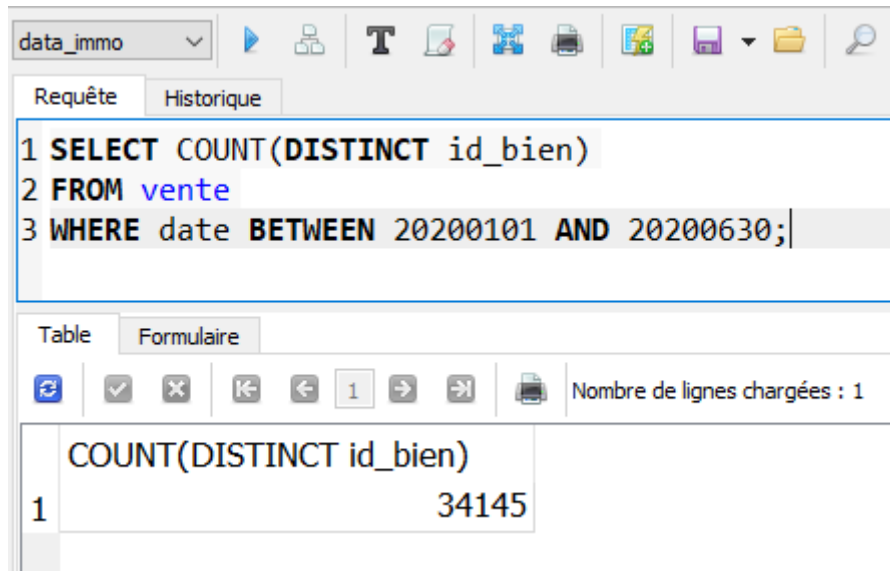


## Table des matières

|  |    |
|--|----|
| 1. Nombre total d'appartements vendus au 1er semestre 2020.....  | 2  |
| 2. Le nombre de ventes d'appartement par région pour le 1er semestre 2020.....   | 2  |
| 3. Proportion des ventes d'appartements par le nombre de pièces. ....  | 4  |
| 4. Liste des 10 départements où le prix du mètre carré est le plus élevé. ....   | 5  |
| 5. Prix moyen du mètre carré d'une maison en Île-de-France.....  | 6  |
| 6. Liste des 10 appartements les plus chers avec la région et le nombre de mètres carrés.....                            | 7  |
| 7. Taux d'évolution du nombre de ventes entre le premier et le second trimestre de 2020.....                             | 9  |
| 8. Le classement des régions par rapport au prix au mètre carré des appartements de plus de 4 pièces.....                | 10 |
| 9. Liste des communes ayant eu au moins 50 ventes au 1er trimestre.....  | 11 |
| 10. Différence en pourcentage du prix au mètre carré entre un appartement de 2 pièces et un appartement de 3 pièces..... | 13 |
| 11. Les moyennes de valeurs foncières pour le top 3 des communes des départements 6, 13, 33, 59 et 69.....               | 15 |

## 1. Nombre total d'appartements vendus au 1er semestre 2020.

```
SELECT COUNT(DISTINCT id_bien)
FROM vente
WHERE date BETWEEN 20200101 AND 20200630;
```



The screenshot shows a database query tool interface. At the top, there's a toolbar with various icons. Below it, there are tabs for 'Requête' and 'Historique'. The 'Requête' tab is active, displaying the following SQL query:

```
1 SELECT COUNT(DISTINCT id_bien)
2 FROM vente
3 WHERE date BETWEEN 20200101 AND 20200630;
```

Below the query editor, there are tabs for 'Table' and 'Formulaire'. The 'Table' tab is active, showing a table with one row and two columns. The first column is labeled 'COUNT(DISTINCT id\_bien)' and the second column contains the value '34145'. The table is titled 'Table' and 'Formulaire'. The status bar at the bottom indicates 'Nombre de lignes chargées : 1'.

|   | COUNT(DISTINCT id_bien) |
|---|-------------------------|
| 1 | 34145                   |

