

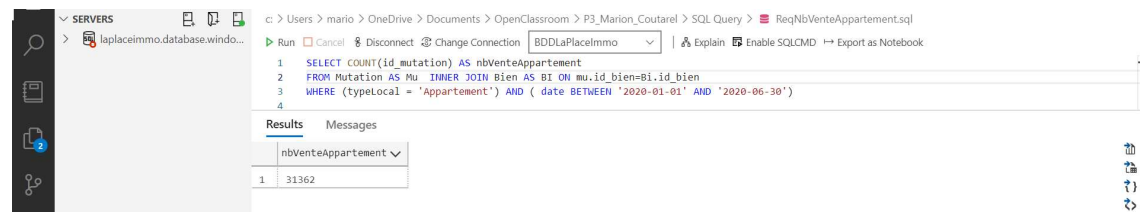


# DATA IMMO

Les requêtes



## Requête 1 : Nombre total d'appartements vendus au premier semestre



The screenshot shows a SQL query in SQL Server Enterprise Manager. The query is as follows:

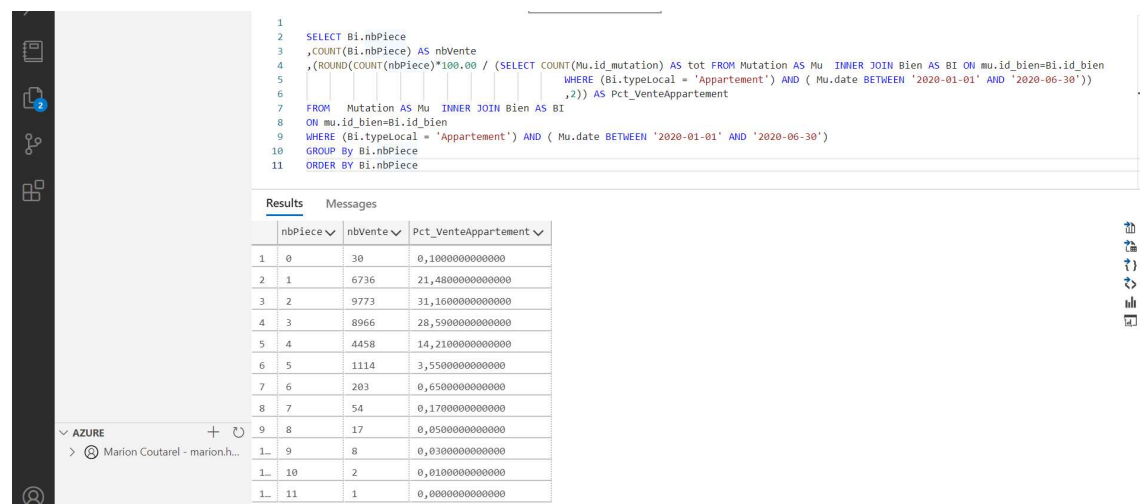
```
1 SELECT COUNT(id_mutation) AS nbVenteAppartement
2 FROM Mutation AS Mu JOIN Bien AS Bi ON Mu.id_bien=Bi.id_bien
3 WHERE (Bi.typeLocal = 'Appartement') AND (Mu.date BETWEEN '2020-01-01' AND '2020-06-30')
```

The results table shows a single row with the value 31362.

	nbVenteAppartement
1	31362

```
SELECT COUNT (id_mutation) AS nbVenteAppartement
FROM Mutation AS Mu JOIN Bien AS Bi ON Mu.id_bien=Bi.id_bien
WHERE (Bi.typeLocal = 'Appartement') AND (Mu.date BETWEEN '2020-01-01' AND '2020-06-30')
```

## Requête 2 : Proportion de ventes par nombre de pièces



The screenshot shows a SQL query in SQL Server Enterprise Manager. The query is as follows:

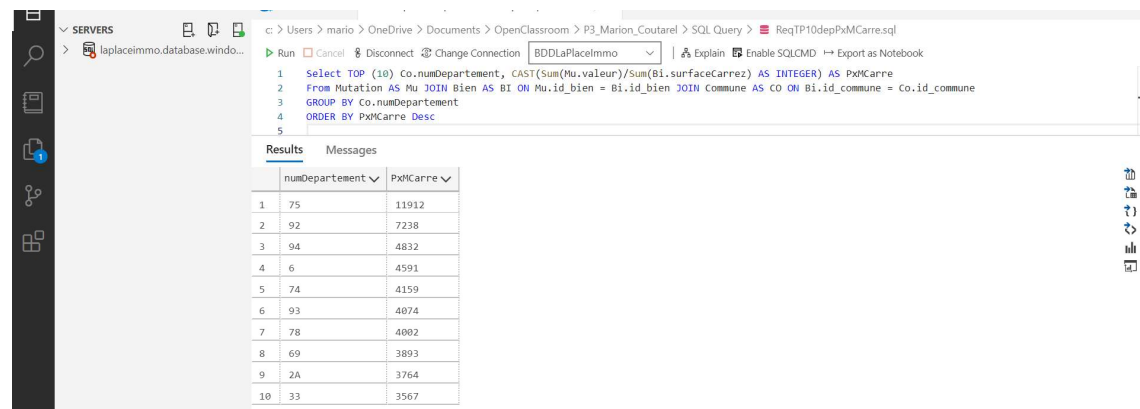
```
1 SELECT Bi.nbPiece
2 ,COUNT(Bi.nbPiece) AS nbVente
3 ,(ROUND(COUNT(nbPiece)*100.00 / (SELECT COUNT(Mu.id_mutation) AS tot FROM Mutation AS Mu JOIN Bien AS Bi ON mu.id_bien=Bi.id_bien
4 WHERE (Bi.typeLocal = 'Appartement') AND ( Mu.date BETWEEN '2020-01-01' AND '2020-06-30'))
5 ,2)) AS Pct_VenteAppartement
6 FROM Mutation AS Mu JOIN Bien AS Bi
7 ON mu.id_bien=Bi.id_bien
8 WHERE (Bi.typeLocal = 'Appartement') AND ( Mu.date BETWEEN '2020-01-01' AND '2020-06-30')
9 GROUP By Bi.nbPiece
10 ORDER BY Bi.nbPiece
```

The results table shows the following data:

	nbPiece	nbVente	Pct_VenteAppartement
1	0	30	0,1000000000000000
2	1	6736	21,4800000000000000
3	2	9773	31,1600000000000000
4	3	8966	28,5900000000000000
5	4	4458	14,2100000000000000
6	5	1114	3,5500000000000000
7	6	203	0,6500000000000000
8	7	54	0,1700000000000000
9	8	17	0,0500000000000000
10	9	8	0,0300000000000000
11	10	2	0,0100000000000000
12	11	1	0,0000000000000000

```
SELECT Bi.nbPiece
,COUNT(Bi.nbPiece) AS nbVente
,(ROUND(COUNT(nbPiece)*100.00 / (SELECT COUNT(Mu.id_mutation) AS tot FROM Mutation AS Mu JOIN Bien AS Bi ON
mu.id_bien=Bi.id_bien WHERE (Bi.typeLocal = 'Appartement') AND ( Mu.date BETWEEN '2020-01-01' AND '2020-06-30') ,2))
AS Pct_VenteAppartement
FROM Mutation AS Mu JOIN Bien AS Bi
ON Mu.id_bien=Bi.id_bien
WHERE (Bi.typeLocal = 'Appartement') AND ( Mu.date BETWEEN '2020-01-01' AND '2020-06-30')
GROUP By Bi.nbPiece
ORDER BY Bi.nbPiece
```

### Requête 3 : Liste des 10 départements où le prix au m<sup>2</sup> est le plus élevé



The screenshot shows a SQL query window with the following query:

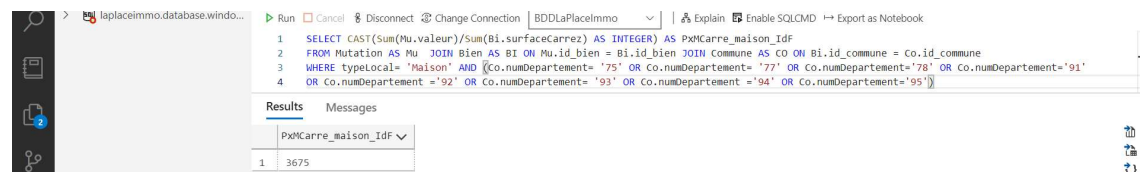
```
1 Select TOP (10) Co.numDepartement, CAST(Sum(Mu.valeur)/Sum(Bi.surfaceCarrez) AS INTEGER) AS PxMCarre
2 From Mutation AS Mu JOIN Bien AS Bi ON Mu.id_bien = Bi.id_bien JOIN Commune AS Co ON Bi.id_commune = Co.id_commune
3 GROUP BY Co.numDepartement
4 ORDER BY PxMCarre Desc
5
```

The results are displayed in a table with two columns: numDepartement and PxMCarre.

	numDepartement	PxMCarre
1	75	11912
2	92	7238
3	94	4832
4	6	4591
5	74	4159
6	93	4074
7	78	4002
8	69	3893
9	2A	3764
10	33	3567

Select TOP (10) Co.numDepartement, CAST(Sum(Mu.valeur)/Sum(Bi.surfaceCarrez) AS INTEGER) AS PxMCarre  
From Mutation AS Mu JOIN Bien AS Bi ON Mu.id\_bien = Bi.id\_bien JOIN Commune AS Co ON Bi.id\_commune =  
Co.id\_commune  
GROUP BY Co.numDepartement  
ORDER BY PxMCarre Desc

### Requête 4 : Prix moyen du m<sup>2</sup> d'une maison en Ile de France



The screenshot shows a SQL query window with the following query:

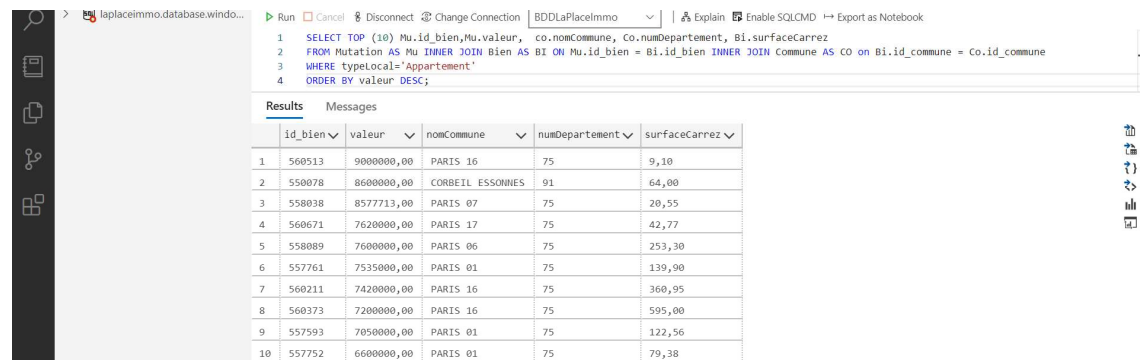
```
1 SELECT CAST(Sum(Mu.valeur)/Sum(Bi.surfaceCarrez) AS INTEGER) AS PxMCarre_maison_IdF
2 FROM Mutation AS Mu JOIN Bien AS Bi ON Mu.id_bien = Bi.id_bien JOIN Commune AS Co ON Bi.id_commune = Co.id_commune
3 WHERE typeLocal= 'Maison' AND ((Co.numDepartement= '75' OR Co.numDepartement= '77' OR Co.numDepartement='78' OR Co.numDepartement='91'
4 OR Co.numDepartement = '92' OR Co.numDepartement= '93' OR Co.numDepartement = '94' OR Co.numDepartement='95'))
```

The results are displayed in a table with one column: PxMCarre\_maison\_IdF.

