# 1. Introduction

## Contenu du chapitre 1

Ce chapitre d'introduction présente les concepts de base du domaine couvert par ce livre et annonce le programme de ce qui suit. Il explique ce qu'est une base de données, ce que l'on entend par modélisation des données et par modèle des données. Il met en évidence l'importance de cette modélisation dans le cadre d'un projet informatique. Il explique comment une base de données est structurée. On y trouve aussi une définition provisoire (on y revient dans le chapitre 3) des principaux termes utilisés.

Ce chapitre sert aussi de porte d'entrée aux personnes sans connaissance aucune des bases de données et de la modélisation. Nous avons jugé bon de commencer par là, vu que, dans la majorité des cas, les équipes chargées des projets informatiques comprennent des personnes qui sont spécialistes de leur métier dans l'organisation, mais n'ont en général pas ou peu de connaissances en informatique. Ce chapitre est écrit d'abord à leur intention, les informaticiens le parcourront en vitesse. Pour ce qui est des débutants, une fois ces connaissances de base acquises, ils devraient pouvoir suivre le reste sans trop de difficultés.

Ces non-informaticiens jouent un rôle essentiel dans les projets. Ce sont eux qui apportent les connaissances de la branche, expriment leurs besoins en matière d'informations à gérer et de fonctionnalités désirées. Ils expliquent aux informaticiens la nature et les détails du domaine de gestion couvert par la future application. Ils vont rédiger le cahier des charges du futur système ou aideront les informaticiens à le faire. Ils vont valider les prototypes et le système terminé. Il faut donc que ces personnes aient compris de quoi il est question ici afin de pouvoir jouer leur rôle de manière efficace.

Néanmoins: la modélisation est un art difficile qui s'apprivoise. Au cours des nombreux séminaires organisé avec des informaticiens et des non-informaticiens, ainsi qu'au cours de séances de modélisation dans le cadre de projets concrets, nous avons maintes fois constaté que certains informaticiens ne possédaient aucun talent pour cette activité alors que des béotiens absolus du domaine devenaient rapidement des cracs. Que ceux pour qui le sujet traité ici est d'une nouveauté totale se rassurent donc. La seule qualité vraiment requise est un esprit ouvert et logique.

## La base de données d'un système

Un système informatique moderne comprend toujours une **base de données**. Nous expliquerons plus tard de quoi il s'agit exactement. Disons pour le moment qu'il s'agit d'une structure qui contient toutes les informations liées à une application informatique particulière. Dans un système de gestion de commandes de clients, par exemple, elle contient les adresses de ces clients, les articles vendus, les commandes enregistrées, etc. Autour de cette base de données, le reste de l'application consiste en programmes qui chargent des données dans cette base (par exemple lors de la saisie d'une commande) ou exploitent les données contenues dans cette base (par exemple pour l'impression d'un bulletin de livraison ou la production de statistiques de vente).

## Concevoir la base de données

Une activité importante de tout projet informatique consiste alors dans la spécification de la structure de cette base de données. Déterminer quelles sont les données qui doivent y figurer et sous quelle forme. On appelle cette activité la **modélisation des données** et son résultat est un **modèle de données** qui décrit la structure de la base. Le présent livre ne traite que d'un seul sujet: comment construire ce modèle. On ne peut pas sous-estimer l'importance de ce modèle et du travail de modélisation. Si une donnée est oubliée lors de ce travail, elle ne pourra pas être gérée par la suite. Bien sûr, la découverte d'une telle lacune peut en général être corrigée plus ou moins facilement. Mais si le travail de modélisation est mal fait, des corrections onéreuses et longues deviendront la règle au lieu de l'exception et personne ne sera satisfait du résultat. Il en sera de même si les données sont mal organisées. Leur exploitation sera alors inefficace et le développement de programmes plus long et plus coûteux.

