Générateur universel de données pour tests logiciels

Projet 2

Projet Tuteuré

LP Logiciels Libres 2016-2017

Semestre 2

Encadrant : Gilles Hunault

Réalisation :

Brice Harismendy

Corentin Couvry

Geoffroy Berry

Table des matières

1Introduction....3
1.1Cadre du travail....3
1.2Objectifs du document....3
2Présentation du projet....3
2.1Présentation....3
2.2A propos de la génération....4
2.3A propos du rapport....4
2.4A propos des formats de sortie....5
2.5A propos de l'interface Web....5
2.6A propos du programme et de l'interface Web....5
3Travail réalisé....5
3.1Analyse de l’existant....5
3.2Fonctionnalité....6
3.3Architecture....7
4Gestion et organisation....9
4.1Groupe....9
4.2Projet....9
5Améliorations possibles....9
5.1Points non traités par manque de temps....9
5.2Points à développer....10
5.3Taille du code....11
6Conclusion....11
6.1État des faits....11
6.2Apport....11
7 Annexes....12
7.1Répartition des tâches....12
7.2Diagramme Gantt du projet....13
7.3Sources....14
Sources des dictionnaires :....14
Source de formation....14
7.4Cas d’utilisation du générateur....15
7.5Cas d’utilisation de l’interface....16
7.6Exemple de fichier de sortie....17
7.7Rapport de génération....20
7.8Fichier de configuration XML....21

1 Introduction

1.1 Cadre du travail

Dans le cadre de nos projets tuteurés du deuxième semestre en Licence professionnelle Logiciels Libres à l'Université d'Angers lors de la session 2016-2017 nous avons eu pour objectif de réaliser la suite du projet de génération de données du premier semestre en l'adaptant à une interface web dans le but de la rendre « user friendly » nous avons également eu à retravailler l'algorithme pour y inclure les fonctionnalités désirés et régler les problèmes existant.

1.2 Objectifs du document

L'objectif de ce document est de présenter au client le travail qui a été réalisé au cours de ce projet. C'est à dire d'une part de lui décrire l'application telle que nous lui avons livré en lui décrivant chaque fonctionnalité qui a été développée. Ce document a également pour objectif d'expliquer au client les méthodes que nous avons employé pour mener le projet à bien, que ce soit au niveau de l'organisation dans le projet ou la communication dans le groupe.

Enfin le dernier objectif de ce document est de faire un point sur le projet, c'est à dire d'analyser ce qui s'est bien déroulé et les points que l'on peut encore améliorer afin d'être plus performant lors d'un prochain projet.

2 Présentation du projet

Ce paragraphe est extrait de la page de présentation du projet réalisée par notre tuteur.

http://forge.info.univ-angers.fr/~gh/Projets/Lp/proj2016.php

Cela à été notre 'fil conducteur' au cours de nos analyses et réalisations.

2.1 Présentation

Lorsqu'on écrit un programme informatique ou lorsqu'on conçoit un système d'informations avec des bases de données, on a souvent besoin de jeux de données de tests, parfois très structurés et avec de gros volumes de données.

Il serait très utile de pouvoir disposer d'un générateur **avancé** de telles données. Le but du projet est de concevoir les types de données et de structures de données liées à des générations automatiques de données pour tests et d'implémenter un tel générateur qui fonctionnera en ligne de commandes et via une interface Web. La configuration de la génération sera décrite dans un fichier **XML.**

La première partie du projet avait pour but de montrer la faisabilité d'un tel générateur. La deuxième partie vient professionnaliser le programme et lui ajouter une interface Web.

2.2 A propos de la génération

On viendra normaliser le fichier XML (tous les noms en français) et adapter la grammaire XSD en conséquence. Pour le choix des formats de sortie, on utilisera les valeurs 0 et 1 plutôt que TRUE et FALSE et on vérifiera qu'au moins un format de sortie est choisi. On ajoutera un élément **<description>** en texte libre pour référence, obligatoire mais pouvant être vide, afin d'identifier l'auteur, la date etc. tout au début du fichier XML.

On trouvera un moyen de générer des noms et des prénoms composés avec des espaces et/ou des tirets. Après avoir lu la page de référence stats du manuel PHP, on ajoutera d'autres lois statistiques de génération.

Pour les variables numériques, on mettra les valeurs MIN=0 et MAX=12 comme valeurs par défaut. Pour les variables caractères, on rajoutera un type élémentaire nommé *email*.

Pour les champs multiples dans une même table, on commencera par générer **Mr/Me + Prénom + Nom** avec des options majuscules/minuscules pour fournir par exemple le(s) prénom(s) en minuscules avec seulement l'initiale en majuscule et les noms tout en majuscules. On essaiera ensuite de fournir des adresses multiples avec des couples cohérent **code postal/communes**.

On trouvera et on implémentera un formalisme pour décrire le mécanisme clé primaire/clé secondaire pour des champs de deux tables distinctes.

2.3 A propos du rapport

Le rapport devra commencer par afficher la date et l'heure et afficher tout de suite la graine de génération. Une durée du temps de génération sera affichée et on viendra ensuite analyser les données produites afin de fournir un rapport complet sur l'ensemble des données générées. Ce rapport affichera la position des valeurs NULL quand ces valeurs existent, leur nombre et leur fréquence (à une décimale près).

De plus, si l'utilisateur a demandé une sortie SQL, on fournira le code SQL qui permet d'analyser [sommairement] la base de données et le rapport d'exécution issu de ce code.

2.4 A propos des formats de sortie

On ajoutera les formats de sortie JSON et XHTML et on essaiera de trouver un système efficace de visualisation pour les données XHTML avec des "volets figés" comme sous Excel.

2.5 A propos de l'interface Web

On choisira une interface «*évoluée*» après avoir comparé les éléments d'interface (que l'on listera dans un fichier Excel) pour les sites Web trouvés dans la première partie du projet.

On prévoira de pouvoir charger au choix un ancien fichier XML correspondant à une génération ou un fichier SQL de commandes CREATE pour définir le générateur avec modifications possibles.

