databricks

Fine-Tuning: Choosing the Right Cluster

LECTURE

Fine-Tuning: Choosing the Right Cluster

This lecture covers how to choose the right cluster type, autoscaling, spot instances, Photon, and cluster optimization best practices.

Cluster Types

ALL PURPOSE COMPUTE

- Designed to handle interactive workloads, including streaming workloads.
- Enable Auto-Scale to add capacity when needed and reduce time to answer
- Security considerations must be considered as auto-scaling can introduce additional risks.

JOBS COMPUTE

- Run on ephemeral clusters that are created for the job, and terminate on completion
- Pre-scheduled or submitted via API
- Single-user
- Great for isolation and debugging
- Production and repeat workloads
- Lower cost

SQL WAREHOUSE

- Built for high concurrency ad-hoc SQL analytics and BI serving
- Photon included
- Recommended shared warehouse for ad-hoc SQL analytics, isolated warehouse for specific workloads
- Serverless available for instant startup and lower TCO

developing code before pushing it into production. These interactive clusters, also known as all purpose compute, are intended for development work and iterative testing. Jobs compute is designed specifically for running workflow jobs. When creating a workflow job, jobs compute should be used rather than all purpose compute, as it provides the appropriate resources tailored for those tasks. Jobs compute clusters are single user, good for isolation and debugging, and generally come at a lower cost. The third type is SQL warehouses, which have Photon built in for performance and are intended for high concurrency, ad hoc SQL queries, and BI serving. SQL warehouses are specifically used for SQL workloads. Each of these cluster types serves a different purpose: all purpose compute for development, jobs compute for job execution, and SQL warehouses for BI and SQL analytics.

Autoscaling

- Dynamically resizes cluster based on workload
  - Can run faster than a statically-sized, under-provisioned cluster
  - Can reduce overall costs compared to a statically-sized cluster
- Setting range for the number of workers requires some experimenting

Use Case	Auto Scaling Range
Ad-hoc usage or business analytics	Large variance
Production batch jobs	Not needed or buffer on upper limit
Streaming	Available in Delta Live Tables

Autoscaling enables a Databricks cluster to automatically adjust its size based on workload demands. When more resources are required for jobs, queries, or tasks, autoscaling can increase the number of worker nodes, improving performance as the workload grows. When the demand decreases, autoscaling scales the cluster down, reducing operational costs compared to keeping a statically sized cluster running at all times.

To use autoscaling, you activate the feature and set a minimum and maximum number of workers. This range often requires some experimentation, typically during development or analytics workloads, to find an optimal balance. For development and ad hoc analytics, it can be helpful to allow for a higher upper limit, giving flexibility for fluctuating use. As a workflow matures and its data volumes become predictable, autoscaling might become less necessary. For some production batch jobs, a fixed-size cluster is sufficient, but setting an upper limit can help handle occasional spikes in data volume.

Autoscaling is also supported in streaming environments and with Delta Live Tables. On the horizon, serverless clusters are becoming available. These handle autoscaling automatically and further simplify workload management by optimizing resources without manual intervention. This flexibility means clusters can efficiently handle changing workloads without wasting resources.

Spot Instances

- Use spot instances to use spare VM instances for below market rate
  - Great for ad-hoc/shared clusters
  - Not recommended for jobs with mission critical SLAs
  - Never use for driver!Combine on-demand and spot instances (with custom spot price) to tailor clusters to different use cases

SLA	Spot or On-Demand
Non-mission critical jobs	Driver on-demand and workers spot
Workflows with tight SLAs	Use spot instance w/fallback to on-demand

You can reduce costs by using spot instances, which are provided by cloud service providers at a lower price because they are currently unused. These instances let you take advantage of available compute resources at below-market rates, with the understanding that the provider can reclaim them if demand increases. This approach is especially useful for non-mission critical jobs. For added stability, you might configure your cluster with an on-demand VM instance for the driver, ensuring the job's control layer remains stable, while using spot instances for the workers to save costs. If the provider needs to reclaim the spot instances, your main job won't fail completely because the driver is still running.

