AWS re:Invent

NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

DAT202

Top 5 patterns to unblock complex relational database migrations

Jatin Singh (He/him)
Principal Database Solutions Architect
AWS

Abhinav Sarin (He/him)
Senior Database Solutions Architect
AWS

Agenda

Introduction

Challenges with complex migrations

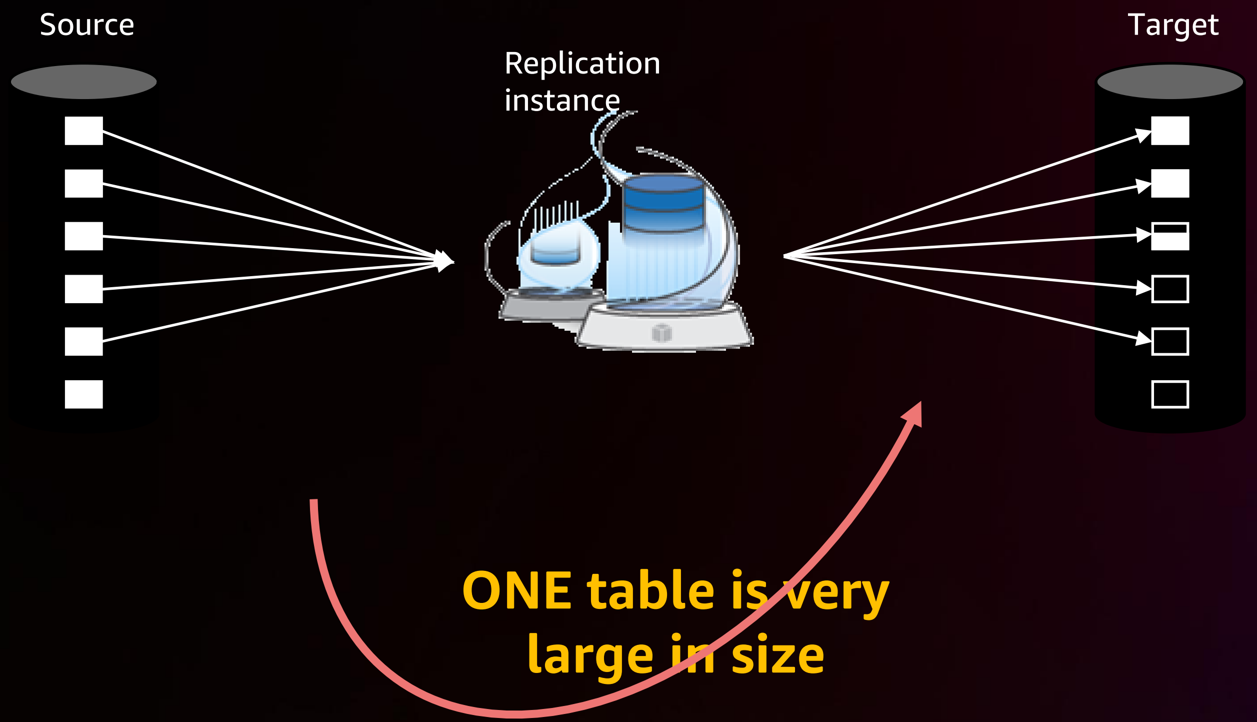
Design to address complex migration

Q&A

Summary

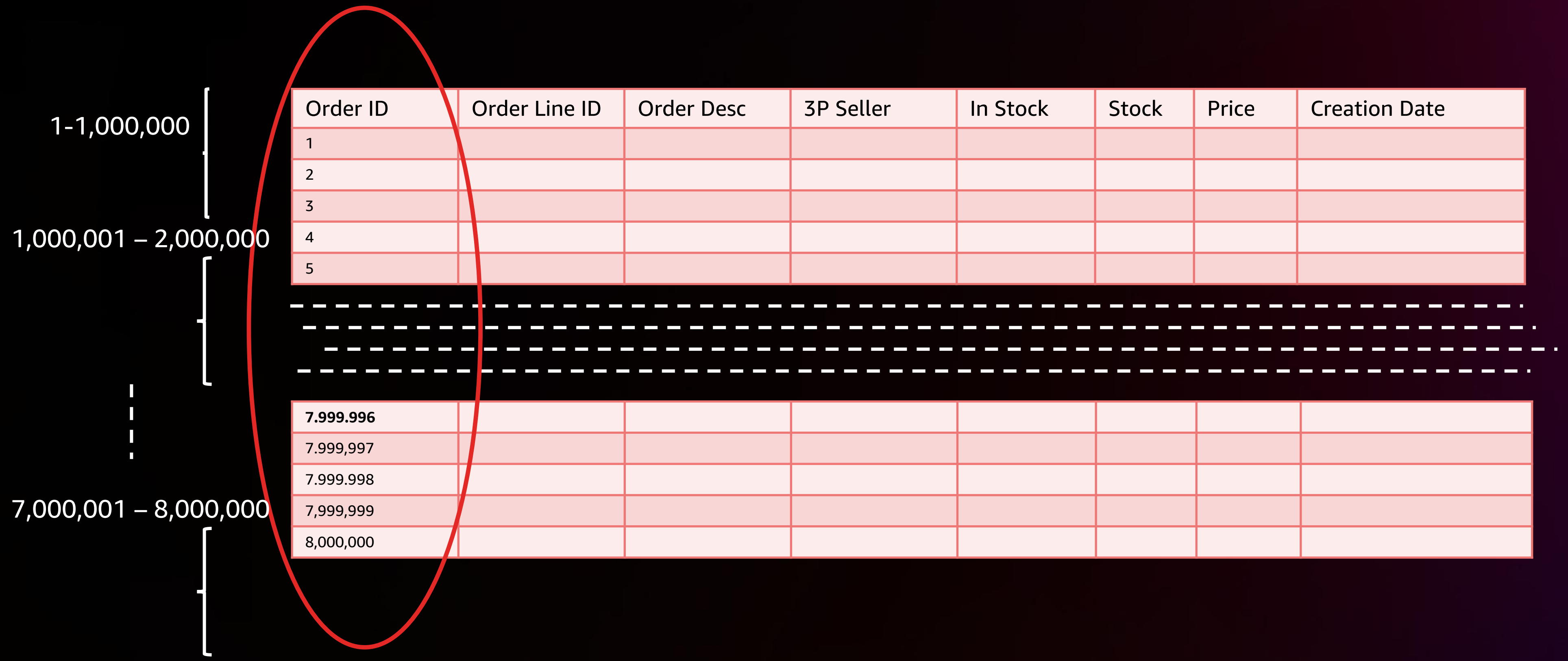
Pattern – 1
Migrating Large Dataset

Problem Statement



ONE table is very large in size

Using Row Filtering in DMS



Order ID	Order Line ID	Order Desc	3P Seller	In Stock	Stock	Price	Creation Date
1							
2							
3							
4							
5							
7.999.996							
7.999,997							
7.999.998							
7,999,999							
8,000,000							