On écrira l'interface en PHP conceptuel, quitte à rajouter des fonctions pour HTML5.

2.6 A propos du programme et de l'interface Web

L'interface Web utilisera le programme de génération en ligne de commandes pour générer les données après avoir construit le fichier XML du modèle de génération. On prendra grand soin d'avoir un code PHP «*propre et concis*» bien commenté, avec des fonctions qui factorisent au maximum le code

3 Travail réalisé

3.1 Analyse de l’existant

Lors de la première partie de projet nous avions déjà réalisé le générateur de données universelle bien que moins complet que celui-ci il était déjà fonctionnelle, Nous avions donc réalisé un générateur qui était un script PHP, exécuté en ligne de commande depuis une console. Le script a pour but de charger le document XML contenant les données à générer et de sortir les données selon le(s) format(s) de sortie spécifié(s) (XML/ SQL et CSV) (cf annexe -7.6 Exemple de fichier de sortie).

Le paramétrage de la génération est quant à lui fait dans un fichier XML qui est lui même validé par un fichier XSD. Un schéma XSD est une feuille de validation d'un fichier XML nous permettant de nous assurer que l'utilisateur rentre les bons paramètres dans le fichier XML servant à la génération.

On exécute d'abord le programme generer.php en ligne de commande :

$ php generer.php Parametre.xml

Séquentiellement, generer.php appelle le fichier parametre.xsd qui va analyser le fichier Parametre.xml. Si le fichier est valide, la fonction d'appel de parametre.xsd renvoi un booléen “True”.

3.2 Fonctionnalité

Pour commencer nous avons fait une correction des bugs majeur dans generer.php comme la limite des 100 000 lignes généré ou encore des variables non initialisé, nous avons également scindé le code en librairie pour plus de lisibilité, enfin certaines parties du code ont été optimisés pour alléger l’exécution du générateur.

Nous avons ensuite mis en place une interface contenant un formulaire c’est donc logiquement que nous avons poursuivit avec le script de traitement du formulaire qui génère un fichier de paramètre XML, lance la génération et propose le téléchargement des données à l’utilisateur.

Les utilisateurs pouvant faire des erreurs lors de la saisie nous avons mis en place une validation des champs qui utilise des expressions régulières pour vérifier par exemple que des lettres ne sont pas entrés dans le champ « nombre de ligne » … Ceci s’accompagne d’une vérification des champs avant l’envoie pour s’assurer que tout les champs nécessaire à la validation sont remplis.

Lors de la génération des données on pourrait avoir besoin de connaître le sexe concernant un prénom ou encore le code postale d’une ville, c’est pourquoi nous avons mis en place la gestion des champs multiples dans le générateur.

Étant donné que l’on permet l’import de fichiers XML nous avons mis en place l’import de fichier de description de base de données SQL à l’aide d’un script de transformation du SQL en XML.

Les utilisateurs peuvent avoir besoin d’exemple pour comprendre le fonctionnement de l’interface c’est pour cela que nous avons mis en place 5 Exemples d’utilisation de l’interface ainsi que deux fichiers accessible dans l’aide qui montre ce que l’on rentrer dans les différents champs.

Jusqu’à présent nous avions que très peu de dictionnaires nous avons donc ajouté plusieur s dictionnaires au générateur pour atteindre le nombre de 20 dictionnaires. (cf : Annexe - Sources des dictionnaires).

Tout au long du projet nous avons corrigé des bugs qui étaient présent dans le générateur ou dans l’interface pour assurer un code de bonne qualité.

Enfin nous avons modifié le fonctionnement du rapport ainsi que sont contenu en ajoutant l’horodatage pour permettre à l’utilisateur de connaître la durée de génération.(cf Annexe – 7.7 Rapport de génération)

3.3 Architecture

Système de paramètre

Pour visualiser le fonctionnement du générateur et de l’interface nous avons créé des cas d’utilisation ce qui nous a permis de mieux faire comprendre comment se déroulerait chaque exécution (cf annexe -7.4 Cas d’utilisation du générateur-7.5 Cas d’utilisation de l’interface)

Au début de ce projet tuteuré nous avons mis en place l'interface web pour permettre aux utilisateurs de générer des données via une interface web.

Ne pouvant pas déterminer le nombre de table ou de colonnes que voudrait générer l’utilisateur nous avons donc mis en place un script de gestion dynamique de l’interface qui joute dynamiquement des colonnes ou des tables ainsi que des champs en fonction des choix de l’utilisateur.

Étant donné que les utilisateurs peuvent récupérer des fichiers de configuration via une utilisation précédente ou en créer nous avons mis en place de l'importation de fichier de configuration dans l'interface en bloquant l’accès qu’à certains formats (.sql/.txt/.xml).

Pour pouvoir aider les utilisateur novice nous avons mis en place l'aide utilisateur qui explique comment utiliser les champs ainsi que des explications en cas d’erreurs d’insertion dans un champ.

Pour prévoir les futurs évolutions du générateur nous avons mis en place une documentation développeur qui explique comment fonctionne le générateur.

Dans un projet aussi complexe nous avons du mettre en place une arborescence de fichiers, le but de cette arborescence étant de garder séparé le générateur et l'interface pour que l'on puisse continuer à utiliser le générateur en ligne de commande, à la racine nous avons donc un dossier « Generateur » et un autre « Interface ».

Intéressons nous tout d'abord au dossier « Generateur » il est composé à ça racine d'un fichier generer.php et de Paramètre.xsd qui sont respectivement le générateur et le validateur de fichiers de configuration, puis nous avons le dossier Librairie qui contient toutes les librairies utilisées dans generer.php, nous avons également le dossier Dictionnaire qui contient tout les dictionnaires des données et enfin le dossier Sortie qui contient les fichiers générés par défaut.

