DynamoDB Hands-On Tasks

What is DynamoDB?

Amazon **DynamoDB** is a fully managed **NoSQL** (non-relational) database service provided by AWS. It allows for **key-value** and **document-based** storage models. It is designed for **high availability, scalability**, and **low-latency performance**.

Key Concepts

Term	Description
Table	A collection of items
Item	A single row in a table (like a record)
Attribute	A single piece of data in an item (like a column)
Primary Key	Uniquely identifies each item (can be a simple or composite key)
Partition Key	Used to distribute data across partitions
Sort Key	Optional secondary part of the primary key
Global Secondary Index (GSI)	Queryable index on any attribute
Local Secondary Index (LSI)	Index on the sort key only

Difference between MySQL & DynamoDB

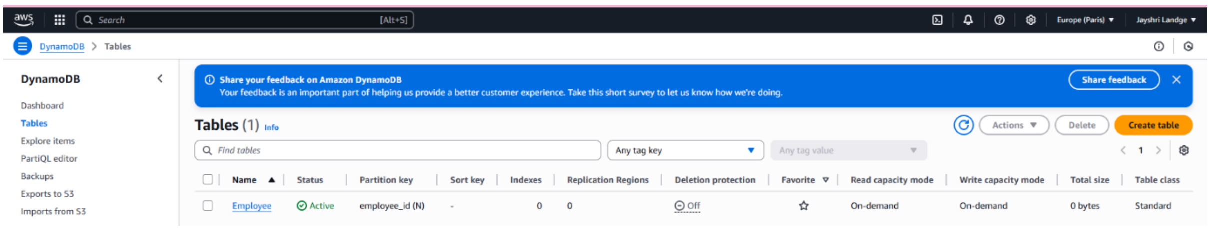
Feature	MySQL	DynamoDB
Type	Relational	Non-relational
Data Structure	Tables with rows and columns	Key-value pairs
Schema	Strict schema	Schema-less
Query Language	SQL	A PartiQL (SQL-like) / DynamoDB API
Performance	Scales vertically	Scales horizontally (very fast)
Interface	Command Line Interface (CLI)	GUI via AWS Console
Data Joins	Supported	Not supported
Transactions	Strong support	Limited, but available

Steps for Hands-On DynamoDB

Step 1: Create a Table

Go to **AWS Console → DynamoDB → Tables → Create table**

- Table name: Users
- Partition key: UserID (String)
- Leave rest as default



Share your feedback on Amazon DynamoDB
Your feedback is an important part of helping us provide a better customer experience. Take this short survey to let us know how we're doing.

Tables (1)

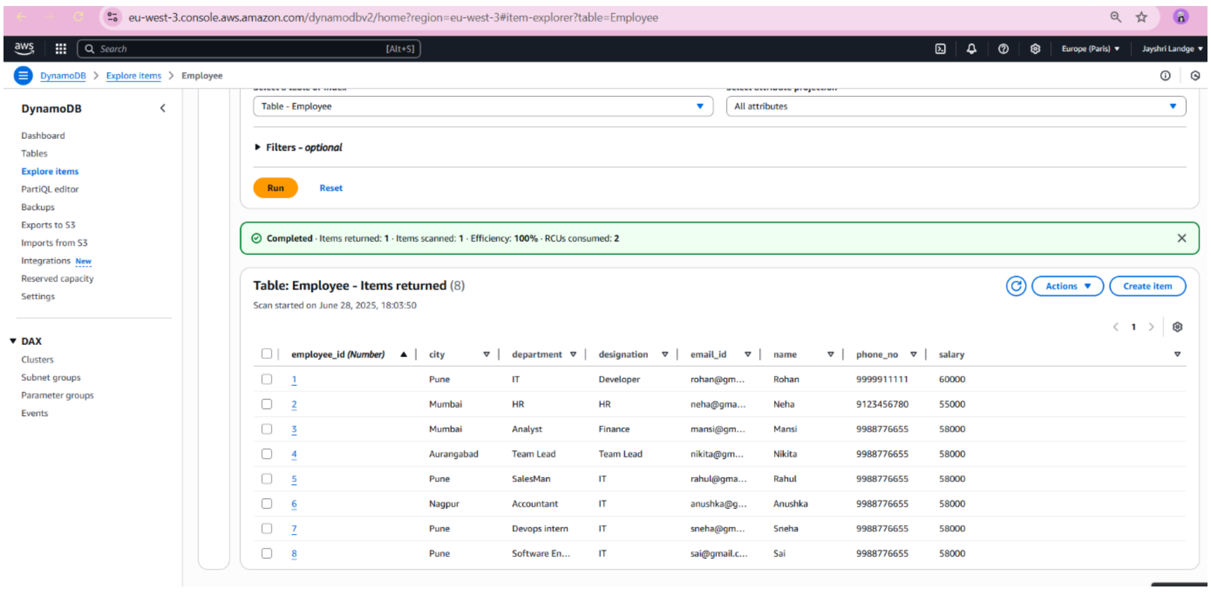
Name	Status	Partition key	Sort key	Indexes	Replication Regions	Deletion protection	Favorite	Read capacity mode	Write capacity mode	Total size	Table class
Employee	Active	employee_id (N)	-	0	0	Off	☆	On-demand	On-demand	0 bytes	Standard

