Projet Python

Dans le cadre de la formation BTS SIO au lycée la Joliverie nous allons créer un projet python capable de lire un fichier log obtenu du Proxy, par la suite en sortir des données et créer un fichier SQL permettant de rentrer les données dans la base de données KLESOMMER du réseau SIO. Nous avons choisi de le faire avec un environement de développement partagé 'Repl-it'

INDEX :

Maxime LARMIER et Kyllian LE SOMMER de 1SIOA groupe A

1. Lecture et écriture dans un fichier texte en Python :

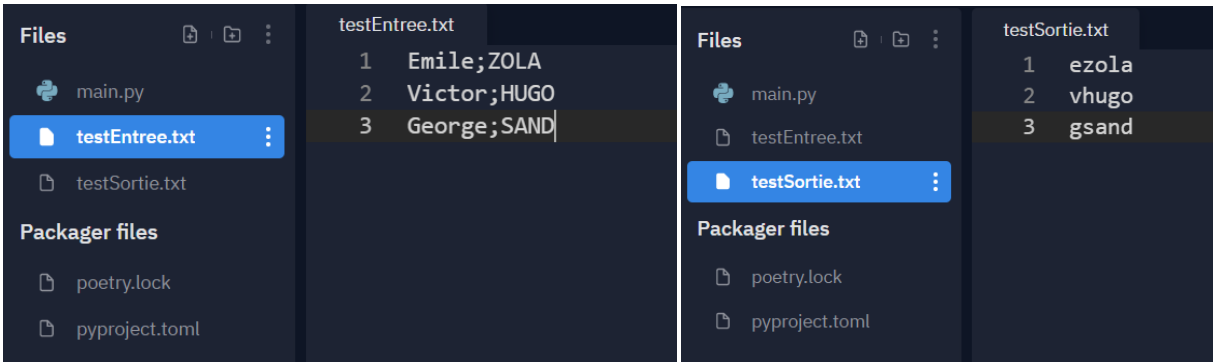
• Création du code pour extraction dans un fichier

Pour commencer nous avons rechercher un moyen de travailler en commun sur le projet.
La solution de partage de fichier est OneDrive de la Joliverie, pour le développement nous avons trouvé un site assez complet qui permet en temps réel de voir les modifications en commun avec notre collègue : [Repl-It]

Pour la Réalisation d'un

```
# -*- coding: utf-8 -*-
"""
date: 26/11/2020
comment: Projet Python
autor: LARMIER Maxime, LE SOMMER Kyllian
"""

import os            #importation de la bibliothèque OS
from os import chdir #importation de chdir depuis la bibliothèque OS

auto = input("Remplissage automatique des variables par le répetoire courant et
'testEntree.txt' en tant que fichier source ? (O/N) > ")
if auto != "O":
  project_path = input("entrer le repertoire du fichier (ex:/home/ex/project) ou 'ici'
pour sélectionner le répertoire courant > ")
  if project_path == "ici":
          project_path = os.getcwd()

#vérification du chemin d'accès
  while not os.path.exists(project_path):
    project_path = input("le repertoire n'existe pas, réessayer > ")

  chdir(project_path)
  print("\nrépertoire courant changé pour: ", project_path,"\n")

  fichierlog = input("entrez le nom du fichier texte (ex: testEntree.txt) > ")

#vérification de l'existence du ficher
  while not os.path.exists(fichierlog):
    fichierlog = input("le fichier n'existe pas, réessayer > ")
  print("\n (", fichierlog, ") Sélectionné\n")
else:
  project_path = os.getcwd()
  fichierlog = "testEntree.txt"
f=open(fichierlog,'r')      # ouverture de testentree en lecture
g=open('testSortie.txt','w')  # ouverture de testSortie en écriture
for ligne in f :  # boucle de lecture de 'testEntrée' et de création de 'testSortie'
    totalMots=ligne.split(";")  # séparation des mots
    a=(totalMots[0][0])         # sélection des premiers caractères
    b=(totalMots[1])            # sélection du deuxieme mot
    tfinal = a+b                # concaténations des caractèes
    f=open('testSortie.txt','w')  # ouverture de testSortie en écriture
    g.write(tfinal.lower())
f.close()                           # fermeture de testEntree
g.close()                           # fermeture de testSortie
print("Fichier testSortie.txt généré") # affichage final
```

