SAÉ 2.04 BDD | Rapport

Table of Contents

Projet réalisé part FANTUZ CHROBOT Léo, groupe A, fondateur de "One Two Script".

Présentation du projet

Savoir importer, comprendre et manipuler des données sont des éléments essentiels dans le domaine de la base de données. Cette SAE réunit tous ces éléments afin d'apprendre de la meilleure des manière la manipulation d'une base de données, avec ici des données issues de Parcoursup.

1. Analyse du fichier récupéré

- **1. Combien y-a t-il de lignes?**
    - Grâce à la commande `wc -l fr-esr-parcoursup.csv`, nous pouvons constater que le fichier contient **13870 lignes**.
- **2. Que représente une ligne ?**
    - Une ligne représente une **formation** comportant ses informations comme sa région, sa filière, ses effectifs, ses candidats et bien plus encore. Cela peut se justifier en regardant la première ligne du fichier à l'aide de la commande `head -1 fr-esr-parcoursup.csv`. De plus, nous verrons par la suite que le code de formation situé à la 110 colonnes est unique pour chacune des 13869 lignes de données.
- **3. Combien y-a t-il de colonnes ?**
    - Dans ce fichier, en utilisant la commande `head -1 fr-esr-parcoursup.csv | tr ";" "\n" | wc -l`, nous pouvons constater que ce dernier comporte **118 colonnes**.

- **4. Quelle colonne identifie un établissement ?**
  - En utilisant la commande `head -1 fr-esr-parcoursup.csv`, nous pouvons voir que la colonne permettant d'identifier un établissement est la colonne **numéro 3**, se nommant Code UAI de l'établissement. Chaque établissement dispose donc d'un code unique.
- **5. Quelle colonne identifie une formation ?**
  - En utilisant la commande `head -1 fr-esr-parcoursup.csv`, nous pouvons voir que la colonne permettant d'identifier une formation est la colonne **numéro 110**, se nommant cod_aff_form. Comme cité précédemment, nous verrons par la suite que cette colonne est unique pour chacune des lignes du fichier.
- **6. Combien de lignes font référence à notre BUT Informatique ?**
  - À l'aide de la commande `cat fr-esr-parcoursup.csv | grep -e "BUT - Informatique" | wc -l`, nous pouvons constater qu'il y a **49 lignes** faisant référence aux BUT Informatique en France.
  - À l'aide de la commande `cat fr-esr-parcoursup.csv | grep -e "BUT - Informatique" | grep -e "Villeneuve-d□ascq" | wc -l`, nous pouvons constater qu'il y a **1 ligne** faisant référence à notre BUT Informatique.
- **7. Quelle colonne identifie un département ?**
  - Grâce à la commande `head -11 fr-esr-parcoursup.csv`, nous pouvons remarquer que la colonne permettant d'identifier un département est la colonne **numéro 5**, se nommant Code départemental de l'établissement.
- **8. Comment envisagez vous d'importer ces données ?**
  - Pour commencer, l'objectif est de vérifier que toutes les lignes soient importables et supprimer celles inimportables à l'aide de la commande `sed -i.old -e "1d" fr-esr-parcoursup.csv` (où "1" est égal au numéro de la ligne à supprimer, et "d" représente une suppression de ligne). Puis, la création d'une table import sera nécessaire à l'aide d'un script java (afin de ne pas perdre trop de temps) avec le type "TEXT" pour toutes les colonnes. Pour finir, certaines colonnes pourraient voir leur type changer afin de garder une certaine cohérence vis-à-vis des données importées.
- **9. Quels problèmes identifiez vous dans ces données initiales ?**
  - Pour commencer, certaines lignes ont des colonnes vides ou des colonnes, à première vue, correspondantes à des nombres, mais comportant du texte sur certaines lignes, ce qui peut poser problème lors de l'importation. Ensuite, certaines lignes peuvent être répétitives, donc inutiles. Pour finir, certaines colonnes peuvent être calculées à l'aide d'autre colonnes, donc inutiles également.

Maintenant, nous aimerions générer un fichier dico.xls afin d'obtenir la première ligne de notre fichier représentant ainsi le nom de chaque colonne, ainsi que la numérotation de chacune d'entre elles ainsi qu'une ligne exemple.

Cela peut se faire dans un premier temps à l'aide de la commande `head -2 fr-esr-parcoursup.csv > dico.xls` (pour la première ligne ainsi que notre ligne exemple).

