Chapitre 1
Les bases de données

1 Introduction

Le besoin de gérer efficacement les données était soulevé dans les grandes entreprises bien avant l'avènement des premières bases de données. A cette époque, les applications logicielles développées autour d'un SGF (Système de Gestion de Fichiers), seul outil disponible du moment, étaient conçues principalement avec leurs propres fichiers. Le procédé ainsi appliqué a commencé très tôt à montrer ses limites particulièrement dans ces entreprises où le nombre d'applications à gérer s'était accru considérablement se traduisant nécessairement par un nombre explosif de fichiers utilisés et donc une très forte duplication de l'information : une même donnée peut se présenter dans plusieurs fichiers appartenant à des applications distinctes. Outre le gaspillage de la mémoire de stockage et l'augmentation du travail de saisie, le problème peut être aussi la cause d'une anomalie inacceptable ; ainsi, si une donnée redondante est mise à jour dans un fichier mais pas dans les autres (suite à un oubli ou encore un manque de coordination des divers services et départements), elle disposera alors de différentes valeurs en même temps : les données de l'entreprise dérivent en conséquence vers un état erroné ; il est évident que des ennuis ou même des problèmes importants peuvent surgir dans ses activités.

D'un autre côté, ces applications étaient confrontées également à deux autres types de difficultés non négligeables :

- un coût de maintenance trop élevé relativement à l'évolution des données : en effet, comme chaque programme devait décrire l'ensemble des fichiers qu'il utilise, toute modification de ces derniers (en vue de satisfaire de nouveaux besoins) implique une mise à jour des programmes existants concernés. La façon de procéder est donc soumise à un coût trop élevé dû à la maintenance des programmes ;

- un gaspillage de coût dans la programmation des applications : le SGF n'étant utile que pour les entrées/sorties, le programmeur devait tout programmer même les tâches qui se répètent dans chaque application.

Ainsi, afin d'apporter une solution adéquate à ces différents obstacles, les chercheurs ont fait émerger le concept très riche de base de données, sans limite quant à son développement (devenu actuellement un domaine informatique à part entière).

Le présent chapitre, spécifiquement introductif, a pour objectif de fournir les premières explications permettant d'aborder le sujet tout en introduisant les diverses notions et connaissances en rapport : SGBD, modèle de données, type de SGBD, modèles hiérarchique et réseau, architecture ANSI/X3/SPARC, architectures matérielles disponibles, etc.

En fait, ces éléments exposés ici naturellement à l'état brut, serviront implicitement comme support à une partie importante de l'ouvrage et seront de mieux en mieux perçus au fur et à mesure de la progression.

Il est important de remarquer enfin que les bases de données d'aujourd'hui ont largement dépassé les objectifs énoncés ci-dessus (cause principale de leur naissance) pour s'intéresser plus intensément à d'autres considérations exigées par l'environnement professionnel contemporain (communication, sécurité, etc.).

Avant de terminer, rappelons encore que les concepts de base sur les fichiers constituent un prérequis à la suite de l'étude.

2 La notion de base de données

Une base de données peut être vue comme une collection de données relative à une grande fonction au sein d'un organisme (entreprise, université, …), mise en commun à la disposition de tous les utilisateurs (humain ou application logicielle) impliqués dans cette fonction et satisfaisant au moins la condition fondamentale suivante : l'information représentée doit obligatoirement être exhaustive (vis-à-vis des besoins des différents utilisateurs) tout en restant sans duplication aucune.

Les fonctions considérées sont supposées être aussi "vaste" que possible afin d'avoir des bases de données autonomes. Des exemples peuvent être :
- dans une entreprise : l'activité commerciale, la gestion du personnel, le suivi de la chaîne de production, etc.
- dans une université : la scolarité, la gestion du personnel, la planification de l'enseignement, les stages et projets de fin d'étude, etc.
- pour la bibliothèque de l'université : la gestion des ouvrages (englobant les emprunts et l'acquisition).

