

Vetux-Line

projet en développement logiciel

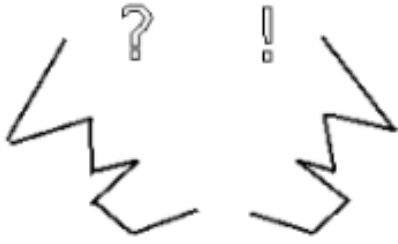
O. Capuozzo

Version 1.2, 2020-09-22

Table des matieres

Présentation.....	1
Vetux-Line	1
Contexte	1
Première partie : Fusion++	1
Les actions de Fusion++	2
Informations techniques de votre chef de projet	2
Ce qui est attendu à l'issue de la première partie	3
Deuxième partie : ETL.....	4
Ce qui est attendu à l'issue de la seconde partie	4
Livraison	5
Annexes	5
Format CSV	5
BOM.....	9
Choisir un composant d'exploitation CSV	10
Exemple de gestion d'upload avec Symfony.....	11

Présentation



Mission de développement d'une application métier (exploitation de documents CSV), de préférence sous la forme d'une application web à base de framework.

Vetux-Line

Contexte

La société de service dans laquelle vous travaillez a comme client l'entreprise VETUX-LINE, createur de ligne de vêtements.

La société VETUX-LINE reçoit de la part de ses partenaires, tous les mois, 2 fichiers clients au format CSV.

Afin d'exploiter ces fichiers partenaires, VETUX-LINE souhaite disposer d'un outil (une application) lui permettant de fusionner ces 2 fichiers en un fichier unique.

La société nous communique des exemples de fichiers reçus. Ces fichiers sont : **french-client.csv** (~3000 clients) et **german-client.csv** (~2000 clients). Ces fichiers ont même structure (même type et nombre de colonnes).

Les fichiers reçus contiennent plus d'information que nécessaire. Le fichier résultant de la fusion sera composé d'un sous-ensemble des colonnes existantes (appelé projection) et une sélection de lignes sera effectuée (sélection des personnes majeures uniquement, suppression de doublons...). La demande du client est détaillée plus loin dans ce document.

Dans un second temps (seconde partie de la mission), le service R&D de la société VETUX-LINE souhaite obtenir ces données sous la forme d'une base de données relationnelle.

Première partie : Fusion++

Votre mission consiste à concevoir une application web qui, à partir de 2 fichiers CSV transmis à l'application par un des gestionnaire de l'entreprise, produira un fichier CSV (nommé **french-german-client[date de creation].csv**).

Ce service ne sera disponible qu'aux gestionnaires de VETUX-LINE.

Le fichier CSV attendu, de nom **french-german-client[date de creation].csv**, sera constitué d'une partie seulement des colonnes existantes (projection) ainsi que d'une sélection de ligne (personnes

majeures uniquement).

Les actions de Fusion++

- **Fusion** : Obtenir un seul fichier à partir de 2 fichiers.
- **Projection** : Toutes les colonnes ne sont pas concernées. Les colonnes souhaitées sont : genre, titre, nom, prénom, email, date de naissance, num tel, CCType, CCNumber, CVV2, CCExpires, adresse physique (plusieurs colonnes), taille, poids, véhicule, coordonnées GPS.
- **Sélection** :
 1. seules les personnes **majeures** à la date de création du fichier devront être sélectionnées.
 2. prise en compte de la **Contrainte-de-taille**. Certains clients ont des incohérences de valeurs entre la taille en inch et celle en cm. Il faudra donc extraire ces clients du fichier résultat.
 3. prise en compte de la **Contrainte-de-ccn** Des doublons sur le numéro de carte de crédit se sont glissés dans les données, ce qui remet en cause l'intégrité des données sur certains clients (dans le système en question, une carte de crédit ne peut être partagée). Nous vous demandons donc d'extraire les clients ne pouvant pas être identifiés par leur numéro de carte de crédit.

Informations techniques de votre chef de projet

Masse des données à traiter

La masse d'information à traiter (~5000 clients) n'aide pas à la mise au point au cours de la première phase de développement.

Il est alors souhaitable de constituer des données de tests afin de réduire, **dans un premier temps**, le volume des données à traiter.

