



# **PROJET DATAIMMO CRÉATION ET UTILISATION DE LA BASE DE DONNÉES**



Laplace Immo

# CONTEXTE DU PROJET

But : création d'un modèle permettant une meilleure prévision des prix de vente des biens immobiliers

Modification de la base de données permettant de collecter les transactions immobilières et foncières en France



# LA STRATÉGIE DE SAUVEGARDE

## Stratégie 3-2-1:

Au moins trois copies des données dans des endroits différents, deux copies sur des supports différents et une copie hors site

Cette stratégie vise à diversifier les lieux de stockage des sauvegardes afin de garantir la protection des données contre tout type de situation de perte de données

# LA CONFORMITÉ RGPD

Dans notre base de données, on n'a pas mentionné des données nominatives  
La colonne acquéreur n'a pas été reprise dans la base de données

Les sites insee.fr, data.gouv.fr utilisés dans notre étude présente une politique de protection des données personnelles

Les données en rapport avec les valeurs foncières sont rendues ouvertes au grand public depuis 2019

la finalité de notre étude est en conformité avec le RGPD

# Protection des données personnelles

L'INSEE ET LA STATISTIQUE PUBLIQUE

Date de publication : 08/01/2024

[> Imprimer](#)

L'Insee s'engage à ce que les traitements de données personnelles qu'il met en œuvre à des fins statistiques soient conformes au **Règlement général sur la protection des données (RGPD)** [🔗](#) et à la **loi Informatique et Libertés** [🔗](#).

## Politique de protection des données

Chaque traitement limite la collecte des données personnelles au strict nécessaire (minimisation des données) et s'accompagne d'une information sur :

- les objectifs du recueil de ces données (finalités) ;
- la base juridique de traitement ;
- le caractère obligatoire ou facultatif du recueil des données et le rappel des catégories de données traitées ;
- la source des données ;
- les catégories de personnes concernées ;
- les destinataires des données ;
- la durée de conservation des données ;
- les mesures de sécurité (description générale) ;
- l'existence éventuelle de transferts de données hors de l'Union européenne ou de prises de décision automatisées ;
- **les droits Informatique et Libertés** [🔗](#) et la façon de les exercer auprès de l'Insee.

## S'agissant de la réutilisation de données publiquement accessibles sur Internet :

- Lorsqu'une personne décide de réutiliser des données sur Internet et détermine, par là même, la finalité (l'objet de la réutilisation) et la caractéristique principale du traitement (utiliser des données publiées sur Internet) : elle est en principe responsable de traitement, y compris si elle recourt à un prestataire. Cela est valable que les données aient ou non été initialement diffusées en open data.

Il en va ainsi de la personne qui décide de constituer une base de données à partir de données publiquement accessibles pour l'exploiter commercialement auprès de différents clients. Cette personne et ses clients pourraient alors être qualifiés de responsables de traitement distincts, à condition que chaque traitement puisse être séparé de l'autre (et que leurs finalités et moyens ne soient pas déterminés conjointement comme cela est détaillé ci-dessous).

- Lorsqu'une personne confie une mission à un prestataire sans lui en imposer les moyens et que c'est ce prestataire qui choisit, pour atteindre l'objectif, de réutiliser des données personnelles publiées sur Internet, c'est en principe le prestataire qui doit être qualifié de responsable du traitement.

### Focus : qu'est-ce qu'une finalité ?

La finalité est le but précis pour lequel les données sont traitées. Elle doit être déterminée, explicite et légitime.

Exemples :

- il peut s'agir de la publication des données dans un annuaire public à des fins d'information du public.
- il peut s'agir de la réutilisation de données publiquement accessibles à des fins de prospection commerciale entre professionnels.

obligations de réutilisation de données

Extrait du guide CNIL pour les utilisateurs de l'open data

### Quelles sont les obligations du réutilisateur de données ?

Lorsqu'un individu réutilise un jeu de données publiques, il est tenu de respecter les conditions de la licence sous laquelle les données publiques ont initialement été publiées. Deux principales licences sont utilisées dans ce cadre, la [Licence ouverte 2.0 - Licence Etalab](#) ou la Licence ODbL. Dans le cas d'une réutilisation de données publiées sous **licence ouverte 2.0**, vous êtes tenu(e) de :

- **Mentionner la source des données ;**
- **Mentionner la date de dernière mise à jour de la réutilisation ;**
- **Ne pas altérer le sens des données.**

Dans le cas de données publiées sous une **licence ODbL**, vous êtes tenu(e) de respecter l'ensemble des conditions fixées par la Licence ouverte 2.0 précédemment mentionnées tout en repartageant votre réutilisation sous licence ODbL. Cette **clause de partage à l'identique** concrétise la logique *share alike*.

- ④ **À défaut de mention d'une licence**, les dispositions de l'article L322-1 du CRPA s'appliquent. Cet article fixe des conditions de réutilisation identiques à celles de la licence ouverte, à savoir : la **non-altération** des données publiques, la **mention de leurs sources** (paternité des données) et la **mention de la date de leur dernière mise à jour**.

# LES DONNÉES INITIALES

Table valeurs foncières: Valeur du bien, nombre de lots, numéro de disposition, nature mutation, surface carrez....

Table référentiel géographique: ancien et nouveau nom de la région, colonnes en rapport avec les unités urbaines, les coordonnées géographiques...

Table données communes: code région et département, nom et population des communes

Noms des acquéreurs: protection des données personnelles, données non reprises...

