Rapport de projet : Base de données et analyses liée à l'épidemie de la Covid-19

Vincent AUCONIE & Chaolei CAI
Université de Paris
UFR Informatique
M1 Informatique

May 16, 2021

1 Introduction

1.1 Avant propos

Notre objectif dans ce projet est la mise en place d'une base de données en lien avec la pandémie de Covid-19.
Nous avons décidé, en explorant les différentes données publiques mises à notre disposition, de focaliser notre projet sur le dépistage et les tests en France (métropole et outre-mer).

1.2 Objectifs de modélisation

Nous allons essayer, dans ce projet, de créer un modèle basé sur les tests en France et l'étude des variants. Les tests de dépistage contre la Covid-19 en France on permis la détection de nombreux cas positifs et cas contact et donc fortement aidé des personnes à se protéger du virus. C'est pour cela que l'on considérait le dépistage en France comme une donnée importante dans l'étude de l'évolution de l'épidémie sur le sol national.

1.3 Sources de données

Au vu des différentes sources de données publiques accessibles aujourd'hui, nous savions qu'un modèle utilisant uniquement ces données serait difficilement réalisable à l'échelle de ce projet. Ainsi, nous avons décidé de réaliser un modèle basé sur les données des tests réalisés dans les différents lieux de dépistage en utilisant des données démographique et des données d'identités artificielles.

Cependant, nous utiliserons dans l'implémentation de cette base de données le fichier publique de **data.gouv.fr** recensant les différents lieux de dépistage possibles sur le territoire pour une personne souhaitant se faire tester.

Ce dataset est disponible et mis à jour à l'adresse suivante :
`https://www.data.gouv.fr/fr/datasets/lieux-de-depistage-covid-19-laboratoires-et-pharmacies-sante-fr/`

2 Modélisation

2.1 Tables et clés primaires

Nous allons ainsi chercher à créer notre modèle autour du système de dépistage de la Covid-19 en France. Notre modèle sera basé sur les tables suivantes (on précise aussi nos clés primaires) :

- test(id_test, id_lieu, id_personne, resultat, variant, date) :
- lieu_test(id_lieu, nom_lieu, type_lieu, adresse, code_postal, adresse_ville) :
- personne_test(id_personne, nom, prenom, sexe, age, domicile, birthdate) :

Nous ajoutons quelques précisions sur certains attributs non triviaux (nous établirons les contraintes et types de chaque attribut dans la section suivante) :

test[date] représente la date à laquelle à eu lieu le test. Il sera implémenté dans le type *Date* afin de faciliter les traitement possible avec les fonctions & triggers.

test[variant] représente le variant de la Covid-19 "crypté" lors du dépistage du patient **positif**. Nous avons fixé les différents variants possible comme tels : Anglais (EN), Sud Africain (SA), Brésilien (BR), Indien (IN) et Français (FR). Nous avons aussi supposé leurs probabilités d'apparition différentes (nous aborderons cela lors de l'implémentation de notre modèle).

2.2 Spécification des contraintes

Notre modèle génère ainsi des contraintes de **clé étrangère** sur notre relation **test**.

- test(id_test, id_lieu, id_personne, resultat, variant, date) :
test[id_personne] $\subseteq$ personne_test[id_personne]
test[id_lieu] $\subseteq$ lieu_test[id_lieu]

- lieu_test(id_lieu, nom_lieu, type_lieu, adresse, code_postal, adresse_ville) :

- personne_test(id_personne, nom, prenom, sexe, age, domicile, birthdate) :

L'implémentation de notre modèle génère ainsi plusieurs autres contraintes.
On précise aussi le type de chaque attribut de nos tables :

- Test :

```
CREATE TABLE test(
    id_test SERIAL PRIMARY KEY,
    id_lieu int,
    id_personne int,
    resultat int,
    variant varchar,
    date Date
);
```

- Lieu_test :

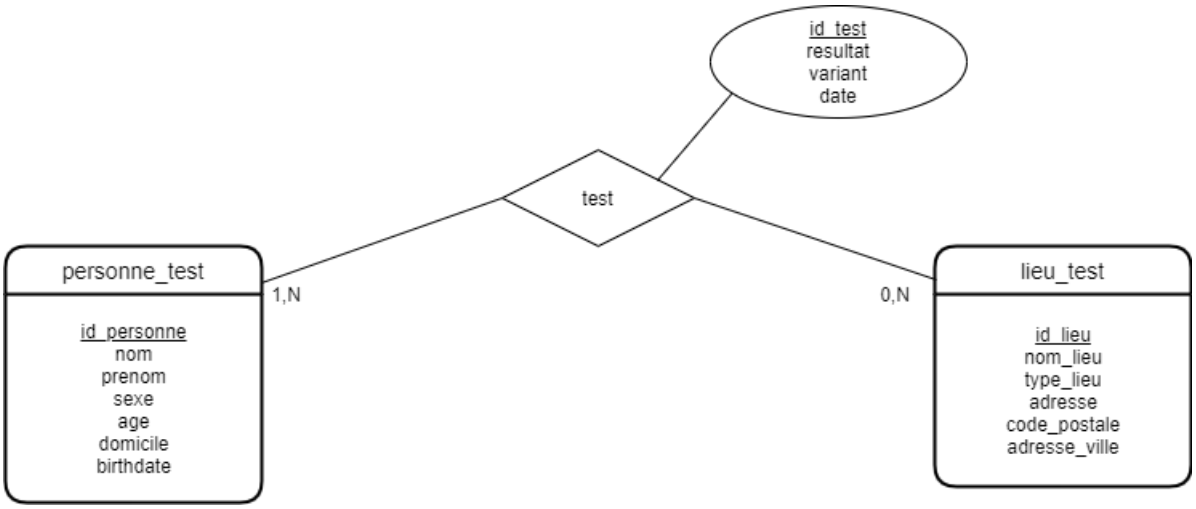
```
CREATE TABLE lieu_test(
    id_lieu SERIAL PRIMARY KEY,
    nom_lieu varchar,
    type_lieu varchar,
    adresse varchar,
    code_postal varchar,
    adresse_ville varchar,
    CONSTRAINT unique_v UNIQUE(nom_lieu, type_lieu, adresse, code_postal, adresse_ville)
);
```

