République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université des Sciences et de la Technologie Houari Boumediene

Faculté d'Electronique de l'Informatique

Département Informatique

Filière : Informatique

Spécialité : Big Data Analytics

Module : BI

Rapport de TP

Travail réalisé par le binôme :

GOUMEZIANE Kenza & AISSANI Anouar

Année Universitaire : 2022/2023

Table des matières

Table des figures

Introduction

Le projet du module Business Intelligence (BI) consiste à réaliser une solution décisionnelle
permettant de répondre au besoin d'analyse suivant :
Le nombre de commandes livrées et le nombre de commandes non encore livrées par client (ID client, Nom client, Ville, Pays), Employé (ID employé, nom et prénom, ville, pays), Mois et Année.

Le présent rapport est consacré à l'explication du travail réalisé.

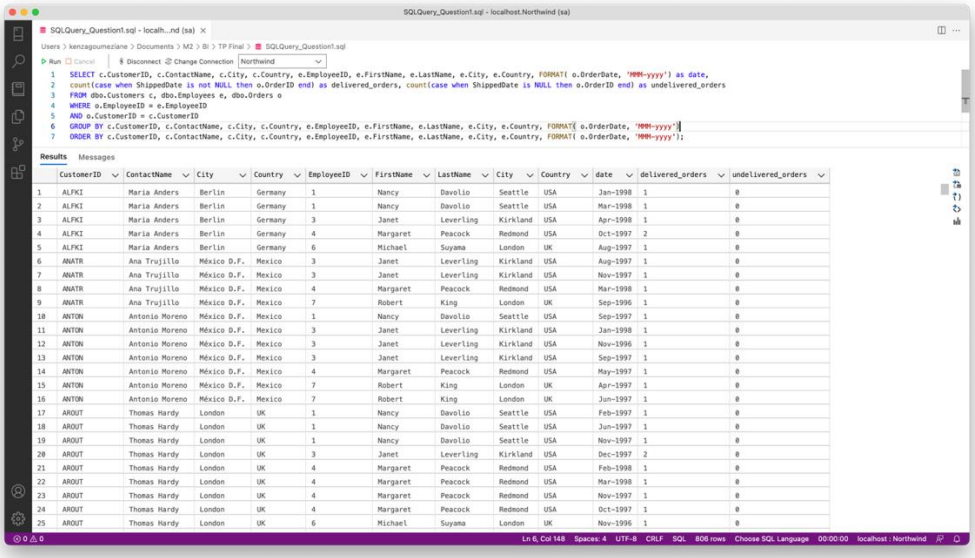
1. La requête qui répond au besoin du décideur

Nous rappelons que le décideur dans le cadre de notre projet vise à avoir **le nombre de commandes livrées** et le **nombre de commandes non encore livrées** par client (ID client, Nom client, Ville, Pays), Employé (ID employé, nom et prénom, ville, pays), Mois et Année.

Pour répondre à cette requête, nous avons utilisé la fonction **CASE** sous SQL Server.
Ainsi, si la date de livraison de la commande est à NULL alors nous incrémentons le nombre de commandes non livrées, sinon (la date de livraison n'est pas à NULL) nous incrémentons le nombre de commandes livrées comme suit :

```
SELECT c.CustomerID, c.ContactName, c.City, c.Country, e.EmployeeID, e.FirstName, e.LastName, e.City,
e.Country, FORMAT( o.OrderDate, 'MMM-yyyy') as date,
scount(case when ShippedDate is NULL then o.OrderID end) as undelivered_orders
FROM dbo.Customers c, dbo.Employees e, dbo.Orders o
WHERE o.EmployeeID = e.EmployeeID
AND o.CustomerID = c.CustomerID
GROUP BY c.CustomerID, c.ContactName, c.City, c.Country, e.EmployeeID, e.FirstName, e.LastName, e.City,
e.Country, FORMAT( o.OrderDate, 'MMM-yyyy')
ORDER BY c.CustomerID, c.ContactName, c.City, c.Country, e.EmployeeID, e.FirstName, e.LastName, e.City,
e.Country, FORMAT( o.OrderDate, 'MMM-yyyy');
```

Le résultat sous SQL Server est le suivant et nous retourne **806 lignes** au total:



	CustomerID	ContactName	City	Country	EmployeeID	FirstName	LastName	City	Country	date	delivered_orders	undelivered_orders
1	ALFKI	Maria Anders	Berlin	Germany	1	Nancy	Davolio	Seattle	USA	Jan-1998	1	0
2	ALFKI	Maria Anders	Berlin	Germany	1	Nancy	Davolio	Seattle	USA	Mar-1998	1	0
3	ALFKI	Maria Anders	Berlin	Germany	3	Janet	Leverling	Kirkland	USA	Apr-1998	1	0
4	ALFKI	Maria Anders	Berlin	Germany	4	Margaret	Peacock	Redmond	USA	Oct-1997	2	0
5	ALFKI	Maria Anders	Berlin	Germany	6	Michael	Suyama	London	UK	Aug-1997	1	0
6	ANATR	Ana Trujillo	México D.F.	Mexico	3	Janet	Leverling	Kirkland	USA	Aug-1997	1	0
7	ANATR	Ana Trujillo	México D.F.	Mexico	3	Janet	Leverling	Kirkland	USA	Nov-1997	1	0
8	ANATR	Ana Trujillo	México D.F.	Mexico	4	Margaret	Peacock	Redmond	USA	Mar-1998	1	0
9	ANATR	Ana Trujillo	México D.F.	Mexico	7	Robert	King	London	UK	Sep-1996	1	0
10	ANTON	Antonio Moreno	México D.F.	Mexico	1	Nancy	Davolio	Seattle	USA	Sep-1997	1	0
11	ANTON	Antonio Moreno	México D.F.	Mexico	3	Janet	Leverling	Kirkland	USA	Jan-1998	1	0
12	ANTON	Antonio Moreno	México D.F.	Mexico	3	Janet	Leverling	Kirkland	USA	Nov-1996	1	0
13	ANTON	Antonio Moreno	México D.F.	Mexico	3	Janet	Leverling	Kirkland	USA	Sep-1997	1	0
14	ANTON	Antonio Moreno	México D.F.	Mexico	4	Margaret	Peacock	Redmond	USA	May-1997	1	0
15	ANTON	Antonio Moreno	México D.F.	Mexico	7	Robert	King	London	UK	Apr-1997	1	0
16	ANTON	Antonio Moreno	México D.F.	Mexico	7	Robert	King	London	UK	Jun-1997	1	0
17	AROUT	Thomas Hardy	London	UK	1	Nancy	Davolio	Seattle	USA	Feb-1997	1	0
18	AROUT	Thomas Hardy	London	UK	1	Nancy	Davolio	Seattle	USA	Jun-1997	1	0
19	AROUT	Thomas Hardy	London	UK	1	Nancy	Davolio	Seattle	USA	Nov-1997	1	0
20	AROUT	Thomas Hardy	London	UK	3	Janet	Leverling	Kirkland	USA	Dec-1997	2	0
21	AROUT	Thomas Hardy	London	UK	4	Margaret	Peacock	Redmond	USA	Feb-1998	1	0
22	AROUT	Thomas Hardy	London	UK	4	Margaret	Peacock	Redmond	USA	Mar-1998	1	0
23	AROUT	Thomas Hardy	London	UK	4	Margaret	Peacock	Redmond	USA	Nov-1997	1	0
24	AROUT	Thomas Hardy	London	UK	4	Margaret	Peacock	Redmond	USA	Oct-1997	1	0
25	AROUT	Thomas Hardy	London	UK	6	Michael	Suyama	London	UK	Nov-1996	1	0

Figure 1: La requête SQL

Le script de la requête est joint avec le rapport de TP.

2. Modélisation du Data Warehouse

Nous passons maintenant à la conception et réalisation de notre entrepôt de données

Pour répondre au besoin du décideur, nous avons jugé nécessaire d'avoir trois tables de dimension :

- La **dimension Client** : Cette dimension nous permet de récupérer les informations qui concernent les clients et intéressent le décideur, notamment le *nom du client, sa ville et son pays*.
- La **dimension Employé** : Cette dimension nous permet de récupérer les informations qui concernent les employés et intéressent le décideur, notamment *le nom de l'employé, son prénom, sa ville et son pays*.
- La **dimension Temps** : Une dimension temporelle est obligatoire dans chaque entrepôt de données. Ici, nous récupérons la date à partir de l'attribut « date de commande » qui se trouve au niveau de la table « Orders » de « Northwind » vue que cet attribut nous permet de récupérer *les années et les mois de chaque année* en relation avec notre cas d'étude.

