

Les requêtes et les résultats

1. Requête 1

- Enoncé :

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

- Script :

```
/* premiere requete*/  
select count(distinct m.id_BienImmo)  
from mutation m  
inner join bienimmo b on m.id_bienimmo = b.id_bienimmo  
where TypeDeBien= 'appartement' and dateMutation between '2020-01-01' and '2020-06-30'
```

- Résultat :

The screenshot shows a SQL IDE interface with multiple tabs. The active tab is 'requete 1'. The SQL editor contains the following query:

```
1 /* premiere requete*/  
2  
3 • select count(distinct m.id_BienImmo)  
4 from mutation m  
5 inner join bienimmo b on m.id_bienimmo = b.id_bienimmo  
6 where TypeDeBien= 'appartement' and dateMutation between '2020-01-01' and '2020-06-30'
```

Below the editor, the 'Result Grid' is displayed, showing the result of the query:

count(distinct m.id_BienImmo)
31358

The interface also includes a toolbar with various icons, a 'Filter Rows' input, an 'Export' button, and a 'Wrap Cell Content' checkbox. The status bar at the bottom indicates 'Result 2' and 'Read Only'.

2.Requête 2

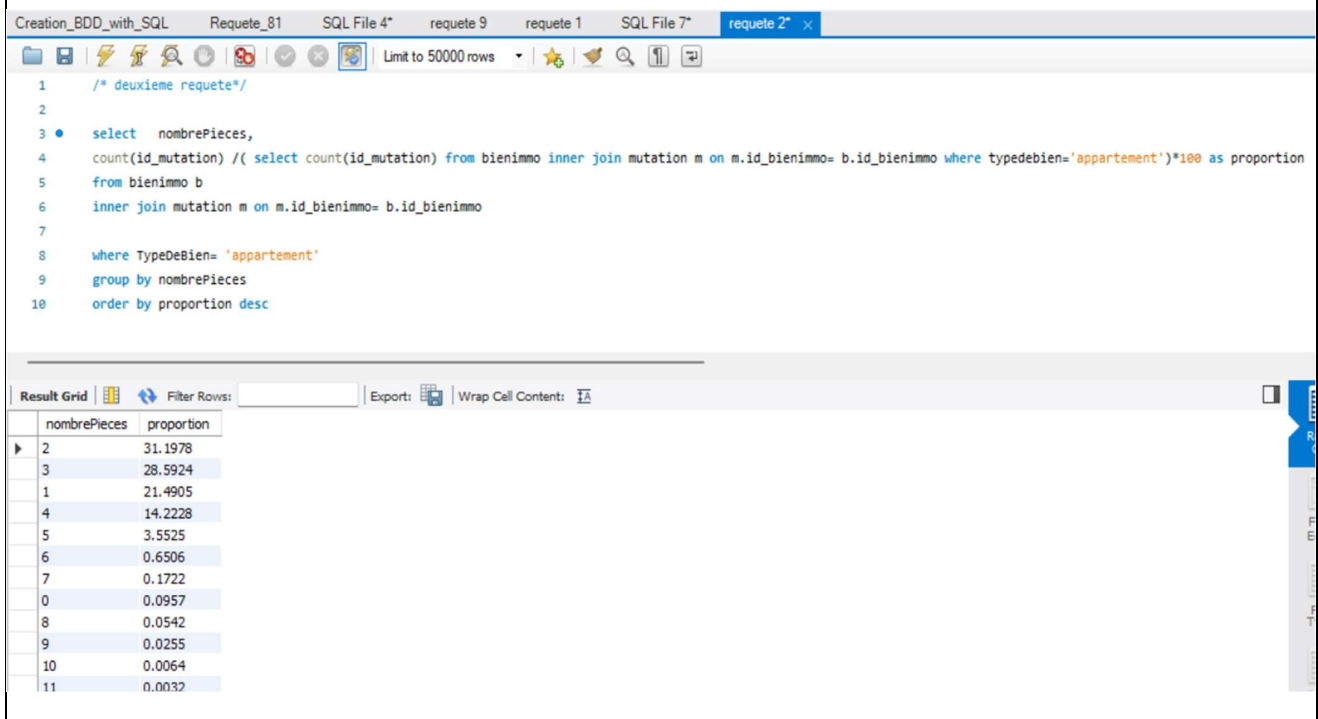
- Enoncé :