Voici le résultat :



2. Mise en place de la base de données en SQL :

o Création de la Base de Donnée

- Création de la table proxy

Pour la mise en place de la base de données nous avons utilisé SQLDeveloper connecté au réseau Jolsio avec l'utilisateur « KLESOMMER »,

```
CREATE TABLE SALARIES (                                    --création de la table SALARIES
      NUM VARCHAR2(10) PRIMARY KEY,
      NOM VARCHAR2(20) NOT NULL,
      PRENOM VARCHAR2(20) NOT NULL,
      ADRESSEIP VARCHAR2(40)
      ,CONSTRAINT UNQ_IP_ADR UNIQUE (ADRESSEIP)    --contrainte donnée à la colonne
ADRESSEIP
      )
;
```

Figure 3: Création de la table SALARIES

```
CREATE SEQUENCE SEQ_ID1 START WITH 1 INCREMENT BY 1

INSERT INTO SALARIES (NUM, NOM, PRENOM, ADRESSEIP)
VALUES ('SEQ_ID.nextVal', 'DUPOND', 'Marie', '192.168.2.2');

INSERT INTO SALARIES (NUM, NOM, PRENOM, ADRESSEIP)
VALUES ('SEQ_ID.nextVal', 'DUBOIS', 'Paul', '192.168.2.1');

INSERT INTO SALARIES (NUM, NOM, PRENOM, ADRESSEIP)
VALUES ('SEQ_ID.nextVal', 'DURANT', 'Quentin', '192.168.2.3');

INSERT INTO SALARIES (NUM, NOM, PRENOM, ADRESSEIP)
VALUES ('SEQ_ID.nextVal', 'LEJAUNE', 'Laurence', '192.168.2.100');
```

Figure 4: Remplissage de la table SALARIES

```
CREATE TABLE PROXY (                              --création de la table SALARIES
      ID NUMBER(*, 0) PRIMARY KEY,
      ADRESSEIP VARCHAR2(40),
      JOURHEURE DATE NOT NULL,
      URL VARCHAR2(300) NOT NULL
      ,CONSTRAINT FK_ADR_IP FOREIGN KEY (ADRESSEIP) REFERENCES SALARIES
(ADRESSEIP)                                       --contrainte donnée à la colone ADRESSEIP
      )
;
```

Figure 3: Création de la table PROXY

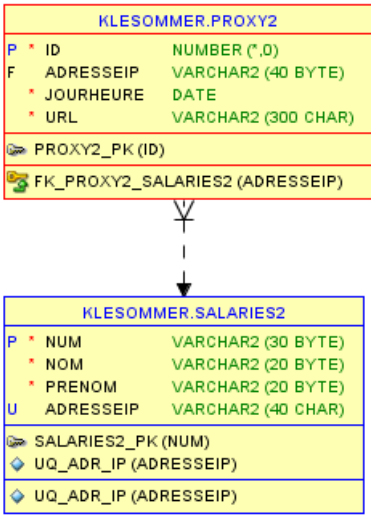


Figure 5: représentation de la base de données KLESOMMER

3. Analyse des fichiers textes de login :

Pour la suite du projet, il faut savoir interpréter le fichier log sur ce qu'il informe et comment il est constitué, chacune des données est séparées par un espace ou une tabulation « important pour la suite »

