**What is Cosmos DB**

Azure Cosmos DB is Microsoft's globally distributed, multi-model database service. With the click of a button, Cosmos DB enables you to elastically and independently scale throughput and storage across any number of Azure's geographic regions. You can elastically scale throughput and storage, and take advantage of fast, single-digit-millisecond data access using your favorite API among SQL, MongoDB, Cassandra, Tables, or Gremlin.

**Scalability**

In traditional relational models, implementing distributed storage is non-trivial. In contrast, Cosmos DB offers global distribution capabilities out of the box. Specifically, Cosmos DB automatically replicates data to other regions and as such guarantees data consistency across various regions where the data is replicated. This is known as Turnkey global distribution.

**Performance**

Since Cosmos DB internally implements local caching of data and automatically indexes incoming data, the response times of read/write operations are typically in the order of 10s of milliseconds. Automatic indexing also reduces the operational burden of maintaining indexes.

**Availability**

Azure Cosmos DB transparently replicates your data across all the Azure regions associated with your Cosmos account. The guarantees for high availability provided by Cosmos DB are 99.99% availability for a read and writes of data in a single region and 99.999% availability for a read and writes of data for multi-region accounts with multi region writes.

**Programming models**

Cosmos DB can be integrated into distributed applications and services using one of the various available programming models, thus reducing the ramp-up and onboarding time. These APIs provide program­matic access to create, query, and delete databases, collections, and documents.

1. SQL API: Cosmos DB SQL API allows data access like relational models.
2. Mongo API: Cosmos DB exposes Mongo API in order to facilitate Mongo DB users to easily ramp up on the offering.
3. Gremlin API: Gremlin is a graph traversal language used to interact with graph databases. Cosmos DB exposes Gremlin API to store and interact with graph data stored in Cosmos DB.
4. Cassandra API: Cosmos DB’s Cassandra API enables interaction with data stored in Azure Cosmos DB using the Cassandra Query Language (CQL), Cassandra-based tools, and Cassandra client drivers. In addition, Spark connector is used to connect to Azure Cosmos DB Cassandra API.
5. Table API: Applications that use Azure Tables can be migrated seamlessly to Cosmos DB by using Cosmos DB Table API.

Further, these APIs are exposed via .NET, Java, Node.js, Python, and Xamarin SDKs.

**What is an Azure Cosmos DB account?**

An Azure Cosmos DB account is an Azure resource that acts as an organizational entity for your databas­es. It connects your usage to your Azure subscription for billing purposes.

Each Azure Cosmos DB account is associated with one of the several data models Azure Cosmos DB supports, and you can create as many accounts as you need.

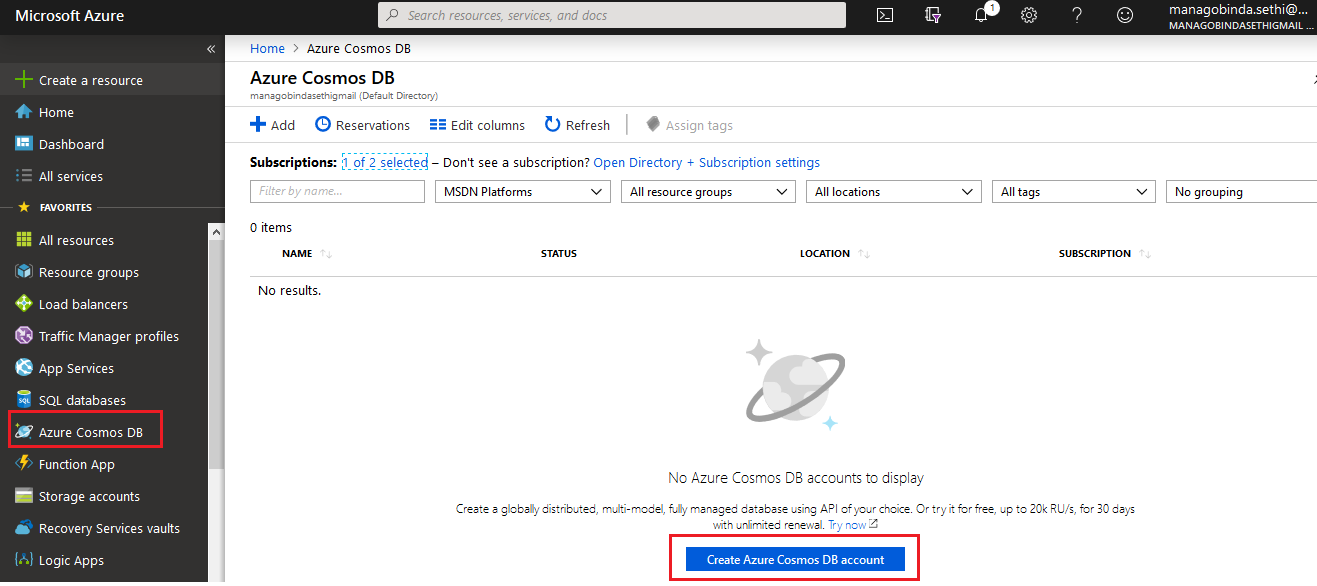
SQL API is the preferred data model if you are creating a new application. If you're working with graphs or tables, or migrating your MongoDB or Cassandra data to Azure, create additional accounts and select relevant data models.

When creating an account, choose an ID that is meaningful to you; it is how you identify your account. Further, create the account in the Azure region that's closest to your users to minimize latency between the datacenter and your users.

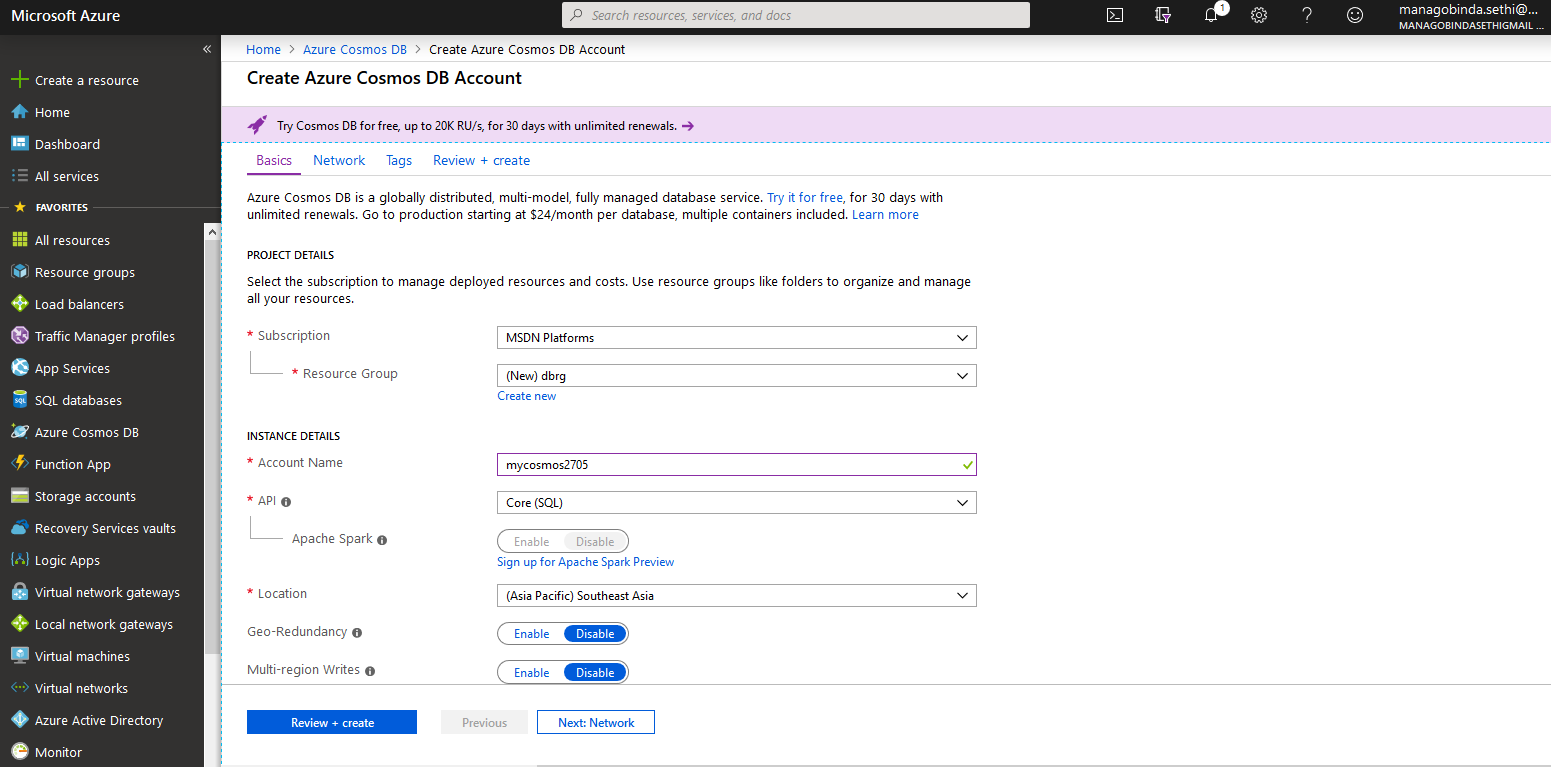
You can optionally set up virtual networks and geo-redundancy during account creation, but this can also be done later. In this module we will not enable those settings.

