Une image contenant texte, clipart

Description générée automatiquement

**Projet Business Intelligence**

**ETL, Data Warehouse et Cube OLAP**

**Á l’attention de Madame I.Sboui**

Marie HOWET et Charlotte LIBERT

Hénallux – Catégorie technique – IG321 Business Intelligence – 2022-2023

Table des matières

[Contexte 3](#_Toc124847322)

[Solution 3](#_Toc124847323)

[Technologies et composants 3](#_Toc124847324)

[SSMS : SQL Server Management Studio 3](#_Toc124847325)

[SSIS : SQL Server Integration Services 4](#_Toc124847326)

[SSAS : SQL Server Analysis Services 6](#_Toc124847327)

[Diagrammes des tables de dimensions et de la table de fait 6](#_Toc124847328)

[Granularité de la table de fait 7](#_Toc124847329)

[Justifications des choix 7](#_Toc124847330)

[Clés primaires du modèle Data Warehouse 7](#_Toc124847331)

[Modélisation des dimensions et des faits 8](#_Toc124847332)

[DimDate 8](#_Toc124847333)

[DimEmployee 8](#_Toc124847334)

[DimSupplier 9](#_Toc124847335)

[DimShipper 9](#_Toc124847336)

[DimCustomer 10](#_Toc124847337)

[DimProduct 10](#_Toc124847338)

[FactSales 10](#_Toc124847339)

[Limites rencontrées 11](#_Toc124847340)

[Stratégie d’exécution de l’ETL 12](#_Toc124847341)

[Cube OLAP 13](#_Toc124847342)

[Reporting 15](#_Toc124847343)

# Contexte

Nous travaillons pour une SSII active dans le domaine de la BI et sommes envoyées en mission chez BlablaCo, groupe commercial gérant deux marques de produits alimentaires : Foodies997 et Zen1984.

Ces marques enregistrent toutes les commandes de produits par le biais de ses employés, commerciaux déployés géographiquement. En plus de ses commerciaux, chaque enseigne dispose d’un webshop proposant le même catalogue de produits que ceux proposés par les commerciaux.

Le groupe souhaite pouvoir tirer de l’information de toutes les données. Il souhaiterait disposer d’un outil simple à utiliser qui permettrait d’obtenir des réponses rapidement à des questions précises. Il souhaiterait pouvoir analyser les données sur plusieurs axes, faire des recoupements. Pour le moment, le groupe s’intéresse particulièrement aux données produites par Foodies997. Celles de Zen1984 suivront dans un futur proche.

Notre système doit être conçu de manière à absorber facilement l’intégration de Zen1984 dans le futur. Le métier souhaiterait également que le multilinguisme soit supporté. Les données doivent être consultable en Anglais et en Français.

# Solution

A partir de la base de données Foodies997 qui nous est fournie, nous proposons 2 outils de BI qui permettent de répondre au besoin métier : le Data Warehouse et le Cube OLAP.

Voici de brèves définitions des structures que nous allons utiliser afin de réaliser notre solution :

* ETL (Extract, Transform, Load) : est une technologie qui est capable à partir d’une/plusieurs sources de données (souvent une base de données relationnelle) d’extraire les données brutes, de les transformer (càd enrichir, nettoyer, agréger les données) puis de les charger dans une nouvelle base de données tel qu’un Data Warehouse par exemple.
* Data Warehouse : est un entrepôt de données (souvent une base de données dimensionnelle) issu d’un processus ETL dont le but est de faciliter la prise de décision dans une entreprise.
* Cube OLAP : est une structure de données multidimensionnelle qui permet une analyse rapide des données en fonction des multiples dimensions qui définissent un problème commercial.

# Technologies et composants

Pour réaliser notre solution, nous avons utilisé différents outils de Microsoft SQL Server.

### SSMS : SQL Server Management Studio

SQL Server Management Studio est un outil intégré permettant d'accéder, de développer, d'administrer et de gérer les bases de données SQL Server ainsi que construire des requêtes et des scripts à l'aide d'une interface graphique.

Ici, nous avons utilisé SSMS pour gérer la BD d’origine Foodies997 ainsi que la BD de destination DataWarehouse.

