Insertion des images dans la table Image

Import des bibliothèques

Entrée [ ]:

```
from lxml import etree
import pandas as pd
import mysql.connector
from mysql.connector import Error
import re
import json
import requests
```

Préambule : Charger productsrooturl

Entrée [ ]:

```
authen_settings = json.load(open("private/authen_prod.json", "r"))
data = {
    'UserName': authen_settings["CustomerId"],
    'PassWord': authen_settings["securityKey"]
        }

# Obtention du token
url = "https://api.bihr.net/api/v2/Authentication/token"
resp = requests.post(url, data).json()
access_token = resp['access_token']
headers = {'Authorization': f'bearer {access_token}', }

# Suffixe de l'adresse des images des produits
rootImage = requests.get(f'https://api.bihr.net/api/v2/Catalog/productsrooturl', h
rootImage = rootImage.text
rootImage
```

Parsage du catalogue Image

Entrée [ ]:

```
# à modifier avant chaque traitement d'un nouveau fichier XML
refPath = 'unzipped_files/cat-images-1.1.2141226-11-2019'

xtree = etree.parse(refPath)
xroot = xtree.getroot()
```

Génération d'un dataframe dfImage

Entrée [ ]:

```
df_cols = ["productId", "defaultDocumentId", "isDefault"]
rows = []

for i,ec in enumerate(xroot.iter('product')):
    # ajout à la liste du dict décrivant l'élément category
    rows.append({"productId": ec.attrib['productId'],
                 "defaultDocumentId": ec.attrib['defaultDocumentId'],
                 "isDefault":ec.attrib['isDefault']})
# transformation de la liste en dataframe
dfImage = pd.DataFrame(rows, columns=df_cols)
dfImage
```

Traitement de isDefault (remplacement de true par 1)

Entrée [ ]:
```
dfImage.isDefault.replace("true","1", regex=True, inplace=True)
dfImage
```

Rapprochement des productID pour la FOREIGN KEY

Entrée [ ]:
```
# à modifier avant chaque traitement d'un nouveau fichier XML
refPath1 = 'unzipped_files/cat-ref-FR3787ED_2019-11-22 11.30.32-22-11-2019_11-48-2

xtree1 = etree.parse(refPath1)
xroot1 = xtree1.getroot()
```

Entrée [ ]:
```
df_cols1 = ["productId"]
rows1 = []
for products in xroot1.iter('products'):
    for a,ec in enumerate(products.getchildren()):
        rows1.append(ec.attrib['productId'])
print(len(rows1))
```

Suppression des 72 productId qui étaient doubles sur le fichier REF

######(TODO : Gérer les doublons REF-Product pour supprimer le traitement ci-dessous)#####

Entrée [ ]:
```
rows2 = []
doublons = []
for i in rows1:
    if i not in rows2:
        rows2.append(i)
    else:
        doublons.append(i)
print(len(rows2))
print(len(doublons))
```

Entrée [ ]:
```
# solution 1 :
# test = list(set(rows2) - set(doublons))
#  print(len(test))
```

Entrée [ ]:
```
# solution 2 :
for i in doublons:
    if i in rows2:
        rows2.remove(i)
print(len(rows2))
```

Comparaison des valeurs de productId entre tables Image et Product

Entrée [ ]:
```
dfImage['productId']
```

Entrée [ ]:
```
integrite = pd.Series(dfImage['productId'].isin(rows2))
integrite
```

Entrée [ ]:
```
#Insertion de la série integrité dans dfImage
dfImage['Integrité'] = integrite
dfImage
```

Insertion des données du dataframe dans la table Image

Entrée [ ]:

```
connection_config = {
                    'host':"localhost",
                    'port': 3308,
                    'database': 'bihr_db',
                    'user': 'BASTIER',
                    'passwd': "DA2019",
                    #'autocommit': True
                    }
```

Entrée [ ]:

```
try:
    connection = mysql.connector.connect(**connection_config)

    for i in range(dfImage.shape[0]):
        img = dfImage.iloc[i]
        if img['Integrité'] == True :
            URL_image = rootImage + "/" + str(img['defaultDocumentId']) + ".jpg"
            print(i)
            print(URL_image)
            imageInsertQuery = """INSERT INTO Image (productId, defaultDocumentId,
                        VALUES
                        ("""+ """'""" + str(img['productId']) + """','"""+ str(
                             """','""" + URL_image + """','""" + str(img['isDef
            #print(imageInsertQuery)
            cursor = connection.cursor()
            result = cursor.execute(imageInsertQuery)
        else:
            continue

    connection.commit()
    print("Insertion datas in Image table successfully ")
    cursor.close()

except mysql.connector.Error as error:
    print("Failed to insert datas in Image table : {}".format(error))

finally:
    if (connection.is_connected()):
        cursor.close()
        connection.close()
        print("MySQL connection is closed")
```

Entrée [ ]:

```

```