Université Le Havre Normandie
UFR des Sciences et Techniques

Laboratoire d' Informatique de Traitement
de l'Information et des Systèmes (LITIS)

# Programmes d'alimentation de la base de données à partir des données Navires et ports

EN VUE D'OBTENTION DE DIPLÔME DE LICENCE EN INFORMATIQUE

Réalisé par : KOUCHE Rabia
Année universitaire 2019/2020

Maître de Stage : Claude Duvallet

Tuteur à l'université : Dominique Fournier

# Abstract

As part of the Computer Science license at Le Havre Normandie University, I had the opportunity to do my internship within LITIS (Laboratory of Computer Science, Information Processing and Systems) -Antenna of Le Havre, whose researchers are working on several research projects.

We have a PIL-23 server, which hosts several websites presenting research projects. Among these sites is the site http : //data-platform.projet -recherche-normandie.fr/ which aims to set up a data platform. It is already in production but it has several major drawbacks including maintenance and performance problems. It is therefore a question of remaking this site. It was developed in Ruby On Rail and uses a MongoDB database, which is a NoSql oriented database. NoSQL databases have a completely different approach from relational databases (Mysql, Postgresql, Oracle ...). A lot of AIS data is stored on this data platform but it is very difficult as it is to exploit it. Our job therefore consisted in re-taking the management of AIS data in a PostgreSQL database. To feed this new database, it was therefore necessary to build a certain number of programs or scripts which can do it automatically. Then, it will be possible to build interfaces to access this data in HTML / CSS / PHP / JS. In this project, I was assigned a certain number of tasks which consisted in creating programs which make it possible to automatically feed a "PostGreSQL" database from the data present in Excel files.

`Keywords:` AIS data, Ships, Ports, data platform, PostGreSQL.

# Résumé

Dans le cadre de la licence Informatique à l'Université Le Havre Normandie, j'ai eu l'opportunité de faire mon stage au sein du LITIS (Laboratoire d'Informatique, de Traitement de l'information et des Systèmes) - Antenne du Havre, dont les chercheurs travaillent sur plusieurs projets de recherche.

Nous disposons d'un serveur PIL-23, qui héberge plusieurs sites web présentant des projets de recherche. Parmi ces sites, se trouve le site http ://data-platform.projet-recherche-normandie.fr/ qui a pour objectif de mettre en place une plate-forme de données. Il est déjà en production mais il possède plusieurs inconvénients majeurs dont des problèmes de maintenance et de performances. Il est donc question de refaire ce site. Il a été développer en Ruby On Rail et utilise une base de données MongoDB, qui est une base de données orienté NoSql. Les bases de données NoSQL ont une approche complètement différentes des bases de données relationnelle (Mysql, Postgresql, Oracle ...). De nombreuses données AIS sont stockées sur cette plateforme de données mais il est très difficile en l'état de les exploiter. Notre travail consistait donc à reprendre la gestions des données AIS dans une base de données PostgreSQL. Pour alimenter cette nouvelle base de données, il était donc nécessaire de construire un certain nombre de programmes ou de scripts qui puissent le faire automatiquement. Ensuite, il sera possible de construire des interfaces d'accès à ces données en HTML/CSS/PHP/JS. Dans ce projet, je me suis vu attribuer un certain nombre de tâches qui consistaient à créer des programmes qui permettent d'alimenter automatiquement une base de données "PostGreSQL" à partir des données présentes dans les fichiers Excel.

`Mots clés :` Messages AIS, Navires, Ports, plateforme de données, PostGreSQL.

# Table des matières

# Remerciements

Avant de commencer le développement de cette expérience professionnelle il me parait primordial de remercier les personnes qui m'ont soutenu tout au long de mon travail.

Tout d'abord je tiens à remercier mon maître de stage Monsieur `Claude Duvallet` pour m'avoir encadré dans mon stage, pour son suivi, sa patience et pour les conseils qu'il m'a donné.

Merci également à notre responsable de formation Madame `Véronique Jay`, de faire preuve d'autant d'empathie de gentillesse et sincérité, et je remercie aussi tout les enseignants de département Informatique pour leurs contribution au bon déroulement de la formation malgré cette crise sanitaire.

Un merci bien particulier à ma famille pour leur soutient autant moral que financier.

# 1 Présentation du contexte professionnel

Il existe plusieurs laboratoires de recherche situés à l'Université Le Havre Normandie. Parmi ces laboratoires de recherches, se trouve le LITIS (Laboratoire d'Informatique de Traitement de l'Information et des Systèmes) qui implique les trois principaux établissements d'enseignement supérieur, Université de Rouen Normandie, l'Institut National des Sciences Appliquées de Rouen (INSAR) et l'Université Le Havre Normandie. Il regroupe 90 chercheurs qui travaillent sur plusieurs projets de recherches. Parmi ces projets de recherche, se trouve le projet CLASSE 2 dans lequel une plateforme de données a été initiée. Notre travail consistait alors à reprendre cette plateforme de données pour pouvoir présenter tout ce qui concerne les messages AIS et leur lien avec les navires et les ports.

A travers ce rapport, je détaillerai donc les neuf premières semaines de mon stage, en commençant par une présentation de l'environnement de travail (cf. section 2). Dans la suite de ce rapport (cf. section 3), j'enchaînerai par une présentation du travail à effectué. Ensuite, dans la dans la section 4 J'aborderai les méthodes, les différents résultats que j'ai pu obtenir et le schéma entité association de mes bases de données. Puis dans la section 12 je vais finir par citer les difficultés que j'ai rencontré dans ce stage. Et enfin, la dernière partie (cf. section 14), je dresserai une conclusion et je pourrai éventuellement vous parler des perspectives envisageable et un bilan 13, tant au niveau des connaissances utilisées qu'au niveau de celles qui m'ont été apportées au sein du laboratoire. Puis, compléter ce rapport par quelques définitions et une Bibliographie et les annexes.

