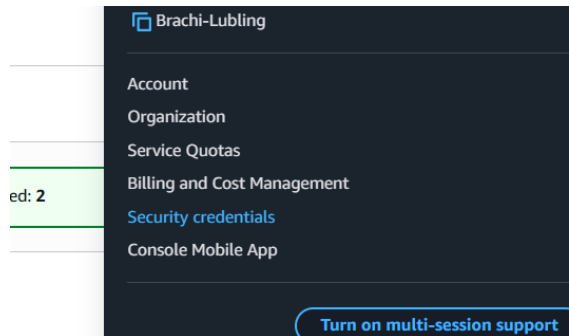


# Dynamodb-aws

# יצירת חיבור בין המחשב לaws

לצורך הגדרת aws cli במחשב  
להתקין מהקישור הזה: <https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>  
לפעול לפי ההוראות לווינדוס



נכנסים לחשבון aws עומדים על השם של היוזר (בצד ימין למעלה) בוחרים security credentials

## Access keys (1)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

Create access key

AKIAXKTM7CUFVSB77UQR

Description

my home first key

Status

Active

Actions

יוצרים access key חדש

אח"כ פותחים את ה command prompt כותבים aws configure  
מכניסים key id את key id של key נוצר  
אח"כ secret את secret של key שנוצר.  
us-east-1 : region  
פלט אפשר להשאיר ריק

# הגדרת פרויקט חדש | פייתון (אפשר בעוד שפות)

יצירת פרויקט:

פתיחה של תקיט הפרויקט ב vs code

לאחר מכן יש להגדיר סביבה וירטואלית venv לצורך בידוד הפרויקט משאר הפרויקטים במחשב

אופן ההגדרה בterminal:

```
python -m venv venv
```

ולאחר מכן:

```
venv\Scripts\activate
```

הוספת הספרייה המתאימה:

לצורך החיבור בין aws dynamodb לpython יש להתקין את הספרייה boto3

בטרמינל:

```
pip install boto3
```

קבצי הפרויקט:

בתקיט הפרויקט מלבד התקיה של venv שנוצרה אוטומטית יש להוסיף את 4 הקבצים הבאים:

App.py – קובץ ההרצה הסופי דרכו יוצרים ושולפים ישויות מהדאטה

Config.py – הגדרת משתנים קבועים, שם הטבלה בדאטה ועוד

Dynamo.py-הלוגיקה של הקוד בפועל (פונק' הוספה ושליפה)

Requirements.txt-קובץ שכולל את השמות של הספריות בהן השתמשנו בפרויקט לצורך התקנה שלהן בהרצה של הפרויקט

פירוט בהמשך. - -

# יצירת טבלה בDB | באמצעות הקונסול-ניתן גם באמצעות קוד

יש להכנס לחשבון aws ושם להכנס לdynamodb

**New global secondary index**  
Create global secondary indexes to query attributes outside the primary key of your original table. [Learn more](#)

**Index name**  
select by x optional y  
Between 3 and 255 characters. Only A-Z, a-z, 0-9, underscore characters, hyphens, and periods allowed.

**Partition key**  
The partition key is part of the table's primary key. It is a hash value that is used to retrieve items from your table and allocate data across hosts for scalability and availability.

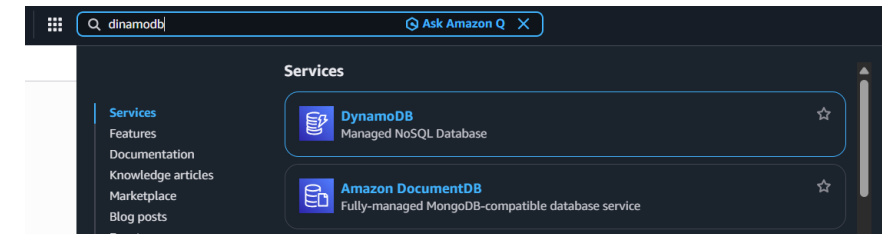
**Attribute** x **Data type** String

[Add new attribute](#)  
You can add up to 3 more attributes.

**Sort key - optional**  
You can use a sort key as the second part of a table's primary key. The sort key allows you to sort or search among all items sharing the same partition key.

**Attribute** y **Data type** String

[Add new attribute](#)  
You can add up to 3 more attributes.



