Rapport sur l'alimentation du datawarehouse du projet "Fantastique" avec l'outil Pentaho

NF26 - A19

Xie Feier
Brizard Clément

4 avril 2019

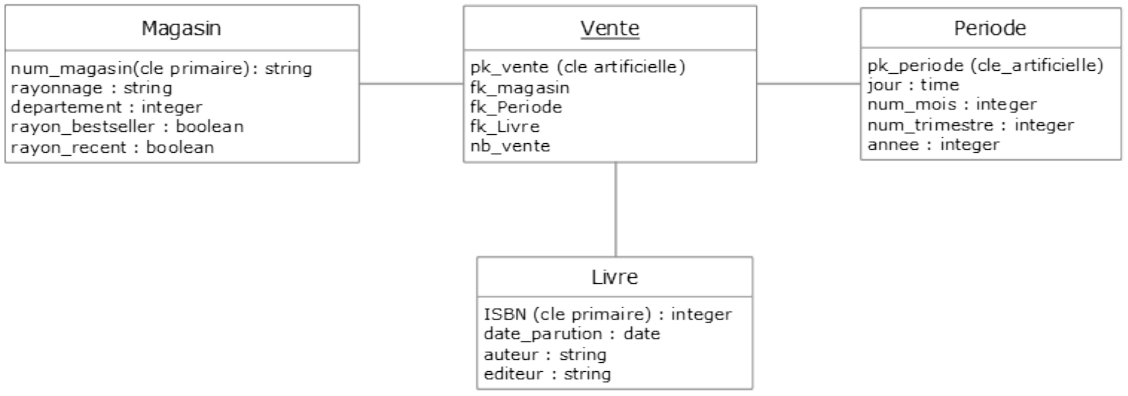
Introduction

Ce rapport présente les choix réalisés lors de la construction du datawarehouse du projet “Fantastique” avec l’outil ETL (*Extract, Transform, Load*) Pentaho.

Rappel du sujet

L’entreprise “Fantastique”, qui regroupe des magasins de ventes d’ouvrages littéraires répartis sur toute la France, souhaite faire une étude sur les ventes d’une année.
L’objectif est d’analyser les performances des ventes des magasins par la mesure du nombre de produits vendus en fonction d’un certain nombre de critères tels que les produits, les dates, les magasins, les départements, la population et les ratios significatifs de ces mesures.
Elle vous charge dans ce cadre de mettre en place une solution logicielle permettant d'intégrer les données pertinentes via un ETL et de pouvoir générer des rapports d’analyse ou des cubes via un logiciel de diffusion pour répondre aux besoins résumés ci-dessus.

Rappel du MCD



Dimension Magasin

Description

La dimension Magasin se caractérise par quatre propriétés : le numéro du magasin (ex : “M146”) qui l’identifie de manière unique, son rayonnage (ex : par auteur, éditeur, année de publication…), le fait que le magasin possède ou non un rayon “Best seller”, le fait qu’il possède ou non un rayon pour les “Livres Récents” et enfin son département (ex : “53” pour la Mayenne).

Construction

La construction de la dimension Magasin n’a pas posé de difficulté particulière. On trouvera dans le dossier de captures d’écran joint à ce rapport l’image `dim_magasin_etapes.png` qui synthétise les deux extractions, les transformations, puis le chargement dans la base de données Oracle.

Nous pouvons néanmoins nous arrêter sur deux étapes.
Tout d’abord, celle de jointure sur les numéros de département. Nous notons l’importance de trier préalablement les deux fichiers en fonction des départements pour minimiser le nombre de comparaisons deux à deux réalisées lors de la jointure.
Deuxièmement, à l’issue de la jointure, nous obtenons dans la table résultante deux colonnes qui portent la même information, à savoir le numéro de département. Le composant “Altération de flux” nous a permis de n’en garder qu’un des deux.

Justification des choix

Pour cette dimension, nous retenons un seul choix de conception. Le fichier `marketing.ods`, que nous avons extrait pour disposer des informations sur chaque magasin, contient deux colonnes associées à la même information : le rayonnage. Une portant le nom complet et une portant l’initiale (ex : “Author” et “A” pour un rayonnage par auteur). Nous choisissons de garder ces deux colonnes. D’une part, la colonne des initiales, permettra des opérations plus rapides, et pourra donc être utilisée pour les calculs. La colonne portant le nom complet du rayonnage sera quant à elle utilisée pour les affichages dans nos *reportings*, car son contenu est plus explicite.

