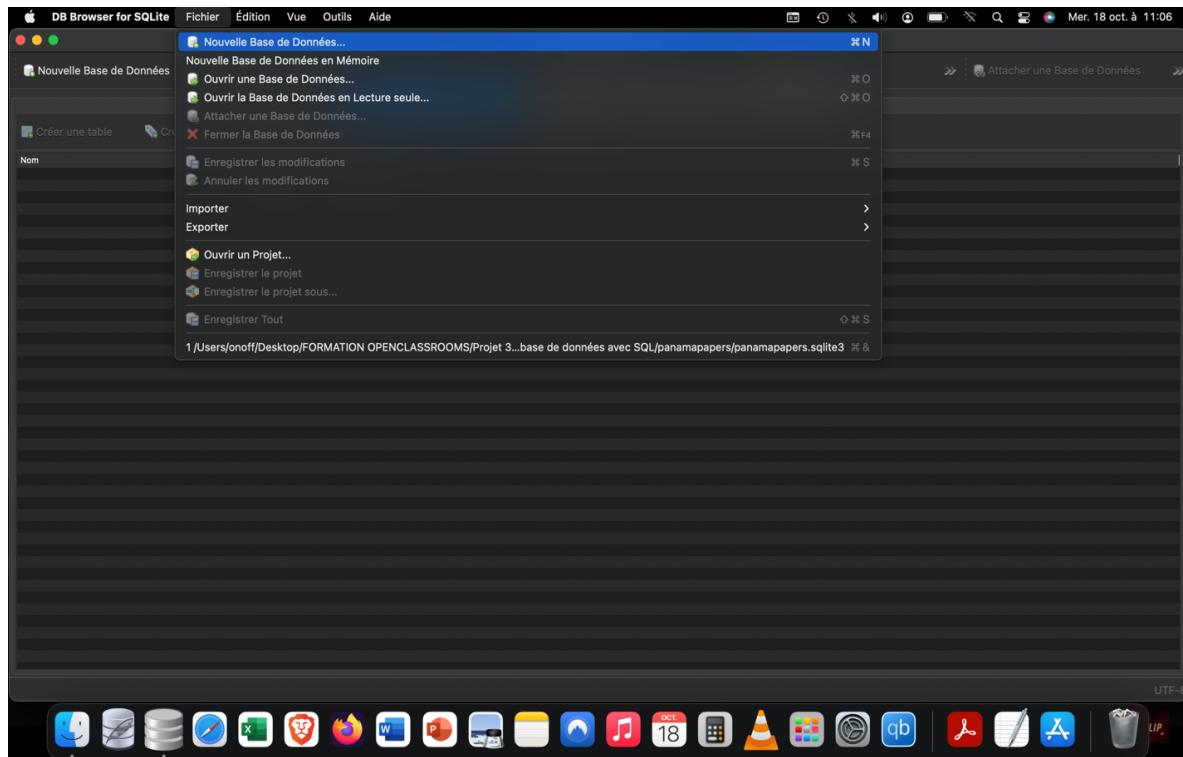


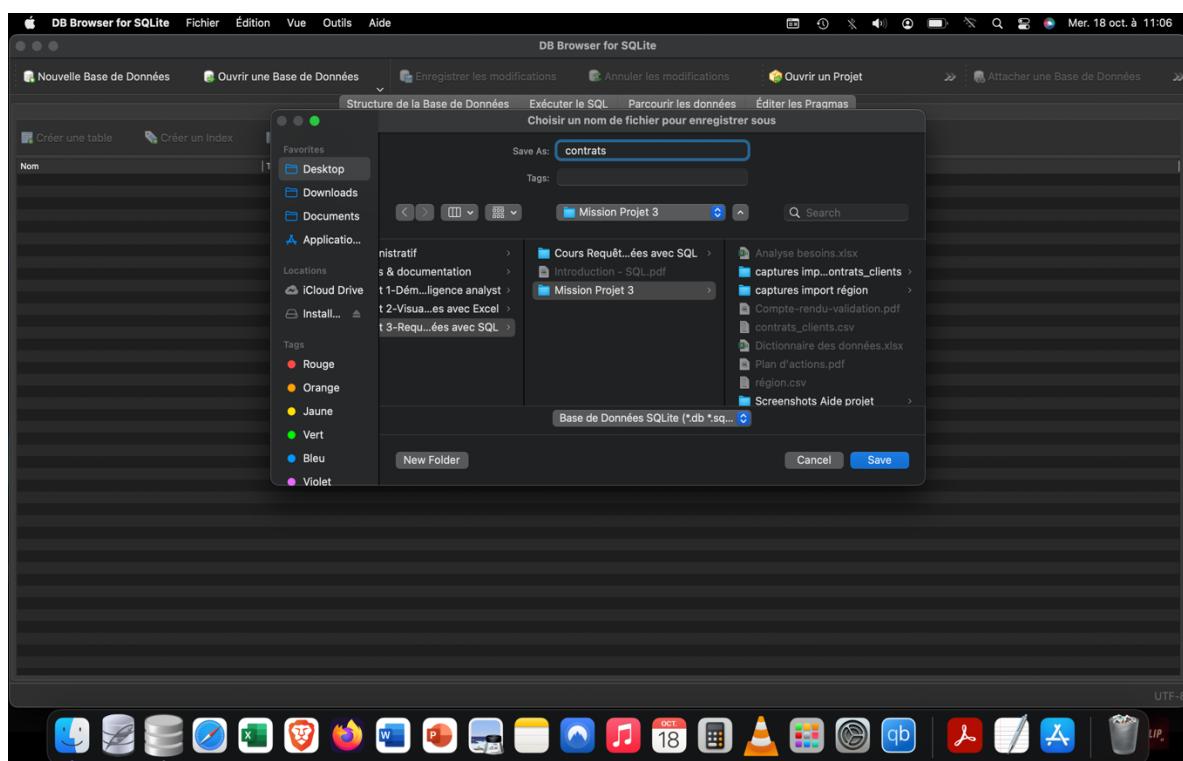
DOCUMENT TECHNIQUE

1/ Procédure de chargement de la base de données contenant les données clients.