Par exemple travailler avec une vingtaine de clients suffit pour commencer. Par convention, vous nommerez ces fichiers **small-french-client.csv** (12 clients) et **small-german-client.csv** (8 clients) - ces fichiers sont à créer (dans le cas général, ces fichiers n'ont pas le même nombre de lignes)

Fusion entrelacée ou séquentielle ?

Considérons les fichiers suivants, respectivement de 4 et 2 éléments :

```
Fic A : * * * *  
Fic B : ° °
```

La fusion de ces 2 fichiers peut être menée par un algorithme séquentiel ou entrelacé.

Entrelacé	Séquentiel
*	*
o	*
*	*
o	*
*	o
*	o

À ce stade, le client n'a pas manifesté de critère de tri. Votre application devra donc proposer à l'utilisateur la possibilité de choisir le type de fusion à appliquer.

Ce qui est attendu à l'issue de la première partie

A/ Le **README.adoc** ou **.md** de votre projet hébergé sur un dépôt GIT distant fera office de rapport de projet.

B/ Les **opérations liées à la fusion** (des méthodes/fonctions) seront placées dans une **classe à part**, en vue d'une utilisation par **injection de service** dans un de vos contrôleurs.

L'injection de service est une opération fortement facilitée par Symfony (c'est lui qui se charge de l'injection).

Ressource à étudier : les 2 premiers chapitres de : https://symfony.com/doc/current/service_container.html (durée : ~20mn) - Vous apprendrez à créer un service (**MessageGenerator**) et à l'utiliser dans un contrôleur, bien suffisant pour commencer.

- Opérations de fusion en tant que **service injecté** tenant compte des contraintes de taille et de ccn.
- Présentation de votre branche de **tests unitaires** couvrant :
 - les fusions++ séquentielles et entrelacées, avec comme données d'entrée les fichiers **small-french-client.csv** et **small-german-client.csv**.
 - les extractions des clients pour données incorrectes

Votre rapport présentera vos travaux dans ce sens (algorithme, et méthode d'injection utilisée)

C/ Gestion du upload

Les fichiers à traiter sont transmis par l'utilisateur gestionnaire.

Un gestionnaire est un utilisateur ayant des droits spécifiques.

Votre rapport présentera vos travaux dans ce sens (compréhension de la gestion de l'upload dans le cadre d'une applications web multi-utilisateurs)

D/ Gestion du download

In fine, le cas d'utilisation "fusion++" (après le upload des 2 fichiers à traiter) produira le fichier CSV

attendu (type MIME `text/csv`) et certaines autres données : statistiques, clients en erreur.

E/ Au moins deux *Evil User Stories* et leur contre-mesure : Prise en compte de scénarios à risque, formulés sous la forme simple d'un *Evil User Stories* et leur **contre-mesure**. Voir exemple ici : <https://www.riskinsight-wavestone.com/2019/12/cybersecurity-transformation-agile/>

Deuxième partie : ETL

L'entreprise VETUX-LINE envisage de se lancer dans des produits en liens avec l'automobile. Le département R&D souhaite disposer d'un modèle de donnée suivant :

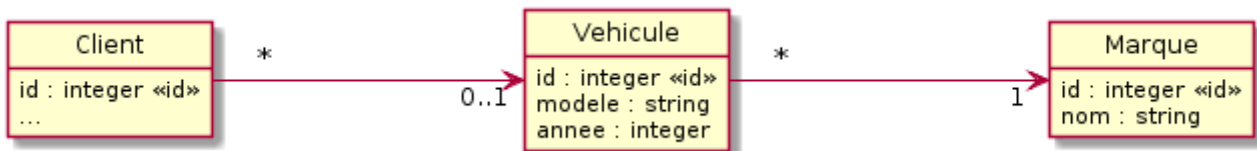


Figure 1. Schema conceptuel client-auto

Votre mission consiste, à partir d'un fichier client CSV issu de la fusion (partie 1), transmis par l'utilisateur (upload), de peupler une base de données de tests correspondant au schéma conceptuel ci-dessus.

Pour cela vous définirez le modèle conceptuel du domaine par un ensemble des entités métier (des classe `entity`).

Votre travail préalable consiste à étudier comment réaliser les relations entre vos objets du domaine.