Pour la partie numérotation des colonnes, un script java fut créée afin de générer plus facilement cette partie du dico.

Nous avons ensuite fait une redirection du résultat à l'aide de la commande `java -cp bin Dico >>`

`dico.xls`.

Maintenant, passons à la création de la table import permettant l'importation des données dans notre base. Comme dit précédemment, un script java fut crée afin de générer rapidement la table import.

Pour finir, l'importation des données a pu se faire à l'aide de la commande `\copy import FROM fr-esr-parcoursup.csv DELIMITER ";";` (à exécuter dans le même fichier .sql que celle pour la table import, ou directement dans la base de données).

NOTE	Il est bon de mentionner que cette importation a été réalisée après avoir supprimé la première ligne du fichier original (comportant ainsi le nom de chaque colonne, donc inutile car n'étant pas des données) grâce à la commande `sed -i.old -e "1d" fr-esr-parcoursup.csv`.

Nous pouvons désormé analyser plus précisément les données en répondant à quelques questions que nous pouvons nous poser:

(a) Combien il y a de formations gérés par ParcourSup ? Il y a 13869 formations gérées sur Parcousup. Nous pouvons définir cela grâce à la requête suivante :

```
SELECT count(DISTINCT n110) FROM import;
```

(b) Combien il y a d'établissements gérés par ParcourSup ? À l'aide de la requête suivante, nous savons qu'il y a 3602 établissements gérés par ParcourSup:

```
SELECT count(DISTINCT n4) FROM import;
```

(c) Combien il y a de formations pour l'université de Lille ? Graĉe à la requête qui suit, nous pouvons en déduire qu'il y a 124 formation à l'université de Lille :

```
SELECT DISTINCT count(*) FROM import WHERE n4 = 'Université de Lille';
```

(d) Combien il y a de formations pour notre IUT ? Il y a 14 formations pour notre IUT. Cela peut être déduit à l'aide de la reqûete suivante :

```
SELECT DISTINCT count(*) FROM import WHERE n8 = 'Lille' AND n4 LIKE 'IUT%';
```

(e) Quel est le code du BUT Informatique de l'unversité de Lille ?

Le code UAI de l'établissement de notre formation est 0597215X :

```
SELECT DISTINCT n3 FROM import WHERE n8 = 'Lille' AND n9 LIKE 'Villeneuve%' AND n10 = 'BUT - Informatique';
```

Le code de notre formation est 6888 :

```
SELECT DISTINCT n110 FROM import WHERE n8 = 'Lille' AND n9 LIKE 'Villeneuve%' AND n10 = 'BUT - Informatique';
```

(f) Citez 5 colonnes contenant des valeurs nulles: Parmis les colonnes contenant des valeurs nulles, nous avons n16, n23, n38, n39 et n55 (où "n..." représente le numéro de la colonne).

2. Ventilation des données

Ventiler les données peut s'avérer utile afin d'éviter les redondances ou les colonnes pouvant être calculées. Ainsi, j'ai décidé de créer 5 tables permettant cette ventilation :

Etablissements(UAI(PK), codeDep(PK), commune(PK), nom, status)

Lieu(#UAI(PK), #codeDep(PK), #commune(PK), departement, region, academie)

Formations(codeFormation(PK), selectivite, filiereDetaillee, filiere, filiereDetailleeBis, filiereTresDetaillee, effectifTotal, candidates, generaux, techno, professionnels, coordonnees, capaEtablissement, lienSite, effectifPhasePrinc, #UAI, #codeDep, #commune)

