



Microsoft Fabric

This presentation is the property of Microsoft and is intended for informational and educational purposes only. You may use, copy, and distribute this presentation for your personal, non-commercial purposes. You may not modify, alter, or create derivative works from this presentation without the prior written consent of Microsoft. You may not use this presentation to misrepresent, defame, or disparage Microsoft or its products, services, or affiliates. You may not use this presentation to endorse or promote any other products, services, or organizations without the prior written consent of Microsoft.

By using this presentation, you agree to abide by these terms. If you do not agree, you must not use this presentation. Microsoft reserves the right to change these terms and conditions at any time without notice. Microsoft disclaims any and all warranties, express or implied, relating to this presentation, including but not limited to the accuracy, completeness, timeliness, or suitability of the information contained herein. Microsoft is not liable for any damages, losses, or liabilities arising from your use of or reliance on this presentation.

Please review the terms of use posted in the content library.

#FABCONSQLCON2026

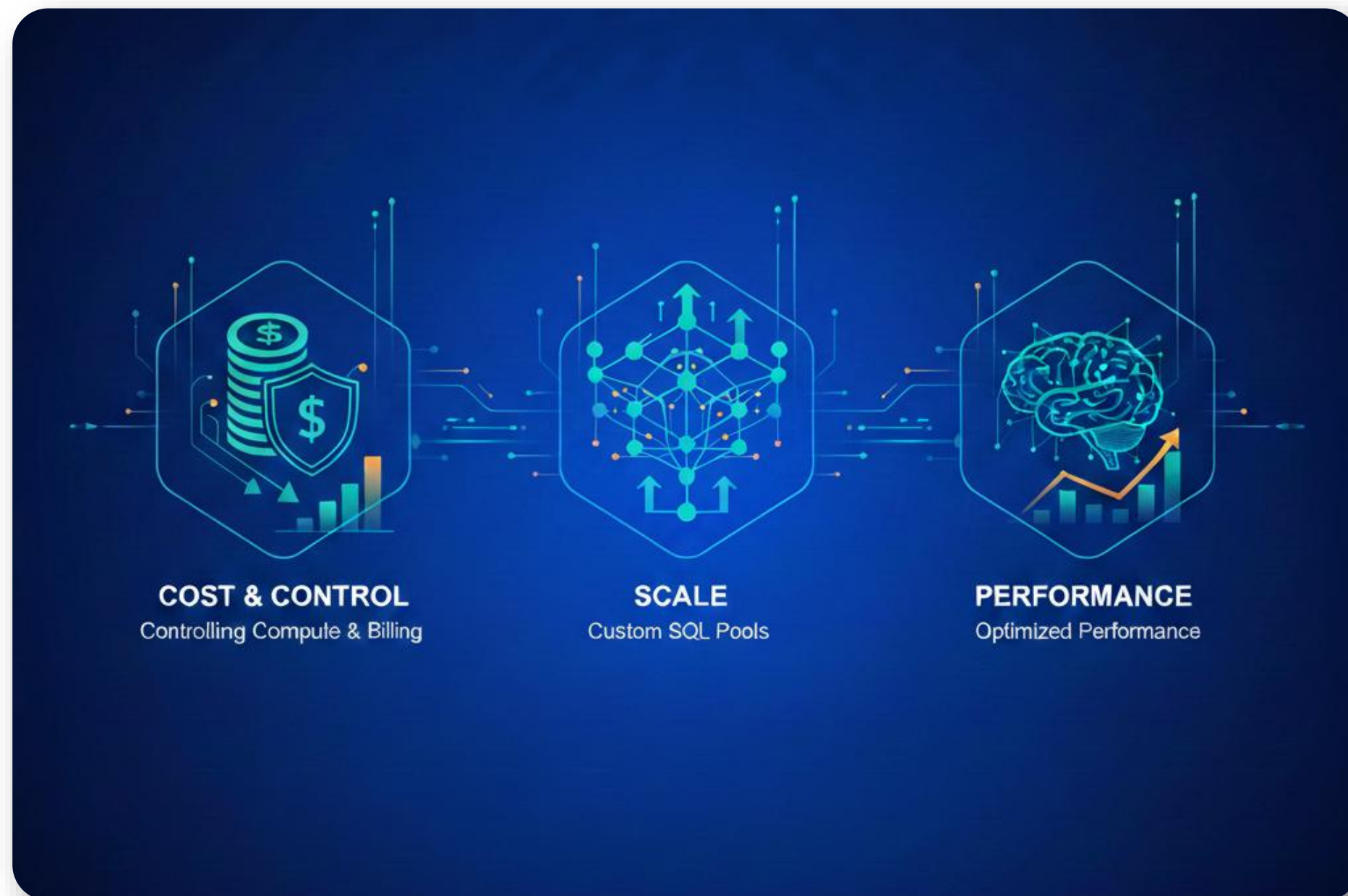
FABCON

Microsoft Fabric
COMMUNITY CONFERENCE

SQLCON

Microsoft SQL
COMMUNITY CONFERENCE

ATLANTA MARCH 16 - 20, 2026



Cost, Scale and Control

Running mission-critical data warehouses on shared Fabric capacity

Meet the Speakers



Bret Myers

Principal Product Manager, Data Warehouse



<https://www.linkedin.com/in/bretamyers/>



Sowmya Sivaraman

Senior Product Manager, Data Warehouse



<https://www.linkedin.com/in/sowmyasivaraman/>

Agenda: From Chaos to Control

- Problem with Shared Capacity
- Scale without knobs
- Control through Isolation
- Cost through visibility

