Projet de programmation en Licence professionnelle

Université d'Angers

Générateur universel de données pour tests logiciels

Tuteur

Gilles Hunault

Étudiants

Chef de projet : Brice Harismendy

Corentin Couvry

Alban Baumard

Antoine Legoubé

Sommaire

1 – 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.

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.

Dans un premier temps d'analyse, on viendra recenser les types de données simples et structurées à générer, comme par exemple des colonnes de noms de personnes et d'âges, ou comme des clés primaires et des clés étrangères pour deux tables de données. A la suite de cette analyse, on décidera de la structure XML du fichier de paramètres pour configurer les données à générer.

Dans un deuxième temps, on viendra à la fois générer et analyser (sommairement) les données produites. Par exemple on pourra compter les nombres de personnes de chaque sexe dans les données générées, ou produire le code MySQL pour comptabiliser les données. Il n'est pas sans doute pas indispensable de tout réaliser mais par contre, essayer de tout prévoir en termes de conception de données est une partie importante du projet.

Il peut s'agir d'un sujet d'ampleur, car on peut potentiellement générer de très grosses structures, comme des stocks d'entreprise, des annuaires d'associations, des arbres binaires ou n-aires, des graphes pour des benchmarks logiciels... On prévoira donc de nombreux formats de sorties, dont les formats TXT, CSV[2], XML et SQL.

L'utilisation du générateur, contrairement au site generatedata, se fera à la fois en ligne de commandes et via une page Web, même si la génération pourra produire des fichiers HTML, Javascript etc. On pourra s'inspirer du site generatedata mais les données devront impérativement être plus "francisées", avec des accents éventuels, parfois des prénoms doubles, avec si possible des lois de distribution plus sophistiquées qu'un simple tirage pseudo-aléatoire uniforme...

On prévoira aussi des valeurs manquantes, en nombre absolu ou en pourcentage. On commencera bien sûr par écrire la version en ligne de commandes, plus facile à développer et à tester.

Tout le projet sera réalisé en PHP.

2 – Analyse

2.1 Projet

Après avoir pris connaissance de l'énoncé du projet, donné par notre tuteur, nous avons défini en sa compagnie une méthodologie pour réaliser ce projet en plusieurs étapes :

-L'analyse de l'existant
-La recherche de type de données intéressantes à générer
-Le regroupement par type de données (Numérique/Alphanumérique)
-Le regroupement par forme (Simple/Multiple)
-Structuration du fichier XML de paramétrage
-Développement du Générateur
-Génération d'un “rapport de génération”

Nous allons donc générer des données et la question du nombre de tuples est importante. Notre tuteur a décidé de fixer l'objectif d'une génération à 100 000 tuples. Ce qui nous donne une avance sur les sites de génération de données existants qui limitent et/ou font payer les utilisateurs voulant créer un plus grand nombre de valeurs (cf - 2.2 Existant). Les données devront être cohérentes, cela implique la mise en place de contraintes (cf - 2.3 Regroupement et organisation des données)(pour l'âge d'une personne, par exemple, un maximum de 130 ans, ou un poids minimum de 30 kg, etc…).

2.2 Existant

Pour cette première analyse de l'existant et suite à la réunion du 17/10/2016 avec notre tuteur, nous avons fait un état des lieux des sites existants de générateurs de données afin de prendre connaissance des solutions déjà proposées. Nous sommes parvenus à l'étude et l'analyse des sites Web suivants : “Generate Data”, “Mockaroo”, “Yan data ellan”, “Free Data Generator” (cf : Annexe Sources - Les différents générateurs de données existant).

Generate Data : Ce site propose plusieurs formats de sortie mais un petit nombre de tuples pouvant être générés (100). Bien-sûr, une version payante est disponible, à raison de 20 euros / ans, grâce à laquelle, vous pourrez générer jusqu'à 5000 tuples.
Les tuples générés pour les noms et prénoms sont uniquement anglophones et la possibilité de rentrer des valeurs « null » n'existe pas.

Mockaroo : « Realistic data generator ». Ce petit nouveau a émergé en 2015 et propose une génération de données très complète. Une possibilité d'insertion de valeurs « null » en pourcentage, un grand nombre de types de données, une interface simple, des ajouts de contraintes et une communauté active, en revanche, le site est exclusivement anglophone.
La génération de tuple est limitée à 1000, pour une quantité supérieur, une première version payante est disponible à raison de 50 euros / ans pour 100 000 tuples, et 500 euros / ans pour 10 Millions.

Ensembles, nous avons dressé un tableau comparatif des principaux sites trouvés, en notant les bons points, les mauvais et les fonctionnalités manquantes. (cf. Annexe -Tableau comparatif des différents sites existant).

En conclusion : Pour un nombre important de tuples à générer, ces sites propose une version payante de génération. La présence de francophonie dans les données générées, où même dans la rédaction des sites, est peu présente. La diversité des types de données est relativement faible, à l'exception de Mockaroo.

