Modèle d'archivage d'entrepôts de données multidimensionnelles

Faten Atigui, Franck Ravat, Olivier Teste, Gilles Zurfluh

IRIT (UMR 5505)
Institut de Recherche en Informatique de Toulouse
118 route de Narbonne
F-31062 Toulouse, France
{atigui, ravat, teste, zurfluh}@irit.fr

RÉSUMÉ. *Les entrepôts de données sont couramment utilisés dans les systèmes d'aide à la décision. Dans l'entrepôt, les données sont rafraîchies périodiquement et conservées de manière permanente. Lorsque ces données sont historisées, les décideurs portent généralement un intérêt moindre pour les données anciennes tout en les intégrant dans leurs analyses. Dans cet article, nous proposons un mécanisme d'archivage de données permettant de synthétiser les données anciennes pour répondre à l'évolution des besoins des décideurs. Pour ce faire, nous définissons un modèle conceptuel d'archivage de données multidimensionnelles. Nous présentons, ensuite, le modèle logique correspondant et un ensemble de règles permettant le passage automatique du modèle conceptuel en modèle logique.*

ABSTRACT. *Data warehouses are commonly used for decision making. Data integrated by these systems is, on the one hand, periodically updated and permanently stored on the other hand. When data is historized, decision-makers are usually less interested in old information, but still integrating it into their analysis. In this paper, we present a new mechanism for data archiving and summarizing across the time. This would satisfy evolving needs expressed by decision-makers. In particular, we propose a conceptual model that represents multidimensional data archiving. Moreover, we propose a logical model and a set of transformation rules to automatically generate the logical model from the conceptual one.*

MOTS-CLÉS : *Modélisation multidimensionnelle des entrepôts , Archivage de données, Transformation QVT*

KEYWORDS: *Data warehouse multidimensional modeling , Data archiving, QVT transformation*

1. Introduction

Un entrepôt de données (ED) est une collection de données thématiques, intégrées, non volatiles et historisées pour des fins décisionnelles. Les données pertinentes pour la prise de décision sont collectées à partir des sources au moyen des processus d'Extraction-Transformation-Chargement. Dans un ED, les données extraites sont souvent structurées selon un format multidimensionnel qui organise l'information en termes de sujets d'analyse (faits) et d'axes d'analyse (dimensions) au sein d'un schéma en étoile (Kimball, 1996). Un fait est composé d'un ensemble d'indicateurs d'analyse (mesures). Les dimensions sont composées de paramètres en fonction desquels les mesures sont étudiées. Les paramètres sont organisés en hiérarchies, de la granularité la plus fine (attribut racine de la dimension) à la plus générale. Par exemple, dans la figure 1, les mesures « quantité » et « montant » représentent les indicateurs d'analyse des « Ventes » en fonction des dimensions « Produit » et « Temps ». Les ronds présentent les différents niveaux de granularité dans chaque dimension.

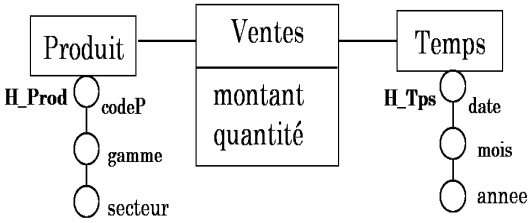

Figure 1 – Exemple d'ED multidimensionnelles

Dans un ED, les données sont conservées de manière permanente et sont rafraîchies de manière récurrente. De ce fait, l'ED possède de gros volume de données dans lequel le décideur risque de « se perdre » lors de ses analyses. De plus, les données historisées perdent de leur intérêt dans le temps : alors que la granularité des informations doit généralement être importante pour des données récentes ; elle peut être plus faible pour des données anciennes. Par exemple, un décideur peut analyser ses ventes au niveau de la granularité produit sur les cinq dernières années alors que pour les périodes antérieures, ces analyses au niveau du produit n'ont aucun sens (les identifiants des produits n'existent pas) et donc le décideur fera des analyses au niveau de la gamme du produit. Afin de faciliter la tâche du décideur et de mieux répondre à ses besoins, il est préférable de garder uniquement l'information dont il a besoin. Ceci permet d'anticiper le problème de temps de réponse aux requêtes multidimensionnelles dès les premières phases de modélisation multidimensionnelle.

Notre objectif est donc d'offrir un environnement d'analyse multidimensionnelle adapté aux besoins des décideurs en leur permettant de supprimer les niveaux de granularité inutiles dans leurs analyses. Nous présentons un mécanisme d'archivage de données multidimensionnelles basé sur les hiérarchies. Ce mécanisme permet de mieux répondre aux besoins des décideurs en synthétisant les données les moins récentes lorsque les détails ne sont pas utiles à la décision. Une archive permet de synthétiser des données en éliminant des niveaux de granularité non utilisés lors des ana-

lyses. La figure 2 montre l'évolution du schéma multidimensionnel tel que le souhaite le décideur. Cette figure illustre un exemple d'un schéma multidimensionnel archivé. Durant les cinq dernières années, l'analyse des ventes se fait par rapport aux niveaux de granularité les plus bas ; le produit et la date. Alors que durant la période antérieure de 2005 à 2007, ces analyses sont synthétisées par rapport aux gammes de produits dans l'archive 1. Avant 2005, seules les ventes annuelles par secteurs de produits sont conservées.

