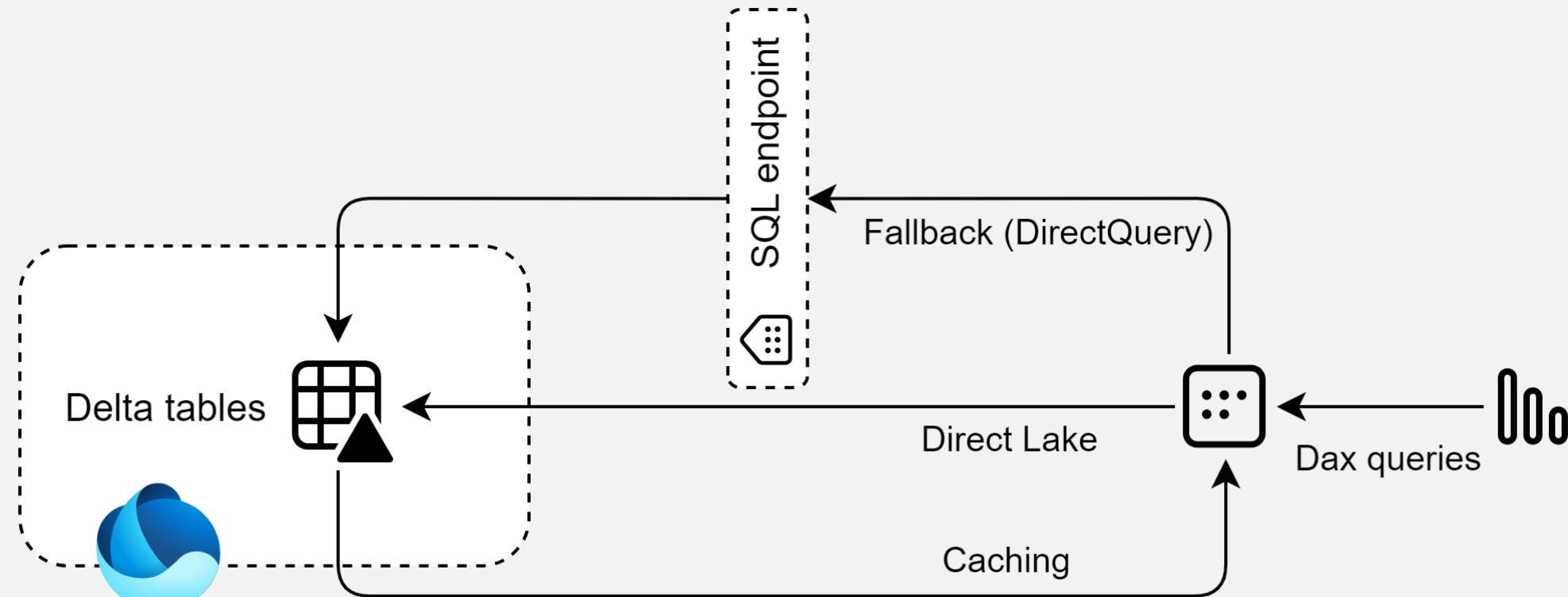
Tables

- > ACR Adjustment
- > Adjustment Type
- > Budget
- > Business
- > Calendar
- > Consumed Revenue
- > Forecast
- > Forecast Type
- > Future Flag
- > SSMO

Tables > Revenue

Name ↑	Size	Type
_delta_log	2 items	Folder
part-00000-87858576-90b7-4aff-8c9e-69dcc52db1	8.4 GB	PARQUET
part-00001-631fb085-0591-46b8-a0b5-0fec8f2255	8.4 GB	PARQUET
part-00002-0469bb29-daa5-4ecd-a3ee-bb90331a6	8.4 GB	PARQUET
part-00003-27e6062b-4d55-4469-b285-7cb2a2f32	8.4 GB	PARQUET
part-00004-b12eea8e-f255-41fa-a943-a78def57ce	8.4 GB	PARQUET

