

ELLYPUNK

Applied

WEEKLY DIGEST / T201

24-30 August 2026



This week

24-30 August 2026

Available archived editions, in their original date order.

MONDAY / 24 August 2026

Data Contracts

4

[Open original online](#)

TUESDAY / 25 August 2026

FHIR Search Paging

9

[Open original online](#)

WEDNESDAY / 26 August 2026

Power Query Folding

14

[Open original online](#)

THURSDAY / 27 August 2026

Power Bi Incremental Refresh Process Only What Changed

19

[Open original online](#)

FRIDAY / 28 August 2026

Sql Explain Read The Query Plan

24

[Open original online](#)

Only retained editions are included; gaps have not been filled.

Copyright Ellypunk. Contains AI-generated artwork.

ellypunk.com

This week

24-30 August 2026

Available archived editions, in their original date order.

SATURDAY / 29 August 2026

Golden Hour Phone Photos Light Before Filters

29

[Open original online](#)

SUNDAY / 30 August 2026

Watercolour Skies Let The Water Work

34

[Open original online](#)

DATA CONTRACTS

CATCH SCHEMA DRIFT EARLY



OBJECTIVE

Define a small **contract**, **validate** incoming data, and **stop** bad changes before they spread.

A data pipeline should know what **valid** means before it loads a row.

INPUT → VALIDATE → LOAD

A CONTRACT SETS EXPECTATIONS



Columns



Types



Required fields



Allowed values

INPUT



Incoming data arrives.

VALIDATE



Check data against the contract.

LOAD



Only valid data gets loaded.



Agree first. *Validate* every run.

THE CONTRACT WORKFLOW

PAGE 2

1. AGREE

Producer and consumer define valid data.



2. EXPRESS

Write the rules as versioned config.



3. VALIDATE

Check data at the pipeline boundary.



4. STOP AND EXPLAIN

Quarantine bad rows and report exact failures.



5. EVOLVE

Review and version intentional changes.



SMALL CONTRACT, CLEAR RULES

```
version: 1
required: [order_id, amount]
types:
  order_id: string
  amount: decimal
```



An explicit **contract** turns assumptions into **tests**.



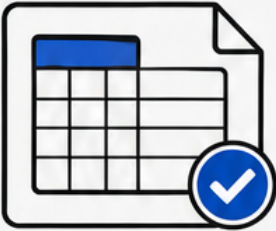
CLEAR CONTRACTS.
CONFIDENT DATA.



WHAT A GOOD CONTRACT NEEDS



1. SCHEMA



Column names and data types.

2. QUALITY



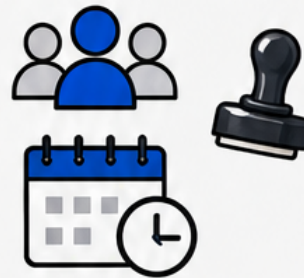
Required, unique, and valid ranges.

3. COMPATIBILITY



Which changes are *safe* or *breaking*.

4. OWNERSHIP



Who *approves*, *versions*, and *responds*.

GUARDRAILS



Reject missing *required* columns.



Allow new *optional* columns.



Never coerce *silently*.



Keep the *raw input*.

USUALLY SAFE



Add an *optional* column.

VS

BREAKING CHANGE



Rename, drop, or *change a type*.

Good contracts are strict about *meaning*, not rigid about *growth*.



WORKED EXAMPLE: ORDERS

PAGE 4

1. PROBLEM

CONTRACT

`order_id`: required string
`amount`: decimal
`status`: new | paid | shipped



INCOMING

A101	19.95	paid
(blank)	"8.50"	done



2. ACTION



Validate before load.



Keep A101.



Quarantine the bad row.



Keep the raw input.

VALIDATION CONSOLE

INCOMING ROWS: 2

✓	ACCEPTED	1
🔒	QUARANTINED	1
📄	RAW PRESERVED	2



3. CHECK



Accepted: 1



Quarantined: 1



3 exact failures



`order_id` is required



`amount` must be decimal



`status` "done" is not allowed

ROW 2 (QUARANTINED)

```
{
  "order_id": null,
  "amount": "8.50",
  "status": "done"
}
```

STATUS: FAILED
3 FAILURES

4. IMPROVEMENT

Producer fixes the row:

A102	8.50	shipped
------	------	---------

RE-RUN: PASS



Loaded: 2 of 2

order_id	amount	status
A101	19.95	paid
A102	8.50	shipped



Fail early. Fix once. Protect every downstream table.

SHIP DATA YOU CAN TRUST

RECAP

-  Define valid data.
-  Validate at the boundary.
-  Quarantine and explain failures.
-  Version intentional changes.
-  Monitor every run.

PRACTICAL CHECKLIST

- ☒ Contract stored with the code?
- ☒ Required fields tested?
- ☒ Types and ranges checked?
- ☒ Bad rows preserved?
- ☒ Owner and version named?
- ☒ Breaking changes reviewed?



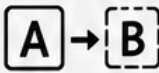
CHANGE GUIDE



Add optional column



Usually safe



Rename, drop, or change type



Review and version

FINAL TIP FROM ELLY



Make invalid data **loud** at the door—not **mysterious** downstream.

FHIR SEARCH PAGING

FOLLOW THE BUNDLE



Objective: retrieve every search result safely—without inventing page URLs.



FHIR search returns a **Bundle** of type **searchset**.

One Bundle may be only **one page**.



SEARCH → BUNDLE PAGE → NEXT LINK



The **server** owns the paging state. We **follow** its links.



Never guess the next page.

THE SAFE PAGING LOOP



1 SEND

GET /Patient?active=true&_count=50

2 READ

Bundle.entry

3 FIND

link.relation = "next"

4 GET

link.url exactly as supplied

5 REPEAT

Stop when there is no next link.



