OPERATIONAL ANALYTICS

Analytics in 15

Cost Optimization for OpenSearch Workloads

Gene Alpert (he/him)

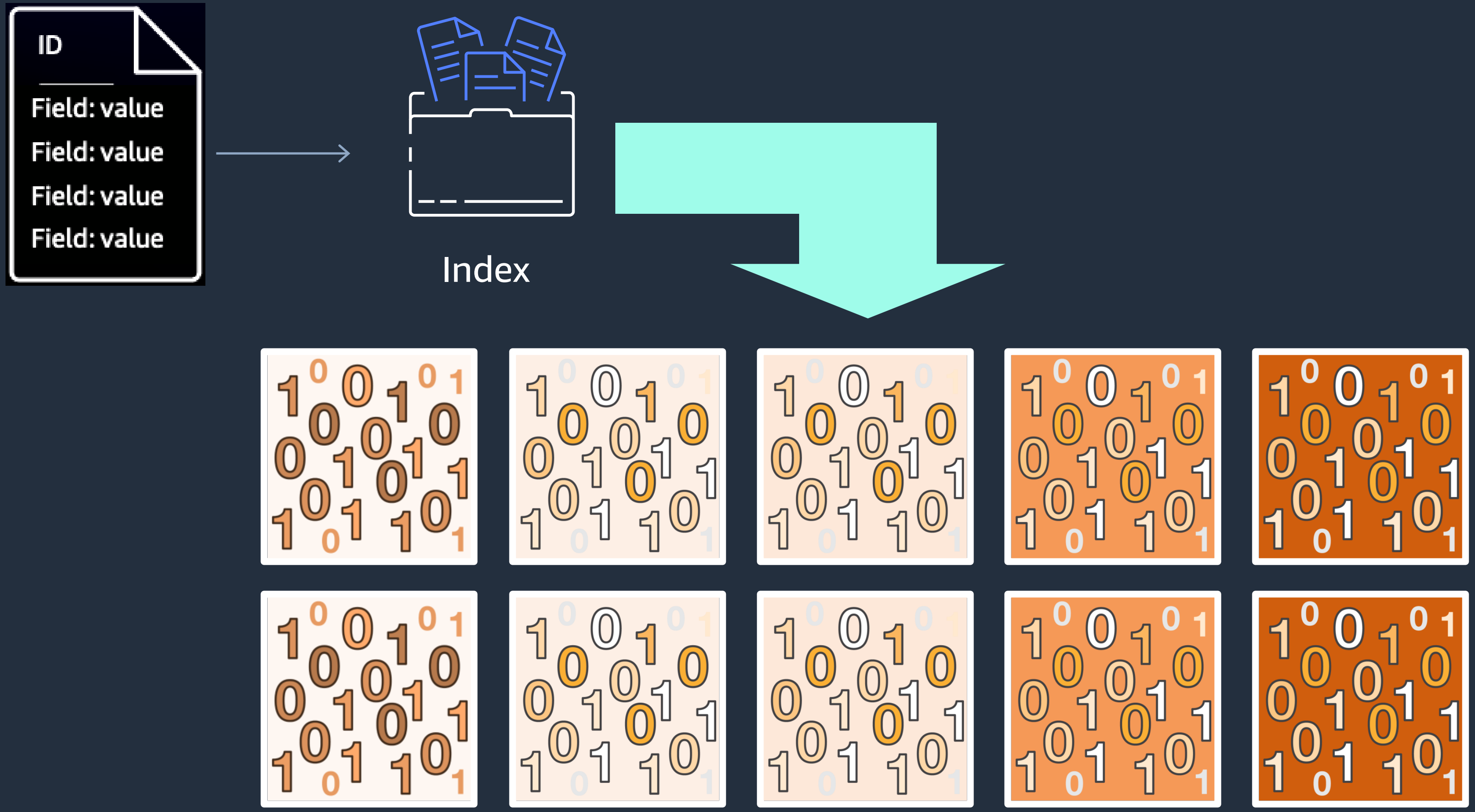
Sr. Analytics Specialist
Amazon Web Services

Four keys to lower cost with OpenSearch Service

- Shard strategy
- Latest generation Graviton2 instances and EBS gp3 volumes
- OpenSearch Serverless
- Storage tiering (for time series data)