Qualité des données ex: Valeurs foncières nulles



# **L'EXTRAIT DU DICTIONNAIRE DES DONNÉES**



## DICTIONNAIRE DES DONNÉES - Valeurs foncières

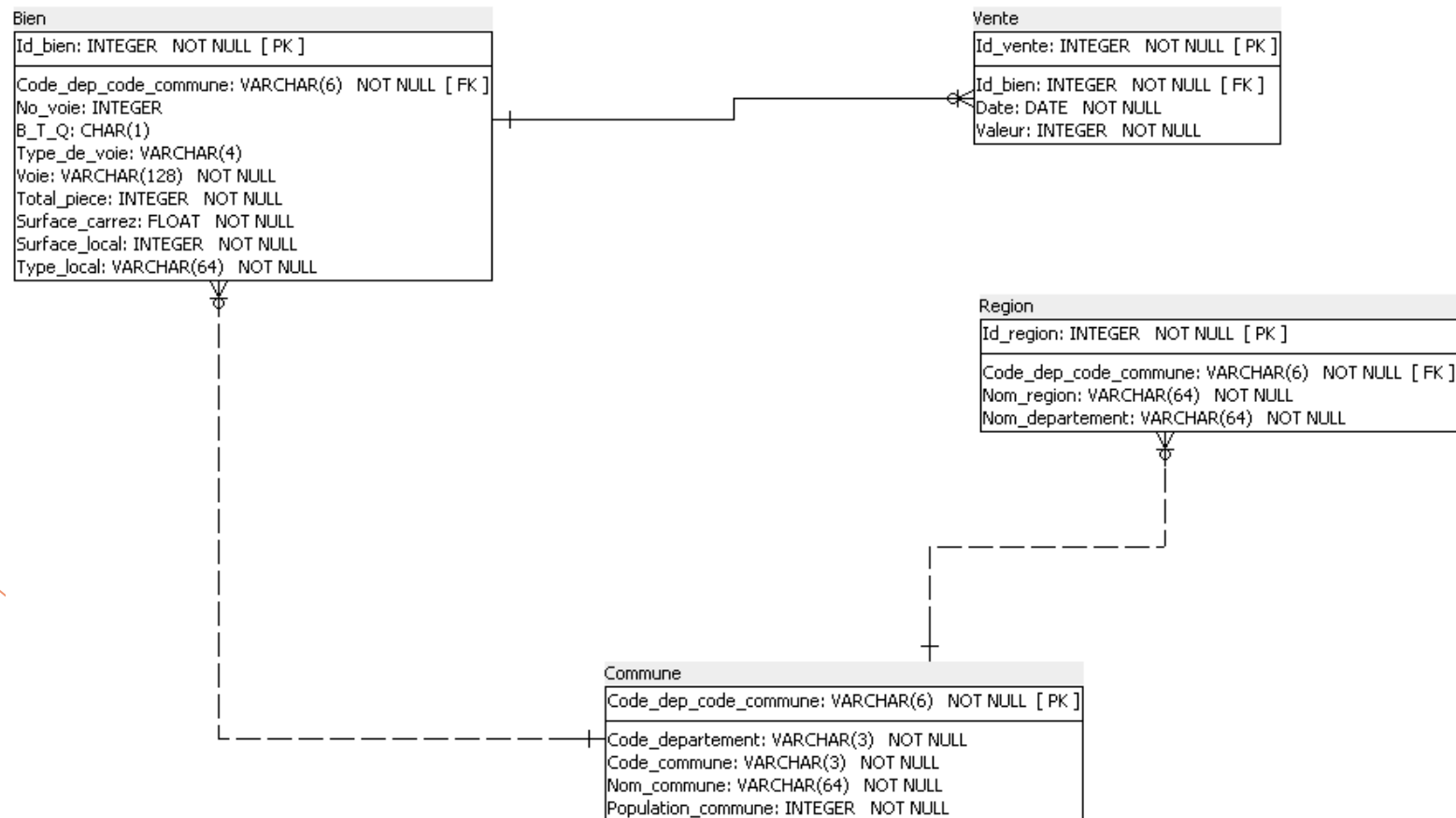
| CODE                      | SIGNIFICATION                                 | TYPE    | LONGUEUR | NATURE      | REGLE DE GESTION     | REGLE DE CALCUL             |
|---------------------------|-----------------------------------------------|---------|----------|-------------|----------------------|-----------------------------|
| Id_bien                   | ID dans la base de données                    | Integer | NC       | Elémentaire | Ne doit pas être nul |                             |
| Date_mutation             | Date de vente du bien                         | Date    |          | Elémentaire | Ne doit pas être nul | Format de date (YYYY/mm/dd) |
| Valeur_fonciere           | Valeur foncière du bien en euros              | Integer | NC       | Elémentaire |                      |                             |
| No_voie                   | Numéro des rues                               | Integer | NC       | Elémentaire |                      |                             |
| B_T_Q                     | Indice de répétition                          | Char    | 1        | Elémentaire |                      |                             |
| Type_de_voie              | Plusieurs valeurs (rue, avenue, chemin, etc.) | Varchar | 4        | Elémentaire |                      |                             |
| Voie                      | Nom de la rue                                 | Varchar | 50       | Elémentaire | Ne doit pas être nul |                             |
| Code_postale              | Code postale où se trouve le bien             | Varchar | 5        | Elémentaire | Ne doit pas être nul |                             |
| Nom_commune               | Nom de la commune où se trouve le bien        | Varchar | 64       | Elémentaire | Ne doit pas être nul |                             |
| Code_departement          | Code département où se trouve le bien         | Varchar | 3        | Elémentaire | Ne doit pas être nul |                             |
| Code_commune              | Code commune où se trouve le bien             | Varchar | 3        | Elémentaire | Ne doit pas être nul |                             |
| Nombre_pieces_principales | Nombre total de pièces du bien                | Integer | NC       | Elémentaire | Ne doit pas être nul |                             |
| Surface_carrez            | Surface habitable (loi carrez) du bien en m2  | Float   | NC       | Elémentaire | Ne doit pas être nul |                             |
| Surface_reelle_bati       | Surface réelle du bien en m2                  | Integer | NC       | Elémentaire | Ne doit pas être nul |                             |
| Type_local                | Type de bien (appartement ou maison)          | Varchar | 64       | Elémentaire | Ne doit pas être nul |                             |