Les informations importantes sont, L'heure, L'adresse IP et l'URL

```
11:33:54        192.168.2.100 GET 200 http://openvpn.net/modules/mod_roknavmenu/themes/fusion/js/sfhover.js 375+1398 OK
11:33:54        192.168.2.100 GET 200 http://openvpn.net/modules/mod_rokajaxsearch/js/rokajaxsearch.js 377+24163 OK
11:33:55        192.168.2.100 GET 200 http://openvpn.net/media/system/js/mootools-more.js 379+238331 OK
11:33:55        192.168.2.100 GET 200 http://openvpn.net/images/openvpn_icon_navy_40.png 361+2158 OK
11:33:56        192.168.2.100 GET 200 http://www.google-analytics.com/__utm.gif? 391+35 OK
11:33:56        192.168.2.100 GET 200 http://openvpn.net/templates/telethra/img/bg-tab.png 359+207 OK
11:33:56        192.168.2.100 GET 200 http://openvpn.net/modules/mod_rokajaxsearch/themes/blue/search-icon.png 361+1370 OK
11:33:56        192.168.2.100 GET 200 http://openvpn.net/modules/mod_roknavmenu/themes/fusion/images/level3-parent.png 359+228 OK
11:33:56        192.168.2.100 GET 200 http://openvpn.net/modules/mod_roknavmenu/themes/fusion/images/level2-parent.png 359+228 OK
11:33:56        192.168.2.100 GET 200 http://openvpn.net/modules/mod_roknavmenu/themes/fusion/images/top-light.png 359+219 OK
11:33:56        192.168.2.100 GET 200 http://openvpn.net/templates/telethra/img/bg-content.png 360+753 OK
```

heures:minutes:secondes

Adresse ip du poste

Succès de la Requête

url de connexion

Quantité de données transférées en byte

requete réussie appartient au proxy

4. Réalisation d'une requête de validation du jeu d'essai :

Pour tester notre Base de données et aussi s'informer sur la suite des choses nous avons besoins de tester via une requête de test que voici :

```
CREATE SEQUENCE SEQ_ID2 START WITH 1 INCREMENT BY 1
INSERT INTO PROXY (ID, ADRESSEIP, JOURHEURE, URL)
VALUES ('SEQ_ID2.nextVal', '192.168.2.100', '11/23/11:30:37', 'http://www.freeradius.org/');
```

Figure 6: requête de test à supprimer après

5. Réalisation du programme Python de génération du script SQL INSERT

Création du programme Python, sur Repl-it

Pour résumer, au début nous devons importer les bibliothèques d'options python pour avoir des possibilités plus complexes, notamment le os qui permet de gérer tout ce qui est dans le système et dans la partie stockage.

Ensuite nous devons créer des fonctions pour après les appeler via le menu du programme.

Le principe de génération du fichier SQL se rapporte à lire le contenu du fichier log et d'en extraire les données voulues, soit L'adresse ip du poste qui s'est connecté, l'heure de connexion, l'adresse web vers laquelle l'adresse ip s'est connectée

1ère étape : Déclaration du Programme

```
1  # -*- coding: utf-8 -*-
2  """
3  date: 26/11/2020
4  comment: Projet Python 1SIOA
5  autor: LARMIER Maxime, LE SOMMER Kyllian
6  """
```

2ème étape : Import des Bibliothèques python

```
8  #librairies
9  import os
10 from os import chdir
11 from os import path
12 import enquiries
```

3ème étape : Définition des Variables Globales

```
14  #global variables
15  logfile_path = ""
16  project_path = ""
```

4ème étape : Fonction de lecture de fichier

```
20  def readfile():
21      #open the file selected \ ouverture du fichier selectinné
22      global logfile_path
23      global project_path
24      openlog = open(project_path+'/'+logfile_path, 'r')
25      print(openlog.read())
26      openlog.close()
27      return 0
```

5ème étape : Fonction de d'initialisation et de vérification de d'existence du répertoire

```
#path initialization \ initialisation du chemin d'accès
def ppath():
  project_path = input("enter project path (ex:/home/ex/project) or 'here' for current dir > ")
  if project_path == "here":
    project_path = os.getcwd()

  #path existence verification \ vérification du chemin d'accès
  while not os.path.exists(project_path):
    project_path = input("path doesn't exist, please reenter the path > ")

  chdir(project_path)
  print("\ncurrent path changed to: ", project_path,"\n")
  return project_path
```

