SQLBITES.NET

Your Warehouse, Your Semantic Model

Stop tuning measures. Fix the layer underneath.

Jonathan Stewart · SQLLocks / SQLBites

Jonathan Stewart

SQLLOCKS / SQLBITES

I live in both layers - the report and the warehouse underneath it. That's the whole point of this talk.

→ **Fabric & Power BI**
Architecture, performance, governance.

→ **SQLBites.net**
Fabric / Power BI education.

Sound BI

Nine seconds.

THE REPORT, IN FRONT OF THE EXECS

9 sec

to load. Every visual. Every click.

YOU HAD ALREADY TRIED EVERYTHING

✗	**Optimized the DAX**	*nothing moved*
✗	**Reduced the visuals**	*nothing moved*
✗	**Checked the model**	*nothing moved*

Because the problem was never in the report layer.

The room has lived this.

One *clustering* key.

UNCLUSTERED

9s

Same report. Same DAX.

CLUSTERED

0.8s

No DAX changes. No schema redesign.

The problem was never your measures. It was the storage layer underneath them.

MOST POWER BI DEVELOPERS HAVE NEVER TOUCHED THE STORAGE LAYER.

It's been someone else's job.

In Import mode you could get away with that. In Direct Lake you can't. And that's good news: the highest-leverage lever is now in your hands.

What this is. And isn't.

IS

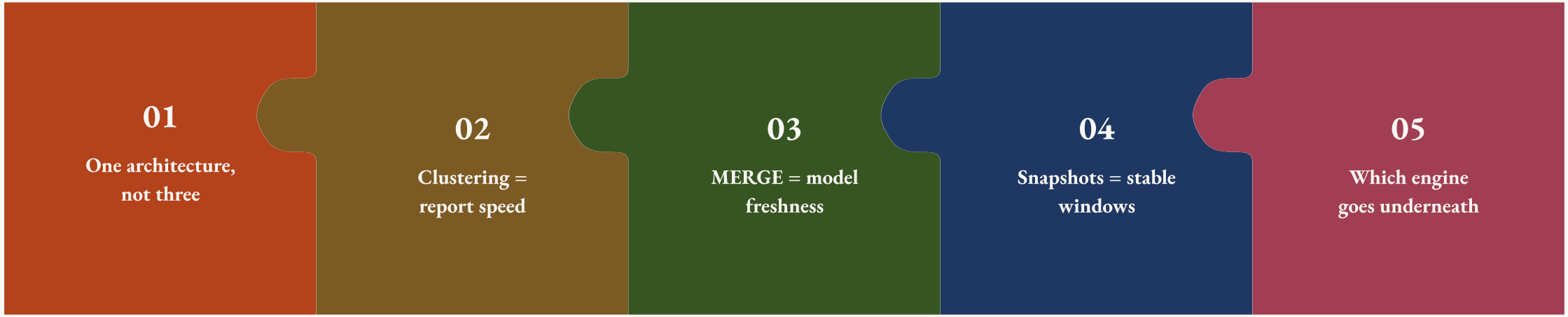
- The layer under your model
- Why your report loads in 1s, not 9s
- Assumes you have a warehouse + model

ISN'T

- A Fabric intro
- "Write faster DAX"
- Someone else's job

Other sessions teach DAX. This goes one level lower. To the layer that decides whether your DAX ever had a chance.

One report, four changes, *zero DAX.*



CHAPTER ONE

ONE ARCHITECTURE, NOT THREE

01

Warehouse to Direct Lake to report.

Three teams' problem. Or one pipeline you own.

THE MENTAL MODEL

- "The warehouse" (someone else)
- "My model"
- "My report"
- Three handoffs, three places to blame

THE REALITY

- Warehouse → Direct Lake → Power BI
- On OneLake, stored once (Delta/Parquet)
- The model reads it in place
- One architecture, end to end

OneLake stores it once. The semantic model reads those same files. No "move the data into the model" step.

What Direct Lake actually means for you

01	**No import**	*No scheduled refresh moving data into the model.*
02	**Reads Parquet directly**	*The model reads the warehouse's files in place.*
03	**Import speed, live data**	*No Import-vs-DirectQuery tradeoff. If the files underneath are organized well.*

That "if" is the rest of the talk.

The abstraction that protected you is gone.

IMPORT MODE	DIRECT LAKE
01 Storage hidden behind a refresh	01 How the warehouse stores =
02 You never saw the files	02 how fast your model reads
03 Tuning = DAX + model	03 Storage is now visible in load time

vs.

Not a threat. The lever. The way rows are physically stored shows up directly in your report's load time.

One copy of the data. Zero refresh.

Direct Lake reads the warehouse's own Parquet files in place. There is no import step to schedule, and nothing to copy.

FABRIC WAREHOUSE

FactSales 10,000,000 rows

T-SQL owns the load: MERGE, clustering, snapshots.

→

ONELAKE

Delta / Parquet stored once

The warehouse's files ARE the model's storage. V-Ordered for reads.

→

DIRECT LAKE MODEL

Transcoded on demand

Columns page into memory at query time. Import speed, live data.

