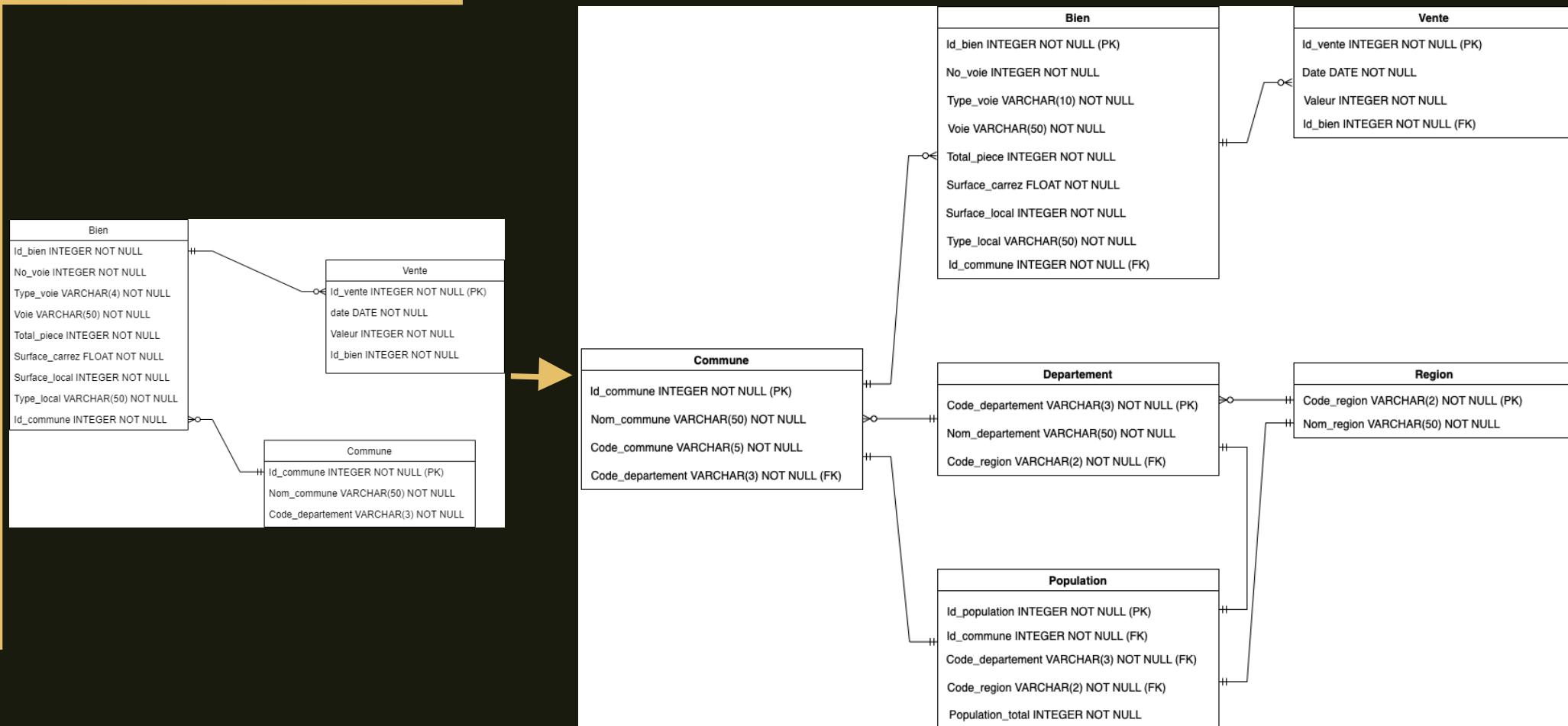


CRÉATION DE BASE DE DONNÉES



Laplace Immo

SCHÉMA UML



DONNÉES IMMOBILIERS

The image shows two screenshots of Microsoft Excel spreadsheets titled "DAN-P3-data.xlsx".

Top Sheet (A1-R10): This sheet contains data for property sales. The columns include: Code service CH, Reference document, 1 Articles CGI, 2 Articles CGI, 3 Articles CGI, 4 Articles CGI, 5 Articles CGI, No disposition, Date mutation, Nature mutation, Valeur fonciere, No voie, B/T/Q, Code type de voie, Type de voie, Code voie, Voie, and Code ID commun. The data spans from row 1 to 10.

Code service CH	Reference document	1 Articles CGI	2 Articles CGI	3 Articles CGI	4 Articles CGI	5 Articles CGI	No disposition	Date mutation	Nature mutation	Valeur fonciere	No voie	B/T/Q	Code type de voie	Type de voie	Code voie	Voie	Code ID commun
1							1	2020-02-03 00:00:00	Vente	56000	190 A	0	RUE	5	CENTRALE		
2							1	2020-01-02 00:00:00	Vente	165000	347	0	RUE	20	DU CHATEAU		
3							1	2020-01-08 00:00:00	Vente	720000	58	1	AV	527	DU MONT BLANC		
4							1	2020-01-06 00:00:00	Vente	429250	140	0	RUE	2	DE L'ABBE JOLIVET		
5							1	2020-01-07 00:00:00	Vente	220900	39	0	RUE	110	BUFFON		
6							1	2020-01-21 00:00:00	Vente	42000	28	1	AV	179	JEAN FALCONNIER		
7							1	2020-01-07 00:00:00	Vente	262000	8	0	RUE	120	DE GENEVE		
8							1	2020-01-08 00:00:00	Vente	190000	2	0	RUE	210	DU RECULET		
9							1	2020-01-16 00:00:00	Vente	563130	1403	0	RUE	465	JEAN DE GINGINS		
10																	

Bottom Sheet (S1-AH10): This sheet contains data for communes. The columns include: Code postal, Commune, Code département, Code commune, Prefixe de section, Section, No plan, No Volume, 1er lot, Surface Carréz du 1er lot, 2eme lot, Surface Carréz du 2eme lot, 3eme lot, Surface Carréz du 3eme lot, 4eme lot, Surface Carréz du 4eme lot, and Surface Carréz du 5eme lot. The data spans from row 1 to 10.

