Projet 3 - Étape 6

ASSURANCE HABITATION

Requêtez une base de données avec SQL

BUILDING BUSINESS REVIEW

Analyse exploratoire des fichiers CSV

Dictionnaire de données

Schéma relationnel

Script SQL

Création BDD

Structure des tables

Import des DATAS

Execution des requetes

ANALYSE EXPLORATOIRE & DICTIONNAIRE DE DONNÉES

L'analyse exploratoire a permis d'identifier le type de données présentent dans les fichiers CSV. Mais également de mettre en lumière certaines données pouvant prêter à confusion ou induire en erreur.

Exemples de données trouvées durant l'analyse :

No_voie	B_T_Q
6	
91	
5186	
5208	
20	
5245	
7002	
7042	

Type_de_voie	Voie
CD	135
PL	01/06/1907 00:00
IMP	10 RUE DE METZ
RUE	11/11/1918 00:00

Code_dep_code_commune
2A004
2A004
2A004

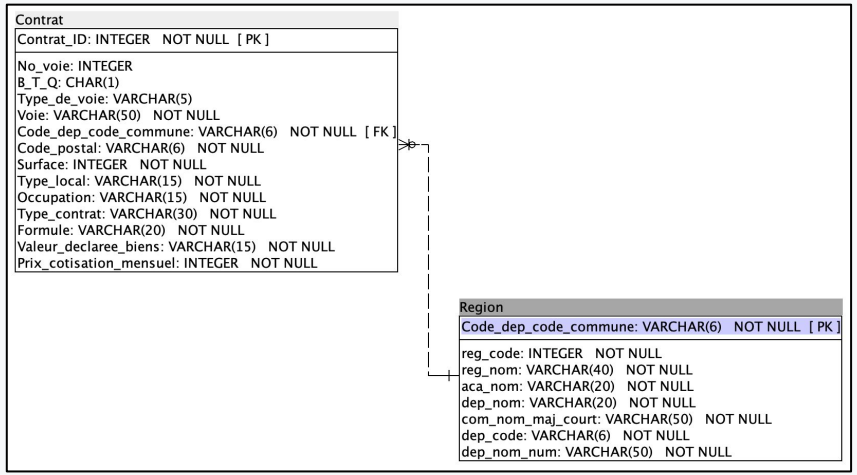
À l'issue de l'analyse, j'ai pu compléter le dictionnaire de données (ci-dessous), qui me permettra par la suite de réaliser un schéma relationnel.

	Nom des colonnes	Type de données	Taille	Clé	Description
CONTRAT.CSV	Contrat_ID	INT		Clé primaire	Id unique pour les contrats
	No_voie	INT			Numéro dans la voie pour l'adresse du logement assuré
	B_T_Q	CHAR	1		Indicateur éventuel de répétition pour l'adresse du logement assuré sur un caractère
	Type_de_voie	VARCHAR	5		Type de voie pour l'adresse du logement assuré: rue, av (Avenue), rte (Route), …
	Voie	VARCHAR	50		Libellé de la voie pour l'adresse du logement assuré
	Code_dep_code_commune	VARCHAR	6	Clé secondaire	Concaténation du code département et code commune pour avoir une clé unique
	Code_postal	VARCHAR	6		Code postal pour l'adresse du logement assuré
	Surface	INT			Superficie du bien immobilier
	Type_local	VARCHAR	15		Précise le type de bien (maison ou appartement)
	Occupation	VARCHAR	15		Précise le type d'occupant du bien (locataire ou propriétaire)
	Type_contrat	VARCHAR	30		Exprime le type de contrat (Résidence principale ou secondaire, ou mise en location)
	Formule	VARCHAR	20		Est le type de formule choisi pour assurer le bien
	Valeur_declaree_biens	VARCHAR	15		Fourchette de la valeur des biens déclarés à assurer
	Prix_cotisation_mensuel	INT			Montant du prix de la cotisation mensuelle
REGION.CSV	Code_dep_code_commune	VARCHAR	6	Clé primaire	Concaténation du code département et code commune pour avoir une clé unique
	reg_code	INT			code de la région
	reg_nom	VARCHAR	40		nom de la région
	aca_nom	VARCHAR	20		Nom de l'académie lié à la région
	dep_nom	VARCHAR	20		nom du département concerné
	com_nom_maj_court	VARCHAR	50		Nom de la commune
	dep_code	VARCHAR	5		code du departement
	dep_nom_num	VARCHAR	50		concaténation du nom puis du numéro de département

SCHÉMA RELATIONNEL & SCRIPT SQL

MODÈLE LOGIQUE DE DONNÉES - MLD

Capture d'écran du schéma relationnel réalisé avec Power Architect. Basé sur la rédaction du Dictionnaire de données.



