



# Rapport Projet Power BI

Présenté par:  
Hind MOUTALATTIF  
Khaoula JERRARI  
Hajar ZAINI

Encadré par:  
Pr. AKKAOUI Zineb

• • •



# Exercice 1:



## Tâche 1:

Tracez les données de ventes de 2011 à 2017. Expliquez les tendances que vous observez :

Afin d'obtenir une visualisation qui montre l'évolution en fonction des années on doit pivoter et préparer les données car ils sont à la base sous forme d'une table croisée

Premièrement on obtient nos données:

The screenshot shows the Power BI desktop application. On the left, a dark sidebar menu lists options: New, Open report (highlighted), Save, Save as, Get data (highlighted), Import, Export, Publish, Options and settings, and Get started. The main area is titled 'Get data' and shows 'Most Common' data sources: Excel workbook, Power BI datasets, Power BI dataflows, SQL Server database, and Azure Analysis Services database. Below this, a preview of a table named 'Sheet1' is shown. The table has columns labeled 'Column1' through 'Column6' and a 'Color' column. The data is as follows:

Column1	Column2	Column3	Column4	Column5	Column6	Color
null	null	null	null	null	null	
MONTH	2011	2012	2013	2014	2015	
January	10	12	11	12	11	
February	7	10	14	13	9	
March	12	16	13	15	20	
April	10	10	11	10	13	
May	19	19	16	17	19	
June	19	21	21	20	21	
July	7	21	16	15	18	
August	22	19	27	20	22	
September	12	12	11	12	16	
October	14	12	14	7	9	
November	6	9	10	11	11	
December	25	21	19	17	13	
	null	null	null	null	null	

Voilà la forme des données à la base:

Ensuite on unpivot la table croisée en utilisant la fonction sélectionnée sur l'image ci-dessus:

The screenshot shows the Power BI Data Editor interface. A context menu is open over the 'MONTH' column header. The menu includes options like Copy, Remove, Remove Other Columns, Duplicate Column, Add Column From Examples..., Remove Duplicates, Remove Errors, Change Type, Transform, Replace Values..., Replace Errors..., Split Column, Group By..., Fill, Unpivot Columns, Unpivot Other Columns (which is highlighted in grey), and Unpivot Only Selected Columns. The 'Unpivot Other Columns' option is the one being used to unpivot the table.

Et voilà la forme finale des données:

	MONTH	year	sales
1	January	2011	10
2	January	2012	12
3	January	2013	11
4	January	2014	12
5	January	2015	11
6	January	2016	13
7	January	2017	20
8	February	2011	7
9	February	2012	10
10	February	2013	14
11	February	2014	13
12	February	2015	9
13	February	2016	11
14	February	2017	16
15	March	2011	12
16	March	2012	16
17	March	2013	13
18	March	2014	15
19	March	2015	20
20	March	2016	22

Ensuite on utilise la fonction "Merge" afin de concaténer les colonnes Year et MONTH afin de bien présenter la visualisation:

The screenshot shows the Power BI Data Editor interface. A context menu is open over the 'MONTH' column header. The menu includes options like Copy, Remove Columns, Remove Other Columns, Add Column From Examples..., Remove Duplicates, Remove Errors, Replace Values..., Fill, Change Type, Transform, Merge Columns (which is highlighted in grey), Group By..., Unpivot Columns, Unpivot Other Columns, and Unpivot Only Selected Columns. The 'Merge Columns' option is the one being used to merge the columns.

### Merge Columns

Choose how to merge the selected columns.

Separator

Space

New column name (optional)

date

Activate Windows  
Or go to Settings to activate Windows.

OK

Cancel

Voilà le résultat du premier merge():

	A date	1.2 sales
1	January 2011	10
2	January 2012	12
3	January 2013	11
4	January 2014	12
5	January 2015	11
6	January 2016	13
7	January 2017	20
8	February 2011	7
9	February 2012	10
10	February 2013	14
11	February 2014	13
12	February 2015	9
13	February 2016	11
14	February 2017	16
15	March 2011	12
16	March 2012	16
17	March 2013	13
18	March 2014	15
19	March 2015	20
20	March 2016	22

Ensuite on change le type des données en Date

The screenshot shows the Power BI Data Editor interface. A context menu is open over the 'date' column, specifically over the first row 'January 2011'. The menu path 'Change Type' -> 'Date' is highlighted. A sub-menu for 'Date' is open, showing options: Decimal Number, Fixed decimal number, Whole Number, Percentage, Date/Time, Date, Time, Date/Time/Timezone, Duration, Text (which is checked), True/False, Binary, and Using Locale... The 'APPLIED STEPS' pane on the right shows the 'Changed Type' step, with 'Date' selected. The 'Source' step is also listed.

	A date	1.2 sales
1	1/1/2011	10
2	1/1/2012	12
3	1/1/2013	11
4	1/1/2014	12
5	1/1/2015	11
6	1/1/2016	13
7	1/1/2017	20
8	2/1/2011	7
9	2/1/2012	10
10	2/1/2013	14
11	2/1/2014	13
12	2/1/2015	9
13	2/1/2016	11
14	2/1/2017	16
15	3/1/2011	12
16	3/1/2012	16
17	3/1/2013	13
18	3/1/2014	15

Et on obtient les données sous cette forme finale.



Voilà la visualisation qu'on a obtenu, ici on observe une augmentation légère au niveau des ventes entre 2011 et 2012, suivi d'une sorte de stabilisation entre 2012 et 2013 et une diminution en 2014. On observe aussi qu'à 2017 on avait une forte augmentation grâce à un fort concurrent qui a quitté le marché, en laissant un marché potentiel.

### Tâche 2:

Supposons que la date d'aujourd'hui est le 1er janvier 2018 et qu'il vous a été demandé de prévoir les chiffres des ventes pour chaque mois en 2018. Remplissez la colonne 2018 du tableau 1 avec les ventes prévues pour chaque mois.

Puisque là on est demandé de prévoir les chiffres des ventes pour chaque mois en 2018 on prend en considération aussi la colonne Month:

**Fields** >>

Search

Sheet1 (5)

date

Date Hierarchy

Year

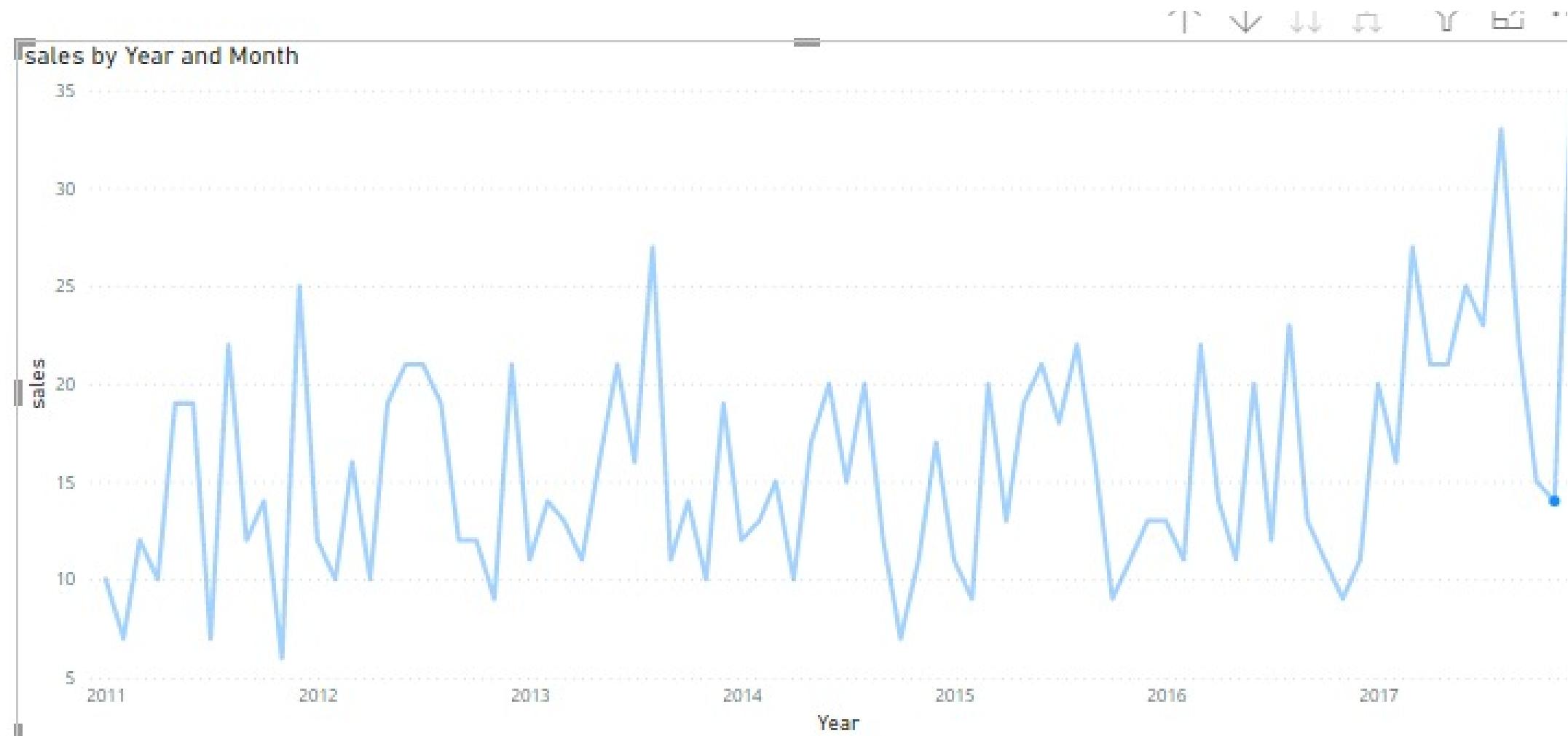
Quarter

Month

Day

$\sum$  sales

En faisant les modifications précédentes, on obtient le plot suivant où les points sont déterminer chaque mois



Puisque le concurrent a décidé de quitter son emplacement à la fin de 2016, on doit prendre en considération les mois de 2017 seulement, dans ce cas là on filtre les données en choisissant seulement l'année 2017

**Filters**

Search:

**sales**  
is (All)

**date - Year**  
is 2017

Filter type ⓘ

Basic filtering

Select all

2011

2012

2013

2014

2015

2016

2017

Require single selection

**date - Month**  
is (All)

Filter type ⓘ

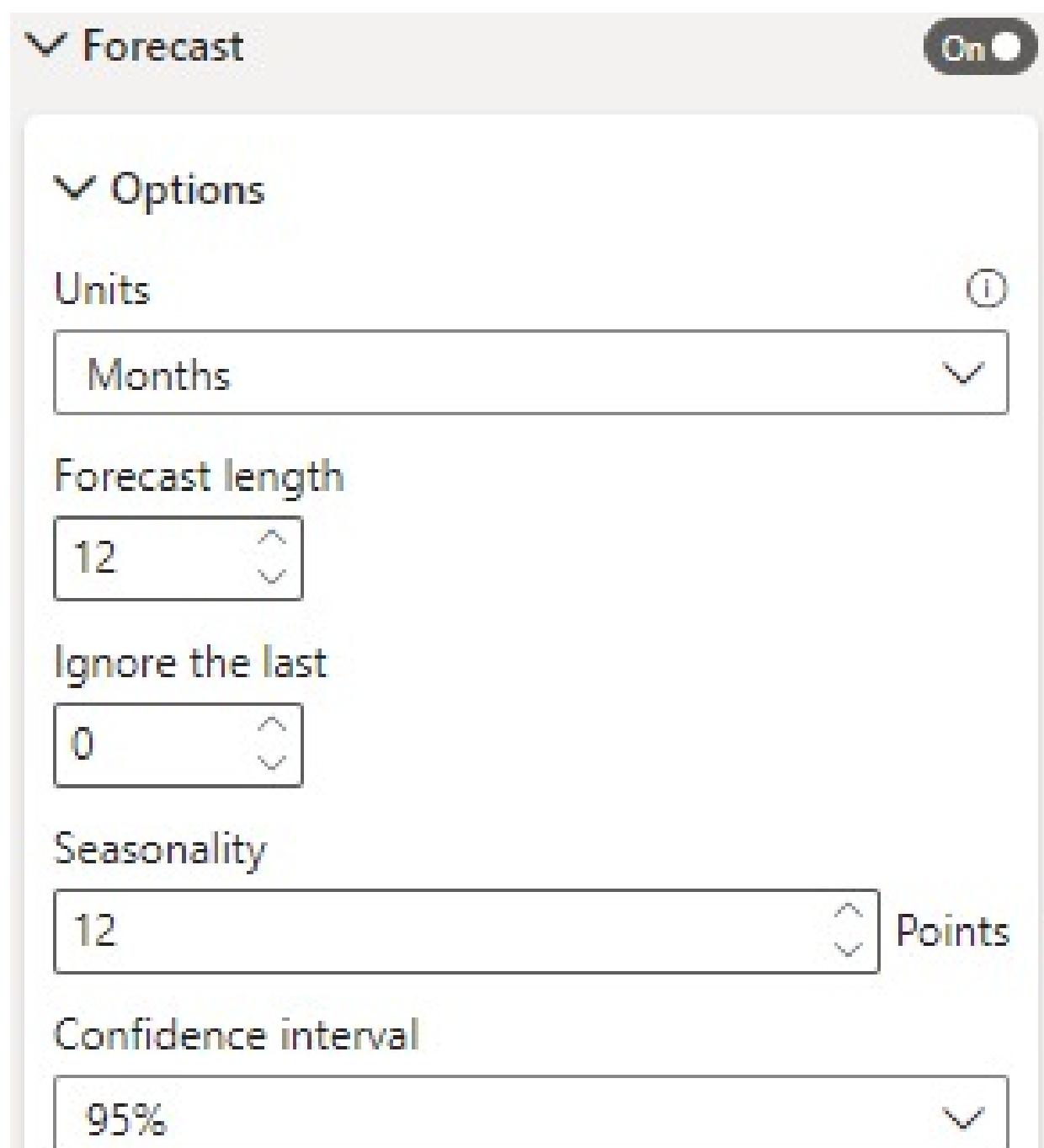
Ensuite on commence à modifier les paramètres de "Forecast" :

Pour "Units": on choisit "Months" car notre prédiction se fera en mois.

Pour "Forecast length" on choisit 12 mois afin de prédire tous les mois de 2018.

Pour "Ignore the last": on a choisi 0 car on va utiliser tous les ventes de mois chaque mois lors de la prédiction.

Pour "Seasonality": on choisit 12 car on a 12 mois et on a choisi 95% pour "Confidence interval"



Et voilà le résultat des prédictions:



Ensuite on doit Remplir la colonne 2018 du tableau 1 avec les ventes prévues pour chaque mois.

Afin de remplir la colonne 2018 on a exporté les données en fichier excel et on a supprimé les données de 2017 afin d'avoir les données demandées qui concernent 2018



	A	B	C	D	E	F
1	Year	Month	Sales			
2	2018	January	35			
3	2018	February	20			
4	2018	March	16			
5	2018	April	27			
6	2018	May	21			
7	2018	June	21			
8	2018	July	25			
9	2018	August	23			
10	2018	September	33			
11	2018	October	22			
12	2018	November	15			
13	2018	December	14			

On importe la table suivante en Power BI et on fait le "MERGE" des deux colonnes pour obtenir une colonne date et ensuite on change le type des données en date

Sheet1

date Sales

Saturday, January 1, 2011	10
Sunday, January 1, 2012	12
Tuesday, January 1, 2013	11
Wednesday, January 1, 2014	12
Thursday, January 1, 2015	11
Friday, January 1, 2016	13
Sunday, January 1, 2017	20
Tuesday, February 1, 2011	7
Wednesday, February 1, 2012	10
Friday, February 1, 2013	14
Saturday, February 1, 2014	13
Sunday, February 1, 2015	9
Monday, February 1, 2016	11
Wednesday, February 1, 2017	16
Tuesday, March 1, 2011	12
Thursday, March 1, 2012	16
Friday, March 1, 2013	13
Saturday, March 1, 2014	15
Sunday, March 1, 2015	20
Tuesday, March 1, 2016	22
Wednesday, March 1, 2017	27

Fields

- Search
- > sales by Year and Month
- New measure
- New column
- New quick measure
- Refresh data
- Edit query
- Manage relationship
- Incremental refresh
- Manage aggregations
- Copy Table
- Rename
- Delete from model
- Hide in report view
- Mark as date table
- Unhide all
- Collapse all
- Expand all

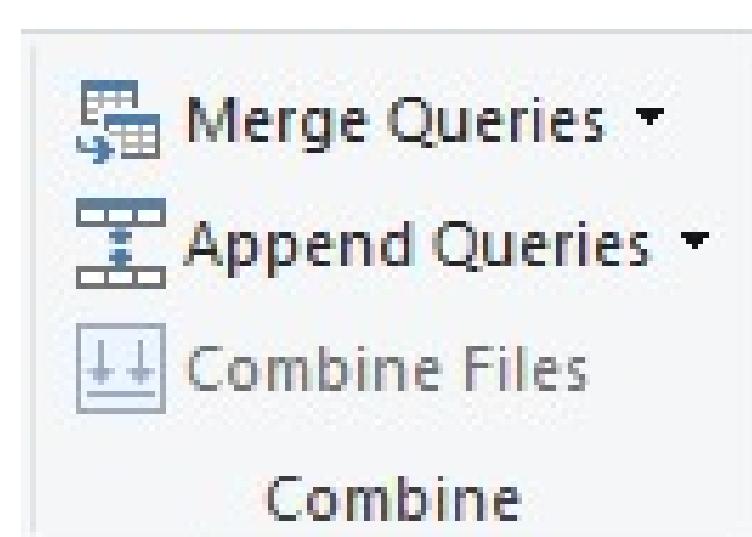
Sheet1 (5)

sales by Year and Month

Sheet1

	date	Sales
1	1/1/2018	35
2	2/1/2018	20
3	3/1/2018	16
4	4/1/2018	27
5	5/1/2018	21
6	6/1/2018	21
7	7/1/2018	25
8	8/1/2018	23
9	9/1/2018	33
10	10/1/2018	22
11	11/1/2018	15
12	12/1/2018	14
13	1/1/2018	35
14	2/1/2018	20

On a fait "append queries" afin de concaténer les données concernant les années précédentes qu'on a déjà préparé avec les données de 2018 qu'on a aussi préparé .



Et voilà une comparaison entre les données avant le forcast et après le forcast

	date	1.2 Sales
1	1/1/2011	10
2	1/1/2012	12
3	1/1/2013	11
4	1/1/2014	12
5	1/1/2015	11
6	1/1/2016	13
7	1/1/2017	20
8	2/1/2011	7
9	2/1/2012	10
10	2/1/2013	14
11	2/1/2014	13
12	2/1/2015	9
13	2/1/2016	11
14	2/1/2017	16
15	3/1/2011	12
16	3/1/2012	16
17	3/1/2013	13
18	3/1/2014	15
19	3/1/2015	20
20	3/1/2016	22
21	3/1/2017	27
22	4/1/2011	10
23	4/1/2012	10
24	4/1/2013	11

	date	1.2 Sales
76	11/1/2016	9
77	11/1/2017	14
78	12/1/2011	25
79	12/1/2012	21
80	12/1/2013	19
81	12/1/2014	17
82	12/1/2015	13
83	12/1/2016	11
84	12/1/2017	35
85	1/1/2018	35
86	2/1/2018	20
87	3/1/2018	16
88	4/1/2018	27
89	5/1/2018	21
90	6/1/2018	21
91	7/1/2018	25
92	8/1/2018	23
93	9/1/2018	33
94	10/1/2018	22
95	11/1/2018	15
96	12/1/2018	14
97	1/1/2018	35
98	2/1/2018	20
99	3/1/2018	16

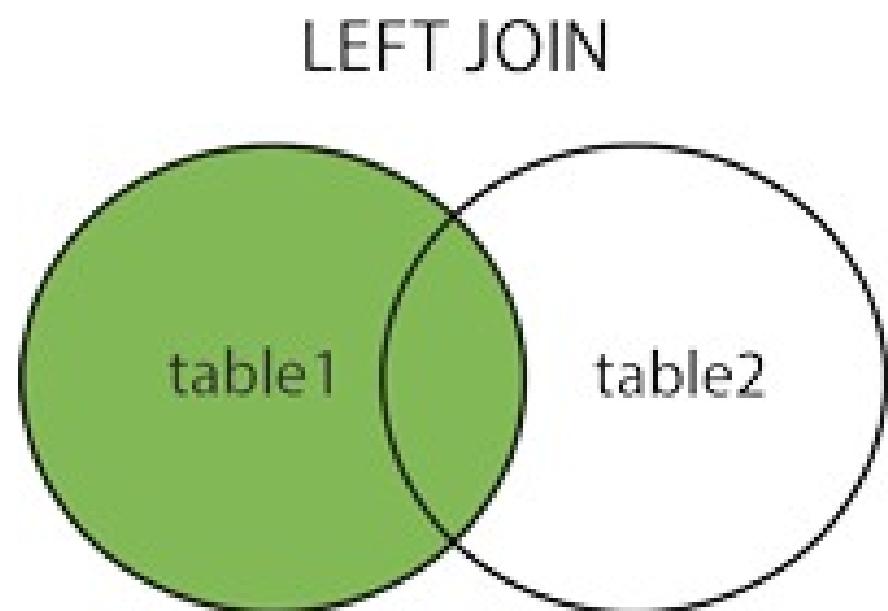
# Exercice 2:



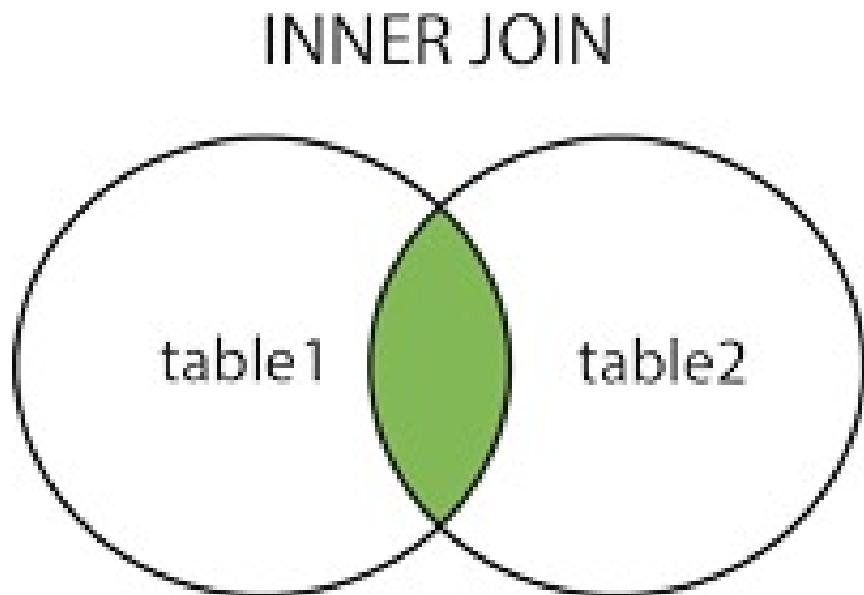
## Tâche 1:

Define the differences between an inner join, a left join and a full outer join.

