



Document de conception – Projet ODE

|  |  |  |
| --- | --- | --- |
| **Suivi des versions** | | |
| **Date** | **Version** | **Objet de la version** |
| 08/07/2015 | 01 | [Olivier] Création du document |
|  |  |  |
|  |  |  |

Sommaire

[Introduction 3](#_Toc424116529)

[Aspects métier 3](#_Toc424116530)

[Quelques chiffres 3](#_Toc424116531)

[Structure organisationnelle 3](#_Toc424116532)

[Architecture de la chaine décisionnel 3](#_Toc424116533)

[Vue d’ensemble 3](#_Toc424116534)

[Partie ETL 3](#_Toc424116535)

[Partie DWH / Cubes 3](#_Toc424116536)

[Partie Reporting 3](#_Toc424116537)

[Base opérationnelle 3](#_Toc424116538)

[Vue d’ensemble 3](#_Toc424116539)

[Table UNIVERS\_PRODUITS 3](#_Toc424116540)

[Table RAYONS\_PRODUITS 3](#_Toc424116541)

[Table FAMILLES\_PRODUITS 3](#_Toc424116542)

[Table SOUS\_FAMILLES\_PRODUITS 3](#_Toc424116543)

[Table PRODUITS 3](#_Toc424116544)

[Table CLIENTS 3](#_Toc424116545)

[Table LIEUX 3](#_Toc424116546)

[Table VILLES 3](#_Toc424116547)

[Table STOCKS 3](#_Toc424116548)

[Table VENTES 3](#_Toc424116549)

[Table TICKETS 3](#_Toc424116550)

[Table PRIXPRODUITS 3](#_Toc424116551)

[Entrepôt de données 3](#_Toc424116552)

[Vue d’ensemble 3](#_Toc424116553)

[Table de dimension CATEGORIES 3](#_Toc424116554)

[Table de dimension PRODUITS 3](#_Toc424116555)

[Table de dimension TEMPS 3](#_Toc424116556)

[Table de dimension LIEUX 3](#_Toc424116557)

[Table de dimension CLIENTS 3](#_Toc424116558)

[Table de dimension VILLES 3](#_Toc424116559)

[Table de faits VENTES 3](#_Toc424116560)

[Table de faits STOCKS 3](#_Toc424116561)

# Introduction

Le projet « Optimisation des Données de l’Entrepôt (ODE) » consiste à utiliser les techniques mathématiques vues dans le Master 2 MIAGE afin de construire l'entrepôt de données (Data-Warehouse - DWH) de manière optimal, en termes de temps de réponse à l’interrogation des cubes et d’occupation disque.

# Aspects métier

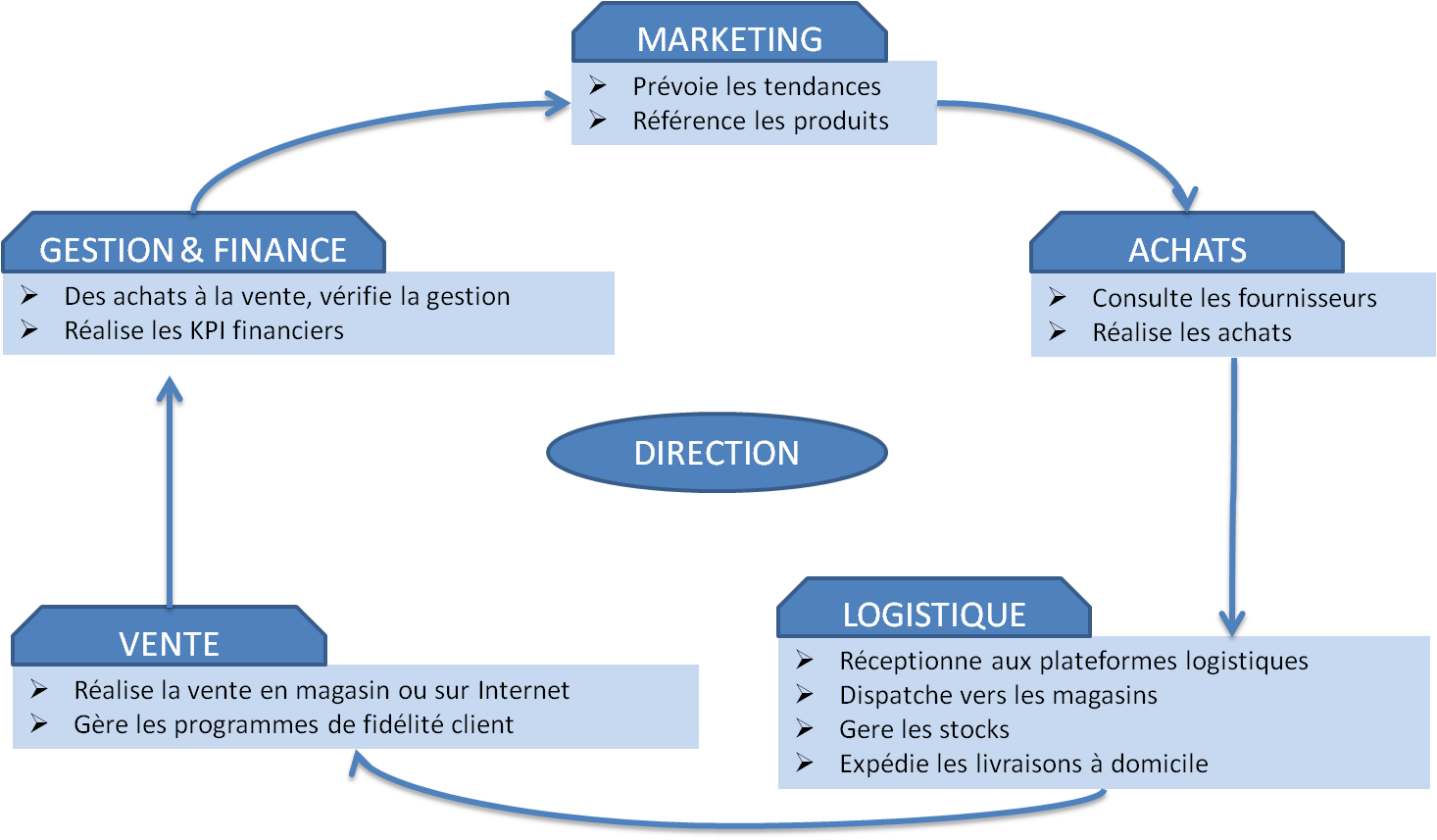
CASTO-MERLIN : Grande distribution de bricolage

