

Opérations sur les données

**Nous allons travailler avec une Base de Données BDD
dans un fichier unique avec SQLite** 

Travail sur les données des régions et des départements en France.

Les sources sur Internet

On va télécharger d'abord sur le site de l'INSEE

<https://www.insee.fr/fr/information/4316069> Le « Code officiel géographique au 1er janvier 2020 » sous forme de fichier csv .

Voici le fichier à télécharger :

https://www.insee.fr/fr/statistiques/fichier/4316069/cog_ensemble_2020_csv.zip

Il faut décompresser ce fichier avec 7zip par exemple.

On travaillera uniquement avec 2 fichiers :

region2020.csv departement2020.csv

Ces 2 fichiers peuvent être ouverts avec un tableur comme Calc de LibreOffice ou Excel.

Nous allons les insérer dans une base de données BDD au format SQLite.

Travail sur les données des régions et des départements en France.

2 solutions pour travailler avec SQLite :

1) Installer DB Browser for SQLite <https://sqlitebrowser.org/dl/>

Vous pouvez télécharger ce fichier qui peut être mis dans n'importe quel dossier sans droit administrateur :

<https://github.com/sqlitebrowser/sqlitebrowser/releases/download/v3.12.1/DB.Browser.for.SQLite-3.12.1-win64.zip>

2) En ligne : <https://sqliteonline.com/>

Solution utilisée dans ce diaporama

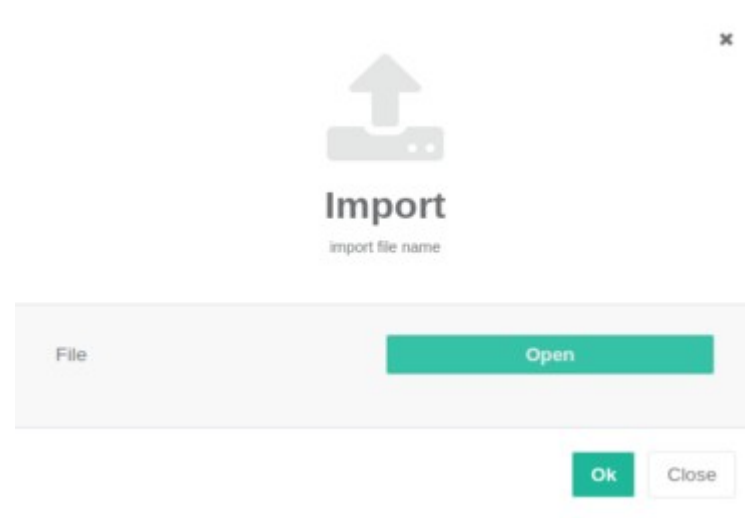
Cliquer à gauche sur SQLite

On va importer des tables avec Import

Dans les menus de la ligne du haut, cliquer sur Import

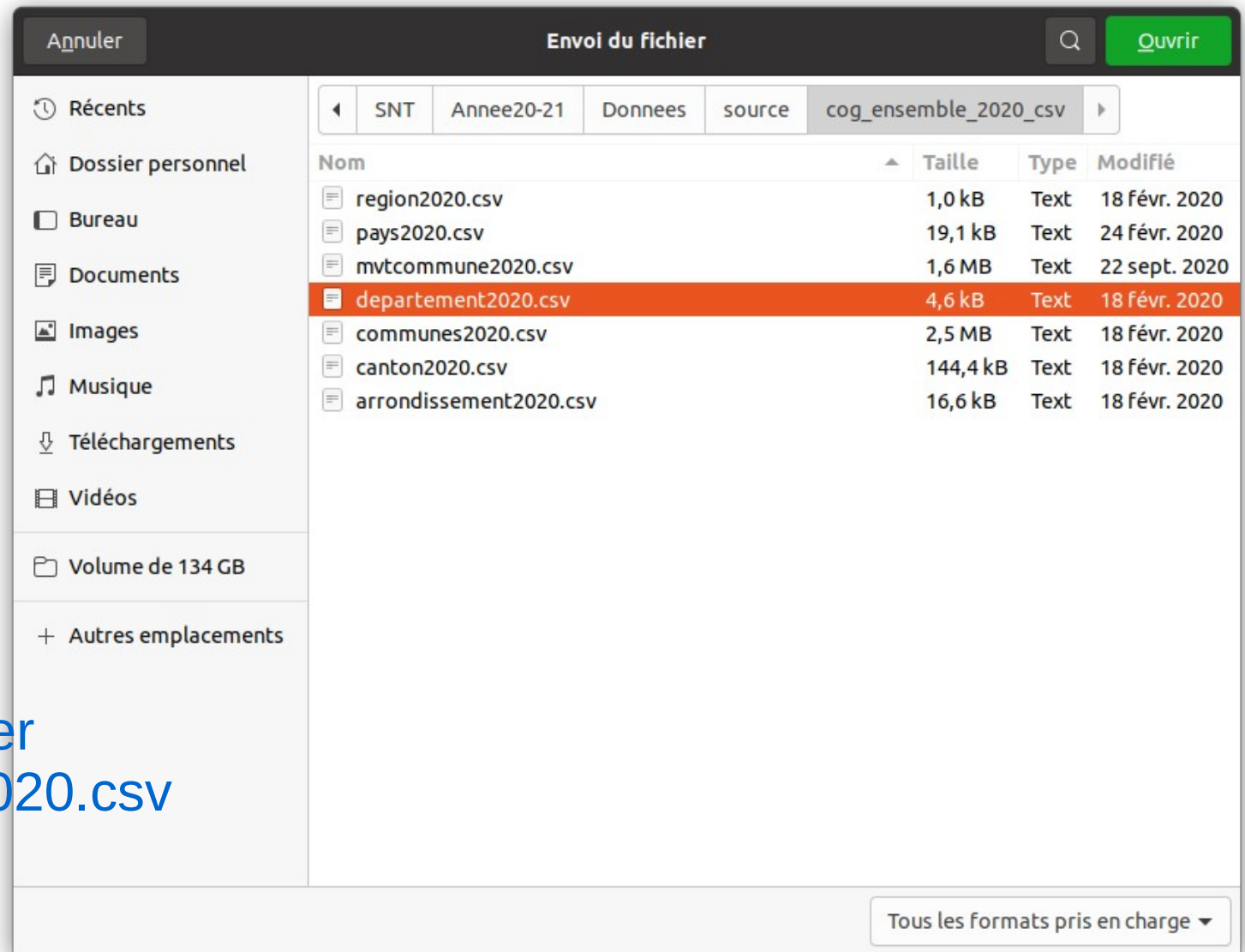
Travail sur les données des régions et des départements en France.

Dans les menus de la ligne du haut, cliquer sur Import



Clic sur Open

Travail sur les données des régions et des départements en France.