	PxMCarre_maison_IdF
1	3675

SELECT CAST(Sum(Mu.valeur)/Sum(Bi.surfaceCarrez) AS INTEGER) AS PxMCarre\_maison\_IdF  
FROM Mutation AS Mu JOIN Bien AS Bi ON Mu.id\_bien = Bi.id\_bien JOIN Commune AS Co ON Bi.id\_commune =  
Co.id\_commune  
WHERE typeLocal= 'Maison' AND (Co.numDepartement= '75' OR Co.numDepartement= '77' OR Co.numDepartement='78' OR  
Co.numDepartement='91' OR Co.numDepartement = '92' OR Co.numDepartement= '93' OR Co.numDepartement = '94' OR  
Co.numDepartement='95')

**Requête 5 :** Liste des 10 appartements les plus chers avec le département et le nombre de mètre carrés :



The screenshot shows a SQL query in the 'Query Editor' window of SQL Server Enterprise Manager. The query is as follows:

```

1 SELECT TOP (10) Mu.id_bien,Mu.valeur, co.nomCommune, Co.numDepartement, Bi.surfaceCarrez
2 FROM Mutation AS Mu INNER JOIN Bien AS BI ON Mu.id_bien = Bi.id_bien INNER JOIN Commune AS CO on Bi.id_commune = Co.id_commune
3 WHERE typeLocal='Appartement'
4 ORDER BY valeur DESC;

```

The 'Results' pane displays the following data:

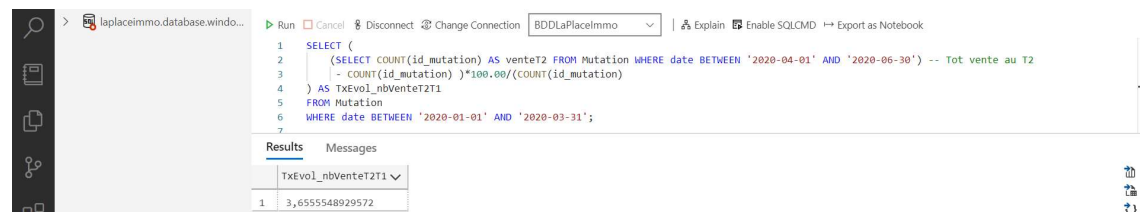
	id_bien	valeur	nomCommune	numDepartement	surfaceCarrez
1	560513	9000000,00	PARIS 16	75	9,10
2	550078	8600000,00	CORBEIL ESSONNES	91	64,00
3	558038	8577713,00	PARIS 07	75	20,55
4	560671	7620000,00	PARIS 17	75	42,77
5	558089	7600000,00	PARIS 06	75	253,30
6	557761	7535000,00	PARIS 01	75	139,90
7	560211	7420000,00	PARIS 16	75	360,95
8	560373	7200000,00	PARIS 16	75	595,00
9	557593	7050000,00	PARIS 01	75	122,56
10	557752	6600000,00	PARIS 01	75	79,38

```

SELECT TOP (10) Mu.id_bien,Mu.valeur, co.nomCommune, Co.numDepartement, Bi.surfaceCarrez
FROM Mutation AS Mu INNER JOIN Bien AS BI ON Mu.id_bien = Bi.id_bien INNER JOIN Commune AS CO on Bi.id_commune
= Co.id_commune
WHERE typeLocal='Appartement'
ORDER BY valeur DESC;

```

**Requête 6 :** Taux d'évolution du nombre de ventes entre le premier et le deuxième trimestre 2020 :



The screenshot shows a SQL query in the 'Query Editor' window of SQL Server Enterprise Manager. The query is as follows:

```

1 SELECT (
2     (SELECT COUNT(id_mutation) AS venteT2 FROM Mutation WHERE date BETWEEN '2020-04-01' AND '2020-06-30') -- Tot vente au T2
3     - COUNT(id_mutation) ) * 100.00 / (COUNT(id_mutation)
4 ) AS TxEvol_nbVenteT2T1
5 FROM Mutation
6 WHERE date BETWEEN '2020-01-01' AND '2020-03-31';
7

```

The 'Results' pane displays the following data:

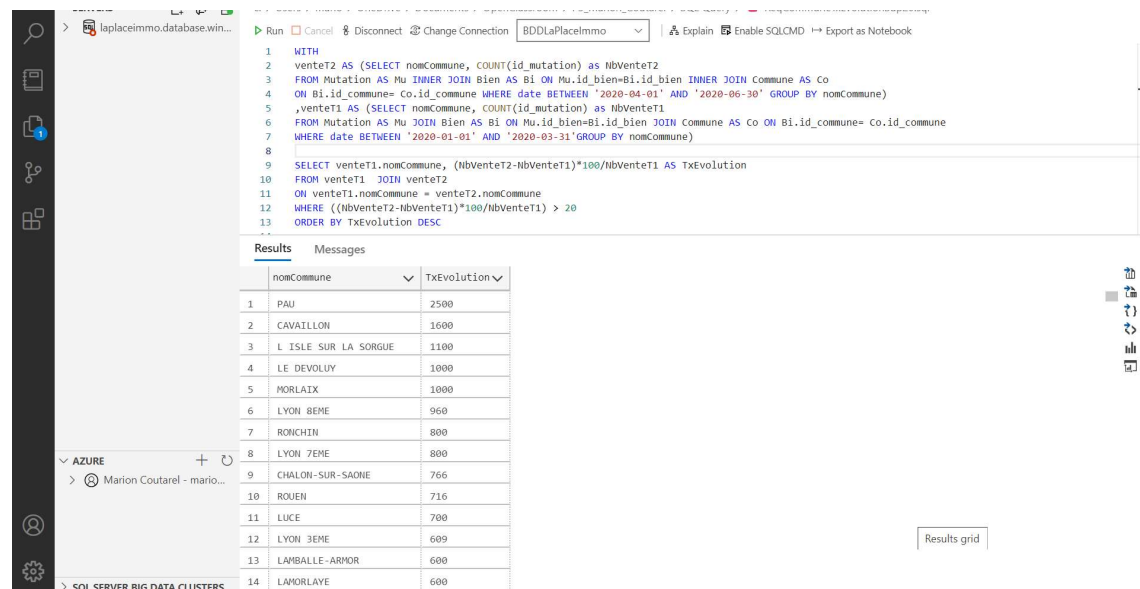
	TxEvol_nbVenteT2T1
1	3,6555548929572

```

SELECT (
    (SELECT COUNT(id_mutation) AS venteT2 FROM Mutation WHERE date BETWEEN '2020-04-01' AND '2020-06-30') -- Tot
vente au T2
    - COUNT(id_mutation) ) * 100.00 / (COUNT(id_mutation)
) AS TxEvol_nbVenteT2T1
FROM Mutation
WHERE date BETWEEN '2020-01-01' AND '2020-03-31';

```

**Requête 7 :** Liste des communes où le taux d'évolution des ventes est supérieur à 20 % entre le premier et le second semestres de 2020 :



The screenshot shows a SQL query execution interface. The query is as follows:

```

1 WITH
2 venteT2 AS (SELECT nomCommune, COUNT(id_mutation) as NbVenteT2
3 FROM Mutation AS Mu INNER JOIN Bien AS Bi ON Mu.id_bien=Bi.id_bien INNER JOIN Commune AS Co
4 ON Bi.id_commune= Co.id_commune WHERE date BETWEEN '2020-04-01' AND '2020-06-30' GROUP BY nomCommune)
5 ,venteT1 AS (SELECT nomCommune, COUNT(id_mutation) as NbVenteT1
6 FROM Mutation AS Mu JOIN Bien AS Bi ON Mu.id_bien=Bi.id_bien JOIN Commune AS Co ON Bi.id_commune= Co.id_commune
7 WHERE date BETWEEN '2020-01-01' AND '2020-03-31' GROUP BY nomCommune)
8
9 SELECT venteT1.nomCommune, (NbVenteT2-NbVenteT1)*100/NbVenteT1 AS TxEvolution
10 FROM venteT1 JOIN venteT2
11 ON venteT1.nomCommune = venteT2.nomCommune
12 WHERE ((NbVenteT2-NbVenteT1)*100/NbVenteT1) > 20
13 ORDER BY TxEvolution DESC

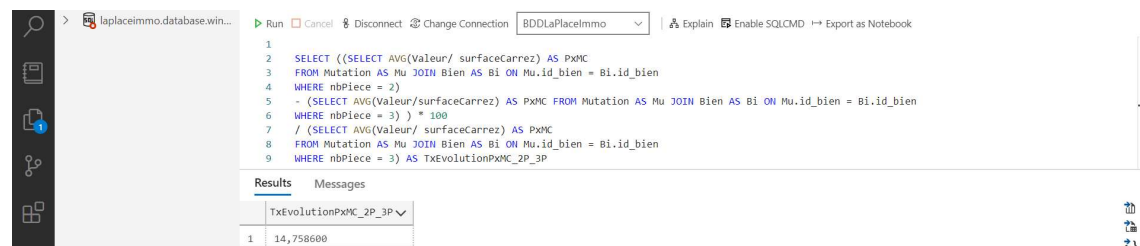
```

