

# RÉALISATION D'UN ENTREPÔT DE DONNÉES

---

## Introduction

Pour ce projet, l'objectif donné était de créer un entrepôt de données à partir de plusieurs datasets, afin de pouvoir effectuer un certain nombre de requêtes.

Nous avons réalisé un entrepôt avec des données sur les affaires de corruption en France, ainsi que des statistiques sur la population - comme par exemple l'âge moyen ou le taux de chômage - par département en utilisant MongoDB.

Dans ce rapport sera explicitée la réalisation de ce projet, en abordant d'abord le choix de dataset et de système de base de données, pour enfin détailler la mise en place de l'entrepôt et enfin présenter une liste de requêtes effectuées.

---

## Table des matières

<b>Introduction</b>	<b>1</b>
<b>I. Datasets utilisés</b>	<b>2</b>
1. Affaires de corruption	2
2. Deuxième dataset : données statistiques	3
<b>II. Système de base de donnée : MongoDB</b>	<b>4</b>
<b>III. Intégration des données</b>	<b>5</b>
1. Formatage des données	5
Dataset sur les affaires de corruption	5
Dataset sur la population	6
2. Agrégation des datasets	6
<b>IV. Requêtes</b>	<b>7</b>

# I. Datasets utilisés

Afin de réaliser un entrepôt assez intéressant, il est apparu nécessaire d'utiliser des datasets pouvant être facilement croisés, contenant un nombre assez important de tuples, et sur lesquelles des requêtes intéressantes pourraient être effectuées.

Nous avons donc réalisé notre projet en nous basant sur deux datasets : le premier concernant les affaires de corruptions et le deuxième concernant des statistiques sur la population française.

## 1. Affaires de corruption

[https://public.opendatasoft.com/explore/dataset/affaires-de-corruption-en-france/?disjunctive\\_tags](https://public.opendatasoft.com/explore/dataset/affaires-de-corruption-en-france/?disjunctive_tags)

Réalisé par l'organisation Transparency International France, ce dataset liste les affaires de corruption ayant eu lieu sur le territoire français entre 1981 et 2016.

Les champs de ce dataset incluent une description de l'affaire, une description de la personne ou de l'entité impliquée, la date des faits, la juridiction du jugement, les entités impliquées et les infractions pertinentes. Sont également inclus des champs géographiques - le lieu (ville, village, etc), le département et la région. Ces derniers champs nous ont paru être particulièrement intéressants, car ils permettent de faire des analyses géographiques vis-à-vis de la corruption.

	description	vie publique	personnes ou entités impliquées	date des faits	date de la condamnation	juridiction du jugement
1	L'ancien maire d'Elne a été condamné	oui	L'ancien maire d'Elne	January 1 2010	November 8 2012	Première instance
2	Un médiateur de la ville de Bordeaux a	oui	Un médiateur de la ville de Bordeaux	January 1 2011	January 22 2014	Première instance
3	Le président de la Région Île-de-Franc	oui	Le président de la Région Île-de-Franc	January 1 2002	November 21 2008	Cour d'Appel
4	M. AAA,ancien maire de Saint-Gratien	oui	M. AAA,ancien maire de Saint-Gratien	January 1 1996	November 8 2007	Première instance
5	Un jeune de 18 ans a été condamné p	non	Un particulier	March 1 2015	May 15 2015	Première instance
6	L'ancien maire de Saint-Martin a été cc	oui	L'ancien maire de Saint-Martin/nDes	January 1 2005	October 5 2011	Première instance
7	Le maire d'Ancenis a été condamné pa	oui	Le maire d'Ancenis	June 1 2008	November 15 2012	Première instance
8	Un ancien employé de la mairie de Sai	oui	Un ancien employé de la mairie de Sai	January 1 2011	May 15 2013	Première instance
9	L'ancien responsable administratif de l'	oui	L'ancien responsable administratif de l'	January 1 2011	December 2 2014	Première instance
10	Le maire de Saint-Louis a été condamn	oui	Le maire de Saint-Louis	January 1 2008	March 11 2014	Cour de Cassation
11	L'ancien maire de Parçay-Meslay a été	oui	L'ancien maire de Parçay-Meslay	January 1 2012	June 17 2015	Première instance
12	L'ex-directeur du centre de gestion de l	oui	L'ex-directeur du centre de gestion de l	January 1 2008	September 24 2015	Première instance
13	Le gérant d'une entreprise de transport	non	Le gérant d'une entreprise de transport	January 1 2008	September 8 2015	Première instance
14	La maire de Tarascon a été condamné	oui	L'ancienne maire de Tarascon	January 1 1999	February 7 2001	Cour de Cassation
15	L'ancien maire de Lettret a été condamn	oui	L'ancien maire de Lettret	January 1 2010	October 30 2014	Première instance
16	Le maire de Prunelli-di-Fium'Orbu en l'	oui	Le maire de Prunelli-di-Fium'Orbu et e	October 1 2011	April 5 2016	Première instance
17	Un ancien député des Pyrénées Atlant	oui	Un ancien député des Pyrénées Atlant	January 1 1981	April 20 2005	Première instance
18	L'ancien président du Conseil général	oui	L'ancien président du Conseil général	January 1 1992	March 1 2000	Cour de Cassation
19	M. AAA a été condamné par la Cour d'	oui	M. AAA,maire de Makemo,ancien repre	January 1 2008	April 21 2011	Cour d'Appel
20	L'ancienne présidente de l'Office publi	oui	L'ancienne présidente de l'Office publi	January 1 2008	July 8 2014	Première instance
21	Un ancien notaire de Lamarche a été c	non	Un ancien notaire	January 1 2005	January 5 2016	Première instance