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

- Script :

```
/* deuxieme requete*/
select  nombrePieces,
count(id_mutation) /( select count(id_mutation)  from bienimmo
inner join mutation m on m.id_bienimmo= b.id_bienimmo where
typedebien='appartement')*100 as proportion
from bienimmo b
inner join mutation m on m.id_bienimmo= b.id_bienimmo
where TypeDeBien= 'appartement'
group by nombrePieces
order by Proportion desc
```

- Résultat :



The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL script:

```
1 /* deuxieme requete*/
2
3 • select  nombrePieces,
4 count(id_mutation) /( select count(id_mutation)  from bienimmo inner join mutation m on m.id_bienimmo= b.id_bienimmo where typedebien='appartement')*100 as proportion
5 from bienimmo b
6 inner join mutation m on m.id_bienimmo= b.id_bienimmo
7
8 where TypeDeBien= 'appartement'
9 group by nombrePieces
10 order by proportion desc
```

The results grid displays the following data:

nombrePieces	proportion
2	31.1978
3	28.5924
1	21.4905
4	14.2228
5	3.5525
6	0.6506
7	0.1722
0	0.0957
8	0.0542
9	0.0255
10	0.0064
11	0.0032

3. Requête 3

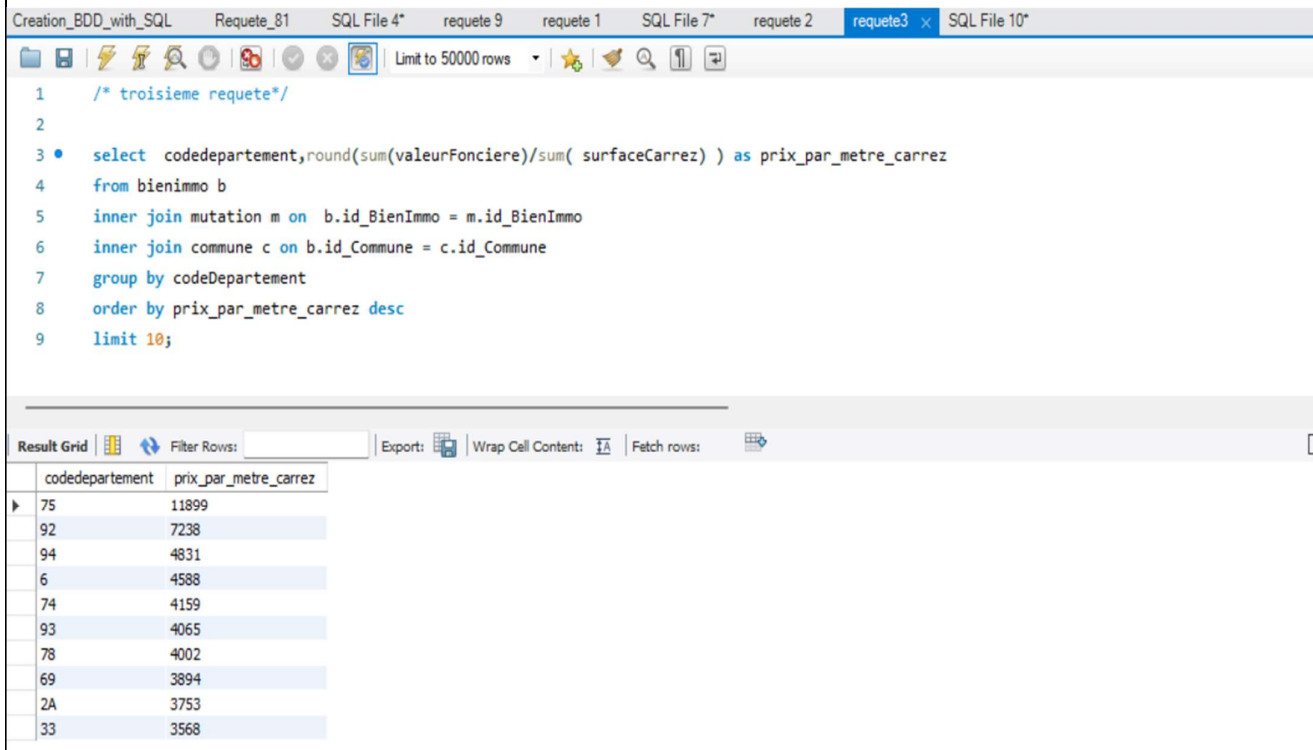
- Enoncé :

