Chapitre 1

Introduction aux bases de données

1. Qu'est-ce qu'une base de données ? 2
2. Évolution des bases de données et de leur utilisation 4
3. Systèmes de gestion de bases de données 13
4. Étapes de la conception des bases de données 17
5. « Métiers » des bases de données 19
6. Plan de l'ouvrage 20
7. Présentation de la BD exemple ... 20

Exercices

1. Notion de base de données..... 22
2. Recherche dichotomique 22
3. Langages d'un SGBD..... 23
4. Modèles de représentation 23
5. Métiers des bases de données 23
6. Utilisateurs d'une base de données 24
7. Vues externes 24
8. Base de données réparties 25
9. XML 25
10. Fouille de données et entrepôts de données 26

Au cours des dernières années, les bases de données ont connu un développement considérable, au point qu'elles jouent désormais un rôle dans chacune de nos opérations quotidiennes – du simple achat effectué avec sa carte bancaire jusqu'à nos déclarations de revenus.

L'objectif de ce chapitre est de définir la notion de base de données ainsi que les principaux concepts qui s'y rattachent. La méthodologie qui permet de les concevoir, les applications informatiques associées à leur mise en œuvre (SGBD) et les différents métiers des bases de données y sont présentés.

1 Qu'est-ce qu'une base de données ?

Le nombre d'informations disponibles et les moyens de les diffuser sont en constante progression. La croissance du *World Wide Web* a encore accru ce développement, en fournissant l'accès à des bases de données très diverses avec une interface commune. Celles-ci se situent au cœur de l'activité des entreprises, des administrations, de la recherche et de bon nombre d'activités humaines désormais liées à l'informatique.

Dans le domaine purement informatique, elles interviennent dorénavant à tous les niveaux. Les développeurs d'applications s'appuient sur des bases de données externes pour gérer leurs données alors qu'auparavant elles étaient intégrées dans le programme. Citons un autre exemple : la gestion des fichiers dans les nouveaux systèmes d'exploitation (par exemple, Panther de chez Apple™ ou le futur Vista de Microsoft™) évolue vers de véritables bases de données mises à jour en permanence. Elles permettent de retrouver les fichiers instantanément, par leur nom mais aussi par leur contenu, à la manière d'un moteur de recherche.

Les **bases de données** reposent sur des théories solides et sont à l'origine d'une des plus importantes disciplines de l'informatique : l'**ingénierie des systèmes d'information**. Cette section présente une idée intuitive de ce qu'est une base de données, de son utilisation puis des éléments de qualité qui lui sont associés.

1.1 Notion de base de données

Tout le monde a une idée naturelle de ce que peut être une base de données : elle peut revêtir la forme d'une liste de CD contenant le nom des artistes et les titres des morceaux ou encore celle de fiches de recettes de cuisine.

On remarque qu'une caractéristique des données contenues dans une base de données est qu'elles doivent posséder un lien entre elles. En effet, des données choisies au hasard ne constituent certainement pas une base de données. Celle-ci est donc une représentation partielle et (très) simplifiée du monde réel, que l'on a obtenu par un processus de **modélisation**. En résumé, on définit une base de données comme l'ensemble des données stockées. Pour les manipuler, on utilise généralement un logiciel spécialisé appelé SGBD (*Système de Gestion de Bases de Données*). Il y a parfois confusion, par abus de langage, entre base de données et SGBD. On appelle aussi « système d'information » l'ensemble composé par les bases de données, le SGBD utilisé et les programmes associés. Plus formellement, on appelle ***Base de Données*** (BD) un ensemble de fichiers – informatiques ou non – structurés et organisés afin de stocker et de gérer de l'information.

1.2 Utilisation d'une base de données

La création d'une base de données recèle un but précis : elle doit permettre de retrouver de l'information par son contenu en se fondant sur des **critères de recherche**. On désire, par exemple, retrouver toutes les recettes qui nécessitent des œufs ou tous les CD qui contiennent un morceau donné.

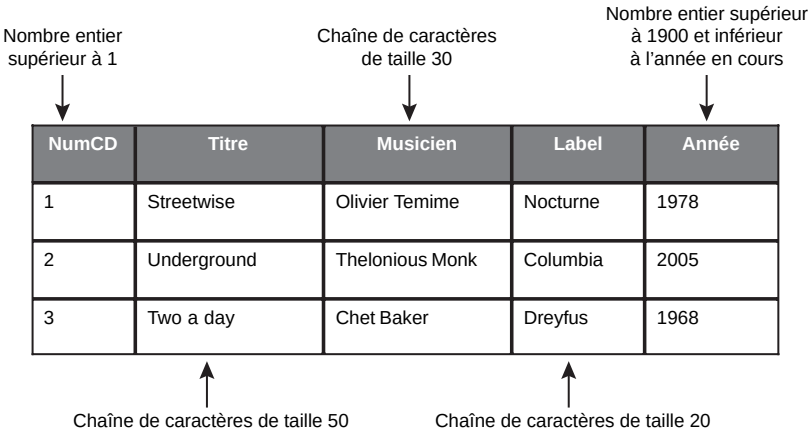
La grande différence avec un programme écrit dans un langage de programmation est qu'une base de données doit pouvoir répondre à des questions pour lesquelles elle n'a pas forcément été prévue à la conception.

Une autre différence est que les données sont susceptibles d'être utilisées par des applications différentes. Dans un programme classique, la structuration des données est décrite

directement dans le code, ce qui rend leur utilisation difficile par d'autres programmes, en particulier lorsque l'on modifie cette structure. Ce que l'on recherche en utilisant une base de données est d'assurer l'indépendance entre le traitement et les données. C'est pourquoi, il est nécessaire que l'application obtienne des informations sur la structure des données (nom, type, taille, etc.). Pour ce faire, on associe à la base de données une description que l'on appelle « **métadonnée** » ou « **catalogue** ». Cette dernière décrit la structure interne de la base de données qui est spécifique au SGBD employé (voir figure 1.1). En plus de la structure et du type des données, on stocke également à cet endroit les informations concernant les règles de cohérence des données abordées à la section suivante.

Figure 1.1

Base de données de CD et métadonnée.



L'idée générale est que l'utilisateur ou l'application utilisatrice des données ne doit pas être dépendante de leur représentation interne, ce qui constitue une abstraction des données. C'est la raison pour laquelle on utilise une description des données sous la forme d'un modèle pour permettre la restitution la plus efficace possible de l'information.

1.3 Qualité d'une base de données

Comme on l'a évoqué précédemment, l'un des objectifs de création d'une base de données est de pouvoir retrouver les données par leur contenu. Dans cette optique, il faut s'assurer que les données contenues dans la base sont de « bonne qualité ».

Comment définir la qualité des données ? De nombreux critères peuvent être pris en compte ; on peut citer parmi les principaux :

- la cohérence des données contenues dans la base ;
- l'absence de redondance.

La **cohérence des données** est fondamentale ; elle nécessite une réflexion préalable sur la normalisation du contenu des champs. On suppose qu'un champ contient la qualité d'une personne (par exemple, Monsieur, Madame, Mademoiselle). Si l'on trouve dans ce champ 'Mr' à la place de 'Monsieur', il est clair que les recherches sur ce champ par le contenu 'Monsieur' risquent d'être erronées. Dans ce cas, les informations seraient moins nombreuses que celles obtenues avec le contenu correct. On qualifie cet état de fait de « **silence** », qui signifie que certains résultats pertinents sont ignorés lors d'une interrogation.

Dans un autre cas, si l'on saisit 'Mme' pour 'Madame' et 'Melle' pour 'Mademoiselle', et qu'il y ait eu par erreur plusieurs saisies de 'Mme' alors qu'il s'agissait d'une demoiselle, la recherche par le contenu 'Mme' donne cette fois plus de résultats qu'il n'y a réellement de dames. On qualifie cet état de fait de « **bruit** », qui signifie que certains résultats non

pertinents sont retournés lors d'une interrogation. La **redondance** est parfois plus délicate à identifier. Si l'on considère le cas très simple d'un carnet d'adresses qui contiendrait en même temps le code postal et le nom de la ville, elle est ici évidente.

Tableau 1.1

Exemple de redondance de l'information.

