Offres BI et Big Data

TP Final

Anasse ZOUGARH
Paul-Alexandre TESSIER

I. Introduction

Dans le cadre du module d'Offres BI et Big Data nous devons réalisés une l'analyse de l'activité d'une entreprise afin de dégager des indicateurs utiles à la prise de décision. Le travail est fait en binôme avec Talend et Qlik .

Le but du TP est donc de mettre en applications les notions vus en cours à travers un cas concret d'entreprise. Nous expliquerons donc par la suite les différentes fonctionnalités et résultats obtenus.

II. Identifier le besoin

1) Présentation du schéma de données

Nous devons traiter le cas de la société Northwind, son modèle du système de gestion SAGE se présente comme ceci :

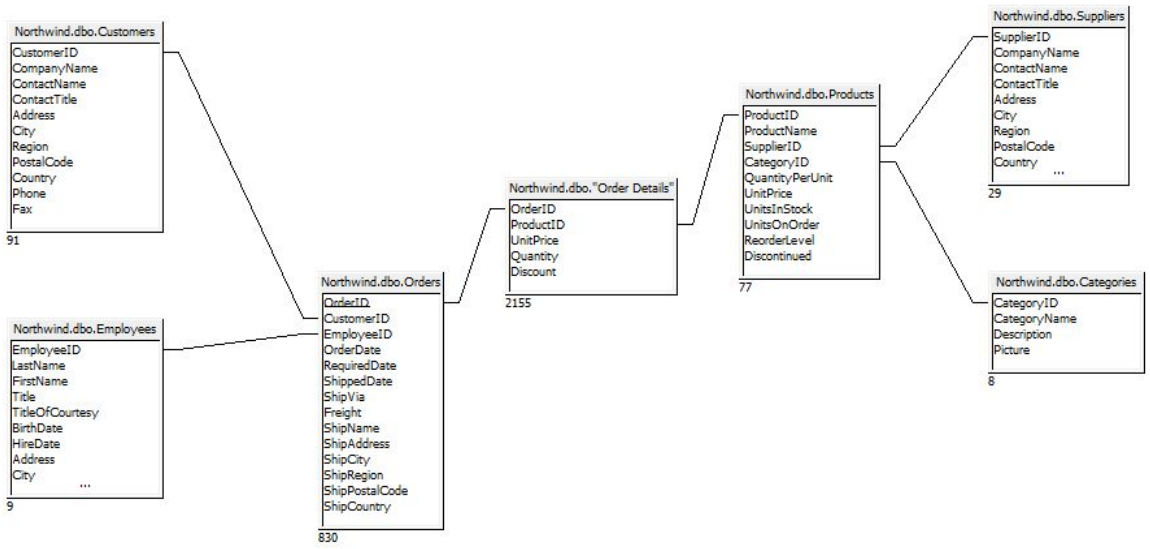


Schéma de donnée de l'entreprise

L'entreprise souhaite obtenir 4 informations :

- les familles de produits les plus rentables
- les employés qui gèrent le plus gros chiffres d'affaires
- les clients les plus fidèles pour mettre en place un programme de fidélité
- les remises accordées

2) Identification du besoin

Après avoir analysé le schéma nous pensons que ces tables sont celles qui nous seront le plus utile:

- Customers
- Employees
- Categories
- Orders
- Order Details
- Products

Nous n'avons pas besoin de la table avec les fournisseurs.

a) Dimensions

Nous avons identifié les dimensions suivantes :

- Temps (jour, mois, trimestre, année)
- Client
- Employé (par employé ou tranche d'age)
- Catégorie
- Produit
- Localisation (par pays de vente)

b) Indicateurs

Nous avons identifié les indicateurs suivants :

- Chiffre d'affaires
- Remise
- Quantités vendues

c) Est-il selon vous intéressant de garder le numéro de commande dans la table de faits? Pour quel besoin?

