

**XAVIER
Anushka**

**BUT Informatique : 1^{er} année
Groupe : Mercure**

**RAPPORT
SAE_104**

ALG choisit : Looping

2025/2026

SOMMAIRE

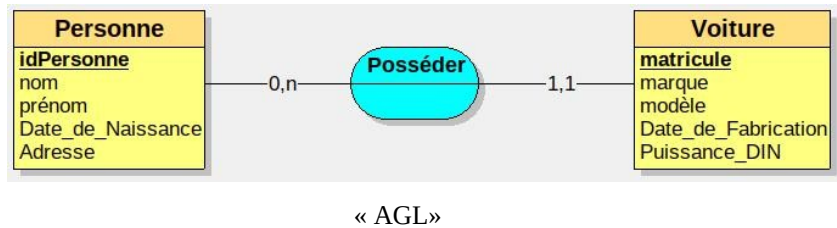
Presentation -----	p.1
2.1 Script manuel de création de la base de données -----	p.3
2.2 Modelisation et script de création « avec AGL » -----	p.4
2.3 Peuplement des tables -----	p.7
Images des résultats des tables -----	p.10

2.1 Script manuel de création de la base de données

```
CREATE TABLE regions (  
  code INTEGER PRIMARY KEY,  
  nom VARCHAR NOT NULL  
);  
  
CREATE TABLE departements (  
  code VARCHAR PRIMARY KEY,  
  region INTEGER REFERENCES regions (code) ON DELETE SET NULL ,  
  nom VARCHAR NOT NULL  
);  
  
CREATE TABLE communes(  
  code VARCHAR PRIMARY KEY ,  
  departement VARCHAR REFERENCES departements (code) ON DELETE SET NULL,  
  nom VARCHAR NOT NULL  
);  
  
CREATE TABLE sites (  
  idSite VARCHAR PRIMARY KEY ,  
  nom VARCHAR NOT NULL ,  
  codeCommune VARCHAR REFERENCES communes (code) ON DELETE SET NULL,  
  dateDeclaration DATE ,  
  typeEau VARCHAR NOT NULL ,  
  longitude FLOAT,  
  latitude FLOAT  
) ;  
  
CREATE TABLE evenements (  
  idEvenement bigint GENERATED ALWAYS AS IDENTITY ,  
  idSite VARCHAR REFERENCES sites (idSite) ON DELETE SET NULL ,  
  evenement VARCHAR,  
  debut DATE,  
  fin DATE,  
  mesure VARCHAR  
);  
  
CREATE TABLE analyses (  
  idAnalyse bigint GENERATED ALWAYS AS IDENTITY,  
  idSite VARCHAR REFERENCES sites (idSite) ON DELETE SET NULL ,  
  datePrelevement DATE ,  
  enterocoques INTEGER ,  
  escherichia INTEGER  
) ;
```

2.2 Modelisation et script de création « avec AGL »

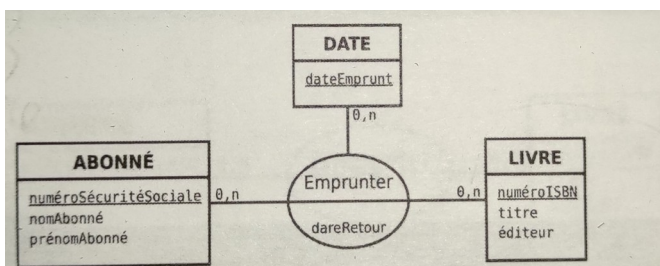
1- Illustrations comparatives cours/AGL commentée d'une association fonctionnelle :



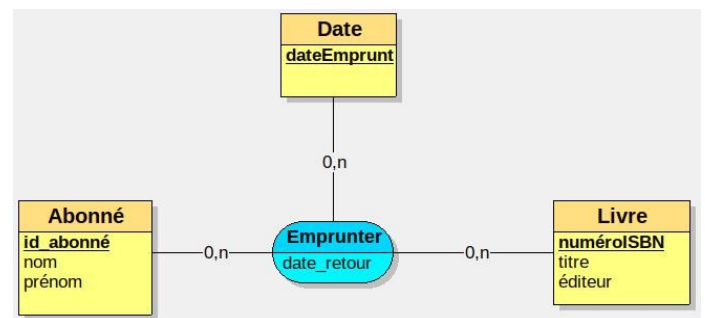
On peut voir dans notre exemple de l'association fonctionnelle qu'une personne peut posséder aucune ou plusieurs voitures mais une voiture est toujours posséder par une seule personne.

La cardinalité pour cette association sont 0,n et 1,1.