Think of Fabric like a carnival



Fabric capacity = Bag of tokens

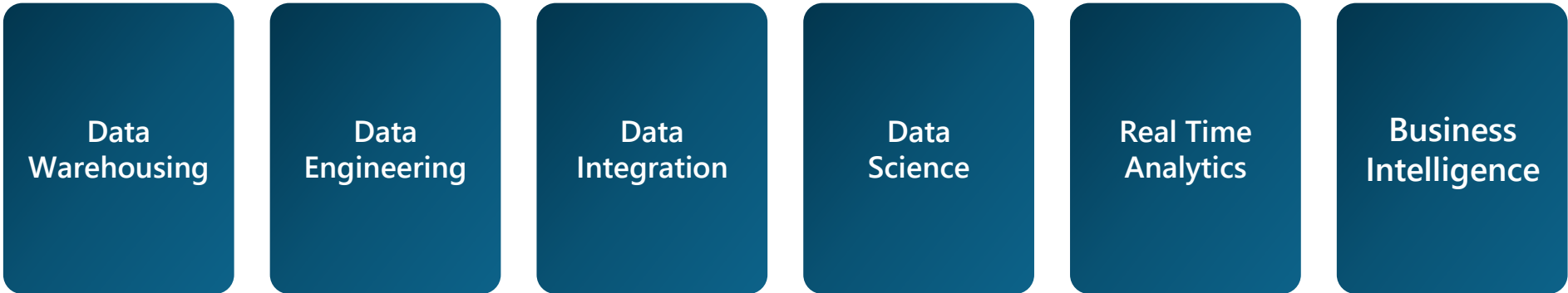
Different workloads = Rides in a carnival

Use-it-or-lose-it capacity = Tokens expire daily

Current State: Shared Capacity

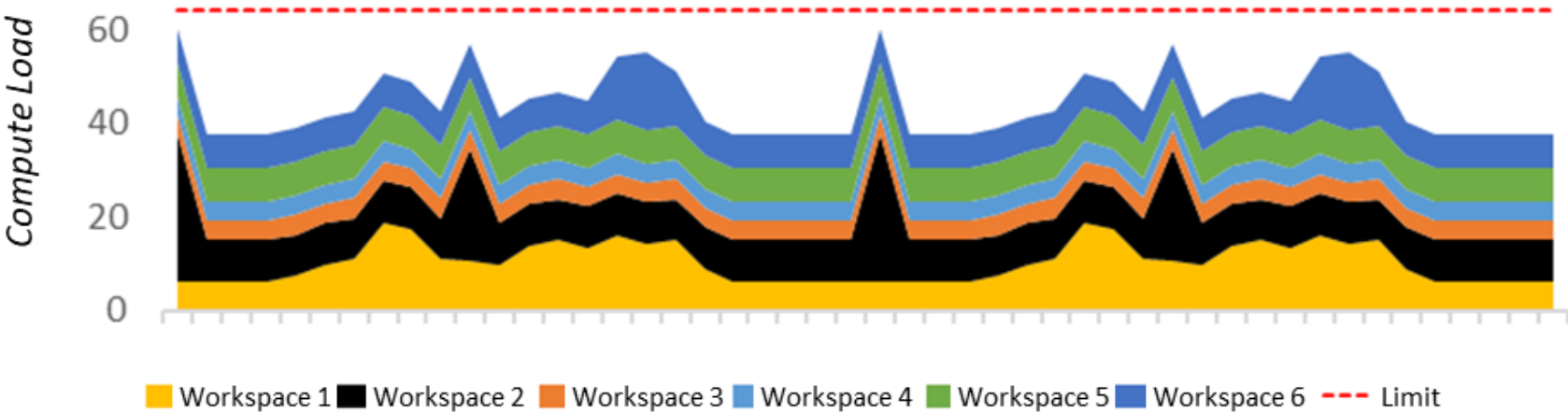
One pool of compute Shared across workloads

There is no need to allocate compute for each workload separately.



Easy to get started

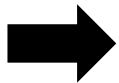
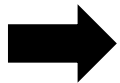
There is no need to allocate capacity per workload. Bursting and Smoothing are automatically applied



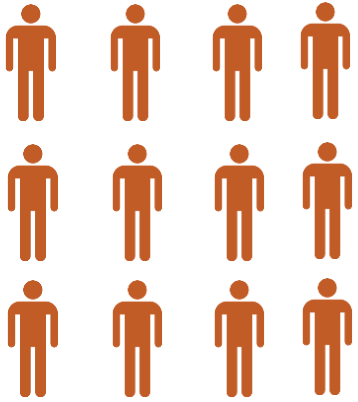
Shared across projects and teams

For each project, many developers will share a workspace where collaborative development and consumption at scale is managed.

Developers/Creators



Consumers



Everyone loves shared capacity... until Monday Morning



In a Carnival - Everyone rushes to the rollercoaster at once. Tokens disappear and other rides stall.

In Fabric - On Monday morning ETL, BI, and ad-hoc all collide

Suddenly cost is unpredictable, performance is slow, and capacity admins have no levers.

