UNIVERSITE SIDI MOHAMED BN ABDLLAH
FACULTE DES SCIENCES ET TECHNIQUES
Département des Mathématiques

Master sciences et techniques

CALCUL SCIENTIFIQUE ET APPLICATIONS

MEMOIRE DE FIN D'ETUDES

Pour obtenir le diplôme de master sciences et techniques

Présenté par :
Aziza Salem

Soutenu le 17 janvier 2019

Titre :

Business intelligence

DATAWAREHOUSE, OLAP, CUBE OLAP

Mémoire dirigé par :
Mme, OUAFAE AMMOR, professeur (FST-FES)

JURY :

Mme. OUAFAE AMMOR, Professeur, (FST-FES)

Mme. EZZAKI FATIMA Professeur, (FST-FES)

Mme. RAHMOUNI HASSANI AZIZA, Professeur, (FST-FES)

M. HILALI ABDELMAJID, Professeur, (FST-FES)

Année universitaire : 2018 /2019

FACULTE DES SCIENCES ET TECHNIQUES FES_SAIS
B.P.2202_Route d'Imouzzer_FES
212(0)53561186_Fax :212(0)535608214
Site web :http://www.fst-usmba.ac.ma

Remerciements

Avant de vous présenter ce rapport, je tiens tout d'abord, à remercier tous ceux qui ont contribué de loin ou de près à la réalisation de ce projet de fin d'études. Je profite de cette occasion pour remercier Madame ***OUAFAE AMMOR****, pour tout le soutien, l'aide, l'orientation, la guidance qu'elle m'a apportés durant la réalisation de ce projet ainsi que pour ses précieux conseils et ses encouragements.*

Je remercie toute l'équipe pédagogique de département mathématiques et applications en particulier Mme. ***EZZAKI FATIMA****, Mme.* ***RAHMOUNI HASSANI AZIZA*** *et* ***M. HILALI ABDELMAJID*** *qui ont accepté de juger ce travail.*

Dédicaces

Je dédie ce mémoire…

À tous ceux qui me sont chers

A MES CHERS PARENTS

Que ce travail soit l'expression de ma reconnaissance pour vos sacrifices consentis, votre soutien moral.

A MON MARI

Pour votre soutien moral et matériel que vous n'avez cesse de prodiguer.

A MES FRERES, SŒURS, LEURS EPOUX ET LEURS ENFANTS

Pour votre soutien et encouragement, vous occupez une place particulière dans mon cœur. Je vous souhaite un avenir radieux, plein de bonheur et de succès.
A tous mes amies qui n'ont cessé de m'encourager et de me soutenir.

Table des matières

Introduction général

Avec l'apparition et le développement de nouveaux phénomènes économiques comme la mondialisation, les entreprises évoluent dans un environnement difficile à appréhender. Le marché évolue très rapidement, la concurrence est de plus en plus forte et les clients de plus en plus exigeants.
Pour les jeunes entreprises, la prise de décision stratégique, politique ou parfois opérationnelle devient cruciale. Aujourd'hui la qualité des décisions prises au sein d'une organisation dépend énormément de la performance de son système d'information.
Pour faire face à ces exigences, l'entreprise doit s'appuyer sur un ensemble d'informations pertinentes. Celles-ci sont à la portée de toute entreprise qui dispose d'un capital de données gérées par ses applications de production.
Mais dans leur état naturel, ces données sont surabondantes, éparpillées dans plusieurs systèmes hétérogènes et non organisées dans une perspective décisionnelle. Il devient donc capital de les rassembler et de les homogénéiser afin de les rendre pertinentes pour la prise de décisions.
Les nouvelles technologies de l'information et de la communication permettent de concevoir des systèmes d'information particulièrement novateurs avec un niveau de performance acceptable.
Pour répondre aux besoins de ses clients et des partenaires s'est donné comme priorité la maîtrise du nouvel outil qu'est le Business Intelligence.
Notre travail s'articulera autour de l'étude conceptuelle d'un système décisionnel et un exemple de manipulation du cube OLAP.

1. Introduction :

Toute organisation peut être décrite selon trois systèmes

– le système opérant représentant l'activité productrice de l'organisation qui consiste à transformer les flux primaires pour répondre aux besoins des clients,
– le système de pilotage correspondant à l'ensemble du personnel dirigeant qui régule, pilote et adapte l'organisation par leurs décisions,
– le système d'information permettant de collecter, conserver, traiter et restituer les données produites dans l'organisation.
Le système d'information assure « le lien » entre les systèmes de pilotage et opérant : le système opérant produit des informations stockées dans le système d'information, qui après traitements assure la transmission de ces informations au système de pilotage lui permettant ainsi de connaître l'activité opérationnelle. Les décisions prises sont répercutées vers le système opérant au travers du système d'information.
Face aux importants défis que doivent relever les organisations (concurrence, développement à l'international, émergence de technologies…), le pilotage réclame aujourd'hui des systèmes dédiés efficaces .

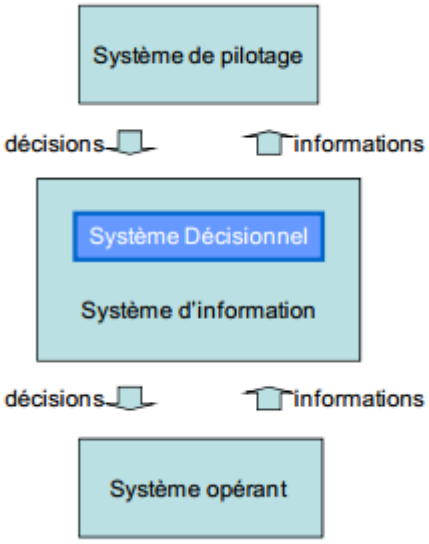


Figure 1 : Positionnement du système décisionnel dans l'organisation.

Afin de piloter au mieux l'entreprise, d'automatiser des processus de plus en plus complexes, de gérer un nombre de collaborateurs de plus en plus important et répartis sur plusieurs sites, les Progiciels de Gestion Intégrés (PGI ou ERP) se montrent de plus en plus efficaces pour faire face à cette complexification de l'organisation.

Cependant, ces outils génèrent de plus en plus de données, dans tous les domaines d'activité et il devient souvent difficile de se rendre compte de l'impact réel d'une décision sur la performance de l'organisation. Or le but de toute décision en entreprise en d'en améliorer la performance, soit en diminuant les coûts ou en augmentant les ventes... Comment savoir que la décision prise est la bonne, quelles sont les contraintes soumises aux problèmes ? Tout problème de décision est soumis à un jeu d'actions et de critères régis par des contraintes. Comment être sûr que tous ces éléments ont été bien pris en compte et sont maîtrisés ? Tel est la tâche du décideur ou du manager en entreprise qui doit tenter d'identifier, de distinguer et s'il le peut de maîtriser les éléments d'un problème de prise de décisions.

2. Problématique :

La prise de décision est un problème central dans les entreprises.
Les décisions concernent différents types d'activités : on peut ainsi distinguer les décisions commerciales, administratives, financières. Les décisions les plus importantes sont :

- les décisions de financement (par exemple, réaliser une augmentation de capital),
- les décisions d'exploitation (par exemple, établir le programme de production de l'année),
- les décisions d'investissement (par exemple, construire une nouvelle usine).

Mais le problème de prise de décision est complexe

- Grand nombre de facteurs
- Structuration du problème (problèmes mal définis), considérations subjectifs et conflits d'intérêt
- Incertitude

3. Notion de système d'information

Un système d'information est un ensemble organisé de ressources (matérielles, logicielles, personnelles, données, procédures...) permettant d'acquérir, de traiter, de stocker des informations (sous forme de données, textes, images, sons...) dans et entre organisations ». Le choix de l'appellation système n'est pas anodin. Il reflète la logique sous-jacente considérant ce dernier comme un ensemble d'entités en interaction entre elles, que l'on pourrait considérer comme autant de maillons formant une chaîne. De ce fait, ce dernier peut être ainsi observé à différents degrés de précision, soit en le considérant comme un système d'information global, soit en accentuant le zoom afin de mettre en valeur deux sous-systèmes.

4. Deux systèmes d'information : transactionnel et décisionnel

D'une part le système d'information transactionnel. Il gère les applications quotidiennes et se rapproche à ce titre de la couche opérationnelle. Il est typiquement utilisé par les acteurs métiers et se voit plus comme un outil utilisé par ces derniers afin de répondre à des besoins de simplification et d'automatisation.
D'autre part le système d'information décisionnel, angle d'approche de cet ouvrage, qui est utilisé pour prendre les décisions de l'entreprise, et à ce titre doit permettre aux décideurs d'avoir un certain recul sur leur entreprise. Il fournit pour cela les informations nécessaires et

pertinentes afin de faire les bons choix. Le Gartner Group définit, en 1993, la Business Intelligence comme «l'ensemble des moyens et méthodes permettant de rassembler, consolider, analyser et rendre accessible les données d'une entreprise dans une perspective d'aide à la décision ». Le décisionnel est donc à l'information de l'entreprise ce que les mathématiques sont à la pensée.
Force est de constater que le concept de Business Intelligence n'est pas récent, et que, depuis sa création, des évolutions notables peuvent être distinguées. Il est nécessaire de connaître ces mutations afin de bien saisir les tenant et aboutissant de leur structure actuelle.

Le tableau suivant résulte la comparaison entre le décisionnel et le transactionnel :

Caractéristique	Système d'information transactionnelle	Système d'information décisionnelle
Objectif	Gestion courante, production	Analyse, support à la décision
Modèle de données	Entité/relation	Etoile, flocon de neige
Normalisation	Fréquente	Plus rare
Données	Actuelles, brutes	Historisées, parfois agrégées
Mise à jour	Immédiate, temps réel	Souvent différée
Niveau de consolidation	Faible	Elevé
Perception	Bidimensionnelle	Multidimensionnelle
Opérations	Lectures, mises à jour, suppressions	Lectures, analyses croisées, rafraîchissements
Taille	En giga-octets	En téraoctets

Figure 2 : la comparaison entre le décisionnel et le transactionnel

5. Système d'information décisionnel :

Le Système d'Information Décisionnel (SID) est un outil d'observation et de description qui, à partir des données de l'entreprise regroupées dans l'entrepôt de données, va donner aux managers les moyens d'identifier des alertes de gestion, de suivre l'évolution de l'activité et de disposer d'outils d'investigation de sujets ou phénomènes particuliers. Les objectifs du SID sont de procurer une présentation synthétique des données de l'entreprise, une consultation plus facile pour minimiser la recherche d'informations et la présentation des résultats, mais aussi de présenter uniquement les informations utiles et donc de paramétrer les statistiques qui seront utilisées par chacun des groupes d'utilisateurs.
Le SID permet de répondre aux objectifs en fournissant :

- Un tableau de bord comportant des alertes.
- Des tableaux préformatés contenant l'essentiel de la statistique d'activité et d'environnement de l'entreprise.
- Des tableaux et des graphiques qui restituent les résultats suite à des interrogations en utilisant la technologie « hypercube ».
- La restitution d'analyses sophistiquées tel que l'analyse de corrélation, la simulation… en utilisant des outils de Data Mining.

Nous définissons le système décisionnel comme le système dédié au support de la prise de décision (pilotage). Il regroupe l'ensemble des outils informatiques permettant d'extraire et de transformer (E.T.L.), de stocker (S.G.B.D.), d'analyser et de restituer les données décisionnelles d'une organisation.

Le SID a pour vocation de fournir des indicateurs de pilotage qui permettent à un responsable opérationnel d'évaluer la qualité et la productivité du travail fourni par des équipes ou des

structures en fournissant des données observées et recoupées avec d'autres sources afin qu'il y ait une meilleur compréhension du marché pour un suivi de l'activité et l'analyse de son impact, l'optimisation des moyens. Mais le SID ne fournit pas des indicateurs pour un pilotage au jour le jour ou un suivi individuel.

Voici comment se présente un Système d'Information Décisionnel :

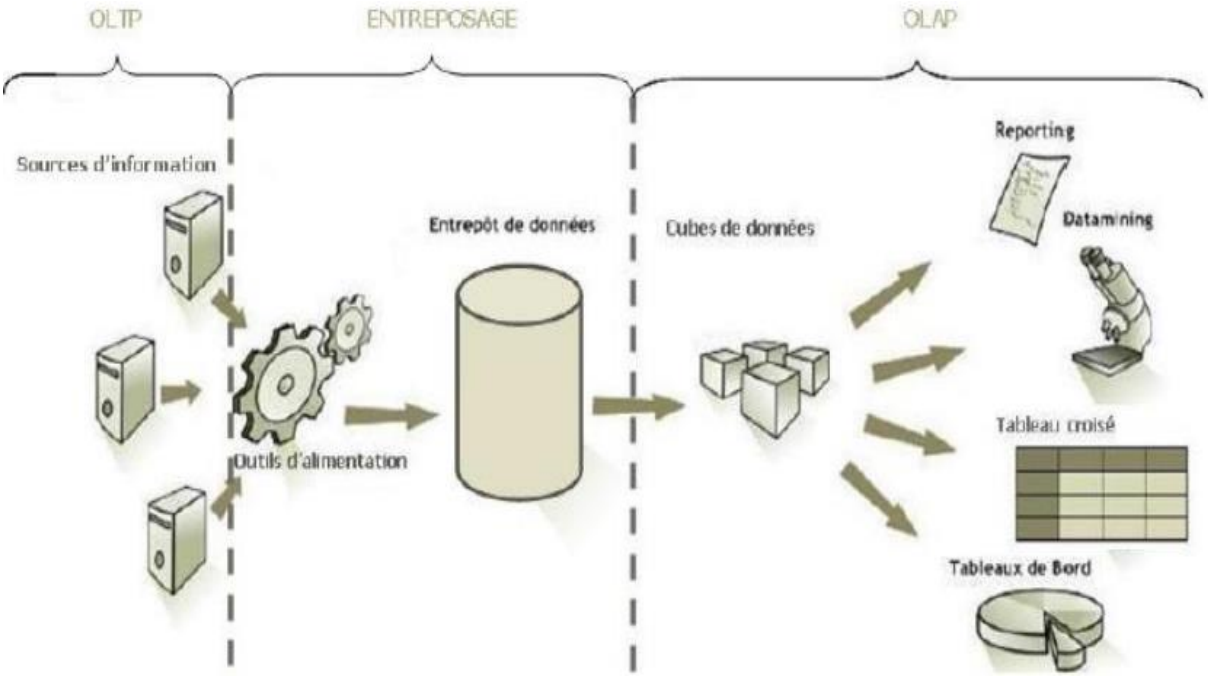


Figure 3 : Architecture d'un système décisionnel

L'informatique décisionnelle en anglais« Business intelligence», parfois appelé tout simplement« le décisionnel» gère l'exploitation des données de l'entreprise dans le but de faciliter la prise de décision par les décideurs, c'est-à-dire la compréhension du fonctionnement actuel et l'anticipation des actions pour un pilotage éclairé de l'entreprise.

Le système d'information décisionnel est un ensemble de données organisées de façon spécifique, facilement accessible et appropriées à la prise de décision ou encore une représentation intelligente de ces données au travers d'outils spécialisés. La finalité d'un système décisionnel est le pilotage de l'entreprise.

6. Historique de la Business Intelligence

Au début des années 90, l'informatique est au service de l'entreprise pyramidale. D'une manière très classique, elle remonte les informations de la base vers le haut. Cette époque est celle des **Executive Information Systems** (EIS).
Milieu des années 90, les besoins d'informations composites révèlent des lacunes dans les systèmes d'informations. Les technologies Data Warehouse et Data Mart se banalisent et l'informatique décisionnelle se tourne vers les cubes OLAP, dans un souci d'analyse plus poussée.

De nos jours, le décisionnel n'est plus l'apanage des instances dirigeantes et toutes les couches de l'entreprise revendiquent un besoin d'information pertinente, propre à leur fonction. Que ce soit dans des soucis de pilotage par les acteurs du top management, pour des besoins particuliers formulés par des experts ou dans des logiques de reporting classique demandées par les acteurs métiers, cette mutation culturelle s'appuie sur la banalisation et l'accessibilité des technologies Web, qui rendent cette divulgation d'information possible à moindre coûts.
Force est de constater également que certaines règles conceptuelles se sont inconsciemment standardisées, et actuellement le système d'information décisionnel peut être schématisé sous trois étapes.

☑ Règles conceptuelles

Tout d'abord, l'extraction des données. L'entreprise étant composée d'informations aussi variées en termes de structure, de format, de taille... le système se doit d'extraire les informations afin de les amener vers la deuxième étape.
Ensuite, la consolidation. Ces données doivent être consolidées afin de pouvoir effectuer le travail nécessaire dessus.
Enfin le traitement. Il doit fournir aux dirigeants les informations pertinentes sous forme d'indicateurs, tout en répondant aux questions que toute mise en place doit se poser : Quelles informations ? Sous quelle forme ? Tous les combiens ?...

Datawarehouse

1. Définition de datawarehouse :

Un datawarehouse (ou entrepôt de données) est un serveur informatique dans laquelle est centralisé un volume important de données consolidées à partir des différentes sources de renseignements d'une entreprise (notamment les bases de données internes). L'organisation des données est conçue pour que les personnes intéressées aient accès rapidement et sous forme synthétique à l'information stratégique dont elles ont besoin pour la prise de décision.

Le Datawarehouse est une collection de données orientées sujet, intégrées, non volatiles et historiées, organisées pour le support d'un processus d'aide à la décision

Le Datawarehouse, ou entrepôt de données, est une base de données dédiée au stockage de l'ensemble des données utilisées dans le cadre de la prise de décision et de l'analyse décisionnelle. Le Datawarehouse est exclusivement réservé à cet usage. Il est alimenté en données depuis les bases de production grâce notamment aux outils d'ETL Extract Transform Load.

Si un entrepôt de données utilise le principe des bases de données relationnelles, il s'en distingue par de nombreux points. Tout d'abord, il n'applique pas un modèle relationnel précis, car les tables n'ont pas toujours une structure commune. Les entrepôts de données servent justement à croiser des informations a priori non liées directement (exemple : rattacher les informations des systèmes de production avec celles du support client pour en tirer des requêtes qui font sens).

2. Les caractéristiques du data warehouse:

Le Datawarehouse n'est pas une simple copie des données de production. Le datawarehouse est organisé et structuré.

