



Data IMMO

Créez et utilisez une base de données immobilière avec SQL.



SOMMAIRE

Contexte du projet	3
La stratégie de sauvegarde et la conformité RGPD	5
Les données initiales	7
Normaliser les données	8
Extrait du dictionnaire de données	9
Modélisation de la base de données	13
Implémenter la base de données	15
Les besoins en analyse de données	19



Contexte du projet



Lancement du projet "DATAImmo"

L'entreprise Laplace Immo a lancé un projet stratégique "DATAImmo" impulsé par la Direction Générale.



Objectif : prévoir le prix de vente des biens immobiliers.

Le but est d'aider les agences régionales à mieux accompagner leurs clients dans leurs décisions immobilières.



Modification de la base de données

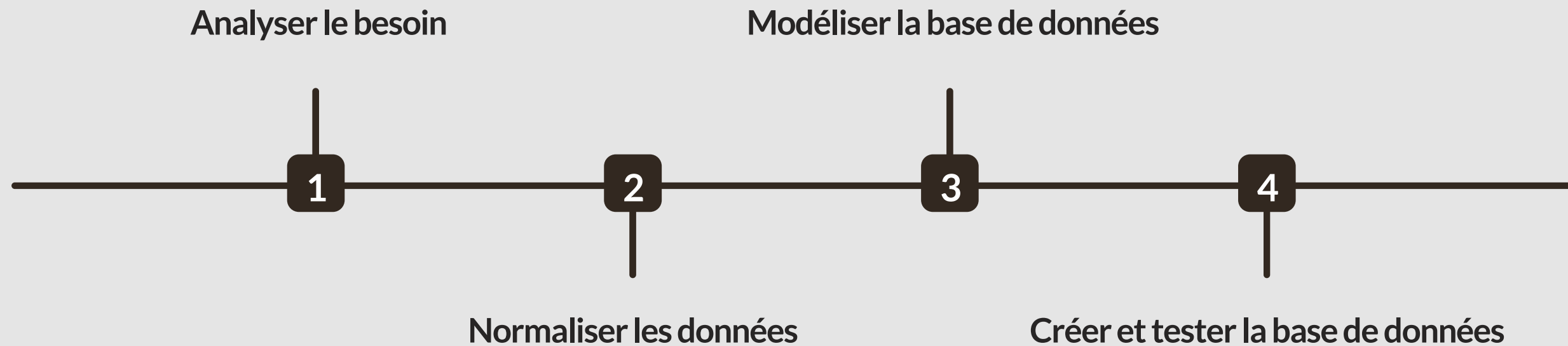
Le projet consiste à modifier la base de données pour collecter les transactions immobilières et foncières en France.



Phase pilote & PoC

Dans un premier temps, seules les données du premier semestre 2020 seront utilisées pour le développement d'un PoC (Proof of Concept).

Étapes de mise en œuvre de la base de données



Conformité au RGPD



Informations

Toute information se rapportant à une personne physique identifiée ou identifiable (ci-après dénommée «personne concernée»); est réputée être une «personne physique identifiable» **une personne physique** qui **peut être identifiée, directement ou indirectement**, notamment par référence à un identifiant, tel qu'un nom, un numéro d'identification, des données de localisation, un identifiant en ligne, ou à un ou plusieurs éléments spécifiques propres à son identité physique, physiologique, génétique, psychique, économique, culturelle ou sociale.



Finalité

Prévoir le prix de vente des biens immobiliers par la collecte des transactions immobilières et foncières en France.



Types
Informations

Personnelles anonymisées entre autres par la suppression du nom de l'acquéreur
Economiques et/ou commerciales – cadastrales et géographiques – Open Data
Stockées/hébergées en France ou en Europe.

Plan de sauvegarde

3

Copies des données



2

Supports différents



1

Dont 1 Hors site



3 sources de données Open Data

1

Données extraites du site open data des Demandes de valeurs foncières (DVF)

Ces données proviennent de la base de données des Demandes de valeurs foncières, une source ouverte qui contient des informations sur les transactions immobilières en France.

2

Données de l'INSEE avec les résultats des recensements de la population

Les données de l'Institut national de la statistique et des études économiques (INSEE) sur les recensements de la population seront utilisées pour enrichir les informations sur les biens immobiliers.

3

Données de data.gouv sur les régions, avec le référentiel géographique français

Le portail data.gouv.fr fournira des données géographiques sur les régions, les communes, les unités urbaines, les aires urbaines, les départements, les académies et les régions en France.

Normaliser pour éviter la redondance des données



Première Forme Normale

Une table est en 1NF si elle possède une clé primaire et si tous ses attributs sont atomiques.



Deuxième Forme Normale

Une table est en 2NF si elle est en 1NF et si tout attribut (n'appartenant pas à une clé candidate) ne dépend pas d'une partie seulement d'une clé candidate.



