Transformation d'un tableau en base de données relationnelle

De nombreuses données sont gérées sous des feuilles de calcul, sous la forme de simples tableaux. C'est bien souvent le cas pour des applications de petite taille, ou lorsque les créateurs/utilisateurs n'ont pas les compétences ou connaissances nécessaires à la conception et exploitation d'une base de données normalisée. C'est au moment où les besoins évoluent, où les volumes augmentent, où les personnes se renouvèlent, où les bugs apparaissent, ... que la décision est bien souvent prise d'une migration vers un SGBD.

Dans cet exemple, on considère les données de résultats du premier tour des élections présidentielles 2017, par bureau de vote, disponibles ici : https://public.opendatasoft.com/explore/dataset/election-presidentielle-2017-resultats-par-bureaux-de-vote-tour-1/information/?disjunctive.libelle_de_la_commune

Plus particulièrement on va s'intéresser aux résultats pour les bureaux de votes de la région Auvergne-Rhône-Alpes composée des départements de l'Ain (01), Allier (03), Ardèche (07), Cantal (15), Drôme (26), Isère (38), Loire (42), Haute-Loire (43), Puy-de-Dôme (63), Rhône (69), Savoie (73) et Haute-Savoie (74).

Première séance

1. Étudiez les données. Dans un fichier texte, pour chaque colonne, décrivez la signification de l'attribut, son rôle dans les données, éventuellement la façon dont il est construit.
2. Importez ces données (disponibles en CSV – pour chaque département – depuis la page WEB) sous PostgreSQL dans une relation unique nommée « election-csv », avec des types de données appropriés (voir le script ci-dessous).
3. Faites l'inventaire des DF qui vous paraissent « plausibles » dans la sémantique de ces données ; vérifiez qu'elles soient bien satisfaites grâce à des requêtes SQL.

Deuxième séance

4. Dressez un schéma Entité-Association modélisant les données. Traduisez en relationnel. Étudiez la forme normale de la modélisation proposée.
5. Implémentez la base de données correspondante, avec les contraintes appropriées. Migrez les données par des requêtes SQL qui seront proprement conservées dans la documentation de la migration.

Troisième séance

6. Faites une fonction qui permette, pour le nom d'un candidat, d'afficher ses résultats proprement : score en nombre de voix et en pourcentage par bureau de vote, par commune, par circonscription, par département.
7. Faites une fonction qui calcule la similarité entre chaque département en fonction des votes exprimés.
8. Écrivez une requête qui, pour chaque candidat, retourne le bureau de vote, la commune où il a fait son meilleur score (retournez aussi le score).

9. Résumez l'ensemble de votre import au sein d'un seul script ; une fois lancé, celui-ci crée ou récrée la base de données, la peuple à partir du fichier CSV, et affiche finalement les résultats par la fonction précédente.

Modalités d'évaluation du TP

Le TP est sur 3 séances ; chaque **binôme** rendra deux fichiers au plus tard le 13 décembre 2019, en utilisant TOMUSS :

- Un fichier « conception.pdf » dans lequel on trouvera les réponses aux questions 3 et 4.
- Un fichier « script.sql » dans lequel on trouvera le script de la question 9.

Script de création de la table « election-csc » pour chargement des données brutes

```
CREATE TABLE "election-csv" (
"Code du département" DECIMAL NOT NULL,
"Département" VARCHAR(5) NOT NULL,
"Code de la circonscription" DECIMAL NOT NULL,
"Circonscription" VARCHAR(21) NOT NULL,
"Code de la commune" DECIMAL NOT NULL,
"Commune" VARCHAR(33) NOT NULL,
"Bureau de vote" DECIMAL NOT NULL,
"Inscrits" DECIMAL NOT NULL,
"Abstentions" DECIMAL NOT NULL,
"% Abs/Ins" DECIMAL NOT NULL,
"Votants" DECIMAL NOT NULL,
"% Vot/Ins" DECIMAL NOT NULL,
"Blancs" DECIMAL NOT NULL,
"% Blancs/Ins" DECIMAL NOT NULL,
"% Blancs/Vot" DECIMAL NOT NULL,
"Nuls" DECIMAL NOT NULL,
"% Nuls/Ins" DECIMAL NOT NULL,
"% Nuls/Vot" DECIMAL NOT NULL,
"Exprimés" DECIMAL NOT NULL,
"% Exp/Ins" DECIMAL NOT NULL,
"% Exp/Vot" DECIMAL NOT NULL,
"N°Panneau" DECIMAL NOT NULL,
"Sexe" VARCHAR(1) NOT NULL,
"Nom" VARCHAR(13) NOT NULL,
"Prénom" VARCHAR(8) NOT NULL,
"Voix" DECIMAL NOT NULL,
"% Voix/Ins" DECIMAL NOT NULL,
"% Voix/Exp" DECIMAL NOT NULL,
"Code Insee" DECIMAL NOT NULL,
"Coordonnées" VARCHAR(19),
"Nom Bureau Vote" VARCHAR(70),
"Adresse" VARCHAR(48),
"Code Postal" DECIMAL,
"Ville" VARCHAR(33),
uniq_bdv VARCHAR(85)
);
```

Requête de vérification de satisfaction d'une dépendance fonctionnelle R :X --> Y

```
SELECT COUNT(*)
FROM (SELECT DISTINCT X
FROM r)
EXCEPT
SELECT COUNT(*)
FROM (SELECT DISTINCT X, Y
FROM r)
```

Le résultat est VIDE si la DF est vérifiée (CF TD2.5)