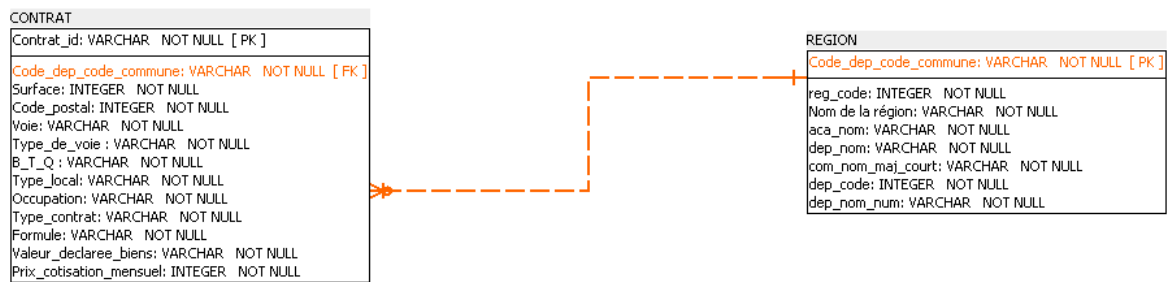


Dictionnaire des données

	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	VARCHAR	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	100		Libellé de la voie pour l'adresse du logement assuré
	Code_dep_code_commune	VARCHAR	10	Clé étrangère	Concaténation du code département et code commune pour avoir une clé unique
	Code_postal	INT	5		Code postal pour l'adresse du logement assuré
	Surface	INT	4		Surface du logement
	Type_local	VARCHAR	15		Type de logement : appartement ou maison
	Occupation	VARCHAR	15		Type d'occupation : locataire ou propriétaire
	Type de contrat	VARCHAR	20		Type de contrat : mise en location, résidence principale ou secondaire
	Formule	VARCHAR	10		Type de formule : classique ou intégral
	Valeur_declaree_biens	VARCHAR	12		Fourchette de valeur du bien
REGION.CSV	Prix_cotisation_mensuel	INT	4		Prix des cotisations mensuel du bien correspondant
	Code_dep_code_commune	VARCHAR	10	Clé primaire	Concaténation du code département et code commune pour avoir une clé unique
	reg_code	INT	2		Numéro de la région
	reg_nom	VARCHAR	25		Nom de la région
	aca_nom	VARCHAR	25		Nom de la ville
	dep_nom	VARCHAR	40		Nom du département
	com_nom_maj_court	VARCHAR	40		Nom de la commune
	dep_code	VARCHAR	3		Numéro du département
	dep_nom_num	VARCHAR	40		Nom du département

Schéma relationnel des 2 tables



Création des 2 tables dans SQL

```
-- Créer database
CREATE DATABASE region_contrat;

-- Afficher les databases créées pour s'assurer qu'elles ont bien été créées
SHOW DATABASES;

-- Dire à MySql qu'on va utiliser la database region
USE region_contrat;

-- Création de la table table_region dans la database region_contrat
CREATE TABLE table_region (
    Code_dep_code_commune VARCHAR (10) NOT NULL,
    Code_region INTEGER NOT NULL,
    Nom_de_la_region VARCHAR (100),
    Nom_academie VARCHAR (100),
    Nom_departement VARCHAR (100),
    Nom_commune VARCHAR (100),
    Code_departement INTEGER NOT NULL,
    Nom_et_numero_departement VARCHAR (100),
    CONSTRAINT Code_dep_code_commune PRIMARY KEY
(Code_dep_code_commune)
);

-- Afficher les différentes colonnes précédemment créées
SHOW COLUMNS FROM table_region;

-- Créer table_contrat dans la database contrat_region
CREATE TABLE table_contrat (
    Contrat_ID INTEGER NOT NULL,
    No_voie INTEGER NOT NULL,
    B_T_Q VARCHAR (1),
    Type_de_voie VARCHAR (5),
    Voie VARCHAR (100),
    Code_dep_code_commune VARCHAR (10) NOT NULL,
    Code_postal INTEGER NOT NULL,
    Surface INTEGER NOT NULL,
    Type_local VARCHAR (15),
    Occupation VARCHAR (15),
    Type_contrat VARCHAR (20),
    Formule VARCHAR (10),
    Valeur_declaree_biens VARCHAR (12),
    Prix_cotisation_mensuel INTEGER NOT NULL,
    CONSTRAINT contrat_id PRIMARY KEY (Contrat_id)
);
```

```

import mysql.connector
import csv
# Connect to MySQL
connection = mysql.connector.connect(
    host=
    user=
    passwd=
    database=
)
cursor = connection.cursor()
##### TABLE CONTRAT #####
# Définir le nom de table
table_name = "table_contrat"
# Fonction pour remplacer valeur vide par Null
def replace_empty_with_null(row):
    return [None if field == '' else field for field in row]
# Ouvrir csv
with open("C:\\Users\\simon\\OneDrive\\Documents\\Openclassrooms - Business Intelligence Analyst\\Projet 3 - sql\\Donnée\\Contrat+(4).csv", mode='r') as file:
    csv_data = csv.reader(file, delimiter=';')
    # Eviter les en-têtes si il y en a
    next(csv_data)
    # Requête SQL pour importer les données (nom de colonne)
    query = f"INSERT IGNORE INTO {table_name} (Contrat_ID, No_voie, B_T_Q, Type_de_voie, Voie, Code_dep_code_commune, Code_postal, Surface, Type_local, Occupation, Type_contrat, Formule, Valeur_declaree_biens, Prix_cotisation_mensuel) VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)"
    #INSERT IGNORE = MySql ignorera les erreurs de doublons pour les clés primaires
    # Parcourir chaque ligne pour les insérer
    for row in csv_data:
        # Remplacer chaque valeur vide par des Null pour chaque ligne
        row = replace_empty_with_null(row)

        # Si chaque ligne contient 14 colonnes ALORS importer la ligne dans la BDD
        if len(row) == 14:
            cursor.execute(query, row)
        else:
            print(f"Ligne non importer car nombre de colonne différent : {row}")

    # Importer les données dans la table
    connection.commit()
print("Les données ont été importées.")
##### TABLE REGION #####
# Définir le nom de table
table_name2 = "table_region"
# Fonction pour remplacer valeur vide par Null
def replace_empty_with_null(row2):
    return [None if field2 == '' else field2 for field2 in row2]
# Ouvrir csv

