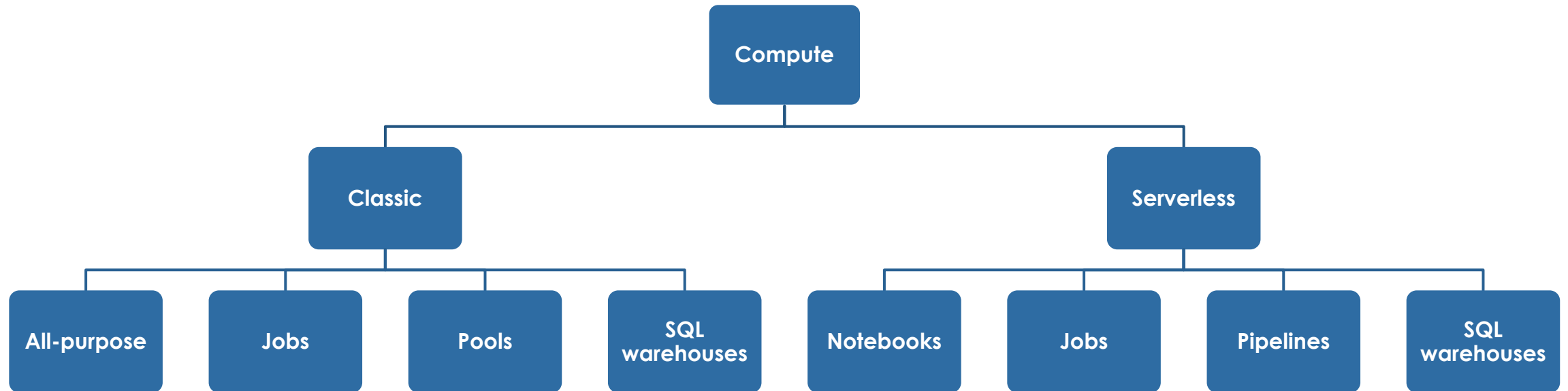


Cluster Best Practices

Learning Objectives

- ▶ Compute types
- ▶ Best practices and recommendations

Databricks Compute



Classic Compute Types

All-purpose cluster

- ▶ Used to run interactive notebooks
- ▶ Can be created using the Web UI, Command Line (CLI), or REST API
- ▶ Can be manually terminated, or auto terminated after a period of inactivity
- ▶ DBU: More expensive

Jobs cluster

- ▶ Used to run automated jobs
- ▶ Created automatically by Databricks job scheduler
- ▶ Terminate when the job is completed
- ▶ DBU: Less expensive

Instance Family

☒ Multi node

☐ Single node

Access mode ⓘ

Dedicated (formerly: Single user) | ▾

Single user or group access ⓘ

Derar Alhussein | ▾

Performance

Databricks runtime version ⓘ

Runtime: 16.4 LTS (Scala 2.12) (Scala 2.12, Spark 3.5.2) | ▾

☒ Use Photon Acceleration ⓘ

Worker type ⓘ

Standard_D4ds_v5 16 GB Memory, 4 Cores | ▾

Min workers

2 | ▴ ▾

Max workers

8 | ▴ ▾

☒ Spot instances

Driver type

Same as worker 16 GB Memory, 4 Cores | ▾

☒ Enable autoscaling ⓘ

Worker type ⓘ

Standard_D4ds_v5 16 GB Memory, 4 Cores | ▾

General purpose

Standard_DS3_v2 14 GB Memory, 4 Cores

Standard_DS4_v2 28 GB Memory, 8 Cores

Standard_DS5_v2 56 GB Memory, 16 Cores

Memory optimized

Standard_DS12_v2 28 GB Memory, 4 Cores

Standard_DS13_v2 56 GB Memory, 8 Cores

Storage optimized

Standard_L8s_v3 64 GB Memory, 8 Cores

Standard_L16s_v3 128 GB Memory, 16 Cores

Standard_L32s_v3 256 GB Memory, 32 Cores

Compute optimized

Derar Alhussein © Udemy | Databricks Certified Data Engineer Associate - Preparation