## 2. Le nombre de ventes d'appartement par région pour le 1er semestre 2020.

```
SELECT
id_region,
nom_region,
type_local,
COUNT(id_vente) AS nombre_de_vente

FROM vente
NATURAL JOIN bien
NATURAL JOIN commune
NATURAL JOIN region

WHERE type_local="Appartement"
AND date BETWEEN 20200101 AND 20200630

GROUP BY nom_region, id_region, type_local

ORDER BY nombre_de_vente DESC;
```

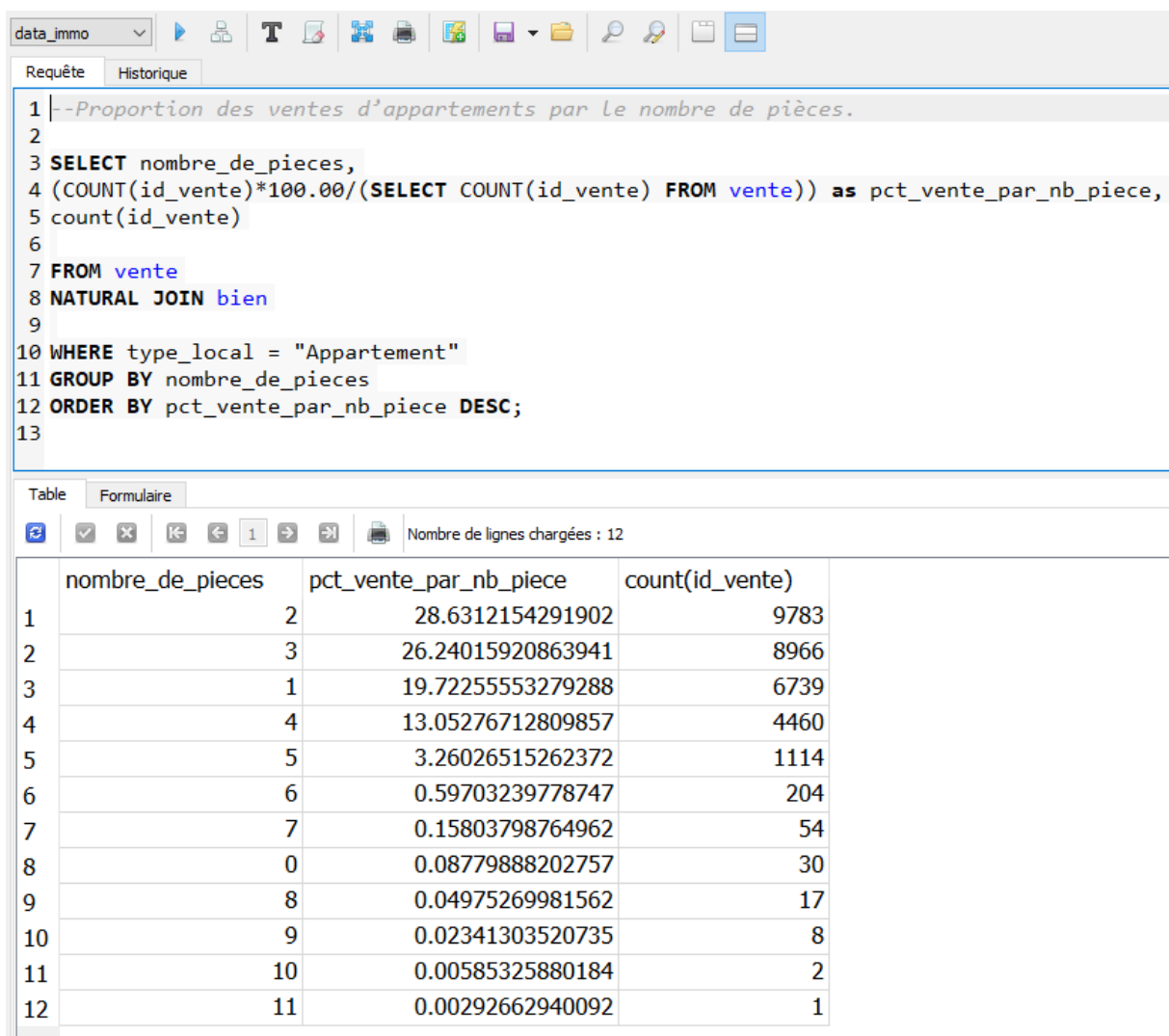


### 3. Proportion des ventes d'appartements par le nombre de pièces.

```
SELECT nombre_de_pieces,  
(COUNT(id_vente)*100.00/(SELECT COUNT(id_vente) FROM vente)) as  
pct_vente_par_nb_piece,  
count(id_vente)
```

```
FROM vente  
NATURAL JOIN bien
```

```
WHERE type_local = "Appartement"  
GROUP BY nombre_de_pieces  
ORDER BY pct_vente_par_nb_piece DESC;
```



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```
1 | --Proportion des ventes d'appartements par Le nombre de pièces.  
2 |  
3 | SELECT nombre_de_pieces,  
4 | (COUNT(id_vente)*100.00/(SELECT COUNT(id_vente) FROM vente)) as pct_vente_par_nb_piece,  
5 | count(id_vente)  
6 |  
7 | FROM vente  
8 | NATURAL JOIN bien  
9 |  
10 | WHERE type_local = "Appartement"  
11 | GROUP BY nombre_de_pieces  
12 | ORDER BY pct_vente_par_nb_piece DESC;  
13 |
```

Below the query editor, there is a table view showing the results of the query. The table has four columns: `nombre_de_pieces`, `pct_vente_par_nb_piece`, and `count(id_vente)`. The results are ordered by `pct_vente_par_nb_piece` in descending order.

|    | nombre_de_pieces | pct_vente_par_nb_piece | count(id_vente) |
|----|------------------|------------------------|-----------------|
| 1  | 2                | 28.6312154291902       | 9783            |
| 2  | 3                | 26.24015920863941      | 8966            |
| 3  | 1                | 19.72255553279288      | 6739            |
| 4  | 4                | 13.05276712809857      | 4460            |
| 5  | 5                | 3.26026515262372       | 1114            |
| 6  | 6                | 0.59703239778747       | 204             |
| 7  | 7                | 0.15803798764962       | 54              |
| 8  | 0                | 0.08779888202757       | 30              |
| 9  | 8                | 0.04975269981562       | 17              |
| 10 | 9                | 0.02341303520735       | 8               |
| 11 | 10               | 0.00585325880184       | 2               |
| 12 | 11               | 0.00292662940092       | 1               |

#### 4. Liste des 10 départements où le prix du mètre carré est le plus élevé.

