

Database on *AWS*

Contents

- RDS Introduction
- DB Multi-AZ, Read Replicas and Backup
- Aurora
- DynamoDB
- ElasticCache
- Caching Strategy on AWS
- RedShift
- Right DB solution
- DMS

RDS introduction

RDS introduction

- RDS stands for **Relational Database Service**.
- It is AWS managed DB service for relational DB
- Supported DB engine
 - MySQL
 - MariaDB
 - PostgreSQL
 - Oracle
 - Microsoft SQL Server
 - Aurora

RDS advantage over self-managed DB on EC2

- Automatically OS patching
- Maintenance window for upgrades
- Automatically backups and restore to specific timestamp (Point in Time Res)
- High Availability with Multi-AZ deployment
- Read replicas for improving read performance
- Quick Insight using Monitoring dashboard
- Scaling capability (Vertical and Horizontal)
- Storage backed by EBS

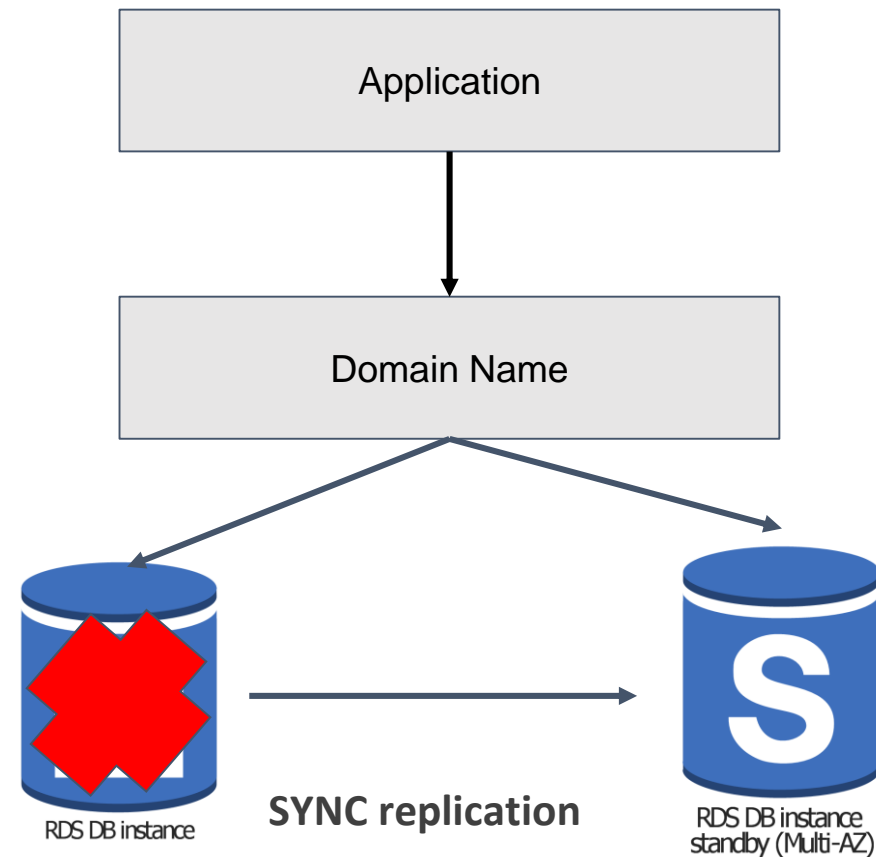
Backups and snapshot

- Backups are enable automatically
- **Automated Backups**
 - Performs a full daily snapshot of your data at specific Window time
 - Captures transaction logs and backups every 5 minutes
 - Restore DB to any specific time you requested (Using daily snapshot + transaction logs)
 - 7 days retention by default (Extent up to 35 days)
- **DB snapshot**
 - Manually trigger by users
 - Retention period as long as you want