→

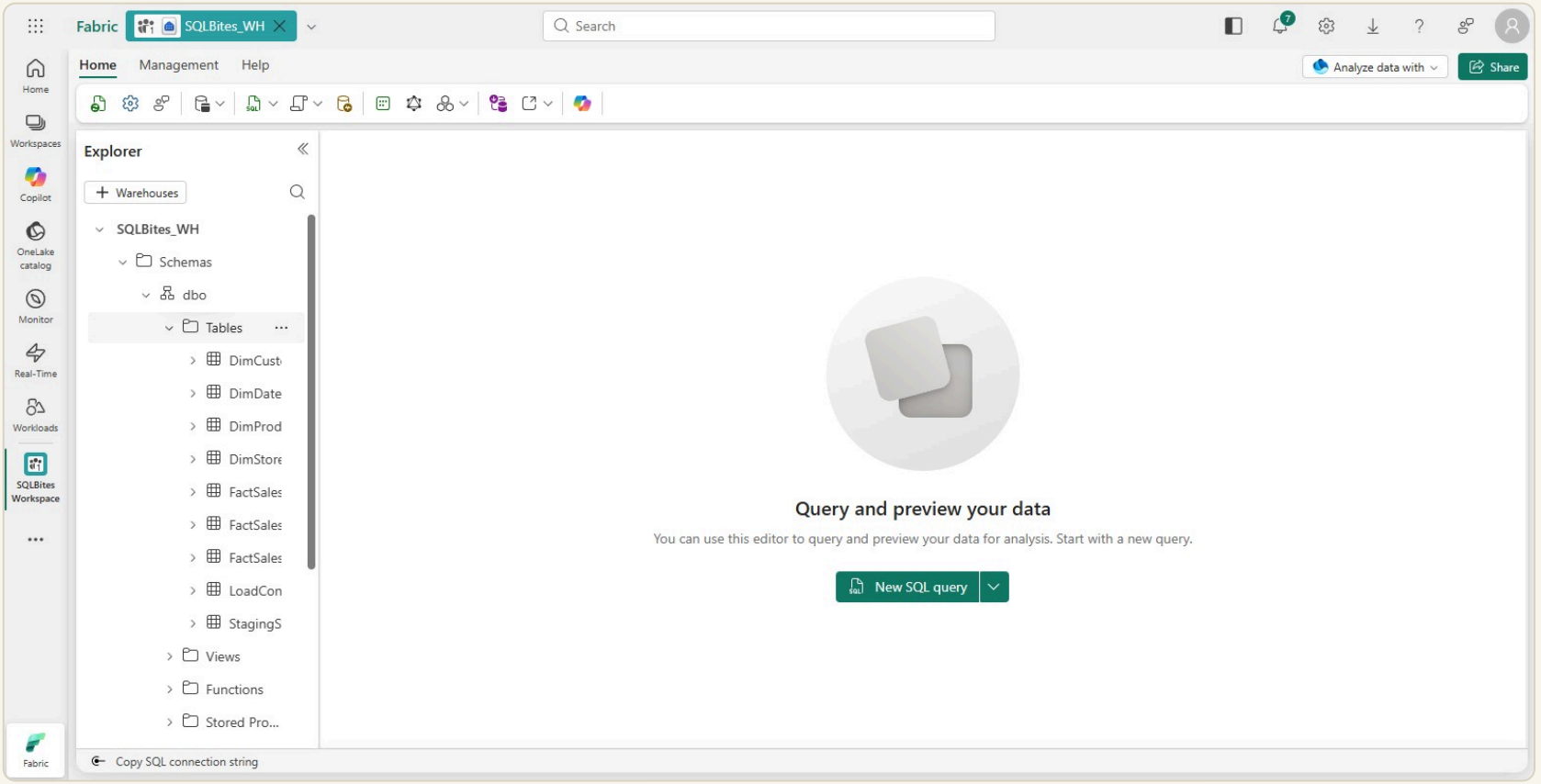
POWER BI REPORT

Your visuals, your DAX

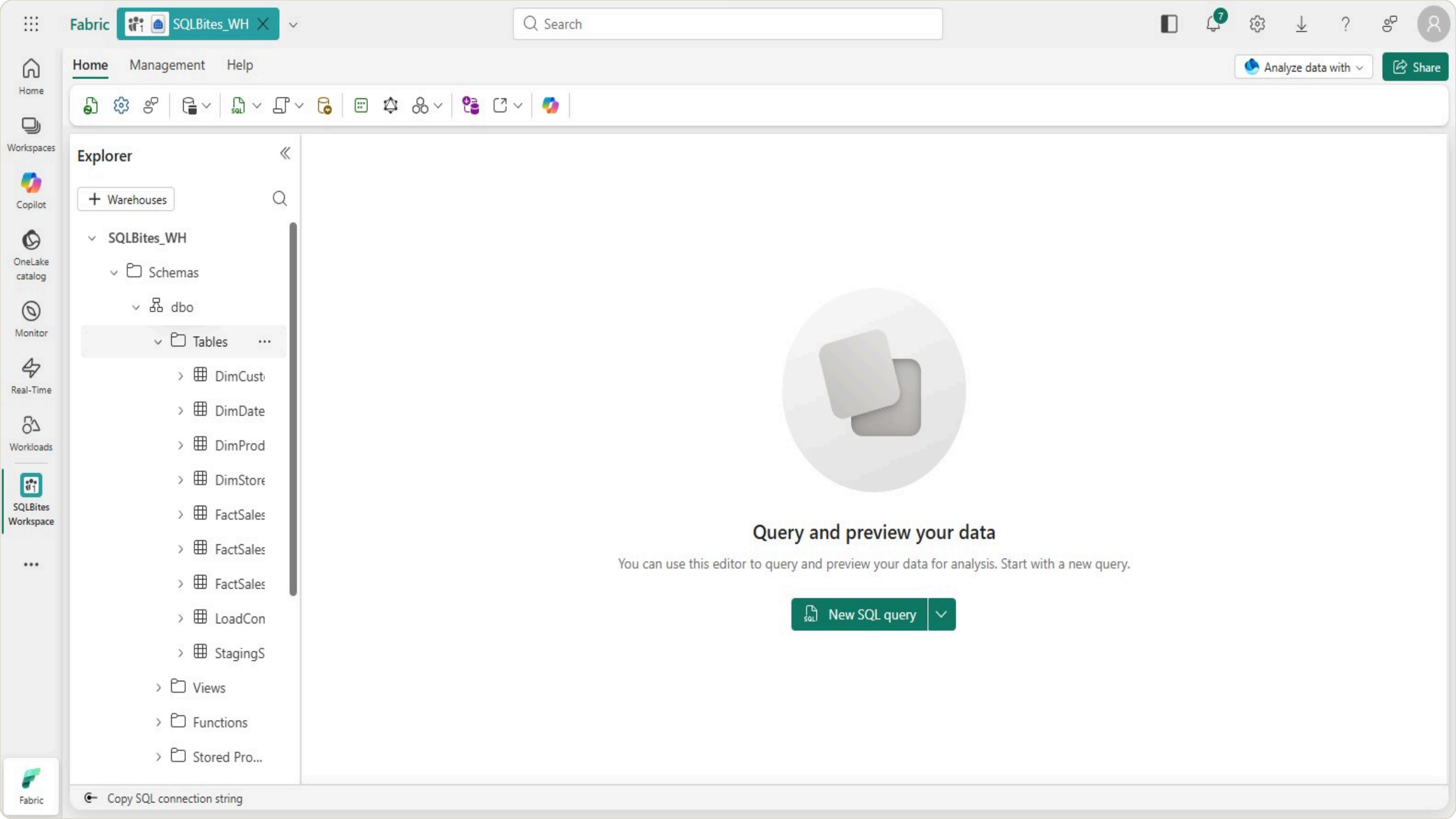
Untouched for the whole hour.

How the warehouse stores rows is now `directly visible` in your report's load time.

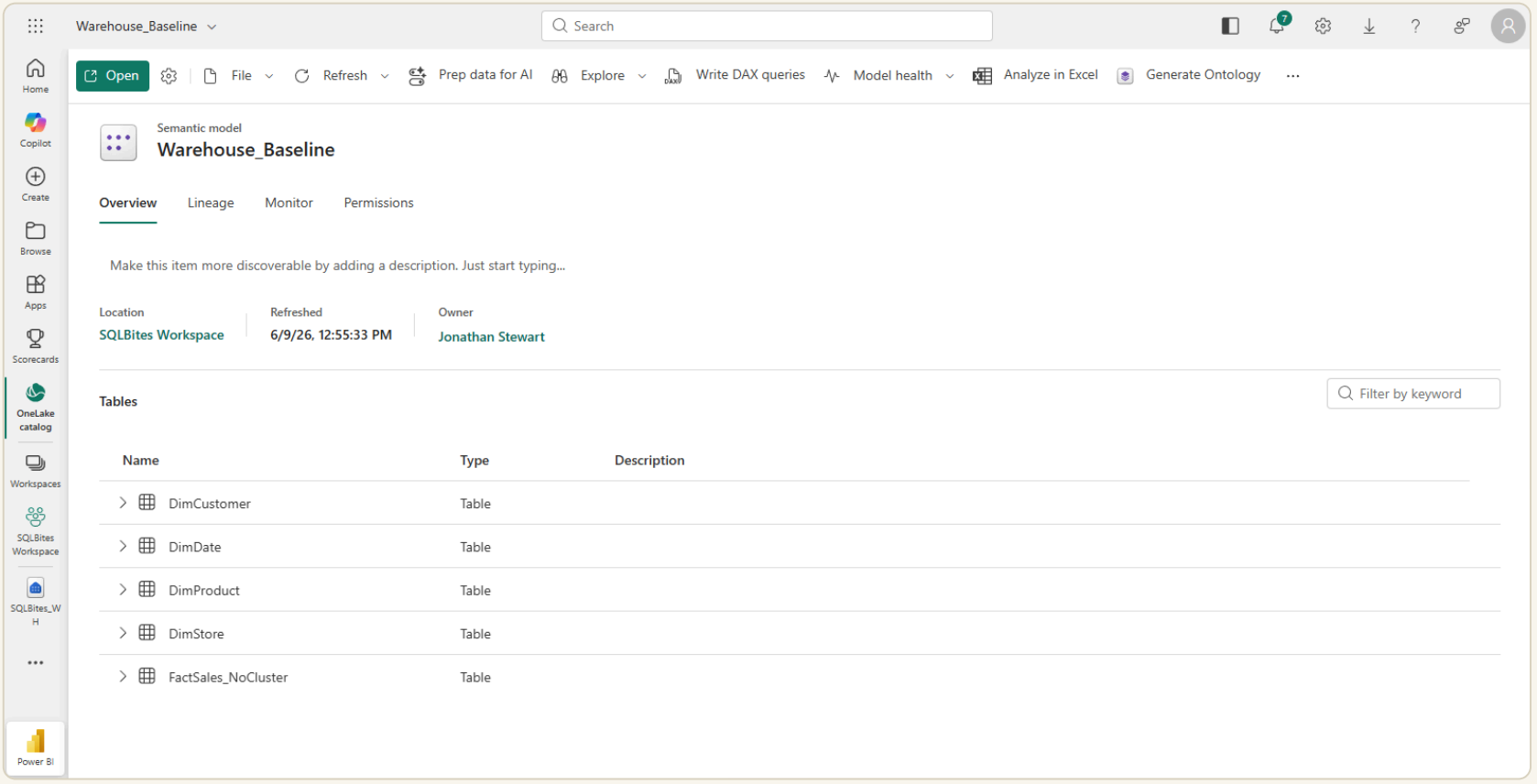
The whole star schema, ready to model



4 dims + 3 fact variants *DimCustomer 100K · DimProduct 5K · DimStore 200 · DimDate 912 · FactSales 10M*


Fabric
SQLBites_WH
Search
Home
Management
Help
Analyze data with
Share
Explorer
Warehouses
SQLBites_WH
Schemas
dbo
Tables
DimCust
DimDate
DimProd
DimStore
FactSales
FactSales
FactSales
LoadCon
StagingS
Views
Functions
Stored Pro...
Query and preview your data
You can use this editor to query and preview your data for analysis. Start with a new query.
New SQL query
Copy SQL connection string
Home
Workspaces
Copilot
OneLake catalog
Monitor
Real-Time
Workloads
SQLBites Workspace
Fabric

The Direct Lake model on that warehouse



Note what's `missing`*: no scheduled refresh, no import, no copy of 10M rows. "Refreshed" is just framing.*

Home
Copilot
Create
Browse
Apps
Scorecards
OneLake catalog
Workspaces
SQLBites Workspace
SQLBites_WH
Power BI

Open | File | Refresh | Prep data for AI | Explore | Write DAX queries | Model health | Analyze in Excel | Generate Ontology | ...

Semantic model

Warehouse_Baseline

Overview | Lineage | Monitor | Permissions

Make this item more discoverable by adding a description. Just start typing...

Location	Refreshed	Owner
SQLBites Workspace	6/9/26, 12:55:33 PM	Jonathan Stewart

Tables

Filter by keyword

Name	Type	Description
DimCustomer	Table	
DimDate	Table	
DimProduct	Table	
DimStore	Table	
FactSales_NoCluster	Table	

Four things to know before you trust it

01 **Transcoding, not importing**

Parquet columns page into memory on first touch, per column. First query warms; the rest fly.

02 **Framing replaces refresh**

The model re-points to new Delta versions when data lands. Seconds of metadata, not a data copy.

03 **Guardrails → DirectQuery fallback**

Blow the row/memory limits for your SKU and queries silently fall back to DirectQuery speed. Watch for it.

04 **The files decide your speed**

Direct Lake is only as fast as the Parquet underneath. Which is why the next demo exists.

CHAPTER TWO

CLUSTERING = REPORT SPEED

02

The 9-second fix.

Write the number down.

BASELINE LOAD · BARE WAREHOUSE

9.0s

Four warehouse changes. Zero DAX. The report never gets touched.

	WHAT WE CHANGE	YOUR NUMBER
01	**Clustering** *report speed*	_____ s
02	**MERGE** *model freshness*	_____ s
03	**Snapshots** *stable windows*	_____ s
04	**Engine choice** *what goes underneath*	_____ s

Everything from here beats it.

Scattered rows vs grouped rows.

SCATTERED

- The rows you need are in every file
- Direct Lake reads everything
- Slow render

GROUPED

- Similar rows live together
- Reads a fraction of the files
- Fast render

When a visual renders, Direct Lake reads Parquet files from the warehouse. File-skipping is the whole game.

PHYSICALLY GROUP SIMILAR ROWS TOGETHER. NOT AN INDEX.

Re-sort the filing cabinet

So a month's invoices are in one drawer instead of one-per-drawer across the whole room. It's the physical layout of the data. No index structure.

Cluster on what your report filters by.

01

Usually the date key

Range/WHERE predicates are exactly what clustering accelerates.

01

02

Mid-to-high cardinality

Equality-join keys don't benefit; pick the slicer column.

02

03

A report-author decision

You know the slicers. You have the info the DBA doesn't.

03

Data clustering is in preview (mid-2026) and is defined at table creation. You CTAS a clustered copy, you can't bolt it on.

The 9-second *fix.*

ALL FILES

9.0s

Read every file.

FILES SKIPPED	ROWS SCANNED
0	100%

FILE-SKIPPED

0.8s

Read a fraction.

DAX CHANGED	ROWS SCANNED
0	↓ 90%+

Same rows. Different drawers.

A visual asks for Q1. The engine can only skip files whose min/max ranges exclude Q1.