```
SELECT code_dep, SUM(valeur)/SUM(surface_carrez) AS moyenne_m²_dep
```

```
FROM vente
```

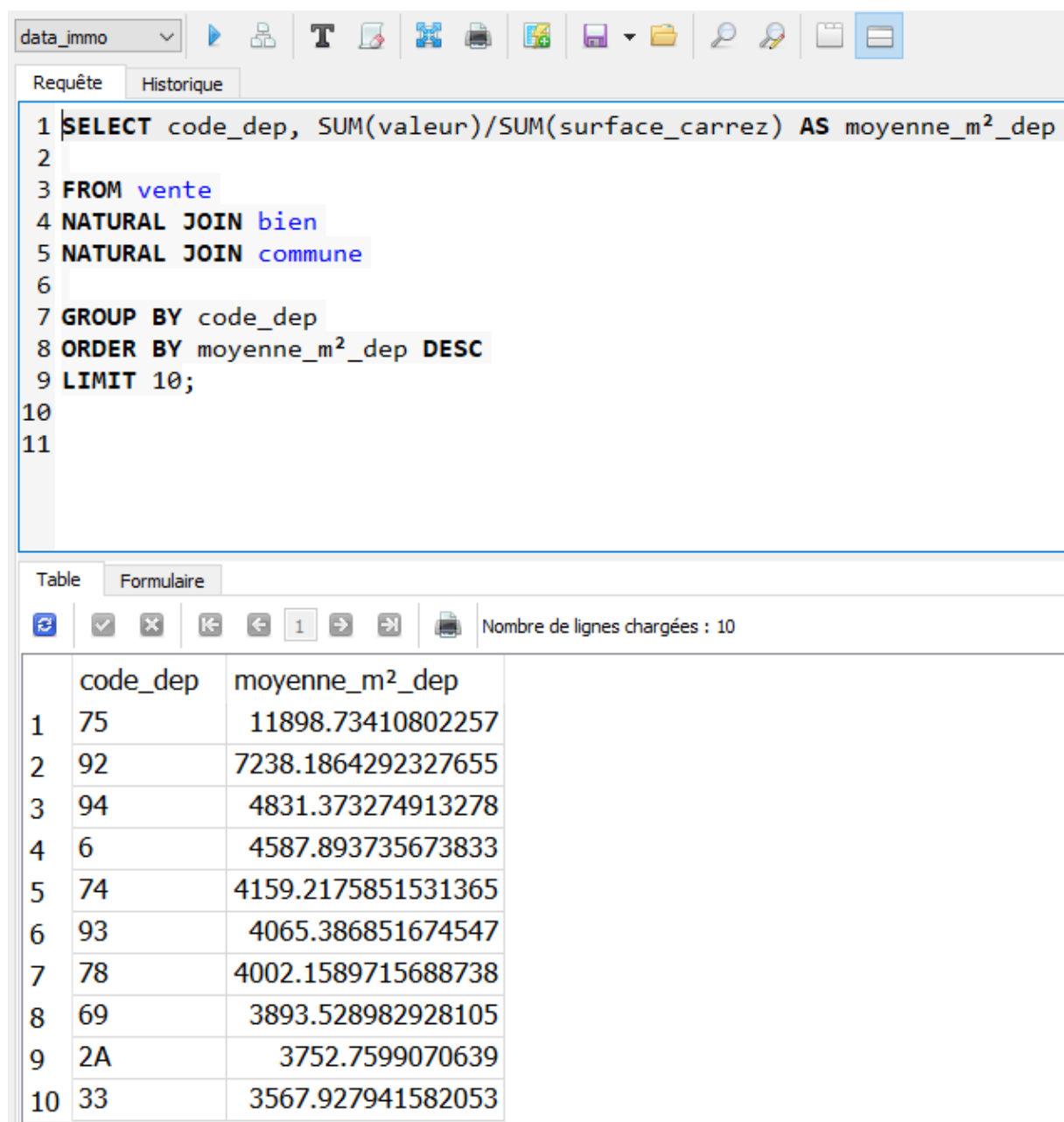
```
NATURAL JOIN bien
```

```
NATURAL JOIN commune
```

```
GROUP BY code_dep
```

```
ORDER BY moyenne_m²_dep DESC
```

```
LIMIT 10;
```



The screenshot shows a database query tool interface. At the top, there's a toolbar with various icons. Below it, a tab labeled 'Requête' is active, displaying the following SQL query:

```
1 SELECT code_dep, SUM(valeur)/SUM(surface_carrez) AS moyenne_m²_dep
2
3 FROM vente
4 NATURAL JOIN bien
5 NATURAL JOIN commune
6
7 GROUP BY code_dep
8 ORDER BY moyenne_m²_dep DESC
9 LIMIT 10;
10
11
```

Below the query editor, there's a tab labeled 'Table' which is active, showing the results of the query. The results are displayed in a table with two columns: 'code\_dep' and 'moyenne\_m²\_dep'. The table contains 10 rows of data, ordered by the average price per square meter in descending order.

|    | code_dep | moyenne_m²_dep     |
|----|----------|--------------------|
| 1  | 75       | 11898.73410802257  |
| 2  | 92       | 7238.1864292327655 |
| 3  | 94       | 4831.373274913278  |
| 4  | 6        | 4587.893735673833  |
| 5  | 74       | 4159.2175851531365 |
| 6  | 93       | 4065.386851674547  |
| 7  | 78       | 4002.1589715688738 |
| 8  | 69       | 3893.528982928105  |
| 9  | 2A       | 3752.7599070639    |
| 10 | 33       | 3567.927941582053  |

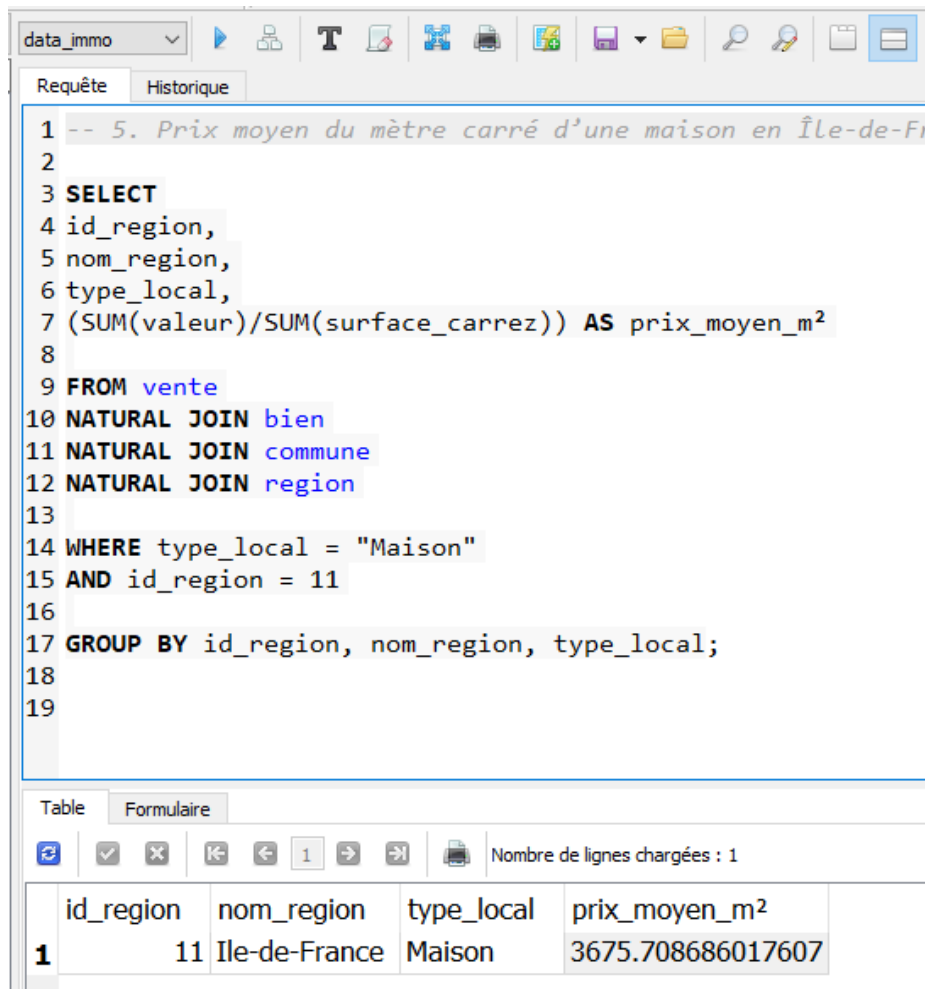
## 5. Prix moyen du mètre carré d'une maison en Île-de-France.

```
SELECT
id_region,
nom_region,
type_local,
(SUM(valeur)/SUM(surface_carrez)) AS prix_moyen_m²

FROM vente
NATURAL JOIN bien
NATURAL JOIN commune
NATURAL JOIN region

WHERE type_local = "Maison"
AND id_region = 11

GROUP BY id_region, nom_region, type_local;
```



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```
1 -- 5. Prix moyen du mètre carré d'une maison en Île-de-Fr
2
3 SELECT
4 id_region,
5 nom_region,
6 type_local,
7 (SUM(valeur)/SUM(surface_carrez)) AS prix_moyen_m²
8
9 FROM vente
10 NATURAL JOIN bien
11 NATURAL JOIN commune
12 NATURAL JOIN region
13
14 WHERE type_local = "Maison"
15 AND id_region = 11
16
17 GROUP BY id_region, nom_region, type_local;
18
19
```

Below the query editor, there is a results table with the following data:

|   | id_region | nom_region    | type_local | prix_moyen_m²     |
|---|-----------|---------------|------------|-------------------|
| 1 | 11        | Ile-de-France | Maison     | 3675.708686017607 |

