# **Day 35 - 20 September 2025**

**Document Name:**Day 35 - hmuvvala@ - Hari Gopal Muvvala

### **Task 01**

**Question:**

In AWS CLI, how would you retrieve only the name and email fields of an item with primary key UserID = 123?

1. aws dynamodb get-item --table-name Users --key '{"UserID":{"S":"123"}}' --attributes 'name,email'
2. aws dynamodb scan --table-name Users --filter-expression 'UserID = :id' --projection-expression 'name,email'
3. aws dynamodb get-item --table-name Users --key '{"UserID":{"S":"123"}}' --projection-expression "name,email"
4. aws dynamodb fetch --table-name Users --key '{"UserID":"123"}' --fields name,email

**Answer:** 3

**Notes:**

* --projection-expression is the correct way to limit attributes.
* Option 1 is invalid (--attributes doesn’t exist).

### **Task 02**

**Question:**

What is a key advantage of using AWS SDK for interacting with DynamoDB over AWS CLI or Console?

1. SDK bypasses IAM restrictions by using service-linked roles.
2. SDK supports only high-level batch operations.
3. SDK allows embedding retry logic, pagination handling, and structured data models.
4. SDK operations are faster because they skip API endpoints.

**Answer:** 3

**Notes:**

* SDK integrates with application code (retry, pagination).
* CLI/Console are manual, not programmatic.

### **Task 03**

**Question:**

When using AWS Console, how can a user filter data without writing queries?

1. Visual Query Builder.
2. Switch to JSON editor and write expressions.
3. Apply attribute filters under “Explore Table Items”.
4. Enable Streams and subscribe to logs.

**Answer:** 3

**Notes:**

* Console → Explore Table Items → attribute filters.
* No SQL-like query builder exists.

### **Task 04**

**Question:**

What does DynamoDB architecture rely on for high availability and fault tolerance?

1. Vertical scaling.
2. Distributed design with replication across ≥3 AZs.
3. CloudFront caching.
4. Peer-to-peer replication in one AZ.

**Answer:** 2

**Notes:**

* DynamoDB replicates across 3 AZs in a region.

### **Task 05**

**Question:**

What happens when enabling On-Demand Capacity Mode?

1. Disables autoscaling.
2. Billed only for actual reads/writes, no need for capacity units.
3. Double charge per write.
4. Migration to new table required.

**Answer:** 2

**Notes:**

* On-demand = pay-per-request.
* Great for unpredictable workloads.

### **Task 06**

**Question:**

How does CLI handle table creation with provisioned throughput?

1. Auto-detects workload.
2. Requires explicit RCUs and WCUs (with optional autoscaling).
3. Capacity is IAM-only.
4. CLI only supports on-demand mode.

**Answer:** 2

**Notes:**

* Provisioned = must define RCUs + WCUs.
* CLI supports both modes.

### **Task 07**

**Question:**

When adding item in Console, how are attributes typed?

1. Must declare all types in schema.
2. Typed at runtime, stored with metadata.
3. Added dynamically, type inferred at input.
4. Console supports only String type.

**Answer:** 3

**Notes:**

* DynamoDB = schemaless (except keys).
* Attributes + types are flexible.

### **Task 08**

**Question:**

How are partition keys critical to scalability?

1. Act as encryption keys.
2. Decide row-level throughput.
3. Logical grouping, not storage distribution.
4. Map items to partitions → scalability and latency.

**Answer:** 4

**Notes:**

* Partition key = determines physical partition.
* Directly affects scalability.

### **Task 09**

**Question:**

Primary functional difference between LSI and GSI?

1. LSIs replicate table across regions.
2. LSI = same partition key, GSI = different partition key.
3. LSI faster writes.
4. GSIs allow duplicate sort keys.

**Answer:** 2

**Notes:**

* LSI tied to base table’s partition key.
* GSI can define new partition key.