```

```

with open("C:\\Users\\simon\\OneDrive\\Documents\\Openclassrooms - Business
Intelligence Analyst\\Projet 3 - sql\\Donnée\\Region+(7).csv", mode='r') as
file2:
    csv_data2 = csv.reader(file2, delimiter=';')
    # Eviter les en-têtes si il y en a
    next(csv_data2)
    # Requête SQL pour importer les données (nom de colonne)
    query2 = f"INSERT IGNORE INTO {table_name2} (Code_dep_code_commune,
Code_region, Nom_de_la_region, Nom_academie, Nom_departement, Nom_commune,
Code_departement, Nom_et_numero_departement) VALUES (%s, %s, %s, %s, %s, %s,
%s, %s)"

    #INSERT IGNORE = MySql ignorera les erreurs de doublons
pour les clés primaires
    # Parcourir chaque ligne pour les insérer
    for row2 in csv_data2:
        # Remplacer chaque valeur vide par des Null pour chaque ligne
        row2 = replace_empty_with_null(row2)
        # Si chaque ligne contient 8 colonnes ALORS importer la ligne dans la BDD
        if len(row2) == 8:
            cursor.execute(query2, row2)
        else:
            print(f"Ligne non importer car nombre de colonne différent :
{row2}")
        # Importer les données dans la table
        connection.commit()
# Fermer la connection et l'importation
cursor.close()
connection.close()
print("Les données ont été importées.")





































```

Table Contrat : nombre de ligne = 30 335

SELECT * FROM table_contrat LIMIT 100												
	* Contrat_ID int	* No_voie int	B.T.Q varchar(1)	Type_de_voie varchar(5)	Voie varchar(100)	* Code_dep_code_comm varchar(5)	* Code_postal int	* Surface int	Type_local varchar(15)	Occupation varchar(15)	Type_contrat varchar(20)	Formule varchar(10)
>	100601	190	A	RUE	CENTRALE	1350	1370	50	Appartement	Locataire	Residence principale	Integral
>	100602	347	(NULL)	RUE	DU CHATEAU	1103	1170	48	Appartement	Locataire	Residence principale	Classique

Table Région : nombre de ligne = 38 916

SELECT * FROM table_region LIMIT 100								
	* Code_dep_code_comm varchar(10)	Code_region int	Nom_de_la_region varchar(100)	Nom_academie varchar(100)	Nom_departement varchar(100)	Nom_commune varchar(100)	Code_departement varchar(3)	Nom_et_numero_depart varchar(100)
>	10001	44	Grand Est	Reims	Aube	L' ABBAYE SOUS PLANCY	10	Aube (10)
>	10002	44	Grand Est	Reims	Aube	AILLEVILLE	10	Aube (10)