- Personne_test :

```
CREATE TABLE personne_test(
    id_personne SERIAL PRIMARY KEY,
    nom varchar,
    prenom varchar,
    sexe varchar,
    age int,
    domicile varchar,
    birthdate Date,
    UNIQUE(nom, prenom, sexe, age, domicile, birthdate)
);
```

On représente ainsi notre modèle, ses tables et ses contraintes à travers

un diagramme Entité/Relation :



2.3 Remarques

Une modélisation plus complexe et plus riche aurait été possible. En effet, nous aurions pu ajouter à notre modèle les données concernant les statistiques démographiques de chaque région/département ou encore des données sur les capacités de test quotidiennes (nombre de test disponibles par jour) mais ces tables et attributs supplémentaires auraient rendus l'implémentation et la création de notre base de données plus complexe et nous avons donc décider de rester sur ce modèle plus raisonnable.

3 Création et peuplement des tables

3.1 Création des tables

Afin de créer notre base de données et plus particulièrement nos tables, nous implémentons 3 fichiers SQL (1 pour chaque table) qui ont pour but de créer les tables en prenant en compte les types définis et nos contraintes (ex: "create_test.sql").

De plus, nous avons crée le fichier "create_prepares.sql" qui implémente nos requêtes préparées, le fichier "create_trigger.sql" qui implémente nos triggers et le fichier "create_fonctions.sql" qui implémente nos fonctions.

On regroupe ces 6 fichiers d'implémentation dans le fichier "create_all.sql"

Concernant les fonctions, on a décidé d'implémenter une fonction qui calcule l'age d'une personne avec sa date de naissance, une fonction qui génère un nombre aléatoire entre 2 bornes qui nous est utile pour le remplissage des tables ainsi que d'autre fonctions de gestion.
On a aussi implémenter une fonction pour choisir aléatoirement une date ainsi pour nos 3 tables des fonctions de recherche, de comparaison, etc...

3.2 Peuplement des tables

Le peuplement de nos tables se fait via un script automatique contenu dans le fichier "insert_data.sql". Ce dernier est composé des fichiers "load_personne_test.sql", "load_lieu_test.sql" et "load_test.sql" qui permette de remplir nos tables :

Remarque : Nos différents fichiers de données sont tous dans le dossier "csv_input" de notre projet.

• "load_personne_test.sql" permet de remplir notre table Personne_test a l'aide d'un fichier créer artificiellement et contenant un set de 1000 identités fictives. Celui ci utilise une table temporaire puis dans un ordre aléatoire peuple notre table.

• "load_lieu_test.sql" utilise aussi une table temporaire dans laquelle on importe les données du fichier **"santefr-lieux-depistage-covid-laboratoires.csv"** récupéré via le site https://www.data.gouv.fr/fr/datasets/lieux-de-depistage-covid-19-laboratoires-et-pharmacies-sante-fr/.

• "load_test.sql" rempli lui notre table **Test**. Il utilise nos 2 tables déjà crées et remplies. Pour ce qui concerne les résultats des tests et donc les attributs "resultat" et "variant" : le résultat est obtenu par tirage aléatoire uniforme et en ce qui concerne le variant, nous avons décidé de crée un fichier

"variant_prob.csv" qui défini les différents variants possible (artificiellement) et leur probabilité d'apparition.

Pour le peuplement, nous avons décidé de remplir notre table Personne_test de 1000 identités artificielles et notre table Lieu_test de 100 lieux de dépistages.

3.3 Tests et démonstration

Pour tester nos différentes fonctions PL/pgSQL et triggers, nous avons crée pour chaque table un fichier "demo_XXX.sql" (nous avons évité le nom conseillé dans le sujet du projet "test_XXX.sql" pour éviter toute confusion avec le nom de nos tables).

4 Conclusion

A la fin de ce projet, nous obtenons ainsi un dossier général composé principalement de 2 sous-dossier "csv_input" et "scripts". Le dossier "csv_input" contient les fichier csv utile à l'importation des différentes données fictives ou réelles et le dossier "scripts" contient l'ensemble des fichier SQL permettant la création de nos tables, leur remplissage et les tests effectués sur nos tables.

Dans l'aspect du modèle que nous avons décidé de construire, il s'est avéré assez difficile de trouver des données publiques pouvant nous aider à remplir nos tables. Ainsi nous avons quand même souhaité importer des données réelles et officielles concernant les lieux de dépistage en France afin de pouvoir créer des relations dans notre modèle entre des instances artificielles (identité de personnes) et des instances réelles (lieux de dépistage).