MODÈLE PHYSIQUE DE DONNÉES - MPD

Traduction du MLD pour obtenir un script SQL à utiliser lors de la création de la BDD avant d'importer les données.

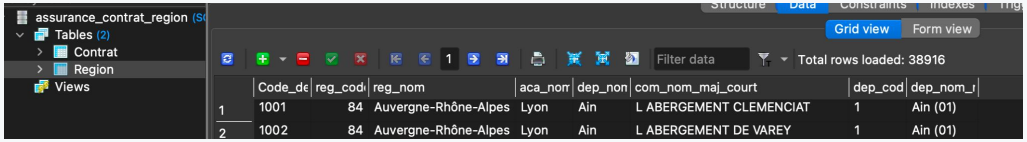
```sql
CREATE TABLE Region (
Code_dep_code_commune VARCHAR(6) NOT NULL,
reg_code INTEGER NOT NULL,
reg_nom VARCHAR(40) NOT NULL,
aca_nom VARCHAR(20) NOT NULL,
dep_nom VARCHAR(20) NOT NULL,
com_nom_maj_court VARCHAR(50) NOT NULL,
dep_code VARCHAR(6) NOT NULL,
dep_nom_num VARCHAR(50) NOT NULL,
PRIMARY KEY (Code_dep_code_commune)
)
```

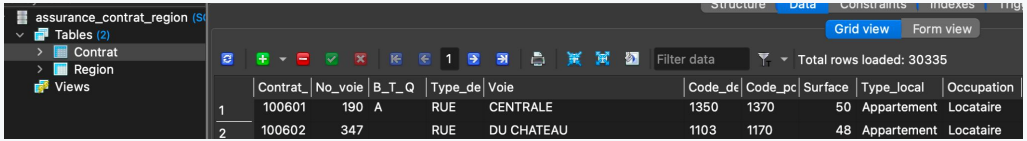
```sql
CREATE TABLE Contrat (
Contrat_ID INTEGER PRIMARY KEY,
No_voie INTEGER,
B_T_Q VARCHAR(1),
Type_de_voie VARCHAR(5),
Voie VARCHAR(50) NOT NULL,
Code_dep_code_commune VARCHAR(6) NOT NULL,
Code_postal VARCHAR(6) NOT NULL,
Surface INTEGER NOT NULL,
Type_local VARCHAR(15) NOT NULL,
Occupation VARCHAR(15) NOT NULL,
Type_contrat VARCHAR(30) NOT NULL,
Formule VARCHAR(20) NOT NULL,
Valeur_declaree_biens VARCHAR(15) NOT NULL,
Prix_cotisation_mensuel INTEGER NOT NULL,
FOREIGN KEY (code_dep_code_commune) REFERENCES region (code_dep_code_commune)
ON DELETE NO ACTION
ON UPDATE NO ACTION
);
```

STRUCTURE DES TABLES & CHARGEMENT DE LA BDD

assurance Table name: Contrat WITHOUT ROWID STRICT

	Name	Data type	Primary Key	Foreign Key	Unique	Check	Not NULL	Collate	Generated	
1	Contrat_ID	INTEGER	🔑							NULL
2	No_voie	INTEGER								NULL
3	B_T_Q	VARCHAR (1)								NULL
4	Type_de_voie	VARCHAR (5)								NULL
5	Voie	VARCHAR (50)								NULL
6	Code_dep_code_commune	VARCHAR (6)								NULL
7	Code_postal	VARCHAR (6)								NULL
8	Surface	INTEGER								NULL
9	Type_local	VARCHAR (15)								NULL
10	Occupation	VARCHAR (15)								NULL
11	Type_contrat	VARCHAR (30)								NULL
12	Formule	VARCHAR (20)								NULL
13	Valeur_declaree_biens	VARCHAR (15)								NULL
14	Prix_cotisation_mensuel	INTEGER								NULL





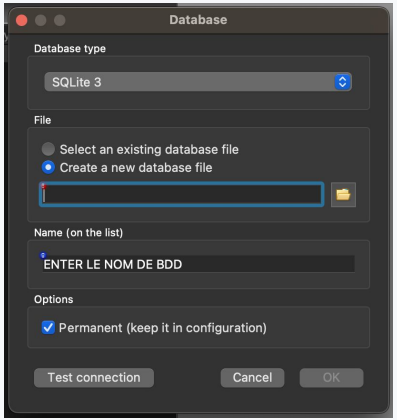
IMPORT DU CODE DANS SQLITE

Résultats après import du script SQL issu de Power Architect dans SQLite. La structure des tables de la BDD sont créés.

