Projet de bases de données

Réalisé par **Julien GIRAUD** et **Léa VIGLIANO**

Question 1 et 2

Nous avons rassemblé toutes les données dans un seul fichier CSV. Le code permettant de créer la table est le suivant :

```
-- Création de la base
drop table if exists "election-csv";
CREATE TABLE "election-csv" (
    "Code du département" DECIMAL NOT NULL,
    "Département" VARCHAR(20) NOT NULL,
    "Code de la circonscription" DECIMAL NOT NULL,
    "Circonscription" VARCHAR(30) NOT NULL,
    "Code de la commune" DECIMAL NOT NULL,
    "Commune" VARCHAR(40) 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(20) NOT NULL,
    "Prénom" VARCHAR(10) NOT NULL,
    "Voix" DECIMAL NOT NULL,
    "% Voix/Ins" DECIMAL NOT NULL,
    "% Voix/Exp" DECIMAL NOT NULL,
    "Code Insee" DECIMAL NOT NULL,
    "Coordonnées" VARCHAR(20),
    "Nom Bureau Vote" VARCHAR(120),
    "Adresse" VARCHAR(60),
    "Code Postal" DECIMAL,
    "Ville" VARCHAR(40),
    uniq_bdv VARCHAR(130)
);
```

```
-- Adresse du fichier à mettre à jour par l'utilisateur du script
copy "election-csv" from './data.csv' delimiter ';' csv header;
```