בתפריט בצד שמאל להכנס לtables וללחוץ על create table

יש להכניס את שם הטבלה

בpartition key יש להכניס את שם השדה (id) primary key  
לשנות את ההגדרות לcustomize setting

בsecondary indexes יש להוסיף בglobal index את השדות שנרצה לבצע שליפה לפיהם (GSI)

לכל אינדקס יש את השם של האינדקס ואת שם השדה עליו יתבצע הquery ב-partition key

במידה ורוצים ליצור אפשרות של שליפה עבור שדה X + שליפה עבור שדה X וגם Y את שדה X מגדירים בpartition key ואת שדה Y בsort key  
(במידה ורוצים שדה בודד לשליפה על פיו-משאירים את sort key ריק)  
הערה: ניתן לשלב יותר מsort key אחד לIndex

ללחוץ על יצירת הטבלה

# כתיבת הקוד בפועל | פייתון

```
import boto3
from config import TABLE_NAME, AWS_REGION, DATE_INDEX, SHOW_CITY_INDEX
from boto3.dynamodb.conditions import Key

dynamodb = boto3.resource('dynamodb', region_name=AWS_REGION)
table = dynamodb.Table(TABLE_NAME)
```

בקובץ config יש להגדיר משתנים קבועים בהם נשתמש בקבצי הלוגיקה. המשתנים הם:

Talbe\_name-עבור שם הטבלה (כמו שנקראה בבניית הטבלה בaws)  
Idx\_name-שם האינדקס עבור כל אינדקס שיצרנו- יוצרים משתנה עם השם שלו. שוב יש להקפיד שהערך של המשתנה זהה לשם האינדקס כפי שהגדרנו בבניית הטבלה)  
Aws\_region-עבור ההתחברות לדאטה שלנו

בקובץ dynamon נבצע את הלוגיקה בפועל:  
בשלב הראשון נייבא את הספריות הדרושות.  
לאחר מכן נגדיר את החיבור לdynamodb של aws ואת החיבור לטבלה שלנו

כעת מימוש הפונקציות בפועל:  
לtable יש את הפונקציות הבאות:  
put\_item(Item={"ID": ID, "field1":f1}) - הפונק' מקבלת אובייקט ומכניסה אותו דגאטה או מעדכנת רשומה לפי הid  
get\_item({"Primary\_Key":Primary\_key}) - הפונק' מקבלת אובייקט עם שם השדה המוגדר כprimary key והערך הרצוי ומחזירה את האובייקט הרצוי  
query(indexName=idx\_name, KeyConditionExpression=Key('first\_GSI\_field').eq(first\_GSI\_val) & Key('second\_GSI\_field').eq(second\_GSI\_val))  
הפונק' מחזירה אובייקט שיש לו רשימה של האובייקטים שמקימים את התנאים שנשלחו (השדות של הסינון הם שדות האינדקסים שהגדרנו בטבלה ובקובץ config)  
update\_item()  
delete\_item()  
Scan() - מחזירה את כל הרשומות

יש לשים לב שכאשר יוצרים אובייקט או מעדכנים קוראים לשדה בשם שהוגדר ביצירת הטבלה (לשדות indexes keys)  
כמו"כ פונקציות שמחזירות אובייקטים מחזירות אותם עטופים באובייקט עם שדות metadata ולכן מומלץ לקבל את האובייקט שחוזר מהפונקציה למשתנה שונה result ולהחזיר מהפונקציה את result.get("Item") או result.get("Items",[])) במקרה של query או scan

# כתיבת הקוד בפועל | פייתון

בקובץ app כותבים את המימוש בפועל של השימוש בפונק',  
מביאים את הפונק' מתוך קובץ dynamo import add,get\_by\_id: dynamo ...  
בודקים את הקוד מפעילים את הפונק', יוצרים משתמשים, שולפים ומציגים אותם.

ניתן לראות את המשתמשים שיצרנו מתוך הקוד גם דרך האתר של aws  
נכנס לדynamodb  
לtables  
נכנס לטבלה שיצרנו  
ונלחץ על הכפתור explore talbe items  
שם נוכל לראות את כל הישועות שיצרנו באופן ויזואלי

בקובץ requirements.txt מכניסים את הספריות בהן השתמשנו,  
נכון לעכשיו הספרייה היחידה היא boto3  
ולכן כותבים בו:  
boto3==1.28.0

## בהצלחה!