Une image contenant table

Description générée automatiquement Une image contenant table

Description générée automatiquement

Figure 1- Bases de données sur SSMS

SSMS peut également être utilisé pour accéder, configurer, gérer et administrer des "Analysis services", utile pour le déploiement de notre Cube OLAP.

### SSIS : SQL Server Integration Services

Microsoft SSIS ou SQL Server Integration Services est un outil de migration et d'intégration de données fourni avec la base de données Microsoft SQL Server qui peut être utilisé pour extraire, intégrer et transformer des données. SSIS est une solution d'extraction, de transformation et de chargement (ETL).

Ici, nous avons utilisé le logiciel Visual Studio 2019 supportant SSIS pour déployer notre Data Warehouse.

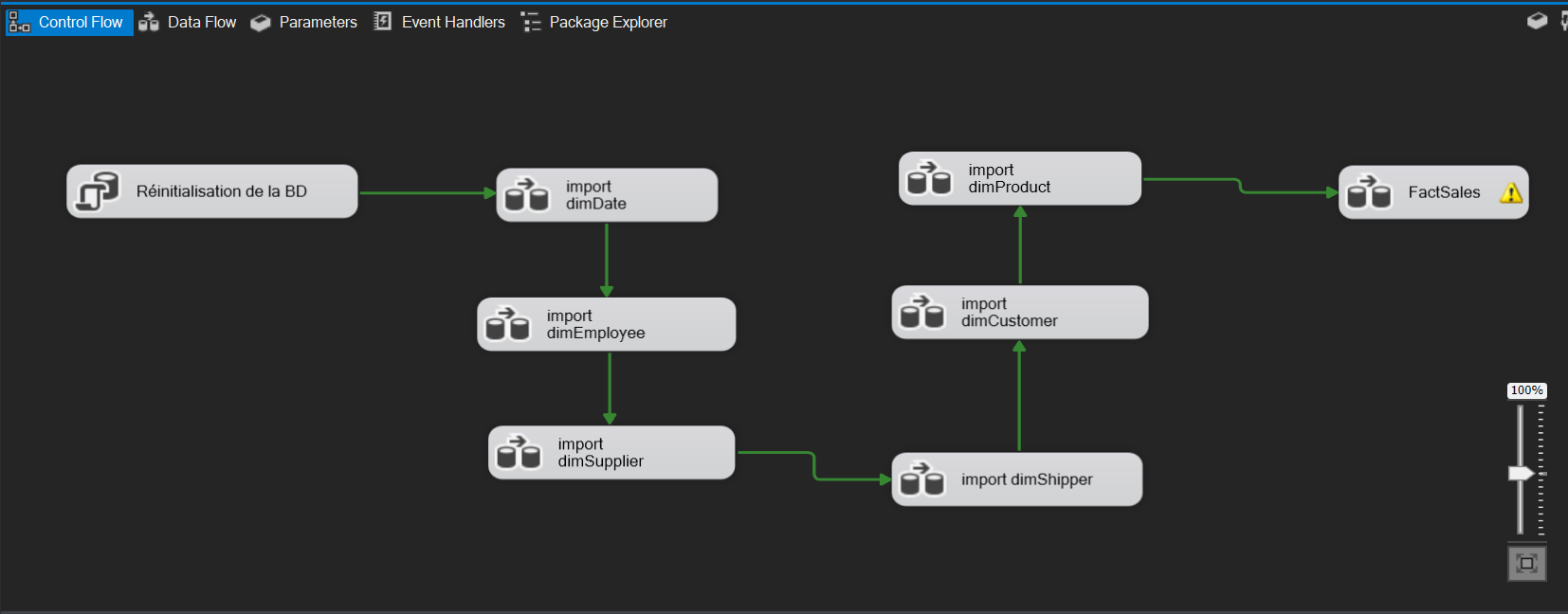


Figure 2 - Le composant Control Flow sur Visual Studio 2019

Composants :

Une image contenant texte

Description générée automatiquementTous les composants sont présents dans la boite à outils SSIS « SSIS Toolbox ».

Figure 3- Boite à outils SSIS

Le package principale de notre projet SSIS sur Visual Studio 2019 (voir ci-dessus) se compose d’un flux de contrôle « Control Flow », lui-même composé de plusieurs flux de données « Data Flow ». Tous les composants « Import dim… » et le composant « FactSales » sont des flux de données.