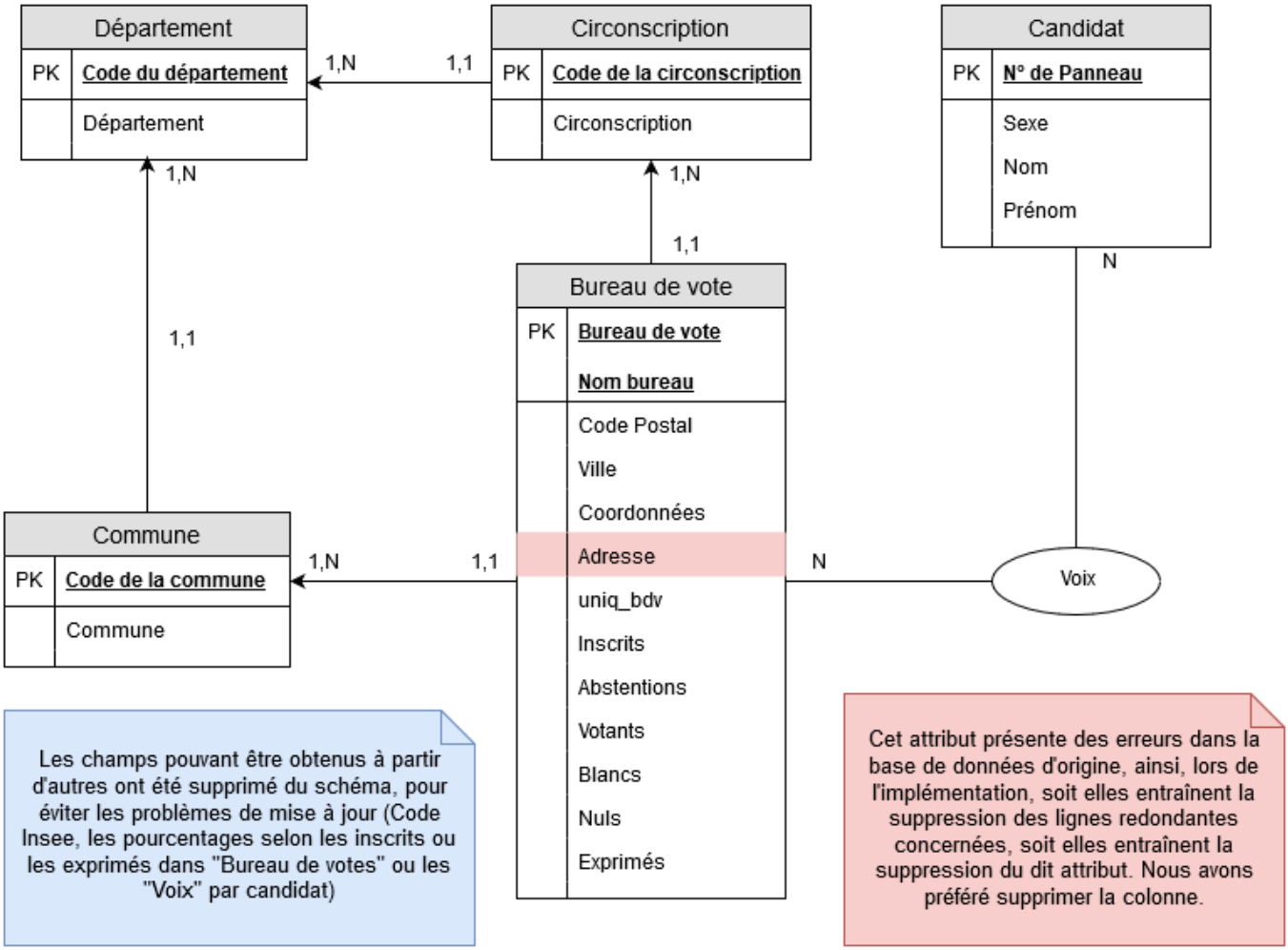
Question 3

Nous avons trouvé les DF suivantes :

```
{
    "Code du département" -> "Département"
    "Département" -> "Code du département"
    "Code de la circonscription" -> "Circonscription"
    "Code du département", "Code de la commune" -> "Commune", "Code Insee"
    "Code du département", "Code de la commune", "Bureau de vote", "Nom
Bureau vote" -> "Coordonnées", "Code Postal", "Ville", "Code de la
circonscription", "uniq_bdv", "Inscrits", "Abstentions", "% Abs/Ins",
"Votants", "% Vot/Ins", "Blancs", "% Blancs/Ins", "% Blancs/Vot", "Nuls",
"% Nuls/Ins", "% Nuls/Vot", "Exprimés", "% Exp/Ins", "% Exp/Vot"
    "N°Panneau" -> "Sexe", "Nom", "Prénom"
    "N°Panneau", "Code du département", "Code de la commune", "Bureau de
vote", "Nom Bureau vote" -> "Voix", "% Voix/Ins", "% Voix/Exp"
}
```

Question 4

Notre schéma entité/association est le suivant :



Cela correspond à ce model relationnel :

```
Candidat(_N°Panneau, Sexe, Nom, Prénom)
Departement(_Code du département, Département)
Circonscription(_Code de la circonscription, Circonscription)
Commune(_#Code du département, _Code de la commune, Commune)
Bureau(_#Code du département, _#Code de la commune, _#Bureau de vote, _Nom
Bureau Vote, Ville, Code Postal, #Code de la circonscription, Coordonnées,
uniq_bdv, Inscrits, Abstentions, Votants, Blancs, Nuls, Exprimés)
ScoreCandidat(_#Code du département, _#Code de la commune, _#Bureau de
vote, _#Nom Bureau Vote, _#N°Panneau, Voix)
```

Ce model passe les formes normales 1, 2, 3 si on enlève les pourcentages qui peuvent être calculés, FNBC

