

# Projet Big Data & Visualisation des données

par :

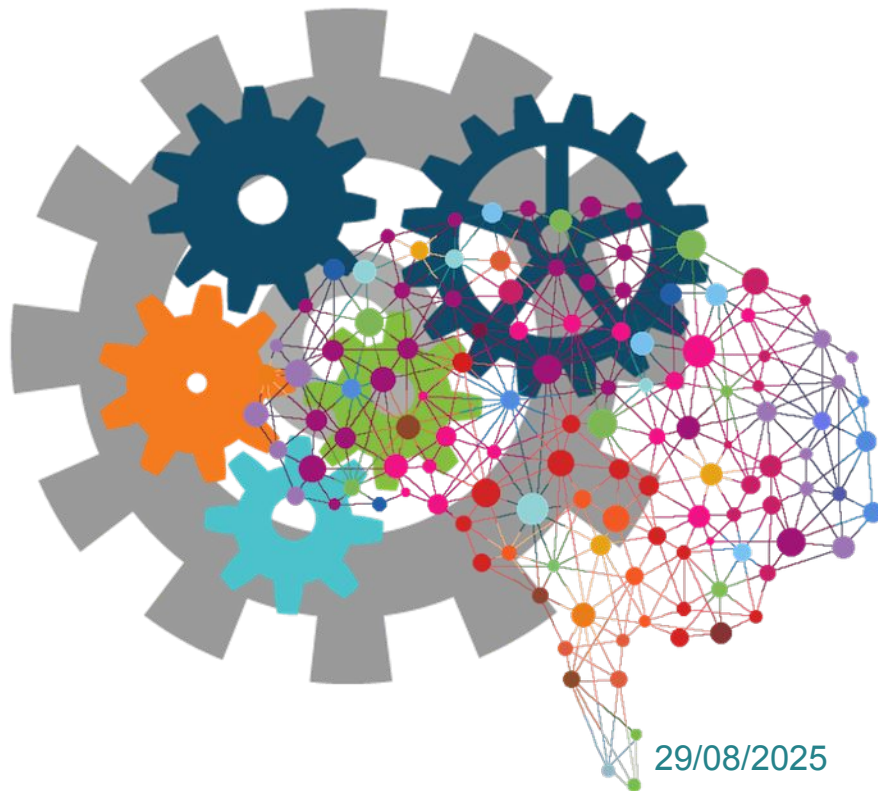
Bendib Hayel  
Dumont François  
El Qotbi Sofyan  
Ouedraogo Célia



# Étude du programme de fidélité Digicheese

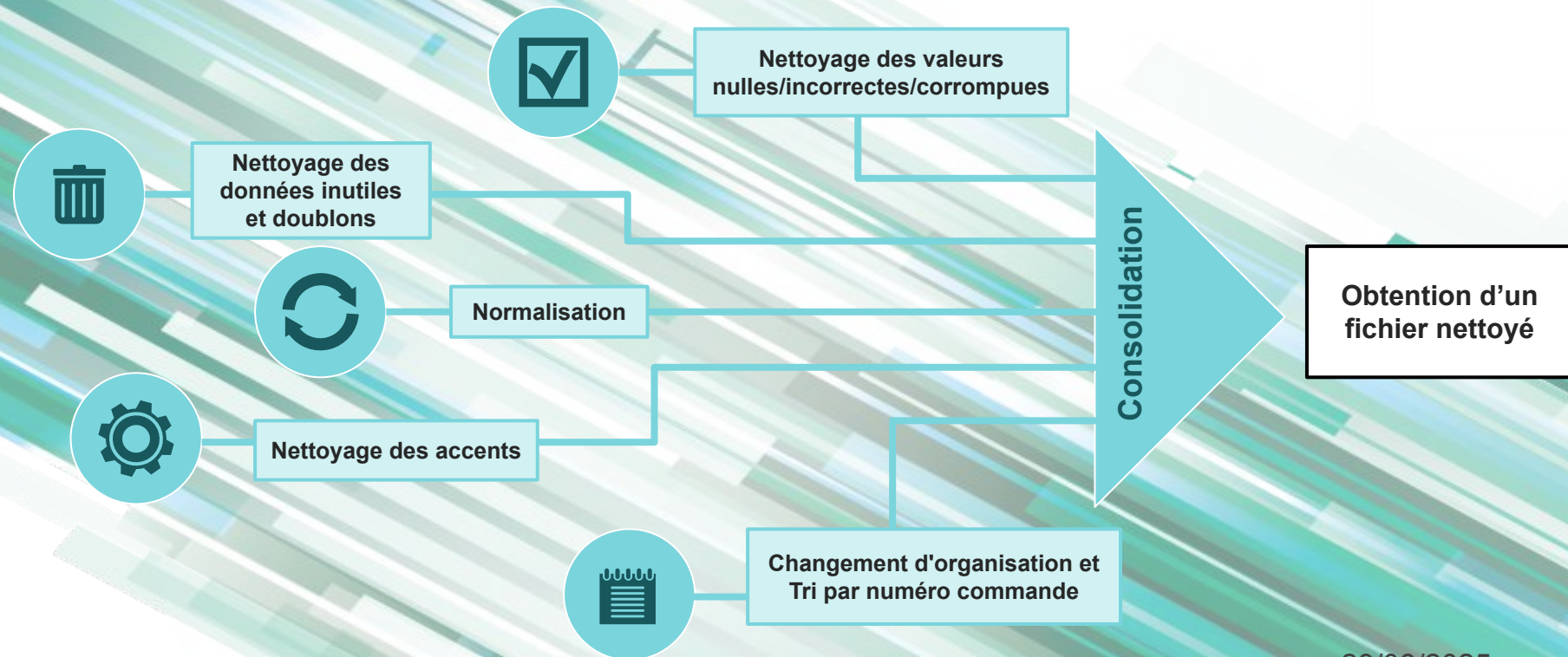
## Objectifs de la présentation :

- Répondre aux demandes spécifiques exprimées dans les lots 1, 2, 3 et 4.
- Valoriser les données de l'entreprise par des analyses complémentaires.
- Offrir un outil décisionnel complet pour:
  - Mieux comprendre les comportements des clients
  - Suivre l'évolution géographique et temporelle
  - Piloter le programme de fidélité



# Nettoyage des données

Modification des données avec Python



# Extract - Transform - Load (ETL)

Script python - pandas

Traitement :

- gestion des valeurs nulles

```
df = df.fillna(0)
```

- suppression des dates invalides

```
df = df[(df['datcde'] >= '1990-01-01') & df['datcde'].notna()]
```

- création des numéros de départements :

```
df['cpcli'] = df['cpcli'].astype(str)
```

```
df['DEP'] = df['cpcli'].str[0:2]
```

- texte :

```
nettoyer texte = ['nomcli', 'prenomcli', 'villecli']
```

```
df[nettoyer texte] = df[nettoyer texte].map(nettoyer caracteres)
```

```
df['nomcli'] = df['nomcli'].str.title()
```

# ETL

## Script python - pandas

### Traitement (suite) :

- gestion des types

```
df['codcde'] = df['codcde'].astype(str)
```

- sélection des colonnes :

```
df=df[['codcde','datcde','codobj', 'libobj','qte', 'timbrecli',  
      'timbrecede','codcli','nomcli','prenomcli','cpcli','villecli','DEP']]
```

- suppression des doublons

```
df = df.drop_duplicates()
```

- suppression des publicités/produits indésirables  
(points bonus/flyers, publicités,...)