Les données d'une base sont nécessairement organisées : sous forme d'objets avec leur description ainsi que les liens pouvant exister entre eux.

Ainsi, l'activité commerciale d'une entreprise par exemple exige certainement des objets comme les produits, commandes, factures, clients, etc.
Si celle-ci commercialise des vêtements, les produits peuvent être décrits avec les caractéristiques code, désignation, couleur, prix-unitaire et quantité-stock, sous la forme :
(100, chemise, bleu, 150.00, 200).
De même les factures peuvent comporter un numéro, une date, un montant, un numéro de client et se présenter alors de la manière suivante :
(3000, 10-12-2011, 40500.00, 100).

Concernant les liens entre les objets, il est possible d'avoir : la correspondance entre un client et ses commandes, une commande et les produits qu'elle contient, etc.

Pour la gestion des ouvrages de la bibliothèque de l'université, des objets comme les lecteurs (étudiants et enseignants), livres, auteurs, … sont à considérer.
Un lien possible peut être la correspondance entre les lecteurs et les livres qu'ils ont empruntés.

En anticipant sur la suite et pour être plus précis, une base de données est organisée selon un modèle de données (se référer à la section 4). Dans ces conditions, la sémantique des données est capturée par ce modèle et devient donc indépendante du fichier, contrairement au procédé classique (rappelé en introduction) ; le fichier ne joue alors plus le rôle que d'un outil réservé spécifiquement au stockage.

Il est important de savoir aussi que les données de la base sont toujours soumises à certaines conditions nommées contraintes d'intégrité ; par exemple la note d'un étudiant ne doit pas être située en dehors de l'intervalle [0, 20], le nom et l'adresse d'un client d'une entreprise doivent toujours être connus, etc. ; plusieurs espèces sont disponibles (pour une étude détaillée, se référer à la section 4, chapitre suivant).
Une base de données est dite dans un état cohérent, si l'ensemble de ses données mémorisées respecte les contraintes d'intégrité ; dans le cas contraire, elle est incohérente.

L'approche base de données permet de fournir une gestion centralisée des données de l'organisme à l'inverse de l'approche classique où les fichiers étaient dispersés à travers ses différentes structures. Cette propriété implique de nombreux avantages, notamment :

- la redondance peut être supprimée :
Comme les données sont rassemblées, il devient possible d'utiliser des méthodes convenables afin de faire disparaître le problème en question.
En particulier, le coût de stockage et le travail de saisie sont considérablement réduits dans les grandes entreprises ;

- l'incohérence due aux données dupliquées est écartée :
En effet, puisque la redondance est éliminée, toute mise à jour d'une donnée nécessite d'être effectuée uniquement en un seul endroit de la base ; par conséquent celle-ci ne pourra pas avoir deux valeurs différentes ;

- les données peuvent être sous la responsabilité d'une personne unique :
Cette personne, nommée administrateur de la base de données (abrégé couramment sous la dénomination de DBA : Data Base Administrator), possède de nombreuses tâches comme nous le verrons ;

- la gestion de la sécurité devient plus adaptée :
Il est important de savoir à ce stade que les données d'une base sont continuellement menacées (de destruction, falsification ou consultation non autorisée) par les fraudeurs (Chap. 7). En conséquence, l'accès doit obligatoirement être contrôlé.
Vu qu'elles sont regroupées, la procédure d'octroi des autorisations d'accès (à l'aide de mot de passe par exemple) aux utilisateurs concernant leurs besoins devient centralisée (donc plus

efficace) et peut ainsi être placée directement sous la responsabilité d'une seule personne à savoir le DBA ; ceci n'était pas toujours le cas avec l'approche classique puisque les données étaient dispersées ;

- la conception est améliorée :
Avec l'approche base de données, des procédés remarquablement évolués sont devenus disponibles et permettent enfin de réaliser des bases efficaces (se référer à la partie 4) ;

- des techniques logicielles et matérielles très adéquates peuvent être développées :
Ces moyens (SGBD, architectures matérielles, etc.), particulièrement nécessaires, sont maintenant mis à disposition par différents constructeurs et se trouvent continuellement perfectionnés.

