

AWS  
re:Invent

**D A T 3 2 5**

# Amazon DynamoDB: Under the hood of a hyperscale database

**James Christopher Sorenson III aka jaso**

Senior Principal Engineer  
Amazon Web Services

# Areas for discussion

Overview GetItem / PutItem

Partitioning

Transactions

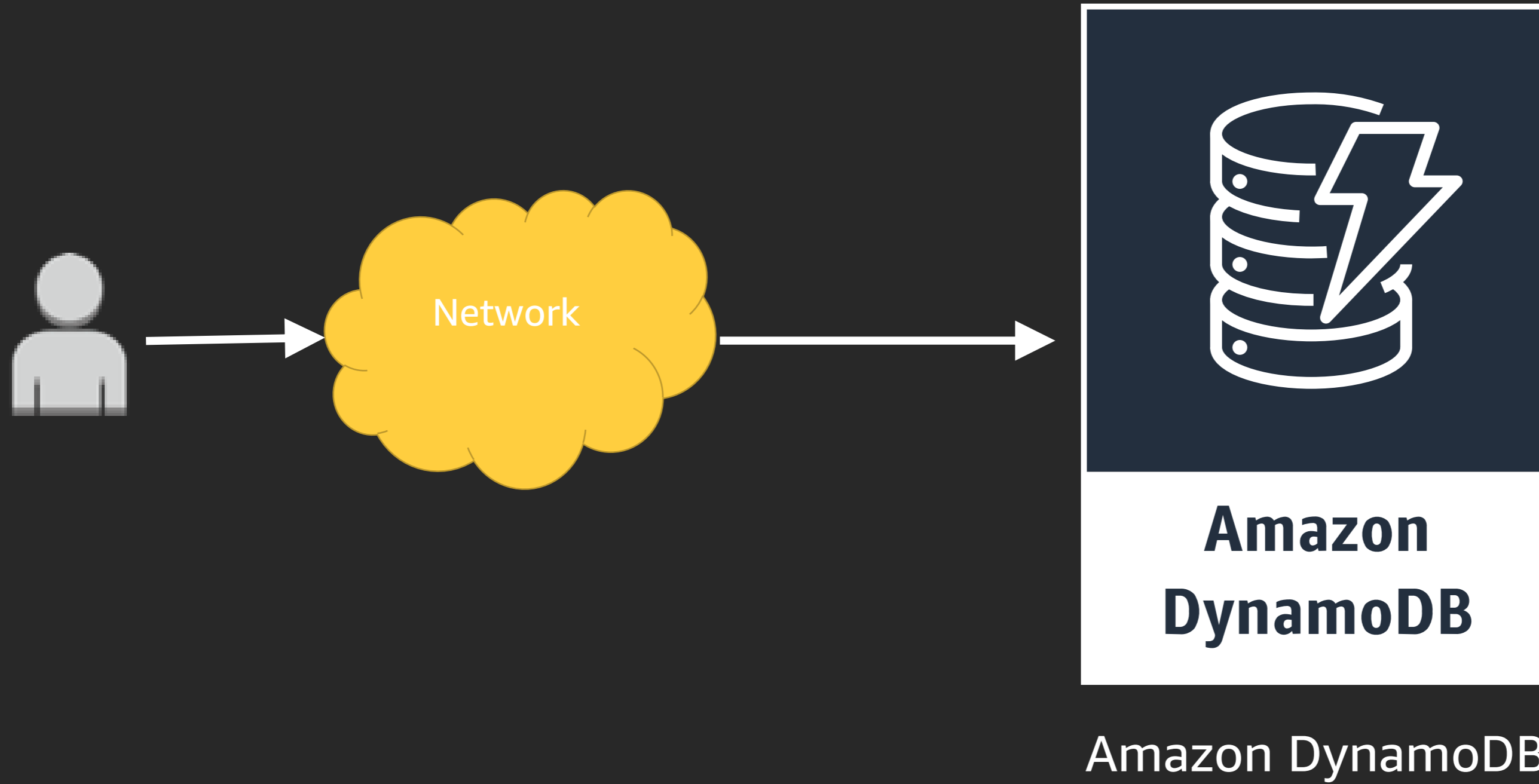
Global Tables

Autoscaling

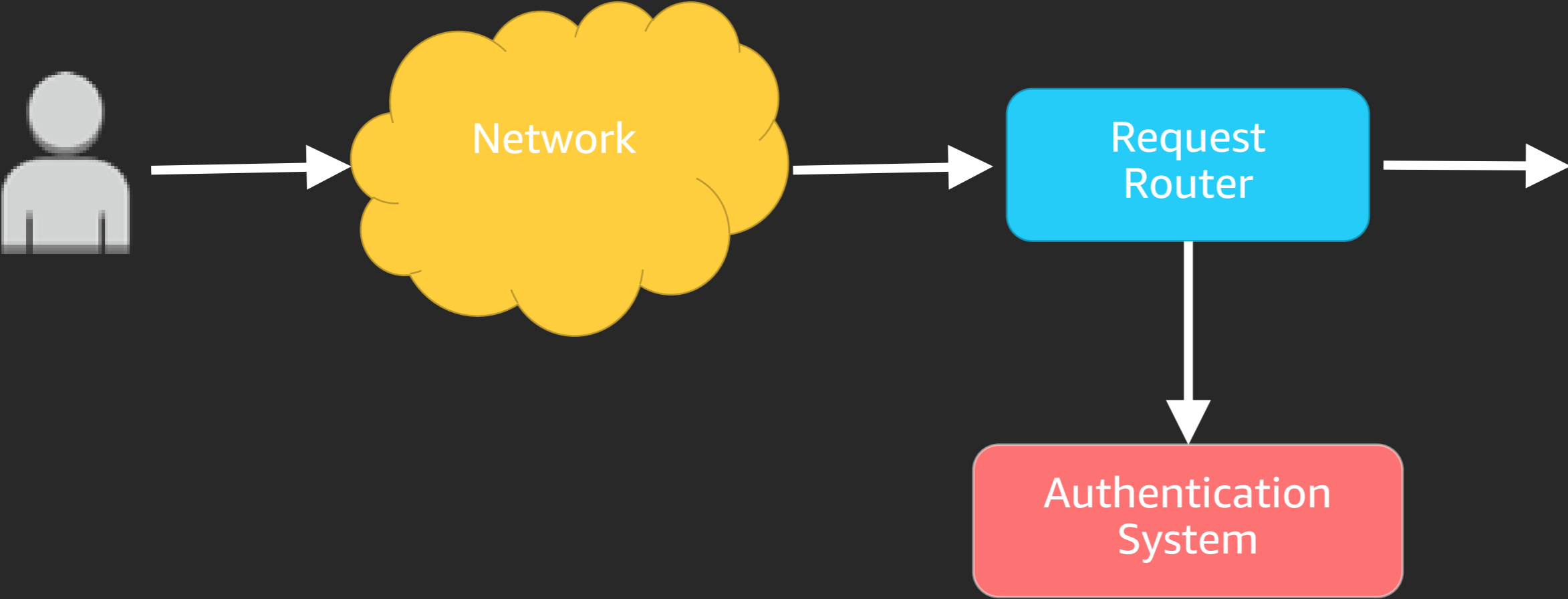
Backup Restore

# GetItem / PutItem

# GetItem



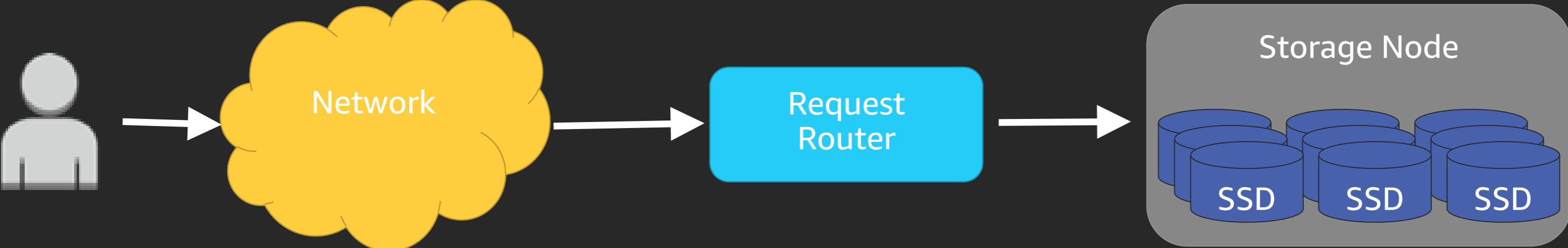
# GetItem (Step 1)



