REPUBLIQUE TUNISIENNE
MINISTERE DE L'EDUCATION

BASES DE DONNÉES

4ème année de l'enseignement secondaire
Sciences de l'informatique

Auteurs

Ridha HADJ ALI
Inspecteur

Moncef GAFSI
Maître-Assistant

Abdelaziz ABDELLATIF
Maître-Assistant

Raouf ELYOUSFI
Professeur Principal

Evaluateurs

Mohamed ROMDHANI

Romdhane JALLOUL

Centre National Pédagogique

Préface

Conformément aux nouveaux programmes d'informatique en 4ème année de l'enseignement secondaire, nous vous proposons ce livre destiné à la section Sciences de l'Informatique et concernant la matière "Bases de données". Il est composé de quatre parties.

La première partie, intitulée introduction aux bases de données, vise à éveiller les apprenants à l'utilité et à l'intérêt qu'offre l'utilisation des bases de données, à définir leurs constituants ainsi que leurs systèmes de gestion. Cette partie servira à définir le jargon employé dans cet environnement, ainsi un certain nombre de définitions et d'exemples de Systèmes de Gestion de Bases de Données (SGBD) sont présentés.

L'étude du modèle relationnel est favorisée dans cette introduction et ce compte tenu de son niveau d'application actuel et de sa plus simple compréhension par nos élèves. Certaines étapes élémentaires, nécessaires à cette compréhension sont objet de l'étude dans cette partie comme les principaux constituants d'une BD.

Nous avons veillé à établir la relation entre l'étude des bases de données et celle exploitée dans la matière TIC concernant la connexion à une base de données en vu de concevoir des sites web dynamiques.

Dans l'objectif de mieux comprendre le fonctionnement et la manipulation d'une base de données et dans le souci de simplifier cette tâche à l'élève, nous occultons les étapes de normalisation ; par contre nous présentons une démarche simplifiée pour déduire la structure d'une base de données.

La deuxième partie traite de la structuration d'une base de données. Elle présente une démarche de détermination de la structure d'une base de données. L'objectif est double : utiliser les fonctions de base d'un SGBD pour créer, gérer, interroger une base de données puis charger des données et imprimer des états résultats de sélection dans les BD manipulées. Ces manipulations contribuent à l'apprentissage progressif au développement autour d'une BD. Une démarche de détermination de la structure d'une base de données est développée pour l'élève, elle l'accompagne pour concevoir cette dernière et aboutir à une solution normalisée. Dans cette partie, nous développons par l'occasion les manipulations sur la structure d'une Base de Données.

La troisième partie traite de la manipulation et la sécurisation des bases de données. L'objectif est de créer des applications de gestion des bases de données et l'exploitation de ces dernières. Cette manipulation sera effectuée selon le mode commande et le mode assisté. Le premier mode se base sur le standard SQL pour lequel nous présentons les principales commandes. Cette partie est à dominance pratique. Il est important d'évoquer la notion de sécurisation d'une BD, nous étudierons la notion de sécurité et les différentes méthodes pour l'assurer.

La quatrième partie traite de l'exploitation des bases de données. A travers un ensemble d'études de cas, nous accompagnons l'élève dans l'étude complète de l'implantation d'exemples de base de données et ce comme synthèse de ce qui a été étudié dans les parties qui précèdent. Un ensemble d'énoncés et de problèmes sont proposés à l'apprenant pour mettre en œuvre ses acquis en la matière.

Nous veillerons à accompagner les élèves dans l'apprentissage de la résolution d'un problème. Comme l'a été défini dans les années précédentes, nous continuons à adopter la même méthode de résolution dans ce livre. Cette démarche se base sur la décomposition d'un problème complexe en sous problèmes de difficultés moindres. Ce choix demeure dicté par un souci très important, celui de développer chez l'élève un esprit d'analyse et une méthodologie de résolution de problèmes. De plus, l'élève apprendra à traduire ses besoins en matière d'accès aux données (consultation et mise à jour) en formulant les commandes correspondantes et en se basant sur les structures adaptées.

Pour développer nos applications solutions et bénéficier de nouveaux services offerts par les nouvelles plateformes, nous utiliserons le langage SQL comme standard d'interrogation des BD.

Chaque chapitre est appuyé par des activités pour mieux faire comprendre les notions à présenter. Il se termine par une série d'exercices pris dans la plupart des cas du vécu de l'apprenant. Ces exercices sont ordonnés par ordre de difficulté croissant.

Nous donnons aussi la bibliographie référence de ce livre. Elle pourra vous guider pour d'autres lectures.

Nous espérons que ce livre vous apportera beaucoup d'aide et d'informations en matière de systèmes de bases de données. Nous vous serons bien reconnaissants de nous faire part de vos remarques et vos suggestions.

Les auteurs.

TABLE DES MATIERES

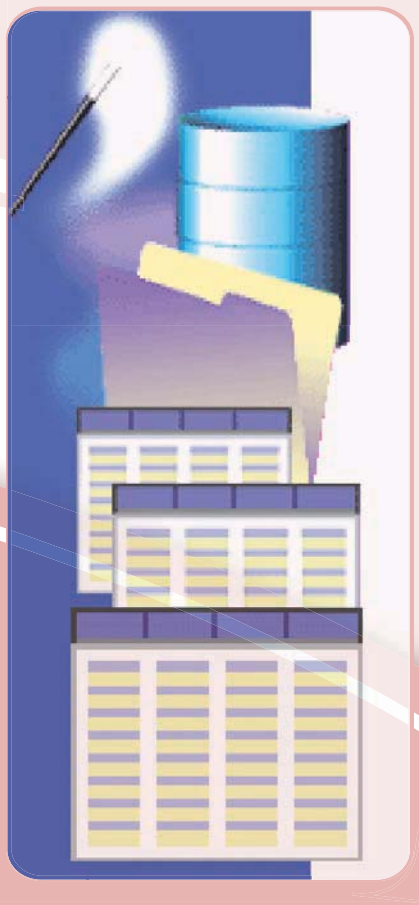
Partie III : Manipulation et Sécurisation de Bases de Données

E-mails des auteurs

Ridha HADJ ALI	**Moncef GAFSI**
Ridha.hajali@inbmi.edunet.tn	Moncef.gafsi@ensi.rnu.tn
Abdelaziz ABDELLATIF	**Raouf ELYOUSFI**
Abdelaziz.abdellatif@fst.rnu.tn	Raouf.elyousfi@edunet.tn

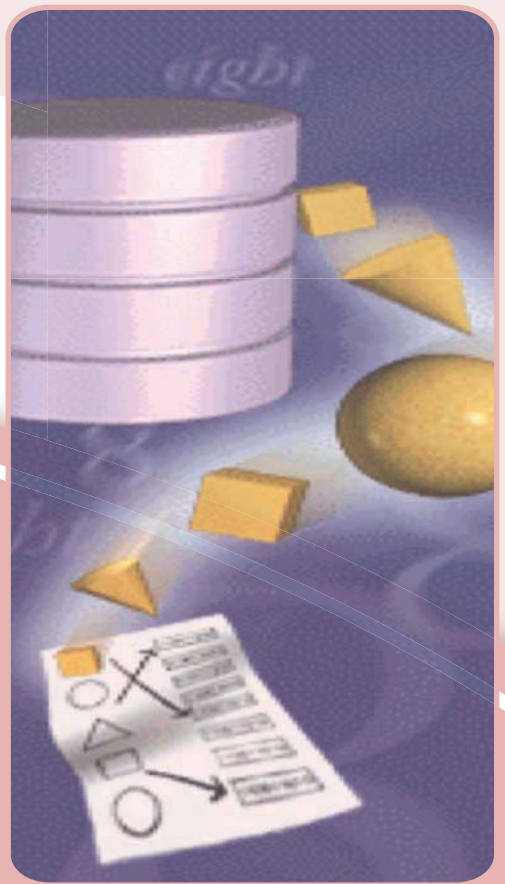
Partie 1

INTRODUCTION AUX BASES DE DONNÉES



Chapitre 1

Notion de Bases de Données



Objectifs :

- Décrire l'intérêt des bases de données et leurs systèmes de gestion
- Présenter l'environnement des bases de données

Plan :

1. Introduction à la gestion des données

1.1 Information et donnée
  1.1.1. Définitions
  1.1.2. Eléments constitutifs d'une information
1.2 La persistance
1.3 L'organisation papier
1.4 L'organisation en fichiers

2. Bases de données : notions de base

2.1 Définition d'une base de données
2.2 Intérêt de l'utilisation d'une base de données
2.3 Les modèles des bases de données
  2.3.1 Le modèle hiérarchique
  2.3.2 Le modèle réseau
  2.3.3 Le modèle relationnel

Retenons

Lecture

Exercices

1. Introduction à la gestion des données

1.1. Notion de donnée et d'information

Activité 1

Une ville dispose d'un service social dont l'une de ses fonctions est d'aider des citoyens à faibles revenus à obtenir des cartes de transport à tarif réduit.
Monsieur Amor Tounsi, retraité, se rend au service social afin d'obtenir une carte lui assurant une réduction des tarifs sur les lignes d'autobus de la société de transport de sa région.
Afin de lui ouvrir un dossier d'ayant droit, une employée du service social lui demande de fournir :

- ✔ une fiche d'état civil,
- ✔ une photo,
- ✔ un avis de non imposition,
- ✔ une quittance de la STEG ou de la SONEDE.

Au service social :

Une employée rassemble les documents fournis par M. Amor afin de constituer un dossier ;

- Elle vérifie l'adresse dans le fichier des résidants de la ville ;
- Elle retranscrit les informations (nom, prénom, adresse …) sur une fiche qui sera mise à jour après avis de la société de transport d'accorder ou de refuser la carte ;
- Elle adresse le dossier à la société de transport.

A la société de transport :

Mme Mansour, responsable du service des cartes de réduction, reçoit le dossier en question. Elle entre les renseignements dans l'ordinateur qui contient en mémoire toutes les informations concernant les demandes.
Les conditions requises pour bénéficier de la gratuité sont :

- ✔ le demandeur doit être résidant dans une commune agréée ;
- ✔ il doit être non imposable ;
- ✔ il doit avoir 65 ans au moins ou handicapé.

La carte de réduction des tarifs a une durée de validité d'une année.

Question 1 : Utiliser la liste de mots suivants afin de compléter les différentes phases pour l'attribution de la carte de gratuité : **Saisie - Contrôlent - Traitées - Collecte - mise à jour.**

1	Les deux employées ont besoin d'informations : - au service social : pour ouvrir un dossier de demande ; - à la société de transport : pour l'attribution de la carte ;	Elles procèdent à la ..... des informations.
2	Chacune d'elles vérifie l'exactitude des informations.	Elles ..... des informations.
3	- au service social : l'information est retranscrite sur une fiche - à la société de transport : elle est introduite sur ordinateur	Elles effectuent une ..... des informations.
4	Les informations externes (documents) sont comparées aux informations internes (conditions d'attribution).	les informations sont .....

Question 2 : En identifiant les moyens utilisés afin de mémoriser les informations au niveau du service social et de la société de transport, compléter la première ligne du tableau suivant :

Au service social : entrée des données sur	**A la société de transport :** entrée des données sur .....
FICHE DE DEMANDE DE CARTE DE REDUCTION **Nom :** Tounsi **Prénom :** Amor **Adresse :** 184, rue Hédi Chaker - Sousse **Décision :** Accordée - Refusée Période	

Question 3 : En identifiant les moyens utilisés afin de stocker les données relatives à plusieurs demandeurs de la carte de Tarif Réduit au niveau du service social et de la société de transport, compléter la première ligne du tableau suivant :

Au service social : classement dans l'ordre alphabétique dans un	**A la société de transport :** les données sont mémorisées et ordonnées dans un(e) .....
	**Carte de Tarif Réduit**

Code	Ville	Nom	Prénom	Adresse
D0001-2007				
D0001-2007				
D0001-2007				

1.1.1. Définitions

Avant d'introduire les concepts relatifs aux Bases de Données, il importe de définir la notion donnée. Plusieurs définitions se présent, parmi lesquelles :

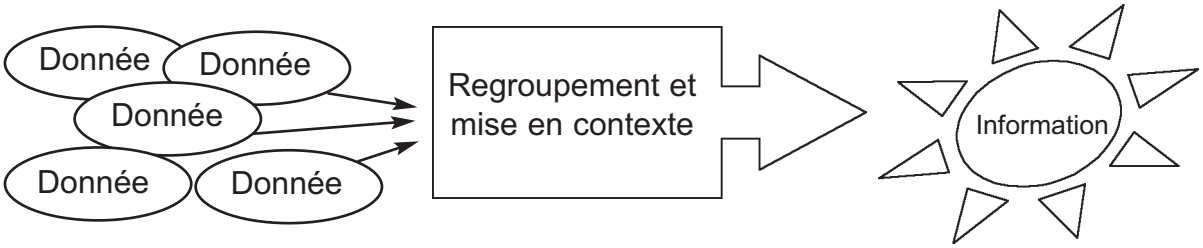
Définition 1:

D'après l'encyclopédie ***Encarta*** : *« Une donnée est une information représentée sous une forme conventionnelle, afin de pouvoir être traitée automatiquement ».*

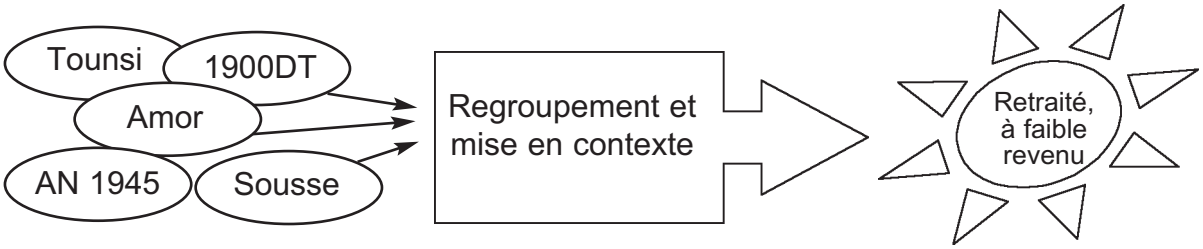
Remarque :

D'après l'encyclopédie libre ***Wikipédia*** : *«Dans les technologies de l'information (TI), une donnée est une description élémentaire, souvent codée, d'une chose, d'une transaction d'affaire, d'un événement, etc. Les données peuvent être conservées et classées sous différentes formes : papier, numérique, alphabétique, images, sons, etc. Le processus d'enregistrement des données dans une mémoire s'appelle la mémorisation ».*

Une donnée peut être considérée comme étant un élément fondamental sur lequel se bâtit un raisonnement, une recherche, une étude ou une œuvre. Ainsi, une donnée est rarement utilisée d'une manière isolée.
Une donnée, regroupée avec d'autres, rattachées à un contexte et éventuellement transformées donnent naissance à une **information.**



Exemple :



Remarque :

L'être humain, depuis son existence, ne cesse de manipuler une quantité plus ou moins importante de données dans sa vie courante (noms de personnes, noms de lieux, quantités, couleurs, poids, etc.).

Activité 2 :

Identifier quelques données utilisées dans les domaines suivants :

- Une administration d'un lycée,
- Un restaurant,
- Un magasin qui vend des articles de sport,
- Une banque.

1.1.2. Eléments constituant une information

Les données sont généralement regroupées selon leur appartenance à un objet du monde réel. L'ensemble d'objets homogènes (cohérents) constitue ce qui est couramment appelé **entité**.

Une entité est décrite par un ensemble d'attributs (ou propriétés). Chacun de ces attributs prend ses valeurs dans un ensemble appelé **Domaine**.

Parfois, l'information provient de deux ou plusieurs entités reliées entres elles par ce qui est couramment appelé **association**.

INFORMATION		
Entité(s)	**Attribut(s)**	**Valeur(s)**
C'est la personne, l'objet, le fait pris en considération. ↓ De qui ou de quoi s'agit-il?	Il est nécessaire pour décrire l'entité. ↓ Quelles sont les caractéristiques proposées pour décrire l'entité ?	C'est le contenu ou la valeur de l'attribut. ↓ Quelle est la valeur de chacune des attributs (caractéristiques ou propriété) ?
Exemple 1 : Fiche d'état civil		
Personne	Nom Prénom Date de naissance Lieu	Tounsi, Benzarti, Baklouti, Tboulbi Amor, Abdelaziz, Ridha, Moncef 20/09/1945, 11/11/1970, Sousse, Bizerte,
Exemple 2 : Carte de gratuité		
Bénéficiaire	N° d'identification Nom et Prénom Ville Validité (Date Limite)	2409 Tounsi Amor Sousse 31/12/2007
Exemple 3 : Un document scolaire		
Carte d'identité Scolaire		
Exemple 4 : Pièce d'Identité		
Carte d'Identité Nationale		

Application :

Compléter les deux derniers exemples du tableau précédent.

1.2. La persistance

Une donnée est rarement utilisée une seule fois. Généralement on a besoin de la mémoriser sur un support quelconque pour pouvoir la retrouver ultérieurement. Par exemple, lorsque quelqu'un vous communique son numéro de téléphone, vous le notez sur un bout de papier, sur votre carnet d'adresses, dans votre téléphone portable ou bien, si vous avez une bonne mémoire, dans votre cerveau. La façon de le mémoriser a une influence sur la rapidité de le retrouver lorsqu'on en a besoin.

Cette capacité de mémoriser et de pouvoir retrouver une donnée est appelée persistance. Son opposé est la volatilité. Lorsqu'une donnée n'est pas persistante, elle est volatile, c'est-à-dire qu'elle disparaît au bout d'un certain temps.

Persistance ≅ mémorisation + disponibilité

Dans le domaine de la programmation, la gestion de persistance des données se réfère au mécanisme responsable de la sauvegarde et la restauration de données, afin qu'un programme puisse se terminer sans que ses données ni son état d'exécution soient perdus.

Ces données peuvent être sauvegardées sur disque ou transmises à un serveur.

Lors de la mémorisation d'une donnée, deux aspects doivent être pris en considération :

Le type du support de mémorisation : Ce type doit être adapté à la durée de la mémorisation.

Exemples

Si je vais utiliser le numéro de téléphone immédiatement, je peux me contenter de ma mémoire. Si je veux l'utiliser dans une heure, je peux le noter sur un bout de papier. Si je vais l'utiliser souvent dans les jours et mois à venir, il vaut mieux le mémoriser dans mon carnet d'adresses ou dans la puce de mon téléphone portable.

Le format de mémorisation : La forme selon laquelle une donnée est mémorisée.

Exemples

Pour mémoriser le numéro de téléphone, si je note uniquement le numéro, je ne serais pas capable de savoir à qui il se réfère si je le retrouve dans un mois. Il me faut donc le décrire par exemple sous la forme suivante : « Monsieur X : 34217788».

La persistance des données peut être donc assurée grâce aux organisations suivantes :

Organisation papier : si la gestion des données n'est pas informatisée.

Organisation en fichiers : si la gestion des données est informatisée.

Nous allons développer ces deux types d'organisation dans les sections qui suivent.

1.2.1. L'organisation papier

L'organisation des données en papier consiste à utiliser différents supports papiers pour assurer la persistance. Ces supports peuvent être des fiches, des registres, des cahiers, etc. Cette organisation peut conduire à plusieurs problèmes parmi lesquels :

- *Classement :* un document ne peut être classé que sous une seule référence, limitant ainsi les possibilités de recherche.
- *Consultation :* le document est difficile d'accès, le délai de mise à disposition peut atteindre plusieurs jours ; en outre, pendant sa consultation par une personne, le document n'est plus disponible à d'autres personnes.
- *Contraintes :* de volume et de taille des documents.

- Délai : de recherche et de restitution mais aussi de conservation.
- Fiabilité : du classement et du reclassement en cas d'utilisation d'une archive.
- Sécurité : destruction ou détérioration (volontaire ou non), vol.
- Qualité de la restitution : elle peut être médiocre (photocopie, fax…)

1.2.2. L'organisation en fichiers

L'organisation des données en fichiers consiste à utiliser des supports informatiques pour assurer la persistance. Ces supports peuvent être des disques durs, des disquettes, des CD, des flashs disques, etc.

Un ***fichier*** (en anglais: ***file***) est un ensemble de données structurées mémorisées sur un support de stockage permanent.

Exemples :

Fichier des employés d'une entreprise, fichier des comptes des clients d'une banque, fichier des clients d'une entreprise de vente par correspondance, fichier d'abonnés (téléphone, électricité, ...), fichier des impôts, etc.

Il faut transférer les données de la mémoire de masse vers la mémoire centrale pour pouvoir la traiter. Par exemple, prenons les données relatives à un fichier des abonnés à une revue. Pour imprimer la lettre proposant à une personne de renouveler son abonnement, un programme va chercher l'information concernant cette personne dans le fichier (nom, adresse, date de fin d'abonnement, ...), l'amener en mémoire centrale, puis l'imprimer.

Cette organisation en fichiers possède les inconvénients suivants :

✔ *Lourdeur d'accès aux données*. En pratique, pour chaque accès, même le plus simples, il faudrait écrire un programme.

✔ *Manque de sécurité*. Si tout programmeur peut accéder directement aux fichiers, il est impossible de garantir la sécurité et l'intégrité des données de ce fichier.

✔ *Redondance de données*. Etant donné que les fichiers sont généralement conçus par des équipes différentes, il y a un risque qu'un même ensemble de données figurent dans deux ou plusieurs fichiers : c'est **la redondance**. En plus du gaspillage de l'espace disque, il y a un risque d'incohérence dans le cas ou de la mise à jour n'a pas touché une la totalité des copies.

Le fichier des employés de telle entreprise contient un enregistrement par employé et le fichier des étudiants de l'université contient un enregistrement par étudiant, ...

Un ***enregistrement*** (article, record) est un élément d'un fichier. Il constitue :

- l'unité logique de transfert entre la mémoire centrale et la mémoire de masse
- l'unité de traitement des programmes exploitant ce fichier.

Limites à l'utilisation des fichiers

L'utilisation de fichiers impose d'une part, à l'utilisateur de connaître l'organisation de chaque fichier qu'il utilise afin de pouvoir accéder aux informations dont il a besoin et, d'autre part, d'écrire des programmes pour pouvoir effectivement manipuler ces informations. Pour des applications nouvelles, l'utilisateur devra obligatoirement écrire de nouveaux programmes et il pourra être amené à créer de nouveaux fichiers qui contiendront peut-être des informations déjà présentes dans d'autres fichiers.

De telles applications sont : rigides, contraignantes, longues et coûteuses à mettre en œuvre.

Les données associées sont : mal définies et mal désignées, redondantes, peu accessibles de manière ponctuelle et peu fiables.
La prise de décision est une part importante de la vie d'une société. cette dernière doit être bien informée sur la situation ce qui nécessite alors des informations à jour et immédiatement disponibles.
Les utilisateurs, quant à eux, ne veulent plus de systèmes d'information constitués d'un ensemble de programmes inflexibles et de données inaccessibles à tout non spécialiste; ils souhaitent avoir des systèmes d'informations globaux, cohérents, directement accessibles (sans qu'ils aient besoin d'écrire des programmes ou de demander à un programmeur de les écrire pour eux) et des réponses immédiates aux questions qu'ils posent. On a donc recherché des solutions tenant compte à la fois des désirs des utilisateurs et des progrès techniques. Cette recherche a abouti au **concept de base de données**.

Activité 3 :

Un libraire qui dispose d'un ensemble d'ouvrages (dont le nombre est variable dans le temps), désire gérer, de préférence par ordinateur, le rayon des livres de sa bibliothèque. Il adresse sa demande à plusieurs intervenants en insistant sur les tâches suivantes :

- ✔ L'ajout d'un livre
- ✔ La suppression d'un livre
- ✔ La modification du numéro de téléphone de l'éditeur d'un livre
- ✔ La recherche d'un ensemble de livres selon un critère
- ✔ Le contrôle de certains accès aux données (sur les noms, le nombre d'exemplaires, le code etc.)

Les informations disponibles sur les livres sont les suivantes : code, titre, nombre d'exemplaires, année d'édition, éditeur, téléphone de l'éditeur, nom de l'auteur et prénom de l'auteur.

Le 1er intervenant propose l'utilisation d'une fiche (papier)
Il envisage utiliser une fiche décrivant chaque livre et un journal gardant une trace des opérations effectuées sur ces livres.
Se passant de l'ordinateur, l'intervenant avoue rencontrer des opérations lentes et des difficultés majeures.

Question : Développer avec votre enseignant les limites et les difficultés rencontrées.

Commentaire sur cette solution :

- Pour ajouter, modifier ou supprimer une information, on doit réécrire toute la fiche.

Le 2ème intervenant propose d'écrire un programme (en **TPascal** par exemple) en utilisant une structure de données de type tableau à une dimension.
Il organise les données dans des tableaux contenant les informations suivantes : Code, Titre, NbExem, Année, Editeur, Tel_Editeur, NomAuteur et PrénomAuteur.

Code	Titre	Nb Exem	Année	Editeur	Tel-Editeur	Nom Auteur	Prénom Auteur
12TA1	Réseaux informatiques	10	1998	Eyrolles	1111111111	Tanenbaum	Henri
13GO1	Algorithmes génétiques	5	1994	Addison Wesley	222222222	Goldberg	Stephen
13GO1	Algorithmes génétiques	5	1994	Addison Wesley	222222222	Holland	John
15TA2	Système d'exploitation	6	1993	Eyrolles	1111111111	Cardy	Ronald
15TA2	Système d'exploitation	6	1993	Eyrolles	444444444	Dumar	Eric
15TA2	Système d'exploitation	6	1993	Eyrolles	1111111111	Tanenboum	Henri

Après réflexion, l'intervenant avoue trouver des limites à son travail.
Question : Développer avec votre enseignant les limites et les difficultés rencontrées.

Le 3ème intervenant propose l'utilisation d'une matrice de données qu'il programme en Pascal.
Il envisage écrire un programme Pascal pour cette gestion, il organise les données dans une matrice où chaque colonne représente un type d'informations.
Après réflexion, il avoue aussi trouver des contraintes à son travail.
Question : Développer avec votre enseignant les limites, les contraintes et les difficultés rencontrées.

Commentaires sur cette proposition :

- ✔ Le nombre de livre est limité par la taille du tableau.
- ✔ Les données sont volatiles. Pour effectuer une recherche, on doit saisir de nouveau les mêmes informations.

Le 4ème intervenant propose l'utilisation d'un logiciel de traitement de texte.
Il envisage saisir les données dans un tableau inséré dans un document texte. Il utilise une fiche pour chaque livre et un journal d'opérations sur l'ensemble des livres.
Il avoue rencontrer d'énormes difficultés à poursuivre le travail et particulièrement dans les recherches des lignes à critères multiples.
Question : Développer avec votre enseignant les limites et les difficultés rencontrées

Commentaires sur cette proposition :

- ✔ Redondance de données : les informations code, titre, …, se répètent pour chacun de ses auteurs.
- ✔ Anomalie de suppression : en supprimant tous les livres d'un éditeur donné, toutes les informations concernant cet éditeur disparaissent avec le dernier enregistrement supprimé. En plus de la suppression des livres, on a finalement supprimé un éditeur.
- ✔ Anomalie d'ajout : si on veut ajouter un nouvel éditeur, il faut nécessairement introduire un livre (code, titre, NbExem, Année, etc.).

✔ Anomalie de mise à jour : en oubliant une ligne lors de la modification d'une valeur, on introduit des contradictions dans la table d'où l'incohérence de nos données.

✔ Anomalie de recherche : dans le cas où que le nombre de lignes est assez élevé, il n'est pas du tout simple de trouver la liste des livres d'un éditeur donné.

Le 5ème intervenant propose l'utilisation d'un tableur.
Il envisage saisir les données dans un tableau à l'aide d'un tableur; Il réserve une ligne pour chaque livre et une colonne pour chaque type de donnée.

Exemple de proposition :

	A	B	C	D	E	F	G
1	**Code**	**Titre**	**Année**	**Editeur**	**Tel-Editeur**	**Nom Auteur**	**Prénom Auteur**
2	12TA1	Réseaux informatiques	1998	Eyrolles	1111111111	Tanenbaum	Henri
3	13GO1	Algorithmes génétiques	1994	Addison Wesley	222222222	Goldberg	Stephen
4	13GO1	Algorithmes génétiques	1994	Addison Wesley	222222222	Holland	John
5	15TA2	Système d'exploitation	1993	Eyrolles	1111111111	Cardy	Ronald
6	15TA2	Système d'exploitation	1993	Eyrolles	444444444	Dumar	Eric
7	15TA2	Système d'exploitation	1993	Eyrolles	1111111111	Tanenboum	Henri

Pouvant répondre à une bonne partie de la demande, l'intervenant avoue, cependant, rencontrer des difficultés à poursuivre le travail et particulièrement dans les recherches multicritères avec le grand nombre d'ouvrages qui peut être manipulé.

Question : Développer avec votre enseignant les limites et les difficultés rencontrées.

Activité Complémentaire :

Déduire les caractéristiques d'une bonne structure de données qui vous évite les inconvénients et résout les difficultés rencontrées dans l'activité précédente.

Conclusion : Il faut alors structurer les données sous une forme qui ne présente pas ces insuffisances.

Dans la suite, on étudiera une approche de structuration des données.

2. Bases de données : notions de base

Activité 4

Cette activité présente les données manipulées par une agence de location de voitures organisées sous forme de fiches :

LOCACAR : AGENCE DE LOCATION DE VOITURE

FICHE D'IDENTIFICATION

Immatriculation : 125 TN 98
Modèle : SAFRANE
Catégorie : C
Date de révision : 03/07/2002
Km à la révision : 4802 km
Km actuels : 9641 km
Disponibilité : N (non)

FICHE D'IDENTIFICATION

Immatriculation : 237 TN 98
Modèle : CLIO
Catégorie : A
Date de révision : 17/10/2002
Km à la révision : 5069 km
Km actuels : 8882 km
Disponibilité : O (oui)

FICHE D'IDENTIFICATION

Immatriculation : 880 TN 98
Modèle : MEGANE
Catégorie : B
Date de révision : 21/08/2002
Km à la révision : 7220 km
Km actuels : 14815 km
Disponibilité : O (oui)

1 - Questions :

- Quelles sont les données nécessaires au vendeur, afin de gérer le parc automobile de l'agence ?
- Comment trouve-t-on la demande du client par rapport aux données des fiches d'immatriculation des voitures ?
- Comment le vendeur a traduit (expliqué) la demande du client ?

2 - Les fiches des descriptions des véhicules sont présentées de la manière suivante : il s'agit d'un tableau.

Recopier les trois premières lignes du tableau et compléter les noms des colonnes (Identifier les intitulés des colonnes) :

.....	.....	.....	.....	.....	.....	.....
125 TN 98	SAFRANE	C	03/07/02	4802	9641	N
880 TN 99	206	A	15/10/02	4998	6250	O
880 TN 98	206	A	23/08/02	1596	3721	O
880 TN 98	605	C	25/09/02	4871	7237	N
880 TN 98	605	C			49	O
237 TN 98	CLIO	A	17/10/02	5069	8882	O
238 TN 98	CLIO	A	02/10/02	5197	9523	N
880 TN 98	CORSA	A	18/10/02	1666	1815	N
880 TN 98	307	B	16/09/02	5285	7965	O
880 TN 98	307	B	01/09/02	14988	17661	O
880 TN 98	307	B			990	N
880 TN 98	MEGANE	B	21/08/02	10220	14815	O

Ce tableau est appelé : .....

L'agence utilise d'autres tableaux comme ceux des clients et des opérations de location. Discuter avec votre enseignant les éventuelles formes de ces tableaux.

2.1. Définition d'une base de données

2.1.1. Définition 1

Une base de données est un ensemble de données, sur un sujet, qui sont : exhaustives, non redondantes, structurées et persistantes.

2.1.2. Définition 2

Une base de données peut être définie comme une collection de données structurées modélisant un univers donné (monde réel), et mémorisée sur un support permanent. En se basant sur les définitions précédentes, on peut retenir comme définition d'une base de données :

Définition

Une base de données est une collection de données structurées relatives à un ou plusieurs domaines du monde réel.

Exemples 1

Une Base de Données « Étudiants » regroupe toutes les données concernant les étudiants (num, nom, prénom, adresse, modules auxquels est inscrit l'étudiant, notes, etc.) et servira à toutes les applications.

Exemples 2

Une base de données «Bibliothèque» regroupe les données concernant les ouvrages, les adhérents qui empruntent les ouvrages, etc. De plus, il y a des règles de fonctionnement comme :

- Un adhérent ne peut emprunter en même temps plus de 5 ouvrages.
- La durée maximale d'emprunt est limitée à 10 jours.
- Seul le bibliothécaire en chef peut déclarer qu'un ouvrage est perdu.
- Les ouvrages ne sont pas rangés de manière aléatoire, mais bien en fonction de certains critères préétablis.

Lexique

Français	Anglais	Arabe	Synonymes
Base de données	Data base	قاعدة بيانات	BD

Dans une architecture client/serveur, une base de données est considérée comme une ressource partagée par un ensemble d'applications situées sur les postes clients. La machine qui gère cette base de données est appelée «Serveur de données» ou bien « Serveur» tout simplement.

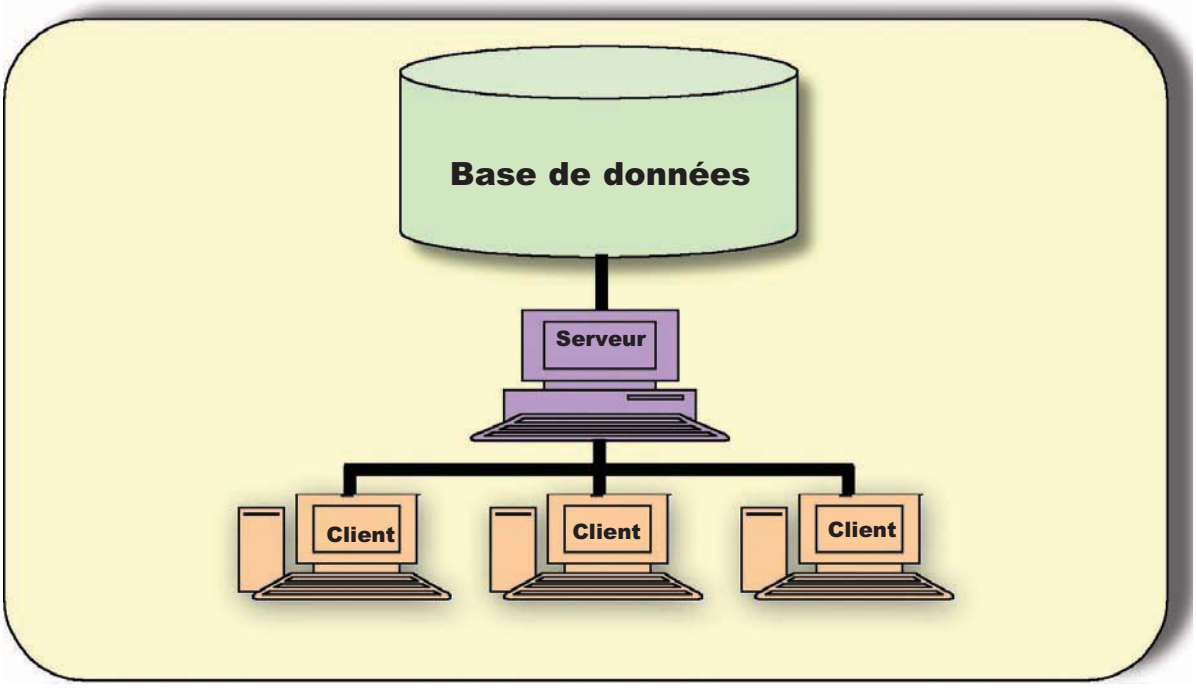


Figure 1.1 : *Une base de données dans une architecture client/serveur.*

2.2. Intérêts de l'utilisation des bases de données

L'organisation des données sous forme de bases de données présente les avantages suivants :

Centralisation : Les données sont regroupées dans une structure centrale permettant d'éviter la redondance. En effet, les données sont saisies une seule fois et peuvent être utilisées par plusieurs programmes et plusieurs utilisateurs.

Indépendance entre données et programmes : Traditionnellement, dans des langages de programmation tels que Cobol, Pascal, les fichiers sont décrits dans

les programmes qui les exploitent. Dans le cas d'une Base de données, les données sont décrites indépendamment des programmes.

Intégration des liaisons entre les données : Les relations entre les données font partie de la base de données et non pas des programmes comme dans le cas où les données sont organisées en fichiers.

Exemples

Dans notre bibliothèque, on a des adhérents et des ouvrages. On devra pouvoir par exemple savoir quel adhérent a emprunté tel ouvrage ou quels sont les ouvrages empruntés par tel adhérents.

Intégrité de données : Il s'agit d'une propriété fondamentale de la base. Elle se traduit par un ensemble de règles (unicité, référence et valeur) permettant d'assurer la cohérence des données.

Les données stockées dans la base doivent représenter exactement ce qui se produit dans la réalité. Une base de données qui n'est pas fidèle à la réalité est tout simplement inutilisable.

Exemples

Aucun adhérent ne doit posséder plus de 5 ouvrages : avoir plus c'est contraire aux règles de fonctionnement de la bibliothèque. A un même moment, un exemplaire d'un ouvrage ne peut être emprunté que par un seul adhérent.

Les données doivent respecter des règles (contraintes d'intégrité) afin d'éviter par exemple, que deux personnes possèdent un même Numéro de la carte d'identité.

Partage des données (ou concurrence d'accès) : Des utilisateurs différents peuvent accéder en même temps aux mêmes données, le système doit régler la concurrence d'accès en ordonnançant les demandes.

Exemples

Un adhérent doit pouvoir rechercher la liste des livres traitant d'un sujet donné pendant que le bibliothécaire enregistre l'achat d'un nouveau livre ou qu'une troisième personne tire une liste des rappels.

Chaque utilisateur a l'impression qu'il est le seul utilisateur de la base.

2.3. Les modèles des bases de données

Les bases de données sont apparues à la fin des années 60, à une époque où la nécessité d'un système de gestion de données souple se faisait ressentir pour éviter les inconvénients de l'organisation en fichiers.

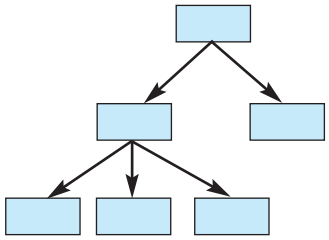
Nous avons vu qu'une base de données représente à la fois les données décrivant les objets du monde réel et les liens (ou associations) existant entre ces objets.

Depuis l'apparition des bases de données, quatre modèles se sont succédés pour permettre la structuration des données :

1. Modèle hiérarchique,
2. Modèle réseau,
3. Modèle relationnel,
4. Modèle orienté objet.

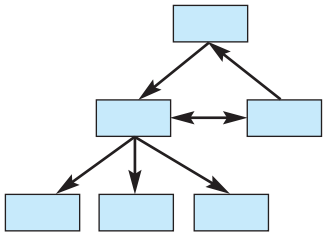
Ces quatre modèles se distinguent par la façon selon laquelle les liens entre les données sont représentés.

2.3.1. Le modèle hiérarchique



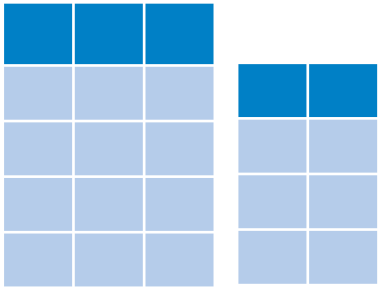
Le seul type de lien possible dans ce modèle est le lien père-fils (ou 1:n). Une base de données se présente ainsi comme un arbre ordonné dont les sommets sont les objets et les arcs sont les liens. Un objet peut avoir plusieurs fils, mais un fils ne peut avoir qu'un seul père.

2.3.2. Le modèle réseau



Une base de données réseau se présente comme un graphe, ou réseau (d'où le nom du modèle). Dans ce graphe les objets sont reliés entre eux à l'aide de pointeurs logiques. Tous les types de liens sont possibles, notamment les liens (n:m).

2.3.3. Le modèle relationnel



Le principe de base du modèle relationnel consiste à représenter aussi bien les objets que les liens à l'aide d'une structure appelé table. Une table est une structure tabulaire dont les colonnes, appelées aussi attributs, correspondent aux caractéristiques de l'objet ou de l'association à représenter et les lignes, appelées aussi tuples ou n-uplets, correspondent aux occurrences.

Ce modèle est aujourd'hui celui le plus utilisé.
Une présentation plus détaillée est faite dans le chapitre 3.

RETENONS

✔ Une base de données est un ensemble structuré de données relatives à un ou plusieurs domaines. Elle est exhaustive, non redondante, structurée et persistante

✔ L'utilisation des bases de données permet de centraliser l'information, d'assurer l'indépendance entre les données et les programmes, d'établir des liaisons entre les entités et de partager les données.

✔ Il existe quatre modèles de base de données : modèle hiérarchique, modèle réseau, modèle orienté objet et modèle relationnel

✔ Le modèle relationnel est, aujourd'hui, le plus utilisé

LECTURE

A Short Database History

Ancient to modern : The origins go back to libraries, governmental, business, and medical records. There is a very long history of information storage, indexing, and retrieval. Don't ignore this history, there is usually something to learn from these folks and their success and failure. Lots of online stuff (and there is lots) does not guarantee quality of data or search technique. Good design principles goes way back and lots is known now about how to make good designs that lead to better reliability and performance.

1960's : Computers become cost effective for private companies along with increasing storage capability of computers. Two main data models were developed: network model (CODASYL) and hierarchical (IMS). Access to database is through low-level pointer operations linking records. Storage details depended on the type of data to be stored. Thus adding an extra field to your database requires rewriting the underlying access/modification scheme. Emphasis was on records to be processed, not overall structure of the system. A user would need to know the physical structure of the database in order to query for information. One major commercial success was SABRE system from IBM and American Airlines.

1970-72 : E.F. Codd proposed relational model for databases in a landmark paper on how to think about databases. He disconnects the schema (logical organization) of a database from the physical storage methods. This system has been standard ever since.

1970's : Several camps of proponents argue about merits of these competing systems while the theory of databases leads to mainstream research projects. Two main prototypes for relational systems were developed during 1974-77. These provide nice example of how theory leads to best practice.

Ingres: Developed at UCB. This ultimately led to Ingres Corp., Sybase, MS SQL Server, Britton-Lee, Wang's PACE. This system used QUEL as query language.

System R: Developed at IBM San Jose and led to IBM's SQL/DS & DB2, Oracle, HP's Allbase, Tandem's Non-Stop SQL. This system used SEQUEL as query language.

The term Relational Database Management System (RDBMS) is coined during this period.

1976 : P. Chen proposed the Entity-Relationship (ER) model for database design giving yet another important insight into conceptual data models. Such higher level modeling allows the designer to concentrate on the use of data instead of logical table structure.

Early 1980's : Commercialization of relational systems begins as a boom in computer purchasing fuels DB market for business.

Mid-1980's : SQL (Structured Query Language) becomes "intergalactic standard". DB2 becomes IBM's flagship product. Network and hierarchical models fade into the background, with essentially no development of these systems today but some legacy systems are still in use. Development of the IBM PC gives rise to many DB companies and products such as RIM, RBASE 5000, PARADOX, OS/2 Database Manager, Dbase III, IV (later FoxBASE, even later Visual FoxPro), Watcom SQL.

Early 1990's : An industry shakeout begins with fewer surviving companies offering increasingly complex products at higher prices. Much development during this period centers on client tools for application development such as PowerBuilder (Sybase), Oracle Developer, VB (Microsoft), etc. Client-server model for computing becomes the norm for future business decisions. Development of personal productivity tools such as Excel/Access (MS) and ODBC. This also marks the beginning of Object Database Management Systems (ODBMS) prototypes.

Mid-1990's : Kaboom! The usable Internet/WWW appears. A mad scramble ensues to allow remote access to computer systems with legacy data. Client-server frenzy reaches the desktop of average users with little patience for complexity while Web/DB grows exponentially.

Late-1990's : The large investment in Internet companies fuels tools market boom for Web/Internet/DB connectors. Active Server Pages, Front Page, Java Servlets, JDBC, Enterprise Java Beans, ColdFusion, Dream Weaver, Oracle Developer 2000, etc are examples of such offerings. Open source solution come online with widespread use of gcc, cgi, Apache, MySQL, etc. Online Transaction processing (OLTP) and online analytic processing (OLAP) comes of age with many merchants using point-of-sale (POS) technology on a daily basis.

Early 21st century : Decline of the Internet industry as a whole but solid growth of DB applications continues. More interactive applications appear with use of PDAs, POS transactions, consolidation of vendors, etc. Three main (western) companies predominate in the large DB market: IBM (buys Informix), Microsoft, and Oracle.

Future trends : Huge (terabyte) systems are appearing and will require novel means of handling and analyzing data. Large science databases such as genome project, geological, national security, and space exploration data. Clickstream analysis is happening now. Data mining, data warehousing, data marts are a commonly used technique today. More of this in the future without a doubt. Smart/personalized shopping using purchase history, time of day, etc.

Successors to SQL (and perhaps RDBMS) will be emerging in the future. Most attempts to standardize SQL successors has not been successful. SQL92, SQL2, SQL3 are still underpowered and more extensions are hard to agree upon. Most likely this will be overtaken by XML and other emerging techniques. XML with Java for databases is the current poster child of the "next great thing". Check in tomorrow to see what else is news.

Mobile database use is a product now coming to market in various ways. Distributed transaction processing is becoming the norm for business planning in many arenas.

Probably there will be a continuing shakeout in the RDBMS market. Linux with Apache supporting mySQL (or even Oracle) on relatively cheap hardware is a major threat to high cost legacy systems of Oracle and DB2 so these have begun pre-emptive projects to hold onto their customers.

Object Oriented Everything, including databases, seems to be always on the verge to sweeping everything before it. Object Database Management Group (ODMG) standards are proposed and accepted and maybe something comes from that.
Ethical/security/use issues tend to be diminished at times but always come back. Should you be able to consult a database of the medical records/genetic makeup of a prospective employee ? Should you be able to screen a prospective partner/lover for genetic diseases? Should amazon.com keep track of your book purchasing ? Should there be a national database of convicted sex offenders/violent criminals/drug traffickers? Who is allowed to do Web tracking ? How many times in the last six months did you visit a particular sex chat room/porn site/political satire site? Who should be able to keep or view such data? Who makes these decisions ?

http://math.hws.edu/vaughn/cpsc/343/2003/history.html

EXERCICES

Exercice 1 :

Un de vos amis a fait une demande de bourse d'étude ; il a du remplir ce document :

Renseignements concernant le demandeur (père, mère, tuteur ou représentant légal du candidat boursier)	
Nom : **ZAHOUANI**	Prénom : **MOHAMED**
Date et lieu de naissance : **10 Juin 1949 à Sousse.**	
Nationalité : **Tunisienne**	N° C.I.N. : **05528077**
Profession : **Professeur**	Profession du conjoint : **Sans**
Nombre d'enfants : **2** dont **2** à charge.	
Adresse : **Avenue de la république - 4000 - Sousse.**	

Précisez, par une croix dans la colonne concernée, les caractéristiques des informations ci-dessus :

INFORMATIONS							
	quantitative	**qualitative**	**alphabétique**	**alphanumérique**	**numérique**	**permanente**	**temporaire**
Nom :							
Date de naissance :							
N° C.I.N. :							
Nombre d'enfants :							
Adresse :							

Exercice 2 :

Citer et expliquer trois inconvénients liés à l'utilisation des fichiers.

Exercice 3 :

Proposer des exemples où l'utilisation d'une base de données pourrait être plus efficace que celle de fiches papiers

Exercice 4 :

En se basant sur l'exercice 1, indiquez, dans le tableau ci-dessous, la caractéristique de chaque donnée entourée sur l'entête de ce bulletin de paye :

BULLETIN DE PAYE

Janvier 2006

Employeur	**Salarié**
Entreprise TrucMuche Zone Industrielle - 1002 - Tunis Tél. : 71 123 321	Nom : Salah Hmida Adresse : Tunisie N° CNSS : 123456789987654321 - S

N°	Désignation de la donnée	Caractéristique
1		
2		
3		

Exercice 5 :

Compléter les vides en utilisant la liste suivante :

Redondance - données - structurée - programmes - base - informations

Une base de données est une entité dans laquelle il est possible de stocker des données de façon ..... et avec le moins de ..... possible. Ces ..... doivent pouvoir être utilisées par des ....., par des utilisateurs différents. Ainsi, la notion de base de données est généralement couplée à celle de réseau, afin de pouvoir mettre en commun ces ....., d'où le nom de ..... On parle généralement de système d'information pour désigner toute la structure regroupant les moyens mis en place pour pouvoir partager des données.

Exercice 6 :

Relier par une flèche :

Indépendance entre données et programmes	●	●	ensemble de règles (unicité, référence et valeur) permettant d'assurer la cohérence des données
Centralisation de l'information	●	●	les données sont décrites indépendamment des programmes
Intégrité	●	●	les données sont saisies une seule fois et peuvent être utilisées par plusieurs programmes et plusieurs utilisateurs

Exercice 7 :

Répondre par vrai V (Vrai) ou Faux (F) pour chacune des propositions suivantes :

1- Une base de données, c'est :

- ☐ Un ensemble organisé et structuré d'informations portant sur le même thème.
- ☐ Un plan de classement de l'information
- ☐ Un serveur Internet

2- Une base de données garde les informations d'une façon :

☐ volatile ☐ permanente ☐ temporaire

3- Lorsque j'utilise une base de données, je manipule :

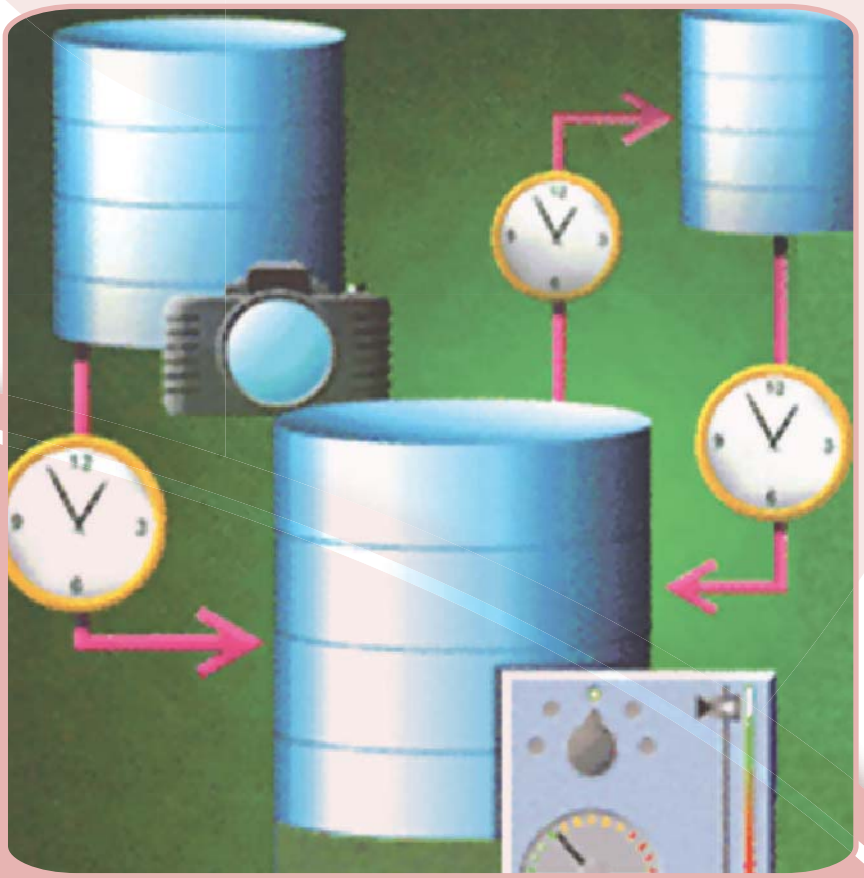
☐ des feuilles de calcul ☐ des pages Web ☐ des diapositives

4- Dans une base de données relationnelle, les données sont :

- ☐ représentées par des objets.
- ☐ reliées par des liens de hiérarchie.
- ☐ regroupées sous formes de tables.

Chapitre 2

Notion de Système de Gestion de Bases de Données



Objectifs :

- Découvrir les principales fonctions d'un SGBD
- Positionner les SGBDR dans le développement des bases de données
- Identifier les principaux SGBD et comparer leur étendus

Plan :

1. Introduction

Nous avons vu dans le chapitre précédent qu'une base de données permet de manipuler un volume important de données d'une manière simple, fiable et plus sécurisée que les approches traditionnelles. Afin de pouvoir contrôler ces données ainsi que les utilisateurs, on a eu recours à un logiciel chargé de gérer les données de la base, de prendre en charge les fonctionnalités de protection et de sécurité et de fournir les différents types d'interface nécessaires l'accès aux données. Ce logiciel s'appelle un **SGBD : Système de Gestion de Base de Données**

2. Définition d'un système de gestion de bases de données

Définition 1

Un système de gestion de base de données (SGBD) est un logiciel qui permet de : décrire, modifier, interroger et administrer les données d'une base de données.

Remarque

Un système de gestion de base de données (SGBD) permet à l'utilisateur de manipuler une ou plusieurs bases de données dans des termes abstraits, sans tenir compte de la façon dont l'ordinateur les maintient.

Un SGBD est constitué de deux composantes principales : un moteur et une interface.

- Le moteur constitue la composante principale d'un SGBD. Il assure un ensemble de fonctions qui seront détaillées dans la section suivante.
- L'interface, située entre les utilisateurs d'une base de données et le moteur, permet un accès facile et convivial aux données. Elle permet aussi d'effectuer les tâches globales sur la base de données telle que la sauvegarde, la *restauration*, etc

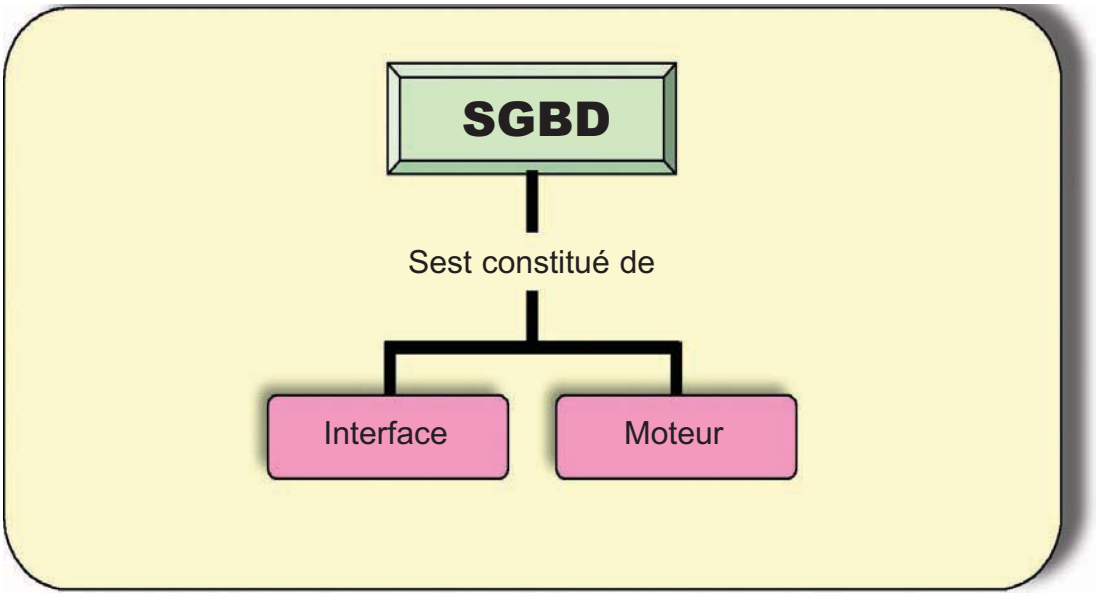


Figure 2.1 : *Les composants d'un SGBD.*

Lexique

Français	Anglais	Arabe	Synonymes
Système de Gestion de Bases de Données	Data bases Management System	نظام التصرف في قواعد البيانات	SGBD

3. Les fonctions d'un système de gestion de bases de données

Le SGBD est le logiciel responsable de gérer à tous les niveaux toutes les structures se trouvant dans une base de données. Cette gestion intègre les fonctions suivantes :

- La définition des données
- La manipulation des données
- L'intégrité des données
- La gestion des accès concurrents
- La confidentialité
- La sécurité du fonctionnement

Les paragraphes suivants détaillent chacune de ces fonctions. Pour illustrer ces fonctionnalités, on prendra l'exemple d'une base de données permettant la gestion des élèves dans un lycée. On suppose que cette base contient au moins les entités Elève, Section et Classe.

3.1. La définition des données

Le SGBD doit offrir à l'utilisateur des moyens pour décrire des «objets» qui vont constituer la base de données (comme des personnes, des voitures,...), leurs attributs (comme le nom des personnes, le type des voitures,...), leurs liens (comme « une personne *possède* des voitures») ainsi que des contraintes éventuelles pouvant concerner ces objets, leurs attributs ou leurs liens.

Ces moyens constituent ce que l'on appelle généralement le Langage de Description de Données (ou LDD). Le schéma d'une base est la description à l'aide du LDD des objets de la base, de leurs liens et des contraintes associées. Cette description est unique et commune aux diverses applications qui utilisent la base. Parfois, seule une partie de ce schéma, appelée sous-schéma, peut être connue d'un programme ou d'un groupe de programmes. En d'autres termes, le programme (ou le groupe de programmes) n'a accès qu'aux seuls objets présents dans le sous-schéma.

Exemple

La création des trois entités : Elève, Section et Classe.

3.2. La manipulation de données

La manipulation de données concerne les outils et les mécanismes qui permettent de manipuler le contenu d'une base de données par les utilisateurs.

Les SGBD offrent, souvent sous diverses formes, des capacités de recherche, de création, de modification et de suppression d'informations. Une de ces formes est le Langage de Manipulation de Données (LMD) qui permet de manipuler les données de la base de manière interactive.

L'action à effectuer sur la base est exprimée comme une phrase de ce langage (requête) qui est évaluée et exécutée par le SGBD.
Une autre forme est constituée par les interfaces que peut offrir le SGBD avec des langages de programmation tels Pascal, C, C++, Java ou Cobol.
Il faut souligner que ces divers moyens de manipulation des données d'une base cachent totalement à l'utilisateur tous les aspects relatifs à l'organisation physique des données sur le support de stockage.
Enfin, de plus en plus de SGBD offrent divers utilitaires pour l'aide à l'écriture de requêtes ou de programmes, à l'élaboration d'écrans de saisie, d'états de sortie, etc. Toutes ces facilités sont regroupées sous le vocable *d'environnement ou langage de quatrième génération (L4G).*

Exemple

- L'insertion d'une nouvelle section (Création d'une nouvelle section comme Sciences Informatique suite à une réforme du système éducatif)
- La modification de l'adresse d'un élève (suite à un déménagement)
- La suppression d'une classe (suite à une réduction des effectifs des élèves)
- La recherche de l'adresse d'un élève (lui envoyer une correspondance)

3.3. L'intégrité des données

Le concept d'intégrité des données est relatif à la qualité de l'information enregistrée. Pour être fiable, celle-ci doit parfois vérifier certaines propriétés, comme l'appartenance à une liste de valeurs permises pour un attribut. Ces propriétés sont appelées *contraintes d'intégrité*. Certaines sont spécifiées lors de la définition du schéma de la base, le SGBD se chargeant de les préserver pendant toute la vie de la base, alors que d'autres, plus complexes, peuvent nécessiter un effort de programmation.

Exemple

- Lors de l'inscription d'un élève, on doit vérifier l'existence de la section ainsi que d'autres contraintes comme l'âge et le niveau d'études

3.4. La gestion des accès concurrents

Pour que plusieurs utilisateurs puissent avoir accès simultanément aux données d'une base, le SGBD doit offrir des mécanismes de gestion des conflits d'accès. Ceux-ci sont similaires à ceux rencontrés dans les systèmes d'exploitation (autorisation des accès multiples en consultation, verrouillage lors d'accès en modification,…).

Exemple

- Lors de la mise à jour des données relatives à un élève, le SGBD verrouille (interdire la modification) ses données pour empêcher d'autres utilisateurs de les modifier. Cependant, ces données peuvent être consultées simultanément par d'autres utilisateurs autorisés.

3.5. La confidentialité

La mise en commun des données sous la forme d'une BD accroît le besoin en confidentialité. L'utilisation de sous-schémas fait partie des moyens qui permettent de l'assurer, vu qu'ils ne rendent visible qu'un sous-ensemble du schéma et par voie de conséquence, de la base. Mais, de façon plus générale, la confidentialité est assurée par le biais de mots de passe et de privilèges d'accès.

Exemple

- Seul le directeur de l'établissement ou son adjoint peut changer l'affectation d'un élève d'une classe à une autre (droit restreint en mise à jour).
- Tout membre de l'administration du lycée peut consulter les informations sur les élèves (droit de consultation).

3.6. La sécurité de fonctionnement

Le SGBD doit offrir des mécanismes permettant de remettre rapidement la base de données dans un état opérationnel en cas d'incident matériel ou logiciel qui en aurait altéré la qualité. Ces mécanismes sont basés sur la journalisation (historique) des opérations réalisées sur la base et leur réexécution automatique en cas de besoin.

Exemple

- Sauvegarde de la base de données une fois par semaine.
- Restauration de la base en cas de panne.

Remarques

- Un bon nombre de ces fonctions font appel ou ressemblent à des fonctions assurées par les systèmes d'exploitation. Aussi, d'un point de vue structure interne, un SGBD peut utiliser des services du système d'exploitation sur lequel il est opérationnel (le SGF pour la gestion physique des données, le système d'entrées-sorties, ...). Aussi, d'un point de vue stratégie de réalisation, la portabilité d'un SGBD dépend fortement du nombre de services du système d'exploitation utilisés: plus l'intersection entre le SGBD et le système d'exploitation est petite, plus l'effort de portage risque d'être faible.
- Par ailleurs, tous les logiciels du marché qui se qualifient de SGBD ne le sont «académiquement» pas au vu des fonctions attendues d'un SGBD que nous venons de passer en revue. Nous pensons particulièrement à certains systèmes de gestion de données sur micro-ordinateurs qui généralement n'offrent que des fonctions de description et de manipulation.

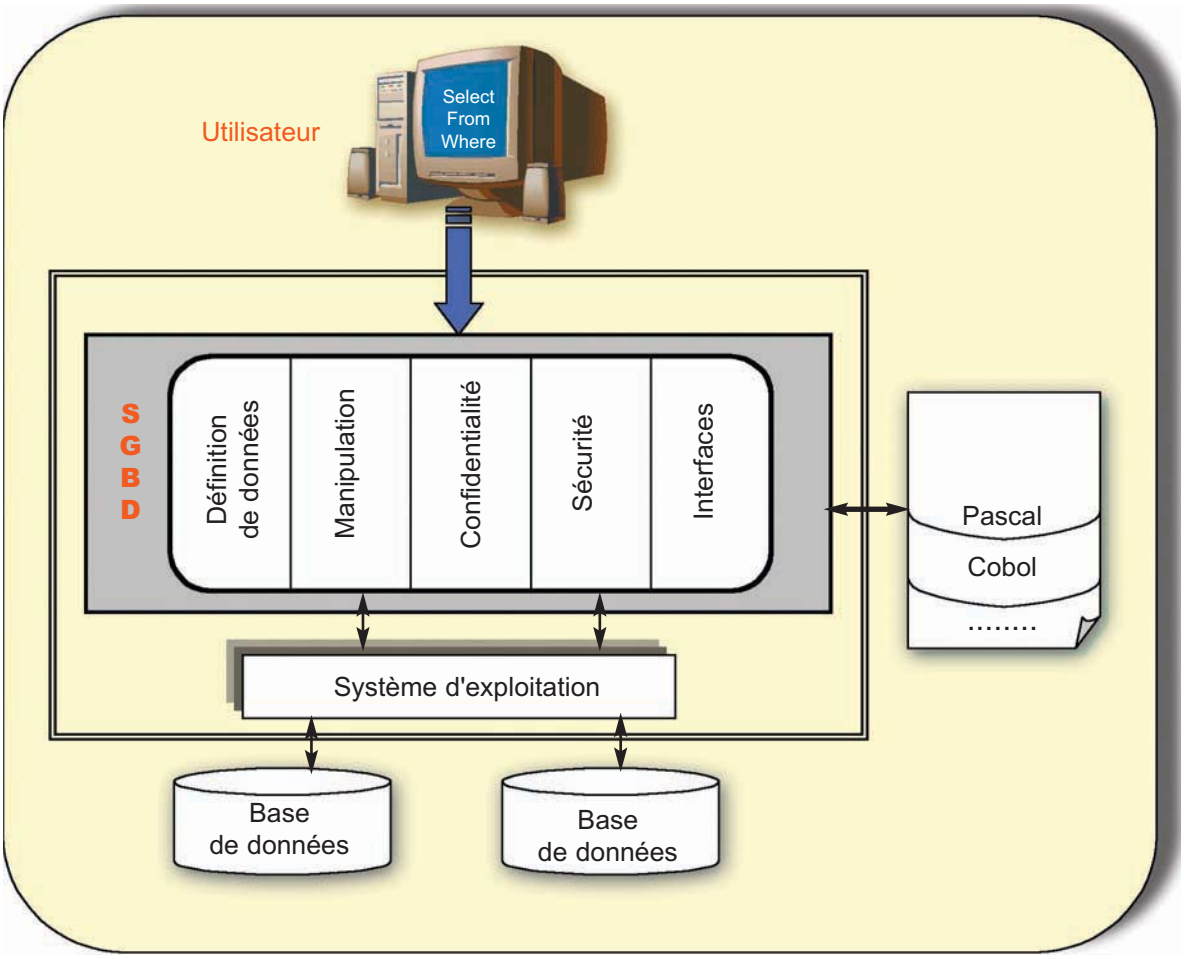


Figure 2.2 : *Architecture fonctionnelle d'un SGBD.*

4. Les principaux SGBD

Activité 1

En effectuant une recherche sur le réseau Internet, énumérer les noms des SGBD les plus utilisés.

Les principaux systèmes de gestion de bases de données sont les suivants :

A propos de certains SGBD

Nom	Commentaires
Oracle	Il s'agit d'un environnement de développement complet comportant notamment un noyau de SGBD relationnel puissant.
DB2	C'est un SGBD relationnel développé par IBM.
MySQL	C'est un SGBD relationnel appartenant à la famille des logiciels libres.
Postgres	C'est un SGBD relationnel appartenant à la famille des logiciels libres, offrant plus de fonctionnalités que MySQL.
Access	Commercialement présenté comme SGBD relationnel. Il appartient à la suite MS Office.
SQL Server	C'est un SGBD relationnel développé par Microsoft pour succéder à Access pour de grosses applications.
...	

5. Cycle de développement des bases de données

Les difficultés méthodologiques rencontrées lors de la mise en place des premières bases de données ont mis davantage en exergue les insuffisances des méthodes d'analyse «classiques» et ont conduit à la recherche de nouvelles méthodes d'analyse et de conception. Suite au succès du rapport ANSI/SPARC, initialement provisoire, d'un groupe de travail de l'institut national américain de normalisation, nombre de méthodes de conception de bases de données s'accordent à distinguer au moins trois niveaux de représentation des données (conceptuel, externe, interne) et à associer une forme de représentation, appelée *schéma*, à chacun de ces niveaux (figure 2.3). Un niveau supplémentaire, le niveau logique, est présent dans certaines méthodes de conception.

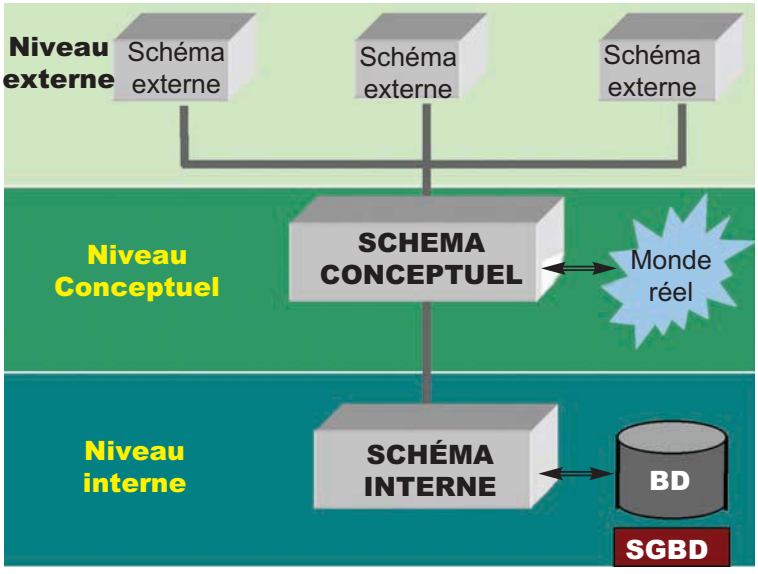


Figure 2.3 : *Architecture ANSI/SPARC et les niveaux de représentation*

Le schéma conceptuel se veut être une représentation abstraite d'une organisation (ou univers ou domaine d'application ou « monde réel ») et de ses règles de gestion et de fonctionnement. Il prend en compte la totalité du domaine de l'étude et non une de ses parties. En outre, il est abstrait dans le sens où il est exprimé dans un formalisme indépendant de toute contrainte technologique et donc, de tout SGBD.
Le niveau externe correspond à la vue de tout ou une partie du schéma conceptuel pour un groupe d'utilisateurs ou d'applications. A ce niveau, sont élaborées des descriptions partielles, appelées *schémas externes*, qui sont les visions qu'auront les futures applications sur la base de données.
Le niveau logique n'était pas mentionné dans le rapport d'ANSI/SPARC. Il a été introduit par des concepteurs de méthodes. Des facteurs quantitatifs, comme le volume des besoins en informations pour chaque application et la fréquence d'exploitation de chaque application, y sont pris en compte pour d'une part, évaluer le coût de la mise en place de la future base de données et d'autre part, pour choisir des chemins d'accès privilégiés aux données.
Au niveau physique interviennent les contraintes liées au matériel et aux logiciels utilisés. Le schéma logique est adapté à ces contraintes donnant ainsi naissance au schéma physique. La structure des données est décrite dans le LDD du SGBD et les traitements sont exprimés en utilisant les différents outils de manipulation de données offerts par le SGBD.
Pour achever la mise en place de la base, nous entrons alors dans des étapes connues du développement de logiciel, à savoir écriture des programmes, chargement d'une base d'essai, test et mise au point des programmes, installation, exploitation et maintenance.