_count requests
a page size.
A server may return
fewer.



Process → Follow **NEXT** → Repeat

PAGING GUARDRAILS

PAGE 3



1 OPAQUE LINKS



Do not edit, decode, or rebuild the next URL.

2 CHECK THE BUNDLE



Expect `type = searchset`. Handle HTTP errors and `OperationOutcome`.

3 COUNT MATCHES



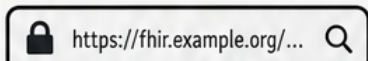
When present, `Bundle.total` is total matches—not this page.

4 PROCESS SAFELY



Deduplicate by `resourceType/id`. Make writes *idempotent*.

5 PROTECT DATA



Use *HTTPS*. Treat *search URLs* and *logs* as *sensitive*.



Paging is *transport* logic—*clinical validation* still applies.



WORKED EXAMPLE: 120 PATIENTS



1 PROBLEM

A client reads only page 1.
Result: **70 patients** are missed.



2 ACTION



1

PAGE 1: **50** → NEXT

2

PAGE 2: **50** → NEXT

3

PAGE 3: **20** → NO NEXT

3 CHECK



120 distinct Patient/id values processed.



No next link remains.



HTTP and OperationOutcome checks passed.



4 IMPROVEMENT



Make downstream upserts idempotent.



On retry, skip IDs already committed.



Follow the chain; verify the set.



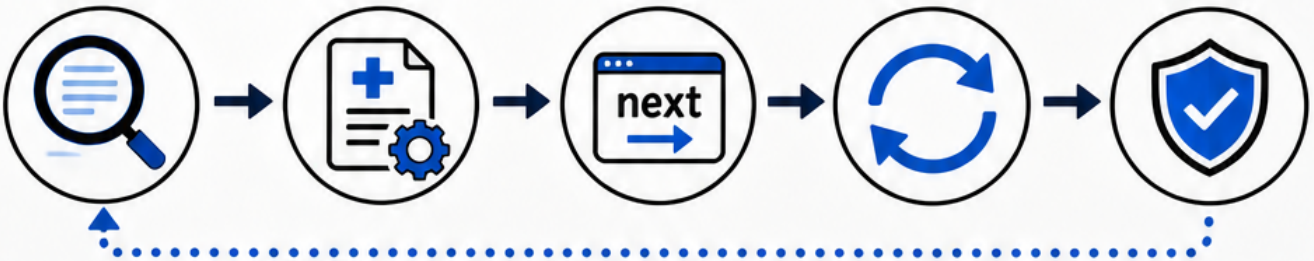
Follow the full chain, check the results, and keep **improving!**

RECAP: FOLLOW THE BUNDLE

- ☐ Send a supported FHIR search
- ☐ Confirm a searchset Bundle
- ☐ Process entry resources
- ☐ Follow the next URL unchanged
- ☐ Stop only when next is absent
- ☐ Deduplicate and use idempotent writes



SEARCH → PROCESS → NEXT → REPEAT → CHECK



FINAL TIP FROM ELLY

Let the server turn the pages. Your job is to process each page safely.



NEXT is data—not a URL you design.



POWER QUERY FOLDING

PUSH WORK TO THE SOURCE

Hi! I'm Elly.
Today we'll make
big refreshes lighter
by letting the source
do the heavy work.



OBJECTIVE

Keep filters and column selection foldable, then verify what runs remotely.



M STEPS



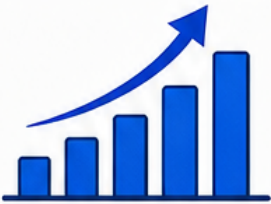
SOURCE QUERY



FEWER ROWS

WHAT IS QUERY FOLDING?

Power Query translates supported M steps into a query the source can run.



