Les Entrepôts de Données Actifs : Modélisation des règles d'analyse

Sonia Bouattour*,**, Omar Boussaid**

*Laboratoire MIRACL,
Université de Sfax, Tunisie.
sonia.bouattour@eric.univ-lyon2.fr
**Laboratoire ERIC,
Université de Lyon, France.
omar.boussaid@univ-lyon2.fr

Résumé. L'entrepôt de données actif est un système permettant d'intégrer dans une même base cible des données et des traitements. Le mécanisme qui rend une telle approche active se base sur les règles d'analyse. Dans ce papier, nous proposons une formalisation simplifiée des règles d'analyse, appelées également règles actives, suivie d'une modélisation permettant d'assurer le stockage physique des ces règles dans l'entrepôt.

1 Introduction

Les entrepôts de données sont une solution aux besoins des entreprises. Ils ont été formalisés dans les années 90 par Inmon et Richard Hackathorn (1994). Leur finalité est de constituer une base de données orientée sujet, intégrée et contenant des informations historisées, non volatiles et destinées aux processus d'aide à la décision. Au delà de leur vocation de stockage de grandes masses de données, les entrepôts offrent également des moyens pour visualiser, résumer et explorer les données qui y sont stockées, et ce au moyen de la technologie d'analyse en ligne (*OLAP : On line Analytical Processing*). Actuellement, les entrepôts de données ne sont pas conçus pour automatiser le traitement des tâches du processus d'entreposage.

Les entrepôts de données actifs (EDAs), Thalhammer et al. (2001), permettent justement d'encapsuler des actions afin d'obtenir automatiquement par exemple des informations pour la prise de décision ou d'automatiser des tâches de rafraîchissement au niveau de l'ETL (*Extract Transform Load*). La technologie des EDAs se développe de plus en plus dans le monde du décisionnel. Elle répond à la croissance constante des volumes de données et aux besoins des utilisateurs. Un EDA est considéré comme un système de gestion d'affaires essentiel pour une organisation, il soutient la prise de décisions opérationnelles. D'un point de vue technique les EDAs s'inspirent des techniques utilisées dans les bases de données actives à savoir : les règles "Événement-Condition-Action" (ECA), Barros (2004), Dayal et al. (1995), Smith (1992). Grâce à ce mécanisme, il est possible alors de penser à stocker des données et des traitements dans un entrepôt à l'instar de l'approche objet. Il existe déjà dans la littérature un certain nombre de travaux sur les EDAs, comme nous l'exposerons dans la section qui suit.

Ce mécanisme d'ECA permet de construire des règles d'analyse, exprimées sous la forme de scripts, qui définissent des tâches d'analyse par exemple ou d'automatiser toute autre tâche du processus d'entreposage.

Un cube OLAP contient des données servant à des analyses. Cependant, il ne donne aucune indication sur les différentes opérations à exécuter pour aboutir à un résultat. D'ailleurs, ceci constitue une limitation des entrepôts classiques. L'exploration des données dans un cube OLAP nécessite une connaissance du métier pour déceler et bien comprendre la sémantique liée à certaines données agrégées. Sans cette connaissance, un utilisateur lambda n'est pas apte à interpréter la constance ou la variation des agrégats mensuels au cours d'une année dans un environnement de grande distribution par exemple. D'où l'intérêt d'intégrer des scénarios d'analyse dans un entrepôt pour aider un utilisateur à effectuer des analyses pertinentes.

Dans cet article, nous présentons d'abord en section 2 un état de l'art sur les EDAs. Nous montrerons ensuite dans la section 3 le positionnement d'une règle d'analyse dans un EDA. Nous exposons dans la section 4 une formalisation des règles d'analyse définissant ainsi une modélisation au niveau conceptuel. Nous évoquerons, dans la section 5, une modélisation logique des règles d'analyse en utilisant des schémas XML. Dans la section 6, nous présentons une étude de cas pour la validation de notre approche. Enfin, nous concluons en évoquant nos perspectives de recherche.

2 État de l'art

Dans la littérature, nous distinguons deux familles d'approches liées aux entrepôts de données actifs. La première porte sur la partie analyse du processus d'entreposage. Alors que la deuxième est plutôt orientée ETL. Parmi les premières approches, nous trouvons principalement les travaux de Thalhammer et al. (2001); Thalhammer et Schrefl (2002), qui portent sur l'automatisation de certaines tâches d'analyse en ligne. Ils utilisent des scripts pour définir des règles d'analyse dont le but est de modifier des valeurs de données sources dans le cadre d'une stratégie commerciale correspondant au démarquage de certains produits. Cependant, selon Thalhammer et *al.*, les entrepôts de données et le système OLAP traditionnels offrent un support limité pour automatiser les tâches de prise de décision qui se produisent fréquemment et pour lesquelles des procédures bien établies de décision sont disponibles, Thalhammer et al. (2001). Les auteurs considèrent les entrepôts de données conventionnels comme des entrepôts passifs. Ils partent d'une architecture classique d'un entrepôt de données et intègrent une couche supplémentaire qui permet de gérer des règles d'analyse. Ces dernières possèdent la même structure que les règles ECA.