### **Task 10**

**Question:**

What is NOT required when creating a table via Console?

1. Defining primary key.
2. Specifying IAM role for throughput scaling.
3. Choosing capacity mode.
4. Setting RCUs/WCUs if provisioned.

**Answer:** 2

**Notes:**

* IAM role not required for scaling at creation.

### **Task 11**

**Question:**

Which CLI command enables PITR?

1. aws dynamodb enable-pitr ...
2. aws dynamodb update-table ... --pitr true
3. aws dynamodb update-continuous-backups --table-name Customers --point-in-time-recovery-specification PointInTimeRecoveryEnabled=true
4. aws dynamodb backup-table ... --pitr-enabled

**Answer:** 3

**Notes:**

* PITR uses update-continuous-backups.
* Restores table to any second in last 35 days.

### **Task 12**

**Question:**

Which CloudWatch metric detects hot partitions?

1. ConsumedWriteCapacityUnits grouped by PartitionID.
2. ThrottledRequests with partition dimension.
3. AccountProvisionedThroughputUtilization.
4. PartitionSplitCount.

**Answer:** 1

**Notes:**

* Hot partitions = uneven consumption of WCUs/RCUs.
* Monitor per-partition metrics.

### **Task 13**

**Question:**

Which is the maximum number of LSIs (Local Secondary Indexes) allowed per table?

1. 5
2. 10
3. 20
4. Unlimited

**Answer:** 1 (5)

**Notes:**

* DynamoDB allows **up to 5 LSIs per table** at creation time.
* LSIs share the base table’s **partition key**, but can define an alternate sort key.
* Cannot be added after table creation → must be defined upfront.
* Best for **alternative sort order queries** on same partition key.

### **Task 14**

**Question:**

Which AWS service can replicate DynamoDB tables across regions automatically?

1. DAX
2. Streams
3. Global Tables
4. CloudFront

**Answer:** 3 (Global Tables)

**Notes:**

* **Global Tables** = multi-master replication across regions.
* Updates in any region propagate automatically.
* Good for low-latency, globally distributed apps.
* Relies internally on **Streams** for replication events.

### **Task 15**

**Question:**

How do you ensure an attribute update only occurs if the current value matches a condition?

1. Use UpdateItem with ConditionExpression.
2. Use PutItem with overwrite enabled.
3. Wrap in transaction.
4. Use GetItem + update manually.

**Answer:** 1

**Notes:**

* ConditionExpression → ensures updates happen only if condition holds.
* Example:

SET balance = :newBalance

ConditionExpression: balance = :oldBalance

* Prevents race conditions and accidental overwrites.

### **Task 16**

**Question:**

Which is true about DynamoDB TTL (Time-to-Live)?

1. Deletes items exactly at expiry time.
2. Deletes items eventually (async), not guaranteed immediate.
3. TTL works only with LSIs.
4. TTL requires manual trigger.

**Answer:** 2

**Notes:**

* TTL marks items for **asynchronous deletion**.
* Good for session expiry, cache, logs.
* Deletes are free and don’t consume WCUs.

### **Task 17**

**Question:**

What is the maximum number of GSIs per table?

1. 5
2. 10
3. 20
4. 50

**Answer:** 3 (20)

**Notes:**

* DynamoDB supports **up to 20 GSIs per table**.
* Can be created anytime after table creation.
* GSIs allow **different partition/sort keys** from base table.
* They consume **separate throughput** (if provisioned).

### **Task 18**

**Question:**

How does DynamoDB Streams deliver data to consumers like Lambda?

1. Push-based (immediate delivery).
2. Pull-based (Lambda polls Streams).
3. Periodic snapshots.
4. Only via Kinesis Firehose.

**Answer:** 2

**Notes:**

* Streams = **pull model**. Lambda polls at short intervals.
* Max retention = 24 hours.
* Event-driven design: insert/update/delete → triggers Lambda.