Étude : <https://symfony.com/doc/current/doctrine/associations.html> vous explique comment réaliser le type de lien `ManyToOne` à travers un exemple (`Product *-----> 1 Category`) (un produit est classé dans une seule catégorie, et une catégorie peut regrouper de nombreux produits)



Attention, à l'issue de cette étude, vous devrez comprendre que le champ `category` de `Product` est une référence à un objet de type `Category` et non à un id de type integer (FK).

Le mapping Objet-Relationnel permettra de représenter les données métier liées, dans la base de données, par des clés étrangères. Exemple : "2000 Ford Galaxy" ⇒ Vehicule (id:123 idMarque:3 model:"galaxy" annee=2000) et Marque (id=3 nom:"Ford")

Ce qui est attendu à l'issue de la seconde partie

- Conception de la partie **Model** (ajout d'entités)
- Lien avec un serveur de base de données (MySQL)
- Conception d'une fonction ELT (*Extract Transform Load*).
- Application de la fonction ETL dans un contrôleur. Mise au point d'un scénario utilisateur intégrant des règles de validation (robustesse de l'application)

- Prise en compte de un ou plus scénarios à risque, formulés sous la forme simple d'un *Evil User Stories* et leur **contre-mesure**. Voir exemple ici : <https://www.riskinsight-wavestone.com/2019/12/cybersecurity-transformation-agile/>



L'utilisateur pourra être en mesure de renouveler son action avec de nouvelles données ou des données mises à jours. **Le chargement de nouvelles données ne devra pas générer de doublons dans la base de données.**

- Une représentation graphique de données statistiques (répartition des marques parmi les clients) est attendue sur le tiers client. Les données exploitées pour cette représentation seront tirées de la base de données. À vous de proposer une vue adaptée pour le service R&D.
- (optionnel) Une fonction d'export de données client serait appréciée (format à déterminer).

Livraison

La date de livraison est : **dimanche 24 octobre 2021 - 23h59**

Vous communiquerez, par un des membres du groupe - les autres en CC, une version **pdf** de votre rapport. Le fichier sera constitué du nom de la team.

Votre rapport sera **daté**, comportera le nom de tous les collaborateurs de la team ainsi qu'un lien vers l'annoncé de la mission (ce doc) et un autre vers votre projet hébergé. Rappel, votre rapport est le README de votre projet.

Annexes

Format CSV

Il existe plusieurs solutions pour que 2 systèmes puissent communiquer des données, indépendamment de leur implémentation interne spécifique (structure, encodage). La plupart du temps, le choix d'un fichier texte est privilégié à celui dit « binaire ». Parmi les solutions actuellement en activité on trouve plus couramment les formats : **XML**, **JSON** et **CSV**.

Le format CSV est le plus ancien. Il est toujours utilisé, (système embarqué, instrument de mesure, données satellitaires, export/import base de données, etc.).

CSV (*Comma-separated values*), est un format informatique ouvert représentant des données tabulaires sous forme de valeurs séparées par des virgules.

La **RFC 4180** décrit la forme la plus courante de ce format et établit son type MIME **text/csv**, enregistré auprès de l'autorité l'IANA qui a autorité sur les noms de domaines et tout ce qui touche à l'interconnexion de réseaux à internet.

Un fichier CSV est un **fichier texte**, par opposition aux formats dits « binaires ». Chaque ligne du texte correspond à une ligne du tableau et les virgules correspondent aux séparations entre les colonnes. Les portions de texte séparées par une virgule correspondent ainsi aux contenus des cellules du tableau.

Une ligne est une suite ordonnée de caractères terminée par un caractère de fin de ligne (line break – CRLF), la dernière ligne pouvant en être exemptée.

Fichier au format .csv	Représentation tabulaire		
Sexe,Prénom,Année de naissance M,Alphonse,1932 F,Béatrice,1964 F,Charlotte,1988	Sexe	Prénom	Année de naissance
	M	Alphonse	1932
	F	Béatrice	1964
	F	Charlotte	1988

⇒ Attention : la première ligne désignant les "entêtes de colonne" est optionnelle.

⇒ Format CSV en détails : <https://tools.ietf.org/html/rfc4180>



Les fichiers CSV sont, par défaut, ouverts par des logiciels tableur (Calc, Excel...). C'est une source de confusion des utilisateurs lambda, confondant **CSV** avec ... Excel.