Passons maintenant à la table de fait.

Notre **table de fait Commande** contient deux faits : le *nombre de commandes livrées* et le *nombre de commandes non livrées*. Bien-sûr, elle contient également une clé primaire qui est formée à partir de trois clés étrangères. Chaque clé étrangère référencie une des clés primaires de nos trois dimensions citées en haut.

Les images suivantes montrent les deux conceptions que nous avons réalisées :

a. Conception en étoile

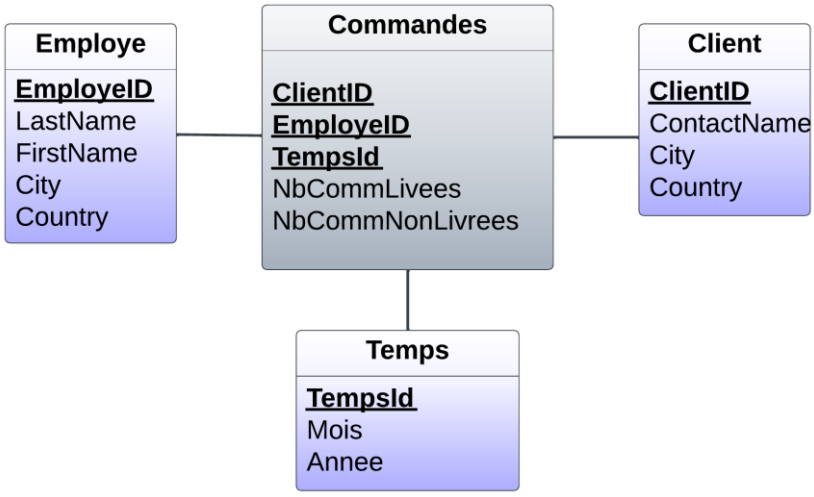


Figure 2: Modélisation en étoile du DW

b. Conception en flocon

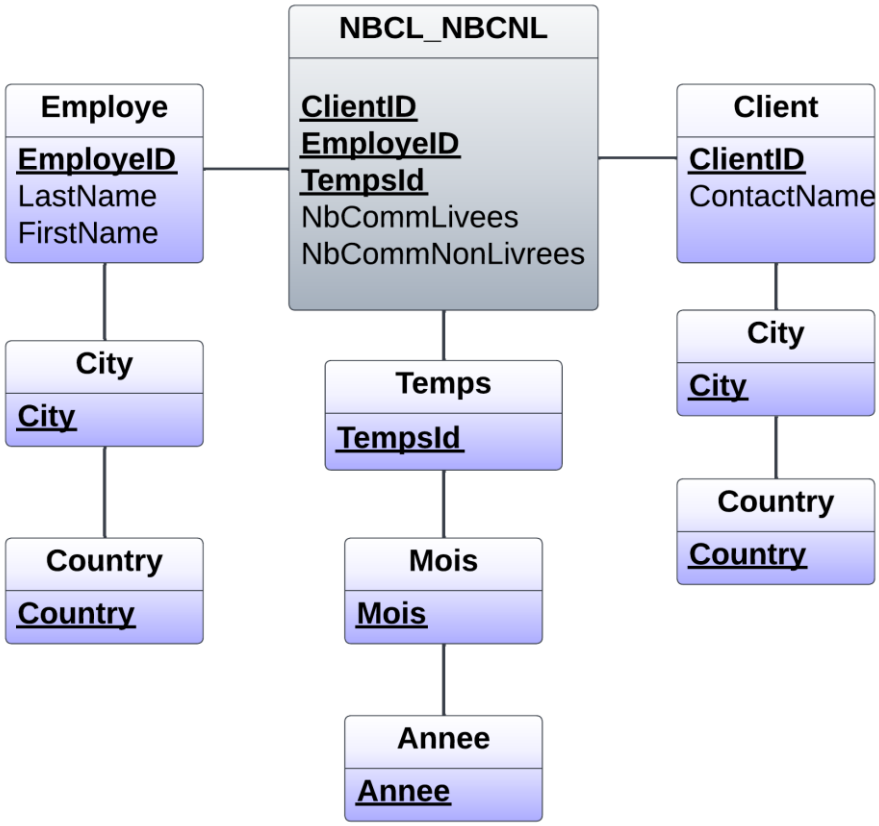


Figure 3: Modélisation en flocon du DW

Après avoir modélisé notre ED, nous passons à sa réalisation sous SQL Server.

c. Réalisation sous SQL Server

Nous avons adopté la modélisation en étoile car elle est plus adéquate pour répondre efficacement à notre besoin.

Nous avons donc créé nos tables de dimension Client, Employé et Temps et ensuite nous avons créé notre table de faits Commandes

Ceci est une capture d'écran de l'exécution de notre script SQL ou après une actualisation, nous constatons que les tables ont bien été créées

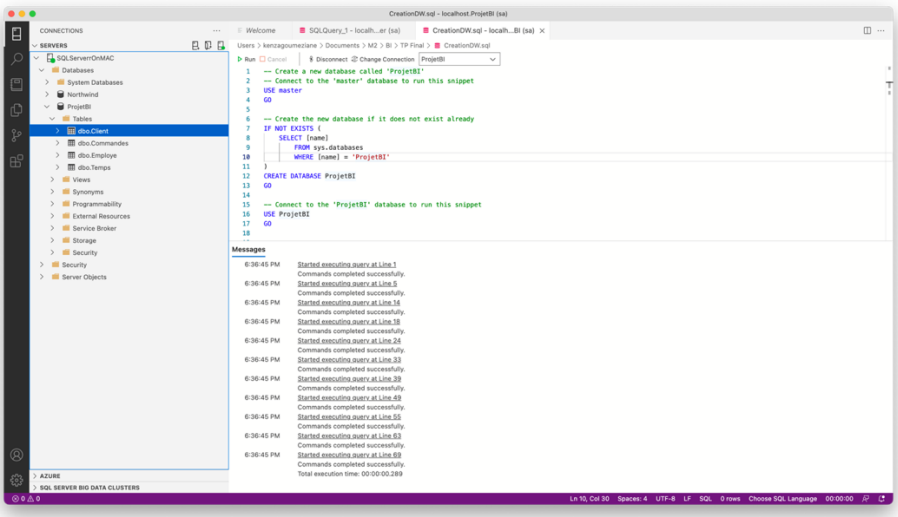


Figure 4: Création du DW sur SQL Server

Le script de création du DW est joint avec le rapport de TP.

3. Réalisation de l'intégration des données avec Talend Open Studio for Data Integration

Nous rappelons que nous avons deux sources de données :
En premier lieu, nous avons la base de données **« Northwind »** et en second lieu, nous avons deux fichiers plats **« Client.txt »** et **« Commande.txt »**.
Afin de pouvoir intégrer leur contenu dans notre entrepôt de données, nous devons premièrement nous connecter à ces deux sources à partir de Talend ou plus précisément, nous devons les charger au niveau du dossier « Metadata »

a. Connexion aux fichiers délimités et importation des données

Pour créer nos fichiers délimités, nous suivons les étapes suivantes :

Premièrement, nous déroulons le menu **« Metadata »,** nous choisissons l'option **« File delimited »** puisqu'un fichier texte appartient à cette catégorie et nous commençons à créer notre connexion.

