



PROJET BIG DATA

Création d'une base de données

Fallou NGOM, Cheikna Amala Yatabare, Mamadou

Lamine NDAO, Ousseynou DIOP, Papa

Abdourahmane CISSE



Sous la supervision de Madame Mously DIAW

PRESENTATION 2024





Plan

INTRODUCTION

Création d'une base de données et liaison à un SGBD

1

Dictionnaire des données

2



Création des tables et schema relationnel

3

4

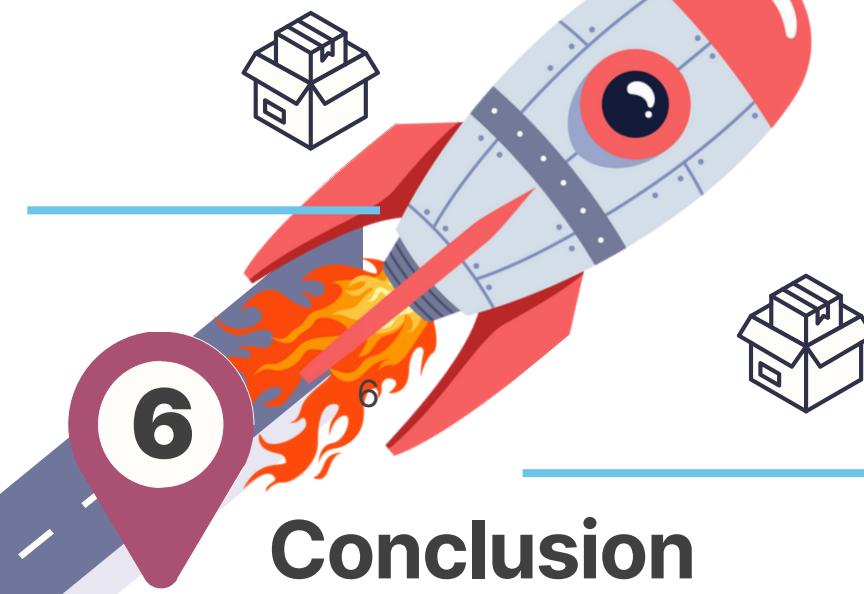
5

Execution des requêtes et présentation du site

6



Conclusion



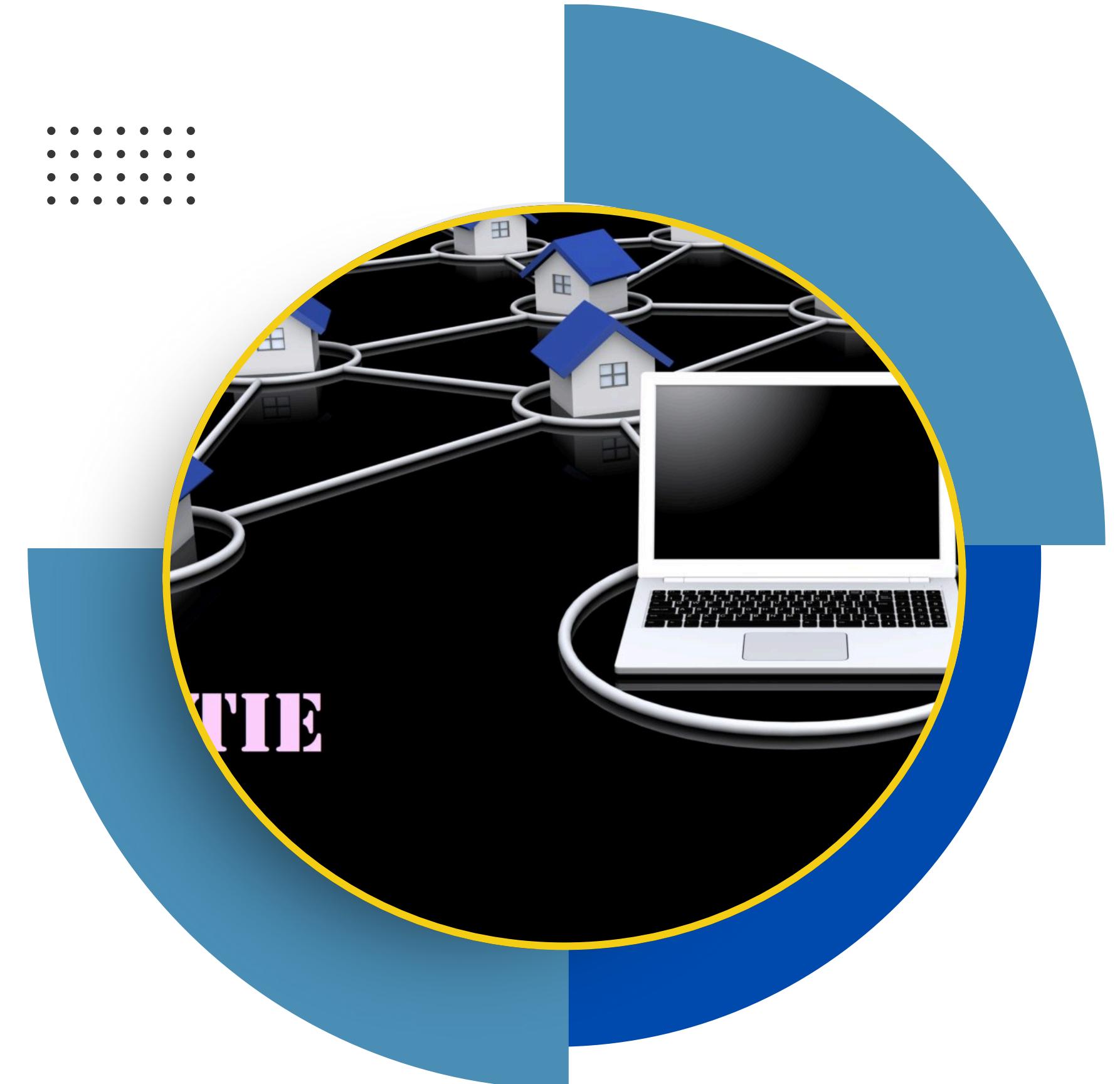
INTRODUCTION

Dans un marché immobilier concurrentiel, la gestion des données est un levier stratégique pour se démarquer. En tant que data analyst chez Immo Laplace, notre mission est de concevoir une base de données relationnelle hébergée sur AWS RDS et gérée avec un SGBD pour structurer et exploiter efficacement les données de l'entreprise. Ce projet vise à développer un modèle de prédiction des prix de vente immobiliers, offrant à Immo Laplace un outil de décision puissant et sécurisé. Ce rapport présente les étapes de création de la base, incluant la modélisation des données et l'écriture de requêtes SQL optimisées, ainsi que le développement d'un site pour l'entreprise . Cette base centralisée constituera un référentiel stratégique pour les analyses de marché et les prévisions, permettant à Immo Laplace d'anticiper les tendances et de renforcer sa position sur le marché.



Dictionnaire des données

PRESENTATION DU DICTIONNAIRE DES DONNÉES



BIEN							
VARIABLE	CODE	SIGNIFICATION	TYPE	LONGUEUR	NATURE	RÈGLE DE GESTION	COMMENTAIRE
Identifiant du bien	ID_bien	Identification du bien	Varchar	15	Clé primaire	Doit être unique pour chaque bien	Identifiant unique du bien vendu
type de local	Type_Local	Type de local ou de bien	Varchar	15	Attribut simple	Non nul, doit comporter obligatoirement au moins un nombre	Cela permet de caractériser la nature du bien
Surface du bâtiment	Surface_Bati	Surface du bâtiment vendus	float	-	Attribut simple	Non nul, doit comporter obligatoirement au moins un nombre	Cela permet de caractériser la taille du bien
Nombre de pièces	Nbre_Pièces_prin	Nombre de pièces du bâtiment	INT	-	Attribut simple	Non nul, doit comporter obligatoirement au moins un nombre	Cela permet de connaître des caractéristiques plus précises sur le bien à vendre
Code de la commune	Code_Commune	Code de la commune	Varchar	6	Clé Secondaire	Clé primaire de commune	Assure la contrainte référentielle entre la table Bien et Commune

COMMUNE							
VARIABLE	CODE	SIGNIFICATION	TYPE	LONGUEUR	NATURE	RÈGLE DE GESTION	COMMENTAIRE
code de la commune	Code_Commune	Identification de la commune	VARCHAR	6	Clé primaire	Doit être unique pour chaque commune et non nul	Identifiant unique de la commune
Nom commune	Nom_COM	Nom de la commune	VARCHAR	50	Attribut simple	Référence au nom de la commune	Il s'agit du nom de la commune, qui peut être facultatif à ne pas mettre dans la base de données.
population commune	POPMUN	Population de la municipalité	INT	-	Attribut simple	Non nul, doit comporter obligatoirement un montant	Il s'agit de la population totale au niveau de chaque commune
Identifiant Du Département	depart_ID	Identifiant du département	VARCHAR	2	Clé étrangère	du département	pour établir la liaison entre la table commune et la table région

On a ici deux tables (Bien et Commune). Chaque table contient des attributs avec des règles de gestion garantissant l'unicité et l'intégrité des données. La relation entre les deux tables est assurée par le code de la commune, permettant la cohérence des informations.

