Structure de l'entrepôt de données de pilotage

Les objectifs de l'entrepôt de données de pilotage des universités et leurs conséquences sur sa structure et sa construction

1- Objectifs de l'entrepôt de données :

Le premier objectif de l'entrepôt de données est de mettre à disposition des décideurs de l'université des informations synthétiques autour d'indicateurs choisis par eux. En première approximation, cela consiste à **mettre à disposition les données permettant de réaliser des tableaux de bord**. Ces tableaux de bord ont une double vocation, de constat, suivi d'opérations, et de prévision. Idéalement, ils doivent permettre de mettre en évidence les causes de certains faits.

Quelles informations contiennent un **tableau de bord** ? Des **indicateurs** qui sont des nombres caractérisés par des **entités** présentées en ligne ou en colonne. Les indicateurs sont généralement placés au croisement d'une ligne et d'une colonne, représentant le croisement de deux entités.

Par exemple un tableau de bord pourrait donner par composante (UFR) les catégories de personnel, orientées vers le suivi des enseignants, décomposées selon les catégories *enseignant du second degré, maître de conférence, professeur, non enseignant.* Nous aurions alors quatre colonnes correspondant à ces catégories, autant de lignes que d'UFR, et un nombre dans chaque case du tableau ainsi formé, représentant le nombre de personnes de chaque catégorie dans chaque composante.

Que signifie **mettre à disposition :** cela doit intégrer **clarté et rapidité**, en tout cas le plus possible. Il faut sortir du vieux schéma, qui pour toute nouvelle information demandée nécessite une nouvelle étude, et un travail d'élaboration du service informatique, schéma qui conduit forcément à une réaction lente. L'idée "d'entreposer" des données ne doit pas être séparée de

l'idée de les rendre visibles à leurs utilisateurs potentiels. Bien sûr, dans certains cas l'information n'aura pas été entreposée et il y aura alors besoin d'une intervention du service informatique, mais il faut avoir cette exigence de mise à disposition à l'esprit dans les choix de réalisation.

L'objectif **à plus long terme** d'une telle démarche, est que les utilisateurs de ces données, puissent les interroger de façon aussi simple et intuitive que l'on utilise un tableur ou un traitement de textes.

2- Structure mise à disposition des utilisateurs (Univers Business Objects) :

21- définitions :

Tous les outils d'interrogation ("requêteurs", "browsers") servent à extraire des données d'une base. Ils s'appuient sur le langage SQL (Structured Query Language, langage d'interrogation normalisé), qui spontanément liste ou somme, à la demande, toutes les valeurs d'une entité existant dans une table. On comprend assez intuitivement qu'ils peuvent facilement générer des tableaux en énumérant toutes les valeurs de deux entités simultanément.

Pour la suite, je ne vais m'intéresser qu'à un de ces outils d'interrogation, Business Objects, puisque c'est celui qui a été choisi par l'AMUE, et que c'est un bon logiciel dans le domaine, car riche en souplesse et en possibilités.

Dans Business Objects (B.O.), on retrouve les deux "objets" dont nous avons besoin : **indicateur** et **dimension.**

Un **indicateur** est un nombre, une somme, un minimum, ou un maximum, donc toujours un nombre. Cela correspond bien à nos cases de tableau de bord.

Une **dimension**, est l'**entité** dont le tableau va énumérer les valeurs en ligne ou en colonne, elle est la base de l'analyse : par exemple dans le tableau évoqué précédemment, l'analyse se fait selon deux dimensions : la catégorisation de personnel, et la composante (ou UFR). Le terme de dimension convient : on a bien affaire à un tableau à deux dimensions.

Dans Business Objects (B.O.), on peut grouper ces objets, indicateur ou dimension en "classes" ; les classes peuvent elles-mêmes se regrouper en classes. On a donc tout loisir de

faire des regroupements en classes et en sous-classes (tous les outils d'interrogation ne sont pas aussi souples à ce niveau, et il faut s'en méfier). Un ensemble de classes basé sur la connexion à une base de donnée est un "univers", toujours dans la terminologie de B.O.

