G. Hunault
L3 Info

Angers, décembre 2017

Décomposition, Conception et Réalisation d'Applications

1. Dénombrement de naissances

Le fichier national officiel des prénoms en France, tel qu'établi par l'I.N.S.E.E. (Institut National de la Statistique et des Études Économiques), est disponible à l'adresse `https://www.insee.fr/fr/statistiques/2540004`. Il contient des données sur les prénoms attribués aux enfants nés en France entre 1900 et 2015.

Ces données sont disponibles au niveau de la France et par département. Nous avons reproduit en annexe les premières et les dernières lignes de ce fichier volumineux, qui compte, à la date où ce sujet est écrit, 589 412 lignes.

La ligne 1 de ce fichier ne correspond pas à des données mais à la description des 4 champs pour chaque ligne : le champ correspond à un code sexe (1 pour homme, 2 pour femme), le champ 2 donne le prénom usuel attribué, le champ 3 donne l'année de naissance ou XXXX lorsque celle-ci n'était pas renseignée. Enfin, le champ 4 donne le nombre de fois où le prénom a été donné cette année-là (même pour l'année XXXX).

Ainsi, pour la ligne numéro 92 165 qui contient les 4 champs

```
1   GILLES   1957   6844
```

cela signifie que GILLES est un prénom masculin, et qu'en 1957 il a été inscrit 6844 fois comme prénom dans un bulletin de naissance au niveau de l'état civil en France.

Question 1

Donner le code d'un programme AWK nommé prenom1.awk tel que l'exécution de l'instruction en ligne de commandes

```
gawk -f prenom1.awk nat2015.txt
```

renvoie le nombre de prénoms masculins et féminins dans le fichier national des prénoms. On trouverait ainsi :

```
268076 hommes et 321335 dans ce fichier
```

Question 2

Donner le code d'un programme AWK nommé prenom2.awk tel que l'exécution de l'instruction en ligne de commandes

```
gawk -f prenom2.awk nat2015.txt
```

renvoie l'année où il y a eu le plus de naissances, sans distinction de sexe, tous prénoms confondus. Voici là-encore un exemple possible d'affichage :

```
le plus de naissances a eu lieu en 1964 avec 778 691 naissances
```

Question 3

Est-il possible de passer des paramètres à un programme Gawk ? Si oui, indiquer la ligne de commande à exécuter pour que le programme AWK nommé prenom3.awk affiche le nombre de fois où le prénom PAUL a été vu, toutes années confondues (y compris XXXX), PAUL étant passé en paramètre. Donner ensuite le code du programme Gawk.

Si selon vous ce n'est pas possible, donner le code d'un programme Gawk qui initialise un prénom à chercher avec la valeur "PAUL" puis qui compte combien de fois on trouve ce prénom.

La réponse serait ici `407011 fois`.

Question 4

Quelles sont les ambiguités possibles et les difficultés pour bien réaliser la question 3 avec tous les prénoms possibles ?

2. Comment trier un tableau d'objets ?

Dans le cadre du T.P. 1, nous avons créé une classe d'objets Personnes mais nous n'avons pas eu le temps de trier le tableau $tabNP des personnes. Avec un peu de réflexion, au moins trois solutions semblent possibles :

1. écrire explicitement en PHP une fonction de tri des objets,

2. transférer les données dans une structure sqlite3 en mémoire et profiter des fonctions de tri de SQL,

3. transférer les données dans un fichier texte temporaire et les trier avec la commande unix nommée sort.

Laquelle de ces solutions vous parait la plus adaptée au problème, et surtout, pourquoi ?

Vous ne vous contenterez pas de choisir une des trois solutions, vous viendrez également expliquer pourquoi les deux autres solutions sont moins bonnes selon vous.

Si maintenant vous aviez le choix de programmer une solution, serait-ce une de ces trois solutions que vous utiliseriez ou une autre ? Dans ce dernier cas, quelle serait cette autre solution ? Là encore, vous détaillerez le pourquoi de votre choix. On ne demande ici aucun programme, mais juste une discussion.