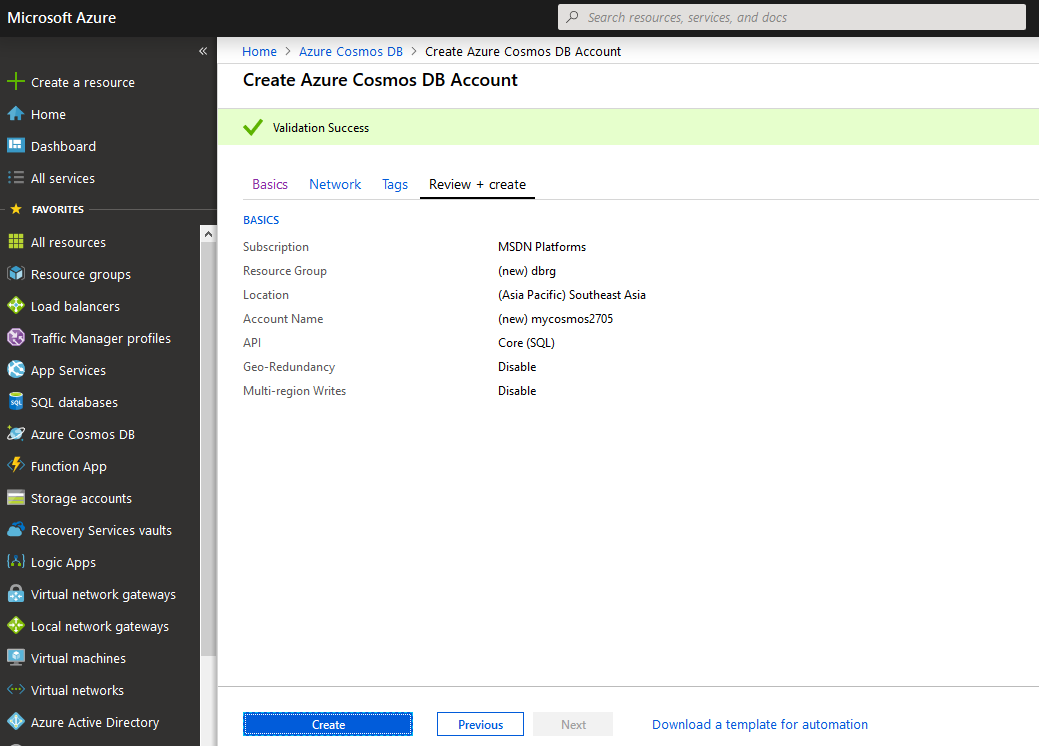
**Create a Cosmos DB database and collection**

1. On the **Azure Cosmos DB** blade, click the **Create** button.



1. On the new **Azure Cosmos DB** blade, perform the following tasks: Leave the **Subscription** drop-down list entry set to its default value.
   1. Resource group: ensure that the **Create new** option is selected and then, in the text box, type **dbrg**.
   2. In the **Account Name** text box, type a globally unique value.
   3. In the **API** drop-down list, select the **Core (SQL)** option.
   4. In the **Location** drop-down list, select the Azure region in which you want to deploy resources in this lab.
   5. Leave all remaining settings with their default values.
   6. Click the **Create** button.





1. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a variable which value designates the name of the resource group that contains the Azure Cosmos DB account you deployed earlier in this task:

RESOURCE\_GROUP='dbrg'

1. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a variable which value designates the name of the CosmosDB account you created earlier in this task:

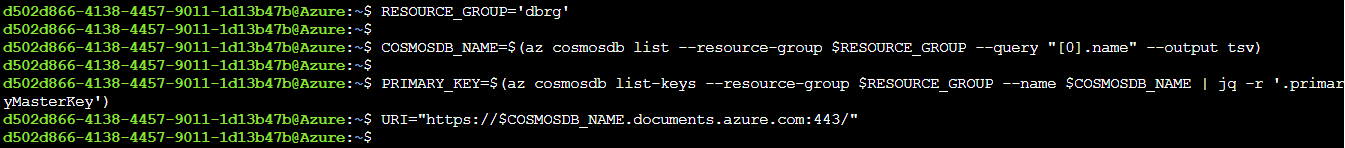
COSMOSDB\_NAME=$(azcosmosdb list --resource-group $RESOURCE\_GROUP --query "[0].name" --output tsv)

1. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a variable which value designates the primary key of the CosmosDB account you created earlier in this task:

PRIMARY\_KEY=$(azcosmosdb list-keys --resource-group $RESOURCE\_GROUP --name $COSMOSDB\_NAME | jq -r '.primaryMasterKey')

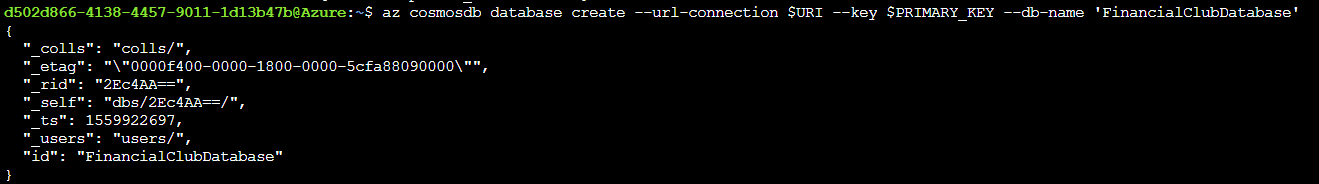
1. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a variable which value designates the URI of the CosmosDB account you created earlier in this task:

URI=<https://$COSMOSDB_NAME.documents.azure.com:443/>



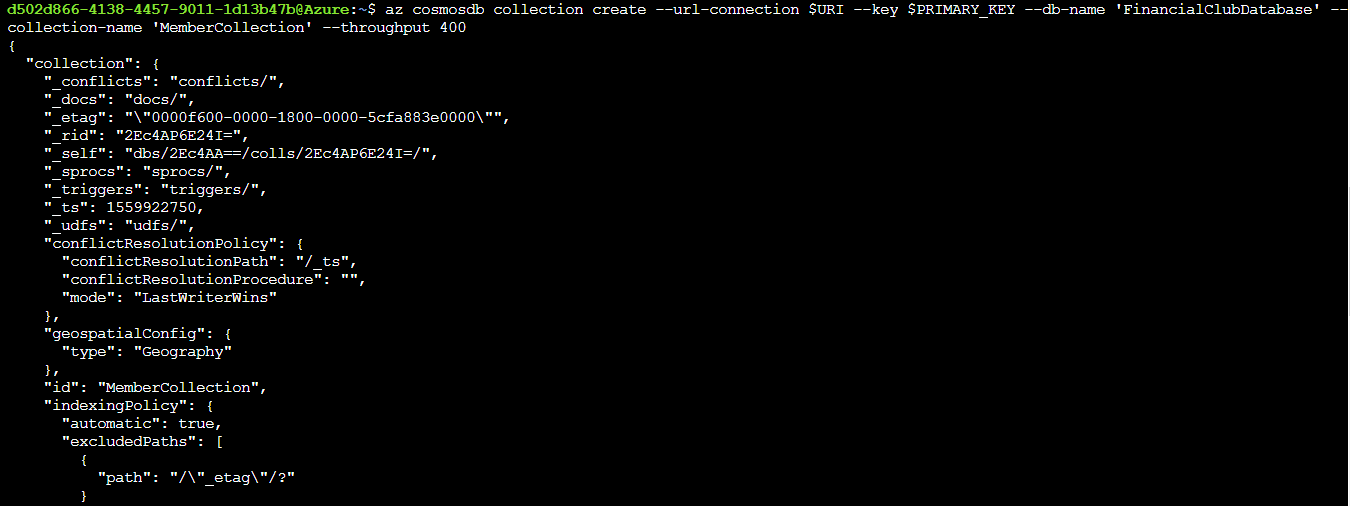
1. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a new CosmosDB database named **FinancialClubDatabase**:

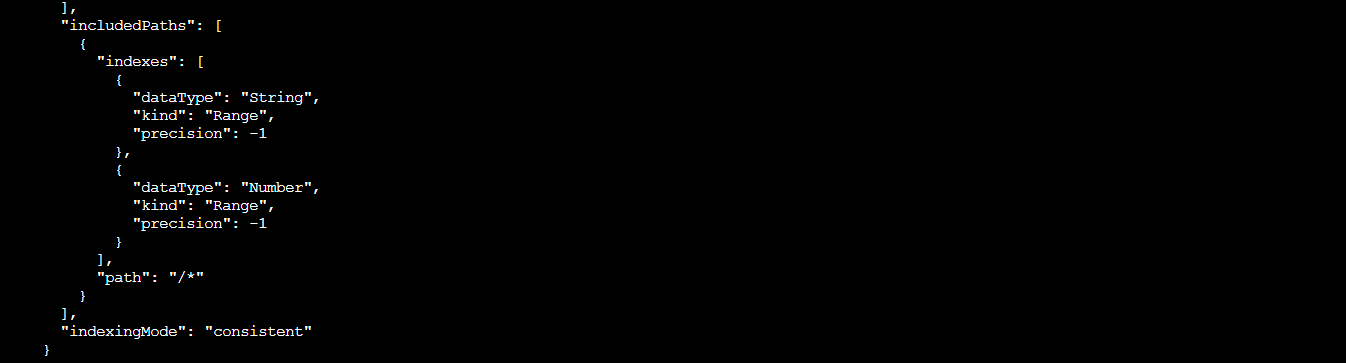
azcosmosdb database create --url-connection $URI --key $PRIMARY\_KEY --db-name 'FinancialClubDatabase'

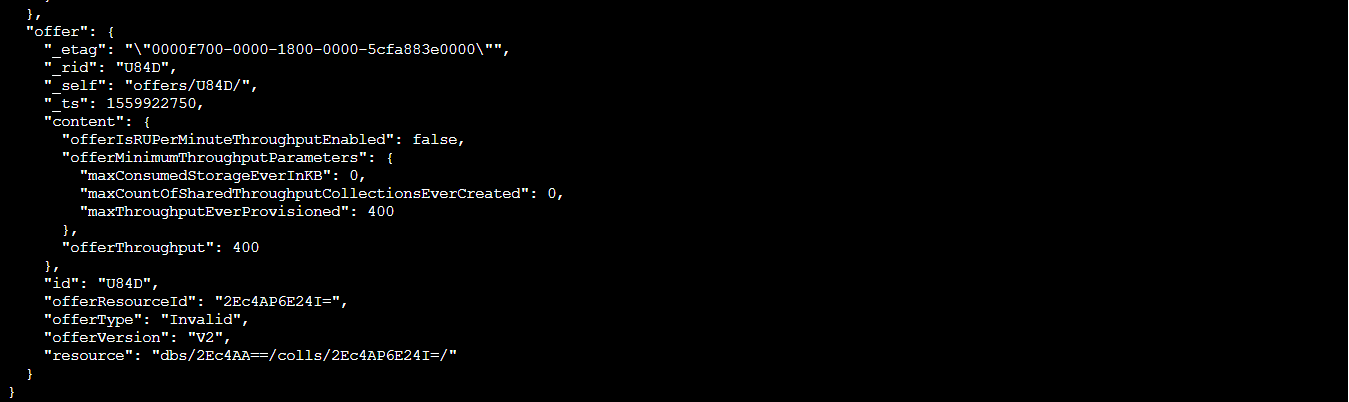


1. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to create a fixed collection named **MemberCollection**in the newly created database:

azcosmosdb collection create --url-connection $URI --key $PRIMARY\_KEY --db-name 'FinancialClubDatabase' --collection-name 'MemberCollection' --throughput 400







1. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to display the value of the PRIMARY\_KEY variable:

echo $PRIMARY\_KEY



1. At the **Cloud Shell** command prompt, type in the following command and press **Enter** to display the value of the URI variable:

echo $URI

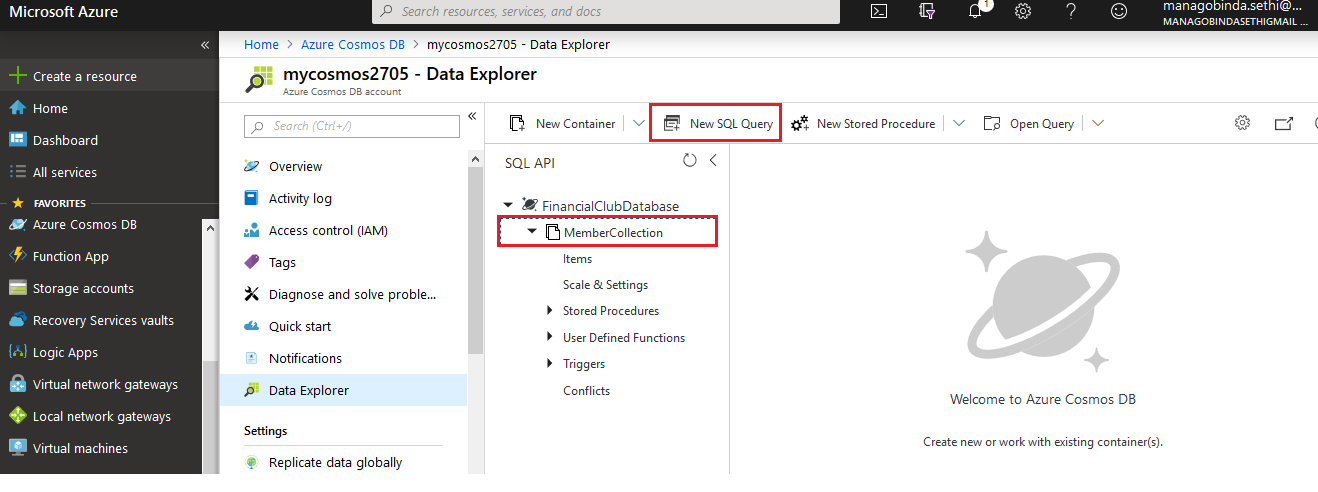


**Create and query documents in Cosmos DB**

1. On the left side of the Azure Cosmos DB account blade, click **Data Explorer**.

2. In the **Data Explorer** pane, click the **MemberCollection**child node of the **FinancialClubDatabase**node.

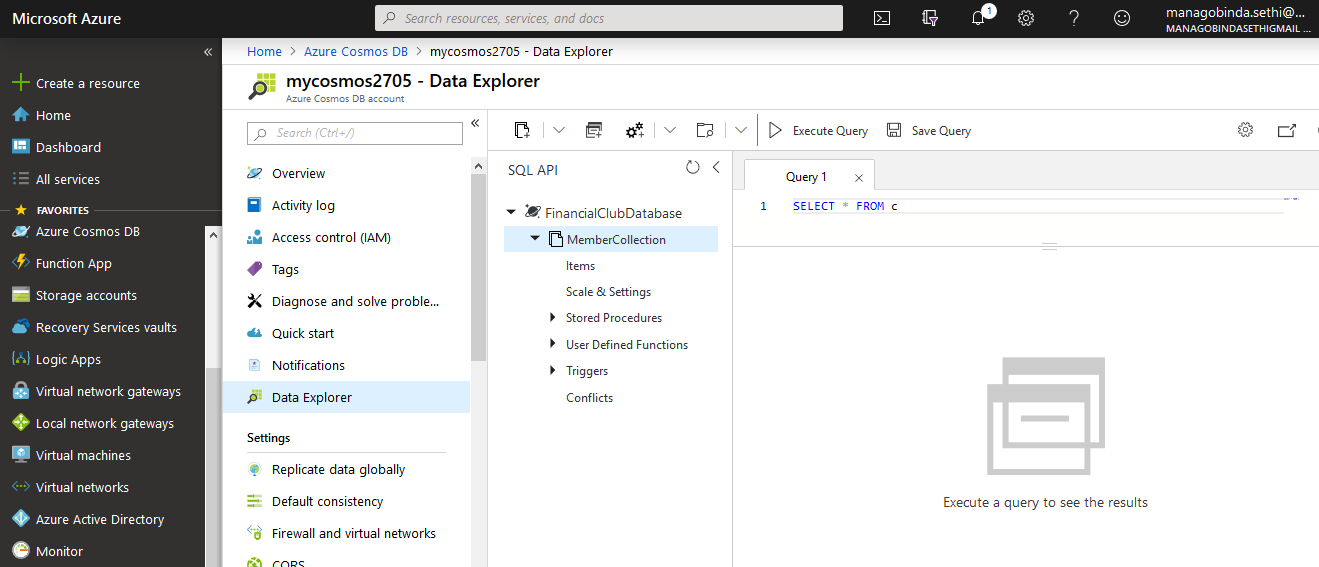
3. Click the **New SQL Query** button at the top of the **Data Explorer** pane.



4. In the **Query 1** tab that opened, view the default query:

SELECT \* FROM c

5.Click the **Execute Query** button at the top of the query editor.