Troisième Forme Normale

Une relation est en 3NF si elle est en 2NF et si tout attribut (n'appartenant pas à une clé candidate) ne dépend pas d'un autre attribut n'appartenant pas à une clé candidate.

Extrait du dictionnaire de données

Table bien

Code	Signification	Type	Longueur	Nature	Règle de gestion	Règle de calcul
bien_id	Identifiant unique - clé primaire	Integer	NC	Elémentaire	Ne doit pas être nul	Auto-incrémenté
code_dep_code_commune_id	Clé étrangère référence la table	Varchar	10	Concaténé	Ne doit pas être nul	Code_dep + Code_commune
b_t_q	Indice de répétition	Varchar	3	Elémentaire		
numero_voie	Numéro des rues	Varchar	5	Elémentaire		
type_voie	Exemple : Rue, avenue, etc.	Varchar	10	Elémentaire		
voie	Libellé de la voie	Varchar	200	Elémentaire	Ne doit pas être nul	
prefixe_section	Préfixe de section de la parcelle	Varchar	5	Elémentaire		
section	Section de la parcelle	Varchar	5	Elémentaire		
numero_plan	Numéro de la parcelle	Integer	NC	Elémentaire	Ne doit pas être nul	
premier_lot	Lot de copropriété	Varchar	10	Elémentaire	Ne doit pas être nul	
surface_carrez_premier_lot	Surface en m2 de la partie privative	Float	NC	Elémentaire	Ne doit pas être nul	
nombre_lot	Nombre total de lots par disposition	Integer	NC	Elémentaire	Ne doit pas être nul	
surface_reelle_batiment	Surface mesurée au sol arrondie	Integer	NC	Elémentaire	Ne doit pas être nul	
nombre_piece_principale	Appartement de type 1, type 2, etc.	Integer	NC	Elémentaire	Ne doit pas être nul	
code_type_local	Clé étrangère de la table type_local	Integer	NC	Elémentaire	Ne doit pas être nul	Auto-incrémenté

Extrait du dictionnaire de données

Table population

Code	Signification	Type	Longueur	Nature	Règle de gestion	Règle de calcul
population_id	Identifiant unique - clé primaire	Integer	NC	Elémentaire	Ne doit pas être nul	Auto-incrémenté
code_dep_code_commune_id	Clé étrangère de la table commune	Varchar	10	Concaténé	Ne doit pas être nul	Code_dep + Code_commune
annee_recensement	Date de réalisation du recensement	Date	NC	Elémentaire	Ne doit pas être nul	
population_mun	Population municipale	Integer	NC	Elémentaire	Ne doit pas être nul	
population_cap	Population comptée à part	Integer	NC	Elémentaire	Ne doit pas être nul	
population_totale	Population totale de la commune	Integer	NC	Calculé	Ne doit pas être nul	Somme(population_mun + population_cap)