Fallback & Caching



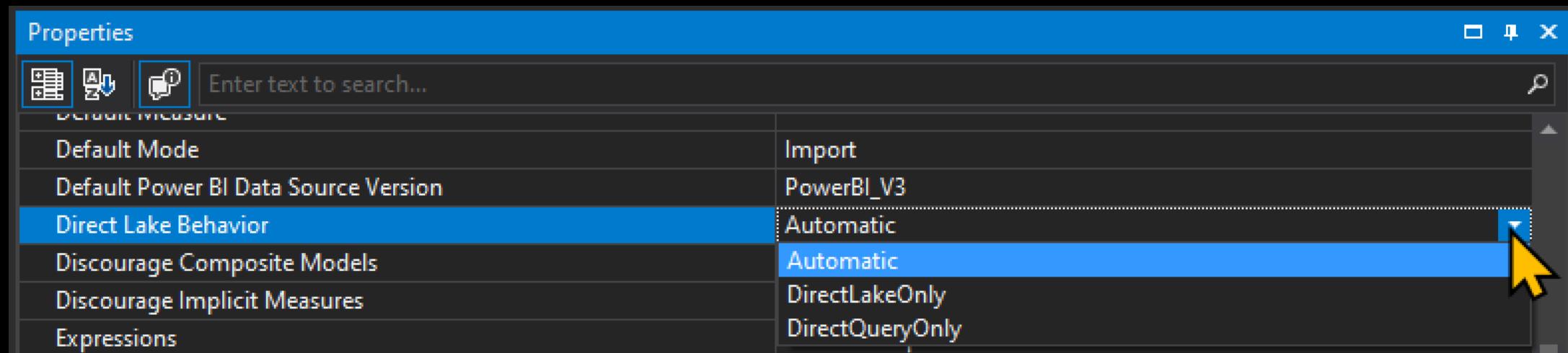
Fallback

When could fallback to DirectQuery happen?

- Special data types
- Large data volumes that does not fit the capacity size
- Composite models
- When you manually configure security
Item level on lakehouse

Disable fallback

Using Tabular Editor (or any other tool using XMLA) you influence Direct Lake behavior. Decide if it is allowed to fall back to DirectQuery or not.