RETENONS

✔ Un système de gestion de base de données (SGBD) est un logiciel qui permet de : décrire, modifier, interroger et administrer les données d'une base de données.
✔ Un SGBD permet la définition, la manipulation, l'intégrité, la confidentialité et la sécurité

LECTURE

Le choix entre un tableur et un logiciel de gestion de base de données est parfois difficile. La comparaison suivante essaie de vous aider à y voir plus clair.

Définition générale

Tableur	SGBD
Un tableur est un logiciel chargé d'aider les humains à construire des tableaux (en lignes et en colonnes), comportant souvent beaucoup de calculs.	Un SGBD est un logiciel chargé d'aider les humains à contrôler des fichiers volumineux et plus complexes que de simples tableaux.

Simplicité d'apprentissage

Tableur	SGBD
Un tableur suit une logique en deux dimensions (lignes et colonnes dans une feuille plane) qui permet de le maîtriser assez rapidement. Par ailleurs, les utilisateurs connaissent un tableur, ce qui facilite les échanges d'information, ainsi que l'entraide. Pour tous les cas simples, un tableur est donc sûrement le meilleur choix.	Un SGBD demande un apprentissage plus long qu'un tableur, mais il permet de régler des problèmes plus complexes et il offre souvent des solutions beaucoup plus satisfaisantes, notamment des applications mieux sécurisées et simples d'emploi.

Volume des données

Tableur	SGBD
Un tableur propose un nombre limité de cellules. On l'utilisera donc pour créer des tableaux de taille «réduite ».	Un SGBD n'impose pas de contraintes sur le nombre d'enregistrements d'une table. Même avec a un enregistrement ayant un nombre important de champs, on peut procéder à un découpage en de multiples tables liées. On va donc pouvoir travailler avec des fichiers beaucoup plus gros.

Mémoire de travail

Tableur	SGBD
Un tableur travaille en mémoire vive, ce qui lui permet d'être très rapide, mais ce qui réduit encore ses possibilités : *souvent, on n'arrivera pas même à remplir toutes ses lignes, faute de mémoire vive suffisante !* Par ailleurs, cette façon de travailler crée un risque pour les données qui n'ont pas encore été sauvegardées par l'utilisateur.	Un SGBD travaille sur disque, sans se soucier de la RAM disponible, ce qui le rend presque insensible à la taille des fichiers : la rapidité reste quasiment identique même avec de grandes bases de données. Par ailleurs, plusieurs SGBD enregistrent sur disque les données saisies au fur et à mesure, sans attendre que l'opérateur lance une sauvegarde. C'est un gage de sécurité.

Organisation des informations

Tableur	SGBD
Un tableur laisse l'opérateur organiser sa petite cuisine de tableaux indépendants les uns des autres, où chaque classeur est un fichier sur disque. Il faut bien penser à les copier tous pour effectuer une sauvegarde ou un transfert. Il est délicat, et très peu sécurisé, de lier les tableaux entre eux.	Un SGBD regroupe toutes les tables dans un seul fichier sur disque, ce qui facilite les sauvegardes et les transferts à une autre personne. Par ailleurs, un SGBD permet de lier les tables entre elles, (il incite même fortement à cette organisation), ce qui lui donne des atouts déterminants en terme de sécurité.

Sécurité des données

Tableur	SGBD
Tri : avec un tableur, il faut être très prudent : sélectionner toutes les colonnes de la table (sans en oublier aucune !) ou ne sélectionner qu'une cellule dans la colonne de tri… sans quoi, le tri ne concerne que certaines cellules (et pas les autres !), ce qui détruit complètement la table… sans le moindre message d'alerte… Si on ne s'en est pas aperçu à temps pour annuler, c'est l'horreur : toute la table est perdue ! **Protection par mot de passe :** un tableur permet de protéger une feuille ou un classeur en bloc, c'est tout… Pas de protection partielle, pour tel utilisateur, pour telles cellules, pour telle action…	Avec un SGBD, un tri ne peut JAMAIS abîmer une table… les données sont en sécurité. Sélectionnez tout ou partie de la colonne de tri et triez… C'est tout, c'est simple et c'est sûr… Par ailleurs, **la gestion des autorisations** est beaucoup plus fine avec un SGBD : on peut donner des droits partiels (soit la lecture, soit l'ajout, soit la modification, soit la suppression…) à certains utilisateurs (pas aux autres, en créant des profils) sur certains objets (et pas les autres) : telles tables, telles requêtes, tels formulaires, tels états…

Contrôles de cohérence

Tableur	SGBD
Un tableur offre peu d'outils Élimination des informations redondantes : un tableur n'offre aucun outil pour organiser les informations en tables liées par des relations : c'est à l'opérateur de tout programmer (par des macros), s'il le juge utile, et ce n'est pas toujours possible ! Contrôle des suppressions et mises à jour : un tableur ne permettant pas d'organiser les infos en tables liées, aucun outil n'est prévu pour contrôler la cohérence des informations : par exemple, rien n'interdira à l'opérateur de supprimer un client alors qu'il existe par ailleurs de nombreuses factures relatives à ce client ! Ces factures deviennent alors incohérentes puisqu'elles pointent sur un client fantôme… Contrôles de saisie : un tableur offre quelques outils très rudimentaires pour aider l'opérateur de saisie en prévoyant des messages et des contrôles de vraisemblance. Enfin, la programmation des réactions du tableau lors des actions de l'opérateur est malaisée (comme par exemple, lorsqu'un code client inexistant est saisi).	**Un SGBD permet presque tout…** Un SGBD, en organisant les informations en tables liées par des clefs externes, offre l'outil idéal pour éliminer toute redondance : chaque information n'est stockée qu'une fois, à un seul endroit, et il suffit de la mettre à jour à cet endroit pour que l'info à jour soit disponible partout dans la base. Ceci est essentiel. Par exemple, l'adresse d'un client n'est mémorisée qu'une fois : dans la table Clients. Chaque fois qu'une facture est adressée à ce client, on ne mémorise pas l'adresse du client dans la facture, on ne mémorise que le code du client, ce code permettant d'aller chercher l'adresse correspondante dans la table Clients. Si le client change d'adresse, on ne met à jour l'information que sur une seule ligne d'une seule table. C'est tout ! Par ailleurs, un SGBD propose très simplement de prendre en charge ce qu'il appelle «l'intégrité référentielle », c'est - à - dire le contrôle de la cohérence de la BD : par exemple, il interdira la suppression d'une catégorie de produits si des produits correspondants existent. Un SGBD permet également de prévoir la suppression en cascade des produits, si l'on préfère cette réaction. Dans tous les cas, la BD reste cohérente : le SGBD s'en charge, rien à programmer… Un SGBD offre également des contrôles très fins de la saisie des informations de base, directement dans les tables.

Intervenants du domaine BD

Les intervenants du domaine peuvent être classifiés en fonction de leur type d'activité et du degré de spécialisation requis pour assurer cette activité.

1. Utilisateurs de bases de données

Nous distinguons :

I. *les utilisateurs occasionnels,* qui ont généralement une technicité moyenne et qui accèdent à la base en utilisant le langage de manipulation de données,

II. *les utilisateurs « naïfs »,* qui accèdent et modifient la base par le biais de transactions préprogrammées (utilisation « presse-bouton »),

III. *les utilisateurs plus* « spécialisés », qui ont généralement une technicité élevée leur permettant de mettre en œuvre les différents outils du SGBD.

2. Concepteurs et développeurs

I. *Le concepteur de bases de données* : il identifie et structure les types de données de la base ainsi que les divers traitements que ces données doivent subir. Ses compétences doivent s'étendre au-delà du domaine strict des bases de données et inclure une ou des méthodes de conception de logiciels.

II. *Les développeurs d'applications :* ils ont pour rôle de déterminer les besoins utilisateurs, de spécifier et d'implanter les transactions et les programmes nécessaires à leur satisfaction. Les compétences requises incluent la construction programmes (algorithmique et programmation) et la maîtrise des langages de manipulation de données.

3. Administrateur des bases et des systèmes

L'administrateur (personne ou équipe de personnes) est responsable d'une ou de plusieurs bases de données. Il a notamment pour rôle la délivrance des autorisations d'accès à la base et à la coordination des activités. Il est également responsable des problèmes de performances et de sécurité de fonctionnement. De ce fait, sa compétence va au-delà de celles des deux précédentes catégories, dans la mesure où il doit avoir connaissance de la façon dont est réalisé le SGBD qu'il doit administrer.

4. Réalisateur de logiciels de gestion et de développement de bases de données

I. *Les développeurs d'ou/ils* : par outil, nous entendons des logiciels facilitant la conception, l'implémentation et l'utilisation de bases de données ou de SGBD existants. Ces outils ne font pas, à proprement parler, partie du SGBD.

II. *Les concepteurs et implémenteurs de SGBD* : une très grande technicité est requise pour cette classe d'acteurs. Leur connaissance doit comprendre, outre les techniques d'implantation des SGBD, des compétences en compilation et langages, en systèmes d'exploitation, en réseaux, etc.

Quelques balises dans le temps

1961

- Apparition du système IDS *(Integrated Data Storage, General Electric).*
- La terminologie utilisée (record types et set types) sera à la base du modèle réseau développé par la *« Conference On DAta SYstems and Languages Data Base Task Group »* (CODASYL DBTG).

1965-1970
- Développement de systèmes de gestion de fichiers généralisés.
- Développement, par IBM, du modèle hiérarchique et du système IMS *(Information Management System).*
- Apparition de IMS DB/DC *(Data Base / DataCom)* qui supporte le modèle réseau « au-dessus » du modèle hiérarchique.
- Les années 70 ont vu une rapide croissance du domaine qui a conduit les BD et les SGBD à devenir une discipline universitaire et de recherche. Ces années sont également marquées par l'apparition de nombreux produits commerciaux implantant partiellement les propositions du rapport CODASYL DBTG : IDS II (Honey Well), DMS1100 (UNIVAC), DMS II (Burroughs), etc.

1970
- Apparition du modèle relationnel de donnée.

1971
- Publication du rapport CODASYL DBTC.

1972
- Première conférence internationale organisée par ACM SIGMOD *(Association of Computing Machinery, Special Interest Group on Management Of Data).*

1975
- Première conférence internationale VLDB *(Very Large Data Base)*.
- Publication du rapport ANSI-SPARC.
- Apparition du modèle individuel, dans le cadre de travaux ayant conduit à la méthode *Merise*.

1976
- Publication du modèle Entité/Association.

1975-1980
- Développement de systèmes relationnels expérimentaux : SYSTEM-R (IBM) et INGRES (Berkeley, University of California).

1980-...
- Apparition et commercialisation de nombreux SGBD relationnels qui ont supplanté les SGBD hiérarchiques et réseaux. Ces années ont également vu la disponibilité des SGBD relationnels sur micro-ordinateurs et la réalisation de systèmes (ou langages) de quatrième génération avec des outils et des interfaces multiples.

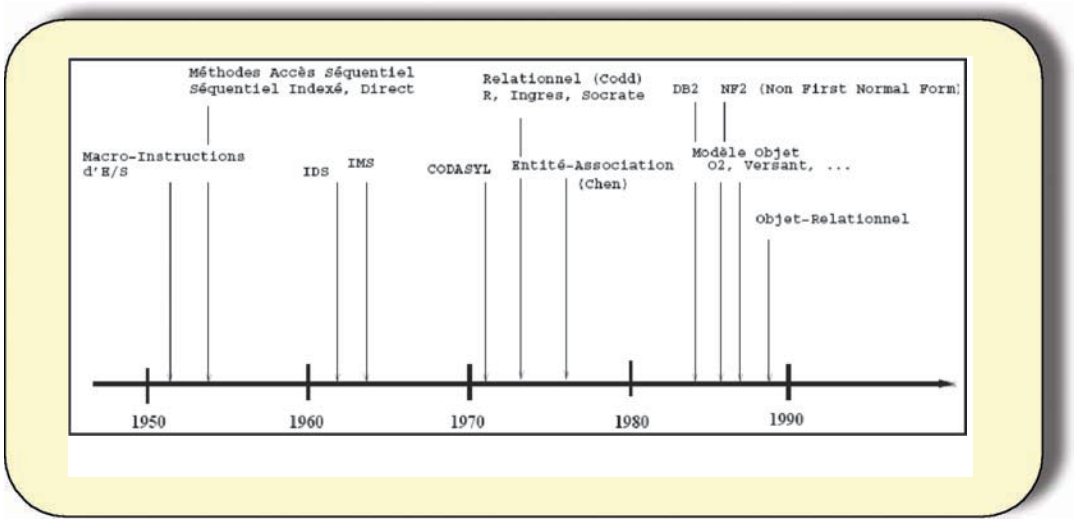
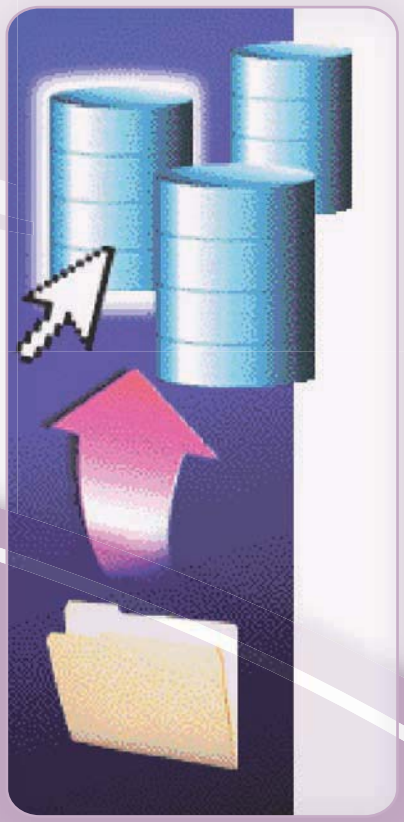


Figure 2.4 : *Quelques balises dans le temps*

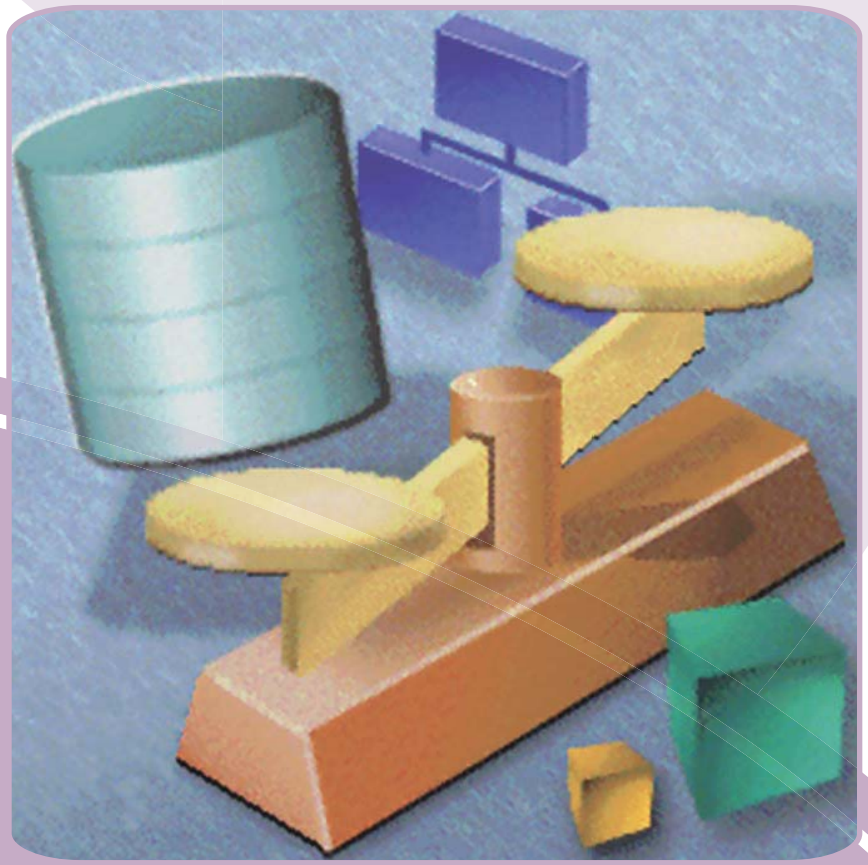
Partie 2

CRÉATION DE BASES DE DONNÉES



Chapitre 3

Structure d'une Base de Données Relationnelle



Objectifs :

- Découvrir les éléments structurels constituant une base de données relationnelle,
- Apprendre à représenter la structure d'une base de données,

Plan :

1. Introduction

Maintenant que vous avez appris qu'une base de données est une collection de données relatives à un ou plusieurs domaines, vous allez découvrir dans ce chapitre comment cette base de données est structurée.

La façon selon laquelle une base de données est structurée, ou organisée, dépend du modèle de données utilisé. Nous avons vu dans le chapitre 1 qu'il y eu différents modèles de données (hiérarchique, réseau, relationnel et objet) et dont le plus utilisé aujourd'hui est le modèle relationnel. C'est à ce modèle que nous allons nous intéresser dans le reste de ce manuel.

Selon le modèle relationnel, une base de données est composé essentiellement de :

- Tables
- Colonnes
- Lignes
- Clés primaires
- Clés étrangères
- Contraintes d'intégrité

Dans ce chapitre vous allez apprendre ces concepts, puis vous allez découvrir les deux formalismes utilisés pour représenter la structure d'une base de données.

Une description de quelques exemples de bases de données est fournie à la fin de ce chapitre.

2. Notion de table

Si on veut décrire une base de données sans trop rentrer dans les détails, on peut dire qu'une base de données est composée d'un ensemble de tables. Autrement dit, les données d'une base de données sont réparties sur un ensemble de tables.

Mais en fait qu'est ce qu'une table ?

Définition

Une table est un ensemble de données relatives à un même sujet (ou entité) et structurées sous forme de tableau.

Comme l'indique la figure suivante, une table est composée horizontalement d'un ensemble de *lignes* et verticalement d'un ensemble de *colonnes* : les colonnes décrivent les *propriétés* relatives au sujet représenté par la table et les lignes correspondent aux *occurrences* du sujet.

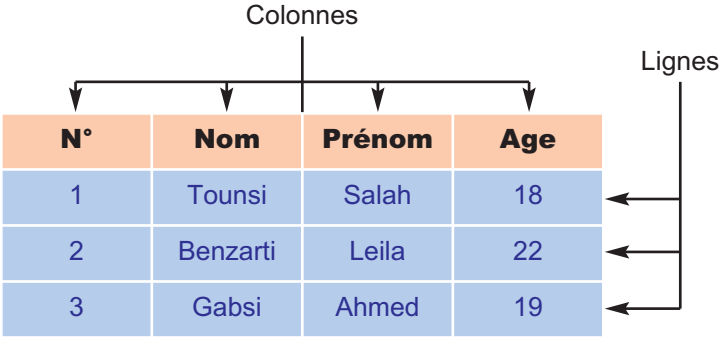


Figure 3.1 : *Structure d'une table*

Un autre terme utilisé également pour désigner une table est celui de la « relation ». Son utilisation se justifie par le fait que dans une table (ou relation), on trouve des données qui sont en relation avec un sujet donné.
Dans une base de données relationnelle, la table constitue la structure la plus importante car la manipulation des données se fait toujours à travers les tables : création, sélection, modification et suppression.

Exemple

La table Article regroupe les données relatives aux articles commercialisés dans un magasin. Chaque article est décrit par :

- **Code article :** C'est un code attribué de façon unique à chaque article.
- **Désignation article :** C'est le nom courant de l'article.
- **Prix unitaire :** C'est le prix de vente de l'article.
- **Quantité stock :** C'est la quantité actuellement disponible pour un article.

A un moment donné, la table Article peut être représentée comme suit :

Code article	Désignation article	Prix unitaire	Quantité en stock
V10	Vis 50x3	40	2500
V20	Vis 20x2	20	1300
B100	Boulon 90x15	450	100
C60	Clou 60x2	5	5000

Figure 3.2 : *Exemple de table*

Remarques

- Il ne faut pas confondre **fichier** et **table**. Les données d'un fichier sont stockées dans un même et seul fichier alors que les données d'une table peuvent être stockées sur un ou plusieurs fichiers, comme on peut regrouper dans un même fichier les données de plusieurs tables.
- Il y a donc une **indépendance** entre la **structure** d'une table et son **implémentation physique** sur les supports de stockage (disque). C'est le système de gestion de base de données (SGBD) qui assure cette indépendance.
- Il y a un lien très étroit entre la notion de **table** et la notion d'**ensemble** en mathématiques. Une table est considérée comme un ensemble de lignes. Certains opérateurs ensemblistes sont applicables à tables telles que l'union, l'intersection et le produit cartésien.

Lexique

Français	Anglais	Arabe	Synonymes
Table	**Table**	**جـــــدول**	**Relation / Table relationnelle**

3. Notion de colonne

Nous avons vu dans la section précédente qu'une table est composée verticalement d'un ensemble de colonnes. Nous allons définir de façon détaillée cette notion.

Définition

Dans une table, une colonne correspond à une propriété élémentaire de l'objet décrit par cette table.

Une colonne est décrite par :

- **Un nom :** C'est le nom de la colonne. Il est sous forme de code et il est généralement soumis aux mêmes règles de nommage des variables dans les langages de programmation.
- **Un type de données :** C'est le type de données prises par cette colonne. Les types de données les plus connus sont : *numérique, chaîne de caractères (ou texte), date et booléen*. Certains systèmes de gestion de base de données supportent des types de données multimédias telles que les images, le son et la vidéo.
- **Une taille éventuelle :** Pour certains types de données tel que le type numérique ou chaînes de caractères, la taille indique la longueur maximale que peut prendre la colonne.
- **Un indicateur de présence obligatoire :** indique si cette colonne doit être toujours renseignée ou peut être vide dans certains cas. Lorsque la colonne n'est pas renseignée, on dit qu'elle contient une valeur nulle. Il est à noter que la valeur nulle est différente de zéro pour les colonnes de type numérique et de la chaîne vide pour les chaînes de caractères.
- **Une valeur par défaut éventuelle :** Permet d'attribuer une valeur par défaut lorsqu'aucune valeur n'a été attribuée à cette colonne.
- **Une règle éventuelle indiquant les valeurs autorisées :** Dans certains cas, une colonne peut être soumise à certaines règles tel que : les valeurs attribuées à cette colonne doivent être inférieures à une certaine valeur, supérieures à une certaine valeur ou bien comprises entre deux valeurs.

Un autre terme utilisé également pour désigner une colonne est celui d' « attribut » ou de « champ ».

La qualité de la description d'une table dépend du choix des colonnes et de la précision dans la description de ces colonnes. Nous verrons plus loin que le processus de description d'une base de données commence par l'identification des colonnes.

Exemple

Nous avons vu dans l'exemple précédent que la table Article regroupe les quatre colonnes suivantes : code article, désignation article, prix unitaire et quantité en stock.

Nous allons décrire de façon détaillée chacune de ces colonnes à travers le tableau suivant. Il est à noter que le nom de la colonne est un code et que nous avons rajouté une colonne pour donner une description de chaque colonne.

Nom de la table : Article
Description : Détail des articles commercialisés

Nom colonne	Description	Type de données	Taille	Obligatoire	Valeur par défaut	Valeurs autorisées
Code_art	Code de l'article	Chaîne de caractères	20	Oui		
Des_art	Désignation de l'article	Chaîne de caractères	50	Oui		
PU	Prix unitaire de l'article	Numérique	8,3	Non		> 0
Qte_stok	Quantité en stock	Numérique	4	Non	0	>= 0

Figure 3.3 : *Description d'une table*

Remarque

- La taille « 8, 3 » pour le prix unitaire signifie que ce prix sera stocké sur huit chiffres dont trois décimales.

Lexique

Français	Anglais	Arabe	Synonymes
Colonne	**Column**	**عمـــود**	**Champ Propriété**

4. Notion de ligne

Une table est composée horizontalement d'un ensemble de lignes. Nous allons définir de façon détaillée cette notion.

Définition

Une ligne correspond à une occurrence du sujet représenté par la table. On dit aussi qu'elle correspond à un objet du monde réel.

Une table est initialement vide lorsqu'elle est créée, c'est-à-dire qu'elle ne contient aucune ligne. L'exploitation d'une table consiste à y *insérer* de nouvelles lignes, à *modifier* certaines lignes ou à *consulter* certaines lignes en fonction de critères bien déterminés. Une ligne peut être *supprimée* lorsqu'on estime qu'on n'en a plus besoin.

Le nombre de lignes d'une table peut être limité à quelques dizaines comme il peut atteindre des milliers, voire des millions de lignes.

Un autre terme utilisé également pour désigner une ligne est celui de l' « enregistrement» ou de « n-uplet ».

Exemple

La représentation de la table Article donnée dans la figure 3.2 contient quatre lignes correspondant chacune à un article commercialisé par le magasin. Cette représentation est valable à un instant donné. La figure suivante donne une représentation de la même table effectuée quelques mois après.

Code article	Désignation article	Prix unitaire	Quantité en stock
V10	Vis 50x3	44	220
V20	Vis 20x2	22	1300
B100	Boulon 90x15	495	300
D120	Disque à meule	3500	30

Figure 3.4 : *Lignes de la table Article*

Remarque

Nous constatons les différences suivantes entre la représentation de la figure 3.2 et celle de le figure 3.4 :

- Les prix ont augmenté de 10%.
- Les quantités en stock ont changé.
- L'article *Clou 60 x 2* n'est plus commercialisé.
- Un nouvel article a été rajouté : *disque à meule*.

Lexique

Français	Anglais	Arabe	Synonymes
Ligne	Record (ou row)	سجـــل	Enregistrement / n-uplet

5. Notion de clé primaire

Activité

1) Dans chacun des tableaux suivants, cocher chaque case permettant d'identifier d'une manière unique l'entité :

Citoyen	Nom	Prénom	Nom et Prénom	Age	N° CIN	N° Sécurité Sociale	Adresse

Voiture	Marque	Modèle	Marque et modèle	Age	N° Immatriculation	N° châssis	Lieu de fabrication

Etudiant	Nom	Prénom	Nom et Prénom	Age	N° CIN	N° Carte Etudiant	Adresse

Elève	Nom	Prénom	Nom et Prénom	Age	N° Téléphone	N° Inscription	Adresse

Ticket de transport	Prix	Ville de départ	Ville d'arrivée	Date	N° Ticket	Société de transport	Validité

2) Développer l'importance du champ qui définit d'une manière unique l'élément
3) Proposer un exemple d'utilisation d'un tel sur l'un des exemples précédents.

Comme dans n'importe quelle organisation de données, on a toujours besoin d'identifier de façon unique chaque occurrence. Dans un fichier papier, on attribue généralement un code ou un numéro pour identifier chaque occurrence. Dans un fichier informatique, on choisit également un ou plusieurs champs pour les utiliser comme identifiant.
Dans une base de données, on a également besoin d'identifier de façon unique chaque ligne d'une table. Ceci se fait à l'aide de la ***notion de clé primaire***.

Définition

La clé primaire d'une table est une colonne ou un groupe de colonnes permettant d'**identifier de façon unique** chaque ligne de la table. Autrement dit, la connaissance de la valeur de la clé primaire, permet de connaître sans aucune ambiguïté les valeurs des autres colonnes de la table.

Exemple

Dans la table Article, la colonne « Code article » peut être utilisée comme clé primaire, car la codification des articles est faite de telle sorte qu'on ne peut pas avoir deux articles qui ont le même code. Connaissant le code article, on peut déterminer de façon unique sa désignation, son prix unitaire et sa quantité en stock.

Remarques

- Chaque table doit comporter une et une seule clé primaire.
- Dans certains cas, dans une même table on peut avoir deux ou plusieurs colonnes qui peuvent jouer le rôle de clé primaires. Dans ce cas on doit choisir une parmi toutes ces colonnes. Par exemple, dans la table article si on suppose que la désignation est unique pour tous les articles, nous pouvons utiliser comme clé primaire, indifféremment, le code ou la désignation de l'article.

- Les colonnes qui constituent la clé primaire sont obligatoires.
- Pour distinguer une colonne qui fait partie de la clé primaire des autres colonnes, on la souligne, ou on la met en **gras**.

Lexique

Français	Anglais	Arabe	Synonymes
Clé primaire	Primary key	مفتـــاح أساســي	identifiant

6. Liens entre tables

Une base de données est la représentation d'un ou plusieurs domaines composé chacun d'un ensemble de sujets (ou entité). Les différents sujets de chaque domaine sont généralement inter-reliés par des liens (ou associations). Pour que la base de données constitue une représentation fidèle du ou des domaines concernés, les liens entre les sujets du monde réel doivent se retrouver dans la base de données.
Si nous avons une table représentant les élèves et une autre table représentant les lycées, la phrase suivante «Un élève est inscrit dans un lycée » correspond à un lien (ou association) entre ces deux tables.

Définition

Un lien entre deux table A et B est représenté par l'ajout dans la table B d'une nouvelle colonne correspondant à la clé primaire de la table A. Cette nouvelle colonne est appelée **clé étrangère**.

Un lien entre deux tables est *orienté* : il part de la table contenant la clé étrangère et arrive vers la table contenant la clé primaire. La table cible (celle contenant la clé primaire) s'appelle **table mère** et la table source (celle contenant la clé étrangère) s'appelle **table fille**. On dit aussi que la table fille **se réfère** à la table mère.

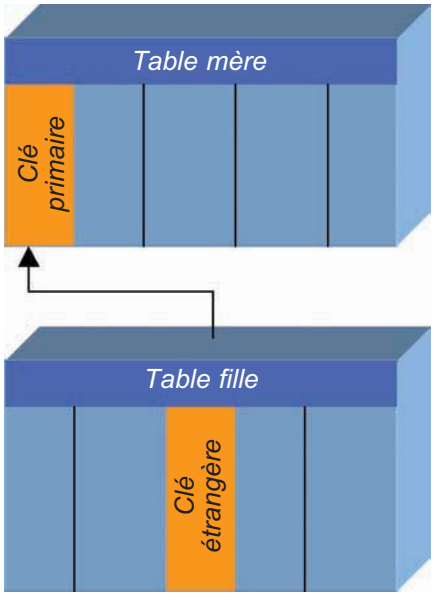


Figure 3.5 : *Lien entre tables*

Exemple

Supposons qu'en plus de la table Article, nous avons une table Commande qui décrit les commandes reçues par le magasin. Dans cet exemple, nous admettons l'hypothèse qu'une commande ne concerne qu'un seul article.
La figure suivante représente ces deux tables avec le lien qui existe entre elles.

Articles

Code article	Désignation article	Prix unitaire	Quantité en stock
V10	Vis 50x3	40	2500
V20	Vis 20x2	20	1300
B100	Boulon 90x15	450	100
C60	Clou 60x2	5	5000

Commandes

N° Commande	Date Commande	Code article#	Quantité commandée
100	01/03/2006	V10	500
101	15/04/2006	B100	30
102	17/04/2006	V10	120

Figure 3.6 : *Lien entre Commandes et Articles*

Remarques

- Une table peut comporter zéro, une ou plusieurs clés étrangères.

Exemple : la table LOCATION décrivant la location des voitures par des clients comporte deux clés étrangères : le numéro d'immatriculation pour désigner les voitures et un code client pour désigner les clients.

- Une clé primaire peut être référencée par zéro, une ou plusieurs clés étrangères.

Exemple : le code client (clé primaire d'une table CLIENT) peut être référencé par la table LOCATION et la table FACTURE.

- Il est fortement recommandé que le nom de la colonne qui est une clé étrangère soit identique au nom de la colonne clé primaire à laquelle elle se réfère.
- Pour distinguer une colonne qui fait partie d'une clé étrangère des autres colonnes, on la double souligne ou bien on la fait suivre d'une dièse (#).

Lexique

Français	Anglais	Arabe	Synonymes
Clé étrangère	**Foreign key**	**مفتـــاح خارجـــي**	**Référence, contrainte d'intégrité référentielle**

7. Notion de contrainte d'intégrité

Assurer la cohérence et l'intégrité des données est l'une des principales fonctions du Système de Gestion de Base de Données (SGBD). Elle consiste à garantir que les données stockées dans une base de données sont cohérentes entre elles, c'est-à-dire qu'elles respectent toutes les règles exigées par le concepteur de la base.
La cohérence et l'intégrité des données sont assurées à l'aide d'un ensemble de règles dites contraintes d'intégrité.

Définition

Une contrainte d'intégrité est une règle appliquée à une colonne ou à une table et qui doit être toujours vérifiée.

Les principaux types de contraintes d'intégrité sont :

- **Les contraintes de domaines :** Ce sont des contraintes appliquées à des colonnes. Elles permettent de fixer le caractère obligatoire ou pas d'une colonne et les règles de validité des valeurs qui peuvent être prises par cette colonne.

Exemple

La note obtenue dans une matière doit être comprise entre zéro et vingt.
La quantité commandée dans la table Commande est obligatoire et doit être positive et supérieure à zéro.

- **Les contraintes d'intégrité de tables :** Elles permettent d'assurer que chaque table a une clé primaire.

Exemple

La table Élève doit avoir une clé primaire, le numéro de carte d'identité par exemple.

- **Les contraintes d'intégrité référentielles :** Elles permettent de s'assurer que les valeurs introduites dans une colonne figurent dans une autre colonne en tant que clé primaire. Elle est représentée sous forme de lien entre tables (voir ***Figure 3.6***.).

Exemple

On n'accepte pas que le Code article saisi dans une Commande n'existe pas dans la colonne Code article de la table Article.

Lexique

Français	Anglais	Arabe	Synonymes
Contrainte d'intégrité	Integrity constraint	قـــيـــد	

8. Représentation de la structure d'une base de données

Après avoir décrit les différentes structures qui constituent une base de données, il nous reste maintenant à donner un formalisme permettant de représenter de façon homogène tous ces concepts.
Cette représentation est appelée **modèle** ou **schéma** de la base de données.
La structure d'une base de données peut être représentée selon deux formalismes :

- Représentation textuelle
- Représentation graphique

Les deux formalismes sont équivalents.

8.1. Représentation textuelle

La représentation textuelle consiste à décrire les tables, les colonnes et les liens entre les tables en utilisant du texte.
Soient les deux tables A et B composées des attributs a1, a2, a3 et a4 pour la première et b1, b2 et b3 pour la deuxième et dont les clés primaires respectives sont a1 et b1. En supposant que B se réfère à A, la représentation textuelle de ces deux tables se fait de la façon suivante :

A (a1, a2, a3, a4)
B (b1, b2, b3, a1#)

Exemple

Les deux tables suivantes représentent les établissements et les élèves qui y sont inscrits.

Etablissement (CodeEtab, NomEtab, AdresseEtab, TelEtab)
Eleve (NumElev, NomElev, PrenomElev, DnaissElev, CodeEtab #)

8.2. Représentation graphique

La représentation graphique consiste à décrire les tables, les colonnes et les liens entre les tables en utilisant des symboles graphiques.
Les deux tables décrites ci-dessus seront représentées comme suit :

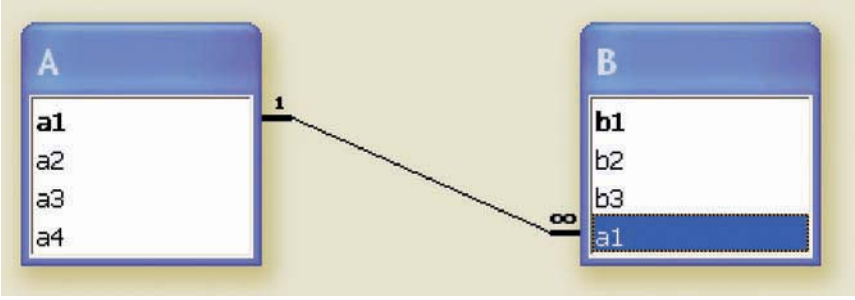
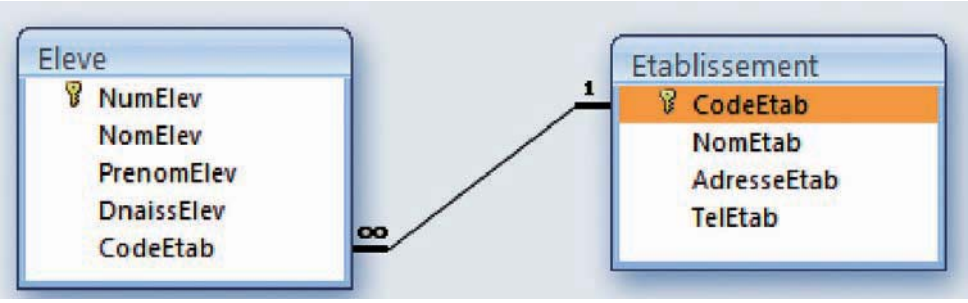


Figure 3.7 : *Représentation graphique des tables*

Les clés primaires sont représentées en gras et les clés étrangères à l'aide d'un lien entre les deux tables : le symbole (∞) est placé du côte de la clé étrangère et le symbole (1) du côté de la clé primaire référencée.
La relation entre les deux tables est dite de type «un à plusieurs» car à une ligne de A peut correspondre plusieurs ligne de B alors qu'à une ligne de B ne peut correspondre qu'une seule ligne de A.

Exemple

Les deux tables, de l'exemple précédent, représentant les établissements et les élèves peuvent être représentées sous la forme graphique suivante :



Remarques

- Dans la représentation textuelle, lorsque la clé étrangère n'a pas le même nom que la clé primaire à laquelle elle se réfère, le lien entre les tables n'est pas visible.
- Les deux représentations ne donnent pas une description détaillée des colonnes (type de données, caractère obligatoire, …). Une description détaillée des tables et des colonnes doit accompagner la description textuelle ou graphique.

9. Exemple de base de données

Considérons l'activité d'un magasin qui commercialise des articles de quincaillerie. Les principaux sujets qu'on peut identifier dans cette activité sont :

- Les articles commercialisés,
- Les clients,
- Les commandes faites par les clients.

Nous appellerons la base de données représentant cette activité «Commercial».
Concernant les articles, la table Article donnée dans les sections précédentes peut être considérée comme suffisante.
Pour les clients, nous allons rajouter une nouvelle table contenant les colonnes suivantes : code client, nom client, adresse client et numéro de téléphone.
En ce qui concerne les commandes, la table Commande donnée dans les sections précédentes se base sur une hypothèse très peu vraisemblable qui suppose qu'une commande est relative à un seul article. Dans la réalité, une commande peut comporter

plusieurs articles. Donc, le sujet commande sera représenté par les deux tables suivantes :

- Une table Commande qui contient le numéro de la commande, la date de la commande et le code du client correspondant.
- Une table Détail commande relative aux lignes de la commande et qui contient les colonnes suivantes : Numéro de la ligne, numéro de la commande, code article et quantité commandée.

Pour représenter ces tables, nous allons utiliser d'abord la représentation textuelle puis la représentation graphique.

Représentation graphique

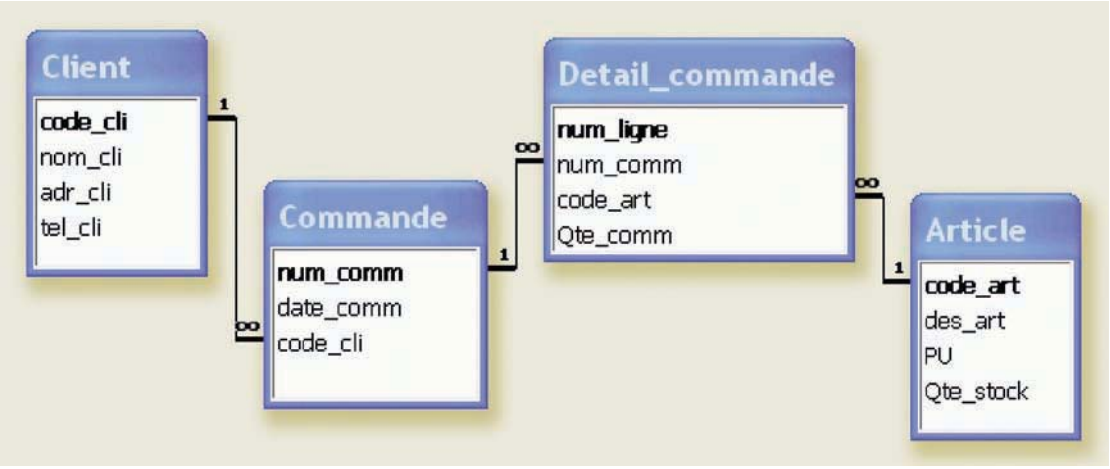


Figure 3.8 : *Structure de la base « Commercial »*

Représentation textuelle :
Article (Code_art, des_art, PU, qte_stock)
Client (Code_cli, nom_cli, adr_cli, tel_cli)
Commande (Num_comm, date_comm, code_cli#)
Detail_commande (num_comm#, Num_ligne, code_art#, qte_comm)

Remarque : num_comm est une clé étrangère, elle fait parti de la clé primaire elle est alors soulignée et suivie de #.

Activités dirigées :

Activité 3.1

Objectif :

Déduire la structure d'une base de données à partir d'un texte.

Enoncé :

On souhaite structurer dans une base de données, les données relatives aux contacts se trouvant actuellement dans un répertoire téléphonique papier.

Pour chaque contact, on dispose des informations suivantes : nom, prénom, adresse postale et adresse mail. Pour chaque contact on peut disposer d'un ou plusieurs numéros de téléphones. Pour chaque numéro on indique son type (P : portable, D : domicile et B : bureau).

Sachant que deux contacts peuvent avoir le même nom et prénom, nous proposons de rajouter un code permettent d'identifier chaque contact.

Question :

Représenter la structure de cette base de données sous forme textuelle puis sous forme graphique

Activité 3.2

Objectif :

- *Comprendre la structure d'une base de données existante.*
- *Avoir un premier contact avec Microsoft Access.*
- *Produire une documentation d'une base de données.*

Enoncé :

On souhaite documenter la base de données existante fournie avec le SGBD mis à votre disposition et intitulée « Comptoir ». Cette base est stockée dans le fichier « comptoir.mdb ».

Questions :

1 - Après avoir ouvert la base de données « Comptoir », établir une liste des tables sous forme d'un tableau donnant pour chaque table son nom et une courte description.

2 - Pour chaque table, établir une fiche donnant la liste de ses colonnes. Utiliser la structure du tableau de la figure 2.3.

3 - Identifier les liens entre les tables. Décrire ces liens à l'aide d'un tableau ayant la structure suivante :

Table mère	Table fille	Clé primaire	Clé étrangère

RETENONS

✔ Une base de données est composée d'un ensemble de tables. Chaque table est une structure tabulaire composée verticalement d'un ensemble de colonnes et horizontalement d'un ensemble de lignes. Chaque ligne est identifiée par les valeurs des colonnes qui constituent la clé

✔ Certaines tables de la base de données sont inter-reliées par des liens permettant d'effectuer un rapprochement de leurs données

✔ La représentation de la structure d'une base de données peut être effectuée selon deux formalismes : une représentation textuelle et une représentation graphique

EXERCICES

Exercice 1

Mettre dans la case correspondante la lettre **V** si la réponse est correcte et la lettre **F** sinon.

1 - Lorsque j'utilise une base de données, je manipule (je créé, je modifie…) :.

☐	Des tables
☐	Des cellules
☐	Des feuilles de calcul

2 - Une ligne correspond à :

☐	Un programme nécessaire pour utiliser la base
☐	L'ensemble des caractéristiques d'un élément de la table
☐	Un annuaire de recherche

3 - Une clé primaire sert à :

☐	Enregistrer une base de données
☐	Identifier les données pour pouvoir les retrouver
☐	L'autre mot pour désigner un programme de base de données

4 - Lorsque je veux ajouter des données à ma base de données, je le fais dans :

☐	une question
☐	un état
☐	un thésaurus
☐	une table

5 - Une colonne correspond à :

☐	un index
☐	une catégorie d'informations, une caractéristique
☐	une zone sur laquelle je peux cliquer pour me déplacer dans la base de données

6 - Une table correspond à :

- ☐ Un ensemble d'informations triées concernant seulement des individus
- ☐ Un ensemble d'informations recueillies sur un site
- ☐ L'ensemble des informations concernant une personne, une entreprise ou un objet (ex. des produits)

7 - Chaque colonne a un format qui peut être de type :

☐ Texte ☐ Date

☐ Code ☐ Age

Exercice 2

Soit la table suivante : **Employé (Matricule, Nom, Adresse, CP, Ville, Date Embauche)**
Compléter le tableau suivant afin de donner une description détaillée des colonnes de cette table.

Nom de la table : .....
Description : .....

Nom colonne	Description	Type de données	Taille	Obligatoire	Valeur par défaut	Valeurs autorisées
.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....

Exercice 3

Soit la table suivante :

NomMed	PrénomMed	CodeMed	NomHôpital	AdrHôpital	TelHôpital
Ben Saleh	Amine	M10	Sahloul	Sousse	73 425001
Amari	Lotfi	M15	Charles Nicolle	Tunis	71 236147
Kaïbi	Henda	M27	Fatouma Bourguiba	Monastir	73 260871

Donner une description détaillée des colonnes de cette table.

Exercice 4

On considère la table suivante décrivant des élèves :

R (Nom, prénom, date_naissance, classe, numéro)

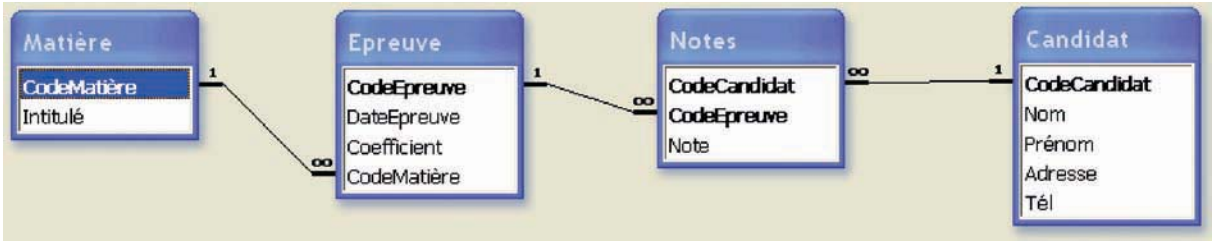
On fait les hypothèses suivantes :

- Dans une classe, un numéro est attribué à chaque élève.
- Dans une classe, les numéros sont attribués par ordre alphabétique des noms des élèves.

Quelle est la clé primaire de cette table ?

Exercice 5

Traduire cette représentation graphique en représentation textuelle.



Exercice 6

Un commerçant veut organiser sa gestion des commandes auprès de ses fournisseurs. Ceux-ci sont définis par un numéro (fno), un nom et une adresse (adr). Les produits ont un numéro (pno), un nom, un prix, un poids et une couleur.

1 - Dégager les tables correspondantes.
2 - Choisir la clé primaire pour chaque table.
3 - Traduire la représentation textuelle en représentation graphique.

Exercice 7

Une compagnie aérienne gère ses données comme suit :

- Un avion a un numéro d'immatriculation et un type. Chaque type d'avion est décrit par son nom, son poids, sa capacité et son rayon d'action.

- Un technicien de la compagnie a un nom, un matricule, une adresse, un numéro de téléphone, un salaire et est expert sur un ou plusieurs type d'avion.
- Un pilote est décrit par les mêmes attributs qu'un technicien. De plus il doit passer un examen médical annuel.
- Chaque avion doit également passer un certain nombre de tests de bon fonctionnement ; chaque test a un numéro qui l'identifie, un nom et une valeur minimale (seuil à atteindre).

1 - Dégager les tables correspondantes.
2 - Choisir la clé primaire pour chaque table.
3 - Traduire la représentation textuelle en représentation graphique.

Exercice 8

On souhaite créer une base de données concernant une entreprise. Une première étude a mis en évidence trois relations. Pour chacune des relations, la clé est soulignée.

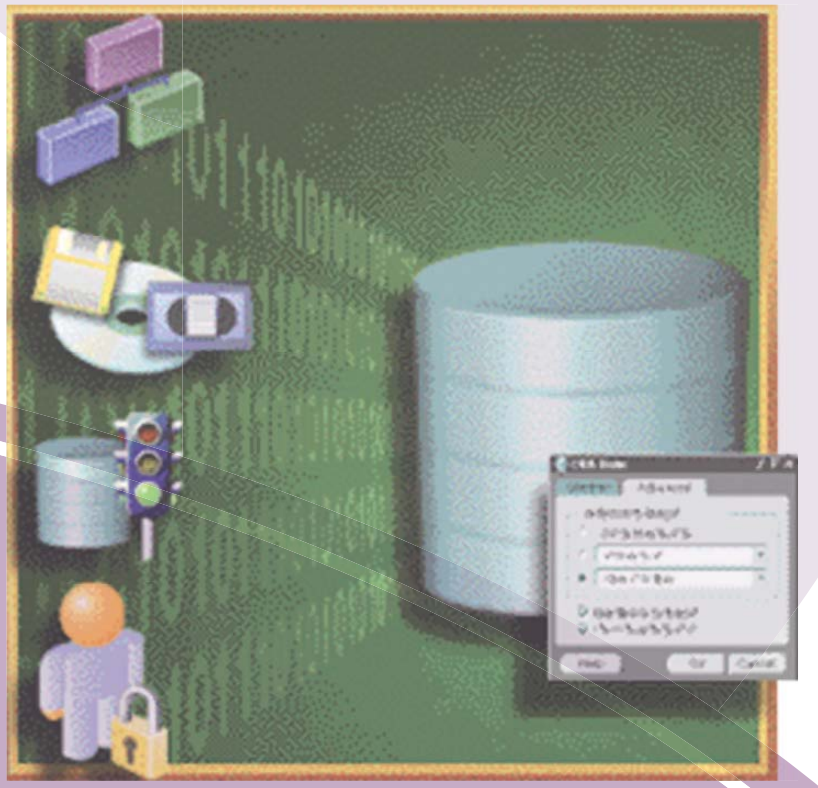
EMPLOYE (NumEmp, Nom, Prénom, Adresse, Téléphone, Qualification)
SERVICE (NomService, Responsable, Téléphone)
PROJET (NomProjet, DateDeb, DateFin, NumEmp)

En considérant les possibilités offertes par ce schéma, répondre aux questions suivantes en justifiant vos réponses par rapport à la sémantique intuitive des relations :

1 - Un employé peut il avoir plusieurs qualifications ?
2 - Un employé peut il faire plusieurs projets en même temps ?
3 - Une personne peut elle être responsable de plusieurs services ?
4 - Un service peut il avoir plusieurs responsables ?

Chapitre 4

Démarche de détermination de la structure d'une Base de Données



Objectifs :

- Découvrir une démarche pour déterminer la structure d'une base de données,
- Apprendre à traduire un énoncé en structures relationnelles,
- Appliquer la démarche proposée à quelques cas concrets.

Plan :

1. Introduction

Dans la vie courante, l'exploitation d'une base de données se fait généralement sur une base de données existante. Dans certains cas, on est amené à créer sa propre base de données ; c'est ce que nous allons voir dans le chapitre suivant.
Comme nous avons déjà défini une base de données comme étant un ensemble structuré de données relatives à un ou plusieurs domaines, la création de la base correspondant à ce(s) domaine(s) doit être précédée par un travail de **réflexion** consistant à trouver la meilleure façon selon laquelle les différents sujets du monde réel doivent être traduits en structures relationnelles, c'est-à-dire en tables, colonnes, clés primaires et clés étrangères. Ce travail de réflexion s'appelle **conception** de la base de données.

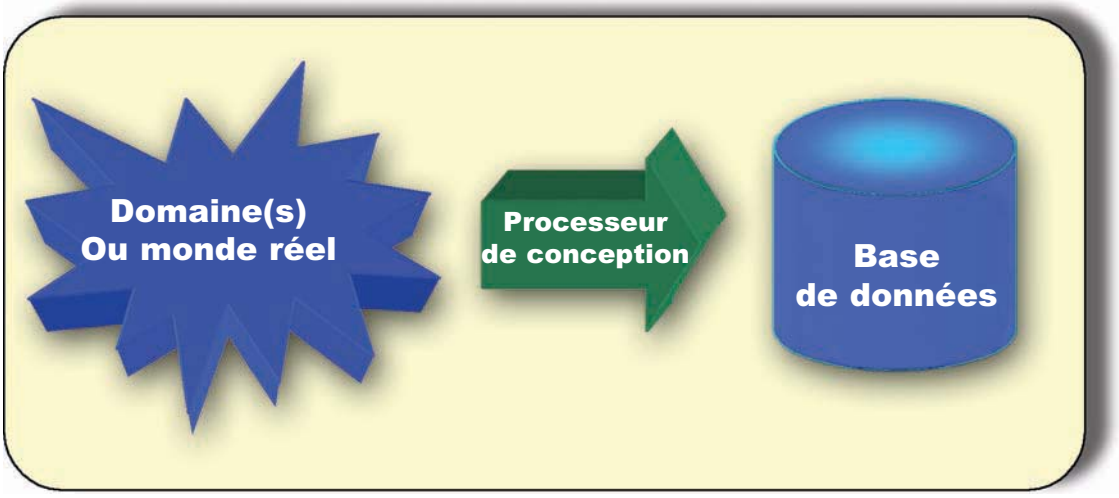


Figure 4.1 : Traduction de domaine(s) en base de données.

La conception d'une base de données se fait selon un processus relativement complexe. Nous proposons ici un processus simplifié, permettant de donner une démarche de conception d'une base de données, sans trop rentrer dans les détails susmentionnés. C'est pour cette raison que nous n'allons pas utiliser le terme « conception », mais plutôt « **déterminer la structure** » d'une base de données.
La démarche pour déterminer la structure d'une base de données est composée des étapes suivantes :

1) Délimiter le(s) domaine(s) concernés
2) Déterminer les colonnes
3) Déterminer les tables
4) Affecter les colonnes aux tables
5) Déterminer les clés primaires
6) Déterminer les liens entre tables
7) Analyser et affiner la structure de la base de données

Nous allons détailler chacune de ces étapes dans les sections suivantes.

2. Délimiter le(s) domaine(s)

Cette première étape consiste à identifier le ou les domaines qui constituent l'objet de la base de données. Est-ce qu'il s'agit par exemple de gérer les données d'une bibliothèque publique, d'une compagnie aérienne, d'un lycée, d'un tournoi sportif, etc. ?
Un autre terme utilisé pour désigner les différents domaines constituant l'objet de la base de données est celui de « monde réel ».
Une fois les domaines identifiés, on doit collecter les informations qui feront l'objet de la base de données. Cette collecte peut être faite à travers des entretiens avec les personnes concernées et/ou un recueil des documents décrivant l'existant (fiches, dossiers, fichiers, feuilles de calcul, …).

3. Déterminer les colonnes

Il s'agit de déduire à partir de la collecte d'informations la **liste des colonnes** qui constitueront la base de données. Chaque colonne est un fait qui se rapporte à un sujet du monde réel. Par exemple, pour le sujet *livre* on peut avoir les faits suivants : code, titre, éditeur et date d'acquisition. Une colonne sera donc associée à chacun de ces faits.
A ce niveau, on doit faire abstraction de la répartition des colonnes en tables. Il s'agit uniquement d'identifier les informations élémentaires.
Lors de la détermination de la liste des colonnes à partir des faits, les règles suivantes doivent être respectées :

- Une colonne doit représenter une information sous sa forme élémentaire, c'est-à-dire que pour la manipuler, on ne doit pas être obligé de la décomposer.

Exemple

Le nom d'un élève doit être représenté par deux colonnes (nom et prénom).

- Une colonne ne doit pas correspondre à une information calculée ou dérivée à partir d'autres colonnes.

Exemple

Si nous avons une colonne « note/20 » et une colonne « coefficient », on ne doit pas rajouter une colonne « note finale » qui est le résultat de la multiplication de la note/20 par le coefficient.

- Des colonnes similaires ne doivent pas être présentes. Il faut garder une seule colonne.

Exemple

Si nous avons trois colonnes « note1 », « note2 » et « note3 », on doit les remplacer par une seule colonne « note ».

- Ne pas omettre de colonnes. L'oubli d'une colonne entraîne un manque d'information dans la base de données.

Exemple

Si nous omettons la colonne « cœfficient » pour une matière, il sera impossible de calculer la note finale.

Pour élaborer la liste de colonnes, on peut utiliser un tableau ayant la structure suivante :

Liste des colonnes							
Nom colonne	**Description**	**Type de données**	**Taille**	**Obligatoire**	**Valeur par défaut**	**Valeurs autorisées**	**Sujet**

Figure 4.2 : *Tableau de description de colonnes.*

Les colonnes de ce tableau ont la même signification que celles du tableau de la figure 3.3 du chapitre 3. La dernière colonne « sujet » a été rajoutée pour permettre d'indiquer quel est le sujet auquel est rattachée la colonne.
Ce tableau sera rempli au fur et à mesure que l'on exploite les documents collectés lors de la première étape. On doit s'assurer que pour chaque colonne les règles citées ci-dessus sont toutes vérifiées.

4. Déterminer les tables

Il s'agit de déterminer la liste des tables qui vont constituer la base de données. Chaque table correspond à un sujet de domaine étudié.
Cette étape est relativement facile car la liste des tables sera déduite à partir du tableau de colonnes. La dernière colonne de ce tableau contient le sujet auquel est associée la colonne. En récupérant la liste des sujets nous obtenons la liste des tables.
Cette liste peut être représentée aussi sous forme du tableau suivant :

Liste des Tables		
Nom table	**Description**	**Sujet**

Figure 4.3 : Tableau de description de tables

5. Affecter les colonnes aux tables

C'est l'étape la plus importante dans le processus de détermination de la structure d'une base de données car c'est de l'attribution d'une colonne à la bonne table que dépend la **qualité** d'une base de données. Affecter une colonne à une table autre que celle à laquelle elle devrait être affectée génère des anomalies lors de la manipulation des données.

Il s'agit donc de reprendre la liste des colonnes et de décider pour chaque colonne à quelle table on doit l'affecter. Ce tableau contient déjà une colonne « sujet » qui correspond aux tables auxquelles les colonnes doivent être affectées.

Pour s'assurer qu'une colonne est affectée à la bonne table, on doit tenir compte des règles suivantes :

- Une colonne doit être affectée à une et une seule table. Une seule exception échappe à cette règle, c'est celle relative aux clés étrangères. Comme une clé étrangère se réfère à une colonne qui est clé primaire dans une autre table, la même colonne existera alors dans les deux tables.
- Si la présence d'une colonne, qui n'est pas clé étrangère, dans une table entraîne que plusieurs lignes de cette dernière vont contenir la même valeur pour cette colonne, c'est que cette colonne se trouve dans la mauvaise table. Il convient donc de l'affecter à une autre table dans laquelle cette anomalie ne se reproduira pas.

Exemple

Si nous plaçons la colonne adresse du client dans la table Commande, l'adresse d'un client donné va se répéter dans toutes les lignes correspondant aux commandes de ce client. S'il a passé 100 commandes, son adresse sera répétée 100 fois. La présence d'une telle anomalie dans une base de données entraîne les inconvénients suivants :

- Une perte d'espace car la même information sera stockée plusieurs fois.
- Un risque d'incohérence entre les données engendré par la modification de cette information dans une ligne et pas dans les autres. Par exemple, modifier l'adresse d'un client dans la ligne correspondant à sa première commande et ne pas le faire dans les autres lignes fait que le même client a plusieurs adresses alors que dans la réalité il n'en a qu'une seule.
- Dégradation des temps de réponses : l'augmentation du volume de données dû à la redondance fait augmenter le temps de réponse lors de la manipulation des données

L'adresse du client doit être donc affectée à la table Client.

6. Déterminer les clés primaires

Chaque table est maintenant décrite par un ensemble de colonnes. On doit alors déterminer parmi ces colonnes celle(s) qui permettent d'identifier les autres colonnes de façon unique, c'est-à-dire les colonnes dont la valeur est unique dans chaque ligne. Ces colonnes constitueront alors la clé primaire de la table

Dans la plupart des cas, une clé primaire est composée d'une seule colonne, mais dans certains cas, elle peut être composée de deux ou plusieurs colonnes. On parle alors de clé composée.

Exemples

- La clé primaire de la table client est composée du « code client ».
- Dans la table « Détail commande », la clé primaire sera composée de la colonne « num_ligne » si nous supposons que le numéro de la ligne est différent pour toutes les lignes de commandes. Par contre, si nous supposons que deux commandes peuvent partager un même numéro de ligne (toutes les commandes commencent par la ligne numéro 1, puis numéro 2, etc.), alors la clé primaire sera composée des deux colonnes « num_ligne » et « num_comm ». C'est une clé primaire composée.

Remarque

Dans le cas où aucune des colonnes de la table ne peut être utilisée comme une clé primaire, nous devons rajouter une nouvelle colonne et la prendre comma clé primaire.

7. Déterminer les liens entre tables

Maintenant que les tables sont complètement décrites avec leurs colonnes et leurs clés primaires, il reste à établir les liens éventuels entre ces tables.
La détermination des liens entre les tables peut être effectuée de deux façons :

- Il existe dans une table B une colonne *b* qui correspond à une colonne *a* dans une autre table *A* et qui est une clé primaire dans cette table. Ceci veut dire que la colonne *b* est une clé étrangère dans la table *B*. Un lien doit être donc établi entre les tables *A* et *B* pour relier les colonnes *a* et *b.*
- D'après la description dont nous disposons sur les sujets représentés par deux tables, nous avons pu déduire qu'il existe un lien entre les deux sujets. Pour établir ce lien entre les deux tables, nous devons d'abord identifier la table « mère » et la table « fille ». Ensuite, on doit rajouter à la table « fille » une colonne qui correspond à la clé primaire de la table « mère ». Cette colonne rajoutée sera une clé étrangère.

Exemple

Supposons les deux tables Client et Commande et que dans la table Commande nous n'avons pas la colonne « code_client ». D'autre part, dans la description de ces deux sujets, nous avons une phrase qui dit « *Une commande est relative à un client* ». Cette phrase veut dire qu'il doit y avoir un lien entre les deux tables Commande et Client. La table mère sera la table Client et la table file sera la table Commande car un client peut avoir plusieurs commandes mais une commande est relative à un et un seul client. On doit donc rajouter dans la table Commande une nouvelle colonne « Code_client » et qui sera une clé étrangère.

8. Analyser et affiner la structure de la base de données

Lorsqu'on a identifié la structure de toutes les tables et établi les liens entre ces tables, il convient de faire une représentation graphique de la base de données afin de l'analyser et de détecter les anomalies éventuelles. Il pourrait s'agir d'un oubli de colonnes ou de liens entre tables.

Lorsque les corrections éventuelles auront été apportées à la structure de la base, nous devons utiliser le SGBD pour créer les différentes tables. Ensuite, on doit insérer suffisamment de lignes dans chaque table pour tester la validité de sa structure. Chaque anomalie constatée, sera corrigée en modifiant la structure (déplacer une colonne d'une table à une autre, créer une nouvelle table, éclater une colonne en plusieurs colonnes, etc.). Il s'agit en fait de s'assurer que toutes les règles vues dans les sections précédentes sont vérifiées.
Certains SGBD disposent d'outils d'analyse permettant d'aider à identifier les anomalies de conception.

Exemple

Dans Microsoft Access, il existe un outil accessible à partir de Outils ⟶ Analyse ⟶Table permettant d'analyser la structure des tables d'une base de données.

RETENONS

✔ La création d'une base de données doit être précédée par un travail de réflexion consistant à déduire la structuration la plus efficace de la base de données en tables

✔ Pour déterminer la structure d'une base de données, on doit d'abord délimiter le monde réel que doit décrire la base de données. Ensuite, on doit identifier la liste des colonnes et la liste des tables puis affecter les colonnes aux tables. Chaque table doit avoir sa propre clé primaire. Les liens entre les tables doivent être ensuite

✔ Cette structuration doit respecter un certain nombre de règles, faute de quoi l'exploitation de la base de données conduira à des anomalies et à un comportement anormal

APPLICATIONS

Application 1

Objectif :

Déduire la structure d'une base de données à partir d'un énoncé décrivant un domaine donné

Enoncé :

Soit à représenter l'activité d'une bibliothèque disposant d'un ensemble de livres qu'elle met à la disposition de ses abonnés.
Chaque livre de la bibliothèque est décrit à l'aide d'un code unique, un titre, un auteur, un éditeur et une date de parution.
L'enregistrement d'un nouvel abonné consiste à renseigner son numéro qui servira comme identifiant, son nom et prénom, son adresse et son numéro de téléphone.
Chaque abonné peut emprunter plusieurs livres. A chaque emprunt on enregistre le code du livre, le numéro de l'abonné et la date d'emprunt. Au retour du livre on enregistre la date de retour.

Questions :

1 - En utilisant un tableau similaire à celui de la figure 4.2, établir la liste des colonnes.
2 - En utilisant un tableau similaire à celui de la figure 4.3, établir la liste des tables.
3 - Affecter les colonnes aux tables et en donner une description textuelle.
4 - Préciser les clés primaires des tables.
5 - Identifier les liens entre les tables en utilisant le tableau de l'activité 2 du chapitre précédent.
6 - Représenter la structure de cette base de données sous forme graphique.

Corrigé :

1- Liste des colonnes :

La liste des colonnes est déduite à partir de l'énoncé et ce en mettant en valeur les propriétés.

Soit à représenter l'activité d'une bibliothèque disposant d'un ensemble de livres qu'elle met à la disposition de ses abonnés. Chaque existe en un seul exemplaire.
Chaque livre de la bibliothèque est décrit à l'aide d'un code unique, un titre, un auteur, un éditeur et une date de parution.
L'enregistrement d'un nouvel abonné consiste à renseigner son numéro qui servira comme identifiant, son nom et prénom, son adresse et son numéro de téléphone.
Chaque abonné peut emprunter plusieurs livres. A chaque emprunt on enregistre le code du livre, le numéro de l'abonné et la date d'emprunt. Au retour du livre on enregistre la date de retour.

Ces propriétés seront regroupées dans le tableau qui suit.
Dans la première colonne, il est recommandé d'utiliser les mêmes règles de nommage qu'en programmation (éviter les caractères accentués, réduire la taille des noms des colonnes, etc.).

Liste des colonnes							
Nom colonne	**Description**	**Type de données**	**Taille**	**Obligatoire**	**Valeur par défaut**	**Valeurs autorisées**	**Sujet**
Code_livre	Code unique du livre	Caractère	10	O			Livres
Titre	Titre du livre	Caractère	50	O			Livres
Auteur	Noms des auteurs	Caractère	50	N			Livres
Editeur	Nom de la maison d'édition	Caractère	30	N			Livres
Date_parution	Date de sortie du livre	Date		N			Livres
Num_abonne	Numéro de l'abonné	Numérique	5	O			Abonnés
Nom_abonne	Nom de famille de l'abonné	Caractère	20	O			Abonnés
Prenom_abonne	Prénom de l'abonné	Caractère	20	O			Abonnés
Adresse	Adresse de l'abonné	Caractère	50				Abonnés
Tel	N° de téléphone de l'abonné	Caractère	20				Abonnés
Code_liv_emp	Code du livre emprunté	Caractère	10	O		(1)	Emprunts
Num_ab_emp	Numéro de l'abonné empruntant un livre	Numérique	5	O		(2)	Emprunts
Date_emprunt	Date de l'emprunt du livre	Date		O	(3)		Emprunts
Date_retour	Date de retour du livre	Date				(4)	Emprunts

(1) Les valeurs de la colonne Code_liv_emp doivent exister dans la colonne Code_livre : un livre emprunté doit exister dans la bibliothèque.
(2) Les valeurs de la colonne Num_ab_emp doivent exister dans la colonne Numero_abonne : une personne ne peut emprunter un livre que lorsqu'il est déjà abonné.
(3) La valeur par défaut de la date d'emprunt est la date système.
(4) La date de retour du livre doit être supérieure ou égale à la date d'emprunt.

2- Liste des tables :

La liste des tables est déduite à partir de la liste des colonnes. A chaque sujet correspond une table. Aux trois sujets «Livre», « Abonne » et « Emprunt » vont correspondre trois tables.

Pour les tables, on appliquera les mêmes règles de nommage que celles adoptées pour les colonnes.

Liste des tables		
Nom table	**Description**	**Sujet**
Livre	Regroupe l'ensemble des livres de la bibliothèque	Livres
Abonne	Regroupe les personnes abonnées à la bibliothèque	Abonnés
Emprunt	Stocke l'historique des emprunts de livres	Emprunts

3- Affectation des colonnes aux tables :

Il s'agit ici d'affecter chacune des colonnes du premier tableau aux trois tables du deuxième tableau en faisant un rapprochement basé sur la colonne « sujet ».
Nous utilisons la représentation textuelle pour décrire la structure des tables.

```
Livre(Code_livre, Titre, Auteur, Editeur, Date_parution)
Abonne(Num_abonne, Nom_abonne, Prenom_abonne, Adresse, Tel)
Emprunt(Code_liv_emp, Num_ab_emp, Date_emprunt, Date_retour)
```

4- Clés primaires des tables :

Pour chaque table, nous allons déterminer une clé primaire. Celle ci sera sélectionnée parmi les colonnes de la table.
Table **Livre** : Il est dit dans le texte que le code de chaque livre est unique. On peut alors l'utiliser comme clé primaire.
Table **Abonné** : Il est également mentionné dans le texte que le numéro de l'abonné sert comme identifiant. On peut donc l'adopter comme clé primaire.
Table **Emprunt** : Un emprunt est identifié par le livre emprunté et l'abonné qui a emprunté ce livre. On peut donc supposer que la clé primaire de cette table est composée des deux colonnes Code_liv_emp et Num_ab_emp.
Cependant, en examinant cette table, on peut constater que cette représentation n'est valide que dans le cas où on ne garde pas un historique des emprunts, c'est-à-dire que lorsque l'abonné retourne le livre, la ligne correspondante sera supprimée de la table Emprunt. Si nous souhaitons garder cet historique on doit rajouter la date d'emprunt à la clé primaire de la table Emprunt, nous adoptons cette alternative.
La description textuelle de la structure des tables en tenant compte des clés primaires sera la suivante :

```
Livre (Code_livre, Titre, Auteur, Editeur, Date_parution)
Abonne (Num_abonne, Nom_abonne, Prenom_abonne, Adresse, Tel)
Emprunt (Code_liv_emp, Num_ab_emp, Date_emprunt, Date_retour)
```

5- Liens entre les tables :

D'après l'énoncé, un emprunt est relatif à un livre et un abonné. Ainsi, on déduit les deux liens suivants :

- un lien entre la table **Emprunt** et la table **Livre**
- un lien entre la table **Emprunt** et la table **Abonne**

Nous utilisons le tableau suivant pour décrire ces liens :

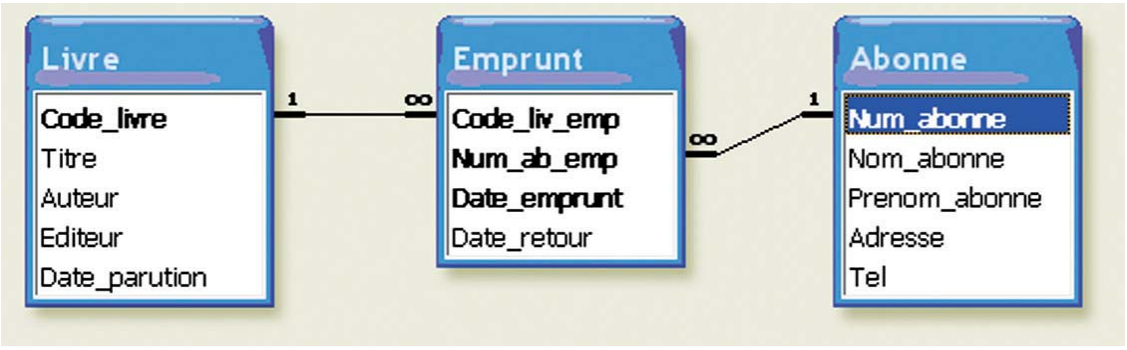
Table mère	Table fille	Clé primaire	Clé étrangère
Livre	Emprunt	Code_livre	Code_liv_emp
Abonne	Emprunt	Num_abonne	Num_ab_emp

Précisons que la clé primaire figure dans la table mère et la clé étrangère dans la table fille.

