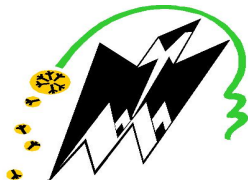
ent Supérieur et de la Recherche Scientifique

Mouloud Mammeri, Tizi-Ouzou

Faculté de Génie Electrique et d'Informatique

Département Informatique



Projet De fin d'études

En vue de l'obtention du diplôme Master2

En Informatique

Option :Conduite de projets informatique

THEME

Personnalisation dans les entrepôts de données

Encadré par :

Mr Ouamrane

Présenté par :

DJEBBARI Sonia

Promotion 2011-2012

Listes des figures

Listedes tableaux

Contexte général

Au cours de ces dernières années, on assiste à une prolifération des ressources hétérogènes, conduisant à une surcharge cognitive. Dans ce contexte, l'accès à l'information pertinente représentant aujourd'hui un enjeu de taille, est confronté au volume plus qu'important, ainsi que l'inintelligibilité de l'information retournée. Il est évident que ces résultats massifs délivrés suite à des requêtes utilisateurs augmentent de plus en plus les temps de réponse, et s'emparent de la patience de ces derniers.

Cette surcroissance à la quelle nous devons faire face aujourd'hui, bien qu'elle soit conséquence directe des récents progrès technologiques, confronte l'utilisateur à des interfaces peu commode (encombrantes), ou il est difficile de distinguer le pertinent du bruit. Face à une telle situation dite paradoxale, il devient nécessaire d'adapter les besoins spécifiques de l'utilisateur à sa requête, c'est pourquoi la personnalisation de l'information fut abordée.

Problématique :

Lôaccès à lôinformation pertinente selon les besoins de lôutilisateur est devenu plus quôune nécessité. Que ce soit dans le contexte des systèmes dôinformation dôentreprise, du commerce électronique, de lôaccès au savoir et aux connaissances ou même des loisirs, lôadéquation de lôinformation délivrée aux usages et préférences des clients constituent des facteurs clés du succès ou du rejet de ces systèmes.

Objectif :

Notre objectif consiste a servir au mieux l'utilisateur en fonction de ses attentes, et pour cela on construira un système dans lequel on pourra recueillir des informations sur celui-ci, pour mieux cerner ses centres d'intérêts, et s'en servir dans le but d'aboutir à des réponses adaptées, autrement dit « ***personnalisées*** ».

ersonnalisation de lôinformation dans le contexte de lôinterrogation des entrepôts de données par des requêtes permettent lôanalyse interactive dôun gros volume de données multidimensionnelles à lôaide dôopérateurs spécifiques de consolidation et agrégation pour résumer lôinformation. Ces données sont organisées sous forme de cubes de données et proviennent de lôentrepôt de données.

Problématique

Les systèmes dôanalyse des entrepôts de données actuels ont peu de connaissances sur lôutilisateur. Ils ne tiennent pas compte de leurs caractéristiques spécifiques pour la restitution des données, à savoir ses objectifs et ses centres dôintérêts. Ceci oblige lôanalyste à naviguer au sein des données par un enchainement dôopérations et une succession de résultats intermédiaires pour obtenir les données pertinentes à sa prise de décision (adaptées à ses besoins spécifiques dôanalyse). Lôanalyse peut alors sôavérer une tâche fastidieuse qui dégrade les performances du processus dôanalyse décisionnelle. Cette dégradation est aggravée par un coût dôexécution important des requêtes dans un environnement avec un grand nombre de dimensions.

Objectif :

Notre objectif est de personnaliser les requêtes décisionnelles interrogeant des bases de données multidimensionnelles en restituant les données en fonction des préférences utilisateur et de ses contraintes. Pour ce faire, chaque utilisateur (décideur) de lôentrepôt de données est invité à donner ses préférences (son profil), ainsi quôun ordre sur ces préférences pour que sa requête puisse être ***personnalisée*** en fonction de ce quôil décide être plus pertinent pour son cas.

Ce document se structure de la manière suivante.

Le chapitre 1 : Personnalisation de lôinformation :
Ce chapitre présente le contexte dôétude, à savoir, la notion de personnalisation et ses domaines ainsi que les différentes méthodes qui furent abordées.

Le chapitre 2 : Profil utilisateur : Le plus souvent la personnalisation est plus accomplie en se munissant dôun profil utilisateur, côest pourquoi dans ce chapitre nous présenterons la notion de profil utilisateur ainsi que se modélisation.

Le chapitre 3 : Les entrepôts de données :Nous parlerons dans ce chapitre des entrepôts de données en général, ce quôils représentent, comment se construisent-t-ils, a quoi ils visent.

Le chapitre 4 : La personnalisation dans les entrepôts de données
Dans ce chapitre, nous avons présenté dôabord les concepts généraux de la personnalisation, puis nous avons dressé un panorama des travaux réalisés sur la personnalisation dans les entrepôts de données en général.
Le chapitre 5 : Réalisation :

Personnalisation de l'information

Aujourdôhui, devenu un concept très répondu et qui a porté ses fruits face a la surabondance informationnelle qui ne cesse de prendre de lôampleur suite aux évolutions hallucinantes rencontrées en informatique, qui génèrent des volumes de données de plus en plus important, ou il est quasi-impossible de se retrouver, et ou lôambiguïté règne.

1.2 Contexte général

La personnalisation peut se définit par une collecte dôinformations propres à lôutilisateur, regroupant ses préférences et centres dôintérêts qui formeront son profil, représentés le plus constamment par des couples (attribut, valeur) ordonnés selon ses désirs, visant à fournir des réponses rapides, réduites et adaptées à lôutilisateur en fonction du profil, ainsi la réponse consistera un sous ensemble du résultat de la requête initiale émise.

La personnalisation fut lôobjet de nombreux travaux dans des domaines tels que la recherche dôinformations (RI), les bases de données (BD) et les interfaces homme-machine (IHM). Cependant, elle constitue un axe de recherche récent dans le domaine des entrepôts de données.

1.1 .1. Domaine RI :

La Recherche dôInformation est une branche en informatique qui s'intéresse à l'acquisition, l'organisation, le stockage et la recherche des informations, dont l'objectif est de fournir des techniques et des outils pour sélectionner les informations pertinentes contenues dans une collection de documents en réponse aux besoins informationnels d'un utilisateur. [1.1]

Un système de recherche d'information personnalisé est un système qui intègre l'utilisateur, en tant que structure informationnelle, tout au long de la chaîne d'accès à l'information et il ne se limite pas seulement à modéliser les caractéristiques des utilisateurs en profils mais il doit être capable de déduire à partir de ces profils, l'intention de l'utilisateur lorsqu'il effectue sa recherche et de détecter l'évolution des profils de manière dynamique. [1.2]

Lôévaluation dôune requête utilisateur se fait généralement de façon incrémentale et à chaque itération, le système tient compte des informations collectées à partir des interactions précédentes avec lôutilisateur et profite de lôexpérience des autres utilisateurs.
Les données collectées à partir dôun utilisateur ne sont pas utilisées directement par le système, mais sont analysées pour déterminer la raison de son comportement. Cette analyse permet aux systèmes dôextraire des mots clé, les caractéristiques des éléments recherchés ou

ntenu des documents, et de déterminer lôintérêt dôun [1.3] [1.4] [1.5]

Le profil est utilisé pour remplacer la requête de lôutilisateur et rajouter les mots clés et il est présenté à l'aide d'un vecteur de mots clés à N dimensions avec éventuellement un poids associé à chaque mot et les dimensions sont définies par les termes les plus significatifs pour les documents recherchés, le système calcule la distance entre le profil et les mots clés significatifs extraits des documents en utilisant une technique basée sur la distance entre vecteurs à N dimensions. Seuls les documents dont la distance dépasse un certain seuil sont inclus dans le résultat. [1.6] [1.7]

Bien que la notion de profil utilisateur existe, les systèmes de personnalisation sont incapables de fournir à lôutilisateur les moyens dôexprimer ses préférences. Les processus de la découverte des informations pertinentes sont souvent longs et décourageants.

1.1 .2.Domaine Bases de données :

Dans le domaine des BD, la requête de lôutilisateur contient lôensemble des critères jugé utiles à la sélection des informations pertinentes par ailleurs le profile qui est intégré directement aux requêtes contient des données qui expriment ses habitudes et il est intégré directement aux requêtes. Bien que, certains systèmes présentent une interface permettant à lôutilisateur dôintervenir pendant le processus dôévaluation de la requête, la personnalisation des données a été envisagée sous lôangle dôextension du langage de requêtes SQL par un ensemble de clauses ou de prédicats sensés traduire les préférences de lôutilisateur, sans pour autant interagir avec ce dernier durant lôévaluation. Les critères approximatifs et les critères de préférences introduits dans certains langages de requêtes BD ouvrent la voie à une adaptation des résultats aux désirs des lôutilisateur. [1.8]

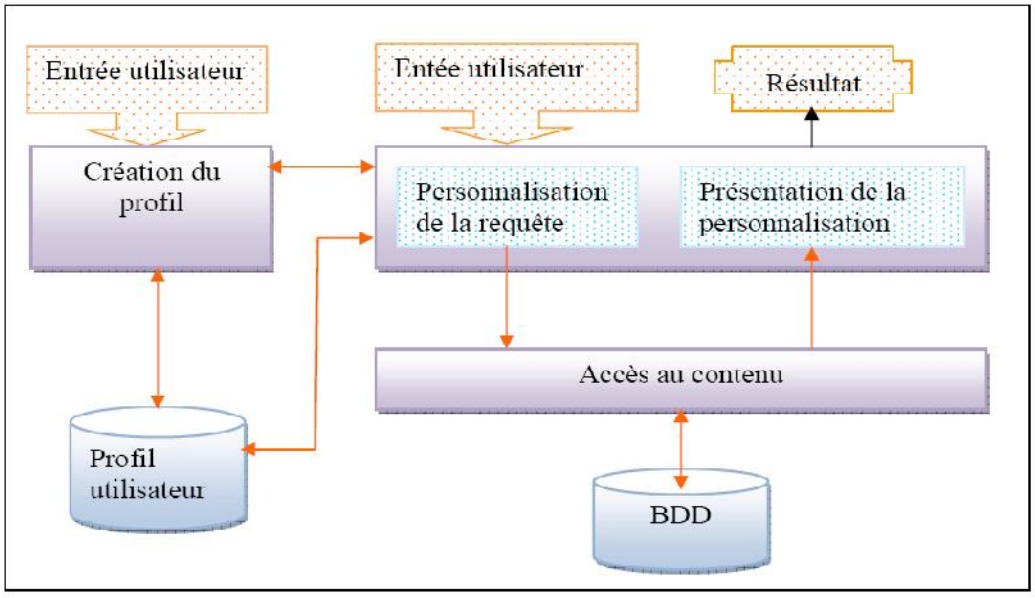


Fig.1.1 : Architecture dôun système de bases de données personnalisé.

1.1 .3.Domaine IHM :

Dans le domaine des IHM la personnalisation sôeffectue au niveau de la navigation. Il sôagit de représenter les habitudes dôanalyse de lôutilisateur, sous forme de coefficients de préférence, pour faciliter sa navigation car en IHM la notion de requête nôexiste pas sous forme langagière. Les systèmes utilisent des connaissances sur lôutilisateur (âge, niveau dôexpertise, handicaps etc.) ou sur la technologie quôil utilise (type du media, logiciels etc.), autrement dit le profil, afin de déterminer le type de dialogue que le système doit avoir avec lui, les métaphores graphiques les plus appropriées ainsi que les modalités de livraison des résultats qu'il attend du système d'information dont le but est de guider les recherches de lôutilisateur et de faciliter lôexpression de ses besoins. [1.9]

Un exemple simple d'un tel profil est celui utilisé par les fournisseurs de services Web, dans lesquels le profil présente un ensemble de données personnelles et de catégorie dôintérêts qui constituent sa page dôaccueil ; comme il peut également contenir les statistiques dôactions avec les systèmes (nombre de clicks, temps de lecture, etc.) (Bradley *et al.*, 2000). Une autre alternative consiste à stocker dans le profil utilisateur des fonctions dôutilités sur un domaine dôintérêt, qui permettent dôexprimer lôimportance relative des sujets de ce domaine, les uns par rapport aux autres. [1.9]

Les profils sont ensuite utilisés dans le processus de traitement du système de différentes façons : remplacer la requête, permettre de lôenrichir (ajout de critères de sélection, de nouveaux mots-clés) ou être utilisé pour adapter les résultats, dans leur contenu ou dans leur forme de présentation. [1.10]

1.1 .4.Domaine entrepôts de données :

La personnalisation dans le cadre des entrepôts de données présente un réel intérêt dans un contexte ou les analyses permettant la prise de décision sont dites centré utilisateur, car il sôest avéré que ce derniers peut avoir des besoins de contexte dôanalyse spécifique, répondant à des besoins spécifiques voir individuelles. Dôoù lôémergence de travaux consacrées à la personnalisation dôanalyses qui placerai lôutilisateur au cî ur du processus décisionnel. Ces travaux exigent des efforts cognitifs de la part des utilisateurs qui souvent se retrouvent dans la nécessité dôexprimer de manière explicite leurs préférences.

En effet, la volumétrie des données connues pour être importante dans les entrepôts de données, et le rôle central attribué à lôutilisateur dans le processus décisionnel, sont deux éléments qui permettent pleinement de justifier le recours à la personnalisation. [1.11]

...es de personnalisation :

Bien que lôobjectif soit le même, c'est-à-dire adapter la réponse a la requête en faisant intervenir les préférences utilisateur, les méthodes de personnalisation différent dans leur manière dôaborder ce problème.

1.2.1.Méthode de Koutrika :

Dans la méthode proposée par [Koutrika and Ioannidis,2004] les préférences du profil utilisateur portent sur le contenu et définissent une relation de préférence (ordre total) entre des requêtes. Les préférences sont représentées par le langage basé sur des fonctions de score. Deux requêtes sont comparées sans accéder aux données. Il sôagit dôenrichir la requête avec le profil utilisateur afin que lôévaluation de requête tienne compte des préférences.

Plus précisément, étant donnés une requête Q et un profil utilisateur U où sont stockées les préférences atomiques, soit P lôensemble de préférences de sélections extraites à partir de U et liées à Q . Une requête personnalisée, notée Q_x ,est une combinaison de Q et dôun sous-ensemble P_x de P telle que : Q_x Q P_x

On note que dans cette méthode la réponse peut être vide puisque la personnalisation est faite sans accès aux données.

[Koutrika and Ioannidis, 2005a] propose de prendre en compte des contraintes telles que le coût dôexécution et/ou la taille de la réponse. Il sôagit dôun problème dôoptimisation avec contraintes visant à maximiser lôintérêt de lôutilisateur pour les résultats dôune requête sous contraintes et qui minimise le coût dôexécution sur lôintérêt et/ou la taille:

$$doi(Q_u) = MAX\{doi(Q_x) | Q_x = Q P_x , P_x P, Q_x \text{ satisfait les contraintes}\}$$
$$cost(Q_u) = Min\{costi(Q_x) | Q_x = Q P_x , P_x P, Q_x \text{ satisfait les contraintes}\}$$

avec Q_u une requête personnalisée qui satisfait des contraintes.

1.2.2. d'Agrawal :

Dans la méthode [Agrawal and Wimmers, 2000], les préférences du profil utilisateur portent sur le contenu et définissent une relation de préférence entre les tuples dôune relation de base de données et sont représentées par le modèle de fonction de score, pas de contraints.

Deux tuples sont comparés après le calcul de la réponse à la requête. On traite des requêtes dôoptimisation, lôobjectif est de déterminer les meilleurs tuples de la réponse à une requête utilisateur en se basant sur le profil:

$$\{<t,score> (<t',score'>)(score' > score)\}$$

omicki,2003] qui est la même que celle proposée par [Kießling,2002], pas de contraintes, les préférences du profil utilisateur portent sur le contenu et définissent une relation binaire entre les tuples de la réponse à la requête. Le problème consiste à trouver les tuples les plus intéressants du résultat par rapport aux préférences utilisateur. Ces préférences sont représentées par une relation binaire [Chomicki,2003] ou par un ordre partiel strict [Kießling,2002] et sont intégrées dans le langage relationnel à lôaide dôun modèle appelé *Best Match Only* dans [Kießling,2002] et qui est identique à lôopérateur de préférence, appelé *winnow* dans [Chomicki,2003],défini ci-dessous :

***Operateur winnow* :** *Si R est un schéma de relation et C une formule de préférence définissant une relation de préférence* $>_C$ *sur R, alors lôopérateur winnow, noté* $w_C(R)$*,est défini pour toute instance r de R :* $w_c(r) = \{t \in r \mid \neg \exists t \in r, t >_c t\}$

On note quel ôopérateur winnow retourne les tuples dôune instance donnée dôune relation qui ne sont dominés par aucun autre tuple de cette instance.

1.2.4.Méthode de Boutilier :

Dans le méthode proposée par [Boutilier etal.,2004a] les préférences du profil utilisateur portent sur le contenu, et définissent une relation dôordre entre les tuples de la réponse à la requête. Deux tuples sont comparés après lôaccès aux données. Le problème consiste à trouver les tuples les plus intéressants du résultat par rapport aux préférences utilisateur et qui satisfont une condition donnée. Pour résoudre ce même problème, [Boutilier et al.,2004b] prend compte des contraintes dites *hard constraints* et qui sont des conditions qui doivent être satisfaites.

Dans le contexte des *CP-nets*, pour déterminer ses préférences, un utilisateur doit spécifier, pour chaque variable X, lôensemble des variables parents *Pa(X)* concernant ses préférences sur les valeurs de lôattribut X. Ensuite, pour chaque instance possible de *Pa(X)*, lôutilisateur doit donner un ordre de préférence pour les valeurs de X et toutes choses égales par ailleurs. Cette information est utilisée pour construire le graphe des *CP-net*.

1.2.5.Méthode de Das

Proposée par [Das et al., 2006], dans cette méthode, les préférences du profil utilisateur portent sur la présentation, et définissent une relation dôordre entre les tuples de la réponse à la requête. Deux tuples sont comparés après lôaccès aux sources de données. Pas de contraintes. La présentation est exprimée par la sélection dôattributs pertinents et en affichant le détail des tuples qui leurs correspondent dans des tables et dans un ordre selon leurs importances du point de vue de lôutilisateur. [1.12]

1.3.1. Eléments du profil

- *Que personnaliser* ? Indique sur quoi porte la personnalisation (contenu, présentation)
- *Quel est lôespace de recherche* ? précise sur quels éléments sont définies les préférences utilisateur (tuples, attributs, etc).
- *Quelles sont les contraintes* ? décrit les contraintes prises en compte.
- *Selon quel langage de préférence* ? indique quel langage de préférence est utilisé par Une méthode donnée parmi les langages déjà décrits.

1.3.2. La requête

- *Quand personnaliser ?* spécifie si la personnalisation est réalisée avant ou après lôaccès aux sources de données.
- *Quel est le type de requête de personnalisation ?* spécifie le type de requête de personnalisation parmi celles précédemment présentées.

1.4. Conclusion

Nous pouvons à présent constater que plusieurs méthodes de personnalisation existent, et procèdent de manières parfois divergentes, malgré un objectif commun qui est lôadaptation des réponses à des requêtes utilisateurs selon leurs attentes, et ce en sôappuyant souvent sur un profil qui nous renseigne sur ce que lôutilisateur préfère, côest pourquoi dans le chapitre qui suit nous présenterons le profil utilisateur ainsi que toutes les notions quôil ya autour.

Profil utilisateur

La notion du profil utilisateur est apparue vers les années 80, avec comme objectif principal de créer des applications personnalisées capables de sôadapter aux besoins de lôutilisateur.

La personnalisation dôune application pour un utilisateur particulier nécessite la disposition dôinformations sur ce dernier, ce qui permettra dôévaluer la pertinence des objets disponibles et dôaider le système à faire des choix. [2.1]

2.1. Notions de profil:

2.1.1.Définition du profil

Un profil utilisateur est une collecte dôinformations regroupant lôensemble de connaissance nécessaire a une évaluation efficace et meilleure de la requête autrement dit a une production dôinformation pertinent adaptée aux besoins de utilisateur .
Le profile utilisateur peut être vu comme un modèle personnalisé dôaccès a lôinformation avec des valeurs associées contenant les préférences de lôutilisateur qui servent à décrire leurs centre dôintérêt selon leurs contexte.

Le profil utilisateur est composé de plusieurs dimensions et chacune dôelle est constituée dôun ensemble dôattributs organisés en entités. Les attributs peuvent être : simples ou composés .[2.2]

- Les attributs simples sont caractérisés par leur nom et le type de valeurs et leur structure.
- Quant aux attributs composés sont appelés des sous dimensions et contient un ensemble dôattributs simples qui sont liés dôune manière sémantique.

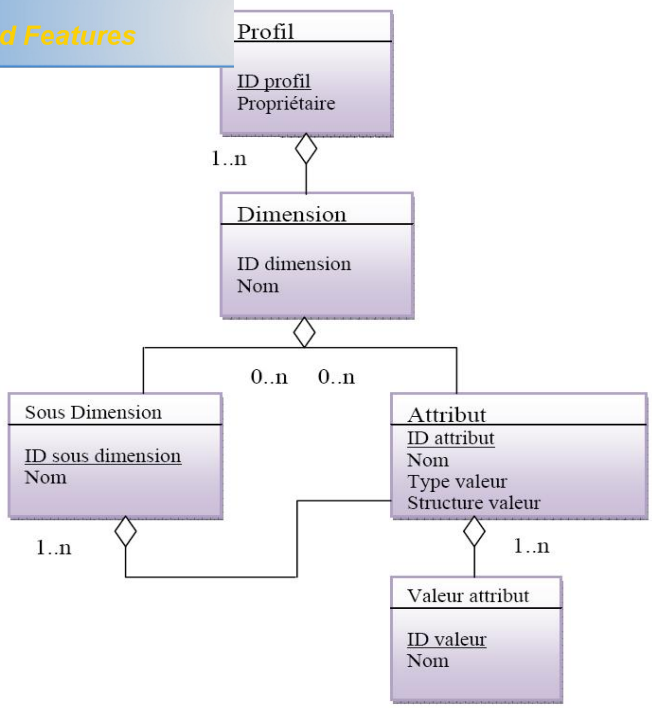


Fig.2.1 : Méta modèle de profil.

La notion de profil est souvent liée à celle de préférence et de contexte. En effet, les préférences de l'utilisateur font partie intégralement de son profil. En plus la description de l'utilisateur et ses préférences peuvent changer en fonction du contexte dans lequel il évolue. Cependant il y a une ambiguïté de la terminologie entre les trois concepts : profil, contexte et préférences ce qui rend difficile l'étude et la compréhension de la problématique liée à la personnalisation. [2.3]

Définition de contexte : Un contexte représente les données décrivant l'environnement d'interaction entre un utilisateur et un système.

Définition de préférence : Une préférence est une expression permettant de hiérarchiser l'importance des informations dans un profil ou un contexte.

2.1.2 Relations entre les trois concepts profil, le contexte et les préférences

Une représentation complète de l'utilisateur et du contexte, contient non seulement des éléments de description, mais également des préférences. En plus, la définition d'un profil utilisateur peut dépendre du contexte dans lequel il est exploité. Par conséquent, l'obtention d'un modèle complet décrivant l'utilisateur et/ou le contexte passe obligatoirement par l'identification des relations qui existent entre les trois composants profil, contexte et préférences. [2.4]

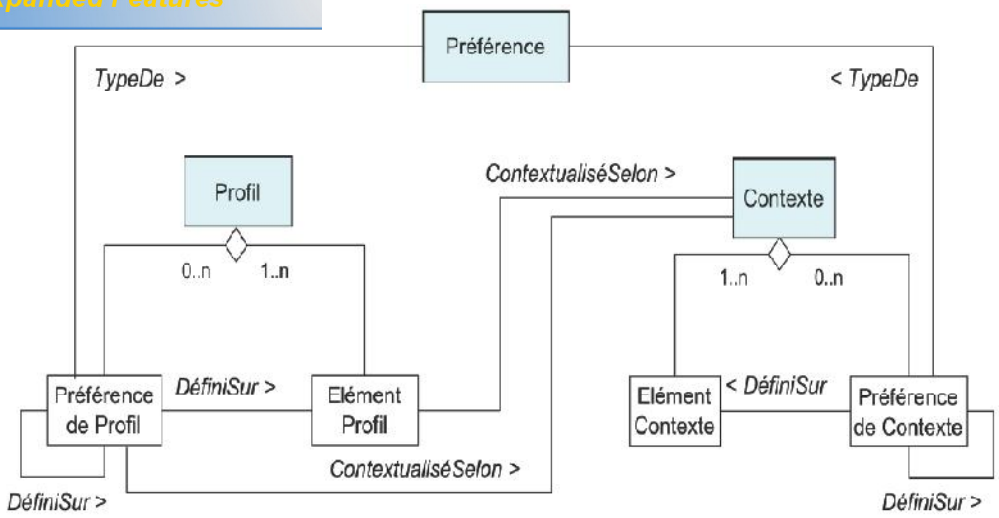


Fig.2.2 : Relations entre profil, le contexte et les préférences.

Selon le schéma on a :

Un profil utilisateur est composé dôun ensemble dôéléments de profil et de préférences. Les éléments du profil peuvent être des dimensions, des sous dimensions, des attributs et des valeurs.

Les préférences peuvent être définies sur lôensemble de ces éléments à condition de combiner des éléments du même type.

Le contexte est défini de la même manière que le profil et est composé dôun ensemble dôéléments de contexte et de préférences.

Les préférences du contexte permettent de définir un choix par défaut sur les caractéristiques du contexte. Par exemple, il est possible dôexprimer le fait que lôordinateur à partir duquel lôutilisateur se connecte est plus important que son emplacement physique pour décider sôil est dans un contexte de travail ou de loisir.

2.2. Création d'un profil utilisateurs :

La création dôun profil utilisateur se fait en deux étapes :

- choix de la structure du profil :
  La sélection et le recopie de la structure dôun profil existant est faite automatiquement sans prendre les valeurs éventuels de ses attributs. Le choix personnalisé est fait en glissant les attributs, les sous-dimensions ou les dimensions voulus du profil générique vers le nouveau profil. La recopie dôune sous-dimension

e ses attributs, de même que le choix dûne dimension
ble de ses sous-dimensions et attributs.

- attribution de valeurs aux attributs du profil:
  La deuxième étape de la création dûn profil revient à attribuer des valeurs aux attributs. Ceci peut être fait manuellement par lûutilisateur via lûinterface graphique. [2.5]

2.3. Le contenu du profil

Les données stockées dans les profils peuvent être classées selon quatre catégories:

Des exemples d'objets de l'espace de recherche

Lûidée principale de ces modèles de profils est de stocker des exemples dûobjets jugés pertinents ou comme inintéressants pour ensuite comparer le contenu des profils des différents utilisateurs entre eux. Sûil y a des similarités entre deux profils le système va recommander à lûutilisateur des objets que dûautres clients de profils semblables ont appréciés.
Comme dans certains cas les objets peuvent être volumineux (comme des livres, rapports etc.), dans ces cas il est possible de ne stocker que leurs identifiants. [2.6]

Des caractéristiques extraites des objets de l'espace de recherche

Lûidée de cette approche est de trouver une forme normale permettant la représentation des profils et des objets de lûespace de recherche. Ceci est fait le plus souvent en analysant les objets pour extraire les caractéristiques pertinentes pour lûutilisateur qui sont dans la plupart du temps des mots clés.

Dans certains cas les mots clés sont combinés avec dûautres informations portant sur lûimportance des mots pour lûutilisateur (fréquences dûapparition, poids, votes etc.).
Dans cette approche un profil est présenté comme un vecteur à N dimensions et à chaque dimensions est associer un poids qui correspondent au degré dûintérêt du lûutilisateur.
Le but visé dans ce cas est la comparaison directe entre les profils et les éléments recherchés qui simplifie la technique de personnalisation. Seuls les documents dont la similarité avec le profil dépasse un certain seuil sont inclus dans le résultat. [2.7] [2.8] [2.9]

Des attributs décrivant les préférences externes de l'utilisateur

Les informations des deux catégories précédentes sont directement liées aux objets de lûespace de recherche et à leur contenu, mais ne portent aucune information sur la qualité des informations ou sur lûutilisateur. [2.10] [2.11]

ions on retrouve les préférences de lôutilisateur liées aux informations et le type et la structure des éléments

Des règles et des formules de préférences

Dans certaines approches le profil de lôutilisateur est remplacé par des formules de préférence qui permettent de définir un ordre entre les éléments selon la pertinence de ces éléments pour lôutilisateur.