Du coté du dossier « Interface » nous avons à la racine le fichier « index.html » qui contient la page par défaut de l'interface et « Aide.xhtml » qui contient la documentation pour les utilisateurs. Nous avons dans ce dossier le répertoire Scripts qui contient le script des exemples, de la gestion du formulaire, celui de l'importation de document ainsi que la récupération des documents importés, enfin il y a le script de validation. Ensuite à la racine de l'interface il y a le dossier Style qui contient tout le css et Bootstrap, enfin il y a le répertoire « ressource » et ses sous répertoire « images » et « Files » qui contiennent respectivement des images et des fichiers exemple.

L'interface en Bootstrap a été un choix stratégique pour économiser du temps, en effet bootstrap permet d'avoir un css responsive design et déjà aboutit nous avons juste ajouté un fichier nommé style.css dans le répertoire css.

Lors de l'importation d'un fichier SQL une conversation est ouverte avec le serveur en Ajax et a pour but d'exécuter le script « XMLversSQL.php » qui retourne ensuite au navigateur client le contenu du fichier au format xml et qui ensuite l'implémente dans l'interface.

Lors de la génération l'on créé en plus des fichiers contenant des données un rapport et un fichier de paramétrage qui sont mis dans l'archive téléchargé par l'utilisateur, ces fichier doivent être momentanément stocké sur le serveur, et sont donc créé dans le /tmp du serveur linux le temps du téléchargement.

Pour permettre de tester toutes les possibilités nous avons mis au point un fichier de paramétrage qui prend en compte toutes les possibilités de génération et ainsi vérifier que tout est fonctionnel. (cf annexe -7.8 Fichier de configuration XML).

4 Gestion et organisation

4.1 Groupe

Lors de la première partie du projet auquel participaient déjà Brice Harismendy et Corentin Couvry, Brice avait pour rôle chef de projet c'est donc dans la logique de la continuité que Brice à décider à nouveau d'endosser le rôle de chef de projet. Cette fois-ci, étant donné que nous étions trois nous avons choisis une organisation orale et ne nous sommes pas appuyé sur des outils tel que Trello comme ce fut le cas précédemment. Geoffroy Berry qui n'était pas familier du projet a pu bénéficier pour les deux premiers jours d'un temps d'adaptation et de compréhension du projet et du code existant afin d'être capable de suivre les améliorations et d'être plus efficace sur le reste du projet.

4.2 Projet

Les tâches été assignés au fil de l'avancé du projet et des besoins de manière équitable entre les membres. D'une manière générale nous avons travaillé ensemble sur chaque fonctionnalités de manière à ce que lorsque un membre rencontré des difficultés ou que la tâche devait être fini afin de permettre d'aller vers de nouvelles fonctionnalités nous étions tous capable de comprendre le problème et d'y trouver une solution, nous avons ainsi réalisé les tâches en via un un diagramme des tâches de type Gant (cf annexe -7.1 Répartition des tâches).

5 Améliorations possibles

5.1 Points non traités par manque de temps

Un système de clé primaire et secondaire qui permettrait de s'approcher du fonctionnement d'une base de donnée , en effet comme dans une base de donnée le but serait d'avoir une correspondance entre deux tables avec un champs dans une table qui sert de clé primaire et un autre qui sert de clé secondaire dans une autre table.

La transformation de l'interface du HTML au PHP Conceptuel dans le but d'alléger le code.

L'ajout d'un identifiant auto incrémenté pour la sortie au format CSV.

5.2 Points à développer

La factorisation du code des scripts en Javascript, en effet dû à notre découverte d'ajax, Jquery et Javascript le code n'est pas optimale et pourrait être amélioré pour simplifier le développement des futurs ajouts.

Le système de « Codage » qui est la correspondance entre un numéro et une donnée pourrait être améliorer en effet nous avons tenter d'implémenter les signes « ≥=, =<, >,< » mais sans succès par manque de temps.

Les Champs multiple rétrocompatible, en effet actuellement on peut obtenir un code postale à partir du nom d'une ville l'idée serait de pouvoir faire l'inverse.

Il faudrait également ajouter l'analyse après génération qui permettrait d'empêcher l'apparition de doublons dans les données générée (par exemple empêcher deux fois même nom et prénom).

Il faudrait augmenter le nombre de mode de génération pour ainsi être capable de générer des nombres selon des lois mathématique comme la loi normale.

De plus il serait à notre sens très intéressant que les utilisateurs puissent ajouter leur propre dictionnaire ainsi qu'augmenter le nombre de dictionnaire présent pour pouvoir satisfaire un maximum de demande.

La génération de texte et donc de phrase est un point qui pourrait être intéressant pour permettre par exemple de simuler des forums…

Un ajout qui nous semble important est une barre de chargement pour permettre à l'utilisateur de savoir où en est sa génération et ainsi l'informer de l'avancement.

5.3 Taille du code

Pour donner une idée de la taille du projet et de la complexité du projet voici le nombre de ligne de code par langage, on a coder 1226 lignes en PHP, 1644 ligne en javascript et 34 lignes en Ajax/Jquery ce qui peut commencer à donner une idée de la complexité du code est également le nombre de fichier, en effet le code est répartit dans 28 fichiers différents.

6 Conclusion

6.1 État des faits

À la fin de ce temps de projet nous avons amélioré considérablement le générateur en l'équipant d'une interface complexe et répond ce qui lui permet de répondre en grande partie à l'énoncé fournit par notre tuteur au début du projet cependant quelques spécificités n'ont pas put être implémenté par manque de temps. Notre programme fonctionne contrairement à beaucoup de ses concurrent en ligne de commande et via l'interface fournit.

En ce qui concerne les fonctionnalités implantées nous avons créez une interface qui permet aux utilisateurs de générer de manière simple des données sans avoir à écrire un fichier de configuration, de plus nous avons considérablement amélioré le générateur existant.