DEPARTEMENT							
VARIABLE	CODE	SIGNIFICATION	TYPE	LONGUEUR	NATURE	RÈGLE DE GESTION	COMMENTAIRE
identification_département	depart_ID	identifiant département	Varchar	3	clé primaire	1°)Le code doit être unique et conforme au format INSEE (2 ou 3 caractères). 2°)Aucune duplication n'est permise. 3°)Le code ne peut pas être vide (obligatoire).	
nom du département	Nom_Depart	nom du département	Varchar	50	attribut simple	1°)Le nom doit être unique dans la base. 2°)Aucun caractère spécial ni abréviation non standard ne doit être utilisé. 3°) Valeur obligatoire, ne peut pas être vide.	
code région	region_ID	Identifiant de la région	Varchar	2	clé secondaire	1°)Doit correspondre à une région existante dans la table des régions.	

VENTE								
VARIABLE	CODE	SIGNIFICATION	TYPE	LONGUEUR	NATURE	RÈGLE DE GESTION	COMMENTAIRE	IMPORTANCE
-	ID_vente	Identification d'une vente	INT	-	Clé primaire	Doit être unique pour chaque vente, non nul	Identifiant unique de la vente.	Relation avec d'autres tables de la base de données
-	ID_bien	Identification du bien vendu	INT	-	Clé étrangère	Référence à la table Bien, non nul	Identifiant unique du bien immobilier vendu. Référence à id_bien dans la table Bien, établissant une relation entre la vente et le bien immobilier concerné.	Assure la contrainte référentielle
Date mutation (Colonne i données brute)	date_vente	date de mutation (transaction)	DATE	-	Attribut simple	Non nul, doit être une date valide	Date de la transaction de vente.	Essentielle pour les analyses temporelles et permet aussi de suivre les variations saisonnières.
Valeur foncière (Colonne k données brute)	prix	prix de vente du bien	DECIMAL	15,2	Attribut simple	Non nul, doit être supérieur à zéro	Montant total de la vente en euros, avec deux décimales.	C'est la variable clé pour l'analyse des prix du marché immobilier. Les modèles de prévision utiliseront cette variable pour prédire les prix futurs et comprendre les tendances de valorisation des biens.

On a ici deux tables (département et vente). Chaque table contient des attributs avec des règles de gestion garantissant l'unicité et l'intégrité des données.

REGION							
VARIABLE	CODE	SIGNIFICATION	TYPE	LONGUEUR	NATURE	RÈGLE DE GESTION	COMMENTAIRE
Région identifiant	region_ID	Identification de la région	varchar	2	clé primaire	1) Le code doit être unique et ne peut pas être modifié une fois attribué. 2) Code à 2 chiffres obligatoire. 3) Doit respecter les standards INSEE.	permet de désigner le nom de la région pour définir le lien avec la table département
Nom_Région	nom_region	Nom de la région	Varchar	50	attribut simple	1) Le nom doit être unique. 2) Ne peut contenir d'abréviations non standard. 3) Valeur obligatoire.	permet de mieux préciser les caractéristiques de la région

Et enfin, nous avons la table des régions qui donne des informations sur le lieu géographique des biens.

SQL

CREATION DE BASE SUR RDS



Creation Base de Donnée RDS (AWS)

Nous avons crée une base de données MySQL à partir de RDS (Relational Database Service) de AWS.

Notre base de données s'appelle database-1 et nous l'avons configuré en mode public afin de permettre à tous les membres utilisateurs d'y accéder via le hostpost et leur IP.

RDS > Bases de données

i Envisagez de créer un déploiement bleu/vert pour minimiser les temps d'arrêt lors des mises à niveau

Vous pouvez envisager d'utiliser les déploiements bleu/vert Amazon RDS et de minimiser vos temps d'arrêt lors des mises à niveau. Un déploiement bleu/vert fournit un environnement intermédiaire pour les modifications apportées aux bases de données de production. [Guide de l'utilisateur RDS](#) [Guide de l'utilisateur Aurora](#)

Bases de données (2)

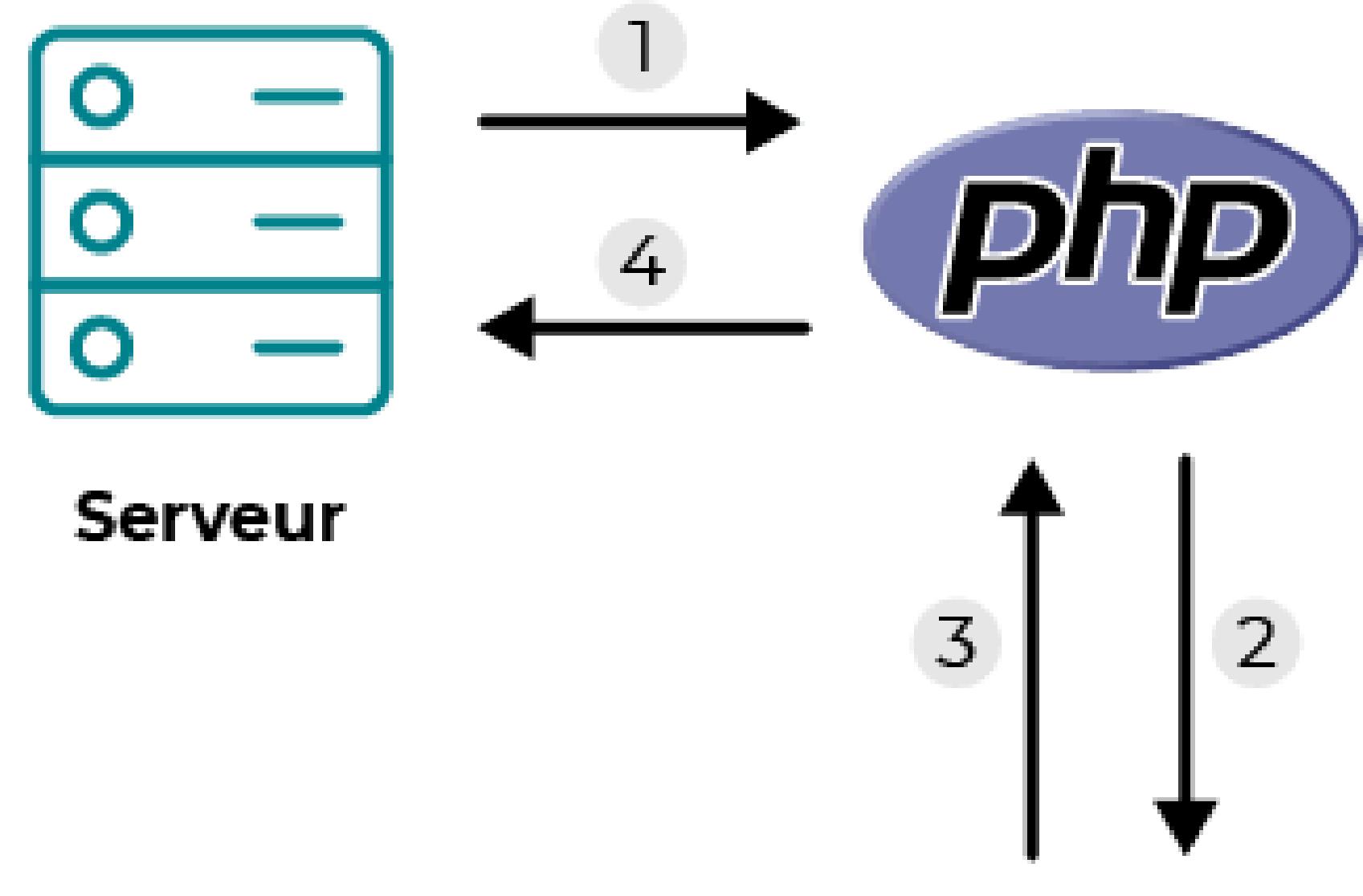
Ressources de groupe [Modifier](#) [Actions ▾](#) [Restaurer à partir de S3](#)