## 6. Liste des 10 appartements les plus chers avec la région et le nombre de mètres carrés.

```
SELECT
id_bien,
nom_de_la_voie,
nom_commune,
type_local,
nom_region,
valeur,
surface_carrez

FROM vente
NATURAL JOIN bien
NATURAL JOIN commune
NATURAL JOIN region

WHERE valeur <> ""

ORDER BY valeur DESC
LIMIT 10;
```

data\_immo

Requête

Historique

```

2 --avec la région et le nombre de mètres carrés.
3
4 SELECT
5 id_bien,
6 nom_de_la_voie,
7 nom_commune,
8 type_local,
9 nom_region,
10 valeur,
11 surface_carrez
12
13 FROM vente
14 NATURAL JOIN bien
15 NATURAL JOIN commune
16 NATURAL JOIN region
17
18 WHERE valeur <>""
19
20 ORDER BY valeur DESC
21 LIMIT 10;
22
23

```

Table

Formulaire

Nombre de lignes chargées : 10

|    | id_bien | nom_de_la_voie  | nom_commune      | type_local  | nom_region    | valeur  | surface_carrez |
|----|---------|-----------------|------------------|-------------|---------------|---------|----------------|
| 1  | 32252   | SUCHET          | PARIS 16         | Appartement | Ile-de-France | 9000000 | 9.1            |
| 2  | 21817   | DE LA CAVIGNON  | CORBEIL ESSONNES | Appartement | Ile-de-France | 8600000 | 64             |
| 3  | 29778   | DU BAC          | PARIS 07         | Appartement | Ile-de-France | 8577713 | 20.55          |
| 4  | 32410   | LEMERCIER       | PARIS 17         | Appartement | Ile-de-France | 7620000 | 42.77          |
| 5  | 29829   | D ASSAS         | PARIS 06         | Appartement | Ile-de-France | 7600000 | 253.3          |
| 6  | 29501   | SAINT HYACINTHE | PARIS 01         | Appartement | Ile-de-France | 7535000 | 139.9          |
| 7  | 31950   | GEORGES MANDEL  | PARIS 16         | Appartement | Ile-de-France | 7420000 | 360.95         |
| 8  | 32112   | DE BEAUSEJOUR   | PARIS 16         | Appartement | Ile-de-France | 7200000 | 595            |
| 9  | 29332   | CAMBON          | PARIS 01         | Appartement | Ile-de-France | 7050000 | 122.56         |
| 10 | 29492   | SAINT HONORE    | PARIS 01         | Appartement | Ile-de-France | 6600000 | 79.38          |



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

```
WITH
vente1 AS (
SELECT round(count(id_vente),2) AS nombre_vente_t1
FROM vente
WHERE date BETWEEN 20200101 AND 20200331),
vente2 AS (
SELECT round(count(id_vente),2) AS nombre_vente_t2
FROM vente
WHERE date BETWEEN 20200401 AND 20200631)

SELECT ( (nombre_vente_t2 - nombre_vente_t1) / nombre_vente_t1 * 100.00) AS
taux_evolution_vente
FROM vente1,
vente2;
```

The screenshot shows a SQL query editor with a toolbar at the top. The query is entered in the 'Requête' tab. Below the query, the 'Table' tab is active, displaying a single row of results. The results table has one column, 'taux\_evolution\_vente', and one row with the value '3.67787315212208'. The status bar at the bottom indicates 'Nombre de lignes chargées : 1'.