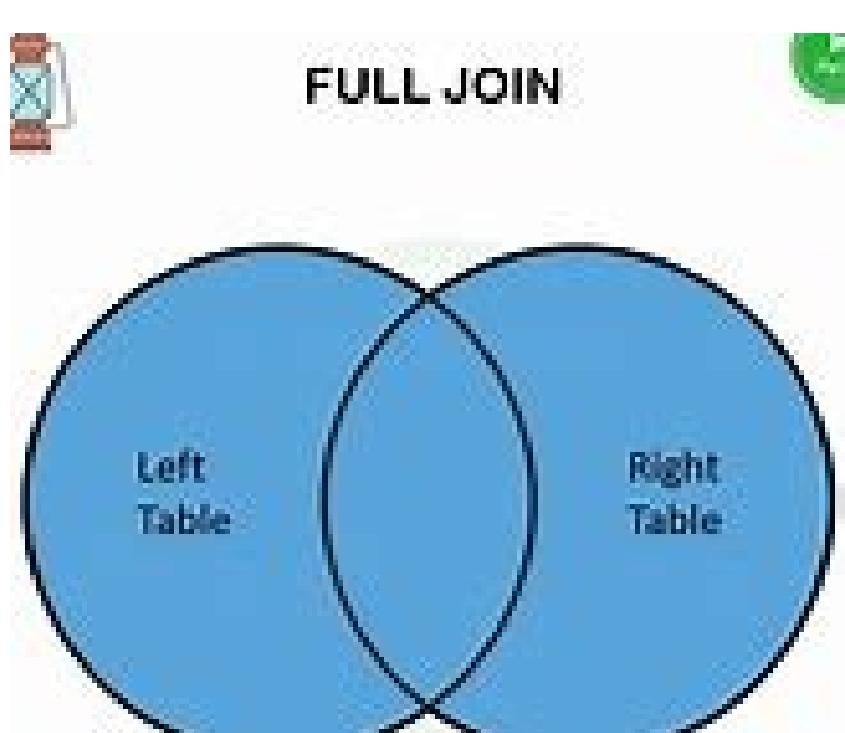
**Left join** renvoie toutes les lignes de la table de gauche et les lignes correspondantes de la table de droite. En cas d'absence de correspondance avec la table de droite, il renverra la valeur NULL.



Dans **Inner join** , chaque enregistrement de la table 1 est mis en correspondance avec chaque enregistrement de la table 2 et ceux correspondants sont ensuite affichés dans la table résultante. Il s'agit donc d'une intersection des données entre la table 1 et la table 2.



**Full outer join** renvoie toutes les lignes de toutes les tables. Peu importe s'il y a une correspondance ou non



## Tâche 2:

What is the difference between UNION vs UNION ALL?

UNION et UNION ALL sont des opérateurs SQL utilisés pour concaténer deux ensembles de résultats ou plus. Cela permet d'écrire plusieurs instructions SELECT, de récupérer les résultats souhaités et de les combiner dans l'ensemble consolidé final.

Les principales différences entre UNION et UNION ALL sont les suivantes :

**UNION** : conserve uniquement des enregistrements uniques

**UNION ALL** : conserve tous les enregistrements, y compris les doublons

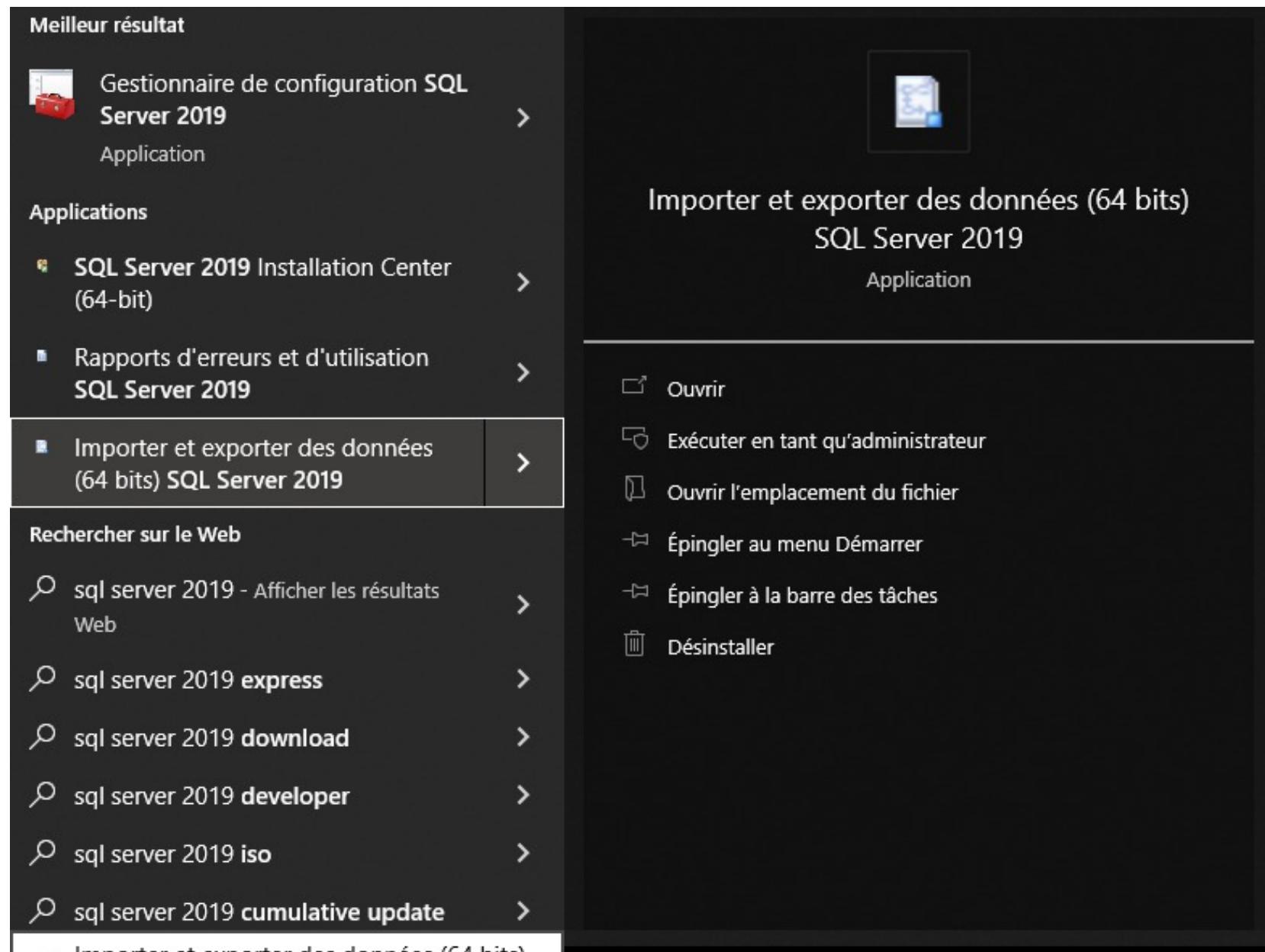
## Tâche 3:

Find all subscribers who activated before 2017.

Premièrement on importe les données dans un fichier Excel

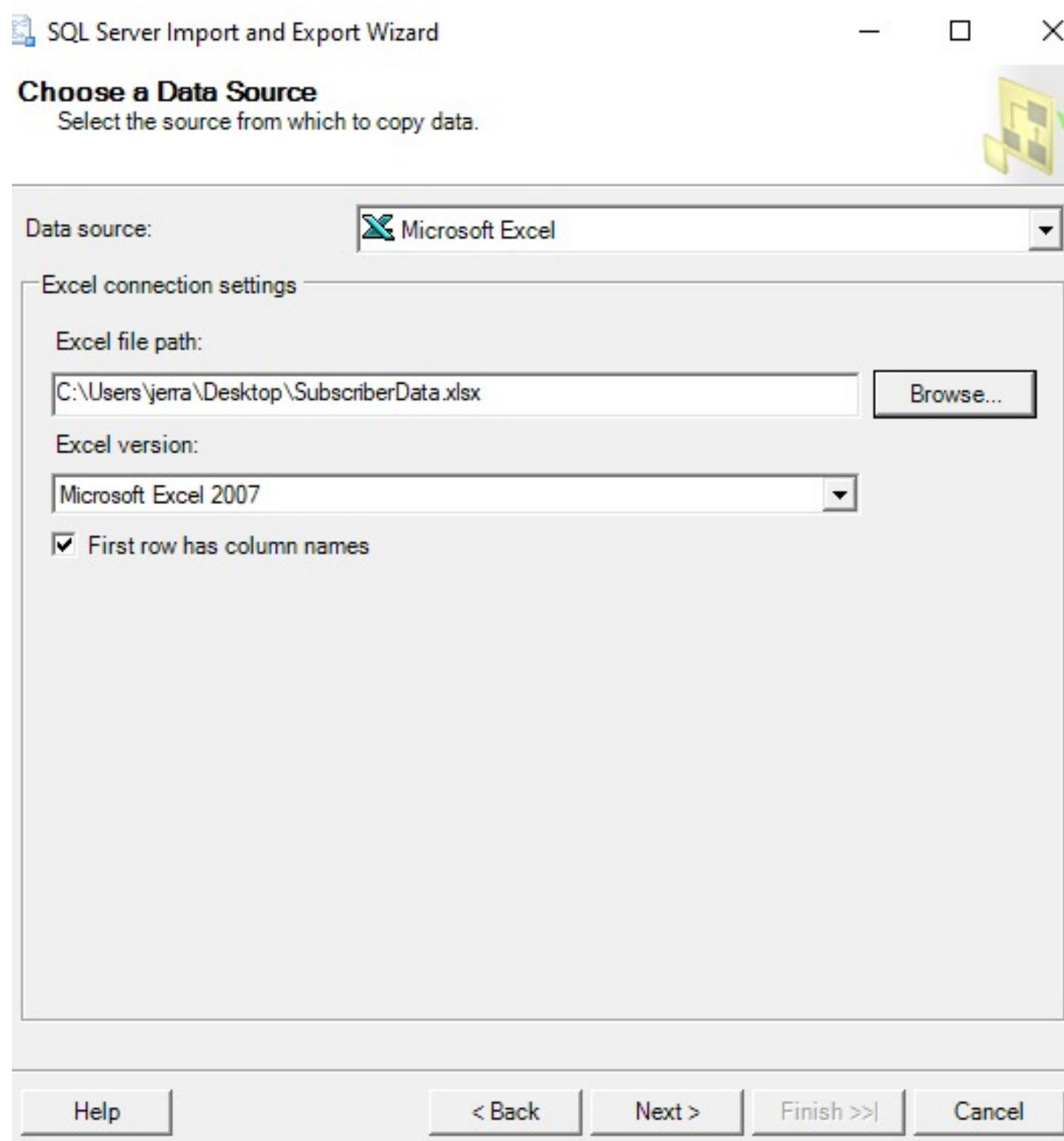
	A	B	C	D	E
1	SubscriberNumber	FirstName	LastName	Province	Activation
2	64	Vance	Rose	British Columbia	05/12/2017
3	39	Dekel	Addison	Quebec	18/09/2017
4	47	Jeffry	Joyner	Ontario	03/05/2016
5	21	Meriwether	Durand	P.E.I	24/06/2017
6	66	Irwin	Dawson	Alberta	13/11/2016
7	30	Cydney	Harrelson	Ontario	16/01/2017
8	84	Laima	Virgo	Nova Scotia	09/08/2016
9	17	Yvonne	Howard	Alberta	09/04/2016
10	42	Petronel	Badcocke	British Columbia	29/03/2017
11	23	Burgundy	Kendal	Quebec	03/07/2017
12	30	Cydney	Harrelson	Ontario	16/01/2017
13	52	Aniyah	Brownlow	Ontario	22/10/2017

Et ensuite on importe les données dans SQL server en utilisant la fonction **importer et exporter des données "SQL Server 2019"**