Screenshot 1: Create Table

Step 2: Add Items (Data)

After creating a table:

- Select the Users table → Click on **Explore Table Items** → **Create item**



eu-west-3.console.aws.amazon.com/dynamodbv2/home?region=eu-west-3#item-explorer?table=Employee

Table - Employee
All attributes

Completed · Items returned: 1 · Items scanned: 1 · Efficiency: 100% · RCUs consumed: 2

Table: Employee - Items returned (8)
Scan started on June 28, 2025, 18:03:50

employee_id (Number)	city	department	designation	email_id	name	phone_no	salary
1	Pune	IT	Developer	rohan@gm...	Rohan	9999911111	60000
2	Mumbai	HR	HR	neha@gma...	Neha	9123456780	55000
3	Mumbai	Analyst	Finance	mansi@gm...	Mansi	9988776655	58000
4	Aurangabad	Team Lead	Team Lead	nikita@gm...	Nikita	9988776655	58000
5	Pune	SalesMan	IT	rahul@gma...	Rahul	9988776655	58000
6	Nagpur	Accountant	IT	anushka@g...	Anushka	9988776655	58000
7	Pune	Devops intern	IT	sneha@gm...	Sneha	9988776655	58000
8	Pune	Software En...	IT	sai@gmail.c...	Sai	9988776655	58000

Screenshot 2: Add Item

Step 3: Update an Item

- Select the item → Click **Edit**
- Change Name to "Jayshri ", City "Chh SambhajiNagar", Designation"HR",phone_no"9797979797" → Click **Save**

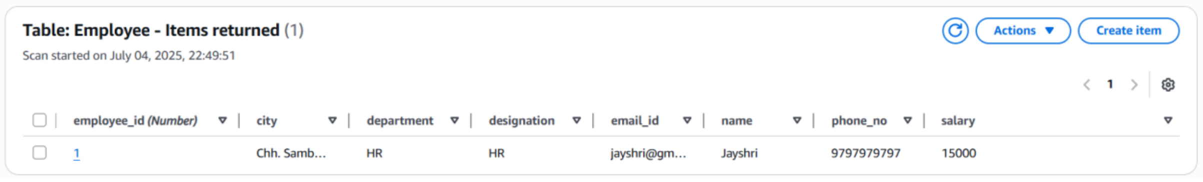


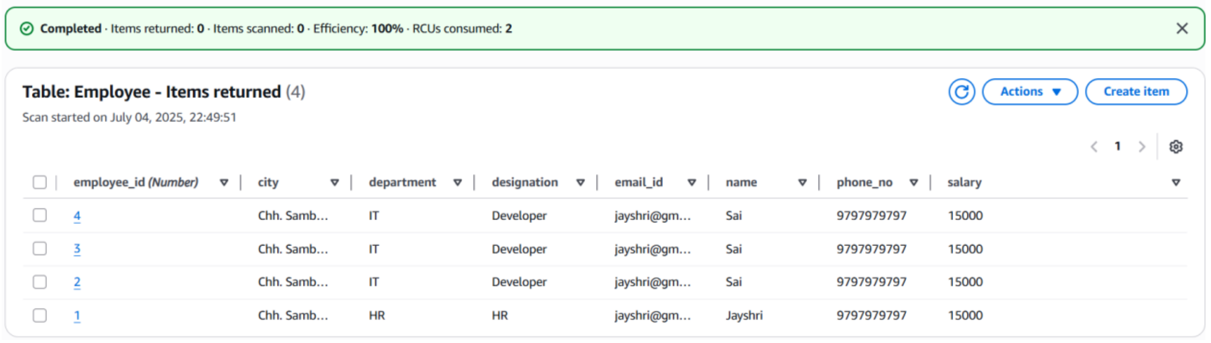
Table: Employee - Items returned (1)
Scan started on July 04, 2025, 22:49:51

