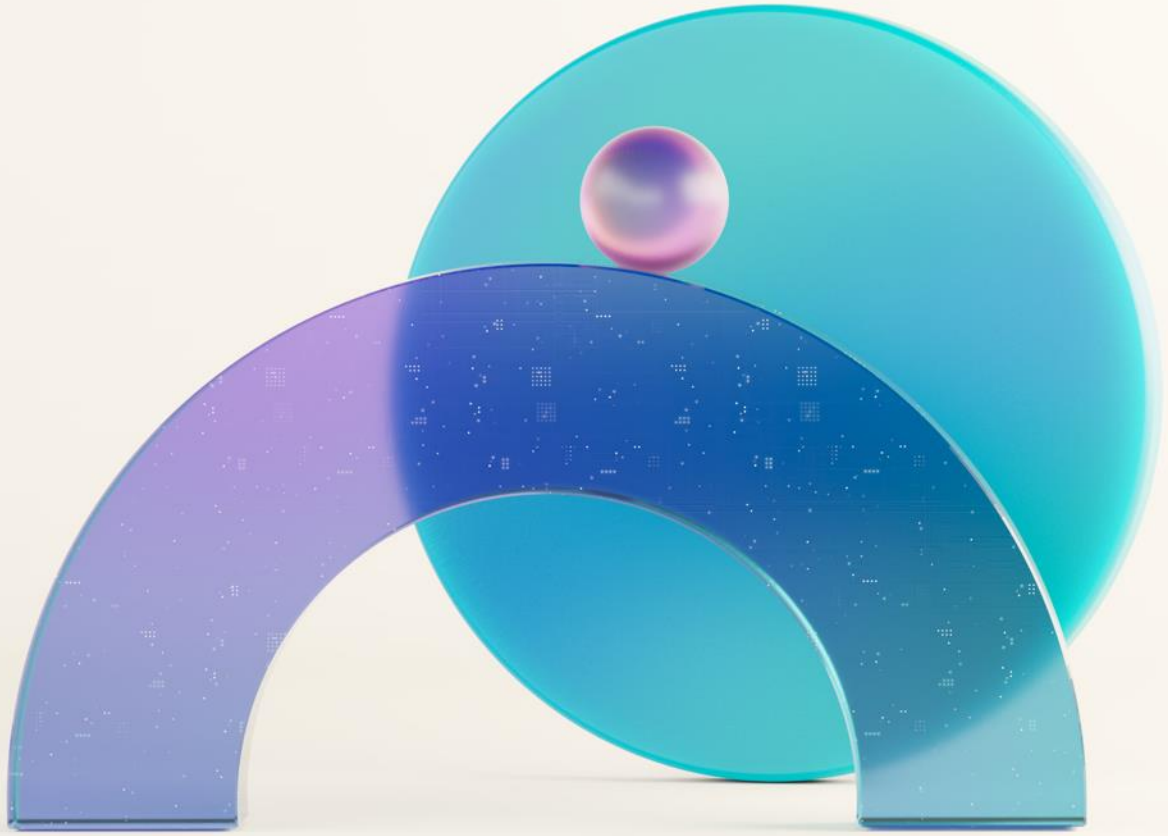
Microsoft



Data Warehouse performance best practices

Premium sponsors

Standard sponsors

Filip Popović

Senior Product Manager, Fabric
Espresso co-host
Microsoft

Artur Vieira

Principal Product Manager, Fabric CAT
Microsoft

Agenda

01

Key ingredients for optimal performance

02

Design for performance

03