Shard strategy

Indexes are composed of shards



ID

Field: value

Field: value

Field: value

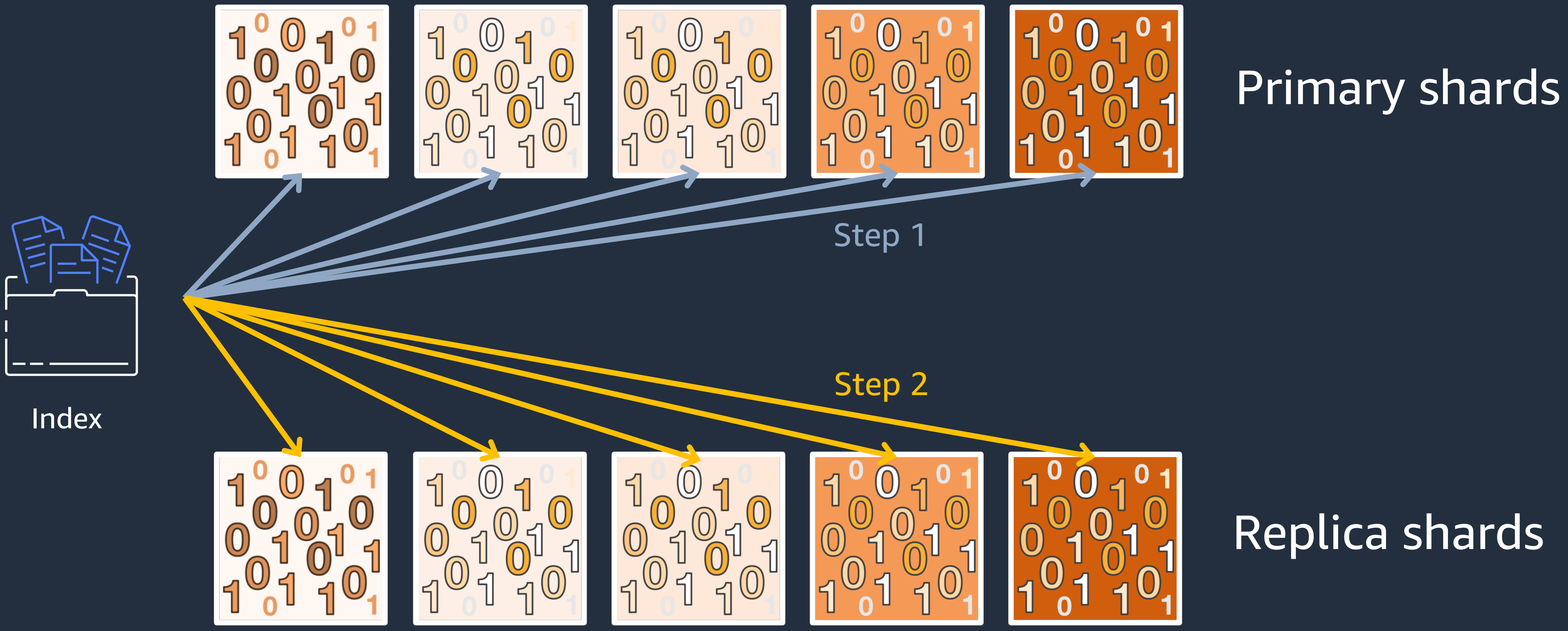
Field: value

Index

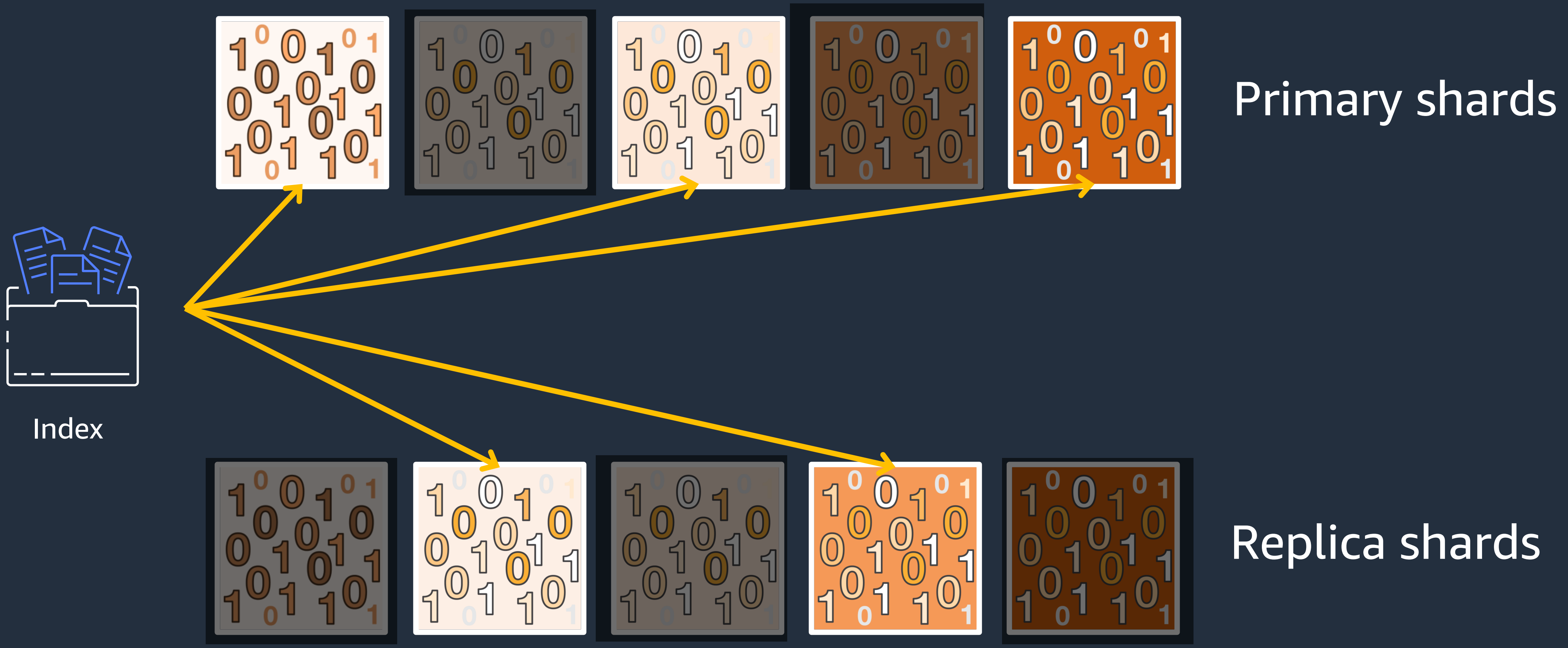
Primary shards

Replica shards

Indexing operations touch all shards

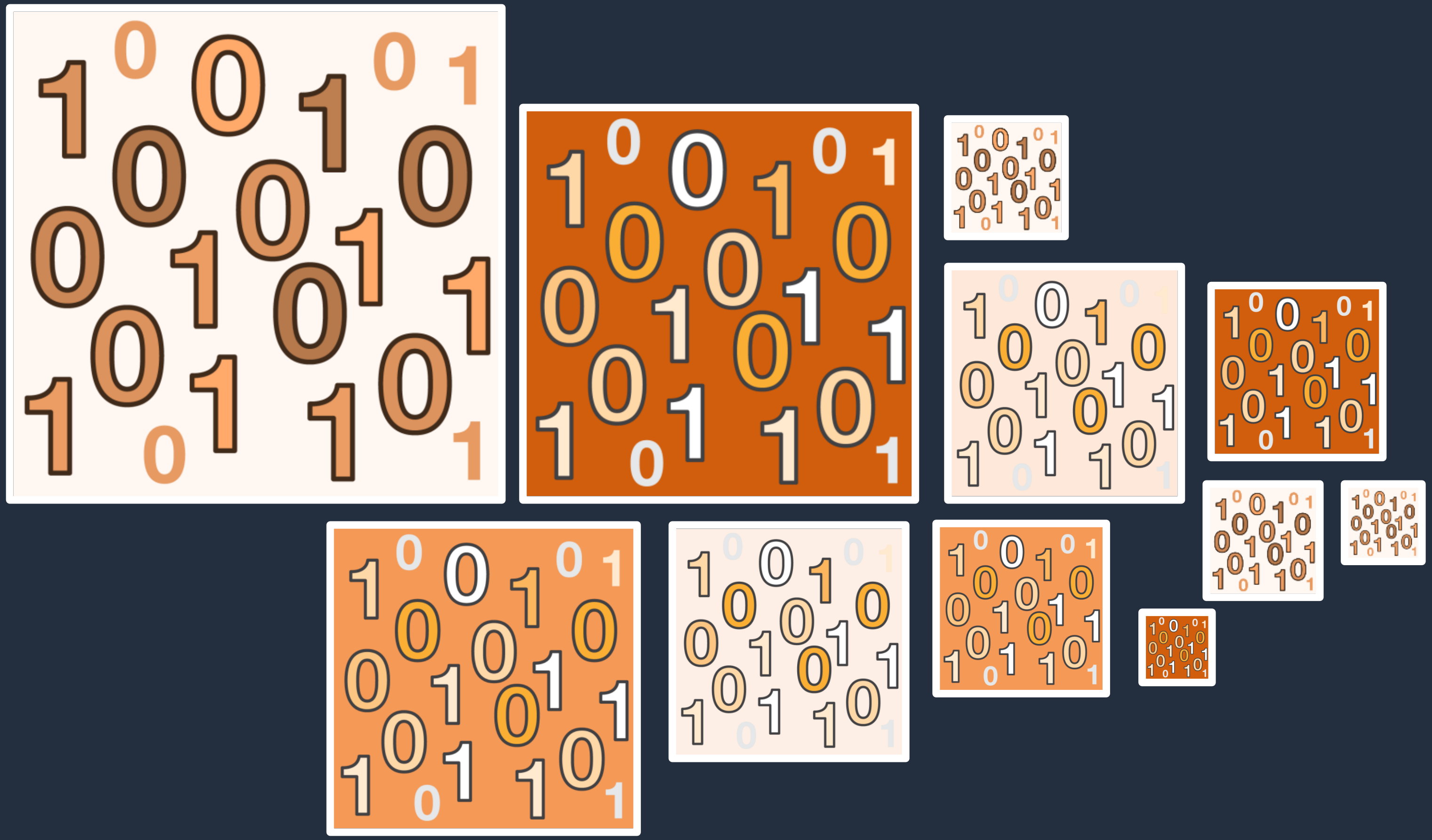


Search operations touch n shards (n=primary shard count)



