SAE 2.04

EXPLOITATION D'UNE BASE DE DONNEES

VANHOUTTE Amaury | LECOCQ Dylan

parcoursup
Entrez dans l'enseignement supérieur

IUT | Université de Lille

Partie 1 : Comprendre les données

Q1.1) Pour connaître le nombre de ligne d'un fichier sous Unix, on utilise la commande **wc** avec l'option **-l**. Celle-ci nous indique que le fichier contient 13 869 lignes.

Q1.2) Dans ce fichier, une ligne correspond à une formation, proposée dans un établissement précis, avec les informations correspondantes (type de la formation, nom et localisation de l'établissement, capacité d'accueil…).

Q1.3) On utilise la commande suivante afin de connaître le nombre de colonnes du fichier : **awk -F ";" 'NR==1 {print NF; exit}' data.csv,** grâce à celle-ci, on obtient 118. Le fichier contient donc 118 colonnes.

Q1.4) Dans ce fichier, un établissement est identifié grâce à la colonne numéro 3 intitulée **cod_uai** et qui corresponds au code d'Unité Administrative Immatriculée, qui est unique à chaque établissement.

Q1.5) Les formations sont quant à elles identifiées grâce à la colonne numéro 110 intitulée **code_aff_form**, qui corresponds au code de la formation, qui est unique à chaque formation.

Q1.6) On utilise une requête SQL afin de connaître le nombre de ligne mentionnant le BUT Informatique. La requête est la suivante : **SELECT COUNT(*) FROM import WHERE n13 LIKE '%BUT - Informatique%';**
Le BUT Informatique est mentionné dans 49 lignes.

Q1.7) Les départements sont identifiés grâce à la 5ème colonne du fichier, intitulée « dep », qui correspond au code de département qui est unique.

Q1.8) Pour importer ces données, il va tout d'abord falloir créer une table ayant le bon nombre de colonnes, chacune du type TEXT dans un premier temps. Puis utiliser la commande \copy de PSQL afin d'importer les données du fichier CSV dans la table désirée. Une fois que les données seront importées, il faudra procéder à une modification des types de données de chaque colonne afin de les rendre le plus restrictif possible.

Q1.9) Le fichier initial comporte des valeurs nulles qui peuvent être mal interprétée lors de l'import et ainsi empêcher celui-ci de s'exécuter correctement. Les noms de colonnes n'étant pas forcément explicite, la compréhension des données est parfois complexe. De plus, le grand nombre de colonnes peut rendre plus difficile la création d'une table qui en possède autant.

Q2.1) Afin de réaliser le fichier dico.xls permettant la correspondance entre les numéros de colonnes et les noms du fichier, nous avons réalisé un script en Bash qui lit chaque colonne sur la ligne d'entête du fichier data.csv, y associe un nombre et écrit le résultat dans le fichier xls.

Q2.3) Afin de s'assurer que les types des colonnes soient les plus restrictifs possibles, on base celle-ci sur la donnée la plus grande contenue dans celle-ci. Ainsi, une colonne comportant des chaines de caractère verra son type déterminé en fonction de la longueur de la plus longue chaine qu'elle contient.

Q2.5) a- Pour connaitre le nombre de formations gérées par ParcourSup, on utilise la requête suivante : **SELECT COUNT(DISTINCT n110) FROM import;** Cette requête renvoie 3207, on peut donc conclure que ParcourSup gère 13 869 formations.

b- De la même façon, pour connaitre le nombre d'établissements gérés par la plateforme, on utilise la requête suivante : **SELECT COUNT(DISTINCT n3) FROM import;** On obtient 3965, ce qui signifie que 3965 établissement sont gérés par ParcourSup.

c- Afin d'obtenir le nombre d'établissements faisant partie de l'Université de Lille, on saisit la requête suivante : **SELECT Count(*) FROM import WHERE n4 LIKE '%Université de Lille%';** Celle-ci nous donne le résultat 146. Donc 146 établissements de l'Université de Lille sont présents sur ParcourSup.