2- Illustrations comparatives cours/AGL commentée d'une association maillé :



« Cours »

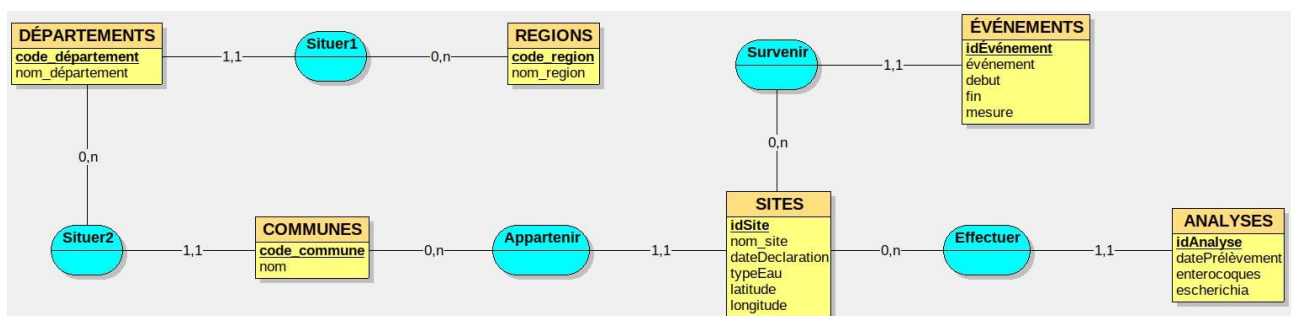


« AGL »

On peut voir dans notre exemple de l'association maillée qu'une personne peut emprunter aucune ou plusieurs livres et un livre peut être emprunter par un ou plusieurs abonnés selon la date.

Alors la cardinalité pour cette association est 0,n.

3- Modélisation physique de données réalisé avec l'AGL :



4- Script SQL de création des tables généré automatique avec l'AGL :

```
Sub Create_Tables()
```

```
DoCmd.RunSQL "CREATE TABLE REGIONS(" & _  
    "code_region INT," & _  
    "nom_region VARCHAR(100) NOT NULL," & _  
    "PRIMARY KEY(code_region)" & _  
    ");"
```

```
DoCmd.RunSQL "CREATE TABLE DÉPARTEMENTS(" & _  
    "code_département INT," & _  
    "nom_département VARCHAR(100) NOT NULL," & _  
    "code_region INT NOT NULL," & _  
    "PRIMARY KEY(code_département)," & _  
    "FOREIGN KEY(code_region) REFERENCES REGIONS(code_region)" & _  
    ");"
```

```
DoCmd.RunSQL "CREATE TABLE COMMUNES(" & _  
    "code_commune INT," & _  
    "nom VARCHAR(100) NOT NULL," & _  
    "code_département INT NOT NULL," & _  
    "PRIMARY KEY(code_commune)," & _  
    "FOREIGN KEY(code_département) REFERENCES DÉPARTEMENTS(code_département)"  
&  
    ");"
```

```
DoCmd.RunSQL "CREATE TABLE SITES(" & _  
    "idSite INT," & _  
    "nom_site VARCHAR(100) NOT NULL," & _  
    "dateDeclaration DATE," & _  
    "typeEau VARCHAR(100) NOT NULL," & _  
    "latitude DOUBLE," & _  
    "longitude DOUBLE," & _  
    "code_commune INT NOT NULL," & _  
    "PRIMARY KEY(idSite)," & _  
    "FOREIGN KEY(code_commune) REFERENCES COMMUNES(code_commune)" & _  
    ");"
```

```
DoCmd.RunSQL "CREATE TABLE ÉVÉNEMENTS(" & _  
    "idÉvénement INT," & _  
    "événement VARCHAR(100) NOT NULL," & _  
    "debut DATE," & _  
    "fin DATE," & _  
    "mesure VARCHAR(100) NOT NULL," & _  
    "idSite INT NOT NULL," & _  
    "PRIMARY KEY(idÉvénement)," & _  
    "FOREIGN KEY(idSite) REFERENCES SITES(idSite)" & _  
    ");"
```

```
DoCmd.RunSQL "CREATE TABLE ANALYSES(" & _  
    "idAnalyse INT," & _  
    "datePrélèvement DATE," & _  
    "enterocoques INT," & _
```

```
"escherichia INT," &  
"idSite INT NOT NULL," &  
"PRIMARY KEY(idAnalyse)," &  
"FOREIGN KEY(idSite) REFERENCES SITES(idSite)" &  
");"
```

End Sub

5- Discussion sur les différences entre les scripts produit manuellement et automatiquement :

L'avantage du script produit manuellement sont à nous de décider de la valeur que nous voulons attribuer ou ne pas attribuer aux variables, comparer au script produit automatiquement qui nous impose les valeurs ce qui limite l'entrée des données dans la base de données. Mais aussi le script produit manuellement est clair, bien structuré et surtout lisible par divers programmeur/ programmeuse contrairement au script produit automatique qui, selon l'ALG choisit, peut générer le script sous différentes formes et créer de la confusion.

