AWS - Aurora

Aurora: OLTP database.

- **Amazon Aurora** typically involves a cluster of DB instances instead of a single instance.
- Each connection is handled by a specific DB instance.
- **When you connect to an Aurora cluster, the hostname and port that you specify point to an intermediate handler called an *endpoint*.**
- Aurora uses the endpoint mechanism to abstract these connections.
- Thus, you don't have to hardcode all the hostnames or write your own logic for load-balancing and rerouting connections when some DB instances aren't available.
- For certain Aurora tasks, different instances or groups of instances perform different roles.
- For example, the primary instance handles all data definition language (DDL) and data manipulation language (DML) statements.
- Up to 15 Aurora Replicas handle read-only query traffic.

END-POINTS

- Using endpoints, you can map each connection to the appropriate instance or group of instances based on your use case.
- For example, to perform DDL statements you can connect to whichever instance is the primary instance.
- To perform queries, you can connect to the reader endpoint, with Aurora automatically performing load-balancing among all the Aurora Replicas.
- For clusters with DB instances of different capacities or configurations, you can connect to custom endpoints associated with different subsets of DB instances.
- For diagnosis or tuning, you can connect to a specific instance endpoint to examine details about a specific DB instance.

READER END-POINT

- A *reader endpoint* for an Aurora DB cluster provides load-balancing support for read-only connections to the DB cluster.
- Use the reader endpoint for read operations, such as queries. By processing those statements on the read-only Aurora Replicas, this endpoint reduces the overhead on the primary instance.
- It also helps the cluster to scale the capacity to handle simultaneous SELECT queries, proportional to the number of Aurora Replicas in the cluster. Each Aurora DB cluster has one reader endpoint.
- If the cluster contains one or more Aurora Replicas, the reader endpoint load-balances each connection request among the Aurora Replicas.
- In that case, you can only perform read-only statements such as SELECT in that session. If the cluster only contains a primary instance and no Aurora Replicas, the reader endpoint connects to the primary instance. In that case, you can perform write operations through the endpoint.

CUSTOM END-POINTS

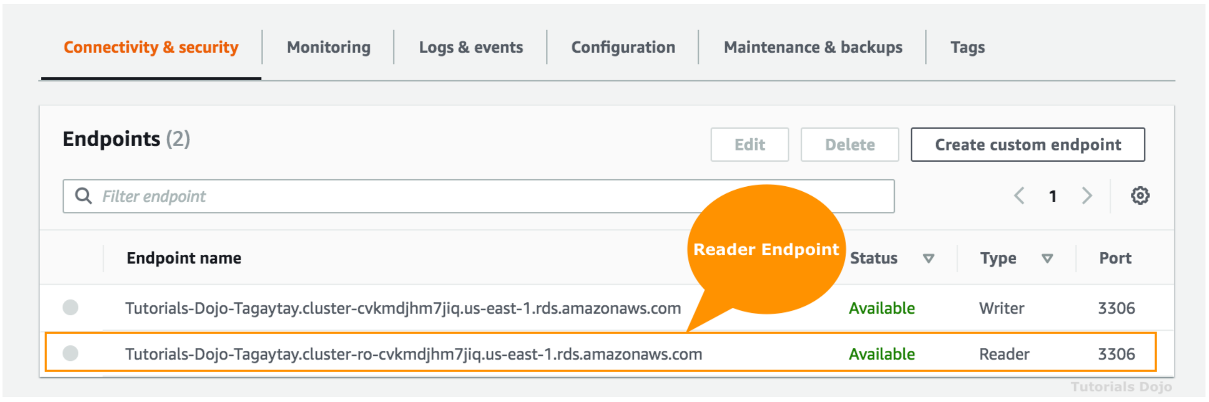
- The custom endpoint provides load-balanced database connections based on criteria other than the read-only or read-write capability of the DB instances.
- For example, you might define a custom endpoint to connect to instances that use a particular AWS instance class or a particular DB parameter group.
- Then you might tell particular groups of users about this custom endpoint.
- For example, you might direct internal users to low-capacity instances for report generation or ad hoc (one-time) querying, and direct production traffic to high-capacity instances.
- Hence, creating a custom endpoint in Aurora based on the specified criteria for the production traffic and another custom endpoint to handle the reporting queries is the correct answer.

References:

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Overview.Endpoints.html

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Overview.html

https://aws.amazon.com/rds/aurora/parallel-query/



Endpoints (2)

	Endpoint name	Status	Type	Port
○	Tutorials-Dojo-Tagaytay.cluster-cvkmdjhm7jiq.us-east-1.rds.amazonaws.com	Available	Writer	3306
○	Tutorials-Dojo-Tagaytay.cluster-ro-cvkmdjhm7jiq.us-east-1.rds.amazonaws.com	Available	Reader	3306

