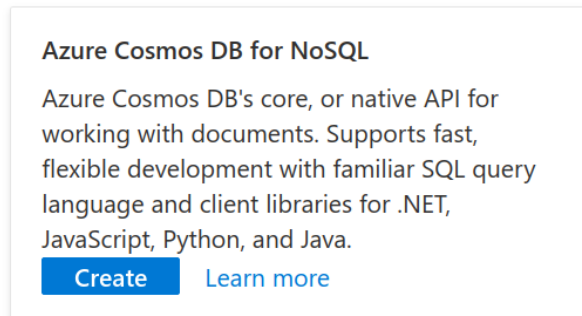
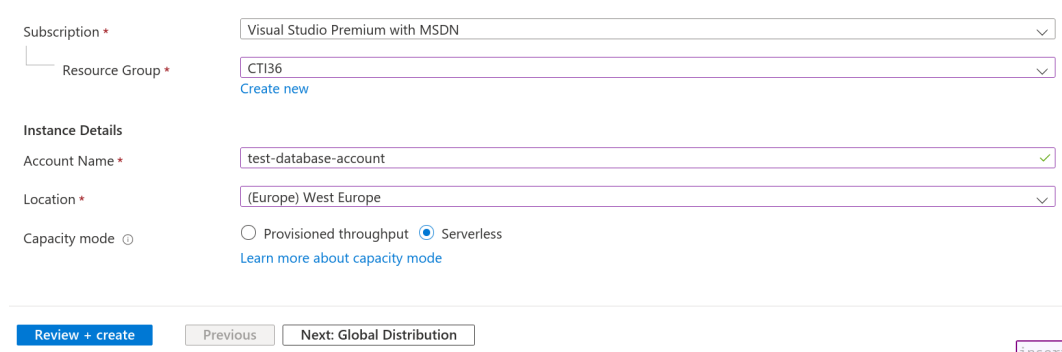


# Azure Cosmos DB for NoSQL

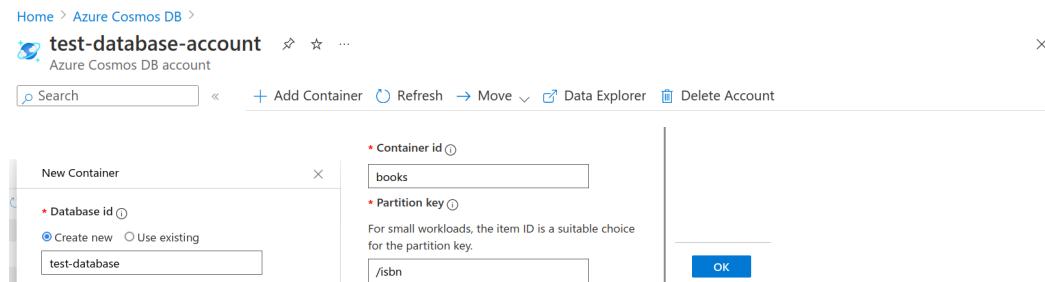
1. Creati o baza de date de tip NoSQL in Azure. Cautati Azure Cosmos DB



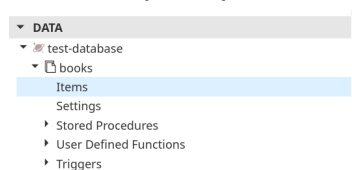
2. Alegeti un nume pentru numele contului si selectati serverless

A screenshot of the Azure portal's "Create new" page for Azure Cosmos DB. The form is titled "Subscription \*". It has two dropdown menus: "Resource Group \*" with "CTI36" selected and a "Create new" link below it, and "Instance Details" with "test-database-account" selected and a green checkmark to its right. Below these are "Location \*" with "(Europe) West Europe" selected and a green checkmark to its right, and "Capacity mode" with "Provisioned throughput" and "Serverless" (selected) radio buttons. A "Learn more about capacity mode" link is below the radio buttons. At the bottom, there are three buttons: "Review + create" (blue), "Previous" (grey), and "Next: Global Distribution" (grey).

3. Dati Next pana la capat sa vedeti ce mai puteti selecta sau apasati Review + create si dupa dati Create. Dupa ce se creeaza baza de data, dati click pe numele ales sa vedeti mai multe detalii. Intrati in Overview si dati + Add Container. Partition Key reprezinta o valoare din JSON pentru a identifica un grup de obiecte.



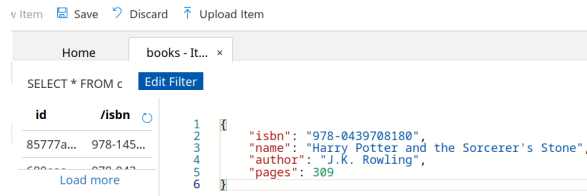
4. In meniul principal trebuie sa va apara container-ul creat anterior



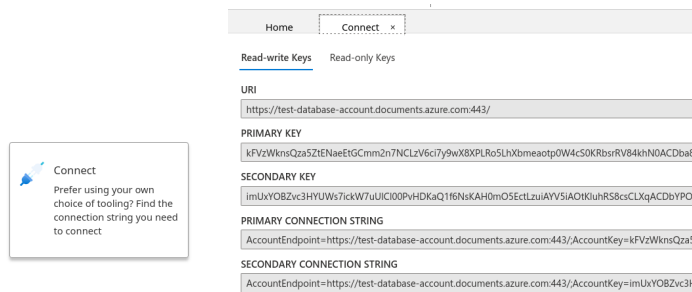
5. Daca dati click pe New Item puteti sa adaugati carti:



6. Scrieti un fisier JSON care sa contina datele despre o carte si dupa dati Save:



7. Dupa acest pas o sa va apara mai multe informatii despre camp. Pentru a sterge o carte, dupa ce ati selectat ID-ul din partea stanga, apasati Delete.
8. Pentru a folosi baza de date dintr-un limbaj de programare trebuie sa folositi SDK-ul oferit de Azure ([tutorial](#)). Aveti nevoie de datele de conectare. Dupa ce ati selectat Data Explorer din partea stanga, apasati Connect din partea dreapta.



