Designing and Implementing Cloud-Native Applications Using Azure Cosmos DB (DP-420)

Duration: 4 Days **Course Code: M-DP420**

Overview:

This course teaches developers how to create application using the SQL API and SDK for Azure Cosmos DB. Students will learn how to write efficient queries, create indexing policies, manage and provisioned resources, and perform common operations with the SDK.

Target Audience:

Software engineers tasked with authoring cloud-native solutions that leverage Azure Cosmos DB SQL API and its various SDKs. They are familiar with C#, Python, Java, or JavaScript. They also have experience writing code that interacts with a SQL or NoSQL database platform

Objectives:

- Create and configure Azure Cosmos DB SQL API account, database, and container
- Use the .NET SDK to manage resources and perform operations
- Perform queries of varying complexity
- Design a data modeling and partitioning strategy
- Optimize queries and indexes based on characteristics of an application
- Use the Azure Resource Manager to manage accounts and resources with CLI or JSON and Bicep templates

Prerequisites:

- Experience writing in an Azure-supported language at the intermediate level. (C#, JavaScript, Python, or Java)
- Ability to write code to connect and perform operations on a SQL or NoSQL database product. (SQL Server, Oracle, MongoDB, Cassandra or similar)
- M-AZ900 - Microsoft Azure Fundamentals
- M-DP900 - Microsoft Azure Data Fundamentals

Content:

Module 1: Get started with Azure Cosmos DB SQL API

Modern apps thrive on real-time data from different sources and shaped in different forms. These apps require a modern database that can handle the variety and velocity of data that will be thrown at it. In this module, we will explore Azure Cosmos DB and how the SQL API can solve some of the problems presented by modern applications.

Lessons M1

Introduction to Azure Cosmos DB SQL API

Try Azure Cosmos DB SQL API

Lab : Exercise: Create an Azure Cosmos DB SQL API account

After completing module 1, students will be able to:

Evaluate whether Azure Cosmos DB SQL API is the right database for your application

Describe how the features of the Azure Cosmos DB SQL API are appropriate for modern applications

Create a new Azure Cosmos DB SQL API account

Create database, container, and item resources for an Azure Cosmos DB SQL API

Creating a new Azure Cosmos DB account often requires making a lot of configuration choices that can, at first, be daunting. While the defaults fit a lot of scenarios, it makes the most sense to familiarize yourself with the configuration options to ensure that your account and resources are optimally configured for your solution. In this module, you will learn how to prepare and configure an Azure Cosmos DB account and resources for a new solution.

Lessons M2

After completing module 5, students will be able to:

Create and execute a SQL query

Project query results

Use built-in functions in a query

Implement a corelated subquery

By default, Azure Cosmos DB automatically indexes all paths of documents stored using the SQL API. This is great for developing new applications as you can create complex queries almost immediately. As your application matures, you can customize your indexing policy to better match the needs of your solution. In this module, you will learn how to create a custom indexing policy.

Lessons M6

Define indexes in Azure Cosmos DB SQL API

Customize indexes in Azure Cosmos DB SQL API

Lab : Exercise: Review the default index policy for an Azure Cosmos DB SQL API container with the portal

Lab : Exercise: Configure an Azure Cosmos DB SQL API container's index policy with the portal

After completing module 6, students will be able to:

View and understand the default indexing policy for a SQL API container

Customize the indexing policy for a container

Use a composite index in an indexing

Choosing indexes in Azure Cosmos DB SQL API

Optimize queries in Azure Cosmos DB SQL API

Implement integrated cache

Lab : Exercise: Optimize an Azure Cosmos DB SQL API container's index policy for common operations

Lab : Exercise: Optimize an Azure Cosmos DB SQL API container's index policy for a specific query

After completing module 10, students will be able to:

Review and compare read-heavy vs. write-heavy index patterns

Update indexing policy to optimize index performance

Measure cost of a query in request units (RUs)

Measure cost of point operations

Work with item and query integrated cache

Configure integrated cache

When you have critical applications and business processes relying on Azure resources such as Azure Cosmos DB, you want to monitor those resources for their availability, performance, and operation. In this module, you will explore how to monitor events and performance of an Azure Cosmos DB account. You will also learn how to implement common security measures along with backup and restore in Azure Cosmos DB.