# ETL

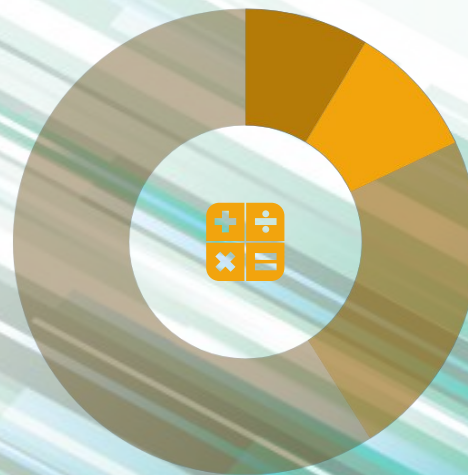
Script python - pandas

**Obtention de 2 fichiers :**



**Fichier 1**

Jeu de données  
épuré/réorganisé sur  
lequel on peut  
travailler sereinement



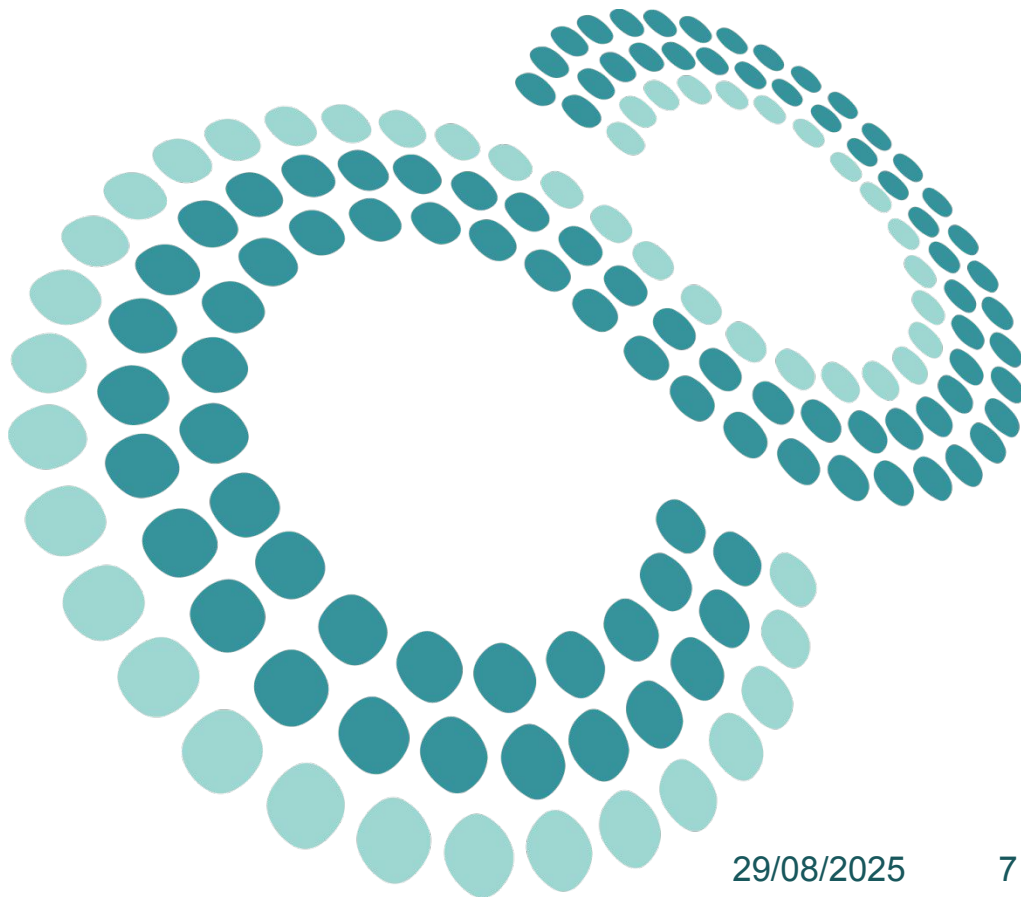
**Fichier 2**

Données  
indésirables qui  
faussent l'analyse

## Outils utilisés

Sur Machine Virtuelle :

- **Hadoop 2.7.2** : Framework pour stocker et gérer les données
  - HDFS
  - YARN
  - MapReduce
- **Docker** : Héberge des conteneurs
- **Python 3.5.2** : Scripts
- **HBase 1.4.9** : Base de données
  - HappyBase
  - Thrift



## Outils utilisés

### Sur Machine Virtuelle :

- **Hadoop 2.7.2**
  - HDFS
  - YARN
  - MapReduce
- **Docker**
- **Python 3.5.2**
- **HBase 1.4.9**
  - HappyBase
  - Thrift

### Sur Windows :

- **Putty** (SSH) : connexion à la VM
- **FileZilla** : transferts de fichier
- **ODBC** (Connecteur) : connecteur entre Hbase/Hadoop et outils BI
- **PowerBI** : Solution d'analyses de données et de visualisation



## Les 4 étapes de traitement

01

### Mapper-Reducer

Limiter le flux d'information  
Isoler les données recherchées et les grouper

02

### Base NoSQL HBASE

Stockage du contenu du fichier CSV  
Interroger la base de données *via* scripts

03

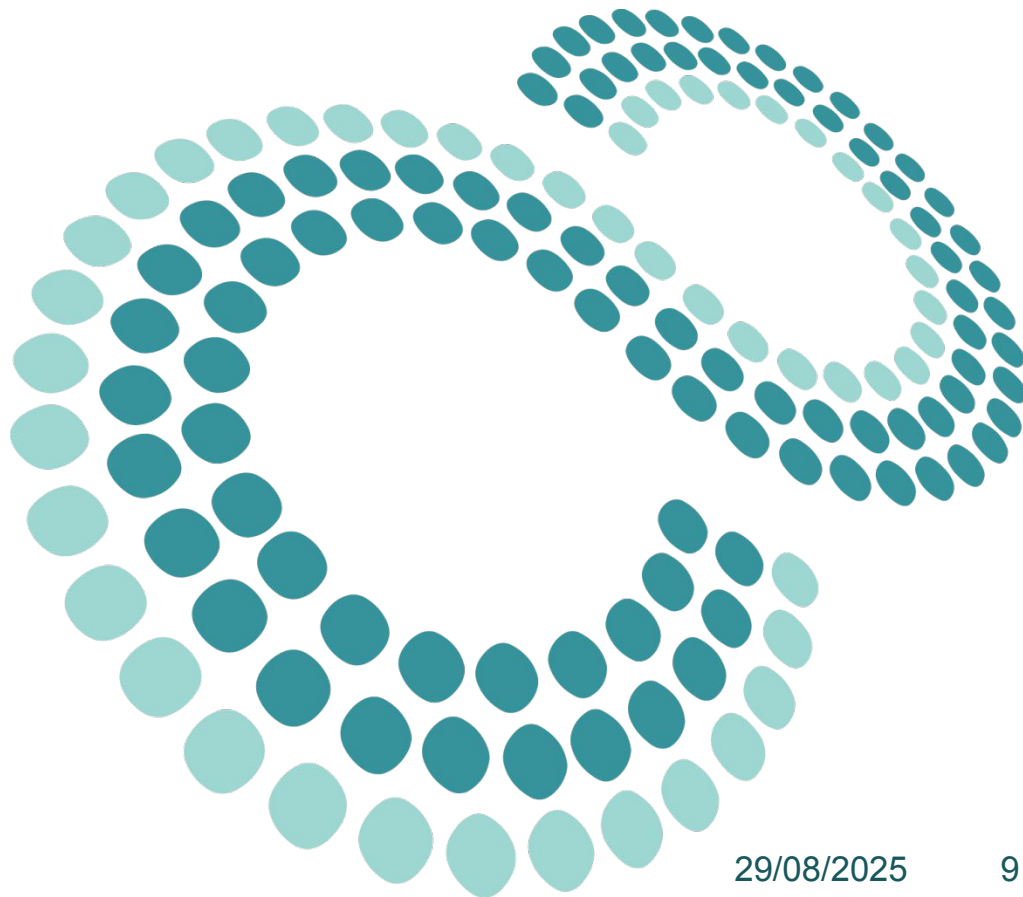
### Connexion par ODBC

Connexion en local via le connecteur ODBC à HBase Thrift  
Import des données sur PowerBI via le connecteur ODBC

