

2022/2023	AP – Projet Python (Groupe 27)
BTS SIO	Auteur : Mbembe Enoc, Kerlau Pierre
1SIOB	Date de rédaction : 03 Janvier 2023

# Compte-rendu de TP

## Etape 1 – Lecture et écriture dans un fichier en Python

Pour commencer il faut créer un premier fichier nommé *TestEntree.txt* dans lequel nous allons mettre les noms des utilisateurs, puis nous allons créer un fichier *TestSortie.txt* dans lequel nous allons recevoir les résultats du premier fichier qui passe par notre programme

Notre programme contient :

```
from os import chdir
chdir("/Users/pierr/Documents/BTS SIO/AP/Python")
x = open('TestEntree.txt','r') # Ouverture du fichier TestEntrée
y = open('TestSortie.txt','w') # Ouverture du fichier TestSortie

for line in x: # Lecture de chaque ligne du fichier TestEntrée
    t = line.split(';') # Prend les mots de la ligne et les ajoute dans un tableau
    mot = (t[0] [0]) + t[1] # Prend la première lettre du premier mot et le deuxième mot
    mot = mot.lower() # Met le résultat en minuscule
    y.write(str(mot) + '\n') # Ecris le résultat dans le fichier TestSortie
x.close()
y.close()
```

Nous obtenons dans le fichier *TestSortie* :



TestSortie - Bloc-notes

Fichier Edition Format Affichage Aide

ezola

vhugo

gsand

<b>Projet python</b>	KERLEAU PIERRE MBEMBE ENOC
<b>Analyse des logs</b>	<b>Etape 02</b>

Dans cette étape nous devons créer une base de données **dbstesiology** avec pour jeu de caractère **utf8** et interclassement **utf8\_unicode\_ci** pour ensuite créer un utilisateur et lui donner tous les droits cette base de donnée uniquement.

Requêtes SQL:

```
CREATE DATABASE dbstesiology
DEFAULT CHARACTER SET utf8
COLLATE utf8_unicode_ci
```

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0.0001 seconde(s).)

```
CREATE DATABASE dbstesiology DEFAULT CHARACTER SET utf8 COLLATE utf8_unicode_ci;
```

[ Éditer en ligne ] [ Éditer ] [ Créer le code source PHP ]

⚠ Error: #1046 Aucune base n'a été sélectionnée

**SHOW DATABASES;**

La requête SQL a été exécutée avec succès.

```
SHOW DATABASES;
```

☐ Profilage [ Éditer en ligne ] [ Éditer ] [ Créer le code source PHP ] [ Actualiser ]

Options supplémentaires

**Database**

dbstesiology

dbwordpress

information\_schema

mysql

performance\_schema

Requête création d'utilisateur **ADMINSTESIO**

```
CREATE USER 'adminstesio' @'localhost'
IDENTIFIED BY 'joliverie';
```

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0.0036 seconde(s).)

```
CREATE USER 'adminstesio'@'localhost' IDENTIFIED BY 'joliverie';
```

[ [Éditer en ligne](#) ] [ [Éditer](#) ] [ [Créer le code source PHP](#) ]

Afficher l'utilisateur crée

**SELECT \* FROM mysql.user**

✓ Affichage des lignes 0 - 5 (total de 6, traitement en 0.0005 seconde(s).)

```
SELECT user FROM mysql.user;
```

☐ Profilage [ [Éditer en ligne](#) ] [ [Éditer](#) ] [ [Expliquer SQL](#) ] [ [Créer le code source PHP](#) ] [ [Actualiser](#) ]

☐ Tout afficher | Nombre de lignes :  | Filtrer les lignes:

Options supplémentaires

User

bmorane

adminstesio

mariadb.sys

mysql

root

Requête donnant l'utilisateur tous les droits uniquement sur la base **dbstesiology**