Lessons M11

Measure performance in Azure Cosmos DB

Plan Resource Requirements

Configure Azure Cosmos DB SQL API database and containers

Moving data into and out of Azure Cosmos DB SQL API

Lab : Exercise: Configure throughput for Azure Cosmos DB SQL API with the Azure portal

Lab : Exercise: Migrate existing data using Azure Data Factory

After completing module 2, students will be able to:

Evaluate various requirements of your application

Plan for scale and retention requirements

Configure throughput allocation

Configure time-to-live values

Migrate data using Azure services

There are various SDKs available to connect to the Azure Cosmos DB SQL API from many popular programming languages including, but not limited to .NET (C#), Java, Python, and JavaScript (Node.js). In this module, you will get hands-on with with the .NET SDK for the Azure Cosmos DB SQL API.

Lessons M3

Use the Azure Cosmos DB SQL API SDK

Configure the Azure Cosmos DB SQL API SDK

Lab : Exercise: Configure the Azure Cosmos DB SQL API SDK for offline development

Lab : Exercise: Connect to Azure Cosmos DB

Azure Cosmos DB has tight integration available with many other Azure servicers such as Azure Functions, Azure Cognitive Search, Azure Event Hubs, Azure Storage, Azure Data Factory, and Azure Stream Analytics. Going even further, you can use the change feed to integrate Azure Cosmos DB with many other services both in and out of Azure. In this module, we will integrate Azure Cosmos DB with both Azure Functions and Azure Cognitive Search. We will also explore the change feed using the SDK.

Lessons M7

Consume an Azure Cosmos DB SQL API change feed using the SDK

Handle events with Azure Functions and Azure Cosmos DB SQL API change feed

Search Azure Cosmos DB SQL API data with Azure Cognitive Search

Lab : Exercise: Archive Azure Cosmos DB SQL API data using Azure Functions

Lab : Exercise: Process change feed events using the Azure Cosmos DB SQL API SDK

Lab : Exercise: Archive data using Azure Functions and Azure Cosmos DB SQL API

After completing module 7, students will be able to:

Process change feed events using the SDK

Implement change feed best practices

Create an Azure Functions trigger for Azure Cosmos DB

Create an Azure Functions input for Azure Cosmos DB

Index Azure Cosmos DB data in Azure

Azure Cosmos DB is both horizontally scalable and nonrelational. To achieve this level of scalability, users need to understand

SQL API

Monitor responses and events in Azure Cosmos DB SQL API

Implementing backup and restore for Azure Cosmos DB SQL API

Implement security in Azure Cosmos DB SQL API

Lab : Exercise: Troubleshoot an application using the Azure Cosmos DB SQL API SDK

Lab : Exercise: Use Azure Monitor to analyze an Azure Cosmos DB SQL API account

Lab : Exercise: Recover a database or container from a recovery point

Lab : Exercise: Store Azure Cosmos DB SQL API account keys in Azure Key Vault

After completing module 11, students will be able to:

Observe rate-limiting events in a container or database

Query resource logs using Azure Monitor

Review and observe transient and rate-limiting errors

Configure alerts

Configure continuous backup and recovery

Perform a point-in-time recovery

Use role-based access control (RBAC)

Access account resources using Azure AD

Once an Azure Cosmos DB SQL API account is ready to go through a release lifecycle, it's not uncommon for an operations team to attempt to automate the creation of Azure

SQL API with the SDK

After completing module 3, students will be able to:

Integrate the Microsoft.Azure.Cosmos SDK library from NuGet

Connect to an Azure Cosmos DB SQL API account using the SDK and .NET

Configure the SDK for offline development

Troubleshoot common connection errors

Implement parallelism in the SDK

The SQL API SDK for Azure Cosmos DB is used to perform various point operations, perform transactions, and to process bulk data. In this module, you will use the SDK to manipulate documents either individually or in groups.

Lessons M4

Implement Azure Cosmos DB SQL API point operations

Perform cross-document transactional operations with the Azure Cosmos DB SQL API

Process bulk data in Azure Cosmos DB SQL API

Lab : Exercise: Create and update documents with the Azure Cosmos DB SQL API SDK

Lab : Exercise: Batch multiple point operations together with the Azure Cosmos DB SQL API SDK

Lab : Exercise: Move multiple documents in bulk with the Azure Cosmos DB SQL API SDK

After completing module 4, students will be able to:

the concepts, techniques, and technologies unique to NoSQL databases for modeling and partitioning data. In this module, you will model and partition data appropriately for a NoSQL database such as Azure Cosmos DB SQL API.