6.2 Apport

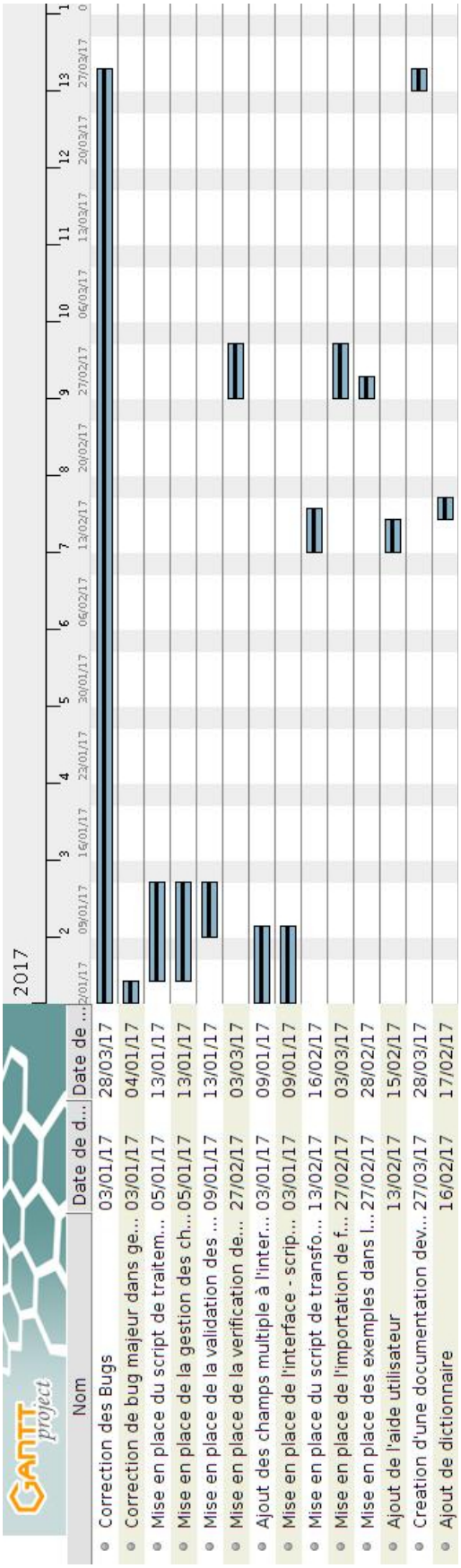
Ce projet nous a beaucoup apporté au niveau de la connaissance des Langages informatiques en effet nous avons pus approfondir considérablement nos connaissance dans les langage du web comme le PHP, le xHTML, le Javascript ou encore l'Ajax.

7 Annexes

7.1 Répartition des tâches

Nom	Rôle par défaut
Brice Harismendy	Chef de projet
Correction des Bugs	
Correction de bug majeur dans generer.php, scindage du code et factorisation du code	
Mise en place de la validation des champs	
Mise en place de la verification des champs avant l'envois	
Mise en place de l'interface - script de gestion dynamique	
Mise en place du script de transformation du sql en XML	
Mise en place de l'importation de fichier de configuration dans l'interface	
Mise en place des exemples dans l'interfaces	
Ajout de l'aide utilisateur	
Ajout de dictionnaire	
Creation d'une documentation developpeur	
Coretin Couvry	Développeur
Correction des Bugs	
Mise en place du script de traitement du formulaire	
Mise en place de la validation des champs	
Mise en place de l'interface - script de gestion dynamique	
Mise en place du script de transformation du sql en XML	
Mise en place de l'importation de fichier de configuration dans l'interface	
Mise en place des exemples dans l'interfaces	
Ajout de l'aide utilisateur	
Geoffroy Berry	Développeur
Correction des Bugs	
Mise en place de la validation des champs	
Ajout des champs multiple à l'interface	
Mise en place de la gestion des champs multiple	
Mise en place de l'interface - script de gestion dynamique	
Mise en place du script de transformation du sql en XML	
Mise en place de l'importation de fichier de configuration dans l'interface	
Mise en place des exemples dans l'interfaces	

7.2 Diagramme Gantt du projet



7.3 Sources

Sources des dictionnaires :

Films : http://www.gogo-films.com/liste-videos.php

Médicaments : https://www.vidal.fr/Sommaires/Medicaments-A.htm

Prenoms : http://www.signification-prenom.net/prenom_francais.htm

Adjectifs : http://www.dramaction.qc.ca/fr/wp-content/files/adjectifs.pdf

http://trem-world.justforum.net/t689-liste-de-traits-de-caracteres

Personnages historiques : https://fr.wikipedia.org/wiki/Liste_des_personnes_d%27importance_historique_nationale

Producteurs : https://fr.wikipedia.org/wiki/Cat%C3%A9gorie:Producteur_de_cin%C3%A9ma_am%C3%A9ricain

Sociétés de production : https://fr.wikipedia.org/wiki/Liste_de_soci%C3%A9t%C3%A9s_de_production_de_cin%C3%A9ma

Réalisateurs : https://www.cineclubdecaen.com/analyse/listealphabetiquerealisateurs.htm

Noms français : http://www.geopatronyme.com/

Langage informatique :

https://fr.wikipedia.org/wiki/Liste_des_langages_de_programmation

Noms d'acteur : http://topcinefilms.free.fr/liste-acteurs-cinema.php

Événement historique : http://keepschool.com/fiches-de-cours/college/histoire/grandes-dates-histoire.html