## DICTIONNAIRE DES DONNÉES - Référentiel géographique

| CODE              | SIGNIFICATION                                          | TYPE    | LONGUEUR | NATURE      | REGLE DE GESTION     | REGLE DE CALCUL          |
|-------------------|--------------------------------------------------------|---------|----------|-------------|----------------------|--------------------------|
| Com_code          | Concaténation du code département avec le code commune | Varchar | 6        | Concaténer  | Ne doit pas être nul | Code dep    Code commune |
| Reg_nom           | Nom de la région                                       | Varchar | 64       | Elémentaire | Ne doit pas être nul |                          |
| Dep_nom           | Nom du département                                     | Varchar | 64       | Elémentaire | Ne doit pas être nul |                          |
| Com_nom_maj_court | Nom de la commune en majuscule                         | Varchar | 64       | Elémentaire | Ne doit pas être nul |                          |
| Aca_nom           | Nom de l'académie                                      | Varchar | 32       | Elémentaire | Ne doit pas être nul |                          |
| Dep_code          | Code département                                       | Varchar | 3        | Elémentaire | Ne doit pas être nul |                          |
| Aca_code          | Code de l'académie                                     | Varchar | 3        | Elémentaire | Ne doit pas être nul |                          |
| Reg_code          | Code région                                            | Varchar | 3        | Elémentaire | Ne doit pas être nul |                          |



# **LE SCHÉMA RELATIONNEL NORMALISÉ**





# **LA BASE DE DONNÉES AVEC LES TABLES CRÉÉES ET LES DONNÉES CHARGÉES**

```
postgres=# \d bien;
```

| Colonne               |  | Type                   | Table « public.bien » |           | Par défaut                            |
|-----------------------|--|------------------------|-----------------------|-----------|---------------------------------------|
|                       |  |                        | Collationnement       | NULL-able |                                       |
| id_bien               |  | integer                |                       | not null  | nextval('bien_id_bien_seq'::regclass) |
| code_dep_code_commune |  | character varying(6)   |                       | not null  |                                       |
| no_voie               |  | integer                |                       |           |                                       |
| b_t_q                 |  | character(1)           |                       |           |                                       |
| type_de_voie          |  | character varying(4)   |                       |           |                                       |
| voie                  |  | character varying(128) |                       | not null  |                                       |
| total_piece           |  | integer                |                       | not null  |                                       |
| surface_carrez        |  | double precision       |                       | not null  |                                       |
| surface_local         |  | integer                |                       | not null  |                                       |
| type_local            |  | character varying(64)  |                       | not null  |                                       |

```
Index :
```

```
    "bien_pk" PRIMARY KEY, btree (id_bien)
```

```
Contraintes de clés étrangères :
```

```
    "commune_bien_fk" FOREIGN KEY (code_dep_code_commune) REFERENCES commune (code_dep_code_commune)
```

```
Référencé par :
```

```
    TABLE "vente" CONSTRAINT "bien_vente_fk" FOREIGN KEY (id_bien) REFERENCES bien (id_bien)
```

```
postgres=# \d commune;
```

| Colonne               |  | Type                  | Table « public.commune » |           | Par défaut |
|-----------------------|--|-----------------------|--------------------------|-----------|------------|
|                       |  |                       | Collationnement          | NULL-able |            |
| code_dep_code_commune |  | character varying(6)  |                          | not null  |            |
| code_département      |  | character varying(3)  |                          | not null  |            |
| code_commune          |  | character varying(3)  |                          | not null  |            |
| nom_commune           |  | character varying(64) |                          | not null  |            |
| population_commune    |  | integer               |                          | not null  |            |

```
Index :
```

```
    "commune_pk" PRIMARY KEY, btree (code_dep_code_commune)
```

```
Référencé par :
```

```
    TABLE "bien" CONSTRAINT "commune_bien_fk" FOREIGN KEY (code_dep_code_commune) REFERENCES commune (code_dep_code_commune)
```

```
    TABLE "region" CONSTRAINT "commune_region_fk" FOREIGN KEY (code_dep_code_commune) REFERENCES commune (code_dep_code_commune)
```

```
postgres=# \d region;
```

| Colonne               |  | Type                  | Table « public.region » |           | Par défaut                                |
|-----------------------|--|-----------------------|-------------------------|-----------|-------------------------------------------|
|                       |  |                       | Collationnement         | NULL-able |                                           |
| id_region             |  | integer               |                         | not null  | nextval('region_id_region_seq'::regclass) |
| code_dep_code_commune |  | character varying(6)  |                         | not null  |                                           |
| nom_region            |  | character varying(64) |                         | not null  |                                           |
| nom_département       |  | character varying(64) |                         | not null  |                                           |

```
Index :
```

```
    "region_pk" PRIMARY KEY, btree (id_region)
```

```
Contraintes de clés étrangères :
```

```
    "commune_region_fk" FOREIGN KEY (code_dep_code_commune) REFERENCES commune (code_dep_code_commune)
```

```
postgres=# \d vente;
```