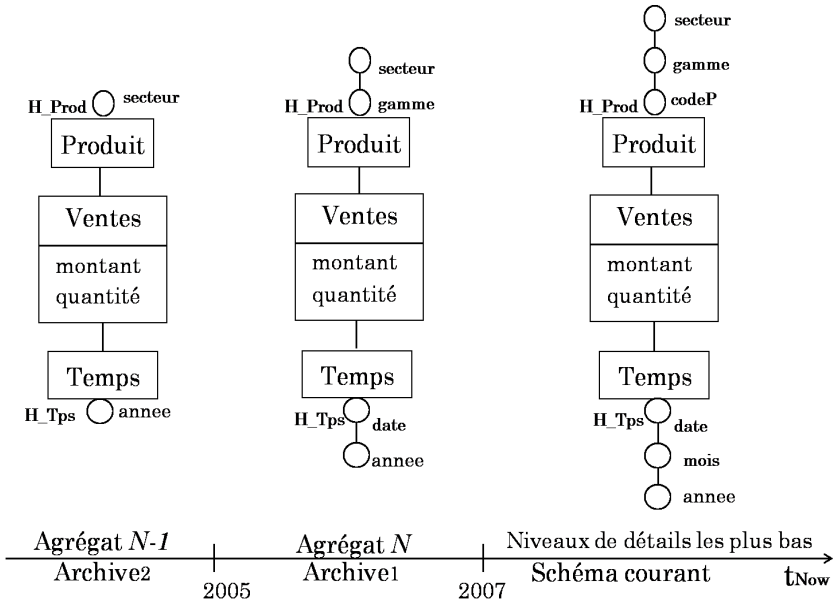

Figure 2 – Exemple d'archivage d'ED multidimensionnelles

L'article est organisé comme suit. En *s*ection 2, nous présentons les travaux qui ont traité l'évolution des données et des schémas dans les ED. En *s*ection 3, nous proposons un modèle conceptuel d'archivage. La *s*ection 4 définit le modèle logique ainsi que les règles d'obtention de ce modèle à partir du modèle conceptuel. En *s*ection 5, nous présentons le prototype implanté. La *s*ection 6 présente les développements futurs de nos travaux.

2. Etat de l'art

A notre connaissance, il n'y a pas de travaux qui portent sur l'archivage dans les entrepôts de données. Cependant, la prise en compte des données temporelles et de l'évolution des schémas utilisent des solutions qui nous sont utiles. Le concept d'archivage nécessite la gestion de l'évolution des données (Favre *et al.,* 2007). L'étude de l'évolution dans les ED a fait l'objet de plusieurs travaux (Golfarelli *et al.,* 2009). Les approches existantes peuvent être classées en quatre types (Wrembel, 2009), à savoir, l'évolution de schéma et de données, les extensions de versions, la simulation et les

extensions temporelles. L'évolution de schémas et de données (Hurtado *et al.,* 1999), (Blaschka *et al.,* 1999), (Fan *et al.,* 2004) proposent de maintenir un seul schéma de l'ED et un ensemble de données qui évoluent dans le temps. Les approches de simulation (Balmin *et al.,* 2000), (Bellahsene, 1998) utilisent des structures de données virtuelles afin de simuler l'évolution de l'ED. Les extensions de versions (Body *et al.,* 2003), (Golfarelli *et al.,* 2004) (Mendelzon *et al.,* 2000), (Ravat *et al.,* 2006), (Rizzi *et al.,* 2007), (Vaisman *et al.,* 2001)) gèrent l'évolution de l'ED par le moyen de versions de schéma et de versions de données.

Les extensions temporelles utilisent les estampilles sur les données modifiées afin de créer des versions temporelles. La plupart des approches se concentrent principalement sur la gestion des modifications dans la structure des instances de dimension. Dans (Kimball, 1996), l'auteur a proposé trois solutions de modélisation permettant la gestion de l'évolution des données dans les dimensions.

(Chamoni *et al.,* 1999) suggèrent de coupler le cube multidimensionnel avec des méta-cubes qui stockent les structures de la dimension avec leurs estampilles. (Eder *et al.,* 2001) étendent un modèle d'ED avec des caractéristiques temporelles permettant d'estampiller les instances de niveaux, leurs liens hiérarchiques, et les instances de fait avec le temps de validité. (Abelló *et al.,* 2003) proposent une structure de stockage bi-temporel où chaque attribut est associé à deux couples d'estampilles afin de suivre l'historique de ses valeurs en fonction du temps de validité et du temps de transaction. (Malinowski *et al.,* 2008) proposent un modèle qui prend en compte des mesures, des niveaux, des relations et des hiérarchies temporels. L'extension de ces objets de schéma avec un support temporel permet de stocker l'historique des modifications de leurs instances. Dans (Schlesinger *et al.,* 2001) les instances de dimensions sont estampillées les dates d'estampillage sont sauvegardées dans des structures supplémentaires.

Enfin, d'autres travaux s'intéressent à la gestion de l'évolution de l'ED provoquée par l'évolution des sources de données. Dans ce contexte, (de Amo *et al.,* 2000) présentent un ED temporelles auto-maintenable qui en plus des vues temporelles, comprend un ensemble de relations auxiliaires contenant des informations temporelles utilisées pour maintenir l'ED. (Lee *et al.,* 2002) proposent la synchronisation des vues en se basant sur les préférences. (Wrembel *et al.,* 2007) définissent un ensemble d'évènement et un ensemble d'actions associé à chaque évènement afin de propager les modifications vers une version d'ED. Les contributions présentées dans (Bellahsene, 2002) se concentrent sur une adaptation progressive des vues matérialisées après des changements structurels dans les sources. (Chen *et al.,* 2004) proposent un algorithme permettant de détecter les dépendances avec les mises à jour des sources qui engendrent des anomalies de requêtes cassées. Une autre technique, décrite dans (Eder *et al.,* 2004), se concentre sur la détection des changements structurels dans les instances de dimension. (Papastefanatos *et al.,* 2009) proposent une approche pour la détection et la propagation des de l'évolution des sources vers les processus ETL et le schéma de l'ED en utilisant la théorie de graphe. (Hurtado *et al.,* 1999) proposent un ensemble d'opérateurs permettant la modification des données, alors que (Letz *et*

al., 2002) définissent un ensemble de contraintes visant à respecter la cohérence des données.