**GRANT ALL**

**ON dbstesiology.\***

**TO 'adminstesio' @'localhost';**

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0.0007 seconde(s).)

```
GRANT ALL ON dbstesiology.* TO 'adminstesio'@'localhost';
```

[ [Éditer en ligne](#) ] [ [Éditer](#) ] [ [Créer le code source PHP](#) ]

Requête création des tables **SALARIES** et **PROXY** avec les contrainte et les contenu de tables

**CREATE INTO SALARIES (**

Id INT PRIMARY KEY, `nom` VARCHAR (15),

`prenom` VARCHAR (15), `adresseip` text

);

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0.0083 seconde(s).)

```
CREATE TABLE `SALARIES` ( `nom` char(64) NOT NULL, `prenom` char(64) NOT NULL, `adresseip` text NOT NULL);
```

[ [Éditer en ligne](#) ] [ [Éditer](#) ] [ [Créer le code source PHP](#) ]

```
CREATE INTO PROXY (  
Id INT PRIMARY KEY, `adresseip` VARCHAR (15),  
`jourheure` VARCHAR (15), `URL` text  
);
```

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0.0070 seconde(s).)

```
CREATE TABLE PROXY ( Id INT PRIMARY KEY, `adresseip` VARCHAR (15), `jourheure` VARCHAR (15), `URL` text );
```

[\[ Éditer en ligne \]](#) [\[ Éditer \]](#) [\[ Créer le code source PHP \]](#)

Requête pour un enregistrement dans la table **SALARIES**  
**INSERT INTO** SALARIES (NOM, PRENOM, ADRESSEIP)  
**VALUES** ('DUFONT', 'Marie', '192.168.2.2' );

✓ 1 ligne insérée. (traitement en 0.0010 seconde(s).)

```
INSERT INTO SALARIES(NOM, PRENOM, ADRESSEIP) VALUES ('DUFONT', 'Marie', '192.168.2.2');
```

[\[ Éditer en ligne \]](#) [\[ Éditer \]](#) [\[ Créer le code source PHP \]](#)

<b>2022/2023</b>	<b>AP – Projet Python (Groupe 27)</b>
<b>BTS SIO</b>	Auteur : Mbembe Enoc, Kerlau Pierre
<b>1SIOB</b>	Date de rédaction : 03 Janvier 2023

# Compte-rendu de TP

## Etape 3 – Analyse des fichiers textes de login

Pour cette étape nous devons analyser les fichiers log\_proxy\_yyyy\_mm\_dd.txt, nous noterons les informations disponibles sur chaque ligne et à quels champs de la base de données ils peuvent correspondre.

```
11:30:37 192.168.2.100 GET 301 http://www.freeradius.org/ 195+185 OK
11:30:38 192.168.2.100 GET 200 http://download.cdn.mozilla.net/pub/firefox/releases/26.0/update/win32/fr/firefox-24.0-26.0.partial.mar
460+8166778 OK
11:30:38 192.168.2.100 GET 200 http://freeradius.org/ 135+5458 OK
11:30:38 192.168.2.100 GET 204 http://pagead2.googlesyndication.com/activeview? 469+0 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
11:30:47 192.168.2.100 GET 200 http://wiki.freeradius.org/create/custom.css 162+2306 OK
11:30:51 192.168.2.100 GET 302 http://wiki.freeradius.org/ 205+0 OK
11:30:52 192.168.2.100 GET 200 http://wiki.freeradius.org/Home 163+14470 OK
11:30:52 192.168.2.100 GET 302 http://wiki.freeradius.org/custom.css 218+0 OK
11:30:52 192.168.2.100 GET 200 http://wiki.freeradius.org/create/custom.css 162+2306 OK
11:30:56 192.168.2.100 GET 200 http://freeradius.org/doc/ 135+11761 OK
11:30:56 192.168.2.100 GET 200 http://www.google-analytics.com/_utm.gif? 391+35 OK
11:31:15 192.168.2.100 GET 302 http://www.google.fr/ 304+219 OK
11:32:09 192.168.2.100 GET 200 http://www.google.fr/url? 365+249 OK
11:32:10 192.168.2.100 GET 200 http://openvpn.se/ 259+2482 OK
11:32:10 192.168.2.100 GET 200 http://openvpn.se/standard.css 335+820 OK
11:32:11 192.168.2.100 GET 304 http://pagead2.googlesyndication.com/pagead/expansion_embed.js 213+0 OK
11:32:12 192.168.2.100 GET 200 http://googleads.g.doubleclick.net/pagead/ads? 463+13992 OK
11:32:12 192.168.2.100 GET 200 http://googleads.g.doubleclick.net/pagead/ads? 463+15455 OK
11:32:12 192.168.2.100 GET 304 http://pagead2.googlesyndication.com/pagead/images/nessie_icon_chevron_white.png 214+0 OK
11:32:13 192.168.2.100 GET 302 http://www.google.com/pagead/drt/ui 417+304 OK
11:32:14 192.168.2.100 GET 200 http://openvpn.se/favicon.ico 293+3262 OK
11:32:14 192.168.2.100 GET 302 http://www.google.com/pagead/drt/ui 417+304 OK
11:32:15 192.168.2.100 GET 200 http://openvpn.se/documentation.html 258+1174 OK
```