En tenant compte de ces liens entre les tables, la description textuelle des tables devient alors :

```
Livre(Code_livre, Titre, Auteur, Editeur, Date_parution)
Abonne(Num_abonne, Nom_abonne, Prenom_abonne, Adresse, Tel)
Emprunt(Code_liv_emp#,Num_ab_emp#,Date_emprunt, Date_retour)
```

6- Représentation graphique de la structure de la base de données :



Cette représentation peut être réalisée de façon manuscrite, en utilisant un outil graphique ou un SGBD.

Remarque

L'énoncé n'a pas donné de détail concernant les auteurs et les éditeurs. Nous avons supposé qu'il s'agit uniquement de leurs noms, ce qui a eu comme conséquence de les considérer comme des simples colonnes dans la table *Livre*.
Si l'énoncé avait donné plus d'informations sur les auteurs et les éditeurs (code, nom, adresse, etc.), la conséquence sera l'ajout de deux autres tables *Auteur* et *Editeur* qui seront reliées à la table *Livre*.

Application 2

Objectif :

Détecter les anomalies dans la structure de tables.

Enoncé :

Pour décrire les employés d'une entreprise et leur répartition entre les différents services la table suivante a été créée.

N° emp	Nom	Prénom	Date naissance	N° service	Nom service	Date création Service
1	TOUNSI	Safa	01/10/1980	20	Financier	01/01/1970
2	KEFI	Ali	12/09/1981	10	Administratif	01/01/1975
3	BEJI	Mohamed	15/04/1977	20	Financier	01/01/1970
4	TOUATI	Lamia	21/06/1980	20	Financier	01/01/1970
5	SOUSSI	Leila	28/11/1982	10	Administrative	01/01/1975
6	SFAXI	Nouri	20/08/1990	30	Juridique	01/04/1980
7	GABSI	Mouna	04/04/1987	10	Administratif	01/01/1957
8	JERBI	Lotfi	09/06/1988	30	Juridique	01/04/1980
9	EZZAR	Samia	12/12/1982	20	Financier	01/01/1970

Question :

1 - Identifier les anomalies de cette structure.
2 - Proposer une autre façon de structurer cette base de données.

Corrigé :

1- Anomalies :

- *Incohérence de données :*
  - *On remarque que pour le service N°10, le nom du service n'est pas le même pour les employés N° 2, 5 et 7 (Administratif, Administrative)*
  - *Pour ce même service, la date de création diffère entre les employés N° 2, 5 et 7 (01/01/1975 et 01/01/1957).*

• *Redondance de données :*

- *On remarque que lorsqu'il ya plusieurs employés appartenant au même service, les informations relatives à ce dernier sont dupliquées ce qui a entraîné les incohérences précédentes.*

2- Proposition de solution :

Pour éviter les anomalies, nous proposons d'éclater la table actuelle en deux tables :

***Employe** : elle contient les données relatives aux employés (N°, Nom, Prénom et Date de Naissance)*

***Service** : elle contient les données relatives aux services (N°, Nom et Date de création)*

Pour établir un lien entre ces deux tables, on ajoutera une colonne N°service dans la table Employé servant de clé étrangère.

La structure de ces tables sera :

Employe(Num_emp, Nom_emp,Prenom_emp,Date_naiss_emp,Num_serv#)

Service(Num_serv, Nom_serv, Date_creat_serv)

Les deux tables auront le contenu suivant :

Employe

N°emp	Nom	Prénom	Date naissance	N°service
1	TOUNSI	Safa	01/10/1980	20
2	KEFI	Ali	12/09/1981	10
3	BEJI	Mohamed	15/04/1977	20
4	TOUATI	Lamia	21/06/1980	20
5	SOUSSI	Leila	28/11/1982	10
6	SFAXI	Nouri	20/08/1990	30
7	GABSI	Mouna	04/04/1987	10
8	JERBI	Lotfi	09/06/1988	30
9	EZZAR	Samia	12/12/1982	20
10	BENZARTI	Maroua		

Service

N°service	Nom service	Date création Service
10	Administratif	01/01/1975
20	Financier	01/01/1970
30	Juridique	01/04/1980

EXERCICES

Exercice 1

Il s'agit de déterminer la structure d'une base de données relative à l'organisation de matchs entre des équipes sportives.
Chaque équipe est désignée par un code équipe qui permet de l'identifier parmi les autres équipes, un nom et une date de création.
Chaque équipe est composée d'un ensemble de joueurs. Chaque joueur est identifié par un numéro d'immatriculation et est désigné par un nom, un prénom et une date de naissance. Un joueur appartient à un moment donné à une et une seule équipe.
Chaque match entre deux équipes est désigné par un numéro identifiant, une date, une heure de début et un résultat. Un match est dirigé par un arbitre. Chaque arbitre est identifié par un numéro et a un nom et un prénom. Un arbitre peut diriger plusieurs matchs.
On souhaite également mémoriser la participation des joueurs à chaque match en précisant le rôle de chaque joueur (gardien, défenseur, etc.) pendant ce match ainsi que la durée pendant laquelle il à joué.

Questions :

1. Élaborer la liste des colonnes.
2. Déduire la liste des tables.
3. Donner la liste des liens entre les tables.
4. Donner une description textuelle de la base de données.
5. Donner une description graphique de la base de données.

Exercice 2

On souhaite concevoir une base de données relative à l'organisation de l'enseignement dans un lycée.

L'enseignement est organisé en sections : informatique, mathématiques, lettres, etc. Chaque section est identifiée par un code section et désignée par un intitulé section. Chaque section est composée d'un certain nombre de classes : 4ère année informatique 1 (4I1), 3ème année lettres 2 (3L2) etc. Chaque classe est caractérisée par un code classe (4I1, 3L2, etc.) qui joue le rôle d'identifiant, une désignation et un niveau (1ère, 2ème, 3ème ou 4ème). Une classe appartient à une et une seule section.

Les élèves inscrits dans les différentes classes sont caractérisés par les informations suivantes : un numéro d'élève attribué la première fois que cet élève s'est inscrit, nom, prénom, date de naissance, adresse et numéro de téléphone des parents.

Chaque matière enseignée est caractérisée par un code matière un libellé matière, la section, le niveau d'enseignement et le coefficient.

Des notes sont attribuées aux élèves dans chaque matière à sa section et à son niveau. Pour chaque matière, chaque élève a une note d'oral, une note de devoir de contrôle et une note de devoir de synthèse.

Questions :

1. Élaborer la liste des colonnes.
2. Déduire la liste des tables.
3. Donner la liste des liens entre les tables.
4. Donner une description textuelle de la base de données.
5. Donner une description graphique de la base de données.

Exercice 3

On souhaite concevoir une base de données relative à la gestion des travaux d'un groupe de recherche.
Ce groupe est constitué de chercheurs dont on connaît pour chacun le numéro, le nom, le diplôme, l'activité de recherche, le responsable de recherche (lui-même un chercheur), l'adresse et le téléphone.
Les chercheurs rédigent des articles dont chacun est caractérisé par un titre, le code et le titre du domaine de recherche et une date de rédaction. Un article peut être rédigé par plusieurs chercheurs. Nous supposons que le titre d'un article permet de l'identifier.
Le groupe de recherche anime également des séminaires. Pour chacun, on détient le titre, le lieu, la date, le responsable et les conférenciers. Responsable et conférenciers font partie du groupe de recherche.
Différents participants assistent aux séminaires. Pour chaque participant, on connaît son nom, son prénom, le nom de l'organisme dans lequel il travaille et l'adresse et le type de l'organisme (ministère, entreprise, banque, université, etc.). Toute personne participante est identifiée par son nom et son prénom. Nous supposons qu'on ne peut pas trouver deux personnes ayant le même nom et le même prénom.

Questions :

1. Élaborer la liste des colonnes.
2. Déduire la liste des tables.
3. Donner la liste des liens entre les tables.
4. Donner une description textuelle de la base de données.
5. Donner une description graphique de la base de données.

Exercice 4

On souhaite concevoir une base de données relative à la gestion de la bibliothèque d'un syndicat intercommunal.
Cette bibliothèque consiste en 5 centres de prêt. Ces centres disposent d'ordinateurs personnels interconnectés qui doivent permettre de gérer les emprunts. L'interview des bibliothécaires permet de déterminer les faits suivants :

- une personne qui s'inscrit à la bibliothèque verse une caution. Suivant le montant de cette caution elle aura le droit d'effectuer en même temps de 1 à 10 emprunts;
- les emprunts durent au maximum 15 jours;
- un livre est caractérisé par son numéro dans la bibliothèque (identifiant), son titre, son éditeur et son (ses) auteur(s);

- on veut pouvoir obtenir, pour chaque abonné les emprunts qu'il a effectué (nombre, numéro et titre du livre, date de l'emprunt) au cours des trois derniers mois;
- toutes les semaines, on édite la liste des emprunteurs en retard : nom et adresse de l'abonné, date de l'emprunt, numéro(s) et titre du (des) livre(s) concerné(s);
- on veut enfin pouvoir connaître pour chaque livre sa date d'achat, son état et s'il est disponible dans quel centre.

Questions :

1. Élaborer la liste des colonnes.
2. Déduire la liste des tables.
3. Donner la liste des liens entre les tables.
4. Donner une description textuelle de la base de données.
5. Donner une description graphique de la base de données.

Exercice 5

Un éditeur souhaite installer une base de données pour mémoriser les informations suivantes:

- Les livres sont identifiés par leur numéro ISBN. Un livre possède un titre et un prix de vente. Il est écrit par un ou plusieurs auteurs.
- Chaque livre est tiré en une ou plusieurs éditions, datées et identifiées par leur ordre (première édition, seconde édition, etc.). Chaque édition comporte un certain nombre d'exemplaires. Le prix de vente peut changer d'une édition à l'autre.
- Un livre peut être primé.
- Les auteurs sont identifiés par leur nom et prénoms et peuvent avoir un pseudonyme. Pour chaque livre, un auteur perçoit des droits d'auteur annuels, calculés comme un pourcentage des ventes (il est aussi fonction du nombre d'auteurs).
- Les libraires (identifiés par leur nom et adresse complète) commandent des livres en précisant l'édition et le nombre d'exemplaires désiré.

Questions :

1. Élaborer la liste des colonnes.
2. Déduire la liste des tables.
3. Donner la liste des liens entre les tables.
4. Donner une description textuelle de la base de données.
5. Donner une description graphique de la base de données.

Exercice 6

On souhaite concevoir une base de données relative à la gestion d'un musée.
En effet :

1. toute œuvre du musée a un titre, un ou plusieurs auteurs, une date d'acquisition et un numéro de catalogue (identifiant);

2. une œuvre est exposée dans l'une des salles du musée (qui est caractérisée par un numéro, son nom, le nombre d'œuvres exposables), ou est en prêt dans un autre musée (nom et adresse de ce musée, début et durée du prêt);

3. certaines œuvres exposées dans le musée peuvent avoir été empruntées par le musée, soit à un autre musée, soit à un particulier (nom et adresse). Dans ce cas, on connaît son titre, son (ou ses) auteur(s), la date de début et la durée de l'emprunt. De plus, l'œuvre doit alors être assurée. On veut savoir le montant de la prime d'assurance, la valeur pour laquelle l'œuvre est assurée, le nom et l'adresse de la compagnie qui l'assure;
4. le conservateur garde le fichier des musées et des particuliers qui ont prêté ou qui sont susceptibles de prêter des œuvres. Pour chacun (musée ou particulier), il garde le nom et l'adresse et la liste des collections qui l'intéressent (art déco, art contemporain, antiquités, etc.).

Questions :

1. Élaborer la liste des colonnes.
2. Déduire la liste des tables.
3. Donner la liste des liens entre les tables.
4. Donner une description textuelle de la base de données.
5. Donner une description graphique de la base de données.

Chapitre 5

Création et modification de la structure d'une Base de Données

Objectifs :

- Exploiter des outils logiciels pour créer et mettre à jour la structure d'une base de données.
- Découvrir les commandes SQL pour la création et la modification de la structure d'une base de données

Plan :

1. Introduction

Après avoir présenté les éléments structurels d'une base de données et la démarche à suivre pour déterminer ces éléments, nous allons décrire dans ce chapitre comment créer les différentes structures d'une base de données.
Il existe deux modes pour créer les éléments d'une base de données :

Mode assisté
Mode commande

2. Création d'une base de données en mode assisté

2.1 Modes de création

Le mode assisté permet de créer les éléments de la base de données à travers des assistants graphiques. Un assistant graphique est une interface utilisateur en mode graphique composée d'une succession de fenêtres correspondant chacune à une étape du processus de création. Certains SGBD disposent leurs propres assistants.

Le mode assisté est le moyen le plus facile pour créer les éléments d'une base de données, notamment pour les débutants. Cependant, il peut dans certains cas être limité et ne permet pas toutes les options de création des tables ou de colonnes.
Dans les sections qui suivent, vous allez apprendre à créer les différentes structures d'une base de données en mode assisté, puis vous allez découvrir les trois types de commandes SQL permettant de créer et de modifier la structure d'une base de données.
Ces commandes seront données selon les conventions syntaxiques et typographiques comme indiqué en Annexe 1.

2.2 Création d'une base de données

La création d'une base de données consiste à créer une structure d'accueil pour les tables de cette base de données. Il s'agit en général de préciser :

- Le nom de la base
- L'emplacement physique de cette base, c'est-à-dire le ou les fichiers où seront stockées les données des tables de cette base.

Le processus de création d'une base de données varie de façon significative d'un SGBD à un autre. Les étapes suivantes peuvent être suivies pour créer une base de données à l'aide du SGBD .

1. Démarrer votre SGBD disponible
2. Dans le menu Fichier, choisir l'option « Nouvelle base de données ». Une fenêtre « Nouveau fichier » s'affiche à droite comme indiqué dans la figure suivante :

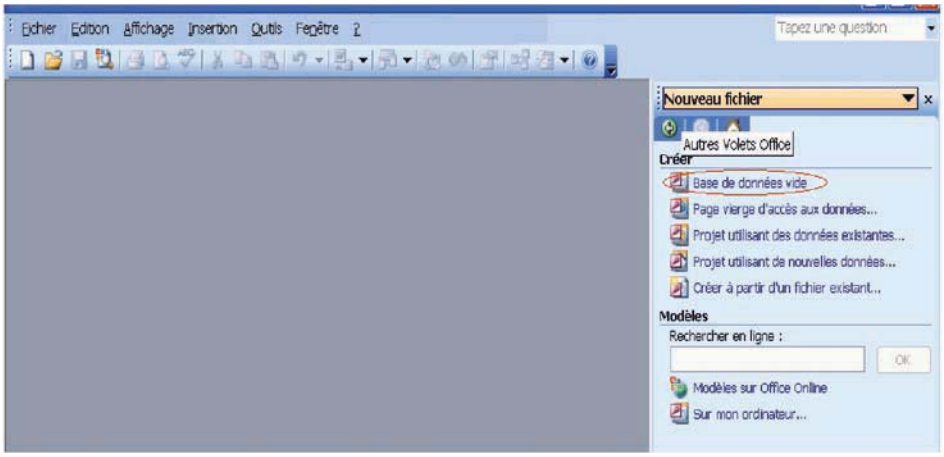


Figure 5.1 : *Fenêtre d'accueil*

3. Choisir «Base de données vide» dans la zone «Créer» de la fenêtre «Nouveau fichier».

4. Dans la fenêtre qui s'affiche, choisir le répertoire dans lequel on souhaite créer la base de données, puis taper le nom de la base en remplaçant le nom **«bd1.mdb»** qui est attribué par défaut.

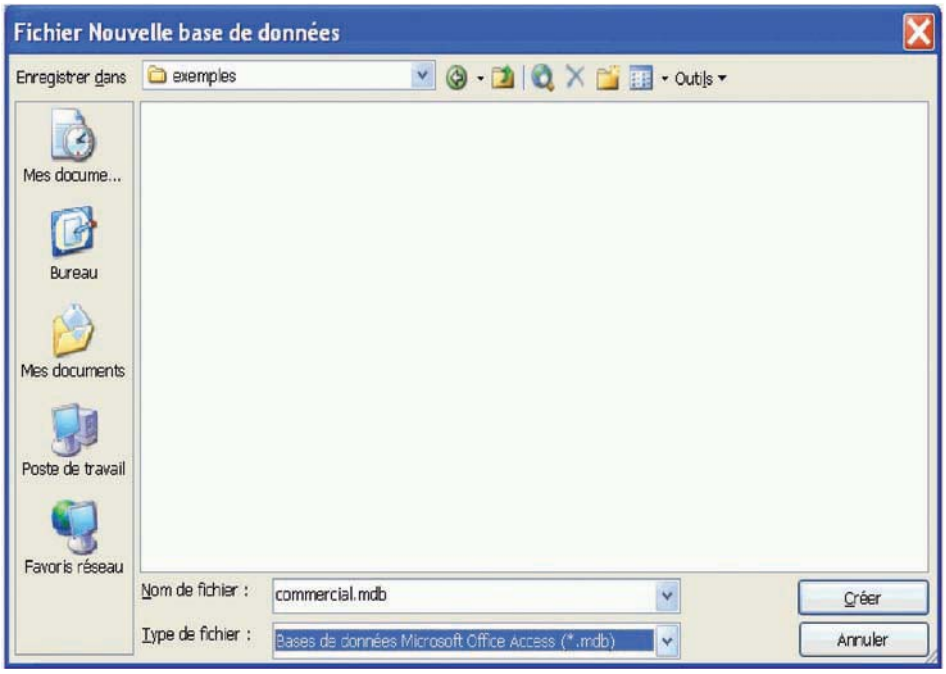


Figure 5.2 : *Fenêtre « Nouvelle base de données »*

5. Cliquer sur le bouton « Créer ». Une base de données vide est alors créée et on peut passer à l'étape suivante consistant à créer des tables dans cette base.
6. Si on souhaite documenter d'avantage la nouvelle base, on peut activer l'option « Propriétés de la base » dans le menu « Fichier » puis renseigner les différents champs de la fenêtre qui s'affiche.

Figure 5.3 : *: Fenêtre « Propriétés de base de données »*

Activité 5.1

Objectif :

Apprendre à créer une base de données.

Enoncé :

On souhaite créer la base de données représentant l'activité d'une bibliothèque.

Question :

Créer une base de données vide intitulée « biblio ».

2.3 Création d'une table

Dans votre SGBD disponible, la création d'une table peut se faire de trois façons différentes :

- **Mode création :** permet de créer une table à l'aide d'une interface qui aide à définir chacune de ses colonnes.

- **Mode assistant :** permet de créer une table en proposant un ensemble de modèles de tables.
- **Mode saisie de données :** permet de générer une table à partir d'un jeu de données saisies par l'utilisateur.

Dans ce qui suit nous allons décrire le premier mode. Les deux autres modes ne présentent pas de difficultés particulières et peuvent être découvertes sans grandes difficultés (voir activité 5.3).

Pour créer une table en mode création, on doit suivre les étapes suivantes.

1. Dans la fenêtre «Base de données», sélectionner l'option «Créer une table en mode création».

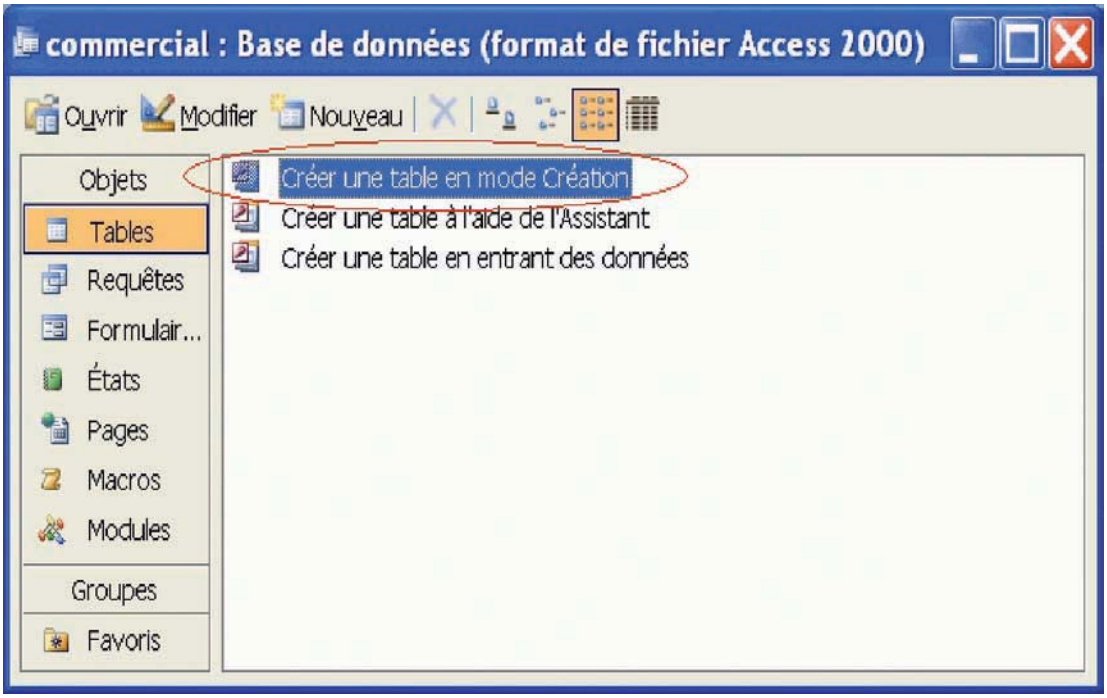


Figure 5.4 : Fenêtre « Base de données »

2. Dans le tableau qui s'affiche, saisir pour chaque colonne de la table son nom, son type de données et sa description. La partie inférieure de cette fenêtre intitulée « Propriétés du champ » peut être utilisée pour décrire de façon détaillée chaque colonne. La partie « Général » permet de préciser la taille, le caractère obligatoire, la condition de validité, la valeur par défaut et d'autres propriétés pour la colonne courante.

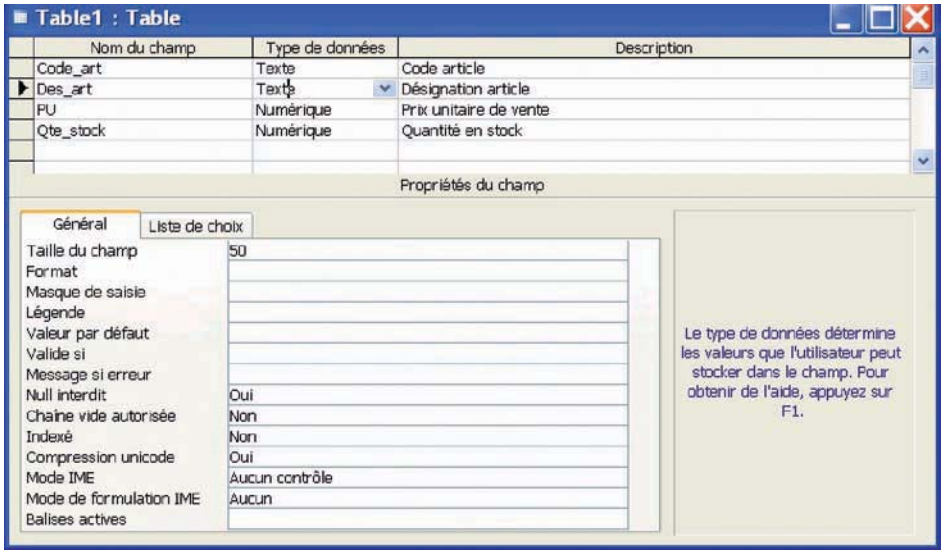


Figure 5.5 : *Fenêtre « Table »*

3. Une fois que toutes les colonnes ont été décrites, fermer cette fenêtre. On vous demandera si vous souhaitez enregistrer votre travail. La réponse par « Oui » entraîne l'affichage d'une fenêtre pour saisir le nom de la nouvelle table.

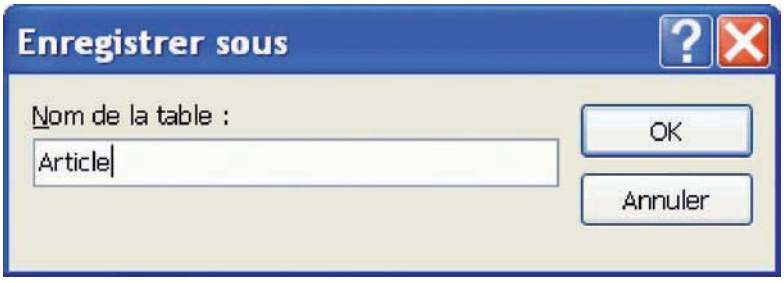


Figure 5.6 : *Fenêtre « Enregistrer table »*

4. Saisir le nom de la table puis cliquer sur Ok. Un message va s'afficher pour indiquer qu'aucune clé primaire n'a été définie pour cette table ; pour proposer d'en rajouter une à cette table. Répondre par « Non ».

5. La nouvelle table est maintenant rajoutée à la liste des tables existantes dans la base de données courante.

6. On peut rajouter une description détaillée de cette nouvelle table en la sélectionnant, puis en activant l'option «Propriétés ». Il convient de préciser que l'activation de la case à cocher «Masquer » entraîne que la table courante ne sera pas visible dans la fenêtre «Base de données ». Ceci peut être utilisé comme un moyen de sécurité pour que cette table ne soit pas visible à tout le monde.

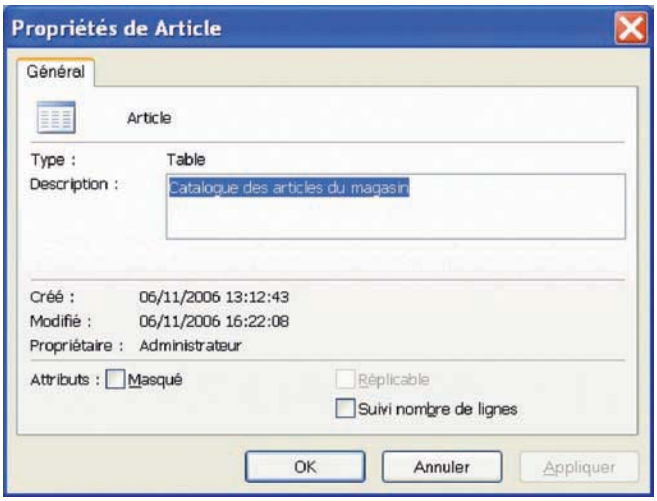


Figure 5.7 : *Fenêtre « Propriétés table »*

Activité 5.2

Objectif :

Apprendre à créer des tables.

Enoncé :

On souhaite créer les tables de la base de données représentant l'activité d'une bibliothèque (suite de l'activité 4.1).

Question :

Créer les tables de la base de données « biblio ».

Activité 5.3

Objectif :

Apprendre à créer une table en mode assisté et en mode feuille de données

Enoncé :

En utilisant l'aide du SGBD disponible, noter les différentes étapes à suivre pour créer une table en mode assisté et en mode feuille de données.

Questions :

1 - *Créer une table relative aux employés d'une entreprise en utilisant le mode assisté.*

2 - *Créer la table T ayant la structure et le contenu suivant en utilisant le mode feuille de données.*

Numéro	Ville	Date_creation
10	Tunis	21/10/2000
20	Bizerte	15/01/2004
30	Kairouan	06/12/2006

2.4 Indiquer la clé primaire d'une table

Pour préciser la clé primaire d'une table, il faut procéder comme suit :

1. Sélectionner la table concernée.

2. Passer en mode création en cliquant sur le bouton 

3. La fenêtre d'édition des colonnes s'affiche. Sélectionner la colonne qui constitue la clé primaire, puis à l'aide du bouton droit de la souris, afficher le menu contextuel et sélectionner l'option « Clé primaire ».

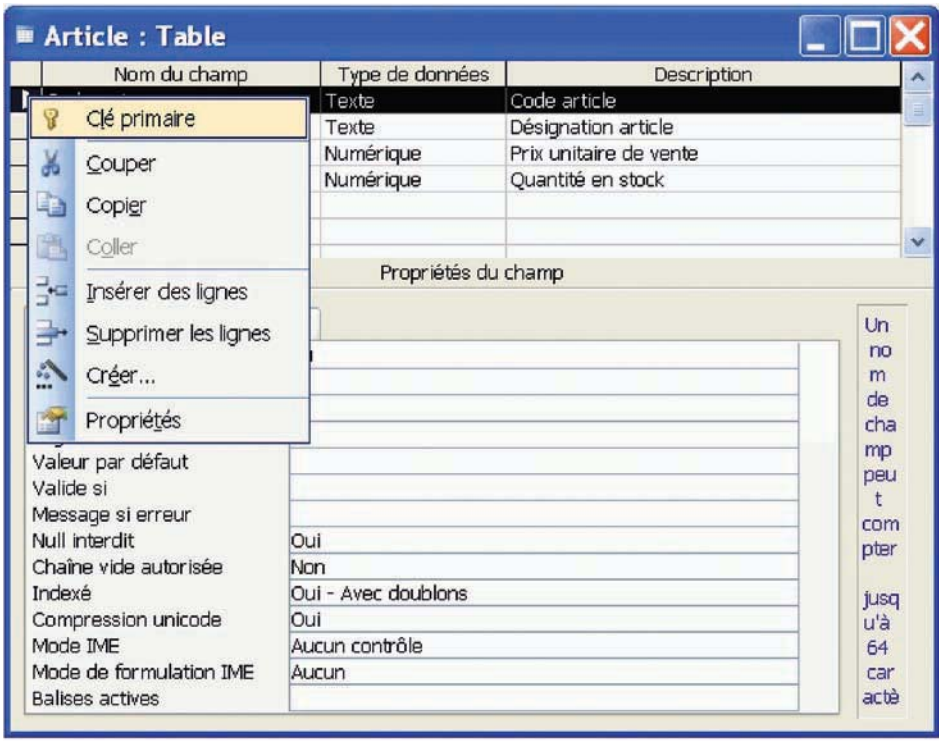


Figure 5.8 : Ajout de clé primaire

4. L'icône apparaîtra à gauche de la colonne constituant de la clé primaire.

▶	Code_art	Texte	Code article
	Des_art	Texte	Désignation article
	PU	Numérique	Prix unitaire de vente
	Qte_stock	Numérique	Quantité en stock

Remarque

Si la clé primaire est composée de plusieurs colonnes, il faut sélectionner toutes les colonnes qui constitueront la clé primaire avant de cliquer sur le bouton droit de la souris.

Activité 5.4

Objectif :

Apprendre à préciser la clé primaire d'une table

Question :

Préciser les clés primaires des tables de la base de données « biblio » déjà créées dans les activités précédentes

2.5 Établir un lien entre deux tables (clé étrangère)

Lorsque toutes les tables ont été créées, on doit rajouter les liens entre ces tables. La procédure à suivre pour établir un lien entre deux tables est la suivante :

1. Sélectionner l'option «Relations» dans le menu «Outils ».Une fenêtre contenant la liste des tables s'affiche.

Figure 5.9 :
Fenêtre « Afficher tables »

2. Sélectionner les tables pour lesquelles vous souhaitez établir un lien puis cliquer sur le bouton Ajouter. Maintenir la touche « Ctrl » pour pouvoir sélectionner plus qu'une table.
3. En fermant la fenêtre, une nouvelle fenêtre contenant les tables sélectionnées s'affiche. Les colonnes constituant les clés primaires sont affichées en gras.
4.

Figure 5.10 : *Fenêtre «Relations »*

5. A l'aide de la souris, cliquer sur la colonne constituant la clé étrangère, maintenir le bouton de la souris enfoncé et pointer sur la colonne constituant la clé primaire dans l'autre table puis lâcher le bouton de la souris. La fenêtre suivante va s'afficher :

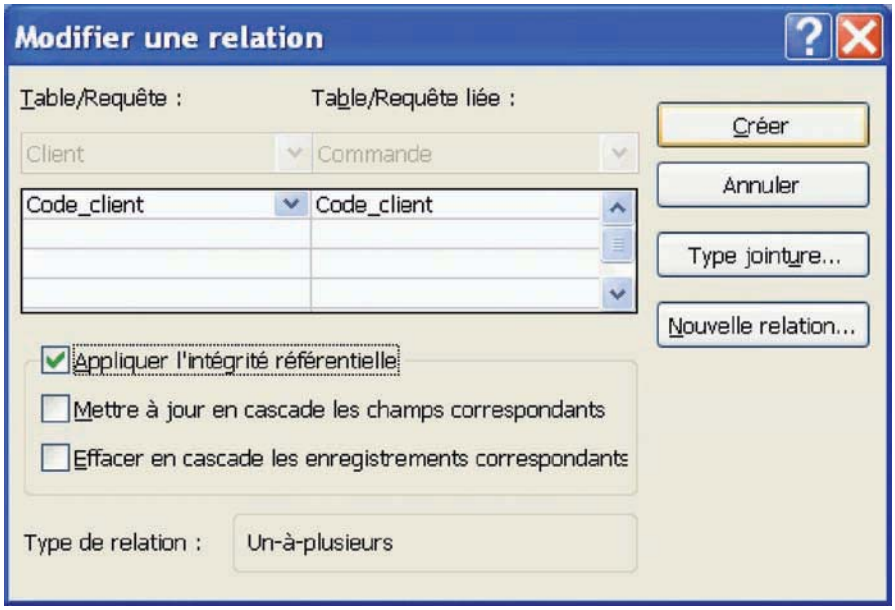


Figure 5.11 : *Fenêtre « Propriétés relation »*

Il est à noter que les colonnes servant à établir le lien entre les deux tables sont affichées dans le tableau. Dans le cas où le lien se base sur plus qu'une colonne dans chaque table, on peut utiliser les lignes vierges de ce tableau pour rajouter les autres colonnes.
Le type du lien est ici « un à plusieurs », signifiant qu'à une ligne de la table mère, peut correspondre une ou plusieurs lignes de la table fille.
6. Cocher les options affichées en fonction du comportement que vous souhaitez appliquer à ce lien.

• **Appliquer l'intégrité référentielle :** Signifie que lorsqu'on insère une ligne dans la table fille, le SGBD vérifie que la valeur saisie dans la colonne clé étrangère existe dans la colonne clé primaire de la table mère. Dans notre exemple, il s'agit de vérifier qu'une commande est relative à un client qui existe dans la table Client.
• **Mettre à jour en cascade les champs correspondants :** Cette option permet de modifier automatiquement la valeur de la clé étrangère dans la table fille lorsqu'on modifie la valeur de la clé primaire dans la table mère. Par exemple, si on modifie le code d'un client dans la table Client, ce code sera modifié dans toutes les lignes de la table Commande correspondant aux commandes de ce client.
• **Effacer en cascade les enregistrements correspondants :** Cette option permet de supprimer automatiquement toutes les lignes dans la table fille correspondant à une ligne supprimée dans la table mère. Par exemple, si on supprime un client dans la table Client, toutes les lignes de la table Commande correspondant aux commandes de ce client seront supprimées.

7. Cliquer sur le bouton « Fermer ». Les deux tables sont maintenant liées.

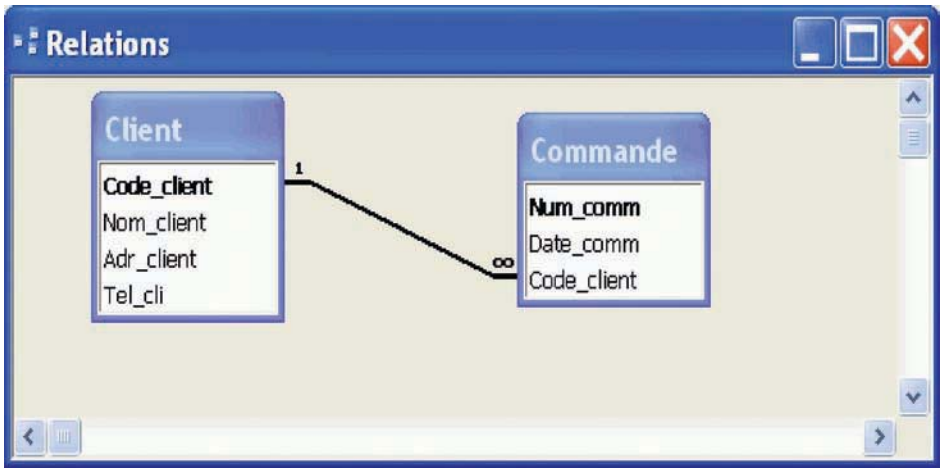


Figure 5.12 : Relation Client-Commande

Remarque

Lorsque les deux colonnes servant à établir le lien entre deux tables sont des clés primaires, le lien crée est de type « un à un ». Ce qui signifie qu'à une ligne de chacune des tables correspond une et une seule ligne de l'autre table. Bien qu'autorisé par certains SGBD, ce cas ne doit pas se produire si on a respecté les règles de conception vues dans le chapitre précédent. Il faut essayer donc d'éviter ce type de lien.

Activité 5.5

Objectif :

Apprendre à établir un lien entre tables

Question :

Etablir les liens entre les tables de la base de données « biblio ».

3. Modification de la structure d'une base de données en mode assisté

3.1 Introduction

Les opérations décrites dans la partie précédente permettent de créer une première version d'une base de données. Nous avons d'autre part vu que dans le processus de description d'une base de données, la dernière étape consiste à analyser et affiner la structure de la base. Affiner la structure de la base de données amènerait éventuellement à apporter des modifications plus ou moins importantes à cette structure telles que :

- Ajout de colonnes à une table
- Suppression de colonnes à une table
- Modification des caractéristiques d'une colonne
- Modification de la clé primaire d'une table
- Suppression d'une table
- Suppression d'une base de données

Nous allons décrire dans les sections qui suivent comment effectuer ces opérations.

Remarque importante

Avant de procéder à la modification de la structure d'une base de données, il est fortement conseillé de commencer par effectuer une sauvegarde de cette base. Il s'agit en effet de faire une copie de la base à laquelle on peut revenir en cas de besoin.
Pour sauvegarder une base de données, en mode Assisté, il faut sélectionner l'option «*Sauvegarder la base de données*» dans le menu *Fichier*. Cette sauvegarde crée un fichier dont le nom est celui de la base de données actuelle suivi par la date courante (exemple : bibio_2007-01-10.mdb).

Activité 5.6

Objectif :

Apprendre à sauvegarder une base de données.

Question :

Effectuer une sauvegarde de la base de données « biblio ».

3.2 Ajout de colonnes à une table (en mode Assisté)

Pour ajouter une colonne à une table existante, on doit suivre les étapes suivantes :

1. Dans la fenêtre «Base de données», sélectionner la table concernée puis cliquer sur le bouton 

ou bien activer l'option «Mode création » du menu contextuel (en cliquant sur le bouton droit de la souris). Une fenêtre «Table» contenant la description de la structure actuelle de la table s'affiche.

2. L'ajout d'une nouvelle colonne peut se faire à la fin du tableau ou à n'importe qu'elle autre position. Il est à noter que l'ordre des colonnes dans une table n'a aucune importance. Pour rajouter une colonne dans une position autre que la dernière, on doit se positionner sur la ligne avant laquelle on veut effectuer l'insertion, puis activer l'option « Lignes » dans le menu Insertion, ou bien « Insérer des lignes »
dans le menu contextuel. Une ligne vierge apparaît et on peut donc créer la nouvelle colonne en procédant de la même façon que lors de la création d'une nouvelle table.
Dans l'exemple suivant, nous avons rajouté à la table client deux colonnes : le prénom du client juste après le nom et l'adresse mail à la fin.

3. Fermer la fenêtre « Table » en confirmant les modifications.

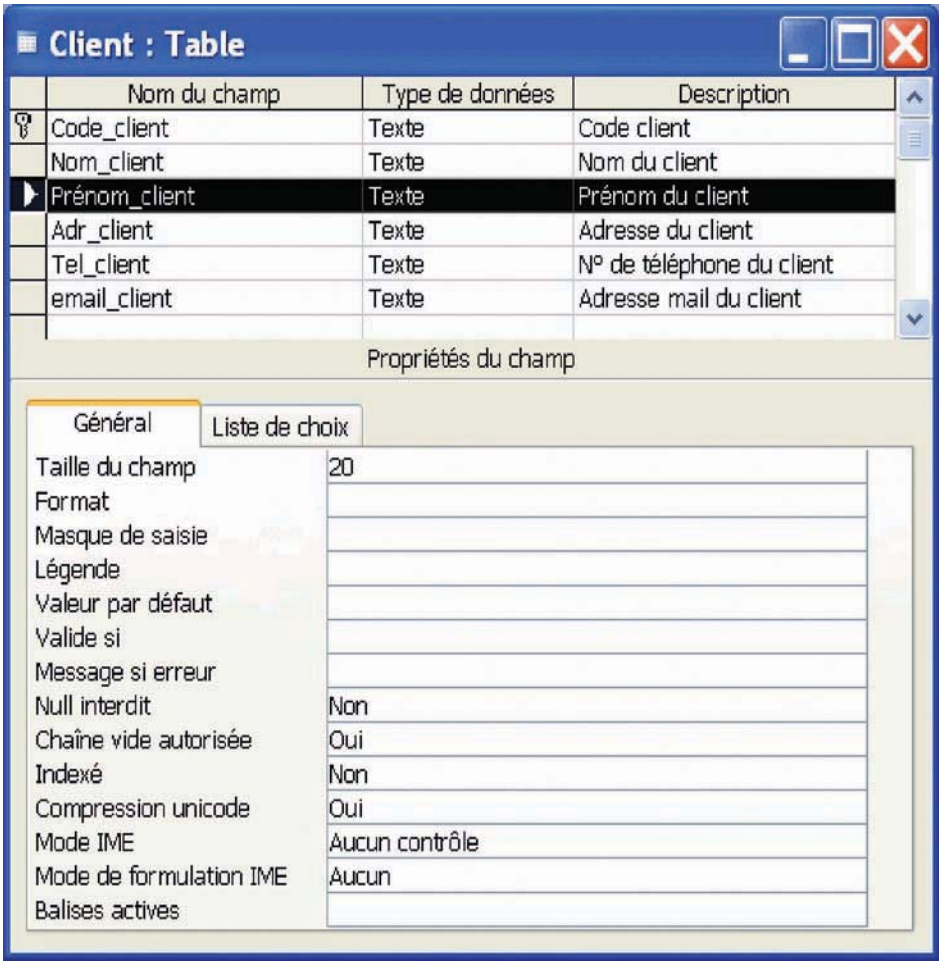


Figure 5.13 : Ajout de colonnes

Activité 5.7

Objectif :

Apprendre à ajouter des colonnes à une table.

Enoncé :

On souhaite enrichir les tables de la base de données «biblio».

Question :

Rajouter une colonne «adresse_mail» pour la table Livre et une colonne «résumé» pour la table Livre.

3.3 Suppression de colonnes d'une table

On peut avoir besoin de supprimer une colonne d'une table dans l'un des cas suivants :

1. Il s'agit d'une colonne qui peut être déduite à partir d'autres colonnes. Nous avons vu dans la section 3.3 que de telles colonnes ne doivent pas exister dans une table.
2. La colonne n'a pas été affectée à la table adéquate. On doit donc la supprimer et la recréer dans la bonne table.

Pour supprimer une colonne d'une table, on doit suivre les étapes suivantes :

1. Dans la fenêtre « Base de données», sélectionner la table concernée puis cliquer sur le bouton 

ou bien activer l'option «Mode création » du menu contextuel (en cliquant sur le bouton droit de la souris). Une fenêtre «Table » contenant la description de la structure actuelle de la table s'affiche.
2. Sélectionner la ou les colonnes à supprimer, puis activer l'option « Supprimer » dans le menu Edition, ou bien « Supprimer des lignes » dans le menu contextuel. Les lignes sélectionnées seront supprimées.
3. Fermer la fenêtre « Table » en confirmant les modifications.

Remarque

Lorsque la colonne à supprimer est une clé primaire référencée dans une autre table comme clé étrangère, sa suppression ne peut se faire qu'après la suppression du lien entre les deux tables.

Activité 5.8

Objectif :

Apprendre à supprimer des colonnes d'une table.

Enoncé :

Supprimer la colonne « résumé » qui vient d'être rajoutée à la table Livre.

3.4 Modification de caractéristiques d'une colonne (en mode Assisté)

La modification des caractéristiques d'une colonne consiste à modifier :

- Le nom de la colonne
- Le type de données
- La description
- Les autres propriétés : taille, caractère obligatoire, la règle de validité, etc.

Pour modifier les caractéristiques d'une colonne d'une table, on doit suivre les étapes suivantes :

1. Dans la fenêtre « Base de données», sélectionner la table concernée puis cliquer sur le bouton 

ou bien activer l'option «Mode création» du menu contextuel (en cliquant sur le bouton droit de la souris). Une fenêtre «Table» contenant la description de la structure actuelle de la table s'affiche.

2. Se positionner sur la colonne à modifier. Si la modification concerne le nom, le type de données ou la description, effectuer cette modification dans la ligne du tableau. Si la modification concerne une autre propriété, effectuer la dans la partie « Propriétés du champ ».
3. Fermer la fenêtre « Table » en confirmant les modifications.

Remarques

- Lorsque des modifications de caractéristiques d'une colonne sont apportées à une table qui contient déjà des données, il faut prendre les précautions suivantes :

  * Pour la modification des types de données, il faut tenir compte de la compatibilité des types. Par exemple, on peut passer du type numérique vers le type chaîne de caractères sans risque alors que l'inverse n'est possible que si cette colonne ne contient que des valeurs numériques.

  * Pour la modification de la taille, on peut passer sans aucun risque d'une taille inférieure à une taille supérieure, alors que l'inverse n'est possible que si toutes les données existantes pour cette colonne ont une taille inférieure ou égale à la nouvelle taille.

- La modification du type de données d'une colonne déjà référencée dans une autre table ne peut se faire que lorsqu'on supprime le lien entre les deux tables. Lors du rétablissement de ce lien après la modification du type de données, le SGBD s'assure de la conformité des types de données des deux colonnes. S'il y a incompatibilité, le lien ne sera pas établi.

Activité 5.9

Objectif :

Apprendre à modifier les caractéristiques d'une colonne.

Enoncé :

Ajouter une colonne «Etat» dans la table Abonné et rajouter la condition de validité suivante : un abonné est dans un état Actif ou Inactif.

3.5 Modification de la clé primaire d'une table (en mode Assisté)

La modification d'une clé primaire peut consister à :

- Rajouter d'autres colonnes à la clé primaire actuelle. On a par exemple une table dont la clé primaire est composée d'une seule colonne et on souhaite enrichir cette clé primaire par une autre colonne.

- Sortir une ou plusieurs colonnes de la clé primaire actuelle. On a par exemple une table dont la clé primaire est composée de deux colonnes et on souhaite sortir une colonne de cette clé primaire.

- Changer complètement de clé primaire, c'est-à-dire que les colonnes de la clé primaire seront complètement différentes de celles qui existent actuellement.

Pour modifier la clé primaire d'une table, on doit suivre les étapes suivantes :

1. Dans la fenêtre « Base de données », sélectionner la table concernée puis cliquer sur le bouton 

ou bien activer l'option « Mode création » du menu contextuel (en cliquant sur le bouton droit de la souris). Une fenêtre « Table » contenant la description de la structure actuelle de la table s'affiche.
2. Pour annuler le fait qu'une colonne est actuellement une clé primaire, sélectionner cette colonne, puis activer l'option « clé primaire » du menu contextuel.
3. Pour intégrer une ou plusieurs colonnes dans la clé primaire, sélectionner ces colonnes, puis activer l'option « clé primaire » du menu contextuel. Toutes ces colonnes seront donc marquées par l'icône  .
4. Fermer la fenêtre « Table » en confirmant les modifications.

Remarques

- Lorsque la clé primaire actuelle est référencée dans une ou plusieurs autres tables, il faut supprimer le lien entre les tables avant de procéder à la modification de la clé primaire.
- Lorsque la table contient déjà des données, la modification de la clé primaire ne peut être acceptée que si les données actuelles vérifient l'unicité des valeurs des colonnes constituant la clé primaire de la table. Par exemple, si on souhaite modifier la clé primaire de la table client de sorte que la nouvelle clé primaire sera le nom du client et non pas le code du client, cette modification ne sera acceptée que si tous les clients actuels ont des noms uniques.

Activité 5.10

Objectif :

Apprendre à modifier la clé primaire d'une table.

Enoncé :

On souhaite vérifier la validité de la clé primaire de la table Emprunt. Pour ce faire, essayer d'y insérer quelques lignes correspondant à des emprunts de livres différents par un même abonné. Ensuite, essayer d'insérer une ligne correspondant à un emprunt d'un livre qui a été déjà emprunté par cet abonné à une date antérieure

Question :

1 - Quelle est la réaction du SGBD ?
2 - Donner une explication.
3 - Proposer une solution.

3.6 Suppression d'une table (en mode Assisté)

La suppression d'une table consiste à supprimer son contenu et sa structure.
Pour supprimer une table, on doit suivre les étapes suivantes :

6. Dans la fenêtre « Base de données », sélectionner la table à supprimer.
7. Activer l'option « Supprimer » dans le menu Edition, ou bien « Supprimer» dans le menu contextuel.
8. Au message de confirmation de la suppression, répondre par Oui ou par Non.

Remarque

- Lorsque la table à supprimer est référencée dans une ou plusieurs autres tables, il faut supprimer le lien entre les tables avant de procéder à la suppression de la table. Ne pas oublier de supprimer les colonnes qui se réfèrent à cette table dans les autres tables.

Activité 5.11

Objectif :

Apprendre à supprimer une table.

Enoncé :

Créer une nouvelle table puis la supprimer.

3.7 Suppression d'une base de données (en mode Assisté)

La suppression d'une base de données consiste à supprimer tous les objets se trouvant dans la base ainsi que la structure physique de cette base, c'est-à-dire le fichier qui la contient. Vu l'importance de cette opération, il faut bien réfléchir avant de l'effectuer. Dans tous les cas, il est fortement conseillé d'effectuer une sauvegarder de la base avant de la supprimer.

Nous remarquons que certains SGBD ne disposent pas de commande de suppression d'une base de données. Cette suppression doit être alors faite au niveau système d'exploitation.

Exemple

Pour supprimer la base de données « Commercial », il faut aller vers le répertoire correspondant et supprimer le fichier « Commrcial.mdb ».

Remarque

Dans le cas où on a procédé par erreur à la suppression d'une base de données, il est possible de la restaurer à partir de la corbeille si celle-ci n'a pas été « vidée » entre temps.

Activité 5.12

Objectif :

Apprendre à supprimer une base de données.

Enoncé :

Créer une base vide puis la supprimer.

4. Création d'une table en mode commande

Le mode commande consiste à créer les différentes structures de la base de données à l'aide de commandes du langage SQL. Ce langage est composé de trois familles de commandes :

- **Commandes de définition de données** : ce sont des commandes qui permettent de créer, modifier et supprimer les différentes structures de la base de données. Nous utiliserons quelques unes de ces commandes dans ce chapitre.

- **Commandes de manipulation de données** : ce sont des commandes qui permettent de manipuler le contenu de la base de données, c'est-à-dire d'insérer, de modifier, de consulter ou de supprimer des lignes dans les tables de la base de données. Ces commandes seront présentées dans le chapitre 6.

- **Commandes de contrôle de données** : ce sont des commandes qui permettent de contrôler l'utilisation de la base de données (sécurité de la base, intégrité des données, cohérences des données). Les commandes relatives à la sécurité seront présentées dans le chapitre 7.

  La commande du langage SQL permettant de créer une table est la commande CREATE TABLE.
  La forme générale de cette commande est la suivante :

```
CREATE TABLE nom_table
(définition_ colonne | définition_ contrainte, ... )
```

Le nom de la table doit être unique au niveau de la base de données et ne doit pas être un mot-clé SQL.

Une table doit contenir au minimum une colonne. Le nombre maximum de colonnes par table dépend du SGBD. Les noms attribués à ces colonnes doivent être uniques dans la même table, mais plusieurs tables peuvent avoir des colonnes qui ont le même nom.

La clause «*définition_colonne*» permet de préciser les caractéristiques d'une colonne. Elle a la syntaxe suivante :

```
Nom_colonne type [[NOT] NULL] [DEFAULT valeur] [contrainte_colonne]
```

Le nom de la colonne est suivi obligatoirement par le type de données. Le tableau suivant donne les principaux types de données. Il est à noter que le nombre et les noms des types de données varient d'un SGBD à un autre.

INT (n)	Numérique à n chiffres
DECIMAL (n, m)	Numérique à n chiffres dont m décimales.
VARCHAR(n)	Chaîne de caractères de longueur variable dont la taille maximale est n
DATE	Date et/ou heure

L'option NULL veut dire que la colonne n'est pas obligatoire. On peut lors de la saisie d'une ligne de la table, laisser la valeur de cette colonne à Null (vide).
A l'inverse, l'option NOT NULL veut dire que la colonne est obligatoire. On doit absolument renseigner cette colonne lors de la saisie d'une ligne de la table.
Lorsqu'une colonne est une clé primaire, on n'a pas besoin de préciser l'option NOT NULL. Elle est implicite.
L'option DEFAULT permet d'attribuer une valeur par défaut à cette colonne lorsqu'aucune valeur ne lui a été affectée. Cette option ne peut pas être indiquée lorsque la colonne est obligatoire (NOT NULL).
L'option «contrainte_colonne » permet de préciser une contrainte d'intégrité relative à la colonne. Cette contrainte peut être une contrainte de clé primaire, de clé étrangère ou de valeurs. La syntaxe correspondante est la suivante :

```
[CONSTRAINT contrainte]
{PRIMARY KEY
| REFERENCES nom_table [(nom_colonne)] [ON DELETE CASCADE]
| CHECK (condition)}
```

Le mot clé CONSTRAINT est optionnel et sert à attribuer un nom à la contrainte.
Le paramètre **contrainte** sert en tant qu'identificateur.
PRIMARY KEY spécifie que la colonne est utilisée comme clé primaire.
REFERENCES définit une contrainte d'intégrité référentielle. Le nom de la table précisé après le mot-clé REFERENCES est celui de la table mère. Le nom de la colonne est celui de la colonne vers laquelle on se réfère et il ne doit être précisé que lorsqu'il est différent du nom de la colonne courante.
ON DELETE CASCADE est une option qui permet de maintenir l'intégrité référentielle en supprimant automatiquement les valeurs d'une clé étrangère dépendant d'une valeur d'une clé primaire si cette dernière est supprimée.
CHECK est un mot clé associé à une condition qui doit être vérifiée pour chaque valeur insérée.

Remarque

Pour exécuter les commandes SQL données dans ce chapitre, on doit disposer d'un éditeur SQL permettant la saisie, l'exécution et l'affichage des résultats. La plupart des SGBD disposent de tels éditeurs.
Un exemple d'éditeur est fourni en annexe 2.

Exemples

La commande suivante permet de créer la table Article correspondant à la figure 5.5.

```
CREATE TABLE article
(Code_art     VARCHAR(20) PRIMARY KEY,
 des_art      VARCHAR (50) NOT NULL,
 PU           DECIMAL(8, 3) CHEK (PU > 0),
 Qte_stock    INT(5) DEFAULT 0 CHECK (qte_stock >= 0)
);
```

La commande suivante permet de créer la table Commande.

```
CREATE TABLE commande
(num_comm       VARCHAR(20) CONSTRAINT pk_cli PRIMARY KEY,
 date_comm      DATE NOT NULL,
 code_client     VARCHAR(20) REFERENCES client(code_client)
);
```

La clause «*définition_contrainte*» de la commande CREATE TABLE permet de définir une contrainte d'intégrité au niveau de la table. Elle doit être utilisée lorsque la contrainte ne s'applique pas à une seule colonne. Elle a la syntaxe suivante :

```
[CONSTRAINT contrainte]
{ PRIMARY KEY} (colonne1, colonne2, ...)
| FOREIGN KEY (colonne1, colonne2, ...)
        REFERENCES nom_table [(colonne1, colonne2, ... )]
        [ON DELETE CASCADE]
| CHECK (condition)}
```

Le mot clé CONSTRAINT est optionnel et sert à attribuer un nom à la contrainte.
PRIMARY KEY spécifie que les colonnes sont utilisées comme clé primaire.
FOREIGN KEY définit une contrainte d'intégrité référentielle relative à plusieurs colonnes. Le nom de la table précisé après le mot-clé REFERENCES est celui de la table mère. Les noms des colonnes sont ceux des colonnes vers lesquelles on se réfère.
CHECK est un mot clé associé à une condition qui doit être vérifiée pour chaque valeur insérée. Elle peut concerner ici plus qu'une colonne.

Exemple

La commande suivante permet de créer la table Détail commande correspondant à la figure 5.5.

```
CREATE TABLE detail_commande
        (num_comm  VARCHAR(20) REFERENCES commande,
         Num_ligne   INT(4),
         Qte_comm   INT(5) CHECK (qte_comm > 0),
        PRIMARY KEY (num_comm, num_ligne)
        );
```

5. Modification de la structure d'une base de données en mode commande

5.1. Modifier la structure d'une table en mode commande

La commande du langage SQL permettant de modifier la structure d'une table est la commande **ALTER TABLE**.
Elle permet les modifications suivantes sur la structure d'une table existante :

- Ajout de nouvelles colonnes,
- Modification de colonnes,

- Suppression de colonnes,
- Ajout de contraintes d'intégrité,
- Suppression de contraintes d'intégrité,
- Activation ou désactivation de contraintes d'intégrité.

La forme générale de cette commande est la suivante :

```
ALTER TABLE nom_table
        [ADD COLUMN définition_colonne]
        [ADD CONSTRAINT définition_contrainte]
        [MODIFY  définition_colonne]
        [DROP COLUMN nom_colonne ]
        [DROP CONSTRAINT nom_contrainte]
        [ENABLE | DISABLE nom_contrainte]
```

Le nom de la table doit correspondre à une table qui existe déjà.
L'option **ADD COLUMN** permet de rajouter des nouvelles colonnes à la table. La clause « *définition_colonne* » a la même syntaxe que celle utilisée dans la commande CREATE TABLE.

Exemple

La commande suivante permet de rajouter une colonne « email » à la table Client.

```
ALTER TABLE client ADD COLUMN
    (email VARCHAR(80)
    );
```

L'option **ADD CONSTRAINT** permet de rajouter une nouvelle contrainte à la table. La clause « *définition_contrainte* » a la même syntaxe que celle utilisée dans la commande CREATE TABLE

Exemple

Supposons que la table Article a été créée sans clé primaire. La commande suivante permet de préciser cette clé primaire.

```
ALTER TABLE article ADD CONSTRAINT PRIMARY KEY (code_article);
```

L'option **MODIFY** permet la modification de certaines caractéristiques d'une colonne existante. La clause « *définition_colonne* » a la même syntaxe que celle utilisée dans la commande CREATE TABLE

Exemple

Supposons qu'on souhaite élargir la taille de la colonne « email » qu'on vient de rajouter à la table client.

```
ALTER TABLE client  MODIFY (email    VARCHAR(100));
```

L'option **DROP COLUMN** permet de supprimer une colonne de la table. Cette option n'est pas prévue chez certaines versions de SQQL.

Exemple

Supposons qu'on souhaite supprimer la colonne « email » qu'on vient de rajouter à la table client.

```
ALTER TABLE client DROP COLUMN  email ;
```

L'option **DROP COSTRAINT** permet de supprimer une contrainte d'intégrité de la table.

Exemple

Supposons qu'on ne souhaite plus assurer l'identification des clients par le code article. La commande suivante permet de supprimer la clé primaire de la table Article.

```
ALTER TABLE article DROP CONSTRAINT PRIMARY KEY ;
```

L'option **DISABLE** permet de désactiver une contrainte d'intégrité. Lorsqu'une contrainte est désactivée, le SGBD ne va plus effectuer le contrôle imposé par cette contrainte.

Exemple

Supposons qu'on ne souhaite plus assurer provisoirement l'identification des clients par le code article. La commande suivante permet de désactiver la clé primaire de la table Article.

```
ALTER TABLE article DISABLE CONSTRAINT PRIMARY KEY ;
```

Lorsque cette clé primaire est référencée dans une ou plusieurs autres tables, il faut supprimer les clés étrangères dans les tables qui s'y réfèrent avant de procéder à la suppression de la clé primaire.

L'option **ENABLE** permet de réactiver une contrainte d'intégrité. Lorsqu'une contrainte est réactivée, le SGBD va de nouveau effectuer le contrôle imposé par cette contrainte.

Exemple

Supposons qu'on souhaite de nouveau assurer l'identification des clients par le code article. La commande suivante permet de réactiver la clé primaire de la table Article.

```
ALTER TABLE article ENABLE CONSTRAINT PRIMARY KEY ;
```

5.2. Suppression d'une table en mode commande

La commande du langage SQL permettant de supprimer la structure d'une table est la commande **DROP TABLE**.

Elle permet de supprimer à la fois le contenu et la structure d'une table existante.

La syntaxe de cette commande est la suivante :

```
DROP TABLE nom_table
```

Le nom de la table doit correspondre à une table qui existe déjà.

Exemple

La commande suivante permet de SUPPRIMER la table Client.

```
DROP TABLE client;
```

RETENONS

✔ Vous avez appris dans ce chapitre comment créer et modifier les différentes structures d'une base de données. Pour réaliser cette tâche, deux modes sont possibles : mode assisté et mode commande

✔ Le mode assisté est spécifique à chaque SGBD. A travers une interface graphique conviviale, l'utilisateur peut créer et modifier les différentes composantes d'une base de données en remplissant certains champs, en cochant d'autres et en effectuant des choix dans des listes déroulantes

✔ Le mode commande a comme principal avantage l'indépendance par rapport au SGBD. La forme générale des trois commandes permettant de créer et de modifier la structure d'une base de données (CREATE, ALTER et DROP) est la même dans tous les SGBD qui utilisent SQL.

LECTURE

Structured Query Language

Structured Query Language (**SQL**), ou langage structuré de requêtes, est un langage informatique (de type requête) normalisé (standard) destiné à interroger ou à manipuler une base de données relationnelle avec :

- Un langage de définition de données (**LDD**, ou en anglais DDL, Data definition language) qui permet de modifier la structure de la base de données.
- Un langage de manipulation de données (LMD, ou en anglais DML, Data manipulation language) qui constitue la partie la plus courante et la plus visible de SQL, permettant de consulter et modifier le contenu de la base de données.
- Un langage de contrôle de données (LCD, ou en anglais DCL, Data control language) qui permet de gérer les privilèges, c'est-à-dire les utilisateurs et les actions qu'ils peuvent entreprendre sur la Base de Données.

Bref historique

C'est Edgar F. Codd qui en juin 1970 rédigea l'article « A Relational Model of Data for Large Shared Data Banks » ("Un modèle de données relationnel pour de grandes banques de données partagées") dans la revue Communications of the ACM (Association for Computing Machinery). Ce modèle a été rapidement admis comme modèle définitif pour les systèmes de gestion de base de données (SGBD). Un langage, **Structured English Query Language ("SEQUEL")** (langage d'interrogation structuré en anglais) a été développé par IBM pour mettre en œuvre le modèle de Codd.

Le langage SQL (Structured Query Langage) est une évolution SEQUEL développé en 1976 par IBM comme langage de recherche. Cette évolution a, entre autres, été de supprimer les fonctions multivalué de SEQUEL qui faisait l'objet jusqu'en 1993 de brevet de Pick System, d'où un coût prohibitif.

En 1979, Relational Software, Inc. (actuellement Oracle Corporation) présenta la première version commercialement disponible de SQL, rapidement imité par d'autres fournisseurs. Malgré le succès du langage SQL qui a suivi, Edgar F. Codd dénoncera cet outil qu'il considère comme une interprétation incorrecte de ses théories.

SQL a été adopté comme recommandation par l'Institut de normalisation américaine (ANSI) en 1986, puis comme norme internationale par l'ISO en 1987 sous le nom de ISO/CEI 9075 - *Technologies de l'information* - Langages de base de données - SQL.

La norme internationale SQL est passée par un certain nombre de révisions :

Année	Nom	Appellation	Commentaires
1986	ISO/CEI 9075:1986	SQL-86 ou SQL-87	Édité par l'ANSI puis adopté par l'ISO en 1987.
1989	ISO/CEI 9075:1989	SQL-89 ou SQL-1	Révision mineure.
1992	ISO/CEI 9075:1992	SQL-92 ou SQL2	Révision majeure.
1999	ISO/CEI 9075:1999	SQL-99 ou SQL3	Expressions rationnelles, requêtes récursives, déclencheurs, types non-scalaires et quelques fonctions orientées objet (les deux derniers points sont quelque peu controversés et pas encore largement implémentés).
2003	ISO/CEI 9075:2003	SQL:2003	Introduction de fonctions pour la manipulation XML, « window functions », ordres standardisés et colonnes avec valeurs auto-produites (y compris colonnes d'identité).

*Extrait d'un article de **Wikipédia**, l'encyclopédie libre.*

EXERCICES

Exercice 1

En utilisant le mode assistant, créer la base de données de l'exercice 1 du chapitre 4.

Exercice 2

En utilisant le mode assistant, créer la base de données de l'exercice 2 du chapitre 4.

Exercice 3

Soit la table T ayant la structure suivante :

T (t1, t2, t3, t4)

En supposant que t1 et t3 sont de type numérique et que t2 est de type alphanumérique et que t4 est de type date, écrire la commande SQL permettant de créer la table T.

Exercice 4

Soit la table P ayant la structure suivante :

P (p1, p2, t1)

En supposant que p1 et p3 sont de type alphanumérique et que t1 est une clé étrangère qui se réfère à la table **T**, écrire la commande SQL permettant de créer la table P.

Exercice 5

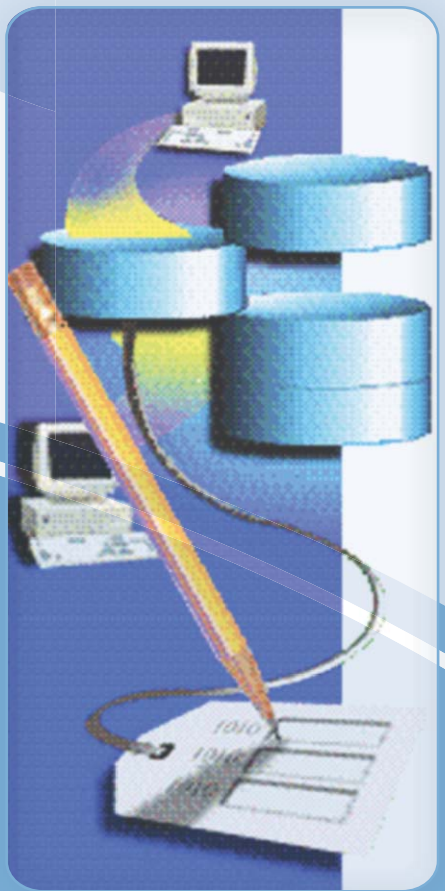
Rajouter à la table **T** une contrainte consistant à vérifier que t3 est toujours inférieure à 100.

Partie 3

MANIPULATION ET SECURISATION DE BASES DE DONNÉES

Chapitre 6

Manipulation d'une Base de Données



Objectifs

Ce chapitre à comme principaux objectifs de permettre à l'élève de

- Mettre à jour une base de données.
- Découvrir les commandes SQL pour l'ajout de nouvelles données, la modification de données déjà existantes et la suppression de celles qui ne sont plus utiles.

Plan :

1. Introduction

Après avoir présenté, au chapitre 5, les éléments permettant d'exploiter des outils logiciels pour créer et mettre à jour la structure d'une base de données ainsi que les commandes SQL pour la création et la modification de cette structure nous allons décrire dans ce chapitre comment manipuler le contenu d'une base de données.

La manipulation de données consiste à effectuer les opérations suivantes sur les tables de la Base de Données :

- insertion de nouvelles lignes dans une table
- modification de lignes existantes dans une table
- suppression de lignes d'une table
- consultation, ou recherche de lignes existantes dans une ou plusieurs tables

Généralement les trois premières opérations constituent la mise à jour de données.

Comme pour la définition des données, la manipulation de données peut être effectuée selon deux modes :

- le mode commande : les quatre types de commandes citées ci-dessus sont for mulés en utilisant le langage SQL (LMD).
- le mode assisté : les mêmes opérations citées ci-dessus peuvent être effectuées en utilisant une interface graphique dépendant du SGBD.

Le premier mode est plutôt destiné aux programmeurs alors que le deuxième est à la portée des utilisateurs peu avertis.

Dans ce qui suit, nous allons commencer par présenter d'abord le mode assisté.

Une forme simplifiée du mode commande sera ensuite présenté.

2. Base de Données exemple

Tout au long de ce chapitre, les exemples seront illustrés au moyen d'une Base de Données, simplifiée, relative à la gestion commerciale d'une entreprise de vente en gros, utilisant la messagerie électronique.

Les principaux sujets qui vont nous intéresser, dans la suite, sont :

- les **Articles** (ou produits) commercialisés
- les **Clients** permanents
- les **Commandes** faites par ces clients
- les **Détails** des articles commandés par les clients.

Les articles sont commandés par les clients à travers des commandes.

Un client peut commande un ou plusieurs articles dans la même commande. L'ensemble d'articles d'une commande constitue les détails de cette dernière.

Un client ne peut commander un article qu'une seule fois dans une commande.

Chaque article est identifié par un code (clé primaire). Il est décrit par sa désignation (ou son nom), son prix unitaire et sa quantité disponible en stock.

Chaque client est identifié par un code (clé primaire). Il est décrit par son nom, son prénom, son adresse postale, son numéro de téléphone, son adresse électronique, le chiffre d'affaires de l'année en cours (cumul des montants des commandes faites par le client) ainsi que le cumul de ses chiffres d'affaires des années antérieures.