Evaluation qualité et intégrité des résultats

Les statistiques des différentes transformations nous indiquent que nous avons bien traité tous les résultats (`dim_magasin_stats.png`) : il y avait bien 152 magasins dans le fichier `marketing.ods`, 99 départements dans le fichier des départements et le résultat de la jointure donne 152 lignes où le numéro de département de chaque magasin est associé avec le nom correspondant.

	Nom étape	N°Copie	Lignes lues	Lignes écrites	Lignes en entrées	Lignes en sortie
1	extraction_magasins	0	0	152	152	0
2	extraction_departements	0	0	99	99	0
3	tri_magasins	0	152	152	0	0
4	tri_departements	0	99	99	0	0
5	jointure	0	251	152	0	0
6	selection_champs	0	152	152	0	0
7	insertion_bdd	0	152	152	0	152

Image 1 : Statistiques de l'exécution des transformations pour la dimension Magasin

En visualisant la table après jointure (`dim_magasin_resultat.png`), nous pouvons vérifier l'association entre départements et constater que le tri a bien fonctionné.
Concernant la qualité des données, il n'y avait pas de données manquantes dans les deux fichiers extraits, et nous n'en avons pas perdu pendant la transformation. La seule vérification supplémentaire que nous aurions pu faire, aurait été de contrôler qu'il n'y avait pas d'erreur d'association numéro-nom de département dans le fichier des départements.

Dimension Date

Description

La dimension Date comprend les cinq colonnes (Day, Year, Month, DayOfWeek, Trimester), selon notre schéma en étoile. Ces différentes périodes offriront au client de plus larges possibilités de calcul des ventes de livres.

Construction

Nous nous appuyons sur un exemple de génération de dates pour construire notre propre table. Les enregistrements du fichier des ventes portent tous une date dont l'année est 2014. Nous créons donc une table de 365 lignes, chacune correspondant à un jour de l'année 2014 au format `YYYYMMDD`. On génère ensuite pour chaque jour, le nombre de jours le séparant du 01/01/2014. A partir de ces deux informations, on utilise le composant "Calculateur" afin d'obtenir pour chacun des 365 jours, l'année, le mois, le jour de la semaine et le trimestre correspondants (*cf.* `dim_date_calcul.png`). On obtient ainsi par exemple le trimestre à partir du jour au format `YYYYMMDD` :

	Nouveau champ	Calcul	Champ A	Champ B	Champ C	Type valeur	Longueur
1	Day	Date A + B Jours	START_DAY	Days_Since		Date	
2	Year	Année d'une date A	Day			Integer	4
3	Month	Mois d'une date A	Day			Integer	2
4	DayOfWeek	Jour de la semaine d'uneA	Day			String	1
5	Trimester	Trimestre d'une date A	Day			Integer	1

Image 2 : Calcul des différentes attributs de la dimension Date

Pour être cohérent avec le format de la date de vente dans le fichier des ventes et la date de publication dans la table Oracle des livres, on met la colonne “Day” au format `YYYY-MM-DD` en utilisant le composant “Altération structure flux”. *Cf.* `dim_date_etapes.png` pour visualiser l’ensemble de ces étapes.

Justification des choix

A la fin de la transformation, les jours de la semaine se présentent comme des entiers, qui correspondent à la position du jour dans la semaine (ex : 1 pour lundi). Pour anticiper l’affichage des jours de la semaine dans le résultat des requêtes dans l’outil de *reporting*, nous avons souhaité générer une nouvelle colonne qui indique le jour correspondant en toutes lettres. Mais le composant “Calculateur” que nous avions utilisé pour obtenir les différents attributs n’offrait pas d’opérateur permettant de générer ces valeurs. Nous verrons si l’outil de *reporting* nous permet de récupérer le jour en toutes lettres.