Data ingestion best practices

04

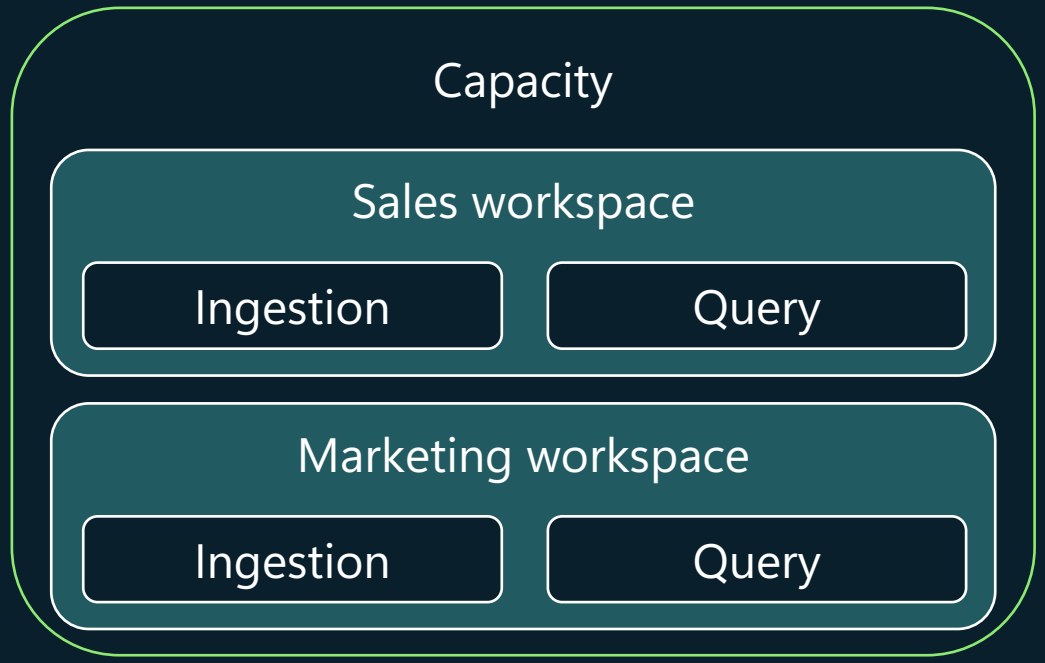
Consumption considerations

Key ingredients for optimal performance

Predictable workload performance

With automatic workload management and workspace as natural isolation boundary

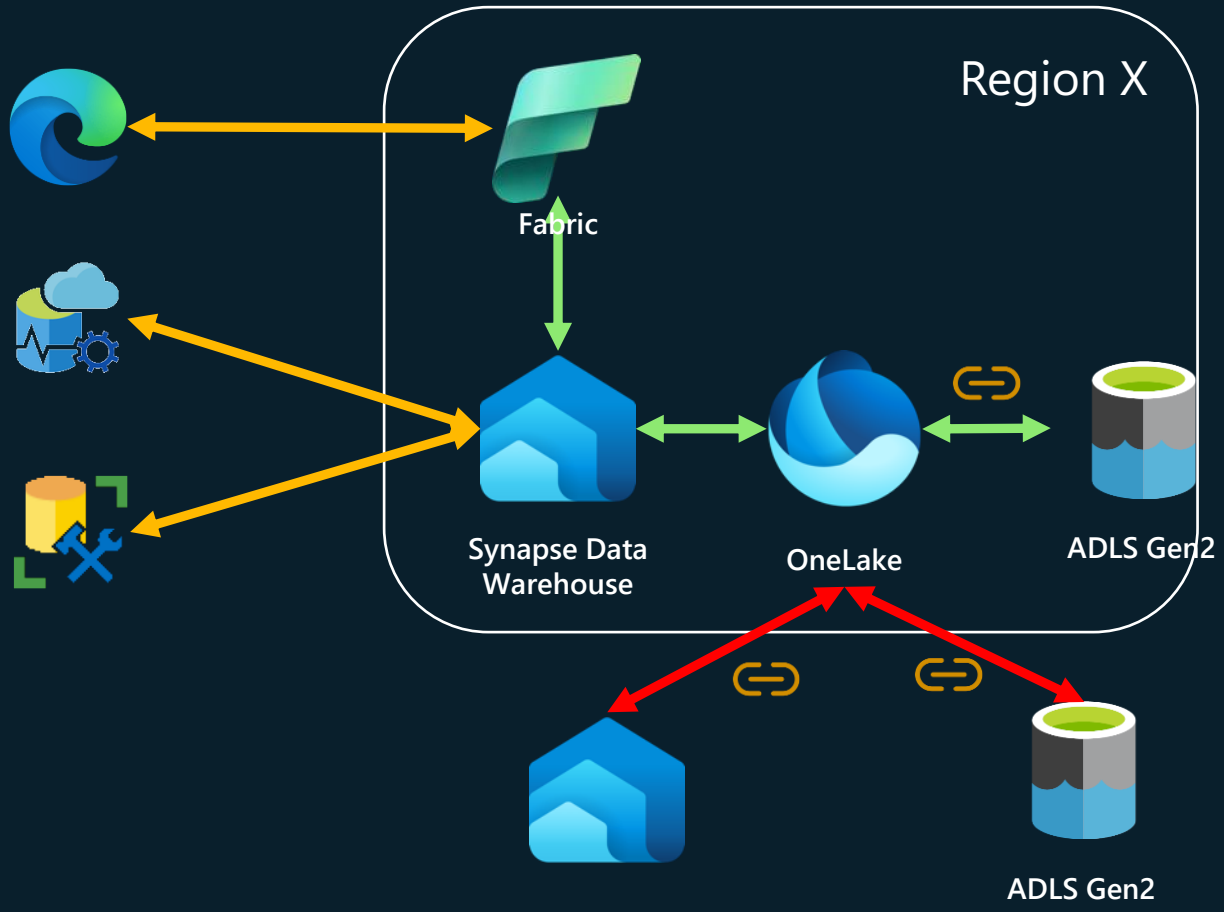
- Predictable ETL performance based on compute isolation
- Scales independently



