OLAP : un pas vers la navigation

Arnaud Giacometti, Patrick Marcel, Elsa Negre

Université François-Rabelais de Tours
Laboratoire d'Informatique, Campus de Blois
3 place Jean Jaurès, 41000 Blois
{Arnaud.Giacometti,Patrick.Marcel,Elsa.Negre}@univ-tours.fr

Résumé. Dans le contexte de l'analyse OLAP, le concept de navigation n'a jamais été défini formellement. Nous montrons pourquoi cette lacune est préjudiciable. Nous proposons ensuite une formalisation du concept de navigation, ainsi qu'une première ébauche de langage de définition de navigations.

1 Introduction

1.1 Contexte

Dans le contexte de l'analyse en ligne de données multidimensionnelles (OLAP), il n'existe toujours pas de modèle et langage de référence pour la manipulation de cubes de données. Tout au plus a-t-on une définition plus ou moins rigoureuse de certaines primitives. En conséquence, certains traitements typiques restent difficiles à exprimer.

1.2 Problème

De notre point de vue, une des raisons à la difficulté d'exprimer des analyses OLAP, constituées d'un enchaînement de plusieurs requêtes, est que les opérations de manipulation de cubes de données sont définies indépendamment de tout contexte particulier. Ceci est lié au fait que tous les langages de manipulation de cubes de données (que ce soit SQL, MDX Microsoft Corporation (1998), voire les langages formels proposés par exemple par Agrawal et al. (1997); Gyssens et Lakshmanan (1997); Cabibbo et Torlone (1997); Vassiliadis et Skiadopoulos (2000); Pedersen et al. (2001); Franconi et Kamble (2004)) sont sans état : les opérations sont définies sans référence à un contexte particulier.

Dans ce travail, nous assimilons la notion de contexte à celle de *navigation*, une navigation étant l'ensemble des opérations réalisées lors d'une analyse (Dittrich et al., 2005). Nous proposons de définir les navigations comme des objets pouvant être manipulés, et nous définissons un langage de définition de navigations. L'archivage de navigations permettra, par exemple, de rejouer une analyse sur des données différentes, sans avoir nécessairement à soumettre l'intégralité des requêtes constituant cette analyse. Les concept et langage de définition de navigations que nous proposons permettent d'apporter des réponses aux problèmes soulevés dans les exemples suivants.

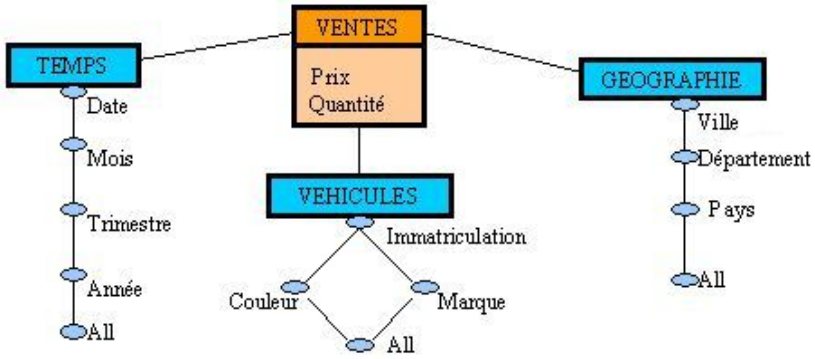

FIG. 1 – *Le cube utilisé pour les exemples*

1.3 Exemples de motivation

Dans ces exemples, nous nous référons aux définitions intuitives des opérations OLAP classiques (comme roll-up, drill-down ou slice). Les données utilisées dans ces exemples sont celles du cube décrit par la figure 1 (nous utilisons la notation de Golfarelli et Rizzi (1998)).

Exemple 1 : Comment revenir en arrière ? Considérons la séquence de requêtes suivante : une sélection de certaines villes d'un département, puis un changement de niveau de ville vers département, et enfin un retour au niveau des villes. Que cherche-t-on à obtenir :

1. toutes les villes du cube (ce qui correspondrait à une opération de drill-down), ou
2. uniquement les villes sélectionnées précédemment ?