S	T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	
Code postal	Commune	Code département	Code commune	Prefixe de section	Section	No plan	No Volume	1er lot	Surface Carréz du 1er lot	2eme lot	Surface Carréz du 2eme lot	3eme lot	Surface Carréz du 3eme lot	4eme lot	Surface Carréz du 4eme lot	Surface Carréz du 5eme lot
2 1370.0	SAINT-ETIENNE-DU-BOIS	1	350		B	1478	22		50,42							
3 1170.0	CHEVRY	1	103		A	302	12		48,22							
4 1220.0	DIVONNE-LES-BAINS	1	143		AK	563	146		130,8							
5 1630.0	PERON	1	288		C	2307	11		109,22							
6 1200.0	VALSERHONE	1	33		AE	440	31		108,65							
7 1350.0	CULOZ	1	138		AP	357	50		31,65							
8 1630.0	ST-GENIS-POUILLY	1	354		BH	79	11		52,58							
9 1630.0	ST-GENIS-POUILLY	1	354		AX	8	304		58,71							
10 1220.0	DIVONNE-LES-BAINS	1	143		H	442	14		93,23							

DONNÉES DÉMOGRAPHIQUES

donnees_communes

CODREG	REG	CODDEP	CODARR	CODCAN	CODCOM	COM	PMUN	PCAP	PTOT
84	Auvergne-Rhône-Alpes	1	2	8	1	L'Abergement-Clémenciat	779	19	798
84	Auvergne-Rhône-Alpes	1	1	1	2	L'Abergement-de-Varey	256	1	257
84	Auvergne-Rhône-Alpes	1	1	1	4	Ambérieu-en-Bugey	14134	380	14514

donnees_departements

CODREG	REG	CODDEP	DEP	NBARR	NBCAN	NBCOM	PMUN	PTOT
84	Auvergne-Rhône-Alpes	1	Ain	4	23	393	652432	668565
32	Hauts-de-France	2	Aisne	5	21	800	531345	543368
84	Auvergne-Rhône-Alpes	3	Allier	3	19	317	335975	344922

donnees_regions

CODREG	REG	NBARR	NBCAN	NBCOM	PMUN	PTOT
1	Guadeloupe	2	21	32	384239	389995
2	Martinique	4	NA	34	364508	369406
3	Guyane	2	NA	22	281678	284085

TABLE PRINCIPALE

The screenshot shows the SQLiteStudio interface with the database 'DATAImmo (SQLite 3)' selected. The left sidebar displays the schema, including tables like Bien, Commune, Departement, Population, Region, Vente, and Z_MasterTable, along with their columns. The SQL editor window at the top contains a query to select all rows from the Z_MasterTable. Below the editor is a grid view of the data, showing 34169 rows. The columns include Date, Valeur, No_voie, Type_voie, Voie, Total_piece, Surface_carrez, Surface_local, Type_local, Code_commune, Nom_commune, and Code_departement. The data shows various street names and locations across different dates and values. The bottom status bar indicates query execution times.

	Date	Valeur	No_voie	Type_voie	Voie	Total_piece	Surface_carrez	Surface_local	Type_local	Code_cc	Nom_commune
1	2020-02-03 00:00:00	56000	190	RUE	CENTRALE	2	50,42	52	Appartement	350	SAINT-ETIENNE-DU-BOIS
2	2020-01-02 00:00:00	165000	347	RUE	DU CHATEAU	3	48,22	48	Appartement	103	CHEVRY
3	2020-01-08 00:00:00	720000	58	AV	DU MONT BLANC	6	130,8	130	Appartement	143	DIVONNE-LES-BAINS
4	2020-01-06 00:00:00	429250	140	RUE	DE L'ABBE JOLIVET	5	109,22	109	Maison	288	PERON
5	2020-01-07 00:00:00	220900	39	RUE	BUFFON	4	108,65	91	Appartement	33	VALSERHONE
6	2020-01-21 00:00:00	42000	28	AV	JEAN FALCONNIER	2	31,65	32	Appartement	138	CULOZ
7	2020-01-07 00:00:00	262000	8	RUE	DE GENEVE	2	52,58	52	Appartement	354	ST-GENIS-POUILLY
8	2020-01-08 00:00:00	190000	2	RUE	DU RECULET	2	58,71	60	Appartement	354	ST-GENIS-POUILLY
9	2020-01-16 00:00:00	563130	1403	RUE	JEAN DE GINGINS	4	93,23	96	Maison	143	DIVONNE-LES-BAINS
10	2020-01-17 00:00:00	535000	226	ALL	DES CAPUCINES	5	117	117	Maison	354	ST-GENIS-POUILLY
11	2020-01-16 00:00:00	330000	276	RTE	DE POUGNY	2	35,6	36	Appartement	288	PERON
12	2020-01-27 00:00:00	110600	79	CRS	DE VERDUN	5	138,03	137	Appartement	283	OYONNAX

Status

- [20:46:24] Query finished in 0.003 second(s).
- [20:47:10] Query finished in 0.001 second(s).
- [20:48:21] Query finished in 0.003 second(s).

TABLE COMMUNE

The screenshot shows a database management interface with the following details:

- Left Panel (Object Explorer):** Shows the database structure for "DATAImmo (SQLite 3)".
 - Tables:** Bien, Commune (selected), Departement, Population, Region, Vente, Z_MasterTable.
 - Commune Table Details:** Contains 4 columns: Id_commune, Nom_commune, Code_commune, Code_departement.
 - Columns:** Id_commune, Nom_commune, Code_commune, Code_departement.
 - Indexes and Triggers:** Indexes, Triggers.
- SQL Editor (Top Right):** Displays the SQL code for creating the Commune table and a select query.

```
205  
206  
207  
208 --- Création de table Commune  
209  
210 CREATE TABLE Commune AS  
211 SELECT Code_commune, Nom_commune, Code_departement  
212 FROM Z_MasterTable  
213 GROUP BY Code_departement,Code_commune  
214 ORDER BY Nom_commune;  
215  
216  
217 SELECT * FROM Commune;  
218  
219  
220
```
- Result Grid View (Bottom):** Shows the first 13 rows of the Commune table.

