Alfonso Ayala Paloma

alfonsoayalapaloma@gmail.com

Descripción breve

Instrucciones para extraer data de AzureBlobStorage, Base de datos PostgreSQL y API de TMDB

EXTRACCión de fuentes de data

Actividad de Contextualización ETL

Contenido

[Descargar archivo de Azure Blob Storage 2](#_Toc128226201)

[Extracción de data de Base de datos POSTGRESQL en Azure 4](#_Toc128226202)

[Extracción de data de API 7](#_Toc128226203)

[Crear el pipeline 8](#_Toc128226204)

## Descargar archivo de Azure Blob Storage

**Como pre-requisito, se debe tener un archivo en Azure BlobStorage, la cuenta de storage, la clave y la URL para acceder al recurso.**

**En colab, se requieren instalar las librerías**

#Azure BlobStorage (Similar to HDFS) extraction example

!pip install azure-storage-blob azure-identity

from azure.storage.blob import BlobServiceClient

import time

import pandas as pd

**Se definen constantes para los prerrequisitos:**

AZ\_STORAGEACCOUNTURL= "https://<storage\_account>.blob.core.windows.net/"

AZ\_STORAGEACCOUNTKEY= "<secret key>"

AZ\_CONTAINERNAME="<container>"

Crear la función para extracción: Crea una instancia de BlobServiceCliente

Obtiene un blob\_client() para el BLOB que queremos descargar (el archivo),

Lee el BLOB y lo guarda en un archivo local con readinto()

def extract\_azure():

BLOBNAME= "netflix\_titles.csv"

LOCALFILENAME= "netflix\_titles.csv"

t1=time.time()

blob\_service\_client\_instance = BlobServiceClient(account\_url=AZ\_STORAGEACCOUNTURL, credential=AZ\_STORAGEACCOUNTKEY)

blob\_client\_instance = blob\_service\_client\_instance.get\_blob\_client(AZ\_CONTAINERNAME, BLOBNAME, snapshot=None)

with open(LOCALFILENAME, "wb") as my\_blob:

blob\_data = blob\_client\_instance.download\_blob()

blob\_data.readinto(my\_blob)

t2=time.time()

print(("It takes %s seconds to download "+BLOBNAME) % (t2 - t1))

df=pd.read\_csv(LOCALFILENAME)

return df

**Y listo!**

## Extracción de data de Base de datos POSTGRESQL en Azure

Como prerequisito se debe tener una base de datos postgresql a la cual conectarse, de esta, tener el nombre, el host/ip, usuario, clave y puerto.

Primero instalamos algunas librerias para conectarnos

!pip install psycopg2

import psycopg2

import pandas as pd

Definimos los prerequisitos como constantes:

PG\_DB="moviedb"

PG\_HOST="" #server Hostname or ip

PG\_USER="" #db user

PG\_PASS="" #db password

PG\_PORT=5432 #db port

Creamos la función que:

Se conecta a la base con .connect()

Crea cun cursor con conn.cursor()

Ejecuta un query y crea un dataframe con read\_sql\_query()

Cierra cursor y conexión

Convierte el dataframe a un archivo de .csv

#PostgreSQL data extraction example.

def extract\_postgres(tablename):

conn = psycopg2.connect(database=PG\_DB,

host=PG\_HOST,

user=PG\_USER,

password=PG\_PASS,

port=PG\_PORT)

cur = conn.cursor()

sql\_query="SELECT \* FROM public."+tablename

df = pd.read\_sql\_query(sql\_query, conn)

cur.close()

conn.close()

df.to\_csv( tablename+".csv" , index=False)

return df

## Extracción de data de API

Como prerequisito se debe tener una API\_KEY de tmdb.

Primer instalar unas librerías

#API extraction example

import pandas as pd

import requests

import json

import ast

Definir como constantes los prerrequisitos, la API\_KEY y las filas a extraer.

TMDB\_API\_KEY = "<key>"

TMDB\_ROWS=10000

Definir la función de extracción:

def extract\_tmdb\_api():

n=TMDB\_ROWS

response\_list=[]

for movie\_id in range(1,n):

url = 'https://api.themoviedb.org/3/movie/{}?api\_key={}'.format(movie\_id, TMDB\_API\_KEY)

r = requests.get(url)

response\_list.append(r.json())

df = pd.DataFrame.from\_dict(response\_list)

df.to\_csv('tmdb\_original\_'+str(n)+'.csv', index=False)

return df

## Crear el pipeline

El pipeline extrae de las dos tablas de postgres dos dataframes.

Luego extrae azure y de tmdb.

Crea un arreglo de dataframes y lo recorre mostrando .info() y .head()

# Pipeline

dfama = extract\_postgres("amazon\_movie")

dfdis = extract\_postgres("disneyplus\_movie")

dfnet = extract\_azure()

dftmdb = extract\_tmdb\_api()

dfarr=[ dfama, dfdis, dfnet, dftmdb]

for i in dfarr:

print("###############")

print(i.info() )

print(i.head() )

Y listo, ya está finalizada la extracción!

Felicitaciones!!!