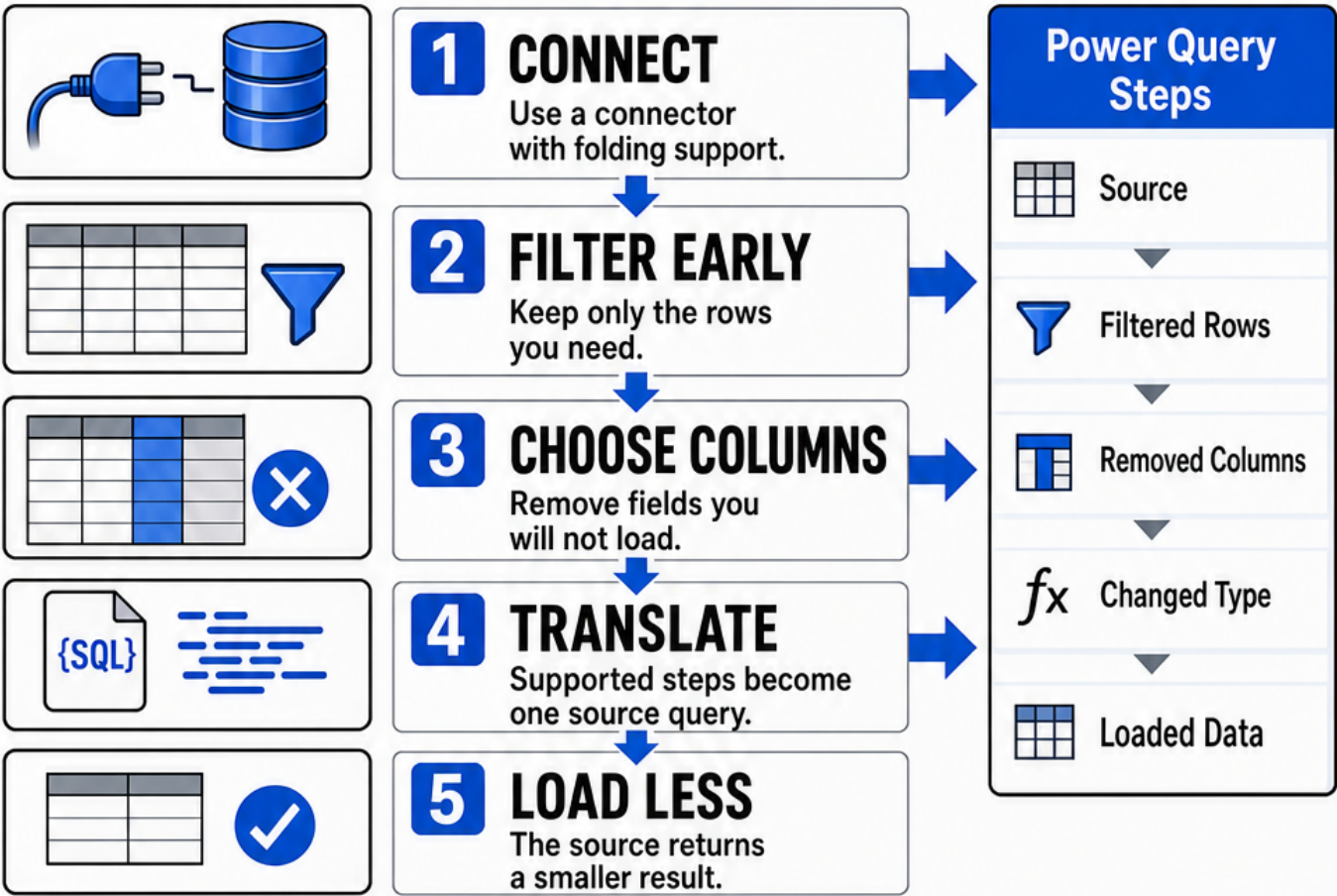
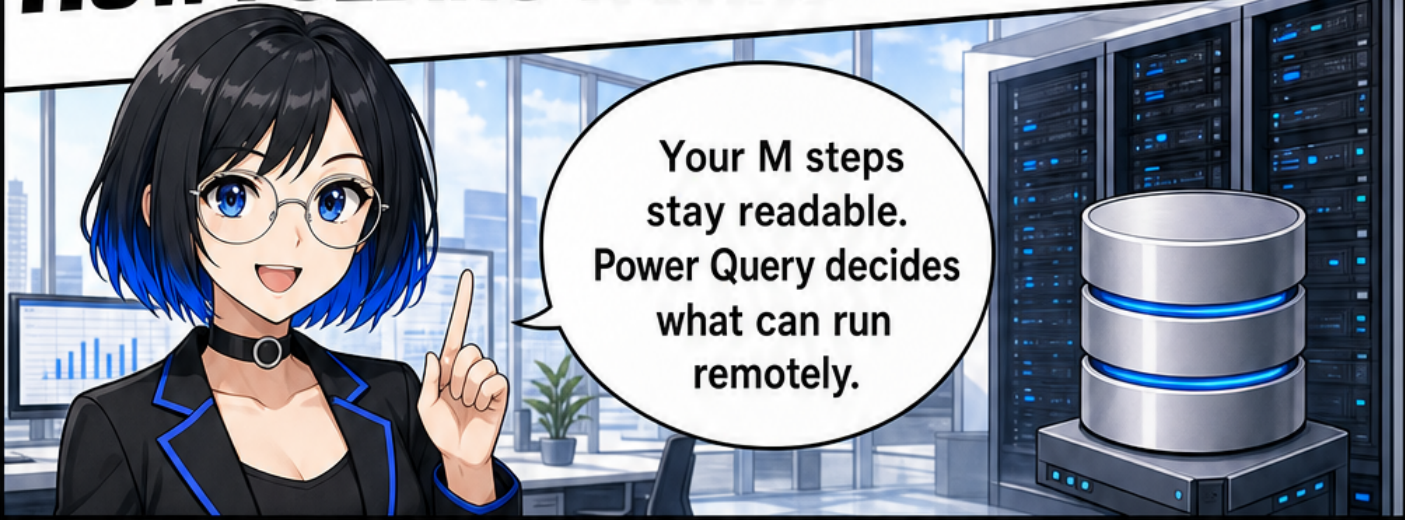
WHY IT HELPS

- Less data in transit
- Less local work
- Often faster refresh

Fold early. Move less data.

HOW FOLDING WORKS

PAGE 2



THE BIG IDEA



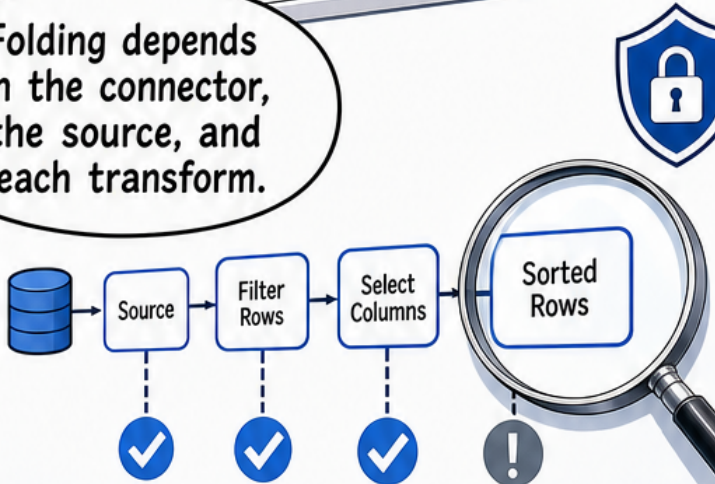
Push down the work. Pull back the result.