| Colonne  |  | Type    | Table « public.vente » |           | Par défaut                              |
|----------|--|---------|------------------------|-----------|-----------------------------------------|
|          |  |         | Collationnement        | NULL-able |                                         |
| id_vente |  | integer |                        | not null  | nextval('vente_id_vente_seq'::regclass) |
| id_bien  |  | integer |                        | not null  |                                         |
| date     |  | date    |                        | not null  |                                         |
| valeur   |  | integer |                        | not null  |                                         |

```
Index :
```

```
    "vente_pk" PRIMARY KEY, btree (id_vente)
```

```
Contraintes de clés étrangères :
```

```
    "bien_vente_fk" FOREIGN KEY (id_bien) REFERENCES bien (id_bien)
```

```

postgres=# select * from bien;
 id_bien | code_dep_code_commune | no_voie | b_t_g | type_de_voie | voie | total_piece | surface_carrez | surface_local | type_local
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
 1 | 01103 | 347 | | RUE | DU CHATEAU | 3 | 48.22 | 48 | Appartement
 2 | 06004 | 4 | | BD | EDOUARD BAUDOUIN | 1 | 39.11 | 40 | Appartement
 3 | 06088 | 20 | B | RUE | MARCEAU | 3 | 80.25 | 82 | Appartement
 4 | 06123 | 550 | | RTE | DES VESPINS RN7 | 1 | 27.51 | 27 | Appartement
 5 | 13005 | 9300 | | RES | LES ARPEGES BD DES ABA | 2 | 47.33 | 47 | Appartement
 6 | 13028 | 27 | | RUE | DU GRAND MADIER | 1 | 25.31 | 24 | Appartement
 7 | 13208 | 360 | | AV | DU PRADO | 3 | 70.84 | 70 | Appartement
 8 | 13212 | 5076 | F | PARC | DESSUARD | 3 | 67.19 | 66 | Appartement
 9 | 14338 | 1194 | | RUE | DE NORMANDIE | 1 | 18.89 | 19 | Appartement
10 | 14366 | 30 | | ALL | DES NOISETIERS | 4 | 105.37 | 99 | Maison
11 | 17300 | 11 | | RUE | ROUGET DE L ISLE | 2 | 31.99 | 34 | Appartement
12 | 25056 | 13 | | RUE | BERTHE MORISOT | 5 | 96.21 | 100 | Maison
13 | 29232 | 1 | | RUE | DU POHER | 1 | 30.86 | 31 | Appartement
14 | 29260 | 2 | | RUE | DES JARDINS | 3 | 66.21 | 67 | Appartement
15 | 31555 | 5 | | AV | DU COMMANDANT TAILLANDIER | 2 | 45.58 | 46 | Appartement
16 | 33063 | 15 | | RUE | PAUL DENUCE | 1 | 23.2 | 27 | Appartement
17 | 33063 | 176 | | RUE | SAINTE CATHERINE | 1 | 13.1 | 12 | Appartement
18 | 33097 | 822 | G | | LA BAYNASSE SUD | 2 | 29 | 29 | Appartement
19 | 33449 | 9 | | RUE | DE BELFORT | 2 | 51.4 | 51 | Appartement
20 | 34145 | 183 | | RUE | MARC ANTOINE MENARD | 1 | 16.79 | 18 | Appartement
21 | 34150 | 50 | | AV | ANDRE CHASSEFIERE | 2 | 51.4 | 51 | Appartement
22 | 34199 | 8 | | PL | DES ETATS DU LANGUEDOC | 3 | 68.22 | 85 | Appartement
23 | 37156 | 7 | | RUE | JEAN-JACQUES ROUSSEAU | 4 | 88.45 | 89 | Maison
24 | 37261 | 197 | | AV | DE GRAMMONT | 2 | 42.94 | 46 | Appartement
25 | 38185 | 48 | | BD | MAL FOCH | 4 | 76.48 | 77 | Appartement
26 | 40266 | 210 | | RTE | DU TAILLEUR | 3 | 48.23 | 53 | Maison
27 | 40328 | 199 | | RES | MER ET GOLF-LE BOUCANIER | 2 | 49.91 | 48 | Appartement
28 | 44143 | 3 | | ALL | JEAN PERRIN | 2 | 46.3 | 45 | Appartement
29 | 44184 | 12 | | RUE | FERDINAND BUISSON | 2 | 47.24 | 48 | Appartement

postgres=# select count(*) from bien;
 count
-----
 34169

```



# REQUÊTES SQL ET RÉSULTATS



# REQUÊTE 1

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

```
postgres=# select count(*) as nb_vente_appartements from bien b
inner join vente v on b.id_bien = v.id_bien
where type_local = 'Appartement'
and date between to_date('2020-01-01', 'YYYY-MM-DD') and to_date('2020-01-01', 'YYYY-MM-DD') + INTERVAL '6 months';
 nb_vente_appartements
-----
                31362
(1 ligne)
```