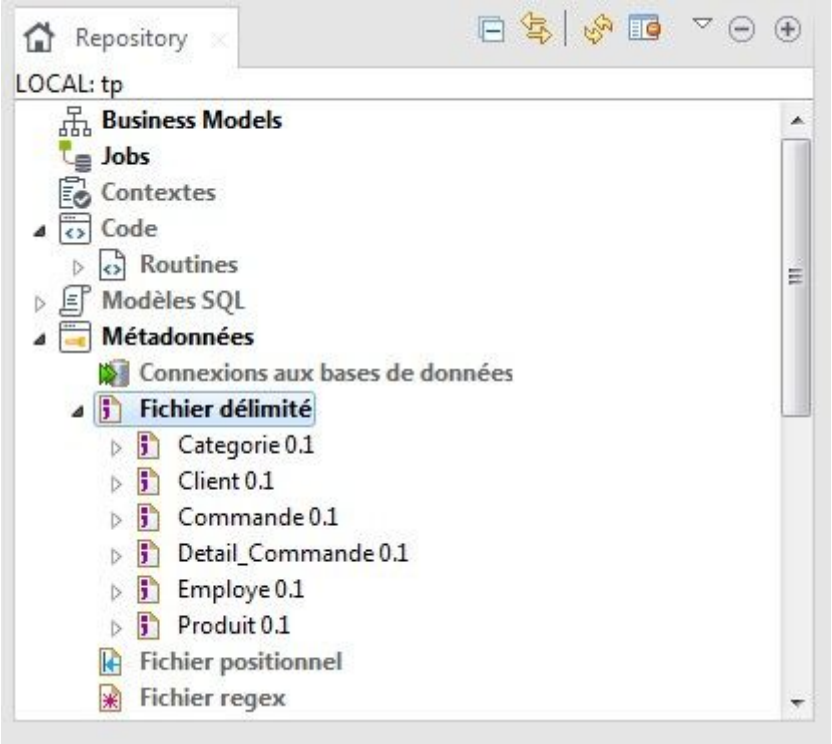
Oui, il faut ajouter à la table de fait le numéro de commande. Ainsi, on pourra, à partir de la table de fait récupérer les autres informations relative comme les informations sur le produit mais aussi quel employé à réalisé la vente, la catégorie du produit, etc..

Dans ce cas ci, il permettra de récupérer la date de commande et le pays de la commande, il nous permettra après de trier les données ou de les comparer dans le temps ou suivant la localisation.