Les règles d'analyse permettent une analyse en ligne complexe moyennant des opérateurs OLAP (e.g. *Roll up, Drill Down, Slice and dice...*). Thalhammer et *al.* proposent une démarche qu'ils qualifient de multidimensionnelle pour définir ces règles d'analyse et pour aider des décideurs à modifier les prix de certains produits (au niveau des bases de données sources) afin d'améliorer les ventes. Les règles d'analyse suivent une analyse multidimensionnelle utilisant le concept de graphes d'analyse. Ceux-ci sont une représentation conceptuelle des règles d'analyse faisant intervenir des vues multidimensionnelles, ou des cubes de données, Bouattour et al. (2007).

Dans la seconde famille d'approches, les travaux de Tho et Tjoa (2004), et ceux de Karakasidis et al. (2005), se concentrent sur la minimisation du temps d'attente entre l'intégration

continue des données et la consultation de ces données dans l'entrepôt. Les solutions avancées proposent d'accompagner les tuples de mise à jour par des messages porteurs d'indicateurs temporels. Le chargement des données doit s'effectuer sans interruption et doit se produire en temps réel permettant une consultation des données en parallèle.

Dans une autre approche, (Polyzotis et al., 2006), considèrent qu'un entrepôt de données actif est régénéré en ligne et réalise ainsi une uniformité dans le temps plus élevée entre l'information stockée et les dernières mises à jour de données. Le besoin de rafraîchissement en ligne de l'entrepôt présente plusieurs défis dans l'exécution des transformations de l'entrepôt. Ceux-ci sont en rapport avec les temps d'exécution et les coûts dans un processus d'entreposage.

L'approche adoptée par Bruckner et Tjoa (2001), dont le principe est de permettre des mises à jour en ligne dans l'entrepôt de données, se propose de modéliser l'intervention des nouveaux faits en temps réel. Cette approche permet de respecter le problème d'uniformité du temps tout en insistant sur la validité dans le temps de la composante "*Événement*" d'une règle d'analyse. La solution proposée par les auteurs consiste à satisfaire les trois propriétés du temps : *(i)* la validité dans le temps de l'information qui porte l'événement ; *(ii)* le temps de la transaction ; et *(iii)* le temps de la prise de décision automatique suite à l'événement. Ces propriétés permettent aux décideurs d'observer et de commander les mécanismes actifs des entrepôts de données. Nous nous intéressons à la première famille de travaux.

Thalhammer *et al.* ont avancé la notion de *règles d'analyse* et de *graphes d'analyse*. Cependant, cette proposition est développée sur la base d'un exemple d'analyse portant sur un problème spécifique à la vente de produits d'alimentation. Les scénarios programmés permettent de modifier les données au niveau des bases sources pour déployer une stratégie commerciale. Nous estimons que la formalisation originelle des règles d'analyse n'est pas simple et ne se prête pas à une utilisation intuitive par un utilisateur OLAP. Nous proposons de simplifier cette formalisation et de la rendre applicable à différents scénarios d'analyse. Notre approche se propose de généraliser celle de Thalhammer et *al.*, en donnant un caractère plus général aux règles et graphes d'analyse. L'objectif est de proposer une solution pour l'automatisation des scénarios d'analyse au sein d'un entrepôt de données. Les règles d'analyses deviennent des informations importantes et doivent être stockées dans l'entrepôt. Pour cela, nous proposons une modélisation conceptuelle pour simplifier la représentation d'une règle d'analyse et une modélisation logique pour permettre par la suite un stockage physique dans l'entrepôt.

Dans les sections qui suivent, nous allons définir les règles d'analyse et les situer dans un entrepôt de données actif.

3 Positionnement des règles d'analyse dans un EDA

L'intégration d'un volet actif dans un entrepôt par le biais des règles d'analyse a pour but de générer automatiquement des rapports d'analyse. Ceci revient à automatiser des règles d'analyse définies préalablement par un utilisateur d'une façon déclarative. Il ne s'agit pas de générer automatiquement des règles d'analyse, mais d'automatiser leur éxecution de façon contidionnelle. La définition des règles d'analyse se fait donc de façon déclarative par le biais d'une API permettant à l'utilisateur de définir la séquence d'opérations OLAP nécessaires à l'exécution d'un scénario d'analyse. Celui-ci exprimé à l'aide d'une règle d'analyse sera intégré dans l'entrepôt. L'exécution automatique du scénario se fait à l'aide du mécanisme inspiré

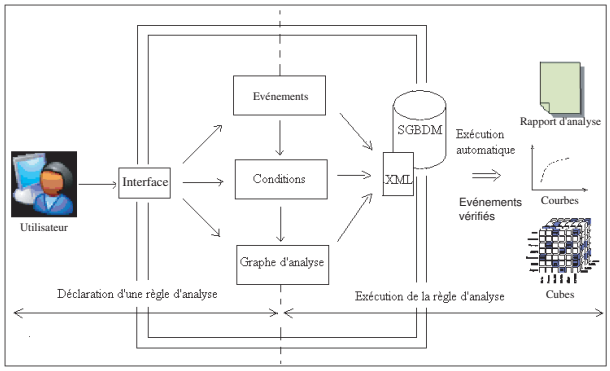

FIG. 1 – *Démarche générale d'un entrepôt de données actif*

de celui d'ECA. Nous avons fait appel au concept des règles actives comme modèle pour définir les règles d'analyse à partir de quoi nous avons construit notre approche d'automatisation de scénarios d'analyse. Un EDA ne peux être forcément perçu comme une base de données active. Un entrepôt de données classique est lié à une analyse en ligne (OLAP) qui consiste en des tâches exogènes. Il s'agit de représenter ces tâches et de permettre leur déclenchement automatiquement. Le point commun entre un EDA et une base de données active est l'usage d'un mécanisme similaire à celui d'ECA. Bien entendu les problèmes liés aux règles actives notamment la cohérence des règles vont se poser. Ils feront l'objet de nos travaux à venir.