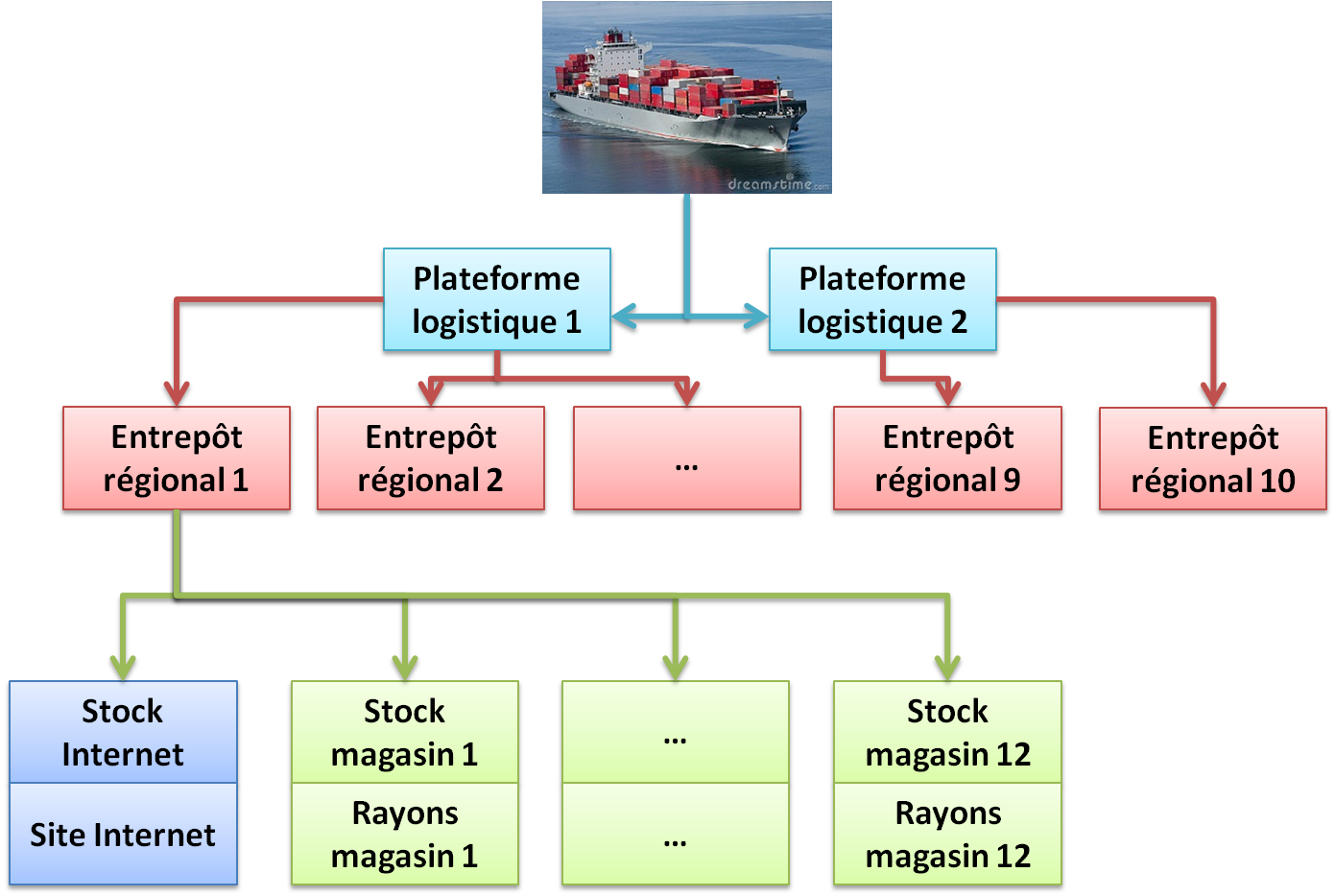


## Quelques chiffres

* 31 300 collaborateurs
* 2 plateformes logistiques nationales
* 10 centrales de distribution régionales
* 121 magasins + 1 site Internet
* 60 000 références magasins plus 20 000 « sur commande »
* 3 856 fournisseurs