Père du concept, Bill Immon dans son livre "Building the Datawarehouse" (John Wiley and Son 1996) le décrit ainsi :

"Subject oriented, integrated, nonvolatile, time variant collection of data in support of management decisions"

Figure 4 : Caractéristiques des données d'un DW

2.1. Données sont orientées sujet

Au coeur du Data warehouse, les données sont organisées par thème. Les données propres à un thème, les ventes par exemple, seront rapatriées des différentes bases OLTP de production et regroupées.

Les données sont orientées « métiers » ou business (par exemple, pour une banque un compte débiteur sera agrégé avec les prêts accordés par la banque et non pas avec les autres comptes restés créditeurs, à la différence de ce qui se passe dans la comptabilité et le système de production d'origine). L'objectif d'un data warehouse est la prise de décisions autour des activités majeures de l'entreprise. Dans un data warehouse, les données sont ainsi structurées par thèmes par opposition à celles organisées, dans les systèmes de production, par processus fonctionnel. L'intérêt de cette organisation est de disposer de l'ensemble des informations utiles sur un sujet le plus souvent transversal aux structures fonctionnelles et organisationnelles de l'entreprise. On peut ainsi passer d'une vision verticale de l'entreprise à une vision transversale beaucoup plus riche en informations. On dit que le data warehouse est orienté « métier », en réponse aux différents métiers de l'entreprise qu'il est censé préparer à l'analyse.

- Données structurées par thèmes (sujets majeurs de l'entreprise) et non suivant les processus fonctionnels.
- Le sujet est transversal aux structures fonctionnelles et organisationnelles de l'entreprise. On peut accéder aux données utiles sur un sujet.
- L'intégration des différents sujets se fait dans une structure unique.
- Il n'y a pas de duplication des informations communes à plusieurs sujets.
- La base de données est construite selon les thèmes qui touchent aux métiers de l'entreprise (clients, produits, risques, rentabilité, …).
- Les données de base sont toutefois issues des Systèmes d'Information Opérationnels (SIO).

2.2. Les données sont intégrées :

Les données proviennent de sources hétérogènes utilisant chacune un type de format. Elles sont intégrées avant d'être proposées à utilisation.

Les données sont intégrées en provenance de sources hétérogènes ou d'origines diverses (y compris des fichiers externes de cotation ou de scoring). Dans un monde idéal, les systèmes d'informations sources (systèmes de production) sont homogènes et l'entreprise dispose de la connaissance parfaite de toutes les codifications dont elle a besoin pour tirer parti de son capital

informationnel. Dans la réalité, les données, issues de différentes applications de production, existent sous des formes différentes. Il s'agit alors de les intégrer afin de les homogénéiser et de leur donner un sens unique, compréhensible par tous les utilisateurs. La transversalité recherchée sera d'autant plus efficiente que le système d'information sera réellement intégré. Cette intégration nécessite une forte normalisation, une bonne gestion des référentiels et de la cohérence, une parfaite maîtrise de la sémantique et des règles de gestion s'appliquant aux données manipulées. Elle concerne des données internes mais aussi des données externes qui posent des problèmes car leur codification et leur niveau de détail différent de ceux des données internes. Ce n'est qu'au prix d'une intégration « réussie » que l'on peut offrir une vision homogène et cohérente de l'entreprise via ses indicateurs. Ceci suppose que le système d'information de l'entreprise soit déjà bien structuré, bien maîtrisé, et bénéficie d'un niveau d'intégration suffisant. Si tel n'était pas le cas, la qualité des données peut empêcher la bonne mise en œuvre du datawarehouse.

- Les données, issues de différentes applications de production, peuvent exister sous toutes formes différentes.
- Il faut les intégrer afin de les homogénéiser et de leur donner un sens unique, compréhensible par tous les utilisateurs.
- Elles doivent posséder un codage et une description unique.
- La phase d'intégration est longue et pose souvent des problèmes de qualification sémantique des données à intégrer (synonymie, homonymie, etc…).
- Ce problème est amplifié lorsque des données externes sont à intégrer avec les données du SIO.

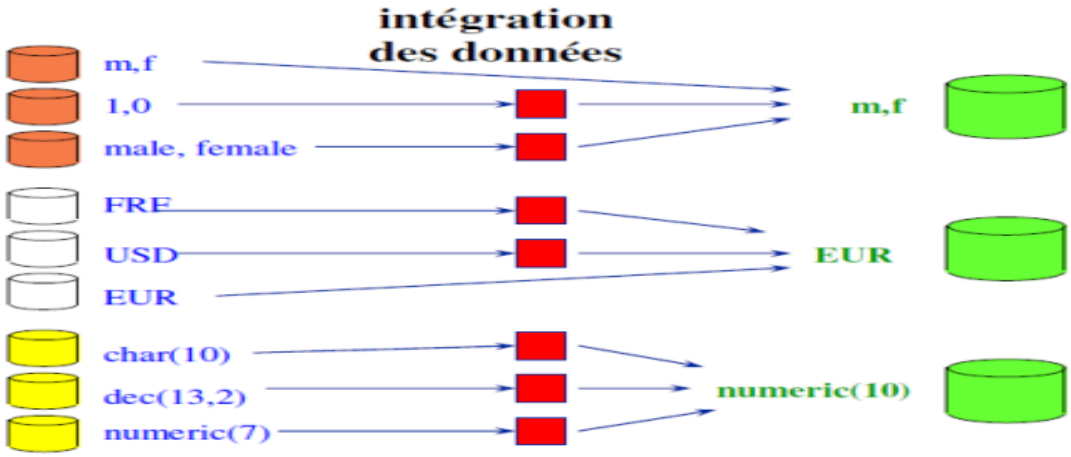


Figure 5 : exemple d'intégration des données

2.3. Les données sont non-volatiles

Les données ne disparaissent pas et ne changent pas au fil des traitements, au fil du temps (Read-Only).

Les données sont stables, en lecture seule, non modifiables. Afin de conserver la traçabilité des informations et des décisions prises, les informations stockées au sein du data warehouse ne doivent pas disparaître. Une même requête lancée plusieurs fois, et c'à des mois d'intervalle, sur une même population doit restituer les mêmes résultats. Ainsi, dès lors qu'une donnée a été qualifiée pour être introduite au sein du data warehouse, elle ne peut ni être altérée, ni modifiée, ni supprimée (ou en tout cas en deçà d'un certain délai de purge). Elle devient, de fait, partie prenante de l'historique de l'entreprise. Cette caractéristique diffère de la logique des systèmes de production qui bien souvent remettent à jour les données par « annule et remplace » à chaque nouvelle transaction.

- Une information est considérée volatile quand les données sont régulièrement mises à jour comme dans les Systèmes d'Information Opérationnels.
- Dans un SIO, les requêtes portent sur les données actuelles. Il est difficile de retrouver un ancien résultat.
- Dans un DW, il est nécessaire de conserver l'historique de la donnée. Ainsi, une même requête effectuée à deux mois d'intervalle en spécifiant la date de référence de la donnée, donnera le même résultat.

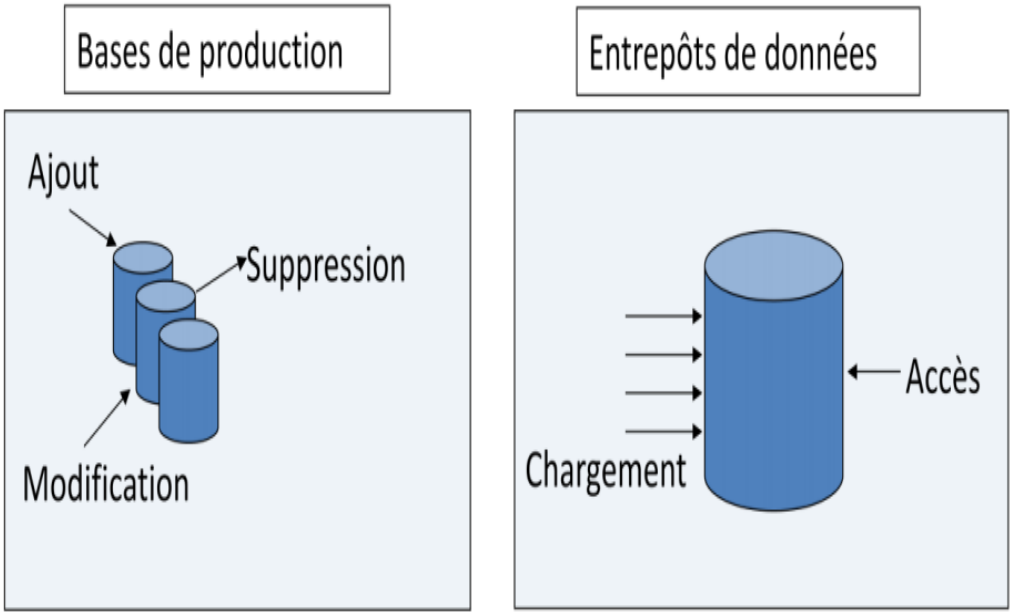


Figure 6 : exemple des données volatiles et non-volatiles

2.4. Les données sont historisées :

Les données non volatiles sont aussi horodatées. On peut ainsi visualiser l'évolution dans le temps d'une valeur donnée.

Le degré de détail de l'archivage est bien entendu relatif à la nature des données. Toutes les données ne méritent pas d'être archivées.

Les données, alors, sont archivées et donc datées : avec une conservation de l'historique et de son évolution pour permettre les analyses comparatives (par exemple, d'une année sur l'autre, etc.). La non-volatilité permet l'historisation. D'un point de vue fonctionnel, cette propriété permet de suivre dans le temps l'évolution des différentes valeurs des indicateurs à analyser. De fait, dans un datawarehouse un référentiel de temps est nécessaire. C'est l'axe temps ou période.

- Dans un SIO, les transactions se font en temps réel, et les données sont mises à jour constamment. L'historique des valeurs de ces données est conservé car elles sont inutiles.
- Dans un DW, la donnée n'est jamais mise à jour.
- Les données du DW s'ajoutent aux données déjà engrangées.
- Le DW stocke donc l'historique des valeurs que la donnée aura prises au cours du temps.
- Un référentiel de temps est alors associé à la donnée afin d'être capable d'identifier une valeur particulière dans le temps.
- Les utilisateurs possèdent un accès aux données courantes ainsi qu'à des données historisées.

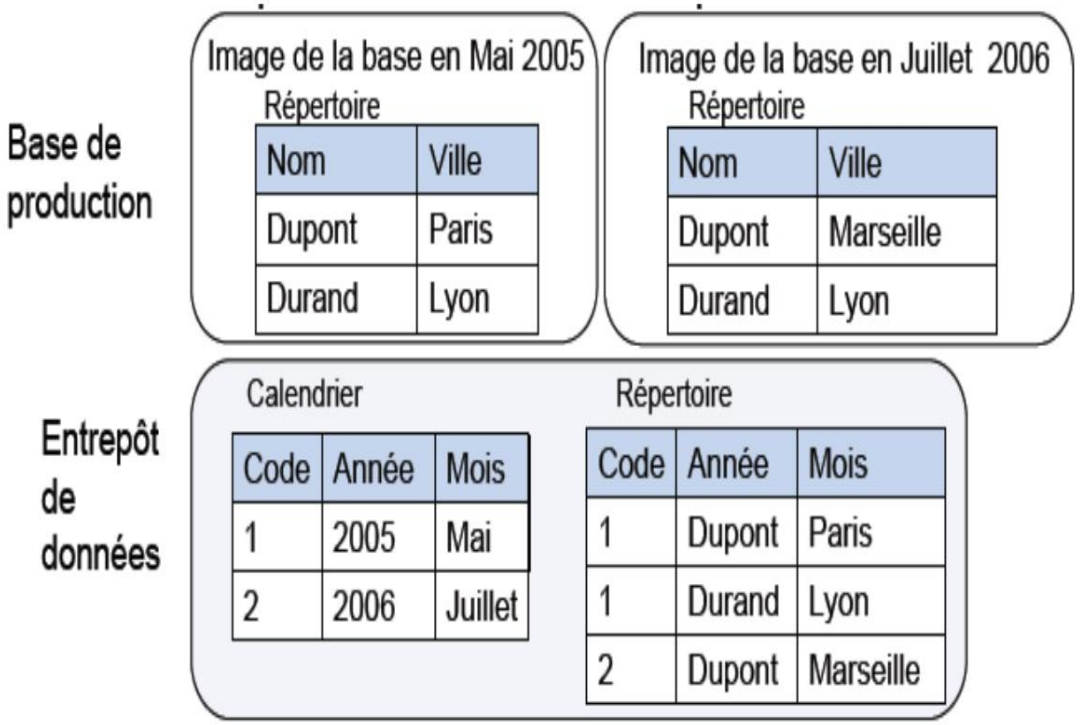


Figure 7 : exemple des données historisées

3. Objectif du data warehouse :

- Permet le développement d'applications décisionnelles et de pilotage de l'entreprise et de ses processus
- Joue un rôle de référentiel pour l'entreprise puisqu' 'il permet de fédérer des données souvent éparpillées dans différentes bases de données
- Offre une vision globale et orientée métiers de toutes les données que manipule l'entreprise
- Permet de faire face aux changements du marché et de l'entreprise
- Offre une information compréhensible, utile et rapide
- Permet l'intégration de différentes bases de données opérationnelles;
- Permet l'accès aux informations historiées;
- Fournir des outils d'analyse sur ces données;
- Résumer les données;
- Réconcilier des données inconsistantes.

4. Architecture de datawarehouse :

Ce schéma représente l'architecture générale du datawarehouse :

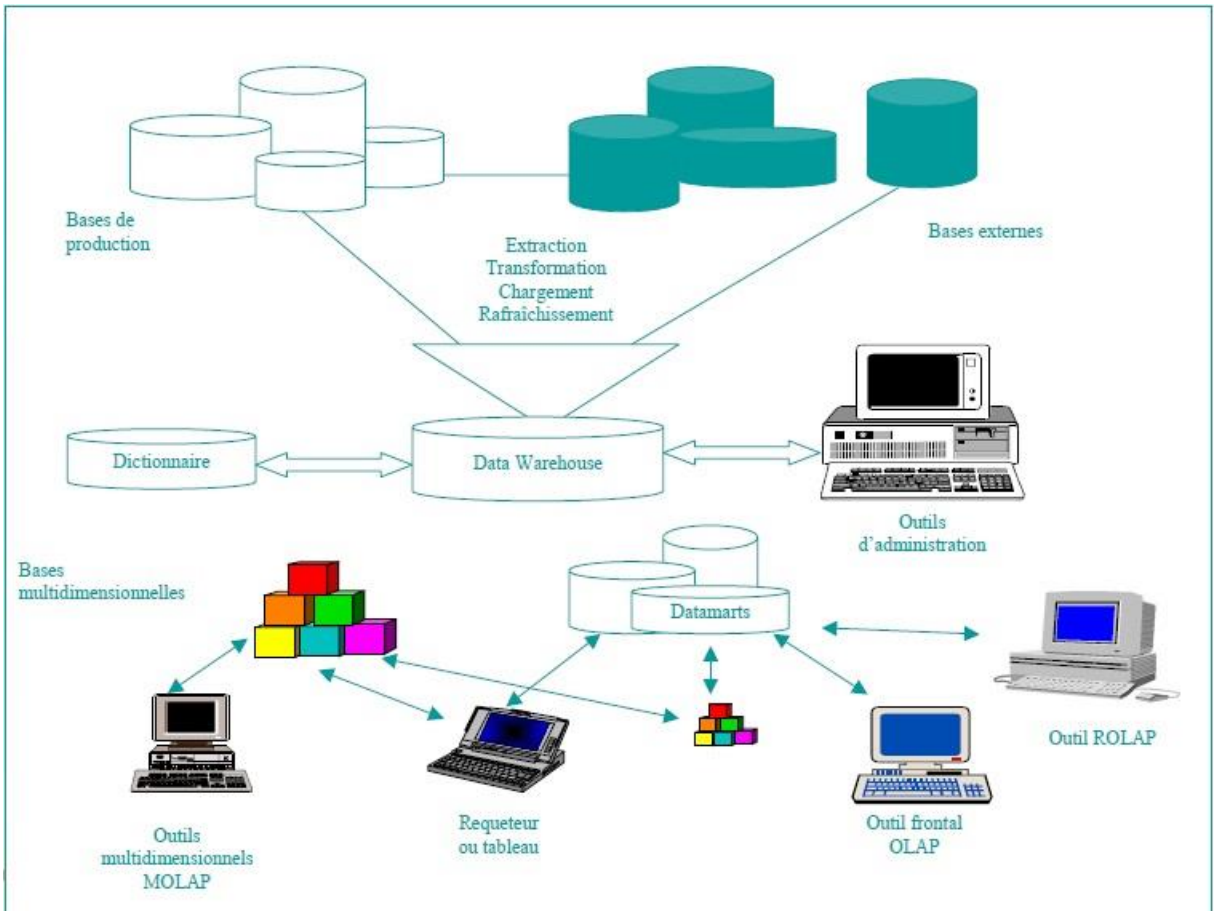


Figure 8 : l'architecture générale du datawarehouse

- Les bases de données :
  - ✓ Bases de production de l'entreprise
  - ✓ Bases créées par les utilisateurs
  - ✓ Bases de données externes à l'entreprise (Nielsen, INSEE, …) qui nécessitent leur identification, leur rapatriement et leur intégration.
- Opérations sur les données
  - ✓ Extraction :
    - ❖ Extraire les données de leur environnement d'origine (bases de données relationnelles, fichiers plats, …).
    - ❖ Utiliser une technique appropriée pour n'extraire que les données nécessaires : données créées ou modifiées depuis la dernière opération d'extraction.
  - ✓ Transformation :
    - ❖ Une même donnée peut avoir une structure ou une valeur différente en fonction de la base (production, externe, utilisateurs) dont elle provient.

- ❖ On peut être confronté à des redondances (un même client peut apparaître avec différents attributs et propriétés selon la source consultée).
- ❖ Il faut supprimer certaines données aberrantes qui risqueraient de fausser les analyses.
- ❖ Il faut donc épurer et transformer les données.

✓ Chargement / rafraichissement

- ❖ Effectuer sur les données des opérations de calcul et d'agrégation.
- ❖ Remplacer certaines bases si aucune solution d'extraction satisfaisante n'est possible.
- ❖ Mettre en place des procédures de chargement (nocturnes?) et de restauration (en cas de problème).
- ❖ Si la disponibilité du système ne peut être interrompue, envisager la mise en place de systèmes redondants.