L'élaboration de tableaux fondée sur le SQL est intrinsèque à B.O., comme à d'autres outils de ce type. Il suffit de faire glisser deux dimensions et un indicateur de la fenêtre de définition de l'univers (fourni à l'utilisateur par le "designer"), vers la fenêtre de création d'un rapport (créé par l'utilisateur), pour obtenir un tableau à deux dimensions.

Une propriété d'une dimension est qu'elle appartient souvent à une **hiérarchie.** C'est un point très important, car ces hiérarchies ordonnent les dimensions et permettent de changer de niveau d'analyse.
Par exemple, dans les inscriptions administratives des étudiants, il existe une hiérarchie qui vient de l'organisation administrative de la scolarité : composante (UFR), cycle, diplôme, étape du diplôme : ces quatre dimensions appartiennent à la même hiérarchie. Si on fait un tableau qui donne le nombre d'inscriptions par composante (UFR) et par année universitaire, on aura un tableau à deux dimensions (composante et temps). Changer de niveau d'analyse consistera à passer à un tableau par cycle et par année, voire par diplôme et par année, puis à remonter au niveau cycle ou composante, etc…
Quel intérêt ? éventuellement de mettre en évidence la cause de certains faits, évoquée dans les objectifs : en effet, si un nombre paraît aberrant, ou au moins surprenant, pour une composante, il sera intéressant de voir si cela vient d'un cycle en particulier, et si oui, peut-être d'un diplôme en particulier. C'est ce que recouvre le terme "analyse dimensionnelle". Il va de soit qu'elle n'est possible que si les dimensions de l'univers mis à disposition sont hiérarchisées, et nous verrons que cela a des conséquences sur la construction des tables de l'entrepôt de données. C'est donc un point très important.

22- Conséquences des objectifs sur les univers Business Objects :

221- Structure de l'univers :

La structure de l'univers doit être claire et logique : le regroupement en classes des objets doit être fait selon la **logique des utilisateurs** et le **bon sens,** et non pas dans la logique de la structure des tables sous-jacentes. Il y a là un premier travail du "designer" d'univers B.O. qui ne doit pas se contenter de la création d'univers par l'assistant automatique.

Chaque donnée ne doit pouvoir être atteinte que **d'une façon** dans l'univers : avoir les mêmes informations accessibles par deux chemins ne peut qu'embrouiller le schéma et l'alourdir. De plus, comme toute redondance, cela augmente les risques d'erreurs au départ et surtout dans les évolutions ultérieures.

L'utilisation des données doit être **sécurisée** : autant il faut profiter de la puissance de SQL, autant il faut se méfier de ses effets de bords ; quand on utilise beaucoup de tables liées par des jointures, certaines de ces jointures peuvent multiplier les résultats par le nombre de lignes d'une de ces tables. C'est de la responsabilité du "designer" Business Object de parer à tous ces effets de bord possibles. Dans un univers correctement construit, toute utilisation de deux objets incompatibles doit être interdite ; il y a plusieurs moyens techniques de s'affranchir de ces inconvénients, il faut toujours choisir le plus simple rendant le service attendu.

Toutes les dimensions qui le peuvent doivent être définies dans des **hiérarchies.**

222- Objets de l'univers :

Toujours dans le double but de mettre à disposition les données, et de faire à long terme de B.O. un outil aussi utilisé qu'un tableur, les objets de l'univers doivent être décrits avec le plus grand soin.

Les noms des objets (et des classes) doivent être pris dans le **langage des décideurs**, et dans le **langage courant**, en fuyant autant que possible le jargon de chaque domaine applicatif qui n'a pas de nécessité absolue. Par exemple une composante (UFR) ne doit pas s'appeler "structure de niveau 2", et encore moins "niveau 2".