Nous présenterons une définition d'une règle d'analyse dans la section 4.2.

3.1 Intégration des règles d'analyse dans le processus d'entreposage

Dans notre démarche, nous admettons que l'entrepôt de données est mis à jour. Cette tâche est en dehors du cadre de notre approche. Les données de l'entrepôt sont connues et les analyses porteront sur des éléments courants bien identifiés et représentés sous une forme multidimensionnelle par exemple (*faits, mesures, dimensions, hiérarchies...*). Dans Thalhammer et al. (2001), les auteurs déterminent trois situations possibles pouvant être automatisées. Nous nous plaçons dans le premier cas tel que nous l'expliquons ci-dessus. Il s'agit de tâches de décision courantes. Le processus d'aide à la prise de décision commence par une déclaration d'un scénario propre à un besoin d'analyse. A chaque scénario correspond une règle d'analyse qui doit être déclarée par les utilisateurs. Les règles déclarées sont stockées dans l'entrepôt et traiteés par le système de gestion de base de données multidimensionnelles comme une requête. La figure 1 présente l'architecture générale d'un entrepôt de données actif.

3.1.1 Déclaration d'une règle d'analyse

Pour exprimer une règle d'analyse, l'utilisateur doit tout d'abord définir les événements associés à cette règle. Un événement peut correspondre à un point fixe du calendrier (par exemple la fin du mois) ou bien à un changement d'état dans les bases de données de production (par exemple le chiffre d'affaire relatif à certains produits dépasse une valeur d'un seuil défini par

les utilisateurs). Ensuite, l'utilisateur définit les conditions relatives au scénario d'analyse. Par exemple, il peut préciser des conditions par rapport à la famille des produits, par rapport à l'année ou par rapport au magasin de ventes. En dernier lieu, l'utilisateur génère, à partir de la règle d'analyse, le graphe d'analyse qui décrit la suite d'opérations à exécuter. Ainsi, nous pouvons énumérer les étapes de la déclaration d'un scénario d'analyse relatif à une règle d'analyse comme suit : (i) définition des événements, (ii) définition des conditions et (iii) définition du graphe d'analyse.

3.1.2 Exécution d'une règle d'analyse

Les règles d'analyse déclarées sont décrites et stockées dans des documents XML. Le formalisme XML est une norme de W3C [1] et est considéré comme un standard dans la description, le stockage et l'échange des données. Les règles d'analyse se limitent actuellement à la formulation de scénarios de navigation. Nous prévoyons dans nos futurs travaux de leur permettre des spécifications de nouveaux calculs. Comme le montre la figure 1, les documents XML sont stockés dans l'entrepôt de données. Ils seront gérés, par la suite, par le système de gestion de base de données multidimensionnelles au même titre que les requêtes et les données. Le choix d'XML comme un langage de représentation de règles a pour principale raison de nous permettre de stocker les règles dans l'entrepôt. Les capcités de ce langage assurent une souplesse dans l'expression de ces règles et facilitent leur manipulation. Au moment du déclenchement des événements, une fois les conditions vérifiées alors les opérations OLAP décrites dans le graphe d'analyse sont exécutées. Les rapports d'analyse sont alors générés et peuvent être exploités pour la prise de décision.

3.2 Taxonomie des événements

Au cours de nos travaux, nous avons été amenés à étudier les différents événements et à les classer selon une taxonomie. Deux grandes catégories d'événements se présentent : (i) les événements OLTP ; et (ii) les événements OLAP Bouattour et al. (2008) .

3.2.1 Les événements OLTP

Les événements OLTP sont généralement liés aux opérations classiques de mise à jour dans les bases de données transactionnelles. L'insertion, la modification et la suppression des données sont des opérations au niveau de ces bases de données mais considérés comme des événements au niveau du processus d'ETL. Les événements dans les bases de données classiques dus à des erreurs constituent aussi un exemple d'événements ETL. Dans le cas des SGBDs relationnels, des déclencheurs sont utilisés pour lancer l'exécution de traitements dès l'apparition de certains événements dans une base de données opérationnelle.

Nous pouvons également associer à une procédure un événement pour représenter la phase "*transformation*" d'un processus ETL. Nous pouvons lier l'évolution des schémas des sources OLTP (module source) avec celle des schémas de l'entrepôt (module destination). Le "*chargement*" constitue la phase d'alimentation de l'entrepôt. Il peut se déclencher suite à une insertion, une modification ou une suppression dans les sources OLTP représentant ainsi un

[1] http ://www.w3c.org

événement relatif aux sources ou suite à un événement récurant (tous les dimanches, tous les 10 minutes, etc.).

Enfin, nous remarquons également une autre catégorie d'événements qui peuvent se déclencher suite à une fréquence importante d'un certain nombre d'opérations. Cette fréquence peut être anormale. Une telle information peut être extraite lors d'une étude du processus métier. Dans Oracle, cette information est extraite en utilisant le fichier *log* (journal) et ceci en calculant les différents enregistrements effectués dans l'entrepôt. Dans ce cas, une modélisation conceptuelle des données du *log* mérite d'être faite.

