Intégration des connaissances utilisateurs pour des analyses personnalisées dans les entrepôts de données évolutifs

Cécile Favre, Fadila Bentayeb, Omar Boussaïd

ERIC, Université Lumière Lyon 2
5 avenue Pierre Mendès-France
69676 Bron Cedex
{cfavre|bentayeb}@eric.univ-lyon2.fr, omar.boussaid@univ-lyon2.fr

Résumé. Dans cet article, nous proposons une approche d'évolution de schéma dans les entrepôts de données qui permet aux utilisateurs d'intégrer leurs propres connaissances du domaine afin d'enrichir les possibilités d'analyse de l'entrepôt. Nous représentons cette connaissance sous la forme de règles de type «si-alors». Ces règles sont utilisées pour créer de nouveaux axes d'analyse en générant de nouveaux niveaux de granularité dans les hiérarchies de dimension. Notre approche est fondée sur un modèle formel d'entrepôts de données évolutif qui permet de gérer la mise à jour des hiérarchies de dimension.

1 Introduction

Les entrepôts de données centralisent des données provenant de différentes sources pour répondre aux besoins d'analyse des utilisateurs. Le schéma de l'entrepôt est défini avec l'objectif d'analyser des *mesures* qui caractérisent des *faits*, en fonction de *dimensions* qui peuvent être organisées sous forme de *hiérarchies*, composées de différents *niveaux de granularité*, déterminant la manière selon laquelle sont agrégées les données.

Pour concevoir le schéma d'un entrepôt, nous distinguons dans la littérature différents types d'approches : celles guidées par les sources de données (Golfarelli et al., 1998), celles guidées par les besoins d'analyse (Kimball, 1996) et les approches mixtes qui combinent les deux approches précédentes, mettant en adéquation des schémas candidats générés à partir des sources de données avec les besoins d'analyse exprimés par les utilisateurs (Nabli et al., 2005).

Cependant, en pratique, les sources de données, tout comme les besoins d'analyse sont amenés à évoluer. Dans la littérature, il existe deux alternatives qui permettent l'évolution de schéma nécessaire suite à ces modifications. D'une part la mise à jour de schéma qui est réalisée grâce à des opérateurs qui font évoluer un schéma donné (Hurtado et al., 1999). D'autre part, la modélisation temporelle qui consiste à garder la trace de ces évolutions en utilisant des labels de validité temporelle. Ces labels sont apposés soit au niveau des instances (Bliujute et al., 1998), soit au niveau des liens d'agrégation (Mendelzon et Vaisman, 2000), ou encore au niveau des versions du schéma (Morzy et Wrembel, 2004). L'inconvénient de ce type de solutions est la nécessité d'une réimplémentation des outils d'analyse, de chargement, ... afin de gérer les particularités de ces modèles.

Les deux alternatives sont intéressantes pour répondre au problème de l'évolution de schéma suite à une modification dans les sources de données, puisque ce sont des solutions techniques

devant être mises en œuvre par l'administrateur. Cependant, elles n'impliquent pas directement les utilisateurs dans le processus d'évolution. De ce fait, elles n'apportent pas de solution au problème posé par l'émergence de nouveaux besoins d'analyse exprimés par les utilisateurs. Or, au cours de l'utilisation de l'entrepôt, de nouveaux besoins apparaissent étant donné que (1) il est difficile de déterminer de façon exhaustive les besoins d'analyse pour l'ensemble des utilisateurs lors de la conception de l'entrepôt, (2) ces besoins dépendent également des propres connaissances des utilisateurs, (3) il est impossible de déterminer les besoins futurs.

Dans cet article, nous proposons alors une approche originale, qui s'inscrit dans l'approche de mise à jour de schéma, impliquant les utilisateurs dans le processus d'évolution de l'entrepôt, dans le but de leur fournir des analyses personnalisées en fonction de leurs propres connaissances du domaine et de leurs besoins. Les connaissances utilisateurs concernent plus précisément la définition de nouvelles données agrégées et sont représentées sous la forme de règles de type «si-alors». Ces règles, dites d'agrégation, sont ensuite utilisées pour générer de nouveaux axes d'analyse en créant dans les hiérarchies de dimension de nouveaux niveaux de granularité. Notre approche est fondée sur un modèle d'entrepôt de données évolutif à base de règles nommé *R-DW* (*Rule-based Data Warehouse*), dont nous proposons ici une formalisation. Pour valider notre approche, nous avons développé une plateforme baptisée WEDriK[1] (data Warehouse Evolution Driven by Knowledge) et avons appliqué notre approche aux données bancaires de LCL[2]. Néanmoins, cette partie n'est pas développée ici en raison du manque de place et peut être consultée dans Favre et al. (2006).