Chaque commande est identifiée par un numéro (clé primaire). Elle est décrite par une date de commande ainsi qu'un code de client (clé étrangère).

Chaque détail de commande est identifié par la concaténation du numéro de commande et d'un numéro séquentiel d'une ligne de commande (ces deux propriétés constituent la clé primaire). Une ligne de commande (ou Détail commande) est décrite par un code de l'article commandé (clé étrangère) ainsi que la quantité commandée. Il est à noter que le numéro de commande, en plus de sa participation à la constitution de la clé primaire, doit être considéré en tant que clé étrangère.

En Adoptant la démarche de détermination de la structure de la Bases de Données présentée dans le chapitre 4, nous proposons la solution suivante :

Liste des colonnes :

Liste des colonnes							
Nom colonne	**Description**	**Type de données**	**Taille**	**Obligatoire**	**Valeur par défaut**	**Valeurs autorisées**	**Sujet**
code_art	Code Article qui sert comme identifiant	Caractère	20	O			Articles
des_art	Désignation (ou nom) du l'article	Caractère	50	O			Articles
PU	Prix Unitaire de l'article	Numérique	8,3	N		> 0	Articles
Qte_stock	Quantité disponible en stock de l'article	Numérique	30	N	0	≥ 0	Articles
code_client	Code Client qui sert comme identifiant	Caractère	20	O			Clients
Nom_client	Nom du client	Caractère	20	O			Clients
Prénom_client	Prénom du Client	Caractère	30	N			Clients
Adr_client	Adresse Postale du Client	Caractère	50	O			Clients
Tel_client	Numéro Téléphone du Client	Caractère	20	N			Clients
email_client	Adresse électronique du Client	Caractère	50	N			Clients
Chiffre_Affaires_Année_encours	Chiffre d'affaires de l'année en cours du Client	Numérique	10,3	N			Clients
Cumul_Chiffre_Affaires	Cumul des Chiffres d'affaires du Client	Numérique	12,3	N			Clients
num_comm	Numéro de Commande qui sert comme identifiant	Caractère	20	O			Commandes
date_comm	Date de la Commande	Date		O			Commandes
num_ligne	Numéro de Ligne de Commande qui sert comme identifiant quand on lui ajoute le Numéro de Commande	Numérique	4	O			Lignes de commandes
Qte_comm	Quantité Commandée d'un produit dans une commande	Numérique	5	N		>0	Lignes de commandes

Liste des tables :

Liste des tables		
Nom table	**Description**	**Sujet**
Client	elle contient la liste des clients de l'entreprise	Clients
Article	elle contient la liste des articles commercialisés	Articles
Commande	elle contient la liste des commandes faites par l'ensemble des clients	Commandes des Clients
Detail_Commande	elle contient la liste des articles commandés par l'ensemble des clients	Lignes des Commandes

Liens entre les tables

Table mère	Table fille	Clé primaire	Clé étrangère
Client	Commande	code_client	code_client
Article	Detail_commande	code_article	code_article
Commande	Detail_commande	Num_comm	Num_comm

Description textuelle des tables :

Article (code_art, des_art, PU, Qte_stock)
Client (code_client, Nom_client, Prénom_client, Adr_client, Tel_client, email_client, Chiffre_Affaires _Année_encours, Cumul_Chiffre_Affaires)
Commande (num_comm, date_comm, code_client#)
Détail_Commande (num_comm#, num_ligne, code_art#, Qte_comm)

Activité 6.1

En utilisant le mode assisté, créer la base de données dont la structure est présentée ci-dessous.

Table Article			
Colonne	**Type**	**Taille**	**Contrainte**
Code_Art	Texte	20	Clé primaire
Des_Art	Texte	50	Non nulle
PU	Numérique	(8,3)	(PU > 0)
Qte_stock	Numérique	5	(Valeur par défaut=0) et (Qte_stock >=0)

Table Client			
Colonne	**Type**	**Taille**	**Contrainte**
Code_client	Texte	20	Clé primaire
Nom_client	Texte	20	Non Nulle
Prenom_client	Texte	30	
Adr_client	Texte	20	Non Nulle
Tel_client	Texte	20	
Email_client	Texte	50	
Chiffre_affaires_annee_encours	Numérique	(10,3)	
Cumul_chiffre_affaires	Numérique	(12,3)	

Table Commande			
Colonne	**Type**	**Taille**	**Contrainte**
Num_comm	Texte	20	Clé primaire
Date_comm	Date		Non nulle
Code_client	Texte	20	Clé étrangère

Table Detail_commande			
Colonne	**Type**	**Taille**	**Contrainte**
Num_comm	Texte	20	Clé primaire
Num_ligne	Numérique	4	
Code_art	Texte	20	Clé étrangère
Qte_comm	Numérique	20	Qte_comm>0

3. Manipulation de données en mode assisté

Nous allons nous intéresser à quatre types d'opérations: l'insertion, la modification, la suppression et la recherche (requête) de données. L'insertion, la modification et la suppression constituent ce qu'on appelle les opérations de mise à jour.

3.1. Mise à jour de données

Dans la suite, nous allons nous intéresser à la manipulation d'une seule table.

3.1.1. Insertion de lignes

L'opération d'insertion de nouvelles données consiste à ajouter de nouvelles lignes dans une table dont la structure a été déjà créée dans la base.

Activité 6.2

Remplir les tables de la base créée dans l'activité précédente par les données indiquées dans les tableaux suivants :

Table Client							
Code	**Nom**	**Prénom**	**Adresse**	**Téléphone**	**Email**	**Chif. Af.**	**Cumul**
S0010	Gloulou	Rania	Sousse	66536658	Rania.gloulou@gnet.tn	15679,355	123765,540
T0122	Attia	Meriam	Tunis	66335297	Meriam.attia@topnet.tn	54987,210	339807,250
M5423	Ladhari	Youssef	Monastir	66481124	Youssef.Ladhari@planet.tn	22765,705	564700,590

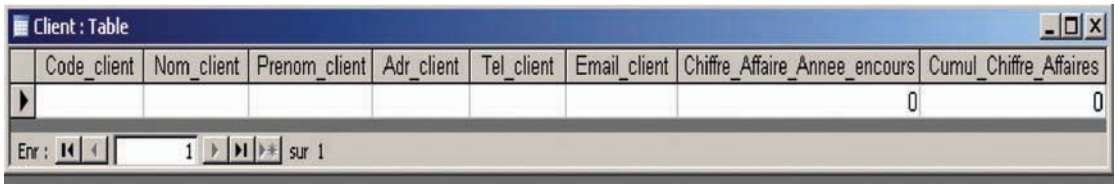
Table Commande		
Numéro	**Date**	**Code Client**
000100/2007	20/01/2007	T0122
002087/2007	30/03/2007	S0010
001123/2007	15/02/2007	M5423
003400/2007	24/04/2007	T0122

Table Article			
Code	**Description**	**Prix Unitaire**	**Quantité Stockée**
Dis312	Disquette 3"1/2		0
CD700	CDROM Vierge 700 Mo		0
ClaBil	Clavier Bilingue		0
SouPS2	Souris		0
AppPho	Appareil Photo Numérique		0
LanCard	Carte Réseau		0

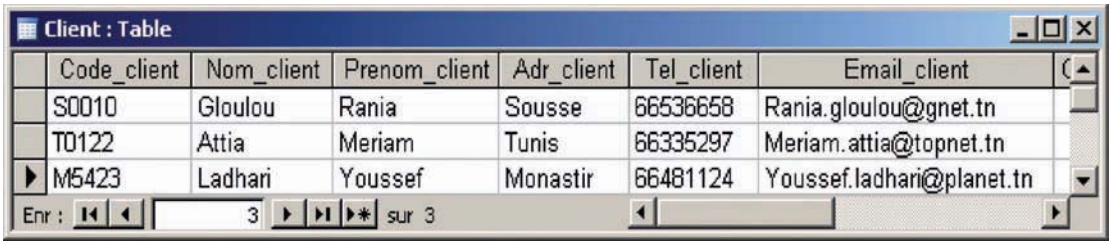
L'insertion de nouvelles lignes dans la table *Client* se fait en suivant les étapes suivantes :

1. Dans la fenêtre « Base de données », sélectionner la table *Client.*
2. Sélectionner le bouton . 

Une nouvelle fenêtre content un tableau ayant la structure de cette table s'affiche et vous permet introduire les données :

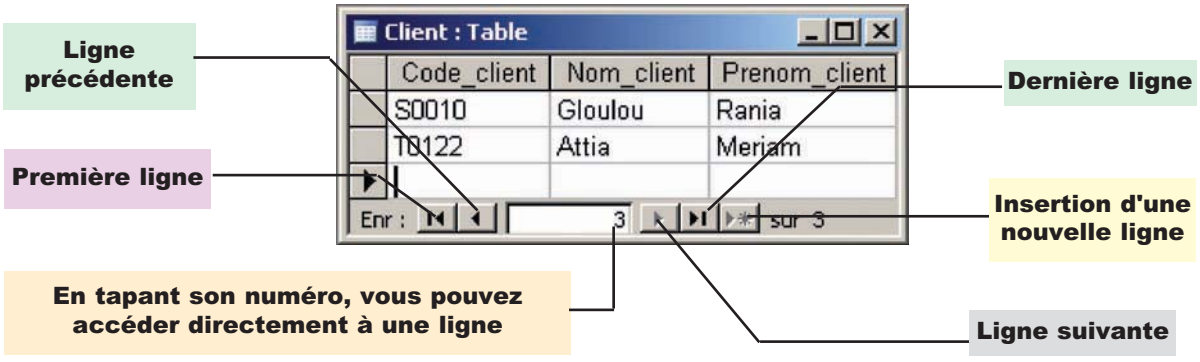


La saisie des données et la modification de la largeur d'une colonne se fait simplement comme dans un logiciel tableur. Les données sont introduites dans les différentes cellules et validées en tapant sur la touche «Entrée» ou sur une flèche de déplacement ou sur la touche tabulation.



Remarques

• Les boutons situés dans le coin inférieur gauche de la fenêtre vous permettent de naviguer aisément entre les lignes de la table :



• Pour ajouter une nouvelle ligne à la fin d'une table, vous pouvez cliquer sur le bouton de la barre d'outils. Le curseur se positionne dans la première colonne de la nouvelle ligne.
• Pour revenir à la structure de la table (mode création), vous pouvez cliquer sur le bouton afin de visualiser la structure de la table, d'ajouter une colonne ou de modifier les propriétés d'une colonne, etc.
• Les lignes sont affichées dans la table dans l'ordre de leur saisie. L'affichage des informations sera modifié par l'utilisation des requêtes ou/et des formulaires (à voir dans le prochain chapitre).
• Attention, la validation de l'insertion d'une ligne se fait automatiquement.

3.1.2. La suppression d'une ligne

Pour supprimer une ligne, il faut se positionner sur cette ligne à l'aide des boutons de navigation et presser la touche «**Suppr**» du clavier ou son équivalent. Cette suppression ne devient définitive qu'après confirmation.

Remarques

• Attention, la suppression confirmée d'une ligne ne peut être annulée !
• Pour supprimer plusieurs lignes adjacentes, les sélectionner puis presser la touche «**Suppr**» du clavier ou son équivalent.

3.1.3. La modification d'une ligne

Pour modifier une ligne, il faut se positionner sur cette ligne à l'aide des boutons de navigation et modifier les cellules concernées.

Remarques

Attention, la validation des modifications effectuées se fait automatiquement.

3.2. Recherche de données : requêtes

En mode assisté, la recherche peut se faire de deux façons :

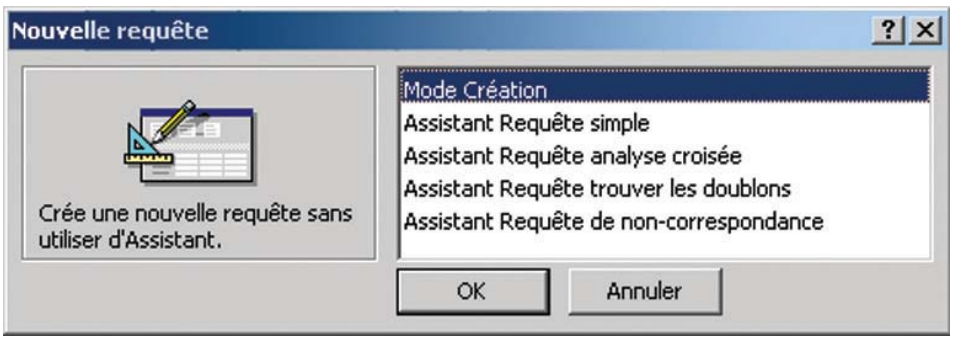
- **La recherche** selon un critère (sur une colonne) consiste à localiser la première ligne vérifiant une condition
- **Le filtrage** consiste à retrouver et afficher l'ensemble des lignes d'une table vérifiant une condition (pouvant porter sur plusieurs colonnes en utilisant les opérateurs logiques)

Une requête sert à exploiter les données contenues dans les tables.
Une requête permet, entre autres, de classer les informations dans l'ordre alphabétique d'un ou de plusieurs champs, de rechercher une donnée selon un ou plusieurs critères de sélection, d'effectuer des calculs, etc.
Le résultat d'une requête peut servir éventuellement de source pour **un formulaire** (présentation des données des champs à l'écran), pour **un état** (impression). Ces éléments seront développés dans le chapitre 7.
Une requête correspond à des ordres de programmation enregistrés au format SQL. Les ordres d'une requête peuvent être mémorisés et exécutés à tout moment. A chaque exécution de la requête, l'ensemble des données de la table est analysé avant d'en afficher le résultat.

3.2.1. Démarche de création d'une requête

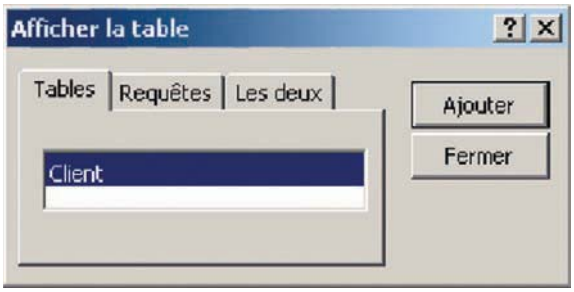
1. Sélectionnez l'onglet « **Requête** »
2. Cliquez sur le bouton 

3. Une boîte de dialogue apparaît : cliquez sur l'option Mode Création.

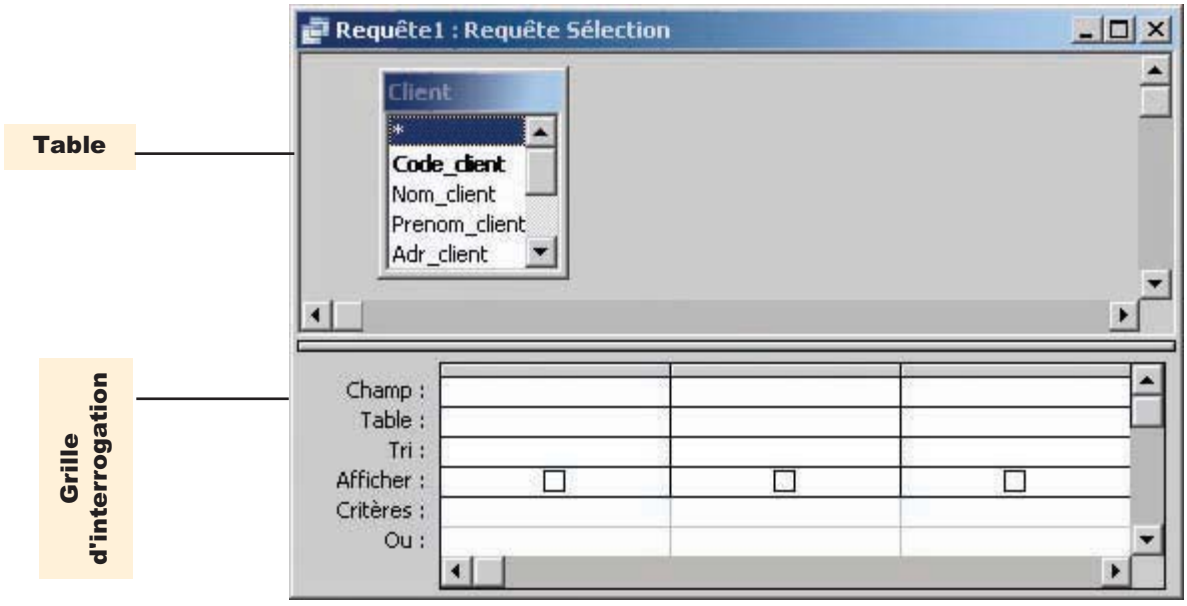


4. Cliquez sur le bouton OK.
5. Une boîte de dialogue apparaît : cliquez sur le nom de la table à insérer, puis appuyez sur le bouton Ajouter.

Répétez cette opération si une autre table doit être ajoutée.



6. Lorsque vous avez sélectionné toutes les tables nécessaires à la création de votre requête, cliquez sur le bouton *Fermer*.
L'écran de création de la requête affiche la ou les tables sélectionnées en haut de la fenêtre. Une grille d'interrogation permet de définir l'ordre des tris, les critères de **filtre** (ou condition), les calculs, ...



7. Dans la grille d'interrogation, vous sélectionnez les champs que vous voulez voir apparaître dans l'affichage de la requête ou que vous utiliserez pour classer, filtrer les lignes.
Vous définissez éventuellement :
- Les tris (ordre croissant, décroissant)
- Les critères de filtre
- Les formules de calcul
- Une propriété pour la requête ou pour des colonnes affichées.

3.2.2. Choix de colonnes à afficher dans une requête

Pour afficher une colonne dans une requête, il faut cliquer sur la colonne, tout en maintenant le bouton de gauche enfoncé, amenez la colonne sur la grille du bas et relâchez le bouton ou double cliquer sur la colonne.

Remarque

Pour afficher toutes les colonnes d'une ligne, vous sélectionnez le champ *. Dans l'exemple ci-dessous, la sélection du caractère * affiche dans la zone champ le terme **Client**.*

3.2.3. Visualisation du résultat d'une requête

Pour visualiser le résultat de votre requête, vous cliquez sur le bouton  . Le résultat apparaît sous la forme d'une feuille de données. Sa présentation peut être modifiée (même principe que dans un tableur).

En cliquant sur le bouton  , vous revenez sur l'écran de création de votre requête et vous pouvez la modifier.

3.2.4. Enregistrement d'une requête

Pour enregistrer la requête, cliquez sur le bouton  et donnez-lui un nom.

Remarque

La «grille d'interrogation» des colonnes et de définition des critères génère automatiquement des ordres de programmation SQL qui permettent d'obtenir le résultat souhaité. La requête sauvegardée mémorise les lignes de programmation et non pas le résultat affiché. Ces commandes seront étudiées dans la partie relative au mode commande.

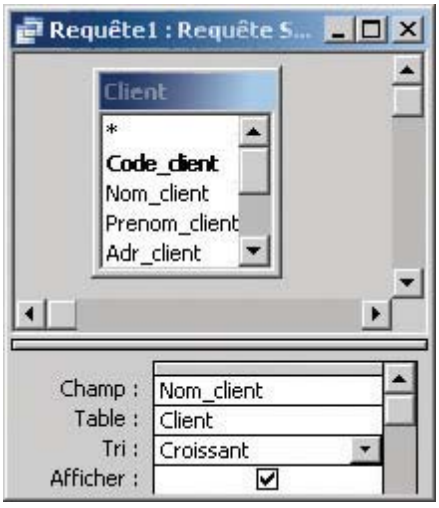
3.3. La définition des clés de tri

Les données fournies par une requête peuvent être classées selon une ou plusieurs clés. Pour trier le résultat selon les données d'une colonne, il faut l'indiquer sur la ligne Tri en dessous de la colonne concernée.
Plusieurs choix vous sont proposés :

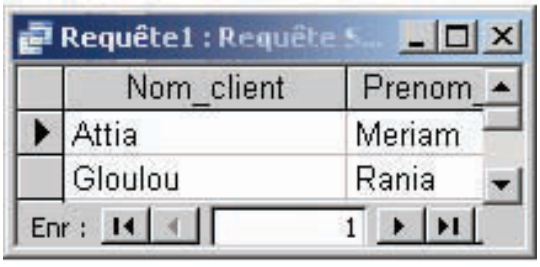
- *Croissant*
- *Décroissant*
- *(Non trié) : pour ne pas trier.*

Exemples 1

Le tri défini permet de classer les lignes de la table dans l'ordre croissant des noms des clients.



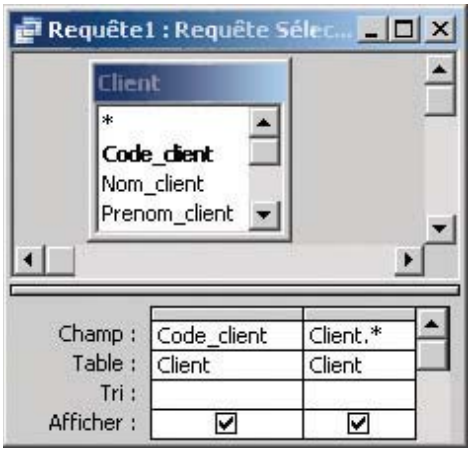
A l'exécution, vous obtenez ...



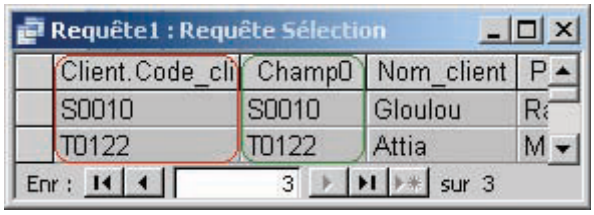
Pour combiner les clés de tri, il suffit de placer les colonnes dans l'ordre du tri souhaité et d'indiquer sous les colonnes concernées *Croissant ou Décroissant*. Les tris sont traités de **gauche à droite**.

Exemples 2

On désire afficher toutes les colonnes de la table **Client** selon l'ordre croissant des codes des clients.

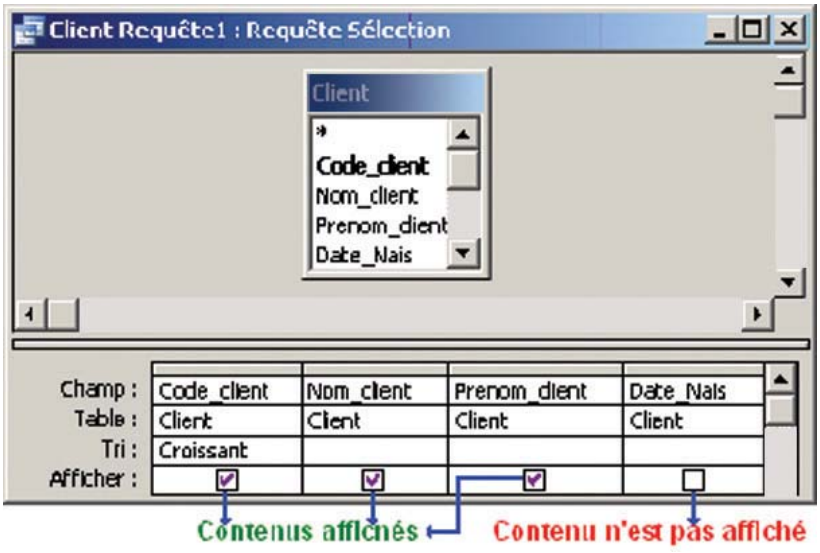


A l'exécution, vous obtenez ce résultat :



Remarquez que les codes des clients sont affichés deux fois car ils sont pris en compte dans les deux champs **Code_Client** et **Client**.*

Les colonnes de la table seront affichées selon l'ordre croissant des codes des clients. Le contenu de la colonne **Date_Nais** n'est pas affiché car l'option d'affichage n'est pas activée.



A l'exécution, vous obtenez ...

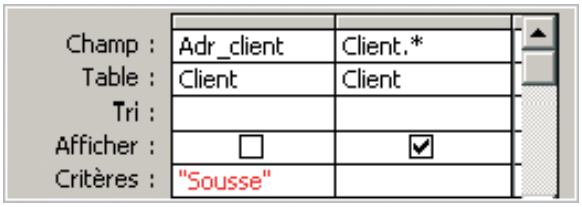
3.4. Les critères de filtres

Le résultat de la recherche ne concerne que les lignes qui vérifient le filtre (un ou plusieurs critères de sélection).

3.4.1. Recherche vérifiant un filtre relatif à une colonne de type texte

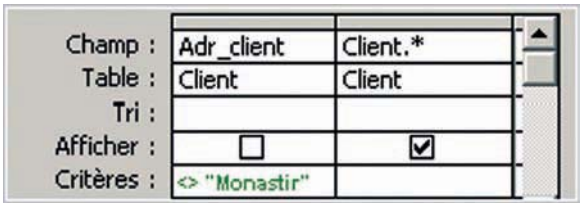
Exemples 1

Pour sélectionner uniquement les lignes ayant le mot Sousse dans la colonne **Adr_client**, il faut écrire Sousse sous le champ **Adr_client** et sur la ligne critère.



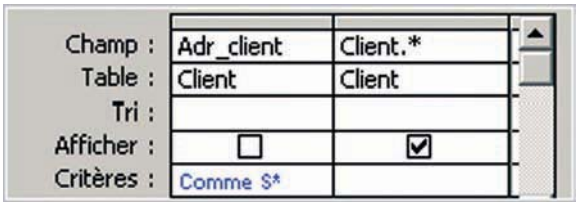
Exemples 2

Il est possible d'exclure les lignes contenant un certain *mot* dans une colonne. Il faut écrire **<>** ou **pas** devant le terme à exclure.



Exemples 3

Il est possible de sélectionner toutes les valeurs d'une colonne commençant par certains caractères. Le critère S* sélectionne tous les mots commençant par cette lettre. Le caractère* représente un nombre de caractères quelconque après la lettres S.



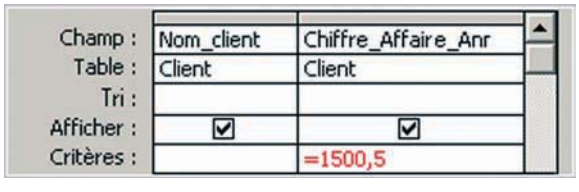
Remarque

- Le caractère * peut se placer à différents endroits :
  - S* : le texte commence par la lettre S.
  - *re : le texte se termine par re.
  - *m* : le texte contient la lettre m.
- Le caractère * s'applique uniquement aux champs texte.
- Lorsque vous tapez un critère dans un champ Texte, certains SGBD ne font pas la distinction entre les majuscules et les minuscules. Par exemple, les critères «fruit» et « Fruit » sont vus identiquement.
- Attention aux caractères accentués, «peche» et «pêche» sont différents.

3.4.2. Recherche vérifiant un filtre relatif à une colonne de type numérique

Exemple 1

Il est possible de sélectionner les colonnes numériques contenant une certaine valeur.

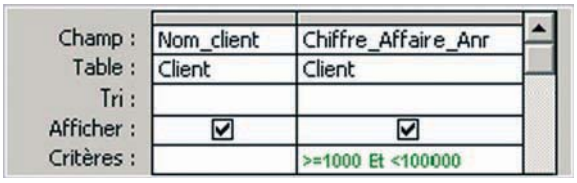


Le critère = 1500,5 sélectionne tous les champs contenant la valeur 1500,5
Les opérateurs de comparaison usuels ci-dessous peuvent être appliqués :

supérieur à	inférieur à	supérieur ou égal à	inférieur ou égal à	différent de	égal
>	<	>=	<=	<>	=

Exemple 2

Il est également possible de sélectionner des valeurs contenues dans un intervalle de valeurs.
Un intervalle de valeurs s'obtient par :
>= 1000 et <100000



Remarques

- Le critère >=3000 **et** <=5000 peut s'écrire entre 3000 et 5000
- Le **ou** est également utilisable dans ce genre de critère.

Par exemple, le critère <2000 **ou** >5000 permet d'afficher les champs contenant des valeurs plus petites que 2000 ou plus grandes que 5000.

3.4.3. Recherche vérifiant un filtre relatif à une colonne de type date

Les opérateurs de comparaison et leur utilisation sont les mêmes que pour les colonnes de type numérique.

Exemple

Le critère utilisé affiche touts les clients dont la date de naissance est supérieure au 01/01/1970.

Remarque

Certains SGBD rajoutent automatiquement des # autour de la date.

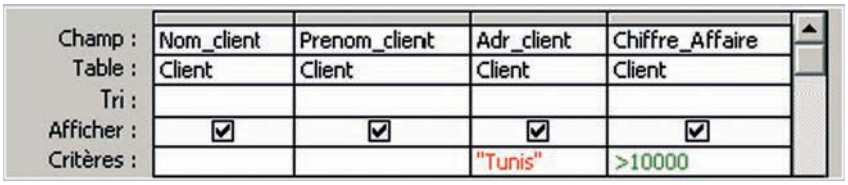
3.4.4. La combinaison de critères

Il est possible d'utiliser plusieurs critères de sélection, il suffit dans ce cas de les placer dans les différentes lignes et colonnes de la grille.

Si les critères sont placés sur la **même ligne**, *ils sont liés par un* **ET** (une ligne est affichée si les deux critères sont remplis);
Si les critères sont placés sur des **lignes différentes**, *ils sont liés par un* **OU** (une ligne est affichée si l'un des deux critères est rempli).

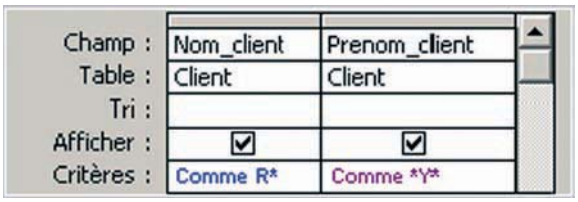
Exemple 1

Les critères utilisés permettent d'afficher les clients de Tunis dont le chiffre d'affaires de l'année en cours est supérieur à 10000.



Exemple 2

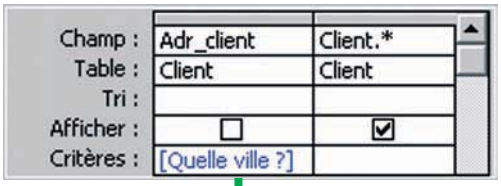
On désire afficher les clients dont le nom commence par « R » ou le prénom contient le caractère « Y ».



3.4.5. La définition d'une requête paramétrée

Il est possible que le critère d'une requête soit introduit dans une fenêtre de dialogue lors de son exécution.
Pour cela, une question écrite entre des crochets **[ ]** est rédigée sur la ligne critère.

Exemple 2



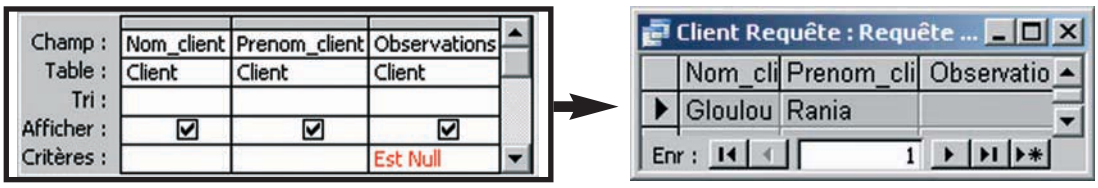
A l'exécution, une fenêtre de dialogue s'ouvre et permet d'introduire le critère de sélection …

3.4.6. Autres critères …

Il est possible en utilisant le critère **est null** ou **=null** de sélectionner les colonnes «vides».

Exemple 1

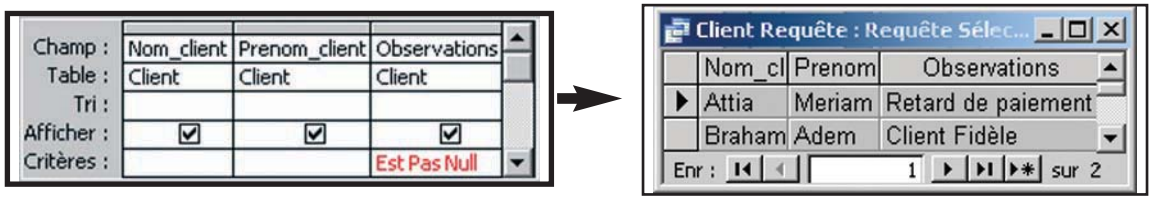
On souhaite afficher les clients pour lesquels aucune observation n'a été saisie.



Contrairement au cas précédent, le critère **est pas null** ou **<>null** permet d'afficher toutes les lignes à colonnes contenant une donnée.

Exemple 2

On souhaite afficher uniquement les clients pour lesquels des observation ont été saisies.



3.5. Les formules

Dans certains SGBD, les intitulés des colonnes peuvent être utilisés comme des variables.

Quand une requête est exécutée, les lignes d'une table sont lues les unes après les autres. A chaque «lecture» d'une ligne, le contenu de ses diverses colonnes est mémorisé dans ces variables.

Exemple :

A la lecture de la première ligne, la colonne **Code_Client** contiendra **S0010**; la colonne **Nom_Client** contiendra **Gloulou**;...

A la lecture de la deuxième ligne, la colonne **Code_Client** contiendra **T0122**; la colonne **Nom_Client** contiendra **Attia**; ...

Cette propriété des colonnes sert à la création de formules permettant d'obtenir une nouvelle colonne ou une nouvelle valeur.

Code_client	Nom_client	Prenom_client
S0010	Gloulou	Rania
T0122	Attia	Meriam
V0233	Braham	Adem

3.5.1. Exemple : concaténation

A partir des données alphanumériques de deux colonnes, il est possible de créer une nouvelle donnée en utilisant le symbole **&**. Les noms des colonnes sont écrits entre des crochets **[ ]** dans la formule.

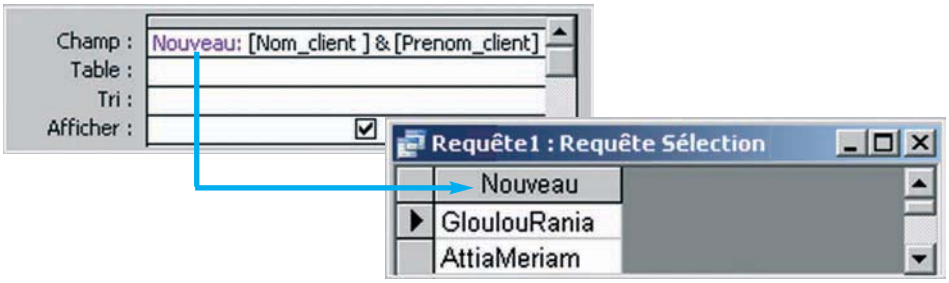
Exemple

Afin de concaténer les deux colonnes **Nom_Client** et **Prenom_Client**, suivez les étapes décrites au niveau du paragraphe 3.2.1 (Démarche de création d'une requête) puis rédigez sur la ligne *Champ* la formule suivante : **[Nom_client]&[Prenom_client]**

L'intitulé **Expr1** est créé automatiquement par le logiciel.

Remarques

• Il est possible de choisir l'intitulé de la nouvelle colonne en l'indiquant avant la formule.



• Pour séparer les colonnes par un espace, il faut écrire :

[Nom_client]&" "&[Prenom_client]

3.5.2. Les champs calculés

Les valeurs numériques contenues dans une table peuvent être utilisées dans les requêtes pour obtenir de nouvelles valeurs.

Une colonne calculée est composée de deux parties :

- le nom donné à cette formule (qui sera le nom de cette nouvelle colonne et qui sera utilisé comme libellé dans l'en-tête de la colonne);
- un calcul utilisant les différents opérateurs mathématiques +, -, /, *, et les ()

Exemple

Pour obtenir le prix total de chaque article en stock, il faut écrire la formule suivante (expression) : **Prix total:[PU]*[Qte_stock]**, où **Prix total:** est le nom donné à la nouvelle colonne qui sera utilisé dans la feuille de données et où les noms des colonnes de la table utilisés dans la formule sont écrit entre des crochets **[ ]**.

Champ :	Code_art	PU	Qte_stock	Prix total: [PU] * [Qte_stock]
Table :	Article	Article	Article	
Tri :				
Afficher :	☑	☑	☑	☑

3.6. Les propriétés d'une colonne ou d'une requête

Il est possible de mettre des propriétés sur les colonnes affichées concernant les valeurs ou les dates. Par exemple, le contenu d'une colonne numérique est affiché avec 2 décimales, le contenu d'une colonne date est affiché en indiquant le nom du mois au complet.

Démarche à suivre

1. Dans la zone des critères, cliquez à côté du nom de la colonne et cliquer sur le bouton afin d'afficher la fenêtre des propriétés ;
2. Sélectionnez le format et le nombre de décimales pour l'affichage des valeurs.

Pour obtenir une requête qui affiche les 5 premières lignes correspondant aux critères de sélection, il est nécessaire de modifier les propriétés de la requête.

Démarche à suivre

1. Cliquer sur l'aire « grisée » située à côté de la table et cliquer sur le bouton afin d'afficher la fenêtre des propriétés de la requête.
2. Indiquer le nombre de lignes à afficher vis-à-vis de **premières valeurs**. Cela peut-être un nombre ou un pourcentage.

Remarque

Par défaut, toutes les lignes sont affichées. Pour revenir à la situation initiale, il faut simplement effacer le champ vis-à-vis de **Premières valeurs**.

3.7. Requête « sélection » basée sur des calculs

Dans une table, on trouve des informations relatives à des lignes de même nature (Exemple : 1 ligne = 1 employé). Toutefois on peut souvent en tirer d'autres informations, par exemple si la table contient l'adresse de chaque employé et son salaire, on doit pouvoir en déduire la liste de toutes les villes représentées dans la table ainsi que le salaire total des employés. On utilise dans ce cas des requêtes avec option de regroupement en activant la ligne dite « Opération ».

La ligne opération est activée ou désactivée en cliquant sur le bouton Σ. Elle permet principalement d'utiliser **des fonctions de statistique** suivantes :

- *Somme* : Totalise toutes les valeurs d'une colonne.
- *Moyenne* : Calcule la moyenne de toutes les valeurs d'une colonne.
- *Minimum* : Calcule la valeur la plus petite dans une colonne.
- *Maximum* : Calcule la valeur la plus grande dans une colonne.
- *Compte* : Affiche le nombre total des lignes répondant à un critère.

Exemple

La table ci-dessous contient les informations concernant les employés d'une entreprise.

T_Employés : Table

	NoEm	NomEmployés	PrénomEmployé	FonctEmployés	SalEmpl	NoPos	VilleEmpl
▶	1	Rochdi	Bouhlel	Vendeur	3560	2000	Nabeul
	2	Chokri	Braham	Directeur	8930	2013	Kairouan
	3	Jalel	Baya	Employée	3300	2014	Béja
	4	Brahim	Chamakhi	Secrétaire de direct	2800	2000	Nabeul
	5	Mohamed	Bouhlel	Mécanicien	4500	2013	Kairouan
	6	Sadok	Ben Abdallah	Manutentionnaire	3900	2014	Béja
	7	Mehdi	Amimi	Secrétaire	3100	2000	Nabeul
	8	Adel	Braham	Employé	4050	2400	Gafsa
	9	Ramzi	Gharbi	Mécanicien	2900	2017	Bizerte
	10	Jamel	Chatti	Vendeur	3600	2400	Gafsa
	11	Béchir	Boukadida	Chauffeur	4200	2000	Nabeul
	12	Moncef	Ghazel	Directeur	9450	2072	Sousse
	13	Nouri	Gloulou	Employé	4100	2000	Nabeul

A partir de cette table et en utilisant la ligne opération, il est possible d'obtenir ...

A. Avec la fonction **Somme** …

Champ :	SalEmployés
Table :	T_Employés
Opération :	Somme
Tri :	
Afficher :	☑

On obtient le montant total des salaires versés …

SommeDeSalEmployés
127830

B. Ou encore avec la fonction **Max**

Champ :	SalEmployés
Table :	T_Employés
Opération :	Max
Tri :	
Afficher :	☑

A l'exécution vous obtenez le salaire le plus élevé ....

MaxDeSalEmployés
9450

C. Avec la fonction **Compte**

Champ :	NoEmployés
Table :	T_Employés
Opération :	Compte
Tri :	
Afficher :	☑

A l'exécution vous obtenez le salaire le plus élevé ....

CompteDeNoEmployés
13

D. Le « calcul » peut être limité à certains critères en utilisant la fonction **où**

Champ :	FonctEmployés	SalEmployés
Table :	T_Employés	T_Employés
Opération :	Où	Moyenne
Tri :		
Afficher :	☐	☑
Critères :	"vendeur"	

Dans le calcul, seules les lignes concernant les vendeurs sont pris en compte.

A l'exécution, vous obtenez

MoyenneDeSalEmployés
3620

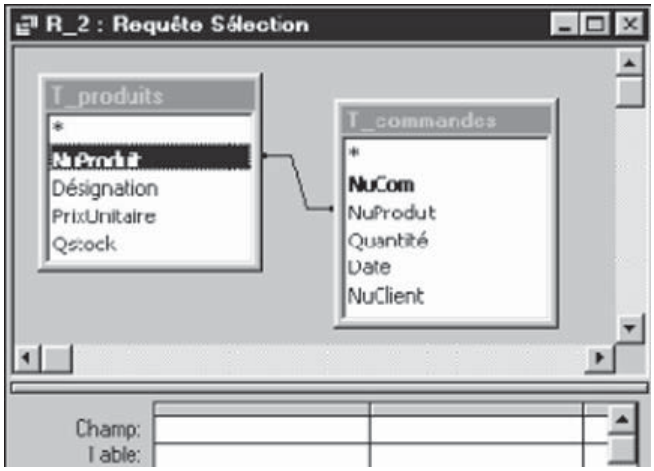
3.8. Les requêtes multi-tables

Par moments, on a besoin de rechercher des données appartenant à deux tables. Dans ce cas ces tables doivent avoir au moins une colonne en commun.
L'opération est dite dans ce cas **jointure**.

3.8.1. La création d'une jointure

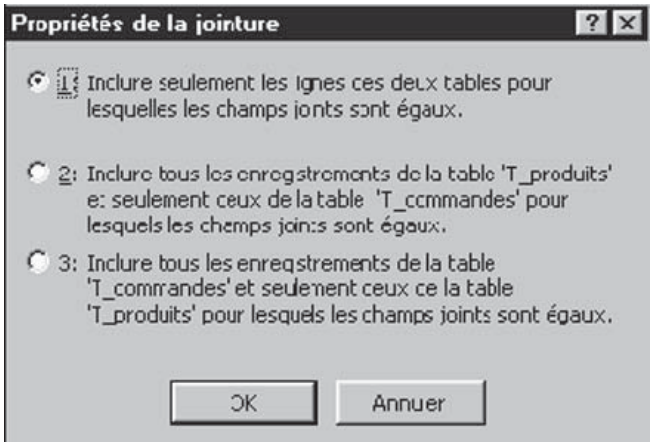
Démarche à suivre

Cliquez dans la première table sur le nom de la colonne à lier, laissez le bouton de gauche enfoncé et glissez ce nom vers le nom de la colonne correspondant situé dans la deuxième table.



3.8.2. La propriété de la jointure

Par défaut, la jointure inclus seulement les lignes des deux tables pour lesquelles les colonnes jointes sont égales. Selon le type de requête, il est nécessaire de modifier la propriété de la jointure pour voir apparaître toutes les lignes de l'une des deux tables.
Un double-clique sur la jointure permet d'ouvrir la fenêtre des propriétés ...



Certains SGBD offrent plusieurs options de jointure.
Ici, seules les lignes ayant une donnée identique dans les colonnes qui les lient seront affichées.

Exemple

Soient les tables :

T_Joueurs

T_Joueurs : Table

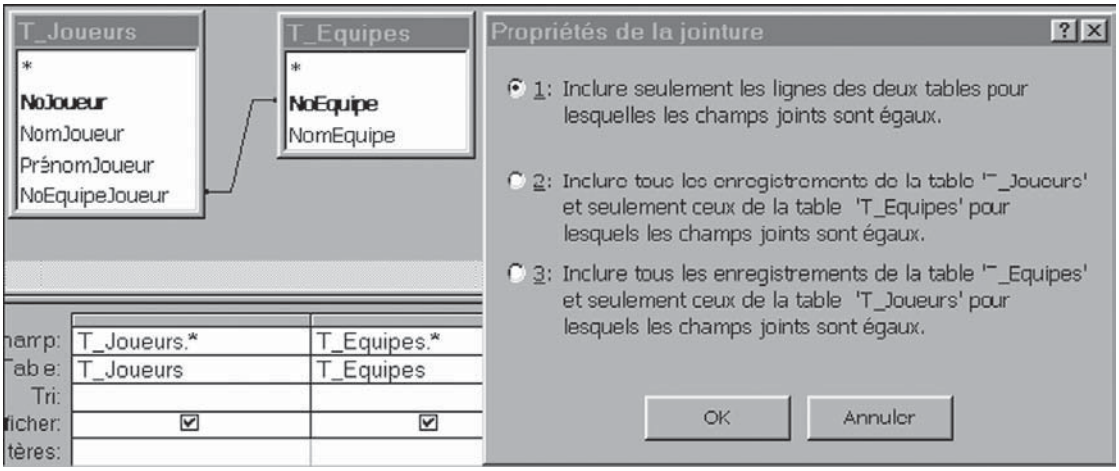
NoJoueur	NomJoueur	PrénomJoueur	NoEquipe
1	Jaziri	Imed	1
2	Salhi	Jamel	2
3	Boukadida	Ali	0
4	Baya	Riadh	1
uméroAuto)			0

T_Equipes

T_Equipes : Table

	NoEquipe	NomEquipe
	1	Cadet
	2	Junior
	3	Senior
▶	(NuméroAuto)	

Dans cet exemple, seuls les joueurs faisant partie d'une équipe sont pris en considération dans la requête.



La requête affiche :

Requête3 : Requête Sélection

	NoJoueur	NomJoueur	PrénomJoueur	NoEquipeJoueur	NoEquipe	NomEquipe
▶	1	Jaziri	Imed	1	1	Cadet
	2	Salhi	Jamel	2	2	Junior
	4	Baya	Riadh	1	1	Cadet

3.9. Echange de données avec un tableur

Dans le cas où les données à insérer dans une table existent déjà dans un fichier créé par un tableur, il est plus intéressant de les récupérer directement dans la table correspondante. Cette opération est appelée importation.

La même opération peut être effectuée dans le sens inverse : des données d'une tables peuvent être envoyées vers un fichier. Cette opération est dite exportation.

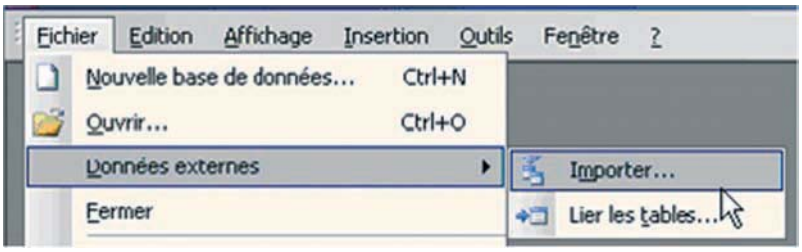
Nous allons présenter chacune de ces opérations dans les sections qui suivent.

3.9.1. Importation de données à partir d'un tableur

Il est possible d'importer, dans une base de données, une table de données réalisée à l'aide d'un logiciel tableur.

Démarche à suivre

1. Sélectionner dans le menu « **Fichier** », la commande «**Données externes**», puis la rubrique « **Importer** ».



2. Dans la fenêtre « **Importer** », sélectionner le type de fichiers (Microsoft Excel (*.xls) par exemple) et votre document.

Cliquer sur le bouton 

pour continuer.

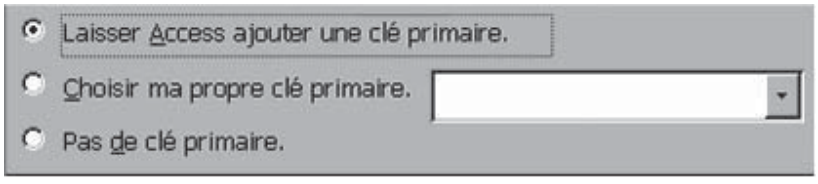
3. Une fenêtre « **Assistant** … » s'ouvre. Elle vous permet de sélectionner la feuille de votre document que vous voulez importer.

4. L'étape suivante permet d'indiquer que le contenu de la première ligne de la feuille de calcul correspondra aux noms des champs sous Access.

5. Les données de la feuille peuvent être mémorisés dans une nouvelle table, voir rajouter à une table Access qui a évidemment la même structure …

6. Dans cette étape, l'assistant vous demande si le premier champ correspondra à la clé primaire. Confirmer les options proposées …

7. Laisser Access gérer la clef primaire. Elle aura la propriété **NuméroAuto.**



8. Indiquer le nom de la table et cliquer sur le bouton. 

3.9.2. Exportation des données vers un tableur

Une table peut être exportée vers une feuille de calcul. La procédure est très simple.

Démarche à suivre

1. Sélectionner la table dont vous souhaitez exporter le contenu.

2. Dans le menu «**Fichier**», sélectionner la commande «**Exporter** ».

3. Dans la fenêtre «**Exporter** …», sélectionner le type de fichier et votre document. Cliquer ensuite le bouton. 

4. Manipulation de données en mode commande

Comme pour la manipulation des données en mode assisté, la manipulation des données en mode commande est composée de deux types de commandes : commandes de mise à jour d données et commande de consultation (ou recherche) de données.

4.1. Mise à jour de données

La mise à jour des tables d'une base de données relationnelle se fait à l'aide de trois commandes : la commande d'**insertion**, la commande de **modification** et la commande de **suppression** de lignes.

4.1.1. Insertion de lignes

L'opération d'ajout de nouvelles données consiste à insérer de nouvelles lignes dans une table dont la structure a été déjà créée dans la base.
La commande du langage SQL permettant d'insérer une ligne dans une table est la commande **INSERT**.
La forme générale de cette commande est la suivante :

```
INSERT INTO nom_table [ liste_Nom_colonne]
VALUES (liste_valeur)
```

• Le paramètre *liste_Nom_colonne* sert à préciser la liste des colonnes dans lesquelles va s'effectuer l'opération d'insertion. Cette liste peut être omise dans le cas où l'opération d'insertion concerne toutes les colonnes de la table. Dans ce cas, l'ordre des valeurs fournies par le paramètre *liste_valeur* doit être le même que celui des colonnes données pour la commande CREATE TABLE.
• Pour que l'opération d'insertion puisse être exécutée, les conditions suivantes doivent être respectées :

1. Les types des données de *liste_valeur* doivent être compatibles avec ceux des colonnes de la table,
2. Unicité des lignes (contrainte de clé primaire),
3. Caractère obligatoire associé à une colonne (clause NOT NULL),
4. Existence de la valeur dans une autre table quand il s'agit d'une clé étrangère (contrainte d'intégrité référentielle) : nécessité de respecter un certain ordre lors de l'insertion des lignes
5. Vérification d'une condition de validité (contrainte valeur : clause CHECK).

Remarque

Pour exécuter les commandes SQL données dans ce chapitre, on doit disposer d'un éditeur SQL permettant la saisie, l'exécution et l'affichage des résultats. La plupart des SGBD disposent de tels éditeurs.

Un exemple d'éditeur est fourni en annexe 2.

Exemples

Objectif :

Apprendre à ajouter des nouvelles lignes à une table.

Enoncé :

On souhaite insérer un nouveau client à la base de données.

On suppose dans la suite que les commandes SQL suivantes ont été exécutées pour créer les tables Client, Commande, Article et Détail_commande :

```
CREATE TABLE client(
                code_client   VARCHAR(20) PRIMARY KEY,
                Nom_client    VARCHAR (20) NOT NULL,
                Prénom_client       VARCHAR(30),
                Adr_client          VARCHAR (100) NOT NULL,
                Tel_client          VARCHAR (20),
                email_client  VARCHAR (50),
                Chiffre_Affaires_Année_encours DECIMAL(10,3),
                Cumul_Chiffre_Affaires DECIMAL(12,3)
                );
CREATE TABLE commande(
                num_comm   VARCHAR(20) PRIMARY KEY,
                date_comm  DATE NOT NULL,
                code_client   VARCHAR(20) REFERENCES client(code_client)
                );
CREATE TABLE article(
                code_art      VARCHAR(20) PRIMARY KEY,
                des_art       VARCHAR (50) NOT NULL,
                PU            DECIMAL(8, 3) CHECK (PU > 0),
                Qte_stock     INT(5) DEFAULT 0 CHECK (qte_stock >= 0)
                );
CREATE TABLE détail_commande(
        num_comm    VARCHAR(20) REFERENCES commande ON DELETE CASCADE,
                Num_ligne    INT(4),
        code_art   VARCHAR(20) REFERENCES article(code_art) ON DELETE CASCADE,
                Qte_comm    INT(5)  CHECK (qte_comm > 0),
                Constraint pk_detail PRIMARY KEY (num_comm, num_ligne)

                );
```

Question 1 :

Rajouter un nouveau client en précisant explicitement toutes les colonnes.

Exemple 1

On souhaite insérer un nouveau client en précisant explicitement toutes les colonnes. On doit donc préciser toutes les valeurs à insérer explicitement. La commande SQL permettant cette insertion est la suivante :

```
INSERT INTO Client
      (code_client, Nom_client, Prénom_client, Adr_client,
      Tel_client,email_client,Chiffre_Affaires_Année_encours,
                    Cumul_Chiffre_Affaires)
 ALUES (1000,'TOUNSI', 'Fares''25 Rue des roses 2036 Ariana',
       "(216)71850310", "Tounsi.Fares@edunet.tn", 0, 0);
```

Remarque

Compte tenu que chaque colonne, de la table Client, reçoit une valeur et que l'ordre de ces valeurs est le même que celui des colonnes, on aurait pu dans ce cas omettre de préciser ces colonnes. Dans ce cas, la commande devient :

```
INSERT INTO Client
 ALUES (1000,'TOUNSI','Fares','25 Rue des roses 2036 ariana',
       "(216)71850310", "Tounsi.Fares@edunet.tn", 0, 0);
```

Exemple 2

On souhaite toujours l'insertion d'une ligne dans la table Client, en précisant explicitement toutes les valeurs, mais cette fois dans un ordre différent que celui des colonnes. Dans ce cas on doit préciser la liste des colonnes de la table Client dans le même ordre que celui des valeurs. La commande SQL sera alors la suivante :

```
INSERT INTO Client
      (Nom_client, Prénom_client, code_client, Tel_client,
                   Adr_client, email_client)
            VALUES ('BENZARTI','Yasmine',1010,'98850310',
                '12 Av de la liberté 2035 Beja',
                 "Yasmine.Benzarti@is.rnu.tn");
```

Exemple 3

On souhaite insérer un nouveau client pour lequel on ne dispose que de valeurs relatives à certaines colonnes de cette table. Ces colonnes sont : code client, son nom et son adresse. Dans ce cas on doit donner les colonnes, qui sont concernées, dans le même ordre que ces valeurs. La commande SQL sera alors la suivante :

```
INSERT INTO Client (Nom_client, code_client, Adr_client)
          VALUES ('Banque Nationale Agricole',1020,
           '130 Bis Av Mohamed V 2080 Ben Arous');
```

Remarque

Les colonnes, de la table Client, qui sont absentes ne sont soumises à aucune des contraintes d'intégrité. Ces colonnes recevront une NULL au moment de l'insertion de la ligne.

Exemple 4

On souhaite l'insertion d'une ligne au sein de la table Client, en précisant explicitement les valeurs relatives à son nom et son adresse. La commande SQL sera alors la suivante :

```
INSERT INTO Client (Nom_client, Adr_client)
       VALUES ('Le Magasin Général',
     'Cité Mahragène 1004 El Manzah);
```

Remarque

Cette commande ne peut pas être exécutée à cause de la contrainte d'intégrité PRIMARY KEY qui est violée. Les attributs constituant la clé primaire doivent obligatoirement être présents.

Exemple 5

On souhaite l'insertion d'une ligne au sein de la table Client, en précisant explicitement les valeurs relatives à son nom, son prénom et son code. La commande SQL sera alors la suivante :

```
INSERT INTO Client (Nom_client, Prénom_client, code_client)
       VALUES ('SOUSSI','Nesrine',1040);
```

Remarque

Cette commande ne peut pas être exécutée à cause de la contrainte d'intégrité NOT NULL pour la colonne Adr_client qui n'a pas été respectée ici.

Exemple 6

On souhaite l'insertion d'une ligne au sein de la table Client, en précisant explicitement les valeurs relatives au code client, à son nom et à son adresse. La commande SQL sera alors la suivante :

```
INSERT INTO Client (Nom_client, code_client, Adr_client)
       VALUES ('Banque de l''Habitat', 1010,
         '65 Av Mongi Slim 1002 Tunis');
```

Remarque

Cette commande ne peut pas être exécutée à cause de la contrainte d'intégrité d'unicité de la clé primaire qui n'a pas été respectée. Le code client 1010 a été déjà attribué.

Exemple 7

On souhaite l'insertion d'une ligne au sein de la table Commande. La commande SQL sera alors la suivante :

```
INSERT INTO Commande
       VALUES (50010,'09/05/2007',2050);
```

Remarque

Cette commande ne peut pas être exécutée à cause de la contrainte d'intégrité FOREIGN KEY (contrainte d'intégrité référentielle) qui est violée. En effet, le client ayant le code égal à 2050 n'existe pas dans la table Client.

Exemple 8

On souhaite l'insertion d'une ligne au sein de la table Article. La commande SQL sera alors la suivante :

```
INSERT INTO Article
VALUES (7050'Informatique 4ème Année Mathématiques, Sciences et Techniques',500,0);
```

Remarque

Cette commande ne peut pas être exécutée à cause de la contrainte d'intégrité CHEK qui n'a pas été respectée. En effet, le prix unitaire d'un article doit respecter la condition qui a été fixée à strictement supérieur à zéro.

4.1.2. Suppression de lignes

L'opération de suppression de données consiste à éliminer une ou plusieurs lignes existantes à partir d'une table.

La commande du langage SQL permettant de supprimer des lignes d'une table est la commande DELETE.

La forme générale de cette commande est la suivante :

```
DELETE FROM nom_table
    [WHERE condition]
```

- La clause WHERE est optionnelle. Si elle est absente, dans ce cas, toutes les lignes de la table concernée seront supprimées.
- Le paramètre *condition* qui apparaît dans la clause WHERE sert à indiquer la condition qui doit être vérifiée par les lignes à supprimer.
- En cas où la commande de suppression ne respecte pas une contrainte d'intégrité référentielle, elle ne peut pas aboutir.
- La suppression d'une ligne appartenant à une table donnée peut entraîner la suppression d'autres lignes appartenant à d'autres tables lorsqu'il existe des contraintes d'intégrité référentielles de suppression en cascade : utilisation de la clause ON DELETE CASCADE dans la définition des clés étrangères.

Exemples

Objectif :

Apprendre à supprimer des données appartenant à une base de données.

Enoncé :

On souhaite supprimer des lignes appartenant à des tables de la base de données.

On suppose, qu'avant la suppression, les commandes SQL suivantes ont été exécutées :

```
INSERT INTO Client
     VALUES (1000,'TOUNSI','Ridha'
          '25 Rue Ibn Khouldoun 2036 Ariana',
               '(216)71850310','Tounsi.Ridha@edunet.tn');
```

```
INSERT INTO Client
    VALUES (1010, 'BENZARTI',' Ahmed',
    '125 Av de la liberté 2035 Le Kram', '(216) 98850310',
         ' Ahmed.Benzarti @ ensi.rnu.tn');
```

Exemple 1

On souhaite supprimer tous les clients.

La commande SQL permettant de répondre à la question est donc la suivante :

```
DELETE FROM Client ;
```

Remarque

Dans la base de données, il n'existe que les deux lignes créées par les commandes INSERT qui précèdent la suppression. La commande DELETE sera donc exécutée et toutes les lignes de la table clients seront supprimées.

Exemple 2

On suppose maintenant, qu'avant la suppression, les commandes SQL suivantes ont été exécutées :

```
INSERT INTO Client
         VALUES (1000,'TOUNSI','Ridha','25 Rue Ibn Khouldoun
              2036 Ariana',
              '(216)71850310','Tounsi.Ridha@edunet.tn');
INSERT INTO Client
    VALUES (1010, 'BENZARTI',' Ahmed',
         '125 Av de la liberté 2035 Le Kram','(216) 98850310',
              ' Ahmed.Benzarti @ ensi.rnu.tn');

INSERT INTO Commande
         VALUES (50010,'09/05/2007',1010);
INSERT INTO Commande
         VALUES (70150,'20/07/2007',1000);
```

La commande SQL permettant de répondre à la question est la même que la précédente. Elle sera donc :

```
DELETE FROM Client;
```

Remarque

Contrairement à l'exemple précédent, la commande DELETE ne sera pas exécutée et aucune ligne de la table Client ne sera supprimée à cause de la contrainte d'intégrité référentielle qui est violée. En effet, on ne peut pas supprimer des clients qui ont encore des commandes dans la base.

Les exemples qui suivent consistent à supprimer des données respectant certaines conditions.

Exemple 3

Supprimer les détails des commandes ayant une quantité commandée (qte_comm) inférieure à 150 unités.

On suppose, qu'en plus des commandes SQL de l'exemple précédent, les commandes suivantes ont été exécutées :

```
INSERT INTO Article
    VALUES (7050,'Informatique 4ème Année Mathématiques,
         Sciences et Techniques',4000,500);
INSERT INTO Article
    VALUES (1020,'4 Stylos Feutres Pointes Fines : Rouge,
         Jaune,Vert et Bleu',1500,3000);
INSERT INTO detail_Commande VALUES (50010,1,7050,200);
INSERT INTO detail_Commande VALUES (50010,2,1020,300);
INSERT INTO detail_Commande VALUES (70150,1,7050,120);
INSERT INTO detail_Commande VALUES (70150,2,1020,4000);
```

La commande SQL permettant de répondre à la question est donc la suivante :

```
DELETE FROM detail_Commande
         WHERE qte_comm < 150 ;
```

Remarque

La commande DELETE sera exécutée et la ligne de la table *Detail_Commande* qui contient les valeurs (70150,1, 7050,120) sera supprimée.

Exemple 4

Supprimer l'article ayant un code égal à 7050. La commande SQL permettant de répondre à la question est la suivante :

```
DELETE FROM Article
      WHERE code_art = 7050;
```

Remarque

La commande DELETE sera exécutée :

- La ligne de la table *Article* qui contient les valeurs (70150, 'Informatique 4ème Année Mathématiques, Sciences et Techniques', 4000,500) sera supprimée puisqu'elle vérifie la condition de suppression.
- De même la ligne de la table *detail_Commande* qui contient les valeurs (50010,1,7050,200) sera également supprimée puisqu'elle doit respecter la contrainte d'intégrité référentielle (ON DELETE CASCADE) définie au moment de la création de cette table.

Exemple 5

Supprimer la commande ayant un numéro (num_comm) égal à 50010. La commande SQL permettant de répondre à la question est la suivante :

```
DELETE FROM Commande
     WHERE num_comm = 50010;
```

Remarque

La commande DELETE ne sera pas exécutée à cause de la contrainte d'intégrité référentielle qui risque d'être violée. En effet, on a encore, dans la table *detail_Commande*, une ligne qui contient les valeurs (50010,2,1020,300) et qui se réfère à la commande numéro 50010.

4.1.3. Modification de lignes

L'opération de mise à jour de données consiste en la modification de colonnes, appartenant à une table, qui contiennent des données qui ne sont plus conformes à la réalité.

La commande du langage SQL permettant de modifier le contenu d'une table est la commande **UPDATE**.

La forme générale de cette commande est la suivante :

```
UPDATE nom_table
SET  Nom_colonne1 = Expression1 [ , Nom_colonne2 = Expression2 …]
[WHERE condition]
```

- La modification peut toucher une ou plusieurs colonnes.
- Les valeurs des colonnes identifiées par $Nom\_colonne_1$, $Nom\_colonne_2$, …, $Nom\_colonne_n$ sont modifiées dans toutes les lignes qui vérifient la condition de la clause WHERE. En cas où cette dernière est absente, toutes les lignes de la table sont mises à jour.
- Les paramètres donnés par $expression_1$, $expression_2$ …, $expression_n$ peuvent faire référence aux anciennes valeurs de la ligne à mettre à jour. Le résultat d'évaluation de chaque $expression_i$ remplace l'ancienne valeur de la colonne identifiée par $Nom\_colonne_i$.
- Les principaux cas pour lesquels l'ordre UPDATE ne peut aboutir sont les mêmes que ceux qui ont étés indiqués pour l'ordre INSERT.

Exemples

Objectif :

Apprendre à mettre à jour des données qui ne sont plus conformes à la réalité.

Enoncé :

On souhaite modifier contenu de colonnes d'une table appartenant à la base de données.

On suppose dans la suite que les commandes SQL suivantes ont été exécutées :

```
INSERT INTO Client
     VALUES (1000,'TOUNSI','Ridha','25 Rue Ibn Khouldoun 2036
          Ariana','(216)71850310','Tounsi.Ridha@edunet.tn');
INSERT INTO Commande
          VALUES (70150, '20/07/2007',1000);
INSERT INTO Article
     VALUES (7050,'Informatique 4ème Année Mathématiques,
          Sciences et Techniques',4000,500);
INSERT INTO Article
     VALUES (1020,'4 Stylos Feutres Pointes Fines : Rouge,
          Jaune, Vert et Bleu',1500,3000);
INSERT INTO detail_Commande VALUES (50010,1,7050,200);
INSERT INTO detail_Commande VALUES (50010,2,1020,300);
INSERT INTO detail_Commande VALUES (70150,1,7050,120);
INSERT INTO detail_Commande VALUES (70150,2,1020,4000);
```

Les deux exemples qui suivent consistent à mettre à jour toutes les lignes d'une table.

Exemple 1

Initialiser à zéro le cumul des chiffres d'affaires de tous les clients. Cette question porte donc sur la modification du contenu d'une seule colonne. Elle concerne toutes les lignes de la table *Client*.

La commande SQL permettant de répondre à la question est donc la suivante :

```
UPDATE Client
     SET Cumul_Chiffre_Affaires = 0;
```

Remarque

Aucune contrainte d'intégrité n'est violée par cette commande. Elle sera donc exécutée et toute la colonne relative au cumul des chiffres d'affaires de la table client sera égale à zéro.

Exemple 2

Supposons que nous sommes le 31 Décembre et qu'on doit à cette date ajouter au cumul des chiffres d'affaires le montant du chiffre d'affaires de l'année en cours. On peut par la même occasion prévoir la préparation de la colonne relative aux chiffres d'affaires des clients pour l'année prochaine. Cette question porte sur la modification du contenu de deux colonnes (colonne des chiffres d'affaires de l'année en cours et celle du cumul des chiffres d'affaires). Comme pour le cas précédent, elle concerne toutes les lignes de la table Client.

La commande SQL permettant de répondre à la question est donc la suivante :

```
UPDATE Client
     SET Cumul_Chiffre_Affaires = Cumul_Chiffre_Affaires +
          Chiffre_Affaires_Année_encours,
          Chiffre_Affaires_Année_encours  = 0;
```

Les exemples qui suivent consistent à mettre à jour des lignes d'une table qui vérifient une certaine condition.

Exemple 3

Mettre à jour le prénom du client dont le code est égal à 1020. Cette question porte donc sur la modification du contenu d'une seule colonne. Elle concerne des lignes, vérifiant une condition.

La commande SQL permettant de répondre à la question est donc la suivante :

```
UPDATE Client
     SET Nom_client = 'BEJI', Prénom_client = 'Fares'
     WHERE code_client = 2050;
```

Remarque

Aucune contrainte d'intégrité n'est violée par cette commande. Elle sera donc exécutée et toute la colonne relative au prénom des clients de la table client sera mise à jour et ce pour les lignes vérifiant la condition exigée.

Exemple 4

Mettre à jour, à NULL, les dates des commandes faites par le client dont le code est égal à 2050.

La commande SQL permettant de répondre à la question est donc la suivante :

```
     UPDATE Commande
          SET date_comm = NULL
          WHERE code_client = 2050;
```

Remarque

Cette commande ne peut pas être exécutée à cause de la contrainte d'intégrité (NOT NULL) relative à la date de commande qui n'est pas respectée.

Exemple 5

Mettre à jour la quantité en stock du produit dont le code est égal à 7050 en lui retranchant une quantité égale à 10.

La commande SQL permettant de répondre à la question est donc la suivante :

```
     UPDATE Article
          SET Qte_stock = Qte_stock - 10
          WHERE code_art = 7050;
```

Remarque

Cette commande ne peut pas être exécutée à cause de la contrainte d'intégrité relative à la quantité en stock (devant être > 0) qui n'est pas respectée. La quantité en stock initiale du produit en question est égale à zéro.

4.2. Recherche de données : requêtes

La recherche de données est une opération qui consiste en la recherche de données appartenant à la base. Cette recherche peut être réalisée en faisant référence à une ou à plusieurs tables. Elle peut se référer à une partie ou à la totalité des colonnes des tables concernées tout en étant, éventuellement, conditionnelle.
Une recherche peut consister à effectuer :

- Une **projection** sur certaines colonnes d'une table
- Une **sélection** sur certaines lignes d'une table
- Une **jointure** sur deux tables
- Toute **combinaison** de projection, sélection et jointure.

La commande permettant d'effectuer une recherche dans une base de données est la commande **SELECT**.

4.2.1. Recherche de colonnes à partir d'une table : Projection

Souvent, l'utilisateur n'a besoin que d'un sous-ensemble des colonnes d'une table. L'opération correspondante dans SQL s'appelle **projection**. Elle ne concerne qu'une seule table de la Base de Données.
Le résultat de la projection doit comporter au moins une colonne de la table. Cependant, toutes les lignes relatives à cette colonne feront parti du résultat.
La forme générale de la commande de projection est la suivante :

```
SELECT [DISTINCT] * / liste_Nom_colonne
FROM nom_table
```

- Le paramètre liste_*Nom_colonne* sert à préciser la liste des colonnes, que l'on veut voir affichées.
- Le paramètre liste_*Nom_colonne* peut être remplacé par * qui désigne toutes les colonnes de la table.
- L'ordre d'affichage du résultat donné par le contenu des colonnes est le même que celui indiqué par le paramètre liste_*Nom_colonne*.
- Dans le cas où le paramètre * est donné, l'ordre sera celui spécifié dans la structure de la table.
- Le paramètre DISTINCT sert à éliminer les lignes en double dans le résultat puisque la commande SELECT peut ramener des doublons si la clé primaire ne figure pasdans la liste des colonnes à afficher).
- Le paramètre nom_table sert à se référer à la table concernée par l'opération de sélection.
- Le résultat de la commande SELECT est une nouvelle table résultat de l'opération de projection de la table initiale.

