

Définition d'une base de données

Par définition, une base de données représente une collection de renseignements classés dans des tables. Elle permet de stocker une grande quantité d'informations, puis de les mettre à jour facilement.

Access est un *Système de Gestion de Base de Données* (SGBD). Il permet d'assurer la gestion de l'information par le biais des différents objets qu'il contient : tables, formulaires, requêtes, états. À l'aide de ces objets, on peut facilement saisir, modifier, supprimer ou trier l'information. Il est aussi possible d'extraire des données selon différents critères prédéfinis, puis d'imprimer le résultat.

Les bases de données d'hier et d'aujourd'hui

Autrefois, l'information était conservée sur papier, à l'intérieur de dossiers rangés dans des classeurs. Ces classeurs contenaient par exemple l'ensemble des dossiers des employés et des clients d'une entreprise, ce qu'on appelle aujourd'hui, à l'ère de l'informatique, une base de données. En effet, un classeur Access contenant par exemple tous les dossiers des clients d'une entreprise constitue une base de données clients.

Afin de bien comprendre en quoi un système de gestion de base de données comme Access peut être utile, effectuons d'abord quelques comparaisons entre un bon vieux classeur en métal gris et Access, au moyen de l'exemple suivant :

Une *nouvelle* compagnie, Papeterie inc., ouvre ses portes en... 1965. (Vous l'aurez deviné : l'ordinateur de bureau n'est pas encore de ce monde !) La nouvelle compagnie vendra des crayons, des boîtes de papier, des gommes à effacer et des enveloppes. Comme les ventes seront effectuées par téléphone, l'entreprise comprend une salle pouvant accueillir 25 téléphonistes prêtes à prendre les commandes.

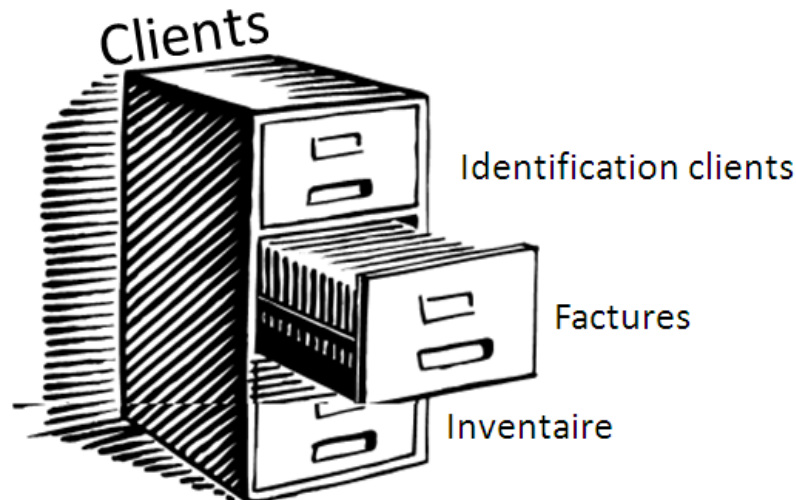
Dès l'ouverture, le patron achète un tout nouveau classeur (en métal gris...) et explique ce qui suit à ses employés :

« Voici le classeur que nous allons utiliser. Je l'ai nommé le *classeur clients*. »

« Chaque tiroir porte également un nom. Le premier se nomme *Identification clients*, et il contiendra les renseignements servant à identifier les clients, comme leur nom, adresse, numéro de téléphone, personne-ressource, etc. »

« Le deuxième tiroir porte le nom *Factures*, car une copie de chaque facture envoyée aux clients y sera conservée. »

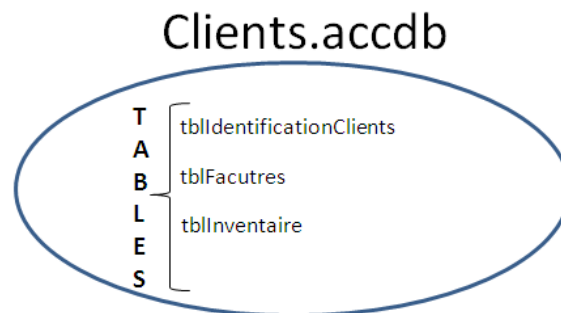
« Le dernier se nomme *Inventaire*. Mais non ! Il ne contiendra pas l'inventaire, bien entendu ! Il contiendra plutôt des feuilles indiquant les quantités de chaque produit en inventaire. Ainsi, lorsqu'un client placera une commande, vous serez en mesure de lui indiquer si les articles qu'il désire sont en inventaire. »



Vous le constatez peut-être déjà : ce système de classement nous rappelle effectivement... 1965 !

