## 1 – Modélisation

### 1.1 – Dictionnaire des données

La première étape dans la mise en place de la base a été l'élaboration du dictionnaire des données, qui recense tous les champs de données inclus dans la base, ainsi que leur signification, leur format, pour certains aussi leurs contraintes et bien évidemment un champ commentaire pour permettre de garder une trace de la source originelle de ces données pour faciliter, au besoin, les évolutions futures de la base.

Les fichiers de départ sont relativement lourds et complexes, étant très riches en information, et il a fallu sélectionner avec soin les champs pertinents pour répondre à nos besoins de données, en sélectionnant tous les champs utiles, mais seulement les champs utiles, pour ne pas alourdir inutilement la base.

Cette démarche a été menée en consultant le descriptif des fichiers d'origine, qui explique la source des données (voir dossier donnees-dot-gouv-dot-fr), et a abouti à la sélection des 17 champs de données suivants :

- tout d'abord les biens (table bien) : ils sont caractérisés par un type (maison ou appartement), une surface Loi Carrez, une surface bâtie totale et un nombre de pièces (ce sont les champs bien_type, bien_carrez_1, bien_surfbat et bien_numpieces respectivement) mais aussi et surtout par une chaîne de caractères unique qui permet de les identifier, c'est l'identifiant bien_id. On aurait pu choisir une chaîne numérique auto-incrémentée pour identifier ces biens, méthodologiquement ça n'est pas incorrect, mais ici on a utilisé une concaténation de chaines de caractères correspondant à des identifiants intrinsèques de ces biens, et notamment les codes département et commune, suivis de 3 identifiants cadastraux uniques – cette méthodologie semblait en effet plus à même de garantir l'unicité des biens présents dans la base. Pour les besoins de la base, seuls les maisons et les appartements sont inclus, il n'a donc pas été jugé utile d'introduire un numéro d'identification des types de biens, mais cela pourrait être relativement facilement ajouté si le besoin de faire évoluer la base se faisait sentir par la suite, par exemple pour s'intéresser aux biens industriels et non plus seulement résidentiels ;

- les communes ensuite (table commune) : comme on le sait, le code postal n'est pas un bon identifiant, car plusieurs communes ou lieux-dits peuvent partager le même code postal. Ici encore, c'est une concaténation de chaînes de caractères qui a été choisie comme identifiant primaire, celle des champs code département et code commune des fichiers de recensement de population de l'INSEE, afin encore une fois de garantir l'unicité des communes présentes dans la table et de leurs identifiants. En plus de cet identifiant, chaque commune est caractérisée par un nom (com_nom) et une population totale (com_pop_tot) ;

- les transactions sur les biens ensuite (table mutation) : chaque bien pouvant changer de mains plusieurs fois, potentiellement à la même date, il a fallu étendre les identifiants des transactions non seulement à leur date, mais à leur numéro de disposition à cette date (mut_numdisp) pour en garantir l'identification certaine et donc l'unicité. Les mutations sont en outre caractérisées par une valeur foncière (mut_val) et un type de mutation (mut_type). Pour les besoins de notre base, seules les mutations à titre onéreux qui sont des ventes sont incluses, à l'exclusion de toutes autres. Il n'a donc pas été jugé utile ici non plus d'inclure une table de codes pour les types de mutation à titre onéreux, mais cela pourrait facilement être rajouté si on ressentait un jour le besoin de faire évoluer la base pour répondre à d'autres besoins de données, comme par exemple les ventes de biens inachevés ou de terrains à bâtir. Veiller dans tous les cas à ne pas nommer cette table « transaction » qui est un mot clé (donc réservé) dans MySQL ;

- les départements ensuite (tables departement et ref_dep) : ils sont caractérisés par un numéro (ref_dep_code et dep_code) et un nom (dep_nom) ;

- les régions enfin (table region) : caractérisées elles aussi par un numéro (reg_code) et un nom (reg_nom).

1.2 – Schéma relationnel

Les données ont été organisées en 6 tables au niveau du schéma relationnel

- les biens : avec leur identifiant unique bien_id et leurs autres caractéristiques. Ils sont reliés aux…

- … mutations : par ce même identifiant bien_id, et la date et le numéro de disposition finissent de nous donner l'identifiant primaire unique des mutations. La relation ici, c'est qu'une mutation ne porte que sur un bien unique mais chaque bien peut faire l'objet de plusieurs mutations dans le temps, ou même dans la même journée, comme l'indique la partie droite de la flèche, d'où les identifiants choisis sur la table mutation. Les biens sont aussi reliés aux…

- … communes, par le biais de l'identifiant primaire com_code, chaque bien étant situé dans une et une seule commune (partie basse de la flèche), et chaque commune pouvant évidemment regrouper plusieurs biens (partie haute de la flèche). Ces communes sont elles-mêmes reliées aux …

- …départements par l'identifiant dep_code, lesquels sont eux-mêmes reliés au référentiel de leurs noms par l'identifiant ref_dep_code et reliés aux …

- régions par l'identifiant reg_code. Ici, on voit qu'un département n'a qu'un seul nom dans le référentiel des noms, et qu'il n'appartient qu'à une seule région, mais qu'une région peut évidemment inclure plusieurs départements.