UNCLUSTERED · Q1 SMEARED ACROSS EVERY FILE

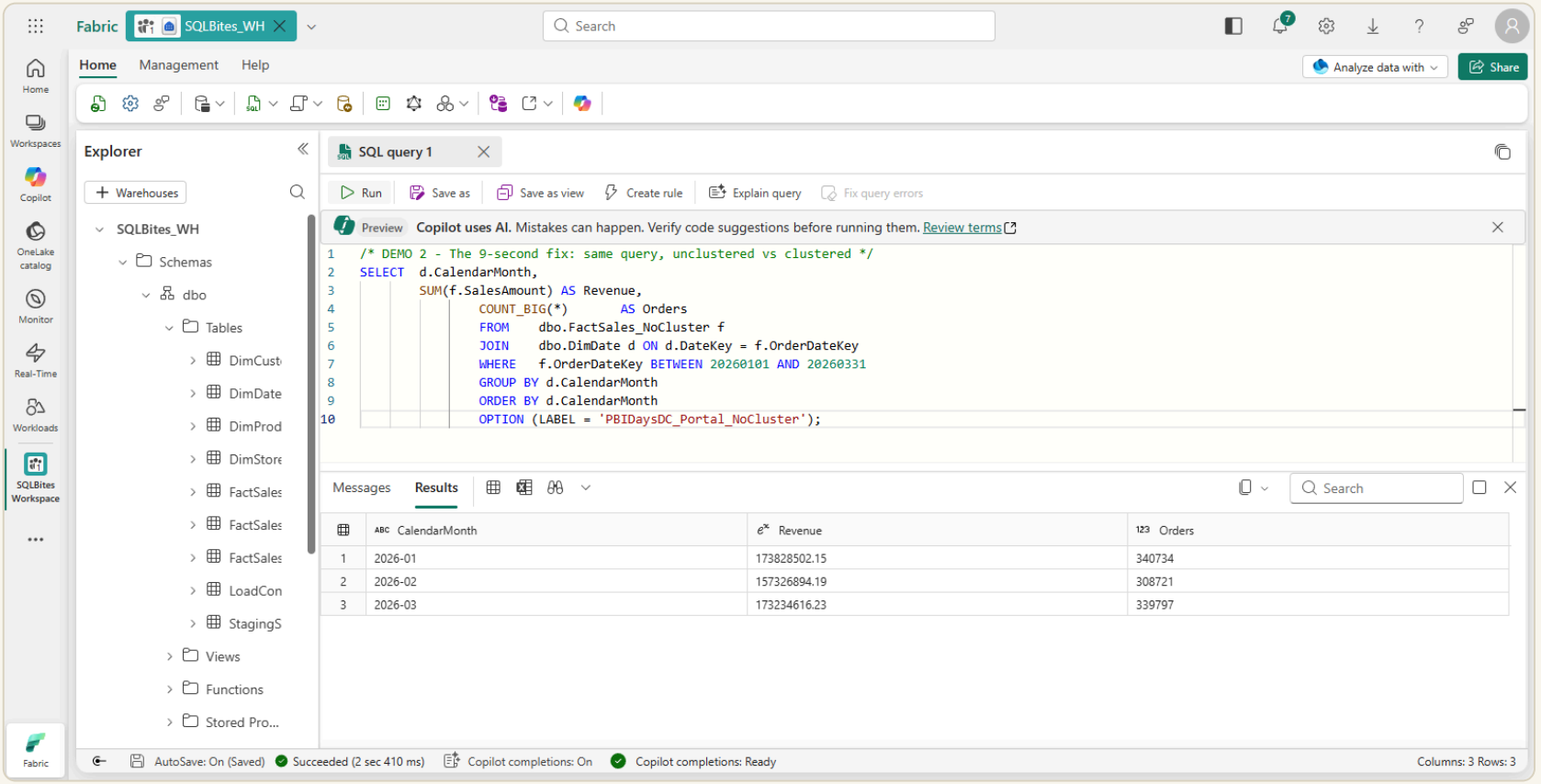
Every file holds a little Q1. ***The scan reads all of them.***

CLUSTERED ON ORDERDATEKEY · Q1 LIVES TOGETHER

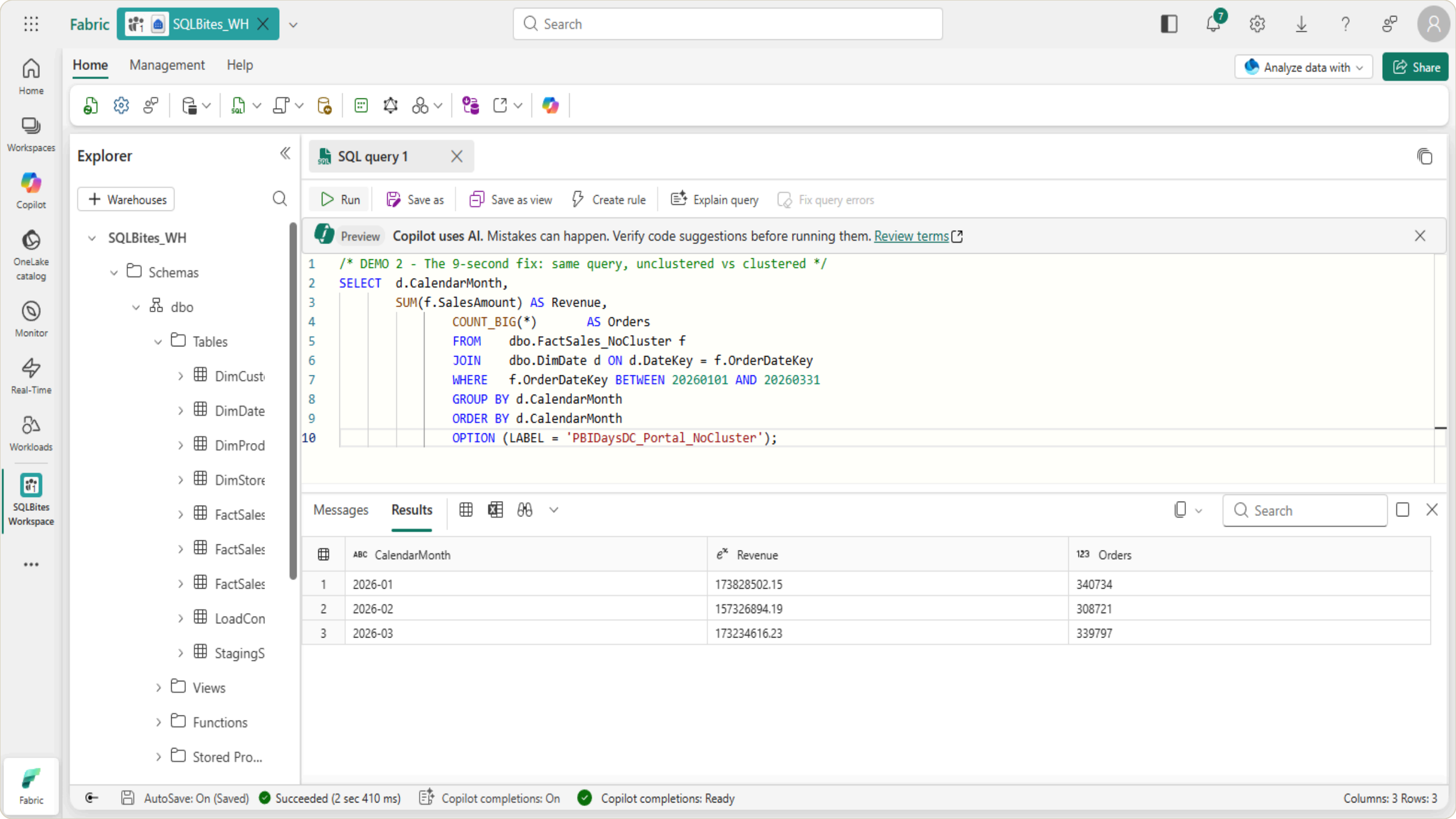
Q1 is two drawers. ***Everything else gets skipped.***

Not an index. The `physical layout` ***of rows across Parquet files, set at CTAS with*** `WITH (CLUSTER BY (OrderDateKey))`***. Preview, mid-2026.***