La gestion de l'évolution telle qu'elle est proposée dans la littérature ne répond pas de manière pertinente à nos besoins. En effet, les travaux précédents permettent de conserver les évolutions de données ou de schémas dans le futur suite à l'évolution des besoins et/ou des sources de données. Toutefois, ils ne proposent pas de faire des agrégations de données afin de conserver uniquement les données utiles pour les prises de décision (Boly *et al.,* 2007).

Notre objectif est de proposer un mécanisme de synthèse des données les plus anciennes. Ce mécanisme permet de stocker les données les plus récentes de façon détaillée alors qu'il synthétise les données anciennes en fonction de leur âge. Plus une activité est vieille, moins elle est détaillée. Notre objectif est donc d'offrir différents niveaux de modélisation et donc de stockage des données décisionnelles afin de conserver dans le temps différents niveaux de synthétisation des indicateurs d'analyse.

3. Archivage : modèle conceptuel

L'archivage de données permet de conserver uniquement l'information utile pour le décideur. Ce mécanisme permet donc de synthétiser les données les moins récentes. La modélisation conceptuelle fournit un niveau d'abstraction indépendant des aspects techniques et se concentrant sur les besoins décisionnels (Rizzi *et al.,* 2006). Bien qu'il n'existe aucun modèle standard pour la conception des entrepôts de données (Sen *et al.,* 2005), de manière générale les ED multidimensionnelles sont organisés selon un modèle en étoile, en flocon ou constellation (Kimball, 1996). Dans les schémas en étoile, le fait est relié à un ensemble de dimension où chaque dimension est représentée par une seule entité. L'inconvénient majeur de cette modélisation est que la hiérarchisation des données n'est pas explicitement représentée. Alors que la particularité des schémas en flocon est que chaque dimension est définie par un ensemble de tables. Quant aux schémas en constellation, qui semblent être les plus génériques, ils permettent de définir un ensemble de faits pouvant partager une ou plusieurs dimensions. Ainsi, afin de montrer la généricité de nos propositions, nous proposons d'étendre le concept de constellation (Golfarelli *et al.,* 1998), (Ravat *et al.,* 2007) à celui de constellation archivée.

Une constellation archivée présente, en plus des concepts de base (faits et dimensions), un ensemble d'archives multidimensionnelles. A partir d'un schéma courant, il est possible de construire plusieurs archives sur les données les plus anciennes. Dans une même archive, un fait peut être synthétisé par rapport à une ou plusieurs dimensions. Le processus d'archivage est déclenché par des évènements temporels. En fonction des intervalles de validité temporels choisis par l'utilisateur, les données peuvent être synthétisées. L'utilisateur peut également décider de supprimer certaines données (souvent les plus anciennes).

Dans ce qui suit nous présentons les définitions des différents composants d'une constellation archivée. Les quatre premières définitions montrent les concepts de base.

Définition 1 *Un schéma multidimensionnel S est défini par $(F^S, D^S, Star^S)$ où :*

– F^S est un ensemble de faits,

– D^S est un ensemble de dimensions,

– $Star^S : F^S \mapsto 2^{D^S}$ associe chaque fait à un ensemble de dimensions.

Définition 2 *Un fait, noté $F_i \in F^S$, est défini par (N^{F_i}, M^{F_i}) où :*

– N^{F_i} est le nom du fait,

– $M^{F_i} : \{f_1(m_1^{F_i}), ..., f_p(m_w^{F_i})\}$ est un ensemble de mesures associées à des fonctions d'agrégation $\{f_1, ..., f_p\}$.

Définition 3 *Une dimension, notée $D_i \in D^S$, est définie par $(N^{D_i}, A^{D_i}, H^{D_i})$ où :*

– N^{D_i} est le nom de la dimension,

– $A^{D_i} : \{a_1^{D_i}, ..., a_u^{D_i}\}$ est un ensemble d'attributs de dimension,

– $H^{D_i} : \{H_1^{D_i}, ..., H_w^{D_i}\}$ est un ensemble de hiérarchies.

Définition 4 *Une hiérarchie, notée $H_j^{D_i} \in H^{D_i}$, est définie par $(N^{H_j}, < p_1, ..., p_n >, \{(p_i, a_f), ...\})$ où :*

– N^{H_j} est le nom de la hiérarchie,

– $< p_1, ..., p_n >$ est une liste de paramètres (attribut identifiant un niveau de granularité d'analyse),

– $\{(p_i, a_f), ...\}$ permet l'association d'attributs faibles aux paramètres.

Ces concepts de base vont permettre de définir les éléments d'un schéma multidimensionnel d'une archivé. Une constellation archivée est composée d'un schéma courant (données les plus détaillées) et d'un ensemble d'archives (données les plus anciennes). Plus précisément, une archive contient contient les données synthétisées pour une période donnée. Celle-ci est toujours construite à partir d'un schéma précédent : la première archive est construite à partir du schéma courant, la seconde est construite à partir du schéma de la première archive, etc.
Dans une même période de temps, une seule archive est définie. Tous les composants (faits, mesures, dimensions, hiérarchies, paramètres et attributs faibles) du schéma multidimensionnel d'une archive sont issus d'un seul composant source dans le schéma précédent.