Dans la suite de cet article, nous introduisons tout d'abord dans la Section 2 un exemple simplifié motivant notre approche orientée utilisateurs. Puis, nous présentons dans la Section 3 notre approche ainsi que le principe du modèle *R-DW* sur lequel elle se base. Nous proposons ensuite la formalisation de ce modèle dans la Section 4. Enfin, nous concluons et indiquons les perspectives de ce travail dans la Section 5.

2 Exemple introductif

Pour illustrer notre approche de modélisation d'entrepôt de données évolutif à base de règles, nous utilisons le cas réel de la banque LCL. Le PNB annuel (Produit Net Bancaire) correspond à ce que rapporte un client à l'établissement bancaire. Cette mesure est analysée selon les dimensions `CLIENT`, `AGENCE` et `ANNEE` (Figure 1). Il est possible d'agréger les données selon le niveau `UNITE_COMMERCIALE`, qui est un regroupement d'agences (Figure 2a).

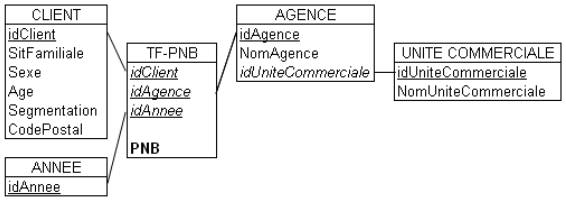

FIG. 1 – *Modèle multidimensionnel pour l'analyse du PNB de LCL*

[1] http ://eric.univ-lyon2.fr/~cfavre/wedrik

[2] Collaboration avec la Direction d'Exploitation Rhône-Alpes Auvergne de LCL–Le Crédit Lyonnais (LCL) dans le cadre d'une Convention Industrielle de Formation par la Recherche (CIFRE)

Supposons qu'un utilisateur veuille analyser les données selon le type d'agence ; il sait qu'il en existe trois : type «étudiant» pour les agences ne comportant que des étudiants, type «non résident» lorsque les clients ne résident pas en France, et le type «classique» pour les agences ne présentant pas de particularité. Ces informations n'étant pas présentes dans l'entrepôt, il est impossible pour lui d'obtenir une telle analyse. Notre objectif est donc de proposer à l'utilisateur d'intégrer dans le schéma de l'entrepôt sa connaissance sur les types d'agence pour créer le niveau de granularité TYPE_AGENCE (Figure 2b).

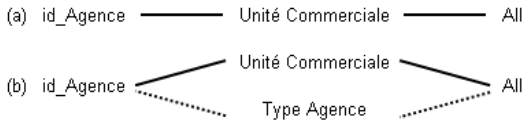

FIG. 2 – *Schémas de la dimension AGENCE : (a) initial, (b) enrichi par l'utilisateur*

3 Une approche orientée utilisateur

L'architecture de notre approche orientée utilisateur comprend quatre phases (Figure 3) : (1) une phase d'acquisition des connaissances utilisateurs sous la forme de règles d'agrégation, (2) une phase d'intégration de celles-ci pour les stocker dans le SGBD (Système de Gestion de Bases de Données), (3) une phase d'évolution du schéma dans laquelle a lieu la création de nouveaux niveaux de granularité au sein des hiérarchies de dimension à partir des règles d'agrégation, et enfin (4) une phase d'analyse sur le modèle qui évolue ainsi de façon incrémentale.