04

### PowerBI

Récupération des données de HBASE  
Mise en place de Dashboards



# Analyse des données demandées : Lot 1



## Lot 1 : 100 meilleures commandes

### Script python

Mapper :

- années : 2006 à 2010

```
2006 <= datcde <= 2010
```

- départements : 53, 61 et 28

```
departement in [28, 53, 61]
```



codcde, qte, ville, timbrecde

## Lot 1 : 100 meilleures commandes

### Script python

Reducer :

- Récupérer les quantités totales

```
qte_tot += qte
```

```
result.append([current cde, ville, qte_tot, timbre_cde])
```

- filtrer les meilleures commandes [ Qté, timbre\_cde]

```
sorted_result = sorted(result,
```

```
key=lambda ele : (ele[2], ele[3]), reverse=True)
```

- Montrer villes et quantités du top 100


```
sorted_result = sorted_result[:100]
```



codcde, qte\_total, ville, timbre\_cde

## Lot 1 : 100 meilleures commandes

Script shell



- suppression du répertoire de sortie
- exécution des commandes hadoop en streaming
- création d'un tsv



# Analyse des données demandées : Lot 2



## Lot 2 : 5% aléatoires des 100 meilleures commandes

Mapper :

- années : 2011 à 2016

```
20011 <= datcde <= 2016
```

- départements : 22, 49 et 53

```
departement in [22, 49, 53]
```

- sans timbrecli

```
(timbrecli == 0.0)
```



codcde, qte, timbrecde, ville

## Lot 2 : 5% aléatoires des 100 meilleures commandes

Reducer :

- Récupérer les quantités totales et timbre cde  
Idem Lot1, sert à définir le top100

- Calcul quantité sans timbres

```
qte sans timbre += qte
```

- Calcul des moyennes de quantité

```
n lines += 1
```

```
q_avg = round(qte tot / n lines, 2)
```



codcde, ville, qte\_sans\_timbre, q\_avg

## Lot 2 : 5% aléatoires des 100 meilleures commandes

Script : Sample5.py

- retourne 5% aléatoirement du TOP 100

```
file = '/root/DW/Lot_02/part-00000'  
echantillon = random.sample(lines, 5)
```



fichier csv

```
file out = 'DW/Lot 02/sample5.csv'
```

## Lot 2 : 5% aléatoires des 100 meilleures commandes

Graph.py :

- ouvre le csv sample5

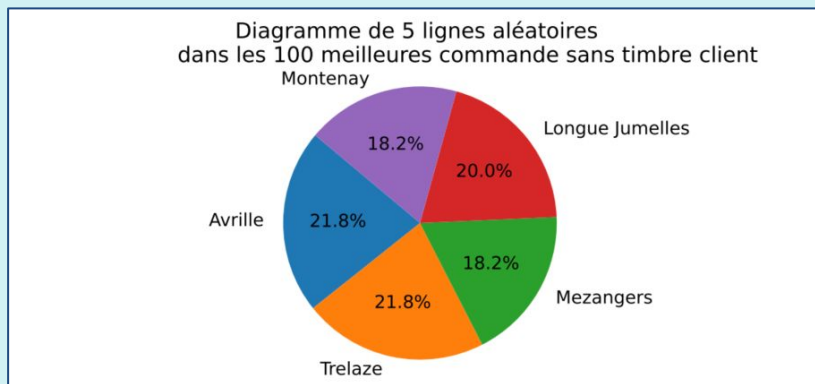
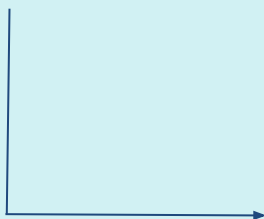
```
df = pd.read_csv('out/sample5.csv', header=None).reset_index(drop=True)
```

- crée un diagramme en camembert

```
plt.pie(sizes = df['qte_tot'], labels=df['ville'], ...
```

- enregistre le diagramme

```
plt.savefig("out/graph.pdf", bbox_inches="tight")
```





## Lot 2 : 100 meilleures commandes

Script shell



- suppression du répertoire de sortie
- exécution des commandes hadoop en streaming
- récupération du fichier dans hadoop
- exécutions des scripts python :  
sample5.py et graph.py

# Analyse des données demandées : Lot 3



## Lot 3 : Informations sur les commandes

### Objectifs

- Charger le CSV **DW.csv** dans une base **NoSQL HBase**.
- Interroger HBase pour produire 3 livrables :
  1. **Meilleure commande de Nantes en 2020**
  2. **Nombre de commandes par année (2010–2015)**
  3. **Client ayant le plus de frais *timbre*de**, avec nb de commandes et somme des quantités
- Générer les sorties au format **CSV**, **Excel** et **PDF (barplot)**.

## Lot 3 : Informations sur les commandes

# Architecture & Modèle HBase

- Connexion via **Thrift (port 9090)** avec **HappyBase**.
- Table : **fromagerie\_dw**
  - Families : commande, client, localisation
  - **RowKey** : code#000001 (garantit l'unicité et l'ordre d'insertion)

```
LOT3_livrables > $ jobLOT3.sh
1 start-hbase.sh
2
3 # Démarrer le service Thrift (port 9090)
4 hbase thrift start -p 9090
5
6 # Vérifier que ça répond
7 echo "status 'simple'" | hbase shell
8
```

```
LOT3_livrables > hb_load_simple.py > ...
1 import os, io, csv, sys
2 try:
3     import happybase # type: ignore
4 except ImportError:
5     sys.stderr.write("HappyBase manquant. Faites: pip3 install happybase thrift\n"); sys.exit(1)
6
7 # Connexion Thrift (VM locale)
8 conn = happybase.Connection(['localhost', 9090, autoconnect=False])
9 conn.open()
```

## Lot 3 : Informations sur les commandes

### Pipeline

→ **Chargement** : `hb_load_simple.py`  
Crée la table si besoin et charge tout le CSV

```
11 TABLE = 'fromagerie_dw'
12 # Créer la table si besoin
13 existing = [t.decode('utf-8') if isinstance(t, bytes) else t for t in conn.tables()]
14 if TABLE not in existing:
15     conn.create_table(TABLE, {
16         'commande': dict(),
17         'client': dict(),
18         'localisation': dict(),
19     })
20
21 table = conn.table(TABLE)
```



## Lot 3 : Informations sur les commandes

→ Analyses & exports : `hb_queries_export.py`

Q1 : La meilleure commande de Nantes de l'année 2020

Q2 : Le nombre total de commandes effectuées entre 2010 et 2015, réparties par année

Q3 : Le nom, le prénom, le nombre de commande et la somme des quantités d'objets du client qui a eu le plus de frais de timbre

```
5 tar -czf /root/LOT3_livrables.tgz \  
6 /root/LOT3_q1_best_order_nantes_2020.csv \  
7 /root/LOT3_q2_counts_2010_2015.csv \  
8 /root/LOT3_q2_counts_bar.pdf \  
9 /root/LOT3_q3_top_client_timbrede.xlsx  
10  
11 echo "OK -> /root/LOT3_livrables.tgz"
```

