Microsoft

Direct Lake Workshop

Patrick LeBlanc, Phil Seamark

Azure Data

Agenda

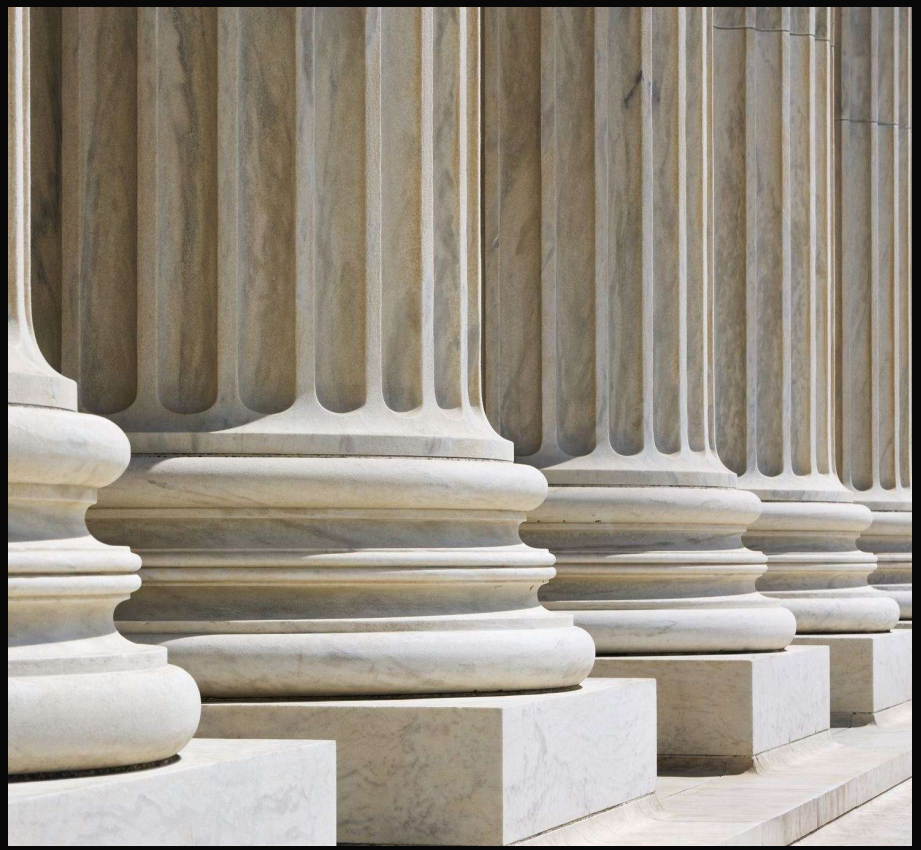
- What is Direct Lake
- Direct Lake Prerequisites
- Anatomy of Parquet
- V-Order
- Direct Lake in action (Demo)
- Direct Query Fallback
- Framing
- Security
- Performance

- aka.ms/FabConDL

- dldm.user1@fabricconf.onmicrosoft.com
- FabCon2025

What Is Direct Lake?

Three Pillars of data modelling



- Data Availability
- Model Size
- Query Speed

Storage Modes

SMALLER MODELS	Data Availability	Model Size	Query Speed
Direct Query	☺	☺	?
Import	☺	☺	☺

Storage Modes

SMALLER MODELS	Data Availability	Model Size	Query Speed
Direct Query	☺	☺	?
Import	☺	☺	☺

LARGER MODELS	Data Availability	Model Size	Query Speed
Direct Query	☺	☺	?
Import	😐	😐	☺

Storage Modes

SMALLER MODELS	Data Availability	Model Size	Query Speed
Direct Query	☺	☺	?
Import	☺	☺	☺
Direct Lake	☺	☺	☺

LARGE MODELS	Data Availability	Model Size	Query Speed
Direct Query	☺	☺	?
Import	😐	😐	☺
Direct Lake	☺	☺	☺

Fundamentals

- Only one data format can be used as a source for Direct Lake
- Direct Lake semantic model starts life with no data in memory
- Data gets *paged* into semantic model triggered by query
- Tables can mix resident and non-resident columns
- Column data can get evicted for multiple reasons
- Direct Lake may opt to use Direct Query to SQL endpoint
- "Framing" determines what data gets used by semantic model

Direct Lake limitations (for now)

- No calculated columns or calculated tables
- No composite models
  - Although calculation groups and field parameters are now allowed
- Can only be used with tables, not views
- Can only be used with security defined in the semantic model
- Not all data types supported
  - No structured data types, binary or GUID columns
  - Date Time relationships not supported
  - String length limited to 4000 characters

