Managing Relational and Non-Relational Data | 2022

NoSQL Lab – Challenge 1

Last Review: 30th April 2022

Cosmos DB Table API Challenge Overview

In this lab you will be introduced to work with Cosmos DB Table API through Azure portal and using a Python SDK. You will learn to create and manage data, load, insert data and query data to Cosmos DB Table API.

Azure Cosmos DB provides the Table API for applications that are written for Azure Table storage and that need premium capabilities like:

- Turnkey global distribution.
- Dedicated throughput worldwide (when using provisioned throughput).
- Single-digit millisecond latencies at the 99th percentile.
- Guaranteed high availability.
- Automatic secondary indexing.

Azure Tables SDKs are available for .NET, Java, Python, Node.js, and Go. These SDKs can be used to target either Table Storage or Cosmos DB Tables. Applications written for Azure Table storage using the Azure Tables SDKs can be migrated to the Azure Cosmos DB Table API with no code changes to take advantage of premium capabilities.

Lab Requirements & Pre-requisites:

To complete the labs, you will need the following:

- An Azure subscription (for students or free using your personal email)
- Install Anaconda (download here)
- Windows machine (please use Boot Camp Assistant if you are using a MAC machine OR adapt the lab below to be used on your MAC machine OR create an Azure Windows Virtual Machine with Windows)

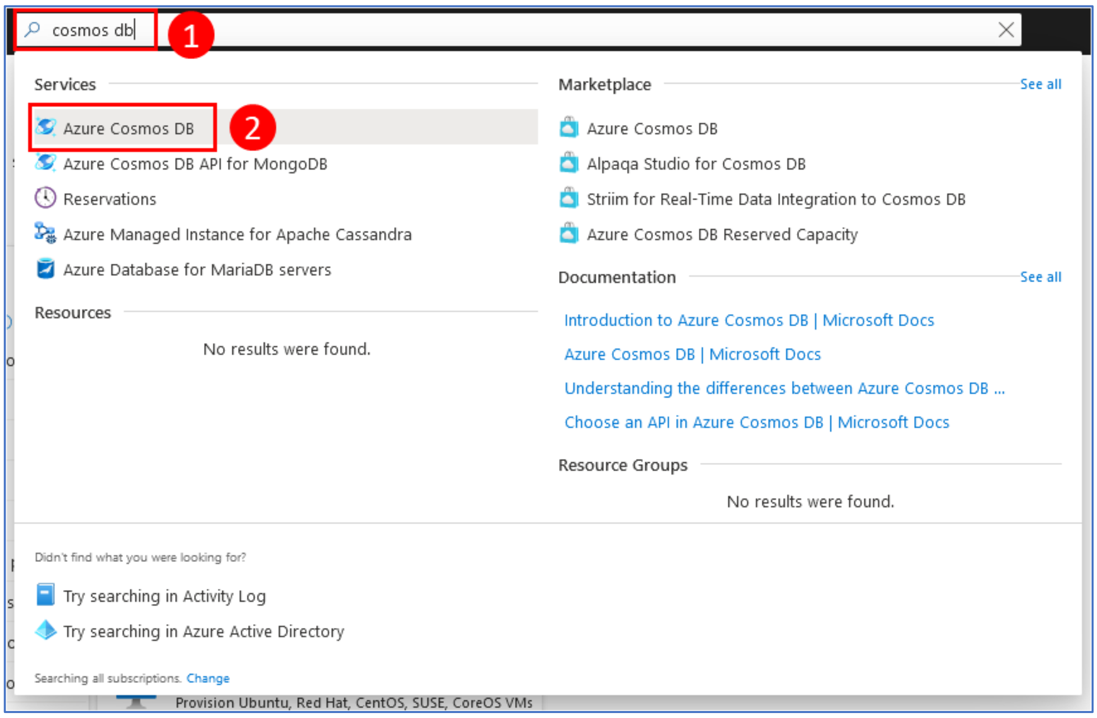
1. Create an Azure Cosmos DB account

You first need to create a Cosmos DB Tables API account that will contain the table(s) used in your application. This can be done using the Azure portal.

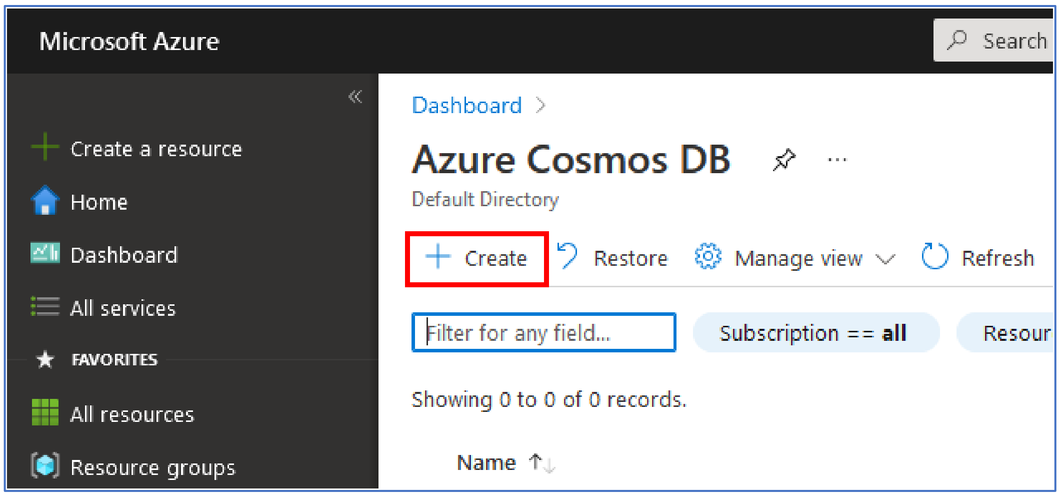
Log in to the Azure portal and follow these steps to create a Cosmos DB account.

1. In the Azure portal follow these instructions:

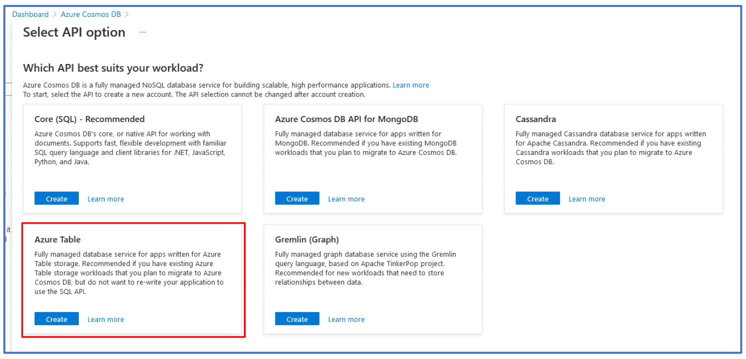
1. In the search bar at the top of the Azure portal, enter "cosmos db".
2. On the menu that appears below the search bar, under **Services**, select the item labeled **Azure Cosmos DB**.


cosmos db
1
Services
Azure Cosmos DB
2
Azure Cosmos DB API for MongoDB
Reservations
Azure Managed Instance for Apache Cassandra
Azure Database for MariaDB servers
Resources
No results were found.
Marketplace
See all
Azure Cosmos DB
Alpaqa Studio for Cosmos DB
Striim for Real-Time Data Integration to Cosmos DB
Azure Cosmos DB Reserved Capacity
Documentation
See all
Introduction to Azure Cosmos DB | Microsoft Docs
Azure Cosmos DB | Microsoft Docs
Understanding the differences between Azure Cosmos DB ...
Choose an API in Azure Cosmos DB | Microsoft Docs
Resource Groups
No results were found.
Didn't find what you were looking for?
Try searching in Activity Log
Try searching in Azure Active Directory
Searching all subscriptions. Change
Provision Ubuntu, Red Hat, CentOS, SUSE, CoreOS VMs

2. On the **Azure Cosmos DB** page select **+Create**.



3. On the **Select API** option page choose the Azure Table option.



Dashboard > Azure Cosmos DB >

Select API option ···

Which API best suits your workload?

Azure Cosmos DB is a fully managed NoSQL database service for building scalable, high performance applications. Learn more
To start, select the API to create a new account. The API selection cannot be changed after account creation.

Core (SQL) - Recommended
Azure Cosmos DB's core, or native API for working with documents. Supports fast, flexible development with familiar SQL query language and client libraries for .NET, JavaScript, Python, and Java.