### **Task 19**

**Question:**

What’s the maximum number of items a single Query or Scan can return?

1. 1 MB of data per call.
2. 10 MB.
3. 1000 items.
4. Unlimited.

**Answer:** 1 (1 MB per call)

**Notes:**

* Both Query and Scan return **up to 1 MB** per response.
* Use pagination (LastEvaluatedKey) to fetch more.
* Even if 1 MB is fewer than 1000 items, the limit is **size-based**.

### **Task 20**

**Question:**

What is the purpose of WCU and RCU?

1. WCU = write consistency, RCU = read consistency.
2. Write/Read Capacity Units controlling provisioned throughput.
3. Reserved Capacity Utilization.
4. Weighted Capacity Utilization.

**Answer:** 2

**Notes:**

* **WCU** = 1 KB write per second.
* **RCU** = 4 KB read per second (eventual consistency).
* For strong consistency → costs 2x RCUs.
* Key to estimating cost in provisioned mode.

### **Task 21**

**Question:**

How are indexes billed in DynamoDB?

1. Free with table.
2. Same cost as base table.
3. GSIs have their own RCUs/WCUs and storage costs.
4. Only LSIs incur charges.

**Answer:** 3

**Notes:**

* LSIs share table throughput.
* GSIs = separate throughput + storage billing.
* Important to avoid unnecessary GSIs.

### **Task 22**

**Question:**

What is the maximum item size limit in a transaction?

1. 256 KB
2. 400 KB
3. 1 MB
4. No limit

**Answer:** 2 (400 KB)

**Notes:**

* Transactional writes/reads follow same **400 KB per item limit**.
* Max transaction = 25 items or 4 MB total.

### **Task 23**

**Question:**

Which consistency model does DynamoDB provide by default?

1. Strong consistency.
2. Eventual consistency.
3. Causal consistency.
4. Tunable consistency.

**Answer:** 2

**Notes:**

* **Eventual consistency by default**.
* Strong consistency must be requested.
* Transactions give strict ACID semantics.

### **Task 24**

**Question:**

What is the minimum RCU/WCU you can allocate in provisioned mode?

1. 0
2. 1
3. 5
4. 10

**Answer:** 2 (1 RCU or 1 WCU)

**Notes:**

* Min = 1 RCU and 1 WCU.
* Very small tables can still run on provisioned mode.
* In practice, on-demand is easier for unpredictable workloads.

### **Task 25**

**Question:**

What is the maximum retention period for DynamoDB Streams?

1. 12 hours
2. 24 hours
3. 48 hours
4. 7 days

**Answer:** 2 (24 hours)

**Notes:**

* DynamoDB Streams retain changes for **24 hours**.
* Must consume events within that window.
* For longer retention → replicate to Kinesis or S3.

### **Task 26**

**Question:**

Which AWS service provides in-memory acceleration for DynamoDB queries?

1. ElastiCache
2. CloudFront
3. DAX (DynamoDB Accelerator)
4. Redshift Spectrum

**Answer:** 3 (DAX)

**Notes:**

* DAX = fully managed, in-memory cache for DynamoDB.
* Cuts read latency from **milliseconds → microseconds**.
* API-compatible with DynamoDB (drop-in replacement).
* Useful for read-heavy workloads like leaderboards or session stores.

### **Task 27**

**Question:**

Which factor does NOT affect DynamoDB pricing?

1. Read and write throughput
2. Data storage
3. Index usage
4. IAM roles assigned

**Answer:** 4 (IAM roles)

**Notes:**

* Pricing factors: **RCUs/WCUs**, on-demand requests, storage size, GSIs, backups, streams.
* IAM roles = security only, no billing impact.

### **Task 28**

**Question:**

Which CLI command is used to list DynamoDB tables in a region?

1. aws dynamodb list-tables
2. aws dynamodb show-tables
3. aws dynamodb describe-tables
4. aws dynamodb scan-tables

**Answer:** 1

**Notes:**

* list-tables → shows all tables in the region.
* describe-table → details of one table only.