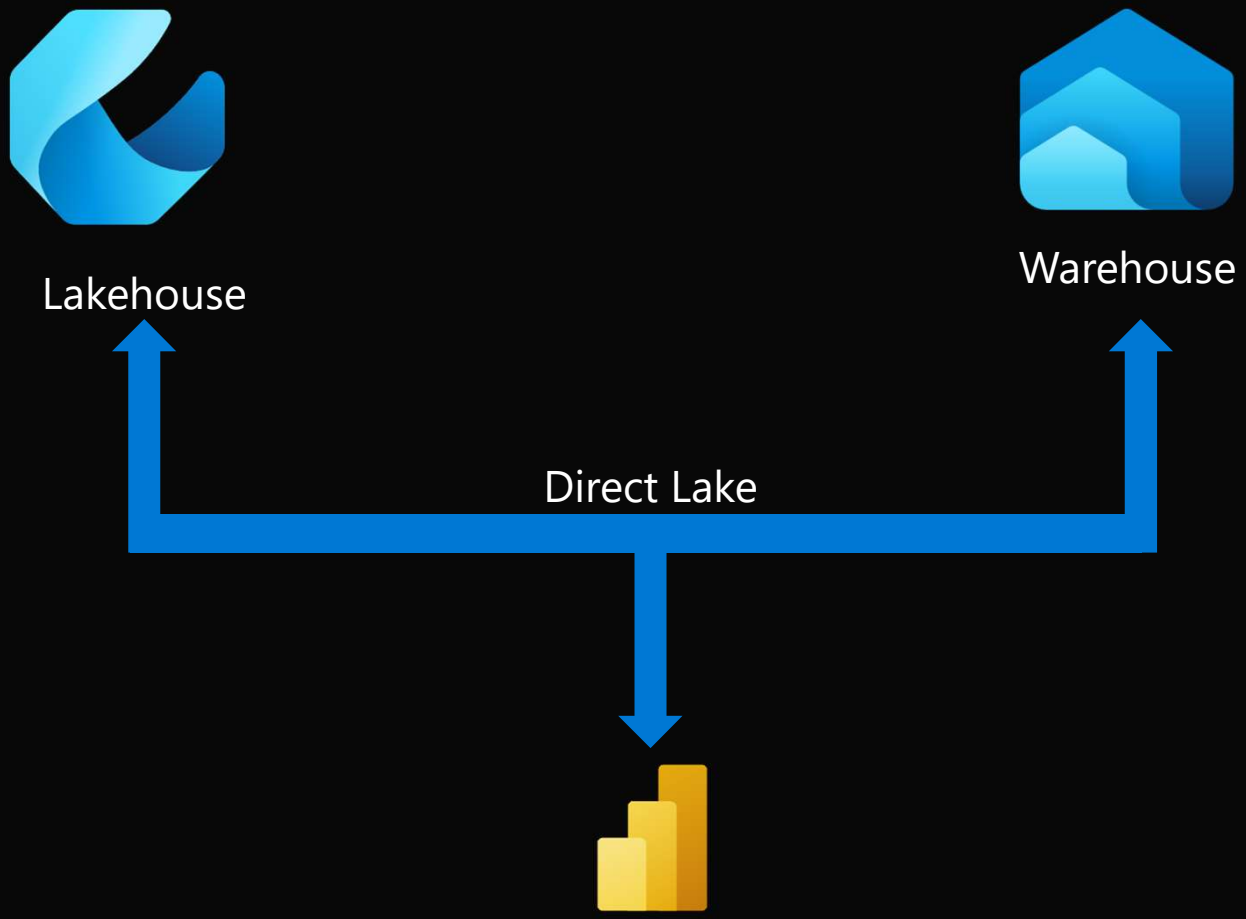
Direct Lake prerequisites

SKU Requirements

- Power BI Premium P
- Microsoft Fabric F SKUs only

- Not supported on:
  - Power BI Pro
  - Premium Per User
  - Power BI Embedded A/EM Skus

Direct Lake prerequisites



Why Parquet?

- Open source/open data format
- Column-oriented format is optimized for data storage and retrieval
- Efficient data compression and encoding especially data in bulk
- Is lingua franca for data storage format
  - Databricks, Microsoft - delta lake and parquet
  - Snowflake - iceberg and parquet/orc

Anatomy of a Parquet File

- CSV, XML, JSON…… Parquet

```
StoreID , DateTime   , ProductID , Value
StoreA  , 2023-01-01 , SKU001    , 10
StoreA  , 2023-01-02 , SKU001    , 15
StoreA  , 2023-01-03 , SKU001    , 12
```

```
<sale>
        <StoreID>StoreA</StoreID>
        <DateTime>2023-01-01</DateTime>
        <ProductID>SKU001</ProductID>
        <Value>10</Value>
</sales>
<sale> … </sale>
```

```
{sales[
        {
        StoreID: “StoreA” ,
        DateTime: “2023-01-01” ,
        ProductID: “SKU001” ,
        Value:10
        },
        {…}
]}
```

Anatomy of a Parquet File

CSV, XML, JSON…… **Parquet**

```
Header:
RowGroup1:
  StoreID   :     StoreA, StoreA, StoreA
  DateTime  :     2023-01-01, 2023-01-02, 2023-01-03
  ProductID:      SKU001, SKU001, SKU001
  Value     :     10, 15, 12
RowGroup2:
  ….
Footer:
```