L'organisation des bases de données

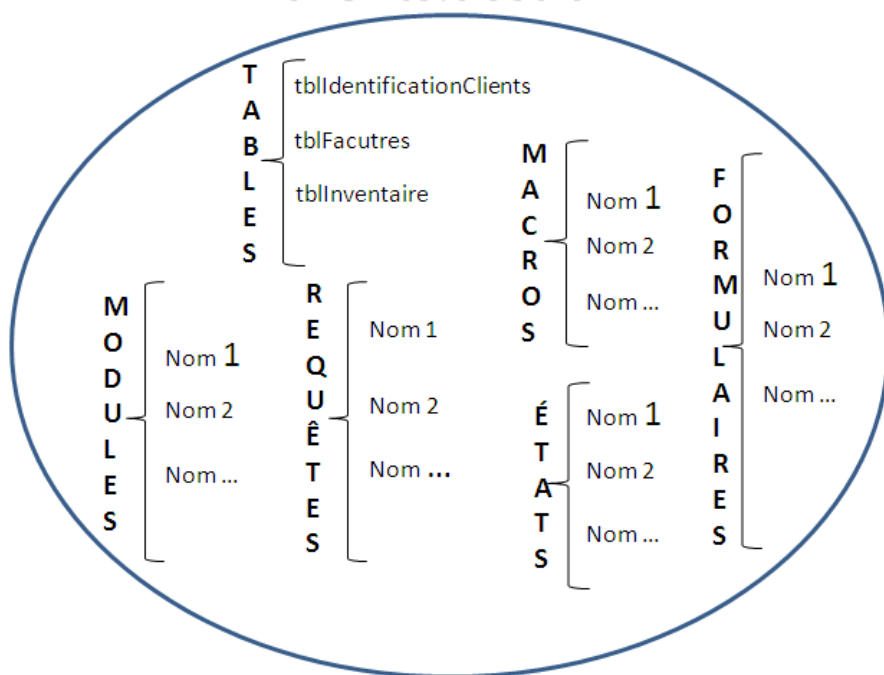
Dans Access, il faut d'abord créer un fichier **Base de données**. On peut le nommer *Clients*, tout comme le classeur en métal gris présenté plus tôt. Un fichier Access ne pouvant cependant pas contenir de tiroirs, l'information est stockée dans ce qu'on appelle des **tables**, comme ici :



*L'extension **accdb** est ajoutée de façon automatique par le logiciel Access à tout fichier base de données.*

Comme on le constate, un fichier base de données Access est assez semblable au classeur en métal et ses tiroirs, à l'exception qu'il peut contenir de une à 256 tables. Un classeur de 256 tiroirs, voilà qui mobiliserait beaucoup d'espace... Non seulement la base de données Access peut contenir un grand nombre de **tables**, mais elle peut également contenir plusieurs **formulaires, requêtes, états, macros et modules**. Ce sont les **objets** de la base de données. En les utilisant tous, on obtient ce qui suit :

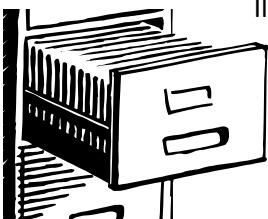
Clients.accdb



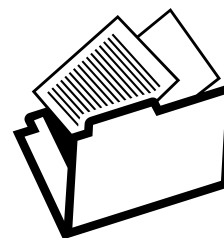
Comme un fichier **base de données** peut contenir plusieurs objets différents, il est inutile de créer une nouvelle base de données pour chaque table désirée. Précisons également qu'on ne peut visualiser le contenu d'une base de données dans l'**Explorateur Windows**, le logiciel **Access** étant essentiel pour ce faire.

Revenons à notre exemple de tantôt...

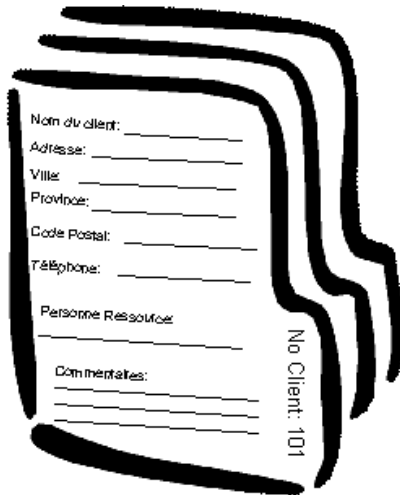
La compagnie Papeterie inc. connaît un succès rapide, et compte un grand nombre de clients après quelques mois d'activité. Le patron décide donc d'effectuer une vérification des dossiers clients contenus dans le premier tiroir de son classeur.