Dico_scénariste : https://fr.wikipedia.org/wiki/Sc%C3%A9nariste

Métiers : http://www.manpower.fr/metiers

Nationalité - Pays : Wikipédia

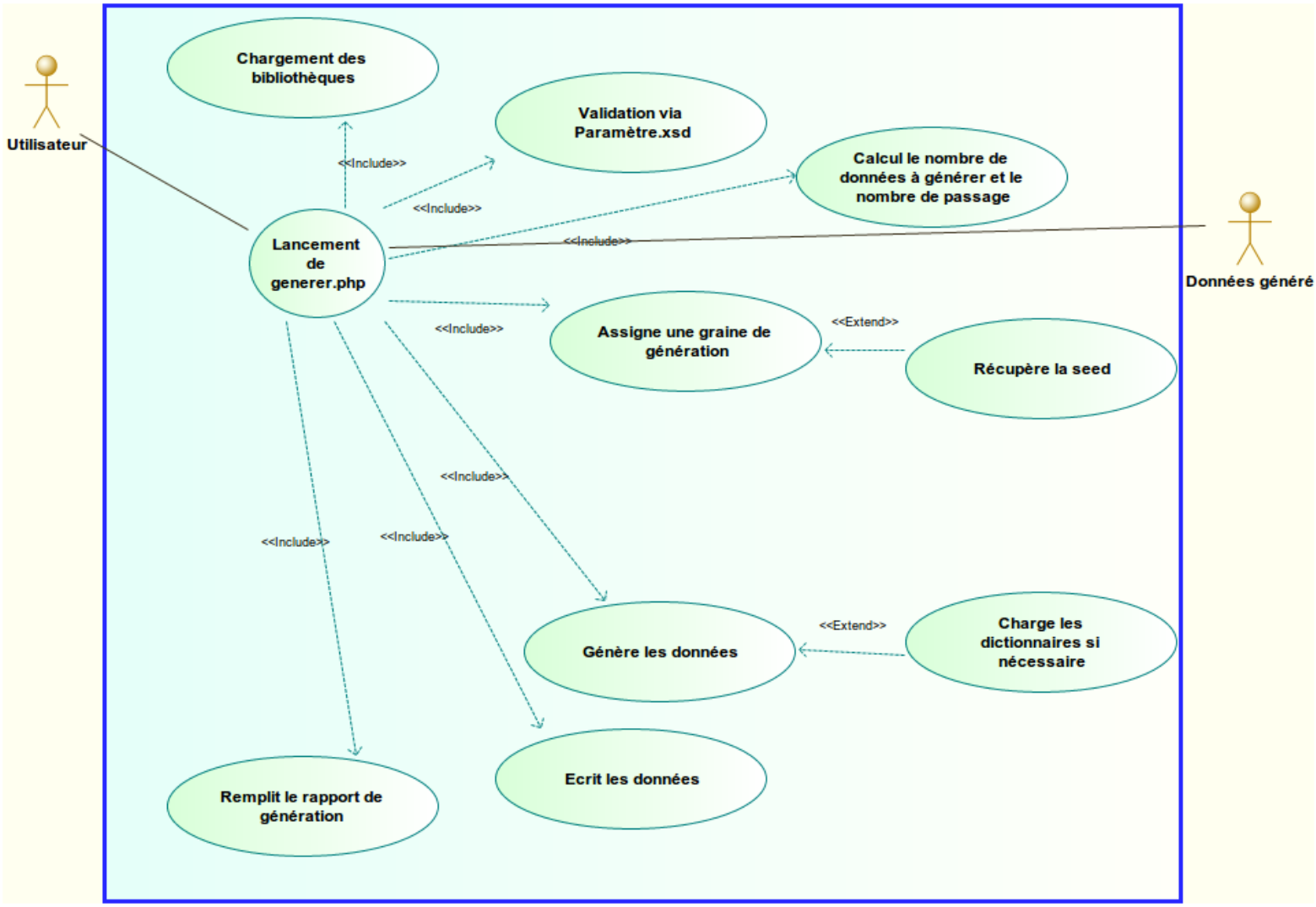
Dico_région/département/ville/pays/société_pharmaceutique : http://www.insee.fr/fr/

Source de formation

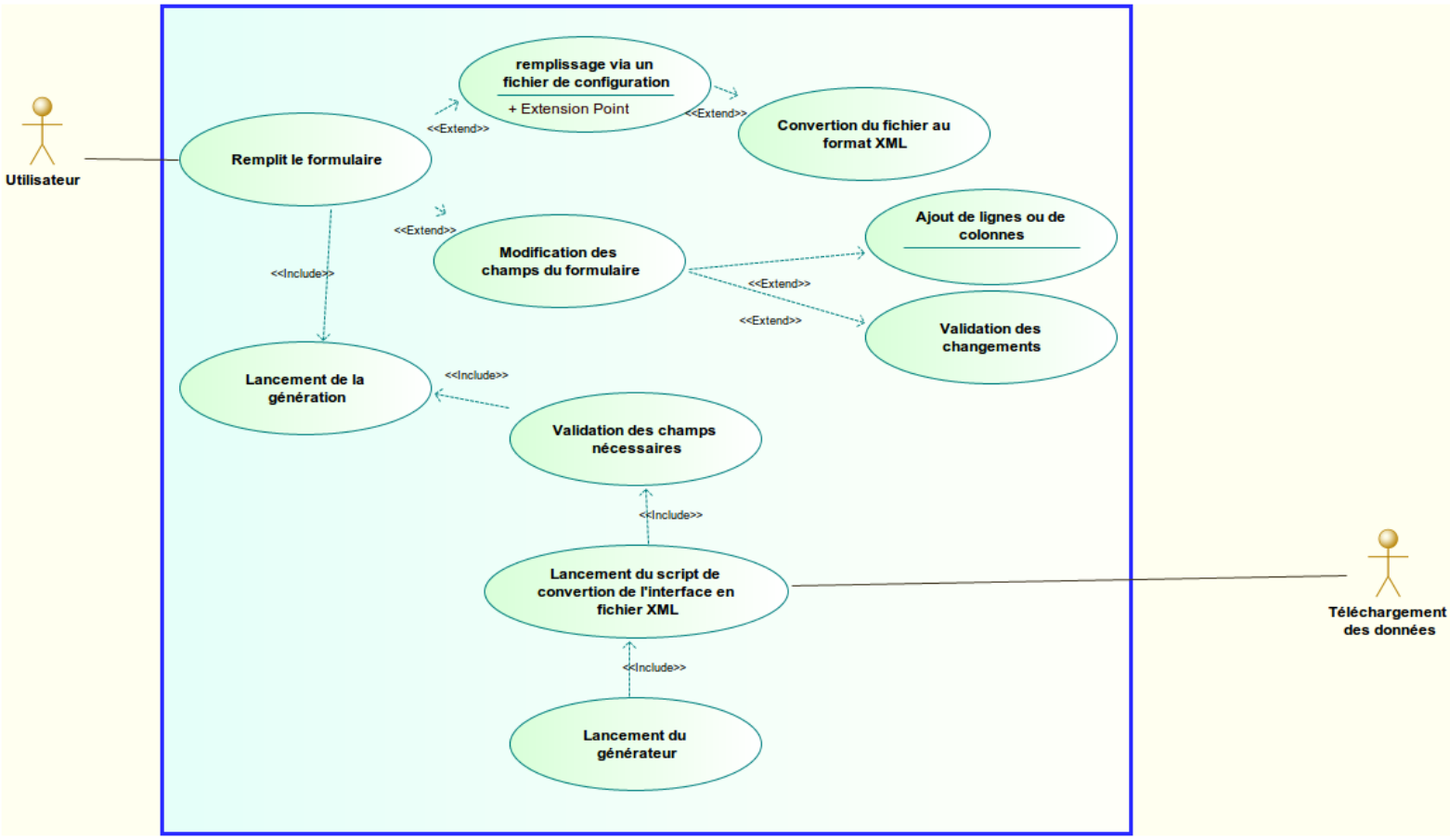
OpenClassroom :https://openclassrooms.com/