Anatomy of a Parquet File – Dictionary IDs

- CSV, XML, JSON…… **Parquet**

```
Header:
RowGroup1:
   StoreID   :   1, 1, 1
   DateTime  :   1, 2, 3
   ProductID :   1, 1, 1
   Value     :   1, 2, 3
RowGroup2:
   ….
Footer:
```

Lab 1

Build a Model

V-Order

- V-Order is a Microsoft-proprietary optimisation for writing data in parquet files (as used in Delta tables)
- V-Order uses the same algorithms used by Power BI Import mode semantic models to compress data
- V-Ordered Delta tables can be read by any tool that can read Delta
- Direct Lake will perform better on V-Ordered Delta tables
- Direct Lake will work on all Delta tables, even without V-Order

V-Order

x3.2

Lab 2

Build a *bigger* Model

Delta Analyzer

- Built into Semantic Link Labs
- Notebook script to load to any Fabric workspace
- Basic script to run per Detla Table
- Outputs useful info about Delta table

Delta Analyzer

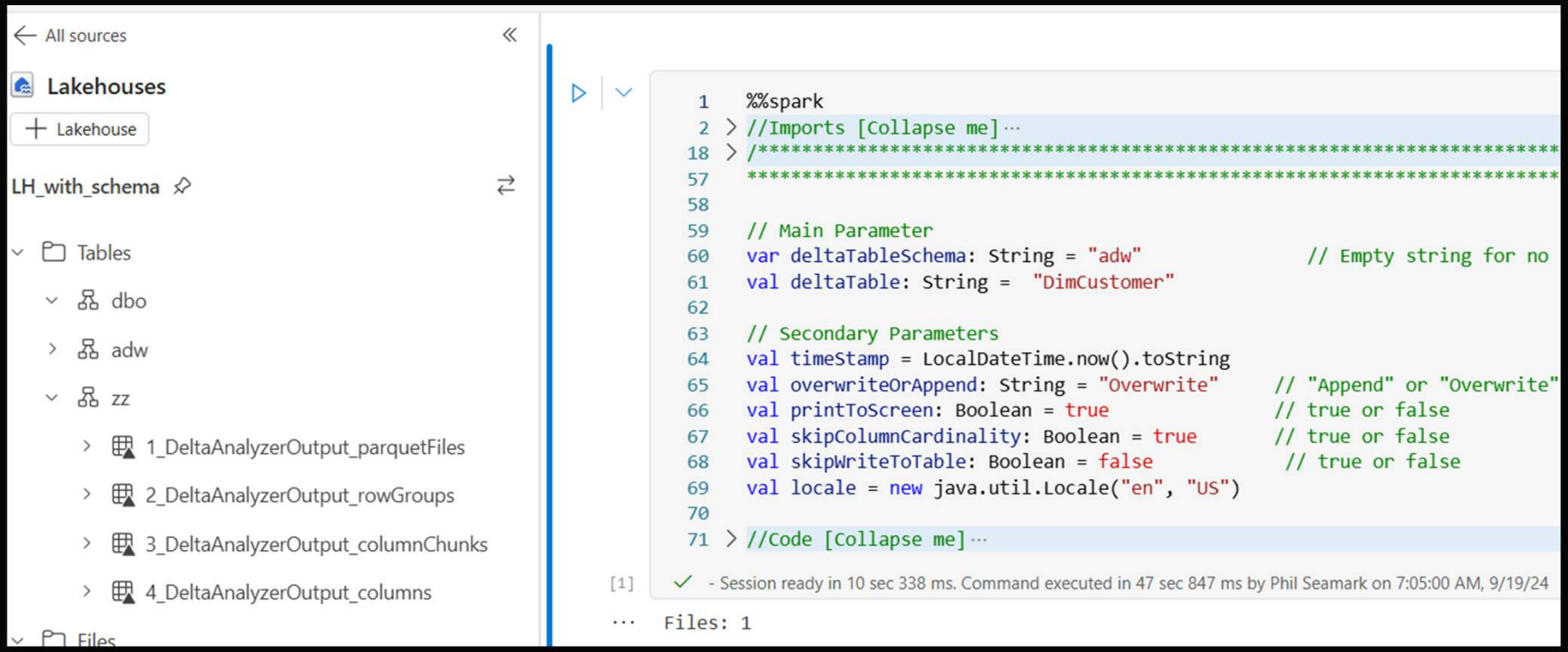
- Helps to size potential semantic model
- Can see if DQ Fallback might be close
- Helps identify potential columns slow to paging
- Works over shortcuts

Delta Analyzer

- Provides four outputs

- 1. One row per parquet file
- 2. One row per rowgroup
- 3. One row per column-chunk
- 4. One row per column (dcount etc.)