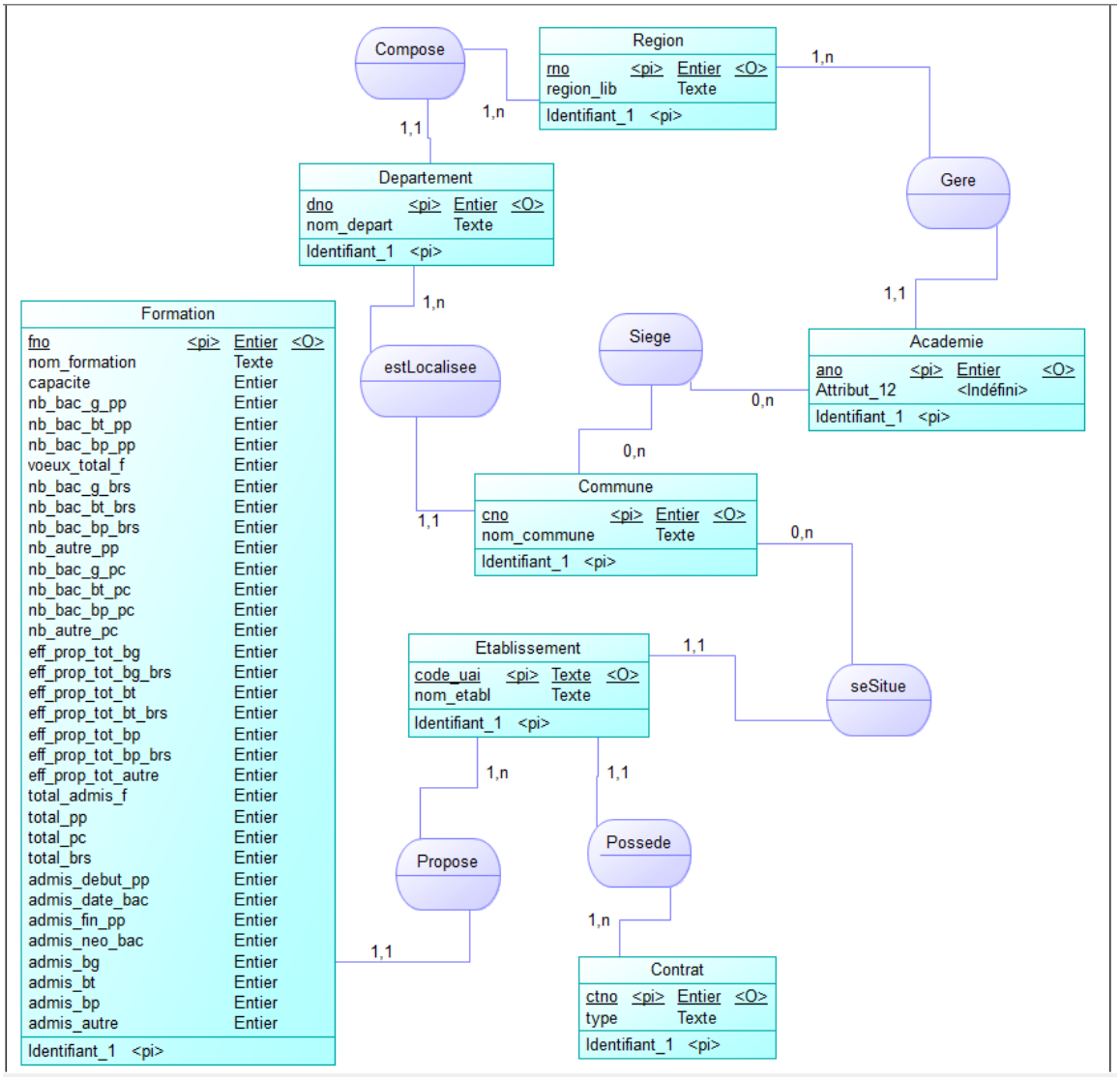
d- Les formation dispensées à l'IUT peuvent être obtenues grâce à une requête s'appuyant sur les colonnes n9 qui corresponds à la ville où est située l'emplacement ainsi que la colonne n4 qui identifie l'établissement, comme par exemple la requête ci-dessous : **SELECT Count(*) FROM import WHERE n9 LIKE 'Villeneuve%' AND n4 LIKE 'Institut%Université de Lille';** On obtient le résultat 10, on peut donc en déduire que l'IUT propose 10 formations différentes.

e- Pour obtenir le code du BUT Informatique de l'IUT, on peut réaliser une requête sur les champs n10 qui corresponds à l'intitulé de la formation ainsi que n9 qui corresponds à la ville ou est situé l'établissement, comme suit : **SELECT n110 FROM import WHERE n10 = 'BUT - Informatique' AND n9 LIKE 'Villeneuve%';** Cette requête renvoie 6888. On conclut donc que le code du BUT Informatique de l'IUT est 6888.

f- Pour obtenir les différents champs qui contiennent des valeurs nulles, nous avons tout d'abord analysé le fichier CSV puis effectué une requête portant sur les champs que nous avions identifié. Nous avons donc utilisé la requête suivante : **SELECT DISTINCT n16, n22, n37, n38, n54 FROM import WHERE n16 IS NULL OR n22 IS NULL OR n37 IS NULL OR n38 IS NULL OR n54 IS NULL;** Les colonnes contenant des valeurs nulles sont donc les colonnes n16, n22, n37, n38 et n54.

Partie 2 : Ventiler les données.