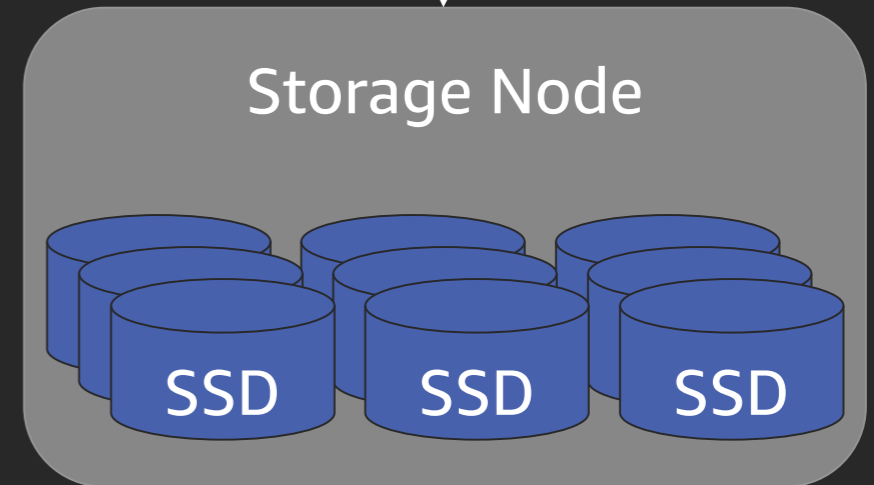
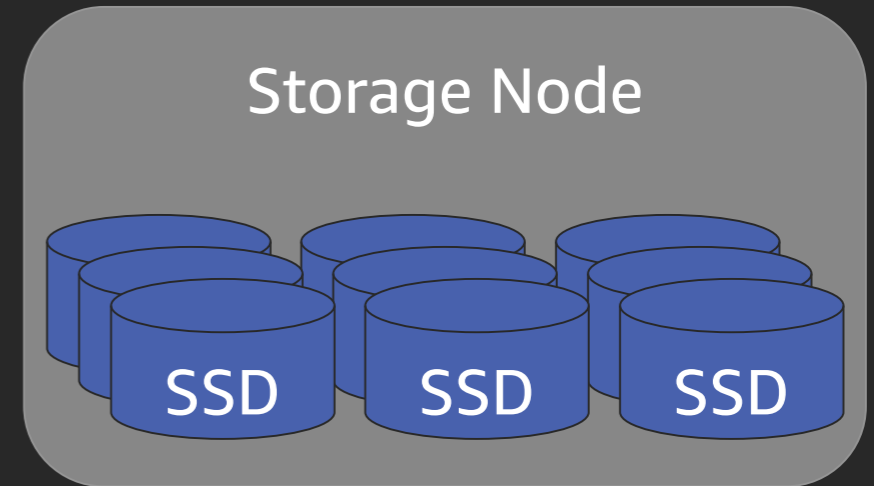
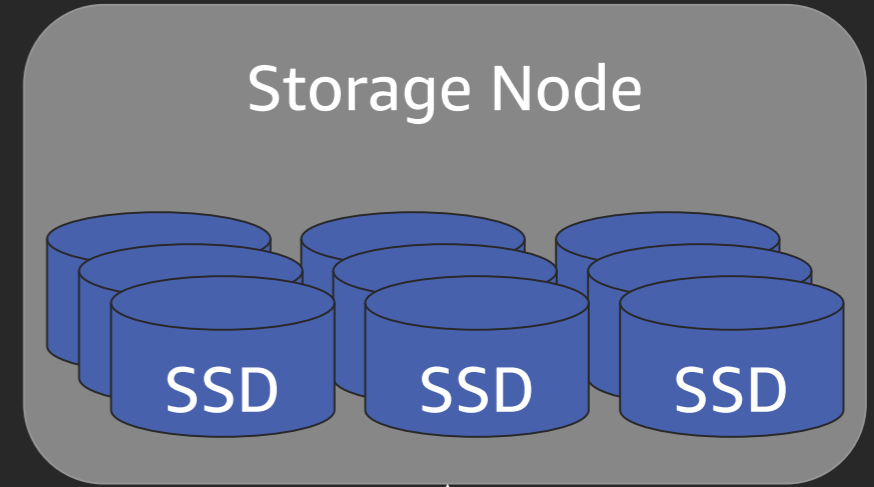
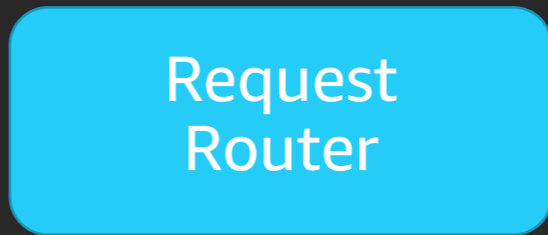
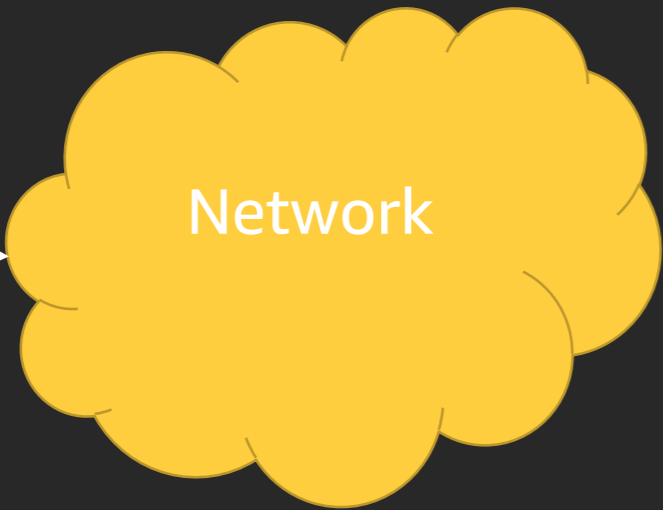
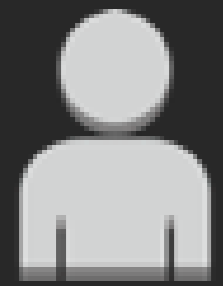
# Sample Policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetItem"
      ],
      "Resource": [
        "arn:aws:dynamodb:us-west-2:1234567890:table/customer"
      ],
      "Condition": {
        "ForAllValues:StringEquals": {
          "dynamodb:LeadingKeys": [
            "${www.amazon.com:user_id}"
          ]
        }
      }
    }
  ]
}
```

# GetItem (Step 2)



# PutItem



# DynamoDB evolved from Dynamo

## **Dynamo: Amazon's Highly Available Key-value Store**

Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati,  
Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall  
and Werner Vogels

Amazon.com

# Paxos

---

## The Part-Time Parliament

---

Leslie Lamport

---

September 1, 1989

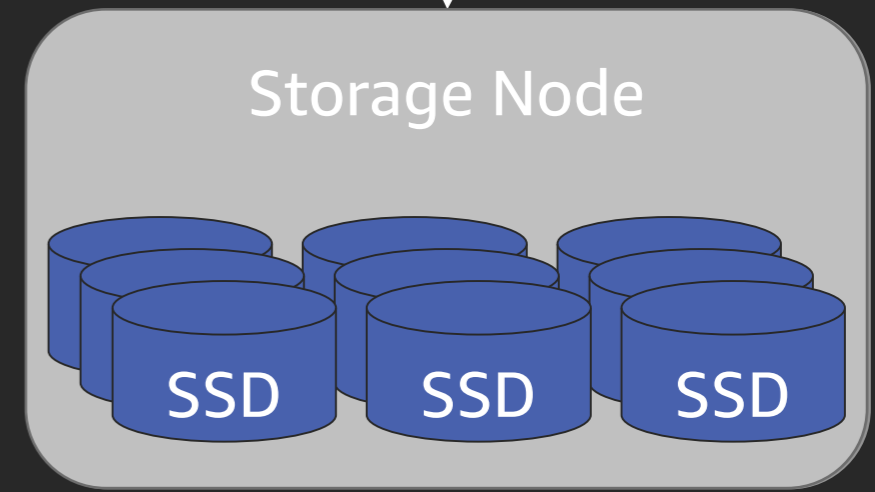
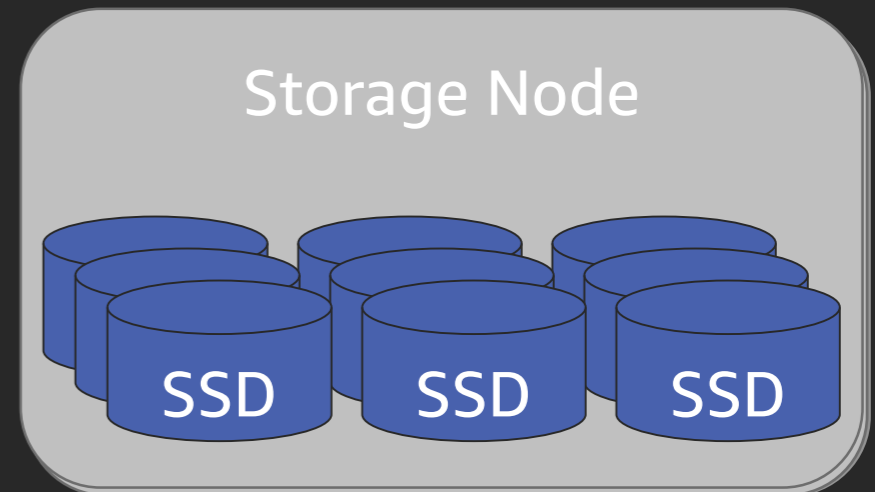
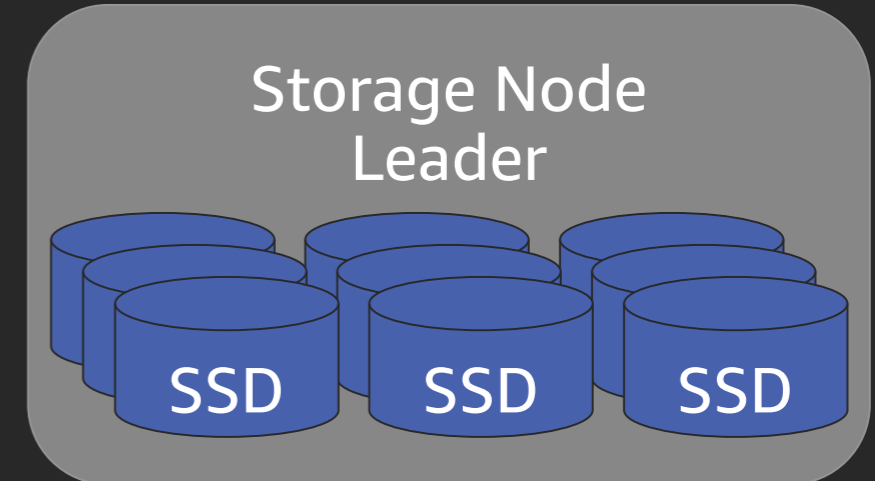
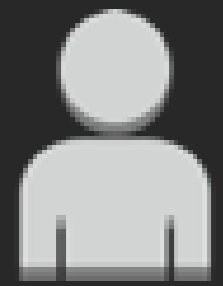
---

## **Paxos Made Simple**

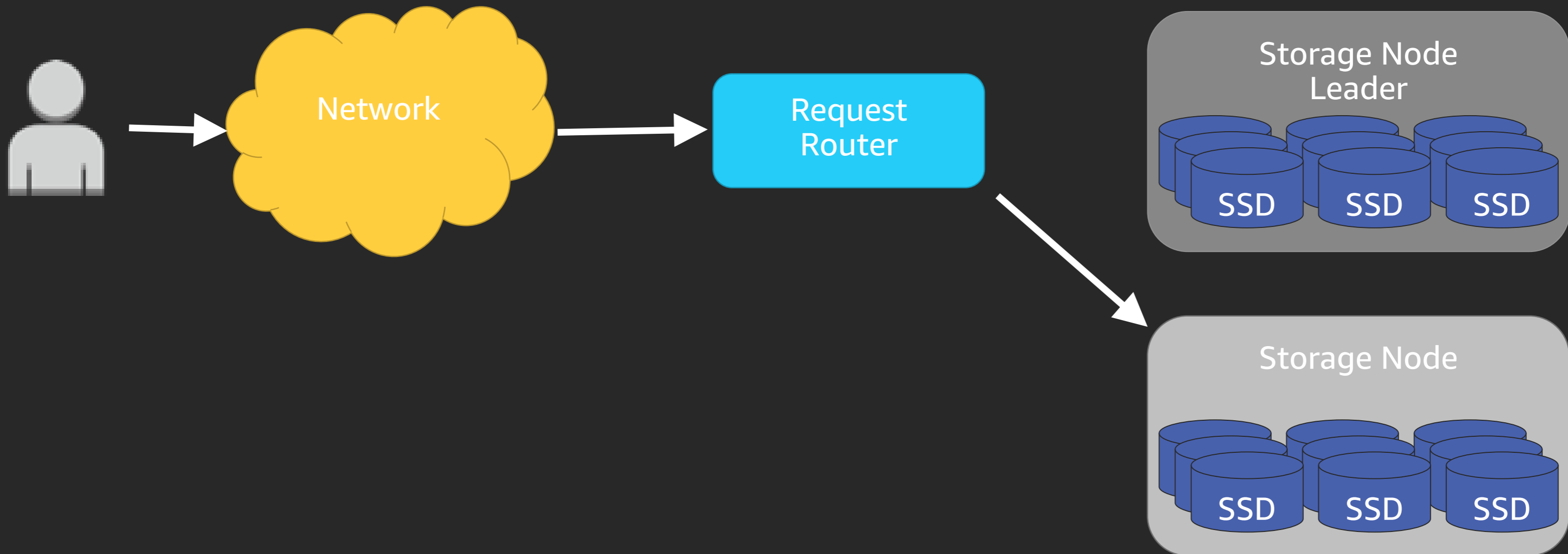
Leslie Lamport

01 Nov 2001

# PutItem



# Eventual Consistency GetItem



# Questions



# Partitioning

# Tables

CustID	Customer Information
145783	{ name:"Bob", city:"London", ...}
236294	{ name:"Sara", city:"Tampa", ...}
333363	{ name:"Betty", city:"Madison", ...}
445104	{ name:"James", city:"Miami", ...}
523422	{ name:"Alex", city:"London", ...}
643145	{ name:"Val", city:"Seattle", ...}
723342	{ name:"Jeff", city:"Toledo", ...}

# Hashing

Hash Value	CustID	Customer Information
0x9531	145783	{ name:"Bob", city:"London", ...}
0x12A8	236294	{ name:"Sara", city:"Tampa", ...}
0x6134	333363	{ name:"Betty", city:"Madison", ...}
0x3391	445104	{ name:"James", city:"Miami", ...}
0xF355	523422	{ name:"Alex", city:"London", ...}
0xB082	643145	{ name:"Val", city:"Seattle", ...}
0xEA8A	723342	{ name:"Jeff", city:"Toledo", ...}

# Partitioning

Hash Value	CustID	Customer Information
0x9531	145783	{ name:"Bob", city:"London", ...}
0x12A8	236294	{ name:"Sara", city:"Tampa", ...}
0x6134	333363	{ name:"Betty", city:"Madison", ...}