## Lot 3 : Informations sur les commandes

### Q1

Python:

```
86 # ===== Q1 : meilleure commande Nantes 2020 =====
87 q1 = orders[(orders["year"] == 2020) & (orders["ville"].str.upper() == "NANTES")]
88 q1 = q1.sort_values(["sum_qte", "timbrecde_ord"], ascending=[False, False]).head(1)
89 q1_out = "/root/LOT3_q1_best_order_nantes_2020.csv"
90 q1_to_write = q1[["codcde", "ville", "sum_qte", "timbrecde_ord"]].rename(columns={"timbrecde_ord": "timbrecde"})
91 if q1_to_write.empty:
92     pd.DataFrame(columns=["codcde", "ville", "sum_qte", "timbrecde"]).to_csv(q1_out, index=False)
93 else:
94     q1_to_write.to_csv(q1_out, index=False)
```

Livrable :

	A
1	codcde,ville,sum_qte,timbrecde
2	85301,Nantes,6,3.05

## Lot 3 : Informations sur les commandes

Q2

Python:

```
96 # ===== Q2 : nb de commandes 2010-2015 =====
97 q2 = orders[(orders["year"] >= 2010) & (orders["year"] <= 2015)].copy()
98 q2_counts = q2.groupby("year").size().to_frame("count_cmds").reset_index()
99 q2_csv = "/root/LOT3_q2_counts_2010_2015.csv"
100 q2_counts.sort_values("year").to_csv(q2_csv, index=False)
```

Livrable :

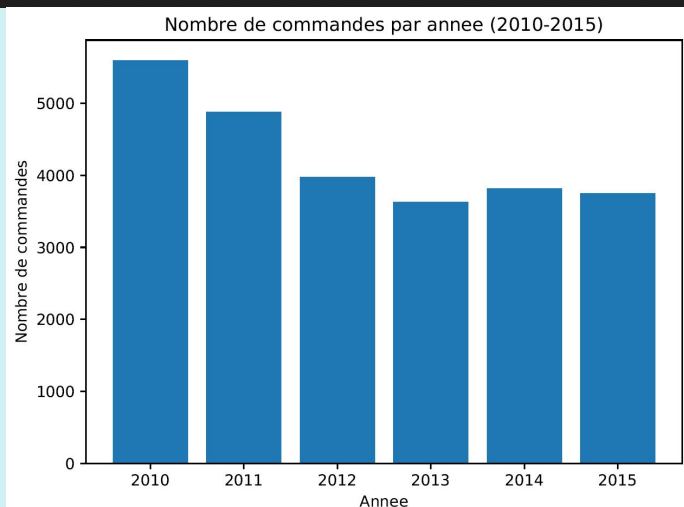
	A
1	year,count_cmds
2	2010,5598
3	2011,4882
4	2012,3978
5	2013,3629
6	2014,3815
7	2015,3749

## Lot 3 : Informations sur les commandes

Q2

Python:

```
try:
    if plt is not None:
        fig = plt.figure()
        xs = [str(int(y)) for y in q2_counts["year"].tolist()]
        ys = [int(c) for c in q2_counts["count_cmds"].tolist()]
        plt.bar(xs, ys)
        plt.title("Nombre de commandes par annee (2010-2015)")
        plt.xlabel("Annee"); plt.ylabel("Nombre de commandes")
        plt.tight_layout()
        plt.savefig("/root/LOT3_q2_counts_bar.pdf", bbox_inches="tight")
        plt.close(fig)
except Exception as e:
    sys.stderr.write("Impossible d'écrire le PDF Q2 : %s\n" % e)
```



Livrable :

## Lot 3 : Informations sur les commandes

Q3

Python:

```
117 # ===== Q3 : client avec le plus de frais timbre =====
118 clients = orders.groupby(["codcli", "nom", "prenom"], as_index=False).agg({
119     "codcde": "count",
120     "sum_qte": "sum",
121     "timbre_cde_ord": "sum"
122 }).rename(columns={"codcde": "nb_cmds", "timbre_cde_ord": "sum_timbre"})
123 clients = clients.sort_values("sum_timbre", ascending=False)
```

```
125 # Excel (xlsxwriter requis). Sinon, CSV fallback.
126 q3_xlsx = "/root/LOT3_q3_top_client_timbre.xlsx"
127 wrote_excel = False
128 try:
129     import xlsxwriter # type: ignore # noqa
130     with pd.ExcelWriter(q3_xlsx, engine="xlsxwriter") as w:
131         clients.to_excel(w, index=False, sheet_name="clients_sorted")
132         if not clients.empty:
133             clients.head(1).to_excel(w, index=False, sheet_name="winner")
134         wrote_excel = True
135 except Exception as e:
136     sys.stderr.write("Excel indisponible (xlsxwriter?) : %s\n" % e)
137
138 clients.to_csv("/root/LOT3_q3_top_client_timbre.csv", index=False)
```



## Lot 3 : Informations sur les commandes

Livrable :

	A	B	C	D	E	F
1	codcli	nom	prenom	sum_timbr	nb_cmds	sum_qte
2	786	Crotte	Nicole	303,4	57	142
3	27059	Le Penven	Michele	266,65	64	135
4	8993	Briard	Antoinette	247,85	55	112
5	16848	Auvray	Georgette	216,15	40	70
6	19658	Le Coutey	Francoise	212,7	32	119
7	4812	Horel	Christian	209,5	42	89
8	5576	Pichon	Alain	208,6	46	100
9	4153	Ruel	Roger	205,35	39	73
10	3190	Trottet	Denise	201,7	35	164
11	21884	Lendormy	Marie-Ang	187,9	32	73
12	1903	Coisel	Jacky	183,37	66	117
13	6666	Lejeune	Josiane	181,25	46	94
14	1893	Dodier	Anick	176,3	34	67
15	3462	Gesquin	Roger	173,65	50	77
16	9345	Granier	Ginette	171,35	30	66
17	3420	Lemiere	Denise	170,35	31	84
18	10268	Brochard	Madeleine	165,6	23	123
19	8163	Baillard	Germaine	159,4	25	105
20	42	Dufour	Paulette	157,45	34	72

# Analyse des données demandées : Lot 4



## Lot 4 : Visualisation des données dans Power BI

- Résultats des lots 1 et 2
- Questions précises sur la BDD : Lot 3
- Exploration des données :
  - Carte interactive
  - Informations générales sur les commandes
  - Etude temporelle des commandes