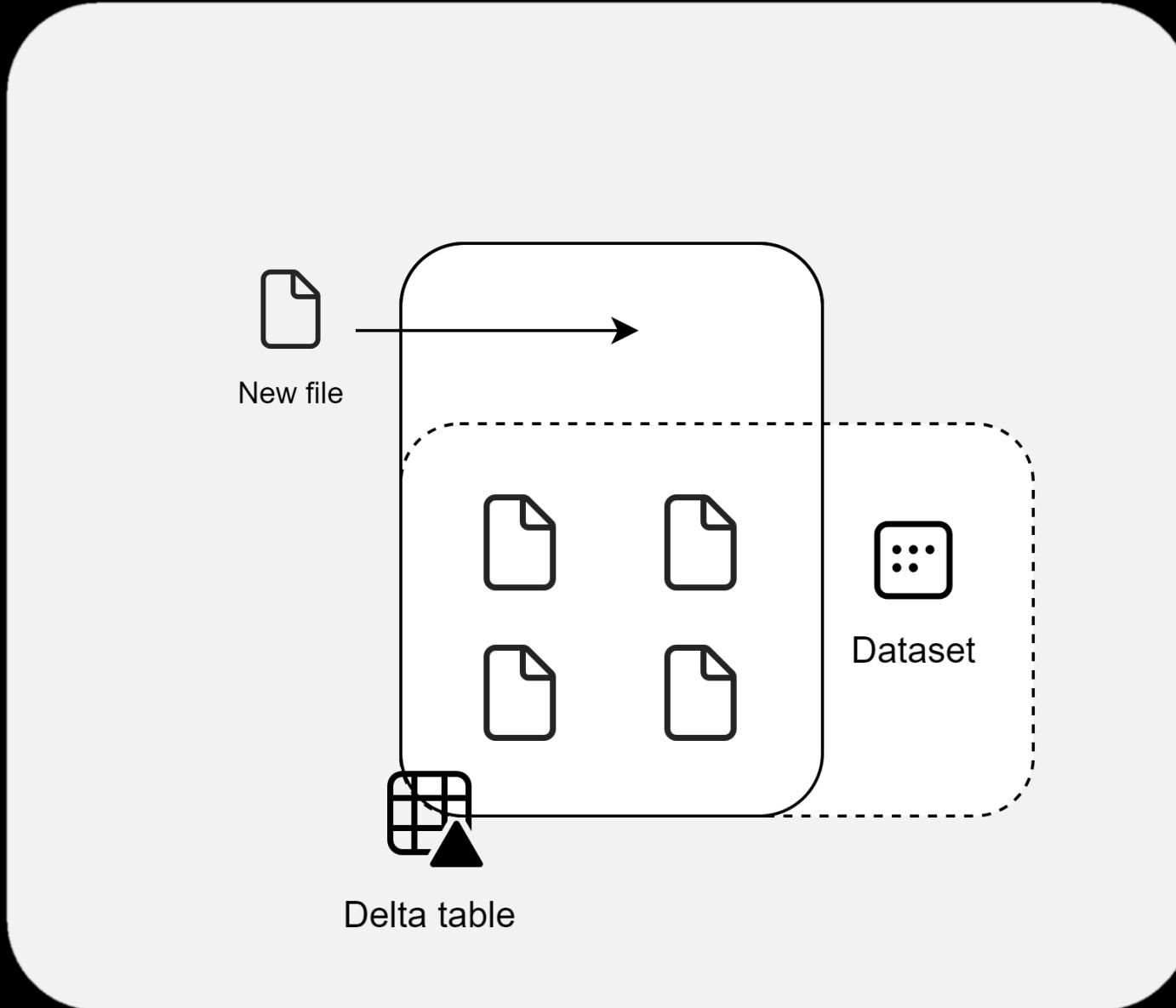


Introducing Framing

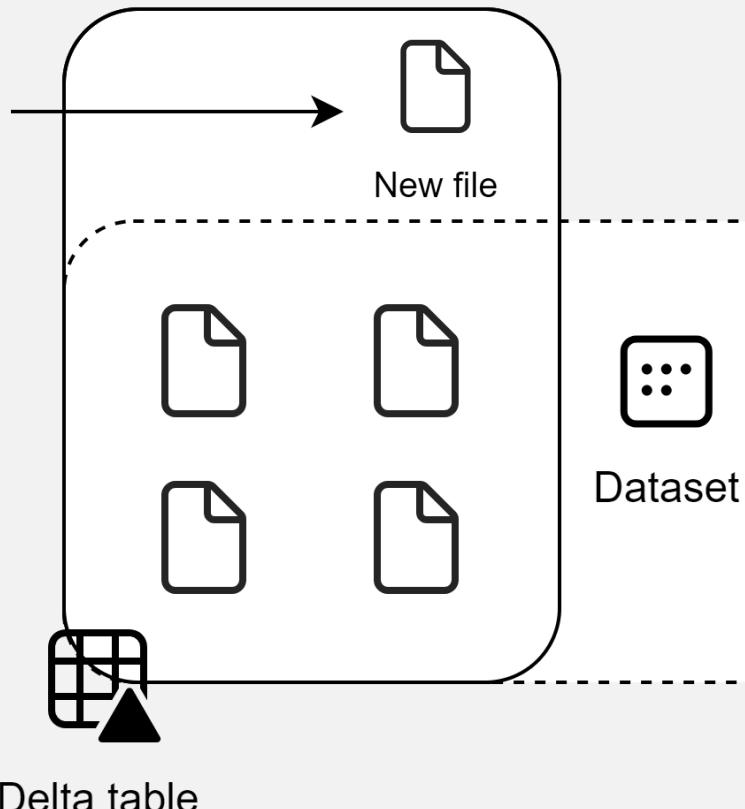
Metadata **refresh** which does not actually load the data, but only the delta table definitions.



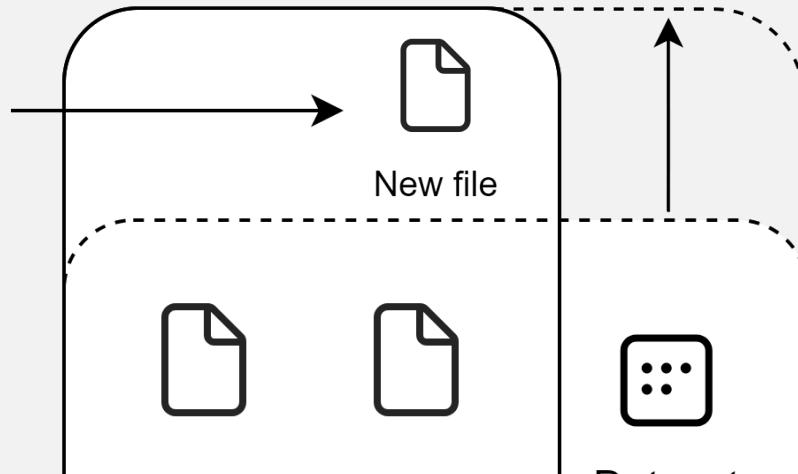
Framing



Framing



Framing



Refresh

Keep your Direct Lake data up to date

Configure Power BI to detect changes to the data in OneLake and automatically update the Direct Lake tables that are included in this dataset. [Learn more](#)