✓ Les outils :

- ❖ On peut automatiser tout ou partie des opérations décrites.
- ❖ Des outils sont disponibles : Extract d'ETI, Genio de Leonard's Logic, …
- ❖ Le développement d'outils spécifiques est envisageable mais risque d'alourdir les tâches.

- Dictionnaire de données :
  - ✓ Le dictionnaire de données regroupe les méta-données.
  - ✓ Une méta-donnée représente une donnée sur les données. Il s'agit de l'ensemble des informations qui permettent de qualifier une donnée, notamment par sa provenance, sa qualité, etc…
  - ✓ les méta-données permettent de préciser de quelle table provient la donnée,à quelles dates et heures elle en a été extraite, l'état de la base à cet instant, etc...
  - ✓ Une méta-donnée permet de « remonter la chaîne » et de reconstituer l'ensemble d'événements et données qui ont servi à obtenir l'information associée.
  - ✓ Le dictionnaire de données contient toutes les informations permettant d'exploiter les données.
  - ✓ C'est un référentiel destiné aux utilisateurs et à l'administrateur du DW.
  - ✓ A ce jour, il n'existe pas de normes en ce qui concerne la structure et la gestion des dictionnaires de données. Chaque outil propose sa solution et son approche.
- Les data-marts

Un datamart (ou magasin de données) est une vue partielle du datawarehouse mais orientée métier. C'est un sous-ensemble du datawarehouse contenant des informations se rapportant à un secteur d'activité particulier de l'entreprise ou à un métier qui y est exercé. Il se situe en aval du datawarehouse et est alimenté par celui-ci. On peut donc créer plusieurs datamart correspondant au différent besoin des utilisateurs.
Cela permet de réduire le nombre d'opération sur les bases de production. De plus cela permet d'offrir aux utilisateurs un outil spécifiquement adapté à leurs besoins. Cet outil sera plus petit et permettra donc un accès plus rapide à l'information.

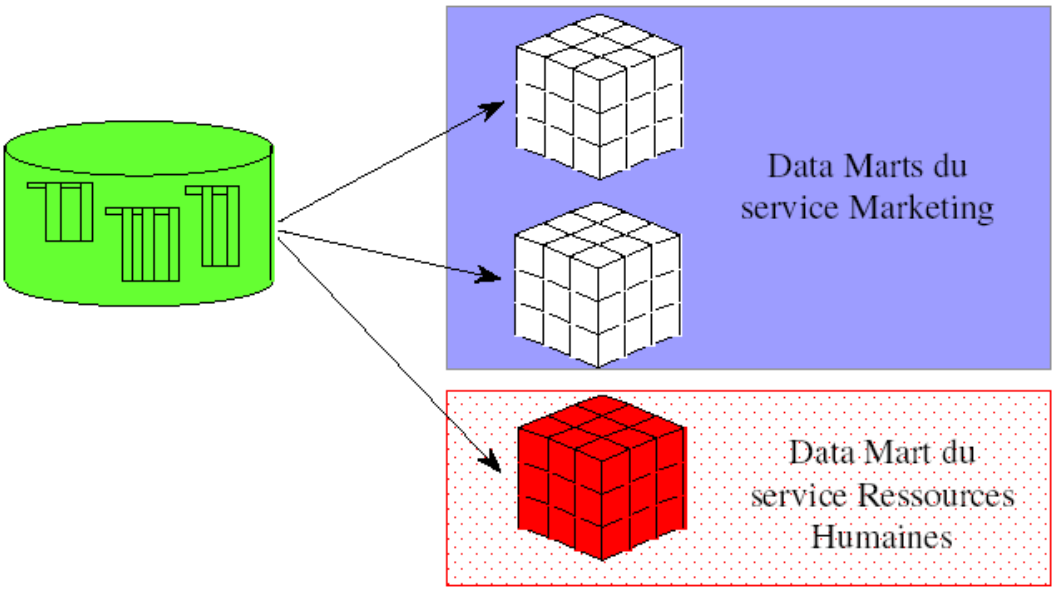


Figure 9 : datamart

- ✓ Un data-mart est un DW focalisé sur un sujet particulier, souvent au niveau départemental ou métier.
- ✓ C'est donc un mini DW lié à un métier particulier de l'entreprise (finance, commercial, …).
- ✓ Un DW est souvent volumineux (plusieurs centaines de Go voire quelques To) avec des performances inappropriées (temps de réponse trop longs). Un Data-mart, quant à lui, comporte moins de 50 Go, ce qui permet des performances acceptables.
- ✓ La création d'un data-mart peut être un moyen de débuter un projet de DW (projet pilote).

- Les bases multidimensionnelles et les outils OLAP

L'analyse multidimensionnelle est la capacité à analyser des données qui ont été agrégées suivant plusieurs dimensions. On veut donc accéder aux données déjà agrégés selon les besoins de l'utilisateur, de façon simple est rapide.
On utilise pour cela des hyper cubes OLAP. Les données sont représentées dans des hyper cubes à n dimensions. Les données sont structurées suivant plusieurs axes d'analyses (dimensions) comme le temps, la localisation ...
Une cellule est l'intersection des différentes dimensions. Le calcul de chaque cellule est réalisé au chargement. Le temps de réponse est ainsi stable quel que soit la requête.

- ✓ Les modèles de données

- ❖ Le modèle d'intégration unifie les données opérationnelles.
- ❖ Le modèle de diffusion représente le modèle conceptuel des données. Il correspond aux bases multidimensionnelles (serveur OLAP).
- ❖ Le modèle de présentation est un complément au modèle conceptuel. C'est à travers ce modèle que l'utilisateur voit les données. Il correspond à différents outils physiques : les tableurs, les requêteurs, les outils clients OLAP, etc...

- ✓ Les outils OLAP (On-Line Analytical Processing)
  - ❖ OLAP caractérise l'architecture nécessaire à la mise en place d'un système d'information décisionnel.
  - ❖ OLAP s'oppose à OLTP (On-Line Transactional Processing) qui caractérise les SIO.
  - ❖ OLAP constitue l'ensemble des outils multidimensionnels nécessaires à l'accès, le stockage et la manipulation des données utiles pour un SID ou pour un EIS.
  - ❖ OLAP désigne les outils d'analyse s'appuyant sur les bases de données multidimensionnelles.

Caractéristiques	OLTP	OLAP
Utilisation	SGBD (base de production)	Datawarehouse
Opération typique	Mise à jour	Analyse
Type d'accès	Lecture écriture	Lecture
Niveau d'analyse	Elémentaire	Global
Quantité d'information échangées	Faible	Importante
Orientation	Ligne	Multidimension
Taille BD	Faible (max qq GB)	Importante (pouvant aller à plusieurs TB).
Ancienneté des données	Récente	Historique

Figure 10 : OLTP vs OLAP

Les règles des bases multidimensionnelles :

- Vue multidimensionnelle : Les données sont structurées en dimensions métiers.
- Transparence : L'utilisateur doit pouvoir utiliser les logiciels habituels (tableurs, …) sans percevoir la présence d'un outil OLAP.
- Accessibilité : L'outil doit se charger d'accéder aux données stockées dans n'importe quel type de bases de données (interne + externe) et le faire simultanément.
- Performance continue dans les restitutions : A mesure que le nombre de dimensions ou la taille de la base augmente, l'utilisateur ne doit pas subir de baisse sensible de performance.
- Architecture client-serveur : Tout produit OLAP doit fonctionner en mode C/S avec une répartition des traitements.

- Dimension générique : Chaque dimension (avec l'analyse) doit être équivalente aux autres à la fois dans sa structure et dans ses capacités opérationnelles. Une seule structure logique dans l'ensemble des dimensions.
- Gestion dynamique des matrices creuses : OLAP doit gérer les cellules non renseignées de manière optimale.
- Support multiutilisateurs : OLAP doit assurer un accès simultané aux données, gérer l'intégrité et la sécurité de ces données.
- Opérations entre les dimensions : OLAP doit gérer des calculs associés entre les dimensions sans faire appel à l'utilisateur pour définir le contenu de ces calculs
- Manipulation intuitive : Minimiser le recours à des menus ou les allers et retours avec l'interface utilisateur
- Flexibilité des restitutions : convivialité des états de gestion ou des états de sortie - ergonomie
- Nombre de dimensions et niveaux de hiérarchie illimité : l 'outil doit gérer au moins quinze dimensions et ne pas limiter le nombre de niveaux hiérarchiques.

5. Les critères d'un datawarehouse performant :

Parvenir à fournir des informations clés aux décideurs si possibles "à la volée". Cela implique non seulement d'avoir extrait ces informations, de s'être assuré qu'elles soient valides et fraîches, mais aussi que les requêtes qui en découlent s'exécutent rapidement. Afin d'établir des statistiques d'évolution, ou de construire des plans, les entrepôts de données conservent généralement un historique des données. Ajouté à cela la diversité des sources, cela provoque des bases de taille colossale, de quelques centaines de giga-octets à plusieurs dizaines de téraoctets.
Un gros travail s'avère donc nécessaire pour optimiser la base de données, notamment en travaillant sur les index, la gestion des doublons, les procédures d'extractions et de transformation des données mais aussi sur la création de petits îlots optimisés, appelés datamarts.

6. Datamarts et datamining :

Le datamart est une extraction d'une partie d'un entrepôt de données pour répondre à une application dédiée (ex : le règlement de contentieux chez une banque). Pour ne pas multiplier l'espace de stockage réservé à l'entrepôt de données, la création de datamarts est souvent limitée. Cependant, elle permet de gagner parfois en efficacité sur le temps d'exécution des requêtes SQL.
Les datamarts sont aussi souvent utilisés lorsqu'une entreprise ne peut plus multiplier les optimisations sur son entrepôt de données sans pénaliser d'autres applications. Elle crée alors un nouvel environnement dédié à cette nouvelle application dont elle peut gérer librement les index. Le datamining regroupe toutes les solutions à même de piocher dans des données éparses pour en tirer des informations d'aide à la décision.

1. Qu'est-ce qu'un ETL :

1.1. Définition d'un outil ETL :

À présent que nous avons vu les grands principes de l'informatique décisionnelle, nous pouvons nous attarder sur les outils ETL. À l'origine, le principe est simple : il s'agit d'alimenter les entrepôts de données. Maintenant, les ETL se sont largement diversités et permettent d'effectuer de nombreuses opérations que nous verrons par la suite. Concrètement, on dispose de sources (souvent hétérogènes) que l'on extrait pour alimenter un entrepôt servant à leurs analyses. Les sources peuvent aussi bien être des bases de données (de n'importe quel SGBD), des fichiers (CSV, XML, Excel) et voire d'autres formats (annuaires LDAP, Web services). Les ETL s'occupent de transformer ces sources, via de nombreux composants, en une ou plusieurs cibles qui peuvent être, là aussi, de n'importe quels formats.

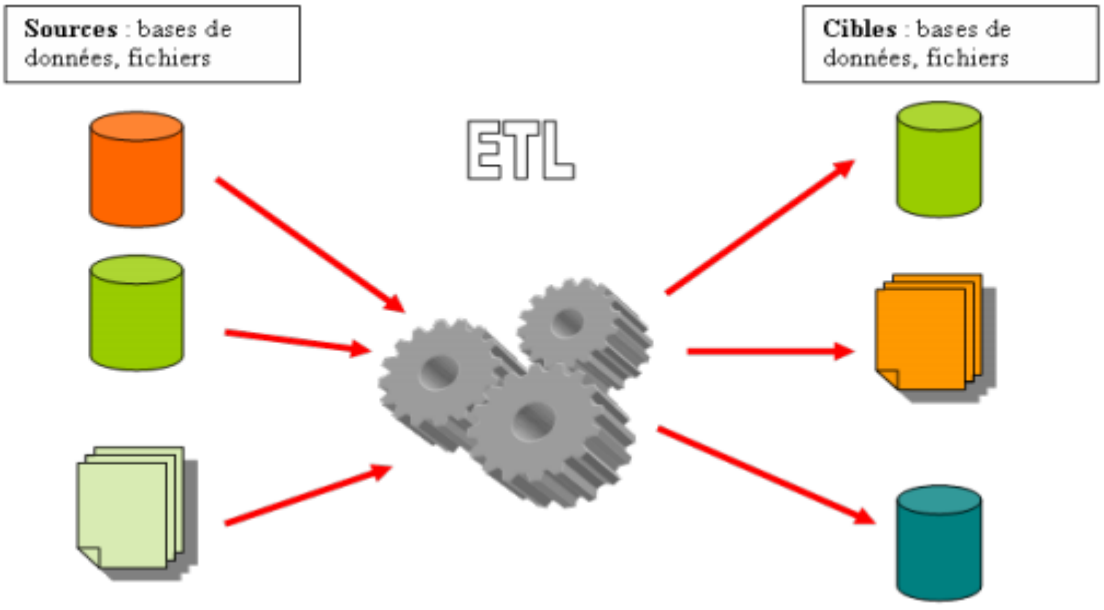


Figure 11 : la fonction d'un outil ETL

1.2. Extraction, Transformation, Load :

Comme expliqué précédemment, ETL signifie Extract, Transform, Load (Extraction, Transformation, Chargement). Ce sont les trois étapes que doit impérativement implémenter un outil ETL. Effectuées dans l'ordre, elles forment un traitement, une tâche ou un scénario (selon les diverses appellations des logiciels). Nous allons à présent les détailler.

1.2.1 Extraction

La première étape concerne l'extraction des données qui sont la plupart du temps hétérogènes. Cela signifie qu'elles peuvent provenir de SGBD (MySQL, Oracle, SQL Server, etc.), de fichiers plats (Txt, Excel, XML, etc.), d'ERP (Enterprise Ressource Planning), de bases hiérarchiques (les ancêtres des SGBD) ou d'autres applications spécifiques. On peut déjà remarquer le premier obstacle aux ETL : la multitude de formats sources possibles à gérer.

Cette étape doit permettre de se connecter aux bases, soit de façon native, soit via

JDBC/ODBC ou encore avec des connecteurs spéciaux.

Il est aussi important, lorsque l'on extrait des données, de pouvoir les analyser. Il faut donc connaître les propriétés de celles-ci : savoir si par exemple cette donnée est de type entier ou chaîne de caractères et quelle est sa taille maximale. Cela peut paraître simple lorsque la source provient d'un SGBD mais s'avère plus complexe lorsqu'elle provient d'un fichier plat. Il faut aussi pouvoir reconnaître les clés primaires et étrangères permettant respectivement d'identifier une table de façon unique, et de garantir l'intégrité des données.

L'extraction peut aussi s'occuper de vérifier les erreurs des sources. Par exemple, il est possible qu'une personne fasse une faute de frappe en écrivant "Canuda" au lieu de "Canada" alors l'outil ETL doit détecter cette erreur et la corriger.

Enfin, cette première étape doit être effectuée le plus rapidement possible en exploitant au minimum les ressources du système. Étant donné qu'un ETL peut occuper et ce, pendant de nombreuses heures, une grande partie du ou des processeurs disponibles, on lance donc souvent les processus ETL la nuit. Pour gagner du temps, l'objectif de cette étape est aussi de filtrer au maximum les données. Par exemple, il faut que l'on puisse extraire les données uniquement mises à jour ou ajoutées après la dernière extraction. L'étape d'extraction est donc très importante. Elle doit être performante et complète pour pouvoir disposer d'un bon outil ETL.

1.2.2 Transformation

Cette seconde étape a pour objectif la transformation des données. Elle est bien évidemment indispensable si l'on veut obtenir des cibles différentes des sources.

C'est cette étape qui va permettre de joindre les différentes sources selon les clés précédemment spécifiées. Elle va aussi permettre de filtrer les données. Le filtrage est bien déifièrent de l'extraction puisque l'on filtre selon des critères à définir, par exemple on va filtrer les produits dont le prix est supérieur à 1000$

Une partie importante de l'étape de transformation est de pouvoir effectuer des calculs.

Ils peuvent être simples comme une addition ou multiplication, mais peuvent être aussi plus complexes. Disposer d'un outil ETL proposant de nombreuses opérations par défaut est donc un

plus. La transformation doit aussi s'occuper des différentes agrégations : effectuer les commandes SQL classiques tels que SUM (somme), COUNT (comptage) ou AVG (moyenne).

En bref, cette étape doit permettre d'effectuer toutes les transformations que l'on souhaite appliquer aux données sources. Il ne faut pas non plus oublier la sélection ou le découpage des colonnes, la traduction des valeurs (les différents formats de dates possibles ou encore le booléen 1 qui peut signifier M pour "Masculin"), la fusion, la gestion des erreurs et encore de nombreuses autres fonctionnalités.

1.2.3 Load

La dernière étape, s'occupe de charger les données, préalablement extraites puis transformées, dans des cibles hétérogènes (le plus souvent des entrepôts de données qui pourront être structurés selon un modèle bien précis (vu précédemment).

Le chargement va permettre d'insérer ou de mettre à jour les données cibles, et, comme dans les deux étapes précédentes, il doit aussi gérer les erreurs (une chaîne de caractère ne doit pas être insérée dans un champ fait pour les entiers).

Le chargement n'est pas à négliger pour un bon outil ETL, il doit, là aussi, être complet et performant.

1.3. Autre fonctionnalité des outils ETL

Un outil ETL, qui se veut être complet, doit implémenter de nombreuses autres fonctionnalités. Il peut par exemple permettre de planifier les exécutions : lancer un traitement un jour précis ou à une fréquence précise selon une contrainte donnée (les possibilités peuvent être nombreuses). Une console d'administration est aussi la bienvenue avec l'enregistrement d'utilisateurs et de leurs privilèges, tout en permettant de surveiller les processus ETL en cours. Tout ceci doit être géré par un système bien sécurisé. Pour un travail collaboratif, on peut bien évidemment penser à des systèmes de contrôle de version genre CVS (Concurrent Versions System).

Pour optimiser les performances, un système ETL peut proposer de paralléliser les traitements et de coordonner les processus. Pour bien gérer les erreurs, l'outil ETL peut proposer des rapports d'erreurs, des outils de correction de bugs, la reprise après une erreur, la vérification d'un traitement avant son exécution ou encore l'affichage des statistiques d'exécutions.