Définition 5 *Une constellation archivée CA est définie par (N^C, S^C, A^C) où :*

– N^C *est le nom de la constellation,*

– S^C *est un schéma multidimensionnel courant,*

– $A^C : \{A_1^C, ..., A_n^C\}$ *est un ensemble d'archives.*

Définition 6 *Une archive,* $A_i \in A^S$ *est définie par* $(N^{A_i}, S^{A_i}, Derive^{S^{A_i}}, T)$ *où :*

– N^{A_i} *est le nom de l'archive,*

– S^{A_i} *est le schéma multidimensionnel de l'archive,*

– $Derive^{S^{A_i}} : F^{S_{i-1}} \longmapsto 2^{D^{S_{i-1}}}$ *permet d'archiver un fait* $F^{S_{i-1}}$ *par rapport à un ensemble de dimensions dans le schéma multidimensionnel précédent* S_{i-1}*. La première archive est construite à partir du schéma courant de la constellation, les autres schémas d'archives sont construit l'un à partir de l'autre.*

- $F^{S_{i-1}} : \{f_1(m_1^{S_{i-1}}), ..., f_p(m_n^{S_{i-1}})\}$ *est l'ensemble de faits du schéma multidimensionnel précédent.* m_i *est l'ensemble des mesures archivées et associées à des fonctions d'agrégation.*

- $D^{S_{i-1}} : \{p_j, ..., p_m\} j \geq 1$ *: est l'ensemble de niveaux de granularités utilisés dans l'archivage du schéma* S_{i-1}*. D'un point de vue graphique, les niveaux de granularité d'un schéma* S_{i-1} *utilisés pour archiver les données dans le schéma* S_i *sont colorés en noir.*

- $\forall D_k \in Derive(F_i), D_k \in Star(F_i)$ *: un fait n'est archivé que par rapport aux dimensions auxquelles il est lié, mais pas nécessairement toutes.*

– $T = [T_{Debut}, T_{Fin}[$ *: intervalle temporel durant lequel l'archive est valide. Le début d'une archive* A_i *représente la fin de l'archive* A_{i+1}*.*

Le niveau utilisé pour l'archivage peut être un niveau existant dans la dimension du schéma courant de la constellation. Par exemple dans la dimension « Produit », l'attribut d'archivage utilisé est la « gamme » du produit. Lorsque l'archivage porte sur la dimension « Temps » le décideur peut synthétiser les données par rapport à une durée particulière ; par exemple, la moyenne des ventes par rapport un nombre d'années ou de mois donné.

Exemple 1 *La figure 3 présente un exemple de schéma multidimensionnel archivé. Le schéma courant (3a) permet l'analyse des montants et des quantités de ventes en fonction des clients selon différents niveaux d'agrégation (codeC, ville et pays), des produits (codeP, gamme, marque et secteur) et du temps (date, mois et année). Les deux archives (3b et 3c) permettent de répondre à un besoin spécifique de synthèse de données. Dans un premier temps, on veut garder une synthèse sur les quantités et les montants des ventes en fonction des villes des clients et des gammes de produits. Pour ce faire, la première archive* A_1 *est construite en se basant sur le schéma courant (SC). Cette archive (3b) décrit les données conservées à* $T_1 = [2005, 2007[$.
$A_1 = \{SC.Ventes\{Sum(montant),\ Sum(quantite)\},\ \{SC.Produit$

$\{gamme, secteur\},\ SC.Client\{ville, pays\},\ SC.Temps$
$\{date, mois, annee\}\}, T_1\}$.
Dans un second temps, le décideur souhaite garder les données les moins récentes de manière encore plus synthétisée. En effet, moins les données sont récentes, plus l'archive est synthétisée. Une archive A_2 *est donc créée à partir de* A_1 *pour une validité portant sur les années qui précèdent* 2005*. L'archive* A_2 *permet de stocker les quantités et les montants des ventes selon les pays et par pas de deux ans. Cette archive est créée à partir de la première archive* A_1*.*
$A_2 = \{A_1.Ventes\{Sum(montant), Sum(quantite)\}, \{A_1.Produit$
$\{gamme, secteur\}, A_1.Client\{pays\}, A_1.Temps\{annee\_2\}\}, T_2\}$.

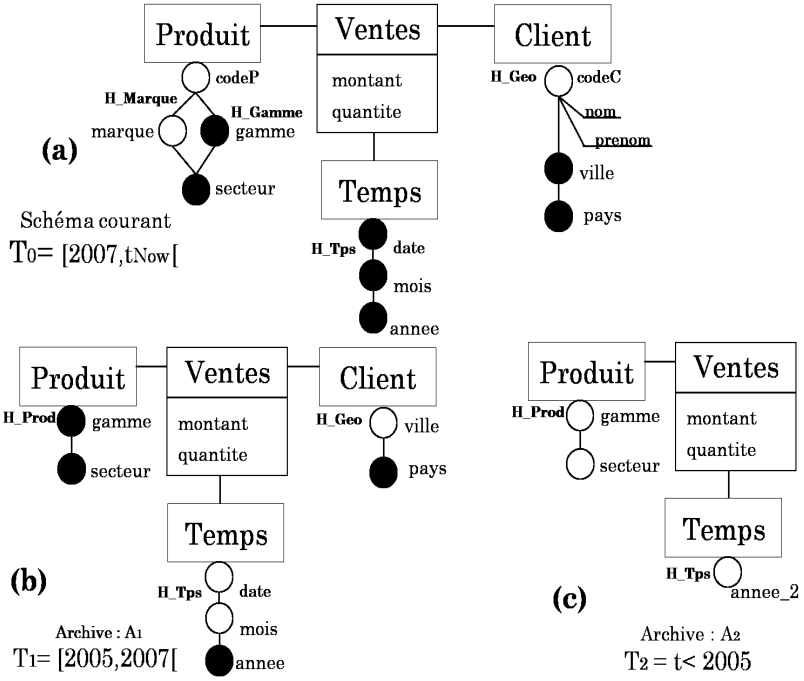

Figure 3 – Exemple d'archivage d'entrepôt de données multidimensionnelles

4. Archivage : modèle logique

Il s'agit de transformer le schéma conceptuel multidimensionnel évoqué avec archivage en un schéma logique relationnel. Dans ce qui suit, nous présentons les principales règles de transformation d'une constellation en schéma logique R-OLAP[1]. Bien qu'il existe différents types de schémas R-OLAP, nous avons décidé de détailler les règles de transformation pour le R-OLAP dénormalisé. Ce modèle est le plus couram-

1. R-OLAP : Relational On Line Analytical Processing

ment utilisé en décisionnel car il repose sur un nombre limité de tables qui permet ainsi une restitution rapide des données (nombre de jointures limitée). Le modèle en constellation représente les données en termes de « Fait » et de « Dimension ». Une constellation archivée permet de conserver des données synthétisées au fil du temps.