Choisir le fichier
departement2020.csv

Clic sur Ouvrir

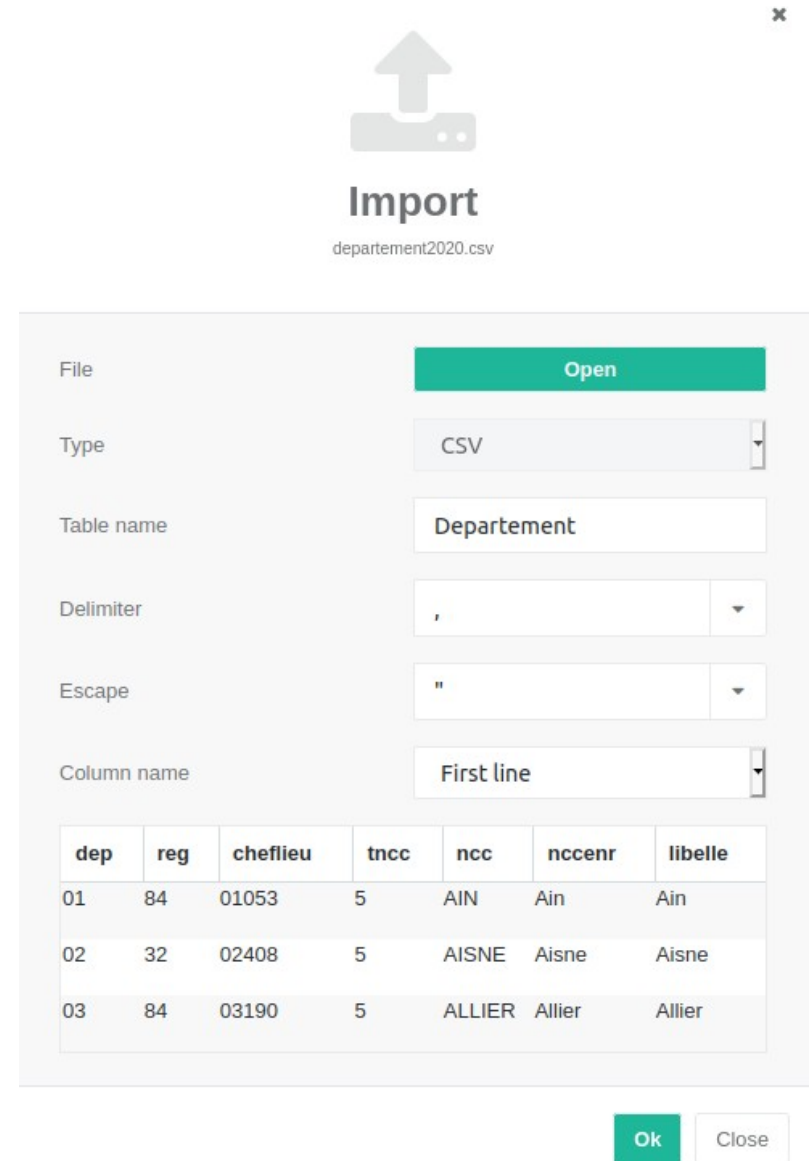
Travail sur les données des régions et des départements en France.

On vérifie que le fichier importer s'appelle departement2020.csv

Important, il faut remplir :

Table name avec **Departement** (sans accent)

Il faut remplacer **New-auto** dans **Column name** par **First line**



The screenshot shows a web-based data import interface. At the top, there is a large upward arrow icon and the text 'Import' followed by 'departement2020.csv'. Below this, there are several configuration options:

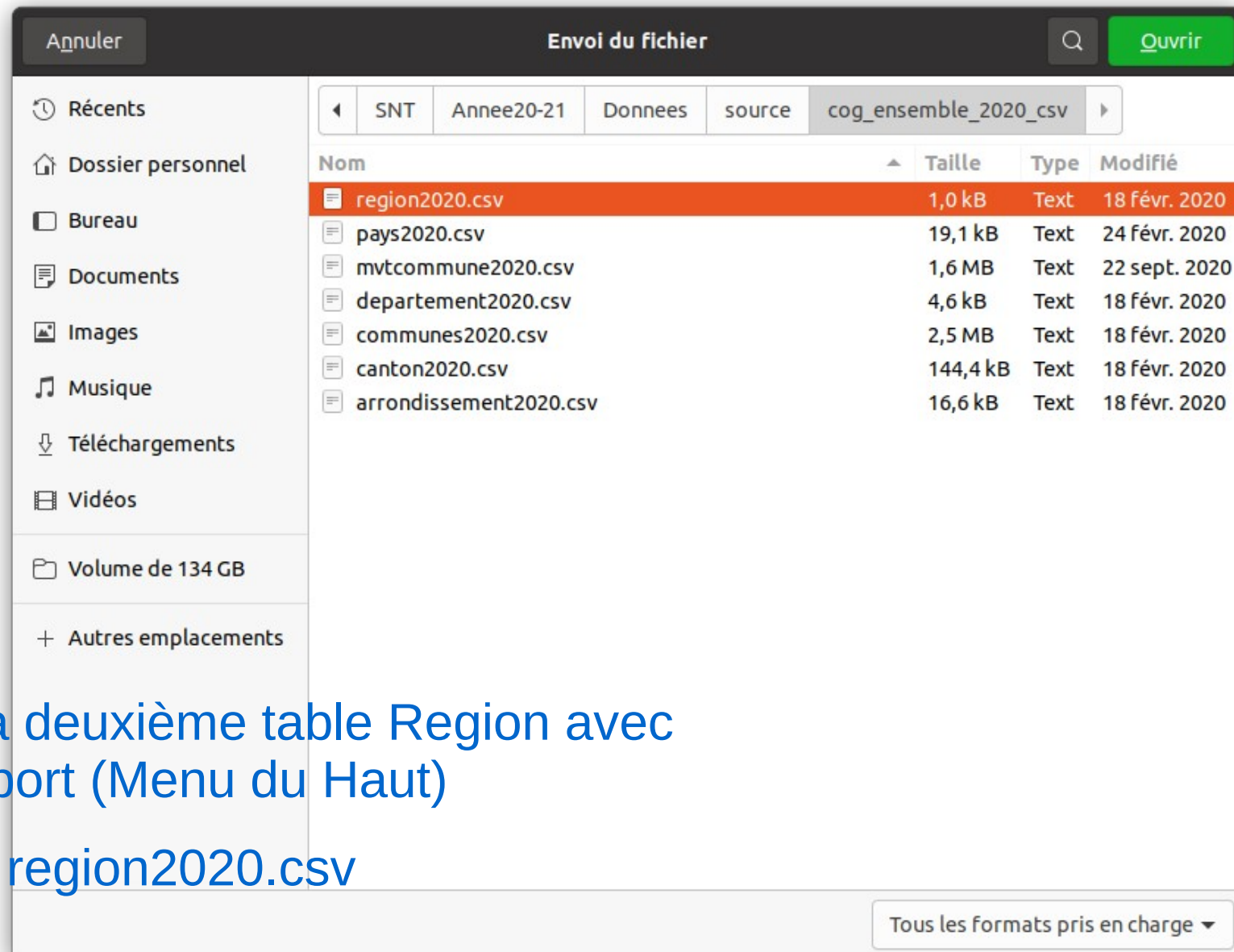
- File**: A green 'Open' button.
- Type**: A dropdown menu set to 'CSV'.
- Table name**: A text input field containing 'Departement'.
- Delimiter**: A dropdown menu set to ','.
- Escape**: A dropdown menu set to '"'.
- Column name**: A dropdown menu set to 'First line'.

