DynamoDB Exam Tips

A CLOUD GURU

- Amazon DynamoDB is a low-latency NoSQL database.
- Consists of Tables, Items, and Attributes
- Supports both document and key-value data models
- Supported document formats are JSON, HTML, XML
- 2 types of Primary Keys: Partition Key and combination of Partition Key + Sort Key (Composite Key)

DynamoDB Exam Tips

A CLOUD GURU

- 2 Consistency models : Strongly Consistent / Eventually Consistent
- Access is controlled using IAM policies.
- Fine grained access control using IAM Condition parameter: **dynamodb:LeadingKeys** to allow users to access only the items where the partition key value matches their user ID

DynamoDB Indexes - Exam Tips

A CLOUD GURU

- Indexes enable fast queries on specific data columns.
- Give you a different view of your data based on alternative Partition / Sort Keys
- Important to understand the differences

Local Secondary Index	Global Secondary Index
Must be created at when you create your table	Can create any time - at table creation or after
Same Partition Key as your table	Different Partition Key
Different Sort Key	Different Sort Key

Scan Vs Query Exam Tips

- A Query operation finds items in a table using only the Primary Key attribute.
- You provide the Primary Key name and a distinct value to search for.
- A Scan operation examines every item in the table.
  - By default, returns all data attributes
- Use the ProjectionExpression parameter to refine the results.

Scan Vs Query Exam Tips

A CLOUD GURU

- Query results are always sorted by the Sort Key (if there is one.)
- Sorted in ascending order
- Set ScanIndexForward parameter to false to reverse the order - queries only
- Query operation is generally more efficient than a Scan.

Scan Vs Query Exam Tips

- Reduce the impact of a query or scan by setting a smaller page size which uses fewer read operations.
- Isolate scan operations to specific tables and segregate them from your mission-critical traffic.
- Try Parallel scans rather than the default sequential scan.
- Avoid using scan operations if you can: design tables in a way that you can use the Query, Get, or BatchGetItem APIs.

DynamoDB Provisioned Throughput Exam Tips

A CLOUD GURU

- Provisioned Throughput is measured in Capacity Units.
- 1 x Write Capacity Unit = 1 x 1KB Write per second.
- 1 x Read Capacity Unit = 1 x 4KB Strongly Consistent Read **OR** 2 x 4KB Eventually Consistent Reads per second.

DynamoDB Provisioned Throughput Exam Tips

A CLOUD GURU

Calculate Write Capacity Requirements (100 x 512 byte items per second):

- First, calculate how many Capacity Units for each write:

  Size of each item / 1KB (for Write Capacity Units)
  512 bytes / 1KB = 0.5

- Rounded-up to the nearest whole number, each write will need 1 x Write Capacity Unit per write operation

- Multiplied by the number of writes per second = 1 x 100 = 100 Write Capacity Units required

DynamoDB Provisioned Throughput Exam Tips

A CLOUD GURU

Calculate Read Capacity Requirements (80 x 3KB items per second):

- First, calculate how many Capacity Units for each read:

  Size of each item / 4KB (for Read Capacity Units)
  3KB / 4KB = 0.75

- Rounded-up to the nearest whole number, each read will need 1 x Read Capacity Unit operation

- Multiplied by the number of reads per second = 1 x 80 = 80 Read Capacity Units required for Strongly Consistent, but if Eventual Consistency is acceptable, divide by 2 = 40 Read Capacity Units required

DAX Exam Tips

A CLOUD GURU

- Provides in-memory caching for DynamoDB tables
- Improves response times tor **Eventually Consistent** reads only.
- You point your API calls to the DAX cluster instead of your table.
- If the item you are querying is on the cache, DAX will return it; otherwise, it will perform an Eventually Consistent GetItem operation to your DynamoDB table.
- Not suitable for write-intensive applications or applications that require Strongly Consistent reads.

Elasticache Exam Tips

A CLOUD GURU

- In-memory cache sits between your application and database
- 2 different caching strategies: Lazy loading and Write Through
- Lazy Loading only caches the data when it is requested
- Elasticache Node failures not fatal, just lots of cache misses
- Cache miss penalty: Initial request, query database, writing to cache
- Avoid stale data by implementing a TTL

Elasticache Exam Tips

A CLOUD GURU

- Write Through strategy writes data into the cache whenever there is a change to the database
- Data is never stale
- Write penalty: Each write involves a write to the cache
- Elasticache node failure means that data is missing until added or updated in the database
- Wasted resources if most of the data is never used