Outils et commandes utiles pour les imports

Lors de l'intégration des données au format d'échange, il peut être nécessaire d'analyser ou corriger les fichiers CSV reçus. Ci-dessous sont indiqués différentes commandes et outils permettant de travailler avec ces fichiers.

Manipulation des fichiers CSV

Les outils ci-dessous permettent de transformer les fichiers CSV et de les analyser :

- CsvKit
- CsvTool
- Miller
    - Exemple de remplacement de valeur de colonne

Extraction de lignes d'un fichier

L'ouverture du fichier `synthese.csv` dans un éditeur de texte peut poser problème contenu de sa taille. Il est possible d'extraire un nombre réduit de lignes du début du fichier à l'aide de la commande suivante :

```
head -1000 synthese.csv > synthese.extract.csv
```

Pour extraire des lignes de la fin du fichier, utiliser la commande :

```
tail -1000 synthese.csv > synthese.extract_end.csv
```

Pour extraire des lignes au milieu d'un fichier, utiliser la commande :

```
sed -n '4100000,4200000 p' synthese.csv > synthese.extract.csv
```

Compter le nombre de ligne d'un fichier :

```
 wc -l synthese.csv
```

Visualiser un fichier TSV dans un terminal

Pour afficher les premières lignes d'un fichier TSV sans l'ouvrir en totalité (gros volume) dans un terminal :

- Créer un fichier `pretty_tsv.sh` dans `~/bin` et y insérer les lignes suivantes :

```
#!/bin/bash
perl -pe 's/((?<=\t)|(?<=^))\t/ \t/g;' "$@" | head -n 10 | column -t -s
```

```
$'\t' | exec less  -F -S -X -K
```

- Editer `~/.bash_aliases` et insérer la ligne suivante :

```
# pretty tsv with first ten lines
alias watch='~/bin/pretty_tsv.sh'
```

- Vous pouvez lancer la commande `watch ../chemin/vers/fichier` dans le Terminal

Extraire les lignes comprenant un nombre de tabulation anormal

Dans le fichier `synthese.csv`, il peut être utile de commencer par repérer les lignes possédant un nombre de tabulation anormal. La tabulation servant à séparer les colonnes, il devrait toujours y en avoir le même nombre pour chaque ligne.

Si ce nombre est supérieur à la normale, cela indique qu'au moins une valeur d'au moins un champ exporté contient une ou plusieurs tabulations.

Si ce nombre est inférieur à la normale, cela indique qu'au moins une valeur d'au moins un champ exporté contient un ou plusieurs caractères de fin de ligne (CR et/ou LF).

Il est possible de repérer les lignes posant problème avec la succession de commandes suivantes :

Si votre espace de stockage principal n'est pas un SSD NVME mais que vous en posséder un sur la machine, l'utilitaire `sort` peut s'en servir en indiquant un dossier sur ce disque à l'aide de l'option `-T`, par exemple : `-T /data-nvme/jpmilcent/tmp/`.

```
# Afficher le nombre de tabulation des lignes du fichier contenant un
extrait de la synthese
# Le chemin /data-nvme/jpmilcent/tmp/ correspond à un dossier temporaire sur
un disque rapide SSD NVM-E
grep -n -o -P "\t" synthese.extract.csv | sort -n  | uniq -c | cut -d : -f 1

# Extraire le nombre de tabulation anormal (**ici différent de 58**) et le
numéro de la ligne correspondante :
# Le chemin /data-nvme/jpmilcent/tmp/ correspond à un dossier temporaire sur
un disque rapide SSD NVM-E
grep -n -o -P "\t" synthese.csv | sort -n | uniq -c | cut -d : -f 1 | grep -
P -v "^\s+58 " > ./synthese.tab_errors.txt

# Supprimer le nombre de tabulation (occupant les 8 premières caractères de
chaque ligne)
# et ne garder que les numéros de ligne dans un fichier
synthese.line_numbers.txt
cp synthese.tab_errors.txt synthese.line_numbers.txt
sed -i 's#^[0-9 ]\{8\}##g' synthese.line_numbers.txt

# Ajouter des 0 initiaux aux numéros des lignes dans une nouveau fichier
synthese.padded_line_numbers.txt :
```

```
while read rownum; do printf '%.12d\n' "$rownum"; done <
synthese.line_numbers.txt > synthese.padded_line_numbers.txt;

# Extraire les lignes qui posent problème (Attention : ordre des lignes non
respecté !)
join -t $'\t' -1 1 -2 1 <(sort synthese.padded_line_numbers.txt) <(nl -w 12
-n rz synthese.csv) | cut -f 2- > synthese.tab_errors.csv

# Après correction, vous pouvez vérifier si le fichier corrigé ne contient
plus de ligne en erreur
# Le chemin /data-nvme/tmp/ correspond à un dossier temporaire sur un disque
rapide SSD NVM-E
grep -n -o -P "\t" synthese.csv | sort -n | uniq -c | cut -d : -f 1 | grep -
P '^(?!     58 )'
# Si tout est OK, la commande ne doit rien afficher...
```