IMPORT DES DONNÉES DANS SQLITE

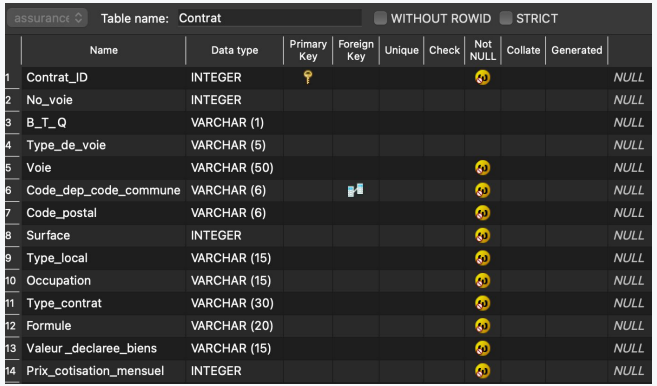
Une fois la base de données paramétrée et structurée, j'ai pu importer les données provenant des tables Region et Contrat.

BASE DE DONNÉES : CRÉATION & CHARGEMENT



création de la base de données

Dans SQLite, créer une nouvelle base de données, la paramétrer puis la connecter. Dans l'éditeur SQL, ajouter le script SQL issu du schéma relationnel

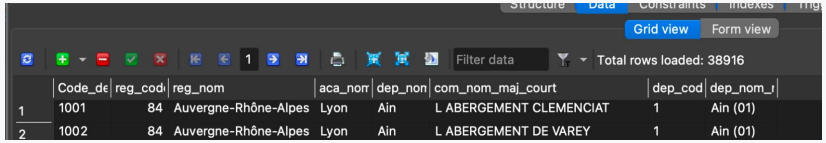


	Name	Data type	Primary Key	Foreign Key	Unique	Check	Not NULL	Collate	Generated	
1	Contrat_ID	INTEGER								NULL
2	No_voie	INTEGER								NULL
3	B_T_Q	VARCHAR (1)								NULL
4	Type_de_voie	VARCHAR (5)								NULL
5	Voie	VARCHAR (50)								NULL
6	Code_dep_code_commune	VARCHAR (6)								NULL
7	Code_postal	VARCHAR (6)								NULL
8	Surface	INTEGER								NULL
9	Type_local	VARCHAR (15)								NULL
10	Occupation	VARCHAR (15)								NULL
11	Type_contrat	VARCHAR (30)								NULL
12	Formule	VARCHAR (20)								NULL
13	Valeur _declaree_biens	VARCHAR (15)								NULL
14	Prix_cotisation_mensuel	INTEGER								NULL

Structure des tables

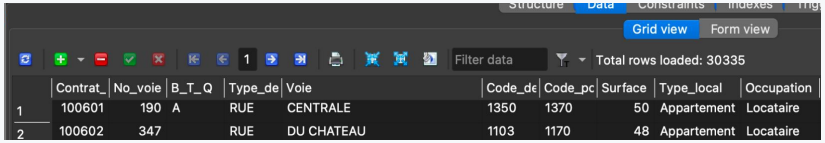
Une fois le script SQL exécuté, on obtient la structure des tables. Ensuite, on peut importer les données des fichiers CSV dans les tables. On obtient une BDD chargée (voir exemples ci-dessous)

Exemple 1 : Table Region chargée



	Code_de	reg_cod	reg_nom	aca_nom	dep_non	com_nom_maj_court	dep_cod	dep_nom_l
1	1001	84	Auvergne-Rhône-Alpes	Lyon	Ain	L ABERGEMENT CLEMENCIAT	1	Ain (01)
2	1002	84	Auvergne-Rhône-Alpes	Lyon	Ain	L ABERGEMENT DE VAREY	1	Ain (01)

Exemple 2 : Table Contrat chargée



	Contrat_	No_voie	B_T_Q	Type_de	Voie	Code_de	Code_pc	Surface	Type_local	Occupation
1	100601	190	A	RUE	CENTRALE	1350	1370	50	Appartement	Locataire
2	100602	347		RUE	DU CHATEAU	1103	1170	48	Appartement	Locataire

REQUÊTES SQL - REQUÊTES 1 & 2

Requête 1 : Lister les numéros de contrats (contrat_ID) avec leur surface pour la commune de Caen.

