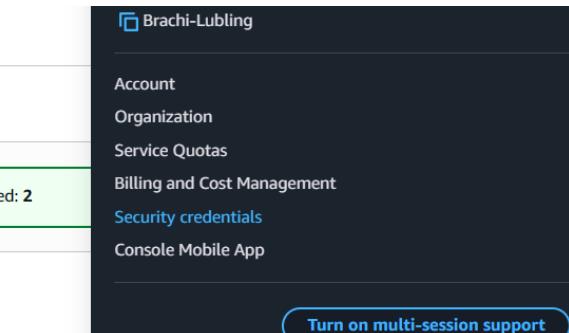


Dynamodb-aws

יצירת חיבור בין המחשב aws

לצורך הגדרת cli aws במחשב
להתקין מהקיים זהה: <https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>
לפעול לפי הוראות לוונדוס



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

A screenshot of the AWS IAM Access Keys page. It shows a table with one row of data. The first column is labeled "Access keys (1)" and contains a link to "AKIAKTM7CUFVSB77UQR". The second column is labeled "Description" and has a link to "aws-home-first-key". The third column is labeled "Status" and shows "Active". The fourth column is labeled "Actions" with a dropdown arrow. At the top right of the table is a blue button labeled "Create access key".

יצרים access key חדש

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

הגדרת פרויקט חדש | פיתון (אפשר בעוד שפות)

יצירת פרויקט:
פיתוח של תקית הפרויקט ב code vs פיתוח מכוון יש להגדיר סביבה וירטואלית `venv` לצורך בידוד הפרויקט משאר הפרויקטים במחשב:
אOPEN הגדירה ב terminal:
`python -m venv venv`
ולאחר מכן:
`venv\Scripts\activate`

הוספת הספריה המתאימה:
לצורך החיבור בין aws dynamodb ל python יש להתקין את הספריה `boto3`
בטרמינל:
`pip install boto3`

קבצי הפרויקט:
בתקית הפרויקט מלבד התקיה של `venv` שנוצרה אוטומטית יש להוסיף את 4 הקבצים הבאים:
App.py – קובץ ההרצה הסופי דרכו יוצרים ושולפים ישווות מהדата
Config.py – הגדרת משתנים קבועים, שם הטללה בדатаה ועוד
Dynamo.py-логיקה של הקוד בפועל (פונק' הוספה ושליפה)
Requirements.txt – קובץ שכולל את השמות של הספריות בהן השתמשנו בפרויקט לצורך התקנה של הרצה של הפרויקט

יצירת טבלה בDB | באמצעות הקונסול-ניתן גם באמצעות קוד

יש להכנס לחשבון aws ושם להכנס לddb

New global secondary index

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

Index name

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	Data type
x	String

[Add new attribute](#)

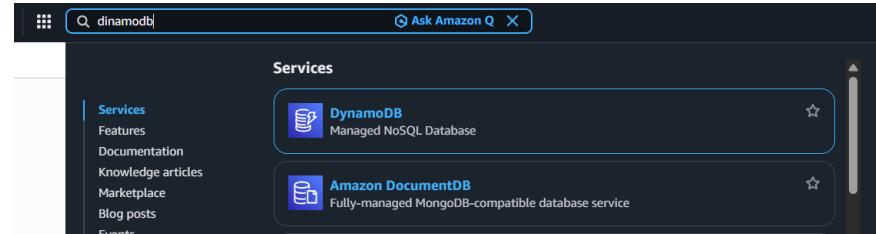
You can add up to 3 more attributes.

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	Data type
א'	String

[Add new attribute](#)

You can add up to 3 more attributes.



בתפריט בצד שמאל להכנס tables ולהזוז על create table

יש להכניס את שם הטבלה
בpartition key יש להכניס את שם השדה ה-primary key (id) costumize setting לשנות את ההגדרות ל-

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

לכל אינדקס יש את השם של האינדקס ואת שם השדה עליו יבוצע הquery בעקבות key-partition key<->attribute

במידה ורוצים ליצור אפשרות של שליפה עבור שדה X + שליפה עבור שדה Y וגם Z 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 מ-1 sort key אחד ל-index

להזוז על יצירת הטבלה

כתיבת הקוד בפועל | פיתון

```
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:

Table_name-עבור שם הטבלה (כמו שנקראת בבנייה הטבלה בaws)

Aws_region-שם האינדקס עבור כל אינדקס שיצרנו. יוצרים משתנה עם השם שלו. שוב יש להזכיר שהערך של המשתנה זהה לשם האינדקס כפי שהגדכנו במבנה הטבלה)

Aws_region-עבור ההתחברות לדאטा שלנו

בקובץYNAMOP� נבצע את הלוגיקה בפועל:
בשלב הראשון ניבא את הספריות הדרשיות.
לאחר מכן נגדיר את החיבור לatable של aws ואת החיבור לatable שלנו

עת מימוש הפונקציות בפועל:
ל

{ "ID": ID, "field1": f1 } - הפונק' מקבלת אובייקט ומכניסה אותו דגאתה או מעדכנת רשומה לפי id

{ "Primary_Key": Primary_Key } - הפונק' מקבלת אובייקט עם שם השדה המוגדר כkey primary 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 (השdots של הסינון הם שdots האינדקסים שהגדכנו בטבלה ובקובץ config)

update_item()

delete_item()

-Scan() - מחזירה את כל הרשומות

יש לשים לב שכאר יוצרים אובייקט או מעדכנים קוראים לשדה בשם שהוגדר ביצירת הטבלה (לשdots indexes וkeys)
כמובן פונקציות שמחזירות אובייקטים מהזרויות אותן עטופים באובייקט עם שdots metadata ולכך מומלץ לקבל את האובייקט שוחרר מהפונקציה למשתנה שונה
וללחזיר מהפונקציה את ("Item") result.get("Item") או result.get([],{}) reslt.get("Items") או scan query או

כתיבת הקוד בפועל | פיתוח

בקובץ הקאם כותבים את המימוש בפועל של השימוש בפונק',
מיבאים את הפונק' מתוך קובץ הסומן dynamo import add,get_by_id:
בודקים את הקוד מפעילים את הפונק', יוצרים משתמשים, שולפים ומציגים אותם.

ניתן לראות את המשתמשים שייצרנו מתוך הקוד גם דרך האתר של aws
נכנס לddb dynamodb tables
נכнес לטבלה שייצרנו explore table items
ונלכז על הכפתור שם נוכל לראות את כל היחסות שייצרנו באופן ויזואלי

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

בהצלחה!