OFPPT

ROYAUME DU MAROC

مكتب التكوين المهني وإنعاش الشغل

Office de la Formation Professionnelle et de la Promotion du Travail

DIRECTION RECHERCHE ET INGENIERIE DE FORMATION

RESUME THEORIQUE

&

GUIDE DE TRAVAUX PRATIQUES

Module : 5

Manipulation d'une base de données

SECTEUR : TERTIAIRE ET NTIC

SPÉCIALITÉ : SYSTÈME ET RÉSEAUX INFORMATIQUES

NIVEAU : TECHNICIEN SPÉCIALISÉ

REMERCIEMENT

La DRIF remercie les personnes qui ont contribué à l'élaboration du présent document.

Pour la supervision :

- MME.BENNANI WAFAE DIRECTRICE CDC TERTIAIRE & TIC
- M. ESSABKI NOURDDINE CHEF DE DIVISION CCFF

Pour la conception :

- ELGHOLABZOURI MOUNIR Formateur à l' ISTA Hay Riad

Pour la validation :

- JELLAL ABDELILAH Formateur animateur au CDC Tertiaire & TIC

Les utilisateurs de ce document sont invités à communiquer à la DRIF toutes les remarques et suggestions afin de les prendre en considération pour l'enrichissement et l'amélioration de ce programme.

Said Slaoui

TABLE DE MATIERE

PARTIE 1 Objectifs

OBJECTIFS OPERATIONNELS DE REMIER NVEAU DE COMPORTEMENT

COMPORTEMENT ATTENDU

Pour démontrer sa compétence, le stagiaire **doit utiliser les techniques de manipulation d'une base de données** à l'aide d'un logiciel de bases de données selon les conditions, les critères et les précisions qui suivent.

CONDITIONS D'EVALUATION

- Travail effectué avec :
    - un micro-ordinateur ;
    - un logiciel de base de données ;
    - une imprimante.
- Travail effectué à partir des source de référence et étude de cas

CRITERES GENERAUX DE PERFORMANCE

- Respect des consignes et du temps alloué.
- Utilisation judicieuse des commandes.
- Interprétation juste des messages apparaissant à l'écran.
- Sauvegarde et restauration appropriées des données.
- Utilisation optimale des fonctions d'aide des logiciels et autres sources de référence.
- Respect des règles de la sécurité.

OBJECTIFS

A. Analyser la demande pour manipuler une Base de données
B. Traiter et manipuler les données
C. Adapter la structure de la base de données à de nouveaux besoins
D. Documenter la base de données

OBJECTIFS OPERATIONNELS DE SECOND NIVEAU

LE STAGIAIRE DOIT MAITRISER LES SAVOIRS, SAVOIRS-FAIRE, SAVOIR-PERCEVOIR OU SAVOIR-ETRE JUGES PREALABLES AUX APPARENTISSAGES DIRECTEMENT REQUIS POUR L'ATTEINTE DE L'OBJECTIF DE PREMIER NIVEAU, TELS QUE :

Avant d'apprendre à analyser la demande pour manipuler une Base de données. (A) :

1. Décrire les conséquences des systèmes de gestion de base de données sur le fonctionnement d'une entreprise.
2. Décrire les caractéristiques des bases de données.
3. Distinguer les types de bases de données.
4. Enumérer les utilisations possibles d'une base de données.
5. Expliquer au stagiaire le principe des bases de données relationnelles..
6. Décrire au stagiaire les phases de création d'une base de données simple
7. Analyser la demande (2 tables avec une relation).

Avant d'apprendre à Traiter et manipuler les données (B) :

1. Enumérer les inconvénients d'une méthode classique de recherche des données.
2. Montrer la souplesse d'utilisation d'un SGBDR pour la recherche des données.

Avant d'apprendre à adapter la structure de la base de données à de nouveaux besoins. (C):

Sensibiliser le stagiaire à l'intérêt de la sécurité des données.

Avant d'apprendre à documenter la base de données (D):

Sensibiliser le stagiaire à la terminologie correctement en anglais et en Français

PARTIE 1 SYSTEME DE GESTION DE BASE DE DONNEE, PRINCIPE ET FONCTIONNEMENT

PARTIE 2 SGBD PRINCIPE ET FONCTIONNEMENT

I Généralités sur les bases de données

Définition et Historique

Une base de données est un ensemble structuré de données enregistrées sur des supports informatisés, pouvant satisfaire simultanément plusieurs utilisateurs de façon sélective, en un délai raisonnable.

Le concept de Base de Données (BDD) est apparu vers 1960, face au nombre croissant d'informations que les entreprises devaient gérer et partager :

Chaque nouvelle application créait alors ses propres fichiers de données et ses propres programmes ;

le concept de base de données va à l'encontre de cette façon de procéder : il permet la centralisation, la coordination, l'intégration et la diffusion de l'information archivée.

La base de données enregistre les faits ou événements qui surviennent dans la vie d'un organisme, pour les restituer à la demande : elle permet également de tirer des conclusions en rapprochant plusieurs faits élémentaires.

Les données peuvent être manipulées par plusieurs utilisateurs ayant des **vues différentes** sur ces données ("points de vue" différents).

La structure d'ensemble des données suit une définition rigoureuse appelée **SCHEMA**.

Facteurs liés au développement des SGBD :

- augmentation des capacités mémoire, et diminution des temps d'accès
- apparition sur le marché d'applications fiables et diversifiées, qui doivent partager leurs données
- développement des systèmes de gestion en temps réel : "Gestion transactionnelle"
- approche globale "orientée données" des problèmes de gestion : les données sont organisées de façon rationnelle plutôt que définies au coup par coup selon les applications à réaliser.