Leveraging DMS Parallel Full Load - Partitioned Table

```
{
  "rule-type": "table-settings",
  "rule-id": "2",
  "rule-name": "2",
  "object-locator":
     { "schema-name": "HR",
       "table-name": "SALES" },
"parallel-load":
     { "type": "partitions-auto" }
}
```

```
[TASK_MANAGER ] I: Start loading segment #1 of 4 of table 'HR'.'SALES' (Id = 1) by subtask 1.
[SOURCE_UNLOAD ] I: Unload finished for segment #1 of segmented table 'HR'.'SALES' (Id = 1). 1 rows sent.
[TARGET_LOAD ] I: Load finished for segment #1 of segmented table 'HR'.'SALES' (Id = 1). 1 rows received. 0 rows skipped.
[TASK_MANAGER ] I: Load finished for segment #1 of table 'HR'.'SALES' (Id = 1) by subtask 1. 1 records transferred.
[TASK_MANAGER ] I: Start loading segment #4 of 4 of table 'HR'.'SALES' (Id = 1) by subtask 1.
[SOURCE_UNLOAD ] I: Unload finished for segment #4 of segmented table 'HR'.'SALES' (Id = 1). 1 rows sent.
[TARGET_LOAD ] I: Load finished for segment #4 of segmented table 'HR'.'SALES' (Id = 1). 1 rows received. 0 rows skipped.
```

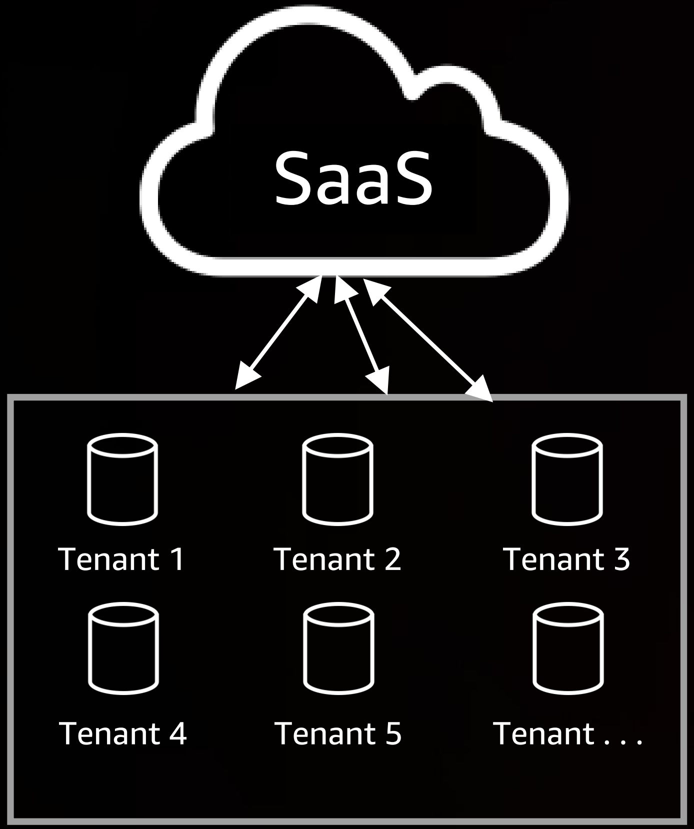
Leveraging DMS Parallel Full Load - Non Partitioned Table

```
{ "rule-type": "table-settings",
  "rule-id": "2",
  "rule-name": "2",
  "object-locator": { "schema-name": "HR", "table-name": "SALES" },
  "parallel-load": { "type": "ranges",
                     "columns": [ "SALES_NO", "REGION" ],
                     "boundaries": [
                        [ "1000", "NORTH" ],
                        [ "3000", ”WEST" ]
                                    ]

}
```

Pattern – 2
Design for database-level multi-tenancy

Multi-tenant applications migration Challenges



Thousands of end users, each with their own database, different traffic pattern

How to scale a tenant independently?

How to do tenant level migration?

How to solve noisy neighbor problem?

How to backup restore a tenant?