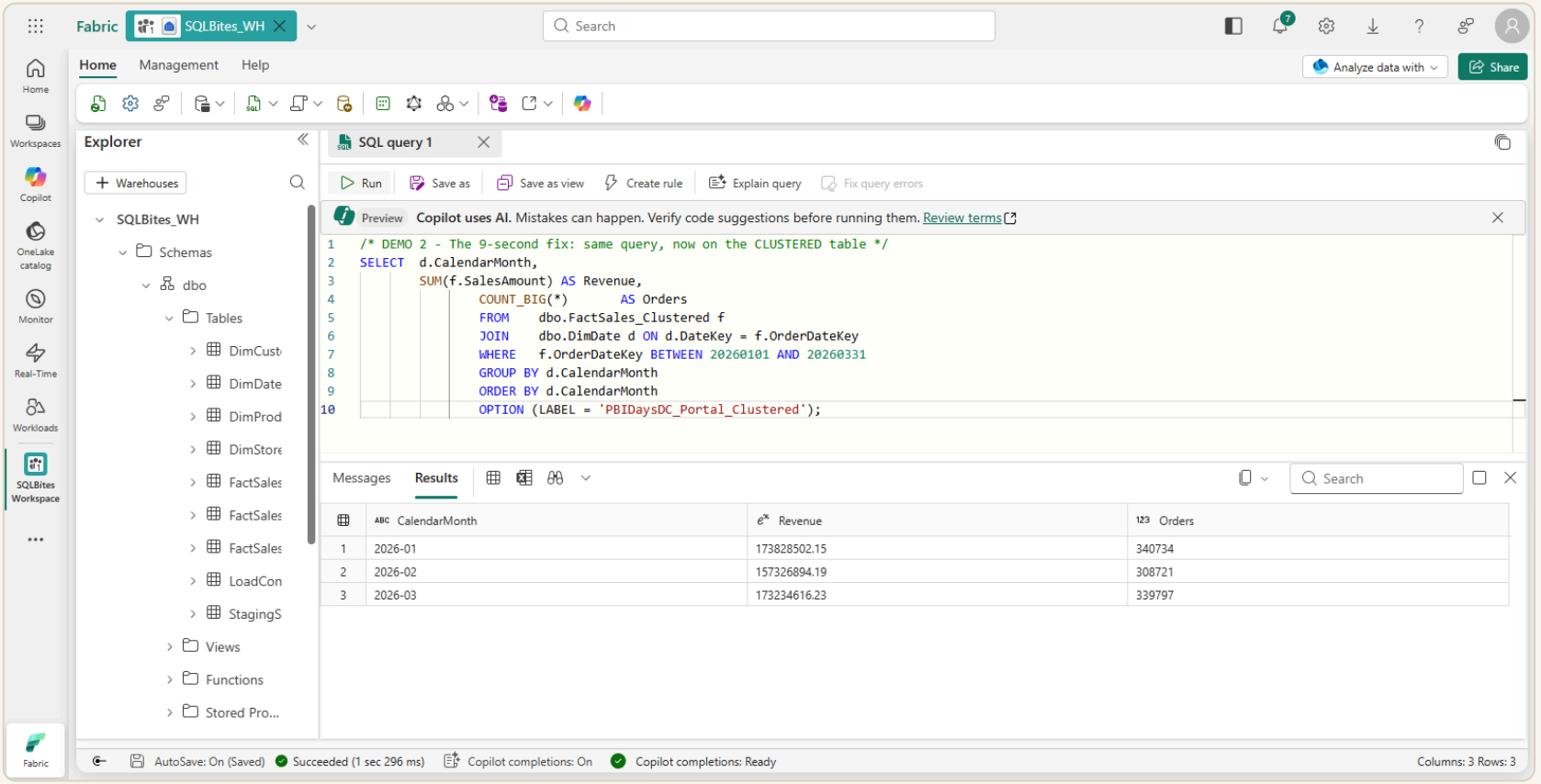
Q1 on the unclustered fact



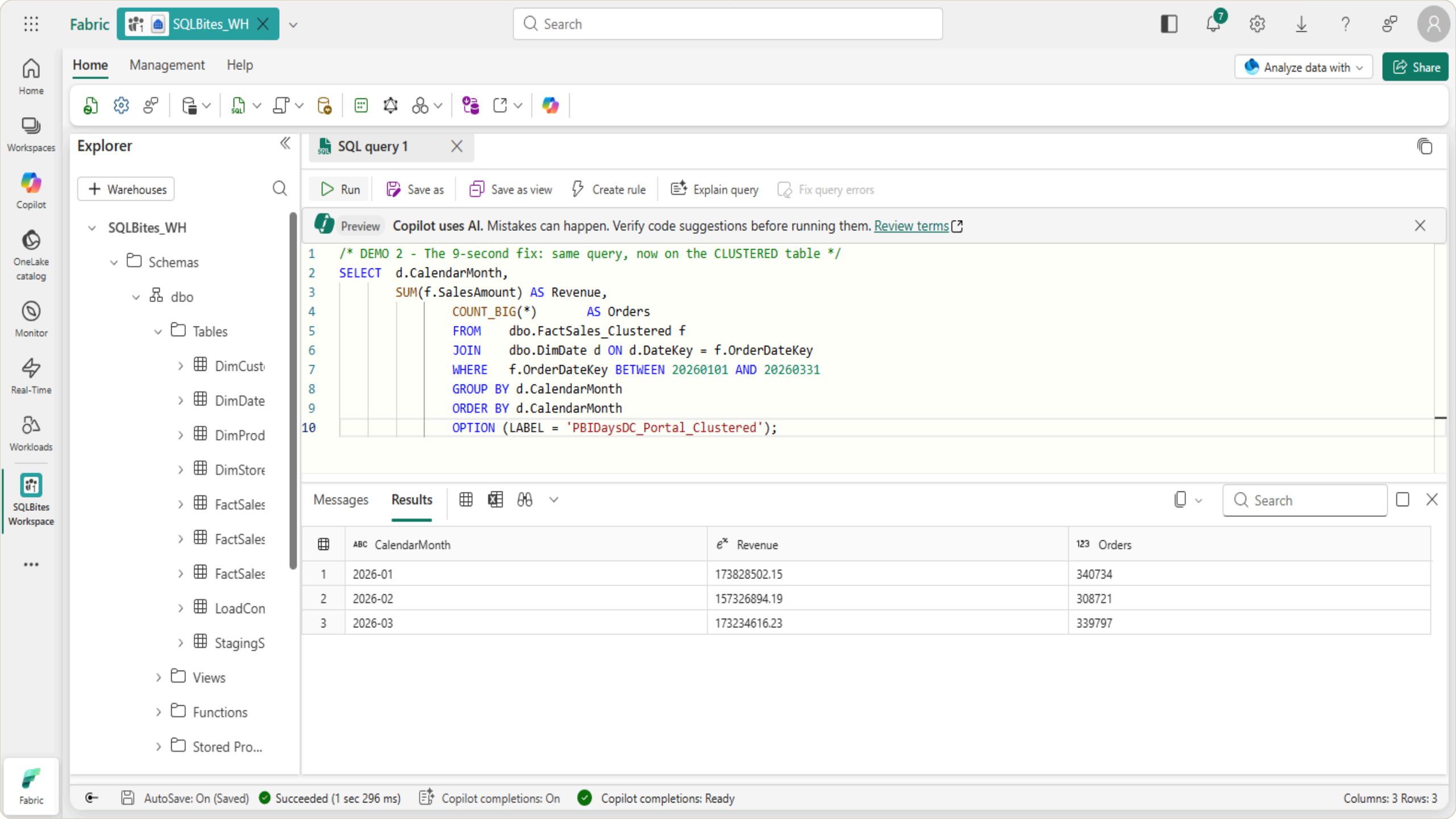
Status bar, bottom left: `Succeeded (2 sec 410 ms)`. *Cold run from a driver: 2,618 ms.*


Fabric
SQLBites_WH
Search
Home
Management
Help
Analyze data with
Share
Explorer
SQL query 1
Warehouses
SQLBites_WH
Schemas
dbo
Tables
DimCust
DimDate
DimProd
DimStore
FactSales
FactSales
FactSales
LoadCon
StagingS
Views
Functions
Stored Pro...
Run
Save as
Save as view
Create rule
Explain query
Fix query errors
Preview
Copilot uses AI. Mistakes can happen. Verify code suggestions before running them. Review terms
1 /* DEMO 2 - The 9-second fix: same query, unclustered vs clustered */
2 SELECT d.CalendarMonth,
3 SUM(f.SalesAmount) AS Revenue,
4 COUNT_BIG(*) AS Orders
5 FROM dbo.FactSales_NoCluster f
6 JOIN dbo.DimDate d ON d.DateKey = f.OrderDateKey
7 WHERE f.OrderDateKey BETWEEN 20260101 AND 20260331
8 GROUP BY d.CalendarMonth
9 ORDER BY d.CalendarMonth
10 OPTION (LABEL = 'PBIDaysDC_Portal_NoCluster');
Messages
Results
Search
CalendarMonth
Revenue
Orders
1 2026-01 173828502.15 340734
2 2026-02 157326894.19 308721
3 2026-03 173234616.23 339797
AutoSave: On (Saved)
Succeeded (2 sec 410 ms)
Copilot completions: On
Copilot completions: Ready
Columns: 3 Rows: 3