Votre texte de paragraphe
[Créer une base de données](#)

[Filtrer par bases de données](#)

<input type="checkbox"/> Identifiant de base de données	▲	Statut	▼	Rôle	▼	Moteur	▼	Région ...	▼	T
<input type="checkbox"/> database-1		Disponible		Instance		MySQL Co...		eu-west-3b		d

LIAISON ENTRE WORKBENCH, RDS ET PHP_MYSQLADMIN



Liaison à database-1 sur RDS

Avant toute connection à la base de donnée sur RDS, Vous devez préalablement avoir certaines autorisations. Vous devez communiquer votre IPV4 à l'administrateur de la base de donnée qui va inclure dans les groupes de sécurité afin de vous donnez l'accès

Connectivité et sécurité

Point de terminaison et port	Mise en réseau	Sécurité
Point de terminaison database-1.cvqgc2ow6zqg.eu-west-3.rds.amazonaws.com	Zone de disponibilité eu-west-3a	Groupes de sécurité VPC default (sg-07bb7960c96a51856) <input checked="" type="checkbox"/> Actif
Port 3306	VPC vpc-098f5673729425e8a	Accessible publiquement Oui
	Groupe de sous-réseaux default-vpc-098f5673729425e8a	Autorité de certification Infos rds-ca-rsa2048-g1
	Sous-réseaux subnet-0a1c72f1d42526c21 subnet-0c4257697d0f4ad9d subnet-0f91506f975f9c65f	Date d'autorité de certification May 25, 2061, 23:18 (UTC+00:00)
	Type de réseau IPv4	Date d'expiration du certificat d'instance de base de données October 22, 2025, 08:27 (UTC+00:00)

Groupes de sécurité (1) [Informations](#)

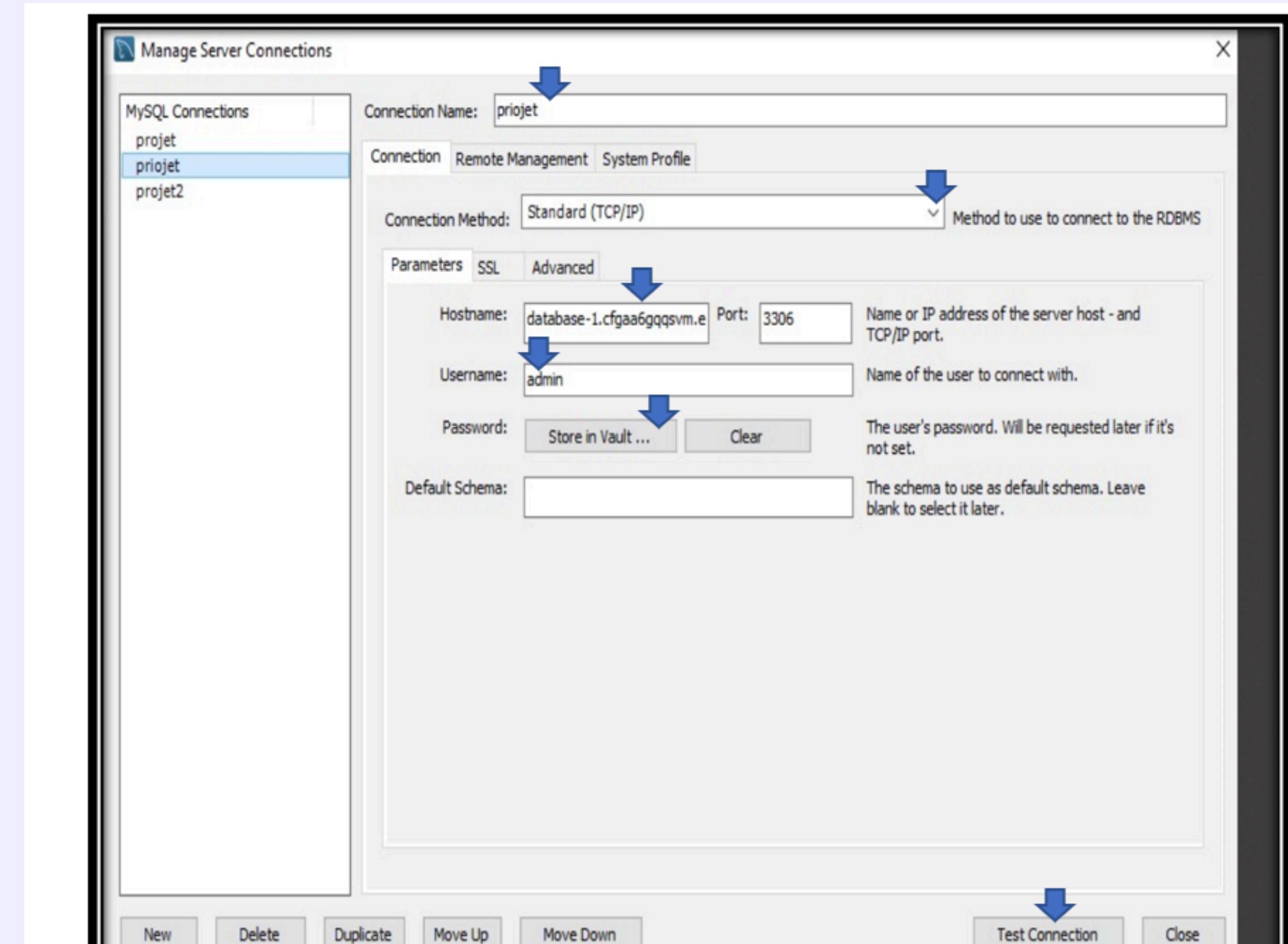
Name	ID du groupe de sécurité	Nom du groupe de sécurité	ID de VPC
-	sg-07bb7960c96a51856	default	vpc-098f5673729425e8a

Règles de sécurité

Type Informations	Protocole Informations	Plage de ports Informations	Destination Informations	Description - facultatif Informations
Tout le trafic	Tous	Tous	Mon IP ▾ 154.124.137.126/32 X	Supprimer

Liaison entre database-1 et MySQL Workbench

La liaison entre MySQL Workbench et Amazon RDS se fait en configurant une connexion avec les identifiants de l'instance RDS (hôte, port, nom d'utilisateur et mot de passe). Cela permet de gérer et interroger la base de données RDS directement depuis MySQL Workbench.



Liaison entre database-1 et Adminer.php

La liaison entre phpMyAdmin et Amazon RDS se fait en configurant phpMyAdmin avec les identifiants de l'instance RDS (hôte, port, nom d'utilisateur et mot de passe). Cela permet de gérer et interroger la base de données RDS directement depuis l'interface web de phpMyAdmin.

