Module 07: Integrate Azure Cosmos DB SQL API with Azure services

In [ ]:
```
using Microsoft.Azure.Cosmos;
using System;
using System.Collections.Generic;

CosmosClient client = new (connectionString);
Database database = client.GetDatabase("cosmicworks");
Container container = database.GetContainer("products");

public class Product
{
        public string id { get; set; }
        public string categoryId { get; set; }
        public string categoryName { get; set; }
        public string sku { get; set; }
        public string name { get; set; }
        public string description { get; set; }
        public double price { get; set; }
}
```

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

Understand change feed features in the SDK

The .NET SDK for Azure Cosmos DB SQL API ships with a change feed processor that simplifies the task of reading changes from the feed.

Component	Description
Monitored container	This container is monitored for any insert or update operations. These changes are then reflected in the feed.
Lease container	The lease container serves as a storage mechanism to manage state across multiple change feed consumers (clients).
Host	The host is a client application instance that listens for and reacts to changes from the change feed.
Delegate	The delegate is code within the client application that will implement business logic for each batch of changes.

Prior to using the change feed processor, you should create a lease container that you will reference when configuring the processor.

Implement a delegate for the change feed processor

For the change feed processor, the library expects a delegate of type ChangesHandler<>. This delegate will handle a list of changes on the feed.

In [ ]:
```
// Create an instance for both the source and lease container.
Container sourceContainer = await database.CreateContainerIfNotExistsAsync("products",
Container leaseContainer = await database.CreateContainerIfNotExistsAsync("productslea
```

In [ ]:
```
// The delegate includes two parameters; a read-only list of changes and a cancellatio
```

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Threading;
using System.Threading.Tasks;
using Microsoft.Azure.Cosmos;
using Microsoft.Azure.Cosmos.Fluent;
using static Microsoft.Azure.Cosmos.Container;

ChangesHandler<Product> changeHandlerDelegate = async (
    IReadOnlyCollection<Product> changes,
    CancellationToken cancellationToken ) =>
        {
            // A foreach loop is used to iterate through the current batch of changes,
            // and print each item to the console window
            foreach(Product product in changes)
            {
                await Console.Out.WriteLineAsync($"Detected Operation:\t[{product.id}]\
                // Do something else with each change
            }
         };
```

Implement the change feed processor

The processor needs to be built with a source and lease container, and then needs to be started.

In [ ]:

```
ChangesHandler<Product> changeHandlerDelegate = async (
    IReadOnlyCollection<Product> changes,
    CancellationToken cancellationToken ) =>
        {
            // A foreach loop is used to iterate through the current batch of changes,
            // and print each item to the console window
```