Notre objectif est de créer un générateur qui se démarque des solutions existantes en ajoutant des fonctionnalités, par exemple un rapport de génération détaillé pour connaître les pourcentages et le nombre de données ayant pour valeur « NULL», la création gratuite de tuples supérieurs à 100 000 pour permettre aux utilisateurs de créer un grand nombre de données, la génération de « NULL » en valeur fixe ou en pourcentage tout en prenant en compte le fait qu'il ne puisse pas y avoir 2 « NULL » sur la même ligne, la mise en place d'un système de codage pour qu'à partir d'une colonne généré (exemple : pour le sexe : 1 → homme, 2 → femme, au choix de l'utilisateur), la création d'une génération multi-tables pour délimiter les groupes de données créées, la gestion des clés primaires et secondaires pour pouvoir lier les données entrent-elles, et pour finir, permettre un large choix de types de données.

2.3 Regroupement et organisation des données

Afin de pouvoir proposer une large diversité de résultats. Nous avons cherché à réunir une grande quantité de type de données,. Pour cela nous avons donc décidé de commencer par classifier* ces types selon quatre catégories :

2.3.1 Numériques

C'est toutes les données ne comportant que des chiffres que nous avons regroupés sous ce type. Elles ont des caractéristiques communes : un minimum, un maximum, un nombre de « null », et une génération similaire. En effet on peut générer les numériques de manière séquentielle, aléatoire, par formule ou en utilisant une loi mathématique (Normale, Binomiale, Poisson).

2.3.2 Alphanumériques

Le type alphanumérique regroupe, quant-à-lui, les données comportant des chiffres et des lettres ou des symboles (: , / , ; , …) comme par exemple les molécules ou les adresses IPv4 et IPv6. Ils peuvent avoir les mêmes paramètres que les numériques.

2.3.3 Dictionnaires

En ce qui concerne les dictionnaires, il regroupe un ensemble de données centrés sur une et une seule catégorie. Par exemple Robert et Élisabeth sont dans la catégorie “Prénoms”. Ainsi on peut générer des noms cohérents, pour cela nous avons dû créer plusieurs dictionnaires en nous appuyant sur des données réels (cf : Annexe - Sources des dictionnaires).

2.3.4 Formes simples et multiples

On définit les types de données en deux formes, la simple pour les données ne nécessitant qu'un champ (exemple : prénoms, noms, âge, …) et la forme multiple qui nécessite plusieurs champs à la génération comme les grilles de résultats de golf ou des adresses.

(* : Le détail de ces regroupements est disponible sous forme de tableau à l'annexe - 7.2 Extrait du regroupement de données numérique/7.3 Extrait du regroupement de données alphanumérique).

3 – Réalisation

3.1 XML et XSD

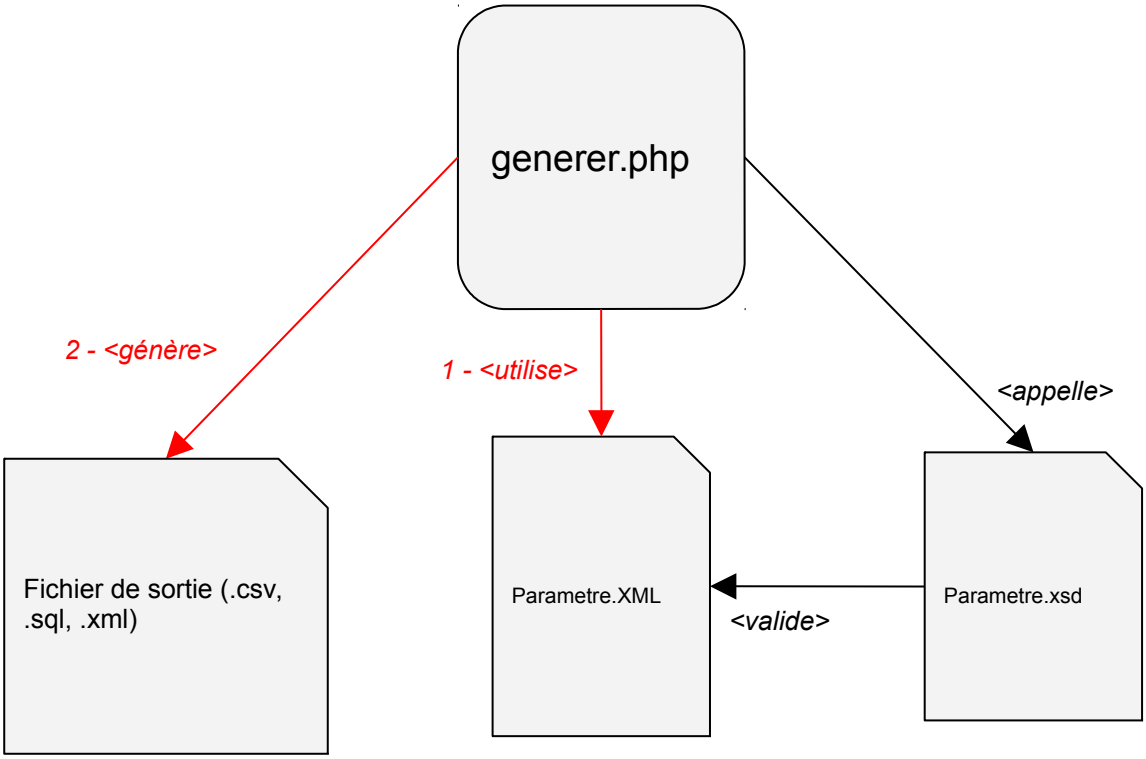
Pour débuter la réalisation de ce projet nous devions tout d'abord nous pencher sur une des fonctionnalités principal citée dans le cahier des charges : “La configuration de la génération sera décrite dans un fichier XML”. Le XML étant un langage de balisage extensible qui permet de définir différents espaces de noms.
À partir de ce moment, nous avons étudié la technologie XSD que nous connaissions peu. Après cette étape nous avons décidé, en commun avec notre tuteur, que nous nous occuperons du XML, tandis que d'un autre côté il nous serait fourni le schéma 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. Suite à la création de ce XML il nous reste encore à récupérer les données demandées ainsi que leurs paramètres via notre script PHP. (cf Annexe - 7.1 Documentation du fichier de paramétrage)