The screenshot shows the Adminer 4.8.1 login interface. At the top left is a language dropdown set to "English". To the right is a "Login" button with a checked "Permanent login" checkbox. Below these are five input fields:

System	MySQL
Server	database-1.cfgaa6gqqsvm.e
Username	admin
Password
Database	Laplacelmmo

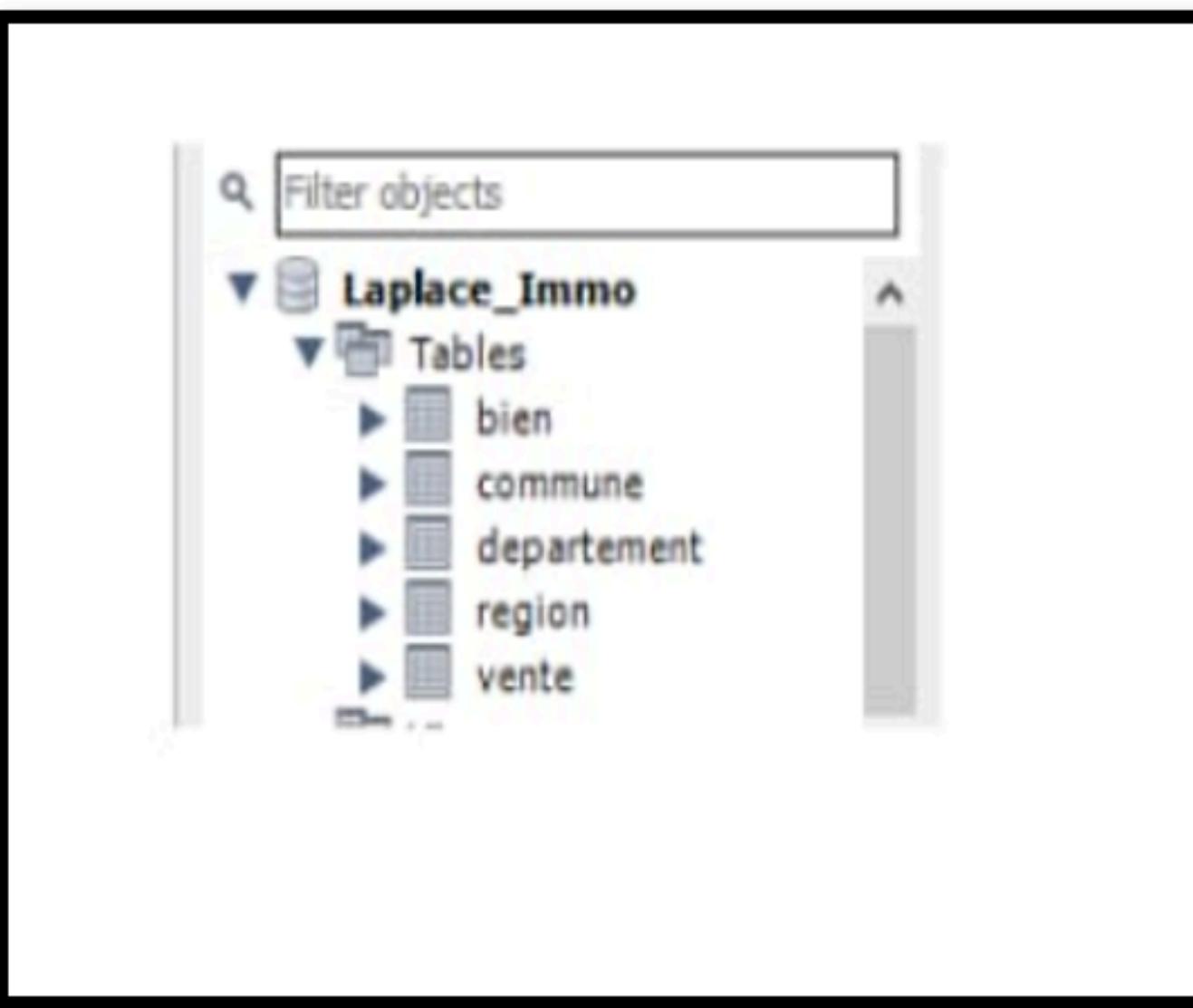
Blue arrows point from the text in the adjacent text block to the "Server", "Username", "Password", and "Database" fields. Another blue arrow points from the "Permanent login" checkbox to the "Login" button.

CREATION DES TABLES ET SCHEMA RELATIONNEL

Creation des tables

Le code pour la creation des tables va partir d'un SGBD.

Ce code est le même quel que soit le le SGBD utilisé.
Nous avons utilisé MySQL Workbench comme SGBD de creation.



```
/*!40101 SET @saved_cs_client      = @@character_set_client */;
SET character_set_client = utf8mb4 ;
CREATE TABLE `bien` (
  `Id_bien` int NOT NULL,
  `Type_Local` varchar(15) NOT NULL,
  `Surface_Bati` float NOT NULL,
  `Nombre_Piece_Princ` int NOT NULL,
  `Code_Commune` int DEFAULT NULL,
  PRIMARY KEY (`Id_bien`),
  KEY `Code_Commune_idx` (`Code_Commune`),
  CONSTRAINT `Code_Commune` FOREIGN KEY (`Code_Commune`) REFERENCES `commune` (`Code_Commune`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

DROP TABLE IF EXISTS `commune`;
/*!40101 SET @saved_cs_client      = @@character_set_client */;
SET character_set_client = utf8mb4 ;
CREATE TABLE `commune` (
  `Code_Commune` int NOT NULL,
  `Nom_Com` varchar(50) NOT NULL,
  `depart_ID` int DEFAULT NULL,
  PRIMARY KEY (`Code_Commune`),
  KEY `depart_ID_idx` (`depart_ID`),
  CONSTRAINT `depart_ID` FOREIGN KEY (`depart_ID`) REFERENCES `departement` (`depart_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

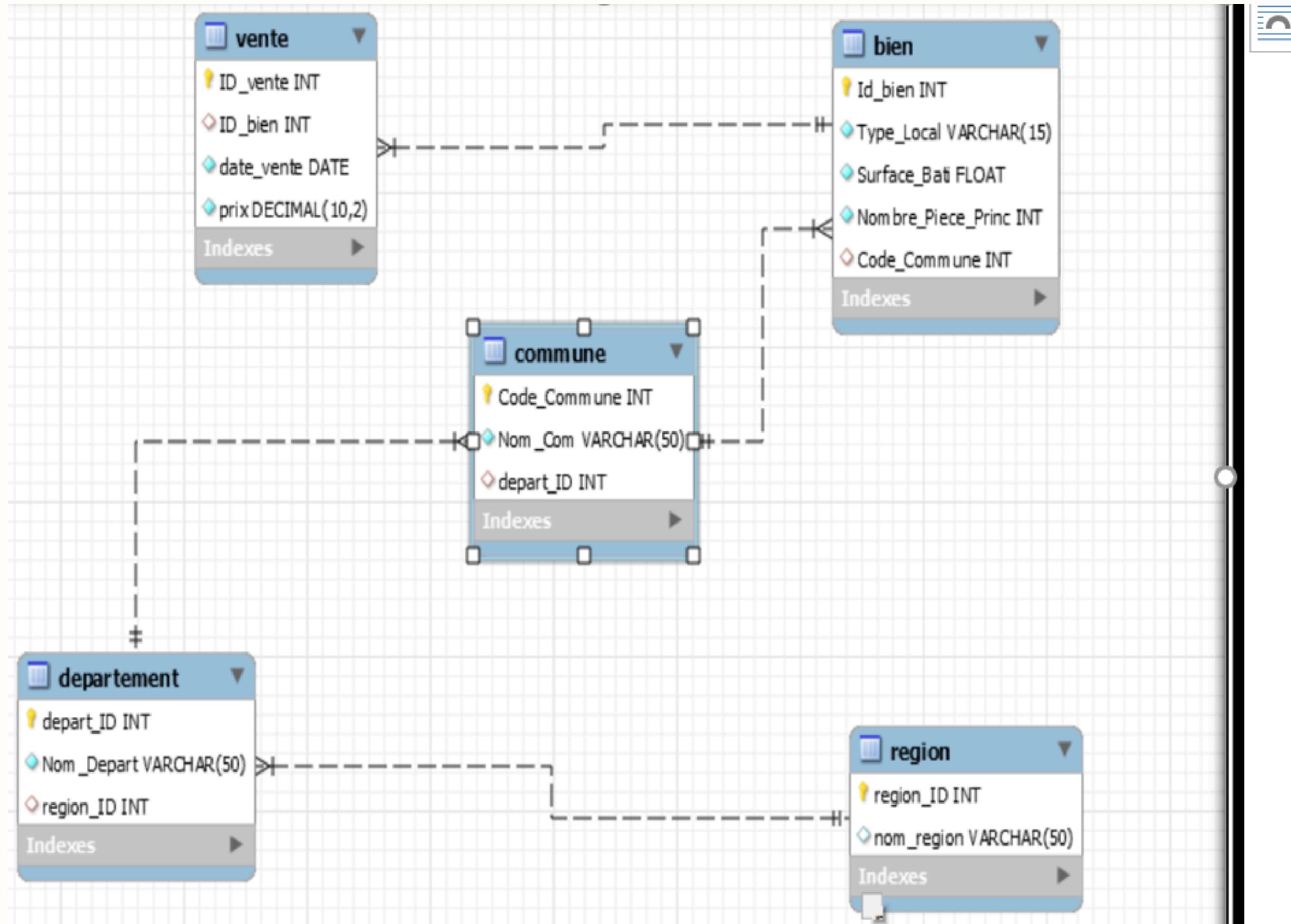
DROP TABLE IF EXISTS `departement`;
/*!40101 SET @saved_cs_client      = @@character_set_client */;
SET character_set_client = utf8mb4 ;
CREATE TABLE `departement` (
  `depart_ID` int NOT NULL,
  `Nom_Depart` varchar(50) NOT NULL,
  `region_ID` int DEFAULT NULL,
  PRIMARY KEY (`depart_ID`),
  KEY `region_ID_idx` (`region_ID`),
  CONSTRAINT `region_ID` FOREIGN KEY (`region_ID`) REFERENCES `region` (`region_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

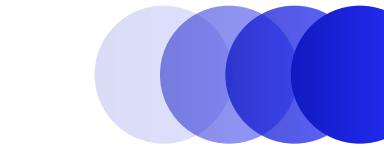
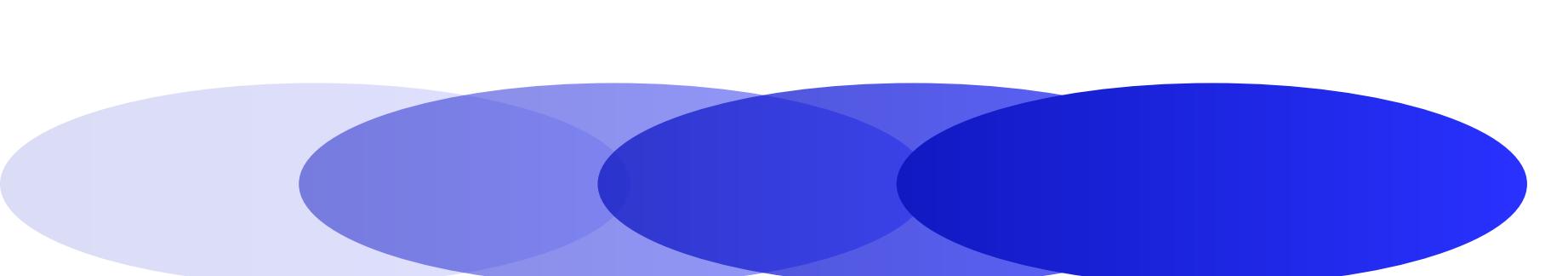
DROP TABLE IF EXISTS `region`;
/*!40101 SET @saved_cs_client      = @@character_set_client */;
SET character_set_client = utf8mb4 ;
CREATE TABLE `region` (
  `region_ID` int NOT NULL,
  `nom_region` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`region_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

DROP TABLE IF EXISTS `vente`;
/*!40101 SET @saved_cs_client      = @@character_set_client */;
SET character_set_client = utf8mb4 ;
CREATE TABLE `vente` (
  `ID_vente` int NOT NULL AUTO_INCREMENT,
  `ID_bien` int DEFAULT NULL,
  `date_vente` date NOT NULL,
  `prix` decimal(10,2) NOT NULL,
  PRIMARY KEY (`ID_vente`),
  KEY `ID_bien_idx` (`ID_bien`),
  CONSTRAINT `ID_bien` FOREIGN KEY (`ID_bien`) REFERENCES `bien` (`Id_bien`)
)
```