Les noms des objets doivent être **normalisés**, tous les codes notés d'une certaine façon, tous les libellés d'une autre ; on aura par exemple "Nationalité - Code" pour le code de la nationalité, "Nationalité - Lib" pour son libellé ; en cas de libellés court et long, on utilisera Libc et Lib, les nombres ne seront pas suffixés, mais commenceront aussi par une majuscule, etc…

Chaque dimension doit avoir une **liste de valeurs** associée : une liste de valeurs permet, par une liste déroulante, de choisir une valeur par le libellé associé à un code. Elle est l'arme d'un choix clair d'une condition : grâce à une liste de valeurs, un utilisateur peut choisir une catégorie comme "les enseignants du second degré" sans connaître la codification correspondante. C'est la liste de valeur qui fera l'association entre le code et le libellé choisi.

3- Structure générale de la base de l'entrepôt de données :

Toutes les caractéristiques des tables d'un entrepôt de données sont liées à ce que l'on veut en faire : produire des **indicateurs au carrefour de dimensions**.

31- Table de "fait" :

Une table de base de l'entrepôt va comprendre au moins un indicateur élémentaire et des dimensions. Cet indicateur élémentaire peut être un nombre entier compris entre 0 et 1, permettant de compter un élément, ou un nombre entier ou décimal représentant une quantité. Par exemple, dans les inscriptions administratives des étudiants, un indicateur d'inscription administrative sera à 1 pour une composante (UFR), un cycle, un diplôme, une étape, et un code étudiant si et seulement si l'étudiant qui a ce code est inscrit à cette étape de ce diplôme, de ce cycle, de cette composante. Si c'est la composante 924, le cycle 3, le diplôme 580, l'étape 2, l'étudiant 19930003 la table contiendra 924,3,580,2,19930003,1.
Pour prendre un autre exemple, dans les postes affectés, la quotité de travail sera à 0,6 pour une composante, une structure d'affectation, une occupation, un numéro d'affectation correspondant à un contrat de travail ou un élément de carrière, et un code personnel, si et seulement si la personne qui a ce code travaille à 60% pour ce numéro d'affectation, ce poste, cette structure d'affectation, cette composante.

Une table de ce type est appelée "table de fait". **Un "fait" est représenté par la valeur d'un indicateur élémentaire associé à une série de dimensions** : il est numérisé pour pouvoir être utilisé. Il est la base de l'indicateur de tableau de bord que nous visons, qui sera une somme, une moyenne, un maximum ou un minimum des indicateurs élémentaires.

Dans notre exemple précédent, le "fait" est l'inscription administrative à une composante (UFR), un cycle, un diplôme, une étape. Ces entités, qui caractérisent le fait, sont des "dimensions".

32- Table de "dimension" :

Une table de dimension contient pour chaque dimension, au moins un libellé. Par exemple la table des composantes (ou UFR), contiendra, pour chaque code, le libellé associé : 901,Services Généraux ; 924,UFR de Mathématiques, etc…

C'est ainsi que l'on obtient un modèle en **"étoile"** typique des entrepôts de données, avec une table de fait au centre, et les tables de dimensions liées à chaque dimension de la table de fait.

4- Construction et mise à disposition de l'entrepôt de données :

Le **couple indicateur-dimension** est le pivot central de la construction de l'entrepôt. Il traverse tout le processus depuis le document recherché par l'utilisateur-décideur jusqu'à la structure de la base, en passant par l'univers Business Objects.
C'est le point essentiel. Si nous n'en tenons pas compte, nous restons dans le vocabulaire des entrepôts de données mais pas dans leur réalité. Une fois déterminés les indicateurs recherchés, et les dimensions qui les caractérisent, ainsi que leurs hiérarchies, et donc le schéma des tables de la base, il reste à définir la dimension historique de leur alimentation et à répartir les travaux nécessaires à la construction de l'entrepôt et à sa mise à disposition entre les outils dont nous disposons, et entre les différents intervenants.

41- Dimensions historiques :

L'entrepôt de données est par essence destiné à durer, toutes les tables de fait doivent donc avoir une dimension historique.

Les faits eux-mêmes sont historiques : un étudiant s'inscrit pour une année universitaire, quand on parle d'un budget, on s'intéresse à un exercice comptable. Il y a donc une dimension historique incluse dans les faits (année, trimestre, date, selon les faits mesurés).