Le deuxième cas ci-dessus illustre l'intérêt de pouvoir exprimer l'action de revenir en arrière au cours de l'analyse sur, par exemple, le résultat de l'opération précédente. En effet, les langages actuels ne permettent pas d'exprimer facilement cette action, autrement qu'en réécrivant complètement la séquence de requêtes menant au résultat attendu.

Exemple 2 : comment rejouer une analyse ? Supposons que l'analyste ait appliqué la séquence de requêtes suivante sur le cube : sélection des pays *France* et *Espagne*, puis sélection de l'année *2005* et enfin agrégation pour faire la somme des ventes. Ensuite, il décide de revenir au cube pour sélectionner le pays *Japon*. Il souhaite maintenant appliquer à ce pays le traitement appliqué aux pays *France* et *Espagne* (la sélection de l'année *2005* et l'agrégation).

Les langages actuels ne permettent pas d'exprimer facilement ce traitement, autrement qu'en réécrivant complètement la requête à évaluer sur le cube.

Exemple 3 : comment changer une opération d'une analyse ? Reprenons le début de l'exemple précédent : l'analyste a sélectionné les pays *France* et *Espagne*, puis l'année *2005* et enfin agrégé pour faire la somme des ventes. Il souhaite maintenant voir cette somme de ventes pour l'année *2004* (en conservant la sélection des pays).

Les langages actuels ne permettent pas d'exprimer facilement ce traitement, autrement qu'en réécrivant complètement la requête à évaluer sur le cube.

Exemple 4 : quelles sont les requêtes à évaluer ? Dans le cas de l'exemple précédent où seule une des opérations d'une analyse change, les systèmes ROLAP actuels soumettent intégralement la requête au SGBD sous-jacent. Supposons par contre qu'il soit possible de préciser quelle est, parmi une composition d'opérations, l'opération qui change. Il devient trivial de trouver sur quel résultat intermédiaire, gardé en cache, l'opération modifiée peut être évaluée, par simple comparaison syntaxique (sans avoir à tester par exemple l'inclusion de requêtes).

Dans le cas de l'exemple précédent, la somme des ventes pour l'année 2004 peut être calculée sur le résultat de la sélection des pays *France* et *Espagne*, plutôt que sur le cube.

1.4 Contribution

Nos contributions sont les suivantes :

- nous proposons une formalisation du concept de navigation. Nous donnons ainsi un statut à un ensemble de requêtes, qui pourront être stockées, réévaluées, partagées entre analystes, etc.
- nous proposons des opérations de définition de navigation, permettant :
  - d'exprimer le retour sur des résultats déjà obtenus ;
  - de faciliter l'expression de traitements sophistiqués, comme rejouer une analyse, ou substituer dans une analyse une opération par une autre.

Certains des problèmes abordés dans ce travail ont été soulevés par Dittrich et al. (2005), sans que des réponses précises soient proposées. A notre connaissance, il n'existe pas de prise en compte de la navigation dans les propositions de langages de manipulation de cubes, si ce n'est au travers des opérations "inverses" l'une de l'autre (comme roll-up et drill-down). Notons que certains langages, comme MDX proposé par Microsoft Corporation (1998), ne possèdent même pas la propriété de clôture.

1.5 Plan

Dans la section suivante, nous donnons une définition intuitive de la navigation dans le contexte OLAP. La section 3 présente notre cadre formel et nous concluons dans la section 4. Les figures relatives aux exemples de navigations sont regroupées dans l'annexe.

2 Navigation dans le contexte OLAP

2.1 Constat

Le terme "navigation" est emprunté au domaine de l'internet, où il est défini comme la possibilité de passer d'une page à l'autre, d'un site à l'autre, d'aller et venir au gré des liens hypertextes contenus dans les pages consultées sur le Web.