3. Un peu de culture...

Essayez de répondre à la question suivante :

> *Comment peut-on écrire un cahier des charges et prévoir tout ce que veut l'utilisateur sachant que lui-même ne connait pas forcément toutes les fonctionnalités qu'il peut attendre de l'application à développer ?*

Votre réponse devra essayer de mettre en évidence votre culture naissante, votre recul et votre esprit de synthèse en matière de modélisation et de traitement de l'information.

Cette réponse devra faire 10 lignes au minimum, sans limite de maximum. On utilisera au moins 3 mots de 4 syllabes ou plus pour « transmettre un contenu rédactionnel fort ».

ANNEXE : extrait du fichier nat2015.txt

```
sexe  preusuel  annais  nombre

1     A         1980     3
1     A         1998     3
1     A         XXXX    21
1     AADEL     1976     5
1     AADEL     1978     3
1     AADEL     1980     3
1     AADEL     1981     5
1     AADEL     1982     4

[...]

1     GILLES    2014    19
1     GILLES    2015    18
1     GILLIAN   1967     4
1     GILLIAN   1972     3
1     GILLIAN   1973     3
1     GILLIAN   1974     3
1     GILLIAN   1977     4

[...]

2     ÖZLEM     1984     6
2     ÖZLEM     1988     4
2     ÖZLEM     1989     7
2     ÖZLEM     1991     4
2     ÖZLEM     1992     5
2     ÖZLEM     1993     7
2     ÖZLEM     2010     3
2     ÖZLEM     2012     6
2     ÖZLEM     2013     3
2     ÖZLEM     2014     5
2     ÖZLEM     2015     3
2     ÖZLEM     XXXX    25
```

CORRIGÉ

1. Dénombrement de naissances

Question 1.1

Il suffit de cumuler combien de fois le mot numéro 1 est égal à 1 ou 2 pour trouver combien il y a d'hommes et de femmes. Dans la mesure où le mot 1 de la ligne 1 n'est pas un nombre, on n'a pas besoin de s'occuper de la ligne 1.

Il n'est pas obligatoire, mais c'est plus "propre" d'initialiser les variables dans la partie BEGIN.

Voici le code correspondant :

Fichier prenom1.awk

```
# comptage par SEXE

BEGIN {
  hom = 0
  fem = 0
} # fin de BEGIN

($1==1) { hom++ }
($1==2) { fem++ }

END {
  print hom " hommes et " fem " dans ce fichier "
} # fin de END
```

Question 1.2

Il faut d'abord cumuler tous les résultats de prénom par année avant de pouvoir comparer les années.

La recherche de la meilleure année doit donc se faire dans la partie END via une boucle `for (an in ...)`.

Il est sans doute prudent ici de ne pas prendre la ligne 1 en compte via `(FNR>1)` car l'ordre sur les caractères place les lettres après les chiffres.

Fichier prenom2.awk

```
# comptage par année, on conserve la plus nombreuse

(FNR>1) { parAn[ $3 ] += $4 }

END {
  man = 0  # meilleure année
  cnt = 0  # comptage
  for (an in parAn) {
    ## pour DEBUG print an " : " parAn[ an ]
    cntC = parAn[ an ]
    if (cntC>cnt) {
      man = an
      cnt = cntC
    } # fin si
  } # fin pour
  print "le plus de naissances a eu lieu en " man " avec " cnt " naissances "
} # fin de END
```

Question 1.3

Si on ne sait pas si GAWK permet de passer des paramètres, on peut écrire le test du mot 2 égal à "PAUL" directement dans le code, soit le script :

Fichier prenom3.awk

```
# comptage du nombre de PAUL en tout

BEGIN { nbPaul = 0 }

($2=="PAUL") { nbPaul += $4 }

END {
  print "on a vu " nbPaul " fois le prénom PAUL "
} # fin de END
```

Il est toutefois beaucoup plus propre de mettre le prénom recherché dans la partie BEGIN de façon à fournir un code plus générique et plus facile à maintenir (il est plus simple de changer PAUL en PIERRE tout en haut de fichier qu'au milieu du script).