D'autre part l'entrepôt se présente comme une succession de photographies des faits, prises à différents moments. Il faut donc inclure dans chaque table, la date de "prise de vue", et on en tiendra compte pour toute interrogation.

Il y a donc **deux dimensions historiques :** une inhérente aux faits eux-mêmes, l'autre provenant du processus d'alimentation de la base de l'entrepôt.

42- Répartition des tâches entre les outils :

Un premier travail consiste à extraire des données des bases des différents domaines d'activité (personnel, scolarité, financier, …), pour alimenter la base de l'entrepôt de données. L'outil pour ce travail est un "ETL", langage d'extractions de données. Certaines données sont transférées d'une base à l'autre, sans transformation, et ne posent pas de problème. D'autres, par contre, sont élaborées par des règles à partir des données préexistantes. Il est important que tout le **travail de transformation ne soit fait qu'une fois, et au niveau du travail d'alimentation,** et ne soit pas déporté au niveau de l'univers Business Objects. D'autant que les données sont historisées et archivées : il est important que cet archivage inclue leur définition.

Le schéma général de construction et de mise à disposition de l'entrepôt est donc :

1- Définition des faits à mesurer : quels sont les **indicateurs et les dimensions**.

2- Création des tables nécessaires en respectant indicateurs et dimensions, en **hiérarchisant les dimensions** : l'outil ici est le langage de définition de la base de données (Data Definition Language de SQL), qui peut être généré par un outil de modélisation comme Power AMC, avec l'avantage de pouvoir alors documenter chaque colonne (d'y noter sa règle d'obtention, si complexe soit-elle).

3- Extraction de données des bases de gestion, hiérarchisation de ces données, combinaison éventuelle de ces données par des **règles précises** : l'outil peut être SQL, mais un outil d'extraction de données (ETL) est préférable pour la rapidité de mise en œuvre, la maintenance et l'évolution des scripts. Data Stage d'Informix, et Genio de Hummingbird sont des produits ce type.

4- Création d'un environnement d'exploration (univers Business Objects) : rangement des données selon la **logique des utilisateurs** et le **bon sens, sécurisation** (éviter les effets de bords du SQL), **utilisation des hiérarchies, codifications des noms des objets et classes,** mise en place de **listes de valeurs** pour toutes les dimensions : l'outil est ici Business Objects Designer.

5- Création de rapports périodiques prédéfinis : l'outil est ici Business Objects (user).

6- Exploration des données, création de rapports à la demande : l'outil est ici Business Objects (user).

43- Répartition des tâches entre les différents acteurs :

Le premier point (définition des faits à mesurer) est à la charge des utilisateurs-décideurs : ce sont eux qui définissent de quels tableaux de bord ils ont besoin.

Le deuxième point (structure de la base) est à la charge des informaticiens.

Le troisième point (alimentation de la base) est à la charge des informaticiens, mais le choix des données et la définition des règles d'élaboration des données complexes se fait en étroite collaboration avec les utilisateurs des bases de gestion.

Le quatrième point (définition de l'univers B.O.) est à la charge des informaticiens. Quoiqu'on puisse en dire, et même chez B.O., toute autre approche me paraît démagogique, car, comme nous l'avons vu précédemment, le travail de "designer B.O." demande une bonne connaissance de l'informatique.

Le cinquième point (création de rapports prédéfinis) peut être à la charge des informaticiens, si les tableaux ont fait partie du "cahier des charges", mais ils peuvent aussi être faits par des utilisateurs expérimentés.

Le sixième point (exploration, ..) est bien sûr l'outil des utilisateurs.

5- Comment construire l'entrepôt de données :

Ce chapitre reprend et développe ce qui a été abordé au chapitre 3 "Structure générale de l'entrepôt de données".
Son objet est de répondre à la question "**comment faire ?**", "comment séparer et organiser les données de l'entrepôt ?".
Nécessairement plus technique, ce chapitre est donc destiné principalement aux informaticiens.
Les paragraphes 51 et 52 sont les compléments respectifs des paragraphes 31 et 32.

