Server-side Programming with Cosmos DB

David Tucker
TECHNICAL ARCHITECT & CTO CONSULTANT
@_davidtucker_ davidtucker.net

Server-side Capabilities

Cosmos DB provides a robust approach for executing code in response to actions taken on the data stored in Cosmos DB. While in some cases these mirror traditional database constructs, they are fundamentally different in implementation and have unique limitations.

Cosmos DB Server-side Concepts

Stored Procedures

Triggers

User Defined Functions (UDF's)

Change Feed

Server-side Execution Environment

Stored procedures, triggers, and UDF's are executed within the database engine

These are supported when using the SQL API

Supports JavaScript

Can be created and managed via the portal and via the SDK

Stored Procedures

- Stored procedure must be defined in JavaScript
- Executes on a single partition, and it only has access to that partition
- Partition key must be provided with the execution request
- Supports a transaction model as all statements will be removed if it fails

Triggers

- Triggers must be defined in JavaScript
- Triggers can be executed either before (pre) of after (post) data is written
- Pre triggers can handle data transformation and validation
- Post triggers can handle aggregation and change notifications
- Triggers are not guaranteed to execute, as they have to be specified in a request
- Errors in either the pre or post trigger will result in data being rolled back

User Defined Functions

User defined functions must be defined in JavaScript

Enables you to define a custom function that can be leveraged in a query

Enables encapsulation of common logic in query conditions

Change Feed Processing

While all other server-side programming approaches enable execution on the Cosmos DB engine, change feed processing enables you to react to data changes using server-side code outside of the Cosmos DB engine.

Change Feed

- Enables you to be notified for any insert and update on your data
- Deletes are not directly supported, but you can leverage a soft-delete flag
- A change will appear exactly once in the change feed
- Reading data from the database will consume throughput
- Partition updates will be in order, but between partitions there is no guarantee
- Is not supported for the Azure Table API

Change Feed Approaches

Azure Functions

Change Feed Processor

Implementations in this Module

Stored Procedures

Triggers

Change Feed
Azure Functions

Implementing a Trigger for a Container

Demo

- Configuring a trigger definition for a container
- Creating the trigger for a container via the SDK
- Utilizing the trigger for data transformation
- Utilizing the trigger for data validation

Implementing a Stored Procedure

Demo

Configuring a stored procedure definition for a container

Creating the stored procedure for a container via the SDK

Utilizing the stored procedure for data transformation

Executing the stored procedure with local data

Change Feed with Azure Functions

Demo

Creating a Function App for Change Feed integration

Creating an Azure Function with a Cosmos DB trigger in the portal

Verifying the Azure Function utilizing the portal