SELECT c.Contrat_ID, c.Surface > Sélectionne l'ID du contrat et la surface.
FROM Contrat AS c > Utilise la table Contrat (alias c).
LEFT JOIN Region AS r > Joint la table Region (alias r).
ON c.Code_dep_code_commune = r.Code_dep_code_commune > Fait correspondre chaque contrat à sa commune via le code géographique.
WHERE r.com_nom_maj_court = 'CAEN' > Filtre pour ne garder que les contrats de la commune « CAEN ».

```sql
SELECT c.Contrat_ID, c.Surface
FROM Contrat AS c
LEFT JOIN Region AS r
    ON c.Code_dep_code_commune = r.Code_dep_code_commune
WHERE r.com_nom_maj_court = 'CAEN';
```

Total rows loaded: 4

	Contrat_ID	Surface
1	103791	35
2	103792	99
3	103793	40
4	103794	20

Requête 2 : Lister les numéros de contrats avec le type de contrat et leur formule pour les maisons du département 71.

SELECT c.Contrat_ID, c.type_contrat, c.formule > Sélectionne l'identifiant, le type de contrat et la formule.
FROM Contrat AS c > Utilise la table Contrat (alias c).
LEFT JOIN Region AS r ON ... > Joint la table Region (alias r) via le code géographique.
WHERE lower(c.type_local) LIKE 'maison%' > Filtre pour ne garder que les enregistrements dont le type_local commence par « maison ». % représente un joker : il signifie « n'importe quelle suite de caractères ».
AND r.dep_code = '71' > Limite les résultats au département 71.

```sql
SELECT c.contrat_ID, c.type_contrat, c.formule
FROM Contrat AS c
LEFT JOIN Region AS r
ON c.Code_dep_code_commune = r.Code_dep_code_commune
WHERE lower(c.type_local) LIKE 'maison%'
AND r.dep_code = '71';
```

Total rows loaded: 4

	Contrat_ID	Type_contrat	Formule
1	114768	Residence principale	Integral
2	114779	Residence principale	Classique
3	114782	Residence principale	Classique
4	114812	Residence principale	Integral

REQUÊTES SQL - REQUÊTES 3 & 4

Requête 3 : Lister le nom des régions de France.

SELECT DISTINCT reg_nom AS Region > Sélectionne les noms des régions de France en éliminant les doublons (DISTINCT).
FROM Region > La requête interroge la table Region, qui contient les noms des différentes régions françaises.
ORDER BY reg_nom > Trie les résultats par ordre alphabétique croissant (A → Z).

```sql
SELECT DISTINCT reg_nom AS Region
FROM Region
ORDER BY reg_nom
```

	Region
1	Auvergne-Rhône-Alpes
2	Bourgogne-Franche-Comté
3	Bretagne
4	Centre-Val de Loire
5	Collectivités d'outre-mer
6	Corse
7	Grand Est
8	Guadeloupe
9	Guyane
10	Hauts-de-France
11	Ile-de-France
12	La Réunion
13	Martinique
14	Mayotte
15	Normandie
16	Nouvelle-Aquitaine
17	Occitanie
18	Pays de la Loire
19	Provence-Alpes-Côte d'Azur

Requête 4 : Quels sont les 5 contrats qui ont les surfaces les plus élevées ?

SELECT Contrat_ID AS Contrat, Surface > Sélectionne l'ID du contrat (renommé en "Contrat") et la surface.
FROM Contrat > Utilise la table Contrat.
ORDER BY Surface DESC > Trie les résultats par surface décroissante, de la plus grande à la plus petite.
LIMIT 5 > Limite l'affichage aux 5 premières lignes.

```sql
SELECT Contrat_ID AS Contrat, Surface
FROM Contrat
ORDER BY Surface DESC
LIMIT 5;
```

	Contrat	Surface
1	104211	815
2	105463	742
3	130878	595
4	100822	570
5	109872	559

REQUÊTES SQL - REQUÊTES 7 & 9

Requête 7 : Quel est le nombre de formules “integral” sur la région Pays de la Loire ?