Fichier prenom4.awk

```
# comptage du nombre de fois en tout où on voit le prénom cherché

BEGIN { nbPre = 0 ; prenom = "PAUL" }

($2==prenom) { nbPre += $4 }

END {
  print "on a vu " nbPre " fois le prénom " prenom
} # fin de END
```

Enfin, GAWK sait utiliser des paramètres, mais le mode de fonctionnement de AWK est particulier car il traite tous les paramètres passés comme des noms de fichier à utiliser. Ainsi écrire

```
gawk -f prenom5.awk nat2015.txt PAUL
```

viendrait demander à GAWK d'ouvrir le fichier PAUL, ce qui n'est pas ce que l'on veut.

Si on lit l'aide GAWK via la commande `man gawk` on apprend que passer des paramètres nommés se fait avec `-v` ou `--assign` et que le paramètre nommé est accessible dès la partie BEGIN.

Fichier prenom5.awk

```
# comptage du nombre de fois en tout où on voit le prénom fourni en paramètre
# syntaxes : gawk -v prenom=PAUL       -f prenom5.awk nat2015.txt
#            gawk --assign=prenom=PAUL -f prenom5.awk nat2015.txt

BEGIN { nbPre = 0 ; print "Recherche du prénom " prenom }

($2==prenom) { nbPre += $4 }

END {
  print "on a vu " nbPre " fois le prénom " prenom
} # fin de END
```

Si aucun prénom n'est fourni, avec ce script, on obtient l'affichage

```
Recherche du prénom
on a vu 0 fois le prénom
```

On pourrait penser qu'un test du prénom au niveau de la partie BEGIN est suffisant pour gérer un prénom vide, mais ce n'est pas le cas : la partie END est quand même exécutée. Donc écrire

```
# comptage du nombre de fois en tout où on voit le prénom fourni en paramètre
# syntaxes : gawk -v prenom=PAUL       -f prenom6.awk nat2015.txt
#            gawk --assign=prenom=PAUL -f prenom6.awk nat2015.txt

BEGIN { nbPre = 0 ;
        if (prenom=="") {
            print "Aucun prénom fourni. Dommage, fin du programme..."
            exit(-1)
        } # fin de si
        print "Recherche du prénom " prenom
} # fin de BEGIN

($2==prenom) { nbPre += $4 }

END {
    print "on a vu " nbPre " fois le prénom " prenom
} # fin de END
```

est insuffisant. Un script complet et propre pour prendre en compte l'absence de paramètre est le suivant

```
# comptage du nombre de fois en tout où on voit le prénom fourni en paramètre
# syntaxes : gawk -v prenom=PAUL       -f prenom7.awk nat2015.txt
#            gawk --assign=prenom=PAUL -f prenom7.awk nat2015.txt

BEGIN { nbPre = 0
        if (prenom=="") {
            print "Aucun prénom fourni. Dommage, fin du programme..."
            exit(-1)
        } else {
           print "Recherche du prénom " prenom
        } # fin de si
} # fin de BEGIN

($2==prenom) { nbPre += $4 }

END {
   if (prenom=="") { exit(-1) }
   print "on a vu " nbPre " fois le prénom " prenom
} # fin de END
```

Question 1.4

On peut se demander comment les prénoms multiples sont stockés et comment les accents sont pris en compte. Dans la mesure où ne dispose pas du fichier, on ne peut qu'émettre des hypothèses, mais il est clair qu'un prénom comme Jean-Pierre doit pouvoir être saisi, de même que Frédérique, avec ou sans distinction du fait qu'il s'agit d'un homme ou d'une femme.

Si on exécute la commande

```
awk -e ' { print $2 } ' nat2015.txt | sort -u | grep "JEAN-"
```

on découvre que les prénoms multiples sont normalisés avec un tiret. On ne sait d'ailleurs pas si seul le premier prénom est enregistré ou si tous les prénoms officiels sont utilisés.