3.2 PHP

Comme il nous l’a été précisé dans le cahier des charges, nous devions réaliser le projet en PHP. Nous avons donc respecté cette demande, en réalisant 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 -7.8 Exemple de fichier de sortie).

Pour cela nous récupérons les balises XML, ainsi que les attributs contenues dans les balises pour appeler les bonnes méthodes de générations, cela nous permet une génération multi-tables avec des paramètres différents (type de sortie, nombre de ligne).

3.3 Fonctionnement



Légende :
→ : Etape 1 - Lors de la lecture du fichier.XML.
→ : Une fois l'étape 1 validée. L'utilisation de *Paramètre.XML* (1 - <utilise>) se fait en amont de la génération (2 - <génère>).

Ci-dessous, une représentation schématique du fonctionnement de notre script generer.php.
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".

Pour la suite des opérations, on utilise en amont le fichier *Parametre.xml*, qui fixe, entre autre, les contraintes, le nom des types, les formules et les modes de générations. Puis le script PHP créer le fichier de sortie au format voulu.

Nous avons créé des fonctions permettant une sélection aléatoire d'un nombre ou d'une chaîne de caractères depuis un dictionnaire, nécessaire pour la génération.

Pour que l'utilisateur est plus de libertés, nous avons prévu un système de formules. À ce stade, nous proposons deux différentes formules de types :

a x → les variables a et x sont choisie, soit avec une valeur, soit avec une donnée d'une table. l'utilisateur doit respecter la syntaxe (un espace doit séparer chaque variable ou opérateur). Si il manque une des variables, elle est remplacé par une valeur par défaut.

a x + b → Ici aussi, les variables a, x, b sont choisie, de la même manière que pour a x. La syntaxe reste la même (un espace doit séparer chaque variable ou opérateur). Les opérateurs pris en charges sont : '+', '-', '/', '%', '*'.

La génération de données (peu importe le format de sortie) se fait par paliers de 100000 valeurs. On a fait ce choix de conception pour une meilleure vitesse de génération.

Pour la génération multi-tables, un fichier est assigné à une table. Par exemple, si l'on veut trois tables, le script va nous générer trois fichiers lorsqu'on est au format CSV, un seul pour le format XML et SQL.

Suite à la génération, un rapport est fait contenant : le nombre de valeurs « null », et leurs pourcentages, le nombre de valeurs demandées, la valeur la plus haute possible et la plus basse, une moyenne, ainsi que l'unité des valeurs.

Exemple de rapport de génération pour l'âge

Age :
Les données sont exprimées en ans
Le nombre de valeur null est de : 60
Le pourcentage de valeur null est de : 6 %
La valeur minimale est de : 15
La valeur maximal est de : 130
La valeur moyenne est de : 73.97

Note : Exemplaire complet disponible en Annexe -7.5 *Rapport de génération*

4 – Améliorations possibles

Le temps qui nous a été alloué n'étant pas suffisant pour aboutir à une version très complète du générateur de données nous pouvons entrevoir différentes améliorations possibles.

La gestion des erreurs utilisateur, en effet si l'utilisateur entre des mauvaises valeurs. Exemple au niveau des formats de sortie : CSV = “TRUE” au lieu de CSV=“True”, le programme ne générera pas le fichier CSV en sortie. Donc une première amélioration possible serait de rendre les valeurs rentrées par l'utilisateur insensible à la casse.

Au moment de la génération nous créons un rapport décrivant ce que nous avons généré et comment. Actuellement ce rapport n'est pas effacé avant de re-générer des tables et le nouveau rapport est donc concaténé au rapport précédent. Idéalement il faudrait vérifier si le fichier existe déjà et le vider dans ce cas.

Autre point problématique, la génération d'un rapport pour plus de 100 000 tuples. Actuellement nous avons une erreur sur le rapport lorsque nous voulons plus de valeurs. Pour temporairement résoudre ce problème, nous ne produisons pas le rapport quand la génération excède 100 000 tuples.

La présence des clés primaires et secondaires était l'une des fonctionnalités manquante sur les autres sites que nous voulions implémenter. La durée du projet étant limitée, nous n'avons malheureusement pas eu le temps de développer cette option mais, nous avons réfléchi à un moyen de l'implémenter plus tard et pensons y arriver rapidement.

Les formules ont également besoin d'améliorations comme la gestion des priorités qui seraient utiles pour des calculs comme l'IMC. Actuellement les formules sont de formes ax+b ce qui fait que le programme calcul d'abord ax puis fait la partie +b.