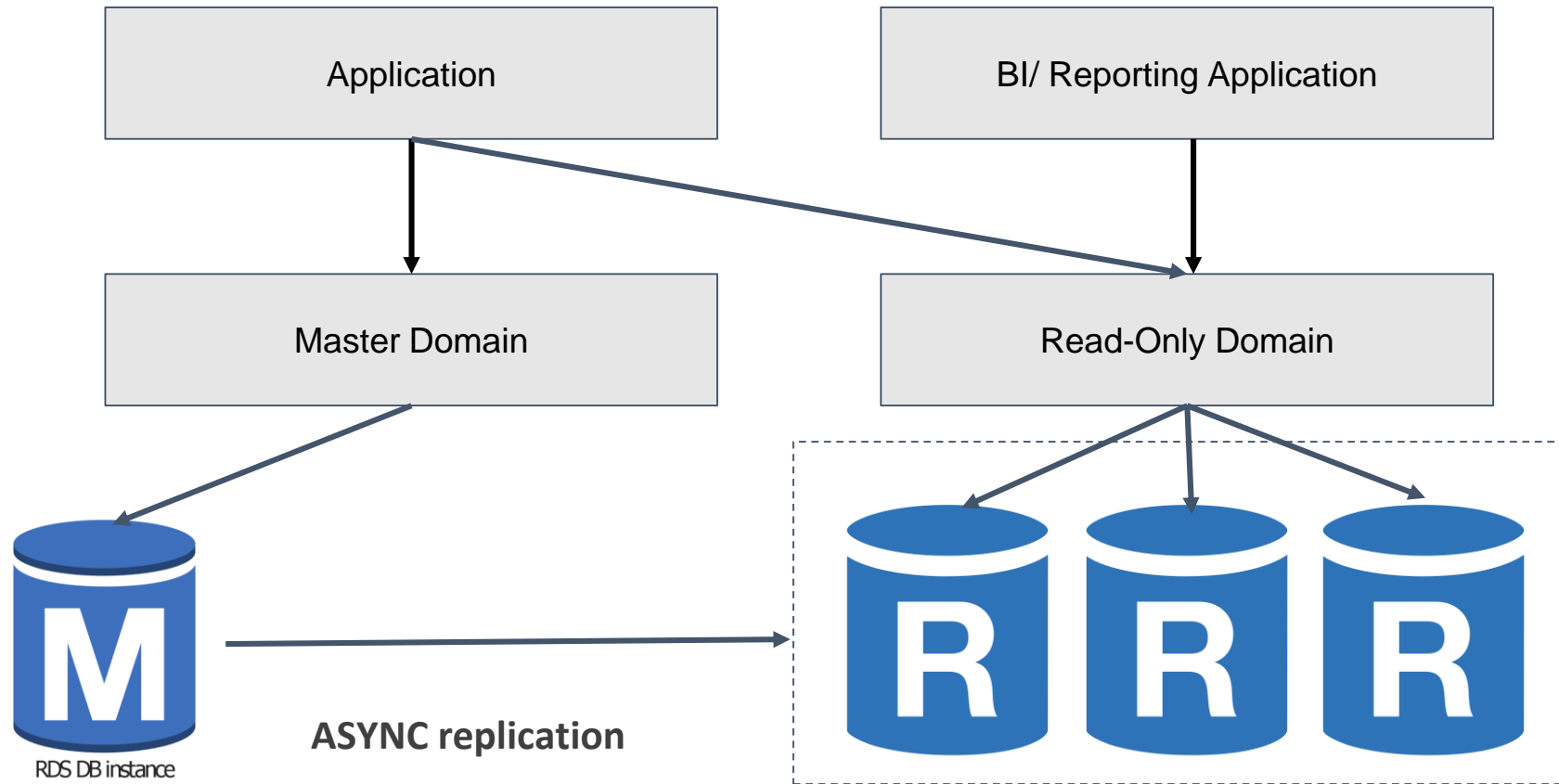
Multi-AZ and Read Replicas

Multi-AZ for High Availability

- One DNS name to application view. Automatically failover under the hood
- Using SYNC replication



Read Replicas for Scalability



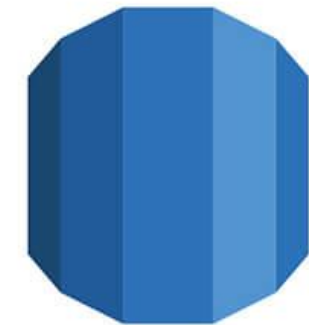
Read Replicas for scalability (cont)

- Up to 5 Replicas for each DB
- Using **ASYNC** replication so
- Each Read Replicas will have its own DNS endpoint
- Read eventually consistent
- Read Replicas can locate in same AZ, cross AZ or cross Region
- Replicas can be promoted to master in case of failure of master
- Can create read Replicas from a read Replicas but considering Replica lags
- Can create Read Replicas of Multi-AZ source DB

Aurora

Aurora

- An AWS own proprietary database. Enterprise grade DB solution.
- MySQL and PostgreSQL-compatible relational DB engine
- Performance ~ 5x MySQL and 3x PostgreSQL
- Data is cloned to 6 copies, across 3 Availability Zones