6ème étape : Fonction d'entrée du nom du fichier

```
def lpath():
  logfile_path = input("enter log file name (log.txt) > ")

  #file existence verification \ vérification de l'existence du ficher
  while not os.path.exists(logfile_path):
    logfile_path = input("file doesn't exist, please reenter the file name > ")
  return logfile_path
```

7ème étape : Fonction de génération du fichier SQL

```
def gensqlfile():
  global project_path
  global logfile_path
  xlog = open(project_path+'/'+logfile_path, 'r')
  xdate = input('enter the date in the format:"yyyy-mm-dd" > ')
  ligne=xlog.readline()
  #les noms des fichiers de log sont 'log_proxy_'+date+'.txt'
  #fichiersql= #variable de nom fichier de fichier
  try:
    for ligne in xlog:
      #début insert
      ligne=ligne.lower()
      ligne=ligne.split(" ")
      fichiersql = open(project_path+'/'+'insert_'+xdate+'.sql', "a")
      fichiersql.write(" INSERT INTO PROXY(ID, ADRESSEIP, DATEHEURE, URL) VALUES (SEQ_ID2.nextVal, "+chr(39)+ligne[1]+chr(39)+", "+chr(39)+xdate+chr(39)+ ", "+chr(39)+ligne[4]+chr(39)+");\n")
      fichiersql.close() # close the logfile reading
  except Exception as error:
    print("\nerreur:\n")
    print(error)
    print("\n")
  xlog.close()
  if os.path.exists(project_path+"/insert_"+xdate+".sql"):
      print("\n fichier sql généré !")
  else:
      print("\n fichier sql n'as pas été généré !")


def line_count():
    global logfile_path
    global project_path
    try:
        file = open(project_path+"/"+logfile_path, 'r')
        line_number = len(file.readline())
        file.close()
        return line_number
    except:
        line_number = 0
        return line_number
```

8ème étape : le Menu

```python
#menu
def menu():
  global project_path
  global logfile_path
  os.system('clear')
  xline ="no"
  choice = ""
  while choice != "exit":
    if project_path != "?" and logfile_path != "?":
        xline = str(line_count())
    print("current directory: [", project_path, "]")
    print("selected file: [", logfile_path, "] ("+xline+" line)")
    options = ['select path', 'select file', 'read logfile', 'generate sql file', 'exit']
    choice = enquiries.choose('Choose one of these options: ', options)

    if choice == "select path":
      project_path = ppath()

    elif choice == "select file":
      logfile_path = lpath()

    elif choice == "read logfile":
      readfile()

    elif choice == "generate sql file":
      gensqlfile()

menu()
```

```
current directory: [ ? ]
selected file: [ ? ] (no line)
enter project path (ex:/home/ex/project) or 'here' for current dir > here

current path changed to:  /home/runner/entrytext-final

current directory: [ /home/runner/entrytext-final ]
selected file: [ ? ] (no line)
enter log file name (log.txt) > log_proxy_2020-11-23.txt
current directory: [ /home/runner/entrytext-final ]
selected file: [ log_proxy_2020-11-23.txt ] (70 line)
Choose one of these options:
> select path
  select file
  read logfile
  generate sql file
  exit
```

Figure 7: exemple de sélection de fichier

6. Exécution des scripts SQL d'insertion puis test des requêtes SQL

7. Rédaction d'un mode opératoire

Mode opératoire pour le fonctionnement du Script Python

- Le format du fichier de log (structure)

hh:mm:ss adresseip GET 301 url 195+185 OK

```
11:30:39        192.168.2.100 GET 200 http://www.google-analytics.com/__utm.gif? 391+35 OK
11:30:43        192.168.2.100 GET 200 http://freeradius.org/download.html 135+7197 OK
11:30:43        192.168.2.100 GET 200 http://www.google-analytics.com/__utm.gif? 391+35 OK
11:30:47        192.168.2.100 GET 200 http://wiki.freeradius.org/guide/faq 163+72328 OK
11:30:47        192.168.2.100 GET 302 http://wiki.freeradius.org/custom.css 218+0 OK
```