CHECK, DON'T GUESS

PAGE 3

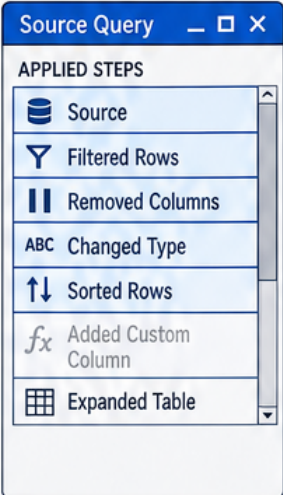


Folding depends on the connector, the source, and each transform.

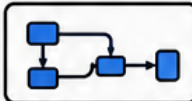


GOOD FIRST STEPS

- ✓ Filter rows
- ✓ Choose columns
- ✓ Keep simple steps together
- ✓ Delay unsupported transforms

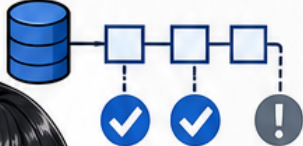


VERIFY THE WORK

-  Query plan — Power Query Online
-  Folding indicators — Online
-  View Native Query — when available
-  Query Diagnostics — Power BI

GUARDRAILS

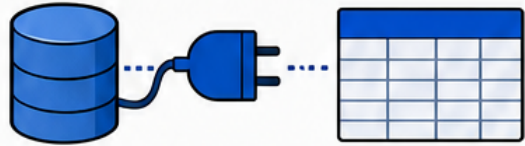


1 
A non-folding step can still leave earlier steps folded.

2  Privacy levels matter when combining sources.

3  Recheck after every significant edit.

Use the inspection tools your connector exposes.



Connector + source + transform = folding outcome

WORKED EXAMPLE: 50M SALES ROWS

1 PROBLEM

A custom text label runs before the date filter. Power Query may pull far too much data.

Sales (50,000,000 rows)

Date	StoreID	ProductID	Net	Quantity	CustomerID	...
2026-07-20	101	P1001	23.45	2	C0001	...
2026-07-20	102	P1002	12.90	1	C0002	...
2026-07-20	103	P1003	19.99	1	C0003	...
...	...	...	...	...	...	...
2026-09-05	250	P9821	45.60	3	C9812	...
2026-09-05	251	P9822	15.75	1	C9813	...
2026-09-05	252	P9823	33.10	2	C9814	...
...	...	...	...	...	...	...



2 ACTION

- 1 Filter Date >= 2026-08-01
- 2 Keep Date, StoreID, Net
- 3 Add the display label later



Put selective, supported steps first.



3 CHECK

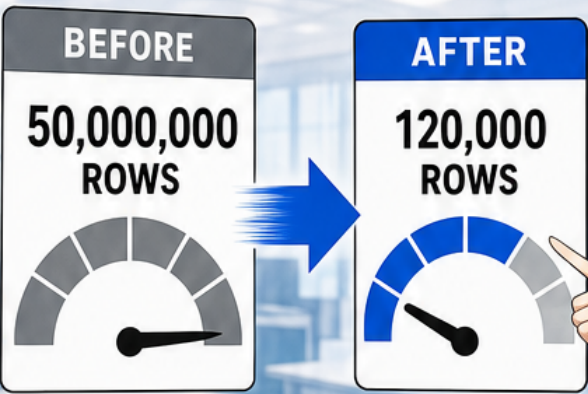
Inspect the last folded step.

```
Source-Query (View Native Query)
SELECT Date, StoreID, Net
FROM Sales
WHERE Date >= '2026-08-01'
— illustrative SQL —
```



4 IMPROVEMENT

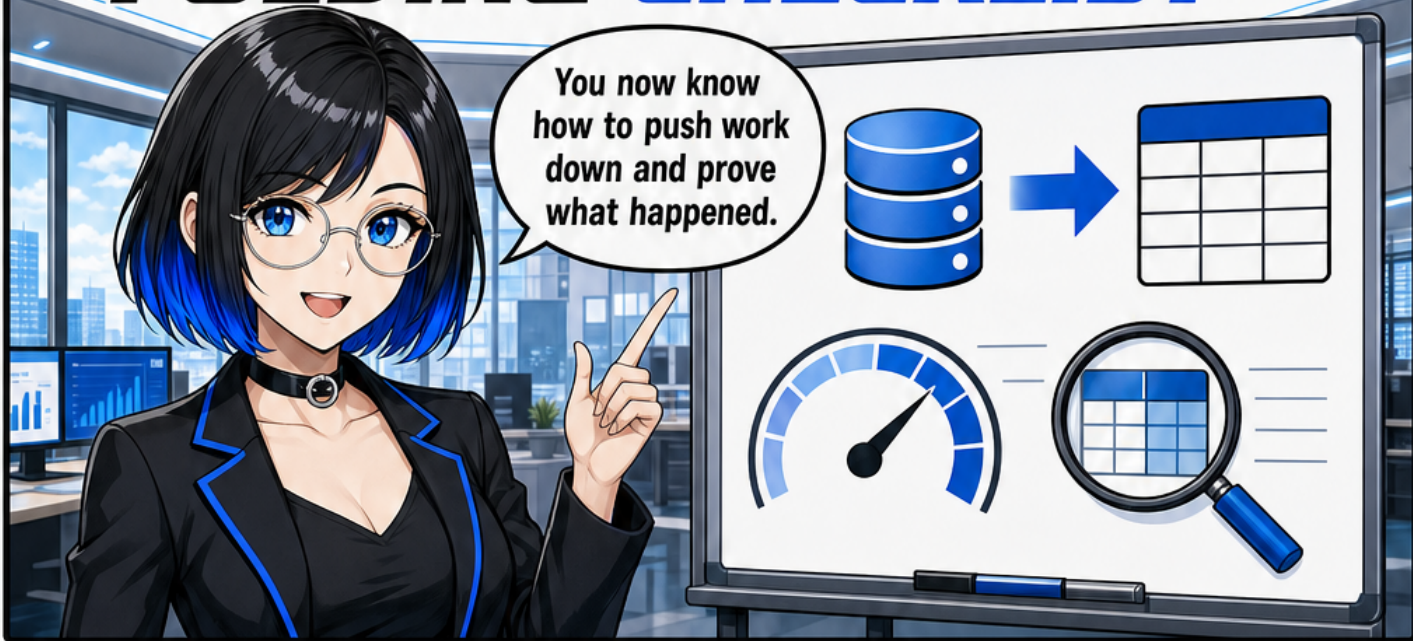
- Illustrative transfer: 50,000,000 → 120,000 rows
- ! Measure your own refresh. Results vary.