Nom	Téléphone	Ville	Code postal
Jaco	0668541087	Bordeaux	33000
Stanley	0654789254	Nancy	54000
Marcus	0658741263	Bordo	33000
Charles	0639517720	Nancy	54000
Steve	0659874120	Boredeaux	33000

On remarque que l'on stocke plusieurs fois la même association d'information (par exemple, Nancy et 54000), ce qui consomme de la place inutilement et peut devenir significatif lorsque la base atteint quelques millions d'enregistrements.

De plus il existe des incohérences dans la saisie du nom de la ville 'Bordeaux'. La recherche par le nom 'Bordeaux' ne donnera pas le même résultat que la recherche par le code '33000'.

On verra plus loin que l'approche relationnelle procure des outils capables de détecter et d'améliorer considérablement ce genre de problèmes de qualité des bases de données.

2 Évolution des bases de données et de leur utilisation

Le développement des bases de données s'étend sur une période d'une quarantaine d'années. Cette section présente tout d'abord le contexte dans lequel elles se sont développées. On aborde ensuite les modèles utilisés successivement pour les représenter. Enfin, la dernière partie présente une vue prospective des changements dans l'utilisation des bases de données.

2.1 Contexte

À partir des années 1960, les ordinateurs évoluent rapidement. Ils sont de plus en plus performants mais aussi de plus en plus répandus du fait de leur coût plus raisonnable. Leur utilisation change également ; on passe de la notion de calculateurs purs à des machines capables aussi de traiter de l'information. On parvient à un niveau d'abstraction supplémentaire par rapport aux machines et on obtient en conséquence une indépendance par rapport à l'architecture et surtout par rapport aux constructeurs.

La décennie des années 1970 est une période « faste » pour la recherche et l'innovation en informatique dont les résultats sont encore utilisés aujourd'hui. On peut utiliser des langages de programmation de haut niveau, afin de guider, voire de façonner, la démarche du programmeur (Pascal), et l'on envisage des systèmes d'exploitation indépendants de la machine employée (Unix). C'est également à cette époque que l'on pose les fondements des techniques qui sont utilisées dans les réseaux (TCP/IP), en particulier pour Internet. C'est dans ce contexte favorable que E. F. Codd définit et développe l'approche relationnelle en base de données.

L'objectif principal est d'éloigner l'utilisateur des détails d'implémentation et de faciliter ainsi l'usage de l'informatique. Un autre but est de rendre « **génériques** » et réutilisables les développements informatiques, parfois devenus caducs en raison d'un changement de machine. Dans le domaine des bases de données, le développement de l'**architecture à trois niveaux** constitue une première étape importante. Les fonctionnalités des systèmes de bases de données sont séparées en trois niveaux : **niveau physique**, **niveau logique** et **niveau externe**.

2.2 MODÈLES

Les **modèles de données** correspondent à la manière de structurer l'information dans une base de données. Ils reposent sur les principes et les théories issus du domaine de la recherche en informatique et permettent de traduire la réalité de l'information vers une représentation utilisable en informatique.

Modèle hiérarchique et modèle réseau

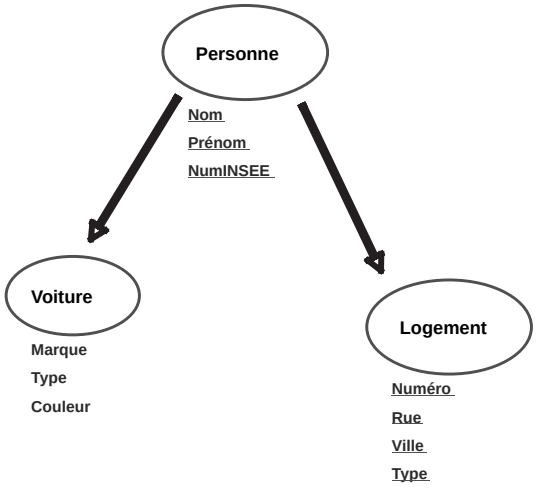
Le traitement de l'information à cette époque est encore très lié à l'organisation des fichiers sur une machine. Les modèles conceptuels de données sont eux aussi très proches du système de fichiers puisque l'on manipule des graphes ou des arbres. Les nœuds de ces structures constituent les informations et les liens entre ces données les arêtes. À ce moment, on n'est pas encore capable de séparer complètement le **niveau logique** du **niveau physique** d'un système de bases de données (voir figure 1.2).

Le modèle « hiérarchique » propose une classification arborescente des données à la manière d'une classification scientifique. Dans ce type de modèle, chaque enregistrement n'a qu'un seul possesseur ; par exemple, une commande n'a qu'un seul client. Cependant, notamment à cause de ce type de limitations, ce modèle ne peut pas traduire toutes les réalités de l'information dans les organisations.

Le modèle « réseau » est une extension du modèle précédent : il permet des liaisons transversales, utilise une structure de graphe et lève de nombreuses limitations du modèle hiérarchique. Dans les deux cas, les enregistrements sont reliés par des pointeurs : on stocke l'adresse de l'enregistrement auquel il est lié. Des SGBD de type hiérarchique ou réseau sont encore employés pour des raisons d'efficacité lorsque la structure des données s'y prête. On utilise à cet effet des SGBD de conception ancienne, comme IMS (Bull) pour le modèle réseau ou IDMS (Computer Associate).

Figure 1.2

Modèle hiérarchique.



Modèle relationnel

En 1970, E. F. Codd propose un nouveau modèle « relationnel » dans un article resté célèbre : « A Relational Model of Data for Large Shared Data Banks », *CACM 13*, n° 6, June 1970. Il cherche à créer un langage d'interrogation des bases de données plus proche du langage naturel. Dans cette optique, il fonde sa recherche sur des concepts mathématiques rigoureux, tels que la théorie des ensembles et la logique du premier ordre. Le modèle relationnel permet de modéliser les informations contenues dans les bases de données en utilisant des **relations**, c'est-à-dire des ensembles d'**attributs** (voir figure 1.3).

De l'idée de départ à la réalisation d'un produit utilisable, le laps de temps est souvent de l'ordre d'une décennie. La mise en œuvre des idées de Codd se fait chez IBM dans le cadre du projet de recherche System-R. Le premier produit commercial sera non pas le fait d'IBM, mais celui d'Honeywell en 1976. Il sera suivi d'un produit réellement abouti de chez Relationnel Software en 1980 : Oracle, qui a connu le succès que l'on sait. De son côté IBM en tirera un produit qui deviendra DB2.

Toujours dans le cadre du projet de recherche System-R, E. F. Codd met au point, en même temps que le modèle relationnel, un langage d'interrogation des données, SEQUEL, qui deviendra ensuite **SQL** (***Structured Query Language***). La normalisation du langage SQL dès 1986 par l'ANSI (institut de normalisation américaine), puis par l'ISO (organisation internationale de normalisation), a assuré pour une grande partie le succès du modèle relationnel auprès des entreprises. Fait rare dans le monde informatique, ce langage a été adopté par la quasi-totalité des éditeurs commerciaux qui participent activement à son évolution. SQL est devenu le standard de fait, même si aucun éditeur ne respecte à la lettre la norme. D'ailleurs, à partir de SQL 2, il existe une définition de quatre niveaux de compatibilité avec la norme officielle. La normalisation de ce langage garantit sa pérennité, même si son évolution s'en trouve ralentie. Les requêtes écrites pour un SGBD fonctionnent en général sans trop de modifications avec un autre SGBD, ce qui permet d'envisager des migrations moins douloureuses et de conserver une partie de l'investissement initial.

Figure 1.3

Modèle relationnel.

Ouvrage (Cote, Titre, Auteur,Editeur)

Cote	Titre	Auteur	Editeur
1	La programmation sous Unix	J. M. Rifflet	MC Graw Hill
2	Les bases de données	G. Gardarin	Eyrolles
3	Internet pour les nuls	C. Baroudi	First Interactive
4	Le langage C	C. Delannoy	Eyrolles
5	Petit Larousse illustré	P. Larousse	Larousse
6	Godel, Escher & Bach	D. Hofstadter	Basic Book Inc.