Remarque 1 *La modélisation logique en R-OLAP normalisé se distingue de la modélisation en R-OLAP dénormalisé ; les tables dimensions sont normalisées conformément aux principes de la 3ème forme normale. Les règles de transformations restent donc les mêmes pour le fait. Alors que les dimensions sont normalisées ; chaque niveau d'attribut est transformé en une table relationnelle. Les relations de composition entre les différents niveaux d'une même hiérarchie sont traduites par des clés étrangères référençant le niveau de granularité suivante. Le modèle conceptuel avec archives peut être transformé en modèle R-OLAP normalisé prenant en compte les données archivées en adaptant les règles présentées dans (Atigui* et al., *2010).*

4.1. *Transformation du modèle conceptuel en modèle logique*

Les règles suivantes montrent les relations de transformation des différents éléments d'une constellation archivée en éléments du schéma R-OLAP dénormalisé.

– R1. Le schéma courant de la constellation est transformée en un schéma relationnel où :

- Toute dimension est transformée en une relation,

- Les attributs = tous les paramètres et les attributs faibles relatifs aux différentes hiérarchies composant cette dimension,

- La clé primaire = paramètre appartenant au niveau de granularité le plus bas (appelé paramètre racine),

- Tout fait est transformé en une relation,

- Les mesures et les paramètres racines relatifs aux dimensions en relation avec le fait sont transformés en attributs de la relation,

- Les clés étrangères correspondent aux paramètres racines des dimensions liées au fait,

– R2. Le schéma d'une archive est transformé en un schéma relationnel où :

- Seules les dimensions qui ont changé de schéma (un ou plusieurs niveaux de granularité ont été supprimés) sont transformées en relation. Les attributs de la dimension correspondent aux attributs de la relation créée.

- Tout fait est transformé en une relation de la même manière que dans un schéma courant. Les clés étrangères référencent une relation dimension qui vient d'être créée dans l'archive ou bien une relation existante dans une archive précédente (ou le schéma courant s'il s'agit de la première archive).

4.2. *Transformation de modèles basée sur QVT (Query/View/Transformation)*

Cette section présente la formalisation en QVT[2] des règles de transformation permettant un passage automatique entre le modèle conceptuel et le modèle logique. QVT est un langage déclaratif standardisé par l'OMG[3]. Ce langage est généralement utilisé dans le cadre d'une Architecture Dirigée par les Modèle (MDA)[4]. Dans nos travaux précédents (Atigui *et al.,* 2011b), (Atigui *et al.,* 2011a) nous avons défini une architecture d'ED dirigé par les modèles. Dans ce travail, le langage QVT nous sert à automatiser le processus de transformation du modèle d'archivage. Une transformation QVT entre deux modèles candidats est spécifiée grâce à un ensemble de relations (appelés aussi règles). Chaque transformation est composée des éléments suivants :

– « Domains » : chaque domaine désigne un modèle candidat et un ensemble d'éléments à relier.

– « Relation Domain » : permet de spécifier le type de relation entre les domaines, elle peut être marquée comme « Checkonly » (C) ou « Enforced » (E). Un domaine « Chekonly » permet de vérifier s'il existe une correspondance valide qui satisfait la relation ; alors qu'un domaine « Enforced » permet de créer un élément dans le modèle si le lien de correspondance n'est pas vérifé. Pour chaque domaine le nom de son méta-modèle sous-jacent doit être spécifié.

– La clause « When » : décrit les pré-conditions qui doivent être remplies pour réaliser la transformation.

– La clause « Where » : détermine les post-conditions qui doivent être remplies par tous les éléments du modèle participant à la relation.

La traduction des règles présentées dans la sous-section précédente en QVT est constituée de plusieurs relations, dans ce qui suit nous présentons les principales relations relatives à la transformation d'une archive, en utilisant la syntaxe graphique du langage QVT. En ce qui concerne la transformation d'un schéma multidimensionnel courant, ceci se fait de la même manière que la transformation d'un schéma en constellation non archivé. Ces règles de transformation ont été détaillées dans (Atigui *et al.,* 2010).

Relation « Main ». Cette relation est le point d'entrée au processus de transformation. La partie gauche de la figure 4 montre les éléments du modèle multidimensionnel archivé (amd : AMD) transformés en éléments du modèle R-OLAP dénormalisé (r-olap-d : Relational) présenté par la partie droite de la figure. Une archive est transformée en un schéma relationnel de même nom. Chaque fait de l'archive est transformé en une relation via la règle « FactToRelation ». Chaque dimension est traduite en une relation du schéma R-OLAP par la règle « DimensionToRelation » spécifiée au niveau de la clause « Where ». Le temps de validité d'une archive correspond au temps de validité du schéma relationnel.