Amazon Aurora Serverless v2

On-demand and autoscaling configuration

Automatically scales capacity based on application needs

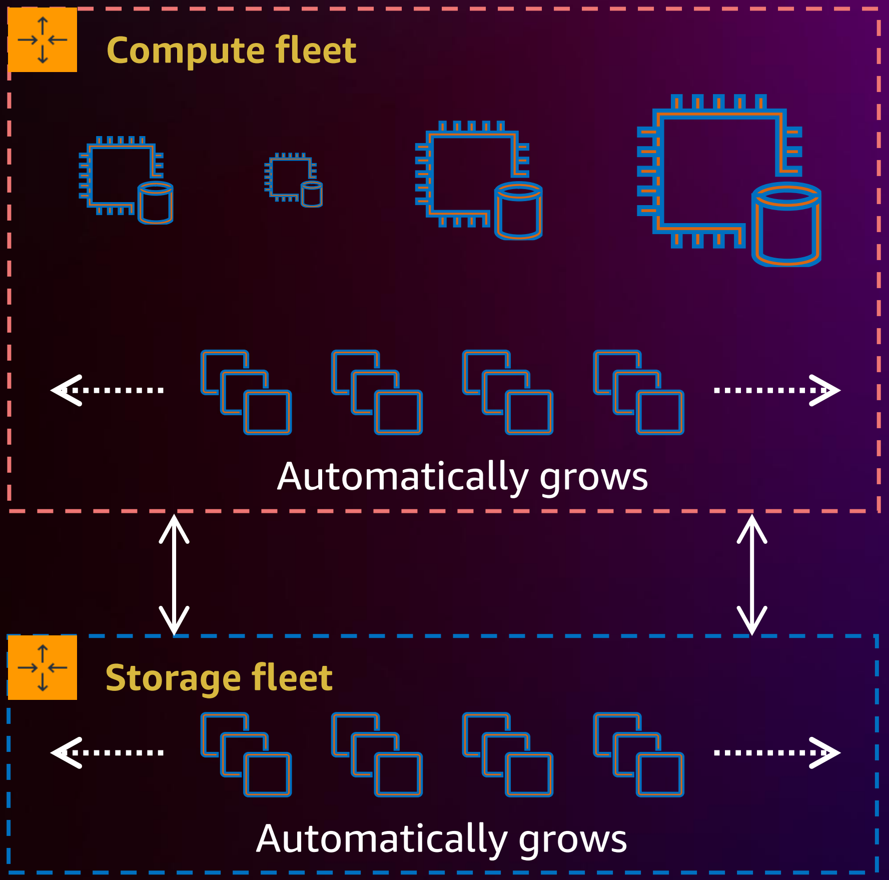
Simple pay-per-use pricing per second

Next version scales instantly to support demanding applications

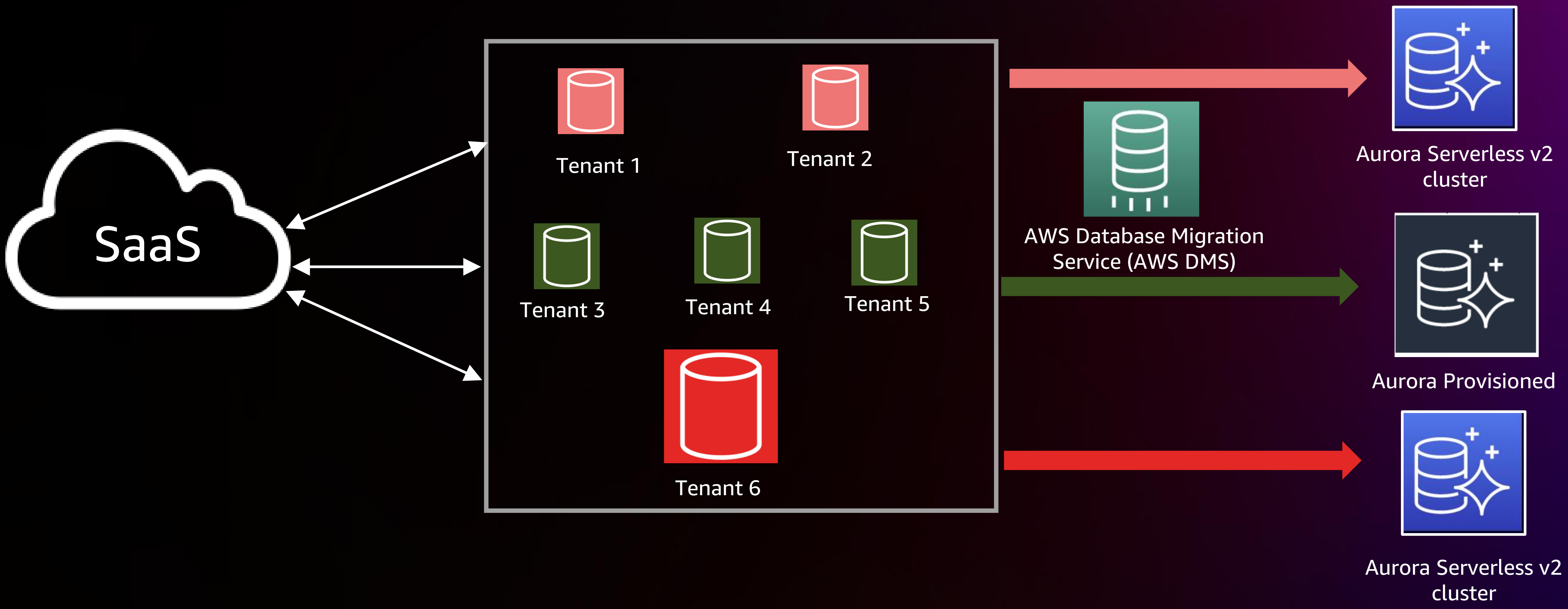
Worry-free database capacity management

Instant, in-place scaling

- Scales in place in under a second by adding more CPU and memory resources
- No impact due to scaling even when running hundreds of thousands of transactions
- Compute fleet continuously monitored and scaled horizontally for heat management
- Background movement of idling instances while preserving state (e.g., buffer pool, connections)
- Up to 15x faster scale downs



Choosing the right target database?


SaaS
Tenant 1
Tenant 2
Tenant 3
Tenant 4
Tenant 5
Tenant 6
AWS Database Migration Service (AWS DMS)
Aurora Serverless v2 cluster
Aurora Provisioned
Aurora Serverless v2 cluster

Pattern - 3
Migrating Third party App

Amazon RDS Custom (Managing Databases)

MANAGED CLOUD DATABASE SERVICE

On-premises	EC2	RDS Custom	RDS
⬛ High availability	⬛ High availability	⬛ High availability	🟨 High availability
⬛ Backups	⬛ Backups	▨ Backups	🟨 Backups
⬛ Patching	⬛ Patching	▨ Patching	🟨 Patching
⬛ Scaling	⬛ Scaling	🟨 Scaling	🟨 Scaling
⬛ Hardware	🟨 Hardware	🟨 Hardware	🟨 Hardware

⬛ You manage

▨ Shared responsibility

🟨 AWS manages

NO MANAGEMENT

Host-level access and full database permissions

Allows features not currently supported by RDS

FULL MANAGEMENT

Allows 3rd-party applications on the database host

Use Cases

Custom & Packaged Business Applications

Customers looking to install applications directly on the host or applications that require elevated privileges