Problem → Action → Check → Improve

FOLDING CHECKLIST

PAGE 5



BEFORE YOU LOAD

- 1 Start with a foldable source
- 2 Filter rows early
- 3 Choose only needed columns
- 4 Delay unsupported transforms
- 5 Inspect the source query
- 6 Measure refresh time and rows

FINAL CHECK

- | | |
|--|--|
| ☒ 1. Full, partial, or none? | ☒ 3. Privacy levels correct? |
| ☒ 2. Earlier steps still fold? | ☒ 4. Diagnostics reviewed? |

FINAL TIP FROM ELLY

Fold first. Fancy later.

PUSH DOWN • PULL LESS • REFRESH SMARTER

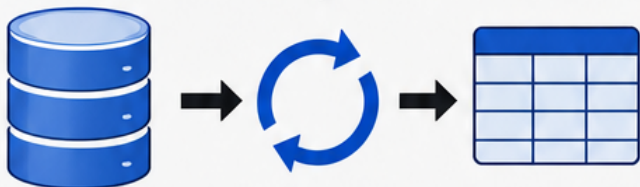
POWER BI INCREMENTAL REFRESH

PROCESS ONLY WHAT CHANGED



FULL RELOAD

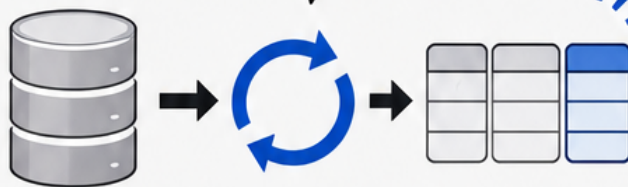
All history, every time



Split a large table
into time-based **partitions**.

INCREMENTAL

Recent partitions only



Refresh the **recent slice**;
keep older history.

OBJECTIVE



Refresh **recent** data
without reloading the **full history**.



INCREMENTAL REFRESH: 5 STEPS

PAGE 2



Power Query

Queries > Events

EventID	EventDate	EventType	Value
1001	2024-05-01 00:00:00	Click	15
1002	2024-05-01 00:05:00	View	23
1003	2024-05-01 00:10:00	Purchase	99
...	...	...	...

Steps

Source

Changed Type

Filtered Rows

2024-01-01

2024-01-08

2024-01-15

...

2029-01-01

Partition 1
2024-01-01
to 2024-01-08

Partition 2
2024-01-08
to 2024-01-15

...

Partition N
2028-12-25
to 2029-01-01

1
CREATE
PARAMETERS


 RangeStart



 RangeEnd



Date/Time — names are case-sensitive

2
FILTER THE
DATE COLUMN


[EventDate] >= RangeStart

[EventDate] < RangeEnd

3
KEEP QUERY
FOLDING




Push the filter to the source.

4
SET THE
POLICY


 Store 5 years

 Refresh 7 days

5
PUBLISH AND
REFRESH




The first refresh creates the partitions.

 Lower boundary (inclusive)
>= RangeStart

 Upper boundary (exclusive)
< RangeEnd

Included

RangeStart (inclusive)

RangeEnd (exclusive)

Excluded



✓ Use one inclusive bound and one exclusive bound.

✓ No duplicate boundary rows.



GUARDRAILS FOR A SAFE REFRESH

PAGE 3



Fast is useful only when the result is correct.

- ✓ MATCH TYPES
- ✓ KEEP THE BOUNDS
- ✓ VERIFY FOLDING
- ✓ SIZE THE WINDOWS
- ✓ EXPECT THE FIRST LOAD
- ✓ CHECK UTC



1 MATCH TYPES



Date column and both parameters use **Date/Time**.

2 KEEP THE BOUNDS



Use **>=** for **RangeStart** and **<** for **RangeEnd**.

3 VERIFY FOLDING



The source should receive the date filter.

4 SIZE THE WINDOWS



Store enough history. Refresh only what can change.

5 EXPECT THE FIRST LOAD



The initial refresh builds the historical partitions.

6 CHECK UTC



Service refreshes run in **UTC**.

OPTIONAL: DETECT DATA CHANGES



Use a separate audit column, such as **ModifiedAt**.



Elly says: Fast is useful only when the result is correct.



WORKED EXAMPLE: FACTVISITS

PAGE 4



FACTVISITS (FACT TABLE)

VisitID	UserID	VisitDate	StoreID	Amount	...
1	8871	2025-04-20	102	42.50	...
2	5321	2025-04-20	103	18.90	...
3	2290	2025-04-20	101	55.10	...
...					