AWS - Aurora Serverless

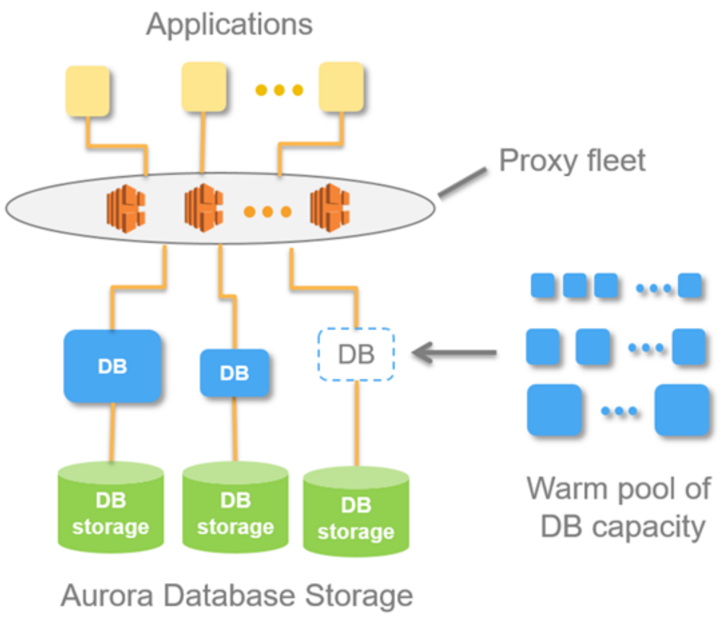
Aurora SERVERLESS - db cluster

- Amazon Aurora Serverless is an on-demand, auto-scaling configuration for Amazon Aurora.
- An Aurora Serverless DB cluster is a DB cluster that automatically starts up, shuts down, and scales up or down its compute capacity based on your application's needs.
- Aurora Serverless provides a relatively simple, cost-effective option for infrequent, intermittent, sporadic or unpredictable workloads.
- It can provide this because it automatically starts up, scales compute capacity to match your application's usage and shuts down when it's not in use.

Aurora PROVISIONED - db cluster

- Take note that a non-Serverless DB cluster for Aurora is called a provisioned DB cluster.
- Aurora Serverless clusters and provisioned clusters both have the same kind of high-capacity, distributed, and highly available storage volume.
- When you work with Amazon Aurora without Aurora Serverless (provisioned DB clusters), you can choose your DB instance class size and create Aurora Replicas to increase read throughput.
- If your workload changes, you can modify the DB instance class size and change the number of Aurora Replicas.
- This model works well when the database workload is predictable, because you can adjust capacity manually based on the expected workload.

- However, in some environments, workloads can be intermittent and unpredictable.
- There can be periods of heavy workloads that might last only a few minutes or hours, and also long periods of light activity, or even no activity.
- Some examples are retail websites with intermittent sales events, reporting databases that produce reports when needed, development and testing environments, and new applications with uncertain requirements.
- In these cases and many others, it can be difficult to configure the correct capacity at the right times.
- It can also result in higher costs when you pay for capacity that isn't used.



- With Aurora Serverless , you can create a database endpoint without specifying the DB instance class size.
- You set the minimum and maximum capacity.

- With Aurora Serverless, the database endpoint connects to a *proxy fleet* that routes the workload to a fleet of resources that are automatically scaled.
- Because of the proxy fleet, connections are continuous as Aurora Serverless scales the resources automatically based on the minimum and maximum capacity specifications.
- Database client applications don't need to change to use the proxy fleet. Aurora Serverless manages the connections automatically.
- Scaling is rapid because it uses a pool of "warm" resources that are always ready to service requests.
- Storage and processing are separate, so you can scale down to zero processing and pay only for storage.

Aurora Serverless introduces a new serverless DB engine mode for Aurora DB clusters.
Non-Serverless DB clusters use the provisioned DB engine mode.

References:

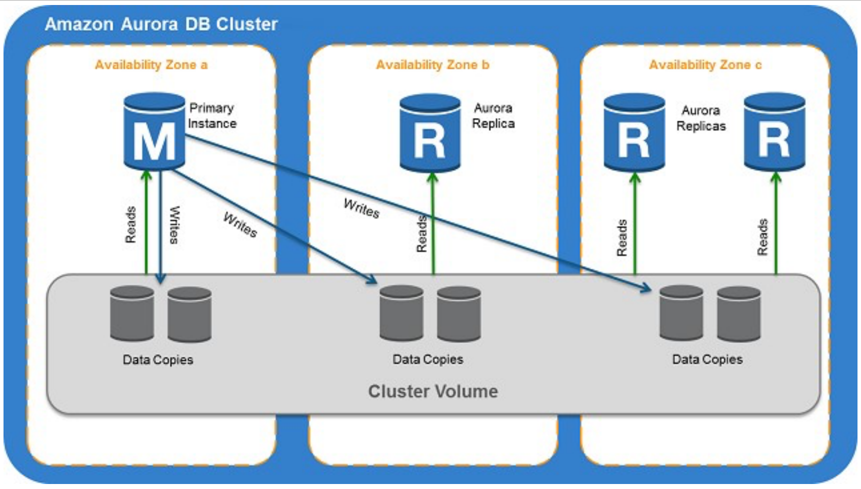
https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/aurora-serverless.how-it-works.html

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/aurora-serverless.html

AWS - Aurora FAIL-OVER

Aurora Fail-Over

Failover is automatically handled by Amazon Aurora so that your applications can resume database operations as quickly as possible without manual administrative intervention.



Fail-Over for - Amazon AURORA REPLICA

- If you have an Amazon Aurora Replica in the same or a different Availability Zone, when failing over, Amazon Aurora flips the canonical name record (CNAME) for your DB Instance to point at the healthy replica, which in turn is promoted to become the new primary.
- Start-to-finish, failover typically completes within 30 seconds.

Fail-Over for - Amazon AURORA SERVERLESS with Diff AZs

- If you are running Aurora Serverless and the DB instance or AZ become unavailable, Aurora will automatically recreate the DB instance in a different AZ.

Fail-Over for - Amazon AURORA SINGLE instance

- If you do not have an Amazon Aurora Replica (i.e. single instance) and are not running Aurora Serverless, Aurora will attempt to create a new DB Instance in the same Availability Zone as the original instance.
- This replacement of the original instance is done on a best-effort basis and may not succeed, for example, if there is an issue that is broadly affecting the Availability Zone.

References:

https://aws.amazon.com/rds/aurora/faqs/

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Concepts.AuroraHighAvailability.html

Check out this Amazon Aurora Cheat Sheet:

https://tutorialsdojo.com/aws-cheat-sheet-amazon-aurora/