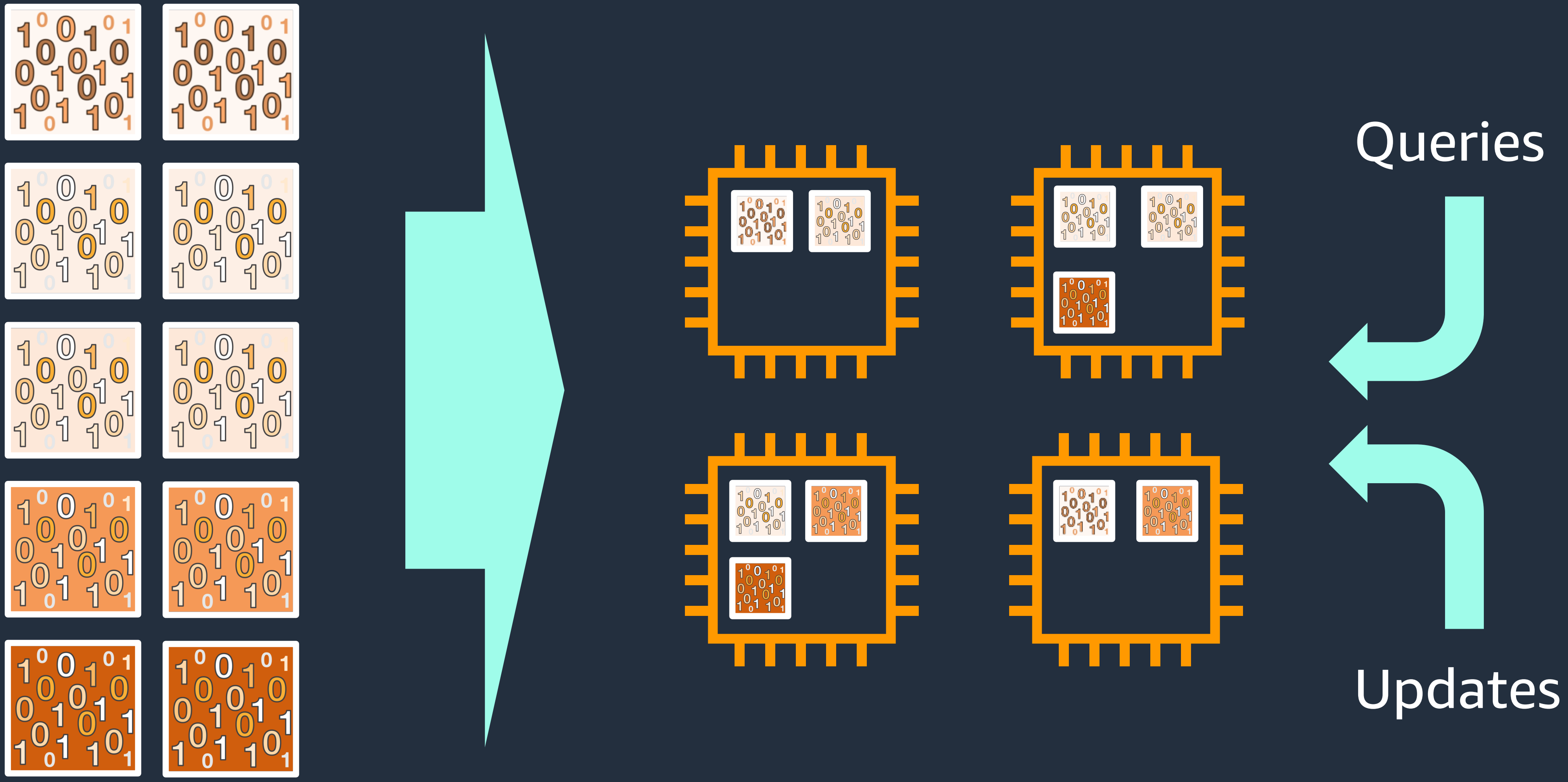
© 2023, Amazon Web Services, Inc. or its affiliates.