3.2.2 Les événements OLAP :

Nous distinguons deux catégories d'événements OLAP : les événements résultats et les événements temporels.

Dans la première catégorie, nous distinguons les événements qui permettent d'enrichir les schémas en étoile ou en constellation et diversifier ainsi les analyses. Ceci peut arriver lorsqu'un nouveau besoin est constaté lors d'une analyse d'une table de faits. L'évaluation de la qualité des résultats dans un cube peut donner lieu à différents événements. Dans le cas d'un cube de données éparses, des indications sur les résultats peuvent être données à l'utilisateur en lui recommandant de faire appel à des opérations OLAP en pré-traitements avant d'entamer l'exploration de celui-ci. L'interprétation des résultats peut amener à générer des événements pouvant déclencher des actions soit au niveau de l'entrepôt, soit au niveau des bases sources.

Dans les EDAs, les événements temporels déterminent l'intervalle de temps d'observation pendant lequel les données nécessaires aux analyses sont recueillies dans l'entrepôt. Les événements temporels sont utilisés pour déterminer quand des règles d'analyse doivent être exécutées. Il existe trois types d'événements temporels à savoir : (i) les événements temporels absolus, (ii) les événements temporels périodiques et (iii) les événements temporels relatifs.

Par exemple, un événement temporel absolu est une date du calendrier : *20 décembre 2008* ; un événement temporel périodique peut être *la fin de chaque trimestre*. Les événements temporels relatifs représentent des points temporels positionnés par rapport à un événement donné. "*20 jours après le lancement d'un produit*" représente un événement temporel relatif au lancement d'un produit. Le lancement du produit est un événement d'une méthode qui existe dans les sources OLTP. Les occurrences des événements des méthodes OLTP doivent être toutes importées dans l'entrepôt avant le début de la détection des événements temporels relatifs.

4 Formalisation des règles d'analyse

Dans cette section, nous commençons d'abord par présenter les notations et la terminologie utilisées dans le cadre de la formalisation d'une règle d'analyse. Nous utilisons un modèle de cube de données et une algèbre d'opérateurs OLAP déjà définis dans Ben Messaoud (2006).

4.1 Notations et terminologie

4.1.1 Cubes de données

Notons $\mathcal{U}$ un ensemble fini et non vide de cubes de données. Nous définissons un cube de données $U \in \mathcal{U}$ selon le quintuplet $\langle \mathcal{C}, \mathcal{A}, \mathcal{M}, \mathcal{L}, \mathcal{F} \rangle$, où :

1ière Famille : gestion de la structure d'un cube	**2ième Famill**e : manipulation du contenu d'un cube	**3ième Famille** : opérateurs n-aires
-ADDCHARACTERISTIC : ajout d'une caractéristique	-ROLLUP : forage vers le haut	-UNION : union de deux cubes de données
-DLTCHARACTERISTIC : suppression d'une caractéristique	-DRILLDOWN : forage vers le bas	-INTERSECT : intersection de deux cubes de données
-NEST : imbrication d'un attribut	-SWITCH : permutation de modalités	-DIFFERENCE : différence de deux cubes de données
-PUSH : conversion d'une dimension	-SLICE : sélection de modalités	
-PULL : conversion d'une mesure	-AGGREGATE : calcul d'agrégat	
-ADDMODALITY : l'ajout d'une modalité		

TAB. 1 – *Tableau des différentes opérations OLAP*

- $\mathcal{C} = \langle C, d \rangle$ est un espace de caractéristiques. C est un ensemble de *noms de caractéristiques* et d est une fonction booléenne qui partitionne C en un sous-ensemble de *caractéristiques dimensionnelles* et un sous-ensemble de *caractéristiques métriques* ;
- $\mathcal{A} = \langle A, f, O_A \rangle$ est un espace d'attributs. A est un ensemble de *noms d'attributs*. f est une bijection qui fait correspondre à chaque caractéristique $c \in C$ un sous-ensemble d'attributs dans A. O_A est un ensemble d'*ordres partiels*, où chaque ordre exprime un *ordre hiérarchique* entre les attributs de A ;
- $\mathcal{M} = \langle M, g, O_M, h \rangle$ est un espace de modalités. M est un ensemble de *noms de modalités*. g est une fonction qui associe à chaque attribut $a \in A$ un sous-ensemble non vide de modalités dans M. g permet d'identifier pour chaque attributs de A son domaine de définition. O_M est un ensemble d'*ordres totaux*, où chaque ordre intervient dans un sous-ensemble d'attributs $g(a)$. Cet ordre correspond à un ordonnancement lexical des modalités d'un niveau de granularité d'une caractéristique. h est une *fonction d'appartenance* qui associe à une modalité un ensemble de modalités qui lui appartiennent selon un *ordre sémantique* ;
- $\mathcal{L}$ est un ensemble fini et non vide de cellules ;
- $\mathcal{F}$ est un ensemble fini et non vide de fonctions d'agrégation comme la somme (*SUM*) et la moyenne (*AVG*).

4.1.2 Opérateurs OLAP

L'algèbre d'opérateurs OLAP que nous considérons se compose de trois familles d'opérateurs (c.f. le tableau des différents opérateurs OLAP, Tab. 1).