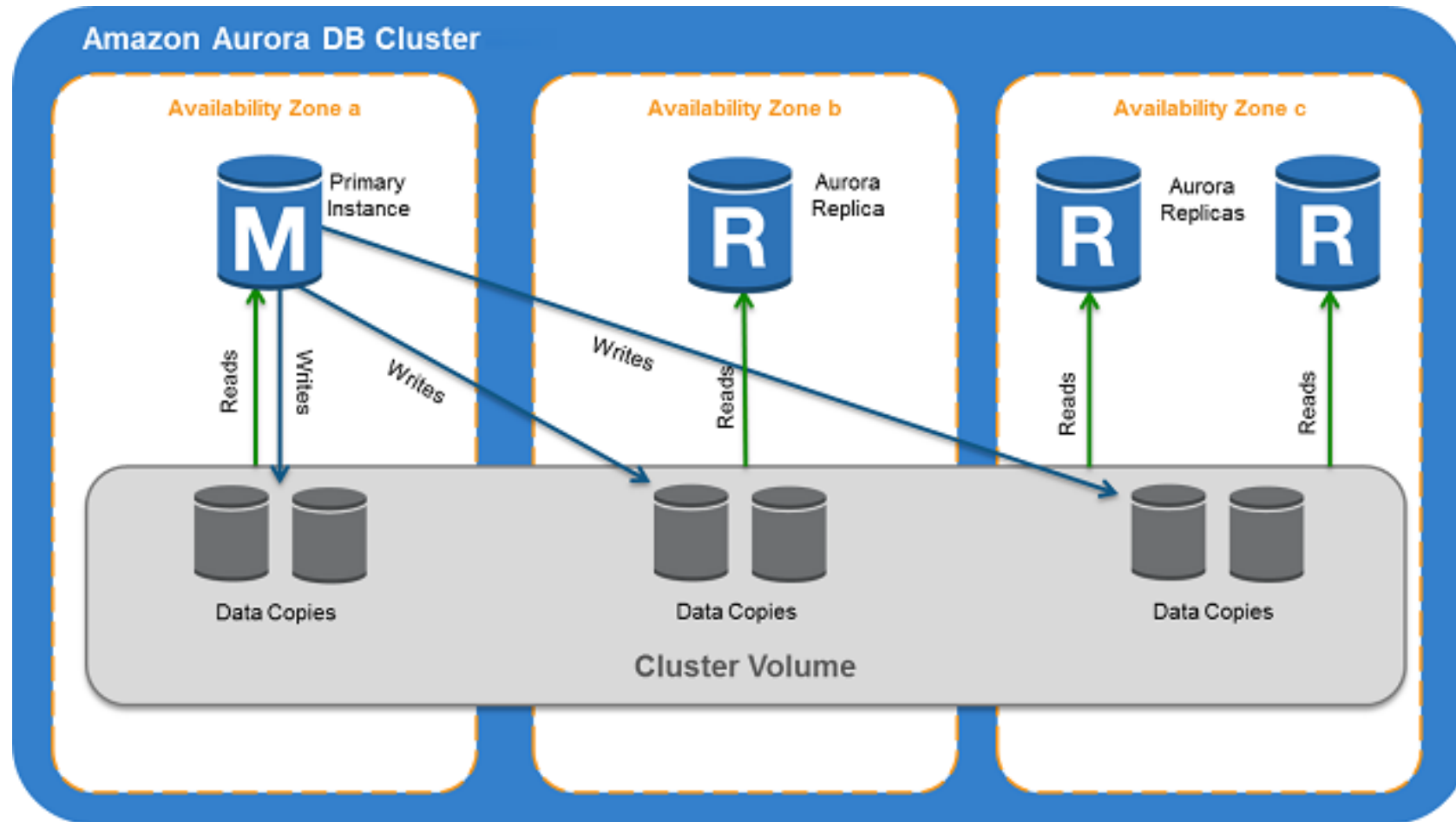


Amazon Aurora

Aurora (cont.)

- Multiz AZ, Auto Scaling Read Replicas, Auto Scaling Storage (up to 128 TiB)
- Auto failover
- Others features are inherit from RDS (Security, Monitoring, maintenance)
- **Use case:**
 - Using as RDS option but much higher performance, less maintenance, flexibility (a bit cost higher)

Aurora Cluster



Exam Tips

- 2 copies of data in each AZ, with 3 minimum AZs, 6 copies of your data
- Snapshots can be created without affecting DB performance, Snapshots can share to other accounts
- Auto failover, Auto scaling storage, replicas
- 3 types of Aurora replica (currently)
 - Aurora replica (up to 15 replicas)
 - Aurora MySQL replica (up to 5 replicas)
 - Aurora PostgreSQL replica (up to 1)
- Aurora has automated backup by default
- Using Aurora Serverless for unpredictable workload, cost-effective

DynamoDB

DynamoDB?

- AWS managed NoSQL database
- Single-digit millisecond latency at any scale
- Support Key-Value and Document data models
- High Availability. Data in DynamoDB is replicated to 3 distinct AZs
- Eventually Consistency Read (default)
- Can setup strong consistency (latency considering)