Modèle objet

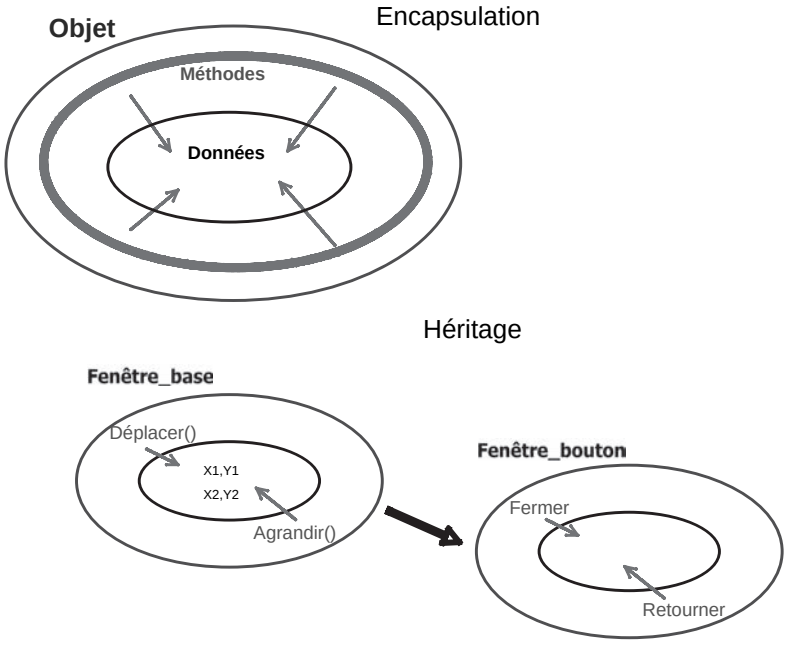
Dans le sillage du développement des langages orientés objet (C++, Java…) dans les années 1980, le **concept objet** a été adapté aux bases de données. Plusieurs raisons, en dehors des qualités reconnues de l'approche objet, ont conduit à définir une extension objet pour les bases de données (voir figure 1.4).

La première est que le modèle relationnel, dans sa simplicité, ne permet pas de modéliser facilement toutes les réalités. La deuxième est qu'un objet permet de représenter directement un élément du monde réel. Les structures d'éléments complexes se retrouvent souvent dispersées entre plusieurs tables dans l'approche relationnelle classique. De plus, le concept objet est mieux adapté pour modéliser des volumes de texte importants ou d'autres types de données multimédias (sons, images, vidéos…). Enfin, il est beaucoup

plus commode de manipuler directement des objets lorsque l'on développe avec un langage à objet (comme C++ ou Java). Les bases de données, on le rappelle, sont dorénavant des briques constitutives des applications. Les **bases de données « orientées objet »** apportent ainsi aux applications développées en langage objet la **persistance** des objets manipulés : ces derniers peuvent ainsi directement être réutilisés par l'application d'origine ou par d'autres sans redéfinition. Ces concepts ont été intégrés à partir de la version 2 de la norme SQL.

Les produits commerciaux adaptés à ces concepts n'ont pas connu une diffusion suffisamment importante. Le monde des bases de données évolue assez lentement : la migration d'un système d'information vers l'objet représente pour une organisation un investissement considérable qui n'est pas toujours justifié. La robustesse et la popularité de l'approche relationnelle, qui a mis presque vingt ans à s'imposer, a également freiné le développement de l'approche objet pure dans les bases de données. Les données modélisées sous forme d'objets sont aussi plus complexes à représenter du point de vue du SGBD et l'on rencontre encore très souvent des problèmes de performance.

Figure 1.4
Modèle objet.



Modèle relationnel-objet

Une demande d'évolution du strict modèle relationnel existe toutefois. En effet, la gestion des données autres que du texte et des nombres – comme des images, du son et des vidéos – implique l'évolution du modèle relationnel. De même, les champs dits « multivalués », disposant de plusieurs valeurs telles qu'une liste de prénoms ou des coordonnées géographiques, ne peuvent pas être modélisés efficacement en utilisant ce type d'approche. L'idée est alors d'intégrer de l'objet au modèle relationnel existant plutôt que d'utiliser l'approche objet pure. Il convient de remarquer que ce type d'évolution a déjà été développé dans le cadre des langages de programmation. Le langage C++ est l'évolution intégrant l'approche objet du langage C et non pas un langage à objet pur comme peut l'être Smalltalk.

Cette extension, adoptée par la plupart des SGBD, se nomme « **relationnel-objet** » et permet aux concepteurs des bases de données de disposer de types évolués « abstraits » plus simples à concevoir et surtout plus commodes à faire évoluer. Elle offre en outre la possi-

bilité de modéliser plus facilement la complexité des organisations (voir figure 1.5). Dans cette optique, la norme SQL a logiquement été adaptée. Dans sa version 3, elle prend en compte l'extension objet. Les types de données sont étendus et les opérations d'encapsulation et d'héritage, typiques de l'approche objet, sont supportées. Cette solution a l'avantage d'offrir un bon niveau de compatibilité avec l'approche précédente très répandue et d'effectuer ainsi une migration plus aisée.

Figure 1.5

Modèle relationnel-objet.

Cote	Titre	Auteur	Editeur
1	Le vide	Nom: Durand, Prénom: Dal Nom: Atamp, Prénom: Charles	Les éditions du temps qui ne passe pas
2	La vie sans mode d'emploi	Nom: Brassé, Prénom: Alexis Nom: Duporche, Prénom: Jean-Marie Nom: Château, Prénom: Romain Nom: Paclaire, Prénom: Anne-Isabelle	Romazava

2.3 ÉVOLUTION DE L'UTILISATION DES BASES DE DONNÉES

Cette section présente une description de nouvelles manières de stocker ou d'utiliser les bases de données. La différence par rapport à la section précédente est que l'on ne remet pas en cause le modèle utilisé pour décrire les données sauf, dans une certaine mesure, pour le cas de XML.

Base de données réparties

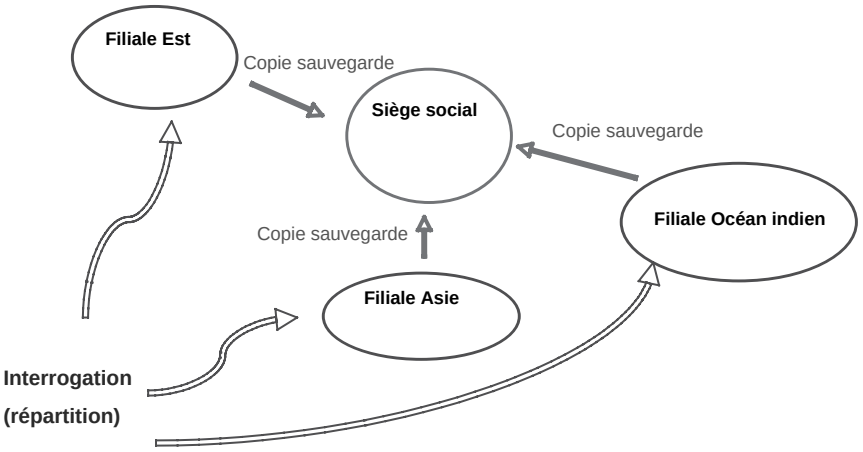
Le déploiement des réseaux ainsi que l'augmentation de leur débit ces dernières années ont conduit à répartir les données sur plusieurs sites géographiques, ce qui facilite la politique de décentralisation des organisations. Ce type d'architecture masque la répartition de l'information tout en garantissant une gestion transparente aux utilisateurs, comme s'ils disposaient d'une seule base de données (voir figure 1.6). Les bases de données réparties assurent ainsi une plus **grande fiabilité**, de **meilleures performances** et facilitent l'**évolution du système d'information**.

- **La fiabilité et la sécurité.** On effectue une copie de sécurité des données sur un site distant à intervalles réguliers pour éviter le désastre de la perte de données due à un incendie par exemple.
- **La disponibilité.** On procède à des réplications quasi permanentes des données dans le but de « rapprocher » les utilisateurs des données d'un point de vue de la topologie du réseau. On améliore également le temps de réponse en répartissant la charge entre les serveurs. Cette distribution est gérée de manière intelligente par les systèmes informatiques, ce qui permet de répartir l'information efficacement sur les différents sites, en fonction des accès utilisateurs. Ces principes sont notamment utilisés par les moteurs de recherche.
- **Les données sont réparties sur des sites séparés.** Le dispositif permet de masquer cet aspect aux utilisateurs et de fonctionner comme si un seul serveur était présent sur un seul site. L'évolution du système est rendue totalement **transparente** pour les utilisateurs : en cas notamment de changement de machine à la suite d'une panne, de

modification de localisation, d'augmentation de la taille de la base, d'ajouts d'ordinateurs sur le réseau afin d'augmenter la capacité de stockage de la base de données.