Shard size is important

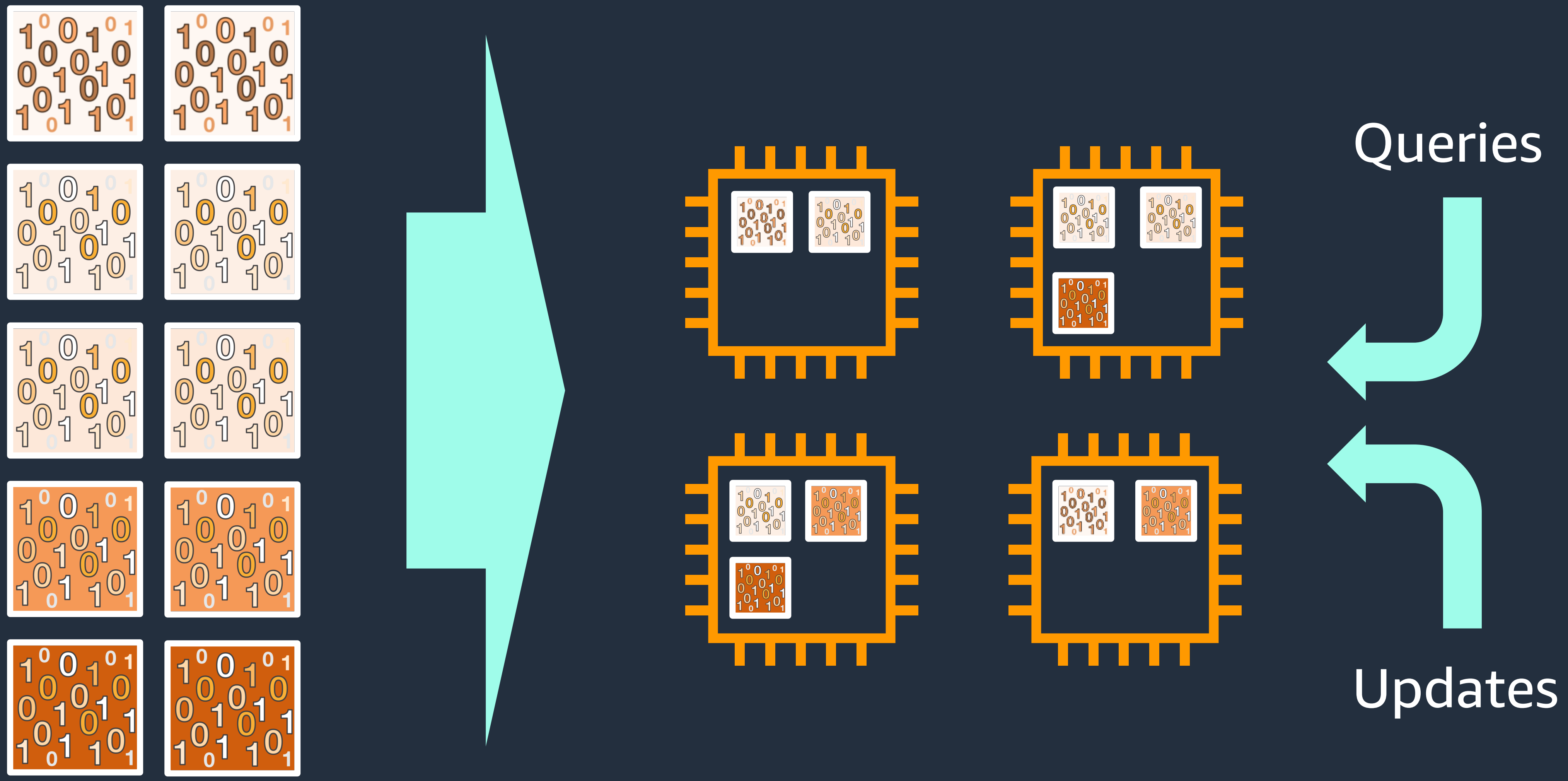


- 10 to 30 GB for search
- 30 to 50 GB for logs
- Do not exceed 50 GB

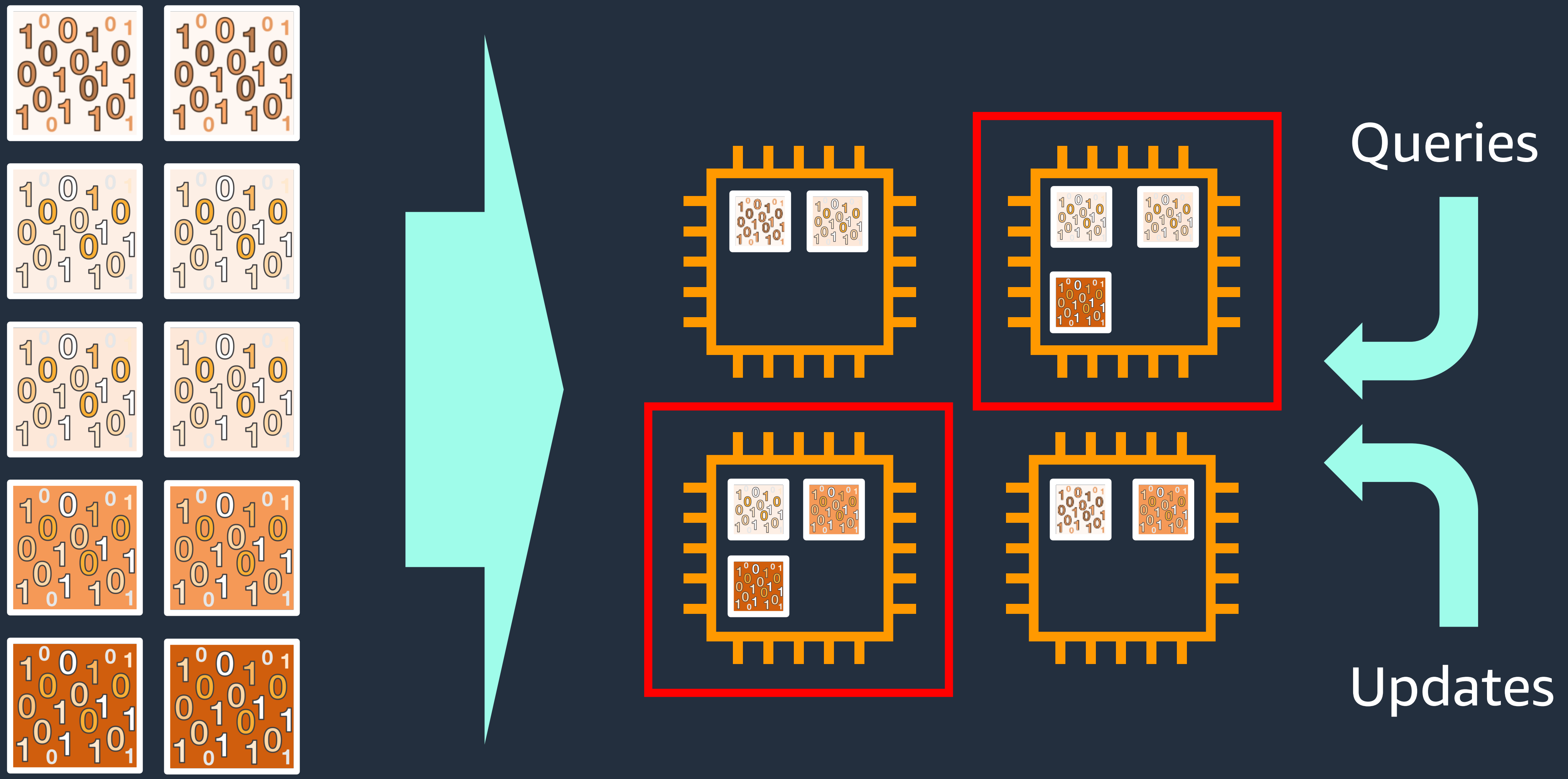
Shards are distributed across data nodes