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

- Script :

```
select codedepartement,
round(sum(valeurFonciere)/sum( surfaceCarrez) ) as
prix_par_metre_carrez
from bienimmo b
inner join mutation m on b.id_BienImmo = m.id_BienImmo
inner join commune c on b.id_Commune = c.id_Commune
group by codeDepartement
order by prix_par_metre_carrez desc
limit 10;
```

- Résultat :



The screenshot shows a SQL client window with multiple tabs. The active tab is 'requete3'. The SQL editor contains the following query:

```
1 /* troisieme requete*/
2
3 • select codedepartement,round(sum(valeurFonciere)/sum( surfaceCarrez) ) as prix_par_metre_carrez
4 from bienimmo b
5 inner join mutation m on b.id_BienImmo = m.id_BienImmo
6 inner join commune c on b.id_Commune = c.id_Commune
7 group by codeDepartement
8 order by prix_par_metre_carrez desc
9 limit 10;
```

Below the editor, the 'Result Grid' shows the results of the query. The table has two columns: 'codedepartement' and 'prix_par_metre_carrez'. The results are as follows:

codedepartement	prix_par_metre_carrez
75	11899
92	7238
94	4831
6	4588
74	4159
93	4065
78	4002
69	3894
2A	3753
33	3568

4. Requête 4

- Enoncé :

Prix moyen du mètre carré d'une maison en Ile-de-France.

- Script :

```
/* quatrieme requete*/
select cast(avg(m.valeurFonciere)/ avg(b.surfacecarrez) as
decimal) as 'prix moyen'
from bienimmo b
inner join mutation m on m.id_BienImmo = b.id_BienImmo
inner join commune c on b.id_commune = c.id_Commune
where b.typeDeBien = 'maison' and c.codeDepartement in (
75,77,78,91,92,93,94,95);
```

- Résultat :

The screenshot shows a SQL IDE interface with multiple tabs: 'sql_creation_base_Immobilier', 'Creation_BDD_with_SQL', 'requete6', 'SQL File 4', 'SQL File 5*', and 'SQL File 12*'. The 'requete6' tab is active, displaying the following SQL query:

```
1 /* quatrieme requete*/
2 select cast(avg(valeurFonciere)/avg(surfacecarrez) as decimal) as 'prix moyen'
3 from bienimmo b
4 inner join mutation m on m.id_BienImmo = b.id_BienImmo
5 inner join commune c on b.id_commune = c.id_Commune
6 where typeDeBien = 'maison' and codeDepartement in ( 75,77,78,91,92,93,94,95);
```

Below the query editor, the 'Result Grid' is visible, showing the results of the query. The grid has two columns: 'prix' and 'moyen'. The first row shows the value '3676' under the 'moyen' column.