Les préférences des utilisateurs sont capturées par des règles actives qui tentent de découvrir des généralités en regroupant les utilisateurs selon les caractéristiques comme lôâge, le sexe, le nom du domaine, le type du browser, le système d'exploitation, les activités personnelles ou la situation familiale. [2.12] [2.13]

2.4. Caractéristiques du profil utilisateur

L'acquisition explicite du profil utilisateur

Les traits qui caractérisent le profil utilisateurs sont : ses connaissances du sujet, ses objectifs, ses préférences et son expérience. Peter Brusilovsky a cité deux autres caractéristiques qui sont : ses centres dôintérêts et ses traits individuels [2.14]

Centres dôintérêts: lôutilisateur peut les formuler à travers un ensemble de mots clés, pour décrire ses centres dôintérêts et sur la base de ces mots clés, le système recommandera des informations qui répondent au mieux à ses attentes.

Traits individuels: ce sont des caractéristiques stables et ils englobent les facteurs de personnalité. Ces traits ne sont pas tirés dôun simple entretien mais après des tests psychologiques.

L'acquisition implicite du profil utilisateur

Traditionnellement la méthode de retour de pertinence exige de lôutilisateur quôil donne explicitement son jugement à travers la spécification des mots clés ou par la réponse à des questions sur ses centres dôintérêts or souvent il y a un décalage entre lôintention de lôutilisateur et ce quôil désire réellement.

Cêest la raison pour laquelle certains systèmes misent sur un processus de découverte implicite des centres dôintérêts de lôutilisateur de manière interactive et incrémentale. Le fonctionnement de base de cette approche est réalisé par lôétablissement dôun dialogue entre le système et lôutilisateur ou mieux encore en observant son comportement à travers ses différentes interactions avec le système pour récolter discrètement l'information nécessaire sur lui.

2.5.1. Représentation du profil

Le profil de lôutilisateur nôa pas de structure explicite qui le représente. Il peut être constitué de divers paquets dôinformations qui traduisent une connaissance éparse sur lôutilisateur.

Il nôexiste pas un modèle spécifique dédié à la représentation du profil de lôutilisateur. Dans ce sens, la représentation des profils rejoint en grande partie la représentation de lôinformation dans le contexte de la recherche dôinformation.

2.5.2. Evolution du profil

Mise à jour des profils

Le gestionnaire de profils offre des fonctionnalités de mise à jour qui sont :
Lôinsertion, la suppression et la modification de la structure et des valeurs des profils.La seule règle à respecter est que le modèle de profil générique doit inclure lôunion des structures de toutes les instances de profils. Pour maintenir le profil générique le plus complet possible, chaque insertion dôune nouvelle dimension, sous dimension ou attribut dans une instance de profil se fait par le modèle générique.

Archivage des profils

Lôarchivage de profils permet de sauvegarder le contenu dôune instance de profil, date de création de lôarchive et lôensemble des valeurs du profil. Plusieurs archives peuvent être créées pour le même profil ce qui permet de suivre lôévolution des préférences dôun utilisateur ou dôanalyser son comportement. Les archives peuvent être visualisées et il est possible de remplacer le contenu actuel du profil par une de ses archives donc possibilité de restauration du profile. [2.15]

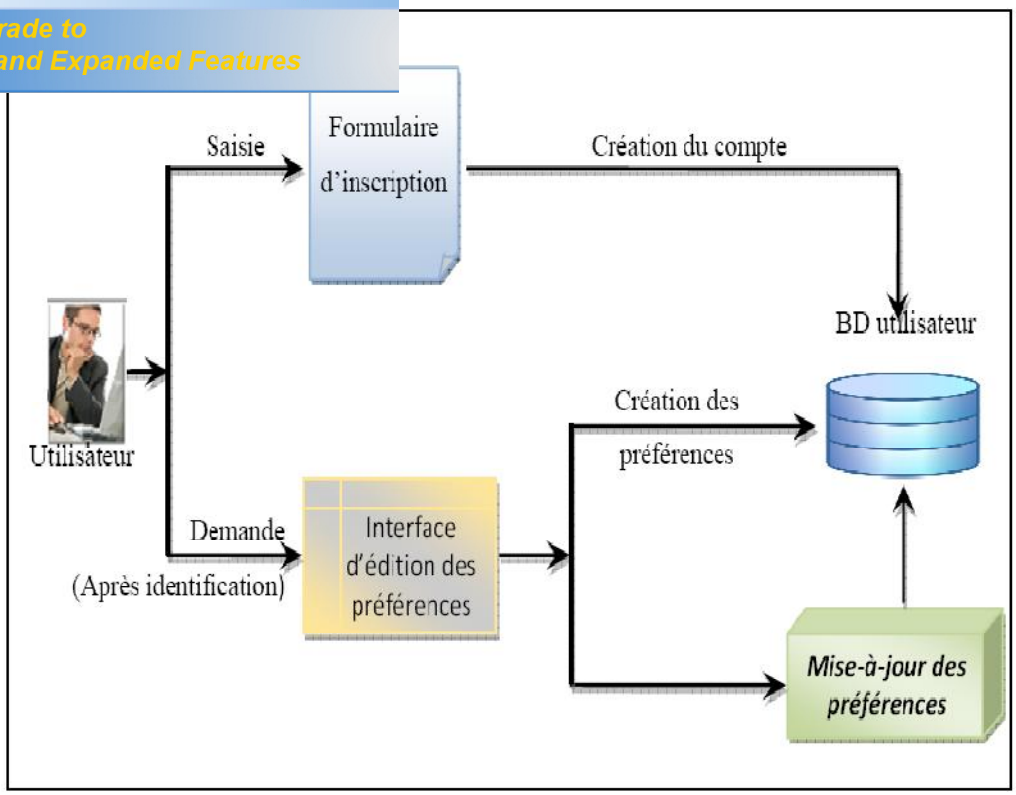


Figure n° 42 : Sous-système de gestion du profil

2.6. Sélection de l'information

Le profil d'un utilisateur est défini pour adapter le système de recherche d'information à et utilisateur. Cette phase consiste à intégrer le profil de l'utilisateur dans le processus de recherche d'information, les informations contenues dans le profil sont exploitées pour réécrire puis exécuter la requête, et enfin de visualiser les résultats de la requête.

2.7. Notion de préférence

Une préférence utilisateur est un ensemble de descriptions englobant: ce qu'un utilisateur envisage d'accomplir dans le système (les activités) et la manière de le faire (séquentielle, concurrente, conditionnelle), le type et l'ordre des résultats de ces activités (le contenu), et la manière dont il aimerait que l'information soit affichée.

De cette définition nous distinguons trois types de préférences :

ncernent les activités quôun utilisateur souhaite et peut

ncernent le contenu et le format préféré des résultats des fonctionnalités (par exemple, de la vidéo, du texte, ou des images).

Å **Les Préférences dôAffichage :** Concernent la manière dont lôutilisateur souhaite que lôinformation soit affichée.

2 .8. Conclusion

On peut constater à présent que la personnalisation fut abordée dans plusieurs domaines, selon différentes méthodes, en se référant souvent a un profil utilisateur dont il nôexiste pas un modèle prédéfini, et qui se construit selon les besoin du système, mais toujours vers un objectif commun, celui dôadapter la réponse aux besoins personnels de lôutilisateur, ce qui consiste a fournir des résultats différents pour une même requête émise par deux utilisateurs distinct (profils distincts).

Entrepôts de données

Pour faire face aux nouveaux enjeux dus aux effets de la mondialisation et la rude concurrence, lôentreprise doit collecter, traiter, analyser les informations de son environnement pour anticiper et rester concurrente. Lôentreprise manipule et stocke une multitude de données au sein de plusieurs applications. Ces données se trouvent dans différentes sources hétérogènes, distribuées et autonomes. Elles peuvent aussi être structurées, semi-structurées ou non structurées et stockées sous plusieurs formes : relationnelles, objets, texte, XML, HTML, fichiers, etc. Ces données sont utilisées quotidiennement et sont souvent inappropriées pour des besoins de prise de décision. Il devient alors fondamental de rassembler et dôhomogénéiser les données afin de permettre lôanalyse des indicateurs pertinents pour faciliter la prise de décision. Côest pourquoi les entrepôts de données furent proposés. Et lorsquôon parle dôentrepôts de données, on parle dôinformatique décisionnelle, autrement du BI (Business Intelligence).

3.1. Business Intelligence

Dans la conjoncture actuelle, les entreprises ont besoin de faire face à une concurrence rude et omniprésente, le besoin dôavoir une longueur dôavance pour mieux appréhender les attentes du marché et anticiper les évolutions économiques et autres. Côest un tel contexte quôa vu le jour le domaine du décisionnel destiné à permettre aux entreprises de se tenir alarmées des diverses évolutions économiques sur divers critères : commercial, concurrentiel, juridique ou encor environnementale

Ces entreprises doivent récupérer une masse très importante dôinformations pour mieux se protéger. La collecte et le traitement dôinformation en vue dôune prise de décision constituent la démarche du business intelligence.

Il appartient désormais à tout acteur économique de comprendre et dôanticiper les mutations qui affectent le marché mondial par une concurrence exacerbée. De plus le marché est caractérisé par une forte présence dôinformation constituant une matière première stratégique à. tous les niveaux

Le défi à relever dôune part est dôobtenir de lôinformation, grâce a des outils dédiés, et les exploiter en temps opportun, et dôautre part protéger les données stratégiques détenues par toute organisation.

3.2 .Historique :

Les principales dates à retenir construisant l'histoire de l'Entrepôt de données sont les suivantes :

et l'Université Dartmouth, dans un projet conjoint, créent
ns".
sa base de données managériale un système exclusivement destiné à la prise de décision.

- 1988 - Barry Devlin et Paul Murphy publient l'article "Une architecture pour les systèmes d'information financiers" ("*An architecture for a business and information systems*") où ils utilisent pour la première fois le terme "*Datawarehouse*".
- 1990 - Red Brick Systems crée Red Brick Warehouse, un système spécifiquement dédié à la construction de l'Entrepôt de données.
- 1991 - Bill Inmon publie *Building the Data Warehouse* (*Construire l'Entrepôt de Données*).
- 1995 - Le *Data Warehousing Institute*, une organisation à but lucratif destinée à promouvoir le datawarehousing, est fondé.
- 1996 - Ralph Kimball publie *The Data Warehouse Toolkit* (*La boîte à outils de l'Entrepôt de données*). [3.1]

3.3.Concepts fondamentaux :

3.3.1. Entrepôts de données :

Un entrepôt de données, ou data Warehouse, est une vision centralisée et universelle de toutes les informations de l'entreprise. C'est une structure (comme une base de données) qui à pour but, contrairement aux bases de données, de regrouper les données de l'entreprise pour des fins analytiques et pour aider à la décision stratégique. La décision stratégique étant une action entreprise par les décideurs de l'entreprise et qui vise à améliorer, quantitativement ou qualitativement, la performance de l'entreprise.

En gros, c'est un gigantesque tas d'informations épurées, organisées, historisées et provenant de plusieurs sources de données, servant aux analyses et à l'aide à la décision. L'entrepôt de données est l'élément central de l'informatique décisionnelle. En effet, l'entrepôt de données est le meilleur moyen que les professionnels ont trouvé pour modéliser de l'information pour des fins d'analyse, mais il ne serait pas étonnant que d'ici quelques années un nouveau concept apparaisse pour révolutionner l'informatique décisionnelle.

3.3.2.Data mart, ou magasin de données :

Les Data Warehouses étant, en général, très volumineux et très complexes à concevoir, on a décidé de les diviser en bouchées plus faciles à créer et entretenir. Ce sont les Data Marts. On peut faire des divisions par fonction (un data mart pour les ventes, pour les commandes, pour les ressources humaines) ou par sous-ensemble organisationnel
(un data mart par succursale). Nous verrons plus tard comment organiser les data marts pour créer un entrepôt proprement dit.

de données et une base de données :

ôle stratégique dans la vie dôune entreprise. Il stocke des données pertinentes aux besoins de prise de décision en provenance des systèmes opérationnels de lôentreprise et dôautres sources externes. Lôinterrogation est lôopération la plus utilisée dans le contexte dôentrepôt de données où la mise à jour consiste seulement à alimenter lôentrepôt.

Le tableau 4.1 montre la différence entre un système opérationnel doté dôune base de données classique et un entrepôt de données :

	Bases de données	Entrepôts de données
But	Exécution dôun processus métier	Evaluation dôun processus métier
Interaction avec lôutilisateur	Insertion, Modification, Interrogation, Suppression	Interrogation
Données	Courantes	Courantes et Historiques
Usages	Support de lôopération de lôentreprise	Support de lôanalyse de lôentreprise
Requêtes	Simples, Prédéterminées	Complexes, Ad-hoc
Type utilisateur	Employé	Décideur
Principe de conception	Troisième forme normale	Conception multidimensionnelle

Tab. 3.1 ï Base de données vs Entrepôt de données

3.4. Principe de fonctionnement :

Intégration

Dans les faits, les données alimentant l'Entrepôt de données sont hétérogènes, issues de différentes applications de production, voire de fichiers dits "plats" (fichiers Excel, fichiers texte, XML...). Il sôagit alors de les intégrer, de les homogénéiser et de leur donner un sens unique compréhensible par tous les utilisateurs. La transversalité recherchée sera dôautant plus efficace que le système dôinformation sera réellement intégré dans sa globalité. Cette intégration nécessite notamment :

- une forte activité de normalisation et de rationalisation, orientée vers la qualité ;
- une bonne gestion des référentiels, incluant une vérification constante de leur intégrité ;
- une parfaite maîtrise de la sémantique et des règles de gestion des métadonnées manipulées.

on repose sur la standardisation de données internes à
s externes (provenant par exemple de clients ou de

Ce nôest quôau prix dôune intégration poussée que lôon peut offrir une vision homogène et véritablement transverse de lôentreprise. Ceci suppose que le système dôinformation de lôentreprise en amont soit bien structuré, bien maîtrisé, et bénéficie déjà dôun niveau dôintégration suffisant. Si tel n'est pas le cas, la mauvaise qualité des données peut empêcher la mise en î uvre de l'entrepôt de données.

Historisation

L' historisation d'un Datawarehouse repose sur le principe de conservation des données (ou de non-volatilité des données). Afin de conserver la traçabilité des informations et des décisions prises, les données une fois entrées dans l'Entrepôt sont stables, en lecture seule, non modifiables par les utilisateurs. Une même requête lancée plusieurs fois à différents moments doit ainsi restituer les mêmes résultats. Dès quôune donnée est qualifiée pour être introduite dans l'Entrepôt de données, elle ne peut donc plus être altérée, modifiée ou supprimée (en delà dôun certain délai de purge). Elle devient, de fait, partie intégrante de lôhistorique de lôentreprise.

Le principe de non-volatilité tranche avec la logique des systèmes de production, qui bien souvent remettent à jour les données par « annule et remplace » à chaque nouvelle transaction. Chaque donnée collectée se voit affecter une date ou un numéro de version pour éviter de recouvrir une information déjà présente dans la base de données, et permettre de suivre son évolution au cours du temps. Il y a de cette manière conservation de l'historique.

Dôun point de vue fonctionnel, cette propriété permet de suivre dans le temps lôévolution des indicateurs et de réaliser des analyses comparatives (par exemple, les ventes d'une année sur l'autre). De ce fait, dans un entrepôt de données, un référentiel de temps unique est nécessaire.

Organisation fonctionnelle

L'Entrepôt de données intègre au sein d'une même base les informations provenant de multiples applications opérationnelles. On passe ainsi dôune vision verticale de lôentreprise, dictée par des contraintes techniques, à une vision transversale, dictée par le besoin métier, qui permet de croiser fonctionnellement les informations. Lôintérêt de cette organisation est de disposer de lôensemble des informations utiles sur un sujet le plus souvent transversal aux structures fonctionnelles (services) de lôentreprise. On dit que l'Entrepôt de données est orienté « métier », en réponse aux différents métiers de lôentreprise dont il prépare lôanalyse.

D'un point de vue conceptuel, les données d'un Data warehouse sont interprétables sous forme d' *indicateurs* répartis selon des *axes* (ou *dimensions*) : par exemple, le nombre de clients (indicateur) réparti par jour de vente, magasin ou segment de clientèle (axes). Techniquement, la modélisation de l'Entrepôt de données peut matérialiser cette organisation sous forme de *tables de fait* ou et de *tables de* référentiel.

ructure de données qui peut en général être représentée par un modèle de données normalisé 3FN (**(en)**3NF) pour les données de détail et/ou en étoile ou en flocon pour les données agrégées et ce dans un SGBD relationnel (notamment lorsqu'il s'agit de données élémentaires ou unitaires non agrégées). La traduction technique de ce modèle se fait souvent au sein d'un cube OLAP.

L'Entrepôt de données est conçu pour contenir les données en adéquation avec les besoins de lôorganisation, et répondre de manière centralisée à tous les utilisateurs. Ainsi, il nôexiste pas de règle unique en matière de stockage ou de modélisation.

Ainsi, ces données peuvent donc être conservées :

- de préférence, sous forme *élémentaire* et détaillée (exemple : pour une banque, chaque opération sur chaque compte de chaque client) si la volumétrie le permet. Les données élémentaires présentent des avantages évidents (profondeur et niveau de détail, possibilité d'appliquer de nouveaux axes d'analyse et même de revenir *a posteriori* sur le « passé ») mais représentent un plus grand volume et nécessitent donc des matériels plus performants.

- éventuellement, sous forme *agrégée* selon les axes ou dimensions d'analyse prévus (mais ces agrégations sont plutôt réalisées dans les datamarts que dans les entrepôts de données proprement dits). Les données agrégées présentent d'autres avantages (facilité d'analyse, rapidité d'accès, moindre volume). Par contre, il est impossible de retrouver le détail et la profondeur des indicateurs une fois ceux-ci agrégés : on prend le risque de figer les données selon une certaine vue avec les axes d'agrégation retenus, et de ne plus pouvoir revenir sur ces critères si l'on n'a pas conservé le détail (par exemple, si l'on a agrégé les résultats par mois, il ne sera plus possible de faire une analyse par journée) [3.2]

3. 5.Modélisation des entrepôts de données

3.5.1. Concepts de base :

La modélisation des entrepôts de données se base sur deux concepts fondamentaux, le concept de fait et le concept de dimension :

- Un ***fait*** représente le sujet ou le thème analysé. Il présente un centre dôintérêt de lôentreprise et est considéré comme un concept clé sur lequel repose le processus de prise de décision. Un fait est formé de « *mesures* » ou attributs du fait (atomiques ou dérivés) qui correspondent aux informations liées au thème analysé. Les mesures sont stockées dans des tables de faits qui contiennent les valeurs des mesures et les clés vers les tables de *dimensions*.

exte dôanalyse dôun fait et se présente sous forme dôune iérarchique. Par exemple, pour la dimension *temps*, nous *année, semestre, trimestre, mois, semaine et jour*.
Les dimensions sont caractérisées par des attributs de dimensions. Elles sont stockées dans des tables de dimensions qui contiennent *les niveaux hiérarchiques* des dimensions ainsi que les formules à appliquer sur les données numériques pour passer dôun niveau à un autre. Une hiérarchie est dite complète si tous les objets dôun niveau de la hiérarchie appartiennent à une seule classe dôobjets dôun niveau supérieur et forment cette classe. [3.3]

3. 5. 2. La modélisation multidimensionnelle *:* Il existe deux types de modélisation logique :

Modèle en étoile : Il se compose dôune table de faits centrale et un ensemble de tables de dimension. La table de fait est normalisée et peut atteindre une taille importante. Les tables de dimension sont dénormalisées et souvent sont de petite taille. Les faits sont généralement quantitatifs alors que les dimensions sont qualitatives (descriptions textuelles des faits). La clé primaire de la table de faits est le résultat de la concaténation des clés de dimensions.

Avantages :- Facilité de navigation

- Nombre de jointures limité.

Inconvénients :- Redondance dans les dimensions

-Toutes les dimensions ne concernent pas les mesures

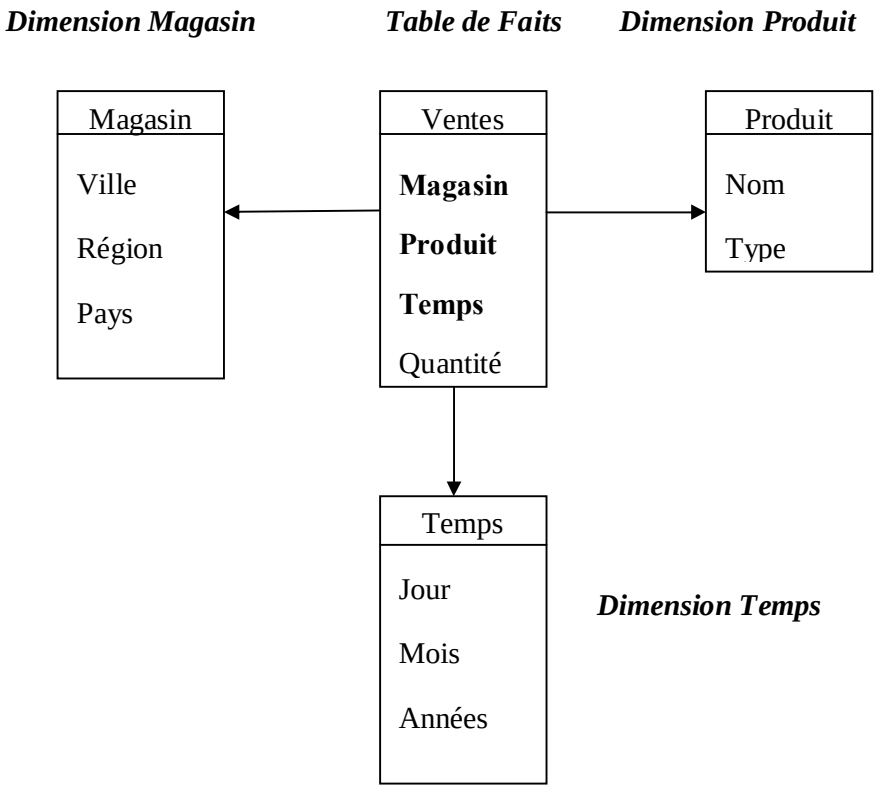


Fig.3.1: Schéma en étoile

schéma en étoile dans lequel les dimensions ont été dœlles et faisant ainsi apparaitre des hiérarchies de

Avantages : -Normalisation des dimensions

- Économie dœspace disque.

Inconvénients : -Modèle plus complexe (jointure)

-Requêtes moins performantes.

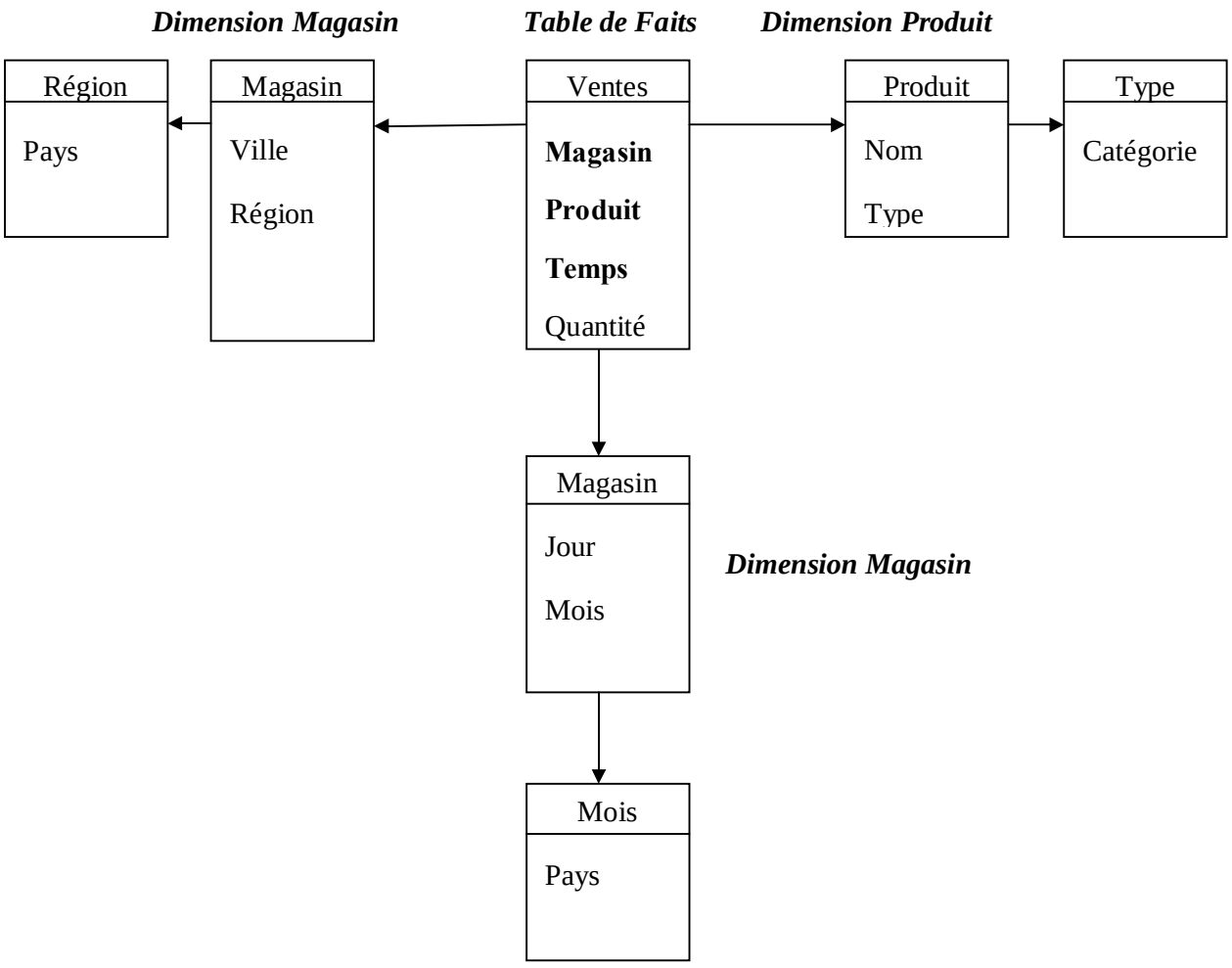


Fig.3.2 : Schéma en flocon de neige

t de données :

Le processus de construction dôun entrepôt de données est composé de trois principales phases :
(1) extraction des données à partir des différentes sources,
(2) organisation et intégration des données dans lôentrepôt,
(3) accès aux données intégrées et analyse de ces dernières.

