AWS DynamoDB Advanced

## DynamoDB TTL

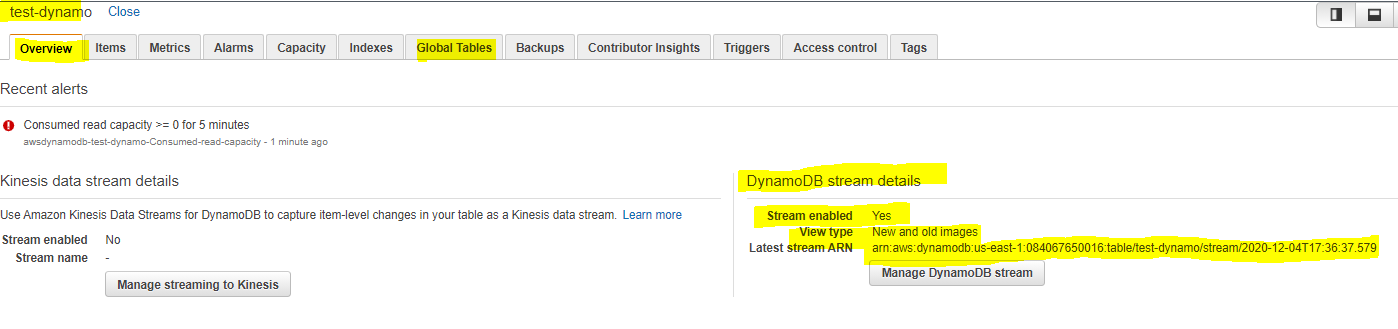
* DynamoDB Time to Live (TTL) enables a per-item timestamp to determine when an item is no longer needed.
* After the date and time of the specified timestamp, DynamoDB deletes the item from the table **without consuming any write throughput**.
* DynamoDB TTL is provided at no extra cost and can help reduce the data storage by retaining only required data.
* Items are deleted from the table are also removed from any local secondary index and global secondary index in the same way as a DeleteItem operation.
* DynamoDB Stream tracks the delete operation as as a system delete and not a regular delete.
* TTL is useful if the stored items lose relevance after a specific time. for e.g.
  + Remove user or sensor data after an year of inactivity in an application
  + Archive expired items to an S3 data lake via DynamoDB Streams and AWS Lambda.
  + Retain sensitive data for a certain amount of time according to contractual or regulatory obligations.

