

# Sberbank Elastic Community

## Index/Data lifecycle Management

---

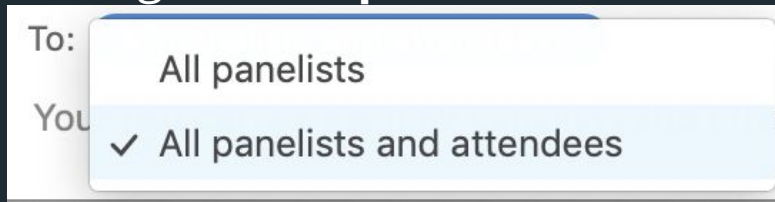
Tatjana Frank, Solution Architect

April 2020



# Housekeeping & Logistics

- Attendees are automatically muted when joining Zoom
- Q+A will be at the end of the webinar
- Ask questions for us in the Zoom chat during the webinar
  - Chat settings **To: All panelists and attendees**



- Ask more questions on our discuss forum: **discuss.elastic.co**
- **Recording** will be available after the webinar and emailed to all registrants

# Agenda

- Index/Data Management in Elastic Stack
  - The Big Picture around Data Management
  - Data Lifecycle Phases
  - Data Lifecycle Features
- Demo ILM



**Solutions**



**Visualize & Manage**



**Store, Search, & Analyze**



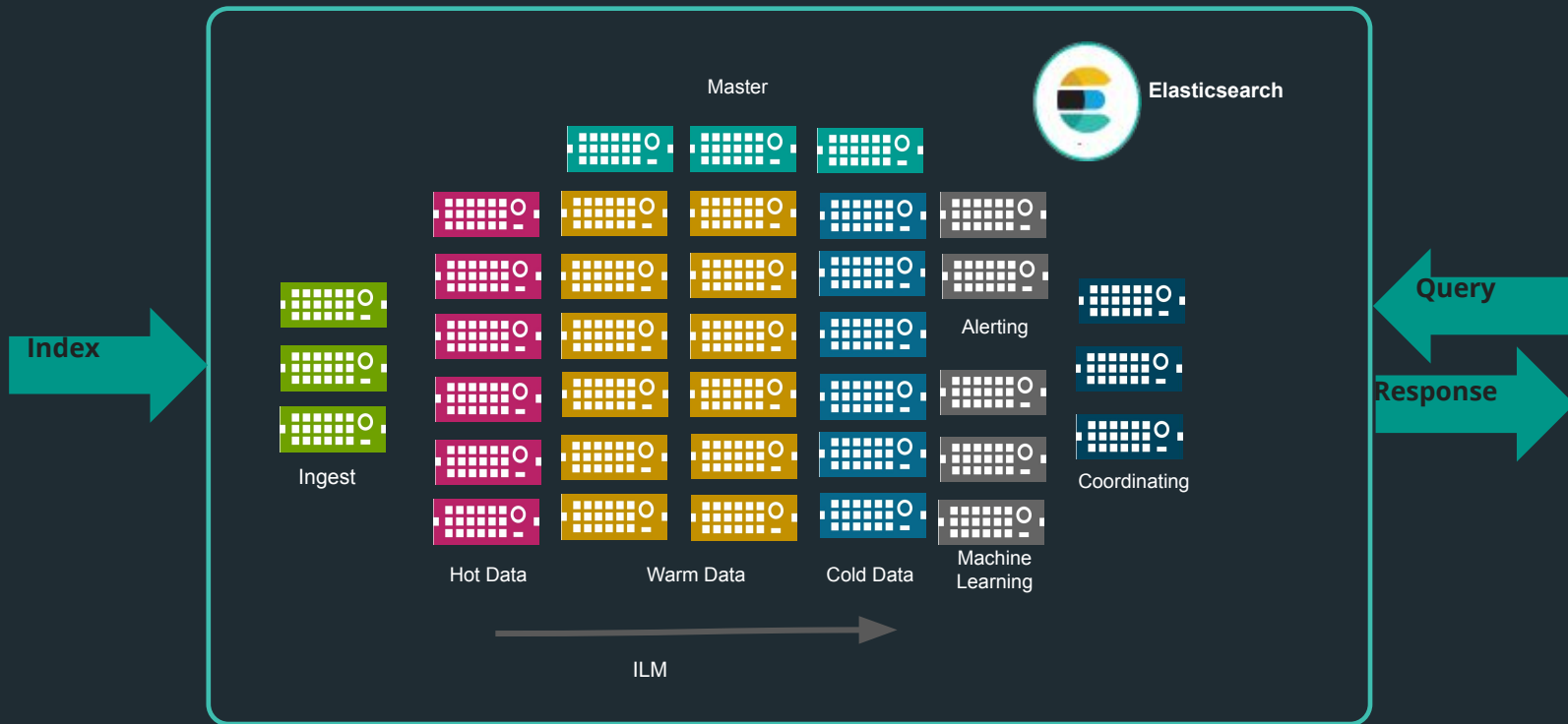
**Ingest**



**Deployment**

# Inside a Large Elasticsearch Logging Cluster

*Reduce infrastructure costs, isolate workloads, and manage data lifecycle*



# Dive Deep into Elasticsearch

## Shards & replicas

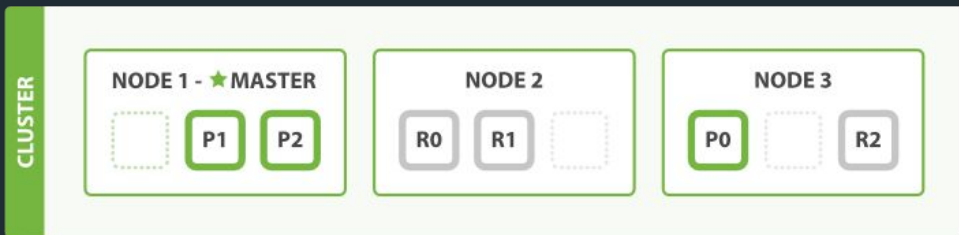


### Shards

- Start with 1 primary shard per index (default starting 7.0)
- How many per node?
  - Max 20 Shards per GB of JVM Heap
  - 30 GB Heap = MAXIMUM 600 Shards
- Add more to scale for ingest volume
- Shard allocation and cluster-level routing:  
settings to control where, when, and how shards are allocated to nodes

### Replicas

- Keep in mind more replicas = slower writes
- Only add more replicas if your use case is search heavy



## Hot

In this phase, you are actively querying and writing to your index.