	prix	moyen
▶		3676

5. Requête 5

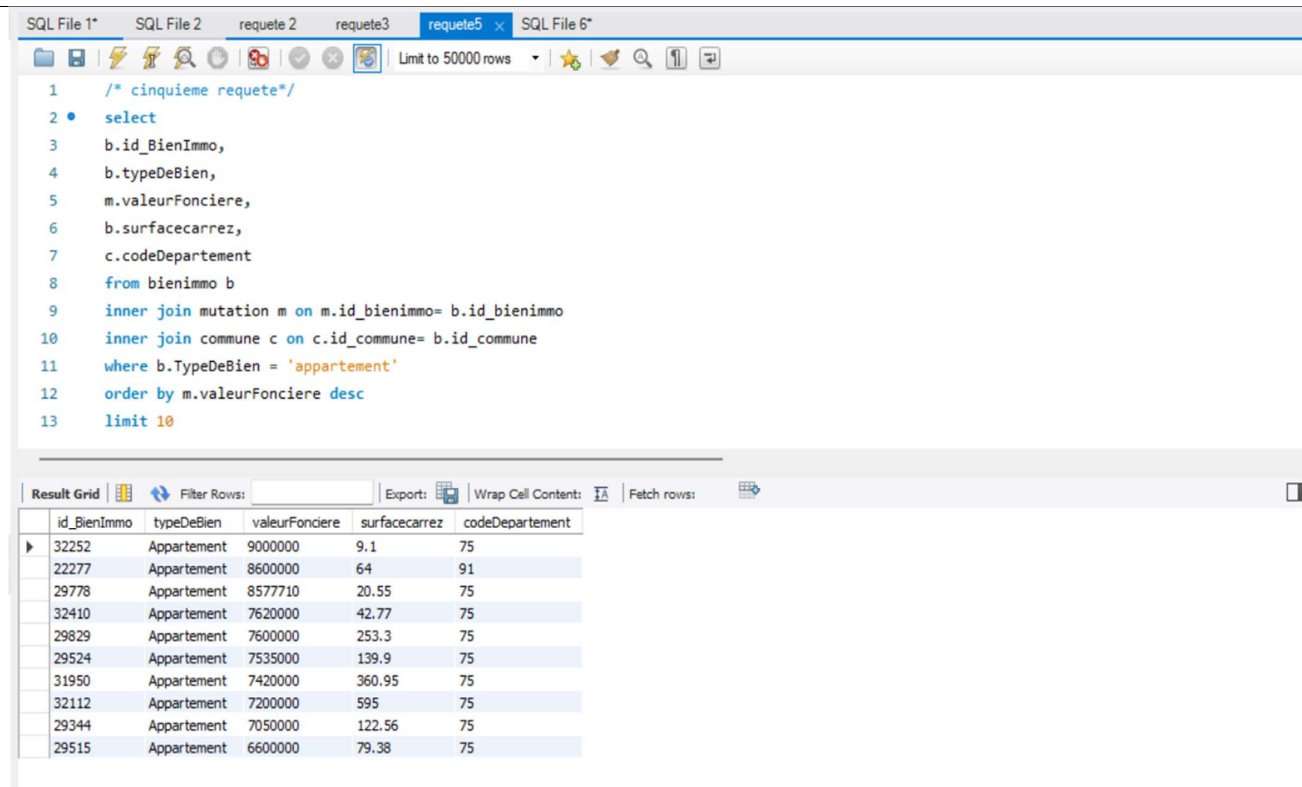
- Enoncé :

Liste des 10 appartements les plus chers avec le département et le nombre de mètres carrés.

- Script :

```
/* cinquieme requete*/  
select  
b.id_BienImmo, b.typeDeBien, m.valeurFonciere, b.surfacecarrez,  
c.codeDepartement from bienimmo b  
inner join mutation m on m.id_bienimmo= b.id_bienimmo  
inner join commune c on c.id_commune= b.id_commune  
where b.TypeDeBien = 'appartement'  
order by m.valeurFonciere desc  
limit 10 ;
```

- Résultat :



The screenshot shows a SQL IDE interface with multiple tabs. The active tab is 'requete5'. The SQL editor contains the following query:

```
1 /* cinquieme requete*/  
2 • select  
3 b.id_BienImmo,  
4 b.typeDeBien,  
5 m.valeurFonciere,  
6 b.surfacecarrez,  
7 c.codeDepartement  
8 from bienimmo b  
9 inner join mutation m on m.id_bienimmo= b.id_bienimmo  
10 inner join commune c on c.id_commune= b.id_commune  
11 where b.TypeDeBien = 'appartement'  
12 order by m.valeurFonciere desc  
13 limit 10
```

Below the editor, the 'Result Grid' is displayed, showing the results of the query. The grid has 5 columns: id_BienImmo, typeDeBien, valeurFonciere, surfacecarrez, and codeDepartement. The results are sorted by valeurFonciere in descending order.

