Serviceless section

Serviceless is a new paradigm in which the developers donot have to manage servers anymore

Just deploy code and function

**Lambda overview**

* Virtual functions – no servers to manage
* Limited by time – short execution
* Run on demand
* **Scaling automately**

1. Benefits

* Easy pricing
* Intergrated with the whole AWS suite of services
* Intergrated with many programming languages
* Easy monitoring through AWS Cloudwatch
* Easy to get more resources per functions (up to 10GB of RAM)
* Increasing RAM will also improve CPU and network

How lambda intergration

Step1: API gateway create a rest API, envolve lambda function

Step 2: Kinesis use lambda to use data tranformation on the flight

Step 3: Dynamodb will be used to create some triggers so whenever sth happen on database, the lambda function will be triggered

Step 4: Aws s3: the lambda function will be triggered

….

1. Serverless thumbmail creation
2. **Lambda function hand- on**

Import json

Print (‘loading function ’)

Def lanbda\_handler( event, context):

#print (“Received event: “ + json.dumps(event, indent=2))

Print(“value1 =” +event[‘key1’])

Print(“value2 =” +event[‘key2’])

Print(“value3 =” +event[‘key3’])

Return event[“key1]

1. Lambda limits to know – per region

* Memory alloction: 128 MB – 10 GB (1 MB increment)
* Maximum execution time: 900 seconds (15 minutes)
* Environment variables (4KB)
* Disk capacity in the “function container” (in/tmp):512MB

**Dynamodb**

* Fully managed, highly available with replication across multiple Azs
* NoSQL database
* Scales to massive workloads, distributed database
* Millions of requests per seconds, trillions of row, 100s of TB of storage

1. DynamoDB basics

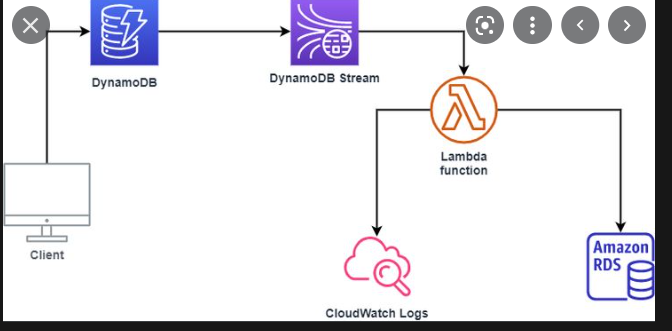
* Dynamodb is made of table (you can manage data through database, but **Dynamodb just only create table**)
* Each table has a primary key (must be decided at creation time)
* Each table can have an infinite number of items (rows)
* Each item has attributes (can be added over time – **can be null**)
* Maxinum size of an item is 400kb (not good for large data)
* Data types: Scalar types (string, number, binary, boolean, null), doccuent types (lists, map), set types (string set, number set, binary set)

1. Dynamodb – read/write capacity modes

* Control how you manage your table’s capacity (read/write throughput)
* Prvisioned mode (default)
  + Specify the number of reads/writes per second
  + Need to plan capacity beforhand
  + Pay for provisioned read capacity units (RCU) & write capacity units (WCU)

1. Dynamodb stream

* Ordered stream of item-level modification (create/update/delete) in a table
* Stream records can be:
  + Sent to Kinessis Data Streams
  + Read by AWS lambda
  + Read by Kinesis Client library application



1. Dynamodb life time

* Automatically delete items after an expiry timestamp (a process)