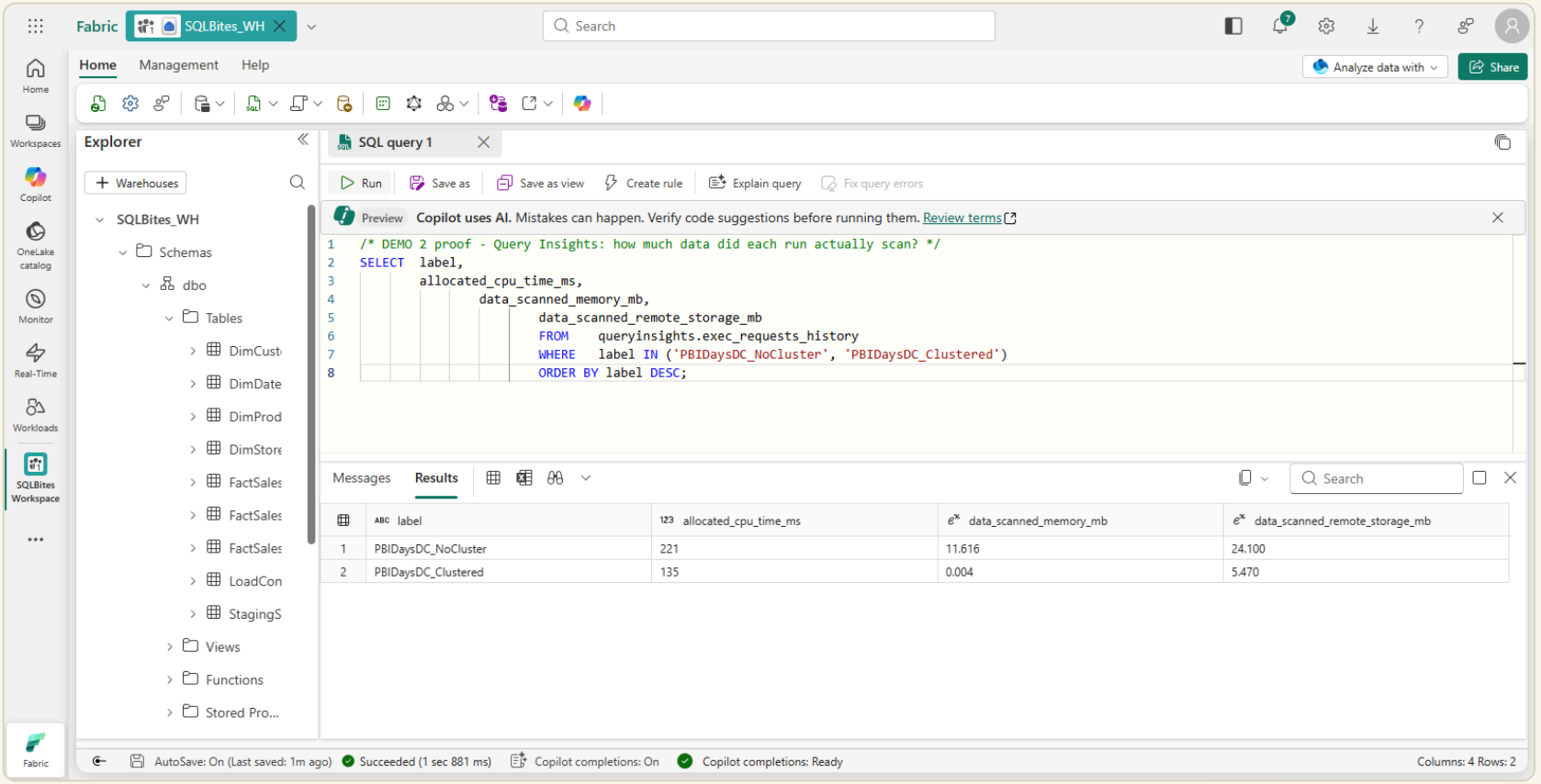
Identical query, clustered fact



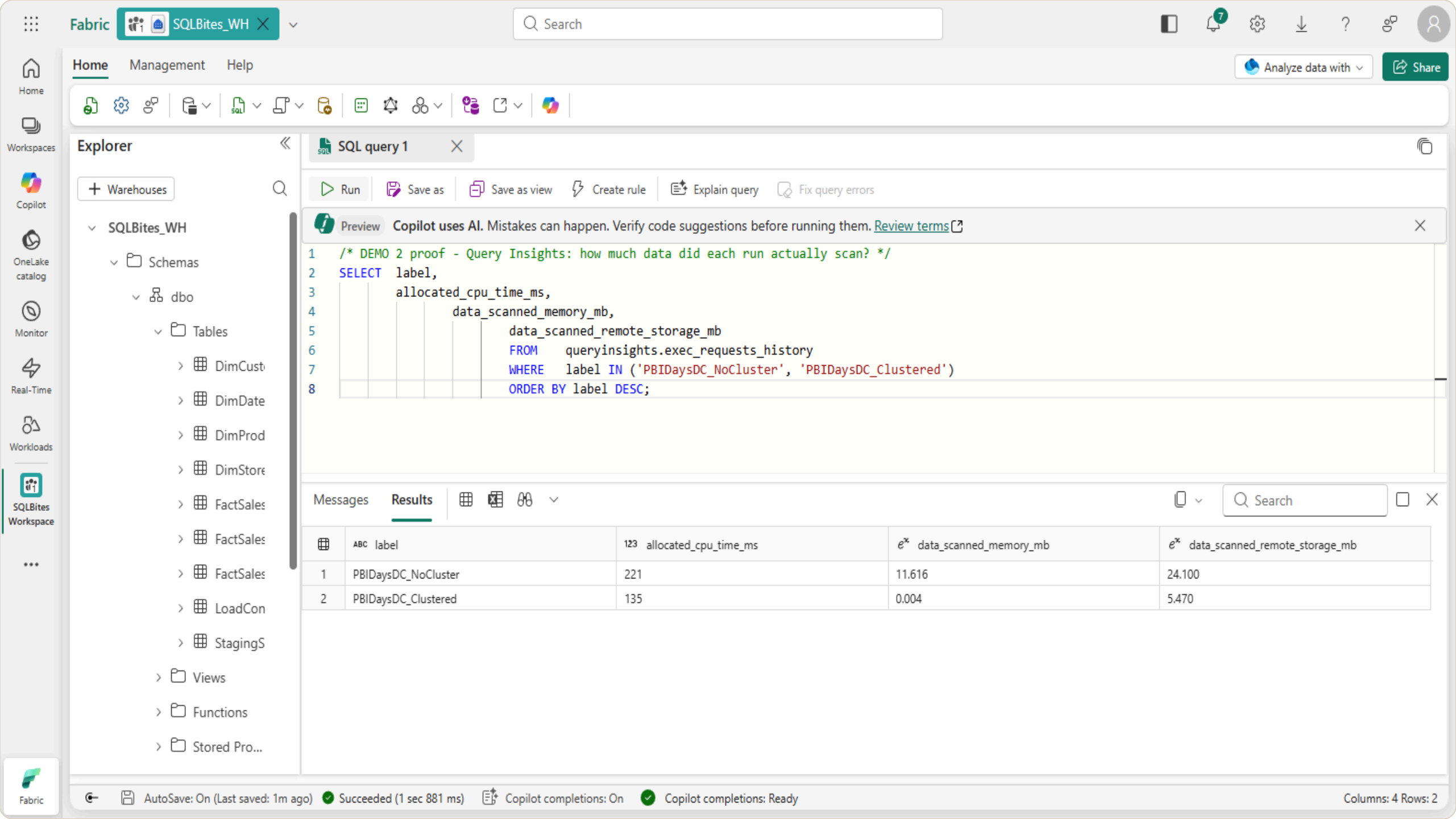
Same three months, same revenue, to the cent. Cold run: `311 ms`*. Nothing in the report changed.*


Fabric
SQLBites_WH
Search
Home
Management
Help
Analyze data with
Share
Explorer
SQL query 1
Warehouses
Run
Save as
Save as view
Create rule
Explain query
Fix query errors
Preview
Copilot uses AI. Mistakes can happen. Verify code suggestions before running them. Review terms
SQLBites_WH
Schemas
dbo
Tables
DimCust
DimDate
DimProd
DimStore
FactSales
FactSales
FactSales
LoadCon
StagingS
Views
Functions
Stored Pro...
1 /* DEMO 2 - The 9-second fix: same query, now on the CLUSTERED table */
2 SELECT d.CalendarMonth,
3 SUM(f.SalesAmount) AS Revenue,
4 COUNT_BIG(*) AS Orders
5 FROM dbo.FactSales_Clustered f
6 JOIN dbo.DimDate d ON d.DateKey = f.OrderDateKey
7 WHERE f.OrderDateKey BETWEEN 20260101 AND 20260331
8 GROUP BY d.CalendarMonth
9 ORDER BY d.CalendarMonth
10 OPTION (LABEL = 'PBIDaysDC_Portal_Clustered');
Messages
Results
Search
CalendarMonth
Revenue
Orders
1 2026-01 173828502.15 340734
2 2026-02 157326894.19 308721
3 2026-03 173234616.23 339797
AutoSave: On (Saved)
Succeeded (1 sec 296 ms)
Copilot completions: On
Copilot completions: Ready
Columns: 3 Rows: 3

The proof underneath: data scanned



Remote storage scanned 24.1 MB → 5.5 MB · *memory scanned* 11.6 MB → 0.004 MB. *The report read a fraction.*


Fabric
SQLBites_WH
Search
Home
Management
Help
Analyze data with
Share
Explorer
SQL query 1
Warehouses
Run
Save as
Save as view
Create rule
Explain query
Fix query errors
Preview
Copilot uses AI. Mistakes can happen. Verify code suggestions before running them. Review terms
/* DEMO 2 proof - Query Insights: how much data did each run actually scan? */
SELECT label,
allocated_cpu_time_ms,
data_scanned_memory_mb,
data_scanned_remote_storage_mb
FROM queryinsights.exec_requests_history
WHERE label IN ('PBIDaysDC_NoCluster', 'PBIDaysDC_Clustered')
ORDER BY label DESC;
Messages
Results
Search
label
allocated_cpu_time_ms
data_scanned_memory_mb
data_scanned_remote_storage_mb
1 PBIDaysDC_NoCluster 221 11.616 24.100
2 PBIDaysDC_Clustered 135 0.004 5.470
SQLBites_WH
Schemas
dbo
Tables
Views
Functions
Stored Pro...
AutoSave: On (Last saved: 1m ago)
Succeeded (1 sec 881 ms)
Copilot completions: On
Copilot completions: Ready
Columns: 4 Rows: 2

One clustering key. Zero DAX.

COLD QUERY TIME

2,618 ms

311 ms

8.4× faster, measured this week, 10M-row fact

REMOTE DATA SCANNED

24.1 MB

5.5 MB

file skipping on the OrderDateKey range

MEMORY SCANNED

11.6 MB

0.004 MB

2,900× less touched in the warm path

Same result set to the cent: `$504,390,012.57 · 989,252 orders`***. Only the physical layout changed.***

When clustering won't save you

01 **Very low-cardinality columns**

Cluster by a 3-value status → no file pruning.

02 **Tiny tables**

Scan cost is already negligible.

03 **A column nobody filters by**

Wasted ingestion overhead, zero report benefit.

BEFORE YOU REWRITE ANOTHER MEASURE:

Is my fact table clustered on what my report filters by?

This one question fixes a huge share of "slow report" tickets.

CHAPTER THREE

MERGE = MODEL FRESHNESS

03

Current data, no full reload.

Rebuild the whole table. Or merge what changed.

FULL RELOAD

- Nightly rebuild of the whole fact
- Slow
- Locks the table your report reads
- Numbers a day stale

INCREMENTAL MERGE

- Only what changed since last load
- Seconds, not minutes
- No big lock window
- Fresher report, sooner

Direct Lake killed the model refresh. But the warehouse still has to get new rows in.

The watermark pattern

01	Track a high-water mark	*Last load timestamp or key, in a control table.*
02	Pull only newer rows	*"Give me rows newer than this."*
03	MERGE	*Update what matched, insert what's new. One statement, no truncate-and-reload.*

Incremental MERGE

MERGE is GA in Fabric Data Warehouse (since Jan 2026).

```
MERGE INTO dbo.FactSales_Clustered AS tgt
USING (SELECT * FROM dbo.StagingSales
       WHERE OrderDateKey > @watermark) AS src
  ON tgt.SalesKey = src.SalesKey
WHEN MATCHED THEN UPDATE SET
  tgt.SalesAmount = src.SalesAmount
WHEN NOT MATCHED BY TARGET THEN
  INSERT (...) VALUES (...);
```

WATERMARK FILTER

Only the new day crosses the wire.

ONE STATEMENT

Update matched, insert new.

Seconds, not a full rebuild.

File the new invoices. Don’t rebuild the cabinet.

1 · LOADCONTROL

Watermark 20260628

One row remembers how far you’ve loaded.

→

2 · STAGINGSALES

Filter > watermark

Only the new day survives the WHERE clause: 10,931 rows, not 10 million.

→

3 · MERGE (GA SINCE JAN 2026)

Update matched, insert new

Restatements update in place; new rows insert. One statement, no truncate.