id_BienImmo	typeDeBien	valeurFonciere	surfacecarrez	codeDepartement
32252	Appartement	9000000	9.1	75
22277	Appartement	8600000	64	91
29778	Appartement	8577710	20.55	75
32410	Appartement	7620000	42.77	75
29829	Appartement	7600000	253.3	75
29524	Appartement	7535000	139.9	75
31950	Appartement	7420000	360.95	75
32112	Appartement	7200000	595	75
29344	Appartement	7050000	122.56	75
29515	Appartement	6600000	79.38	75

6.Requête 6

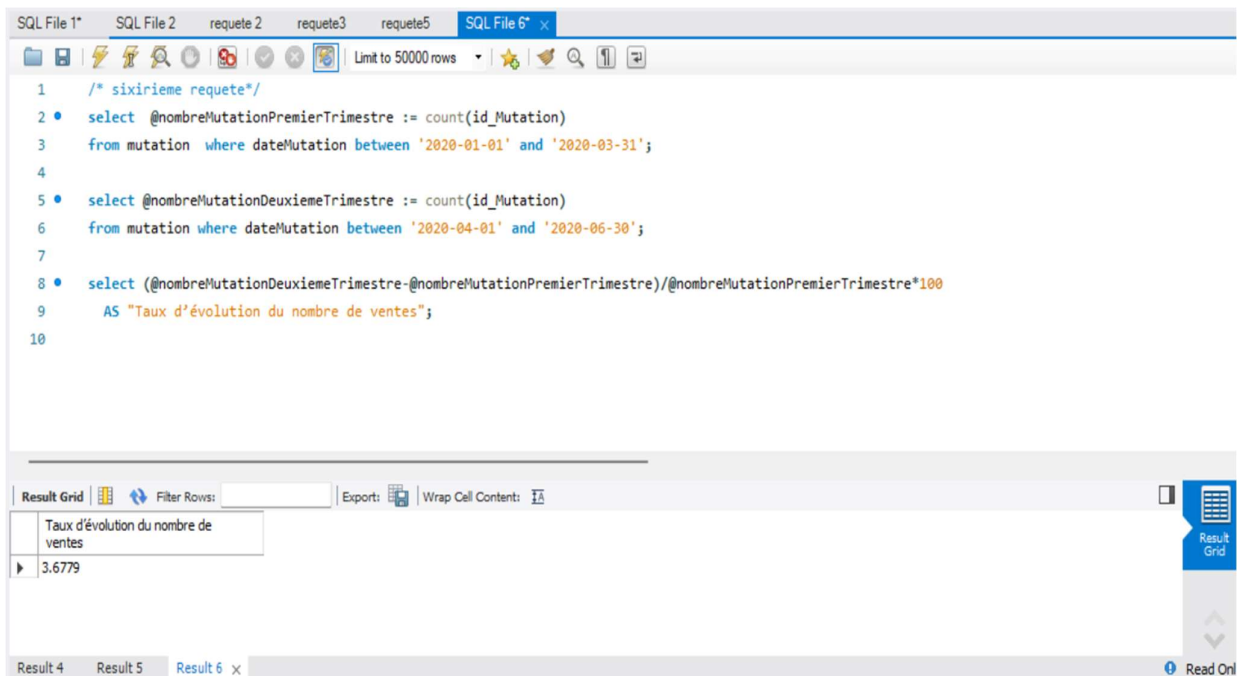
- Enoncé :

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

- Script :

```
/* sixieme requete*/
select @nombreMutationPremierTrimestre := count(id_Mutation)
from mutation where dateMutation between '2020-01-01' and '2020-03-31';
select @nombreMutationDeuxiemeTrimestre := count(id_Mutation)
from mutation where dateMutation between '2020-04-01' and '2020-06-30';
select (@nombreMutationDeuxiemeTrimestre-
@nombreMutationPremierTrimestre)/@nombreMutationPremierTrimestre*100
AS "Taux d'évolution du nombre de ventes";
```

- Résultat :