Rappel sur les systèmes de gestion de fichiers

Toute manipulation de fichier exige trois niveaux d'intervention, et trois couches logicielles :

- Gestion du support physique : disques durs, disquette, streamers…

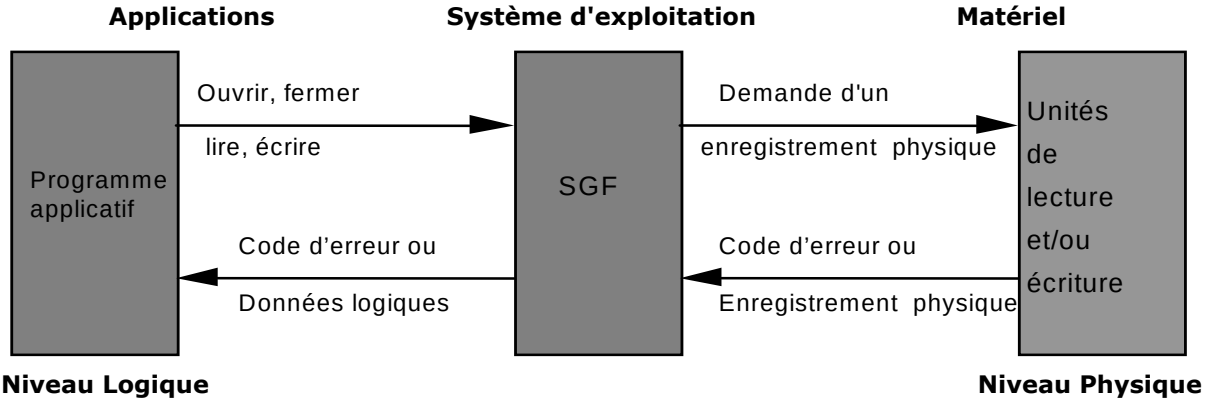
⇨ *Pilote d'entrées-sorties (Driver)*

- Gestion des structures internes des fichiers, et des méthodes d'accès : ouverture, fermeture, lecture, écriture…

⇨ *Système de gestion de fichiers (SGF)*

- Gestion des contenus : calculs, tests, affichages ...

⇨ *Programmes applicatifs*

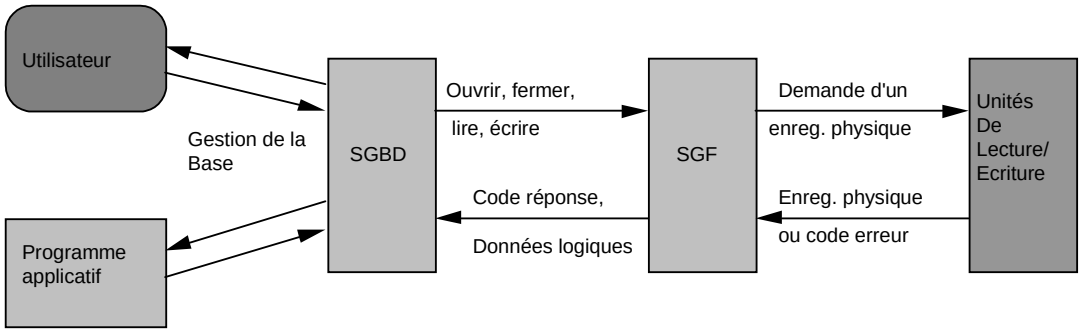


Système de Gestion de Base de données : SGBD

- Ensemble des programmes et des langages de commande qui permettent de :

- définir des "bases de données", et des relations entre les éléments de chaque base ;
- spécifier le traitement de ces données : interrogations, mises à jour, calculs, extractions...

- Le SGBD reçoit des commandes aussi bien des programmes d'application que des utilisateurs : il commande les manipulations de données, généralement par l'intermédiaire d'un SGF.



II Objectifs de l'approche SGBD

- Pour pallier aux inconvénients des méthodes classiques de gestion de fichiers, les SGBD visent quatre objectifs : intégration et corrélation, flexibilité (indépendance), disponibilité, sécurité.
- Ces objectifs exigent une distinction nette entre les données et les procédures de manipulation de ces données : aux données, on associera une fonction **d'administration des données**, aux procédures de manipulation une **fonction de programmation**.

Intégration et corrélation

Dans les systèmes classiques, chaque application gère ses données dans ses propres "fichiers", d'où :

- Un risque de redondance, et un danger d'incohérence des données
- La même donnée peut appartenir à plusieurs applications, induisant une déperdition de stockage.
- Toute modification de cette donnée est à enregistrer plusieurs fois : si cette mise à jour multiple n'est pas effectuée correctement, les données deviennent incohérentes.
- Le coût de la mise à jour augmente du fait de la multiplication des entrées-sorties physiques.

- Une difficulté pour créer de nouveaux traitements
- Les nouvelles applications entraînent des duplications supplémentaires de données.
- Leur intégration avec les applicatifs en exploitation entraîne des modifications importantes.

Dans l'approche SGBD, un "réservoir" commun (**intégration**) est constitué, représentant une modélisation (**corrélation**) aussi fidèle que possible de l'organisation réelle de l'entreprise :

- Toutes les applications puisent dans ce réservoir, les données qui les concernent, évitant ainsi les duplications.
- Mais le partage des données entre les utilisateurs pose le problème de la synchronisation des accès concurrents.