Le problème du système de codage est que l'on doit assigner pour chaque valeur son correspondant il faudrait donc pouvoir gérer des groupes de valeurs en faisant des formules de codage de la forme : 1;Homme;>;Femme. De cette façon toutes les valeurs supérieurs à 1 prendront comme valeurs Femme.

L'enrichissement du nombre de dictionnaires : le nombre de dictionnaires supportés est assez restreint (noms,prénoms,villes) il faut donc mettre en place un plus grand nombre de dictionnaires .

Amélioration du rapport : le rapport est déjà assez complet mais des données supplémentaires pourrait facilement y être ajouté comme le temps de génération les formats de sortie …

Une autre amélioration majeure du rapport serait la gestion d'option pour ce dernier dans le fichier XML, pour qu'il ne comporte que ce qu'on désire.

L'augmentation des formats de sortie : en effet l'état des lieux nous a permis de voir que les solutions existantes proposais un grand nombre de format comme le javascript le JSON, le Perl … Au niveau des formats on pourrait complexifier les formats comme le SQL en intégrant le Postgres, Oracle…

Une possibilité d'amélioration concernent les base de données. En effet une entreprise a soumis l'idée qu'à partir d'un squelette de base de données (contenant le code SQL de création des tables) on puisse générer des données au format SQL.

5 – Organisation du projet

Afin de structurer au mieux notre équipe nous avons décidé de choisir un chef de projet, Brice s'étant spontanément présenté nous avons tous donné notre accord. Nous nous sommes ensuite demandé quels seraient les meilleurs moyens d'organiser et de gérer notre projet.

Pour la partie organisation il fut d'abord proposé de lister les tâches du projet, celles en cours ainsi que celles terminées, Trello un logiciel de gestion de projet nous semblait tout à fait adapté à notre idée première. Nous avons donc créé un “Board” et commencé à nous attribuer les différentes tâches. Ensuite il nous fallait un espace de stockage pour les documents et rapports sur lesquels nous devions travailler en commun, nous avons donc séparé nos fichiers en deux parties, les recherches et rapports du projets seraient sur un “google drive” tandis que nous avons donc utilisé le dépôt gitlab installé sur notre serveur pour nos codes.
La séparation des tâches se fit naturellement et nous en sommes venu à un diagramme des tâches de type Gant (cf annexe -7.6 Répartition des tâches/).

Pour finir lorsqu'un d'entre nous avait des problèmes pour avancer dans sa tâche on se mettait à plusieurs pour l'aider

6 – Conclusion

6.1 État des faits

À la fin de ce temps de projet nous avons réalisé un générateur assez complet qui répond en grande partie à l'énoncé fournit par notre tuteur au début du projet. Cependant quelques spécificités ont été abandonnées comme la sortie au format texte. Notre programme fonctionne contrairement à beaucoup de ses concurrent en ligne de commande.

En ce qui concerne les fonctionnalités implantées nous pouvons générer des numériques symboliques ou non (comportant un codage) entre des bornes minimum et maximum, avec un nombre de décimales et la gestion des « null ». Il n'y a qu'un mode de génération (Aléatoire). On peut également générer des données via Dictionnaire avec également la gestion des « null ».

6.2 Apport

Ce projet nous a beaucoup apporté au niveau de la connaissance des langages informatiques. En effet certains d'entre nous via leurs parcours n'avaient jamais programmer en PHP ou utiliser de XML et fait de validation par un XSD. De plus nous avons beaucoup appris à travailler en équipe et à nous organiser pour avancer sur nos temps libres. Enfin nous avons particulièrement apprécié le sujet du projet et ainsi nous avons pris plaisir à travailler dessus donc nous sommes intéressés pour faire la suite qui se déroulera en janvier.

7 Annexes

7.1 Documentation du fichier de paramétrage

Le fichier est encadré par les balises <Generateur></Generateur>.
Pour chaque nouvelle table on utilise la balise <Table></Table>.
Pour chaque table il y a 2 partie pour les configurer :
-la déclaration des champs
-la définition des réglages

7.1.1 La déclaration des champs

Pour chaque table il faut déclarer les champs entre les balises <Champs></Champs>
entre ces balises on va déterminer les données grâce à la balise <Donnee />
la configuration se fait via les attributs :
-“Type” qui prend en valeur “Dictionnaire”, “Numerique”, “Formule” ou “IMC” (ce dernier est temporaire et permet de calculer l'IMC et sera enlevé lorsque le type formule sera plus poussé)
- “NomColonne” qui prend comme valeur le nom de la données que l'on veut générer (“Noms”, “Prénoms”, “Age”,....)
- “NomPerso” prend comme valeur un nom choisi par l'utilisateur.
- “ModeGeneration” prend pour valeur “unique”, “uniforme”(aléatoire), “Normale” ou “Binomiale”. attention pour le moment le mode de génération de toutes les données sont générées de manière aléatoire.
- “Null” prend une valeur en pourcent ou en entier et permet de remplacer un certain nombre de données par “Null” (ex : “6%”, “60”).
--pour les numériques il y a 4 autres attribut :
-Tout d'abord il y a “Min” et “Max” qui permettent de donner des bornes au nombre généré. De plus si on veut générer des nombres à virgule on utilise l'attribut “NbDecimale”
- ensuite il y a l'attribut “codage” qui sert à créer une colonne partir de la colonne générer elle se comporte comme suit : “valeur;donnéesDeRemplacement”.
--Pour le type Formule :
-Un attribut “Formule” permet de mettre une formule de la forme “ax+b”

7.1.2 La définition des réglages

pour définir les réglages il y a 4 balises :

-<Seed/> qui prend pour attribut “valeur” qui lui contient une seed de génération si cette balise est absente une seed est généré, exemple :
<Seed valeur="123" />.