The screenshot shows a SQL IDE interface with a query editor and a results pane. The query editor contains the following SQL script:

```
1 /* sixieme requete*/
2 • select @nombreMutationPremierTrimestre := count(id_Mutation)
3   from mutation where dateMutation between '2020-01-01' and '2020-03-31';
4
5 • select @nombreMutationDeuxiemeTrimestre := count(id_Mutation)
6   from mutation where dateMutation between '2020-04-01' and '2020-06-30';
7
8 • select (@nombreMutationDeuxiemeTrimestre-@nombreMutationPremierTrimestre)/@nombreMutationPremierTrimestre*100
9   AS "Taux d'évolution du nombre de ventes";
10
```

The results pane shows a single row with the value 3.6779. The column header is "Taux d'évolution du nombre de ventes".

Taux d'évolution du nombre de ventes
3.6779

7. Requête 7

- Enoncé :

Liste des communes où le nombre de ventes a augmenté d'au moins 20% entre le premier et le second trimestre de 2020

- Script :

```
1  /* septieme requete*/
2  • drop table if exists requet71;
3  • create temporary table requet71 as
4    select count(*) as nombreDeVente1Trimestre,
5          b.id_commune, commune
6    from bienimmo b
7    inner join commune c on b.id_commune= c.id_commune
8    inner join mutation m on b.id_bienimmo= m.id_bienimmo
9    where m.dateMutation between '2020-01-01' and '2020-03-31' group by commune;
10
11 • drop table if exists requet72;
12 • create temporary table requet72 as
13   select count(*) as nombreDeVente2Trimestre,
14         b.id_commune, commune
15   from bienimmo b
16   inner join commune c on b.id_commune= c.id_commune
17   inner join mutation m on b.id_bienimmo= m.id_bienimmo
18   where m.dateMutation between '2020-04-01' and '2020-06-30' group by commune;
19
20 • select R1.commune,nombreDeVente1Trimestre, nombreDeVente2Trimestre,
21       round((nombreDeVente2Trimestre-nombreDeVente1Trimestre)/nombreDeVente1Trimestre*100) as tauxEvolutionVente
22   from requet71 R1
23   inner join requet72 R2 on
24     R1.id_commune= R2.id_commune
25   where ((nombreDeVente2Trimestre-nombreDeVente1Trimestre)/nombreDeVente1Trimestre)*100>20 order by commune
```

- Résultat :

```
1  /* septieme requete*/
2  • drop table if exists requet71;
3  • create temporary table requet71 as
4    select count(*) as nombreDeVente1Trimestre,
5          b.id_commune, commune
6    from bienimmo b
```

commune	nombreDeVente1Trimestre	nombreDeVente2Trimestre	tauxEvolutionVente
ABBEVILLE	2	9	350
ABLON-SUR-SEINE	5	13	160
AGDE	22	45	105
AIGUES-MORTES	6	11	83
AIRE-SUR-L'ADOUR	1	5	400
ALENCON	1	2	100
AMPUIS	1	2	100
ANCENIS-SAINT-GEREON	1	2	100
ANDRESY	6	8	33
ANGLET	2	4	100
ANNECY	28	42	50
ANNEMASSE	5	10	100
APT	2	7	250
ARCUEIL	8	19	138
ARDON	3	4	33
ARGENTEUIL	29	49	69
ARMENTIERES	4	9	125
ARNAS	1	2	100
ARPAJON	9	20	122
AUBERGENVILLE	1	3	200
AUBERVILLIERS	25	37	48
AUCAMVILLE	2	3	50
AUCH	1	3	200
AULT	3	4	33

8.Requête 8

- Enoncé :

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

- Script :