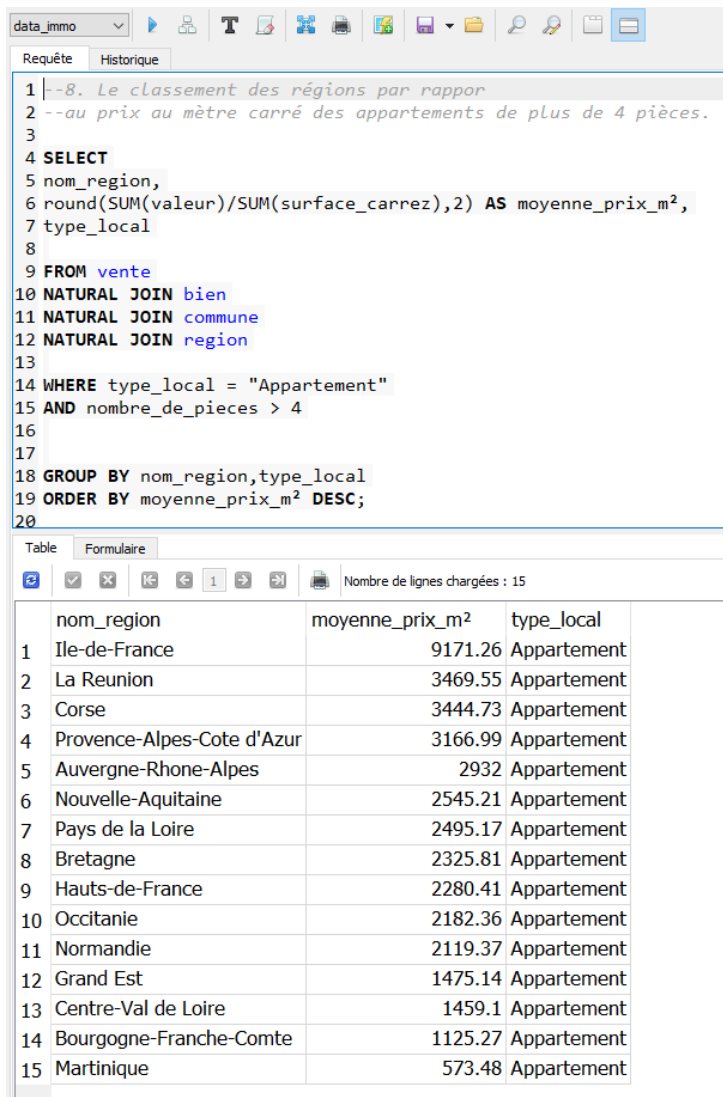
| taux_evolution_vente |
|----------------------|
| 3.67787315212208     |

## 8. Le classement des régions par rapport au prix au mètre carré des appartements de plus de 4 pièces.

```
SELECT
nom_region,
round(SUM(valeur)/SUM(surface_carrez),2) AS moyenne_prix_m²,
type_local

FROM vente
NATURAL JOIN bien
NATURAL JOIN commune
NATURAL JOIN region

WHERE type_local = "Appartement" AND nombre_de_pieces > 4
GROUP BY nom_region,type_local
ORDER BY moyenne_prix_m² DESC;
```



data\_immo

Requête Historique

```
1 --8. Le classement des régions par rappor
2 --au prix au mètre carré des appartements de plus de 4 pièces.
3
4 SELECT
5 nom_region,
6 round(SUM(valeur)/SUM(surface_carrez),2) AS moyenne_prix_m²,
7 type_local
8
9 FROM vente
10 NATURAL JOIN bien
11 NATURAL JOIN commune
12 NATURAL JOIN region
13
14 WHERE type_local = "Appartement"
15 AND nombre_de_pieces > 4
16
17
18 GROUP BY nom_region,type_local
19 ORDER BY moyenne_prix_m² DESC;
20
```

Table Formulaire

Nombre de lignes chargées : 15

|    | nom_region                 | moyenne_prix_m² | type_local  |
|----|----------------------------|-----------------|-------------|
| 1  | Île-de-France              | 9171.26         | Appartement |
| 2  | La Reunion                 | 3469.55         | Appartement |
| 3  | Corse                      | 3444.73         | Appartement |
| 4  | Provence-Alpes-Cote d'Azur | 3166.99         | Appartement |
| 5  | Auvergne-Rhone-Alpes       | 2932            | Appartement |
| 6  | Nouvelle-Aquitaine         | 2545.21         | Appartement |
| 7  | Pays de la Loire           | 2495.17         | Appartement |
| 8  | Bretagne                   | 2325.81         | Appartement |
| 9  | Hauts-de-France            | 2280.41         | Appartement |
| 10 | Occitanie                  | 2182.36         | Appartement |
| 11 | Normandie                  | 2119.37         | Appartement |
| 12 | Grand Est                  | 1475.14         | Appartement |
| 13 | Centre-Val de Loire        | 1459.1          | Appartement |
| 14 | Bourgogne-Franche-Comte    | 1125.27         | Appartement |
| 15 | Martinique                 | 573.48          | Appartement |