Instance Family

| VM Category | Workload Type |
|--------------------------|---|
| Memory Optimized | <ul style="list-style-type: none">• Where a lot of shuffle and spills are involved• When Spark caching is needed• For ML workloads |
| Compute Optimized | <ul style="list-style-type: none">• Structured Streaming jobs• ELT with full scan and no data reuse• To run OPTIMIZE and Z-order Delta commands |
| Storage Optimized | <ul style="list-style-type: none">• To leverage Delta caching• For ad hoc and interactive data analysis• ML and DL workloads with data caching |
| GPU Optimized | <ul style="list-style-type: none">• ML and DL workloads with an exceptionally high memory requirement |
| General Purpose | <ul style="list-style-type: none">• Used in absence of specific requirements• To run VACUUM Delta command |

| Worker type ⓘ | |
|--------------------------|---------------------------|
| Standard_D4ds_v5 | 16 GB Memory, 4 Cores ▼ |
| General purpose | |
| Standard_DS3_v2 | 14 GB Memory, 4 Cores |
| Standard_DS4_v2 | 28 GB Memory, 8 Cores |
| Standard_DS5_v2 | 56 GB Memory, 16 Cores |
| Memory optimized | |
| Standard_DS12_v2 | 28 GB Memory, 4 Cores |
| Standard_DS13_v2 | 56 GB Memory, 8 Cores |
| Storage optimized | |
| Standard_L8s_v3 | 64 GB Memory, 8 Cores |
| Standard_L16s_v3 | 128 GB Memory, 16 Cores |
| Standard_L32s_v3 | 256 GB Memory, 32 Cores |
| Compute optimized | |

Spot Instances

- ▶ Low-cost virtual machines
- ▶ Users bid for unused capacity, but the instances can be interrupted if the market price exceeds the bid.

| Worker type ⓘ | Min workers | Max workers | |
|--|-------------|-------------|--|
| Standard_D4ds_v5 16 GB Memory, 4 Cores ▼ | 2 ⬆️⬆️ | 8 ⬆️⬆️ | <input checked="" type="checkbox"/> Spot instances ⓘ |

Databricks pools

- ▶ Set of idle, and ready-to-use instances
- ▶ Reduce cluster start and auto-scaling times
- ▶ Cloud provider charges

Compute

All-purpose compute Job compute **Pools** SQL warehouses

[Compute](#) > [Pools](#)

Create Pool

Name

Min Idle ⓘ

Max Capacity ⓘ

Idle Instance Auto Termination ⓘ

Terminate instances above minimum after minutes of idle time.