Il est trop tôt ici pour expliquer ce qu'est une base de données bien ou mal structurée. Ce sera fait par la suite. Pour le moment insistons simplement sur le fait que tout projet informatique implique en général de:

**Créer une base de données complète et bien structurée.**

Pour faciliter le processus d'initiation, nous conseillons aux débutants de commencer par consacrer un moment, éventuellement avec l'aide d'une personne susceptible de les guider, à se familiariser avec un **système de gestion de base de données** (SGBD) sur PC, Access de Microsoft, par exemple. Ce n'est pas très difficile! Définir une ou deux tables, saisir des valeurs dans ces tables, créer des requêtes pour extraire des données. Les exemples présentés dans ce chapitre se prêtent parfaitement à une telle initiation.

L'alternative consiste à réunir les personnes qui participeront au processus de modélisation et à leur donner un petit cours sur la matière présentée dans ce chapitre. Une ou deux heures suffiront amplement car, effectué en équipe formée à la fois d'informaticiens et de non-informaticiens, le travail de modélisation réel constituera le meilleur apprentissage.

## Architecture des systèmes informatiques modernes

Tous les progiciels (le terme signifie produit logiciel) et tous les systèmes informatiques modernes utilisés en gestion sont aujourd'hui construits autour d'une **base de données**. Cela s'applique également et notamment aux grands prodduits très répandus tels que SAP R/3 ou Oracle e-Business Suite. Cette base de données est gérée au moyen d'un **système de gestion de base de données** (SGBD), outil standard dont les plus répandus à l'heure actuelle sont Oracle Database, UDB d'IBM, SQL Server de Microsoft ou encore MySQL de Sun Microsystems.

Une base de données contient de façon permanente toutes les données gérées par l'entreprise, globalement ou dans le cadre d'une application particulière. Pour des raisons de sécurité, de sauvegarde, de pérennité, aucune information de nature permanente ne devrait être stockée ailleurs que dans une base de données centralisée, partagée, soumise à une administration professionnelle. Les données stockées individuellement sur un PC, dans le contexte de tableaux par exemple, appelées quelques fois "îlots d'informations", représentent un danger pour l'entreprise et doivent être évités dans la mesure du possible. Lorsqu'un collaborateur se met à gérer des données dans un tableur tel qu'Excel sur son PC, ses collègues n'y ont plus accès et rien en garantit que ces données ne seront pas perdues s'il arrive un malheur au PC ou à la personne. L'information doit être centralisée, partagée, accessible à toutes les collaborateurs (dans la mesure des autorisations qui leur sont accordées, évidemment, un aspect que les SGBD sont en mesure de faire respecter).

Dans sa forme la plus simple, une base de données n'effectue elle-même aucun traitement: elle n'est qu'un réservoir recevant des données, les préservant et les mettant à disposition par la suite.

Une base de données n'est en fait rien d'autre qu'un ensemble de fichiers qui pourraient (théoriquement!) être tenus manuellement sur des paquets de cartes. Le SGBD, pour sa part, est un programme standard qui automatise et facilite la création et la définition de fichiers, l'enregistrement de données sur des fiches, la modification de données enregistrées, la suppression de fiches et, surtout, la mise à disposition de d'informations recherchées ou encore l'élaboration de statistiques à partir de données sélectionnées.

On peut affirmer que les programmes contenus dans les grandes applications informatiques utilisées dans les entreprises, tels que les systèmes de gestion des ventes, des achats et de la production (ERP ou GPAO), de gestion des relations avec la clientèle (CRM) ou encore de gestion du personnel et des salaires (RH) ne sont rient d'autres que des interfaces entre les utilisateurs de ces systèmes et une base de données.

## Base de données: un exemple

Dans une entreprise de vente, par exemple, la base de données contiendra entre autres (cette présentation fortement simplifiée!):

- le fichier des clients enregistrés, avec toutes les informations qui les concernent;
- le catalogue des articles avec toutes les indications nécessaires (désignation, prix de vente, prix d'achat, catégorie, niveau de stock);
- le carnet des commandes passées, avec indication, pour chacune d'entre elles, du client, de la date de commande, des articles commandés, de la quantité commandée, du prix et du délai convenus, ce qui a été livré;
- les factures émises avec montant et date, la date de paiement;
- les mouvements de stock, (article, type de mouvement, quantité, date);
- les comptes financiers et les écritures passées sur ces comptes.
- etc.