0x3391	445104	{ name:"James", city:"Miami", ...}
0xF355	523422	{ name:"Alex", city:"London", ...}
0xB082	643145	{ name:"Val", city:"Seattle", ...}
0xEA8A	723342	{ name:"Jeff", city:"Toledo", ...}

0x12A8	236294	{ name:"Sara", city:"Tampa", ...}
0x3391	445104	{ name:"James", city:"Miami", ...}
0x6134	333363	{ name:"Betty", city:"Madison", ...}

0x9531	145783	{ name:"Bob", city:"London", ...}
0xB082	643145	{ name:"Val", city:"Seattle", ...}

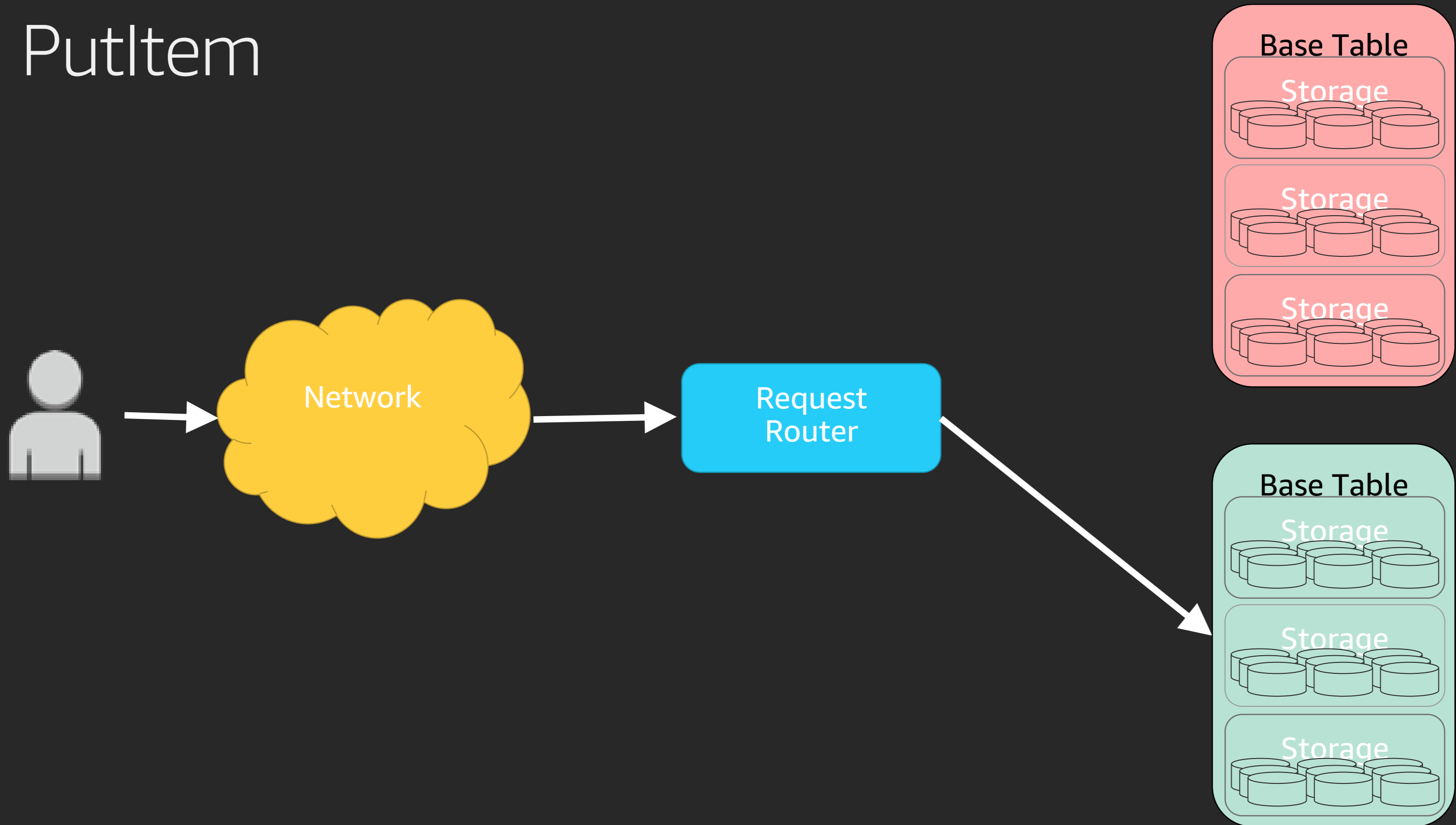
0xEA8A	723342	{ name:"Jeff", city:"Toledo", ...}
0xF355	523422	{ name:"Alex", city:"London", ...}

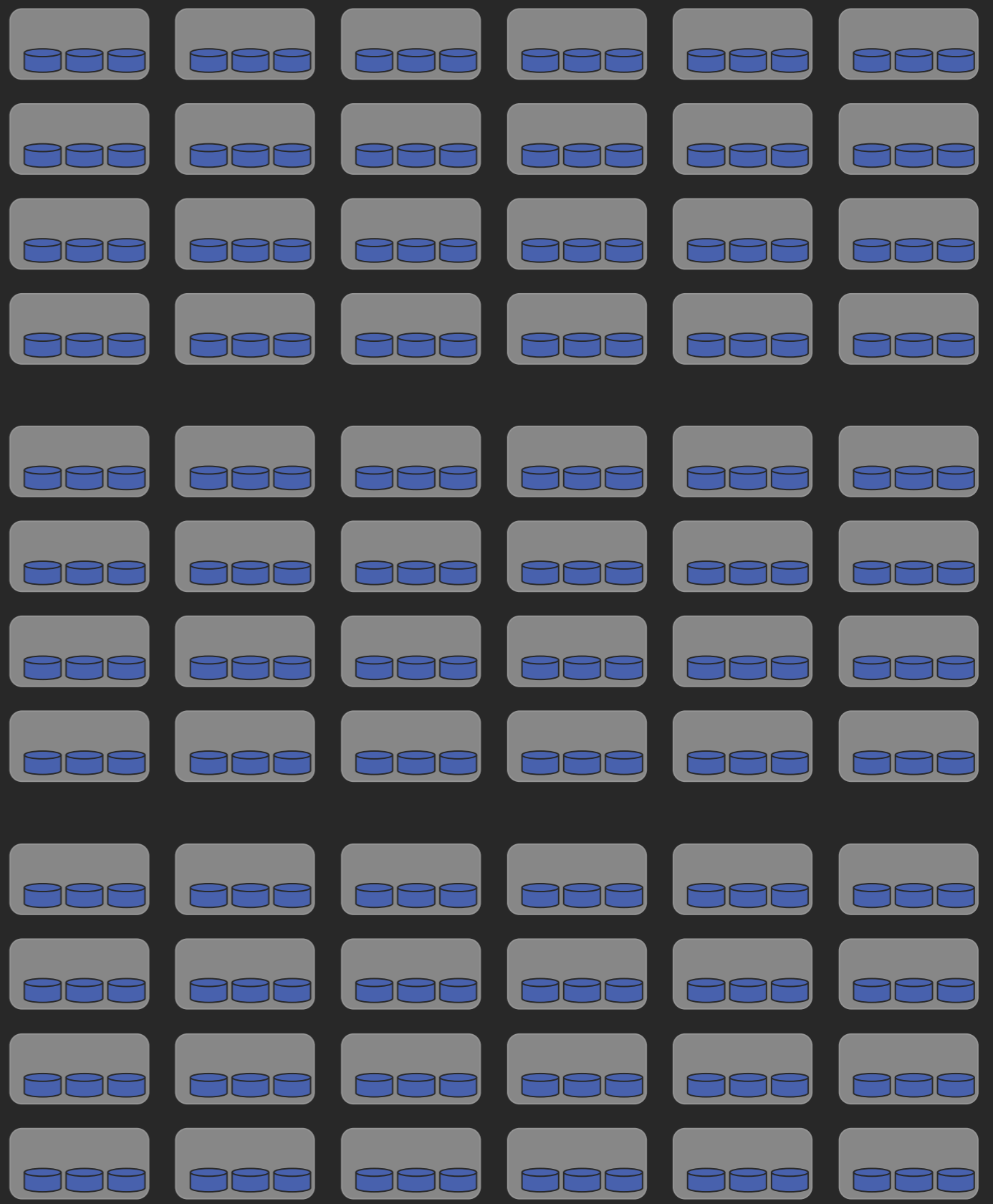
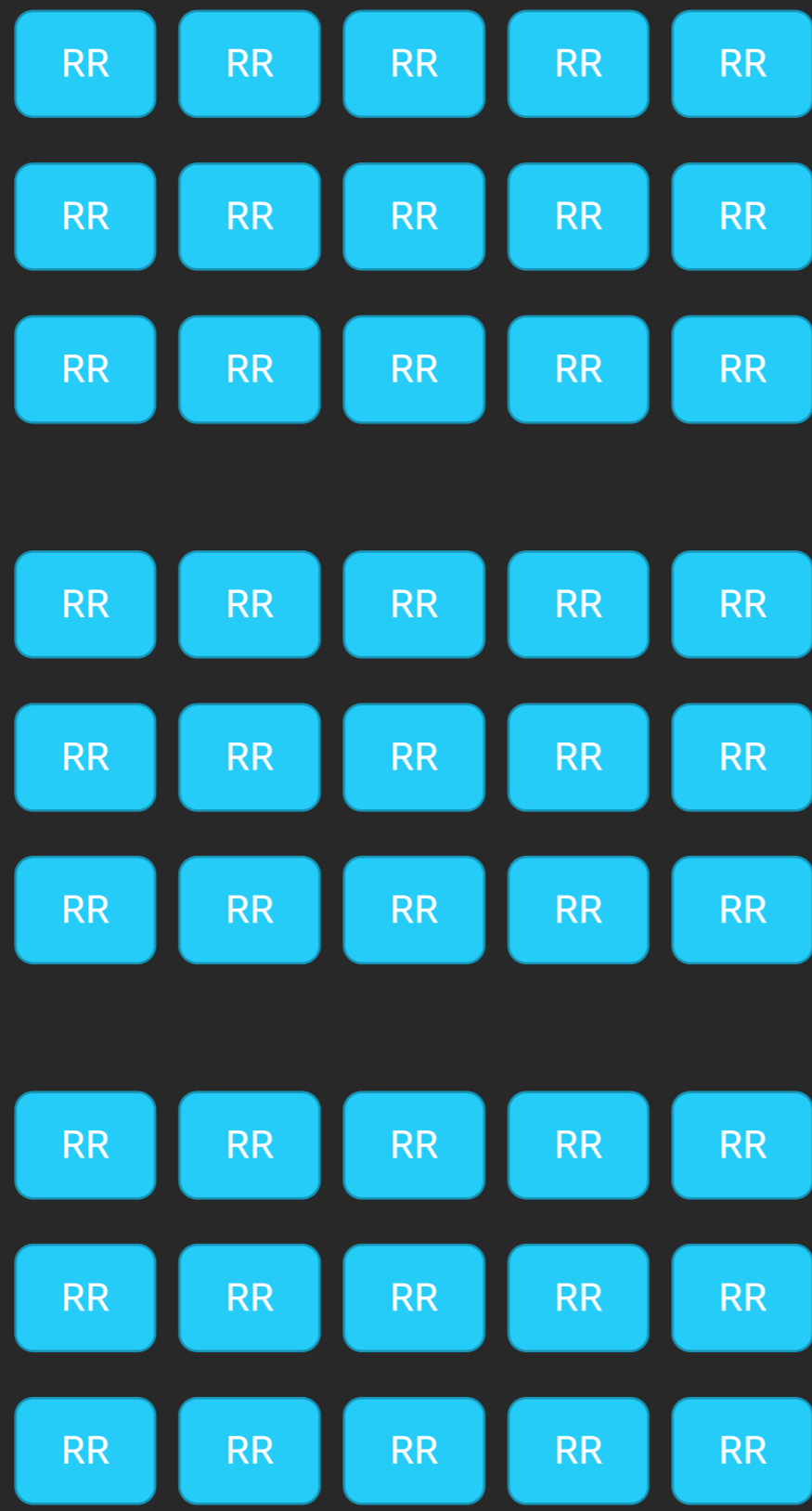
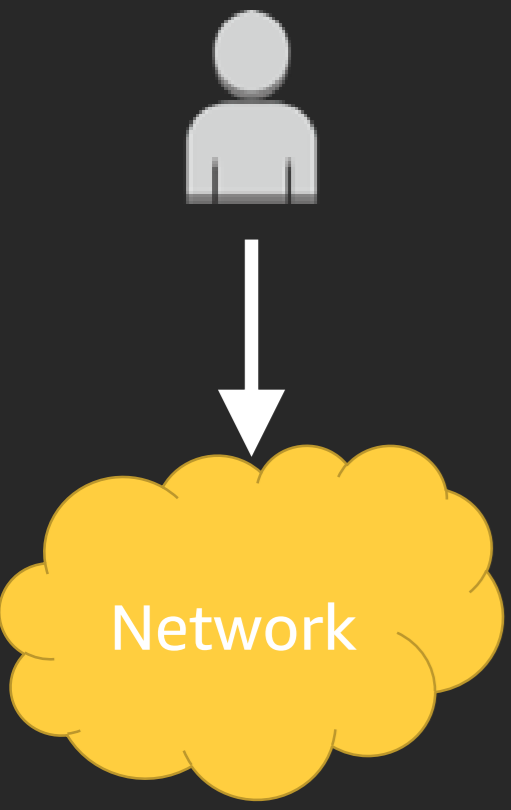
0x12A8	236294	{ name:"Sara", city:"Tampa", ...}
0x3391	445104	{ name:"James", city:"Miami", ...}
0x6134	333363	{ name:"Betty", city:"Madison", ...}

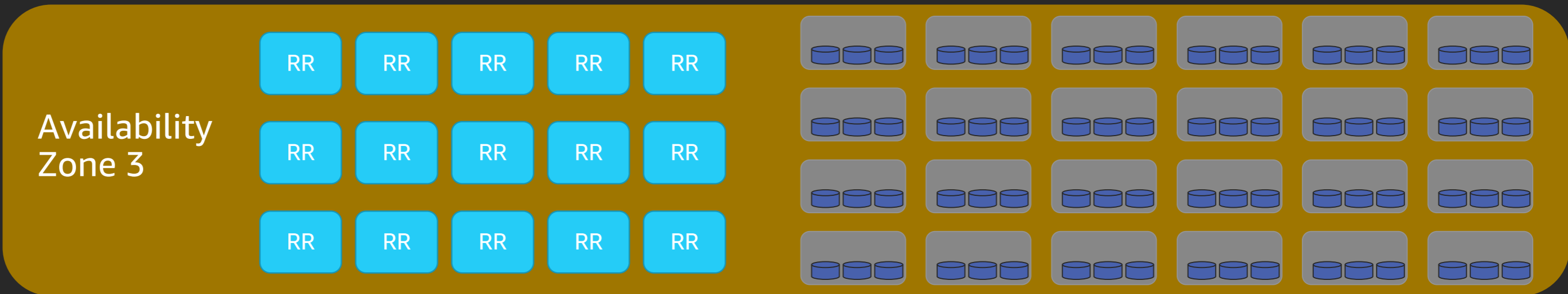
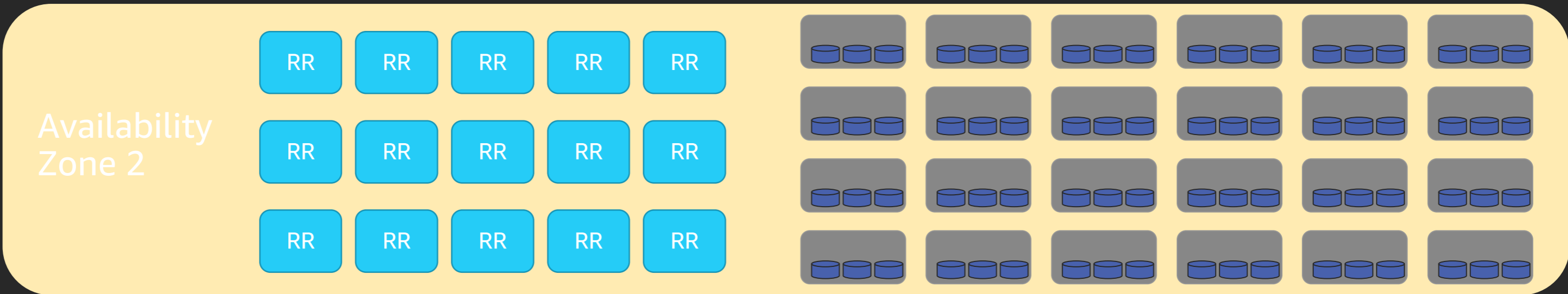
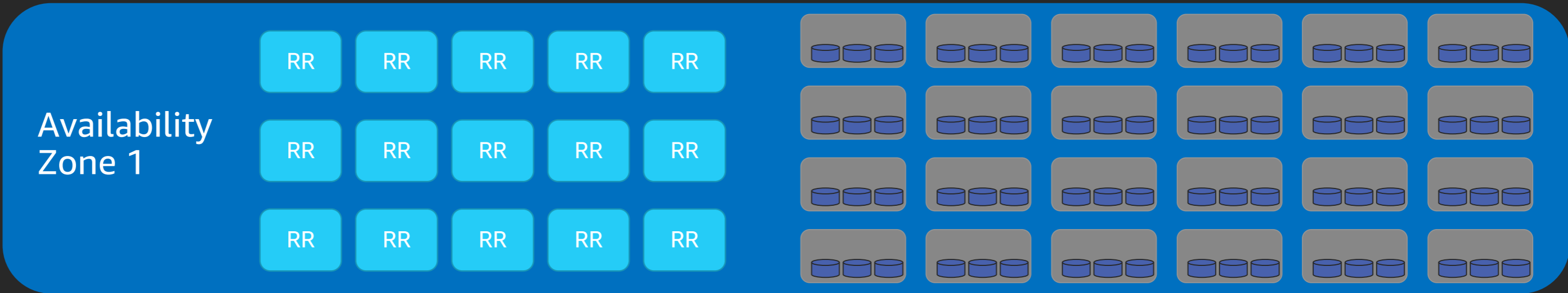
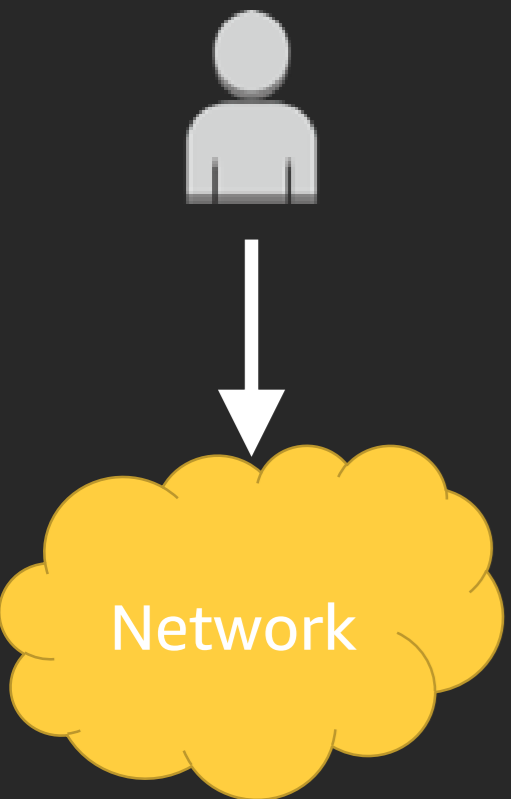
0x9531	145783	{ name:"Bob", city:"London", ...}
0xB082	643145	{ name:"Val", city:"Seattle", ...}

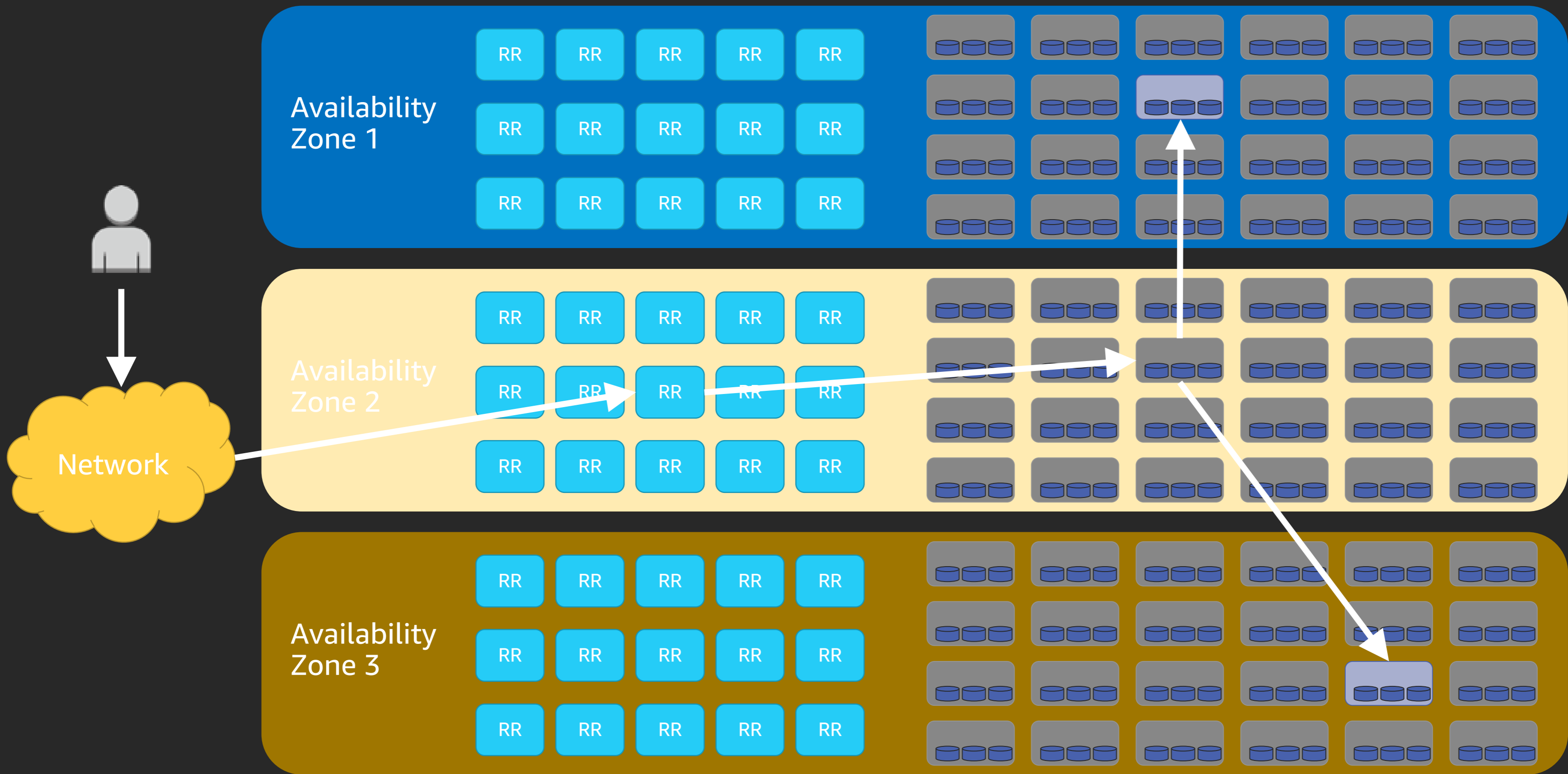
0xEA8A	723342	{ name:"Jeff", city:"Toledo", ...}
0xF355	523422	{ name:"Alex", city:"London", ...}

# PutItem







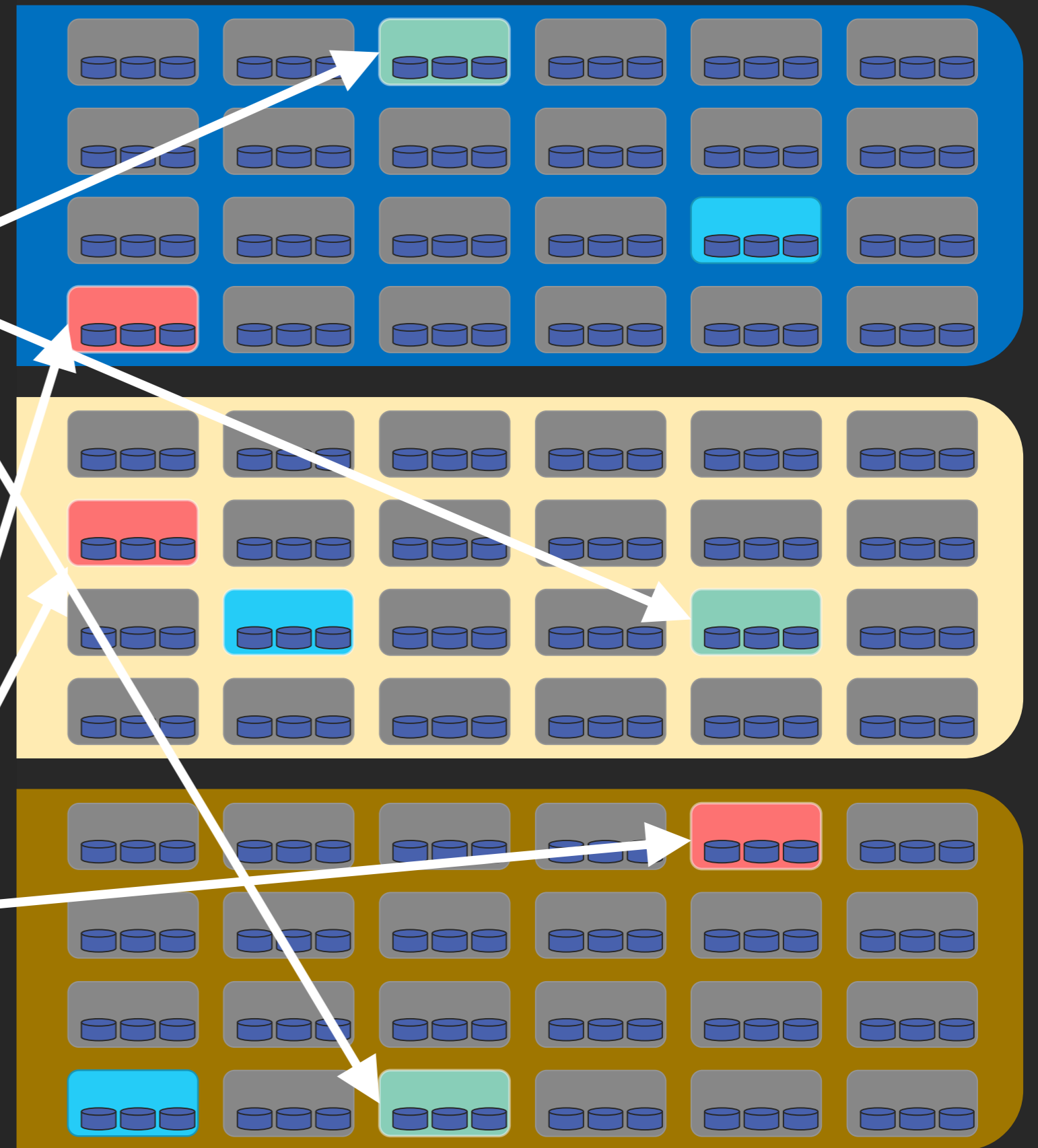


# PutItem

0x12A8	236294	{ name:"Sara", city:"Tampa", ...}
0x3391	445104	{ name:"James", city:"Miami", ...}
0x6134	333363	{ name:"Betty", city:"Madison", ...}

0x9531	145783	{ name:"Bob", city:"London", ...}
0xB082	643145	{ name:"Val", city:"Seattle", ...}

0xEA8A	723342	{ name:"Jeff", city:"Toledo", ...}
0xF355	523422	{ name:"Alex", city:"London", ...}

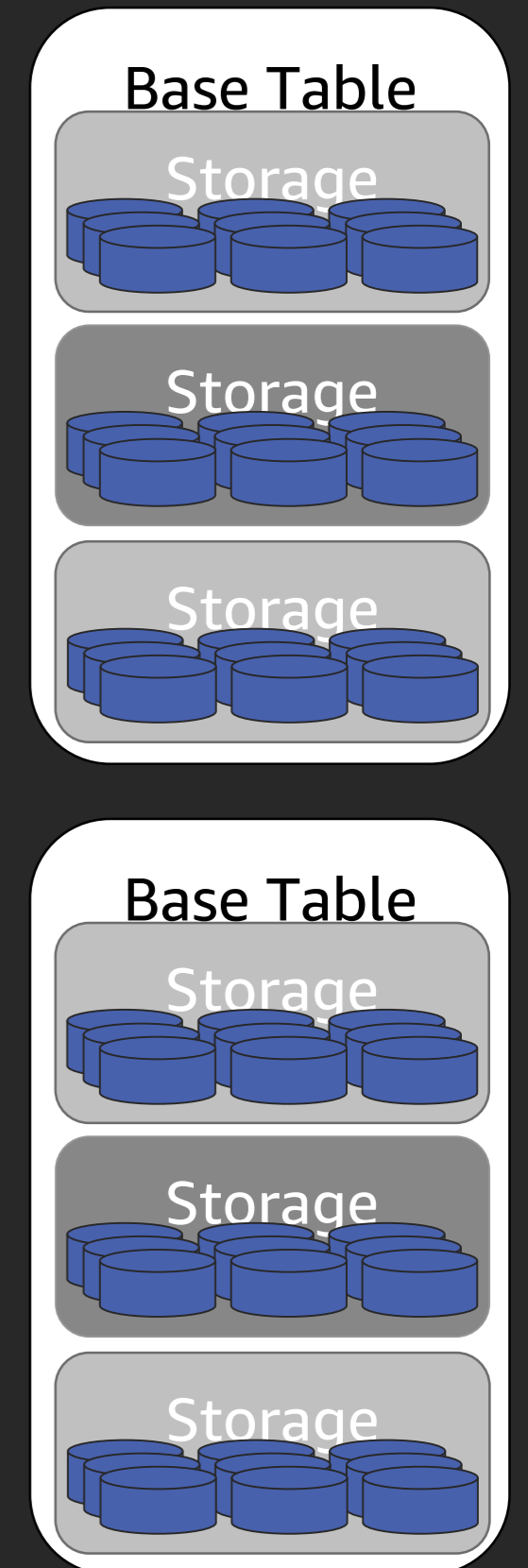
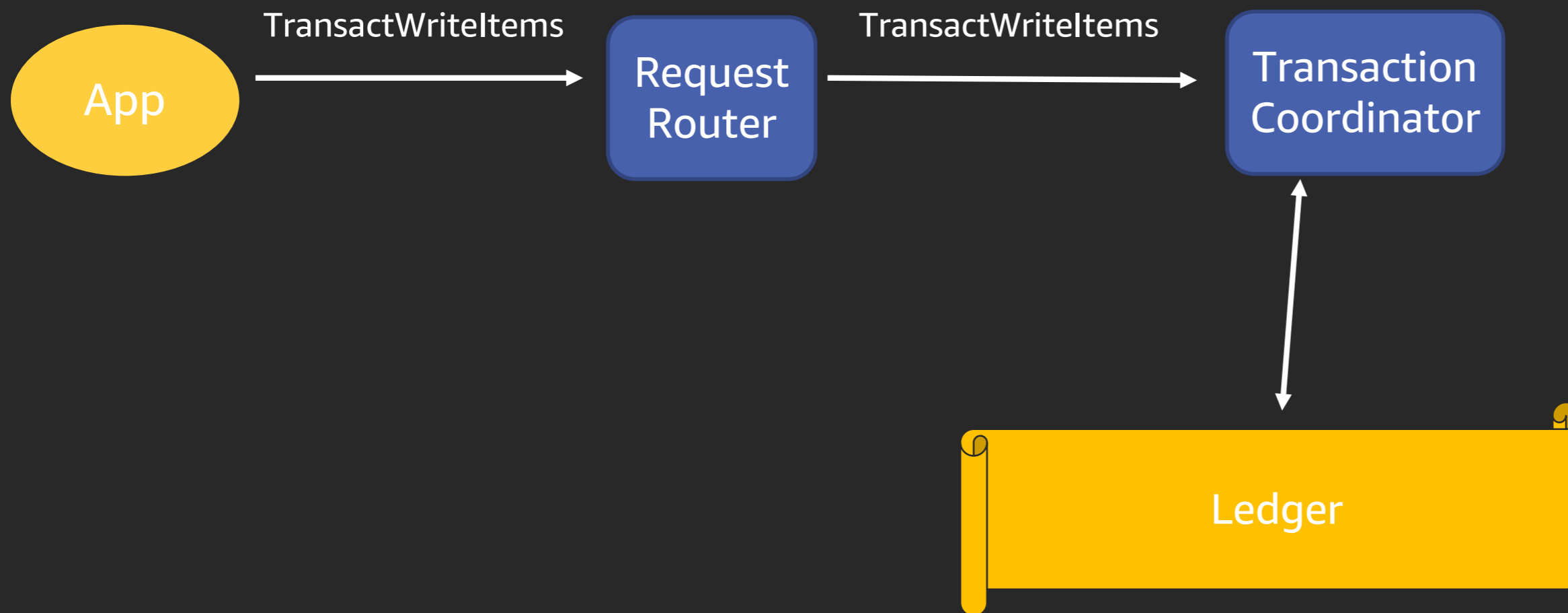


# Questions

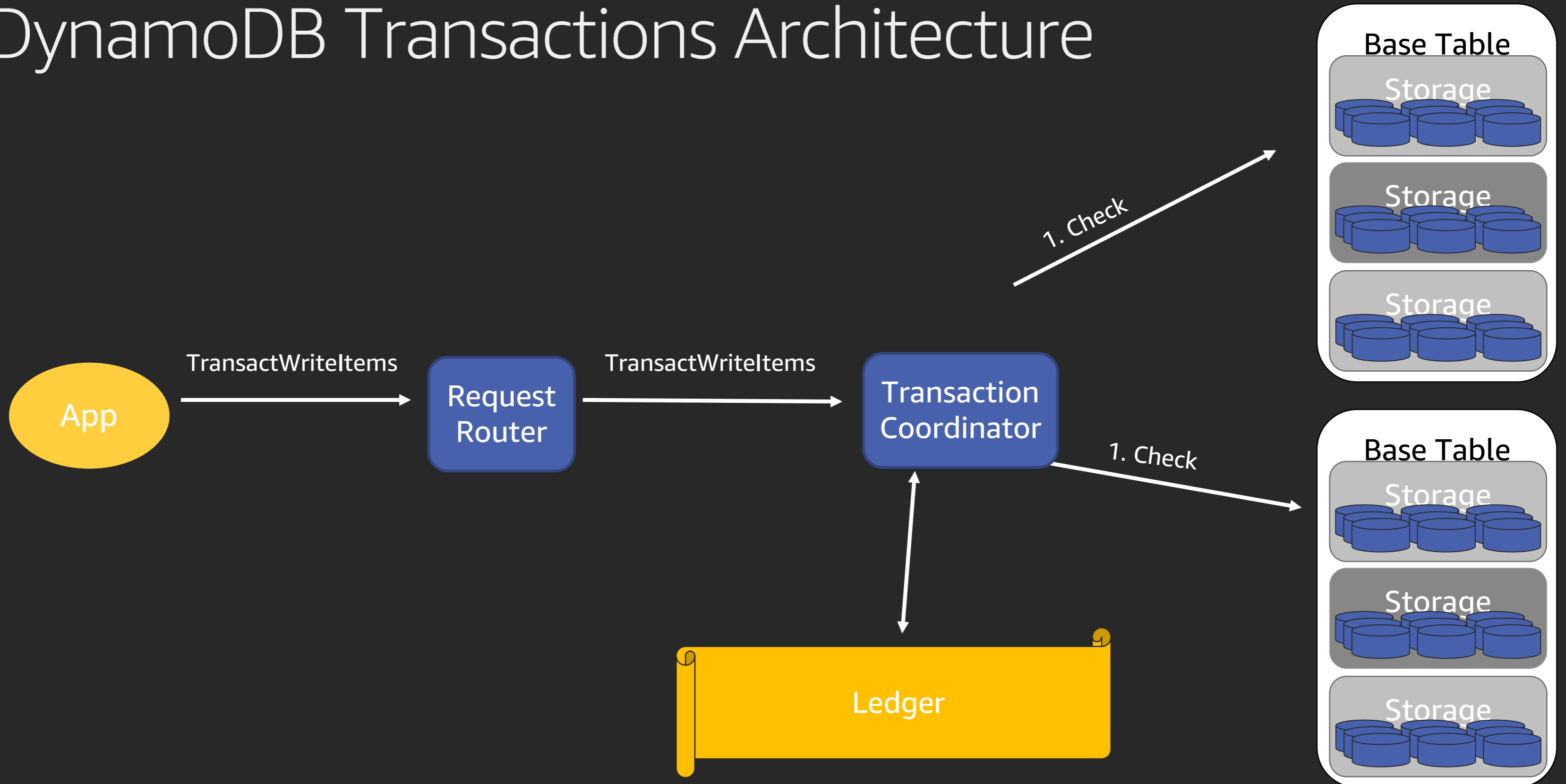


# Transactions

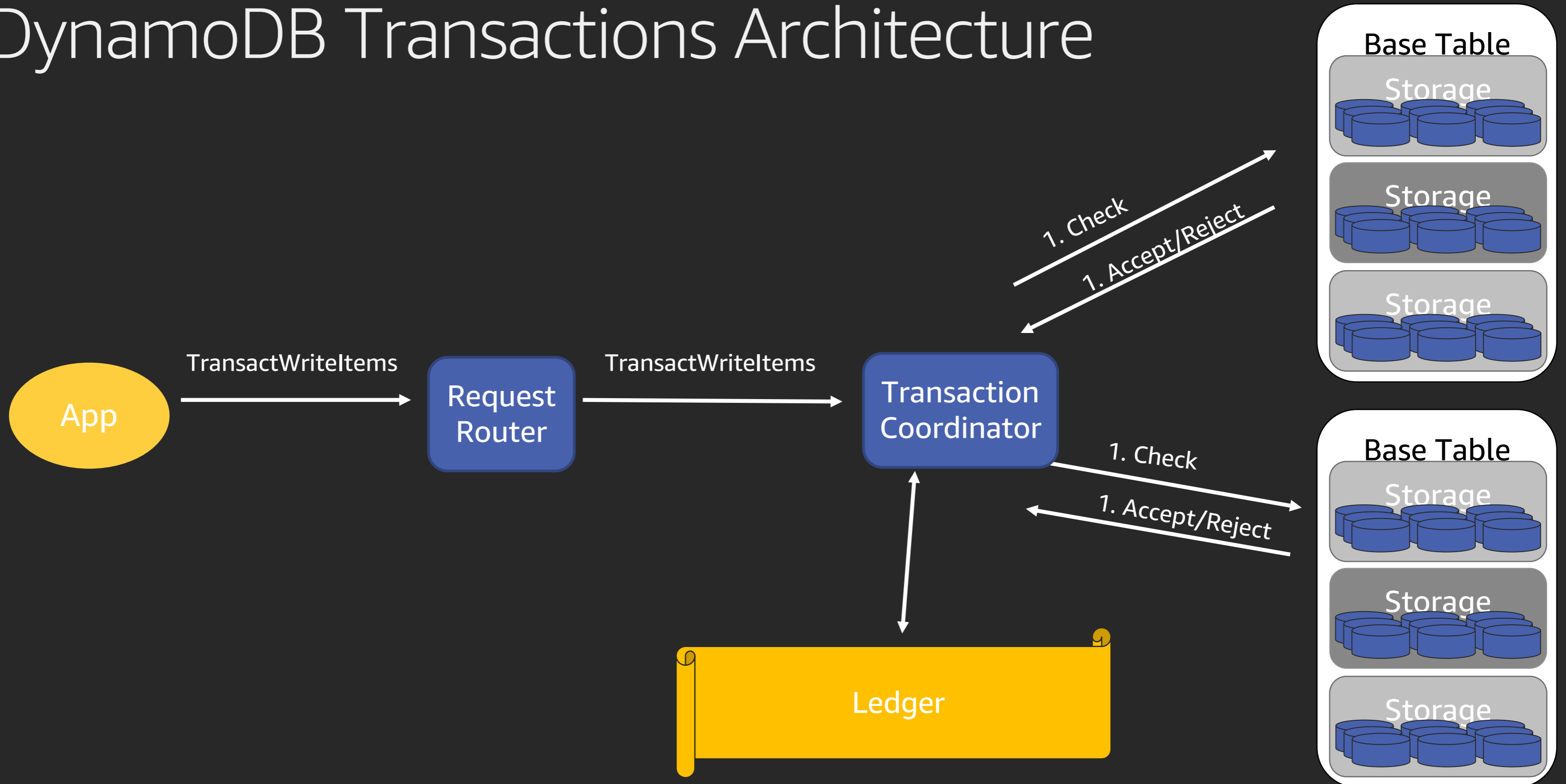
# DynamoDB Transactions Architecture



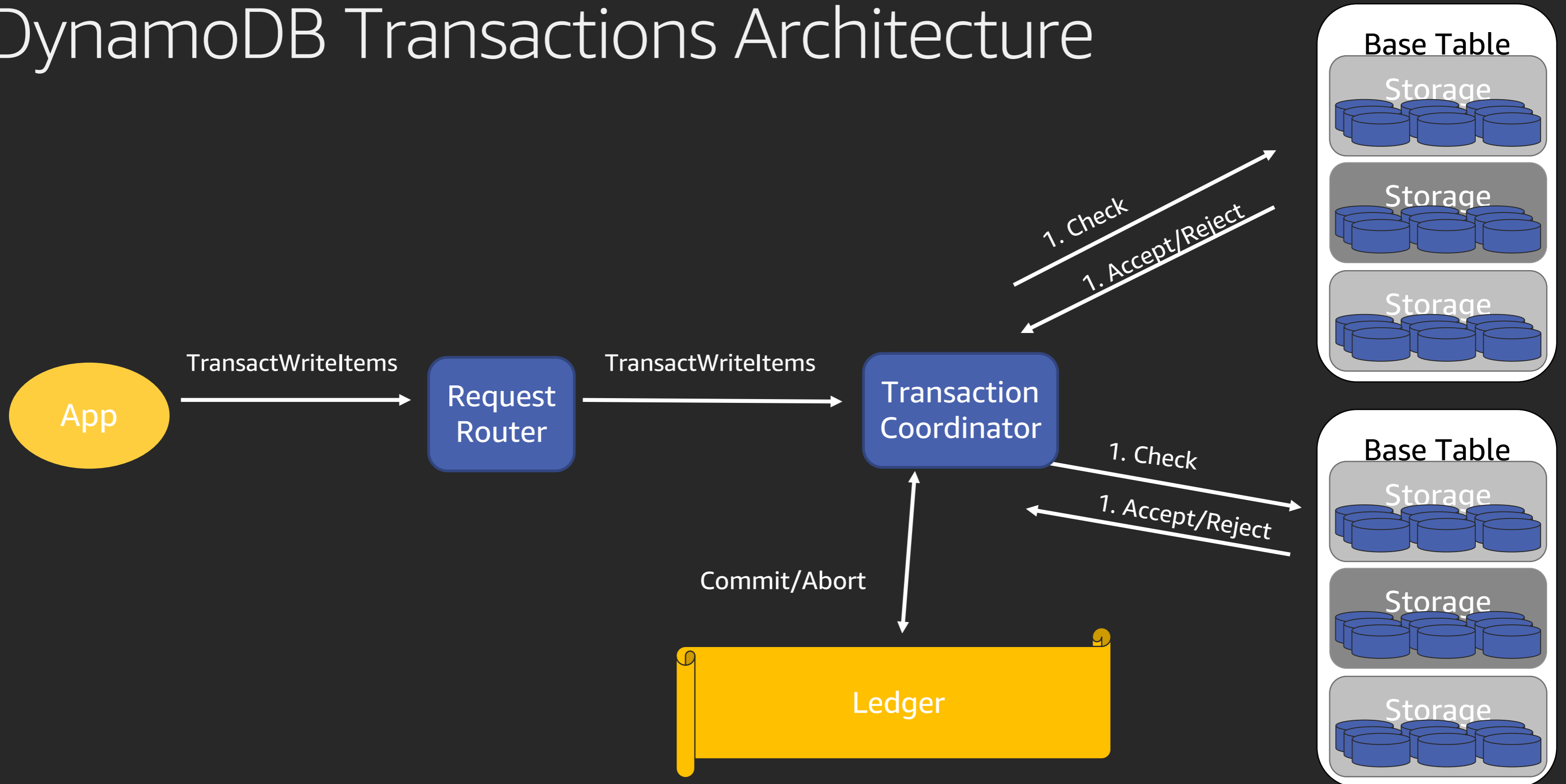
# DynamoDB Transactions Architecture



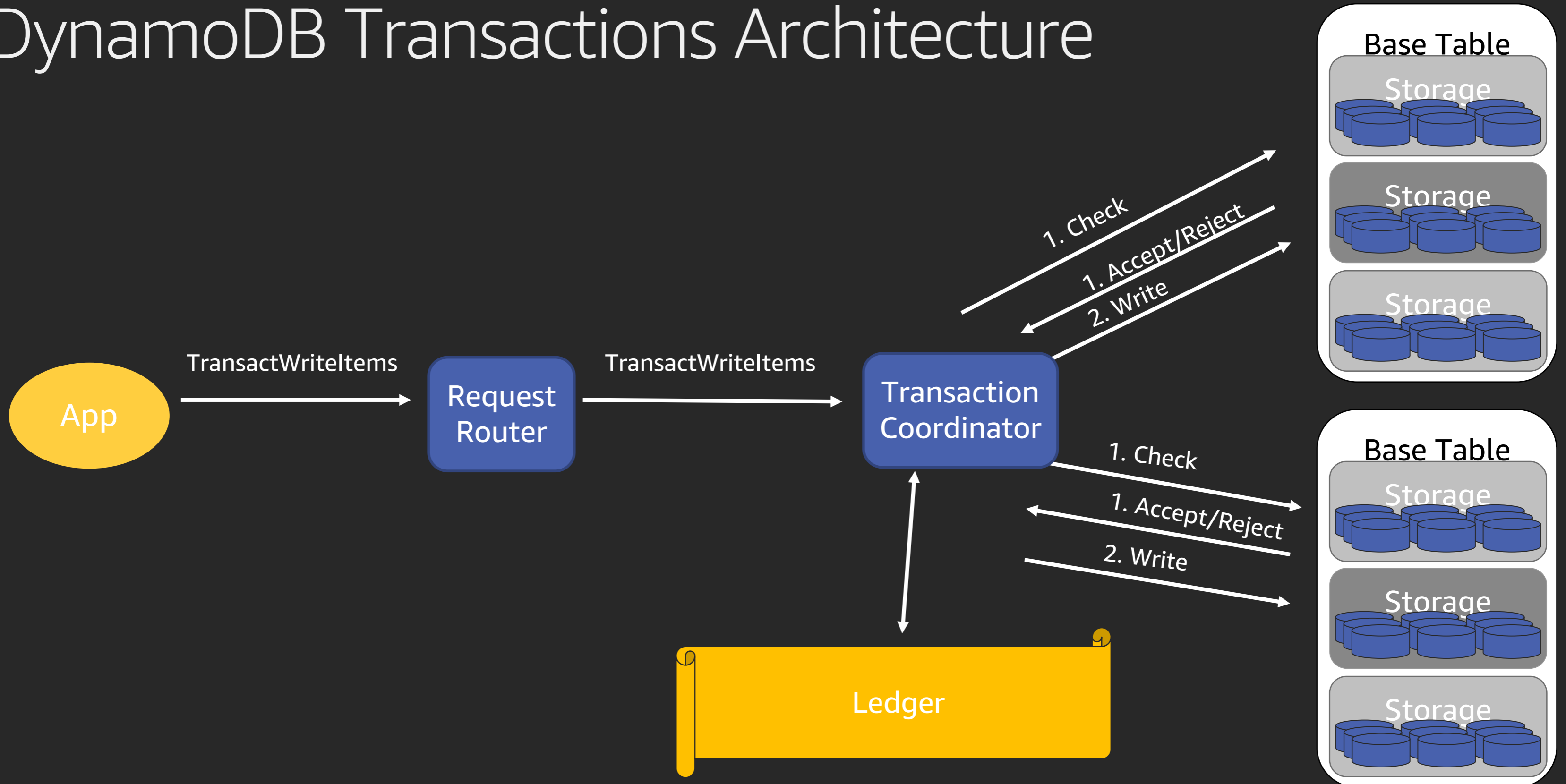
# DynamoDB Transactions Architecture



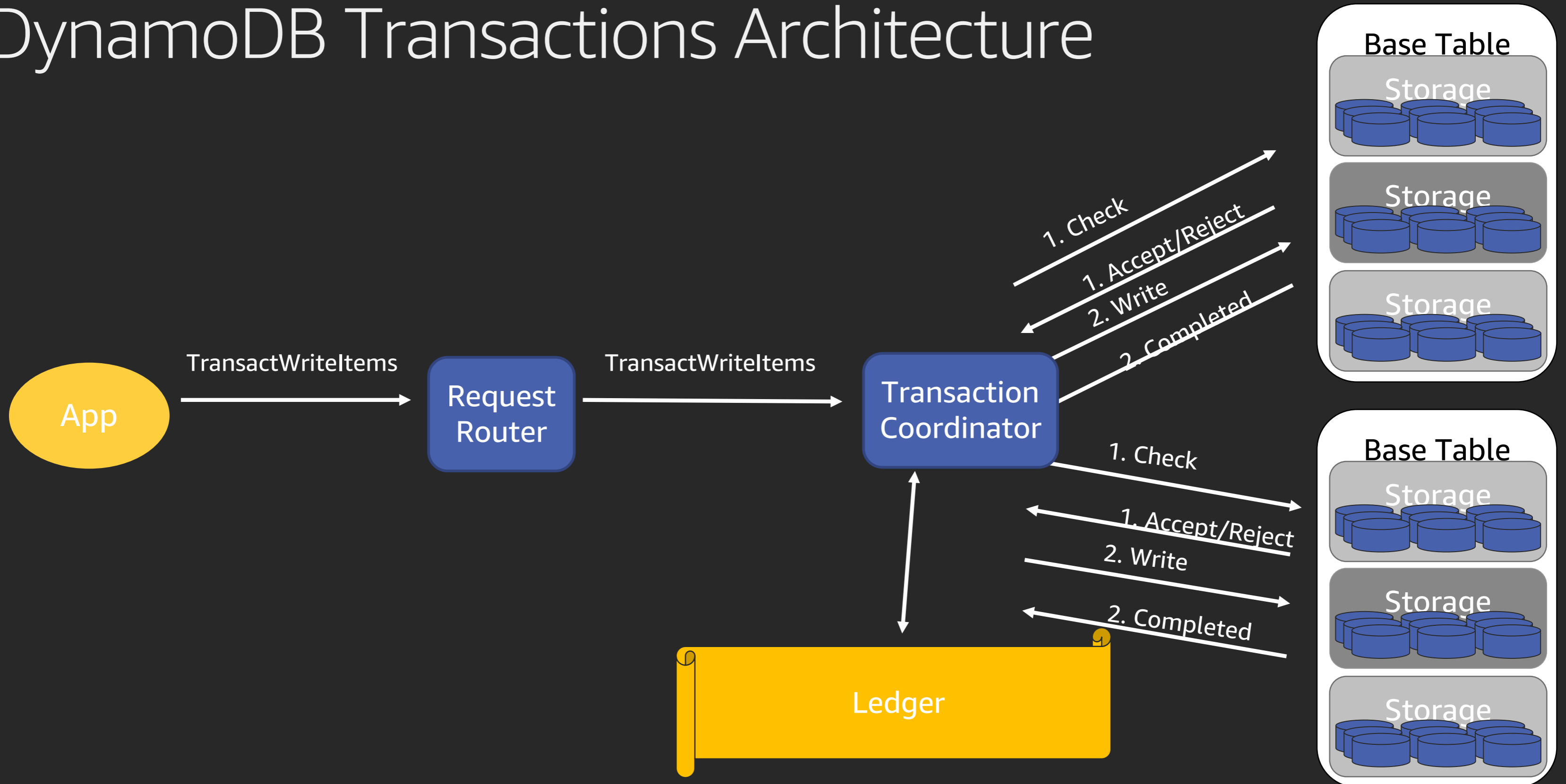
# DynamoDB Transactions Architecture



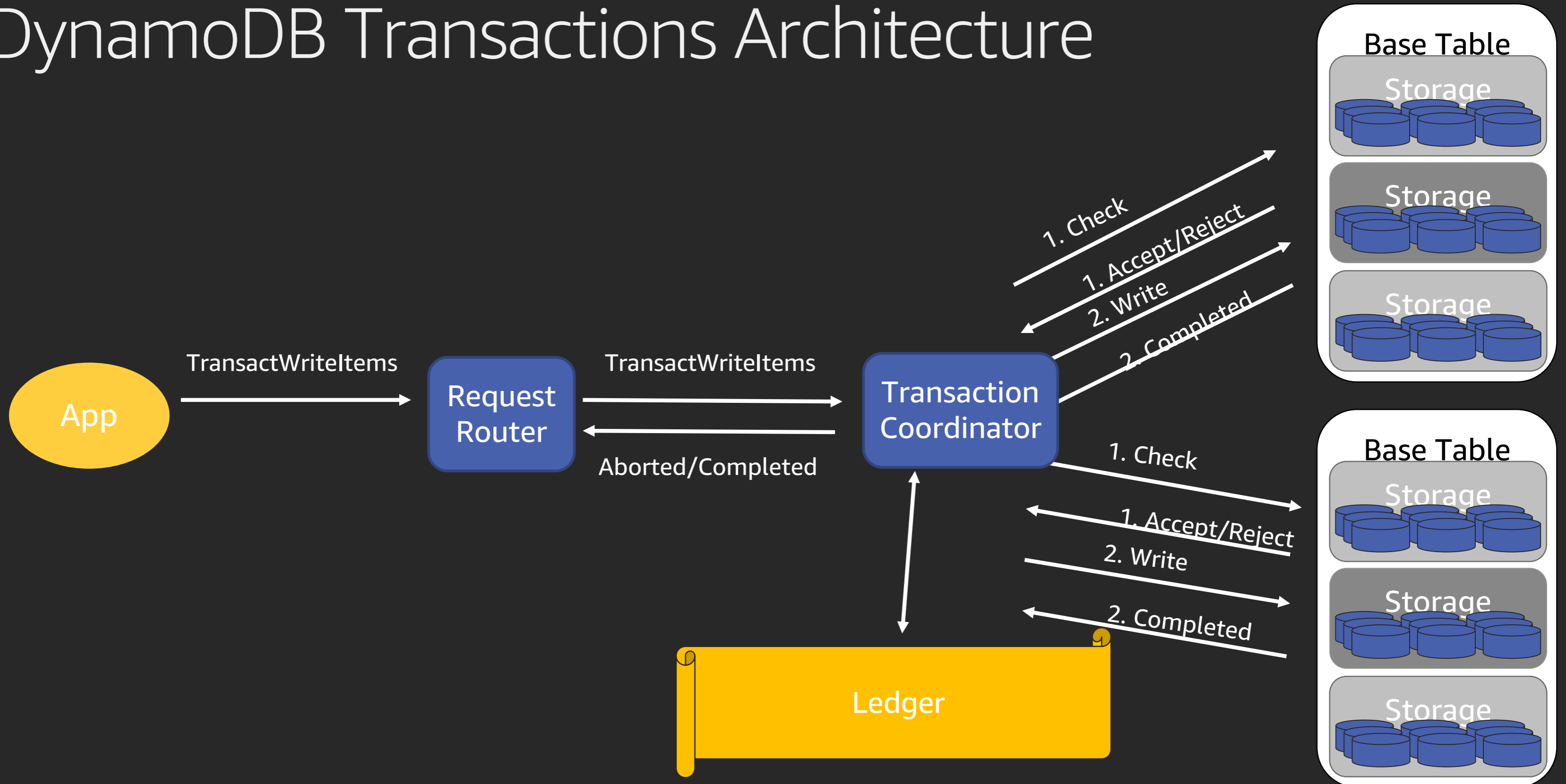