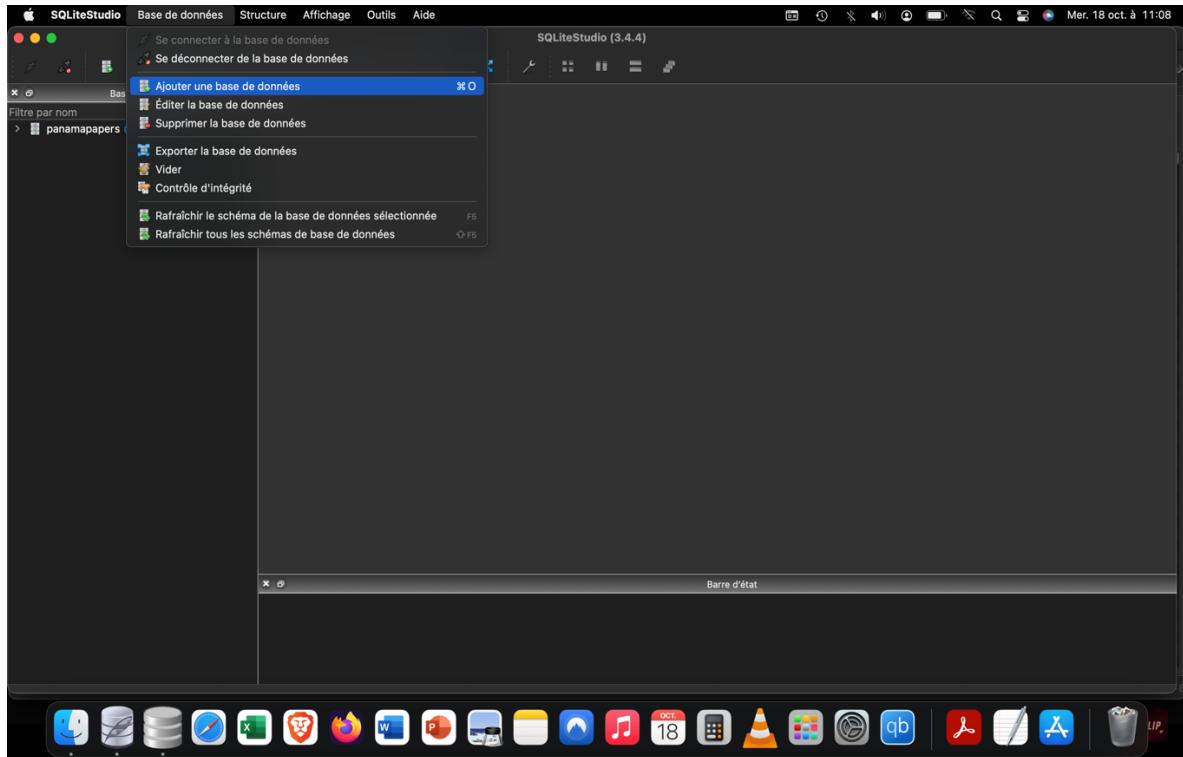
- Création d'une base de données nommée « **contrats** » avec **DB Browser for SQLite**



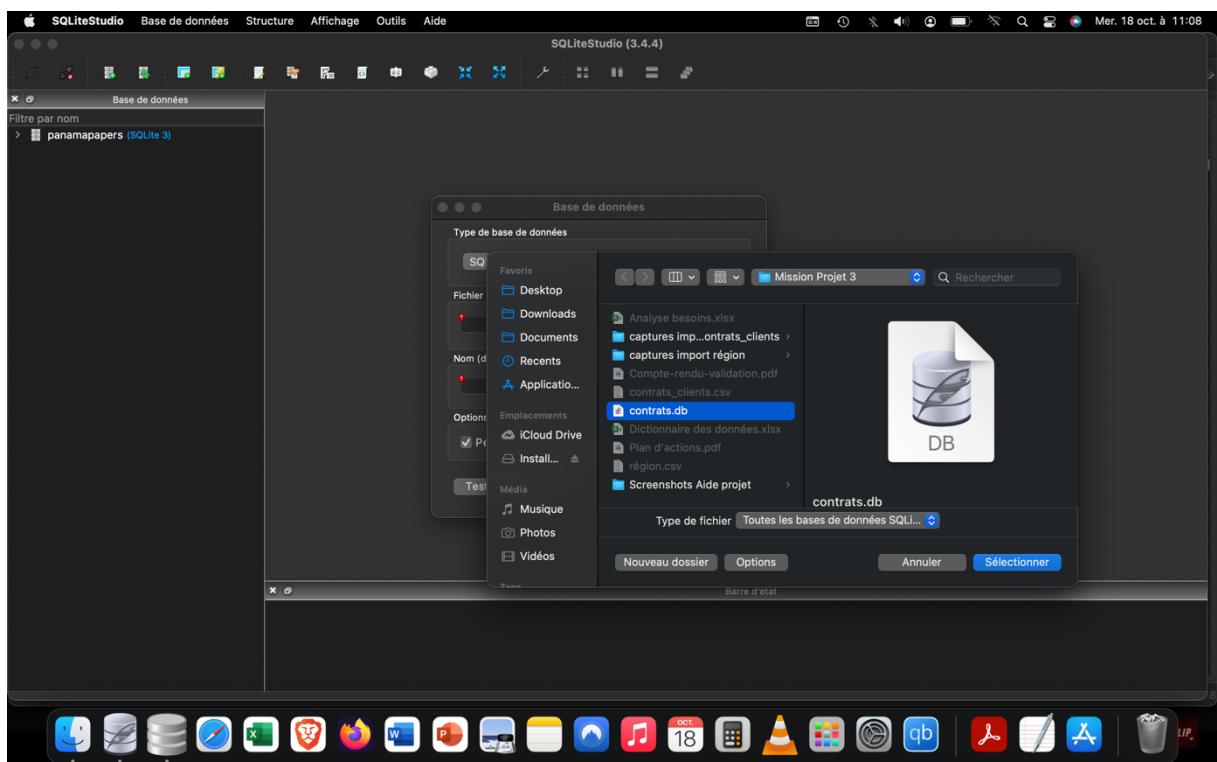
- Enregistrement du nom de fichier sous nommé « **contrats** »