Flexibilité ou indépendance

- Dans les systèmes classiques, tout changement intervenant dans le stockage des données (support, méthode d'accès physique) entraîne des modifications lourdes des applications correspondantes.
- L'approche SGBD poursuit trois objectifs, pour assurer l'indépendance des données par rapport aux traitements :
- indépendance physique: tout changement de support, de méthode d'accès reste transparent au niveau de l'utilisateur.
- indépendance logique : les programmes d'application sont rendus transparents à une modification dans l'organisation logique globale, par la définition de sous-schémas couvrant les besoins spécifiques en données.

- indépendance vis-à-vis des stratégies d'accès : l'utilisateur n'a plus à prendre en charge l'écriture des procédures d'accès aux données. Il n'a donc pas à intégrer les modifications tendant à optimiser les chemins d'accès (ex: création d'index).

Disponibilité

- Le choix d'une approche SGBD ne doit pas se traduire par des temps de traitement plus longs que ceux des systèmes antérieurs.
- L'utilisateur doit ignorer l'existence d'utilisateurs concurrents.
- L'aspect "performance" est donc crucial dans la mise en oeuvre d'une base de données. Un tel objectif ne peut être atteint que si la conception d'une base de données est menée de façon rigoureuse avec un découpage fonctionnel adéquat. Les règles et contraintes inhérentes sont évoquées lors de l'apprentissage d'une méthodologie d'analyse (exemple MERISE).

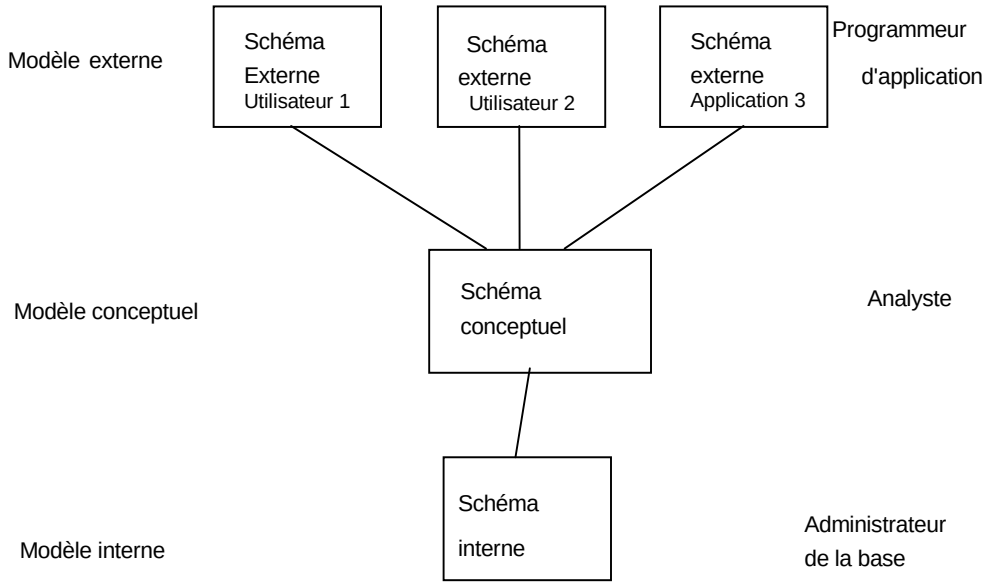
Sécurité

- La sécurité des données recouvre deux aspects :
- **l'intégrité**, ou protection contre l'accès invalide (erreurs ou pannes), et contre l'incohérence des données vis-à-vis des contraintes de l'entreprise.
- **la confidentialité**, ou protection contre l'accès non autorisé ou la modification illégale des données.

- Pour ne pas trop affecter les performances, la sécurité doit également être prise en compte dès la phase de conception.

III Architecture fonctionnelle d'un SGBD : ANSI-SPARC

- Dans le cadre du groupe de normalisation nord américain (ANSI), un groupe d'études a été créé en 69, *Standard Planning and Requirement Committee* (SPARC) avec pour mission, une standardisation des SGBD.
- Les travaux ont abouti en 75 (ANSI 75) par la proposition d'une architecture multi-niveaux : à chaque niveau fonctionnel, sont associés un modèle et un schéma de données, un langage de description de données (LDD) permettant de décrire les données du schéma, et un langage de manipulation de données (LMD) permettant de les utiliser (accès pour consultation, mise à jour...).



Niveau conceptuel

- C'est une abstraction aussi fidèle que possible, de l'univers de l'entreprise, après modélisation et indépendamment de toute référence à l'utilisation et à l'implantation en machine.

- Le modèle conceptuel de données (MCD) permet le passage d'un concret inaccessible (l'univers réel) à un abstrait manipulable : le schéma conceptuel. Celui-ci peut donc être considéré comme la description du contenu de la base : c'est le résultat d'un travail d'analyse et de conception d'un système d'information automatisé.
- Un schéma conceptuel doit offrir les caractéristiques suivantes :
- puissance de représentation : aspects structurels, contraintes existant dans l'univers réel.
- stabilité et flexibilité : l'ajout d'une nouvelle donnée ou d'une nouvelle contrainte ne doit pas entraîner de changement important dans le schéma.
- simplicité de compréhension : nombre d'éléments réduit, dissociation claire des différents concepts.
- simplicité d'utilisation : nombre restreint d'outils ou de primitives de manipulation.
- base formelle : la définition du schéma doit s'appuyer sur une méthode rigoureuse, mathématique, pour éviter toute ambiguïté d'interprétation et pour garantir la fiabilité des données.

