Unité d'Enseignement Mathématiques/Informatique
Année 2020-2021

2e année ESTIA – Étudiants
Examen de Système d'Information – Première session du 3 mars 2021
(durée 2h00, aucun document autorisé)

– – CORRECTION – –

Remarque : Ce document ne présente que des éléments de correction. Les variantes possibles sont nombreuses.

Exercice 2 : Connecteur pour le *datawarehouse* d'une usine « Green Manufacturing » (7 points sur 10)

Les environnements industriels se préparent pour les futurs audits énergétiques. Les usines devront pouvoir justifier leur consommation d'énergie, non seulement auprès des réglementations, mais aussi auprès de leurs clients et de leurs donneurs d'ordre (contexte du *Green Manufacturing*). Les audits énergétiques aideront à identifier les actions à conduire pour diminuer l'impact environnemental des usines. Dans ce cadre, les « **usines micro-réseaux** » seront capables de produire, stocker et utiliser localement leur propre énergie (ex. éolien, photovoltaïque, etc.) pour assurer leur fonctionnement (voir figure 1).

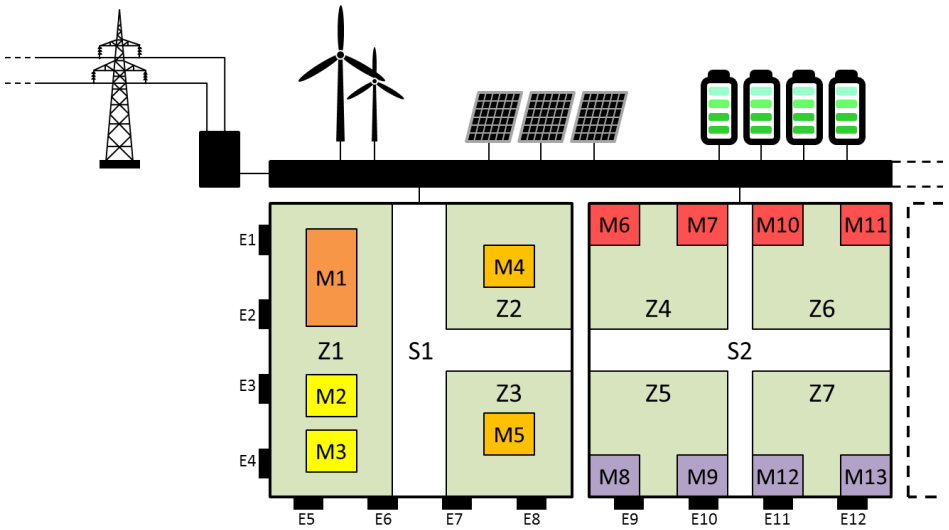

Figure 1 : L'usine (salles, zones, machines, équipements…) sont alimentés par la production centrale (réseau national) et la production du micro-réseau local (éolien, photovoltaïque, stockage…).

Afin, d'une part, de faciliter le pilotage et la prise de décision dans ces usines, de manière à optimiser le taux d'énergie consommé localement, et d'autre part, d'accompagner la conduite des audits énergétiques, des logiciels centraliseront les données énergétiques de l'usine. Ces logiciels permettront de consulter l'historique des données passées et de simuler des opérations de production (*Manufacturing Operations Management, MOM*) notamment en fonction des **prévisions de disponibilité de l'énergie locale**. L'objectif sera alors pour l'usine de consommer le plus possible de l'énergie locale.

Les données de consommation d'énergie (réseau central, réseau d'énergie renouvelable locale direct et stockage d'énergie renouvelable locale) sont enregistrées dans des fichiers CSV[1] où

[1] CSV : Les fichiers Comma Separated Values sont des fichiers ASCII où les données peuvent êtres stockées sur plusieurs lignes et où les colonnes sont marquées par le symbole ";". Ces fichiers peuvent êtres affichés dans un tableur.

chaque ligne indique l'énergie consommée sur des tranches d'une semaine. L'objectif de cet exercice est d'extraire les données utiles pour les calculs de préparation des audits énergétiques en les mettant à disposition du *datawarehouse* (voir figure 2).