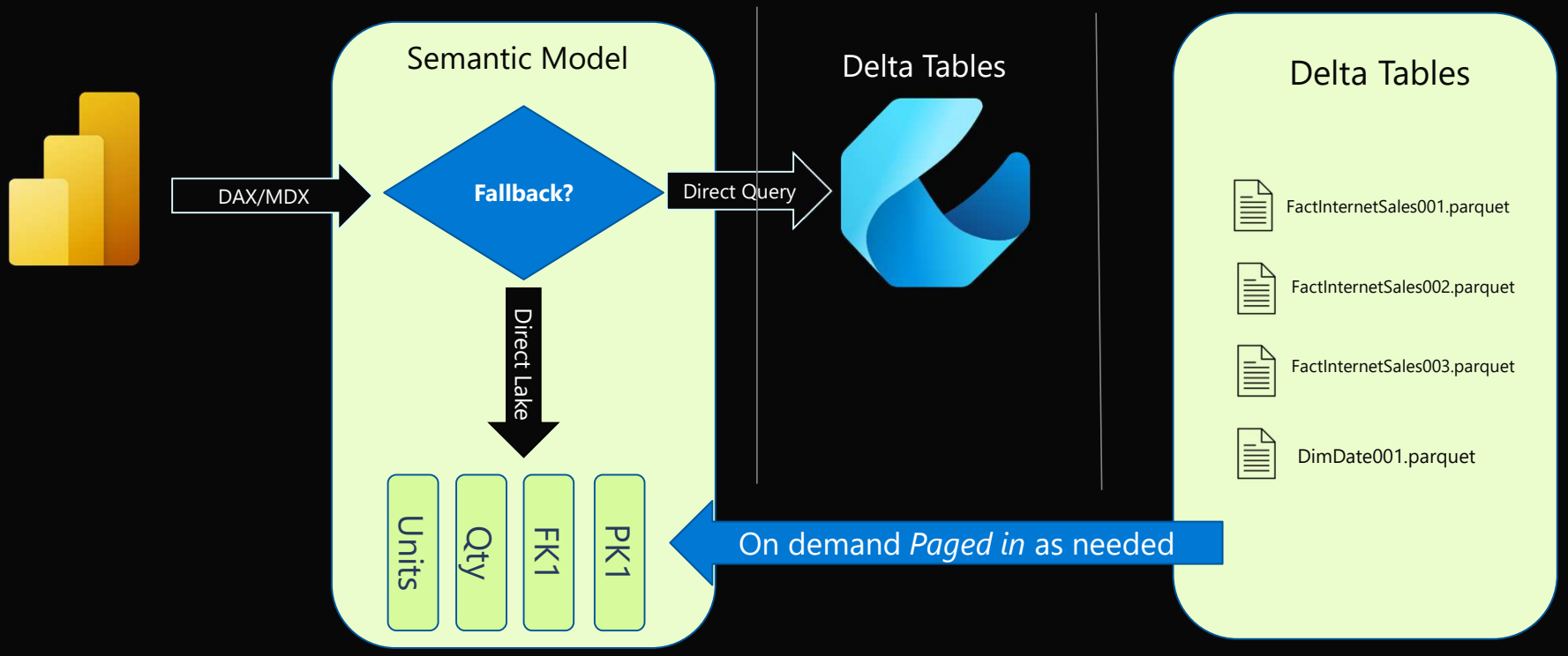
Delta Analyzer



Lab 3

Delta Analyzer

DAX to SQL Fallback



Fallback to DirectQuery – metadata

- You are using features that prevent Direct Lake
- Warehouse views are not allowed because they don’t have corresponding tables stored in a Lakehouse
- RLS or OLS is defined in a Warehouse
  - Security rules take high priority when defined

Fallback to DirectQuery – data volumes

- There are limits on how much data used for Direct Lake
- These limits vary by capacity SKU size
- If you exceed these limits, Direct Lake will use Direct Query
  - Query performance may be noticeably worse
- Fabric checks limits during reframing process
- Can be turned On/Off using Direct Lake Behaviour property

Fallback to DirectQuery - Current guardrails

Fabric/Power BI SKUs	Parquet files per table	Row groups per table	Rows per table (millions)	Max model size on disk/OneLake[1] (GB)	Max memory (GB)
F2	1,000	1,000	300	10	3
F4	1,000	1,000	300	10	3
F8	1,000	1,000	300	10	3
F16	1,000	1,000	300	20	5
F32	1,000	1,000	300	40	10
F64/FT1/P1	5,000	5,000	1,500	Unlimited	25
F128/P2	5,000	5,000	3,000	Unlimited	50
F256/P3	5,000	5,000	6,000	Unlimited	100
F512/P4	10,000	10,000	12,000	Unlimited	200
F1024/P5	10,000	10,000	24,000	Unlimited	400
F2048	10,000	10,000	24,000	Unlimited	400

Detecting fallback to DirectQuery

- Performance Analyzer, Profiler traces and/or Log Analytics will show what happens for individual queries
  - Direct Query End Event (SQL Fallback)
  - Vertipaq SE End Event (Direct Lake)
  - SQL vs SCAN in DAX Studio Server Timings
- Limits on data volumes can be checked with Python notebooks (Delta Analyzer) and in some cases DMVs