Extrait de la RFC 4180

(<https://tools.ietf.org/html/rfc4180>)

Definition of the CSV Format While there are various specifications and implementations for the CSV format, there is no formal specification in existence... but :

1. Each record is located on a separate line, delimited by a line break (CRLF). For example:

```
aaa,bbb,ccc CRLF
zzz,yyy,xxx CRLF
```

2. The last record in the file may or may not have an ending line break. For example:

```
aaa,bbb,ccc CRLF
zzz,yyy,xxx
```

3. There maybe an optional header line appearing as the first line of the file with the same format as normal record lines. This header will contain names corresponding to the fields in the file and should contain the same number of fields as the records in the rest of the file (the presence or absence of the header line should be indicated via the optional "header" parameter of this MIME type). For example:

```
field_name,field_name,field_name CRLF
aaa,bbb,ccc CRLF
zzz,yyy,xxx CRLF
```


4. Within the header and each record, there may be one or more fields, separated by commas. Each line should contain the same number of fields throughout the file. Spaces are considered part of a field and should not be ignored. The last field in the record must not be followed by a comma. For example:

```
aaa,bbb,ccc
```

5. Each field may or may not be enclosed in double quotes (however some programs, such as Microsoft Excel, do not use double quotes at all). If fields are not enclosed with double quotes, then double quotes may not appear inside the fields. For example:

```
"aaa","bbb","ccc" CRLF  
zzz,yyy,xxx
```

6. Fields containing line breaks (CRLF), double quotes, and commas should be enclosed in double-quotes. For example:

```
"aaa","b CRLF  
bb","ccc" CRLF  
zzz,yyy,xxx
```

7. If double-quotes are used to enclose fields, then a double-quote appearing inside a field must be escaped by preceding it with another double quote. For example:

```
"aaa","b""bb","ccc"
```

Principes d'exploitation d'un fichier CSV

Voici l'algorithme générique de lecture d'un fichier texte de type CSV :

- (1) Ouvrir le fichier en lecture / ou écriture (création du fichier possible)
- (2) Tentative de lecture de la première ligne
- (3) TantQue nous obtenons une ligne
- (4) Faire quelque chose avec la ligne en question
- (5) Tentative de lecture de la prochaine ligne
 FTQ
- (6) Fermeture du fichier

Exemple avec PHP

r : lecture seule
r+ : lect/ecr
a : mode ajout
voir doc : fopen

```
1  <?php
2  $row = 1;
(1) 3  $handle = fopen("french-data.csv", "r");
4
5  if ($handle) {
(2) 6      $ligne = fgetcsv($handle, 1000, ",");
(3) 7      while ($ligne) {
8          $num = count($ligne);
9          echo "<p> $num champs à la ligne $row : <br /></p>\n";
10         $row++;
(4) { 11         for ($c=0; $c < $num; $c++) {
12             echo $ligne[$c] . "<br />\n";
13         }
(5) 14         $ligne = fgetcsv($handle, 1000, ",");
15     }
(6) 16     fclose($handle);
17 } else {
18     echo "Ouverture impossible !";
19 }
```

À l'image d'un curseur qui avance à chaque nouveau caractère injecté dans un texte (par l'action d'une touche sur le clavier), la fonction `fget` « consomme » le contenu du fichier (fait avancer le curseur de lecture, après chaque lecture de ligne) jusqu'à atteindre la fin du fichier.

Voir la documentation en ligne de la fonction `fgetcsv` : <http://php.net/manual/fr/function.fgetcsv.php>

Testez vos connaissances

Voici un exemple de lecture d'un fichier CSV, proposé par la communauté PHP (aide en ligne)

Reportez sur la colonne de gauche, le numéro d'étape de l'algorithme générique de lecture.

```

$row = 1;
if (($handle = fopen("test.csv", "r")) !== FALSE) {
    while (($data = fgetcsv($handle, 1000, ",", "")) !== FALSE) {
        $num = count($data);
        echo "<p> $num champs à la ligne $row: <br /></p>\n";
        $row++;
        for ($c=0; $c < $num; $c++) {
            echo $data[$c] . "<br />\n";
        }
    }
    fclose($handle);
}

```

BOM

Parceque les fichiers CSV sont des fichiers "texte" (par opposition au fichier "binaire"), il est nécessaire de savoir que ce type de fichier peut intégrer une méta-donnée, nommée **BOM** dans les tous premiers octets.