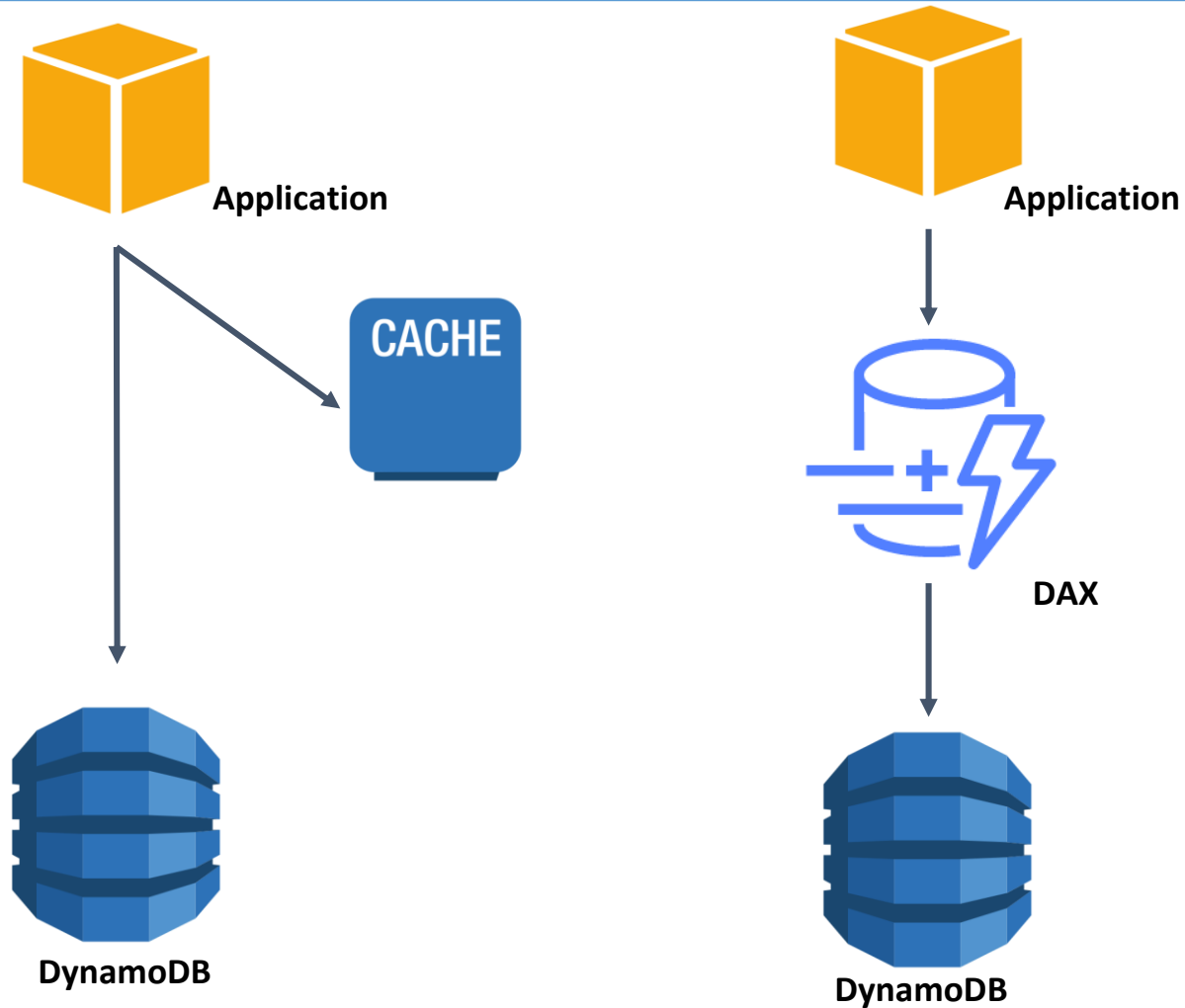
DynamoDB Accelerator (DAX)?

- Fully managed, high available, in-memory cache
- Performance improvement from **milliseconds** to **microsecond** (10x faster)
- Developers don't need to handle caching logic
- Compatible with DynamoDB API calls

DynamoDB Accelerator use cases?

- Use for fastest possible response time for reads
- Repeated reads against a large set of data

DynamoDB Accelerator use cases? (cont.)