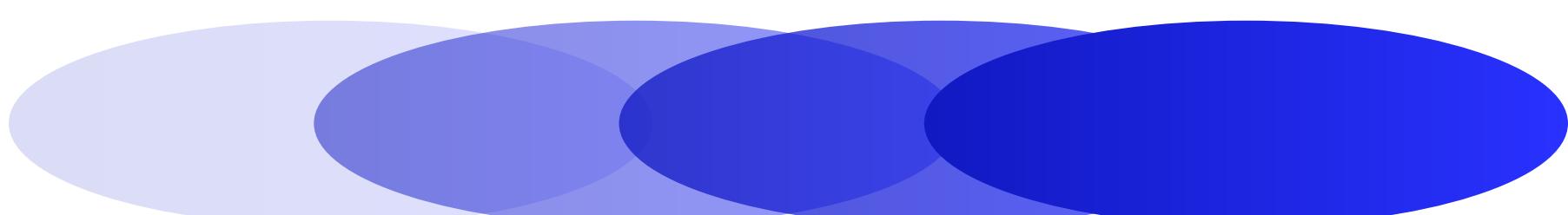
Schéma relationnel

Voici présenté le shema relationnel normalisé de notre base de données





Execution des requêtes



```

//REQUETE 1-A (appartement)
//L objectif de cette requête SQL est de fournir un aperçu des ventes d appartements
//entre le 1er janvier et le 30 juin 2020, en regroupant les résultats par région.

SELECT
    Type_Local AS 'Type de bien',
    COUNT(Type_Local) AS 'Nombre de biens vendus',
    departement.region_ID AS 'Région',
    Nom_region AS 'Nom Région'
FROM
    vente
    LEFT JOIN bien USING (ID_bien)
    LEFT JOIN commune USING (Code_Commune)
    LEFT JOIN departement USING (depart_ID)
    LEFT JOIN region USING (region_ID)
WHERE
    (date_vente >= '2020-01-01')
    AND (date_vente <= '2020-06-30')
    AND LOWER(Type_Local) = 'appartement'
GROUP BY departement.region_ID, Type_Local
ORDER BY COUNT(Type_Local) DESC

//REQUETE 1-B (maison)
//L objectif de cette requête SQL est de fournir un aperçu des ventes de
//maison entre le 1er janvier et le 30 juin 2020, en regroupant les résultats par région.

SELECT
    Type_Local AS 'Type de bien',
    COUNT(Type_Local) AS 'Nombre de biens vendus',
    departement.region_ID AS 'Région',
    Nom_region AS 'Nom Région'
FROM
    vente
    LEFT JOIN bien USING (ID_bien)
    LEFT JOIN commune USING (Code_Commune)
    LEFT JOIN departement USING (depart_ID)
    LEFT JOIN region USING (region_ID)
WHERE
    (date_vente >= '2020-01-01')
    AND (date_vente <= '2020-06-30')
    AND LOWER(Type_Local) = 'maison'
GROUP BY departement.region_ID, Type_Local
ORDER BY COUNT(Type_Local) DESC

```

Type de bien	Nombre de biens vendus	Région	Nom Région
Appartement	13839	11	Ile-de-France
Appartement	3180	93	Provence-Alpes-Côte d'Azur
Appartement	3036	84	Auvergne-Rhône-Alpes
Appartement	2262	NULL	NULL
Appartement	1648	75	Nouvelle-Aquitaine
Appartement	1333	76	Occitanie
Appartement	1115	32	Hauts-de-France
Appartement	1056	52	Pays de la Loire
Appartement	983	44	Grand Est
Appartement	947	53	Bretagne
Appartement	808	28	Normandie
Appartement	645	24	Centre-Val de Loire