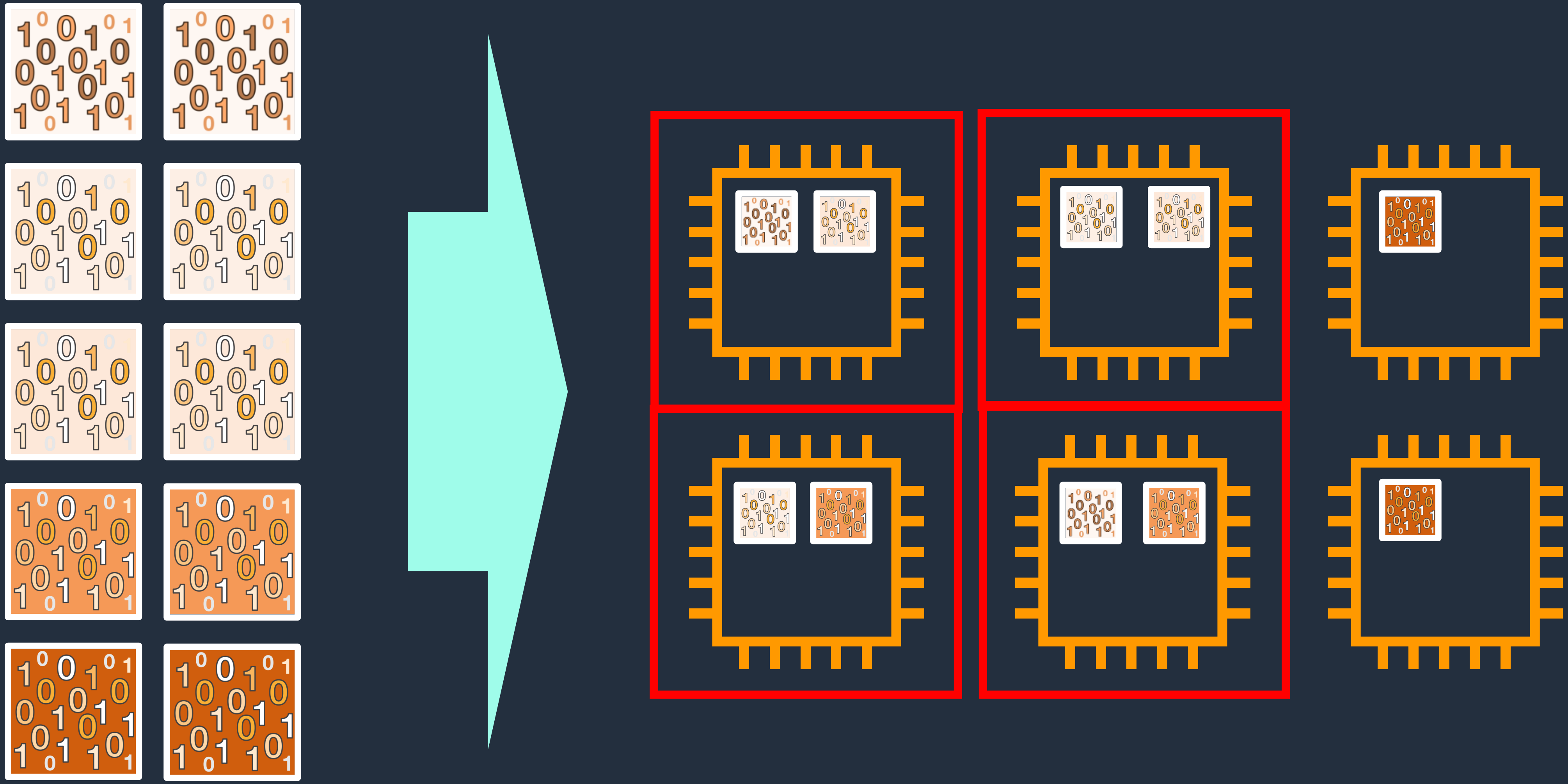
Beware of storage skew



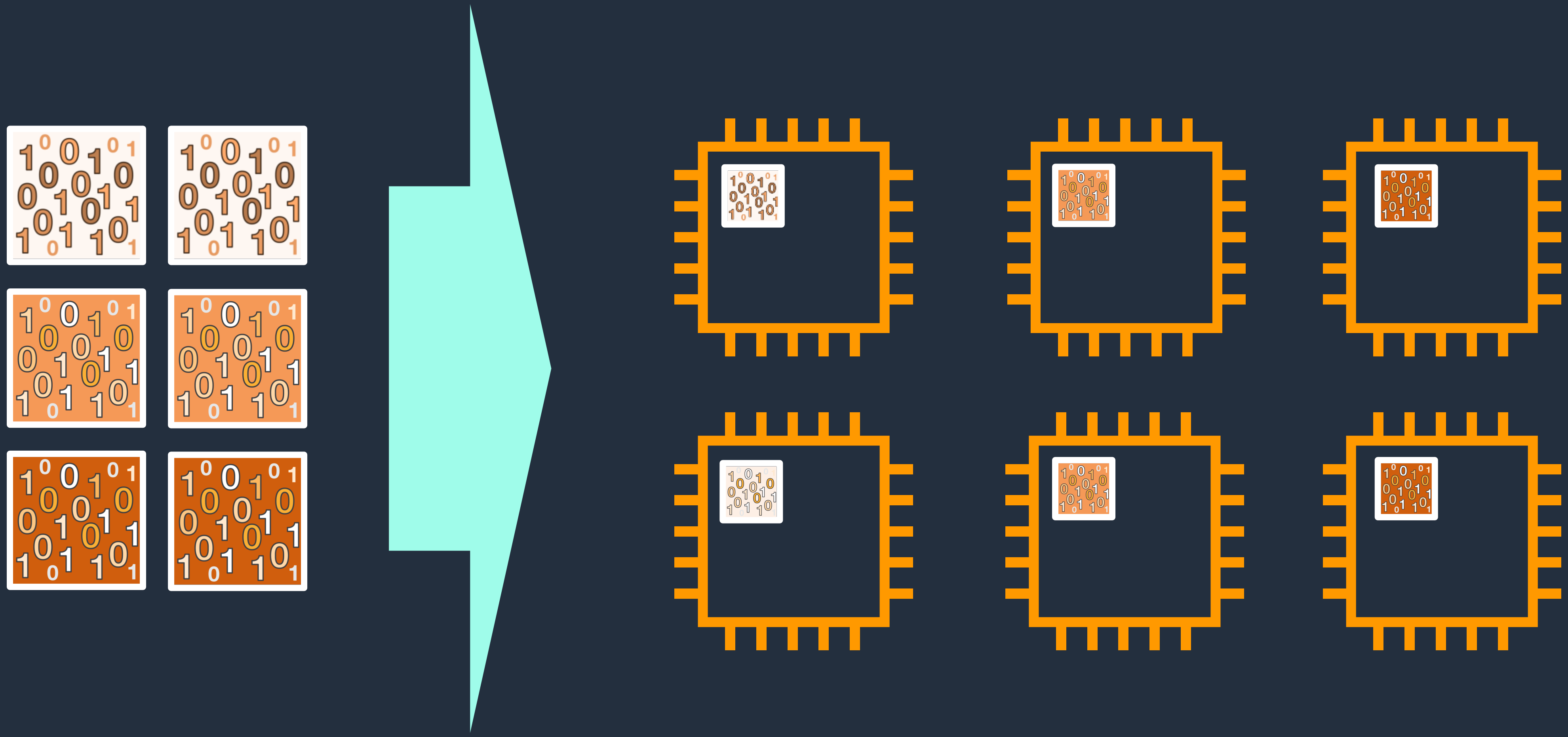
Storage skew



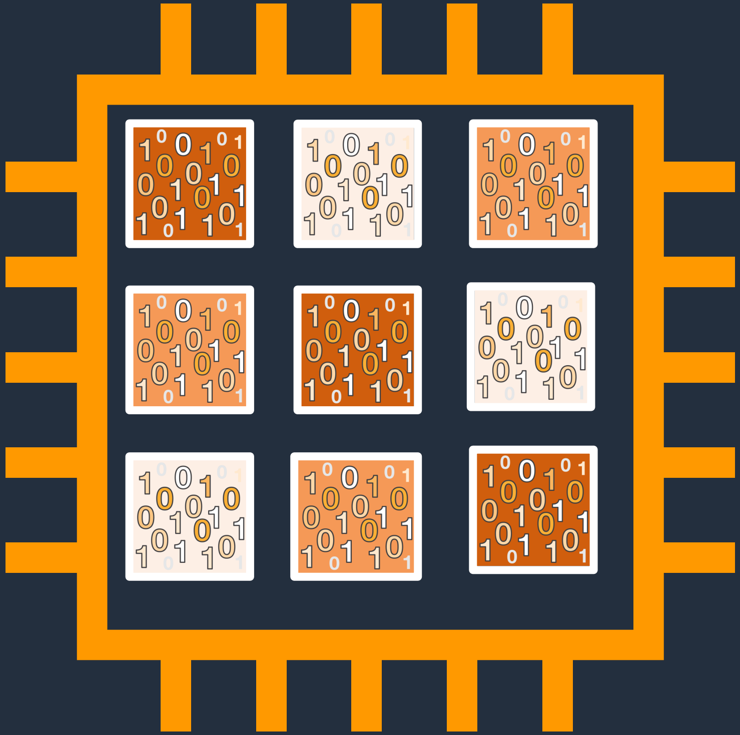
Storage skew



Balanced shard and storage distribution



Shards per data node



Shard to JVM heap: <25 per GiB

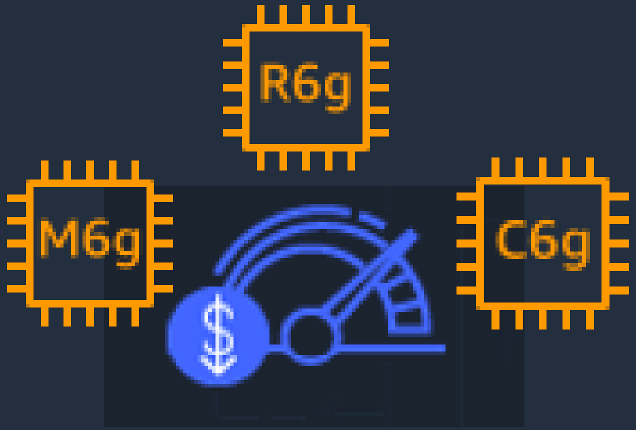
Shard to CPU: 1.5 shards

Use the _cat/allocation API to see shard count and distribution

Graviton2 instances and EBS gp3 volumes

AWS Graviton2 instances

- 38% improvement in indexing throughput
- 50% reduction in indexing latency
- 40% improvement in query performance
- 10% lower price per instance hour



Compared to corresponding Intel-based instances of the M5/C5/R5 families.