- ▼  region_contrat 12M
 - >  Query
 - ▼  Tables
 - ▼  table_contrat 29660
 - ▼  Columns
 -  Contrat_ID int
 -  No_voie int
 -  B_T_Q varchar(1)
 -  Type_de_voie varchar(5)
 -  Voie varchar(100)
 -  Code_dep_code_commune varchar(5)
 -  Code_postal int
 -  Surface int
 -  Type_local varchar(15)
 -  Occupation varchar(15)
 -  Type_contrat varchar(20)
 -  Formule varchar(10)
 -  Valeur_declaree_biens varchar(12)
 -  Prix_cotisation_mensuel int
 - >  Index
 - >  Partitions
 - ▼  table_region 37594
 - ▼  Columns
 -  Code_dep_code_commune varchar(10)
 -  Code_region int
 -  Nom_de_la_region varchar(100)
 -  Nom_academie varchar(100)
 -  Nom_departement varchar(100)
 -  Nom_commune varchar(100)
 -  Code_departement varchar(3)
 -  Nom_et_numero_departement varchar(100)
 - >  Index
 - >  Partitions
 - >  Views
 - >  Functions
 - >  Procedures

Correction des requêtes SQL

Requête exemple : Lister les contrats avec le prix de la cotisation et leur surface pour les appartements.

```
select contrat_id, prix_cotisation_mensuel, surface
from contrat
where type_local = 'Appartement'
```

	contrat ID	surface
1	103791	35
2	103792	99
3	103793	40
4	103794	20

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

```
SELECT c.Contrat_ID, c.Surface
FROM table_contrat c
LEFT JOIN table_region r ON r.Code_dep_code_commune =
    c.Code_dep_code_commune
WHERE r.Nom_commune = 'Caen'
;
```

Q	Nom_commune varchar	Contrat_ID int	Surface int
	Filter	Filter	Filter
>	CAEN	103791	35
>	CAEN	103792	99
>	CAEN	103793	40
>	CAEN	103794	20

Nb ligne : 4

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, r.Code_departement
FROM table_contrat c
LEFT JOIN table_region r ON r.Code_dep_code_commune =
    c.Code_dep_code_commune
WHERE r.Code_departement = 71
;
```

Q	Contrat_ID int	Type_contrat varchar	Formule varchar	Code_departement int
	Filter	Filter	Filter	Filter
>	114767	Residence secondaire	Classique	71
>	114768	Residence principale	Integral	71
>	114769	Residence principale	Classique	71
>	114770	Residence principale	Classique	71
>	114771	Residence principale	Integral	71
>	114772	Residence principale	Classique	71
>	114773	Residence principale	Integral	71
>	114774	Residence principale	Classique	71
>	114775	Residence principale	Integral	71
>	114776	Residence principale	Classique	71
>	114777	Residence principale	Integral	71
>	114778	Residence principale	Classique	71

Nb ligne : 48

Requête 3 : Lister le nom des régions de France.

```
SELECT DISTINCT (Nom_de_la_region)
FROM table_region
ORDER BY Nom_de_la_region ASC
;
```

Q	Nom_de_la_region varchar(100)
	Filter
>	Auvergne-Rhône-Alpes
>	Bourgogne-Franche-Comté
>	Bretagne
>	Centre-Val de Loire
>	Collectivités d'outre-mer
>	Corse
>	Grand Est
>	Guadeloupe
>	Guyane
>	Hauts-de-France
>	Ile-de-France
>	La Réunion
>	Martinique
>	Mayotte
>	Normandie
>	Nouvelle-Aquitaine
>	Occitanie

Nb ligne : 19

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

```
SELECT *
FROM table_contrat
ORDER BY Surface DESC
LIMIT 5
;
```

Contrat_ID int	No_voie int	B_T varchar(100)	Type_de_voie varchar(100)	Voie varchar(100)	Code_dep_code_comm int	Code_postal int	Surface int	Type_local varchar(100)	Occupation varchar(100)
Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
104211	34	B	CRS	VICTOR HUGO	17421	17600	815	Appartement	Locataire
105463	5229	(NULL)	RUE	DE LA PRIVADIÈRE	30126	30190	742	Maison	Propriétaire
130878	23	(NULL)	BD	DE BEAUSEJOUR	75116	75016	595	Appartement	Propriétaire
100822	5	A	RUE	FERNAND BAZIN	2201	2600	570	Maison	Propriétaire
109872	18	(NULL)	RUE	PAUL BELLAMY	44109	44000	559	Appartement	Locataire

Nb ligne : 30 335 (5 avec le LIMIT 5)

Requête 5 : Quel est le prix moyen de la cotisation mensuelle ?