La fenêtre suivante s'ouvre et nous invite à donner un **nom** à notre fichier délimité ainsi qu'un **objectif** et une **description** comme suit :

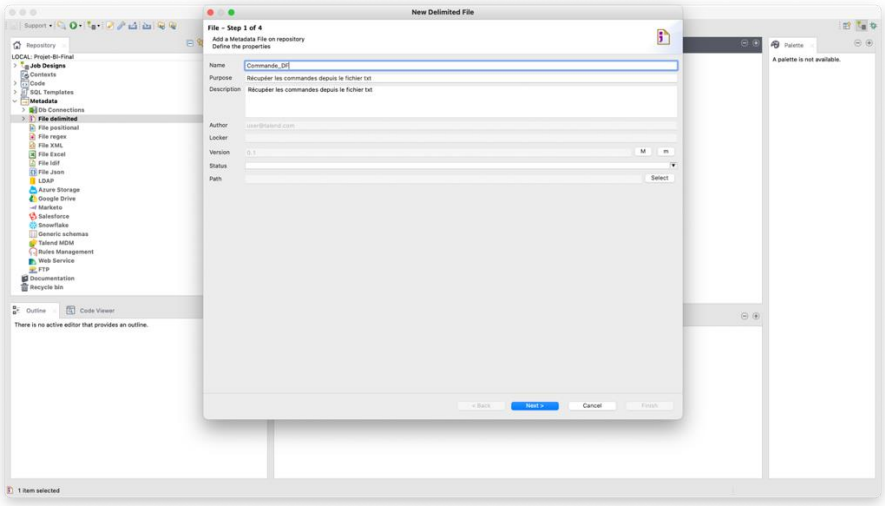


Figure 5: : Création d'une connexion à un fichier délimité – Étape 01

Après cela, une nouvelle fenêtre s'ouvre où nous spécifions l'encodage de notre fichier qui est **UTF-8**, le séparateur de données (colonnes) qui est le caractère **« # »** ainsi que le séparateur de lignes qui est le caractère de retour à la ligne **« \n ».**
Après actualisation nous obtenons la visualisation suivante :

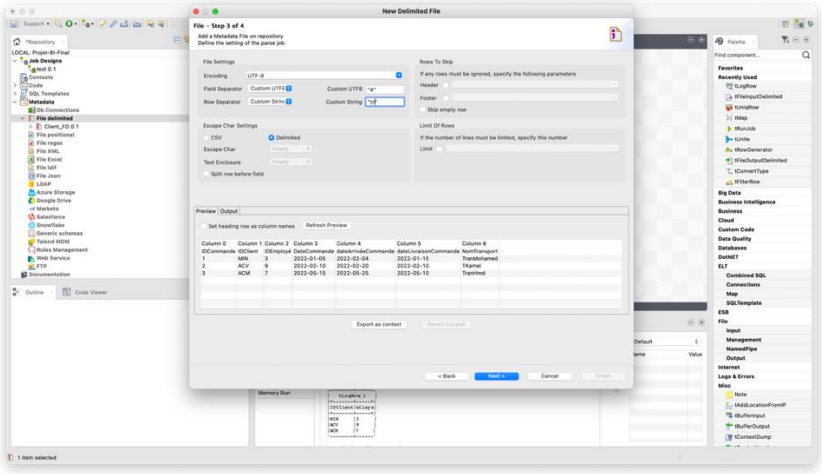


Figure 6: Création d'une connexion à un fichier délimité – Étape 02

Nous choisissons les données souhaitées et nous vérifions leurs types puis enfin, nous clôturons l'étape de connexion au premier fichier.

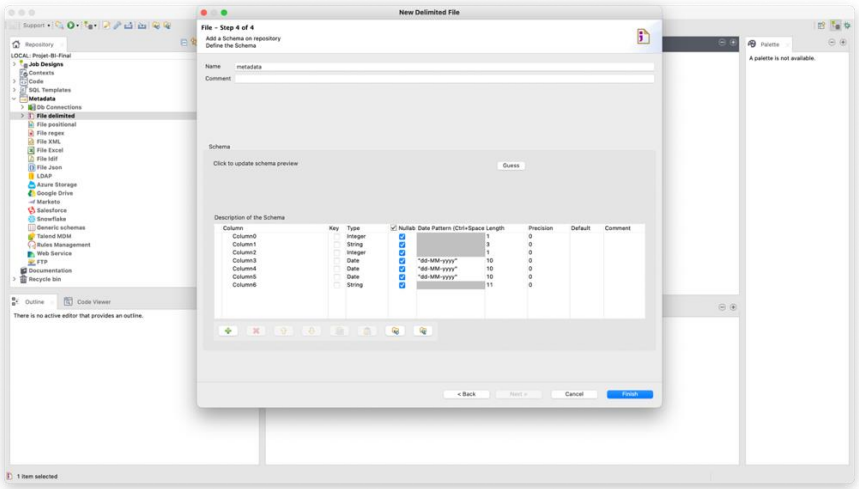


Figure 7 : Création d'une connexion à un fichier délimité – Étape 03

Nous répétons les mêmes étapes pour l'autre fichier délimité « Client ».

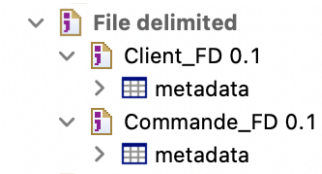


Figure 8: Fichiers délimités connectés

b. Connexion à la base de données Northwind et importation des données

A ce stade, nous avons nos connexions aux fichier txt de notre projet, passons maintenant aux connexions à nos bases de données.

Pour ce faire, nous choisissons l'option **« Db connections »** à la place de de « File Delimited ». La fenêtre qui s'ouvre nous permet comme pour celle des fichiers délimités de donner un **nom**, un **objectif** et une **description** à notre connexion.

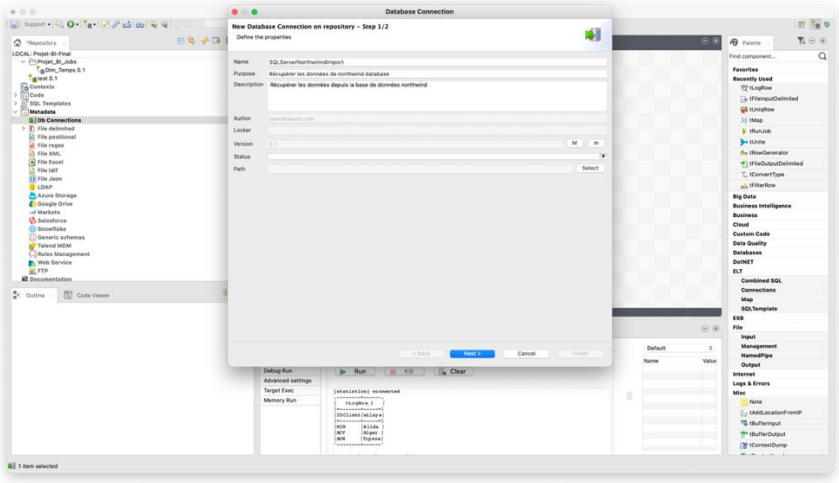


Figure 9: Création d'une connexion à une BD - Étape 01

Dans la prochaine fenêtre, nous saisissons les informations de connexion à notre base de données « Northwind » :

- Le type de base de données qui est « **Microsoft SQL Server** ».
- Notre « **jdbc** ».
- Les identifiants de connexion « **Login** » et « **Password** ».
- Le serveur « **Localhost** ».
- Le port.
- Le schéma qui est « **dbo** » dans ce cas-là.

Nous testons la connexion et nous la validons comme suit :

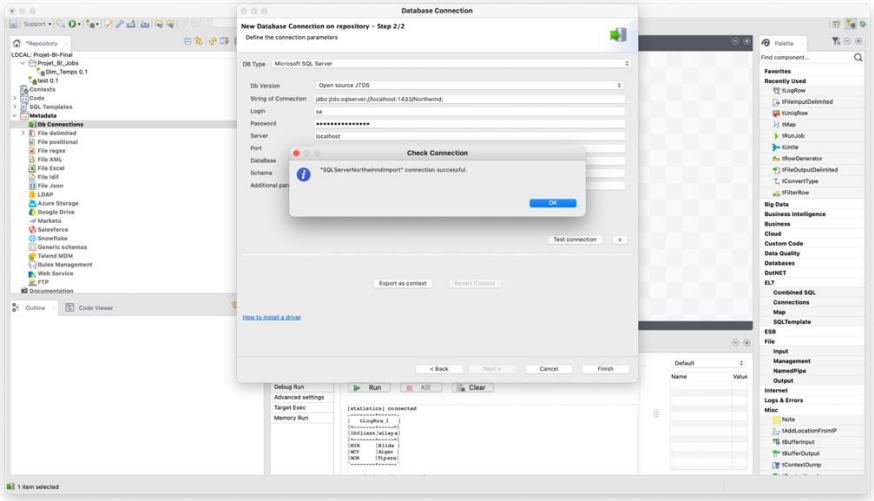


Figure 10: Création d'une connexion à une BD - Étape 02

Maintenant que la connexion est réussie, nous devons récupérer les tables et les colonnes qui nous intéressent.