Example: Customer looking to migrate their Oracle ERP application

Granular Control

Customers looking to install custom drivers or enable features that require elevated privileges

Example: Customers looking to utilize features likes DB Vault and Flashback

Lift and Shift

Customers looking to migrate to a managed service without making any changes to the application

Example: Customer looking to migrate using native DB technologies like RMAN

Disaster Recovery

Customers looking to setup DR from self-managed environments

Example: Customer looking to set up Data Guard Stand-By in the cloud

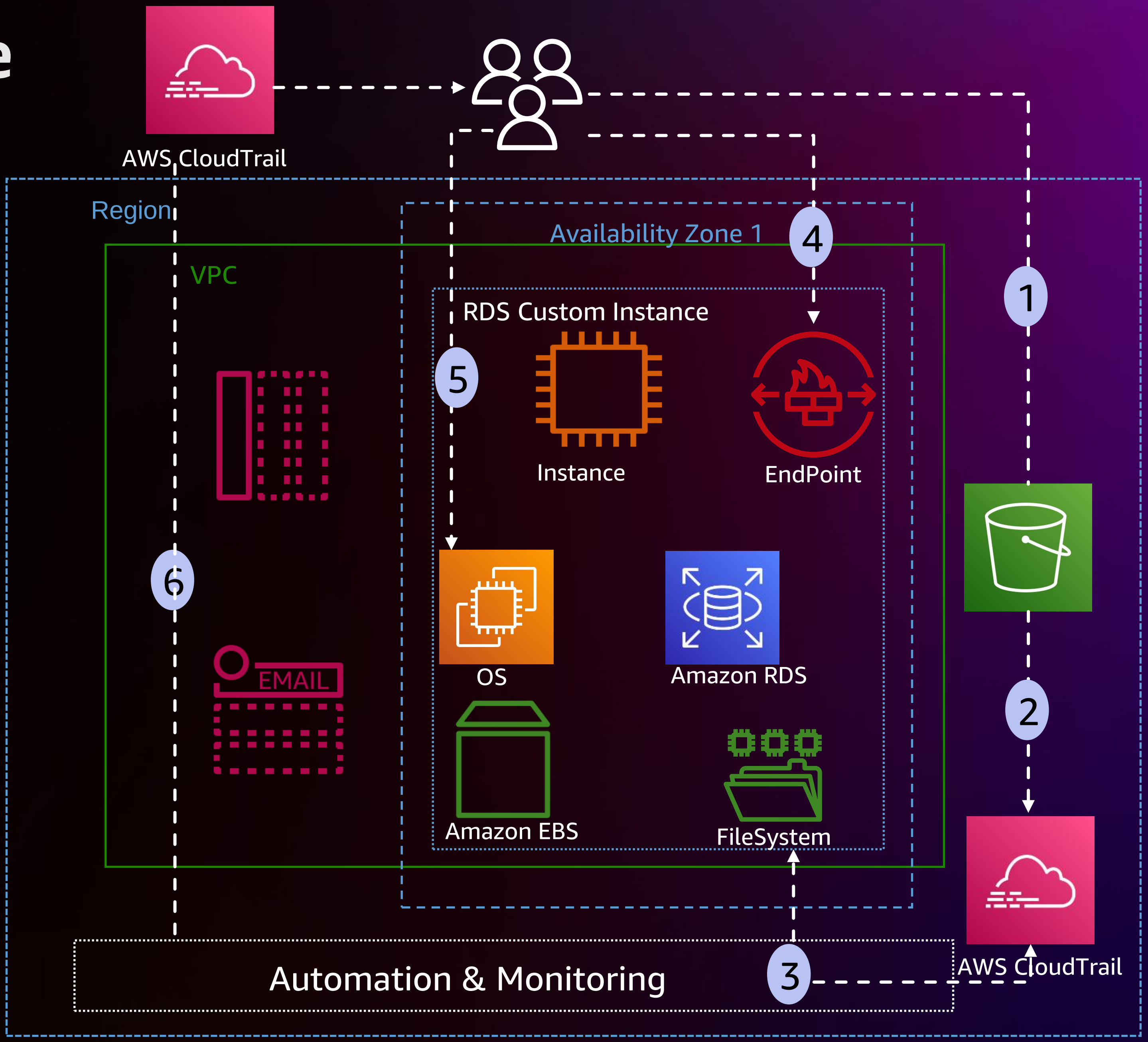
RDS Custom Architecture

DB Instance deployed to customer VPC

Full access with sysdba privileges for Oracle and root access to OS

SSH access to database EC2 instance

SSM Agent, Amazon CloudWatch Agent & Amazon RDS Custom Agent on RDS Custom DB instance



Pattern – 4
Design for Operational excellence

Operational Challenges

Database major version upgrades

Test schema changes with minimal impacts [New Column, Index]