Your Capacity Isn't the Problem

Ingestion

A workload is an identifiable group of queries/statements—e.g., ETL jobs, Power BI or Tableau report, ad-hoc queries.

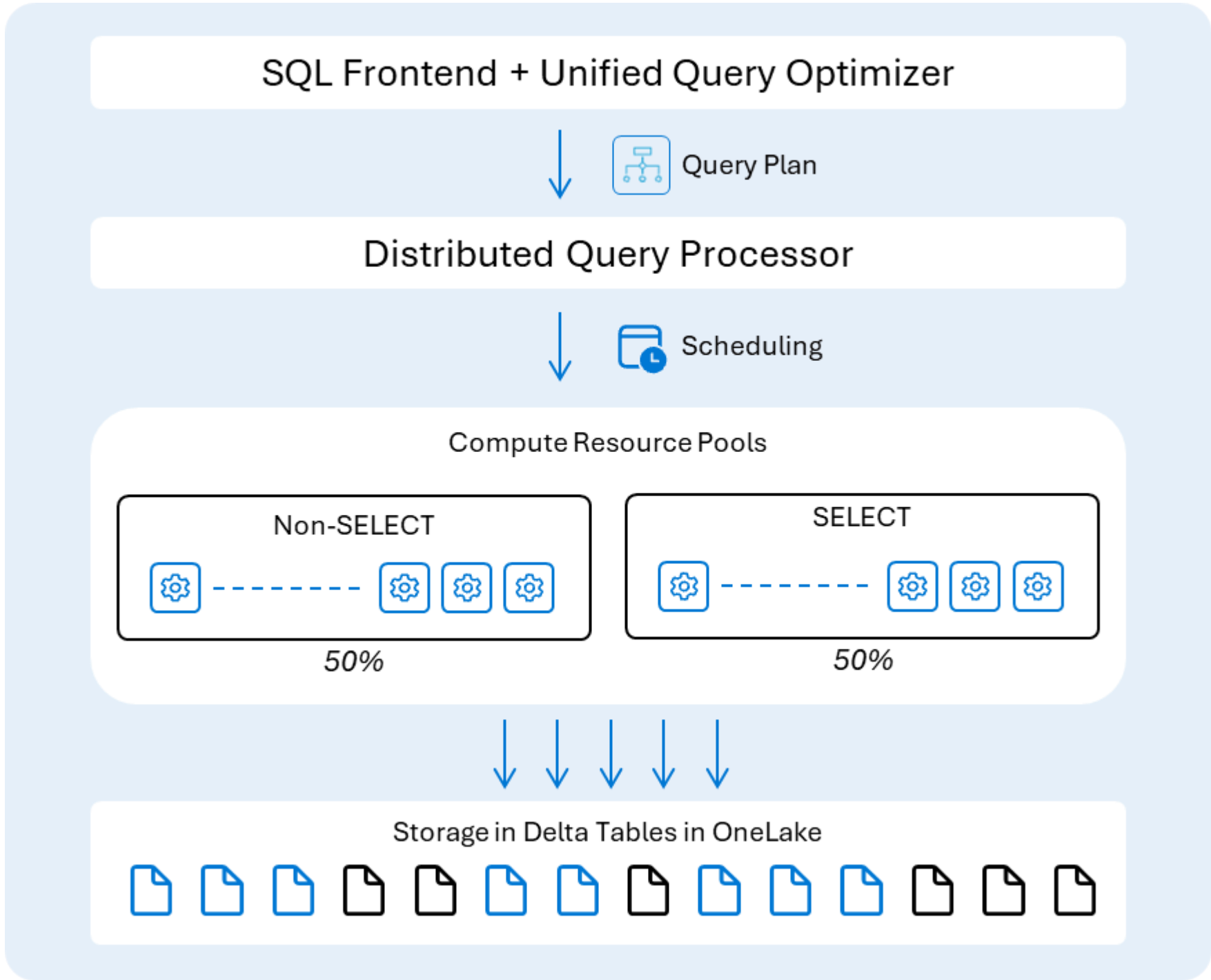
Reporting bursts

Three workload types show up in every enterprise warehouse

Ad-hoc Exploration

Real issue is a workload collision

Isolation Is the Only Real Form of Predictability



Physical isolation exists between Ingestion (ETL workloads) and Query (analytics/reporting workloads)

Maximum burstable capacity is split evenly between these two-resource pools based on statement type (50/50).

Burstable Compute is available when needed, up to a SKU specified maximum

Built-in Workload Isolation



Great carnivals don't just add more rides.
They manage the lanes better.

Fabric DW does the same thing with built-in
workload isolation

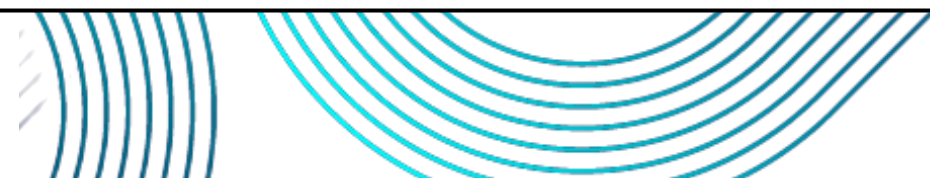
Ingestion doesn't compete with reporting.
Reporting doesn't get starved by background
work