Les possibilités d'un outil ETL sont donc très nombreuses et c'est bien évidemment un critère à retenir pour disposer d'un outil ETL complet.

1.4. Application des outils ETL a le BI

La plupart du temps, les ETL sont utilisés dans le domaine de la Business Intelligence (BI) décrit auparavant. La BI est, le plus souvent, composée de déférentes parties intimement liées, permettant d'aider à prendre une décision pour répondre aux problèmes décisionnels tels que :

- ✓ L'intégration de données qui alimente des entrepôts de données. C'est ici qu'interviennent les ETL

- ✓ La génération de rapports qui fournit aux utilisateurs des rapports sur l'état des ventes, des stocks, du chiffre d'affaires, etc. Cette partie est gérée via des outils de Reporting qui piocheront dans un entrepôt de données alimenté par un ETL.
- ✓ Les tableaux de bords (ou dashboards en anglais) mettent en place de nombreux graphiques et schémas, pour observer, d'un coup d'oeil, ce qui va ou qui ne va pas dans l'entreprise. Par exemple si un stock est quasiment vide, une jauge en rouge peut s'afficher pour prévenir rapidement l'utilisateur. Là encore, les données seront récupérées à partir d'un entrepôt de données.
- ✓ L'analyse des données permet d'aller plus en profondeur par rapport aux rapports mais aussi d'interagir et de vérifier les données selon plusieurs niveaux (années, trimestres, mois, semaines, jours par exemple). Les données seront récupérées via des cubes multidimensionnels OLAP, qui sont eux même alimentés par des entrepôts de données.
- ✓ Le Data Mining est la partie la moins utilisée puisque c'est la plus complexe. Cette branche fait intervenir de nombreux algorithmes (touchant souvent au domaine de l'intelligence artificielle) essayant d'apporter à l'utilisateur les futures évolutions probables de son entreprise.

La majorité des outils ETL mettent à disposition des outils spécifiques pour alimenter les entrepôts de données. Les clés de substitutions présentées plus haut, l'alimentation de cubes OLAP ou encore la gestion des dimensions à évolution lente (slow changing dimension) en sont des exemples.

OLAP

La finalité d'un entrepôt de données est de supporter le traitement analytique en-ligne (OLAP). Les techniques de type OLAP (**O**n-**L**ine **A**nalytical **P**rocessing) effectuent la synthèse, l'analyse et la consolidation dynamique des techniques multidimensionnelles. Les techniques OLAP sont la manière la plus naturelle d'exploiter un entrepôt à cause de son organisation multidimensionnelle.

Pour permettre des analyses et des visualisations complexes, les données dans l'entrepôt sont organisées selon le modèle de données multidimensionnel. La modélisation de données multidimensionnelles (modélisation dimensionnelle) est le nom d'une méthode de conception logique souvent associée aux entrepôts de données. Elle signifie l'agrégation partielle des données de l'entrepôt selon différents critères. Un système OLAP emploie ce concept.

Quand on dit que l'information est multidimensionnelle, cela veut dire qu'elle peut être représentée sous forme de tableaux croisés dynamiques. En effet elle est visible sous forme de cubes et les outils offrent des possibilités de naviguer dans ceux-ci en pivotant les axes, en consolidant les données à des niveaux hiérarchiques supérieurs, tout en désagrégeant d'autres données à des niveaux de détails très fins.

1. Analyse multidimensionnelle:

OLAP (Online Analytical Processing), désigne les bases de données multidimensionnelles (aussi appelées cubes ou hypercubes) destinées à des analyses complexes sur ses données. Ce terme a été défini par Ted Codd en 1993 au travers de règles que doit respecter une base de données si elle veut adhérer au concept OLAP.
Ce concept est appliqué à un modèle virtuel de représentation de données appelé cube ou hypercube OLAP, qui nous intéresse plus particulièrement.
Cette hypercube est une représentation abstraite des données prévue à des fins d'analyses interactives par une ou plusieurs personnes (souvent ni informaticiens ni statisticiens) du métier que ces données sont censées représenter.
Les cubes OLAP ont les caractéristiques suivantes :

- Obtenir des informations déjà agrégées selon les besoins de l'utilisateur.
- Simplicité et rapidité d'accès
- Capacité à manipuler les données agrégées selon différentes dimensions
- Un cube utilise les fonctions classiques d'agrégation : min, max, count, sum, avg, mais peut utiliser des fonctions d'agrégations spécifiques.

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 relationnelle) .

Ce modèle de cube OLAP existe dans plusieurs implémentations :

- **M-OLAP** : La forme la plus classique car la plus rapide. Elle utilise des tables multidimensionnelles pour sauver les informations et réaliser les opérations.

- Les données sont stockées comme des matrices à plusieurs dimensions : Cube [1:m, 1:n, 1:p] (mesure)
- Accès direct aux données dans le cube
- Avantage :
- Rapidité
- Inconvénients :
- Difficile à mettre en place
- Formats souvent propriétaires
- Ne supporte pas de très gros volumes de données
- Exemple de moteurs MOLAP :
- Microsoft Analysis Services
- Hyperion

· **R-OLAP** : Celle qui demande le moins d'investissement. Elle travaille sur des tables relationnelles. Une nouvelle table est créée pour contenir chaque agrégat.

- Les données sont stockées dans une BD relationnelle
- Avantage :
- Facile à mettre en place
- Peu couteux
- Evolution facile
- Stockage de gros volumes
- Inconvénients :
- Moins performant lors des phases de calculs
- Exemple de moteur ROLAP : Mondrian

· **H-OLAP** : (Hybrid OLAP) : Elle utilise à la fois les tables relationnelles pour stocker les informations brutes, et des tables multidimensionnelles pour les agrégats d'informations prédictives.

- Solution hybride entre ROLAP et MOLAP
- Données de base stockées des tables de faits et de dimensions + données agrégées stockées dans un cube
- Avantages / inconvénients :

Bon compromis au niveau des coûts et des performances (les requêtes vont chercher les données dans les tables et le cube)

Les analyses OLAP consistent à suivre des indicateurs considérés comme des points observés dans un espace défini par différents axes d'analyse. Cette vision multidimensionnelle des données peut être vue comme un cube de données.
Le cube de données est formé d'arêtes représentants les axes d'observations d'indicateurs placés dans les cellules. Sur chaque arête, une graduation est choisie afin d'observer les données à un niveau adéquat de granularité.

Exemple. La figure suivante présente un cube de données formé de montants de vente en cellules et de trois arêtes graduées respectivement par des catégories de produits, des villes de magasins et des trimestres. La notion de cube de données ne se limite pas à trois axes mais se généralise en hyper-cube où le nombre d'axes est quelconque pouvant aller jusqu'à plusieurs dizaines. Les utilisateurs accèdent aux cubes OLAP grâce à des outils d'analyse offrant ainsi la capacité de réaliser à la volée des tableaux de synthèse, des rapports graphiques et des indicateurs pour réaliser des tableaux de bord.

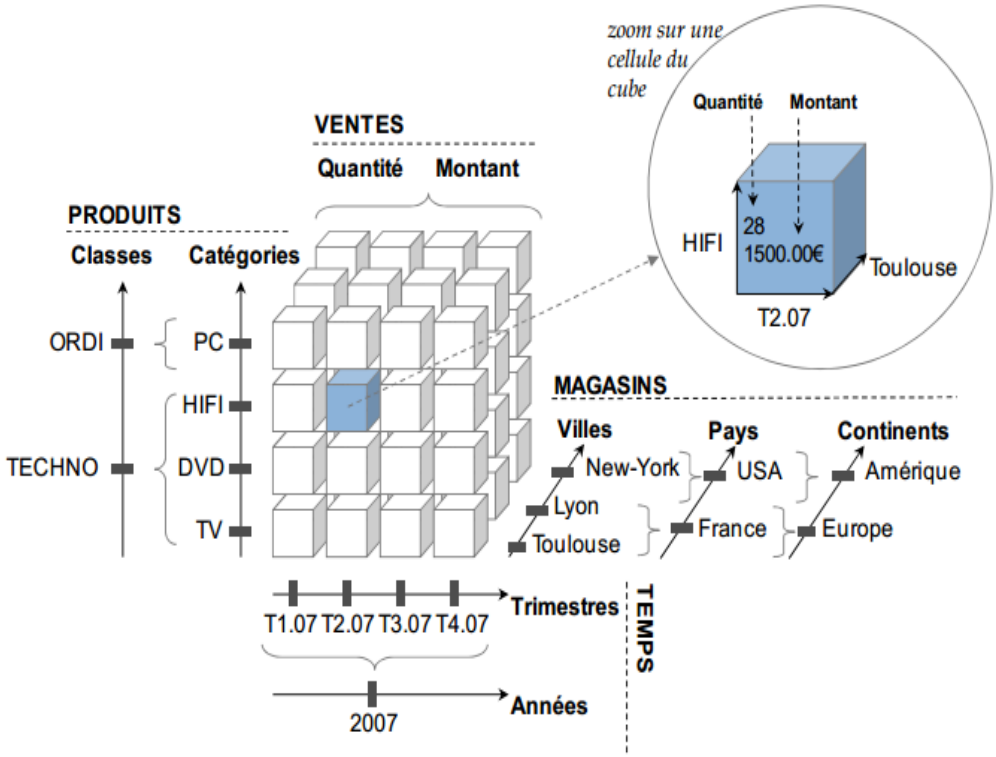


Figure 12 : exemple d'un cube de données

Les opérations typiques exécutées par les clients OLAP peuvent être de nature :

- **Opérations de forage:** les opérations de forage font reposer la navigation sur la structure hiérarchique des axes d'analyses, afin de permettre l'analyse d'un indicateur avec plus ou moins de précision.

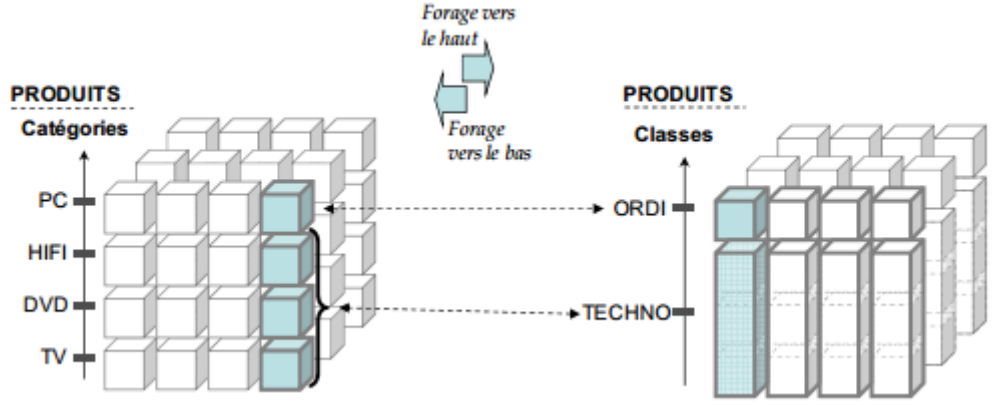
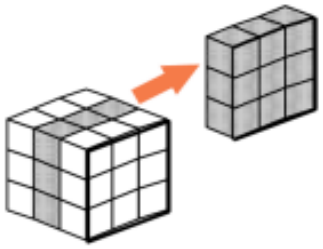


Figure 13 : exemple d'opération de forage

- **Opérations de projection (Slice et Dice)**

Quantité des ventes		Géographie.Département					
		Loiret	Eure et Loir	Indre	Indre et Loire	Loir et Cher	Gironde
Temps.Année	2005	100	200	100	150	1	2
	2006	250	210	120	200	2	2
	2007	260	230	250	200	4	5
	2008	280	220	220	200	5	4
Véhicules.AllV							

Slice (Année = « 2005 »)



Quantité des ventes		Géographie.Département					
		Loiret	Eure et Loir	Indre	Indre et Loire	Loir et Cher	Gironde
Temps.Année	2005	100	200	100	150	1	2
Véhicules.AllV							

Figure 14 : exemple d'opération slice

Quantité des ventes		Géographie.Département					
		Loiret	Eure et Loir	Indre	Indre et Loire	Loir et Cher	Gironde
Temps.Année	2005	100	200	100	150	1	2
	2006	250	210	120	200	2	2
	2007	260	230	250	200	4	5
	2008	280	220	220	200	5	4
Véhicules.AllV							

Dice (Département = « Loir et Cher » ou « Gironde », Année = « 2007 » ou « 2008 »)

Quantité des ventes		Géographie.Département	
		Loir et Cher	Gironde
Temps.Année	2007	4	5
	2008	5	4
Véhicules.AllV			

Figure 15 : exemple d'opération dice

- **Les opérations de rotation** réorientent une analyse. L'opération la plus courante consiste à changer l'axe d'analyse en cours d'utilisation (rotation de dimension).

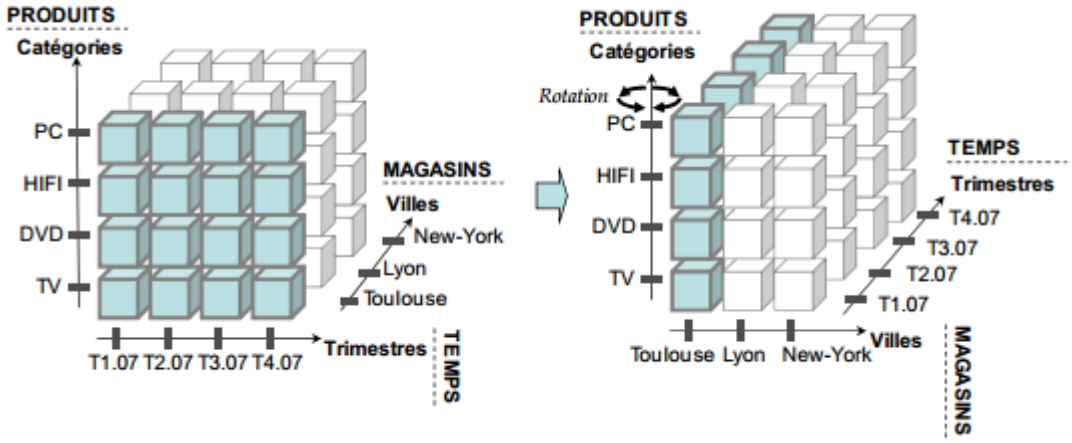


Figure 16 : exemple d'opération de rotation

Les technologies OLAP, par leur aspect dynamique et synthétique complètent les outils de reporting. Les outils de reporting sont généralement utilisés afin de fournir des vues statiques au travers de rapports instantanés à partir des données de l'entrepôt. A la différence des outils de requêtage OLAP, les fonctions de forage dynamique et de changement d'axes à la demande y sont absentes.

2. Terminologie d'OLAP :

Les serveurs olap utilisent des schémas xml qui décrit les cubes et leurs dimensions d'une datawerhouse.

On va établir le vocabulaire propre à ce domaine.

Comme nous l'avons vu OLAP propose une approche multidimensionnelle ce qui nous amené à la notion d' (hyper)cube.

- un **cube** est composé de dimensions
- une **dimension** peut contenir une ou plusieurs **hiérarchies** : la dimension "Time" contient 2 hiérarchies : "Year, Quarter, Month" et "Year, Week, Day"
- Une **hiérarchie** est composée de **niveaux** ("levels") correspondant à un des attributs de la base de données :
    - hiérarchie "Time" est composée des niveaux "Year", "Quarter" et "Month "
    - hiérarchie "Store" est composée des niveaux "Country", "State", "City","Store_Name"
- Un **niveau** est composé de membres qui sont les valeurs d'un niveau détectées par le moteur OLAP et stockées dans les métadonnées :
    - les membres du niveau "Country" sont "France", "Canada" et "USA"
    - les membres du niveau "City" sont "Marseille", "Lyon" et "Paris".
- Une **mesure** est une quantité intéressante que l'on souhaite observer, par exemple :

- montant des ventes,
- quantité de produits vendus

- Un **schéma** = modèle logique définissant une BD multidimensionnelle ainsi que les structures associées : cubes, dimensions, hiérarchies, niveaux et membres
- Il est en général en étoile, se traduit par un ensemble de tables relationnelles
- Composants majeurs d'un schéma :
    - **cube** = collection de dimensions et de mesures dans un domaine particulier.
    - **dimension** = attribut, ou ensemble d'attributs, à travers lesquels sont observées les mesures
    - **mesure** = quantité intéressante, qu'on souhaite observer (Ex : montant des ventes, …)