The results are displayed in a table with two columns: nomCommune and TxEvolution. The results are as follows:

nomCommune	TxEvolution
1 PAU	2500
2 CAVAILLON	1600
3 L ISLE SUR LA SORGUE	1100
4 LE DEVOLUY	1000
5 MORLAIX	1000
6 LYON BEME	960
7 RONCHIN	800
8 LYON 7EME	800
9 CHALON-SUR-SAONE	766
10 ROUEN	716
11 LUCE	700
12 LYON 3EME	609
13 LAMBALLE-ARMOR	600
14 LANORLAYE	600

WITH  
venteT2 AS (SELECT nomCommune, COUNT(id\_mutation) as NbVenteT2  
FROM Mutation AS Mu INNER JOIN Bien AS Bi ON Mu.id\_bien=Bi.id\_bien INNER JOIN Commune AS Co  
ON Bi.id\_commune= Co.id\_commune WHERE date BETWEEN '2020-04-01' AND '2020-06-30' GROUP BY nomCommune)  
,venteT1 AS (SELECT nomCommune, COUNT(id\_mutation) as NbVenteT1  
FROM Mutation AS Mu JOIN Bien AS Bi ON Mu.id\_bien=Bi.id\_bien JOIN Commune AS Co ON Bi.id\_commune=  
Co.id\_commune  
WHERE date BETWEEN '2020-01-01' AND '2020-03-31' GROUP BY nomCommune)  
SELECT venteT1.nomCommune, (NbVenteT2-NbVenteT1)\*100/NbVenteT1 AS TxEvolution  
FROM venteT1 JOIN venteT2  
ON venteT1.nomCommune = venteT2.nomCommune  
WHERE ((NbVenteT2-NbVenteT1)\*100/NbVenteT1) > 20  
ORDER BY TxEvolution DESC

**Requête 8 :** 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 query in a database client. The query calculates the percentage difference in price per square meter between 2-room and 3-room apartments. The results table shows a value of 14,758600.

```

1  SELECT ((SELECT AVG(Valeur/ surfaceCarrez) AS PxMC
2  FROM Mutation AS Mu JOIN Bien AS Bi ON Mu.id_bien = Bi.id_bien
3  WHERE nbPiece = 2)
4  - (SELECT AVG(Valeur/surfaceCarrez) AS PxMC FROM Mutation AS Mu JOIN Bien AS Bi ON Mu.id_bien = Bi.id_bien
5  WHERE nbPiece = 3) ) * 100
6  / (SELECT AVG(Valeur/ surfaceCarrez) AS PxMC
7  FROM Mutation AS Mu JOIN Bien AS Bi ON Mu.id_bien = Bi.id_bien
8  WHERE nbPiece = 3) AS TxEvolutionPxMC_2P_3P