Built-in pools are not always the perfect fit

Noisy neighbors
from other queries
within a pool

Limited ability to
prioritize workloads

Fixed 50/50 split
doesn't fit all
workloads



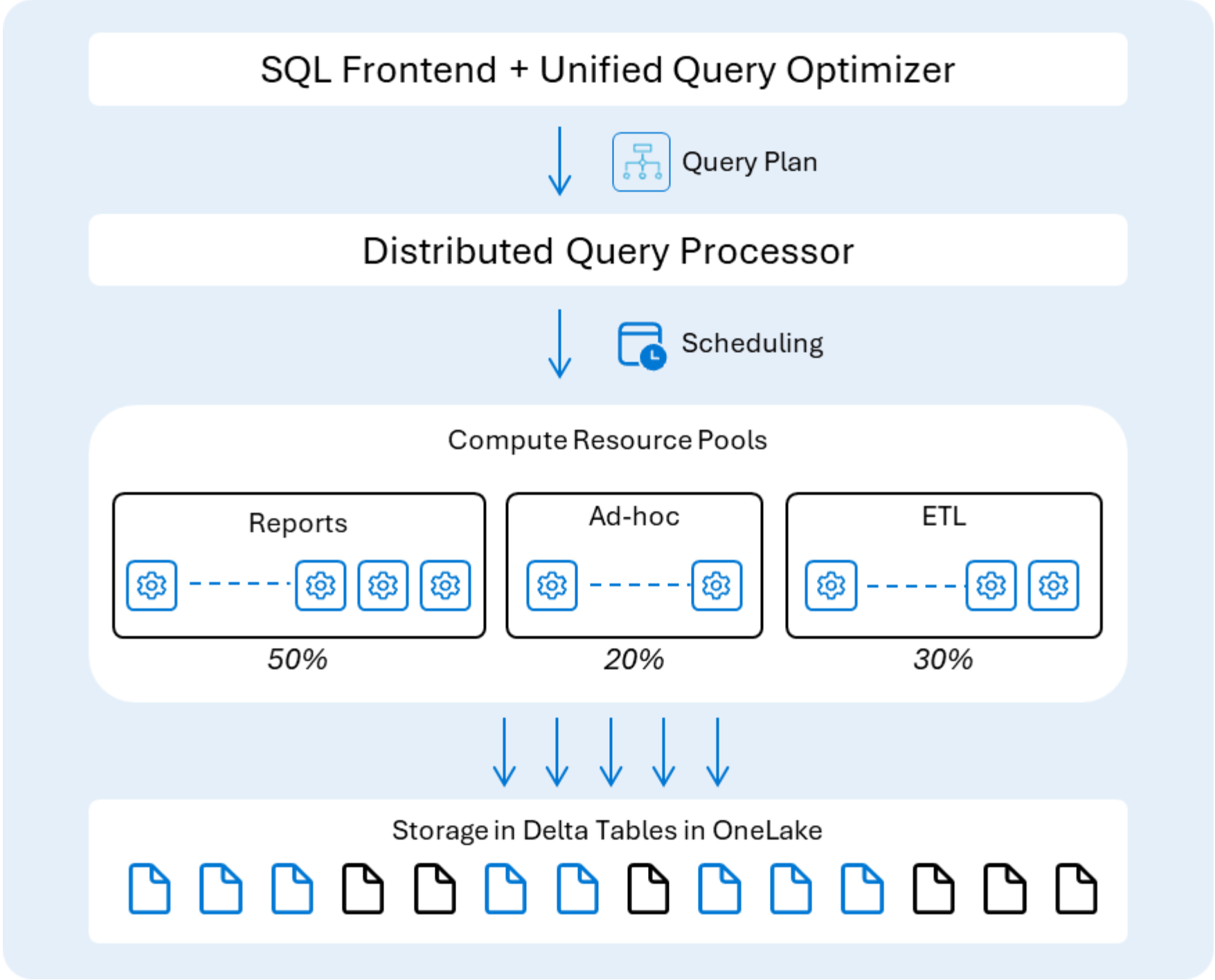
Custom SQL Pools

Introducing Custom SQL Pools

Custom SQL Pools within a warehouse gives you the ability to easily define and customize pools of resources

- Allocate resources where they are needed the most by having customizable pool configurations
- Prioritize queries by routing queries to specific pools by application
- Physical resource isolation allows for concurrent queries not to interfere with each other
- Simplify the architecture by not needing to split the workload between workspaces

Custom SQL Pools

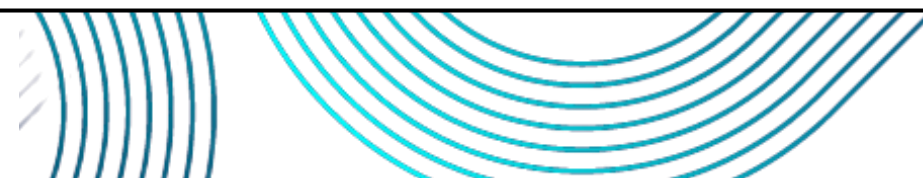


Maximum resource allocation percentage is customizable for each pool

A query is routed to a specific pool based on its classification and gets executed on the pool without impacting queries running on other pools