Off

Automatic update (Refresh)

- Meta data updates
- Frequency of data updates (framing)
- Capacity impact with regards to caching + flushes the cache (partly)
- Date Integrity – keep data in sync (Fact updated but Dim not etc)

▫ Refresh

Keep your Direct Lake data up to date

Configure Power BI to detect changes to the data in OneLake and automatically update the Direct Lake tables that are included in this dataset. [Learn more](#)

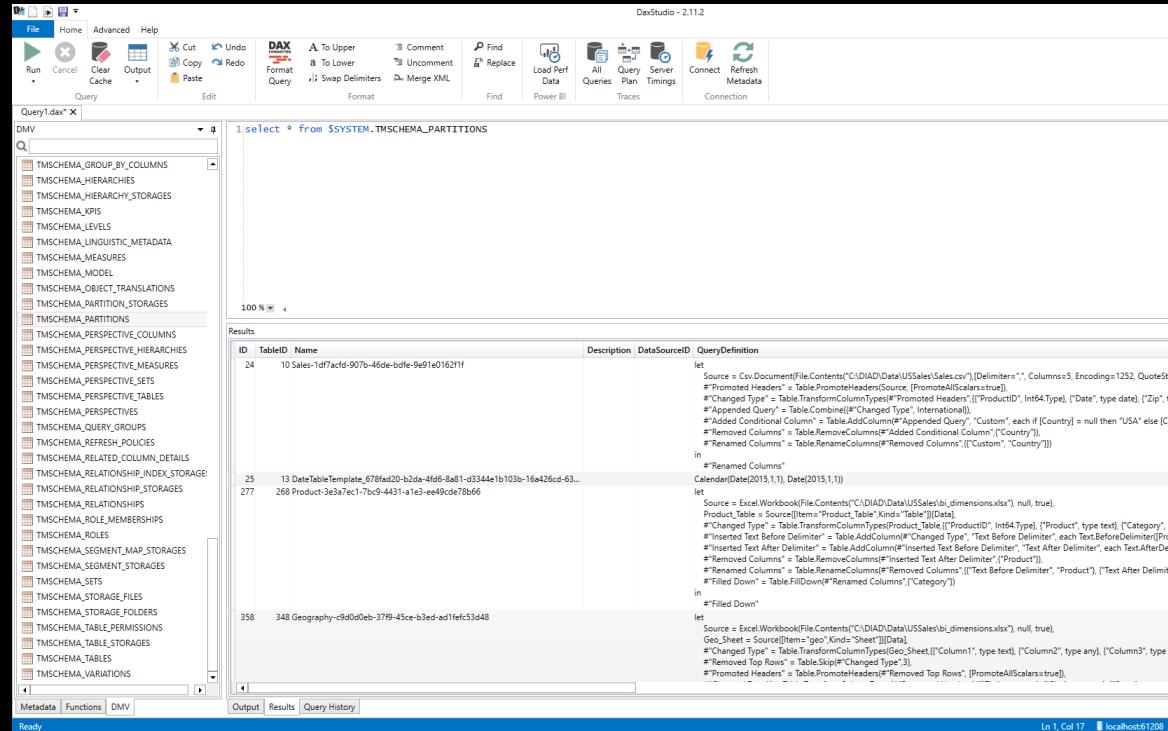


Off

Dynamic Management Views

Analysis Services Dynamic Management Views (DMVs) are queries that return information about model objects, server operations, and server health.

- DB Schema = Database model
- DISCOVER = Operations & Sessions
- TM Schema = Tabular = Power BI / AAS
- MD Schema = MDX = Multidimensional



The screenshot shows the DaxStudio interface with a query window open. The query is:

```
1:select * from $SYSTEM.TMSCHEMA_PARTITIONS
```

The results pane displays a table with three rows of data:

ID	TableID	Name	Description	DataSourceID	QueryDefinition
24	10	Sales-1d7acf-907b-46de-bdfc-9e91e0162f1f			let Source = Csv.Document(File.Contents("C:\DIAD\Data\USSales\Sales.csv"),[Delimiters=","], Columns=5, Encoding=1252, QuoteStyle="None");#PivotTable Headers = Table.TransformColumnTypes(Source, [Header=1, MaxColumnWidth=100, MaxRows=100, UseFirstRowAsHeaders=true]);#Changed Row = Table.AddRowColumn(#"PivotTable Headers", "Sales", Int64.Type);#Appended Query = Table.Combine({#"Changed Row", {"ProductID", Int64.Type}, {"Date", type date}, {"Zip", type string}, {"Country", type string}});#Added Conditional Column = Table.AddColumn(#"Appended Query", "Custom", each if [Country] = null then "USA" else [Country]);#Removed Columns = Table.RemoveColumns(#"Added Conditional Column",{"Country"});#Renamed Columns = Table.RenameColumns(#"Removed Columns",{{"Custom", "Country"}});in #Renamed Columns;Calendar(Date(2015,1,1), Date(2015,1,1))
25	13	DateTableTemplate_678fed20-b2da-4fd6-8a81-d3344e1b103b-16a426cd-63...			#Filled Down
277	268	Product-3e3a7ec1-7bc9-4431-a1e3-eed49cde78b66			#Filled Down

The QueryDefinition pane shows the DAX code used to generate the results:

```
let
    Source = Csv.Document(File.Contents("C:\DIAD\Data\USSales\Sales.csv"),[Delimiters=","], Columns=5, Encoding=1252, QuoteStyle="None");
    #PivotTable Headers = Table.TransformColumnTypes(Source, [Header=1, MaxColumnWidth=100, MaxRows=100, UseFirstRowAsHeaders=true]);
    #Changed Row = Table.AddRowColumn(#"PivotTable Headers", "Sales", Int64.Type);
    #Appended Query = Table.Combine({#"Changed Row", {"ProductID", Int64.Type}, {"Date", type date}, {"Zip", type string}, {"Country", type string}});
    #Added Conditional Column = Table.AddColumn(#"Appended Query", "Custom", each if [Country] = null then "USA" else [Country]);
    #Removed Columns = Table.RemoveColumns(#"Added Conditional Column",{"Country"});
    #Renamed Columns = Table.RenameColumns(#"Removed Columns",{{"Custom", "Country"}});
in #Renamed Columns;
Calendar(Date(2015,1,1), Date(2015,1,1))
let
    Source = Excel.Workbook(File.Contents("C:\DIAD\Data\USSales\bi_dimensions.xlsx"), null, true),
    Product_Table = Source[[Items["Product_Table",Kind="Table"]]];
#Filled Down
#Inserted Text Before Delimiter = Table.AddColumn(#"Changed Type", "Text Before Delimiter", each Text.BeforeDelimiter([Product_ID], Int64.Type));
#Inserted Text After Delimiter = Table.AddColumn(#"Inserted Text Before Delimiter", "Text After Delimiter", each Text.AfterDelimiter([Product_ID], Int64.Type));
#Removed Columns = Table.RemoveColumns(#"Inserted Text After Delimiter", {"Product_ID"});
#Renamed Columns = Table.RenameColumns(#"Removed Columns", {"Text Before Delimiter", "Product"}, {"Text After Delimiter", "Category"});
#Filled Down = Table.FillDown(#"Renamed Columns", {"Category"});
in #Filled Down
let
    Source = Excel.Workbook(File.Contents("C:\DIAD\Data\USSales\bi_dimensions.xlsx"), null, true),
    Geography_Sheet = Source[[Geography_Sheet["Geography_Sheet"]]];
#Changed Row = Table.AddRowColumn(#"Geography_Sheet", "Column1", Int64.Type);
#Removed Row = Table.Skip(#"Changed Type", 3);
#Promoted Headers = Table.PromoteHeaders(#"Removed Top Rows", [PromoteAllScalars=true]);
```

Demo



But wait...

Previously, all your data got evicted from cache after reframing.

But now;

- Data does not necessarily get evicted, if the segments in which the data was loaded did not change at the source.
- Behavior dependent on row groups (partitioning) of delta tables.

Forced eviction can be triggered by ProcessClear + ProcessFull commands.

Demo



Optimization

A good model – is key to get the most out of Direct Lake!

So, Semantic Link (Labs) will be your best friend to optimize your models!



run_model_bpa

Run Best Practice Analyzer scan for a semantic model, and display an HTML visualization, or return a pandas dataframe.