Nous avons ajouté un composant de tâche d’exécution des requêtes SQL « Execute SQL » qui exécute une requête qui remet à 0 chaque table de la base de données de destination lorsqu’on (re)démarre l’ETL.

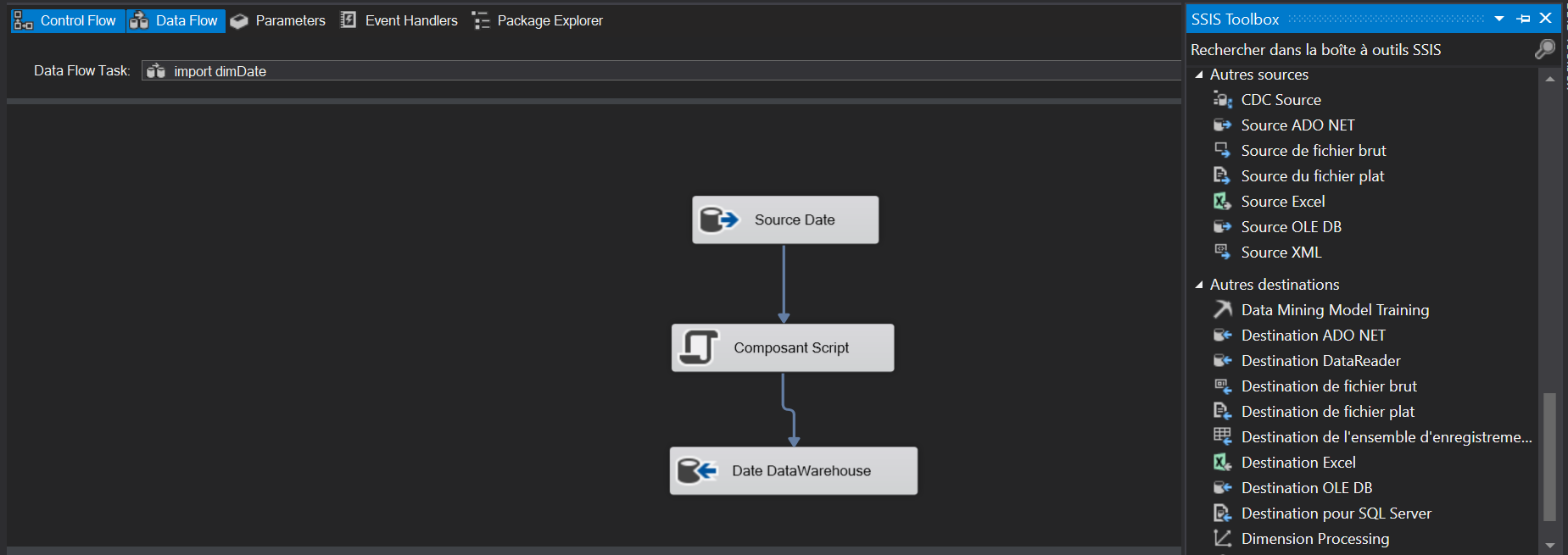


Figure 4-Exemple d'un Data Flow

Chaque flux de données pour les dimensions sont composés au minimum d’un composant source OLE DB « OLE DB Source » qui permet d’accéder aux données de la base de données source et d’un composant destination OLE DB « OLE DB Destination » qui permet d’injecter les nouvelles données (transformées ou non) dans la base de données de destination. Ici, dans la plupart des dimensions, nous avons besoin de transformer les données avant de les injecter dans la nouvelle BD. C’est pour cela, que nous utilisons un composant script « Script Component » en mode Transformation.

