Using Lakehouse data at scale with Power BI. Featuring Power BI Direct Lake mode!

Stijn Wynants
Benni De Jagere

Slides

CAT

Premium Sponsors

snowflake

twoday
kapacity

ITPA
CONSULTING

Microsoft

data on
Power your data

FRANK SLOOTMAN
AMP IT UP
Leading for Hypergrowth by Raising Expectations, Increasing Urgency, and Elevating Intensity
WILEY

Raffle Prizes

Standard Sponsors

TIMEXTENDER

[stoltze][it]

ORANGEMAN

exmon

KPMG

Fellowmind

Microsoft

Benni De Jagere

Senior Program Manager | Fabric Customer Advisory Team (FabricCAT)

Fabric CAT

dataMinds.be Member

@BenniDeJagere

/bennidejagere

sessionize /bennidejagere

/bennidejagere

#SayNoToPieCharts

Microsoft

Stijn Wynants

Senior Customer Engineer | FastTrack Engineering

FastTrack

dataMinds.be Member

@SQLStijn

/stijn-wynants-ba528660/

sessionize /Stijn-wynants

Fabric Espresso

#OneMore?

Disclaimer: We’re not benchmarking

Session Objectives

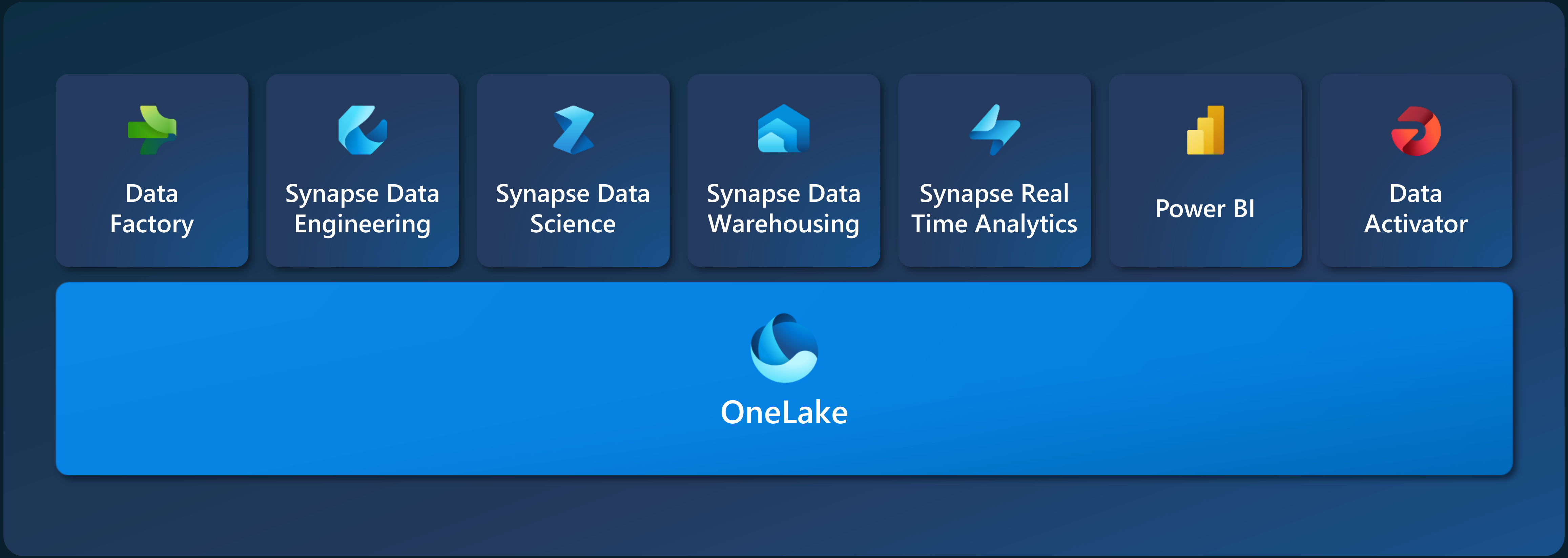
Session Objectives

- Introduce Fabric and OneLake
- Set the scene for Direct Lake
- Take it for spin.. ☺