## Structure organisationnelle





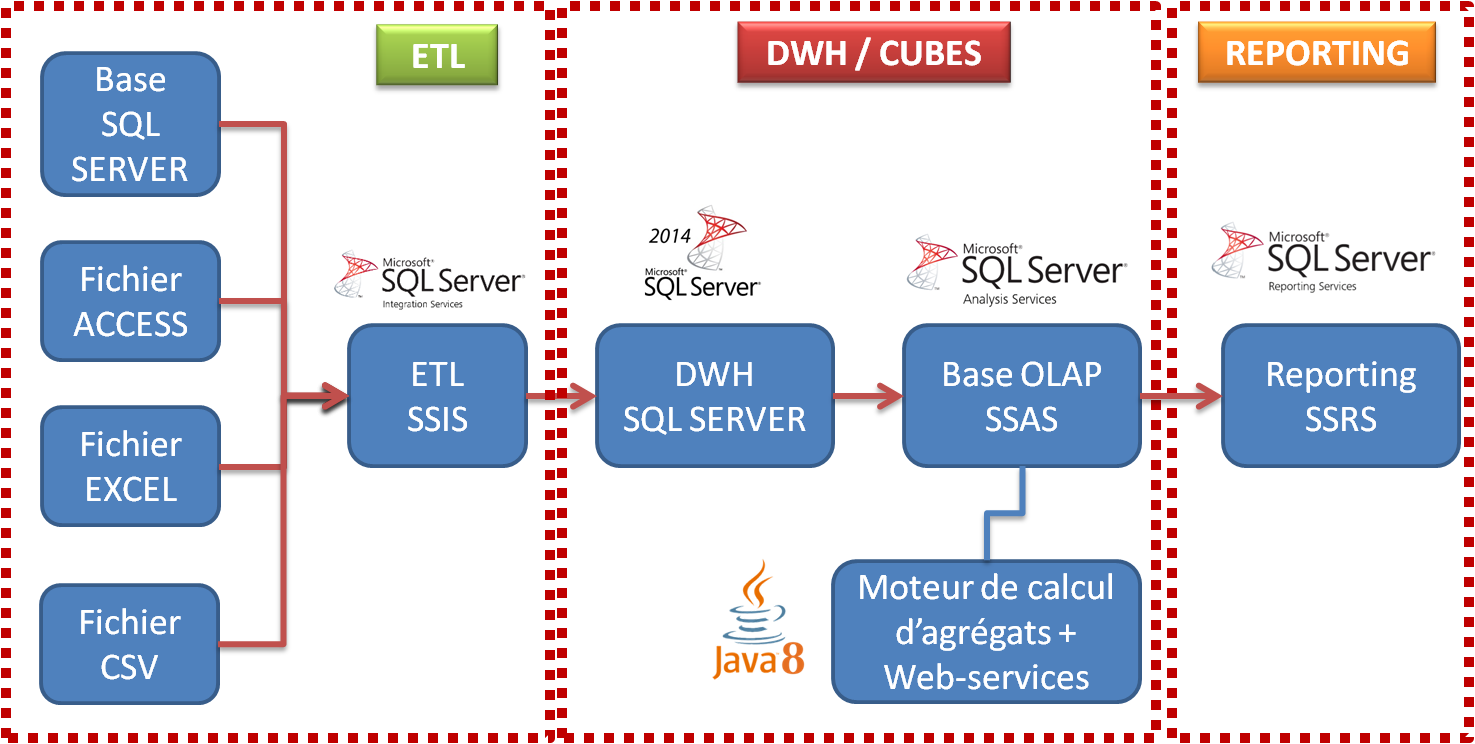
# Architecture de la chaine décisionnel

## Vue d’ensemble

Les données opérationnelles de l’entreprise, c'est-à-dire celle générées et utilisés pour la gestion de l’activité courante de l’entreprise, sont stockées principalement dans une base OLTP SQL Server 2014. Nous la désignerons par « base opérationnelle » dans la suite de ce document.

D’autres sources de données sont possibles :

* Fichiers « plats » type CSV
* Fichiers EXCEL
* Fichiers ACCESS



## Partie ETL

L’ETL sera en charge de collecter les informations à partir des différentes sources de données opérationnelles de l’entreprise :

* Base opérationnelle
* Fichiers CSV
* Fichiers EXCEL
* Fichiers ACCESS

Cette partie sera réalisée avec **SQL Server 2014 – Integration Services (SSIS)**

## Partie DWH / Cubes

C’est le « cœur » de la chaine décisionnelle que nous allons réaliser.

Le Datawarehouse (DWH – Entrepot de données) est modélisé en flocon, et hébergé sur une base OLTP **SQL Server 2014.**

Le cube sera hébergé sur une base OLAP **SQL Server 2014 Analysis Service (SSAS)**

Le moteur de calcul d’agrégats est un programme Java servant à « optimiser » la structure et le calcul des cubes (Agrégats, cf. cours du D111)