Type de bien	Nombre de biens vendus	Région	Nom Région
Maison	1048	11	Ile-de-France
Maison	324	75	Nouvelle-Aquitaine
Maison	254	93	Provence-Alpes-Côte d'Azur
Maison	242	76	Occitanie
Maison	224	84	Auvergne-Rhône-Alpes
Maison	159	52	Pays de la Loire
Maison	137	NULL	NULL
Maison	101	32	Hauts-de-France
Maison	72	24	Centre-Val de Loire
Maison	71	28	Normandie
Maison	61	53	Bretagne
Maison	54	44	Grand Est
Maison	25	27	Bourgogne-Franche-Comté
Maison	12	2	Martinique
Maison	5	3	Guyane
Maison	2	4	La Réunion

```

// REQUETE 2 A (maison)
// L objectif de cette requête SQL est d analyser les ventes de maisons vendues et d appartement
// entre le 1er janvier et le 30 juin 2020, en mettant en évidence la distribution des ventes
// par nombre de pièces principales.
SELECT
    Type_Local AS 'Type de bien',
    Nbre_Piece_prin AS 'Nb de pièces',
    CONCAT(
        FORMAT(
            COUNT(Type_Local) * 100 /
            (SELECT COUNT(Type_Local) FROM bien WHERE LOWER(Type_Local) = 'appartement'),
            4
        ),
        '%'
    ) AS '% du total des ventes'
FROM
    bien
JOIN vente ON bien.ID_bien = vente.ID_bien
WHERE
    (date_vente >= '2020-01-01')
    AND (date_vente <= '2020-06-30')
    AND LOWER(Type_Local) = 'maison'
GROUP BY Type_Local, Nbre_Piece_prin
ORDER BY Nbre_Piece_prin ASC

```

// REQUETE 2 B (appartement)

```

SELECT
    Type_Local AS 'Type de bien',
    Nbre_Piece_prin AS 'Nb de pièces',
    CONCAT(
        FORMAT(
            COUNT(Type_Local) * 100 /
            (SELECT COUNT(Type_Local) FROM bien WHERE LOWER(Type_Local) = 'maison'),
            4
        ),
        '%'
    ) AS '% du total des ventes'
FROM
    bien
JOIN vente ON bien.ID_bien = vente.ID_bien
WHERE
    (date_vente >= '2020-01-01')
    AND (date_vente <= '2020-06-30')
    AND LOWER(Type_Local) = 'maison'
GROUP BY Type_Local, Nbre_Piece_prin
ORDER BY Nbre_Piece_prin ASC

```

	Type de bien	Nb de pièces	% du total des ventes
4	Appartement	0	0.0956%
5	Appartement	1	21.4761%
6	Appartement	2	31.1769%
7	Appartement	3	28.5732%
8	Appartement	4	14.2133%
9	Appartement	5	3.5501%
10	Appartement	6	0.6501%
11	Appartement	7	0.1721%
12	Appartement	8	0.0542%
13	Appartement	9	0.0255%
14	Appartement	10	0.0064%
15	Appartement	11	0.0032%
16			
17			

	Type de bien	Nb de pièces	% du total des ventes
	Maison	0	0.1075%
	Maison	1	2.1856%
	Maison	2	9.5665%
	Maison	3	21.5693%
	Maison	4	38.4808%
	Maison	5	19.9570%
	Maison	6	5.9119%
	Maison	7	1.5765%
	Maison	8	0.3941%
	Maison	9	0.1791%
	Maison	10	0.0717%

```

// REQUETE 3 A (Maison)
// L objectif de cette requête SQL est d analyser les ventes d appartements et de maison
// en les regroupant par nombre de pièces principales et par région pour la période entre le 1er janvier et le 30 juin 2020

SELECT
    Type_Local AS 'Type de bien',
    Nbre_Piece_prin AS 'Nb de pièces',
    CONCAT(
        FORMAT(
            COUNT(Type_Local) * 100 /
            (SELECT COUNT(Type_Local) FROM bien WHERE LOWER(Type_Local) = 'maison'),
            4
        ),
        '%'
    ) AS '% du total des ventes',
    region.Nom_region AS 'Région' -- Ajout de la colonne région
FROM
    bien
JOIN vente ON bien.ID_bien = vente.ID_bien
JOIN commune ON bien.Code_Commune = commune.Code_Commune -- Jointure avec la table commune
JOIN departement ON commune.depart_ID = departement.depart_ID -- Jointure avec la table département
JOIN region ON departement.region_ID = region.region_ID -- Jointure avec la table région
WHERE
    (date_vente >= '2020-01-01')
    AND (date_vente <= '2020-06-30')
    AND LOWER(Type_Local) = 'maison'
GROUP BY Type_Local, Nbre_Piece_prin, region.Nom_region
ORDER BY Nbre_Piece_prin ASC

// REQUETE 3 B (appartement)

SELECT
    Type_Local AS 'Type de bien',
    Nbre_Piece_prin AS 'Nb de pièces',
    CONCAT(
        FORMAT(
            COUNT(Type_Local) * 100 /
            (SELECT COUNT(Type_Local) FROM bien WHERE LOWER(Type_Local) = 'appartement'),
            4
        ),
        '%'
    ) AS '% du total des ventes',
    region.Nom_region AS 'Région' -- Ajout de la colonne région
FROM
    bien
JOIN vente ON bien.ID_bien = vente.ID_bien
JOIN commune ON bien.Code_Commune = commune.Code_Commune -- Jointure avec la table commune
JOIN departement ON commune.depart_ID = departement.depart_ID -- Jointure avec la table département
JOIN region ON departement.region_ID = region.region_ID -- Jointure avec la table région
WHERE
    (date_vente >= '2020-01-01')
    AND (date_vente <= '2020-06-30')
    AND LOWER(Type_Local) = 'appartement'
GROUP BY Type_Local, Nbre_Piece_prin, region.Nom_region
ORDER BY Nbre_Piece_prin ASC

```

Type de bien	Nb de pièces	% du total des ventes	Région
Appartement	0	0.0159%	Provence-Alpes-Côte d'Azur
Appartement	0	0.0127%	Auvergne-Rhône-Alpes
Appartement	0	0.0414%	Ile-de-France
Appartement	0	0.0064%	Grand Est
Appartement	0	0.0032%	Bretagne
Appartement	0	0.0032%	Hauts-de-France
Appartement	0	0.0032%	Normandie
Appartement	0	0.0032%	Centre-Val de Loire
Appartement	0	0.0032%	Pays de la Loire
Appartement	1	2.6483%	Provence-Alpes-Côte d'Azur

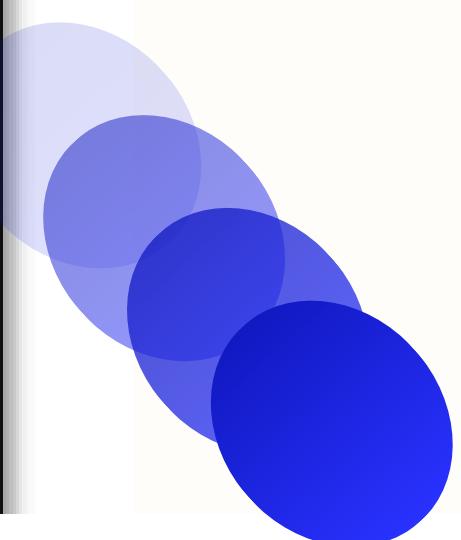
Type de bien	Nb de pièces	% du total des ventes	Région
Maison	0	0.0358%	Ile-de-France
Maison	0	0.0358%	Normandie
Maison	0	0.0358%	Pays de la Loire
Maison	1	0.2866%	Nouvelle-Aquitaine
Maison	1	0.3941%	Ile-de-France
Maison	1	0.1075%	Centre-Val de Loire
Maison	1	0.0717%	Hauts-de-France
Maison	1	0.3583%	Pays de la Loire
Maison	1	0.0717%	Normandie
Maison	1	0.2866%	Occitanie

// REQUETE 4

//L objectif de cette requête SQL est d analyser la valeur foncière moyenne par mètre
//carré pour un certain type de bien immobilier dans une région spécifique, pour une période donnée.
//Elle regroupe et structure les informations sur les biens vendus afin de donner
//une vue d ensemble sur les prix moyens des biens en fonction des régions et des types de biens

SELECT

```
Type_Local AS 'Type de bien', /* Sélectionner le type de bien */  
CONCAT(FORMAT(AVG(prix / Surface_Bati), 2), ' EUR/m²') AS 'Valeur foncière moyenne', /* Calculer la valeur foncière moyenne par m² */  
region.region_ID AS 'Région', /* Récupérer l'ID de la région */  
Nom_region AS 'Nom Région' /* Récupérer le nom de la région */  
FROM  
    vente /* Table des ventes */  
LEFT JOIN bien ON bien.ID_bien = vente.ID_bien /* Jointure avec la table des biens */  
LEFT JOIN commune ON bien.Code_Commune = commune.Code_Commune /* Jointure avec la table des communes */  
LEFT JOIN département ON commune.depart_ID = département.depart_ID /* Jointure avec la table des départements */  
LEFT JOIN region ON département.region_ID = region.region_ID /* Jointure avec la table des régions */  
WHERE  
    (date_vente >= '2020-01-01') /* Filtrer les ventes à partir du 1er janvier 2020 */  
    AND (date_vente <= '2020-06-30') /* Filtrer les ventes jusqu'au 30 juin 2020 */  
    AND LOWER(Type_Local) = '$type_bien' /* Filtrer par type de bien (insensible à la casse) */  
    AND region.region_ID = '$region_id' /* Filtrer par ID de région */  
GROUP BY region.region_ID, Type_Local /* Regrouper par ID de région et type de bien */
```



L'objectif de cette requête SQL est d'analyser la valeur foncière moyenne par mètre carré pour un certain type de bien immobilier dans une région spécifique, pour une période donnée. Elle regroupe et structure les informations sur les biens vendus afin de donner une vue d'ensemble sur les prix moyens des biens en fonction des régions et des types de biens (Voir site web pour le résultats).