Figure 8: en voici un exemple

- L'utilisation du programme Python

Quand nous lançons le programme, nous avons un menu qui nous permet par les flèches du clavier d'y naviguer

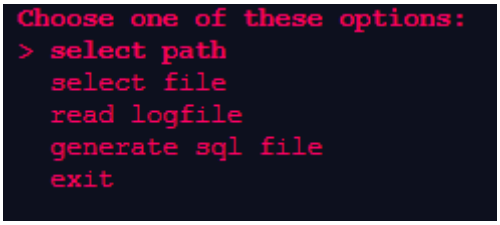
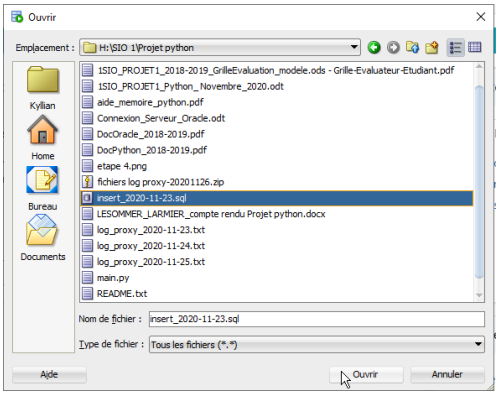


Figure 9: Menu du Programme

Afin de générer notre fichier sql nous allons en premier sélectionner le répertoire du projet en choisissant l'option "select path" du menu, une fois cette option sélectionnée il nous suffiras d'entrer le chemin du dossier (tapez "here" pour remplir sélectionner automatiquement le dossier actuel, et si le dossier n'existe pas il vous demandera de le renseigner à nouveau). Une fois le chemin sélectionné nous pouvons maintenant sélectionner le fichier log en choisissant l'option "select file". (renseignez le nom du fichier présent dans le dossier précédemment sélectionné). Vous pourrez constatez les informations entrées précédemment au dessus du menu, ainsi que le nombre de ligne que comporte le fichier. Si vous souhaitez vérifier le contenue du fichier sélectionné, utilisez l'option "read logfile" qui affichera en brut le contenue du fichier. Pour générer le fichier sql veuillez choisir l'option "generate sql file" et renseignez la date que vous souhaitez. Le fichier est maintenant généré ! Il ne vous reste plus qu'a l'importer dans votre base de données.

- L'utilisation des scripts SQL générés

Pour l'utilisation des fichiers SQL nous devons les ouvrir via SQLDeveloper,



Projet Python

Figure 10: ouverture du fichier insert

Puis pour le lancement de la requête nous devons cliquer sur le bouton ‘exécuter le script’