Si on exécute la commande

```
awk -e ' { print $2 } ' nat2015.txt | sort -u | grep "^FR"
```

on découvre que tous les prénoms sont en majuscules, y compris les accents, comme pour FRÉDÉRIQUE.

Les difficultés sont donc de prendre en compte les accents et les prénoms multiples au niveau de la saisie. Une ambiguité peut être de ne pas pas préciser le sexe de la personne, comme par exemple avec le prénom Dominique.

2. Comment trier un tableau d'objets ?

Il n'y a sans doute pas une solution meilleure que les autres si on ne définit pas précisément le critère de qualité ni l'objectif : veut-on seulement l'affichage trié du tableau ou avoir le tableau trié en mémoire ?

Utiliser sqlite3 ou la commande système sort sont des solutions simples car le tri est effectué par des méthodes robustes. Il n'y a qu'à fournir les "bonnes" informations à trier et à interfacer PHP avec SQLITE3 ou UNIX.

Par contre cela oblige à recourir des programmes externes, méthodes peu généralisables si on utilise un serveur non personnel ou configuré sans interface avec SQLITE3 ou UNIX.

Ecrire sa propre méthode de tri (dans la classe des objets Personnes) est une solution pratique mais rien ne garantit que le tri sera juste et efficace.

La "bonne" solution est donc sans doute comme souvent une question de choix personnel.

La méthode la moins risquée passe certainement par sqlite3 en mémoire parce que c'est l'outil le plus naturel pour des tris multicritères, à condition de réinjecter ensuite les données dans le tableau car cela semble bien être la question : trier le tableau (sous-entendu : en mémoire).

Avantages de la solution PHP

- on ne dépend pas du système d'exploitation et on ne fait pas appel à des ressources externes (SQLITE3 ou UNIX)
- le tableau est trié en mémoire donc il reste trié dès que l'opération de tri est effectuée
- on peut insérer directement au bon endroit les personnes dans le tableau dès leur création/instanciation de façon à avoir tout le temps un tableau trié
- il est très classique (et donc assez facile) d'écrire une méthode de tri en PHP

Inconvénients de la solution PHP

- il faut écrire une méthode "propre" de tri multi-critères
- il faut que la méthode de tri prenne bien en compte les accents
- il faut choisir l'algorithme de tri en fonction du volume des données

Avantages de la solution SQLITE3

- un tri multi-critères s'écrit facilement via le paramètre ORDER dans un SELECT
- les tris sont efficaces en SQLITE3
- il est simple d'interfacer PHP et SQLITE3
- la base de données est en mémoire (comme indiqué dans la question) donc les performances seront bonnes, même pour un gros volume de données
- la base de données assure la persistance des données (si on en a besoin) une fois le programme PHP terminé

Inconvénients de la solution SQLITE3

- il faut créer une base de données, y insérer les données
- il faut que PHP soit installé et configuré avec SQLITE3
- il faut s'assurer que la gestion des accents ne pose pas de problème (par exemple : données PHP en iso8859 et base de données SQLITE3 en utf)
- le tableau des objets n'est pas trié, c'est son export qui l'est, il faut donc peut-être réimporter les données pour avoir un tableau trié

Avantages de la solution SORT

- un tri multi-critères s'écrit facilement via les paramètres de SORT
- les tris sont efficaces avec SORT sous UNIX
- il est simple d'exécuter une commande UNIX sous PHP si PHP est configuré en ce sens

Inconvénients de la solution SORT

- il faut que la configuration de PHP autorise d'exécuter des commandes système, ce qui est souvent désactivé pour des raisons de sécurité
- il faut disposer d'une commande SORT, ce qui n'est peut-être pas le cas sous Windows
- il faut s'assurer que la gestion des accents ne pose pas de problème (par exemple : données PHP en iso8859 et UNIX en utf)
- le tableau des objets n'est pas trié, c'est son export qui l'est, il faut donc peut-être réimporter les données pour avoir un tableau trié

3. Un peu de culture...

Il ne fallait pas rappeler ou détailler ce qu'est un cahier des charges mais plutôt essayer d'expliquer comment on peut l'adapter au cours du cyle de développement.