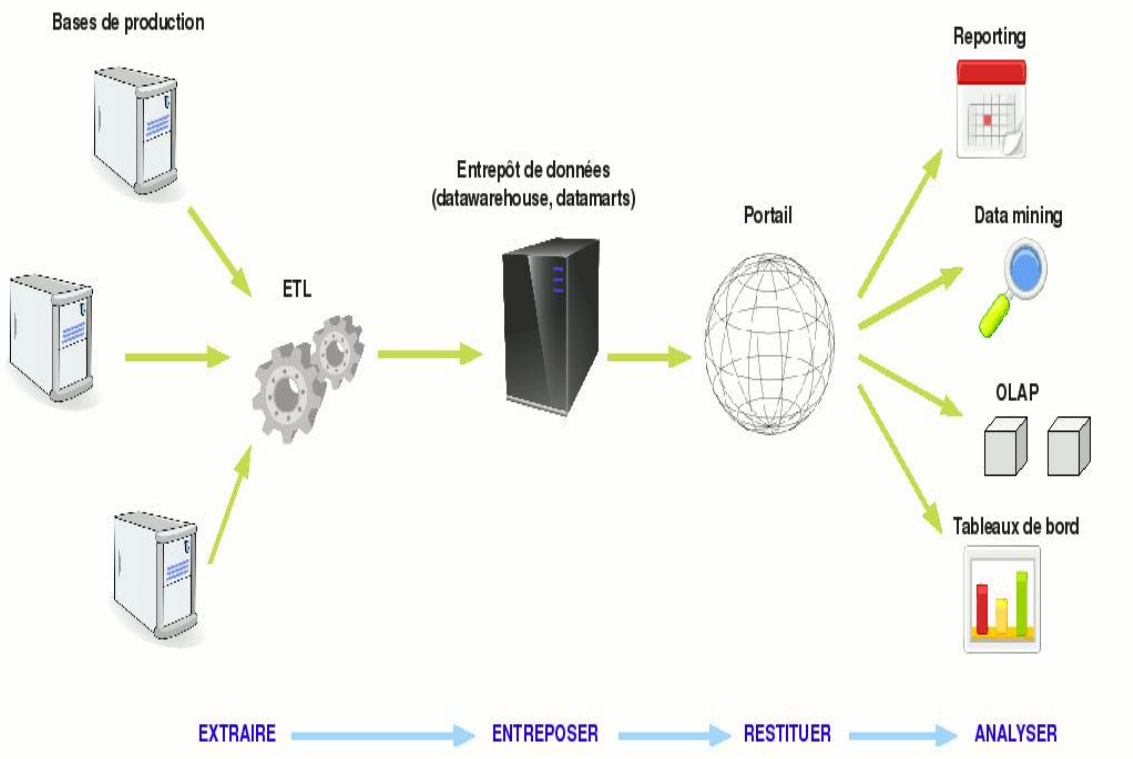


Fig.3.3 : Processus de construction dôun entrepôt de données

Les métadonnées contiennent des informations utiles sur la création, lôutilisation et la gestion de lôentrepôt. Durant la troisième phase, un serveur OLAP se charge de présenter les informations demandées par les utilisateurs sous plusieurs formes : tableaux, rapports, statistiques, etc. [3.4]

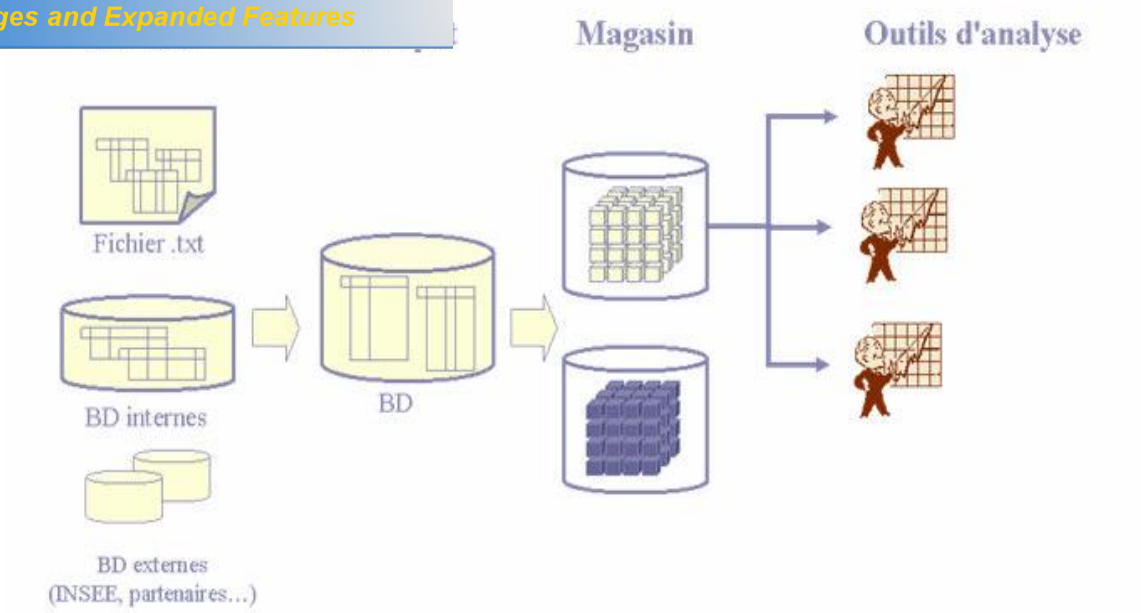


Fig. 3.4 : Architecture dôun entrepôt de données

3.7. ETL (Extract, Transform and Loading):

Les données du Data Warehouse sont, pour la plupart, issues des différentes sources de données opérationnelles de l'entreprise. Des solutions logicielles sont alors nécessaires à leur intégration et à leur homogénéisation. Ces outils ont pour objet de s'assurer de la cohérence des données du Data Warehouse et d'homogénéiser les différents formats trouvés dans les bases de données opérationnelles.

L'ETL (Extract Transform Load) est donc lôensemble de processus dôintégration des données dans un entrepôt, qui commence par lôextraction des données des sources, ensuite vient la transformation qui consiste à nettoyer et filtrer les données (extraire que les données pertinentes pour lôaide à la prise de décision) et se termine par le chargement des données dans lôentrepôt. Ce processus nécessite la mise en î uvré dôune stratégie pour le rafraichissement (mise à jour) des données qui doit être effectuée périodiquement.
[3.5]

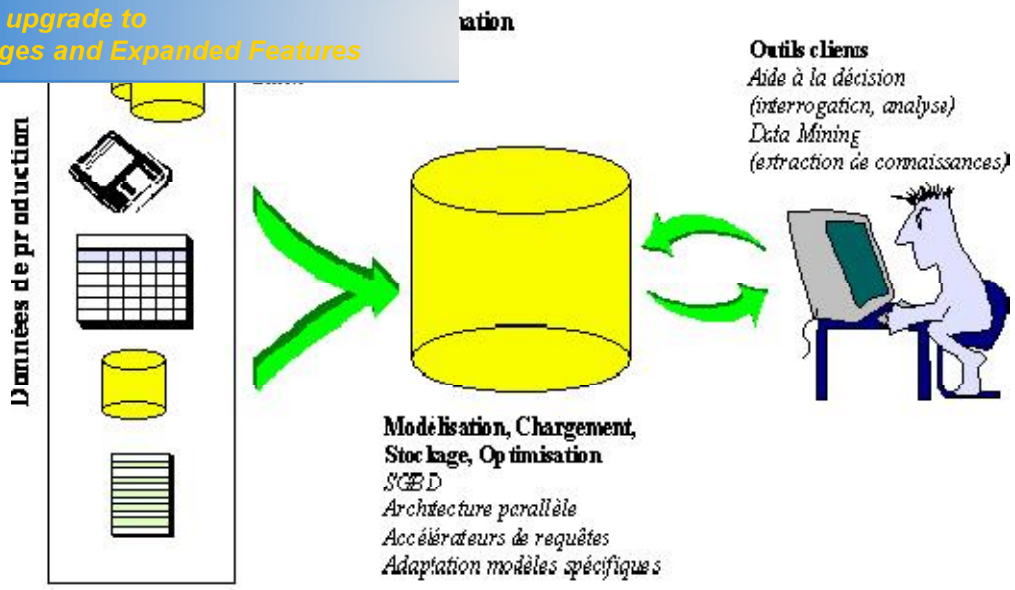


Fig. 3.5: Besoins et outils d'un Data Warehouse

3.7.1 Extraction des données

L'extraction des données consiste à collecter les données utiles dans le système de production (SGBD, ERP, cahiers à plat, etc). Pour rafraîchir la base de données décisionnelle, il faut identifier les données ayant évolué afin d'en extraire le minimum, puis planifier ces extractions afin d'éviter les saturations de production.
Généralement l'extraction de données est différentielle, les données sont historisées. Cette fonctionnalité devient importante lorsque le volume des données est important. L'intégrité des données est indispensable et nécessite la synchronisation des différents processus d'extraction. Les problèmes liés à cette synchronisation peuvent être complexe, soit fonctionnellement, soit techniquement dans des environnements très hétérogènes. Un autre problème est de traiter les données externes. Il faut maintenir une surveillance du système d'information pour pouvoir les identifier et s'assurer que ce sont les bonnes données qui sont recensées. De plus, la forme de ces données, qui est souvent totalement anarchique, accentue la difficulté. Pour être utilisables, ces données nécessitent un reformatage pour pouvoir les incorporer dans une forme exploitable pour l'entreprise.
Enfin le troisième problème vient de l'apparition imprévisible de ces données qui les rend difficiles à capter. En conséquent, l'outil d'extraction doit attaquer toutes sorte de sources de données sans être perturbé et s'adapter au future.
Pour extraire les données sources, il y a plusieurs technologies utilisables :
- des passerelles, fournies principalement par les éditeurs de bases de données. Ces passerelles sont généralement insuffisantes car elles sont mal adaptées aux processus de transformation complexes ;
-des utilitaires de réplication, utilisables si les système de production et décisionnel sont homogènes et si la transformation à appliquer aux données est légère ;
-des outils spécifiques d'extraction. Ces outils sont certainement la solution opérationnelle au problème de l'extraction, mais leur prix relativement élevé est unfrein à leur utilisation dans les premières applications.

t une discipline sur laquelle de nombreux éditeurs travaillent actuellement. Outre la qualité des données qu'ils permettent d'auditer et éventuellement d'améliorer, les outils de nettoyage permettent de supprimer les doublons dans les fichiers. Il s'agit à ce stade d'appliquer des filtres prédéfinis sur les données afin d'attribuer des valeurs cohérentes aux variables mal ou non renseignées ou encore d'harmoniser les formats (date : jj/mm/aaaa). On peut également avoir à convertir les données d'un format EBCDIC (codage de caractères) vers ASCII. Aussi, des données du système opérationnel doivent être agrégées ou calculées avant leur chargement dans la basse décisionnelle. Il faut également pouvoir associer des champs sources avec des champs cibles. Il existe plusieurs niveaux de complexité pour ces associations (cardinalités 1-1, 1-N,N-1, N-N), comme par exemple :
-le transfert du nom du client vers le champs cible
-la décomposition d'une adresse vers les champs numéro, rue, ville ou l'inverse.

Certains outils peuvent également réaliser des analyses lexicales des champs sources. Ils seront dons capables de comprendre que les champs suivants signifient la même chose : Boulvd , Bd, Boulevard.
En complément, on trouve des outils d'audit et d'analyse pour assurer le suivi du processus afin de pouvoir contrôler les rejets par exemple.

3.7.3. Le chargement des données

Le chargement est la dernière phase de l'alimentation du Data Warehouse. C'est une phase délicate notamment lorsque les volumes sont importants. Pour obtenir de bonnes performances en chargement, il est impératif de maîtriser les structures du SGBD (tables et index) associées aux données chargées afin d'optimiser au mieux ces processus. Les techniques de parallélisation optimisent les chargements lourds. Pour les mettre en oeuvre, des utilitaires particuliers existent chez la majorité des éditeurs de bases de données.

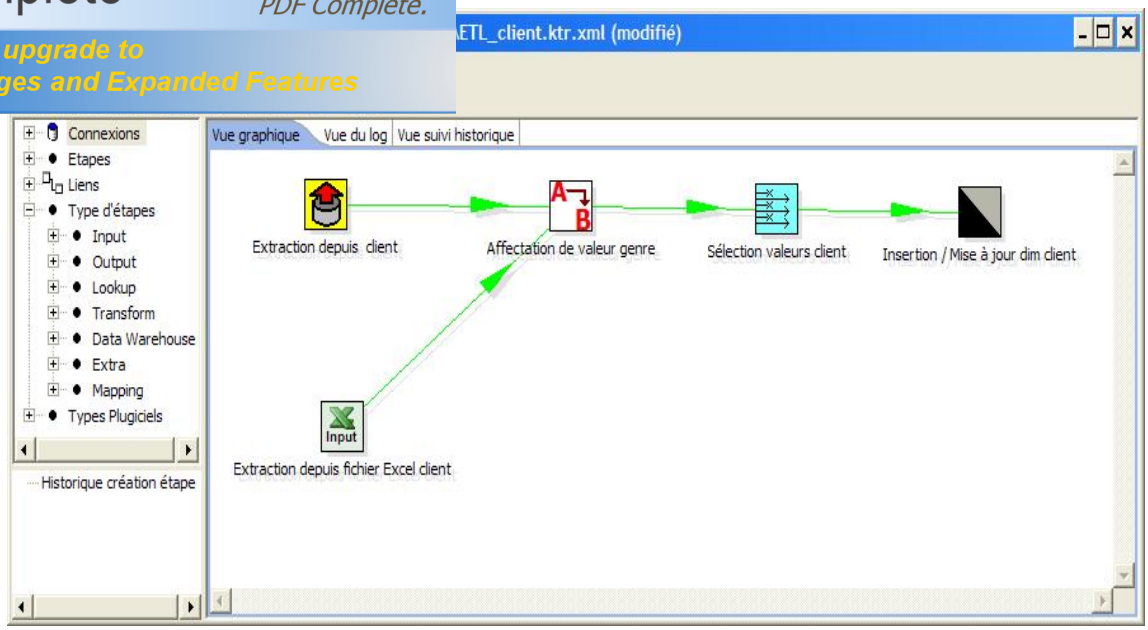


Fig 3.6 : Aperçu dôun ETL

3.8. L'outil OLAP :

3.7.1. Apparition :

En 1993, Codd, fondateur des bases de données relationnelles, définit le concept OLAP (On Line Analytical Processing) dans un document technique sous le titre de « Providing OLAP (On-Line Analytical Processing) to User-Analysts : An IT Mandate»[Cod93]. Il prescrit 12 regles de conception pour ce quôon appelle le « modèle OLAP » :

ï Règle 1 - Multidimensionnalité : le modèle OLAP est multidimensionnel par nature
ï Règle 2 - Transparence : lôemplacement physique du serveur OLAP est transparent pour lôutilisateur
ï Règle 3 - Accessibilité : lôutilisateur OLAP dispose de lôaccessibilité a toutes les données nécessaires à ses analyses
ï Règle 4 - Stabilité : la performance des interrogations reste stable indépendamment du nombre de dimensions
ï Règle 5 - Client-Serveur : le serveur OLAP sôintègre dans une architecture client serveur
ï Règle 6 - Dimensionnement : le dimensionnement est générique afin de ne pas fausser les analyses
ï Règle 7 - Gestion complète : le serveur OLAP assure la gestion des données clairsemées
ï Règle 8 - Multiutilisateurs : le serveur OLAP offre un support multiutilisateurs (gestion des mises a jour, intégrité, sécurité)
ï Règle 9 - Inter Dimension : le serveur OLAP permet la réalisation dôopérations interdimensions sans restriction
ï Règle 10 - Intuitif : le serveur OLAP permet une manipulation intuitive des données
ï Règle 11 - Flexibilité : la flexibilité (ou souplesse) de lôédition des rapports est intrinsèque au modèle

: le nombre de dimensions et de niveaux dôagrégation es analyses les plus poussées.

Sôen sont suivies, en 1995, 6 règles supplémentaires moins connues, ainsi quôune restructuration de ces règles en « factures », que lôon peut traduire par dispositifs.

Nigel Pendse, lôinitiateur de lôOlap Report1, évoque que ce terme OLAP nôest pas explicite, il ne fournit pas une définition et ne permet pas de savoir si un outil relève ou non de cette technologie. Par ailleurs, 12 règles ou 18 dispositifs, côest, selon lui, une quantité trop importante pour être facilement retenue. Nigel Pendse suggère alors dôassimiler le terme OLAP a une définition comprenant cinq termes :
Fast Analysis of Shared Multidimensional Information (le modèle FASMI)2. Cette définition correspond finalement aux critères retenus pour simplifier les règles deCodd et faciliter lôévaluation des outils OLAP. Elle a été traduite en français par « Analyse Rapide dôInformation Multidimensionnelle Partagée ».
Ainsi, on parle généralement de système OLAP, sous-entendant ainsi un système dôinformations répondant aux règles évoquées par Codd ou aux critères FASMI.

3.8.2.Comparaison OLTP/ OLAP :

A la différence dôune base de données classique supportant des requêtes transactionnelles de type OLTP (On-Line Transaction Processing), un entrepôt de données est conçu pour supporter des requêtes de type OLAP (On-Line Analytical Processing).

Les applications OLTP ont une architecture permettant de gérer des transactions en temps réel. Elles peuvent être vues comme des applications de production permettant dôeffectuer des traitements factuels tels que lôajout, la suppression et la modification de clients par exemple. Les applications OLAP sont, quant a elles, des applications dôaide a la décision permettant dôeffectuer des traitements ensemblistes qui ont pour objectif de résumer lôinformation en réduisant une population a une valeur, un comportement. [3.6]

	OLTP	OLAP
Données	Exhaustives, détaillées	Agrégées, résumées
	Courantes	Historiques
	Mises à jour	Recalculées
	Dynamiques	Statiques
	Orientées applications	Orientées sujets dôanalyse
	De lôordre des gigaoctets	De lôordre des téraoctets
Utilisateurs	Agents opérationnels	Décideurs
	Nombreux	Peu nombreux
	Concurrents	Non concurrents
	Mises à jour et interrogations	Interrogations
	Requêtes prédéfinies	Requêtes imprévisibles
	Réponses immédiates	Réponses moins rapides
	Accès à peu dôinformations	Accès à de nombreuses informations

Tab 3.2 ï OLTP versus OLAP

Remarque :Les modèles de conception des systèmes transactionnels OLTP ne sont pas adaptés aux systèmes OLAP dont les requêtes sont souvent très complexes, utilisent beaucoup de jointures, demandent beaucoup de temps de calcul et sont de nature ad hoc. Pour ce type dôenvironnement OLAP, une nouvelle approche de modélisation a été proposée : *la modélisation multidimensionnelle*.

3.8.3.Types dôOLAP :

ROLAP(Relational On-Line Analytical Process) **:**

Permet le stockage infini des données dans un SGBD relationnel. SQL (Structured Query Language) est utilisée pour effectuer les différents traitements sur les données, en utilisant des requêtes en générale très complexes et très exigeantes en termes de ressources et de temps dôexécution. Les systèmes relationnels sont plus performants pour de gros volumes de données et proposent des mécanismes de sécurité plus poussés (poser un filtre sur les requêtes, réduisant lôaccès à certaines données à un groupe dôutilisateurs).

MOLAP (Multidimensional On-Line Analytical Process)**:**

Côest une approche multidimensionnelle « pure », les données sont stockées dans une base de données multidimensionnelle (cube) le plus souvent propriétaire. Il ya la limitation quand à la quantité des données traitées. Serveur de traitement OLAP est lôapproche la plus adaptée aux traitements. Un serveur, conjointement avec la base de données, est dédié à effectuer les différents traitements de données. Dans ce cas les performances en termes de rapidité sont généralement très bonnes.

HOLAP (Hybrid On-Line Analytical Process)**:**

Côest une approche qui permet de combiner les avantages de ROLAP et de MOLAP. Il sôagit dôexploiter une technologie multidimensionnelle pour les données agrégées (agrégats) et une approche relationnelle pour les données détaillées.

DOLAP (Desktop On-Line Analytical Process)**:**

Apparue assez récemment, côest une petite quantité de données (mini-base multidimensionnelle ou extraction de cube) dans un fichier sur le poste client de l'utilisateur. Un nombre limité de traitements OLAP se font sur le poste client de lôutilisateur et sont assurés par un client de traitement OLAP. Il sôagit dôune alternative « bureau ».
Aujourdôhui plusieurs termes sont apparus comme JOLAP (JAVA OLAP) qui est une API (Application Programming Interface) JAVA (Langage de programmation orienté objet) et qui permet de se connecter à des applications et des serveurs OLAP, tentant de normaliser lôaccès aux bases de données multidimensionnelles et également comme OOLAP (Object OLAP), faisant référence à lôutilisation du modèle objet, mais cette technologie nôapparait pas dans les solutions commerciales et elle fait lôobjet de travaux de recherche [3.7]

L'hypercube OLAP donne accès à des fonctions d'extraction de l'information (pour visualisation, analyse ou traitement), et à des fonctions de requête en langage MDX (comparable à SQL pour une base de données relationnelles).

- *Rotate* : sélection du couple de dimensions à cibler,
- *Slicing* : extraction d'une tranche d'information,
- *Scoping* : extraction d'un bloc de données (opération plus générale que le *slicing*),
- *Drill-up* : synthèse des informations en fonction d'une dimension (exemple de *drill-up* sur l'axe temps : passer de la présentation de l'information jour par jour sur une année, à une valeur synthétique pour l'année),
- *Drill-down* : c'est l'équivalent d'un « zoom », opération inverse du *drill-up*,
- *Drill-through* : lorsqu'on ne dispose que de données agrégées (indicateurs totalisés), le *drill through* permet d'accéder au détail élémentaire des informations (voir notamment les outils H-OLAP). [3.8]

3.8 .5.Langage MDX :

Le MDX (*Multidimensional Expressions*, « expressions multidimensionnelles ») est un langage de requête pour les bases de données OLAP, analogue au rôle de SQL pour les bases de données relationnelles. C'est aussi un langage de calcul avec une syntaxe similaire à celle des tableurs.

Le langage des expressions multidimensionnelles propose une syntaxe spécialisées pour interroger et manipuler les données multidimensionnelles mémorisées dans un cube OLAP. Bien qu'il soit possible de traduire certaines expressions dans le langage SQL traditionnel, cela nécessite une syntaxe SQL souvent maladroite même pour des expressions MDX très simples. MDX a été adopté par une large majorité de fournisseur de la technologie OLAP et est devenu un standard de facto pour les systèmes OLAP. [3.9]

Caractéristiques du langage MDX :

Le MDX offre une syntaxe assez similaire à celle de SQL (Structured Query Language), mais il ne s'agit pas d'une extension du langage SQL, en fait, certaines des fonctionnalités prises en charge par MDX peuvent être fournies avec certes moins d'efficacité ou de facilité, par SQL.

A l'instar d'une requête SQL, chaque requête MDX nécessite une demande de données (la clause SELECT), un point de départ (la clause FROM) et un filtre (la clause WHERE). Ces mots clés, ainsi que d'autres, fournissent les outils nécessaires à l'extraction de portions de données spécifiques d'un cube en vue d'une analyse. Le MDX fournit également un jeu robuste de fonctions pour la manipulation des données extraites, ainsi que la possibilité d'enrichir MDX avec des fonctions définies par l'utilisateur.

MDX, comme, SQL fournit un langage de définition de données, DDL (Data Definition Language), pour la gestion des structures de données. Certaines commandes de MDX

des cubes, des dimensions, des mesures, ainsi que les

Forme d'une requête OLAP sous MDX :

Le cube est l'élément (structure) clé dans les bases de données multidimensionnelles. Un cube peut avoir plusieurs dimensions et la dimension Mesures qui est la seule que chaque cube doit contenir. Chaque dimension des membres provenant d'attributs et chaque membre représente un élément unique au sein d'une dimension. Les mesures sont les membres de la dimension Mesures et ces mesures représentent les données organisées par les autres dimensions du cube. Les concepts de cube, dimension, membre et mesure sont importants pour mieux se rendre compte de la syntaxe MDX (voir exemple).
Une requête MDX permet de rechercher des mesures à des niveaux différents de granularité, en formulant une expression unique et peut construire une visualisation pour l'ensemble des résultats.

Pour cela, une syntaxe a été proposée pour rédiger une requête OLAP sous MDX dans et qui se présente ainsi : [3.10] [3.11] [3.12]

SELECT *{Produit.[P1],Produit.[P2]} ON COLUMNS,*
{Temps.[T1],Temps.[T2]} ON ROWS
FROM *Ventes*
WHERE *Mesures.quantite*

Exemples :

1-
```
SELECT
  { [Measures].[Store Sales] } ON COLUMNS,
  { [Date].[2002], [Date].[2003] } ON ROWS
FROM Sales
WHERE ( [Store].[USA].[CA] )
```

Dans cet exemple, la requête renvoie les résultats suivants :

La clause SELECT définit les axes de la requête. Ce s ont le montant des ventes en magasin (Store Sales Amount) ainsi que les dates 2002 et 2003. La clause FROM indique que la requête utilise le cube "Sales" comme source de données. La clause WHERE définit le membre "California" de la dimension "Store" en tant qu'axe de transition.

2-

```
SELECT {[year].members} ON COLUMNS,
CROSSJOIN ({[Location].Tours, [Location].Orleans, [Location].Centre},
{[category].food, [category].drink}) ON ROWS
FROM SalesCube
WHERE [Measures].quantity
```

Cette expression MDX recherche la quantité de produits vendus par année et catégorie aux niveaux indiqués de la dimension *Location (city, region).*

Structure du résultat d'une requête MDX :

Le résultat d'une telle requête est un tableau multidimensionnel, dont les cellules ne contiennent qu'une valeur (contrairement aux cubes) et dans lequel on peut naviguer avec les opérations décrites dans le chapitre précédent.
Sachant qu'une requête MDX permet de rechercher des mesures à des niveaux différents de granularité en formulant une expression unique et de spécifier comment afficher la réponse sous forme de table croisée. De plus la syntaxe MDX distingue l'identification des axes d'affichage par 'COLUMNS' et 'ROWS'. Ainsi, l'emplacement des informations de la requête sur ces axes est précisé.
Exemple: si nous avons un cube 'Ventes', le résultat de la requête de l'exemple précédent est une table croisée de deux lignes et deux colonnes peut se présenter ainsi :

	P1	P2
T1	25	18
T2	42	80

Tab 3.3: Résultat d'une requête MDX

Les avantages et inconvénients de MDX :

Le MDX possède une syntaxe un peu ardue. Néanmoins le temps que nous passons à son assimilation est largement compensé par celui que nous gagnons lors de l'écriture des requêtes du langage SQL .De plus, les fondements et les formalisations théoriques de ce langage sont mal connus, et il présente l'inconvénient de ne pas être clos.

Si l'utilisation interactive des cubes est intéressante, leur structure multidimensionnelle associée à la puissance de ce langage en fait des outils de prédilection pour le reporting opérationnel d'entreprise ou d'administration. Au-delà de cet aspect, le MDX permet d'explorer les données de cubes et crée la vraie valeur autour des données et c'est ici que se trouve la plus value (méritée). Il n'en reste pas moins que le MDX reste un langage, qui n'est pas forcément simple à appréhender. [3.13] [3.14]

Exemple de fouille :

Il s'agit de trouver des informations dans un entrepôt, allant au delà de ce que l'on peut aisément et efficacement exprimer avec une requête.

Dans une base de données de ventes,
Ventes {id_panier,produit} ,on essaie de trouver quels éléments apparaissent souvent simultanément.
Le **but** est, par exemple, de cibler une offre promotionnelle ou de proposer des achats groupés.

3.9. Construction d'un entrepôt de données :

Nous avons vu comment créer une étoile ou un flocon, nous avons vu que les data marts sont des étoiles regroupées par fonction ou par utilité dans l'entreprise et nous savons qu'un entrepôt est l'ensemble de tous les data marts de l'entreprise. Nous savons faire une étoile, mais comment les regrouper pour mettre en î uvre un entrepôt de données ? Et bien trois méthodes s'offrent à nous :

Top-Down : c'est la méthode la plus lourde, la plus contraignante et la plus complète en même temps. Elle consiste en la conception de tout l'entrepôt (ie : toutes les étoiles), puis en la réalisation de ce dernier. Imaginez le travail qu'une telle méthode implique : savoir à l'avance toutes les dimension et tous les faits de l'entreprise, puis réaliser tout ça. Le seul avantage que cette méthode comporte est qu'elle offre une vision très claire et très conceptuelle des données de l'entreprise ainsi que du travail à faire.

Bottom-Up : c'est l'approche inverse, elle consiste à créer les étoiles une par une, puis les regrouper par des niveaux intermédiaires jusqu'à obtention d'un véritable entrepôt pyramidal avec une vision d'entreprise. L'avantage de cette méthode est qu'elle est simple à réaliser (une étoile à la fois), l'inconvénient est le volume de travail d'intégration pour obtenir un entrepôt de données ainsi que la possibilité de redondances entre les étoiles (car elles sont faites indépendamment les unes des autres).