6. In the left pane of the Data Explorer, expand the **MemberCollection**node.

7. Click the **Items** child node within the **MemberCollection**node.

8. In the new **Items**tab that opened, click the **New Item** button at the top of the tab.

9. In the **Items**tab, replace the existing document with the following document:

{

"firstName": "Pennington",

"lastName": "Oneal",

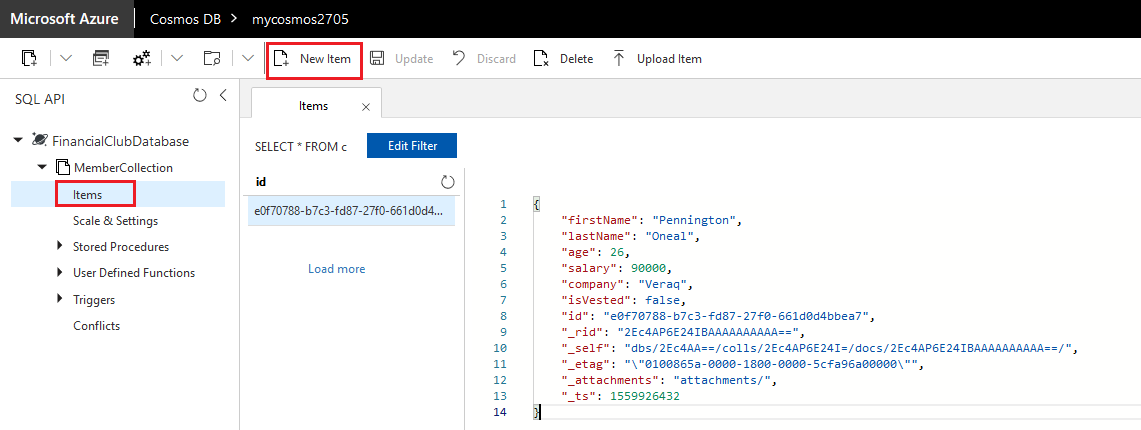
"age": 26,

"salary": 90000.00,

"company": "Veraq",

"isVested": false

}



10. Click the **Save** button at the top of the **Items**tab.

11. In the **Items**tab, click the **New Items** button at the top of the tab.

1. In the **Items**tab, replace the existing document with the following document:

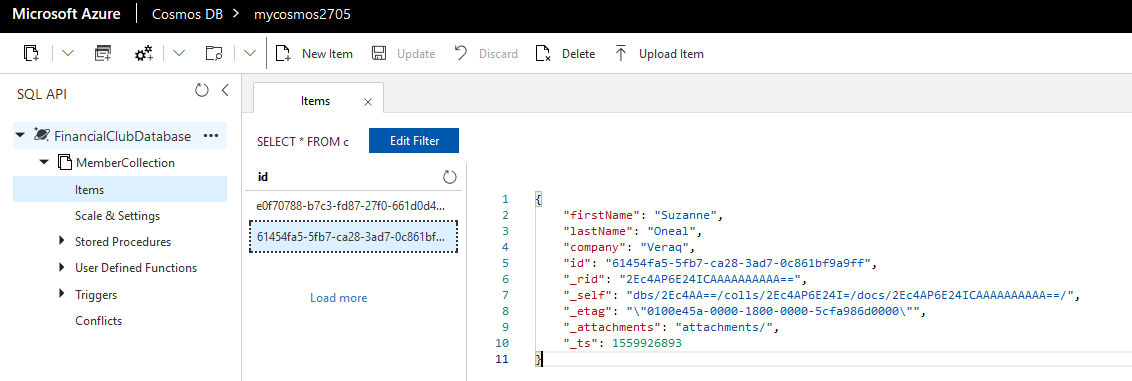
{

"firstName": "Suzanne",

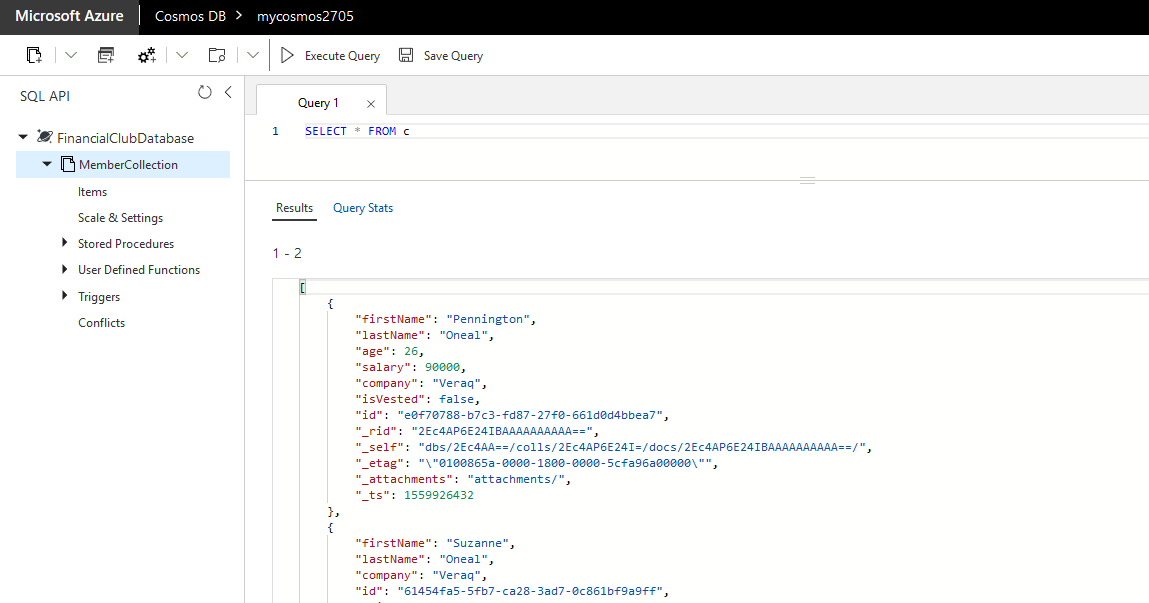
"lastName": "Oneal",

"company": "Veraq"

}



1. Switch back to the **Query 1** tab, re-run the default query SELECT \* FROM c by clicking the **Execute Query** button at the top of the query editor, and review the results.



1. In the query editor, replace the default query with the following query:

SELECT

c.id,

c.firstName,

c.lastName,

c.isVested,

c.company

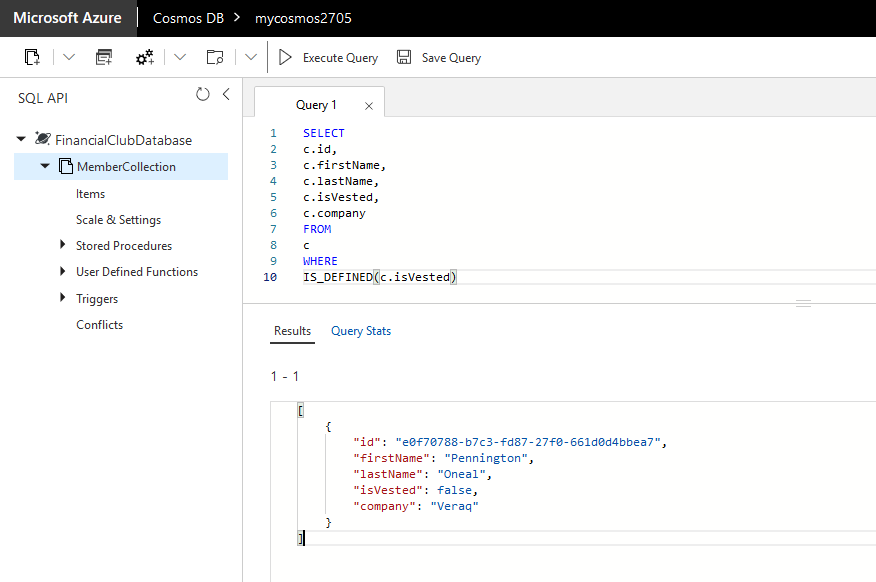
FROM

c

WHERE

IS\_DEFINED(c.isVested)