-<Sortie /> qui contient les formats de sortie (CSV/SQL/XML) si on veut par exemple avoir une sortie au format CSV on met : “CSV="True"”,exemple :
<Sortie CSV="True" SQL="True"/>.

-<Nbligne/> qui a un attribut “valeur” qui contient le nombre de ligne que l'on souhaite pour cette table, exemple <Nbligne valeur="1000000"/>.

-<NomTable/> qui a pour attribut “nom” qui permet de nommer la table, exemple : <NomTable nom="test2"/> .

Exemple de fichier valide :

```
<?xml version="1.0" encoding="UTF-8"?>
<Generateur>
 <Table>
        <Champs>
            <Donnee Type="Dictionnaire" NomColonne="Prénoms"
ModeGeneration="unique"/>
            <Donnee Type="Dictionnaire" NomColonne="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="520" />
        </Parametre>
    </Table>
</Generateur>
```

7.2 Extrait du regroupement de données Numérique

								mode de génération			
nom du type	description	Label	null	unité	min inclus	max inclus	nb décimal	Séquentielle	Modèle	Dictionnaire	Formule
Précipitation	Quantité d'eau tombée	Choix utilisateur	% | nb	Millimètre	0	500	0	----------	min;max (loi uniforme), aléatoire	----------	----------
Expérience	expérience acquise dans un jeu	Choix utilisateur	% | nb	xp	0	1000000	0	de 0 à 100 000 avec un pas de 1000	min;max (loi uniforme), aléatoire	----------	moyenne expérience par niveau = expérience / niveau max
Défense	Points de défense d'un joueur	Choix utilisateur	% | nb	Points	0	1000	0	0 à 1000 avec un pas de 100	min;max (loi uniforme), aléatoire	----------	moyenne de défense = defense / nb de piece de stuff
Agilité	Points d'agilité d'un joueur	Choix utilisateur	% | nb	Points	0	1000	0	0 à 1000 avec un pas de 100	min;max (loi uniforme), aléatoire	----------	agilité = niveaux *0.5
Intelligence	Points d'intelligence d'un joueur	Choix utilisateur	% | nb	Points	0	1000	0	0 à 1000 avec un pas de 100	min;max (loi uniforme), aléatoire	----------	intelligence = niveau *2
Réparation(vehicule)	Nombres de réparation d'un véhicule	Choix utilisateur	% | nb	Réparations	0	150	0	0 à 150 avec un pas de 1	min;max (loi uniforme), aléatoire	----------	accident = réparation * 0.90
Jetons (poker)	Nombres de jetons	Choix utilisateur	% | nb	Jetons	0	100000000	0	0 à 100 000 000 avec un pas de 1	min;max (loi uniforme), aléatoire	----------	nombre de plaques = nombre de jeton / 1000
nb PC (informatique)	Nombre de PC dans un parc informatique	Choix utilisateur	% | nb	PC	0	5000	0	0 à 5000 avec un pas de 1	min;max (loi uniforme), aléatoire	----------	nbpcpanne = nb_pc*0.03
Mise (poker)	Quantité d'argent pariée au poker	Choix utilisateur	% | nb	Jetons	1	100000	0	0 à 100 000 avec un pas de 1	min;max (loi uniforme), aléatoire	----------	mise = jetons
Tirs (Laser game)	Nombre de Tirs	Choix utilisateur	% | nb	Tir	0	1000	0	0 à 1 000 avec un pas de 1	min;max (loi uniforme), aléatoire	----------	tirs allié = tirs *0.20
Score equip	Nombre de points de l'équipement	Choix utilisateur	% | nb	GS (gear score)	0	10000	0	0 à 1 000 avec un pas de 100	min;max (loi uniforme), aléatoire	----------	score = somme(score_allié)

Nous avons regroupé les données de type numérique en définissant plusieurs colonnes, nom du type, description, label, null, unité, min inclus, max inclus, nb décimal, et le mode de génération (Séquentielle, Modèle, Dictionnaire, Formule).

7.3 Extrait du regroupement de données Alphanumérique

								mode de génération			
nom du type	description	Label	null	nombre de valeurs	Alphabet	min (inclus ?)	max (inclus?)	Séquentielle	Modèle	Dictionnaire	Formule
Type d'alimentation moteur (véhicule)	Définie l'alimentation d'un moteur (diesel gazoil essence)	Choix utilisateur	% | nb	8	[1-8]	1	8	----------	Aléatoire uniforme	----------	1 = diesel, 2= gazoil 3 = essence 4= electrique 5= essence 6 = GPL 7= hydrogène 8 = Kérosène
Nom	Patronyme d'une personne	Choix utilisateur	% | nb	10000	[A-Z]	----------	----------	----------	Aléatoire uniforme	noms_francais	----------
Minéraux (données alimentaire)	Minéraux contenu dans une boisson	Choix utilisateur	% | nb	9	[1-9]	1	9	----------	Aléatoire uniforme	----------	1= Calcium ; 2 = Chrome: 3 = Iode ; 4 = Fer; 5 = Magnésium ; 6 = Manganèse ; 7 = Potassium ; 8 = Sélénium ; 9 = Zinc
Collier	Nom d'un collier	Choix utilisateur	% | nb	>10	[A-Z]	----------	----------	----------	Aléatoire uniforme	nom_collier_rpg	----------
Anneau	Nom d'un anneau	Choix utilisateur	% | nb	>10	[A-Z]	----------	----------	----------	Aléatoire uniforme	anneau_rpg	----------
Nom de regiment	Nom du régiment du joueur	Choix utilisateur	% | nb	>10	[A-Z]	----------	----------	----------	Aléatoire uniforme	nom_regiment_rpg	----------
Nom de monde	Nom du monde dans lequel joue le joueur	Choix utilisateur	% | nb	>10	[A-Z]	----------	----------	----------	Aléatoire uniforme	nom_monde_rpg	----------

