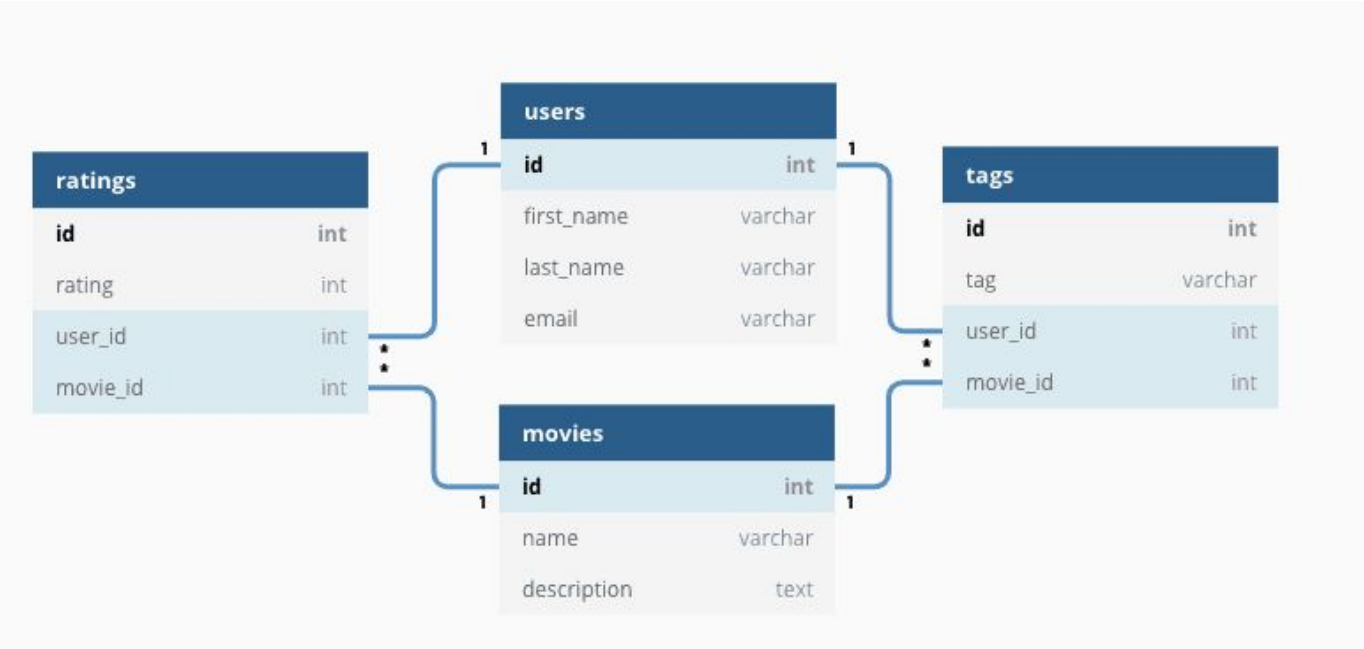
DynamoDB

Part 1

DynamoDB

Fast and flexible NoSQL database service for any scale

Relational Databases

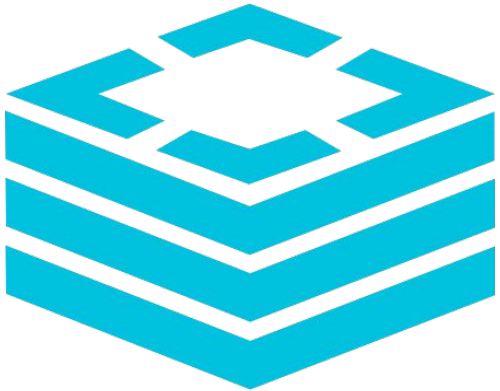


Relational Databases prioritized storage

Non-Relational Databases

Non-relational Databases prioritize other aspects

DynamoDB



Low Latency NoSQL Database
It consists of Tables, Items and Attributes

DynamoDB

Supports key-value and document data models
DynamoDB is based on independent tables