Question 5 - Création et remplissage des tables

```
-- #### Nettoyage du projet ####
do $nettoyage$ begin
    drop function if exists get_similarite();
    drop type if exists similarite_data;
    drop type if exists similarite;
    drop function if exists get_score(varchar, varchar);
    drop table if exists score;
```

```
    drop table if exists bureau;
    drop table if exists complement_commune;
    drop table if exists commune;
    drop table if exists candidat;
    drop table if exists circonscription;
    drop table if exists departement;
end $nettoyage$;

-- Table "Département"
drop table if exists departement;
create table departement (
    code decimal primary key,
    nom varchar(20) not null
);
insert into departement (
    select distinct "Code du département", "Département"
    from "election-csv" order by "Code du département"
);

-- Table "Circonscription"
drop table if exists circonscription;
create table circonscription (
    code decimal primary key,
    nom varchar(30) not null
);
insert into circonscription (
    select distinct "Code de la circonscription", "Circonscription"
    from "election-csv"
    order by "Code de la circonscription"
);

-- Table "Candidat"
drop table if exists candidat;
create table candidat (
    id decimal primary key,
    sexe varchar(1) not null,
    nom varchar(20) not null,
    prenom varchar(10) not null
);
insert into candidat (
    select distinct "N°Panneau", "Sexe", "Nom", "Prénom"
    from "election-csv"
    order by "N°Panneau"
);

-- Table "Commune"
drop table if exists commune;
create table commune (
    code_departement decimal references departement(code),
    code_commune decimal,
    commune varchar(40) not null,
    primary key (code_departement, code_commune)
);
insert into commune (
```

```
    select distinct "Code du département", "Code de la commune", "Commune"
    from "election-csv"
);

-- Table "Bureau"
update "election-csv" set "Nom Bureau Vote" = '' where "Nom Bureau Vote" is
null;
drop table if exists bureau;
create table bureau (
    code_departement decimal,
    code_commune decimal,
    code_bureau decimal,
    nom_bureau varchar(120),
    ville varchar(40),
    code_postal decimal,
    code_circonscription decimal references circonscription(code),
    coordonnees varchar(20),
    uniq_bdv varchar(130),
    inscrits decimal not null,
    abstentions decimal not null,
    votants decimal not null,
    blancs decimal not null,
    nuls decimal not null,
    exprimes decimal not null,
    foreign key (code_departement, code_commune) references
commune(code_departement, code_commune),
    primary key (code_departement, code_commune, code_bureau, nom_bureau)
);
insert into bureau (
    select distinct "Code du département", "Code de la commune", "Bureau de
vote", "Nom Bureau Vote",
    "Ville", "Code Postal", "Code de la circonscription", "Coordonnées",
"uniq_bdv",
    "Inscrits", "Abstentions", "Votants", "Blancs", "Nuls", "Exprimés"
    from "election-csv"
);

-- Table "Score Candidat"
drop table if exists score;
create table score (
    code_departement decimal,
    code_commune decimal,
    code_bureau decimal,
    nom_bureau varchar(120),
    id_candidat decimal references candidat(id),
    voix decimal,
    foreign key (code_departement, code_commune, code_bureau, nom_bureau
    ) references bureau(code_departement, code_commune, code_bureau,
nom_bureau),
    primary key (code_departement, code_commune, code_bureau, nom_bureau,
id_candidat)
);
insert into score (
    select distinct "Code du département", "Code de la commune",
```

```
    "Bureau de vote", "Nom Bureau Vote", "N°Panneau", "Voix"
    from "election-csv"
);
```

Question 6 - Affichage des scores

```
drop function if exists get_score(varchar, varchar);
create or replace function get_score(n varchar, p varchar) returns table (
    voix decimal,
    "% voix/inscrits" decimal,
    "% voix/exprimés" decimal,
    "% voix/votants" decimal,
    bureau varchar(140),
    commune varchar(40),
    circonscription varchar(30),
    departement varchar(20),
    code_bureau decimal,
    nom_bureau varchar(120),
    code_commune decimal,
    code_departement decimal
    ) as $get_score$
    begin
        return query
        select s.voix, s.voix/inscrits, s.voix/nullif(exprimes, 0),
s.voix/nullif(votants, 0),
        concat('Bureau ', s.code_bureau, nullif(concat(' - ',
s.nom_bureau), ' - '))::varchar(140),
        co.commune, ci.nom, d.nom, s.code_bureau, s.nom_bureau,
s.code_commune, s.code_departement
        from score s
        join candidat ca on s.id_candidat = ca.id
        join bureau b on s.code_departement = b.code_departement and
s.code_commune = b.code_commune and s.code_bureau = b.code_bureau
        join departement d on s.code_departement = d.code
        join circonscription ci on b.code_circonscription = ci.code
        join commune co on s.code_departement = co.code_departement and
s.code_commune = co.code_commune
        where ca.nom = n and ca.prenom = p
        order by d.nom, ci.nom, co.commune, s.code_bureau;
    end;
$get_score$ language plpgsql;

-- Quelques exemples d'utilisation
select * from get_score('LE PEN', 'Marine');
select * from get_score('MACRON', 'Emmanuel');
select * from get_score('MÉLENCHON', 'Jean-Luc');
select avg(voix) from get_score('LE PEN', 'Marine');
select avg(voix) from get_score('MACRON', 'Emmanuel');
select avg(voix) from get_score('MÉLENCHON', 'Jean-Luc');
```