// Requete 5

```
// Pour savoir le nombre de vente du premier trimestre et du deuxième trimestre de 2020(CTE)
WITH Trim1 AS (
    SELECT COUNT(ID_bien) AS Ventes20Q1
    FROM vente
    WHERE date_vente >= '2020-01-01' AND date_vente <= '2020-03-31'
),
Trim2 AS (
    SELECT COUNT(ID_bien) AS Ventes20Q2
    FROM vente
    WHERE date_vente >= '2020-04-01' AND date_vente <= '2020-06-30'
)
SELECT Trim1.Ventes20Q1, Trim2.Ventes20Q2
FROM Trim1, Trim2;
```

Nombre de vente par semestre

Ventes20Q1	Ventes20Q2
16776	17393

/ Requete 6

```
// Afficher le prix moyens pour les appartements de plus de 4 pièces par région

WITH moyenne AS (
    SELECT region.region_ID AS 'Région',
           region.Nom_region AS 'NomRégion',
           AVG(vente.prix / bien.Surface_Bati) AS 'AvgPpxM2Apparts4p'
      FROM vente
     LEFT JOIN bien ON vente.ID_bien = bien.ID_bien
     LEFT JOIN commune ON bien.Code_Commune = commune.Code_Commune
     LEFT JOIN departement ON commune.depart_ID = departement.depart_ID
    LEFT JOIN region ON departement.region_ID = region.region_ID
   WHERE (vente.date_vente >= '2020-01-01')
     AND (vente.date_vente <= '2020-06-30')
     AND lower(bien.Type_Local) = 'appartement'
     AND bien.Nbre_Piece_prin > 4
  GROUP BY region.region_ID, region.Nom_region
)
SELECT Région,
       NomRégion AS 'Nom Région',
       CONCAT(FORMAT(AvgPpxM2Apparts4p, 2), ' EUR/m2') AS 'Prix moyen appartements > 4 pièces'
  FROM moyenne
 ORDER BY AvgPpxM2Apparts4p DESC;
```

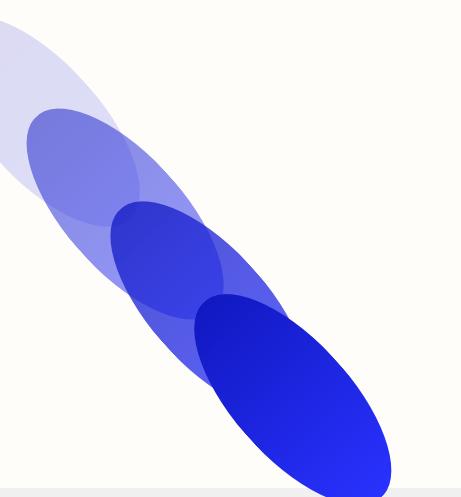
Afficher le prix moyens pour les appartements de plus de 4 pièces par région

Région	Nom Région	Prix moyen appartements > 4 pièces
11	Ile-de-France	8,030.81 EUR/m ²
4	La Réunion	4,801.67 EUR/m ²
NULL	NULL	3,034.71 EUR/m ²
93	Provence-Alpes-Côte d'Azur	2,942.35 EUR/m ²
84	Auvergne-Rhône-Alpes	2,803.59 EUR/m ²
53	Bretagne	2,234.71 EUR/m ²
32	Hauts-de-France	2,219.64 EUR/m ²
75	Nouvelle-Aquitaine	2,149.55 EUR/m ²

// Requête 7

// Comparaison prix moyen appartements 3 pièces et 2 pièces

```
WITH Moy2pieces AS (
    SELECT AVG(prix / Surface_Bati) AS Apparts2Pieces
    FROM vente
    JOIN bien USING (ID_bien)
    WHERE (date_vente >= '2020-01-01')
        AND (date_vente <= '2020-06-30')
        AND LOWER(Type_Local) = 'appartement'
        AND Nbre_Piece_prin = '2'
),
Moy3pieces AS (
    SELECT AVG(prix / Surface_Bati) AS Apparts3Pieces
    FROM vente
    JOIN bien USING (ID_bien)
    WHERE (date_vente >= '2020-01-01')
        AND (date_vente <= '2020-06-30')
        AND LOWER(Type_Local) = 'appartement'
        AND Nbre_Piece_prin = '3'
)
SELECT
    CONCAT(FORMAT(Apparts2Pieces, 2), ' EUR/m2') AS 'Prix moyen appartements 2 pièces',
    CONCAT(FORMAT(Apparts3Pieces, 2), ' EUR/m2') AS 'Prix moyen appartements 3 pièces',
    CONCAT(FORMAT((Apparts3Pieces - Apparts2Pieces) * 100 / Apparts2Pieces, 2), '%') AS 'Ecart'
FROM
    Moy2pieces
JOIN
    Moy3pieces;
```



	Prix moyen appartements 2 pièces	Prix moyen appartements 3 pièces	Ecart
	4,932.85 EUR/m ²	4,285.37 EUR/m ²	-13.13%

// Requete 8

// Afficher les informations sur les communes ayant un nombre de ventes supérieur à 50

```
SELECT
    commune.Code_Commune AS 'Commune',
    commune.Nom_Com AS 'Nom commune',
    COUNT(vente.ID_vente) AS 'Nb de ventes',
    CONCAT(' Du ', MIN(date_vente), ' au ', MAX(date_vente)) AS 'Période'
FROM
    vente
        LEFT JOIN bien USING (ID_bien)
        LEFT JOIN commune USING (Code_Commune)
        LEFT JOIN departement USING (depart_ID)
        LEFT JOIN region USING (region_ID)
WHERE
    date_vente >= '2020-01-01'
    AND date_vente <= '2020-03-31'
GROUP BY commune.Code_Commune
HAVING COUNT(vente.ID_vente) >= 50
ORDER BY commune.Code_Commune DESC;
```