Extrait du dictionnaire de données

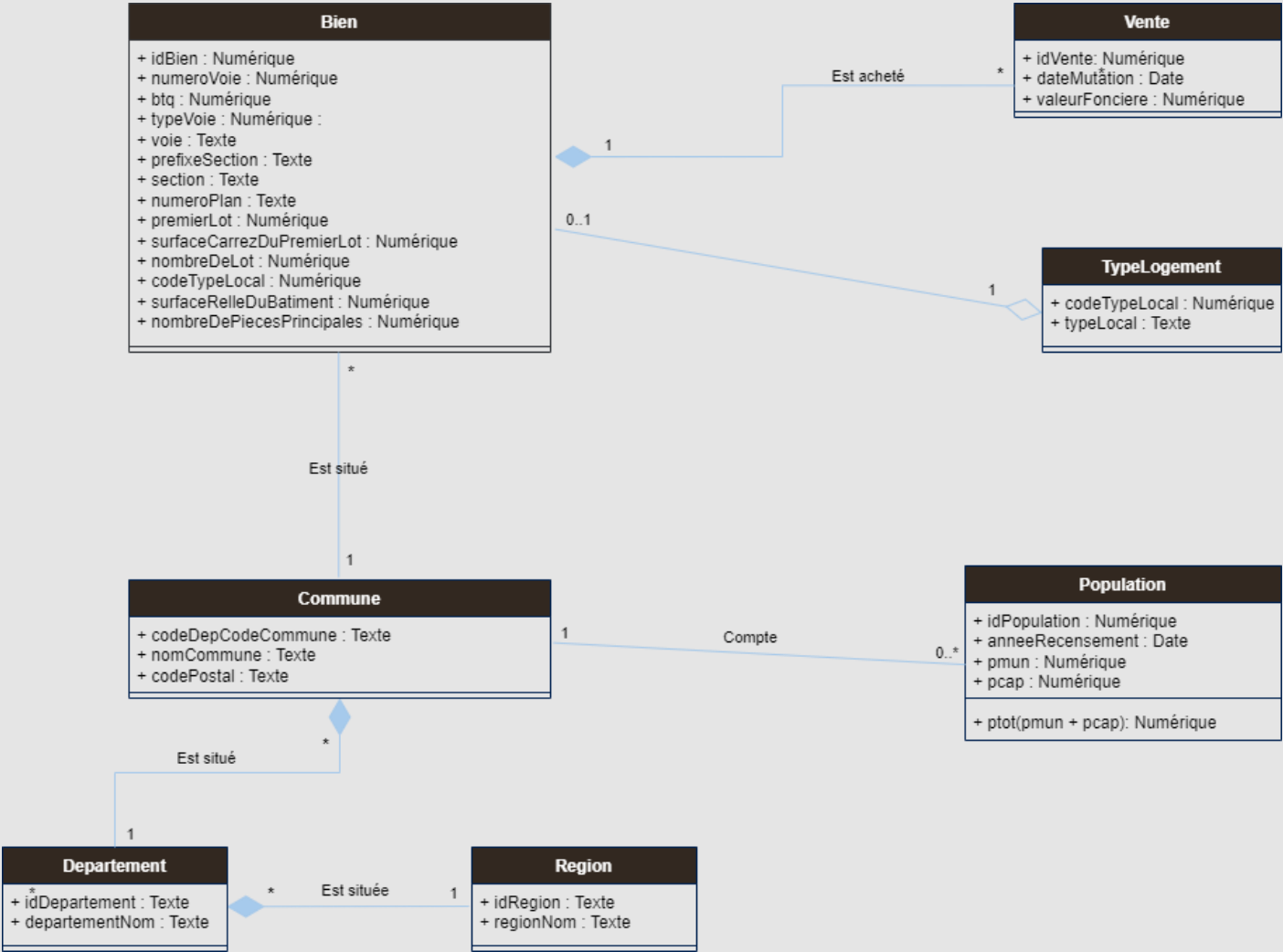
Table region et commune

Code	Signification	Type	Longueur	Nature	Règle de gestion	Règle de calcul
region_id	Identifiant unique - clé primaire	Varchar	3	Elémentaire	Ne doit pas être nul	
region_nom	Libellé de la région	Varchar	100	Elémentaire	Ne doit pas être nul	

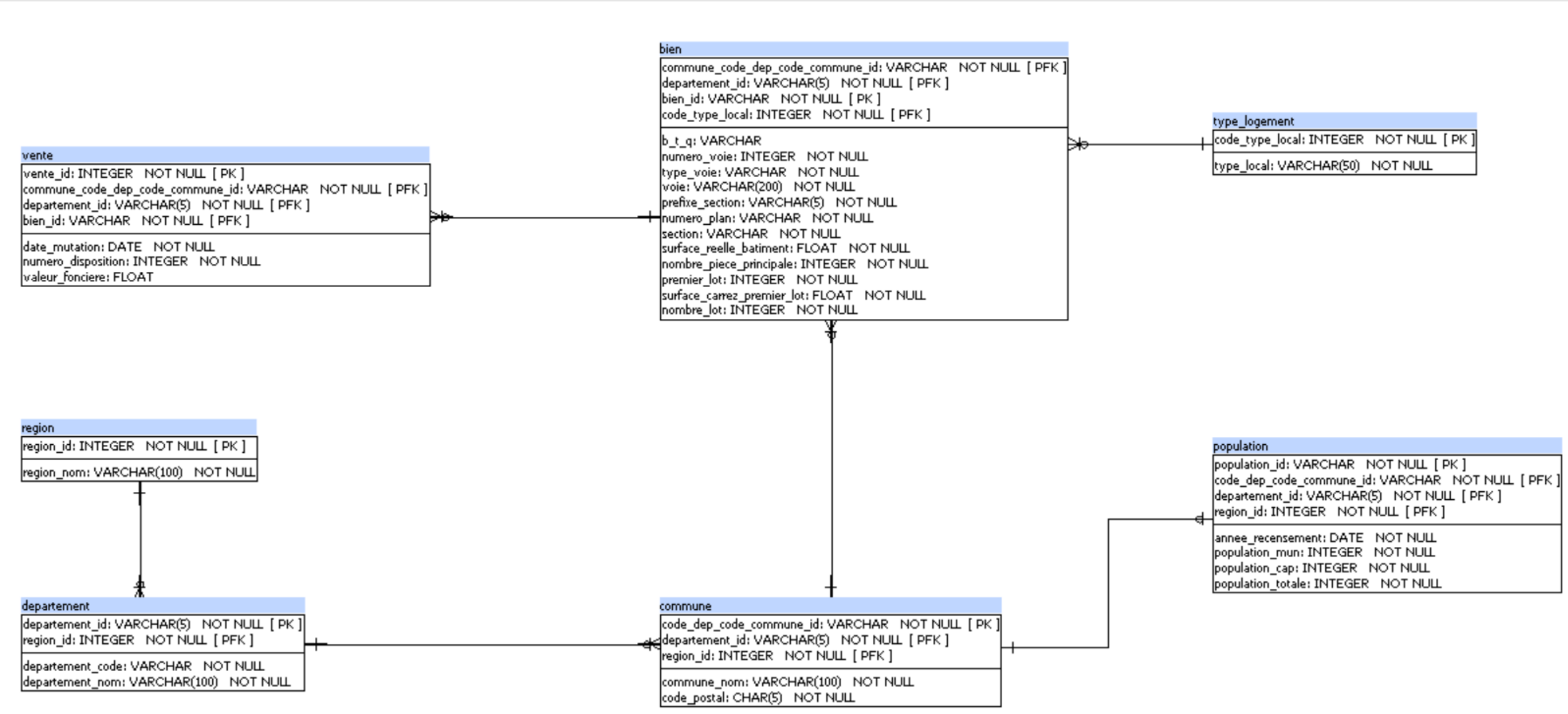
Code	Signification	Type	Longueur	Nature	Règle de gestion	Règle de calcul
code_dep_code_commune_id	Identifiant unique - clé primaire	Varchar	10	Concaténé	Ne doit pas être nul	Code_dep + Code_commune
region_id	Clé étrangère de la table région	Varchar	3	Elémentaire	Ne doit pas être nul	
departement_id	Clé étrangère de la table département	Varchar	5	Elémentaire	Ne doit pas être nul	
commune_nom	Libellé de la commune	Varchar	100	Elémentaire	Ne doit pas être nul	
code_postal	Code postal de la commune	Char	5	Elémentaire	Ne doit pas être nul	
code_commune	Code de la commune	Varchar	7	Elémentaire	Ne doit pas être nul	