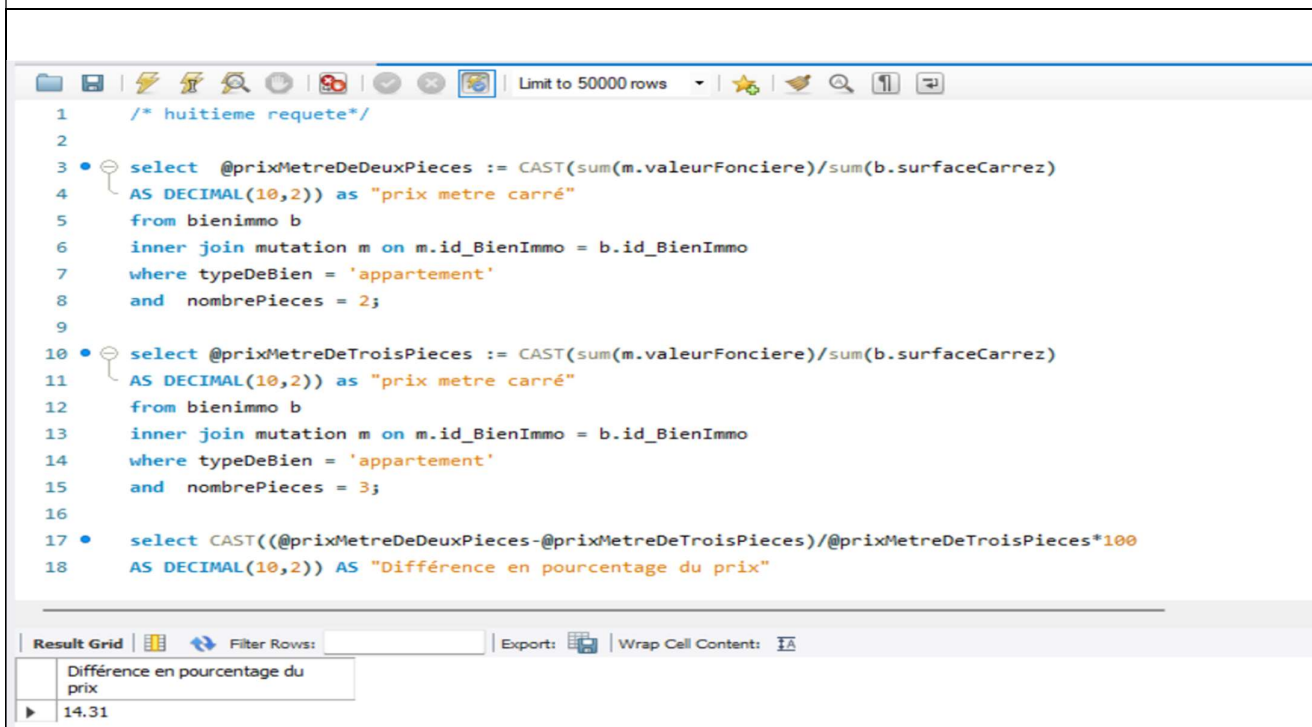
```
/* huitieme requete*/

select  @prixMetreDeDeuxPieces :=
CAST(sum(m.valeurFonciere)/sum(b.surfaceCarrez)
AS DECIMAL(10,2)) as "prix metre carré"
from bienimmo b
inner join mutation m on m.id_BienImmo = b.id_BienImmo
where typeDeBien = 'appartement'
and  nombrePieces = 2;

select  @prixMetreDeTroisPieces :=
CAST(sum(m.valeurFonciere)/sum(b.surfaceCarrez)
AS DECIMAL(10,2)) as "prix metre carré"
from bienimmo b
inner join mutation m on m.id_BienImmo = b.id_BienImmo
where typeDeBien = 'appartement'
and  nombrePieces = 3;

select  CAST((@prixMetreDeDeuxPieces -
@prixMetreDeTroisPieces)/@prixMetreDeTroisPieces*100
AS DECIMAL(10,2)) AS "Différence en pourcentage du prix"
```

- Résultat :



The screenshot shows a SQL query editor with a toolbar at the top. The query is displayed in a text area with line numbers 1 through 18. The query is the same as the one in the previous block. Below the query, there is a 'Result Grid' section. It shows a single row with the column name 'Différence en pourcentage du prix' and the value '14.31'.