## Warm

You are still querying your index, but it is read-only. You can allocate shards to less performant hardware. For faster searches, you can reduce the number of shards and force merge segments.

## Cold

You are querying your index less frequently, so you can allocate shards on significantly less performant hardware. Because your queries are slower, you can reduce the number of replicas.

---

## Frozen

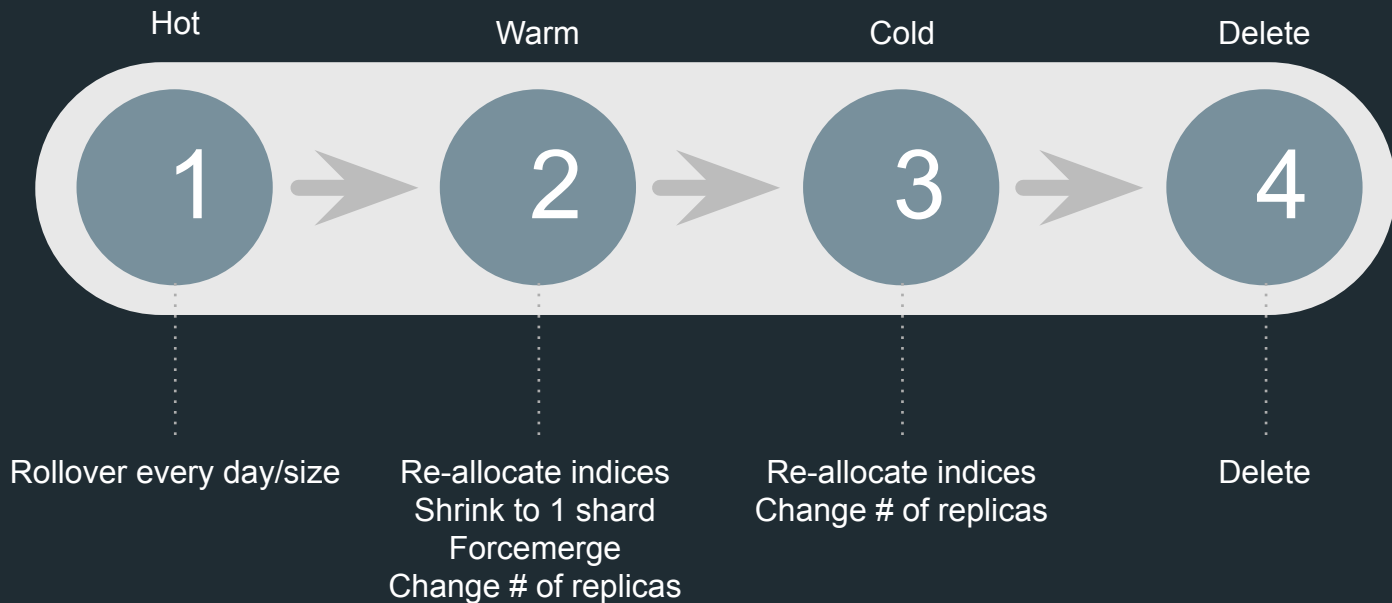
A frozen index has little overhead on the cluster and is blocked for write operations. You can search a frozen index, but expect queries to be slower.

## Backup

Using Snapshot and Restore API, the data will be moved out of an Elasticsearch cluster and archived on a defined storage. It can be restored back into an Elasticsearch cluster.