Figure 1.6

Base de données réparties.



Ces technologies possèdent néanmoins des inconvénients. La sécurité sur les réseaux informatiques nécessite beaucoup plus de travail que dans le cas d'un système non réparti. Les techniques de sécurité à mettre en œuvre sont plus complexes et plus coûteuses.

Extraction d'informations non explicitement stockées dans une base de données

Il existe deux manières de « créer » de la nouvelle information à partir de l'information stockée explicitement dans une base de données. Soit on est capable d'énoncer des règles précises de constitution de l'information à partir du sens même des données, soit on utilise des méthodes d'analyse afin de trouver des corrélations sur un volume de données important, ce qui permet ensuite d'en déduire des règles.

La première possibilité a été formalisée et mise en œuvre sous le nom de **base de données déductives**, dans le but d'utiliser des méthodes semblables à celles pratiquées pour la déduction en intelligence artificielle. On définit un ensemble de règles et on les applique aux données de la base, à l'aide d'un programme que l'on appelle un « moteur d'inférence ». Les SGBD qualifiés de déductifs utilisent à cet effet un dérivé du langage Prolog : **Datalog**.

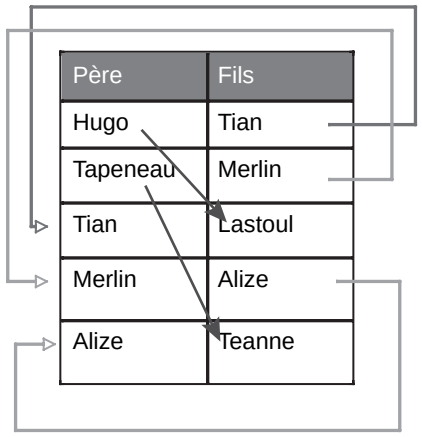
Les relations de parenté entre individus sont une bonne illustration de l'utilisation des bases de données déductives. Intuitivement, on peut appréhender le fonctionnement général de ces SGBD déductifs en considérant l'exemple suivant. On suppose que l'on a une base de données qui représente les relations « père-fils » (voir figure 1.7). Une règle très simple permet de définir un lien de parenté « ancêtre » :

Si père-fils(X,Y) et père-fils(Y,Z), alors ancêtre(X,Z).

La seconde possibilité d'obtenir l'information nouvelle à partir de l'information existante relève des méthodes dites de « **fouilles de données** » (ou **data mining**). Ce type d'exploitation des bases de données a connu un grand succès ces dernières années, par l'analyse de grands volumes de données afin d'identifier des corrélations entre des valeurs de champs. Par exemple, « les personnes moustachues de plus de quarante ans habitant une maison individuelle lisent plutôt des revues de psychologie ». Les techniques d'analyses employées sont une alchimie concoctée à partir de statistiques, de réseaux de neurones, de classifications et autres techniques employées en intelligence artificielle. Une fois ces « règles » établies, on peut les utiliser au sein des bases de données déductives décrites précédemment.

Figure 1.7

Base de données déductives père-fils.



L'information décisionnelle ainsi extraite a de nombreux débouchés : du ciblage marketing à la médecine en passant par la prévision financière. Les méthodes s'affinent et deviennent réellement efficaces, ce qui a cependant donné lieu à quelques escroqueries, les entreprises spécialisées dans le domaine refusant bien sûr de donner les spécifications de leurs méthodes d'analyses, qui relevaient parfois de la divination, au motif de ne pas perdre leur avantage concurrentiel.

Entrepôts de données (datawarehouse)

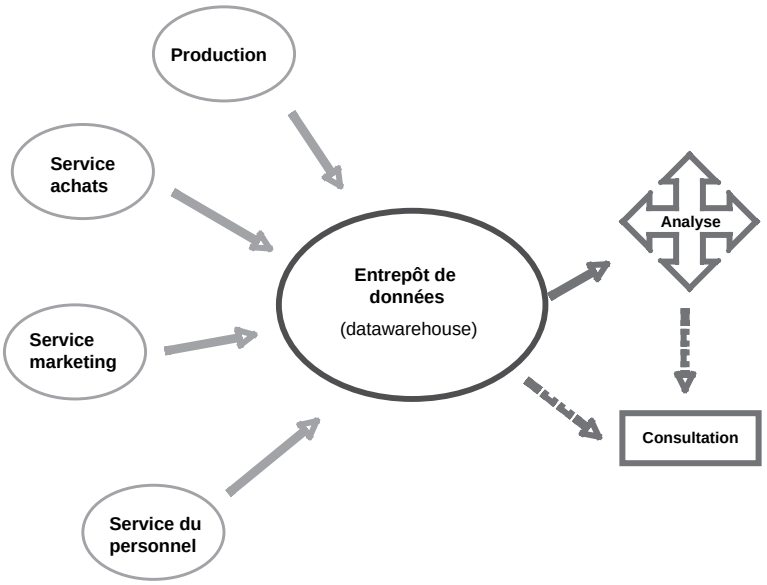
Les entrepôts de données sont des bases de données « récapitulatives », constituées à partir de différentes sources de l'entreprise (comptabilité, ventes, achats, service du personnel…) pour disposer d'un accès homogène à l'ensemble des données. Les données dispersées ne peuvent pas être exploitées directement en tant qu'information décisionnelle. En effet, il n'est pas possible d'analyser efficacement dans le temps des indicateurs de gestion par métiers ou par individus. Les données sont alors rapatriées et stockées dans un **entrepôt de données** ou **datawarehouse** (voir figure 1.8).

Si l'information est disponible, on stocke les différentes valeurs d'une donnée, ce qui permet de conserver un historique dans le temps de certaines données : elles sont dites **historisées**. Ce dernier aspect est désigné parfois aussi comme une **base de données temporelle**, dans la mesure où le SGBD procure des outils d'analyse adaptés. Il est alors possible d'effectuer des analyses décisionnelles qui combinent tous les paramètres de l'entreprise : c'est l'**analyse décisionnelle multidimensionnelle**. Les techniques précédentes de fouille de données et de déduction peuvent être utilisées. Elles sont combinées avec d'autres outils d'analyse de l'évolution d'une donnée à partir de son historique pour en extrapoler des tendances et ainsi faire des prévisions.

L'idée des entrepôts de données est plus qu'intéressante, mais il est évident que sa mise en œuvre est délicate tant d'un point de vue du stockage de l'information (volume de données, hétérogénéité des données à traiter…) que de l'analyse des données (comment analyser, quelle information stocker…). En outre, il s'agit le plus souvent du poste budgétaire le plus onéreux dans un projet d'informatique décisionnelle.

Figure 1.8

Entrepôts de données.



XML

Le *World Wide Web* est la mise en œuvre du concept d'hypertexte par l'utilisation de la structure de communication fournie par Internet. Les fichiers dispersés sur différentes machines du réseau Internet peuvent ainsi être associés. Son succès a provoqué un glissement de l'idée originale, qui consistait à relier simplement des textes entre eux, vers la notion d'interface universelle. On ne transmet plus seulement le contenu d'un fichier statique, mais également le résultat de l'exécution d'un programme : le contenu devient donc dynamique. Par extension, on imagine aisément que le mécanisme du Web peut également transmettre le résultat de l'interrogation d'une base de données.

Le concepteur du Web, T. B. Lee, s'est appuyé pour sa mise en œuvre, outre la structure technique existante d'Internet, sur un langage de description de documents utilisant des balises : le SGML (*Standard Generalized Markup Language*). Le découpage du document par les balises est décrit dans un document associé que chacun peut créer en fonction des ses besoins : la DTD (*Data Type Definition*). Cette dernière, formulée dans un langage normalisé, permettra à des programmes spécialisés, les « parsers », de vérifier si le document est conforme à la structure attendue et d'en extraire facilement le contenu. Le langage SGML est assez complexe à manipuler, car il est très général. Pour des besoins spécifiques, on utilise des versions simplifiées, telles que HTML ou XML.