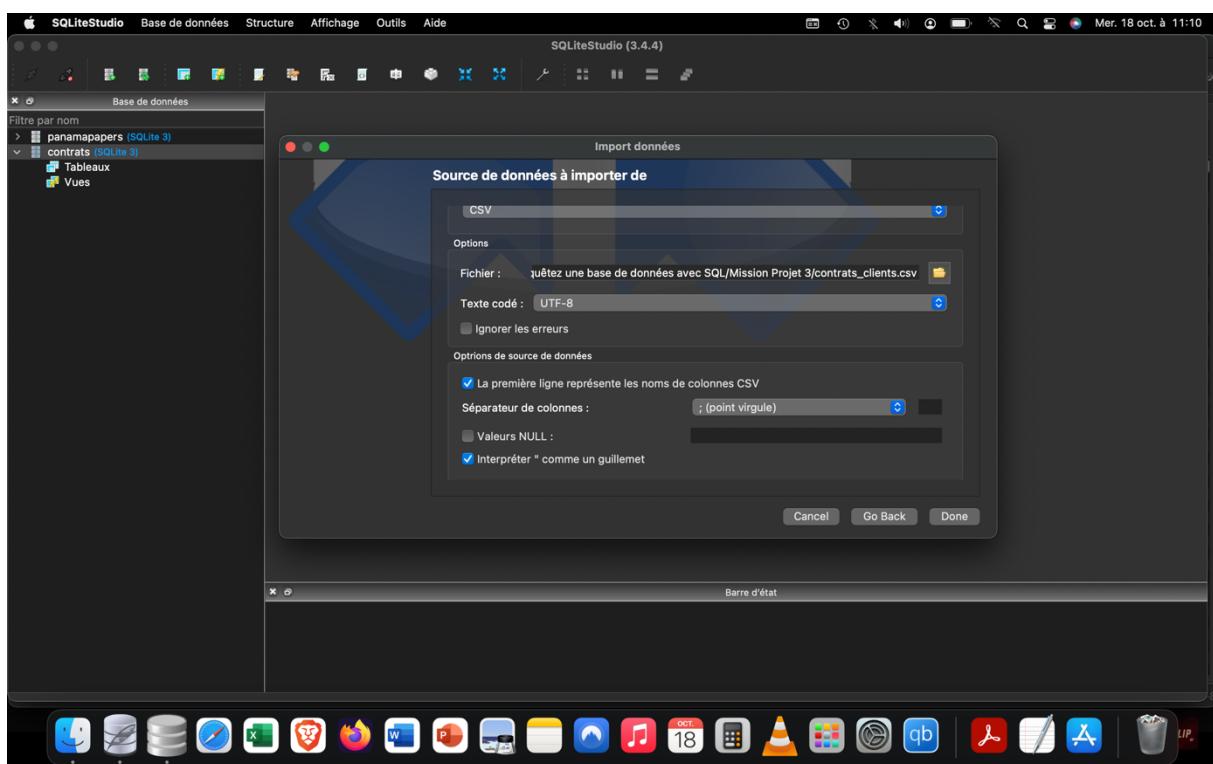
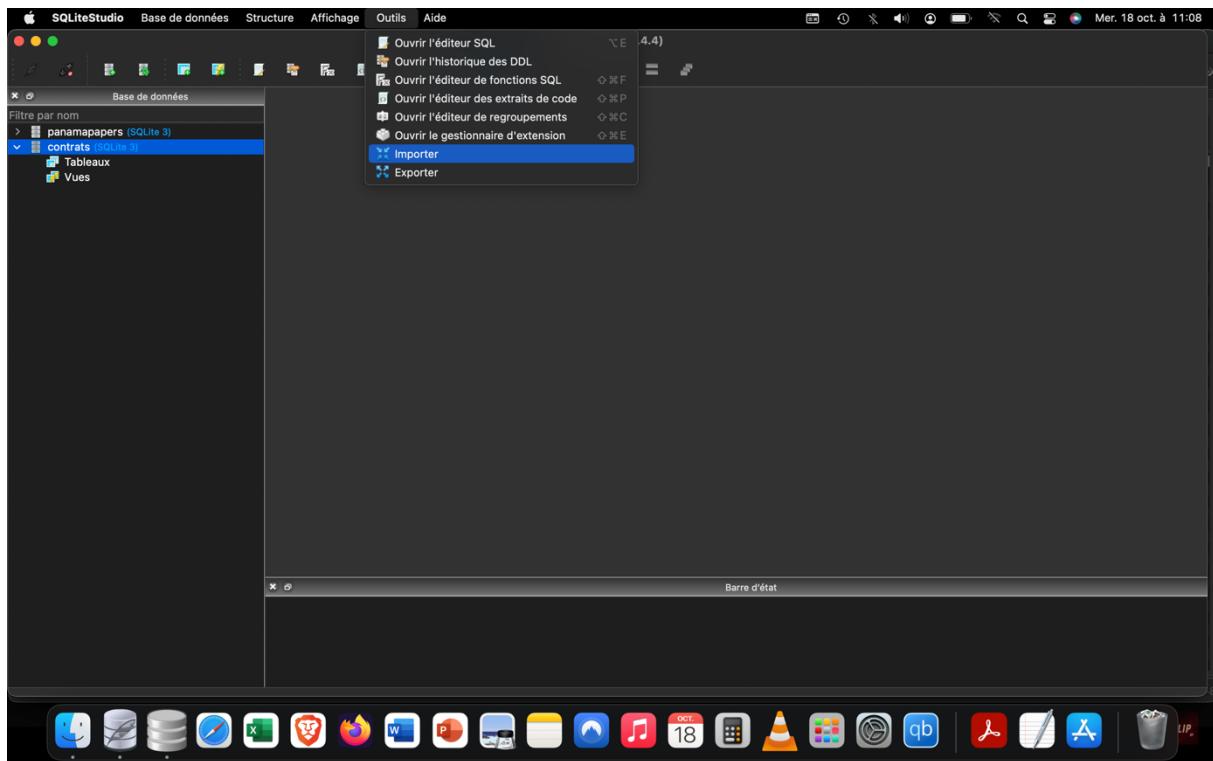
- Ajout d'une base de données avec l'outil **SQLite Studio**