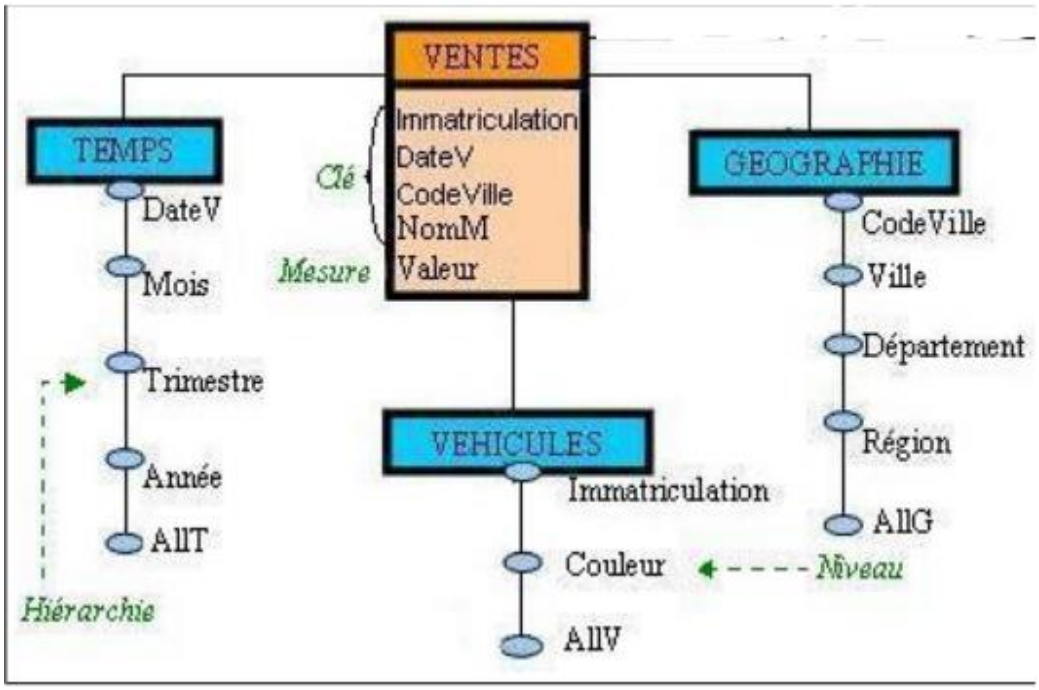


Figure 17 : schéma en étoile de cube de données

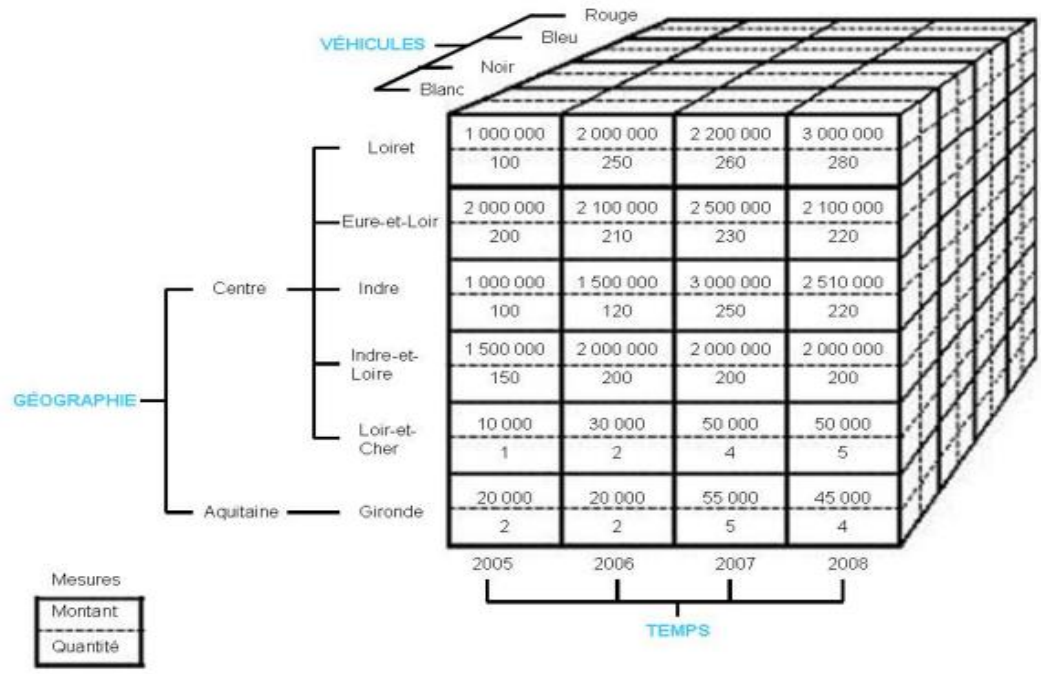


Figure 18 : exemple de cube de données

Les types de schémas :

Schéma en étoile : Une table de fait centrale et des dimensions

- Les dimensions n'ont pas de liaison entre elles
- Avantage :
  - ✓ Facilité de navigation
  - ✓ Nombre de jointures limité
- Inconvénients :
  - ✓ Redondance dans les dimensions

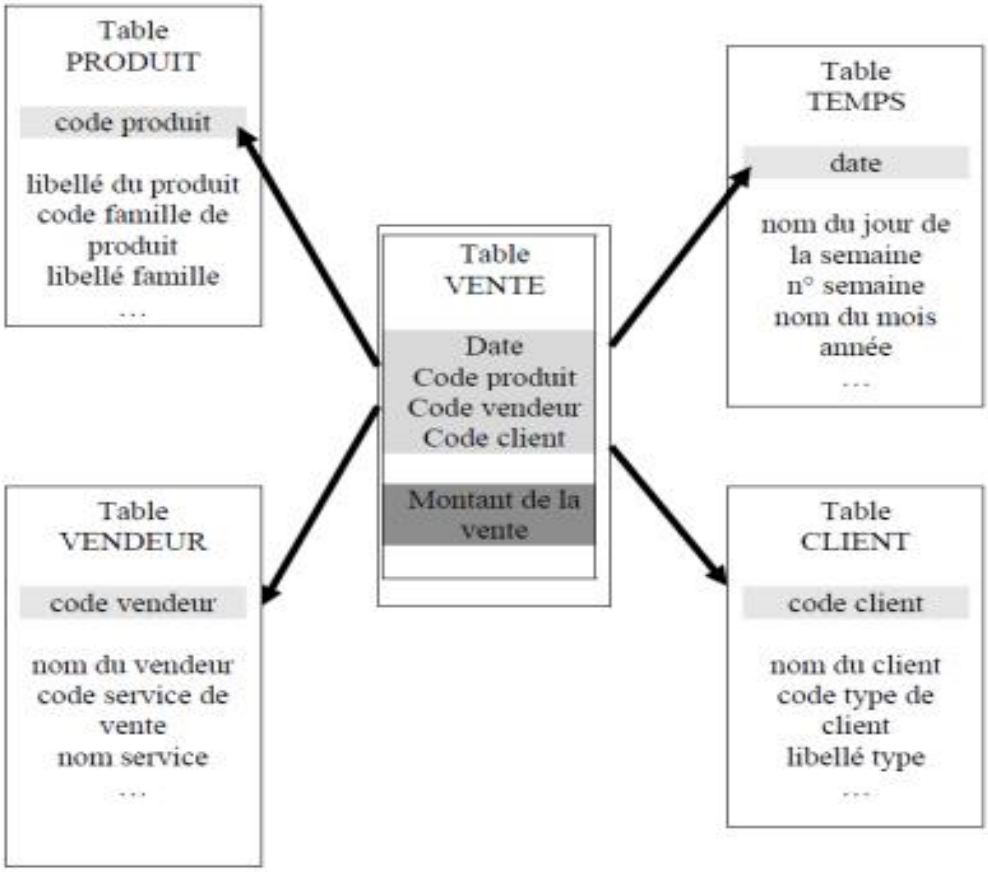


Figure 19 : exemple de schéma en étoile

Schéma en constellation

- Série d'étoiles

✓ Fusion de plusieurs modèles en étoile qui utilisent des dimensions communes

✓ Plusieurs tables de fait et tables de dimensions, éventuellement communes

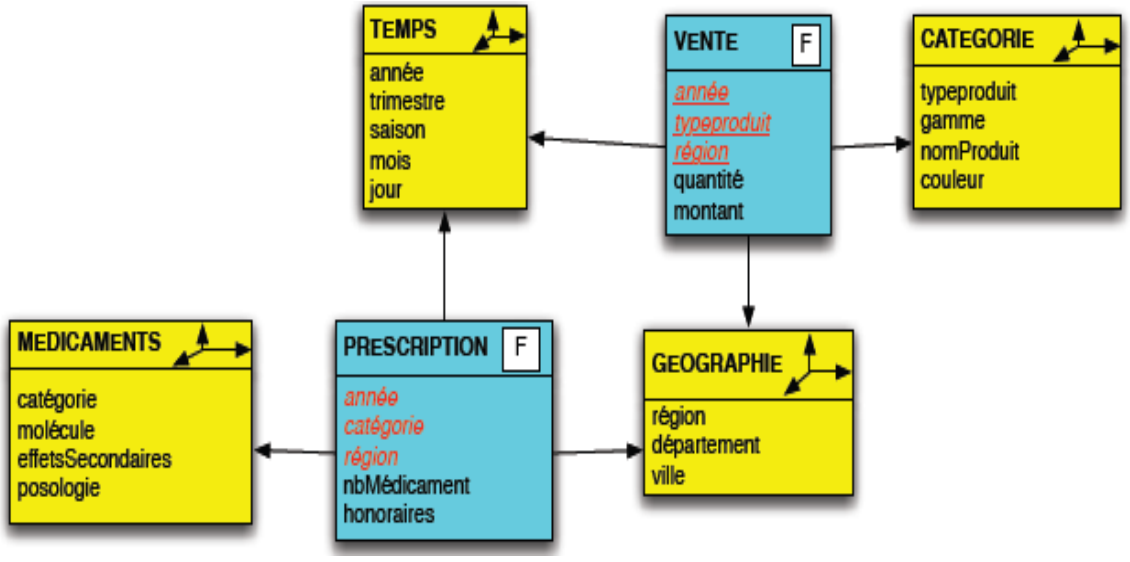


Figure 20 : exemple de schéma en constellation

Modèle en flocon

- Une table de fait et des dimensions en sous-hiérarchies
- Un seul niveau hiérarchique par table de dimension
- La table de dimension de niveau hiérarchique le plus bas est reliée à la table de fait (elle a la granularité la plus fine)

- Avantage :

- Normalisation des dimensions
- Economie d'espace disque (réduction du volume)

- Inconvénients :

- Modèle plus complexe (nombreuses jointures)
- Requêtes moins performantes
- Navigation difficile

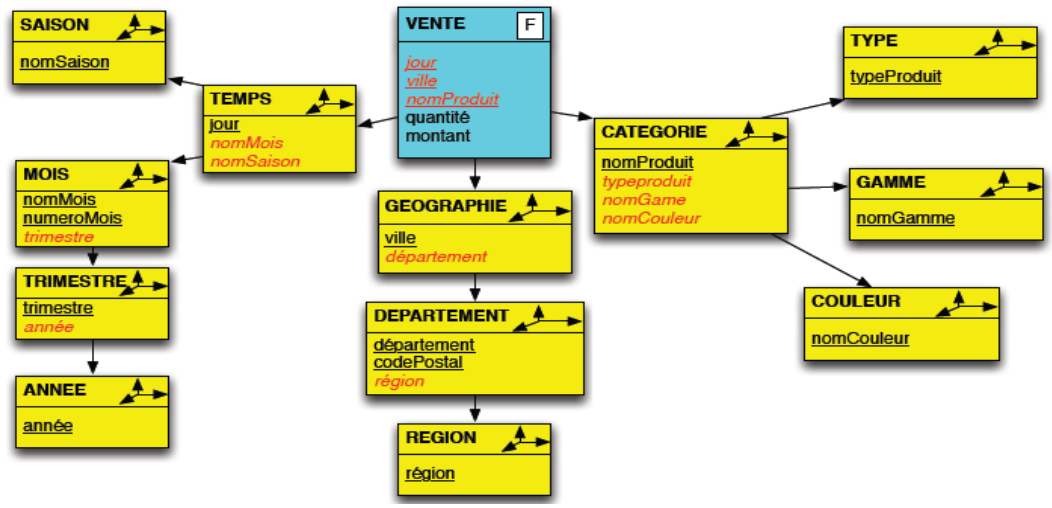


Figure 21 : exemple de schéma en flocon

3. Schéma de données :

L'analyse multidimensionnelle est l'un des modes d'analyse le plus courants dans le décisionnel (les autres concernent les statistiques, le datamining, les systèmes d'aide à la décision, …).
Une base de données traditionnelle ne permet aux utilisateurs que des visions en deux dimensions comme par exemple l'étude des produits par région. Une base de données multidimensionnelle permet aux utilisateurs une analyse intégrant plusieurs dimensions comme par exemple, l'étude des ventes de produits par région par couleur, par taille et ce dans le temps.

Un schéma définit une base de données multidimensionnelle. Il contient le modèle logique des cubes, des hiérarchies, des membres et la mise en correspondance (mapping) qui transforme ce modèle logique en un modèle physique. Le modèle logique consiste à définir les structures utilisées dans une requête MDX : cubes, dimensions, hiérarchies, niveaux et membres. Il donne également la source des données représentées dans le modèle logique. C'est le schéma en étoile qui est traduit par un ensemble de tables relationnelles. Les schémas sous Mondrian sont représentés à l'aide d'un document XML. Un moyen de créer un tel schéma est d'éditer un document XML et de respecter la syntaxe proposée dans Mondrian.

Les plus importants des composants du schéma sont les cubes, les mesures et les dimensions. Un cube est une collection de dimensions et de mesures dans un domaine particulier. Une mesure est une quantité qui vous intéresse, que vous souhaitez observer (par exemple, le montant des ventes, nombre de produits inventoriés, etc.). Une dimension est un attribut, ou un ensemble d'attributs, à travers lesquels sont observées les mesures. Par exemple, vous pouvez être intéressés à observer la vente des produits selon leurs couleurs, le sexe du client et le magasin où sont vendus ces produits. La couleur du produit, le sexe du client et le magasin de vente sont des dimensions. Ci-dessous, un exemple d'une définition XML d'un schéma Mondrian simple.

```
<?xml version="1.0"?>
<Schema name="DemoCube">
    <Cube name="Sales">
        <Table name="sales_fact"/>
        <Dimension name="Gender" foreignKey="customer_id">
            <Hierarchy hasAll="true" allMemberName="All Genders" primaryKey="customer_id">
                <Table name="customer"/>
                <Level name="Gender" column="gender" uniqueMembers="true"/>
            </Hierarchy>
        </Dimension>
        <Dimension name="Time" foreignKey="time_id">
          <Hierarchy hasAll="false" primaryKey="time_id">
                <Table name="time_by_day"/>
                <Level name="Year" column="the_year" type="Numeric" uniqueMembers="true"/>
                <Level name="Quarter" column="quarter" uniqueMembers="false"/>
                <Level name="Month" column="month_of_year" type="Numeric" uniqueMembers="false"/>
          </Hierarchy>
          <Hierarchy name="Time Weekly" hasAll="false" primaryKey="time_id">
                <Table name="time_by_week"/>
                <Level name="Year" column="the_year" type="Numeric" uniqueMembers="true"/>
                <Level name="Week" column="week" uniqueMembers="false"/>
                <Level name="Day" column="day_of_week" type="String" uniqueMembers="false"/>
          </Hierarchy>
        </Dimension>
        <Measure name="Unit Sales" column="unit_sales" aggregator="sum" formatString="#,###"/>
        <Measure name="Store Sales" column="store_sales" aggregator="sum" formatString="#,###.##"/>
        <Measure name="Store Cost" column="store_cost" aggregator="sum" formatString="#,###.00"/>
        <Measure name="Sales Count" column="product_id" aggregator="count" formatString="#,###"/>
    <CalculatedMember name="Profit" dimension="Measures" formula="[Measures].[Store Sales]-[Measures].[Store Cost]">
    <CalculatedMemberProperty name="FORMAT_STRING" value="$#,##0.00"/>
    </CalculatedMember>
    </Cube>
</Schema>
```

Figure 22 : schéma d'un cube

Ce schéma contient un seul cube de ventes, appelé "Sales". Les ventes sont observées sur deux dimensions "Time" et "Gender", et quatre mesures "Unit Sales", "Store Sales", "Store cost" et "Profit". Notez bien que la mesure profit est calculée à partir des mesures "Store Sales" et "Store cost".

3.1. Cube:

Un cube (<Cube>) est une collection de mesures et de dimensions, identifiée par un nom. Les dimensions et les mesures ont la table de faits en commun (dans l'exemple, la table de faits est "sales_fact"). La table de faits contient les colonnes à partir desquelles les mesures sont calculées et des références vers les tables contenant les dimensions.

```
<Cube name="Sales">
        <Table name="sales_fact"/>
        ...
</Cube>
```

La table de faits est définie en utilisant <Table>. Si la table de faits n'est pas dans le schéma par défaut, vous pouvez fournir explicitement son schéma en utilisant l'attribut "schema". Par exemple,<Table schema="dmart" name="sales_fact"/>. Vous pouvez aussi utiliser <View> et <Join> pour construire des commandes SQL encore plus complexes.

3.2. Mesures:

Il existe deux sortes de mesures : les mesures calculées et les non calculées. Le cube des ventes définit plusieurs mesures, dont "Unit Sales" et "Store Sales". La mesure "Profit" est une mesure calculée à partir des mesures "Store Sales" et "Store Cost"

```
<Measure name="Unit Sales" column="unit_sales" aggregator="sum" datatype="Integer" formatString="#,###"/>

<Measure name="Store Sales" column="store_sales" aggregator="sum" datatype="Numeric" formatString="#,###.00"/>

<CalculatedMember name="Profit" dimension="Measures" formula="[Measures].[Store Sales]-[Measures].[Store Cost]">
```

Chaque mesure (<Measure>) a un nom, une colonne de correspondance dans la table de faits, un opérateur d'agrégation. L'opérateur d'agrégation est souvent "sum", mais d'autres opérateurs comme "count", "min", "max", "avg" et "distinct-count" peuvent être utilisés. L'opérateur "distinct-count" a des limitations si le cube contient une hiérarchie parent-fils.

3.3. Dimensions, huérarchies, niveaux:

Un membre est un point dans une dimension déterminé par les valeurs d'attributs de cette dimension. La hiérarchie "Gender" a deux membres 'M' et 'F'.
Une hiérarchie est un ensemble de membres organisés selon une structure appropriée pour l'analyse.
Par exemple, les villes peuvent être regroupées par région et les régions par pays. Les mesures sont agrégées pour chaque niveau de la hiérarchie. Les ventes d'un pays sont calculées à partir des ventes de ses régions.
Un niveau est une collection de membres qui ont la même distance de la racine de la hiérarchie.
Une dimension est une collection de hiérarchies selon laquelle les faits sont observés.
Ci-dessous un exemple d'une représentation XML de la dimension "Gender".

```
<Dimension name="Gender" foreignKey="customer_id">
        <Hierarchy hasAll="true" primaryKey="customer_id">
                <Table name="customer"/>
                <Level name="Gender" column="gender" uniqueMembers="true"/>
```

```
        </Hierarchy>
</Dimension>
```

Cette dimension a une seule hiérarchie et un seule niveau. La dimension prend ses valeurs à partir de la colonne "gender" de la table "customer".
La colonne "gender" a deux valeurs 'F' et 'M'. La dimension "Gender" a donc deux membres "[Gender].[F]" et "[Gender].[M]".
Pour chaque vente, la dimension "Gender" donne le sexe du client ayant réalisé un achat. Cela s'exprime par la jointure entre la table de faits "sales_fact" et la table de dimensions "customer" sur l'attribut "customer_id".