- Pour aboutir au schéma conceptuel, l'analyste doit repérer dans le réel, et recenser de manière exhaustive, toutes les entités et toutes les associations :

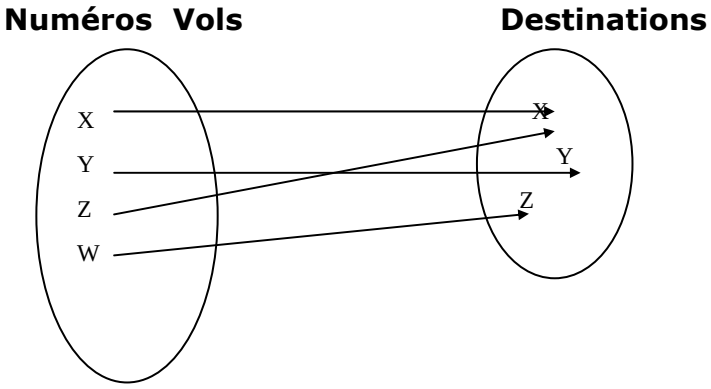
- Une **entité** peut être définie comme une personne, un objet, un lieu, un statut, un événement qui ont une existence dans le monde réel. C'est un objet concret ou abstrait, possédant un certain nombre de caractéristiques spécifiques (exemple : le produit x coûte y francs).
- Généralement, les entités du monde réel se manifestent à travers des faits élémentaires.
- Certains faits faisant intervenir plusieurs entités, il apparaît la notion d'**association**. Une association (ou **lien**) est un ensemble de deux ou plusieurs entités, chacune d'elles jouant un rôle particulier.

Exemple : le fait que la "voiture x" appartienne à la "personne y" est une association entre les entités "voiture " et "personne".

- Selon la notation CODASYL, trois types de liens peuvent être envisagés :

- les liens fonctionnels notés N : 1

On a un lien fonctionnel N:1 de A vers B si toute occurrence de A détermine au plus une occurrence de B, et si à toute occurrence de B, correspond un nombre quelconque d'occurrences de A.
Exemple : dans une compagnie aérienne, connaissant le numéro d'un vol, on en déduit d'une manière unique la destination, mais plusieurs vols peuvent avoir la même destination.



- les liens hiérarchiques notés 1 : N.

On a un lien hiérarchique 1:N de A vers B si une occurrence de A peut déterminer un nombre quelconque d'occurrences de B et si, à une occurrence de B, correspond au plus une occurrence de A.
Exemple : la polygamie est un lien 1 : N de "homme" vers "femme".

- les liens maillés notés N : M.

On a un lien maillé de A vers B s'il n'existe aucune restriction sur le nombre d'occurrences de A et B intervenant dans le lien.

Exemple : dans un lycée donné, un enseignant peut dispenser des cours dans plusieurs matières différentes ; de la même façon, une matière peut être dispensée par plusieurs enseignants.

Niveau externe

- Le niveau externe comprend les "vues" spécifiques définies pour la manipulation des données. Il prend en compte les contraintes d'accès imposées par la nature des applications à considérer (indépendamment des caractéristiques techniques) et exprime les besoins en données des différents utilisateurs, ou applications.

- Le modèle logique des données (MLD) utilisé à ce niveau externe peut différer de celui utilisé au niveau conceptuel. Ainsi, certaines vues peuvent ne pas être construites dans la base, mais déduites par calcul à partir de certaines données du schéma conceptuel (exemple : ancienneté obtenue par différence entre année en cours et annnée d'embauche dans la société).

Niveau interne ou Physique

- Il correspond à la représentation en machine, aussi efficace que possible, du schéma conceptuel : le schéma physique intègre les caractéristiques techniques (choix du SGBD, du matériel, du système d'exploitation…).

- L'efficacité doit tenir compte d'une part des contraintes d'implantation (taille des disques, optimisation du système de fichiers…), d'autre part des critères d'utilisation (traitement interactif ou en batch, selon la fréquence d'utilisation et la durée du traitement…).

IV Fonctionnement d'un SGBD

Chronologie des opérations dans l'interrogation d'un SGDB

Un programme d'application A émet une demande de lecture de données au SGBD sur une des bases :

- Le SGBD traite la demande en consultant le sous-schéma externe relatif au programme d'application A, obtenant ainsi la description des données.
- Le SGBD consulte le schéma conceptuel et détermine le type logique de données à extraire.
- Le système examine la description physique de la base en rapport avec la requête logique et détermine le (ou les) enregistrement(s) physique(s) à lire.
- Le système lance une commande au système d'exploitation pour rechercher physiquement l'enregistrement désiré.
- Le système d'exploitation, par le biais de ses méthodes d'accès, accède à l'enregistrement physique.
- Les données demandées sont transférées dans les buffers, ou mémoires tampons.
- Le SGBD, à partir d'une comparaison entre le schéma logique global (conceptuel) et le sous-schéma externe de l'application A, extrait des données stockées dans le buffer, l'enregistrement logique réclamé par le programme d'application. Il effectue également les transformations éventuelles de format.
- Le SGBD transfère les données des buffers dans la zone de liaison du programme d'application A.
- Le SGBD fournit également des informations "d'état" au programme d'application, lui signalant en particulier les erreurs éventuellement constatées au cours du processus d'extraction.
- Le programme d'application, qui dispose des données et d'informations de "service" en assure la bonne exploitation !
- Les ordres d'écriture dans la base physique sont traités par un processus similaire, toute modification ou adjonction étant en général précédée d'une opération de lecture.
- A signaler que, dans la majorité des cas, le SGBD doit traiter simultanément plusieurs demandes de données en provenance de plusieurs programmes d'application, utilisant plusieurs schémas externes différents.