# Connexion des données



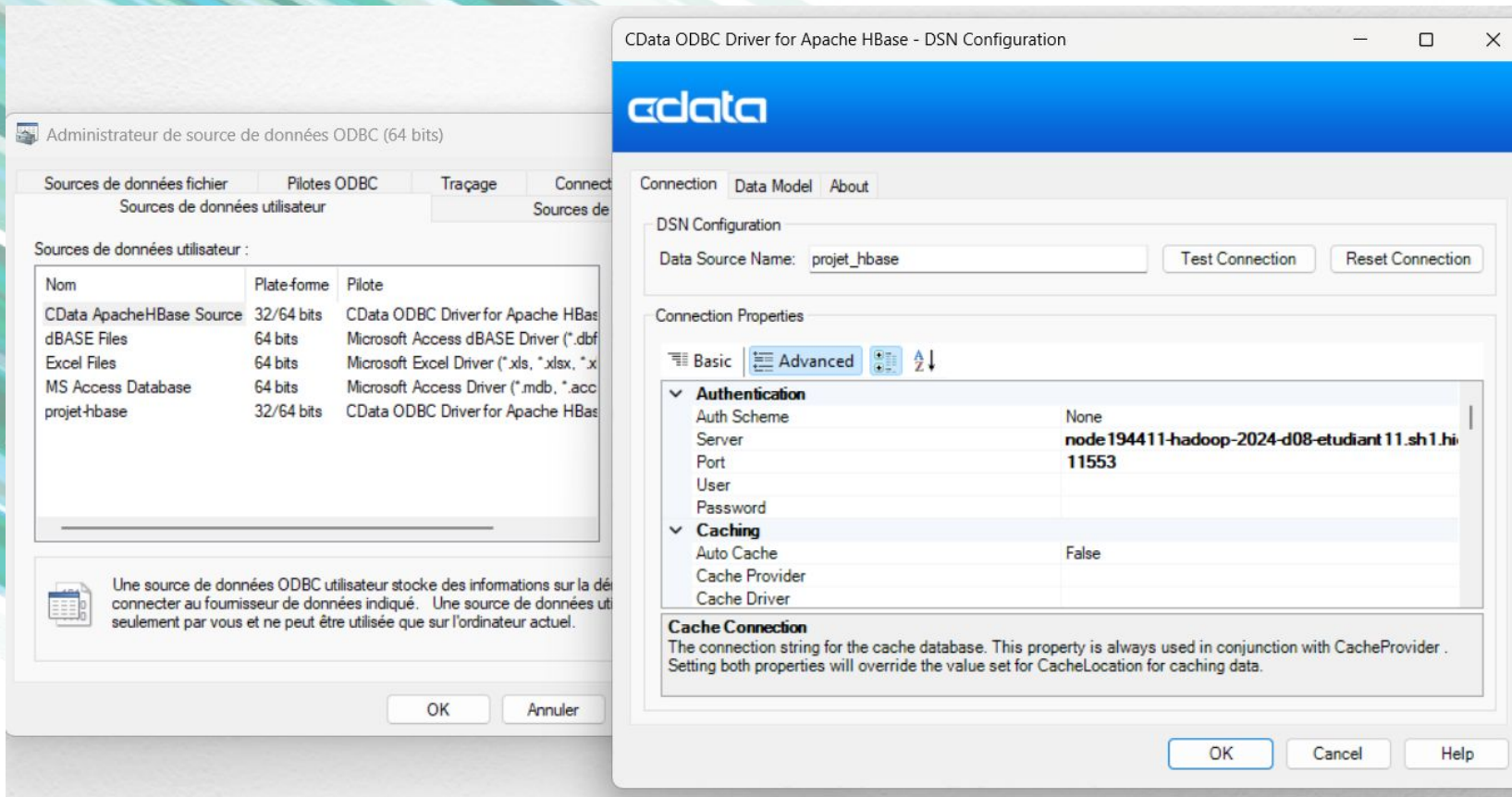
Arrêt/redémarrage automatique des services nécessaires :

- **HDFS** : stockage distribué
- **YARN** : gestionnaire de ressources
- **ZooKeeper** : coordination du cluster
- **HBase Master & RegionServer** : gestion et accès aux tables HBase

**Objectif :**

- Avoir un cluster HBase opérationnel et accessible depuis l'extérieur
- Préparer la connexion via ODBC pour que Power BI interroge les données stockées dans HBase.

## Configuration de l'ODBC côté Windows



The image shows two overlapping Windows dialog boxes. The background dialog is the 'Administrateur de source de données ODBC (64 bits)' (ODBC Data Source Administrator). It has tabs for 'Sources de données fichier', 'Pilotes ODBC', 'Traçage', and 'Connect'. The 'Pilotes ODBC' tab is active, showing a list of installed drivers under 'Sources de données utilisateur'. The foreground dialog is the 'CData ODBC Driver for Apache HBase - DSN Configuration' window. It has tabs for 'Connection', 'Data Model', and 'About'. The 'Connection' tab is active, showing the 'DSN Configuration' section with 'Data Source Name' set to 'projet\_hbase' and buttons for 'Test Connection' and 'Reset Connection'. Below this is the 'Connection Properties' section with 'Basic' and 'Advanced' tabs. The 'Basic' tab is active, showing 'Authentication' and 'Caching' sections. The 'Authentication' section has 'Auth Scheme' set to 'None', 'Server' set to 'node194411-hadoop-2024-d08-etudiant11.sh1.hi', 'Port' set to '11553', and 'User' and 'Password' fields are empty. The 'Caching' section has 'Auto Cache' set to 'False', and 'Cache Provider' and 'Cache Driver' fields are empty. At the bottom, there is a 'Cache Connection' section with a text box containing the connection string for the cache database.

**Administrateur de source de données ODBC (64 bits)**

Sources de données fichier | Pilotes ODBC | Traçage | Connect

Sources de données utilisateur

Sources de données utilisateur :

Nom	Plate-forme	Pilote
CData ApacheHBase Source	32/64 bits	CData ODBC Driver for Apache HBase
dBASE Files	64 bits	Microsoft Access dBASE Driver (*.dbf)
Excel Files	64 bits	Microsoft Excel Driver (*.xls, *.xlsx, *.xlsm)
MS Access Database	64 bits	Microsoft Access Driver (*.mdb, *.accdb)
projet_hbase	32/64 bits	CData ODBC Driver for Apache HBase

Une source de données ODBC utilisateur stocke des informations sur la façon de se connecter au fournisseur de données indiqué. Une source de données utilisateur n'est utilisée que sur l'ordinateur actuel.

OK Annuler

**CData ODBC Driver for Apache HBase - DSN Configuration**

Connection | Data Model | About

DSN Configuration

Data Source Name: projet\_hbase Test Connection Reset Connection

Connection Properties

Basic Advanced

**Authentication**

Auth Scheme	None
Server	node194411-hadoop-2024-d08-etudiant11.sh1.hi
Port	11553
User	
Password	

**Caching**

Auto Cache	False
Cache Provider	
Cache Driver	

**Cache Connection**

The connection string for the cache database. This property is always used in conjunction with CacheProvider. Setting both properties will override the value set for CacheLocation for caching data.

OK Cancel Help



## Connexion PowerBI

Importation des données à  
partir de ODBC, via la  
source de données créée :  
**projet-hbase**

**Navigateur**

Options d'affichage ▾

- ODBC (dsn=projet-hbase) [1]
  - CData [1]
    - ApacheHBase [5]
      - ☒ bibliotheque
      - ☒ fromagerie\_dw
      - ☐ maTable
      - ☐ music\_trends
      - ☐ store

**fromagerie\_dw**

RowKey	client:codcli	client:nom	client:prenom	commande
10655#000027	5042	Hetzel	Christiane	
11155#000028	10419	Fernandes	Jose	
11155#000029	10419	Fernandes	Jose	
11815#000030	10964	Gaillard	Laurent	
12661#000031	6807	Thuault	Sylvain Et Evelyne	
1386#000006	1330	Robert	Yvonne	
13996#000032	4758	Lemonnier	Aurelie	
14090#000033	908	Bonhomme	Francoise	
14096#000034	12708	Melisson	Isabelle	
14132#000035	189	Herbert	Juliette	
14235#000036	12084	Leroy	Emilie	
14335#000037	3174	Gaulier	Rene	
14351#000038	6304	Osouf	Chantal	
14352#000039	10261	Richard	Marc	
14489#000040	2304	Maignan	Therese	
14536#000041	5181	Bourreau	Annie	
14544#000042	186	Bobel	Louise	
14544#000043	186	Bobel	Louise	
14548#000044	42	Dufour	Paulette	
14560#000045	417	Louvet	Nicole	
14649#000046	2763	Martin	Irene	
14667#000047	2807	Letondeur	Micheline	
14667#000048	2807	Letondeur	Micheline	