Developpez.net : https://www.developpez.net

7.4 Cas d’utilisation du générateur

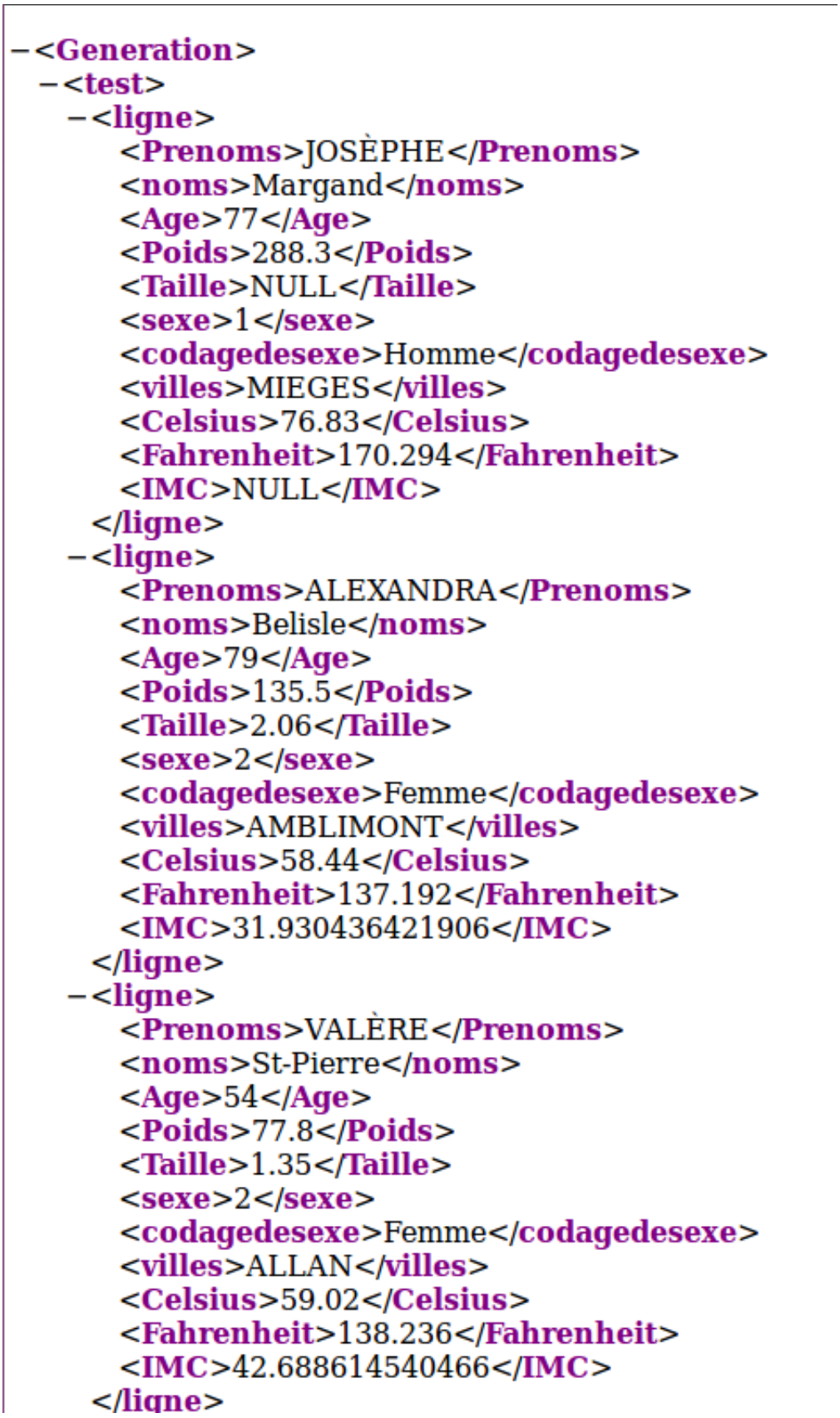


7.5 Cas d’utilisation de l’interface



7.6 Exemple de fichier de sortie

Nous avons plusieurs exemples de format de sortie, tout d'abord le XML :



```
−<Generation>
 −<test>
  −<ligne>
     <Prenoms>JOSÈPHE</Prenoms>
     <noms>Margand</noms>
     <Age>77</Age>
     <Poids>288.3</Poids>
     <Taille>NULL</Taille>
     <sexe>1</sexe>
     <codagedesexe>Homme</codagedesexe>
     <villes>MIEGES</villes>
     <Celsius>76.83</Celsius>
     <Fahrenheit>170.294</Fahrenheit>
     <IMC>NULL</IMC>
   </ligne>
  −<ligne>
     <Prenoms>ALEXANDRA</Prenoms>
     <noms>Belisle</noms>
     <Age>79</Age>
     <Poids>135.5</Poids>
     <Taille>2.06</Taille>
     <sexe>2</sexe>
     <codagedesexe>Femme</codagedesexe>
     <villes>AMBLIMONT</villes>
     <Celsius>58.44</Celsius>
     <Fahrenheit>137.192</Fahrenheit>
     <IMC>31.930436421906</IMC>
   </ligne>
  −<ligne>
     <Prenoms>VALÈRE</Prenoms>
     <noms>St-Pierre</noms>
     <Age>54</Age>
     <Poids>77.8</Poids>
     <Taille>1.35</Taille>
     <sexe>2</sexe>
     <codagedesexe>Femme</codagedesexe>
     <villes>ALLAN</villes>
     <Celsius>59.02</Celsius>
     <Fahrenheit>138.236</Fahrenheit>
     <IMC>42.688614540466</IMC>
   </ligne>
```

au format CSV :