Azure Cosmos DB API for MongoDB
Fully managed database service for apps written for MongoDB. Recommended if you have existing MongoDB workloads that you plan to migrate to Azure Cosmos DB.

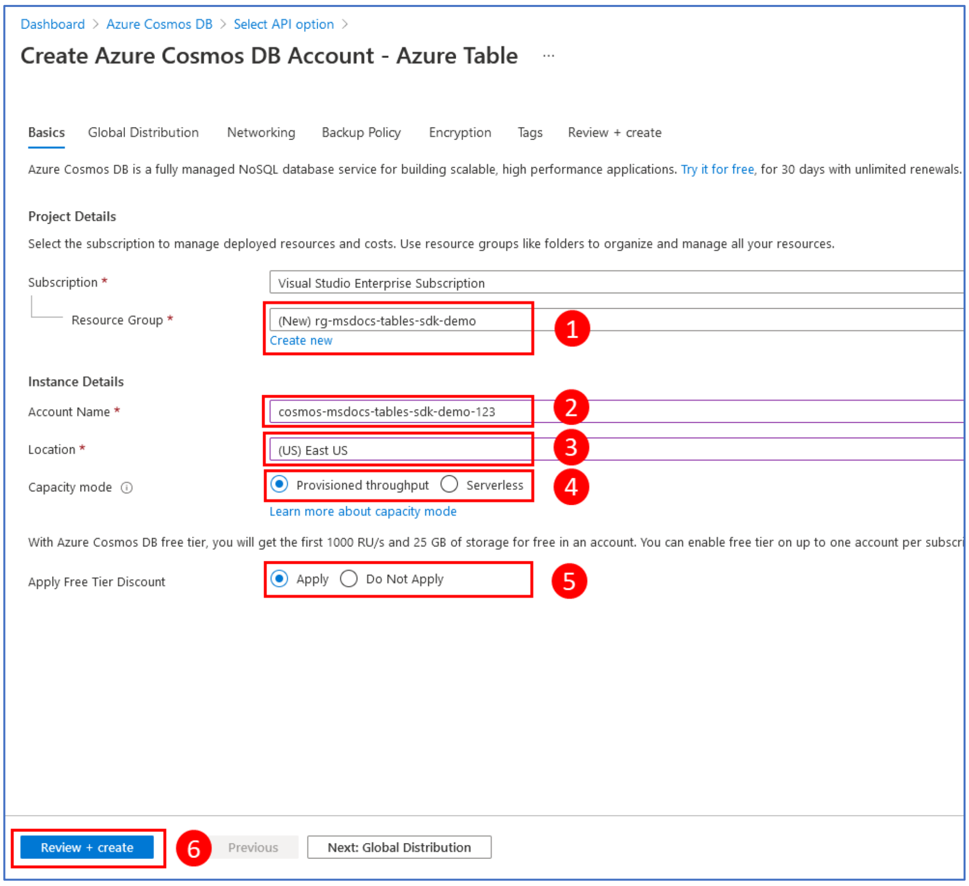
Cassandra
Fully managed Cassandra database service for apps written for Apache Cassandra. Recommended if you have existing Cassandra workloads that you plan to migrate to Azure Cosmos DB.

Azure Table
Fully managed database service for apps written for Azure Table storage. Recommended if you have existing Azure Table storage workloads that you plan to migrate to Azure Cosmos DB, but do not want to re-write your application to use the SQL API.

Gremlin (Graph)
Fully managed graph database service using the Gremlin query language, based on Apache TinkerPop project. Recommended for new workloads that need to store relationships between data.

4. On the **Create Azure Cosmos DB Account - Azure Table** page, fill out the form as follows.

1. Create a new resource group for the Azure Cosmos DB name by selecting the **Create new** link under **Resource group**.
2. Give your Azure Cosmos DB account a name. Azure Cosmos DB account names must be between 3 and 44 characters in length and may contain only lowercase letters, numbers or the hyphen (-) character.
3. Select the **region** for your Azure Cosmos DB account.
4. Select **Standard** performance.
5. Select **Provisioned throughput** for this example under *Capacity mode*.
6. Select **Apply** under *Apply Free Tier Discount* for this example.
7. Select the **Review + create** button at the bottom of the screen and then select "Create" on the summary screen to create your Azure Cosmos DB account. This process may take several minutes.



Dashboard > Azure Cosmos DB > Select API option >

Create Azure Cosmos DB Account - Azure Table ···

Azure Cosmos DB is a fully managed NoSQL database service for building scalable, high performance applications. Try it for free, for 30 days with unlimited renewals.

Project Details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *: Visual Studio Enterprise Subscription

Resource Group *: (New) rg-msdocs-tables-sdk-demo
Create new

Instance Details

Account Name *: cosmos-msdocs-tables-sdk-demo-123

Location *: (US) East US

Capacity mode: [✓ Provisioned throughput] Serverless
Learn more about capacity mode

With Azure Cosmos DB free tier, you will get the first 1000 RU/s and 25 GB of storage for free in an account. You can enable free tier on up to one account per subscri

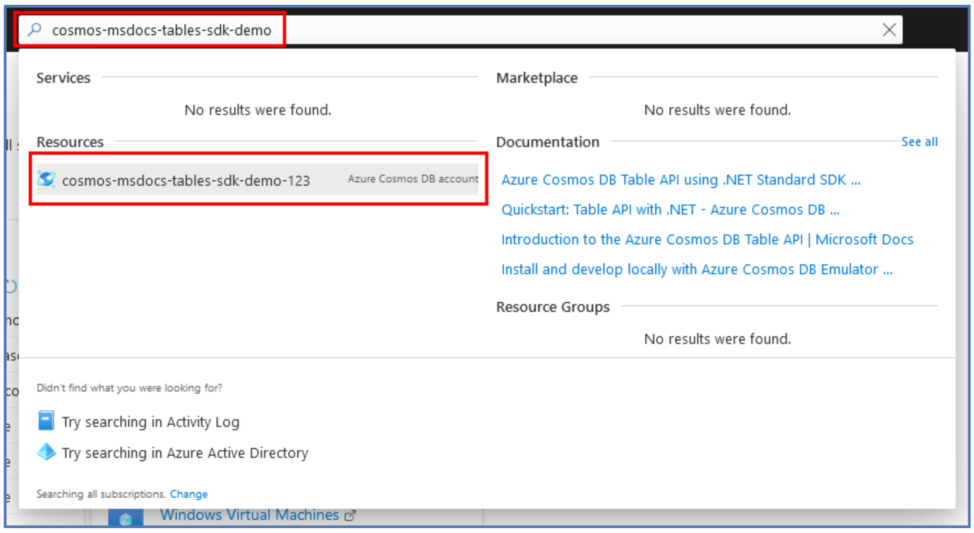
Apply Free Tier Discount: [✓ Apply] Do Not Apply

2. Create a table

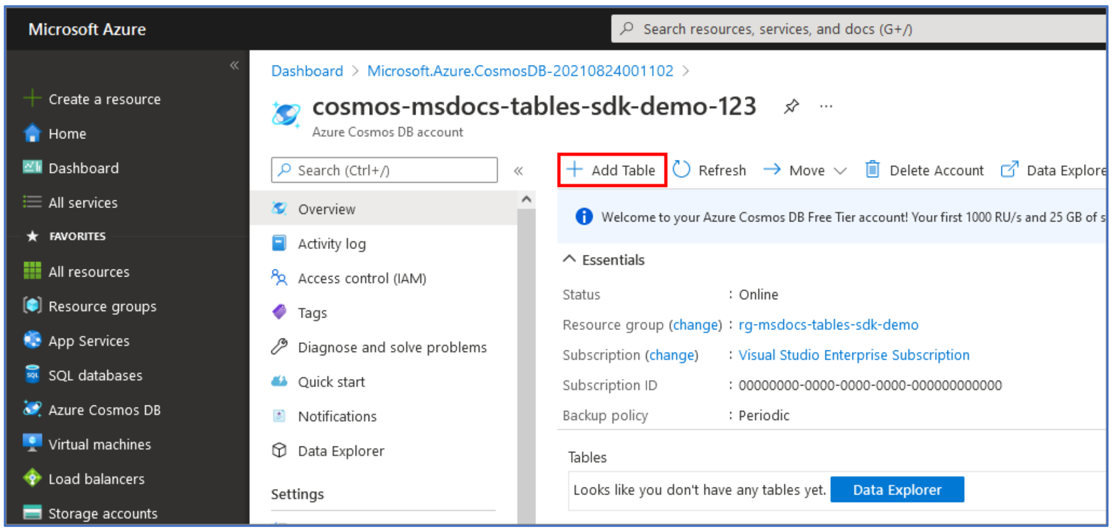
Next, you need to create a table within your Cosmos DB account for your application to use. Unlike a traditional database, you only need to specify the name of the table, not the properties (columns) in the table. As data is loaded into your table, the properties (columns) will be automatically created as needed.