Design for performance

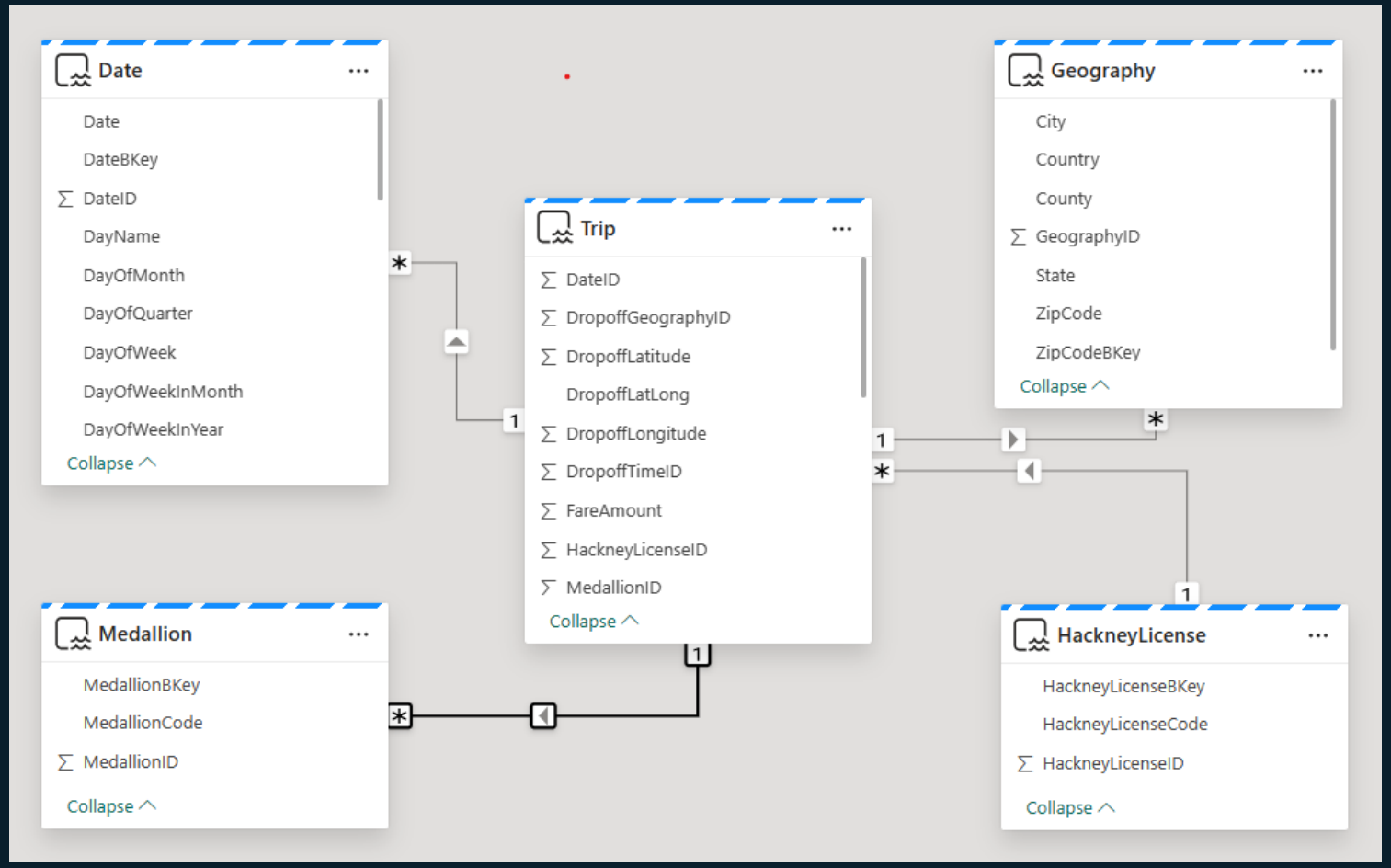
Where are my compute and data?

- Collocate resources
- Compute is where capacity is
- Mind network latency between:
  - The client and the endpoint
  - The engine and the data in case of shortcuts



