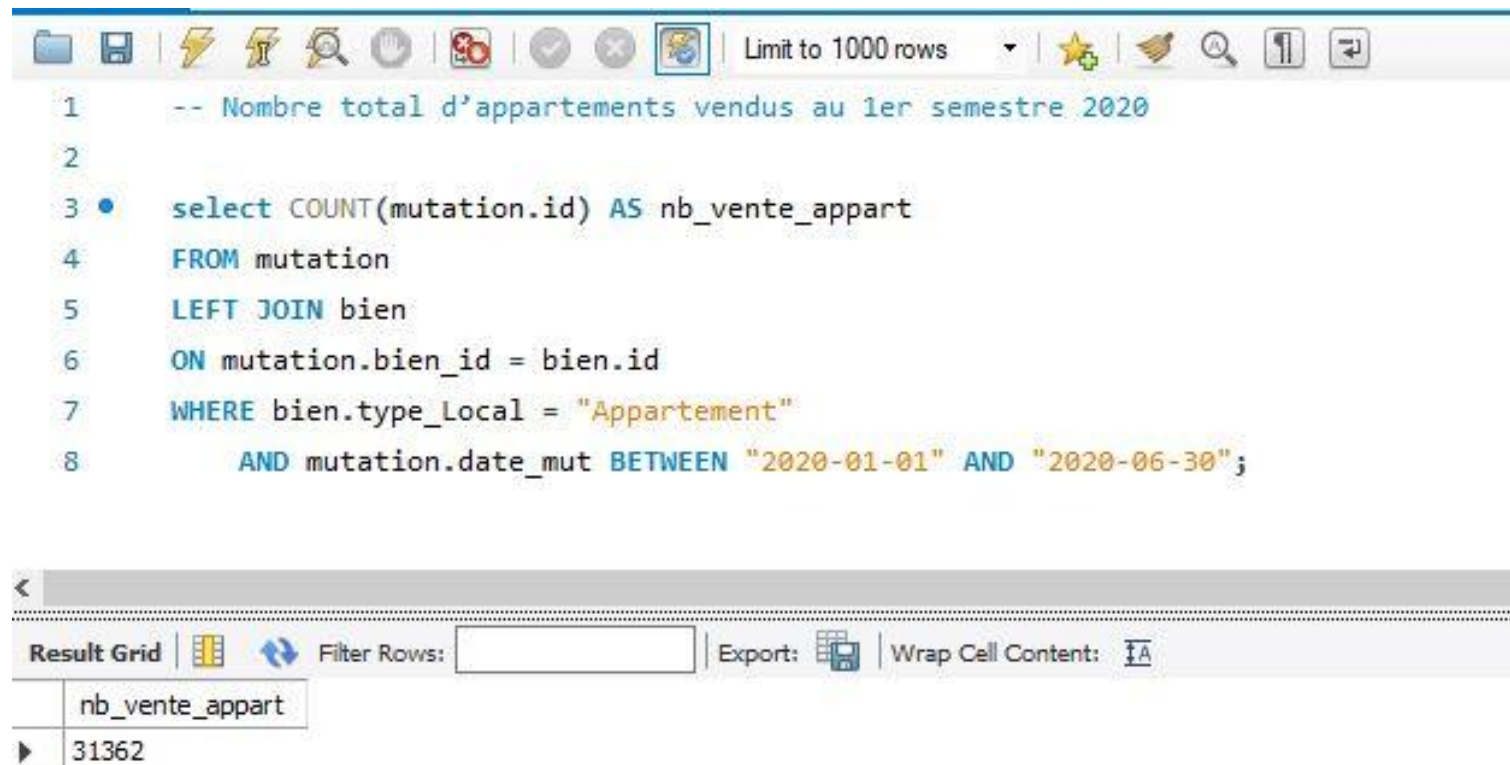


## Requête 1



The screenshot shows a SQL query editor interface. The top toolbar includes icons for file operations, execution, and search, along with a dropdown menu set to "Limit to 1000 rows". The query text is as follows:

```
1  -- Nombre total d'appartements vendus au 1er semestre 2020
2
3  • select COUNT(mutation.id) AS nb_vente_appart
4  FROM mutation
5  LEFT JOIN bien
6  ON mutation.bien_id = bien.id
7  WHERE bien.type_Local = "Appartement"
8  AND mutation.date_mut BETWEEN "2020-01-01" AND "2020-06-30";
```