# 2 Environnement de travail

Le stage s'est déroulé en télétravail à cause de la situation actuelle (COVID-19), ce qui n'a pas permis mon intégration dans le laboratoire et d'avoir un contact direct avec mon encadrant et les autres étudiant travaillant sur le même projet, ce qui aurait pu s'avérer bénéfique. Cependant, nous avons réussi à bien organiser le travail avec l'utilisation des outils suivant :

`Discord :` est un logiciel de communication. Il est basé sur le principe de serveur. Chaque utilisateur peut créer un ou plusieurs serveurs et il devient donc administrateur. Les utilisateurs peuvent rejoindre ce serveur grâce à des invitations. Nous avons donc crée notre propre groupe de discussion privé, nommé "Stage AIS", pour pouvoir faire des réunions et poser les questions jugées utiles.

`Le serveur PIL-23 :` c'est un serveur sur lequel sont hébergés des sites WEB, et sur lequel on va héberger la plateforme de données.

`Overleaf :` Il s'agit d'un outil de travail collaboratif qui permet de travailler à plusieurs sur un document latex. Cet outil nous a permis de faire des corrections des documentations directement sur le fichier source.

`Trello :` Trello est un outil de gestion de projets en ligne. Il repose sur une organisation des projets en planches listant des cartes, chacune représentant des tâches. Les cartes sont assignables à des utilisateurs et sont mobiles d'une planche à l'autre, traduisant leur avancement.

Cet outil nous a beaucoup aidé durant le stage (voir annexe (1) dans la section 17). À chaque fin d'une tâche, nous exposons le travail réalisé au maître de stage afin de le valider et déplacer la carte de "tâches en cours" à "tâches terminées", puis passer à une autre tâche ou évoquer les erreurs ou les manques et proposer des solutions afin de compléter ou corriger le travail effectuer.

# 3 Présentation du travail à effectuer

Nous disposons d'un ensemble de fichiers Excel comportant des données sur les navires et les ports du monde entier. L'objectif de notre programme est de disposer d'un outil permettant d'alimenter automatiquement une base de données "PostGreSQL" à partir des données présentes dans les fichiers Excel. nous avons concernait une contrainte qui est le langage de programmation qui devait être au choix soit Java, soit Python. Nous avons opté pour le langage Python.

En effet ma mission première était de d'installer tout les outils nécessaire pour la réalisation de ce projet notamment 'PyCharm' qui est environnement de développement intégré utilisé pour programmer en Python, et de configurer le serveur PIL-23 afin de pouvoir y accéder et importer les données dans une base de données PostgreSQL.

Dans un second temps, j'ai du réalisé des programmes en Python qui ont pour rôle d'alimenter notre base de données PostgreSQL à partir des fichiers Excel présent. Enfin Création du schéma de bases de données.

# 4 Travail effectué

# 5 Les outils

## 5.1 Se connecter à une base de données PostGreSQL en Python

La première étape de ce travail a été de commencer par effectuer des recherches sur les moyens de connexion entre une base de données PostGreSQL et un programme en langage Python. Nous avons sélectionné la librairie **psycopg2** qui permet de se connecter à une base de données PostGreSQL et de l'interroger. Il s'agit d'un adaptateur de bases de données PostGreSQL qui est à la fois très utilisé, léger et efficace. Sous Linux, son installation relativement simple se fait de la façon suivante :

```
pip install psycopg2
```

Pour l'installer sous l'environnement Mac OS X, il suffira de procéder de la façon suivante :

```
pip install psycopg2
pip install psycopg2-binary
```

## 5.2 Lister les fichiers présents dans le répertoire courant

Pour récupérer tous les fichiers du répertoire courant, nous avons opté pour la librairie `glob`. Elle nous permet d'utiliser la méthode `glob.glob(" *.csv")` qui retourne, d'une façon arbitraire, les fichiers au format CSV puis les ajoute dans un tableau.
La methode `split()` nous permet de récupérer la partie centrale du nom de fichier.

## 5.3 Estimer le temps d'exécution du programme

Il est très utile de savoir combien de temps un script a mis à s'exécuter.
Pour cela, nous avons la fonction `time()` inclue dans la librairie `time`. L'appel de cette fonction renvoie le temps courant en secondes. Pour connaître la durée d'exécution de notre programme, il faut soustraire le temps du début de l'exécution à celui de la fin.

# 6 Conception et alimentation de la base de données

Nous avons repris l'ensemble des fichiers Excel comportant nos données et les avons convertis au format CSV qui permet de séparer les différents champs par des points-virgules. Nous avons fait ce choix car il s'avérait plus facile de traiter des fichiers CSV par le biais de notre programme Python.

### 6.1 Création de la base de données

Pour effectuer le travail demandé, il fallait d'abord créer une base de données locale. Cette base de données possède cinq tables :

1. La table 'Navire' qui est destinée à recevoir l'ensemble des données concernant les navires. Elle contient donc toutes les colonnes des fichiers "navire" ainsi que le nom du fichier décrivant le type de navire.
2. La table 'Arborescence_navires' qui est destinée à recevoir les données concernant la classification, selon 4 niveaux, des navires stockées dans la table précédente.
3. La table 'Ports' qui est destinée à recevoir les données concernant les ports. Elle contient donc toutes les colonnes de fichiers "port" et compris un lien vers les terminaux et les quais (berths).
4. La table 'Berth' qui est destinée à recevoir les données concernant les postes d'amarrage (quais) de chaque port. Elle contient donc un lien vers la table port et toutes les colonnes du fichier 'Berths'.
5. La table 'Terminal' est destinée à recevoir les données des terminaux de chaque port. Elle contient donc un lien vers la table 'Port' en tant que clé étrangère ainsi que les colonnes du fichier 'Terminal'.