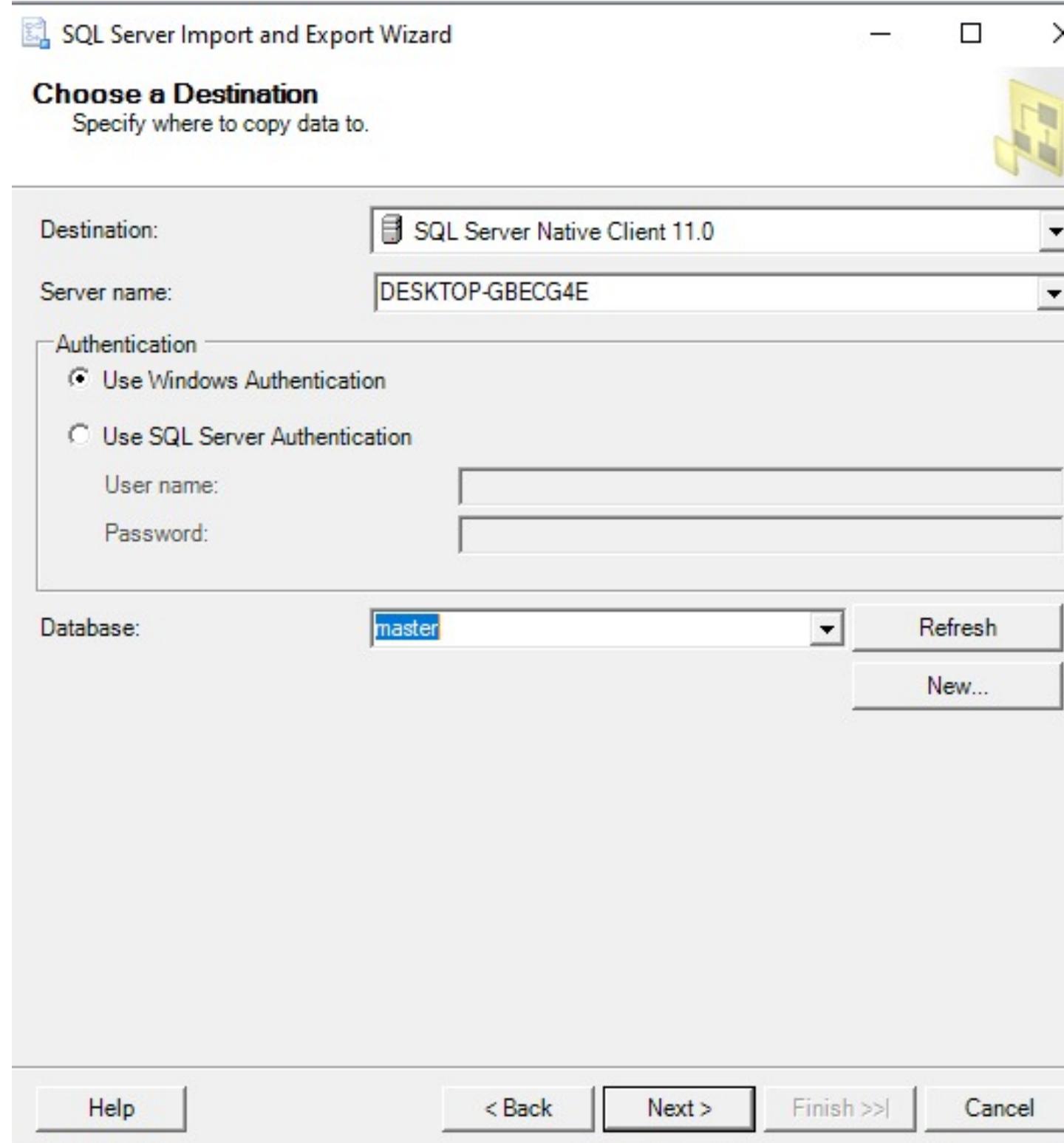


## Tâche 3:

On précise le type de la source des données

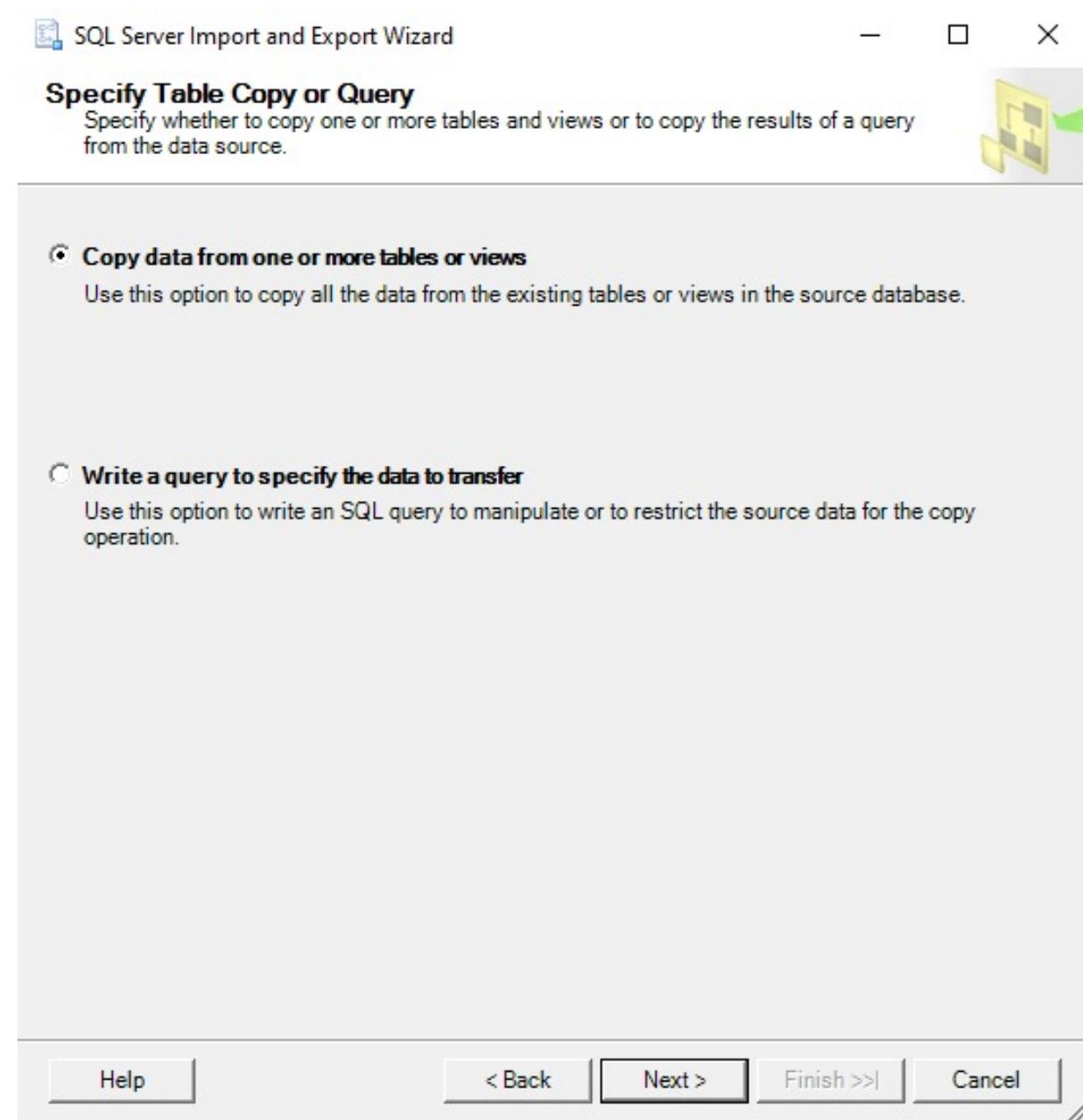


On précise aussi la destination de ces données : SQL Server Native client

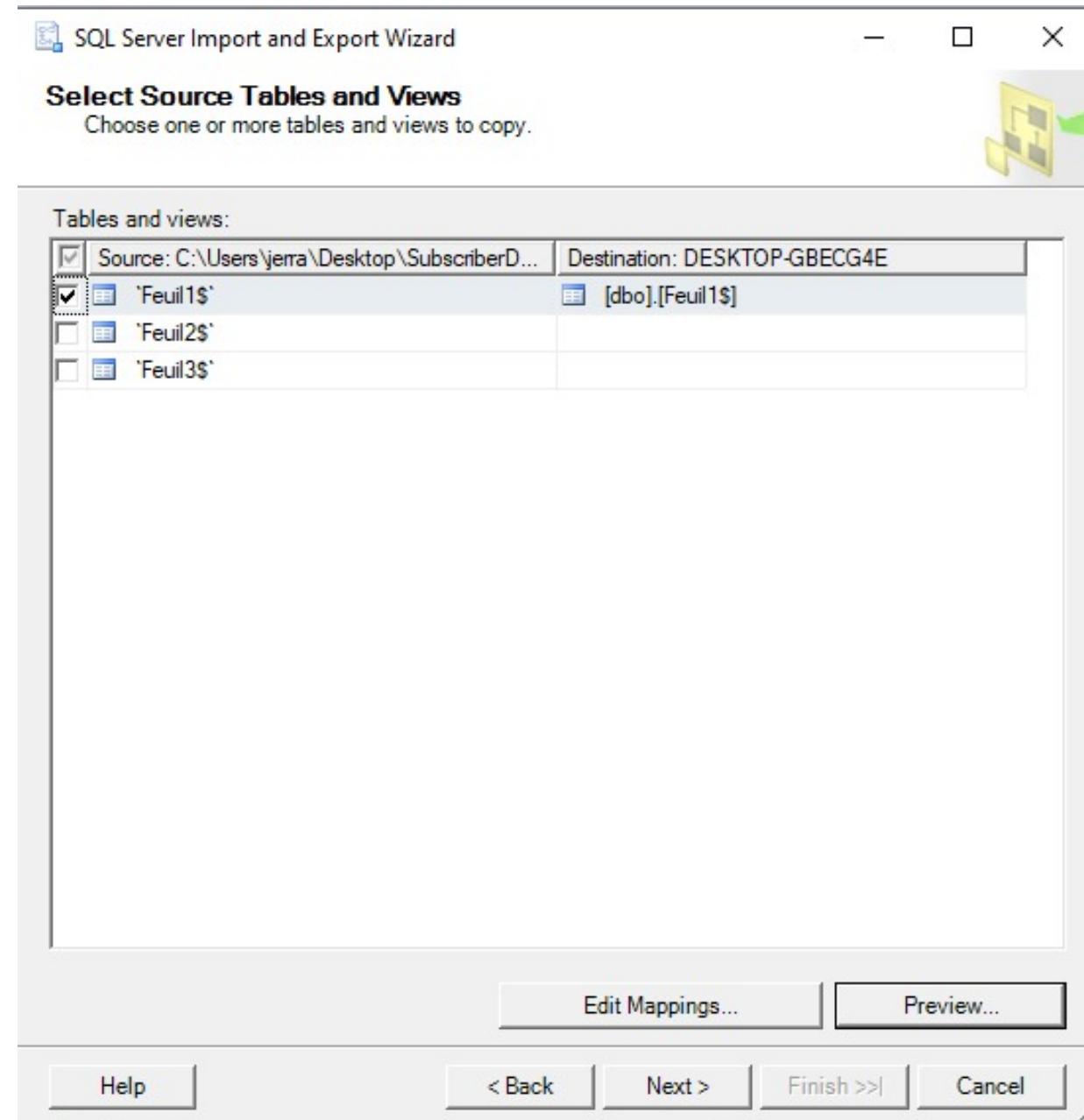


## Tâche 3:

On coche la première option  
c'est de copier les données  
de la source



On prend la feuille qui est  
remplis de la source:



## Tâche 3:

Là on s'assure que la table de destination comporte le même nombre de colonnes que celles de source de données:

Column Mappings

Source: 'Feuil1\$'

Destination: [dbo].[Feuil1\$]

Create destination table

Delete rows in destination table  Drop and re-create destination table

Append rows to the destination table  Enable identity insert

Mappings:

Source	Destination	Type	Nullable	Size	Precision	Scale
SubscriberNum...	SubscriberNum...	float	<input checked="" type="checkbox"/>			
First Name	FirstName	nvarchar	<input checked="" type="checkbox"/>	255		
Last Name	LastName	nvarchar	<input checked="" type="checkbox"/>	255		
Province	Province	nvarchar	<input checked="" type="checkbox"/>	255		
Activation	Activation	datetime	<input checked="" type="checkbox"/>			

Source column: SubscriberNumber Double (15)