Below the query editor, the "Result Grid" tab is active. It displays a single column named "nb\_vente\_appart" with a value of 31362. The interface also includes a "Filter Rows" input field, an "Export" button, and a "Wrap Cell Content" toggle.

nb_vente_appart
31362

## Requête 2

```
1  -- Proportion des ventes d'appartements par le nombre de pièces.
2
3  SET @req = (          -- Nombre total d'appartements vendus
4      SELECT COUNT(mutation.id)
5      FROM mutation
6      JOIN bien ON bien.id = mutation.bien_id
7      WHERE type_local = "Appartement"
8      AND type_local = "Appartement");
9
10 SELECT nb_piece, COUNT(mutation.id) AS nb_ventes, COUNT(mutation.id) / @req * 100 AS proportion_vente
11 FROM mutation
12 JOIN bien ON bien.id = mutation.bien_id
13 WHERE type_local = "Appartement"
14 GROUP BY nb_piece
15 ORDER BY nb_piece
16
```

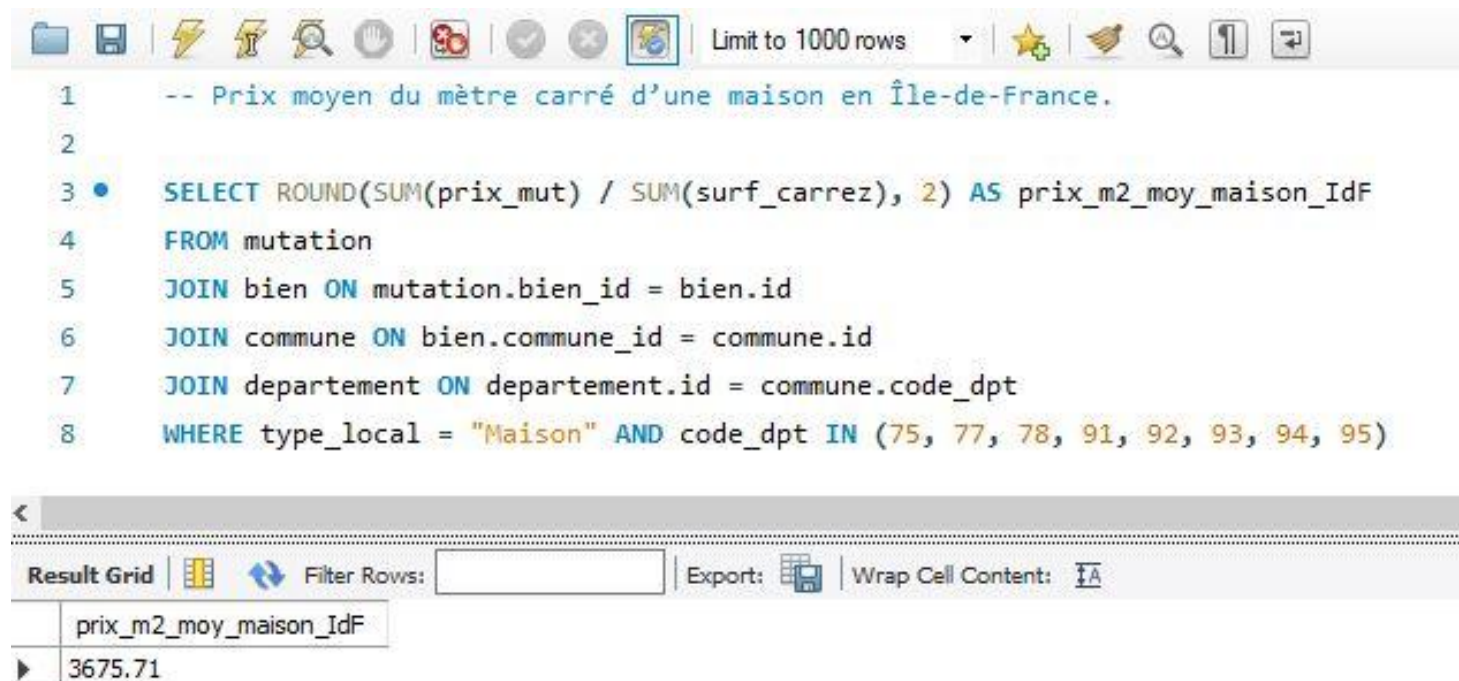
Result Grid			
Filter Rows: <input type="text"/>			
Export: <input type="button" value=""/>			
Wrap Cell Content: <input type="button" value=""/>			
	nb_piece	nb_ventes	proportion_vente
0	30	0.0957	
1	6736	21.4782	
2	9773	31.1619	
3	8966	28.5887	
4	4458	14.2147	
5	1114	3.5521	
6	203	0.6473	
7	54	0.1722	
8	17	0.0542	
9	8	0.0255	
10	2	0.0064	
11	1	0.0032	