Example where 100 nodes are available from the capacity SKU size

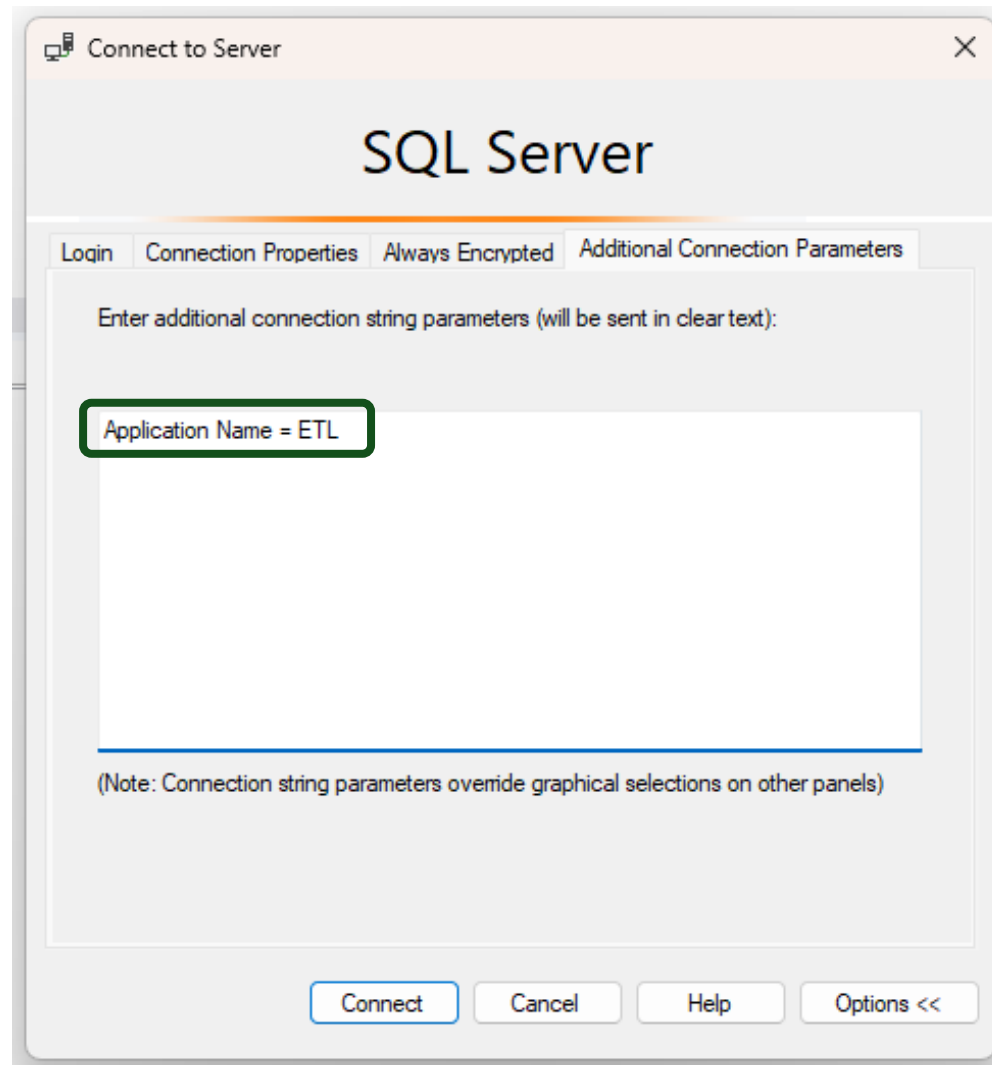
Pool Name	Max Resource Pct	Nodes Available
Reports	50%	50
Ad-hoc	20%	20
ETL	30%	30