3 Le SGBD

Le SGBD (Système de Gestion de Bases de Données) est le logiciel qui crée et gère les bases de données. Celui-ci offre aux utilisateurs les fonctions de base suivantes :

- un langage de description des données (LDD) : en vue de créer la structure de la base ainsi que les contraintes d'intégrité imposées ;
- un langage de manipulation des données (LMD) : permettant d'interroger la base de données (à l'aide de requêtes) et aussi d'effectuer des mises à jour sur les données (insertion, modification et suppression).

Les avantages qui en découlent sont considérables ; en particulier dans le cadre des systèmes relationnels :

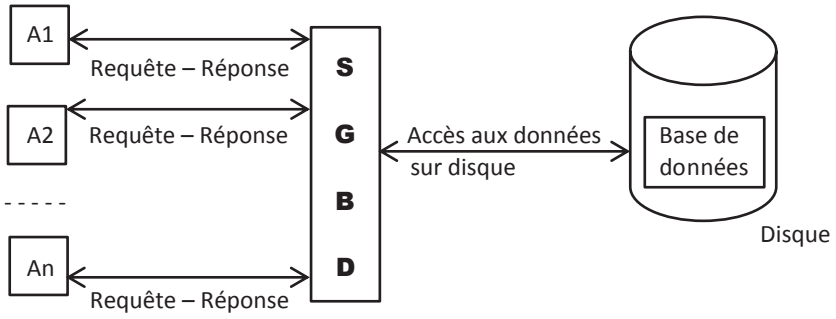
- Le programmeur est déchargé d'une grande partie de la programmation :
Celui-ci n'a plus à décrire ses requêtes par des programmes comme c'était le cas auparavant. Considérons par exemple la requête : donner les commandes de chaque client.
En utilisant le langage SQL (étudié dans la partie 2), il suffit d'écrire tout simplement :
SELECT * FROM CLIENT NATURAL JOIN COMMANDE ;
Et c'est au SGBD de remplacer cet énoncé par les instructions nécessaires à l'opposé du SGF où l'application doit fournir tout un programme.

- Le code des applications est notablement réduit et devient plus lisible :
Cet avantage est une conséquence directe du point précédent.

- Les deux langages sont de haut niveau (en se référant au standard SQL) :
L'utilisation ne nécessite donc pas beaucoup de compétence pour les opérations courantes.

- La formulation des requêtes est simplifiée :
Les langages en question étant de haut niveau, l'expression des requêtes devient plus naturelle. Il suffit à l'utilisateur de formuler seulement ce qu'il veut et non la manière d'obtenir les réponses (en programmant).

En fait, le SGBD joue le rôle d'interface entre les applications logicielles et la base de données ; ces dernières ne peuvent donc accéder à celle-ci qu'en lui soumettant leurs commandes.



A1, A2, …, An : désignent les applications logicielles.

Le principe général de l'évaluation se déroule ainsi :

1) L'application transmet sa requête au SGBD sous forme d'une commande conforme au langage offert.
   Par exemple la requête : quels sont les articles dont la quantité en stock est inférieure ou égale à 10, s'exprime dans le langage SQL :
   SELECT * FROM ARTICLE WHERE quantite_stock <= 10 ;
2) Le SGBD étudie la commande afin de déterminer l'ensemble des données nécessaires à l'évaluation.
   Pour l'exemple, il s'agit de tous les articles stockés.
3) Effectue des accès disque en vue de ramener ces données en mémoire centrale.
4) Traite ces données afin de déterminer la réponse à la requête.
   En revenant à l'exemple, le SGBD consulte chaque article et le sélectionne si la quantité en stock est inférieure ou égale à 10.
5) Renvoie enfin le résultat obtenu à l'application concernée.