Python

```
run_model_bpa(dataset: str | UUID  
'table', 'zip', 'none') = 'html'
```

`smpy_labs.directlake.show_unsupported_direct_lake_objects(dataset: str | UUID, workspace: str | UUID | None = None) → Tuple[DataFrame, DataFrame, DataFrame]`

Returns a list of a semantic model's objects which are not supported by Direct Lake based on [official documentation](#).

Parameters: • `dataset (str | uuid.UUID)` – Name or ID of the semantic model.

• `workspace (str | uuid.UUID, default=None)` – The Fabric workspace name or ID. Defaults to None which resolves to the workspace of the attached lakehouse or if no lakehouse attached, resolves to the workspace of the notebook.

Returns: 3 pandas dataframes showing objects in a semantic model which are not supported by Direct Lake.

Return type: `pandas.DataFrame, pandas.DataFrame, pandas.DataFrame`



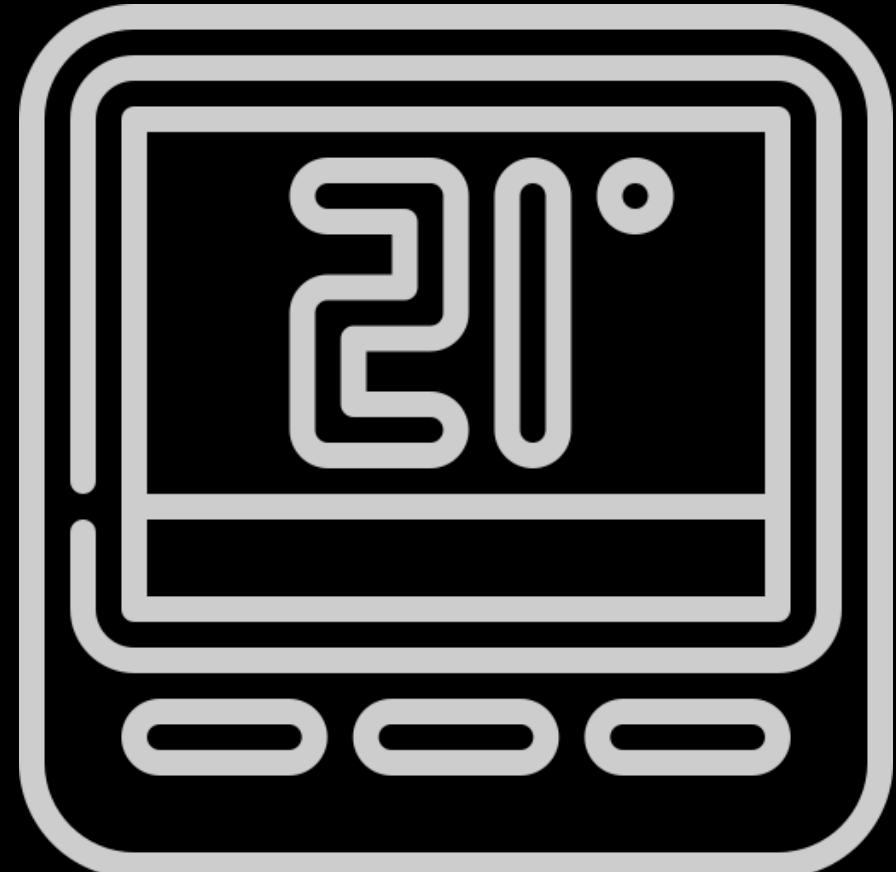
Advanced patterns

Temperature management

Keep it WARM!

Make sure your users are served optimally
and avoid the capacity memory to be flushed.