# Lifecycle

Time-based indices

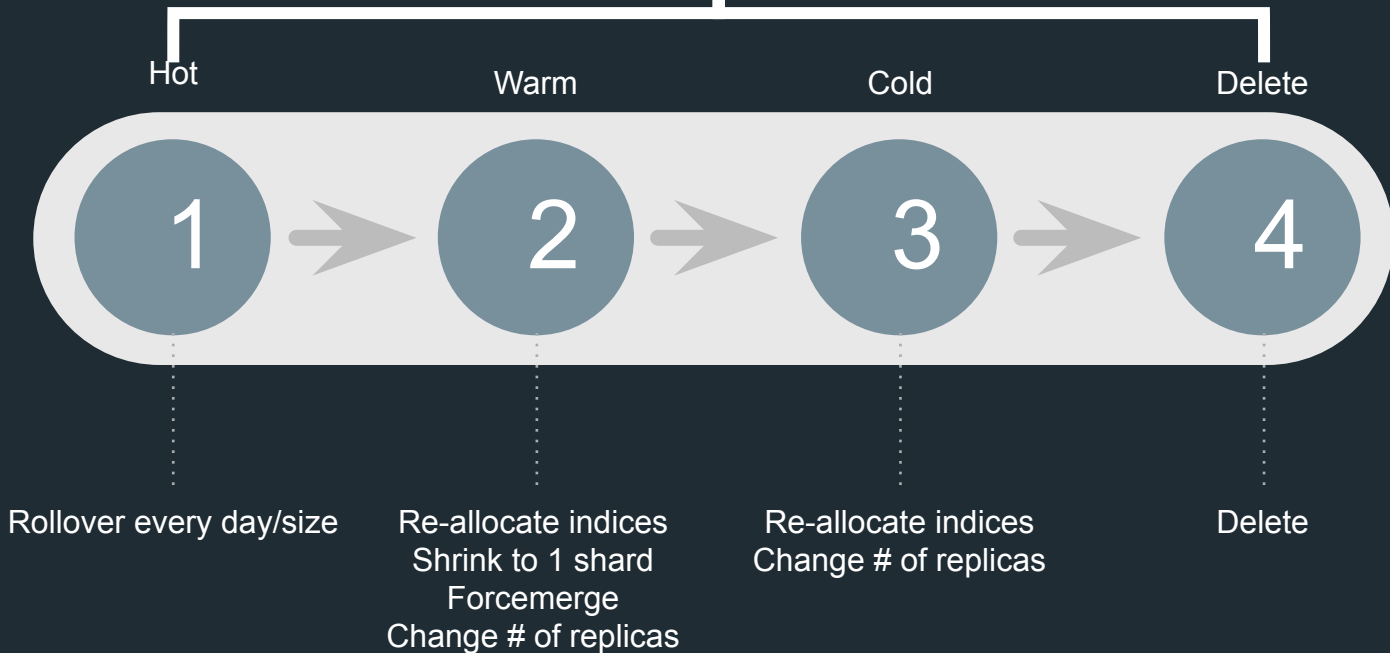




# Lifecycle

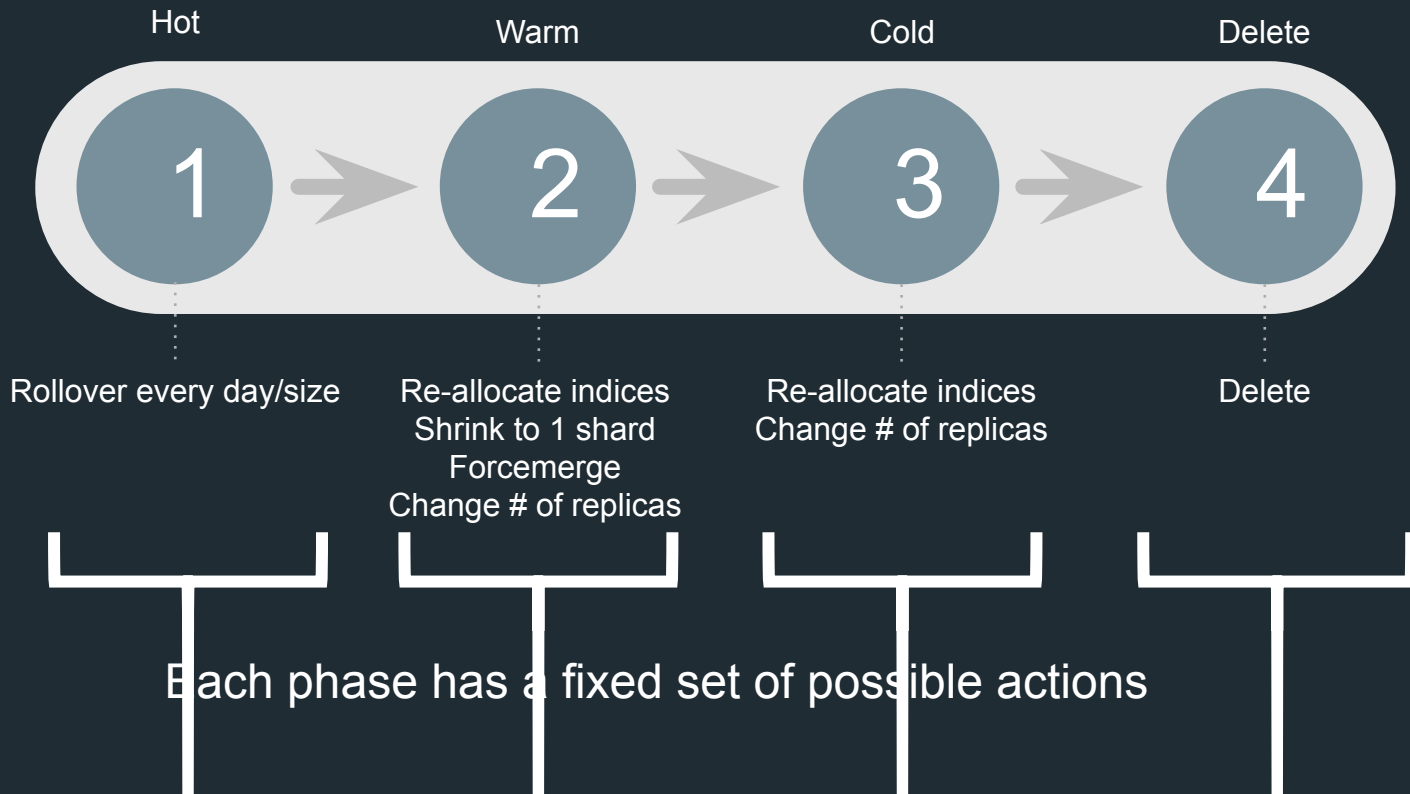
Time-based indices

Fixed set of phases



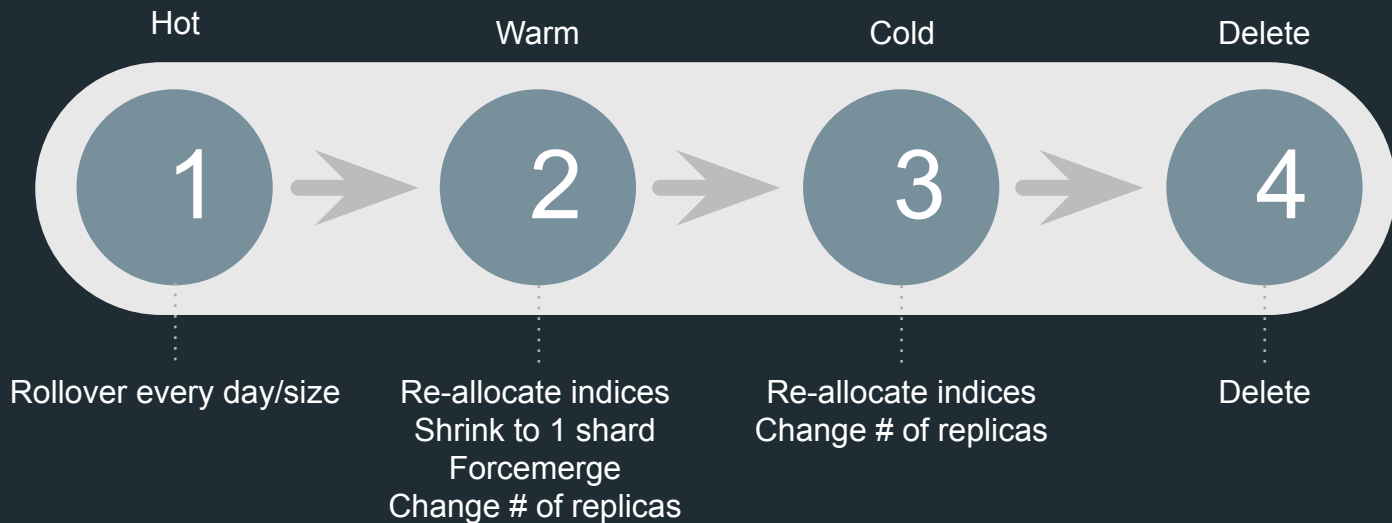