Controlling fallback to DirectQuery

- The **DirectLakeBehavior** property sets fallback behaviour
- Automatic (default): allows fallback to DirectQuery if data can’t be loaded into memory
- DirectLakeOnly: allows use of DirectLake but prevents fallback and returns an error instead of using DirectQuery
- DirectQueryOnly: forces all queries to use DirectQuery mode

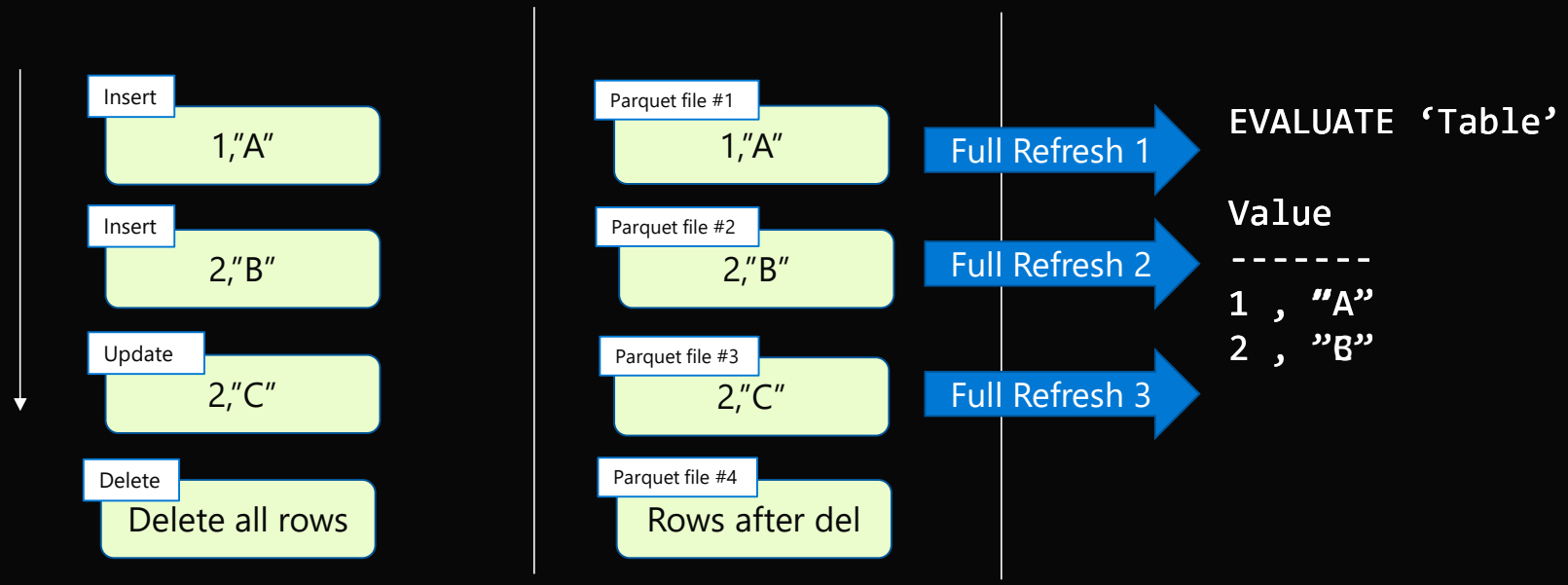
Lab 4

Direct Lake Fallback

Framing

- What is framing
  - "point in time" way of tracking what data can be queried by Direct Lake
- Why is this important
  - Data consistency for some Power BI Reports
  - Delta-lake data is transient for many reasons
- ETL Process
  - Ingest data to delta lake tables
  - Transform as needed using preferred tool
  - When ready, perform *Framing* operation on dataset
- Framing is near instant and acts like a cursor
  - Determines the set of .parquet files to use/ignore for *transcoding* operations

Framing



Framing - Options

- Automatic
  - Default - can be turned off
  - Triggered each time Delta table gets modified
- Via Fabric Service
  - Manually by refreshing the semantic model
  - Configure a schedule
- Via Notebook
  - Use Semantic-link to call reframe using native method
  - Execute_tmsl for fine grain reframing
  - Consider cache-warming as option