## Partie Reporting

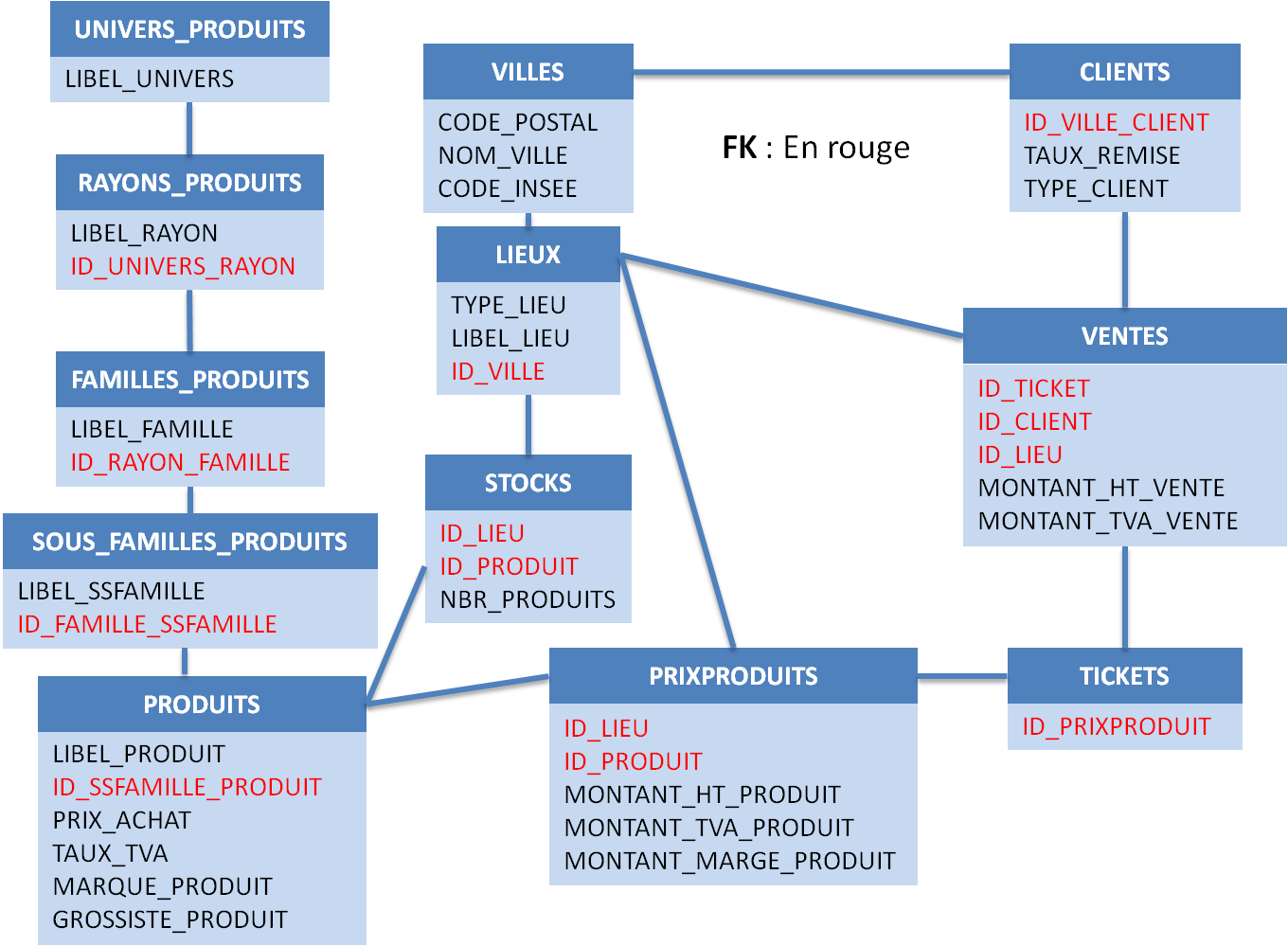
La chaine décisionnelle génère des reports sur un sujet « métier ». Par exemple : Calcul du CA, de la marge, évolution des ventes…

Cette partie sera réalisée avec **SQL Server 2014 – Reporting Services (SSRS)**

# Base opérationnelle

## Vue d’ensemble

Dans la base dédiée « **BaseOperationelleODE** », nous créons un schéma dédié « **ODE\_VENTES** », qui contient les tables suivantes :



## Table UNIVERS\_PRODUITS

But

D

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| ID\_UNIVERS | int | Non |
| LIBEL\_UNIVERS | nvarchar(256) | Non |
| DATE\_CREAT | datetime | Non |
| OPER\_CREAT | nvarchar(64) | Non |
| DATE\_MODIF | datetime | Oui |
| OPER\_MODIF | nvarchar(64) | Oui |

Remplissage

/\* A COMPLETER LORS DU LOT 3 : ETL \*/

## Table RAYONS\_PRODUITS

But

/\* **OLIVIER** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| ID\_RAYON | int | Non |
| LIBEL\_RAYON | nvarchar(256) | Non |
| ID\_UNIVERS\_RAYON | int | Non |
| DATE\_CREAT | datetime | Non |
| OPER\_CREAT | nvarchar(64) | Non |
| DATE\_MODIF | datetime | Oui |
| OPER\_MODIF | nvarchar(64) | Oui |

Remplissage

/\* A COMPLETER LORS DU LOT 3 : ETL \*/

## Table FAMILLES\_PRODUITS

But

/\* **OLIVIER** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| ID\_FAMILLE | int | Non |
| LIBEL\_FAMILLE | nvarchar(256) | Non |
| ID\_RAYON\_FAMILLE | int | Non |
| DATE\_CREAT | datetime | Non |
| OPER\_CREAT | nvarchar(64) | Non |
| DATE\_MODIF | datetime | Oui |
| OPER\_MODIF | nvarchar(64) | Oui |

Remplissage

/\* A COMPLETER LORS DU LOT 3 : ETL \*/

## Table SOUS\_FAMILLES\_PRODUITS

But

/\* **OLIVIER** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| ID\_SSFAMILLE | int | Non |
| LIBEL\_FAMILLE | nvarchar(256) | Non |
| ID\_RAYON\_SSFAMILLE | int | Non |
| DATE\_CREAT | datetime | Non |
| OPER\_CREAT | nvarchar(64) | Non |
| DATE\_MODIF | datetime | Oui |
| OPER\_MODIF | nvarchar(64) | Oui |

Remplissage

/\* A COMPLETER LORS DU LOT 3 : ETL \*/

## Table PRODUITS

But

/\* **OLIVIER** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| ID\_PRODUIT | int | Non |
| LIBEL\_PRODUIT | nvarchar(256) | Non |
| DESC\_PRODUIT | nvarchar(1024) | Non |
| ID\_SSFAMILLE\_PRODUIT | int | Non |
| CODE\_BARRE\_PRODUIT | int | Non |
| PRIX\_ACHAT | money | Non |
| TAUX\_TVA | decimal(4, 1) | Non |
| MARQUE\_PRODUIT | nvarchar(256) | Non |
| GROSSISTE\_PRODUIT | nvarchar(256) | Oui |
| DATE\_CREAT | datetime | Non |
| OPER\_CREAT | nvarchar(64) | Non |
| DATE\_MODIF | datetime | Oui |
| OPER\_MODIF | nvarchar(64) | Oui |