Nous pouvons trouver dans ces fichiers (ici le fichier : log\_proxy\_2022\_11\_23) plusieurs informations, de la gauche vers la droite nous avons :

- L'heure de la requête
- L'adresse IP du poste dans lequel la requête a été effectué
- GET correspond à la commande qui a été faite pour la quête
- Les chiffres (200,302,304...) correspond à un code HTTP
  - 200 : succès de la requête
  - 301 et 302 : redirection, respectivement permanente et temporaire
  - 401 : utilisateur non authentifié
  - 403 : accès refusé
  - 404 : ressource non trouvée
  - 500, 502 et 503 : erreurs serveur
  - 504 : le serveur n'a pas répondu.
- L'adresse du serveur vers lequel la requête a été envoyé
- XXX+XXX correspond à la quantité de données qui ont été transférés
- OK correspond au fait que la requête a été effectué sans problème

Nous pouvons alors mettre ces informations dans un tableau avec :

- La date
- L'heure
- L'adresse IP
- Le numéro du poste
- Le code HTTP
- L'adresse du serveur vers lequel la requête a été envoyé
- La quantité de données qui ont été transférés

2022/2023	AP – Projet Python (Groupe 27)
BTS SIO	Auteur : Mbembe Enoc, Kerlau Pierre
1SIOB	Date de rédaction : 03 Janvier 2023

# Compte-rendu de TP

## Etape 5 – Réalisation d'un programme Python récupérant les informations de log

Pour pouvoir récupérer les informations de log d'un fichier texte nous devons créer un programme qui va ouvrir un premier fichier texte, analyser les informations et garder les données qui nous sont utiles

```
from os import chdir
chdir("/Users/pierr/Documents/BTS SIO/AP/Python/ProjetPython")

date = str(input("Veuillez rentrer la date en faisant : yyyy-mm-dd : ")) # Demande
la date des logs
fichier = "log_proxy_"+date+".txt" # Trouve
le fichier avec la date
f = open(fichier, 'r') # Ouvre
ture du fichier
sortie = open("loginfo_"+date+".csv", 'w') # Ouvre
le fichier loginfo.csv

lignes = f.readlines() # Lecture
du fichier log_proxy lignes par lignes
nbLignes = len(lignes)

for i in range(nbLignes): # Boucle
avec le nombre de lignes du premier fichier
    lignesseparees=lignes[i].split(" ") # Split
les informations a chaque fois qu'il y a un espace
    heure=lignesseparees[0] # Assigne
la valeur du split 0 a la variable heure
    ip=lignesseparees[1] # Assigne
la valeur du split 1 a la variable ip
    url=lignesseparees[4] # Assigne
la valeur du split 4 a la variable url
    resultat= date +";"+heure+";"+ip+";"+url+"\n" # Donne
le résultat de toutes les valeurs lignes par lignes
    sortie.write(resultat) # Ecris
le résultat dans le fichier loginfo
f.close() # Ferme-
ture du fichier Log_Proxy
sortie.close()

print("Les informations ont été ajoutés au programme loginfo_"+date+".csv")
```

```
BTS SIO/AP/Python/ProjetPython/Etape5Projet.py"
Veuillez rentrer la date en faisant : yyyy-mm-dd : 2022-11-24
Les informations ont été ajoutés au programme loginfo_2022-11-24.csv
PS C:\Users\pierr\Documents\BTS SIO\AP\Python\ProjetPython> █
```