*Note : il est assez probable que les données du dataset sur les affaires de corruption ne soient pas exhaustives, et que beaucoup d'affaires avérées ne soient pas présentes. Les conclusions que nous ferons par la suite - sur des possibles corrélations entre des données différentes - ne devront être considérées comme s'appliquant uniquement aux datasets concernés.*

## 2. Deuxième dataset : données statistiques

<https://public.opendatasoft.com/explore/dataset/resume-statistique-communes-departements-et-regions-france-2012-2013-2014/>

Le deuxième dataset du projet regroupe un certain nombre de statistiques sur la population française en fonction de la commune - chômage, population, etc. Ce dataset comporte énormément de données différentes, et le fait de trier par commune lui donne une taille assez conséquente. (la France ayant environ 36 000 communes)

Libellé commune ou ARM	Région	Département	Population en 2012 ↕	Population en 2007 ↕	Superficie ↕	Naissanc
Castelmoron-d'Albret	72	33	53	56	0.04	6
Cazats	72	33	315	314	7.48	15
Cleyrac	72	33	154	141	6.07	7
Coimères	72	33	904	724	12.91	44
Cubzac-les-Ponts	72	33	2,261	1,927	8.92	122
Cudos	72	33	835	788	34.21	48
Les Églisottes-et-Chalaures	72	33	2,193	2,105	17.16	140
Étauliers	72	33	1,499	1,437	12.98	105
Lamarque	72	33	1,235	1,101	8.91	96
Peujard	72	33	1,906	1,655	10.98	93
Ponducat	72	33	479	372	8.74	18
Saint-Christoly-Médoc	72	33	285	295	7.55	18
Saint-Julien-Beychevelle	72	33	653	719	16.3	51
Saint-Morillon	72	33	1,542	1,394	20.4	93
Saint-Vivien-de-Monségur	72	33	382	378	15.83	18
Le Teich	72	33	6,891	6,284	87.08	361
Vensac	72	33	932	772	34	32
Vertheuil	72	33	1,251	1,161	21.94	59

Ce dataset nous a semblé être intéressant car il pourrait permettre d'effectuer, en combinaison avec le dataset sur la corruption, des requêtes permettant de chercher des corrélations entre certaines données statistiques et la présence de corruption - par exemple, le taux de chômage a-t-il une influence sur la corruption - ou de pouvoir comparer le nombre d'affaires avec la population d'une ville, d'un département, d'une région, etc.

## II. Système de base de donnée : MongoDB

Une fois les datasets sélectionnés, il a fallu choisir le système de base de données pour créer le projet.

Notre choix s'est assez vite orienté sur du NoSQL, spécifiquement sur une base de donnée orientée document, étant donné que nos datasets sont d'une taille assez conséquente et que les données sont facilement organisables par commune ou par département.

MongoDB paru donc le choix évident pour réaliser le projet, étant le seul système de base de données NoSQL orienté document que nous connaissions. De plus, MongoDB étant conçu pour travailler sur des données de type JSON, il était parfaitement adapté pour le projet. Le fait de ne pas avoir à prévoir de schéma en étoile ou en flocon nous a également paru être intéressant.