Below these options is a preview table with 7 columns: dep, reg, cheflieu, tncc, ncc, nccentr, and libelle. The table contains 3 rows of data:

dep	reg	cheflieu	tncc	ncc	nccentr	libelle
01	84	01053	5	AIN	Ain	Ain
02	32	02408	5	AISNE	Aisne	Aisne
03	84	03190	5	ALLIER	Allier	Allier

At the bottom right, there are two buttons: 'Ok' (green) and 'Close' (white).

Travail sur les données des régions et des départements en France.



On va ajouter la deuxième table Region avec de nouveau Import (Menu du Haut)

Ouvrir le fichier region2020.csv

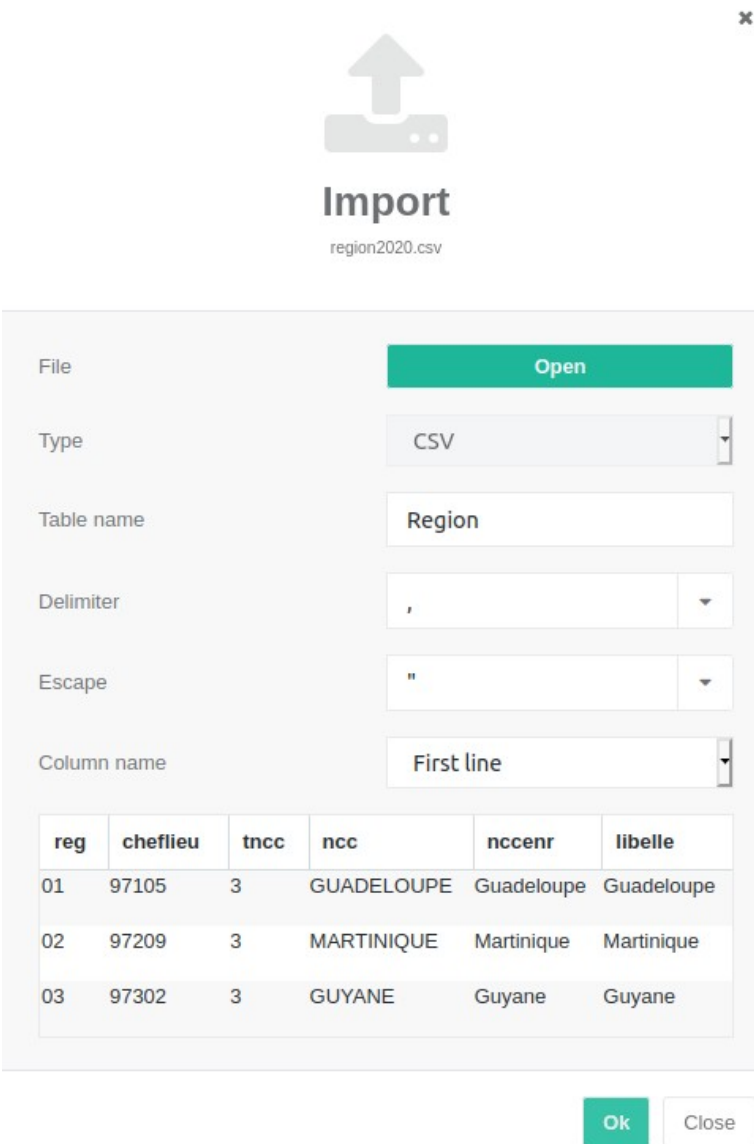
Travail sur les données des régions et des départements en France.

On vérifie que le fichier importer s'appelle region2020.csv

Important, il faut remplir :

Table name avec **Region** (sans accent)

Il faut remplacer **New-auto** dans **Column name** par **First line**



File

Type

Table name

Delimiter

Escape

Column name

reg	cheflieu	tncc	ncc	nccenr	libelle
01	97105	3	GADELOUPE	Guadeloupe	Guadeloupe
02	97209	3	MARTINIQUE	Martinique	Martinique
03	97302	3	GUYANE	Guyane	Guyane

Travail sur les données des régions et des départements en France.

Double-clic sur la table Region (colonne de gauche)



The screenshot shows a database application interface. At the top, there is a green header bar with a menu icon, a 'File' dropdown, 'Owner DB', a 'Run' button, and 'Share', 'Export', and 'Import' buttons. On the left, a sidebar shows a tree view with 'SQLite' selected, containing 'Table' (demo, Departement, Region) and 'Column' (reg INTEGER, cheflieu TEXT, tncc INTEGER, ncc TEXT, nccentr TEXT, libelle TEXT). The main area displays a SQL query: 'SELECT * FROM Region'. Below the query, a table of results is shown with 6 columns: reg, cheflieu, tncc, ncc, nccentr, and libelle. The table contains 7 rows of data representing French regions.

reg	cheflieu	tncc	ncc	nccentr	libelle
1	97105	3	GUADELOUPE	Guadeloupe	Guadeloupe
2	97209	3	MARTINIQUE	Martinique	Martinique
3	97302	3	GUYANE	Guyane	Guyane
4	97411	0	LA REUNION	La Réunion	La Réunion
6	97608	0	MAYOTTE	Mayotte	Mayotte
11	75056	1	ILE DE FRANCE	Île-de-France	Île-de-France
24	45234	2	CENTRE VAL DE LOIRE	Centre-Val de Loire	Centre-Val de Loire

Tous les champs sont affichés, la vue affichée correspond à la requête SQL :

`SELECT * FROM Region ;`

Sélectionner tous les champs (*) depuis la table Région

Travail sur les données des régions et des départements en France.

Si on ne veut afficher que 2 champs, voici un exemple :

`SELECT reg, libelle FROM Region ;`

`SELECT` permet donc de choisir les champs que l'on souhaite voir.

`FROM` indique la table utilisée, ici `Region`

N'oubliez pas d'exécuter la requête avec **► RUN**

Il faut mettre un « ; » à la fin de la requête pour être compatible avec d'autres logiciels.

The screenshot shows a database application interface. On the left, there is a sidebar with a tree view showing the database structure. The 'SQLite' database is selected, and the 'Region' table is expanded, showing its columns: 'reg' (INTEGER), 'cheflieu' (TEXT), 'tncc' (INTEGER), 'ncc' (TEXT), 'nccenr' (TEXT), and 'libelle' (TEXT). The main area displays the SQL query: `1 SELECT reg, libelle FROM Region`. Below the query, the results are shown in a table with two columns: 'reg' and 'libelle'. The results list 18 regions of France, each with a unique 'reg' value and its corresponding 'libelle'.

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

Travail sur les données des régions et des départements en France.

Requêtes SQL

Nous venons de faire notre première requête, mais c'est quoi une requête ?