	Id_commune	Nom_commune	Code_commune	Code_departement
1		ABBEVILLE	1	80
2		ABLON-SUR-SEINE	1	94
3		ABRIES-RISTOLAS	1	5
4		ACHERES	5	78
5		ACHERES-LA-FORET	1	77
6		ACHICOURT	4	62
7		ACIGNE	1	35
8		ADE	2	65
9		AGDE	3	34
10		AGEN	1	47
11		AGNEAUX	2	50
12		AGON-COUTAINVILLE	3	50
13		AIGREFEUILLE SUR MAINE	2	44
- Status Bar (Bottom):** Shows three log entries indicating query completion times.

TABLE BIEN

The screenshot shows a database management interface with the following details:

- Left Panel (Object Explorer):** Shows the database structure under "DATAImmo (SQLite 3)".
 - Tables:** Bien (selected), Commune, Departement, Population, Region, Vente, Z_MasterTable.
 - Bien Details:** Columns (9) include Id_bien, No_voie, Type_voie, Voie, Total_piece, Surface_carrez, Surface_local, Type_local, Id_commune.
 - Views:** None.
- Top Bar:** Includes icons for file operations (New, Open, Save, Print, etc.) and a toolbar with various tools.
- SQL Editor:** Titled "SQL editor 1". It contains the following SQL code:

```
220 --- Création de table Bien
221 CREATE TABLE Bien AS
222 SELECT No_voie, Type_voie, Voie, Total_piece, Surface_carrez, Surface_local, Type_local, Id_commune
223 FROM Z_MasterTable z
224 LEFT JOIN Commune c
225 ON (z.Code_commune = c.Code_commune)
226 AND (z.Code_departement = c.Code_departement)
227 GROUP BY id_commune, Voie, No_voie
228 ORDER BY Voie, Type_voie, No_voie;
229
230 SELECT * FROM Bien;
```
- Data Grid:** Shows the data from the Bien table in "Grid view".

	Id_bien	No_voie	Type_voie	Voie	Total_piece	Surface_carrez	Surface_local	Type_local	Id_commune
1	1	5	IMP	10 RUE DE METZ	1	31	31	Appartement	2851
2	2	164	CC	119 DIT DU VIEUX REYNIER	3	53,77	55	Appartement	1302
3	3	289	CC	119 DIT DU VIEUX REYNIER	3	64,54	64	Appartement	1302
4	4	325	CC	119 DIT DU VIEUX REYNIER	2	34,3	34	Appartement	1302
5	5	345	CC	119 DIT DU VIEUX REYNIER	3	59,57	59	Appartement	1302
6	6	402	CC	119 DIT DU VIEUX REYNIER	3	71,15	75	Appartement	1302
7	7	567	CC	119 DIT DU VIEUX REYNIER	4	97	80	Appartement	1302
8	8	129	RES	129 RUE LAVOISIER	4	61,19	59	Appartement	1156
9	9	9	RUE	12E REGIMENT DE CHASSEURS	2	48,98	48	Appartement	2662
10	10	736	CD	135	4	115,11	100	Maison	2920
11	11	1	RUE	16E ET 22E DRAGONS	2	57,64	58	Appartement	2209
12	12	4	PL	1907-06-01 00:00:00	3	45,81	47	Appartement	167
13	13	12	RUE	1918-11-11 00:00:00	4	85,01	84	Appartement	1168
- Status Bar:** Shows three log entries indicating query completion times.

TABLE VENTE

The screenshot shows a database management interface with the following details:

- Left Panel (Object Explorer):** Shows the database structure under "DATAImmo (SQLite 3)".
 - Tables:** Bien, Commune, Departement, Population, Region, Vente.
 - Vente Details:** Contains 4 columns: Id_vente, Date, Valeur, Id_bien.
 - Z_MasterTable:** A linked table.
- Top Bar:** Includes standard icons for file operations, database management, and toolbars.
- SQL Editor 1 (Query Tab):** Displays the SQL code for creating the Vente table and inserting data from Z_MasterTable.

```
237 --- Création de table Vente
238
239 CREATE TABLE Vente AS
240 SELECT
241     Date, Valeur, Id_bien
242 FROM Z_MasterTable z
243 LEFT JOIN
244     (SELECT * FROM Bien b
245     LEFT JOIN Commune c
246     ON b.Id_commune = c.Id_commune) AS a
247 ON
248 (z.No_voie = a.No_voie) AND (z.Voie = a.Voie) AND (z.Code_commune = a.Code_commune);
249
250
251 SELECT * FROM Vente;
252
```
- Data Grid View:** Shows the first 13 rows of the Vente table.

	Id_vente	Date	Valeur	Id_bien
1	1	2020-02-03 00:00:00	56000	2496
2	2	2020-01-02 00:00:00	165000	14932
3	3	2020-01-08 00:00:00	720000	16799
4	4	2020-01-06 00:00:00	429250	6094
5	5	2020-01-07 00:00:00	220900	2076
6	6	2020-01-21 00:00:00	42000	21821
7	7	2020-01-07 00:00:00	262000	5285
8	8	2020-01-08 00:00:00	190000	17585
9	9	2020-01-16 00:00:00	563130	21782
10	10	2020-01-17 00:00:00	535000	11539
11	11	2020-01-16 00:00:00	330000	9832
12	12	2020-01-27 00:00:00	110600	10692
13	13	2020-01-30 00:00:00	50000	15189
- Status Bar:** Shows query execution times.
 - [20:48:55] Query finished in 0.001 second(s).
 - [20:49:31] Query finished in 0.002 second(s).
 - [20:51:04] Query finished in 0.002 second(s).