4. MDX:

MDX (**M**ulti **D**imensional e**X**pression) 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 leurs objets (dimensions, hiérarchies, niveaux, membres et cellules) afin d'obtenir (simplement) une représentation sous forme de tableaux croisés
- MDX ressemble à SQL par ses mots clé SELECT, FROM, WHERE, mais :
    - SQL construit des vues relationnelles
    - MDX construits des vues multidimensionnelles des données
    - Analogies entre termes multidimensionnels (MDX) et relationnels (SQL) :

Multidimensionnel MDX	Relationnel SQL
Cube	Table
Niveau	Colonne (chaîne de caractères ou valeurs numériques)
Dimension	Plusieurs colonnes liées ou une table de dimension
Mesure	Colonne (discrète ou numérique)
Membre de dimension	Valeur dans une colonne et une ligne particulière de la table

Figure 23 : la différence entre MDX et SQL

• Structure générale d'une requête SQL :
SELECT column1, column2, …, columnn FROM table
• Structure générale d'une requête MDX :
SELECT axis1 ON COLUMNS, axis2 ON ROWS FROM cube
• FROM spécifie la source de données :
- en SQL : une ou plusieurs tables
- en MDX : un cube

• SELECT indique les résultats que l'on souhaite récupérer par la requête :
- en SQL :
  • une vue des données en 2 dimensions (lignes (rows) et colonnes (columns))
  • les lignes ont la même structure définie par les colonnes
- en MDX :
  • nb quelconque de dimensions pour former les résultats de la requête.
  • terme d'axe pour éviter confusion avec les dimensions du cube.
  • pas de signification particulière pour les rows et les columns, mais il faut définir chaque axe : axe1 définit l'axe horizontal et axe2 définit l'axe vertical

Structure générale

Dans son approche MDX est proche du SQL sur son aspect select et where, même si la similarité ne va pas plus loin. Un prototype d'une requête MDX est donné par la syntaxe suivante.

```
SELECT [<axis_specification> [, <spécification_des_axes>...]]
FROM [<spécification_d_un_cube>]
    [WHERE [<spécification_de_filtres>]]
```

a. Spécification des axes :

La spécification d'un axe doit être un set suivi du mot clef on qui est suivi à son tour d'un nom d'axe. (Daniel: pédagogiquement, on utilise axe sans le définir, et après l'avoir utilisé (on columns), ce n'est pas idéal. Kamel: Je n'ai pas compris ta phrase mais si tu veux parler de l'utilisation de colonne dans l'exemple de membres, je peux les enlever car ils ne sont pas nécessaires pour comprendre ce que c'est un membre, un tuple et un set.)

```
{
        ([Measures].[Unit Sales], [Product].[Food]),
        ([Measures].[Unit Sales], [Product].[Drink])
}
on columns
```

La notion d'axe peut faire référence à un numéro d'ordre s'il y a plus de deux axes de restitution, ou tout simplement aux noms d'axes explicites "columns" tout d'abord et "rows". Sous Mondrian, seulement deux axes peuvent être utilisés.
La requête suivante donne les unités vendues "[Measures].[Unit Sales]" par an pour les produits "Drink" et "Food".

```
select
{
  ([Measures].[Unit Sales], [Product].[Food]),
  ([Measures].[Unit Sales], [Product].[Drink])
```

```
} on Axis(0),
{
        ([Time].[1997]),
        ([Time].[1998])
} on Axis(1)
from [Sales]

/* ou bien */

select
{
   ([Measures].[Unit Sales], [Product].[Food]),
   ([Measures].[Unit Sales], [Product].[Drink])


} on columns,
{
        ([Time].[1997]),
        ([Time].[1998])
} on rows
from [Sales]
```

b. Spécifications des filtres (slicers) :

Dans la syntaxe d'une requête MDX, nous disposons d'une clause where dans laquelle on indique un set pour filtrer les données.
La requête suivante donne les unités vendues aux clients de sexe masculin par an pour les produits "Drink" et "Food".

```
select
{
   ([Measures].[Unit Sales], [Product].[Food]),
   ([Measures].[Unit Sales], [Product].[Drink])
}
on columns,
{
        ([Time].[1997]),
        ([Time].[1998])
}
on rows
from [Sales]
where
{
        ([Gender].[M])
}
```

MDX permet d'utiliser plusieurs mesures par cube. Il suffit d'utiliser la dimension "[Measure]". Cependant, l'agrégation ne s'opère pas sur toutes les mesures mais seulement sur une seule définie par défaut dans le schéma modélisant le cube. Pour changer la mesure par défaut, mentionnez celle qui vous intéresse dans la clause where.
La première requête ci-dessus, utilise la mesure par défaut ("[Unit Sales]") du cube "Sales" pour agréger les données. Dans la deuxième, nous spécifions dans la clause where d'utiliser la mesure "[Store Cost]".

```
select
        {([Product].[Drink])}on columns,
        {([Time].[1997])} on rows
from [Sales]
```

c. Insertion des commentaires :

Les commandes MDX peuvent être commentées de trois façons différentes.

```
// Commentaire en fin de ligne
-- Commentaire en fin de ligne
/* Commentaire
  sur plusieurs lignes
*/
```

Les commentaires peuvent être imbriqués comme le montre l'exemple ci-desous

```
/* Commentaire sur
   plusieurs lignes /* Commentaire imbriqué */
*/
```

5. La Suite de SpagoBI :

SpagoBI est une suite décisionnelle développée par la société italienne Engineering. Ce projet a été initié en 2005. Elle a comme particularité d'être la seule solution open source 100 % free, une seule version stable avec 100 % des fonctionnalités disponibles.

SpagoBI est une suite flexible. Elle offre de nombreux moteurs pour un même domaine d'analyse, permettant aux développeurs de choisir librement leur propre solution. Basé sur des standards ouverts, SpagoBI s'appuie sur des solutions pérennes et open source. En plus des fonctionnalités de reporting, il est à noter qu'elle permet d'intégrer des fonctionnalités de MDM et ETL.

Seul le contexte d'utilisation permettra de choisir les moteurs à utiliser pour répondre au mieux aux exigences de réalisation, de segmentation des analyses, des cibles d'utilisation, de récupération ou de sauvegarde des investissements passés. L'évolution de SpagoBI va également dans le sens d'une augmentation du nombre de moteurs supportés pour chaque domaine, de façon à permettre une plus grande flexibilité, une installation légère et peu invasive vis à vis des solutions préexistantes et de garantir une meilleure protection des investissements déjà entrepris.

SpagoBI est disponible en licence LGPL, c'est-à-dire uniquement en open source, il n'existe pas de version commerciale. Si on le souhaite, l'éditeur propose des offres de services de support logiciel.

5.1. Historique de SpagoBI :

SpagoBI appartient à SpagoWorld3, l'initiative open source par génie Groupe. L'histoire a commencé en 2001, avec la nécessité de l'entreprise de réaliser un Cadre Java Enterprise qui

pourrait soutenir le développement de projets pour ses clients. À la fin de 2004, ce cadre a été repensé et libéré comme un logiciel open source sur SourceForge, c'est le cadre Spago.

SpagoBI projet4 a commencé à la fin de 2004. En Juillet 2005 SpagoBI a été publié pour la première fois sur SourceForge, puis déplacé sur la forge de consortium ObjectWeb. En 2006 SpagoWorld initiative a été officiellement lancé et la base de code de projets SpagoWorld a été déplacé sur la forge du nouveau Consortium OW2.

En 2007, deux autres projets ont été libérés: Spagic, l'open source universelle middleware, et Spago4Q, qui est une application verticale de SpagoBI concentré sur la qualité des produits, procédés et services. En 2010, quand génie Groupe rejoint la Fondation Eclipse, les eBPM et Ebam projets d'éclipse, ce dernier avance par l'équipe SpagoBI, et l'initiative GeoBI, ont été lancés.
La figure suivante montre le processus et le parcours de développement de la Suite SpagoBI :

Stratégie de développement

Centre de compétences SpagoBI favorise le développement des projets et des applications de renseignement, afin de permettre aux organisations et entreprises à atteindre leurs objectifs stratégiques, dans un climat de confiance, garantis par la qualité des logiciels, l'utilisation de standards ouverts et de logiciels libres, et de l'innovation.

La stratégie adoptée par le centre de compétence SpagoBI est basé sur le partage des développements avec la communauté, la consolidation des solutions open source au niveau de l'entreprise, renforcer les partenariats internationaux et orienter le projet feuille de route pour les initiatives industrielles.
En particulier, la vision SpagoBI est que les besoins de l'utilisateur sont plus précieux que le produit adopté, en contraste avec de nombreuses solutions d'imposer de contraintes sur les utilisateurs de produits aux exigences. De cette façon, une bonne solution peut réellement améliorer un projet de business intelligence, si elle favorise:

- une solution axée sur les projets, qui favorise le démarrage du projet en offrant une

nouvelle équilibre entre les éléments impliqués dans la phase de développement - efforts, les coûts, le temps et la qualité des résultats. Pour cette raison, SpagoBI offre de nombreuses autres solutions, qui sont à base de produits exclusifs et caractérisé par des écarts de prix.

- une conception cohérente, grâce à l'adoption des normes ouvertes, une architecture modulaire, le développement de composants innovants et l'intégration des meilleures solutions open source, en soutenant leur intégration et la réutilisation dans les environnements existants, ce qui augmente ainsi la valeur des solutions qui sont déjà en cours d'utilisation.
- un modèle d'entreprise partagé entre les fournisseurs et les utilisateurs dans une perspective gagnant-gagnant: le partage des objectifs et priorités d'affaires peuvent être le meilleur moyen d'atteindre les résultats prévues et souhaitées.
- un processus de développement évolutif ou agile qui aide les utilisateurs à atteindre leurs buts depuis leurs premiers pas: commencer petit, penser grand, à obtenir rapidement les premiers résultats.

Selon cette vision, la suite SpagoBI à une caractéristique distinctive et unique c'est son approche entièrement open source, qui obtiennent sur l'attitude culturelle de garder hors de nouvelles règles et idées.

SpagoBI Server c'est un environnement pour:

- Utilisateurs finaux: ils peuvent utiliser leurs documents d'analyse par le biais de ce point d'accès unifié via un navigateur Web
- Les administrateurs système: ils peuvent gérer le serveur via une interface web.

Exemple d'OLAP

Dans cette partie on va voir un exemple dans SpagoBI, pour commencer on a utilisé une base de données open source FoodMart et on a migré vers Mysql.

Nous avons vu jusqu'à présent comment créer un cube à partir de la table de faits et de dimensions jointes à la table de faits. C'est le cas le plus courant de la modélisation des données dans un entrepôt de données. Un tel schéma est dit un schéma en étoile.

Cependant, une dimension peut être dans plus d'une seule table, fournissant un chemin pour joindre ces tables à la table de faits. Ce type de dimension est appelé en flocon de neige et est défini en utilisant l'opérateur <Join>. La figure ci-dessous montre un exemple simple d'un modèle en étoile du cube "Sales".

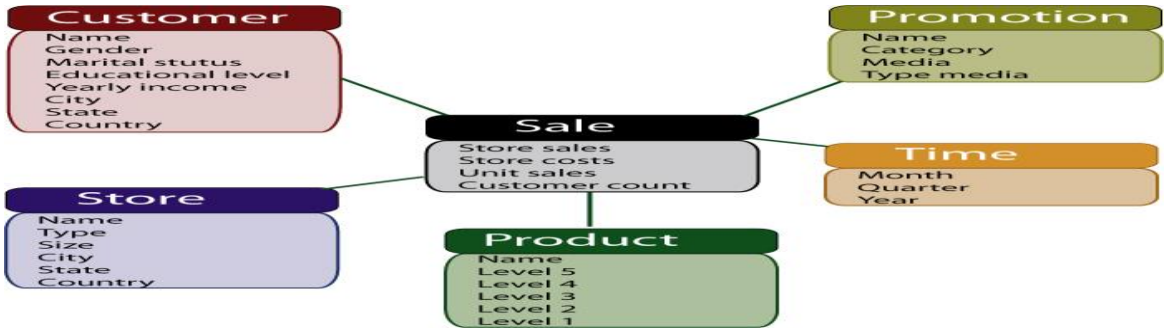


Figure 24 : exemple de schéma en étoile de cube « sales »

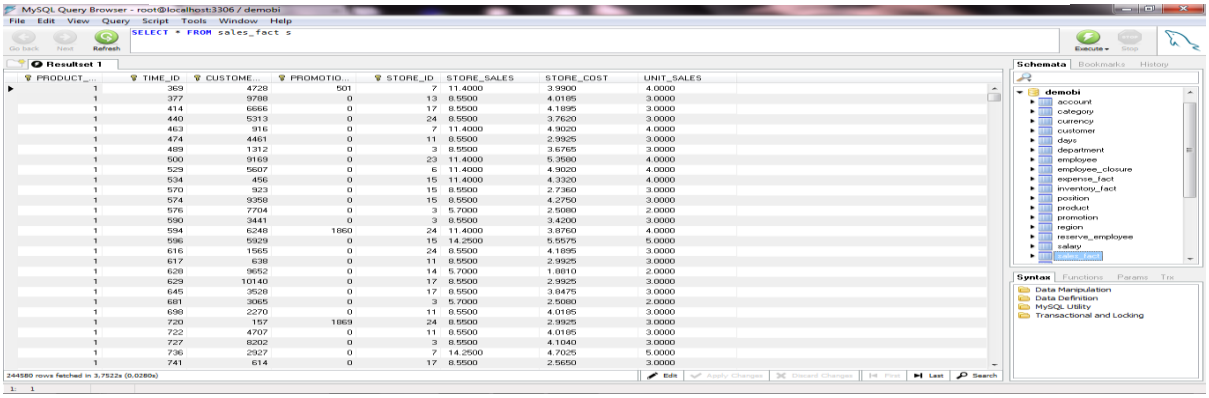


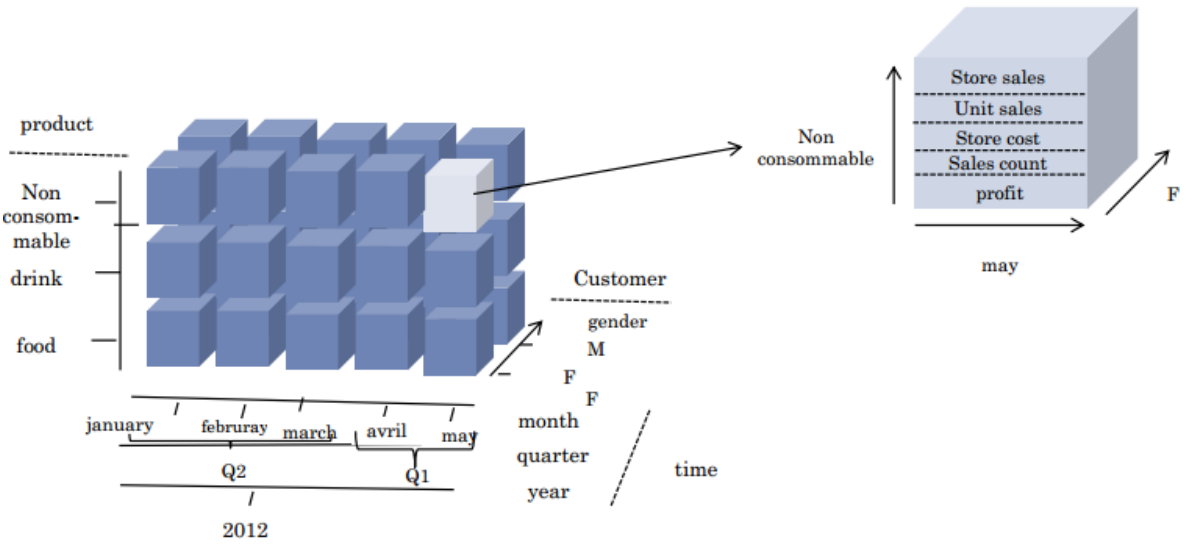
Figure 25 : base de données

Ensuite on a créé notre schéma de cube on se basant sur notre schéma en étoile, notre schéma se compose d'un seul cube comme le montre la figure suivante :

```
<?xml version="1.0"?>
<Schema name="DemoCube">
    <Cube name="Sales">
        <Table name="sales_fact"/>
        <Dimension name="Gender" foreignKey="customer_id">
            <Hierarchy hasAll="true" allMemberName="All Genders" primaryKey="customer_id">
                <Table name="customer"/>
                <Level name="Gender" column="gender" uniqueMembers="true"/>
            </Hierarchy>
        </Dimension>
        <Dimension name="Time" foreignKey="time_id">
          <Hierarchy hasAll="false" primaryKey="time_id">
                <Table name="time_by_day"/>
                <Level name="Year" column="the_year" type="Numeric" uniqueMembers="true"/>
                <Level name="Quarter" column="quarter" uniqueMembers="false"/>
                <Level name="Month" column="month_of_year" type="Numeric" uniqueMembers="false"/>
          </Hierarchy>
          <Hierarchy name="Time Weekly" hasAll="false" primaryKey="time_id">
                <Table name="time_by_week"/>
                <Level name="Year" column="the_year" type="Numeric" uniqueMembers="true"/>
                <Level name="Week" column="week" uniqueMembers="false"/>
                <Level name="Day" column="day_of_week" type="String" uniqueMembers="false"/>
          </Hierarchy>
        </Dimension>
        <Measure name="Unit Sales" column="unit_sales" aggregator="sum" formatString="#,###"/>
        <Measure name="Store Sales" column="store_sales" aggregator="sum" formatString="#,###.##"/>
        <Measure name="Store Cost" column="store_cost" aggregator="sum" formatString="#,###.00"/>
        <Measure name="Sales Count" column="product_id" aggregator="count" formatString="#,###"/>
    <CalculatedMember name="Profit" dimension="Measures" formula="[Measures].[Store Sales]-[Measures].[Store Cost]">
    <CalculatedMemberProperty name="FORMAT_STRING" value="$#,##0.00"/>
    </CalculatedMember>
    </Cube>
</Schema>
```

Figure26 : schéma de cube « sales»

Une partie de cube sales :