Middle-Out : c'est l'approche hybride, et conseillée par les professionnels du BI. Elle consiste en la conception totale de l'entrepôt de données (ie : concevoir toutes dimensions, tous les faits, toutes les relations), puis créer des divisions plus petites et plus gérables et les mettre en î uvre. Cela équivaut à découper notre conception par éléments en commun et réaliser les découpages un par un. Cette méthode tire le meilleur des deux précédentes sans avoir les contraintes. Il faut juste noter que cette méthode implique, parfois, des compromis de découpage (dupliquer des dimensions identiques pour des besoins pratiques). [3.15]

3.10. Exploitation de l'entrepôt :

L'entrepôt a pour objectif final l'analyse des données en vue de la prise de décision. Différents types d'analyse peuvent être réalises : analyses statistiques, fouille de données, etc. Mais on peut également parler d'analyse en ligne OLAP. Il s'agit en l'occurrence d'une navigation dans les données. Cette analyse peut être qualifiée d'exploratoire. Le principe général est d'arriver au cours de la navigation à détecter des points intéressants qu'on essaye de décrire, d'expliquer en naviguant, par exemple en allant chercher davantage de détails. Le rôle de l'utilisateur est ici central puisque c'est lui qui réalise la navigation ; celle-ci nécessite une connaissance du domaine afin d'être en mesure de savoir si les valeurs des mesures sont intéressantes ou non. En l'occurrence, elles peuvent être intéressantes lorsqu'elles sont aberrantes. Certains travaux s'intéressent aujourd'hui à la détection automatique de ces points aberrants [3.16]

érents operateurs sôappliquent au niveau dôun cube de deux catégories : les operateurs lies a la structure et les ne manière générale, les operarteurs lies a la structure permettent la manipulation et la visualisation du cube.
Quant aux opérations lies a la granularité, il sôagit dôagréger les données pour obtenir des données résumées et inversement. Pour enrichir ces operateurs de navigation, certains auteurs se sont focalises sur la construction dôoperateurs plus puissants, qui intègrent des méthodes dôextraction de connaissances, telles que la découverte de règles dôassociation par exemple.

3.11. Domaines d'utilisation des entrepôts de données :

Commerce : Ciblage de clientèle, Déterminer des promotions
Logistique : Adéquation demande/production
Banque :Risques dôun prêt, prime plus précise
Santé : Épidémiologie, Risque alimentaire
Assurance : Risque lié à un contrat dôassurance (voiture)
é

3.12. Quelques métiers du décisionnel :

-Customer Relationship Management (gestion de la relation client) :Améliorer la connaissance client, identifier et prévoir la rentabilité client, accroitre lôefficacité du marketing client

-Supplier Relationship Management (gestion de la relation fournisseur) : Classifier et évaluer lôensemble des fournisseurs. Planifier et piloter la stratégie Achat.

-Strategic Performance Management :Déterminer et contrôler les indicateurs clé de la performance de lôentreprise

-Finance Intelligence :Planifier, analyser et diffuser lôinformation financière. Mesurer et gérer les risques

-Human Capital Management (gestion de la relation avec les employés) :Aligner les stratégies RH, les processus et les technologies.

Dans ce chapitre, nous avons évoqué les principaux concepts lies à la conception et a lœexploitation des entrepôts de données. Cette présentation nous a permis de positionner le contexte général des travaux qui seront présentés. La réussite du processus dœentreposage repose entre autres sur une bonne conception du schéma, puisque cœest ce dernier qui va déterminer les possibilités dœanalyse de lœentrepôt.

Ainsi, de nombreux travaux de recherche ont été menés sur la conception de schema des entrepôts de données. Ces travaux témoignent aujourdœhui de la nécessite deprendre en compte a la fois les sources de donnes et les besoins dœanalyse , plutôt que dœavoir recours, par exemple, a une approche uniquement guidée par les données telle que celle proposée par [3.17] [3.18]

Personnalisation dans les entrepôts de données

Dédiés aux applications d'analyse et de prise de décision, les entrepôts de données sont souvent interrogés par des requêtes complexes caractérisées par des opérations de sélections, de jointures et d'agrégations. Exécuter de telles requêtes sur un entrepôt volumineux nécessite un temps de réponse très élevé et abouti à des résultats souvent impertinents, ce qui est inacceptable par les décideurs qui exigent un temps de réponse raisonnable et des résultats précis selon leurs attentes afin de répondre aux besoins décisionnels.

De plus, les travaux menés ces quinze dernières années concernant les entrepôts de données ont essentiellement porté sur des aspects techniques tels que les processus ETL (« Extract-Transform-Load ») (Vassiliadis et al., 2002 ; Trujillo et Luján-Mora, 2003), les techniques d'optimisation et de sélection des vues matérialisées (Widom, 1995 ; Kotidis et Roussopoulos, 1999). Bien que les architectures décisionnelles soient considérées centrées utilisateurs, la prise en compte de ces derniers dans les systèmes décisionnels a finalement été peu étudiée.

Pour remédier à cette situation et satisfaire au mieux les attentes et besoins des décideurs, la personnalisation a été introduite dans les entrepôts de données après avoir fait un tram plein dans le domaine de la recherche d'information ainsi que les bases de données.

Il existe cependant différentes manières de personnaliser un système, le plus fréquemment on se munit d'un profil utilisateur qui souvent en dit long sur son propriétaire, dans notre cas nous parlerons de la réécriture de requêtes, c'est-à-dire que la requête est reformulée de telle sorte à inclure le profil utilisateur pour mieux cibler ses attentes. Nous allons dans ce qui suit introduire les notions mathématiques utilisées dans le cadre général.

4.1. Concept mathématique de base :

Nous commençons par donner des rappels sur les notions de base :

Définition 4.1 (relation binaire) *: Une relation binaire R sur un ensemble E est un sous-ensemble du produit cartésien E × E, Si un couple (x, y) appartient à la relation R, on notera xRy au lieu de (x, y) ∈ R.*

Une relation binaire peut être:

- réflexive si pour tout $x \in E$, xRx
- irréflexive si pour tout $x \in E$, $\neg (xRx)$
- symétrique si pour tout $x, y \in E$, $xRy \Rightarrow yRx$
- antisymétrique si pour tout $x, y \in E$, xRy et $yRx \Rightarrow x = y$
- asymétrique si pour tout $x, y \in E$, $xRy \Rightarrow \neg (yRx)$

, y ? xRy ou yRx
Ry ou yRx
, xRy et yRz ? xRz

- négativement transitive si pour tout x, y, z ? E, ¬ (xRy) et ¬ (yRz) ? ¬ (xRz)

***Définition 4.2** (- **relation dãindifférence**). Étant donnée une relation R sur un ensemble E, la relation dãindifférence notée $\sim_R$ est définie pour tout x, y ? E par :*

x $\sim_R$ y si xRy ? yRx

On note que la relation dãindifférence $\sim R$ est symétrique. [4.1]

***Définition 4.3** (- **relation dãincomparabilité**). Étant donnée une relation R sur un ensemble E, la relation dãincomparabilité notée* Ɛ $_R$ *est définie pour tout x, y ? E par :*

x Ɛ $_R$ y si ¬ (xRy) ? ¬ (yRx)

On note que la relation dãincomparabilité Ɛ $_R$ est symétrique. [4.1]

***Définition 4.4** (- **relation de préférence**). Une relation de préférence sur un ensemble dãéléments E est une relation binaire.*
Dans ce cas, étant donnée une relation de préférence notée $>_R$:

$x >_R y$ signifie que x est strictement préféré à y.

$x \sim_R y$ signifie que x et y sont à préférences égales.

x Ɛ $_R$ y signifie quãil nãy a ni indifférence, ni préférence entre x et y et on dit que x et y sont incomparable.

***Définition 4.5**(- **relation dãordre**). On appelle relation dãordre sur un ensemble E toute relation binaire sur E qui est réflexive, antisymétrique et transitive. Si la relation dãordre est complète alors lãordre est dit total, sinon cãest un ordre partiel.*

A une relation dãordre on peut associer une relation obtenue en restreignant celle-ci aux couples dãéléments distincts.

***Définition 2.6** (- **relation dãordre strict**). Une relation dãordre strict sur un ensemble E est une relation binaire irréflexive et transitive.*
Lãordre strict est dit faible, sãil est partiel et négativement transitif.

Le passage dãune relation dãordre à une relation dãordre strict est toujours possible et on peut lãobtenir comme suit :

- pour une relation dãordre (?), on définit la relation dãordre strict associée (?) par :

pour tout x, y ? E, x ? y ? $x \neq y$ et x , y

- pour une relation dãordre strict (?), on définit la relation dãordre associée (?) par :

pour tout x, y ? E, $x \neq y$? x ? y ou $x = y$

…semble, on peut utiliser une fonction qui sera définie de …né tel que celui des réels [0, 1] par exemple.

Définition 4.7(- fonction dôutilité)*. Une fonction dôutilité u est une fonction définie de E vers* [0, 1] *: u : E* Ÿ [0, 1]

x Ÿ $u(x)$

On note que pour toute fonction dôutilité u, on peut associer une relation dôordre total $Ó_u$ *définie pour tout x, y* ? *E par : x* $Ó_u$ *y si u(x)* $Ó$ *u(y).*

On note aussi que dans cette représentation, deux éléments de même utilité sont considérés comme indifférents.

La fonction dôutilité permet ainsi dôattribuer des valeurs aux éléments dôun ensemble de telle sorte que les éléments les plus intéressants reçoivent des valeurs supérieures à ceux qui le sont moins. [4.1]

4.2. Préférences sur un ensemble d'objets

Des recherches récentes ont été entreprises dans le but de définir les relations de préférence comparant des ensembles dôéléments. Nous en citerons les deux approches suivantes:

La première approche consiste à passer dôune relation de préférence sur des éléments atomiques dôun ensemble à une relation de préférence sur des sous ensembles de cet ensemble. Plusieurs solutions ont été proposées à ce sujet. [4.2]

La seconde approche part des préférences exprimées sur les propriétés des sous ensembles, et on définit un ordre sur ces sous ensembles. L es préférences sur les propriétés peuvent être exprimées sous forme de préférences sur des tuples : préférer la cardinalité 3 à la cardinalité 2, par exemple.

4.3. Différents types de requêtes

Un langage de préférence est composé dôun ensemble dôexpressions de préférence. Chaque expression M permet de définir une relation de préférence ?$_M$ sur un ensemble *E*, e est dit préféré a é si : e?$_M$ é, avec e, é ? *E* .On distingue deux types de requêtes :

4. 3.1 Requêtes de comparaison : Permet de déterminer si un élément est préféré à un autre, On définit pour ce cas :

- Requête de dominance : Une requête de dominance vérifie si la relation e?$_M$ é est vrai, alors on dit que e domine é par rapport a M ,on écrit :

  dom(M,e,é) = vrai si M permet de déduire que e ?$_M$ é

e requête dôordre vérifie si é$_M$ *e*.Si cette relation est ordre de préférence dont lequel e M é si M nôest pas connexe .On écrit :

ord(M,e,é) = vrai si M ne permet pas de vérifier que é$_M$e donc e $_M$ é.

4.3.2 Requêtes d'optimisation : Permet de retrouver lôensemble des éléments préférés avec ou sans contraintes :

- Sans contraintes : Soit une expression de préférence M et X un sous ensemble dôéléments de *E*, une requête dôoptimisation détermine lôensemble X* des meilleurs éléments de X tel que :

  X*=opt(M,X)={e (é e$_M$ e)}

- Avec contraintes : Soit une contrainte qui est une fonction définie sur lôensemble E telle que :

  : *E* {true, false}

Soit une expression de préférence M et X un sous ensemble dôéléments de *E*, une requête dôoptimisation avec contrainte dé termine lôensemble X* des meilleurs de X qui satisfont la contrainte tel que : X*=opt(M,X)={e (é e)} c (e) = true signifie que e satisfait . [4.1]

4.4. Langage de préférence

Les données manipulées sont représentées par un type de structure appelé *relation* .

Autour de ce contexte on définit :

- *domaine :* ensembe de constantes défini en extension ou en intension.
- *produit cartésien* : le produit cartésien dôun ensemble de domaines D_1 ,D_2 ,é ,D_K est lôensemble des tuples <t_1,é ,t_k> tels que t_i D_i,i 1, .
- *relation :*sous-ensemble du produit cartésien *de k domaines (k*>0).Si on note *R* cette relation alors *R* D_1 ... D_k .Une relation est représentée par une table a deux dimensions.
- *tuple :* correspond à une ligne dôune table.
- *attribut :* correspond à une colonne dôune table.
- *instance dôune relation* : ensemble de tuples dôune relation qui existent dans la base de données a un instant donné.
- *schéma dôune relation :* le schéma dôune relation *R* est noté *R*( ) ou { A_1,é ,A_K} est un ensemble dôattributs. Il y a un domaine *dom*(*Ai*) pour chaque attribut *Ai*, *i* [1, *k*].

Définition 4. 8 - Relation de préférence sur une relation. *Une relation de préférence ≻R sur une relation de schéma R*(A) *est une relation binaire sur* $x^{k}_{i=1}$ *dom*(*Ai*).

Définition 4.9 (**- Fonction d'utilité sur une relation**). *Étant donnée une relation de schéma R*(A), *une fonction d'utilité u sur R*(A) *est une fonction définie de* $x^{k}_{i=1}$ *dom*(*Ai*) *vers* [0, 1].

4 .5 .Critères d'étude des langages de préférence

- *type d'approche de modélisation* : permet la distinction entre les deux type de langage de préférence, à savoir le type quantitatif(ou la représentation des préférences utilisateur est très précise , les préférences sont exprimées par des fonctions qui attribuent des valeurs a chaque élément)et le type qualitatif(ou la représentation des préférences utilisateur est plus ou moins générale ,les préférences sont exprimées par des relations de préférence).

- *préférences utilisateur* : détermine sur quels éléments se dirigent les préférences exprimées par l'utilisateur.

- *type de relation de préférence* : précise la relation de préférence définie entre les éléments d'un entrepôt de données permettant de les comparer.

- *composition possible des préférences :* indique si on peut combiner les relations de préférence et quelle type de composition est possible .

- *est-ce que les préférences conditionnelles sont exprimées* : indique si un langage permet de modéliser une préférence dépendante d'une condition donnée.

Définition 4.10 (- formule de préférence) Soit *R* une relation de schéma *R*(≻) ou ≻ ≻ A_1,é ,A_K} est un ensemble d'attributs avec *dom*(≻) = *dom*(≻ $_1$≻ ... ≻ *dom*(≻ $_K$). Etant donnée deux tuples t_1,t_2≻ *dom*(≻), une formule de préférence notée $C(t_1,t_2)$,est une formule de premier-ordre définissant une relation de préférence $>_C$ comme suit :

$$t_1 >_c t_2 \equiv C(t_1,t_2)$$

Une formule de préférence de type *intrinsèque* établit une relation de préférence entre deux tuples uniquement sur la base des valeurs de ces tuples .Cette formule est de la forme :

$$C=D_1 \wedge \ldots \wedge D_m \, , \, m>0$$

njonction $D = Ci,1 \wedge \ldots \wedge C_{i,k}, i \in [1,m]$.

- chaque $C_{i,j}$ est une formule atomique de type *contraintes d'égalité(x=y,x≠y,x≠c ou x=c ou x et y sont des variables et c une constante)* ou de type *contraintes d'ordre rationnel (x≤y ou x≥c ou ∈ {=,≠,≤,≥,<,>}* x et y sont des variables et c un nombre rationnel)

Si $m = 1$ alors C est une formule de préférence conjonctive.

Les langages de préférence ont été introduits dans le but de permettre la représentation des préférences utilisateur .Ces préférences indiquent que certaines données sont plus pertinentes que d'autres. Chacun des langages cités précédemment spécifie une manière propre de définir les relations de préférence existantes entre les différents éléments d'une base de données. On a vu qu'une relation de préférence peut être une relation binaire, un ordre total, ou un ordre partiel. Cependant il n'est pas toujours possible de modéliser toutes les préférences avec un ordre total, car un utilisateur peut ne pas avoir de préférence entre deux éléments, mais les préférer tous deux a un tierce élément.

4.6. Processus de personnalisation

Afin de procurer à l'utilisateur l'information selon ses attentes, le système de personnalisation s'appuie sur un langage de préférence .Ce langage va permettre la mise au point d'un algorithme exploitant les préférences utilisateur, et ainsi aboutir à une réponse personnalisée. Les préférences sont souvent représenter et stockées dans un profil propre à l'utilisateur.

- **Le profil utilisateur** Le profil utilisateur peut inclure : Les préférences sur le contenu de la réponse à la requête, les préférences sur la présentation de la réponse, et/ou les contraintes utilisateur.
- **Contenu** : Pour déterminer le contenu personnalisé, le profil doit porter sur la réponse à la requête.

- **La présentation :** La personnalisation de la présentation permet de construire une façon adaptée pour afficher le contenu de la réponse en faisant intervenir les informations du profil. Ainsi la réponse à une même requête peut varier en fonction des besoins des utilisateurs. C'est dans ce sens qu'a été introduite la *sélection et l'ordonnancement :* la sélection permet de déterminer les attributs utiles à afficher, et l'ordonnancement permet d'ordonner les attributs sélectionnés par ordre décroissant de leur utilité à l'affichage. [4.3]

- **Les conditions d'exploitation :** Ce sont les conditions liées au matériel utilisé et/ou les contraintes liées à l'utilisateur lui-même.

ues matérielles utilisés tel que la taille dôaffichage, la dôafficher la réponse dans un format cohérent avec les exemple, la prise en compte les contraintes suivantes: le coût dôexécution dôune requête doit être réduit au minimum et la taille de la réponse doit être comprise dans un intervalle donné. [4.4]

Les contraintes utilisateur sont des règles qui déterminent par exemple le type de représentation souhaitée.

4.7. Personnalisation après exécution :

Dans cette section, nous présentons comment personnaliser une instance de visualisation en transformant la réponse à une requête utilisateur. Étant donné un cube de données c_0, il sôagit de la personnalisation après lôaccès aux données du cube (ou aux faits). On veut trouver une instance personnalisée v_N de $v = qMDX(c_0)$ telle que :
$v_N = F_{<V,\varepsilon V}(v) = F_{<V,\varepsilon V}(qMDX(c_0))$
Lôinstance de visualisation personnalisée v_N est donc obtenue par lôapplication de lôopérateur $F_{<V,\varepsilon V}$ sur v selon les étapes suivantes :
ï tout dôabord, exécuter la requête utilisateur q_{MDX} sur lôinstance de cube de données c_0 pour obtenir un résultat v,
ï ensuite, appliquer lôopérateur de préférence $F_{<V,\varepsilon V}$ sur le résultat v pour sélectionner les faits qui vont intéresser lôutilisateur et obtenir le résultat v_N,
ï enfin, présenter à lôutilisateur le résultat v_N.
La figure 1.1 illustre les étapes ci-dessus de cette démarche de personnalisation et leur enchaînement.

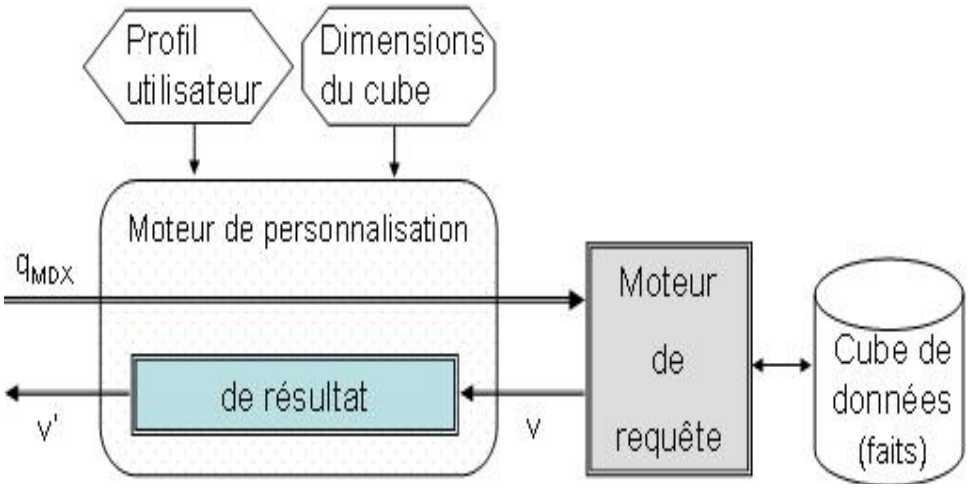


Fig.4.1 ï Étapes de la démarche de personnalisation après accès aux données de cube.

écution

ion comment personnaliser une instance de visualisation en transformant la requête utilisateur. Il sôagit de la personnalisation avant lôaccès aux données du cube (ou aux faits). Étant données une instance de cube *c*0 et une requête *qMDX*, on veut trouver une requête personnalisée q_{MDX} telle que :

$v = q_{MDX(c0)} = F_{<V,q(qMDX(c0))}$

Une fois que la requête personnalisée q_{MDX} est exécutée sur lôinstance de cube *c*0, le résultat obtenu est une instance de visualisation personnalisée *v*. Les étapes suivies sont les suivantes :

- tout dôabord, trouver une nouvelle requête q_{MDX} pour remplacer la requête utilisateur q_{MDX},
- ensuite, exécuter la nouvelle requête q_{MDX} sur lôinstance de cube de données c_0 pour obtenir un résultat *v*contenant les faits qui vont intéresser lôutilisateur,
- enfin, présenter à lôutilisateur le résultat *v*

La figure 1.2 illustre les étapes décrites ci-dessus de cette démarche de personnalisation. [4.1]

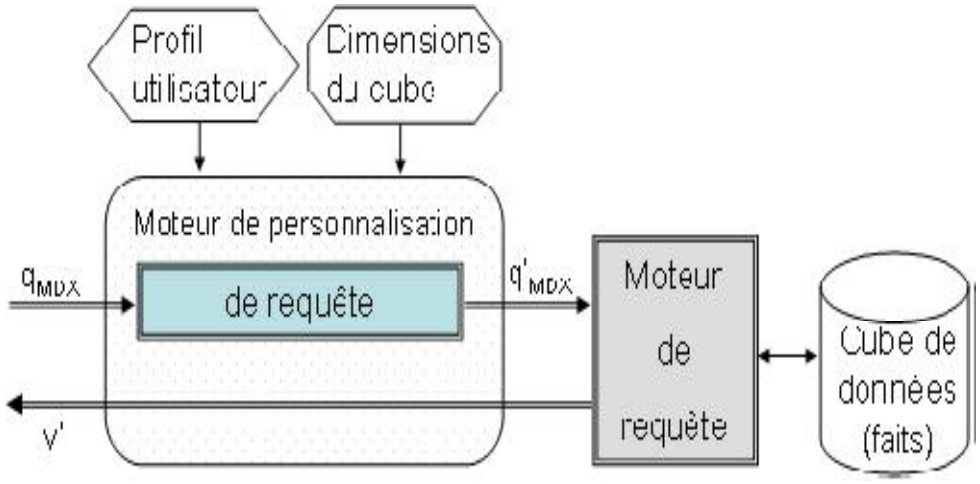


Fig.4.2 - Étapes de la démarche de personnalisation avant accès aux données de cube.

4.9. Travaux de personnalisation réalisés sur les entrepôts de données :

La personnalisation dans les entrepôts de données peut être vue sous deux angles :

(i) Personnalisation de schéma ou bien de lôévolution du schéma de lôentrepôt [Favre, 2007] ou

(ii) Personnalisation de la navigation [Ravat et Teste 2008].

(iii) Personnalisation de la visualisation du résultat dôune requête OLAP [Bellatreche et al, 2006].

Lôobjectif de notre travail sôinscrit dans le troisième type de personnalisation, en effet,

idation ainsi que sur la personnalisation de la navigation
lant une présentation détaillée sera réservée au dernier

4.9. 1. Travaux de [Favre 2007] :

Afin d'enrichir les possibilités d'analyse d'un entrepôt de données, une proposition permettant la création de niveaux d'analyse supplémentaires dans les hiérarchies de dimension ou définissant de nouvelles hiérarchies de dimension a été faite dans [Favre, 2007]. Cette approche permet aux utilisateurs d'intégrer leurs propres connaissances sur la façon d'agréger les données sous la forme de règles de type « si-alors ». Une évolution du modèle de l'entrepôt permet le partage des chemins d'agrégation nouvellement créés entre les utilisateurs du système. Supposons qu'un utilisateur veuille analyser les ventes du produit « *food* », non pas par « *year* » ni par « *day* » et non plus par « *month* », mais par semaine, information qui n'est pas présente dans l'entrepôt. L'utilisateur peut alors exprimer ses connaissances sur les types d'agence, afin de créer un niveau « *semaine* », correspondant à une nouvelle hiérarchie de la dimension « *Time* ». L'utilisateur pourra ainsi réaliser des analyses des ventes du produit « *food* » par semaine.

4.9. 2. Travaux de [Giacometti et al, 2008] :

Une session d'analyse OLAP peut être définie comme une session interactive durant laquelle un utilisateur lance une requête pour naviguer au sein du cube. Très souvent, le choix de la partie du cube à explorer et donc la requête correspondante à cette partie est une tâche difficile. [Giacometti et al, 2008] Proposent un système de recommandation d'analyses multidimensionnelles en se basant sur la navigation qu'effectue un utilisateur donné par rapport aux navigations réalisées par les autres utilisateurs.
La proposition de Giacometti et al. (2008) permet la recommandation de requêtes pour anticiper sur une séquence de requêtes d'un utilisateur grâce à l'analyse des historiques de navigations réalisées par les autres utilisateurs. Par exemple, supposons qu'un utilisateur a réalisé une analyse des ventes par année et par country, puis avec un forage vers le bas par année et par région, et enfin par année (avec un second forage vers le bas). Si un nouvel utilisateur réalise une analyse des ventes par année et par country, puis une analyse par année et par région, une analyse par année et par city lui sera recommandée, sa navigation étant similaire à une navigation réalisée précédemment.

4.9. 3. Travaux de [Ravat et Teste 2008]

Ravat et Teste (2008) proposent une solution pour la personnalisation de la navigation OLAP en exploitant des préférences exprimées par des poids sur les éléments du schéma de la base de données multidimensionnelles, des poids reflétant l'intérêt de l'utilisateur. Dans ce cas, l'utilisateur assigne des poids aux concepts multidimensionnels afin d'obtenir directement les analyses désirées, évitant ainsi des opérations de navigation.Un système basé sur des règles ECA (Event, Condition, Action) permet de générer des tables multidimensionnelles contenant uniquement les données identifiées comme pertinentes en fonction des poids. Cette solution quantitative permet de simplifier l'expression des requêtes d'analyse. Pour aller au-delà de cette approche quantitative, une solution de personnalisation qualitative est introduite par [Jerbi et al,.2008].

]

s poids, mais plutôt des ordres (représentation qualitative des préférences), ce qui rend la tâche plus aisée pour lôutilisateur. En outre, ces ordres ne sont pas exprimés de façon absolue, mais par rapport à un contexte dôanalyse donné. Ceci permet de prendre en compte le fait que les préférences peuvent varier dôun contexte dôanalyse à un autre. [Jerbi et al,.2009] poursuivent ces travaux en présentant un environnement OLAP intégrant des mécanismes de recommandation contextuelle des requêtes.

4.9. 5. Travaux de [Bellatreche et al, 2005].

Ces travaux se concentrent sur la personnalisation des réponses aux requêtes utilisateur interrogeant les données du cube, c.-à-d. des visualisations. Dans cette approche, la visualisation est sous forme dôune table croisée.
Lôobjectif de ces travaux est de pouvoir fournir à lôutilisateur un résultat focalisé sur son centre dôintérêt, tout en prenant en compte des contraintes de visualisation.

4 .10.Principaux axes de recherche :

L'étude précédente montre que les travaux qui étudient la prise en compte des utilisateurs au sein des entrepôts de données sont en pleine émergence. Les travaux proposés jusque là ont consisté principalement à étendre les approches habituelles en permettant de modifier les structures multidimensionnelles et/ou les mécanismes de présentation des données par une meilleure connaissance des utilisateurs. Pour ce faire, ces travaux exigent des efforts cognitifs de la part de l'utilisateur qui doit souvent exprimer de manière explicite les préférences qui le caractérisent.
La prise en compte de l'utilisateur est une problématique nouvelle qui pose plusieurs enjeux peu ou pas étudiés. Une plus grande interaction de l'usager avec le système permettrait d'envisager des bénéfices à deux niveaux :

- du point de vue système, la connaissance accrue de l'utilisateur ou du groupe d'utilisateurs doit pouvoir servir à mieux paramétrer celui-ci, et par conséquent, doit permettre un fonctionnement plus proche des utilisateurs ;

- du point de vue utilisateur, un système mieux adapté doit permettre une réduction des efforts nécessaires pour accéder, manipuler et structurer une information pertinente afin de faciliter davantage le processus décisionnel qui en découle. Cela doit permettre au décideur de se centrer sur le processus d'analyse décisionnelle en lui simplifiant la manipulation des données.