Evaluation et correction

Nous vérifions dans les statistiques de la transformation (`dim_date_stats.png`) que nous avons bien traité 365 lignes tout au long de notre transformation. Nous vérifions aussi dans le résultat final (`dim_date_resultat.png`) qu’il n’y a pas d’erreur, notamment pour le mois de février. En revanche, en consultant un calendrier de l’année 2014, nous constatons que le 1er janvier était un mercredi, or notre table finale indique “4” dans la colonne “DayOfWeek”. Nous complétons donc le composant “Calculateur” pour soustraire 1 à chaque valeur :

	Nouveau champ	Calcul	Champ A	Champ B
1	one	Affecter au champ une constante de valeur A	1	
2	Day	Date A + B Jours	START_DAY	Days_Since
3	Year	Année d'une date A	Day	
4	Month	Mois d'une date A	Day	
5	DayOfWeekFalse	Jour de la semaine d'uneA	Day	
6	Trimester	Trimestre d'une date A	Day	
7	DayOfWeekTrue	A - B	DayOfWeekFalse	one

Image 3 : Calcul supplémentaire pour corriger “DayOfWeek”

Notons qu’il est probable que l’opérateur “Jour de la semaine d’une <Date>” codé dans Pentaho considère le dimanche comme le premier jour de la semaine. Néanmoins, et pour éviter des incompréhensions ou ambiguïtés au moment du *reporting*, nous préférons conserver notre modification, le lundi nous apparaissant plus intuitivement comme le premier jour de la semaine.

Dimension Livre

Description

La dimension Livre contient les informations sur les livres vendus par l'entreprise "Fantastique" : ISBN, titre, auteurs, langue, date de publication, publisher, tags et genre. Toutes ces données nous sont fournies par la table Oracle des livres. Nous disposons de plus d'attributs que ce qu'annonçait notre MCD.

Construction

La construction de la dimension Date a impliqué plusieurs opérations de nettoyage :

- dates au format `YYYY-MM-DD`
- pas de chiffre dans le nom des auteurs
- décomposer les auteurs en deux champs
- normalisation des ISBN comme des entiers de 13 chiffres

Nous avons ensuite relevé plusieurs problèmes dans les données nécessitant des choix de correction.

Corrections et choix

Tout d'abord, l'ISBN des livres. Nous avons relevé que 20,4 % des livres de la table Oracle ont un ISBN mal formé. Cette proportion nous semblant relativement élevée, nous avons préféré ne pas supprimer ces livres, mais de remplacer leur ISBN par une suite de 13 "0".

Nous remarquons également des erreurs ou données manquantes dans la colonne "Publisher", où la valeur "?" apparaît sur plusieurs lignes. Nous notons que la plupart d'entre eux font partie des livres avec un ISBN incorrect. Enfin, nous notons plusieurs erreurs de frappe ou d'encodage. Par exemple, au lieu de "Denoël", la valeur est "Denoe¿¿l". Nous construisons une Regex pour en nettoyer une partie. L'exercice est difficile car certains noms de publisher sont composés par des caractères spéciaux, ou des majuscules accentuées. Nous renonçons donc à l'idée de tout corriger.

	Champ en entrée	Champ en sortie	RegEx	Rechercher	Remplacer par valeur
1	PUBLISHER		O	['?"¿]	

Image 4 : Regex de nettoyage du publisher (composant "Replacer dans chaîne de caractères")

Néanmoins, ayant notamment supprimé le caractère "?", nous nous retrouvons avec des publishers vides. Nous utilisons alors le composant "Remplacer valeur nulle", pour inscrire la valeur "inconnu", en l'appliquant, par la même occasion, à tous les champs d'un seul coup.

Se pose également le problème du champ “Langue” parfois vide, que nous remplaçons par “inconnu”.

Enfin, nous avons choisi de garder toutes les informations, jugeant qu’elles pouvaient être pertinentes pour le client.

Evaluation

Les statistiques d’exécution nous montrent que nous avons bien conservé les 1443 lignes tout au long de la transformation.

Construction du Datawarehouse

Transformation de la table de faits

22,5 % des lignes de ventes ont un ISBN incorrect. Nous décidons de ne pas priver le client de ces données, et remplaçons la valeur par une suite de 13 “9”. En conséquence, en anticipation de la jointure avec la table des livres, nous créons, dans Oracle, un enregistrement “factice” dans la table des livres où l’ISBN vaut une suite de 13 “9”. Il en découle que nous devrons être vigilants lors de la présentation des résultats des requêtes à ce que les résultats ne soient pas faussés par les associations factices entre les ventes à l’ISBN incorrect, et leur correspondant dans la table des livres.
Nous effectuons la même opération pour les magasins : remplacement des identifiants incorrects dans la table des ventes et création d’une ligne factice dans la table des magasins pour anticiper la jointure.
Même processus pour la date.
Le numéro de ticket est faux pour plusieurs ventes, son format étant une suite de neuf chiffres, nous remplaçons les valeurs incorrectes par une suite de 9 “0”.