TABLE REGION

The screenshot shows a database management interface with the following details:

- Left Panel (Object Explorer):** Shows the database structure for "DATAImmo (SQLite 3)".
 - Tables:** Bien, Commune, Departement, Population, Region.
 - Region Table Details:** Contains 2 columns: Code_region and Nom_region.
 - Views:** Vente, Z_MasterTable.
- Top Bar:** Includes various icons for database operations like Create, Drop, Refresh, and Tools.
- Toolbar:** Includes icons for Database, Table, Column, View, Index, Trigger, and others.
- SQL Editor:** Labeled "SQL editor 1". It contains the following SQL code:

```
266  
267  
268  
269 --- Création de table Region  
270  
271 SELECT * FROM Region;
```
- Data Grid:** Shows the results of the query in "Grid view".

	Code_region	Nom_region
1	01	Guadeloupe
2	02	Martinique
3	03	Guyane
4	04	La Réunion
5	11	Île-de-France
6	24	Centre-Val de Loire
7	27	Bourgogne-Franche-Comté
8	28	Normandie
9	32	Hauts-de-France
10	44	Grand Est
11	52	Pays de la Loire
12	53	Bretagne
13	75	Nouvelle-Aquitaine

Total rows loaded: 17
- Status Bar:** Shows three log entries indicating query completion times.
 - [20:51:04] Query finished in 0.002 second(s).
 - [20:51:44] Query finished in 0.003 second(s).
 - [20:52:25] Query finished in 0.001 second(s).

TABLE DEPARTEMENT

The screenshot shows a database management interface with the following details:

- Left Panel (Object Browser):** Shows the **Databases** section with **DATAImmo (SQLite 3)** selected. Under **Tables (7)**, the **Departement** table is expanded, showing its **Columns (3)**: **Code_departement**, **Nom_departement**, and **Code_region**. Other tables listed include **Bien**, **Commune**, **Population**, **Region**, **Vente**, and **Z_MasterTable**.
- Top Bar:** Contains various icons for database management, including a magnifying glass, a key, and a gear.
- SQL Editor:** Titled "SQL editor 1", it displays the following SQL code:

```
276  
277  
278  
279  
280  
281  
282 --> Cr eation de table Departement  
283  
284 SELECT * FROM Departement;
```
- Table View:** A grid view showing the data from the **Departement** table. The columns are **Code_departement**, **Nom_departement**, and **Code_region**. The data is as follows:

	Code_departement	Nom_departement	Code_region
1	01	Ain	84
2	02	Aisne	32
3	03	Allier	84
4	04	Alpes-de-Haute-Provence	93
5	05	Hautes-Alpes	93
6	06	Alpes-Maritimes	93
7	07	Ard�che	84
8	08	Ardennes	44
9	09	Ari�ge	76
10	10	Aube	44
11	11	Aude	76
12	12	Aveyron	76
13	13	Bouches-du-Rh�ne	93

- Status Bar:** Shows three recent query execution times: [20:51:44] Query finished in 0.003 second(s), [20:52:25] Query finished in 0.001 second(s), and [20:53:13] Query finished in 0.001 second(s).
- Bottom Bar:** Shows the title "SQL editor 1" again.

TABLE POPULATION

The screenshot shows a database management interface with the following details:

- Left Panel (Databases):** Shows the **DATAImmo (SQLite 3)** database with its tables: **Bien**, **Commune**, **Département**, **Population**, and their respective columns, indexes, triggers, regions, and views.
- Top Bar:** Includes standard icons for file operations, database management, and toolbars.
- Toolbar:** Includes icons for creating, deleting, and modifying tables, as well as various analysis and export tools.
- SQL Editor:** Labeled **SQL editor 1**, showing the creation of the **Population** table and a query to select all rows from it.

```
253  
254  
255  
256  
257 --- Création de table Population  
258  
259 SELECT * FROM Population;  
260  
261  
262  
263  
264  
265  
266  
267  
268
```
- Table View:** A grid view showing the first 13 rows of the **Population** table. The columns are **Id_population**, **Id_commune**, **Code_departement**, **Code_region**, and **Population_total**. The data includes commune codes like 1_01, 2_01, etc., and population totals ranging from 118 to 14514.
- Status Bar:** Displays three status messages indicating query completion times: [20:52:25] Query finished in 0.001 second(s), [20:53:13] Query finished in 0.001 second(s), and [20:59:59] Query finished in 0.002 second(s).

NOMBRE TOTAL DE VENTES

Requête 1 : Nombre total d'appartements vendus au 1^{er} semestre 2020

The screenshot shows a SQLite database interface with the following details:

- Databases:** DATAImmo (SQLite 3) is selected.
- Tables:** Bien, Commune, Departement, Loc, Population, Region, and Vente are listed.
- SQL editor 1:** The query is:129 --- Nombre total d'appartements vendus au 1er semestre 2020
130
131
132 SELECT SUM(Id_vente) FROM Vente
133 JOIN Bien
134 ON Vente.Id_bien = Bien.Id_bien
135 WHERE Type_local = "Appartement";
136
137
138
139
140
141
142
143
- Results:** The result table shows 1 row with the value 543449273.

NOMBRE DE VENTES PAR RÉGION

Requête 2 : Nombre de ventes d'appartement par région pour le 1^{er} semestre 2020

The screenshot shows a database management interface with a sidebar containing a tree view of tables and their columns from a SQLite database named 'DATAImmo'. The tables include 'Bien', 'Loc', 'Vente', and others. The main area is a SQL editor window titled 'SQL editor 1' with the tab 'Query' selected. The query itself is a multi-step SELECT statement designed to calculate the number of apartment sales per region for the first half of 2020. It involves joining four tables: 'Commune', 'Departement', 'Region', and 'Vente', and then further joining 'Bien' to filter for apartment sales ('Type_local = "Appartement"'). The final step groups the results by region code and name, and counts the number of sales ('Nombre_vente'). The results are displayed in a grid view below the query.

```
143 --- Nombre de ventes d'appartement par région pour le 1er semestre 2020
144
145 CREATE TABLE Loc AS
146 SELECT Id_commune, Nom_commune, Code_commune, c.Code_departement, Nom_departement, Code_region, Nom_region
147 FROM Commune c
148 JOIN
149 (SELECT * FROM Departement d JOIN Region r ON d.Code_region = r.Code_region) AS dr
150 ON dr.Code_departement = c.Code_departement;
151
152 SELECT * FROM Loc;
153
154 SELECT Code_region, Nom_region, count(Id_vente) as Nombre_vente FROM Loc l
155 JOIN
156 (SELECT * FROM Bien b JOIN Vente v ON b.Id_bien = v.Id_bien WHERE b.Type_local = "Appartement") AS bv
157 ON l.Id_commune = bv.Id_commune
158 GROUP BY Nom_region
```