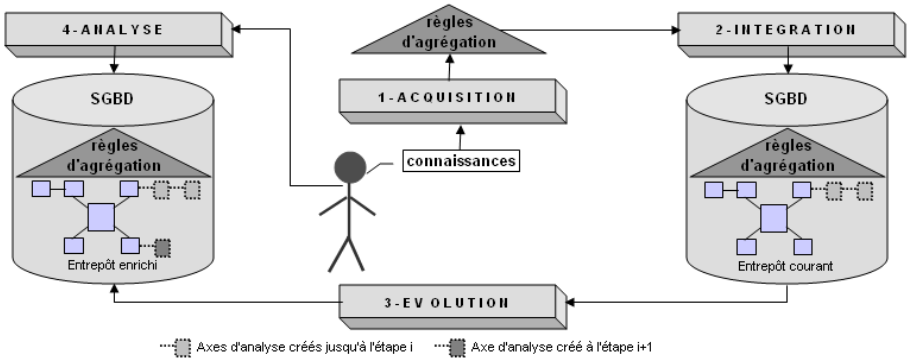

FIG. 3 – *Architecture pour l'évolution de schéma*

Les règles d'agrégation sont des règles de type «*si-alors*» utilisées pour représenter des liens d'agrégation entre deux niveaux de granularité où une instance d'un niveau inférieur correspond à une instance dans le niveau supérieur. Ainsi, l'ensemble de règles d'agrégation définissant un nouveau niveau doit construire une partition des instances du niveau inférieur (clauses «si»), et mettre en correspondance chaque classe de la partition avec une instance du nouveau niveau (clauses «alors»).

Notre approche se base sur un modèle d'entrepôt évolutif à base de règles d'agrégation (*R-DW*), qui est composé d'une partie «fixe», correspondant au schéma de l'entrepôt qui répond à des besoins d'analyse globaux, et une partie «évolutive» définie par les règles d'agrégation créant de nouveaux niveaux de granularité dans les hiérarchies de dimension, répondant aux besoins d'analyse personnalisés.

4 Formalisation du modèle *R-DW*

Nous désignons le modèle d'entrepôt évolutif à base de règles *R-DW* par le triplet suivant : *R-DW* $= (\mathcal{F}, \mathcal{E}, \mathcal{U})$ où $\mathcal{F}$ est la partie fixe, $\mathcal{E}$ la partie évolutive et $\mathcal{U}$ l'univers de *R-DW*.

Définition 1. *Univers de l'entrepôt*
Soit *R-DW* $= (\mathcal{F}, \mathcal{E}, \mathcal{U})$. L'*univers* $\mathcal{U}$ *de l'entrepôt R-DW* est un ensemble d'attributs, tel que : $\mathcal{U} = \mathcal{U}_1 \cup \mathcal{U}_2$ où $\mathcal{U}_1 = \{B_\alpha, 1 \leq \alpha \leq z\}$ est l'ensemble des z attributs prédéfinis (définis dans la partie fixe $\mathcal{F}$ de *R-DW*) et $\mathcal{U}_2 = \{C_\beta, \beta \geq 1\}$ est l'ensemble des attributs générés (définis par la partie évolutive $\mathcal{E}$ de *R-DW*).

Définition 2. *Partie fixe de R-DW*
Soit *R-DW* $= (\mathcal{F}, \mathcal{E}, \mathcal{U})$.
La *partie fixe de R-DW* est définie par : $\mathcal{F} =< F, \mathcal{D} >$ où F est une table de faits, et $\mathcal{D} = \{D_s, 1 \leq s \leq t\}$ est l'ensemble des t dimensions de premier niveau qui ont un lien direct avec F. Nous supposons que ces dimensions sont indépendantes.
Exemple 2. Dans la Figure 1, $< TF\_PNB, \{AGENCE, ANNEE, CLIENT\} >$ constitue la partie fixe de l'entrepôt *R-DW* pour l'analyse du PNB.

Définition 3. *Hiérarchie de dimension et niveau de granularité*
Soit *R-DW* $= (< F, \mathcal{D} >, \mathcal{E}, \mathcal{U})$.
$D_s.H_k, D_s \in \mathcal{D}, k \geq 1$ est une *hiérarchie de la dimension* D_s. La hiérarchie de dimension $D_s.H_k$ est composée d'un ensemble de w niveaux de granularité ordonnés notés L_i :
$D_s.H_k = \{L_1, L_2, \ldots, L_i, \ldots, L_w, w \geq 1\}$, avec $L_1 \prec L_2 \prec \ldots \prec L_i \prec \ldots \prec L_w$ où $\prec$ exprime l'ordre partiel sur les L_i.
Le *niveau de granularité* L_i de la hiérarchie H_k de la dimension D_s est noté $D_s.H_k.L_i$ ou plus simplement L_i^{sk}. Les niveaux de granularité peuvent être définis par des attributs générés.
Exemple 3. Selon le schéma de la Figure 2(b), on a :
$D_{AGENCE}.H_2$={AGENCE, TYPE AGENCE} ; $L_2^{AGENCE\ 2}$=TYPE AGENCE