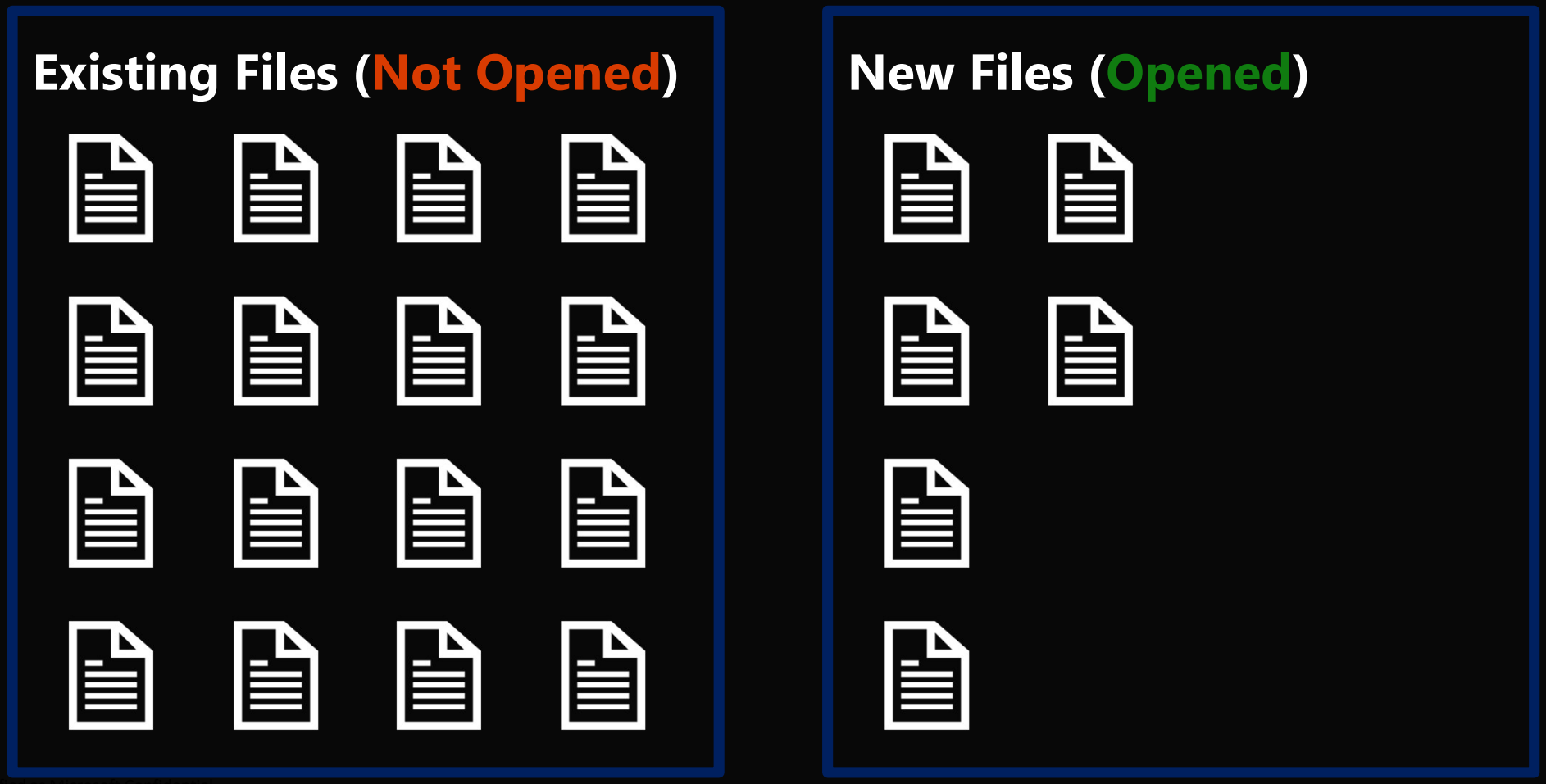
Manual Framing - Options

- SSMS (TMSL)
- Rest API
- Pipeline
- Notebooks (semantic-link)
- Power Automate etc.