In the Azure portal, complete the following steps to create a table inside your Cosmos DB account.

1. In the Azure portal, navigate to the overview page for the Azure Cosmos DB account. You can navigate to the overview page for your Cosmos DB account by typing the name (*cosmos-msdocs-tables-sdk-demo-XYZ* or having a different name) of your Cosmos DB account in the top search bar and looking under the resources heading. Select the name of your Azure Cosmos DB account to go to the overview page.

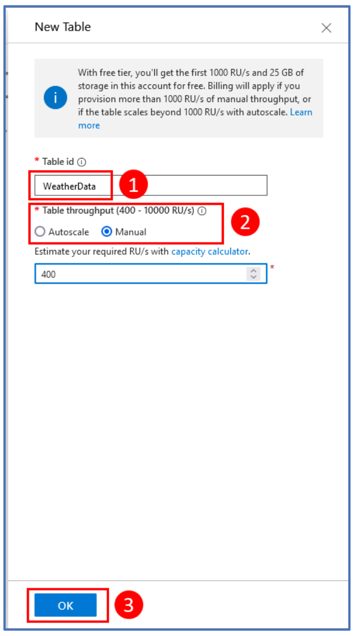

cosmos-msdocs-tables-sdk-demo
Services
No results were found.
Marketplace
No results were found.
Resources
cosmos-msdocs-tables-sdk-demo-123
Azure Cosmos DB account
Documentation
See all
Azure Cosmos DB Table API using .NET Standard SDK ...
Quickstart: Table API with .NET - Azure Cosmos DB ...
Introduction to the Azure Cosmos DB Table API | Microsoft Docs
Install and develop locally with Azure Cosmos DB Emulator ...
Resource Groups
No results were found.
Didn't find what you were looking for?
Try searching in Activity Log
Try searching in Azure Active Directory
Searching all subscriptions. Change
Windows Virtual Machines

2. On the overview page, select **+Add Table**. The New Table dialog will slide out from the right side of the page.


Microsoft Azure
Search resources, services, and docs (G+/)
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
App Services
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Dashboard > Microsoft.Azure.CosmosDB-20210824001102 >
cosmos-msdocs-tables-sdk-demo-123
Azure Cosmos DB account
Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Quick start
Notifications
Data Explorer
Settings
Add Table
Refresh
Move
Delete Account
Data Explore
Welcome to your Azure Cosmos DB Free Tier account! Your first 1000 RU/s and 25 GB of s
Essentials
Status : Online
Resource group (change) : rg-msdocs-tables-sdk-demo
Subscription (change) : Visual Studio Enterprise Subscription
Subscription ID : 00000000-0000-0000-0000-000000000000
Backup policy : Periodic
Tables
Looks like you don't have any tables yet.
Data Explorer

3. In the **New Table** dialog, fill out the form as follows.

1. Enter the name **WeatherData** for the Table ID. This is the name of the table.
2. Select **Manual** under *Table throughput (autoscale)* for this example.
3. Use the default value of **400** under your estimated RU/s.
4. Select the **OK** button to create the table.



New Table

With free tier, you'll get the first 1000 RU/s and 25 GB of storage in this account for free. Billing will apply if you provision more than 1000 RU/s of manual throughput, or if the table scales beyond 1000 RU/s with autoscale. Learn more

* Table id: WeatherData

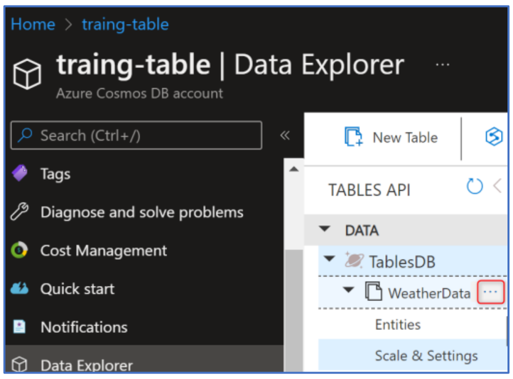
* Table throughput (400 - 10000 RU/s): Autoscale [✓ Manual]

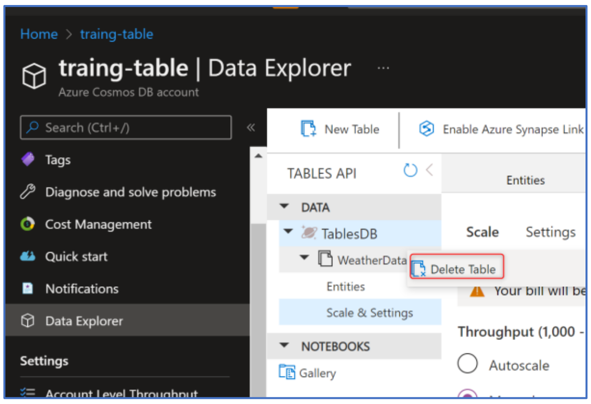
Estimate your required RU/s with capacity calculator.

400

You can delete the table after creating it. This was just an example and we will be using a different table in the next pages.

4. To delete find the "…" on right side of *WeatherData* table, click on the "…", click Delete Table and typed the table id (*WeatherData*) to confirm you want to remove it.

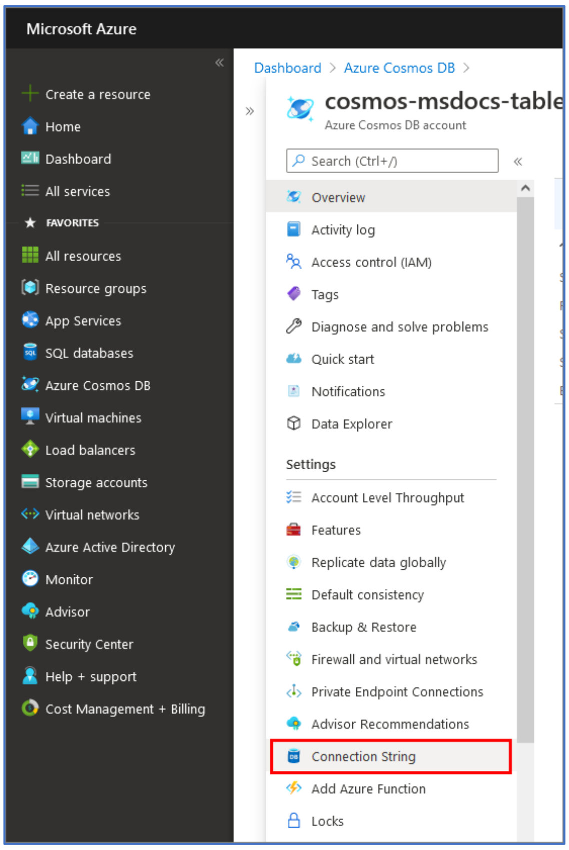




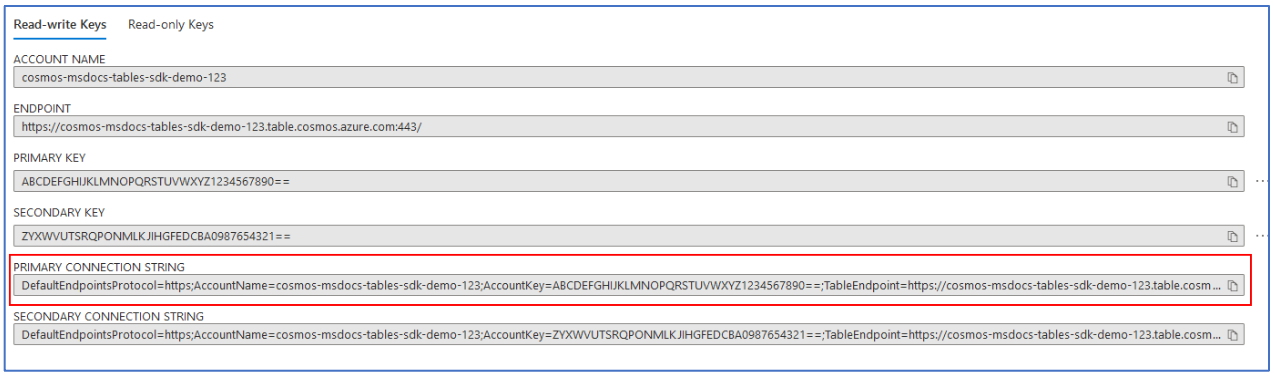
3. Get Cosmos DB connection string

To access your table(s) in Cosmos DB, your app will need the table connection string for the CosmosDB Storage account. The connection string can be retrieved using the Azure portal.