Définition 4. *Attribut généré*
Soient *R-DW* $= (< F, \mathcal{D} >, \mathcal{E}, \mathcal{U})$, L_i^{sk} le niveau de granularité L_i de la hiérarchie H_k de la dimension D_s et $\mathcal{U}_2$ l'ensemble des attributs définis par la partie évolutive $\mathcal{E}$ de *R-DW*.
Un *attribut généré* $A \in \mathcal{U}_2$ caractérise le niveau de granularité L_i^{sk} et est noté $L_i^{sk}.A$.
Remarque : pour des raisons de simplification, nous supposons ici que chaque niveau de granularité est caractérisé par un seul attribut généré.
Exemple 4. Dans l'exemple de la section 2, $L_2^{AGENCE\ 2}.A = NomTypeAgence$

Définition 5. *Partie évolutive de R-DW*
Soit *R-DW* $= (\mathcal{F}, \mathcal{E}, \mathcal{U})$.
La *partie évolutive de R-DW* est définie par : $\mathcal{E} = \{< \mathcal{R}_{i.}, L_i^{sk}.A >\}$ où $\mathcal{R}_{i.} = \{r_{ij}, 1 \leq j \leq v, 1 \leq i \leq w\}$ est un ensemble de v règles d'agrégation définissant les valeurs de l'attribut

généré $L_i^{sk}.A$ qui représente le niveau de granularité L_i de la hiérarchie H_k de la dimension D_s. $\mathcal{E}$ représente donc l'ensemble des niveaux de granularité créés et les règles associées qui permettent de les définir.
Exemple 5. Dans l'exemple de la section 2, on a :

$$\mathcal{E} = \left\{ \begin{array}{l} si\ idAgence \in \{\text{'01903','01905','02256'}\}\ alors\ NomTypeAgence = \text{'étudiant'} \\ si\ idAgence = \text{'01929'}\ alors\ NomTypeAgence = \text{'non résident'} \\ si\ idAgence \notin \{\text{'01903','01905','02256','01929'}\}\ alors\ NomTypeAgence = \text{'classique'} \end{array}, L_2^{AGENCE\,2}.A \right\}$$

Définition 6. *Règle d'agrégation*
Une *règle d'agrégation* permet de définir le lien d'agrégation qui existe entre deux niveaux de granularité dans une hiérarchie de dimension.
Elle est basée sur un ensemble $\mathcal{T}$ de n termes de règles, notés RT_p, tel que :
$\mathcal{T} = \{RT_p, 1 \leq p \leq n\} = \{U\ op\ \{ens|val\}\}$
où U est un attribut de l'univers $\mathcal{U}$ de l'entrepôt ; op est un opérateur ($=$, $<$, $\leq$, $\geq$, $\neq$, $\in$, ...) ; ens est un ensemble de valeurs et val est une valeur finie.
Exemple 6a. $RT_1 : idAgence \in \{\text{'01903','01905','02256'}\}$; $RT_2 : Age > 80$

Une règle d'agrégation est une règle de type «*si-alors*». La conclusion de la règle (clause «alors») définit la valeur de l'attribut généré. La prémisse de la règle (clause «si») est basée sur une composition de *(conjonctions|disjonctions)* des termes de règles :
$r_{ij}\ :\ si\ RT_1\ (et|ou)\ RT_2\ ...\ (et|ou)\ RT_n\ alors\ L_i^{sk}.A = val_{ij}$
où $val_{ij} \in Dom_{L_i^{sk}.A}$ le domaine de définition de l'attribut $L_i^{sk}.A$.
Exemple 6b. Les règles suivantes déterminent les valeurs de l'attribut `NomClasseAge` qui définit le niveau hiérarchique `AGE` :
$r_{11}\ :\ si\ Age\ <\ 18\ alors\ NomClasseAge =$ 'mineur'
$r_{12}\ :\ si\ Age\ \geq\ 18\ alors\ NomClasseAge =$ 'majeur'

Les règles doivent définir une partition du domaine de définition de l'attribut généré.