## Structure de la base de données

Lorsqu'on se consacre à la modélisation des données, sujet de ce livre et partie essentielle de chaque projet de développement de logiciel en gestion, un seul sujet intéresse: déterminer la structure de la future base de données. Quels fichiers (nous dirons **table** ou **entité** au lieu de fichier par la suite) doit-elle contenir, quelles données doivent être stockées sur chaque type de fiche. De quelle façon ces fiches seront alimentées ou exploitées n'est d'aucun intérêt à ce stade. Ce sont les traitements, les programmes nécessaires à l'application. Ils seront définis à un autre moment.

On pourrait considérer la base de données d'une entreprise (ou d'une application) de façon isolée, indépendamment de toute exploitation des données qui y sont stockées. Cette base doit simplement contenir *toutes* les informations nécessaires à la bonne marche de l'entreprise/application et ces informations doivent être structurées de manière idéale (nous expliquerons dans la partie principale de livre ce que nous entendons par là).

Précisons tout de même qu'une séparation totale de l'analyse des données et des traitements n'est pas vraiment réaliste. D'une part ce sont évidemment les processus de gestion, familiers aux utilisateurs des applications, qui déterminent quelles informations doivent être gérées, donc contenues dans la base. En outre, dans la réalité, une base de données peut contenir elle-même des parties de programmes qui effectuent certaines opérations. Il s'agit notamment, mais pas exclusivement, de parties de programmes chargés de la vérification de la plausibilité de ces données.

Néanmoins, pour des raisons pédagogiques, nous conseillons d'adopter pour le moment le parti pris d'une séparation absolue entre données et utilisations de ces données. L'objectif est de construire une base de données complète et correctement structurée. Aux programmeurs ensuite de développer les programmes pour alimenter cette base et en exploiter le contenu de manière efficace. Aussi longtemps que les informations nécessaires figureront dans la base, il sera toujours possible de les mettre à disposition d'une façon ou d'une autre. Si, par contre, des informations manquent dans la base, aucune programmation, aussi géniale soit-elle, ne permettra de les restituer.

## Base de données, tables, lignes, colonnes, attributs

Les systèmes de gestion de base de données actuels, appelées relationnels, enregistrent les données dans un ensemble de **tables**, chacune composée de **lignes**.

**La base de données est un ensemble de tables**

**La table est un ensemble de lignes**

Une table est l'équivalent d'un fichier composé de fiches (cartes) telles qu'on les connaissait avant l'informatique. Dans une base de données:

Un fichier devient une table

Une fiche devient une ligne d'une table

Dans un fichier, chaque fiche contient des informations sur un *objet différent*, mais toutes les cartes d'un même fichier contiennent des informations sur un *seul et même **type** d'objet*. Si l'on gère différents types d'objets, on utilise différents fichiers. Exemple: le fichier des membres d'une société est composé de cartes portant chacune les informations sur un membre. Si l'on veut également gérer les manifestations organisées par cette société, on construit un deuxième fichier, dans lequel chaque carte porte les détails d'une manifestation. Il ne viendrait à aucune personne ordrée l'idée de mélanger dans un même paquet de fiches des cartes de membres et des cartes de manifestations.

Il en est de même dans une base de données. Chaque table contient un ensemble de lignes qui comprennent chacune les informations concernant un **objet** spécifique d'un même **type**. Reprenant l'exemple ci-dessus, on aurait une table des membres avec une ligne par membre et une table des manifestations avec une ligne par manifestation.