```
SELECT ROUND(AVG(Prix_cotisation_mensuel),2) as Prix_moyen  
FROM table_contrat  
;
```

Prix_moyen newdecimal
Filter
19.33

Nb ligne : 1

Requête 6 : Quel est le nombre de contrats pour chaque catégorie de prix de la valeur déclarée des biens ?

```
SELECT Valeur_declaree_biens, COUNT(Contrat_ID) as nb_contrat  
FROM table_contrat  
GROUP BY Valeur_declaree_biens  
ORDER BY nb_contrat DESC  
;
```

Valeur_declaree_biens varchar(100)	nb_contrat bigint
Filter	Filter
0-25000	22720
25000-50000	6815
50000-100000	696
100000+	104

Nb ligne : 4

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

```
SELECT r.Nom_de_la_region, COUNT(c.Formule) as nb_formule  
FROM table_contrat c  
LEFT JOIN table_region r ON r.Code_dep_code_commune =  
c.Code_dep_code_commune  
WHERE c.Formule = "Integral"  
and r.Nom_de_la_region = 'Pays de la Loire'  
;
```

Nom_de_la_region varchar	nb_formule bigint
Filter	Filter
Pays de la Loire	589

Nb ligne : 1

Requête 8 : 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
FROM table_contrat c
LEFT JOIN table_region r ON r.Code_dep_code_commune =
c.Code_dep_code_commune
WHERE r.Code_departement = 71
;

```

Contrat_ID int	Type_contrat varchar	Formule varchar
Filter	Filter	Filter
114767	Residence secondaire	Classique
114768	Residence principale	Integral
114769	Residence principale	Classique
114770	Residence principale	Classique
114771	Residence principale	Integral
114772	Residence principale	Classique
114773	Residence principale	Integral
114774	Residence principale	Classique
114775	Residence principale	Integral
114776	Residence principale	Classique
114777	Residence principale	Integral

Nb ligne : 48

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

```

SELECT r.Nom_academie, ROUND(AVG(c.Surface),2) as surface_moyenne
FROM table_contrat c
LEFT JOIN table_region r ON r.Code_dep_code_commune =
c.Code_dep_code_commune
WHERE r.Nom_academie = 'Paris'
;

```

Nom_academie varchar	surface_moyenne newdecimal
Filter	Filter
Paris	51.77

Nb ligne : 1

Requête 10 : Classement des 10 départements où le prix moyen de la cotisation est le plus élevé.

```
SELECT Code_dep_code_commune, Prix_cotisation_mensuel
FROM table_contrat
ORDER BY Prix_cotisation_mensuel DESC
LIMIT 10
;
```

Code_dep_code_commune int	Prix_cotisation_mensuel int
Filter	Filter
75116	450
91174	430
75107	429
75117	382
75106	381
75101	378
75116	372
75116	361
75101	354
75101	332

Nb ligne : 30 335 (10 avec le LIMIT 10)

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

```
SELECT r.Nom_commune, COUNT(c.Contrat_ID) as nb_contrat
FROM table_contrat c
LEFT JOIN table_region r on r.Code_dep_code_commune =
c.Code_dep_code_commune
GROUP BY r.Nom_commune
HAVING nb_contrat >= 150
ORDER BY nb_contrat DESC
;
```

Nom_commune varchar	nb_contrat bigint
Filter	Filter
PARIS 18	515
PARIS 17	468
PARIS 15	407
PARIS 16	394
NICE	387
PARIS 11	381
BORDEAUX	302
PARIS 20	302
NANTES	291

Nb ligne : 20

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

```
SELECT r.Nom_de_la_region, COUNT(c.Contrat_ID) as nb_contrat
FROM table_contrat c
LEFT JOIN table_region r ON r.Code_dep_code_commune =
    c.Code_dep_code_commune
WHERE r.Nom_de_la_region IS NOT NULL
GROUP BY r.Nom_de_la_region
ORDER BY r.Nom_de_la_region ASC
;
```

Nom_de_la_region varchar	nb_contrat bigint
Filter	Filter
Auvergne-Rhône-Alpes	3042
Bourgogne-Franche-Comté	293
Bretagne	947
Centre-Val de Loire	598
Corse	247
Grand Est	769
Hauts-de-France	1189
Ile-de-France	14177
Normandie	824
Nouvelle-Aquitaine	2038
Occitanie	1609
Pays de la Loire	1196
Provence-Alpes-Côte d'Azur	3279

Nb ligne : 13