Les langages d'un SGBD

- Cette présentation des SGBD fait apparaître la nécessité de bien différencier deux étapes :
- la définition des données par l'administrateur de la base (DBA)
- leur utilisation par les utilisateurs ou les programmeurs d'application.
- Le SGBD met donc à disposition deux types de langage : LDD et LMD

Langage de Description de Données : LDD

- Il permet de décrire précisément la structure de la base et le mode de stockage des données. Alors que l'utilisation de fichiers permet seulement une description de données interne au programme, dans une approche Base de Données, on effectue la description de toutes les données une fois pour toutes : elle constitue l'ensemble des **tables et dictionnaires de la base**, son **schéma** (terminologie CODASYL).

- En particulier, il précise la **structure logique des données** (nom, type, contraintes spécifiques...), la **structure physique** (mode d'implantation sur les supports, mode d'accès), la définition des **sous-schémas ou "vues".**

Langage de Manipulation de Données : LMD

L'utilisation d'une BDD suppose un grand nombre d'utilisateurs, souvent non informaticiens, ayant des tâches et des besoins variés auxquels le LMD doit pouvoir répondre. Le SGBD fournit deux niveaux d'accès :

- le langage d'interrogation, ou langage de requête interactif

évite le recours à des langages généraux de programmation. Il doit avoir une syntaxe souple, si possible graphique, être accessible aux non-spécialistes et permettre la formulation de demandes utilisant des critères variés et combinés.

- le langage hôte

pour les traitements réguliers, le SGBD doit fournir une interface permettant l'utilisation de la base à l'aide des langages procéduraux (COBOL, Pascal, C/C++….), en incorporant les requêtes dans des programmes classiques.

Classification des LMD

- langages navigationnels (ex : SYMBAD)

dans les SGBD hiérarchiques ou réseaux. Les requêtes du langage décrivent les chemins d'accès aux différentes données, celles-ci étant généralement chaînées entre elles.

- langages algébriques (ex : SQL voir en détail chap. 4 part. 3)

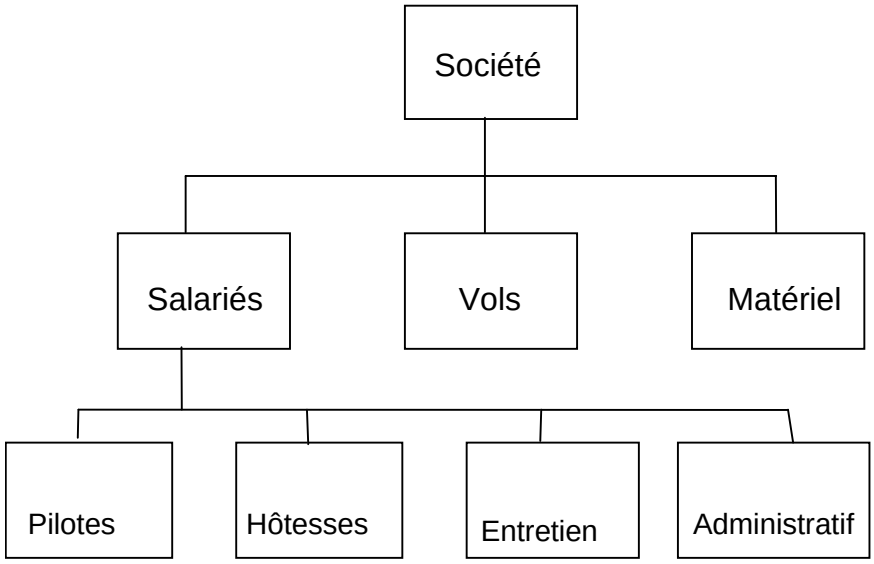
Dans les SGBD relationnels. Ils utilisent, pour fournir des résultats aux requêtes, les opérateurs de l'algèbre relationnelle.

V Principaux modèles logiques

Les trois principaux modèles sont, dans l'ordre chronologique de leur arrivée sur le marché, le modèle hiérarchique, le modèle réseau (ou navigationnel), le modèle relationnel.

Le modèle hiérarchique

Exemple : le Système d'information d'une compagnie aérienne



- L'ancêtre le plus répandu est le SGBD IMS (Information Management System), développé et commercialisé par IBM dans les années 70

- Caractéristiques générales du modèle :

- Forte dépendance entre la description de la structure des données et la manière dont celles-ci sont enregistrées sur le support physique.
- Les éléments de base du modèle sont des enregistrements logiques reliés entre eux pour constituer un arbre ordonné.
- Les entités (ou **segments**) constituent les noeuds, celui de plus haut niveau portant le nom de racine ; les branches (pointeurs logiques entre entités) constituent les **liens**. Chaque segment est une collection d'objets appelés champs (ou **Fields**).
- Chaque segment a obligatoirement un père (sauf la racine), et peut avoir plusieurs fils.

- Avantages :
- rigueur des structures et des chemins d'accès
- simplicité relative de l'implémentation
- adéquation parfaite du modèle à une entreprise à structure arborescente.

- Inconvénients :
- les accès se font uniquement depuis la racine
- la structure interdit les liens N:M, ne permettant que le lien 1:N. La représentation d'autres relations impose de ce fait une redondance de l'information.

Exemple : comment représenter dans ce modèle, un parc de véhicules et un ensemble de chauffeurs, chaque chauffeur pouvant conduire plusieurs véhicules, et un véhicule pouvant être conduit par plusieurs chauffeurs ?