### 6.2 Connexion à une base de données PostGreSQL en Python

La première étape de notre algorithme d'alimentation de la base de données "Navire" consistait à créer une connexion à cette base de donnée via notre programme Python. Pour cela, voici le code qu'il faut exécuter :

```
# Connexion a la base de données bd_navire
# Attention, il faut que l'utilisateur soit le propriétaire
# de la base de données ou possède des droits suffisant
connexionBD = psycopg2.connect(
  host = "localhost", # Nom du serveur où se trouve la BD
  database = "bd_navire", # BD contenant la structure de données
  user = "USER",  # Nom de l'utilisateur
  password = "PASSWORD" # Mot de passe de l'utilisateur
)

curseur = connexionBD.cursor()
```

À partir de l'objet `connexionBD`, il sera possible d'insérer, de valider (*commit)* ou d'annuler (*rollback*) des transactions.

## 7 Description du programme d'alimentation de la base de données bd_navire

Nous allons décrire dans cette partie les différentes partie de notre programme réalisé en Python, Nous avons commencé par ajouter tous les fichiers `navire` et le fichier `Arborescence_navires` présent dans le répertoire dans un tableau, ensuite nous parcourons ces fichiers afin de faire une extraction des données depuis ces derniers. Ensuite, il faut insérer ces données dans les deux tables `Navire` et `Arborescence_navires` qui s'effectue à l'aide de la commande `INSERT INTO`. et enfin faire une sauvegarde de notre base de données "PostGreSQL".

### 7.1 Extraction des données depuis les fichiers CSV

On parcourt avec la boucle 'for' tous les fichiers présents dans le tableau. Ensuite, il faut récupérer les lignes de ces fichiers avec la méthode `readlines()` dans une liste, puis, découper ces lignes sur le caractère de séparation (';') avec la méthode `split()`.

## 7.2 Insertion des données dans la table Navire

Après avoir extraites les données des fichiers CSV, il faut les insérer dans la table navire. Nous procédons alors de la manière suivante :
Parcourir chaque fichier navire dans le tableau `tableau_navires` et Insérer chaque valeur d'une ligne extraite dans la table en passant l'ordre d'insertion à la base de données au travers le curseur issu de la connexion `(curseur.execute(" insert into ... values (...)").`
Nous avons constaté qu'il y avait des navires en double dans certain fichiers. Alors, nous avons du gérer des exceptions lors de l'insertion de ces données, pour éviter la violation de la clé primaire. Puis, il faut faire un 'rollback' sur la transaction en échec sinon cela bloque toutes les transaction suivantes.

## 7.3 Insertion des données dans la table Arborescence_navire

Nous procédons de la même manière que précédemment pour la table 'Navire' lors de l'insertion des données dans la table 'Arborescence_navires'. La structure de cette table étant différente, nous avons du la traiter à part. Pour chaque ligne lu dans le fichier, on la découpe et on insère les champs dans la base de données avec la requête SQL suivante :

```
curseur.execute("INSERT INTO Arborescence_navires
   (niveau1, niveau2, niveau3, niveau4) "
   "VALUES(%s, %s, %s, %s)",
   (infos[0], infos[1], infos[2], infos[3]))
```

## 7.4 Exécution du programme Python et résultats

Il existe des navires en double pouvant provoquer une violation de la clé étrangère présente dans la table arborescence_navire et liée à la table navire alors nous avons du utiliser les exceptions et faire un rollback sur la transaction echouée.

```
##########################################################
### Programme d'alimentation de la base de données navire  ###
### Importation de données au format CSV dans une base de  ###
### données PostGreSQL                                     ###
##########################################################
Other Activities Dredging              : importation de    4775 de navires et rejet de      11 navires en double
Other Offshore                         : importation de    2201 de navires et rejet de       0 navires en double
Research                               : importation de    1078 de navires et rejet de       0 navires en double
Passenger Ro-Ro Cargo                  : importation de     366 de navires et rejet de       0 navires en double
Towing Pushing                         : importation de   19339 de navires et rejet de     571 navires en double
Yacht                                  : importation de    3348 de navires et rejet de      13 navires en double
Inland Waterways Tanker                : importation de     972 de navires et rejet de       0 navires en double
Other activities                       : importation de     740 de navires et rejet de       0 navires en double
Oil                                    : importation de    8806 de navires et rejet de      94 navires en double
Passenger General Cargo                : importation de       1 de navires et rejet de     366 navires en double
Refrigerated Cargo                     : importation de     923 de navires et rejet de       0 navires en double
Barge                                  : importation de     696 de navires et rejet de       0 navires en double
Liquefied Gas                          : importation de    2236 de navires et rejet de       0 navires en double
Fish Catching                          : importation de   25171 de navires et rejet de       1 navires en double
General Cargo                          : importation de   15136 de navires et rejet de     255 navires en double
Bulk Dry                               : importation de   11366 de navires et rejet de       0 navires en double
Ro-Ro Cargo                            : importation de    2974 de navires et rejet de      19 navires en double
Non Ship Structures                    : importation de    1459 de navires et rejet de       0 navires en double
Passenger                              : importation de    4411 de navires et rejet de      19 navires en double
Inland Waterways Other Non Seagoing    : importation de     126 de navires et rejet de       0 navires en double
Chemical                               : importation de    5841 de navires et rejet de      81 navires en double
Other Bulk Dry                         : importation de    1247 de navires et rejet de       0 navires en double
Other Dry Cargo                        : importation de     328 de navires et rejet de       0 navires en double
Offshore Supply                        : importation de    7372 de navires et rejet de      92 navires en double
Bulk Dry Liquid                        : importation de      72 de navires et rejet de       0 navires en double
Other Fishing                          : importation de    1489 de navires et rejet de       2 navires en double
Le nombre total de navires importé dans la base de données est 128806
Arborescence navires.csv               : importation de     281 de navires et rejet de       0 navires en double
Temps d'exécution : 56.5936 secondes
```

Figure 1 – Résultats importation des navires

# 8 Description du programme d'alimentation de la base de données 'Port'

Nous disposons de trois fichiers Excel contenant des données sur les ports et quais. Chacun de ces fichiers possède une structure différente. De ce fait, nous convertissons ces fichiers au format CSV afin de créer pour des tables qui contiennent les colonnes concernant chaque fichier. Ensuite, nous procédons de la même manière que dans la section 7. La seule différence réside dans le fait de ne pas ajouter ces fichiers dans un tableau car leur structure est différente.