# Lifecycle

Time-based indices



# Lifecycle

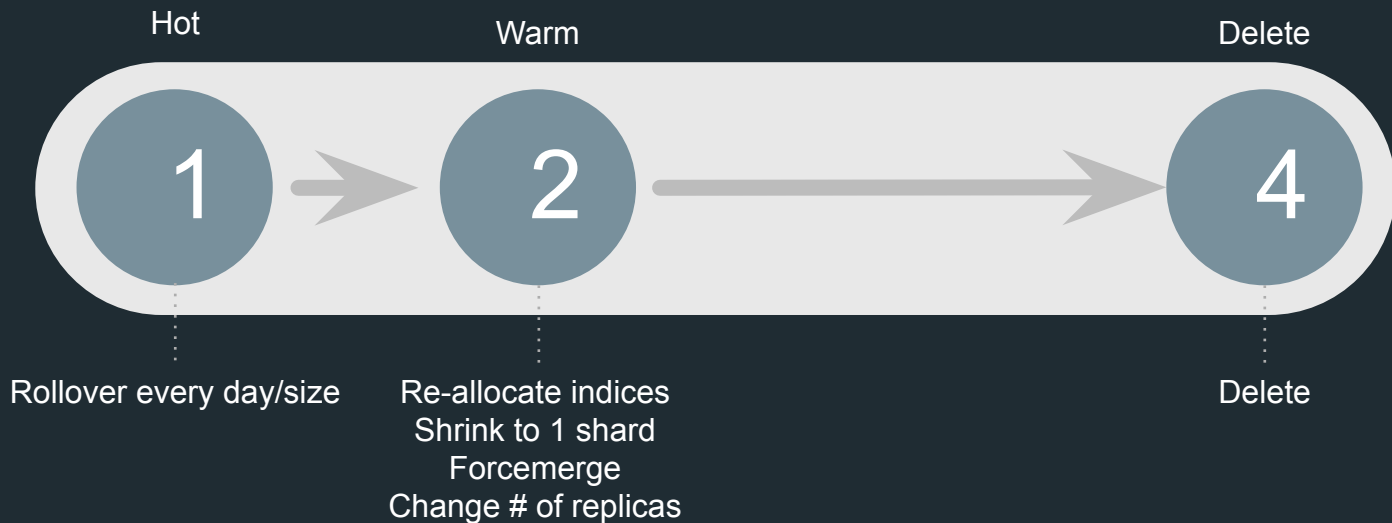
Time-based indices



**Sensible defaults driven by UI**

# Lifecycle

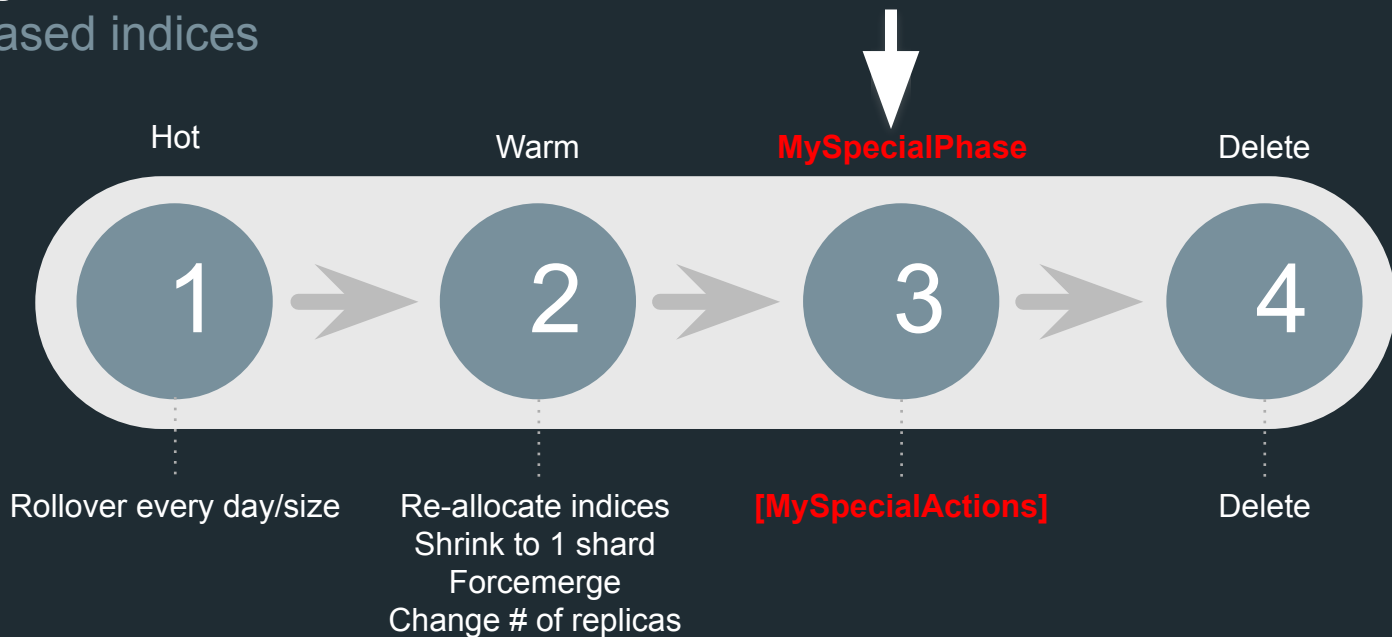
Time-based indices