Sélectionner les tables associées

Charger Transformer les données Annuler

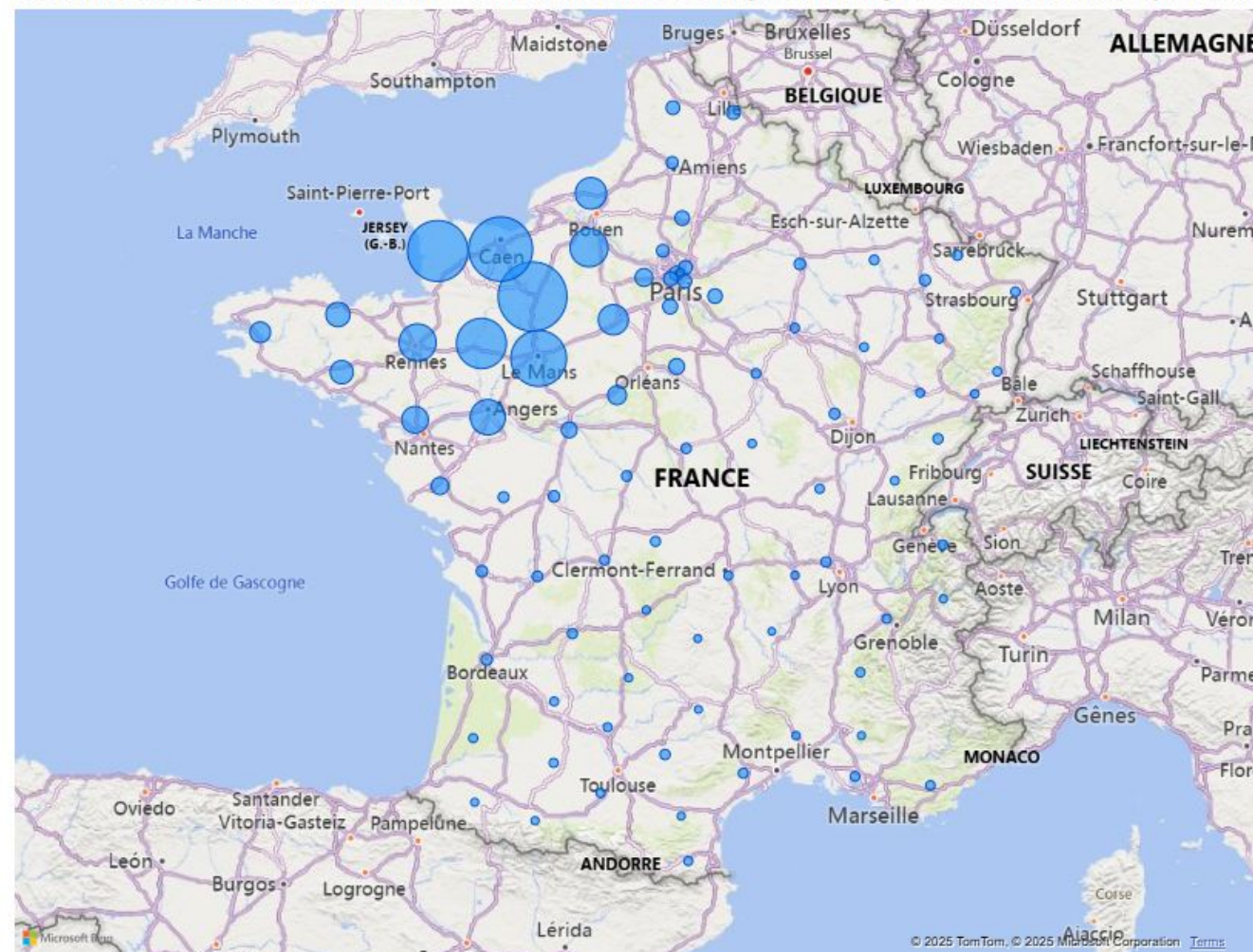
# Carte interactive





## Quantité d'objets commandés dans chaque département...

Le numéro de département, la ville, le nombre de commandes et la quantité d'objets commandés sont disponible...



Année

- ☐ 2004
- ☐ 2005
- ☐ 2006
- ☐ 2007
- ☐ 2008
- ☐ 2009
- ☐ 2010
- ☐ 2011
- ☐ 2012
- ☐ 2013
- ☐ 2014

Départem... ▼

- ☐ 10
- ☐ 11
- ☐ 12
- ☐ 13
- ☐ 14
- ☐ 15
- ☐ 16
- ☐ 17
- ☐ 18
- ☐ 19
- ☐ 20

## Lot 1 :

\* Départements 28, 53 et 61

\* De 2006 à 2010

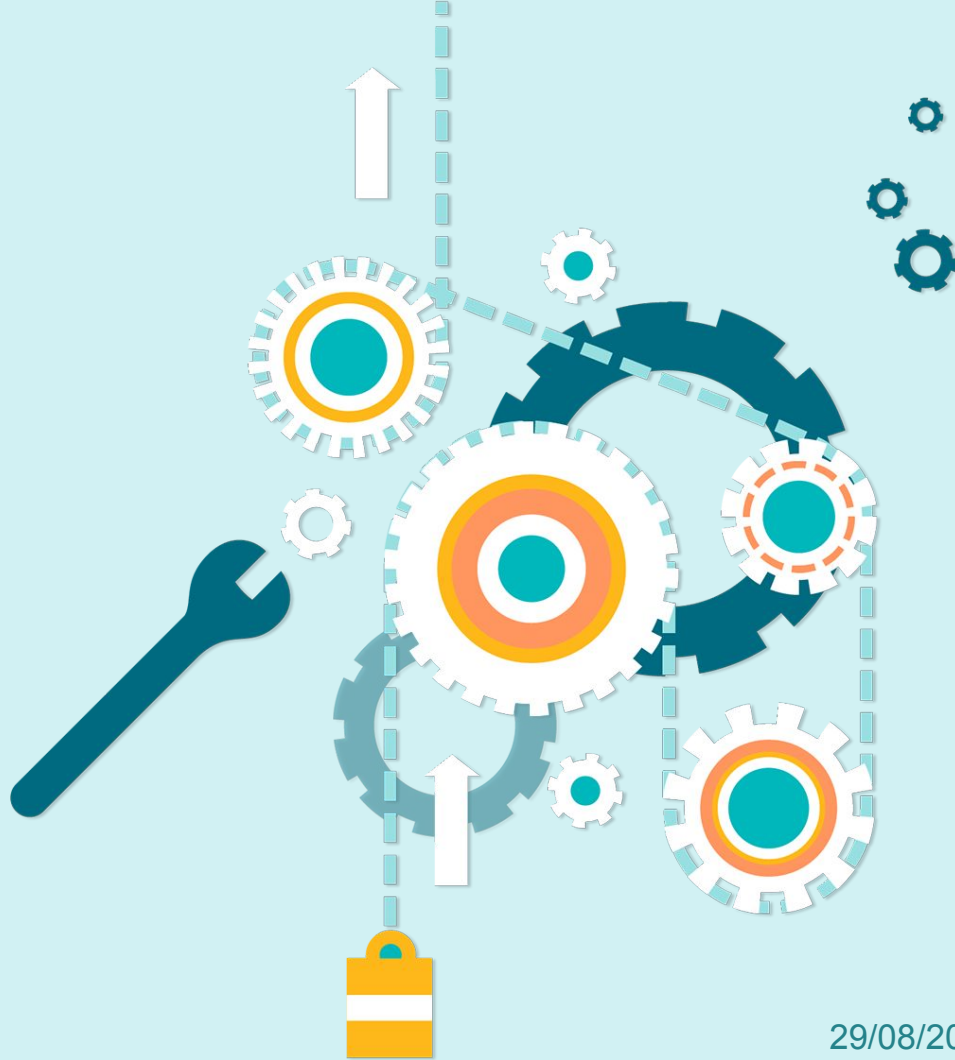
## Lot 2 :