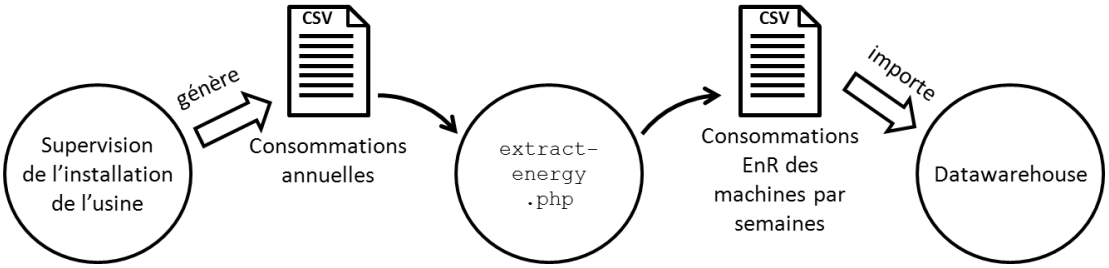

Figure 2 : Le flux de données pour extraire les informations pertinentes dans le *datawarehouse* qui servira aux calculs.

Au sein du flux des données, le script `extract-energy.php` est une « moulinette » qui lit le fichier CSV des consommations annuelles et produit comme résultat un nouveau fichier CSV qui contient les consommations des machines par semaine. Le but de cet exercice est d'écrire ce script `extract-energy.php` qui automatisera ce traitement.

Chaque ligne du fichier CSV des consommations annuelles est structurée avec les éléments suivants :

1. Le numéro de la semaine concernée (entre 1 et 52) ;
2. La machine ou l'équipement concerné (parmi M1-M13 pour les machines et E1-E12 pour les équipements) ;
3. La consommation d'énergie depuis le réseau central sur la semaine concernée, suivie d'un espace, puis de l'unité de mesure (seulement en kWh pour le moment) ;
4. La consommation d'énergie renouvelable depuis le réseau d'énergie renouvelable locale sur la semaine concernée, suivie d'un espace, puis de l'unité de mesure (seulement en kWh pour le moment) ;
5. La consommation d'énergie renouvelable depuis le stockage d'énergie renouvelable locale sur la semaine concernée, suivie d'un espace, puis de l'unité de mesure (seulement en kWh pour le moment).

Sachant que chaque élément est suivie du caractère point-virgule « ; ».

Exemple de trois lignes :

```
1;M7;0.42 kWh;0.22 kWh;0.06 kWh;
1;E9;0.08 kWh;0.05 kWh;0.07 kWh;
2;M1;2.25 kWh;0.23 kWh;0.03 kWh;
```

Pour ses calculs, le *datawarehouse* doit pouvoir recevoir les données de consommation d'énergie renouvelable locale (directe ou stockée) des machines pour chaque semaine. Le *datawarehouse* devra recevoir les éléments suivants :

1. Le numéro de la semaine concernée, précédé d'un « S » ;
2. La consommation totale d'énergie renouvelable par les machines sur la semaine concernée (depuis le réseau d'énergie renouvelable locale ou depuis le stockage), suivie d'un espace, puis de l'unité de mesure (en kWh).

Chaque élément sera suivi du caractère point-virgule « ; ».

Exemple d'une ligne :

```
S1;0.92 kWh;
```

1) Écrivez le code du script `extract-energy.php` qui lit le fichier `consommation_2021.csv` et pour chaque ligne extrait et transforme les données utiles et produit une nouvelle ligne dans le fichier `semaines_machines_2021.csv`. Ci-après un exemple d'exécution de ce script et un mémento des fonctions PHP utiles pour ce travail.

extract-energy.php

```php
<?php

$filename_in = 'consommation_2021.csv' ;

if (!($fdin = fopen ($filename_in, 'r'))) {
    die ('Error: cannot open file '.$filename_in.' in read mode'."\n") ;
}

$filename out = 'semaines machines 2021.csv' ;

if (!($fdout = fopen ($filename_out, 'w'))) {
    die ('Error: cannot open file '.$filename out.' in write mode'."\n") ;
}

$current week = 1 ;
$sum = 0 ;

while (!feof ($fdin)) {
    $line = trim(fgets ($fdin)) ;

    if ($line != '') {
        $tab_line = explode (';', $line) ;
        $week = $tab_line[0] ;
        $name = $tab_line[1] ;

        if ($week != $current_week) {
            fputs($fdout, 'S'.$current_week.';'.$sum.' kWh;'."\r\n") ;

            $sum = 0 ;
            $current_week = $week ;
        }

        if ($name[0] == 'M') {
            $renewable_direct = explode(' ', $tab_line[3]) ;
            $renewable_from_stock = explode(' ', $tab_line[4]) ;

            $sum += $renewable_direct[0] + $renewable_from_stock[0] ;
        }
    }
}

fputs($fdout, 'S'.$current_week.';'.$sum.' kWh;'."\r\n") ;

fclose ($fdin) ;
fclose ($fdout) ;

?>
```