Il remarque alors que l'information contenue dans le tiroir *Identification clients* diffère pour chaque dossier. Le nom des clients et leur adresse y sont bien inscrits, mais certains dossiers contiennent beaucoup plus de détails que d'autres.



En quête de cohérence, le patron décide de faire imprimer des dossiers sur lesquels est précisée l'information que les employés doivent obtenir de chaque client au moment de leur ouvrir un compte. Chaque dossier client aura désormais exactement la structure suivante :



Une fois définie la structure des dossiers *Identification clients*, il ne reste plus qu'à inscrire pour chaque nouveau client les informations correspondantes, dans la partie de droite.

Notions de champ et d'enregistrement

Comparons maintenant les dossiers *Identification clients* du classeur en métal gris à une table Access. On dit que dans une table, l'information est présentée en lignes et en colonnes, comme ci-dessous :

NoClient	Nom	Adresse	Ville	Province	Téléphone	Fax	Personne ressource	Commentaires
_____	_____	_____	_____	_____	_____	_____	_____	_____
_____	_____	_____	_____	_____	_____	_____	_____	_____
_____	_____	_____	_____	_____	_____	_____	_____	_____

On peut le constater facilement : l'information demeure la même que dans les dossiers *Identification clients*, sauf qu'elle est maintenant positionnée en lignes.

Et si chaque dossier dans le classeur en métal contient toute l'information sur un seul client, on peut dire que dans une table Access, c'est une ligne, ou plus précisément un **enregistrement**, qui présente toute l'information sur un seul client.

Voyons maintenant l'information contenue dans la colonne *Nom*. Elle contient une partie seulement de l'information regroupée dans chaque ligne. Il s'agit d'un **champ**. Ainsi, plusieurs champs composent un enregistrement, et plusieurs enregistrements composent une table.

Par définition, une table représente donc une collection d'informations regroupées selon une même structure, présentées en lignes et en colonnes. Chaque ligne représente un enregistrement, et chaque colonne, un champ.

La clé primaire et ses rôles

Revenons au premier tiroir du classeur. Comme le patron trouvait fastidieux de classer les dossiers clients en ordre alphabétique, il décida de le faire par numéros, en attribuant un

numéro à chaque dossier client. Il fallait toutefois s'assurer qu'un même numéro ne soit pas attribué à deux clients différents, une erreur très courante à l'époque...

Dans une table Access, si l'on désire que les données d'un champ soient uniques, on définit celui-ci comme étant la **clé primaire**. La clé primaire joue trois rôles :

- le premier consiste à préciser l'ordre dans lequel seront affichés les enregistrements (ordre de classement, de tri), peu importe l'ordre dans lequel ils auront été entrés;
- son deuxième rôle est d'assurer l'unicité des données de ce champ;
- le troisième rôle de la clé primaire trouve son sens lors de l'établissement d'un lien *un-à-plusieurs*. Dans ce cas, le côté *un* de la relation doit obligatoirement être la clé primaire. (Les notions de relations *un-à-plusieurs* sont abordées dans le manuel *Access 2007, Élaboration d'une base de données.*)

Donc, si le champ *Numéro de client* est défini comme étant la clé primaire dans une table Access, tous les enregistrements qu'il contient seront affichés en ordre croissant (ou décroissant), et Access veillera à ce qu'un même numéro ne soit jamais attribué à deux clients, puisqu'il ne tolère pas les **doublons**.

Microsoft recommande de définir une clé primaire dans chaque table. Il est donc important de définir un champ qui contienne des données uniques.

Les risques d'erreurs

Revenons encore une fois à notre vieux classeur. Chaque fois qu'un client passait une commande, les opérations suivantes devaient être effectuées :

- demander d'abord le numéro du client;
- sortir ensuite le dossier du client;
- confirmer son nom et ses coordonnées;
- prendre en note sa commande;
- aller vérifier dans le 3^e tiroir la disponibilité des articles;
- produire la facture en inscrivant le nom du client, ses coordonnées et les articles commandés.

Et si le même client passait une nouvelle commande trois jours plus tard, il fallait recommencer tout le processus ! L'information était donc sans cesse réinscrite, ce qui multipliait les risques d'erreurs.

Les relations entre les tables

Revenons à Access, dans lequel nous avons déjà créé une table *Identification Clients*.