Une fois ces corrections effectuées, nous regroupons les lignes identiques de la table des ventes : si un livre a été vendu *n* fois, le même jour, dans le même magasin, alors il n’y aura plus *n* lignes identiques mais une seule ligne avec un champ “Quantité” valant *n*. En conséquence, ce champ vaut “1” pour de nombreuses lignes. Nous réduisons ainsi cependant le nombre de lignes de 200 000 à 163 852, soit un peu moins de 20 % des lignes en moins, ce qui est souhaitable étant donné les jointures à venir.

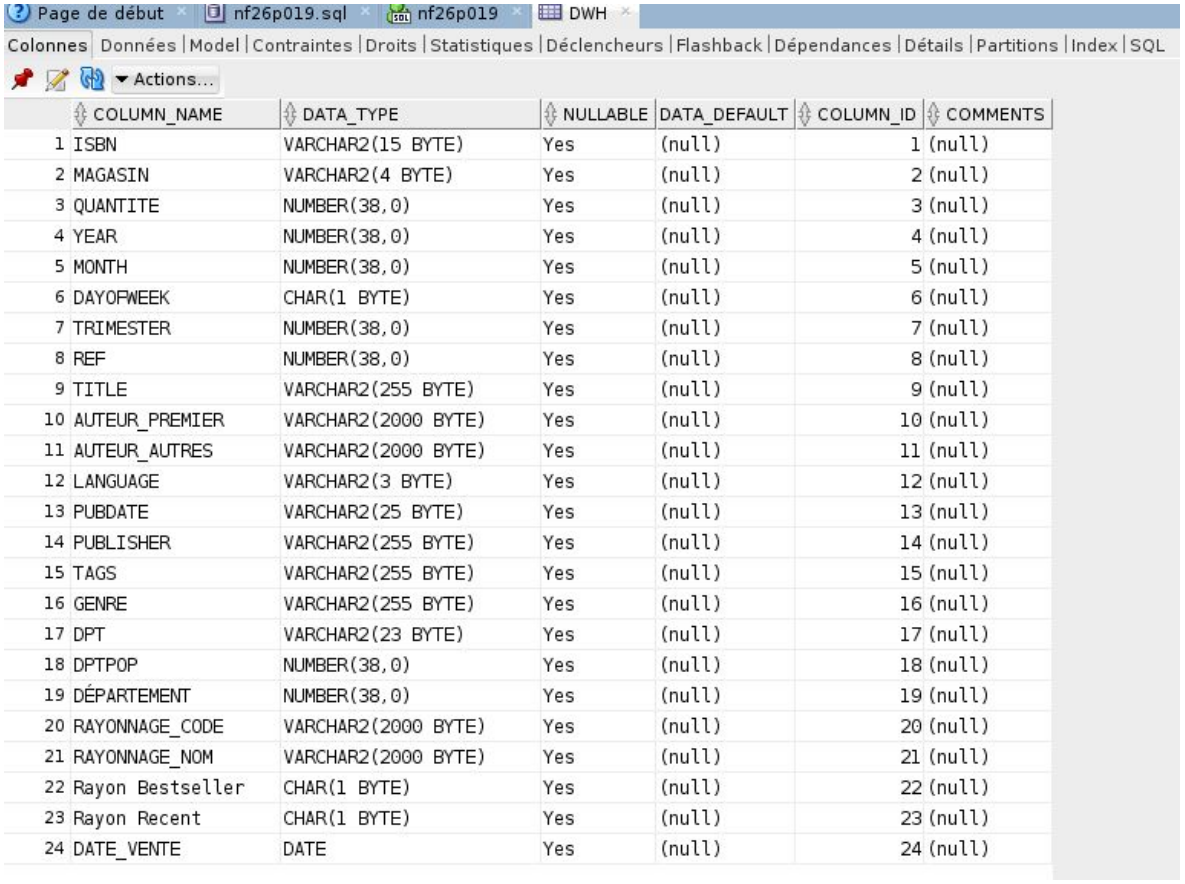
Jointures successives

Nous effectuons trois jointures successives : d’abord sur la date, puis sur les livres, et enfin sur les magasins, en triant à chaque fois les deux tables à joindre en fonction de la colonne de jointure, puis en supprimant une des deux colonnes utilisées pour la jointure, comme nous l’avions fait après la jointure entre les magasins et les départements.