- Par défaut, les colonnes de la table résultat portent les mêmes noms que ceux de la table de départ.
- Il est possible de donner aux colonnes de la table résultat des noms différents de ceux de la table d'origine. On doit dans ce cas donner une entête de colonne juste après son nom donné par le paramètre liste_*Nom_colonne*. Cette entête est généralement appelée Alias.

Exemples

Objectif :

Apprendre à rechercher des données relatives à une opération relationnelle de projection.

Enoncé :

On souhaite afficher des colonnes appartenant à la table Client.

Exemple 1

Donner les codes, noms et prénoms de tous les clients.
La commande SQL permettant de répondre à la question est la suivante :

```
SELECT code_client, Nom_client, Prénom_client
     FROM Client;
```

Remarque

Les colonnes demandées par la commande SELECT seront affichées avec comme entêtes des colonnes d'affichage les noms des colonnes de la Table Client.

Exemple 2

Donner les caractéristiques de tous les clients.

La commande SQL permettant de répondre à la question est la suivante :

```
SELECT *
     FROM Client;
```

Remarque

Toutes les colonnes de la Table Client seront affichées.

Exemple 3

Donner les codes, noms et prénoms de tous les clients. Au moment de l'affichage, les entêtes des colonnes doivent être respectivement 'Code du Client', 'Nom du Client' et 'Pénom du Client'.

La commande SQL permettant de répondre à la question est la suivante :

```
  SELECT code_client 'Code Client', Nom_client 'Nom Client',
                 Prénom_client 'Prénom Client'
         FROM Client;
```

Exemple 4

Donner les chiffres d'affaires de l'année en cours de tous les clients.

La commande SQL permettant de répondre à la question est la suivante :

```
SELECT Chiffre_Affaires _Année_encours
     FROM Client;
```

Remarque

La colonne relative aux chiffres d'affaires de l'année en cours sera affichée. En cas où plusieurs clients ont le même montant, on aura plusieurs lignes affichées avec ce même montant.

Exemple 5

Donner les chiffres d'affaires de l'année en cours de tous les clients. L'affichage d'un montant doit se faire une seule fois en cas d'égalité de certains montants.
La commande SQL permettant de répondre à la question est la suivante :

```
SELECT DISTINCT Chiffre_Affaires _Année_encours
     FROM Client;
```

Exemple 6

Donner la liste des produits et pour chacun calculer la valeur du stock.
La commande SQL permettant de répondre à la question est la suivante :

```
SELECT des_art 'Désignation Produit', PU 'Prix Unitaire',
Qte_stock 'Quantité Stock', PU*Qte_stock 'Valeur Stock'
FROM Article;
```

Remarque

La colonne résultat, relative à la valeur du stock, sera calculée à partir du prix unitaire et la quantité en stock.

4.2.2. Recherche de lignes à partir d'une table : Sélection

Souvent, l'utilisateur n'a besoin que d'un sous-ensemble des lignes d'une table. L'opération correspondante dans SQL s'appelle **sélection**. Elle ne concerne qu'une seule table de la Base de Données.
La différence par rapport à l'opération de projection c'est que le résultat est composé d'un sous-ensemble de lignes de la table et de toutes ses colonnes.
La sélection est souvent combinée avec la projection. Dans ce cas le résultat est un sous ensemble de lignes et de colonnes de cette table.
La forme générale de la commande de sélection est la suivante :

```
SELECT [DISTINCT] * / liste_Nom_colonne
FROM nom_table
WHERE condition
```

• Le paramètre condition sert à préciser le critère (ou prédicat) qui doit être vérifié par les lignes à afficher. Ce prédicat est donné sous la forme d'une expression logique.
• Si le résultat de l'évaluation de l'expression logique est vrai, pour une ligne, celle-ci fera partie du résultat.
• Dans l'expression logique, on peut utiliser en particulier :
  1. Les opérateurs de comparaison : =, >, <, >=, <= et <>.
  2. L'opérateur BETWEEN pour les intervalles de valeurs, bornes incluses.
  3. L'opérateur IN pour les listes de valeurs.
  4. L'opérateur IS NULL et IS NOT NULL pour les valeurs indéterminées.
  5. L'opérateur LIKE pour filtrer une chaîne de caractères.
  6. Les opérateurs logiques : AND, OR et NOT.

Exemples

Objectif :

Apprendre à rechercher des données relatives à une opération relationnelle de sélection.

Enoncé :

On souhaite afficher certaines lignes appartenant à la table Client.

Exemple 1

Donner la liste des clients qui ont atteint ou dépassé un chiffre d'affaires de 100000 Dinars de l'année en cours.
La commande SQL permettant de répondre à la question est la suivante :

```
SELECT *
    FROM Client
    WHERE Chiffre_Affaires _Année_encours >= 100000;
```

Remarque

Toutes les colonnes de la table Client sont concernées, mais seules les lignes qui vérifient la condition indiquée feront parti du résultat.

Exemple 2

On demande la même liste que l'exemple précédent sauf qu'on ne s'intéresse qu'aux colonnes relatives aux noms et aux prénoms.
La commande SQL permettant de répondre à la question est la suivante :

```
SELECT Nom_client 'Nom du Client',
          Prénom_client 'Prénom Du Client'
    FROM Client
    WHERE Chiffre_Affaires _Année_encours >= 100000;
```

Exemple 3

Donner la liste des clients qui ont un cumul des chiffres d'affaires compris entre 500000 et 800000 et dont le code appartient à l'ensemble (1000, 1010, 1050, 1090) et qui ont, une adresse qui commence par la lettre 'T' ou qui a comme deuxième caractère la lettre 'O'.

La commande SQL permettant de répondre à la question est la suivante :

```
SELECT *
FROM Client
WHERE Cumul_Chiffre_Affaires BETWEEN 500000 AND 800000
AND code_client IN (1000, 1010, 1050, 1090)
AND (Adr_client IS NULL OR Adr_client LIKE 'T%' OR
     Adr_client LIKE '_O%');
```

Remarques

- Les valeurs 500000 et 800000, utilisées avec l'opérateur BETWEEN, sont les bornes de l'intervalle.
- Les valeurs de l'ensemble de l'opérateur IN doivent être données entre ().
- Le caractère % utilisé avec l'opérateur LIKE sert à remplacer une liste de caractères qui peuvent être quelconques (y compris vide). Le caractère _, par contre, sert à remplacer un seul caractère. Pour remplacer deux caractères, il suffit de mettre deux fois successives le caractère _.

4.2.3. Recherche de données à partir de plusieurs tables : Jointure

Dans certaines situations, l'utilisateur a besoin de données provenant de plusieurs tables différentes de la Base de Données. Il s'agit dans ce cas d'une opération de **jointure**.

La forme générale de la commande de Jointure est la suivante :

```
SELECT [DISTINCT] * / liste_Nom_colonne
    FROM nom_table1 [alias1], nom_table2 [alias2] …
    WHERE condition
```

- L'opération relationnelle de jointure réalise une liaison entre deux tables en se basant sur l'égalité des valeurs entre l'une des colonnes de chaque table.
- Un alias permet de donner un nom synonyme, abrégé, à une table. Il permet d'alléger l'écriture de la commande SELECT en cas d'ambiguïté.
- Le paramètre condition sert, particulièrement, à préciser la condition de jointure. Cette condition ne doit pas être confondue avec celle indiquée pour l'opération relationnelle de sélection.
- La condition de jointure doit porter sur les colonnes en commun aux tables utilisées pour l'opération de jointure (voir contrainte d'intégrité référentielle : clé primaire - clé étrangère).

Exemples

Objectif :

Apprendre à rechercher des données relatives à une opération relationnelle de jointure.

Enoncé :

On souhaite afficher certaines lignes appartenant aux tables Client et Commande.

Exemple 1

Donner la date de chaque commande ainsi que le nom et le prénom du client qui a fait la commande.
La commande SQL permettant de répondre à la question est la suivante :

```
SELECT Nom_client, Prénom_client, date_comm
  FROM Client, Commande
  WHERE Client.code_client = Commande.code_client;
```

Remarques

• Pour lever toute ambiguïté sur les noms des colonnes, nous avons précisé les noms des tables comme préfixes, mais il est possible de simplifier en utilisant des alias. La commande devient alors :

```
SELECT Nom_client, Prénom_client, date_comm
  FROM Client X, Commande Y
  WHERE X.code_client = Y.code_client;
```

• Si un même client peut faire plusieurs commandes, le même jour, et qu'on ne veut pas avoir de doublons dans le résultat, la commande devient :

```
SELECT DISTINCT Nom_client, Prénom_client, date_comm
  FROM Client X, Commande Y
  WHERE X.code_client = Y.code_client;
```

Exemple 2

Donner le numéro et la date de chaque commande ainsi que les quantités commandées et les désignations des articles correspondants.
La commande SQL permettant de répondre à la question est la suivante :

```
SELECT C.num_comm, date_comm, Qte_comm, des_art
    FROM Commande C, détail_commande D, article A
    WHERE (C.num_comm=D.num_comm) AND (D.code_art=A.code_art);
```

Remarque

Trois relations ont été utilisées pour répondre à ce besoin. Deux opérations de jointure seront donc exécutées : la première entre les tables Commande et détail_commande et la seconde entre le résultat obtenu et la table Article.

Exemple 3

Même besoin que l'exemple précédent sauf que les commandes qui nous intéressent sont celles qui sont relatives à des produits dont la quantité en stock est suffisante. La commande SQL permettant de répondre à la question est la suivante :

```
SELECT C.num_comm, date_comm, Qte_comm, des_art
     FROM Commande C, détail_commande D, article A
     WHERE (Qte_stock <= Qte_comm AND C. num_comm = D. num_comm)
                         AND (D. code_art = A. code_art);
```

Remarque

Dans cet exemple, à la condition de Jointure, nous avons ajouté une condition de sélection.

Exemple 4

Donner la liste des clients qui ont atteint les mêmes montants des chiffres d'affaires de l'année en cours. La commande SQL permettant de répondre à la question est la suivante :

```
SELECT DISTINCT X.Nom_client,   X.Prénom_client,
  X.Chiffre_Affaires_Année_encours
  FROM Client X, Client Y
  WHERE X.Chiffre_Affaires_Année_encours =
         Y.Chiffre_Affaires _Année_encours);
```

Remarque

Dans cet exemple nous avons utilisé deux fois la même table. Il s'agit d'une opération d'auto-jointure. Dans ce cas l'utilisation d'un **alias** dans la commande SELECT devient indispensable.

4.2.4. Recherche de données avec Tri

Certaines requêtes ont besoin de rechercher des données de la Base et d'avoir un résultat qui soit trié selon un ordre croissant ou décroissant des valeurs d'une ou de plusieurs colonnes. La forme générale de la commande de recherche est la suivante :

```
SELECT [DISTINCT] * / liste_Nom_colonne
   FROM nom_table1 [alias1] [, nom_table2 [alias2] …]
   [WHERE condition]
   ORDER BY Nom_colonne1 [ASC / DESC] [, Nom_colonne2 [ASC / DESC] …]
```

- La clause ORDER BY sert à exiger le classement (ou tri) du résultat.
- Le classement peut se faire selon un ordre croissant (ASC) ou décroissant (DESC) des valeurs d'une ou de plusieurs colonnes. Ces dernières sont données par le paramètre Nom_colonne, et elles doivent obligatoirement figurer parmi celles indiquées par le paramètre liste_Nom_colonne.

• Par défaut c'est l'ordre croissant (ASC) qui est pris en considération.
• Les éléments dont les valeurs des colonnes concernées par le tri sont indéterminées (NULL) sont donnés ensemble de manière successive.
• Le tri peut être associé à n'importe quelle opération de recherche (Projection, Sélection et Jointure).

Exemples

Objectif :

Apprendre à rechercher des données relatives à une opération relationnelle quelconque et de donner le résultat de manière triée.

Enoncé :

On souhaite afficher certaines lignes appartenant aux tables Client et Commande. Le résultat doit être trié.

Exemple 1

Donner la date de chaque commande ainsi que le nom et le prénom du client qui a fait la commande. Le résultat doit être classé par ordre croissant des noms des clients et par ordre décroissant des dates de commandes pour le même client. La commande SQL permettant de répondre à la question est la suivante :

```
SELECT Nom_client, Prénom_client, date_comm
     FROM Client X, Commande Y
     WHERE X.code_client = Y.code_client
     ORDER BY Nom_client, date_comm DESC;
```

Remarque

On aurait pu, pour la clause ORDER BY, remplacer les noms des colonnes par leur rang dans le paramètre liste_Nom_colonne. La commande devient alors :

```
SELECT Nom_client, Prénom_client, date_comm
   FROM Client X, Commande Y
   WHERE X.code_client = Y.code_client
   ORDER BY 1, 3 DESC;
```

4.2.5. Utilisation des fonctions de calculs dans les opérations de recherche (fonctions Agrégat)

Certaines requêtes ont besoin de faire un certain nombre de calculs sur les lignes recherchées. Pour cela, SQL offre certaines fonctions standards de calcul appelées **fonctions Agrégat**. Ces fonctions ne peuvent être utilisées qu'avec la commande SELECT et en dehors de la clause WHERE.

Les fonctions de calcul offertes par SQL sont les suivantes :

• La fonction COUNT qui sert à compter le nombre de lignes du résultat obtenu par la commande SELECT.

- La fonction SUM qui sert à faire la somme des valeurs d'une colonne dont le type de données est numérique.
- La fonction MIN qui sert à déterminer la valeur minimale d'une colonne.
- La fonction MAX qui sert à déterminer la valeur maximale d'une colonne.
- La fonction AVG qui sert à déterminer la moyenne (average) des valeurs numériques d'une colonne.

Dans la suite nous donnons quelques exemples simples d'utilisation de ces fonctions.

Exemples

Objectif :

Apprendre à utiliser les fonctions de calcul dans des commandes de recherche de données.

Enoncé :

On souhaite calculer certaines données appartenant aux tables Client et Commande.

Exemple 1

Donner le nombre de clients qui existent dans la base.
La commande SQL permettant de répondre à la question est la suivante :

```
SELECT COUNT(*)
FROM Client ;
```

Exemple 2

Donner le nombre de clients qui ont fait des commandes.
La commande SQL permettant de répondre à la question est la suivante :

```
SELECT COUNT(DISTINCT (code_client))
     FROM Commande;
```

Remarque

Le mot clé DISTINCT a été utilisé dans la commande SELECT pour éliminer les doublons vu qu'un même client peut avoir plusieurs Commandes.

Exemple 3

Donner le nombre de clients qui ont un cumul des chiffres d'affaires supérieur à 100000 Dinars. La commande SQL permettant de répondre à la question est la suivante :

```
SELECT COUNT(*)
     FROM Client
     WHERE Cumul_Chiffre_Affaires > 100000 ;
```

Exemple 4

Donner la somme des chiffres d'affaires de l'année en cours de l'ensemble des clients qui ont un cumul des chiffres d'affaires supérieur à 100000 Dinars. La commande SQL permettant de répondre à la question est la suivante :

```
SELECT SUM(Chiffre_Affaires _Année_encours)
 FROM Client
 WHERE Cumul_Chiffre_Affaires > 100000;
```

Exemple 5

Pour l'ensemble des clients qui ont un cumul des chiffres d'affaires supérieur à 100000 Dinars, donner le plus haut et le plus bas chiffre d'affaires de l'année en cours ainsi que leur moyenne. La commande SQL permettant de répondre à la question est la suivante :

```
SELECT MIN(Chiffre_Affaires_Année_encours) 'CA Min',
       MAX(Chiffre_Affaires_Année_encours) 'CA Max',
       AVG(Chiffre_Affaires _Année_encours) 'CA Moyen'
  FROM Client
  WHERE Cumul_Chiffre_Affaires > 100000 ;
```

RETENONS

- ✓ La manipulation d'une base de données consiste à insérer, supprimer, modifier ou rechercher des lignes. Elle peut se faire en mode assisté ou en mode commande.

  Le mode assisté se fait en utilisant une interface graphique d'un SGBD alors que le mode commande consiste à écrire explicitement des commandes conformément à la syntaxe du langage SQL.

- ✓ L'insertion d'une ligne, en SQL, se fait par l'intermédiaire de la commande **INSERT**.
- ✓ La suppression de lignes, en SQL, se fait par l'intermédiaire de la commande **DELETE**. Elle permet de supprimer plusieurs lignes qui vérifient la condition spécifiée.
- ✓ La modification de lignes, en SQL, se fait par l'intermédiaire de la commande **UPDATE**. Elle permet de modifier plusieurs lignes qui vérifient la condition spécifiée.
- ✓ La recherche de lignes, en SQL, se fait par l'intermédiaire de commande **SELECT**. Elle peut consister à effectuer des opérations de projection, de sélection, de jointure ou toute combinaison de ces dernières. Le résultat peut être trié.

  On parle de **Projection** si la recherche est relative à un sous ensemble de colonnes d'une table.

  On parle de **Sélection** si la recherche est relative à un sous ensemble de lignes d'une table vérifiant une certaine condition.

  On parle de **Jointure** si la recherche est relative à deux tables ayant au moins une colonne en commun. La condition de jointure doit porter sur ces colonnes communes.
- ✓ La clause **ORDER BY**, utilisée dans la commande **SELECT**, permet de trier le résultat obtenu.
- ✓ SQL propose certaines fonctions, dites fonctions agrégat : **SUM, COUNT, MIN**, **MAX** et **AVG**.

APPLICATION

Soit la base de données relative à l'exploitation des chambres d'un hôtel définie comme suit :

```
Chambre (Num_Chambre, Prix, Nbr_Lit, Nbr_Pers, Confort, Equ)
Client (Num_Client, Nom, Prenom, Adresse)
Reservation (Num_Client#, Num_Chambre#, Date_Arr, Date_Dep)
```

Exemples :

Table « Chambre »					
Num_Chambre	**Prix**	**Nbr_Lit**	**Nbr_Pers**	**Confort**	**Equ**
10	80	01	02	WC	Non
20	100	02	02	Douche	Non
25	180	03	03	Bain	TV
...	...	...	...	...	...

Table « Client »			
Num_Client	**Nom**	**Prénom**	**Adresse**
1000	Sahli	Mohamed	Sousse
1001	Chatti	Saleh	Tunis
...	...	...	...

Table « Réservation »			
Num_Client	**Num_Chambre**	**Date_Arr**	**Date_Dep**
1000	20	28/12/2006	01/01/2007
1001	10	01/01/2007	
...	...	...	...

Exprimer les requêtes suivantes en SQL :

1. Les numéros de chambres avec TV.
2. Les numéros de chambres et leurs capacités.
3. La capacité théorique d'accueil de l'hôtel.
4. Le prix par personne des chambres avec TV.
5. Les numéros des chambres et le numéro des clients ayant réservé des chambres pour le 25/12/2006.
6. Les numéros des chambres coûtant au maximum 80 Dinars ou ayant un bain et valant au maximum 120 Dinars.
7. Les noms, prénoms et adresses des clients dont les noms commencent par «RA»
8. Le nombre de chambres dont le prix est entre 85 et 120 Dinars.
9. Les noms des clients n'ayant pas fixé la date de départ.

Correction

1.
```
Select Num_Chambre
From Chambre
Where Confort = 'TV' ;
```

2.
```
Select Num_Chambre, Nbr_Pers
From Chambre ;
```

3.
```
Select Sum(Nbr_Pers)
From Chambre ;
```

4.
```
Select Prix/Nbr_Pers, Num_Chambre
From Chambre
Where Equ = 'TV'
```

5.
```
Select Num_Chambre, Num_Client
From Reservation
Where Date_Arr <= '25/12/2006'
And (Date_Dep > '25/12/2006' Or Date_Dep is Null)
```

6.
```
Select Num_Chambre
From Chambre
Where Prix<='80' Or (Confort='bain' And Prix <='120');
```

7.
```
Select Nom
From Client
Where Nom like 'RA%';
```

8.
```
Select Count(Num_Chambre)
From Chambre
Where Prix between 85 and 120;
```

9.
```
Select Nom
From Reservation R, Client C
Where Date_Dep is Null And R.Num_Client = R. Num_Client;
```

EXERCICES

Exercice 1

En utilisant le mode assisté du SGBD disponible :

1. Créer une base de données nommée « **Base1** ».
2. Créer la table « **Fournisseur** » ayant la structure ci-dessous (La colonne **Num_fsr** est la clé primaire de la table).

Table « Fournisseurs »		
Nom du champ	**Type de champ**	**Description**
Num_fsr	Numéro auto	Numéro du fournisseur
Societe	Texte de 10 caractères	Nom de la société
Adresse	Texte de 30 caractères	Adresse de la société
Ville	Texte de 10 caractères	Ville de la société
Tel	Texte de 8 caractères	N° téléphone de la société

Les lignes de la table « **Fournisseurs** » sont :

Num_frs	Societé	Adresse	Ville	Tél
1	Globo	102, Rue de palmiers	Mégrine	71 254 458
2	Avicenne	45, Rue des jasmins	Tunis	71 565 855
3	Ets Belkhiria	32, Av H. Bourguiba	Djemmel	73 874 654
4	Bounici	5, Rue Mongi Slim	Tunis	71 548 458

3. Créer une nouvelle table « **Article** » qui comportera les champs suivants :

Table « Article »		
Noµm du champ	**Type de donnée**	**Description**
Référence	Texte de 6 caractères	Clé primaire
Catégorie	Texte de 15 caractères	Doit appartenir à la liste suivante (Papeterie, Informatique, communication et audiovisuel)
Nom	Texte de 50 caractères	
Num_fsr	Numérique	
PrixTVA	Monétaire Format général	Doit être positif
TVA	Numérique réel double	Doit être 0.18 ou 0.10

Les lignes de la table « **Articles** » sont :

Réf	Catégorie	Nom	Num_fsr	Prix HTVA	TVA
R01	Papeterie	Rame papier	1	3.450	18%
R03	Informatique	Imprimante	2	120	10%
R04	Communication	Fac-simile	1	256	18%
R07	Audiovisuel	TV	3	387	10%
R09	Informatique	Souris	4	7.600	10%
R11	Papeterie	Spirale	1	1.250	18%
R14	Informatique	Streamer	2	175.500	10%

4. En imaginant que plusieurs articles peuvent correspondre à un même fournisseur et qu'en revanche, un même article ne peut pas avoir plusieurs fournisseurs, créer une relation entre les deux tables «Fournisseurs» et «Articles».
5. Créer la requête qui permet d'obtenir le prix TTC des articles sachant que :
   Prix TTC = Prix Htva *(1 + TVA))
6. Créer la requête qui permet d'obtenir la liste des articles ayant la TVA = 10% triés par ordre alphabétique selon la référence.
7. Créer la requête qui permet d'avoir la liste des articles da la catégorie «informatique».

Exercice 2

On veut implémenter sous le SGBD disponible, la base de donnée qui gère les réservations des salles de l'auberge le plus simplement possible.
Le schéma relationnel de la base de données se présente comme suit :

Clients (Ref_Cl, Nom_Cl, Prenom_Cl, Adr_Cl, CP, Ville)
Cuisiniers (Ref_Cu, Num_SS, Nom_Cu, Prenom_Cu, Nom_Jeune_Fille, Adr_Cu, CP, Ville, Date_Nais)
Réservations (Ref_Re, Ref_Cl#, Ref_Cu#, Salle, Date, Heure, Couverts, Montant)

I. Structure de la base de données :

1. Créer une base de données nommée « **Réservation.mdb** »
2. Pour chaque table ci-dessous, créer sa structure. N'oublier pas de spécifier pour chacune d'elles, sa clé primaire. Les colonnes des différentes tables sont résumées dans les tableaux suivants :

Table « Clients »			
Nom du champ	**Type**	**Taille**	**Propriétés**
Ref_Cl	Texte	3	Masque de saisie : 2 lettres suivies d'un chiffre.
Nom_Cl	Texte	50	
Prenom_Cl	Texte	50	
Adr_Cl	Texte	255	
CP	Numérique	4	
Ville	texte	50	

Table« Cuisiniers »			
Champ	**Type**	**Taille**	**Propriétés**
Ref_Cu	Texte	3	Masque de saisie : 2 lettres suivies d'un chiffre
Num_SS	Texte	30	
Nom_Cu	Texte	50	
Prenom_Cu	Texte	50	
NomJeuneFille	texte	50	
Adr_Cu	Texte	255	
CP	Numérique	4	
Ville	Texte	50	
Date_Nais	Date/heure		

Table « Réservations »			
Champ	**Type**	**Taille**	**Propriétés**
Ref_Re	Numéroauto		
Ref_Cl	Texte	3	Masque de saisie : deux lettres suivies d'un chiffre
REf_Cu	Texte	3	Masque de saisie : deux lettres suivies d'un chiffre
Salle	Texte	10	
Date	Date/heure		
Heure	Texte	10	
Couverts	Numérique	Entier	
Montant	Monétaire		

Dans la table « **Réservation** », créer une liste de choix pour **Ref_Cl,** basé sur la table «Clients» et une liste de choix pour **Ref_Cu**, basé sur la table «**Cuisiniers** ».

II. Relations entre tables

Créer les relations entre les tables.

III. Peuplement de la base de données

1. Saisir les informations suivantes dans les tables correspondantes.

Table «Clients»					
Ref_Cl	**Nom_Cl**	**Prenom_Cl**	**Adr_Cl**	**CP**	**Ville**
AA1	Toumi	Mounir	12, rue des jasmins	4000	Sousse
AA2	Ben Brahim	Fethi	5, place des jacinthes	1000	Tunis
AA3	Rouatbi	Maher	3, avenue des oxalis	6000	Sfax
AA4	Bouhlel	Saïf	31, rue des pâquerettes	7000	Gabes

Table «Cuisiniers»								
Ref_Cu	**NumSS**	**Nom_Cu**	**Pre_Cu**	**NomJF**	**Adr_Cu**	**CP**	**ville**	**D_Nais**
BB1	259017	Berriri	Fatma	Slama	2, rue des magnolias	2000	Béja	04/01/60
BB2	164089	Benzarti	Jawhar		51, rue des violettes	5000	Siliana	15/08/64

Table « Réservations »							
Ref_Re	**Ref_Cl**	**Ref_Cu**	**Salle**	**Date**	**Heure**	**Couverts**	**Montant**
1	AA1	BB1	Séminaire	04/03/07	12 :00	25	4,000
2	AA4	BB1	Mariage	11/05/07	20 :00	140	16,800
3	AA1	BB1	Séminaire	12/05/07	12 :00	20	3,600
4	AA2	BB1	Salon	28/07/07	19 :00	5	1,150
5	AA2	BB2	Séminaire	30/01/07	12 :00	14	2,240
6	AA1	BB2	Salon	20/04/07	20 :00	6	1,680
7	AA2	BB2	Séminaire	12/07/07	13 :00	10	2,000
8	AA3	BB2	Mariage	12/11/07	19 :00	110	15,950

IV. Requêtes

1. Afficher les données concernant les réservations (Ref_Cl, Ref_Cu, Salle, Date, Heure, Couverts, Montant) avec le nom du client ainsi que le nom de cuisinier.
2. Afficher le nom et le prénom des clients qui ont fait réservation avant le mois du juillet 2007, ainsi que la date de réservation.
3. Afficher toutes les réservations d'un client dont le nom est saisi au clavier (Requête sélection paramétrée).
4. Afficher pour chaque cuisinier le nombre des réservations dont il se charge.
5. Afficher les montants des réservations et les regrouper par date.
6. Afficher les montants des réservations et les regrouper par trimestre.
7. Afficher les montants des réservations regrouper par salle, pour chaque cuisinier et pour chaque client.
8. Créer une table **Codes Postaux** avec les champs **CP** et **Ville** de la table «**Clients**» (Requête création).
9. A l'aide d'une **Requête ajout**, ajouter les données de la table «**Cuisiniers**» («CP » et «Ville») à la table crée «**Codes Postaux** ».

Exercice 3

L'objectif est la création de la base de données « CHANGE » et les tables qui la constituent, à savoir :

CLIENT (Num_Passeport, Nom_Cl, Adr_Cl, Nationalite_Cl)
AGENCE (Code_Ag, Nom_Ag, Adr_Ag)
DEVISE (Code_Devise, Libelle, Unite, Cours_Achat, Cours_Vente)
OPERATION (Code_Op, Type_Op, Date_Op, Num_Passeport#, Code_Ag#)
DETAIL_OPERATION (Code_Op#, Code_Devise#, Montant_Dev, Cours_Jour)

Table « Client »

Num_Passeport	Nom_Cl	Adr_Cl	Nationalite_Cl
K029534	Asma Tlili	Tunis	Tunisienne
K054679	Rim Fakhfakh	Sfax	Tunisienne
K145309	Ines Lakhoua	Tunis	Tunisienne
K549027	Ahmed ben Miled	Sfax	Tunisienne
M123497	Zied Soltani	Tunis	Tunisienne
M125302	Med Bes Salah	Tunis	Tunisienne
M132089	Zahra Sallemi	Sfax	Tunisienne
M134236	Amna Riahi	Sousse	Tunisienne
M202745	Samir Elouni	Bizerte	Tunisienne
M243678	Sami Meddeb	Tunis	Tunisienne

Table « Opération »

Code_Op	Type_Op	Date_Op	Num_Passeport
1	Vente	01/01/01	M123497
2	Vente	01/01/01	K145309
3	Achat	01/01/01	K054679
4	Vente	05/02/01	M125302
5	Achat	05/02/01	M132089
6	Achat	10/02/01	M123497
7	Vente	22/03/01	M134236

Table « Détail Opération »			
Code_Op	**Code_Devise**	**Montant_Dev**	**Cours_Jour**
1	BEF056	7000	3.221
1	FRF250	2500	1.981
2	CAD124	600	0.914
2	USD124	500	1.462
3	ESP724	1250	7.603
4	DEM280	600	6.644
4	GBP826	200	2.121
5	SAR682	200	3.802
5	USD840	100	1.423
6	EUR978	100	1.264
7	FRF250	2000	1.981
7	EUR978	150	1.300
7	DEM280	400	6.664

Table « Devise »				
Code_Devise	**Libelle**	**Unite**	**Cours_Achat**	**Cours_Vente**
BEF056	FRANC BELGE	100	3.137	3.221
CAD124	DOLLAR CANADIEN	1	0.885	0.914
CHF756	FRANC SUISSE	10	8.638	8.865
DEM280	DEUTSCHE MARK	10	6.474	6.644
ESP724	PESETA ESPANOL	1000	7.603	7.818
EUR978	EURO	1	1.264	1.300
FRF250	FRANC FR	10	1.932	1.981
GBP826	LIVRE STERLING	1	2.076	2.212
ITL380	LIRA ITL	1000	0.654	0.670
JPY392	YEN JAPON	1000	11.463	12.561
SAR682	RIYAL SAOUD	10	3.802	3.926
USD840	DOLLAR U.S	1	1.432	1.462

Requêtes :

1. Donner les numéros des passeports des clients qui habitent à Tunis.
2. Donner les natures des opérations effectuées par le client Zahra Sallemi ainsi que le cours du jour à la date du 05/02/01.
3. Affichez le nombre d'opérations regroupées par ville à la date du 01/01/2001.
4. Quelles sont les différentes devises qui ont été achetées par les clients ?

Exercice 4 : Gestion d'une conférence

Soit la base de données définie comme suit :

PARTICIPANTS (numparticipant, nom, prenom, datenaissance, numrue, nomrue, codepostal, ville, pays, langue)
CONFERENCES (numconference, titre, date, heuredebut, duree, theme, reforateur, refanimateur, numsalle#)
RESERVATIONS (numconference#, numparticipant#)
SALLES (numsalle, nom, batiment, numero, superficie, capacite, equipement)

Exprimer les requêtes suivantes en SQL :

1. Afficher la liste des personnes (numéro et nom) qui ont réservé une place pour la conférence numéro 16, dans l'ordre alphabétique des noms.
2. Afficher pour chaque conférence la liste des personnes ayant réservé dans l'ordre des numéros de conférences et dans l'ordre alphabétique pour chaque conférence.
3. Afficher pour chaque conférence son numéro et le nom du conférencier ainsi que la liste des personnes ayant réservé.
4. Afficher le programme de chaque salle pour le 15/11/2007.
5. Afficher pour chaque participant la liste des conférences pour lesquelles il a réservé.
6. Id mais on affichera également ceux n'ayant pas réservé.
7. Un participant signale qu'il arrivera le 15/11/2007 à 11h et qu'il doit repartir le 16/11/2007 à 17h. Donnez la liste des conférences auxquelles il pourrait assister.
8. Afficher la liste des participants parlant la même langue que monsieur BEN SALEH.
9. Afficher la liste des salles données par leur numéro, leur emplacement et la précision petite, moyenne ou grande selon que la capacité est inférieure à 40, entre 40 et 100 ou supérieure à 100.
10. La salle A7125 du bâtiment A ne peut être utilisée. Afficher les salles de capacité supérieure.
11. La salle A7125 du bâtiment A ne peut être utilisée. Afficher les salles de capacité supérieure et de même niveau d'équipement.
12. Afficher le nombre de salles et la capacité totale des salles.
13. Afficher la capacité totale des salles de moins de 100 places.
14. Afficher la capacité totale des salles par niveau d'équipement.
15. Afficher pour chaque conférence le nombre de participants.
16. Afficher la liste des conférences ayant plus de 100 participants.
17. Lorsque la langue du participant est différente de la langue de l'orateur, il faut prévoir un casque pour la traduction. Donner le nombre de casques à prévoir pour chacune des conférences.
18. Afficher la liste des participants aux conférences du thème SGBD et des participants aux conférences données en anglais.
19. Afficher la liste des conférences ayant le lieu le même jour et dans la même salle que la conférence 37.
20. Afficher la liste des salles où sont données des conférences avec des participants parlant l'arabe.

Exercice5 : utilisation de SQL

1. Création des Tables

Liste_Produit	Type_Reduc	Liste_Famille
IdLP INT (clé primaire auto incrementée) Produit CHAR(50) Prix FLOAT PrixR FLOAT LPidTR INT LPidLF INT NbStock INT	IdTR INT (clé primaire auto incrementée) Reduction CHAR(50) Montant INT	IdLF INT (clé primaire auto incrementée) Famille CHAR(50)

Ecrivez les requêtes SQL correspondantes. Aucun champ ne doit pouvoir être vide.

2. Saisie d'information

Maintenant, il faut entrer du contenu dans les tables. Voici des tableaux contenant les informations qui doivent y être ajoutées.

2.1. Table Liste_Produit

IdLP	Produit	Prix	PrixR	LPidTR	LPidLF	NbStock
1	CD Delerm	12	0	2	1	200
2	CD The Darkness	18	0	2	1	200
3	CD Placebo	14	0	2	1	200
4	Livre	8	0	1	1	2000
5	DVD	25	0	2	1	1000
6	Lecteur Mini-Disc	150	0	3	4	100
7	Disquette	5	0	4	5	3000

2.2. Table Type_Reduc

IdTR	Reduction	Montant
1	Aucune	0
2	Multimedia	20
3	Materiel	10
4	Alimentaire Base	10
5	Alimentaire vital	25

2.3. Table Liste_Famille

IdLF	Famille
1	Multimedia
2	Litterature
3	Informatique
4	Hi-Fi
5	Alimentaire

3. Mise à jour des Prix Réduit

Faites une requête qui met à jour le champ "PrixR" de la table Liste_Produit en faisant le pourcentage depuis la table réduc. Par exemple, pour le CD, le résultat doit être :
Prix = 14, Montant = 20 (pourcentage !), donc PrixR = 14*(100-20)/100 = 11,2

4. Ajout d'une colonne dans une table existante

On va maintenant ajouter la colonne "Commentaire" à la table Liste_Produit en initialisant tous les enregistrements à "" (chaîne vide) et d'une taille maximale de 250 caractères.

5. Vente d'un livre

On vient de vendre un livre. Il faut donc réduire le stock d'une unité. Ecrivez la requête correspondante.

6. Affichage des CDs selon le prix

6.1. Prix compris entre 13 et 15

On vous demande la liste des CDs dont le prix est compris entre 13 et 15. Vous devez donc récupérer uniquement les CDs (grâce au nom du produit) et ceux dont le prix est 13, 14 ou 15. Écrivez la requête correspondante.

6.2. Commentaire en fonction du prix

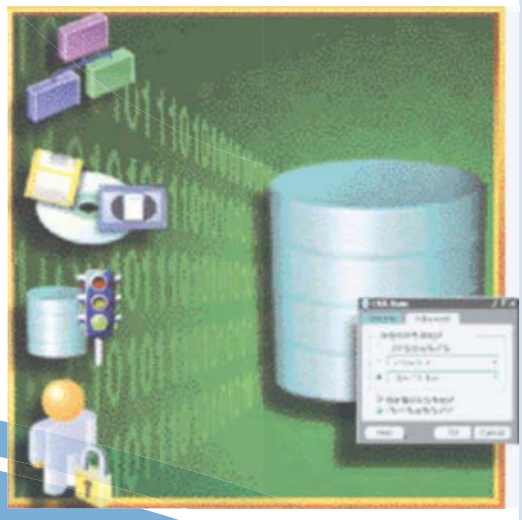
On souhaite afficher, selon le prix et pour tous les CDs, les commentaires suivants :

- inférieur à 13 : bon marché ;
- entre 13 et 15 : parfait ;
- supérieur à 15 : trop cher ;

Ce commentaire doit être dans un champ "Comment". Seuls les champs "Produit" et "Comment" doivent être affiché.

Chapitre 7

Développement d'applications autour d'une Base de Données



Objectifs :

- Découvrir la structure d'une application
- Apprendre à développer des formulaires
- Apprendre à développer des états
- Apprendre à générer une page web de données

Plan :

1. Introduction :

Une base de données peut être utilisée par deux catégories de personnes : des informaticiens et des non informaticiens. Toutes les manipulations que nous avons vues dans les chapitres précédents (création des structures, insertion, modification, suppression et consultation des données) sont faites par des informaticiens ou des personnes ayant des connaissances de base en informatique en général et en bases de données en particulier.

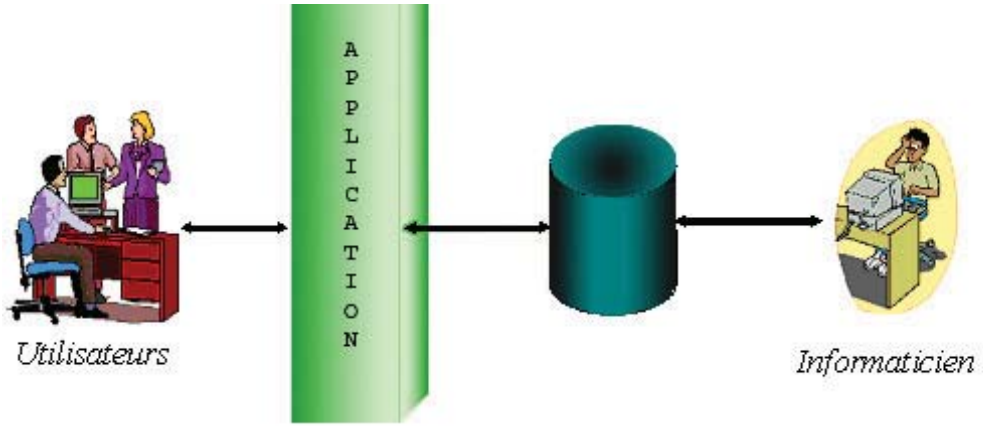


Figure 7.1 : *Types d'utilisateurs d'une base de données*

Pour que des non informaticiens puissent manipuler facilement le contenu d'une base de données, on doit mettre à leur disposition des applications qui leur permettent de communiquer facilement avec la base de données. Ces applications sont composées de deux types de modules (ou programmes) :

- Des formulaires permettant de saisir, consulter, modifier ou supprimer des données
- Des états permettant d'éditer à l'écran ou sur papier des données provenant de la base de données.

Même si l'objectif principal de ce manuel est de présenter les concepts, les principes et les techniques permettant de créer et de manipuler une base de données, nous allons décrire dans ce chapitre les éléments de base relatifs au développement d'une application autour d'une base de données.

2. Structure d'une application

2.1 Introduction

Toute application informatique est constituée de deux composantes principales :

- Une partie visible aux utilisateurs : interface utilisateur
- Une partie cachée : code

Les utilisateurs ne voient qu'une partie minime de l'application : l'interface utilisateur. De point de vue charge de travail, la partie cachée de l'application, c'est-à-dire le code à écrire par le développeur, constitue la partie majeure de l'application. On assimile souvent une application informatique à un iceberg dont la partie visible ne constitue qu'une partie infime par rapport à la partie immergente.

2.2 Interface utilisateur

L'interface utilisateur, appelée également interface homme-machine est composée de deux éléments :

- **Des formulaires** : Ils permettent aux utilisateurs d'interagir avec l'application à travers des champs de type texte, liste, case à cocher et autres types d'objets graphiques. Les opérations que peut faire l'utilisateur à travers ces formulaires sont la consultation, l'insertion, la modification et la suppression de données.
- **Des états** : Ils permettent aux utilisateurs d'obtenir des données afin de les imprimer ou les stocker sous une forme quelconque.

2.3 Code d'une application

La partie dynamique d'une application est constituée du code qui est associé aux différents objets composant l'application.

Exemple

Exemples d'objets auxquels peut être associé du code :

- Une case à cocher
- Un bouton d'action
- Une liste déroulante

Généralement, l'activation du code est déclenchée par la réalisation d'un événement. La programmation est dite alors événementielle.

Exemple

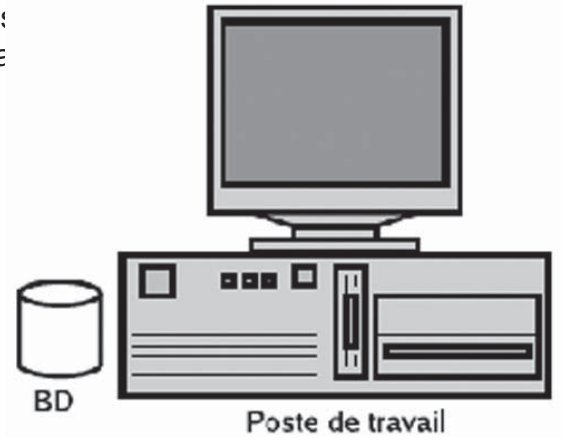
Exemples d'événements :

- Cocher ou décocher une case
- Cliquer sur un bouton d'action
- Sélectionner un élément dans une liste déroulante

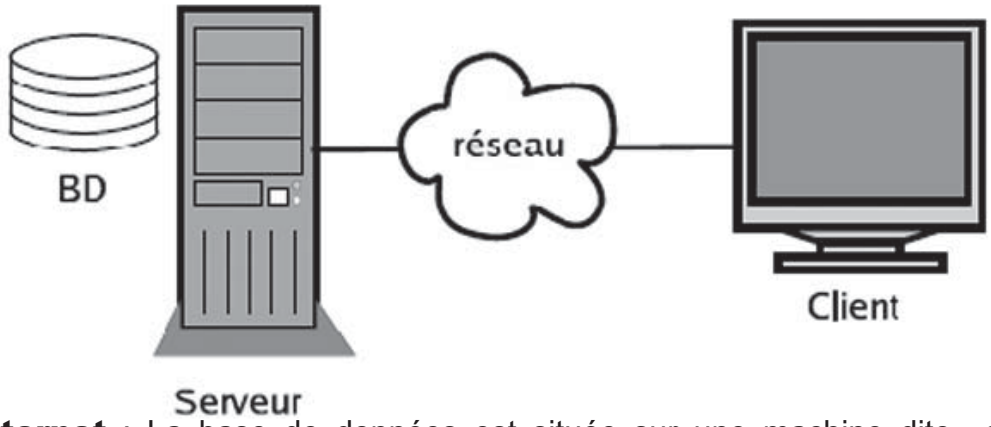
2.4 Modes d'utilisation des applications

Une application développée autour d'une base de données peut être exploitée (déployée) selon trois mode

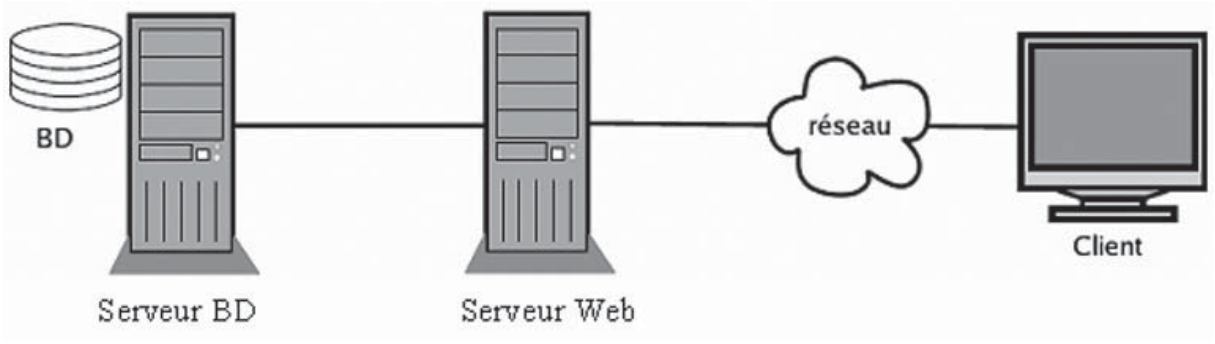
- **Mono poste** : L'application et la ba même machine.



• **Client/serveur** : La base de données est située sur une machine dite «serveur de données» et l'application est installée sur des machines dites «clients». L'ensemble est interconnecté à travers un réseau local ou distant.



• **Internet** : La base de données est située sur une machine dite «serveur de données» et l'application est sous forme de pages web gérées par un serveur web. Les utilisateurs accèdent aux pages web à partir d'un navigateur installé sur leurs postes de travail.



3. Les formulaires

3.1 Introduction

Activité

Enoncé :

Pour formuler une demande d'abonnement à une bibliothèque, deux propositions courantes peuvent se présenter :

Écrire une demande sous forme de texte libre.
Remplir un formulaire pré établi par la bibliothèque.

Question :

1- Proposer un exemple de texte libre pour cette demande.
2- Proposer un exemple de formulaire à remplir pour cette demande
3- Présenter une exploitation ultérieure de chacune des deux formulations au sein de la direction de la bibliothèque.
4- Comparer les deux documents et présenter les avantages de la solution qui prête à l'informatisation.

Eléments de réponse à développer avec votre enseignant :
Il est évident que la deuxième solution est préférable à la première car elle cumule plusieurs avantages, notamment :

- La rapidité : Aussi bien lors de la saisie que lors du traitement de la demande
- L'harmonisation : N'importe quelle personne remplie le formulaire de la même façon alors que chacun peut écrire une demande à sa manière.

5- Reprendre la structure étudiée dans l'application 1 du Chapitre 4 et proposer alors un modèle de formulaire à prévoir.

Ceci n'est pas valable uniquement pour les formulaires papiers. Les formulaires informatiques se basent sur le même principe. Ils permettent, à travers une interface graphique conviviale, d'effectuer essentiellement les tâches suivantes :

- Rechercher et afficher des données en provenance de la base de données selon des critères fixés par l'utilisateur.
- Saisir de nouvelles données pour alimenter la base de données.
- Mettre à jour les données existantes.
- Supprimer des données de la base.

Les formulaires permettent en général d'effectuer sur une base de données les mêmes opérations que nous avons vues dans le chapitre précédent, à savoir la consultation, l'insertion, la modification et la suppression.

Les formulaires les plus simples se basent sur une seule table. Lorsque les données que l'on souhaite afficher, insérer, modifier ou supprimer proviennent de deux ou plusieurs tables, le formulaire correspondant se base aussi sur deux ou plusieurs tables.

Dans le reste de ce chapitre nous allons nous intéresser aux formulaires informatiques. Nous allons montrer comment créer, modifier et exploiter ces formulaires.

3.2 Création d'un formulaire

Vous allez apprendre à créer un formulaire simple permettant de saisir et d'afficher des données d'une table.
Dans certains SGBD, il existe trois façons pour créer un formulaire :

- **Création de formulaire instantané** : c'est la forme la plus rapide pour créer un formulaire
- **Utilisation d'un assistant** : C'est une forme combinant la rapidité et la diversité lors de la création d'un formulaire
- **Création libre** : C'est la forme la plus complète (offre le plus de possibilités) mais aussi la plus lente en temps de création.

Ces trois façons peuvent être combinées. Généralement on commence par la création d'un formulaire instantané ou en utilisant un assistant, puis on finalise le résultat obtenu en création libre.
Dans ce qui suit, nous allons présenter ces modes pour créer un formulaire simple.

3.2.1. Création de formulaire instantané

La création d'un formulaire instantané est le mode le plus facile et le plus rapide. Il est souvent utilisé pour créer un premier prototype d'un formulaire.

Dans la suite, nous allons créer un formulaire instantané basé sur la table :

Detail_commande (N° commande, N° ligne, Code_art, Qté commandée)

La création d'un formulaire instantané se fait en suivant les étapes suivantes :

3. Dans la fenêtre «Base de données », sélectionner la table Detail_commande
4. Dans le menu principal, sélectionner le sous-menu nouvel objet ( ) puis l'option «Formulaire instantané».

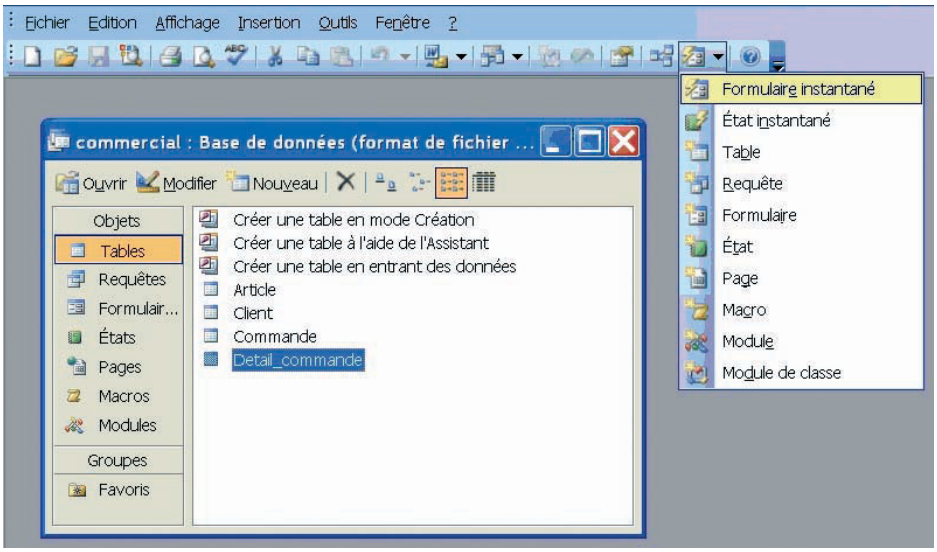


Figure 7.1 *: Sélection de la table Detail_commande*

5. Dés l'activation de l'option « Formulaire instantané », une nouvelle fenêtre contenant le nouveau formulaire est affichée :

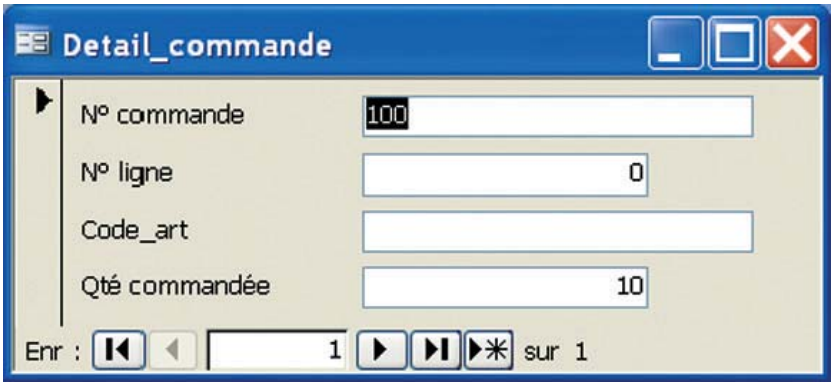


Figure 7.2 *: Formulaire instantané de la table Detail_commande*

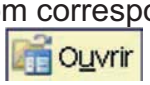
6. Cliquer sur « Enregistrer » dans le menu principal. Une fenêtre vous invitant à saisir un nom pour le nouveau formulaire s'affiche :

Figure 7.3 *: Enregistrement d'un formulaire*

Par défaut, c'est le nom de la table qui est attribué au nouveau formulaire. Vous pouvez attribuer un autre nom. Cliquer ensuite sur « Ok » pour enregistrer le nouveau formulaire.

Le nouveau formulaire est maintenant créé, il apparaît dans la liste des formulaires (fenêtre Formulaires).

Figure 7.4 *: Liste des formulaires*

L'affichage de ce formulaire se fait en double-cliquant sur le nom correspondant dans la liste des formulaires ou bien en activant l'option «Ouvrir» () dans la fenêtre « Base de données ».

Le formulaire que nous venons de créer est composé de deux parties :

- *Le corps du formulaire* constitué par les quatre champs correspondant aux colonnes de la table *Detail_commande*. C'est dans cette partie que se fait la saisie et l'affichage des données échangées avec la base de données à travers le formulaire.

- Une zone de contrôle permettant la navigation entre les lignes de la table. Cette zone permet de se déplacer entre les lignes de la table sur laquelle est basé le formulaire. Une description détaillée de cette zone sera donnée dans la section «Exploitation d'un formulaire».

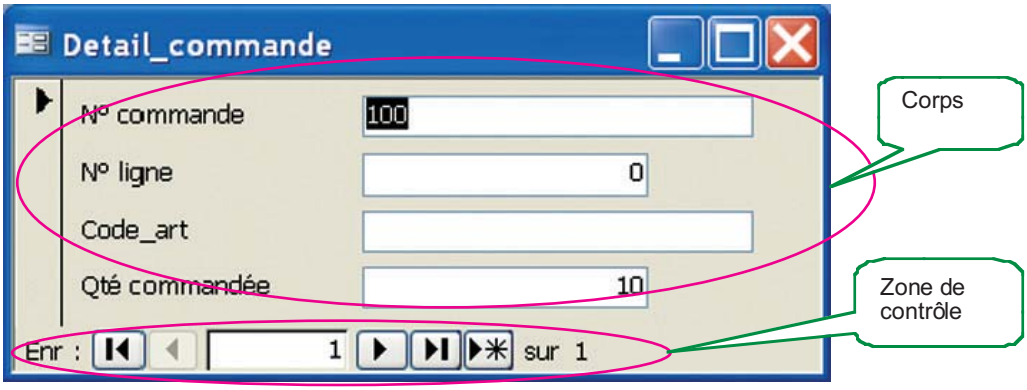


Figure 7.5 : *Structure d'un formulaire*

Le formulaire que nous venons de créer constitue la forme la plus courante. Il a la particularité qu'il permet d'afficher une seule ligne à la fois de la table correspondante.

La forme **tabulaire** est la deuxième forme de point de vue manipulation des données d'une table à travers un formulaire. Comme son nom l'indique, elle permet d'afficher les données d'une table sous forme tabulaire.

Par exemple, pour la table *Detail_commande*, le formulaire sous forme tabulaire se présente comme suit :

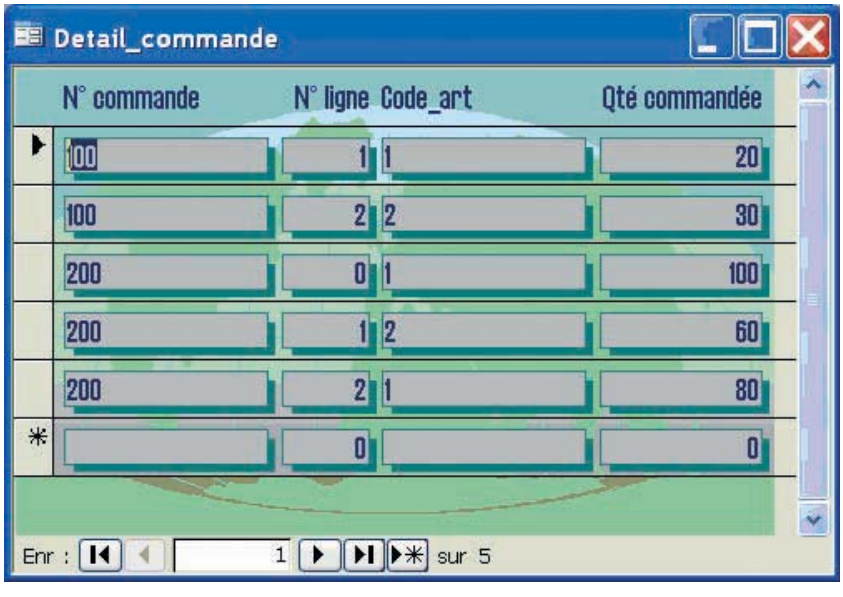


Figure 7.6 : *Formulaire tabulaire de la table Detail_commande*

Ce formulaire permet d'effectuer sur la table *Detail_commande* les mêmes opérations que celles de la figure 7.2 C'est uniquement le mode d'affichage qui change : le premier affiche une ligne à la fois alors que le deuxième en affiche plusieurs.

La création d'un formulaire instantané de type tabulaire se fait en suivant les étapes suivantes :

1. Dans la fenêtre «Base de données», sélectionner «Formulaires » dans la liste des objets, puis dans le menu de cette fenêtre sélectionner l'option «Nouveau».

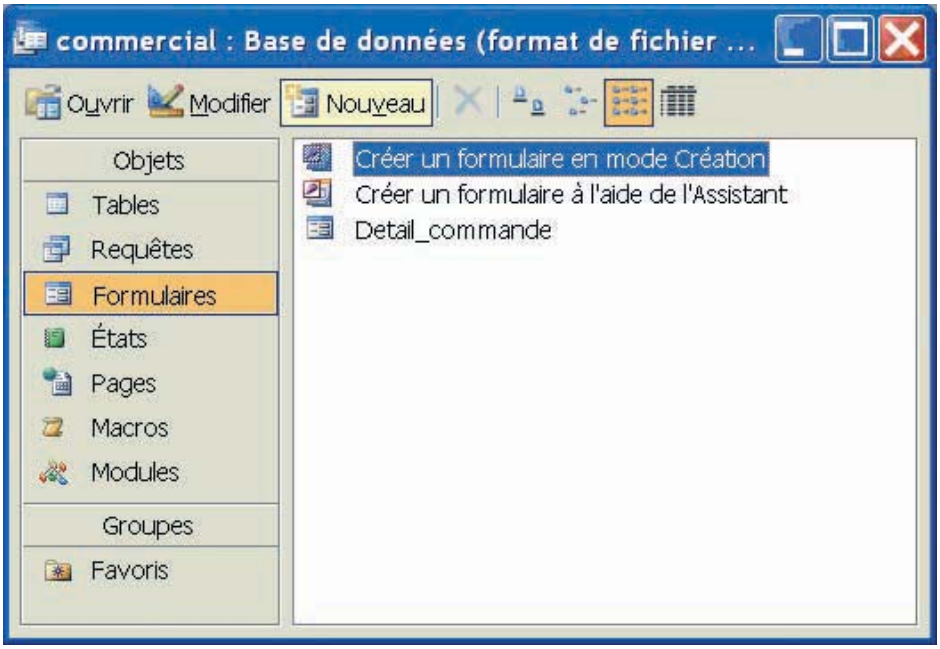


Figure 7.7 : *Lancement de l'opération création d'un formulaire tabulaire*

2. Dans la fenêtre «Nouveau formulaire» qui s'affiche, sélectionner la ligne «Formulaire instantané : tableau», puis dans la partie inférieure sélectionner la table sur laquelle sera basé le nouveau formulaire (*Detail_commande* dans notre cas). Cliquer ensuite sur «Ok».

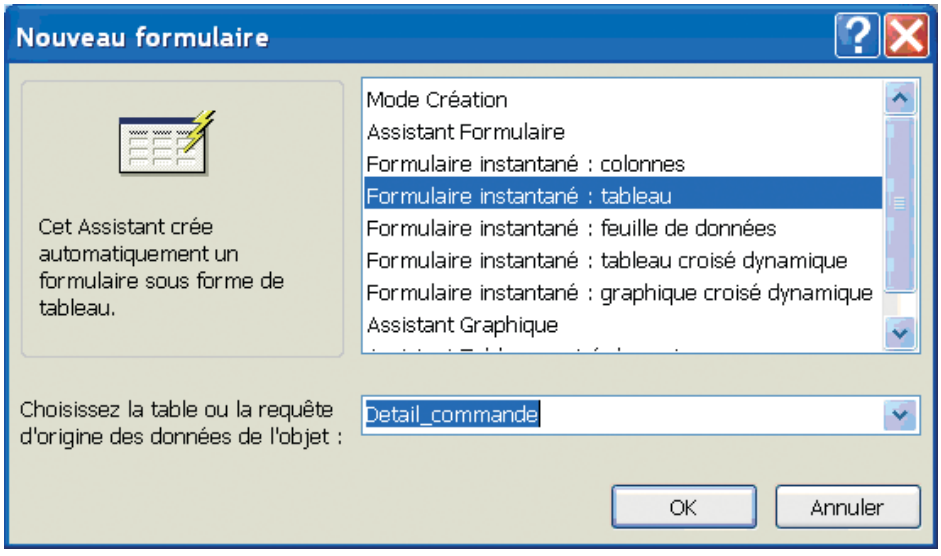


Figure 7.8 : *Création d'un formulaire tabulaire*

3. Dés l'activation du bouton «Ok», une nouvelle fenêtre contenant le nouveau formulaire tabulaire est affichée :

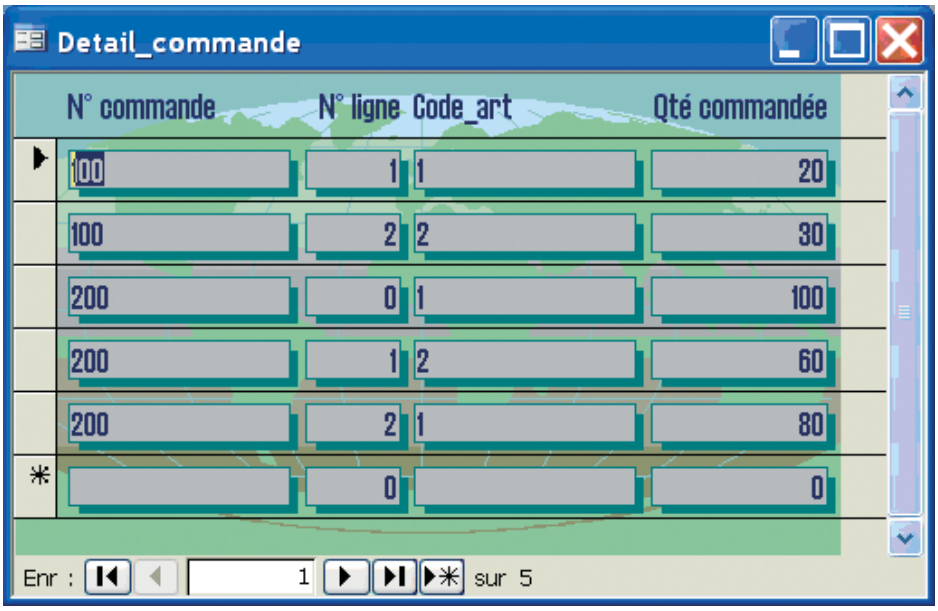


Figure 7.9 : *Formulaire tabulaire de la table Detail_commande*

4. Cliquer sur «Enregistrer» dans le menu principal. Une fenêtre vous invitant à saisir un nom pour le nouveau formulaire s'affiche :

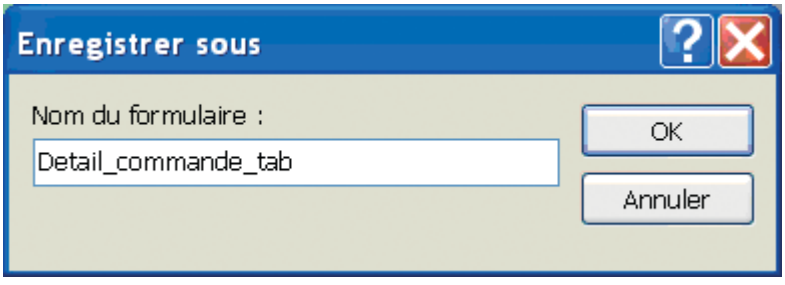


Figure 7.10 : *Enregistrement du nouveau formulaire*

Par défaut, c'est le nom de la table qui est attribué au nouveau formulaire étendu éventuellement par un numéro (1, 2, …) si ce nom existe déjà. Dans notre cas nous allons attribuer le nom Detail_commande_tab. Cliquer ensuite sur «Ok» pour enregistrer le nouveau formulaire.

Le nouveau formulaire est maintenant créé. Il apparaît dans la liste des formulaires (fenêtre Formulaires).

3.2.2. Création de formulaire à l'aide d'un assistant

La création d'un formulaire peut se faire également en utilisant un assistant. Ce mode prend plus de temps que le mode précédent (instantané) mais il offre plus de possibilités.

La création d'un formulaire à l'aide d'un assistant se fait en suivant les étapes suivantes :

1. Dans la fenêtre «Base de données», sélectionner «Formulaires» dans la liste des objets, puis dans le menu de cette fenêtre sélectionner l'option «Nouveau».

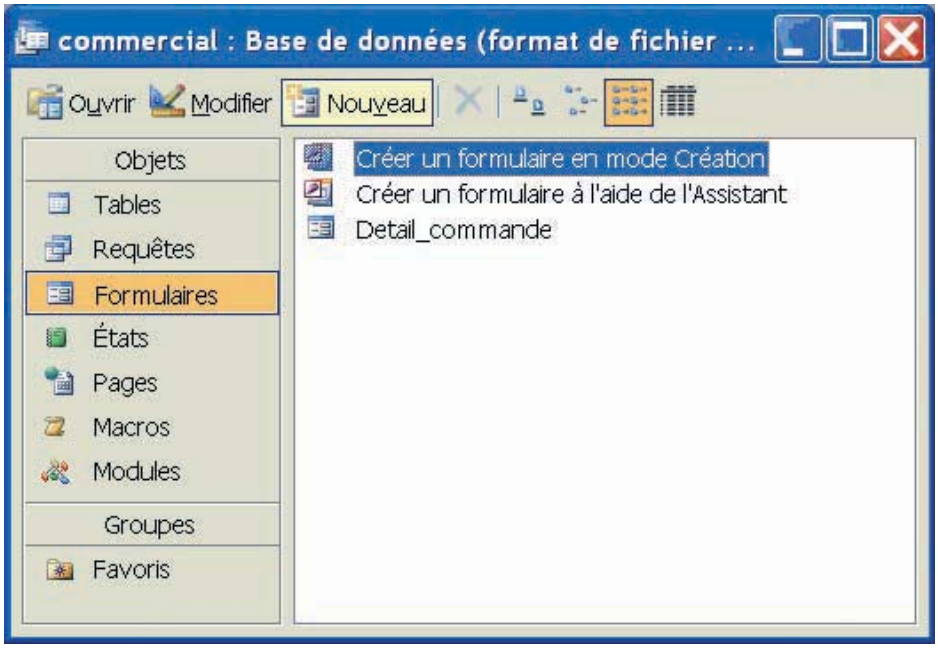


Figure 7. 11 : *Lancement de l'opération création d'un formulaire avec assistant*

2. Dans la fenêtre «Nouveau formulaire» qui s'affiche, sélectionner la ligne «Assistant formulaire», puis dans la partie inférieure sélectionner la table sur laquelle sera basé le nouveau formulaire (*Detail_commande* dans notre cas). Cliquer ensuite sur le bouton « Ok».

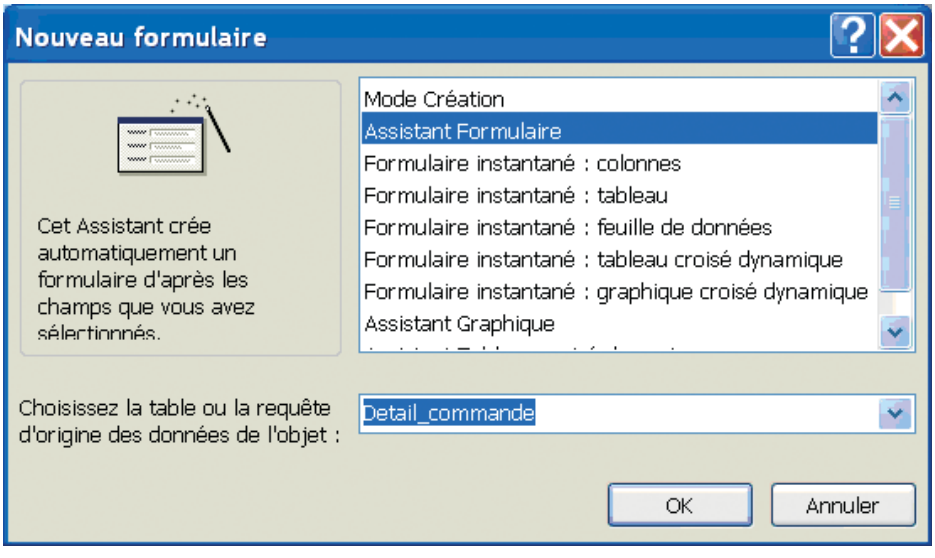


Figure 7. 12 : *Création d'un formulaire à l'aide d'un assistant : page 1*

3. Dés l'activation du bouton « Ok», la première page de l'assistant s'affiche. Elle permet de sélectionner les champs qui vont apparaître dans le nouveau formulaire.

Il est à préciser que nous pouvons intégrer dans le même formulaire des champs provenant de plus qu'une table, à condition bien sur qu'il existe des liens entre ces tables.

Dans notre cas, nous allons rajouter la colonne désignation article à partir de la table Article, de sorte que lors de l'affichage d'une ligne du détail commande, on peut voir la désignation de l'article commandé en plus de son code.