Une part de curiosité nous a probablement également influencé dans cette décision, aucun de nous n'ayant jamais travaillé en NoSQL dans le passé - contrairement au SQL, avec lequel nous étions tous familiers.

Travailler sur MongoDB a amené une certaine difficulté au projet, dans le sens où nous ne connaissions pas si bien la syntaxe des requêtes - ce qui nous causa donc une grande perte de temps à corriger des requêtes - mais l'étape de l'agrégation a été rendu largement plus simple que si nous avions utilisé du SQL.

# III. Intégration des données

## 1. Formatage des données

### Dataset sur les affaires de corruption

Le dataset sur les affaires de corruption présentait un certain nombre de problèmes de formatage. Tout d'abord, les départements étaient uniquement stockés sous forme de texte, et non sous forme de numéro. Ces noms de départements étaient fréquemment écrits de façon légèrement différente selon l'instance - par exemple, le même département pouvait être noté comme "Charente-Maritime", "Charente Maritime" ou "Charente-maritime".

De plus, un grand nombre de champs avaient été mal remplis, entraînant des cas où un nom de ville avait été entré au lieu du département.

Afin de rendre les données plus facilement utilisables, un script JS a été créé pour résoudre ces problèmes :

```
function find(n)
{
  return dep.findIndex(function(elt){return elt.nom.toLowerCase() == n.toLowerCase()});
}

for(var i in data)
{
  var fields = data[i].fields;
  data[i] = {};

  data[i].description = fields.description;

  var index = find(fields.departement);

  if(index != -1)
  {
    data[i].code_departement = dep[index].code;
  }
  else
  {
    console.log(find(fields.departement));
    data[i].code_departement = fields.departement+" AREMPLACER";
  }

  data[i].departement = fields.departement;
  data[i].tags = fields.tags;
  data[i].region = fields.region;
  data[i].lieu = fields.lieu;

  // MODIFICATIONS "MANUELLES"
  if(data[i].code_departement.includes("AREMPLACER"))
  {
    if(data[i].lieu.includes("uyane") || data[i].lieu.includes("Guyane") || data[i].region == "Cayenne")
    {
      data[i].departement = "Guyane";
      data[i].code_departement = "973";
    }
  }
}
```

Un certain nombre de modifications ont dû être faites "à la main" afin de rendre toutes les données utilisable, car tous les cas d'erreurs d'insertions de données ne pouvaient pas être prévus à l'avance.

Certaines des possessions et dépendances de la France ne rentrent pas dans la catégorie de "département" (Polynésie française, Saint-Martin, Saint-Pierre-et-Miquelon, etc). Ces cas-ci ont été simplement ignorés, étant donné qu'ils n'étaient de toute façon pas présents dans le second dataset.

## Dataset sur la population

Ce dataset contenant un très grand nombre de colonnes, il nous a semblé utile de supprimer celles qui ne nous seraient pas utiles - ou celles qui n'étaient pas remplies pour la plupart des lignes.

## 2. Agrégation des datasets

La première tentative d'agrégation était de lier les informations sur les communes aux départements dans le dataset des affaires de corruption.

Si cette solution donnait un résultat qui semblait correct, toutes nos tentatives de faire des requêtes dessus ont été un échec, sans que nous n'arrivions à en comprendre la raison.

Après avoir passé beaucoup de temps à tenter de faire fonctionner ce modèle, nous avons finalement décidé de partir sur une autre agrégation :

```
1 db.affairesCorruption.aggregate([
2 {
3   $lookup:
4   {
5     from: "communes",
6     localField: "code_departement",
7     foreignField: "departement",
8     as: "bidule"
9   }
10 },
11 {$out: "test1"}
12 ])
```

```
2 db.communes.aggregate(
3 [
4   {
5     $lookup:
6     {
7       from: "affairesCorruption",
8       localField: "commune",
9       foreignField: "lieu",
10      as: "affaires"
11    }
12  },
13  {$out: "affairesCommune"}
14 ])
```

Dans ce nouveau cas, les datasets sont agrégés par communes - chaque affaire est donc associée à la commune dans laquelle elle a eu lieu.



## IV. Requêtes

Une fois les données agrégés, il est possible d'effectuer un grand nombre de requêtes entre les deux datasets.