# DynamoDB Transactions Architecture



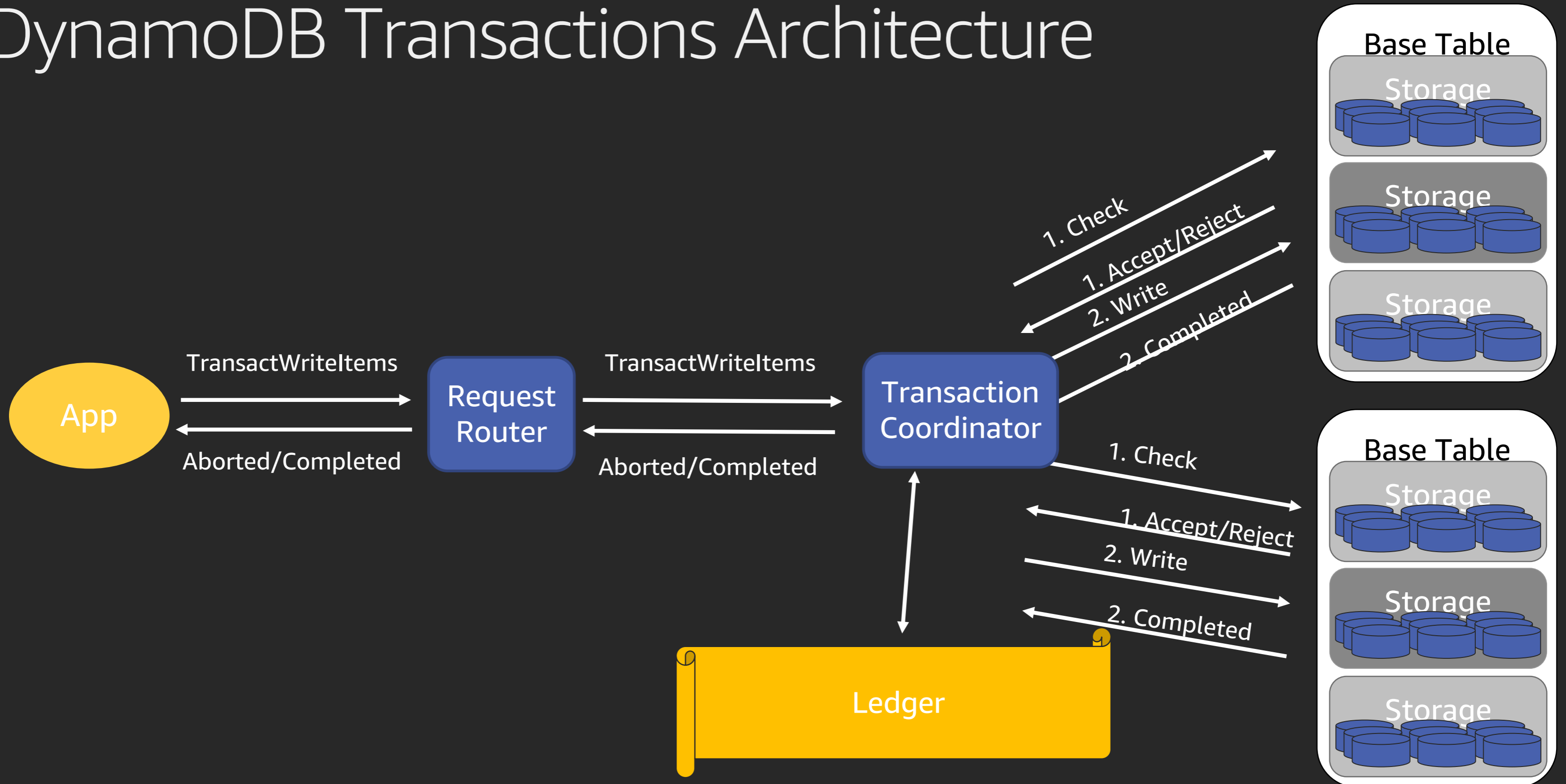
# DynamoDB Transactions Architecture



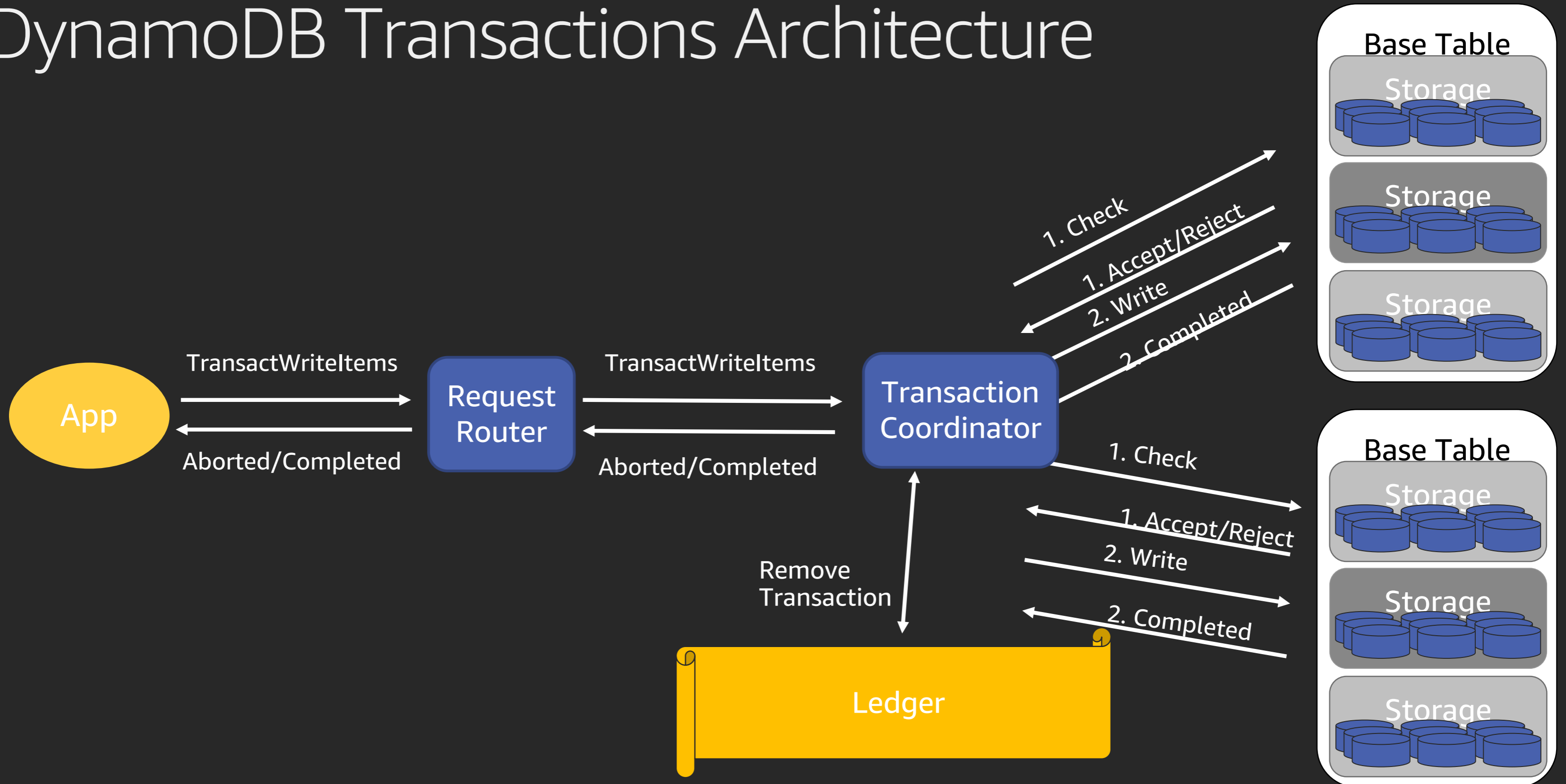
# DynamoDB Transactions Architecture



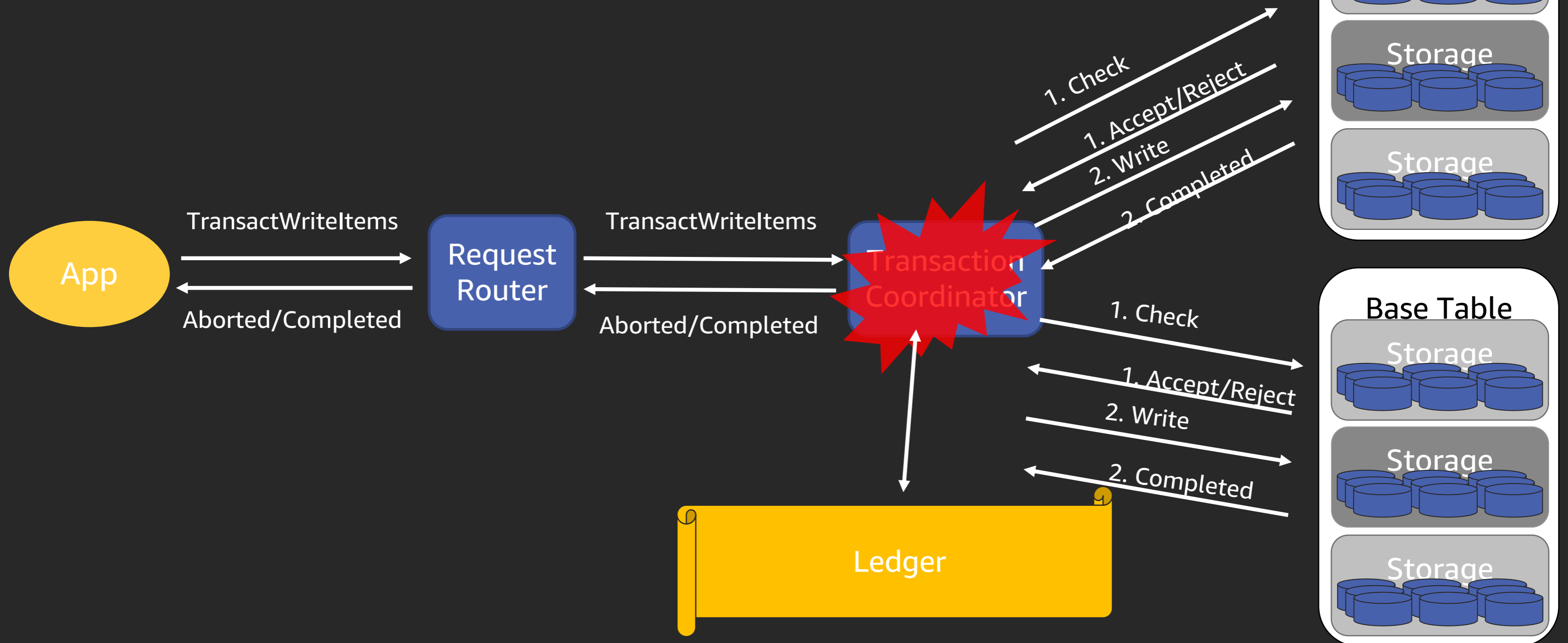
# DynamoDB Transactions Architecture



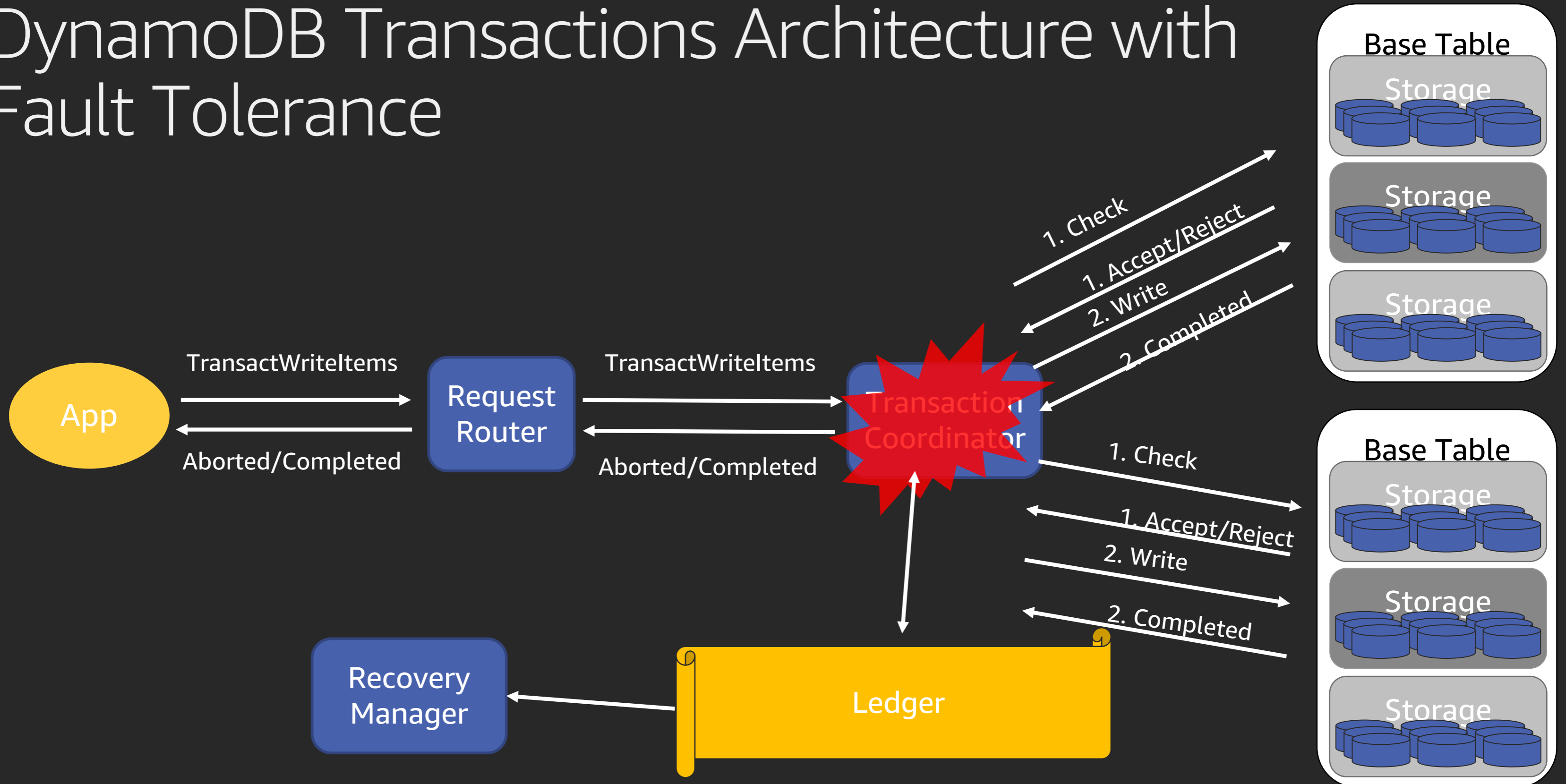
# DynamoDB Transactions Architecture



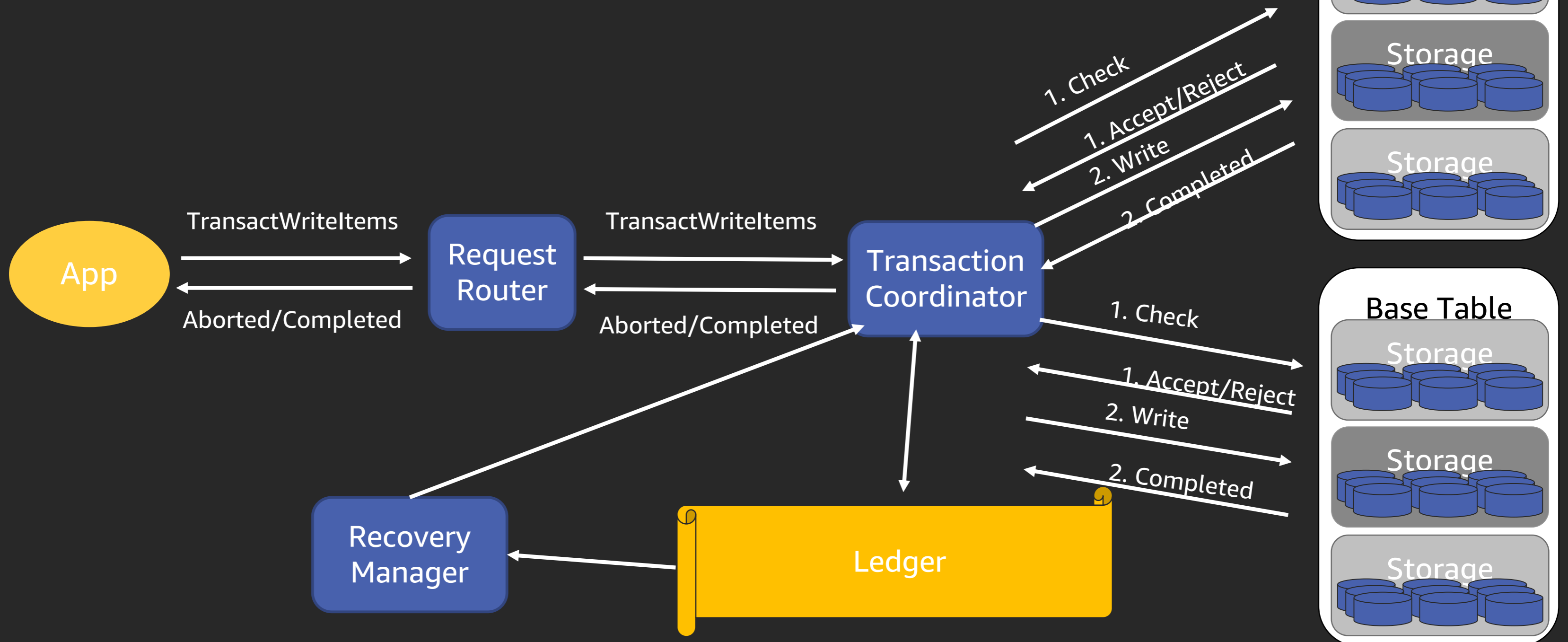
# DynamoDB Transactions Architecture with Fault Tolerance



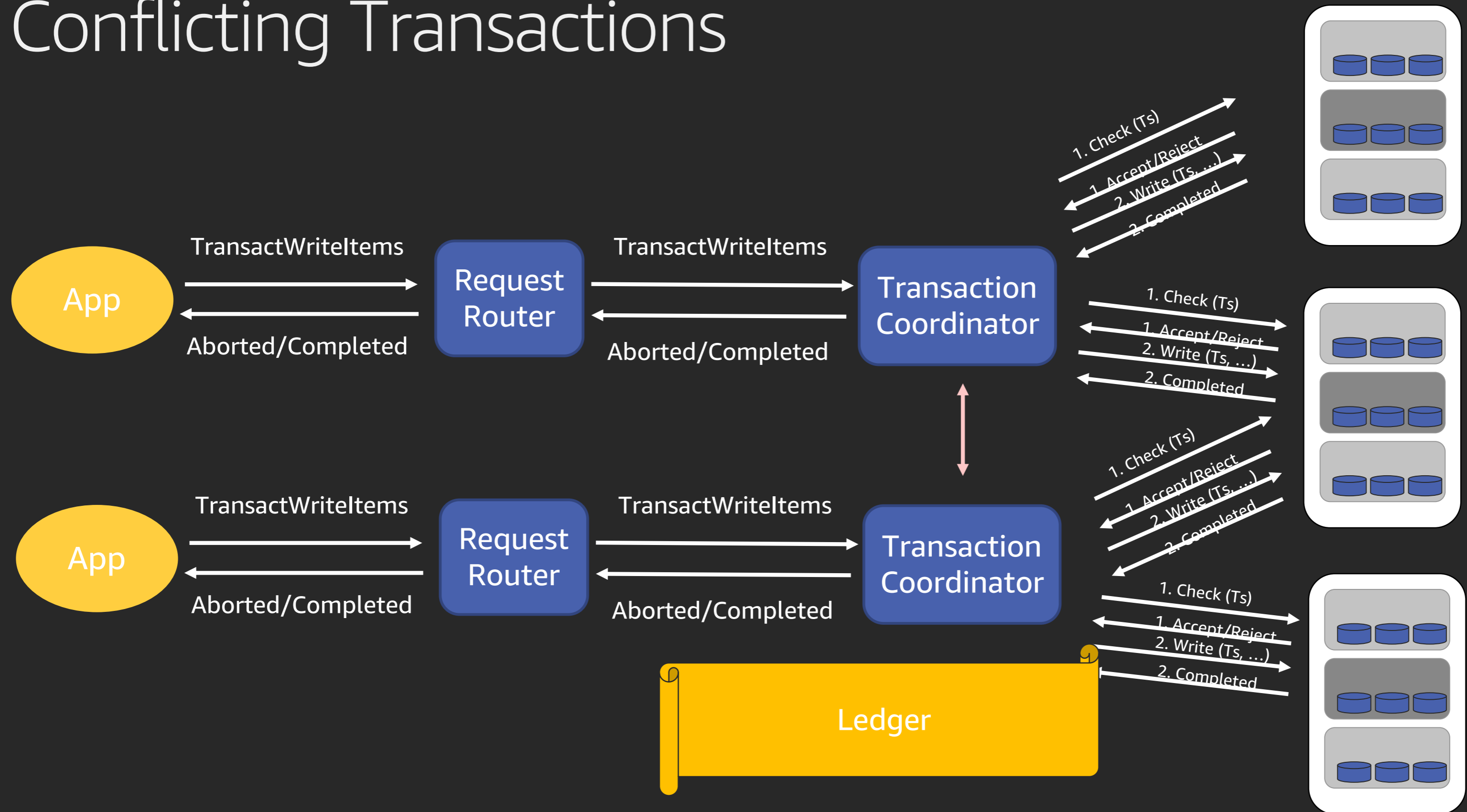
# DynamoDB Transactions Architecture with Fault Tolerance



# DynamoDB Transactions Architecture with Fault Tolerance



# Conflicting Transactions

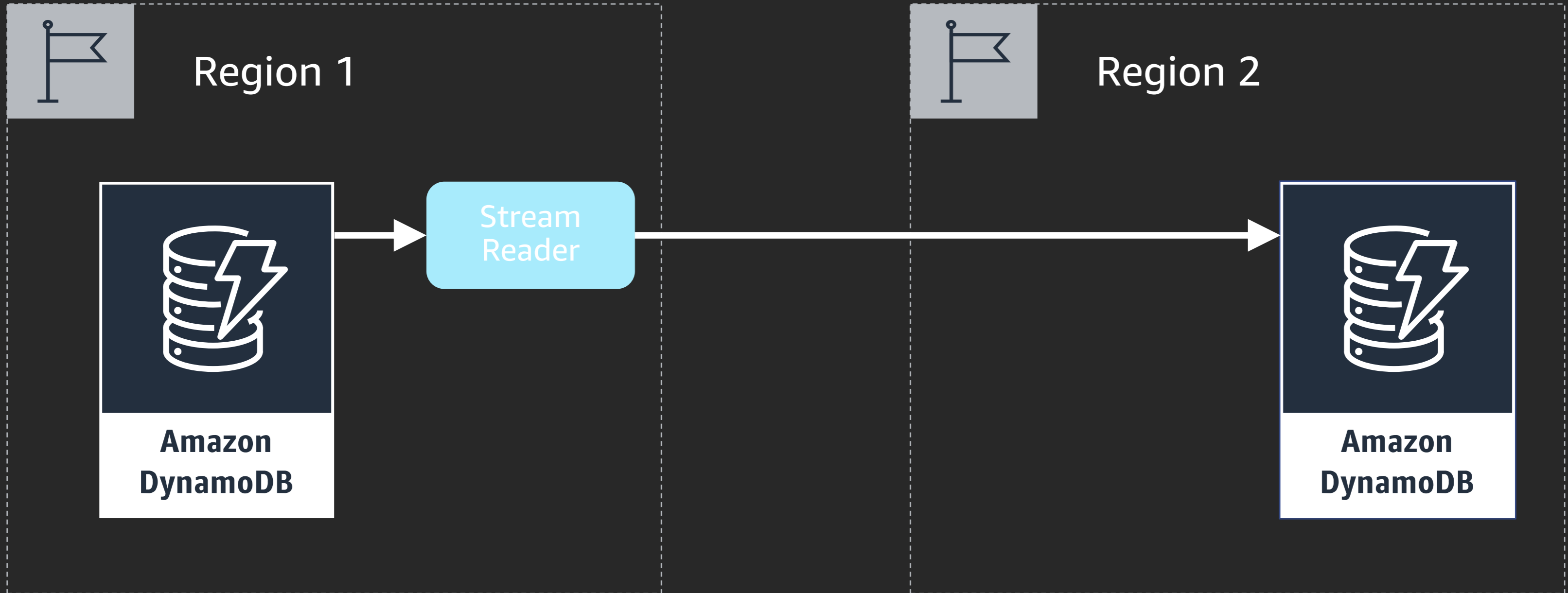


# Questions

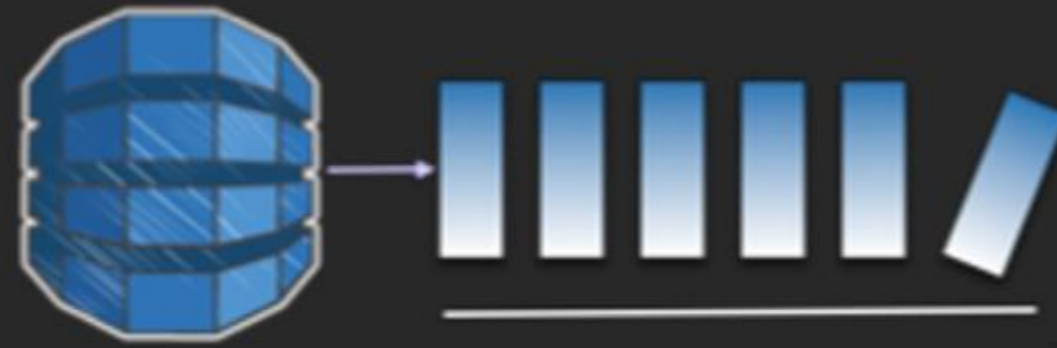


# Global Tables

# High-Level Architecture

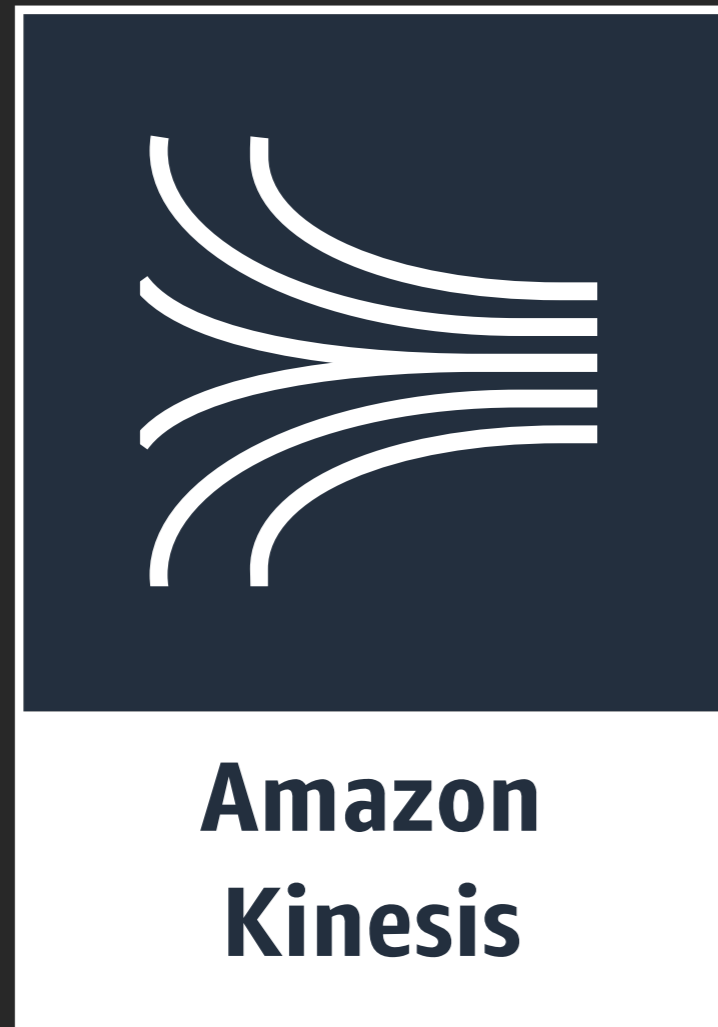


# DynamoDB Streams



- All table mutations (Put, Update, Delete)
- No duplicates
- In Order (By Key)
- New and Old Item Image Available

# DynamoDB Streams



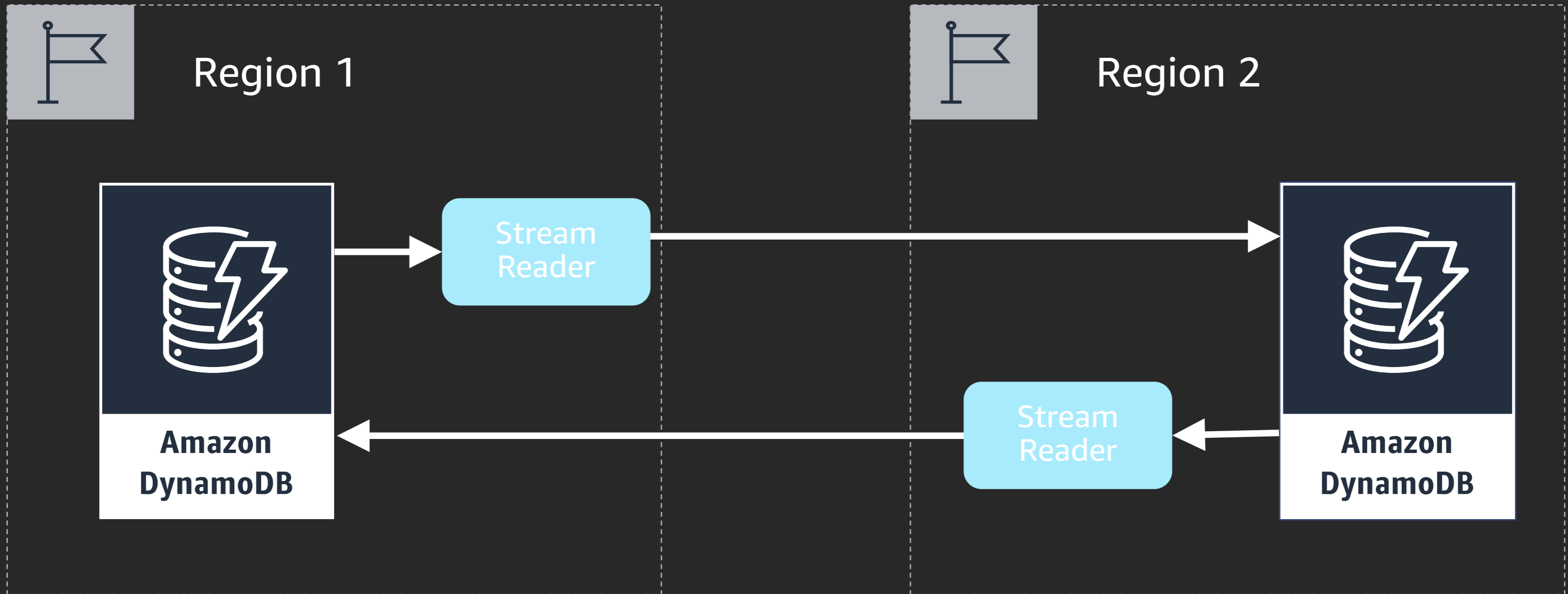
Shards

Kinesis Client Library

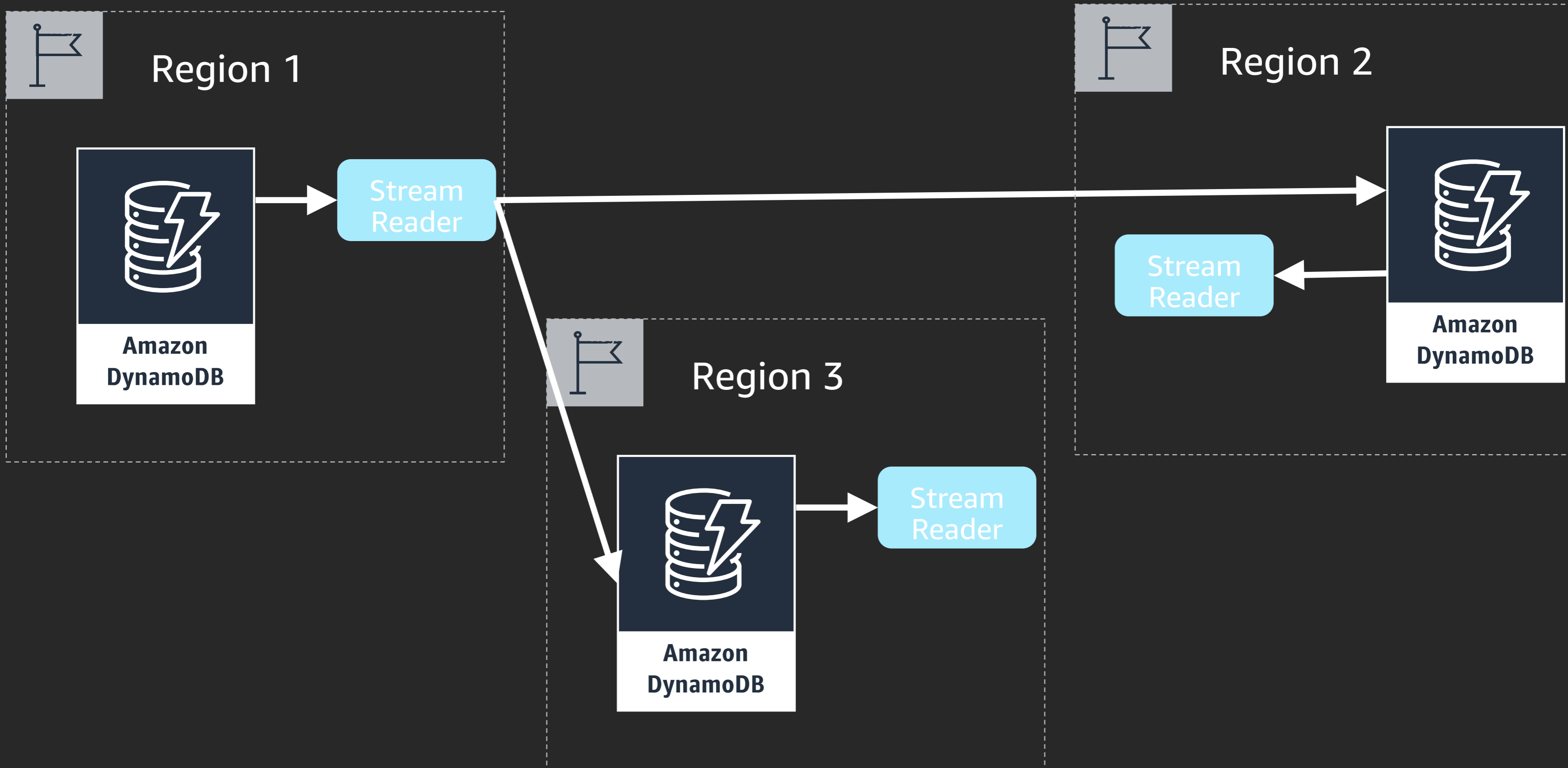
Records

Stream Checkpointing

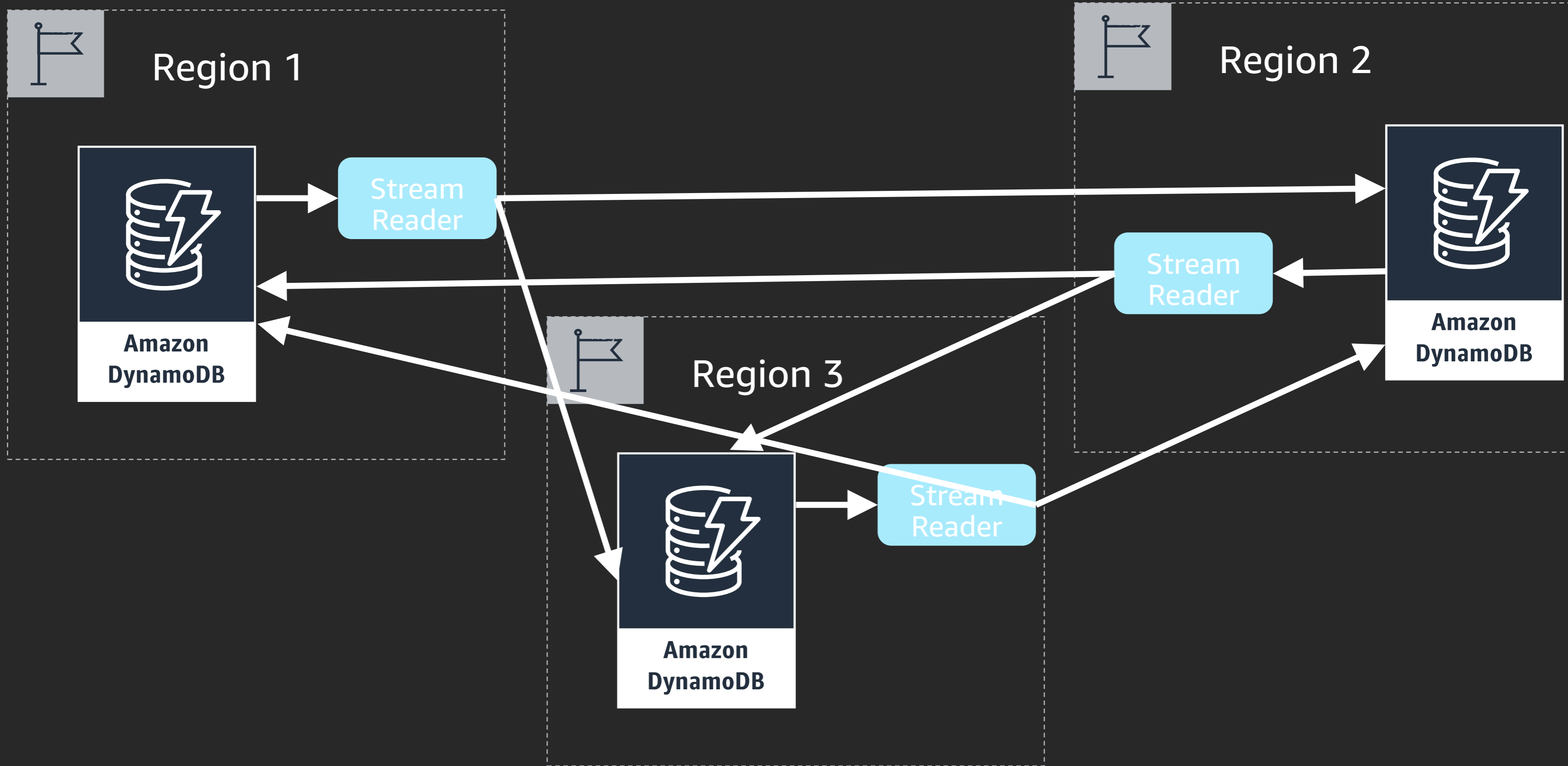
# Multi-master replication



# Multi-region replication



# Multi-region



# Questions



# Autoscaling

# Provisioning Example

300 Read Capacity Units (RCU)

Ignoring Writes

# Provisioning

0x12A8	236294	{ name:"Sara", city:"Tampa", ...}
0x3391	445104	{ name:"James", city:"Miami", ...}
0x6134	333363	{ name:"Betty", city:"Madison", ...}

← 100 RCUs

0x9531	145783	{ name:"Bob", city:"London", ...}
0xB082	643145	{ name:"Val", city:"Seattle", ...}

← 100 RCUs

0xEA8A	723342	{ name:"Jeff", city:"Toledo", ...}
0xF355	523422	{ name:"Alex", city:"London", ...}

← 100 RCUs

# Token Bucket Algorithm



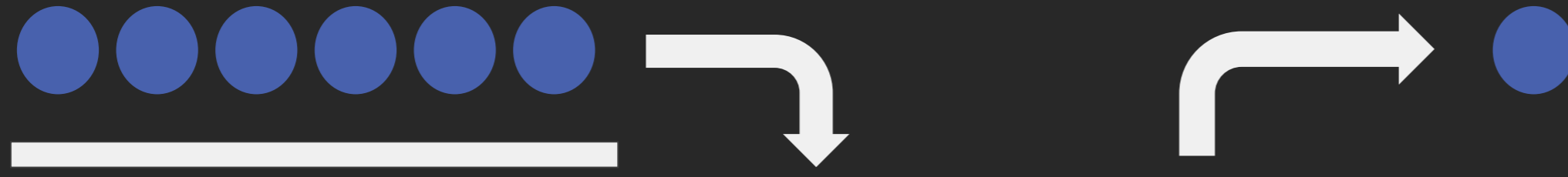
Refilled at RCU rate  
100 tokens/second



Emptied 1 Token per  
Request\*

\*Tokens deducted  
depends on Item size

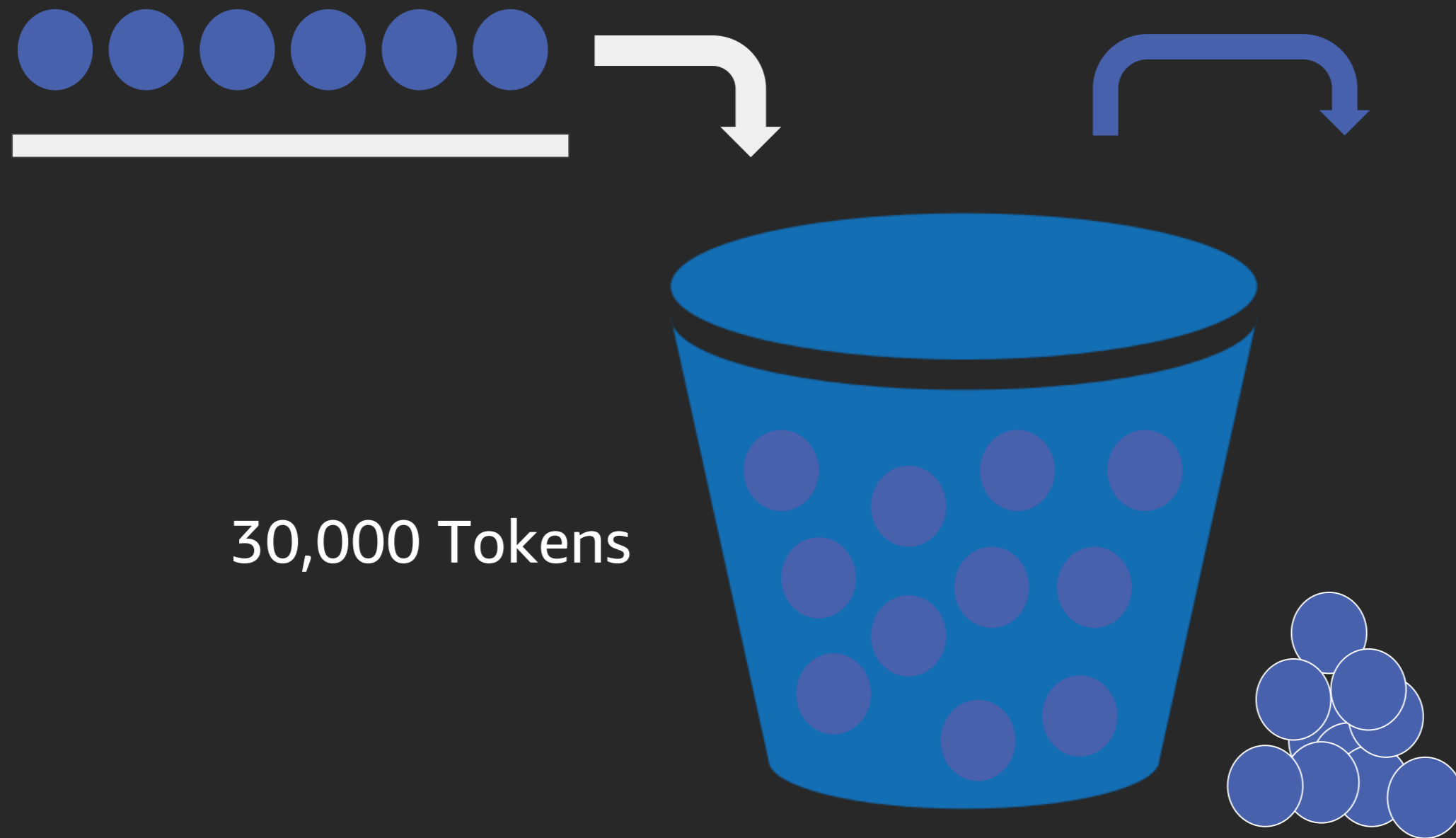
# Token Bucket Capacity



300 \* RCUs

5 minutes

# Full Token Bucket



# Partition Bursting



300 \* RCUs

5 minutes

# Unbalanced Load

0x12A8	236294	{ name:"Sara", city:"Tampa", ...}
0x3391	445104	{ name:"James", city:"Miami", ...}
0x6134	333363	{ name:"Betty", city:"Madison", ...}