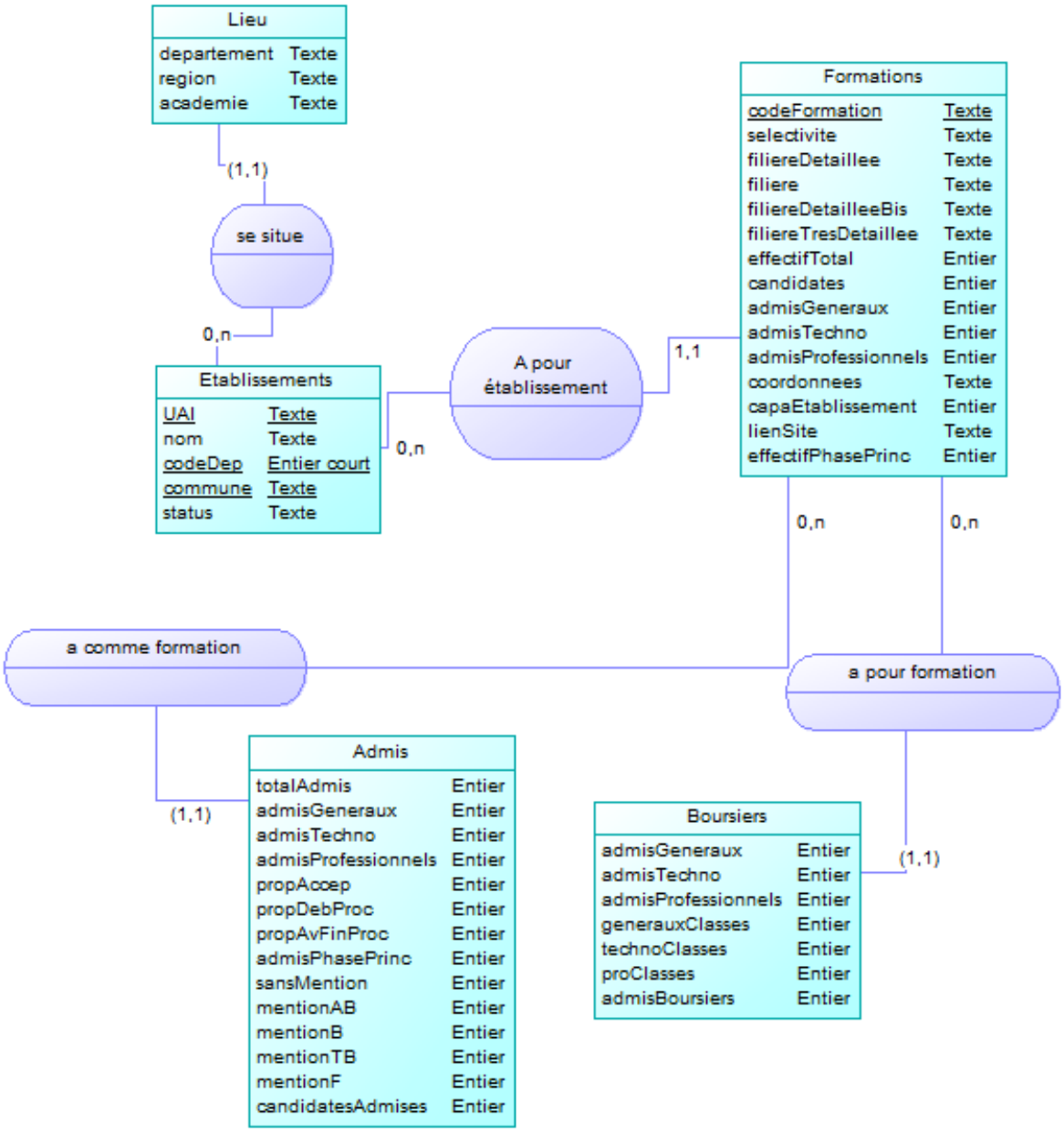
Admis(#formation(PK), totalAdmis, generaux, techno, professionnels, propAccep, propDebProc, propAvFInProc, admisPhasePrinc, sansMention, mentionAB, mentionB, mentionTB, mentionF, candidatesAdmises)

Boursiers(#formation(PK), generaux, techno, professionnels, generauxClasses, technoClasses, proClasses, admisBoursiers)