1. On the left hand side of the Azure Cosmos DB account page, locate the menu item named **Connection String** under the *Settings* header and select it. You will be taken to a page where you can retrieve the connection string for the storage account.



2. Copy the *PRIMARY CONNECTION STRING* value to use in your application.


Read-write Keys Read-only Keys

ACCOUNT NAME
cosmos-msdocs-tables-sdk-demo-123

ENDPOINT
https://cosmos-msdocs-tables-sdk-demo-123.table.cosmos.azure.com:443/

PRIMARY KEY
ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890==

SECONDARY KEY
ZYXWVUTSRQPONMLKJIHGFEDCBA0987654321==

PRIMARY CONNECTION STRING
DefaultEndpointsProtocol=https;AccountName=cosmos-msdocs-tables-sdk-demo-123;AccountKey=ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890==;TableEndpoint=https://cosmos-msdocs-tables-sdk-demo-123.table.cosm...

SECONDARY CONNECTION STRING
DefaultEndpointsProtocol=https;AccountName=cosmos-msdocs-tables-sdk-demo-123;AccountKey=ZYXWVUTSRQPONMLKJIHGFEDCBA0987654321==;TableEndpoint=https://cosmos-msdocs-tables-sdk-demo-123.table.cosm...

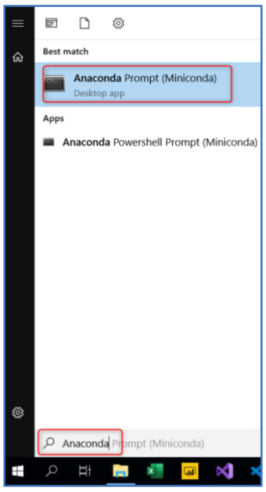
The connection string for your Cosmos DB account is considered an app secret and must be protected like any other app secret or password.

4. Using Anaconda to work with Cosmos DB Table API Python SDK

We are going to work with **Azure Table API** Python SDK to connect to ou Azure Cosmos DB Table API. Since the sample application is written in Python3.6, we are going to use a **Python 3 interpreter**, therefore, we need to create a python 3 environment.

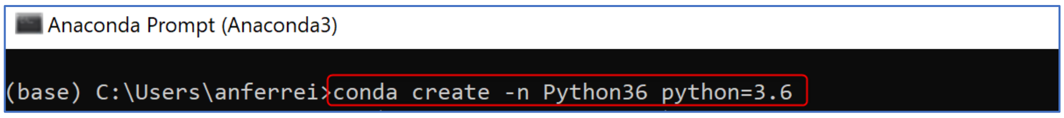
Important Note: Make sure you install Anaconda (download here) on your computer. We are going to use it to create a **Python version 3 environment**.

1. After downloading Anaconda, search for "**Anaconda**" in your computer you should see as result from search the Anaconda Prompt appearing


Best match
Anaconda Prompt (Miniconda)
Desktop app
Apps
Anaconda Powershell Prompt (Miniconda)
Anaconda Prompt (Miniconda)

2. In the Anaconda prompt paste the following command to create a Python 3.6 environment. If asked to proceed type "y".

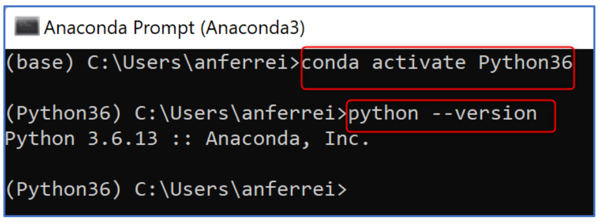
Anaconda prompt:
```
conda create -n Python36 python=3.6
```


Anaconda Prompt (Anaconda3)
```
(base) C:\Users\anferrei>conda create -n Python36 python=3.6
```

3. To activate the environment created execute the following command:

Anaconda prompt:
```
conda activate Python36
```

Important: Any time you need to open a new Anaconda prompt we need to activate this environment!


Anaconda Prompt (Anaconda3)
```
(base) C:\Users\anferrei>conda activate Python36

(Python36) C:\Users\anferrei>python --version
Python 3.6.13 :: Anaconda, Inc.

(Python36) C:\Users\anferrei>
```

4. After activating the *Python36* environment, make sure the version of Python is indeed 3.6, for this run the following command:

Anaconda prompt:
```
python --version
```

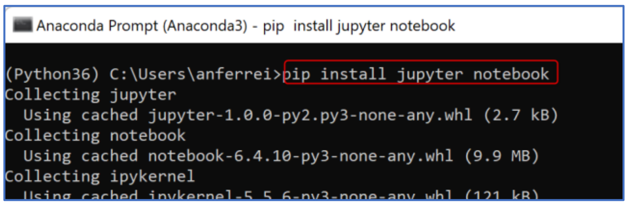
You have successfully installed a Python 3.6 environment in Anaconda.

5. Run Cosmos DB Table API with Jupyter Notebook

After you've created a **Cosmos DB Table API account** and **Anaconda**, your next step is to install the Jupyter Notebook in Anaconda.

1. In the Anaconda prompt run the following command:

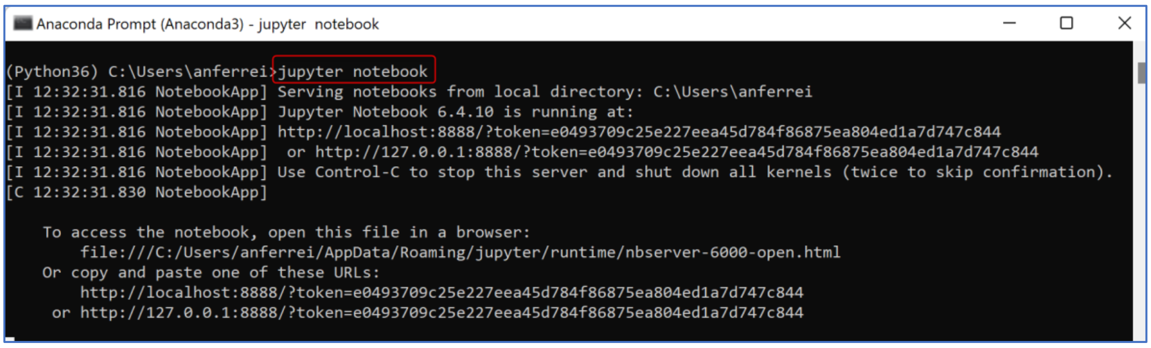
Anaconda prompt:
```
pip install Jupyter notebook
```



```
Anaconda Prompt (Anaconda3) - pip  install jupyter notebook

(Python36) C:\Users\anferrei>pip install jupyter notebook
Collecting jupyter
  Using cached jupyter-1.0.0-py2.py3-none-any.whl (2.7 kB)
Collecting notebook
  Using cached notebook-6.4.10-py3-none-any.whl (9.9 MB)
Collecting ipykernel
  Using cached ipykernel-5.5.6-py3-none-any.whl (121 kB)
```

2. Now let's open a Jupyter Notebook with the Anaconda Prompt. The Anaconda Prompt window should look something like the image below. At the Anaconda Prompt type:

Anaconda prompt:
```
jupyter notebook
```

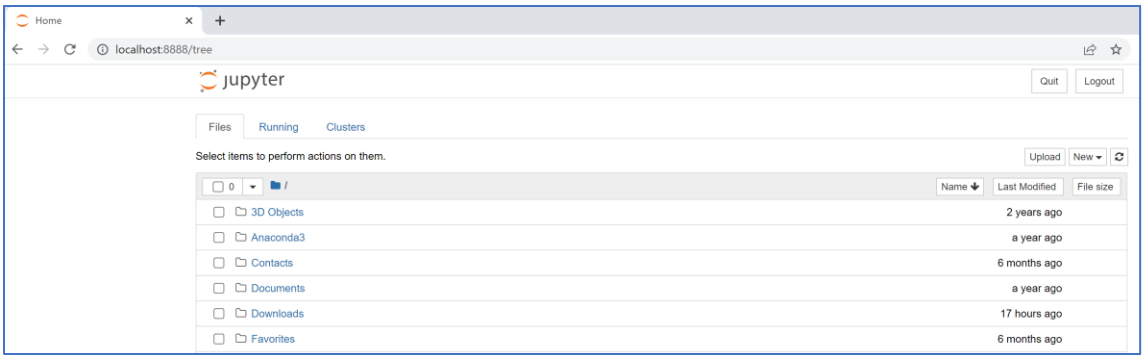


```
Anaconda Prompt (Anaconda3) - jupyter  notebook

(Python36) C:\Users\anferrei>jupyter notebook
[I 12:32:31.816 NotebookApp] Serving notebooks from local directory: C:\Users\anferrei
[I 12:32:31.816 NotebookApp] Jupyter Notebook 6.4.10 is running at:
[I 12:32:31.816 NotebookApp] http://localhost:8888/?token=e0493709c25e227eea45d784f86875ea804ed1a7d747c844
[I 12:32:31.816 NotebookApp]  or http://127.0.0.1:8888/?token=e0493709c25e227eea45d784f86875ea804ed1a7d747c844
[I 12:32:31.816 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 12:32:31.830 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///C:/Users/anferrei/AppData/Roaming/jupyter/runtime/nbserver-6000-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=e0493709c25e227eea45d784f86875ea804ed1a7d747c844
     or http://127.0.0.1:8888/?token=e0493709c25e227eea45d784f86875ea804ed1a7d747c844
```

This command starts the Jupyter notebook server. The output in the Anaconda Prompt will look something like the output shown in the above image.

3. A web browser should open, and you should be able to see the **Jupyter file browser**. If a web browser doesn't open automatically, you can copy the web address from the **Anaconda Prompt** and paste it into a web browser's address bar (you should paste the URL http://localhost:8888 in your browser in case the web browser does not open automatically).

4. You should see in your browser something similar to the image below:



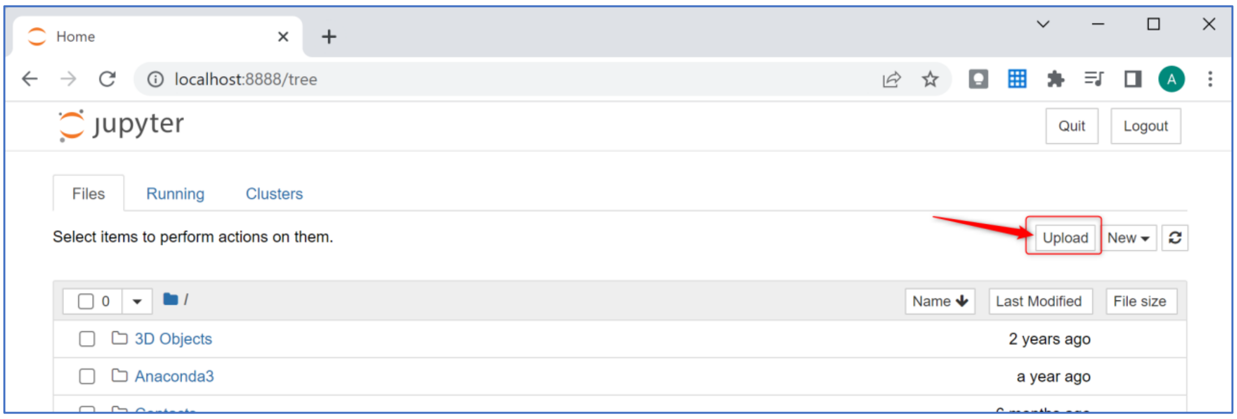
Select items to perform actions on them.

3D Objects	2 years ago
Anaconda3	a year ago
Contacts	6 months ago
Documents	a year ago
Downloads	17 hours ago
Favorites	6 months ago

6. Download files provided in the Moodle

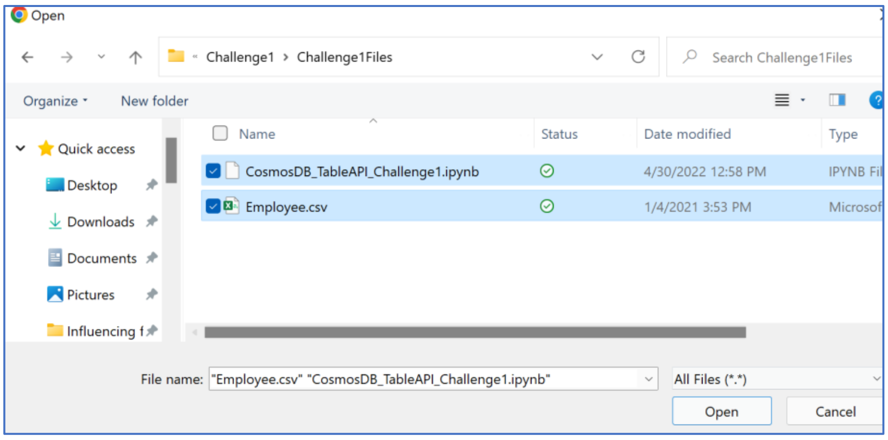
Please make sure you download the file ***CosmosDB_TableAPI_Challenge1.ipynb*** and the file ***Employee.csv*** to your local computer. This file is provided in the Moodle page and it will be used to connect to your Cosmos DB Table API account using a Python SDK.

1. In the web browser upload your files (***CosmosDB_TableAPI_Challenge1.ipynb*** and ***Employee.csv***), by clicking the **Upload** button on the top right corner. You can upload both files at the same time.



Select items to perform actions on them.

3D Objects		2 years ago
Anaconda3		a year ago

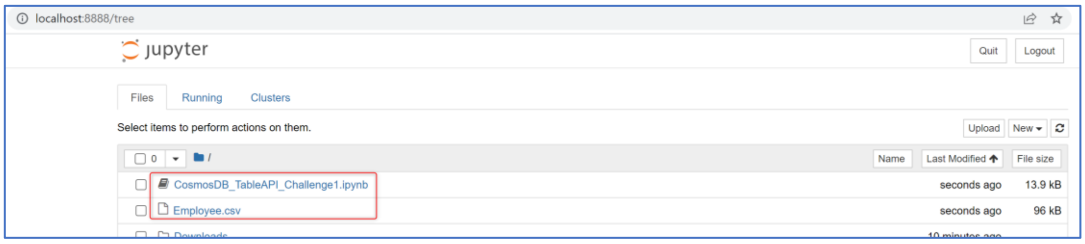


Name	Status	Date modified	Type
CosmosDB_TableAPI_Challenge1.ipynb	✓	4/30/2022 12:58 PM	IPYNB Fi
Employee.csv	✓	1/4/2021 3:53 PM	Microsof

File name: "Employee.csv" "CosmosDB_TableAPI_Challenge1.ipynb"
All Files (*.*)

2. After uploading, the file should be visible in the Jupyter Notebook web browser. **Make sure the Employee.csv file is in the same folder.**

You can filter by "Last Modified" to make it easy to find the files you just downloaded.



Select items to perform actions on them.