Propriété 1.
Soient $L_i^{sk}.A$ l'attribut généré qui caractérise le niveau de granularité L_i de la hiérarchie H_k de la dimension D_s, $\mathcal{R}_{i.} = \{r_{ij}, 1 \leq i \leq w, 1 \leq j \leq u\}$ l'ensemble des u règles d'agrégation définissant les valeurs de l'attribut généré $L_i^{sk}.A$ et $body(r_{ij})$ la clause *si* de la règle r_{ij}. Les conditions exprimées dans un ensemble de règles qui définissent un attribut généré donné prises deux à deux doivent être disjointes :
$\forall i, \forall j, q\ tels\ que\ j < q, j \in [1, v-1], q \in [2, u], \quad body(r_{ij}) \bigcap body(r_{iq}) = \emptyset$

Propriété 2.
Soient $L_i^{sk}.A$ l'attribut généré qui caractérise le niveau de granularité L_i de la hiérarchie H_k de la dimension D_s, $\mathcal{R}_{i.} = \{r_{ij}, 1 \leq i \leq w, 1 \leq j \leq u\}$ l'ensemble des u règles d'agrégation définissant les valeurs de l'attribut généré $L_i^{sk}.A$, val_{ij} la valeur de $L_i^{sk}.A$ définie dans la règle r_{ij} et $Dom_{L_i^{sk}.A}$ le domaine de définition de $L_i^{sk}.A$. L'ensemble du domaine de définition $Dom_{L_i^{sk}.A}$ doit être couvert par les règles qui définissent l'attribut $L_i^{sk}.A$:
$\forall i, \bigcup_{j=1}^{u} val_{ij} = Dom_{L_i^{sk}.A}$

5 Conclusion

Dans cet article, nous avons proposé une approche originale qui exploite les connaissances utilisateurs pour faire évoluer le schéma de l'entrepôt, afin d'obtenir des analyses personna-

lisées. Notre approche est fondée sur un modèle formel d'entrepôt de données évolutif basé sur des règles d'agrégation. Notre approche présente plusieurs avantages : une flexibilité pour l'évolution du schéma de l'entrepôt puisque les hiérarchies de dimension sont mises à jour dynamiquement, une interaction accrue entre l'utilisateur et le système d'aide à la décision. L'utilisateur devient alors un réel acteur du processus décisionnel, dont le rôle va au-delà de la navigation dans les données selon le modèle initialement défini.

Ce travail ouvre de nombreuse perspectives. D'une part, nous voulons aborder d'autres opérations de mise à jour des hiérarchies de dimension. D'autre part, notre approche permet de prendre en compte l'évolution des besoins d'analyse. Nous nous intéressons également à la prise en compte de l'évolution conjointe des sources de données et des besoins d'analyse. Enfin, il nous faut gérer la mise à jour des règles et étudier l'impact de celle-ci sur l'entrepôt de données.

Références

Bliujute, R., S. Saltenis, G. Slivinskas, et C. Jensen (1998). Systematic Change Management in Dimensional Data Warehousing. In *IIIrd International Baltic Workshop on Databases and Information Systems, Riga, Latvia*, pp. 27–41.

Favre, C., F. Bentayeb, et O. Boussaïd (2006). Une approche orientée utilisateur pour l'évolution de schémas dans les entrepôts de données. Technical report, Laboratoire ERIC, Université Lyon 2.

Golfarelli, M., D. Maio, et S. Rizzi (1998). Conceptual Design of Data Warehouses from E/R Schemes. In *XXXIst Annual Hawaii International Conference on System Sciences (HICSS 98), Big Island, Hawaii, USA*, Volume 7, pp. 334–343.

Hurtado, C. A., A. O. Mendelzon, et A. A. Vaisman (1999). Updating OLAP Dimensions. In *IInd ACM International Workshop on Data Warehousing and OLAP (DOLAP 99), Kansas City, Missouri, USA*, pp. 60–66.

Kimball, R. (1996). *The Data Warehouse Toolkit*. John Wiley & Sons.

Mendelzon, A. O. et A. A. Vaisman (2000). Temporal Queries in OLAP. In *XXVIth International Conference on Very Large Data Bases (VLDB 00), Cairo, Egypt*, pp. 242–253.

Morzy, T. et R. Wrembel (2004). On Querying Versions of Multiversion Data Warehouse. In *VIIth ACM International Workshop on Data Warehousing and OLAP (DOLAP 04), Washington, District of Columbia, USA*, pp. 92–101.

Nabli, A., A. Soussi, J. Feki, H. Ben-Abdallah, et F. Gargouri (2005). Towards an Automatic Data Mart Design. In *VIIth International Conference on Enterprise Information Systems (ICEIS 05), Miami, Florida, USA*, pp. 226–231.

Summary

In this paper, we propose an approach to involve users in the data warehouse schema evolution process by allowing them to define new aggregated data. For this end, we use "if-then" rules to represent users' analysis needs. These rules are exploited to enrich and personalize analysis possibilities by creating new granularity levels in dimension hierarchies. We define a data warehouse formal model to support our approach.