BOM (de l'anglais *Byte Order Mark*, parfois traduit en français par *indicateur d'ordre des octets*) est une donnée qui indique l'utilisation d'un encodage unicode ainsi que l'ordre des octets. Cette donnée est située au début de certains fichiers texte.

La donnée du BOM, lorsqu'elle est correctement traitée, est transparente pour les utilisateurs lambda, dans le cas contraire où la séquence de BOM est traitée comme du texte, elle apparait souvent sous cette forme : `ï»¿` et peut alors perturber certains traitements.

Voir plus loin : https://fr.wikipedia.org/wiki/Indicateur_d%27ordre_des_octets

Les 2 fonctions de cette donnée optionnelle, placée en tête des fichiers texte renseigne :

- Unicode : UTF-8, UTF-16, UTF-32, ...
- Ordre des octets : big ou little indian. Concerne la représentation mémoire de groupes d'octets : les représentations de poids fort sont-elles en premier ou en dernier ? (voir : <https://fr.wikipedia.org/wiki/Boutisme>)

Table 1. Exemples de BOM

Information de codage	Séquence d'octets de BOM (hexa)
UTF-8	EF BB BF
UTF-16 Big Endian	FE FF
UTF-16 Little Endian	FF FE
UTF-32 Big Endian	00 00 FE FF
UTF-32 Little Endian	FF FE 00 00

Information de codage	Séquence d'octets de BOM (hexa)
UTF-EBCDIC	DD 73 66 73

Le standard Unicode n'impose pas BOM pour les fichiers texte, mais le permet ; c'est le cas en particulier pour UTF-8, où l'indicateur est facultatif. (voir : https://fr.wikipedia.org/wiki/Indicateur_d'ordre_des_octets)



L'acceptabilité de BOM dépend des protocoles utilisés. À des fins d'interopérabilité, les logiciels ont tendance à le reconnaître lorsqu'il est présent, et les utilisateurs à l'enlever lorsqu'il n'est pas reconnu par un logiciel.

Remarque, voici une commande pour connaître l'encodage de votre système (*big endian* ou *little endian* ?) :

```
python -c "import sys; print(sys.byteorder)"
```

Choisir un composant d'exploitation CSV

En PHP, les fonctions `fgetcsv` et `fputcsv` sont qualifiées de relativement « bas niveau ». L'usage de ces fonctions nécessite de prendre quelques précautions comme l'encodage des fichiers à exploiter, la présence de BOM, le format de fin de ligne, etc. Autant de paramètres qui, normalement, sont pris en charge par des composants dédiés, et il y en a plus d'un !

Voir les composants disponibles via `composer` : <https://packagist.org/?query=csv>

À la date de cette recherche (7 octobre 2020), on ne compte pas moins de 627 composants ! Il est donc nécessaire de considérer des critères de sélection.

Critères de sélection communs les plus courants

- Nombre de téléchargements
- Nombre d'étoiles
- Dépendances (requires)

Critères indirects

- Nombre d'applications dépendantes

Critères spécifiques à la fonction

- Charge mémoire
- Style de programmation (procédurale vs événementielle)
- Prise en compte d'autres formats (import/export)
- ...




Selon la taille des fichiers à manipuler, le critère de charge mémoire peut être déterminant.

Exemple de recherche CSV sur packagist (trié)

Packagist

BrowseSubmitCreate accountSign in



CSV

Packagist is the main Composer repository. It aggregates public PHP packages installable with Composer.

maatwebsite/excel Supercharged Excel exports and imports in Laravel	PHP	21 976 488 9 211
league/csv CSV data manipulation made easy in PHP	PHP	22 328 680 2 705
box/spout PHP Library to read and write spreadsheet files (CSV, XLSX and ODS), in a fast and scalable way	PHP	7 497 892 3 242
goodby/csv CSV import/export library	PHP	2 514 263 942

Exemple de gestion d'upload avec Symfony

Un code simple mais bien détaillé, avec une dose de sécurité

<http://zetcode.com/symfony/uploadfile/>