---

2. http ://www.omg.org/spec/QVT/1.1/PDF/
3. Object Management Group
4. http ://www.omg.org/cgi-bin/doc ?omg/03-06-01

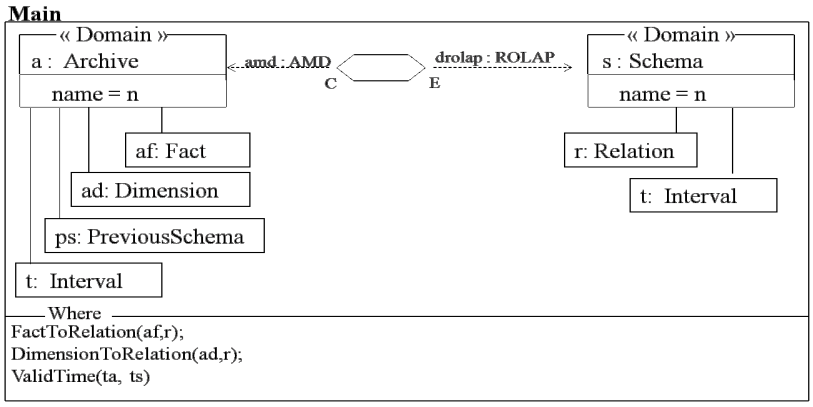

Figure 4 – Relation « Main »

Relation « DimensionToRelation ». La figure 5 montre qu'une dimension est transformée en une relation de même nom. Seules les dimensions qui ont changé de schéma (qui ont moins de niveaux de granularité) sont transformées en relation. Ceci est précisé dans la pré-condition de la clause « When ». Tous les attributs (les paramètres et les attributs faibles) des différents niveaux de granularité composant cette dimension sont transformés en attributs de la relation en utilisant les règles « ParameterToAttribute » et « WeakAttributeToAttribute ». Le paramètre racine détermine la clé primaire de cette relation (relation « ParameterToPrimaryKey » définie dans la clause « Where »).

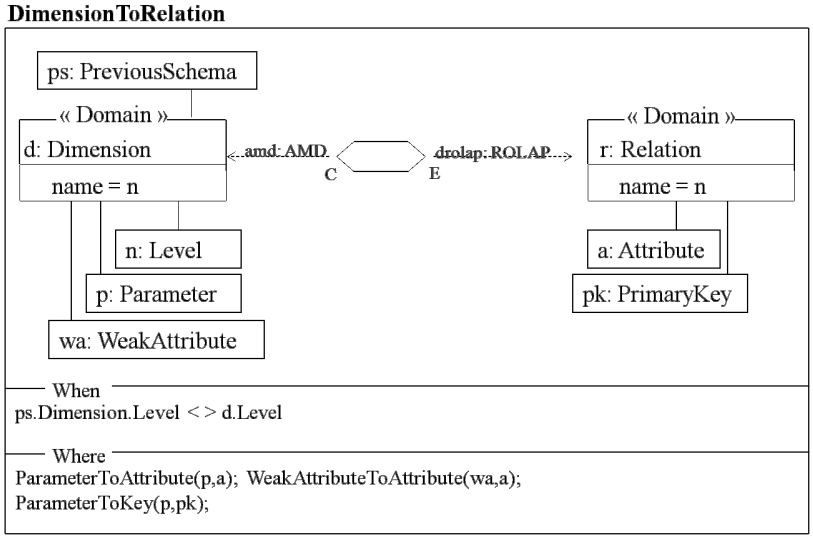

Figure 5 – Règle QVT de transformation de dimension en relation

Relation « FactToRelation ». Cette relation est présentée par la figure 6. Une fois les dimensions liées au fait transformées (la pré-condition « DimensionToTable » de la

clause « When »), le fait est converti en une relation ayant le même nom. Les mesures sont transformées en attributs au moyen de la relation « MeasureToAttribute » de la clause « Where ». Les paramètres racines des dimensions liées au fait sont transformés en clés étrangères par la relation « ParameterToForeignKey » et par la suite en clés primaires grâce à la relation « ParameterToPrimaryKey ».

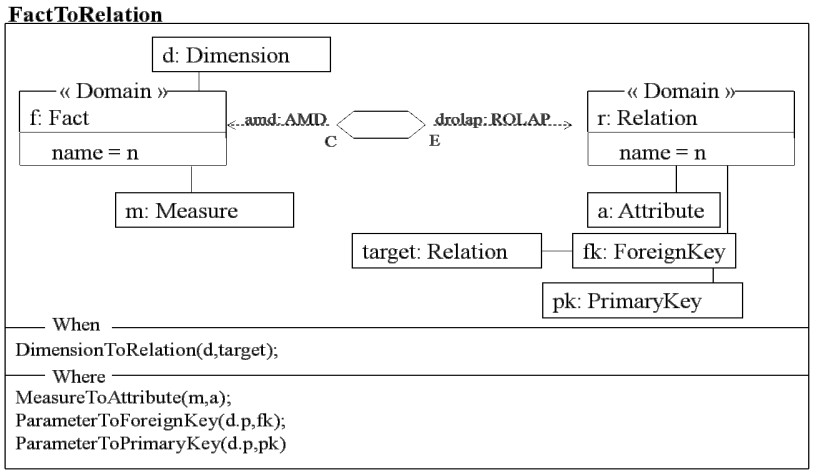

Figure 6 – Règle QVT de transformation de fait en relation

Exemple 2 *L'exemple ci-dessous correspond au résultat de la transformation du schéma en constellation archivée des « Ventes » présenté dans l'exemple 1 en schémas relationnels. Ce résultat est obtenu suite à une succession de transformation de modèles permettant d'aboutir au modèle logique. Ces transformations visent à convertir les faits et les dimensions du schéma courant et des archives en relations du schéma logique. Les dimensions et le fait du schéma courant correspondent aux relations suivantes :*
$SC.Produit(\underline{codeP}, gamme, marque, secteur)$
$SC.Client(\underline{codeC}, nom, sexe, ville, pays)$
$SC.Temps(date, mois, annee)$
$SC.Ventes(\underline{codeP\#, codeC\#, date\#}, montant, quantite)$

Dans l'archive A_1, dérivée à partir du schéma courant, tous les niveaux de granularité de la dimension Temps sont archivés, ainsi, cette dimension n'est pas transformée en une nouvelle relation, le fait Ventes de l'archive référence la relation du schéma courant :
$A_1.Produit(\underline{gamme}, secteur)$
$A_1.Client(\underline{ville}, departement, pays)$
$A_1.Ventes(\underline{gamme\#, ville\#, SC.Temps.date\#}, montant, quantite)$