Lessons M8

Model and partition your data in Azure Cosmos DB

Optimize databases by using advanced modeling patterns for Azure Cosmos DB

Lab : Exercise: Measure performance for customer entities

Lab : Exercise: Advanced modeling patterns

After completing module 8, students will be able to:

Identify application access patterns for an existing application

Decide when to embed or reference data

Use change feed to manage referential integrity

Combine multiple entities in a single container

Denormalize aggregated data in a single

Today's applications are required to be highly responsive and always online. To achieve low latency and high availability, instances of these applications need to be deployed in datacenters that are close to their users. In this module, you will explore how to replicate data and manage consistency across the globe using Azure Cosmos DB SQL API.

Lessons M9

Configure replication and manage failovers in Azure Cosmos DB

Use consistency models in Azure Cosmos DB

Cosmos DB resources in the cloud. Automation makes it easier to deploy new environments, restore past environments, or scale a service out. In this module, you will explore how to use Azure Resource Manager to manage an Azure Cosmos DB account and its child resources using JSON templates, Bicep templates, or the Azure CLI.

Lessons M12

Write scripts for Azure Cosmos DB SQL API

Create resource template for Azure Cosmos DB SQL API

Lab : Exercise: Adjust provisioned throughput using an Azure CLI script

Lab : Exercise: Create an Azure Cosmos DB SQL API container using Azure Resource Manager templates

After completing module 12, students will be able to:

View arguments, groups, and subgroups for a specific CLI command

Create Azure Cosmos DB accounts, databases, and containers using the CLI

Manage an indexing policy using the CLI

Configure database or container throughput using the CLI

Initiate failovers and manage failover regions using the CLI

Identify the three most common resource types for Azure Cosmos DB SQL API accounts

Create and deploy a JSON Azure Resource Manager template for Azure Cosmos DB SQL API

Create and deploy a Bicep Azure Resource Manager template for Azure Cosmos DB SQL API

Perform CRUD operations using the SDK

Configure TTL for a specific document

Implement optimistic concurrency control for an operation

Create a transactional batch and review results

Create a bulk operation

Review the results of a bulk operation

Implement bulk operation best

The Azure Cosmos DB SQL API supports Structured Query Language (SQL) as a JSON query language. In this module, you will learn how to create efficient queries using the SQL query language.

Lessons M5

Query the Azure Cosmos DB SQL API

Author complex queries with the Azure Cosmos DB SQL API

Lab : Exercise: Paginate cross-product query results with the Azure Cosmos DB SQL API SDK

Lab : Exercise: Execute a query with the Azure Cosmos DB SQL API SDK

SQL API

Configure multi-region write in Azure Cosmos DB SQL API

Lab : Exercise: Configure consistency models in the portal and the Azure Cosmos DB SQL API SDK

Lab : Exercise: Connect to different regions with the Azure Cosmos DB SQL API SDK

Lab : Exercise: Connect to a multi-region write account with the Azure Cosmos DB SQL API SDK

After completing module 9, students will be able to:

Distribute data across various geographies

Define automatic failover policies

Perform manual failovers

Configure default consistency model

Change per-session consistency model

Configure multi-region write in the SDK

Create a custom conflict resolution

Azure Cosmos DB offers a rich set of database operations that operate on the items within a container. The cost associated with each of these operations varies based on the CPU, IO, and memory required to complete the operation. In this module, you will explore how to manage indexing policies and edit queries to minimize per-query request unit (RU) cost.

Lessons M10

Manage throughput and index policies using

Azure Cosmos DB provides language-integrated, transactional execution of JavaScript. When using the SQL API in Azure Cosmos DB, you can write stored procedures, triggers, and user-defined functions (UDFs) in the JavaScript language. In this module, you will author JavaScript logic that executes directly inside the database engine.

Lessons M13

Build multi-item transactions with the Azure Cosmos DB SQL API

After completing module 13, students will be able to:

Expand query and transaction functionality in Azure Cosmos DB SQL API

Lab : Exercise: Implement and then use a UDF using the SDK

Lab : Exercise: Create a stored procedure with the Azure Portal

After completing module 13, students will be able to:

Author stored procedure

Rollback stored procedure transaction

Create UDF

Create pre-* and post-* triggers

Further Information:

For More information, or to book your course, please call us on 0800/84.009

info@globalknowledge.be

www.globalknowledge.com/en-be/