

Projet Foody

Nanterre P10_ Dev data

Zhiying, Nedjma

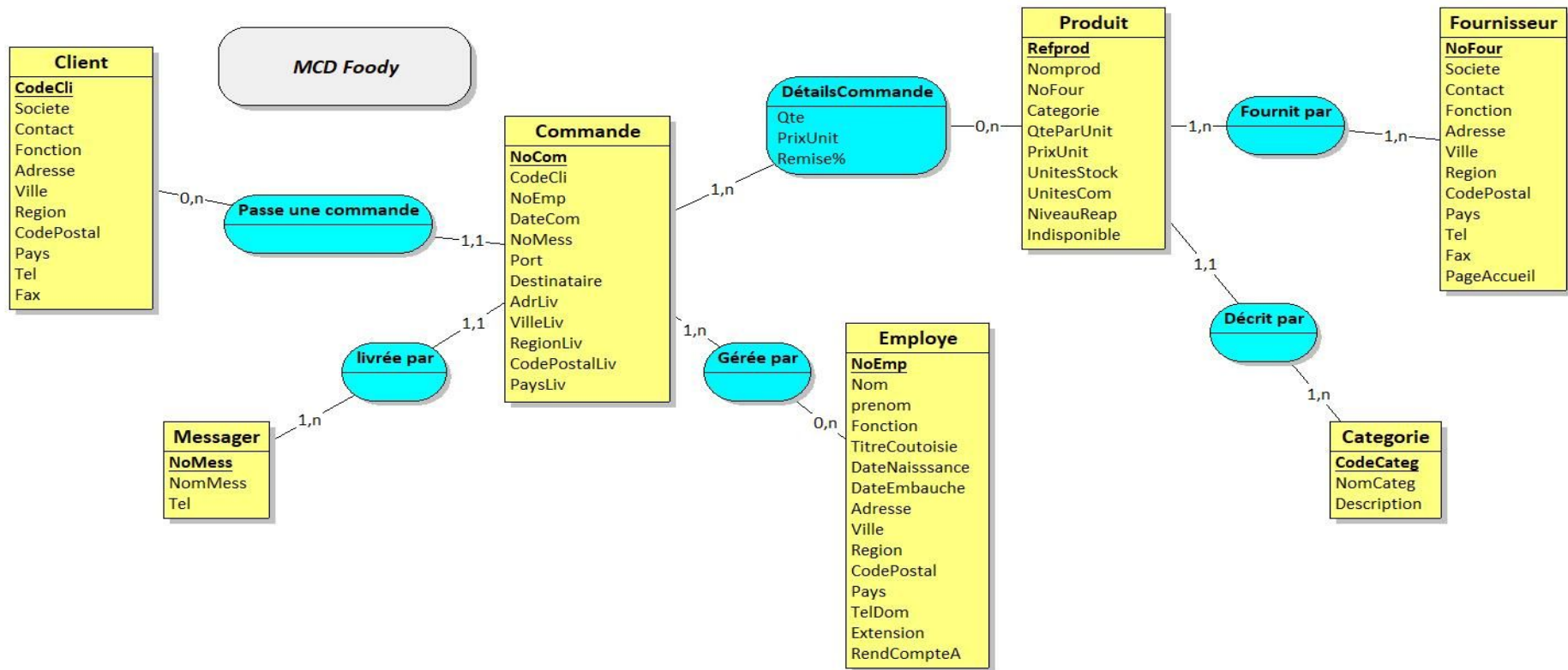
12/02/2021



TABLE DES MATIÈRES

- 1) Modèle conceptuel de données
- 2) Modèle logique de données
- 3) Modèle physique de données
- 4) Quelques requêtes SQL
- 5) Exemples de visualisations
- 6) Difficultés rencontrées et points positifs

I. MCD



II. MLD



Categorie (CodeCateg, NomCateg, Description)

Messenger (NoMess, NomMess, Tel)

Client (CodeCli, Societe, Contact, Fonction, Adresse, Ville, Region, CodePostal, Pays, Tel, Fax)

Fournisseur (NoFour, Societe, Contact, Fonction, Adresse, Ville, Region, CodePostal, Pays, Tel, Fax, PageAccueil)

Employe (NoEmp, Nom, Prenom, Fonction, TitreCourtoisie, DateNaissance, DateEmbauche, Adresse, Ville, Region, CodePostal, Pays, TelDom, Extension, RendCompteA*)