Juste à titre informatif, voici une requête SELECT qui possède presque toutes les commandes possibles :

SELECT *

FROM table

WHERE condition

GROUP BY expression

HAVING condition

{ UNION | INTERSECT | EXCEPT }

ORDER BY expression

A noter : cette requête imaginaire sert principalement d'aide-mémoire pour retenir l'ordre d'apparition de chacune des commandes au sein d'une requête.

Travail sur les données des régions et des départements en France.

Changeons de table en double-cliquant sur la table Département à gauche.

Tous les champs sont affichés, la vue affichée correspond à la commande SQL :

`SELECT * FROM Département ;`



The screenshot shows a database application interface. On the left, a sidebar lists tables: SQLite (demo, Département, Region) and MariaDB. The main area displays a table view for the 'Département' table. The table has 7 columns: dep, reg, cheflieu, tncc, ncc, nccenr, and libelle. The data is as follows:

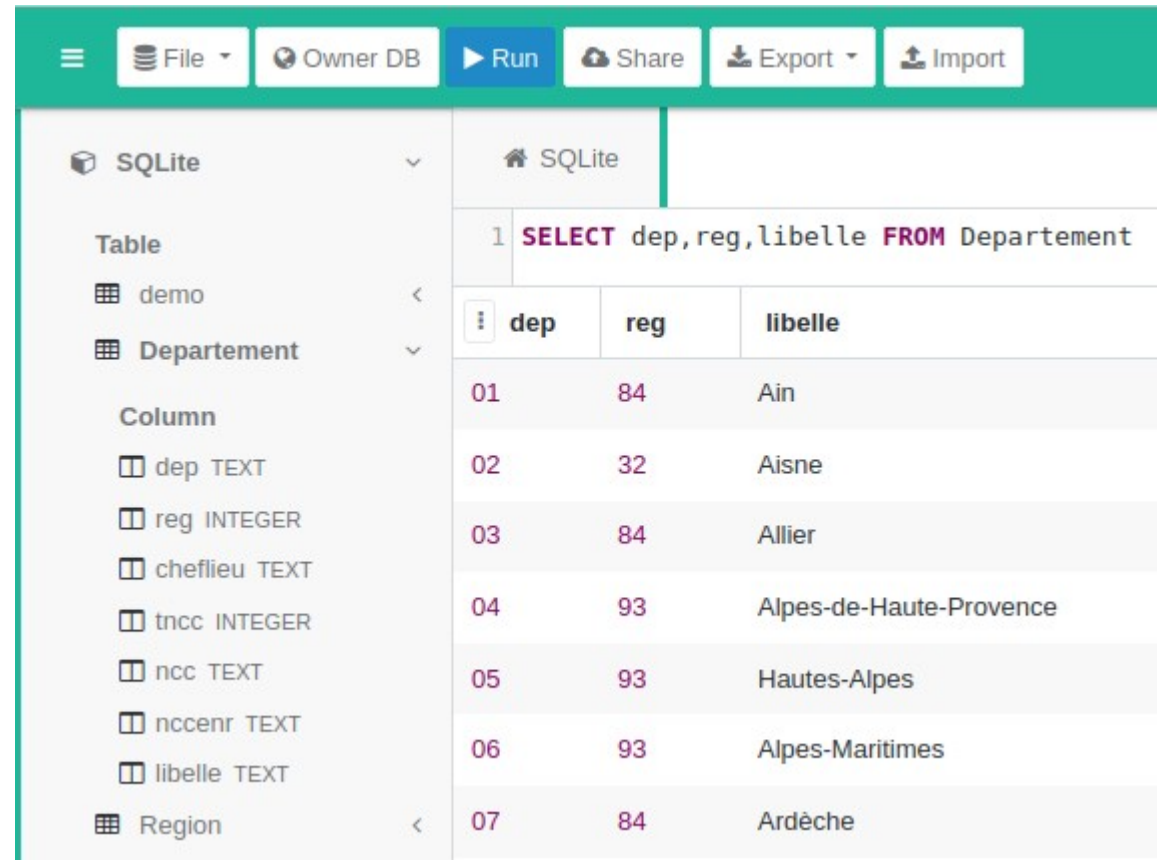
dep	reg	cheflieu	tncc	ncc	nccenr	libelle
01	84	01053	5	AIN	Ain	Ain
02	32	02408	5	AINES	Aisne	Aisne
03	84	03190	5	ALLIER	Allier	Allier

Travail sur les données des régions et des départements en France.

Si je ne veux afficher que le numéro des départements et le numéro des régions ainsi que le nom des départements

On pourra faire la requête SQL :

```
SELECT dep,reg,libelle  
FROM Departement ;
```



The screenshot shows a SQLite database interface. On the left, a sidebar lists the database structure: 'demo' (Table), 'Departement' (Table), and 'Region' (Table). Under 'Departement', the columns are listed: 'dep' (TEXT), 'reg' (INTEGER), 'cheflieu' (TEXT), 'tncc' (INTEGER), 'ncc' (TEXT), 'nccenr' (TEXT), and 'libelle' (TEXT). The main area displays a SQL query: 'SELECT dep,reg,libelle FROM Departement'. Below the query, the results are shown in a table with three columns: 'dep', 'reg', and 'libelle'. The results list seven departments: Ain (01, 84), Aisne (02, 32), Allier (03, 84), Alpes-de-Haute-Provence (04, 93), Hautes-Alpes (05, 93), Alpes-Maritimes (06, 93), and Ardèche (07, 84).

dep	reg	libelle
01	84	Ain
02	32	Aisne
03	84	Allier
04	93	Alpes-de-Haute-Provence
05	93	Hautes-Alpes
06	93	Alpes-Maritimes
07	84	Ardèche