Pour les données de type alphanumérique nous avons également défini plusieurs colonnes qui sont le nom du type, description, label, null, nombre de valeurs, Alphabet (l'alphabet utilisé), min inclus, max inclus, et enfin les mode de génération (Séquentielle, Modèle, Dictionnaire, Formule).

7.4 Tableau comparatif des différents site existant

	Generate Data	Mockaroo	yan data ellan	Free Data Generator
Bien	Plusieurs formats de sortie	Possibilité de valeurs NULL en %	Très grand nombre de tuples de base (10000).	Génération selon différentes options de langage
	Donnée personnalisé pour plusieurs pays	Nombre de type	Plusieurs formats de sortie	Création de plusieurs tables possible
		Simplicité de l'interface	Donnée personnalisé pour plusieurs pays	
		Communauté et Forum actif		
		Génération d'image		
		Type de données personnalisables		
		Aide bien fournie et documentée		
		Existence de contraintes sur les types de champs		
		Plusieurs formats de sortie		
Mauvais	Nombre de type restreint	Limite de 1000 tuples par download en version gratuite	Peut de type de données différents	Peu d'aide / informations
	Anglophone	Anglophone		Interface peu attractive et peu intuitive.
	Aide utilisateurs trop sommaire			
	limitation à 100 tuples en version gratuite			
Manquant	Possibilité de valeur NULL	noms de famille francophones (les prénoms y sont présent)	création de son propre type	création de son propre type
	Les contraintes complexes	Francophonie	plus de type de données	Manque de certaines données
	Clé (primaire/étrangère)		contrainte complexes	francophonie
	Francophonie			contrainte complexes
payant	OUI	OUI	OUI	non

Tableau récapitulant les bons et mauvais points des sites existant pour générer des données nous avons également listé les points manquants et indiqué la gratuité de chacun d'entre eux.