Dans le cadre d'une analyse à but décisionnel typique, un décideur interroge un cube, en lançant des requêtes dont il visualise le résultat sous forme de cross-tab. Comme le montrent les exemples de la section 1, il peut s'avérer que ses besoins en données changent au cours de

cette analyse. Il n'est pas rare de voir un décideur relancer toutes les requêtes de son analyse (ou du moins, une grande partie) parce qu'il se rend compte qu'il a écarté certaines données trop tôt.

Ainsi, dans ce contexte, la navigation doit permettre :

– d'explorer les données contenues dans un cube ;
– de revenir à des données déjà vues pour répondre à une nouvelle requête ;
– de faire référence à des requêtes dont le résultat a déjà été vu ;
– de se déplacer parmi tous les chemins empruntés lors de l'analyse.

Comme dit dans l'introduction, l'utilisation de langages de manipulation de données sans état, qui ne permettent donc pas de faire référence au contexte, ne permet pas de donner un statut à ce type de navigation.

2.2 Notre approche

Pour combler cette lacune, nous proposons un langage à états, où navigation et interrogation sont mélées (comme dans le travail de Mukhopadhyay et Papakonstantinou (2002) dans le cadre des documents XML).

Dans ce travail, nous avons choisi de représenter une navigation par une structure arborescente. Ce type de représentation n'est pas nouveau, mais son application à la navigation OLAP nous parait appropriée. Ainsi, chaque nœud est composé d'un cube et de l'opération qui a permis de le générer. Notons qu'un même cube peut être contenu dans plusieurs nœuds, s'il a été généré à différents moments de l'analyse (la suppression de telles redondances nécessiterait de tester des équivalences entre requêtes). Plus précisément, nous définissons une navigation comme un arbre, dotée d'un nœud particulier représentant le cube courant, c'est-à-dire le dernier visualisé.

Le langage de définition proposé permet de produire de nouveaux cubes (pour explorer les données du cube initial), ou bien de se déplacer parmi les cubes produits (pour revenir à des cubes déjà vus). La production de nouveaux cubes se fait en appliquant une ou des opérations OLAP sur un cube existant. Le déplacement parmi les cubes existants, permettant de revenir à des cubes déjà produits, se fait en exploitant les liens existants entre les cubes. Ce lien indique à partir de quel cube existant un cube a été produit.

3 Modèle et langage

3.1 Le modèle

Dans ce travail préliminaire, pour simplifier notre présentation, nous considérons que :

– les cubes manipulés sont représentés sous la forme de schémas en étoile ;
– les opérations effectuées sur les cubes sont les opérations de l'algèbre relationnelle étendue avec l'agrégation (Abiteboul et al., 1995). Dans ce qui suit, ce langage est noté $\mathcal{L}$.

Nous définissons une navigation comme un tuple constitué d'un arbre, qui représente les différents cubes générés lors de l'analyse, et d'un nœud particulier, appelé nœud courant, représentant le cube actuellement vu.

Nous donnons maintenant les définitions formelles.

```
union(A,A')

 A_temp = constructeur(racine(A))
 pour tous les fils $n$ de $A$
   si n n'existe pas parmi les fils de $A'$
     A_temp = ajout_arbre(A_temp,racine(A_temp),sous_arbre(A,n))

 pour tous les fils $n'$ de $A'$
   si n' n'existe pas parmi les fils de A
     A_temp = ajout_arbre(A_temp,racine(A_temp),sous_arbre(A',n'))
   sinon
     A_temp = ajout_arbre(A_temp,racine(A_temp),
                      union(sous_arbre(A,n'),sous_arbre(A',n')))
 retourne A_temp
```

FIG. 2 – *Algorithme d'union de deux arbres A et A'*

Nœud Dans ce qui suit, un nœud est un tuple $\langle C, op \rangle$ où C est un cube et op un opérateur de $\mathcal{L}$. Nous appelons *Nœud* l'ensemble des nœuds.