SELECT COUNT(c.Contrat_ID)) AS Nombre_formules_integral > Compte le nombre total de formules ‘integral’ (via l’ID du contrat).
FROM Contrat c > Utilise la table Contrat (alias c).
LEFT JOIN Region r ON... > Effectue une jointure à gauche avec la table Region (alias r).
WHERE lower(c.Formule) = 'integral' > Filtre pour les contrats dont la formule est "integral" (insensible à la casse).
AND lower(r.reg_nom) = 'pays de la loire' > Limite les résultats à la région Pays de la Loire (insensible à la casse).

```
1 SELECT COUNT(c.Contrat_ID) AS Nombre_formules_integral
2 FROM Contrat c
3 LEFT JOIN Region r
4  ON c.Code_dep_code_commune = r.Code_dep_code_commune
5 WHERE lower(c.Formule) = 'integral'
6  AND lower(r.reg_nom) = 'pays de la loire';
```

Total rows loaded: 1

	Nombre_formules_integral
1	589

Requête 9 : Quelle est la surface moyenne des contrats à Paris ?

SELECT ROUND(AVG(c.Surface), 2) AS Surface_moyenne > Calcule la surface moyenne arrondie à deux décimales.
FROM Contrat AS c > Utilise la table Contrat (alias c).
LEFT JOIN Region AS r ON c... > Effectue une jointure à gauche avec la table Region (alias r) pour obtenir les informations de localisation.
WHERE r.com_nom_maj_court LIKE 'PARIS%' > Filtre pour les enregistrements correspondant à Paris.
AND r.dep_code = '75'; > Limite les résultats aux contrats du département 75.

```
1 SELECT ROUND(AVG(c.Surface), 2) AS Surface_moyenne
2 FROM Contrat AS c
3 LEFT JOIN Region AS r
4 ON c.Code_dep_code_commune = r.Code_dep_code_commune
5 WHERE r.com_nom_maj_court LIKE 'PARIS%'
6 AND r.dep_code = '75' ;
```

Total rows loaded: 1

	Surface_moyenne
1	51.77

REQUÊTES SQL - REQUÊTES 11

Requête 11 : Liste des communes ayant eu au moins 150 contrats.

SELECT r.com_nom_maj_court AS Commune, COUNT(c.Contrat_ID) AS Nombre_de_contrats > Liste les communes et compte le nombre de contrats (via l'ID du contrat).

FROM Contrat AS c > Utilise la table Contrat (alias c).

LEFT JOIN Region AS r ON c... > Effectue une jointure à gauche avec la table Region (alias r)

GROUP BY r.com_nom_maj_court > Regroupe les résultats par commune.

HAVING COUNT(c.Contrat_ID) >= 150 > Ne conserve que les communes ayant au moins 150 contrats.

ORDER BY Nombre_de_contrats DESC; > Trie les résultats du plus grand au plus petit nombre de contrats.

```
2         COUNT(c.Contrat_ID) AS Nombre_de_contrats
3 FROM Contrat AS c
4 LEFT JOIN Region AS r
5  ON c.Code_dep_code_commune = r.Code_dep_code_commune
6 GROUP BY r.com_nom_maj_court
7 HAVING COUNT(c.Contrat_ID) >= 150
8 ORDER BY Nombre_de_contrats DESC;
```

Total rows loaded: 20

	Commune	Nombre_de_contrats
1	PARIS 18	515
2	PARIS 17	468
3	PARIS 15	407
4	PARIS 16	394
5	NICE	387
6	PARIS 11	381
7	PARIS 20	302
8	BORDEAUX	302
9	NANTES	291
10	PARIS 19	266
11	PARIS 10	263
12	PARIS 12	252
13	PARIS 14	222
14	GRENOBLE	220
15	PARIS 9	204
16	TOULOUSE	187

REQUÊTES SQL - REQUÊTE 12

Requête 12 : Quel est le nombre de contrats pour chaque région ?

