Why Cluster Size Matters in Databricks  
  
Performance (Execution Time)  
A small cluster may not have enough compute to handle large data volumes (e.g., billions of sales transactions).  
Results in slow joins, shuffles, and out-of-memory errors.  
Larger clusters with more worker nodes can parallelize Spark jobs → faster execution.  
  
Cost Optimization  
Bigger clusters = more $$ per hour.  
Over-provisioning wastes money if the workload doesn’t actually need that much compute.  
Under-provisioning causes job retries/failures → also more $$ indirectly.  
  
Workload Type  
ETL Batch Jobs → Often need burst compute power → use autoscaling clusters (scale up/down).  
Streaming Jobs → Need steady resources to avoid micro-batch delays.  
Ad-hoc Analytics / ML → Need flexibility; often use smaller clusters for exploration.  
  
Data Skew & Shuffles  
If data is skewed (e.g., one product has 70% of sales), a small cluster can’t distribute workload evenly.  
Bigger clusters (or optimized partitioning) reduce bottlenecks.