DATE FILTER



All Dates
2018-01-01
to
2025-04-20

REFRESH TIMER



1 PROBLEM



120 million rows



Nightly full
refresh: 95 min

BEFORE

2022	2023	2024	2025

(All data refreshed every night)

2 ACTION



Filter **VisitDate**



Store **3** years



Refresh **3** days

AFTER

HISTORICAL (Read-Only)		HISTORICAL (Read-Only)		REFRESH WINDOW (3 DAYS)
2022	2023	2024	2025	

(Only recent 3 days refreshed nightly)

3 CHECK



Confirm **query folding**



Inspect refreshed **partitions**



Reconcile **row counts**

HISTORICAL (Read-Only)	HISTORICAL (Read-Only)	REFRESH WINDOW (3 DAYS) ✓
2022	2023	2024

4 IMPROVEMENT



Detect data changes
with **ModifiedAt**



Only **affected**
days refresh

HISTORICAL (Read-Only)	HISTORICAL (Read-Only)	D-3	D-2	D-1	TODAY ✓
2022	2023	2024			2025

ILLUSTRATIVE RESULT



Measure your own model.
The result must stay **correct**.

READY TO REFRESH LESS?



Partition	Status
2025-04-01	✓ Refreshed
2025-04-02	✓ Refreshed
2025-04-03	✓ Refreshed
2025-04-04	Skipped
2025-04-05	✓ Refreshed
...	

PRACTICAL CHECKLIST

- ✓ RangeStart and RangeEnd use Date/Time
- ✓ [Date] >= RangeStart
- ✓ [Date] < RangeEnd
- ✓ Query folding verified
- ✓ Store and refresh windows chosen
- ✓ Initial refresh completed
- ✓ Second refresh checked

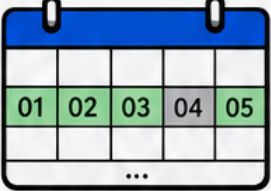
RECAP

FILTER



[Date] >= RangeStart
[Date] < RangeEnd

PARTITION



Split by date.
Only what changed
gets refreshed.

VERIFY



Check results.
Confirm what refreshed,
what was skipped,
and that it's correct.

FINAL TIP FROM ELLY



The second refresh **proves** the design.

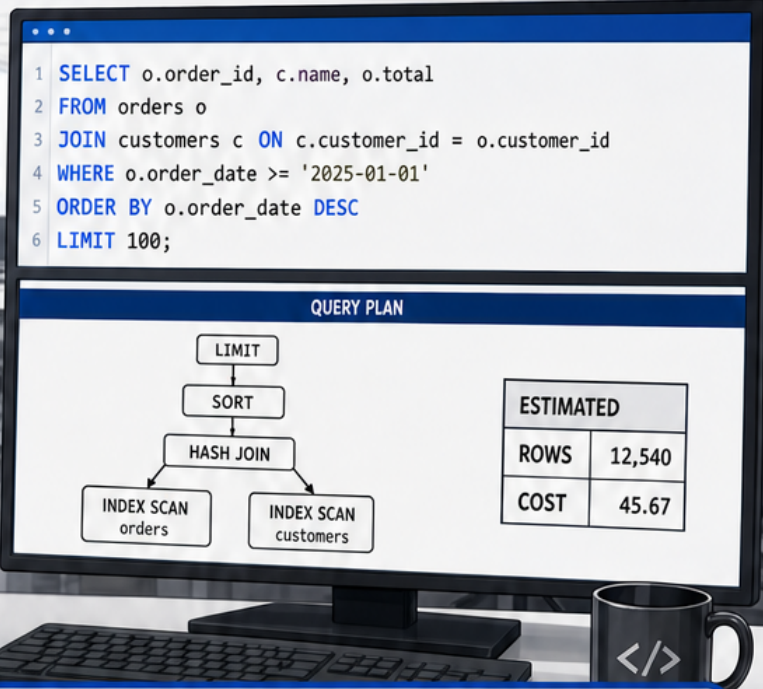


Check what was **skipped** – not only what finished.

**REFRESH LESS.
VERIFY MORE.**



SQL EXPLAIN: READ THE QUERY PLAN

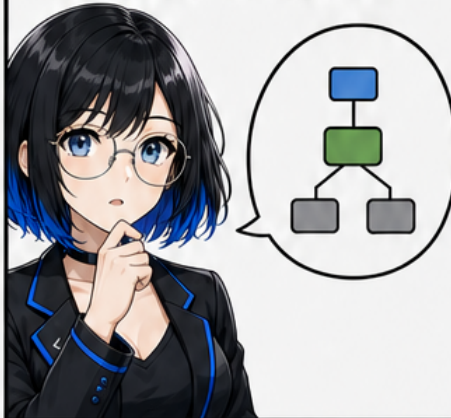


Objective: See how the database plans a query before tuning it.

YOUR SQL SAYS
WHAT YOU WANT.



THE QUERY PLAN SHOWS
HOW THE DATABASE
INTENDS TO GET IT.



DON'T GUESS
ASK THE PLANNER
BEFORE YOU TUNE.