Arbre Nous considérons le type abstrait de données **Arbre** représentant des arbres ordonnés, où les nœuds sont des éléments de *Nœud*.

Dans la suite, nous utilisons les opérations suivantes sur les arbres :

nom	signature		
constructeur	Nœud	$\rightarrow$	Arbre
racine	Arbre	$\rightarrow$	Nœud
père	Arbre$\times$Nœud	$\rightarrow$	Nœud
ième_fils	Arbre$\times$Nœud$\times$Entier	$\rightarrow$	Nœud
feuille	Arbre$\times$Nœud	$\rightarrow$	Booléen
sous_arbre	Arbre$\times$Nœud	$\rightarrow$	Arbre
ajout	Arbre$\times$Nœud$\times$Nœud	$\rightarrow$	Arbre
ajout_arbre	Arbre$\times$Nœud$\times$Arbre	$\rightarrow$	Arbre
union	Arbre$\times$Arbre	$\rightarrow$	Arbre

Les opérations constructeur, racine, père, ième_fils, feuille et sous_arbre ont leur sens habituel. L'opération $ajout(A, n, n')$ ajoute un fils n' au nœud n de l'arbre A. L'opération $ajout\_arbre(A, n, A')$ ajoute l'arbre A' au nœud n de l'arbre A. L'opération $union(A, A')$ est définie, si $racine(A) = racine(A')$, par l'algorithme donné figure 2.

Par la suite, pour un nœud n et un arbre A, nous noterons $n \in A$ si n est un nœud de A.

Navigation Une navigation N est un tuple $\langle A, n \rangle$, où A est un Arbre et $n \in A$ est appelé *nœud courant* de la navigation. Les figures 3 à 12 présentent des exemples de navigations (le nœud courant est représenté noirci).

3.2 Les opérateurs

Dans les définitions qui suivent, étant donné un nœud $n = \langle C, op \rangle$, nous notons $C(n) = C$ et $op(n) = op$.

Navigation initiale Une navigation initiale est composée d'un arbre constitué d'un seul nœud racine $n_0 = \langle C_0, \top \rangle$, C_0 étant le cube sur lequel va porter l'analyse. Notons que ce cube n'est pas le résultat d'une analyse, ce qui est représenté par le symbole $\top$.

Apply Etant donné une navigation $N = \langle A, n \rangle$, $apply_{op}(N)$ permet d'appliquer une opération de $\mathcal{L}$ sur le cube du nœud courant. Formellement, si $N = \langle A, n \rangle$

$apply_{op}(N) =$

1. $\langle A, n' \rangle$ si $(\exists i)$ tel que $n' = ième\_fils(A, n, i)$ et $op(n') = op$
2. $\langle ajout(A, n, n'), n' \rangle$ sinon, où $n' = \langle op(C), op \rangle$

Notons que, si, à partir du nœud courant, une opération déjà effectuée est rejouée, nous ne rajoutons pas de nouveau nœud fils au nœud courant, nous modifions seulement le nœud courant.

Exemples de apply Nous considérons que notre première navigation est N_0 dont le nœud courant est $n_0 = \langle C_0, \top \rangle$. Nous décidons d'effectuer une sélection sur les Pays (France ou Espagne) sur la navigation N_0 :

$$apply_{\sigma_{pays=France \vee pays=Espagne}}(N_0) = N_1.$$

Nous obtenons la navigation N_1 ayant pour nœud courant n_1 (Figure 3).

Puis, nous effectuons une sélection sur les années (2005) sur la navigation N_1 :

$$apply_{\sigma_{année=2005}}(N_1) = N_2.$$

Nous obtenons la navigation N_2 ayant pour nœud courant n_2 (Figure 4).

Enfin, nous effectuons une aggrégation (somme) sur la navigation N_2 :

$$apply_{\pi_{pays,année,all;sum(quantité)}}(N_2) = N_3$$

Nous obtenons la navigation N_3 ayant pour nœud courant n_3 (Figure 5).