	Code_region	Nom_region	Nombre_vente
1	84	Auvergne-Rhône-Alpes	3252
2	27	Bourgogne-Franche-Comté	376
3	53	Bretagne	986
4	24	Centre-Val de Loire	697
5	44	Grand Est	984
6	01	Guadeloupe	2
7	03	Guyane	34
8	32	Hauts-de-France	1476
9	04	La Réunion	44
10	02	Martinique	94
11	28	Normandie	861
12	75	Nouvelle-Aquitaine	1929
13	76	Occitanie	1642
14	52	Pays de la Loire	1358
15	93	Provence-Alpes-Côte d'Azur	3652
16	11	Île-de-France	14005

VENTES PAR NOMBRE DE PIÈCES

Requête 3 : Proportion des ventes d'appartements par le nombre de pièces

The screenshot shows a SQLite database interface with the following details:

- Databases:** DATAImm (SQLite 3) is selected.
- Tables:** Bien, Commune, Departement, Loc, Population, Region, Vente.
- Bien Columns:** Id_bien, No_voie, Type_voie, Voie, Total_piece, Surface_carrez, Surface_local, Type_local, Id_commune.
- Vente Columns:** Id_vente, Date, Valeur, Id_bien.

SQL editor 1:

```
316
317
318 --- Proportion des ventes d'appartements par le nombre de pièces
319
320 SELECT Total_piece, count(Id_vente) as Nombre_vente, round(count(Id_vente)*100.0/(SELECT count(Id_vente) FROM Vente),2) AS Proportion_vente
321 FROM Bien b
322 JOIN Vente v
323 ON b.Id_bien = v.Id_bien
324 WHERE b.Type_local = "Appartement"
325 GROUP BY Total_piece;
326
```

Results: Grid view showing the proportion of sales by number of pieces.

	Total_piece	Nombre_vente	Proportion_vente
1	0	28	0.08
2	1	6720	19.66
3	2	9810	28.7
4	3	9011	26.36
5	4	4439	12.99
6	5	1095	3.2
7	6	206	0.6
8	7	54	0.16
9	8	18	0.05
10	9	8	0.02
11	10	2	0.01
12	11	1	0

DÉPARTEMENTS – PRIX ÉLEVÉ

Requête 4 : Liste des 10 départements où le prix du mètre carré est le plus élevé

The screenshot shows a SQLite database interface with the following details:

- Left Panel (Schema):** Shows the database structure with tables: DATAImmo (SQLite 3), Tables (8), Bien (9 columns: Id_bien, No_voie, Type_voie, Voie, Total_piece, Surface_carrez, Surface_local, Type_local, Id_commune), Loc (7 columns: Id_commune, Nom_commune, Code_commune, Code_departement, Nom_departement, Code_region, Nom_region), and Vente (4 columns: Id_vente, Date, Valeur, Id_bien).
- Top Bar:** Shows the database name "DATAImmo" and various toolbar icons.
- SQL Editor:** Contains the following SQL query:

```
186
187 --- Liste des 10 départements où le prix du mètre carré est le plus élevé
188
189 SELECT Nom_departement, Valeur/Surface_carrez AS Prix_metre_carre FROM Loc
190 JOIN (SELECT * FROM Vente JOIN Bien ON Vente.Id_bien = Bien.Id_bien) as BV
191 ON Loc.Id_commune = BV.Id_commune
192 GROUP BY Nom_departement
193 ORDER BY Prix_metre_carre DESC LIMIT 10;
```
- Result Grid:** Displays the top 10 departments with the highest price per square meter.

	Nom_departement	Prix_metre_carre
1	Val-de-Marne	14285
2	Paris	13600
3	Yvelines	7454
4	Ille-et-Vilaine	6428
5	Alpes-Maritimes	6231
6	Nord	5944
7	Savoie	5140
8	Gard	5000
9	Haute-Garonne	4665
10	Calvados	4264