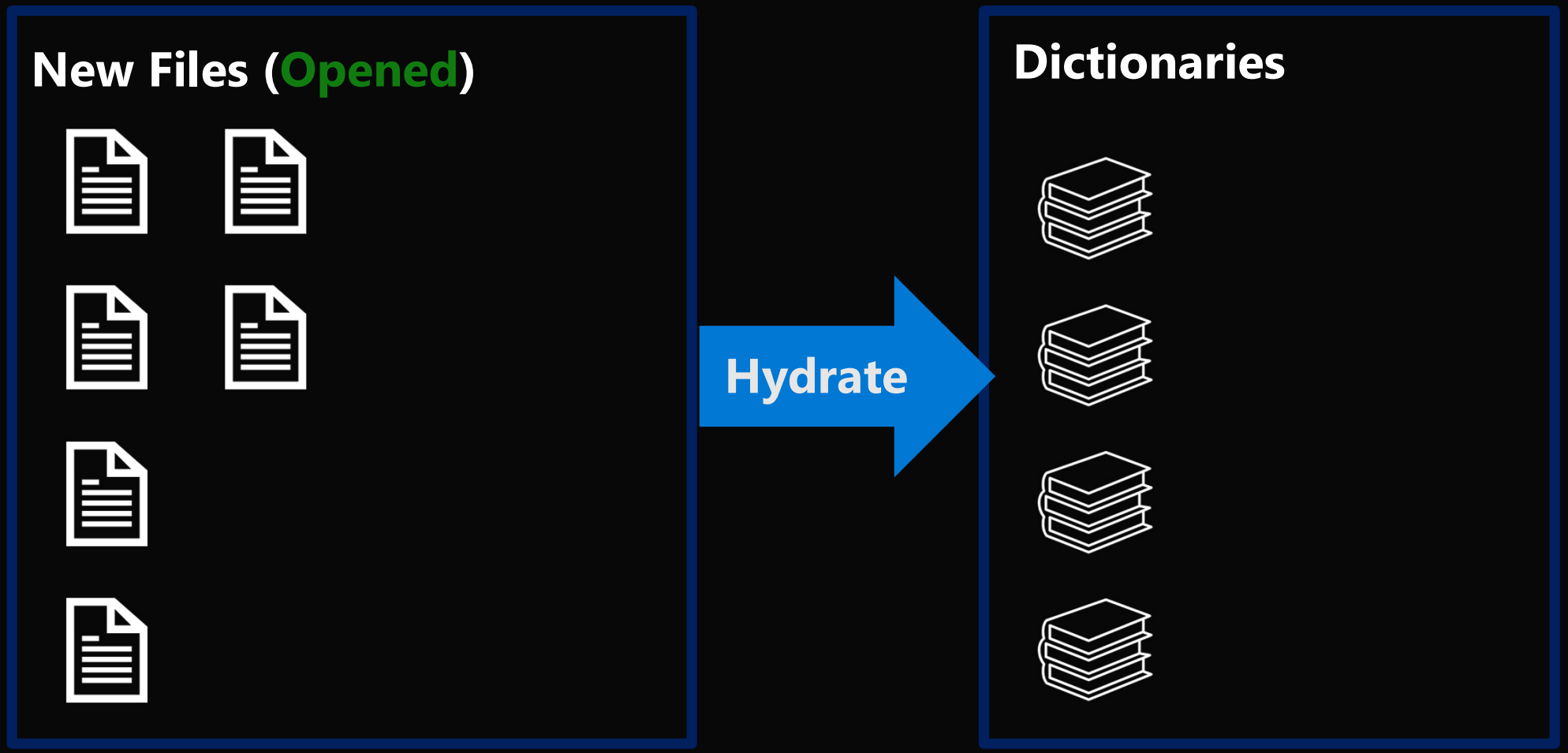
Lab 5

Framing

Reframing – Open New Files Only



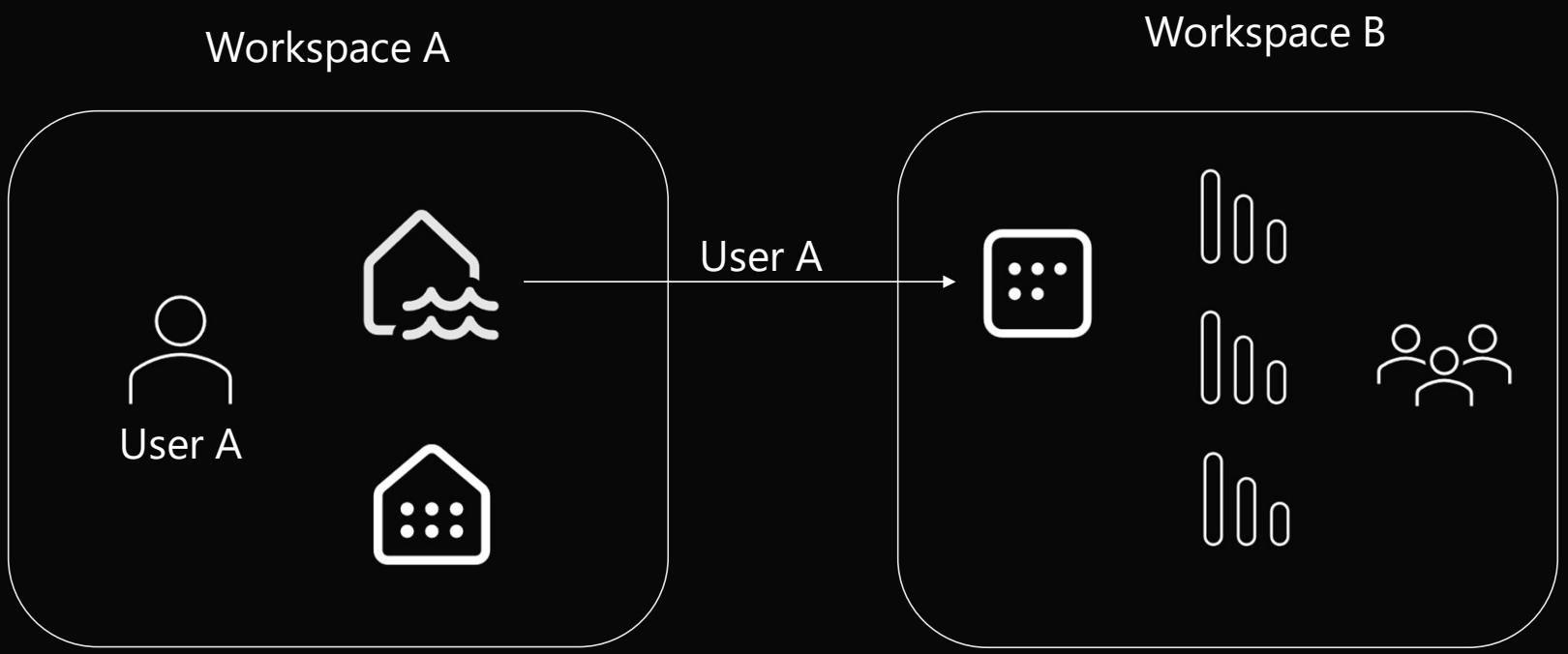
Reframing – Hydrate Dictionary Delta



Security options

- Security defined in the Semantic Model
  - Needs to be done per model
  - Does not auto force Direct Query Fallback
  - Recommend to use Fixed Identity to Lakehouse in different workspace
  - Can open door for viewer to skip security and use SQL Endpoint
  - RLS or OLS
- Security in the SQL Endpoint
  - Forces Direct Lake model to use DQ fallback per table
  - Applies to all clients who connect to SQL Endpoint (models, SSMS, other)
  - Need to apply RLS policies on ALL relevant tables in SQL Endpoint

Security options – RLS in Semantic Model



Demo

Security

Performance

Performance considerations

- Reframing – time taken to reframe a model
- Cold Cache – time to page data into model
- Warm Cache – DAX query speed once data in model
- *Optimising tips*

Performance considerations – Reframing

- Time to evict columns and load certain objects
- Loads Delta metadata and some metadata from parquet files

Performance considerations – Cold Cache

- The time needed to page data into a model from One Lake
- Number/layout of data across Parquet files
  - Optimise Parquet files/rowgroups
- Cache warming tricks
- High cardinality columns
  - Consider splitting
  - Review Datatype (avoid float/double e.g. **7.45** instead of **7.4523462734**)

Performance considerations – Cold Cache

- Run Vertipaq Analyzer
- Look for columns with highest cardinality/largest dictionary
- Schedule Notebook to run every 5 to 10 mins

```
EVALUATE
    {
        {COUNTROWS(VALUES('Table'[Column1]))},
        {COUNTROWS(VALUES('Table'[Column2]))},
        {COUNTROWS(VALUES('Table'[Column3]))}
    }
```

Performance considerations – Warm Cache

- Query Plans