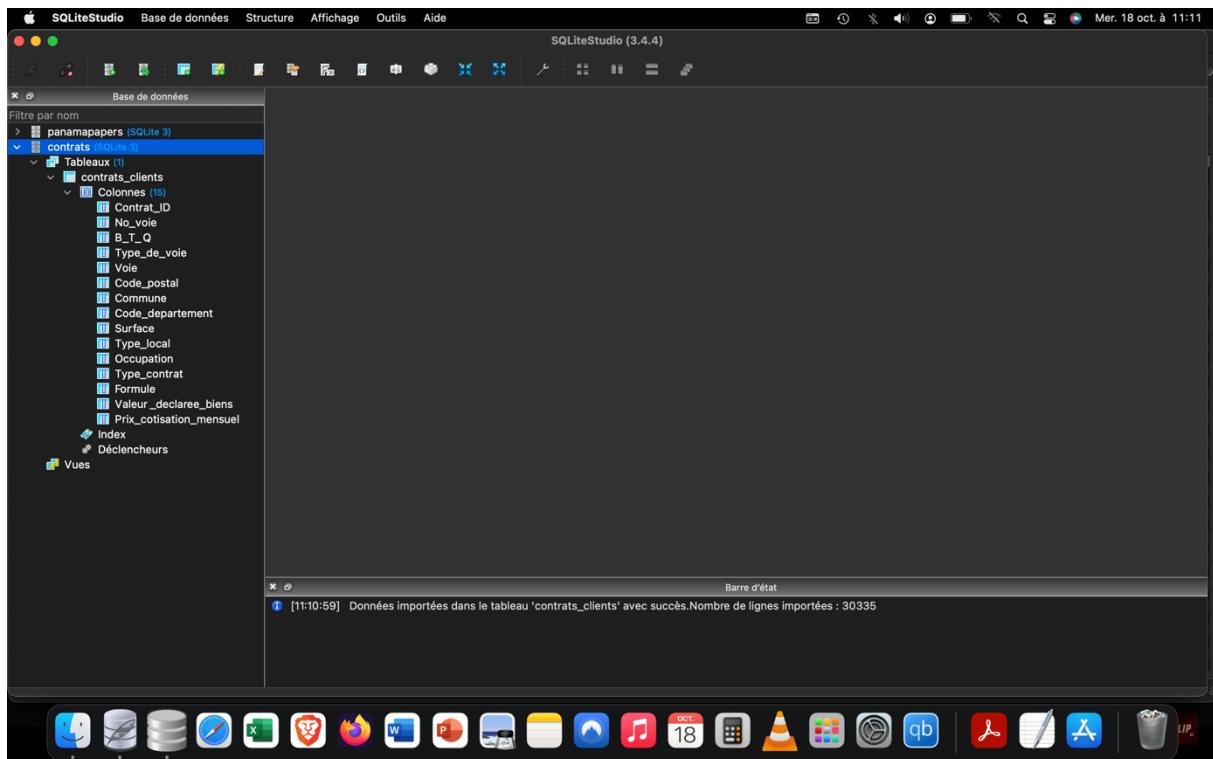
- Sélection du fichier nommé « **contrats.db** »