L'archive A_2 est dérivée à partir de l'archive A_1. Comparé au schéma de l'archive précédente, tous les niveaux de granularité de la dimension Produit sont archivés,

ainsi, cette dimension n'est pas transformée en une nouvelle relation, et le fait Ventes de l'archive référence la relation du schéma de l'archive A_1 :
$A_2.Client(\underline{pays})$
$A_2.Temps(\underline{annee_2})$
$A_2.Ventes(\underline{A_1.Produit.gamme\#, pays\#, annee_2\#}, montant, quantite)$

Remarque 2 *Dans ce papier, nous présentons uniquement les règles QVT permettant les transformations entre modèles. Cependant, ces règles nécessitent d'être validées afin de garantir la qualité des modèles produits en sortie. En effet, un ensemble de contraintes OCL (Object-Constraint-Language)*[5] *peuvent être appliquées à cette fin. Ces contraintes permettent de vérifier la conformité entre les modèles en entrée et les modèles en sortie, c'est-à-dire, de s'assurer qu'à chaque élément du modèle en entrée correspond un élément dans le modèle en sortie. Ces règles permettent également de vérifier la conformité de chaque modèle par rapport à son méta-modèle.*

5. Implantation

La figure 7 présente l'architecture de notre prototype. Ce dernier permet de créer le schéma conceptuel de la constellation archivée à partir d'un schéma multidimensionnel existant. Le schéma conceptuel obtenu est ensuite transformé automatiquement en un schéma logique (R-OLAP) restitué en sortie. Cette transformation de modèles, implantée en utilisant le langage QVT, fait appel à un ensemble de méta-modèles stockés en amont sous forme de fichier « Ecore ». Les règles de transformation de modèles ont été implantées en utilisant Medini-QVT[6]. Ce dernier est un plug-in Eclipse[7] qui permet d'implanter les règles de transformation de modèles-vers-modèles en utilisant la syntaxe textuelle de QVT.

6. Conclusion

Dans cet article, nous avons présenté une approche pour l'archivage de données multidimensionnelles dans un entrepôt. Le modèle conceptuel permet de décrire l'évolution des données conservées dans le temps en réponse aux besoins des décideurs. Ce modèle présente un schéma courant et un ensemble de schémas archivés. Nous avons également présenté le modèle logique R-OLAP dénormalisé ainsi que les règles de transformation du modèle conceptuel vers le modèle logique. Afin d'automatiser le processus d'archivage de données, ces règles ont été formalisées en utilisant le langage QVT.

Dans les futurs travaux, nous envisageons, dans un premier temps, de compléter le processus d'archivage en proposant un modèle physique, et un ensemble de règles formalisées en QVT permettant de générer ce modèle. Dans un second temps, nous

5. http ://www.omg.org/spec/OCL/2.2/PDF
6. http ://projects.ikv.de/qvt
7. http ://www.eclipse.org/

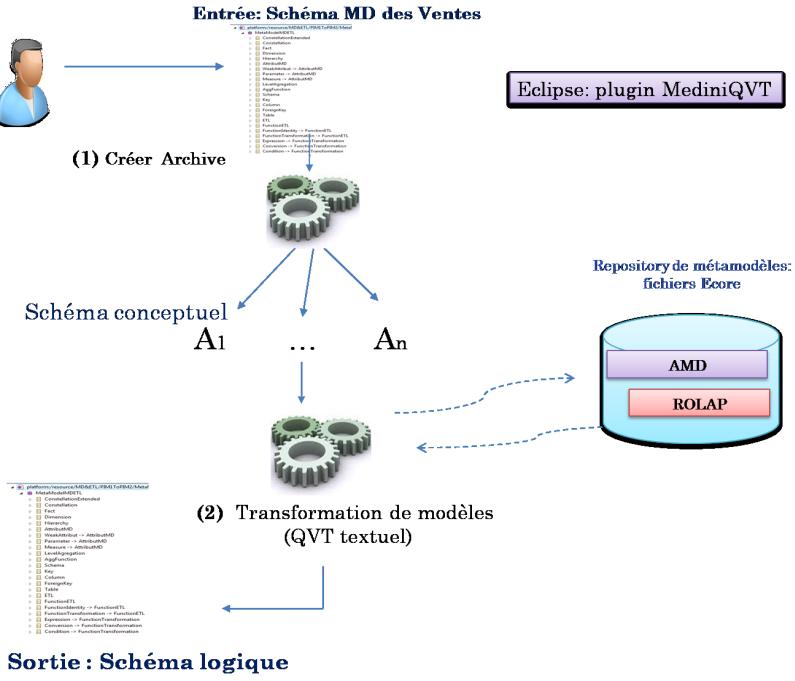

Figure 7 – Architecture du prototype

aborderons une problématique liée à l'interrogation d'entrepôt de données multidimensionnelles archivées.

Nous avons l'intention de développer l'outil via une interface graphique qui permet à l'utilisateur de saisir l'entrée du système. En outre, nous envisageons d'appliquer l'approche sur des cas d'études du monde réel et de la faire évaluer par différents utilisateurs.

7. Bibliographie

Abelló A., Martín C., « A Bitemporal Storage Structure for a Corporate Data Warehouse », *ICEIS (1)*, p. 177-183, 2003.

Atigui F., Ravat F., Teste O., Zurfluh G., « Démarche dirigée par les modèles pour la conception d'entrepôts de données multidimensionnelles (regular paper) », *Journées Bases de Données Avancées (BDA), Toulouse, 18/10/2010-22/10/2010*, 2010.