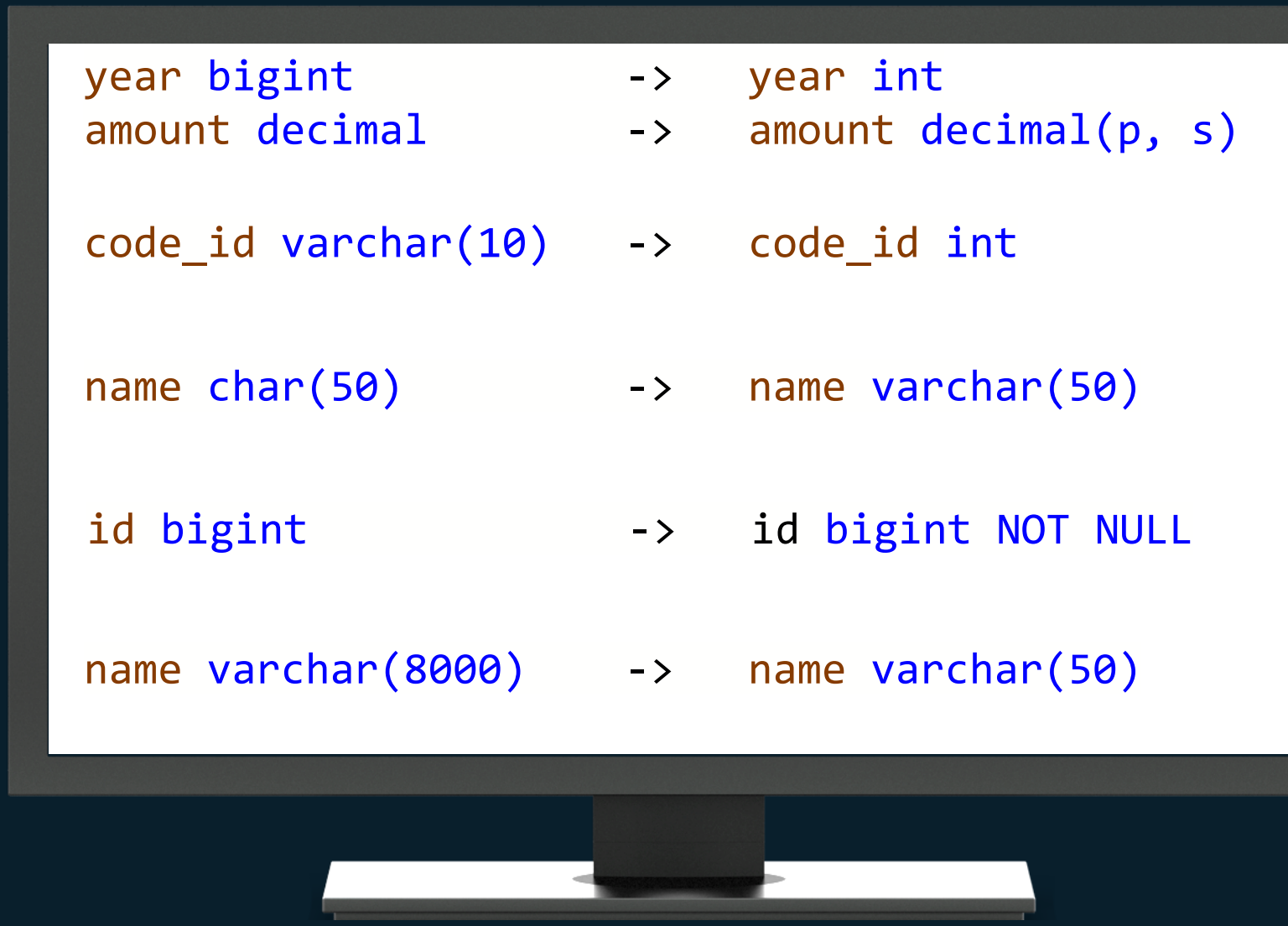
Design for performance

- Utilize a Star Schema data design
- Fact tables
- Dimension Tables
- Integration Tables



Pick optimal data types

- Use smallest data type that accommodates values
- Use proper types
- Use varchar instead of char
- Use NOT NULL instead of NULL
- Check table schema created by tools (ETL tools)

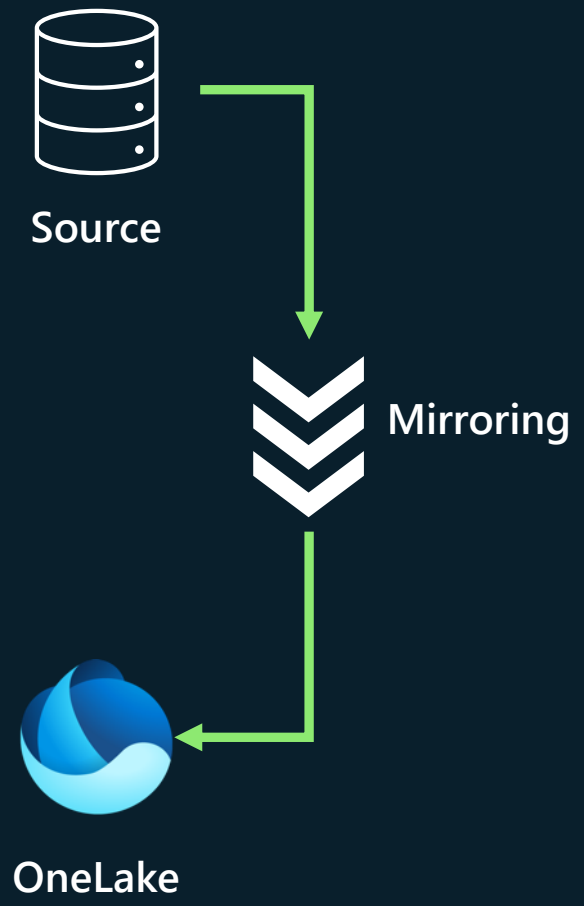


Data ingestion best practices

Don’t ingest, mirror instead

- Seamlessly connect your data to Microsoft Fabric
- Near real time replication with zero ETL
- Optimal format for analytics by default

- Currently supported sources:
  - Snowflake
  - Azure Cosmos DB
  - Azure SQL DB