- Either due to reframing
- Or by eviction due to capacity utilization



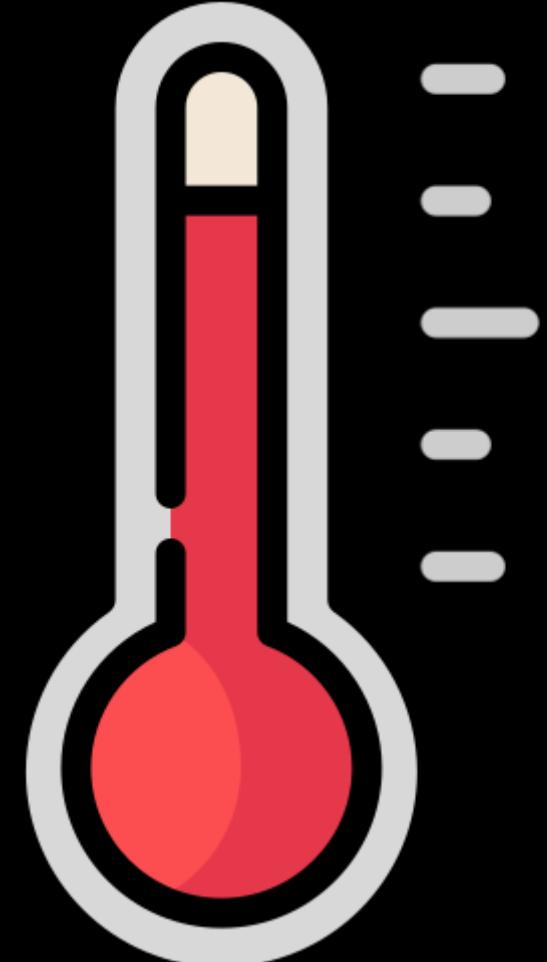
Temperature management

What will be evicted?

Basically, your data will be evicted from active memory, that you want to always have available!

How can you influence that?

Consider setting up a process (notebook, other automated setup) to pro-actively execute queries to keep certain data **WARM**!



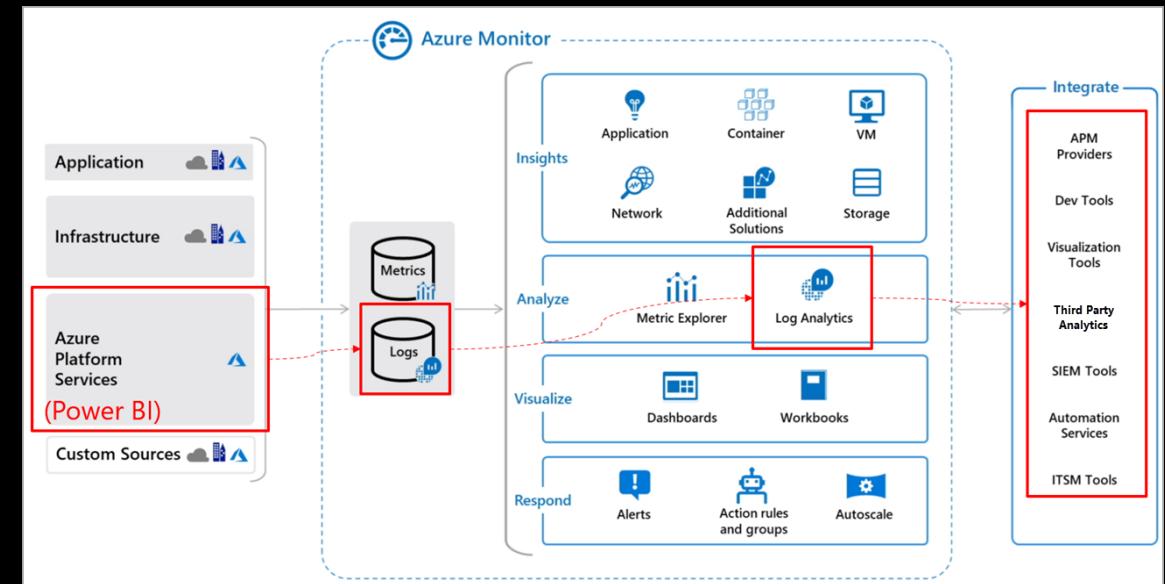
What should stay in memory?

Azure Monitor / Log Analytics

Or

Fabric Workspace Monitoring (preview)

Delivers a comprehensive solution for collecting, analyzing, and acting on telemetry from your cloud. It helps you understand how your applications are performing and proactively identifies issues affecting them and the resources they depend on.



Warm up your data using Semantic Link Labs

```
sempy_labs.directlake.warm_direct_lake_cache_isresident(dataset: str | UUID, workspace: str | UUID | None = None)→ DataFrame
```

Performs a refresh on the semantic model and puts the columns which were in memory prior to the refresh back into memory.

Parameters:

- `dataset` (`str | uuid.UUID`) – Name or ID of the semantic model.
- `workspace` (`str | uuid.UUID | None`) – The Fabric workspace name or ID. Defaults to None which resolves to the workspace of the attached lakehouse or if no lakehouse attached, resolves to the workspace of the notebook.

Returns:

Returns a pandas dataframe showing the columns that have been put into memory.

