G. Hunault
L3 Info

Angers, décembre 2018

Décomposition, Conception et Réalisation d'Applications

1. Ventes cumulées et meilleures ventes

On s'intéresse ici à des informations de ventes par mois pour des produits différents (colonne `Article`), détaillées par vendeur. Voici quelques exemples de lignes à traiter, sachant que le véritable fichier comporte plusieurs dizaines de milliers de lignes, soit plusieurs dizaines d'années de ventes :

Article	Vente	Mois	Année	Vendeur	refVendeur
NZ01854	3000	10	2017	Louis	456123
NZ01851	9000	10	2017	Olivier	784398
NZ01855	6000	10	2017	Bastien	007007
PK93212	6000	10	2017	Bastien	007007
NZ01856	9000	10	2017	Olivier	784398
[...]					
NZ01855	6000	11	2018	Bastien	007007
YK27869	2545	11	2018	Olivier	784398
YK27867	1789	11	2018	Olivier	888999

On veut en général trouver le vendeur qui a réalisé le meilleur total de ventes pour un mois donné et une année donnée, ou le meilleur cumul des ventes par mois, tous vendeurs et tous articles confondus. Un vendeur est repéré de façon unique par un prénom (colonne `Vendeur`) et par un matricule (colonne `refVendeur`).

On admettra que s'il y a égalité de ventes par mois pour plusieurs vendeurs, c'est le vendeur de plus petit matricule qui est le plus ancien dans l'entreprise et qui doit donc être affiché.

Question 1.1

En admettant que les ventes sont enregistrées dans un fichier texte nommé ventes.txt au format indiqué, donner la fin du code AWK (partie END) correspondant à la commande ci-dessous

```
gawk -f ventes.awk --assign=nbLig=3 ventes.txt
```

afin d'obtenir le résultat suivant, qui fournit sur l'ensemble des années et des mois, les meilleures ventes par mois, tous vendeurs confondus sur les `nbLig` derniers mois, avec des sommes bien cadrées :

```
Informations ventes
   mois 2018/12 somme des ventes     7789
   mois 2018/11 somme des ventes    10334
   mois 2018/09 somme des ventes     6000
```

Le nombre de lignes affichées, hormis la ligne de titre, correspond donc au paramètre `nbLig` fourni en ligne de commandes.

Voici le début du programme `ventes.awk` à compléter :

```
# correspondance entre colonne et numéro de colonne

BEGIN  { somme = 2 ;    mois  = 3 ;    an    = 4 ; }

# parcours des données et stockage par triplet

(FNR>1) { infoMois             = $an "/" $mois
          vntMois[ infoMois ] += $somme
          tabMois[ infoMois ]++
} # fin sur FNR>1

END { # affichage
  # VOTRE CODE ICI
} # fin de END
```

Remarque : ce n'est pas la peine de recopier le début du code AWK. Donner juste la partie qui doit être écrite dans le `END`.

Commencez par décrire les actions à effectuer.

Question 1.2

En admettant que les informations précédentes sur les ventes sont stockées dans une table nommée `vpm` (Ventes Par Mois) d'une base de données

ve (Ventes Entreprise), quelle commande MySQL permettrait d'afficher les meilleurs vendeurs par mois sur l'ensemble des mois, avec les informations suivantes par ligne : cumul vente, vendeur, matricule, mois, année ? On se limitera aux 5 premières lignes d'affichage.

Question 1.3

Les données du fichier `ventes.txt` ont été transférées dans un fichier Excel nommé `ventes.xls`. Donner le code Python quit lit ce fichier et affiche les trois premières lignes correspondant à des ventes de 2017 de plus de 15 000 euros par ordre décroissant de vente. Donner ensuite le code R correspondant.

Dans les deux cas, on commencera par transférer ces informations dans une structure adaptée (tableau, data frame, etc.) afin de pouvoir d'abord afficher le nombre de lignes lues puis celles trouvées correspondant au filtre.

2. Expressions régulières

On admet maintenant que l'ensemble du rapport sur les ventes est disponible sur Internet, sous forme d'une page Web avec des tableaux, des images et des liens pointant vers des fichiers PDF qui donnent le détail des ventes.