## 8.1 Insertion des données dans la table 'Port'

Pour chaque ligne lu dans le fichier CSV, on la découpe et on insère les champs dans la table en utilisant la requête SQL suivante :

```
curseur.execute("INSERT INTO Ports (ID, GlobalPortID, WorldPortNumber, MasterPOID,
PortName, AlternativeName, Status, Country, Latitude, Longitude) "
   "VALUES(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)",
   (infos[0],infos[1], infos[2], infos[3], infos[4],
   infos[5], infos[6], infos[7], infos[8].replace(",","."),
   infos[9].replace(",",".")))
```

Les données réelles sont présentées avec des virgules dans les fichiers CSV ce qui ne pourra pas être accepté par PostGreSQL. Nous utilisons donc la méthode `replace(",",".")` pour résoudre ce problème.

## 8.2 Insertion des données dans la table Terminal

De la même manière, on utilise la commande SQL suivante :

```
curseur.execute("INSERT INTO Terminal (ID, Port_ID, Name, Operator, Facility_Type,
Latitude, Longitude, GlobalPortID, GlobalTerminalID, Status, PORT)"
 "VALUES(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)",
 (infos[0],infos[1], infos[2], infos[3], infos[4],
 infos[5].replace(",",".") or 0.0, infos[6].replace(",",".")
 or 0.0, infos[7], infos[8], infos[9],
 infos[10]))
```

Il existe des champs de type réel ou entier qui sont vides dans les fichiers CSV. Pour qu'ils ne soient pas vu comme des chaînes de caractères vides, nous les remplaçons par 0 (infos [?] OR 0).

## 8.3 Insertion des données dans la table Berth

De la même manière, nous insérons chaque valeur d'une ligne extraite dans la table en passant l'ordre d'insertion à la base de données au travers du curseur issu de la connexion :

```
curseur.execute("INSERT INTO Berth (ID,Port_ID, Terminal_ID, Name, No, Operator,Type,
Status, Facility_Type,Latitude,Longitude, GlobalPortID, GlobalTerminalID, GlobalBerthID) "
 "VALUES(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)",
 (infos[0],infos[1] or 0, infos[2] or 0, infos[3], infos[4],
  infos[5], infos[6], infos[7], infos[8], infos[9].replace(",",".")
  or 0.0, infos[10].replace(",",".") or 0.0, infos[11],infos[12],
  infos[13]))
```

### 8.4 Exécution du programme Python et résultats

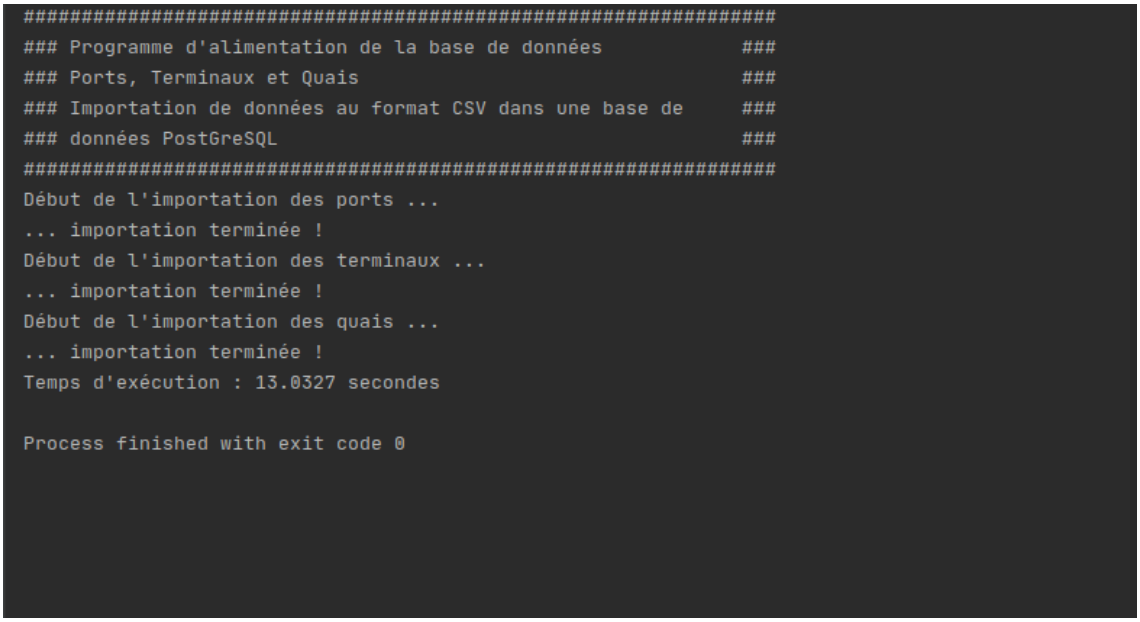



Figure 2 – Résultats importation des ports et quais

## 9 Création d'une sauvegarde de la base de données (dump)

Pour créer une sauvegarde de la base de données "PostGreSQL", il faut utiliser la commande suivante :

```
pg_dump bd_navire > bd_navire_structure_et_donnees.sql
```

Il faut préciser en argument le nom de la base de données local

## 10 Migration des données dans la base PostGreSQL sur PIL-23

Pour effectuer la migration de la base de données vers le serveur PIL-23, nous avons du procéder de la manière suivante :

1. Reprendre la sauvegarde (effectuée précédemment) de la base de données (fichier SQL).
2. Effectuer le transfert de ce fichier vers le serveur PIL-23.
3. Se connecter sur le serveur PIL-23.
4. Importer le fichier SQL au sein de la base de données PostGreSQL présente sur le serveur PIL-23.