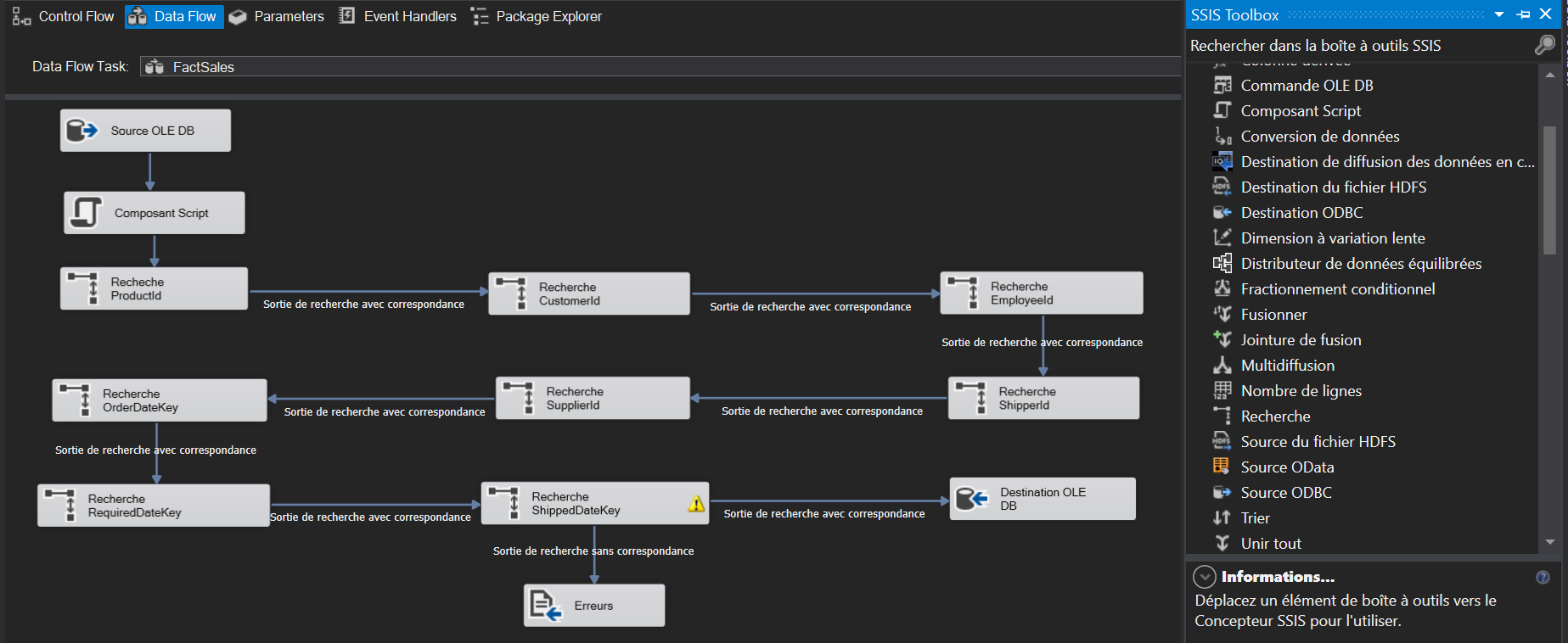


Figure 5-Les différents composants du flux de données de FactSales

Le flux de données pour la table de fait est composé des mêmes 3 composants que dans les autres flux de données. Nous avons rajouté également des composants de recherche « Lookup » afin de vérifier qu’on retrouve bien pour chaque clé étrangère, la référence vers la clé primaire de la table de dimension correspondante. Dans le cas contraire, s’il n’y a pas de correspondante, on insère ces données dans un fichier plat de destination « Flat File Destination » afin d’observer les éventuelles erreurs lors de l’exécution.

### SSAS : SQL Server Analysis Services

SQL Server Analysis Services (SSAS) est un serveur OLAP multidimensionnel ainsi qu'un moteur d'analyse qui permet de découper de grands volumes de données. Il permet aussi d'effectuer des analyses en utilisant diverses dimensions. Il existe deux variantes : multidimensionnelle et tabulaire. Nous utilisons ici la variante multidimensionnelle.

Ici, nous avons utilisé à nouveau le logiciel Visual Studio 2019 supportant SSAS pour déployer notre Cube OLAP.

# Diagrammes des tables de dimensions et de la table de fait

La Data Warehouse se base sur un modèle dimensionnel car la redondance est acceptée. Le but d’une telle structure est d’accélérer le traitement des requêtes et de faciliter la compréhension pour un utilisateur métier. Nous avons décidé d’utiliser un modèle STAR pour représenter nos tables de dimensions et notre table de fait.