Introducing Fabric

Microsoft Fabric

The unified data platform for the era of AI



Data Factory

Synapse Data Engineering

Synapse Data Science

Synapse Data Warehousing

Synapse Real Time Analytics

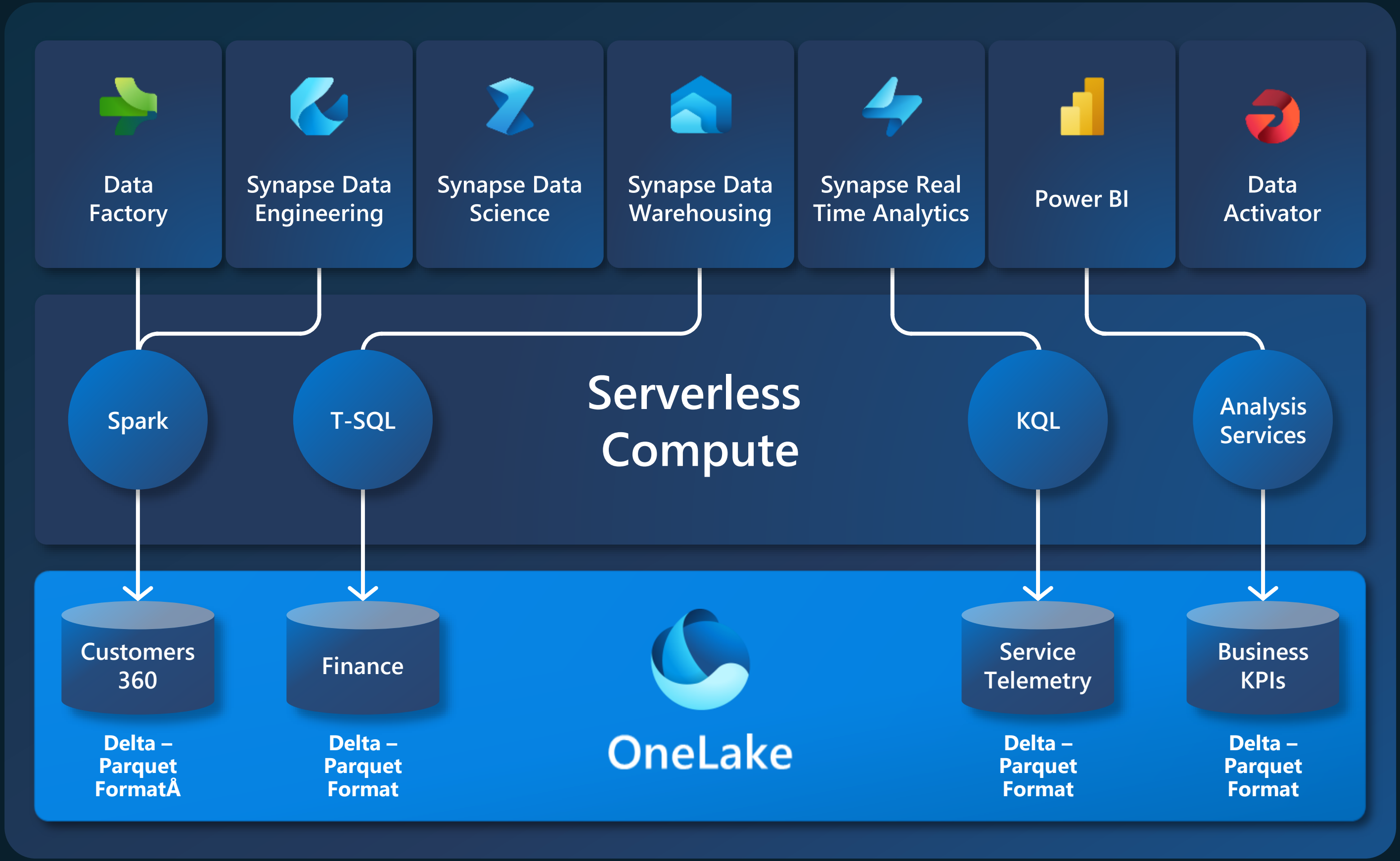
Power BI

Data Activator

OneLake

One Copy for all computes

Real separation of compute and storage



All the compute engines store their data automatically in OneLake

The data is stored in a single common format

Delta – Parquet, an open standards format, is the storage format for all tabular data in Analytics vNext

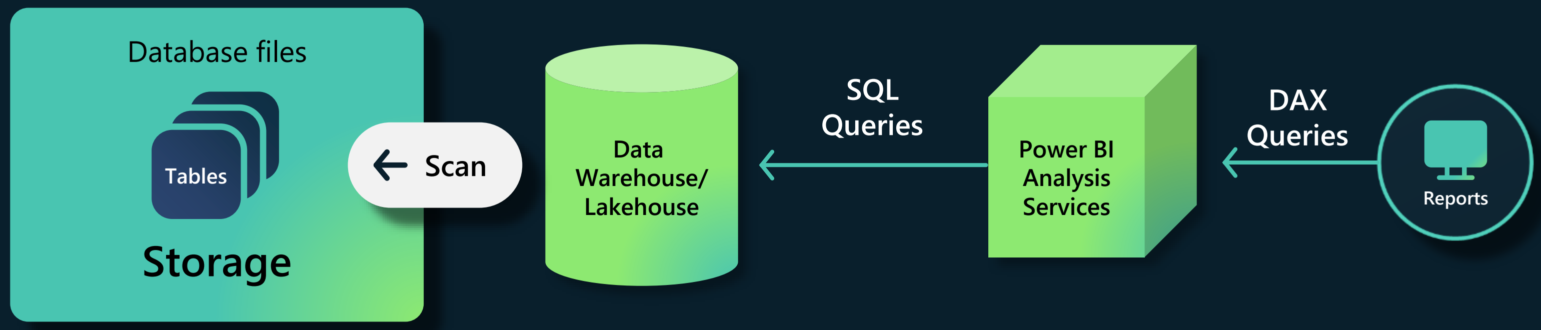
Once data is stored in the lake, it is directly accessible by all the engines without needing any import/export

All the compute engines have been fully optimized to work with Delta Parquet as their native format

Shared universal security model is enforced across all the engines

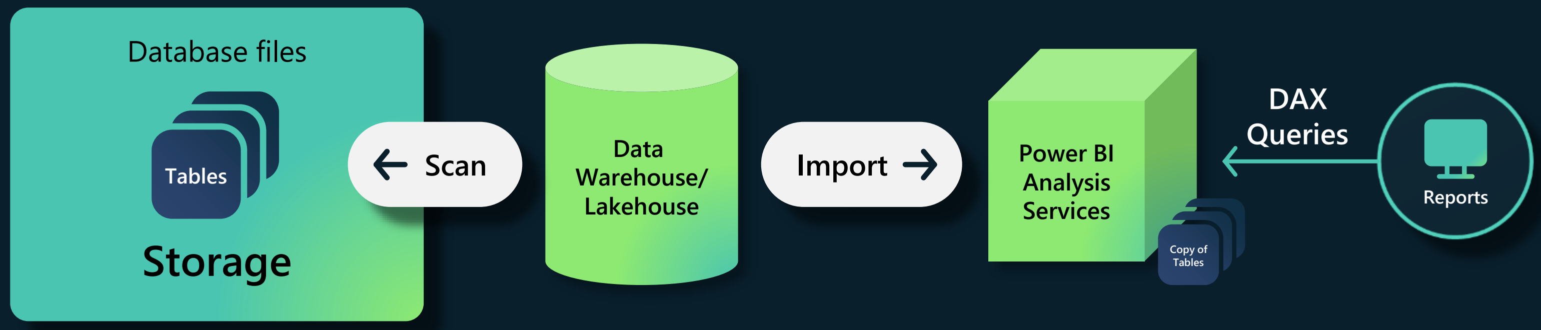
“Direct Query Mode”