- Import du fichier CSV

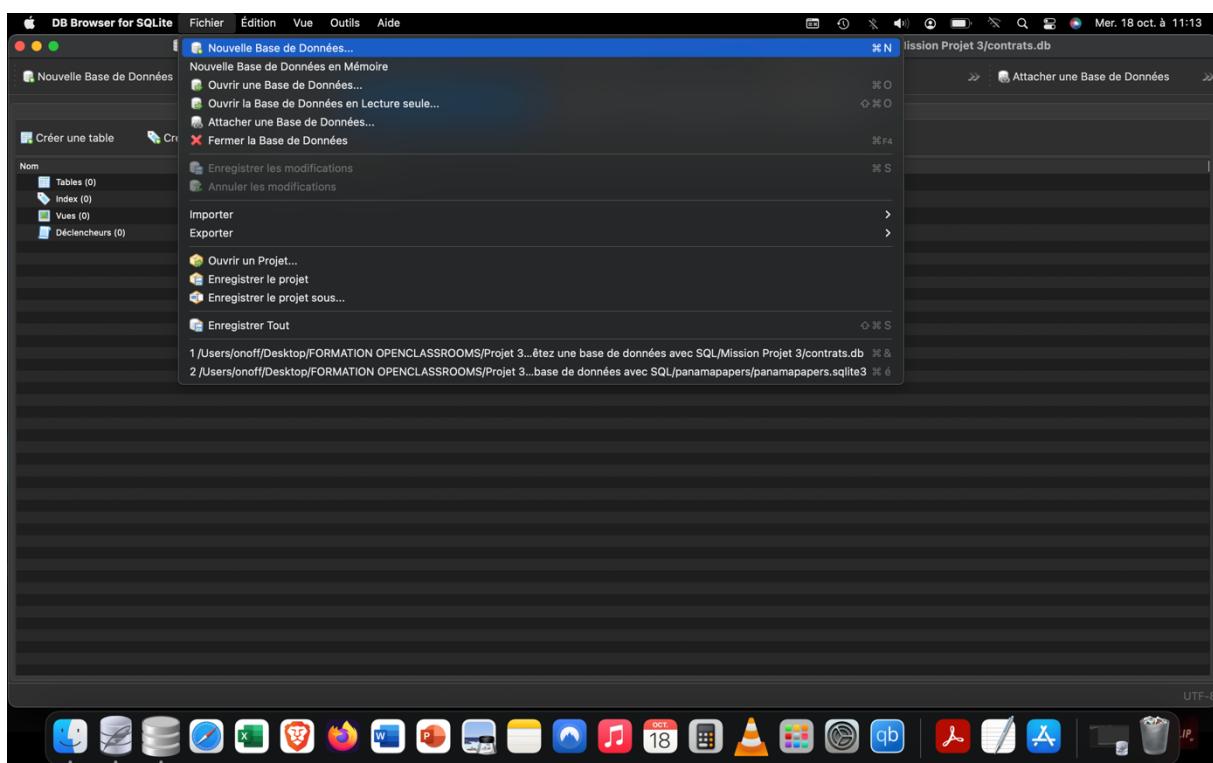


- Données importées dans le tableau « **contrats_clients** » avec succès

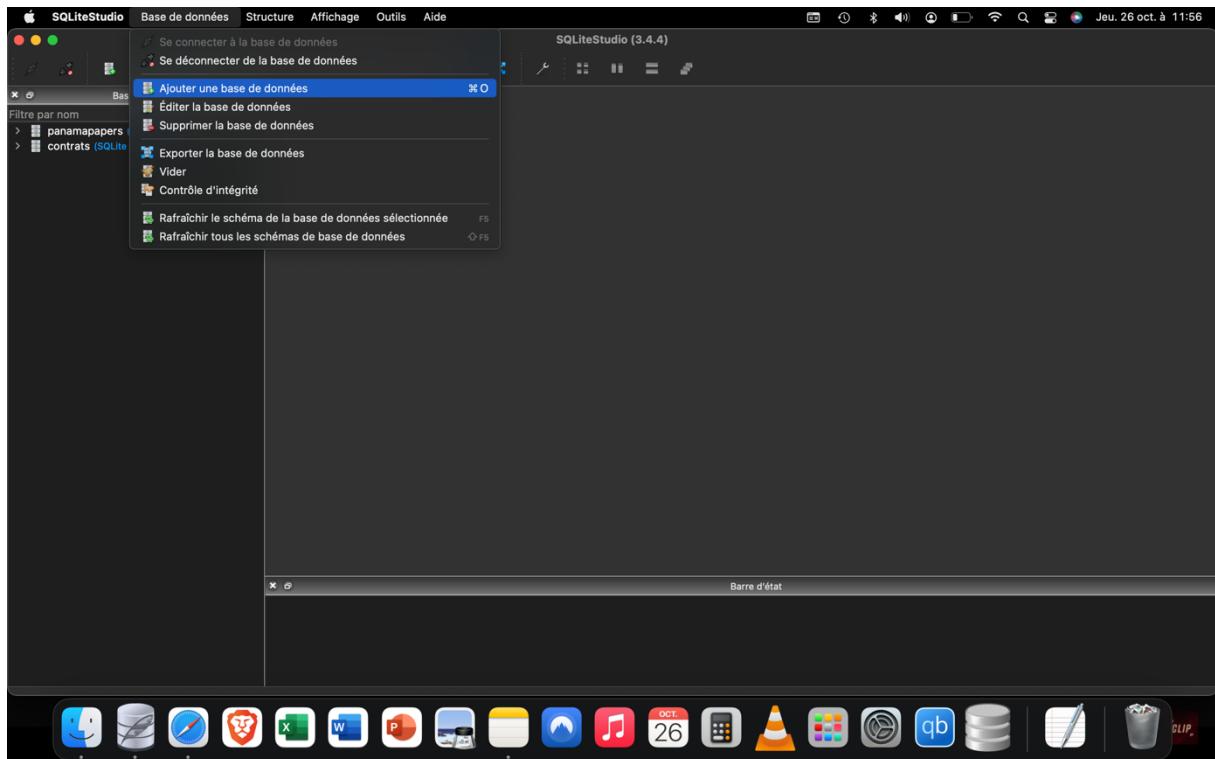


2/ Procédure de chargement de la base de données contenant les données du référentiel régions.

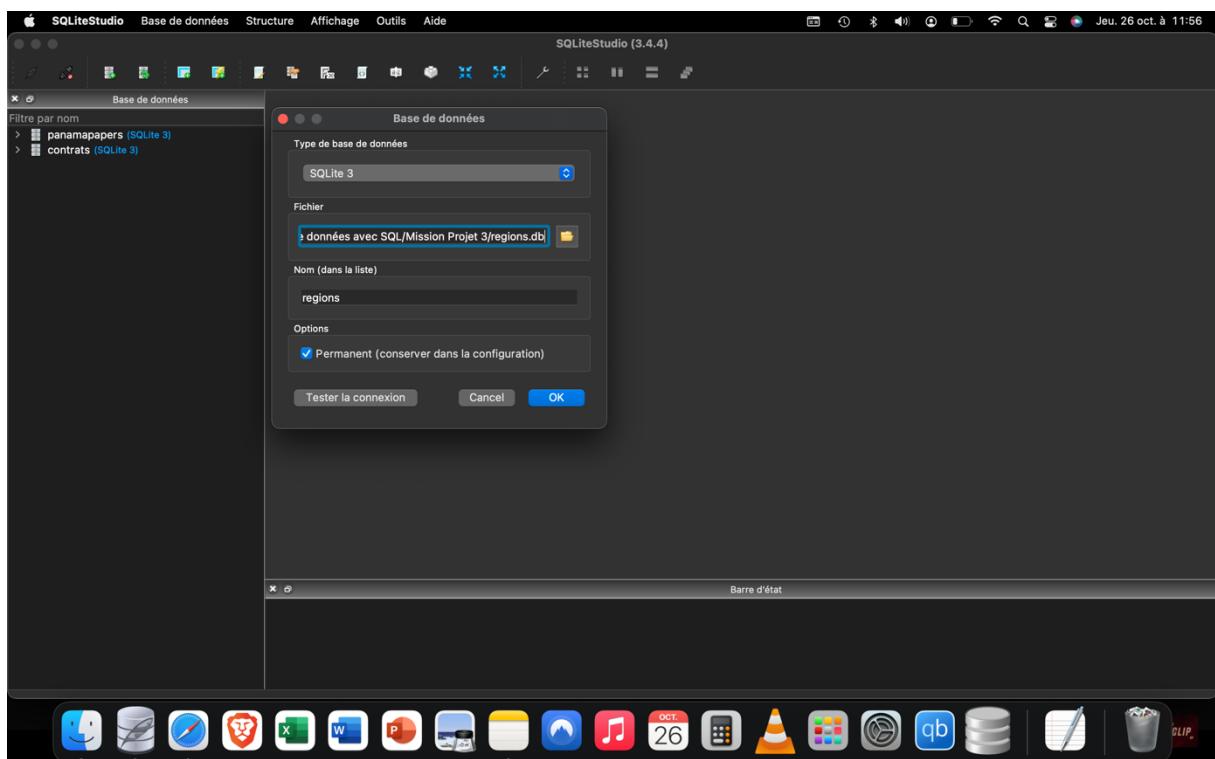
- Création d'une base de données nommée « **regions** » avec **DB Browser for SQLite**