Trouver les doublons

Extraire les lignes dupliquées

- Extraire les lignes dupliquées :

```
 sort -T /data-nvme/jpmilcent/tmp/ synthese.csv | uniq -cd >
synthese.duplicates.csv
```

- Extraire les lignes dupliquées en se basant seulement sur le contenu de la première colonne (remplacer le chiffre dans $1 pour indiquer une autre colonne) :

```
 awk 'cnt[$1]++{if (cnt[$1]==2) print prev[$1]; print} {prev[$1]=$0}'
synthese.csv > synthese.duplicates-first-column.csv
```

Supprimer les lignes dupliquées

- Supprimer les lignes dupliquées en gardant la première trouvée tout en maintenant l'ordre des lignes :

```
 cat -n synthese.csv | sort -T /data-nvme/tmp/ -uk2 | sort -T /data-
nvme/tmp/ -nk1 | cut -f2- > synthese.unduplicated.csv
```

- Supprimer les lignes dupliquées en se basant sur la première colonne tout en maintenant l'ordre des lignes :

```
 awk '!a[$1]++' synthese.csv > synthese.unduplicated-first-column.csv
```

Afficher les lignes dupliquées pour une colonne donnée dans Libre Office

En ouvrant les fichiers CSV à l'aide de Libre Office, il est possible de repérer les doublons présent dans une colonne.

Pour cela sélectionner une colonne, puis ouvrir le menu *Format* > *Conditionnel* > *Condition...*.

Dans la fenêtre qui s'ouvre sélectionner : - "*La valeur de la cellule est*" - puis "*dupliquer*" à la place de "*égale à*". - dans "*Appliquer le style*" choisir le style "*Warning*"

Cliquer sur OK.

Les lignes possédant une valeur dupliquée dans la colonne sélectionnée devraient apparaître avec un texte rouge.

Trouver les valeurs NULL dans les champs obligatoires

- Vérifier la présence de valeur NULL (=\N) dans la colonne 33 (= *nom_cite*) :
  - Vérifier que la colonne 33 correspond bien au champ *nom_cite* avec :

```
head -1 synthese.csv | cut -f33
```

  - Extraction des lignes contenant "\N" dans la colonne 33 :

```
grep -P '^(?:[^\t]+\t){32}\\N\t' synthese.csv  >
synthese.col33_null.csv
```

Affichage/Extraction de lignes contenant une chaîne particulière

Par exemple, pour extraire les lignes du fichier `synthese.csv` contenant la chaine ''' Stéphane\t\tpointage''' (noter la présence de tabulation *\t*) dans un nouveau fichier `synthese.problems.csv` contenant l'entête des colonnes, utiliser les commandes suivantes :

```
# Extraction de la ligne des entêtes de colonne :
head -1 synthese.csv > synthese.problems.csv
# Recherche via RegExp :
grep -P ' Stéphane\t\tpointage' synthese.csv >> synthese.problems.csv
# Recherche d'un texte 100% littéral (pas de regexp) :
grep -F " - Téléphérique de l'Aiguille du Midi. Gare intermédiaire.  Abords
de la gare. \nLa zone parcouru" synthese.csv >> synthese.problems.csv
```