Slow, but real time



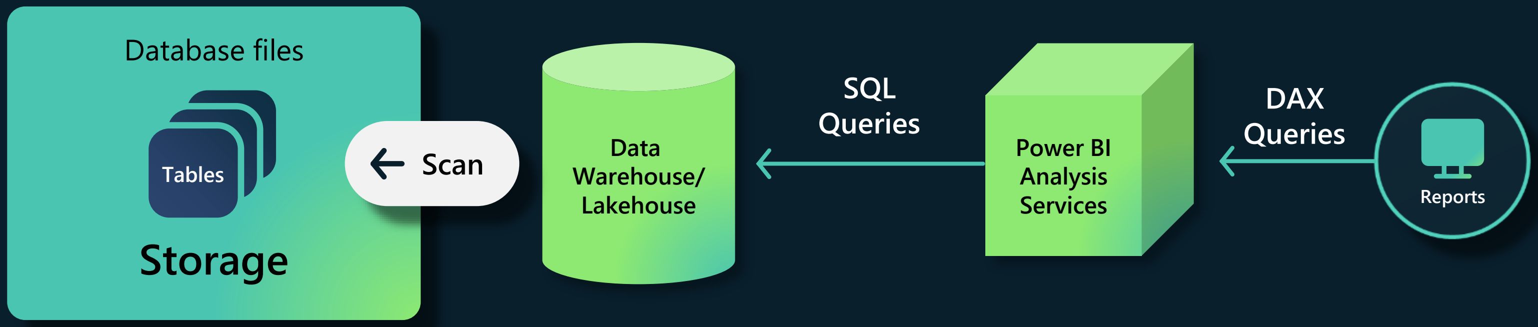
“Import Mode”

Latent & duplicative but fast



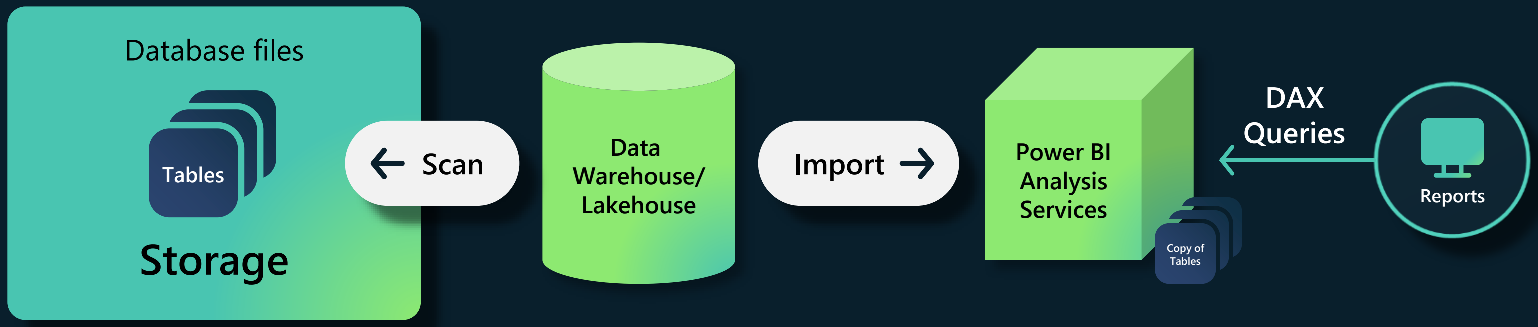
“Direct Query Mode”

Slow, but real time



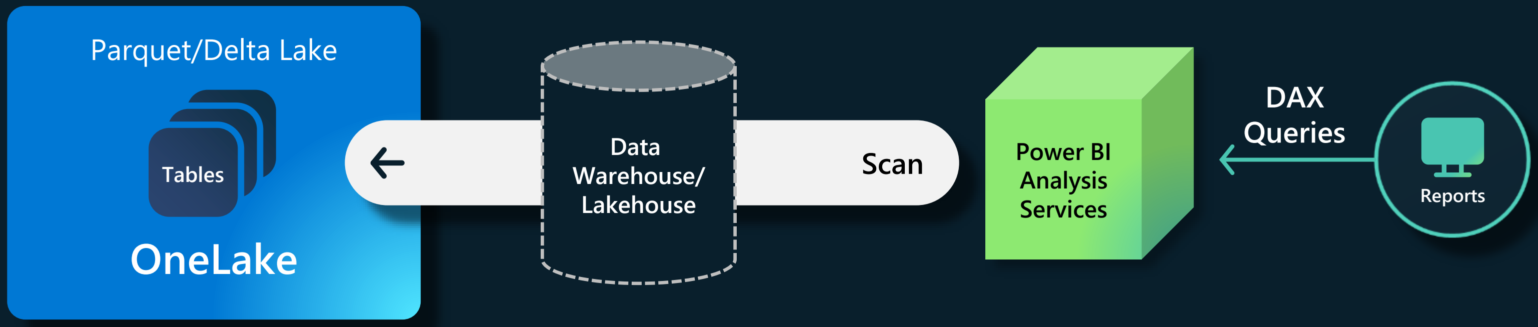
“Import Mode”

Latent & duplicative but fast



“Direct Lake Mode”