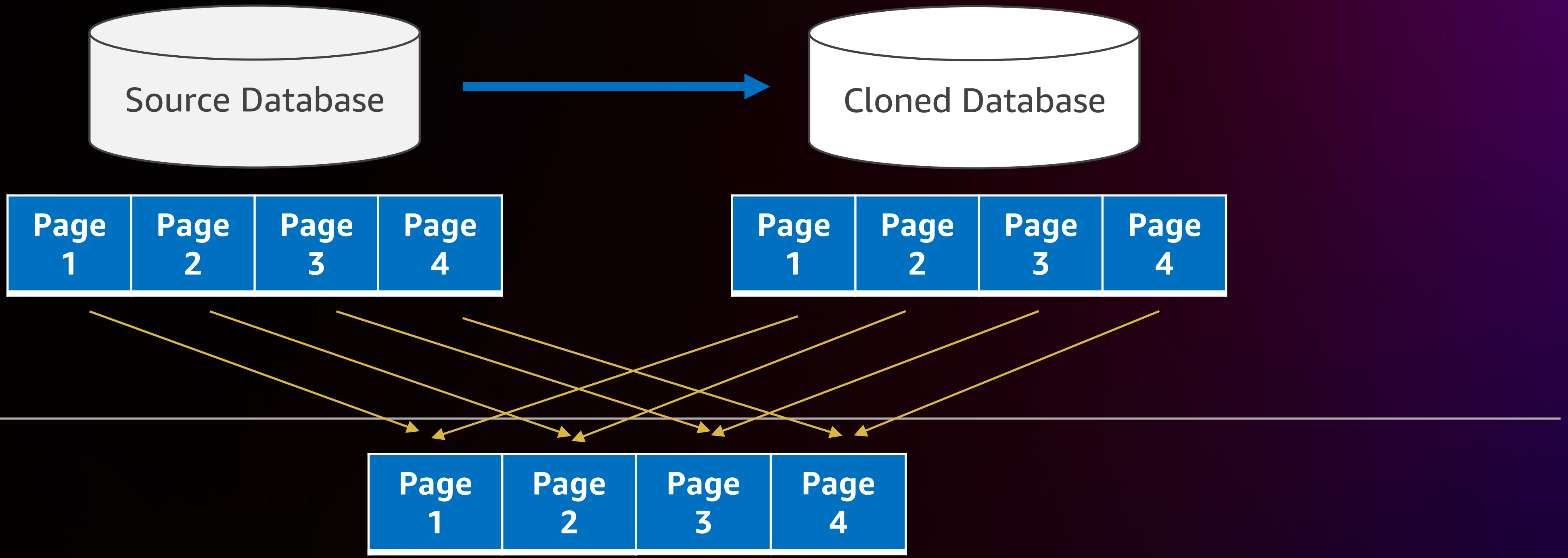
Test new Application version requiring several Database changes

Performance optimization [Index rebuild, Table reorganization]

Aurora Fast database cloning

Create a copy of a DB cluster (storage volume) without duplicate storage cost

Creation is fast – we don’t physically copy data



Source Database

Cloned Database

Page 1 | Page 2 | Page 3 | Page 4

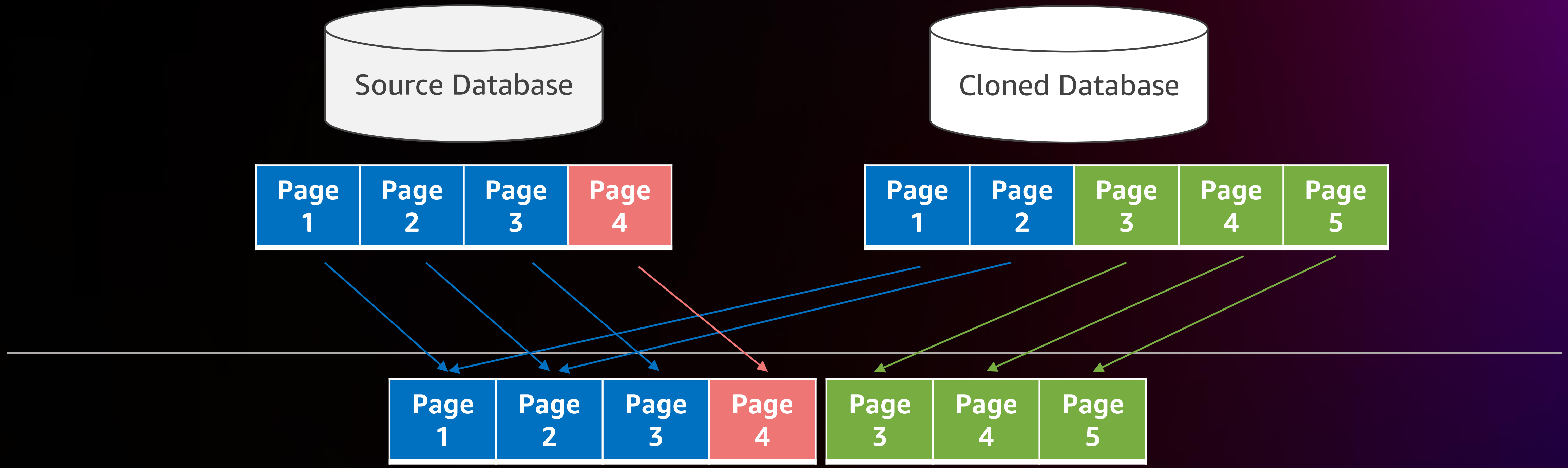
Page 1 | Page 2 | Page 3 | Page 4

Page 1 | Page 2 | Page 3 | Page 4

State: Created a clone, made no storage changes
Both databases reference **same** pages on the shared distributed storage system