Il existe pourtant une différence entre les fiches traditionnelles et les tables dans une base de données. Sur une fiche papier, je peux librement structurer mes données. Je ne suis pas obligé (mais c'est évidemment recommandable) de disposer les données des membres de façon totalement identique sur chaque carte. Par contre, dans une table d'une base de données, les informations sur un objet sont, dans chaque ligne, toujours les mêmes et présentées dans le même ordre. **Toutes les lignes d'une table possèdent exactement la même structure et contiennent les mêmes informations sur des objets différents.**

Les informations sont donc stockées en colonnes et chaque colonne d'une table contient un type d'information différent, le même dans chaque ligne. On désigne les colonnes par le terme **champ** ou, mieux, **attribut de l'objet**.

**En résumé:**

- **Une table contient les informations liées à un ensemble d'objets d'un même type.**
- **Ces informations sont stockées sous forme de lignes et de colonnes.**
- **Chaque ligne décrit un objet particulier avec ses attributs**
- **Chaque colonne décrit un attribut particulier de l'ensemble des objets.**

L'ensemble des définitions sera repris dans le chapitre 3.

## Exemple: livres dans une bibliothèque

La table *Livres* est un tableau composé de lignes décrivant chacune un livre particulier. De plus, chaque ligne contient, pour tous les livres, exactement les mêmes attributs dans le même ordre.

Ces attributs sont ici: le numéro ISBN, l'auteur, le titre, l'éditeur, l'emplacement de stockage.

| Numéro ISBN | Auteur | Titre | Éditeur | Rayon |
|---|---|---|---|---|
| 2-200-42002-1 | Patrick Jaulent | Génie logiciel | Armand Colin | A3 |
| 0-340-67207-2 | Nick Rosen | The Net-head Handbook | Hodder&Stoughton | A2 |
| 3-9520606-0-7 | René Egli | Das LOL$^2$A-Prinzip | Editions d'Olt | C7 |
| 0-201-32563-2 | Martin Fowler | UML Distilled | Addison-Wesley | E3 |

Figure 1.1: structure de la table *Livres*.

## Création de la base de données réelle

Une fois le travail de modélisation terminé et validé, l'**administrateur de la base de données** crée sur l'ordinateur une base de données avec, dans cette base, une table pour chaque entité du modèle. Chaque attribut de chaque entité devient une colonne dans la table correspondante. De fait, certains programmes appelés outils de génie logiciel effectuent automatiquement la création d'une telle base à partir d'un modèle établi lors de l'analyse des données. Ces outils permettent aussi de saisir un tel modèle sous forme graphique et de le documenter.

Ceci dit, nous ne discuterons plus dans ce livre des bases de données créées dans la réalité sur un ordinateur, mais exclusivement de la modélisation et des modèles de données.

## Exemple de modèle de données

Le résultat (fortement simplifié) d'une modélisation des données se présentera typiquement comme suit pour une petite application de gestion des ventes.

On énumère d'une part la liste des tables avec leurs attributs entre parenthèses

Client (numéro client, raison sociale, rue et no, code postal, localité, pays)

Article (code article, désignation, catégorie, prix de base)

Commande (no de commande, client, date commande)

Ligne de commande (no de commande, no de ligne, article, quantité, prix)

D'autre part on présente la structure de la base de données sous forme d'un schéma graphique. Dans le cas ci-dessus, le graphique sera le suivant:

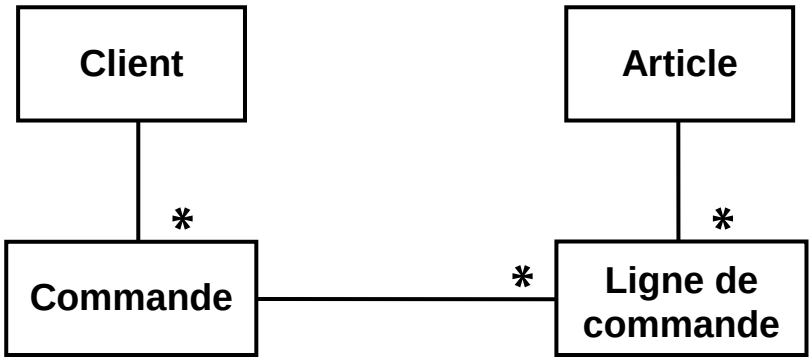


Fig. 1.2. Modèle de données, gestion des ventes

Dans ce graphique, chacun des symboles rectangulaires représente un objet *quelconque* des différentes tables ("un" client, "un" article, etc.). Les lignes qui relient les rectangles montrent les associations qui existent entre ces objets. Le symbole étoile (*) indique que plusieurs objets d'un type peuvent être associés à un objet unique figurant à l'autre bout de l'association. Ainsi, un client peut être associé à (peut passer) plusieurs commandes, mais une commande n'est associée qu'à un seul client. Une commande peut se composer de plusieurs lignes de commandes. Chaque ligne de commande est associée à

un article, mais un article peut figurer dans plusieurs lignes. Tout ceci sera discuté en détail dans le prochain chapitre. Il ne s'agit que d'un premier exemple.

## Types de données

Dans cette brève introduction sur les bases de données, il reste un point important à expliciter: c'est celui du type de chaque attribut. Chaque attribut d'une entité possède un type défini. La notion décrit le genre de valeurs que l'attribut peut posséder. On distingue entre types de base et types dérivés. Les types de base sont de nature très technique, leur contenu sémantique est faible. Les types dérivés, composés à partir des types de base possèdent un contenu sémantique précis. On pourrait par exemple définir un type *adresse*, composé de *rue* (alphabétique), *no* (entier), *code postal* (entier), localité (alphabétique). En fait, la définition d'un type d'objet est un type dérivé.

Les types de base les plus courants sont les suivants:

| **Type de données** | **Valeurs possibles de l'attribut** |
|---|---|
| Alphabétique | Texte de longueur limitée, formé des lettres de l'alphabète et de caractères de ponctuation. Aucun chiffre. Usage limité. |
| Alphanumérique | Texte de longueur limitée, formé de lettres, chiffres et caractères spéciaux. Aucune opération mathématique n'est possible. Type couramment utilisé pour des attributs tels que noms, désignations, parties d'adresses, libellés, etc. Tri par valeur possible. Extraction de parties possible (le premier mot, les cinq derniers caractères, etc.) |
| Numérique | Nombre sur lequel il est possible d'effectuer des opérations mathématiques et des comparaisons de valeurs. On distingue plusieurs types numériques, les plus importants sont décrits ci-dessous. |
| Entier | Nombre entier uniquement (positif ou négatif) |
| Décimal | Nombre (positif ou négatif) possédant un nombre défini de chiffres significatifs et de décimales après le point (la virgule). Décimal(9,2) signifie par exemple que le nombre peut contenir 9 chiffres au maximum, dont deux après le point décimal et donc 7 avant. Le maximum pouvant être stocké dans un tel nombre vaut 9'999'999.99. |
| Date/heure | Date valable / heure valable. Il est en général possible de faire des opérations mathématiques sur les dates et les heures (nombre de jours entre deux dates, heure + intervalle de temps, etc.). |
| Logique (booléen) | Valeur "vrai" ou "faux". |
| Texte | Texte de longueur quelconque |
| Blob | Image, texte, séquence sonore ou multimédia |

Les systèmes de gestion de bases de données offrent une grande variété de types de base. Ils exigent que lorsqu'un attribut est défini, son type soit précisément indiqué. Aucune valeur non conforme ne sera acceptée et enregistrée par la suite. Il ne sera pas possible par exemple de stocker un texte dans un champ supposé contenir une date ou un nombre.

Nous recommandons encore une fois aux novices de s'initier un tant soit peu à un outil tel qu'Access de Microsoft pour bien comprendre cette notion de type de données. À titre d'exercice, le lecteur peut essayer de déterminer le type correct des attributs figurant dans les exemples de ce chapitre.