COPY INTO

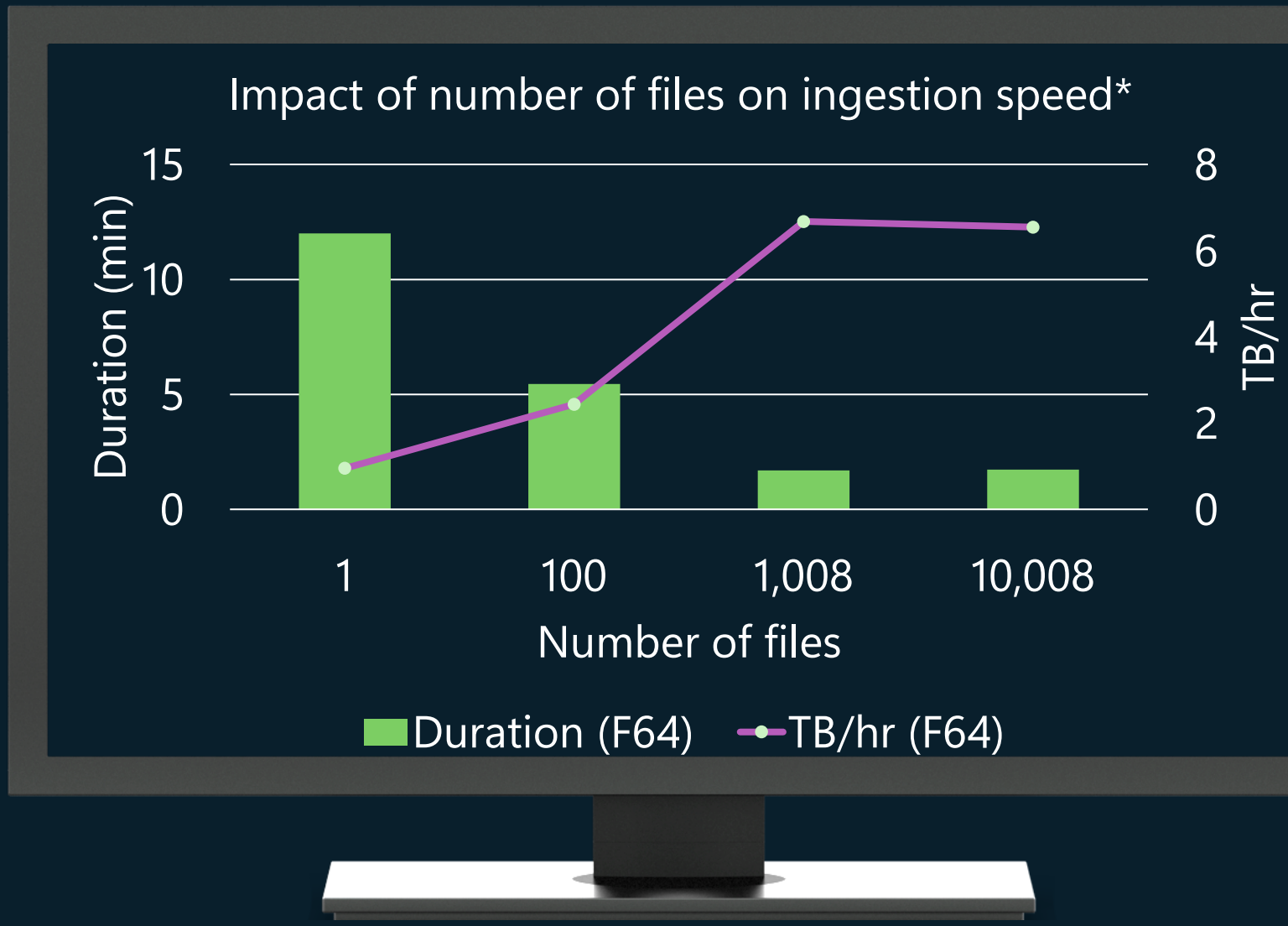
- Enables flexible, high-throughput data ingestion from external Azure storage accounts


```
COPY INTO table_name.[(Column_list)]
FROM '<external_location>' [,...n]
WITH
 (
 [FILE_TYPE = {'CSV' | 'PARQUET' '} ]
 [,CREDENTIAL = (AZURE CREDENTIAL) ]
 [,ERRORFILE = '[http(s)://account/container]/directory[/]]']
 [,ERRORFILE_CREDENTIAL = (AZURE CREDENTIAL) ]
 [,MAXERRORS = max_errors ]
 [,COMPRESSION = { 'Gzip' | 'Snappy'}]
 [,FIELDQUOTE = 'string_delimiter']
 [,FIELDTERMINATOR = 'field_terminator']
 [,ROWTERMINATOR = 'row_terminator']
 [,FIRSTROW = first_row]
 [,ENCODING = {'UTF8'|'UTF16'}]
 [,PARSER_VERSION = {'1.0'|'2.0'}]
 )
```

Ingestion best practices - size and number of files

COPY INTO:

- More small files > few large files
- At least 4MB file size
- Keep the number of files to be at least 1,000
- Best throughput using 1,000 files is 7x faster than single large file