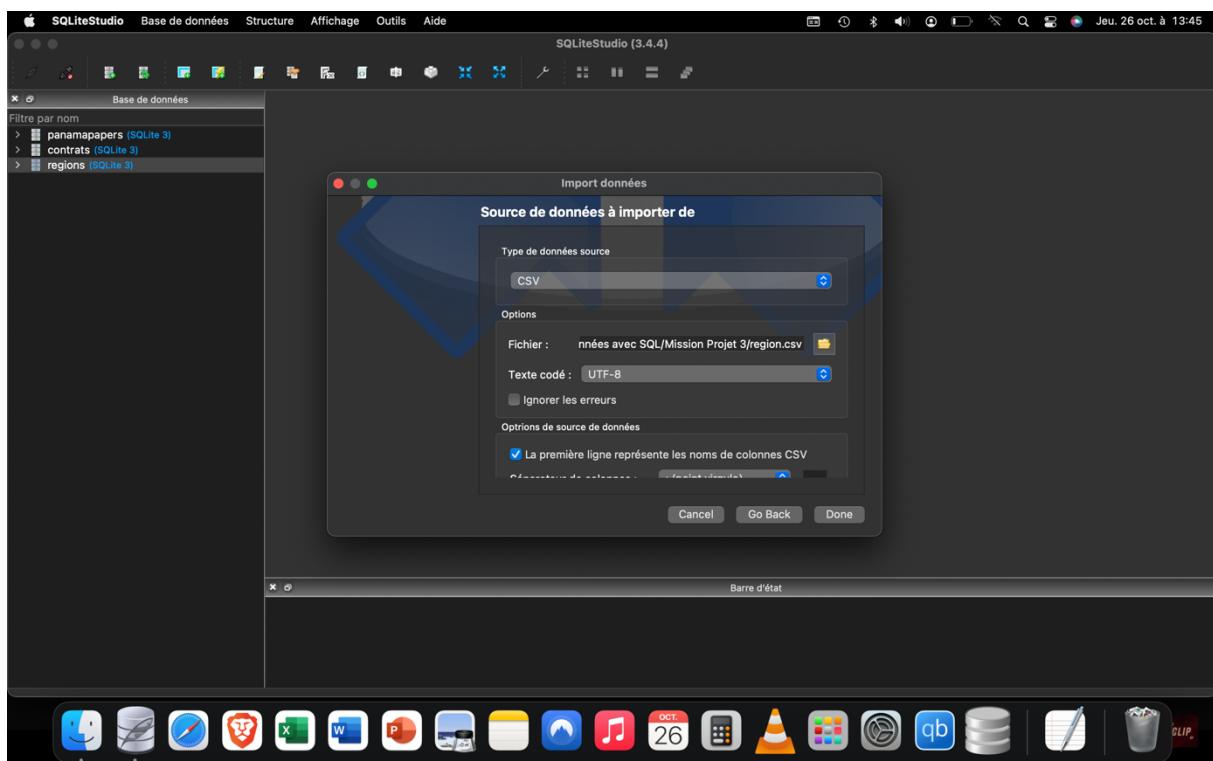
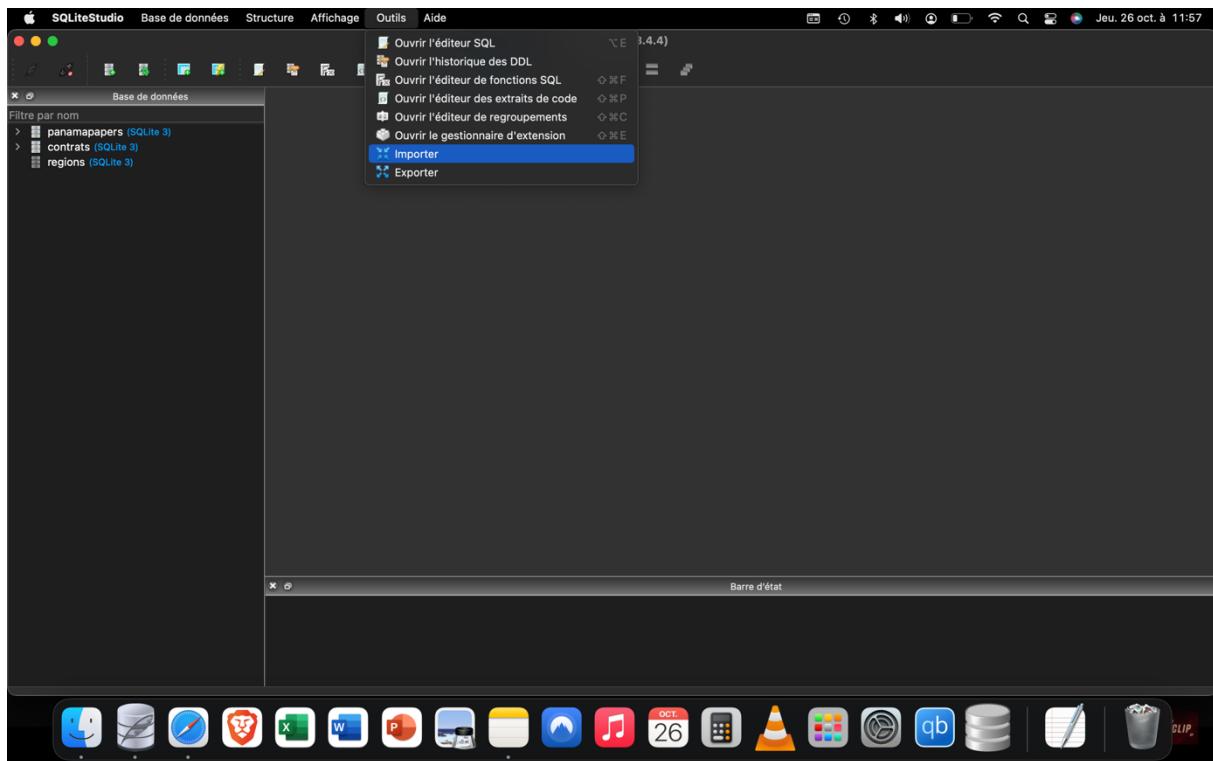
- Ajout d'une base de données dans **SQLite Studio**