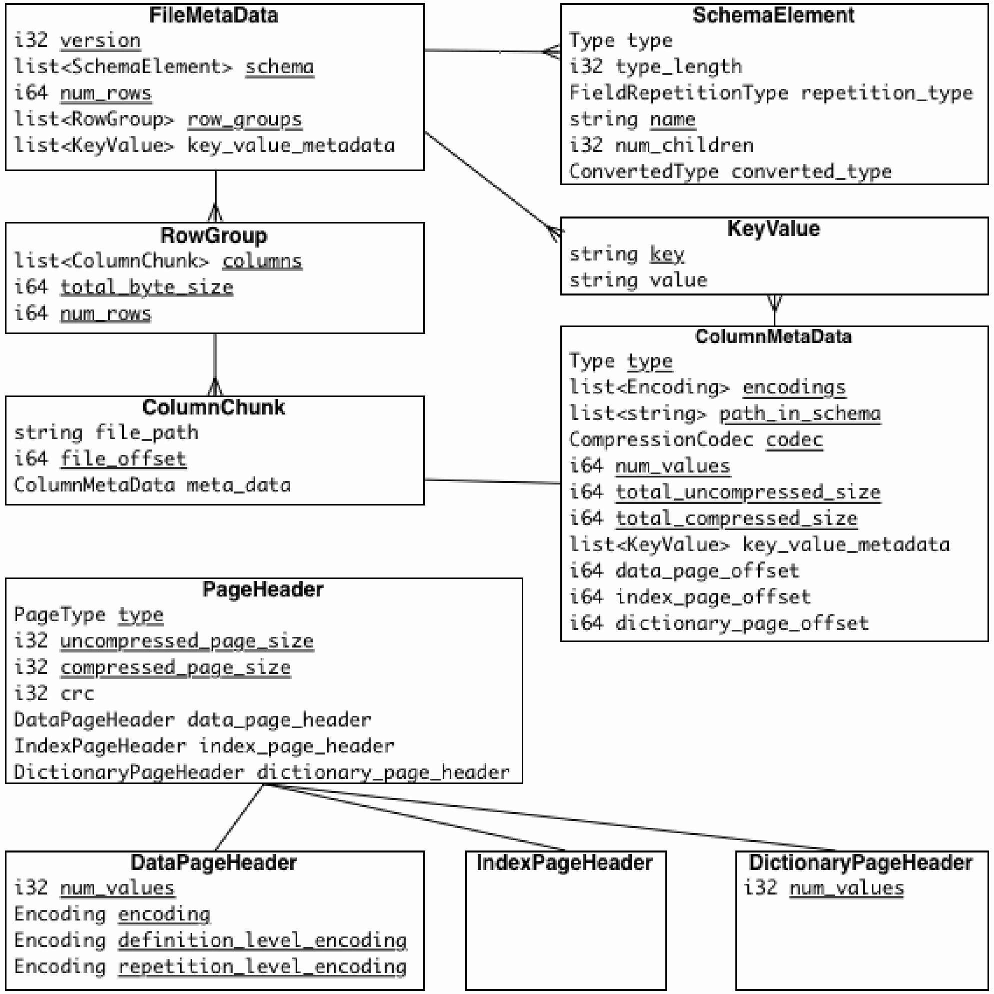
Perfect!



Why Delta?

Why Delta (Parquet)?

- Open Standard for data format
- Column oriented, efficient data storage and retrieval
- Efficient Data Compression and Encoding
- Becoming the Industry Standard
- Well suited for pruning (Column, rowgroup)
- Thrives on bulk operations



FileMetaData

```
i32 version
list<SchemaElement> schema
i64 num_rows
list<RowGroup> row_groups
list<KeyValue> key_value_metadata
```

SchemaElement

```
Type type
i32 type_length
FieldRepetitionType repetition_type
string name
i32 num_children
ConvertedType converted_type
```

RowGroup

```
list<ColumnChunk> columns
i64 total_byte_size
i64 num_rows
```

KeyValue

```
string key
string value
```

ColumnChunk

```
string file_path
i64 file_offset
ColumnMetaData meta_data
```

ColumnMetaData

```
Type type
list<Encoding> encodings
list<string> path_in_schema
CompressionCodec codec
i64 num_values
i64 total_uncompressed_size
i64 total_compressed_size
list<KeyValue> key_value_metadata
i64 data_page_offset
i64 index_page_offset
i64 dictionary_page_offset
```

PageHeader

```
PageType type
i32 uncompressed_page_size
i32 compressed_page_size
i32 crc
DataPageHeader data_page_header
IndexPageHeader index_page_header
DictionaryPageHeader dictionary_page_header
```

DataPageHeader

```
i32 num_values
Encoding encoding
Encoding definition_level_encoding
Encoding repetition_level_encoding
```

IndexPageHeader

DictionaryPageHeader

```
i32 num_values
```

Enums

Type: BOOLEAN, INT32, INT64, INT96, FLOAT, DOUBLE, BYTE_ARRAY, FIXED_LENGTH_BYTE_ARRAY

ConvertedType: UTF8, MAP, MAP_KEY_VALUE, LIST

FieldRepetitionType: REQUIRED, OPTIONAL, REPEATED

Encoding: PLAIN, GROUP_VAR_INT, PLAIN_DICTIONARY, RLE, BIT_PACKED

CompressionCodec: UNCOMPRESSED, SNAPPY, GZIP, LZO

PageType: DATA_PAGE, INDEX_PAGE

Inside Delta (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:

Inside Delta (Parquet) – Dictionary IDs

Header:

RowGroup1:

StoreID:	1, 1, 1
DateTime :	1, 2, 3
ProductID :	1, 1, 1
Value:	1, 2, 3

RowGroup2:

....

Footer:

Introducing V-Ordering

Write time optimization to parquet files

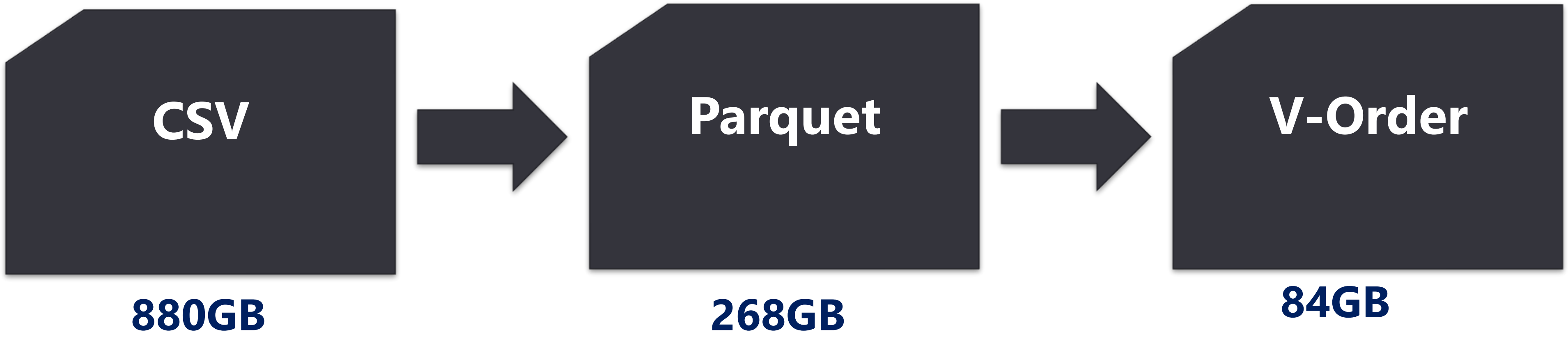
Sorting, row group distribution, dictionary encoding, and compression (Shuffling)