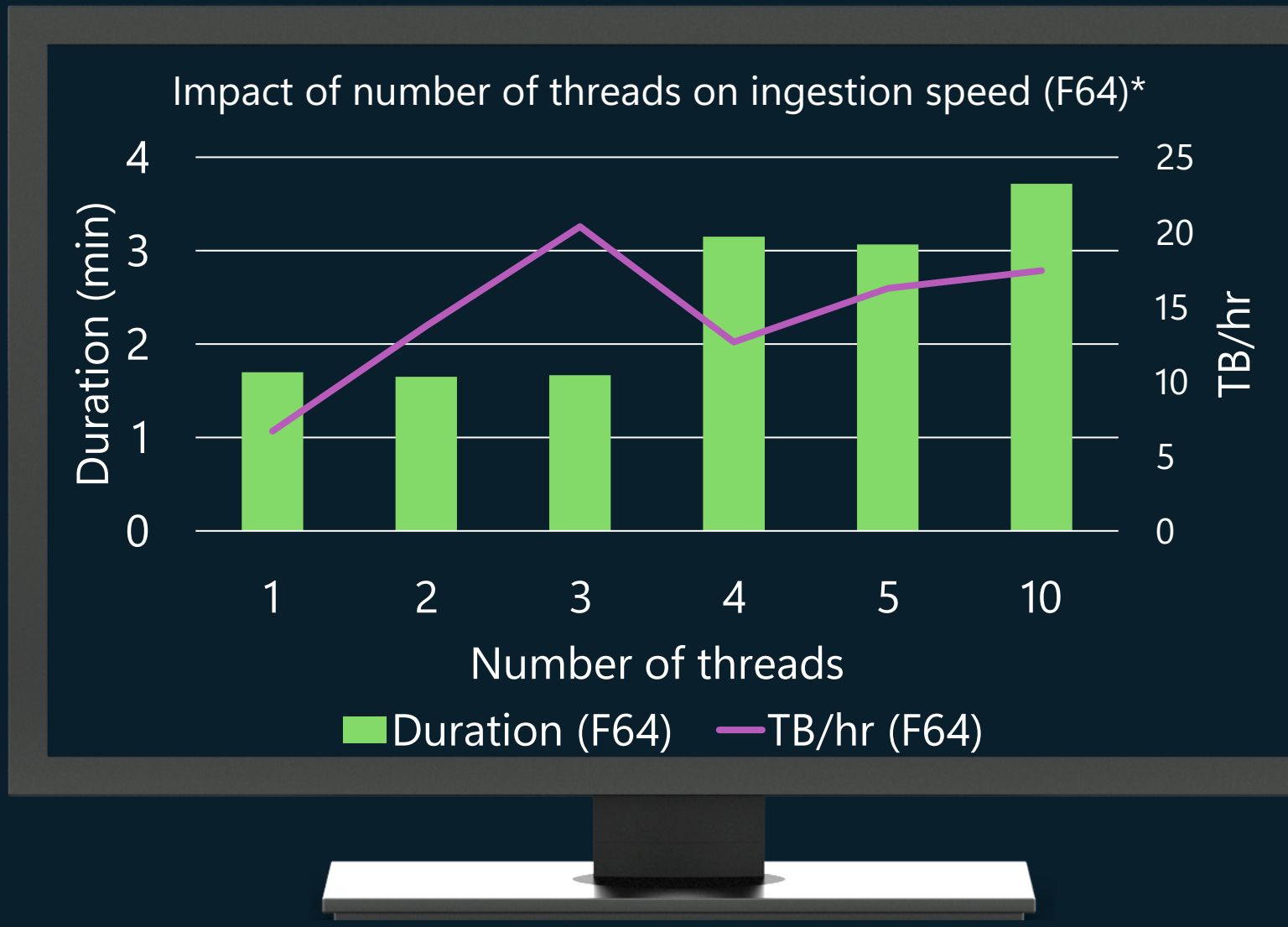


*TPCH 1TB, Orders table

Ingestion best practices - parallelize

COPY INTO:

- Parallel loads achieve higher throughput

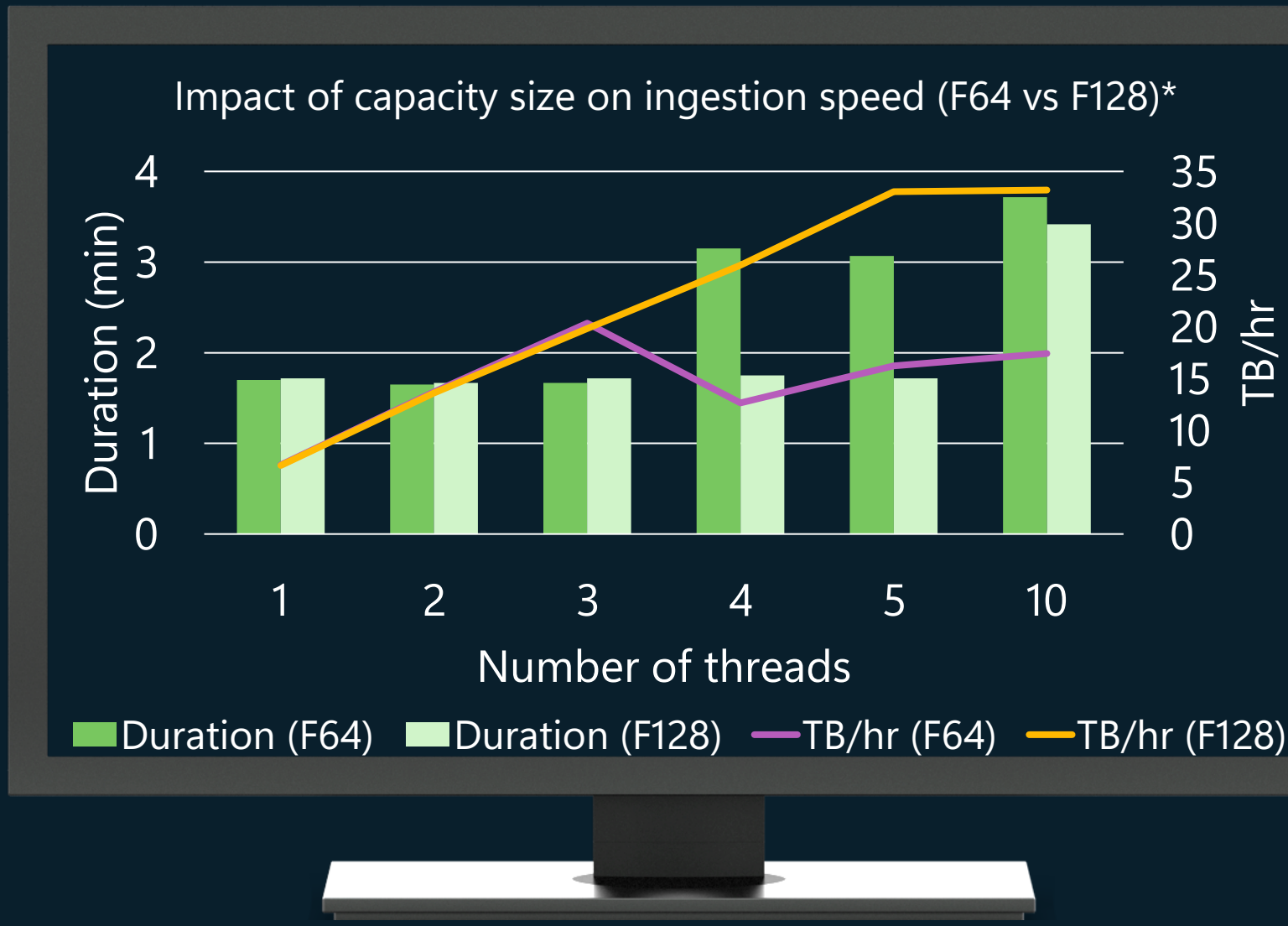


*TPCH 1TB, Orders table, 170GB each thread

Increase throughput with larger Fabric capacities

All ingestion (pipelines, dataflows, SQL):