PRIX MOYEN

Requête 5 : Prix moyen du mètre carré d'une maison en Île-de-France

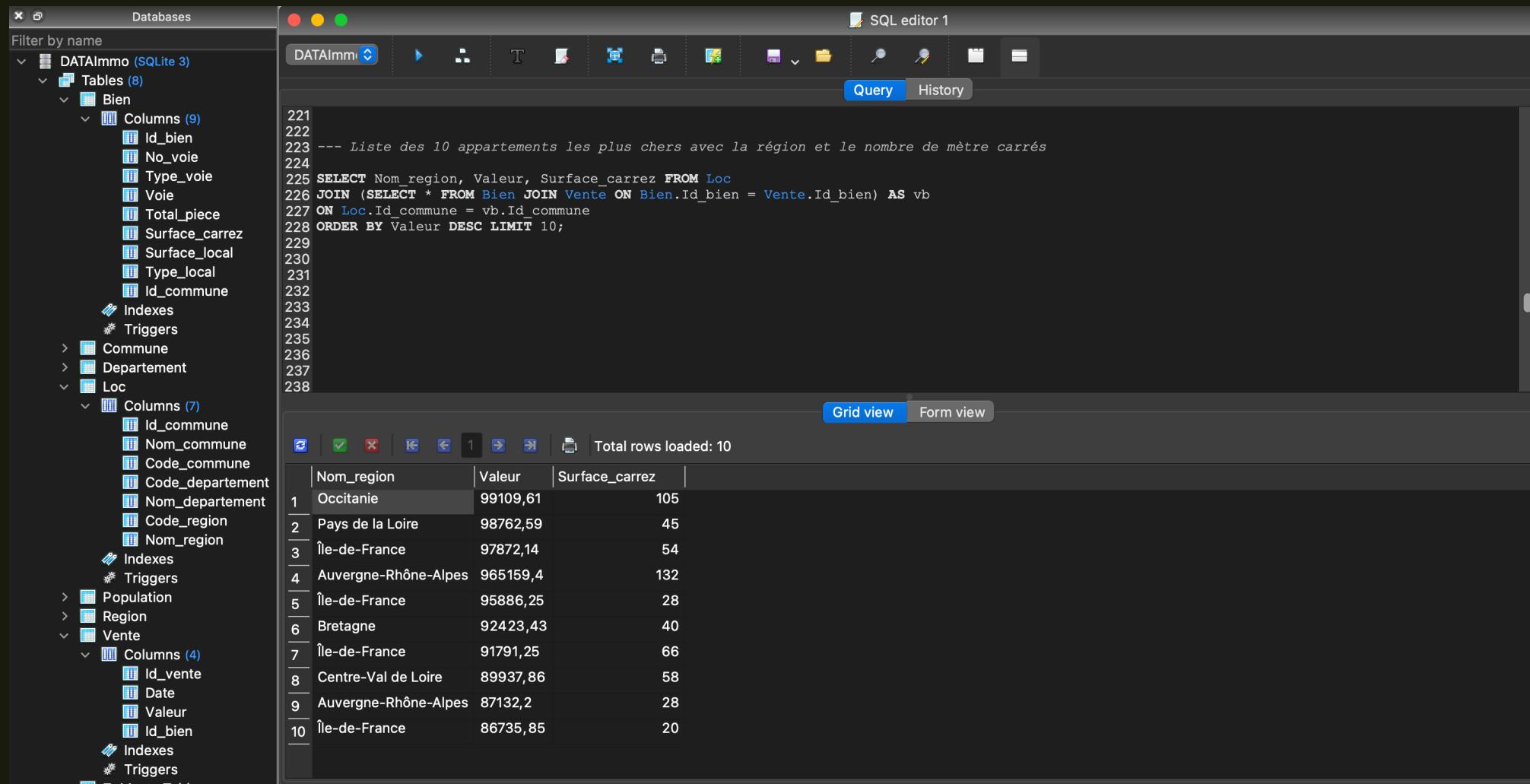
The screenshot shows a SQLite database interface with the following details:

- Databases:** DATAImmobilier (SQLite 3) is selected.
- Tables:** There are 8 tables: Bien, Loc, Commune, Departement, Population, Region, and Vente.
- Bien Table:** Contains columns: Id_bien, No_voie, Type_voie, Voie, Total_piece, Surface_carrez, Surface_local, Type_local, Id_commune.
- Loc Table:** Contains columns: Id_commune, Nom_commune, Code_commune, Code_departement, Nom_departement, Code_region, Nom_region.
- Vente Table:** Contains columns: Id_vente, Date, Valeur, Id_bien.
- SQL Editor:** Displays the following query:

```
206
207 --- Prix moyen du mètre carré d'une maison en Île-de-France
208
209 SELECT Nom_region, ROUND(AVG(Valeur/Surface_carrez),2) AS Prix_moyen_metre_carre FROM Loc
210 JOIN (SELECT * FROM Vente JOIN Bien ON Vente.Id_bien = Bien.Id_bien WHERE Type_local = "Maison") as BV
211 ON Loc.Id_commune = BV.Id_commune
212 WHERE Nom_region = "Île-de-France";
213
214
215 SELECT Nom_region, ROUND(AVG(Valeur/Surface_carrez),2) AS Prix_moyen_metre_carre FROM Loc
216 JOIN (SELECT * FROM Vente JOIN Bien ON Vente.Id_bien = Bien.Id_bien WHERE Type_local = "Maison") as BV
217 ON Loc.Id_commune = BV.Id_commune
218 WHERE Code_region = 11;
219
220
221
222
223
```
- Results Grid:** Shows the output of the query. Total rows loaded: 1. The result is: Île-de-France | 3787.09

APPARTEMENTS LES PLUS CHERS

Requête 6 : Liste des 10 appartements les plus chers avec la région et le nombre de mètre carrés



The screenshot shows a SQLite database interface with the following details:

- Databases:** DATAImmo (SQLite 3) is selected.
- Tables:** Bien, Commune, Departement, Loc, Population, Region, Vente.
- Bien Columns:** Id_bien, No_voie, Type_voie, Voie, Total_piece, Surface_carrez, Surface_local, Type_local, Id_commune.
- Loc Columns:** Id_commune, Nom_commune, Code_commune, Code_departement, Nom_departement, Code_region, Nom_region.
- Vente Columns:** Id_vente, Date, Valeur, Id_bien.

SQL editor 1:

```
221
222
223 --- Liste des 10 appartements les plus chers avec la région et le nombre de mètre carrés
224
225 SELECT Nom_region, Valeur, Surface_carrez FROM Loc
226 JOIN (SELECT * FROM Bien JOIN Vente ON Bien.Id_bien = Vente.Id_bien) AS vb
227 ON Loc.Id_commune = vb.Id_commune
228 ORDER BY Valeur DESC LIMIT 10;
```

Results Grid:

	Nom_region	Valeur	Surface_carrez
1	Occitanie	99109,61	105
2	Pays de la Loire	98762,59	45
3	Île-de-France	97872,14	54
4	Auvergne-Rhône-Alpes	965159,4	132
5	Île-de-France	95886,25	28
6	Bretagne	92423,43	40
7	Île-de-France	91791,25	66
8	Centre-Val de Loire	89937,86	58
9	Auvergne-Rhône-Alpes	87132,2	28
10	Île-de-France	86735,85	20

ÉVOLUTION DE VENTES

Requête 7 : Taux d'évolution du nombre de ventes entre le premier et le second trimestre de 2020

The screenshot shows a SQLite database interface with the following details:

- Databases:** DATAImmo (SQLite 3) is selected.
- Tables:** Tables listed include Bien, Commune, Departement, Loc, Population, Region, and Vente. The Vente table has 4 columns: Id_vente, Date, Valeur, and Id_bien.
- SQL editor 1:** The query is as follows:

```
243 --- Taux d'évolution du nombre de ventes entre le premier et le second trimestre de 2020
244
245
246 SELECT Trimestre1, Trimestre2, ROUND((CAST(Trimestre2 AS FLOAT)/CAST(Trimestre1 AS FLOAT)-1)*100,2) AS Taux_evolution FROM
247 (SELECT COUNT(*) AS Trimestre1 FROM Vente
248 WHERE Date BETWEEN "2020-01-01" AND "2020-04-01")
249 JOIN
250 (SELECT COUNT(*) AS Trimestre2 FROM Vente
251 WHERE Date BETWEEN "2020-04-01" AND "2020-07-01");
252
253
254
255
256
257
```
- Results:** The results are displayed in a grid view, showing one row with the following values:

Trimestre1	Trimestre2	Taux_evolution
16782	17397	3.66

CLASSEMENT DES RÉGIONS

Requête 8 : Classement des régions par rapport au prix au mètre carré des appartement de plus de 4 pièces

The screenshot shows a SQLite database interface with the following details:

- Databases:** DATAImmo (SQLite 3)
- Tables:** Bien, Commune, Departement, Loc, Population, Region, Vente
- Bien Columns:** Id_bien, No_voie, Type_voie, Voie, Total_piece, Surface_carrez, Surface_local, Type_local, Id_commune
- Loc Columns:** Id_commune, Nom_commune, Code_commune, Code_departement, Nom_departement, Code_region, Nom_region
- Vente Columns:** Id_vente, Date, Valeur, Id_bien

SQL editor 1:

```
252
253
254 --- Classement des régions par rapport au prix au mètre carré des appartement de plus de 4 pièces
255
256 SELECT Nom_region, Valeur/Surface_carrez AS Prix_metre_carre FROM Loc
257 JOIN (SELECT * FROM Bien JOIN Vente ON Bien.Id_bien = Vente.Id_bien WHERE Total_piece >= 4) AS vb
258 ON vb.Id_commune = Loc.Id_commune
259 GROUP BY Nom_region ORDER BY Prix_metre_carre DESC;
260
261
262
```

Grid view:

	Nom_region	Prix_metre_carre
1	Auvergne-Rhône-Alpes	5538
2	Pays de la Loire	3554
3	Martinique	2991
4	Île-de-France	2792
5	La Réunion	2791
6	Normandie	2782
7	Bourgogne-Franche-Comté	2437
8	Guyane	2130
9	Centre-Val de Loire	2018
10	Occitanie	1315
11	Nouvelle-Aquitaine	1221
12	Grand Est	1172
13	Hauts-de-France	1087
14	Provence-Alpes-Côte d'Azur	800
15	Bretagne	616

Total rows loaded: 15

COMMUNE AYANT 50 VENTES+

Requête 9 : Liste des communes ayant eu au moins 50 ventes au 1^{er} trimestre

The screenshot shows a SQLite database interface with the following details:

- Databases:** DATAImm (SQLite 3) contains 8 tables: Bien, Commune, Departement, Loc, Population, Region, and Vente.
- Loc Table (selected):** Columns include Id_commune, Nom_commune, Code_commune, Code_departement, Nom_departement, Code_region, and Nom_region.
- SQL Editor:** The query is as follows:

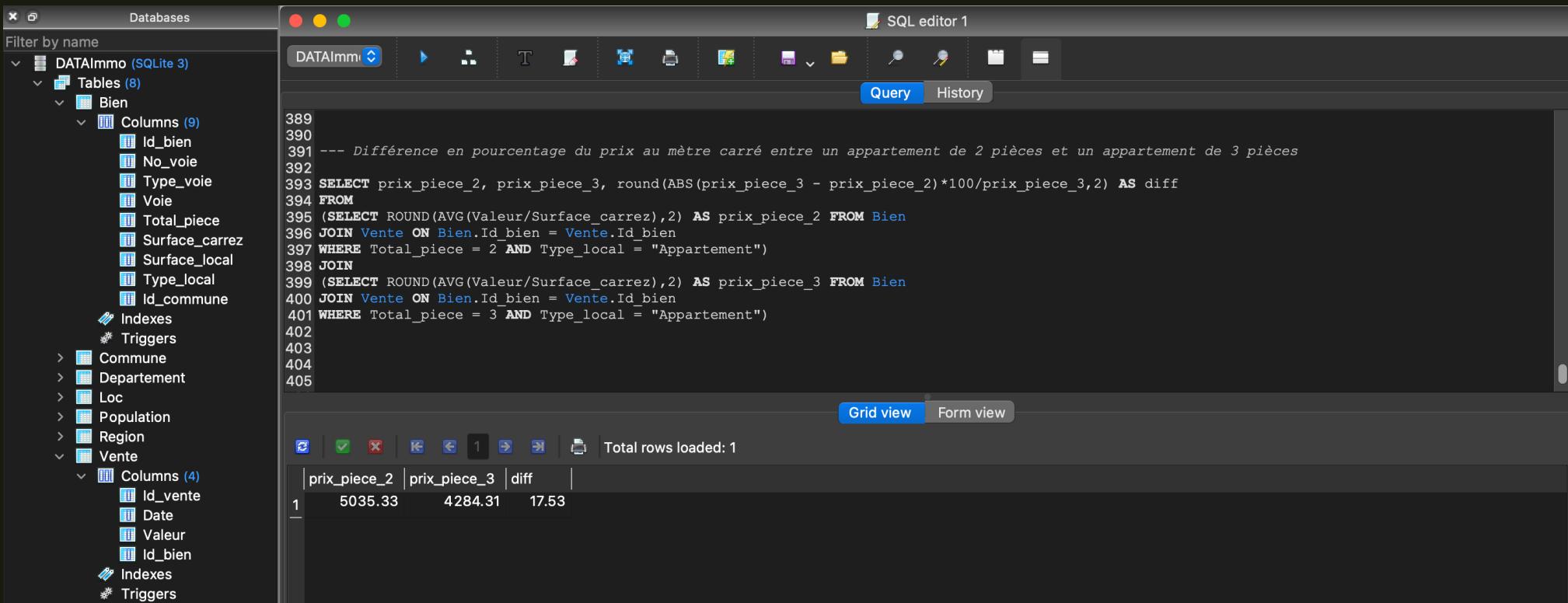
```
278 --- Liste des communes ayant eu au moins 50 ventes au 1er trimestre
279
280 SELECT Id_commune, Nom_commune, COUNT(Id_vente) AS Nombre_vente FROM Loc
281 JOIN (SELECT * FROM Bien JOIN VENTE ON Bien.Id_bien = Vente.Id_bien) AS vb
282 ON vb.Id_commune = Loc.Id_commune
283 WHERE Date BETWEEN "2020-01-01" AND "2020-04-01"
284 GROUP BY Loc.Id_commune
285 HAVING Nombre_vente >= 50;
286
287
```