employee_id (Number)	city	department	designation	email_id	name	phone_no	salary
1	Chh. Samb...	HR	HR	jayshri@gm...	Jayshri	9797979797	15000

Screenshot 3: Update Item

Step 4: Delete an Item

- Select the item → Click **Actions → Delete** → Confirm



Completed · Items returned: 0 · Items scanned: 0 · Efficiency: 100% · RCUs consumed: 2

Table: Employee - Items returned (4)
Scan started on July 04, 2025, 22:49:51

employee_id (Number)	city	department	designation	email_id	name	phone_no	salary
4	Chh. Samb...	IT	Developer	jayshri@gm...	Sai	9797979797	15000
3	Chh. Samb...	IT	Developer	jayshri@gm...	Sai	9797979797	15000
2	Chh. Samb...	IT	Developer	jayshri@gm...	Sai	9797979797	15000
1	Chh. Samb...	HR	HR	jayshri@gm...	Jayshri	9797979797	15000

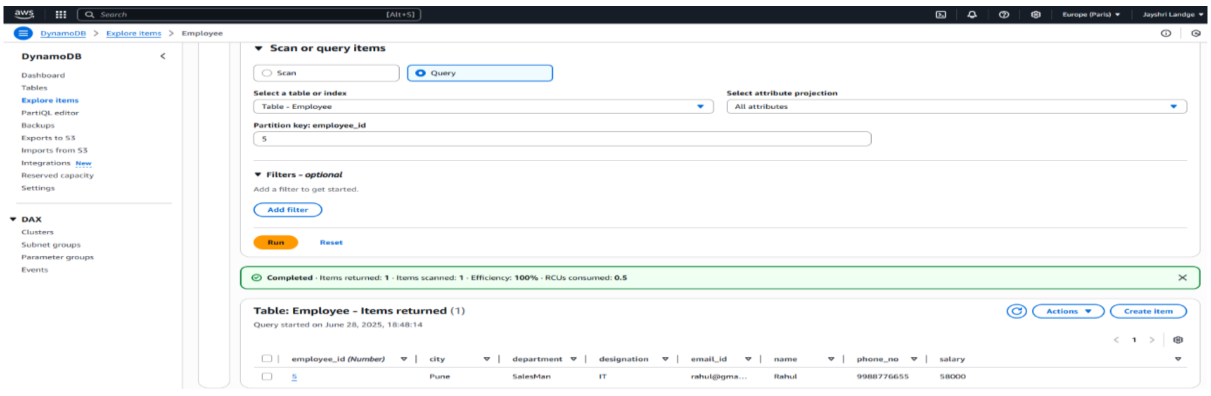
Screenshot 4: Delete Item

Step 5: Query and Scan Data

- Click **Explore Table Items**
- Choose between **Query** (based on partition/sort key) or **Scan** (full table)

Example Query:

- Partition key: employee_id
- Value: 5


DynamoDB > Explore items > Employee

Scan or query items

Scan [✓ Query]

Select a table or index: Table - Employee

Select attribute projection: All attributes

Partition key: employee_id: 5

Filters - *optional*

Add a filter to get started.