Back Etant donné une navigation $N = \langle A, n \rangle$, $back(N)$ permet de revenir en arrière. Formellement :

$back(N)$ =

1. $\langle A, n \rangle$ si $n = racine(A)$
2. $\langle A, pere(A, n) \rangle$ sinon

Exemple de back Nous considérons la navigation N_3, dont le nœud courant est n_3. Nous décidons d'effectuer l'opération $back$ sur cette navigation : $back(N_3) = N_4$. Nous obtenons la navigation N_4 ayant pour nœud courant n_2 (Figure 6).

De la même manière, nous souhaitons "remonter" d'un nœud et arriver en n_1 : $back(N_4) = N_5$. Nous obtenons la navigation N_5 ayant pour nœud courant n_1 (Figure 7).

Replace Soit une navigation $N = \langle A, n\rangle$, et une substitution $\Theta = \{op/op'\}$, $replace_\Theta(N)$ permet d'ajouter à A une nouvelle branche, où toutes les occurrences de l'opération op dans N sont remplacées par l'opération op'.

$replace_\Theta(N)$ =

1. N si $n = racine(A)$.
2. $apply_{\Theta(op(n))}(replace_\Theta(back(N))$ sinon

Exemple de replace Nous considérons la navigation N_5, dont le nœud courant est n_1. Nous décidons d'effectuer un replace de l'opération $\sigma_{année=2005}$ par l'opération $\sigma_{année=2004}$ sur la navigation N_5. Etant donnée la substitution $\Theta = \{\sigma_{année=2005}/\sigma_{année=2004}\}$, nous avons $replace_\Theta(N_5) = N_6$. Nous obtenons la navigation N_6 ayant pour nœud courant n_5 (Figure 8).

Forward Etant donné une navigation $N = \langle A, n\rangle$, $forward_i(N)$ permet de consulter le $i^{ème}$ fils du nœud courant. Formellement :

$forward_i(N) = \langle A, ième\_fils(A, n, i)\rangle$ si le ième fils existe, N sinon.

Exemple de forward Nous considérons la navigation N_6, dont le nœud courant est n_5. Suite à deux opérations $back$ successives :

$$back(N_6) = N_7, back(N_7) = N_8$$

nous obtenons la navigation N_8 ayant pour nœud courant n_1 (Figure 9).

Nous décidons d'effectuer l'opération $forward$ sur la navigation N_8 :

$$forward_1(N_8) = N_9$$

Nous obtenons la navigation N_9 ayant pour nœud courant n_2 (Figure 10).

Replay Etant donné deux navigations $N = \langle A, n\rangle$, et $N' = \langle A', n'\rangle$, $replay(N, N')$ permet de rejouer sur le nœud courant de N une navigation à partir du nœud courant de N'.

1. $replay(N, N') = N$ si $feuille(A', n') = true$,
2. $replay(N, N') = \bigcup_i replay(apply_{op_i}(N), forward_i(N'))$ sinon,

où (n_i, op_i) = $ième\_fils(A', n', i)$ et $\bigcup$ désigne l'union de deux navigations de même racine, définie comme suit : étant données deux navigations $N = \langle A, n\rangle$ et $N' = \langle A', n'\rangle$, $N \bigcup N' = \langle A'', n''\rangle$ avec $A'' = union(A, A')$ et $n'' = racine(A)$.

Exemple de replay Nous considérons la navigation N_9, dont le nœud courant est n_2. L'analyste effectue différentes opérations, à savoir :

$$back(N_9) = N_{10}, back(N_{10}) = N_{11}, apply_{\sigma_{pays=Japon}}(N_{11}) = N_{12}$$

Nous obtenons la navigation N_{12} ayant pour nœud courant n_6 (Figure 11).

Nous décidons d'effectuer l'opération replay sur la navigation N_{12} :

$$replay(N_{12}, N_8) = N_{13}$$

où N_8 est la navigation représentée Figure 9. Nous obtenons la navigation N_{13} ayant pour nœud courant n_{10}, représentée sur la figure 12.