Produit (RefProd, NomProd, NoFour*, CodeCateg*, QteParUnit, PrixUnit, UnitesStock, UnitesCom, NiveauReap, Indisponible)

Commande (NoCom, CodeCli*, NoEmp*, DateCom, ALivAvant, DateEnv, NoMess*, Port, Destinataire, AdrLiv, VilleLiv, RegionLiv, CodePostalLiv, PaysLiv)

DetailsCommande (IdDetail, NoCom*, RefProd*, PrixUnit, Qte, Remise%)

III. MPD

3.1 Tables categorie et messenger



```
CREATE DATABASE Foody;
```

```
USE Foody;
```

```
CREATE TABLE categorie (
```

```
CodeCateg integer primary key NOT NULL,
```

```
NomCateg varchar(25) NOT NULL,
```

```
Descriptionn varchar(100)
```

```
);
```

```
CREATE TABLE messenger (
```

```
NoMess integer primary key NOT NULL,
```

```
NomMess varchar(25) NOT NULL,
```

```
Tel varchar(25) NOT NULL
```

```
);
```


3.2 Table client



```
CREATE TABLE client (  
  
CodeCli char(5) primary key NOT NULL,  
  
Societe varchar(100),  
  
Contact varchar(25) NOT NULL,  
  
Fonction varchar(100),  
  
Adresse varchar(100),
```

```
Ville varchar(25),  
  
Region varchar(25),  
  
Codepostal varchar(25),  
  
Pays varchar(25),  
  
Tel varchar(25),  
  
Fax varchar(25)  
  
);
```


3.3 Table fournisseur



```
CREATE TABLE fournisseur (  
  
NoFour integer primary key NOT NULL,  
  
Societe varchar(100) NOT NULL,  
  
Contact varchar(100) NOT NULL,  
  
Fonction varchar(100),  
  
Adresse varchar(100),  
  
Ville varchar(25),
```

```
Region varchar(25),  
  
CodePostal varchar(25),  
  
Pays varchar(25),  
  
Tel varchar(25),  
  
Fax varchar(25),  
  
PageAccueil varchar(100)  
  
);
```

3.4 Table employé



```
CREATE TABLE employe (  
  
NoEmp integer primary key NOT NULL,  
  
Nom varchar(25) NOT NULL,  
  
Prenom varchar(25) NOT NULL,  
  
Fonction varchar(25) NOT NULL,  
  
TitreCourtoisie varchar(25),  
  
DateNaissance datetime,  
  
DateEmbauche datetime,
```

```
Adresse varchar(50),  
  
Ville varchar(25),  
  
Region varchar(25),  
  
Codepostal varchar(25),  
  
Pays varchar(25),  
  
TelDom varchar(25),  
  
Extension int,  
  
RendCompteA int  
  
);
```

Remarque: on rajoute notre clé étrangère RendCompte A à la main après avoir rentré nos données.

3.5 Table produit



```
CREATE TABLE produit (  
RefProd integer primary key NOT NULL,  
NomProd varchar(100) NOT NULL,  
NoFour integer NOT NULL,  
CodeCateg integer NOT NULL,  
QteParUnit varchar(25),  
PrixUnit float,
```

```
UnitesStock integer,  
UnitesCom integer,  
NiveauReap int,  
Indisponible int,  
foreign key (NoFour) references  
fournisseur (NoFour),  
foreign key (CodeCateg) references  
categorie (CodeCateg)  
);
```

3.6 Table commande



```
CREATE TABLE Commande (  
  
NoCom integer primary key NOT NULL,  
  
CodeCli char(5) NOT NULL,  
  
NoEmp integer NOT NULL,  
  
DateCom datetime NOT NULL,  
  
ALivAvant datetime,  
  
DateEnv datetime,  
  
NoMess integer NOT NULL,  
  
Portt float,
```

```
Destinataire varchar(100) NOT NULL,  
  
AdrLiv varchar(100) NOT NULL,  
  
VilleLiv varchar(25) NOT NULL,  
  
RegionLiv varchar(25),  
  
CodePostalLiv varchar(25),  
  
PaysLiv varchar(25) NOT NULL,  
  
foreign key (CodeCli) references client  
(CodeCli),  
  
foreign key (NoEmp) references employe  
(NoEmp),  
  
foreign key (NoMess) references messenger  
(NoMess));
```