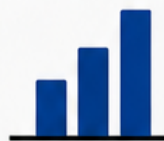
**LOOK
FOR**



Scans



Joins & sorts



Estimated rows



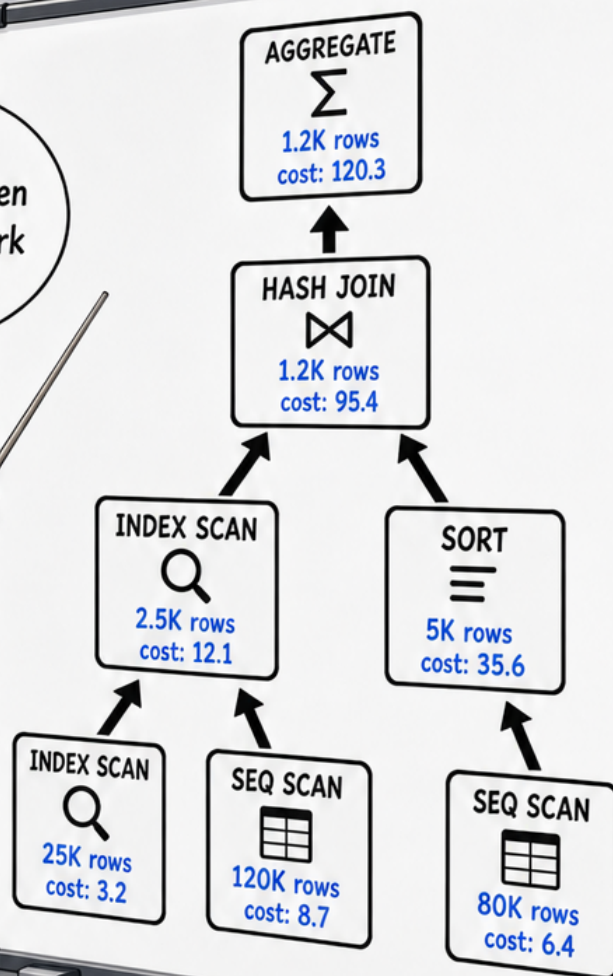
Estimated cost

PLAN FIRST. CHANGE ONE THING. MEASURE AGAIN.



READ THE PLAN AS A TREE

Start at the **leaves**, then follow the work **upward**.



1

READ THE DATA



Leaf nodes scan tables or indexes.

2

FOLLOW THE FLOW



Parent nodes join, sort, filter, or aggregate rows.

3

CHECK EACH NODE



- Operation
- Estimated rows
- Estimated cost

4

FIND THE FIRST SURPRISE



- A huge row jump
- Repeated loops
- An unexpected scan or sort



**COST IS A
RELATIVE ESTIMATE —
NOT MILLISECONDS.**



**READ BOTTOM-UP.
QUESTION THE
FIRST SURPRISE.**



SMART READERS



SMARTER DECISIONS



ESTIMATED vs ACTUAL



ESTIMATED PLAN



- ✓ Predicts operations, rows, and relative cost.
- ✓ Does not run the query.

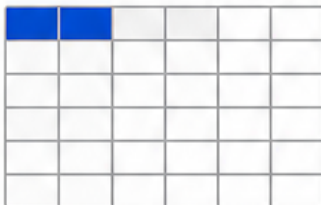
ACTUAL PLAN / ANALYZE



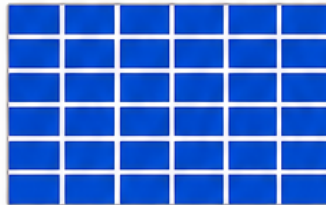
- ✓ Runs the query and records actual rows and timing.

THE GAP IS A CLUE

Estimated: 100 rows



Actual: 50,000 rows



Large gaps can point to *stale statistics*, *skewed data*, or *parameter effects*.

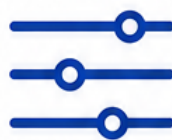
4 GUARDRAILS



1. Start with a safe read-only query.



2. Remember: **ANALYZE** executes the statement.



3. Test representative parameters.



4. Change one thing, then measure again.



NEVER RUN AN ACTUAL PLAN ON A RISKY WRITE BY ACCIDENT.

WORKED EXAMPLE: STOP THE FULL SCAN

1. PROBLEM



5 million orders. One customer lookup takes 1.8 s.

```
SELECT * FROM orders  
WHERE customer_id = 42;
```

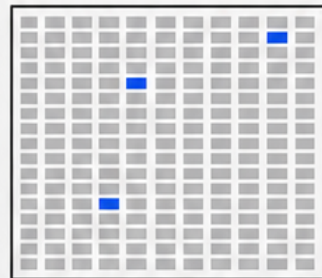
ACTUAL PLAN



Seq Scan

Rows checked: 5,000,000

Rows returned: 52



2. ACTION



```
CREATE INDEX idx_orders_customer_id  
ON orders(customer_id);
```



3. CHECK



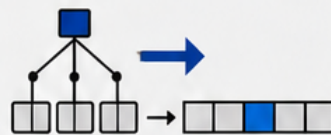
Run the same
safe query and
compare the
actual plan.

- ☒ Same 52 rows?
- ☒ Same values?
- ☒ Different access path?

4. IMPROVEMENT



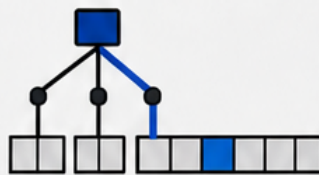
Index Scan



Rows returned: 52

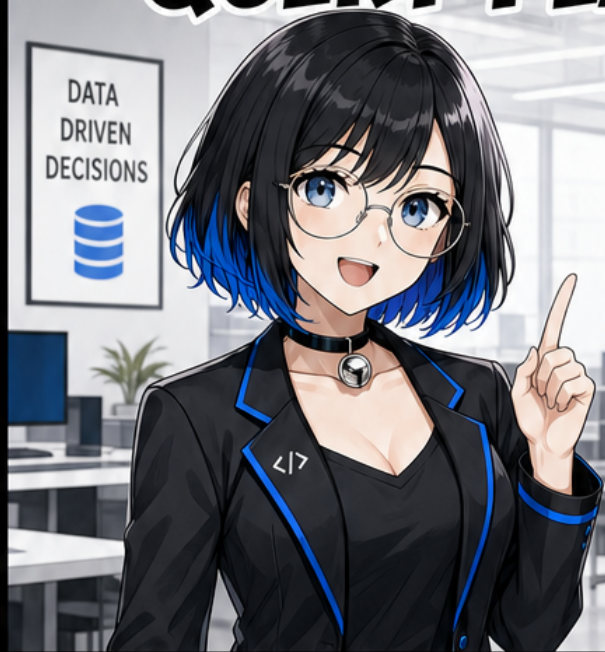
Time: 12 ms

RESULT: SAME ANSWER. FAR LESS WORK.