25 RCU attempted

0x9531	145783	{ name:"Bob", city:"London", ...}
0xB082	643145	{ name:"Val", city:"Seattle", ...}

150 RCU attempted

0xEA8A	723342	{ name:"Jeff", city:"Toledo", ...}
0xF355	523422	{ name:"Alex", city:"London", ...}

50 RCU attempted

# Unbalanced Load

0x12A8	236294	{ name:"Sara", city:"Tampa", ...}
0x3391	445104	{ name:"James", city:"Miami", ...}
0x6134	333363	{ name:"Betty", city:"Madison", ...}

0x9531	145783	{ name:"Bob", city:"London", ...}
0xB082	643145	{ name:"Val", city:"Seattle", ...}

0xEA8A	723342	{ name:"Jeff", city:"Toledo", ...}
0xF355	523422	{ name:"Alex", city:"London", ...}

75 RCU unused

25 RCU admitted

OK

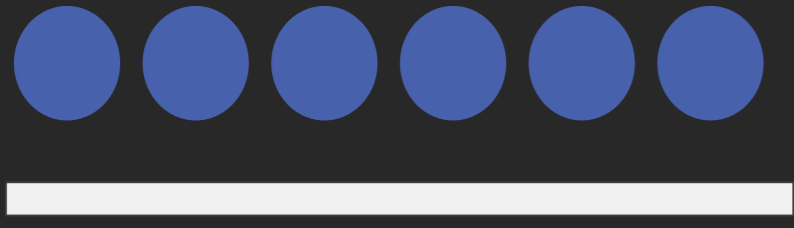
100 RCU admitted

50 throttles  
per second

50 RCU admitted

OK

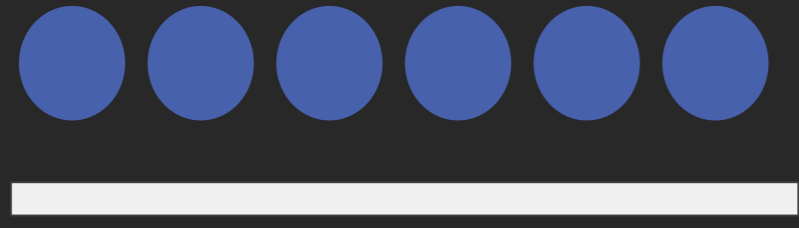
# Adaptive Capacity



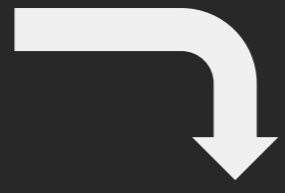
Refilled at RCU rate  
~~100 tokens/second~~  
150 tokens/second



# Adaptive Capacity



Refilled at RCU rate  
150 tokens/second



Adaptive Capacity  
Multiplier = 1.5

# Adaptive Capacity Active

75 RCU unused

0x12A8	236294	{ name:"Sara", city:"Tampa", ...}
0x3391	445104	{ name:"James", city:"Miami", ...}
0x6134	333363	{ name:"Betty", city:"Madison", ...}

25 RCU admitted

OK

0x9531	145783	{ name:"Bob", city:"London", ...}
0xB082	643145	{ name:"Val", city:"Seattle", ...}

150 RCU admitted

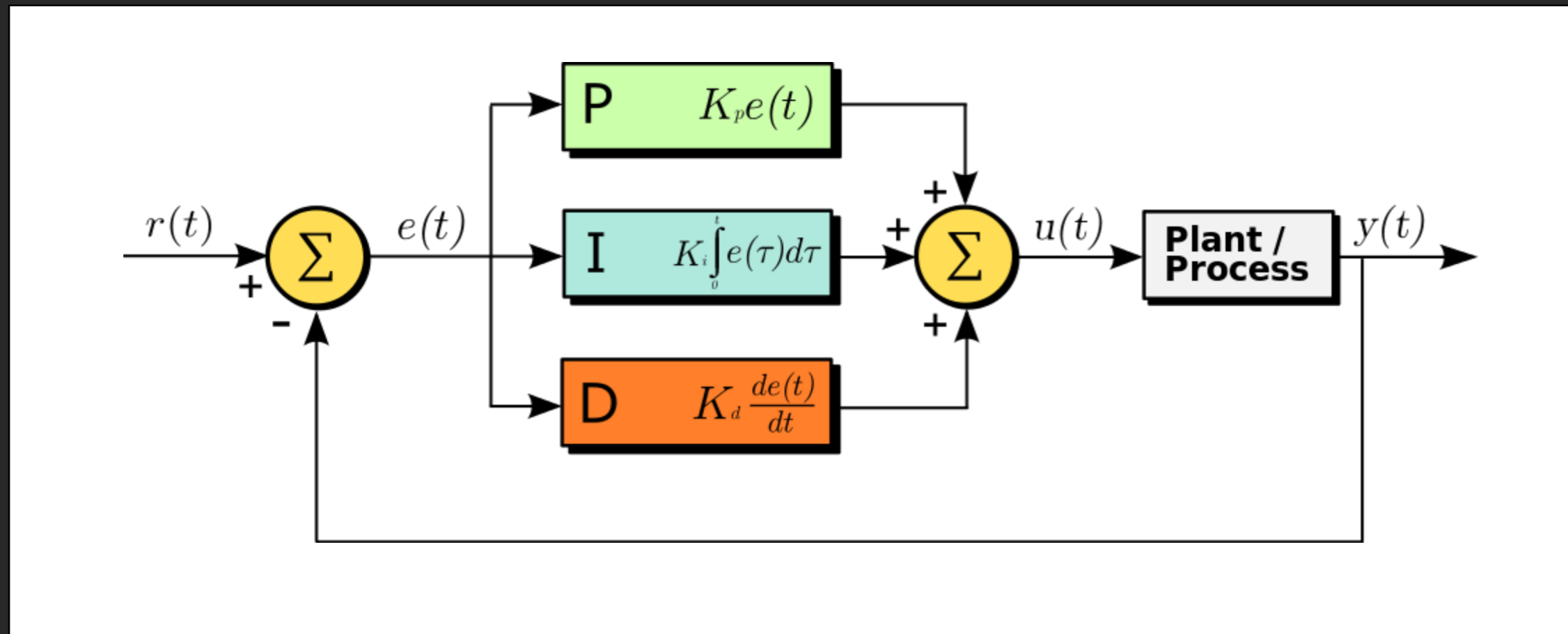
OK

0xEA8A	723342	{ name:"Jeff", city:"Toledo", ...}
0xF355	523422	{ name:"Alex", city:"London", ...}

50 RCU admitted

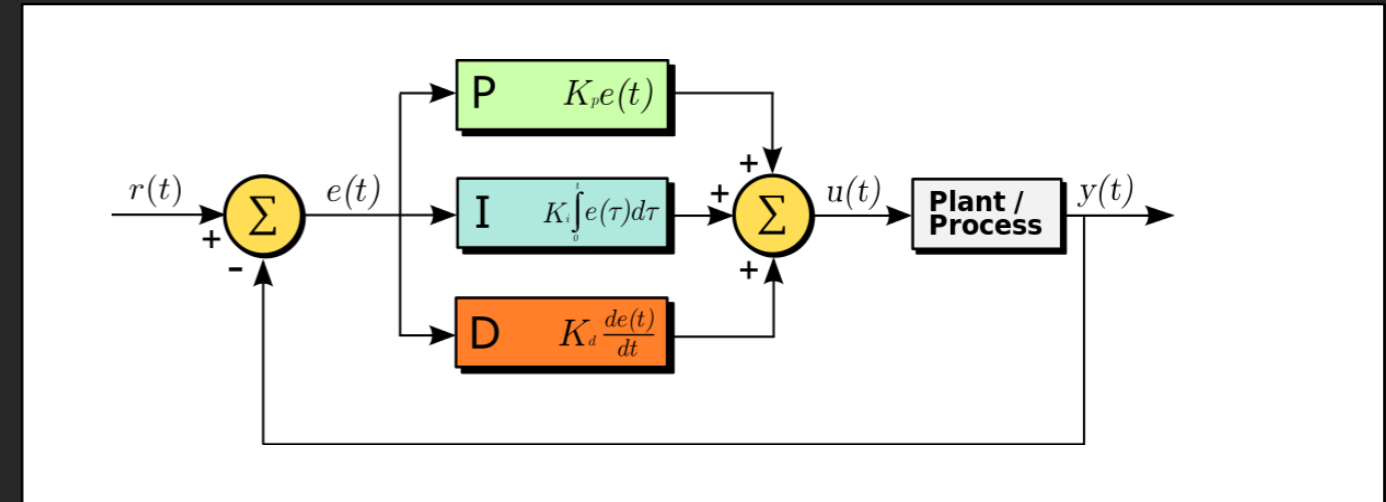
OK

# PID Controller



<https://commons.wikimedia.org/wiki/File:PID.svg>

# PID Controller



<https://commons.wikimedia.org/wiki/File:PID.svg>

PID inputs:

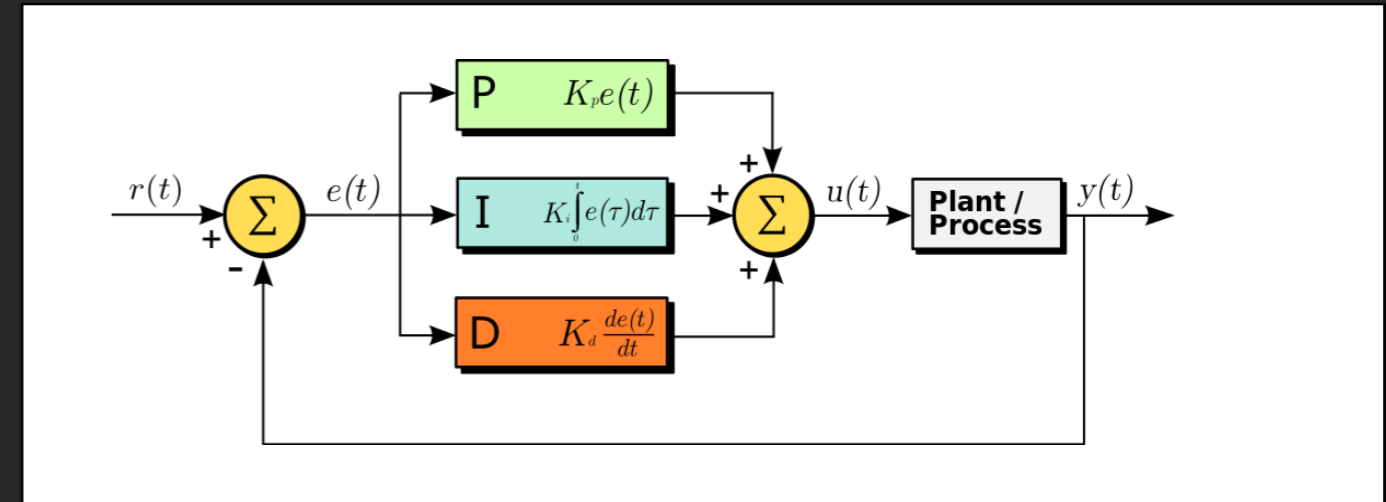
Consumed Capacity

Provisioned Capacity

Throttling Rate

Current Multiplier

# PID Controller



<https://commons.wikimedia.org/wiki/File:PID.svg>

PID inputs:

- Consumed Capacity
- Provisioned Capacity
- Throttling Rate
- Current Multiplier

PID output:

New Multiplier

# Too Much Allowed

150 RCU over provisioned

0x12A8	236294	{ name:"Sara", city:"Tampa", ...}
0x3391	445104	{ name:"James", city:"Miami", ...}