Q1) a- Voici le MCD correspondant à la structuration que nous avons choisi :



Q2) a- Pour connaitre la taille du fichier téléchargé, on utilise la commande **wc -c data.csv** on obtient le résultat suivant : *12417716 data.csv On peut donc en déduire que le fichier data.csv a une taille de 12 417 716 octets.*

b- Pour connaitre la taille en octet d'une table avec PostgreSQL, on peut utiliser la fonction **pg_total_relation_size('import') ;** Ce qui nous indique que la table import a une taille de 28 876 800 octets.

c- On calcule la somme des tailles des différentes tables créées avec la commande : **SELECT pg_total_relation_size('Region') + pg_total_relation_size('Departement') + pg_total_relation_size('Commune') + pg_total_relation_size('Academie') + pg_total_relation_size('Etablissement') + pg_total_relation_size('Contrat') + pg_total_relation_size('Formation') AS Total;** Celle-ci nous donne 8241152, donc l'ensemble des tables a une taille totale de 8 241 152 octets.

d- Pour calculer la taille totale des fichiers CSV créés à partir des tables, on utilise la commande **wc -c Academie.csv Formation.csv Commune.csv Contrat.csv Region.csv Departement.csv Etablissement.csv** qui nous donne un total de 5 944 540 octets.

Partie 3 : Requêtage.

Q1) La colonne n56 (acc_neobac) est le total de l'addition des colonnes n57(acc_bg), n58(acc_bt) et n59(acc_bp). On peut donc recaculer celle-ci grâce à la requête suivante : **SELECT n56, n57 + n58 + n59 AS Recalcul FROM import;**

Q2) Pour vérifier que ce calcul est exact, on peut faire l'union entre les deux requêtes afin de vérifier s'il y a des différences entre les valeurs, on procède donc ainsi : **SELECT n56 FROM import EXCEPT SELECT n57 + n58 + n59 AS Recalcul FROM import;**

Q3) La colonne n74(pct_acc_debutpp) représente le % d'admis ayant reçu leur proposition d'admission le 27 mai (ouverture phase principale) elle peut être recalculée à partir des colonnes n51(acc_debutpp) qui représente le nombre d'admis ayant reçu leur proposition d'admission le 27 mai et n47(acc_tot) qui représente le nombre d'admis total, on divise donc n51 par n47. On peut procéder ainsi : **SELECT n74, ROUND((n51/n47)*100) AS Recalcul FROM import WHERE n47 != 0;**

Q4) Pour vérifier que ce calcul est exact, on procède de la même manière que pour la Q2 en utilisant l'opérateur EXCEPT : **SELECT n74 FROM import EXCEPT SELECT ROUND((n51/n47)*100) AS Recalcul FROM import WHERE n47 != 0;**

Q5) La colonne n76(pct_acc_finpp) représente le % d'admis ayant reçu leur proposition d'admission avant la fin de la procedure principale, on peut la recalculer à partir des colonnes n47(acc_tot) qui représente le nombre d'admis total et la colonne n53(acc_finpp) On divise donc n53 par n47 avec la requête suivante : **SELECT n76, ROUND((n53/n47)*100) AS Recalcul FROM import WHERE n47 != 0;**
Les données obtenues avec cette requête sont exactes à une décimale près.

Q6) Pour réaliser le même calcul sur nos tables ventilées, on utilise la commande suivante : **SELECT n76, ROUND(CAST(admis_fin_pp AS NUMERIC)/(admis_bg + admis_bt + admis_bp + admis_autres)*100) FROM formation INNER JOIN import ON code_formation = n110 WHERE (admis_bg + admis_bt + admis_bp + admis_autres) != 0;**

Q7) La colonne n81(pct_bours) représente le % d'admis néo bacheliers boursiers. On le recalcule à partir des colonnes n55(acc_brs) qui représente le nombre de néo-bacheliers boursiers admis et la colonne n56(acc_neobac) qui réprésente le nombre total de néo bacheliers admis. On utilise la requête suivante pour le recalculer : **SELECT n81, ROUND((CAST(n55 AS NUMERIC)/n56)*100) AS Recalcul FROM import WHERE n56 != 0;**
Il est nécessaire de convertir au moins l'une des deux valeurs de n55 ou n56 afin de pouvoir réaliser la division euclidienne. On arrondit donc le résultat à l'entier le plus proche avec la fonction ROUND().
Les données obtenues avec cette requête sont exactes à une décimale près.

Q 8) Pour réaliser le même calcul sur nos tables ventilées, on utilise la requête suivante :
**SELECT n81, ROUND(CAST(total_boursiers AS NUMERIC)/admis_neobac*100)
FROM formation INNER JOIN import ON code_formation = n110 WHERE
admis_neobac != 0;**