\* Départements 22, 49 et 53

\* De 2011 à 2016

# PowerBI

## Lots 1, 2 et 3



Année ▼ Département

- ☐ 2011  
☐ 2012  
☐ 2013  
☐ 2014  
☐ 2015  
☐ 2016

- ☐ 22  
☐ 49  
☐ 53

Ville

- Chemaze  
● Beaufort En Anjou  
● Le Bignon Du Maine  
● Durtal  
● Le Ribay

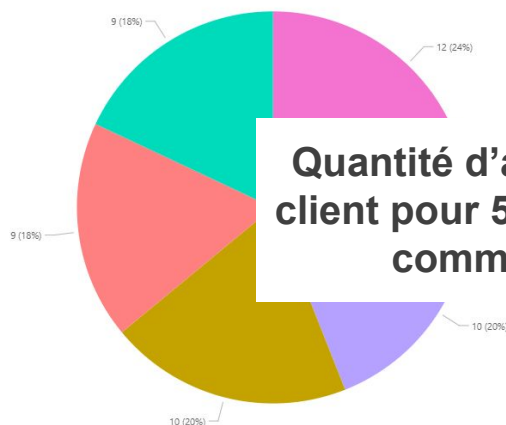
## 100 meilleures commandes entre 2006 et 2010, pour les départements 28, 53 et 61

Code commande	Année	Département	Ville	Quantité d'objets commandés	Quantité de timbres commande
25862	2007	28	Arrou	42	9,85
42997				3	6,40
25861				0	7,85
24701				5	5,10
26620				0	5,80
16235				8	6,40
22609				8	6,40
36865				8	5,80
20375				7	5,80
16234	2006	28	Arrou	16	6,40
16236	2006	28	Arrou		
22146	2006	61	Vimoutiers		
42676	2009	61	La Gonfrière		
45087	2010	61	Domfront En Poirais		
47308	2010	61	La Gonfrière		
46141	2010	61	Cerisy Belle Etoile		
15348	2006	61	Sainte Opportune		
41571	2009	61	Goulet		
42331	2009	61	Athis Val De Rouvre		
15369	2006	61	Sai		
17351	2006	61	Saint Maurice Du Desert		
44514	2010	61	Ri		
48735	2010	61	La Ferte Mace		
34461	2008	61	Ronfeugerai		

## 100 meilleures commandes entre 2011 et 2016, pour les départements 22, 49 et 53

Code commande	Année	Département	Ville	Quantité d'articles sans timbre client	Quantité moyenne d'objets commandés
57069	2012	49	Beaufort En Vallee	300	300,00
70366					5,75
61965					3,00
63126					2,22
73061					3,33
63611					2,38
72129					3,80
51717					5,67
73656					3,40
65754					5,00
66304					5,00
69796					
2015	53	Juvigne		15	2,50
2016	53	Le Buret		15	5,00
2015	49	Longue Jumelles		14	3,50
2012	49	Sainte Gemmes Sur Loire		13	3,25
2014	53	Saint Martin Du Limet		13	4,33
2014	53	Montenay		13	2,60
2011	49	La Menitre		12	4,00
				12	12,00
				12	4,00
				12	2,40
				12	4,00
				12	4,00
				12	12,00
				12	4,00
				12	2,40
				12	2,40
				12	2,40
2015	49	Trelaze		12	2,40
2015	53	Gorron		12	2,00
				1426	3,35

## Quantité d'articles sans timbres client pour 5% des 100 meilleures commandes, par ville



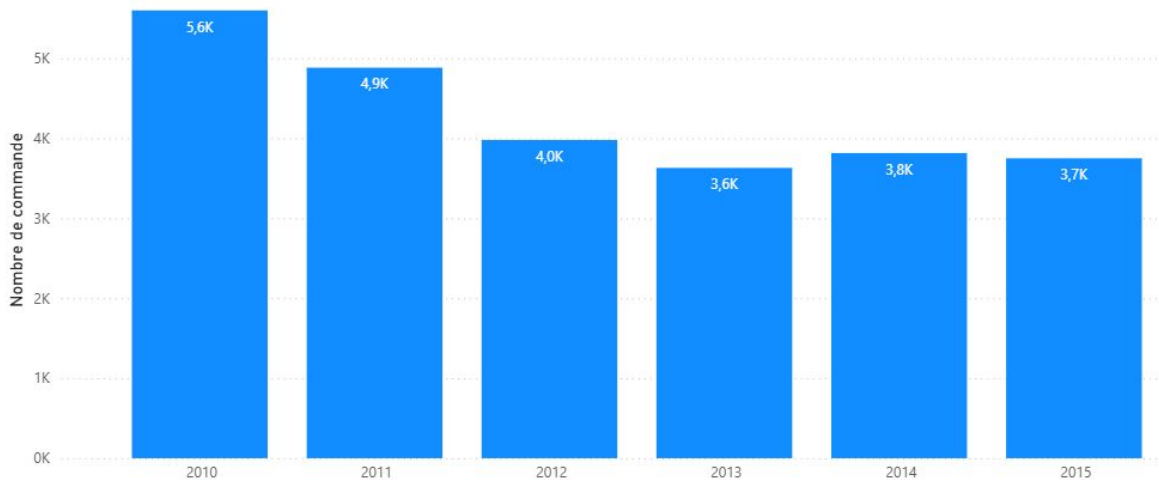
- ❖ Meilleure commande de Nantes en 2020
- ❖ Client avec le plus de frais de timbres commande
- ❖ Nombre de commande par années de 2010 à 2015

#### Meilleure commande de Nantes en 2020

Code commande	Quantité d'objets commandés	Nombre de timbres commande
85301	6	3,05

#### Client avec le plus de frais de timbres commande

Nom	Prénom	Nombre de timbres commande	Nombre de commandes	Nombre d'objets commandés
Crotte	Nicole	303,40	57	142