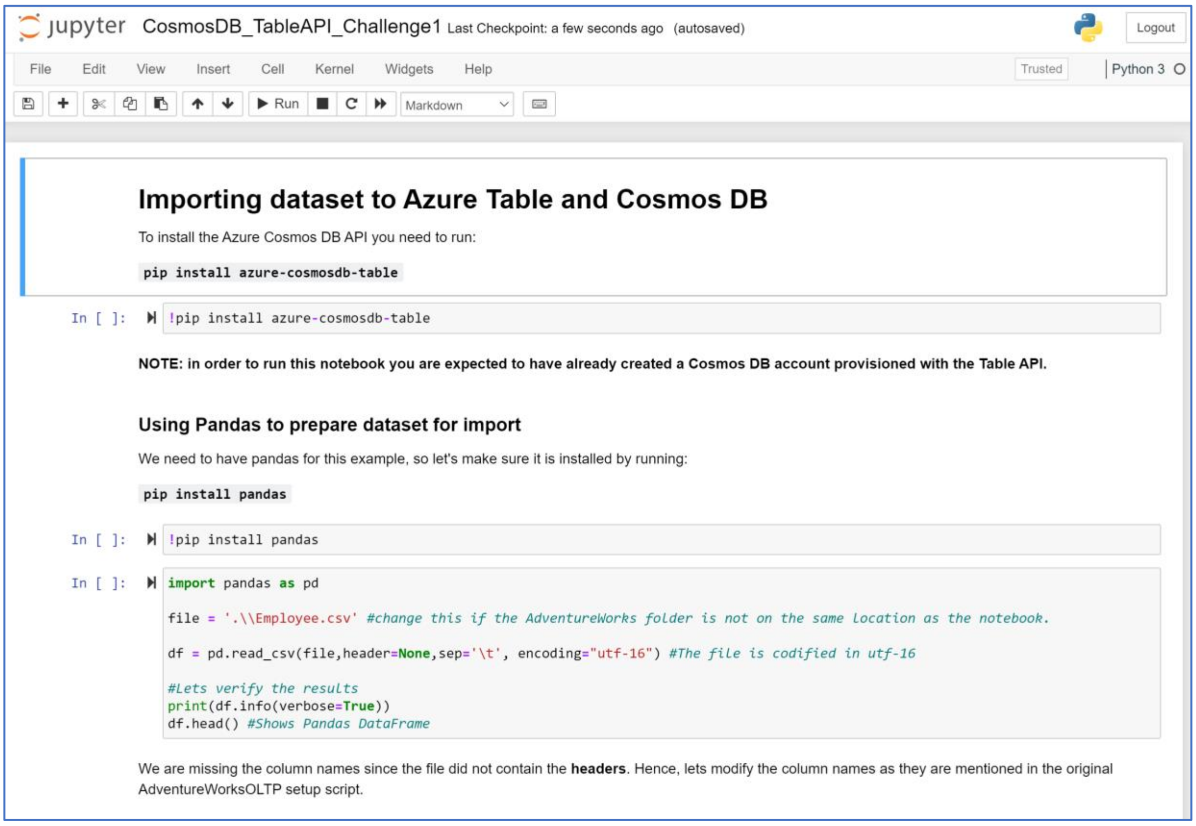
CosmosDB_TableAPI_Challenge1.ipynb	seconds ago	13.9 kB
Employee.csv	seconds ago	96 kB

4. Make sure you have both files(***CosmosDB_TableAPI_Challenge1.ipynb*** and ***Employee.csv***) downloaded.

7. Execute *CosmosDB_TableAPI_Challenge1.ipynb* in Jupyter Notebook

1. Click on the file ***CosmosDB_TableAPI_Challenge1.ipynb*** to check the code.

You should see the following information:



jupyter CosmosDB_TableAPI_Challenge1 Last Checkpoint: a few seconds ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Markdown

Importing dataset to Azure Table and Cosmos DB

To install the Azure Cosmos DB API you need to run:

`pip install azure-cosmosdb-table`

```
In [ ]: !pip install azure-cosmosdb-table
```

NOTE: in order to run this notebook you are expected to have already created a Cosmos DB account provisioned with the Table API.

Using Pandas to prepare dataset for import

We need to have pandas for this example, so let's make sure it is installed by running:

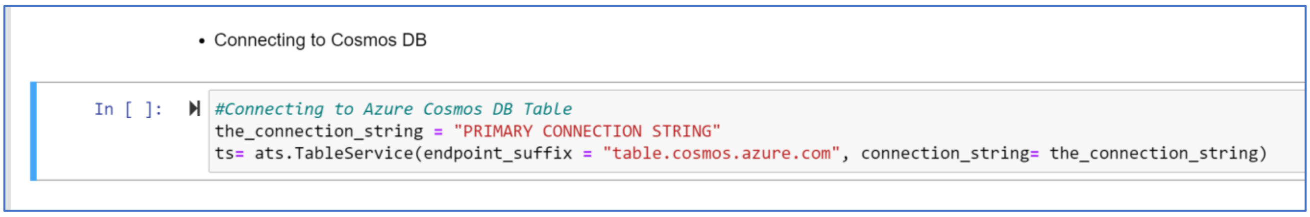
`pip install pandas`

```
In [ ]: !pip install pandas
```

```
In [ ]: import pandas as pd

        file = '.\\Employee.csv' #change this if the AdventureWorks folder is not on the same location as the notebook.

        df = pd.read_csv(file,header=None,sep='\t', encoding="utf-16") #The file is codified in utf-16

        #Lets verify the results
        print(df.info(verbose=True))
        df.head() #Shows Pandas DataFrame
```

We are missing the column names since the file did not contain the **headers**. Hence, lets modify the column names as they are mentioned in the original AdventureWorksOLTP setup script.

2. Like a Data Scientist, we are going first use Pandas to prepare and work on the dataset to import data. Make sure that you follow all the steps carefully until "**Connecting to Cosmos DB Table API**".

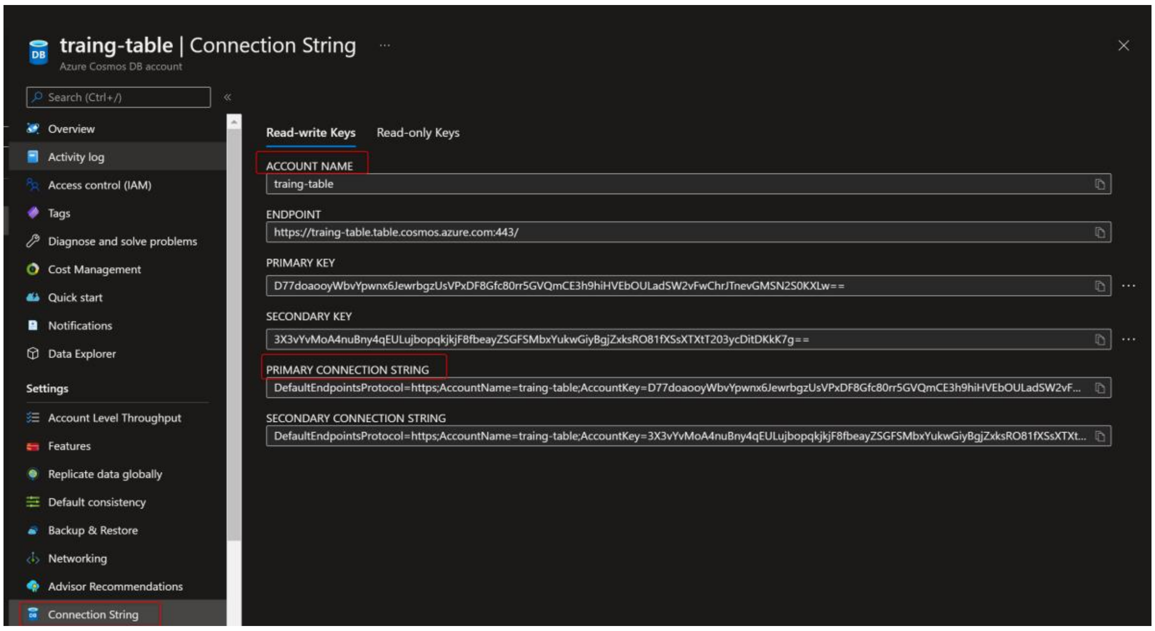


• Connecting to Cosmos DB

```
In [ ]: #Connecting to Azure Cosmos DB Table
        the_connection_string = "PRIMARY CONNECTION STRING"
        ts= ats.TableService(endpoint_suffix = "table.cosmos.azure.com", connection_string= the_connection_string)
```

You can start running and interacting with the Jupyter Notebook application. When you reach "**Connecting to Cosmos DB Table API**" stop and read the next instructions.

3. When you reach "**Connecting to Cosmos DB Table API**". Please make sure that you copy the *PRIMARY CONNECTION STRING* value to use in your application.