### **Task 29**

**Question:**

What is the maximum number of attributes that can be projected into a GSI?

1. Only partition + sort keys
2. All attributes (using ALL)
3. Only 20 attributes
4. Only string attributes

**Answer:** 2

**Notes:**

* GSI projection types:
  + **ALL** → every attribute
  + **KEYS\_ONLY** → only keys
  + **INCLUDE** → specific subset
* Choose carefully to reduce cost.

### **Task 30**

**Question:**

What is the difference between BatchWriteItem and TransactWriteItems?

1. Both identical; names differ.
2. BatchWriteItem is non-transactional; TransactWriteItems supports ACID guarantees.
3. BatchWriteItem supports conditions; TransactWriteItems does not.
4. BatchWriteItem writes up to 100 items; TransactWriteItems only 10.

**Answer:** 2

**Notes:**

* **BatchWriteItem**: up to 25 put/delete operations, no atomic guarantee.
* **TransactWriteItems**: up to 25 operations with ACID guarantees.

### **Task 31**

**Question:**

What is the maximum number of items BatchGetItem can return?

1. 10
2. 25
3. 100
4. 1000

**Answer:** 3 (100)

**Notes:**

* BatchGetItem = up to **100 items or 16 MB** per request.
* Requires handling unprocessed keys with retry logic.

### **Task 32**

**Question:**

Which backup type allows you to restore DynamoDB to any second in the last 35 days?

1. On-demand backup
2. Snapshot backup
3. PITR (Point-in-Time Recovery)
4. Cross-region backup

**Answer:** 3

**Notes:**

* PITR = continuous backups.
* On-demand = manual snapshot at a point.

### **Task 33**

**Question:**

Which is true about DynamoDB Auto Scaling?

1. Scales instantly to infinite throughput.
2. Adjusts provisioned RCUs/WCUs based on usage.
3. Requires manual CloudWatch setup.
4. Works only for GSIs.

**Answer:** 2

**Notes:**

* Auto Scaling adjusts capacity in **provisioned mode**.
* Uses CloudWatch metrics + Application Auto Scaling.

### **Task 34**

**Question:**

How many strongly consistent reads does 1 RCU provide?

1. 1 × 4 KB
2. 2 × 4 KB
3. 1 × 8 KB
4. 2 × 8 KB

**Answer:** 1 (1 × 4 KB)

**Notes:**

* **1 RCU** = 1 strongly consistent read of 4 KB/sec.
* Eventual consistency = doubles (2 reads per RCU).

### **Task 35**

**Question:**

How does DynamoDB ensure durability of data?

1. Snapshots every minute
2. Multi-AZ synchronous replication
3. Journaling with CloudTrail
4. Using DAX clusters

**Answer:** 2

**Notes:**

* Every write is replicated across **3 AZs** before acknowledged.
* Guarantees durability and high availability.

### **Task 36**

**Question:**

Which AWS service integrates directly with DynamoDB Streams for real-time processing?

1. Redshift
2. Lambda
3. CloudWatch
4. ElastiCache

**Answer:** 2

**Notes:**

* **Lambda** consumes Streams events.
* Enables event-driven architectures (triggers, pipelines).

### **Task 37**

**Question:**

What happens if you try to insert an item larger than 400 KB?

1. Stored across multiple partitions.
2. Request fails with validation error.
3. Truncated to 400 KB.
4. Stored but inaccessible.

**Answer:** 2

**Notes:**

* Hard limit of **400 KB per item**.
* Best practice: offload large objects to **S3**.

### **Task 38**

**Question:**

Which statement about LSIs is true?

1. Can be added any time after table creation.
2. Share throughput with base table.
3. Require separate billing for RCUs/WCUs.
4. Can span multiple partition keys.