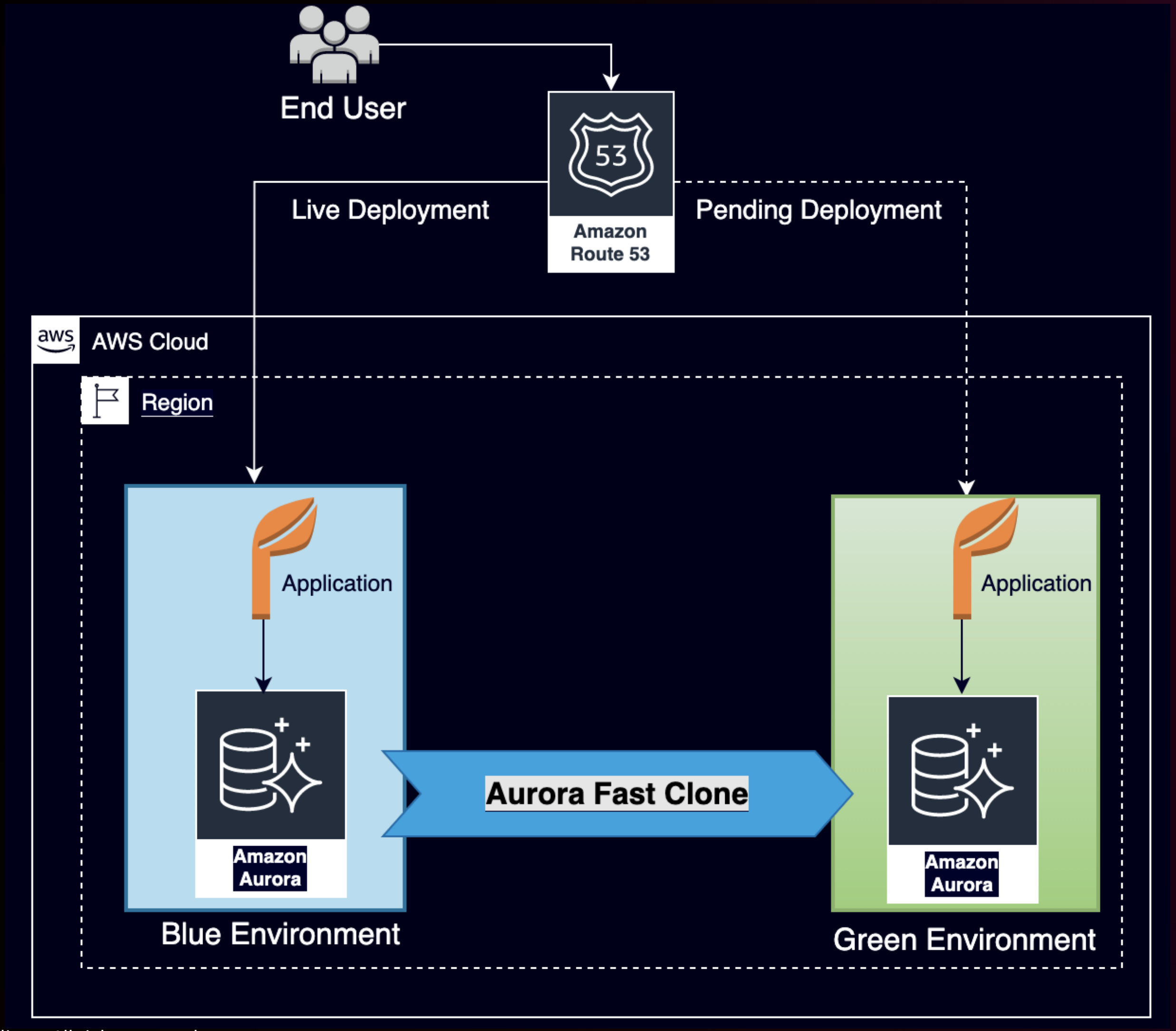
Aurora Fast database cloning

Isolation: Activity on the clone doesn’t impact performance of the source (and vice-versa)

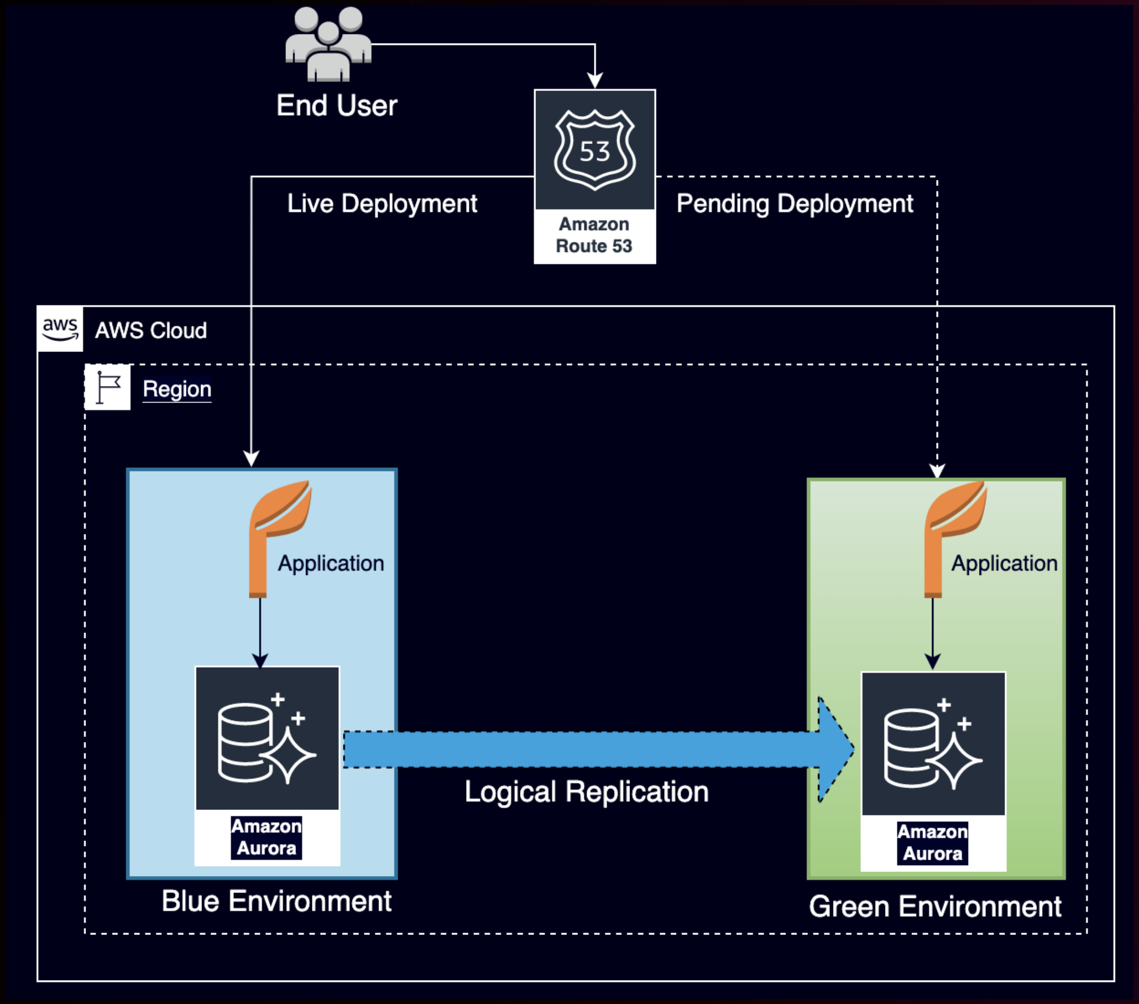


State: Created a clone, made storage changes on both source & clone
Both databases reference **common** pages on the shared distributed storage system