Quelles expressions régulières permettent de récupérer (en deux temps), à l'aide de parenthèse(s) capturante(s), toutes les adresses qui commencent par **https** et qui finissent par **PDF** écrit indifféremment en majuscules ou en minuscules dans les liens (éléments `<a>`, attribut `href`) ? On admettra que les URLS sont écrites avec des guillemets et non pas avec des apostrophes.

Si la page est nommée `pdv.html` (pdv=page des ventes), donner le code PHP qui lit ce fichier et produit un affichage bien cadré comme ci-dessous où `Ligne` désigne le numéro de ligne dans le fichier où on trouve le lien :

```
 Numéro  Ligne  Adresse
     1      60   https://us.rf.com/450wm/ventes.pdf
     2     465   https://www.jumpNoSmile.fr/wp/other.pdf
  [...]
```

Attention : il faut prendre en compte le fait que l'attribut `href` n'est pas forcément le premier attribut de l'élément `<a>`, comme ci-dessous :

```
  <a id="lien217" href="https://us.rf.com/450wm/ventes.pdf"> [...]
[...]
  <a class="styleGRK" id="grk31" href="https://us.rf.com/450wm/ventes.pdf"> [...]
```

3. Affichage triable interactif

On dispose d'une page Web "propre" finalisée avec plusieurs tableaux de résultats numériques écrits dans des éléments `<table>`.

Rappeler les deux manipulations à effectuer, présentées dans le cours et les TD, pour qu'on puisse trier les colonnes des tableaux de façon interactive via *Javascript*.

On pourra supposer que le document HTML est écrit en XHTML 1.0 strict ou en HTML5.

4. Un peu de culture...

Imaginez que vous êtes dans la situation suivante : embauché(e) dans une PME (petite ou moyenne entreprise) d'une soixantaine de personnes dont 5 au service informatique, on vous demande de développer un nouveau site pour l'entreprise parce que l'ancien site, écrit en LAMP classique (Linux/Apache/MySql/Php), ne respectait pas MVC. On vous laisse le choix entre développer ce nouveau site avec Python/Django ou avec Ruby/Rails.

Après avoir choisi selon votre goût entre Python/Django et Ruby/Rails, essayez de répondre à la question suivante :

> *Quels arguments techniques et technologiques permettent de confirmer votre choix pour le développement du nouveau site et de convaincre ainsi la direction ?*

Votre réponse devra essayer de mettre en évidence votre culture naissante, votre recul et votre esprit de synthèse en matière de modélisation, de de traitement de l'information et de la programmation. Vous utiliserez les commandes vues en TP et présentées en cours.

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 ».

Contraintes de vocabulaire :

1. Vous devez impérativement utiliser le mot *homoscédasticité* ou le mot *leptokurticité* au moins une fois. Vous pouvez utiliser les deux.

2. Vous devez impérativement utiliser le mot *sapin* ou le mot *olivier* au moins une fois. Vous pouvez utiliser les deux.

ESQUISSE DE CORRIGÉ

1. Ventes cumulées et meilleures ventes

Question 1.1 : dans la partie END du script AWK, il afficher la ligne `Informations ventes`, trier le tableau `tabMois` selon les clés, puis afficher les `nbLig` lignes de valeurs associées dans `vntMois` à partir de la fin, soit le code :

```
END {

  print "Informations ventes "

  nbm  = asorti(tabMois)

  nbla = 1
  while (nbla <= nbLig) {
     jdm = nbm + 1 - nbla
     pdm = tabMois[jdm]
     print "   mois " pdm " somme des ventes " sprintf("%8d",vntMois[pdm])
     nbla++
  } # fin pour idm

} # fin de END
```

Question 1.2 : pour la commande MySQL, il faut définir le cumul des ventes comme un alias sur la somme des ventes, regrouper sur les quatre champs année, mois, vendeur et référence vendeur, trier par ordre décroissant sur le cumul et enfin se limiter à 5 lignes, soit l'instruction :

```
SELECT SUM(Vente) AS cumul  ,
          Vendeur , refVendeur ,
          Mois    , Année
       FROM ve
       GROUP BY Vendeur,refVendeur,Mois, Année
       ORDER BY cumul DESCENDING
       LIMIT 5 ;
```

Question 1.3 : le code Python doit lire le fichier Excel, conserver en mémoire le nombre de lignes lues, filtrer pour ne retenir que les ventes de 2017 supérieures à 15000, trier les lignes filtrées par ventes décroissante puis afficher le nombre de lignes en tout et le nombre de grosses ventes avant d'afficher les premières lignes.