Pour cela, nous cliquons sur « **Rtrieve Schema** » à partir de la connexion déjà créée :

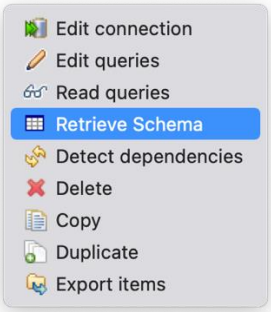


Figure 11: Extraire des données depuis Northwind DB

Nous sélectionnons trois tables : la table qui contient les clients **« Customers »,** la table qui contient les employés **« Employees »** et la tables des commandes **« Orders ».**

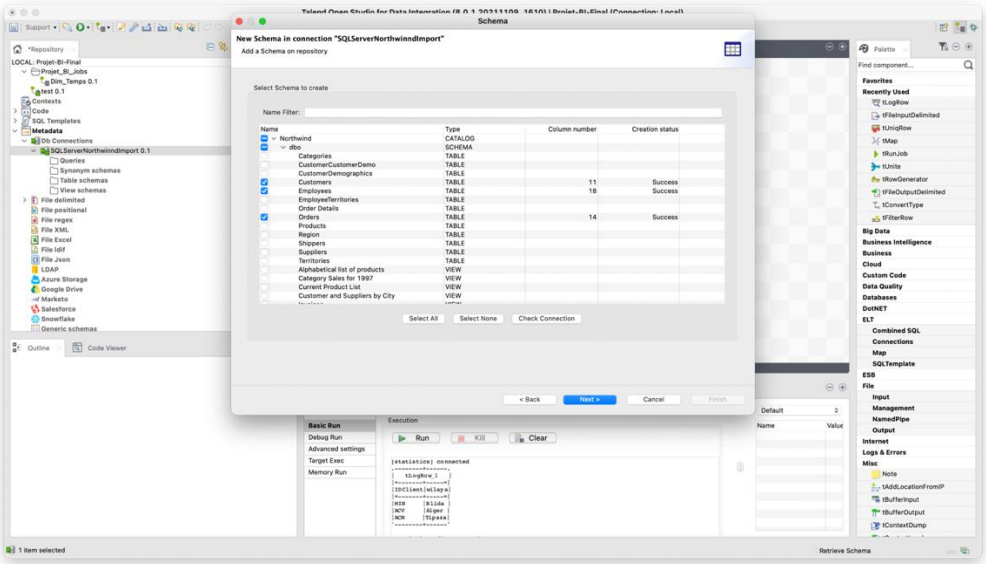


Figure 12: Choix des tables à Importer depuis Northwind DB

Par la suite, nous sélectionnons pour chaque table, les colonnes dont nous aurons besoin pour la suite du TP

- Pour la table des clients, nous récupérons l'identifiant de chaque client, son nom, son pays et sa ville.
- Pour la table des employés, nous récupérons l'identifiant de chaque employé, son nom, son prénom, son pays et sa ville.

- Pour la table commandes, nous récupérons l'identifiant de chaque commande, l'identifiant de chaque client, l'identifiant de chaque employé, la date de commande et la date de livraison.

Clients :

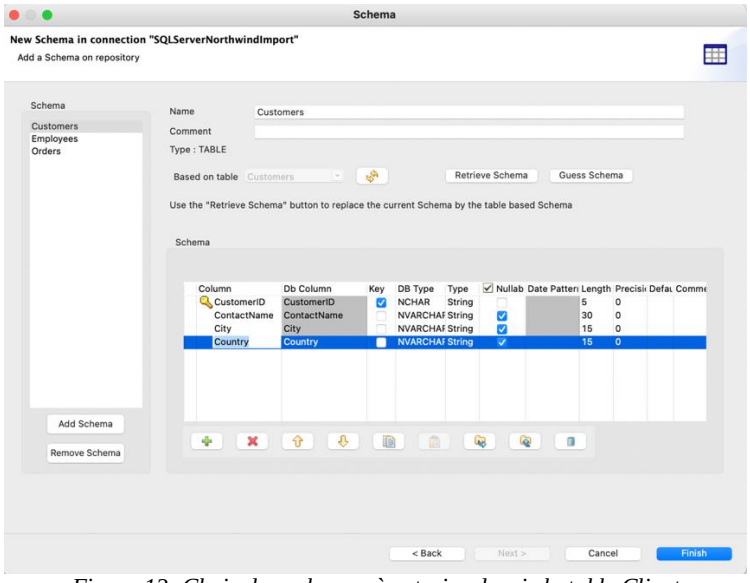


Figure 13: Choix des colonnes à extraire depuis la table Client

Employés :

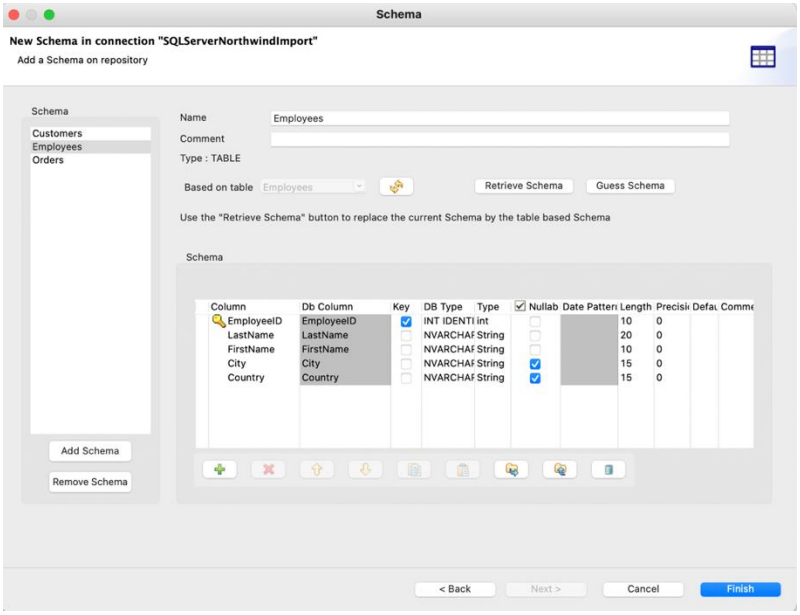


Figure 14: Choix des colonnes à extraire depuis la table Employé

Commandes :

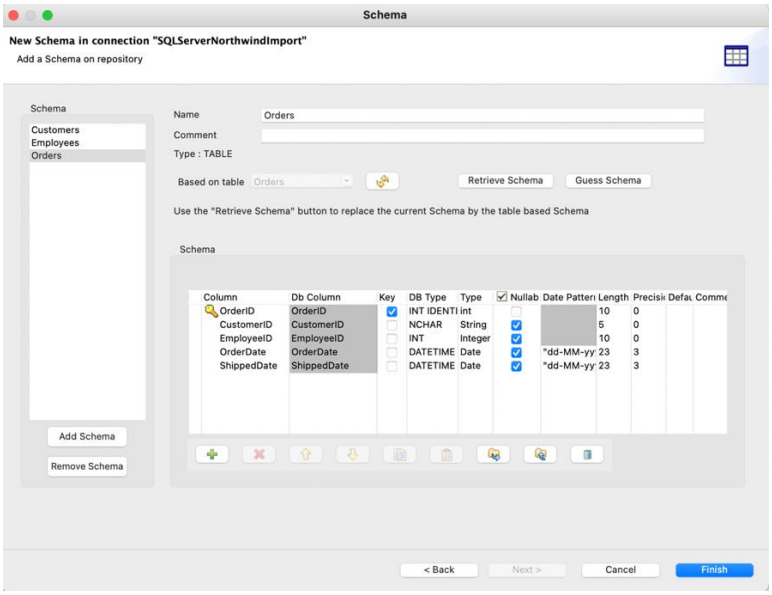


Figure 15: Choix des colonnes à extraire depuis la table Commande

Nous validons nos choix et nous aurons clôturé notre connexion à la base de données « Northwind » ainsi que l'extraction de ses données.

c. Connexion à notre entrepôt de données

Nous devons nous connecter à l'ED que nous avons créé sur SQL Server car il représente notre output final.

Nous suivons les mêmes étapes vues dans le titre précédent pour nous connecter à notre entrepôt de données :

Nous spécifions le nom, l'objectif, et la description.

Nous spécifions les informations de connexion et nous testons la connexion.