**This is allowed**

# Lifecycle

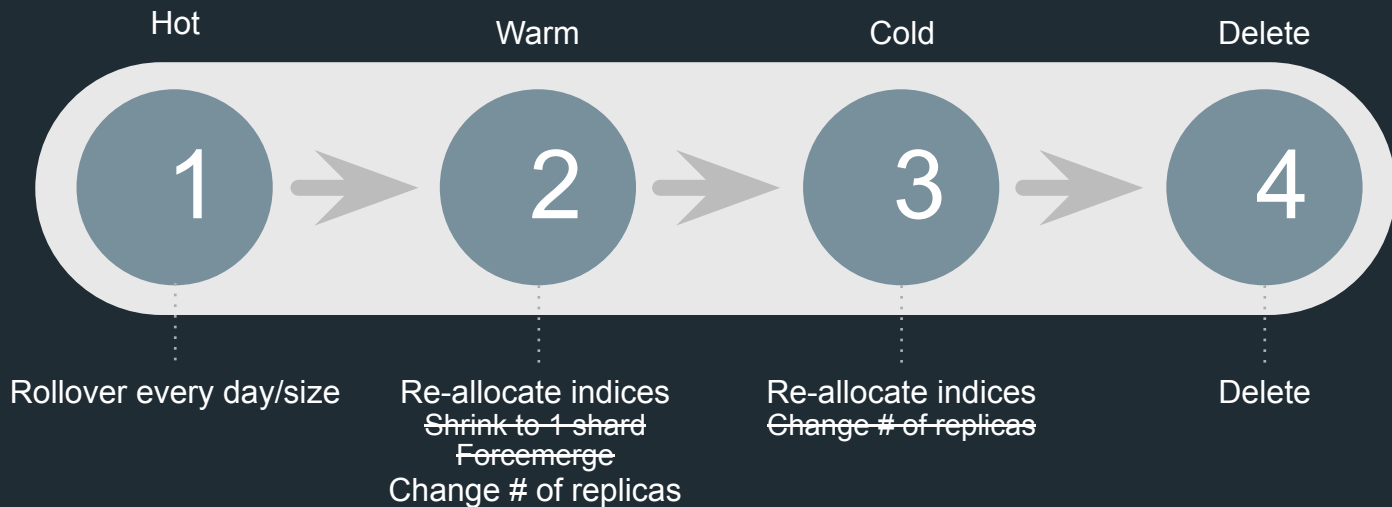
Time-based indices



**This is not allowed**

# Lifecycle

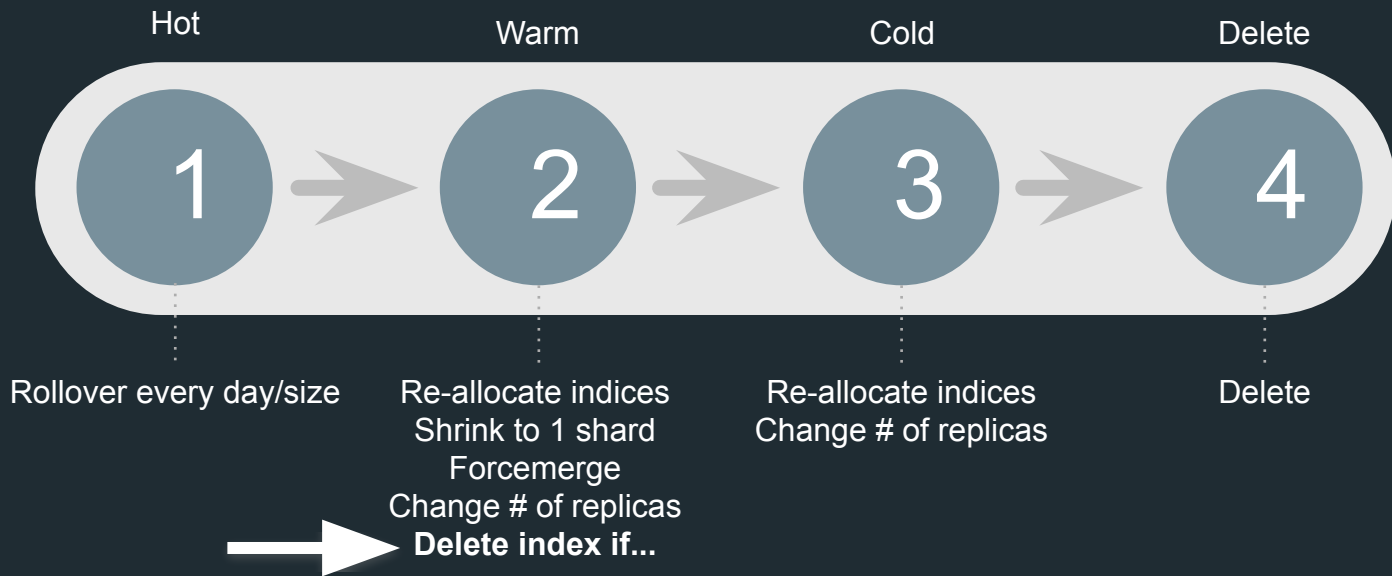
Time-based indices



**This is allowed**

# Lifecycle

Time-based indices