4.2 Définition d'une règle d'analyse

Nous définissons une règle d'analyse comme suit :

Règle d'analyse : *Une règle d'analyse, notée R, est un triplet de la forme* $\langle \mathcal{E}, \mathcal{N}, G \rangle$*, où :*

- *$\mathcal{E}$ est une conjonction d'événements ;*
- *$\mathcal{N}$ est un ensemble de conditions ;*
- *G est un graphe d'analyse.*

4.2.1 Conjonction d'événements

La conjonction d'événements $\mathcal{E}$ définit le déclenchement d'une règle d'analyse. Elle s'écrit sous la forme d'une conjonction de disjonctions d'un ensemble fini d'événements $\{E_1, E_2, \ldots, E_n\}$. Nous pouvons supposer qu'un événement E_i peut être un point fixe du calendrier ou bien un changement d'état dans une base de données OLTP. A titre d'exemple, dans une entreprise commerciale, considérons l'ensemble des événements $\{E_1, E_2\}$ où E_1 représente la fin de chaque mois et E_2 est l'événement correspondant à un chiffre d'affaire inférieur à 1000 euros. Ainsi, à partir de cet ensemble d'événements, nous pouvons obtenir la conjonction d'événements $\mathcal{E} = E_1 \wedge E_2$. Cette conjonction est responsable du déclenchement d'une analyse due au fait que le chiffre d'affaire de l'entreprise est inférieur à 1000 euros à la fin du mois. Cette analyse pourrait expliquer par exemple les raisons de la décroissance du chiffre d'affaire de l'entreprise.

4.2.2 Ensemble de conditions

L'ensemble $\mathcal{N}$ contient un nombre fini de conditions $\{N_1, N_2, \ldots, N_n\}$ qui vont intervenir dans le scénario d'analyse. Ces conditions répondent aux besoins d'analyse et représentent les clauses d'exécution de la requête avant qu'une décision ne soit prise. Par exemple, en se basant sur un cube des ventes concernant une entreprise commerciale, on peut considérer l'ensemble des conditions $\{N_1, N_2, N_3\}$ où : N_1 est une sélection sur la famille des produits ("*MP3*") ; N_2 est une sélection sur le pays ("*Espagne*") et N_3 est une restriction sur la quantité des produits vendus qui doit être supérieure à *300 unités*.

4.2.3 Graphe d'analyse

Un graphe d'analyse G est une représentation graphique d'un processus d'analyse. Il est de la forme $\langle \mathcal{U}, \mathcal{P} \rangle$, où $\mathcal{U}$ est un ensemble de cubes de données et $\mathcal{P}$ est un ensemble de chemins d'analyse représentant des opérations OLAP. $\mathcal{P}$ est un ensemble fini de r chemins d'analyse $\{P_1, P_2, \ldots, P_r\}$. Nous définissons un chemin d'analyse comme suit :

Chemin d'analyse : *Un chemin d'analyse, noté P, est une fonction de $\mathcal{U}^n$ dans $\mathcal{U}$ qui fait associer à un ou à plusieurs cubes de données un autre cube $P(U)$ qui résulte d'une opération OLAP. Formellement, nous écrivons :*

$$\begin{array}{rcl} P & : & \mathcal{U}^n \longrightarrow \mathcal{U} \\ & & (U_1, U_2, \ldots, U_n) \longmapsto P(U) \end{array}$$

Graphiquement, comme le montre la figure 2, nous représentons un chemin d'analyse par un lien orienté d'un ensemble de cubes de départ vers un cube résultat $P(U)$.

Un chemin d'analyse représente une ou plusieurs opérations OLAP. Nous utilisons l'ensemble des opérateurs OLAP (c.f. Tab. 1) pour définir des graphes d'analyse. D'un point de

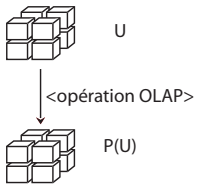

FIG. 2 – *Représentation graphique d'un chemin d'analyse*

vue graphique, un graphe d'analyse G est schématisé par un ensemble de nœuds $\mathcal{U}$ (cubes de données) et un ensemble d'arcs $\mathcal{P}$ (chemins d'analyse). Un graphe d'analyse permet de représenter de façon graphique un processus d'analyse dans le but de produire automatiquement par exemple des rapports d'analyse (*reportings*). Il permet d'exprimer un scénario d'analyse formulé sous la forme d'une règle d'analyse et l'automatise à l'aide du mécanisme ECA. C'est ce qui donne ainsi un caractère actif à un entrepôt de données.

Sous-graphe d'analyse Afin de répondre à des besoins de la modélisation d'un graphe d'analyse, que nous aborderons dans la section suivante, nous introduisons la notion d'un sous-graphe d'analyse.

Définition : *Un sous-graphe d'analyse, noté SG et ayant la forme $\langle \mathcal{U}', \mathcal{P}' \rangle$, est une sous-partie d'un graphe d'analyse $G = \langle \mathcal{U}, \mathcal{P} \rangle$, où $\mathcal{U}' \subset \mathcal{U}$ et $\mathcal{P}' \subset \mathcal{P}$. En plus, SG vérifie les propriétés suivantes :*

- *SG commence par un cube d'entrée dans G, ou deux cubes intervenant dans une opération binaire ;*
- *SG finit par un cube de sortie dans G, ou un cube intervenant dans une opération binaire.*

La figure 3 de la section 6 illustre une graphe d'analyse composé de trois sous-graphes.