Instance Type ⓘ

 16 GB Memory, 4 Cores ▼

Preloaded Databricks Runtime Version

 ▼

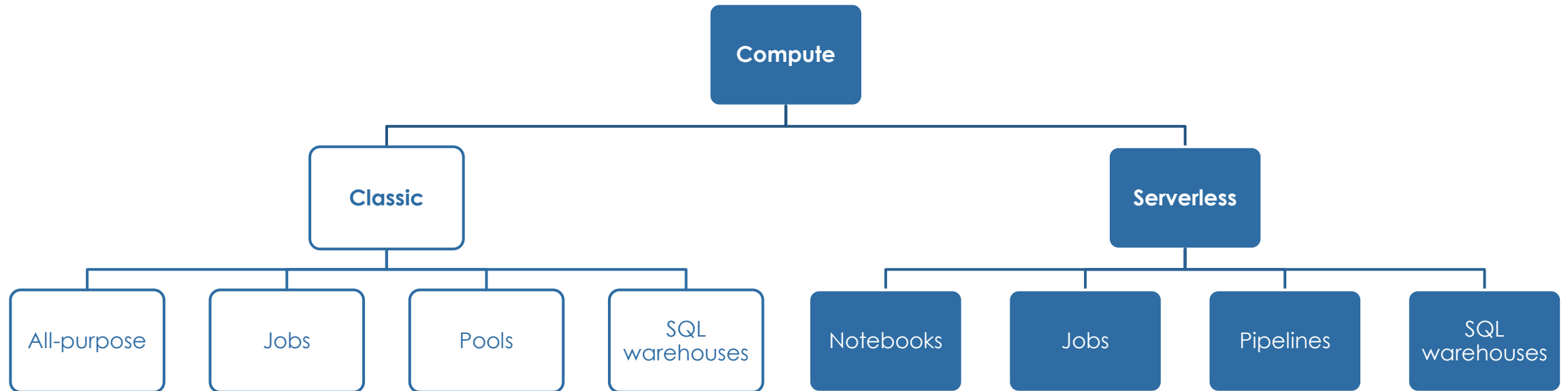
☐ Use Photon preloaded runtimes ⓘ

Instances Tags

On-demand/Spot

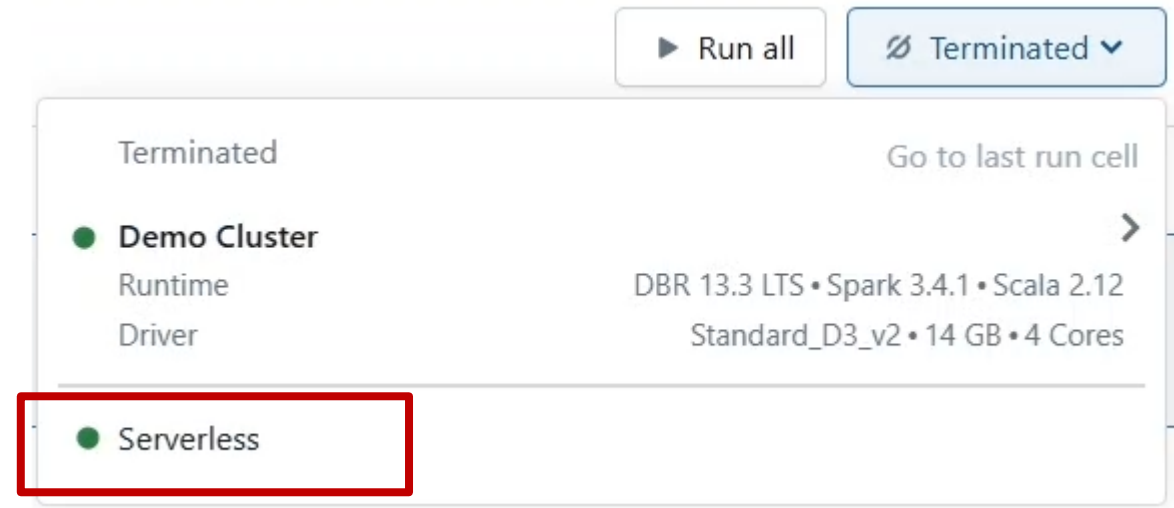
☒ All On-demand ☐ All Spot

Serverless Compute



Serverless Compute

- ▶ On-demand, scalable computing resources without configuration
- ▶ Avoid the operational overhead of managing and tuning clusters.
- ▶ Rapid start-up and scaling times
- ▶ Runs latest Databricks Runtime
- ▶ Support only Python and SQL



Serverless Compute Types

- ▶ Serverless compute for notebooks
- ▶ Serverless compute for jobs
- ▶ Serverless compute for pipelines