3.7 Table detailsCommande



```
CREATE TABLE detailsCommande (
```

```
  IdDetail integer primary key  
  auto_increment,
```

```
  NoCom integer NOT NULL,
```

```
  RefProd integer NOT NULL,
```

```
  PrixUnit float,
```

```
  Qte integer,
```

```
  Remise float,
```

```
  foreign key (NoCom) references  
  commande (NoCom),
```

```
  foreign key (RefProd) references produit  
  (RefProd)
```

```
);
```

Remarque: on a décidé de créer un IdDetail comme primary key.

On aurait pu créer une **primary key composée** (NoCom, RefProd)

IV. Quelques requêtes SQL

4.1 Jointures à la main : exercice 1

#EXERCICE 14: (p28)

#1.Récupérer les informations des fournisseurs pour chaque produit, avec jointure à la main

```
query = "" SELECT produit.NomProd, fournisseur.* FROM fournisseur, produit
```

```
WHERE fournisseur.NoFour = produit.NoFour; ""
```

```
sql_to_df(query).head()
```

Out[8]:

	NomProd	NoFour	Societe	Contact	Fonction	Adresse	Ville	Region	CodePostal	Pays	Tel	Fax	PageAccueil
0	Chai	1	Exotic Liquids	Charlotte Cooper	Purchasing Manager	49 Gilbert St.	London	None	EC1 4SD	UK	(171) 555-2222	None	None
1	Chang	1	Exotic Liquids	Charlotte Cooper	Purchasing Manager	49 Gilbert St.	London	None	EC1 4SD	UK	(171) 555-2222	None	None
2	Aniseed Syrup	1	Exotic Liquids	Charlotte Cooper	Purchasing Manager	49 Gilbert St.	London	None	EC1 4SD	UK	(171) 555-2222	None	None
3	Chef Anton's Cajun Seasoning	2	New Orleans Cajun Delights	Shelley Burke	Order Administrator	P.O. Box 78934	New Orleans	LA	70117	USA	(100) 555-4822	None	#CAJUN.HTM#
4	Chef Anton's Gumbo Mix	2	New Orleans Cajun Delights	Shelley Burke	Order Administrator	P.O. Box 78934	New Orleans	LA	70117	USA	(100) 555-4822	None	#CAJUN.HTM#

Suite, Jointures à la main : exercice 2

#EXERCICE 14: (p28)

#2.Afficher les informations des commandes du client "Lazy K Kountry Store", avec jointure à la main

```
query = "" SELECT client.societe, commande.* FROM commande,client
```

```
WHERE commande.CodeCli = client.CodeCli AND client.societe = "Lazy K Kountry Store"; ""
```

```
sql_to_df(query)
```

Out[54]:

	societe	NoCom	CodeCli	NoEmp	DateCom	ALivAvant	DateEnv	NoMess	Portt	Destinataire	AdrLiv	VilleLiv	RegionLiv	CodePostalLiv	PaysLiv
0	Lazy K Kountry Store	10482	LAZYK	1	2007-03-21	2007-04-18	2007-04-10	3	7.48	Lazy K Kountry Store	12 Orchestra Terrace	Walla Walla	WA	99362	USA
1	Lazy K Kountry Store	10545	LAZYK	8	2007-05-22	2007-06-19	2007-06-26	2	11.92	Lazy K Kountry Store	12 Orchestra Terrace	Walla Walla	WA	99362	USA

Suite, Jointures à la main: exercice 3



#EXERCICE 14: (p28)

#3.Afficher le nombre de commande pour chaque messenger (en indiquant son nom), avec jointure à la main

```
query = "" SELECT Messenger.NomMess, COUNT(DISTINCT Commande.NoCom) AS "nb_commande/messenger"
FROM Messenger, Commande WHERE messenger.NoMess = commande.NoMess
GROUP BY NomMess; ""
```

sql_to_df(query)

Out[55]:

	NomMess	nb_commande/messenger
0	Federal Shipping	255
1	Speedy Express	249
2	United Package	326

4.2 Sous-requêtes : exercice 1

#EXERCICE 15: (p30)