# REQUÊTE 2

Le nombre de ventes d'appartement par région pour le 1er semestre 2020:

```
postgres=# select r.nom_region, count(*) as nb_vente_appartements from bien b
inner join vente v on b.id_bien = v.id_bien inner join region r on r.code_dep_code_commune = b.code_dep_code_commune
where type_local = 'Appartement' and date between to_date('2020-01-01', 'YYYY-MM-DD') and to_date('2020-01-01', 'YYYY-MM-DD') + INTERVAL '6 months'
group by r.nom_region order by 2 desc;
```

| nom_region                 | nb_vente_appartements |
|----------------------------|-----------------------|
| Ile-de-France              | 13989                 |
| Provence-Alpes-Côte d'Azur | 3645                  |
| Auvergne-Rhône-Alpes       | 3253                  |
| Nouvelle-Aquitaine         | 1931                  |
| Occitanie                  | 1640                  |
| Pays de la Loire           | 1357                  |
| Hauts-de-France            | 1252                  |
| Grand Est                  | 984                   |
| Bretagne                   | 983                   |
| Normandie                  | 861                   |
| Centre-Val de Loire        | 695                   |
| Bourgogne-Franche-Comté    | 376                   |
| Corse                      | 222                   |
| Martinique                 | 94                    |
| La Réunion                 | 44                    |
| Guyane                     | 34                    |
| Guadeloupe                 | 2                     |

(17 lignes)

L'Île-de-France compatibilise presque 50% des ventes d'appartements

# REQUÊTE 3

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

```
postgres=# ;with liste_ventes as (select count(*) as nb_ventes, total_piece from bien b
inner join vente v on b.id_bien = v.id_bien where type_local = 'Appartement' group by total_piece)
select l.total_piece, round((l.nb_ventes / b.total)*100, 2) as proportion_ventes_appartements from liste_ventes l
cross join (select sum(nb_ventes) as total from liste_ventes) b
order by 2 desc;
 total_piece | proportion_ventes_appartements
-----+-----
          2 |                31.16
          3 |                28.59
          1 |                21.48
          4 |                14.21
          5 |                 3.55
          6 |                 0.65
          7 |                 0.17
          0 |                 0.10
          8 |                 0.05
          9 |                 0.03
         10 |                 0.01
         11 |                 0.00
(12 lignes)
```

Les appartements de petites et moyennes surfaces sont les plus vendus

# REQUÊTE 4

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

```
postgres=# ;with liste_ventes as (select v.id_vente, b.code_dep_code_commune, cast(v.valeur as float) / b.surface_carrez as prix_metre_carre from bien b
inner join vente v on b.id_bien = v.id_bien where b.type_local = 'Appartement')
select r.nom_departement, c.code_departement, round(avg(l.prix_metre_carre::numeric), 2) as prix_metre_carre from liste_ventes l
inner join region r on l.code_dep_code_commune = r.code_dep_code_commune
inner join commune c on c.code_dep_code_commune = r.code_dep_code_commune
group by r.nom_departement, c.code_departement order by 3 desc limit 10;
```

| nom_departement   | code_departement | prix_metre_carre |
|-------------------|------------------|------------------|
| Paris             | 75               | 12052.06         |
| Hauts-de-Seine    | 92               | 7194.48          |
| Val-de-Marne      | 94               | 5426.66          |
| Alpes-Maritimes   | 06               | 4686.94          |
| Haute-Savoie      | 74               | 4654.36          |
| Seine-Saint-Denis | 93               | 4367.43          |
| Yvelines          | 78               | 4259.22          |
| Rhône             | 69               | 4082.90          |
| Corse-du-Sud      | 2A               | 3900.51          |
| Hautes-Alpes      | 05               | 3842.49          |

(10 lignes)

Paris et sa région restent le secteur immobilier le plus cher de France

# REQUÊTE 5

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

```
postgres=# ;with liste_maisons as (select v.id_vente, b.code_dep_code_commune, cast(v.valeur as float) / b.surface_carrez as prix_metre_carre from bien b
inner join vente v on b.id_bien = v.id_bien inner join region r on b.code_dep_code_commune = r.code_dep_code_commune
where b.type_local = 'Maison' and nom_region = 'Ile-de-France')
select round(avg(prix_metre_carre::numeric), 2) as prix_moyen_maison from liste_maisons;
 prix_moyen_maison
-----
          3745.09
(1 ligne)
```

# REQUÊTE 6

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

```
postgres=# ;with liste_appartements as (select v.id_bien, b.code_dep_code_commune, cast(v.valeur as float) / b.surface_carrez as prix_metre_carre, b.surface_carrez from bien b inner join vente v on
b.id_bien = v.id_bien where b.type_local = 'Appartement')
select l.id_bien, r.nom_region, round(l.prix_metre_carre::numeric, 2) as prix_metre_carre, l.surface_carrez from liste_appartements l
inner join region r on l.code_dep_code_commune = r.code_dep_code_commune
order by 3 desc limit 10;
```

| id_bien | nom_region          | prix_metre_carre | surface_carrez |
|---------|---------------------|------------------|----------------|
| 30603   | Ile-de-France       | 989010.99        | 9.1            |
| 21031   | Ile-de-France       | 673480.00        | 0.5            |
| 3625    | Ile-de-France       | 417406.96        | 20.55          |
| 33463   | Ile-de-France       | 349539.17        | 4.34           |
| 12394   | Ile-de-France       | 262318.84        | 3.45           |
| 18813   | Centre-Val de Loire | 221297.71        | 1.31           |
| 28890   | Ile-de-France       | 187117.58        | 4.38           |
| 7602    | Ile-de-France       | 178162.26        | 42.77          |
| 3131    | Ile-de-France       | 166688.06        | 7.79           |
| 5261    | Ile-de-France       | 134375.00        | 64             |

(10 lignes)