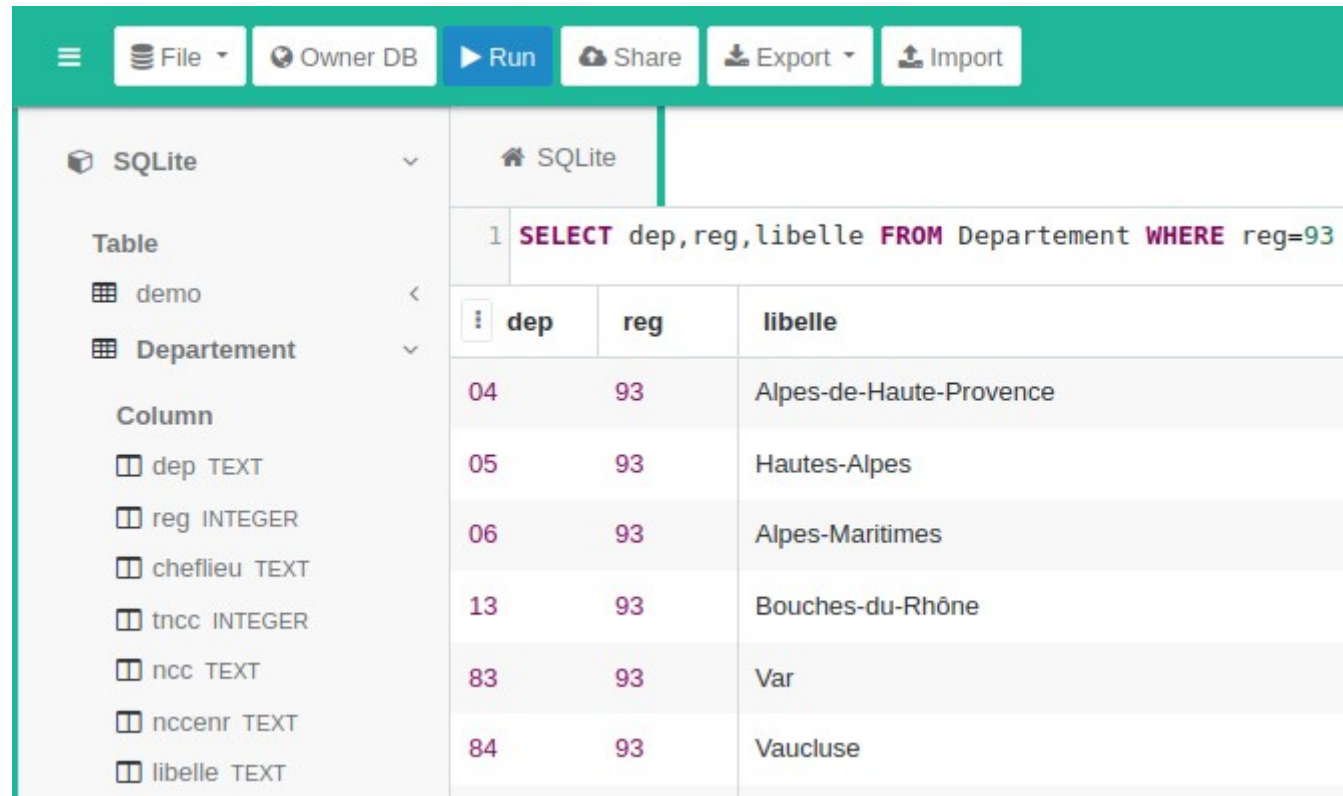
Travail sur les données des régions et des départements en France.

Si je regarde les premiers départements dans la diapositive précédente, je vois que 3 départements sont dans la même région dont le numéro est 93.

Je peux avoir la liste des départements de la région dont le numéro est 93 avec la requête SQL :

```
SELECT dep,reg,libelle
FROM Departement
WHERE reg=93
```

Cette instruction
WHERE + condition
permet de n'afficher que
les lignes qui respectent
cette condition.



The screenshot shows a SQLite database interface. On the left, a sidebar lists the database structure: 'Table' (demo, Departement) and 'Column' (dep TEXT, reg INTEGER, cheflieu TEXT, tncc INTEGER, ncc TEXT, nccentr TEXT, libelle TEXT). The main area displays a SQL query: 'SELECT dep,reg,libelle FROM Departement WHERE reg=93'. Below the query, the results are shown in a table with three columns: 'dep', 'reg', and 'libelle'. The results list six departments with 'reg' value 93.

dep	reg	libelle
04	93	Alpes-de-Haute-Provence
05	93	Hautes-Alpes
06	93	Alpes-Maritimes
13	93	Bouches-du-Rhône
83	93	Var
84	93	Vaucluse

Travail sur les données des régions et des départements en France.

Malheureusement, le numéro 93 n'est pas très pratique, quelle est cette région ?

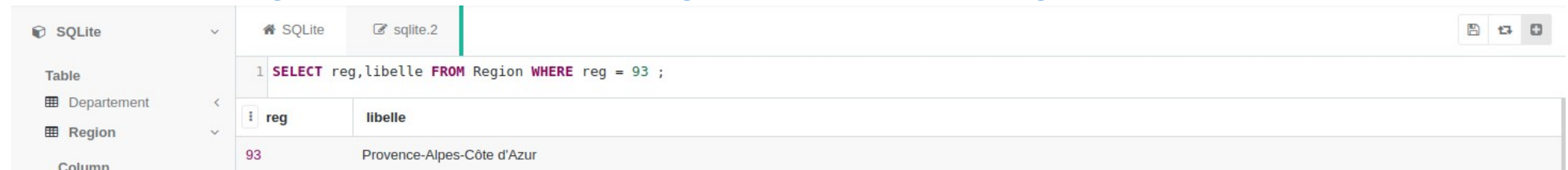
Si je remonte à la requête SQL : `SELECT reg,libelle FROM Region ;`

L'avant dernière ligne m'indique :

reg 93 **libelle** Provence-Alpes-Côte d'Azur

Je peux aussi obtenir cette ligne (row) avec la requête SQL :

`SELECT reg,libelle FROM Region WHERE reg=93 ;`



The screenshot shows a SQLite database interface. On the left, a sidebar lists 'Table' (Departement, Region) and 'Column'. The main area displays a query: `1 SELECT reg,libelle FROM Region WHERE reg = 93 ;`. Below the query, the results are shown in a table with two columns: 'reg' and 'libelle'. The first row contains the values '93' and 'Provence-Alpes-Côte d'Azur'.

reg	libelle
93	Provence-Alpes-Côte d'Azur

Remarque : on peut ajouter des requêtes SQL en cliquant sur le plus à droite, on voit ici que nous avons la requête 2 avec sqlite2.

Travail sur les données des régions et des départements en France.

Comment mettre cette information dans ma requête ?

Il faut déjà mettre les 2 tables Departement et Region

On peut mettre les deux requêtes en une avec UNION mais il faut mettre le même nombre de colonnes.

Voici cette requête SQL :