T. B. Lee a donc développé un langage de présentation, HTML (*Hyper Text Markup Language*), basé sur les notions développées par SGML. Ce langage permet essentiellement de spécifier des critères de présentation (gras, souligné, couleur…) et, bien sûr, de décrire les liens entre les fichiers. Il ne comprend aucun élément de structuration des données du texte contenu dans la page HTML. Les moteurs de recherche parcourent le Web et indexent le contenu des pages par rapport à des listes de mots clés combinés à d'autres méthodes (beaucoup) plus sophistiquées. Cependant, ils ne peuvent différencier dans le texte le titre d'un résumé ou d'une légende associée à une image. L'efficacité et la pertinence de l'indexation en sont diminuées d'autant.

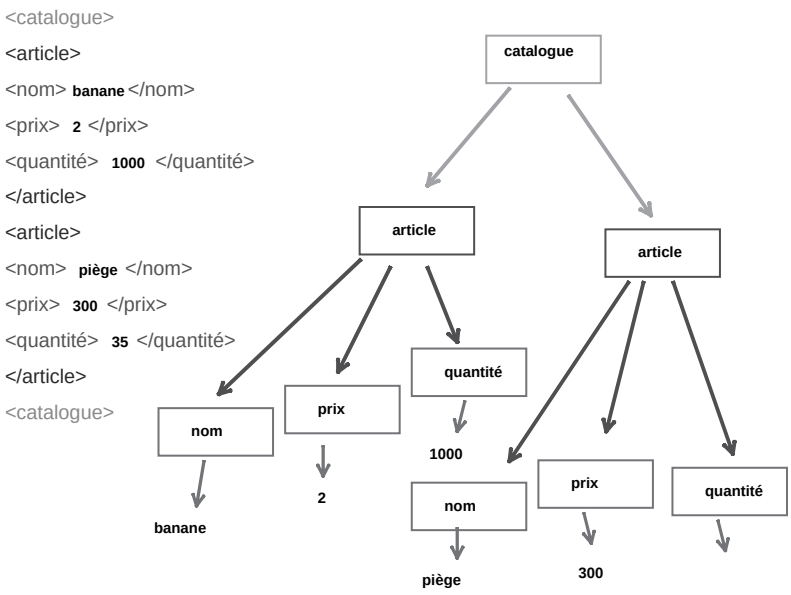
Pour remédier à cela, le W3C (*World Wide Web Consortium*) a défini un langage, qui est également une simplification de SGML, permettant de décrire la structure interne d'un document : XML (*eXtended Markup Language*). La structure d'un document XML est représentée sous la forme d'un arbre tout comme celle d'un document HTML. La description de

cette structure arborescente se trouve dans une DTD ou plus récemment dans un schéma XML. Le schéma est plus souple et permet d'employer les mêmes outils de traitement que pour les fichiers XML. Le langage HTML possède évidemment lui aussi une DTD, mais, à la différence de XML, elle est normalisée par le W3C et ne peut être modifiée et adaptée pour ses besoins propres (voir figure 1.9).

L'objectif à terme est que les fichiers du *World Wide Web* soient désormais décrits en utilisant XML à la place de HTML. La présentation des données repose alors sur les « feuilles de styles » (telles que *eXtended Stylesheet Language*) pour générer les formats de présentation classiques (HTML, PDF, PostScript…) à partir du format XML. De cette manière, les moteurs de recherche pourront extraire directement par exemple le résumé ou le titre d'un paragraphe d'un fichier XML. L'indexation peut ainsi être plus précise ; un mot figurant dans un titre étant plus important que le même mot dans une note de bas de page.

Figure 1.9

Structure arborescente XML.



Quel est le rapport avec les bases de données ? Comme on l'a vu précédemment, une page Web peut être le résultat d'une requête provenant d'un SGBD ; c'est même devenu le moyen le plus courant d'interroger un SGBD. Dans cette optique, si le SGBD est capable de générer directement du XML, cela facilite le processus. C'est d'autant plus vrai que le passage du modèle relationnel à un modèle arborescent de type XML est parfois complexe et qu'il est bien agréable que le SGBD sache le faire.

Le langage de description XML, par sa versatilité, s'impose comme un format d'échange universel. C'est évident pour des fichiers générés par un traitement de texte : on sépare ainsi l'aspect structurel de l'aspect présentation. On se donne également la possibilité d'ouvrir le document indépendamment du logiciel utilisé, ce qui garantit sa pérennité. De la même manière, XML est utilisé comme format d'échange entre SGBD et d'un SGBD vers d'autres logiciels (tableurs, traitements de texte…).

Le langage XML est adopté petit à petit par la plupart des éditeurs et il est amené à jouer un rôle croissant dans les échanges de données. De plus en plus de SGBD sont capables de produire des résultats de requête en XML, d'importer du XML et acceptent même de gérer les données directement dans ce format. Cette dernière possibilité implique que les SGBD supportent des données moins bien structurées : cette capacité constitue l'une des évolutions futures des SGBD.

Contenu multimédia

Le multimédia s'est fortement développé depuis la fin des années 1990. La demande étant très forte dans ce domaine, les éditeurs de SGBD se doivent d'intégrer de l'information multimédia (image, son, vidéo…) dans les bases de données. Les **bases de données multimédias** posent de nouveaux problèmes, en particulier pour effectuer des recherches sur les contenus multimédias, ce qui est par nature difficile.

Une solution est d'effectuer une indexation préliminaire manuelle à l'aide de mots clés qui permettent d'opérer par la suite des interrogations, mais cela semble illusoire de réaliser ce traitement pour des volumes importants de documents multimédias. Dans le cas contraire, il existe des méthodes de recherche sur des fichiers de type image par rapport à des schémas prédéfinis. Cette possibilité reste pour l'instant plutôt du domaine de la recherche, même si l'on est déjà (malheureusement) capable d'identifier des visages par rapport à un modèle dans certaines conditions.Dans le même ordre d'idée, il est déjà possible d'utiliser des techniques pour évaluer le style de musique (classique, jazz, pop…) d'un fichier en analysant son contenu. On est cependant assez loin de pouvoir identifier un genre de film en se basant sur l'analyse des images.

Les bases de données multimédias constituent un sujet de recherche très prometteur et très actif. Ces problématiques relèvent pour l'instant plutôt des préoccupations des sociétés développant les moteurs de recherche que des bases de données au sens strict. Cependant, le groupe de normalisation ISO/IEC prépare l'évolution pour le multimédia de la norme SQL-MM, qui est une évolution pour le multimédia de la norme SQL.

3 Systèmes de gestion de bases de données

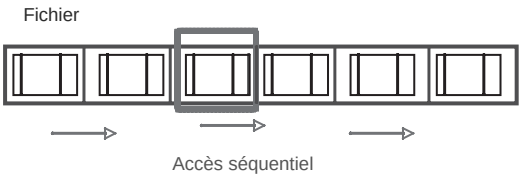
On a beaucoup parlé dans les sections précédentes des SGBD. Un SGBD est un logiciel complexe qui permet de gérer et d'utiliser les données que l'on stocke en utilisant les modèles cités précédemment. La première partie de la section permet de comprendre le mécanisme d'abstraction à partir duquel se fait le passage du simple fichier informatique à la gestion de l'information qui se fonde sur l'utilisation d'un SGBD. Une seconde partie détaille le modèle en couches des SGBD ainsi que leurs fonctionnalités de base.

3.1 Fichiers informatiques

Un fichier peut être vu (historiquement) comme un morceau de bande magnétique. Les mécanismes de gestion de fichiers des langages de programmation, comme le langage C, fonctionnent encore avec cette métaphore. Lorsqu'on utilise un fichier pour stocker de l'information, il est nécessaire de prévoir un découpage de celui-ci par enregistrements, souvent de taille fixe. Pour passer d'un enregistrement à l'autre, il suffit alors d'avancer la « tête » de lecture de la taille d'un enregistrement (voir figure 1.10). Les données sont stockées dans l'enregistrement par un découpage interne suivant la taille de chaque donnée. On utilise généralement des structures de données (par exemple, en langage C) pour récupérer directement chaque valeur de champ dans une variable. Dans une base de données, on recherche les données par leur contenu. Pour retrouver l'une d'entre elles, il faut donc parcourir tous les enregistrements un à un (**recherche séquentielle**), et en examiner le contenu.

