Developing Flexible NoSQL Solutions

AWS DATABASE OPTIONS · KEY CONCEPTS · PARTITIONS · SECONDARY INDEXES · THROUGHPUT · STREAMS & GLOBAL TABLES · BACKUP · SDK OPERATIONS

STUDY & REVIEW

01 MUST KNOW

if you remember five things, remember these

01 **Two database families: relational (SQL)** stores rows and columns under a *fixed* schema and scales **vertically**; **nonrelational (NoSQL)** stores key-value, document, or graph data under a *dynamic* schema and scales **horizontally**. **Amazon DynamoDB** is a fully managed key-value and document (NoSQL) database with single-digit millisecond performance and ACID transactions.

02 **Components and keys:** a **table** holds **items** (like rows); each item is a set of **attributes** (like columns) and can be up to **400 KB**. Every table has a **primary key** — either a **simple** key (*partition key* only) or a **composite** key (*partition key* + *sort key*). No two items can share the full primary key.

03 **Partitions follow the partition key:** DynamoDB hashes the **partition key** to choose which SSD-backed partition (auto-replicated across AZs) stores an item. With a **composite** key, items sharing a partition key are stored together and **ordered by sort key** — two items may share a partition-key value only if their **sort keys differ**.

04 **GSI vs. LSI:** a **global secondary index** uses a *different* partition and sort key, has its **own throughput**, no size limit, and up to **20** per table; a **local secondary index** keeps the *same* partition key with a *different* sort key, **shares** the table's throughput, is capped at **10 GB** per partition-key value, and up to **5** per table.

05 **Capacity and consistency: 1 RCU** = one *strongly consistent* read/sec of an item ≤ 4 KB (or **two** *eventually consistent* reads); **1 WCU** = one 1-KB write/sec. Pick **provisioned** (set RCU/WCU) or **on-demand** (pay per request) mode. **Eventually consistent** reads may be stale; **strongly consistent** reads return the latest data but are **not** supported on GSIs.

02 KEY TERMS

TERM	DEFINITION
Item	A group of attributes uniquely identifiable among all other items in a table; analogous to a row. A table holds zero or more items.
Attribute	A fundamental data element that needs no further breakdown; analogous to a column. Items in one table can have different attributes.
Primary key	Uniquely identifies each item; either a simple (partition key) or composite (partition + sort key) key. No two items share it.
Partition key	Also called the hash attribute; its value is hashed to determine which partition stores the item.
Sort key	Also called the range attribute; orders items that share a partition-key value within a partition.
Secondary index	An alternate key that lets you query non-primary-key attributes; holds the alternate key, primary key, and optional projected attributes.
Global secondary index (GSI)	Index with partition and sort keys different from the base table; own throughput, no size limit, queries span all partitions. Up to 20 per table.
Local secondary index (LSI)	Index with the same partition key but a different sort key; shares table throughput; ≤ 10 GB per partition-key value. Up to 5 per table.
Read capacity unit (RCU)	One strongly consistent read/sec for items up to 4 KB (or two eventually consistent reads); throughput divides evenly among partitions.
Write capacity unit (WCU)	One write per second for items up to 1 KB.
DynamoDB Streams	Optional feature capturing a time-ordered sequence of item-level changes (records grouped into shards), retained 24 hours; consumable by services such as AWS Lambda.

TERM	DEFINITION
Global table	A collection of replica tables in multiple AWS Regions that DynamoDB keeps synchronized for a multi-Region, multi-active database.