Différence en pourcentage du prix
14.31

9.Requête 9

- Enoncé:

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

- Script:

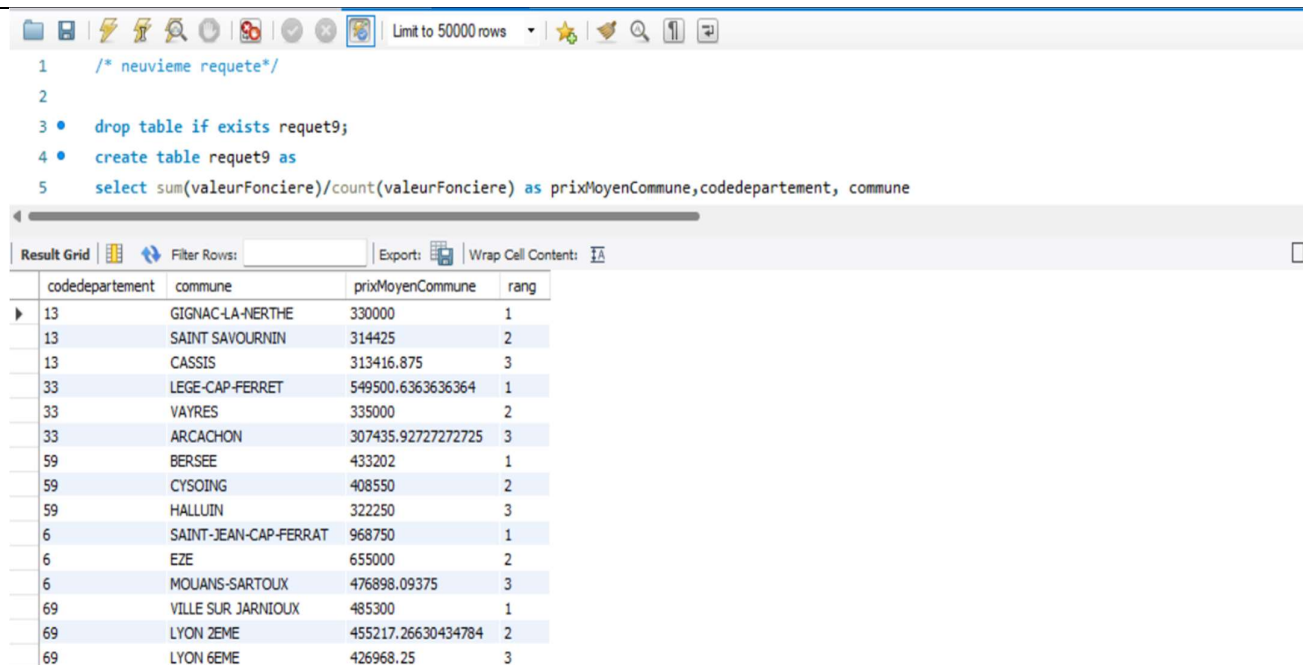
```
/* neuvieme requete*/

CREATE TEMPORARY TABLE requet9 as
select sum(valeurFonciere)/count(valeurFonciere) as
prixMoyenCommune,codedepartement, commune
from bienimmo b
inner join mutation m on m.id_bienimmo= b.id_bienimmo
inner join commune c on c.id_commune= b.id_commune
where codeDepartement in ( '6', '13', '33', '59', '69')
group by commune;

select codedepartement,commune, prixMoyenCommune, rang from
(
select prixMoyenCommune,codedepartement, commune,
rank() OVER ( partition by codedepartement ORDER BY
prixMoyenCommune desc) as rang
from requet9
) v
where rang<=3
order by codedepartement,rang;

drop table requet9;
```

- Résultat :



codedepartement	commune	prixMoyenCommune	rang
13	GIGNAC-LA-NERTHE	330000	1
13	SAINT SAVOURNIN	314425	2
13	CASSIS	313416.875	3
33	LEGE-CAP-FERRET	549500.6363636364	1
33	VAYRES	335000	2
33	ARCACHON	307435.92727272725	3
59	BERSEE	433202	1
59	CYSOING	408550	2
59	HALLUIN	322250	3
6	SAINT-JEAN-CAP-FERRAT	968750	1
6	EZE	655000	2
6	MOUANS-SARTOUX	476898.09375	3
69	VILLE SUR JARNIOUX	485300	1
69	LYON 2EME	455217.26630434784	2
69	LYON 6EME	426968.25	3