Remplissage

/\* A COMPLETER LORS DU LOT 3 : ETL \*/

## Table CLIENTS

But

/\* **OLIVIER** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| ID\_CLIENT | int | Non |
| NOM\_CLIENT | nvarchar(256) | Non |
| TYPE\_CLIENT | char(1) | Non |
| DATE\_NAISSANCE | date | Non |
| DATE\_SOUSCRIPTION | date | Non |
| ID\_VILLE\_CLIENT | int | Non |
| LIBEL\_ADRESSE | nvarchar(256) | Oui |
| CODE\_CARTE\_FIDEL | nvarchar(32) | Oui |
| TAUX\_REMISE | decimal(4, 1) | Non |
| DATE\_CREAT | datetime | Non |
| OPER\_CREAT | nvarchar(64) | Non |
| DATE\_MODIF | datetime | Oui |
| OPER\_MODIF | nvarchar(64) | Oui |

Remplissage

/\* A COMPLETER LORS DU LOT 3 : ETL \*/

## Table LIEUX

But

/\* **OLIVIER** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| ID\_LIEU | int | Non |
| TYPE\_LIEU | char(1) | Non |
| LIBEL\_LIEU | nvarchar(256) | Non |
| ID\_VILLE | int | Non |
| DATE\_CREAT | datetime | Non |
| OPER\_CREAT | nvarchar(64) | Non |
| DATE\_MODIF | datetime | Oui |
| OPER\_MODIF | nvarchar(64) | Oui |

Remplissage

/\* A COMPLETER LORS DU LOT 3 : ETL \*/

## Table VILLES

But

/\* **OLIVIER** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| ID\_VILLE | int | Non |
| CODE\_POSTAL | nvarchar(6) | Non |
| CODE\_COMMUNE | tinyint | Non |
| CODE\_REGION | tinyint | Non |
| CODE\_DEPARTEMENT | tinyint | Non |
| CODE\_ARRONDISEMENT | tinyint | Non |
| CODE\_CANTON | tinyint | Non |
| NOM\_VILLE\_MAJ | varchar(64) | Oui |
| NOM\_VILLE\_MIN | varchar(64) | Oui |
| DATE\_CREAT | datetime | Non |
| OPER\_CREAT | nvarchar(64) | Non |
| DATE\_MODIF | datetime | Oui |
| OPER\_MODIF | nvarchar(64) | Oui |

Remplissage

/\* A COMPLETER LORS DU LOT 3 : ETL \*/

## Table STOCKS

But

/\* **OLIVIER** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| ID\_STOCK | int | Non |
| DATE\_RECENSEMENT | datetime | Non |
| OPER\_RECENSEMENT | nvarchar(64) | Non |
| ID\_LIEU | int | Non |
| ID\_PRODUIT | int | Non |
| NBR\_DISPO | int | Non |
| NBR\_DEFECTUEUX | int | Non |
| NBR\_RETOUR\_SAV | int | Non |
| DATE\_CREAT | datetime | Non |
| OPER\_CREAT | nvarchar(64) | Non |
| DATE\_MODIF | datetime | Oui |
| OPER\_MODIF | nvarchar(64) | Oui |

Remplissage

/\* A COMPLETER LORS DU LOT 3 : ETL \*/

## Table VENTES

But

/\* **OLIVIER** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| ID\_VENTE | int | Non |
| OPER\_VENTE | nvarchar(64) | Non |
| ID\_TICKET | int | Non |
| ID\_CLIENT | int | Non |
| ID\_LIEU | int | Non |
| MONTANT\_HT\_VENTE | money | Non |
| MONTANT\_TVA\_VENTE | money | Non |
| DATE\_CREAT | datetime | Non |
| OPER\_CREAT | nvarchar(64) | Non |

Remplissage

/\* A COMPLETER LORS DU LOT 3 : ETL \*/

## Table TICKETS

But

/\* **OLIVIER** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| ID\_TICKET | int | Non |
| ID\_PRIXPRODUIT | int | Non |
| QUANTITE | int | Non |
| DATE\_CREAT | datetime | Non |
| OPER\_CREAT | nvarchar(64) | Non |

Remplissage

/\* A COMPLETER LORS DU LOT 3 : ETL \*/

## Table PRIXPRODUITS

But

/\* **OLIVIER** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| ID\_PRIXPRODUIT | int | Non |
| OPER\_PRIXPRODUIT | nvarchar(64) | Non |
| ID\_LIEU | int | Non |
| ID\_PRODUIT | int | Non |
| MONTANT\_HT\_PRODUIT | money | Non |
| MONTANT\_TVA\_PRODUIT | money | Non |
| MONTANT\_MARGE\_PRODUIT | money | Non |
| DATE\_CREAT | datetime | Non |
| OPER\_CREAT | nvarchar(64) | Non |
| DATE\_MODIF | datetime | Oui |
| OPER\_MODIF | nvarchar(64) | Oui |