Certaines valeurs doivent être vérifiées notamment les surfaces habitables

# REQUÊTE 7

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

```
postgres=# ;with ventes_premier_trimestre as (select count(*) from bien b inner join vente v on b.id_bien = v.id_bien
where date between to_date('2020-01-01', 'YYYY-MM-DD') and to_date('2020-01-01', 'YYYY-MM-DD') + INTERVAL '3 months')
,ventes_second_trimestre as (select count(*) from bien b inner join vente v on b.id_bien = v.id_bien
where date between to_date('2020-04-01', 'YYYY-MM-DD') and to_date('2020-04-01', 'YYYY-MM-DD') + INTERVAL '3 months')
select round((cast(vst.count - vpt.count as numeric)/ vpt.count) * 100, 2) as evolution_ventes from ventes_premier_trimestre vpt
cross join (select count from ventes_second_trimestre) vst;
 evolution_ventes
-----
              3.32
(1 ligne)
```

vpt (vente premier trimestre):16824

vst (vente second trimestre): 17382

Taux d'évolution positif de 3,32% signifiant une bonne dynamique du marché immobilier

# REQUÊTE 8

Le classement des régions par rapport au prix au mètre carré des appartements de plus de 4 pièces:

```
postgres=# with liste_appartements as (  
select v.id_bien, b.code_dep_code_commune, cast(v.valeur as float) / b.surface_carrez as prix_metre_carre, b.surface_carrez from bien b  
inner join vente v on b.id_bien = v.id_bien  
where b.type_local = 'Appartement' and b.total_piece > 4)  
select r.nom_region, round(avg(l.prix_metre_carre::numeric), 2) as prix_metre_carre from liste_appartements l  
inner join region r on l.code_dep_code_commune = r.code_dep_code_commune  
group by r.nom_region  
order by 2 desc;  
nom_region | prix_metre_carre  
-----  
Ile-de-France | 8770.44  
La Réunion | 3641.81  
Provence-Alpes-Côte d'Azur | 3587.65  
Corse | 3104.88  
Auvergne-Rhône-Alpes | 2891.38  
Nouvelle-Aquitaine | 2465.48  
Bretagne | 2412.05  
Pays de la Loire | 2315.76  
Hauts-de-France | 2189.93  
Occitanie | 2097.23  
Normandie | 2015.77  
Grand Est | 1540.89  
Centre-Val de Loire | 1453.11  
Bourgogne-Franche-Comté | 1251.19  
Martinique | 573.48  
(15 lignes)
```

Sans surprise l'Ile-de-France arrive en première position suivie de la côte d'Azur




# REQUÊTE 9

Liste des communes ayant eu au moins 50 ventes au 1er trimestre:

```
postgres=# select c.nom_commune, count(*) as nb_ventes from bien b inner join vente v on b.id_bien = v.id_bien
inner join commune c on b.code_dep_code_commune = c.code_dep_code_commune
where date between to_date('2020-01-01', 'YYYY-MM-DD') and to_date('2020-01-01', 'YYYY-MM-DD') + INTERVAL '3 months'
group by c.nom_commune having count(*) > 50
order by 2 desc;
```

| nom_commune                  | nb_ventes |
|------------------------------|-----------|
| Paris 17e Arrondissement     | 228       |
| Paris 15e Arrondissement     | 215       |
| Paris 18e Arrondissement     | 210       |
| Nice                         | 173       |
| Paris 11e Arrondissement     | 170       |
| Paris 16e Arrondissement     | 166       |
| Bordeaux                     | 157       |
| Paris 14e Arrondissement     | 146       |
| Paris 20e Arrondissement     | 127       |
| Nantes                       | 120       |
| Paris 19e Arrondissement     | 116       |
| Paris 12e Arrondissement     | 110       |
| Paris 10e Arrondissement     | 109       |
| Paris 9e Arrondissement      | 107       |
| Grenoble                     | 106       |
| Boulogne-Billancourt         | 99        |
| Paris 13e Arrondissement     | 94        |
| Paris 6e Arrondissement      | 87        |
| Paris 7e Arrondissement      | 87        |
| Marseille 8e Arrondissement  | 81        |
| Asnières-sur-Seine           | 81        |
| Paris 5e Arrondissement      | 80        |
| Courbevoie                   | 80        |
| Paris 3e Arrondissement      | 79        |
| Toulouse                     | 78        |
| Antibes                      | 77        |
| Marseille 4e Arrondissement  | 73        |
| Marseille 1er Arrondissement | 71        |
| Vincennes                    | 68        |
| Rueil-Malmaison              | 68        |
| Lille                        | 67        |
| Marseille 9e Arrondissement  | 66        |
| Montrouil                    | 65        |
| Angers                       | 64        |
| Paris 8e Arrondissement      | 63        |
| La Ciotat                    | 63        |
| Nîmes                        | 63        |
| Rennes                       | 62        |
| Sète                         | 62        |
| Paris 2e Arrondissement      | 61        |
| Levallois-Perret             | 60        |
| Paris 4e Arrondissement      | 59        |
| Toulon                       | 59        |
| Saint-Maur-des-Fossés        | 56        |



```
Ajaccio      |      54  
Versailles   |      54  
Puteaux      |      53  
(47 lignes)
```

La région parisienne est fortement présente ainsi que les grandes villes comme Nice, Bordeaux ou Nantes

# REQUÊTE 10

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

```
postgres=# ;with liste_appartements as (select b.total_piece, sum(v.valeur/b.surface_carrez)/count(*) as prix_metre_carre from bien b
inner join vente v on b.id_bien = v.id_bien where b.type_local = 'Appartement'
group by b.total_piece having b.total_piece in (2, 3))
,pivot_pmc as (select max("prix_metre_carre") filter (where total_piece = 2) as prix_p2, max("prix_metre_carre") filter (where total_piece = 3) as prix_p3 from liste_appartements)
select round(cast(((prix_p3 - prix_p2) / prix_p2) as numeric)*100, 2) as difference_pourcentage from pivot_pmc;
 difference_pourcentage
-----
-12.40
(1 ligne)
```