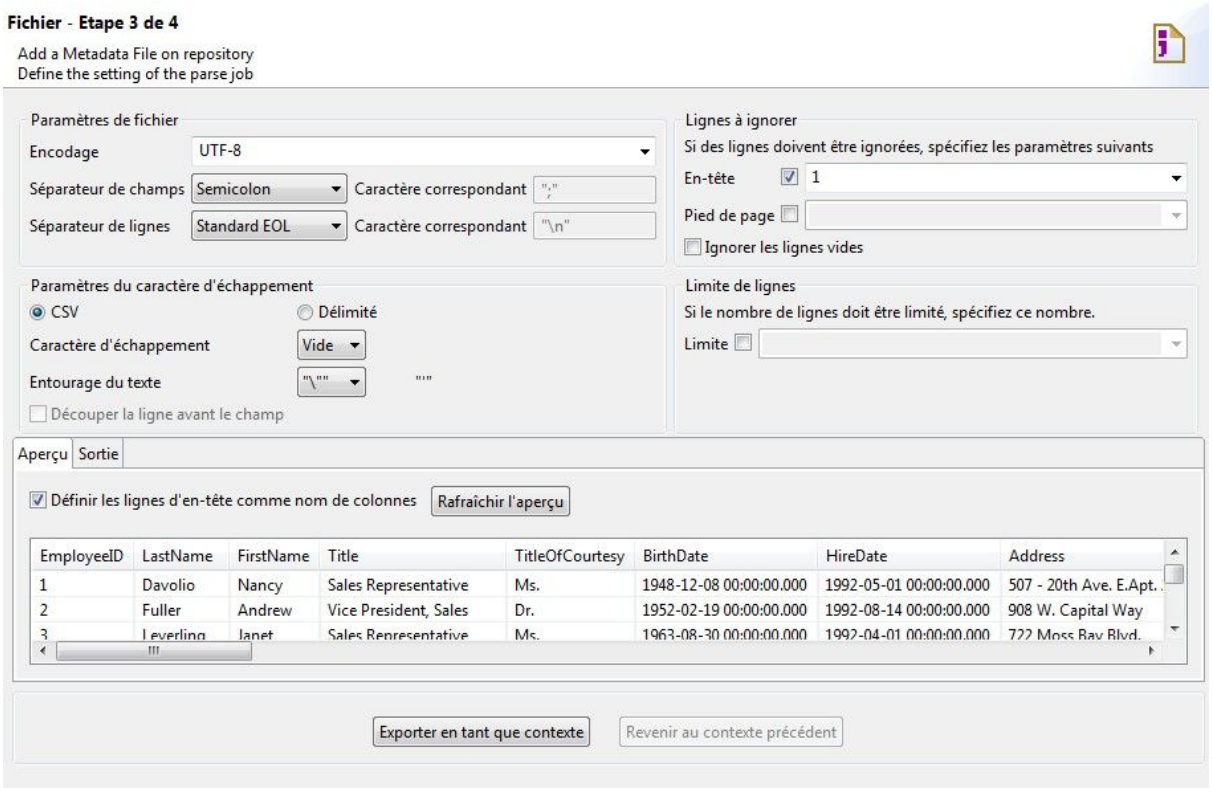
III. Développement Talend

1) Intégration des données

Pour pouvoir traiter les fichiers, il faut d'abord les ajouter dans Talend en tant que fichier délimité car il s'agit de CSV. Il est aussi l'occasion de les retraiter en cas de problèmes.



Un seul fichier a posé problème, il s'agit de l'intégration des données des employés. Un champ contenait des données textes avec des " qui contenait aussi des retours chariots. Ainsi le logiciel traitait cela comme une nouvelle ligne ce qui fausse complètement l'intégration. Il faut ainsi supprimer cet entourage de texte avec les paramètres suivant.



2) Traitement des fichiers

Pour traiter les fichiers et les transformer en tables utiles définies précédemment, il y a eu plusieurs particularités.

a) Les dates

Les dates sont représenté comme ceci “1996-07-08 00:00:00.000” ce qui pose problème car on ne souhaite pas les données de temps. Les données sont intégrées en tant que chaîne de caractère dans un premier temps. Puis nous avons fait passer les données dans une variable pour y appliquer les traitements suivants : *TalendDate.parseDate("yyyy-MM-dd",StringHandling.LEFT(row1.OrderDate,10))*. Cela se décompose en 2 fonctions :

- *StringHandling.LEFT* permet de ne garder que le nombre de caractère indiqué en partant de la gauche. Ainsi cela permet de ne garder que les données de date.
- *TalendDate.parseDate* permet de créer une date à partir d’une chaine de caractère avec le modèle prédéfini qui est l’année, le mois puis le jour.

Enfin, en sortie, il nous suffit de déterminer un champ de type date et de modifier le modèle pour qu’il corresponde au critère français soit "dd-MM-yyyy".

Petit problème que nous avons rencontré plus loin dans le sujet, il s’agit du moment où nous ne souhaitons extraire que le mois. La fonction getMonth donnait les mois de 0 a 11. Il a donc fallu refaire et ajouter un a ces valeurs.

b) Les tranches d'âges

Pour les employés, il est demandé de calculer l'âge et de déterminer les tranches d'âges associé. Pour cela on a utilisé 2 variables. La première permet d'afficher l'âge de l'employé et l'autre permet de déterminer la tranche d'âge.

Pour déterminer l'âge on réalise la commande suivante :

TalendDate.diffDateFloor(TalendDate.parseDate("yyyy-MM-dd", "1998-12-01"),TalendDate.parseDate("yyyy-MM-dd",StringHandling.LEFT(row1.BirthDate,10)) ,"yyyy")

Elle fonctionne comme la gestion des dates sauf qu'on ajoute la fonction *TalendDate.diffDateFloor* qui permet de faire la différence entre 2 dates et de récupérer le résultat sous un format de date. Ainsi on récupère la différence avec comme année de base le 1er décembre 1998 car dans la situation du sujet il s'agit de la fin de l'année 1998. On récupère le résultat en année uniquement grâce au 3eme paramètre qui permet de choisir.