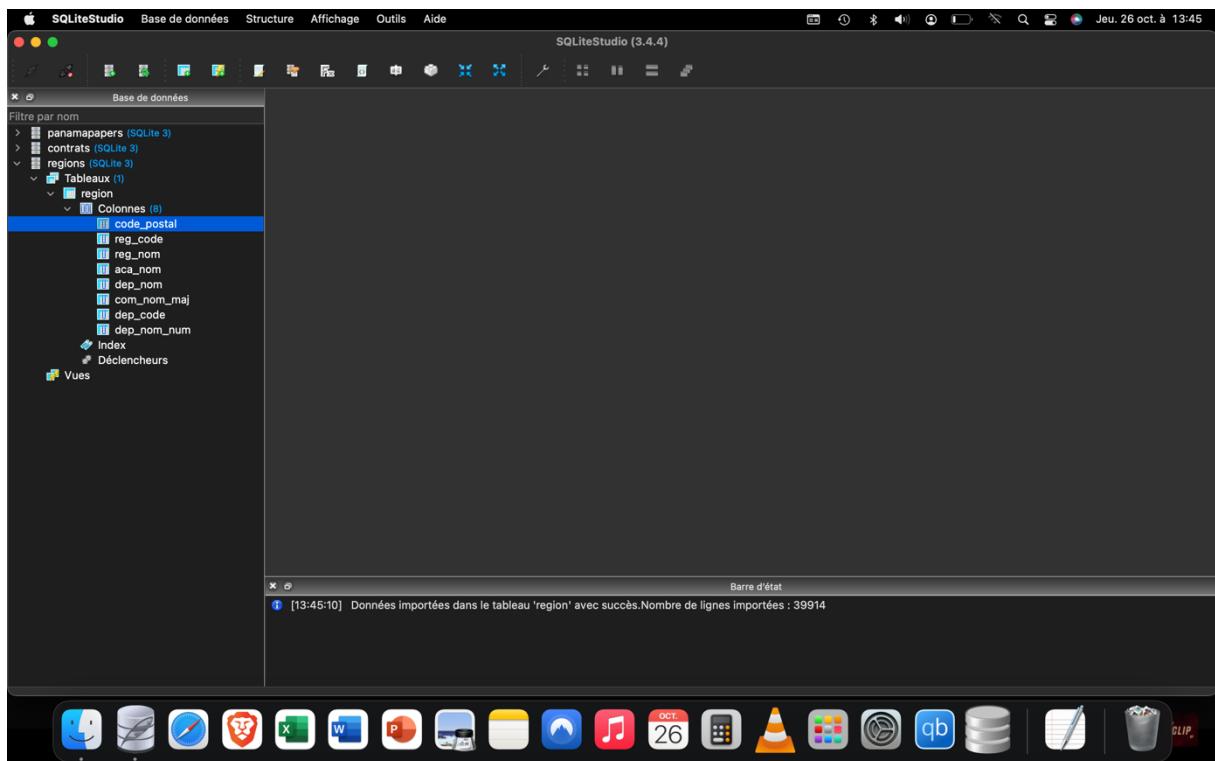
- Sélection du fichier nommé « **region.db** »



- Import du fichier CSV



- Données importées dans le tableau nommé « **region** »

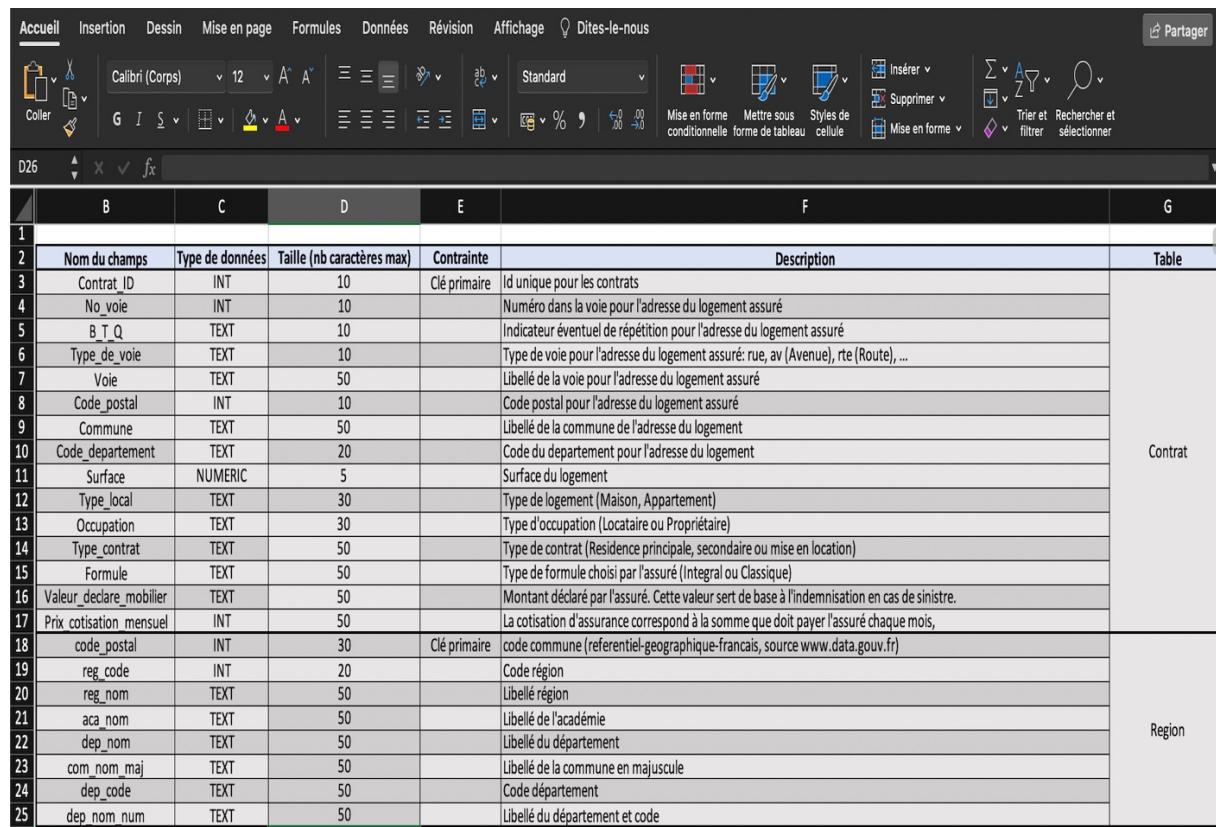


2/ Dictionnaire de données

En SQL, les données sont séparées en plusieurs types (par exemple : texte, nombre entier, date...). Lorsque nous définissons une colonne dans une table de la base, il faut donc lui donner un type, et toutes les données stockées dans cette colonne devront correspondre au type de la colonne. Choisir un mauvais type de données pourrait entraîner :

- **Un gaspillage de mémoire** (ex. : stocker de toutes petites données dans une colonne faite pour stocker de grosses quantités de données) ;
- **Des problèmes de performances** (ex. : il est plus rapide de faire une recherche sur un nombre que sur une chaîne de caractères) ;
- L'impossibilité d'utiliser des **fonctionnalités propres à un type de données** (ex. : stocker une date comme chaîne de caractères nous prive des nombreuses fonctions temporelles disponibles).

D'où l'importance du dictionnaire des données présenté ci-dessous pour exécuter correctement les requêtes.