→

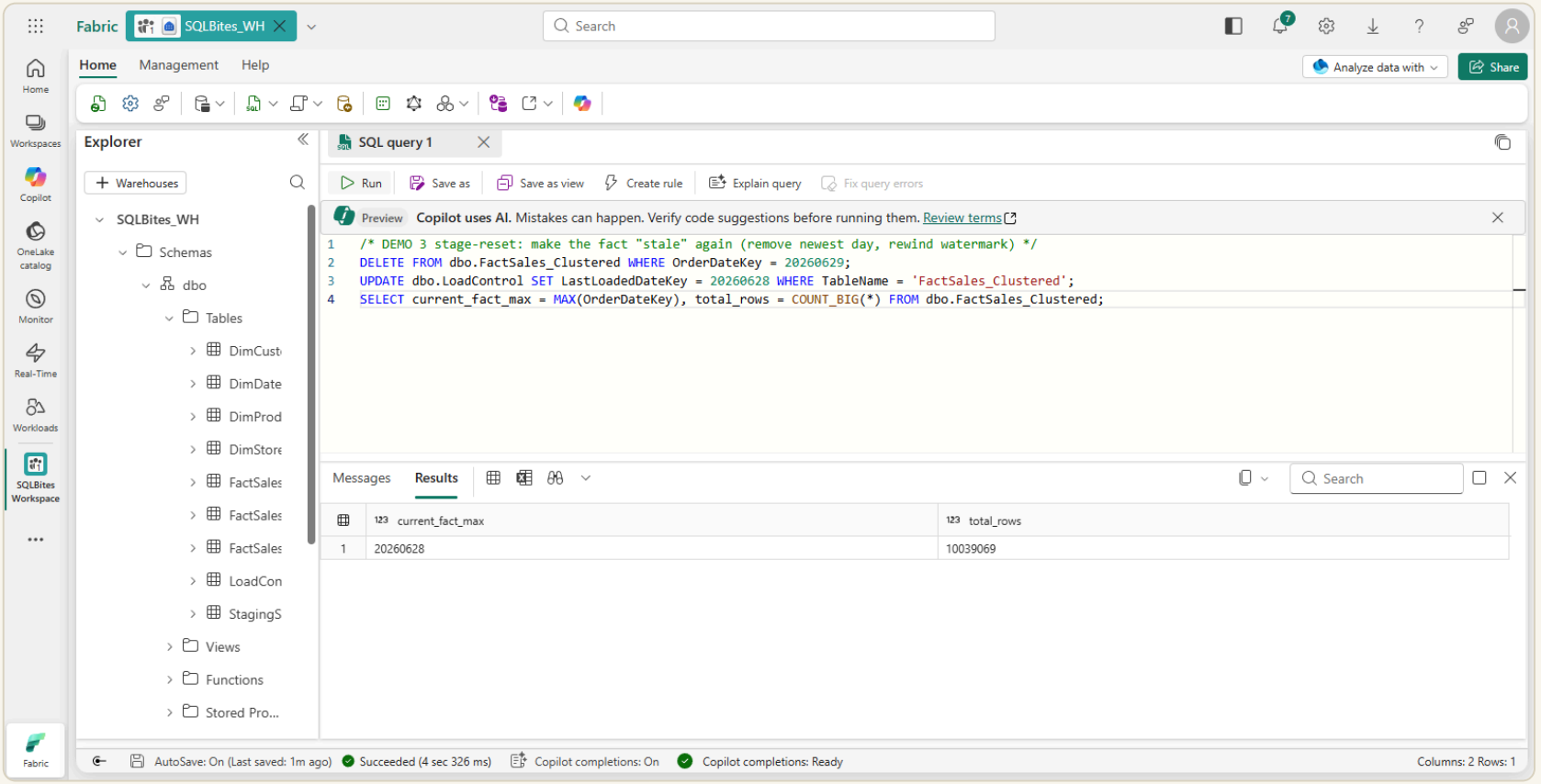
4 · ADVANCE

Watermark 20260629

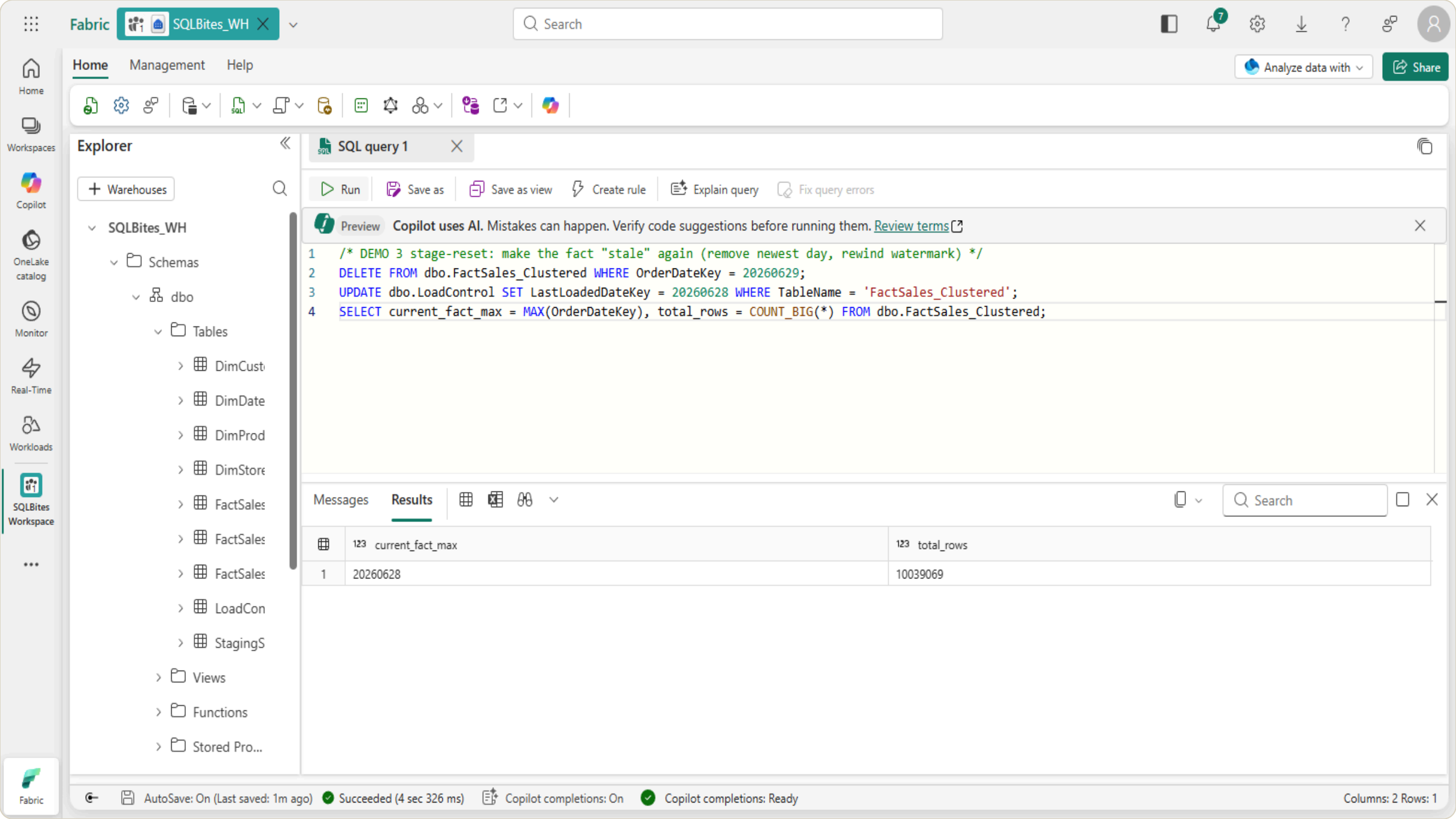
Ready for tomorrow’s day.

Locking truth: MERGE takes an `IX` table lock, readers take `Sch-S` under snapshot isolation. `Reports are never blocked` ***while the load runs.***