En revanche, puisque nous avons besoin de remplir tout notre Datawarehouse alors on récupère son schéma en entier :

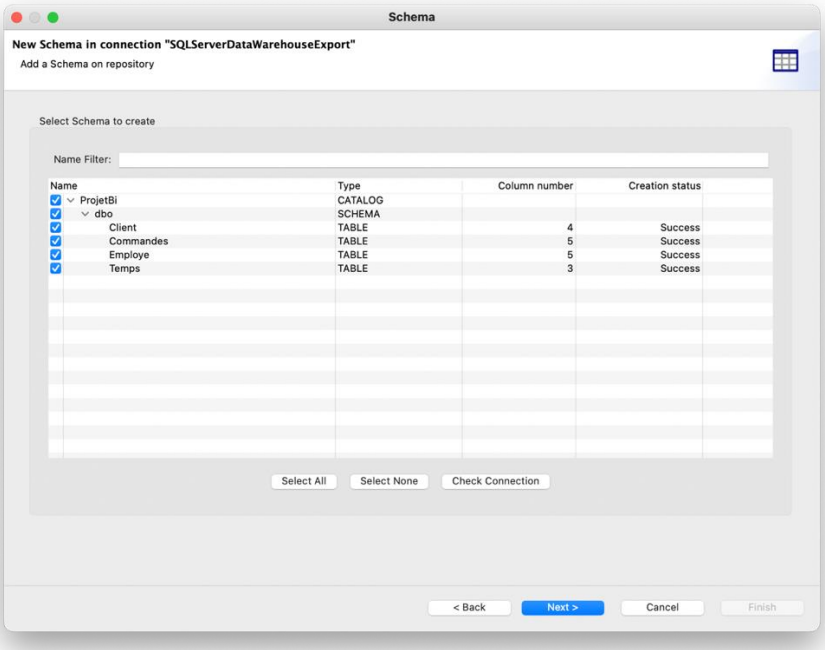


Figure 16: : Récupération du schéma de l'entrepôt de données

Nous validons et nous aurons maintenant accès à toutes les données dont nous avons besoin comme le montre la figure suivante :

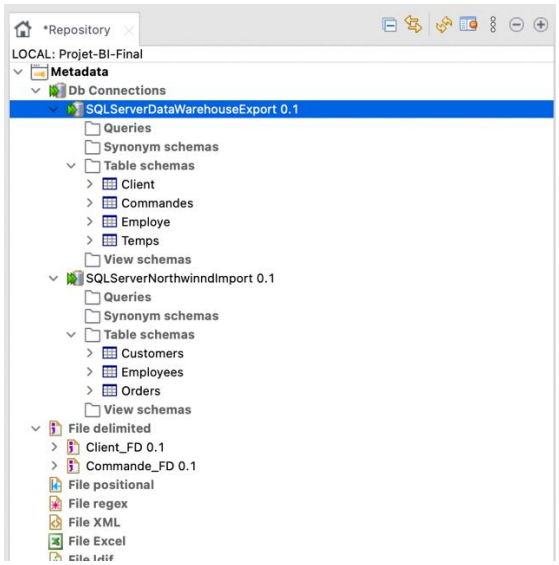


Figure 17: : Affichage des connexions

Le titre suivant est consacré aux transformations effectuées sur nos données afin de les intégrer dans notre ED.

d. Création des jobs

Avant d'insérer nos données dans l'entrepôt de données, nous devons d'abord y effectuer les transformations nécessaires.

Commençons par le remplissage de la dimension Temps.

- **La dimension Temps**

Nous créons un job depuis « **Job Designs** » que nous nommons Dim_Temps.
La fenêtre suivante nous permet de saisir son nom, son objectif et sa description

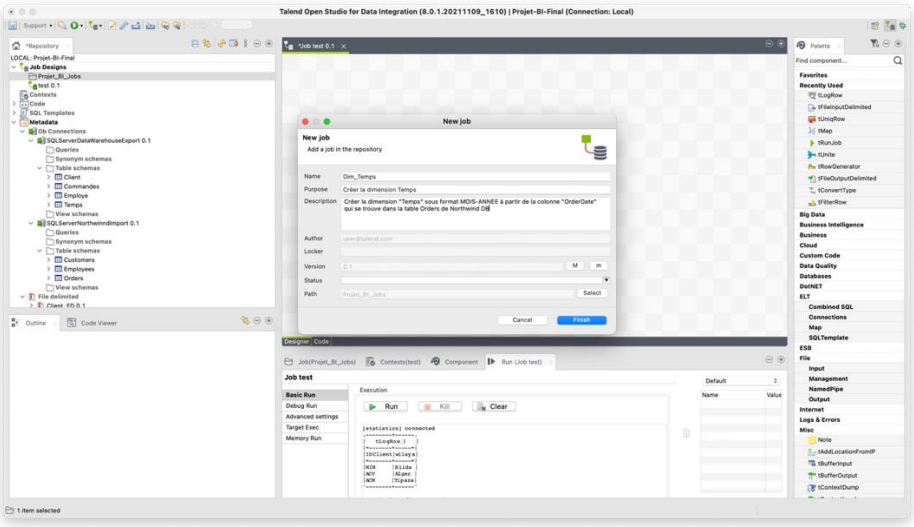


Figure 18: : Création d'un job sous Talend

La figure suivante montre les composants utilisés dans ce job ainsi que son exécution.

Nous constatons que nous avons deux inputs. Effectivement, nous récupérons nos dates à partir de la base de données Northwind d'où le composant tDBInput mais aussi à partir du fichier commande d'où le composant tFileInputDelimited.

Les deux premiers composants Tmap nous permettent de récupérer la date sous format 'MM-yyyy', le mois sous format 'MM' et l'année sous format 'yyyy' pour chacune des deux sources.

Le composant tUnite nous permet de faire l'union entre les deux listes de dates obtenues.
Le composant tUniqueRow filtre les dates pour éliminer les doubons.
Et enfin, nous insérons nos données dans un composant tDbOutput qui est nul autre que notre table Temps de l'ED.

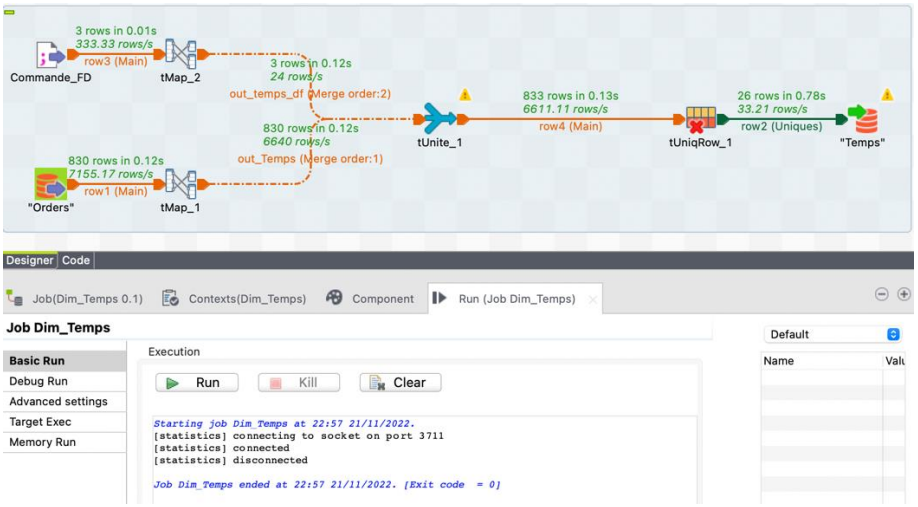


Figure 19: : Exécution du job Dim_Temps

Nous constatons que le job a inséré 26 lignes dans la table Temps. Nous vérifions ce résultat sur SQL Server :

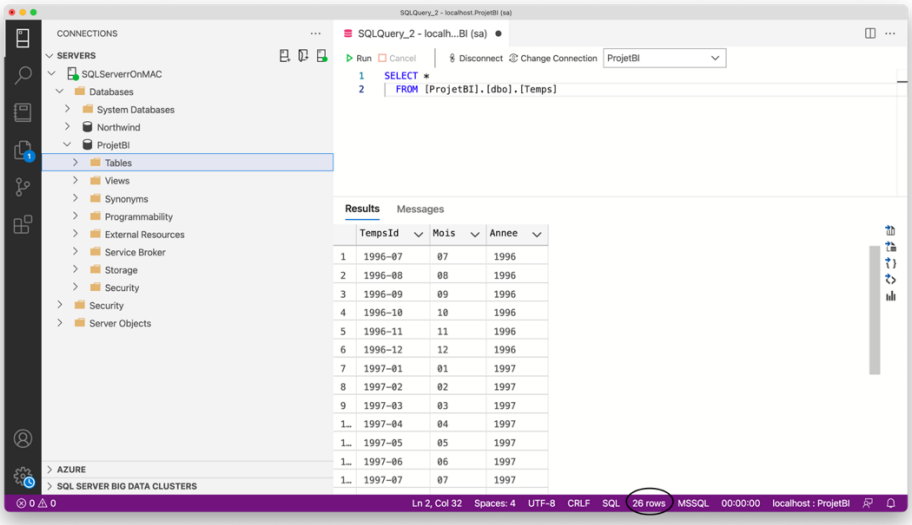


Figure 20: : Résultat de l'exécution du job Dim_Temps dans SQL Server

- **La dimension Employe**

Nous créons un nouveau job depuis « **Job Designs** » comme vu en haut et nous le nommons Dim_Employe.