Ce schéma relationnel est donc ce que l'on appelle « normalisé », il répond aux trois premières formes normales :

- 1NF : chaque table a bien un identifiant, une clé, primaire, et chaque champ d'information dans cette table, chaque attribut, ne contient qu'une seule information (ont dit qu'il est atomique, par opposition à composite ou multivalué) ; de plus…

- 2NF : …tout champ d'information, tout attribut, qui ne fait pas partie d'un identifiant primaire, d'une clé, ne dépend pas seulement que d'une partie de cette clé ; et enfin…

- 3NF : … tout champ d'information, tout attribut, qui ne fait pas partie d'un identifiant primaire, d'une clé, ne dépend pas d'un autre attribut qui lui non plus ne fait pas partie de la clé.

## 2 - Implémentation

2.1 - Tri & sélection des données

Deux outils principaux ont été utilisés :

- Excel, dans lequel un certain nombre de retraitements ont été effectués afin de garantir la qualité des données, et en particulier une inspection des fichiers, l'identification de la dernière cellule utilisée, la vérification de la cohérence des données (en insérant une moyenne sur un champ nombre par exemple, pour vérifier que le calcul tourne et qu'il n'y a pas par exemple une donnée texte égarée à la ligne 25,000), la vérification de l'absence de blancs dans les chaînes de texte et de cellules vides dans les fichiers, particulièrement dans les colonnes servant à produire les identifiants primaires, et enfin la création des concaténations de chaînes de caractères pour les identifiants primaires des biens et des communes. Sur la vérification de l'absence de blancs par exemple, on observe que 18 ventes au total n'avaient pas de prix, mais on les a conservées dans la base car au regard du nombre total de plus de 30,000 ventes, c'est statistiquement négligeable. Ces vérifications ont également permis d'identifier que dans le fichier source, l'identifiant du type de mutation était blanc (NULL) pour toutes les ventes du premier trimestre ; ces identifiants ont dont été renseignés à la main (attention toutefois cette solution ne s'applique ici que parce que les autres types de mutation à titre onéreux sont exclus de notre fichier de départ). Enfin, ces vérifications ont permis de déceler des problèmes d'import UTF8 dans le fichier originel (et qui par la suite se sont avéré bloquer l'import des données dans MySQL) et que plusieurs caractères ont donc dû être remplacés en utilisant la fonction « cherche/remplace » d'Excel.

Ces vérifications une fois effectuées…

- …PowerQuery, a permis de mettre en forme les tables de données présentes dans Excel, de les formatter au format des champs de la base (texte, nombre etc…), de supprimer les colonnes inutiles, de renommer les colonnes utiles d'après leurs noms dans la base pour faciliter l'import et de vérifier l'absence de doublons dans les tables ;

- enfin les données ont été exportées au format .csv. Éventuellement, pour gagner du temps, on pourra utiliser Python pour uploader des données dans la base (voir le notebook Jupyter inclus à cet effet). Si l'on choisit d'utiliser le Table Data Import Wizard de MySQL Workbench, veiller à bien uploader les tables dans l'ordre de dépendance des clés étrangères ou à désactiver la vérification des clés étrangères avant l'import (et à la réactiver après !).

### 2.2 - Choix des outils informatiques & création de la base

Le schéma relationnel de la base créé avec MySQL Workbench, ainsi que le code du script de création sont inclus.

### 2.3 – Formatage & chargement des données

Le formatage des données a été effectué avec Excel et PowerQuery, suivi d'un export au format .csv, qui est l'un des formats acceptés par MySQL pour le chargement des tables de la base.

Les données peuvent être chargées directement dans MySQL Workbench en utilisant le Table Data Import Wizard (accessible par un clic droit sur la table concernée), mais pour les tables les plus complexes comme bien ou mutation, le chargement peut prendre plusieurs minutes ; il est quasi-instantané avec le script Python. Dans les 2 cas, les données à utiliser sont celles du dossier « donnees-nettoyees ».

## 3 – Exploitation

### 3.1 - Principes génériques de construction des requêtes

Les requêtes ont été créées en utilisant un certain nombre de concepts fondamentaux du langage SQL :

- des projections : c'est la fonction SELECT, qui permet de sélectionner les valeurs de certaines colonnes de la base ;

- des restrictions : c'est la fonction WHERE, qui permet de spécifier les contraintes auxquelles ces valeurs doivent répondre. Ces contraintes peuvent être modulées avec des opérateurs comme AND et OR, selon le besoin ;

- des agrégations : soit par des formules (ex. moyenne AVG) qui transforment plusieurs données en une seule, soit par des fonctions comme GROUP BY, qui permettent de grouper certaines lignes de résultat ; et enfin

- des jointures : représentées par la fonction JOIN et ses variantes (LEFT JOIN, etc…), qui permettent d'unir deux ou plusieurs tables pour aller y chercher des informations dans plusieurs colonnes.