DynamoDB

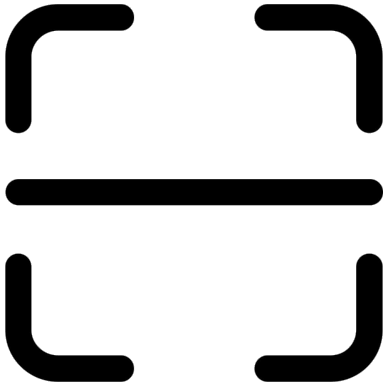
Two types of Keys

- Partition Key
- Sort Key

Capacity Units

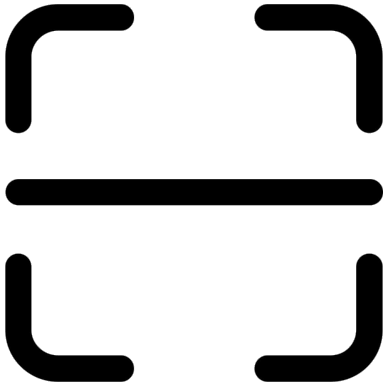
- **Read Capacity Units.-** One consistent read or two eventually consistent reads per second for items of up to 4Kb
- **Write Capacity Units.-** One write per second for items up to 1Kb

Scan Operation



1. Returns all data in a table
2. Results can be filtered, but only once all items have been retrieved
3. Scans can use up the provision throughput for a large table in a single operation.

Scan Operation



4. Retrieves data 1Mb at a time.
5. It can be configured to use Parallel Scan, that improves how long the operation takes

Query Operation

1. Searches data in a table based on a Partition Key and an optional Search Key
2. By default eventually consistent

DynamoDB

DynamoDB Lab #1

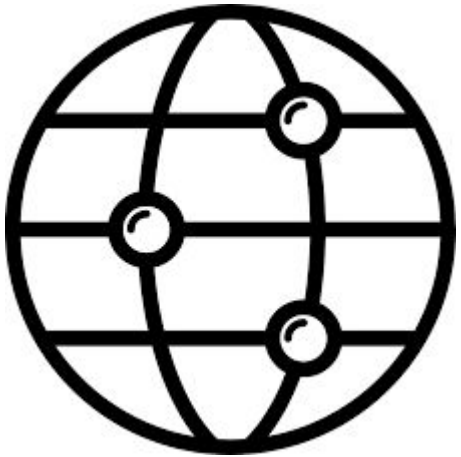
Query and Scan

Local Secondary Index

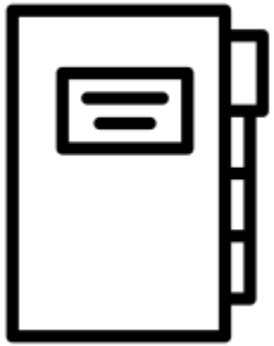
- Can only be created when you are creating your table
- You cannot add, remove or modify it later
- Gives you a different view of your data based on the Sort Key

Global Secondary Index

- Can be created with the main table or later on.
- Provides searching capabilities based on its Partition Key



Indexes



- **Local Secondary Index.-** Same Partition Key different Sort Key
- **Global Secondary Index.-** Different Partition Key

DynamoDB

DynamoDB Lab #2

Local and Global Indexes

Provision Throughput

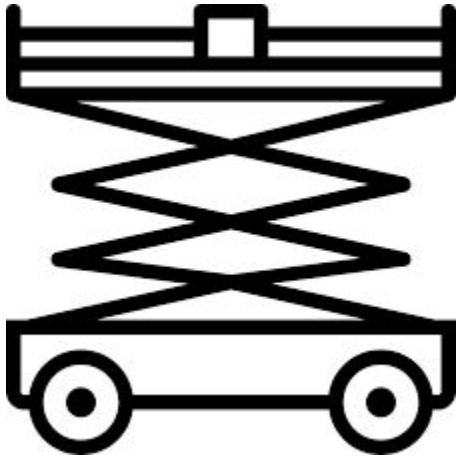
1. Your application needs to make 80 read requests per second
2. Each item is 3Kb in size
3. Consistent Reads are not needed

3Kb/4Kb = 0.75 = 1 RCU per request

1RCU x 80$_{\text{requests/second}}$ x ½ = 40 RCU

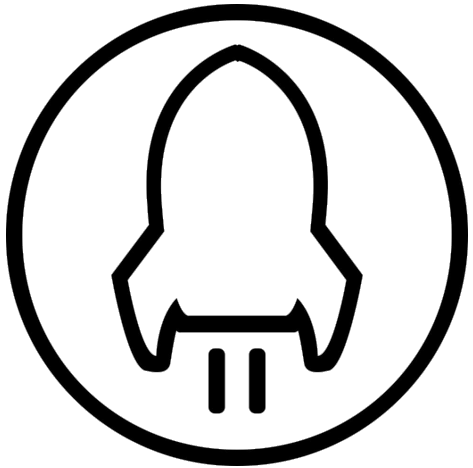
RCU = Read Capacity Unit,
½ because of eventually consistent reads