Cette fois-ci la figure suivante montre que nous avons un seul input de type tDBInput puisque tous les employés sont dans la base de données Northwind.

Le composant tMap nous permet donc de récupérer les informations des employés sans faire d'autres transformations.

Et enfin, nous insérons nos données dans un composant tDbOutput qui est la table Employe de l'ED.

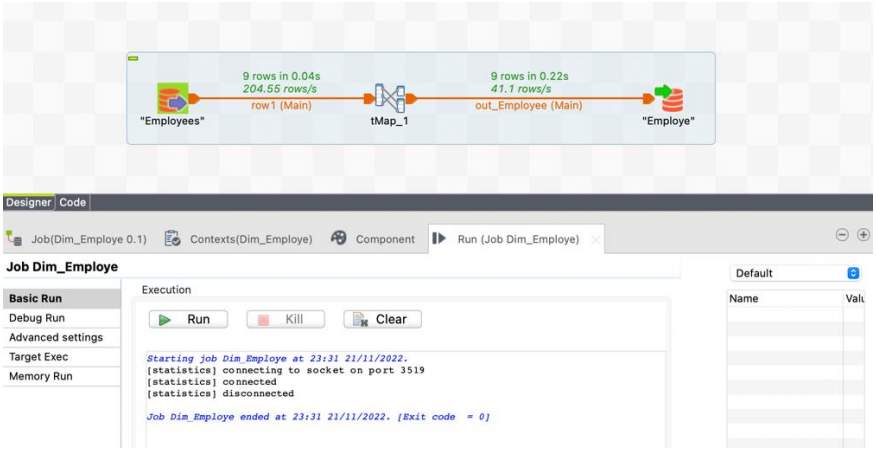


Figure 21: : Exécution du job Dim_Employe

L'exécution montre que **9** lignes ont été insérées dans la table Employe. Nous vérifions ce résultat sur SQL Server :

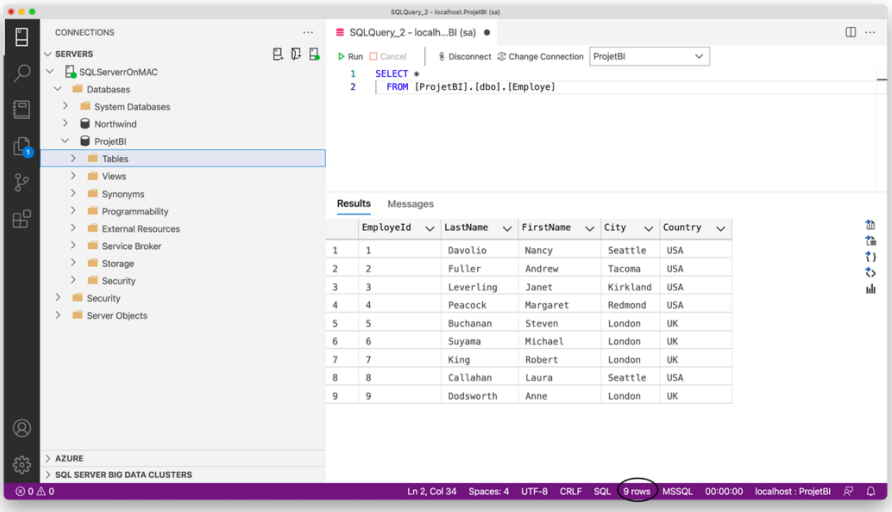


Figure 22: : Résultat de l'exécution du job Dim_Employe dans SQL Server

- **La dimension Client**

Nous créons un nouveau job depuis « **Job Designs** » et nous le nommons Dim_Client.
Pour les clients, nous avons deux inputs. Un composant tDBInput pour récupérer les clients qui sont dans Northwind et un composant tFileInputDelimited pour récupérer les clients qui sont dans le fichier txt.

Le composant tMap associé au fichier délimité nous permet d'ajouter les deux colonnes manquantes aux clients à savoir :

- Le nom du client avec **TalendDataGenerator.getLastName()**
- Son pays que nous avons mis par défaut à « Algeria ».

Le composant tUnite nous permet de faire l'union entre les deux listes de clients obtenues.

Enfin, nous insérons nos clients dans la base de données Client.

On remarque que trois clients ont été récupérés à partir du fichier txt tandis que 91 clients ont été récupérés depuis la base de données Northwind.

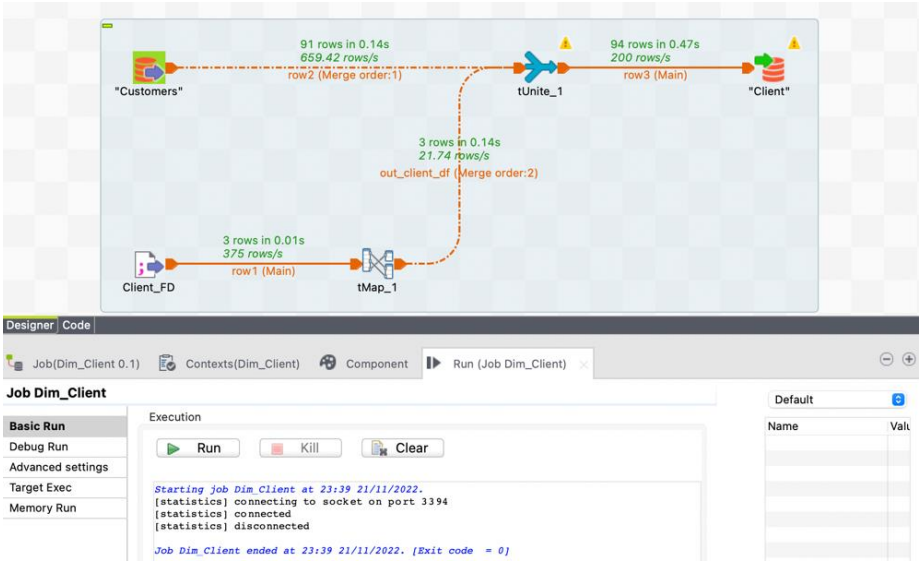


Figure 23: : Exécution du job Dim_Client

Vérification du résultat sous SQL Server :

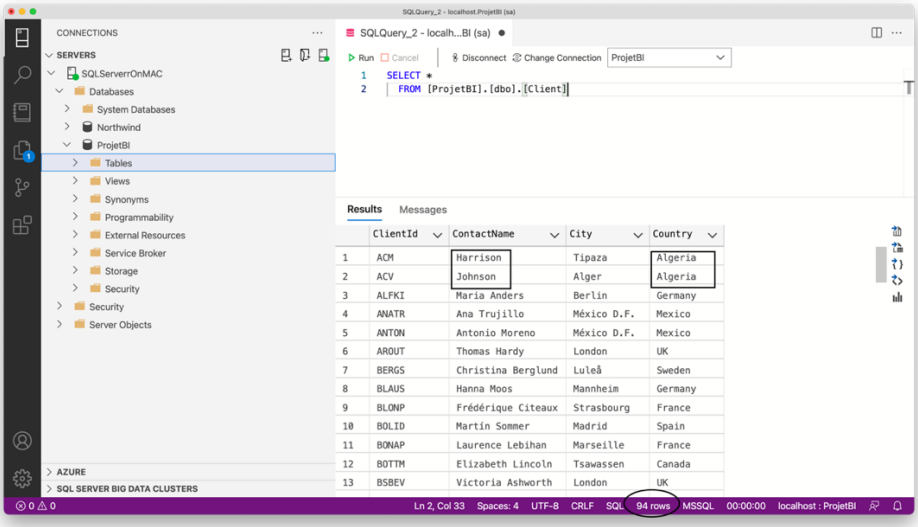


Figure 24: : Résultat de l'exécution du job Dim_Client dans SQL Server

- **La table de faits Commandes**

Nous créons un nouveau job depuis « **Job Designs** » et nous le nommons Fait_Commandes.

Pour les commandes, nous avons également deux inputs. Un composant tDBInput pour récupérer les commandes qui sont dans Northwind et un composant tFileInputDelimited pour récupérer les clients qui sont dans le fichier txt.

