Exam 70-777: Implementing Microsoft Azure Cosmos DB Solutions – Skills Measured

Audience Profile

Candidates for this exam are developers and architects who leverage Azure Cosmos DB. Candidates should understand fundamental concepts of partitioning, replication, and resource governance for building and configuring scalable applications that are agnostic of a Cosmos DB API. Candidates should also have basic working knowledge of the Cosmos DB SQL API.

Candidates for this exam design, build, and troubleshoot Cosmos DB solutions that meet business and technical requirements.

Skills Measured

NOTE: The bullets that appear below each of the skills measured are intended to illustrate how we are assessing that skill. This list is not definitive or exhaustive.

NOTE: In most cases, exams do NOT cover preview features, and some features will only be added to an exam when they are GA (General Availability).

Partition and Model Data

Plan a partitioning strategy

- select a partition key for a container; differentiate between partition keys and partition key ranges; partition data across multiple containers; calculate throughput distribution across partition key ranges; control cross-partition queries; plan for transactions

Model data based on business use cases

- identify when to co-locate data within the same container or across multiple containers; identify when to co-locate data within the same partition key or across multiple partition keys; identify when to co-locate data within the same document or across multiple documents; share properties between documents

Replicate Data Across the World

Implement global distribution and high availability

- replicate data to additional regions; define automatic failover policies for disaster recovery; perform manual failovers to change the write region; set the preferred location of applications for low-latency access; design patterns for multi-write regions; resolve conflicts surfaced by the conflict feed

Select a data consistency approach based on business use cases

- identify use cases for consistencies; differentiate between consistencies by using characteristics; differentiate between consistencies by using trade-offs between performance and consistency; use session tokens

Tune and Debug Azure Cosmos DB Solutions

Estimate and provision request units

- differentiate requests and request units; retrieve request unit cost of an operation; estimate request unit allocation for a container; tune throughput for uneven workloads and manage throttling; monitor Azure portal metrics; recommend solutions based on query metrics

Tune container settings

- manage lifecycle of data by using TTL; tune an index policy; include and exclude properties from index paths

Implement security

- secure access to data; rotate keys; understand encryption at rest and in transit; configure IP firewalls; create and manage users; configure fine-grained access to resources

Debug a Cosmos DB solution

- configure diagnostic logging; recommend solutions based on data retrieved from logs; evaluate response status code categories; throttle; review metrics and performance tips

Perform Integration and Develop Solutions

Develop applications with the SQL API

- optimize SDK and concurrency control; tune request options for CRUD and queries; examine response headers; implement optimistic concurrency control with ETAG; query geospatial data; use advanced SQL query operators for complex documents (nested objects and arrays); perform intra-document JOINs; perform SQL queries; implement user-defined functions; use multi-record transactions with stored procedures; implement

a continuation model for bounded execution with stored procedures; implement server-side logic and transactions

Migrate from MongoDB to MongoDB API in Cosmos DB

- choose appropriate tools to migrate data; transfer data

Implement event-driven applications by using Azure functions, triggers and Cosmos DB change feed

- use Cosmos DB triggers for Azure functions; change feed mechanics

Analyze Cosmos DB data with Apache Spark connector

- set up and configure a Cosmos DB Spark connector; push down predicates