- les "anomalies" que l'on constate lors des opérations de mise à jour (insertion, destruction, modification) : l'élimination d'un noeud entraîne l'élimination de tous les segments de niveau inférieur qui lui sont rattachés (risque de perdre des données uniques)
- indépendance logique très réduite : la structure du schéma doit refléter les besoins des applications.
- pas d'interface utilisateur simple.

Le modèle en réseau

- Evolution du modèle hiérarchique intégrant les résultats du travail du groupe CODASYL (comité de langage de programmation), qui avait démarré l'étude d'une extension de COBOL pour manipuler les bases de données. En 1969, il donne ses premières recommandations concernant syntaxe et sémantique du LDD et du LMD.

- Même si cette vue est un peu simplificatrice, une base en réseau peut être décrite comme un certain nombre de fichiers comportant des références les uns vers les autres. Les entités sont connectées entre elles à l'aide de pointeurs logiques :

- un enregistrement d'un ensemble de données A est associé à une série d'enregistrements (ou records) d'un autre ensemble de données B. On constitue ainsi des SET, ou COSET, structure fondamentale du modèle en réseau
- le lien entre les enregistrements de A et ceux de B est 1:N
- le COSET comporte un type d'enregistrement "propriétaire" (l'enregistrement de A est dit OWNER) et un type d'enregistrement "membre" (les enregistrements de B sont MEMBER).

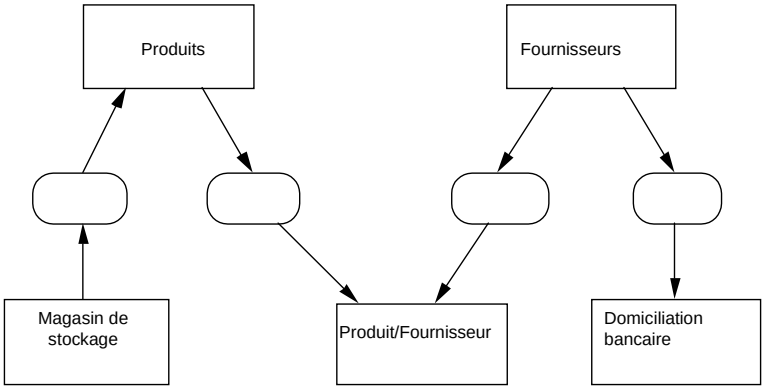
- Avantages et inconvénients du modèle :
- aucune restriction dans la conception : un type de "record" peut à la fois être propriétaire et membre de plusieurs sets
- représentation naturelle des liens maillés N:M
- pas d'anomalies pour les opérations de stockage
- commercialisation importante des systèmes correspondants (DMS, IDMS, TOTAL, IDS II, SOCRATE...),

MAIS

- pas d'indépendance par rapport aux stratégies d'accès
- procéduralité importante des langages de manipulation ; l'utilisateur doit "naviguer" dans le réseau logique constitué par les enregistrements et les chaînes de pointeurs.

- *Exemple : schéma représentant le sous-système d'information*

Produits / magasins de stockages / fournisseurs / domiciliations bancaires



Le modèle relationnel

- C'est un article publié en 1969 par un mathématicien du centre de recherche IBM, **Codd**, qui définit les bases de ce modèle relationnel. Codd s'est intéressé au concept d'information et a cherché à le définir sans se préoccuper de la technique informatique, de ses exigences et de ses contraintes. Il a étudié un modèle de représentation des données qui repose sur la notion mathématique de "relation". Dans la pratique, une relation sera représentée par une **table** de valeurs.

Exemple: représentation d'une table du personnel

Matricule	Nom	poste	Salaire	N° dept
350	Durand	Employé	8000	320
780	Dupond	Cadre	15000	870
320	Veillon	PDG	25000	400
490	Martin	Cadre	15000	320

Définitions

- Une relation est un ensemble de tuples (lignes), dont l'ordre est sans importance. Les colonnes de la table sont appelées attributs ou champs. L'ordre des colonnes est défini lors de la création de la table.
- Une clé est un ensemble ordonné d'attributs qui caractérise un tuple. Une clé primaire le caractérise de manière unique, à l'inverse d'une clé secondaire.
- On dit qu'un attribut A est un **déterminant** si sa connaissance détermine celle de l'attribut B (B dépend fonctionnellement de A).

Caractéristiques du modèle

- Schéma de données facile à utiliser : toutes les valeurs sont des champs de tables à deux dimensions.
- Améliore l'indépendance entre les niveaux logique et physique : pas de pointeurs visibles par l'utilisateur.
- Fournit aux utilisateurs des langages de haut niveau pouvant éventuellement être utilisés par des non-informaticiens (SQL, L4G) et un ensemble d'opérateurs basé sur l'algèbre relationnelle : union, intersection, différence, produit cartésien, projection, sélection, jointure, division.
- Optimise les accès aux bases de données
- Améliore l'intégrité et la confidentialité : unicité de clé, contrainte d'intégrité référentielle
- Prend en compte une variété d'applications, en gestion et en industriel
- Fournir une approche méthodologique dans la construction des schémas.

PARTIE 2 CONCEPTION ET DEMARCHE

PARTIE 3 Conception et démarche

I Conception de bases de données

Les formes normales

Les formes normales permettent de construire un schéma conceptuel correct à partir des relations « brutes » issues des données recueillies auprès des clients.

1ère forme normale

Une relation est dite en première forme normale si chaque attribut possède une seule valeur (ce qui exclut les groupes), et si elle admet une clé primaire.

Exemple:

L'exemple porte sur un ensemble de données concernant des tests de types différents, effectués sur les éléments matériel d'un système de production :

R (libellé matériel, code marque, libellé marque, type de test, date du test, résultat du test) n'est pas en 1ère forme normale car aucun attribut ne peut être clé primaire : le libellé matériel peut être identique pour plusieurs éléments.