Maintenant, il suffit de déterminer quel catégorie en fonction de l'âge, pour cela nous avons utilisé cette méthode *: Var.var1 < 30 ? "-30" : Var.var1 <= 45 ? "30-45" : "+45"*
Var.var1 correspond à l'âge précédemment calculé. Il s'agit d'un simple if imbriqué.

c) La table de fait

Pour réaliser la table de fait, il est nécessaire d'intégrer toute les tables utiles que nous avons retenu. L'intérêt ici est de récupérer dans chacune de ces tables leur identifiant unique qui seront présent dans la table de fait. Cela permettra, à partir d'une table de fait, d'accéder à toutes les autres données. Il faut aussi ajouter les indicateurs que nous avons identifié :

- Le chiffre d'affaires
- Le taux de remise
- La quantité vendu

Pour calculer le CA, nous avons eu des problèmes pour faire passer les valeurs en type double. Cela est en raison que le prix unitaire et la remise contiennent comme séparateur une virgule alors que dans Talend il faut que ce soit un point. Ainsi nous avons dans un premier temps traité les données en type double. Pour cela nous avons fait ce traitement : *Double.parseDouble(row1.UnitPrice.replaceAll(",", "."))*

Une fois les variables en type double, nous pouvons maintenant calculer le CA. Nous allons prendre le CA après remise, qui est calculé comme ceci : (double)Math.round(Var.prixU * (1-Var.reduction) * row1.Quantity * 100) / 100
Le réduction est en % donc on fait 1 - remise pour avoir le taux. Pour arrondir à 2 chiffres après la virgule on utilise une petite astuce, on multiplie par 100 puis on utilise la fonction round qui va supprimer toute les décimales restantes et arrondir. Ainsi il ne reste plus qu'a diviser par 100 pour avoir un nombre arrondi à 2 chiffres après la virgule.

3) Tables d'agrégats

Notre table d'agrégat contient les éléments suivants :

- nom_catégorie du produit
- id_employee pour trier par employé ou tranche d'âge
- le mois, le trimestre et l'année correspondant
- le pays de la vente
- le CA avec et sans remise car il est nécessaire pour la suite dans Qlik
- quantité vendues

Pour créer la table d'agrégat, nous avons utilisé les données de la table "Orders, "Order Details", "Products" et "Categories". Il était plus simple d'utiliser ces fichiers que les dimensions et la table de fait comme demandé.

Dans un premier temps, il n'est pas demandé d'ajouter le nom de la catégorie, cela donne 696 résultats sur les 2155 de la table de fait. Normalement, pour qu'une table d'agrégat soit utile et optimisé il faut qu'on ait 10x moins de résultats que la table de fait, ici on a 5x moins ce qui est pas mal mais encore optimisable.

Cependant, dans Qlik, il est demandé de faire une comparaison par catégorie de produit. La table d'agrégat actuelle ne permet pas cela. Nous avons donc choisi d'ajouter le nom de la catégorie de produit. Cela donne 1791 lignes. Forcément, cela alourdit la table d'agrégat mais il était indispensable de le rajouter pour pouvoir analyser par catégorie de produit par la suite.