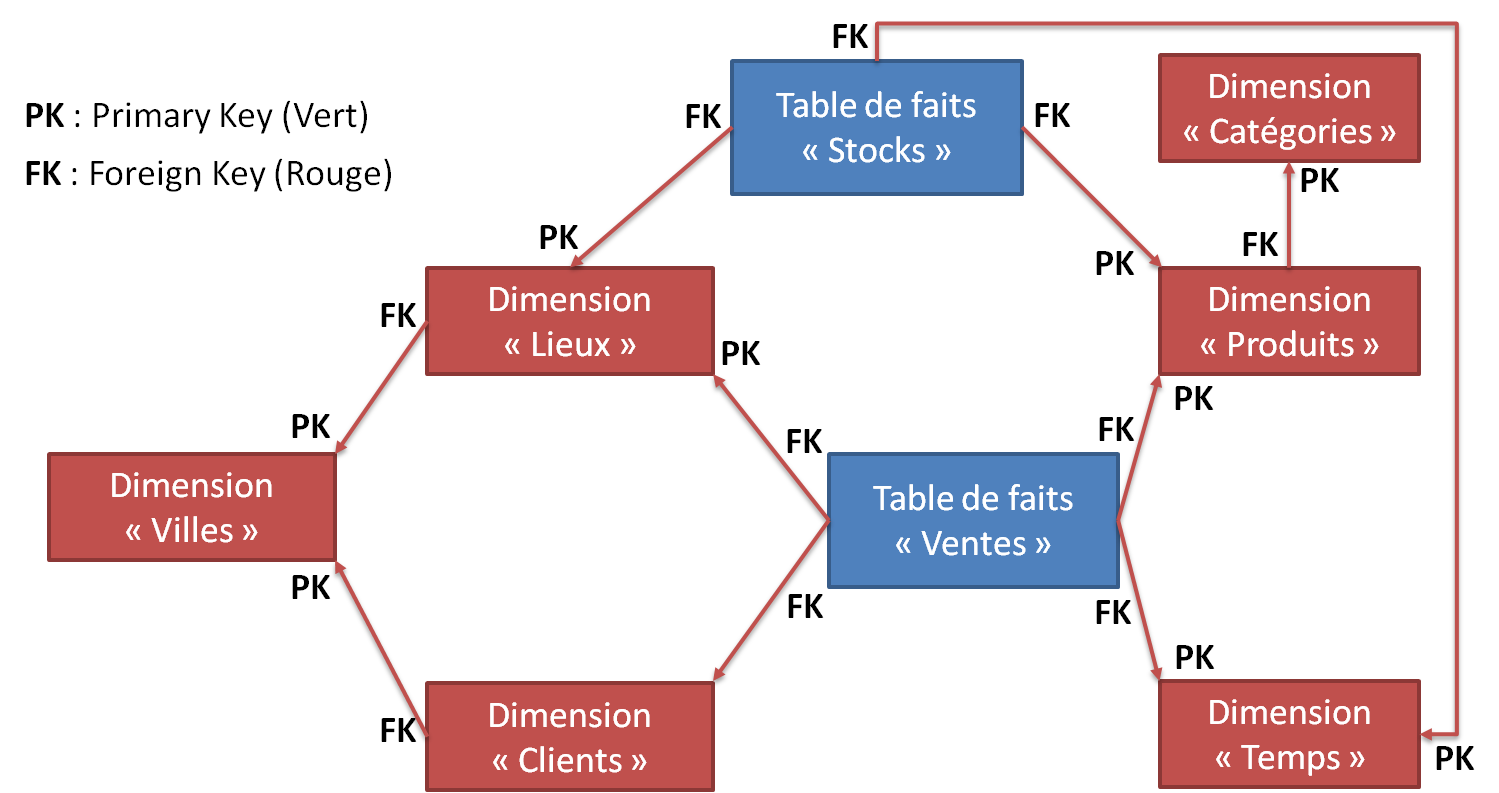
Remplissage

/\* A COMPLETER LORS DU LOT 3 : ETL \*/

# Entrepôt de données

## Vue d’ensemble

Dans la base dédiée « **DataWarehouseODE** », nous créons un schéma dédié « **ODE\_DATAWAREHOUSE** », qui contient les tables suivantes :



## Table de dimension CATEGORIES

But

/\***BRICE** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| CATEGORIE\_PK | int | Non |
| LIBEL\_UNIVERS | nvarchar(256) | Non |
| ID\_UNIVERS | int | Non |
| LIBEL\_RAYON | nvarchar(256) | Non |
| ID\_RAYON | int | Non |
| LIBEL\_FAMILLE | nvarchar(256) | Non |
| ID\_FAMILLE | int | Non |
| LIBEL\_SSFAMILLE | nvarchar(256) | Non |
| ID\_SSFAMILLE | int | Non |

**Clé primaire** : CATEGORIE\_PK (Clé technique)

**Clé étrangère** : ∅ (C’est une table de dimension en fin de branche du flocon)

Remplissage via l’ETL

/\* A COMPLETER LORS DU LOT 3 : ETL \*/

Remplissage pour les tests

/\***BRICE** : A COMPLETER \*/

## Table de dimension PRODUITS

But

La table PRODUITS est une table de dimension dont le but est de pouvoir analyser les ventes cet axe (ventes par marque, par fournisseur, par produit etc.) mais également de calculer certains indicateur de mesures grâce aux PRIX\_ACHAT et à au TAUX\_TVA notamment.

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| PRODUIT\_PK | int | Non |
| CATEGORIE\_FK | int | Non |
| LIBEL\_PRODUIT | nvarchar(256) | Non |
| PRIX\_ACHAT | money | Non |
| TAUX\_TVA | decimal(4, 1) | Non |
| MARQUE\_PRODUIT | nvarchar(256) | Non |
| GROSSISTE\_PRODUIT | nvarchar(256) | Oui |

**Clé primaire** : PRODUIT\_PK (Clé technique)

**Clé étrangère** : CATEGORIE\_FK → CATEGORIES.CATEGORIE\_PK

La table PRODUITS a également été dénormalisée en ajoutant une table CATEGORIE\_PRODUIT (cf. ci-après) toujours dans la même optique que pour la table VILLE : gagner en performance de mise à jour. En effet, la table PRODUITS sera sans doute mise à jour de manière très régulière alors que les CATEGORIE\_PRODUIT subiront sans doute moins de changements.