#1.Lister les employés n'ayant jamais effectué une commande, via une sous-requête

```
query = ''' SELECT * FROM employe
WHERE NoEmp NOT IN(SELECT NoEmp FROM commande WHERE NoEmp = employe.NoEmp); '''

sql_to_df(query)
```

Out[96]:

	NoEmp	Nom	Prenom	Fonction	TitreCourtoisie	DateNaissance	DateEmbauche	Adresse	Ville	Region	Codepostal	Pays	TelDom	Extension	Rei
0	5	Buchanan	Steven	Sales Manager	Mr.	1955-03-04	1993-10-17	14 Garrett Hill	London	None	SW1 8JR	UK	(71) 555- 4848	3453	

Suite, sous-requêtes: exercice 2

#EXERCICE 15: (p30)

#2. Nombre de produits proposés par la société fournisseur "Ma Maison", via une sous-requête

```
query = " SELECT COUNT(*) AS nb_produit FROM produit
```

```
WHERE NoFour IN (SELECT NoFour FROM fournisseur WHERE fournisseur.Societe = "Ma Maison"); "
```

```
sql_to_df(query)
```

Out[97]:

nb_produit	
0	2

Suite, sous-requêtes: exercice 3

#EXERCICE 15: (p30)

#3. Nombre de commandes passées par des employés sous la responsabilité de "Buchanan Steven"

```
query = """ SELECT COUNT(*) AS nb_commande FROM Commande WHERE NoEmp IN  
(SELECT NoEmp FROM employe WHERE RendCompteA IN  
(SELECT NoEmp FROM employe WHERE Nom = "Buchanan" AND Prenom = "Steven")); """  
sql_to_df(query)
```

Out[100]:

nb_commande	
0	224

V. Exemples de visualisation

5.1 Le factor plot



Information recherchée: on s'intéresse aux différents produits commandés par nos pays clients

```
On fait la requête suivante:

query = """SELECT PaysLiv, COUNT(DISTINCT RefProd) AS nb_de_produit FROM produit

JOIN detailscommande USING(RefProd)

JOIN commande USING(NoCom)

GROUP BY Paysliv

ORDER BY PaysLiv; """

df=sql_to_df(query)

df
```

Out[6]:

	PaysLiv	nb_de_produit
0	Argentina	8
1	Austria	39
2	Belgium	17
3	Brazil	46
4	Canada	35
5	Denmark	19
6	Finland	23
7	France	51
8	Germany	67
9	Ireland	23
10	Italy	15
11	Mexico	34
12	Norway	5
13	Poland	5
14	Portugal	19
15	Spain	19
16	Sweden	30
17	Switzerland	18
18	UK	46

Le factor plot



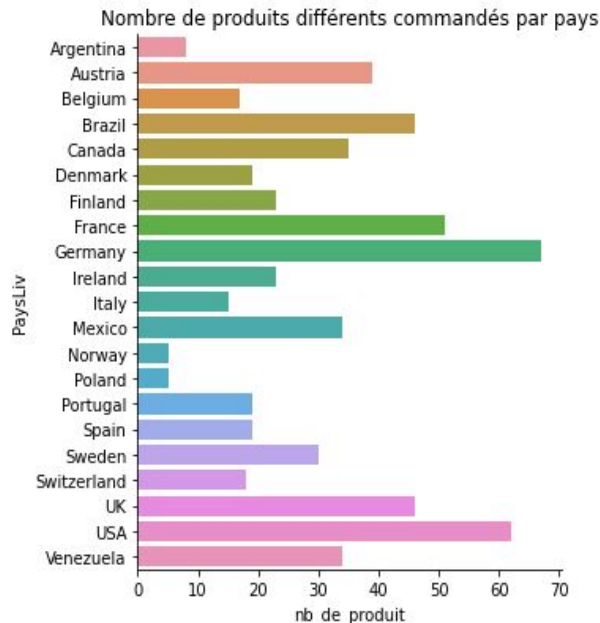
```
sns.factorplot(x='nb_de_produit',y='PaysLiv',data=df,kind='bar')

plt.title("Nombre de produits différents commandés par pays")
```

Out[7]: Text(0.5, 1.0, 'Nombre de produits différents commandés par pays')

- *On remarque que l'Allemagne et les Etats-Unis ont commandé le plus de produits différents, au contraire de l'Argentine, la Norvège et la Pologne qui commandent peu de produits différents.*