**Answer:** 2

**Notes:**

* **LSIs** must be defined at table creation.
* They share base table’s throughput and partition key.

### **Task 39**

**Question:**

Which API is used to check table status after creation?

1. ListTables
2. DescribeTable
3. Scan
4. Query

**Answer:** 2

**Notes:**

* DescribeTable → returns status (CREATING, ACTIVE, DELETING).
* Useful in scripts before performing operations.

### **Task 40**

**Question:**

Which DynamoDB feature can prevent accidental data loss from a table delete?

1. Auto Scaling
2. PITR + On-demand backups
3. LSIs
4. Streams

**Answer:** 2

**Notes:**

* PITR → continuous restore for last 35 days.
* On-demand backup → manual snapshots.
* Together they provide strong recovery options.

### **Task 41**

**Question:**

Which consistency model does DynamoDB transactions use?

1. Eventual consistency only
2. Strong consistency (ACID)
3. Causal consistency
4. Configurable per request

**Answer:** 2 (Strong consistency – ACID)

**Notes:**

* Transactions (TransactWriteItems, TransactGetItems) = full **ACID compliance**.
* Useful for mission-critical operations where correctness > latency.

### **Task 42**

**Question:**

What is the maximum number of items per TransactWriteItems call?

1. 10
2. 25
3. 50
4. 100

**Answer:** 2 (25)

**Notes:**

* Limit: **25 items or 4 MB total**.
* Same limit applies to TransactGetItems.
* Atomic across all items in the request.

### **Task 43**

**Question:**

Which of these is a valid DynamoDB data type?

1. Set (SS, NS, BS)
2. Date
3. Char
4. Float

**Answer:** 1

**Notes:**

* DynamoDB supports **String Set (SS)**, **Number Set (NS)**, **Binary Set (BS)**.
* No native Date → store as String or Number.
* Float stored as Number (precision issues possible).

### **Task 44**

**Question:**

Which API is used to create a new backup of a DynamoDB table?

1. CreateBackup
2. SnapshotTable
3. BackupTable
4. SaveTable

**Answer:** 1

**Notes:**

* CreateBackup = manual on-demand backup API.
* Works without disrupting table performance.

### **Task 45**

**Question:**

What is the default maximum Streams shard count for a DynamoDB table?

1. 1
2. 2
3. 10
4. 100

**Answer:** 1

**Notes:**

* By default, one shard is allocated.
* Scaling depends on write throughput (one shard ≈ 1 MB/sec write capacity).

### **Task 46**

**Question:**

Which statement about parallel scans is true?

1. They divide the table into segments and scan them concurrently.
2. They provide strongly consistent reads.
3. They eliminate the 1 MB scan limit.
4. They automatically balance hot partitions.

**Answer:** 1

**Notes:**

* Parallel scans = divide table into **segments** → scan in parallel.
* Useful for large tables, but still limited by overall RCU capacity.

### **Task 47**

**Question:**

What happens if you exceed provisioned throughput?

1. Extra charges applied automatically.
2. Requests are throttled (ProvisionedThroughputExceededException).
3. Table auto-scales instantly without config.
4. Items are dropped silently.

**Answer:** 2

**Notes:**

* If reads/writes > provisioned, DynamoDB **throttles requests**.
* Best practice: enable Auto Scaling or use On-demand mode.

### **Task 48**

**Question:**

Which feature enables DynamoDB to integrate with analytics tools like Redshift?

1. DynamoDB Export to S3
2. DAX
3. Streams
4. TTL

**Answer:** 1

**Notes:**

* DynamoDB → export table data to **S3** (no performance impact).
* Then query via Athena or load into Redshift.

### **Task 49**

**Question:**

Which IAM policy is needed to allow an app to perform GetItem on a table?

1. dynamodb:\*
2. dynamodb:GetItem
3. dynamodb:Query
4. dynamodb:ReadData

**Answer:** 2

**Notes:**

* IAM must grant exact action: dynamodb:GetItem.
* Least-privilege principle → only allow required actions.