Il existe deux façons différentes pour soumettre les requêtes au SGBD :

- en mode interactif : l'utilisateur humain tape la commande à l'aide de son clavier et la transmet au SGBD (en réalité, celui-ci se sert d'un outil — généralement mis à disposition par le SGBD — qui prend en charge ses commandes, les envoie au SGBD, récupère ensuite les résultats puis finalement les affiche) ;
- intégrées dans un programme : ici la commande est placée au sein d'un programme ; afin de l'évaluer, il faut exécuter ce dernier. Le langage employé (Java, C++, Pascal, C, …) dans l'écriture du code est appelé langage hôte.

La première possibilité est intéressante envers les requêtes occasionnelles alors que la seconde pour les requêtes réutilisables (pouvant se répéter un grand nombre de fois dans le temps). Tout SGBD doit donc nécessairement mettre à la disposition des programmeurs un certain nombre de langages hôtes.

4 Modèle de données et type de SGBD

Un modèle de données est un ensemble de concepts permettant d'établir une description formelle de toutes les données de la base à créer. Son objectif principal consiste à organiser ces données sous forme d'objets de même espèce avec leurs caractéristiques et d'exprimer aussi les liens pouvant exister entre eux.

Des exemples d'objets et liens ont été donnés dans la section 2 pour l'activité commerciale d'une entreprise ainsi que la gestion des ouvrages d'une bibliothèque universitaire.

Les modèle de données diffèrent principalement par :
- les types de données proposés ;
- la manière de décrire les liens (relations entre les objets) ;
- les notations mises en jeu.

Tout SGBD crée et gère les bases de données conformément à un modèle de données.

A chaque modèle supporté par un SGBD doit correspondre au moins un langage permettant la description et la manipulation des données (c'est-à-dire un LDD et un LMD).

Les SGBD se distinguent et sont classés fondamentalement par les modèles de données qu'ils proposent ; nous avons :

- Les modèles hiérarchique et réseau :
Ils correspondent à la première génération de SGBD ; ces modèles ont débuté vers le milieu des années 60 et ont eu du succès jusqu'aux années 80.
Comme exemple de SGBD hiérarchique, on peut citer IMS.
Pour les SGBD réseau : IDMS, SOCRATE, IDS-II.

Les SGBD réseau sont plus puissants que les SGBD hiérarchique et sont apparus après ces derniers afin de les améliorer.

- Le modèle relationnel :
C'est la deuxième génération. Le modèle relationnel a été introduit en 1969 par le chercheur Codd d'IBM [5].
Les SGBD relationnels ont commencé à être commercialisés au début des années 1980 ; actuellement, ils dominent le marché des SGBD.
Un nombre remarquable de systèmes est proposé, notamment : ORACLE, DB2, SQL SERVER, INFORMIX, ACCES, MySQL, etc.

- Les modèles objets-relationnels et objets :
Il s'agit de la troisième génération. Ces modèles ont été introduits afin de pouvoir gérer des données complexes (images, cartes, documents, …) et ont commencé à s'imposer sur le marché au début des années 90 (en particulier pour répondre à la nouvelle génération des applications internet).

Le modèle objets-relationnels constitue une extension du modèle relationnel ; il complète ce dernier par tous les concepts nécessaires à la gestion des données complexes. Comme produit très répandu, on peut citer le SGBD Oracle à partir de la version 8.
Le modèle objets quant à lui, est basé sur les principes de la technologie objets (domaine du génie logiciel). Parmi les systèmes les plus connus figurent, notamment Object Store et Gemstone.
Sur le plan commercial, les premiers, distribués principalement par des sociétés très puissantes, occupent sans conteste la plus grande partie du marché.
De leur côté, les SGBD objets, bien que fondés aussi sur un modèle très développé, sont confrontés au problème crucial de la portabilité (facilité de fonctionnement des applications en changeant l'environnement de création) et n'arrivent toujours pas à se hisser au stade des premiers.