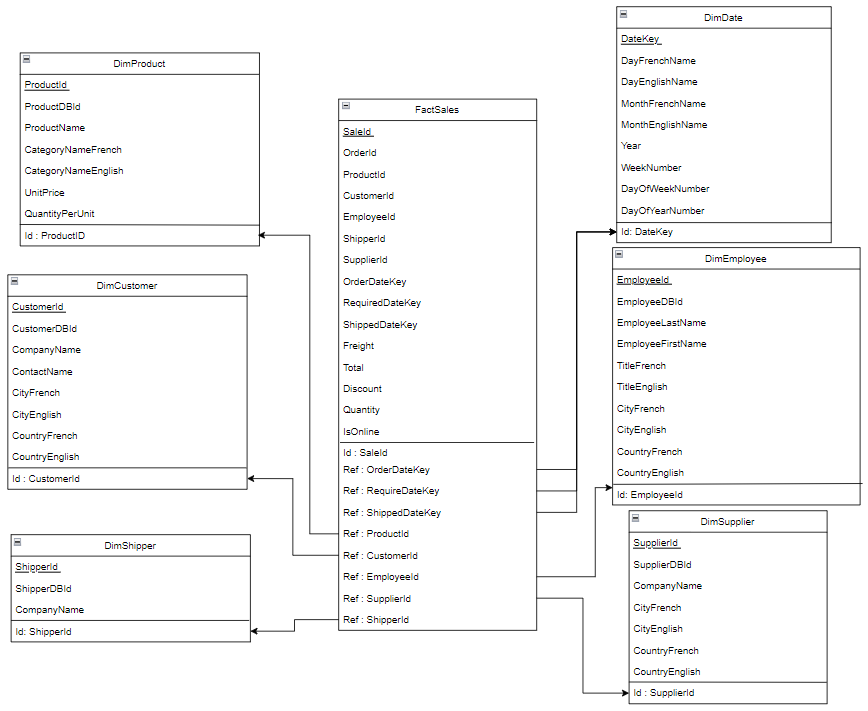


Figure 6-Modélisation dimensionnelle Star

# Granularité de la table de fait

Pour la table de fait FactSales, nous avons choisi de travailler sur chaque ligne de commande car le besoin métier était de pouvoir analyser les ventes par critère temporel, par produit et catégorie et par les caractéristiques des clients qui ont acheté des produits. Nous considérons qu’une vente est un produit acheté en un ou plusieurs exemplaires : c’est-à-dire, une ligne de commande.

# Justifications des choix

## Clés primaires du modèle Data Warehouse

Pour éviter les collisions entre les clés primaires si une nouvelle base de données s’ajoutait à notre Data Warehouse, nous avons décidé de ne pas réutiliser les mêmes clés primaires que celles de base de données d’origine. Afin de respecter cette contrainte, chaque table (exceptée DimDate) dispose d’une clé primaire auto-incrémentée indépendante de la base de données d’origine.

## Modélisation des dimensions et des faits

Comme nous l’avons mentionné ci-dessus, chaque table de dimensions (exceptée DimDate) a une clé primaire auto-incrémentée. Mais, pour faciliter la transition vers le Data Warehouse, nous avons décidé de garder les identifiants de base de la table Foodies997 dans chaque table de dimensions.

Que ce soit pour les tables de dimensions et/ou la table de fait, nous avons gardé les attributs de la BD Foodies997 qui nous semblaient les plus pertinents pour répondre aux questions du métier.

### DimDate

Seules les dates dont nous avons besoin dans la table de fait sont enregistrées dans la table DimDate, c’est-à-dire OrderDate, RequiredDate et ShippedDate de la table Orders dans la BD Foodies997.

Une image contenant texte

Description générée automatiquement

Figure 7-Requête SQL du composant source OLE DB

Donc, à partir de ces dates, nous avons pu remplir les attributs dans la table DimDate en transformant les dates en fonction du type de chaque attribut. Nous avons les attributs suivants : DayFrenchName (nom du jour en français), DayEnglishName (nom du jour en anglais), MonthFrenchName (nom du mois en français), MonthEnglishName (nom du mois en anglais), Year (année), WeekNumber (numéro de la semaine), DayOfWeekNumber (numéro du jour de la semaine), DayOfYearNumber(numéro du jour de l’année). En ce qui concerne la clé primaire DateKey, celle-ci se base sur la concaténation des 4 chiffres de l’année suivi des 2 chiffres du mois suivi des 2 chiffres du jour.