Résultat du programme dans le terminale

<b>2022/2023</b>	<b>AP – Projet Python (Groupe 27)</b>
<b>BTS SIO</b>	Auteur : Mbembe Enoc, Kerlau Pierre
<b>1SIOB</b>	Date de rédaction : 03 Janvier 2023

Cela nous donne un fichier nommé *loginfo\_2022-11-24.cvs* dans lequel nous avons :

	A	B	C	D	E	F	G	H	I	J	K
1	24/11/2022	07:02:07	192.168.2.1	http://www.lequipe.fr/Quotidien/v2/img/arrow_offre.png							
2	24/11/2022	07:02:07	192.168.2.1	http://static.lequipe.fr/Quotidien/une/93/une30102013.jpg							
3	24/11/2022	07:02:07	192.168.2.1	http://www.lequipe.fr/Quotidien/v2/img/10article.jpg							
4	24/11/2022	07:02:07	192.168.2.1	http://www.lequipe.fr/Quotidien/v2/img/sprite.png							
5	24/11/2022	07:02:07	192.168.2.1	http://static.lequipe.fr/Quotidien/une/93/une05112013.jpg							
6	24/11/2022	07:02:07	192.168.2.1	http://www.lequipe.fr/Quotidien/v2/img/bg-papier.jpg							
7	24/11/2022	07:02:07	192.168.2.1	http://www.lequipe.fr/Quotidien/v2/img/premium.jpg							
8	24/11/2022	07:02:07	192.168.2.1	http://www.lequipe.fr/Quotidien/v2/img/visuel2.jpg							
9	24/11/2022	07:02:08	192.168.2.1	http://www.lequipe.fr/club/compte/ajax/quoti.php							
10	24/11/2022	07:02:08	192.168.2.1	http://mmtro.com/p?							
11	24/11/2022	07:02:08	192.168.2.1	http://r.turn.com/r/beacon?							
12	24/11/2022	07:02:09	192.168.2.1	http://d2yp9b3a29g3i2.cloudfront.net/ergotest/8e296a067a.js							
13	24/11/2022	07:02:09	192.168.2.1	http://mmtro.com/seg/6546622.js							
14	24/11/2022	07:02:09	192.168.2.1	http://mmtro.com/sync.js							
15	24/11/2022	07:02:10	192.168.2.1	http://static.content-square.net/mousetest/lequipe/mousetest.js							
16	24/11/2022	07:02:11	192.168.2.1	http://ib.adnxs.com/seg?							
17	24/11/2022	07:02:11	192.168.2.1	http://ib.adnxs.com/getuid?							
18	24/11/2022	07:02:11	192.168.2.1	http://at.alenty.com/trk/1?							
19	24/11/2022	07:02:11	192.168.2.1	http://mmtro.com/s?							
20	24/11/2022	07:02:11	192.168.2.1	http://mmtro.com/s?							
21	24/11/2022	07:02:23	192.168.2.10	http://www.cnet.com/							
22	24/11/2022	07:02:25	192.168.2.10	http://asset1.cbsistatic.com/cnwk.1d/cb1383771027000/html/rb/js/tron/Build/Filelists/2000/2000.1.0.js							
23	24/11/2022	07:02:25	192.168.2.10	http://asset3.cbsistatic.com/cnwk.1d/cb1383773313000/css/rb/Build/global/site1.css							
24	24/11/2022	07:02:25	192.168.2.10	http://i.i.cbsi.com/cnwk.1d/css/rb/Build/global/ie8.css							
25	24/11/2022	07:02:25	192.168.2.10	http://asset1.cbsistatic.com/cnwk.1d/cb1383773315000/css/rb/Build/2000/2000.1.1.css?							
26	24/11/2022	07:02:25	192.168.2.10	http://dw.cbsi.com/js/dw.js							
27	24/11/2022	07:02:25	192.168.2.10	http://asset2.cbsistatic.com/cnwk.1d/cb1383773313000/css/rb/Build/global/base.css?							
28	24/11/2022	07:02:26	192.168.2.10	http://asset1.cbsistatic.com/cnwk.1d/Adc/common/manta/adFunctionsD.cnet.ie							