DynamoDB On-Demand Capacity



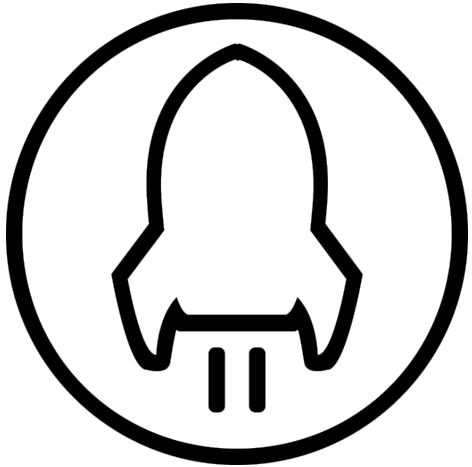
1. DynamoDB scales your capacity units automatically based on the activity of your application
2. Great for unpredictable workloads

DynamoDB Accelerator (DAX)



1. Fully managed, clustered, in-memory cache for DynamoDB
2. Delivers up to 10x performance only for Reads
3. Microsecond response times with million requests per second
4. Perfect for gaming or retail applications

DynamoDB Accelerator (DAX)

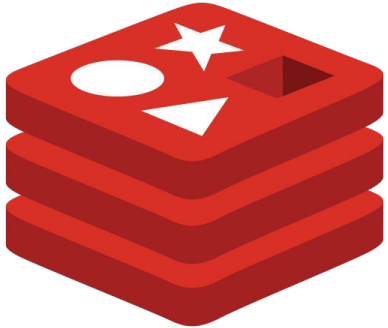


5. DAX is a write-through cache implementation
6. DAX can help you reduce the read load in a DynamoDB table
7. It only works with eventually consistent reads

Elastic Cache

1. In-memory Cache in the cloud
2. It sits between your database and your application
3. Can be used for databases/compute

Types of Elastic Cache



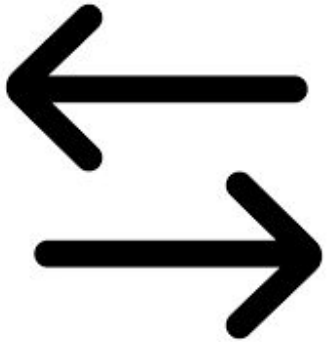
Redis

- Snapshots and Replication
- Complex data structures
- Multi AZ Capability

Memcached

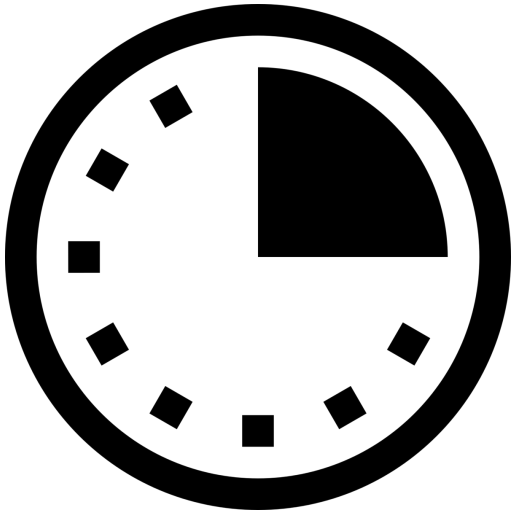
- Multi-Threaded
- No Multi AZ Capability

DynamoDB Transactions



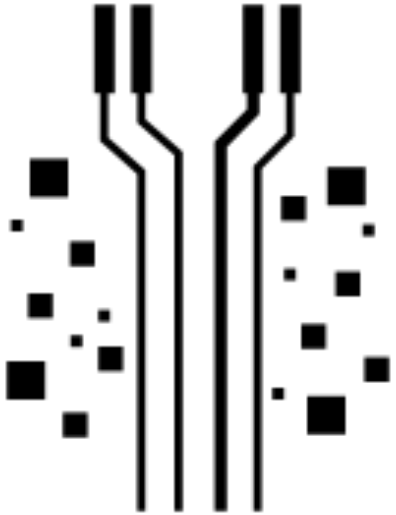
1. ACID Transactions (Atomic, Consistent, Isolated, Durable)
2. All or nothing operation

DynamoDB TTL



1. Items are marked for deletion after a TTL
2. Once marked the item is deleted after 48h

DynamoDB Streams



1. They record operations (insert, update, delete)
2. Before and After images are captured
3. Logs are created and encrypted up to 24h

Provisioned Throughput Exceeded

1. When your request rate is too high
2. SDK will automatically retry requests until successful
3. You should use Exponential Backoff or reduce frequency

DynamoDB Pricing

Pricing