EBS gp3 volumes

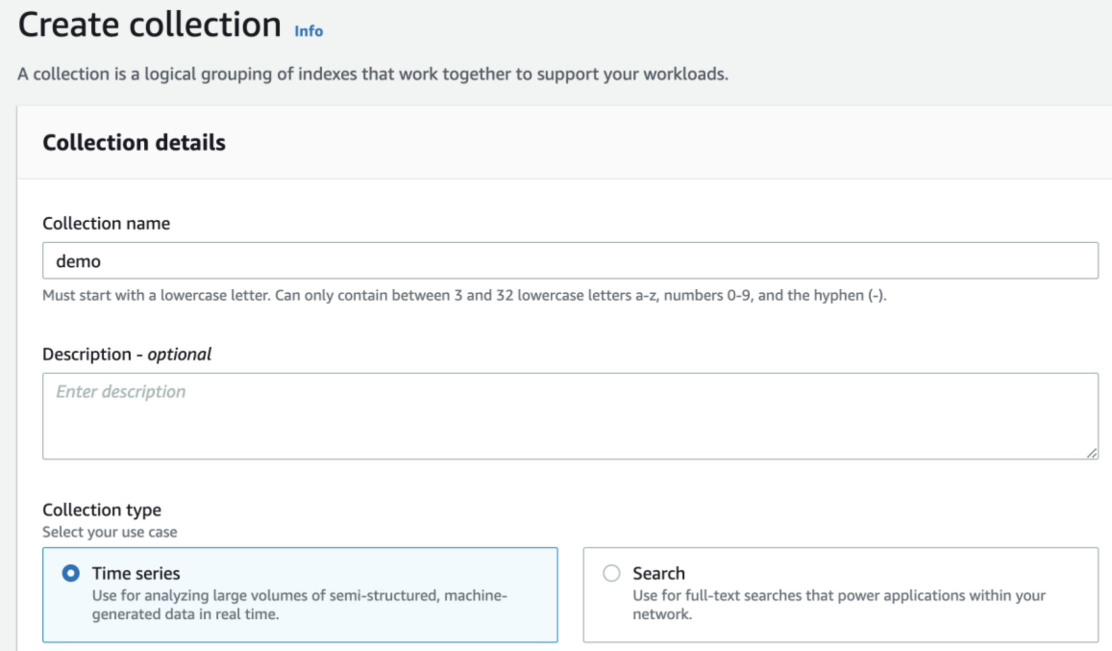
- Increased baseline performance (IOPS and throughput)
- Provision additional IOPS and throughput without increasing volume size
- 9.6% lower cost than EBS gp2 volumes

Amazon OpenSearch Serverless *preview*

OpenSearch Serverless key concepts

- **Collections:** A set of indexes that work together
  - Separate endpoints for OpenSearch and OpenSearch Dashboards
  - Can have specific or inherited access, network, and encryption policies
  - Optimized for ”time series” or “search”

- **OpenSearch Compute Units (OCUs):** Used to index and search collections
  - 6 GB RAM increments (min 4 per account)
  - Max OCUs can be set to control costs
  - Automatically provisioned for the workload
  - Shared across collections



Create collection

A collection is a logical grouping of indexes that work together to support your workloads.

Collection details

Collection name

demo

Must start with a lowercase letter. Can only contain between 3 and 32 lowercase letters a-z, numbers 0-9, and the hyphen (-).

Description - *optional*

Enter description

Collection type

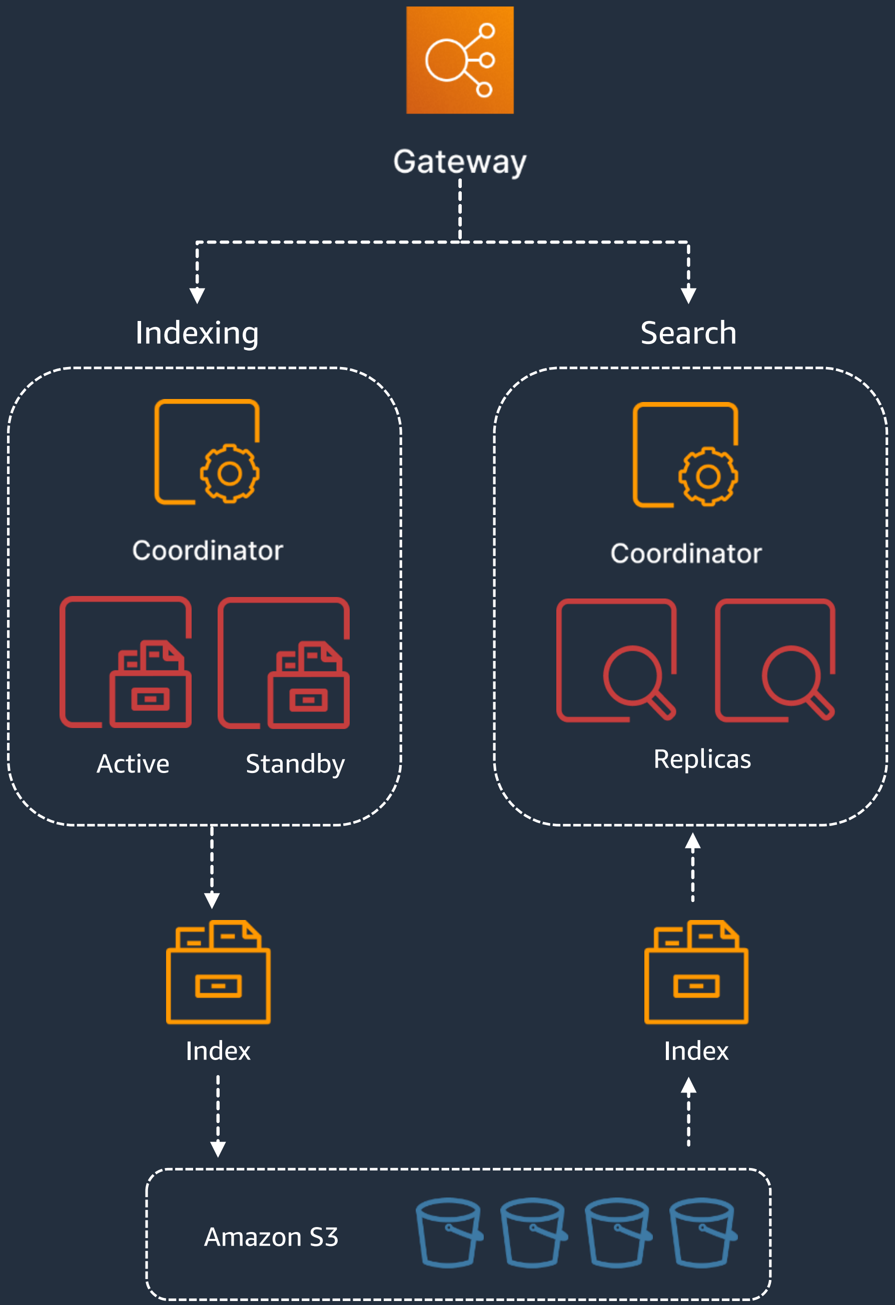
Select your use case

● Time series
Use for analyzing large volumes of semi-structured, machine-generated data in real time.

○ Search
Use for full-text searches that power applications within your network.

Technical innovations

- Storage and compute decoupled
- Separate indexing and search pipelines
- Built-in hot-warm tier
- Active-standby data nodes
- Serverless Dashboards



Reduce cost and complexity

Cost reduction for workloads with

- Batch indexing or search patterns
- Spiky or unpredictable demand patterns
- Large volumes of data