Serverless vs. Classic

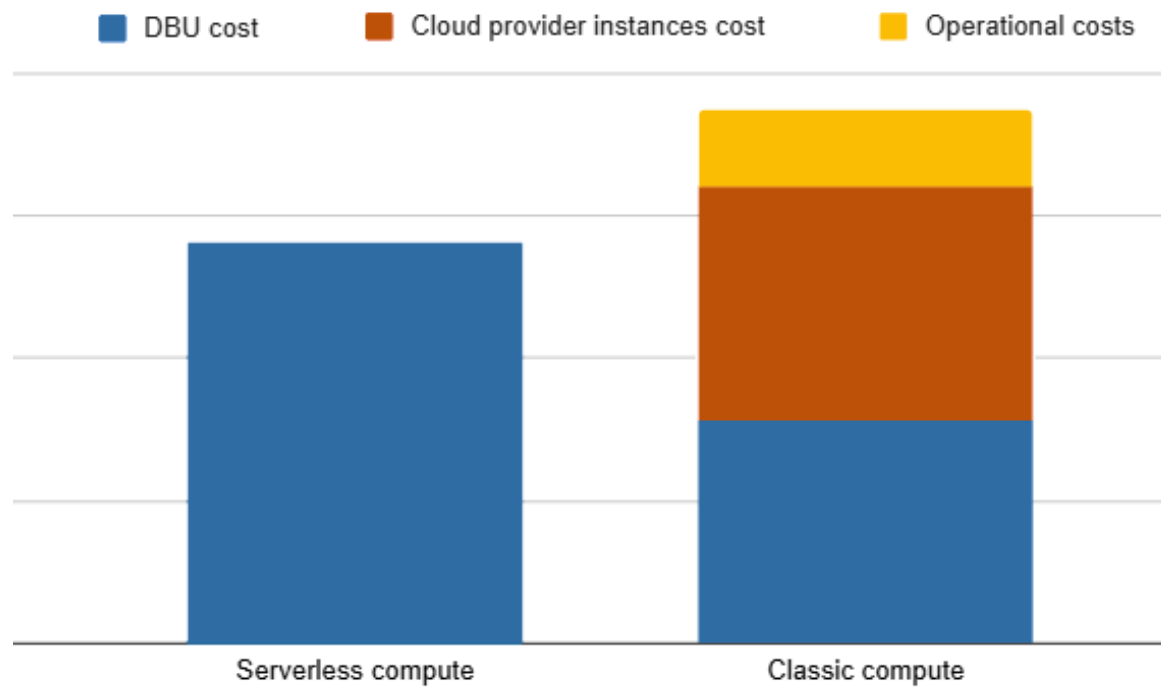
Serverless compute

- ▶ Fully managed without configuration
 - ▶ Photon Engine is enabled by default
 - ▶ Automatic access to latest features
 - ▶ Automatic instance type selection
 - ▶ Automatic scaling
- ▶ Rapid Startup time
- ▶ Supports only Python and SQL

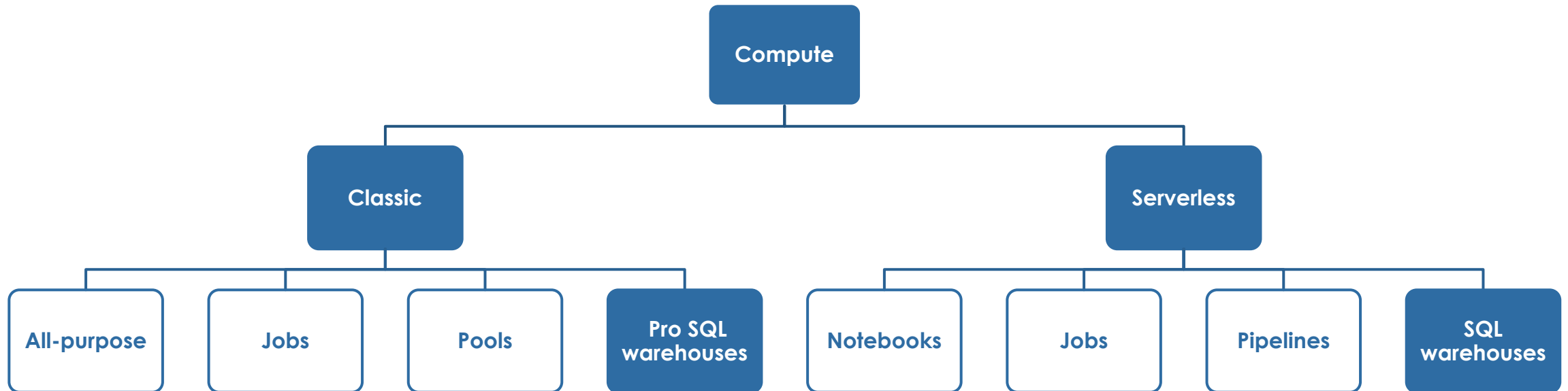
Classic compute

- ▶ Offers additional control and configuration
 - ▶ Photon Engine is optional
 - ▶ Requires manual upgrade of runtime versions
 - ▶ Requires manual selection of instance type
 - ▶ Autoscaling requires manual configuration
- ▶ Takes few minutes to start
- ▶ Supports Python, SQL, Scala, R, and Java

Pricing



SQL Warehouse



SQL Warehouse

- ▶ Highly optimized for SQL workloads
- ▶ Serverless SQL warehouses:
 - ▶ ETL
 - ▶ Business intelligence
 - ▶ Exploratory analysis
- ▶ Pro SQL warehouses:
 - ▶ If Serverless are not available
 - ▶ For custom-defined networking (databases/event buses in your cloud network or on-premises)

New SQL warehouse

Name: SQL warehouse name

Cluster size: X-Large 80 DBU / h

Auto stop: ☒ After 10 minutes of inactivity.

Scaling: Min. 1 Max. 1 clusters (80 DBU)

Type: ☒ Serverless ☐ Pro

Cancel Create