Le modèle conceptuel de données

Le diagramme de classes

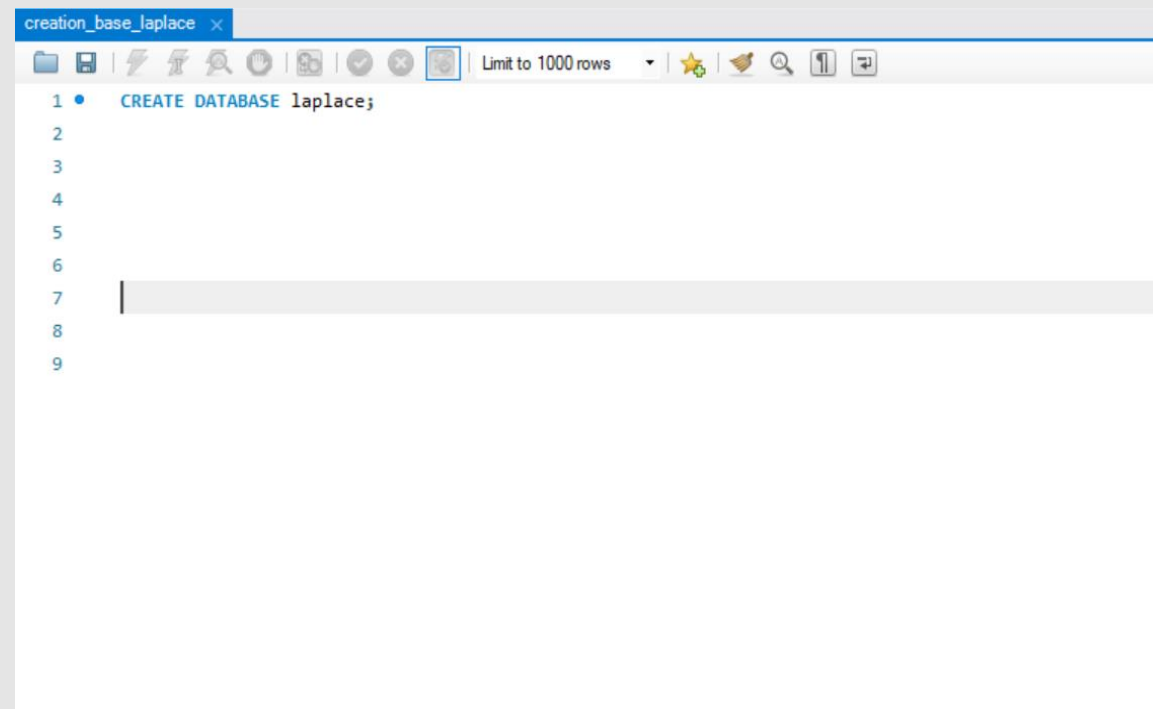


Le modèle logique de données : le schéma relationnel



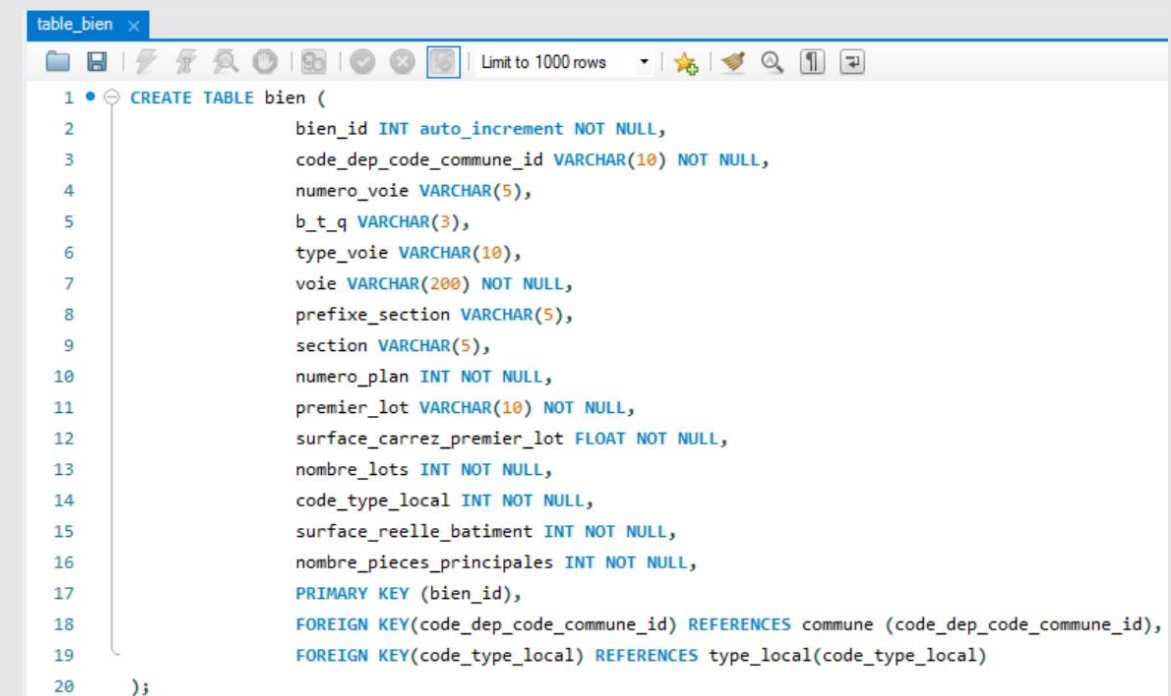
Le modèle physique de données

Traduire le modèle relationnel en code SQL compréhensible par le SGBDR : extrait issue de MySQL



```
1 • CREATE DATABASE laplace;
2
3
4
5
6
7
8
9
```








Création de la base de données laplace



```
1 • CREATE TABLE bien (
2     bien_id INT auto_increment NOT NULL,
3     code_dep_code_commune_id VARCHAR(10) NOT NULL,
4     numero_voie VARCHAR(5),
5     b_t_q VARCHAR(3),
6     type_voie VARCHAR(10),
7     voie VARCHAR(200) NOT NULL,
8     prefixe_section VARCHAR(5),
9     section VARCHAR(5),
10    numero_plan INT NOT NULL,
11    premier_lot VARCHAR(10) NOT NULL,
12    surface_carrez_premier_lot FLOAT NOT NULL,
13    nombre_lots INT NOT NULL,
14    code_type_local INT NOT NULL,
15    surface_reelle_batiment INT NOT NULL,
16    nombre_pieces_principales INT NOT NULL,
17    PRIMARY KEY (bien_id),
18    FOREIGN KEY(code_dep_code_commune_id) REFERENCES commune (code_dep_code_commune_id),
19    FOREIGN KEY(code_type_local) REFERENCES type_local(code_type_local)
20 );
```

Création de la table bien ainsi que ses attributs

Implémentation – les tables

Info	Tables	Columns	Indexes	Triggers	Views	Stored Procedures	Functions
Name	Engine	Version	Row Format	Rows			
 bien	InnoDB	10	Dynamic				
 commune	InnoDB	10	Dynamic				
 departement	InnoDB	10	Dynamic				
 population	InnoDB	10	Dynamic				
 region	InnoDB	10	Dynamic				
 type_logement	InnoDB	10	Dynamic				
 vente	InnoDB	10	Dynamic				