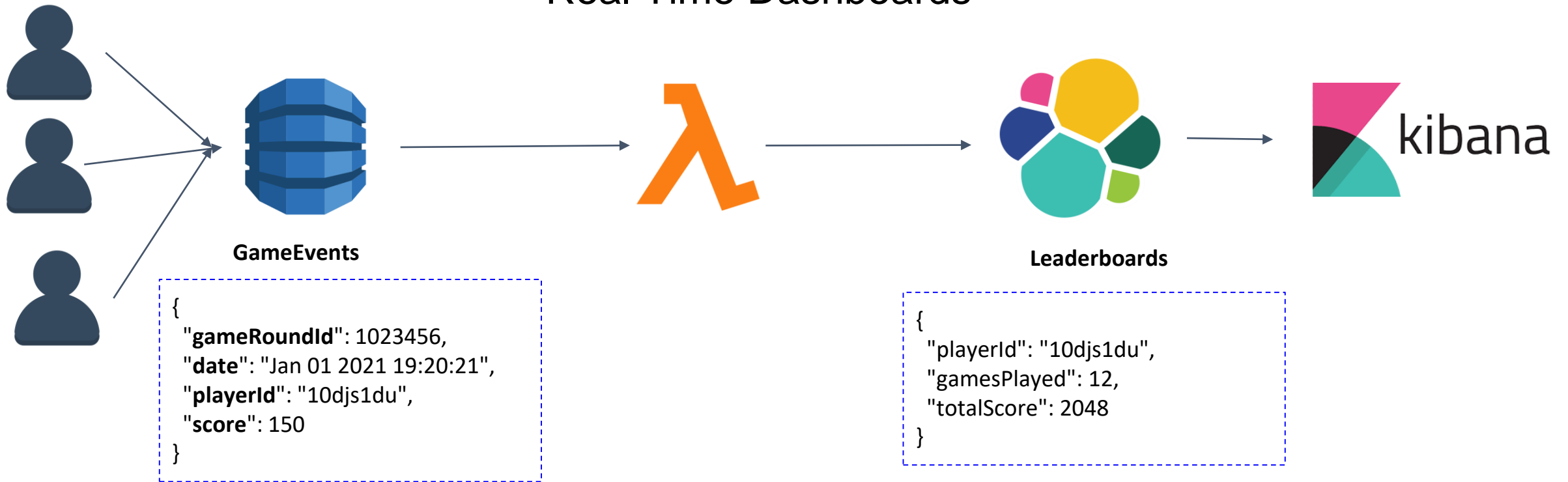


DynamoDB Stream?

- Feature that emits events when record modifications occur on a DynamoDB table
- Events can be INSERT, UPDATE and REMOVE
- Events can carry the content of the rows being modified (old and new)
- Events are guaranteed to be in the same order the modifications took place
- Batch processing
- No performance impact on source table

DynamoDB Stream Use cases?

Real Time Dashboards



Elasticache

Elasticache

- Managed, In-memory key/value caches
- Improve application performance
- Elasticache supports 2 open source in-memory cache engine



Memcached and Redis?

Requirement	Memcached	Redis
Sub-millisecond	Yes	Yes
Data partitioning	Yes	Yes
Scale horizontally	Yes	Yes
Multithreaded architecture	Yes	-
Advanced data structures	-	Yes
Persistence	-	Yes
Multi-AZ - Yes	-	Yes
Replication	-	Yes
Transactions	-	Yes
Pub/Sub	-	Yes
Backup and restore capability	-	Yes

Exam tips

- Using ElastiCache to improve application performance
- Memcached supports multi thread, Redis supports only single thread
- Redis cluster can span across AZs (multi-AZ), auto failover
- Performance: Sub-milliseconds performance
- Cost: Pay for EC2 instances + storage