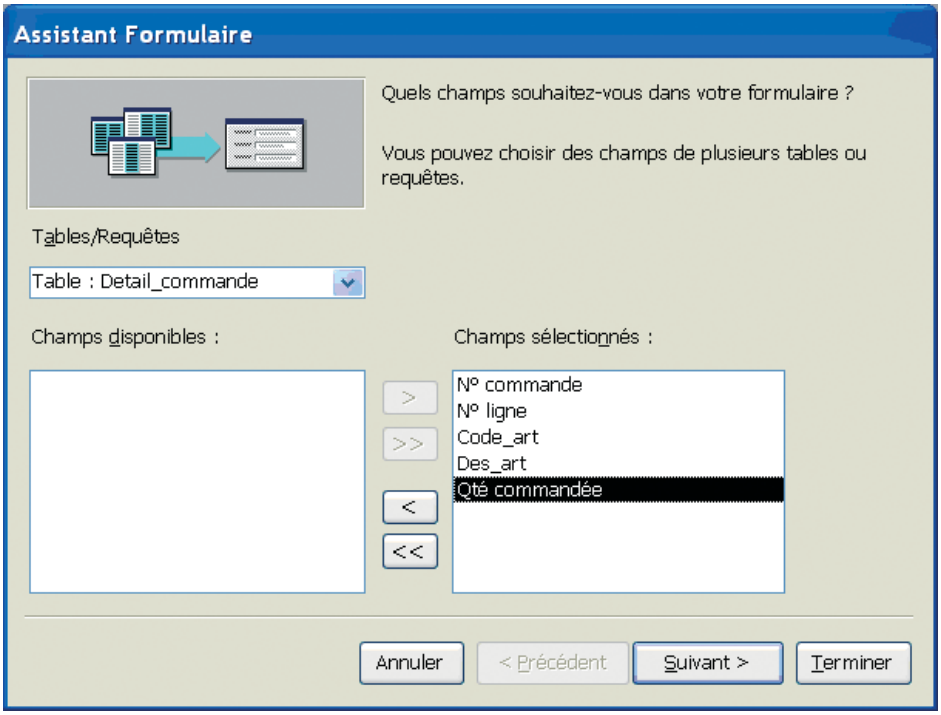


Figure 7.13 : *Création d'un formulaire à l'aide d'un assistant : page 2*

4. Cliquer sur le bouton « Suivant ». Si le formulaire est basé sur deux ou plusieurs tables, la page suivante de l'assistant s'affiche. Dans le cas où le formulaire est basé sur une seule table, c'est la page de l'étape 5 qui s'affiche.

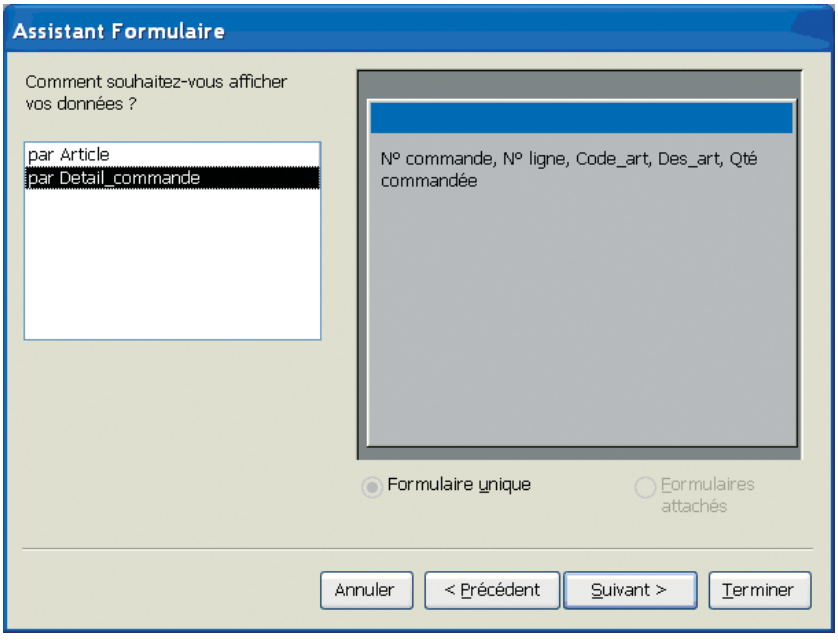


Figure 7.14 : *Création d'un formulaire à l'aide d'un assistant : page 3*

Cette page permet de préciser l'organisation de l'affichage des données en fonction de leur provenance (tables). Dans ce cas, nous choisissons la deuxième option (par Detail_commande).

Cliquer ensuite sur le bouton « Suivant».

5. La fenêtre qui s'affiche permet de sélection la disposition des données du formulaire (colonne simple, tabulaire, feuille de données, …).

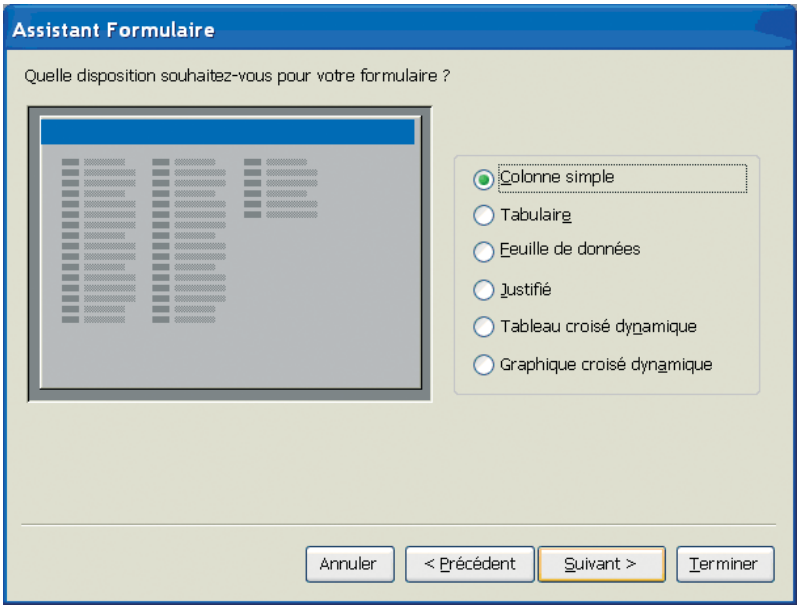


Figure 7.15 : *Création d'un formulaire à l'aide d'un assistant : page 4*

Nous allons sélectionner l'option « Colonne simple ».

6. En cliquant sur « Suivant », la page suivante de l'assistant s'affiche. Elle permet de choisir un style d'affichage parmi la liste disponible. En fonction du type d'utilisation du formulaire, sélectionner le style qui vous semble le mieux approprié.

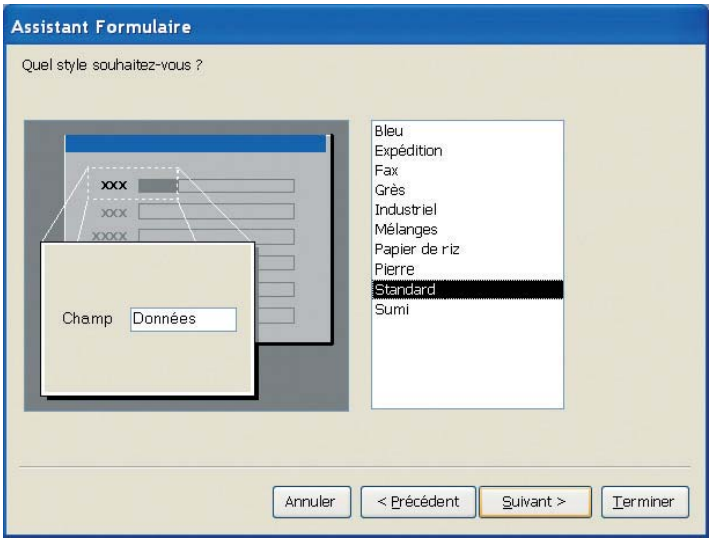


Figure 7.16 : *Création d'un formulaire à l'aide d'un assistant : page 5*

7. La dernière page de l'assistant permet d'attribuer un nom au nouveau formulaire :

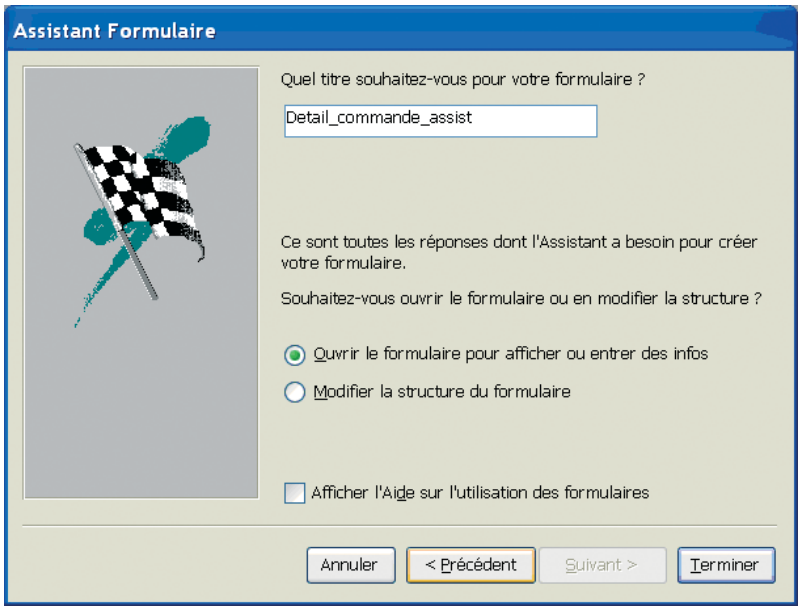


Figure 7.17 : *Enregistrement du nouveau formulaire*

8. Cliquer sur le bouton «Terminer». Le nouveau formulaire s'affiche.

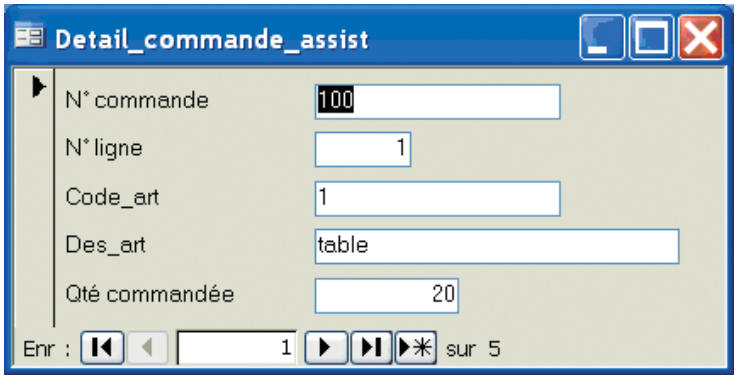


Figure 7.18 : *Formulaire Detail_commande_assist*

Noter la différence par rapport au premier formulaire que nous avons créé : pour chaque ligne de commande, la désignation est affichée en plus du code article.

3.2.2. Modification d'un formulaire

Les deux modes de création de formulaires que nous avons vus (formulaire instantané et assistant) permettent de créer automatiquement des formulaires sans laisser beaucoup de possibilités à celui qui réalise cette opération. On ne peut pas par exemple choisir la disposition des champs, l'intitulé de ces champs, leurs types (liste, case à cocher, …).

Ces deux modes ont généralement utilisés uniquement pour entamer le développement d'une application. Les formulaires obtenus ne constituent qu'une première ébauche de l'application finale.

Nous allons voir dans cette section comment affiner un formulaire en modifiant sa structure et sa présentation. Nous prenons comme exemple le dernier formulaire (Detail_commande_assist).

La modification d'un formulaire se fait en suivant les étapes suivantes :

1. Dans la fenêtre « Base de données », sélectionner « Formulaires » dans la liste des objets, le formulaire à modifier, puis dans le menu de cette fenêtre sélectionner l'option « Modifier ».

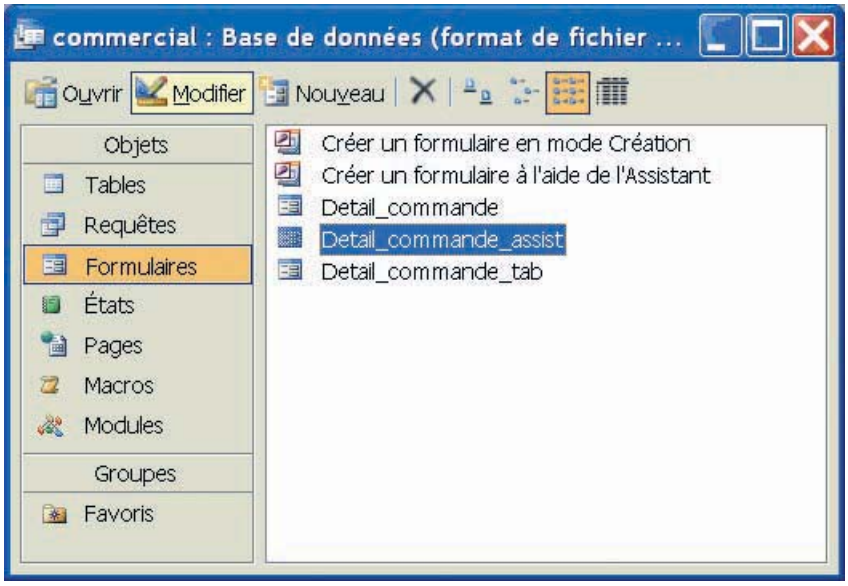


Figure 7.19 : *Lancement de l'opération modification d'un formulaire*

2. Dés l'activation de l'option « Modifier », un éditeur de formulaires s'affiche :

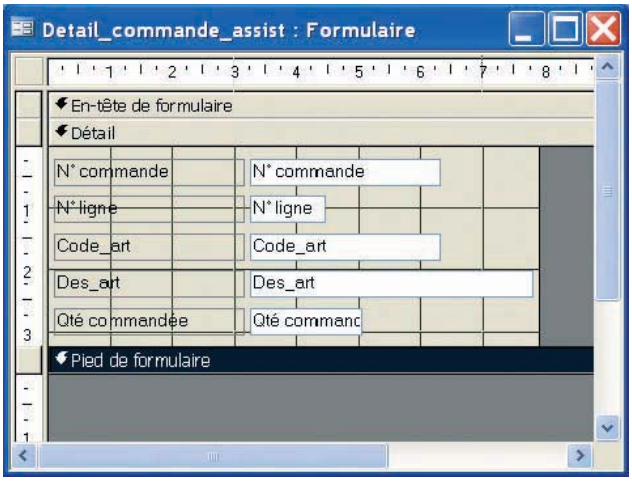


Figure 7.20 : *Éditeur de formulaire*

A l'aide de cet éditeur on peut modifier la présentation d'un formulaire. On peut par exemple :

- Rajouter un titre au formulaire
- Modifier le libellé de certains champs :

  Code_art ➔ Code

  Des_art ➔ Désignation
- Modifier la disposition des champs : mettre sur une même ligne le numéro de commande et le numéro de la ligne.
- Modifier la taille des champs : réduire la taille du champ Qté commandée.

3. Pour rajouter un titre au formulaire :

- Déplacer vers le bas la zone « Détail ».
- Afficher la boite à outils si elle n'apparaît pas en activant l'option « Boîte à outils» dans le menu « Affichage ».
- Dans la boite à outils, sélectionner le bouton «Étiquette » :

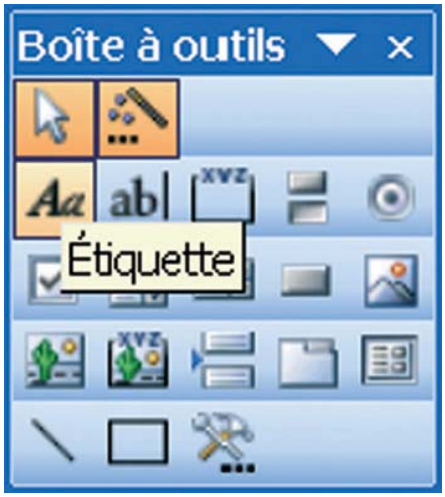


Figure 7.21 : *Barre à outils de l'éditeur de* formulaires

- Créer une zone texte dans la partie En-tête du formulaire et taper le titre du formulaire. Modifier le style d'affichage de ce titre à l'aide des outils de mise en forme de texte dans le menu principal (police, taille, couleur, …).

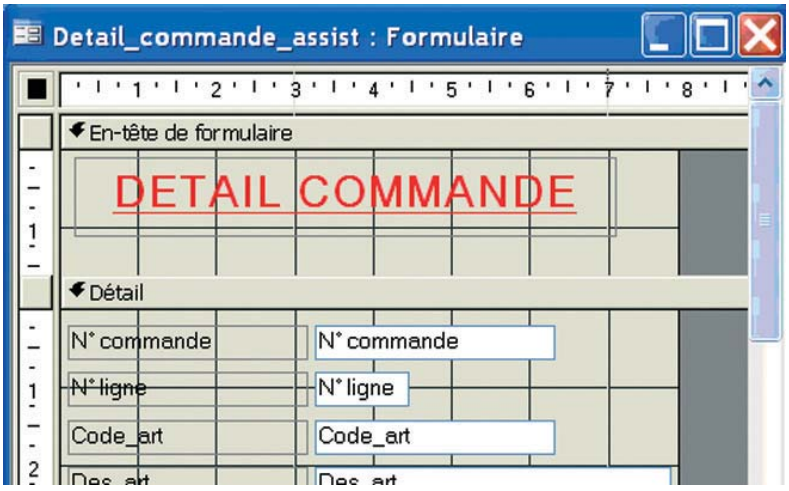


Figure 7.22 : *Ajout d'un titre à 'un formulaire*

4. Pour modifier le libellé d'un champ :

- ➢ Sélectionner le champ concerné, puis éditer le texte de ce champ.
- ➢ On peut également modifier le style du champ (police, taille, couleur, justification, …).

5. Pour modifier l'emplacement d'un champ, il suffit de le sélectionner et le déplacer vers l'emplacement souhaité. On peut sélectionner plusieurs champs en même temps.

6. Pour modifier la taille d'un champ, il faut le sélectionner puis modifier sa taille en largeur et/ou en hauteur.

Le résultat des modifications effectuées apparaît dans la fenêtre suivante :

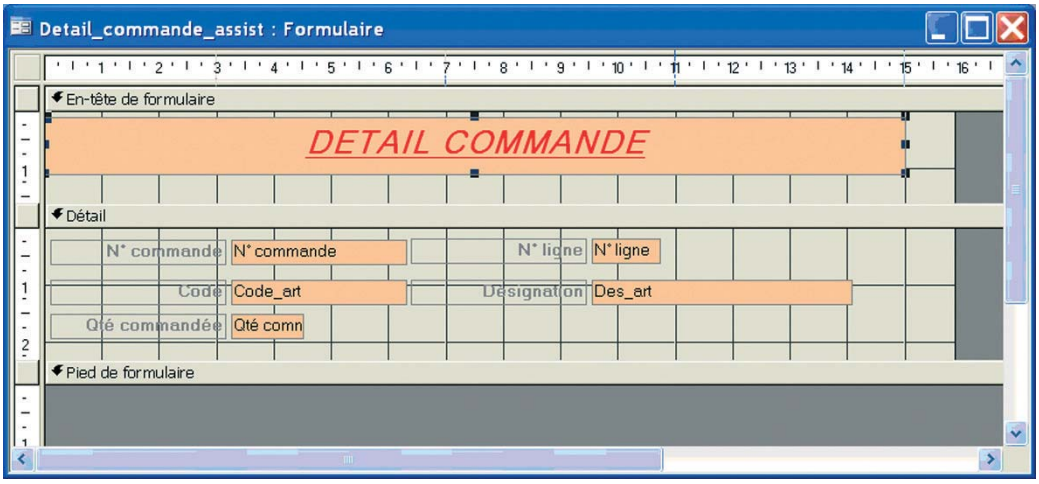


Figure 7.23 : *Résultat de l'édition du formulaire Detail_commande_assist*

Pour valider ces modifications, cliquer sur le bouton « Enregistrer » du menu principal puis fermer la fenêtre d'édition de formulaires.
L'exécution de ce formulaire apparaît comme suit :

Figure 7.24 : *Exécution du nouveau formulaire*

3.3 Exploitation de formulaire

L'exploitation d'un formulaire est très simple. Ceci est normal puisque les formulaires ont pour principal objectif de faciliter la tache des utilisateurs d'une base de données.
L'exploitation d'un formulaire consiste à effectuer les taches suivantes :

- Lancement du formulaire,
- Affichage des données à travers un formulaire,
- Insertion de données à travers un formulaire,
- Suppression de données à travers un formulaire,
- Fermeture d'un formulaire.

Dans ce qui suit, nous allons présenter chacune de ces opérations.

3.3.1 Lancement d'un formulaire

Le lancement d'un formulaire peut se faire de différentes façons :

- Sélectionner le formulaire puis double-cliquer.

➢ Sélectionner le formulaire puis activer le bouton « Ouvrir ».

➢ Créer un raccourci (option « Créer raccourci » du menu « Édition » et exécuter le formulaire directement en double-cliquant sur le raccourci.

3.3.2 Affichage de données à travers un formulaire

L'une des principales utilisations des formulaires est de permettre aux utilisateurs de visualiser le contenu de la base de données.

Lorsqu'un formulaire est ouvert, il affiche automatiquement toutes les lignes de la table sur laquelle il est basé. Si le formulaire est de type colonne, il affiche le premier enregistrement. Les autres enregistrements peuvent être atteints par navigation. Le formulaire de type tabulaire affiche un ensemble d'enregistrements à la fois. La navigation permet d'atteindre les autres enregistrements.

Exemples

Le formulaire suivant affiche le contenu de la table Detail_commande sous forme tabulaire.

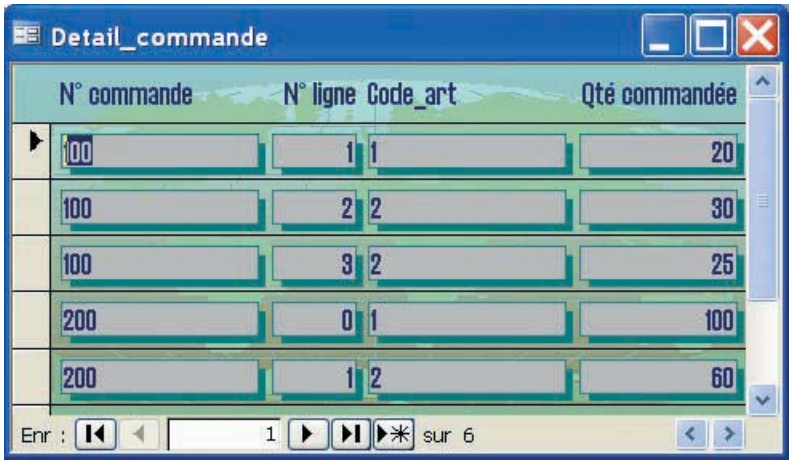


***Figure 7.25** : Affichage des données dans un formulaire*

Lorsqu'un formulaire est utilisé en consultation, on peut effectuer les actions suivantes :

➢ Navigation entre les lignes

➢ Recherche (ou filtrage) de lignes

➢ Tri des données

3.3.2.1. Navigation entre les lignes

La navigation entre les enregistrements se fait à l'aide de la zone de contrôle située en bas de la fenêtre. La figure suivante décrit les différentes composantes de cette zone de contrôle.

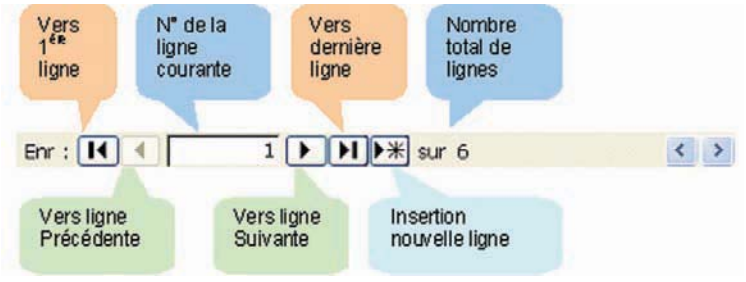


***Figure 7.26 :** Éléments de la zone de contrôle d'un formulaire*

3.3.2.2. Recherche de lignes

Lorsque le nombre de lignes d'une table est très élevé, effectuer une recherche par navigation entre toutes les lignes n'est pas pratique. La recherche de lignes se fait en précisant des critères de recherche. Dans ce cas, seules les lignes vérifiant ces critères sont affichées. On dit alors qu'on applique **un filtre** sur les données.
Il existe trois façons d'effectuer un filtre :

- **Par formulaire** : La saisie des critères de recherche se fait à l'aide d'un formulaire ayant la même structure que le formulaire courant. Ce mode permet d'appliquer des filtres complexes (combinaison des opérateurs logiques ET et OU).
- **Par sélection** : Il n'y a pas ici de saisie de critère de recherche. Il suffit de se positionner sur le champ dont la valeur est égale au critère de recherche. Ce mode est à utiliser en cas de filtres simples
- **Hors sélection** : C'est le même principe que le filtre par sélection, sauf que ce sont les lignes qui ont une valeur différente de celle mentionnée qui seront affichées.

Dans ce qui suit nous allons détailler ces trois types de filtres.

Filtre par formulaire :

Pour effectuer un filtre par formulaire, on doit procéder comme suit :

1. Sélectionner l'option «Filtrer/Filtrer par formulaire » du menu «Enregistrements» ou cliquer sur le bouton « Filtrer par formulaire » ( ).
La fenêtre suivante s'affiche :

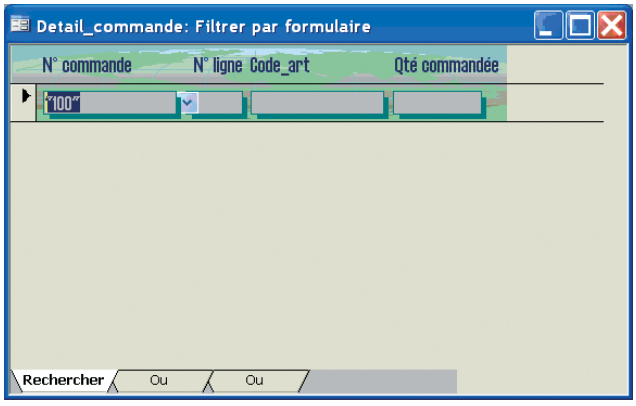


Figure 7.27 : *Filtre par formulaire*

2. La fenêtre qui s'affiche permet de saisir les critères de recherche. Il faut bien noter deux aspects pour cette fenêtre :

- Les champs sont sous forme de liste, ce qui facilite la saisie des critères de recherche. Pour la colonne N° Commande, on peut soit saisir ce numéro, soit le sélectionner dans la liste.
- La fenêtre est composée de plusieurs pages onglets visibles dans la partie inférieure de la fenêtre. Ceci permet de composer plusieurs critères de recherche avec l'opérateur logique «**OU**».

Pour définir un filtre, c'est-à-dire un critère de recherche, saisissez dans la même page les conditions qui sont combinées avec l'opérateur logique « **ET**» et dans des pages séparées les critères qui doivent être combinés à l'aide de l'opérateur «**OU**».

Exemple

Pour afficher les commandes vérifiant la condition suivante :

(N° Commande = 200 et N° ligne = 1) ou (code article = 2)

Effectuer la saisie telle comme elle est représentée dans la figure suivante :

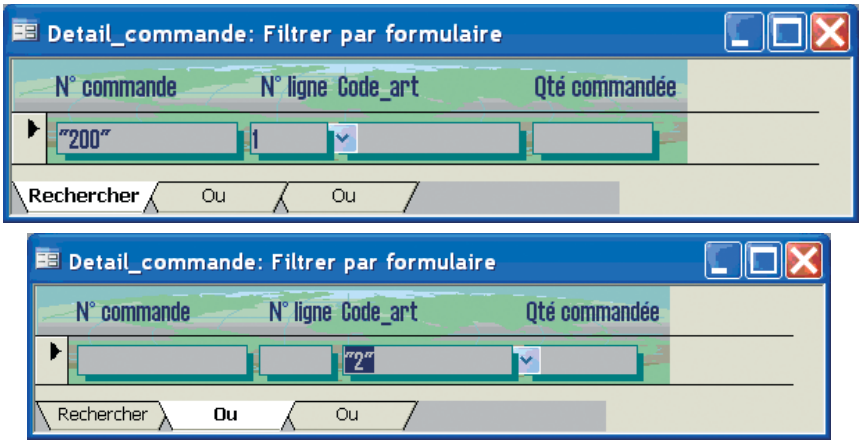


***Figure 7.28** : Saisie des critères de recherche dans un filtre par formulaire*

3. Pour lancer la recherche, cliquer sur le bouton «Appliquer filtre» ( ) du menu principal. Le résultat de la recherche s'affiche dans le formulaire.

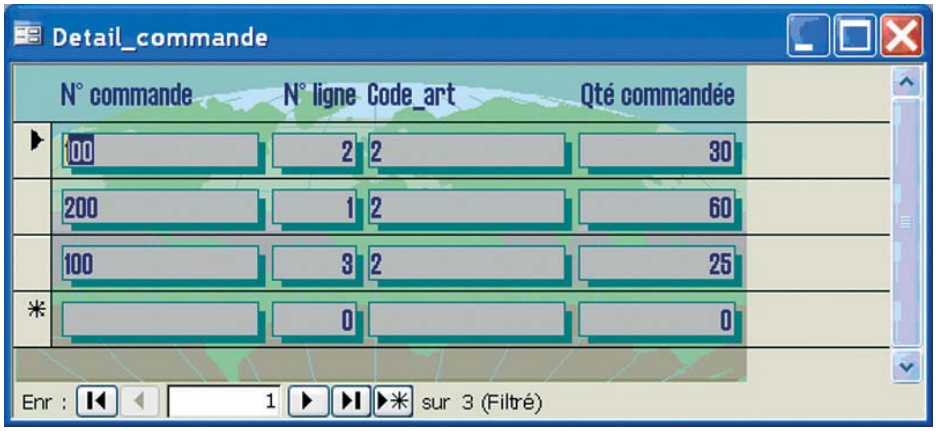


***Figure 7.29 :** Résultat du filtrage de données*

Remarque

- Noter bien que dans la ligne de contrôle (partie inférieure de la fenêtre), le nombre d'enregistrements est suivi de la mention « (Filtré) » pour indiquer que les données affichées sont le résultat de l'application d'un filtre.
- Pour annuler le filtre actuel, cliquer sur le bouton «Supprimer filtre » ( ).

Filtre par sélection :

Pour effectuer un filtre par sélection, on doit procéder comme suit :

1. Dans le formulaire dans lequel on désire effectuer un filtre, se positionner sur le champ contenant la valeur qui sera utilisée comme filtre. Par exemple, si on veut afficher toutes les lignes relatives à la commande numéro 100, se positionner sur le champ *N° Commande* de n'importe quelle ligne ayant la valeur 100 pour ce champ.

2. Sélectionner l'option «Filtrer/Filtrer par sélection» du menu « Enregistrements» ou bien cliquer sur le bouton « Filtrer par sélection» ( ). La fenêtre contenant le résultat de la recherche s'affiche.

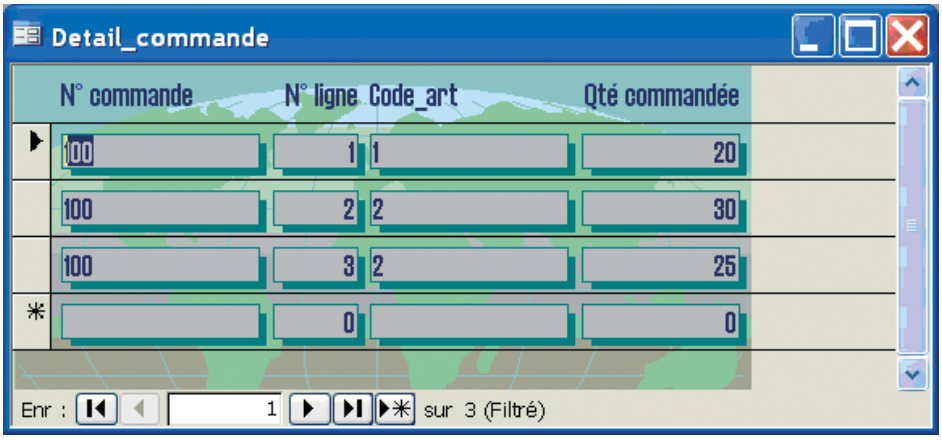


Figure 7.30 : *Résultat du filtrage par sélection*

3. Si on veut appliquer d'autres filtres sur les lignes affichées, on peut procéder de la même façon. Par exemple si on veut afficher toutes les lignes de la commande numéro 100 relatives à l'article dont le code est 2, on doit se positionner sur le champ *Code Article* dont la valeur est 2.

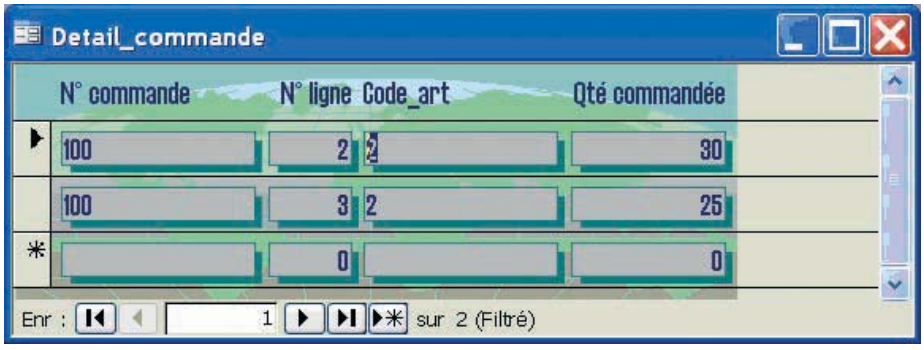


Figure 7.31 : *Filtrage par sélection sur le résultat d'un autre filtre*

Filtre hors sélection :

Pour effectuer un filtre hors sélection, on doit procéder comme suit :

1. Dans le formulaire dans lequel on désire effectuer un filtre, se positionner sur le champ contenant une valeur différente de celle qui sera utilisée comme filtre. Par exemple, si on veut afficher toutes les lignes dont le numéro de commande est différent de 100, se positionner sur le champ *N° Commande* de n'importe quelle ligne ayant la valeur 100 pour ce champ.
2. Sélectionner l'option «Filtrer/Filtrer hors sélection» du menu «Enregistrements». La fenêtre contenant le résultat de la recherche s'affiche.

3.3.2.3. Tri de lignes

Lorsque le nombre de lignes affichées dans un formulaire est très élevé, il est préférable de trier ces lignes. Le tri peut être croissant ou décroissant.

Pour effectuer un tri des lignes affichées, on doit procéder comme suit :
1. Se positionner sur le champ qu'on souhaite utiliser comme critère de tri.
2. Sélectionner l'option «Trier/Tri croissant ou décroissant» du menu «Enregistrements» ou bien cliquer sur le bouton «Tri croissant» ( ) ou «Tri décroissant » ( ). La fenêtre contenant les lignes triées s'affiche.

3.3.3 Insertion de données à travers un formulaire

Pour déclencher l'insertion de nouvelles lignes dans un formulaire déjà ouvert, il existe plusieurs façons :

- Dans la zone de contrôle du formulaire, cliquer sur le bouton «Nouvel enregistrement » ( ).
- Dans la barre d'outils, cliquer sur le bouton « Nouvel enregistrement » ( ).
- Dans le menu principal, sélectionner l'option « Saisie de données » dans le menu « Enregistrements ».

L'activation de l'une de ces options entraîne l'affichage d'une ligne vierge. On peut alors saisir les nouvelles données.
On peut saisir plusieurs lignes. Pour ce faire, utiliser le bouton « Enregistrement suivant » ( ).
La validation des données d'une ligne saisie se fait automatiquement lors de la navigation vers une autre ligne. Avant de valider les données saisies, le SGBD effectue la vérification de ces données et affiche un message approprié à l'erreur éventuelle rencontrée.

Remarque

Dans le cas d'un formulaire basé sur deux tables liées par une clé étrangère, la saisie d'un champ entraîne l'affichage automatique d'une valeur dans le champ provenant de la deuxième table.

Exemples

Dans le formulaire «Detail_commande_assist», la saisie du code article entraîne l'affichage automatique de la désignation correspondante.

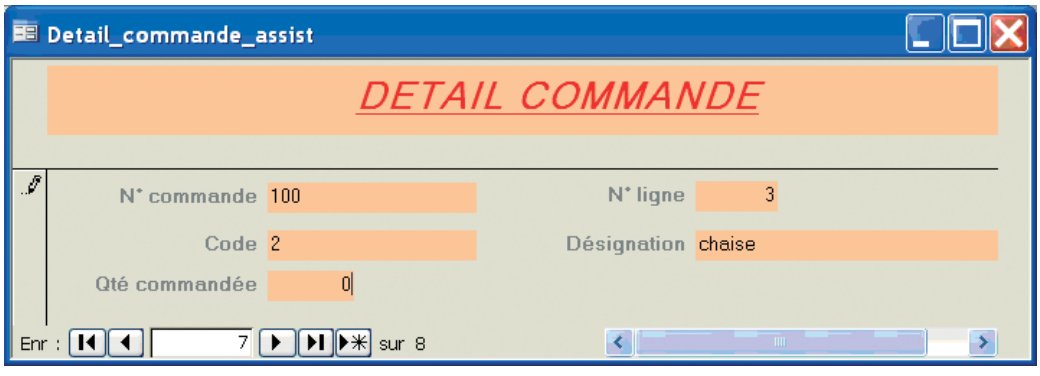


Figure 7.32 : *Saisie de données à l'aide d'un formulaire*

3.3.4 Suppression de données à travers un formulaire

Pour supprimer une ligne dans un formulaire déjà ouvert, on peut procéder comme suit :

1. Se positionner sur l'enregistrement à supprimer. Si le formulaire est de type tabulaire, et si on veut supprimer plus qu'un enregistrement, on doit d'abords marquer les enregistrements à supprimer. Pour marquer des enregistrements, maintenir la touche Shift enfoncée et cliquer à gauche des enregistrements à marquer.
2. Dans la barre d'outils, cliquer sur le bouton «Supprimer enregistrement» ( ).
3. Un message de confirmation s'affiche :

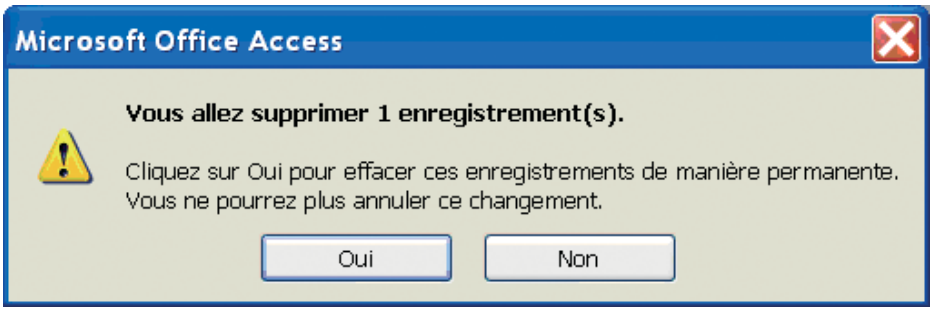


Figure 7.33 : *Confirmation de suppression de lignes*

Après confirmation, les lignes sont définitivement supprimées.

3.3.5 Fermeture d'un formulaire :

La fermeture d'un formulaire peut se faire de deux façons :

- En cliquant sur le bouton «Fermer» ( ) de la fenêtre dans laquelle s'affiche le formulaire.
- En activant l'option « Fermer » dans le menu « Fichier ».

3.4 Les sous formulaires

La plupart des formulaires que nous avons vus dans les sections précédentes sont des formulaires simples, c'est-à-dire qu'ils se basent sur une seule table.

Lorsque les données que l'on souhaite manipuler à travers un formulaire sont réparties sur des tables liées par une clé étrangère, la notion de sous formulaire constitue alors un moyen efficace pour atteindre cet objectif.

Exemple

Si on veut afficher les données relatives à une commande, celles-ci sont réparties sur les deux tables *Commande* et *Detail_commande*, tel que mentionné dans la figure suivante :

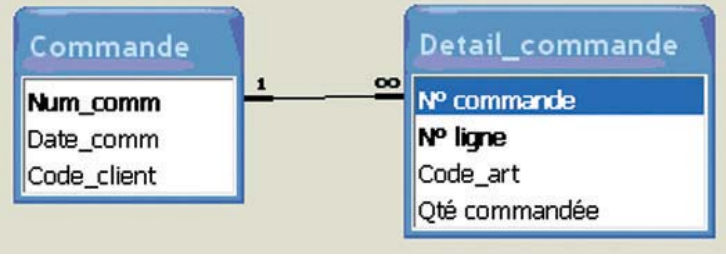


Figure 7.34 : *Tables liées par clé étrangère*

Le formulaire correspondant sera composé de deux parties : un formulaire principal (partie supérieure) contenant les données en provenance de la table *Commande* et un sous formulaire (partie inférieure) contenant les données en provenance de la table *Detail_commande.*

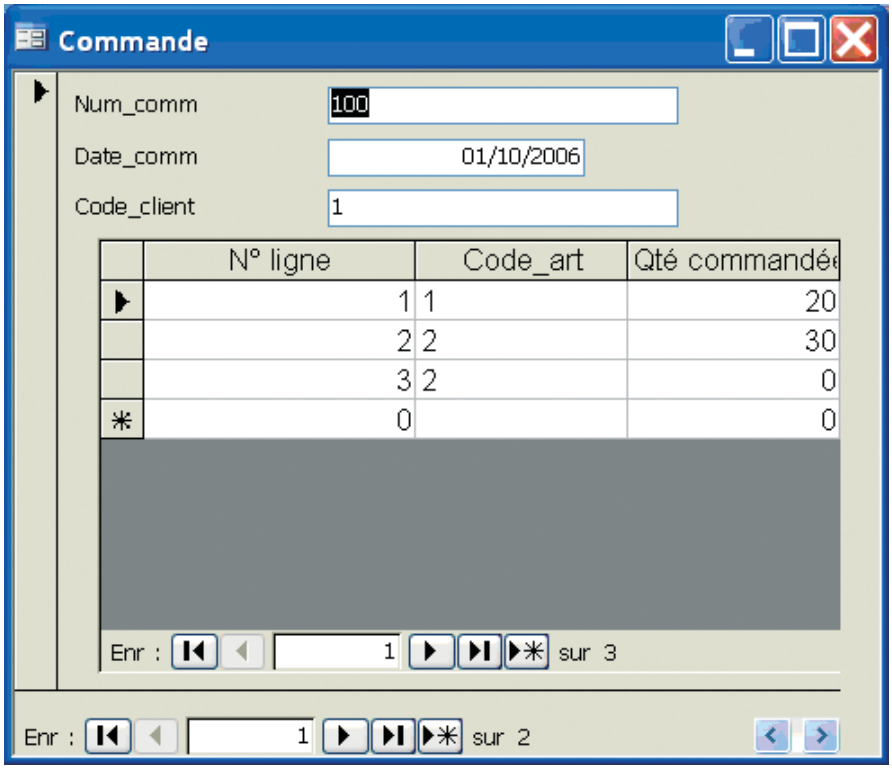


Figure 7.35 : *Exemple de sous formulaire*

Par rapport aux formulaires déjà créés dans les sections précédentes, ce formulaire présente les particularités suivantes :

- Le sous formulaire a une présentation tabulaire alors que le formulaire principal a une présentation colonnes.
- Le sous formulaire a sa propre zone de contrôle, en plus de la zone de contrôle du formulaire principal.
- Dans le sous formulaire, le champ *N° Commande* n'apparaît pas. Il est à préciser que la colonne *N° Commande* est une clé étrangère dans la table *Detail_commande*. Donc toutes les lignes affichées dans le sous formulaire sont relatives à la commande affichée dans le formulaire principal.

Malgré son apparence complexe, ce formulaire n'est pas difficile à réaliser. Sa création se fait selon la démarche «formulaire instantané» en prenant comme table de base la table *Commande* (voir § 3.2.1 Création d'un formulaire instantané).

4. Les états

4.1 Introduction

Bien que les formulaires constituent le moyen le plus courant à travers lequel les utilisateurs interagissent avec une base de données, il existe un autre moyen permettant aux utilisateurs d'extraire des données à partir d'une base de données : ce sont les états.

Un état est un programme permettant d'afficher des données en vue de les imprimer, les stocker sous forme de fichier, les envoyer comme un message électronique ou bien les exporter vers un autre outil tel qu'un logiciel de type tableur ou traitement de texte.
Il existe pas mal de ressemblances entre les opérations relatives aux états et celles relatives aux formulaires.
Nous allons voir dans les sections qui suivent comment créer et exploiter un état.

4.2 Création d'un état

Comme pour les formulaires, il existe trois façons pour créer un état :

- o Création d'un état instantané
- o Utilisation d'un assistant
- o Création libre

Ces trois façons peuvent être combinées. Généralement, on commence par la création d'un état instantané ou en utilisant un assistant, puis on finalise l'état obtenu en création libre.
Dans ce qui suit, nous allons voir ces trois modes pour créer un état simple.

4.2.1. Création d'état instantané

La création d'un état instantané est le mode le plus facile et le plus rapide pour créer un état. Il est souvent utilisé pour créer un premier prototype d'un état qui peut être finalisé par la suite.
La création d'un état instantané se fait en suivant les étapes suivantes :

1. Dans la fenêtre « Base de données», sélectionner la table sur laquelle sera basé l'état (ici la table *Detail_commande*), puis dans le menu principal, sélectionner le sous-menu nouvel objet ( ) puis l'option « État ». On peut également dans la fenêtre «Base de données» sélectionner l'objet «États», puis cliquer sur le bouton «Nouveau» (Nouveau). La fenêtre suivante s'affiche :

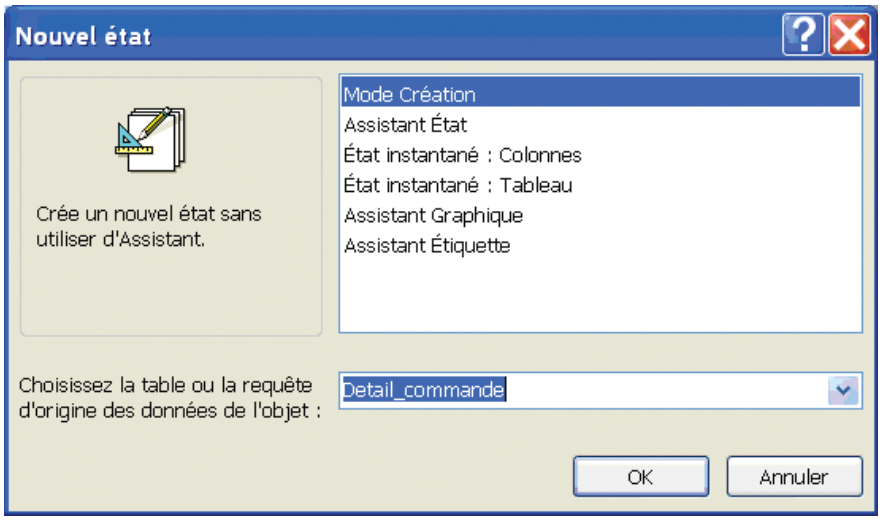


Figure 7.36 : *Création d'état instantanée*

2. Sélectionner le mode de création «État instantané colonnes» ou «État instantané tableau» en fonction du mode d'affichage selon lequel vous souhaitez que les données apparaissent dans l'état. Généralement c'est le mode tableau qui est utilisé. Dans la liste, sélectionner la table sur laquelle sera basé l'état.
3. Dés l'activation du bouton « Ok », une nouvelle fenêtre contenant le nouvel état est affichée :

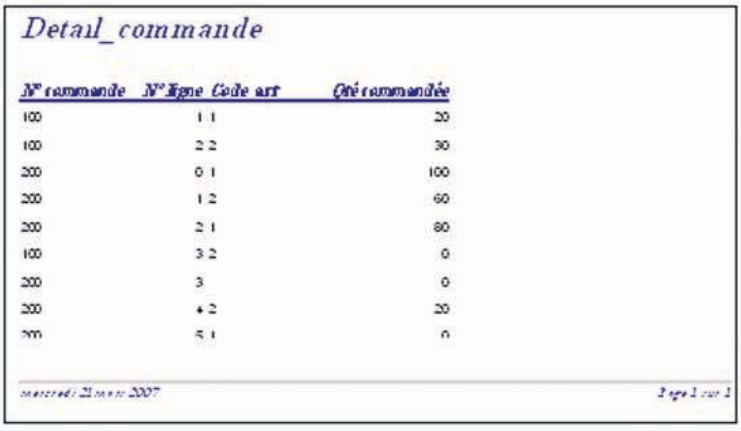


Figure 7.37 : *Exécution de l'état instantané*

4. Cliquer sur «Enregistrer» dans le menu principal. Une fenêtre vous invitant à saisir un nom pour le nouvel état s'affiche :

Figure 7.38 : *Enregistrement d'un état*

Par défaut, c'est le nom de la table qui est attribué au nouvel état. Vous pouvez attribuer un autre nom. Cliquer ensuite sur « Ok » pour enregistrer le nouvel état.

Le nouvel état est maintenant créé. Il apparaît dans la liste des états (fenêtre États).

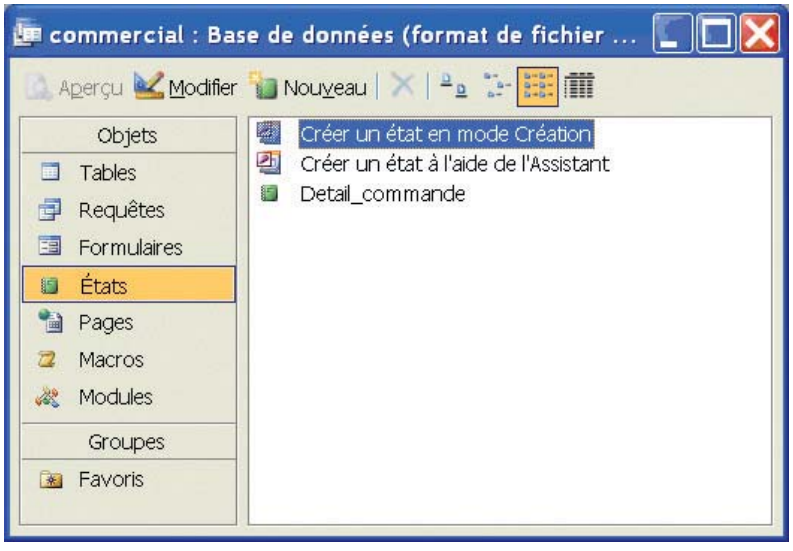


Figure 7.39 : *Enregistrement d'un état*

L'exécution de cet état se fait en double-cliquant sur le nom correspondant dans la liste des états ou bien en activant l'option « Aperçu» (Aperçu) dans la fenêtre «Base de données ».

4.2.2. Création d'état à l'aide d'un assistant

La création d'un état peut se faire également en utilisant un assistant. Ce mode prend plus de temps que le mode précédent (instantané) mais il offre plus de possibilités.

La création d'un état à l'aide d'un assistant se fait en suivant les étapes suivantes :

1. Dans la fenêtre «Base de données», sélectionner «États» dans la liste des objets, puis dans le menu de cette fenêtre sélectionner l'option « Nouveau ».
2. Dans la fenêtre «Nouvel état» qui s'affiche, sélectionner la ligne «Assistant état», puis dans la partie inférieure sélectionner la table sur laquelle sera basé le nouvel état (*Detail_commande dans notre cas*). Cliquer ensuite sur le bouton « Ok».
3. Dés l'activation du bouton « Ok», la première page de l'assistant s'affiche. Elle permet de sélectionner les champs qui vont apparaître dans le nouvel état.

   Il est à préciser que nous pouvons intégrer dans le même état des champs provenant de plus qu'une table, à condition bien sur qu'il existe des liens entre ces tables.

   Dans notre cas nous allons rajouter la colonne *Désignation Article* à partir de la table *Article*, de telle sorte que lors de l'affichage d'une ligne du détail commande, on peut voir la désignation de l'article commandé à la place du code.

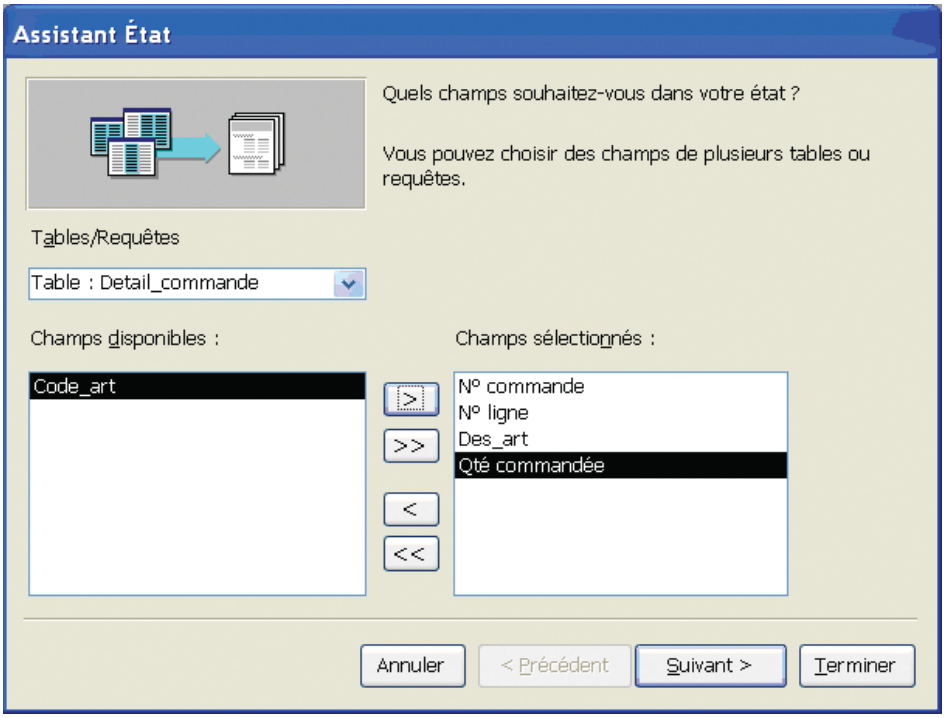


Figure 7.40 : *Assistant de création d'un état : page 1*

4. Cliquer sur le bouton « Suivant ». Si l'état est basé sur deux ou plusieurs tables, la page suivante de l'assistant s'affiche. Dans le cas où l'état est basé sur une seule table, c'est la page de l'étape 5 qui s'affiche.

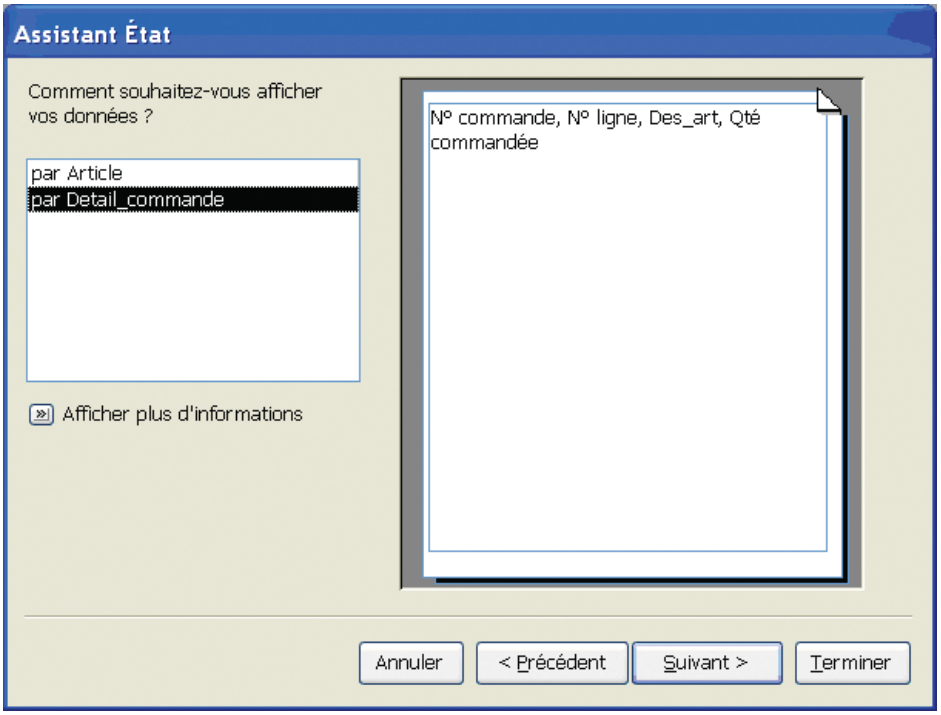


Figure 7.42 : *Assistant de création d'un état : page 2*

Cette page permet de préciser l'organisation de l'affichage des données en fonction de leur provenance (tables). Dans ce cas, nous choisissons la deuxième option (par Detail_commande). Cliquer ensuite sur le bouton «Suivant ».

5. La fenêtre qui s'affiche permet de préciser un ordre de tri pour les données. On peut préciser 0, 1 ou plusieurs champs selon lesquels les données seront triées. Pour chaque champ on peut choisir également l'ordre de tri (croissant ou décroissant).

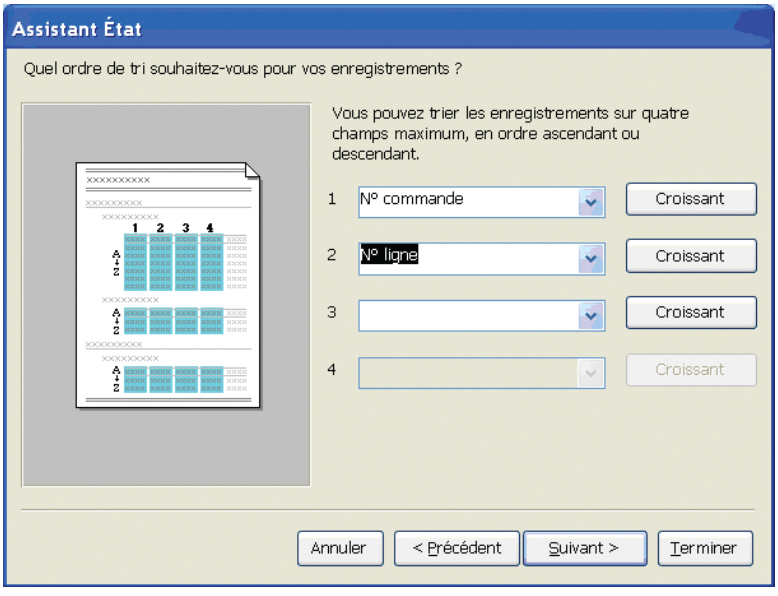


Figure 7.43 : *Assistant de création d'un état : page 3*

6. En cliquant sur « Suivant», la page suivante de l'assistant s'affiche. Elle permet de choisir la disposition des données (verticale, tabulaire ou justifiée) et l'orientation de la page (portrait ou paysage).

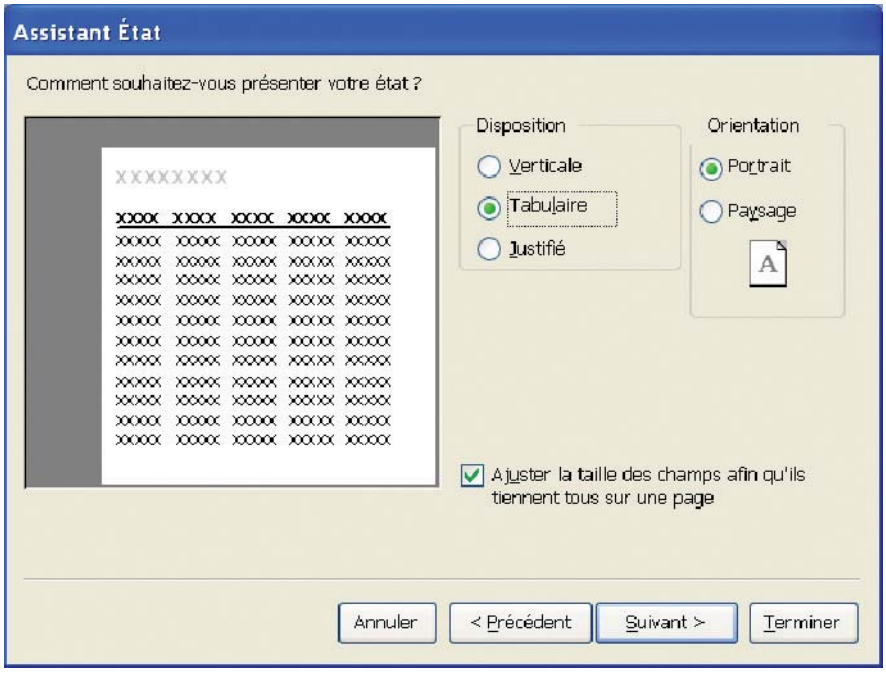


Figure 7.44 : *Assistant de création d'un état : page 4*

7. En cliquant sur « Suivant », la page suivante de l'assistant s'affiche. Elle permet de choisir le style de présentation de l'état.

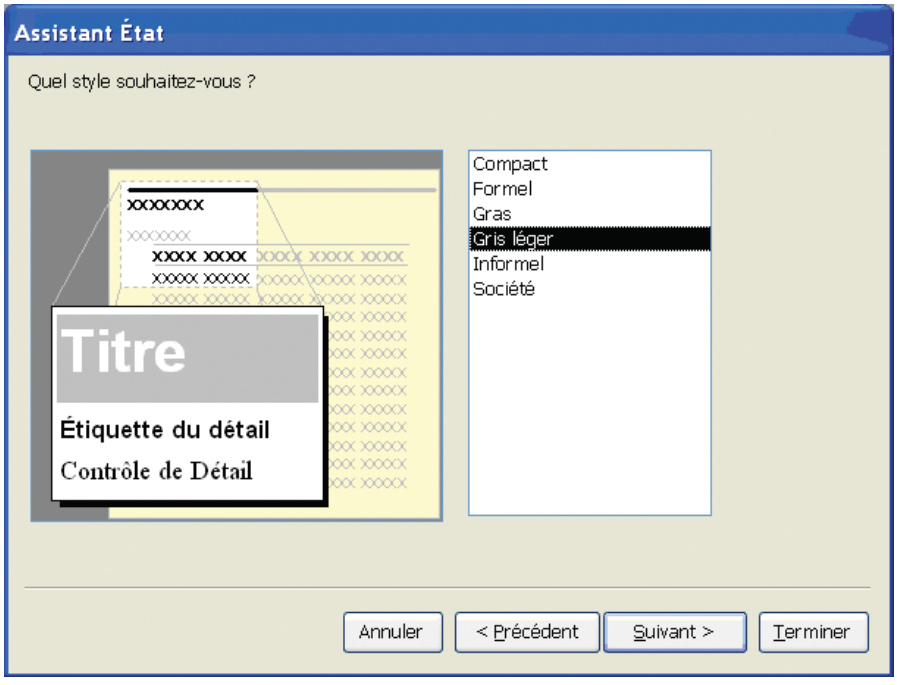


Figure 7.45 : *Assistant de création d'un état : page 5*

8. La dernière page de l'assistant permet d'attribuer un nom au nouvel état :

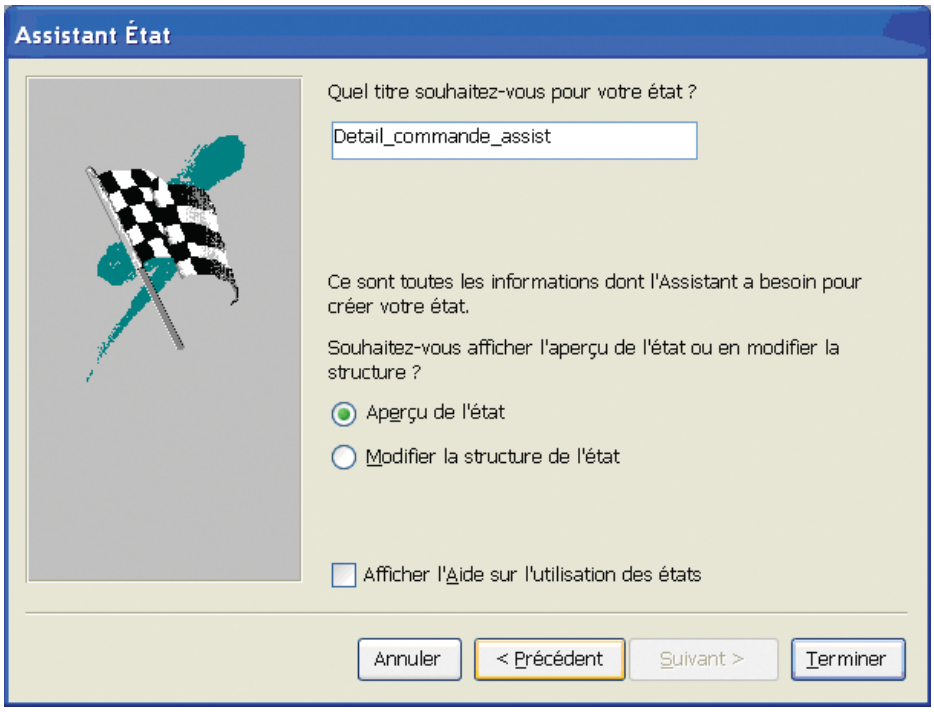
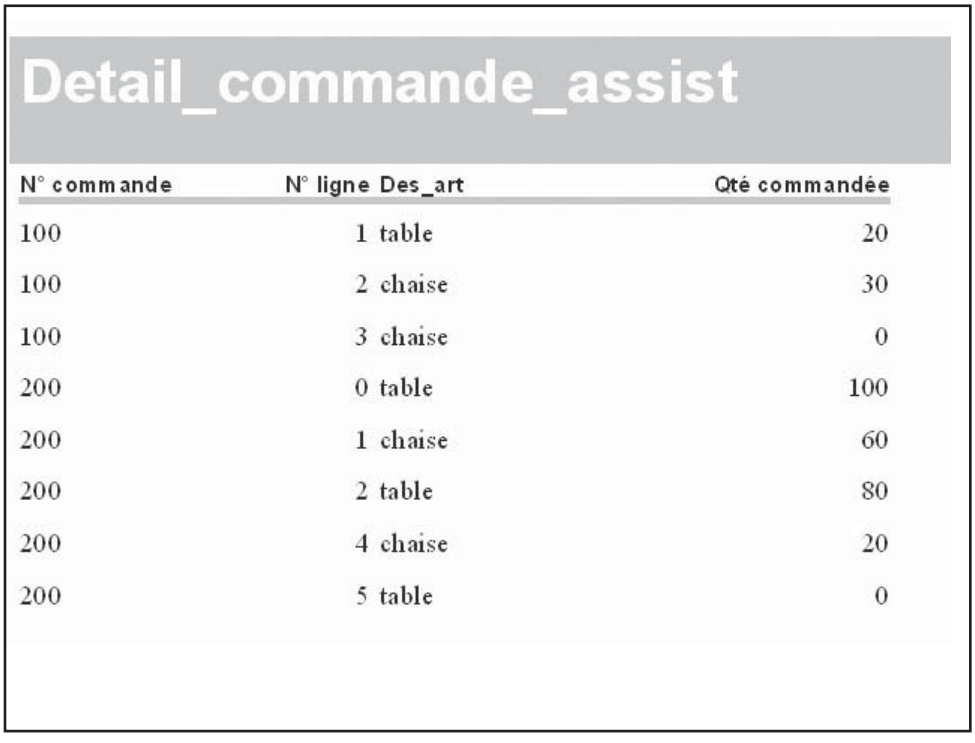


Figure 7.46 : *Assistant de création d'un état : page 6*

9. Cliquer sur le bouton «Terminer ». Le nouvel état s'affiche.



Detail_commande_assist

N° commande	N° ligne	Des_art	Qté commandée
100	1	table	20
100	2	chaise	30
100	3	chaise	0
200	0	table	100
200	1	chaise	60
200	2	table	80
200	4	chaise	20
200	5	table	0

Figure 7.47 : *Résultat de l'état créé à l'aide d'un assistant*

Noter la différence par rapport au premier état que nous avons créé : pour chaque ligne de commande, la désignation est affichée à la place du code article. Les données sont triées par numéro de commande puis par numéro de ligne.

3.2.2. Modification d'un état

Les deux modes de création d'états que nous avons vus (état instantané et assistant) permettent de créer automatiquement des états sans laisser beaucoup de possibilités à celui qui réalise cette opération. On ne peut pas par exemple choisir l'emplacement des champs, l'intitulé de ces champs, leurs types (liste, case à cocher, …).
Ces deux modes ont généralement utilisés uniquement pour entamer le développement d'états. Les états obtenus ne constituent qu'une première ébauche de l'application finale.
Nous allons voir dans cette section comment affiner un état en modifiant sa structure et sa présentation.
La modification d'un état se fait en suivant les étapes suivantes :

1. Dans la fenêtre «Base de données», sélectionner «États» dans la liste des objets, l'état à modifier, puis dans le menu de cette fenêtre sélectionner l'option «Modifier».

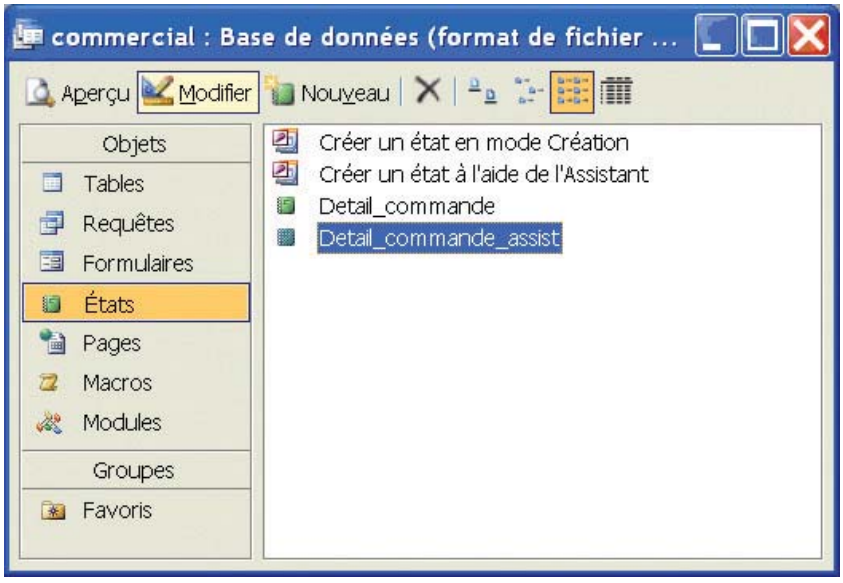


Figure 7.48 : *Liste des états*

2. Dés l'activation de l'option «Modifier», un éditeur d'états s'affiche :

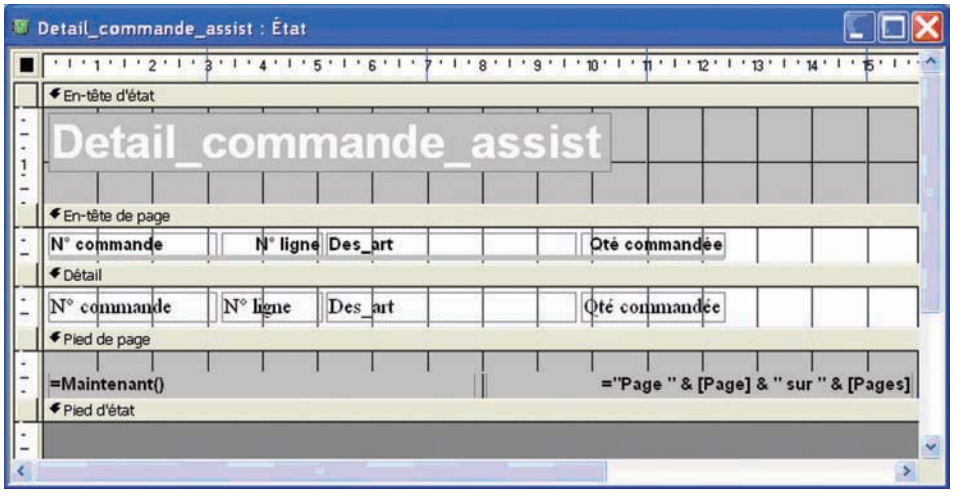


Figure 7.49 : *Éditeur d'états*

A l'aide de cet éditeur on peut modifier la présentation de l'état. On peut modifier par exemple :

- Le titre de l'état
- Le libellé de certains champs : Des_art → Désignation
- La disposition des champs : justifier à gauche le numéro de la ligne.
- La taille des champs : réduire la taille du champ Qté commandée.