```

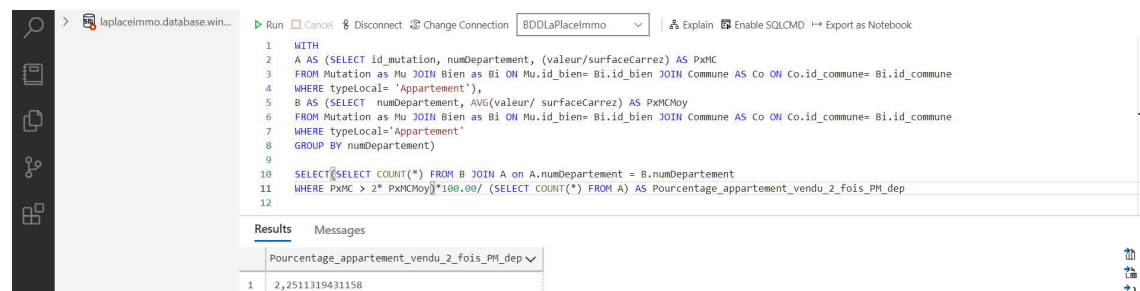
Results	Messages
TxEvolutionPxMC_2P_3P	
1	14,758600

```

SELECT ((SELECT AVG(Valeur/ surfaceCarrez) AS PxMC
FROM Mutation AS Mu JOIN Bien AS Bi ON Mu.id_bien = Bi.id_bien
WHERE nbPiece = 2)
- (SELECT AVG(Valeur/surfaceCarrez) AS PxMC FROM Mutation AS Mu JOIN Bien AS Bi ON Mu.id_bien = Bi.id_bien
WHERE nbPiece = 3) ) * 100
/ (SELECT AVG(Valeur/ surfaceCarrez) AS PxMC
FROM Mutation AS Mu JOIN Bien AS Bi ON Mu.id_bien = Bi.id_bien
WHERE nbPiece = 3) AS TxEvolutionPxMC_2P_3P

```

**Requête 9 :** Taux d'appartements qui ont été vendus à un prix du mètre carré deux fois plus élevé que le prix du mètre carré moyen du département :



The screenshot shows a SQL query in a database client. The query calculates the percentage of apartments sold at a price per square meter at least twice the departmental average. The results table shows a value of 2,2511319431158.

```

1  WITH
2  A AS (SELECT id_mutation, numDepartement, (valeur/surfaceCarrez) AS PxMC
3  FROM Mutation as Mu JOIN Bien as Bi ON Mu.id_bien= Bi.id_bien JOIN Commune AS Co ON Co.id_commune= Bi.id_commune
4  WHERE typeLocal= 'Appartement'),
5  B AS (SELECT numDepartement, AVG(valeur/ surfaceCarrez) AS PxMCMoy
6  FROM Mutation as Mu JOIN Bien as Bi ON Mu.id_bien= Bi.id_bien JOIN Commune AS Co ON Co.id_commune= Bi.id_commune
7  WHERE typeLocal='Appartement'
8  GROUP BY numDepartement)
9
10 SELECT(SELECT COUNT(*) FROM B JOIN A on A.numDepartement = B.numDepartement
11 WHERE PxMC > 2* PxMCMoy)*100.00/ (SELECT COUNT(*) FROM A) AS Pourcentage_appartement_vendu_2_fois_PM_dep
12