Prix m2 pour un appartement 2 pièces: 4908€

Prix m2 pour un appartement 3 pièces: 4300€

La différence en pourcentage est de -12,4%

Ceci est logique car plus un bien est grand et plus le prix au m2 diminue

# REQUÊTE 11

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

```
postgres=# ;with liste_commune as (select round(avg(v.valeur::numeric), 2) as moyenne_fonciere, c.nom_commune, c.code_departement from bien b
inner join vente v on b.id_bien = v.id_bien inner join commune c on c.code_dep_code_commune = b.code_dep_code_commune
group by c.nom_commune, c.code_departement having c.code_departement in ('06', '13', '33', '59', '69'))
,rank_commune as (select moyenne_fonciere, nom_commune, code_departement,
rank() over (partition by code_departement order by moyenne_fonciere desc) from liste_commune)
select nom_commune, code_departement, moyenne_fonciere from rank_commune where rank <= 3;
```

| nom_commune            | code_departement | moyenne_fonciere |
|------------------------|------------------|------------------|
| Saint-Jean-Cap-Ferrat  | 06               | 968750.00        |
| Eze                    | 06               | 655000.00        |
| Mouans-Sartoux         | 06               | 476898.13        |
| Gignac-la-Nerthe       | 13               | 330000.00        |
| Saint-Savournin        | 13               | 314425.00        |
| Cassis                 | 13               | 313416.88        |
| Lège-Cap-Ferret        | 33               | 549500.64        |
| Vayres                 | 33               | 335000.00        |
| Arcachon               | 33               | 307435.93        |
| Bersée                 | 59               | 433202.00        |
| Cysoing                | 59               | 408550.00        |
| Halluin                | 59               | 322250.00        |
| Ville-sur-Jarnioux     | 69               | 485300.00        |
| Lyon 2e Arrondissement | 69               | 455217.26        |
| Lyon 6e Arrondissement | 69               | 426968.25        |

(15 lignes)


# REQUÊTE 12

Les 20 communes avec le plus de transactions pour 1000 habitants pour les communes qui dépassent les 10 000 habitants:

```
postgres=# ;with com_plus_dix_milles as (select code_dep_code_commune, nom_commune, (cast(population_commune as float) / 1000) as ratio
from commune where population_commune > 10000)
,nb_transactions as (select count(*), c.nom_commune, c.code_dep_code_commune from bien b inner join vente v on b.id_bien = v.id_bien
inner join com_plus_dix_milles c on b.code_dep_code_commune = c.code_dep_code_commune group by c.nom_commune, c.code_dep_code_commune)
select c.nom_commune, round(cast(t.count / c.ratio as numeric), 2) as tran_pour_mille from nb_transactions t
inner join com_plus_dix_milles c on t.code_dep_code_commune = c.code_dep_code_commune
order by 2 desc limit 20;
```

| nom_commune              | tran_pour_mille |
|--------------------------|-----------------|
| Paris 2e Arrondissement  | 5.84            |
| Paris 1er Arrondissement | 4.86            |
| Paris 3e Arrondissement  | 4.69            |
| Arcachon                 | 4.62            |
| La Baule-Escoublac       | 4.58            |
| Paris 4e Arrondissement  | 4.05            |
| Roquebrune-Cap-Martin    | 3.99            |
| Paris 8e Arrondissement  | 3.83            |
| Sanary-sur-Mer           | 3.50            |
| Paris 9e Arrondissement  | 3.43            |
| La Londe-les-Maures      | 3.43            |
| Paris 6e Arrondissement  | 3.38            |
| Saint-Cyr-sur-Mer        | 3.16            |
| Chantilly                | 3.13            |
| Pornichet                | 3.06            |
| Saint-Mandé              | 3.06            |
| Paris 10e Arrondissement | 3.04            |
| Menton                   | 2.94            |
| Saint-Hilaire-de-Riez    | 2.87            |
| Vincennes                | 2.81            |

(20 lignes)



La région Ile de France est la première en termes de nombre de vente d'appartements (13995 soit 44,6 %) au premier semestre de 2020

Les communes Paris 11, 14 à 18 et Paris 20 figurent dans le top 10 des communes ayant réalisées plus de 50 ventes au premier trimestre

La région Ile de France est aussi la région où les départements sont les plus chers en termes de prix du mètre carré



Un autre indicateur de la dynamique du marché est le nombre de transactions pour 1000 habitants

Il permet de voir à quelle fréquence les biens immobiliers changent de propriétaire par rapport à la population totale de la région

Il peut mettre en évidence, avec le prix du mètre carré d'une bulle immobilière sur Paris notamment