4 Conclusion

4.1 Contribution

Dans ce travail, nous proposons de donner un statut précis à la navigation d'un analyste OLAP au sein d'un cube, afin notamment :

- d'exprimer le retour sur des résultats déjà obtenus ;
- de faciliter l'expression de traitements typiques sophistiqués, comme rejouer une analyse ou substituer, dans une analyse, une opération par une autre.

Ainsi, nous définissons une navigation comme un arbre de cubes générés via des opérations OLAP, dans lequel l'analyste peut se déplacer grâce à un langage de définition de navigations.

4.2 Perspectives

Ce travail servira de base aux extensions suivantes :

1. l'optimisation de requêtes dans le cadre d'une navigation : outre l'étude des propriétés des opérateurs sur les navigations, nous étudierons comment la connaissance, lors de chaque nouvelle définition de requête, de l'ensemble des réponses aux requêtes précédentes et de leurs liens, permet de trouver les données sur lesquelles évaluer la nouvelle requête (sans avoir à tester l'inclusion de requêtes) et de la même manière, dans quelle mesure les nœuds de l'arbre peuvent être réutilisés pour visualiser des résultats intermédiaires ;
2. la prise en compte du chemin parcouru dans l'arbre. Cela demande de changer la définition de navigation : une navigation N est constitué d'un arbre A et d'une pile p de nœuds, le nœud courant étant le sommet de la pile.

   Par exemple, l'opération *undo* permettant de défaire la dernière opération, pourrait être définie comme suit : étant donné une navigation $N = \langle A, n \rangle$, $undo(N)$ permet de revenir sur le nœud à partir duquel la dernière opération a été lancée. C'est-à-dire $undo(N) = \langle A, pop(p) \rangle$, où pop est l'opération de dépilement.
3. l'implantation sur machine en trouvant une méthode de stockage des navigations ainsi qu'une manière de les partager entre les différents utilisateurs ;
4. un langage de manipulation des navigations ainsi que la mise en place d'une base de navigations.

Références

Abiteboul, S., R. Hull, et V. Vianu (1995). *Foundations of Databases*. Addison-Wesley.

Agrawal, R., A. Gupta, et S. Sarawagi (1997). Modeling multidimensional databases. In W. A. Gray et P.-Å. Larson (Eds.), *Proceedings of the Thirteenth International Conference on Data Engineering, April 7-11, 1997 Birmingham U.K*, pp. 232–243. IEEE Computer Society.

Cabibbo, L. et R. Torlone (1997). Querying multidimensional databases. In S. Cluet et R. Hull (Eds.), *Database Programming Languages, 6th International Workshop, DBPL-6, Estes Park, Colorado, USA, August 18-20, 1997, Proceedings*, Volume 1369 of *Lecture Notes in Computer Science*, pp. 319–335. Springer.

Dittrich, J.-P., D. Kossmann, et A. Kreutz (2005). Bridging the gap between OLAP and SQL. In *VLDB*, pp. 1031–1042.

Franconi, E. et A. Kamble (2004). The GMD data model and algebra for multidimensional information. In *CAiSE*, pp. 446–462.

Golfarelli, M. et S. Rizzi (1998). Methodological framework for data warehouse design. In *DOLAP*, pp. 3–9.

Gyssens, M. et L. V. S. Lakshmanan (1997). A foundation for multi-dimensional databases. In M. Jarke, M. J. Carey, K. R. Dittrich, F. H. Lochovsky, P. Loucopoulos, et M. A. Jeusfeld (Eds.), *VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece*, pp. 106–115. Morgan Kaufmann.

Microsoft Corporation (1998). OLEDB for OLAP. Available at http ://www.microsoft.com/data/oledb/olap.

Mukhopadhyay, P. et Y. Papakonstantinou (2002). Mixing querying and navigation in mix. In *ICDE*, pp. 245–.