Après la connexion à la suite SpagoBI

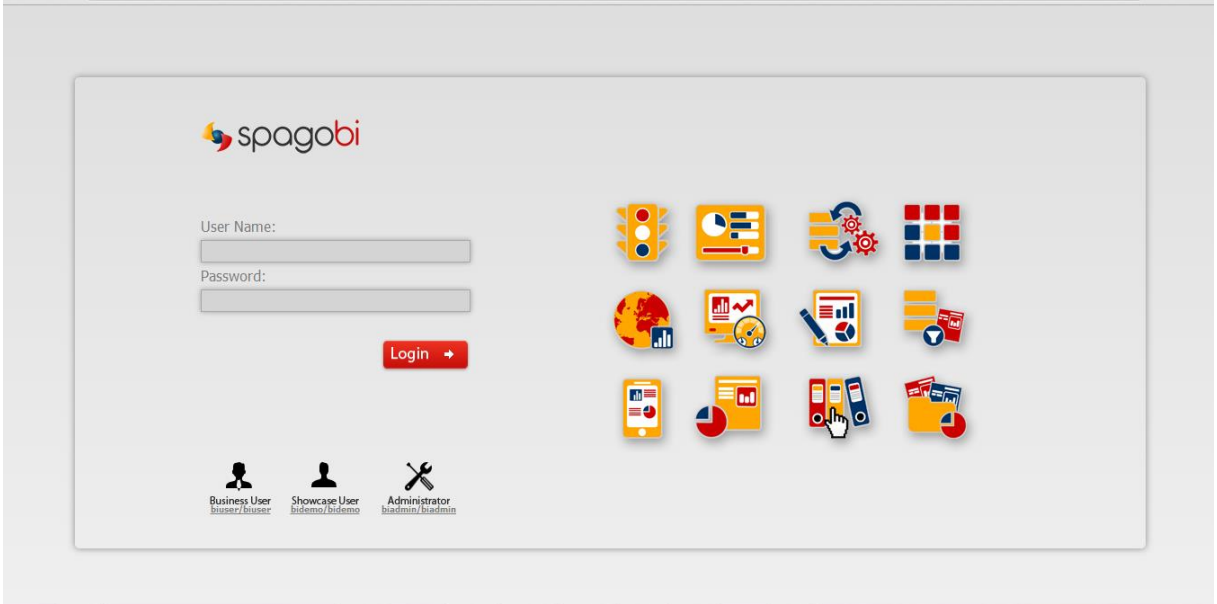
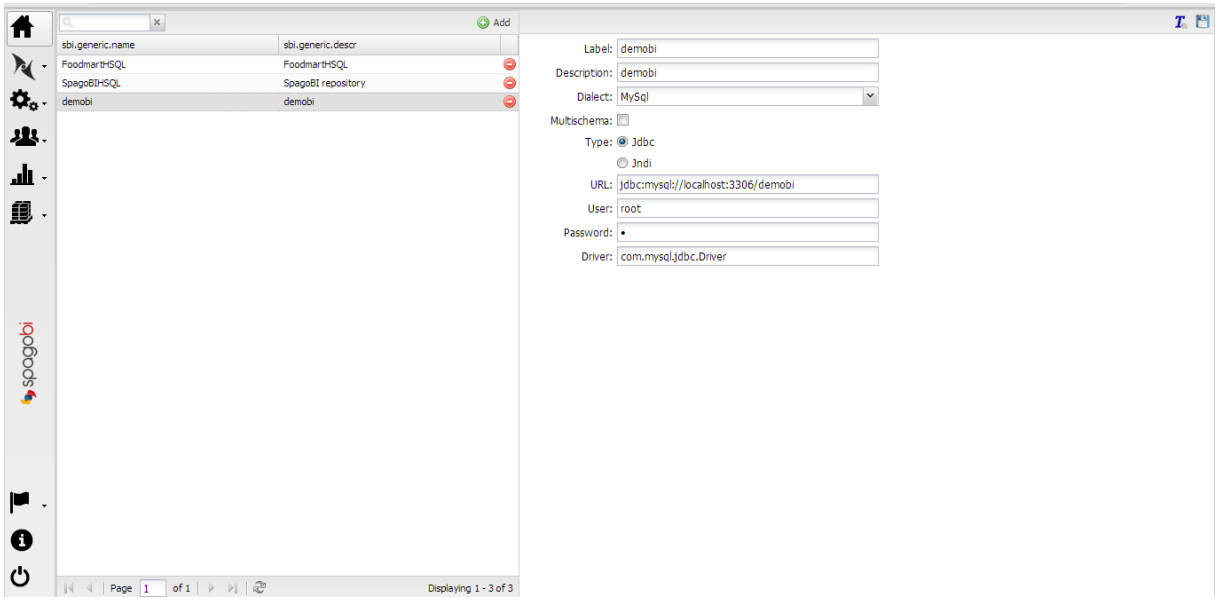
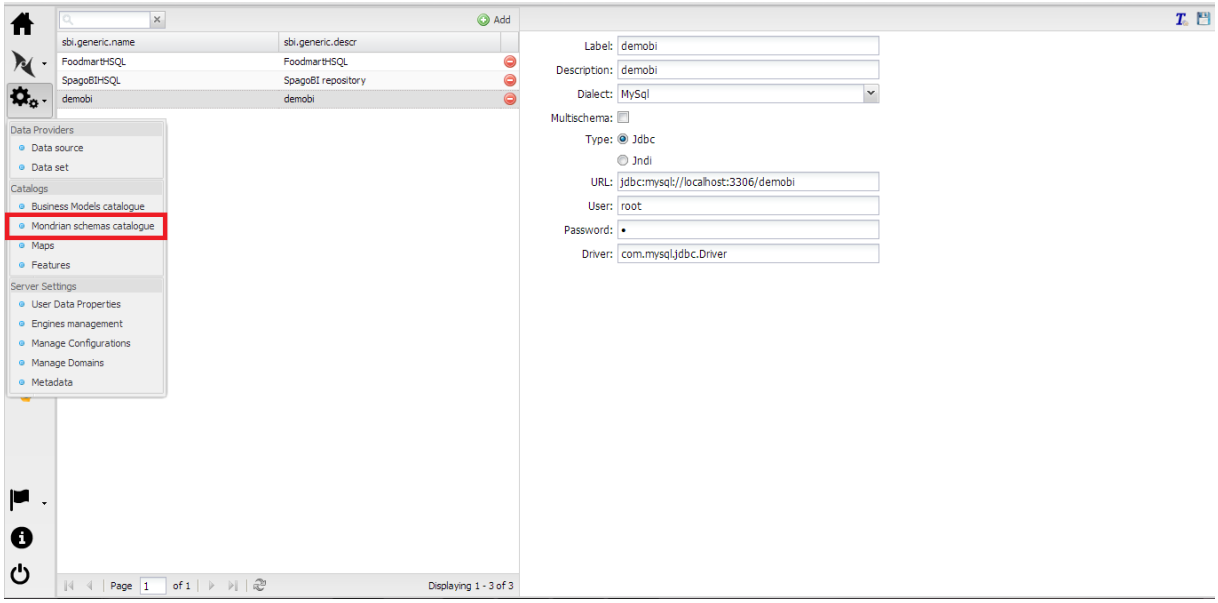


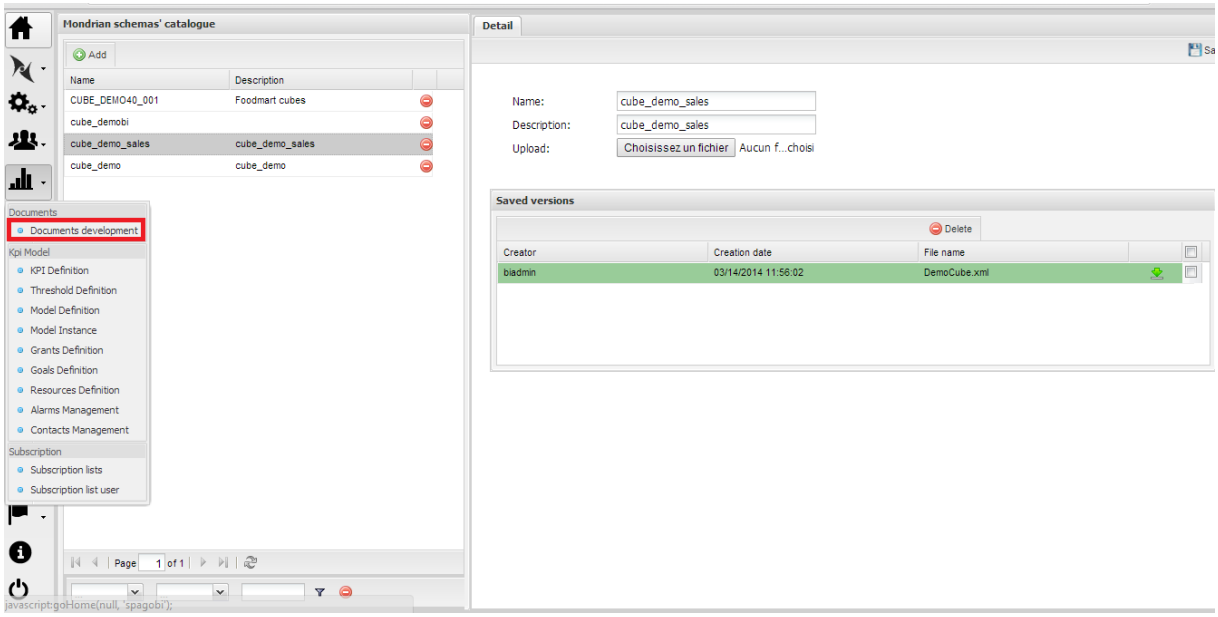
Figure 27 : la page d'accueil de spagobi

On va connecter notre base de données MySql avec SpagoBI, pour faire cela on va dans l'onglet **ressources -> datasource**, et comme montre la figure on va remplir le champ :

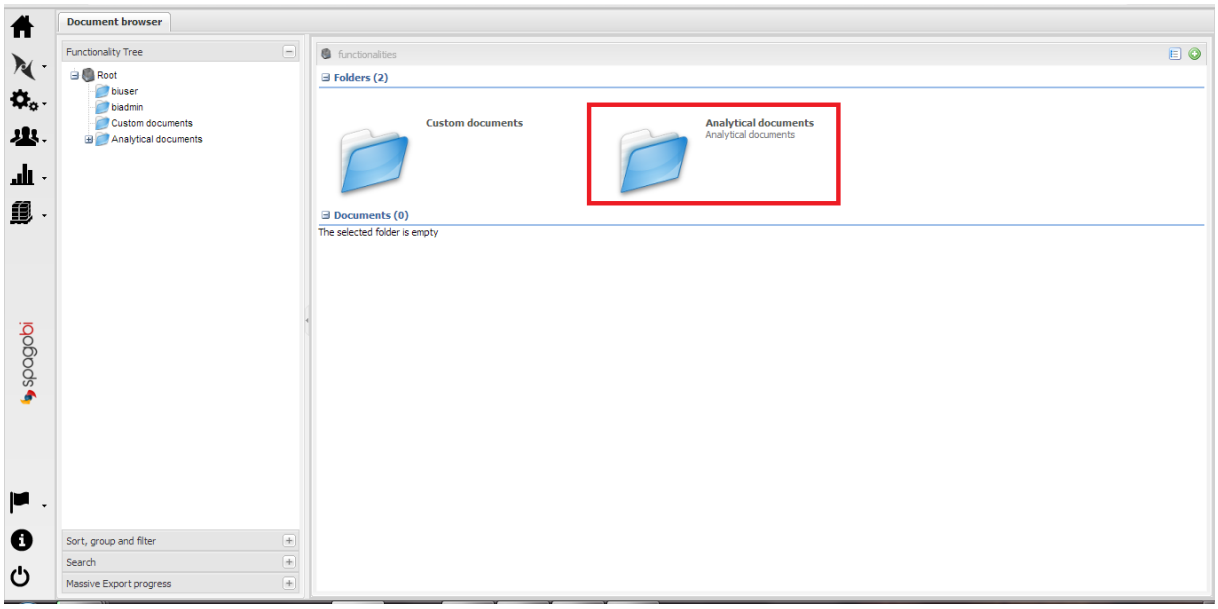


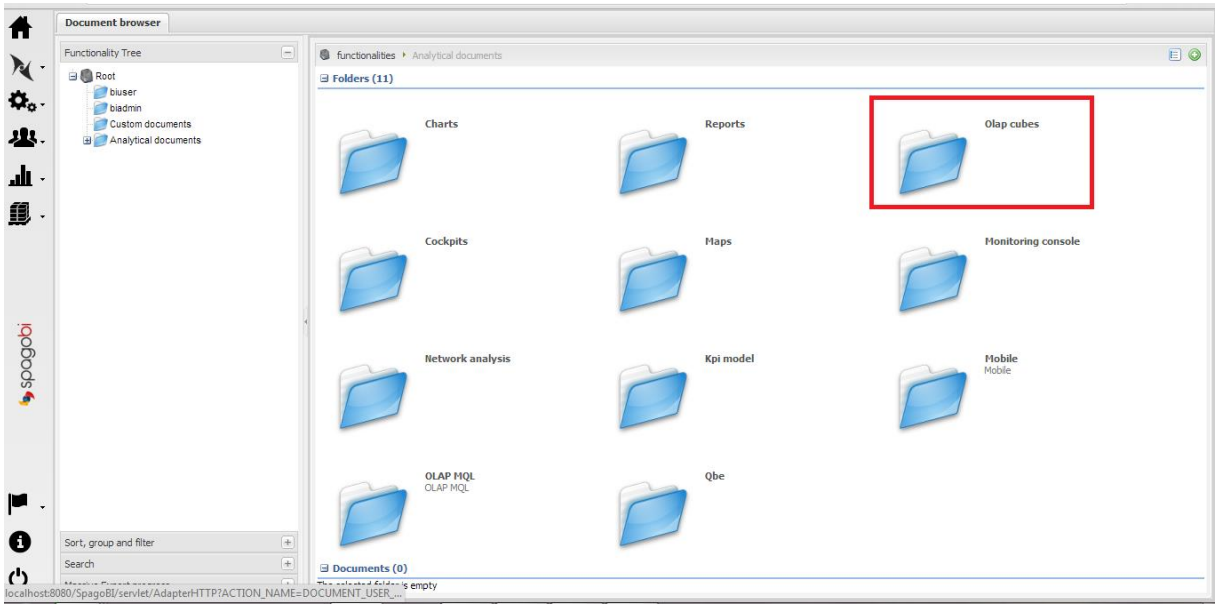
Après on va importer notre shéma.xml, monré au dessus, dans SpagoBI :



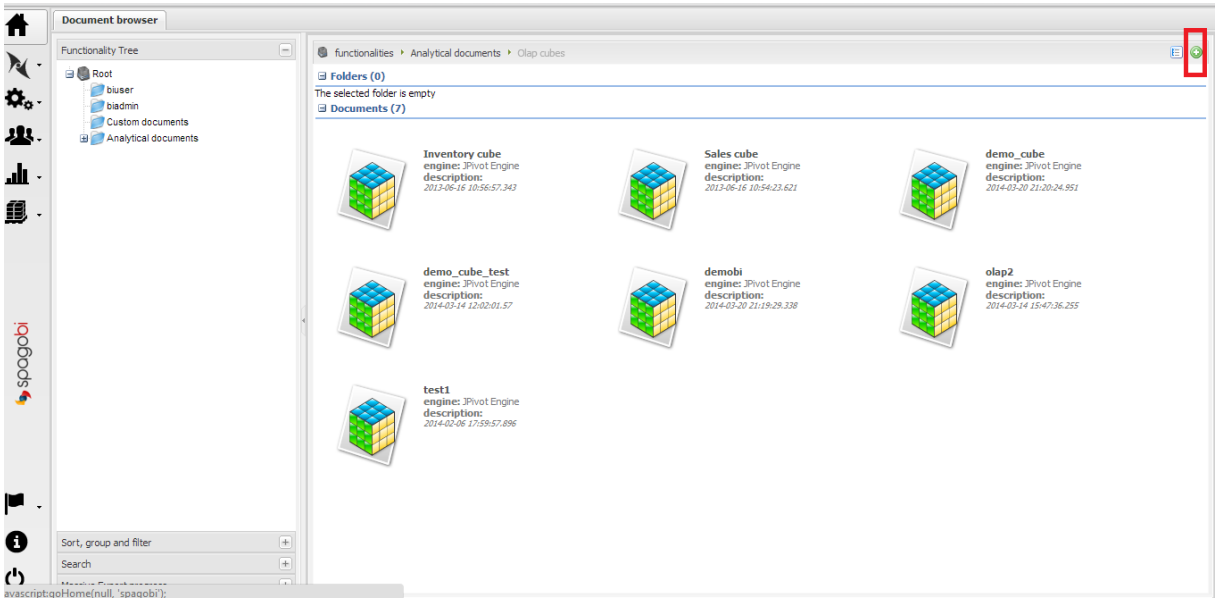


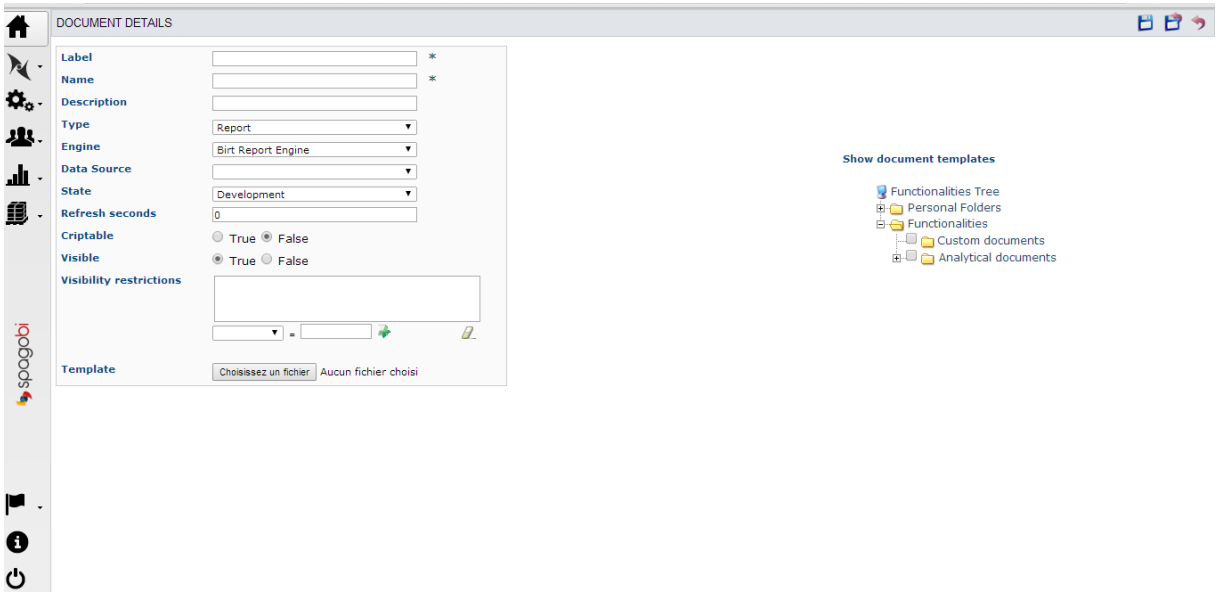
Après l'importation de notre fichier on va maintenant créer notre cube, pour le faire on va sélectionner l'onglet **Document devloppment** puis **Analytical Developpments-> Olap Cubes**