SELECT r.reg_nom AS Region, COUNT(c.Contrat_ID) AS Nombre_de_contrats > Liste les régions et compte le nombre de contrats (via l'ID du contrat).
FROM Contrat AS c > Utilise la table Contrat (alias c).
LEFT JOIN Region AS r ON c… > Effectue une jointure à gauche avec la table Region (alias r) pour lier les informations régionales.
GROUP BY r.reg_nom > Regroupe les résultats par région.
ORDER BY Nombre_de_contrats DESC > Trie les résultats du plus grand au plus petit nombre de contrats.

```
1 SELECT r.reg_nom AS Region, COUNT(c.Contrat_ID) AS Nombre_de_contrats
2 FROM Contrat AS c
3 LEFT JOIN Region AS r
4 ON c.Code_dep_code_commune = r.Code_dep_code_commune
5 GROUP BY r.reg_nom
6 ORDER BY Nombre_de_contrats DESC;
```

Total rows loaded: 17

	Region	Nombre_de_contrats
2	Provence-Alpes-Côte d'Azur	3279
3	Auvergne-Rhône-Alpes	3042
4	Nouvelle-Aquitaine	2038
5	Occitanie	1609
6	Pays de la Loire	1196
7	Hauts-de-France	1189
8	Bretagne	947
9	Normandie	824
10	Grand Est	769
11	Centre-Val de Loire	598
12	Bourgogne-Franche-Comté	293
13	Corse	247
14	Martinique	73
15	Guyane	37
16	*NULL*	9

Requête 12 : Trouver les Contrat_ID NULL :

SELECT c.Contrat_ID, c.Code_dep_code_commune, r.reg_nom > Sélectionne trois colonnes : l'identifiant du contrat (Contrat_ID), le code département/commune (Code_dep_code_commune) et le nom de la région (reg_nom).
FROM Contrat AS c > Utilise la table Contrat, que l'on renomme par l'alias c.
LEFT JOIN Region AS r > Effectue une jointure à gauche avec la table Region (alias r).
ON c.Code_dep_code_commune = r.C… > Spécifie la condition de jointure : les lignes sont reliées lorsque le champ Code_dep_code_commune de Contrat correspond au champ Code_dep_code_commune de Region.
WHERE r.reg_nom IS NULL > Filtre les résultats pour ne conserver que les contrats dont le champ reg_nom n'est pas renseigné (la jointure n'a pas trouvé de correspondance), permettant ainsi d'identifier les contrats sans région associée.

```
1 SELECT c.Contrat_ID, c.code_dep_code_commune, r.reg_nom
2 FROM Contrat AS C
3 LEFT JOIN Region AS r
4 ON c.Code_dep_code_commune = r.Code_dep_code_commune
5 WHERE reg_nom IS NULL
```

Total rows loaded: 9

	Contrat_ID	Code_dep_code_commune	reg_nom
1	128054	97460	*NULL*
2	128056	97434	*NULL*
3	128059	97470	*NULL*
4	128061	97460	*NULL*
5	128064	97434	*NULL*
6	128068	97434	*NULL*
7	128070	97434	*NULL*
8	128077	97460	*NULL*
9	128082	97460	*NULL*

CONCLUSION & RETOURS

COMPRÉHENSION DE LA BDD

Apprentissage du fonctionnement global d'une base de données.
Importance d'analyser les types de données (choix entre CHAR et VARCHAR, traitement des valeurs alphanumériques) pour garantir l'intégrité future.
Cette compréhension m'a permis de saisir l'impact d'une bonne structuration sur la qualité des analyses.
Elle m'a également sensibilisé aux conséquences d'une mauvaise définition des types sur les résultats des requêtes.

RÉDACTION DES REQUÊTES SQL

Maîtrise de l'ordre d'exécution des requêtes et des fonctions d'agrégation (AVG, COUNT, ROUND, etc.).
Utilisation des jointures pour lier les tables et extraire des informations cohérentes.
Ces techniques m'ont aidé à optimiser mes requêtes et à garantir des résultats précis et pertinents.
J'ai découvert comment combiner efficacement plusieurs sources de données pour obtenir une vue d'ensemble complète.

PRÉPARATION ET MODÉLISATION

Analyse des fichiers CSV et complétion du dictionnaire pour repérer les incohérences.
Passage du MLD (Modèle Logique de Données) au MPD (Modèle Physique de Données) via SQL Power Architect pour structurer correctement la BDD.
Cette étape a été essentielle pour transformer des données brutes en une structure cohérente et exploitable.
Elle a aussi montré l'importance de planifier minutieusement avant de passer à l'implémentation technique.

IMPACT GLOBAL

Cette démarche m'a permis de mieux composer mes requêtes et d'assurer la fiabilité de l'analyse future, tout en renforçant ma compréhension globale du processus Data Analyst.
Elle m'a offert une vision intégrée du cycle complet de gestion des données, de la préparation à l'extraction d'informations.
Ce projet constitue une base solide pour aborder des analyses plus complexes et optimiser mes méthodes de travail pour de futurs projets.