Soulignons que la définition du sous-graphe que nous proposons est plus spécifique que celle connue dans la théorie des graphes. Il s'agit d'un découpage du graphe d'analyse adapté à des fins que nous expliquerons dans la section qui suit.

5 Modélisation logique d'une règle d'analyse

Pour représenter une règle d'analyse, nous avons opté d'utiliser le formalisme XML. Une règle d'analyse est exprimée par trois balises principales : <Events>, <Conditions> et <AnalysisGraph> représentant respectivement les événements, les conditions et le graphe d'analyse. Un graphe d'analyse est composé d'un ou de plusieurs sous-graphes d'analyse représentés par la balise <SubGraph>. Dans la Table 1, nous avons décrit une règle d'analyse à l'aide de ses trois éléments constitutifs. A savoir les événements, les conditions et le graphe d'analyse qui peut être constitué de plusieurs sous-graphes contenant chacun les données (cubes) et les opérations OLAP à utilser (chemins d'analyse). Ces différents éléments sont modélisés par un schéma XML représenatnt un modèle logique d'une règle d'analyse.

5.1 Schéma logique d'une règle d'analyse

```
<?xml version="1.0"
encoding="ISO-8859-1"?> <xsd:schema
xmlns:xsd='http://www.w3.org/2001/XMLSchema'> <!-- Déclarations des
éléments de type simple -->
    <xsd:element name="where" type="xsd:string"/>
    <xsd:element name="when" type="xsd:string"/>
    <xsd:element name="inputCubes" type="xsd:string"/>
    <xsd:element name="nameOperator" type="xsd:string"/>
    <!-- Déclarations des éléments de type complexe -->
    <xsd:element name="AnalysisRule">
        <xsd:complexType>
            <xsd:attribute name="ruleName" type="xsd:string" use="required"/>
            <xsd:sequence>
                <xsd:element ref="Event"/>
                <xsd:element ref="Condition"/>
                <xsd:element ref="AnalysisGraph"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="Event">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="where"/>
                <xsd:element ref="when"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="Condition">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="where"/>
                <xsd:element ref="when"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AnalysisGraph">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="SubGraph" minOccurs="1" maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SubGraph">
        <xsd:complexType>
            <xsd:attribute name="SubGraphName" type="xsd:string" use="required"/>
            <xsd:sequence>
                <xsd:element ref="inputCubes"/>
                <xsd:element ref="AnalysisPath"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AnalysisPath">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="Path" minOccurs="1" maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="Path">
        <xsd:complexType>
            <xsd:attribute name="number" type="xsd:integer" use="required"/>
            <xsd:sequence>
                <xsd:element ref="nameOperator"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
```

```
</xsd:schema>
```

TABLE 1 - *Schéma XML représentant une modélisation logique d'une règle d'analyse*

Ce schéma XML permet de générer un document XML qui contient la description des différents éléments d'une règle d'analyse. A savoir un ou plusieurs événements, une ou plusieurs conditions et un graphe d'analyse pouvant être composé de sous-graphes. Ces graphes représentent les données (cubes) et les opérations OLAP, sous la forme de chemins d'analyse, et transcrivent ainsi l'analyse à effectuer automatiquement.

5.2 Modélisation XML des opérateurs OLAP

Opération OLAP	Modélisation XML
ADDCARACTERISTIC	<AddCaracteristic caracteristic= "..."/>
DLTCARACTERISTIC	<DltCaracteristic caracteristic= "..."/>
NEST	<Nest caracteristic="..." attribute_new="..."/>
PUSH	<Push attribute_new = "..."/>
PULL	<Pull attribute = "..."/>
ADDMODALITY	<AddModality modality_new = "..." attribute = "..."/>
ROLLUP	<RollUp caracteristic = "..." attribute_old = "..." attribute_new = "..."/>
DRILLDOWN	<DrillDown caracteristic = "..." attribute_old = "..." attribute_new = "..."/>
SWITCH	<Switch attribute = "..." modality_1 = "..." modality_2 = "..."/>
SLICE	<Slice caracteristic = "..." Predicate = "..." />
AGGREGATE	<Aggregate caracteristic = "..." modality = "..."/>
UNION	<Union/>
INTERSECT	<Intersect/>
DIFFERENCE	<Diff/>

TAB. 2 – *Modélisation XML des opérateurs OLAP*

Le tableau (Tab.2) présente une modélisation XML des différents opérateurs OLAP munis de leurs paramètres (c.f. la section 4.1.2). Pour les opérateurs n-aires, seul le nom de l'opérateur est représenté dans la balise. Ils n'ont pas de paramètres puisque les cubes participant à l'opération binaire sont représentés dans la balise <inputcubes> en tant que cubes d'entrées.

Les sous-graphes d'analyse permettent une formalisation physique cohérente. En effet, lorsqu'il s'agit de modéliser une opération binaire dans un graphe d'analyse un problème de séquencement peut avoir lieu en ce qui concerne les cubes participant à cette opération. La notion de sous-graphe permet de pallier à ce problème en découpant le graphe d'analyse en plusieurs parties qui s'enchaînent. Ainsi, le cube résultat d'un sous-graphe d'analyse peut représenter le cube d'entrée d'un autre sous-graphe. Dans notre modélisation XML, nous utilisons la valeur spécifique ResultOf :[SubGraph_name] pour désigner le cube résultat du sous-graphe d'analyse [SubGraph_name]. Cette valeur spécifique peut alors faire l'objet d'un cube d'entrée dans un autre sous-graphe d'analyse.