Affichage/Extraction des numéros de ligne contenant une chaîne particulière

```
# Recherche via RegExp :
grep -nP ' Stéphane\t\tpointage' synthese.csv | cut -d: -f1
```

```
# Recherche d'un texte 100% littéral (pas de regexp) :
grep -nF " - Téléphérique de l'Aiguille du Midi. Gare intermédiaire.  Abords de la gare. \nLa zone parcouru" synthese.csv | cut -d: -f1
```

- `-n` (*Line number*) : Demande à grep d'afficher le numéro de la ligne avant le texte (ex: 45231: - Téléphérique...).
- `| cut -d: -f1` : permet de garder seulement le numéro de ligne.
- Pour rediriger vers un fichier ajouter à la fin `> synthese.pattern-lines.txt`

Commandes d'extractions (TSV)

- Extraire les lignes dont la 29ème colonne contient la valeur "*0*" :

```
grep -P '^(?:[^\t]+\t){28}0\t' synthese.csv  > synthese.col29_zero.csv
```

- Présence de guillemets doubles non protégés :
  - cas où aucun champ n'est encadré par des guillemets doubles :

```
grep -P '[^"]"[^"]' synthese.csv > synthese.quote_err.csv
```

  - présence de champ encadré par des guillemets doubles :

```
grep -P '[^"\t]"[^"\t]' synthese.csv > synthese.quote_err.csv
```

- Présence du caractère UTF-8 SUB issu d'un mauvais encodage :

```
grep -P '\x1A' synthese.csv > synthese.utf8_err.csv
```

- Présence d'une succession de chaine NULL (\N) possiblement mal encodé :

```
grep -P '\tN\tN\t' synthese.csv  > synthese.null_err.csv
```

Commandes d'extractions (Semicolon-SV)

- Extraire les lignes dont la 1ère colonne est vide (avec CsvKit) :

```
csvgrep -d ";" -q '"' -u 0 -e "UTF-8" -c 1 -r "^$" synthese.csv > synthese.col1_empty.csv
```

Supprimer des colonnes

Pour supprimer des colonnes dans un fichier utilisant les tabulations comme séparateur de champ, il est possible d'utiliser la commande `cut`.

Par exemple, pour supprimer les colonnes 36 et 37 d'un fichier *synthese.csv*, le résultat étant stocké dans le fichier *synthese.cuted.csv* :

```
cut --complement -f 36-37 synthese.csv > synthese.cuted.csv
```

Sélections de colonnes pour réaliser un fichier de corrections. Ex. sélection des colonnes 1, 3 et 5 à 6, les autres sont supprimées :

```
cut --complement -f 2,4,7- synthese.csv > synthese.fix-2022-03-29.csv
```

NOTES : préalablement à l'utilisation de `cut` assurez vous d'avoir remplacer tous les retours à la ligne présent dans les colonnes par des caractères tel que `\n` ou `\r\n` voir la section *Contourner l'erreur : "sed: la taille du tampon d'entrée d'expression régulière est plus grand que INT_MAX*" ci-dessous.

Remplacer le contenu d'une colonne

Remplacer le contenu de la 33ème colonne dans le fichier *synthese.csv* quand elle contient `\N` avec la commande :

```
sed -i -E 's#^(([^\t]*\t){32})\\N\t#\1\t#' synthese.csv
```

Remplacement de chaîne

Remplacement de chaîne courte

Pour remplacer une chaîne dans un fichier texte donnée, il est possible d'utiliser `sed` avec l'option `-i`. Par exemple, pour remplacer '' Stéphane\t\tpointage'' par '' Stéphane\tpointage'' utiliser la commande :

```
sed -i 's# Stéphane\t\tpointage# Stéphane\tpointage#g' synthese.csv
```

