Joindre 2 fichiers CSV par l'intermédiaire d'un troisième contenant les relations entre les 2 premiers

Le problème

- Nous voulons faire en sorte de joindre les données contenues dans deux fichiers CSV représentant par exemple un dump d'une table. Ces fichiers disposent d'une colonne contenant un identifiant unique pour chaque ligne/enregistrement.
- Il y a enfin un troisième fichier CSV contenant lui au moins deux colonnes qui référencent les identifiants des enregistrements à la fois dans le premier tableau-CSV et le deuxième tableau-CSV déjà mentionnés.
- Le but est de consolider des données des deux côtés pour produire des traitements utilisant les informations provenant des deux tableaux à la fois (pas seulement l'identifiant). C'est, en quelque sorte, une jointure SQL mais sous forme de fichiers-tabeaux-CSV.

Une solution

Voici donc une proposition de code *python* qui permet de faire cette jonction.

join_files.py

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

# si python < 3
from __future__ import print_function

usage = """
[usage] %s --table1 <fichier tableau CSV1> --table2 <fichier tableau CSV2> --jointable <fichier jointure
tableau CSV> --sep <séparateur de champs> --format <format affichage des données à afficher avec %s pour chaque
colonne à afficher> --colonnes <couples numéro de fichier (1 ou 2) «.» numéros de colonnes pour chaque %s du
format donné précédemment ; séparés par des «,»>

Ce script a pour but de faire une jointure entre 3 fichiers :

- Un permier contenant une liste (genre CSV) séparée par un séparateur
(par défaut «|») avec plusieurs champs dont le premier est
l'identifiant unique (ou alors un autre numéro de colonne considérée
comme contenant l'identifiant).

- Un deuxième contenant une liste (genre CSV) séparée par un
séparateur (par défaut «|») avec plusieurs autres champs et entrées
dont le premier (par défaut) est l'identifiant unique. On peut
également donner un numéro de colonne pour cet identifiant.

- un troisième fichier contenant au moins deux colonnes avec, dans le
premier champ (séparateur = «|»), une liste d'identifiants
correspondants au premier fichier (on peut aussi donner un numéro de
colonne si ce n'est pas la première – valeur par défaut).et le
deuxième champ correspondant à l'identifiant du deuxième fichier (avec
un séparateur = «|» par défaut) et un numéro de colonne = 2 par défaut
mais qui peut être précisé.

Les numéros de colonnes commencent à 1.

"""


import sys
import os
from optparse import OptionParser

# traitement des options

parser = OptionParser(usage)
parser.add_option("--table1", dest="ftable1",
                  help="Tableau CSV 1")
parser.add_option("--colonne_id1", dest="ct1", default=1, type="int",
```

```
                  help="Numero de colonne de l'identifiant du premier tableau CSV")
parser.add_option("--table2", dest="ftable2",
                  help="Tableau CSV 2")
parser.add_option("--colonne_id2", dest="ct2", default=1, type="int",
                  help="Numero de colonne de l'identifiant du deuxieme tableau CSV")
parser.add_option("--jointable", dest="fjointable",
                  help="Tableau CSV contenant les identifiants uniques des 2 tableaux CSV 1 et 2")
parser.add_option("--colonne_jid1", dest="cj1", default=1, type="int",
                  help="Numero de colonne de l'identifiant du premier tableau CSV dans le fichier de jointure
(le troisieme fichier)")
parser.add_option("--colonne_jid2", dest="cj2", default=2, type="int",
                  help="Numero de colonne de l'identifiant du deuxieme tableau CSV dans le fichier de jointure
(le troisieme fichier)")
parser.add_option("--sep", dest="sep", default="|",
                  help="Separateur utilise pour tous les fichiers-tableaux CSV - un seul caractere attendu -
valeur par defaut le pipe")
#parser.add_option("--sep1", dest="sep1",
#                  help="Separateur utilise pour le fichiers-tableaux CSV 1 - un seul caractere attendu - pas
de valeur par defaut")
#parser.add_option("--sep2", dest="sep2",
#                  help="Separateur utilise pour le fichiers-tableaux CSV 2 - un seul caractere attendu - pas
de valeur par defaut")
#parser.add_option("--sepj12", dest="sepj12",
#                  help="Separateur utilise pour le fichiers-tableaux CSV de jointure - un seul caractere
attendu - pas de valeur par defaut")
parser.add_option("--format", dest="format",
                  help="Format avec des %s le ou seront les donnees reprises des colonnes des deux tableaux
CSV")
parser.add_option("--colonnes", dest="colonnes",
                  help="Couples numero de fichier (1 ou 2) '.' numeros de colonnes pour chaque %s du format
donne precedemment; separes par des ','")
(options, args) = parser.parse_args()
#if len(args) <= 6:
#    parser.error("Incorrect number of arguments")
#    parser.error(usage % sys.argv[0])
#    sys.exit(2)