Plus précisément, nous traçons cinq axes majeurs de recherche :

Un premier axe concerne la construction de profils utilisateurs dans les entrepôts de données. Cet aspect a été largement étudié, notamment en RI, mais, à l'heure actuelle, aucune solution unanimement reconnue ne semble émerger (Bouzeghoub et Kostadinov, 2005).
L'étude des travaux a montré qu'un premier point intéressant est la définition de mécanismes rendant la collecte des caractéristiques utilisateur automatique et transparente à ce dernier. Cette approche vise à définir un entrepôt de données adaptatif.

essaire de définir des solutions permettant l'évolution
s caractéristiques de ces derniers sont constituées
ateur sur les données et les structures de l'entrepôt. Un aspect original pour la constitution du contenu d'un profil serait d'ajouter l'aspect navigation opérée par l'utilisateur sur les données à l'image de la navigation Internet. Une meilleure connaissance des navigations OLAP sur les données peut par exemple servir à l'entrepôt de données pour préparer des données (agrégations, ...) qui anticipent la demande de l'utilisateur. Plus généralement, la définition d'un profil englobant les caractéristiques complètes couvrant l'ensemble des besoins de présentation et d'interaction peut être envisagé. Un dernier point, qui n'a fait l'objet que d'un seul travail dans le cadre des entrepôts de données alors qu'il est largement exploité d'ores et déjà en RI, est la prise en compte de l'aspect social dans les entrepôts en tenant compte du groupe d'utilisateurs auquel appartient un décideur. Dans ce contexte, l'apport du web social peut présenter un réel intérêt dans une perspective de personnalisation sociale des entrepôts de données. Les travaux proposés par Morfonios et Koutrika (2008), ainsi que Aouiche et al. (2008) constituent des pistes intéressantes. Ainsi, la définition d'un entrepôt de données proposant des recommandations pourrait servir à aider un décideur en tirant bénéfice de l'expertise des autres membres du groupe d'utilisateurs.
Cet axe couvre des problématiques récurrentes dans les systèmes d'informations et se heurte à la définition d'une (des) mesure (s) de similarité dans les entrepôts de données. Ce verrou scientifique pose en effet le problème de la constitution d'une mesure de similarité permettant de comparer et d'aligner les profils des utilisateurs ainsi que les contextes d'analyse formés de sujets et d'axes d'analyses.
Une des manières d'aborder ce problème de recommandation est également de se placer dans le contexte des travaux concernant les requêtes de type « top k » ou « skyline » (Brando et al., 2007). Ces travaux traités dans le contexte des bases de données tentent également d'inclure un processus de personnalisation (Lee et al., 2009). Il serait intéressant d'étudier les requêtes OLAP et leurs éventuelles spécificités par rapport à cette problématique.
La navigation, qui est propre aux entrepôts de données, constitue également un enjeu par rapport à la construction de profil. La détermination de poids proposée par Ravat et al. (2008) pour personnaliser la navigation constitue une première approche de construction de profil. Mais il est vrai que la détermination de ces profils de façon explicite par l'utilisateur peut s'avérer fastidieuse. Un travail sur l'apprentissage de ces préférences en termes de navigation au fur et à mesure de la navigation de l'utilisateur pourrait alléger le processus, et par là même devenir un processus davantage évolutif. La proposition de Giacometti et al. (2008) pourrait constituer un point de départ intéressant pour cette réflexion.

Le **deuxième axe** concerne la gestion des données. L'analyse multidimensionnelle consiste à restituer des données agrégées en fonction de différents axes d'analyse. L'objectif de l'OLAP est une restitution rapide de ces données agrégées. La solution envisagée repose sur le principe de vue matérialisée. Si le profil enregistre les différentes navigations qu'effectue un décideur sur un cube de données, voire précise les composants des axes d'analyse souhaités, ceci aura un impact sur la définition des vues matérialisées.
Un des enjeux qui en découle est alors de pouvoir définir des algorithmes de sélection de vues à matérialiser avec prise en compte du profil des utilisateurs (ou groupe d'utilisateurs). Classiquement, les algorithmes de sélection se basent sur une charge de requêtes qui représente l'utilisation de l'entrepôt (Kotidis et Roussopoulos, 1999), et donc en quelque sorte les préférences des utilisateurs en matière de requêtes. Il s'agit d'aller au-delà, en prenant également en compte l'aspect enchaînement des requêtes. Ceci pourrait avoir non seulement un impact sur le choix des vues à matérialiser, mais également sur l'usage de ces vues. Il s'agit

araison du treillis de vues matérialisables et de l'arbre de

ormances, on peut imaginer qu'en connaissant mieux la navigation des utilisateurs, il soit possible, dans un contexte d'une architecture client serveur, que le système soit pro-actif dans cette gestion. Par exemple, grâce à la connaissance sur la navigation des utilisateurs, il est envisageable de charger certaines vues en mémoire pour anticiper l'usage de l'utilisateur en s'inspirant des travaux de Cherniack et al. (2003).
Un autre aspect spécifique aux entrepôts de données concerne la personnalisation de la phase d'ETL (Extract, Transform and Load). Une perspective envisageable serait d'étudier l'apport de la personnalisation dans le cadre de ce processus. Par exemple, une meilleure connaissance des profils utilisateurs pourrait permettre une adaptation de la stratégie de rafraîchissement afin de déterminer quelles données rafraîchir à quels moments en fonction des utilisateurs, de leur usage des données.

Un **troisième axe** porte sur l'influence de la personnalisation dans la modélisation multidimensionnelle.
Des travaux ont étudié cet aspect en proposant d'enrichir et de transformer
les hiérarchies du modèle en tenant compte de spécificités exprimées par l'utilisateur (Favre et al., 2007). Ces travaux peuvent être complétés notamment par la génération de hiérarchies basée sur la collecte automatique des caractéristiques de l'utilisateur. Ces nouvelles hiérarchies peuvent être proposées à l'utilisateur dans un système de recommandation. Une autre voie concerne le couplage de techniques de fouille de données et de la modélisation multidimensionnelle pour inférer des hiérarchies à recommander (Bentayeb, 2008). Il est intéressant de pouvoir exploiter les connaissances cachées dans les données, ce qui constitue un des objectifs de la fouille de données. Les techniques d'apprentissage non supervisé peuvent trouver dans ce contexte un intérêt. Le couplage avec la fouille de données constitue un axe prometteur, car cet apprentissage peut concerner les données elles-mêmes, mais également F. Bentayeb et al. l'utilisateur si l'on applique des techniques d'apprentissage sur les usages individuels qu'il fait du système.

Un **quatrième axe** d'étude repose sur la définition de structures de visualisation alternatives à la visualisation traditionnelle sous forme de table à deux dimensions (Gyssens et Lakshmanan, 1997). La présentation des données est un point important car les utilisateurs réclament des moyens d'accès facilitant l'analyse des données. Différents travaux proposent de nouvelles formes de visualisation (Mansmann et Scholl, 2007 ; Choong et al., 2006 ; Sifer, 2003 ; Stolte et al., 2002), mais au-delà des structures de visualisation, des travaux relatifs au résumé de données tels que ceux proposés par Choong et al. (2008) semblent prometteurs. La problématique est alors d'inclure le processus de personnalisation par rapport à la visualisation et au résumé d'informations. Outre la définition de différentes structures de visualisation, une problématique nouvelle est l'adaptation de la structure de présentation des données en fonction des données analysées. Une première approche existante consistait à réorganiser les modalités dans les dimensions selon les ordonnancements fournis par l'Analyse en Correspondances Multiples (ACM). Grâce à cette réorganisation, les modalités d'une dimension sont agrégées selon l'ordre de leur proximité et non selon l'ordre de leur appartenance hiérarchique établi (Ben Messaoud et al., 2007). L'objectif sera alors de prendre en compte, dans la définition et l'utilisation de structures de visualisation, des caractéristiques propres à l'utilisateur, par exemple, son niveau d'expertise.

est transversal par rapport aux axes cités précédemment. nes relatifs à la sécurité. Il est possible de considérer que ns certains contextes, peuvent reposer sur des concepts similaires, celui de profil entre autres. La principale différence est alors la suivante : dans un cas lôutilisateur préfère ou non accéder à telles informations, et dans lôautre lôutilisateur a le droit ou non dôy accéder. Selon la solution de personnalisation proposée, celle-ci peut se baser sur des données plus ou moins « personnelles ». Se pose alors le problème de la protection des données servant à la personnalisation. Ce problème a été évoqué dans le contexte du Web (Wang et Kobsa, 2007). Il peut également se poser dans le contexte des entrepôts de données au niveau de lôaccès sécurisé aux données (dont les stratégies présentent des particularités en entrepôt de données, vis-à-vis des niveaux de granularité de lôinformation).

4.11. Conclusion

Il est désormais évident que la personnalisation a entièrement sa place dans le monde actuel, y compris les domaines du décisionnel, a savoir lôentrepôt de données, pas beaucoup mais certains travaux se sont orientés dans ce sens, prenant des chemin différent mais toujours avec le même esprit qui de cerner lôutilisateur pour mieux le servir.

Dans notre cas nous nous sommes inspirés de certains dôentre eux, espérant atteindre une analyse centrée utilisateur, en incluant son profil lors de son interaction avec lôentrepôt de données. Ce profil renfermera des informations sur ce dernier incluant ses préférences et centres dôintérêt. Le chapitre qui suit portera sur la réalisation du système que nous avons tenter de personnaliser.

Choix des outils technologiques

La mise en î uvre d'un système informatique efficace ne dépend pas seulement de sa conception mais aussi du choix des outils de développement. Ainsi, nous avons porté notre dévolu sur les outils, qui nous paraissaient les plus appropriés. Hormis leur prépondérance, nous allons de ce chapitre vous expliquer les autres raisons qui ont motivé nos choix.

5.1. Les langages de programmation

5.1.1 Le java :

Java a acquis une immense popularité depuis son apparition vers les années 91, avec plus de 6,5 millions de développeurs convaincus, il est aujourdôhui le langage le plus important et le plus actif a travers le monde. Ses nombreux avantages ont eu raison de son ascension fulgurante, parmi ces atouts on citera :

- facile à utiliser et apprendre orienté objet, ce qui permet de créer des programmes modulaires et code réutilisable.
- indépendant vis-à-vis de la plate-forme, qui consiste en sa capacité à exécuter le même programme sur de nombreux systèmes différents
- distribué, java a été conçu pour rendre l'informatique répartie facile avec la fonctionnalité de réseau qui est intrinsèquement intégrée
- multi-thread, ce qui consiste en la possibilité dôexécuter plusieurs tâches simultanément dans un programme.
- sécurisé, Java considère la sécurité dans le cadre de son design. Le langage Java, le compilateur, lôinterprète, et l'environnement d'exécution ont chacun été développés avec la sécurité à l'esprit
- robuste et fiable, Java met beaucoup d'accent sur la vérification rapide pour d'éventuelles erreurs, que des compilateurs Java sont en mesure de détecter.

Contrairement à la plupart des autres langages, Java met à la disposition du développeur une API très riche lui permettant de faire de un bon nombre de choses. Java propose à peu près tout ce dont on peut avoir besoin directement dans le JDK. Ce qui est un énorme avantage, qui augmente grandement la productivité de développement. [5.1]

5.1.2.Les servlets :

La technologie des servlets Java est une technologie introduite par Sun jouant un rôle symétrique à celui des applets, mais côté serveur. Si les applets Java ont pour but d'enrichir les fonctionnalités d'un navigateur client, les servlets permettent d'étendre les fonctionnalités d'un serveur Web.

omatiquement lors du démarrage du serveur Web ou lors
e fois chargées, les servlets restent actives dans l'attente
ermettent de gérer des requêtes HTTP et de fournir au client une réponse HTTP dynamique.

Les servlets possèdent nombreux **atouts** par rapport aux autres technologies côté serveur. Etant donné qu'il s'agit d'une technologie Java, les servlets fournissent un moyen d'améliorer les serveurs web sur n'importe quelle plateforme, d'autant plus que les servlets sont indépendantes du serveur web. De plus, l'utilisation de servlets se fait par le biais d'un conteneur de servlets (framework), qui constitue l'environnement d'exécution de la servlet et lui permet de persister entre les requêtes des clients, il fournit également tout un ensemble de services standards pour simplifier la gestion des requêtes et des sessions. Ainsi le programmeur n'a pas à se soucier de détails techniques tels que la connexion au réseau, la mise en forme de la réponse HTTP, é [5.2]

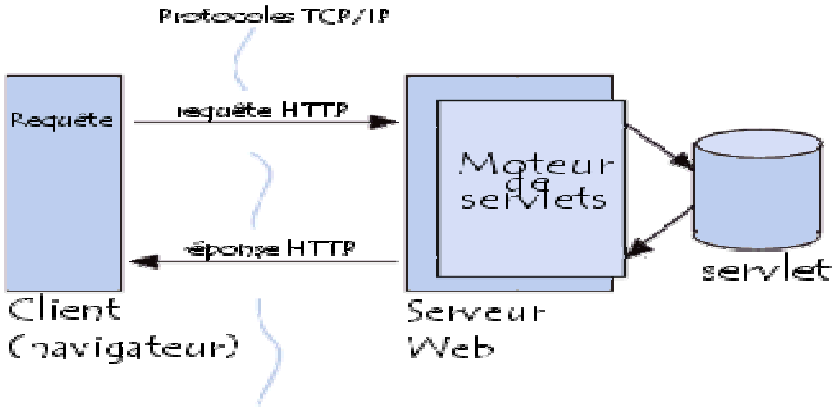


Fig.5.1:Schema dœxecution dœune servlet

5.1.2.Le cycle de vie d'une servlet

Le cycle de vie d'une servlet est assuré par le conteneur de servlet. Ainsi afin d'être à même de fournir la requête à la servlet, récupérer la réponse ou bien tout simplement démarrer/arrêter la servlet, cette dernière doit possèder une interface (un ensemble de méthodes prédéfinies) déterminée par le JSDK afin de suivre le cycle de vie suivant :

- le serveur crée un pool de threads auxquels il va pouvoir affecter chaque requête
- La servlet est chargée au démarrage du serveur ou lors de la première requête
- La servlet est instanciée par le serveur
- La méthode *init()* est invoquée par le conteneur
- Lors de la première requête, le conteneur crée les objets Request et Response spécifiques à la requête

pelée à chaque requête dans une nouvelle thread. Les
sont passés en paramètre
éthode *service()* va pouvoir analyser les informations en provenance du client

- Grâce à l'objet *Response*, la méthode *service()* va fournir une réponse au client

La méthode *destroy()* est appelée lors du déchargement de la servlet, c'est-à-dire lorsqu'elle n'est plus requise par le serveur. La servlet est alors signalée au *garbage collector* (communément francisé en *ramasse-miette*). [5.3]

5.1.3.Les JavaServer Pages :

Les JavaServer Pages, dites les JSP sont une technologie développée par Sun permettant d'inclure dans des templates HTML des programmes Java exécutés lors de l'appel du template et dont les résultats sont formatés et intégrés au code HTML en retour. La technologie JSP s'apparente très fort à celle des ASP, mais aussi au PHP, avec lesquels elle partage le point commun de se baser sur les templates HTML.

En utilisant les JSP, il est possible de développer un site Web dynamique, s'appuyant sur des applications tierces sur le serveur, en séparant la mise en forme du contenu. Cette séparation est due au fait que les pages sont générées activement sur le serveur avant d'être envoyées, finalisées, vers le navigateur client. L'interaction avec des applications tierces, bases de données notamment, permet de gérer un volume de données très important et d'adapter les résultats en fonction des visiteurs.

Les JSP sont donc une technique d'intégration de code Java, appelé scriptlet, ou d'appels de code Java à des pages HTML. L'avantage de développer un site à l'aide des JSP est qu'il n'est plus nécessaire de retoucher le code HTML pour formater les éléments de mise en page en fonction des données: la page est générée activement depuis le serveur.

Les avantages des JSP

Aujourd'hui, les JSP et les outils associés (Java Beans, servlets,librairie de balises ...) sont intégrés dans l'architecture J2EE qui est une spécification du langage Java et de l'architecture pour serveurs d'applications associée. Cette architecture et l'utilisation des JSP offre dès lors de nombreux avantages et un environnement de développement très performant. Au nombre des qualités des JSP, on peut souligner les suivantes:

- Une très bonne intégration du code Java et du HTML, minimisant par conséquent les problèmes de mise en page. Ceci permet surtout de développer des pages Web dynamiques s'appuyant sur un grand volume de données (par exemple provenant d'un SGBD) et dont le contenu est adaptable au caractère évolutif de ces données et aux requêtes particulières des utilsateurs.

Les JSP reposent sur un langage de programmation orienté Web puissant, Java, et des technologies côté serveur comme les servlets ayant fait leurs preuves. Par ailleurs,

ur sa très bonne portabilité, au contraire des ASP de

5.1.4.Java Script :

Java script est un langage de script incorporé dans un document HTML. Il permet dœffectuer des contrôles de saisie pour valider les champs dûun formulaire, dœouvrir ou de fermer de nouvelles fenêtres, ou encor de gérer des éléments graphiques.

5.2. Environnement de développement (Eclipse)

Eclipse est un environnement de développement intégré (Integrated Development Environment) dont le but est de fournir une plate-forme modulaire pour permettre de réaliser des développements informatiques. I.B.M. est à l'origine du développement d'Eclipse qui est d'ailleurs toujours le cî ur de son outil Websphere Studio Workbench (WSW), lui même à la base de la famille des derniers outils de développement en Java d'I.B.M. Tout le code d'Eclipse a été donné à la communauté par I.B.M afin de poursuivre son développement. [5.5]

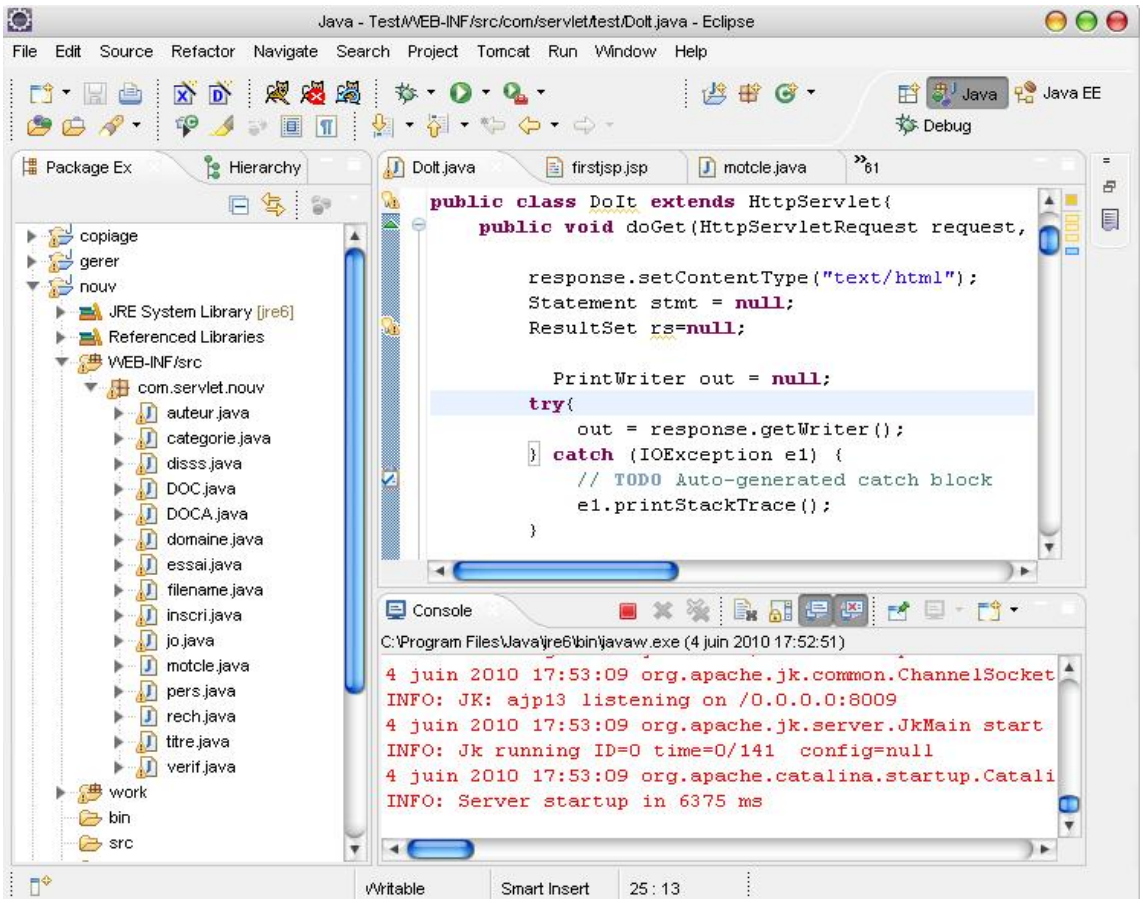


Fig.5.2. Interface eclipse

x points forts qui sont à l'origine de son énorme succès dont les principaux sont :

- Une plate-forme ouverte pour le développement d'applications et extensible grâce à un mécanisme de plug-ins
- Plusieurs versions d'un même plug-in peuvent cohabiter sur une même plate-forme.
- Un support multi langage grâce à des plug-ins dédiés : Cobol, C, PHP, C# , ...
- Support de plusieurs plate-formes d'exécution : Windows, Linux, Mac OS X, ...
- Malgré son écriture en Java, Eclipse est très rapide à l'exécution grâce à l'utilisation de la bibliothèque SWT
- Les nombreuses fonctionnalités de développement proposées par le JDT (refactoring très puissant, complétion de code, nombreux assistants, ...)
- Une ergonomie entièrement configurable qui propose selon les activités à réaliser différentes « perspectives »
- Un historique local des dernière modifications
- La construction incrémentale des projets Java grâce à son propre compilateur qui permet en plus de compiler le code même avec des erreurs, de générer des messages d'erreurs personnalisés, de sélectionner la cible (java 1.3 ou 1.4) et de mettre en oeuvre le scrapbook (permet des tests de code à la volée)
- Une exécution des applications dans une JVM dédiée sélectionnable avec possibilité d'utiliser un débogueur complet (points d'arrêts conditionnels, visualiser et modifier des variables, évaluation d'expression dans le contexte d'exécution, changement du code à chaud avec l'utilisation d'une JVM 1.4, ...)
- Propose le nécessaire pour développer de nouveaux plug-ins
- Possibilité d'utiliser des outils open source : CVS, Ant, Junit
- La plate-forme est entièrement internationalisée dans une dizaine de langue sous la forme d'un plug-in téléchargeable séparément
- Le gestionnaire de mise à jour permet de télécharger de nouveaux plug-ins ou nouvelles versions d'un plug-in déjà installées à partir de sites web dédiés (Eclipse 2.0).

Pour notre application, nous avons choisit eclipse Galileo ,version plate-forme 3.5,apparu en juin 2009 .

5.2.1 Les plug-in rajoutés à Eclispe :

Eclipse utilise énormément le concept de modules nommés "plug-ins" dans son architecture. D'ailleurs, hormis le noyau de la plate-forme nommé "Runtime", tout le reste de la plate-forme est développé sous la forme de plug-ins. Ce concept permet de fournir un mécanisme pour l'extension de la plate-forme et ainsi fournir la possiblité à des tiers de développer des fonctionnalités qui ne sont pas fournies en standard par Eclipse.

Le plug-in Tomcat de sysdeo : est un plug-in proposé par la société Sysdeo pour faciliter le développement d'applications utilisant Tomcat notamment en permettant le démarrage et l'arrêt de Tomcat, le packaging d'une application sous la forme d'un fichier .war et son déploiement dans Tomcat.

Le plug-in XML Buddy : est un Plug-in Eclipse permettant d'éditer des fichiers XML et DTD

permettant à Eclipse dîntegrer le serveur mondrian.

5 .2.2. Les Drivers rajoutés à Eclispe :

Appelés JDBC (Java DataBase Connectivity) , ces drivers sont des interfaces de programmation qui permettent au langage Java d'accéder à des bases de données par l'intermédiaire du langage SQL. Comme des interpréteurs Java sont disponibles sur la plupart des postes de travail, cela permet d'écrire des applications indépendantes des bases de données. [5.6]

5. 3 .Les serveurs utilisés:

5.3.1.Le serveur Apache :

Le serveur HTTP Apache a été conçu comme un serveur web puissant et flexible pouvant fonctionner sur une très grande variété de plateformes et toute une gamme d'environnements différents. Plateformes différentes et environnements différents signifient souvent fonctionnalités différentes, ou utilisation de différentes méthodes pour implémenter la même fonctionnalité le plus efficacement possible. Apache s'est toujours accomodé d'une grande variété d'environnements grâce à sa conception modulaire. Cette conception autorise le webmaster à choisir quelles fonctionnalités seront incluses dans le serveur en sélectionnant les modules à charger soit à la compilation, soit à l'exécution. Tout ceci justifie la domination des serveurs Apache sur tout autre serveur. [5.7]

FIG.5.3.Logo Apache

Les avantages d'apache :

Si le logiciel serveur Apache connaît un tel succès aujourd'hui, c'est parce qu'il sécurisé et permet de mettre en î uvre de nombreuses applications, outre lôavantage d'être gratuit.

Apache possède dans le domaine des performances une excellente réputation ; cela dit, il est important de toujours veiller à ce qu'il fonctionne toujours de manière optimale. Voici donc les actions possibles en matière de performance sous Apache :

- contrôle des processus Apache sous Unix ou Windows
- limitation des requêtes HTTP
- gestion du cache
- limitation de la bande passante ;

Apache peut également faire office de proxy intermédiaire, que l'on nomme *proxy inverse*. Par ce biais, il est possible d'interdire le relayage des requêtes vers certains serveurs distants, ou même de gérer plusieurs serveurs web à partir d'une seule adresse (serveurs en grappe). Il est également possible d'acheminer d'autres protocoles que HTTP (telnet, news, etc.). Cela dit, il est parfois plus approprié de faire appel à un proxy dédié (Squid par exemple) pour ce genre de tâche. [5.8] [5.9] [5.10]

5.3.2. Le Conteneur de servlet Tomcat :

Apache Tomcat est un conteneur libre de servlet Java 2 Enterprise Edition. Issu du projet Jakarta, Tomcat est un projet principal de la fondation Apache. Il implémente les spécifications des servlets et des JSP du Java Community Process . Il est paramétrable par des fichiers XML et de propriétés, et inclut des outils pour la configuration et la gestion. Il comporte également un serveur HTTP.

Tomcat a été écrit en langage Java, il peut donc s'exécuter via la JVM (machine virtuelle java) sur n'importe quel système d'exploitation la supportant.

Tomcat est souvent utilisé en association avec un autre serveur web, en général Apache. Apache s'occupe de toutes les pages web traditionnelles, et Tomcat uniquement des pages d'une application web Java.

On peut utiliser le module mod_jk pour paramétrer la communication entre Apache et Tomcat. Techniquement, Apache communique avec Tomcat sur le port 8009 (via le protocole ajp13), mais Tomcat peut aussi être atteint via son propre port (8080 par défaut). [5.7]

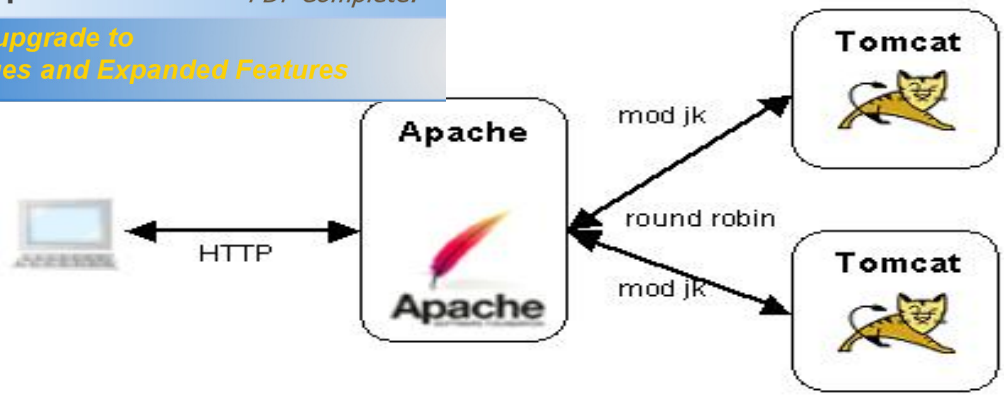


Fig.5.4.Fonctionnement Tomcat

5.3.3.Le serveur mondrian :

La solution MONDRIAN est développée en Open Source depuis 2001 sous licence CPL et se présente sous la forme d'un ensemble de librairies Java. Elles permettent la représentation de données relationnelles dans un cube ROLAP, grâce à un fichier XML décrivant les axes de mesures et les dimensions.