1. Click the **Execute Query** button at the top of the query editor and review the results.



1. In the query editor, replace the existing query with the following query:

SELECT

c.id,

c.firstName,

c.lastName,

c.age

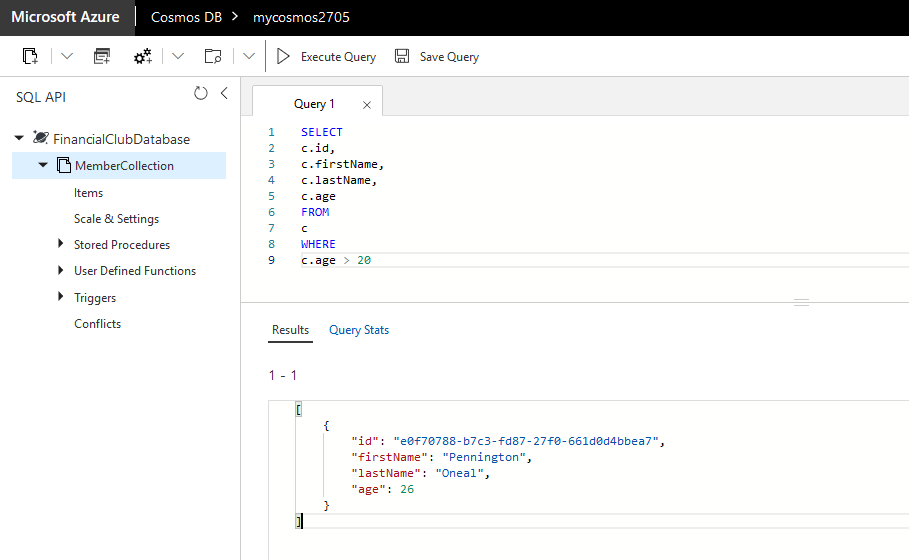
FROM

c

WHERE

c.age> 20

1. Click the **Execute Query** button at the top of the query editor and review the results.



1. In the query editor, replace the existing query with the following query:

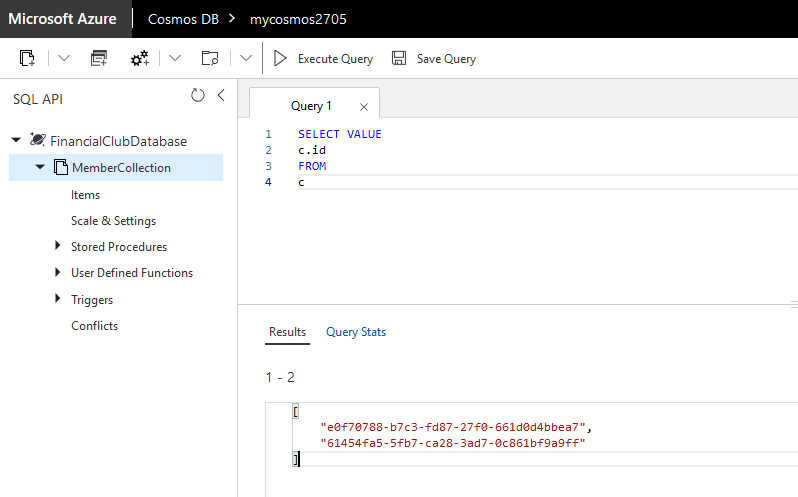
SELECT VALUE

c.id

FROM

c

1. Click the **Execute Query** button at the top of the query editor and review the results.



1. In the query editor, replace the existing query with the following query:

SELECT VALUE {

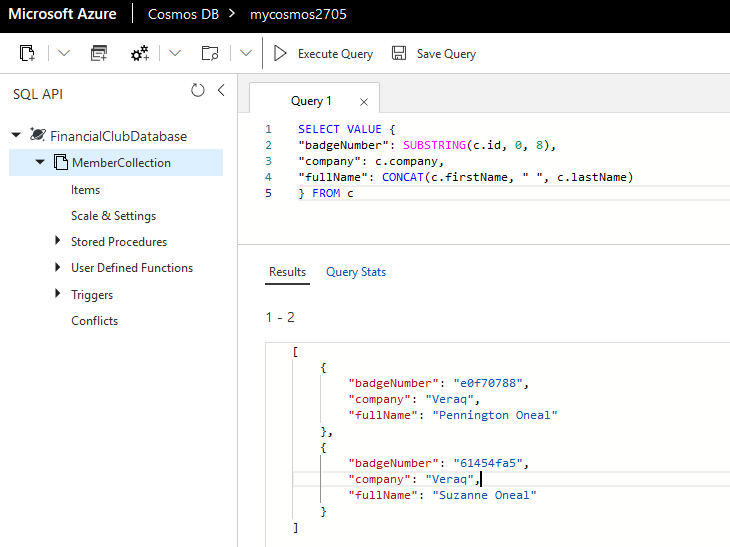
"badgeNumber": SUBSTRING(c.id, 0, 8),

"company": c.company,

"fullName": CONCAT(c.firstName, " ", c.lastName)