ACCOUNT NAME
traing-table

ENDPOINT
https://traing-table.table.cosmos.azure.com:443/

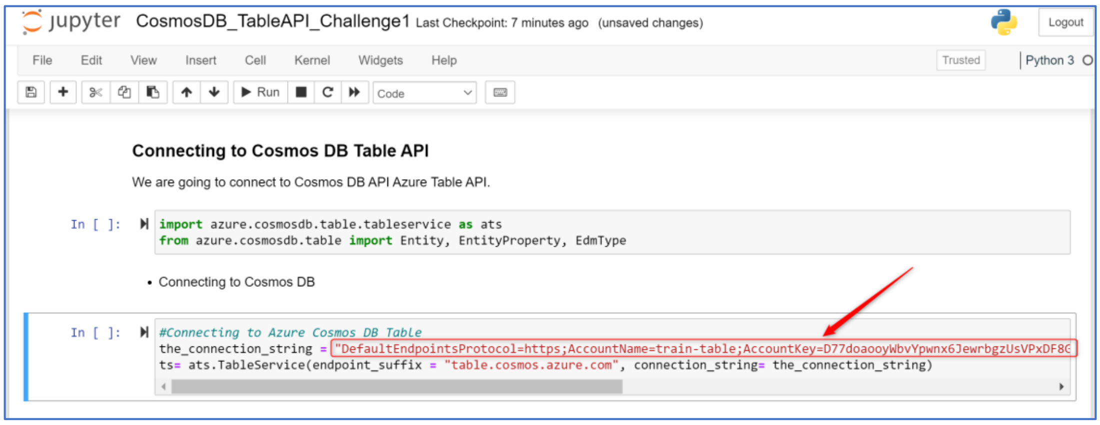
PRIMARY KEY
D77doaooyWbvYpwnx6JewrbgzUsVPxDF8Gfc80rr5GVQmCE3h9hiHVEbOULadSW2vFwChrJTnevGMSN2S0KXLw==

SECONDARY KEY
3X3vYvMoA4nuBny4qEULujbopqkjkjF8fbeayZSGFSMbxYukwGiyBgjZxksRO81fXSsXTXtT203ycDitDKkK7g==

PRIMARY CONNECTION STRING
DefaultEndpointsProtocol=https;AccountName=traing-table;AccountKey=D77doaooyWbvYpwnx6JewrbgzUsVPxDF8Gfc80rr5GVQmCE3h9hiHVEbOULadSW2vF...

SECONDARY CONNECTION STRING
DefaultEndpointsProtocol=https;AccountName=traing-table;AccountKey=3X3vYvMoA4nuBny4qEULujbopqkjkjF8fbeayZSGFSMbxYukwGiyBgjZxksRO81fXSsXTXt...

4. This will be only change that you are going to do in the Jupyter notebook file.


jupyter CosmosDB_TableAPI_Challenge1 Last Checkpoint: 7 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

Connecting to Cosmos DB Table API

We are going to connect to Cosmos DB API Azure Table API.

```
In [ ]:  import azure.cosmosdb.table.tableservice as ats
         from azure.cosmosdb.table import Entity, EntityProperty, EdmType
```

• Connecting to Cosmos DB

```
In [ ]:  #Connecting to Azure Cosmos DB Table
         the_connection_string = "DefaultEndpointsProtocol=https;AccountName=train-table;AccountKey=D77doaooyWbvYpwnx6JewrbgzUsVPxDF8G
         ts= ats.TableService(endpoint_suffix = "table.cosmos.azure.com", connection_string= the_connection_string)
```

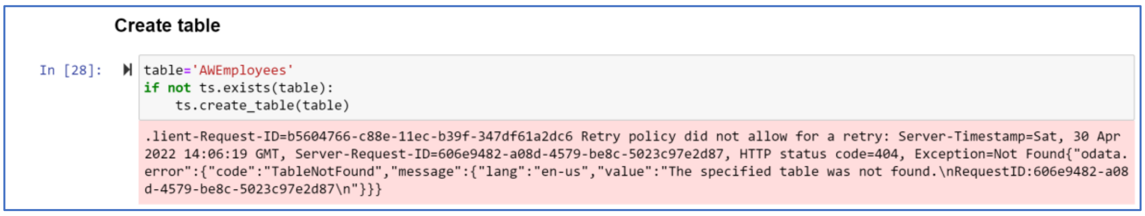
5. After changing the connection string to your *PRIMARY CONNECTION STRING* don't forget to save the file.

6. You can now execute this cell. If everything goes right, you will not receive any error.

Please proceed to the next page.

8. Create Table

1. Create the table **AWEmployees** by executing the following step:

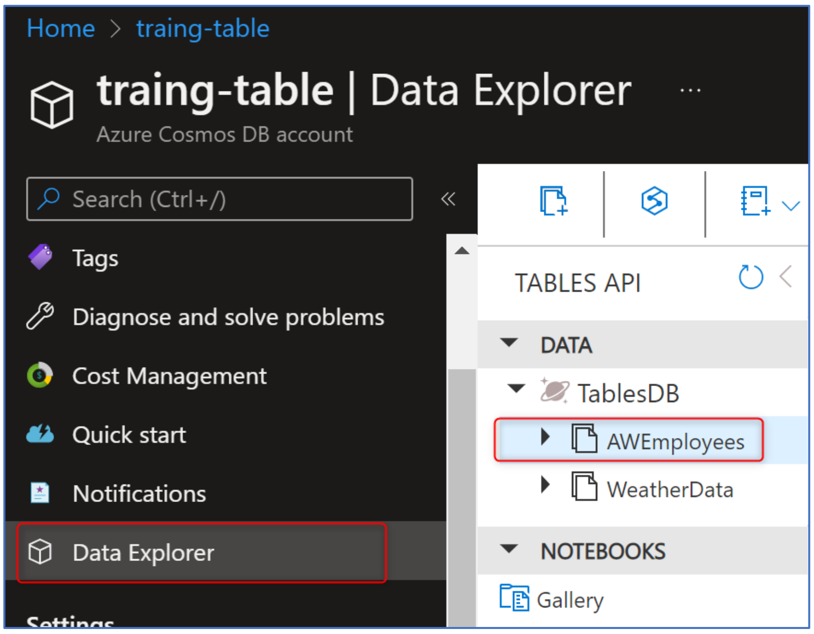


Create table

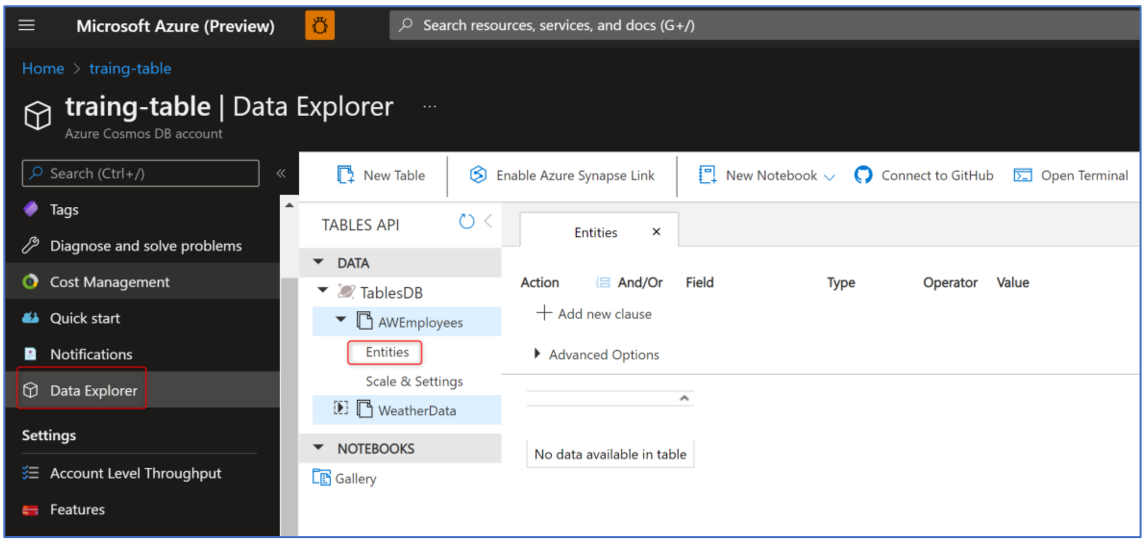
```
In [28]:  table='AWEmployees'
          if not ts.exists(table):
              ts.create_table(table)
```

```
.lient-Request-ID=b5604766-c88e-11ec-b39f-347df61a2dc6 Retry policy did not allow for a retry: Server-Timestamp=Sat, 30 Apr
2022 14:06:19 GMT, Server-Request-ID=606e9482-a08d-4579-be8c-5023c97e2d87, HTTP status code=404, Exception=Not Found{"odata.
error":{"code":"TableNotFound","message":{"lang":"en-us","value":"The specified table was not found.\nRequestID:606e9482-a08
d-4579-be8c-5023c97e2d87\n"}}}
```