EXAMPLE NUMBERS — YOUR DATA WILL DIFFER.

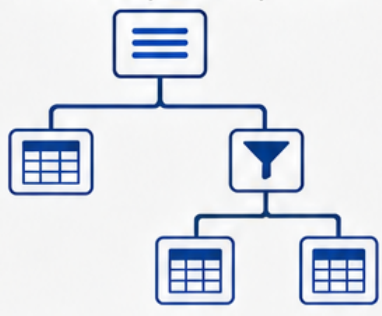
QUERY-PLAN CHECKLIST



- ☒ 1. Capture the query and representative parameters.
- ☒ 2. Get the estimated plan.
- ☒ 3. Read the tree from leaves upward.
- ☒ 4. Check scans, joins, sorts, row gaps, and repeated work.
- ☒ 5. Use an actual plan only when execution is safe.
- ☒ 6. Change one thing; rerun; verify the same result.

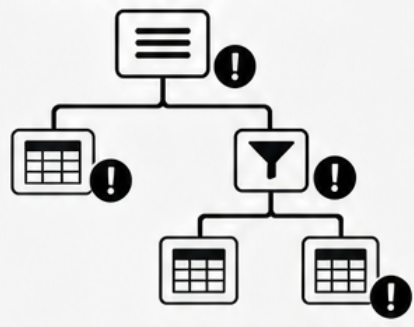
ESTIMATED

What the planner predicts



ACTUAL

What execution reveals



ELLY'S FINAL TIP

TUNE THE EVIDENCE, NOT THE HUNCH.

Small measured changes beat big guesses.



YOU CAN READ THE PLAN!

GOLDEN HOUR PHONE PHOTOS

LIGHT BEFORE FILTERS



OBJECTIVE:

Make one warm, story-rich photo using only light, position and timing.



Golden hour is the soft, low-angle light soon after sunrise or before sunset.

Your best filter is a **sun** that's nearly off duty.

SOON AFTER SUNRISE



BEFORE SUNSET



CHASE THE LIGHT

PAGE 2

1. ARRIVE EARLY.

Pick one simple subject and a clean background.



2. MOVE AROUND IT.

Front light = clear.



Side light = texture.



Backlight = glow.



3. TAP THE SUBJECT.

Lower exposure a little if highlights lose detail.



4. TAKE THREE VERSIONS.

Wide. Medium. Close.



5. STAY AFTER SUNSET.

The afterglow can be the winner.

Where it's safe, zoom with your feet.



THE FIVE-SECOND CHECK

Clean lens,
clean edges,
clean **story**.

1 LIGHT — Soft and directional?



2 FRAME — Horizon straight? Edges clean?



3 FOCUS — Main detail sharp?



4 EXPOSURE — Highlights still have detail?



5 STORY — One clear subject?



GUARDRAILS



Wipe
the lens.



Skip
digital zoom.



Look before
you step.

WORKED EXAMPLE: *FLAT TO CINEMATIC*

PAGE 4

PROBLEM

Friend in open shade: grey face, busy bins behind.



ACTION

Turn 90° for side/backlight.

Step left; hide bins behind a tree.

Tap the face; lower exposure until sky detail returns.

Shoot *wide*, *medium*, *close*.



CHECK

Sharp eyes.
Bright rim.
Straight horizon.
Detailed sky.



IMPROVE

Wait two minutes and catch the *warmer* *afterglow*.





Now go make the sun look expensive.

THE GOLDEN HOUR POCKET LIST ☆

- ☐ Arrive early.
- ☐ Wipe the lens.
- ☐ Find one clear subject.
- ☐ Move for front, side or backlight.
- ☐ Tap focus and protect highlights.
- ☐ Shoot wide, medium, close.
- ☐ Stay for the afterglow.



FINAL TIP

When the light changes, retake the shot.
☆ Tiny waits make huge pictures.



WATERCOLOUR SKIES

LET THE WATER WORK

PAGE 1

WEEKEND
WITH ELLY

TODAY:

paint a soft
blue sky with
lifted clouds.



The secret?
A wet sheet and
the **courage** to
leave it alone.

WET PAPER, SOFT EDGES

ONE LITTLE SKY

Diluted paint spreads
into damp paper.

We'll wash, lift,
then let it dry.

Clouds are
terrible at staying
inside the lines.
Good.

WET, WASH, LIFT

PAGE 2

1 WET

Brush clean water across the sky area. Aim for an **even** sheen, not puddles.



2 WASH

Sweep diluted blue into the damp paper. Use a **light** touch.



3 LIFT

While damp, lift clouds with a **clean, blotted** brush. Rinse and blot between lifts.



Soft edges need **water**.
They don't need **endless fiddling**. ☆



SMALL KIT, BIG SKY

PAGE 3



TEST FIRST



LET IT DRY



FROM STIFF TO SOFT

PAGE 4

PROBLEM

Hard edges?
The paper was
too **dry** for this
soft-sky effect.

My clouds
look like
blue potatoes.

ACTION

Try a
fresh scrap.
Wet it evenly,
then add **blue**.

CHECK

Does the
colour spread
softly?

IMPROVE

Lift two
clouds while
damp. **Stop**
before you
scrub.

Oh!
Less wrestling.
More sky.

A LITTLE SKY, A LOT LESS FUSS

One wash.
Two clouds.
A *very* good
Sunday.



YOUR SKY CHECKLIST

- ☒ Test your blue.
- ☒ Wet evenly.
- ☒ Sweep in colour.
- ☒ Lift, rinse, blot.
- ☒ Let it dry.



JUDGE IT DRY

Watercolour dries lighter.
Check before adding more.



ELLY'S FINAL TIP

Sometimes
the best
brushstroke is
putting the
brush down.