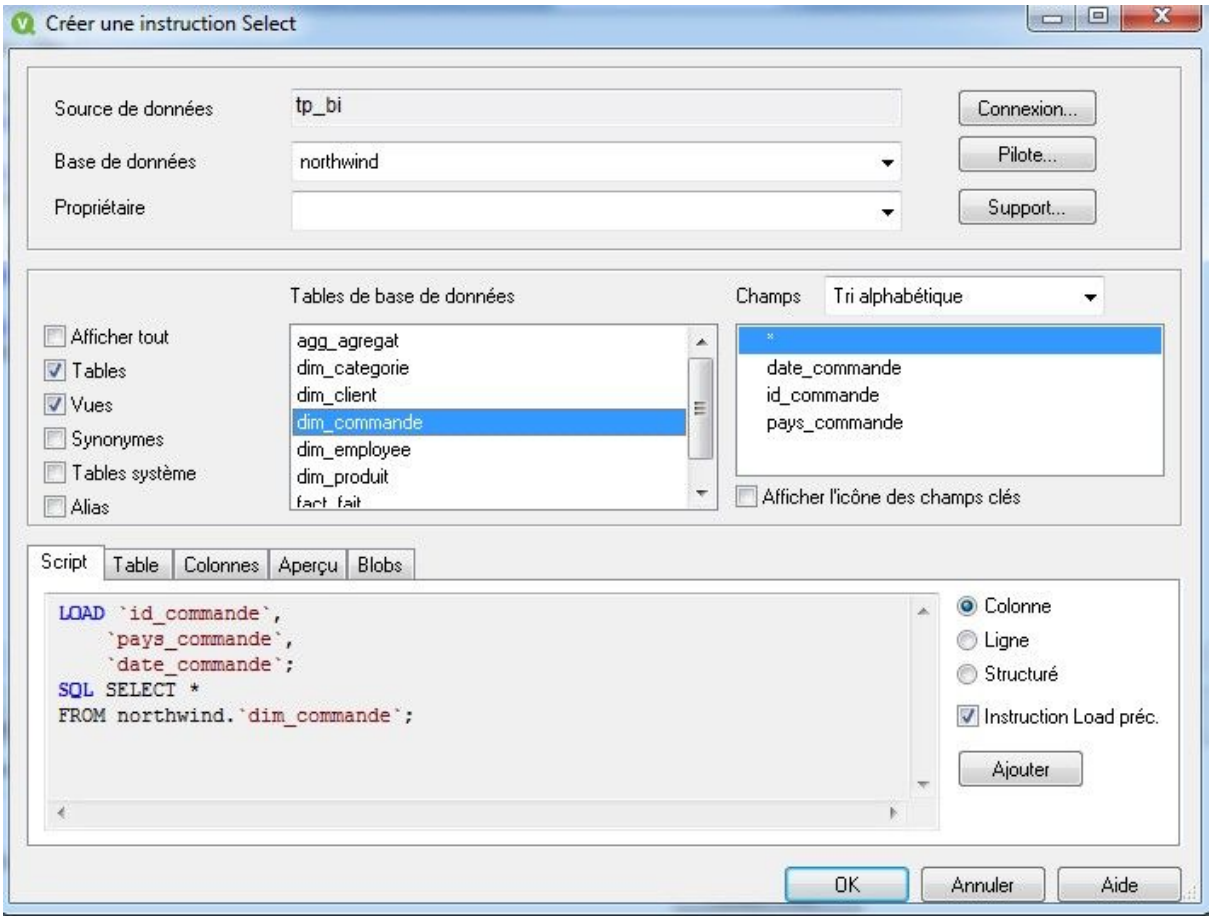
4) Sortie des données

Toutes les données traitées, dans les Jobs de Talend sont en sortie dans une base de donnée MySQL créé en local. L'objet tMysqlOutput est utilisé pour se connecter à celle ci et réaliser les opérations d'insertions. Il était possible autrement en exportant en fichier délimité ou en base Access.

IV. Développement Qlik

1) Ajout des tables utiles

Après avoir téléchargé l'ODCB pour Mysql, on peut réaliser la connection à la base de donnée situé en local. Une fois la connection réalisé on peut ajouter les tables utiles comme montré dans l'exemple :



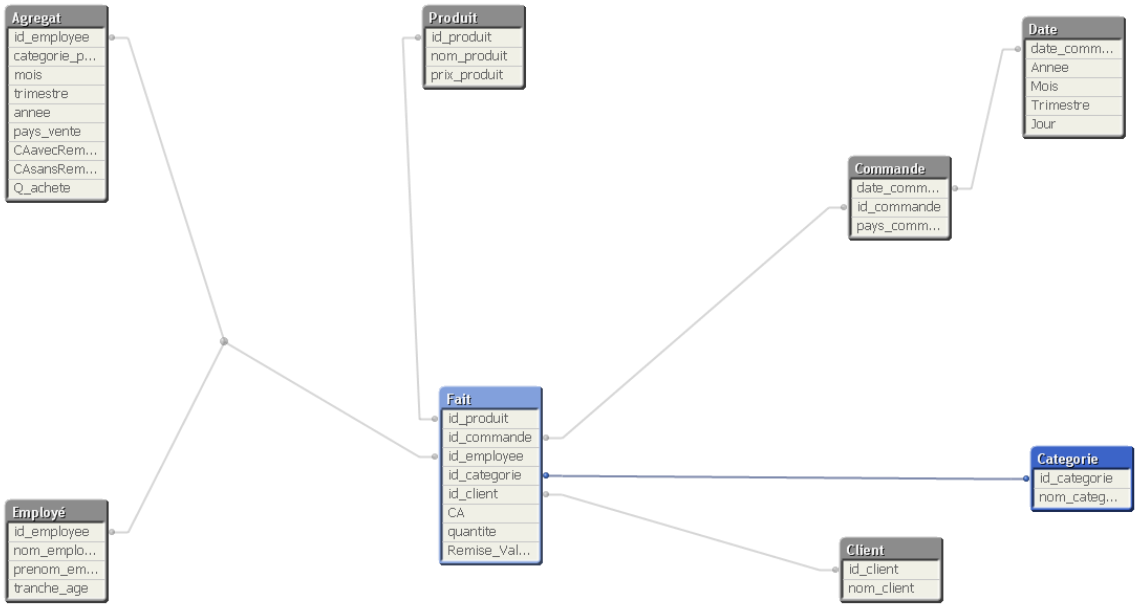
En cochant “Instruction Load préc.” on peut avoir la requête SQL déja créé. Une fois que nous avons ajoutés toutes les tables, il a fallut modifier le champ remise pour ne plus l’afficher en taux de remise mais en valeur décimale. Nous avons réalisé ceci comme cela :

```
Round(`prix_produit` * remise * quantite,0.01) AS Remise_Valeur;
```