La table PRODUITS contient donc une clé primaire et une clé étrangère correspondant à la clé primaire de la table CATEGORIE\_PRODUIT ainsi que les champs :

* Libellé produit, par exemple : « Rouleau antigoutte L180 »
* Le prix d'achat du produit qui servira à calculer la mesure Marge.
* Le taux de TVA du produit qui servira également pour les indicateurs financiers.
* La marque du produit qui permettra par exemple d'effectuer des comparaisons entre marque sur un même type de produit et de réajuster les approvisionnements ensuite.
* Le fournisseur du produit qui pourra également permettre d'affiner la stratégie d'achat en regroupant les produits à forte demande auprès d'un seul fournisseur en négociant les prix par exemple.

Remplissage via l’ETL

/\* A COMPLETER LORS DU LOT 3 \*/

Remplissage pour les tests

Le peuplement de cette table s'est principalement basé sur des fichiers CSV qui ont ensuite permis de panacher les valeurs des champs de chaque produits : un fichier pour la spécification du modèle du produit (exemple : L180), un pour les marques des produits et un pour les grossistes.

Le choix de ces différents attributs s'est fait grâce à l'utilisation de la fonction T-SQL RAND() et une boucle while a ensuite permis d'insérer dans la table le nombre de lignes voulues.

## Table de dimension TEMPS

But

/\***OLIVIER** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| TEMPS\_PK | int | Non |
| DATE | date | Non |
| JOUR | nvarchar(16) | Non |
| ANNEE\_CODE | int | Non |
| ANNEE\_DATE | date | Non |
| ANNEE\_NOM | nvarchar(16) | Non |
| TRIMESTRE\_CODE | int | Non |
| TRIMESTRE\_DATE | date | Non |
| TRIMESTRE\_NOM | nvarchar(16) | Non |
| MOIS\_CODE | int | Non |
| MOIS\_DATE | date | Non |
| MOIS\_NOM | nvarchar(16) | Non |
| SEMAINE\_CODE | int | Non |
| SEMAINE\_DATE | date | Non |
| SEMAINE\_NOM | nvarchar(16) | Non |
| JOUR\_CODE | int | Non |
| JOUR\_DATE | date | Non |
| JOUR\_NOM | nvarchar(16) | Non |

**Clé primaire** : TEMPS\_PK (Clé technique)

**Clé étrangère** : ∅ (C’est une table de dimension en fin de branche du flocon)

Remplissage via l’ETL

/\* A COMPLETER LORS DU LOT 3 \*/

Remplissage pour les tests

/\***OLIVIER** : A COMPLETER \*/

## Table de dimension LIEUX

But

La table LIEUX est une table de dimension qui permettra d'analyser les ventes selon les différents lieux : type de lieu, ville d'implantation etc.

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| LIEU\_PK | int | Non |
| VILLE\_FK | int | Non |
| TYPE\_LIEU | char(1) | Non |
| LIBEL\_LIEU | nvarchar(256) | Non |
| DATE\_OUVERTURE | date | Non |
| DATE\_FERMETURE | date | Non |
| SURFACE\_M2 | numeric(6, 1) | Oui |

**Clé primaire** : LIEU\_PK (Clé technique)

**Clé étrangère** : VILLE\_FK → VILLES.VILLE\_PK

Nous avons choisis de relier la table LIEUX à la table VILLE par une clé étrangère représentant l'identifiant unique de la Ville. A noter que la table VILLE est également reliée à la table CLIENT puisque ces derniers habitent dans des villes : cette solution est cohérente comme l'indique Thomas Gauchet dans son livre SQL Server 2014 : Implémentation d'une solution de Business Intelligence. L'autre solution aurait été de créer deux tables : VILLES\_CLIENTS et VILLES\_LIEUX ou encore d'intégrer les informations de ces tables directement dans les tables LIEUX et CLIENTS. Notre choix s'est basé sur le critère de la limitation du nombre de tables (une seule table VILLE dans notre solution) tout en dénormalisant les informations des villes pour gagner en performance puisque celles-ci ne seront sans doute pas mise à jour très régulièrement en comparaison de la table CLIENTS.

En plus d'une clé primaire et d'une clé étrangère (vers la table VILLE), la table LIEUX contient les champs :

* Type de lieu, permettant d'identifier un lieu parmi Magasin, Entrepôt, Stock Magasin, Stock site Internet, Site Internet et Plateforme logistique à l'aide d'un caractère.
* Le libellé du lieu, par exemple : « Magasin de Talence ZAC (33) ».
* Les dates d'ouverture et de fermeture du magasin du magasin : pour permettre de comparer les performances des magasins en fonction de leurs "ancienneté".
* La surface du magasin qui nous permettra de matérialiser la performance des magasins en terme de CA/m² ou de Quantité vendues/m². Cet indicateur permettra de mesurer non seulement la performance d'un magasin (et éventuellement d'adapter les meilleurs techniques d'organisation pour les autres magasins) mais également de ramener le Chiffre d'Affaires et les quantité vendues à des ratios comparables entre établissements : c'est à dire pouvoir comparer un grand magasin de 1000m² de la banlieue bordelaise avec un magasin de 200m² du centre de la Creuse.