6 Etude de cas

Pour valider notre approche, nous avons réalisé une étude de cas pour montrer l'intérêt des modèles que nous avons proposés dans le cadre de notre approche.

6.1 Exemple d'une règle d'analyse

Nous partons d'un exemple de cube de données extrait d'Oracle. Ce cube contient des données sur les salariés d'une entreprise. Nous envisageons d'automatiser à l'aide d'une règle d'analyse le scénario qui permet de comparer les salaires des employés pour les mois de janvier de 2007 et 2008 et voir lesquels ont eu un salaire supérieur à 3000 euros dans le département numéro 1. Pour modéliser une telle règle d'analyse, nous définissons l'événement correspondant, la condition de déclenchement et enfin le graphe d'analyse qui indique les opérations OLAP à appliquer pour mettre en œuvre une telle analyse.

L'événement correspondant à notre règle d'analyse est de type temporel et désigne la fin du mois de janvier. Les conditions relatives à la règle d'analyse sont : $Année = 2008$; $Année = 2007$; $Salaire\ supérieur$ à 3000 $euros$ et $Département = 1$. La figure 3 présente le graphe d'analyse associé à la règle d'analyse présentée. Celui-ci est composé de trois sous-graphes.

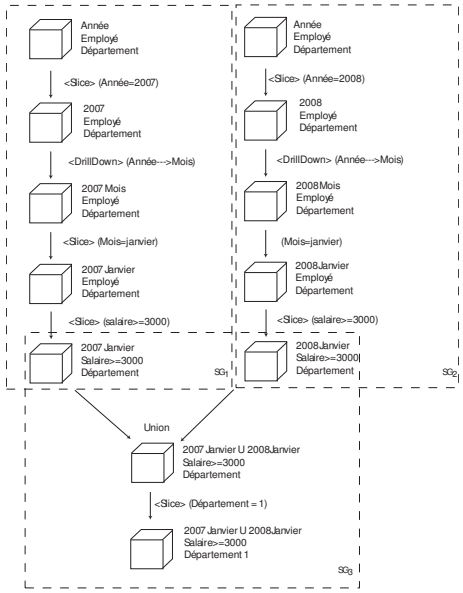

FIG. 3 – *Graphe d'analyse associé à l'exemple*

1. Le premier sous-graphe SG_1 commence par un cube représentant les employés par département et par année. Ce sous-graphe suit un enchaînement d'opérations OLAP pour obtenir un cube qui représente les employés ayant un salaire supérieur à 3000 euros en janvier 2007. Celui-ci va intervenir dans une opération binaire (UNION) dans le sous-graphe suivant.

2. Parallèlement, le deuxième sous-graphe SG_2 commence par un cube représentant les employés par département et par année. Ce sous-graphe suit un enchaînement d'opérations OLAP pour obtenir un cube qui représente les employés ayant un salaire supérieur à 3000 euros en janvier 2008. Celui-ci va intervenir dans une opération binaire (UNION) dans le sous-graphe suivant.

3. Le troisième sous-graphe SG_3 commence, quant à lui, par les deux cubes finaux des deux précédents sous-graphes. Il permet d'obtenir un cube contenant les données sur les employés qui touchent un salaire supérieur à 3000 euros en janvier 2007 et en janvier 2008. Ce cube permet ainsi de procéder à la comparaison des salaires comme le stipuler le scénario d'analyse initial.

Le modèle logique de la règle d'analyse initiale est décrit par un document XML généré à partir du schéma XML de la TABLE 1. Ce document XML (Table 2) décrit l'événement (<event>), les trois parties composant la condition (<conditions>) et le graphe d'analyse (<AnalysisGraph>) qui est composé de trois sous-graphes (<SubGraph>) avec leurs différents chemins d'analyse (<AnalysisPath>).

6.2 Modèle XML de la règle d'analyse

```
<?xml version="1.0" encoding="ISO-8859-2" ?>
 <AnalysisRule name = "Comparaison des salaires des employés>
    <events>
        <event E1="E1">
            <when>T1 = fin janvier</when>
        </event>
    </events>
    <conditions>
        <when>Annee = 2007</when>
        <when>Annee = 2008</when>
        <when>DeptNo = 1</when>
        <where> </where>
    </conditions>
    <AnalysisGraph>
        <SubGraph name="I">
            <input_cubes>
                <cube> name="cube des employés par département et par année " </cube>
            </input_cubes>
            <AnalysisPaths>
                <path number="1">
                    <Slice caracteristic="Time" predicate = " Year=2007 " />
                </path>
                <path number="2">
                    <DrillDown caracteristic="Time" attribute_old="Year" attribute_new="Month" />
                </path>
                <path number="3">
                    <Slice caracteristic="Time" predicate = " Month = "January" " />
                </path>
                <path number="4">
                    <Slice caracteristic="Employee" predicate = " Sal >= 3000 " />
                </path>
            </AnalysisPaths>
        </SubGraph>
        <SubGraph name="II">
            <input_cubes>
                <cube> name="cube des employés par département et par année " </cube>
            </input_cubes>
            <AnalysisPaths>
```

```
                <path number="1">
                    <Slice caracteristic="Time" predicate = " Year=2008 " />
                </path>
                <path number="2">
                    <DrillDown caracteristic="Time" attribute_old="Year" attribute_new="Month" />
                </path>
                <path number="3">
                    <Slice caracteristic="Time" predicate = " Month = "January" " />
                </path>
                <path number="4">
                    <Slice caracteristic="Employee" predicate = " Sal >= 3000 " />
                </path>
                </AnalysisPaths>
        </SubGraph>
        <SubGraph name="III">
            <input_cubes>
                <cube> name="ResultOf:SubGraph I"
            </cube>
            <input_cubes>
                <cube> name="ResultOf:SubGraph II" </cube>
            </input_cubes>
            <AnalysisPaths>
                <path number="1">
                    <Union />
                </path>
                <path number="2">
                    <Slice caracteristic="Employee" predicate = " DeptNo = 1 " />
                </path>
            </AnalysisPaths>
        </SubGraph>
    </AnalysisGraph>
</AnalysisRule>
```