Implémentation nombre de lignes

1

•

SELECT COUNT(*) AS nombre_lignes

2

FROM commune

Result Grid

nombre_lignes

38916

1

•

SELECT COUNT(*) AS nombre_lignes

2

FROM departement

Result Grid

nombre_lignes

109

1

•

SELECT COUNT(*) AS nombre_lignes

2

FROM population

Result Grid

nombre_lignes

34991

1

•

SELECT COUNT(*) AS nombre_lignes

2

FROM region

Result Grid

nombre_lignes

19

Implémentation nombre de lignes

Limit to 1000 rows

1 • SELECT COUNT(*) AS nombre_lignes

2 FROM vente;

Result Grid

nombre_lignes

▶ 34169

Limit to 1000 rows

1 • SELECT COUNT(*) AS nombre_lignes

2 FROM type_logement;

Result Grid

nombre_lignes

▶ 2

Limit to 1000 rows

1 • SELECT COUNT(*) AS nombre_lignes

2 FROM bien

Result Grid

nombre_lignes

▶ 34169



Requêtes SQL et résultats

Besoins en analyse des données

01

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

```
SELECT COUNT(DISTINCT bien.bien_id) AS nombre_appartements_vendus_S1_2020
FROM bien
JOIN vente ON bien.bien_id = vente.bien_id
WHERE code_type_local = '2'
      AND date_mutation BETWEEN '2020-01-01' AND '2020-06-30'
```

Nombre_appartements_vendus_S1_2020
31 378

02

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

```
SELECT region_nom AS region,
       COUNT(bien_id) AS nombre_appartements_vendus_S1_2020
FROM bien
JOIN vente USING (bien_id)
JOIN commune USING (code_dep_code_commune_id)
JOIN region USING (region_id)
WHERE code_type_local = '2'
      AND date_mutation BETWEEN '2020-01-01' AND '2020-06-30'
GROUP BY region
ORDER BY nombre_appartements_vendus_S1_2020 DESC
```

Region	Nombre_appartements_vendus_S1_2020
Ile-de-France	13 995
Provence Alpes Côte d’Azur	3649
Auvergne Rhône Alpes	3253
Nouvelle Aquitaine	1932
Occitanie	1640
Pays de la Loire	1357
Hauts-de-France	1254
Grand-Est	984
Bretagne	983
Normandie	862
Centre-Val-de-Loire	696
Bourgogne Franche Comté	376
Corse	223
Martinique	94
La réunion	44
Guyane	34
Guadeloupe	2

03

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

```
SELECT nombre_pieces_principales AS nombre_de_pieces,  
       ROUND(COUNT(bien.bien_id) /  
             (SELECT COUNT(*)  
              FROM bien  
              INNER JOIN vente ON vente.bien_id = bien.bien_id  
              WHERE code_type_local = '2') * 100, 1) AS proportion_des_ventes  
FROM bien  
INNER JOIN vente ON vente.bien_id = bien.bien_id  
WHERE code_type_local = '2'  
GROUP BY nombre_de_pieces  
ORDER BY nombre_de_pieces ASC
```

nombre_de_pieces	proportion_des_ventes
0	0,1
1	21,5
2	31,2
3	28,6
4	14,2
5	3,6
6	0,7
7	0,2
8	0,1
9	0,0
10	0,0
11	0,0

04

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

```
SELECT departement_nom AS departement,
       ROUND(AVG(valeur_fonciere / surface_reelle_batiment), 2) AS prix_m2
FROM vente
JOIN bien USING (bien_id)
JOIN commune USING (code_dep_code_commune_id)
JOIN departement USING (departement_id)
WHERE code_type_local = '2'
ORDER BY prix_m2 DESC
LIMIT 10
```

departement	Prix_m2
Paris	12 124.73
Hauts-de-Seine	7379.81
Val-de-Marne	5457.64
Alpes-Maritimes	4645.97
Seine-Saint-Denis	4376.3
Haute-Savoie	4108.85
Yvelines	4100.01
Rhône	4070.4
Gironde	3850.87
Corse-du-Sud	3805.56

05

Prix moyen du mètre carré d’une maison en Île-de-France.

```
SELECT region_nom,  
       ROUND(AVG(valeur_fonciere / surface_reelle_batiment), 2) AS prix_moyen_m2  
FROM vente  
JOIN bien USING (bien_id)  
JOIN commune USING (code_dep_code_commune_id)  
JOIN region USING (region_id)  
WHERE code_type_local = '1'  
      AND region_nom = 'Île-de-France'
```

region	prix_moyen_m2
Ile-de-France	3997,71

06

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

```
SELECT bien_id,
       region_nom AS region,
       surface_reelle_batiment AS surface_m2,
       valeur_fonciere
FROM vente
JOIN bien USING (bien_id)
JOIN commune USING (code_dep_code_commune_id)
JOIN region USING (region_id)
WHERE code_type_local = '2'
ORDER BY valeur_fonciere DESC
LIMIT 10
```

bien_id	region	surface_m2	valeur_fonciere
30603	Ile-de-France	10	9 000 000
5261	Ile-de-France	62	8 600 000
3625	Ile-de-France	289	8 577 710
7602	Ile-de-France	42	7 620 000
9988	Ile-de-France	200	7 600 000
17823	Ile-de-France	143	7 535 000
410	Ile-de-France	357	7 420 000
16357	Ile-de-France	241	7 200 000
1924	Ile-de-France	310	7 050 000
19161	Ile-de-France	76	6 600 000