Question 7 - Similarités entre départements

```
-- Type de la table du résultat de la fonction get_similarite()
drop type if exists similarite;
create type similarite as (
    "Département 1" varchar(30),
    "Département 2" varchar(30),
    "Similarité" decimal
);
-- Type des couples de département
drop type if exists similarite_data;
create type similarite_data as (
    d1 decimal, -- Code département 1
    n1 varchar(30), -- Nom département 1
    d2 decimal, -- Code département 1
    n2 varchar(30) -- Nom département 1
);
drop function if exists get_similarite();
create or replace function get_similarite() returns setof similarite as
$get_similarite$
    declare
        ligne similarite_data;
        sortie similarite;
        i decimal; -- Pour les boucles
        v1 decimal array; -- Vecteur 1
        v2 decimal array; -- Vecteur 2
        v1v2 decimal; -- v1.v2
        _v1 decimal; -- Norme de v1 : ||v1||
        _v2 decimal; -- Norme de v2 : ||v2||
        s decimal; -- Pour requette de score
    begin
        -- Pour chaque couple de département
        for ligne in (
            select d1.code, d1.nom, d2.code, d2.nom
            from departement d1, departement d2
            where d1.code <> d2.code and d1.code < d2.code
        ) loop
            -- Récupération du nombre de voix de chaque candidat du
département 1
            v1 := '{}';
            for s in (
                select sum(voix)
                from score
                where code_departement = ligne.d1
                group by id_candidat
                order by id_candidat
            ) loop
                select array_append(v1, s) into v1; -- Remplissage du
vecteur 1
            end loop;
            -- Récupération du nombre de voix de chaque candidat du
département 2
```

```
            v2 := '{}';
            for s in (
                select sum(voix)
                from score
                where code_departement = ligne.d2
                group by id_candidat
                order by id_candidat
            ) loop
                select array_append(v2, s) into v2; -- Remplissage du
vecteur 2
            end loop;
            i := 0;
            v1v2 := 0;
            _v1 := 0;
            _v2 := 0;
            loop
                -- Calcul de v1.v2
                exit when i = (select array_length(v2, 1));
                i := i+1;
                v1v2 := v1v2 + v1[i]*v2[i];
                -- Calcul de v1² et v2² (pour préparer les normes)
                _v1 := _v1 + v1[i]*v1[i];
                _v2 := _v2 + v2[i]*v2[i];
            end loop;
            -- Calcul de ||v1|| et ||v2||
            _v1 := sqrt(_v1);
            _v2 := sqrt(_v2);
            -- Écriture des données dans la ligne de résultat
            sortie."Département 1" := concat(ligne.n1, ' (',
lpad(ligne.d1::text, 2, '0'), ')')::varchar(30);
            sortie."Département 2" := concat(ligne.n2, ' (',
lpad(ligne.d2::text, 2, '0'), ')')::varchar(30);
            sortie."Similarité" := v1v2 / (_v1 * _v2);
            return next sortie;
        end loop;
    end;
$get_similarite$ language plpgsql;
select * from get_similarite();
```

Question 8 - Meilleurs scores des candidats

```
-- Score max de chaque candidat le bdv, la commune où il a fait le meilleur
score
select distinct sr.nom, sr.prenom, sr."pourcentage", gs.bureau, gs.commune,
gs.departement
from (
    select max(gs."% voix/votants") as "pourcentage", c.nom, c.prenom
    from candidat c, get_score(c.nom, c.prenom) gs
    group by c.nom, c.prenom
) sr, candidat c, get_score(c.nom, c.prenom) gs
where sr."pourcentage" = gs."% voix/votants"
```

```
  and c.nom = sr.nom
  order by sr.pourcentage desc;
```

Question 9 - Script SQL

Voir le fichier script.sql, il contient tout le code contenu dans ce fichier à l'exception des exemples d'utilisation. Il faudra cependant y mettre à jour l'adresse du fichier de données CSV de notre archive.