Dremio Enterprise Operating Model

Executive Overview

The Dremio operating model for self-managed Dremio Software environments supports predictable and repeatable operation as the platform scales across the enterprise. As adoption grows and more users and workloads are added, the model provides a clear framework for maintaining performance and reliability. Note, this is taken care of for you with Dremio Cloud.

This model defines how capacity is planned, performance is monitored, and scaling decisions are made. This will ensure consistent outcomes for concurrent and business-critical workloads.

Core Design Principles

- **Workload Isolation**: Different workloads must not compete for the same resources
- **Proactive Scaling**: Capacity is added based on leading indicators, not reactive metrics
- **Reactive Controls**: Lagging indicators trigger corrective scaling actions
- **Standardization***: Refer to the Dremio system requirements documentation here. Scaling is only supported by adding additional engines or by selecting a different Dremio-prescribed T-shirt engine size
- **Operational Transparency**: SLAs, metrics, and alerts are clearly defined

Architecture & Operating Model

Generally, if the metadata is separate between workloads - which is often the case with different business units, we recommend different Dremio clusters for separating change management schedules and resource/upgrade decisions in the organization.

If data or metadata is shared across workloads, then Dremio scaling can be done with separate engines aligned to workload type:

- User Query Engines (interactive and application queries)
- Metadata Engines (catalog and metadata operations)
- Reflection Refresh Engines (reflection planning and refreshes)

This architecture enforces isolation, prevents resource contention, and enables targeted scaling. For further information on Dremios architecture, please review our documentation.

Capacity Management Framework

Leading Indicators (Proactive Scaling)

- New use cases or analytics workloads

- Increase in concurrent users
- Growth in query volume or complexity

Standard Actions:

- New workloads: Deploy a minimum of **two engines of the appropriate T-shirt size via round-robin distribution model**
- Concurrency or volume growth: Deploy **one or more additional appropriate T-shirt size engines via round-robin distribution model**
- Buffer capacity: Provision **25% buffer capacity** within each engine to absorb demand spikes and maintain performance stability

Lagging Indicators (Reactive Scaling)

- Query execution errors (e.g., out of memory, resource error)
- SLA breaches at query execution stages

Standard Action:

- Deploy **one or more additional appropriate T-shirt size engines** and rebalance workloads via WLM rules
- Vertically scale the Master Coordinator when pending, planning, or execution planning SLAs are not met

Business Outcomes

Together, these principles and architecture enable Dremio to deliver predictable, scalable, and reliable performance for business-critical workloads.Consistent, predictable analytics performance

Dremio Standard (T-Shirt) Engine Sizes

Executor Size	Executors per Engine	Memory per Executor
2XSmall	1	64 GB
XSmall	1	128 GB
Small	2	128 GB
Medium	4	128 GB
Large	8	128 GB
XLarge	12	128 GB
2XLarge	16	128 GB
3XLarge	24	128 GB
4XLarge	32	128 GB

Capacity Scaling Triggers and Metrics

Query Execution Error Due to Insufficient Resources (*outcomeReason* in queries.json)

Description	Recommended Threshold	Action
OUT_OF_MEMORY	1% of queries running out of direct memory	Add Engine and move workload
RESOURCE ERROR	1% of queries running out of heap memory	Add Engine and move workload
Execution Setup Exception	1% of queries exhibiting node disconnects	Add Engine and move workload
ChannelClosedException (fabric server)	1% of queries exhibiting node disconnects	Add Engine and move workload
CONNECTION ERROR: Exceeded timeout	1% of queries exhibiting node disconnects	Add Engine and move workload

Query Stage Performance SLAs (queries.json via external monitoring tool / Query Profile)

Query Stage	Recommended Threshold	Action
0: TotalDuration (*finish - start*)	p90 SLA align with customer need	See below (adding 1-8)
1: Pending (*pendingTime*)	p90 should never be above 2sec	Vertically scale Master based on supported ratios
2: Metadata Retrieval (*metadataRetrievalTime*)	p90 should never be above 5sec	Switch to Iceberg Table Format if raw Parquet
3: Planning (*planningTime*)	p90 should never be above 2sec	Vertically scale Master
4: Engine Start [IGNORE] (*engineStartTime*)	N/A	N/A
5: Queued (*queuedTime*)	p90 should never be above 2sec	Add Engine and move workload
6: Execution Planning(*executionPlanningTime*)	p90 should never be above 2sec	Vertically scale Master
7: Starting (*startingTime*)	p90 should never be above 2sec	Add Engine and move workload
8: Running (*runningTime*)	p90 SLA align with customer need	Add Engine and move workload

All italicized values can be found in queries.json as represented in parentheses above

* *A higher node count may be recommended depending on query memory requirements and other contributing factors, such as data volume*