Cependant, le taux d'erreur des résultats attendus est faible avec le script produit automatiquement comparé au script produit manuellement.

2.3 Peuplement des tables

Pendant le peuplement des tables j'ai utilisé 2 fonctions qui m'ont permis de faciliter la création des tables temporaires. Ces fonctions sont :

- **TEXT** permet de stocker des valeurs de type (chaîne de caractères, nombre à virgule, les combinaisons entre les chaînes de caractères et entiers) où la longueur est supérieure à certain

taille.

- **TEMP** permet de créer une table temporaire et l'effacer quand on se déconnecte du PostgreSQL.

-----Peupler les tables régions et départements

/* Comme on utilise le même fichier CSV pour le peuplement des tables régions et des départements, j'ai utilisé une table temporaire pour peupler les deux tables */

```
CREATE TEMP TABLE departement_regions_doc( --
  code_departement VARCHAR,
  nom_departement VARCHAR,
  code_region INTEGER,
  nom_region VARCHAR
);
```

/* La commande \copy m'a permis d'importer des données du fichier CSV pour les exploiter. */

```
\copy departement_regions_doc(code_departement, nom_departement,
code_region, nom_region)
FROM './departements-france.csv'
DELIMITER ','
CSV HEADER;
```

--Peuplement de la table régions

```
INSERT INTO regions(code, nom)
SELECT DISTINCT code_region, nom_region
FROM regions_doc;
```

---Peuplement de la table départements

```
INSERT INTO departements (code, region, nom)
SELECT code_Departement, code_region, nom_departement
FROM departement_doc;
```

-----Peuplement de la table Commune et site

/* Comme on utilise le même fichier CSV pour le peuplement des tables Communes et des sites, j'ai utilisé une table temporaire pour peupler les deux tables */

```

CREATE TEMP TABLE communes_sites_doc(
    saison_balnéaire INTEGER,
    région VARCHAR,
    département VARCHAR,
    code_unique_identification_du_site_de_baignade VARCHAR,
    précédent_code_unique_identification_du_site_de_baignade VARCHAR,
    evolution_2024_vs_2023 VARCHAR,
    nom_du_site_de_baignade VARCHAR,
    code_insee_de_la_commune VARCHAR,
    nom_de_la_commune VARCHAR,
    date_déclaration_UE DATE,
    type_eau VARCHAR,
    longitude TEXT,
    latitude TEXT
);

\copy communes_sites_doc(Saison_balnéaire, Région, Département,
Code_unique_identification_du_site_de_baignade,
Précédent_code_unique_identification_du_site_de_baignade,
Evolution_2024_vs_2023, Nom_du_site_de_baignade, Code_INSEE_de_la_commune,
Nom_de_la_commune, Date_déclaration_UE, Type_eau, Longitude, Latitude)
FROM './liste-des-sites-de-baignade-saison-balneaire-2024.csv'
DELIMITER '; '
CSV HEADER
ENCODING 'LATIN1';

ALTER TABLE communes_sites_doc
ALTER COLUMN longitude TYPE FLOAT USING REPLACE (longitude, ',',
'.')::FLOAT,
ALTER COLUMN latitude TYPE FLOAT USING REPLACE(latitude, ',', '.')::FLOAT;

---Peuplement du table Commune

INSERT INTO communes (code, département, nom)
SELECT DISTINCT code_insee_de_la_commune, département, nom_de_la_commune
FROM communes_sites_doc;

---Peuplement du table sites

INSERT INTO sites (idSite, nom, dateDeclaration, typeEau, longitude,
latitude)
SELECT code_unique_identification_du_site_de_baignade,
nom_du_site_de_baignade, date_déclaration_UE, type_eau, longitude, latitude
FROM communes_sites_doc;

```

-----Peuplement du table evenements

```
CREATE TEMP TABLE evenements_doc(  
    saison_balnéaire INTEGER,  
    région VARCHAR,  
    departement VARCHAR,  
    code_unique_identification_du_site_de_baignade VARCHAR,  
    type_événement VARCHAR,  
    debut DATE,  
    fin DATE,  
    mesure VARCHAR  
);  
  
\copy evenements_doc(saison_balnéaire, région, departement,  
Code_unique_identification_du_site_de_baignade, type_événement, debut, fin,  
mesure)  
FROM './saison-balneaire-2024-informations-sur-la-saison.csv'  
DELIMITER ';'   
CSV HEADER  
ENCODING 'LATIN1';  
  
INSERT INTO evenements (idSite, evenement, debut, fin, mesure)  
SELECT code_unique_identification_du_site_de_baignade, type_événement,  
debut, fin, mesure  
FROM evenements_doc;
```