Prenoms	noms	Age	Poids	Taille	sexe	codage de sexe	villes	Celsius	Fahrenheit	IMC
JOSÈPHE	Margand	77	288.3	NULL	1	Homme	MIEGES	76.83	170.294	NULL
ALEXANDRA	Belisle	79	135.5	2.06	2	Femme	AMBLIMONT	58.44	137.192	31.930436421906
VALÈRE	St-Pierre	54	77.8	1.35	2	Femme	ALLAN	59.02	138.236	42.688614540466
NINETTE	Tollmache	58	67.8	1.72	1	Homme	OLMETA-DI-TUDA	32.50	90.5	22.917793401839
RAINIER	Levesque	31	48.6	1.59	1	Homme	OSTHEIM	42.79	109.022	19.223923104308
JACQUETTE	Lafreniere	25	228.3	NULL	2	Femme	CHAMBOIS	20.75	69.35	NULL
WANDA	Bondy	125	55.6	NULL	2	Femme	SAINTE-MERE-EGLISE	57.18	134.924	NULL
ÉLISABETH	Cadieux	74	104.3	NULL	1	Homme	MOULICENT	82.42	180.356	NULL
THÉODORE	Gamelin	112	39.9	1.80	2	Femme	SALLE-ET-CHAPELLE-AUBRY	52.36	126.248	12.314814814815
GAÉTAN	Laforest	39	78.4	1.76	2	Femme	TOURNEUR	37.83	100.094	25.309917355372
CÉCILE	Miron	60	73.7	NULL	2	Femme	CHERES	30.26	86.468	NULL
MADELEINE	Goulet	98	214.9	1.58	1	Homme	SAINTE-SUZANNE-ET-CHAMMES	89.08	192.344	86.083960903701
CLAUDINE	Fugere	34	222.5	1.58	2	Femme	FLETRANGE	14.70	58.46	89.128344816536
ÉMILE	Lafreniere	97	109.1	1.18	1	Homme	BRANTOME EN PERIGORD	96.05	204.89	78.353921286987
GÉRAUD	Bourgeois	17	299.3	1.30	1	Homme	MAGNY	52.44	126.392	177.10059171598
ANATOLE	Grignon	35	260.0	1.50	1	Homme	PLESNOIS	78.34	173.012	115.55555555556
THIBAULT	Sciverit	112	200.5	2.31	1	Homme	BLEURVILLE	39.91	103.838	37.574258353479
HENRI	Martel	76	141.4	1.29	1	Homme	SAINT-MARTIN-SUR-OUANNE	73.34	164.012	84.97085511688
IGNACE	Theriault	64	90.9	NULL	1	Homme	ORNANS	90.39	194.702	NULL
ARIANNE	Louis	NULL	298.5	1.83	2	Femme	MITTELWIHR	44.85	112.73	89.13374540894
PAUL	Therrien	20	NULL	1.63	1	Homme	VALS-LES-BAINS	42.01	107.618	NULL
LAURETTE	de Brisay	60	227.4	NULL	1	Homme	CASTELNAU-DE-MEDOC	84.75	184.55	NULL
ISAURE	Hebert	124	NULL	2.31	1	Homme	ANDIGNE	41.54	106.772	NULL
OLIVIE	Berard	NULL	115.6	1.05	1	Homme	TRANCAULT	80.04	176.072	104.85260770975
PHARAMOND	Huard	111	201.0	2.27	1	Homme	SAN-GAVINO-DI-TENDA	88.54	191.372	39.007161016127
TRISTAN	Berthiaume	67	65.5	1.77	1	Homme	SAINT-PIERRE-DE-MAILLOC	90.02	194.036	20.907146733059
VÉRONIQUE	Grignon	126	41.2	1.79	2	Femme	WUISSE	5.19	41.342	12.858525014825
RAPHAËL	Desroches	92	58.7	2.39	1	Homme	COUME	33.38	92.084	10.276430734756
FRANCIS	LHiver	98	201.6	1.99	1	Homme	ROUGET-PERS	61.96	143.528	50.907805358451
SIMON	LHiver	68	174.9	1.35	1	Homme	BARRO	46.19	115.142	95.9670781893
MELINA	Marcoux	75	145.1	1.81	1	Homme	VAL-DU-LAYON	5.99	42.782	44.290467323952
SIMONE	Brian	45	178.9	1.89	1	Homme	BETTANGE	69.40	156.92	50.082584474119
THIBAULT	Goulet	17	102.1	1.42	2	Femme	AUDIERNE	61.32	142.376	50.634794683595
MAGALIE	Berard	114	63.9	1.61	2	Femme	CLERY-LE-GRAND	92.59	198.662	24.651826704217
LUCAS	Marcil	NULL	52.1	2.26	1	Homme	CHAPELLE-IGER	47.66	117.788	10.200485550944
PASCALINE	Langlais	33	184.0	1.05	2	Femme	CHARTRE-SUR-LE-LOIR	16.08	60.944	166.89342403628
BAPTISTE	Picard	NULL	295.2	1.40	2	Femme	PUECHABON	20.23	68.414	150.61224489796
LUCRÈCE	Drouin	83	223.4	1.71	1	Homme	PONTIGNE	79.31	174.758	76.399575937895
AUDE	Lagrange	93	134.7	1.76	1	Homme	CHEMILLE-EN-ANJOU	92.10	197.78	43.48527892562
ADOLPHE	Therrien	106	137.9	1.90	1	Homme	SAINT-MAUR	25.29	77.522	38.19944598338
AMANDINE	Sevier	107	275.7	1.71	2	Femme	CERVIONE	89.96	193.928	94.285421155227
CONSTANTIN	Drouin	22	238.4	1.84	1	Homme	AMNE	65.48	149.864	70.415879017013
MARTIN	Parenteau	73	252.7	1.28	2	Femme	GEMAGES	94.18	201.524	154.23583984375
CÉSAIRE	LaCaille	126	143.2	1.12	1	Homme	BOUCHET-MONT-CHARVIN	26.60	79.88	114.15816326531
JOSUÉ	Bilodeau	105	72.1	1.33	1	Homme	BLAESHEIM	50.86	123.548	40.759794222398
BARTHÉLÉMY	Guilmette	76	94.8	1.11	2	Femme	HUNSPACH	28.63	83.534	76.941806671536
EVE	Brian	31	NULL	1.57	2	Femme	LACANAU	24.83	76.694	NULL
IRIS	Rossignol	86	119.5	1.73	2	Femme	SAINT-LAURENT-L'ABBAYE	28.35	83.03	39.927829195763
LUCINDE	Dumont	22	234.2	2.15	1	Homme	HURTIGHEIM	51.50	124.7	50.665224445646
PADRIG	Leblanc	81	244.0	1.28	1	Homme	MILLY	94.18	201.524	148.92578125