Une image contenant texte

Description générée automatiquementUne image contenant texte

Description générée automatiquement

Figure 8-Script de transformation dans le composant script

### DimEmployee

En plus des identifiants, nous avons gardé les attributs LastName et FirstName de la table Employees dans la BD Foodies997 dans EmployeeLastName, EmployeeFirstName. Comme Title, City et Country peuvent être à *null* dans la BD d’origine, nous leur avons donné des valeurs par défaut grâce au composant script.

En ce qui concerne Title, City et Country, pour répondre à la demande du métier, nous avons traduit Title, City et Country en français dans TitleFrench, CityFrench et CountryFrench et en anglais dans TitleEnglish, CityEnglish et CountryEnglish.

Une image contenant texte

Description générée automatiquement

Figure 9-Script de transformation dans le composant script

### DimSupplier

En plus des identifiants, nous avons gardé CompanyName de la table Suppliers de la BD Foodies997.

Comme City et Country peuvent être à *null* dans la BD d’origine, nous leur avons donné des valeurs par défaut grâce au composant script.

En ce qui concerne City et Country, pour répondre à la demande du métier, nous avons traduit City et Country en français dans CityFrench et CountryFrench et en anglais dans CityEnglish et CountryEnglish.

Une image contenant texte

Description générée automatiquement

Figure 10-Script de transformation dans le composant script

### DimShipper

En plus des identifiants, nous avons gardé l’attribut CompanyName de la table Shippers dans la BD Foodies997.

### DimCustomer

En plus des identifiants, nous avons gardé les attributs CompanyName et ContactName de la table Customers dans la BD Foodies997. Comme ContactName, City et Country peuvent être à *null* dans la BD d’origine, nous leur avons donné des valeurs par défaut grâce au composant script. En ce qui concerne City et Country, pour répondre à la demande du métier, nous avons traduit City et Country en français dans CityFrench et CountryFrench et en anglais dans CityEnglish et CountryEnglish

Une image contenant texte

Description générée automatiquement

Figure 11-Script de transformation dans le composant script

### DimProduct

En plus des identifiants, nous avons gardé les attributs ProductName, UnitPrice et QuantityPerUnit de la table Products dans la BD Foodies997. Comme UnitPrice et QuantityPerUnit peuvent être à *null* dans la BD d’origine, nous leur avons donné des valeurs par défaut grâce au composant script.

En ce qui concerne CategoryName, pour répondre à la demande du métier, nous avons traduit CategoryName en français dans CategoryNameFrench et nous gardons CategoryName de base dans CategoryNameEnglish.

Pour récupérer les catégories, nous avons fait une requête SQL de jointure entre la table Products et la table Categories dans la BD Foodies997.

Une image contenant texte

Description générée automatiquement

Figure 12-Script de transformation dans le composant script

### FactSales

Nous avons gardé les attributs OrderId, Freight, Discount, Quantity, isOnline des tables Orders et Order Details de la BD Foodies997.

Comme CustomerId, EmployeeId, OrderDate, RequiredDate, ShippedDate et Freight peuvent être à *null* dans la BD d’origine, nous avons donné des valeurs par défaut aux attributs qui leur sont associés grâce au composant script.

Une image contenant texte

Description générée automatiquement

Figure 13-Script de transformation dans le composant script

Pour récupérer les données dont nous avions besoin dans la table FactSales, nous avons fait une requête SQL de jointure entre la table Orders, la table Order Details et la table Products dans la BD Foodies997.