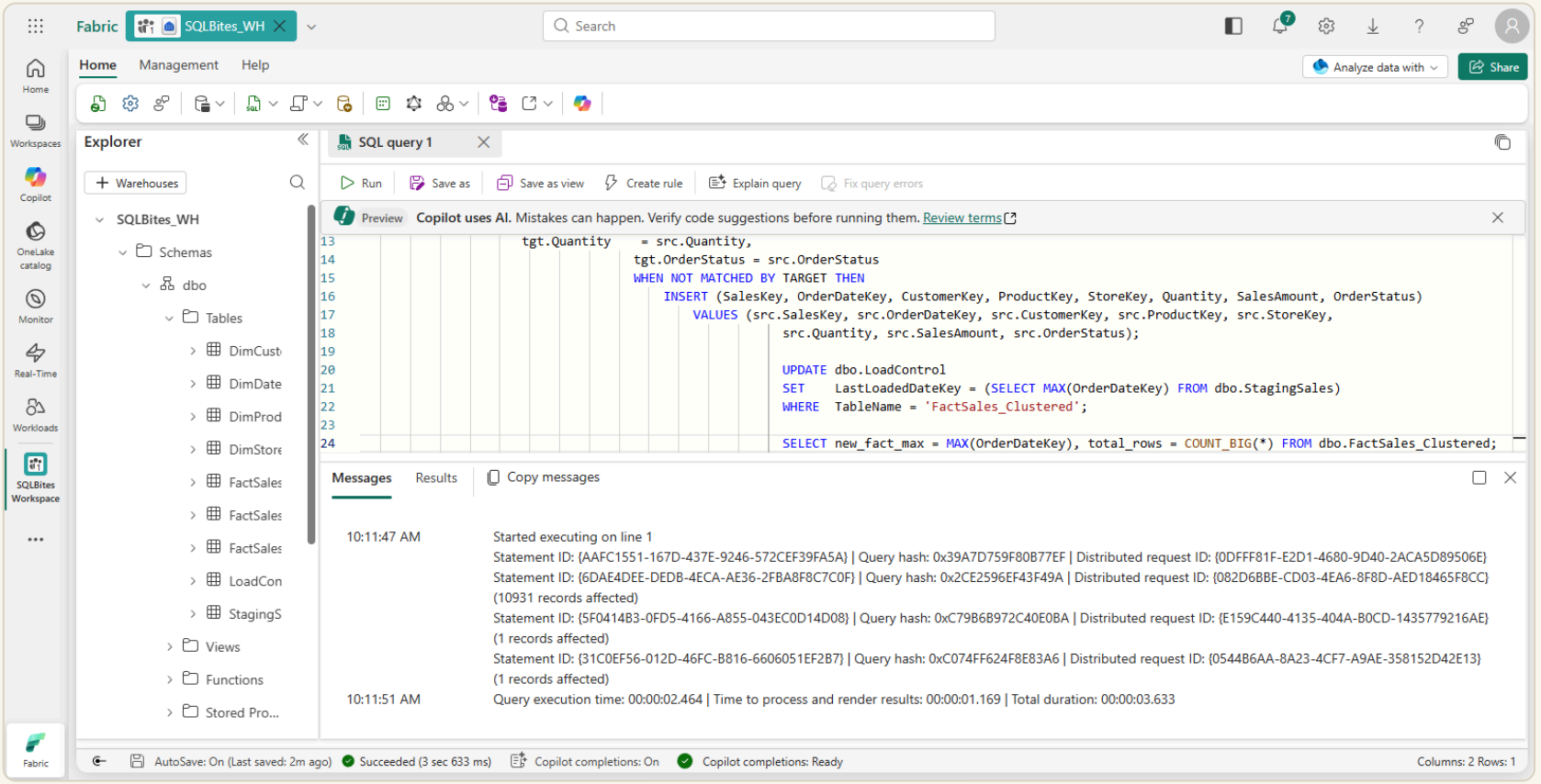
The report is a day behind



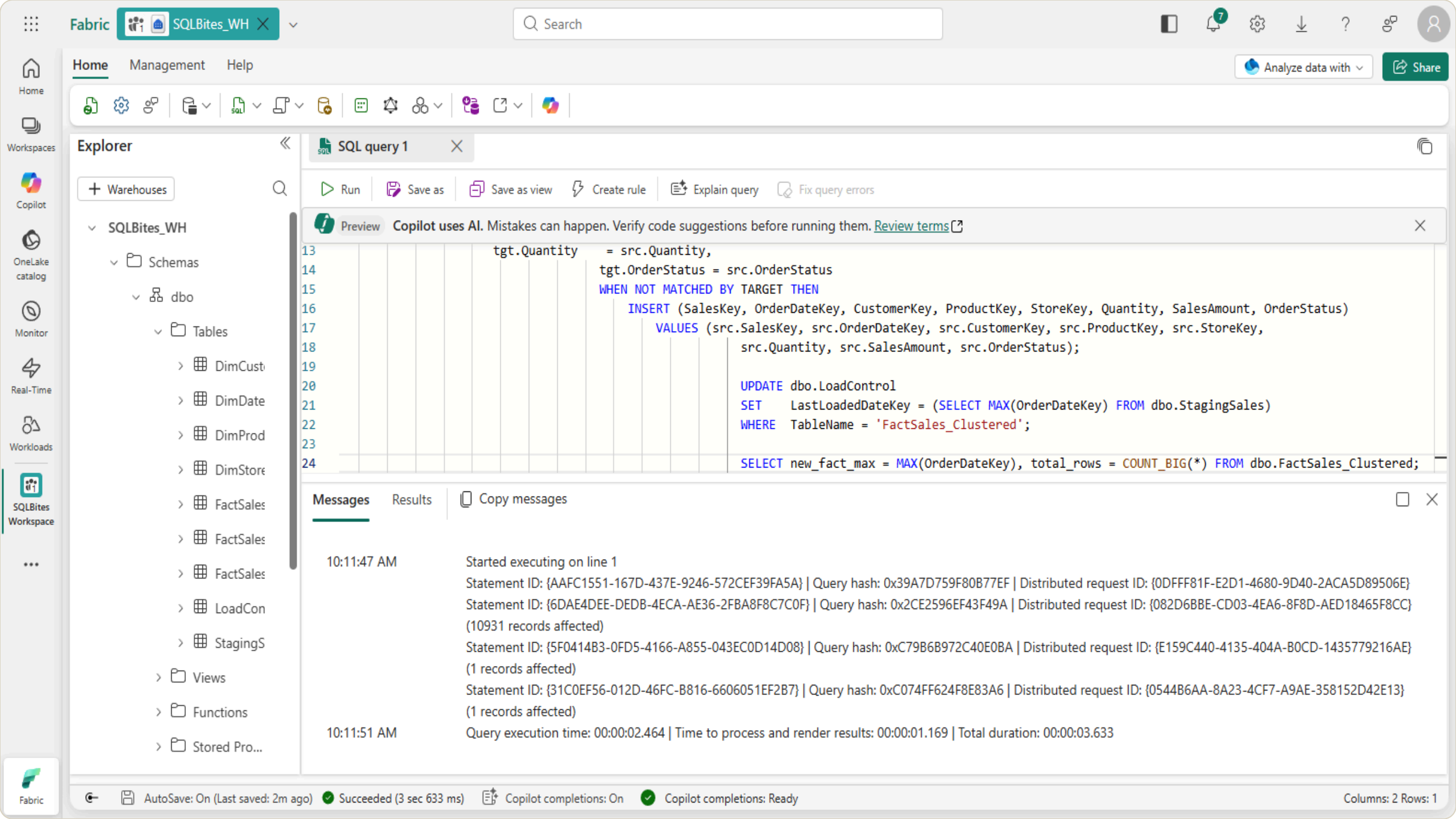
Fact max is 20260628 with 10,039,069 rows. Yesterday's orders exist only in staging. The exec sees stale cards.


Fabric
SQLBites_WH
Search
Home
Management
Help
Analyze data with
Share
Explorer
SQL query 1
Warehouses
Run
Save as
Save as view
Create rule
Explain query
Fix query errors
Preview
Copilot uses AI. Mistakes can happen. Verify code suggestions before running them. Review terms
1 /* DEMO 3 stage-reset: make the fact "stale" again (remove newest day, rewind watermark) */
2 DELETE FROM dbo.FactSales_Clustered WHERE OrderDateKey = 20260629;
3 UPDATE dbo.LoadControl SET LastLoadedDateKey = 20260628 WHERE TableName = 'FactSales_Clustered';
4 SELECT current_fact_max = MAX(OrderDateKey), total_rows = COUNT_BIG(*) FROM dbo.FactSales_Clustered;
SQLBites_WH
Schemas
dbo
Tables
DimCust
DimDate
DimProd
DimStore
FactSales
FactSales
FactSales
LoadCon
StagingS
Views
Functions
Stored Pro...
Messages
Results
Search
current_fact_max
total_rows
1
20260628
10039069
AutoSave: On (Last saved: 1m ago)
Succeeded (4 sec 326 ms)
Copilot completions: On
Copilot completions: Ready
Columns: 2 Rows: 1

One MERGE later



(10931 records affected) · *execution* 2.46 s · *fact max now* 20260629, *back to 10,050,000 rows. No reload. No lock window. No model refresh.*


Fabric
SQLBites_WH
Search
Home
Management
Help
Analyze data with
Share
Explorer
SQL query 1
Warehouses
SQLBites_WH
Schemas
dbo
Tables
DimCust
DimDate
DimProd
DimStore
FactSales
FactSales
FactSales
LoadCon
StagingS
Views
Functions
Stored Pro...
Run
Save as
Save as view
Create rule
Explain query
Fix query errors
Preview
Copilot uses AI. Mistakes can happen. Verify code suggestions before running them. Review terms
tgt.Quantity = src.Quantity,
tgt.OrderStatus = src.OrderStatus
WHEN NOT MATCHED BY TARGET THEN
INSERT (SalesKey, OrderDateKey, CustomerKey, ProductKey, StoreKey, Quantity, SalesAmount, OrderStatus)
VALUES (src.SalesKey, src.OrderDateKey, src.CustomerKey, src.ProductKey, src.StoreKey,
src.Quantity, src.SalesAmount, src.OrderStatus);
UPDATE dbo.LoadControl
SET LastLoadedDateKey = (SELECT MAX(OrderDateKey) FROM dbo.StagingSales)
WHERE TableName = 'FactSales_Clustered';
SELECT new_fact_max = MAX(OrderDateKey), total_rows = COUNT_BIG(*) FROM dbo.FactSales_Clustered;
Messages
Results
Copy messages
10:11:47 AM
Started executing on line 1
Statement ID: {AAFC1551-167D-437E-9246-572CEF39FA5A} | Query hash: 0x39A7D759F80B77EF | Distributed request ID: {0DFFF81F-E2D1-4680-9D40-2ACA5D89506E}
Statement ID: {6DAE4DEE-DEDB-4ECA-AE36-2FBA8F8C7C0F} | Query hash: 0x2CE2596EF43F49A | Distributed request ID: {082D6BBE-CD03-4EA6-8F8D-AED18465F8CC}
(10931 records affected)
Statement ID: {5F0414B3-0FD5-4166-A855-043EC0D14D08} | Query hash: 0xC79B6B972C40E0BA | Distributed request ID: {E159C440-4135-404A-B0CD-1435779216AE}
(1 records affected)
Statement ID: {31C0EF56-012D-46FC-B816-6606051EF2B7} | Query hash: 0xC074FF624F8E83A6 | Distributed request ID: {0544B6AA-8A23-4CF7-A9AE-358152D42E13}
(1 records affected)
10:11:51 AM
Query execution time: 00:00:02.464 | Time to process and render results: 00:00:01.169 | Total duration: 00:00:03.633
AutoSave: On (Last saved: 2m ago)
Succeeded (3 sec 633 ms)
Copilot completions: On
Copilot completions: Ready
Columns: 2 Rows: 1

What makes this safe to run at 9am

01	**GA since January 2026**	*Preview Sept 2025, GA Jan 2026. If your linter says "MERGE restricted," the rule pack predates GA.*
02	**Readers never wait**	*Table-level locking: MERGE holds IX, SELECTs hold Sch-S, and snapshot isolation serves readers the prior consistent version.*
03	**Direct Lake picks it up via framing**	*No semantic-model refresh job. The new Delta version is what the next visual reads.*
04	**The one gotcha: duplicate keys**	*Duplicate SalesKeys in staging break the MERGE contract. Dedupe with ROW_NUMBER() OVER (PARTITION BY key) first.*

Direct Lake killed one refresh, not the other.

KILLED	STILL REAL
01 The MODEL refresh	01 The WAREHOUSE load
02 No data-move into the model	02 New rows must still land
03 No scheduled import	03 Incremental MERGE keeps it cheap

vs.

"Direct Lake = no refresh = nothing to manage" is wrong. The warehouse load is still real.

CHAPTER FOUR

SNAPSHOTS = STABLE WINDOWS

04

The number that doesn't move.

"WHY IS LAST MONTH'S REVENUE DIFFERENT FROM LAST WEEK'S NUMBER?"

Because the warehouse kept loading

Late-arriving rows changed your "May" report. You didn't do anything wrong. The number is a moving target.

A FROZEN, POINT-IN-TIME WINDOW.

Read-only. The number never moves.

Point the period-end report at the snapshot and it's immutable, even as the live warehouse keeps changing.

The pattern you'll actually use

01	**Snapshot at month-end close**	*Created via the Fabric portal / REST API.*
02	**Official report → the snapshot**	*The locked number, immutable.*
03	**Live warehouse keeps loading**	*For current-month reporting. Free audit trail.*

The live number drifts. The official one holds.

MONTH-END CLOSE

SNAPSHOT · TAKEN AT MONTH-END CLOSE

Read-only, point-in-time

Full DQL, zero DML. Late-arriving rows cannot reach it. The exec's May number is carved in stone.