```

Results	Messages
Pourcentage_appartement_vendu_2_fois_PM_dep	
1	2,2511319431158

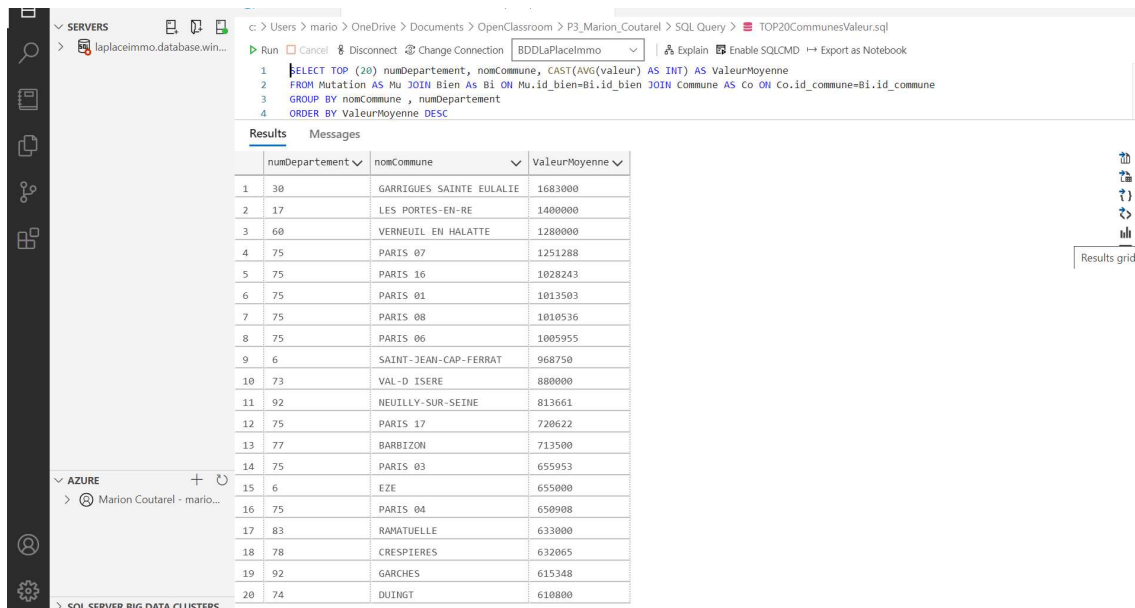
```

WITH
A AS (SELECT id_mutation, numDepartement, (valeur/surfaceCarrez) AS PxMC
FROM Mutation as Mu JOIN Bien as Bi ON Mu.id_bien= Bi.id_bien JOIN Commune AS Co ON Co.id_commune=
Bi.id_commune
WHERE typeLocal= 'Appartement'),
B AS (SELECT numDepartement, AVG(valeur/ surfaceCarrez) AS PxMCMoy
FROM Mutation as Mu JOIN Bien as Bi ON Mu.id_bien= Bi.id_bien JOIN Commune AS Co ON Co.id_commune=
Bi.id_commune
WHERE typeLocal='Appartement'
GROUP BY numDepartement)

SELECT(SELECT COUNT(*) FROM B JOIN A on A.numDepartement = B.numDepartement
WHERE PxMC > 2* PxMCMoy)*100.00/ (SELECT COUNT(*) FROM A) AS Pourcentage_appartement_vendu_2_fois_PM_dep

```

## Requête 10 : Donnez les moyennes de valeurs foncières pour le top 20 des communes :



The screenshot shows the SQL Server Enterprise Manager interface. The query editor displays the following SQL query:

```

1 SELECT TOP (20) numDepartement, nomCommune, CAST(AVG(valeur) AS INT) AS ValeurMoyenne
2 FROM Mutation AS Mu JOIN Bien As Bi ON Mu.id_bien=Bi.id_bien JOIN Commune AS Co ON Co.id_commune=Bi.id_commune
3 GROUP BY nomCommune , numDepartement
4 ORDER BY ValeurMoyenne DESC

```

The results grid shows the top 20 communes by average value:

	numDepartement	nomCommune	ValeurMoyenne
1	30	GARRIGUES SAINTE EULALIE	1683000
2	17	LES PORTES-EN-RE	1400000
3	60	VERNEUIL EN HALATTE	1280000
4	75	PARIS 07	1251288
5	75	PARIS 16	1028243
6	75	PARIS 01	1013503
7	75	PARIS 08	1010536
8	75	PARIS 06	1005955
9	6	SAINT-JEAN-CAP-FERRAT	968750
10	73	VAL-D ISERE	880000
11	92	NEUILLY-SUR-SEINE	813661
12	75	PARIS 17	720622
13	77	BARBIZON	713500
14	75	PARIS 03	655953
15	6	EZE	655000
16	75	PARIS 04	650908
17	83	RAMATUELLE	633000
18	78	CRESPIERES	632065
19	92	GARCHES	615348
20	74	DUINGT	610800

```

SELECT TOP (20) numDepartement, nomCommune, CAST(AVG(valeur) AS INT) AS ValeurMoyenne
FROM Mutation AS Mu JOIN Bien As Bi ON Mu.id_bien=Bi.id_bien JOIN Commune AS Co ON
Co.id_commune=Bi.id_commune
GROUP BY nomCommune , numDepartement
ORDER BY ValeurMoyenne DESC

```