Return type:

pandas.DataFrame

```
sempy_labs.directlake.warm_direct_lake_cache_perspective(dataset: str | UUID, perspective: str, add_dependencies: bool = False, workspace: str | UUID | None = None)→ DataFrame
```

Warms the cache of a Direct Lake semantic model by running a simple DAX query against the columns in a perspective.

Parameters:

- `dataset` (`str | uuid.UUID`) – Name or ID of the semantic model.
- `perspective` (`str`) – Name of the perspective which contains objects to be used for warming the cache.
- `add_dependencies` (`bool, default=False`) – Includes object dependencies in the cache warming process.
- `workspace` (`str | uuid.UUID, default=None`) – The Fabric workspace name or ID. Defaults to None which resolves to the workspace of the attached lakehouse or if no lakehouse attached, resolves to the workspace of the notebook.

Returns:

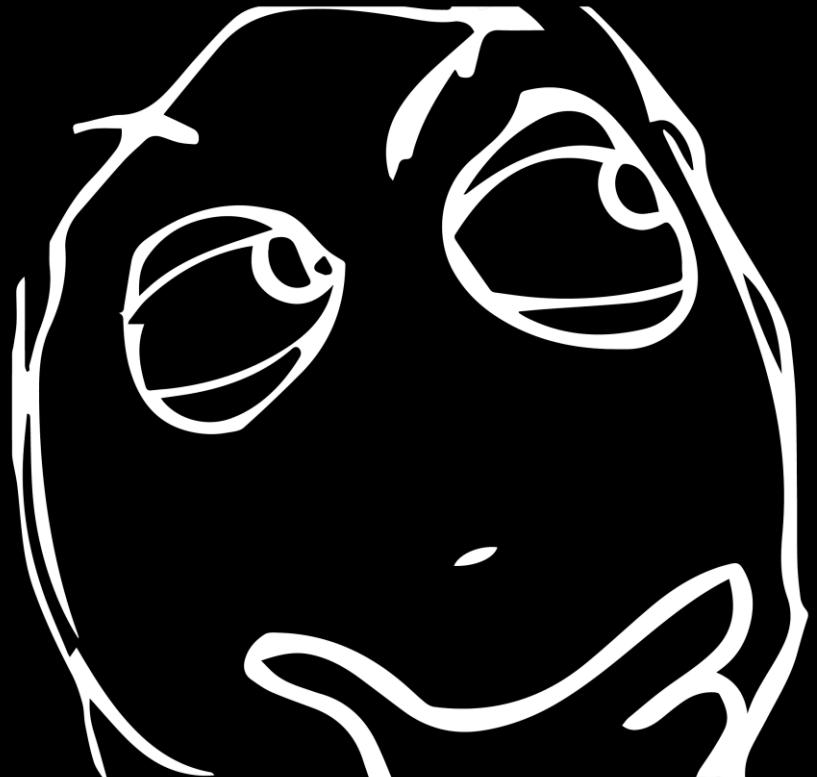
Returns a pandas dataframe showing the columns that have been put into memory.

Return type:

pandas.DataFrame



Considerations



IT DEPENDS

Should I change all my solutions to start using Direct Lake?

- Do you really need “real time” data?
- Consider impact on capacities when falling back to DQ
- Performance is better than DQ

Resources

Direct Lake generic documentation

<https://learn.microsoft.com/en-us/power-bi/enterprise/Direct-Lake-overview>

Analyze performance for Direct Lake

<https://learn.microsoft.com/en-us/power-bi/enterprise/directlake-analyze-qp>

On-demand loading of Direct Lake Power BI datasets in Fabric

<https://blog.crossjoin.co.uk/2023/07/02/on-demand-loading-of-direct-lake-power-bi-datasets-in-fabric/>

Direct Lake Frequently Asked Questions

<https://fabric.guru/power-bi-direct-lake-mode-frequently-asked-questions/>

Direct Lake framing & warm-up

<https://data-marc.com/exploring-direct-lake-framing-and-warm-up-data-using-semantic-link-in-fabric-notebooks/>

Wrap up

Direct Lake...

- Only applicable when using MS Fabric
- No data is imported / copied
- On-demand loading
- Reads data from Lake / Parquet format – Delta is a must
- Performance dependent on capacity size/utilization
- Falls back to DirectQuery when limitations are hit!

TELL US WHAT
YOU THINK!