51- Conditions de création d'une table de fait :

Le même "fait" apparent, comme l'inscription d'un étudiant, peut être lié à des dimensions sans rapport entre elles : l'inscription d'un étudiant à un élément pédagogique, à un module d'enseignement, est une conséquence de son inscription administrative à une étape d'un diplôme. Mais la réciproque est fausse : un même élément pédagogique peut être suivi dans le cadre de diplômes différents. Si l'on utilise une seule table de fait pour représenter les

inscriptions pédagogiques et administratives, le domaine de définition d'un indicateur pose problème : en effet, si notre table de fait inclus tous les éléments pédagogiques suivis par un étudiant, l'indicateur d'inscription administrative ne devra être mis à 1 qu'une fois par étape de diplôme, donc pour un seul élément pédagogique. Nous avons alors tout intérêt à avoir deux tables de fait. En généralisant, si des dimensions sont dans des hiérarchies différentes, on a intérêt à faire des tables de fait séparées. **Un fait peut donc être représenté comme un indicateur élémentaire associé à une hiérarchie de dimensions.**

Cela n'empêche pas que d'autres dimensions hiérarchisées complémentaires soient liées à ce fait. C'est le cas de dimensions complètement **indépendantes du fait lui-même :** par exemple la nationalité de l'étudiant peut désigner un pays qui appartient à une communauté (Europe ou non), c'est une dimension hiérarchisée, cela ne change rien à son inscription administrative ou pédagogique. Par contre, nous l'avons vu, une dimension liée au même fait mais n'appartenant pas à une même hiérarchie pose problème. Nous disposons donc maintenant d'un critère de décision pour créer une table de fait : nous créerons **une table de fait pour chaque hiérarchie de dimension intrinsèque au domaine mesuré** (liée à un indicateur au moins).

52-Structure des tables de dimension :

Si une dimension est hiérarchisée, sa table contiendra non seulement un libellé, mais en plus un renvoi vers la dimension de niveau supérieur. Par exemple la nationalité de code 100, aura pour libellé "France", et un code de communauté "UE", code de l'Union Européenne dans la table des communautés, dimension de niveau supérieur. On obtient alors un schéma "en flocon".

Dans la pratique, pour éviter de multiplier les jointures, pour des raisons de performance, on se ramène souvent au schéma en étoile en "dénormalisant" les tables de dimension hiérarchisées (le terme vient de la négation de la forme "normale" du modèle relationnel). Il y a des informations répétées dans plusieurs lignes, à la place d'une table distincte. Dans notre exemple, on aura une table de dimension unique contenant la hiérarchie. En face du code nationalité 100, on aura le libellé "France", le code communauté "UE", et le libellé "Union Européenne" ; ce code "UE" et son libellé seront répétés pour tous les pays de l'union européenne, et tous les autres pays auront un autre code, par exemple "ET", avec comme libellé "Etranger".

53- Bien séparer indicateur et dimension :

Il faut prendre garde à ne pas faire glisser une dimension en un fait, à ne pas multiplier des indicateurs qui suivent les valeurs d'une dimension, car on se prive alors de la puissance du langage SQL. Par exemple, si je crée un indicateur "européen" et un indicateur "étranger" dans la table des inscriptions administratives, tout nouveau pays européen m'oblige à revoir l'alimentation de ma table, alors que si je passe par la nationalité et la dimension communauté, je n'ai qu'une ligne de la table des nationalités à modifier. Je peux aussi m'adapter à tout regroupement de pays différent, par simple action sur cette table. De la même manière, dans les postes affectés, si je crée des indicateurs "enseignant du second degré", "professeur", … au lieu d'utiliser un témoin à valeur multiple à côté de l'indicateur quotité de travail, le jour où il faut tenir compte d'une nouvelle catégorie d'enseignant, je suis bloqué, alors que dans l'autre cas, je n'ai qu'une ligne à ajouter dans la table des catégories d'enseignants.