Aurora Blue Green Deployment



Aurora Blue Green Deployment



Pattern – 5
Designing For Resilience

Tiers

Level1

Planned : min / hrs
Unplanned : hrs/days

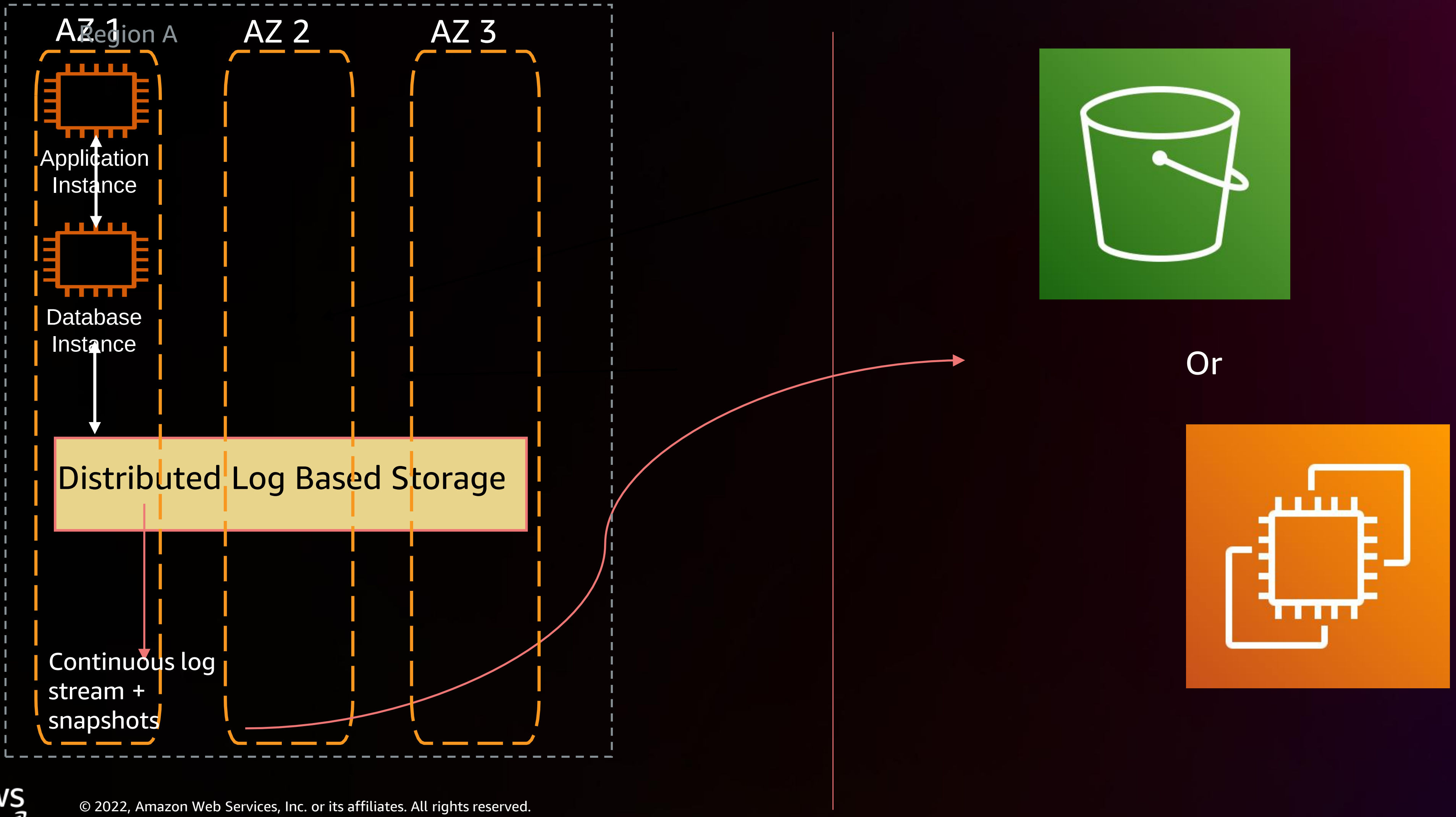
Level2

RPO/RTO
Downtime

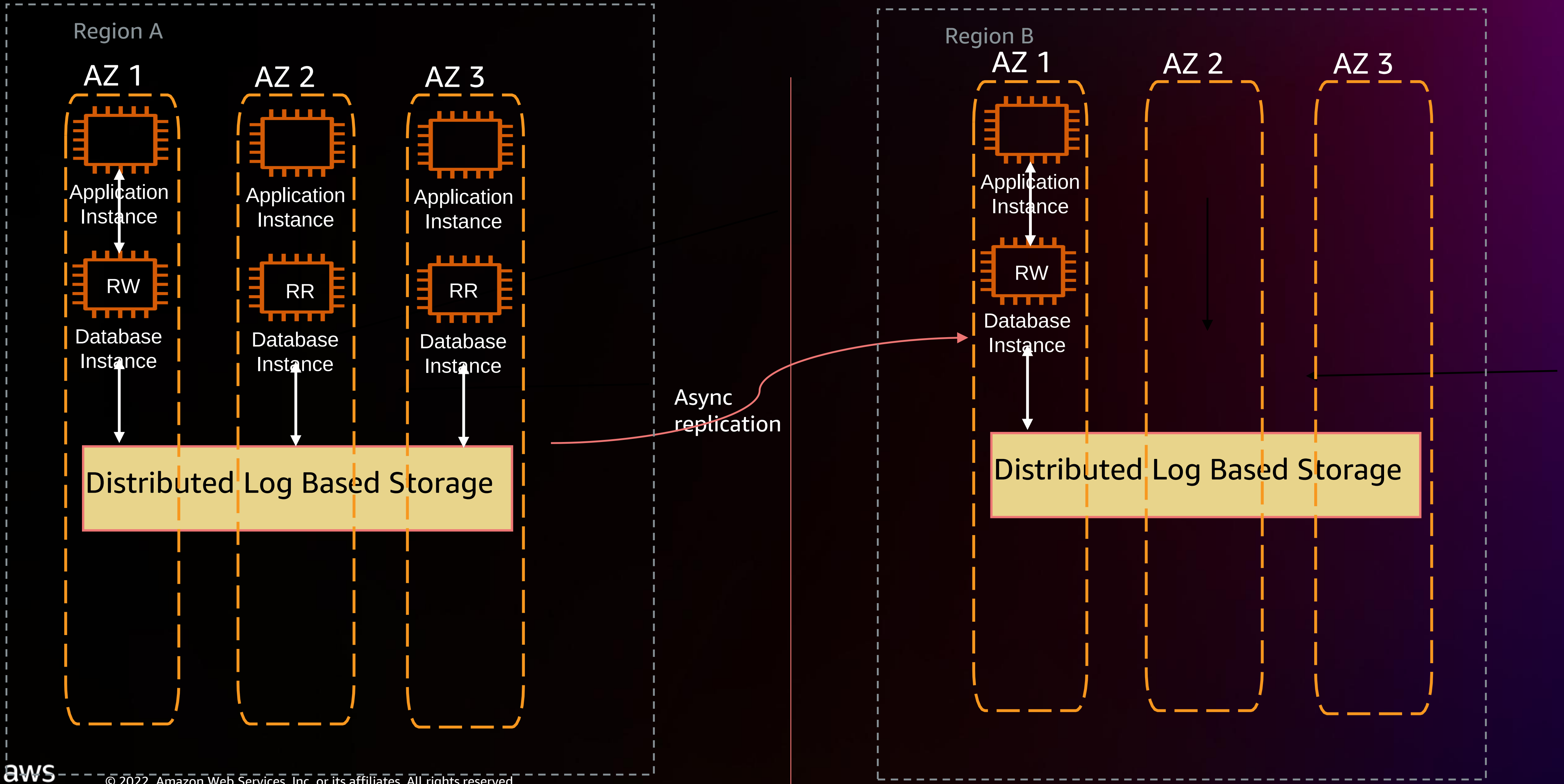
Level3

RPO/RTO
Downtime

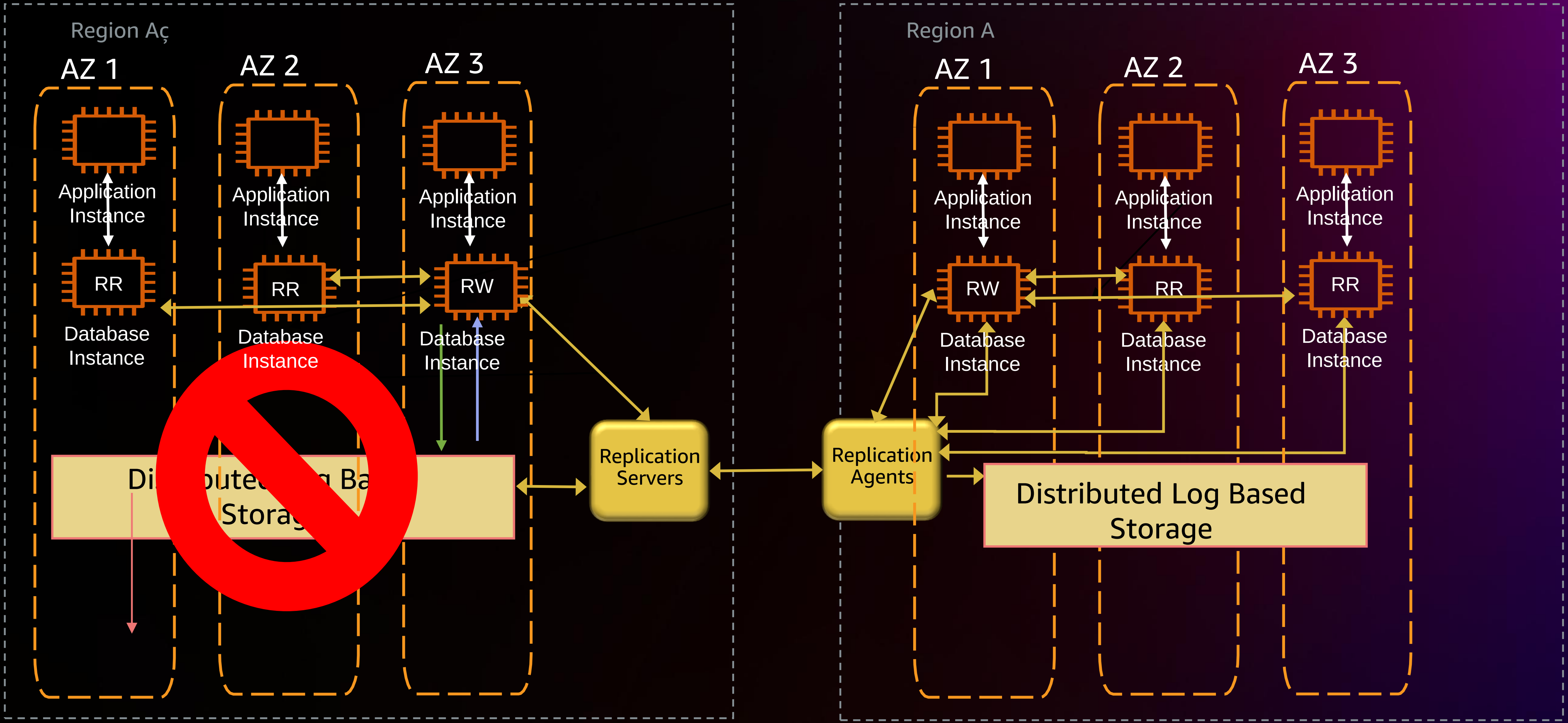
Level1 / bronze / low HA needs


Region A
AZ 1
AZ 2
AZ 3
Application Instance
Database Instance
Distributed Log Based Storage
Continuous log stream + snapshots
Or

Level2 / silver / medium HA needs



Level3 / gold / high HA needs



Summary

Design for migrating Large Data set using DMS

Design best practices in migrating multi-tenant application

Achieving operation excellence using Blue-Green Deployment

Migrating Third party applications using RDS Oracle for custom

Application driven Database HA, DR design best practices

Thank you!

Jatin Singh

sinjatin@amazon.com

Abhinav Sarin

abhsarin@amazon.com

Please complete the session survey in the **mobile app**