0x6134	333363	{ name:"Betty", city:"Madison", ...}

150 RCU admitted

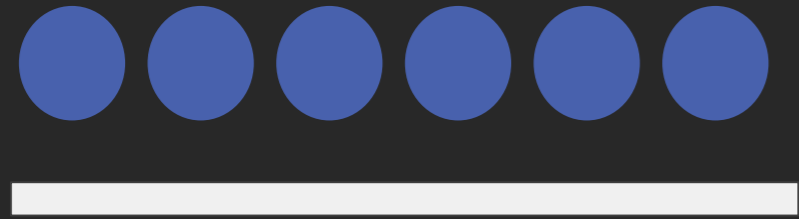
0x9531	145783	{ name:"Bob", city:"London", ...}
0xB082	643145	{ name:"Val", city:"Seattle", ...}

150 RCU admitted

0xEA8A	723342	{ name:"Jeff", city:"Toledo", ...}
0xF355	523422	{ name:"Alex", city:"London", ...}

150 RCU admitted

# Partition Bursting



Refilled at RCU rate  
100 tokens/second



Adaptive Capacity

~~Multiplier = 1.5~~

Multiplier = 1.0

# Throttling

0x12A8	236294	{ name:"Sara", city:"Tampa", ...}
0x3391	445104	{ name:"James", city:"Miami", ...}
0x6134	333363	{ name:"Betty", city:"Madison", ...}

100 RCU admitted **50 Throttled**

0x9531	145783	{ name:"Bob", city:"London", ...}
0xB082	643145	{ name:"Val", city:"Seattle", ...}

100 RCU admitted **50 Throttled**

0xEA8A	723342	{ name:"Jeff", city:"Toledo", ...}
0xF355	523422	{ name:"Alex", city:"London", ...}

100 RCU admitted **50 Throttled**

# Auto scaling

customer [Close](#)

[Overview](#) [Items](#) [Metrics](#) [Alarms](#) **Capacity** [Indexes](#) [Global Tables](#) [Backups](#) [Triggers](#) [Access control](#) [Tags](#)

▸ [Scaling activities](#)

---

### Provisioned capacity

	Read capacity units	Write capacity units
Table	<input type="text" value="300"/>	<input type="text" value="5"/>
name-index	<input type="text" value="300"/>	<input type="text" value="5"/>

Estimated cost \$62.87 / month ([Capacity calculator](#))

---

### Auto Scaling

Read capacity  Write capacity

Target utilization  %

Minimum provisioned capacity  units

Maximum provisioned capacity  units

Apply same settings to global secondary indexes

---

IAM Role I authorize DynamoDB to scale capacity using the following role:

DynamoDB AutoScaling Service Linked Role

Role Name\*

# Auto scaling

customer [Close](#)

[Overview](#) [Items](#) [Metrics](#) [Alarms](#) **Capacity** [Indexes](#) [Global Tables](#) [Backups](#) [Triggers](#) [Access control](#) [Tags](#)

▸ [Scaling activities](#)

---

### Provisioned capacity

	Read capacity units	Write capacity units
Table	<input type="text" value="300"/>	<input type="text" value="5"/>
name-index	<input type="text" value="300"/>	<input type="text" value="5"/>

Estimated cost \$62.87 / month ([Capacity calculator](#))