   ```
   psql dataplatform < navires_donnees.sql
   ```

# 11 Création du schéma de bases de données

## 11.1 Schèma de base de données Navires

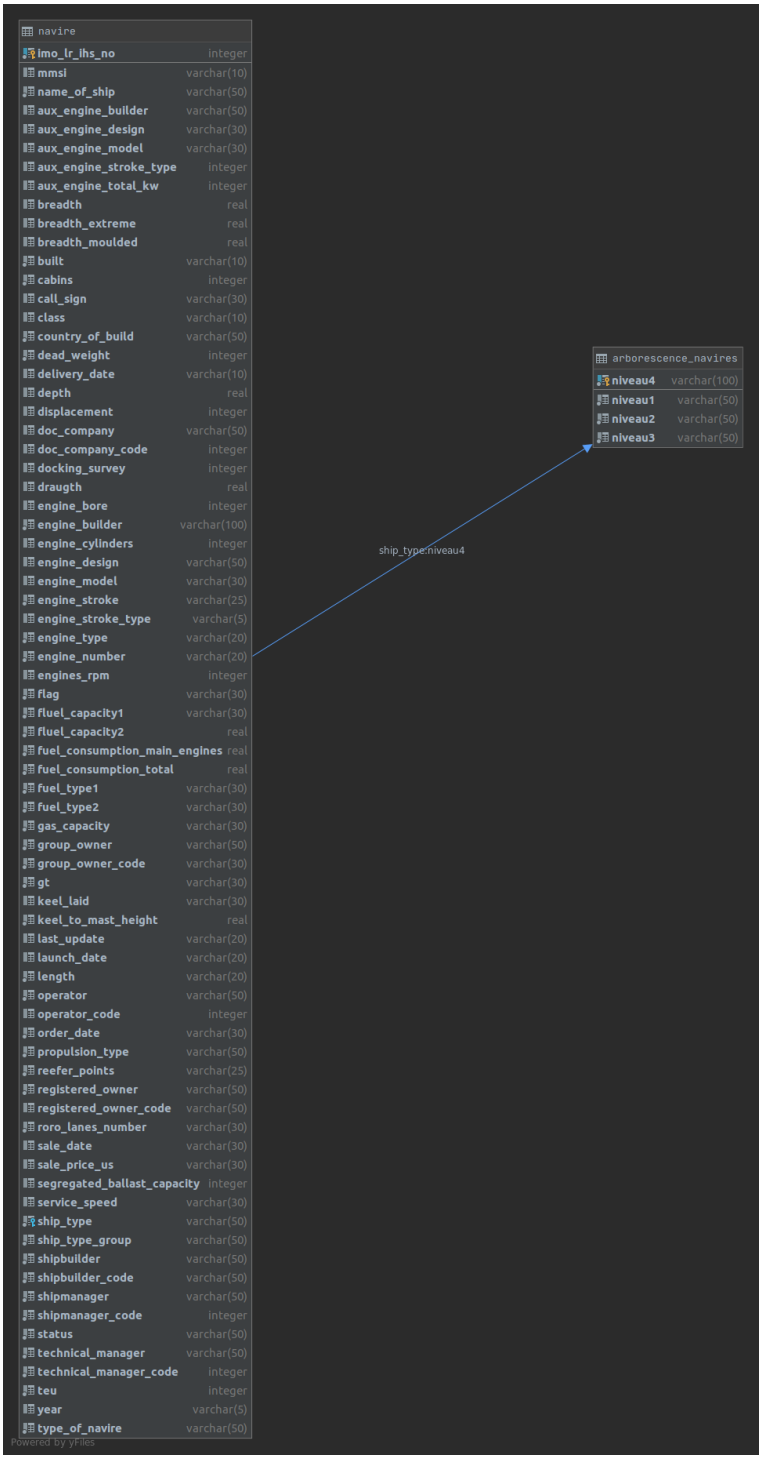



Figure 3 – Schèma de base de données des navires et le lien avec l'arborescence navire