9. Creati un venv din python si instalati aplicatia azure-cosmos
- ```
sudo apt-get install python3-venv  
python3 -m venv venv  
source venv/bin/activate  
pip install azure-cosmos
```
10. Scrieti un program de Python care sa citeasca cartile din baza de date si sa le afiseze pe ecran.
- ```
import json  
from azure.cosmos import CosmosClient, PartitionKey
```

```
COSMOS_URI = "https://test-database-account.documents.azure.com:443/"  
COSMOS_KEY = "kFVzWkns..."
```

```
client = CosmosClient(url=COSMOS_URI, credential=COSMOS_KEY)  
database = client.create_database_if_not_exists(id="test-database")  
partitionKeyPath = PartitionKey(path="/isbn")  
container = database.create_container_if_not_exists(  
    id="books", partition_key=partitionKeyPath  
)
```

```

QUERY = "SELECT * FROM books"
items = container.query_items(
    query=QUERY, enable_cross_partition_query=True
)

```

```

for item in items:
    print(json.dumps(item, indent=True))

```

11. Adaugati o carte noua:  

```

from uuid import uuid4
book_id = str(uuid4())
new_book = {
    "isbn": "979-8745274824",
    "name": "The Great Gatsby",
    "author": "Scott Fitzgerald",
    "pages": 110,
    "id": book_id
}
container.create_item(new_book)

```
12. Pentru a gasi un item aveti nevoie de doua informatii: ID si Partition\_Key  

```

item = container.read_item(id="ca33bb2f-6330-4820-bffd-61c9e51deda7",
partition_key="979-8745274824")

```

## Azure Storage (Blob - Object Store)

1. Navigati catre Storage account si creati un cont nou

**Instance details**

If you need to create a legacy storage account type, please click [here](#).

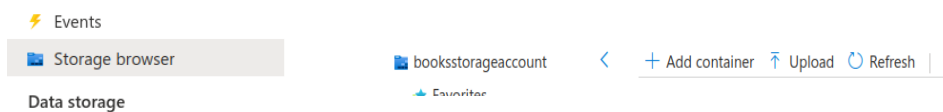
Storage account name ⓘ \*

Region ⓘ \*

Performance ⓘ \*   
☒ Standard: Recommended for most scenarios (general-purpose v2 account)  
☐ Premium: Recommended for scenarios that require low latency

Redundancy ⓘ \*

2. Apasati Review + Create
3. Apasati Storage browser din partea stanga si apoi Add container



4. Creati un nou container cu Public access level: Blob

Name \*

books ✓

Public access level ⓘ

Blob (anonymous read access for blobs only) ▾

⚠ Blobs within the container can be read by anonymous request, but container data is not available. Anonymous clients cannot enumerate the blobs within the container.

▼ Advanced

5. Selectati noul container si incarcati o imagine folosind butonul Upload

booksstorageaccount < + Add Directory ↑ Upload 🔒 Change access level ↻ Refresh

★ Favorites

> ⚙ Recently viewed

▼ ☑ Blob containers

📁 \$logs

📁 books

View all

📁 File shares

📁 Queues

📁 Tables

Blob containers > books

Authentication method: Access key (Switch to Azure AD User Account)

▼ Add filter

🔍 Search blobs by prefix (case-sensitive)

Showing all 0 items

☐ Name

No items found

Upload blob

1 file(s) selected: fahrenheit-451.jpg

Drag and drop files here or [Browse for files](#)

☐ Overwrite if files already exist

▼ Advanced

Upload

6. Pentru a gasi URL-ul la care se afla imaginea, apasati pe cele 3 puncte din dreptul numelui imaginii si apasati Properties

Overview

Blob: fahrenheit-451.jpg

📁 Save ✕ Discard ⬇ Download ↻ Refresh 🗑 Delete ↺ Change tier

Properties

URL <https://booksstorageaccount.blob.core.windows.net/books/fahrenheit-451.jpg>

LAST MODIFIED 11/5/2022, 4:01:18 PM

CREATION TIME 11/5/2022, 4:01:18 PM

VERSION ID -

7. Apasati pe butonul albastru si dati paste in browser.

8. Pentru a incarca o imagine folosind Python, urmariti [tutorialul](#)

9. Instalati aplicatiile necesare:

pip install azure-storage-blob azure-identity

10. Gasiti connection\_string-ul si salvati-l

Events

Storage browser

Data storage

Containers

File shares

Queues

Tables

Security + networking

Networking

Azure CDN

Access keys

key1 🔒 Rotate key

Last rotated: 11/5/2022 (0 days ago)

Key

..... Show

Connection string

..... Show

key2 🔒 Rotate key

Last rotated: 11/5/2022 (0 days ago)

Key

..... Show

Connection string

..... Show

11. Salvati imaginea in directorul curent si creati un fisier de Python care sa contina urmatorul cod:

```
from azure.identity import DefaultAzureCredential
from azure.storage.blob import BlobServiceClient, BlobClient, ContainerClient
```

```
connect_str = "..."  
container_name = "books"  
file_name = "hp-1.jpg"
```

```
blob_service_client = BlobServiceClient.from_connection_string(connect_str)  
blob_client = blob_service_client.get_blob_client(container=container_name,  
blob=file_name)
```

```
with open(file=file_name, mode="rb") as data:  
    blob_client.upload_blob(data, content_type="image/jpeg")
```

```
print(blob_client.url)
```