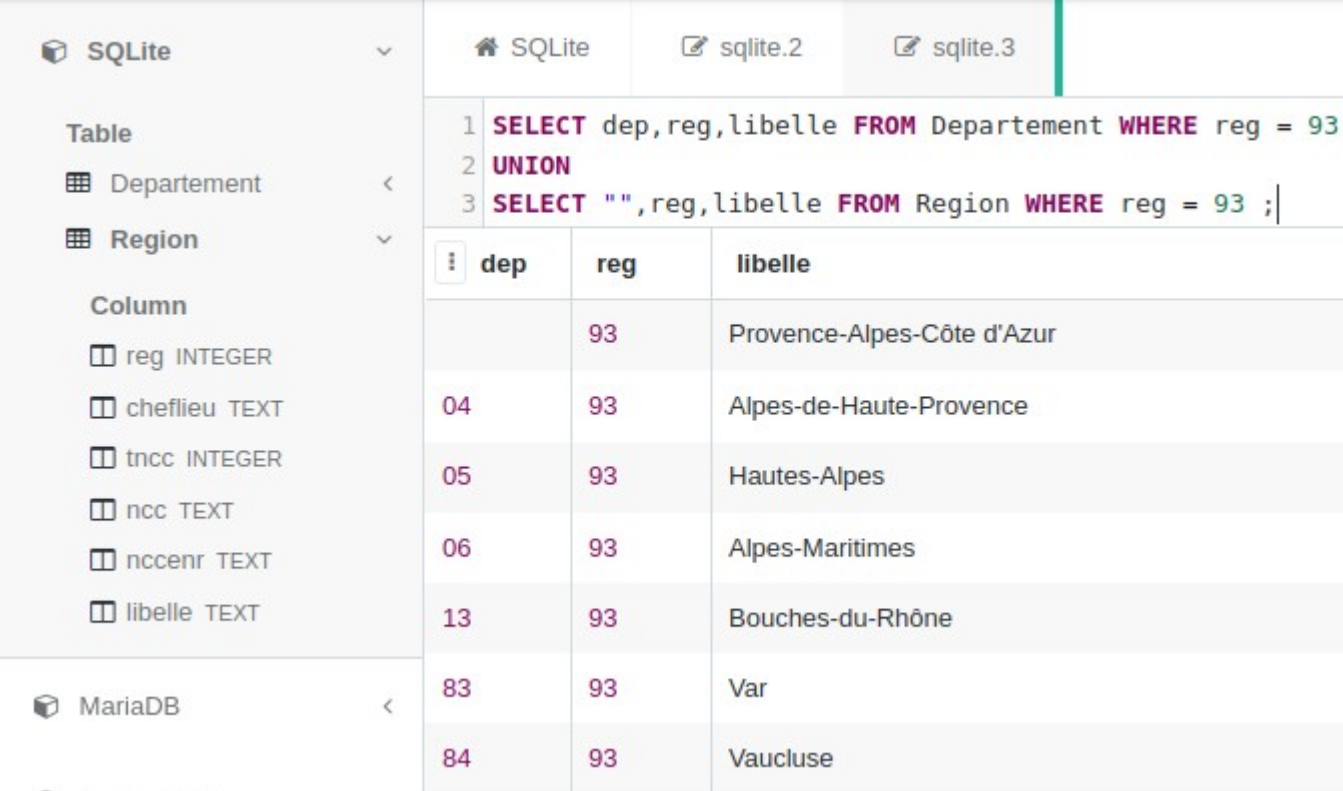
```
SELECT dep,reg,libelle FROM Departement WHERE reg=93  
UNION  
SELECT "",reg,libelle FROM Region WHERE reg=93 ;
```


Travail sur les données des régions et des départements en France.

Problème les noms des champs sont les mêmes dans les deux tables mais les champs libellé n'ont pas la même signification.

Dans notre cas Le libellé indique tantôt le nom de la région, tantôt le nom du département.

Cette requête est donc inutilisable.



The screenshot displays a database management interface with a SQLite database. The left sidebar shows the database structure with two tables: 'Departement' and 'Region'. The 'Region' table has columns: reg (INTEGER), cheflieu (TEXT), tncc (INTEGER), ncc (TEXT), nccenr (TEXT), and libelle (TEXT). The main area shows a SQL query editor with the following query:

```
1 SELECT dep,reg,libelle FROM Departement WHERE reg = 93
2 UNION
3 SELECT "",reg,libelle FROM Region WHERE reg = 93 ;
```

The query results are displayed in a table with three columns: dep, reg, and libelle. The results show the following data:

dep	reg	libelle
	93	Provence-Alpes-Côte d'Azur
04	93	Alpes-de-Haute-Provence
05	93	Hautes-Alpes
06	93	Alpes-Maritimes
13	93	Bouches-du-Rhône
83	93	Var
84	93	Vaucluse

Travail sur les données des régions et des départements en France.

Comme les deux tables ont un champ commun `reg` nous allons utiliser SQL INNER JOIN <https://sql.sh/cours/jointures/inner-join>

Intersection de 2 ensembles

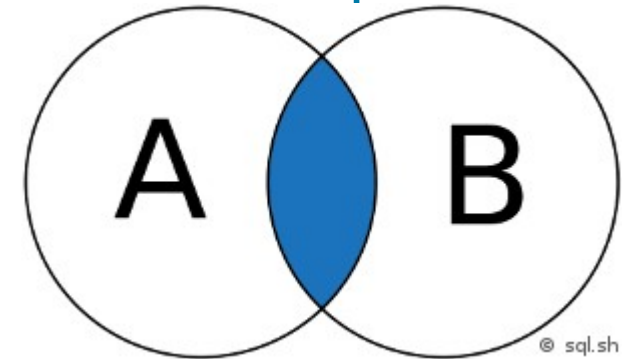
Pour utiliser ce type de jointure il convient

d'utiliser une requête SQL avec cette syntaxe :

```
SELECT * FROM table1 INNER JOIN table2 ON table1.id = table2.fk_id ;
```

On peut aussi faire cette requête :

```
SELECT * FROM table1,table2 WHERE table1.id = table2.fk_id ;
```



Travail sur les données des régions et des départements en France.

Voici la requête pour nos 2 tables :

```
SELECT * FROM Region INNER JOIN Departement ON  
Region.reg = Departement.reg ;
```

<div><div>FileOwner DBRunShareExportImport</div><div>SQLite</div><div>Table</div><div>demo</div><div>Departement</div><div>Region</div><div>MariaDB</div><div>PostgreSQL</div><div>MS SQL</div><div>Oracle</div><div>Docker</div><div>Syntax</div><div>Business</div></div>													
1 SELECT * FROM Region INNER JOIN Departement ON Region.reg = Departement.reg													
	reg	cheflieu	tncc	ncc	nccenr	libelle	dep	reg	cheflieu	tncc	ncc	nccenr	libelle
1	97105	3	GUADELOUPE	Guadeloupe	Guadeloupe	971	1	97105	3	GUADELOUPE	Guadeloupe	Guadeloupe	Guadeloupe
2	97209	3	MARTINIQUE	Martinique	Martinique	972	2	97209	3	MARTINIQUE	Martinique	Martinique	Martinique
3	97302	3	GUYANE	Guyane	Guyane	973	3	97302	3	GUYANE	Guyane	Guyane	Guyane
4	97411	0	LA REUNION	La Réunion	La Réunion	974	4	97411	0	LA REUNION	La Réunion	La Réunion	La Réunion
6	97608	0	MAYOTTE	Mayotte	Mayotte	976	6	97608	0	MAYOTTE	Mayotte	Mayotte	Mayotte
11	75056	1	ILE DE FRANCE	Île-de-France	Île-de-France	75	11	75056	0	PARIS	Paris	Paris	Paris
11	75056	1	ILE DE FRANCE	Île-de-France	Île-de-France	77	11	77288	0	SEINE ET MARNE	Seine-et-Marne	Seine-et-Marne	Seine-et-Marne
11	75056	1	ILE DE FRANCE	Île-de-France	Île-de-France	78	11	78646	4	YVELINES	Yvelines	Yvelines	Yvelines
11	75056	1	ILE DE FRANCE	Île-de-France	Île-de-France	91	11	91228	5	ESSONNE	Essonne	Essonne	Essonne
11	75056	1	ILE DE FRANCE	Île-de-France	Île-de-France	92	11	92050	4	HAUTS DE SEINE	Hauts-de-Seine	Hauts-de-Seine	Hauts-de-Seine
11	75056	1	ILE DE FRANCE	Île-de-France	Île-de-France	93	11	93008	3	SEINE SAINT DENIS	Seine-Saint-Denis	Seine-Saint-Denis	Seine-Saint-Denis
11	75056	1	ILE DE FRANCE	Île-de-France	Île-de-France	94	11	94028	2	VAL DE MARNE	Val-de-Marne	Val-de-Marne	Val-de-Marne
11	75056	1	ILE DE FRANCE	Île-de-France	Île-de-France	95	11	95500	2	VAL D OISE	Val-d'Oise	Val-d'Oise	Val-d'Oise

Travail sur les données des régions et des départements en France.