Complies to the open standard

Z-Order, compaction, vacuum, time travel, etc. are orthogonal to V-Order

V-ordering in action

Microsoft Internal DB (162 tables)



CSV → Parquet → V-Order

880GB

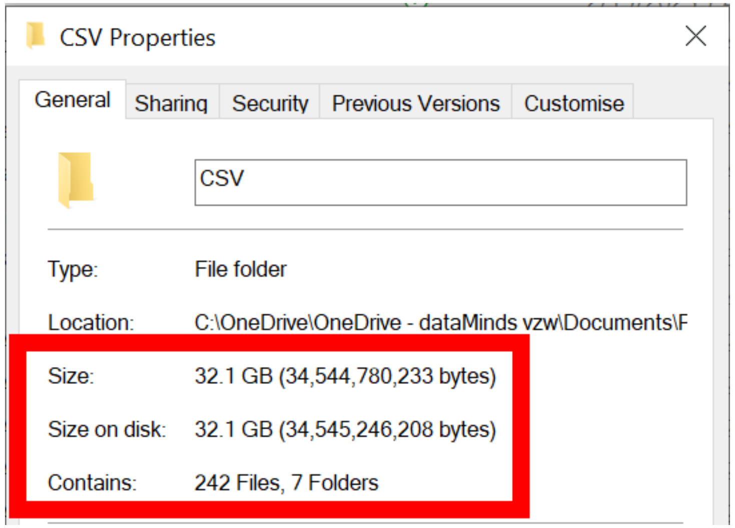
268GB

84GB

x3.2

Reduced IO for workloads

V-ordering in our demo case



CSV

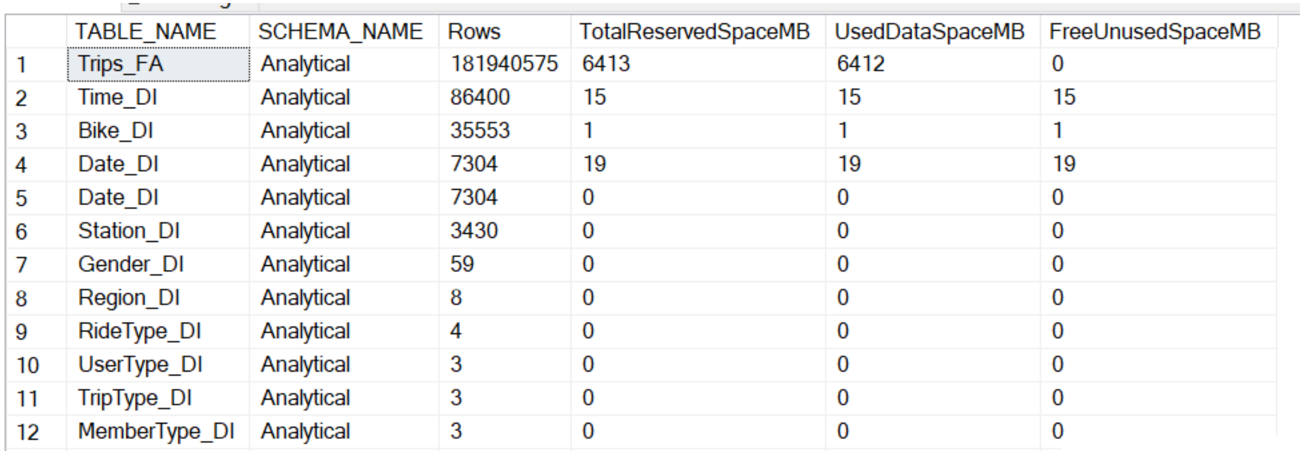
Type: File folder

Location: C:\OneDrive\OneDrive - dataMinds vzw\Documents\F

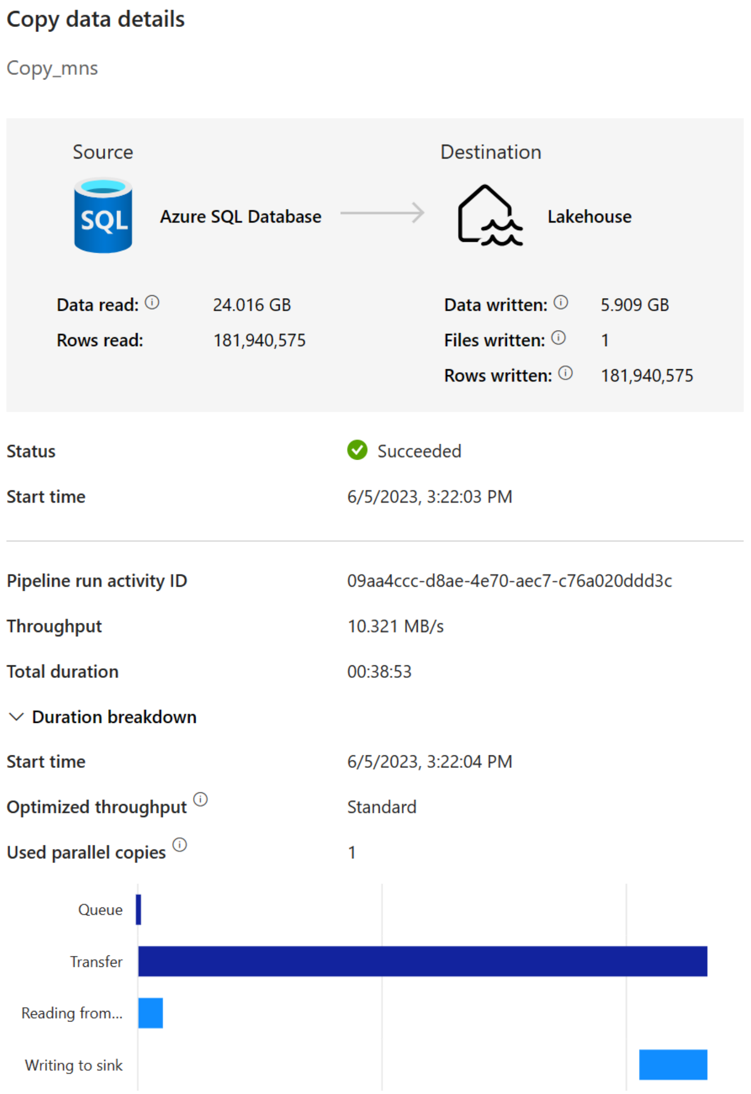
Size: 32.1 GB (34,544,780,233 bytes)