### Nombre d'affaires de corruption par commune

```
1 db.affairesCommune.aggregate([
2   {
3     $project: {
4       commune: 1,
5       count: {$size: "$affaires"}
6     }
7   },
8   {
9     $match: {
10      count: {$gt: 0}
11    }
12  }
13 ])
```

_id	commune	affaires
5a0add9bf15f...	Peujard	1
5a0add9bf15f...	Moltifao	1
5a0add9bf15f...	Saint-Pierre	4
5a0add9bf15f...	Nice	4
5a0add9bf15f...	Bagneux	1
5a0add9bf15f...	Grenoble	5
5a0add9bf15f...	Saint-Hilaire-du...	1
5a0add9bf15f...	Courvaudon	1
5a0add9bf15f...	Troyes	1
5a0add9bf15f...	Toulouse	2
5a0add9bf15f...	Mende	1
5a0add9bf15f...	Narbonne	1
5a0add9bf15f...	Cuges-les-Pins	1

### Nombre d'affaires de corruption par département

```
1 db.affairesCommune.aggregate([
2   {
3     $project: {
4       commune: 1,
5       departement: 1,
6       nb_affaires: {$size: "$affaires"}
7     }
8   },
9   {
10    $group: {
11      _id: "$departement",
12      count: {$sum: "$nb_affaires"}
13    }
14  }
15 ])
```

_id	count
56	8
59	22
58	0
24	9
973	5
57	9
974	22
972	4
23	0
93	8
95	4

(Version améliorée de la requête précédente)

```
1 db.affairesCommune.aggregate([
2   {
3     $group: {
4       _id: "$departement", nb_affaires: {$sum: {$size: "$affaires"}}
5     }
6   }
7 ])
```

## Département triés par nombre d'affaire par 100000 habitants

```
1 db.affairesCommune.aggregate([
2   {
3     $group: {
4       _id: "$departement",
5       nb_affaires: {$sum: {$size: "$affaires"}},
6       pop_departement: {$sum: "$population"}
7     }
8   },
9   {
10    $project: {
11      affaires_habitant: {$multiply: [{$divide: ["$nb_affaires", "$pop_departement"]},100000]}
12    }
13  },
14  {
15    $sort: {
16      affaires_habitant: -1
17    }
18  }
19 ])
```



















_id	affaires_habitant
" 2B	8.195377806916...
" 32	4.748588613939...
" 04	4.341327577973...
" 09	3.937886405103...
" 05	3.582842483913...
" 65	3.495678467494...
" 66	3.495029412856...
" 39	3.065932886729...
" 19	2.901590486099...
" 46	2.867860461381...
" 2A	2.750483053586...
" 15	2.713428077197...
" 974	2.638066824630...
" 61	2.413668258538...



## Villes les plus corrompues (nombre d'affaires par habitant)

```

1 db.affairesCommune.aggregate([
2   {
3     $match: {
4       population: {
5         $gt: 0
6       }
7     }
8   },
9   {
10    $project: {
11      _id: "$commune",
12      population: 1,
13      nb_affaires: {$size: "$affaires"},
14      affaires_par_habitant: {$divide: [{$size: "$affaires"}, "$population"]}
15    }
16  },
17  {
18    $sort: {
19      affaires_par_habitant: -1
20    }
21  }
22 ])
```

_id	population	nb_affaires	affaires_par_habitant
 Saint-Martin	58	5	0.086206896551...
 Saint-Martin	69	5	0.072463768115...
 Saint-Pierre	102	4	0.039215686274...
 La Rochelle	37	1	0.027027027027...
 Saint-Pierre	152	4	0.026315789473...
 Chaumont	47	1	0.021276595744...
 Saint-André	99	2	0.020202020202...
 Saint-Denis	265	5	0.018867924528...
 Saint-André	107	2	0.018691588785...
 Gélaucourt	57	1	0.017543859649...
 La Rochette	58	1	0.017241379310...
 Pommiers	61	1	0.016393442622...
 Larche	62	1	0.016129032258...
 Mélagues	62	1	0.016129032258...
 La Rochette	62	1	0.016129032258...
 Saint-Pierre	262	4	0.015267175572...
 Saint-Paul	133	2	0.015037593984...
 Contrazy	68	1	0.014705882352...

Création d'une vue regroupant les données des deux datasets

```
1 db.createView("affairesCommuneV", "communes",
2 [
3   {
4     $lookup:
5     {
6       from: "affairesCorruption",
7       localField: "commune",
8       foreignField: "lieu",
9       as: "affaires"
10    }
11  }
12 ])
```