07

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

```
CREATE VIEW nb_ventes_t1_2020_vw AS
(SELECT COUNT(bien.bien_id) AS nombre_de_ventes_t1_2020
FROM bien
JOIN vente ON bien.bien_id = vente.bien_id
AND date_mutation BETWEEN '2020-01-01' AND '2020-03-31')

CREATE VIEW nb_ventes_t2_2020_vw AS
(SELECT COUNT(bien.bien_id) AS nombre_de_ventes_t2_2020
FROM bien
JOIN vente ON bien.bien_id = vente.bien_id
AND date_mutation BETWEEN '2020-04-01' AND '2020-06-30')

SELECT nombre_de_ventes_t1_2020,
       nombre_de_ventes_t2_2020,
       ROUND((nombre_de_ventes_t2_2020 - nombre_de_ventes_t1_2020) /
nombre_de_ventes_t1_2020 *100, 1) AS taux_evolution
FROM nb_ventes_t1_2020_vw,
     nb_ventes_t2_2020_vw
```

nombre_de_ventes_t1_2020	nombre_de_ventes_t2_2020	taux_evolution
16 776	17 393	3,7

08

Le classement des régions par rapport au prix au mètre carré des appartement de plus de 4 pièces.

```
SELECT region_nom AS region,
       ROUND(AVG(valeur_fonciere / surface_reelle_batiment), 2) AS prix_m2,
       nombre_pieces_principales
FROM vente
JOIN bien USING (bien_id)
JOIN commune USING (code_dep_code_commune_id)
JOIN region USING (region_id)
WHERE code_type_local = '2'
      AND nombre_pieces_principales > 4
GROUP BY region, nombre_pieces_principales
ORDER BY prix_m2 DESC
```

region	prix_m2	nombre_pieces_principales
Ile-de-France	13647.65	8
Ile-de-France	13291.9	9
Ile-de-France	12214.24	7
Ile-de-France	12053.3	10
Ile-de-France	9836.11	6
Ile-de-France	7142.3	5
Grand Est	1257.38	5
Hauts-de-France	1215.19	7
Bourgogne-Franche-Comté	1050.92	5
Grand Est	929.09	6
Centre-Val de Loire	639.55	6
Martinique	564.22	5

09

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

```
SELECT commune_nom AS commune,
       COUNT(bien_id) AS nombre_ventes_t1_2020
FROM bien
JOIN vente USING (bien_id)
JOIN commune USING (code_dep_code_commune_id)
WHERE date_mutation BETWEEN '2020-01-01' AND '2020-03-31'
GROUP BY commune
HAVING nombre_ventes_t1_2020 >= 50
ORDER BY nombre_ventes_t1_2020 DESC
```

commune	nombre_ventes_t1_2020
PARIS 17 ^E ARRONDISSEMENT	228
PARIS 15 ^E ARRONDISSEMENT	215
PARIS 18 ^E ARRONDISSEMENT	209
NICE	173
PARIS 11 ^E ARRONDISSEMENT	169
PARIS 16 ^E ARRONDISSEMENT	165
LEVALLOIS-PERRET	59
SAINT-MAUR-DES-FOSSES	56
VERSAILLES	54
AJACCIO	54
PUTEAUX	53
ISSY-LES-MOULINEAUX	50

10

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

```
WITH t2 AS
  (SELECT nombre_pieces_principales,
    ROUND(AVG(valeur_fonciere / surface_carrez_premier_lot), 1) AS
prix_m2_t2
  FROM vente
  JOIN bien ON bien.bien_id = vente.bien_id
  WHERE code_type_local = '2'
    AND nombre_pieces_principales = 2),
  t3 AS
  (SELECT nombre_pieces_principales,
    ROUND(AVG(valeur_fonciere / surface_carrez_premier_lot), 1) AS
prix_m2_t3
  FROM vente
  JOIN bien ON bien.bien_id = vente.bien_id
  WHERE code_type_local = '2'
    AND nombre_pieces_principales = 3)
SELECT prix_m2_t2,
  prix_m2_t3,
  ROUND((prix_m2_t3 - prix_m2_t2) / prix_m2_t3 * 100, 2) AS
variation_prix_en_pourcentage
FROM t2,
  t3
```