t1 = {} # tableau de hashage contenant les enregistrements du premier fichier-tableau avec la colonne
identifiant unique comme clé
t2 = {} # tableau de hashage contenant les enregistrements du deuxième fichier-tableau avec la colonne
identifiant unique comme clé

sep = options.sep # le séparateur utilisé
ct1 = options.ct1 - 1 # le numéro de colonne contenant l'identifiant unique du premier tableau du premier
fichier
ct2 = options.ct2 - 1 # le numéro de colonne contenant l'identifiant unique du second tableau du deuxième
fichier
cj1 = options.cj1 - 1 # le numéro de colonne contenant l'identifiant unique du premier tableau dans le
troisième fichier, fichier de jonction entre les deux premiers
cj2 = options.cj2 - 1 # le numéro de colonne contenant l'identifiant unique du second tableau dans le troisième
fichier, fichier de jonction entre les deux premiers

with open(options.ftable1, 'r') as table1:
    for ligne in table1:
        ligne = ligne.rstrip()
        tl1 = ligne.split(sep)
        if len(tl1) <= ct1:
            print("[Erreur] Le numéro de colonne de l'identifiant du tableau CSV 1 dépasse le nombre de colonne
de la ligne: %d pour %d max colonnes" % (ct1, len(tl1)), file=sys.stderr)
            continue
        id1 = tl1[ct1]
        t1[id1] = tl1

with open(options.ftable2, 'r') as table2:
    for ligne in table2:
        ligne = ligne.rstrip()
        tl2 = ligne.split(sep)
        if len(tl2) <= ct2:
            print("[Erreur] Le numéro de colonne de l'identifiant du tableau CSV 2 dépasse le nombre de colonne
de la ligne: %d pour %d max colonnes" % (ct2, len(tl2)), file=sys.stderr)
            continue
```

```
        id2 = tl2[ct2]
        t2[id2] = tl2

with open(options.fjointable, 'r') as jointure_t12:
    for ligne in jointure_t12:
        ligne = ligne.rstrip()
        jt12 = ligne.split(sep)
        if len(jt12) <= cj1 or len(jt12) <= cj2:
            print("[Erreur] Le numéro de colonne de l'identifiant du tableau de jointure dépasse le nombre de
colonne de la ligne: %d (identifiant tableau1) - %d (identifiant tableau2) - pour %d max colonnes" % (cj1, cj2,
len(jt12)), file=sys.stderr)
            continue
        jid1 = jt12[cj1]
        jid2 = jt12[cj2]
        try:
            tableau_a_afficher = []
            for i in options.colonnes.split(","):
                (filenumber, colnumber) = i.split(".")
                if int(filenumber) == 1:
                    # on enlève 1 au numéro de colonne car ce numéro commence à 1 contrairement aux tableaux
python
                    tableau_a_afficher.append(t1[jid1][int(colnumber) - 1])
                elif int(filenumber) == 2:
                    tableau_a_afficher.append(t2[jid2][int(colnumber) - 1])
            # pour que le format d'affichage (tableau_a_afficher) soit interprété correctement par print,
            # il faut que ce soit un tuble et nom un tableau.
            print(options.format % tuple(tableau_a_afficher))
        except KeyError as k:
            print("Erreur de clé pour jid1 %s- jid2 %s" % (jid1, jid2), file=sys.stderr)
            continue
```

Exemple d'utilisation

Utilisation