03 TWO PRIMARY-KEY TYPES

every table has exactly one

SIMPLE PRIMARY KEY

partition key only

- One attribute — the **partition key** (hash attribute)
- DynamoDB builds an **unordered** (hash) index on it
- Partition-key value **must be unique** per item
- Example: `SensorLocation` on `SensorId`

COMPOSITE PRIMARY KEY

partition key + sort key

- Two attributes — **partition key** + **sort key**
- Unordered partition index; **sorted** sort-key index
- Same partition key OK if **sort keys differ**
- Example: `SensorReadings` on `SensorId` + `Time`

04 GLOBAL VS. LOCAL SECONDARY INDEX

a frequent exam distinction

ASPECT	GLOBAL SECONDARY INDEX (GSI)	LOCAL SECONDARY INDEX (LSI)
Keys	Partition **and** sort key both differ from the base table	**Same** partition key; **different** sort key
Throughput	Its own provisioned throughput, separate from the table	Shares the table's provisioned throughput
Size limit	None	≤ 10 GB indexed items per partition-key value
Query scope	Spans all data across all base-table partitions	Scoped to one base-table partition-key value
Max per table	20 (default limit)	5

05 DYNAMODB API OPERATIONS

four categories

CATEGORY	PURPOSE	EXAMPLES
Control	Create and manage tables, indexes, and streams	`CreateTable`
Data	Create, read, update, delete items	Write: `PutItem`, `UpdateItem`, `DeleteItem` · Read: `GetItem`, `Query`, `Scan`
Batch	Get or write batches of items across tables	`BatchGetItem`, `BatchWriteItem`
Transaction	Coordinated, all-or-nothing multi-item changes	`TransactWriteItems`, `TransactGetItems`

06 SIZING AN RCU (EVENTUAL READ)

20 reads/sec of 11–KB items

01 Round the 11-KB item up to the next 4-KB multiple: **12 KB** → 02 Divide by 4 KB per RCU: **3** RCUs per item (strong) → 03 Multiply by 20 items/second: **60** → 04 Halve for eventual consistency: **30 RCUs**

07 EXAM TRAPS

where the knowledge check gets you

- × "Two items can never share a partition-key value." — Only true for a **simple** key. With a **composite** key, items may share a partition key as long as their **sort keys differ**.
- × "Eventually consistent reads always return the latest data." — They may return **stale** data immediately after a write; repeat the read, or use a **strongly consistent** read, to get the most up-to-date data.
- × "A strongly consistent read costs the same as an eventually consistent read." — A strong read costs **twice** as much: 1 RCU = one strong read/sec **or two** eventually consistent reads/sec (items ≤ 4 KB).
- × "You can request strongly consistent reads from a global secondary index." — **Consistent reads are not supported on GSIs.** They return eventually consistent data only.
- × "A local secondary index has its own throughput and no size limit." — That describes a **GSI**. An **LSI shares** the table's throughput and is capped at **10 GB** of indexed items per partition-key value.
- × "Global tables give strongly consistent reads across Regions." — DynamoDB does **not** support strongly consistent reads across Regions; do strong reads and writes in the **same Region**, or a cross-Region read may be stale.
- × "A Scan is more efficient than a Query." — **Query** is more efficient; **Scan reads every item** in the table or index. Use a `Query` with a key condition, or a parallel scan if a scan is unavoidable.
- × "`PutItem` only creates new items." — `PutItem` creates a new item **or replaces (overwrites)** an existing item that has the same primary key.

08 SELF-QUIZ

answers are upside-down below

Q1 Which primary-key type is composed of a partition key plus a sort key?

Q2 DynamoDB runs a hash function on which attribute value to choose an item's partition?

Q3 In a composite-key table, two items with the same partition-key value must differ in what?

Q4 A local secondary index shares which key with its base table, and what must be different?

Q5 Which secondary index type has its own provisioned throughput and no size limit?

Q6 How many eventually consistent reads per second of ≤ 4 KB items does one RCU provide?

Q7 One WCU supports how many 1-KB writes per second?

Q8 Which read type returns the most up-to-date data, reflecting all prior successful writes?

Q9 Which operation reads all items from a table or index and is less efficient than a Query?

Q10 Sample exam: every table update must be sent to AWS Lambda so it can notify Operations. What do you enable on the table?

ANSWER KEY (rotate the page)

Q1 the composite primary key **Q2** the partition key **Q3** their sort key values **Q4** same partition key; a different sort key **Q5** a global secondary index (GSI) **Q6** two (one RCU = one strongly consistent read/sec) **Q7** one **Q8** a strongly consistent read **Q9** Scan **Q10** DynamoDB Streams