Demo

Classifier Types

SSMS 21

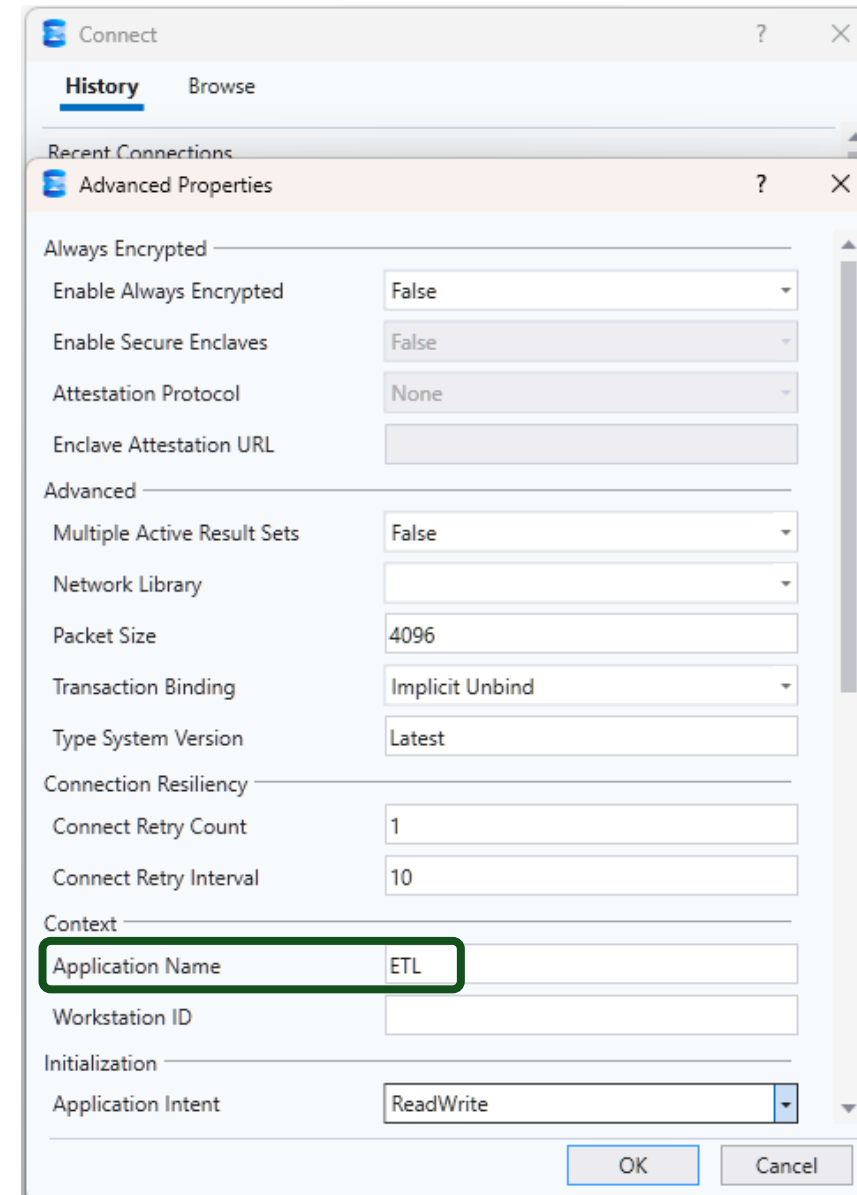


The SSMS 21 'Connect to Server' dialog box is shown. It has tabs for 'Login', 'Connection Properties', 'Always Encrypted', and 'Additional Connection Parameters'. The 'Additional Connection Parameters' tab is selected, showing a text area with 'Application Name = ETL' entered. Below the text area is a note: '(Note: Connection string parameters override graphical selections on other panels)'. At the bottom are buttons for 'Connect', 'Cancel', 'Help', and 'Options <<'.

jdbc

```
connection_string = (  
    f"jdbc:sqlserver://{sql_analytics_endpoint}:1433;"  
    f"database={warehouse_name};"  
    f"authentication=ActiveDirectoryInteractive;"  
    f"applicationName=ETL"  
)
```

SSMS 22



The SSMS 22 'Connect' dialog box is shown. It has tabs for 'History' and 'Browse'. The 'Advanced Properties' sub-dialog is open, showing various settings. The 'Context' section has 'Application Name' set to 'ETL'. Other settings include 'Always Encrypted' (False), 'Enable Always Encrypted' (False), 'Enable Secure Enclaves' (False), 'Attestation Protocol' (None), 'Enclave Attestation URL' (empty), 'Multiple Active Result Sets' (False), 'Network Library' (empty), 'Packet Size' (4096), 'Transaction Binding' (Implicit Unbind), 'Type System Version' (Latest), 'Connect Retry Count' (1), 'Connect Retry Interval' (10), 'Workstation ID' (empty), 'Initialization' (empty), and 'Application Intent' (ReadWrite). At the bottom are 'OK' and 'Cancel' buttons.

pyodbc

```
connection_string = (  
    f"DRIVER={{ODBC Driver 18 for SQL Server}};"  
    f"SERVER={sql_analytics_endpoint};"  
    f"Database={warehouse_name};"  
    f"App=ETL"  
)
```