Remplacement de chaîne longue

Remplacement de grande chaîne de texte avec Perl :

```
# Texte actuel
export OLD="La zone parcourue se situe dans le secteur de contact du granite
du Mont-Blanc et des gneiss du sous-bassement du massif, d'où la présence
sur le socle de gneiss de nombreux blocs de granite venus des sommets, épars
ou sous forme de moraines. Le glacier des Pèlerins, recouvert de blocs dans
sa partie aval, est relativement statique et ne marque pas à l'heure
actuelle un \"\"recul\"\" caractéristique, en comparaison avec d'autres
glaciers proches.  "
# Nouveau texte
export NEW="Au contact du granite du Mont-Blanc et du socle de gneiss, la
zone est jonchée de blocs granitiques venus des sommets (épars ou en
```

```
moraines). Couvert de blocs en aval, le glacier des Pèlerins reste
relativement statique : contrairement aux glaciers voisins, il ne marque
actuellement aucun recul caractéristique."
# Remplacement
perl -pe 's/\Q$ENV{OLD}\E/$ENV{NEW}/g' synthese.csv > synthese_propre.csv
```

- \Q et \E (*Quotemeta*) : ordonne à Perl de traiter tout le texte comme du texte littéral.
- Très rapide car Perl lit une ligne, la modifie si besoin, et l'écrit immédiatement dans le nouveau fichier.

Remplacement multi-lignes

Remplacement multi-lignes avec Sed (option `-z` de *sed*) :

```
# Remplacement des retours charriot (CR) suivi de retour à la ligne (LF) (si
erreur INT_MAX, voir ci-dessous)
sed -i -z 's/\r\n/\\r\\n/g' synthese.csv
# Remplacement d'un simple retour à la ligne (LF) présent dans un champ
protégé par des guillemets
sed -i -z 's/\(\t"[^"\t\n]*\)\n/\1\\n/g' synthese.csv
```

Remplacement multi-lignes avec Awk (rassembler 2 lignes successives) :

```
# Exemple de texte de fin de ligne : anticlinal\n
awk '/anticlinal\\n$/ { printf "%s", $0; next } { print }' synthese.save.csv
> synthese.csv
```

- Ici la fin de ligne se termine par les caractères littérals \n. Il faut protéger "\".
- `printf "%s", $0` affiche la ligne courante contenant le pattern sans le caractère de fin de ligne et passe à la ligne suivante `next`.
- Si la ligne ne contient pas le pattern, elle est renvoyée tel quel.

Stats

- Date d'observation la plus ancienne :

```
csvstat -t -q '"' -u 0 -e "UTF-8" -c date_min --min synthese.csv
```

Contourner l'erreur : "sed: la taille du tampon d'entrée d'expression régulière est plus grand que INT_MAX"

```
# Sur le fichier de 4 millions de lignes nous obtenons l'erreur suivante
avec sed et l'option -z
# sed: la taille du tampon d'entrée d'expression régulière est plus grand
que INT_MAX
# Pour contourner le problème nous découpons le fichier en lot de 2 millions
```

```
# Découpage du fichier synthese.csv en 2 fichiers de 2 millions de lignes
sed -n '1,2000000 p' synthese.csv > synthese.1.csv
sed -n '2000001,4000000 p' synthese.csv > synthese.2.csv

# Suppression des retours à la ligne
sed -i -z 's/\r\n/\\r\\n/g' synthese.1.csv
sed -i -z 's/\r\n/\\r\\n/g' synthese.2.csv
sed -i -z 's/\(\t"[^"\t\n]*\)\n/\1\\n/g' synthese.1.csv
sed -i -z 's/\(\t"[^"\t\n]*\)\n/\1\\n/g' synthese.2.csv

# Recréation du fichier synthese.csv à partir des 2 fichiers de 2 millions de lignes
cat synthese.2.csv >> synthese.1.csv ; mv synthese.1.csv synthese.csv ; rm -f synthese.2.csv
```

From:
https://sinp-wiki.cbn-alpin.fr/ - **CBNA SINP**

Permanent link:
https://sinp-wiki.cbn-alpin.fr/database/utilitaires-imports?rev=1787220617

Last update: **2026/08/20 10:10**