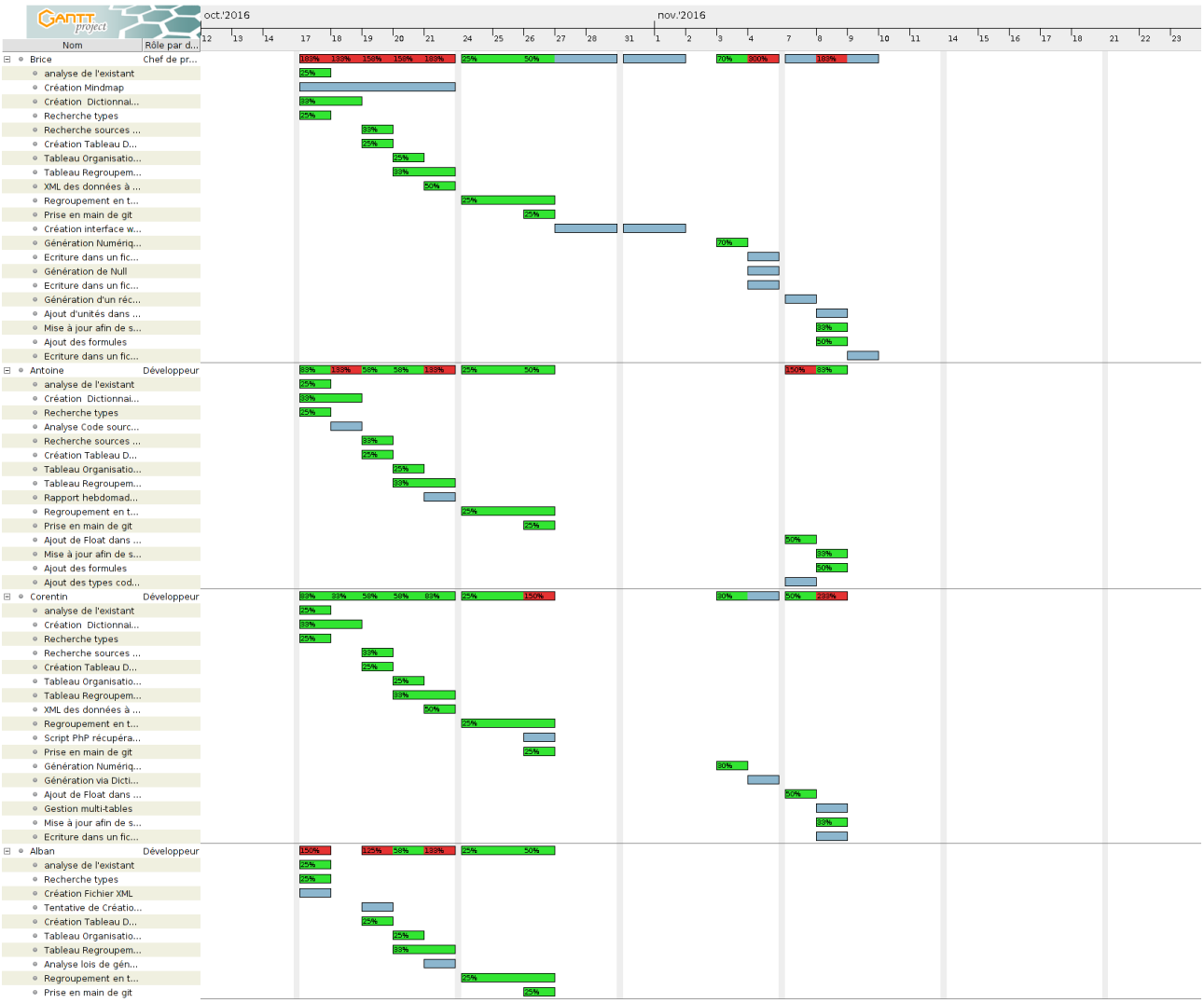
7.5 Rapport de génération

Voici un rapport de génération complet :

```
Table n°1
================================================
Prénoms :
Le nombre de Valeur null est de : 0
Le pourcentage de valeur null est de : 0 %
================================================
Age :
Les données sont exprimer en ans
Le nombre de Valeur null est de : 60
Le pourcentage de valeur null est de : 6 %
La valeur minimale est de : 15
La valeur maximale est de : 130
La valeur moyenne est de : 74.10
================================================
Données Générales :
La seed de génération est : 2300
================================================
Table n°2
================================================
Prénoms :
Le nombre de Valeur null est de : 0
Le pourcentage de valeur null est de : 0 %
================================================
noms :
Le nombre de Valeur null est de : 0
Le pourcentage de valeur null est de : 0 %
================================================
Données Générales :
La seed de génération est : 46875
================================================
```

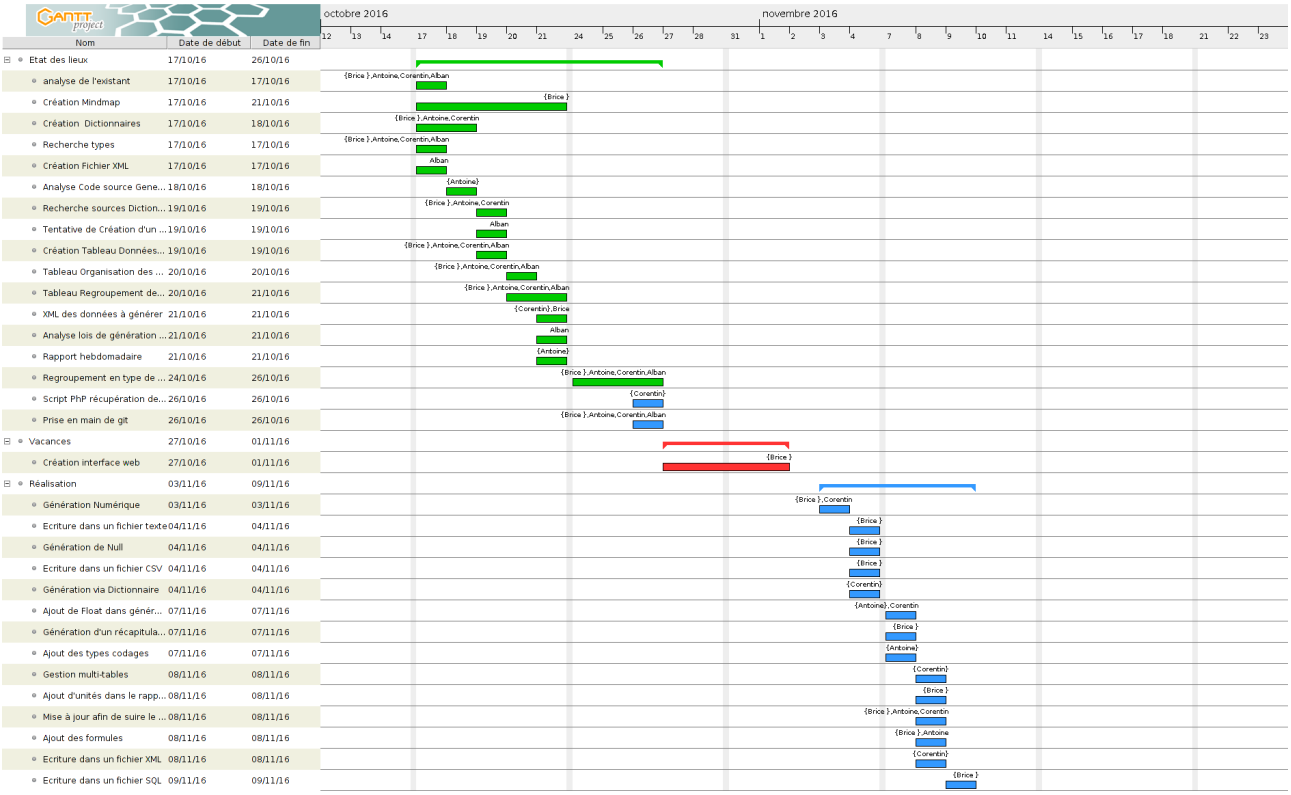
7.6 Répartition des tâches

Voici la répartition des tâches entre les différentes personnes du groupe :