2. Go to the Azure portal and confirm the table **AWEmployees** has been created.

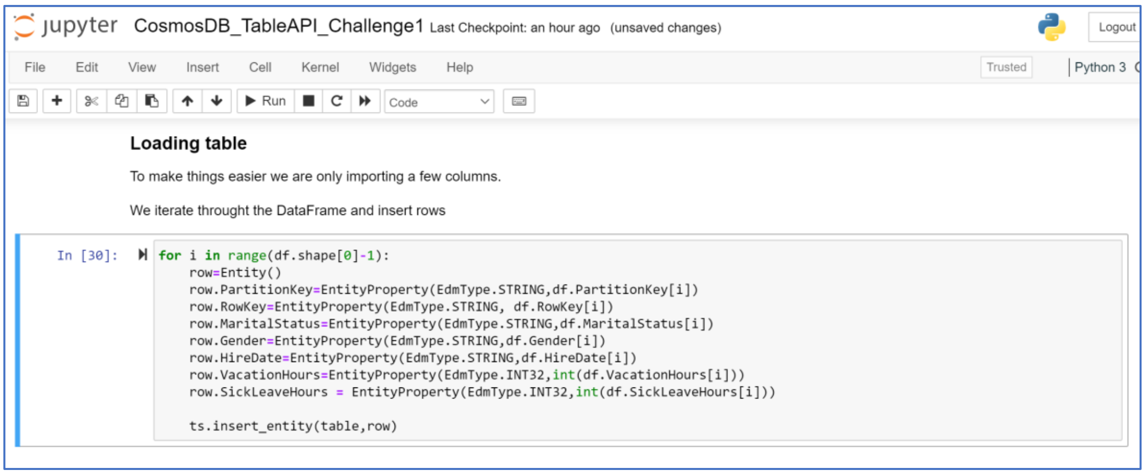


3.In the Azure portal click on the table **AWEmployees** and then in **Entities**. You will be able to confirm the table is empty.



9. Loading data to table AWEmployees

1. After creating the table, execute the **Loading table** cell which will iterate through the DataFrame and insert rows in the table **AWEmployees**. This step may take a few seconds to complete.


jupyter CosmosDB_TableAPI_Challenge1 Last Checkpoint: an hour ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

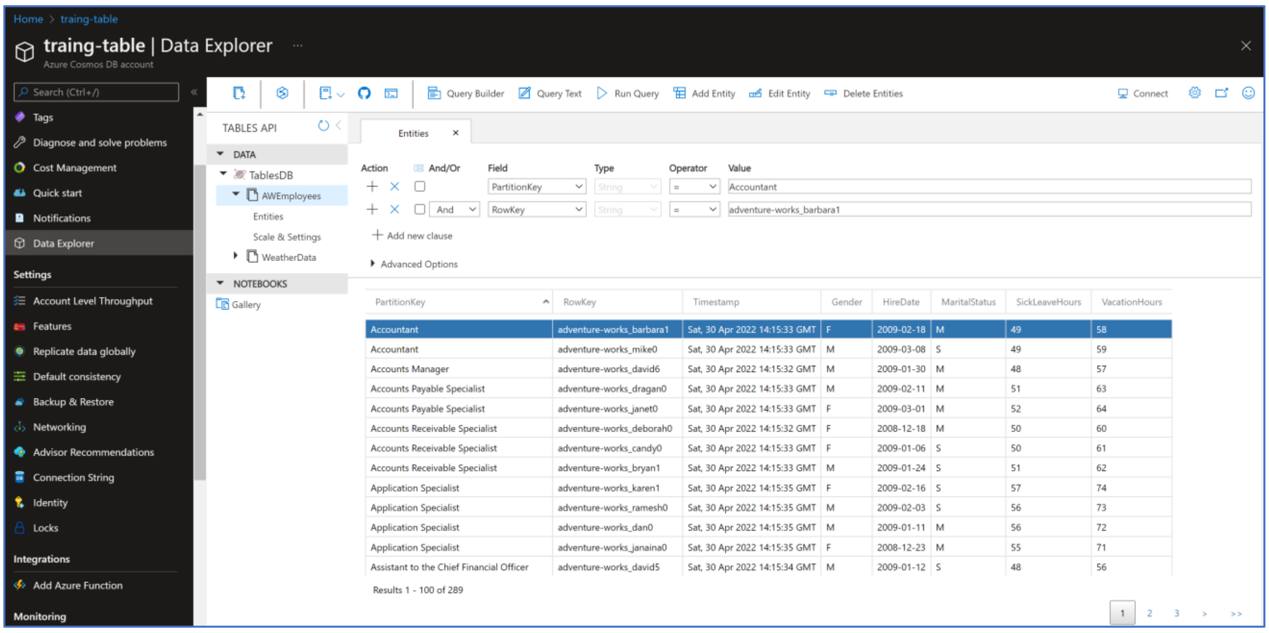
Loading table

To make things easier we are only importing a few columns.

We iterate throught the DataFrame and insert rows

```
In [30]:  for i in range(df.shape[0]-1):
              row=Entity()
              row.PartitionKey=EntityProperty(EdmType.STRING,df.PartitionKey[i])
              row.RowKey=EntityProperty(EdmType.STRING, df.RowKey[i])
              row.MaritalStatus=EntityProperty(EdmType.STRING,df.MaritalStatus[i])
              row.Gender=EntityProperty(EdmType.STRING,df.Gender[i])
              row.HireDate=EntityProperty(EdmType.STRING,df.HireDate[i])
              row.VacationHours=EntityProperty(EdmType.INT32,int(df.VacationHours[i]))
              row.SickLeaveHours = EntityProperty(EdmType.INT32,int(df.SickLeaveHours[i]))

              ts.insert_entity(table,row)
```

2. Go to the Azure portal and confirm the table **AWEmployees** has been populated.



Action	And/Or	Field	Type	Operator	Value
+ × ☐		PartitionKey	String	=	Accountant
+ × ☐	And	RowKey	String	=	adventure-works_barbara1

+ Add new clause

Advanced Options

PartitionKey	RowKey	Timestamp	Gender	HireDate	MaritalStatus	SickLeaveHours	VacationHours
Accountant	adventure-works_barbara1	Sat, 30 Apr 2022 14:15:33 GMT	F	2009-02-18	M	49	58
Accountant	adventure-works_mike0	Sat, 30 Apr 2022 14:15:33 GMT	M	2009-03-08	S	49	59
Accounts Manager	adventure-works_david6	Sat, 30 Apr 2022 14:15:32 GMT	M	2009-01-30	M	48	57
Accounts Payable Specialist	adventure-works_dragan0	Sat, 30 Apr 2022 14:15:33 GMT	M	2009-02-11	M	51	63
Accounts Payable Specialist	adventure-works_janet0	Sat, 30 Apr 2022 14:15:33 GMT	F	2009-03-01	M	52	64
Accounts Receivable Specialist	adventure-works_deborah0	Sat, 30 Apr 2022 14:15:32 GMT	F	2008-12-18	M	50	60
Accounts Receivable Specialist	adventure-works_candy0	Sat, 30 Apr 2022 14:15:33 GMT	F	2009-01-06	S	50	61
Accounts Receivable Specialist	adventure-works_bryan1	Sat, 30 Apr 2022 14:15:33 GMT	M	2009-01-24	S	51	62
Application Specialist	adventure-works_karen1	Sat, 30 Apr 2022 14:15:35 GMT	F	2009-02-16	S	57	74
Application Specialist	adventure-works_ramesh0	Sat, 30 Apr 2022 14:15:35 GMT	M	2009-02-03	S	56	73
Application Specialist	adventure-works_dan0	Sat, 30 Apr 2022 14:15:35 GMT	M	2009-01-11	M	56	72
Application Specialist	adventure-works_janaina0	Sat, 30 Apr 2022 14:15:35 GMT	F	2008-12-23	M	55	71
Assistant to the Chief Financial Officer	adventure-works_david5	Sat, 30 Apr 2022 14:15:34 GMT	M	2009-01-12	S	48	56

Results 1 - 100 of 289

1 2 3 > >>

10. Other Jupyter Notebook steps

1. Continue executing all the steps in the Jupyter notebook. You are going to:

- Query the **AWEmployees** table
- Learn how to update a record
- Simulate an entity group transaction

2. Execute/Run the notebook cell-by-cell and follow and read the Jupyter Notebook instructions carefully.

3. Don't forget to delete and clean up your **AWEmployees** table, so you don't lose Azure account credits unnecessarily.