---

### Auto Scaling

Read capacity  Write capacity

Target utilization  %

Minimum provisioned capacity  units

Maximum provisioned capacity  units

Apply same settings to global secondary indexes

---

IAM Role I authorize DynamoDB to scale capacity using the following role:

DynamoDB AutoScaling Service Linked Role

Role Name\*

# Auto scaling

300 → 640 provisioned

70% of 640 ~ 450

0x12A8	236294	{ name:"Sara", city:"Tampa", ...}
0x3391	445104	{ name:"James", city:"Miami", ...}
0x6134	333363	{ name:"Betty", city:"Madison", ...}

150 RCU admitted

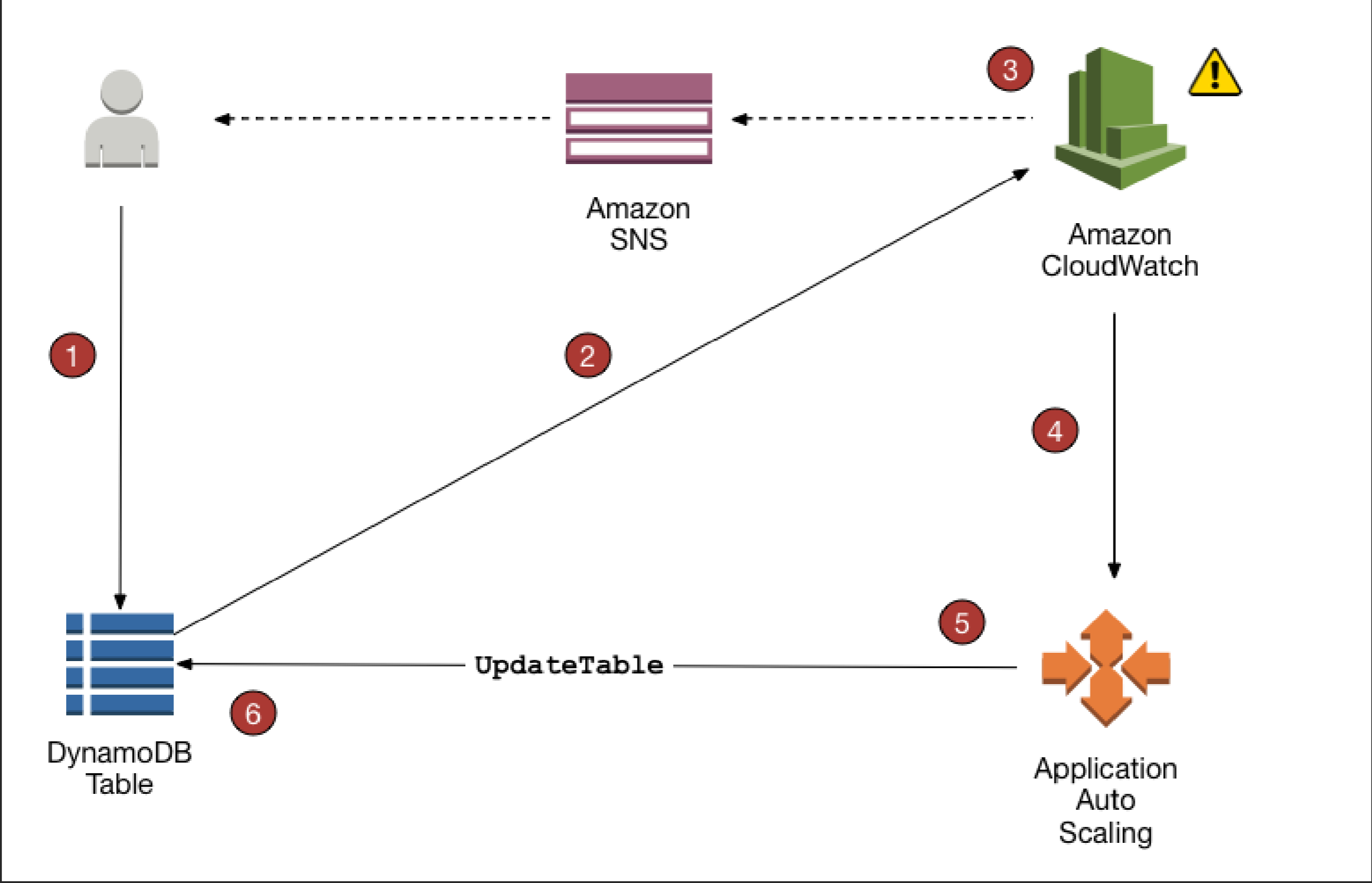
0x9531	145783	{ name:"Bob", city:"London", ...}
0xB082	643145	{ name:"Val", city:"Seattle", ...}

150 RCU admitted

0xEA8A	723342	{ name:"Jeff", city:"Toledo", ...}
0xF355	523422	{ name:"Alex", city:"London", ...}

150 RCU admitted

# Auto scaling



# Auto scaling

450 → 43 provisioned

70% of 43 ~ 30

0x12A8	236294	{ name:"Sara", city:"Tampa", ...}
0x3391	445104	{ name:"James", city:"Miami", ...}
0x6134	333363	{ name:"Betty", city:"Madison", ...}

10 RCU Consumed

0x9531	145783	{ name:"Bob", city:"London", ...}
0xB082	643145	{ name:"Val", city:"Seattle", ...}

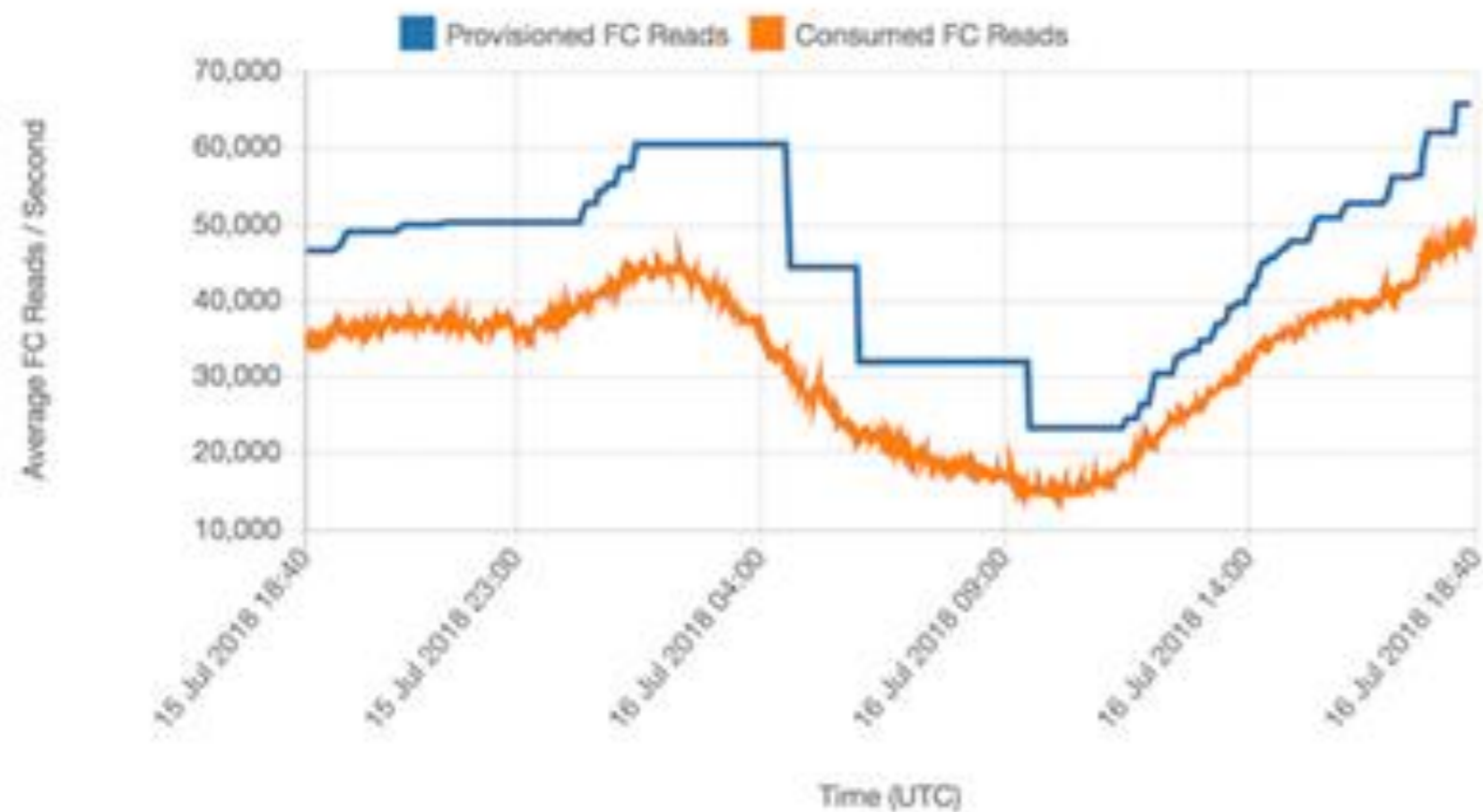
10 RCU Consumed

0xEA8A	723342	{ name:"Jeff", city:"Toledo", ...}
0xF355	523422	{ name:"Alex", city:"London", ...}

10 RCU Consumed

# Actual Auto Scaled Table

## Reads



## Writes



# Questions

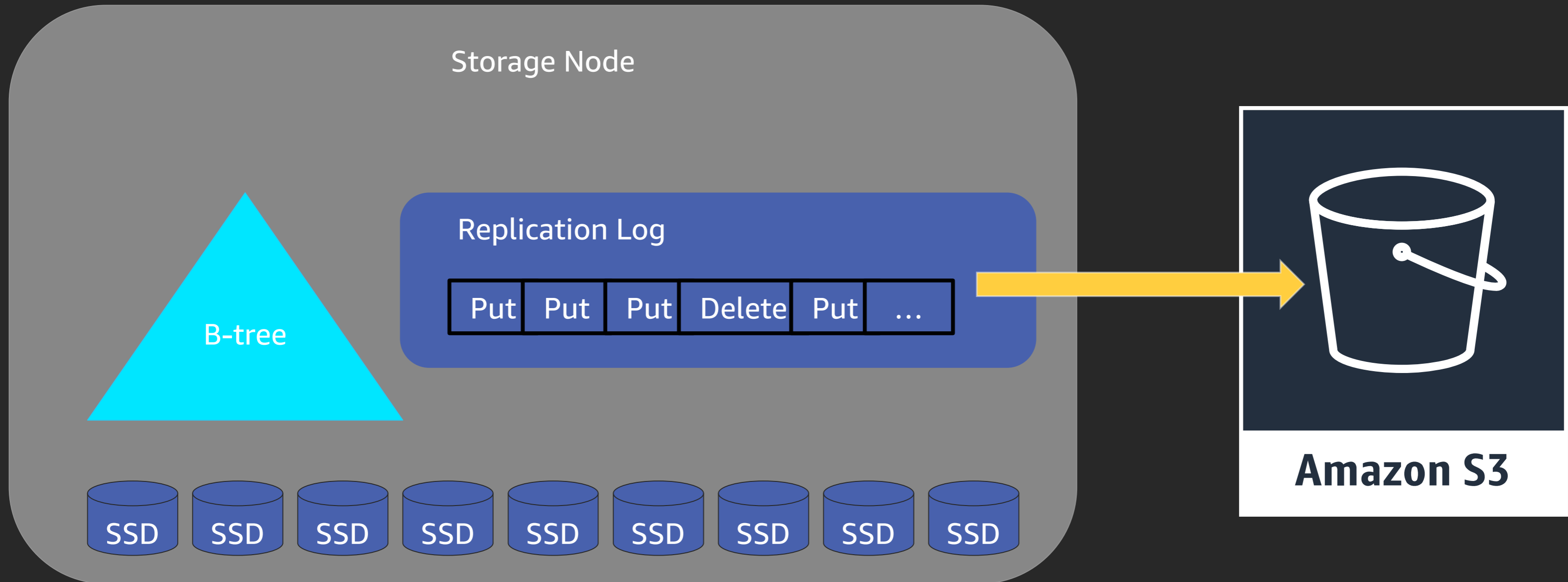


# Backup / Restore

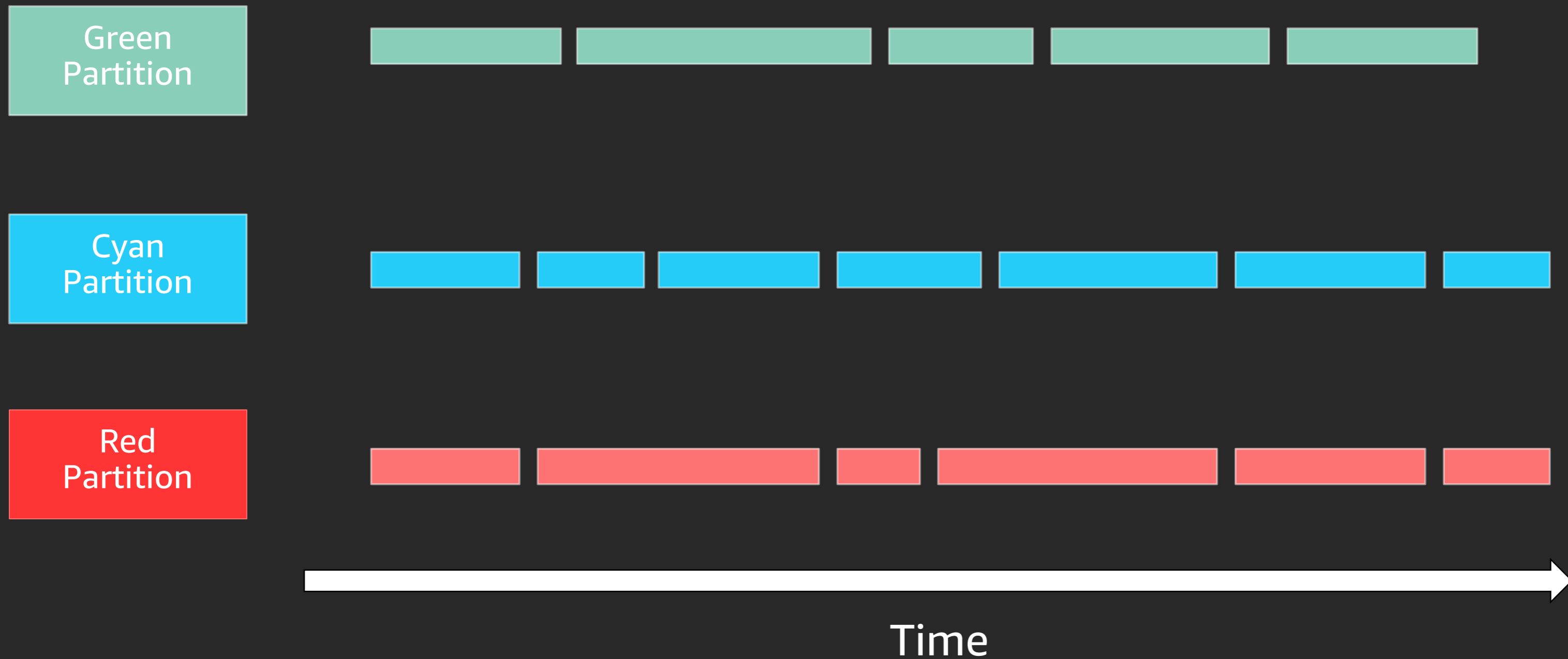
# Backup and Restore

- Point in Time
- On-demand Backups

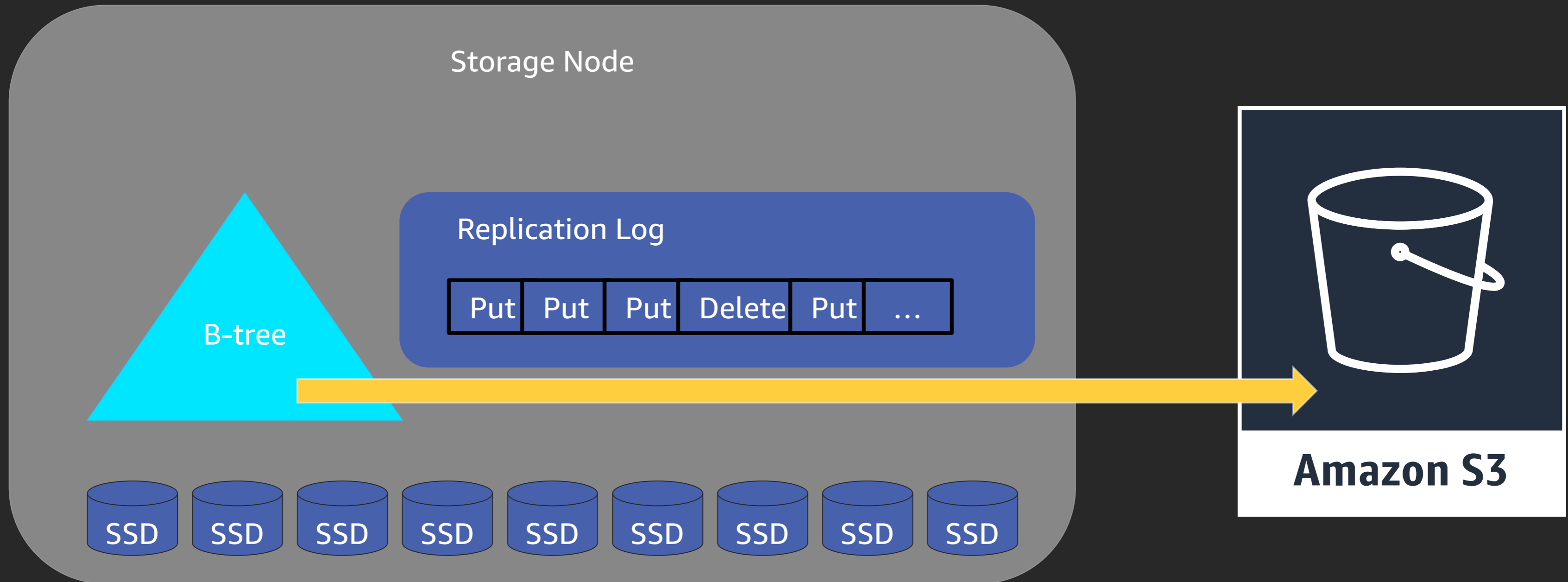
# Replication Log to S3



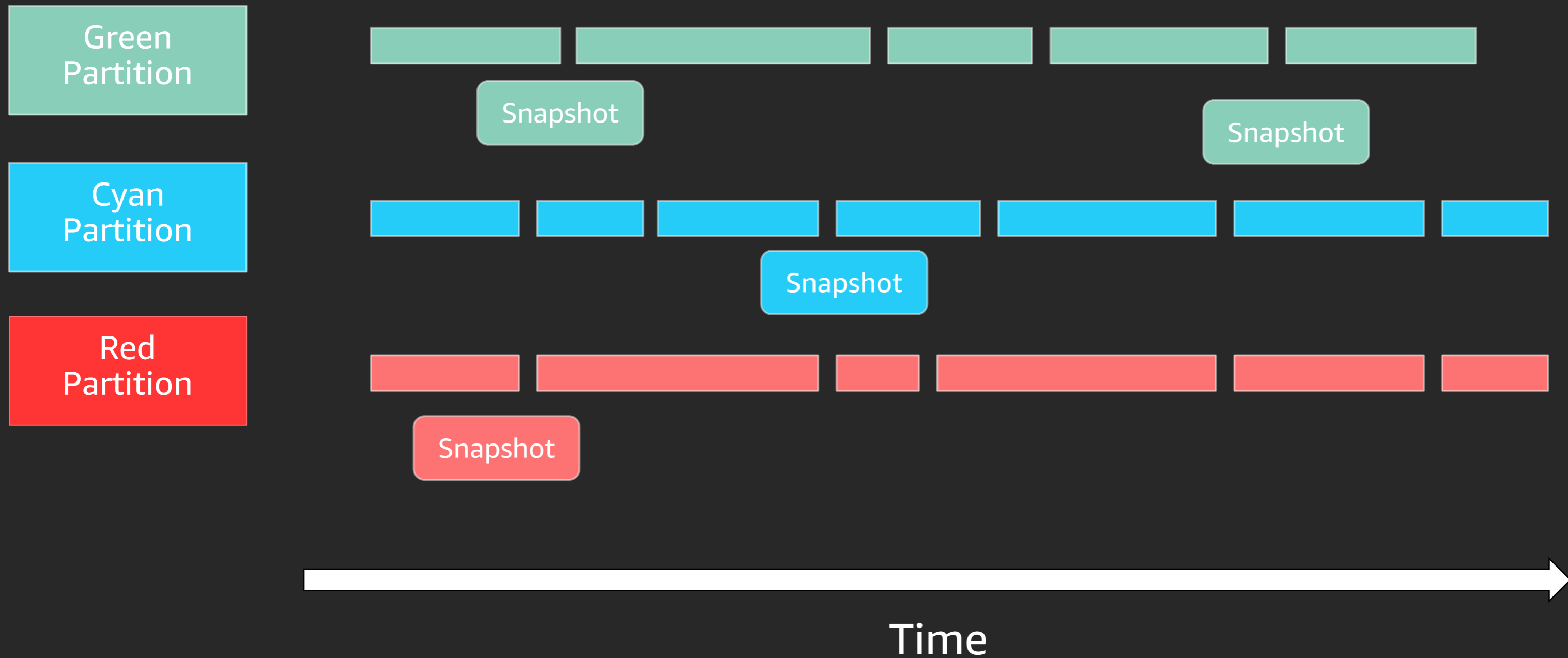
# Replication Log to S3



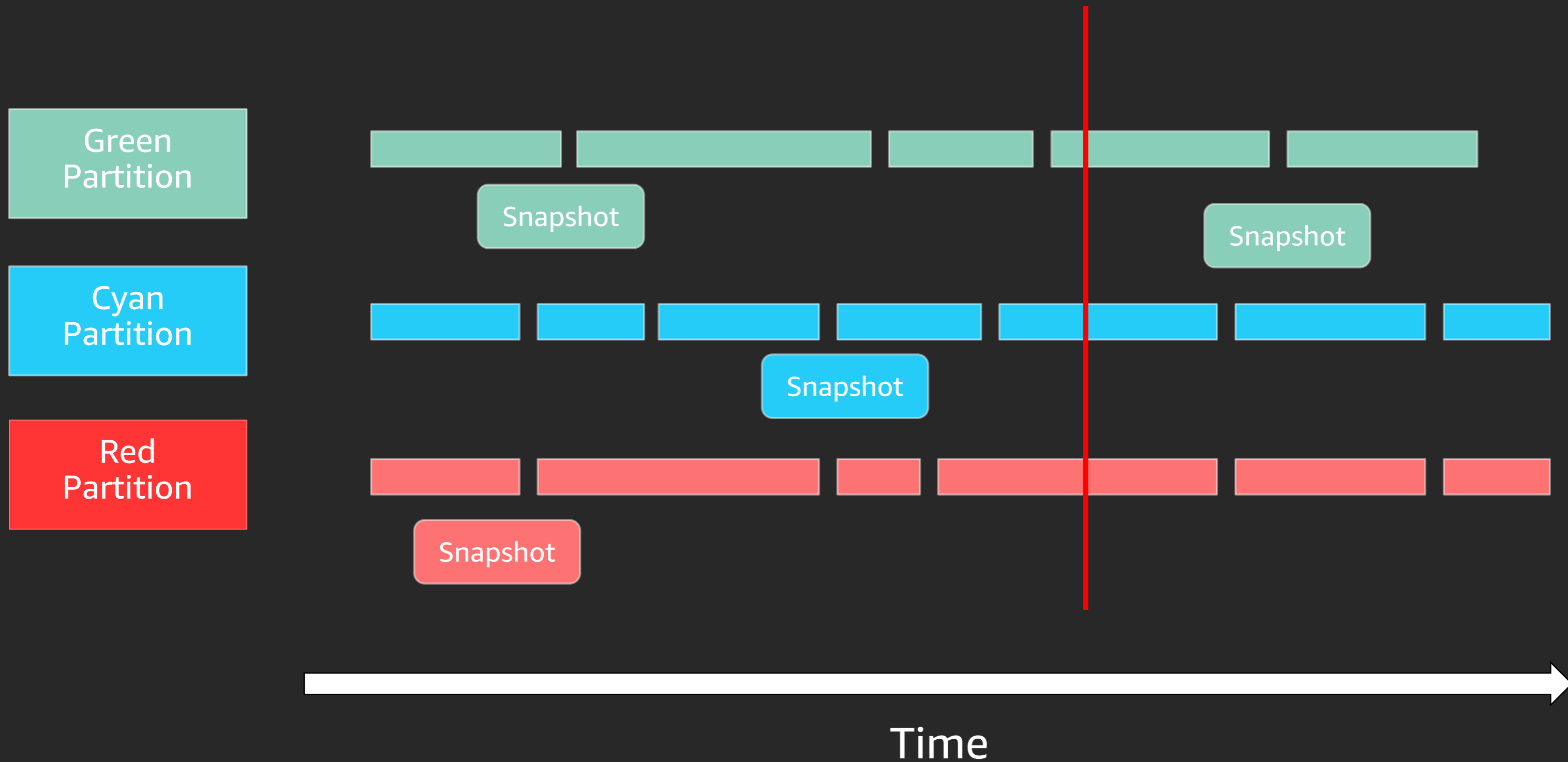
# "Snapshot" the B-tree to S3



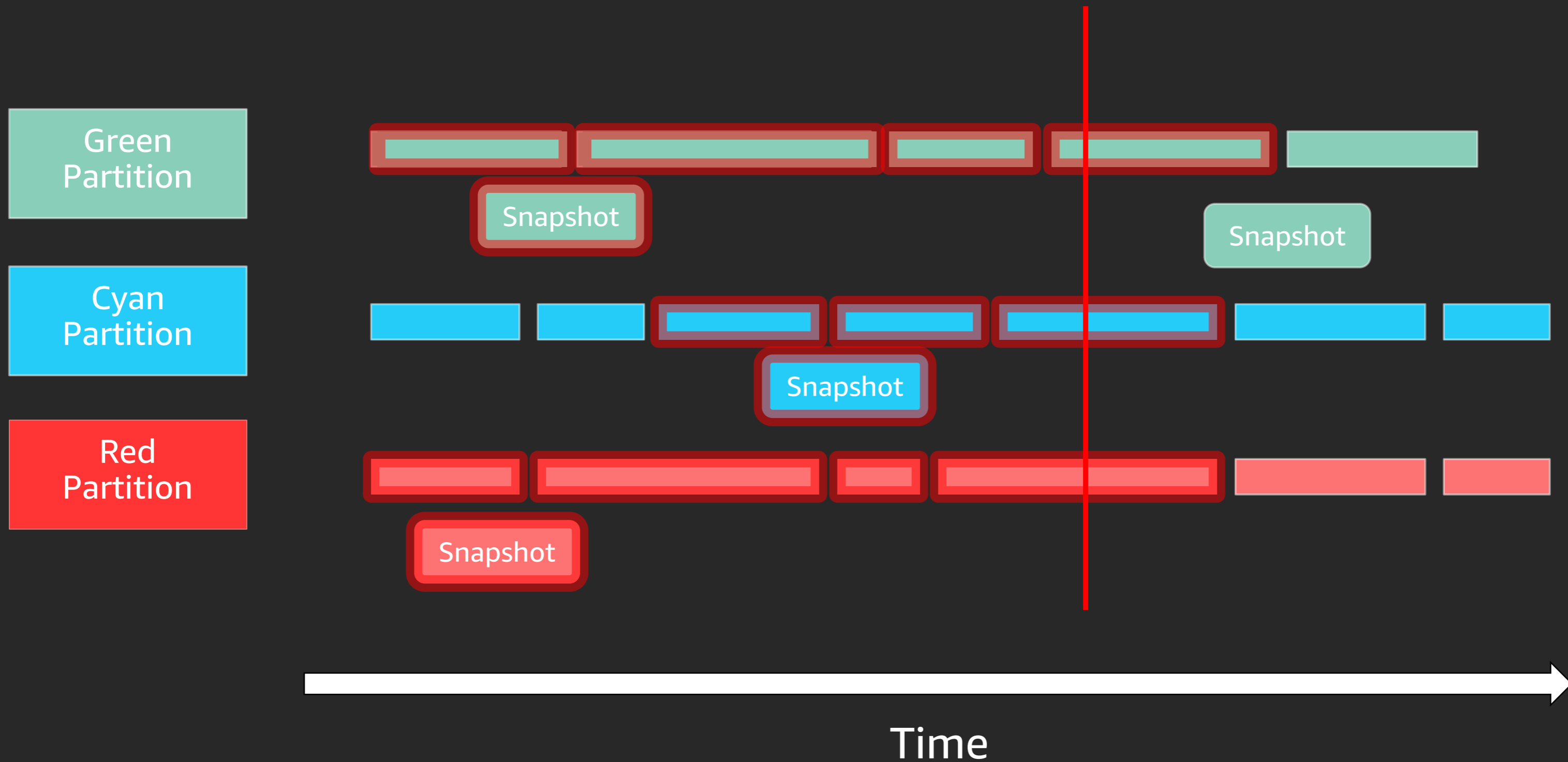
# "Snapshot" the B-tree to S3



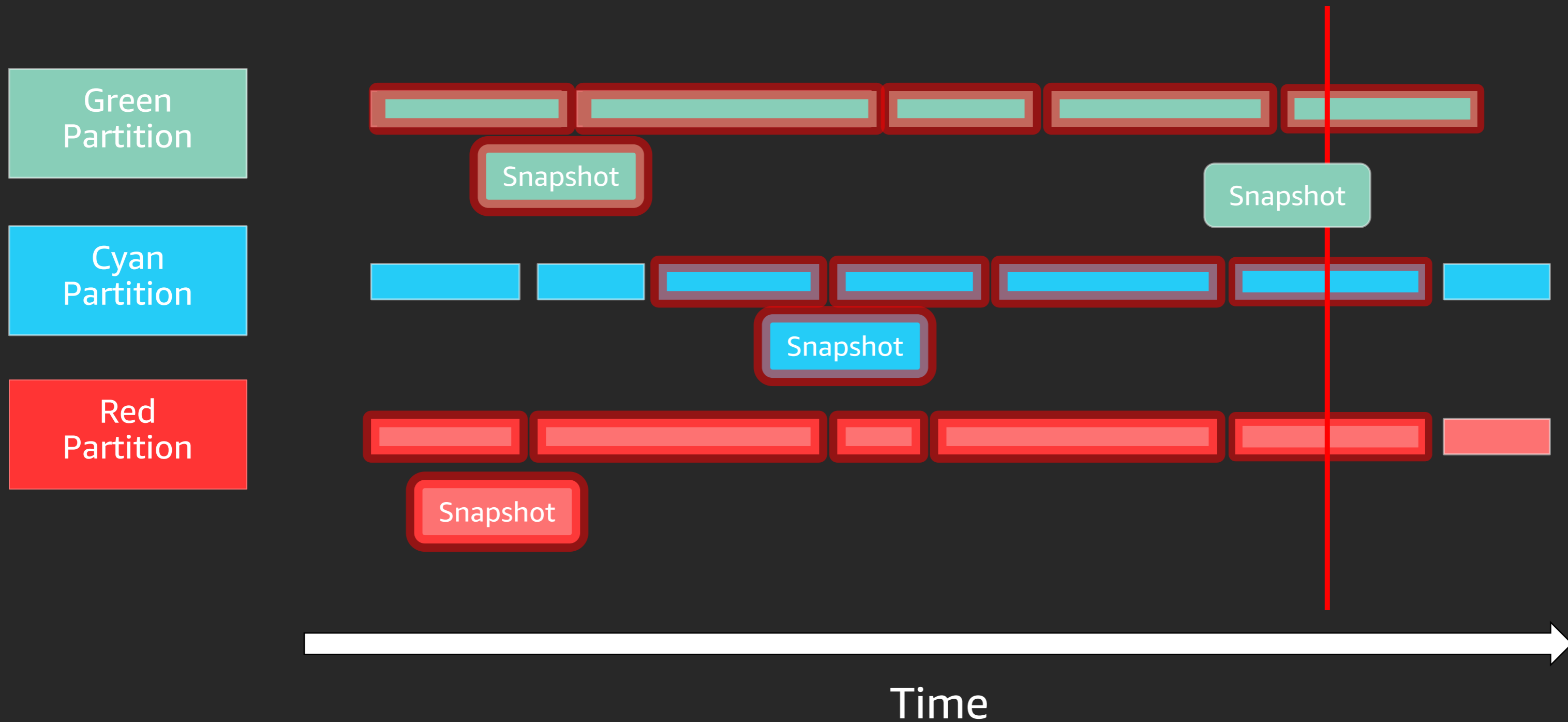
# Restore to a Point in Time



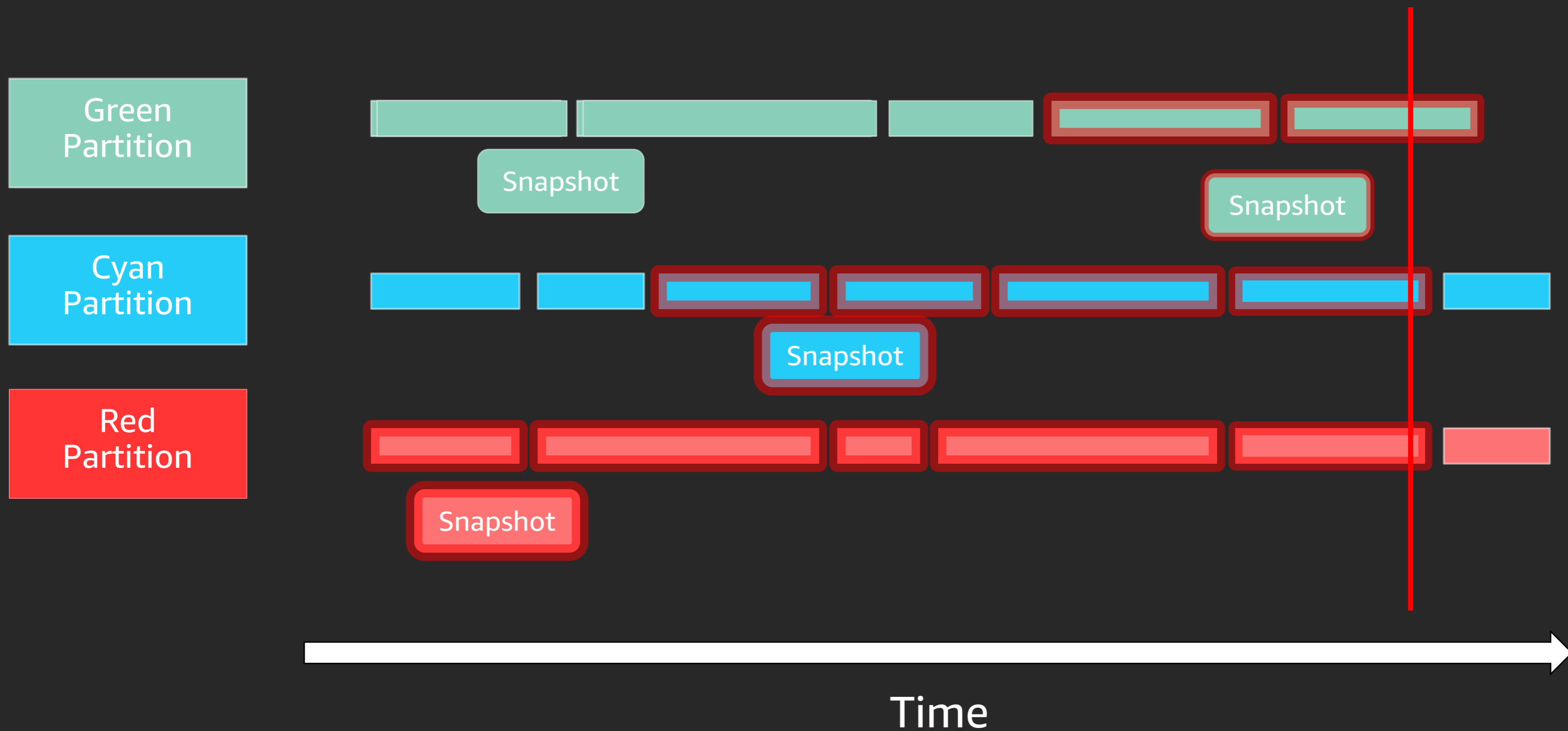
# Restore to a Point in Time



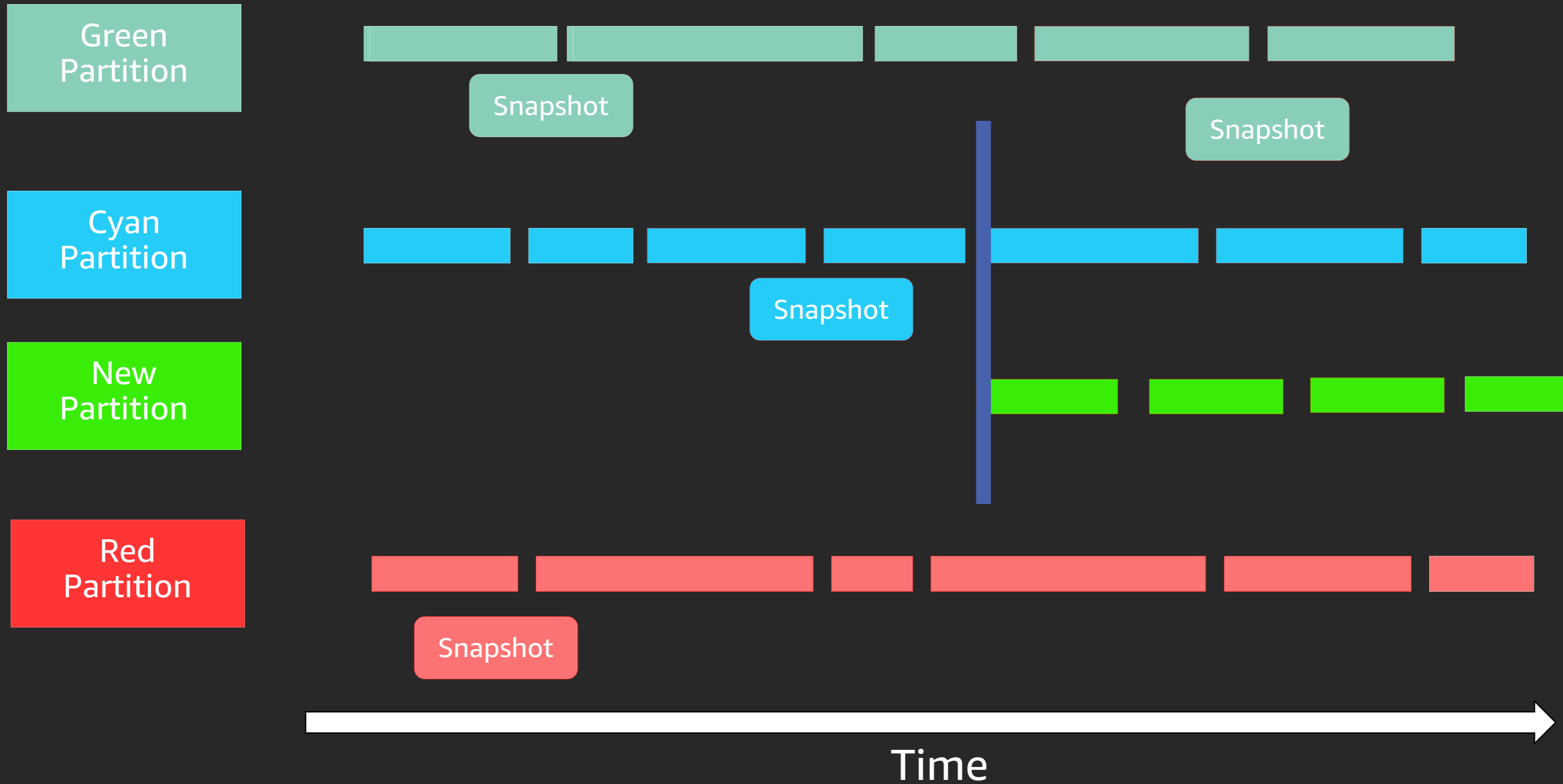
# Restore to a Point in Time



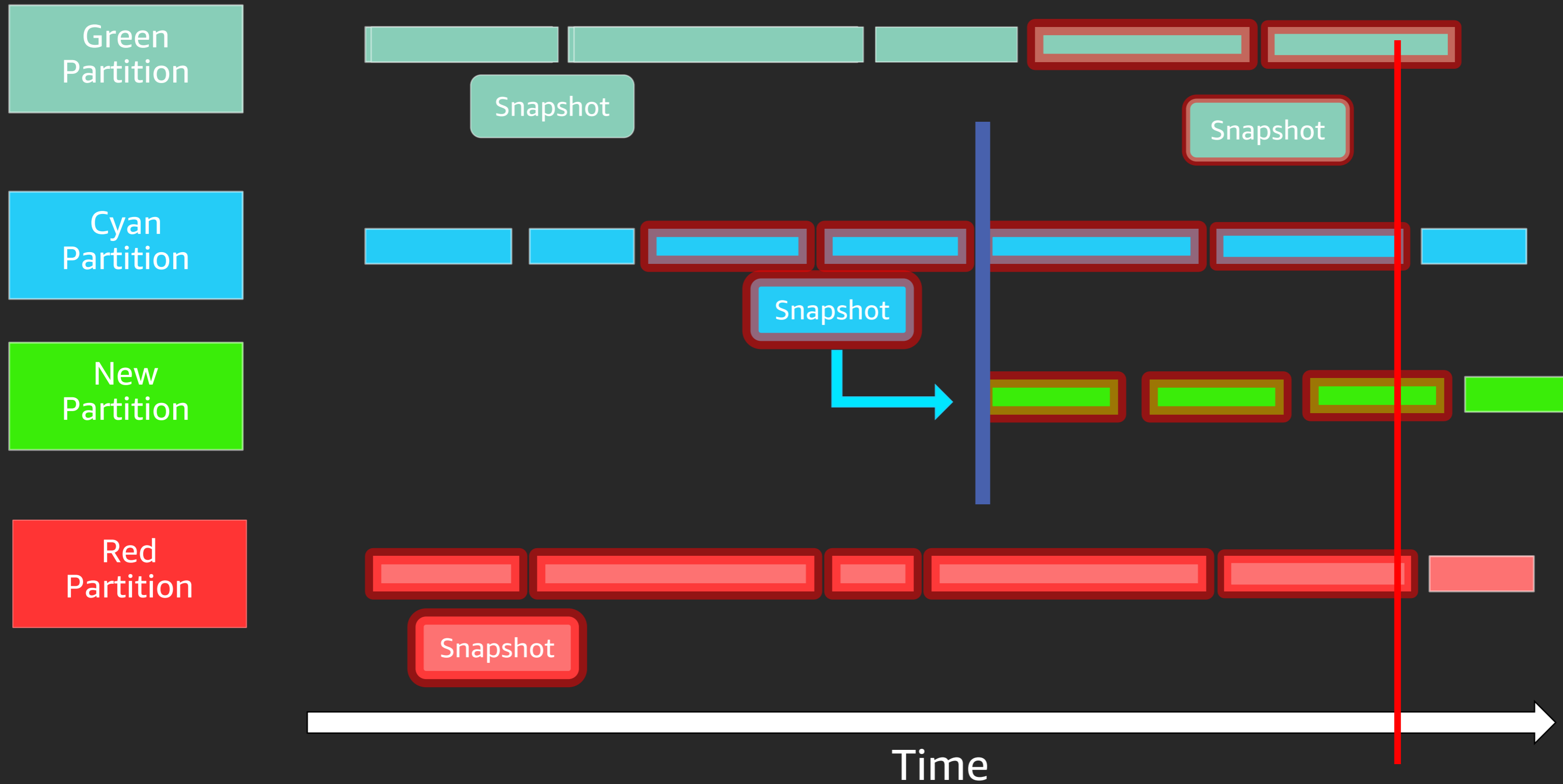
# Restore to a Point in Time



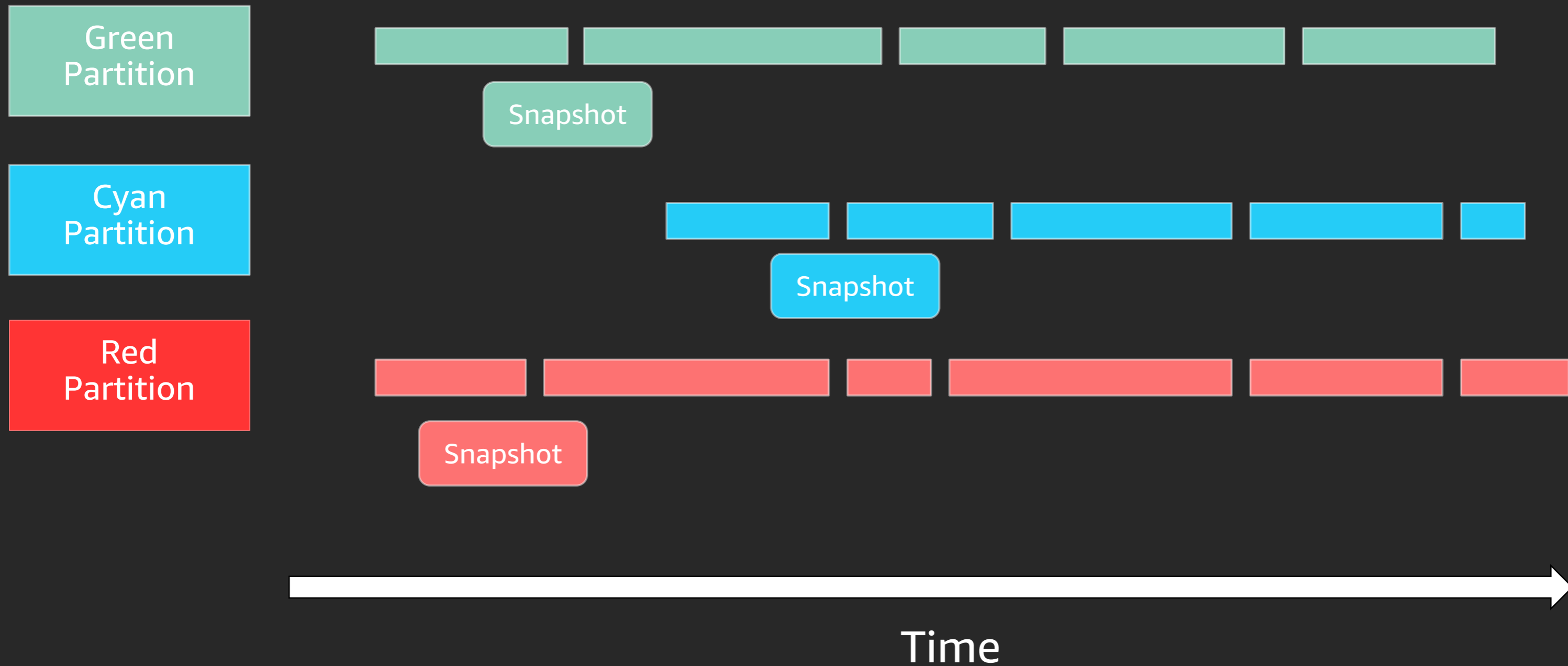
# Partition Splits



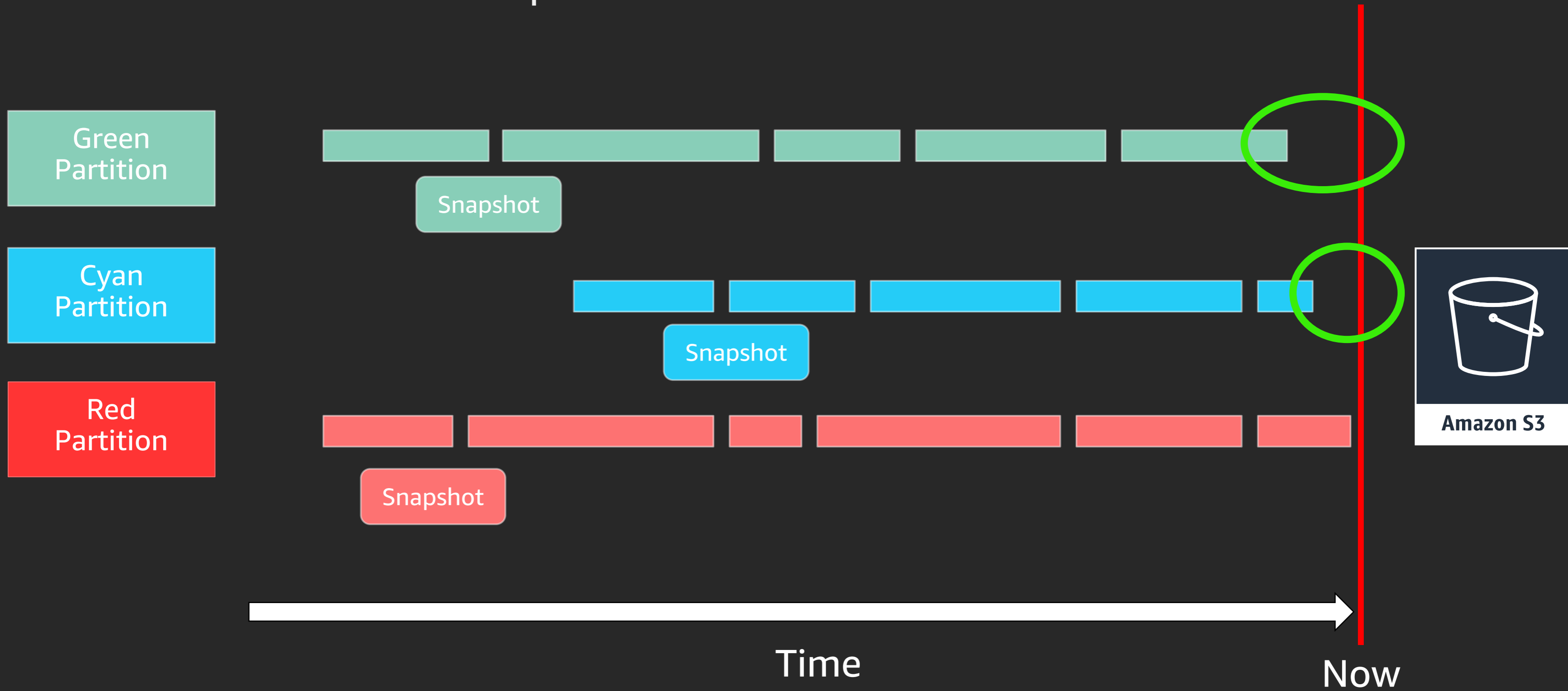