Fig.5.5 :Logo Mandrian Olap Server

Mondrian est un moteur OLAP (Online Analytical Processing) écrit en Java par Julian Hyde qui permet la conception, la publication et le requêtage de cubes multidimensionnels. Il permet lœexécution de requêtes en langage MDX sur des entrepôts de données sœappuyant sur des SGBDR, dœoù sa caractérisation de « ROLAP » (Relational OLAP). En matière de ROLAP, Mondrian est la référence open source.

Mondrian permet dœaccéder aux résultats dans un format multidimensionnel compréhensible par une API de présentation côté client, le plus souvent en mode Web, avec par exemple JPivot, Pentaho Analyzer, Pentaho Analysis Tool, Geo Analysis Tool (G.A.T.)

Mondrian sœappuie sur une modélisation OLAP standard et peut donc se connecter à nœimporte quel entrepôt de données conçu dans les règles de lœart de la Business Intelligence.Il est intéressant de noter que Mondrian est le composant OLAP utilisé par la plupart des suites de BI Open Source notamment Pentaho, JasperServer et SpagoBI.

t les suivantes :

- Stockage des données dans un SGBDR: les données sont entreposées dans des tables de faits et de dimension, selon la modélisation habituelle en étoile et/ou flocons (Mondrian sait gérer les 2)

- Mondrian permet (si nécessaire) l'utilisation de tables d'agrégation afin d'optimiser au mieux les performances. Ces tables doivent être chargées en même temps que l'entrepôt de données, via les mécanismes habituels d'alimentation ETL. Il est également possible de s'appuyer sur des « vues matérialisées » si le SGBD les supporte. Le serveur Mondrian fournit également un mécanisme de mise en cache qui permet d'obtenir des délais de réponse minimums.

- Utilisation des dimensions partagées (« shared dimension », ou « dimensions conformes ») pour la mutualisation de celles-ci entre plusieurs cubes

- Ajout de membres calculés (très utile pour effectuer des calculs de pourcentage par exemple, recalculés dynamiquement selon le niveau d'agrégation choisi)

- Gestion des hiérarchies multiples, dimensions dégénérées et « inline tables »

- Gestion de la sécurité d'accès aux cubes via des rôles: sécurisation au niveau du cube, des dimensions, des hiérarchies et des membres à afficher ou non

- Internationalisation possible (multilinguisme) via l'implémentation du standard i18n

Mondrian s'appuie sur des schémas XML pour la définition des cubes. Un schéma Mondrian permet donc de définir le modèle logique ainsi que le mapping sur le modèle physique :

- Le modèle logique décrit les cubes, les dimensions, les hiérarchies, les niveaux et les membres (et plus encoreé) sur lesquels vont s'appuyer les requêtes MDX.

Le modèle physique correspond à la source de données sur laquelle s'appuie le modèle logique (le modèle en étoile et/ou flocon). [5.11]

5.4. Le SGBD Oracle:

Oracle est un système de gestion de base de données (SGBD) relationnel fourni par Oracle Corporation et couramment utilisé dans les applications sur différentes plate-formes. Il s'agit du SGBD le plus vendu au monde (environ 25% du marché mondial) et c'est celui que nous utilisons à l'INT. En plus du noyau de SGBD proprement dit qui assure les fonctions de base (support de SQL, contrôle de concurrence, ...), Oracle est composé de nombreux outils comme Sql*Plus un interprète SQL, Pro*C l'interface embedded SQL pour le langage C, un langage de programmation pour les procédures stockées appelé PL/SQL, etc. Il existe également une version répartie, Oracle réparti. [5.12]

- Fonction d'audit évolué
- Row level storage security (RLSS) : permet de ne faire apparaître que certaines lignes des tables pour un utilisateur/une application donné.
- Intégration LDAP, SSL, Unicode; réplication intégrée; capable de mapper un fichier plat en table
- Parallélisme, caches nommés; haute disponibilité; grande possibilité de tuning
- Procédures stockés en PL-Sql (langage propriétaire Oracle, orienté ADA) ou ... en JAVA (depuis la 8.1.7) ce qui peut s'avérer utile pour les équipes de développement.
- Assistants performants via Oracle Manager Server, possibilité de gérer en interne des tâches et des alarmes
- Gestion centralisée de plusieurs instances
- Concept unique de retour arrière (Flashback)
- Pérennité de l'éditeur : avec plus de 40% de part de marché, ce n'est pas demain qu'Oracle disparaîtra
- Réglages fins : dans la mesure ou l'on connait suffisament le moteur, presque TOUT est paramétrable.
- Accès aux données système via des vues, bien plus aisément manipulable que des procédures stockées.
- Interface utilisateur remaniée et extrêmement riche, permettant - enfin ! - le tuning fin de requêtes par modification des plans d'exécution.
- Architecture Multi-Générationelle (MGA)
- Services Web, support XML
- Ordonnanceur intégré
- Compression des données et des sauvegardes
- Support technique Metalink extrêmement riche et fourni. [5.13]

5.5.L'outil TALEND:

Talend Open Studio est un ETL open source, développé par la société Talend, basée en France. Talend est un ETL de type « générateur de code », cœst-à-dire quœil permet de créer graphiquement des processus de manipulation et de transformation de données puis de générer lœexécutable correspondant sous forme de programme Java ou Perl. Ce programme Java ou Perl doit être déployé sur le serveur dœexécution.

Définition de processus

Les processus de manipulation de données sont créés en utilisant un modèle graphique à base de glisser/déposer. De nombreux types dœétapes sont disponibles pour se connecter aux principales bases de données ainsi que pour traiter différents types de fichiers (CSV, Excel, XML).

Talend dispose de fonctionnalités inédites pour un ETL open source :

Transform), qui tirent parti de la base de données cible s, ce qui améliore grandement les performances au prix

- des connecteurs pour certaines applications de CRM (SugarCRM, SalesForce et CentricCRM), ce qui évite de manipuler les modèles relationnels de ces outils.

- possibilité dôajouter simplement de nouvelles fonctions et composants afin de réaliser des processus plus complexes, de développer des connecteurs supplémentaires.

On notera que Talend facilite la construction des requêtes sur les bases de données en détectant les relations entre tables grâce aux clés étrangères.

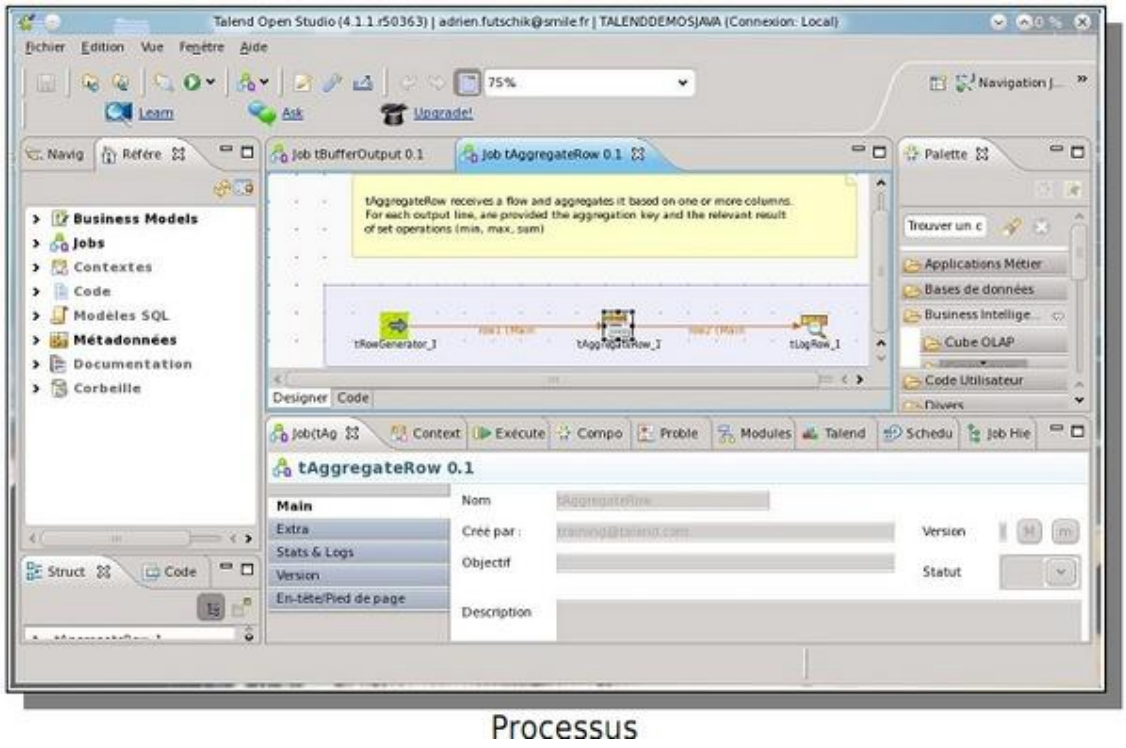


Fig.5.6 :Interface de lôoutil TALEND

Modèles métier

Talend permet également de créer un modèle métier (Business Model) afin de modéliser les interactions entre les différents systèmes et bases de données. Les informations de connexion, les métadonnées ainsi que des documents peuvent être associés à chaque élément. Le modèle constitue alors une véritable documentation du système dôinformation.

Ce mode est désormais complété avec un générateur de documentation technique permettant aux développeurs de gagner du temps dans lôétape souvent oubliée des spécifications techniques.

textes et de les appliquer aux transformations. Le même processus peut donc s'exécuter en environnement de développement, de test ou de production, avec à chaque fois l'utilisation des bonnes connexions aux bases de données et les bonnes métadonnées.

Déploiement des processus

Une fois le programme généré, celui-ci est installé par un administrateur sur la machine cible et son exécution est planifiée en utilisant le service cron d'Unix ou les tâches planifiées de Windows selon le cas.

Avec la version TIS (Talend Integration Suite), est proposée une fonction « Distant Run » et « CPU Balancer » permettant à Talend de parfaitement s'intégrer dans des environnements professionnels industrialisés. A travers la console de supervision en mode client Java ou en mode Web, les personnes chargées de l'exploitation des interfaces disposent de tableaux de bord synthétisant la bonne ou mauvaise exécution des traitements.

Talend Open Studio est un produit complet. Il a noué des partenariats avec de nombreuses sociétés éditrices de solutions décisionnelles ou de bases de données, ce qui renforce sa position sur le marché.

En 2010, Talend a complété son offre avec un nouvel outil « Talend MDM ». Ce nouvel outil vient compléter l'offre de l'éditeur avec un module permettant de construire et maintenir des référentiels. Son offre s'articule donc aujourd'hui autour de trois domaines :

- L'intégration de données, Data Integration
- La qualité de données, Data Quality
- La gestion de données référentielles, Master Data Management

Notons que Talend propose une suite « Talend Integration Suite (TIS) », soumise à souscription annuelle, qui comprend des fonctionnalités très avancées comme la gestion des déploiements complexes, la supervision des exécutions et la gestion de référentiels partagés. Comme PDI, Talend sera avantageusement utilisé dans des projets décisionnels mais trouvera encore plus sa place dans des projets d'urbanisation de systèmes d'information permettant d'uniformiser les modes d'échanges entre les différentes applications de l'entreprise. [5.14]

s ce chapitre lœensemble des outils technologiques que nous utiliserons lors de la realisation de notre application a savoir le langage de programmation, lœenvironnement de développement, le serveur, et le SGBD. Nous poursuivons notre cheminement par le chapitre qui suit à travers lequel se définira la réalisation de notre application.

Réalisation

Après avoir fixé nos choix sur les outils technologiques mentionnés dans le chapitre précédent, nous allons à présent entamer la réalisation de notre système .
Nous commençons par une description globale de lôarchitecture de notre système, ainsi quôune présentation lôentrepôt utilisé. Nous formalisons par la suite le problème de personnalisation traité.

6.1 .Architecture globale du système *:*

Fig.6.1.Architecture globale du système

Notre système est doté une architecture à 3 niveaux (appelée *architecture 3-tier*), ou il existe un niveau intermédiaire, c'est-à-dire que l'on a généralement une architecture partagée entre

- couche présentation
- couche métier
- couche accès aux données

Cette architecture peut être définie comme étant un modèle logique d'architecture applicative qui vise à séparer très nettement trois couches logicielles au sein d'une même application ou système, à modéliser et présenter cette application comme un empilement de trois strates dont le rôle est clairement défini :

- la **présentation** des données : correspondant à l'affichage, la restitution sur le poste de travail, le dialogue avec l'utilisateur
- le **traitement** métier des données : correspondant à la mise en î uvre de l'ensemble des règles de gestion et de la logique applicative

ersistantes (*persistancy* en anglais) : correspondant aux conservées sur la durée, voire de manière définitive.

Dans cette approche, les couches communiquent entre elles au travers d'un « modèle d'échange », et chacune d'entre elles propose un ensemble de services rendus. Les services d'une couche sont mis à disposition de la couche supérieure. On s'interdit par conséquent qu'une couche invoque les services d'une couche plus basse que la couche immédiatement inférieure ou plus haute que la couche immédiatement supérieure (chaque niveau ne communique qu'avec ses voisins immédiats).[6.1]

6.2. Serveurs d'application

Un serveur d'application est un serveur (ordinateur), sur lequel sont installées des applications utilisées par les particuliers. Elles sont accédées par le réseau. Ces applications, qui étaient souvent disposées sur différents postes, sont aujourd'hui de plus en plus centralisées sur des serveurs d'application. Ces serveurs sont de larges systèmes contenant les différentes applications de l'entreprise.

Dans une infrastructure régulière, on trouve plusieurs serveurs d'applications, mais il n'est pas impossible qu'il n'y en a qu'un seul, sur lequel toutes les applications seraient installées. Les applications sont chargées sur le serveur et leur résultat est affiché sur les écrans des terminaux utilisés par les clients. [6.2]

6.2. 2. Serveurs Web

Le World Wide Web est le modèle client/serveur qui consiste en des clients, légers, portables et universels qui communiquent avec de très gros serveurs. Dans la concrétisation la plus simple, un serveur Web renvoie des documents lorsque le client les demande par leur nom. Clients et serveurs communiquent par un protocole appelé HTTP. Celui-ci définit un jeu de commandes simples, où les paramètres sont transmis comme des chaînes de caractères sans typage particulier.

Dans notre cas, nous avons choisi le couple Apache / Tomcat qui assure simultanément les deux services précédemment décrits.

Présentation du couple Apache / Tomcat

Tomcat est un serveur d'application Java permettant d'exécuter des servlets et des pages serveur Java (JSP). Il est développé sous licence open-source par la fondation Apache. Il peut être utilisé ou couplé avec un serveur Web (dont Apache), et porté sur n'importe quel système sur lequel une machine virtuelle Java est installée. [6.3]
Notre choix de ce couple est justifié par ses principaux avantages qui tiennent de son mode de distribution et à son mode de licence. Il s'agit d'un produit gratuit, ouvert, disponible aussi bien sous forme binaire exécutable que sous forme de source. Les avantages de ce type de distribution sont énormes. L'avantage principal n'est pas la gratuité. Le Serveur HTTP est de plus libre, et offre une totale indépendance de l'utilisateur pour la maintenance et le développement.

Le module Tomcat

est le moteur Tomcat développé par la fondationApache.
vlet et le serveur Web sôeffectue à lôaide dôun module
peut fonctionner sur dôautres serveurs Web mais seul le couple Tomcat/Apache a été testé. Le module Tomcat du serveur Apache a été développé à partir des sources de Sun Microsystems. Il représente une implémentation de référence pour les servlets. Tomcat peut fonctionner seul, mais cela nôest pas une solution efficace. En exploitation il est préférable dôassocier Tomcat avec un serveur http plus puissant, qui se chargera du contenu statique. Tomcat pourra ainsi être mis à contribution uniquement pour les requêtes mettant en î uvre des servlets, et côest pour ce principe que nous opterons [6.4].

6.2. 3. Serveurs de données

Concernant le serveur de bases données, le client émet des requêtes SQL sous forme de messages en direction du serveur. Le résultat de chaque requête SQL est renvoyé sur leréseau. Le code qui traite la requête ainsi que les données résident sur la même machine. Le serveur utilise sa propre capacité de traitement pour rechercher les données demandées au lieu de transmettre tous les articles au client et de laisser ce dernier les traiter. Dans notre cas nous avons opté pour le serve ORACLE.

6.2. 4. Serveurs OLAP

On Line Analytical Processing : technologie d'analyse directe de données. Technologie de calcul et d'analyse de données à des fins commerciales ou de production, fondée sur des requêtes structurées suivant des critères combinés, appelés dimensions. Le serveur utilisé dans notre application est Mondrian.

Mondrian est un serveur OLAP (On Line Analytical Processing) disponible sous licence open source. Il fait partie de la catégorie des serveurs R-OLAP, c'est-à-dire qu'il accède à des données contenues dans une base relationnelle. Mondrian exécute des requêtes utilisant le langage MDX, également utilisé dans ORACLE. Ce langage permet de créer des requêtes dont lôéquivalent en langue SQL nécessiterait un grand nombre de requêtes et des temps dôexécution beaucoup plus longs. [6.4]
Lorsque Mondrian interroge lôentrepôt de données, il stocke les résultats dans un cache, et ce cache influence lôexécution des requêtes. Lorsque Mondrian est interrogé pour la première fois, le temps dôexécution de la requête est aléatoire selon la taille du résultat renvoyé. Cependant, lorsque la même requête est exécutée deux fois, la seconde exécution prend quelques millisecondes.

Le MDX :

MDX (Multi Dimensional eXpression) est un langage de requêtes pour les bases de données multidimensionnelles, de la même manière que SQL est utilisé pour les requêtes sur les bases de données relationnelles. Dans son approche, MDX est proche du SQL sur son aspect select et where même si la similarité ne va pas plus loin. Le but des expressions multidimensionnelles MDX est de rendre aisé et intuitif l'accès aux données de différentes dimensions.

MDX est fait pour naviguer dans les bases multidimensionnelles et pour définir des requêtes sur tous les objets (dimensions, hiérarchies, niveaux, membres et cellules) afin d'obtenir une représentation sous forme de tableaux croisés.

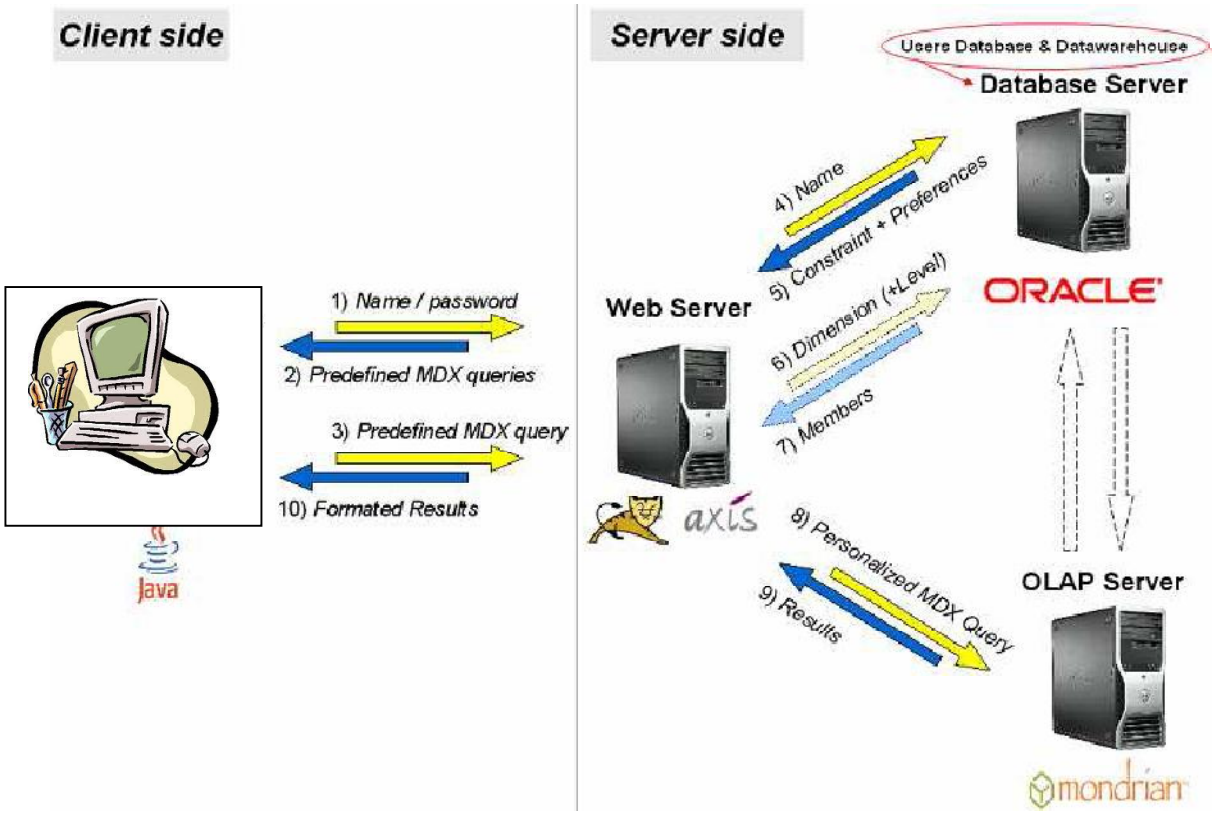


Fig.6.2 : Architecture technique de lôapplication.

Lôapplication est divisée en deux parties qui sont toutes les deux développées en langage Java : un côté client sôexécutant avec *J2ME* [6.5] et un côté serveur sôexécutant sur un serveur *TomCat* . Dans lôapplication, chaque utilisateur peut avoir son propre compte sur le serveur. Pour chaque compte, les propriétés suivantes sont stockées dans une base de données :

Nom : le nom de lôutilisateur afin de lôidentifier
Mot de passe : le mot de passe de lôutilisateur
Profil : le profil utilisateur correspond à ses préférences, ou centres dôintérêt.
Requêtes MDX prédéfinies : On proposera des menus afin que lôutilisateur puisse construire sa requête.

La figure 6.2 illustre lôutilisation de base de notre application. Tout dôabord, lôutilisateur lance lôapplication cliente sur ordinateur et se logue en entrant son nom dôutilisateur et son mot de passe. Une requête est envoyée vers le serveur (message 1). Si le nom dôutilisateur et le mot de passe sont corrects, les requêtes prédéfinies sont envoyées vers lôappareil de lôutilisateur (message 2) et ces requêtes sont temporairement stockées dans la mémoire de lôappareil. Lôutilisateur choisit une des requêtes prédéfinies. La requête sélectionnée est ensuite envoyée vers le serveur web (message 3). Une fois que le serveur web a reçu la

s de lôutilisateur en interrogeant la base de données
ur lôentrepôt de données peut aussi être effectuée si la
icitement tous les membres nécessaires à lôexécution de celle-ci (par exemple, si la requête MDX contient des éléments du langage MDX tels que ômembersô) (messages 6 et 7). Une fois que toutes ces informations ont été récupérées, le serveur web peut procéder à la personnalisation de la requête.
Cette requête générée est alors envoyée à un serveur OLAP (message 8). Nous utilisons pour notre prototype un serveur OLAP open-source nommé Mondrian. Les résultats de ce serveur (message 9) sont envoyés vers le serveur web qui les envoie vers lôutilisateur (message 10). Lôordinateur de lôutilisateur peut alors afficher le résultat sur lôécran.

6.3 .Déploiement de l'application

Notre avons bâti notre application avec une panoplie de servlets, servant à effectuer les traitements nécessaires, des JSP pour une mise en page convenable , Ainsi quôun document XML qui sôoccupera de la configuration de notre application déployée au sein du serveur Tomcat.

Cette ultime phase comporte un certain nombres dôétapes qui se succèdent.
Nous avons entamer notre système par la création de nos bases au sein du SGBD Oracle.

Création des bases de données

Après avoir installé Oracle SGBD, nous avons entamer la création de nos bases de données, deux pour être exacte, la pour stocker les données propres à lôutilisateur, et la seconde qui servira à alimenter notre entrepôt de données.

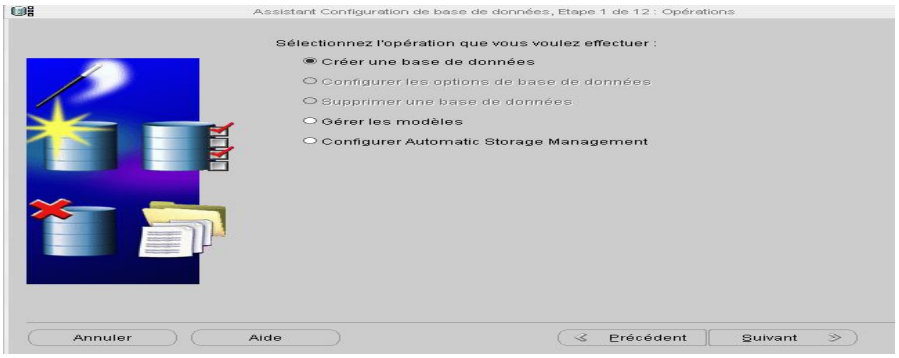


Fig.6.3. Assistant configuration de bases de données

Une fois la base crée, on saisit lôurl suivant http://localhost:5560/isqlplus/ que nous avions récupéré lors de lôinstallation dôOracle dans la barre de navigation ce qui nous donne accès a la page suivante après avoir saisie le nom du système utilisé et le mot de passe introduit lors de la configuration de la base en question :

Fig.6.4. Interface SQL Plus

Cêest dans cet espace de travail que nous avons crée lêintégralité des tables constituant chacune des bases, et ce avec le langage SQL PLUS qui est un langage est un langage L4G (entendez par ce terme un langage de quatrième génération), fournissant une interface procédurale au SGBD Oracle. Le langage PL/SQL intègre parfaitement le langage SQL en lui apportant une dimension procédurale.

Création dêun projet tomcat :Après avoir rajouter les plug-ins nécessaires dans le document plug-in de eclipse, on passe à la création du projet

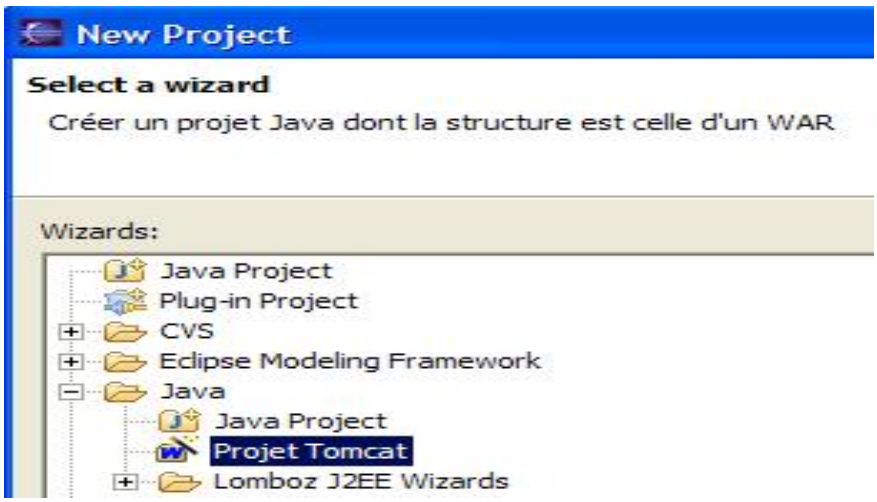


Fig.6.5. Projet Tomcat

ts accèdent à notre base crée sous Oracle, ceci pour insertion, suppression, modification, sélection, ou vérification. Donc avant de passer à la programmation, lôajoute du jar du JDBC dans le path du projet crée est primordial.

Nous avons aussi par la même occasion, rajouter les jar du serveur Mondrian afin de pouvoir accéder aux cubes do données via eclipse.