SQL Server Import and Export Wizard

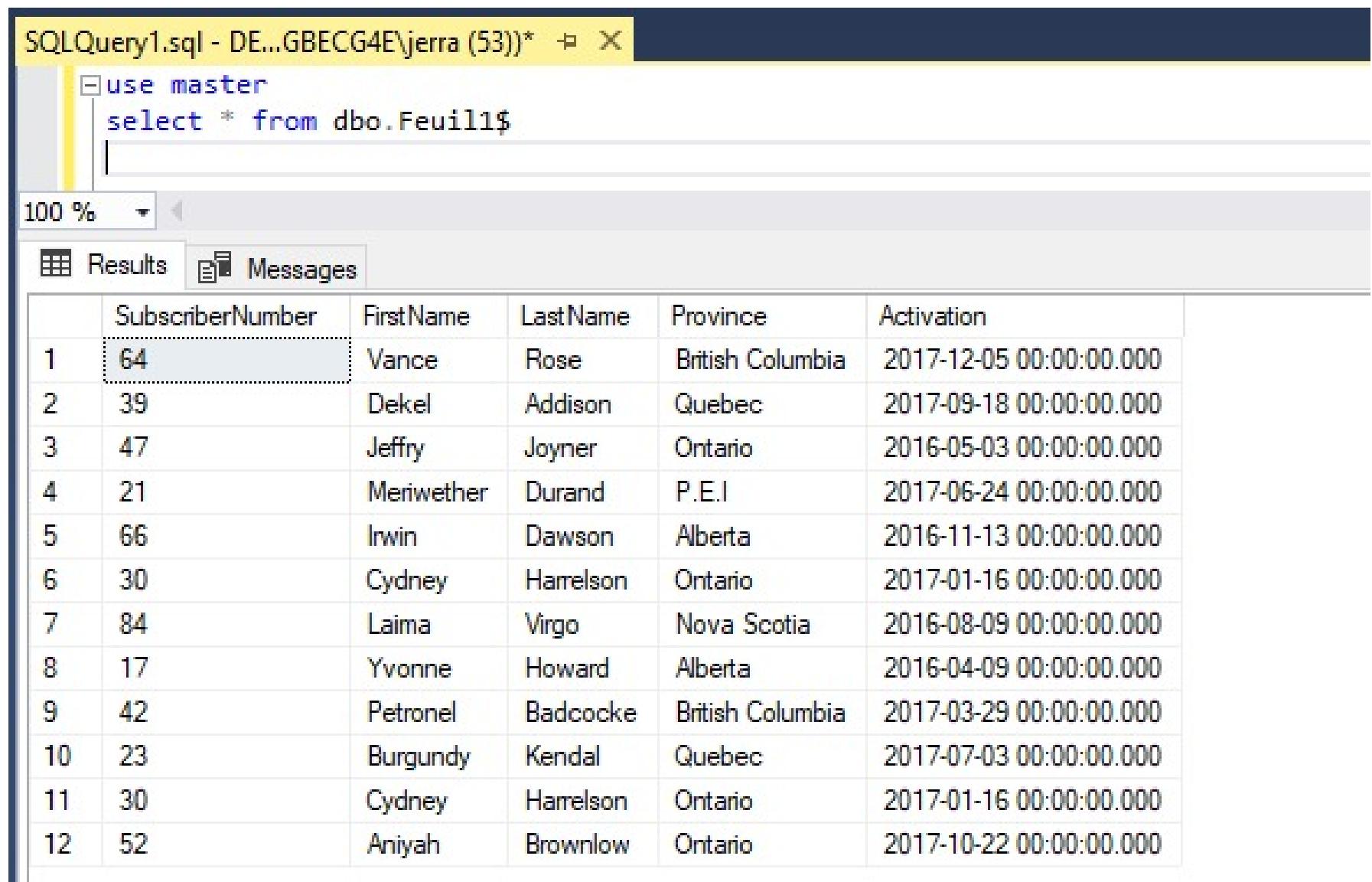
Run Package

Run immediately

In SQL Server Express, Web, or Workgroup, you can run the package that the Import and Export Wizard creates, but cannot save it. To save packages that the wizard creates, you must upgrade to SQL Server Standard, Enterprise, Developer or Evaluation.

## Tâche 3:

Et voilà on a bien importé les données de Subscribers dans SQL server:



SQLQuery1.sql - DE...GBECG4E\jerra (53)\*

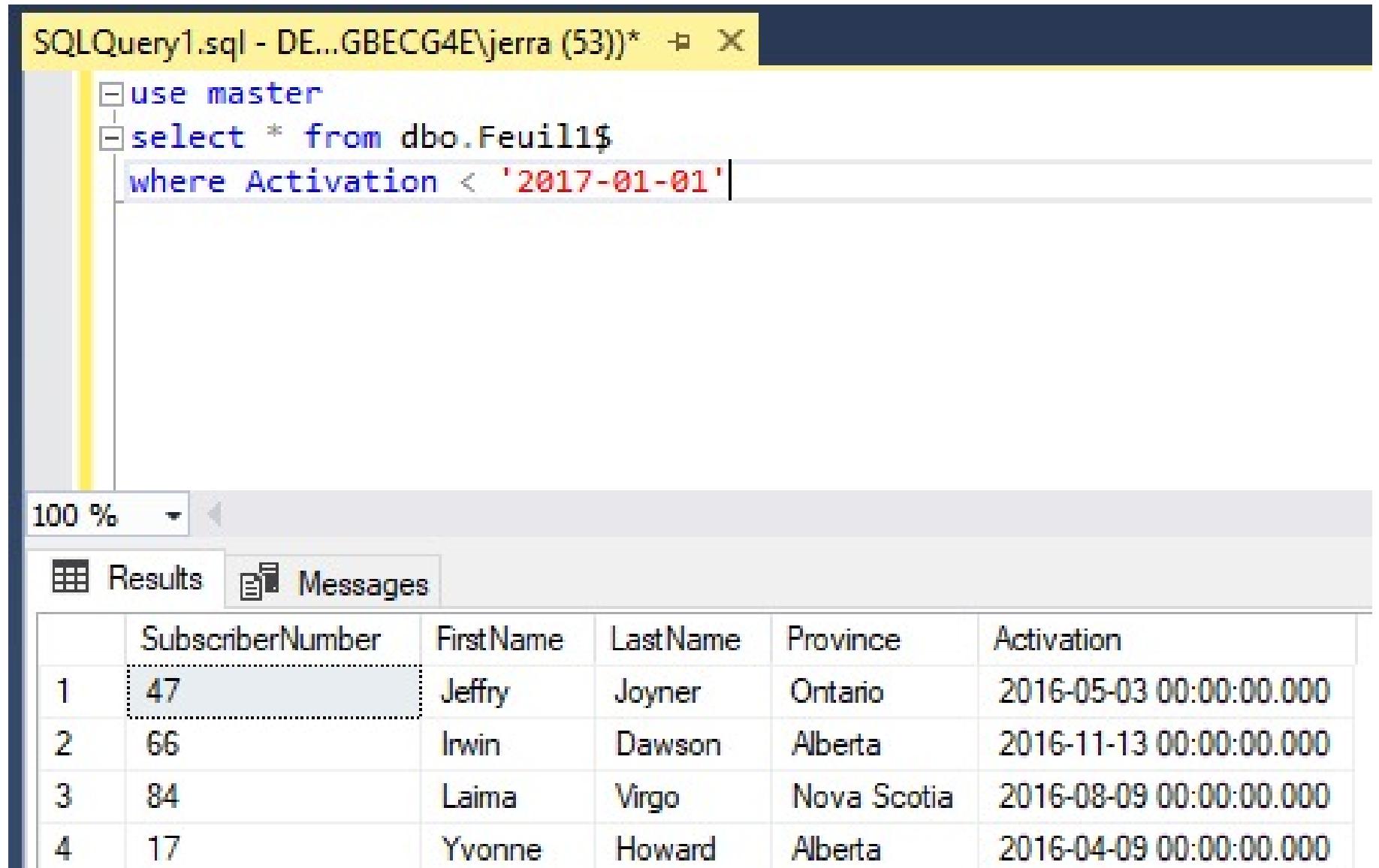
```
use master
select * from dbo.Feuill1$
```

100 %

Results

	SubscriberNumber	First Name	Last Name	Province	Activation
1	64	Vance	Rose	British Columbia	2017-12-05 00:00:00.000
2	39	Dekel	Addison	Quebec	2017-09-18 00:00:00.000
3	47	Jeffry	Joyner	Ontario	2016-05-03 00:00:00.000
4	21	Meriwether	Durand	P.E.I	2017-06-24 00:00:00.000
5	66	Irwin	Dawson	Alberta	2016-11-13 00:00:00.000
6	30	Cydney	Harrelson	Ontario	2017-01-16 00:00:00.000
7	84	Laima	Virgo	Nova Scotia	2016-08-09 00:00:00.000
8	17	Yvonne	Howard	Alberta	2016-04-09 00:00:00.000
9	42	Petronel	Badcocke	British Columbia	2017-03-29 00:00:00.000
10	23	Burgundy	Kendal	Quebec	2017-07-03 00:00:00.000
11	30	Cydney	Harrelson	Ontario	2017-01-16 00:00:00.000
12	52	Aniyah	Brownlow	Ontario	2017-10-22 00:00:00.000

Et afin de trouver les participants qui ont activé leurs comptes avant 2017 on utilise la fonction where activation est moins de 2017 et on obtient la table souhaitée:



SQLQuery1.sql - DE...GBECG4E\jerra (53)\*

```
use master
select * from dbo.Feuill1$
where Activation < '2017-01-01'
```

100 %

Results

	SubscriberNumber	First Name	Last Name	Province	Activation
1	47	Jeffry	Joyner	Ontario	2016-05-03 00:00:00.000
2	66	Irwin	Dawson	Alberta	2016-11-13 00:00:00.000
3	84	Laima	Virgo	Nova Scotia	2016-08-09 00:00:00.000
4	17	Yvonne	Howard	Alberta	2016-04-09 00:00:00.000

## Tâche 4:

Un insert a été accidentellement exécuté deux fois ! Écrivez une requête pour trouver tous les enregistrements en double dans SubscriberData. Qu'est-ce qui pourrait être changé dans la définition de table pour se protéger contre l'insertion de doublons ?

Voilà la requête permettant de trouver les enregistrements en double .

Premièrement on calcule le nombre des subscribers, on a sur la table 12 et en utilisant la fonction **DISTINCT** on obtient que les participants uniques sont 11 donc on a une inscription dupliquée. Afin d'obtenir l'inscription dupliqué on a groupé les données par "SubscriberNumber" et on affiche ceux qui apparaît plus qu'une fois en utilisant la clause '**Having**'.

```
SQLQuery1.sql - DE...GBECG4E\jerra (53)* ➔ X
use master
SELECT COUNT(SubscriberNumber) AS total_records FROM dbo.Feuill1$;
SELECT DISTINCT(SubscriberNumber) FROM dbo.Feuill1$;
SELECT SubscriberNumber,COUNT(*)
FROM dbo.Feuill1$
GROUP BY SubscriberNumber
HAVING COUNT(*) > 1
```

100 %

Results Messages

	total_records
1	12

	SubscriberNumber
1	17
2	21
3	23
4	30
5	39
6	42
7	47
8	52
9	64
10	66
11	84

	SubscriberNumber	(No column name)
1	30	2

## Tâche 4:

Il existe deux méthodes qu'on peut utiliser lors de la définition de table pour se protéger contre l'insertion de doublons.