LIVE WAREHOUSE · KEEPS LOADING

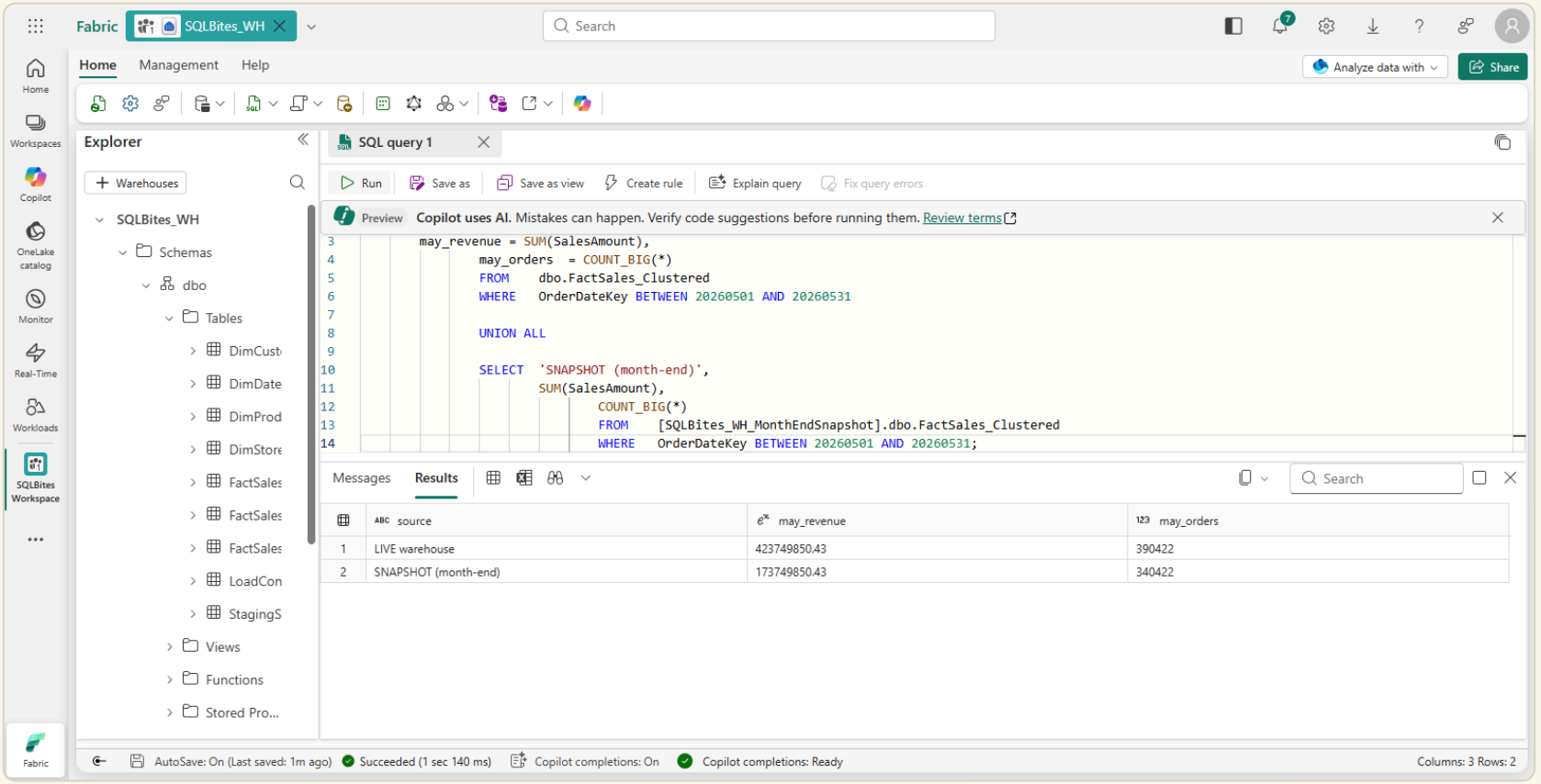
Late rows keep landing

Current-month reporting stays live and fresh. May keeps "moving" here, and that's fine.

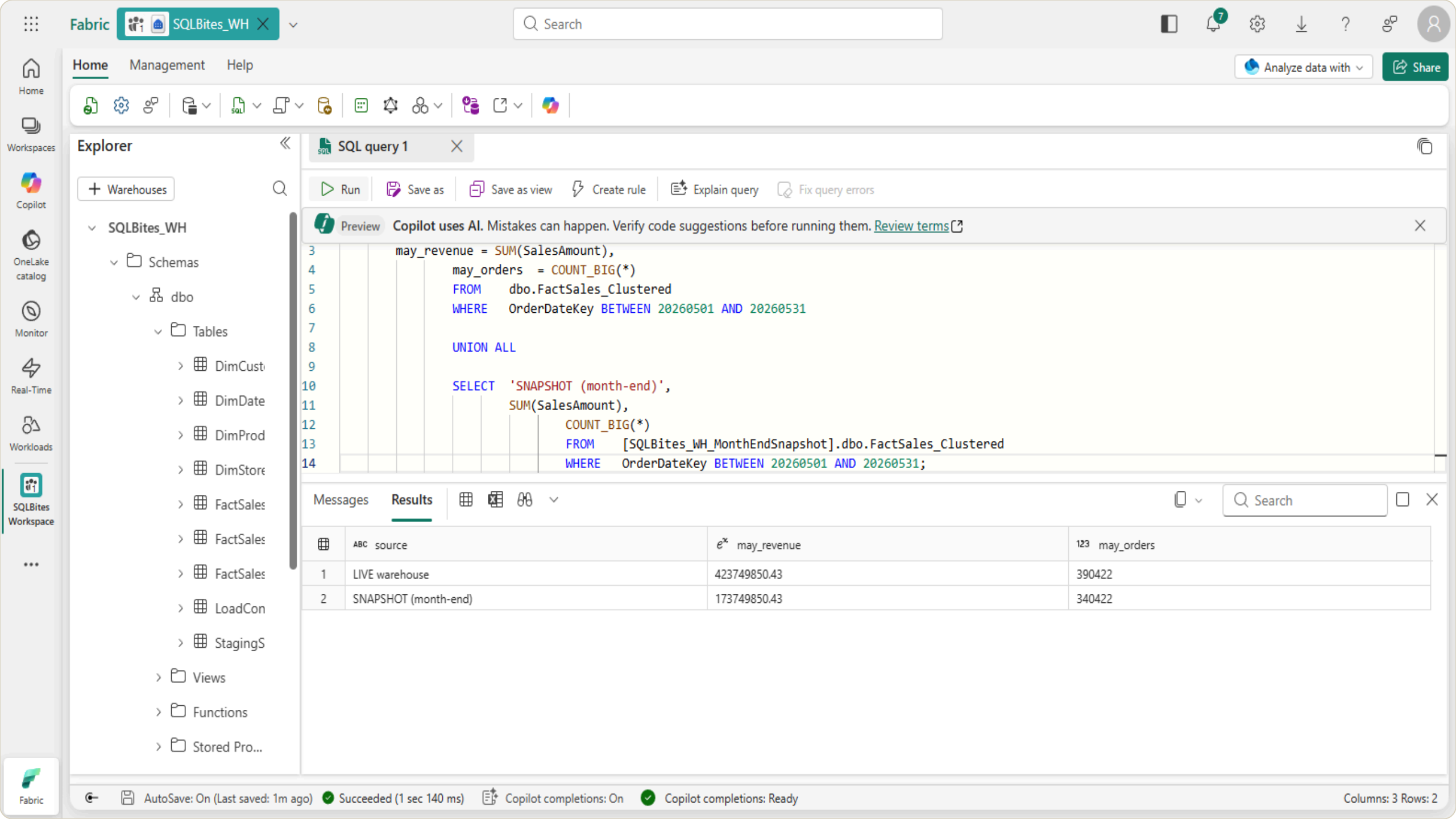
Roll the official number forward only when YOU decide: `ALTER DATABASE [snapshot] SET TIMESTAMP = CURRENT_TIMESTAMP;`

Created from the portal or REST (not T-SQL) · retention via `TIME_TRAVEL_RETENTION_PERIOD` *(30 days) · queryable cross-database by* `3-part name`*.*

May revenue, live vs snapshot



50,000 late-May rows landed after close. Live moved to $423.7M. The snapshot still answers $173.7M, from the same connection, via its 3-part name.


Fabric
SQLBites_WH
Search
Home
Management
Help
Analyze data with
Share
Explorer
SQL query 1
Warehouses
Run
Save as
Save as view
Create rule
Explain query
Fix query errors
Preview
Copilot uses AI. Mistakes can happen. Verify code suggestions before running them. Review terms
may_revenue = SUM(SalesAmount),
may_orders = COUNT_BIG(*)
FROM dbo.FactSales_Clustered
WHERE OrderDateKey BETWEEN 20260501 AND 20260531
UNION ALL
SELECT 'SNAPSHOT (month-end)',
SUM(SalesAmount),
COUNT_BIG(*)
FROM [SQLBites_WH_MonthEndSnapshot].dbo.FactSales_Clustered
WHERE OrderDateKey BETWEEN 20260501 AND 20260531;
Messages
Results
Search
source
may_revenue
may_orders
1 LIVE warehouse 423749850.43 390422
2 SNAPSHOT (month-end) 173749850.43 340422
SQLBites_WH
Schemas
dbo
Tables
Views
Functions
Stored Pro...
AutoSave: On (Last saved: 1m ago)
Succeeded (1 sec 140 ms)
Copilot completions: On
Copilot completions: Ready
Columns: 3 Rows: 2

"Why did last month change?" Never again.

LIVE WAREHOUSE · MAY 2026

$423,749,850

390,422 orders · moved when 50,000 late rows landed

SNAPSHOT · MAY 2026

$173,749,850

340,422 orders · identical before and after the late load

Point the official month-end report at the snapshot. Keep the live report on the warehouse. Two reports, one fact, free audit trail.

CHAPTER FIVE

WHICH ENGINE GOES UNDERNEATH

05

Warehouse vs Lakehouse vs SQL DB - pick by three questions.

Which engine goes under your model?

ENGINE	BEST WHEN	MERGE · CLUSTER · SNAPSHOT
FABRIC WAREHOUSE	T-SQL ETL, you own the load	✓ best default here
LAKEHOUSE (SQL ENDPOINT)	Data lands as files / Spark	You consume what engineering produces
FABRIC SQL DATABASE	OLTP app-of-record + light reporting	Data born transactionally

Pick by three questions

01

Who owns the load?

You (Warehouse) vs engineering (Lakehouse).

01

02

T-SQL or Spark/files?

T-SQL ETL → Warehouse.

02

03

Need MERGE + clustering + snapshots?

Then you want Warehouse.

03

Not dogma. A decision tree.

FROM TUNING WHAT YOU CAN SEE → FIXING THE LAYER UNDERNEATH.

One architecture. End to end. Yours.

Four changes, all storage-layer, zero DAX. You now own Warehouse → Direct Lake → report as one thing.

5 things for Monday

01	**Audit your slowest report**	*Is its fact clustered on the date key it filters by?*
02	**CTAS a clustered copy**	*WITH (CLUSTER BY (DateKey)). Can't add in place. Repoint, re-measure.*
03	**Confirm Direct Lake**	*No import, on a warehouse.*
04	**Convert one full-reload to MERGE**	*Incremental + watermark.*
05	**Snapshot before month-end**	*Point the official report at it.*

#1 alone is worth the session.

Questions I always get

Q.01

Q

Does clustering need a full rebuild?

No - re-cluster on the next write.

Q.02

Q

Direct Lake vs Import for this?

Direct Lake - the storage layer IS the model.

Q.03

Q

Snapshots: how much storage?

Delta - only the changed rows version.

Q.04

Q

Warehouse or Lakehouse underneath?

Depends on three questions - see the framework.

Take it with you.

SQLLOCKS

SQLBites.net - sqllocks@sqlbites.com