R (code matériel, libellé matériel, code marque, libellé marque, code type de test, libellé du test, date du test, résultat du test) n'est pas en 1ère forme normale car on peut faire plusieurs tests sur un même matériel, ce qui exige de répéter les informations "code type de test", "libellé du test", "date du test", "résultat du test", dans un même nuple.
La relation doit être éclatée en deux, pour être exprimée en 1ère forme normale :

R-MATERIEL (code matériel, libellé matériel, code marque, libellé marque)
R-TEST (code matériel, code type, libellé test, date du test, résultat du test)
Les deux relations ne comportent que des attributs sans répétition. Dans R_TEST, la clé primaire est composée de "code matériel" et "code type" : un type de test peut concerner plusieurs matériels, un matériel peut être testé plusieurs fois, mais chaque matériel ne subit qu'une fois un type de test donné.

2ème forme normale

Une relation est dite en deuxième forme normale si elle est en première forme normale, et si tout attribut n'appartenant pas à la clé primaire ne dépend pas que d'une partie de cette clé.

R-TEST(code matériel, code type, libellé test, date du test, résultat du test)

n'est pas en 2ème forme normale car l'attribut "libellé test" ne dépend que du "code type" et pas du "code matériel" ;

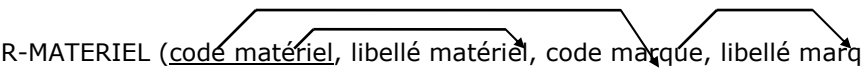
La relation doit éclatée en deux, pour être exprimée en deuxième forme normale :

R-TEST (code matériel, code type, date du test, résultat du test)

R-TYPETEST (code type, libellé test)

3ème forme normale

Une relation est dite en troisième forme normale si elle est en deuxième forme normale, et si toutes les dépendances fonctionnelles issues de la clé primaire sont directes



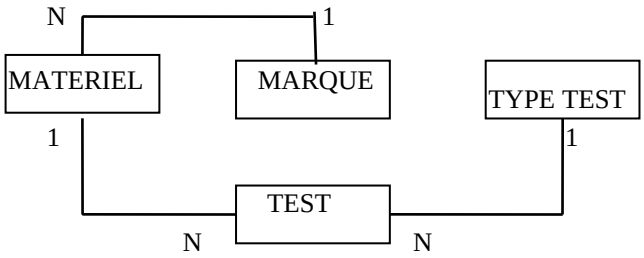
La dépendance entre "code matériel" et "libellé marque" n'est pas directe, "libellé marque" est en dépendance fonctionnelle directe avec le "code marque".
La relation doit être éclatée en deux, pour être exprimée en troisième forme normale :

R-MATERIEL (<u>code matériel</u>, libellé matériel, code marque)
R-MARQUE (<u>code marque</u>, libellé marque)
Le schéma conceptuel final de la base de données est donc :

R-MATERIEL (<u>code matériel</u>, libellé matériel, code marque)
R-MARQUE (<u>code marque</u>, libellé marque)
R-TYPETEST (<u>code type</u>, libellé test)
R-TEST (<u>code matériel, code type</u>, date du test, résultat du test)

<u>Commentaires:</u>

- Le schéma conceptuel fait apparaître 3 relations entités : **R-MATERIEL, R-MARQUE, R-TYPETEST**
- et la relation association **R-TEST** qui réalise le lien **Matériel <--> Type test** de type N:M
- Le lien fonctionnel **Matériel <--> Marque** de type N:1 est réalisé par la présence du "code marque" dans la relation R-MATERIEL.



II Démarche de création d'une base de données

Avant la création de la base de données un travail d'analyse préalable est indispensable. Il est nécessaire d'analyser le problème à traiter en partant des résultats à obtenir (en sortie) avec leur fréquence.
Etudions par exemple la création d'une base de données de gestion d'un Centre sportif.
LE PROBLEME A RESOUDRE

Le Centre sportif **NATURE ET SANTE** permet à ses adhérents d'utiliser des équipements sportifs, sur certains sites moyennant une cotisation dont le montant est fonction des équipements nécessités par l'activité choisie et le lieu de pratique.
Les activités possibles et les cotisations annuelles sont les suivants :

• BALNEO	2 035,00 F
• GOLF	4 200,00 F
• GYMNASTIQUE	1 815,00 F
• MUSCULATION	2 420,00 F
• SQUASH	1 320,00 F
• TENNIS	3 500,00 F

Le montant du droit d'entrée par lieu de pratique est le suivant :

• Decazeville	150,00 F
• Millau	190,00 F
• Rodez	300,00 F
• Saint Affrique	90,00 F
• Villefranche	160,00 F

Les cotisations sont payées pour l'année au début du mois de janvier.
Les adhérents peuvent choisir plusieurs activités et utiliser les équipements correspondant aux activités choisies, quand bon leur semble, aux heures d'ouverture de chaque lieu. Chaque adhérent ne s'inscrit que dans un seul lieu de pratique.
Le responsable du Centre souhaite gérer sur Base de données les cotisations d'environ 500 adhérents. De plus, il voudrait gérer les renseignements concernant les adhérents, les activités, les lieux, les tarifs, et pouvoir éditer les états correspondants.