3. Pour modifier le titre de l'état :
   - Sélectionner le texte et l'éditer directement.
   - Modifier le style d'affichage de ce titre à l'aide des outils de mise en forme de texte dans le menu principal (police, taille, couleur, …).

4.Pour modifier le libellé d'un champ :
   - Sélectionner le champ concerné, puis éditer le texte de ce champ.
   - On peut également modifier le style du champ (police, taille, couleur, justification, …).

5. Pour modifier la disposition d'un champ, il suffit de le sélectionner et de choisir le style d'alignement à l'aide du bouton correspondant dans la boite à outils.

6. Pour modifier la taille d'un champ, il faut le sélectionner puis modifier sa taille en largeur et/ou en hauteur.
   Le résultat de ces modifications apparaît dans la fenêtre suivante :

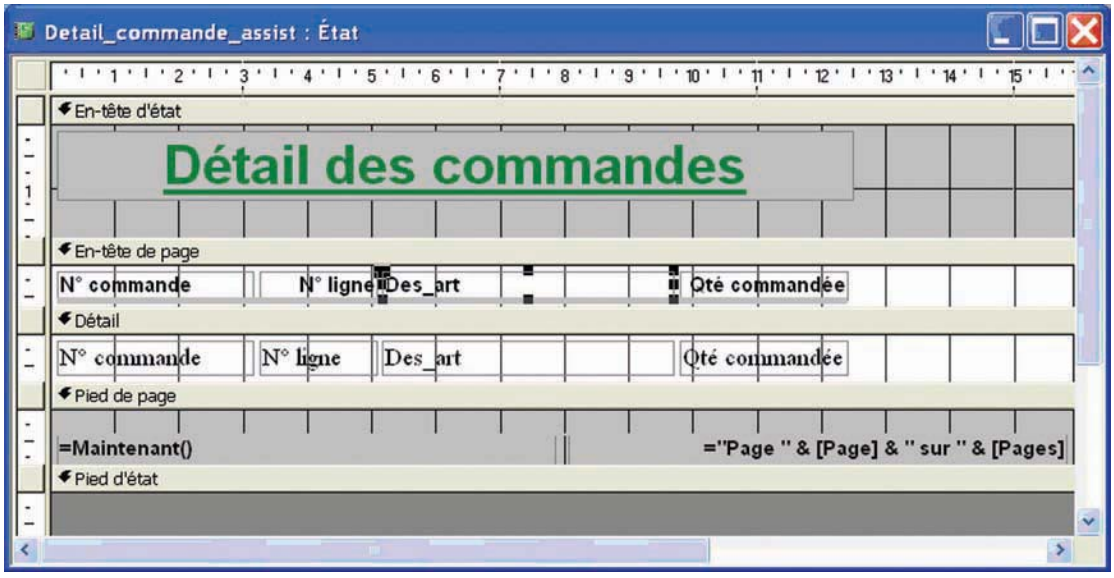


Figure 7.50 : *Résultat de l'édition d'un état*

Pour valider ces modifications, cliquer sur le bouton « Enregistrer » du menu principal puis fermer la fenêtre d'édition d'états.
L'exécution de cet état apparaît comme suit :

Détail des commandes

N° commande	N° ligne	Des_art	Qté commandée
100	1	table	20
100	2	chaise	30
100	3	chaise	0
200	0	table	100
200	1	chaise	60
200	2	table	80
200	4	chaise	20
200	5	table	0

Figure 7.51 : *Exécution d'un état*

4.3 Exploitation d'un état

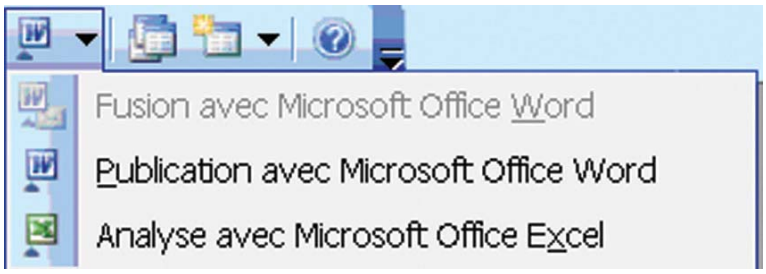
L'exploitation d'un état est plus simple que l'exploitation d'un formulaire car la seule opération possible à travers un état est l'affichage des données.

Le lancement d'un état peut se faire de différentes façons :

- Sélectionner l'état puis double-cliquer.
- Sélectionner l'état puis activer le bouton «Aperçu».
- Créer un raccourci (option «Créer raccourci» du menu «Édition») et exécuter l'état directement en double-cliquant sur le raccourci.

Lorsque le résultat de l'état est affiché, on peut effectuer l'une des opérations suivantes :

- Imprimer le contenu : choisir l'option «Imprimer » dans le menu «Fichier» ou cliquer sur le bouton «Imprimer» ( ).
- Exporter le contenu vers un traitement de texte ou un tableur : cliquer sur le bouton correspondant :



- Envoyer le document en tant que pièce jointe d'un courrier électronique : choisir l'option «Envoyer vers» dans le menu «Fichier».

5. Interaction entre base de données et sites web dynamiques

5.1 Introduction

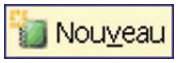
Les données diffusées à travers des pages web proviennent dans la plus part des cas de bases de données. Ces pages sont dites des pages dynamiques par opposition aux pages statiques dont le contenu est figé.

Par exemple un site web qui affiche un catalogue des produits alimente la page correspondante à partir d'une base de données.

Nous allons voir dans la section suivante, d'une façon très brève, comment alimenter une page web dynamique à l'aide d'un assistant. Il est à noter qu'il existe d'autres façons pour effectuer cette opération.

5.2 Alimentation de pages dynamiques

La façon selon laquelle se fait l'alimentation d'une page dynamique à partir d'une base de données varie en fonction du SGBD et du langage avec lequel la page est écrite. Pour créer une page dynamique dont le contenu provient de la base de données, on peut procéder comme suit :

1. Dans la fenêtre «Base de données», sélectionner la table sur laquelle sera basé l'état (ici la table *Detail_commande*), puis dans le menu principal, sélectionner le sous-menu nouvel objet () puis l'option «Pages». On peut également dans la fenêtre «Base de données » sélectionner l'objet «Pages», puis cliquer sur le bouton «Nouveau» (). La fenêtre suivante s'affiche :

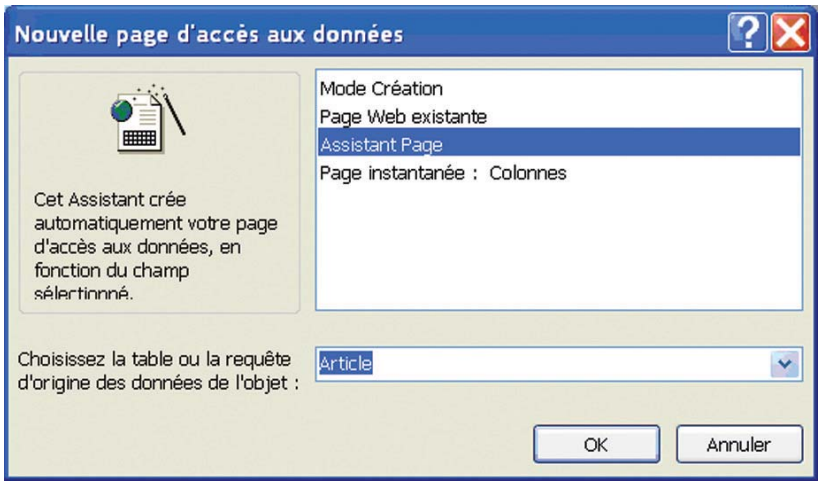


Figure 7.52 : *Assistant de création de page web : page 1*

2. Sélectionner le mode de création «Assistant Page». Dans la liste, sélectionner la table sur laquelle sera basé la page.
3. Dés l'activation du bouton «Ok», la première page de l'assistant s'affiche. Elle permet de sélectionner les champs qui vont apparaître dans la nouvelle page.

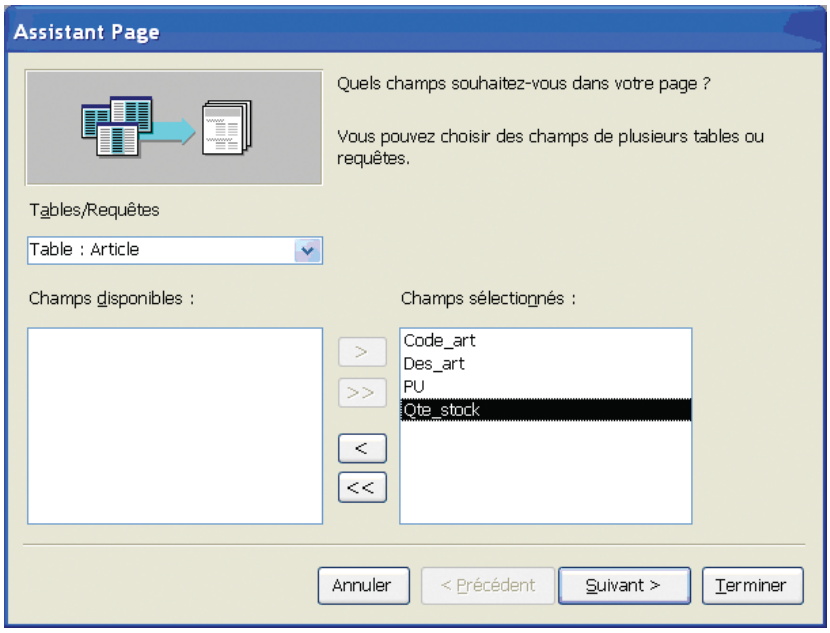


Figure 7.53 : *Assistant de création de page web : page 2*

4. La fenêtre qui s'affiche permet de préciser un ordre de tri pour les données. On peut préciser 0, 1 ou plusieurs champs selon lesquels les données seront triées. Pour chaque champ on peut choisir également l'ordre de tri (croissant ou décroissant).

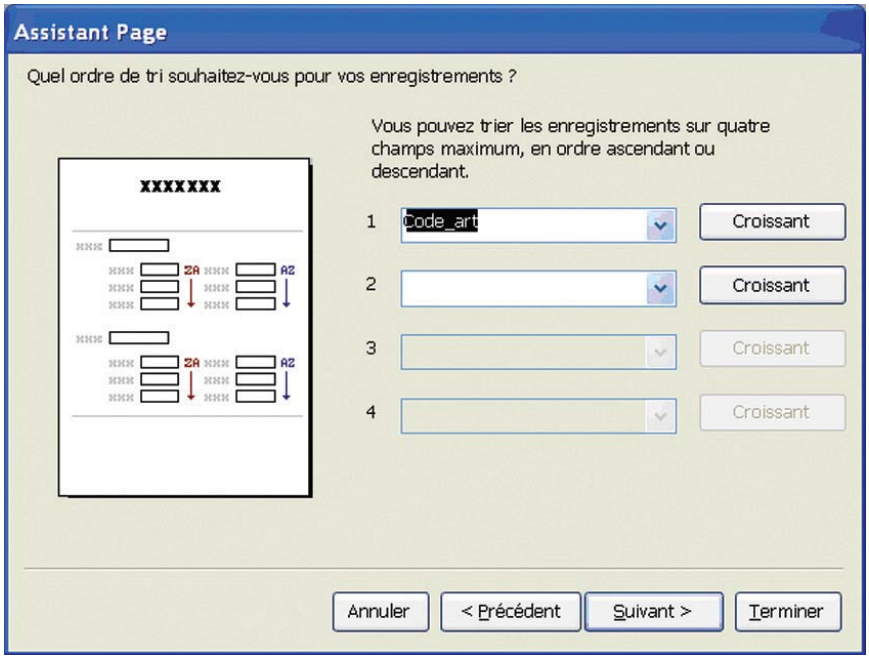


Figure 7.54 : *Assistant de création de page web : page 3*

5. La dernière page de l'assistant permet d'attribuer un nom à la nouvelle page :

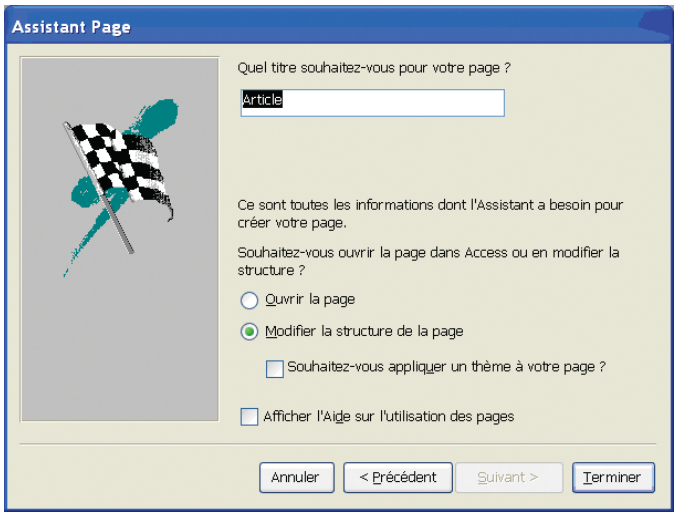


Figure 7.55 : *Assistant de création de page web : page 4*

6. Cliquer sur le bouton «Terminer ». La nouvelle page s'affiche.

Figure 7.56 : *Éditeur de pages web*

On peut modifier l'apparence de la nouvelle page en précisant un titre et en modifiant l'emplacement et les propriétés des éléments de la page.
Voici la page après modification :

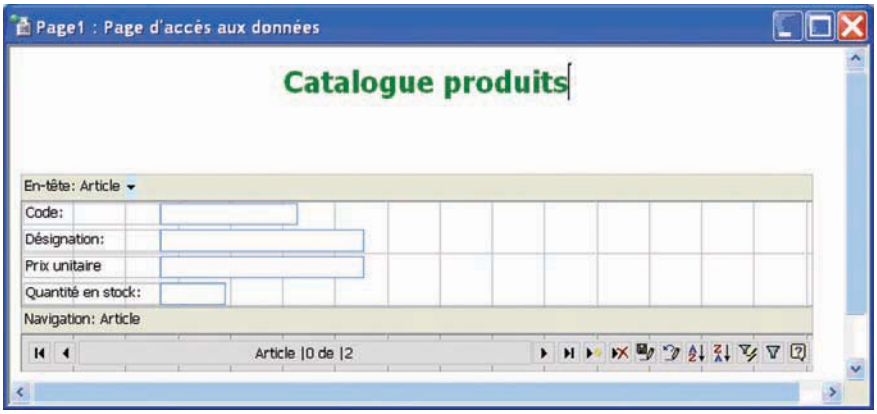


Figure 7.56 : *Résultat de l'édition de page web*

7. On termine par l'enregistrement de la page. Elle aura comme nom «Article.html». Un raccourci à cette page est créé dans la fenêtre «Base de données» :

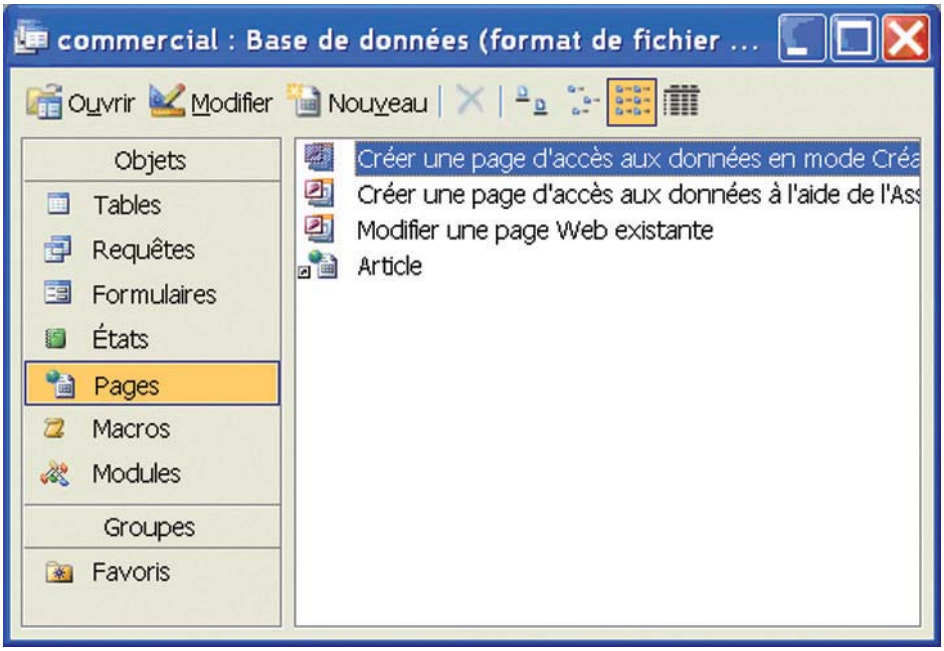
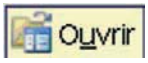


Figure 7.57 : *Liste de pages web*

L'exécution de la page que nous venons de créer se fait en double-cliquant sur le nom correspondant dans la liste des pages ou bien en activant l'option «Ouvrir» () dans la fenêtre «Base de données».

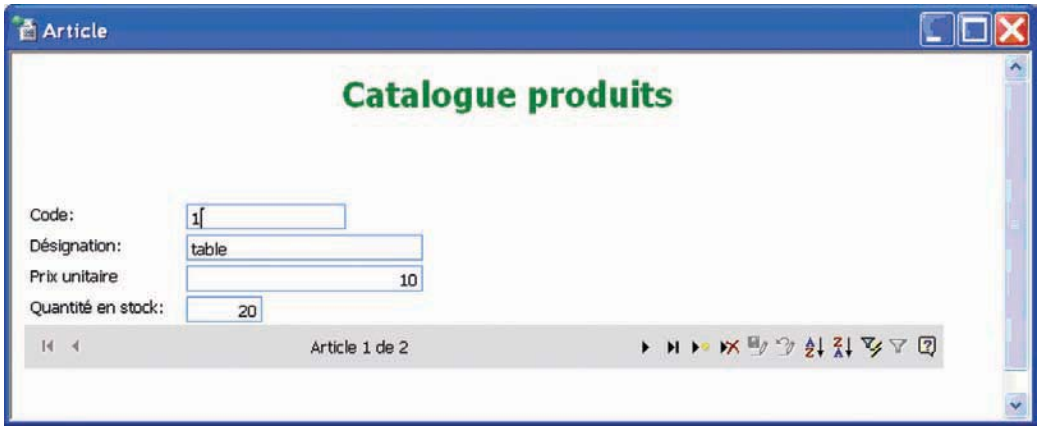


Figure 7.58 : *Exécution de page web*

On remarque en bas de cette page la présence d'une barre d'outils permettant la navigation entre les lignes, l'insertion et la suppression de lignes, l'enregistrement et l'annulation des mises à jour, le tri des données (croissant et décroissant) et enfin le filtrage des données.

RETENONS

- ✓ Vous avez appris dans ce chapitre comment développer une application autour d'une base de données. Les composantes de cette application sont les formulaires, les états et les pages Web.
- ✓ Les formulaires offrent aux utilisateurs un moyen facile pour interagir avec une base de données en consultant, insérant, modifiant ou supprimant des données.
- ✓ Les états permettent aux utilisateurs de visualiser, imprimer et éventuellement exporter des données provenant d'une base de données.
- ✓ Les pages Web dynamiques permettent de rajouter au contenu d'une page Web des données provenant d'une base de données.

EXERCICES

Exercice 1

Reprenez l'exercice 2 du chapitre précédent (Chapitre 6) et créer un formulaire à l'aide de l'assistant, avec un sous-formulaire pour les réservations dont se chargent les cuisiniers.

Exercice 2

Un vidéoclub dispose d'un grand nombre des cassettes vidéo. Le volume de ces ressources est devenu si important que la tâche de recherche des films est devenue une tâche laborieuse.
Pour résoudre ce problème, le responsable du vidéoclub décide d'automatiser la gestion des prêts de cassettes vidéo. Chaque cassette peut faire l'objet d'un prêt d'au maximum 24h. Au delà, le responsable décide du montant d'amende que le client doit payer pour ce retard.
Le responsable du vidéoclub souhaite en outre, garder l'histoire de tous les prêts accomplis par ces clients à fin d'accorder une remise pour ses clients les plus fidèles.

La structure de la base de données :

Cassette (Code_Cas, Titre_Film, Realisateur, Heros, Date_Sortie, Etat, Duree, Type)
Abonné (Code_Ab, Nom_Ab, Prenom_Ab, Tel, Adresse)
Emprunt (Code_Cas, Code_Ab, Date_Emprunt, Date_Retour)

Manipulation 1 : Création des tables

Table	Champs	Type
Cassette	Code_Cas	Entier long (clé)
	Titre_Film	Texte (40)
	Heros	Texte (40)
	Date_Sortie	Mémo
	Duree	Date
	Type	Assistant liste choix
Abonné	Code_Ab	Entier long (clé)
	Nom_Ab	Texte (30)
	Prenom_Ab	Texte (20)
	Tel	Texte (10)
	Adresse	Texte (50)
Emprunt	Code_Cas	Entier long (clé)
	Code_Ab	Entier long (clé)
	Date_Emprunt	Date (clé)
	Date_Jour	Date

Remarque : Pour le type «assistant liste des choix», choisir : policier, sciences fiction, suspens, dessins animés, comédie.

Manipulation 2 : **Création des formulaires Fabonné et Fcassette**

Créer les formulaires «Fabonné» et «Fcassette» puis saisir les lignes des tables «Cassette» et «Abonné» par le biais de ces formulaires.

Table « CASSETTE »						
Code	**Titre**	**Réalisateur**	**Héros**	**Date Sortie**	**Durée**	**Type**
1	Défi	Bob Swaim	Gregory smith, John Hurt	01/01/1995	2h30	Policier
2	Le saint	Philip	Val Kilmer, Elisabet Shue	02/05/1993	1h30	Suspens
3	Les aventures	Stéphane	Blake, Mortimer	20/02/1989	2h30	Dessins animés
4	Serial lovers	James	Michèle Laroque	05/11/1993	1h30	Comédie
5	Vampires	John	James Woods, Daniel Baldwin	20/09/1989	3h00	Sciences fiction
6	Kissed	Lynne	Molly Parker, peter	15/08/1995	3h00	Suspens
7	Tout le monde	Woody	Woody Allen, Julia Roberts	13/05/1993	2h30	Comédie
8	Poursuivre	Andrew	Kean Reevers, Morgan Freeman	20/04/1993	3h00	Suspens
9	Grève party	Fabien	Daniel Russo, Vincent Elabaz	28/07/1993	1h30	Comédie

Table « Abonné »				
Code	**Nom**	**Prénom**	**Tél**	**Adresse**
4851236	Ben Ahmed	Ali	71.120.356	Sousse
5230148	Salhi	Amine	71.270.614	M'Saken
1235647	Mannoubi	Hatem	71.558.665	Akouda
6982513	Moussa	Monia	71.987.021	Akouda
1230458	Ben Sassi	Amor	71.561.447	Sidi Bou Ali
7891245	Ben Nejma	Montassar	71.480.645	Sidi Bou Ali
8921325	Radi	Ahmed	71.987.854	Kalaa Kbira
4581202	Mabrouki	Nabil	71.746.217	M'Saken
5558960	Ben Mahmoud	Donia	71.225.559	Hammam Sousse

Manipulation 3 : Création du formulaire principal «Femprunt» et le sous formulaires «Sfemprunt»

Créer le formulaire «Femprunt» puis saisir les lignes de la table «emprunt» par le biais de ce formulaire.

Table « EMPRUNT »			
Code_Cas	**Code_Abonné**	**Date_Emprunt**	**Date_Retour**
1	4851236	01/01/2005	02/01/2005
2	4851236	01/01/2005	02/02/2005
3	1235647	02/02/2005	03/03/2005
4	6982513	14/03/2005	15/03/2005
5	1230458	05/04/2005	07/04/2005
6	7891245	01/05/2005	02/05/2005
7	8921325	02/05/2005	03/05/2005
8	8921325	02/05/2005	03/05/2005
1	8921325	02/05/2005	03/05/2005
2	6982513	03/05/2005	04/05/2005
5	7891245	04/05/2005	
6	5230148	05/05/2005	
9	6982513	06/05/2005	

Manipulation 4 : **Interrogation de la base**

1. Créer la requête « **R_Abonné_Donné** » qui nous permet, en donnant le code de l'abonné, d'afficher : le code de l'abonné, son nom, son prénom, son numéro de téléphone et son adresse.
2. Créer la requête «**R_Cassette_Donnée** » qui nous permet d'afficher les informations concernant une cassette donnée.
3. Créer la requête «**R_Emprunt_Retard**» qui nous permet de calculer le nombre de jours pour chaque retard d'un prêt.

Manipulation 5 : **Formulaire basé sur les requêtes**

Créer les formulaires suivants :

1. **F_Emprunt_Retard** basé sur la requête **R_Emprunt_Retard**
2. **F_Abonné_Donné** basé sur la requête **R_Abonné_Donné**
3. **F_Cassette**_Donné basé sur la requête **R_Cassette_Donné**

Manipulation 6 : **Création des états**

1. Créer l'état «**E_Emprunt_Retard**» en se basant sur la requête «**R_Emprunt_Retard** ».
2. Créer de l'état « **E_Cassette** », en se basant sur la table Cassette.
3. Créer de l'état « **E_Abonné** », en se basant sur la table Abonné.

Manipulation 7 : Création d'un formulaire Menu

Créer un formulaire «Menu» qui contiendra des boutons de commandes permettant d'ouvrir ou de fermer les formulaires déjà réalisés.

Chapitre 8

Sécurisation d'une Base de Données

Objectifs :

- Découvrir les concepts fondamentaux de la sécurité des Bases de Données
- Etre capable de mettre en œuvre des techniques de contrôle d'accès à des Bases de Données
- Assurer en différents modes, la sécurité d'une Bases de Données

Plan :

1. Problématique

La préservation de la confidentialité est devenue une priorité pour les citoyens ainsi que pour les administrations. Le besoin d'accumuler, de partager et d'analyser des données personnelles est multiple : pour rendre plus simples et efficaces les procédures administratives, pour l'amélioration de la qualité des soins grâce au dossier médical électronique, pour personnaliser les services rendus par une grande quantité d'objets électroniques dans un environnement d'intelligence ambiante ou même pour la lutte contre le terrorisme (croisement de bases de données commerciales et gouvernementales pour la recherche de suspects). Bien que le traitement de données personnelles ait généralement un but louable, il constitue une menace sans précédent aux droits élémentaires à la protection de la vie privée.

Partout dans le monde, les gouvernements adoptent des lois spécifiques pour cadrer l'utilisation de données personnelles. Il est cependant difficile de traduire ces lois en moyens technologiques convaincants garantissant leur application.

Comme l'atteste le rapport « *Computer Crime and Security Survey* » établi par le Computer Security Institute et le FBI[1] , le nombre d'attaques de serveurs de bases de données est croissant malgré la mise en place de politiques de sécurité de plus en plus drastiques.

1.1. Les piliers de la sécurité des BD

On envisage souvent la sécurité sous un angle fermé, essentiellement celui de la confidentialité. Mais bien d'autres concepts sous-tendent la sécurité.

- **Authentification**

Elle consiste à s'assurer de l'identité d'un utilisateur avant de lui donner l'accès à une base de données (identification par mot de passe, login, certification etc.).

- **Confidentialité**

Tout n'est pas accessible à tout le monde. Se connecter à la base de données, donne un certain nombre de droits et de ressources en fonction d'un profil défini et maintenu par un administrateur .

- **Disponibilité**

Faculté de délivrer correctement un service en termes de délai et de qualité à l'utilisateur. Des mécanismes de sauvegarde variés doivent être mis en place pour assurer la disponibilité (mécanismes de journalisation, de reprise permettent de restaurer une information sans pratiquement aucune perte, …)

- **Intégrité**

Que les données soient réparties ou non (dans ce dernier cas les mécanismes mis en jeux seront plus complexes) elles doivent être cohérentes. Cela sous entend, d'une part que les accès concurrents d'utilisateurs, notamment lors de mises à jour,

---

[1] Computer Security Institute. 'CSI/FBI Computer Crime and Security Survey', 2004. http://www.gocsi.com/forms/fbi/pdf.html.

ne doivent pas compromettre ***la cohérence des données*** et d'autre part que ces dernières satisfassent aux contraintes d'intégrité du modèle, et / ou aux règles de gestion de l'entreprise.

- **Traçabilité**

  En cas de problème important ou d'attaque du système, on peut recourir à l'analyse de traces ou de journaux. Le niveau de détail de ces traces est paramétrable, et concerne soit les aspects système, soit le réseau, soit l'accès aux données élémentaires elles-mêmes.

1.2. Les mécanismes mis en œuvre pour la sécurité

Les SGBD se doivent de fournir un certain nombre de mécanismes internes ou de fonctionnalités assurant un niveau satisfaisant de sécurité.

- **L'authentification**, est le processus qui permet de vérifier qu'un utilisateur réclamant un accès est bien celui qu'il prétend être, ou plus simplement le processus qui contrôle l'identité de l'utilisateur. Cette action (dite aussi login) se fait en général via la fourniture du couple nom d'utilisateur / mot de passe. Dans certains cas l'authentification peut être implicite et héritée d'une authentification précédente, ou reconnue automatiquement (Adresse IP de l'utilisateur sur le réseau par exemple), bien que simplifiant les accès de ce choix peut évidemment s'avérer dangereux.
- **Les droits et privilèges** : une fois correctement identifié, l'utilisateur doit pouvoir accéder aux informations et ressources auxquelles il a droit. Ce problème est résolu le plus simplement avec la gestion de droits élémentaires accordés à un individu, ou plus efficacement avec des rôles et/ou profils affectés à des groupes d'individus.
- **Les LOGs ou traces** : permettent d'enregistrer tout ou partie des informations concernant les accès (réussis ou échoués). Cette trace pourra être plus ou moins verbeuse et son volume étroitement surveillée. De ce fait, on l'utilisera de manière ciblée sur des périodes de temps spécifiques.
- **Tolérance aux pannes** : permet par du matériel ou du logiciel redondant (CPUs, disques, …) de supporter, de manière partiellement ou complètement transparentes, différents types de pannes, tant au niveau du client, que du réseau, que du serveur. Une tolérance totale a bien sûr un coût certain.
- **Sauvegarde et restauration** : sauvegarder les données sur des supports externes (disques, CDs, etc.) et pouvoir les restituer, les plus à jour possible. Le but est de ne pas perdre de données suite à un problème matériel (panne disque), logiciel (bug) ou une fausse manipulation d'un utilisateur.
- **Mécanismes transactionnels** : l'atomicité des transactions, par définition, assure la cohérence des données, même dans des environnements distribués. L'image avant modification, stockée de manière redondante dans ou hors de la BD, permet de faire d'éventuels retours arrière pour retrouver le dernier état cohérent, ou de s'assurer qu'il n'y a pas eu d'opérations partielles ou incomplètes.

2. Gestion des droits d'accès

2.1. Définir un mot de passe pour une base mono-utilisateur

Pour empêcher l'accès à une base mono-utilisateur, il est conseillé de définir un mot de passe sur cette base. Si, au contraire, la base est multiutilisateur, il est préférable d'avoir recours à un mot de passe pour chaque utilisateur.

Activité 1

❶ Charger une base de données existante puis définir un mot de passe sur cette base.

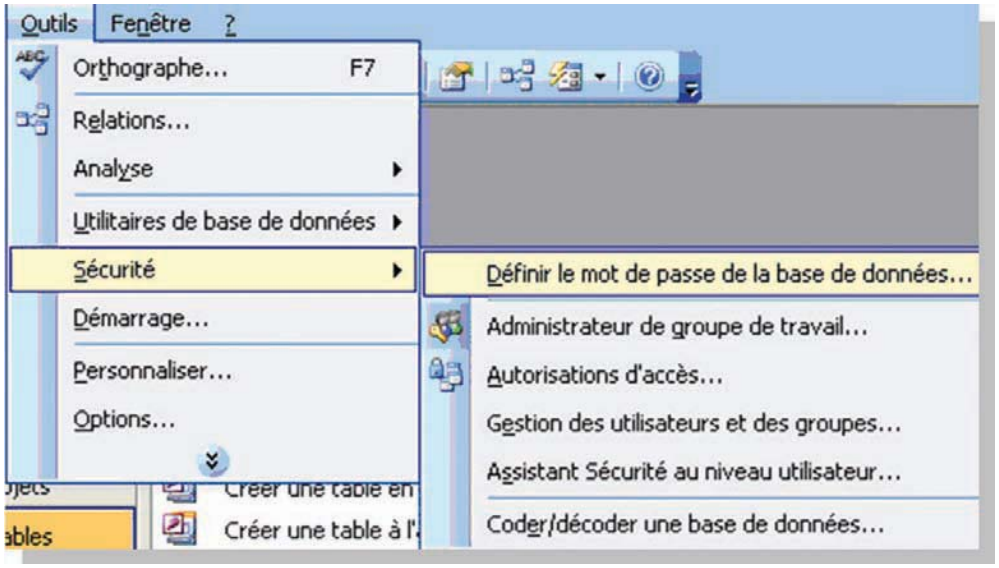
❷ Vérifiez lors de l'ouverture de la base de données qu'un mot de passe est demandé.

Marche à suivre :

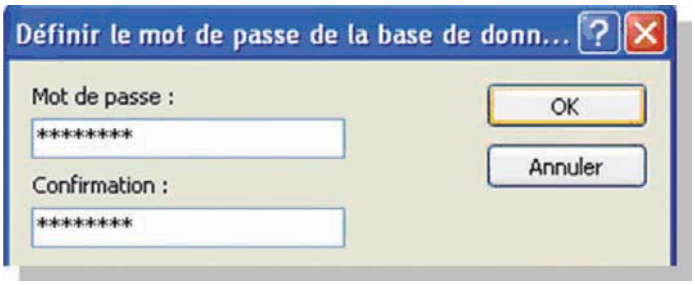
Pour définir un mot de passe sur une base de données, on peut procéder comme suit :

1. Ouvrez la base de données en mode exclusif (menu *Fichier - Ouvrir*).

2. Allez dans le menu *Outils - Sécurité - Définir le mot de passe de la base de données...*



3. Choisissez un mot de passe.



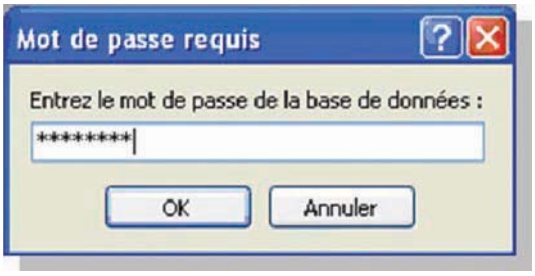
Les règles suivantes sont à appliquer sous Access pour le choix d'un mot de passe :

- ✓ Entre 1 et 20 caractères (un minimum de 8 caractères est recommandé).
- ✓ «Case sensitive » (un mot de passe doit contenir au moins des minuscules, des majuscules, des chiffres et des symboles).
- ✓ Ne peut pas commencer par un espace et ne peut pas contenir les symboles suivants : \ [ ] : | < > + = ; , . ? * (utilisez les symboles # ou $ par exemple).
- ✓ Un mot de passe ne doit appartenir à aucun dictionnaire.

4. Valider

Constatons

Lors de l'ouverture de la base de données qu'un mot de passe est demandé.



2.2. Sécuriser les accès à une base de données

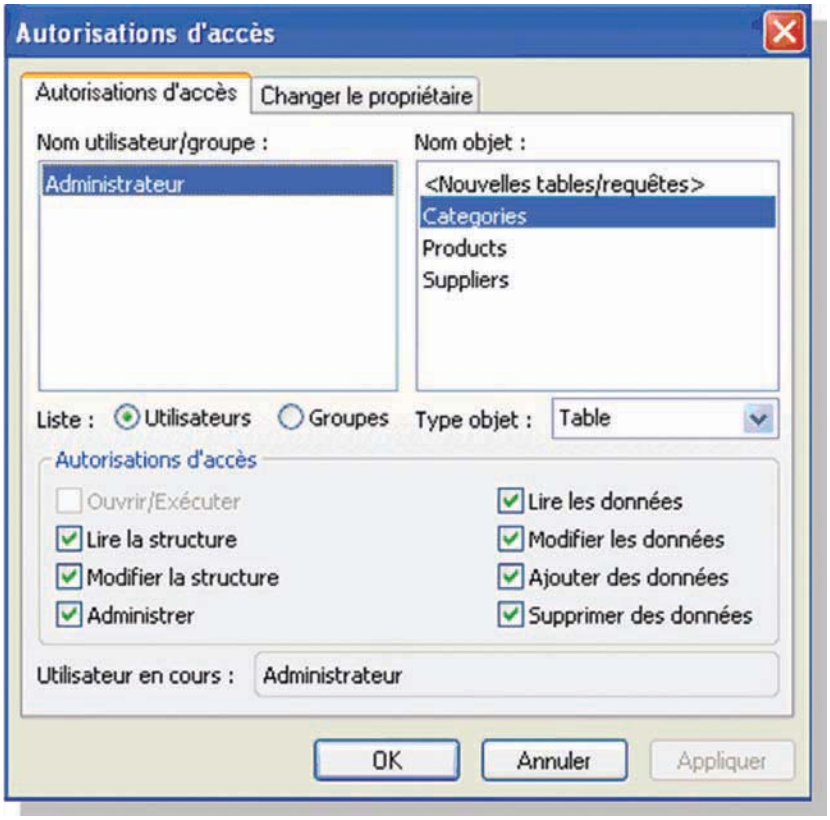
Activité 2

Charger une base de données existante puis définir les droits d'accès à cette base.

Marche à suivre :

Pour définir les droits d'accès à une base de données, on peut procéder comme suit :

1. Ouvrez la base de données.
2. Activez le menu Outils, la commande Sécurité puis l'option Autorisations d'accès.
3. À partir de la fenêtre affichée, fixez les droits d'accès à votre base.
4. Validez.



3. Cryptage d'une base de données

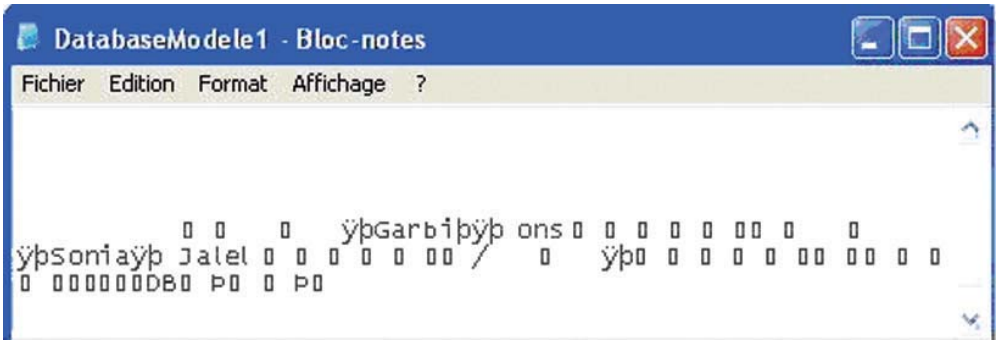
Une base de données n'est qu'un document binaire. Si une personne non autorisée récupère ce document, à défaut de pouvoir l'ouvrir si un mot de passe a été mis en place, elle pourra toujours l'ouvrir avec un éditeur de texte, et y récupérer quelques mots clés ou informations sur notre base de données. Crypter la base de données permet de rendre le déchiffrement par un éditeur de texte totalement impossible.

Activité 3

❶Ouvrir un fichier .mdb non crypté avec un éditeur de texte. Que constatez-vous ?
❷ Crypter cette base de données
❸Ouvrir le fichier .mdb crypté avec un éditeur de texte. Que constatez-vous ?

Constatons

En ouvrant un fichier .mdb non crypté avec un éditeur de texte, on peut constater que l'on peut y retrouver des données.



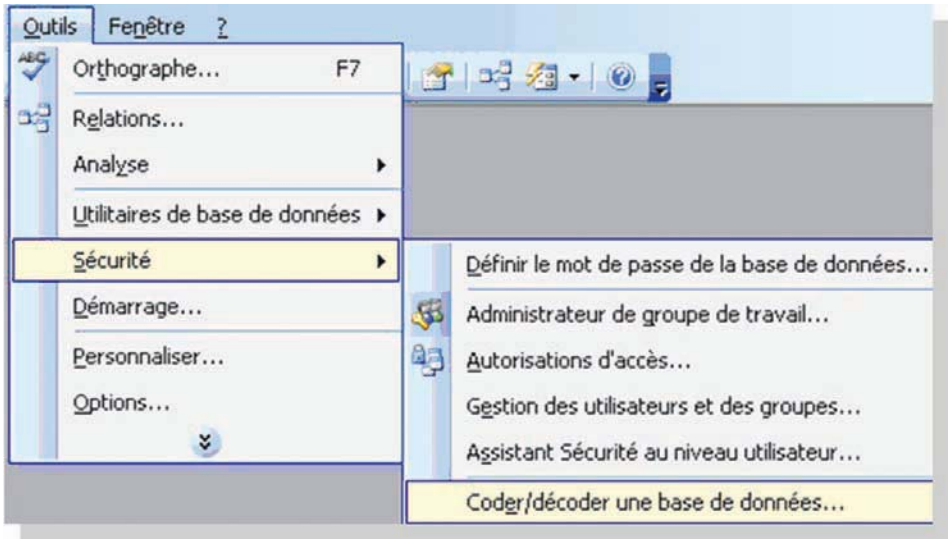
Marche à suivre :

Pour crypter une base de données, on peut procéder comme suit :

1. Allez dans le menu Outils

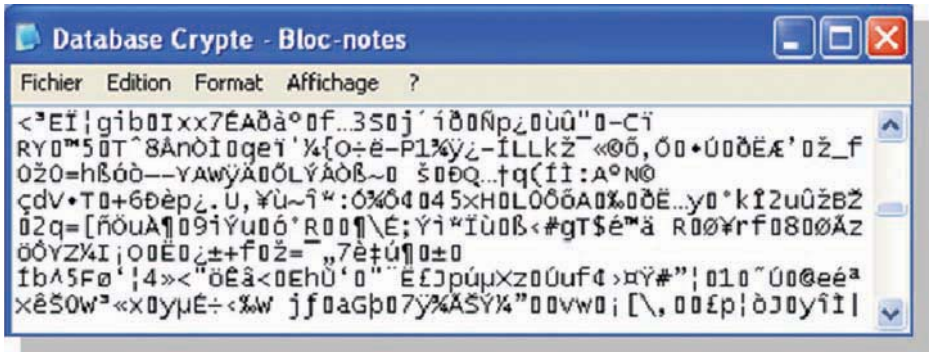
2. Activez la commande Sécurité

3. Sélectionnez l'option Coder/Décoder une base de données....



Constatons

En ouvrant le fichier .mdb crypté avec un éditeur de texte, on peut constater qu'il est devenu illisible.



4. Gestion des utilisateurs

Lorsqu'une base de données multiutilisateur grandit, elle peut être utilisée par plusieurs personnes ayant des fonctions différentes et n'ayant pas besoin de l'intégralité de la base de données pour travailler. Le service commercial aura besoin de la partie commerciale et le service ressources humaines de la partie gestion des employés par exemple. Il convient alors de mettre en place des groupes de travail pour contrôler les droits d'accès aux données et les modifications de structure de la base. Puis de créer des utilisateurs et de les assigner à un ou plusieurs de ces groupes de travail.

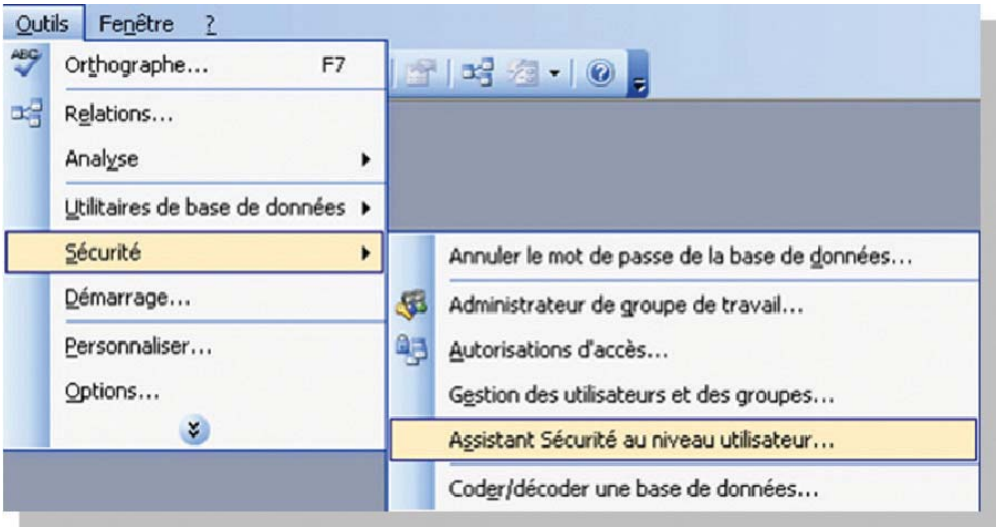
4.1. Via l'assistant

Activité 4

1. A l'aide de l'assistant, créer des groupes de travail.
2. Créer des utilisateurs et assigner les à un ou plusieurs des groupes déjà prédéfinis.

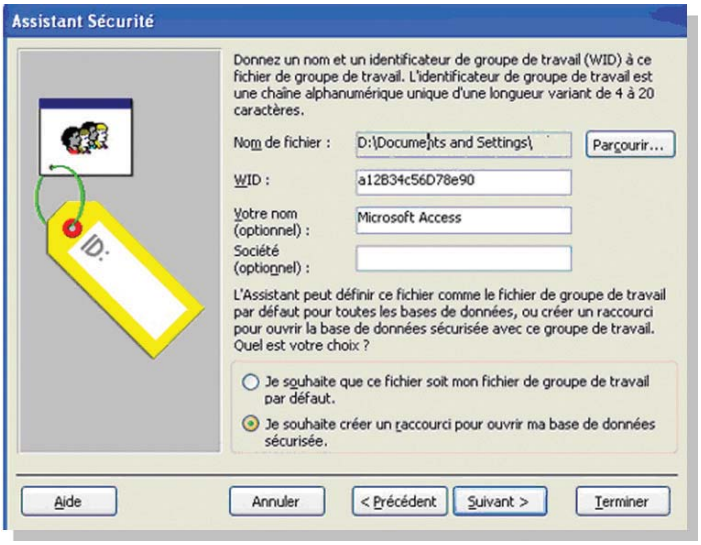
Marche à suivre :

Pour créer des groupes de travail via l'assistant, on peut procéder comme suit :
1. Allez dans le menu *Outils - Sécurité - Assistant Sécurité au niveau utilisateur...*

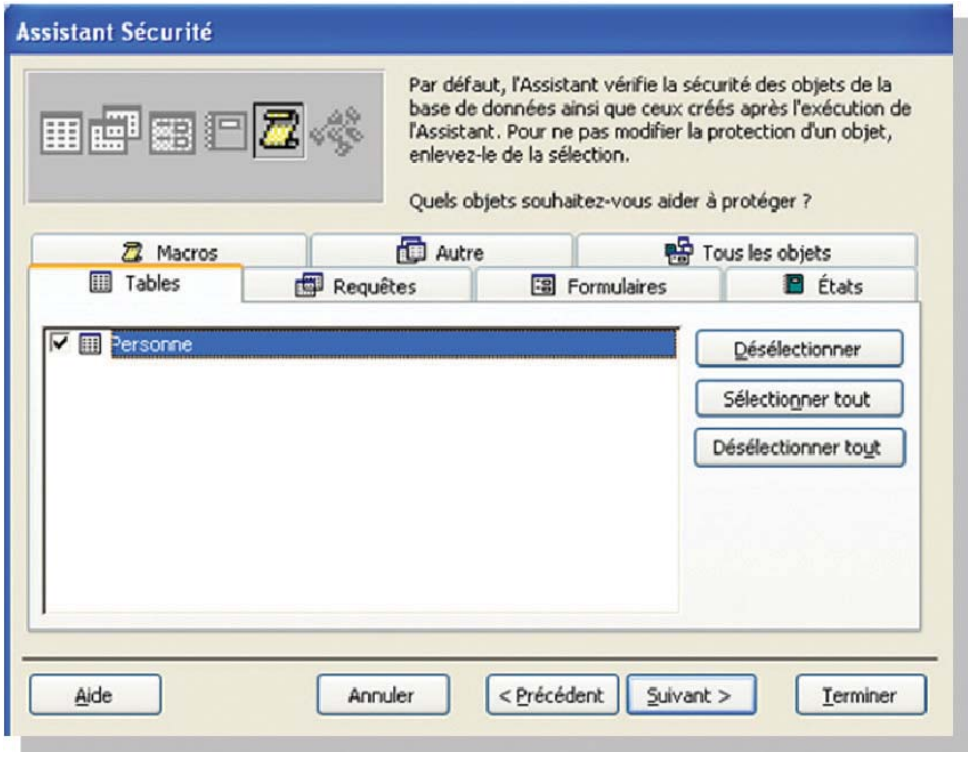


2. Choisissez l'option «Créer un nouveau fichier de groupe de travail». Un fichier de groupe de travail porte l'extension .mdw.

3. Choisissez le nom du fichier de groupe de travail et son identifiant unique (WID). Ensuite, vous avez le choix entre placer un raccourci sur le bureau, qui ouvrira la base de données en appliquant les stratégies de sécurité contenues dans ce fichier de groupe de travail, ou d'assigner par défaut ce fichier de groupe de travail à la base de données. Nous choisirons dans l'exemple le raccourci sur le bureau.



4. Choisissez les objets sur lesquels vont s'appliquer les stratégies définies ultérieurement



5. Choisissez les groupes de travail prédéfinis dans lesquels on assignera les utilisateurs. Les permissions sur ces groupes pourront être changées dans la procédure manuelle, et l'on pourra également créer de nouveaux groupes.

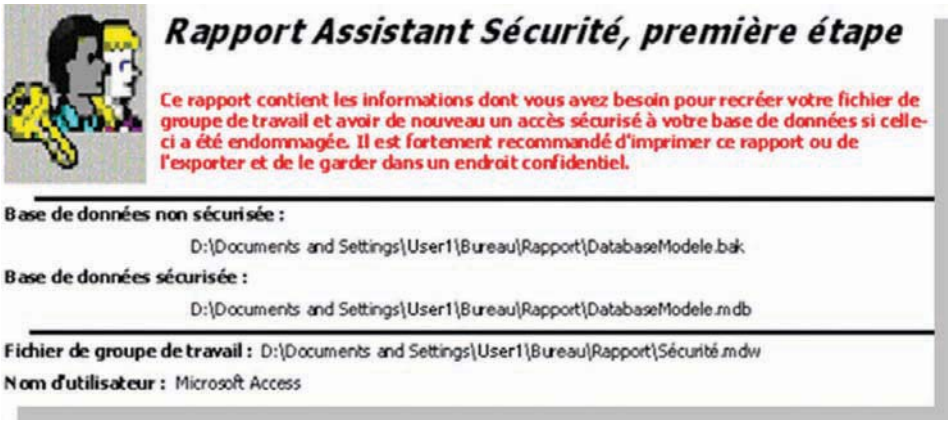
6. Par défaut, tous les utilisateurs appartiennent au groupe « Utilisateurs ». On peut décider de donner des autorisations par défaut à ce groupe, si l'on souhaite par exemple, que tous les utilisateurs de la base de données aient le droit de lecture sur les tables.

7. Créez les utilisateurs un par un, en leur assignant un mot de passe et un identifiant unique (PID).

8. Assignez des groupes de travail à vos utilisateurs ou vice-versa.

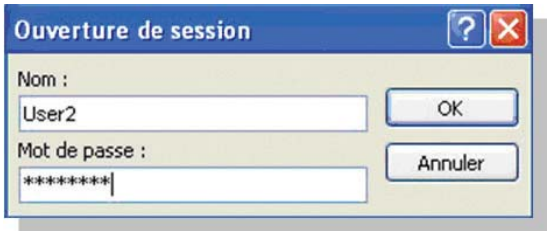
9. Spécifiez le chemin d'une copie non sécurisée de la base de données, au cas où la nouvelle base créée ne vous conviendra pas.

10. Imprimez et stockez soigneusement et de manière sécurisée le rapport assistant sécurité. Il contient les emplacements des bases de données et du fichier de groupe de travail, les groupes de travails utilisés, les utilisateurs créés avec leurs mots de passe, les permissions assignées aux utilisateurs.



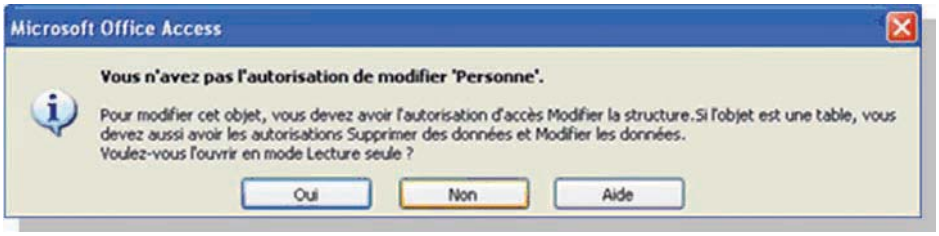
Constatons

En ouvrant la base, après fermeture, grâce au raccourci placé sur le bureau, on peut constater qu'un compte d'ouverture de session est demandé.



Constatons

Si l'utilisateur User2 tente d'ajouter une colonne à la table "Personne", le message suivant apparaît. Les droits de modification de structure de la base n'ont pas été accordés à l'utilisateur User2.



4.2. Manuellement

Il est possible sans utiliser l'assistant de créer son propre fichier de groupe de travail, de créer ses propres groupes de travail, de leur assigner des permissions et de les appliquer à des utilisateurs. Cela permet également de vérifier ou d'affiner le résultat de l'assistant.

Marche à suivre :

Pour créer des groupes de travail via l'assistant, on peut procéder comme suit :

1. Pour créer un fichier de groupe de travail, allez dans le menu *Outils - Sécurité - Administrateur de groupe de travail...*
2. Pour créer un groupe de travail, un utilisateur, pour assigner un utilisateur à un groupe de travail, ou pour changer le mot de passe d'un utilisateur, allez dans le menu *Outils - Sécurité - Gestion des utilisateurs et des groupes...*
3. Pour vérifier ou modifier les permissions assignées à un groupe, allez dans le menu *Outils - Sécurité - Autorisations d'accès...*

5. Intégrité des données

L'application de l'intégrité des données garantit la qualité des données stockées dans la base. Par exemple, si un employé est défini avec un **ID d'employé** égal à **123**, la base de données ne doit pas autoriser qu'un autre employé ait la même valeur d'ID. Si une colonne **grade** est destinée à accueillir des valeurs comprises entre **1** et **5**, la base de données ne doit pas accepter de valeur en dehors de cette plage. Si la table contient une colonne **dept_id** qui stocke le numéro de service de l'employé, la base de données ne doit accepter que des valeurs correspondant bien aux identificateurs de service de la société.

Lors de la planification des tables, deux étapes importantes consistent d'une part à identifier les valeurs valides pour une colonne et d'autre part à décider de la façon d'appliquer l'intégrité des données dans cette colonne. L'intégrité des données se répartit entre les catégories suivantes :

- Intégrité d'entité : L'intégrité d'entité définit une ligne comme étant une entité unique pour une table particulière
- Intégrité de domaine : L'intégrité de domaine fait référence à la fourchette (au domaine) des entrées valides pour une colonne spécifique.

- Intégrité référentielle : L'intégrité référentielle préserve les relations définies entre les tables lors de l'insertion ou de la suppression de lignes.
- Intégrité définie par l'utilisateur : L'intégrité définie par l'utilisateur vous permet de définir des règles propres à l'entreprise, qui n'appartiennent à aucune des autres catégories d'intégrité. Toutes les catégories d'intégrité acceptent l'intégrité définie par l'utilisateur.

6. Sauvegarde et restauration de bases de données

Le composant de sauvegarde et restauration apporte une sécurité essentielle pour la protection des données cruciales stockées dans les bases de données. L'implémentation d'une stratégie de sauvegarde et de restauration correctement planifiée permet de protéger les bases de données contre toute perte de données due à des dégâts provoqués par une série de défaillances. Vous pouvez tester votre stratégie en restaurant un ensemble de sauvegardes et récupérer votre base de données pour vous préparer à réagir efficacement en cas de sinistre.

Une copie de données qui peut être utilisée pour restaurer et récupérer des données porte le nom de *sauvegarde*. Les sauvegardes permettent de restaurer les données après une défaillance. Avec des sauvegardes appropriées, vous pouvez effectuer des récupérations après de nombreux types d'échecs, notamment :

- défaillance du support ;
- erreurs utilisateur (telles que la suppression d'une table par inadvertance) ;
- défaillances matérielles (telles qu'un lecteur de disque endommagé ou la perte permanente d'un serveur) ;
- catastrophes naturelles.

Par ailleurs, il peut être utile d'effectuer des sauvegardes d'une base de données afin d'effectuer des tâches administratives de routine, telles que la copie d'une base de données d'un serveur vers un autre, la configuration de la mise en miroir de base de données et l'archivage.

7. Contrôle de données dans le langage SQL

7.1. Création d'utilisateurs

L'administrateur d'une base de données peut créer à tout moment un nouvel utilisateur à l'aide de la commande suivante :

```
CREATE USER nom_utilisateur
    IDENTIFIED BY mot_de_passe ;
```

L'utilisateur créé ne dispose d'aucun droit. Il ne peut même pas se connecter à la base de données.

7.2. Attribution des droits

Il existe deux types de droits (ou privilèges) :

• *des droits globaux sur la base de données dits* ***droits système*** : Ce sont des droits permettant à leurs détenteurs d'effectuer des opérations globales sur la base de données, tel que la connexion, la sauvegarde de la base de données.
La commande SQL permettant d'attribuer un droit système à un utilisateur est la suivante :

```
GRANT droit1, droit2, ..., droitn
    TO utilisateur1, utilisateur2, ..., utilisateurp
    [WITH ADMIN OPTION]
```

On peut utiliser **PUBLIC** pour désigner tous les utilisateurs.
L'option WITH ADMIN OPTION autorise le nouvel utilisateur à accorder les droits reçus à d'autres utilisateurs

• *des droits sur des objets de la base de données dits* ***droits objet*** : Ce sont des droits permettant à leurs détenteurs d'effectuer des opérations sur des objets la base de données, tel que des tables, des vues.
La commande SQL permettant d'attribuer un droit objet à un utilisateur est la suivante :

```
GRANT droit1, droit2, ..., droitn
    ON objet
    TO utilisateur1, utilisateur2, ..., utilisateurp
[WITH GRANT OPTION]
```

Il est à noter que toute table n'est initialement accessible que par l'utilisateur qui l'a créée.
On peut utiliser **ALL** pour désigner tous les droits et **PUBLIC** pour désigner tous les utilisateurs.

• Quelques exemples de droits objet : ALTER, DELETE, INDEX, INSERT, UPDATE, SELECT, ou aussi ALL pour toutes les opérations.
• L'option WITH GRANT OPTION autorise le nouvel utilisateur à accorder les droits reçus à d'autres utilisateurs.

Exemple

1. L'utilisateur IG21 décide d'attribuer à l'utilisateur ID22 le droit de sélection et de mise à jour à sa table commandes, il tape :

```
GRANT SELECT, UPDATE
ON commandes
TO IG22 ;
```

2. L'utilisateur IG22 peut créer une table cde à partir de cette commande:

```
CREATE TABLE cde
AS SELECT * FROM ID21.commandes ;
```

7.3. Retrait des droits

La commande qui permet de supprimer un ou plusieurs droits sur un objet est :

REVOKE $droit_1$, $droit_2$, ..., $droit_n$
[ON objet]
FROM $utilisateur_1$, $utilisateur_2$, ...,$utilisateur_n$

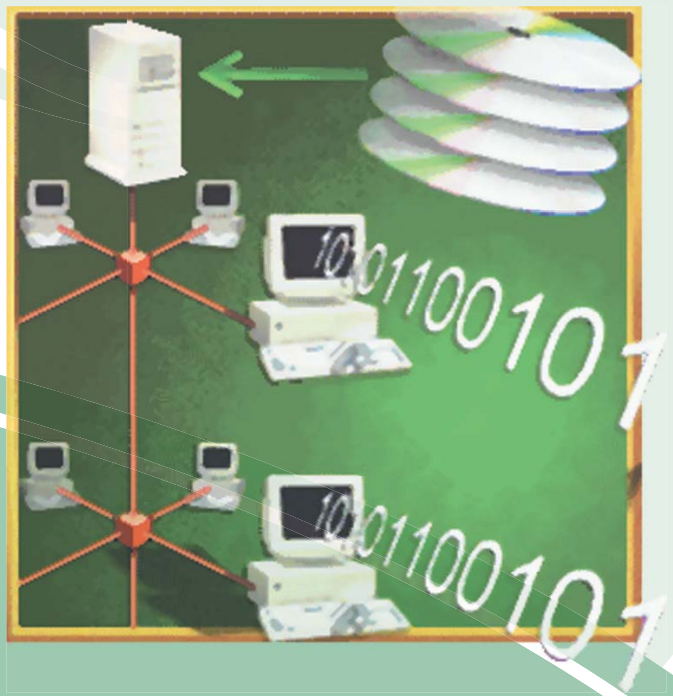
Cette commande permet de retirer des droits système ou objet. La clause ON objet est utilisée uniquement en cas de retrait de droit objet.
Avec **REVOKE** aussi on peut utiliser **ALL** pour désigner tous les droits et **PUBLIC** pour désigner tous les utilisateurs.

Exemple

```
REVOKE ALL
        ON commande
        FROM PUBLIC ;
```

Partie 4

APPLICATION
ETUDES DE CAS



Application 1 : Gestion d'une agence de location de voiture

Application 2 : Gestion d'un club vidéo

Application 3 : Gestion d'un établissement scolaire

Projets

Application 1

Gestion d'une agence de location de voiture

Enoncé :

Une agence « LV » de location de voitures gère manuellement son parc, composé d'une centaine de véhicules à partir d'un paquet de fiches cartonnées.
Ci-après, un exemple de fiche de voiture :

Ø	Numéro de l'immatriculation	:	124 TN 4951
Ø	La marque	:	Peugeot
Ø	Le type	:	205
Ø	La couleur	:	bleu
Ø	La puissance	:	6 CV
Ø	Le kilométrage	:	45000
Ø	L'âge de la voiture	:	2 ans
Ø	Prix par jour	:	65 dinars
Ø	Type carburant	:	Essence/Diesel

Le responsable de la société « LV » décide d'implanter une base de données pour améliorer la gestion de son parc de voitures.
Après étude, une voiture est décrite par le numéro d'immatriculation comme identifiant, une marque, un type, une couleur, une puissance, un kilométrage, un age de voiture, un prix par jour et un type de carburant.
Chaque locataire est identifié par un numéro locataire et chacun a un nom et une adresse.
A chaque location un enregistrement sera effectué : Le numéro d'immatriculation de la voiture, le numéro du locataire et la date de location.
Le kilométrage de retour et la date de retour (date fin location) seront enregistré au retour de la voiture.

Questions

I. Partie théorique :

1. Etablir la liste des colonnes.
2. Etablir la liste des tables.
3. Affecter les colonnes aux tables et proposer une description textuelle.
4. Préciser les clés primaires des tables.
5. Identifier les liens entre les tables.

II. Partie pratique : à traiter en mode assisté et en mode commande

1. En utilisant le SGBD disponible dans votre laboratoire, implémenter cette base.
2. Saisir, à votre choix, quelques informations dans les tables.
3. Représenter la structure de cette base de données sous forme graphique.
4. Créer les requêtes suivantes :

• **Requête1** : définir une requête qui permet d'afficher la liste, selon les numéros d'immatriculation, des voitures.

- **Requête2** : définir une requête qui permet d'afficher la liste des locataires.
- **Requête3** : définir une requête qui permet d'afficher les noms des locataires, par ordre alphabétique, qui ont loué des voitures pendant le mois de janvier de l'année en cours.

5. Réaliser les formulaires de saisie et les états que vous jugez nécessaires ainsi qu'un menu principal pour toute l'application.

Eléments de correction

I. Partie théorique :

1. Liste des colonnes :

La liste des colonnes est déduite à partir de l'énoncé et ce en mettant en valeur les propriétés. Ces propriétés seront regroupées dans le tableau qui suit.

Dans la première colonne, il est recommandé d'utiliser les mêmes règles de nommage qu'en programmation (éviter les caractères accentués, réduire la taille des noms des colonnes, etc.).

Liste des colonnes							
Nom colonne	**Description**	**Type de données**	**Taille**	**Obligatoire**	**Valeur par défaut**	**Valeurs autorisées**	**Sujet**
IMMAT	Numéros d'immatriculation	Caractère	10	O			Voiture
MARQUE	Marque de la voiture	Caractère	10	O			Voiture
TYPE	Type de la voiture	Caractère	10	O			Voiture
COULEUR	Couleur de la voiture	Caractère	10	N			Voiture
PUISS	Puissance de la voiture	Numérique	2	N			Voiture
KILOM	Le kilométrage indiqué par le compteur	Numérique	10	O			Voiture
AGE	Age de la voiture	Numérique	2	N			Voiture
PRIX_PAR_JOUR	Prix de location par jour	monétaire	4				Voiture
TYPE_CARBURANT	Type de carburant de la voiture	Caractère	10				Voiture
LOC	Numéro de locataire	Caractère	10	O			Locataire
LOCNOM	Nom de locataire	Caractère	20	O			Locataire
ADR	Adresse de locataire	Caractère	20	O			Locataire
DL	Date de location	Date		O	(1)		Location
KILD	Kilométrage de départ	Numérique	10				Location
KILR	Kilométrage de retour	Numérique	10			(2)	Location
DR	Date de retour de location	Date		O		(3)	Location

(1) La valeur par défaut de la date de location est la date du jour.
(2) La valeur de kilométrage de retour reste avec le kilométrage de départ jusqu'à la da te de retour
(3) La date de retour d'une voiture doit être supérieure ou égale à la date de location.

2. Liste des tables :

La liste des tables est déduite à partir de la liste des colonnes. A chaque sujet correspond une table. Aux trois sujets « **Voiture** », « **Locataire** » et «**Location** » vont correspondre trois tables.
Pour les tables, on appliquera les mêmes règles de nommage que celles adoptées pour les colonnes.

Liste des tables		
Nom table	**Description**	**Sujet**
Voiture	Regroupe l'ensemble des voitures de la société	Voiture
Locataire	Regroupe les personnes qui louent des voitures de la société	Locataire
Location	Stocke l'historique des locations de voitures	Location

3. Affectation des colonnes aux tables :

Il s'agit ici d'affecter chacune des colonnes du premier tableau aux trois tables du deuxième tableau en faisant un rapprochement basé sur la colonne « sujet ».
Nous utilisons la représentation textuelle pour décrire la structure des tables.

```
Voiture (Immat, Marque, Type, Couleur, Puiss, Kilom, Age,
Prix_par_jour, Type_carburant)
Locataire(Loc, Locnom, Adr)
Location(Immat, Loc, Dl, kild, Dr, kilr)
```

4. Clés primaires des tables :

Pour chaque table, nous allons déterminer une clé primaire. Celle ci sera sélectionnée parmi les colonnes de la table.

- Table *Voiture* : Il est dit dans le texte qu'une voiture est décrite par *le numéro d'immatriculation comme identifiant*. On peut alors l'utiliser comme clé primaire.
- Table *Locataire* : Il est également mentionné dans le texte que chaque *locataire est identifié par un numéro locataire*. On peut donc l'adopter comme clé primaire.
- Table *Location* : Une location est *identifié par une voiture louée et le locataire* qui a loué cette voiture. On peut donc supposer que la clé primaire de cette table est composée des deux colonnes *IMMAT et LOC*.

Cependant, en examinant cette table, on peut constater que cette représentation n'est valide que dans le cas où on ne garde pas un historique des locations, c'est-à-dire que suite au retour d'une voiture, la ligne correspondante sera supprimée de la table Location. Si nous souhaitons garder cet historique on doit rajouter la date de location DL à la clé primaire de la table Location, nous adoptons cette dernière alternative.
La description textuelle de la structure des tables en tenant compte des clés primaires sera la suivante :

```
Voiture (Immat, Marque, Type, Couleur, Puiss, Kilom, Age,
Prix_par_jour, Type_carburant)
Locataire(Loc, Locnom, Adr)
Location(Immat, Loc, Dl, Kild, Dr, Kilr)
```

5. Liens entre les tables :

D'après l'énoncé, une location est relative à une voiture et un locataire. Ainsi, on déduit les deux liens suivants :

- un lien entre la table **Location** et la table **Voiture**
- un lien entre la table **Location** et la table **Locataire**

Nous utilisons le tableau suivant pour décrire ces liens :

Table mère	Table fille	Clé primaire	Clé étrangère
Voiture	Location	Immat	Immat
Locataire	Location	Loc	Loc

Précisons que la clé primaire figure dans la table mère et la clé étrangère dans la table fille.