Les deux premiers composants Tmap nous permettent de récupérer la date avec le bon format puisque les deux formats sont différents selon la source et de transformer la date de commande en type String pour être conforme à la clé primaire de la dimension Temps.

Le composant tUnite nous permet de faire l'union entre les deux listes de commandes obtenues.

Le deuxième composant tmap est le plus important car il nous permet de joindre nos trois dimensions « Client », « Employe » et « Temps » avec le résultat de l'union obtenu précédemment.

Et enfin, nous insérons nos données dans un composant tDbOutput qui représente la table de faits « commandes ».

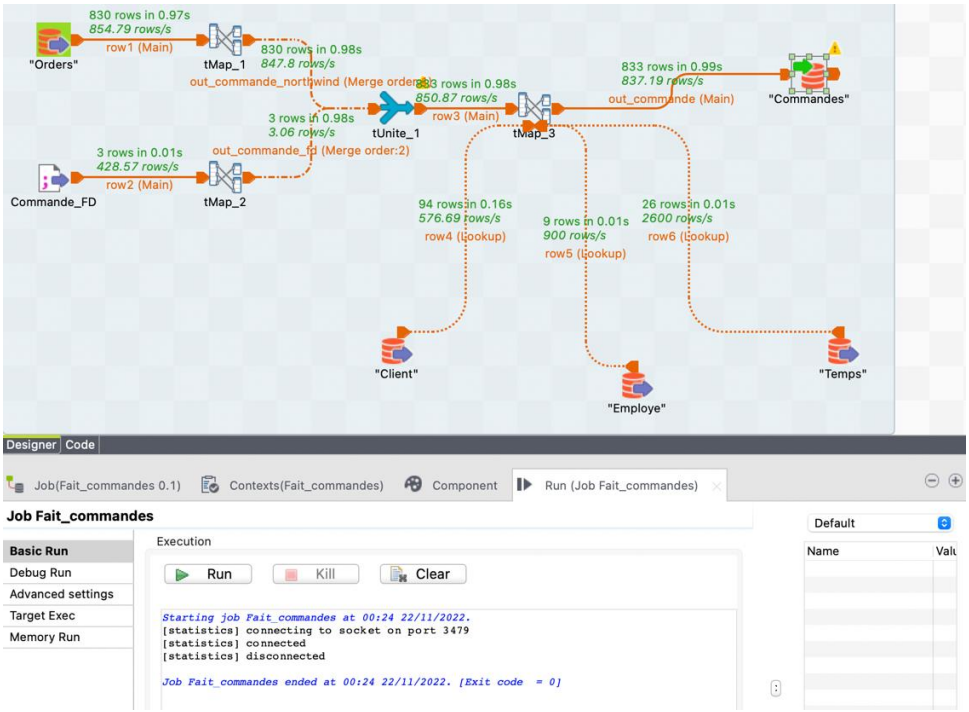


Figure 25: : Exécution du job Fait_Commandes

Vérification du résultat sous SQL Server :

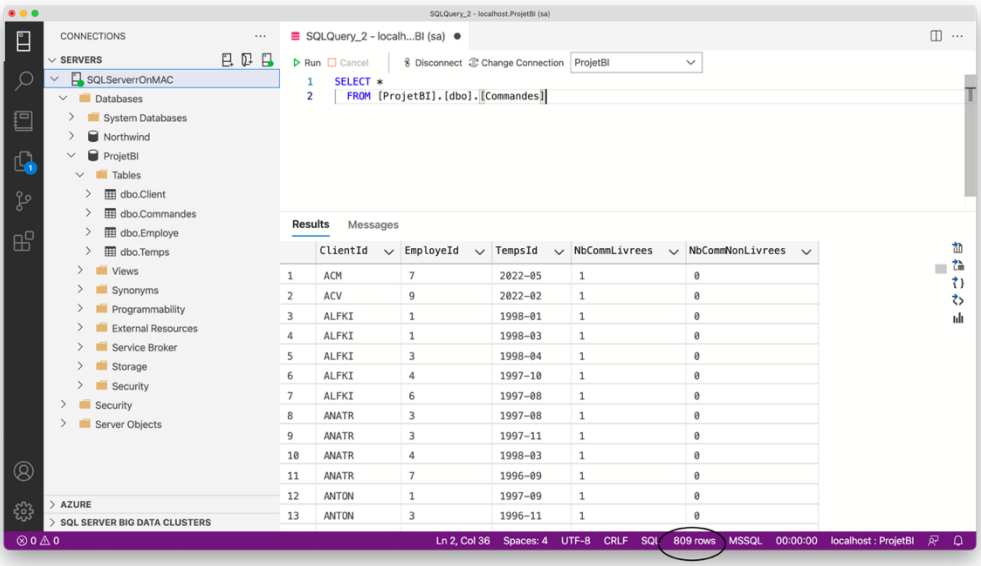


Figure 26: : Résultat de l'exécution du job Fait_Commandes dans SQL Server

<u>Remarque</u> : Nous avons spécifié la condition suivante « Insert if not exist » avant d'éxécuter les jobs et c'est pour cela que nous pouvons les exécuter autant de fois que nous le voulons sans affecter le résultat et sans avoir des erreurs.

`

4. Création des tableaux de bord

La dernière partie du rapport est consacrée à la visualisation de nos données dans les tableaux de bord.
Nous avons créé trois tableaux de bord que nous définissons comme suit :

a. Le tableau de bord des Employés

Ce dashboard nous informe sur le comportement des employés.

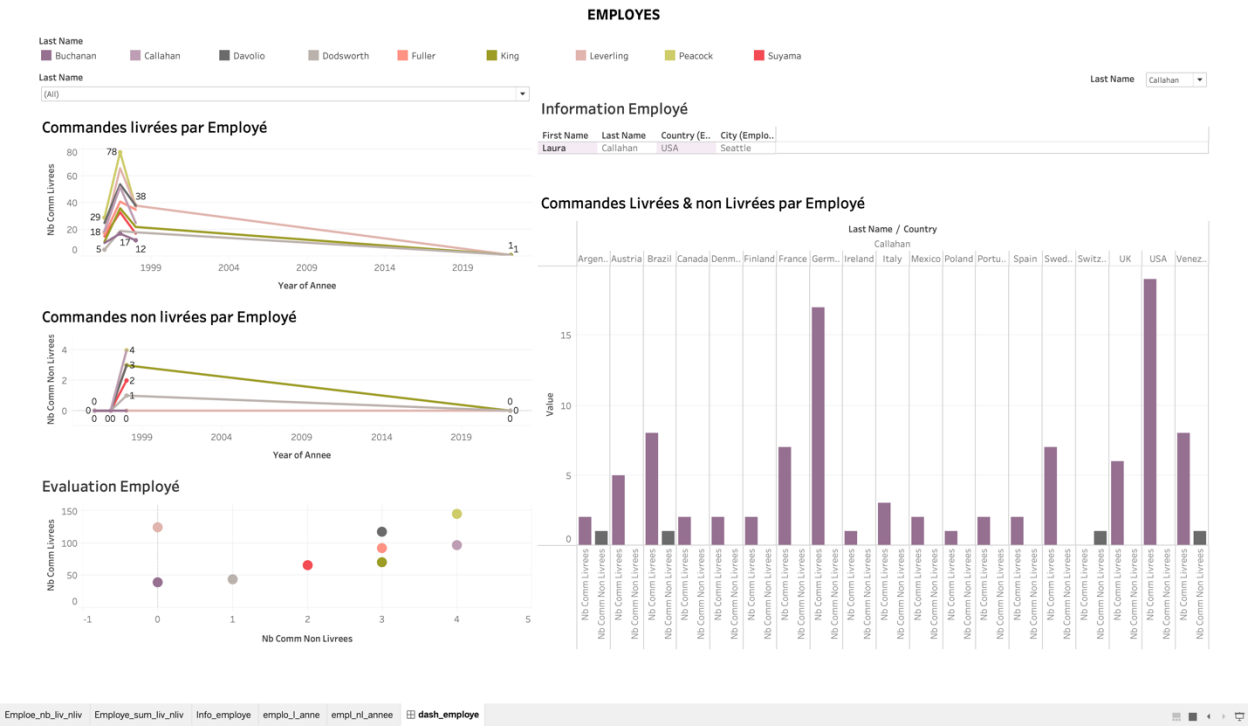


Figure 27: : Tableau de bord – Employés

- A droite, nous avons un **filtre** qui nous permet de sélectionner **un seul employé**. Après cela le **graphique à deux barres** change et nous donne le **nombre de commandes que l'employé a livré et le nombre de commandes qu'il n'a pas livré par pays**. Au même temps, le **tableau** d'au-dessus est mis à jour et affiche les **informations de l'employé** à savoir son nom, son prénom, sa ville et son pays.