→ *Du coup, on se demande ce que l'Argentine, la Norvège et la Pologne consomment...*



5.2 Le strip plot



Information recherchée: on s'intéresse aux différents produits commandés par la Norvège, l'Argentine et la Pologne

On fait la requête suivante:

```
query = """SELECT PaysLiv, NomProd FROM commande
JOIN detailsCommande USING(NoCom)
JOIN produit USING(RefProd)
WHERE commande.PaysLiv IN("Argentina", "Norway", "Poland")
GROUP BY PaysLiv, NomProd
ORDER BY PaysLiv; """

df1 = sql_to_df(query)

df1
```

Out[10]:

	PaysLiv	NomProd
0	Argentina	Boston Crab Meat
1	Argentina	Gumbr Gummibirchen
2	Argentina	Jack's New England Clam Chowder
3	Argentina	Raclette Courdavault
4	Argentina	Scottish Longbreads
5	Argentina	Sir Rodney's Scones
6	Argentina	Steeleye Stout
7	Argentina	Tofu
8	Norway	Flotemysost
9	Norway	Guaran Fantstica
10	Norway	Perth Pasties

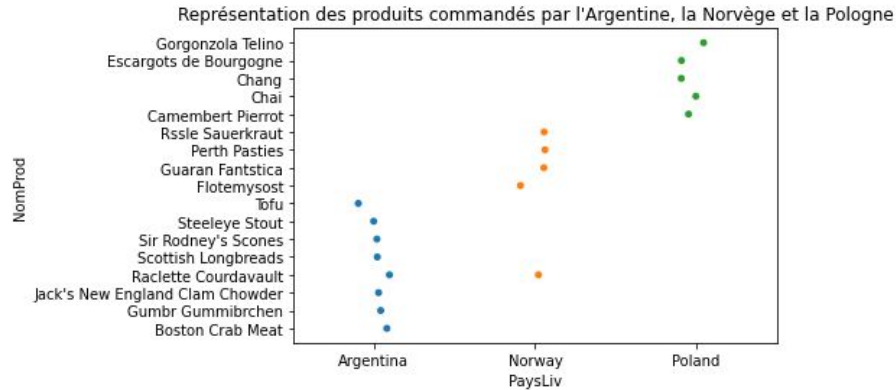
Le strip plot



```
sns.stripplot(x="PaysLiv", y="NomProd", data=df1)
```

```
plt.title("Représentation des produits commandés par l'Argentine, la  
Norvège et la Pologne")
```

```
Out[11]: Text(0.5, 1.0, "Représentation des produits commandés par l'Argentine, la Norvège et la Pologne")
```



- *Ces trois pays n'ont pas les mêmes consommations de produits Foody. Quasi aucun produit en commun!*

→ *Des conséquences en termes de stratégie marketing?*

5.3 Le heat map

Information recherchée: plus globalement, on s'intéresse aux catégories de produits commandées par nos pays clients

```
On fait la requête suivante:

query = "" SELECT PaysLiv, NomCateg, COUNT(DISTINCT NoCom)
AS nb_com_diff FROM categorie

LEFT JOIN produit USING(CodeCateg)

LEFT JOIN detailscommande USING(RefProd)

LEFT JOIN commande USING(NoCom)

GROUP BY PaysLiv, CodeCateg

ORDER BY Paysliv, CodeCateg; ""

df2 = sql_to_df(query)

df2
```

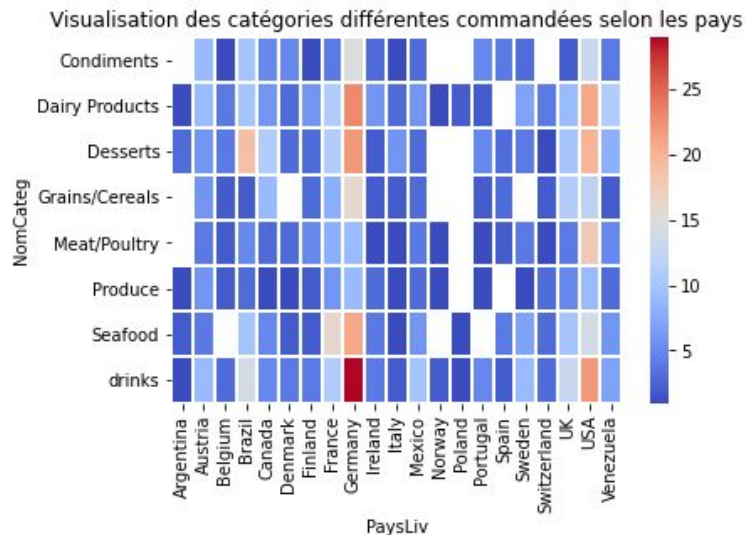
Out[12]:

	PaysLiv	NomCateg	nb_com_diff
0	None	Meat/Poultry	0
1	Argentina	drinks	1
2	Argentina	Desserts	3
3	Argentina	Dairy Products	1
4	Argentina	Produce	1
...
145	Venezuela	Dairy Products	11
146	Venezuela	Grains/Cereals	2
147	Venezuela	Meat/Poultry	5
148	Venezuela	Produce	3
149	Venezuela	Seafood	6

150 rows x 3 columns

Le heat map

Out[20]: Text(0.5, 1.0, 'Visualisation des catégories différentes commandées selon les pays')



```
nProduitPays = df2.pivot_table(values =  
'nb_com_diff', index = 'NomCateg', columns =  
'PaysLiv')
```

```
sns.heatmap(nProduitPays, cmap = 'coolwarm',  
linecolor = 'white', linewidths = 1)
```

```
plt.title("Visualisation des catégories différentes  
commandées selon les pays")
```

- *L'Allemagne et les USA se démarquent: on sait déjà qu'ils commandent plus de produits différents,
→ maintenant on sait aussi qu'ils commandent des catégories diversifiées*

5.4 Le map chart

Information recherchée: on change de sujet! Intéressons-nous maintenant aux meilleurs ventes par pays

On fait la requête suivante:

```
query = "" SELECT Pays, ROUND(SUM(PrixUnit),2) AS "Somme des ventes"
FROM detailscommande
```

```
LEFT JOIN commande USING(NoCom)
```

```
JOIN client USING(CodeCli)
```

```
GROUP BY client.Pays ORDER BY client.Pays;"
```

```
df5 = sql_to_df(query)
```

On ajoute une colonne pour les besoins de la visualisation:

```
df5['location']=[[-38.42,-63.58],[-26.26,133.16],[50.50,4.48],[-14.24,-53.19],[61.07,-1
07.99],[56.5,9.51],[65.14,27.78],[48.52,2],[51.18,10.80],[53.17,-8.29],[44.92,10.93],[2
5.12,-102.83],[61.09,10.41],[52.66,19.30],[40.58,-7.83],[40.46,-3.74],[63.78,16.16],[46
.83,8.12],[51.50,-0.12],[37,-95.71],[8.57,-68.20]]
```

Out[47]:

	Pays	Somme des ventes	location
0	Argentina	161.35	[-38.42, -63.58]
1	Austria	1690.90	[-26.26, 133.16]
2	Belgium	473.10	[50.5, 4.48]
3	Brazil	2061.13	[-14.24, -53.19]
4	Canada	1213.90	[61.07, -107.99]
5	Denmark	616.25	[56.5, 9.51]
6	Finland	687.64	[65.14, 27.78]
7	France	2100.15	[48.52, 2]
8	Germany	3786.83	[51.18, 10.8]
9	Ireland	722.75	[53.17, -8.29]
10	Italy	300.50	[44.92, 10.93]
11	Mexico	1288.40	[25.12, -102.83]
12	Norway	138.50	[61.09, 10.41]
13	Poland	91.60	[52.66, 19.3]
14	Portugal	439.05	[40.58, -7.83]
15	Spain	397.20	[40.46, -3.74]
16	Sweden	807.09	[63.78, 16.16]
17	Switzerland	455.65	[46.83, 8.12]
18	UK	1430.74	[51.5, -0.12]