Reduced complexity

- No dealing with shard sizing and counts
- No sizing and provisioning capacity

Pricing: managed cluster vs. serverless

Managed Domain

🟦🟦

Configure your instances

🟦🟦🟦🟦

+ Add instances for high availability

🟦🟦🟦🟦🟦🟦

+ Add instances for UltraWarm

🟦🟦🟦🟦🟦🟦🟦🟦

+ Add some buffer for peak workloads

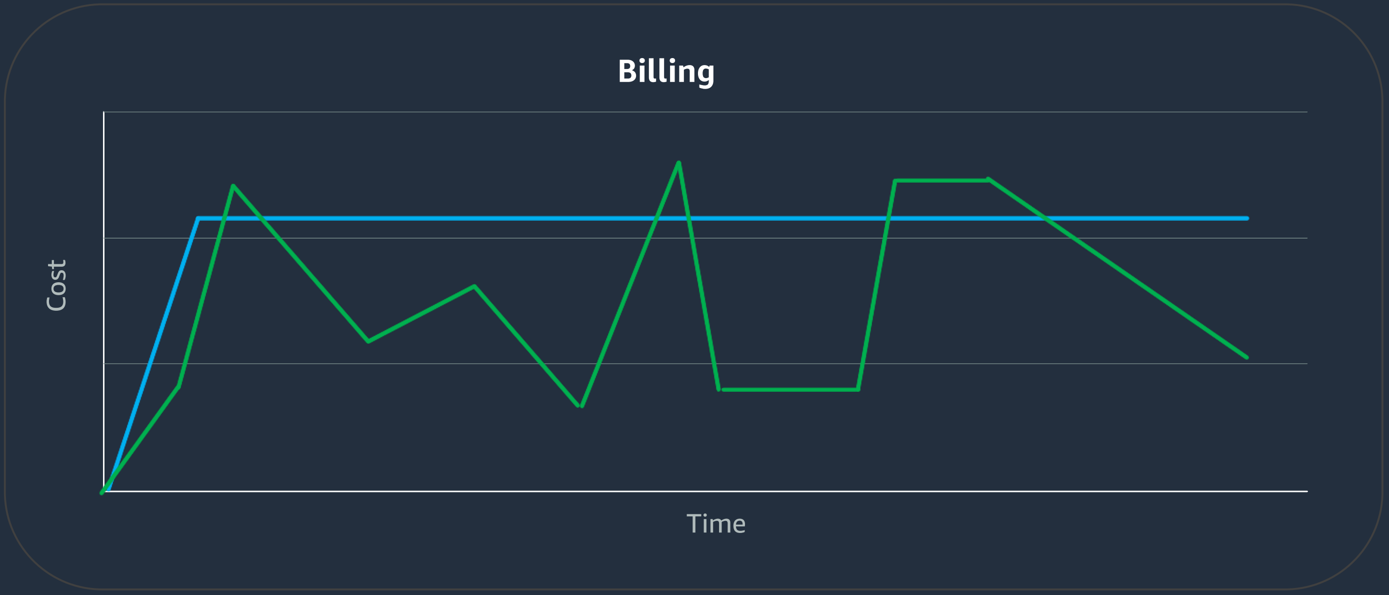
Serverless

🟩🟩🟩🟩🟩🟩

OCU

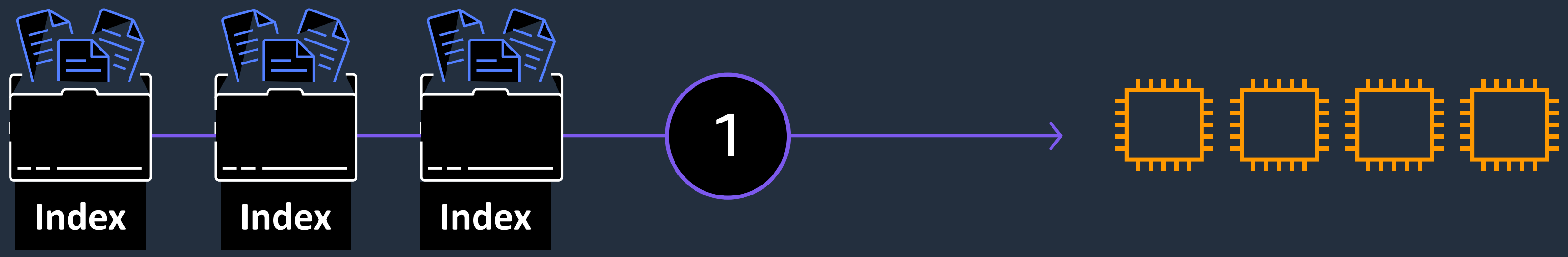
🟩🟩🟩🟩🟩🟩

Storage



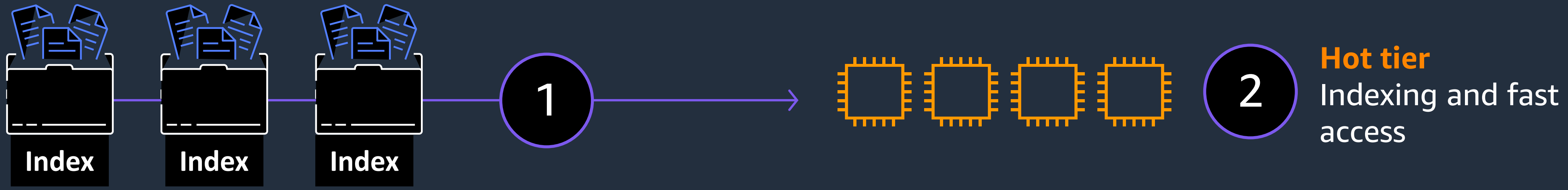
Storage tiering (for time series data)

Data lifecycle (for time-series data)



1 Send data to Amazon OpenSearch Service and use Index State Management (ISM) to automate index migrations and deletions

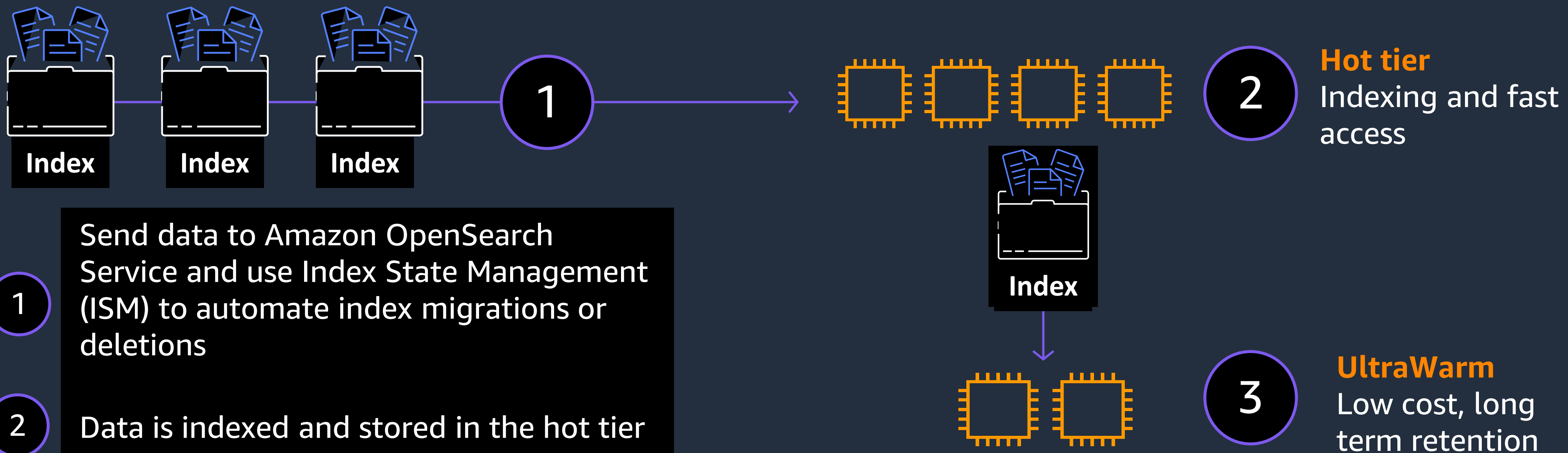
Data lifecycle (for time-series data)



1 Send data to Amazon OpenSearch Service and use Index State Management (ISM) to automate index migrations or deletions

2 Data is indexed and stored in the hot tier

Data lifecycle (for time-series data)