**This is not allowed**

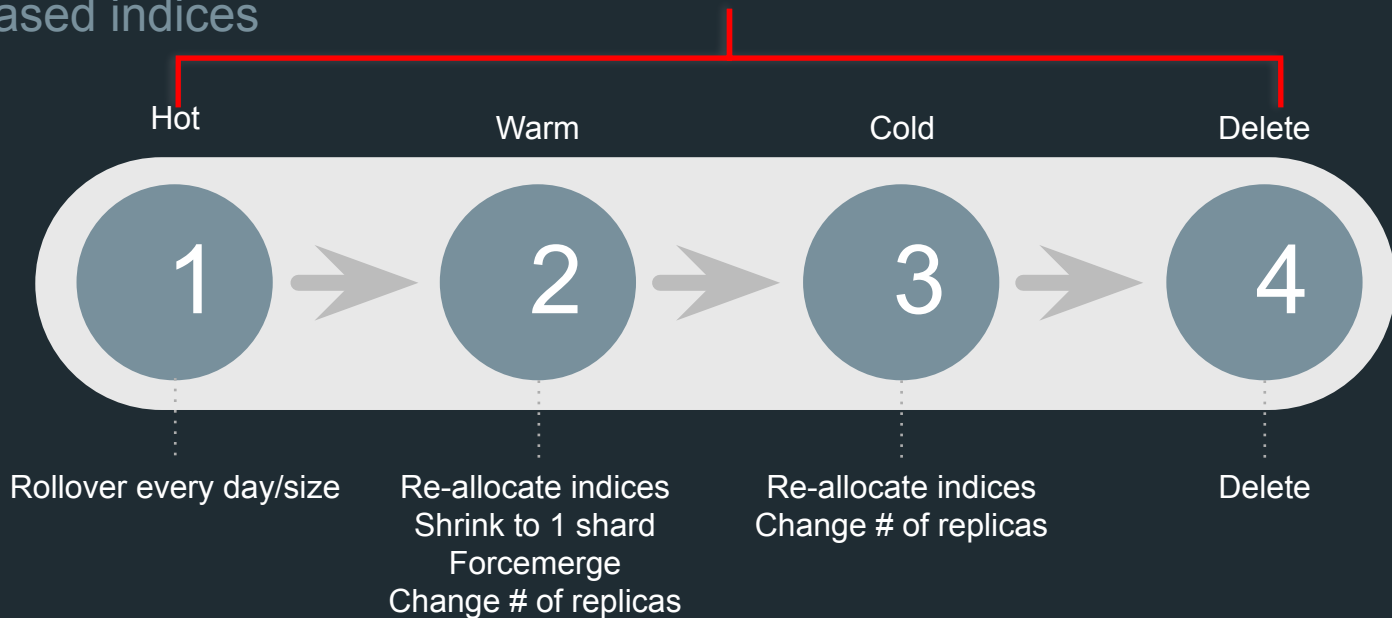
**Questions to ask...**



# Lifecycle

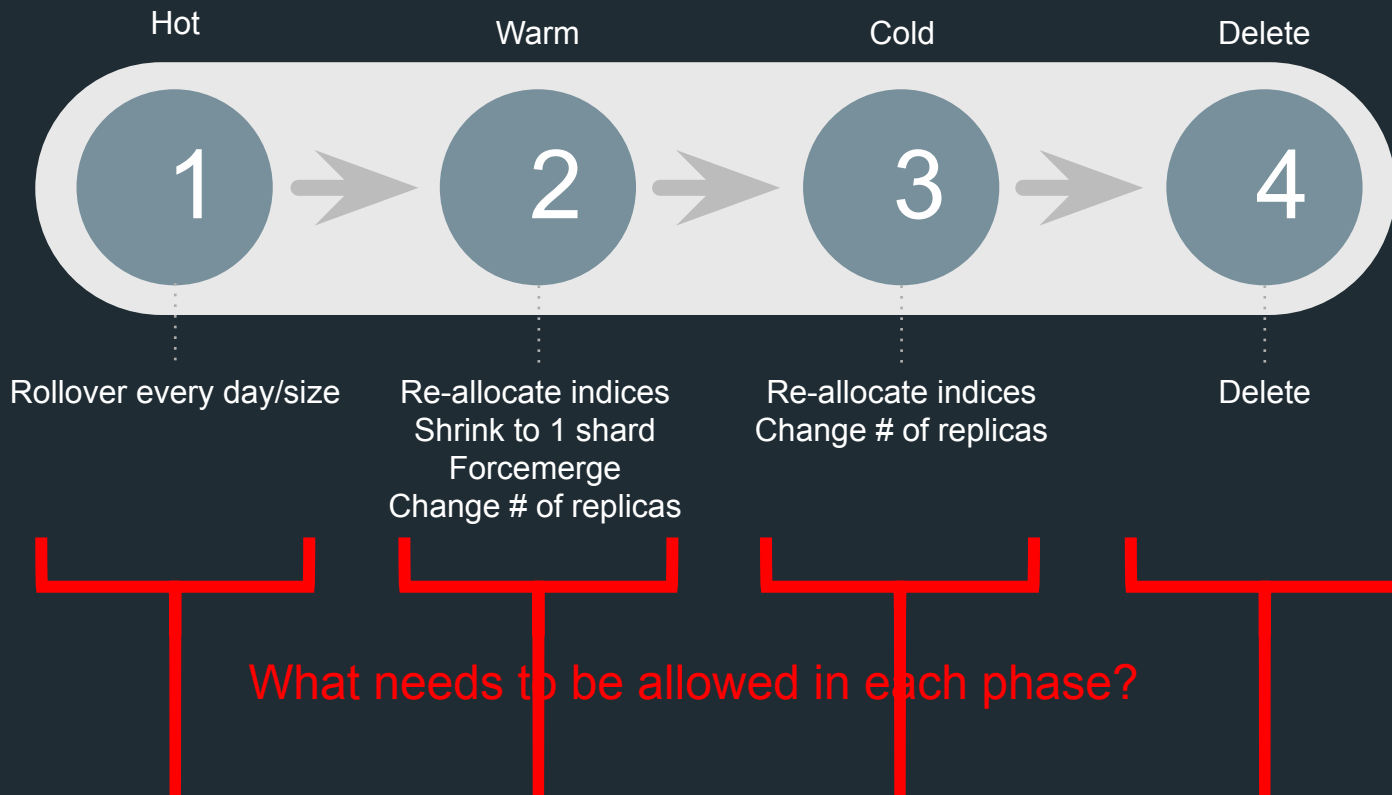
Time-based indices

Are these the right phases?