- Scales with larger Fabric Capacities



*TPCH 1TB, Orders table, 170GB each thread

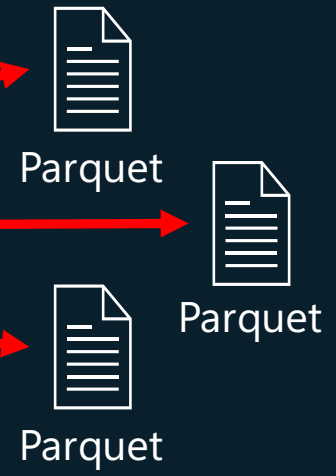
Few more ingestion tips

- **Shortcuts** into the lakehouse
- **CTAS** or **INSERT** for ingestion from the lakehouse, instead of pipelines
- **CLONE** TABLE for instantly available replica

INSERT/UPDATE/DELETE data in batches

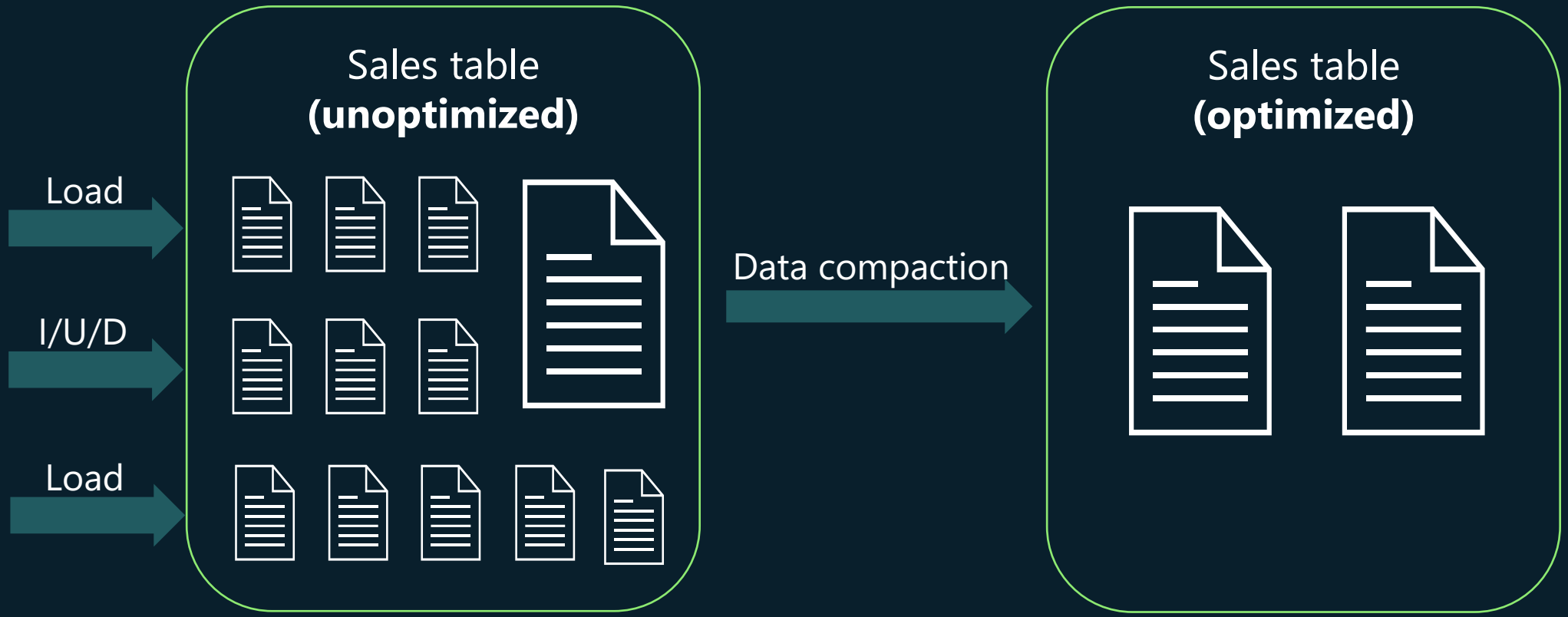
- Avoid singleton INSERT INTO statements – ineffective for both writes and reads

```
INSERT INTO person VALUES (1, 'Mike')
INSERT INTO person VALUES (2, 'Peter')
INSERT INTO person VALUES (3, 'John')
```




```
INSERT INTO person VALUES (1, 'Mike'),
    (2, 'Peter'),
    (3, 'John')
```