Vérification

Après ajout de la colonne “Quantité”, la table des ventes contenait 163 852 lignes. Les statistiques d’exécution (`DWH_stats.png`) nous permettent de vérifier que nous avons bien conservé ce nombre de lignes après les différentes jointures.

Ci-dessous les colonnes de la table finale dans Oracle :


Page de début | nf26p019.sql | nf26p019 | DWH

Colonnes | Données | Model | Contraintes | Droits | Statistiques | Déclencheurs | Flashback | Dépendances | Détails | Partitions | Index | SQL

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ISBN	VARCHAR2(15 BYTE)	Yes	(null)	1	(null)
2	MAGASIN	VARCHAR2(4 BYTE)	Yes	(null)	2	(null)
3	QUANTITE	NUMBER(38,0)	Yes	(null)	3	(null)
4	YEAR	NUMBER(38,0)	Yes	(null)	4	(null)
5	MONTH	NUMBER(38,0)	Yes	(null)	5	(null)
6	DAYOFWEEK	CHAR(1 BYTE)	Yes	(null)	6	(null)
7	TRIMESTER	NUMBER(38,0)	Yes	(null)	7	(null)
8	REF	NUMBER(38,0)	Yes	(null)	8	(null)
9	TITLE	VARCHAR2(255 BYTE)	Yes	(null)	9	(null)
10	AUTEUR_PREMIER	VARCHAR2(2000 BYTE)	Yes	(null)	10	(null)
11	AUTEUR_AUTRES	VARCHAR2(2000 BYTE)	Yes	(null)	11	(null)
12	LANGUAGE	VARCHAR2(3 BYTE)	Yes	(null)	12	(null)
13	PUBDATE	VARCHAR2(25 BYTE)	Yes	(null)	13	(null)
14	PUBLISHER	VARCHAR2(255 BYTE)	Yes	(null)	14	(null)
15	TAGS	VARCHAR2(255 BYTE)	Yes	(null)	15	(null)
16	GENRE	VARCHAR2(255 BYTE)	Yes	(null)	16	(null)
17	DPT	VARCHAR2(23 BYTE)	Yes	(null)	17	(null)
18	DPTPOP	NUMBER(38,0)	Yes	(null)	18	(null)
19	DÉPARTEMENT	NUMBER(38,0)	Yes	(null)	19	(null)
20	RAYONNAGE_CODE	VARCHAR2(2000 BYTE)	Yes	(null)	20	(null)
21	RAYONNAGE_NOM	VARCHAR2(2000 BYTE)	Yes	(null)	21	(null)
22	Rayon Bestseller	CHAR(1 BYTE)	Yes	(null)	22	(null)
23	Rayon Recent	CHAR(1 BYTE)	Yes	(null)	23	(null)
24	DATE_VENTE	DATE	Yes	(null)	24	(null)

Image 5 : Colonnes de la table finale dans Oracle

Améliorations

Nous n’avons pas implémenté l’utilisation de clés artificielles pour la table de dates. Cela aurait d’abord nécessité de générer une clé étrangère pour identifier chaque enregistrement de la table de dates. Puis, dans la table des ventes, il aurait fallu remplacer chaque valeur de date par la clé correspondante. De même avec les dates de publication dans la table des livres. En conséquence, notre besoin de stockage est plus important que si nous avions utilisé des clés artificielles.

Par ailleurs, nous avons fait le choix de ne pas rejeter de lignes au cours de nos transformations, mais de remplacer les valeurs incorrectes par des constantes, et de générer des enregistrements

factices pour que les jointures puissent être réalisées. Il en découle que nous devrons redoubler de vigilance lors de la prochaine étape, celle du *reporting*, pour éviter des mauvaises interprétations des résultats des requêtes.