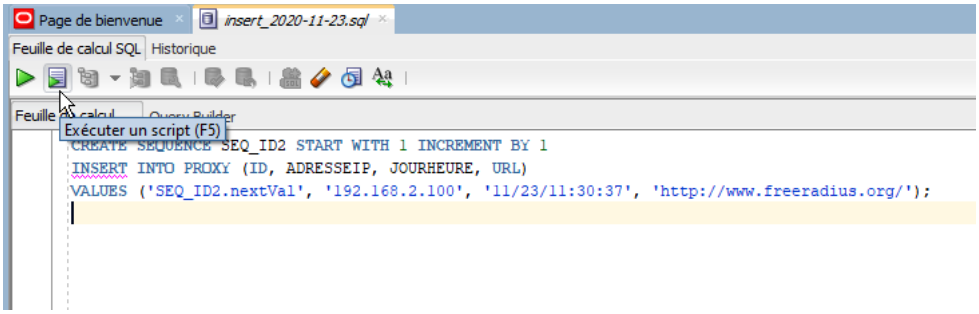
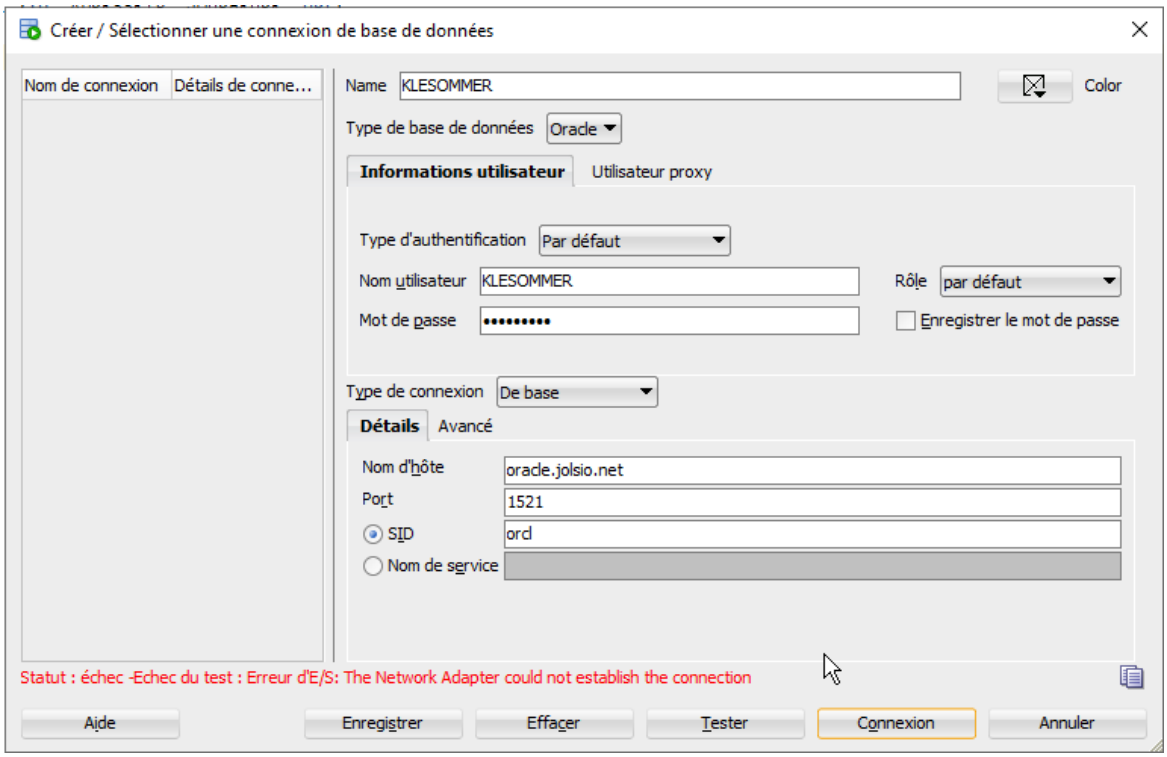


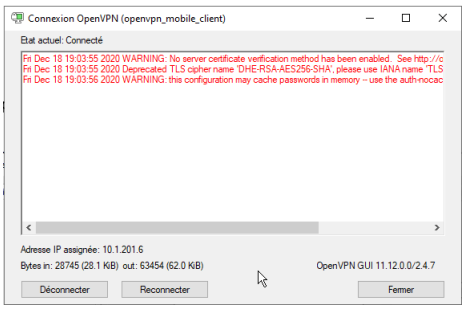
Figure 11: Lancement de la requête

- La procédure de test des requêtes de vérification des jeux d’essai

Incomplet, pas d’accès à la base de données



Maxime LARMIER et Kyllian LE SOMMER de 1SIOA groupe A



Haut de Document

Projet Python