### Requête 3

```
1  -- Liste des 10 départements où le prix du mètre carré est le plus élevé.
2
3  • SELECT code_dpt, nom_dpt, ROUND(SUM(prix_mut) / SUM(surf_carrez), 2) AS prix_m2
4  FROM mutation
5  JOIN bien ON mutation.bien_id = bien.id
6  JOIN commune ON bien.commune_id = commune.id
7  JOIN departement ON departement.id = commune.code_dpt
8  GROUP BY code_dpt
9  ORDER BY prix_m2 DESC
10 LIMIT 10;
```

Result Grid				Filter Rows:	Export:	Wrap Cell Content:	Fetch rows:
	code_dpt	nom_dpt	prix_m2				
▶	75	Paris	11912.46				
	92	Hauts-de-Seine	7238.19				
	94	Val-de-Marne	4832.75				
	6	Alpes-Maritimes	4591.20				
	74	Haute-Savoie	4159.22				
	93	Seine-Saint-Denis	4074.47				
	78	Yvelines	4002.16				
	69	Rhône	3893.53				
	2A	Corse-du-Sud	3764.55				
	33	Gironde	3567.93				

## Requête 4



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

```
1  -- Prix moyen du mètre carré d'une maison en Île-de-France.  
2  
3  • SELECT ROUND(SUM(prix_mut) / SUM(surf_carrez), 2) AS prix_m2_moy_maison_IdF  
4  FROM mutation  
5  JOIN bien ON mutation.bien_id = bien.id  
6  JOIN commune ON bien.commune_id = commune.id  
7  JOIN departement ON departement.id = commune.code_dpt  
8  WHERE type_local = "Maison" AND code_dpt IN (75, 77, 78, 91, 92, 93, 94, 95)
```

Below the query editor, there is a "Result Grid" section. It includes a "Filter Rows" input field, an "Export" button, and a "Wrap Cell Content" checkbox. The result grid shows a single row with the column name "prix\_m2\_moy\_maison\_IdF" and the value "3675.71".

	prix_m2_moy_maison_IdF
▶	3675.71

## Requête 5

```
1  -- Liste des 10 appartements les plus chers avec le département et le nombre de mètres carrés.
2
3  • SELECT bien.id, bien.surf_carrez, mutation.prix_mut AS prix_mut, commune.code_dpt, nom_dpt
4  FROM bien
5  JOIN mutation ON mutation.bien_id = bien.id
6  JOIN commune ON bien.commune_id = commune.id
7  JOIN departement ON departement.id = commune.code_dpt
8  WHERE bien.type_local = "Appartement"
9  ORDER BY mutation.prix_mut DESC
10 LIMIT 10
```

Result Grid



Filter Rows:

Export:



Wrap Cell Content:



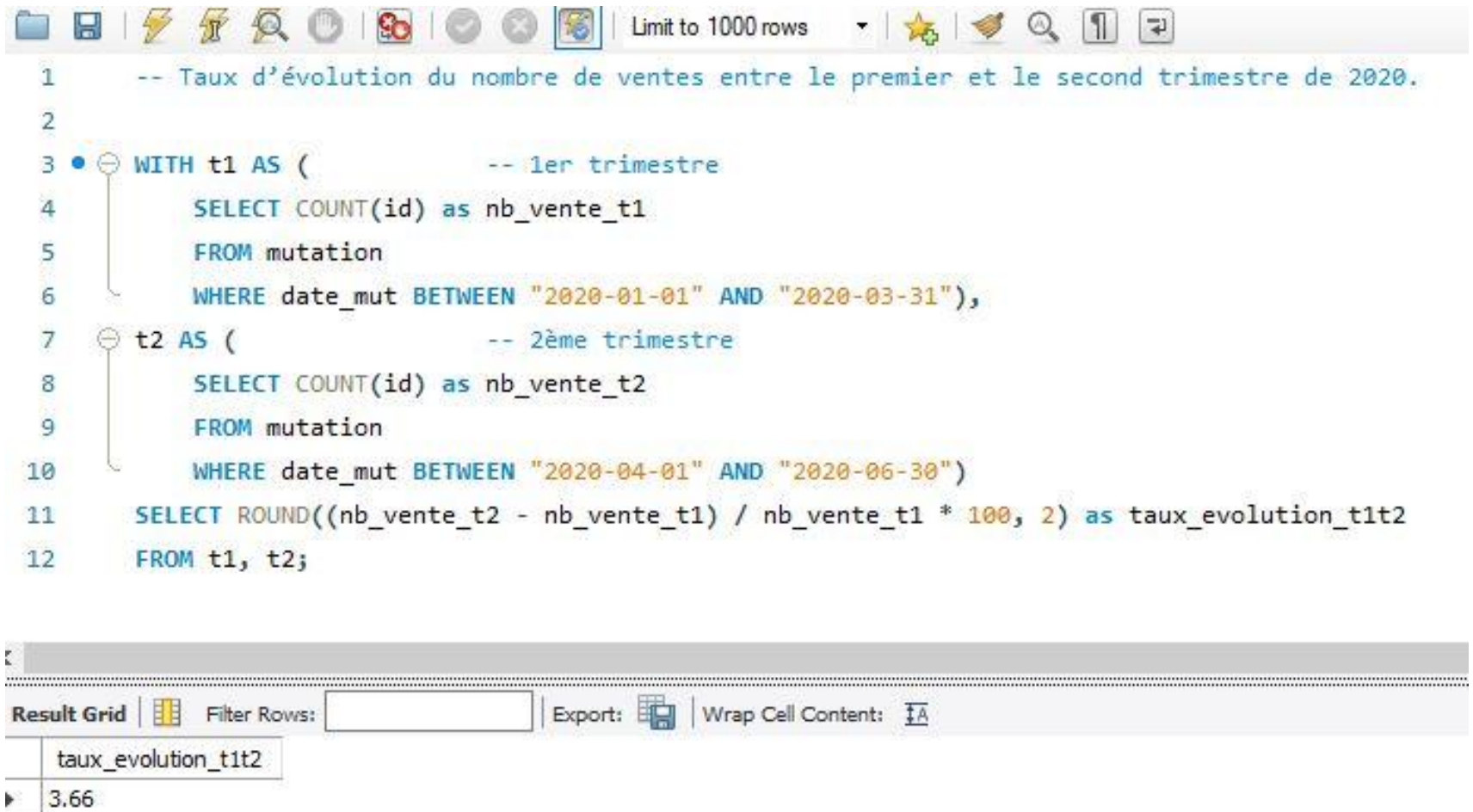
Fetch rows:



	id	surf_carrez	prix_mut	code_dpt	nom_dpt
•	75116nanDD1853	9.10	9000000.00	75	Paris
	91174nanAX16322	64.00	8600000.00	91	Essonne
	75107nanAK34100	20.55	8577713.00	75	Paris
	75117nanCO10015	42.77	7620000.00	75	Paris
	75106nanAU14104	253.30	7600000.00	75	Paris
	75101nanAZ8679	139.90	7535000.00	75	Paris
	75116nanDV465	360.95	7420000.00	75	Paris
	75116nanCN2411	595.00	7200000.00	75	Paris
	75101nanBC672	122.56	7050000.00	75	Paris
	75101nanAT4181	79.38	6600000.00	75	Paris



## Requête 6



```
1  -- Taux d'évolution du nombre de ventes entre le premier et le second trimestre de 2020.
2
3  WITH t1 AS (                -- 1er trimestre
4      SELECT COUNT(id) as nb_vente_t1
5      FROM mutation
6      WHERE date_mut BETWEEN "2020-01-01" AND "2020-03-31"),
7  t2 AS (                    -- 2ème trimestre
8      SELECT COUNT(id) as nb_vente_t2
9      FROM mutation
10     WHERE date_mut BETWEEN "2020-04-01" AND "2020-06-30")
11  SELECT ROUND((nb_vente_t2 - nb_vente_t1) / nb_vente_t1 * 100, 2) as taux_evolution_t1t2
12  FROM t1, t2;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [⌕](#)

taux_evolution_t1t2
3.66

## Requête 7