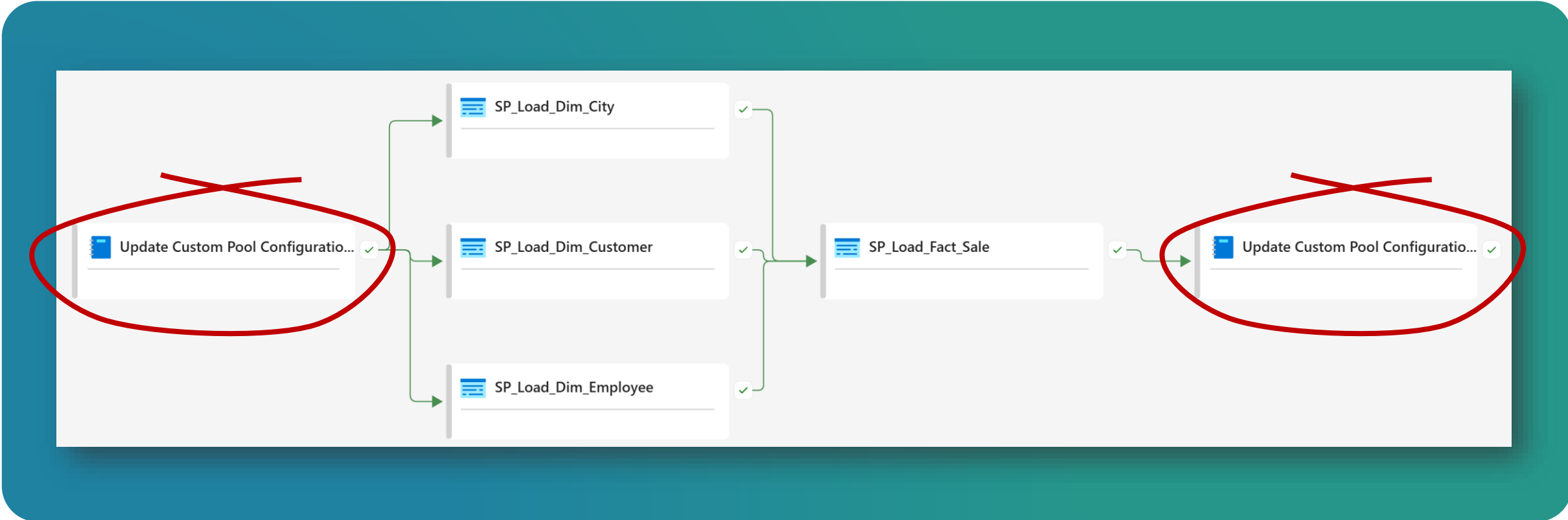
Application Name

A user defined list of static values

Application Name Regex

A single regular expression for greater flexibility on matching

Configure Programmatically



PATCH Request Example

```
{
  "customSQLPoolsEnabled": true,
  "customSQLPools": [
    {
      "name": "ETL",
      "isDefault": false,
      "maxResourcePercentage": 80,
      "optimizeForReads": false,
      "classifier": {
        "type": "Application Name",
        "value": [
          "ETL",
          "Load",
          "Pipeline"
        ]
      }
    },
    {
      "name": "Adhoc",
      "isDefault": true,
      "optimizeForReads": false,
      "maxResourcePercentage": 20
    }
  ]
}
```


Monitoring = Watching the token booth



The best carnival operators always watch the token booth

Not to stop the fun, but to ensure the park runs smoothly

Monitoring Usage

ID	Column Name	Description
1	sql_pool_name	OOTB, Default or user created Custom SQL Pool.
2	timestamp	Timestamp for when there is a state change for the pools ie. A pool goes under pressure.
3	max_resource_percentage	Max resource percentage for the pool.
4	is_optimized_for_reads	Indicates if the SQL pool is configured for read-optimized workloads.
5	is_default_sql_pool	Indicates whether this SQL pool is the default pool for the workspace.
6	current_workspace_capacity	Capacity SKU being used by the workspace.
7	is_pool_under_pressure	Determined by DQP. New event logged when state changes.