En résumé, il faut bien un indicateur au moins qui mesure le fait étudié (inscription administrative, ou quotité de travail, par exemple), mais il ne faut pas projeter une dimension dans la table de fait sous formes d'indicateurs qui ne mesurent plus alors le fait brut, mais la combinaison de ce fait avec une de ses dimensions.

54- Indicateurs définis dans Business Objects :

Certains indicateurs élémentaires peuvent ne pas exister dans la table de fait et être définis dans l'univers Business Objects. Ce n'est possible que s'ils peuvent s'obtenir par un ordre SQL simple, utilisant une "fonction de regroupement" **sans clause restrictive** ("*where*") : s'ils sont obtenus par un "*select count(...)", "select count (distinct ...)", "select sum(...)"*, ... Ainsi, si l'on s'intéresse au nombre d'étudiants physiques, dans le domaine des inscriptions administratives des étudiants, on pourra définir un indicateur dans B.O. comme un "*count distinct*" des numéros d'étudiants, qui permet de ne compter que les numéros d'étudiants distincts, donc de ne compter qu'une fois un étudiant qui a plusieurs inscriptions administratives. Par contre tout indicateur qui s'obtiendrait par une clause restrictive est à proscrire ; en effet, il n'est pas combinable dans des calculs (de type ratio par exemple), avec d'autres indicateurs issus de la même table car la clause restrictive joue sur les deux nombres et fausse le résultat.

55- Et les agrégats ?

Les tables d'agrégats sont des tables dans lesquelles une partie des additions, moyennes, maxima ou minima est calculée par avance. Il ne s'agit que d'une optimisation éventuelle de performance au cas où une table de fait contiendrait énormément de lignes. On va alors "regrouper" des lignes, **par des ordres SQL simples,** basés sur les fonctions somme, moyenne, maximum, minimum, ... Avec la fonction "aggregate aware" de Business Objects, ces agrégats peuvent et doivent rester **invisibles à l'utilisateur**. Selon les objets demandés dans un rapport, B.O. va effectuer sa requête à la base de données en utilisant la table de fait, ou un agrégat déclaré par "aggregate aware" de façon à extraire de la base le moins de lignes possibles. Cela nécessite simplement que les agrégats sur une table de fait soient créés **en respectant la hiérarchie de la table de fait.**

6- Conclusion :

J'espère que cet exposé permettra de clarifier et de concrétiser la notion d'entrepôt de donnée des universités, et aidera à comprendre ce qui est amorcé et à le développer. J'ai essayé en même temps de prendre des exemples du monde universitaire, et de généraliser, essayé d'en extraire une méthode pour répondre à la question : "comment faire, comment choisir les tables de l'entrepôt ?". J'ai essayé de montrer comment le couple indicateur-dimension est à la base de tout l'édifice, ce qui n'est d'ailleurs pas une nouveauté. Mais, comme toujours, la théorie sert surtout à ceux qui l'élaborent, car elle leur permet de clarifier leur pratique, et il est bien difficile de faire siennes les idées des autres quand on n'a pas leur pratique. Ce n'est que la confrontation **pratique** à notre propre expérience (locale à Paris 6), puis la confrontation avec l'expérience de l'équipe de Grenoble dans le cadre du projet de l'AMUE, qui m'a permis de, et parfois obligé à, clarifier certaines notions intuitives ou empruntées à des ouvrages en la matière. Par contre, en retour, la clarification permet de faire **consciemment certains choix concrets**, et c'est en cela qu'elle est utile.

Pour ce qui est du processus en cours, je suis bien conscient, que même si je me suis trouvé dans la situation qui m'a permis d'effectuer cette clarification, la source en est cette expérience dans laquelle nous sommes nombreux à être engagés collectivement. Je suis bien conscient aussi, que dans tout ce que j'affirme ici, la pratique ultérieure montrera probablement des

erreurs. La pratique se développant, il est à souhaiter que d'autres pages "théoriques" soient écrites, en écho à la solution de problèmes probablement de plus en plus complexes.

Paris, le 9 Avril 2001, *Jean-Baptiste Nataf*