2022/2023	AP – Projet Python (Groupe 27)
BTS SIO	Auteur : Mbembe Enoc, Kerlau Pierre
1SIOB	Date de rédaction : 03 Janvier 2023

# Compte-rendu de TP

## Etape 6 – Réalisation du programme Python de généralisation du script SQL INSERT

Dans cette étape nous devons créer un programme qui permet de créer des requêtes SQL dans un dossier sortant *insert\_date.sql* à l'aide des données contenu dans le dossier entrant *log\_proxy\_date.txt*

```
from os import chdir
chdir("/Users/pierr/Documents/BTS SIO/AP/Python/ProjetPython")

date = str(input("Veuillez rentrer la date en faisant : yyyy-mm-dd : ")) #
Demande la date des logs #
fichier = "log_proxy_"+date+".txt" #
Trouve le fichier avec la date #
f = open(fichier, 'r') #
Ouverture du fichier #
sortie = open("insert_"+date+".sql", 'w') #
Ouvre le fichier insert_date.sql

lignes = f.readlines() #
Lecture du fichier log_proxy lignes par lignes
nbLignes = len(lignes)

for i in range(nbLignes): #
    lignesseparees=lignes[i].split(" ") #
    Split les informations a chaque fois qu'il y a un espace #
    heure=lignesseparees[0] #
    Assigne la valeur du split 0 a la variable heure #
    ip=lignesseparees[1] #
    Assigne la valeur du split 1 a la variable ip #
    url=lignesseparees[4] #
    Assigne la valeur du split 4 a la variable url #
    resultat= date +";"+heure+";"+ip+";"+url+"\n" #
    Donne le résultat de toutes les valeurs lignes par lignes
    sortie.write("INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)\nVA-
LUES("+str(nbLignes)+", "+ip+", "+date+", "+heure+", "+url+";"+")'\n') # Ecris le
résultat dans le fichier instert_date.sql
f.close() #
Fermeture du fichier Log_Proxy
sortie.close()

print("Les informations ont été ajoutés au programme log_proxy_"+date+".sql")
```



2022/2023	AP – Projet Python (Groupe 27)
BTS SIO	Auteur : Mbembe Enoc, Kerlau Pierre
1SIOB	Date de rédaction : 03 Janvier 2023

```
BTS SIO/AP/Python/ProjetPython/Etape6Projet.py"
Veuillez rentrer la date en faisant : yyyy-mm-dd : 2022-11-24
Les informations ont été ajoutés au programme log_proxy_2022-11-24.sql
PS C:\Users\pierr\Documents\BTS SIO\AP\Python\ProjetPython> s
```