Zero maintenance with automatic data compaction

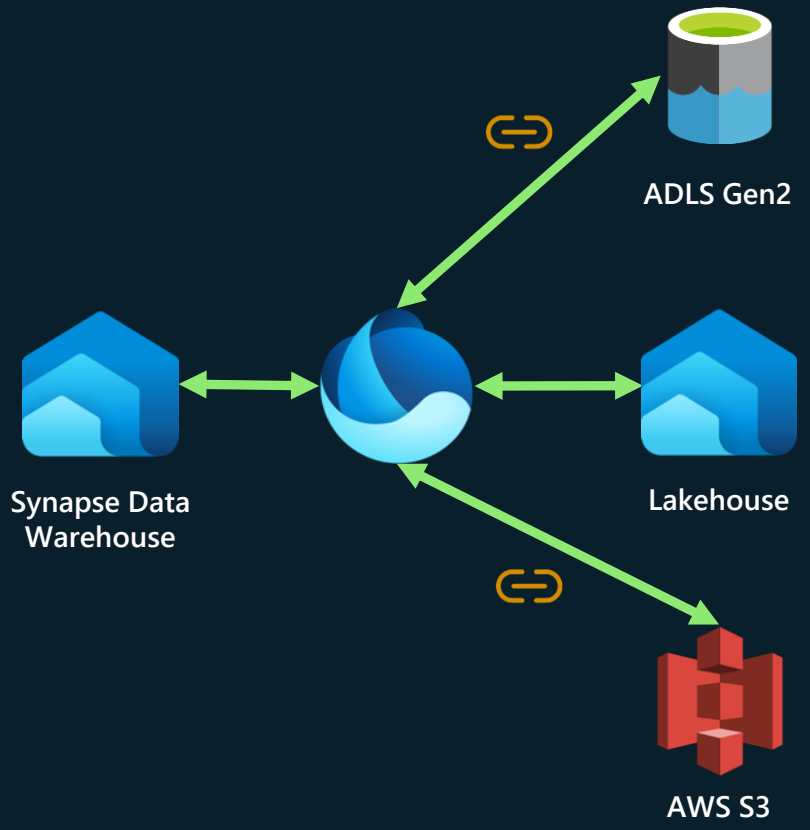


Microsoft Fabric
COMMUNITY CONFERENCE

Consumption considerations

Optimize Lakehouse tables

Fabric Warehouse can optimize only Warehouse tables



Yet another table to SQL engine, same best practices apply

Check data types, particularly string lengths, decimal precision and scale and use NOT NULL if possible

OPTIMIZE table to get optimal number and sizes of files

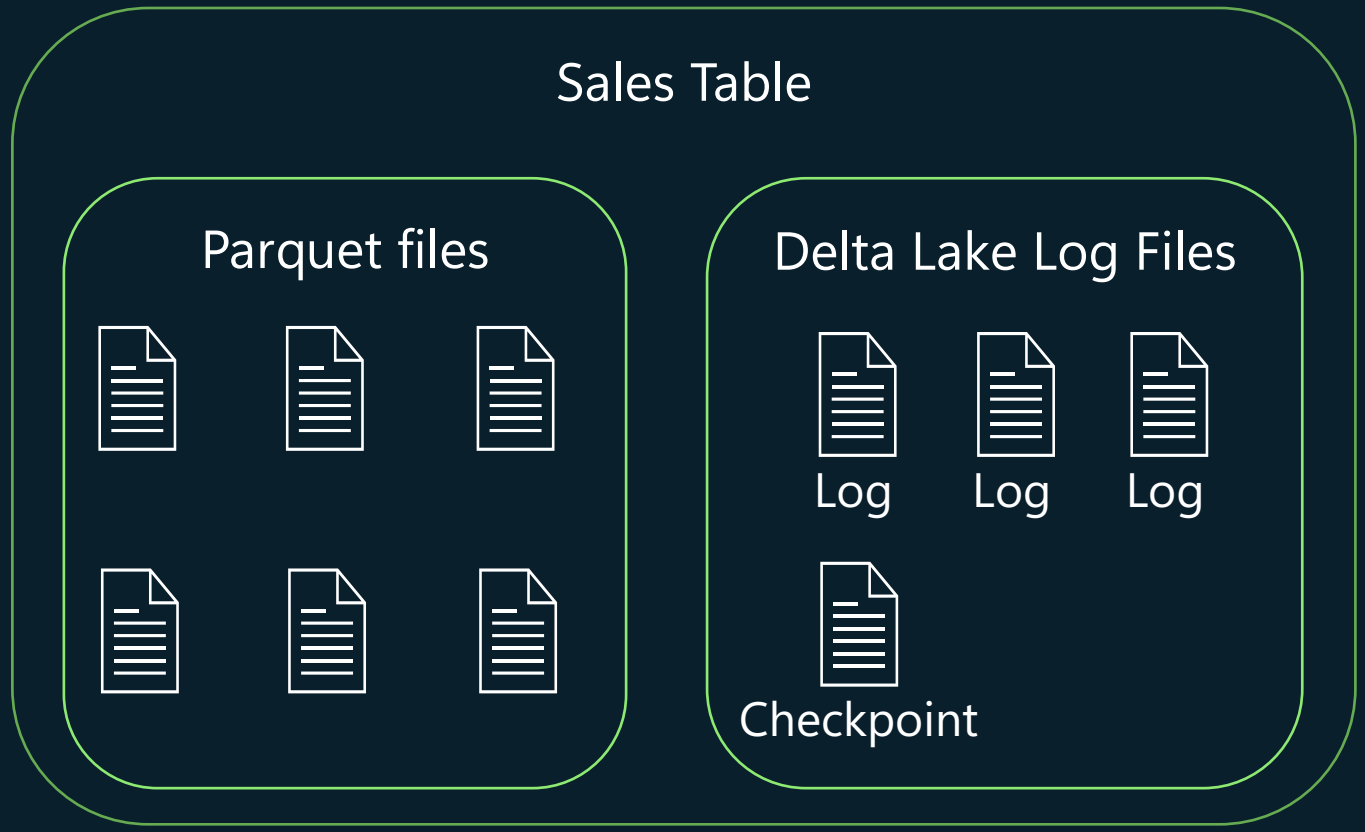
Partition table by columns commonly used for filtering

Z-ORDER table by non-partitioning columns commonly used for filtering.

Consume warehouse tables from other services

Fabric plays nicely with the ecosystem

- Tables published to OneLake as Delta tables
- Any engine can access the table
- Any engine can benefit from V-ordering
- Checkpointing enables faster reads



Raffle Prizes