D'autres fonctions comme AS pour les alias, FORMAT et CONCAT ont été utilisées pour des raisons « cosmétiques » de clarté de présentation des résultats, pour pouvoir leur donner des noms lisibles et un format (monétaire ou autre) facilement compréhensible.

Parfois, de groupements de fonctions appelées CTE (common table expression) ont été utilisées pour éviter la répétition de syntaxe dans les requêtes et faciliter leur lecture et leur interprétation (exemple des requêtes 7 et 11).

3.2 - Discussion des résultats

**- requête 1 :**

Problématique : calculer le nombre total d'appartements vendus au 1er semestre 2020 ;

Résultat : 31,375 appartements ont été vendus au premier semestre 2020 ;

**- requête 2 :**

Problématique : calculer le nombre de ventes d'appartement par région pour le 1er semestre 2020 ;

Résultat : La région Ile de France arrive en tête des ventes d'appartements au premier semestre 2020 avec 13,994 ventes ;

**- requête 3 :**

Problématique : calculer la proportion des ventes d'appartements par le nombre de pièces ;

Résultat : Les appartements de 2 pièces ont été les plus vendus au premier semestre 2020, avec plus de 31% des ventes ;

**- requête 4 :**

Problématique : créer une liste des 10 départements où le prix du mètre carré est le plus élevé ;

Résultat : Le département de Paris a le prix au $m^2$ le plus élevé à plus de 12,000 EUR ;

**- requête 5 :**

Problématique : calculer le prix moyen du mètre carré d'une maison en Île-de-France ;

Résultat : Le prix moyen au $m^2$ d'une maison en Île-de-France est de 3,745.01 EUR ;

**- requête 6 :**

Problématique : créer une liste des 10 appartements les plus chers avec la région et le nombre de mètres carrés ;

Résultat : Les 10 appartements les plus chers au $1^{er}$ semestre 2020 étaient tous situés en Île-de-France ;

**- requête 7 :**

Problématique : calculer le taux d'évolution du nombre de ventes entre le premier et le second trimestre de 2020 ;

Résultat : Les ventes ont évolué de +3.56% entre le $2^{ème}$ et le $1^{er}$ trimestre 2020 ;

**- requête 8 :**

Problématique : créer un classement des régions par rapport au prix au mètre carré des appartements de plus de 4 pièces ;

Résultat : La région Île-de-France arrive en tête du classement des régions par rapport au prix du $m^2$ des appartements de plus de 4 pièces, suivie de la Réunion et de la région PACA ;

**- requête 9 :**

Problématique : créer une liste des communes ayant eu au moins 50 ventes au 1er trimestre ;

Résultat : La commune de Paris 17ème arrive en tête des communes ayant eu au moins 50 ventes au 1er trimestre 2020, suivie de Paris 15ème et Paris 18ème. La première commune de province est Nice en 4ème position ;

**- requête 10 :**

Problématique : calculer la différence en pourcentage du prix au mètre carré entre un appartement de 2 pièces et un appartement de 3 pièces ;

Résultat : Les appartements de 3 pièces étaient en moyenne moins chers de 12.34% au $m^2$ que les appartements de 2 pièces au 1er semestre 2020 ;

**- requête 11 :**

Problématique : calculer les moyennes de valeurs foncières pour le top 3 des communes des départements 6, 13, 33, 59 et 69 ;

Résultat : Voici les valeurs foncières moyennes pour les 3 communes les plus chères des départements 06, 13, 33, 59 et 69, avec les villes de Saint-Jean-Cap-Ferrat, Gignac-la-Nerthe, Lège-Cap-Ferret, Bersée et Ville sur Jarnioux arrivant en tête de leurs départements ;

**- requête 12 :**

Problématique : établir la liste des 20 communes avec le plus de transactions pour 1000 habitants pour les communes qui dépassent les 10 000 habitants ;

Résultat : La commune de Paris 2ème arrondissement arrive en tête du classement des villes de plus de 10,000 habitants en terme du nombre de transactions pour 1,000 habitants, suivie des 1er et 3èmes arrondissements. La première commune de province représentée est Arcachon en 4ème position.

***

## AMELIORATIONS POSSIBLES A APPORTER A LA BASE

On pourrait envisager d'enrichir la base avec d'autres types de biens (industriels par exemples) ou d'autres types de mutations (ventes en l'état futur d'achèvement ou terrains à bâtir par exemple) et créer dans la base des tables d'identifiants séparés pour cela.

On pourrait aussi envisager de créer une nouvelle table pour les communes qui permettrait de grouper les arrondissements pour Paris, Lyon et Marseille et de considérer ces agglomérations comme un tout, et non plus de voir chacun de leurs arrondissements comme une commune comme c'est le cas actuellement ; bien que la présentation actuelle réponde à la définition administrative stricte de ce qui constitue une « commune », les agrégats au niveau total de ces villes permettraient d'apporter un éclairage légèrement différent à l'analyse.