Size on disk: 32.1 GB (34,545,246,208 bytes)

Contains: 242 Files, 7 Folders



	TABLE_NAME	SCHEMA_NAME	Rows	TotalReservedSpaceMB	UsedDataSpaceMB	FreeUnusedSpaceMB
1	Trips_FA	Analytical	181940575	6413	6412	0
2	Time_DI	Analytical	86400	15	15	15
3	Bike_DI	Analytical	35553	1	1	1
4	Date_DI	Analytical	7304	19	19	19
5	Date_DI	Analytical	7304	0	0	0
6	Station_DI	Analytical	3430	0	0	0
7	Gender_DI	Analytical	59	0	0	0
8	Region_DI	Analytical	8	0	0	0
9	RideType_DI	Analytical	4	0	0	0
10	UserType_DI	Analytical	3	0	0	0
11	TripType_DI	Analytical	3	0	0	0
12	MemberType_DI	Analytical	3	0	0	0
13	FileType_DI	Analytical	3	0	0	0



Copy data details

Copy_mns

Data read: 24.016 GB

Rows read: 181,940,575

Data written: 5.909 GB

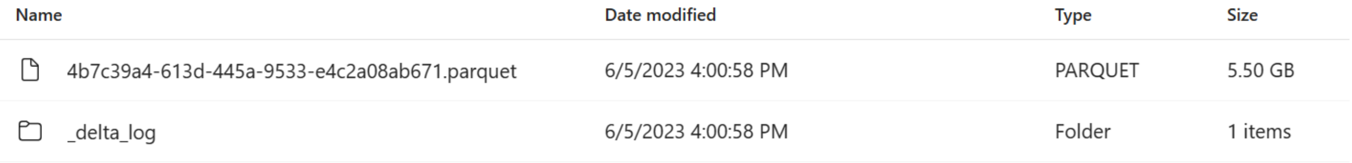
Files written: 1

Rows written: 181,940,575

Status	✓ Succeeded
Start time	6/5/2023, 3:22:03 PM
Pipeline run activity ID	09aa4ccc-d8ae-4e70-aec7-c76a020ddd3c
Throughput	10.321 MB/s
Total duration	00:38:53

Duration breakdown

Start time	6/5/2023, 3:22:04 PM
Optimized throughput	Standard
Used parallel copies	1



Name	Date modified	Type	Size
4b7c39a4-613d-445a-9533-e4c2a08ab671.parquet	6/5/2023 4:00:58 PM	PARQUET	5.50 GB
_delta_log	6/5/2023 4:00:58 PM	Folder	1 items

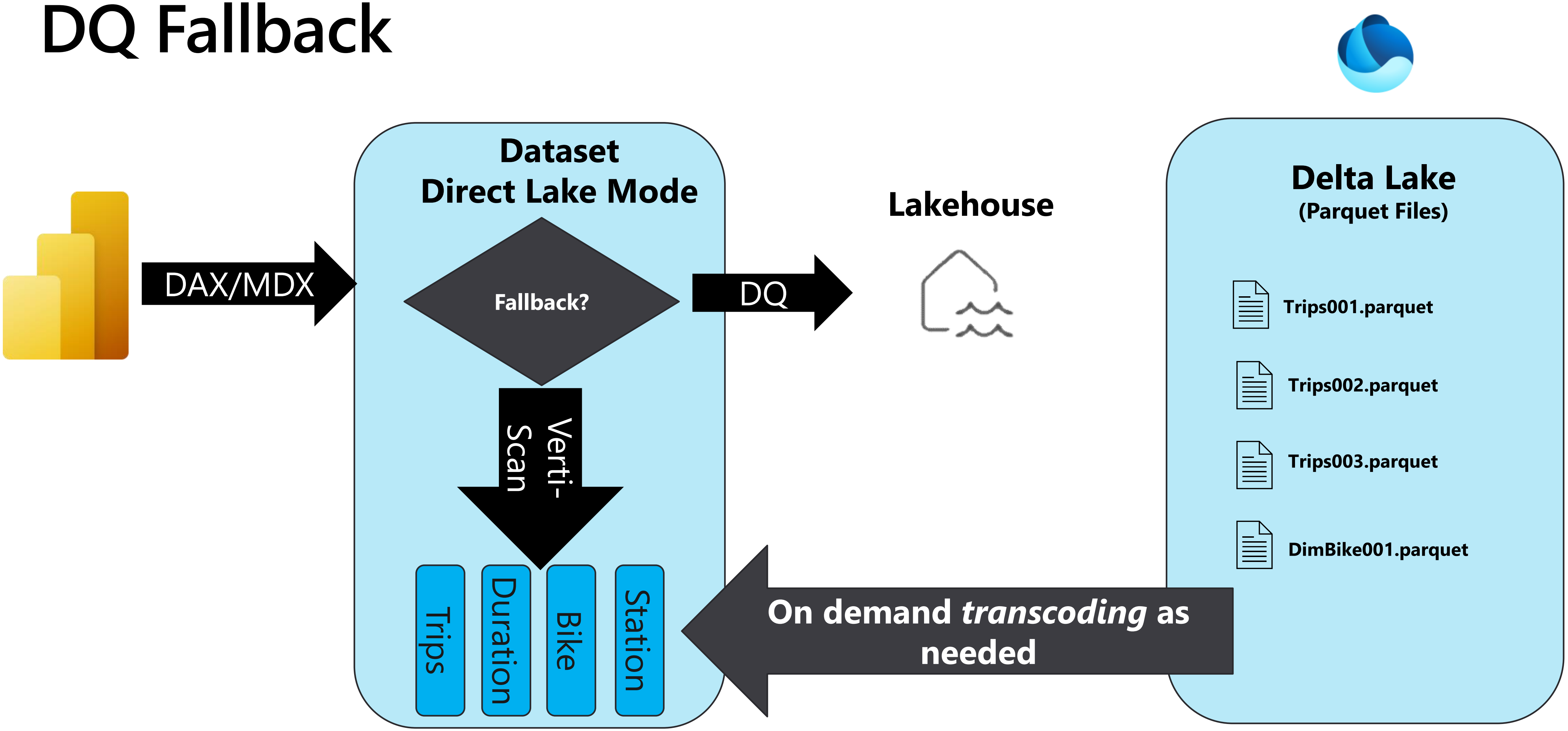
STOP! Demo time!