Le map chart

```
worldmap = folium.Map([25, 13], zoom_start=2)

for i in range(len(df5)):

    folium.Marker(location = df5.location[i], popup=df5.Pays[i] + ": " + str(df5["Somme des ventes"][i]),
    tooltip = df5.Pays[i] + ": " + str(df5["Somme des ventes"][i])).add_to(worldmap)

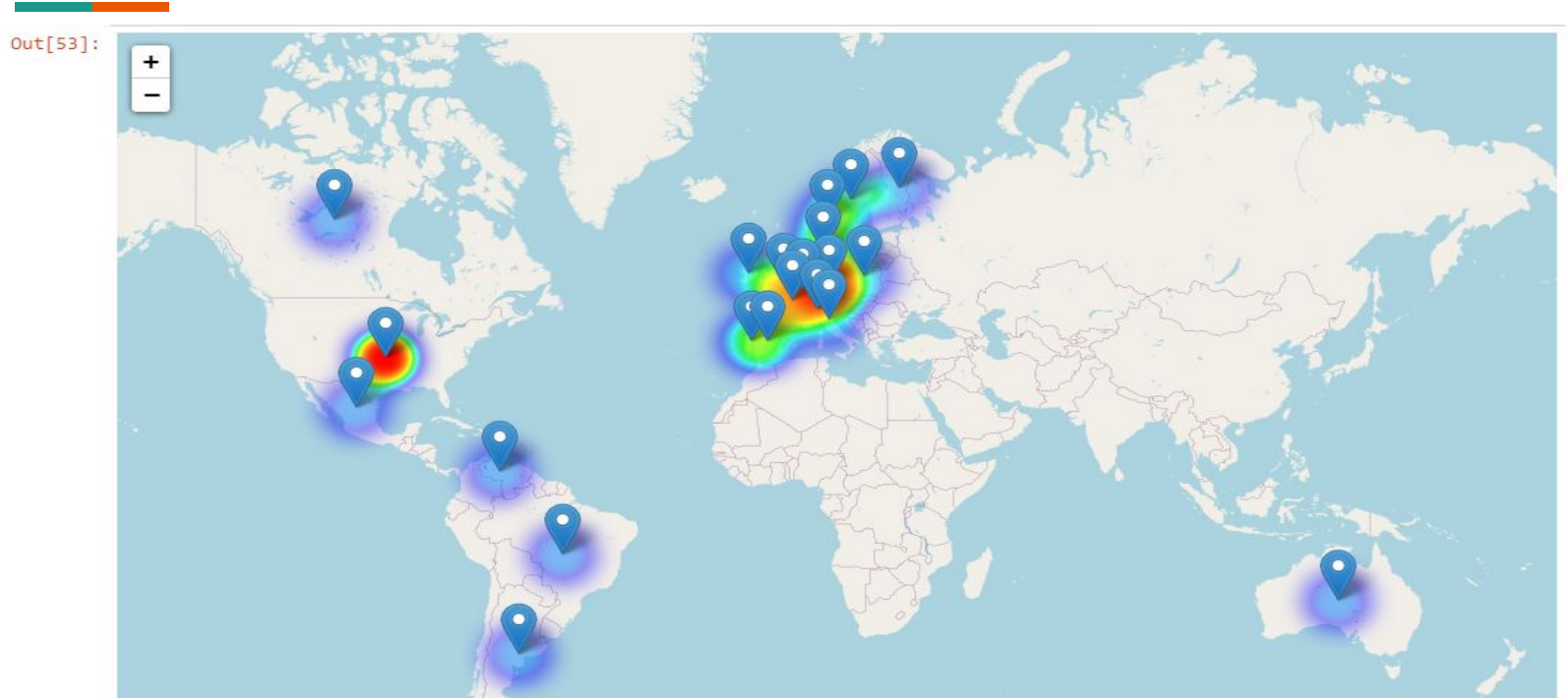
data1=[[df5.location[i][0],df5.location[i][1],df5["Somme des ventes"][i]] for i in range(len(df5))]

worldmap.add_child(HeatMap(data1))

worldmap
```

Le map chart

CARTE



- *Visuellement, c'est probant: on vend plus aux USA et en Europe. On pourra zoomer pour regarder de plus près l'Europe*

VI. Difficultés rencontrées et points positifs



Difficultés rencontrées	Points positifs
<ul style="list-style-type: none"><li data-bbox="247 494 909 598">● Comprendre quand mettre en place une clé composée<li data-bbox="247 696 562 734">● Sous-requêtes	<ul style="list-style-type: none"><li data-bbox="1012 494 1534 532">● S'adapter, travail en équipe<li data-bbox="1012 631 1572 734">● Meilleure compréhension des sous-requêtes<li data-bbox="1012 833 1547 871">● Un binôme complémentaire!

**Merci pour votre attention !
Zhiying & Nedjma,
pour le projet Foody**