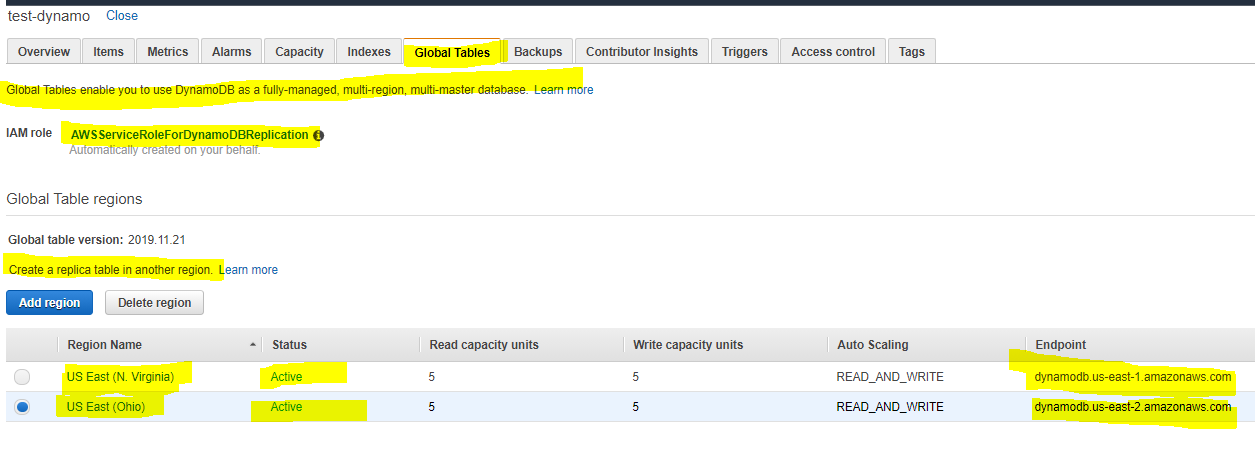
## DynamoDB Cross-region Replication

* DynamoDB cross-region replication allows identical copies (called replicas) of a DynamoDB table (called master table) to be maintained in one or more AWS regions
* Writes to the table will be automatically propagated to all replicas
* Cross-region replication currently supports **single master mode**. A single master has one master table and one or more replica tables
* Read replicas are updated **asynchronously** as DynamoDB acknowledges a write operation as successful once it has been accepted by the master table. The write will then be propagated to each replica with a slight delay.
* Cross-region replication can be helpful in scenarios
  + **Efficient disaster recovery,**in case a data center failure occurs.
  + **Faster reads**, for customers in multiple regions by delivering data faster by reading a DynamoDB table from the closest AWS data center.
  + **Easier traffic management,** to distribute the read workload across tables and thereby consume less read capacity in the master table.
  + **Easy regional migration**, by promoting a read replica to master
  + **Live data migration**, to replicate data and when the tables are in sync, switch the application to write to the destination region
* Cross-region replication costing depends on
  + Provisioned throughput (Writes and Reads)
  + Storage for the replica tables.
  + Data Transfer across regions
  + Reading data from DynamoDB Streams to keep the tables in sync.
  + Cost of EC2 instances provisioned, depending upon the instance types and region, to host the replication process.
* **NOTE** : Cross Region replication on DynamoDB was performed defining AWS Data Pipeline job which used EMR internally to transfer data before the DynamoDB streams and out of box cross region replication support

## DynamoDB Global Tables

* DynamoDB Global Tables is a new **multi-master, cross-region replication** capability of DynamoDB to support data access locality and regional fault tolerance for database workloads.
* Applications can now perform reads and writes to DynamoDB in AWS regions around the world, with changes in any region propagated to every region where a table is replicated.
* Global Tables help in building applications to advantage of data locality to reduce overall latency.
* Global Tables ensures eventual consistency
* Global Tables replicates data among regions within a single AWS account, and currently does not support cross account access
* **Global Tables requires DynamoDB streams enabled with New and Old image setting**





Diagram

Description automatically generated with low confidences

## DynamoDB Streams

* DynamoDB Streams provides a**time-ordered sequence of item-level changes** made to data in a table
* DynamoDB Streams stores the data for last 24 hours, after which they are erased
* DynamoDB Streams maintains ordered sequence of the events per item however sequence across item are not maintained.
* Example
  + For e.g., suppose that you have a DynamoDB table tracking high scores for a game and that each item in the table represents an individual player. If you make the following three updates in this order:
    - Update 1: Change Player 1’s high score to 100 points
    - Update 2: Change Player 2’s high score to 50 points
    - Update 3: Change Player 1’s high score to 125 points
  + DynamoDB Streams will maintain the order for Player 1 score events. However, it would not maintain the order across the players. So Player 2 score event is not guaranteed between the 2 Player 1 events
* DynamoDB Streams APIs helps developers consume updates and receive the item-level data before and after items are changed
* DynamoDB Streams allows read at up to twice the rate of the provisioned write capacity of the DynamoDB table
* **DynamoDB Streams have to be enabled on a per-table basis**
* DynamoDB Streams is designed for **No Duplicates**, so that every update made to the table will be represented exactly once in the stream.
* DynamoDB streams can be used for multi-region replication to keep other data stores up-to-date with the latest changes to DynamoDB or to take actions based on the changes made to the table

## Text Description automatically generated

## DynamoDB Triggers

* DynamoDB Triggers (just like database triggers) is a feature which allows execution of custom actions based on item-level updates on a table
* DynamoDB triggers can be used in scenarios like sending notifications, updating an aggregate table, and connecting DynamoDB tables to other data sources
* DynamoDB Trigger flow
  + Custom logic for a DynamoDB trigger is stored in an AWS Lambda function as code.
  + A trigger for a given table can be created by associating an AWS Lambda function to the stream (via DynamoDB Streams) on a table.
  + When the table is updated, the updates are published to DynamoDB Streams.
  + In turn, AWS Lambda reads the updates from the associated stream and executes the code in the function.