Using Direct Lake mode over a Lakehouse

DirectLake Mode

- On start, no data is loaded in-memory
- Column data is ***transcoded*** from Parquet files when queried
- Multi-column tables can have mix of ***transcoded*** (resident) and non-resident
- Column data can get evicted over time
- DirectLake fallback to SQL Server for suitable sub-queries
- “Framing” of dataset determines what gets loaded from DeltaLake

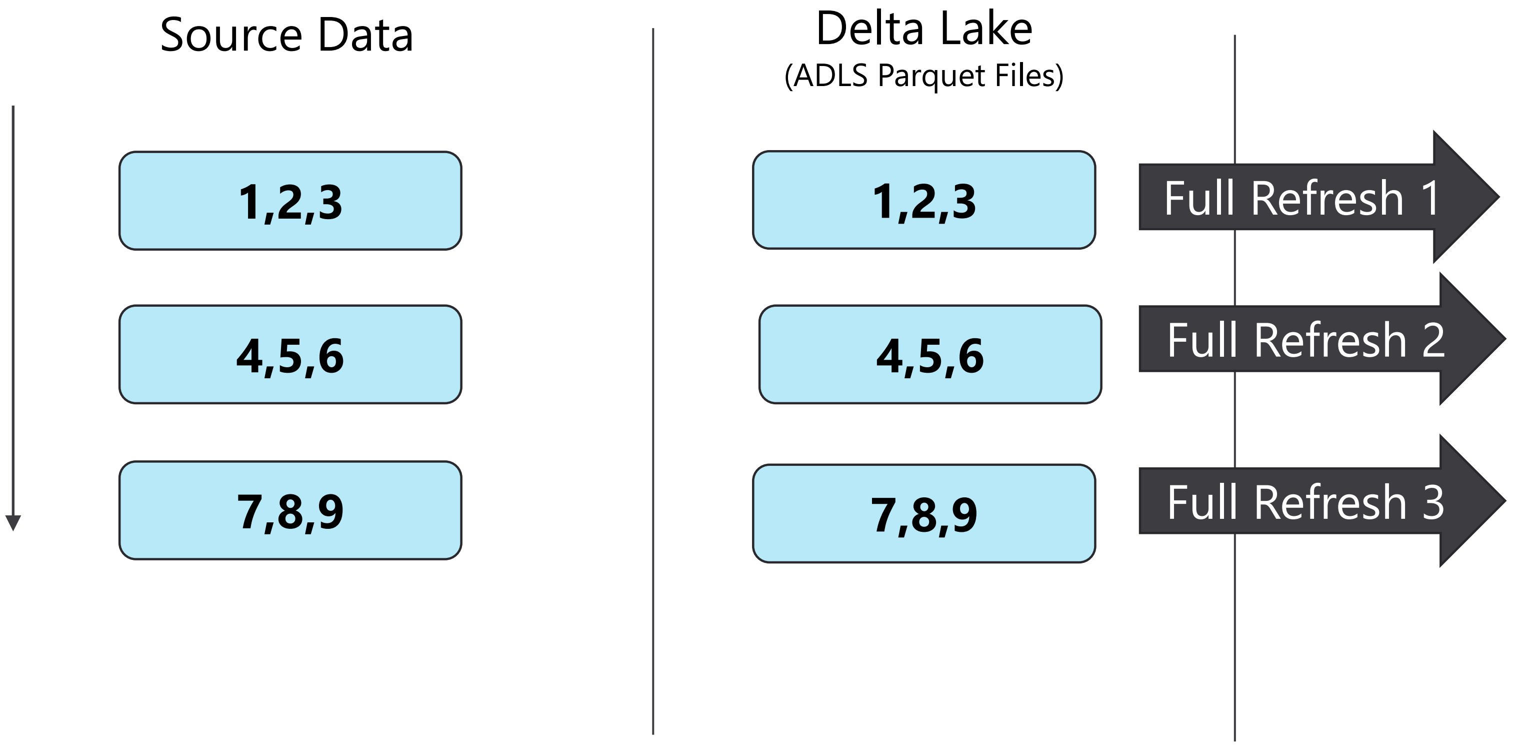
DQ Fallback


DAX/MDX
Dataset
Direct Lake Mode
Fallback?
DQ
Lakehouse
Verti-Scan
Trips
Duration
Bike
Station
On demand *transcoding* as needed
Delta Lake
(Parquet Files)
Trips001.parquet
Trips002.parquet
Trips003.parquet
DimBike001.parquet

Framing

- What is framing
  - "point in time" way of tracking what data can be queried by DirectLake
- Why is this important
  - 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



Source Data

1,2,3

4,5,6

7,8,9

Delta Lake
(ADLS Parquet Files)

1,2,3

4,5,6

7,8,9

Full Refresh 1

Full Refresh 2

Full Refresh 3

Power BI
Dataset

```
EVALUATE ‘Table’

Value
-------
1
2
3
4
5
6
7
8
9
```

STOP! Demo time!