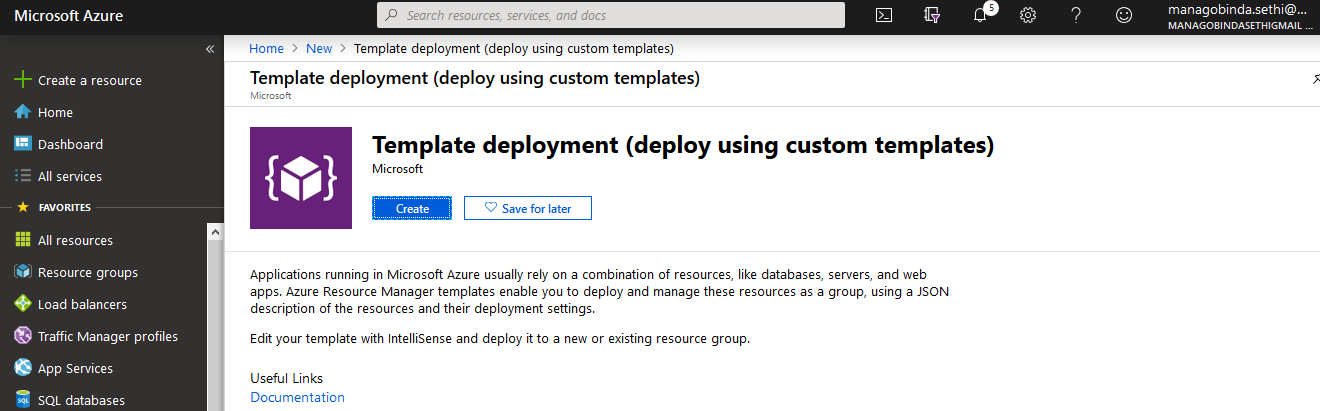
} FROM c

1. Click the **Execute Query** button at the top of the query editor and review the results.

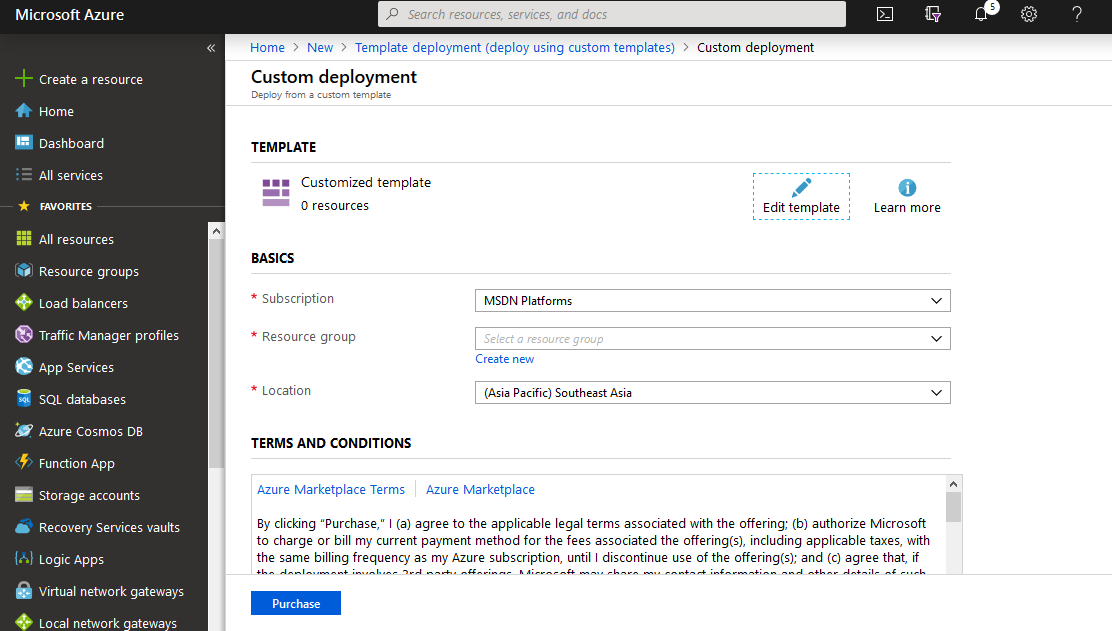


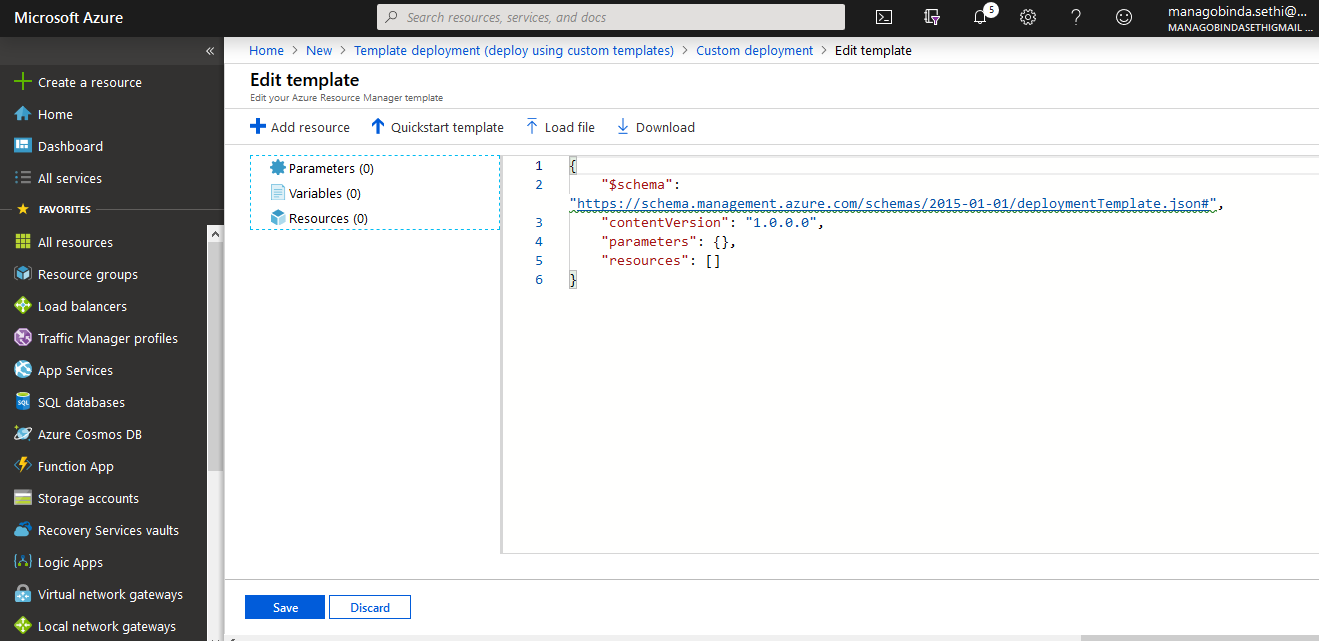
**Deploy Application using Cosmos DB**

1. In the upper left corner of the Azure portal, click **Create a resource**.
2. At the top of the **New** blade, in the **Search the Marketplace** text box, type **Template Deployment** and press **Enter**.
3. On the **Template deployment** blade, click the **Create** button.



1. On the **Edit template** blade, click the **Load file** link.





1. Create an json template as below and name it as api.json and save it on a local path of your laptop.

|  |
| --- |
| { |
|  |

|  |
| --- |
| "$schema": "http://schemas.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#", |
|  |

|  |
| --- |
| "contentVersion": "1.0.0.0", |
|  |

|  |
| --- |
| "variables": { |
|  |

|  |
| --- |
| "appService": { |
|  |

|  |
| --- |
| "apiApp": { |
|  |

|  |
| --- |
| "name": "[concat('api', uniqueString(resourceGroup().id))]" |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| "plan": { |
|  |

|  |
| --- |
| "name": "[concat('pln', uniqueString(resourceGroup().id))]", |
|  |

|  |
| --- |
| "sku": { |
|  |

|  |
| --- |
| "name": "Shared", |
|  |

|  |
| --- |
| "code": "D1", |
|  |

|  |
| --- |
| "workerSize": 0 |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| "repository": { |
|  |

|  |
| --- |
| "url": "https://github.com/azure-labs/cosmos-api", |
|  |

|  |
| --- |
| "branch": "master" |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| "resources": [ |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| "apiVersion": "2016-09-01", |
|  |

|  |
| --- |
| "type": "Microsoft.Web/serverfarms", |
|  |

|  |
| --- |
| "name": "[variables('appService').plan.name]", |
|  |

|  |
| --- |
| "location": "[resourceGroup().location]", |
|  |

|  |
| --- |
| "properties": { |
|  |

|  |
| --- |
| "name": "[variables('appService').plan.name]", |
|  |

|  |
| --- |
| "workerSizeId": "[variables('appService').plan.sku.workerSize]", |
|  |

|  |
| --- |
| "numberOfWorkers": "1" |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| "sku": { |
|  |

|  |
| --- |
| "tier": "[variables('appService').plan.sku.name]", |
|  |

|  |
| --- |
| "name": "[variables('appService').plan.sku.code]" |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| "apiVersion": "2016-03-01", |
|  |

|  |
| --- |
| "type": "Microsoft.Web/sites", |
|  |

|  |
| --- |
| "name": "[variables('appService').apiApp.name]", |
|  |

|  |
| --- |
| "location": "[resourceGroup().location]", |
|  |

|  |
| --- |
| "kind": "api", |
|  |

|  |
| --- |
| "dependsOn": [ |
|  |

|  |
| --- |
| "[resourceId('Microsoft.Web/serverfarms', variables('appService').plan.name)]" |
|  |

|  |
| --- |
| ], |
|  |

|  |
| --- |
| "properties": { |
|  |

|  |
| --- |
| "name": "[variables('appService').apiApp.name]", |
|  |

|  |
| --- |
| "serverFarmId": "[resourceId('Microsoft.Web/serverfarms', variables('appService').plan.name)]", |
|  |

|  |
| --- |
| "siteConfig": { |
|  |

|  |
| --- |
| "cors": { |
|  |

|  |
| --- |
| "allowedOrigins": [ |
|  |

|  |
| --- |
| "\*" |
|  |

|  |
| --- |
| ] |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| "appSettings": [ |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| "name": "CosmosDB:DatabaseId", |
|  |

|  |
| --- |
| "value": "Employee" |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| "name": "CosmosDB:CollectionId", |
|  |

|  |
| --- |
| "value": "myDataBase" |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| "name": "CosmosDB:StoredProcedureId", |
|  |

|  |
| --- |
| "value": "genDocuments" |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| "name": "CosmosDB:EndpointUrl", |
|  |

|  |
| --- |
| "value": "" |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| "name": "CosmosDB:AuthorizationKey", |
|  |

|  |
| --- |
| "value": "" |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| "name": "Search:IndexId", |
|  |

|  |
| --- |
| "value": "memberindex" |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| "name": "Search:AccountName", |
|  |

|  |
| --- |
| "value": "" |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| "name": "Search:QueryKey", |
|  |

|  |
| --- |
| "value": "" |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| ] |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| }, |
|  |

|  |
| --- |
| "resources": [ |
|  |

|  |
| --- |
| { |
|  |

|  |
| --- |
| "apiVersion": "2015-08-01", |
|  |

|  |
| --- |
| "name": "web", |
|  |

|  |
| --- |
| "type": "sourcecontrols", |
|  |

|  |
| --- |
| "dependsOn": [ |
|  |

|  |
| --- |
| "[resourceId('Microsoft.Web/Sites', variables('appService').apiApp.name)]" |
|  |