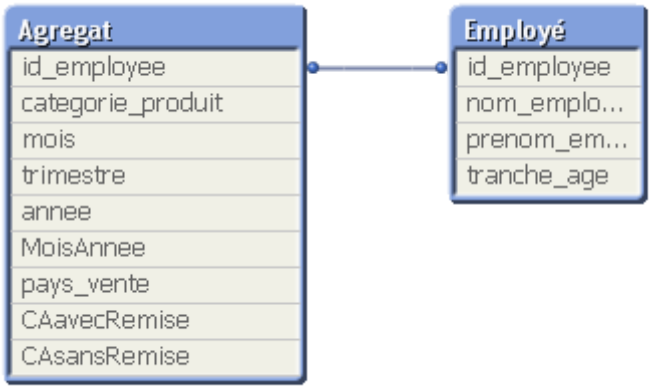
Round permet d’arrondir à un nombre après la virgule si on le spécifie, nous l’avons mis car il est demandé de le faire à de chiffres après la virgule.

2) Dimension de temps

Pour ajouter la dimension de temps, il était demandé de le faire avec une boucle. Malgré plusieurs essais avec AutoGenerate et While nous n’avons pas réussi. Nous avons préféré une autre solution, plus simple et tout aussi fonctionnelle. Nous avons simplement utilisé le champ date_commande et avec l’instruction “resident” on pouvait utiliser la table “Commande” précédemment chargé sans la recharger une seconde fois. Ainsi pour chaque commande on faisait une extraction de l’année, du mois et du jour. À partir du mois, on pouvait calculer le trimestre qui correspondait. Notre schéma se présente comme ceci pour le moment :



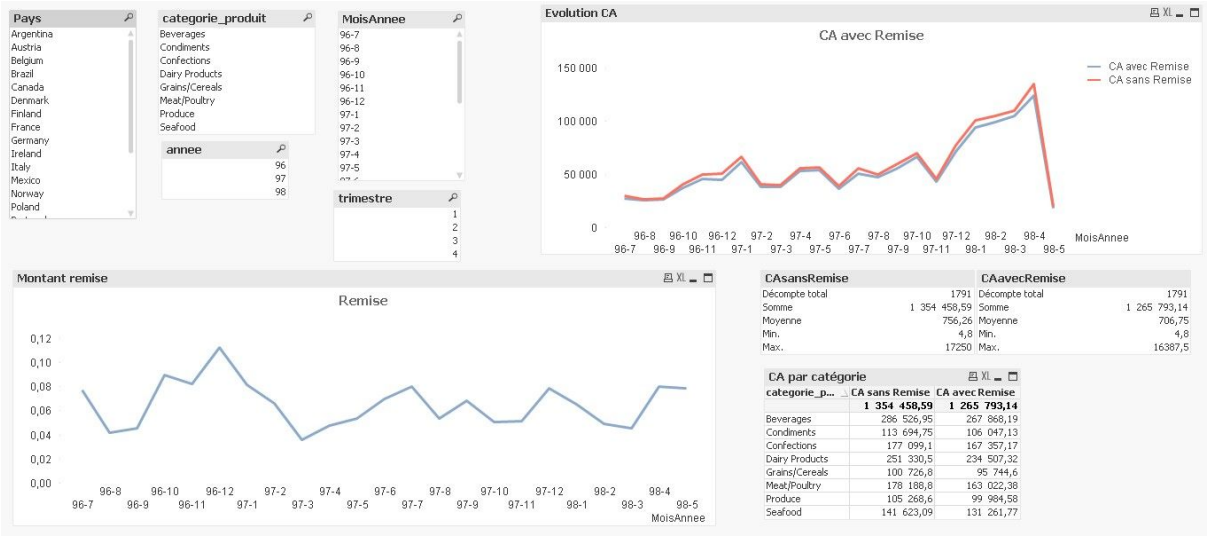
Pour lier les tables, nous avons chargé la date de commande dans la table dimension pour pouvoir la lier. Même chose avec la table d'agrégat, on lie avec l'id de l'employé. Cependant, ce schéma n'est pas adapté pour les analyses suivants. Quand on regarde les élements nécessaires pour la réalisation des tableaux et graphiques, on se rend compte que notre schéma peut etre simplifié comme ceci :