7.7 Le Gantt

Voici le Gantt qui nous a permis de nous organiser et de nous répartir les tâches entre nous



7.8 Exemple de fichier de sortie

7.8.1 SQL

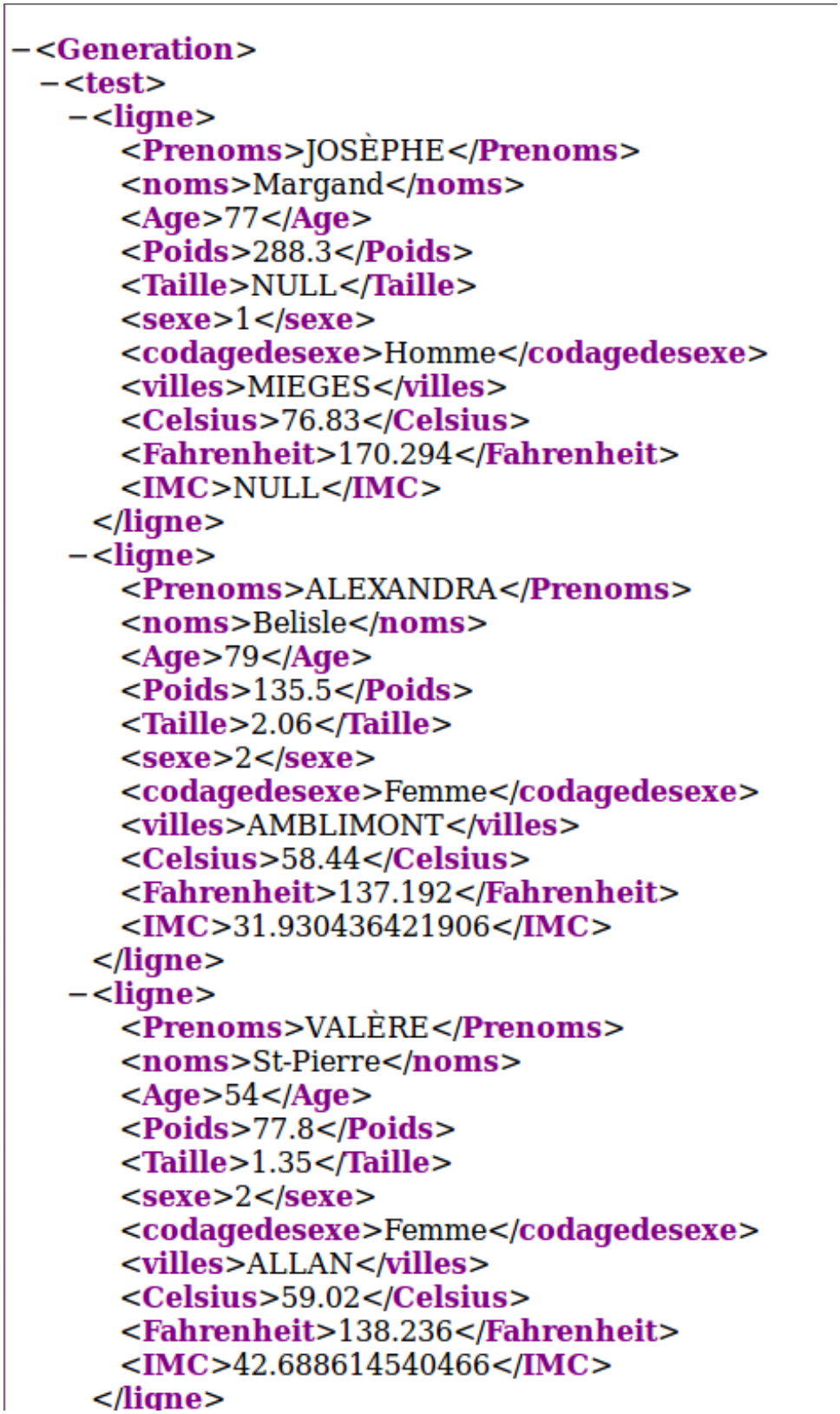
Voici ce que donne le fichier SQL généré à partir du fichier XML :

```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);
```

7.8.2 XML

Voici ce que donne le fichier XML généré à partir du fichier 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>
```

7.8.3 CSV

Voici ce que donne le fichier CSV généré à partir du fichier XML :

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

7.9 Sources

7.9.1 Sources des dictionnaires :

Films : http://www.gogo-films.com/liste-videos.php
Médicaments : https://www.vidal.fr/Sommaires/Medicaments-A.htm
Prénoms : 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/
Voirie/taille vêtement/groupe sanguin : nous même

7.9.2 Les différents générateurs de données existant :

generatedata : http://www.generatedata.com/?lang=fr#t1
Mockaroo : https://www.mockaroo.com/
Yan Data Ellan : http://www.yandataellan.com/
Free Data Generator : http://www.freedatagenerator.com/