Afin de produire une facture pour un client, il serait de mise d'utiliser l'information que contient la table *Identification Clients*. Ainsi, la table facture ne contiendra pas le nom et l'adresse du client, mais uniquement un champ donnant accès à cette information, soit le *numéro du client*.

No. client	Nom	Adresse	Ville	Province....
100	Soft Logique SJP Inc.	28, rue aaa	St-Hubert	Québec
101	Les Éditions Renaissens	82, rue ccc	Montréal	Québec
102	Les Productions KJB Inc.	66, rue aaa	Montréal	Québec
103	La compagnie Profit Inc.	59, rue aaa	Montréal	Québec

No. facture	Date	No. client	No. produit	Qté commandée....
F9822	10 jan 07	100	PR8888	2

Cette façon de faire évite de retranscrire des informations déjà existantes, ce qui minimise le risque d'erreur. Mais s'il est diminué, le risque d'erreur n'est pas pour autant enrayé, car si l'information dans la table *Identification Clients* contient une erreur, chaque facture qui sera produite pour ce client contiendra l'erreur. Mais en revanche, une fois l'erreur corrigée, elle le sera sur toutes les factures en même temps !

Comme on peut le remarquer, la table *Factures* ne contient ni le nom du produit ni son prix, mais uniquement un champ donnant accès à cette information, soit le *numéro du produit*.

No. client	Nom	Adresse	Ville	Province....
100	Soft Logique SJP Inc.	28, rue aaa	St-Hubert	Québec
101	Les Éditions Renaissens	82, rue ccc	Montréal	Québec
102	Les Productions KJB Inc.	66, rue aaa	Montréal	Québec
103	La compagnie Profit Inc.	59, rue aaa	Montréal	Québec

No. facture	Date	No. client	No. produit	Qté commandée....
F9822	10 jan 07	100	PR8888	2

No. produit	Nom	Description	Prix
PR8888	Boîte d'enveloppes	9 X 11	4,99\$
PR 8877	Plume fontaine	Noire	39,95\$

On crée en fait une relation entre deux tables, à l'aide d'un champ commun à chacune d'elles. C'est pourquoi on définit Access comme un *système de gestion de bases de données relationnelles*. Lorsqu'une relation existe, l'information contenue dans chaque table devient disponible, alors les informations déjà inscrites dans une table n'ont pas à l'être dans une autre.

Développement d'une base de données répondant à vos besoins

Afin d'utiliser Access efficacement, il convient de structurer la démarche d'informatisation des données, en planifiant et en structurant toute l'information à inclure dans la base de données.

Le mot clé est *besoin*. Les besoins représentent le point de départ de la structure des données. Il importe de déterminer quels renseignements devront apparaître à l'écran, ainsi que les listes et les rapports qui devront être produits à partir de ces derniers.

Quelques règles de normalisation

Afin de structurer l'information, il faut procéder à quelques choix. Combien de tables y aura-t-il dans la base de données? Quel champ ira dans quelle table? Quelles seront les relations entre les tables?

Suivre certaines règles de normalisation permettra de trouver réponse à ces questions. L'application de ces règles aura pour résultat de structurer efficacement la base de données, mais au contraire, sa non-application pourrait entraîner plusieurs problèmes au moment de produire des requêtes et des rapports.

Première forme normale

Chaque champ doit être divisé en la plus petite parcelle d'information possible. En d'autres termes, chacune des colonnes d'une table doit être indivisible.

De plus, aucune colonne d'information ne doit être répétée dans une même table, ou d'une table à une autre, sauf celle qui servira de lien entre deux tables.

Deuxième forme normale

Une table est réputée suivre la deuxième forme normale lorsqu'elle suit la première forme normale et qu'en plus, tous les champs qui la composent dépendent entièrement de la clé primaire. Donc, une table de *deuxième forme normale* doit stocker des données dépendant ou traitant d'une seule entité décrite par la clé primaire.

Troisième forme normale

Une table est réputée suivre la troisième forme normale lorsqu'elle suit la deuxième forme normale et qu'en plus, tous les champs de cette table (sauf la clé primaire) sont indépendants les uns des autres. Prenons pour exemple une table qui contient un champ *PrixUnitaire* et un champ *Quantité*. On peut opter pour le calcul du prix unitaire multiplié par la quantité, puis stocker le résultat de ce calcul dans un champ *Total*. Cette table ne suivrait donc pas la troisième forme normale puisque le champ *Total* est dépendant des champs *Prix unitaire* et *Quantité*.

D'autres règles de normalisation existent, mais seules celles mentionnées ci-dessus seront abordées dans ce livre.