# Analyses supplémentaires



## ❖ Nombre de commande par année

Annee

☐ Sélectionner tout

☐ 2000

☐ 2001

☐ 2002

☐ 2003

☐ 2004

☐ 2005

☐ 2006

☐ 2007

☐ 2008

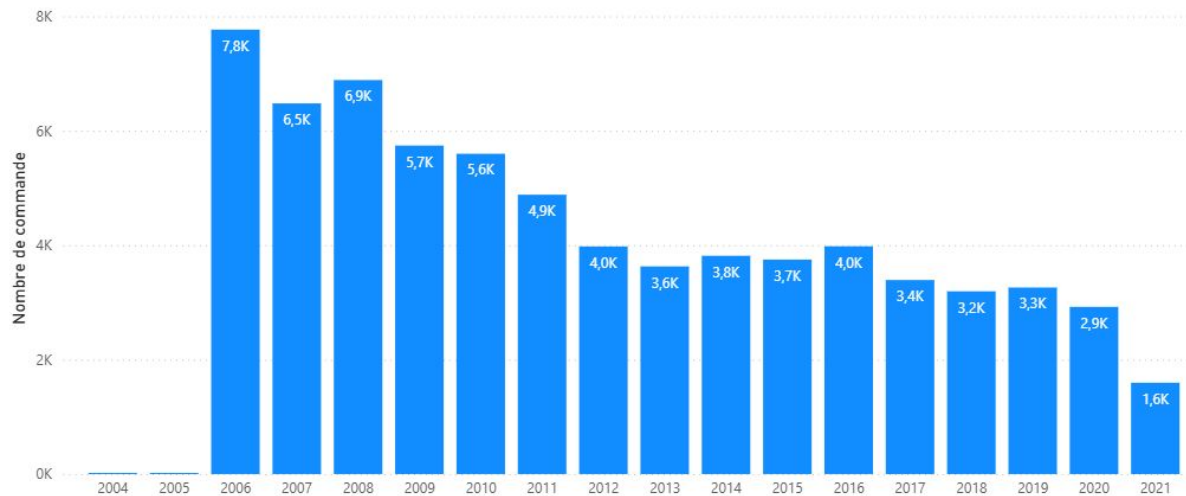
☐ 2009

☐ 2010

☐ 2011

☐ 2012

Nombre de commande par année





Nombre de commandes

10,647K

Quantité d'objets commandés

23K

Année

- ☐
- 2006
- 
- ☐
- 2007
- 
- ☐
- 2008
- 
- ☐
- 2009
- 
- ☐
- 2010

Département

- ☐
- 28
- 
- ☐
- 53
- 
- ☐
- 61

Nombre de commandes

3,637K

Quantités d'objets commandés

11K

Année

- ☐
- 2011
- 
- ☐
- 2012
- 
- ☐
- 2013
- 
- ☐
- 2014
- 
- ☐
- 2015
- 
- ☐
- 2016

Département

- ☐
- 22
- 
- ☐
- 49
- 
- ☐
- 53

## Informations totales

Nombre de commandes

70,892K

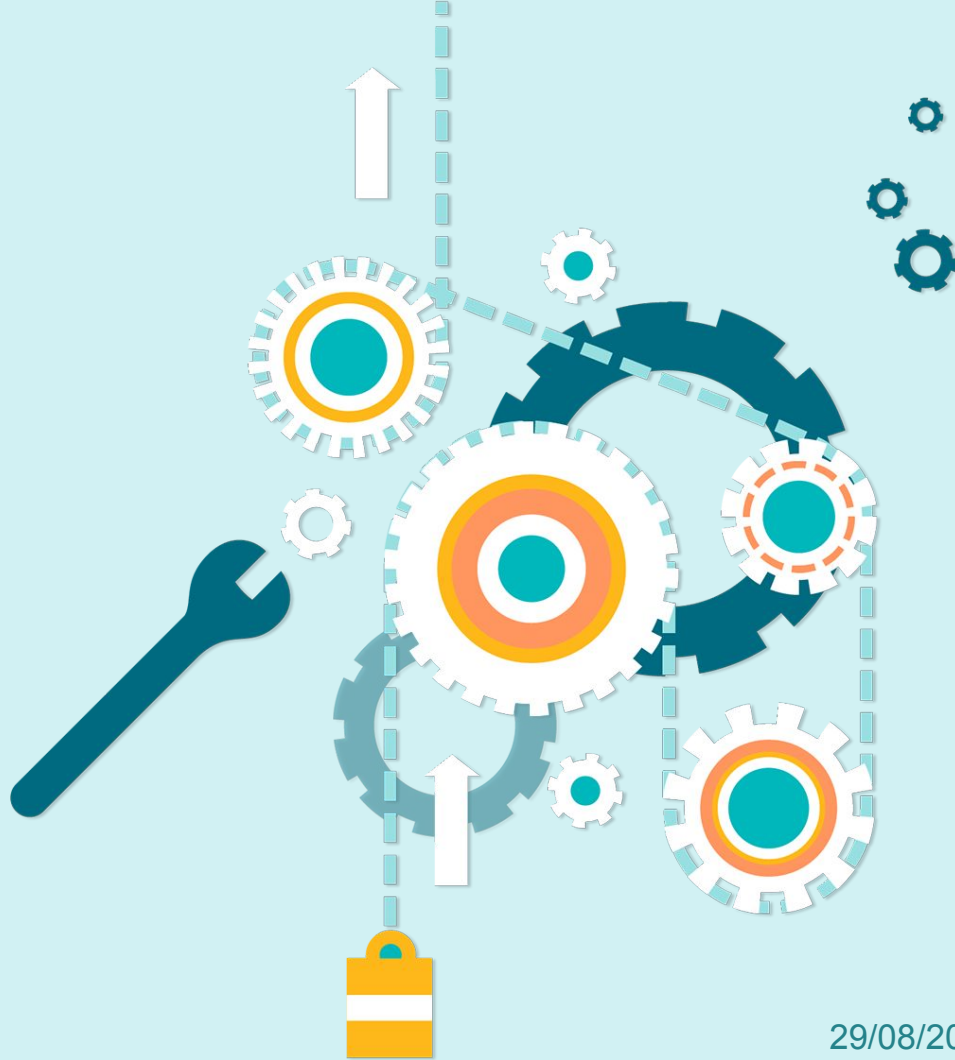
Quantités d'objets commandés

195K

Année

Tout

# Synthèse





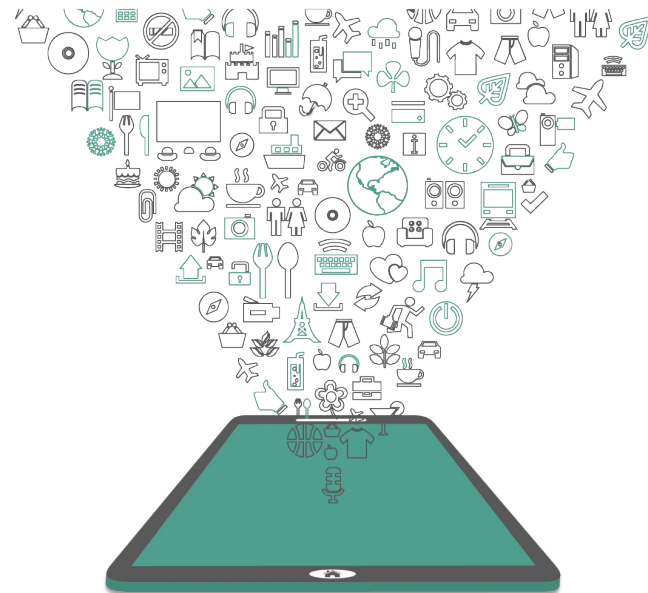
## Résumé des résultats majeurs

- Diminution des commandes au cours du temps
- Majorité des commandes dans le nord-ouest de la France
- Grand nombre d'articles sans timbres clients (lot 2)

## Synthèse du projet

### Enjeux :

- Nettoyer et qualifier les données brutes fournies en CSV
- Les stocker dans une architecture Big Data adaptée (HDFS, HBase)
- Produire des analyses exploitables au travers de visualisations interactives dans PowerBI.



## Possibilités d'exploitation futures

- Automatiser le pipeline complet (ETL → HBase → PowerBI) afin de rendre l'analyse reproductible en temps réel.
- Étendre le modèle aux données postérieures à 2021 pour un suivi plus actuel.
- Élargir le périmètre d'analyse (par produit, par saison, par zone géographique) afin d'offrir une meilleure compréhension de ses ventes et clients.



# Merci pour votre attention



# Digicheese