Remplissage via l’ETL

/\* A COMPLETER LORS DU LOT 3 \*/

Remplissage pour les tests

Pour le peuplement de la table, nous avons dans un premier temps fait le choix d'un remplissage semi-aléatoire de notre table. Dans les faits nous avons respecté le nombre de type de lieux décrits dans la description en introduction (2 plateforme logistique, 121 magasins...), en revanche l'implantation géographique de ces lieux a été faite de manière aléatoire. Cela signifie qu'en théorie, nous pourrons avoir un Entrepôt logistique isolé dans les DOM-TOM sans magasins à proximité. En effet, la contrainte de cohérence alourdirait fortement le travail de remplissage qui ne constitue pas le cœur de notre projet et qui ne serait d'aucune plus-value pour nos futurs projet d'entreprise : les magasins, les clients, les villes reflétant naturellement la cohérence de l'organisation de l'entreprise.

La génération des lieux a donc été réalisée en T-SQL grâce à la table CSV des villes pour choisir aléatoirement une ville d'implantation du lieux. Les autres valeurs ont été choisies grâce à l'utilisation de la fonction RAND() de T-SQL permettant de générer des nombres de manière aléatoire. Enfin une boucle while a permis de réaliser le nombre voulu d'insertions dans notre table.

## Table de dimension CLIENTS

But

/\***CEDRIC** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| CLIENT\_PK | int | Non |
| VILLE\_FK | int | Non |
| TAUX\_REMISE | decimal(6, 2) | Non |
| TYPE\_CLIENT | char(1) | Non |
| NOM\_CLIENT | nvarchar(256) | Non |
| DATE\_NAISSANCE | date | Non |
| DATE\_SOUSCRIPTION | date | Non |
| CODE\_FIDELITE | nvarchar(32) | Oui |

**Clé primaire** : CLIENT\_PK (Clé technique)

**Clé étrangère** : VILLE\_FK → VILLES.VILLE\_PK

Remplissage via l’ETL

/\* A COMPLETER LORS DU LOT 3 \*/

Remplissage pour les tests

/\***CEDRIC** : A COMPLETER \*/

## Table de dimension VILLES

But

/\***BERNARD** : A COMPLETER \*/

Le but de cette table de dimension est de permettre une analyse des ventes suivant l’axe Ville. Il est également possible de définir la hiérarchie suivante : **Region ->Departement->Arrondissement->Commune->Canton->Ville,** à l’effet de mener des analyses de ventes suivant plusieurs niveaux de vue de la table VILLE.

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| VILLE\_PK | int | Non |
| CODE\_POSTAL | nvarchar(6) | Non |
| CODE\_COMMUNE | int | Non |
| CODE\_REGION | int | Non |
| CODE\_DEPARTEMENT | int | Non |
| CODE\_ARRONDISEMENT | int | Non |
| CODE\_CANTON | int | Non |
| NOM\_VILLE\_MAJ | nvarchar(256) | Non |
| NOM\_VILLE\_MIN | nvarchar(256) | Non |
| POPULATION | int | Oui |

**Clé primaire** : VILLE\_PK (Clé technique)

Ici, les champs **CODE\_POSTAL, CODE\_COMMUNE, CODE\_REGION, CODE\_DEPARTEMENT, CODE\_ARRONDISEMENT, CODE\_CANTON**, considérés comme les clés étrangères n’ont pas les tables de dimensions associées. Nous avons fait le choix de ne pas développer le schéma en flocon pour la table VILLE.

--------------------------

Remplissage via l’ETL

/\* A COMPLETER LORS DU LOT 3 \*/

Remplissage pour les tests

/\***BERNARD** : A COMPLETER \*/

## Table de faits VENTES

But

/\***OLIVIER** : A COMPLETER \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| DATE\_VENTE\_FK | int | Non |
| PRODUIT\_FK | int | Non |
| CLIENT\_FK | int | Non |
| LIEU\_FK | int | Non |
| MONTANT\_HT\_VENTE | money | Non |
| MONTANT\_TVA\_VENTE | money | Non |
| MARGE\_BRUTE | money | Non |
| UNITES\_VENDUES | int | Non |
| NUM\_TICKET | nvarchar(256) | Non |

**Clé primaire** : ∅ (C’est une table de faits)

**Clés étrangères** :

* DATE\_VENTE\_FK → TEMPS.TEMPS\_PK
* PRODUIT\_FK → PRODUITS.PRODUIT\_PK
* CLIENT\_FK → CLIENTS.CLIENT\_PK
* LIEU\_FK → LIEUX.LIEU\_PK

Remplissage via l’ETL

/\* A COMPLETER LORS DU LOT 3 \*/

Remplissage pour les tests

/\***OLIVIER** : A COMPLETER \*/

## Table de faits STOCKS

But

/\* A COMPLETER SI VALORISATION ULTERIEURE \*/

Structure

|  |  |  |
| --- | --- | --- |
| **Nom colonne** | **Type** | **Nullable ?** |
| DATE\_INVENTAIRE\_FK | int | Non |
| PRODUIT\_FK | int | Non |
| LIEU\_FK | int | Non |
| NBR\_DISPO | int | Non |
| NBR\_DEFECTUEUX | int | Non |
| ID\_INVENTAIRE | int | Non |

**Clé primaire** : ∅ (C’est une table de faits)

**Clés étrangères** :

* DATE\_INVENTAIRE\_FK → TEMPS.TEMPS\_PK
* PRODUIT\_FK → PRODUITS.PRODUIT\_PK
* LIEU\_FK → LIEUX.LIEU\_PK

Remplissage via l’ETL

/\* A COMPLETER SI VALORISATION ULTERIEURE \*/

Remplissage pour les tests

/\* A COMPLETER SI VALORISATION ULTERIEURE \*/