et enfin au format SQL :

```sql
DROP TABLE IF EXISTS test2;
CREATE TABLE test2 (
id mediumint(8) unsigned NOT NULL auto_increment,
Prenoms VARCHAR(255),
noms VARCHAR(255),
Age NUMERIC(9),
PRIMARY KEY (`id`)
) AUTO_INCREMENT=1;

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('GHYSLAINE','Margand',42);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('JOSÉPHINE','Carignan',67);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('SYLVIANNE','Batard',30);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('FRANCIS','Leblanc',73);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('LILIANE','Denis',NULL);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('LAURE','Pepin',41);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('PATRICE','LHiver',54);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('PHILIPPE','Doucet',57);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('LÉOPOLDINE','Chicoine',39);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('YVES','Bellemare',22);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('YANICK','Querry',71);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('AURÈLE','Givry',109);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('STÉPHANIE','LHiver',112);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('BAPTISTE','Charlebois',56);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('ANNABELLE','Champagne',58);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('MAGALIE','Doiron',77);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('JULIEN','Asselin',125);

INSERT INTO test2 (Prenoms,noms,Age) VALUES ('ANNICK','Boulanger',117);
```

7.7 Rapport de génération

Voici le contenu d’un rapport de génération :

```
La génération du jeudi 30 mars 2017 a durée : 0.03 secondes
==========================================
Table n°1
==================================================
Données Générales :
La seed de génération est : 0413221995
==================================================
Prenom :
Le nombre de Valeur null est de : 0
Le pourcentage de valeur null est de : 0 %
==================================================
Sexe :
==================================================
Noms :
Le nombre de Valeur null est de : 0
Le pourcentage de valeur null est de : 0 %
==================================================
Villeactuelle :
Le nombre de Valeur null est de : 0
Le pourcentage de valeur null est de : 0 %
==================================================
CodePostale :
==================================================
```

7.8 Fichier de configuration XML

```
- <Generateur>
  - <Table>
    - <Champs>
        <Donnee Type="Dictionnaire" NomColonne="Prenoms" NomDictionnaire="Prenoms" ModeGeneration="uniforme" GenererDependance="True"/>
        <Donnee Type="Dictionnaire" NomColonne="noms" NomDictionnaire="noms" ModeGeneration="uniforme"/>
        <Donnee Type="Numerique" NomColonne="Age" Min="15" Max="130" ModeGeneration="uniforme" Null="6%" Unite="ans"/>
        <Donnee Type="Numerique" NomColonne="Poids" NbDecimale="1" Min="35.1" Max="300.9" ModeGeneration="uniforme" Null="52"/>
        <Donnee Type="Numerique" NomColonne="Taille" NbDecimale="2" Min="1.01" Max="2.40" ModeGeneration="uniforme" Null="6%"/>
        <Donnee Type="Dictionnaire" NomColonne="villes" NomDictionnaire="villes" ModeGeneration="uniforme"/>
        <Donnee Type="Numerique" NomColonne="CodeSexe" NomPerso="sexe" Min="1" Max="5" ModeGeneration="uniforme" codage="1;Homme;2;Femme"/>
        <Donnee Type="Numerique" NomColonne="Temperature" NomPerso="Celsius" NbDecimale="2" Min="0.00" Max="100" ModeGeneration="uniforme"/>
        <Donnee Type="Formule" NomColonne="Fahrenheit" Formule="Celsius 1.8 + 32"/>
      </Champs>
    - <Parametre>
        <Sortie XML="True" CSV="True"/>
        <Nbligne valeur="1000"/>
        <NomTable nom="test"/>
        <Seed valeur="123"/>
      </Parametre>
    </Table>
  - <Table>
    - <Champs>
        <Donnee Type="Dictionnaire" NomColonne="Prenoms" NomDictionnaire="Prenoms" ModeGeneration="unique"/>
        <Donnee Type="Dictionnaire" NomColonne="noms" NomDictionnaire="noms" ModeGeneration="uniforme"/>
        <Donnee Type="Numerique" NomColonne="Age" Min="15" Max="130" ModeGeneration="uniforme" Null="6%"/>
      </Champs>
    - <Parametre>
        <Sortie CSV="True" SQL="True"/>
        <Nbligne valeur="1000000"/>
        <NomTable nom="test2"/>
        <Seed valeur="58125"/>
      </Parametre>
    </Table>
  </Generateur>
```