Caching strategy

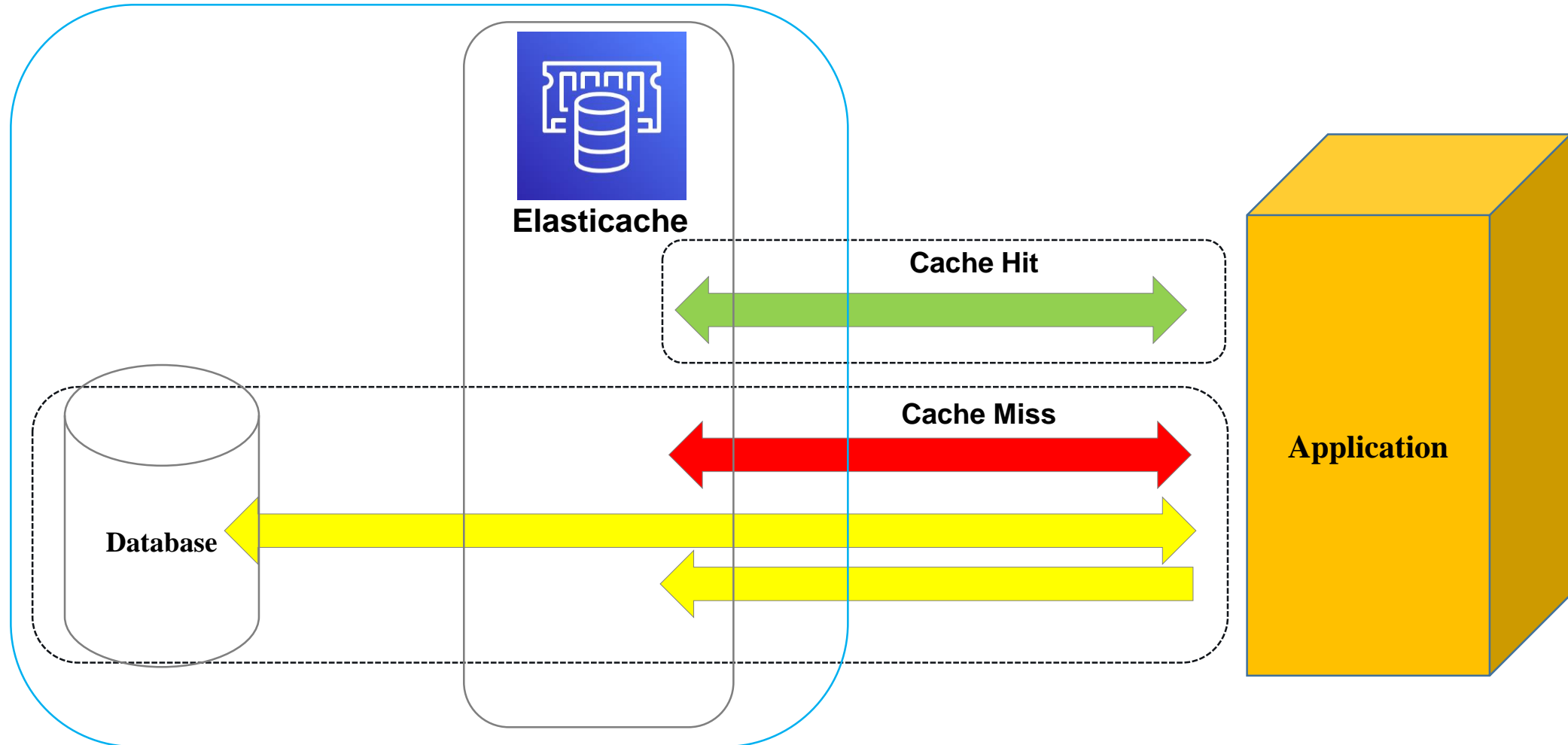
Elasticache Security

- SSL in-transit encryption
- Don't support IAM for authentication (Only AWS API for creating, delete cluster...)
- REDIS AUTH can set **password/token** at time of creating cluster
- REDIS AUTH is extra level of security beside security groups
- Memcached supports SAML based authentication

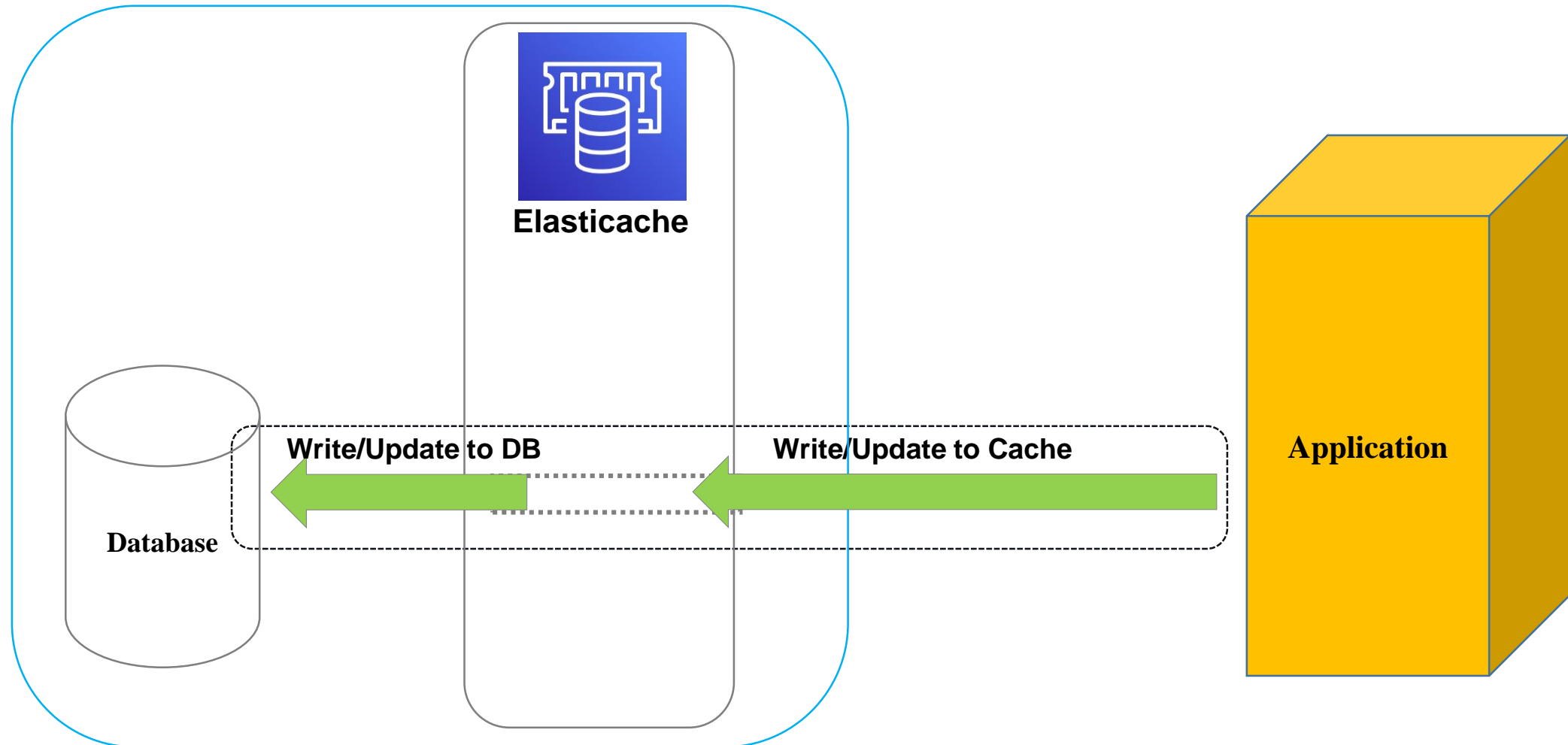
Caching strategy

- Lady Loading
- Write-Through
- TTL

Lady Loading



Write-Through



TTL (Time To Live)