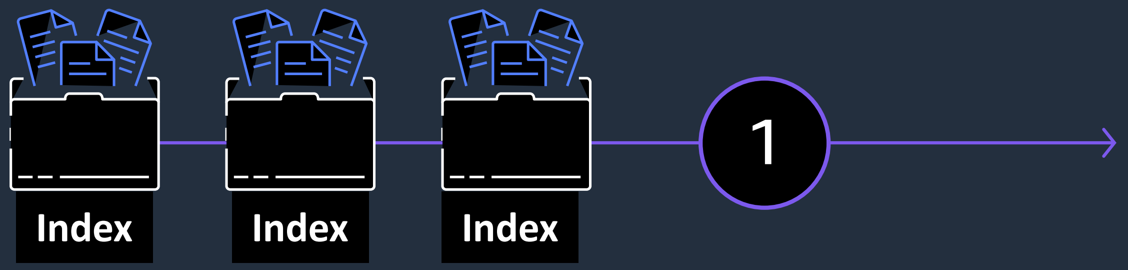


1 Send data to Amazon OpenSearch Service and use Index State Management (ISM) to automate index migrations or deletions

2 Data is indexed and stored in the hot tier

3 Migrate the index to UltraWarm storage for long-term, low cost storage

Data lifecycle (for time-series data)

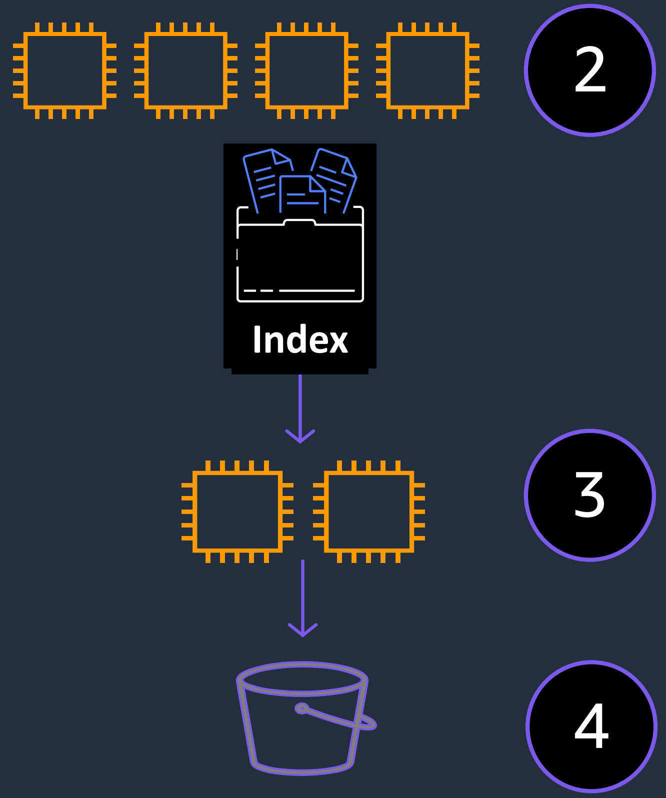


1. Send data to Amazon OpenSearch Service and use Index State Management (ISM) to automate index migrations or deletions

2. Data is indexed and stored in the hot tier

3. Migrate the index to UltraWarm storage for long-term, low cost storage

4. Store data in Cold Storage for longer-term, lowest cost storage

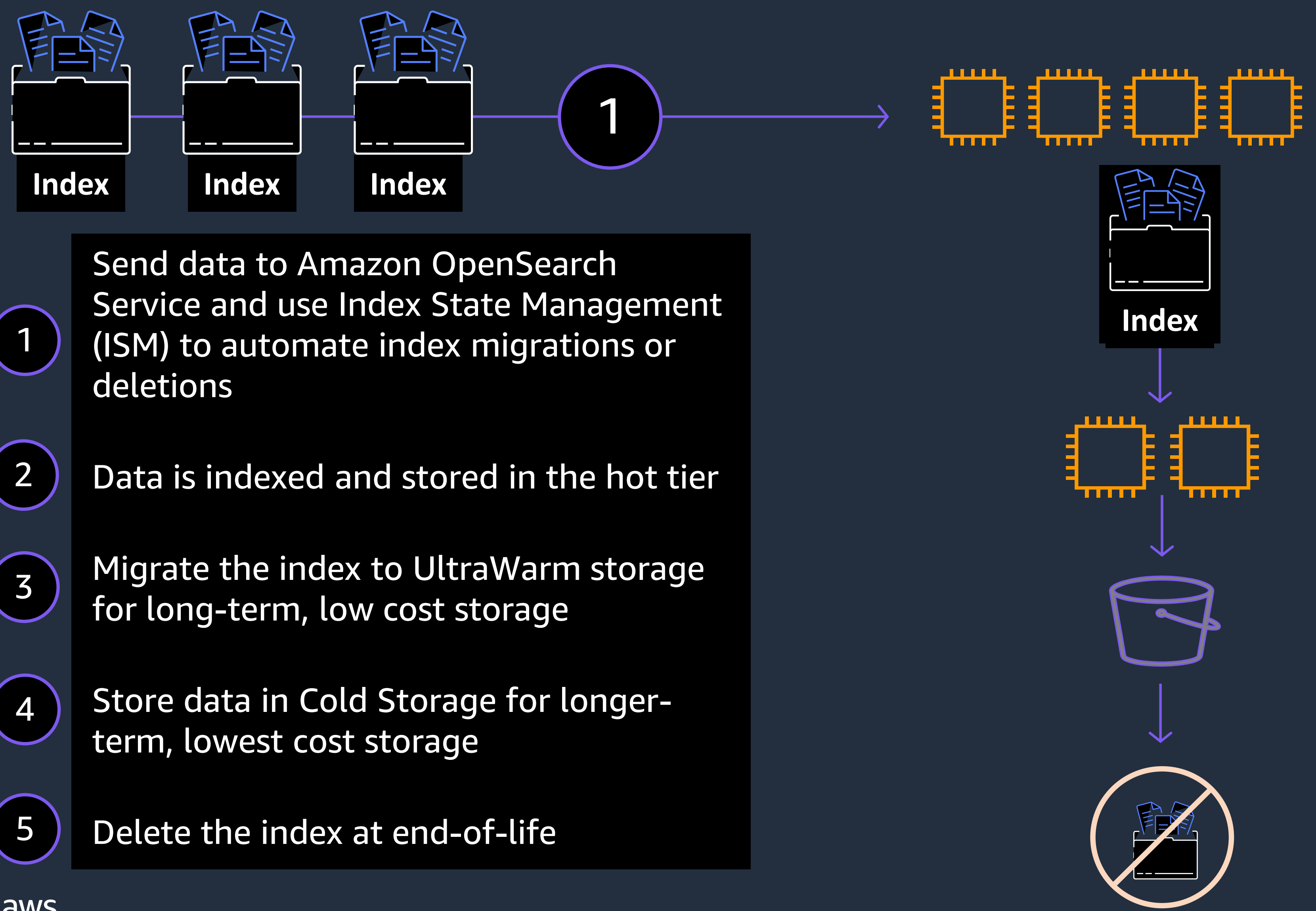


Hot tier
Indexing and fast access

UltraWarm
Low cost, long term retention

Cold Storage
Lowest cost, longer term retention and on-demand access

Data lifecycle (for time-series data)



2 **Hot tier**
Indexing and fast access

3 **UltraWarm**
Low cost, long term retention

4 **Cold Storage**
Lowest cost, longer term retention and on-demand access

5

1 Send data to Amazon OpenSearch Service and use Index State Management (ISM) to automate index migrations or deletions

2 Data is indexed and stored in the hot tier

3 Migrate the index to UltraWarm storage for long-term, low cost storage

4 Store data in Cold Storage for longer-term, lowest cost storage

5 Delete the index at end-of-life

Thank you!

Gene Alpert (he/him)

Sr. Analytics Specialist

Amazon Web Services