Ce code ressemble fortement à celui des TD :

```
# -*- coding: iso-8859-15 -*-

import pandas as pd

ventes = pd.read_excel("ventes.xls") # lecture du fichier Excel
print(str(ventes.shape[1])+" lignes lues")

# filtre sur 2017 puis filtre sur Vente>15000

ventes2017 = ventes[ ventes.An == 2017 ]
grdVentes  = ventes2017[ ventes2017.Vente > 15000 ]

# tri par ventes décroissantes

ventesOrd = grdVentes.sort_values(["Vente"],ascending=False)

# afichage des nbVal premières lignes

print(str(ventesOrd.shape[1])+" lignes selon filtre")
print(ventesOrd.head(n=3))
```

De la même façon le code R réalise ces actions très simplement, là encore comme avec le code vu en TD :

```
suppressMessages( library("gdata") )
ventes <- read.xls("ventes.xls")  # lecture du fichier Excel
cat(nrow(ventes)," lignes lues\n")

# filtre sur 2017 puis filtre sur Vente>15000

ventes2017 <- ventes[ ventes$An == 2017 , ]
grdVentes  <- ventes2017[ ventes2017$Vente > 15000 , ]

# tri par ventes décroissantes

idx       <- order(grdVentes$Vente,decreasing=TRUE)
ventesOrd <- grdVentes[ idx, ]

# afichage des nbVal premières lignes

cat(nrow(ventesOrd)," lignes selon filtre\n")
print(head(ventesOrd,n=3))
```

2. Expressions régulières

Comme l'énoncé l'a précisé, l'expression régulière `<a href="(.*?)"` n'est pas suffisante pour récupérer les URLS car `href` n'est pas forcément le premier attribut. La solution est sans doute `<a.*? href="(.*?)"` pour les URLS. Il reste ensuite à ne retenir que ce qui commence par https et qui finit par PDF en majuscules ou minuscules, soit l'expression régulière `/^https.*?pdf/i`.

Le code PHP doit donc lire le fichier construire un tableau des URLS puis ne retenir que celles commençant par https et finissant par PDF, soit le code :

```
<?php ## ventes.php

# lecture du fichier pdv.html et stockage en tableau

$tventes = preg_split("/\n/", file_get_contents("pdv.html") ) ;

# récupération des URL

$liens   = array() ; $lignes  = array() ; # non obligatoire mais déclaratif
$nbliens =  -1 ;
$nbligne =   0 ;

foreach ($tventes as $ligne) {
  $nbligne++ ;
  if (preg_match('/<a.*? href="(.*?)"/',$ligne,$res)) {
    $nbliens++ ;
    $lignes[$nbliens] = $nbligne ;
    $liens[$nbliens]  = $res[1] ;
  } ; # fin si
} # fin pour chaque ligne de tventes

# affichage des liens https pour des PDF/pdf

$nburl = 0 ;

foreach ($liens as $numero=>$url) {
  if (preg_match('/^https.*?pdf$/i',$url)) {
      $nburl++ ;
      echo "  ".sprintf("%4d",$nburl) ;
      echo "  ".sprintf("%4d",$lignes[$numero]) ;
      echo "  ".$url ;
      echo "\n" ;
  } ; # fin si
} # fin pour chaque url de liens

?>
```

3. Affichage triable interactif

Comme vu en cours et en TD, un tri interactif dans une page Web peut se faire avec le code Javascript nommé sorttable.

Il faut inclure le fichier `sorttable.js` dans la partie `<head>` de la page, via l'élément `<script>`, attribut `src` puis définir ou ajouter la valeur `sortable` à l'attribut `class` de chaque élément `<table>` des tableaux à trier.