## 9. Liste des communes ayant eu au moins 50 ventes au 1er trimestre

```
SELECT COUNT(id_vente) AS nombre_vente,  
nom_commune
```

```
FROM vente  
NATURAL JOIN bien  
NATURAL JOIN commune
```

```
WHERE date < 20200401
```

```
GROUP BY nom_commune  
HAVING nombre_vente > 49  
ORDER BY nombre_vente DESC;
```

|   |              |                      |
|---|--------------|----------------------|
| data_immo   |              |                      |
| Requête Historique  |              |                      |
| <pre> 1 --9. Liste des communes ayant eu au moins 50 ventes au 1er trimestre 2 3 SELECT COUNT(id_vente) AS nombre_vente, 4 nom_commune 5 6 FROM vente 7 NATURAL JOIN bien 8 NATURAL JOIN commune 9 10 WHERE date &lt; 20200401 11 12 GROUP BY nom_commune 13 HAVING nombre_vente &gt; 49 14 ORDER BY nombre_vente DESC; 15 </pre> |              |                      |
| Table Formulaire  |              |                      |
| <div> <div> <div></div> <div></div> <div></div> <div></div> <div>1</div> <div></div> <div></div> <div></div> <div></div> </div> <div>Nombre de lignes chargées : 48</div> </div>  |              |                      |
|   | nombre_vente | nom_commune          |
| 1   | 228          | PARIS 17             |
| 2   | 215          | PARIS 15             |
| 3   | 209          | PARIS 18             |
| 4   | 173          | NICE                 |
| 5   | 169          | PARIS 11             |
| 6   | 165          | PARIS 16             |
| 7   | 157          | BORDEAUX             |
| 8   | 146          | PARIS 14             |
| 9   | 127          | PARIS 20             |
| 10  | 119          | NANTES               |
| 11  | 116          | PARIS 19             |
| 12  | 110          | PARIS 12             |
| 13  | 109          | PARIS 10             |
| 14  | 106          | PARIS 09             |
| 15  | 106          | GRENOBLE             |
| 16  | 99           | BOULOGNE BILLANCOURT |
| 17  | 94           | PARIS 13             |
| 18  | 87           | PARIS 07             |
| 19  | 86           | PARIS 06             |

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

```
WITH
P2 AS (
SELECT (SUM(valeur)/SUM(surface_carrez)) AS moyenne_prix_m²_P2
FROM vente
NATURAL JOIN bien
WHERE nombre_de_pieces = 2
AND type_local = "Appartement"),

P3 AS (
SELECT (SUM(valeur)/SUM(surface_carrez)) AS moyenne_prix_m²_P3
FROM vente
NATURAL JOIN bien
WHERE nombre_de_pieces = 3
AND type_local = "Appartement")

SELECT (
((moyenne_prix_m²_P2 - moyenne_prix_m²_P3)/moyenne_prix_m²_P2) * 100
) AS difference_pourcentage,
moyenne_prix_m²_P2,
moyenne_prix_m²_P3

FROM P2, P3;
```



## 11. Les moyennes de valeurs foncières pour le top 3 des communes des départements 6, 13, 33, 59 et 69

```
WITH t2 AS
(
  SELECT nom_commune,
  ROW_NUMBER() OVER (PARTITION BY code_dep ORDER BY AVG(valeur) DESC)
  row_num,
  code_dep,
  AVG(valeur) AS moyenne_valeur
  FROM vente
  NATURAL JOIN
  bien
  NATURAL JOIN
  commune
  WHERE code_dep IN (6, 13, 33, 59, 69) AND
  valeur <> ""
  GROUP BY nom_commune,
  code_dep
)
SELECT code_dep,
nom_commune,
row_num,
ROUND(moyenne_valeur, 2)
FROM t2
WHERE row_num IN (1, 2, 3)
ORDER BY code_dep;
```

Voir capture d'écran page suivante :