- A gauche nous avons un autre **filtre** qui nous permet de sélectionner cette fois-ci **un ou plusieurs employés**. Après ça, les **deux graphiques en lignes** affichent le **nombre de commandes livrées par année et le nombre de commandes non livrées par année** respectivement pour le ou les clients sélectionnés.

- Pour le dernier **graphique en points** qui se trouve en bas à gauche, il nous permet de rapidement **évaluer un employé** en fonction du **nombre de commandes qu'il a livré et du nombre de commandes qu'il n'a pas livré.**

  Par exemple l'employé Leverling est bon car il a livré plusieurs commandes et il n'a aucune commande non livrée ce qui n'est pas le cas pour l'employé King.

b. Le tableau de bord des Clients

Ce tableau de bord nous informe sur les clients.

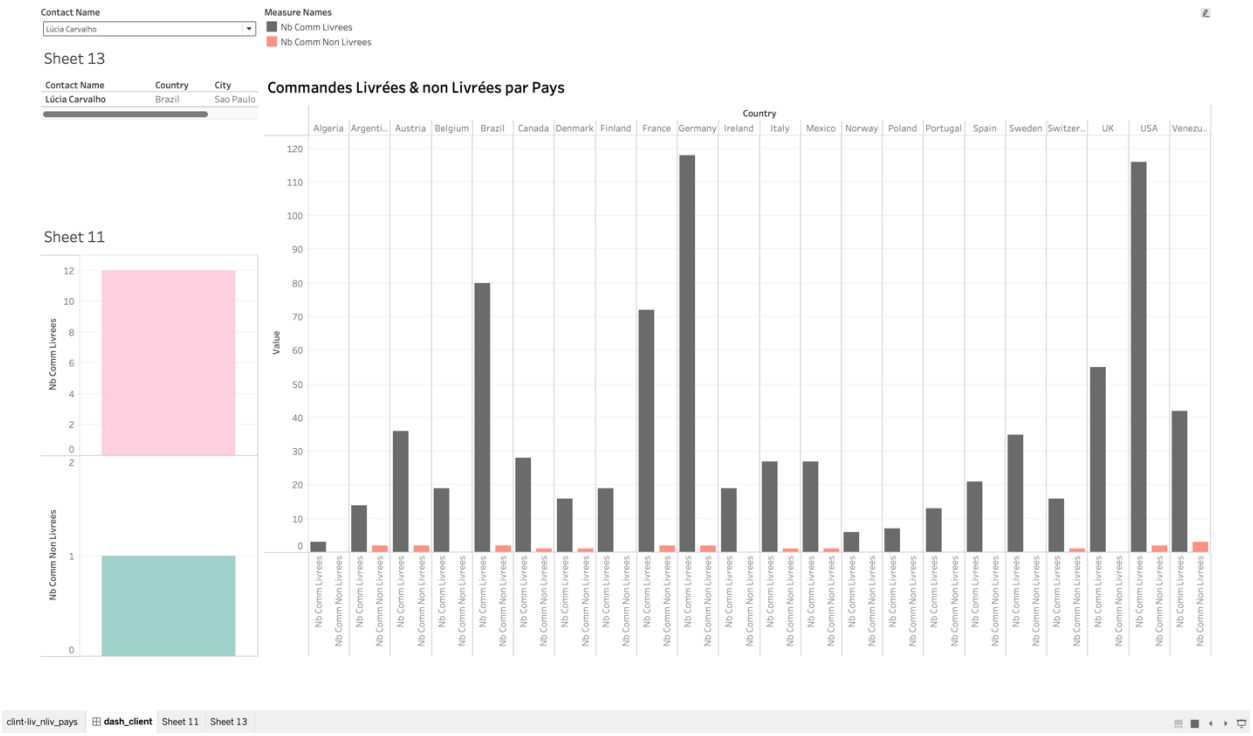


Figure 28: : Tableau de bord – Clients

- Le **graphique à deux barres** à droite recense **le nombre de commandes livrées et non livrées par pays des clients**. Il nous permet de détecter les pays avec le plus de commandes non livrées et ainsi tenter de trouver une solution au problème.

- A gauche, nous avons mis un **filtre** qui nous permet de sélectionner **un client** parmi les 94 clients du DW et d'afficher ses **informations sous forme de tableau** ainsi que **le nombre de commandes qui lui ont été livrées et le nombre de commandes que ne lui ont pas été livrées** sous forme de **graphe à deux barres**.

c. Le tableau de bord du Temps

Ce tableau de bord nous informe sur l'évolution des commandes en fonction du temps :

.

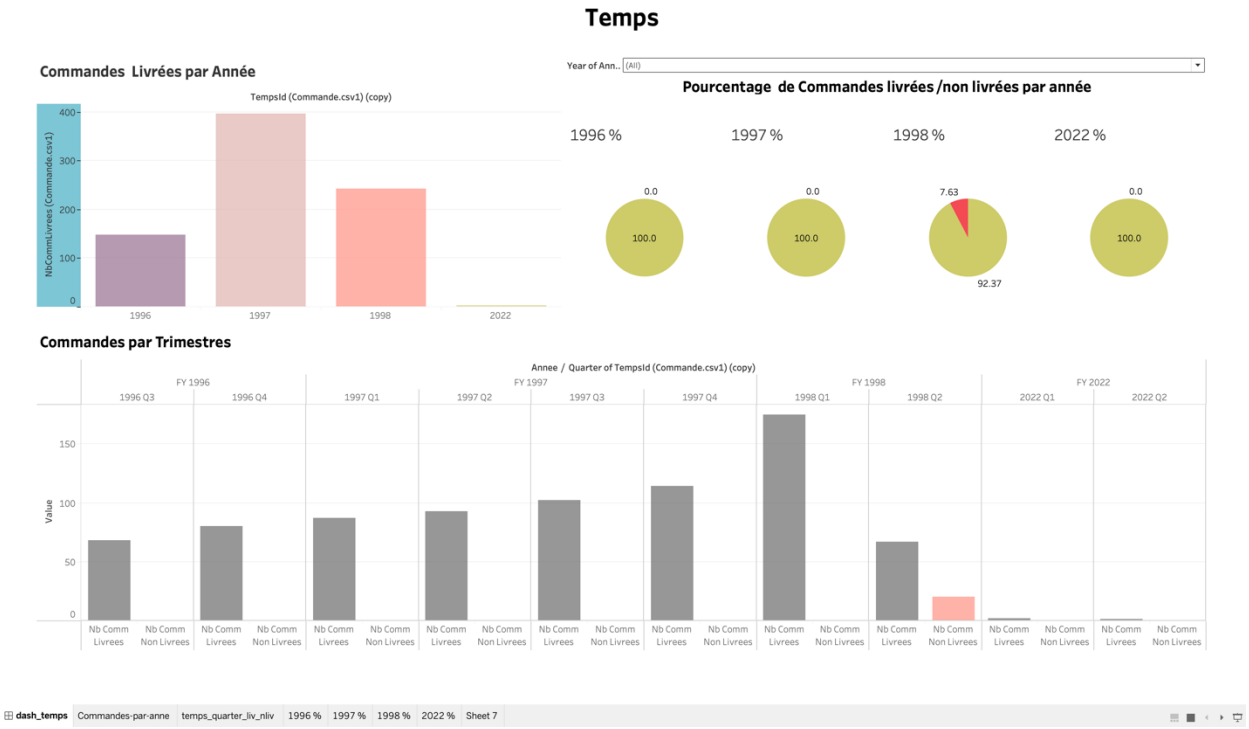


Figure 29: : Tableau de bord – Temps

- Les **quatre diagrammes circulaires** nous donnent le **pourcentage de commandes livrées et le pourcentage de commandes non livrées par année**.
  On remarque à travers eux qu'en 1998 il y a plusieurs commandes non livrées.

- Le **diagramme en barre** en haut à gauche nous donne **la somme des commandes livrées par années**.

- **Le diagramme en barre** en bas nous donne la **somme des commandes livrées par trimestres**.

- Dans ce tableau de bord nous pouvons **filtrer selon les années**.

Conclusion

Ce projet a été très bénéfique pour nous et nous a permis de mettre en œuvre nos connaissances acquises tout au long du semestre dans le module Business Intelligence.

Nous sommes satisfaites du résultat obtenu et espérons avoir atteint notre objectif initial.