- Data in Cache is set a period for expiration time (TTL)
- After TTL, data is expired and it is treated as though the data is not found
- Combine with Lazy-Loading and Write-Through for better fit your purpose

Caching strategy comparison

	Lazy-Loading	Write-Through
Advantaged	<ul style="list-style-type: none">• Only cache requested data• Node failure doesn't affect application	<ul style="list-style-type: none">• Data is never stale
Dissavantaged	<ul style="list-style-type: none">• Cache miss => delay time• Stale Data	<ul style="list-style-type: none">• Missing data in case of node failure• Waste resources in Cache

Redshift

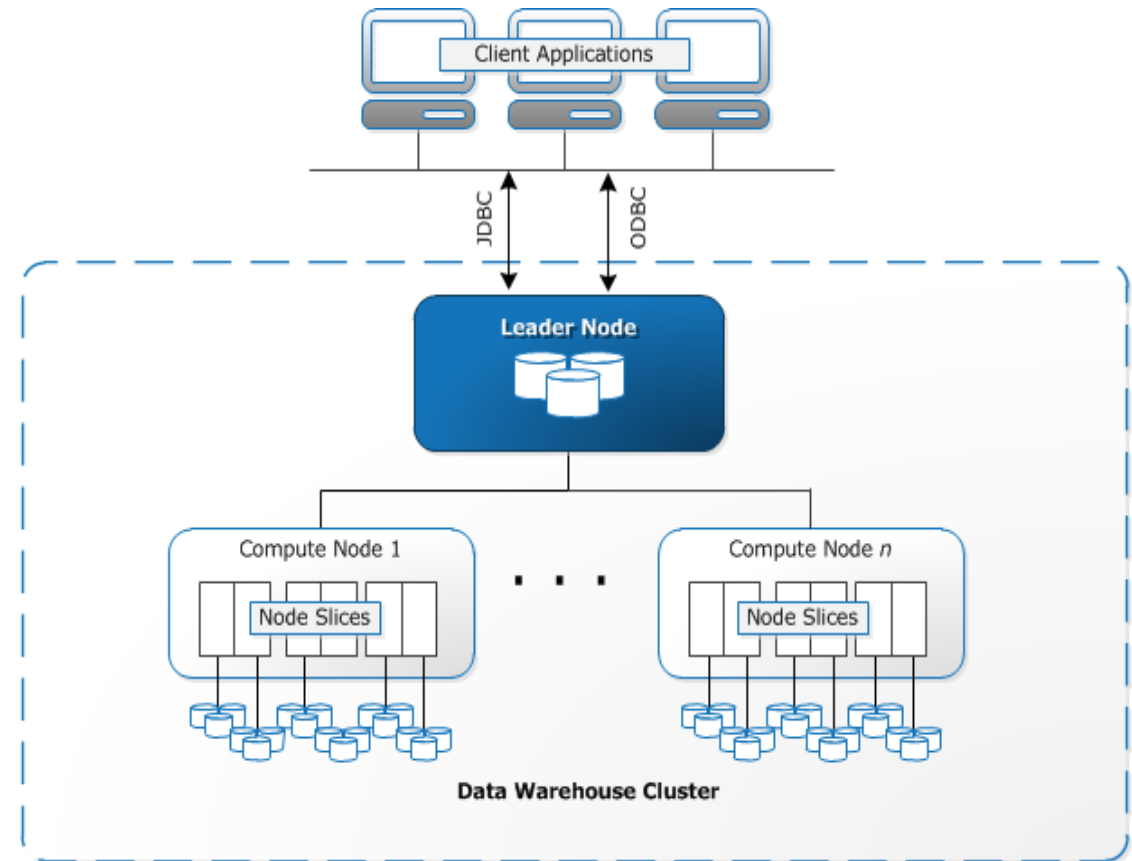
Redshift

- Redshift is OLAP (Online Analytical Processing)
- Redshift is used for analytic. BI, and data warehousing
- Columnar data stores
- Massively Parallel Processing (MPP)



Redshift configuration

- Single Node (master + compute in same instance)
- Multi-Node
 - **Leader Node** (manages client connection and receive queries, aggregation)
 - **Compute Node** (Store data and perform queries, computation)



Redshift Availability and Backup

- **Availability**

- Currently only available to setup in one AZ
- It can be restore from snapshot to another AZ in case of outage

- **Backup**

- Enabled by Default with 1 day retention
- Maximum 35 days retention
- Redshift can asynchronous replica your snapshots to S3 in another region

Security for Redshift

- Encrypt in transit using SSL
- Encrypt at rest using AES-256 encryption
 - Key through HSM
 - KMS