- Définissez la propriété Indexé du champ sur Oui (pas de doublons):  
on peut le faire en ouvrant la table en mode Création. Cette méthode est simple et constitue un bon choix si on ne souhaite modifier qu'un seul champ à la fois.
- Créer une requête de définition de données qui crée l'index unique.  
on peut le faire en utilisant la vue SQL. Cette méthode n'est pas aussi simple que d'utiliser le mode Création, mais présente un avantage : on peut enregistrer la requête de définition de données et la réutiliser ultérieurement. Ceci est utile si on supprime et recrée périodiquement des tables et souhaite utiliser des index uniques sur certains champs.

voilà la requête qu'on insert dans le code de la création de la base de données, et qui nous permet de créer un indice unique, pour cette base de données l'indice est "SubscriberNumber".

```
CREATE UNIQUE INDEX [IX_SubscriberNumber]
ON [SubscriberData] (SubscriberNumber)
```

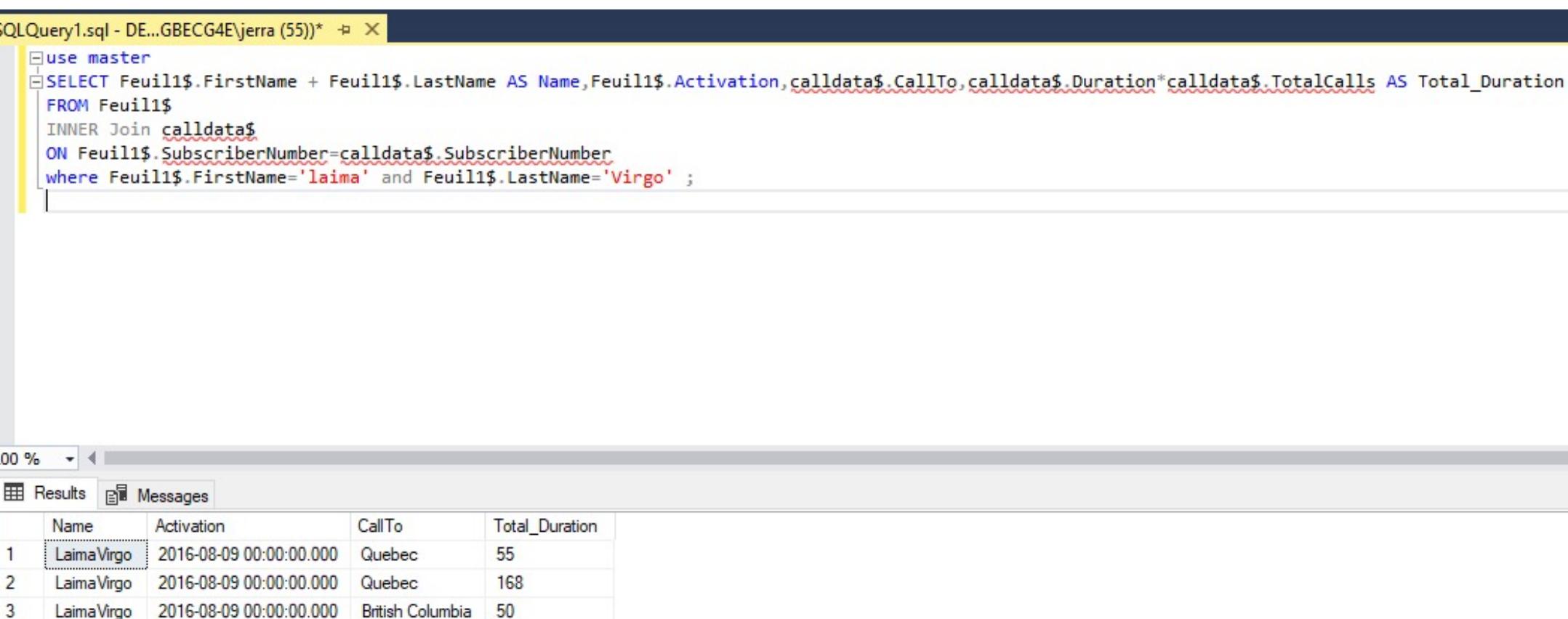
## Tâche 5:

Rédigez une requête pour signaler le nom, 'CallTo', la date d'activation et la durée totale des appels pour Laima Virgo.

Afin d'obtenir ces colonnes, on a utiliser un "INNER JOIN" entre la feuille1 de subscriberData et CallData.

Premièrement on concaténe FirstName et LastName comme Name et on a obtenue Activation de la feuil1, CallTo de CallData et Total\_Duration en faisant le produit de la duration et TotalCalls.

Pour obtenir les informations de laima Virgo on utilise la fonction Where



SQLQuery1.sql - DE...GBECG4E\jerra (55)\*

```
use master
SELECT Feuill1$.FirstName + Feuill1$.LastName AS Name, Feuill1$.Activation, calldata$.CallTo, calldata$.Duration*calldata$.TotalCalls AS Total_Duration
FROM Feuill1$
INNER Join calldata$ 
ON Feuill1$.SubscriberNumber=calldata$.SubscriberNumber
where Feuill1$.FirstName='laima' and Feuill1$.LastName='Virgo' ;
```

00 %

Results Messages

	Name	Activation	CallTo	Total_Duration
1	LaimaVirgo	2016-08-09 00:00:00.000	Quebec	55
2	LaimaVirgo	2016-08-09 00:00:00.000	Quebec	168
3	LaimaVirgo	2016-08-09 00:00:00.000	British Columbia	50

# Exercice 3:

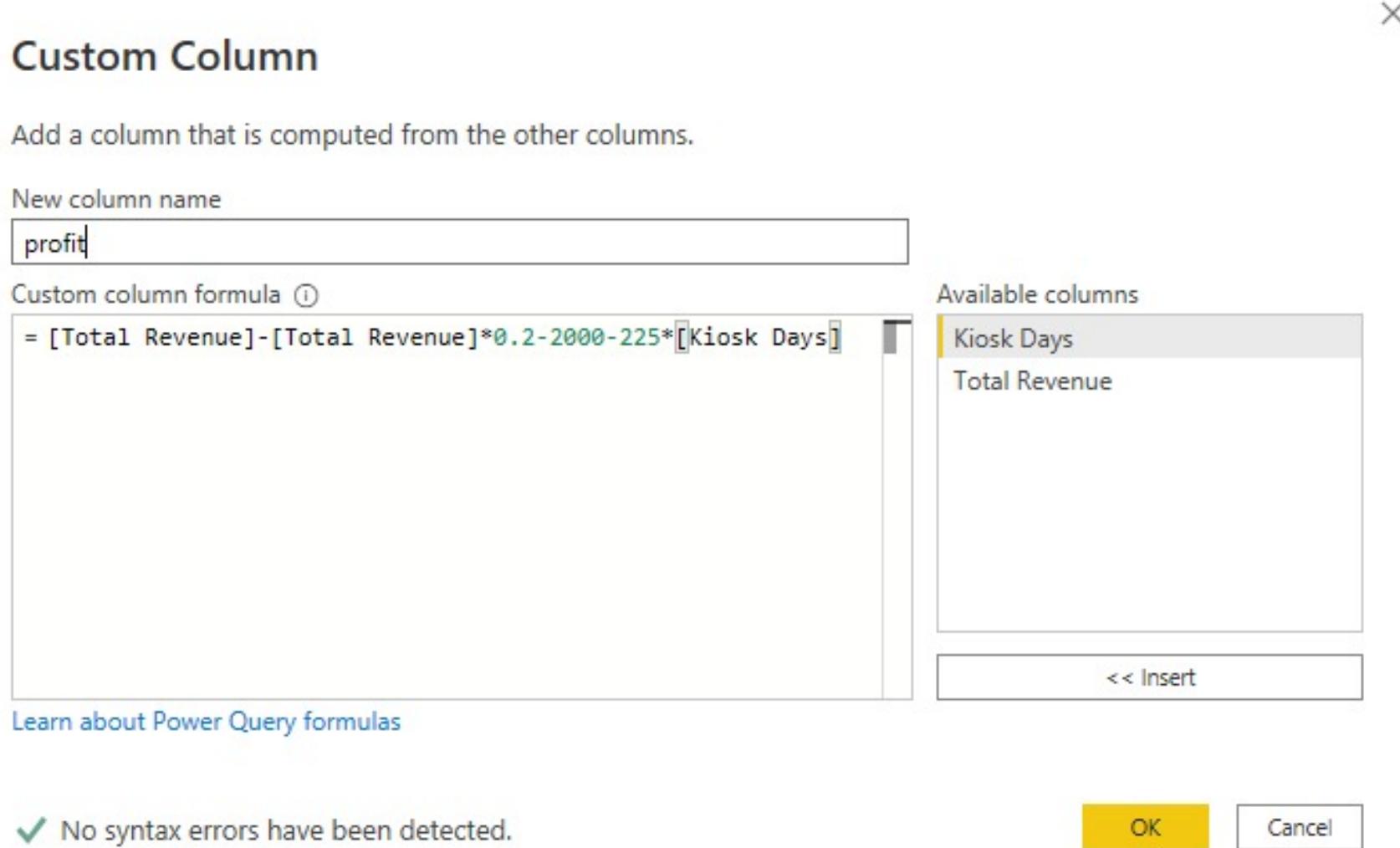


Dans cette exercice on doit déterminer le nombre optimal des jours qu'un groupe d'agents doivent s'installer dans une location avant de se déplacer à une autre place.

On a comme données l'historique du revenu total selon le nombre des jours avant le déplacement qui seront utiliser pour calculer le bénéfice de chaque "Kiosque days".

on calculera le bénéfice en faisant : [le revenue total] - [cout de déplacement = 2000\$] - [coût fixe = 225\$]\*[nombre de jours] - [coûts variables = revenue total\*20%]

Pour cela on crée une nouvelle colonne dans Power BI nommée "Profit"



Voilà la nouvelle colonne qu'on a créé:

	Kiosk Days	Total Revenue	profit
1	1	5050	1815
2	2	5150	1670
3	3	5300	1565
4	4	5500	1500
5	5	5750	1475
6	6	6050	1490
7	7	6400	1545
8	8	6800	1640
9	9	7250	1775
10	10	7750	1950
11	11	8100	2005
12	12	8350	1980
13	13	8500	1875
14	14	8600	1730
15	15	8650	1545

Là on essaye de mettre un graphe qui va nous permettre de visualiser le nb de jours optimale où l'équipe peut générer plus de profit. Pour cela on utilise la colonne "Kiosk Days" et la colonne "profit":

Visualizations >> Fields

Build visual

Search

Sheet1

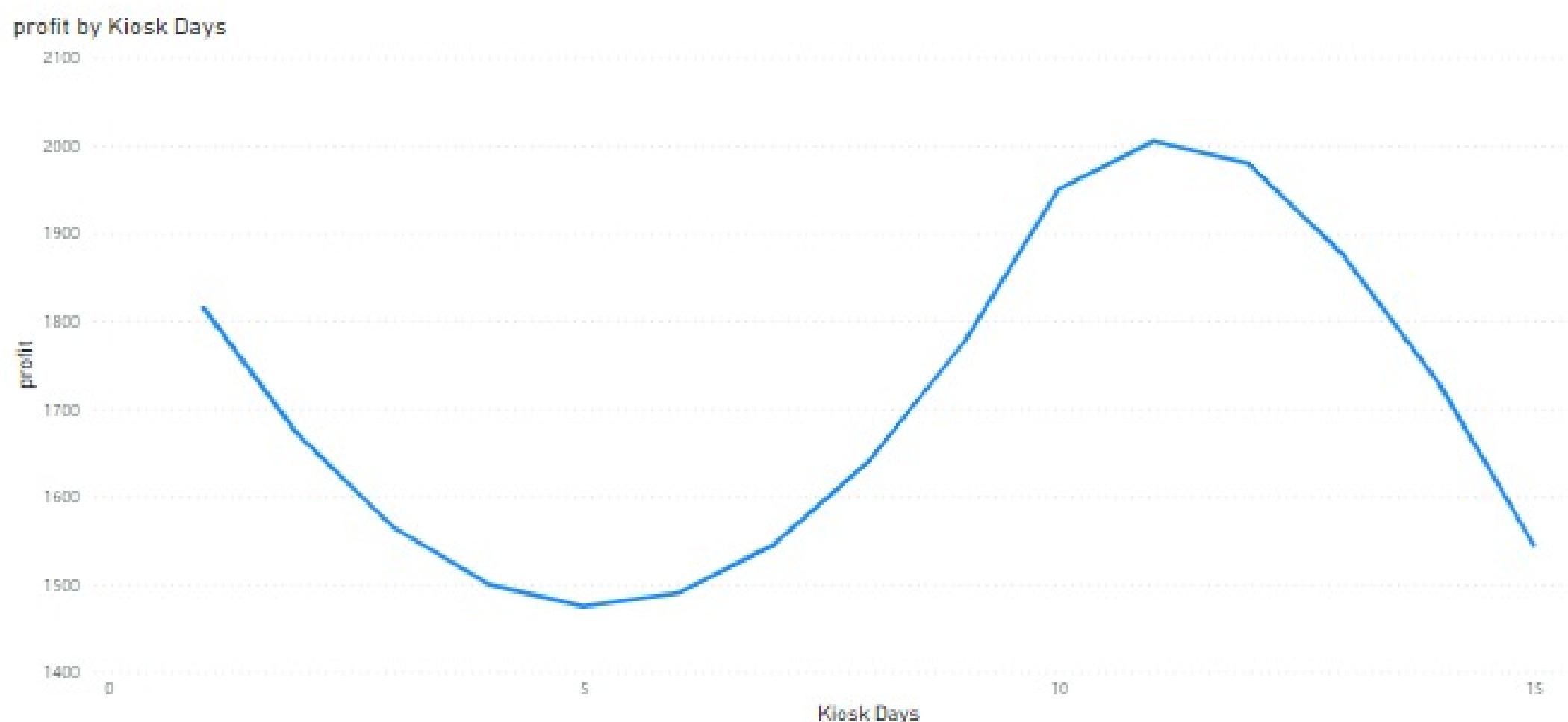
- $\sum$  Kiosk Days
- $\sum$  profit
- $\sum$  Total Revenue