Let’s look at Framing

Optimizing Delta for Direct Lake mode

Optimizing Delta for Direct Lake mode

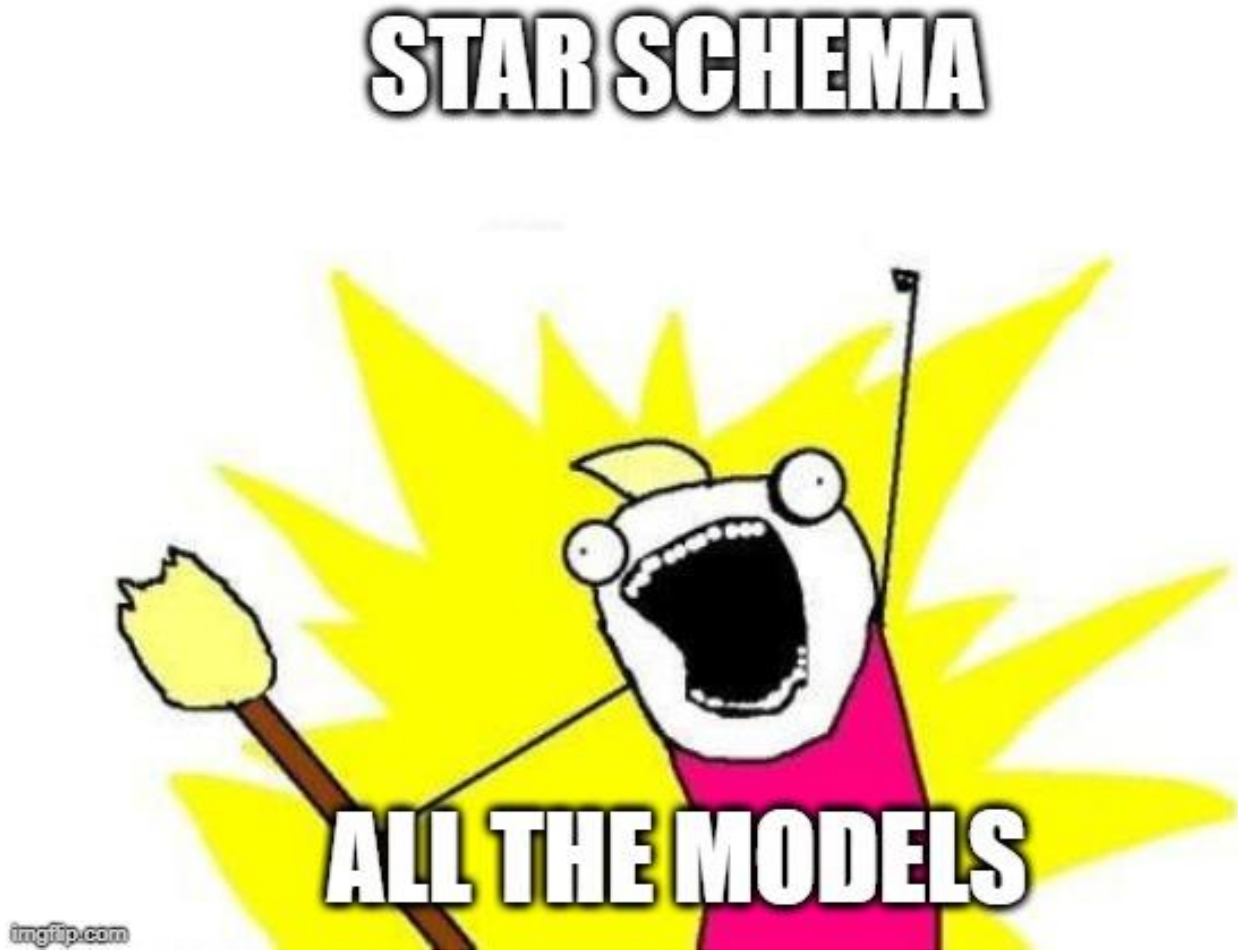
- V-Order makes a big difference, as it’s tailored for Verti-Scan
- Direct Lake will work over Shortcuts to external data
  - Expect a performance impact, because reasons ..
- Direct Lake thrives on fewer, larger .parquet files
  - Physical structure will always be crucial
  - OPTIMIZE (bin-compaction) and VACUUM in the Data Engineering process will be key
  - Especially with streaming/small batch architectures, keep this in mind
- Principle of lean models will still apply
  - Only include what’s needed for the reports and datasets

Common Answers to Common Questions

“Greatest Hits”

- Delta doesn’t like spaces in object names ☺
- Delta Tables are a hard requirement for Direct Lake mode
  - Dataflows Gen2, Pipelines, Notebooks can create them for you in the lakehouse
- Web modelling is the only way to use DirectLake for now
- XMLA Read/Write is not yet supported
  - No External Tools, Calc Groups, ..
- DirectLake doesn’t have unique DAX limitations
  - DQ does ..
- No confirmed plans right now to support Apache Iceberg, HUDI, ..
- No, you can’t have Copilot yet

What does this mean for my data modelling?



Thanks, @KoVer!

Data should be transformed as far upstream as possible, and as far downstream as necessary.

Matthew Roche, 2021
(The purple haired sword afficionado)
https://ssbipolar.com/2021/05/31/roches-maxim

Resources

- https://learn.microsoft.com/en-us/power-bi/enterprise/directlake-overview
- https://learn.microsoft.com/en-us/power-bi/enterprise/directlake-analyze-qp
- https://learn.microsoft.com/en-us/fabric/data-engineering/lakehouse-pbi-reporting
- https://learn.microsoft.com/en-us/fabric/data-engineering/delta-optimization-and-v-order?tabs=sparksql
- https://fabric.guru/power-bi-direct-lake-mode-frequently-asked-questions
- https://www.fourmoo.com/2023/05/24/using-power-bi-directlake-in-microsoft-fabric/

Slides

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

Thank you