```

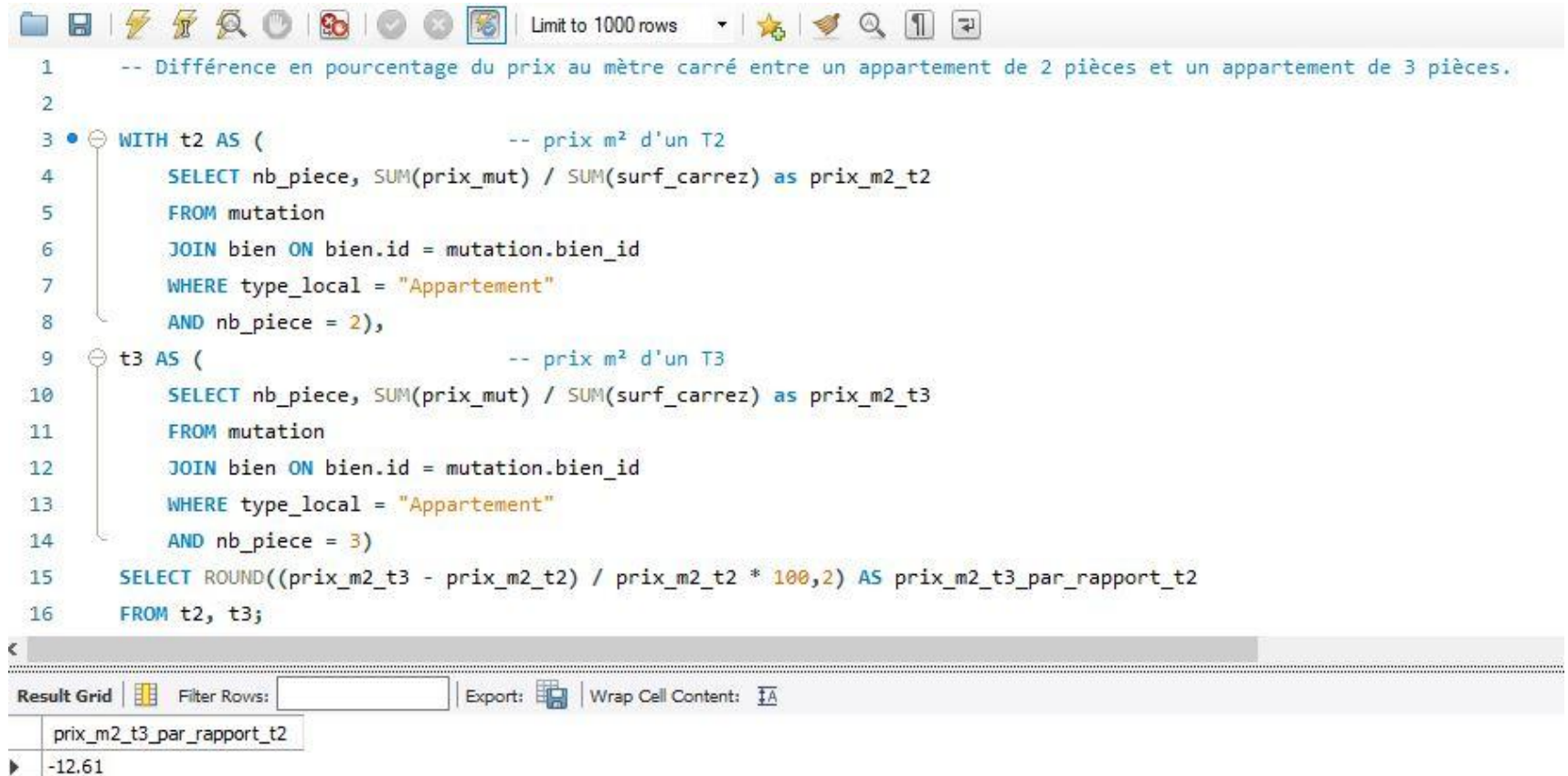
1  -- Liste des communes où le nombre de ventes a augmenté d'au moins 20% entre le premier et le second trimestre de 2020
2
3  • ⊖ WITH t1 AS (                -- 1er trimestre
4      SELECT commune_id, nom_commune, COUNT(mutation.id) as nb_vente_t1
5      FROM bien
6      JOIN mutation ON bien.id = mutation.bien_id
7      JOIN commune ON bien.commune_id = commune.id
8      WHERE date_mut BETWEEN "2020-01-01" AND "2020-03-31"
9      GROUP BY commune_id),      -- Arrondissements, communes ?
10  ⊖ t2 AS (                -- 2ème trimestre
11      SELECT commune_id, COUNT(mutation.id) as nb_vente_t2
12      FROM bien
13      JOIN mutation ON bien.id = mutation.bien_id
14      WHERE date_mut BETWEEN "2020-04-01" AND "2020-06-30"
15      GROUP BY commune_id)      -- Arrondissements, communes ?
16  SELECT t1.commune_id, t1.nom_commune, (nb_vente_t2 - nb_vente_t1) / nb_vente_t1 *100 as taux_evolution,
17      nb_vente_t1, nb_vente_t2
18  FROM t1
19  JOIN t2 USING (commune_id)
20  HAVING taux_evolution > 20

```

Result Grid |  Filter Rows:  | Export:  | Wrap Cell Content: 

commune_id	nom_commune	taux_evolution	nb_vente_t1	nb_vente_t2
301	LEUCATE	20.6897	29	35
205	ANTIBES	20.8333	24	29
2885	ETAMPES	20.8333	24	29
729	TOULOUSE	21.4286	14	17
2416	LE CHESNAY-ROCQUENCOURT	21.4286	14	17
3037	ORLY	22.2222	9	11
3044	BOISSY-SAINT-LEGER	22.2222	9	11
3164	FORT DE FRANCE	22.2222	9	11
3197	PARIS 08	24.1935	62	77
513	VALENCE	25.0000	28	35
811	CADAUJAC	25.0000	4	5
1176	SOLIGNY	25.0000	8	10

## Requête 8