Pedersen, T. B., C. S. Jensen, et C. E. Dyreson (2001). A foundation for capturing and querying complex multidimensional data. *Inf. Syst. 26*(5), 383–423.

Vassiliadis, P. et S. Skiadopoulos (2000). Modelling and optimisation issues for multidimensional databases. In *CAiSE*, pp. 482–497.

Annexe

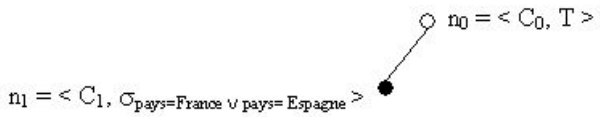

FIG. 3 – *N_1 : Sélection sur les pays : France et Espagne*

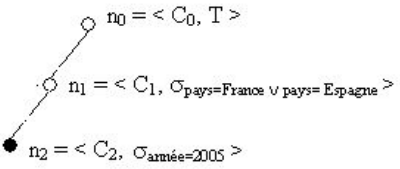

FIG. 4 – *N_2 : Sélection sur les années :* 2005

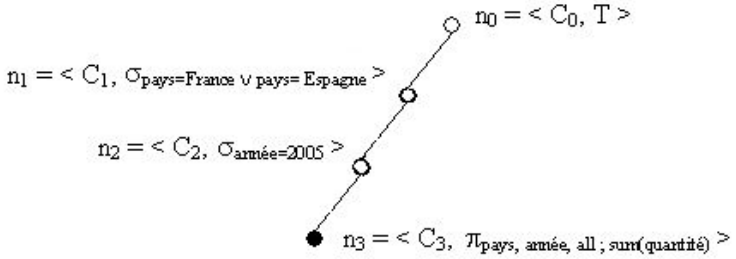

FIG. 5 – *N_3 : Aggrégation : Somme*

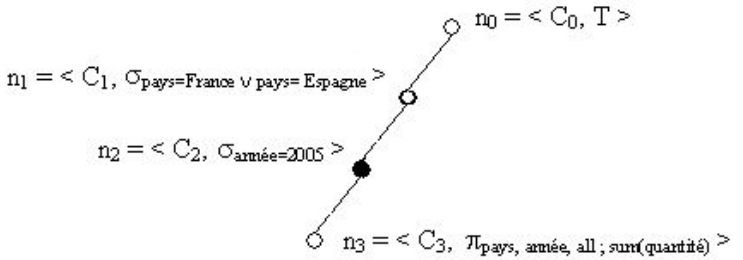

FIG. 6 – N_4 *: Exemple 1 d'opération back*

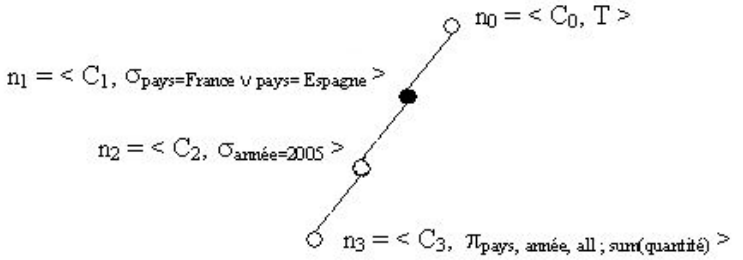

