Hands-On Objective: Azure Cosmos DB (NoSQL) – Account, Database, Container, and Items

---

Objective

The objective of this hands-on exercise is to create an Azure Cosmos DB account using the Azure portal, configure a database and container using the NoSQL API, insert items (documents) into the container, and query the data using the Azure portal Data Explorer.

Goals of the Hands-On

1. Create an Azure Cosmos DB account
2. Create a database in Azure Cosmos DB
3. Create a container with a partition key
4. Insert items (documents) into the container
5. Query and view data using Data Explorer

A) Create Azure Cosmos DB Account

1. Logged in to the Azure portal.
2. Selected **Create a resource** from the Home page.
3. Searched for **Azure Cosmos DB** and selected **Create → Azure Cosmos DB**.
4. Chose **Azure Cosmos DB for NoSQL**.

Basic Configuration

Setting	Value
Subscription	Selected active subscription
Resource Group	New or existing resource group
Account Name	Unique lowercase name

Setting	Value
Location	Region closest to users
Capacity Mode	Provisioned throughput
Free Tier	Applied
Limit Total Account Throughput	Enabled

B) Global Distribution Configuration

The default values were retained:

Setting	Value
Geo-Redundancy	Disabled
Multi-region Writes	Disabled
Availability Zones	Disabled

Optional settings such as Networking, Backup Policy, Encryption, and Tags were left as default.

The configuration was reviewed and the account was created successfully.

C) Access Azure Cosmos DB Account

After deployment completion, **Go to resource** was selected to navigate to the Azure Cosmos DB account dashboard.

D) Create Database and Container using Data Explorer

1. Selected **Data Explorer** from the left navigation pane.
2. Clicked **New Container**.

Database and Container Configuration

Setting	Value
Database id	ToDoList
Share throughput	Enabled
Database Throughput	Manual
Max RU/s	400
Container id	Items
Partition Key	/category

The database and container were created successfully.

E) Insert Items into the Container

1. Expanded **ToDoList → Items** in Data Explorer.
2. Selected **New Item**.

Sample Document Inserted

```
{
  "id": "1",
  "category": "personal",
  "name": "groceries",
  "description": "Pick up apples and strawberries.",
  "isComplete": false
}
```

3. Clicked **Save**.
4. Added another item with a unique id and custom fields.

Azure Cosmos DB allows flexible schema, so documents can vary in structure.

F) Query Data using Data Explorer

By default, Data Explorer displays the following query:

SELECT * FROM c

This query retrieves all documents from the container.

Modified Query

To view the most recently updated documents:

ORDER BY c._ts DESC

The filter was applied, and the results were displayed successfully.

Result

- Azure Cosmos DB account was created successfully
- A database and container were configured using the NoSQL API
- Items were inserted into the container
- Queries were executed using Data Explorer

Conclusion

This hands-on demonstrated the creation and usage of a schema-less NoSQL database using **Azure Cosmos DB**.
The exercise highlighted how Azure Cosmos DB supports flexible data models, scalable throughput, and interactive querying through the Azure portal.