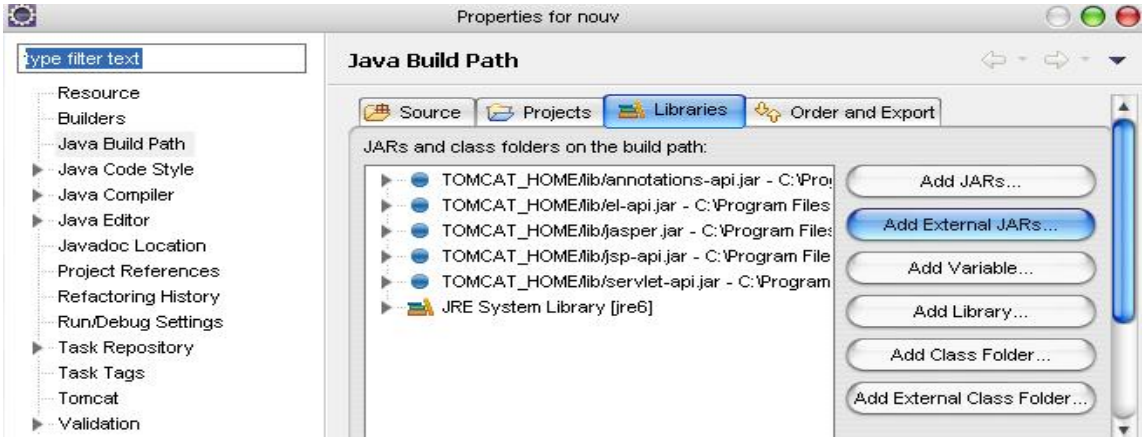


Fig.6.6. Java build path

Création des classes associées aux servlets :

Après avoir attribuer un nom a la servlet, on passe au code.
Comme nous lôavons indiquer un peu plus haut, les servlets de notre applications effectuent différentes tâches, qui requiert lôaccès à la base.
Nous expliciterons la structure des servlets formants notre plate-forme dans le tableau suivant :

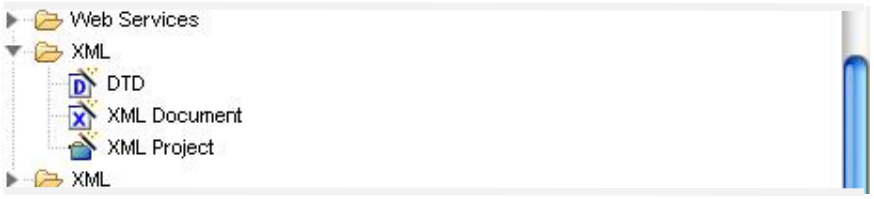
Etapes	*Code*

Chargement du pilote JDBC pour Oracle	Try {
	Class.forName(" oracle.jdbc.driver.OracleDriver") ;
	}

Définition de lôURL de connexion	String url = "jdbc:oracle:thin:@localhost:1521:GESTION";
Etablissement de la connexion	Connection con = DriverManager.*getConnection*(url,"SYSTEM","sonia");
Création dôune instruction	Statement stmt = con.createStatement();
Exécution de la requête	String req = "select * from abonné where pseudo=' +pseudo+" '" ; stmt.executeUpdate(req);
Traitement des resultats	while(rs.next()) { out.println(""+rs.getString("nom"));}
Fermeture de la connexion	c.close();

TAB.6.1. Structure des servlets dôaccès à la base

Création du document XML
Click droit sur le projet> new>other>

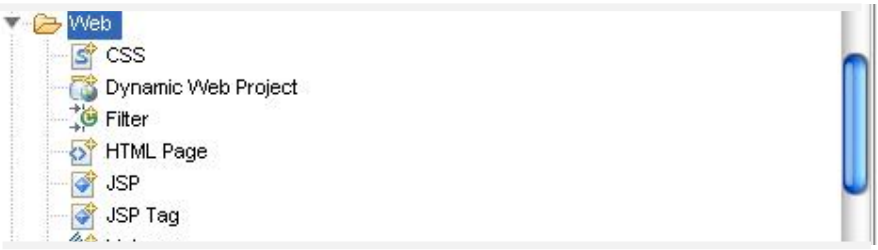


Qui nous servira à localiser nos servlets dans le serveur Tomcat, chaque servlet crée devra être rajouter dans le document xml nommé web.xml.
Voici le code XML de la servlet inscription

```
vlet.nouv.inscriptio</servlet-class>
                <servlet-name>inscription</servlet-name>
</servlet>
<servlet-mapping>
            <servlet-name>inscription</servlet-name>
           <url-pattern>/ininscription</url-pattern>
</servlet-mapping>
</web-app>
```

Pour lôinterface nous avons opter pour les jsp

Création des JSP



Voici un fragment de JSP provenant de notre page dôaccueil :

```
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1">
<title>Sans titre</title>
<body bgcolor="white" text="black" link="blue" vlink="purple" alink="red"
background="C:\Documents and Settings\GENICOM\Bureau\sites\fb.jpg"
OnLoad="namosw_rotate_banner();">
<table border="1" width="1062" height="500" align="center">
   <tr>
      <td width="1052" colspan="3" height="172">
         <p align="center"><a href="javascript:namosw_rotate_banner_go()"
target="_self"><img border="0" src="alph-x.jpg" width="1062" height="167"
name="image1"></a></p>
      </td>
   </tr>
   .
   .
   .
</body>
</html>
```

et :

Lôinvocation dôune servlet sôeffectue par lôintroduction de lôurl liée à la servlet dans le champ dôadresse du navigateur, lôurl est définit comme suit
http://localhost:8080/ nom_du_projet/Mapping_url, tel que Mapping_url est le nom par lequel la servlet sera invoquée et qui sera déterminé dans le document XML, quôon a appelé web.xml.

Fig.6.7. Page dôaccueil du site

Notre entrepôt sera alimenté par un ETL, nous avons dans notre cas employer Talend Open Source.

Présentation de TALEND

L'ETL TOS est un générateur de code, c'est à dire que, pour chaque job créé, un moteur va s'occuper de générer du code Java ou Perl (selon le choix de l'utilisateur) qui permettra d'exécuter la transformation. TOS possède une interface graphique basée sur Eclipse. La gestion des metadonnées se fait via un référentiel très complet (au format XML). TOS possède aussi un Business Modeler permettant de modéliser les architectures décisionnelles de haut niveau et de façon non technique. [6.6]

on a la base de données ORACLE, en spécifiant le nom
ondant, nous créons ensuite ce que TALEND appelle un
« Job » qui fera le lien entre les tables de la base et celle de notre entrepôt. Pour le langage
nous avons spécifier Java, qui sera généré lors de la création du job.

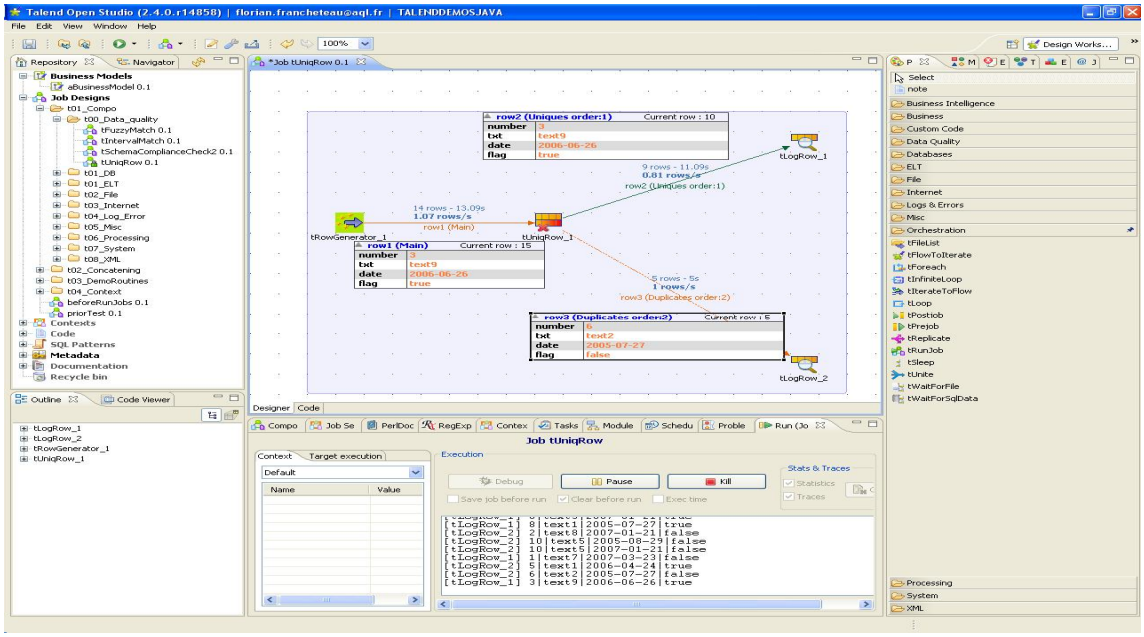


Fig.6.8 :Talend Open Studio : Création d'un job

6.4. Mise en œuvre de la solution proposée

Nous allons a ce niveau, présenter les fonctionnalités de notre système, et la logique sur la quelle on sôest basé, toujours avec lôobjectif de servir efficacement lôutilisateur. Chacune de ces fonctionnalités résulte de lôexécution dôun certain nombre de servlets combinées de telle sorte à obtenir un enchainement dôinterfaces cohérent et plausible.

6.4.1. Présentation du système

Faute de ne pas avoir eu le privilège de collaborer avec une entreprise de taille a affronter un entrepôt de données, de par son volume de données, nous avons imaginer un système appelé à être utilisé par une multinationale de distribution de divers produits , une grande entreprise sollicité par divers pays tels que :la France, lôAngleterre, lôEspagne, lôAllemagne et lôItalie. Il serait dans ce genre de situation difficile de réaliser un tableau de bord avec trop de données, ce qui pourrait intéresser lôentreprise entre autre serai de savoir **quel** produit a été vendu a **combien**, **quand**, et **ou**.

Ce système donnerait la possibilité aux décideurs selon la hiérarchie de lôentreprise dôinterroger lôentrepôt de données pour une meilleure connaissance de lôétat de lôavancement des ventes.

Notre travail a commencé par la création dûne base de données sous oracle, qui servira alimenter notre entrepôt de données oracle toujours. Cette base a pour modèle E/A le schéma suivant :

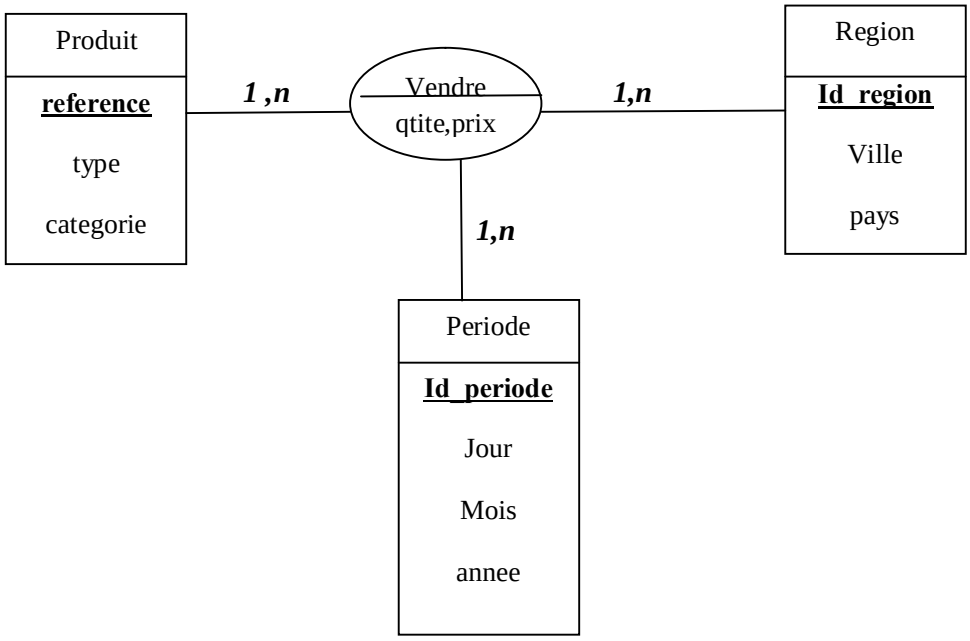


Fig.6.9 :Modèle E/A

On en déduit les tables suivantes :

Produit (**référence**, type, catégorie) ;
Période (**id_période**, jour, mois, année) ;
Région (**id_région**, ville, pays) ;
Vente (réference, id_période, id_région, quantité, prix) ;

6.4.3. L'entrepôt de données :

Comme nous avons défini le système précédemment, ce dernier sera amené à procurer aux décideurs des informations à savoir des analyses surs les produits commercialisés. Il aura pour modélisation le diagramme suivant :

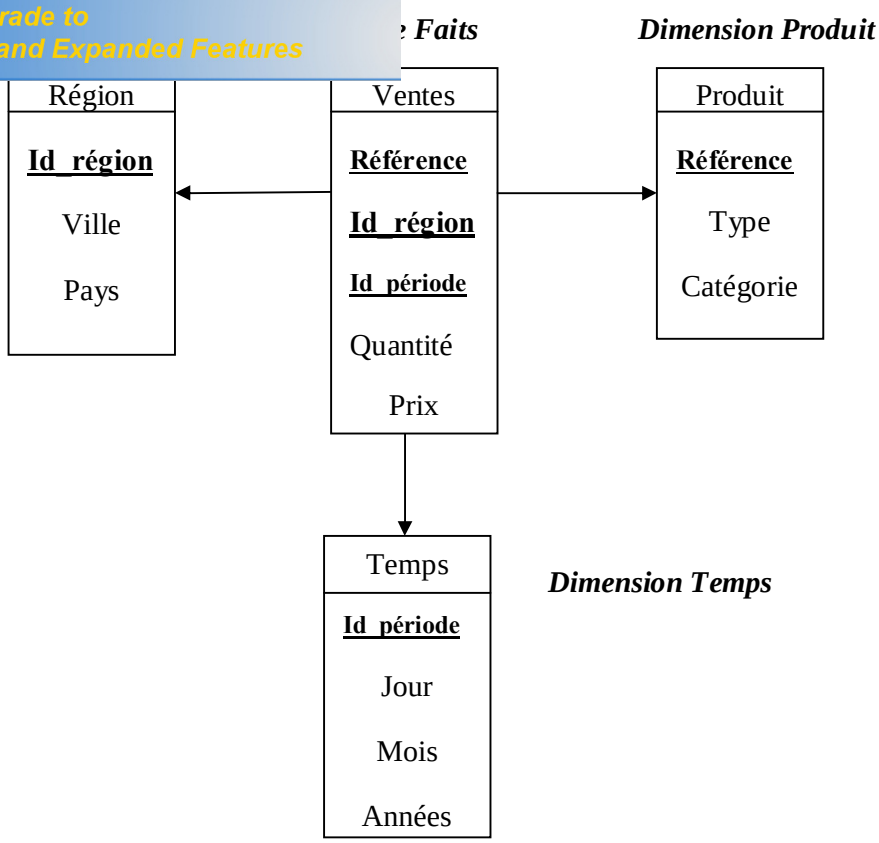


Fig.6 .10: Schéma en étoile de lôentrepôt

Lôentrepôt contient contient les tables suivantes :

Ventes: cette table est la table de fait qui représente les ventes , elle contient tous les données relatives à la vente.

Période : cette table représente la dimension ôôtempsôô, elle contient toutes les données relatives à une date (jour, mois, année,é).

Région : cette table représente la dimension ôôrégionôô, elle contient toutes les données relatives à une localisation (ville, pays, é),

Produit : cette table représente la dimension ôôproduitôô, elle contient toutes les informations relatives à un produit (la réference, la catégorie, le type).

6.4.4. La base de données utilisateur :

Nous avons conçu le profil utilisateur en nous supposant lôorganigramme suivant :

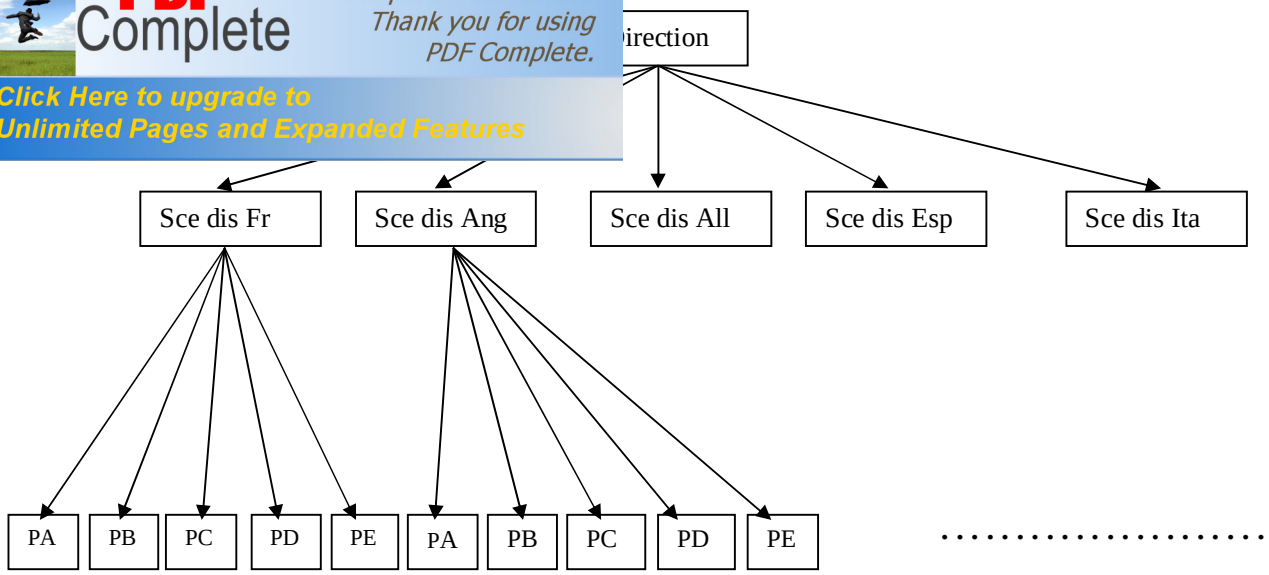


Fig.6.11:Organigramme de lôentreprise fictive

Direction : qui représente dans notre cas le propriétaire directeur général, qui est un des utilisateurs du système mais dôune manière assez généralisé, c'est-à-dire que nous estimons que ce dernier nôa pas besoin de disposer dôun profil, mais plutôt dôassimiler de manière abstraite et vague lôétat de lôavancement des ventes.

Service distribution France: Ce service a pour rôle dôapprovisionner les locaux de France de tous les produits commercialisés par lôentreprise. Ces locaux sont situés dans plusieurs villes a savoir :Paris, Lyon, Strasbourg.

Service distribution Angleterre: Ce service a pour rôle dôapprovisionner les locaux de dôAngleterre de tous les produits commercialisés par lôentreprise. Ces locaux sont situés dans plusieurs villes a savoir : Londres, Manchester, Liverpool.

Service distribution Allemagne: Ce service a pour rôle dôapprovisionner les locaux dôAllemagne de tous les produits commercialisés par lôentreprise. Ces locaux sont situés dans plusieurs villes a savoir :Berlin, Munich, Francfort, .

Service distribution Espagne: Ce service a pour rôle dôapprovisionner et gérer les locaux dôEspagne de tous les produits commercialisés par lôentreprise. Ces locaux sont situés dans plusieurs villes à savoir : Madrid, Barcelone, valence.

Service distribution Italie: Ce service a pour rôle dôapprovisionner les locaux de France de tous les produits commercialisés par lôentreprise. Ces locaux sont situés dans plusieurs villes à savoir : Rome, Naples, Venise.

Pour chacun des services cités ci-dessus, en descendent des bureaux, ou pour chacun la priorité dôun type de produit spécifique c'est-à-dire par exemples pour le service distribution France nous avons :

r rôle de gérer les ventes des locaux situés en France, et produits de type A, qui pourraient être par exemple des

Bureau produit B : Ce bureau a pour rôle de gérer les ventes des locaux situés en France, et ce uniquement en ce qui concerne les produits de type B, qui pourraient être par exemple des chaussures.

Bureau produit C : Ce bureau a pour rôle de gérer les ventes des locaux situés en France, et ce uniquement en ce qui concerne les produits de type C, qui pourraient être par exemple des sacs.

Bureau produit D : Ce bureau a pour rôle de gérer les ventes des locaux situés en France, et ce uniquement en ce qui concerne les produits de type D, qui pourraient être par exemple des accessoires.

Bureau produit E : Ce bureau a pour rôle de gérer les ventes des locaux situés en France, et ce uniquement en ce qui concerne les produits de type E, qui pourraient être par exemple des parfums.

Remarque : Nous pouvons aussi supposer quôa lôintérieur de chaque bureau, un employé pour chaque ville de la localité du service dont dérive son bureau, exemple : dans le service distribution France, dans le bureau produit A, un employé pour chacune des villes : Paris, Lyon, Strasbourg. Il serait intéressant dans ce cas de munir cet employé dôun profil spécifiant le type de produit, ainsi que la ville dans laquelle il intervient,

Tout ceci pour dire que dans une entreprise, les tâches des employés sont différentes, et que les dimensions de leurs besoins dôanalyses ne sont pas forcement les mêmes, et quôavec un nombre considérable de dimension, difficile pour un employé donnée de dôatteindre de façon directe lôinformation qui concerne son statut au sein de lôentreprise, côest de ce fait que la notion de profil peut apporter sa contribution, et remédier à ce genre de situation qui peut être pénalisante surtout du point de vue temps.

La figure ci-dessous schématise le modèle de profil que nous avons choisi :

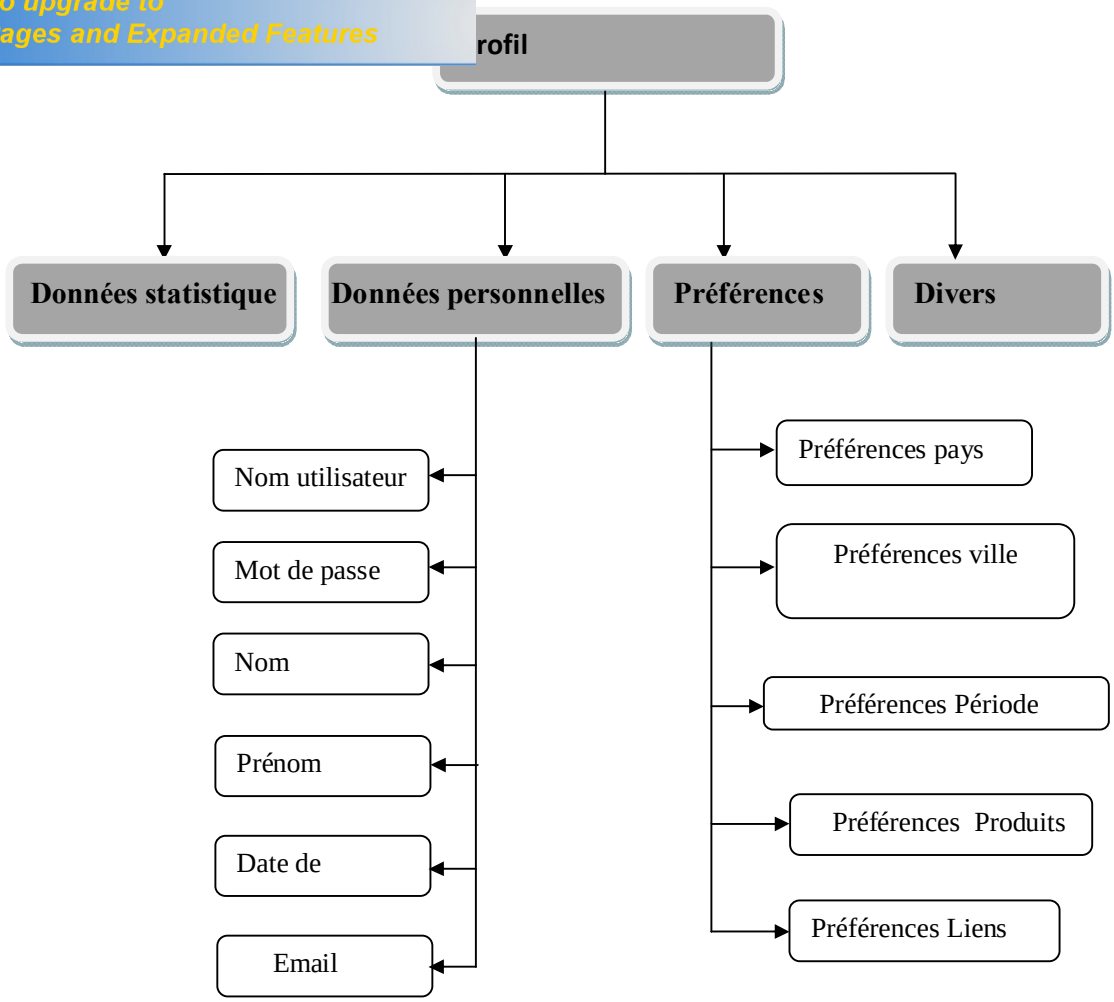


Fig.6.11: représentation multidimensionnelle dûun profil

Nous avons ainsi copté pour un profil incluant les informations suivantes :

Données personnelles

Nom utilisateur : une chaine de caractères représentant le nom ou le pseudo nom de lûutilisateur ;
Mot de passe : une chaine de caractères représentant un mot de passe ;
Nom : une chaine de caractères représentant le nom propre de lûutilisateur ;
Prénom : une chaine de caractères représentant le prénom de lûutilisateur ;
Date de naissance : une chaine de caractères représentant la date de naissance ;
Email : une chaine de caractères représentant son adresse pour contacter utilisateur;

Pays : Une liste déroulante contenant tout les pays que couvre lôentreprise ;
Ville : Une liste déroulante contenant toutes les villes que couvre lôentreprise ;
Période : une liste déroulante contenant les membres de la dimension période suivant : mois, année, ou jour, pour préciser le degré dôanalyse (journalier, mensuel, annuel)

Préférences implicites

Liens : représente les liens dôanalyses dont lôutilisateur fut intéressé lors de ses connections, et qui sôinsèrent implicitement dans son profil

La base de données utilisateur contiendra ainsi les tables suivantes :

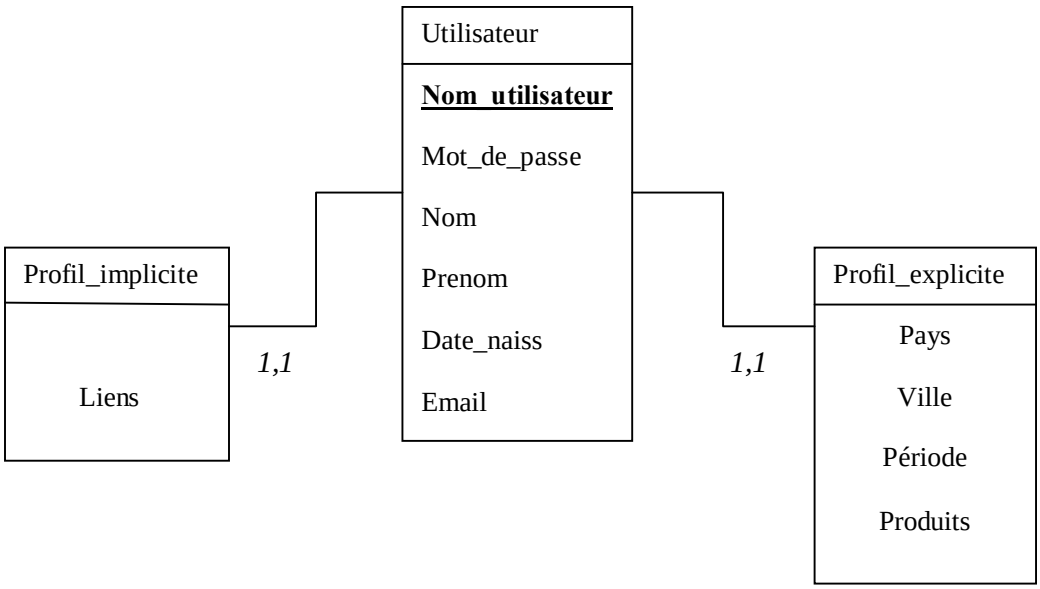


Fig.6.12 :Diagramme de la base de données Utilisateurs.

6.5 .Personnalisation des requêtes

Nous avons dans notre cas modéliser les données de ventes en utilisant un cube qui est décrit sous la forme de schéma en étoile par la figure 6.6.