Figure 1.10

Fichier informatique.



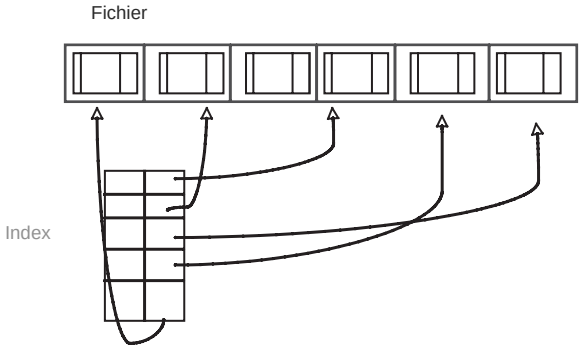
Dans le cas d'un accès séquentiel, la recherche d'un enregistrement en position n nécessite d'accéder aux n–1 enregistrements qui le précèdent. Si l'on recherche l'article de référence DR-NetCard10.102, contenu dans le fichier Article, il sera nécessaire de parcourir tous les enregistrements, depuis le début du fichier jusqu'à l'article recherché. Pour retrouver une information, il faut donc parcourir tous les enregistrements un à un et en examiner le contenu.

Une alternative au parcours séquentiel est de construire des tables descriptives afin d'accélérer l'accès aux données. Une première table permet l'**accès direct** à un enregistrement par une « clé » associée à l'adresse (pointeur) de l'enregistrement. On rappelle que c'est ce mécanisme de « pointeurs » sur des enregistrements qui est modélisé dans les modèles hiérarchiques ou réseaux pour faire le lien entre des enregistrements. On constitue ainsi le graphe qui permet de « naviguer » dans l'ensemble des enregistrements.

Une seconde table contient l'ordre relatif des enregistrements ordonnés suivant les valeurs d'un champ : on appelle cette table un **index** (voir figure 1.11). Cette seconde table permet d'employer des méthodes de recherche par le contenu du champ indexé beaucoup plus efficaces qu'une recherche séquentielle. La recherche dichotomique bien connue est l'une d'entre elles. Une fois l'enregistrement identifié, on y accède directement grâce à la première table. Les techniques de constitution des index constituent un sujet à part entière ainsi que les algorithmes de recherche qui leur sont associés.

Figure 1.11

Fichier et index.



On constate que la simple recherche d'informations recourt déjà à des algorithmes assez sophistiqués et nécessite la construction et le maintien de structures annexes pour être efficace. De même, la destruction d'enregistrements et l'évolution de la structure de la base sont également des opérations lourdes à mettre en œuvre ; elles requièrent souvent la recopie complète des informations dans un nouveau fichier. Cette section nous permet de comprendre pourquoi on utilise préférentiellement des SGBD pour gérer les données. Toutes ces fonctionnalités et bien d'autres que nous allons détailler sont intégrées dans le logiciel.

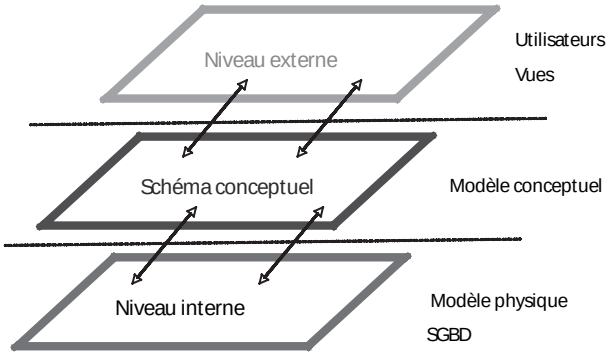
3.2 FONCTIONNALITÉS D'UN SGBD

De même que l'ISO a déterminé un modèle théorique en sept couches pour distinguer les applications réseaux et leurs interactions, il existe désormais un modèle théorique en trois couches (trois niveaux d'abstraction) afin de concevoir et d'organiser les fonctionnalités des SGBD. Ce modèle est élaboré par la commission SPARC de l'ANSI : c'est l'**architecture ANSI/SPARC** (voir figure 1.12). Cette dernière, qui date de 1975, s'inscrit dans les concepts et théories de la première génération des bases de données, dont l'objectif est d'avoir une **indépendance entre les données et les traitements** :

- Niveau **interne** ou **physique**. C'est le niveau le plus « bas ». On décrit les structures de stockage de l'information, ce qui le rend très dépendant du SGBD employé. Il se fonde sur un **modèle de données physique**.
- Niveau **conceptuel**. Il correspond à l'implémentation du **schéma conceptuel de la base de données**, que l'on réalise lors de la phase d'analyse (voir *Modèle Conceptuel des Données*, *Modèle Logique des Données*). Il est utilisé pour décrire les éléments constitutifs de la base de données et les contraintes qui leur sont associées. Il s'agit d'une certaine façon de la « documentation » de la base de données.
- Niveau **externe**. Le niveau externe sert à décrire les **vues** des utilisateurs, c'est-à-dire le **schéma de visualisation des données** qui est différent pour chaque catégorie d'utilisateurs. Un schéma externe permet de masquer la complexité de la base de données complète en fonction des droits ou des besoins des utilisateurs. Cela facilite la lecture et la sécurité de l'information.

Figure 1.12

Niveaux ANSI/ SPARC.



Il s'agit comme pour le modèle réseau de l'ISO d'un cadre de réflexion ; les SGBD ne respectent pas à la lettre le découpage proposé. Ils se doivent cependant de posséder les principales caractéristiques qui découlent de ce modèle en couches :

- **Indépendance physique des données.** Masquer la représentation interne des données ainsi que les méthodes système d'accès aux utilisateurs.
- **Indépendance logique des données.** Permettre la modification du schéma conceptuel des données sans remettre en cause les mécanismes de stockage et de manipulation internes des données.
- **Intégrité des données.** Faire en sorte que l'information résultant des liens entre les données soit cohérente.

Il peut apporter également des fonctionnalités supplémentaires utilisées dans le cadre de bases de données réparties décrites précédemment :

- **Réplication des données.** Copie automatisée de sauvegarde.

- **Virtualisation des données.** Masquage de la distribution géographique des données.
- **Haute disponibilité des données.** Duplication de la base de données sur différents sites pour diminuer la distance client/serveur et la charge des serveurs.

Le but principal de l'utilisation d'un SGBD est de masquer la représentation physique des données et les méthodes d'accès que l'on vient de voir précédemment. Cependant les mécanismes de création, d'indexation et de recherche sous-jacents sont globalement les mêmes. Évidemment, pour des questions d'efficacité, les SGBD utilisent leur propre gestion de fichiers et parfois même contournent le système de fichiers fourni avec le système d'exploitation de la machine.

Un SGBD doit permettre également la manipulation de la structure de la base de données, comme l'ajout et la modification de champs, de manière transparente. Il conserve à cet effet une description de la structure de la base de données que l'on appelle le « dictionnaire de données ». Pour réaliser ces opérations avec l'indépendance souhaitée par rapport à la représentation, le SGBD offre deux langages de haut niveau :

- un *Langage de Description de Données* (LDD) qui permet d'agir sur la structure de la base de données (ajout, suppression et modification des tables) ;
- un *Langage de Manipulation de Données* (LMD) qui permet d'interroger et de mettre à jour le contenu de la base de données.

Ces langages sont de type « non procédural », c'est-à-dire que l'on s'intéresse à l'effet de l'opération (le quoi) et non pas à la manière dont elle est réalisée (le comment). On a pu se rendre compte dans la section précédente du niveau de complexité de certaines opérations qui, grâce à ces langages, sont énoncées simplement. Par exemple, la modification de la taille d'un champ peut être énoncée en une seule instruction avec le LDD. Il est courant que les SGBD modernes implémentent ces langages de manipulation à l'aide d'objets graphiques.

Le SGBD doit également assurer la protection des données en cas de problèmes. Ceux-ci peuvent être la conséquence d'une manipulation malheureuse, mais également d'une panne du système qui survient par exemple à la suite d'une coupure de courant. Dans tous les cas, le SGBD doit permettre de restaurer les données. Ces opérations sont généralement réalisées en utilisant des « journaux » qui enregistrent au fur et à mesure les opérations faites sur la base : c'est le **mécanisme de la journalisation**. Ce journal est utilisé pour refaire, ou défaire le cas échéant, ces opérations.