Une image contenant texte

Description générée automatiquement

Figure 14-Requête SQL du composant source OLE DB

# Limites rencontrées

1. Le multilinguisme

Une des premières limites rencontrées étaient la question du multilinguisme. En effet, excepté la table DimDate où nous avons pu traduire les jours et les mois grâce à la classe C# CultureInfo, pour les autres tables qui nécessitaient une traduction, nous n’avons pas trouvé d’autres solutions C# permettant de faire la traduction automatiquement. C’est pourquoi, pour les tables DimProduct, DimSupplier, DimCustomer et DimEmployee, nous avons dû effectuer les traductions manuellement. Néanmoins, cette méthode n’est pas viable sur le long terme et ne permettra pas d’accueillir une nouvelle base de données sans modifier l’ETL.

1. Les questions du métier

Nous ne pourrons pas répondre aux questions prenant en compte la superficie de ventes des clients car nous ne disposons pas de cette information dans la BD d’origine.

1. Les clés primaires d’origine

Nous avons dû garder les clés primaires de la plupart des tables d’origine car nous en avions besoin pour trouver les correspondances entre les clés étrangères de la table de fait et des tables de dimensions. En effet, les clés primaires de notre BD sont automatiquement générées et indépendante de la BD d’origine.

# Stratégie d’exécution de l’ETL

Afin d’éviter tout problème lors de l’exécution de l’ETL, nous avons mis au point différents mécanismes de sécurité.

1. Le composant de réinitialisation de la BD de destination

* **Remise à 0 de la base de données de destination**

Dans le cas où l’ETL s’arrêterait durant le traitement et qu’il faille la relancer, on exécute en premier lieu un script présent dans le composant de tâche d’exécution de requêtes.

Dans ce script, nous supprimons toutes les données présentes dans les tables de dimensions et de fait afin de répartir d’une base de données vierge lors de la (ré-)exécution de l’ETL.

* **Valeurs par défaut**

Comme certaines clés étrangères pouvaient être à *null* dans la table FactSales, nous avons décidé d’ajouter à chaque table de dimension une ligne avec des valeurs par défaut sur laquelle on peut faire pointer les clés étrangères qui seraient à *null*.

Une image contenant texte

Description générée automatiquement

Figure 15-Requête SQL dans le composant Execute SQL

1. La gestion des attributs facultatifs (à *null*) de la BD d’origine

Dans les composants script en mode Transformation, nous avons inséré des valeurs par défaut dans le script C# si les données d’origine étaient à *null* (voir le point sur la modélisation des dimensions et des faits). Par exemple, pour les attributs de type chaine de caractères, nous insérons ‘Inconnu/Unknown’ si c’est *null* et pour ceux de type nombre, nous insérons ‘0’.

1. Les composants « Lookup » et le fichier plat

Dans le flux de données de la table de fait « FactSales », nous avons pour chaque clé étrangère fait des recherches pour trouver les correspondances avec les clés primaires de l’ancienne BD et ainsi récupérer les clés primaires des tables de dimensions. Si aucune correspondance n’est trouvée, les lignes sans correspondance sont enregistrées dans un fichier plat.

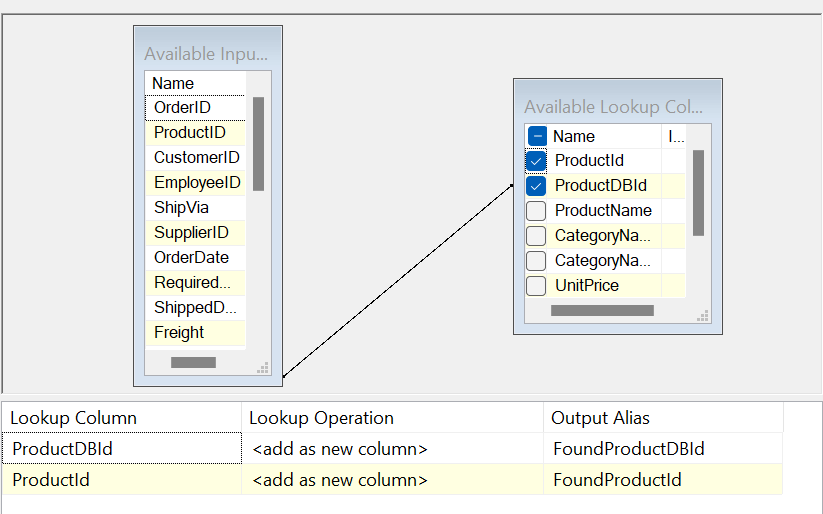


Figure 16-Exemple de mapping dans le composant Lookup

# Cube OLAP

À partir de notre ETL, nous avons réalisé un cube OLAP, qui nous sera utile pour faciliter nos recherches dans notre table de faits. Voici la structure de celui-ci :