En tenant compte de ces liens entre les tables, la description textuelle des tables devient alors :

```
Voiture (Immat, Marque, Type, Couleur, Puiss, Kilom, Age,
Prix_par_jour, Type_carburant)
Locataire(Loc, Locnom, Adr)
Location(Immat#, Loc#, Dl, Kild, Dr, Kilr)
```

II. Partie pratique :

A - Mode assisté :

1. Implémentation de la base :

Voiture : Table

Nom du champ	Type de données	Description
Immat	Texte	Numéro d'Immatriculation
Marque	Texte	Marque de la voiture
Type	Texte	Type de la voiture
Couleur	Texte	Couleur de la voiture
Puiss	Numérique	Puissance de la voiture
Kilom	Numérique	Le kilométrage présenté dans le compteur
Age	Numérique	Age de la voiture
Prix_par_jour	Monétaire	Prix de location par jour
Type_carburant	Texte	Type du carburant de la voiture

La Table voiture

Locataire : Table

	Nom du champ	Type de données	Description
🔑	Loc	Texte	Numéro du locataire
	Locnom	Texte	Nom du locataire
▶	Adr	Texte	Adresse du locataire

La table Locataire

Location : Table

	Nom du champ	Type de données	Description
🔑	Immat	Texte	Numéro d'immatriculation
🔑	Loc	Texte	Numéro du locataire
🔑	Dl	Date/Heure	Date de location
	Kild	Numérique	Kilométrage de départ
	Dr	Date/Heure	Date de retour
▶	Kilr	Numérique	Kilométrage de retour

La table Location

2. Saisie des données :

Saisir, à votre choix, quelques informations dans les tables.

Voiture : Table

Immat	Marque	Type	Couleur	Puiss	Kilom	Age	Prix_par_jour	Type_carburant
				0	0	0	0	

Enr : 1 sur 1

Locataire : Table

Loc	Locnom	Adr

Enr : 1 sur 1

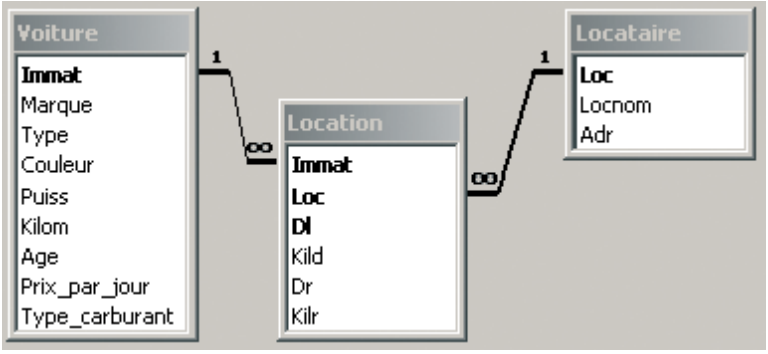
Location : Table

Immat	Loc	Dl	Kild	Dr	Kilr
			0		0

Enr : 1 sur 1

3. Représentation graphique :

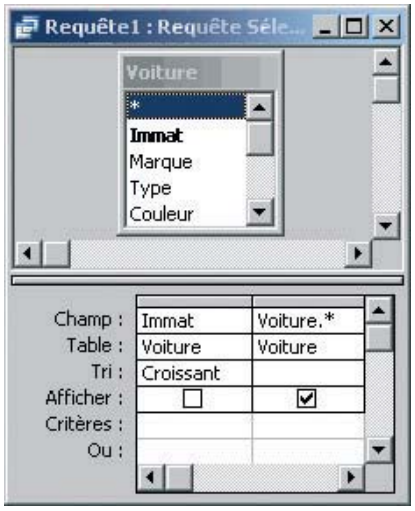
On obtient la représentation graphique suivante de la structure de la base de données :

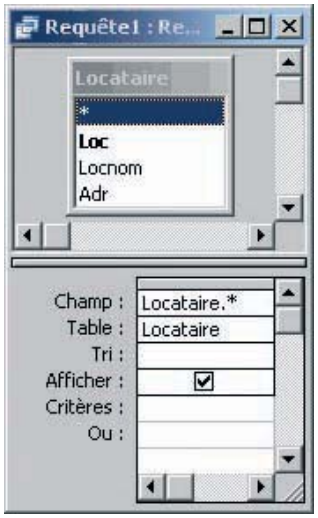


4. Requêtes :

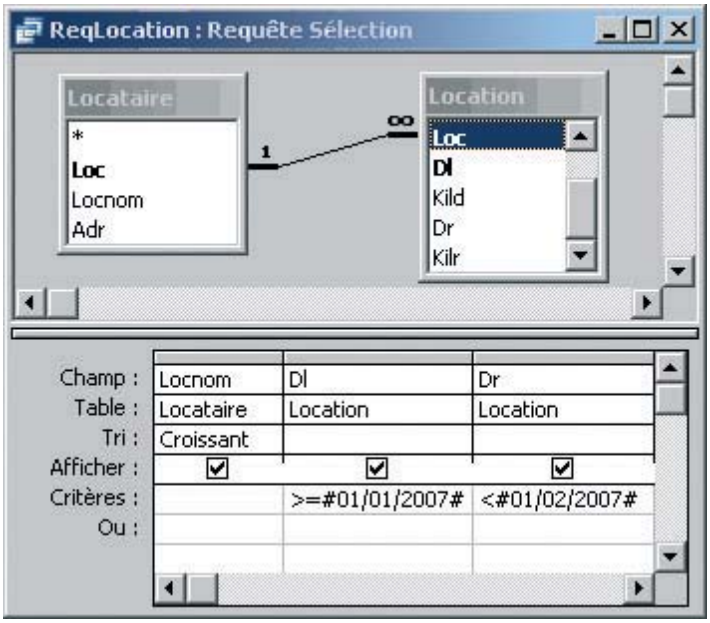
Requête 1 : Affichage de la liste des voitures triée par numéro d'immatriculation.

Requête 2 : Affichage de la liste des Locataires.





Requête3 : Définir une requête qui permet d'afficher les noms des locataires, par ordre alphabétique, qui ont loué des voitures pendant le mois de janvier de l'année en cours.



5. Exemples de formulaires :

B - Mode commande :

En utilisant le langage SQL, disponible et en se basant sur la syntaxe vue dans les chapitres 5 et 6, créer les tables précédentes et répondre aux questions.

Sans tri	Avec Tri Croissant
Requête 1 : affichage de la liste des voitures **SELECT *** **FROM voiture**	**Requête 1** : affichage de la liste des **SELECT *** **FROM voiture** **ORDER BY Immat ;**

Requête 2 : affichage de la liste des Locataires

SELECT *
FROM locataire;

Requête 3 : Affichage des noms des locataires, par ordre alphabétique, qui ont loué des voitures pendant le mois de janvier de l'année en cours.

Utiliser la commande SELECT de la manière suivante :

SELECT Distinct LOCNOM
FROM voiture X, Locataire Y, Location Z
WHERE (DL **Between** "01/01/2008" **AND** "31/01/2008")
AND (X.Immat=Z.Immat **AND** Y.Loc=Z.Loc)
ORDER BY LOCNOM;

Amélioration de la solution

Sachant qu'une voiture ne peut être louée qui si elle est disponible : non louée, en état de circulation ou non réservée.

Problématique : déterminer d'une manière automatique la liste des voitures disponibles pour la location.

Eléments de réponse :

- ajouter une colonne **Etat** dans la table voiture qui renseigne sur l'état de la voiture (**D** pour disponible, **L** pour louée, **R** pour réservée et **P** pour en panne)
- en mode commande,

Utiliser la commande ALTER TABLE de la manière suivante :

ALTER TABLE voiture **ADD COLUMN** (Etat VARCHAR(1)) ;

Utiliser la commande SELECT de la manière suivante :

SELECT *
FROM voiture
WHERE Etat='D';

Application 2

Gestion d'un club vidéo

Énoncé

On se propose de concevoir une base de données relative à la gestion d'un club vidéo.
Dans ce club, un client a le choix de louer des films disponibles en plusieurs exemplaires.
Un **film** est identifié par un code. Il est caractérisé par un titre, une année de sortie, une durée, un type (comédie, policier, science-fiction, …) et les principaux acteurs.
Un film existe en un ou plusieurs exemplaires. Chaque exemplaire est identifié par un code unique attribué par le club vidéo, une date d'acquisition et le support.
Un **support** est identifié par un code et est caractérisé par un type (comme cassette, CD ou DVD).
Un exemplaire, sur un support donné, ne peut contenir qu'un seul film.
Chaque **acteur** est identifié par un code et est caractérisé par un nom, un prénom et éventuellement une nationalité.
Le club détient, pour chaque **client**, un ensemble d'informations : un identifiant qui est le numéro de sa carte d'identité, un nom, un prénom, une adresse, un numéro de téléphone et une profession.
Lors de la location d'un film, à une date donnée, le prix de cette location est fixé par le personnel du club.

Questions

I. Partie théorique :

1. Etablir la liste des colonnes.
2. Etablir la liste des tables.
3. Affecter les colonnes aux tables et proposer une description textuelle.
4. Préciser les clés primaires des tables.
5. Identifier les liens entre les tables.

II. Partie pratique :

1. En utilisant le SGBD disponible dans votre laboratoire, implémenter cette base.
2. Saisir, à votre choix, quelques informations dans les tables.
3. Représenter la structure de cette base de données sous forme graphique.
4. Réaliser les formulaires de saisie et les états que vous jugez nécessaires ainsi qu'un menu principal pour toute l'application.

Eléments de correction

I. Partie théorique :

1. Liste des colonnes :

La liste des colonnes est déduite à partir de l'énoncé et ce en mettant en valeur les propriétés. Ces propriétés seront regroupées dans le tableau qui suit.
Dans la première colonne, il est recommandé d'utiliser les mêmes règles de nommage qu'en programmation (éviter les caractères accentués, réduire la taille des noms des colonnes, etc.).

Liste des colonnes							
Nom colonne	Description	Type de données	Taille	Obligatoire	Valeur par défaut	Valeurs autorisées	Sujet
Code_Film	Le code d'un film	Numérique	5	O			Film
Titre_Film	Le titre du film	Caractère	30	O			Film
An_Sortie	L'année de sortie du film	Numérique	4	N			Film
Type_Film	Le type du film	Caractère	20	O			Film
Duree_Film	La durée du film en minutes	Numérique	3	O			Film
Code_Exempl	Le code de l'exemplaire du film	Caractère	10	O			Exemplaire
Date_Acqui	Date d'acquisition de l'exemplaire	Date		O			Exemplaire
Code_Acteur	Le code d'un acteur	Numérique	5	O			Acteur
Nom_Acteur	Le nom de l'acteur	Caractère	30	O			Acteur
Prenom_Acteur	Le prénom de l'acteur	Caractère	30	O			Acteur
Nat_Acteur	La nationalité de l'acteur	Numérique	8	N			Acteur
Code_Support	Le code d'un support	Numérique	5	O			Support
Type_Support	Le type du support (Cassette, CD, DVD…)	Caractère	10	O			Support
CIN_Client	Le numéro de la carte d'identité d'un client	Numérique	8	O			Client
Nom_Client	Le nom du client	Caractère	30	O			Client
Prenom_Client	Le prénom du client	Caractère	30	O			Client
Adresse_Client	L'adresse du client	Caractère	50	O			Client
Tel_Client	Le numéro de téléphone du client	Numérique	8	N			Client
Prof_Client	La profession du client	Caractère	20	N			Client
Date_Location	La date de location du support	Date		O	(1)		Location
Prix_Location	Le prix de location d'un film	Monétaire		O			Location

(1) La valeur par défaut de la date de location est la date du jour.

2. Liste des tables :

La liste des tables est déduite à partir de la liste des colonnes. A chaque sujet correspond une table. A chacun des sujets **« Film »**, **« Exemplaire »**, **« Acteur »**, **« Client »**, **« Support »**, et «Location » correspond une table.

Pour les tables, on appliquera les mêmes règles de nommage que celles adoptées pour les colonnes.

Liste des tables		
Nom table	**Description**	**Sujet**
Film	Regroupe l'ensemble des films du club.	Film
Exemplaire	Regroupe les exemplaires d'un film	Exemplaire
Client	Regroupe les personnes qui louent un support du club.	Client
Acteur	Regroupe les différents acteurs.	Acteur
Support	Regroupe les différents supports.	Support
Location	Stocke l'historique des locations de films.	Location

3. Affectation des colonnes aux tables :

Il s'agit ici d'affecter chacune des colonnes du premier tableau aux tables du deuxième tableau en faisant un rapprochement basé sur la colonne «Sujet».
Nous utilisons la représentation textuelle pour décrire la structure des tables.

```
Film (Code_Film, Titre_Film, Nbr_Copie_Film, An_Sortie,
Type_Film, Duree_Film)
Exemplaire (Code_exempl, Date_Acqui)
Acteur (Code_Acteur, Nom_Acteur, Prenom_Acteur, Nat_Acteur)
Client (CIN_Client, Nom_Client, Prenom_Client,
Adresse_Client, Tel_Client, Prof_Client)
Support (Code_Support, Type_Support)
Location (Date_Location, Prix_Location)
```

4. Clés primaires des tables :

Pour chaque table, nous allons déterminer une clé primaire. Celle ci sera sélectionnée parmi les colonnes de la table.

• Table **Film** : Il est dit dans le texte qu'un film est identifié par un *code*. On peut alors l'utiliser comme clé primaire.

• Table **Exemplaire** : Il est dit dans le texte qu'un exemplaire est identifié par un code. On peut alors l'utiliser comme clé primaire.

• Table **Acteur** : Il est également mentionné dans le texte que *chaque acteur* est identifié par un code. On peut donc l'adopter comme clé primaire.

• Table **Client** : Le numéro de la carte d'identité peut être adopté comme clé primaire.

• Table **Support** : Il est aussi cité dans le texte que *chaque support est identifié par un code*. On peut donc l'adopter comme clé primaire.

• Table **Location** : Une location est identifiée par un exemplaire d'un film loué et le client qui l'a loué. On peut donc supposer que la clé primaire de cette table est composée des deux colonnes *Code_Exempl et Code_Client*.

Cependant, en examinant cette table, on peut constater que cette représentation n'est valide que dans le cas où on ne garde pas un historique des locations, c'est-à-dire que lorsque le client retourne l'exemplaire du film, la ligne correspondante sera supprimée de la table Location. Si nous souhaitons garder cet historique, on doit rajouter la date de location *Date_Location* à la clé primaire de la table Location, nous adoptons cette alternative.

La description textuelle de la structure des tables en tenant compte des clés primaires sera la suivante :

```
Film (Code_Film, Titre_Film, Nbr_Copie_Film, An_Sortie,
Type_Film, Duree_Film)
Exemplaire (Code_exempl, Date_Acqui)
Acteur (Code_Acteur, Nom_Acteur, Prenom_Acteur, Nat_Acteur)
Client (CIN_Client, Nom_Client, Prenom_Client, Adresse_Client,
Tel_Client, Prof_Client)
Support (Code_Support, Type_Support)
Location (Code_Film, CIN_Client, Date_Location, Prix_Location)
```

5. Liens entre les tables :

D'après l'énoncé, on peut déduire les liens suivants :

- un film peut exister en plusieurs exemplaires → un lien entre la table **Film** et la table **Exemplaire**
- un exemplaire est proposé sur un support → un lien entre la table **Exemplaire** et la table **Support**
- un acteur joue dans un ou plusieurs films et un film est joué un ou plusieurs acteurs → nécessité d'une nouvelle table **Acteur** et la table **Acteur** – **Film** liée aux tables **Acteur** et **Film**
- une location est liée à un client et à un exemplaire → un lien entre la table client et la table Location et un second lien entre la table **exemplaire** et la table **Location**

Nous utilisons le tableau suivant pour décrire ces liens :

Table mère	Table fille	Clé primaire	Clé étrangère
Film	Exemplaire	Code_Film	Code_Film
Support	Exemplaire	Code_Support	Code_ Support
Acteur	Acteur_Film	Code_Acteur	Code_Acteur
Film	Acteur_Film	Code_Film	Code_Film
Client	Location	CIN_Client	CIN_Client
Exemplaire	Location	Code_Exempl	Code_Exempl

Précisons que la clé primaire figure dans la table mère et la clé étrangère dans la table fille.

En tenant compte de ces liens entre les tables, la description textuelle des tables devient alors :

```
Film (Code_Film, Titre_Film, Nbr_Copie_Film, An_Sortie,
Type_Film, Duree_Film)
Exemplaire (Code_exempl,Date_Acqui, Code_Film#, Code_Support#)
Acteur (Code_Acteur, Nom_Acteur, Prenom_Acteur, Nat_Acteur)
Client (CIN_Client, Nom_Client, Prenom_Client, Adresse_Client,
Tel_Client, Prof_Client)
Support (Code_Support, Type_Support)
Location(Code_Exempl#,CIN_Client#,Date_Location,
Prix_Location)
Acteur_Film (Code_Film#, Code_Acteur#)
```

II. Partie pratique (Mode assisté) :

1. Implémentation de la base :

FILM : Table

	Nom du champ	Type de données	Description
🔑	Code_Film	Numérique	Le code d'un film
	Titre_Film	Texte	Le titre du film
	Nbr_Copie_Film	Numérique	Le nombre de copies du film
	An_Sortie	Numérique	L'année de sortie du film
	Type_Film	Texte	Le type du film (Comédie, Policier, ...)
	Duree_Film	Numérique	La durée du film en minutes

La table Film

EXEMPLAIRE : Table

	Nom du champ	Type de données	Description
🔑	Code_Exempl	Numérique	Le code de l'exemplaire du film
	Date_Acqui	Date/Heure	Date d'acquisition de l'exemplaire
	Code_Film	Numérique	Le code d'un film
	Code_Support	Numérique	Le code d'un support

La table Exemplaire

ACTEUR : Table

	Nom du champ	Type de données	Description
🔑	Code_Acteur	Numérique	Le code d'un acteur
	Nom_Acteur	Texte	Le nom de l'acteur
	Prenom_Acteur	Texte	Le prénom de l'acteur
	Nat_Acteur	Numérique	La nationalité de l'acteur

La table Acteur

CLIENT : Table

	Nom du champ	Type de données	Description
🔑	CIN_Client	Numérique	Le numéro de la carte d'identité d'un client
	Nom_Client	Texte	Le nom du client
	Prenom_Client	Texte	Le prénom du client
	Adresse_Client	Texte	L'adresse du client
	Tel_Client	Numérique	Le numéro de téléphone du client
	Prof_Client	Texte	La profession du client

La table Client

SUPPORT : Table

	Nom du champ	Type de données	Description
🔑	Code_Support	Numérique	Le code d'un support
	Type_Support	Texte	Le type du support (Cassette, CD, DVD)

La table Support

ACTEUR_FILM : Table

	Nom du champ	Type de données	Description
🔑	Code_Film	Numérique	Le code d'un film
🔑	Code_Acteur	Numérique	Le code d'un acteur

La table Acteur_Film

LOCATION : Table

	Nom du champ	Type de données	Description
🔑	CIN_Client	Numérique	Le numéro de la carte d'identité d'un client
🔑	Code_Exempl	Numérique	Le code d'un exemplaire
🔑	Date_Location	Date/Heure	La date de location d'un film
	Prix_Location	Monétaire	Le prix de location d'un film

La table Location

2. Saisie des données :

Saisir, à votre choix, quelques informations dans les tables.

Exemple d'informations à saisir dans la table Film

FILM : Table

Code_Film	Titre_Film	Nbr_Copie_Film	An_Sortie	Type_Film	Duree_Film
103	The fixer	3	1999	science fiction	120
174	Hellboy	0	2004	policier	100
237	Assagha	6	1996	romince	130
315	Wilder	2	1997	policier	125
554	Pavement	4	1999	policier	140
560	The crow	4	2000	science fiction	110
563	Domino	4	2000	comedie	128

Enr : 1 sur 11

Exemple d'informations à saisir dans la table Support

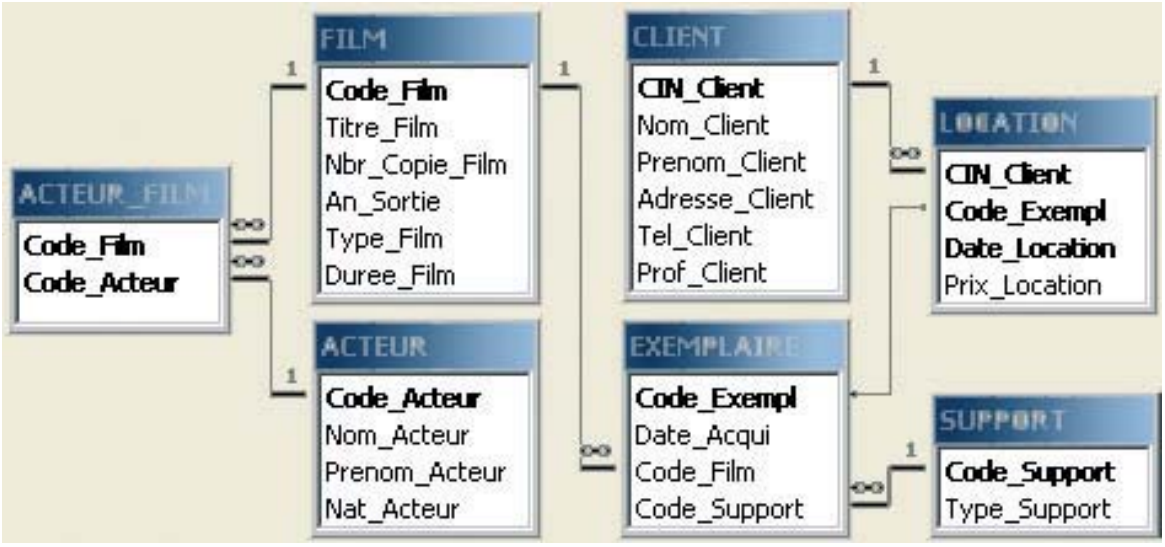
SUPPORT : Table

Code_Support	Type_Support
1	CD
2	CASSETTE
12	CD
45	CASSETTE
78	CD
122	CASSETTE
123	CD
455	DVD
2028	DVD

Enr : 10 su

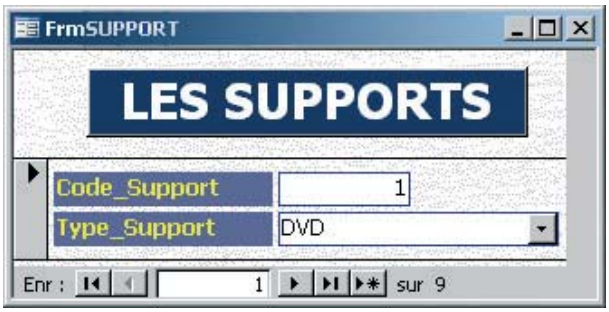
3. Représentation graphique :

On obtient la représentation graphique suivante de la structure de la base de données :

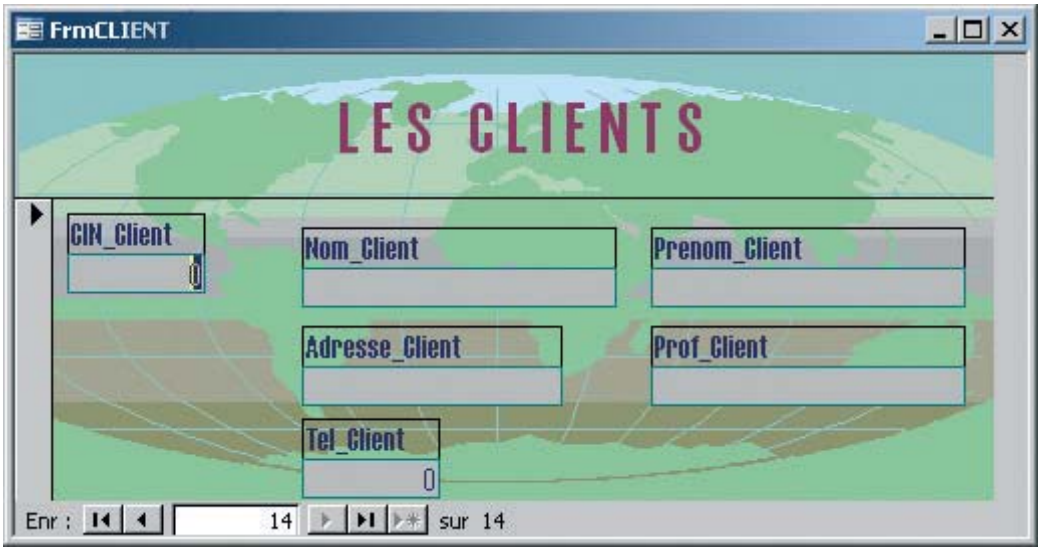


4. Exemples de formulaires :

Le formulaire Films



formulaire Supports



formulaire Supports

Application 3

Gestion d'un Établissement scolaire

Énoncé

Pour ce projet, nous allons aborder la gestion d'un établissement scolaire de type collège, l'objectif étant de stocker les notes des élèves et d'établir les bulletins trimestriels. Au travers de cette application, un certain nombre d'aspects techniques seront traités, de la mise en place d'une base de données (création des tables) jusqu'à la construction d'une interface utilisateur cohérente (formulaires, états, menus).

Un aperçu du projet final

1. Structure de la base

La mise en place d'une base de données consiste d'abord en une analyse des besoins. La partie la plus importante de votre base de données se joue avant même d'allumer l'ordinateur ! L'analyse doit dégager les tables, leurs colonnes et les liens entre elles.

Pour cette application, nous adopterons la structure suivante :

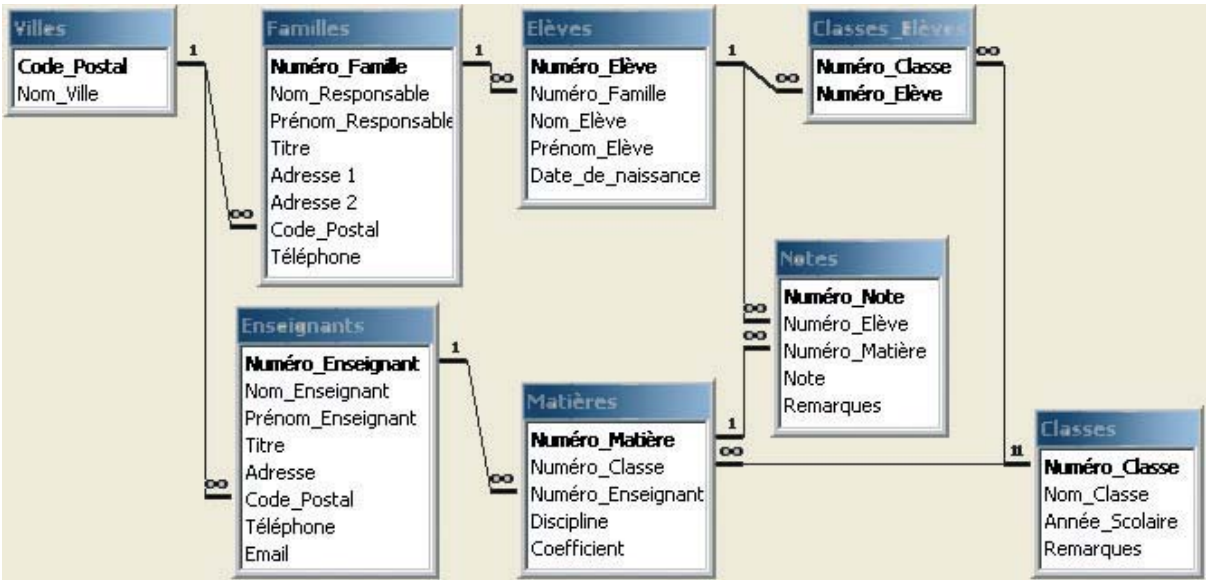
```
Villes (Code_Postal, Nom_Ville)
Familles (Numéro_Famille, Nom_Responsable, Prénom_Responsable,
Titre, Adresse1, Adresse2, Code_Postal#, Téléphone)
Elèves (Numéro_Elève, Numéro_Famille#, Nom_Elève,
Prénom_Elève, Date_de_naissance)
Enseignants (Numéro_Enseignant, Nom_Enseignant,
Prénom_Enseignant, Titre, Adresse, Code_Postal#, Téléphone,
Email)
Classes (Numéro_Classe, Nom_Classe, Année_Scolaire, Remarques)
Classes_Elèves (Numéro_Classe#, Numéro_Elève#)
Matières (Numéro_Matière, Numéro_Classe#, Numéro_Enseignant#,
Discipline, Coefficient)
Notes (Numéro_Note, Numéro_Elève#, Numéro_Matière#, Note,
Remarques)
```

Travail à faire

- Analysez la représentation ci-dessus.
- A l'aide du SGBD disponible dans votre laboratoire, créez la base susmentionnée en tenant compte des remarques suivantes :
  - Réalisez une liste de choix pour la colonne Titre (Monsieur, Madame, Mademoiselle) des tables « **Familles** » et « **Enseignants** » afin de faciliter la saisie.
  - Réalisez une liste de choix pour la colonne *Code_Postal* des tables «**Familles**» et «**Enseignants**» construite à partir de la table « **Villes** »

2. Les relations du projet dans les grandes lignes

- Une famille peut comprendre plusieurs enfants, donc plusieurs élèves. Un élève suit plusieurs matières. Un enseignant peut enseigner plusieurs matières …
- Il est important de définir ce qu'est une classe : dans notre exemple, il s'agit d'un ensemble d'élèves sur une période scolaire déterminée. En d'autres termes, la Quatrième SI 1/Année 2007 est une classe, la Quatrième SI 1/Année 2008 en est une autre.
- Pour des raisons de simplification, nous avons également attribué une clé primaire *Numéro_Matière dans la table* ***Matières***. Une clé idéale aurait été la combinaison *Numéro_Elève, Numéro_Classe* et *Discipline*.



Travail à faire

- Créez les tables constituant cette base de données.

3. Mise en place de l'interface

Pour tester votre projet, vous pouvez insérer quelques lignes dans les différentes tables. Il est cependant plus pratique d'effectuer la saisie au travers de formulaires

les utilisateurs y gagneront en confort, tandis que vous-même pourrez à terme effectuer plus de contrôles de saisies et plus de traitements. En résumé, le formulaire est l'objet dédié à la saisie et à la consultation à l'écran.

3.1. Construction du formulaire Villes

La saisie des villes pourrait se dérouler en même temps que la saisie des familles ou des enseignants.

Cependant, pour centraliser la saisie (et les vérifications ultérieures), nous allons construire un formulaire uniquement consacré aux villes.

LE FORMULAIRE VILLES

Travail à faire

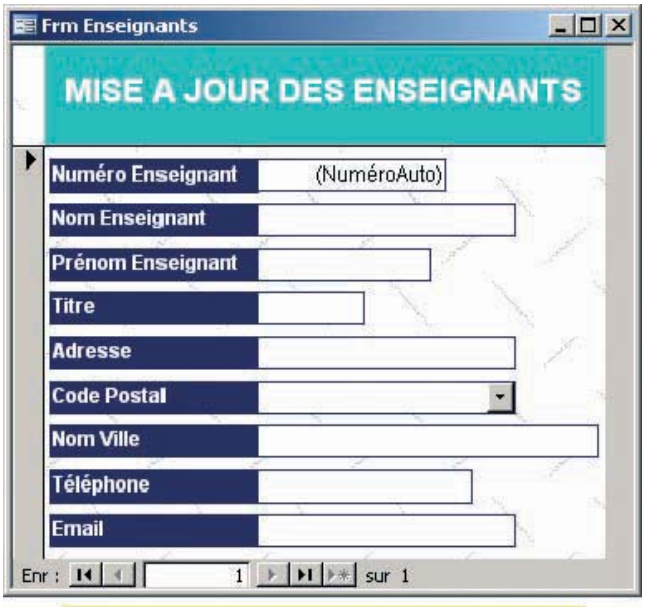
Décrivez puis appliquez la démarche à suivre pour construire un formulaire pour les villes.

Essayez ensuite d'améliorer son aspect graphique.

3.2. Construction du formulaire Enseignants

Globalement, les techniques utilisées pour le formulaire *Enseignants* sont proches de ce qui a déjà été abordé, à quelques détails près :

- Le formulaire *Enseignants* comprend un peu plus de champs, il sera donc plus fonctionnel en mode *Colonne* simple qu'on mode *Tabulaire*.
- Le code postal suffira pour afficher automatiquement le nom de la ville.



LE FORMULAIRE ENSEIGNANTS

Travail à faire

- Décrivez puis appliquez la démarche à suivre pour construire le formulaire *Enseignants.*

3.3. Construction du formulaire Familles et de son sous-formulaire

Le formulaire Familles apporte quelques notions techniques intéressantes :

- Comme le formulaire *Enseignants*, il est basé de préférence sur les deux tables *Familles et Villes*, de façon à gérer de façon automatique l'affichage du code postal en fonction de la ville.
- Pour améliorer encore la saisie, il serait pratique de pouvoir renseigner les enfants d'une famille sur ce formulaire. Autant la relation Familles/Villes implique dans ce sens une relation Un à Un (une famille est rattachée à une seule ville), autant la relation Familles/Enfants implique une relation de Un à Plusieurs (une famille peut comprendre plusieurs enfants). Ceci va nous imposer de réaliser un formulaire avec sous-formulaire.

Travail à faire

- Décrivez puis appliquez la démarche à suivre pour construire le formulaire Familles.

LE FORMULAIRE FAMILLES

3.4. Construction du formulaire Classes

Cette fois, nous allons encore créer un formulaire synthétique pour une saisie optimale. Dans notre schéma, une classe regroupe un ensemble d'élèves et est associée à plusieurs matières, ce qui implique un formulaire avec deux sous-formulaires.

Travail à faire

Décrivez puis appliquez la démarche à suivre pour construire le formulaire *Classes* avec un premier sous-formulaire pour les matières et un deuxième pour les élèves.

LE FORMULAIRE CLASSES

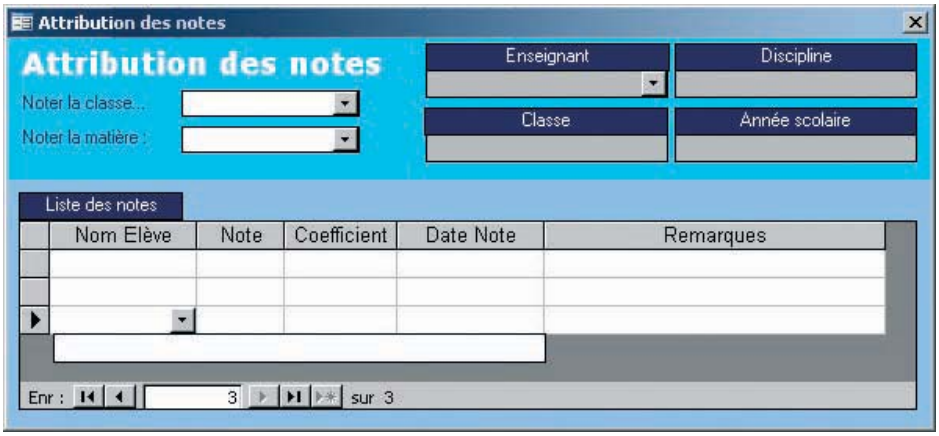
3.5. Construction du formulaire Notes

Le formulaire *Notes* va bien sûr nous permettre de renseigner les notes des élèves, ce qui consiste à associer une matière, un élève et une note.

Globalement, il suffirait de faire un formulaire simple à partir de la table Notes pour que la saisie soit fonctionnelle. Cependant, pour des raisons d'ergonomie et de cohérence, nous allons procéder autrement : a priori, les *notes* s'attribuent en fonction d'une classe et d'une matière, puis élève par élève.

Travail à faire

Décrivez puis appliquez la démarche à suivre pour construire le formulaire Notes.



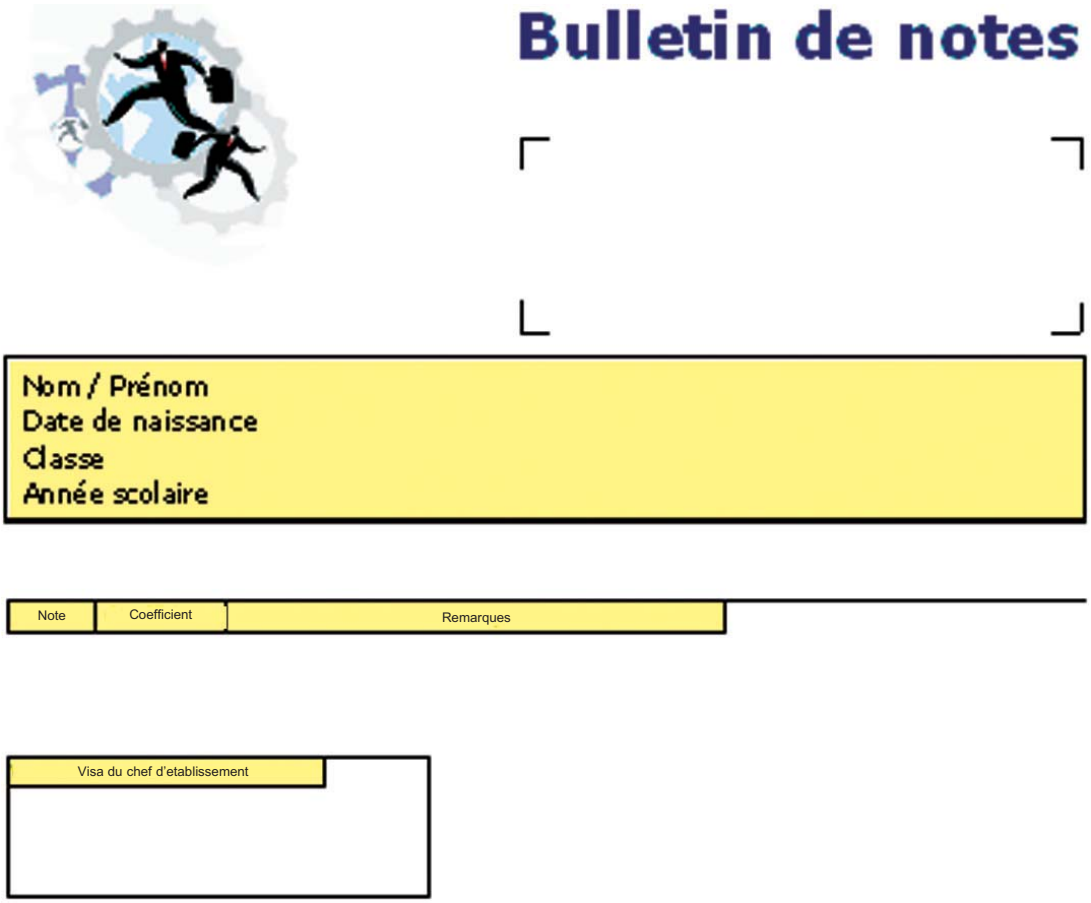
Le formulaire Notes

3.6. Construction du bulletin de notes

Maintenant que quasiment tous les éléments de l'interface sont en place, il s'agit de produire quelques résultats papier, donc de créer quelques états. Les tables, requêtes et formulaires peuvent être imprimés, comme l'attestent la commande **Fichier/Imprimer** et l'icône *Imprimer* qui accompagnent tous ces objets. Cependant, pour une maîtrise optimale de la mise en page et de divers réglages (sauts de pages entre les sections, options de regroupement), il est fortement conseillé de passer par un état pour tout document imprimé de qualité.

Travail à faire

- Préparez l'état Bulletins de notes.
- Ajoutez un en-tête au bulletin de notes.
- Ajoutez la moyenne par matière.
- Ajouter la période sur l'état.



Un exemple de bulletin de notes

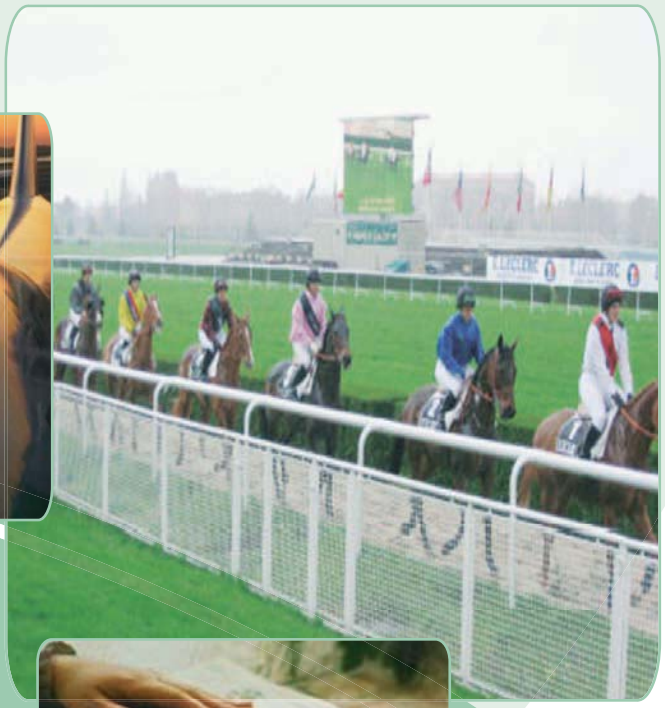
3.7. Finalisation de l'application

Tous vos éléments fonctionnels sont créés. Reste à finaliser votre application pour donner encore un peu plus d'ergonomie au projet.

Travail à faire

- Créez un menu général comme celui présenté au début du projet.
- Réglez les options de démarrage afin de définir le comportement de votre base de données à l'ouverture.

Projets



Objectifs :

En se basant sur ses acquis et sur l'ensemble des connaissances assimilées durant toute l'année scolaire, l'élève doit pouvoir réaliser un certain nombre de projets inspirés de sa vie courante.

Plan :

1.1. Enoncé

Un parieur assidu des champs de courses et des bases de données, voudrait mémoriser dans une base de données les courses de chevaux, les paris qu'il a faits et les résultats.

Plus précisément, il veut enregistrer les informations suivantes pour chaque course :

- Le nom et la date (exemple: Prix de la Tunisie, 04-09-2000)
- Le numéro, le nom et la cote des chevaux partants (exemple: "1, Salicorne, 20/1", "2, Solstice, 8/1", "3, Samovar, 17/1", etc.)
- Ses paris, avec pour chacun: le type de pari (couplé, tiercé, quarté, quinté, etc.), la somme jouée et les numéros de chevaux dans l'ordre du pari (exemple: "tiercé, 20 Dt, 13-2-8")

Une fois la course jouée, on enregistre aussi :

- Le résultat : l'ordre d'arrivée des chevaux (exemple: <1er,13>, <2ème,8>, <3ème,14>, etc.) et les rapports pour chacun des types de pari (exemple: <tiercé dans l'ordre, 900 Dt>, <tiercé dans le désordre: 112 Dt>, <quarté dans l'ordre: 5430 Dt>, <quarté dans le désordre: 750 Dt>, <2 sur 4 : 21 Dt>, etc.)
- Le gain total du parieur pour la course.

Les noms de chevaux sont uniques, et les noms de courses sont uniques.

1.2. Questions

1.2.1. Partie théorique :

1. Etablir la liste des colonnes.
2. Etablir la liste des tables.
3. Affecter les colonnes aux tables et proposer une description textuelle des tables.
4. Préciser les clés primaires des tables.
5. Identifier les liens entre les tables.

1.2.2. Partie pratique :

1. En utilisant le SGBD disponible dans votre laboratoire, implémenter cette base.
2. Saisir, à votre choix, quelques informations dans les tables.
3. Représenter la structure de cette base de données sous forme graphique.
4. Réaliser les formulaires de saisie et les états que vous jugez nécessaires ainsi qu'un menu principal pour toute l'application.

2.1 Enoncé

Pour les besoins de la gestion d'un aéroport, on souhaite mémoriser dans une base de données les informations nécessaires à la description des faits suivants:

- Chaque avion géré est identifié par un numéro d'immatriculation. Il est la propriété soit d'une société, soit d'un particulier: dans les deux cas on doit connaître le nom, l'adresse et le numéro de téléphone du propriétaire, ainsi que la date d'achat de l'avion;
- Chaque avion est d'un certain type, celui-ci étant caractérisé par son nom, le nom du constructeur, la puissance du moteur, le nombre de places;
- La maintenance des avions est assurée par les mécaniciens de l'aéroport. Par sécurité, les interventions sont toujours effectuées par deux mécaniciens (l'un répare, l'autre vérifie). Un même mécanicien peut, selon les interventions, effectuer la réparation ou la vérification. Pour toute intervention effectuée, on conserve l'objet de l'intervention, la date et la durée;
- Pour chaque mécanicien on connaît son nom, son adresse, son numéro de téléphone et les types d'avion sur lesquels il est habilité à intervenir;
- Un certain nombre de pilotes sont enregistrés auprès de l'aéroport. Pour chaque pilote on connaît son nom, son adresse, son numéro de téléphone, son numéro de brevet de pilote et les types d'avion qu'il est habilité à piloter avec le nombre total de vols qu'il a effectué sur chacun de ces types.

2.2 Questions

2.2.1. Partie théorique :

1. Etablir la liste des colonnes.
2. Etablir la liste des tables.
3. Affecter les colonnes aux tables et proposer une description textuelle des tables.
4. Préciser les clés primaires des tables.
5. Identifier les liens entre les tables.

2.2.2. Partie pratique :

1. En utilisant le SGBD disponible dans votre laboratoire, implémenter cette base.
2. Saisir, à votre choix, quelques informations dans les tables.
3. Représenter la structure de cette base de données sous forme graphique.
4. Réalisez les requêtes suivantes:
   - Liste des avions de la société "Voltige";
   - Liste des avions qui sont la propriété de particuliers;
   - Durée totale des interventions faites par le mécanicien « HADDED » au mois de janvier;
   - Liste des types d'avion de plus de 4 places;
   - Liste des pilotes habilités pour le type d'avion «B727 »;
   - Liste des interventions (objet, date) faites sur l'avion numéro « 3242XZY78K3 ».
5. Réaliser les formulaires de saisie et les états que vous jugez nécessaires ainsi qu'un menu principal pour toute l'application.

3.1. Enoncé

Les organisateurs d'un colloque annuel d'informatique veulent monter une base de données pour gérer les inscriptions des participants, la préparation des actes qui contiennent le texte des articles qui sont présentés au colloque, le choix de ces articles, et l'organisation des différentes sessions. Le colloque dure quatre jours et chaque demi-journée est consacrée à une session qui regroupe des articles portant sur le même thème (systèmes temps réel, multimédia, bases de données, ...).

Les articles présentés au colloque et imprimés dans les actes sont choisis de la façon suivante: ce sont des articles de 15 à 20 pages, présentant des résultats de recherche, proposés par une (ou plusieurs) personne, appelée ici auteur, qui travaille dans un laboratoire de recherche d'une université ou entreprise. Un comité de lecture regroupant une trentaine d'experts fait la sélection. Chaque article est évalué par trois experts qui attribuent chacun une note. Les experts ne doivent pas proposer eux-mêmes d'articles ni être de la même université ou entreprise que les auteurs des articles qu'ils évaluent. A partir des notes, le comité classe les articles, choisit les meilleurs et les affecte aux différentes sessions.

Les organisateurs veulent conserver les informations suivantes pour la préparation du prochain colloque:

- pour chaque article proposé: titre, nombre de pages, mots clés, auteur(s) avec mention de l'auteur principal à qui envoyer la réponse (acceptation ou refus), les trois experts avec les notes qu'ils ont mises à l'article. Si l'article est accepté, la session et l'heure à laquelle il sera présenté. S'il y a plusieurs auteurs, celui qui le présentera (appelé l'orateur).
- pour chaque auteur: nom, titre, université ou entreprise, adresse, le(s) article(s) qu'il propose. S'il est auteur principal, on enregistre en plus ses numéros de téléphone et de télécopie, et son adresse électronique. S'il est orateur, on enregistre en plus son CV résumé sur cinq lignes pour que le président de la session puisse le présenter.
- pour chaque expert: nom, titre, université ou entreprise, adresse, numéro de téléphone, numéro de télécopie, adresse électronique, les articles qu'il évalue avec la note qu'il leur attribue.
- pour chaque session: thème, jour, heure de début, heure de fin, le président (celui qui anime la session, présente les orateurs, lance la discussion,....; c'est un expert, un participant ou un auteur d'un autre article), liste des articles de la session, avec leur heure de passage, coût de l'inscription à la session (toutes les sessions n'ont pas le même prix).

- pour chaque participant: nom, affiliation (nom de l'entreprise, université, ...), adresse, s'il a déjà participé à ce colloque une (des) année précédente: quelles années et s'il y était simple participant ou auteur ou expert. On enregistre aussi les sessions auxquelles il s'inscrit et s'il a réglé les frais de son inscription.

3.2. Questions

3.2.1. Partie théorique :

1. Etablir la liste des colonnes.
2. Etablir la liste des tables.
3. Affecter les colonnes aux tables et proposer une description textuelle des tables.
4. Préciser les clés primaires des tables.
5. Identifier les liens entre les tables.

3.2.2. Partie pratique :

1. En utilisant le SGBD disponible dans votre laboratoire, implémenter cette base.
2. Saisir, à votre choix, quelques informations dans les tables.
3. Représenter la structure de cette base de données sous forme graphique.
4. Réalisez les requêtes suivantes:
   - Liste des orateurs de telle session
   - Liste des auteurs principaux dont un article au moins a été accepté
   - Liste des participants à telle session
   - Liste des experts qui n'ont pas encore fait leur évaluation
   - Liste des articles acceptés
   - Liste des articles de note moyenne supérieure à 8
5. Réaliser les formulaires de saisie et les états que vous jugez nécessaires ainsi qu'un menu principal pour toute l'application.

Projet 4 Inventaire d'œuvres d'art

4.1. Enoncé

Les musées d'art veulent constituer une base de données commune des œuvres d'art qu'ils possèdent. Actuellement le conservateur de chaque musée garde, pour chaque œuvre, les informations suivantes: type (peinture, collage, sculpture, lithographie, etc.), titre, année, nom de(s) artiste(s), matière(s), dimensions, le courant artistique (impressionnisme, cubisme, etc.) auquel elle appartient s'il est défini (certaines œuvres sont inclassables) et éventuellement le numéro de l'exemplaire possédé par le musée (certains types d'œuvres comme les lithographies et les sculptures en bronze sont tirées en plusieurs exemplaires, le musée peut alors posséder l'œuvre ou/et l'un, voire plusieurs, des exemplaires de l'œuvre).

En plus, certains conservateurs se sont constitués des fiches techniques décrivant:

- les principaux courants artistiques: nom du courant, période (année de début, année de fin), texte descriptif;
- les artistes: nom, prénom, nationalité, date de naissance, éventuellement date de décès, les courants auxquels il/elle a participé par ses œuvres, texte descriptif.

Ils veulent aussi mettre ces fiches en commun dans la base de données.

4.2. Questions

4.2.1. Partie théorique :

1. Etablir la liste des colonnes.
2. Etablir la liste des tables.
3. Affecter les colonnes aux tables et proposer une description textuelle des tables.
4. Préciser les clés primaires des tables.
5. Identifier les liens entre les tables.

4.2.2. Partie pratique :

1. En utilisant le SGBD disponible dans votre laboratoire, implémenter cette base.
2. Saisir, à votre choix, quelques informations dans les tables.
3. Représenter la structure de cette base de données sous forme graphique.
4. Réalisez les requêtes suivantes:
   - Nom et ville du musée où se trouve telle œuvre de tel(s) artiste(s) ?
   - Liste (titre, année) des œuvres créées par tel artiste.
   - A quels courants a participé tel artiste?
   - Où sont (nom et ville du musée) les œuvres de tel courant artistique ?
   - Liste des titres et des noms de(s) artiste(s) des œuvres d'un musée.
   - Renseignements sur tel artiste (information sur l'artiste et liste de ses œuvres).
   - Renseignements sur tel courant artistique.
   - Liste des musées de telle ville.
5. Réaliser les formulaires de saisie et les états que vous jugez nécessaires ainsi qu'un menu principal pour toute l'application.

5.1. Enoncé

Une compagnie téléphonique souhaite gérer un annuaire des ses abonnés, ainsi que la facturation des appels.

L'annuaire répertorie les personnes, les sociétés et leurs numéros de téléphone. Un même numéro peut être partagé par plusieurs personnes ou sociétés situées à la même adresse. Une même personne ou société peut posséder plusieurs numéros. Les personnes et sociétés sont répertoriées avec leurs noms, adresses, éventuellement un commentaire et leur(s) numéro(s) de téléphone. Les personnes et sociétés possèdent en plus un numéro d'abonné unique permettant à la compagnie de les identifier, même après résiliation ou changement de numéro de téléphone (il n'y a pas deux abonnés avec le même numéro d'abonné). Dans le cas d'une personne, on mémorise aussi ses prénoms, et, dans celui d'une entreprise, sa rubrique professionnelle.

Pour la gestion de la facturation, qui est fonction de l'heure, de la durée et de la distance, on mémorise pour chaque appel le numéro appelé, la date, l'heure et la durée. On mémorise aussi, afin de pouvoir calculer la distance, pour chaque numéro de téléphone l'indicatif de la région correspondant à ce numéro. Dans le cas d'un numéro affecté à plusieurs personnes ou sociétés, une de ces personnes/sociétés est l'abonné principal: c'est à elle que sont envoyées les factures.

5.2. Questions

5.2.1. Partie théorique :

1. Etablir la liste des colonnes.
2. Etablir la liste des tables.
3. Affecter les colonnes aux tables et proposer une description textuelle.
4. Préciser les clés primaires des tables.
5. Identifier les liens entre les tables.

5.2.2. Partie pratique :

1. En utilisant le SGBD disponible dans votre laboratoire, implémenter cette base.
2. Saisir, à votre choix, quelques informations dans les tables.
3. Représenter la structure de cette base de données sous forme graphique.
4. Réaliser les formulaires de saisie et les états que vous jugez nécessaires ainsi qu'un menu principal pour toute l'application.

6.1. Enoncé

L'administration d'Enregistrement des Véhicules désire connaître les informations relatives aux propriétaires et aux transactions (achat/vente) effectuées sur les véhicules.

À chaque véhicule, elle assigne un numéro d'enregistrement. Il n'existe pas deux véhicules ayant le même numéro d'enregistrement.

À tout moment, un véhicule n'appartient qu'à un seul propriétaire, qui est soit un constructeur, soit un garage, ou encore une personne privée. Il peut avoir été possédé par plusieurs propriétaires (à des moments distincts). Un constructeur, un garage ou une personne privée est connu de l'administration d'Enregistrement des Véhicules, c'est à dire considéré comme faisant partie de l'ensemble des propriétaires s'il possède ou a possédé un véhicule.

Qu'il soit constructeur, garage ou personne privée, un propriétaire est caractérisé par un numéro l'identifiant. Pour un constructeur, on connaît son nom, son adresse ainsi que les garages avec lesquels il travaille (garages concessionnaires). Un garage est caractérisé par un nom, une adresse et un numéro de registre de commerce. On connaît le nom, le prénom et l'adresse d'une personne privée.

Pour toute transaction effectuée sur un véhicule, on connaît le vendeur (ancien propriétaire), l'acheteur (nouveau propriétaire), la date de la transaction et le prix d'achat/vente. Un véhicule peut faire l'objet de plusieurs transactions (à des dates différentes). Il n'est pas exclu que deux transactions réalisées à des dates différentes puissent porter sur un même véhicule, un même vendeur et un même acheteur.

Un constructeur ne peut vendre ses véhicules à d'autres constructeurs, ni directement à des personnes privées. Il ne les vend qu'à ses garages concessionnaires. Il n'achète aucun véhicule. Un garage peut vendre ou acheter des véhicules à des personnes privées ou à des garages. Il peut, bien sûr, acheter également des véhicules aux constructeurs pour lesquels il est concessionnaire. Une personne privée ne peut vendre ou acheter des véhicules qu'à des personnes privées ou à des garages. Ceci signifie donc que seuls, les véhicules dont le propriétaire actue l est un constructeur, n'ont été l'objet d'aucune transaction.

6.2. Questions

6.2.1. Partie théorique :

1. Etablir la liste des colonnes.
2. Etablir la liste des tables.
3. Affecter les colonnes aux tables et proposer une description textuelle des tables.
4. Préciser les clés primaires des tables.
5. Identifier les liens entre les tables.

6.2.2. Partie pratique :

1. En utilisant le SGBD disponible dans votre laboratoire, implémenter cette base.
2. Saisir, à votre choix, quelques informations dans les tables.
3. Représenter la structure de cette base de données sous forme graphique.
4. Réaliser les formulaires de saisie et les états que vous jugez nécessaires ainsi qu'un menu principal pour toute l'application.

7.1. Enoncé

Soit la base de données d'université suivante :

ETUDIANT (NumEt, NomEt, Adresse)
ENSEIGNANT (NumEns, NomEns, Grade)
UE (NumUE, Titre)
ENSEIGNE (NumUE, NumEns)
INSCRIPTION (NumEt, NumUE, Annee, Note)

La table **Etudiant** donne pour chaque étudiant son numéro d'étudiant, son nom et son adresse. La table **Enseignant** donne pour chaque enseignant, son numéro, son nom et son grade. La table **UE** donne pour chaque unité d'étude (matière) son numéro et son nom. La table **Enseigne** fait le lien entre les UE et leurs enseignants. La table **Inscription** fait le lien entre les étudiants, les UE qu'ils suivent et la ou les années pendant lesquelles ils ont suivis cette UE et la note qu'ils ont obtenus.

7.2. Questions

7.2.1. Partie théorique :

1. Préciser la clé primaire de chaque table.
2. Identifier les liens entre les tables.
3. Donner les commandes en langage SQL permettant de créer ces tables.

7.2.2. Partie pratique :

1. En utilisant le SGBD disponible dans votre laboratoire, implémenter cette base.
2. Saisir, à votre choix, quelques informations dans les tables.
3. Représenter la structure de cette base de données sous forme graphique.
4. Réalisez les requêtes suivantes:
   - Donner la liste des étudiants (Nom, NumEt) par ordre alphabétique de leur nom.
   - Donner le nombre d'étudiants répertoriés dans la base.
   - Donner les numéros des enseignants qui enseignent au moins une UE enseignée par l'enseignant numéro 150.
   - Donner les noms des enseignants qui enseignent au moins une UE enseignée par l'enseignant numéro 150.
   - Donner le numéro des UE qui ne sont pas enseignées par l'enseignant numéro 150.
   - Donner la liste des étudiants qui n'ont pas suivi d'UE en 2005.
   - Donner dans l'ordre décroissant des numéros d'UE, le nom de l'UE et, par ordre alphabétique, le nom des élèves qui ont suivis l'UE en 2007.

- Donner le nombre d'étudiants ayant suivit au moins une UE en 2007.
- Donner, pour chaque étudiant, le nombre d'UE dans lesquelles il est inscrit.
- Donner, pour chaque étudiant son nom et le nombre d'UE dans lesquelles il est inscrit.
- Donner le nom des UE enseignées par plus de trois enseignants.
- Donner pour chaque étudiant (NumEt) ayant suivi l'UE 34 sa moyenne annuelle en 2007 (les UE ont toutes un coefficient 1).
- En supposant que la relation UE soit la suivante : UE (NumUE, Titre, Coef), répondre à la question précédente en tenant compte des coefficients des UE.

5. Réaliser les formulaires de saisie et les états que vous jugez nécessaires ainsi qu'un menu principal pour toute l'application.

8.1. Enoncé

On travaille dans cet exercice sur la base de données FOURNISSEURS définie par la structure suivante :

FOURNISSEUR (Numfou, Nomfou, Status)
PRODUIT (Numpro, Nompro, Couleur, Poids, Villepro)
PROJET (Numproj, Nomproj, Villeproj)
AFFECTATION (Numfou#, Numpro#, Numproj#, Qte)

8.2. Questions

1. Donner les commandes en langage SQL permettant de créer ces tables.
2. Ajouter, à l'aide des commandes SQL, la colonne « **Villefour** » dans la table «**Fournisseur** ».
3. Donner les commandes SQL nécessaires pour insérer quelques lignes dans chaque table.
4. Exprimer les requêtes suivantes en SQL.
   - Donner toutes les affectations dont la quantité est non nulle
   - Donner le nombre de projets livrés par le fournisseur S1.
   - Donner la quantité totale du produit P1 livrée par le fournisseur S1.
   - Pour chaque produit affecté à un projet, donner le numéro de produit, le numéro de projet et la quantité totale correspondante.
5. Donner la commande SQL permettant d'attribuer à tous les utilisateurs tous les droits sur la table « **Fournisseur** ».

9.1. Enoncé

Soit le schéma de la base de données suivant :

PALETTE (Code, Libellé, Nature, Poids)
MAGASIN (NumM, Nom, VilleM)
CENTRE (numC, société, villeC, classe)
LIVRAISON (numM#, numC#, numP, quantité)

numP étant un code

9.2. Questions

Exprimer les requêtes suivantes en SQL.

1. Les villes où l'on trouve des magasins (sans duplication d'information!).
2. Le numéro des centres (de distribution) effectuant une livraison au magasin "1" (dans l'ordre des numéros de centre).
3. Le code des palettes livrées par un centre de Sousse à un magasin de M'Saken.
4. Les couples "NumM, NumC" correspondant chacun à un magasin et un centre localisés dans la même ville.
5. La nature des palettes livrées par le centre "5".
6. Le numéro des centres dont la classe est inférieure à celle du centre "1".
7. La quantité totale de palettes "5" en cours de livraison par le centre "5".
8. Le nom et la classe des centres (de distribution) localisés à Sousse.
9. Le nom et la ville de tous les magasins.
10. Le nom des magasins à qui le centre "5" livre au moins une palette.
11. Le numéro des magasins à qui un centre, d'une ville différente, livre une ou plusieurs palettes.
12. Le numéro des centres livrant à un magasin une quantité de palettes "4" plus grande que la quantité moyenne de ces palettes "4", livrées à ce même magasin.
13. Le code (-barre) des palettes livrées par un centre de Sousse.
14. Les couples de villes tels qu'un centre de la première livre quelque chose à un magasin de la seconde.
15. Le nombre total de magasins à qui le centre "1" livre au moins une palette.
16. Pour chaque groupe de palettes livrées (au même magasin), le code de la palette, le numéro du magasin et la quantité totale correspondante.
17. Le code des palettes livrées par un centre à un magasin de la même ville.
18. Le code des palettes livrées à un magasin de Sahloul.
19. Le code des palettes dont la nature commence par "Tn".

10.1. Enoncé

Soit le schéma de la base de données suivant :

AGENCE (Num_Agence, Nom, Ville, Actif)
CLIENT (Num_Client, Nom, Ville)
COMPTE (Num_Compte, Num_Agence#, Num_Client#, Solde)
EMPRUNT (Num_Emprunt, Num_Agence#, Num_Client#, Montant)

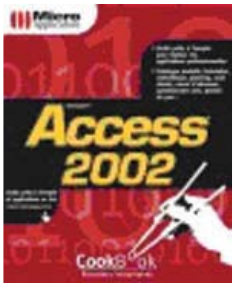
10.2. Questions

1. Créer les tables « **AGENCE** », « **CLIENT** », «**COMPTE**» et « **EMPRUNT»,** avec définition des clés primaires et étrangères.
2. Insérer des n-uplets dans ces tables de manière à pouvoir répondre aux requêtes ci-dessous.
3. Ecrire les requêtes suivantes en SQL :

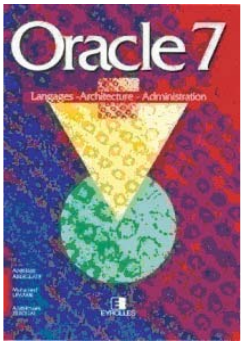
- Liste des agences ayant des comptes-clients
- Liste des clients ayant un compte à l'agence numéro "014".
- Liste des clients ayant un compte ou un emprunt à l'agence numéro "014".
- Liste des clients ayant un compte et un emprunt à l'agence numéro "014".
- Liste des clients ayant un compte et pas d'emprunt à l'agence numéro "014".
- Liste des clients ayant un compte dans une agence où "Adel" a un compte.
- Liste des agences ayant un actif plus élevé que toute agence de "Tunis".
- Liste des clients ayant un compte dans chaque agence de "Sousse".
- Liste des clients ayant un compte dans au-moins une agence de "Sfax".
- Les emprunteurs de l'agence "231" classés par ordre alphabétique.
- Le solde moyen des comptes-clients de chaque agence.
- Le solde moyen des comptes-clients des agences dont le solde moyen est > "10000".
- Le nombre de clients habitant "Kairouan".
- Le nombre de clients de l'agence "017" n'ayant pas leur adresse dans la table « **CLIENT** ».
- Diminuer de "5%" l'emprunt de tous les clients habitant "Nabeul".
- Fermer les comptes de "Amor Tounsi"

BIBLIOGRAPHIE

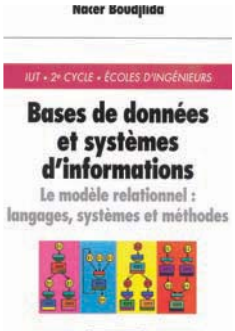
1.Hervé INISAN, ***Microsoft Access 2002***. Mico Application, 1ère édition - Octobre 2001.



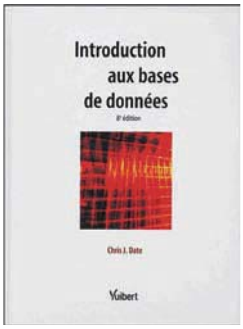
2. Abdelaziz ABDELLATIF, Mohamed LIMAME et Abdelmalek ZEROUAL, ***Oracle 7 : langages, architecture et administration***. Edition Eyrolles.



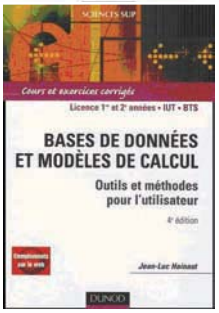
3.Nacer BOUDJLIDA, ***Bases de données et systèmes d'information. Le modèle relationnel : langages, systèmes et méthodes.*** Edition DUNOD.



4. Chris-J DATE, ***Introduction aux bases de données.*** Edition Vuibert.

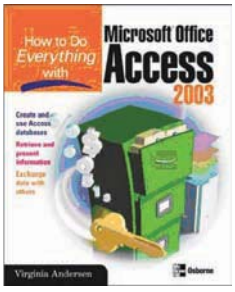


5.Jean-Luc HAINUT, Bases de données et modèles de calcul. Outils et méthodes pour l'utilisateur. Cours et exercices. Edition DUNOD.



6. Collectif, ***L'Essentiel Microsoft Access 2000***. Microsoft Press.

7.Virginia ANDERSEN, ***How to do everything with Microsoft Office Access 2003***. Edition McGraw Hill/Osborne.



WEBOGRAPHIE

1. http://math.hws.edu/vaughn/cpsc/343/2003/history.html
2. http://www.wikipedia.org
3. http://www.supinfo-projects.com/fr/2005/securiser_access_2003/
4.www.lirmm.fr/~ips/enseignement05-06/supports/BaseDeDonnees/Cours/BdCours.pdf

A N N E X E 1

Conventions syntaxiques et typographiques

Afin de faciliter la tâche du lecteur, nous avons utilisé le long de ce manuel des conventions syntaxiques et typographiques suivantes :

MAJUSCULES — Tous les mots clés du langage SQL sont écrits en majuscules.
Exemple : CREATE TABLE, ALTER TABLE, ...

italiques — Les variables de substitution sont représentées en italique. Une variable de substitution est une chaîne de caractères à laquelle sera substituée une valeur d'un type donné (numérique, chaîne de caractères, date, ...).
Exemple : nom_table, nom_colonne, ...

[ ] — Les crochets indiquent une présence optionnelle des éléments qu'ils contiennent.
Exemple : [NOT] NULL

{ } — Les crochets indiquent le choix d'un seul élément parmi la liste contenue à l'intérieur de ces accolades. Les éléments de la liste sont séparés par une barre verticale '|'.
Exemple : {PRIMARY KEY | CHECK}

... — Les trois points indiquent que l'élément qui les précède peut être répété.
Exemple : nom_colonne, ..

A N N E X E 2

Exemple d'éditeur SQL : MySQL

MySQL est un SGBD relationnel faisant partie de la famille des logiciels libres. Cette annexe est réservée à la présentation de quelques possibilités offertes par le SGBD à travers l'exécution des ordres SQL en mode commande notamment :

- La création d'une base de données,
- La création de tables
- L'exécution de commandes de mise à jour de données (INSERT, UPDATE et DELETE) et les commandes de recherche de données (SELECT).

Nous prendrons comme exemple la base de données **Commercial** utilisée essentiellement dans les chapitres 5 et 6.

Création d'une base de données :

Pour créer une base de données on doit suivre les étapes suivantes :

1. Lancer MySQL. La fenêtre d'accueil suivante s'affiche :

2. Dans le champ 'Créer une base de données' saisir le nom de la base à créer («Commercial » par exemple) puis cliquer sur le bouton (Créer). Si l'opération réussit, la fenêtre suivante s'affiche :

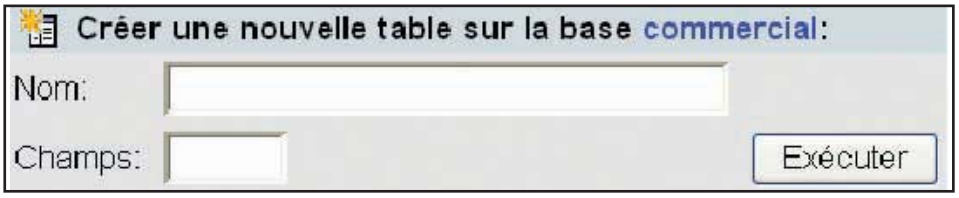
3. Après cette opération, vous pouvez soit quitter MySQL, soit entamer la création de votre première table.

Création de tables :

Pour créer une table on doit suivre les étapes suivantes :

1. Lancer MySQL et sélectionner votre base de données (Commercial)

2. Vous pouvez alors créer les tables en mode assisté ou bien en mode commandes. En mode assisté, il suffit de saisir le nom de la table et le nombre de colonnes (champs) comme l'indique la figure suivante. L'activation du bouton «Exécuter » entraîne l'affichage de l'assistant de création de table.



3. Pour lancer le mode commande, cliquer sur le bouton ( ). La fenêtre de l'éditeur SQL apparaît

4.On peut alors saisir la commande SQL de création de table. La fenêtre suivante illustre la création de la table Client.

5. Cliquer sur « Exécuter ». En cas d'erreurs vous êtres invités à apporter les corrections nécessaires. En cas d'absence d'erreurs, on obtient alors

6. Ce processus peut être itéré pour créer les autres tables. Il est à noter qu'on peut lancer lus qu'une commande à la fois.
   Après la création de toutes les tables, une liste de ces tables créées apparaît sous le nom de la base de données.

Insertion de lignes :

Pour insérer, modifier ou supprimer une ligne dans une table, il suffit de saisir la commande correspondante dans l'éditeur SQL puis exécuter cette commande.
Exemple d'insertion d'une ligne dans la table client :

Remarques

Pour les colonnes de type chaîne de caractères, il faut délimiter les valeurs correspondantes par une simple cote. Lorsque ces valeurs contiennent des caractères spéciaux, il faut délimiter la chaine par des doubles cotes.

Recherche de lignes :

Pour sélectionner des lignes, il faut taper la commande SELECT appropriée à l'aide de l'éditeur SQL puis l'exécuter.