Kiosk Days

Y-axis

profit

Voilà le résultat où il est clair que le nombre optimal de jours est :11 jours

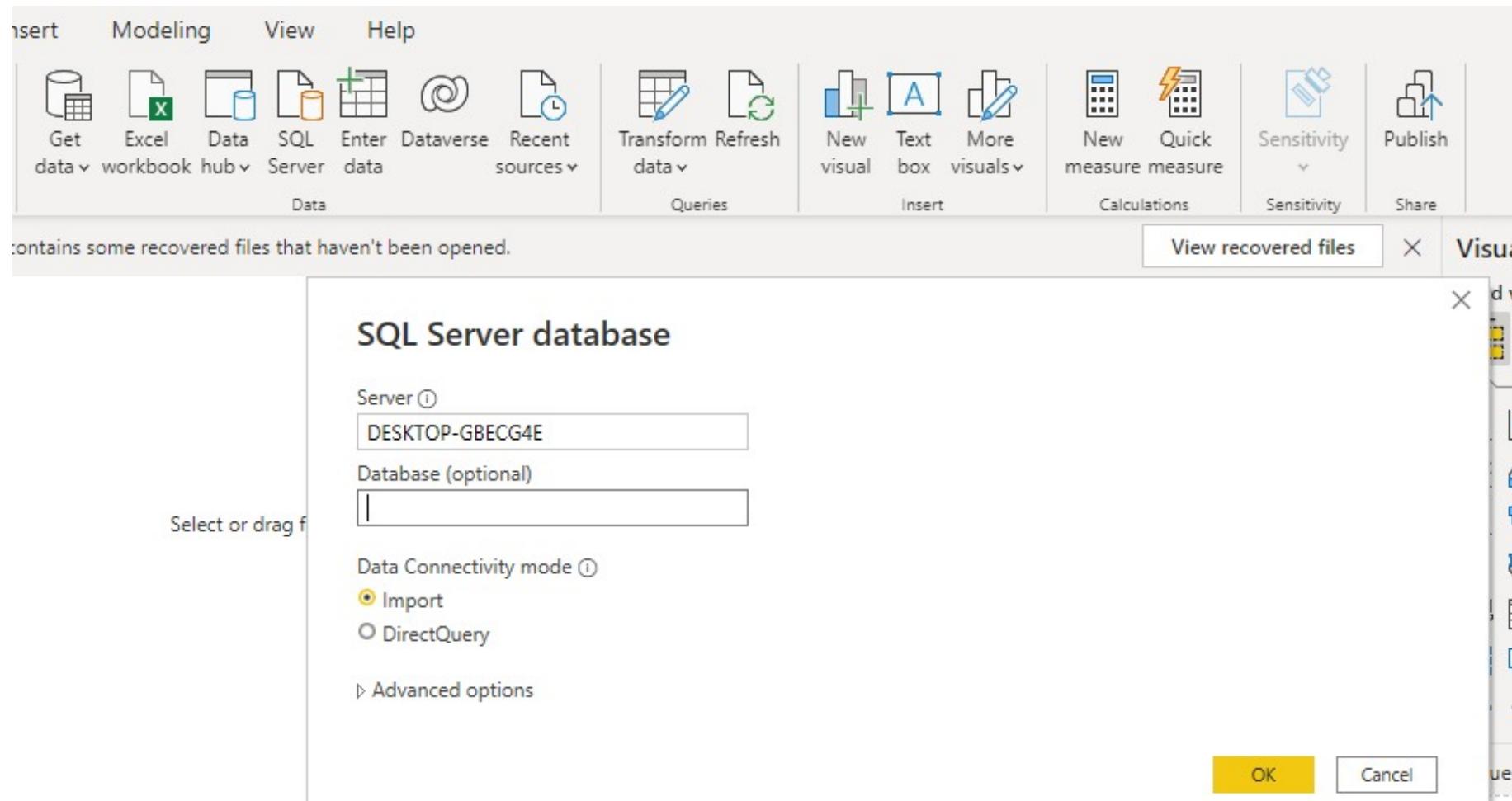


# Power BI sur Northwind

## Tâche 1:

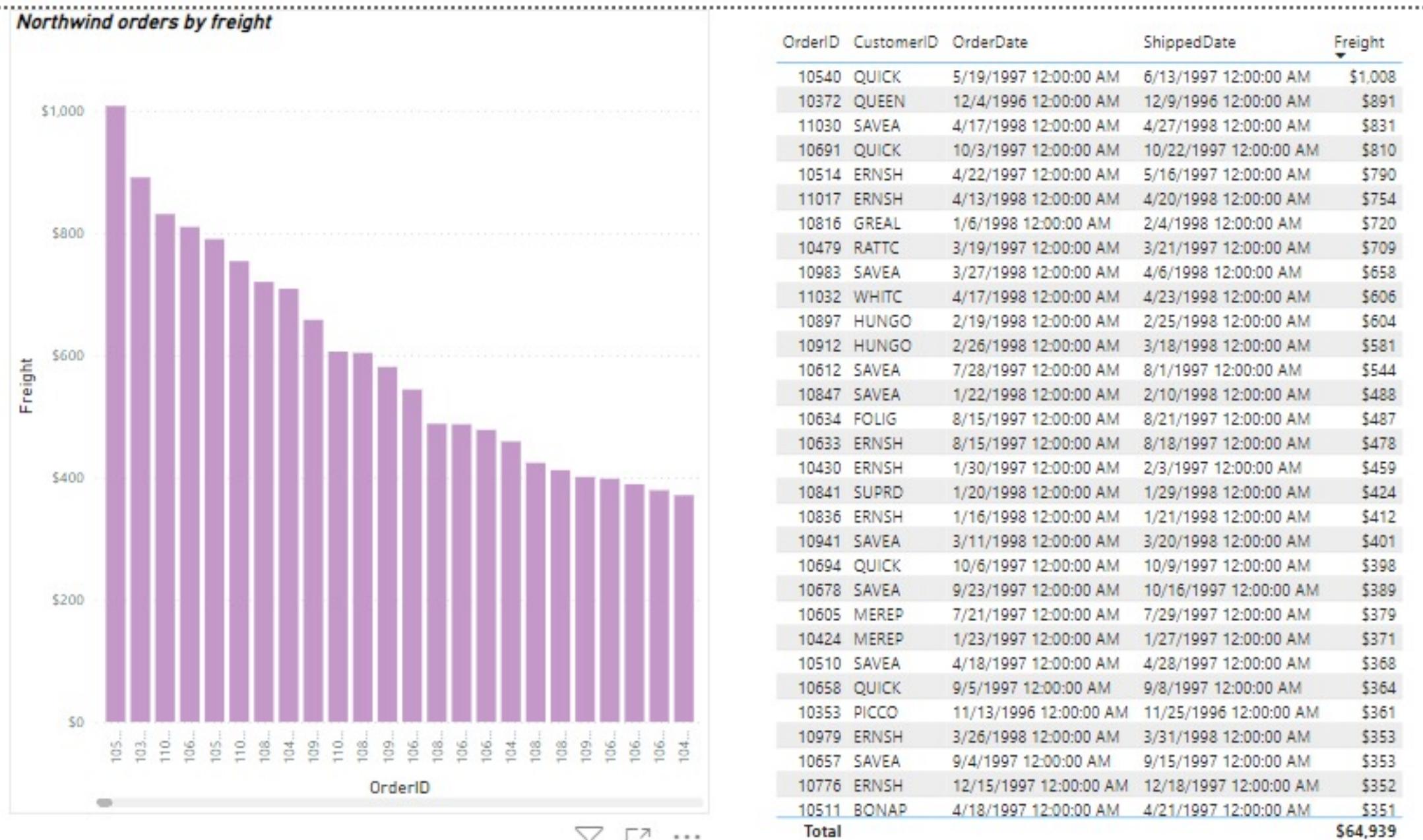
Créez un visuel montrant orders de Northwind triées par Freight du plus cher au moins cher. Afficher OrderID, OrderDate, ShippedDate, CustomerID et Freight.

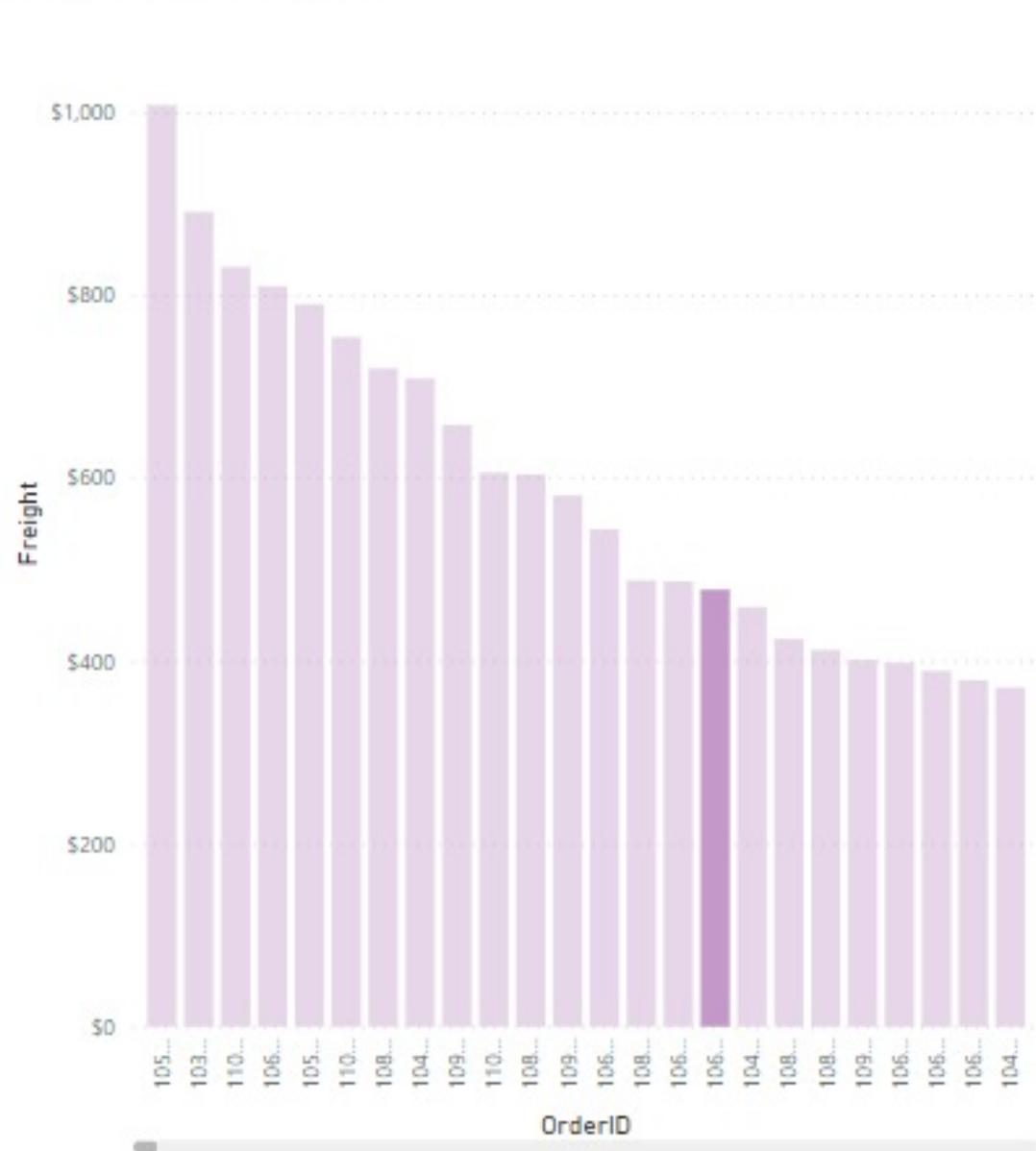
Premièrement on importe nos données: à partir de sql server à Power BI:



Premièrement on a créé un graphe pour montrer orders de Northwind triées par Freight

Ensuite on visualise les colonnes demandés dans la tâche 1:





## Tâche 2:

Apportez un "slicer" indiquant le titre et les noms et prénoms de tous les commerciaux. Ici afin de créer un slicer de les titres, les noms et les prénoms des "Sales Representative" on fait premièrement un merge entre last et first name:

Queries [11] < X ✓ fx = Table.CombineColumns(dbo\_Employees, {"LastName", "FirstName"}, Combiner.CombineTextByDelimiter(" "),

	EmployeeID	first and last name	Title	TitleOfCourtesy	BirthDate	HireDate
1	1	Davolio Nancy	Sales Representative	Ms.	12/8/1948 12:00:00 AM	5/1/1992
2	2	Fuller Andrew	Vice President, Sales	Dr.	2/19/1952 12:00:00 AM	8/14/1992
3	3	Leverling Janet	Sales Representative	Ms.	8/30/1963 12:00:00 AM	4/1/1992
4	4	Peacock Margaret	Sales Representative	Mrs.	9/19/1937 12:00:00 AM	5/3/1992
5	5	Buchanan Steven	Sales Manager	Mr.	3/4/1955 12:00:00 AM	10/17/1992
6	6	Suyama Michael	Sales Representative	Mr.	7/2/1963 12:00:00 AM	10/17/1992
7	7	King Robert	Sales Representative	Mr.	5/29/1960 12:00:00 AM	1/2/1992
8	8	Callahan Laura	Inside Sales Coordinator	Ms.	1/9/1958 12:00:00 AM	3/5/1992
9	9	Dodsworth Anne	Sales Representative	Ms.	1/27/1966 12:00:00 AM	11/15/1992

Pour le slicer on choisit les colonnes "first and last name" et "Title"

## Fields

Search

- > Categories
- > Customers
- ✓ Employees
  - Address
  - >  BirthDate
    - City
    - Country
    - EmployeeID
    - Extension
    - first and last name
  - >  HireDate
    - HomePhone
    - Notes
    - PhotoPath
    - PostalCode
    - Region
    - $\sum$  ReportsTo

Ensuite on selecte les "Sales Representative" dans le slicer

Filters on this visual		...
first and last name is (All)		
<b>Title</b>		
is Sales Representative		
Filter type ⓘ		
Basic filtering		▼
<input type="text"/> Search		
<input checked="" type="checkbox"/> Select all		
<input type="checkbox"/> (Blank)		
<input type="checkbox"/> Inside Sales Coordinator 1		
<input type="checkbox"/> Sales Manager 1		
<input checked="" type="checkbox"/> Sales Representative 6		
<input type="checkbox"/> Vice President, Sales 1		
<input type="checkbox"/> Require single selection		

Et voilà le résultat:

Title, first and last name	
Sales Representative	
□	Davolio Nancy
□	Dodsworth Anne
□	King Robert
□	Leverling Janet
□	Peacock Margaret
□	Suyama Michael

### Tâche 3:

Si le coût du frais est supérieur ou égal à 500,00 \$, il sera désormais taxé de 10 %. Créez un rapport indiquant order id, freight cost, and freight cost avec cette taxe pour toutes les commandes de 500 \$ ou plus.

Afin d'obtenir la colonne de "freight\_taxes" on utilise le DAX de Power BI

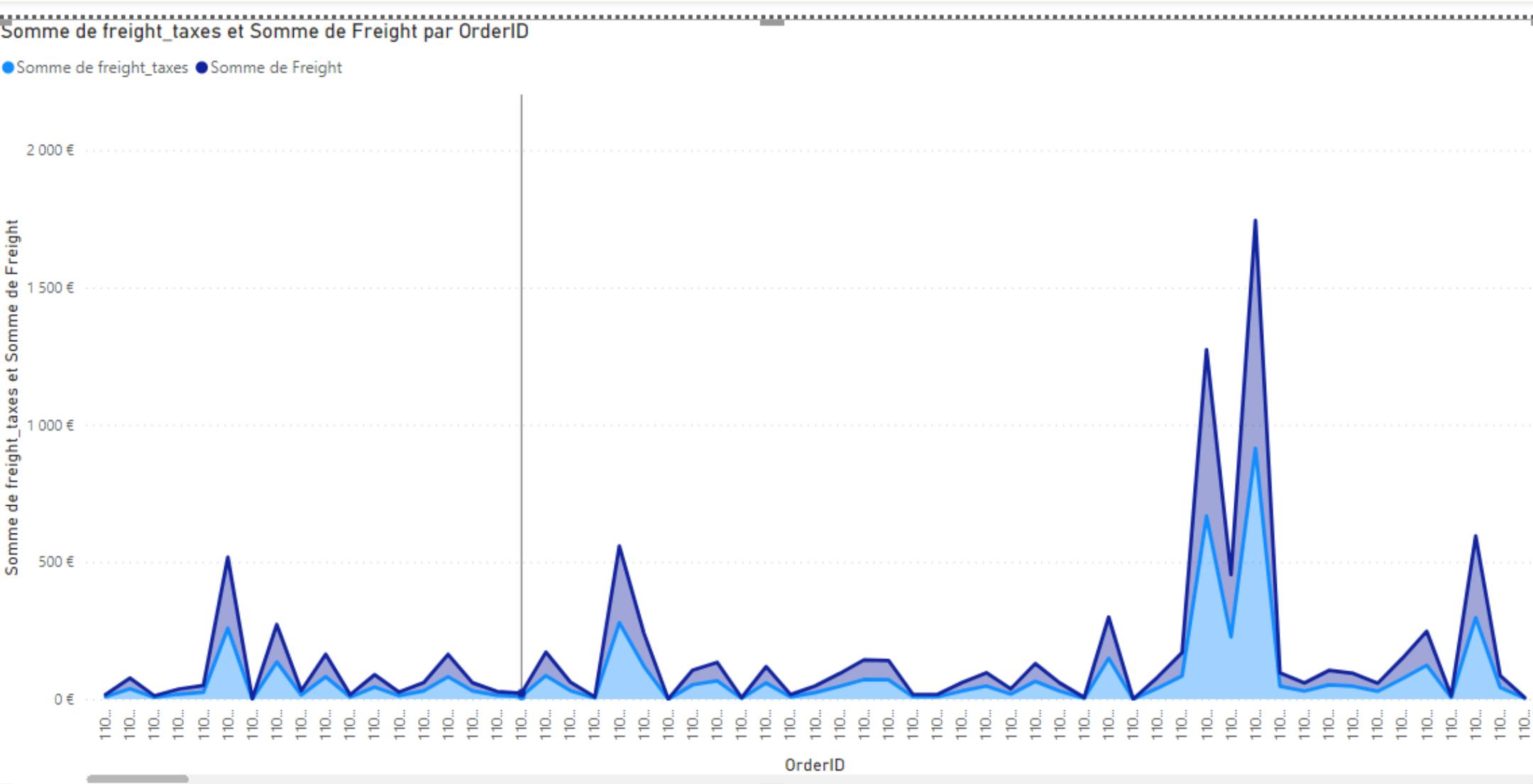
on a utilisé la fonction AddColumn en lui donnant le nom de la table utilisé avec le nom de la colonne qu'on veut créer et en utilisant each-then pour préciser la condition de l'ajout de taxe :

```
= Table.AddColumn(dbo_Orders, "freight_taxes", each if [Freight] >= 500 then [Freight]+[Freight]*0.1 else [Freight])
```

Voilà la nouvelle colonne freight\_taxes:

	OrderID	\$ Freight	ABC freight_taxes
1	10248	32,38	32,38
2	10249	11,61	11,61
3	10250	65,83	65,83
4	10251	41,34	41,34
5	10252	51,30	51,3
6	10253	58,17	58,17
7	10254	22,98	22,98
8	10255	148,33	148,33
9	10256	13,97	13,97
10	10257	81,91	81,91
11	10258	140,51	140,51
12	10259	3,25	3,25
13	10260	55,09	55,09
14	10261	3,05	3,05
15	10262	48,29	48,29
16	10263	146,06	146,06
17	10264	3,67	3,67
18	10265	55,28	55,28
19	10266	25,73	25,73
20	10267	208,58	208,58

Ensuite on a visualisé freight sans taxes et celui de freight avec taxes tout en montrant la différence entre les deux freights:



## Tâche 4:

Trouver le nombre total d'unités commandées du produit ID 3 13. Récupérer le nombre d'employés dans chaque ville. Utiliser des visuels pour représenter l'histoire.

Premièrement on choisit le visuel "Multi\_row\_card" et on sélectionne les colonnes ProductID et Quantity:

Visualizations > Order Details

Build visual

Fields

- Quantity
- ProductID

Drill through

Cross-report  Off

Keep all filters  On

Add drill-through fields here

Order Details

- $\sum$  Discount
- OrderID
- ProductID
- $\sum$  Quantity
- $\sum$  UnitPrice

Orders

Products

- CategoryID
- Discontinued

on filtre l'input ProductID et on choisit 3 et 13 :

The screenshot shows a 'Filters' interface with a search bar at the top. Below it are two filter sections. The first section, 'ProductID', has the condition 'is 3 or 13'. It includes a 'Filter type' dropdown set to 'Basic filtering' and a list of checkboxes for Product IDs 2 through 9, with checkboxes for 3 and 13 selected. The second section, 'Quantity', has the condition 'is (All)'. At the bottom, there is a button 'Add data fields here'.

ProductID	Count
2	44
3	12
4	20
5	10
6	12
7	29
8	13
9	5

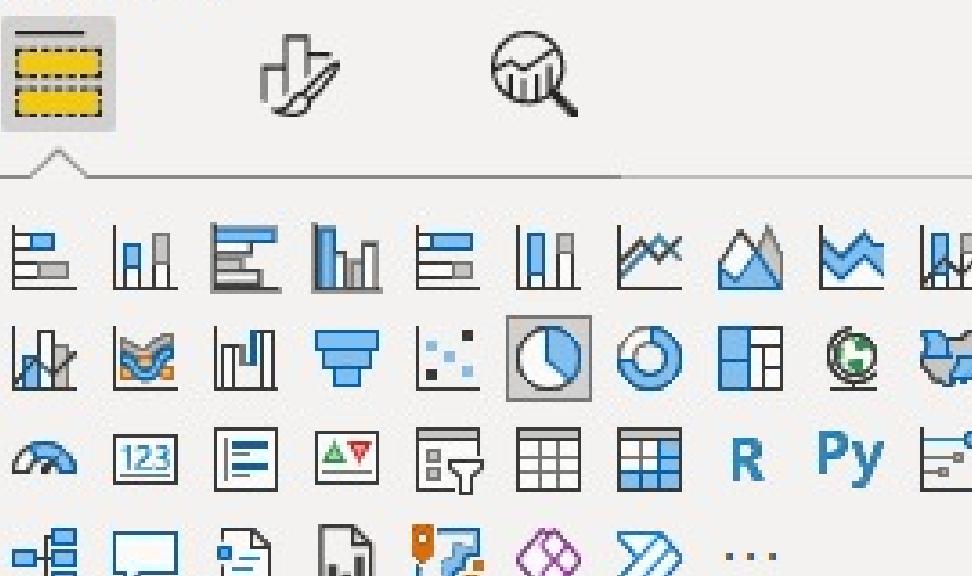
Et voilà le résultat finale

891	13
Quantity	ProductID
328	3
Quantity	ProductID

Afin de récupérer le nombre d'employés dans chaque ville on utilise une visualisation de 'Count of EmployeeID' en fonction de 'City'

**Visualizations**

Build visual



**Legend**

City

**Values**

Count of EmployeeID

**Details**

Add data fields here

**Fields**

Search

- Categories
- CustomerCustomerDemo
- CustomerDemographics
- Customers
- Employees
  - Address
  - BirthDate
  - City
  - Country
  - EmployeeID
  - Extension
  - FirstName
  - HireDate
  - HomePhone
  - LastName
  - Notes

Et voilà la distribution des employées selon les villes et on remarque que la majorité des employées sont situé à London.

Count of EmployeeID by City