Ce cube, nommé ôSalesCubeô, est composé dôune table de faits ôVenteô et de quatre dimensions qui sont ôPériodeô, ôProduitô, et ôRégionô .Chaque cellule de ce cube stocke la

produit particulier sur une région particulière pour une

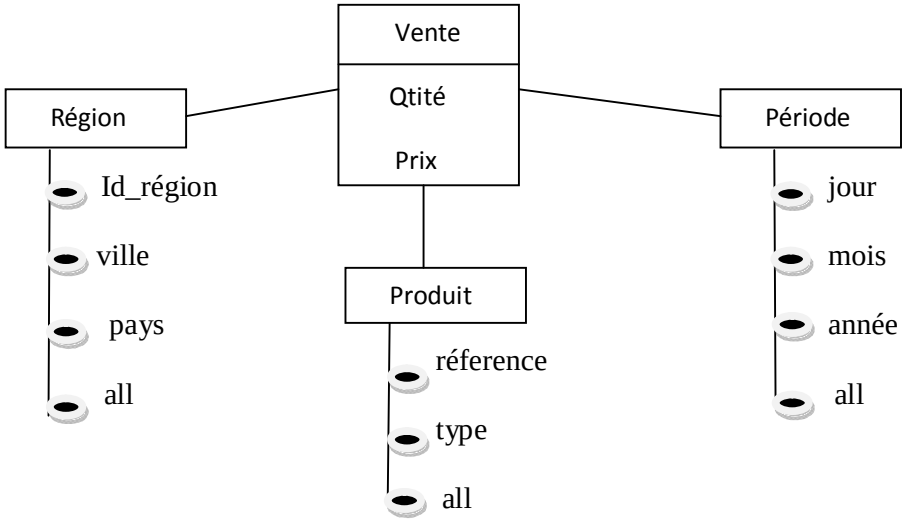


Fig. 6.13 :Le cube SalesCube en schéma en étoile

Dans le contexte OLAP, la communauté des bases de données sôintéresse à la définition dôune modélisation logique en prenant en compte la nature multidimensionnelle et hiérarchisée des données. Plusieurs modèles multidimensionnels formels et langages dôinterrogation correspondants ont été proposés (présentant un état de lôart) où lôobjectif majeur est la vision multidimensionnelle des données pour faciliter les prises de décisions. [6.7]

Cependant, chaque approche présente une vision particulière des besoins, de la terminologie et du formalisme de lôanalyse multidimensionnelle. En conséquence, il nôy a aucun modèle multidimensionnel formel consensuel établi.

Notons quôen pratique, le langage utilisé est le langage MDX proposé par Microsoft ([Microsoft, 1998]). Il est doté dôune syntaxe de type SQL et représente un moyen syntaxique de définition et dôinterrogation des données multidimensionnelles de lôentrepôt. A ldifférence dôune requête SQL, une requête MDX renvoie une grille multidimensionnelle de cellules (ou table croisée) comme résultat à la requête. De plus, la présentation du résultat est indiquée dans la requête MDX (nombre de dimensions, nombre dôaxes, etc.).

Nous décrivons dans ce qui suit, le cube SalesCube . Ce cube est de schéma sch(SalesCube) = {Période, Région, Produit} où :
ï Lôensemble des dimensions est D = {Période, Région, Produit }. Les schémas de ces quatre tables de dimension, c.-à-d :

_ sch(Période) = {id_période, jour, mois, année, all}.

née, all}.
egorie, all}.
e ôVenteô, est de schéma sch(Vente) = {Période, Région, Produit, val} où le domaine de lôattribut ôvalô est lôensemble de toutes les valeurs de mesure possibles.

Nous avons décrit notre cube en décrivant son schéma. Un cube de données peut aussi être vu comme un ensemble de cellules où chacune dôelles représente une association entre un membre de chaque dimension et une mesure (ou contenu de cellule).
Chaque cellule du cube SalesCube stocke les ventes pour un produit sur une région à une période donnée et pour une mesure donnée.

Une des particularités de lôOLAP, est de pouvoir formuler des requêtes sur des cubes de données et de pouvoir ensuite construire des visualisations pour lôensemble résultat. Dans notre modèle, nous avons distingué deux niveaux : données et visualisation. Nous avons considéré dôune part un ensemble de données, détaillées ou agrégées, organisées sous forme de cubes de données et dôautre part, la présentation de ces données sous-forme de tables croisées.

Sachant quôune requête MDX permet de rechercher des mesures à des niveaux différents de granularité en formulant une expression unique et de spécifier comment afficher la réponse sous forme de table croisée, nous considérons quôune instance de visualisation correspond au résultat dôune requête MDX sur une instance du cube. Une instance de visualisation est composée dôun ensemble de faits que lôutilisateur souhaite voir et dôune structure pour lôaffichage de la table croisée.

Par exemple, considérons la requête MDX suivante :

Q1:SELECT {[annee].2003, [annee].2004, [annee].2005, [annee].2006} ON COLUMNS,
CROSSJOIN({[ville].Paris,[ville].Lyon,},
{[Type].A, [Type].B}) ON ROWS
FROM SalesCube
WHERE [Measures].quantity

Cette expression MDX recherche la quantité de produits vendus par année et type au niveaux indiqués de la dimension Région.
De plus, elle indique que les membres des dimensions Région et Produit seront imbriqués et seront affichés en lignes, et que les membres de la dimension Période apparaîtront en colonnes. Cela signifie que dans cet exemple, la structure dôaffichage du résultat est spécifiée par lôutilisateur en précisant la clause ôONô de MDX dans ôSELECTô.

		2003	2004	2005	2006
Paris	A	1314	2334	2544	3012
	B	956	1204	1143	1045
Lyon	A	876	956	1324	1234
	B	967	834	1323	1567

Tab.6.2 : Résultat de la requête Q1

e une petite comparaison entre deux types de requêtes,
ndrait en compte le statut de son émetteur.

Requête non personnalisé :

Dans ce cas la requête ne précise pas forcément un pays donnée, ni un type de produit donné

SELECT {[mois].janvier, [annee].fevrier, [annee].mars, [annee].avril,[annee].mai, [annee].juin } ON COLUMNS,
CROSSJOIN ({[pays].France, [pays].Angleterre,[pays].Espagne, [pays].Allemagne, [pays].Italie },
{[Type].A, [Type].B,{[Type].A, [Type].B{[Type].A }) ON ROWS
FROM SalesCube
WHERE [Measures].quantity

Le tableau ci-dessous représente la réponse à la requête non personnalisée sur notre cube de données :

Fevri	*Mar*	*Avril*	*Mai*	*Juin*	*Juille*	*Août*	*Septem*

				325	754	345	445	667	325	754	345
				909	889	334	234	734	909	889	334
		C	988	976	398	776	095	835	976	398	776
		D	364	746	384	345	342	985	746	384	345
		E	536	737	634	664	876	894	737	634	664
FRANCE	Lyon	A	474	439	949	098	234	223	439	949	098
		B	747	883	439	131	990	489	883	439	131
		C	563	871	672	889	234	435	871	672	889
		D	546	776	234	890	432	987	776	234	890
		E	435	667	534	778	254	876	667	534	778
	Strasbourg	A	523	728	898	984	234	111	728	898	984
		B	546	776	835	399	234	456	776	835	399
		C	554	736	112	546	947	255	736	112	546
		D	213	664	777	982	112	987	664	777	982
		E	536	773	425	829	992	092	773	425	829
ALLEMAGNE	Berlin	A	641	637	937	663	921	112	637	937	663
		B	355	636	672	635	873	934	636	672	635
		C	324	635	534	435	424	777	635	534	435
		D	332	435	762	723	637	993	435	762	723
		E	332	324	553	937	993	563	324	553	937
	Munich	A	738	736	663	324	435	645	736	663	324
		B	323	443	546	745	660	363	443	546	745
		C	076	099	078	123	128	536	099	078	123
		D	453	526	314	324	123	637	526	314	324
		E	213	443	122	166	176	553	443	122	166
		A	738	736	663	324	435	645	736	663	324

				443	546	745	660	363	325	754	345
				099	078	123	128	536	909	889	334
	Francfort	D	453	526	314	324	123	637	976	398	776
		E	213	443	122	166	176	553	746	384	345
ANGLETERRE	Londres	A	641	637	937	663	921	112	737	634	664
		B	355	636	672	635	873	934	439	949	098
		C	324	635	534	435	424	777	883	439	131
		D	332	435	762	723	637	993	871	672	889
		E	332	324	553	937	993	563	776	234	890
	Manchester	A	738	736	663	324	435	645	667	534	778
		B	323	443	546	745	660	363	728	898	984
		C	076	099	078	123	128	536	776	835	399
		D	453	526	314	324	123	637	736	112	546
		E	213	443	122	166	176	553	664	777	982
	Liverpool	A	641	637	937	663	921	112	773	425	829
		B	355	636	672	635	873	934	637	937	663
		C	324	635	534	435	424	777	636	672	635
		D	332	435	762	723	637	993	635	534	435
		E	332	324	553	937	993	563	435	762	723
	Madrid	A	738	736	663	324	435	645	324	553	937
		B	323	443	546	745	660	363	736	663	324
		C	076	099	078	123	128	536	443	546	745
		D	453	526	314	324	123	637	099	078	123
		E	213	443	122	166	176	553	526	314	324
		A	641	637	937	663	921	112	443	122	166
		B	355	636	672	635	873	934	736	663	324

				635	534	435	424	777	325	754	345
				435	762	723	637	993	325	754	345
		E	332	324	553	937	993	563	909	889	334
	Valence	A	345	325	754	345	445	667	976	398	776
		B	578	909	889	334	234	734	746	384	345
		C	988	976	398	776	095	835	737	634	664
		D	364	746	384	345	342	985	439	949	098
		E	536	737	634	664	876	894	883	439	131
ITALIE	Rome	A	641	637	937	663	921	112	871	672	889
		B	355	636	672	635	873	934	776	234	890
		C	324	635	534	435	424	777	667	534	778
		D	332	435	762	723	637	993	728	898	984
		E	332	324	553	937	993	563	776	835	399
	Venise	A	738	736	663	324	435	645	736	112	546
		B	323	443	546	745	660	363	664	777	982
		C	076	099	078	123	128	536	773	425	829
		D	453	526	314	324	123	637	637	937	663
		E	213	443	122	166	176	553	636	672	635
	Naples	A	345	325	754	345	445	667	635	534	435
		B	578	909	889	334	234	734	435	762	723
		C	988	976	398	776	095	835	324	553	937
		D	364	746	384	345	342	985	736	663	324
		E	536	737	634	664	876	894	443	546	745

Tab. 6.3 : Résultat de la requête non personnalisée

données dont dispose notre cube, le tableau résultant de e assez exorbitant du point de vu volume, et quôil ne de tout employé, qui naturellement nôeffectuent pas tous la même tâche. Il serait alors difficile pour un employé chargé dôun seul type de produit, ex :type C, et ce dans tous les pays qui le commercialisent, de constater clairement lôévolution du produit donnée a travers le temps dans différents endroits. Côest pourquoi le fait dôétablir un profil selon lôintérêt de lôemployé pour une quelconque analyse en fonction de son rôle au sein de lôentreprise, pourrait lui donner directement accès à lôinformation pertinente dont il a exactement besoin et lui faire gagner un temps considérable, c'est-à-dire quôon lui affichant directement son objectif, il nôaura plus a trier, et lorsquôon parle dôentrepôt de données, on parle bien évidement de volume de données immenses, ce qui rendrait le tri dôautant plus pénible. Nous verrons dans ce qui suit que le fait de se servir dôun profil peu bien faciliter la vie de lôentreprise.

Requête personnalisé :

Nous prenons le cas dôun employé chargé de lôanalyse à savoir la quantité vendu dôun seul type de produits dans tous les pays concernes par mois, la requête après avoir récupérer les préférences de lôemployé donné pourrait être :

```
SELECT {[mois].janvier, [annee].fevrier, [annee].mars, [annee].avril,[annee].mai,
[annee].juin } ON COLUMNS,
CROSSJOIN ({[Type].C },
{[pays].France, [pays].Angleterre,[pays].Espagne, [pays].Allemagne, [pays].Italie })
ON ROWS
FROM SalesCube
WHERE [Measures].quantity
```

Le tableau ci-dessous représente la réponse à la requête personnalisée dans ce cas de figure sur notre cube de données :

		Janvier	Février	Mars	Avril	Mais	juin
C	France	256	287	807	878	523	829
	Angleterre	546	635	834	113	435	534
	Espagne	435	874	534	934	127	738
	Allemagne	443	637	773	735	524	234
	Italie	435	776	601	201	534	778

Tab.6.4 : Résultat de la requête personnalisée

Il est désormais claire que le résultat ci-dessus pour le type dôemployé cité précédemment témoigne beaucoup plus de clarté et de facilité de compréhension, mais comme nous disions avant, les employés nôont pas tous les mêmes tâches a réaliser, côest

ème sur lôidentité de lôémetteur, en incluant ses centres
alyses, pour mieux cibler ses attentes et lui faire ainsi
que le temps côest de lôargent.

La comparaison entre les deux tableaux est amplement suffisant pour démontrer lôutilité que peut avoir la personnalisation au sein de ce genre dôentreprises, face a chacun des résultats des deux types de requêtes précédentes, lôapport de la personnalisation utilisant un profil nôest plus a denier.

6.6. Déroulement :

Lorsquôun utilisateur se connecte au système 4 scénarios sont alors possible :

Scénario 1 : Utilisateur déjà inscrit, saisie son nom dôutilisateur et mot de passe,

Fig.6.14 : Interface de connexion

Si ces derniers sont corrects il accède à son espace utilisateur, qui lui offre le choix entre une analyse personnalisée, et dôautres types dôanalyses qui nôincluent pas son profil.

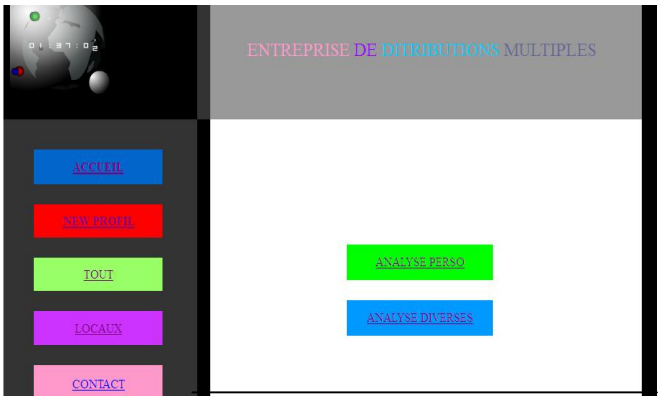


Fig.6.15 : Interface utilisateur

ne analyse personnalisée, lôinterface suivante lui sera

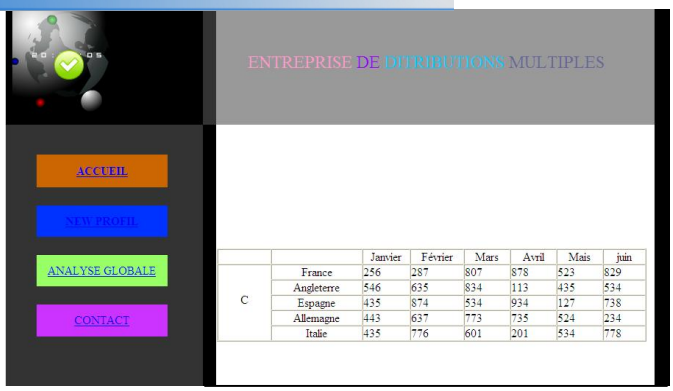


Fig.6.16 : Interface analyse personnalisée

2.Si par contre il choisit la seconde alternative, une page contenant un certain nombres de liens proposant diverses analyses. Celle qui susciterons lôintérêt de lôutilisateur seront prises en compte dans son profil.

Fig.6.17 : Interface liens multiples

Scénario 2 : Pour un utilisateur non connu du système, la démarche à suivre est donc de sôinscrire, lors de lôinscription il est amené à renseigner les champs que le système lui retourne. Ces derniers sont ensuite enregistré dans la base de donnée utilisateur pour permettre à lôutilisateur lors de ses prochaines connections avec le système dôaccéder à la partie de lôinformation dont il nécessite.

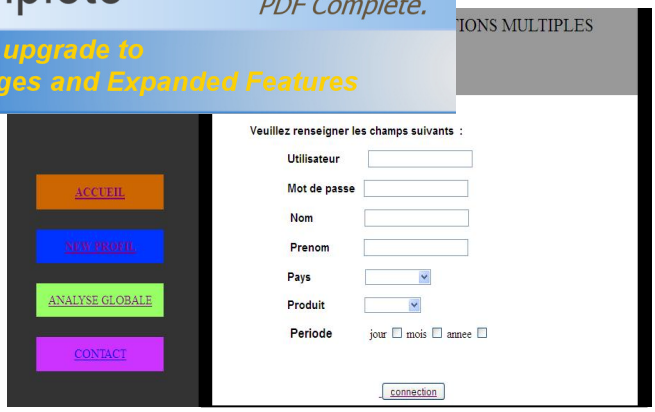
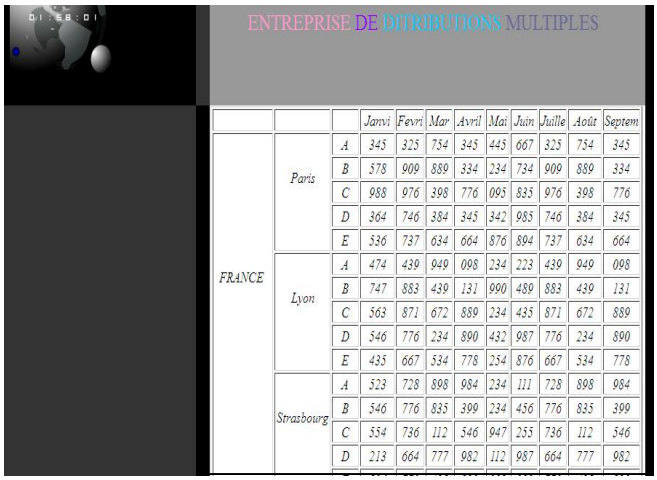


Fig.6.18: Interface formulaire profil

Scénario 3 : Ce scenario propose une analyse extrêmement détaillée de toutes les données de lœntreprise, et ce pour mieux apprécier lœanalyse personnalisée**.**



			Janvi	Fevri	Mar	Avril	Mai	Juin	Juille	Août	Septem
FRANCE	Paris	A	345	325	754	345	445	667	325	754	345
		B	578	909	889	334	234	734	909	889	334
		C	988	976	398	776	093	835	976	398	776
		D	364	746	384	345	342	985	746	384	345
		E	536	737	634	664	876	894	737	634	664
	Lyon	A	474	439	949	098	234	223	439	949	098
		B	747	883	439	131	990	489	883	439	131
		C	563	871	672	889	234	435	871	672	889
		D	546	776	234	890	432	987	776	234	890
		E	435	667	534	778	254	876	667	534	778
	Strasbourg	A	523	728	898	984	234	111	728	898	984
		B	546	776	835	399	234	456	776	835	399
		C	554	736	112	546	947	255	736	112	546
		D	213	664	777	982	112	987	664	777	982

Fig.6.19 : Interface analyse détaillée

é pour nous une aubaine a travers laquelle nous avons acquis d'une part des nouvelles connaissances, et d'autre part, assimiler les différents outils usités en matière de développement répondant dûne manière spécifique aux besoins en information de lûutilisateur.

Nous avons lors de cette étape tenté une construction dynamique du profil utilisateur, notre contribution se situe à plusieurs niveaux à savoir dûne part la modélisation des dimensions composant le profil, nous avons estimé que pour offrir à lûutilisateur un espace de recherche personnalisé, le profil doit être caractérisé par plusieurs dimension pour contenir la description la plus exhaustive que possible des préférences de lûutilisateur, et dûautre part, la modélisation multidimensionnelle nous a permis de savoir ce que un utilisateur attend de notre système.

Il est clair que la personnalisation a encor du chemin à faire, elle on est quôà son début, je suis cependant sûre quôavec le surabondance de informationnelle rencontrée dans tous les domaines existants, son indispensabilité ne sera plus a prouver.

Les entrepôts de données quand a eux sont devenus aujourdôhui incontournable dans le domaine de lôaide à la prise de décision. Couplés avec les possibilités des techniques OLAP, ils permettent une étude analytique poussée des données, procurant une efficace aide à la décision dans tous les domaines.

Aujourdôhui lôinformation joue un rôle capital dans le processus dôaide à la décision : connaitre la bonne information au bon moment, et comme la volumétrie des données connues pour être importante dans les entrepôts de données, le rôle central que joue lôutilisateur dans le processus décisionnel, au niveau de lôanalyse en ligne et les limités montrées par les systèmes OLAP quant la prise en compte de ses préférences sont les éléments qui justifient pleinement le recours à la personnalisation afin de lui offrir des informations pertinentes répondant a ses besoins attendus.

Lôobjectif principal, fixé au début du mémoire, était de concevoir et de réaliser un système dôinterrogation personnalisé dôun entrepôt de données. La modélisation et la gestion des préférences utilisateur, lôexploitation du profil utilisateur dans la transformation de ses requêtes, dans la manière avec laquelle sont présentés ces résultats sont les contributions principales apportées dans ce mémoire.

Afin de parvenir cet objectif, nous avons, au cours de notre étude bibliographique, passé en revue quelques notions théoriques entourant le sujet , et essayer de mieux exploiter les données de ces derniers.
Par la suite, lôétude sur les langages et outils disponibles qui permettent lôinterrogation des entrepôts de données nous a conduits à opter par la personnalisation des requêtes écrites en langage MDX. Vue que, côest le plus utilisé actuellement dans lôinterrogation des entrepôts de données.

Comme nous lôavons déjà signalé précédemment, la personnalisation dans le domaine des entrepôts de données est un axe émergent et fait lôobjet de recherches actives. De ce fait nous avons rencontré des difficultés au niveau de la modélisation du profil utilisateur (comme les préférences utilisateur sur les niveaux). Cependant, les objectifs tracés ont été atteints et il y a toujours place pour son amélioration et son enrichissement.

Pour ce qui est de la réalisation, nous avons pu réaliser tant bien que mal le prototype que nous avons conçu, à savoir lôacquisition des préférences utilisateur dôune manière explicites, la personnalisation de la requête MDX générée par utilisateur qui consiste à la transformer, exécution de celle -là, la présentation de ses résultats et sa réutilisation.

Enfin, notant que très peut de travaux sur la personnalisation de lôinformation dans le domaine des entrepôts de données. Et notre travail nôest quôune démarche initiale dans ce domaine et ne peut pas être qualifié de parfait alors les perspectives que nous proposons sont :

ule pour lôacquisition des préférences dôutilisateur dôune

- Le développement dôun module pour visualiser les résultats des requêtes personnalisées sous dôautres formes que celle des tableaux croisés par exemple sous forme de graphes, et histogrammes.
- Guider lôutilisateur pendant sont analyse en lu proposant des recommandations pour une bonne réutilisation de ses requêtes.
- Perfectionner les interfaces de notre application.
- Extension du système à ce quôil prenne en compte les contraintes des dispositifs dôinteractions (a savoir des téléphones portables, des PDAé)

Nous espérant que notre travaille bien médiocre soit-il, vous ai apporté un minimum.

Bibliographies

[1.1] : [Personnalisation des requêtes OLAP sous contraintes ,thèse Doctorat par HASSINA MOULOUDI ,2007]

[1.2] :[Personnalisation de données, mémoire dôingénieurs, DJEMAI Khadidja,GHOUALI KHADRA 2007,2008]

ey on the use of relevance feedback for information
5, 2003]

[1.4] : [L.Scime ,A.and Kershberg.Websifter :An ontology based personalizable search agent for the web , page 439-446, 2003]

[1.5] : [Access personnalisé à lôinformation: Vers un modèle base sur les diagrammes dôinfluence L.Tamine ï Lechani, M.Boughanem, C. Chrisment]

[1.6] : [J. Budzik et K .J Hamond.User interactions with every applications as context for just in time information access. page 44-51, 2000]

[1.7] : [J.Su et M.Lee. An exploration in personalized and context-sensitive search.2003]

[1.8] : [Kosba A. « Generic User Modeling Systems « Journal On User Modeling and User Adapted Interaction, vol.11,2001, P49-63]

[1.9] : [Cherniak et al.,2003]

[1.10] : [F.Bentayeb et al.]

[1.12] : [Personnalisation des requêtes OLAP sous contraintes ,thèse Doctorat par HASSINA MOULOUDI ,2007]

[2.1] : [Personnalisation de lôinformation : une approche de gestion de profils et de reformulation de requêtes ,thèse Doctorat par Dimitre Kostadiniv,2008]

[2.2] :[I.Ruthven et M.Lalams.A survey on the use of relevance feedback for information access system.volume 18, page 95-145, 2003]

[2.3] :[Bradley 00] Case Based User Profiling For Content Personalization.

[2.4]: [Personnalisation de lôinformation : une approche de gestion de profils et de reformulation de requêtes ,thèse Doctorat par Dimitre Kostadiniv,2008]

[2.5] :[Mobasher 00] Discovery and Evaluation of Aggregate Usage Profiles for Web Personalizaion, 2002.

[2.6] : [Pretschner 99] Ontology Based Personalized Search

[2.7] : [Croft 01] Relevance Feedback and Personalization ,DELOS Workshop,2001

[2.8] : [Shearin 01] Intelligent Profiling by example,2001

[2.9]: [Fink 00] A Review and Analysis of Commercial User Modeling Servers for Personalization on the World Wide Web,2000

[2.10]: [Amato 99] User Profile Modeling and Application to Digital Libreries,1999

ntrinsic Prefernces,2002

refered Item? ,2002

[2.13] : [BRU96] Brulisovsky P.1996.Methods and Techniques of Adaptative Hypermedia

[2.14] : [Chomicki 02] Quering with Intrinsic Prefernces,2002

[2.15] : [Personnalisation de lôinformation : une approche de gestion de profils et de reformulation de requêtes ,thèse Doctorat par Dimitre Kostadiniv,2008]

[3.1] : [Conception d'un entrepôt de données , thèse par Yazid Grim]

[3.2] : http://fr.wikipedia.org/wiki/entrepots_de _donnees

[3.3] : [Personnalisation des requêtes OLAP, thèse Magister par BOUCETHA lila,2010]

[3.4] : [Tuning des entrepôts de données, thèse DOCTORA par BOUKHELFA Kamel,2010]

[3.5] : [Conception et réalisation d'un outil de personnalisation pour l'interrogation d'un entrepôt de données, par MEZIANI Nadir, TEBANI Ali,2008/2009]

[3.6] :[FAVRE 07]

[3.7] :[JUNG et AL, 2004].

[3.8] : http://fr.wikipedia.org/wiki/OLAP

[3.9] : http://fr.wikipedia.org/wiki/MDX

[3.10] : [SPOFFORD, 2001]

[3.11] : [ISRAEL, 2001]

[3.12] : [GRAUA, 2004]

[3.13] [PEGUIRON, 2006].

[3.14] : [Conception et réalisation d'un outil de personnalisation pour l'interrogation d'un entrepôt de données, par MEZIANI Nadir, TEBANI Ali,2008/2009]

[3.15] :[MAR2006MASTER]

[3.16] :[SAM98, LB03, PLT07].

[3.17] [GMR98a]

[3.18] [THESE FAVRE]

et visualisation OLAP sous contraintes » thèse doctorat
LOUDI, 2007]

[4.2] :[Barbera et al.,2004]

[4.3] : [Das e tal.,2006]

[4.4] : [Koutrika and Ioannidis,2005a]

[5.1]: http://www.jmdoudoux.fr/java/dejae/chap001.htm#chap_1

[5.2] :www.commentcamarche.net/contents/servlets/servintro.php3

[5.3] :www.commentcamarche.net/contents/servlets/servcycle.php3

[5.4] :http://www.ulb.ac.be/cours/acohen/travaux_2004_infodoc/projettechnologiesweb/html/glossaire.html

[5.5] :http://www.enseignement.polytechnique.fr/informatique/profs/Julien.Cervelle/eclipse/#introTitle

[5.6] :http://www.journaldunet.com/encyclopedie/definition/java_database_connectivity.shtml

[5.7] http://fr.wikipedia.org/wiki/Apache_Tomcat

[5.7] :http://httpd.apache.org/docs/2.2/fr/mpm.html

[5.8] :www.apache.org

[5.9] www.apachetoday.com

[5.10] :www.apache-server.com

[5.11] :http://www.osbi.fr/?p=1335

[5.12] :http://www.oracle.com/

[5.13] :http://fadace.developpez.com/sgbdcmp/#LII-I-1

[5.14] : http://decisionnel-open-source.smile.fr/Les-composants-decisionnels/Talend

[6.1] : http://fadace.developpez.com/sgbdcmp/

[6.2]: http://www.serveur-dedie.com/serveur/serveurs/15-serveur-d-application.html

[6.3]: http://www.journaldunet.com/encyclopedie/definition/972/34/20/tomcat.shtml

smile.fr/les-composants-decisionnels/mondrian

[6.5]: http://java.sun.com/javame/index.jsp

[6.6] :[Rapport de stage Étude des ETL Open Source réalisé par Florian FRANCHETEAU]

[6.7] : [Romero and Abelló, 2007]