### 11.2 Schéma de base de données Ports

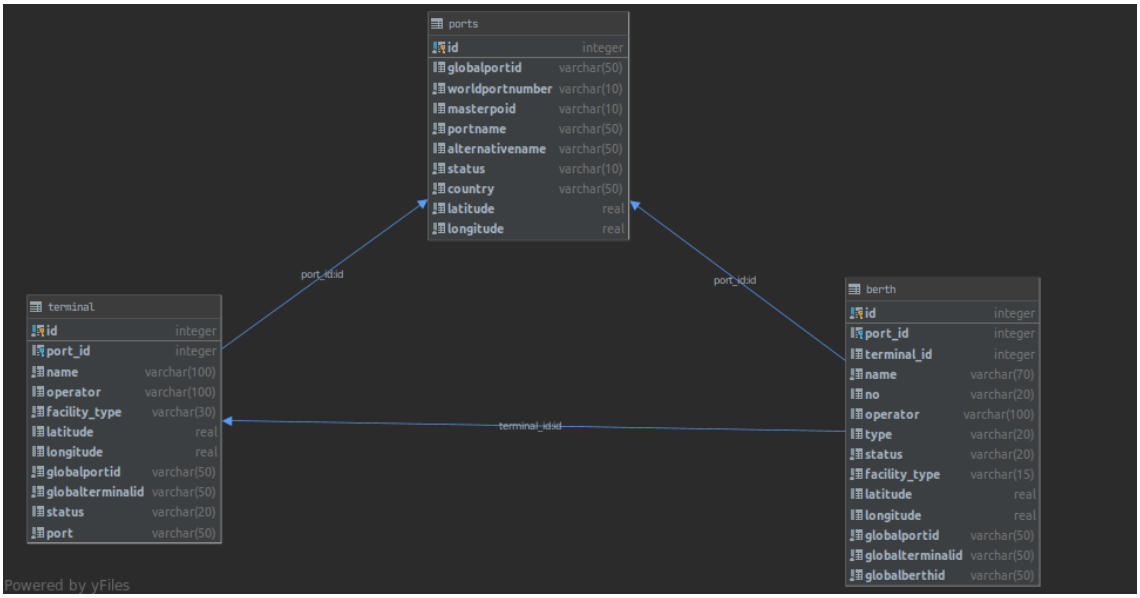



Figure 4 – Schèma de base de données des ports et leurs liens avec les terminaux et les quais

## 12 Difficultés rencontrées

Les difficultés rencontrées ont été de deux ordres. Des difficultés techniques et des difficultés d'organisation.

### 12.1 Difficultés technique

On peut évoquer par exemple la connexion d'une base de données dans un programme Python. Au début de ce projet, je n'avais pas toutes les notions de Python nécessaires. Finalement, en me documentant, j'ai pu trouver une solution adéquate.

De plus, les fichiers 'Excel' des navires ne sont pas complet au début. Il leur manquait des colonnes. J'ai perdu un temps précieux. J'ai néanmoins réussi à obtenir l'aide nécessaire, soit avec l'aide de mon maître de stage, soit en cherchant sur Internet.

### 12.2 Difficultés d'organisation

Dans ce projet, j'ai rencontré surtout des difficultés d'organisation. En effet, je devais travailler, dans le même laboratoire en télétravail, pour un projet impliquant 6 personnes. Bien que ma tâche soit secondaire et autonome par rapport aux tâches des autres participants, mes collègues avaient néanmoins besoin des bases de données et des programmes que je réalisais.

Ensuite, un autre délai impératif m'a été fixé pour finir mes projets de deuxième année. J'ai donc du m'y consacrer de manière intensive en travaillant les week-end et les jours fériés.

# 13 Perspective avec la formation et bilan

Grâce à ce stage j'ai pu mettre en oeuvre mes acquis durant toute la licence, j'ai beaucoup appris tant au niveau théorique que pratique, notamment en base de données et en python. Je me suis aperçu de l'importance de trouver des solutions non pas seulement qui marchent mais qui sont efficaces en termes de temps et de coûts.

Au cours de ce stage, j'ai été amené à réaliser des tâches qui consistaient à alimenter des bases de données à partir de fichiers Excel de tailles très importantes. J'estime que cela a une très grande importance dans le monde informatique notamment le WEB. J'estime que les objectifs de toutes les tâches qui m'ont été confiées, sont atteint même si de petites améliorations sont possibles.

Pour conclure, mon intérêt pour le domaine des technologies du WEB et des bases de données s'est considérablement accrue ce qui m'a donné l'envie de suivre un master orienté WEB et BIG DATA qui pourra venir enrichir mes connaissances et me permettre d'emprunter la voie professionnelle que je souhaite.

# 14 Conclusion

Ce stage m'est apparu comme une expérience très satisfaisante et enrichissante, pouvoir mettre en pratique le savoir que j'ai acquis durant toute ma formation. Ce sont autant de choses positives que j'en retire.

J'ai pu, grâce à ce stage, améliorer mes compétences et élargir mes connaissances en programmation en Python et en bases de données, mais aussi découvrir et acquérir de nouvelles connaissances en matière de programmation et de développement.

Le résultat de ce stage sera donc une application réalisée dans le respect de plusieurs règles de bonnes pratiques de programmation, qui permettent d'optimiser l'application. Parmi ces bonnes pratiques, se retrouve l'optimisation des données importées depuis les fichiers excel, l'utilisation de framework...

Pour conclure, l'application complète n'est pas encore fonctionnelle, car toutes les tâches n'ont pas pu être terminées par les autres stagiaires travaillant sur ce même projet. Néanmoins, les deux programmes que j'ai développés fonctionnent et permettent une alimentation de la base de données qui servira à mettre en place la plateforme données. Des petites amélioration sont toujours possibles.

# 15 Glossaire

`Données AIS :` c'est l'abréviation d'Automatic Identification System AIS en anglais, Système d'identification automatique (SIA) en français. Il s'agit d'un système d'identification automatique, anti-collision qui envoie des informations sur l'identité, le statut, la position, la vitesse d'un navire et la route des navires se situant dans une zone de navigation.

`Fichiers CSV :` Comma-Separated Values en anglais, ce sont des fichiers contenant plusieurs lignes, chaque ligne contient plusieurs champs séparés par un caractère : un espace, une virgule, un point virgule ou autre.

`PostGreSQL :` C'est un système de gestion de base de données. C'est un outil open-source (gratuit).

`SQL :` C'est l'abréviation de Structured Query Language. C'est un langage informatique utiliser pour l'exploitation des bases de données. Il permet de façon générale la définition, la manipulation et le contrôle de sécurité de données.

Dans la pratique, le langage SQL est utilisé pour créer des tables, ajouter des enregistrements sous forme de lignes, interroger une base de données, la mettre à jour, ou encore gérer les droits.