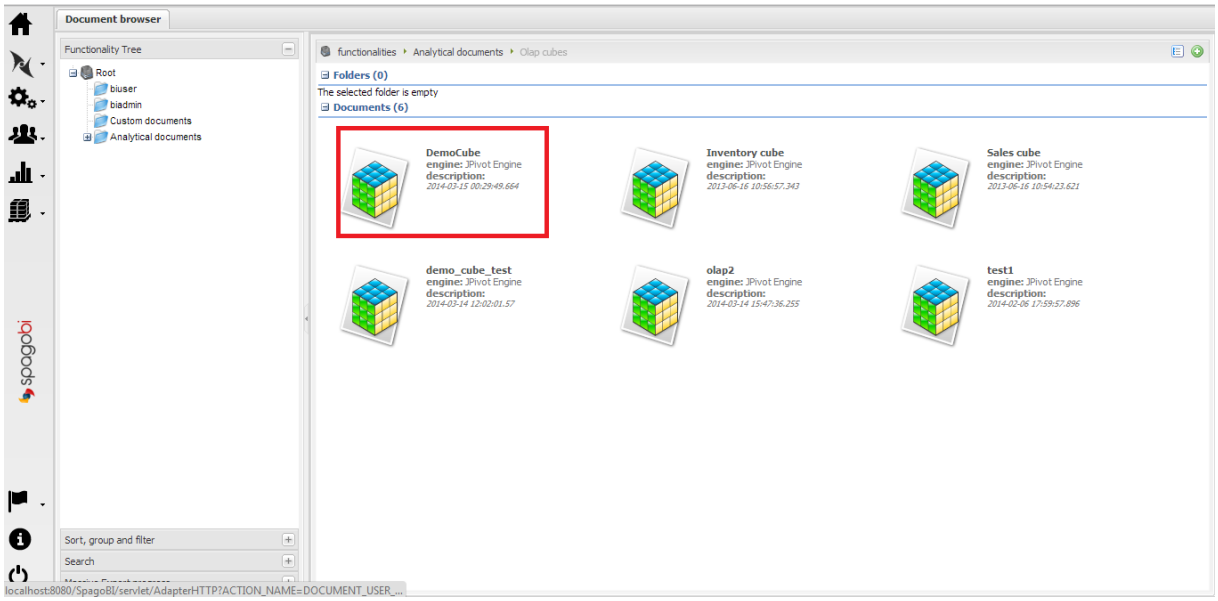


On va ajouter le cube en suivant les étapes suivantes :





Notre cube maintenant est créé :

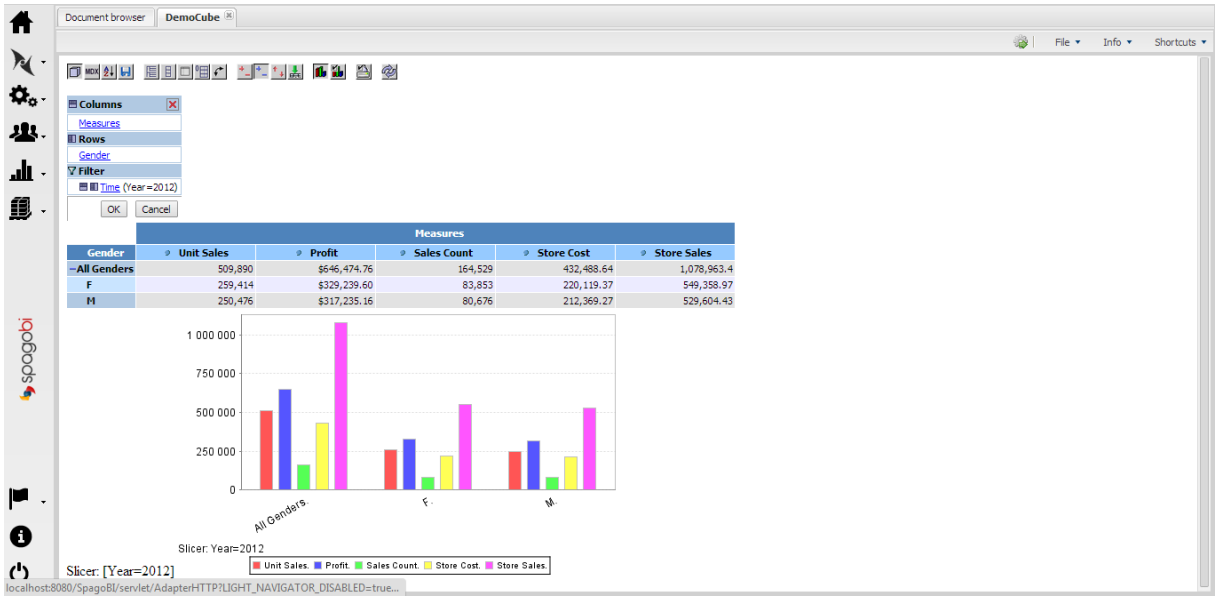


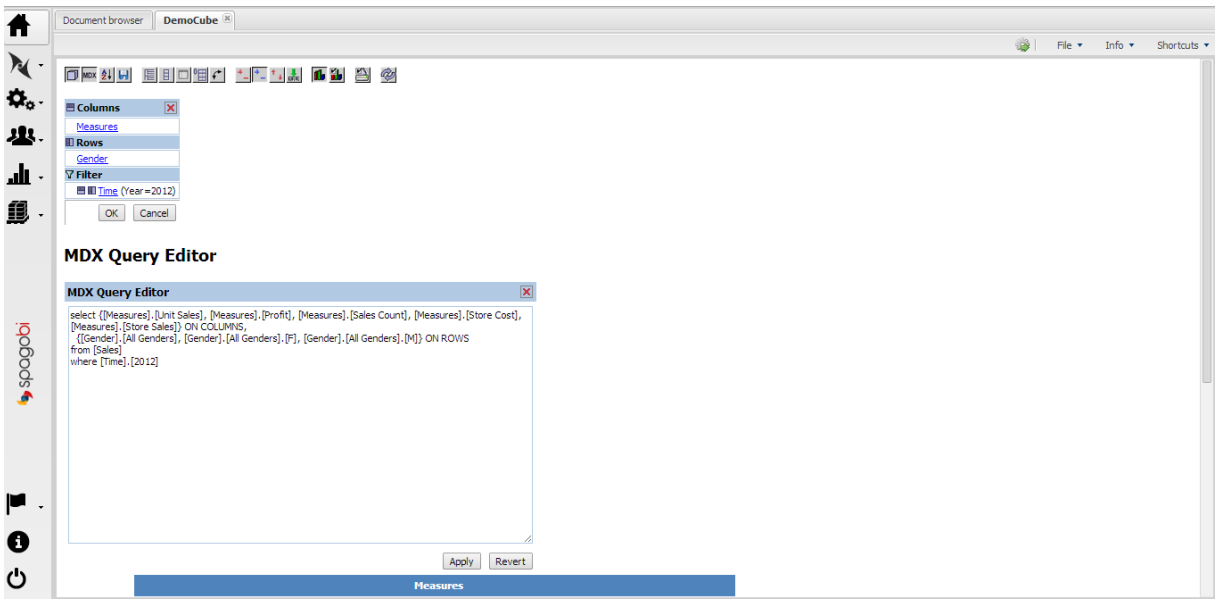
On va l'accéder pour faire nos analyses ; on peut choisir les colonnes et les champs qu'on veut selon nos besoins, écrire les requêtes MDX et choisir le modèle d'affichage qui nous satisfait.

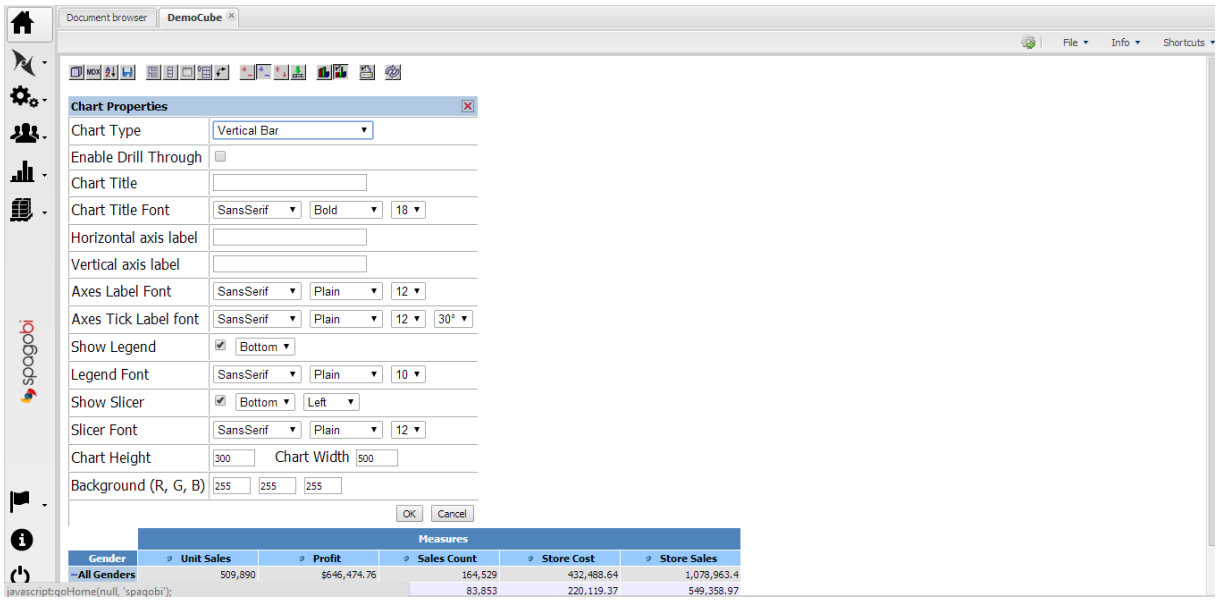
Dans notre exemple on fait l'analyse de ventes d'une compagnie hypothétique et on veut savoir s'il y a une différence entre les hommes et les femmes concernant leur consommation.

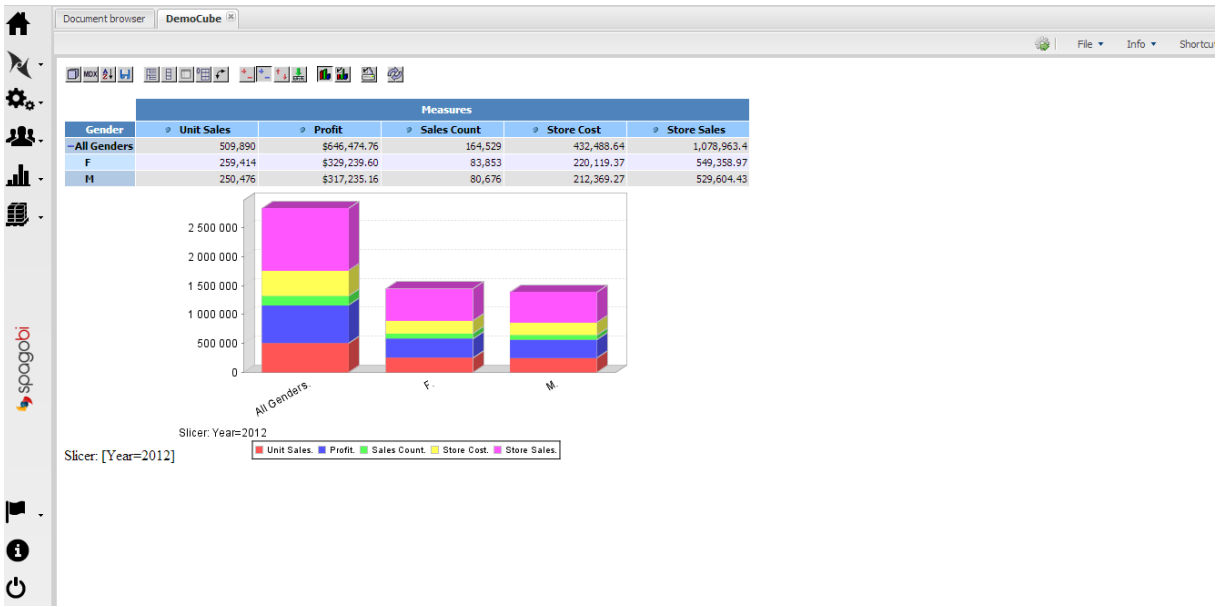
On peut répondre à des questions notamment :

- Quel est la répartition des ventes entre les différents types de produit en 2013 ?
- Quel est le produit le plus vendu en 2013 ?
- Y-a-t-il des variations de consommation saisonnières ou mensuelles ?


Document browser
DemoCube
File
Info
Shortcuts
Columns
Measures
Rows
Gender
Filter
Time (Year=2012)
OK
Cancel
Measures
Gender
Unit Sales
Profit
Sales Count
Store Cost
Store Sales
All Genders
509,890
$646,474.76
164,529
432,488.64
1,078,963.4
F
259,414
$329,239.60
83,853
220,119.37
549,358.97
M
250,476
$317,235.16
80,676
212,369.27
529,604.43
1 000 000
750 000
500 000
250 000
0
All Genders
F
M
Slicer: Year=2012
Slicer: [Year=2012]
Unit Sales
Profit
Sales Count
Store Cost
Store Sales
localhost:8080/SpagoBI/servlet/AdapterHTTP?LIGHT_NAVIGATOR_DISABLED=true...


Document browser
DemoCube
File
Info
Shortcuts
Columns
Measures
Rows
Gender
Filter
Time (Year=2012)
OK
Cancel
MDX Query Editor
MDX Query Editor
select {[Measures].[Unit Sales], [Measures].[Profit], [Measures].[Sales Count], [Measures].[Store Cost],
[Measures].[Store Sales]} ON COLUMNS,
{[Gender].[All Genders], [Gender].[All Genders].[F], [Gender].[All Genders].[M]} ON ROWS
from [Sales]
where [Time].[2012]
Apply
Revert
Measures


Document browser
DemoCube
File
Info
Shortcuts
Chart Properties
Chart Type
Vertical Bar
Enable Drill Through
Chart Title
Chart Title Font
SansSerif
Bold
18
Horizontal axis label
Vertical axis label
Axes Label Font
SansSerif
Plain
12
Axes Tick Label font
SansSerif
Plain
12
30°
Show Legend
Bottom
Legend Font
SansSerif
Plain
10
Show Slicer
Bottom
Left
Slicer Font
SansSerif
Plain
12
Chart Height
300
Chart Width
500
Background (R, G, B)
255
255
255
OK
Cancel
Measures
Gender
Unit Sales
Profit
Sales Count
Store Cost
Store Sales
All Genders
509,890
$646,474.76
164,529
432,488.64
1,078,963.4
83,853
220,119.37
549,358.97
javascript:goHome(null, 'spagobi');


Document browser
DemoCube
File
Info
Measures
Gender
Unit Sales
Profit
Sales Count
Store Cost
Store Sales
All Genders
509,890
$646,474.76
164,529
432,488.64
1,078,963.4
F
259,414
$329,239.60
83,853
220,119.37
549,358.97
M
250,476
$317,235.16
80,676
212,369.27
529,604.43
2 500 000
2 000 000
1 500 000
1 000 000
500 000
0
All Genders
F
M
Slicer: Year=2012
Slicer: [Year=2012]
Unit Sales.
Profit.
Sales Count.
Store Cost.
Store Sales.


Document browser
DemoCube
File
Info
Shortcuts
MDX Query Editor
MDX Query Editor
select {[Measures].[Profit]} ON COLUMNS,
{[Gender].[All Genders].[F], [Gender].[All Genders].[M]} ON ROWS
from [Sales]
where [Time].[2012]
Apply
Revert
Measures
Gender
Profit
F
$329,239.60
M
$317,235.16
300 000
250 000
200 000
150 000
100 000
50 000
0
Slicer: Year=2012
Slicer: [Year=2012]
Profit

Conclusion

Dans le cadre de ce mémoire l'ensemble des données sont rassemblées dans un DATAWAREHOUSE et elles sont modélisées et représentées dans un cube OLAP pour être facilement visible et plus appréhensible pour les utiliser afin de prendre une décision. Cette masse d'information est grande assez importante, cette modélisation permet au décideur non statisticien à y faire face afin de trouver des solutions pertinentes.

Dans ce mémoire on a intéressé par les systèmes OLAP dont les données numériques sont structurées de manière multidimensionnelle, comment on peut intégrer des données complexes telles que les documents dans les systèmes OLAP ?

Bibliographie

Elsa NEGRE. Décembre 2009.. *« EXPLORATION COLLABORATIVE DE CUBES DE DONNÉES », THÈSE pour obtenir le grade de Docteur spécialité informatique,.* l'Université François Rabelais Tours, : s.n., Décembre 2009.

GOUARNE, Jean-Marie. Novembre 1997. *Le projet décisionnel – Enjeux, Modèles, Architectures du Data Warehouse.* s.l. : Editions Eyrolles, Novembre 1997. 246 pages.

KIMBALL, Ralph. Octobre 2000. *Concevoir et déployer un Data Warehouse-Guide de conduite de projet.* s.l. : Editions Eyrolles, Octobre 2000. 576 pages.

Olivier Teste. Décembre 2009,. *«Modélisation et manipulation des systèmes OLAP : de l'intégration des documents à l'usager», MEMOIRE pour l'obtention de l'HABILITATION à DIRIGER des RECHERCHES Spécialité Informatique.* Université Paul Sabatier (Toulouse III), : s.n., Décembre 2009,.

Sites web

https://fr.m.wikipidia.org/wiki/informatique_décisionnelle.

https://www.piloter.org/business-intelligence/business-intelligence.htm.

http://www-igm.univ-mlv.fr/~dr/XPOSE2005/entrepôt/sgbd.html.

https://fr.m.wikipidia.org/wiki/multidimensional_Expressions.

https://fr.m.wikipidia.org/wiki/Spagobi.