Pool Usage Monitoring

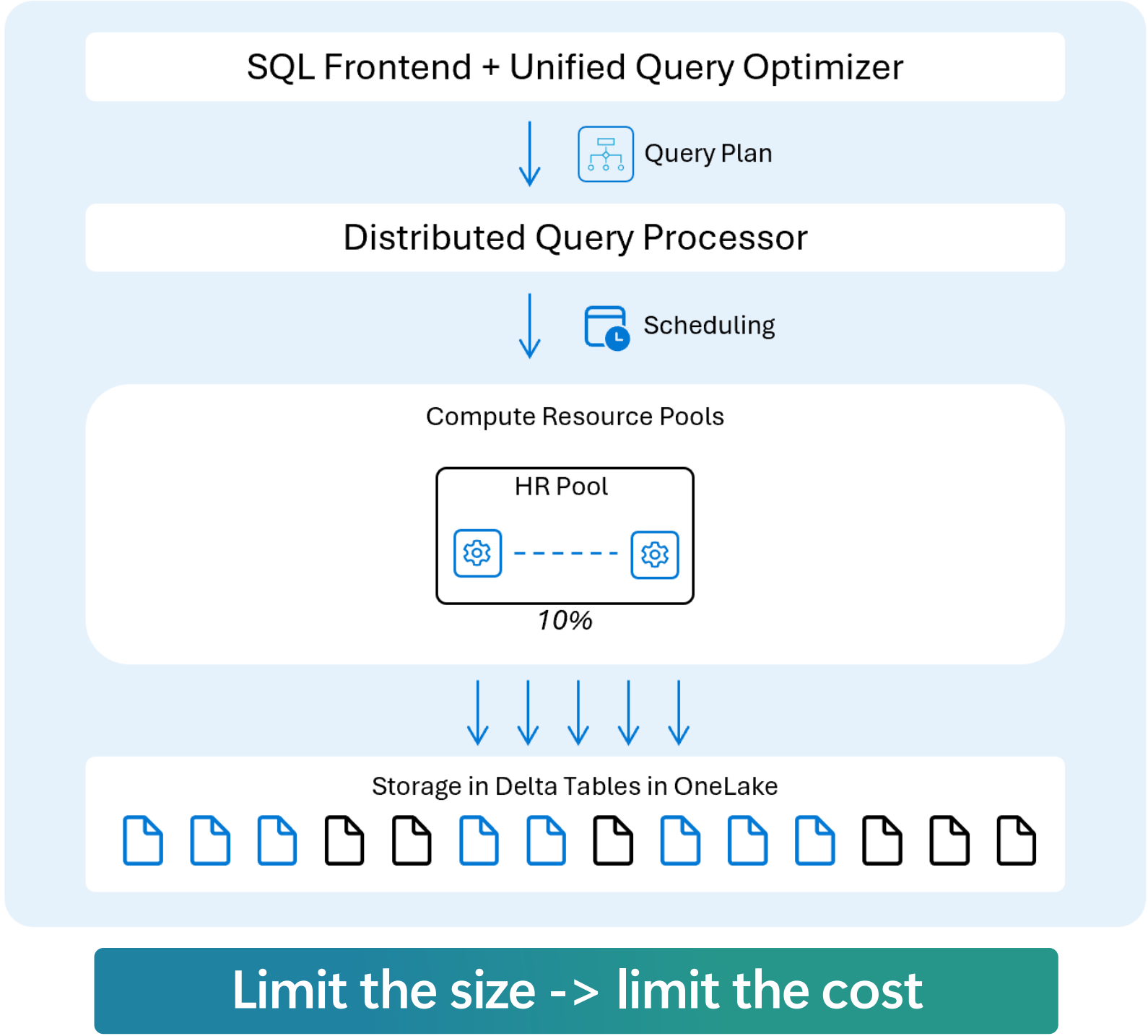
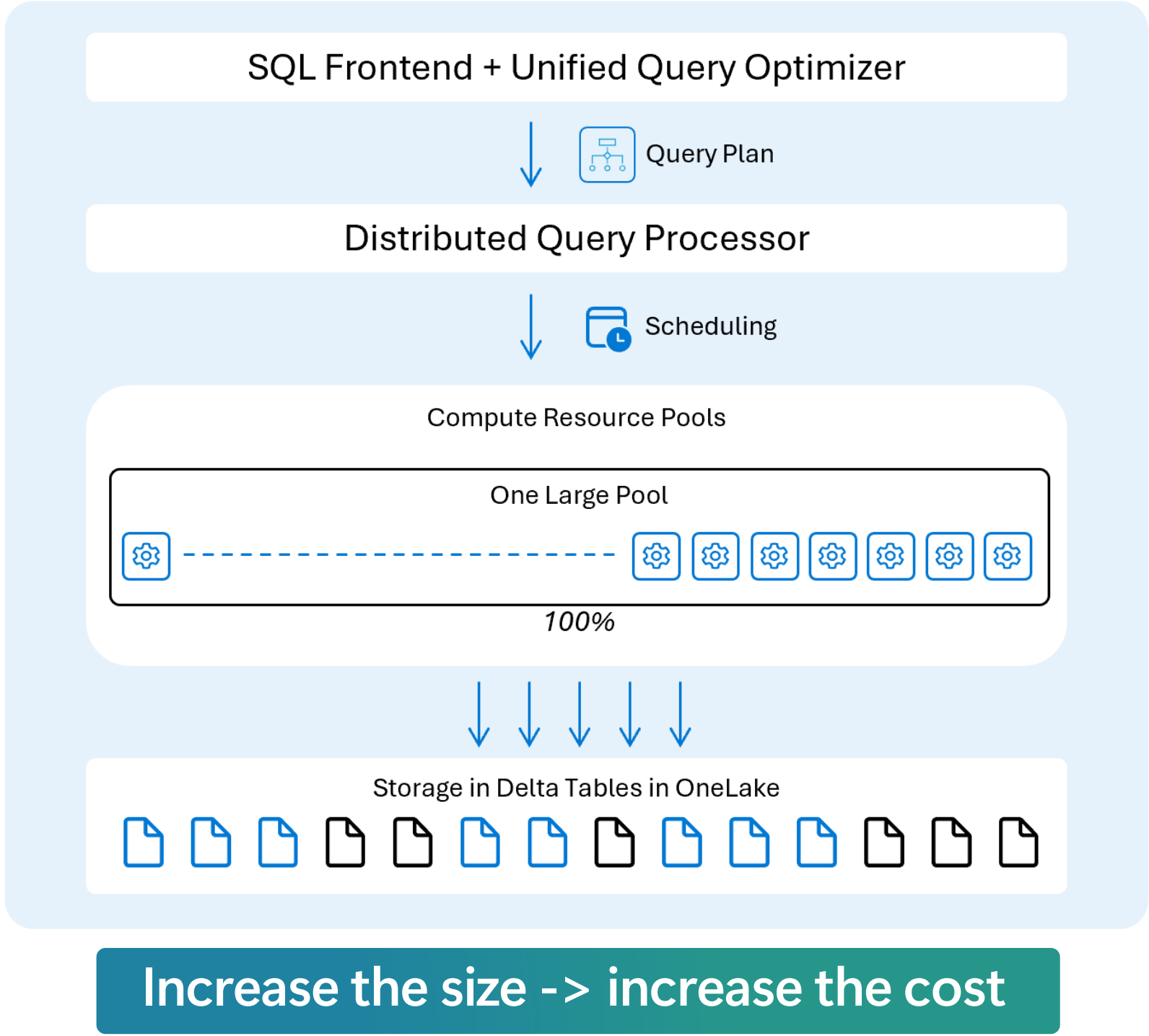
A new field, **sql_pool_name**, has been added to the **queryinsights.exec_requests_history** view to identify which queries went to what pool

Pool Pressure Monitoring

A new view, **queryinsights.sql_pool_insights**, allows you to monitor and identify changes in custom pool configurations and understand when a pool is under pressure

i.e. the pool is using all available resources allocated and has sustained that utilization for 1 minute.

Price Performance Scenarios



Demo

Closing Thoughts



Scale is automatic

(Distributed execution + Elastic compute)

Predictability = isolation

(Built-in Pools + Custom SQL Pools)

Cost monitoring = attribution

(Query Insights)

Quick Survey:

Tell us what works — and what does not — in Fabric Data Warehouse!



<https://aka.ms/fabric-data-warehouse-survey>

How was the session?



Complete Session Surveys in
Whova for your chance to WIN
PRIZES!