`Requête :` Dans une base de données, une requête offre la possibilité de rechercher des données en spécifiant des critères.

`HTML :` HyperText Markup Language, sert à représenter le contenu de la page web (la structure). On écrit le contenu et on s'interdit de mettre la forme, juste des indications de titre, des sous-titres, des paragraphes, blocs et d'autres structures de texte.

`CSS :` Cascading Style Sheets. Contrairement à HTML qui est conçu pour la représentation de contenu, CSS contrôle la police, le positionnement, la couleur, et le style des informations des sites WEB.

`JavaScript (JS)` est un langage de programmation. Il sert à coder des interactions qui se dérouleront dans le navigateur sans appel au serveur.

# 16 Bibliographie

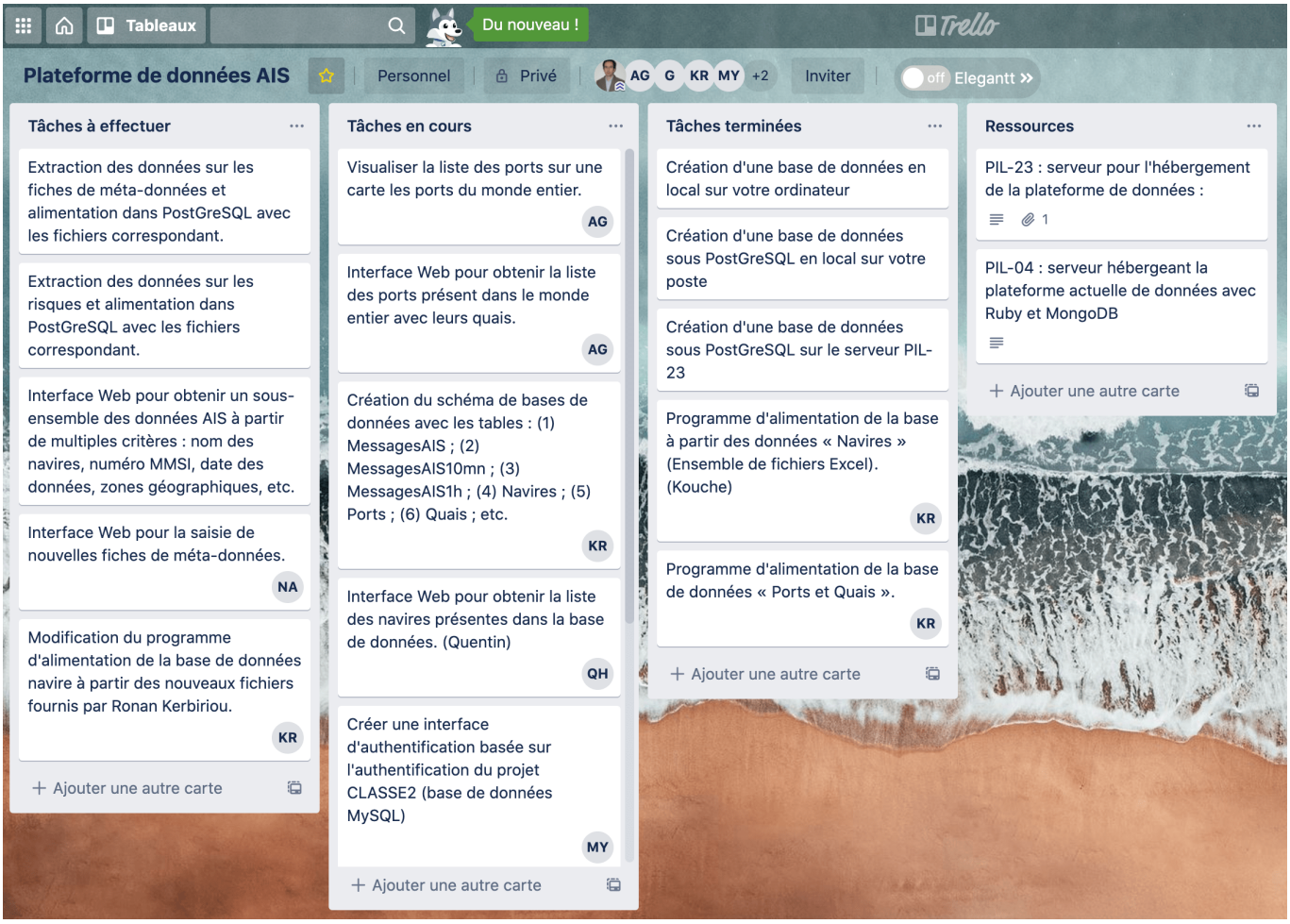
[1] Platforme de données http ://data-platform-classe.projet-recherche-normandie.fr/

[2] Exécution de requêtes PostgreSQL sous Python https ://librecours.net/module/dwh/etl01/pyt2c02.xhtml

[3] Documentation Python https ://docs.python.org/fr/3.6/index.html

# 17 Annexes

Annexe 1 :