|  |
| --- |
| ], |
|  |

|  |
| --- |
| "properties": { |
|  |

|  |
| --- |
| "RepoUrl": "[variables('repository').url]", |
|  |

|  |
| --- |
| "branch": "[variables('repository').branch]", |
|  |

|  |
| --- |
| "IsManualIntegration": true |
|  |

|  |
| --- |
| } |
|  |

|  |
| --- |
| } |
|  |

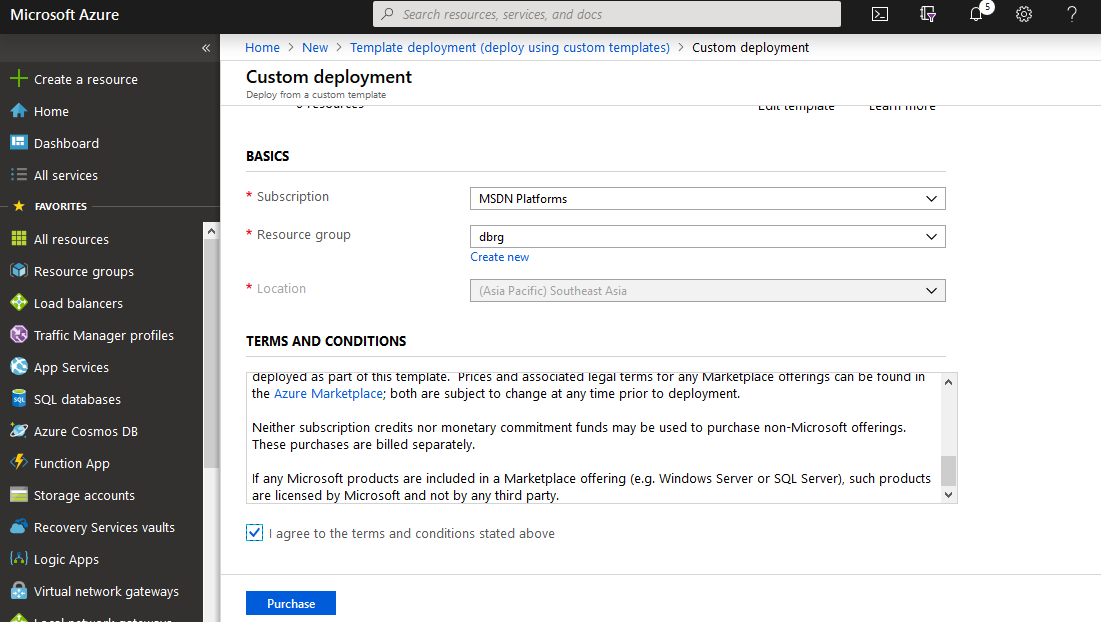
|  |
| --- |
| ] |
|  |

|  |
| --- |
| } |
|  |

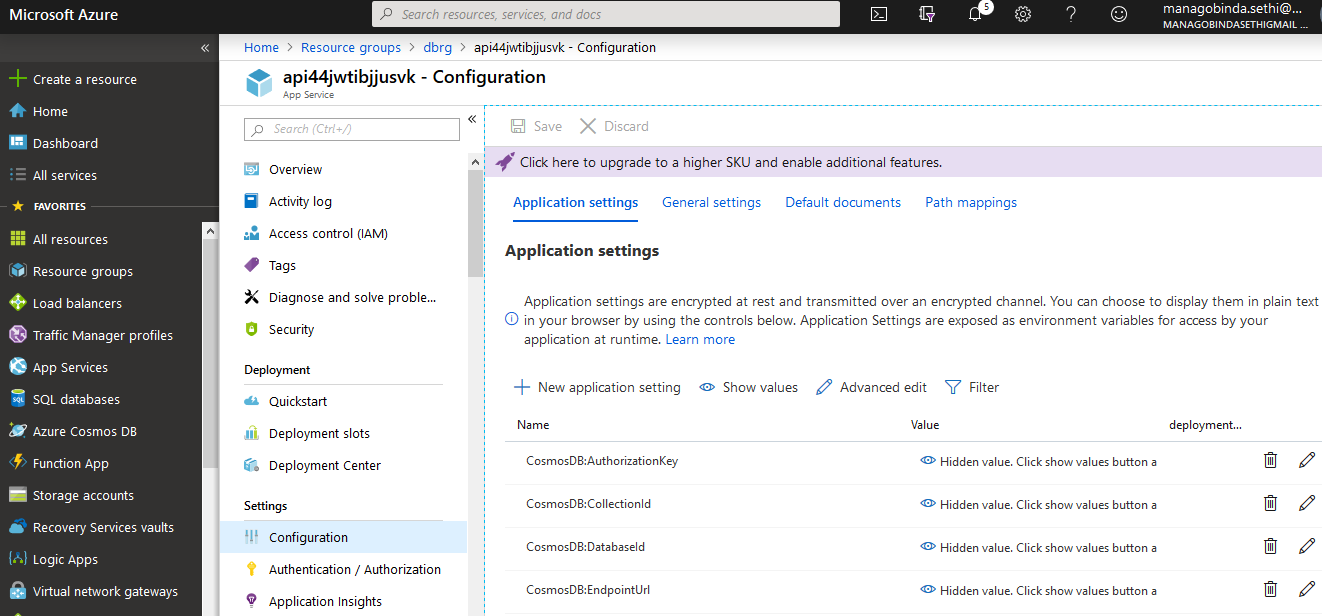
|  |
| --- |
| ] |
|  |

}

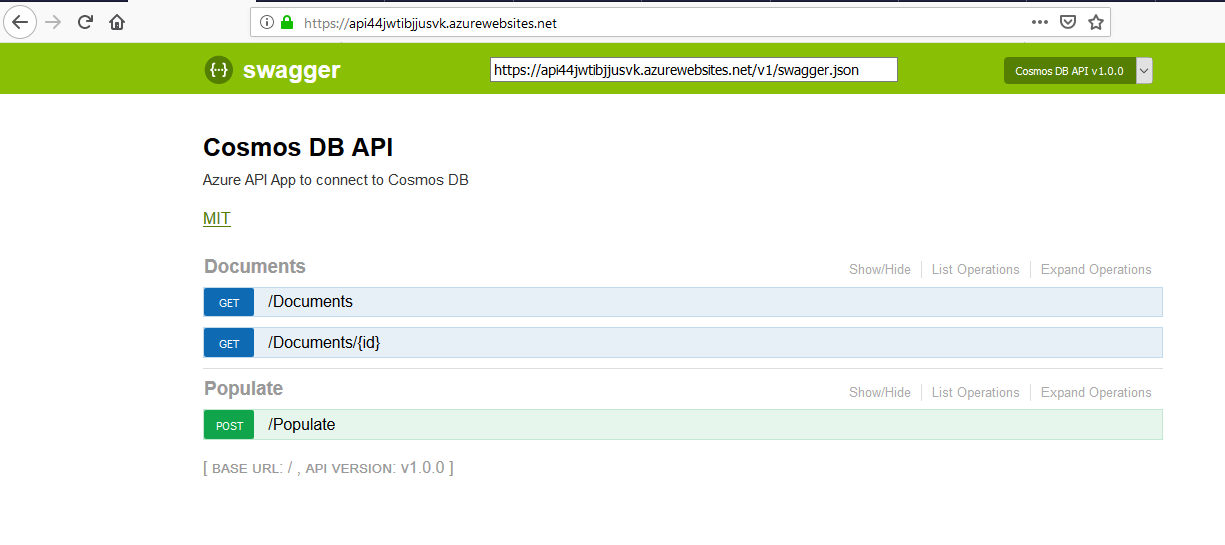
1. Select the **api.json**file. And save the template.
   1. 7. Back on the **Custom deployment** blade, perform the following tasks: Leave the **Subscription** drop-down list entry set to its default value.
   * In the **Resource group** section, select the **Use existing** option and then, in the drop-down list, select **dbrg**
   * In the **Terms and Conditions** section, click the **I agree to the terms and conditions stated above** checkbox.
   * Click the **Purchase** button.



1. Once the app service is created, please click on it and go to settings, configuration section.



1. Under application settings tab, take out the information ofecho $PRIMARY\_KEY from previous section and edit the **CosmosDB:AuthorizationKey**
2. Take out the information ofecho $URI from previous section and edit the **CosmosDB:EndpointUrl**and save it.
3. On the left-side of the API app blade, click **Overview**.
4. Click the **Restart** button at the top of the blade and, when prompted to confirm, click **Yes**.
5. Click the **Browse** button at the top of the blade. This will open a new browser tab displaying the **Swagger UI** homepage



1. On the **Swagger UI** homepage, click **GET/Documents**.
2. Click the **Try it out!** button.
3. Review the results of the request.
4. Back on the **Swagger UI** homepage, click **POST/Populate**.
5. In the **Parameters** section, in the **Value** field for the **options** parameter, paste in the following JSON content:

{

"quantity": 50

}

1. In the **Response Messages** section, click the **Try it out!** button.
2. Review the results of the request.
3. Back on the **Swagger UI** homepage, click **GET/Documents**.
4. Locate the **Response Messages** section. Click the **Try it out!** button.
5. Review the results of the request.