Results: Total rows loaded: 48

	Id_commune	Nom_commune	Nombre_vente
1	26	AJACCIO	54
2	68	ANGERS	64
3	81	ANTIBES	77
4	121	ASNIERES-SUR-SEINE	81
5	345	BORDEAUX	157
6	359	BOULOGNE-BILLANCOURT	100
7	724	COURBEVOIE	80
8	1087	GRENOBLE	106
9	1176	ISSY-LES-MOULINEAUX	50
10	1248	LA CIOTAT	62
11	1506	LEVALLOIS-PERRET	59
12	1518	LILLE	67
13	1666	MARSEILLE 1ER	71
14	1669	MARSEILLE 4EME	72
15	1673	MARSEILLE 8EME	81
16	1674	MARSEILLE 9EME	66

DIFFÉRENCE DU PRIX

Requête 10 : Différence en pourcentage du prix au mètre carré entre un appartement de 2 pièces et un appartement de 3 pièces



The screenshot shows a SQL editor interface with a sidebar for database navigation and a main area for writing and executing SQL queries.

Database Sidebar:

- Databases (1): DATAImmobilier (SQLite 3)
- Tables (8):
 - Bien
 - Vente
 - Commune
 - Departement
 - Loc
 - Population
 - Region

SQL Editor Area:

```
389
390
391 --- Différence en pourcentage du prix au mètre carré entre un appartement de 2 pièces et un appartement de 3 pièces
392
393 SELECT prix_piece_2, prix_piece_3, round(ABS(prix_piece_3 - prix_piece_2)*100/prix_piece_3,2) AS diff
394 FROM
395 (SELECT ROUND(AVG(Valeur/Surface_carrez),2) AS prix_piece_2 FROM Bien
396 JOIN Vente ON Bien.Id_bien = Vente.Id_bien
397 WHERE Total_piece = 2 AND Type_local = "Appartement")
398 JOIN
399 (SELECT ROUND(AVG(Valeur/Surface_carrez),2) AS prix_piece_3 FROM Bien
400 JOIN Vente ON Bien.Id_bien = Vente.Id_bien
401 WHERE Total_piece = 3 AND Type_local = "Appartement")
402
403
404
405
```

Result Grid View:

	prix_piece_2	prix_piece_3	diff
1	5035.33	4284.31	17.53

Total rows loaded: 1

VALEUR FONCIÈRE MOYENNE

Requête 11 : Moyenne de valeur foncières pour le top 3 des communes des départements 6, 13, 33, 59 et 69.

The screenshot shows a database interface with a sidebar containing a tree view of tables and their columns. The main area displays a SQL query and its execution results.

Databases:

- DATAImmo (SQLite 3)
 - Tables (8)
 - Bien
 - Columns (9)
 - Loc
 - Columns (7)
 - Vente
 - Columns (4)
 - Commune
 - Departement
 - Z_MasterTable
 - Views

SQL editor 1:

```
308 --- Moyenne de valeur foncières pour le top 3 des communes des départements 6, 13, 33, 59 et 69
309 WITH a AS
310 (
311 SELECT
312 Code_departement, Nom_departement, Loc.Id_commune, Nom_commune, ROUND(AVG(Valeur),2) AS Valeur_moyenne,
313 RANK() OVER (PARTITION BY Nom_departement ORDER BY ROUND(AVG(Valeur),2) DESC) AS Rank_valeur
314 FROM Loc
315 JOIN
316 (SELECT * FROM Bien JOIN Vente ON Bien.Id_bien = Vente.Id_bien) AS vb
317 ON vb.Id_commune = Loc.Id_commune
318 GROUP BY Code_commune
319 HAVING Code_departement = "06" OR Code_departement = "13" OR Code_departement = "33" OR Code_departement = "59" OR Code_departement = "69"
320 )
321 SELECT Code_departement, Nom_departement, Nom_commune, Valeur_moyenne, Rank_valeur
322 FROM a
323 WHERE Rank_valeur < 4 ;
324
```

Grid view:

	Code_departement	Nom_departement	Nom_commune	Valeur_moyenne	Rank_valeur
1	06	Alpes-Maritimes	LA ROQUETTE SUR SIAGNE	534390.19	1
2	06	Alpes-Maritimes	ROQUEBRUNE CAP MARTIN	516114.93	2
3	06	Alpes-Maritimes	SAINT-ANDRE DE LA ROCHE	418647.66	3
4	13	Bouches-du-Rhône	VITROLLES	710418.08	1
5	13	Bouches-du-Rhône	CHATEAUNEUF-LES-MARTIGUES	350952.31	2
6	13	Bouches-du-Rhône	GRAVESON	288783.1	3
7	33	Gironde	CADAUJAC	401427.1	1
8	33	Gironde	SAINT-QUENTIN-DE-BARON	321013.16	2
9	33	Gironde	LEGE-CAP-FERRET	308410.33	3
10	59	Nord	WASQUEHAL	402124.8	1
11	59	Nord	WATTRELOS	384581.19	2
12	59	Nord	MARQUETTE-LEZ-LILLE	316774.69	3
13	69	Rhône	SATHONAY-CAMP	283191.67	1
14	69	Rhône	SAINT-PIERRE-DE-CHANDIEU	239600	2
15	69	Rhône	SAINT LAURENT D'AGNY	220134.07	3