Atigui F., Ravat F., Teste O., Zurfluh G., « Modèle unifié pour la transformation des schémas en constellation », *7èmes journées francophones sur les Entrepôts de Données et l'Analyse en ligne (EDA 2011), Clermont-Ferrand*, vol. B-7 of *RNTI*, Hermann, Paris, p. 5-22, Juin, 2011a.

Atigui F., Ravat F., Tournier R., Zurfluh G., « A Unified Model Driven Methodology for Data Warehouses and ETL Design », *ICEIS (1)*, p. 247-252, 2011b.

Balmin A., Papadimitriou T., Papakonstantinou Y., « Hypothetical Queries in an OLAP Environment », *VLDB*, p. 220-231, 2000.

Bellahsene Z., « View Adaptation in Data Warehousing Systems », *DEXA*, p. 300-309, 1998.

Bellahsene Z., « Schema Evolution in Data Warehouses », *Knowl. Inf. Syst.*, vol. 4, n° 3, p. 283-304, 2002.

Blaschka M., Sapia C., Höfling G., « On Schema Evolution in Multidimensional Databases », *DaWaK*, p. 153-164, 1999.

Body M., Miquel M., Bédard Y., Tchounikine A., « Handling Evolutions in Multidimensional Structures », *ICDE*, p. 581-591, 2003.

Boly A., Goutier S., Hébrail G., « Des fonctions d'oubli intelligentes dans les entrepôts de données », *EGC*, p. 223-234, 2007.

Chamoni P., Stock S., « Temporal Structures in Data Warehousing », *DaWaK*, p. 353-358, 1999.

Chen S., Chen J., Zhang X., Rundensteiner E. A., « Detection and Correction of Conflicting Source Updates for View Maintenance », *ICDE*, p. 436-447, 2004.

de Amo S., Alves M. H. F., « Efficient Maintenance of Temporal Data Warehouses », *IDEAS*, p. 188-196, 2000.

Eder J., Koncilia C., « Changes of Dimension Data in Temporal Data Warehouses », *DaWaK*, p. 284-293, 2001.

Eder J., Koncilia C., Mitsche D., « Analysing Slices of Data Warehouses to Detect Structural Modifications », *CAiSE*, p. 492-505, 2004.

Fan H., Poulovassilis A., « Schema Evolution in Data Warehousing Environments - A Schema Transformation-Based Approach », *ER*, p. 639-653, 2004.

Favre C., Bentayeb F., Boussaïd O., « Evolution de modèle dans les entrepôts de données : existant et perspectives », *3èmes journées francophones sur les Entrepôts de Données et l'Analyse en ligne (EDA 07), Poitiers*, vol. B-3 of *Revue des Nouvelles Technologies de l'Information*, Cépaduès Editions, Toulouse, p. 21-36, Juin, 2007.

Golfarelli M., Lechtenbörger J., Rizzi S., Vossen G., « Schema Versioning in Data Warehouses », *ER (Workshops)*, p. 415-428, 2004.

Golfarelli M., Rizzi S., « Methodological Framework for Data Warehouse Design », *DOLAP*, p. 3-9, 1998.

Golfarelli M., Rizzi S., « A Survey on Temporal Data Warehousing », *IJDWM*, vol. 5, n° 1, p. 1-17, 2009.

Hurtado C. A., Mendelzon A. O., Vaisman A. A., « Maintaining Data Cubes under Dimension Updates », *ICDE*, p. 346-355, 1999.

Kimball R., *The Data Warehouse Toolkit : Practical Techniques for Building Dimensional Data Warehouses*, John Wiley, 1996.

Lee A. J., Nica A., Rundensteiner E. A., « The EVE Approach : View Synchronization in Dynamic Distributed Environments », *IEEE Trans. Knowl. Data Eng.*, vol. 14, n° 5, p. 931-954, 2002.

Letz C., Henn E. T., Vossen G., « Consistency in Data Warehouse Dimensions », *IDEAS*, p. 224-232, 2002.

Malinowski E., Zimányi E., « A conceptual model for temporal data warehouses and its transformation to the ER and the object-relational models », *Data Knowl. Eng.*, vol. 64, n° 1, p. 101-133, 2008.

Mendelzon A. O., Vaisman A. A., « Temporal Queries in OLAP », *VLDB*, p. 242-253, 2000.

Papastefanatos G., Vassiliadis P., Simitsis A., Vassiliou Y., « Policy-Regulated Management of ETL Evolution », *J. Data Semantics*, vol. 13, p. 147-177, 2009.

Ravat F., Teste O., Tournier R., Zurfluh G., « Graphical Querying of Multidimensional Databases », *ADBIS*, p. 298-313, 2007.

Ravat F., Teste O., Zurfluh G., « A Multiversion-Based Multidimensional Model », *DaWaK*, p. 65-74, 2006.

Rizzi S., Abelló A., Lechtenbörger J., Trujillo J., « Research in data warehouse modeling and design : dead or alive ? », *DOLAP*, p. 3-10, 2006.

Rizzi S., Golfarelli M., « X-Time : Schema Versioning and Cross-Version Querying in Data Warehouses », *ICDE*, p. 1471-1472, 2007.

Schlesinger L., Bauer A., Lehner W., Ediberidze G., Gutzmann M. M., « Efficiently Synchronizing Multidimensional Schema Data », *DOLAP*, 2001.

Sen A., Sinha A. P., « A comparison of data warehousing methodologies », *Commun. ACM*, vol. 48, n° 3, p. 79-84, 2005.

Vaisman A. A., Mendelzon A. O., « A Temporal Query Language for OLAP : Implementation and a Case Study », *DBPL*, p. 78-96, 2001.

Wrembel R., « A Survey of Managing the Evolution of Data Warehouses », *IJDWM*, vol. 5, n° 2, p. 24-56, 2009.

Wrembel R., Bebel B., « Metadata Management in a Multiversion Data Warehouse », *J. Data Semantics*, vol. 8, p. 118-157, 2007.