# Lifecycle

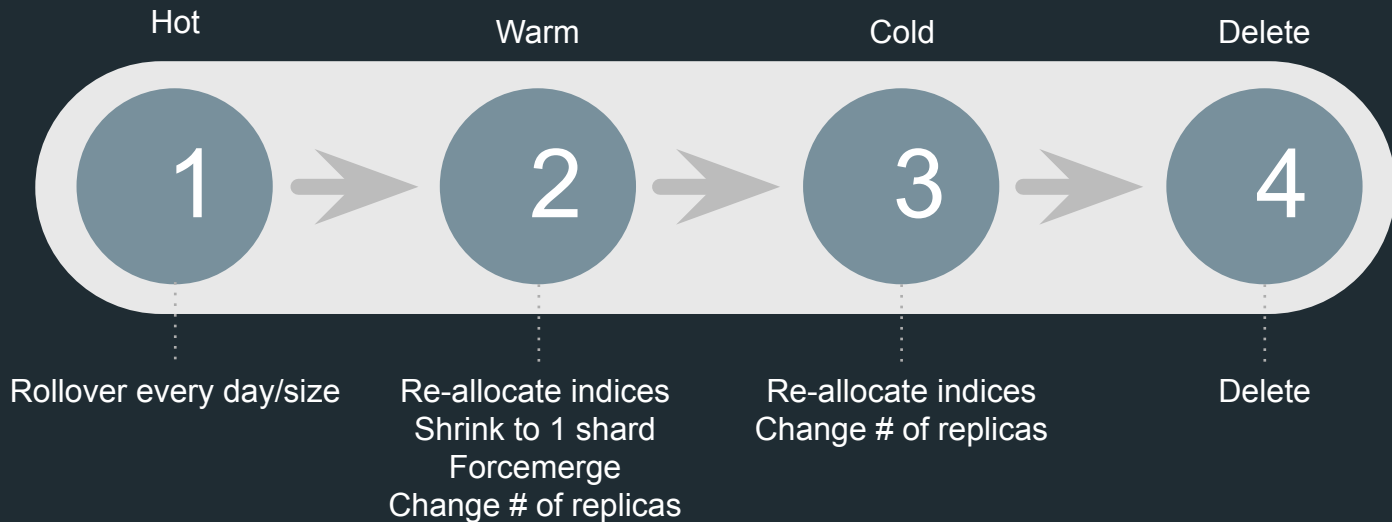
Time-based indices



# Lifecycle

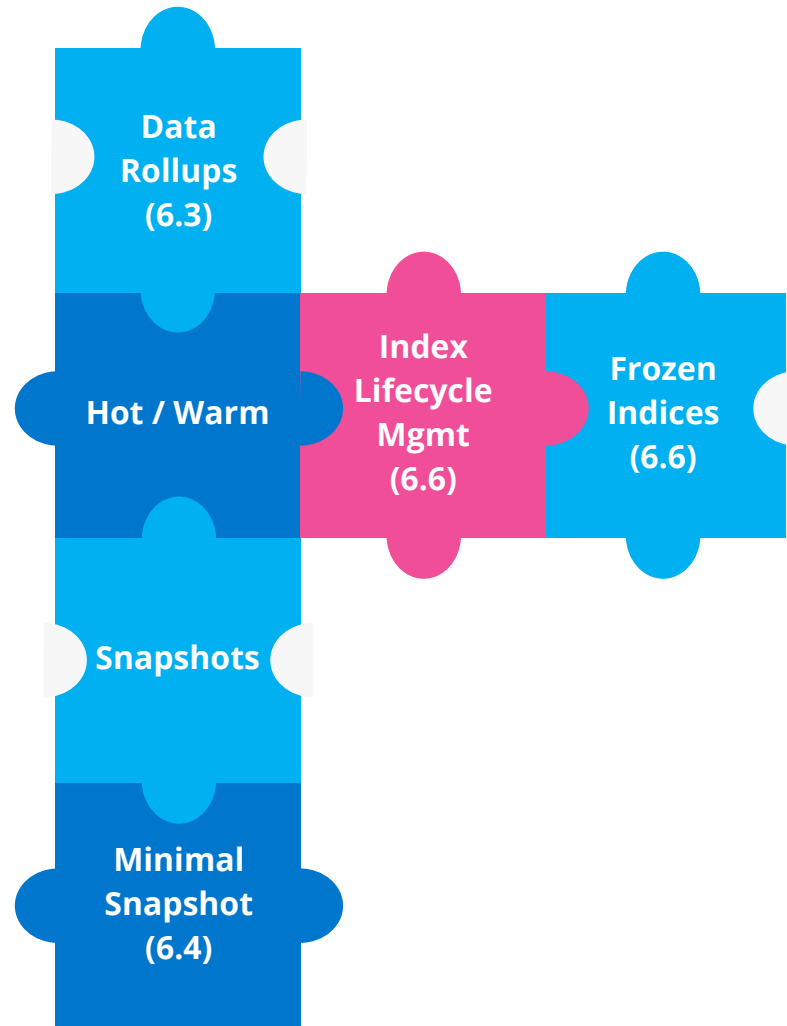
Time-based indices

What other life cycles should be considered?  
What do/could they look like?



# Index Lifecycle Management

Part of larger story around data management. ILM helps to **automate** the lifecycle of data without the need to create own tooling.



# Index Mgmt Improvements

GA | Basic (free)

Badges to mark frozen, rolled-up and follower indices

Freeze / Unfreeze actions

## Index management

Update your Elasticsearch indices individually or in bulk.