---Peuplement du table analyses

```
CREATE TEMP TABLE analyses_doc (  
    saison_balnéaire INTEGER,  
    région VARCHAR,  
    département VARCHAR,  
    code_unique_identification_du_site_de_baignade VARCHAR,  
    datePrélèvement DATE,  
    enterocoques INTEGER,  
    escherichia INTEGER,  
    statut_du_prélèvement VARCHAR  
    c9 VARCHAR,  
    c10 VARCHAR,  
    c11 VARCHAR;  
);  
  
\copy analyses_doc(saison_balnéaire, région, département,  
code_unique_identification_du_site_de_baignade, datePrélèvement,  
enterocoques, escherichia, statut_du_prélèvement, c9, c10, c11)  
FROM './saison-balneaire-2024-resultats-danalyses.csv'  
DELIMITER ';'   
CSV HEADER  
ENCODING 'LATIN1';  
  
INSERT INTO analyses (idSite, datePrelevement, enterocoques, escherichia)
```

```
SELECT code_unique_identification_du_site_de_baignade, datePrélèvement ,
enterocoques, escherichia
FROM analyses_doc;
```

```
xavier_anushka=> \d
```

List of relations			
Schema	Name	Type	Owner
pg_temp_4	analyses_doc	table	xavier_anushka
pg_temp_4	communes_sites_doc	table	xavier_anushka
pg_temp_4	departement_doc	table	xavier_anushka
pg_temp_4	evenements_doc	table	xavier_anushka
pg_temp_4	regions_doc	table	xavier_anushka
public	analyses	table	xavier_anushka
public	analyses_idanalyse_seq	sequence	xavier_anushka
public	communes	table	xavier_anushka
public	departements	table	xavier_anushka
public	evenements	table	xavier_anushka
public	evenements_idevenement_seq	sequence	xavier_anushka
public	regions	table	xavier_anushka
public	sites	table	xavier_anushka

(13 rows)

idanalyse	idsite	dateprelevement	enterocoques	escherichia
1	FRK2101010001530	2024-06-18	15	15
2	FRK2101010001530	2024-07-02	15	15
3	FRK2101010001530	2024-07-17	15	15
4	FRK2101010001530	2024-08-06	15	15
5	FRK2101010001530	2024-08-21	15	15
6	FRK21010530001570	2024-04-29	77	110
7	FRK21010530001570	2024-05-20	30	15
8	FRK21010530001570	2024-06-10	15	15
9	FRK21010530001570	2024-07-01	15	15
10	FRK21010530001570	2024-07-26	15	15
11	FRK21010530001570	2024-08-19	15	15
12	FRK21010580001580	2024-06-18	61	15
13	FRK21010580001580	2024-07-02	15	15
14	FRK21010580001580	2024-07-17	15	15
15	FRK21010580001580	2024-08-06	15	15
16	FRK21010580001580	2024-08-21	15	15
17	FRK21010870001600	2024-06-17	15	15
18	FRK21010870001600	2024-07-02	15	15
19	FRK21010870001600	2024-07-18	15	15
20	FRK21010870001600	2024-08-06	61	353
21	FRK21010870001600	2024-08-20	15	15
22	FRK21010990001150	2024-06-13	15	15
23	FRK21010990001150	2024-07-01	15	30
24	FRK21010990001150	2024-07-17	15	94
25	FRK21010990001150	2024-08-05	15	46
26	FRK21010990001150	2024-08-19	15	15
27	FRK21010990001605	2024-06-13	15	15
28	FRK21010990001605	2024-07-01	142	15
29	FRK21010990001605	2024-07-17	30	15
30	FRK21010990001605	2024-08-05	15	15
31	FRK21010990001605	2024-08-19	307	15
32	FRK21011230001607	2024-04-22	15	15
33	FRK21011230001607	2024-05-16	15	15
34	FRK21011230001607	2024-06-05	15	15
35	FRK21011230001607	2024-06-20	15	15
36	FRK21011230001607	2024-07-01	15	15
37	FRK21011230001607	2024-07-18	15	15
38	FRK21011230001607	2024-08-05	15	15
39	FRK21011230001607	2024-08-20	15	15
40	FRK21011330001311	2024-05-23	15	15
41	FRK21011330001311	2024-06-18	46	127
42	FRK21011330001311	2024-07-02	15	15
43	FRK21011330001311	2024-07-17	61	438

```
xavier_anushka=> SELECT * FROM regions;
```

code	nom
84	Auvergne-Rhône-Alpes
2	Martinique
93	Provence-Alpes-Côte d'Azur
6	Mayotte
44	Grand Est
53	Bretagne
11	Île-de-France
1	Guadeloupe
75	Nouvelle-Aquitaine
76	Occitanie
28	Normandie
4	La Réunion
3	Guyane
52	Pays de la Loire
32	Hauts-de-France
94	Corse
24	Centre-Val de Loire
27	Bourgogne-Franche-Comté

(18 rows)