En ce qui concerne les opérations de modification effectuées sur la base de données, que l'on appelle des **transactions**, des propriétés de mesure de la qualité de ces transactions sont proposées sous le terme ACID :

- **Atomicité.** Une transaction est « atomique » ; elle est exécutée entièrement ou abandonnée.
- **Cohérence.** La transaction doit se faire d'un état cohérent de la base vers un autre état cohérent.
- **Isolement.** Des transactions simultanées ne doivent pas interférer entre elles.
- **Durabilité.** La transaction a des effets permanents même en cas de panne.

À noter que tous les SGBD ne réalisent pas cette propriété ACID pour les transactions.

Les machines sont connectées au réseau ou sont simplement multi-utilisateurs : le SGBD doit permettre de donner l'accès aux bases de données à plusieurs utilisateurs concurremment. **L'accès concurrentiel** implique des opérations algorithmiques complexes à réaliser, puisqu'il faut par exemple empêcher la modification d'une valeur par un utilisateur alors qu'elle est en lecture par un autre. Cela nécessite la gestion d'une structure de description

des utilisateurs comprenant les droits qui leur sont associés pour chaque élément (lecture, modification…) : les **droits d'accès aux données**. Les mécanismes sont les mêmes que ceux qui sont mis en œuvre dans les systèmes d'exploitation multi-utilisateurs.

4 Étapes de la conception des bases de données

On peut décomposer le processus de conception d'une base de données en plusieurs étapes :

- l'analyse du système du monde réel à modéliser ;
- la mise en forme du modèle pour l'intégrer dans un SGBD ;
- la création effective dans le SGBD des structures et leur remplissage (voir figure 1.13).

4.1 Analyse du monde réel

La première étape de la démarche de modélisation des données consiste à effectuer l'analyse de la situation du monde réel à considérer. Cette action s'apparente au travail effectué par une entreprise de consulting. C'est une approche « humaine » qui se fonde en partie sur des entretiens avec les personnels concernés et ressemble plutôt à une analyse du discours et de l'organisation de l'entreprise. C'est lors de cette phase d'analyse que l'on détermine les objectifs du système d'information à concevoir et que l'on identifie tous les éléments à prendre en compte dans le système ; ce sont les champs qui contiendront les données. Un ensemble de champs peut constituer un objet du monde réel. Par exemple les champs « nom », « prénom » et « adresse » que l'on regroupe constituent une « personne ».

Enfin, il faut identifier les liens à modéliser entre ces objets ainsi que les éléments caractéristiques de ces liens. Par exemple une personne achète une voiture à 10 000 euros. Ici les deux objets liés sont « personne » et « voiture », et le prix est le composant du lien. Cette étape est délicate et fondamentale, car elle conditionne l'aspect représentatif et la qualité du modèle du monde réel considéré. Lors de cette phase, il convient également d'exprimer les règles qui définissent le domaine de validité du contenu des champs. Par exemple, le prix d'une voiture ne peut pas être inférieur à 500 euros ou supérieur à 150 000 euros.

Cette modélisation du réel permet de proposer un schéma conceptuel qui servira à la description générale du système d'information. La notion de sens des données et surtout des liens entre les entités ne sera réellement exprimée que dans ce schéma qui est plus proche du monde réel. Ce schéma est souvent réalisé à l'aide de la symbolique du modèle « entité-association » ou, plus couramment aujourd'hui, exprimé avec le langage UML (*Unified Modeling Language*). Il existe différentes méthodes intégrant les concepts présentés ci-dessus. L'objectif est de guider le travail d'analyse et d'aider à la réalisation d'un modèle de données le plus juste possible. Parmi celles-ci, la méthode Merise a connu un certain succès dans le domaine en France.

4.2 Passage au SGBD

La représentation précédente doit être transformée pour la rendre acceptable par le SGBD, qu'il soit relationnel, objet ou relationnel-objet. Souvent, cette étape modifie considérablement les objets du monde réel ainsi que les liens définis dans le schéma précédent. C'est lors de cette phase que l'on vérifie la qualité de la base de données en utilisant les critères

vus précédemment, comme l'élimination de la redondance. Le modèle relationnel procure à cette fin des outils capables de tester la cohérence du système et de le modifier le cas échéant : ce sont les « formes normales », qui seront vues au chapitre 3.

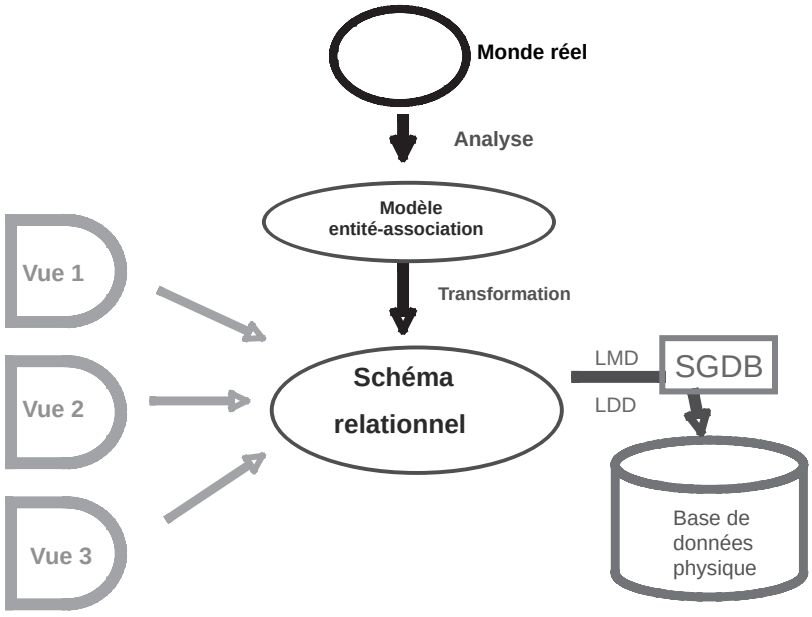
Il est possible de constater des incohérences à ce niveau de l'analyse, ce qui implique de modifier le modèle conceptuel de données développé à l'étape précédente. On obtient un schéma des données qui fournira aux utilisateurs les informations nécessaires pour effectuer leurs requêtes, par exemple la description des noms de tables, de champs et leurs types. Par contre, on perd à ce niveau l'information du « sens » des données et du lien entre elles. Ce schéma n'est guère utilisable en pratique sans le précédent. En effet, comment savoir que les personnes achètent des voitures et non pas le vendeur si l'on ne dispose pas de l'information de liaison entre les objets du monde réel ? C'est également lors de cette phase que l'on définit les « vues » du système d'information qui sont adaptées à chaque catégorie d'utilisateurs.

4.3 Création et utilisation de la base de données

Une fois le schéma précédent défini, on utilise le SGBD pour passer à la création des tables qui constituent la base de données. Puis, on insère évidemment les valeurs dans les tables. Le cas échéant, on crée les vues définies à l'étape précédente et par là même les utilisateurs concernés. Le système est alors opérationnel. Toute cette étape se fait forcément en utilisant le SGBD, alors que les précédentes étaient plus théoriques. La création des tables et l'utilisation de la base de données nécessiteront le langage SQL. Cependant, il existe de nos jours de nombreux outils graphiques dans les SGBD qui masquent l'utilisation du SQL.

Figure 1.13

Étapes de la conception d'une base de données.



5 « Métiers » des bases de données

Comme on peut le constater lorsque l'on considère les différentes étapes de la conception d'une base de données, des acteurs aux compétences très diverses interviennent dans ce processus.

5.1 Consultants/analystes

Ils prennent en charge la première étape qui consiste en l'analyse des activités et des flux d'information mis en jeu dans le monde réel à modéliser. Le profil de ces acteurs n'est pas toujours purement technique, puisque cette phase nécessite parfois beaucoup de dialogues et de psychologie pour parvenir à faire exprimer leurs besoins réels par les futurs utilisateurs. La gageure est de parvenir à faire exprimer correctement les besoins d'informatisation par les utilisateurs du système d'information, afin de proposer un modèle conceptuel de données le plus juste possible.