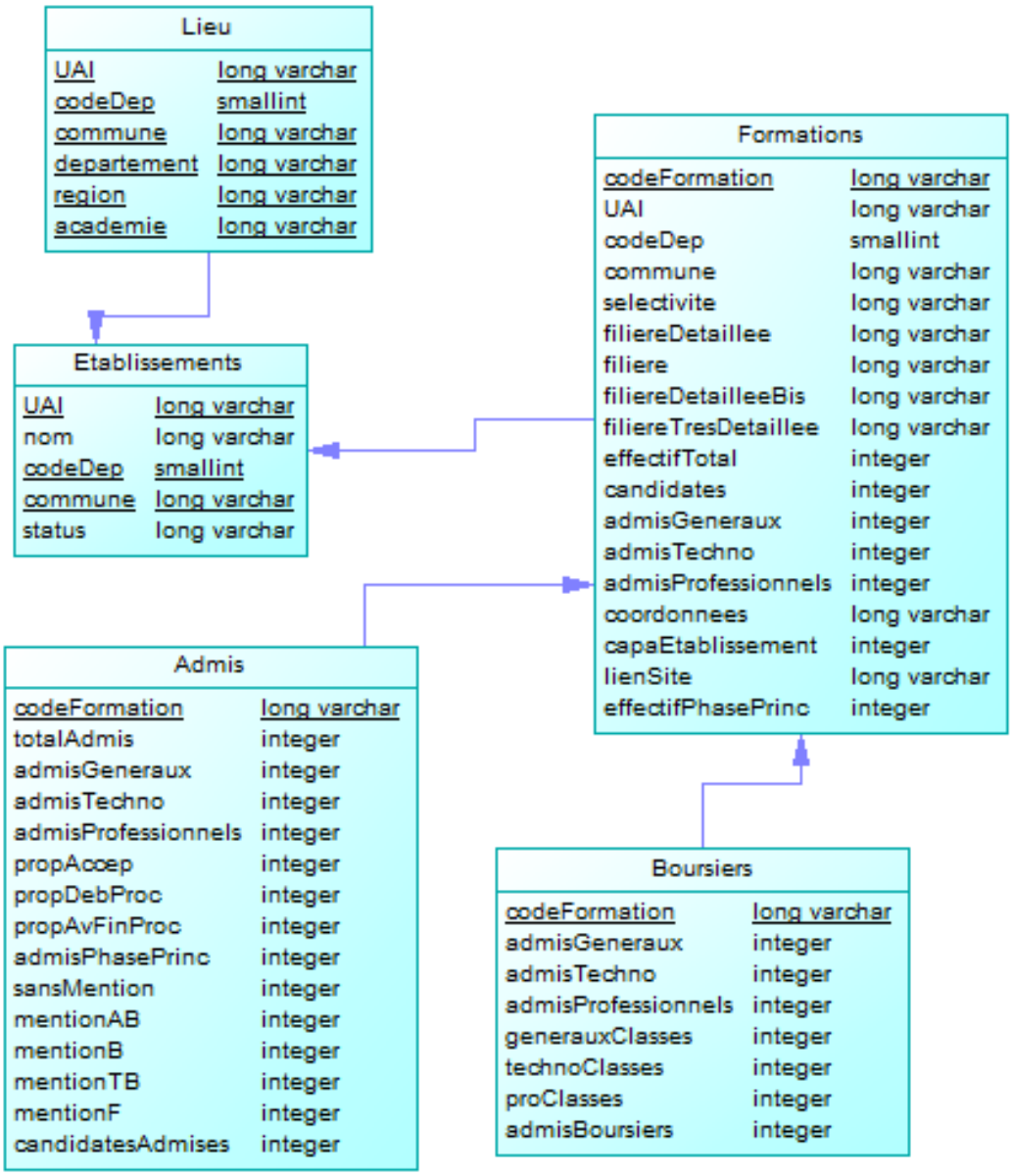
Il faut comprendre la modélisation des tables de la façon suivante:

- Chaque lieu contient un ou plusieurs etablissements.
- Chaque établissements comporte une ou plusieurs formations.
- Chaque formation comporte des informations concernant les admis de cette dernière ainsi que les boursiers.

Ainsi, voici le MCD associé à cette base:



Cela nous à ainsi généré le MPD suivant:



2.1. Quelques informations

Voici quelque information concernant mes décisions de tables ventilées:

J'ai décidé de créer 5 tables qui à première vue peuvent paraître évidentes. Une table "Etablissements" regroupant les établissements ainsi que certaines informations sur ces derniers tels que leur nom ou statut, par exemple. Pour sa clé primaire, malgré le code unique "UAI" permettant d'identifier chaque établissement, je me suis rendu compte que chaque établissement était présent plusieurs fois, mais parfois avec un département et/ou une commune différente. En effet, si nous prenons l'exemple de l'université de Lille, nous pouvons nous rendre compte que cet établissement est présente dans plusieurs commune, comme Villeneuve d'Ascq et bien plus. Ainsi, j'ai décidé de mettre sous clé primaire la colonne UAI, ainsi que la colonne codeDep et commune afin d'avoir une meilleure précision concernant la géolocalisation de nos formations. Nous avons également une table "Lieu" comportant quelques informations concernant l'académie, le département et la région des établissements (ainsi présente en tant que clé primaire et étrangère dans la table Lieu). Ensuite, une table "Formations" permettant de regrouper la plupart des colonnes importantes concernant les formations, comme leur code (ainsi unique, donc clé primaire), ainsi que certaines autres informations comme la sélectivité ou la filière, par