TABLE 2 - *Modélisation XML de la règle d'analyse*

Cette règle d'analyse décrite par le document XML peut-être alors stockée physiquement dans l'entrepôt de données.

7 Conclusion

Rappelons que notre objectif est de proposer une formalisation simplifiée et généralisant l'utilisation des règles d'analyse dans le cadre des entrepôts de données actifs. En se basant sur le formalisme des règles ECA utilisé dans les bases de données actives, nous avons défini un nouveau cadre formel pour les règles d'analyse. Nous avons transformé ce mécanisme ECA en un autre mécanisme ECG (*Événements-Conditions-Graphe d'analyse*). En nous servant d'un modèle multidimensionnel et d'une algèbre OLAP existante que nous avons complétée par des opérations binaires sur des cubes, Bouattour (2007), nous avons généralisé les règles d'analyse en transformant le mécanisme ECA en ECG. Nous avons d'autre part modélisé ces règles à un niveau logique par un schéma XML. Facilitant ainsi leur stockage physique dans l'entrepôt sous forme de documents XML. Cette approche a été validée par une étude de cas démontrant la possibilité de modéliser un scénario d'analyse à l'aide des modèles proposés.

Les travaux réalisés ouvrent diverses perspectives de recherche. Nous pensons, pour ce faire, rajouter un parseur qui permettra d'interpréter des scénarios d'analyse modélisés en XML et de les transformer en des requêtes OLAP de base. Nous envisageons également d'élargir nos travaux, sur les EDAs, aux différentes tâches du processus d'ETL.

Références

Barros, J. (2004). Etat de l'art des bases de données actives. In *IHM'04*.

Ben Messaoud, R. (2006). *Couplage de l'analyse en ligne et de la fouille de données pour l'exploration, l'agrégation et l'explication des données complexes*. Ph. D. thesis, Université Lyon2, France.

Bouattour, S. (2007). Les entrepôts de données actifs : automatisation des scénarii d'analyse. Master's thesis, Université Lyon2, France.

Bouattour, S., R. Ben Messaoud, et O. Boussaid (2007). Modélisation de règles d'analyse dédiées aux entrepôts de données actifs. In *Deuxième Atelier des Systèmes Décisionnels (ASD'2007), Sousse, Tunisie. Octobre*.

Bouattour, S., J. Feki, H. Ben Abdallah, R. Ben Messaoud, et O. Boussaid (2008). Proposition d'une taxonomie d'événement dans les entrepôts de données actifs. In *Les huitième journée de GEI 2008, Sousse, Tunisie.*

Bruckner, R. M. et A. M. Tjoa (2001). Managing Time Consistency for Active Data Warehouse Environments. *Lecture Notes in Computer Science 2114*, 254–263.

Dayal, U., E. N. Hanson, et J. Widom (1995). Active Database Systems. In *Modern Database Systems*, pp. 434–456.

Inmon, W. H. et D. Richard Hackathorn (1994). *Using the Data Warehouse*. Somerset, NJ, USA : Wiley-QED Publishing.

Karakasidis, A., P. Vassiliadis, et E. Pitoura (2005). ETL Queues for Active Data Warehousing. In *IQIS'05 : Proceedings of the 2nd international workshop on Information quality in information systems*, New York, NY, USA, pp. 28–39. ACM Press.

Polyzotis, N., S. Skiadopoulos, P. Vassiliadis, A. Simitsis, et N.-E. Frantzell (2006). Supporting Streaming Updates in an Active Data Warehouse. In *Personal and Ubiquitous Computing*, pp. 55–68.

Smith, K. P. (1992). *Managing Rules in Active Databases*. Ph. D. thesis, Champaign, IL, USA.

Thalhammer, T. et M. Schrefl (2002). Realizing Active Data Warehouses with Off-The-Shelf Database Technology. *Softw. Pract. Exper. 32*(12), 1193–1222.

Thalhammer, T., M. Schrefl, et M. K. Mohania (2001). Active Data Warehouses : Complementing OLAP with Analysis Rules. *Data Knowledge Engineering 39*(3), 241–269.

Tho, M. N. et A. M. Tjoa (2004). Zero-Latency Data Warehousing For Heterogeneous Data Sources and Continuous Data Streams. In *Services Computing*, pp. 357– 365.

Summary

The active data warehouse is a system for integrating in the same target database data and treatments. The mechanism that makes such an active approach is based on analysis rules . In this paper we propose a simplified formalization of the analysis rules, also called active rules, followed by modeling to ensure them a physical storage into a data warehouse.