Toujours trop de colonnes et des colonnes avec le même nom, nous allons nous limiter à 4 champs :


le numéro de département le nom du département, le numéro de région et le nom de la région.

Pour afficher le nom du département : champ libelle de département nous écrirons `Departement.libelle` et pour le nom de la région : champ libelle de région nous écrirons `Region.libelle`

Voici notre requête :

```
SELECT dep,Departement.libelle,Region.reg,Region.libelle
FROM Region INNER JOIN Departement
ON Region.reg = Departement.reg
```

Travail sur les données des régions et des départements en France.



The screenshot shows a database application interface. At the top, there is a green header bar with buttons for 'File', 'Owner DB', 'Run', 'Share', 'Export', and 'Import'. Below this, a sidebar on the left lists database types: SQLite, MariaDB, PostgreSQL, and MS SQL. Under SQLite, a table list shows 'demo', 'Departement', and 'Region'. The main area displays a SQL query: `1 SELECT dep,Departement.libelle,Region.reg,Region.libelle FROM Region INNER JOIN Departement ON Region.reg = Departement.reg`. Below the query, a table of results is shown with columns 'dep', 'libelle', 'reg', and 'libelle'. The results list several French regions and their corresponding departments.

dep	libelle	reg	libelle
971	Guadeloupe	1	Guadeloupe
972	Martinique	2	Martinique
973	Guyane	3	Guyane
974	La Réunion	4	La Réunion
976	Mayotte	6	Mayotte
75	Paris	11	Île-de-France

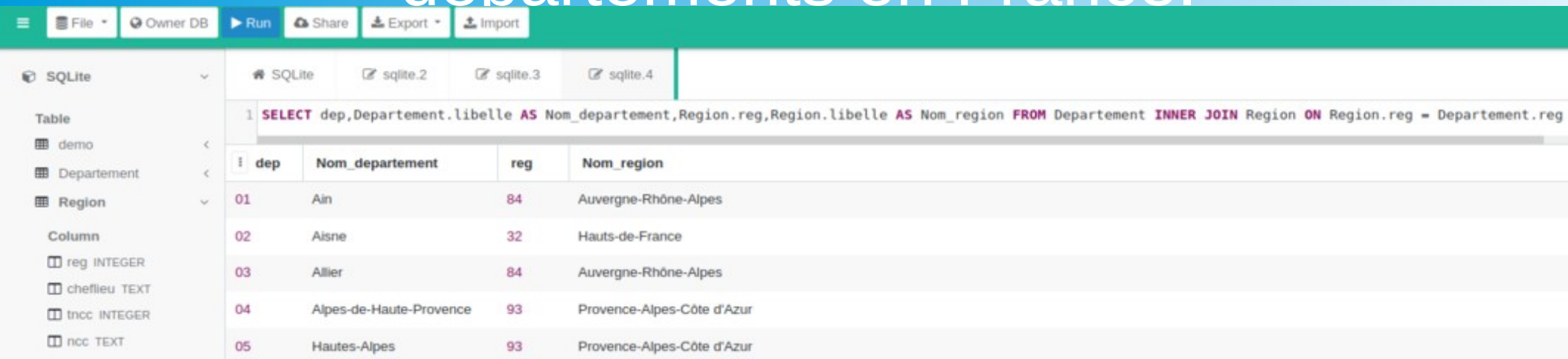
Voici notre requête :

```
SELECT dep,Departement.libelle,Region.reg,Region.libelle
FROM Region INNER JOIN Departement
ON Region.reg = Departement.reg ;
```

Il reste un problème, les champs restent les mêmes pour l'affichage.

Nous allons utiliser un alias des champs avec AS pour l'affichage.

Travail sur les données des régions et des départements en France.



The screenshot shows a SQLite database interface. The top menu bar includes 'File', 'Owner DB', 'Run', 'Share', 'Export', and 'Import'. The left sidebar shows a tree view with 'SQLite' selected, containing 'Table' (demo, Departement, Region) and 'Column' (reg INTEGER, cheflieu TEXT, tncc INTEGER, ncc TEXT). The main area displays a SQL query: `1 SELECT dep,Departement.libelle AS Nom_departement,Region.reg,Region.libelle AS Nom_region FROM Departement INNER JOIN Region ON Region.reg = Departement.reg`. Below the query, the results are shown in a table with 4 columns: dep, Nom_departement, reg, and Nom_region. The results are ordered by region name.

dep	Nom_departement	reg	Nom_region
01	Ain	84	Auvergne-Rhône-Alpes
02	Aisne	32	Hauts-de-France
03	Allier	84	Auvergne-Rhône-Alpes
04	Alpes-de-Haute-Provence	93	Provence-Alpes-Côte d'Azur
05	Hautes-Alpes	93	Provence-Alpes-Côte d'Azur

Voici la requête :

```
SELECT dep,Departement.libelle AS Nom_departement,  
Region.reg, Region.libelle AS Nom_region FROM Departement  
INNER JOIN Region ON Region.reg = Departement.reg ;
```

Si on veut trier par nom de région on ajoutera : `ORDER BY Nom_region`

Travail sur les données des régions et des départements en France.

Table		<pre>1 SELECT dep,Departement.libelle AS Nom_departement,Region.reg,Region.libelle AS Nom_region 2 FROM Departement INNER JOIN Region ON Region.reg = Departement.reg 3 ORDER BY Nom_region</pre>			
demo	<				
Departement	<				
Region	∨				
Column					
reg INTEGER					
cheflieu TEXT					
tncc INTEGER					
ncc TEXT					
nccenr TEXT					
libelle TEXT					
MariaDB	<				
PostgreSQL	<				
MS SQL	<				
Oracle	<				
Docker	<				
Syntax	<				
		dep	Nom_departement	reg	Nom_region
		01	Ain	84	Auvergne-Rhône-Alpes
		03	Allier	84	Auvergne-Rhône-Alpes
		07	Ardèche	84	Auvergne-Rhône-Alpes
		15	Cantal	84	Auvergne-Rhône-Alpes
		26	Drôme	84	Auvergne-Rhône-Alpes
		38	Isère	84	Auvergne-Rhône-Alpes
		42	Loire	84	Auvergne-Rhône-Alpes
		43	Haute-Loire	84	Auvergne-Rhône-Alpes
		63	Puy-de-Dôme	84	Auvergne-Rhône-Alpes
		69	Rhône	84	Auvergne-Rhône-Alpes
		73	Savoie	84	Auvergne-Rhône-Alpes
		74	Haute-Savoie	84	Auvergne-Rhône-Alpes
		21	Côte-d'Or	27	Bourgogne-Franche-Comté

Travail sur les données des régions et des départements en France.