# Restore to a Point in Time



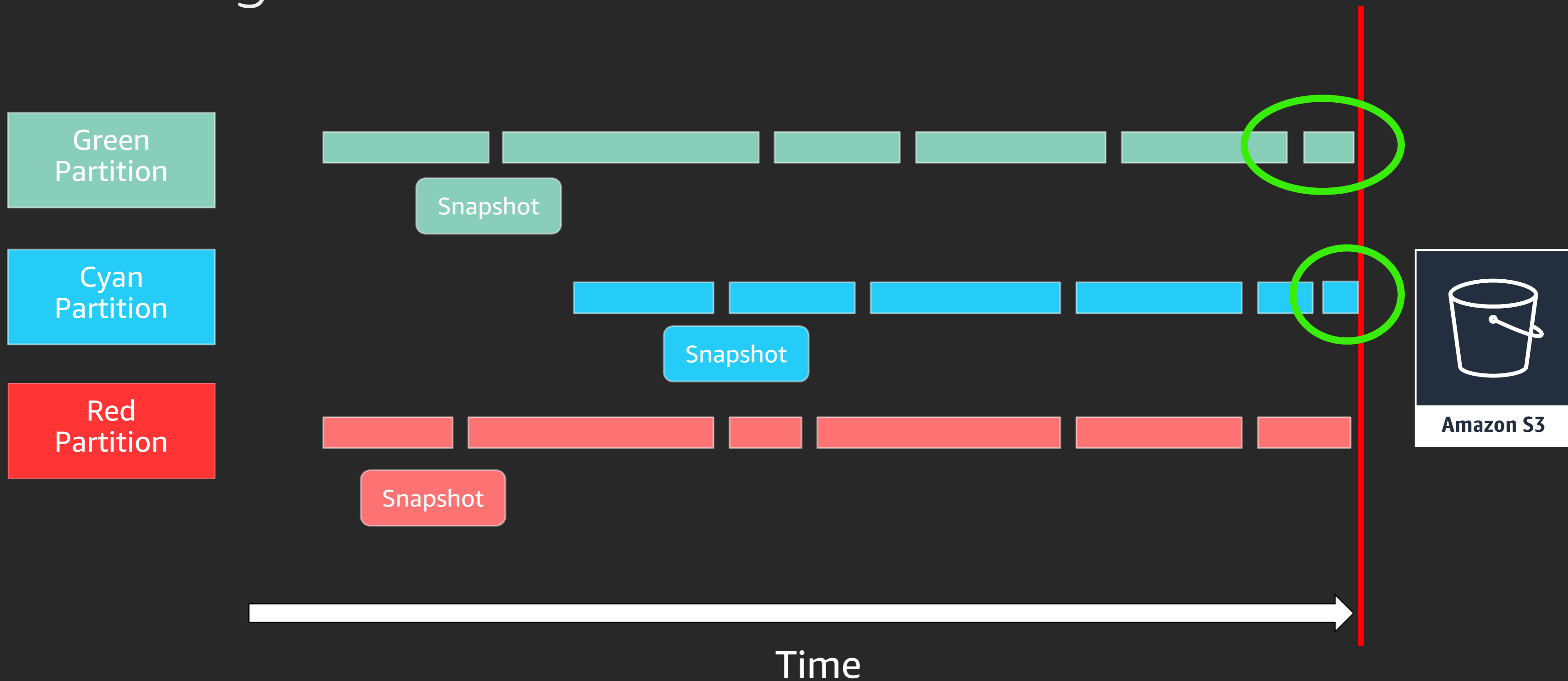
# On-demand Backups



# On-demand Backups



# Write Logs to S3



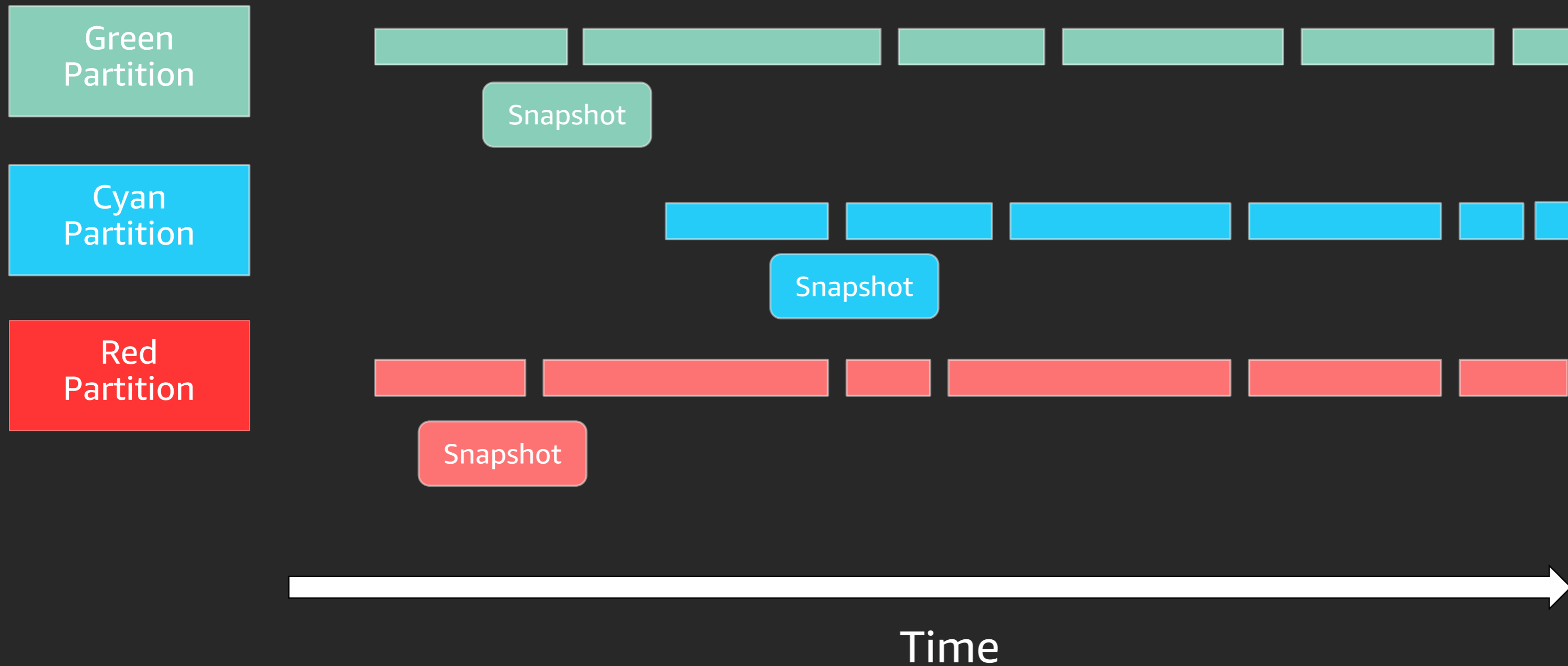
# On-demand backup

The screenshot shows the AWS Management Console interface for DynamoDB Backups. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and user information (jaso, Oregon, Support). The left sidebar lists navigation options: DynamoDB, Dashboard, Tables, Backups (highlighted), Reserved capacity, Preferences, DAX, Dashboard, Clusters, Subnet groups, Parameter groups, and Events.

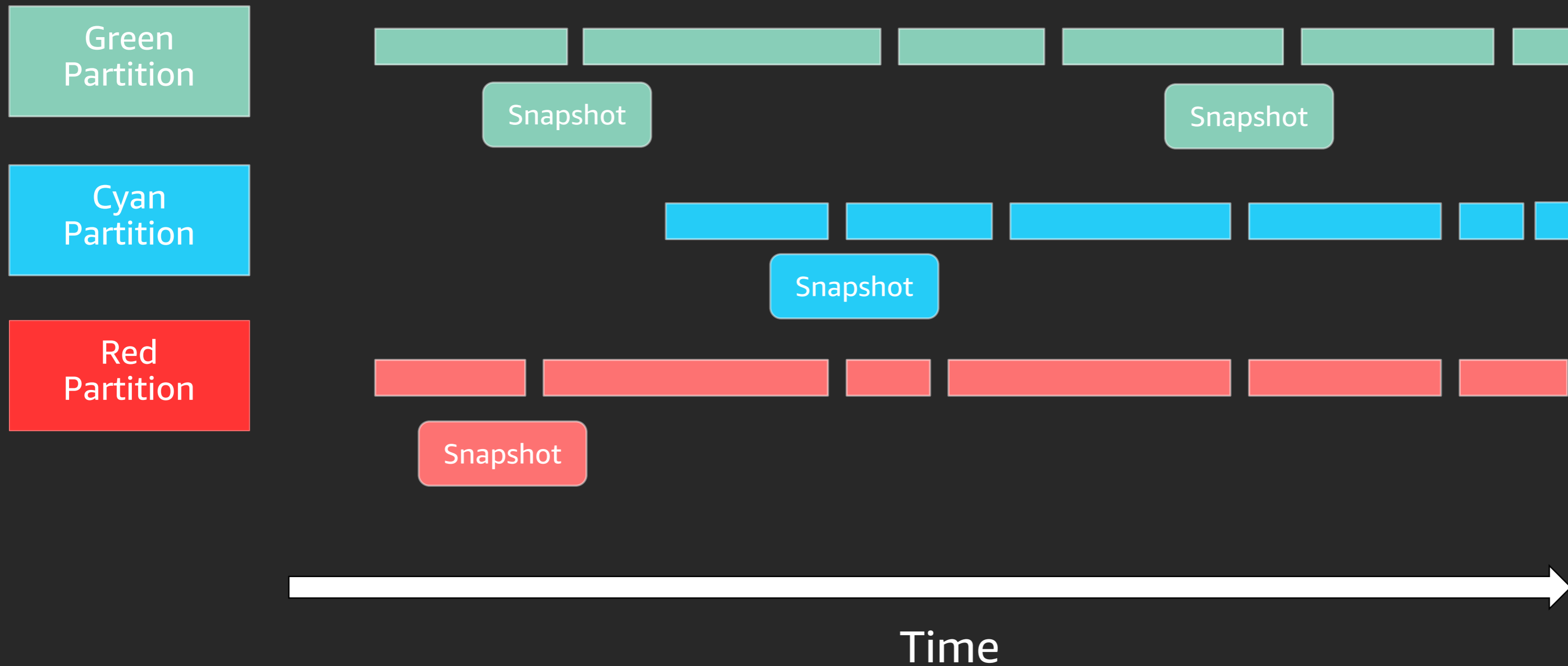
The main content area features a 'Create backup' button (highlighted in blue), 'Restore backup', and 'Delete backup' buttons. Below these are filters: a search box for 'Filter by backup or table name', a 'Time Range' dropdown set to 'Last 30 Days', and a 'Backup type' dropdown set to 'All backups'. A pagination indicator shows '1 to 1 of 1 Backups'.

Backup name	Status	Table	Creation time	Size	Backup type	Expiration date	Backup ARN
SaturdayBackup	Available	customer	November 24, 2018 at 5:46:39 PM UTC-8	59.00 bytes	User	-	arn:aws:dynamodb:us-west-2:241781661306:table/cu

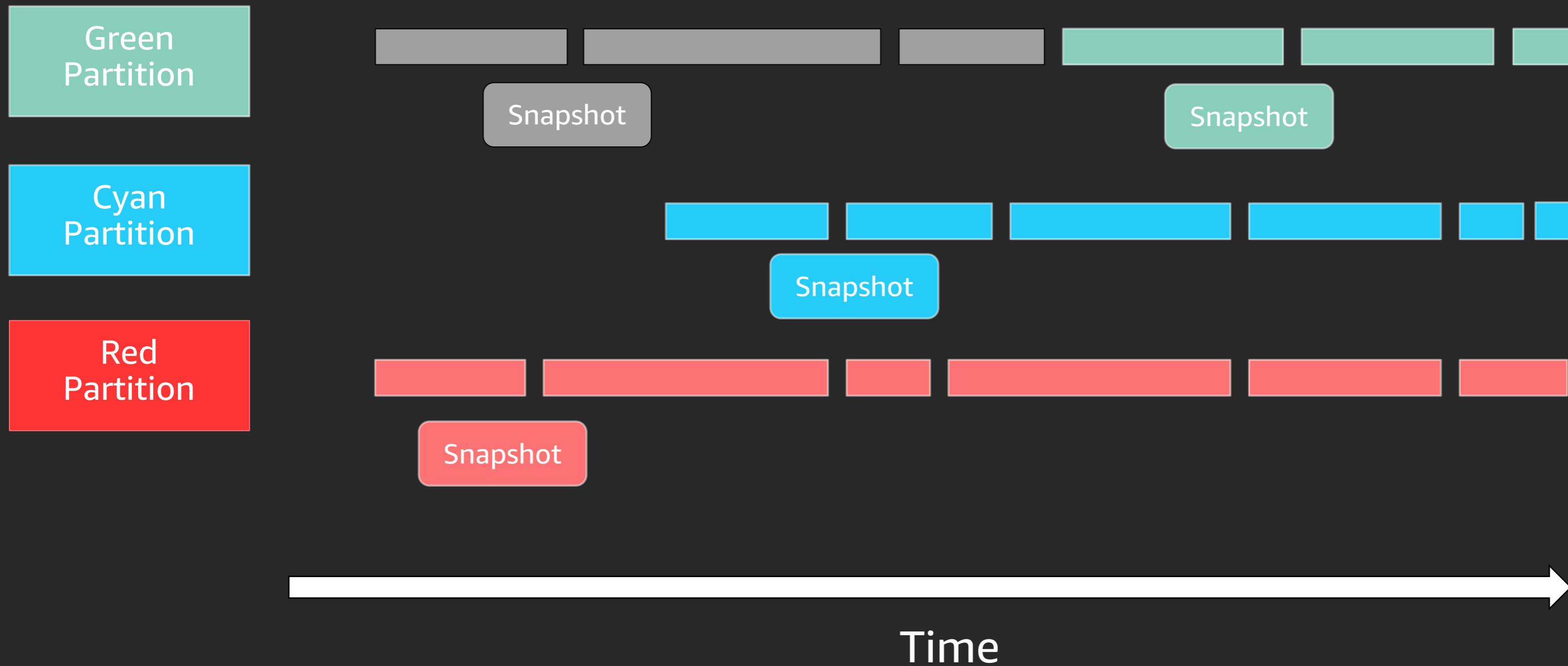
# Without PITR enabled



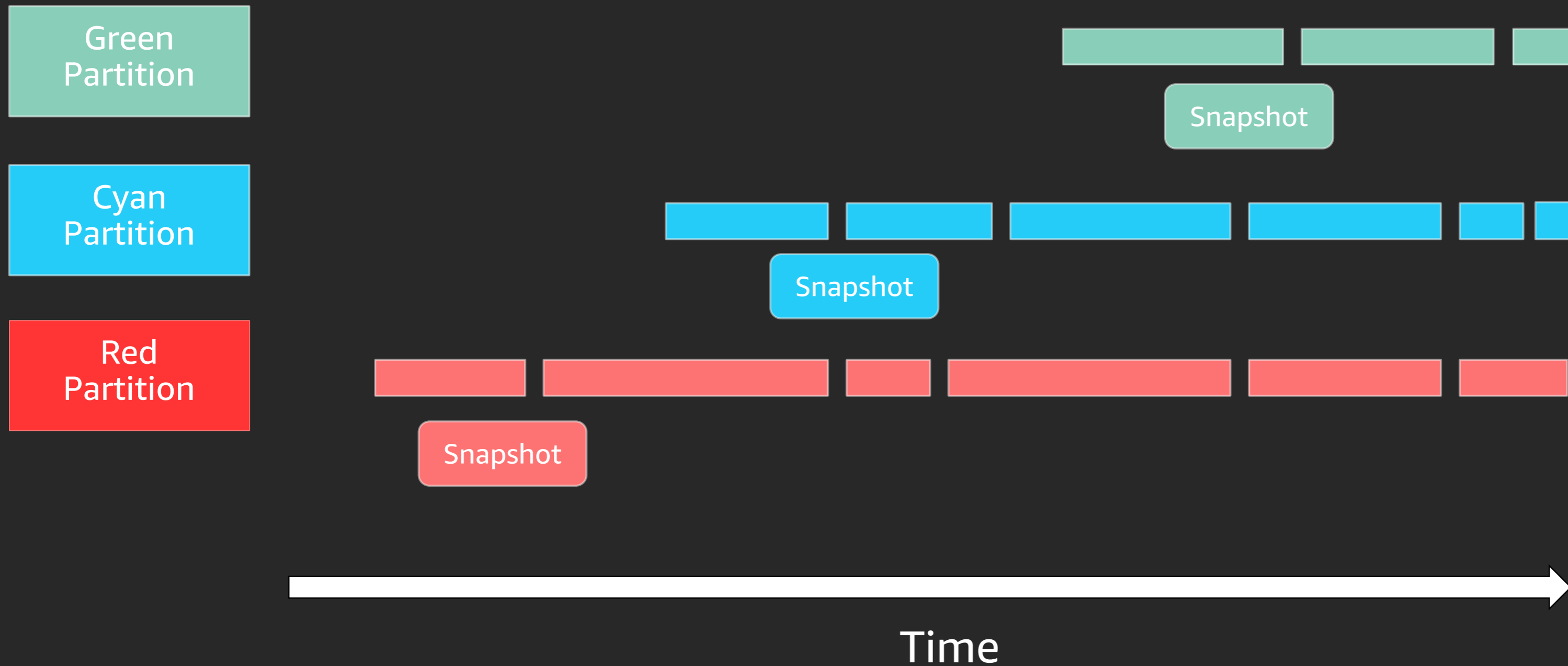
# Without PITR enabled



# Without PITR enabled



# Without PITR enabled



# With PITR enabled

Overview

Items

Metrics

Alarms

Capacity

Indexes

Global Tables

Backups

Triggers

Access control

Tags

## Point-in-time Recovery

DynamoDB maintains continuous backups of your table for the last 35 days. [Learn more](#)

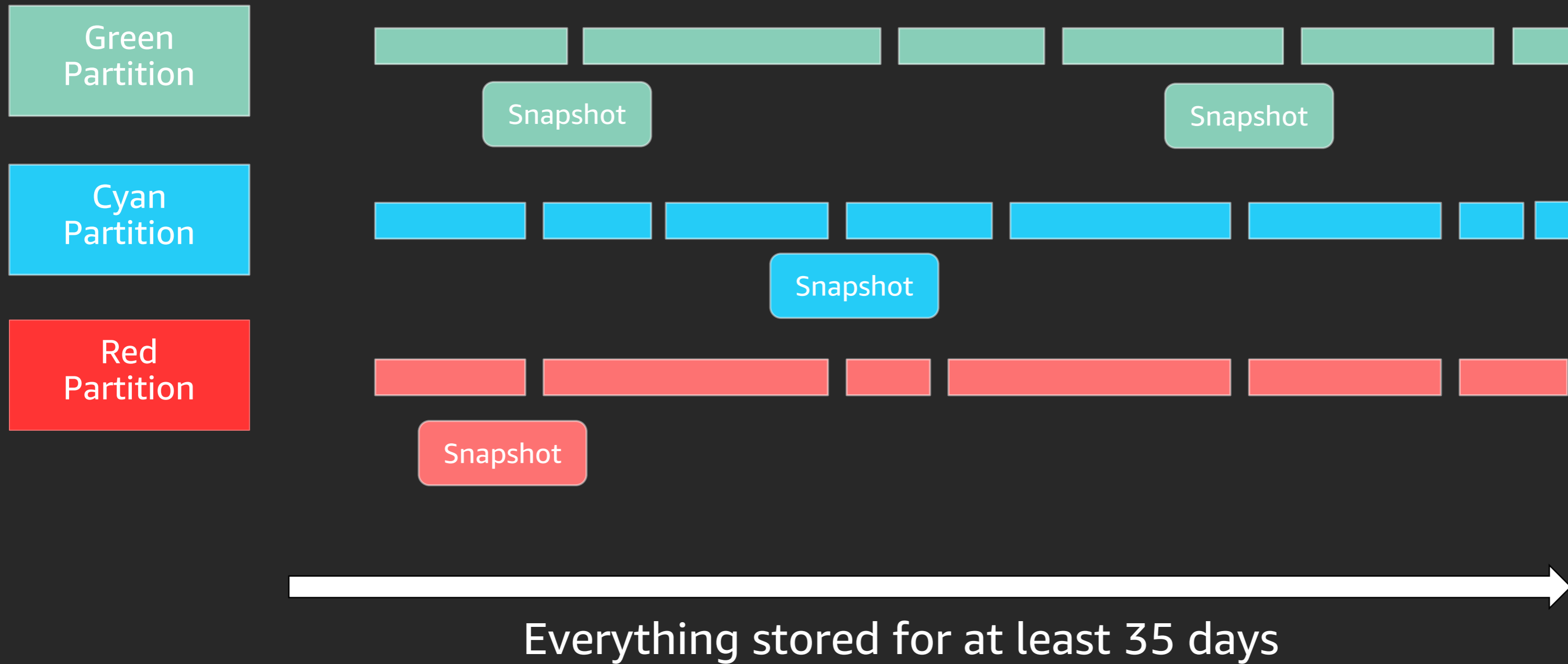
Status	ENABLED <a href="#">Disable</a>
Earliest restore date	November 24, 2018 at 5:59:04 PM UTC-8
Latest restore date	November 24, 2018 at 6:07:08 PM UTC-8

Restore to point-in-time

## On-Demand Backup and Restore

You can create and restore a complete backup of your DynamoDB table data and its settings at any time. [Learn more](#)

# With PITR enabled



# Questions



# Thank you!

**James Christopher Sorenson III aka jaso**

[jaso@amazon.com](mailto:jaso@amazon.com)



Please complete the session survey in the mobile app.