FIG. 7 – N_5 *: Exemple 2 d'opération back*

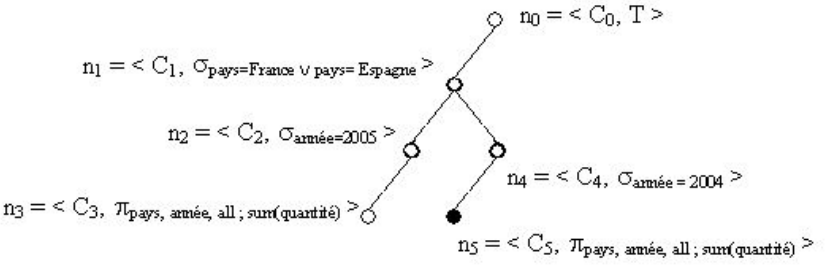

FIG. 8 – N_6 *: Exemple d'opération replace*

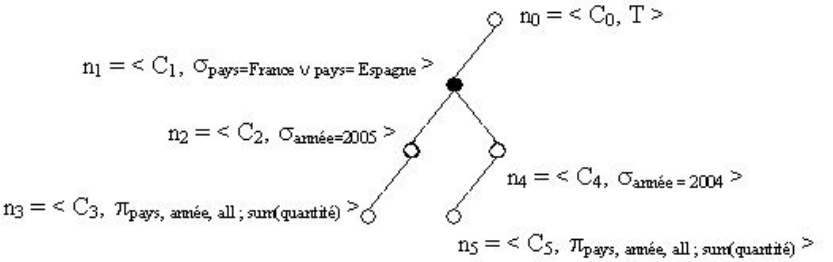

FIG. 9 – *Navigation* N_8

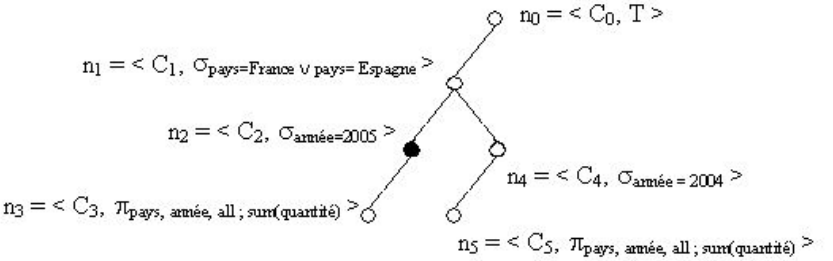

FIG. 10 – N_9 *: Exemple d'opération* $forward$

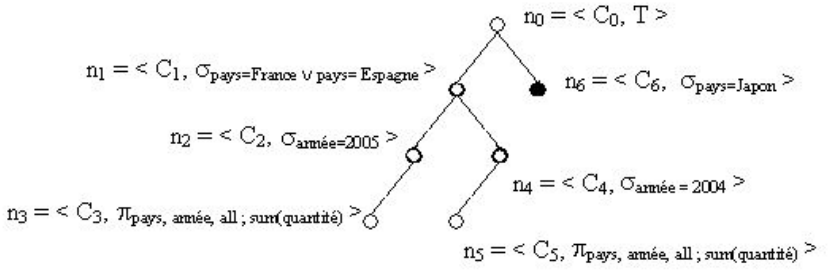

FIG. 11 – *Navigation* N_{12}

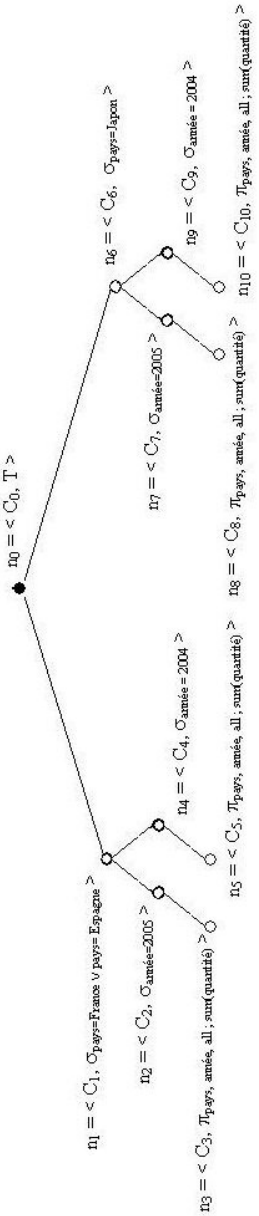

FIG. 12 – N_{13} *: Exemple d'opération* $replay$

Summary

In the context of OLAP analysis, the concept of navigation has never been formaly defined. We show why that this lack should be filled. We propose a formal definition of navigation, and a first step towards a navigation definition language.