Toutes les informations nécessaires sont dans notre table d'agrégat. Il nous suffit de garder la table employé pour pouvoir avoir les tranches d'âges et les infos de l'employé. Nous avons ajouté un champ MoisAnnee qui est uniquement la concaténation du champ mois et annee, il va permettre d'afficher l'évolution dans le temps par mois. Si on ne le mettait pas, les données seraient additionnés par si elles avaient le même mois sans faire la différence des années.

3) Analyse de ventes

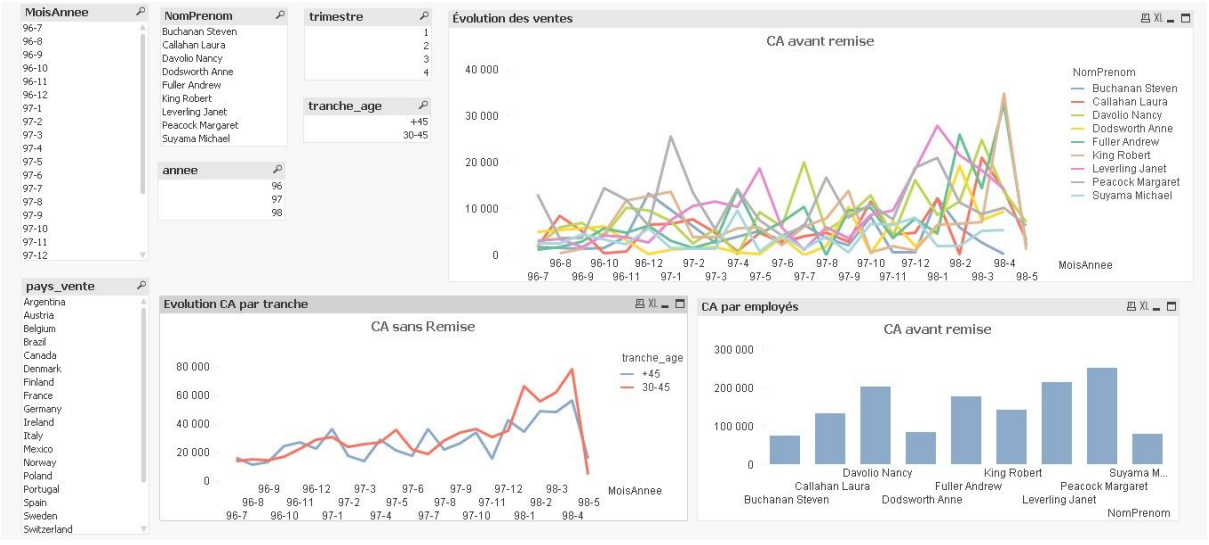
Notre page d'analyse de vente ressemble à ceci :



Avec en haut à gauche les différentes dimensions permettant d'affiner les données. On peut analyser par mois, trimestre, année, pays de vente et catégorie de produit. On a mis les chiffre d'affaires avec et sans remise dans le temps. Pour les 2 indicateurs, on a les informations comme le montant total et la moyenne. En bas à gauche, l'évolution du taux de remise dans le temps. Enfin, en bas à droite, les CA par catégories.

4) Ventes par employés

Notre tableau ressemble à ceci :



Nous avons mis l'évolution du CA dans le temps par mois et par employé. Le graphique est rapidement illisible, il faut donc sélectionner au maximum 3 employés pour plus de lisibilité.

En bas à droite, cela permet de voir le CA total par salariés, avec une sélection dans les dimensions on peut accéder aux données souhaités.

V. Conclusion

Malgré que nous ne connaissions rien a Talend et a Qlik avant ce semestre, ce sujet de TP nous a permi de prendre en main ces outils et de découvrir la BI. Après avoir traité les données dans Talend, nous avons pu créer des graphiques dans Qlik permettant une analyse dans le futur.