Exam Tips








- Redshift is used for business intelligence and data warehousing
- Redshift is OLAP (Online Analytical Processing)
- Available in only 1 AZ
- Enabled by Default with 1 day retention
- Maximum 35 days retention
- Redshift can asynchronous replica your snapshots to S3 in another region

Choosing Right DB solution

Do **NOT** fit all things in one place



Common data categories and use cases

						
Relational	Key-value	Document	In-memory	Graph	Time-series	Ledger
Referential integrity, ACID transactions, schema-on-write	High throughput, low-latency reads and writes, endless scale	Store documents and quickly access querying on any attribute	Query by key with microsecond latency	Quickly and easily create and navigate relationships between data	Collect, store, and process data sequenced by time	Complete, immutable, and verifiable history of all changes to application data
Lift and shift, ERP, CRM, finance	Real-time bidding, shopping cart, social, product catalog, customer preferences	Content management, personalization, mobile	Leaderboards, real-time analytics, caching	Fraud detection, social networking, recommendation engine	IoT applications, event tracking	Systems of record, supply chain, health care, registrations, financial

Database Migration Service

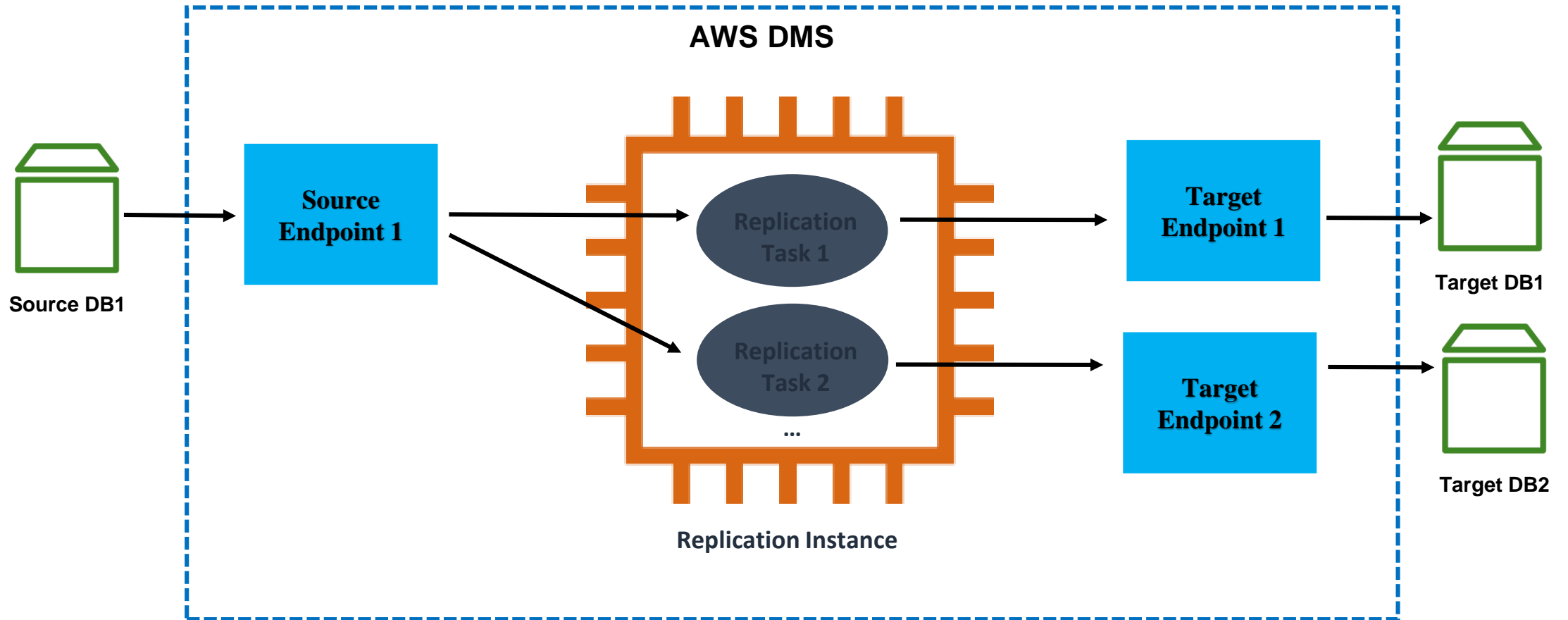
Database Migration Service

- AWS managed service
- Migrate relational databases, NoSQL, data warehouse in/out to AWS cloud
- Support 2 types of migration
 - Same DB engine (**homogenous**)
Ex: Oracle => Oracle
 - Differ DB engine (**heterogeneous**)
Ex: SQL Server => Amazon Aurora



DMS

Components of DMS



Exam Tips

- DMS allows to migrate Relational DB, Data Warehouse, NoSQL from one source to AWS
- Source DB can locate in On-Premise, AWS cloud, Azure...
- Can migrate use **homogenous** (same DB engine) or heterogeneous (differ DB engine) migrations
- Need to use **AWS SCT** (Schemal Conversion Tool) to convert schema in case of heterogeneous migrations