```
# dump de la base de donnée centreon sous forme d'un ensemble de fichiers sql contenant la description sql de
la création de ces tables
# et de fichiers txt contenant le dump des données de ces tables sous un format CSV (séparateur = «|»)
mkdir -p /tmp/Sauvegardes/centreon
mysqldump -u root --quick centreon --tab=/tmp/Sauvegardes/centreon --fields-terminated-by="|" -c

# insertion des contactgroups
# il faut avoir installé les clapi de centreon pour que ça fonctionne
cd /tmp/Sauvegardes/centreon
cut -d"|" -f 2,3 contactgroup.txt \
  | perl -e 'while(<>) { chomp; ($name, $alias) = split(/\|/); print "centreon -u admin -p ************ -o CG -
a add -v \"$name;$alias\"\n"; }' | bash

# et maintenant, la partie qui nous intéresse avec l'utilisation de join_files.py
/usr/local/bin/join_files.py --table1 /tmp/Sauvegardes/centreon/contact.txt \
                             --table2 /tmp/Sauvegardes/centreon/contactgroup.txt \
                             --jointable /tmp/Sauvegardes/centreon/contactgroup_contact_relation.txt \
                             --sep "|" \
                             --colonne_id1 1 \
                             --colonne_id2 1 \
                             --colonne_jid1 2 \
                             --colonne_jid2 3 \
                             --format 'centreon -u admin -p ************ -o CG -a addcontact -v "%s;%s"' \
                             --colonnes '2.2,1.4'
# car la colonne 2 du fichier des contactgroups contient le nom du contactgroup qu'il faut alimenter par le nom
du contact donné dans la 4ième colonne de «contact.txt».
# le tableau CSV de jointure (contactgroup_contact_relation.txt) contient 3 champs dont
# le 2ième contient l'identifiant du tableau des contacts et le 3ième qui contient l'identifiant du tableau des
contactgroups.
```

Référence

- https://documentation.centreon.com/docs/centreon-clapi/en/latest/objects/contact_groups.html

Remarques

- Il existe, sous unix, la commande *join* qui permet de faire des jointures entre fichiers préalablement triés. Cependant, la jonction ne se fait qu'entre deux fichiers. Il serait donc nécessaire, si on voulait utiliser une solution à base de *join* de faire du *pipe* entre deux de ces commandes.

Exemple avec join

```
cd /tmp/Sauvegardes/centreon

# le fichier contactgroup_contact_relation.txt contient 3 champs
# - le premier est un nombre propre à cette table
# - le deuxième est l'identifiant de contact
# - le troisième est l'identifiant de contactgroup
# Pour le premier tri, on fait un tri numérique sur l'identifiant du contactgroup, donc sur la colonne 3.
# La jointure se fait entre cette colonne 3 et la colonne 1 du fichier des contactgroup déjà trié sur la
colonne 1.
# Les seules informations que l'on retient sont la colonne de l'identifiant du contactgroup (1ière colonne), la
colonne
# du nom du contactgroup (2ième colonne) et la colonne des identifiants de contact de la table de relation
entre contacts
# et contactgroups (3ième colonne).
# Enfin, on joint le résultat au fichier contenant les contacts.
sort -n -k 3 -t "|" contactgroup_contact_relation.txt | join -i -1 1 -2 3 -t "|"  -o "1.1 1.2 2.2" contactgroup.
sorted.txt - \
  | sort -n -t "|" -k 3 \
  | join -i -1 1 -2 3 -t "|" -o "2.2 1.4" contact.sorted.txt - \
  | tr '|' ';' \
  | sed -e 's/\(.*\)/centreon -u admin -p ************ -o CG -a addcontact -v "\1"/'
```