Résultat du programme dans le terminal

```
insert_2022-11-23.sql
1  INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
2  VALUES(87, 192.168.2.100, 2022-11-23, 11:30:37, http://www.freeradius.org/);
3  INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
4  VALUES(87, 192.168.2.100, 2022-11-23, 11:30:38, http://download.cdn.mozilla.net/pub/firefox/releases/26.0/update/win32/fr/firefox-24.0-1
5  INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
6  VALUES(87, 192.168.2.100, 2022-11-23, 11:30:38, http://freeradius.org/);
7  INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
8  VALUES(87, 192.168.2.100, 2022-11-23, 11:30:38, http://pagead2.googlesyndication.com/activeview?);
9  INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
10 VALUES(87, 192.168.2.100, 2022-11-23, 11:30:39, http://www.google-analytics.com/_utm.gif?);
11 INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
12 VALUES(87, 192.168.2.100, 2022-11-23, 11:30:43, http://freeradius.org/download.html);
13 INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
14 VALUES(87, 192.168.2.100, 2022-11-23, 11:30:43, http://www.google-analytics.com/_utm.gif?);
15 INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
16 VALUES(87, 192.168.2.100, 2022-11-23, 11:30:47, http://wiki.freeradius.org/guide/faq);
17 INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
18 VALUES(87, 192.168.2.100, 2022-11-23, 11:30:47, http://wiki.freeradius.org/custom.css);
19 INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
20 VALUES(87, 192.168.2.100, 2022-11-23, 11:30:47, http://wiki.freeradius.org/create/custom.css);
21 INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
22 VALUES(87, 192.168.2.100, 2022-11-23, 11:30:51, http://wiki.freeradius.org/);
23 INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
24 VALUES(87, 192.168.2.100, 2022-11-23, 11:30:52, http://wiki.freeradius.org/Home);
25 INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
26 VALUES(87, 192.168.2.100, 2022-11-23, 11:30:52, http://wiki.freeradius.org/custom.css);
27 INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
28 VALUES(87, 192.168.2.100, 2022-11-23, 11:30:52, http://wiki.freeradius.org/create/custom.css);
29 INSERT INTO TABLE proxy (ID, Adresse IP, Jour, Heure, URL)
30 VALUES(87, 192.168.2.100, 2022-11-23, 11:30:56, http://freeradius.org/doc/);
```

Résultat du programme dans le document crée insert\_2022-11-24.sql

2022/2023	AP – Projet Python (Groupe 27)
BTS SIO 1SIOB	Auteur : Mbembe Enoc, Kerlau Pierre
	Date de rédaction : 03 Janvier 2023

# Compte-rendu de TP

Binôme 27, embembe\_SRVWEB\_T\_Rocky9\_mariadb\_apache\_phpmyadmin\_1B\_V2, 10.15.117.170

## Etape 8 – Rédaction d'un mode opératoire

### Introduction

Dans cette étape, nous allons rédiger un mode opératoire à l'intention d'un administrateur système.

### Mode Opératoire

1 – Télécharger les fichiers de log (Ils sont pour le moment sous le format .txt)

2 – Télécharger et lancer le programme de l'étape 6. Il a pour objectif de prendre les informations importantes d'un fichier le log et créer automatiquement un fichier .sql qui va être implanté dans notre base de donnée

3 – Importer le fichier .sql crée dans votre base de donnée

4 – Pour être sûr que l'utilisateur ne se trompe pas dans sa requête, il y a dans le programme une procédure de test des requêtes qui permet de vérifier si le fichier demandé existe, ou s'il n'est pas vide.

### Modification du programme pour ajouter une vérification des fichiers demandés

```
while (os.path.isfile('log_proxy_'+date+'.txt') == False) or (os.path.get-  
size('log_proxy_'+date+'.txt') == 0):  
    date = str(input("Le fichier n'existe pas ou est vide. Veuillez entrer une date corres-  
pondant à un fichier valide: "))
```

*Partie du programme qui vérifie l'existence d'un fichier demandé*