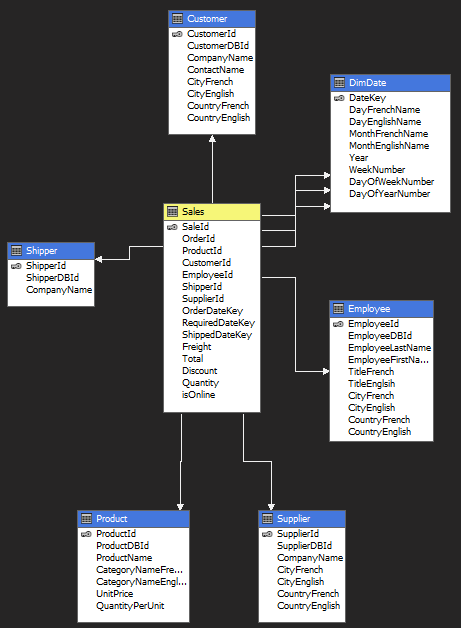


Figure 17-Structure du Cube OLAP

Notre cube multidimensionnel pour le reporting des ventes est composé de 8 dimensions :

Une image contenant texte

Description générée automatiquement

Figure 18-Dimensions du cube

Les structures des dimensions Required date, Order date et Shipped date sont basées sur celle de dimDate.

Une image contenant texte

Description générée automatiquement

Figure 19-Exemple de la dimension OrderDate

Nous avons également différentes mesures de la table Sales pour pouvoir effectuer nos recherches :

Une image contenant texte

Description générée automatiquement

Figure 20-Mesures de la table Sales

# Reporting

À l’aide de notre cube et de notre table FactSales, nous sommes à présent capable de répondre à des questions telles que celles-ci :

* Quel est la société de livraison qui a acheminé le plus grand nombre de colis sur une période donnée ?

Voici, par exemple, ce que nous obtenons pour une période allant du 4 juillet 1996 au 14 août 1996. Nous faisons intervenir dans cette requête la colonne « Company Name » de la dimension « Shipper », ainsi que la mesure « Sales nombre » de notre table Sales de notre cube.

Une image contenant texte

Description générée automatiquement

Figure 21-Requête à l’aide de l’interface graphique

* Quel commercial vend le plus de produits d’une catégorie X ?

Nous utilisons ici la colonne « Category Name French » de la dimension Product, la colonne « Employee Last Name » de la dimension « Employee » et la mesure « Sales Nombres » de notre table Sales.

Une image contenant table

Description générée automatiquement

Figure 22- Requête à l’aide de l’interface graphique

* Le top 5 des produits les plus vendus en ligne

Pour répondre à cette question, nous interrogeons notre table FactSales dans notre DataWarehouse via la requête ci-dessous (nous questionnons la table via SQL server par facilité, mais ces données sont tout à fait disponible dans notre table Sales de notre cube) :

Une image contenant texte

Description générée automatiquement

Une image contenant table

Description générée automatiquement

Figure 23-Requête SQL

* Pour une période donnée, vend-t-on plus en ligne ou par les commerciaux ?

Pour cette question , on peut interroger la table FactSales via une requête sur SQL server (les données sont également disponibles dans le cube) :

Une image contenant texte

Description générée automatiquement

Une image contenant table

Description générée automatiquement

Figure 24-Requête SQL

* Les montants vendus répartis par catégorie de produit

Pour cette requête, nous faisons intervenir les colonnes « Category Name French » et « Category Name English » de la dimension Product, ainsi que les mesures « Sales nombre » et « Total » de la table Sales du cube.

Une image contenant table

Description générée automatiquement

Figure 25-Résultats d'une requête

* Les clients disposant disposant d’’une grande superficie de vente achètent-ils plus de produits que les autres ?

Le cube que nous avons créé ne nous permet pas de répondre à cette question, étant donné que nous ne disposons pas, dans les informations de la base de données initiales, d’informations sur la superficie de de vente dont disposent les clients.