5.2 Concepteurs de la base

Ce sont les personnes qui s'occupent de traduire le modèle précédent en un modèle logique exploitable par le SGBD. Le concepteur est un spécialiste des bases de données qui prépare les tables, les vues, les schémas d'accès. C'est lui qui renseigne les utilisateurs et programmeurs pour la définition des requêtes. Il n'a pas, en principe, à être spécialisé sur un SGBD particulier, mais en pratique les éléments qu'il manipule sont liés au SGBD qui sera employé. C'est ordinairement lui qui crée les éléments nécessaires à la base de données (tables, vues…) en collaboration avec l'administrateur de la base. C'est parfois la même personne qui est en charge de la partie analyse et de la conception, ce qui peut induire une vision un peu trop orientée techniquement – comme celle d'un programmeur qui écrirait le cahier des charges d'une application. Par contre, le concepteur peut aussi être administrateur du SGBD, ce qui ne pose pas de problèmes particuliers d'approche.

5.3 Administrateurs de base de données (DBA, DataBase Administrator)

L'administrateur a la responsabilité du fonctionnement général du SGBD. Il crée les ressources (bases, comptes) à la demande. Il donne les droits d'accès et gère les personnes qui accèdent au système. Il vérifie que les ressources sont suffisantes (taille du disque, puissance de la machine), effectue les sauvegardes, vérifie les failles de sécurité. Pour ces opérations, il est en relation avec l'administrateur système et réseau de la structure. Ce métier est extrêmement lié au SGBD employé. Il n'y a pas vraiment de normalisation pour les opérations d'administration des SGBD qui sont spécifiques au SGBD et à la version utilisés.

5.4 Utilisateurs standard et programmeurs d'applications

Ce sont eux qui utilisent le système d'information. Ils y ont accès grâce aux vues définies par le concepteur de la base. Ils utilisent les schémas déterminés aux deux premières étapes de la conception. Ils n'ont pas besoin théoriquement d'être spécialisés sur le SGBD employé. En pratique il est préférable, surtout pour les développeurs d'applications, d'avoir de bonnes connaissances du fonctionnement du SGBD. Par exemple, pour optimiser les performances, la manière d'écrire les requêtes peut être assez différente suivant le SGBD employé.

6 Plan de l'ouvrage

Le plan de l'ouvrage est déterminé par les différentes étapes de la conception d'une BD.

Le **deuxième chapitre** traite de l'étape d'analyse du monde réel pour en concevoir un modèle descriptif. La modélisation est effectuée classiquement par le modèle « entité-association ». On aborde également la représentation du modèle avec UML.

Le **troisième chapitre** est essentiellement consacré au modèle relationnel et aux opérations qui lui sont associées. On s'intéresse ensuite aux méthodes grâce auxquelles il est possible de passer du modèle précédent à un ensemble de tables du modèle relationnel. Puis, on continue par l'étape de normalisation qui permet de mettre en évidence les incohérences du système d'information ainsi créé et de les rectifier. On présente dans ce chapitre les trois premières formes normales et celle de Boyce-Codd. Une autre approche de la création d'une base de données à partir de la « relation universelle » est abordée à la fin du chapitre.

Le **quatrième chapitre** présente le langage SQL. Après avoir exposé la syntaxe générale du langage, il s'attache à la réalisation en SQL des opérations relationnelles vues précédemment. On aborde ensuite la correspondance entre les questions classiques des bases de données en langage parlé et leur représentation avec SQL. La dernière partie du chapitre traite de la partie « description de données » de SQL, qui permet de créer et gérer les tables dans un SGBD.

Le **cinquième chapitre** constitue un exemple complet de réalisation d'une base de données par la pratique. Cet exemple part de l'énoncé du monde réel en langage parlé jusqu'à la réalisation du système d'information, puis bien sûr son utilisation. Ce chapitre reprend les notions vues aux chapitres précédents de manière pratique.

Le **sixième chapitre** aborde les mécanismes généraux de préservation des données associés aux SGBD. On aborde la sécurité des accès et les sauvegardes, mais aussi les outils qui participent à la sécurisation des données comme les transactions ou les « triggers ».

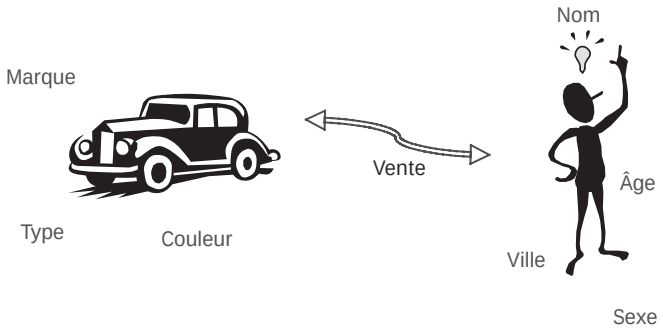
7 Présentation de la BD exemple

Une base de données extrêmement simplifiée est utilisée régulièrement tout au long de l'ouvrage afin d'illustrer les concepts développés au fil de l'ouvrage**.** D'autres bases de données plus complexes sont présentées au fur et à mesure des explications (voir figure 1.14).

Ce système d'information qui constitue l'étude de cas basique modélise l'activité de vente de voitures d'occasion. Dans ce système, deux entités du monde réel sont identifiées : les personnes et les voitures. Une voiture est caractérisée par sa marque, son type, sa couleur. Une personne est caractérisée par son nom, son âge, sa ville, son sexe. L'action modélisée est la vente qui est caractérisée par le prix de la vente et sa date. Une personne peut acheter une plusieurs voitures. Une voiture peut être vendue une seule fois ou jamais.

Figure 1.14

Base de données exemple.



Le modèle « entité-association » et la représentation UML correspondant à cette description seront créés au chapitre 2, « Analyse du monde réel ». Le modèle relationnel sera créé au chapitre 3, « Approche relationnelle ». La base de données résultante sera utilisée pour les exemples du chapitre 4, « SQL ».

Résumé

Une base de données désigne l'ensemble des données stockées. Pour manipuler ces données, on utilise un SGBD (*Système de Gestion de Bases de Données*) qui est un logiciel complexe. L'ensemble composé par les données et le SGBD constitue un système d'information. La conception d'une base de données – de la modélisation du monde réel à son implémentation dans le SGBD – fait appel à des techniques et des méthodes très différentes pour chaque étape. Des métiers spécifiques se sont donc développés autour de ces concepts et les mettent en œuvre. Par exemple, l'approche du monde réel s'apparente à l'analyse faite par un cabinet de consulting alors que l'implémentation dans le SGBD et son administration sont proches des métiers informatiques.

Les SGBD ont évolué parallèlement aux concepts de modélisation des bases de données. On est passé d'une organisation comparable à celle des fichiers informatiques (modèles hiérarchiques ou réseaux) à un modèle plus abstrait : le modèle relationnel. Ce modèle est toujours le plus utilisé actuellement. Il est associé étroitement à SQL, un langage normalisé de description, de manipulation et d'interrogation de données. La modélisation objet, adaptée aux bases de données, n'a pas connu un développement considérable, et ce, malgré les avantages qu'elle procure par rapport au modèle relationnel – en particulier pour le typage des données. Comme c'est le cas dans le domaine de la programmation, une approche mixte semble prendre de l'ampleur : le modèle relationnel-objet. Il s'agit d'apporter au modèle relationnel les possibilités étendues de modélisation procurées par les objets sans remettre profondément en question l'existant.

Le développement des réseaux apporte d'autres manières d'utiliser les bases de données, comme la répartition des données pour améliorer leur disponibilité et leur sécurité. L'interfaçage avec le *World Wide Web* a introduit la prise en compte du langage XML comme format d'échange et de stockage par les SGBD. De nouvelles formes d'interrogation, telles que la « fouille de données » (ou data mining) et les bases de données déductives, permettent d'extrapoler de l'information non explicitement stockée dans les bases de données. Ces approches ainsi que la prise en compte des données multimédias vont faire évoluer les modèles de bases de données et les SGBD que l'on utilise actuellement. Cela se fera probablement sans remettre totalement en cause le modèle relationnel, mais plutôt en le faisant évoluer progressivement.