  - Direct Lake Behaviour property
  - Other optimisations
- Encoding
  - All data is HASH encoded – no option to use VALUE encoding
- Segment data profile
  - Number and layout of data within segments can impact scan performance
  - Depends greatly on filters used per query

Performance numbers – sample model

- With V-Order
- No V-Order
- Column partitioned by Date
- V-Order and Z-Order

Performance - some numbers

	V-Order	No V-Order	Partitioned (V-Order)	Z-Order & V-Order
Rows	1,000,000,000	1,000,000,000	1,000,000,000	1,000,000,000
Columns	10	10	10	10
V-Order	TRUE		TRUE	TRUE
Z-Order				DateKey
Parquet Size	7.1GB	11.6GB	8.4GB	6.9GB
Files	14	200	807	6
Row Groups	26	200	807	24
Model Size				
Data	7.1GB	14.9GB	6.6GB	6.9GB
Total	9.6GB	17.5GB	9.1GB	9.4GB

Performance numbers – Cold cache

	V-Order	No V-Order	Partitioned (V-Order)	Z-Order & V-Order
Test 1	2m 24s	7m 35s	7m 16s	2m 18s
Test 2	2m 26s	8m 34s	7m 30s	2m 17s
Test 3	2m 27s	7m 46s	7m 27s	2m 18s

	V-Order	No V-Order	Partitioned (V-Order)	Z-Order & V-Order
Parquet Size	7.1GB	11.6GB	8.4GB	6.9GB
Files	14	200	807	6
Row Groups	26	200	807	24
Model Size				
Data	7.1GB	14.9GB	6.6GB	6.9GB
Total	9.6GB	17.5GB	9.1GB	9.4GB

Performance numbers – Warm Cache

	V-Order	No V-Order	Partitioned (V-Order)	Z-Order & V-Order
Query 1				
Total Time	223	863	47	203
FE CPU	1,449	12,840	550	1,148
Query 2				
Total Time	1,594	2,891	94	1,379
FE CPU	11,890	39,125	1,070	10,754
Query 3				
Total Time	4,817	4,129	5,851	10,845
FE CPU	29,937	43,933	42,523	18,867

Lab 6 & 7

Performance

Summary

- Will my reports run faster with Direct Lake?
- Do I *have* to use Direct Lake with Fabric?
- Incremental Refresh?
- Aggregations?

Announcements

- Direct Lake over One Lake
- Create Direct Lake reports using Power BI Desktop
- Composite Models (import + DL)
- Other…..

Questions

Microsoft

Appendix