Trello de la plateforme de données AIS

Annexe 2 :
L'importation des navires à partir des fichiers excel convertis en fichiers CSV.

```
########################################################################
### Alimentation de l'arborescence des navires sur 4 niveaux
### Le niveau 2 est lié au type de navire (nom des fichiers Excel)
### Le niveau 4 correspond à la colonne 'ship type'
########################################################################

# Ouverture de fichier Arborescence navires.csv en mode lecture
with open("Arborescence navires.csv",'r') as fp:
    next(fp)
     # La méthode readlines () lit jusqu'à la fin de fichier, et retourne une liste contenant les lignes
    lignes = fp.readlines()

nombre_ligne_importe=0
nombre_ligne_en_double=0

# Pour chaque ligne lu dans le fichier, on la découpe et on insère les champs dans la base de données
for ligne in lignes :
    infos = ligne.split(";",3)
    # insertion des données dans la table Arborescence navires
    infos[3].split(" ")
    try:
      curseur.execute("INSERT INTO Arborescence_navires (niveau1, niveau2, niveau3, niveau4) "
                      "VALUES(%s, %s, %s, %s)",(infos[0].strip(), infos[1].strip(), infos[2].strip(), infos[3].strip().title()))

      nombre_ligne_importe+=1 # On compte le nombre de navires importés
      #except (Exception, psycopg2.DatabaseError) as error:
    except:
      #print(error)
      nombre_ligne_en_double+=1 # On compte le nombre de navires non importés car déjà présent dans la base

# La méthode commit () : enregistre toutes les modifications apportées depuis le dernier commit dans la base de données.
connexionBD.commit()
print("{:40s}: importation de {:7d} de lignes et  rejet de {:5d} lignes en double".format ("Arborescence navires.csv",
                                                                                          nombre_ligne_importe,nombre_ligne_en_double))
```

Annexe 3 :
La connexion à la base de données ports et l'importation des ports **??**.

```
connexionBD  = psycopg2.connect(
    host     = "localhost",         # Nom du serveur où se trouve la base de données
    database = "bd_navire",     # Nom de la base de données contenant déjà la structure
    user     = "rabiak",              # Nom de l'utilisateur
    password = "13041996"       # Mot de passe de l'utilisateur
)
curseur = connexionBD.cursor()
#########################################################################
### Alimentation des ports sur 10 colonnes
### PortID est la clé prémaire
###
#########################################################################

print ("Début de l'importation des ports ...")

# Ouverture de fichier  Ports.csv en mode lecture
with open("Ports.csv",'r') as fp:
    next(fp)
     # La méthode readlines () lit jusqu'à la fin de fichier, et retourne une liste contenant les lignes
    lignes = fp.readlines()


# Pour chaque ligne lu dans le fichier, on la découpe et on insère les champs dans la base de données
for ligne in lignes :
    infos = ligne.split(";",10)
    # insertion des données dans la table Ports
    curseur.execute("INSERT INTO Ports (ID, GlobalPortID, WorldPortNumber, MasterPOID, PortName, "
                                       "AlternativeName, Status, Country, Latitude, Longitude) "
                    "VALUES(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)",
                    (infos[0],infos[1], infos[2], infos[3], infos[4],
                     infos[5], infos[6], infos[7], infos[8].replace(",","."), infos[9].replace(",",".")))


# La méthode commit () : enregistre toutes les modifications apportées depuis le dernier commit dans la base de données.
connexionBD.commit()
print ("... importation terminée !")
```

Annexe 4 :
Fichier de sauvegarde de la base de données (dump).

```
--
-- Data for Name: berth; Type: TABLE DATA; Schema: public;
--

COPY public.berth (id, port_id, terminal_id, name, no, operator, type, status, facility_type, latitude, longitude, globalportid,
globalterminalid, globalberthid) FROM stdin;
57212   10003   0       Tanker Berth                            Never Existed   Tanker  -8.501667       179.19495
PO619700T000000B000000  PO619700T000000B000000  PO619700T000000B057212\n
59133   10003   39032   Vaiaku                  Wharf   Active  Dry Cargo       -8.503683       179.19482       PO619700T000000B000000
PO619700T039032B000000  PO619700T039032B059133\n
89650   10012   50448   No 01           Stena Line / Irish Ferries      Pier    Active  Dry Cargo       52.25455        -6.338
PO188600T000000B000000  PO188600T050448B000000  PO188600T050448B089650\n
89651   10012   50448   No 02           Stena Line / Irish Ferries      Pier    Active  Dry Cargo       52.254665       -6.3377
PO188600T000000B000000  PO188600T050448B000000  PO188600T050448B089651\n
89652   10012   50448   No 03           Stena Line / Irish Ferries      Quay    Active  Dry Cargo       52.255432       -6.3355665
PO188600T000000B000000  PO188600T050448B000000  PO188600T050448B089652\n
89653   10012   50448   No 04           Stena Line / Irish Ferries      Quay    Active  Dry Cargo       52.25655        -6.337233
PO188600T000000B000000  PO188600T050448B000000  PO188600T050448B089653\n
89697   10012   50448   No 05           Stena Line / Irish Ferries      Quay    Active  Dry Cargo       52.253315       -6.339617
PO188600T000000B000000  PO188600T050448B000000  PO188600T050448B089697\n
81985   10016   47449   No 01           Petroperu SA    MBM     Active  Tanker  -6.9669833      -79.86132       PO236500T000000B000000
PO236500T047449B000000  PO236500T047449B081985\n
73674   10021   44151   Concreate Pier                  Pier    Active  Dry Cargo       19.864117       -90.52917
PO332300T000000B000000  PO332300T044151B000000  PO332300T044151B073674\n
73675   10021   44152   Pemex Lerma                     Pier    Active  Tanker  19.81565        -90.591866      PO332300T000000B000000
PO332300T044152B000000  PO332300T044152B073675\n
56777   10028   37489   SPM             Ras Al Khaimah Gas Commission (RAKGAS Co)       SPM     Active  Tanker  25.970984
55.935165       PO619800T000000B000000  PO619800T037489B000000  PO619800T037489B056777\n
32938   10032   18638   Kanokawa No. 1 Berth (Mitsubishi)               Kanokawa Terminal       Jetty   Active  Tanker  34.183
132.43466       PO380300T000000B000000  PO380300T018638B000000  PO380300T018638B032938\n
32939   10032   18638   Kanokawa No. 3 Berth (Mitsubishi)               Kanokawa Terminal       T Jetty Active  Tanker  34.178165
132.43367       PO380300T000000B000000  PO380300T018638B000000  PO380300T018638B032939\n
```