### **Task 50**

**Question:**

What is the maximum number of partitions DynamoDB can create for a single table?

1. 10
2. 100
3. 10,000
4. Practically unlimited (based on scaling)

**Answer:** 4

**Notes:**

* DynamoDB auto-scales partitions as data/throughput grows.
* No fixed upper limit → managed internally by AWS.

### **Task 51**

**Question:**

Which of the following is NOT supported by DynamoDB condition expressions?

1. attribute\_exists
2. begins\_with
3. between
4. substring

**Answer:** 4

**Notes:**

* Supported: attribute\_exists, attribute\_not\_exists, begins\_with, between, etc.
* No direct string functions like substring.

### **Task 52**

**Question:**

Which is the recommended way to store time series data in DynamoDB?

1. One table per day.
2. Use composite key with DeviceID (partition key) and Timestamp (sort key).
3. Use LSIs for each time interval.
4. Store all timestamps as a single list attribute.

**Answer:** 2

**Notes:**

* Best practice: **DeviceID + Timestamp** (composite key).
* Efficient for queries on time ranges.
* Avoid creating many small tables or overloading one key.

### **Task 53**

**What is Boto3?**

* Official AWS SDK for Python.
* Allows developers to create, configure, and manage AWS services via Python code.
* Developed and maintained by AWS.
* Used heavily in automation, scripts, and backend integrations with AWS.

**Notes:**

* Equivalent of AWS SDK for Java, but in Python.
* Common in Data Engineering, ML pipelines, and automation scripts.

### **Task 54**

**Question:**

How would you use Boto3 to query the Orders table by CustomerID, using a secondary index CustomerIndex and retrieve only OrderID and Amount attributes?

1. table.get\_item(Key={'CustomerID': customer\_id}, ProjectionExpression="OrderID,Amount", IndexName="CustomerIndex")
2. table.query(KeyConditionExpression=Key('CustomerID').eq(customer\_id), ProjectionExpression="OrderID,Amount", IndexName="CustomerIndex")
3. dynamodb.query(TableName="Orders", KeyCondition={'CustomerID': {'S': customer\_id}}, Index="CustomerIndex", ReturnFields="OrderID,Amount")
4. table.scan(FilterExpression="CustomerID = :id", ExpressionAttributeValues={":id": customer\_id}, Attributes="OrderID,Amount", IndexName="CustomerIndex")

**Answer:** 2

**Notes:**

* query() is the correct method to use with secondary indexes.
* KeyConditionExpression specifies the partition key condition.
* ProjectionExpression limits attributes returned.

### **Task 55**

**Question:**

What is the correct Boto3 code to perform a scan operation with a filter where score > 80 in Students table?

1. table.scan(FilterExpression=Key('score').gt(80))
2. table.get\_item(Filter='score > 80')
3. table.scan(FilterExpression=Attr('score').gt(80))
4. table.query(FilterExpression=Attr('score').gt(80))

**Answer:** 3

**Notes:**

* Attr is used for non-key attribute conditions.
* Key is only for partition/sort key conditions in queries.
* Correct usage:

from boto3.dynamodb.conditions import Attr

table.scan(FilterExpression=Attr('score').gt(80))

### **Task 56**

**Question:**

In Boto3, how would you read an item from Books table with a key {"ISBN": "123-456-789"}?

1. table.read({"ISBN": "123-456-789"})
2. table.get\_item(Key={"ISBN": "123-456-789"})
3. table.fetch\_item(Key={"ISBN": "123-456-789"})
4. dynamodb.get(Key={"ISBN": "123-456-789"})

**Answer:** 2

**Notes:**

* Correct method: get\_item() with Key parameter.
* Example:

response = table.get\_item(Key={"ISBN": "123-456-789"})

item = response['Item']

* read/fetch\_item are invalid methods.