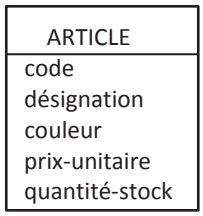
5 Les modèles de la première génération

Les premiers vrais modèles apparus pour représenter les bases de données sont incontestablement les modèles hiérarchique et réseau que nous allons expliquer dans cette section. Bien que dépassés maintenant, ils restent tout de même très remarquables sur le plan pédagogique (le modèle relationnel utilisé actuellement et que nous introduirons au chapitre 2 est tellement plus simple qu'il dissimule toute la complexité du concept).

5.1 Le modèle hiérarchique

Les notions de base sont le segment, les champs, l'occurrence d'un segment, les clés et le lien dont les définitions sont les suivantes :
Le segment désigne un ensemble d'objets de même espèce.
Une occurrence d'un segment est un objet particulier.
Les caractéristiques permettant de décrire les objets sont appelées champs du segment.
Une clé d'un segment est un ensemble de champs dont les valeurs permettent de distinguer ses occurrences.

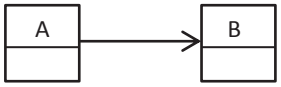
Un segment est représenté graphiquement sous la forme :



Dans cette notation, le segment ARTICLE modélise l'ensemble des articles à gérer. Il a pour champs : code, désignation, couleur, prix-unitaire et quantité-stock ; le premier représente une clé. Des occurrences possibles peuvent être (en supposant que les objets sont des vêtements) : (10, pantalon, noir, 200.00, 40) ; (11, veste, bleu, 400.00, 30), etc.

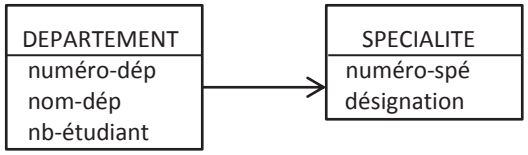
Considérons maintenant deux segments A et B. Dans ce modèle, le seul type de lien pouvant exister entre A et B est le lien 1 : n (un à plusieurs) dont la représentation graphique s'effectue

à l'aide d'une flèche sans dénomination sous la forme :



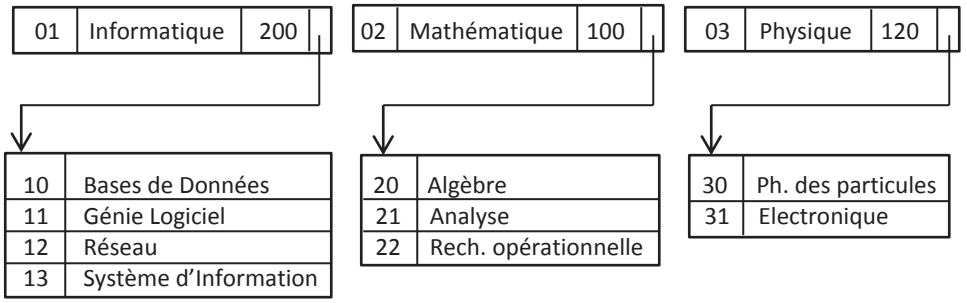
La signification est la suivante : à toute occurrence de A peut correspondre une ou plusieurs occurrences de B.
Dans cette représentation A est appelé le segment père, B le fils.
Ce modèle ne permet donc au plus qu'un lien entre deux segments puisque la flèche ne porte pas de nom.

Exemple :



Le lien établi ici associe à chaque département, les spécialités enseignées.

Les données sont organisées par le SGBD en se basant sur les pointeurs et la contiguïté :



On peut avoir autant de niveau hiérarchique que l'on veut :