prix_m2_t2	prix_m2_t3	variation_prix_en_pourcentage
4908,6	4299,9	14,16

11

Les moyennes de valeurs foncières pour le top 3 des communes des départements 06, 13, 33, 59 et 69.

```
WITH prix_moyen_communes AS
  (SELECT departement_code,
    commune_nom,
    AVG(valeur_fonciere) AS prix_moyen,
    RANK() OVER(PARTITION BY departement_code
      ORDER BY AVG(valeur_fonciere) DESC) AS top_3_communes
  FROM bien
  INNER JOIN vente ON vente.bien_id = bien.bien_id
  INNER JOIN commune ON commune.code_dep_code_commune_id =
    bien.code_dep_code_commune_id
  WHERE departement_code IN ('06',
    '13',
    '33',
    '59',
    '69')

  GROUP BY departement_code,
    commune_nom)
SELECT top_3_communes,
  departement_code AS departement,
  commune_nom AS commune,
  ROUND(prix_moyen, 2) AS prix_moyen
FROM prix_moyen_communes
WHERE top_3_communes <= 3
ORDER BY departement ASC,
  top_3_communes ASC
```

top_3_communes	departement	commune	prix_moyen
1	06	SAINT-JEAN CAP FERRAT	968 750
2	06	EZE	655 000
3	06	MOUANS-SARTOUX	476 898.09
1	13	GIGNAC-LA-NERTHE	330 000
2	13	SAINT SAVOURNIN	314 425
3	13	CASSIS	313 416.88
1	33	LEGE-CAP-FERRET	549 500.64
2	33	VAYRES	335 000
3	33	ARCACHON	307 435.93
1	59	BERSEE	433 202
2	59	CYSOING	408 550
3	59	HALLUIN	322 250
1	69	VILLE-SUR-JARNIOUX	485 300
2	69	LYON 2 ^E ARRONDISSEMENT	455 217.27
3	69	LYON 6 ^E ARRONDISSEMENT	426 968.25

12

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

```
CREATE VIEW population_totale_vw3 AS (  
  SELECT code_dep_code_commune_id,  
         (population_mun + population_cap) AS population_totale  
  FROM population  
)  
WITH transactions AS  
(SELECT commune_nom AS commune,  
         commune.code_dep_code_commune_id,  
         COUNT(bien.bien_id) AS nombre_transactions  
  FROM bien  
  INNER JOIN commune ON commune.code_dep_code_commune_id =  
bien.code_dep_code_commune_id  
  INNER JOIN vente ON vente.bien_id = bien.bien_id  
  GROUP BY code_dep_code_commune_id)  
SELECT commune,  
       ROUND((nombre_transactions / population_totale) * 1000, 1) AS  
transactions_1000_habitants  
FROM population_totale_vw3  
INNER JOIN transactions USING(code_dep_code_commune_id)  
WHERE population_totale > 10000  
ORDER BY transactions_1000_habitants DESC  
LIMIT 20
```

commune	transactions_pour_1000_habitants
PARIS 2E ARRondissement	5.8
PARIS 1ER ARRondissement	4.9
PARIS 3E ARRondissement	4.7
ARCACHON	4.6
LA BAULE-ESCOUBLAC	4.6
PARIS 4E ARRondissement	4.1
ROQUEBRUNE-CAP-MARTIN	4
PARIS 8E ARRondissement	3.8
SANARY-SUR-MER	3.5
PARIS 9E ARRondissement	3.4
PARIS 6E ARRondissement	3.4
LA LONDE-LES-MAURES	3.4
SAINT-CYR-SUR-MER	3.2
CHANTILLY	3.1
SAINT-MANDE	3.1

12

Les 20 communes avec le plus de transactions pour 1000 habitants pour les communes qui dépassent les 10 000 habitants. (suite)

```
CREATE VIEW population_totale_vw3 AS (  
  SELECT code_dep_code_commune_id,  
         (population_mun + population_cap) AS population_totale  
  FROM population  
)  
WITH transactions AS  
(SELECT commune_nom AS commune,  
         commune.code_dep_code_commune_id,  
         COUNT(bien.bien_id) AS nombre_transactions  
  FROM bien  
  INNER JOIN commune ON commune.code_dep_code_commune_id =  
bien.code_dep_code_commune_id  
  INNER JOIN vente ON vente.bien_id = bien.bien_id  
  GROUP BY code_dep_code_commune_id)  
SELECT commune,  
       ROUND((nombre_transactions / population_totale) * 1000, 1) AS  
transactions_1000_habitants  
FROM population_totale_vw3  
INNER JOIN transactions USING(code_dep_code_commune_id)  
WHERE population_totale > 10000  
ORDER BY transactions_1000_habitants DESC  
LIMIT 20
```

commune	transactions_pour_1000_habitants
PORNICHET	3.1
PARIS 10E ARRONDISSEMENT	3
MENTON	2.9
SAINT-HILAIRE-DE-RIEZ	2.9
ENGHIEN-LES-BAINS	2.8