exemple. Nous avons ensuite une table "Admis" comportant une clé primaire et étrangère "formation" afin de regrouper toutes les informations les plus importantes pour chaque formation concernant les personnes admises, tel que les mentions, la proportion d'admis selon leur bac, et bien plus. Pour fini, nous avons une table "Boursiers" comportant une clé primaire et étrangère "formation" afin de regrouper toutes les formations les plus importantes concernant les boursiers admis, tel que le total d'admis ou encore le nombre d'admis en fonction de leur bac, par exemple. Seules les colonnes les plus importantes et utiles ont été ajoutées. Ainsi certaine colonne jugées "inutiles" n'ont pas été ajoutées afin d'éviter une surcharge des tables. Aussi, certaines colonnes peuvent être recalculées à l'aide des colonnes ajoutées dans les tables ventilées, notamment les colonnes représentant des proportions ou taux (autrement dit, des pourcentages).

Cette partie de la SAE fut compliquée, car il est difficile de choisir les colonnes utiles et d'en faire des tables cohérentes. Ainsi, cette partie m'a pris plusieurs heures de compréhension et de travail, donnant ainsi le résultat ici présent.

Par la suite, j'ai ainsi créé un fichier nommé "parcoursup.sql" contenant toutes les requêtes et commandes permettant de créer toutes les tables (également la table "import"), ainsi que les commandes d'importation et modification du fichier (afin d'enlever la première ligne inutile), ainsi que les affectations de clé primaire et étrangère (cf parcoursup.sql).

Afin de mieux connaître le fichier ainsi que les tables de notre base, voici certaines questions pouvant être posées :

- **1. Quelle taille en octet fait le fichier récupéré ?**
  - La taille du fichier récupéré est de 12.4 Mo. Cela peut être justifié à l'aide de la commande suivante dans le terminal de commande à l'emplacement du fichier: `stat fr-esr-parcoursup.csv`
- **2. Quelle taille en octet fait la table import ?**
  - À l'aide de la requête `SELECT pg_total_relation_size("import");`, nous pouvons en déduire que la table import fait 17 014 784 octets.
- 3. **Quelle taille en octet fait la somme des tables créées ?**
  - À l'aide de la commande précédente (en modifiant "import" par le nom de nos différentes tables), nous pouvons en déduire que la table "Etablissements" fait 802 816 octets, la table "Formations" fait 5 619 712 octets, la table "Admis" fait 1 998 848 octets et la table "Boursiers" fait 1 318 912 octets, soit un total de 9 740 288 octets (environ 9.74 Mo).
- **4. Quelle taille en octet fait la somme des tailles des fichiers exportés correspondant à ces tables ?**
  - Afin d'exporter ces tables, nous pouvons utiliser la requête `\copy nom_table TO chemin_vers_fichier DELIMITER ";";` (en modifiant "table_name" et "chemin..." par les informations souhaitées). Nous savons ainsi que le fichier de la table "Etablissements" fait 277.7 ko, la table "Formations" fait 4.9 Mo, la table "Admis" fait 553.2 ko, la table "Boursiers" fait 4.9 Mo et la table "Lieu" fait 238.4 ko. Ainsi, toutes les tables (une fois exportées sous format "CSV") font environ 10 819 300 octets (soit environ 10.8 Mo).

3. Requêtage des données

Pour finir, nous avons évidemment manipulé ces données. Ainsi, toutes les requêtes réalisées pour cette partie sont disponibles dans le fichier "requetes.sql" de manière très détaillées et commentées. Ainsi, toutes les explications concernant les requêtes ainsi que leur fonctionnement et précisions supplémentaires sont à retrouver dans ce fichier. Il s’agit là de la partie la plus dure de cette SAE selon moi, car comprendre les données fut une réelle difficulté pour moi. Cependant, toutes les requêtes sont tout de même réussi et cela m’a permis d’améliorer ma compréhension d’une base quelconque de données.