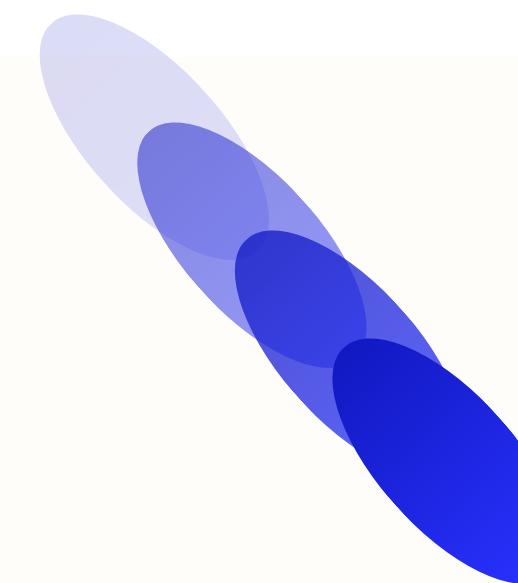
Commune	Nom commune	Nb de ventes	Période
901	SETE	62	Du 2020-01-06 au 2020-03-30
331	LA CIOTAT	62	Du 2020-01-02 au 2020-03-20
327	MARSEILLE 8EME	81	Du 2020-01-02 au 2020-03-16
325	MARSEILLE 9EME	66	Du 2020-01-03 au 2020-03-30
3214	PARIS 02	61	Du 2020-01-03 au 2020-03-30
3213	PARIS 04	60	Du 2020-01-03 au 2020-03-25
3212	PARIS 19	116	Du 2020-01-08 au 2020-03-31
3211	PARIS 20	127	Du 2020-01-03 au 2020-03-27
3210	PARIS 18	209	Du 2020-01-02 au 2020-03-31
3209	PARIS 17	228	Du 2020-01-03 au 2020-03-31
3208	PARIS 16	165	Du 2020-01-06 au 2020-03-31
3207	PARIS 15	215	Du 2020-01-02 au 2020-03-31
3206	PARIS 14	146	Du 2020-01-03 au 2020-03-31
3205	PARIS 13	94	Du 2020-01-06 au 2020-03-31
3204	PARIS 12	110	Du 2020-01-02 au 2020-03-27
3203	PARIS 11	169	Du 2020-01-02 au 2020-03-31
3202	PARIS 09	106	Du 2020-01-07 au 2020-03-31
3201	PARIS 10	109	Du 2020-01-02 au 2020-03-30
3200	PARIS 05	79	Du 2020-01-02 au 2020-03-30
3199	PARIS 07	87	Du 2020-01-03 au 2020-03-27
3198	PARIS 06	86	Du 2020-01-03 au 2020-03-31
3197	PARIS 08	62	Du 2020-01-10 au 2020-03-20
3195	PARIS 03	79	Du 2020-01-03 au 2020-03-31
315	MARSEILLE 1ER	71	Du 2020-01-03 au 2020-03-30
314	MARSEILLE 4EME	72	Du 2020-01-03 au 2020-03-16
3061	VINCENNES	68	Du 2020-01-02 au 2020-03-27

```

// Requête 9
/*L'objectif de cette requête SQL est d'extraire les 3 communes les plus chères par département pour plusieurs départements français sélectionnés,
en se basant sur la valeur foncière moyenne des biens immobiliers vendus. Cette analyse permet de comparer la valeur foncière moyenne dans différentes
communes au sein de chaque département, en mettant en évidence les communes ayant les prix moyens les plus élevés.*/

WITH OrdreCommunesParDept AS(
SELECT departement.depart_ID AS 'Département', commune.Code_Commune AS Commune, CONCAT(FORMAT(AVG(prix),2), ' EUR') AS 'Valeur foncière moyenne',
RANK() OVER(PARTITION BY departement.depart_ID ORDER BY AVG(prix) DESC) AS 'Classement'
FROM bien
LEFT JOIN vente USING (ID_bien)
LEFT JOIN commune USING (Code_Commune)
LEFT JOIN departement USING (depart_ID)
GROUP BY Code_Commune
)
SELECT * FROM OrdreCommunesParDept
WHERE ((Classement <=3) AND (Département = '06')) OR ((Classement <=3) AND (Département = '13')) OR ((Classement <=3)
AND (Département = '33'))
OR ((Classement <=3) AND (Département = '59'))
OR ((Classement <=3) AND (Département = '69'))
;

```



	Département	Commune	Valeur foncière moyenne	Classement
▶	13	356	330,000.00 EUR	1
	13	336	314,425.00 EUR	2
	13	329	313,416.88 EUR	3
	33	822	695,051.00 EUR	1
	33	873	335,000.00 EUR	2
	33	809	307,435.93 EUR	3
	59	1665	433,202.00 EUR	1
	59	1662	408,550.00 EUR	2
	59	1672	322,250.00 EUR	3
	69	2023	485,300.00 EUR	1
	69	1906	455,217.26 EUR	2
	69	1929	426,968.25 EUR	3

voici mon premier

Bouger

Je suis Grand

Nouveau ?

Rescent ?

oui

non

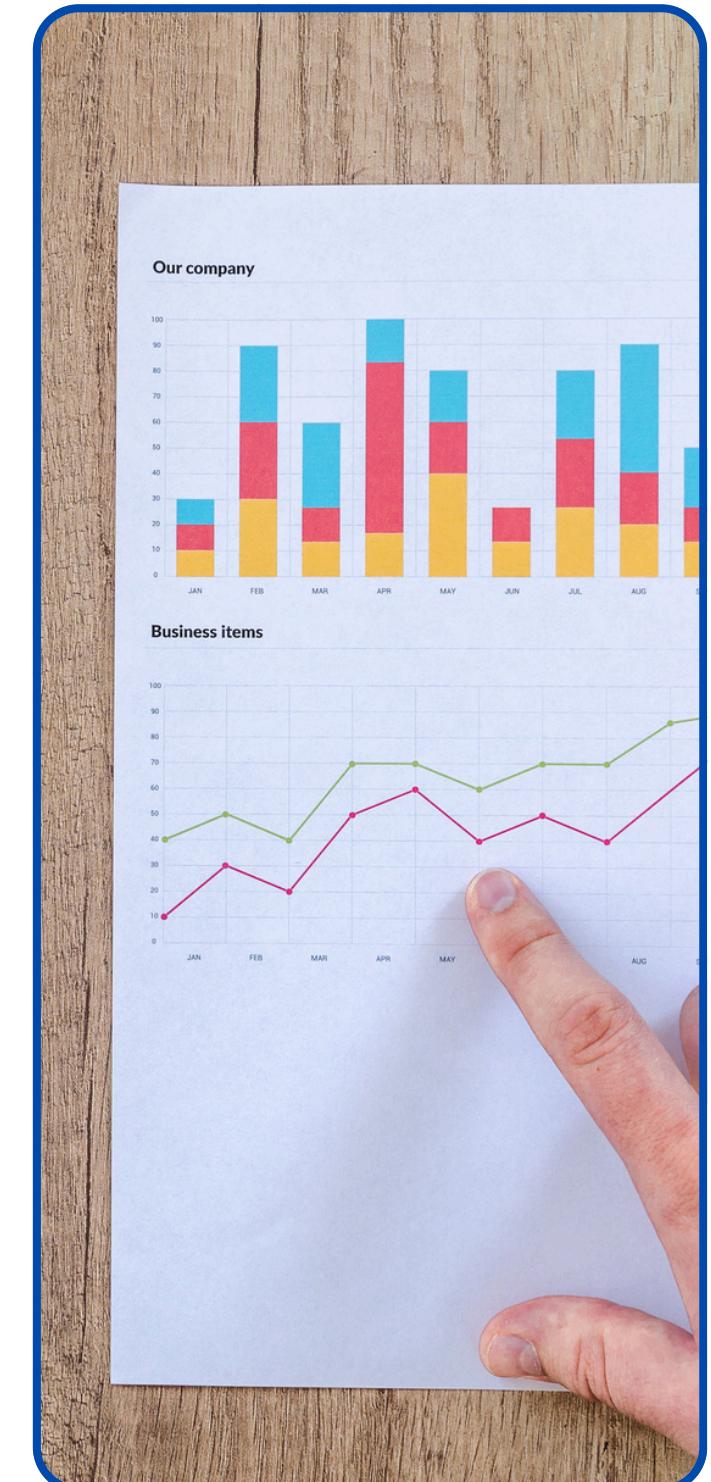
ABoubacry
Alhouseynou
Mamadou

Site web de Laplace Immo



CONCLUSION

La création de cette base de données pour Immo Laplace représente une avancée significative dans l'exploitation des données immobilières. En centralisant et en structurant les informations clés de l'entreprise, nous avons développé un outil qui permet non seulement de répondre efficacement aux besoins actuels de gestion de données, mais aussi de soutenir les analyses prédictives et les prises de décisions stratégiques. Grâce à l'intégration sur AWS RDS et l'interaction facilitée via une interface PHP, cette solution offre une infrastructure évolutive et sécurisée. Ce projet ouvre des perspectives prometteuses pour Immo Laplace, en lui permettant d'anticiper les fluctuations du marché immobilier et de renforcer sa compétitivité. À l'avenir, l'entreprise pourra ainsi optimiser ses stratégies de vente et affiner ses prévisions pour répondre aux exigences d'un secteur en mutation rapide.



**MERCIE DE
VOTRE
AIMABLE
ATTENTION**