Completed · Items returned: 1 · Items scanned: 1 · Efficiency: 100% · RCUs consumed: 0.5

Table: Employee - Items returned (1)

Query started on June 28, 2025, 18:48:14

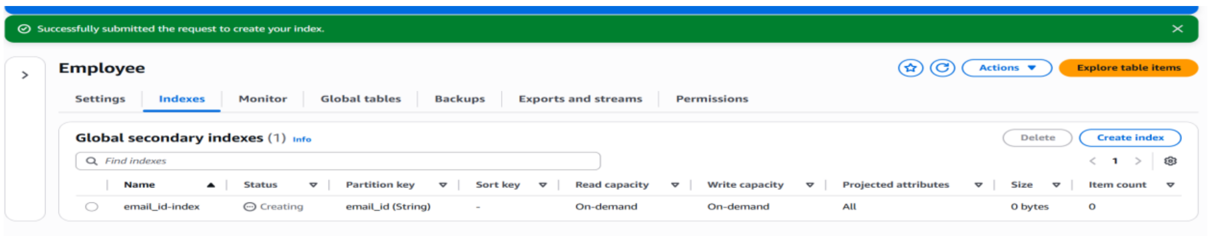
employee_id *(Number)*	city	department	designation	email_id	name	phone_no	salary
5	Pune	SalesMan	IT	rahul@gma...	Rahul	9988776655	58000

Screenshot 5: Query And Scan Data

Step 6: Create a Global Secondary Index (GSI)

- Go to the Indexes tab → **Create Index**
- Partition key: Email
- Click **Create**

Now you can query users by their email addresses.


Successfully submitted the request to create your index.

Employee

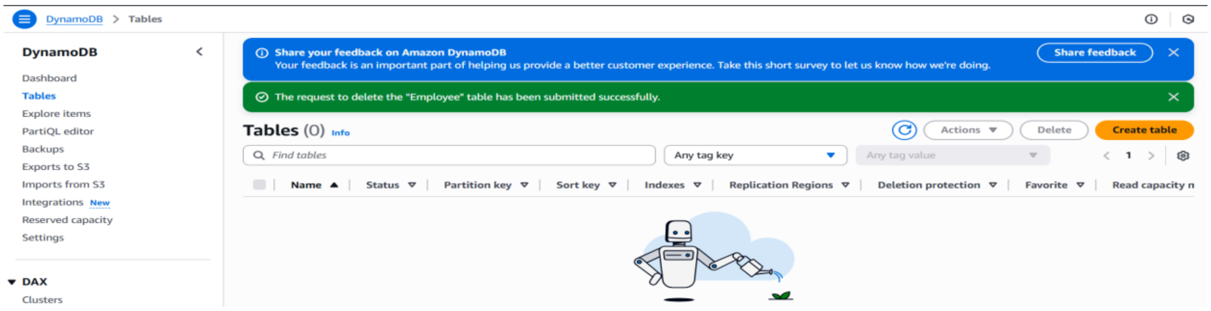
Global secondary indexes (1)

Name	Status	Partition key	Sort key	Read capacity	Write capacity	Projected attributes	Size	Item count
email_id-index	Creating	email_id (String)	-	On-demand	On-demand	All	0 bytes	0

Screenshot 6: Create Global Secondary Index

Step 7: Delete the Table

- Select your table → **Actions → Delete Table**
- Confirm deletion


DynamoDB > Tables

Share your feedback on Amazon DynamoDB

Your feedback is an important part of helping us provide a better customer experience. Take this short survey to let us know how we're doing.

The request to delete the "Employee" table has been submitted successfully.

Tables (0)

Name	Status	Partition key	Sort key	Indexes	Replication Regions	Deletion protection	Favorite	Read capacity n

Screenshot 7: Delete Table

Additional Notes

- DynamoDB does **not support joins** or complex transactions like MySQL.
- Suitable for applications that require **high throughput** and **horizontal scaling**.
- Can integrate with **Lambda**, **API Gateway**, and **S3** for serverless applications.
- **DynamoDB Streams** can capture table activity in real-time.
- You can write data with TTL (Time-to-Live) to auto-delete records.

Conclusion

- **DynamoDB** is ideal for applications requiring **low-latency and high scalability**, such as real-time data, mobile apps, gaming, or IoT.
- It is **schema-less**, which makes it flexible, but also requires **careful design of keys and access patterns**.
- Compared to MySQL, it is **faster for unstructured data** and **scales horizontally**, but lacks the traditional relational features like joins or complex transactions.
- **Hands-on** AWS experience with DynamoDB helps in understanding modern database approaches in the cloud ecosystem.