The screenshot shows a Microsoft Excel spreadsheet with a table titled "Contrat". The table has columns for "Nom du champs", "Type de données", "Taille (nb caractères max)", "Contrainte", and "Description". The "Contrainte" column contains specific constraints like "Clé primaire" and "Type de voie". The "Description" column provides detailed explanations for each field. The table is part of a larger document where another table titled "Region" is also visible on the right side.

	B	C	D	E	F	G
1						
2	Nom du champs	Type de données	Taille (nb caractères max)	Contrainte	Description	Table
3	Contrat_ID	INT	10	Clé primaire	Id unique pour les contrats	Contrat
4	No_voie	INT	10		Numéro dans la voie pour l'adresse du logement assuré	
5	B_T_Q	TEXT	10		Indicateur éventuel de répétition pour l'adresse du logement assuré	
6	Type_de_voie	TEXT	10		Type de voie pour l'adresse du logement assuré: rue, av (Avenue), rte (Route), ...	
7	Voie	TEXT	50		Libellé de la voie pour l'adresse du logement assuré	
8	Code_postal	INT	10		Code postal pour l'adresse du logement assuré	
9	Commune	TEXT	50		Libellé de la commune de l'adresse du logement	
10	Code_departement	TEXT	20		Code du département pour l'adresse du logement	
11	Surface	NUMERIC	5		Surface du logement	
12	Type_local	TEXT	30		Type de logement (Maison, Appartement)	
13	Occupation	TEXT	30		Type d'occupation (Locataire ou Propriétaire)	
14	Type_contrat	TEXT	50		Type de contrat (Résidence principale, secondaire ou mise en location)	
15	Formule	TEXT	50		Type de formule choisi par l'assuré (Integral ou Classique)	
16	Valeur_declare_mobilier	TEXT	50		Montant déclaré par l'assuré. Cette valeur sert de base à l'indemnisation en cas de sinistre.	
17	Prix cotisation_mensuel	INT	50		La cotisation d'assurance correspond à la somme que doit payer l'assuré chaque mois,	
18	code_postal	INT	30	Clé primaire	code commune (referentiel-geographique-francais, source www.data.gouv.fr)	Region
19	reg_code	INT	20		Code région	
20	reg_nom	TEXT	50		Libellé région	
21	aca_nom	TEXT	50		Libellé de l'académie	
22	dep_nom	TEXT	50		Libellé du département	
23	com_nom_maj	TEXT	50		Libellé de la commune en majuscule	
24	dép_code	TEXT	50		Code département	
25	dep_nom_num	TEXT	50		Libellé du département et code	

3/ Analyse des 3 premières requêtes

A	B	C	D	E
	Besoin	Informations recherchées	Clauses	Requête
1	Lister les numéros de contrats (contrat_ID) avec leur surface pour la commune de Caen	Numéros de contrats (contrat_ID), surface, commune Caen	select from where	select contrat_id, surface, commune from contrats_clients where commune = 'CAEN'
2	Lister les numéros de contrats (contrat_ID), avec le type de contrat et leur formule pour les maisons du département de la Saône-et-Loire (Département 71)	Numéros de contrats (contrat_ID), type de contrat et leur formule, type local (maison), code département 71	select from where And	select contrat_ID, type_contrat, formule, type_local, code_departement from contrats_clients where code_departement = '71' and type_local = 'Maison'
3	Lister le nom des régions de France	régions de France (reg_nom)	select distinct	select distinct reg_nom from region

- **Requête 1 : lister des numéros de contrats (contrat_ID) avec leur surface pour la commune de Caen**

The screenshot shows the SQLiteStudio interface. On the left, the database structure is displayed in a tree view. Under the 'contrats' table, there is a 'contrats_clients' table which contains columns: contrat_id, surface, commune, Contrat_ID, No_voie, B_T_Q, Type_de_voie, Voie, Code_postal, Commune, Code_departement, Surface, Type_local, Occupation, Type_contrat, Formule, Valeur_declaree_biens, Prix_cotisation_mensuel, Index, and Déclencheurs. A query is being run in the central editor:

```
1 select contrat_id,surface,commune from contrats_clients
2 where commune = 'CAEN'
```

The results are shown in a table below:

	contrat_id	surface	commune
1	103791	35	CAEN
2	103792	99	CAEN
3	103793	40	CAEN
4	103794	20	CAEN

At the bottom, the status bar shows several log entries:

- [10:21:54] Requête terminée en 0.001 seconde(s).
- [10:23:35] Erreur pendant l'exécution de la requête sur la base de données « contrats » : no such column: contrat_id
- [10:23:52] Requête terminée en 0.001 seconde(s).
- [10:24:38] Requête terminée en 0.001 seconde(s).
- [10:24:47] Requête terminée en 0.002 seconde(s).

- **Requête 2** : lister les numéros de contrats, avec le type de contrat et leur formule pour les maisons du département de la Saône-et-Loire (département 71)

The screenshot shows the SQLiteStudio interface with the following details:

- Left Panel (Base de données):** Shows the database structure with tables: contrats_clients, contrats, and régions.
- Central Panel (Éditeur SQL 1):** Displays the SQL query:


```
1 SELECT contrat_ID, type_contrat, formule, type_local, code_departement
2 FROM contrats_clients
3 WHERE
4   code_departement = '71'
5   and type_local = 'Maison'
6
```
- Table View:** Shows the results of the query:

contrat_ID	type_contrat	formule	type_local	code_departement
114768	Résidence principale	Integral	Maison	71
114782	Résidence principale	Classique	Maison	71
114812	Résidence principale	Integral	Maison	71
114779	Résidence principale	Classique	Maison	71
- Log Window (Barre d'état):** Shows the execution log with several entries indicating successful imports and query completions.

- **Requête 3** : lister le nom des régions de France

The screenshot shows the SQLiteStudio interface with the following details:

- Left Panel (Base de données):** Shows the database structure with tables: contrats_clients and régions.
- Central Panel (Éditeur SQL 1):** Displays the SQL query:


```
1 select distinct reg_nom from région
```
- Table View:** Shows the results of the query:

reg_nom
Auvergne-Rhône-Alpes
Hauts-de-France
Provence-Alpes-Côte d'Azur
Grand Est
Occitanie
Normandie
Nouvelle-Aquitaine
Centre-Val de Loire
Corse
Bourgogne-Franche-Comté
- Log Window (Barre d'état):** Shows the execution log with several entries indicating successful imports and query completions.