Comme on le voit le tri se fait par région, on commence par les départements de la région Auvergne-Rhône-Alpes, ainsi de suite ...

Si maintenant, je ne veux afficher que les départements d'une région, je pourrais utiliser WHERE pour choisir la Bretagne par exemple.

```
SELECT dep,Departement.libelle AS Nom_departement, Region.reg,  
Region.libelle as Nom_region
```

```
FROM Departement INNER JOIN Region
```

```
ON Region.reg = Departement.reg
```

```
WHERE
```

```
  Nom_region =  
'Bretagne'
```

```
ORDER BY
```

```
Nom_region ;
```

```
1 SELECT dep,Departement.libelle AS Nom_departement,Region.reg,Region.libelle AS Nom_region  
2 FROM Departement INNER JOIN Region ON Region.reg = Departement.reg  
3 WHERE Nom_region = 'Bretagne'  
4 ORDER BY  Nom_region
```

dep	Nom_departement	reg	Nom_region
22	Côtes-d'Armor	53	Bretagne
29	Finistère	53	Bretagne
35	Ille-et-Vilaine	53	Bretagne
56	Morbihan	53	Bretagne

Travail sur les données des régions et des départements en France.

Voici comment à partir de deux tables distinctes qui ont un champ commun reg (numéro de région) nous avons pu retrouver les départements qui font partie d'une région.

Que faut-il faire si on veut afficher les départements de la région Occitanie ?

Travail sur les données des régions et des départements en France.

- Requête pour obtenir les départements de la région Occitanie

Travail sur les données des régions et des départements en France.

Nous pouvons aussi compter les départements d'une région avec Count(*) qui compte le nombre de lignes par région avec le GROUP BY Nom_region

Voici la requête :

```
SELECT Region.libelle as Nom_region,COUNT(*) AS nombre_departement  
FROM Departement INNER JOIN Region ON Region.reg = Departement.reg  
GROUP BY Nom_region
```

Travail sur les données des régions et des départements en France.

Table	
demo	<
Departement	<
Region	▼
Column	
reg INTEGER	
cheflieu TEXT	
tncc INTEGER	
ncc TEXT	
nccenr TEXT	
libelle TEXT	
MariaDB	<
PostgreSQL	<
MS SQL	<
Oracle	<
Docker	<
Syntax	<
Business	

```
1 SELECT Region.libelle AS Nom_region, COUNT(*) AS nombre_departement
2 FROM Departement INNER JOIN Region ON Region.reg = Departement.reg
3 GROUP BY Nom_region
4 --ORDER BY Nom_region
```

Nom_region	nombre_departement
Auvergne-Rhône-Alpes	12
Bourgogne-Franche-Comté	8
Bretagne	4
Centre-Val de Loire	6
Corse	2
Grand Est	10
Guadeloupe	1
Guyane	1
Hauts-de-France	5
La Réunion	1
Martinique	1
Mayotte	1
Normandie	5
Nouvelle-Aquitaine	12
Occitanie	13
Pays de la Loire	5
Provence-Alpes-Côte d'Azur	6
Île-de-France	8

Travail sur les données des régions et des départements en France.

On peut savoir combien de régions ont moins de 6 départements avec HAVING condition

Voici la requête commentée :

```
SELECT Region.libelle,Count(Departement.libelle) as 'nb_dep'  
FROM Departement  
INNER JOIN Region ON Departement.reg = Region.reg --intersection des  
2 tables Departement et Region si et seulement si les numéros de la région sont égaux  
Group By Region.reg -- regroupe tous les lignes dont le n° de région est identique  
-- l'instruction Count(Departement.libelle) compte le nombre des départements pour chacun  
des numéros de régions  
HAVING Count(Departement.libelle) < 6 -- les régions qui ont moins de 6  
départements  
ORDER BY Region.libelle;-- on trie de manière croissante avec le nom de la région
```

Travail sur les données des régions et des départements en France.

```
1 SELECT Region.libelle, COUNT(Departement.libelle) AS 'nb_dep'
2 FROM Departement
3 INNER JOIN Region ON Departement.reg = Region.reg --intersection des 2 tables Departement et Region si et seulement si les numéros de la région sont égaux
4 GROUP BY Region.reg -- on regroupe tous les lignes dont le numéro de région est identique
5 -- l'instruction Count(Departement.libelle) compte le nombre de départements pour chacun des numéros de régions
6 HAVING COUNT(Departement.libelle) < 6 -- les régions qui ont moins de 6 départements
7 ORDER BY Region.libelle;-- on trie de manière croissante avec le nom de la région
```

libelle	nb_dep
Bretagne	4
Corse	2
Guadeloupe	1
Guyane	1
Hauts-de-France	5
La Réunion	1
Martinique	1
Mayotte	1
Normandie	5
Pays de la Loire	5

Travail sur les données des régions et des départements en France.

Pour finir voici une requête plus complexe avec 2 SELECT :

```
SELECT Departement.libelle, Departement.dep, Departement.reg,  
Region.libelle, nb_dep
```

```
FROM (SELECT COUNT(Departement.dep) AS 'nb_dep',  
Region.libelle AS 'lib' FROM Departement INNER JOIN Region ON  
Departement.reg = Region.reg GROUP BY Region.libelle) AS  
'count_dep'
```

```
INNER Join (Departement INNER JOIN Region ON  
Departement.reg = Region.reg)
```

```
ON Region.libelle = count_dep.lib
```

```
ORDER BY Region.libelle ;
```

Travail sur les données des régions et des départements en France.

```
1 SELECT Departement.libelle ,Departement.dep,Departement.reg,Region.libelle, nb_dep
2 FROM
3     (SELECT COUNT(Departement.dep) AS 'nb_dep', Region.libelle AS 'lib' FROM Departement INNER JOIN Region ON Departement.reg = Region.reg GROUP BY Region.libelle) AS 'count_dep'
4 INNER JOIN (Departement INNER JOIN Region ON Departement.reg = Region.reg)
5 ON Region.libelle = count_dep.lib
6 ORDER BY Region.libelle ;
```

libelle	dep	reg	libelle:1	nb_dep
Ain	01	84	Auvergne-Rhône-Alpes	12
Allier	03	84	Auvergne-Rhône-Alpes	12
Ardèche	07	84	Auvergne-Rhône-Alpes	12
Cantal	15	84	Auvergne-Rhône-Alpes	12
Drôme	26	84	Auvergne-Rhône-Alpes	12
Isère	38	84	Auvergne-Rhône-Alpes	12
Loire	42	84	Auvergne-Rhône-Alpes	12
Haute-Loire	43	84	Auvergne-Rhône-Alpes	12
Puy-de-Dôme	63	84	Auvergne-Rhône-Alpes	12
Rhône	69	84	Auvergne-Rhône-Alpes	12
Savoie	73	84	Auvergne-Rhône-Alpes	12
Haute-Savoie	74	84	Auvergne-Rhône-Alpes	12
Côte-d'Or	21	27	Bourgogne-Franche-Comté	8