## DynamoDB Accelerator DAX

A picture containing diagram

Description automatically generated

Text

Description automatically generated

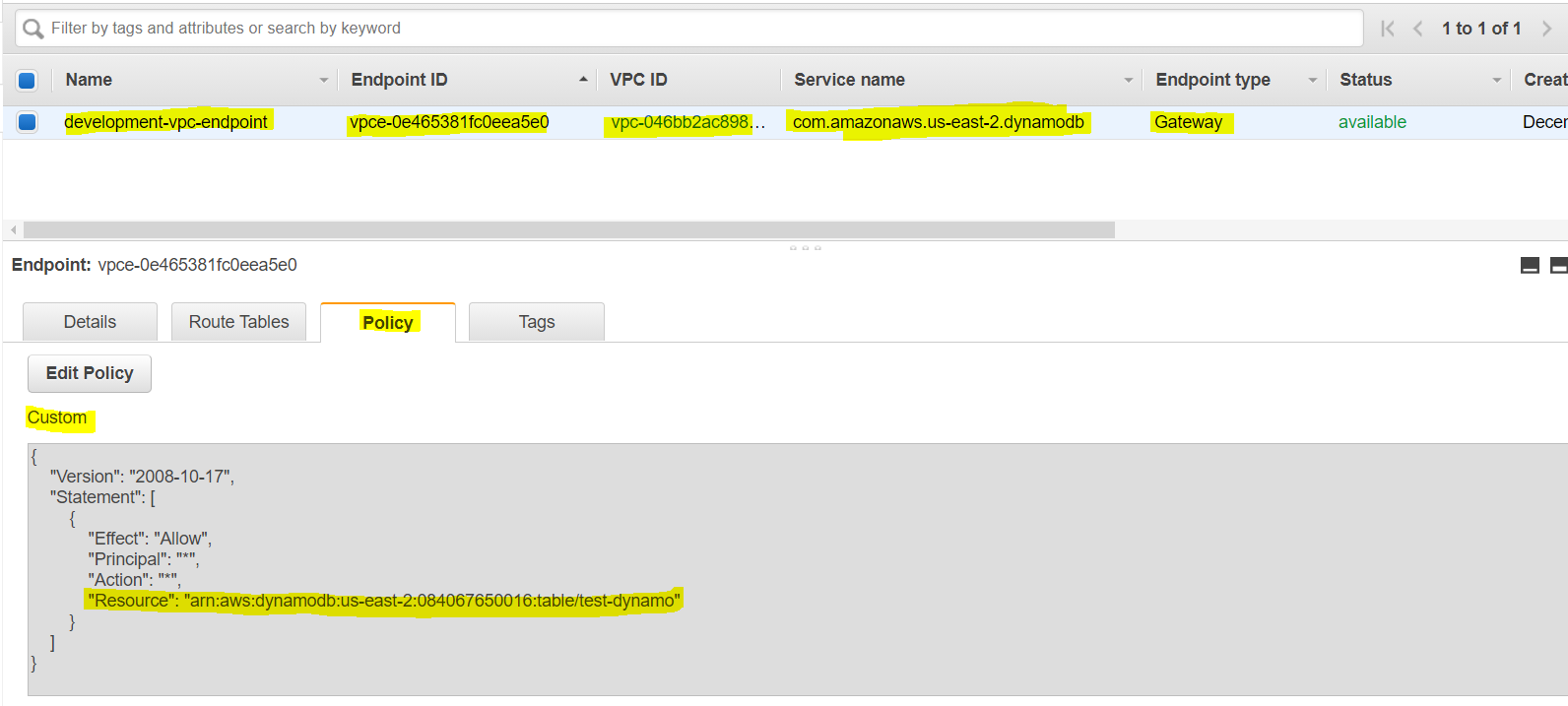
Text

Description automatically generated

Refer blog post [DynamoDB Accelerator – DAX](https://jayendrapatil.com/aws-dynamodb-accelerator-dax/)

## VPC Endpoints

* VPC endpoints for DynamoDB improve privacy and security, especially those dealing with sensitive workloads with compliance and audit requirements, by enabling private access to DynamoDB from within a VPC without the need for an internet gateway or NAT gateway.
* VPC endpoints for DynamoDB support IAM policies to simplify DynamoDB access control, where access can be restricted to a specific VPC endpoint.
* VPC endpoints can be created only for Amazon DynamoDB tables in the same AWS Region as the VPC
* **DynamoDB Streams cannot be access using VPC endpoints for DynamoDB**



## AWS Certification Exam Practice Questions

1. What are the services supported by VPC endpoints, using Gateway endpoint type? Choose 2 answers
   1. **Amazon S3 (Gateway endpoint type)**
   2. Amazon EFS
   3. **Amazon DynamoDB (Gateway endpoint type)**
   4. Amazon Glacier
   5. Amazon SQS
2. A company has setup an application in AWS that interacts with DynamoDB. DynamoDB is currently responding in milliseconds, but the application response guidelines require it to respond within microseconds. How can the performance of DynamoDB be further improved? [SAA-C01]
   1. Use ElastiCache in front of DynamoDB
   2. Use DynamoDB inbuilt caching
   3. **Use DynamoDB Accelerator**
   4. Use RDS with ElastiCache instead