## Liste des requêtes au format texte:

```
select count(*) as nb_vente_appartements from bien b
inner join vente v on b.id_bien = v.id_bien
where type_local = 'Appartement'
and date between to_date('2020-01-01', 'YYYY-MM-DD') and to_date('2020-01-01', 'YYYY-MM-DD') + INTERVAL '6 months';
```

```
select r.nom_region, count(*) as nb_vente_appartements from bien b
inner join vente v on b.id_bien = v.id_bien inner join region r on r.code_dep_code_commune = b.code_dep_code_commune
where type_local = 'Appartement' and date between to_date('2020-01-01', 'YYYY-MM-DD') and to_date('2020-01-01', 'YYYY-MM-DD') + INTERVAL '6 months'
group by r.nom_region order by 2 desc;
```

```
;with liste_ventes as (select count(*) as nb_ventes, total_piece from bien b
inner join vente v on b.id_bien = v.id_bien where type_local = 'Appartement' group by total_piece)
select l.total_piece, round((l.nb_ventes / b.total)*100, 2) as proportion_ventes_appartements from liste_ventes l
cross join (select sum(nb_ventes) as total from liste_ventes) b
order by 2 desc;
```

```
;with liste_ventes as (select v.id_vente, b.code_dep_code_commune, cast(v.valeur as float) / b.surface_carrez as prix_metre_carre from bien b
inner join vente v on b.id_bien = v.id_bien where b.type_local = 'Appartement')
select r.nom_departement, c.code_departement, round(avg(l.prix_metre_carre::numeric), 2) as prix_metre_carre from liste_ventes l
inner join region r on l.code_dep_code_commune = r.code_dep_code_commune
inner join commune c on c.code_dep_code_commune = r.code_dep_code_commune
group by r.nom_departement, c.code_departement order by 3 desc limit 10;
```



```
;with liste_maisons as (select v.id_vente, b.code_dep_code_commune, cast(v.valeur as float) / b.surface_carrez as prix_metre_carre from bien b
inner join vente v on b.id_bien = v.id_bien inner join region r on b.code_dep_code_commune = r.code_dep_code_commune
where b.type_local = 'Maison' and nom_region = 'Ile-de-France')
select round(avg(prix_metre_carre::numeric), 2) as prix_moyen_maison from liste_maisons;
```

```
;with liste_appartements as (select v.id_bien, b.code_dep_code_commune, cast(v.valeur as float) / b.surface_carrez as prix_metre_carre, b.surface_carrez from bien b inner join vente v on
b.id_bien = v.id_bien where b.type_local = 'Appartement')
select l.id_bien, r.nom_region, round(l.prix_metre_carre::numeric, 2) as prix_metre_carre, l.surface_carrez from liste_appartements l
inner join region r on l.code_dep_code_commune = r.code_dep_code_commune
order by 3 desc limit 10;
```

```
;with ventes_premier_trimestre as (select count(*) from bien b inner join vente v on b.id_bien = v.id_bien
where date between to_date('2020-01-01', 'YYYY-MM-DD') and to_date('2020-01-01', 'YYYY-MM-DD') + INTERVAL '3 months')
,ventes_second_trimestre as (select count(*) from bien b inner join vente v on b.id_bien = v.id_bien
where date between to_date('2020-04-01', 'YYYY-MM-DD') and to_date('2020-04-01', 'YYYY-MM-DD') + INTERVAL '3 months')
select round((cast(vst.count - vpt.count as numeric) / vpt.count) * 100, 2) as evolution_ventes from ventes_premier_trimestre vpt
cross join (select count from ventes_second_trimestre) vst;
```

```
with liste_appartements as (
select v.id_bien, b.code_dep_code_commune, cast(v.valeur as float) / b.surface_carrez as prix_metre_carre, b.surface_carrez from bien b
inner join vente v on b.id_bien = v.id_bien
where b.type_local = 'Appartement' and b.total_piece > 4)
select r.nom_region, round(avg(l.prix_metre_carre::numeric), 2) as prix_metre_carre from liste_appartements l
inner join region r on l.code_dep_code_commune = r.code_dep_code_commune
group by r.nom_region
order by 2 desc;
```

```
select c.nom_commune, count(*) as nb_ventes from bien b inner join vente v on b.id_bien = v.id_bien
inner join commune c on b.code_dep_code_commune = c.code_dep_code_commune
where date between to_date('2020-01-01', 'YYYY-MM-DD') and to_date('2020-01-01', 'YYYY-MM-DD') + INTERVAL '3 months'
group by c.nom_commune having count(*) > 50
order by 2 desc;
```

```
;with liste_appartements as (select b.total_piece, sum(v.valeur/b.surface_carrez)/count(*) as prix_metre_carre from bien b
inner join vente v on b.id_bien = v.id_bien where b.type_local = 'Appartement'
group by b.total_piece having b.total_piece in (2, 3))
,pivot_pmc as (select max("prix_metre_carre") filter (where total_piece = 2) as prix_p2, max("prix_metre_carre") filter (where total_piece = 3) as prix_p3 from liste_appartements)
select round(cast(((prix_p3 - prix_p2) / prix_p2) as numeric)*100, 2) as difference_pourcentage from pivot_pmc;
```

```
;with liste_commune as (select round(avg(v.valeur::numeric), 2) as moyenne_fonciere, c.nom_commune, c.code_departement from bien b
inner join vente v on b.id_bien = v.id_bien inner join commune c on c.code_dep_code_commune = b.code_dep_code_commune
group by c.nom_commune, c.code_departement having c.code_departement in ('06', '13', '33', '59', '69'))
,rank_commune as (select moyenne_fonciere, nom_commune, code_departement,
rank() over (partition by code_departement order by moyenne_fonciere desc) from liste_commune)
select nom_commune, code_departement, moyenne_fonciere from rank_commune where rank <= 3;
```

```
;with com_plus_dix_milles as (select code_dep_code_commune, nom_commune, (cast(population_commune as float) / 1000) as ratio
from commune where population_commune > 10000)
,nb_transactions as (select count(*), c.nom_commune, c.code_dep_code_commune from bien b inner join vente v on b.id_bien = v.id_bien
inner join com_plus_dix_milles c on b.code_dep_code_commune = c.code_dep_code_commune group by c.nom_commune, c.code_dep_code_commune)
select c.nom_commune, round(cast(t.count / c.ratio as numeric), 2) as tran_pour_mille from nb_transactions t
inner join com_plus_dix_milles c on t.code_dep_code_commune = c.code_dep_code_commune
order by 2 desc limit 20;
```



**Merci !**