For important jobs or workflows with strict SLAs, you can use spot instances but enable fallback to on-demand VMs. This ensures you benefit from cost savings when possible, but also avoid disruption if spot capacity is withdrawn. Using this setup, you maintain reliability for critical tasks while minimizing costs whenever the cloud environment allows.

Photon

World record achieving query engine with zero tuning or setup

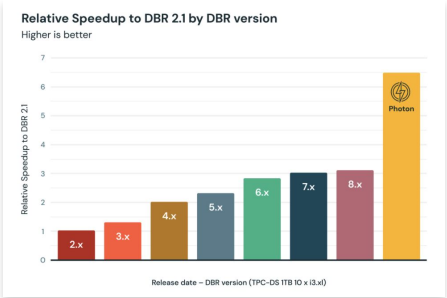
Save on compute costs
- ETL customers are saving up to 40% on their compute cost

Fast query performance
- Built for modern hardware with up to 12x better price/perf compared to other cloud data warehouses

No code changes
- Spark APIs that can do exploration, ETL, big data, small data, low latency, high concurrency, batch, and streaming

Broad language support
- Support for SQL, Python, Scala, R, and Java

Databricks Sets Official Data Warehousing Performance Record



So what is Photon?
Photon, our World record setting engine is seeing serious adoption by customers:

- Photon obliterated the previous TPCDS data warehouse world record by more than 2x!
- In ETL, customers are seeing a 40% reduction in their compute spend
- 6x better price/perf of other cloud data warehouses and an overall 2-3x reduction in their query times over OSS Spark
- In the past quarter, usage increased by 5x by our customers!
- Customers are able to adoption Photon easily: no code changes or tuning and are able to use their language of choice; SQL, Python, Scala, R, and Java

Finally, customers are able to do exploration, ETL, big data, small data, low latency, high concurrency, batch, and streaming; all on a single engine and a single API set.
Chart

3. Fill out the following template for the feature(s) described on this slide
Choices: supported, not planned, planned insert quarter*

Target release: v3.47

AWS E2:Q2 (June) LEGACY [ST:Q2, PVC:Q2 (target)]
Azure E2:Q3 (target) GFM [Gov Q2, China Q2]:
GCP E2 : TBD
*future dates are subject to change

Cluster Optimization Recommendations

1. **DS & DE development:** all-purpose compute, auto-scale and auto-stop enabled, develop & test on a subset of the data
2. **Ingestion & ETL jobs:** jobs compute, size accordingly to job SLA
3. **Ad-hoc SQL analytics**: (serverless) SQL warehouse, auto-scale and auto-stop enabled
4. **BI Reporting:** isolated SQL warehouse, sized according to BI SLAs
5. **Best practices**:
    a. Enable spot instances on worker nodes
    b. Use the latest LTS Databricks Runtime when possible
    c. Use Photon for best TCO when applicable
    d. Use latest gen VM, start with general purpose, then test memory/compute optimized

For cluster optimization, the recommended approach depends on your workload. For data science and data engineering development, use all purpose compute clusters with autoscale enabled and remember to set up auto stop, so you're not paying for idle compute resources. It's also best to develop and test on a subset of data to keep cluster resource use minimal during development. For ingestion or ETL jobs, configure jobs compute clusters and size them according to the job's SLA requirements. For ad hoc analysis and SQL analytics, use SQL warehouses, enable autoscaling, and specify the number of workers needed. Auto stop should also be enabled, or better yet, use serverless compute, which has significant cost and efficiency benefits for a range of jobs, including BI reporting. Isolated SQL warehouses should be used for BI workloads and sized as per business requirements.

Best practices also include enabling spot instances for worker nodes and running the latest LTS Databricks runtime for performance and stability—at the time of this video, version 14.3 LTS is standard, but version 15 is in beta. Use Photon for improved performance and cost optimization where possible, and select the latest generation of VMs, trying general purpose first before testing memory or compute-optimized types. Always tailor your cluster size and configuration according to workload needs, refining sizing as you move from development to production.

Thank you for completing this lesson and continuing your journey to develop your skills with us.