```
1  -- Différence en pourcentage du prix au mètre carré entre un appartement de 2 pièces et un appartement de 3 pièces.
2
3  WITH t2 AS (                                -- prix m² d'un T2
4      SELECT nb_piece, SUM(prix_mut) / SUM(surf_carrez) as prix_m2_t2
5      FROM mutation
6      JOIN bien ON bien.id = mutation.bien_id
7      WHERE type_local = "Appartement"
8      AND nb_piece = 2),
9  t3 AS (                                -- prix m² d'un T3
10     SELECT nb_piece, SUM(prix_mut) / SUM(surf_carrez) as prix_m2_t3
11     FROM mutation
12     JOIN bien ON bien.id = mutation.bien_id
13     WHERE type_local = "Appartement"
14     AND nb_piece = 3)
15     SELECT ROUND((prix_m2_t3 - prix_m2_t2) / prix_m2_t2 * 100,2) AS prix_m2_t3_par_rapport_t2
16     FROM t2, t3;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

prix_m2_t3_par_rapport_t2
-12.61



## Requête 9

```
1  -- Les moyennes de valeurs foncières pour le top 3 des communes des départements 6, 13, 33, 59 et 69
2
3  • WITH valeur_par_ville AS (                -- Calcul des valeurs foncières moyennes par commune
4      SELECT code_dpt, nom_commune, avg(prix_mut) as valeur
5      FROM mutation
6      JOIN bien ON mutation.bien_id = bien.id
7      JOIN commune ON bien.commune_id = commune.id
8      WHERE code_dpt IN (6,13,33,59,69)
9      GROUP BY code_dpt, nom_commune)
10 SELECT code_dpt, nom_commune, round(valeur,2) AS prix_moyen
11 FROM (
12     SELECT code_dpt, nom_commune, valeur,
13     RANK() OVER (PARTITION BY code_dpt ORDER BY valeur DESC) AS rang
14 FROM valeur_par_ville) AS resultat
15 WHERE rang <= 3;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [IA](#)

code_dpt	nom_commune	prix_moyen
13	GIGNAC-LA-NERTHE	330000.00
13	SAINT SAVOURNIN	314425.00
13	CASSIS	313416.88
33	LEGE-CAP-FERRET	549500.64
33	VAYRES	335000.00
33	ARCACHON	307435.93
59	BERSEE	433202.00
59	CYSOING	408550.00
59	HALLUIN	322250.00
6	SAINT-JEAN-CAP-FERRAT	968750.00
6	EZE	655000.00
6	MOUANS-SARTOUX	476898.10
69	VILLE SUR JARNIOUX	485300.00
69	LYON 2EME	455217.27
69	LYON 6EME	426968.25