Include rollout indices



Include system indices

Search

Reload indices

<input type="checkbox"/>	Name	Health	Status	Primaries	Replicas	Docs count	Storage size
<input type="checkbox"/>	kibana_sample_data_ecommerce Frozen	green	open	1	0	4675	4.8mb
<input type="checkbox"/>	twitter Follower	yellow	open				
<input type="checkbox"/>	kibana_sample_data_flights	green	open				
<input type="checkbox"/>	rolledup_kibana_sample Rollup	yellow	open				
<input type="checkbox"/>	kibana_sample_data_logs	green	open				

Rows per page: 10

### INDEX OPTIONS

Close index

Force merge index

Refresh index

Clear index cache

Flush index

Freeze index

Delete index

Add lifecycle policy

Manage

# Rollups in Kibana

Beta | Basic (free)

Automatically roll up data into coarser time buckets as it ages

- Save on storage space & costs
- Smaller indices = faster analytics

## 6.3 - Rollups API in Elasticsearch

## 6.5 - Rollups support in Kibana

- Job management UI
- Visualize rolled up indices

Aggregation functions:

- Avg, min, max, sum, count

Rollup jobs / Create

## Create rollup job



### Metrics (optional)

Select the metrics to collect while rolling up data. By default, only doc\_counts are collected for each group.

Q Search

#### Field

bytes	<input checked="" type="checkbox"/> Average	<input type="checkbox"/> Maximum	<input type="checkbox"/> Minimum	<input checked="" type="checkbox"/> Sum
machine.ram	<input checked="" type="checkbox"/> Average	<input type="checkbox"/> Maximum	<input type="checkbox"/> Minimum	<input checked="" type="checkbox"/> Sum
memory	<input checked="" type="checkbox"/> Average	<input type="checkbox"/> Maximum	<input type="checkbox"/> Minimum	<input checked="" type="checkbox"/> Sum
phpmemory	<input checked="" type="checkbox"/> Average	<input type="checkbox"/> Maximum	<input type="checkbox"/> Minimum	<input checked="" type="checkbox"/> Sum

Rows per page: 200 ▾

< [Back](#)

Next >

# Frozen Indices

Basic (free)

Enable higher storage: memory ratio

Trades off search speeds for lower memory footprint (i.e. lower costs)

Keep data searchable (online) in an cost-efficient way

Operationally much simpler than alternatives like snapshots or archival.

## Open Index

Searchable  
High heap (memory)  
Fast searches

## Frozen Index

Searchable  
No heap (memory)  
Slower searches

## Closed Index

NOT searchable  
No heap (memory)

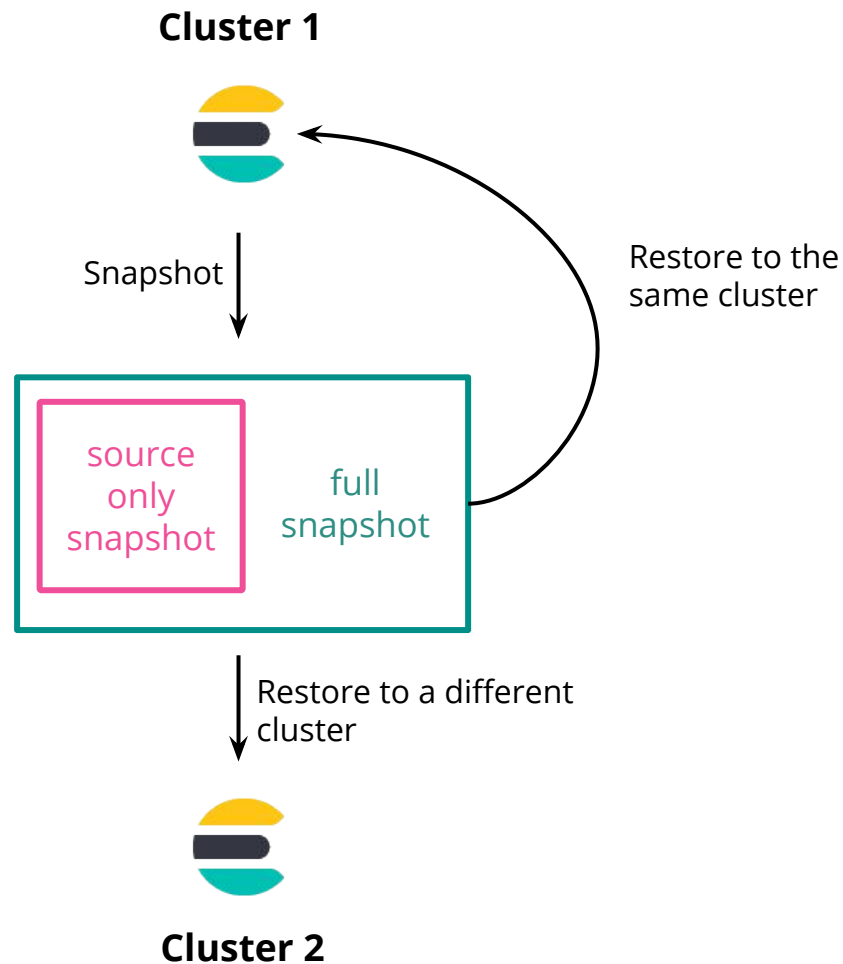
# \_source Only Snapshot

Basic (free)

\_source only snapshots can be 50% smaller than full snapshots

Requires a reindex to make the data searchable again

Trades off restore time for smaller storage space / costs





# Using origination\_date

What if there's a gap between the index age and the data age?

1) data transition from other systems of records into Elasticsearch at various points in the data's lifecycle

2) data already in Elasticsearch is reindexed into new indexes

```
PUT /events-2020.01.01
{
  "settings" : {
    "index" : {
      "lifecycle.name": "readonly_and_delete_policy",
      "lifecycle.parse_origination_date": true # <1>
    }
  }
}
```

```
PUT /events
{
  "settings" : {
    "index" : {
      "lifecycle.name": "readonly_and_delete_policy", # <1>
      "lifecycle.Origination_date": 1577836800000 # <2>
    }
  }
}
```

# Demo ILM

# Thank You

---