Alternative : *(plus court à écrire, mais moins lisible et moins efficace en temps d'exécution et occupation mémoire… à éviter !)*

extract-energy.php

```php
<?php

$i = -1 ;
$filename in = 'consommation 2021.csv' ;
$filename_out = 'semaines_machines_2021.csv' ;

$fdin = fopen ($filename_in, 'r')
      or die ('Error: cannot open file '.$filename in.' in read mode'."\n") ;

$fdout = fopen ($filename_out, 'w')
       or die ('Error: cannot open file '.$filename_out.' in write mode'."\n") ;
```

```php

while (!feof ($fdin)) {
    $line = trim(fgets ($fdin)) ;

    if ($line != '') {
        list($week, $name, $central, $renewable_direct, $renewable_from_stock) = explode (';', $line) ;

        if ($week != $i) {
            $i = $week ;
            $sums[$i] = 0 ;
        }

        if (substr($name[0], 0, 1) == 'M')
            $sums[$i] += explode(' ', $renewable_direct)[0] + explode(' ', $renewable_from_stock)[0] ;
    }
}

foreach ($sums as $week => $sum)
    fputs($fdout, 'S'.$week.';'.$sum.' kWh;'."\r\n") ;

fclose ($fdin) ;
fclose ($fdout) ;

?>
```

Exemple d'exécution :

(Illustration avec le début du fichier des consommations des deux premières semaines de 2021)

consommation_2021.csv

```
1;M1;0.96 kWh;0.24 kWh;0.00 kWh;
1;M2;0.30 kWh;0.00 kWh;0.00 kWh;
1;E2;0.08 kWh;0.01 kWh;0.01 kWh;
1;E5;0.09 kWh;0.00 kWh;0.00 kWh;
1;M6;0.40 kWh;0.32 kWh;0.08 kWh;
1;M7;0.42 kWh;0.22 kWh;0.06 kWh;
1;E9;0.08 kWh;0.05 kWh;0.07 kWh;
2;M1;2.25 kWh;0.23 kWh;0.03 kWh;
2;M3;1.26 kWh;0.43 kWh;0.11 kWh;
2;M5;0.35 kWh;0.12 kWh;0.03 kWh;
2;E1;0.05 kWh;0.03 kWh;0.02 kWh;
2;E2;0.10 kWh;0.06 kWh;0.04 kWh;
2;M10;0.66 kWh;0.22 kWh;0.22 kWh;
2;M11;0.66 kWh;0.22 kWh;0.22 kWh;
2;M12;0.66 kWh;0.22 kWh;0.22 kWh;
2;M13;0.66 kWh;0.22 kWh;0.22 kWh;
2;E7;0.08 kWh;0.13 kWh;0.19 kWh;
2;E8;0.08 kWh;0.13 kWh;0.19 kWh;
```

semaines_machines_2021.csv

```
S1;0.92 kWh;
S2;2.71 kWh;
```

MÉMENTO : *Voici quelques éléments de PHP pour vous aider à mener à bien ce travail*

- La fonction **fopen** permet d'obtenir un descripteur (type `resource`) d'un fichier `$filepath` ouvert en lecture quand `$mode` vaut `"r"` et en écriture quand `$mode` vaut `"w"`. La valeur retournée par fopen est le descripteur qui a été ouvert, ou faux si l'ouverture a échoué (par exemple en cas de fichier inexistant).

```
resource fopen ( string $filepath , string $mode ) ;
```

- La fonction **fclose** permet de libérer le descripteur de fichier `$fid`.

```
bool fclose ( resource $fid ) ;
```

- La fonction **feof** permet de tester si le descripteur de fichier `$fid` a atteint la fin du fichier. La valeur retournée est vrai si la fin de fichier a été atteinte, faux sinon.

```
bool feof ( resource $fid ) ;
```

- La fonction **fgets** permet de lire une ligne dans le fichier décrit par le descripteur `$fid`. La valeur retournée est une chaîne de caractères contenant la ligne du fichier.

```
string fgets ( resource $fid ) ;
```

- La fonction **explode** permet de décomposer une chaîne de caractères en plusieurs morceaux dans un tableau selon un délimiteur spécifié.

```
array explode ( string $delimiter, string $line ) ;
```

- La fonction **substr** permet de créer une chaîne de caractères de longueur `$length` à partir de la position `$start` de la chaîne de caractères `$line`. Si le paramètre `$length` est omis, alors le résultat sera une chaîne de caractères à partir de la position `$start` jusqu'à la fin de `$line`.

```
string substr ( string $line, int $start, int $length ) ;
```