LES RESULTATS A OBTENIR

Recenser tous les résultats que votre application doit pouvoir vous fournir. Il s'agit généralement d'états à produire. Ces états doivent contenir des données. Une maquette papier des états peut être réalisée afin de ne rien oublier.
Si nous reprenons notre exemple, les résultats à obtenir sont :

- la liste des adhérents avec leur code, nom, prénom, date de naissance, adresse, code postal, ville et numéro de téléphone
- la liste des équipements mis à leur disposition avec le code, le nom et le tarif d'utilisation
- la liste des adhérents et des équipements qu'ils utilisent, ainsi que le montant payé.
- …

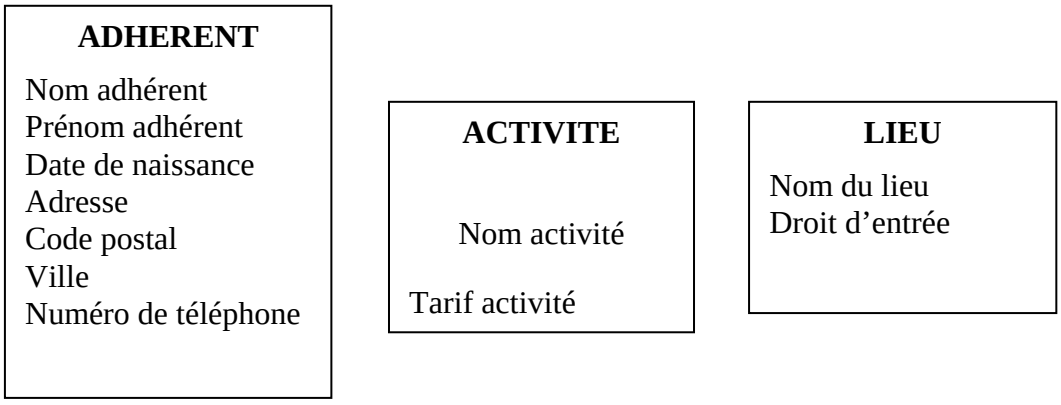
LE DICTIONNAIRE DES DONNEES

Il faut alors créer le dictionnaire des données c'est-à-dire recenser tous les renseignements à gérer sans distinguer ce à quoi ils se rapportent.
Nous aurons donc :

- Nom adhérent
- Prénom adhérent
- Date de naissance
- Adresse
- Code postal
- Ville
- Numéro de téléphone
- Nom activité
- Tarif activité
- Lieu de pratique
- Droit d'entrée

LA DEFINITION DES ENTITES

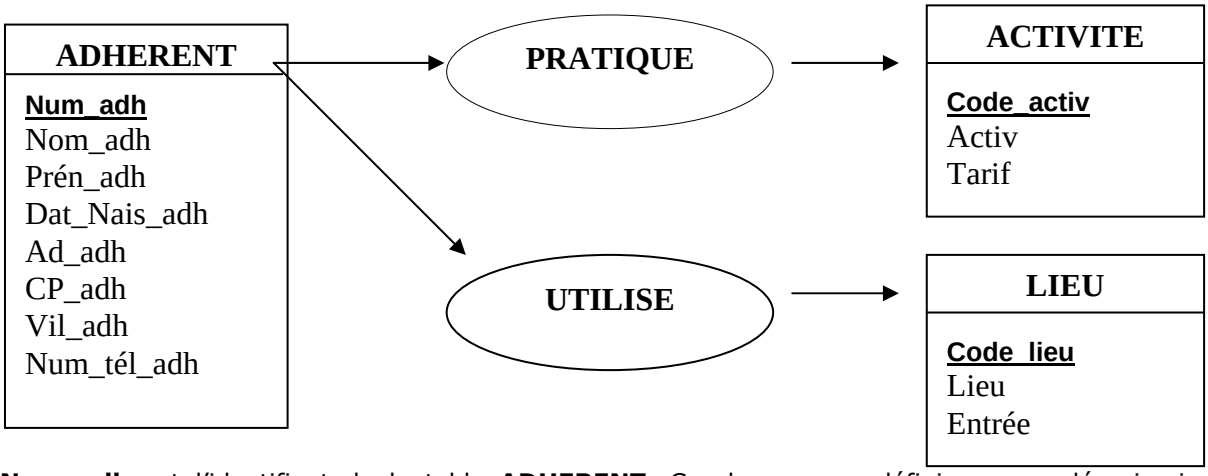
L'entité peut être un individu (client, adhérent), un bien (article, dépôt, magasin, équipement…), un concept (description d'une commande, inscription…).
Nous voyons apparaître ici trois entités : les **adhérents** les **activités** et les **lieux de pratique.**
Il s'agit maintenant de définir à quelle entité se rapportent les données recensées plus haut, c'est-à-dire de quel **objet** ou **entité** elles deviennent **l'attribut** (ou la **caractéristique**).
Nous pouvons définir le schéma qui suit :



A chaque entité correspondra une table dans la base de données.

LE MODELE ENTITE ASSOCIATION

L'association est un lien entre 2 (ou plusieurs) entités.
Entre l'entité **ADHERENT** et l'entité **ACTIVITE**, l'association correspond à la notion de **PRATIQUE** de l'activité, et est matérialisée par le verbe **Pratiquer.**
Entre l'entité **ADHERENT** et l'entité **LIEU**, l'association correspond à la notion d'utilisation et est matérialisée par le verbe **Utiliser**
De plus nous allons rajouter un **identifiant unique** dans chaque table sous forme de **Numéro** ou de **Code.**
Le **modèle Entité Association** prend l'allure suivante :



Num_adh est l'identifiant de la table **ADHERENT**. Ce champ sera défini comme clé primaire indexé sans doublon.
Code_activ est l'identifiant de la table **ACTIVITE**. Il sera défini comme clé primaire indexé sans doublon.
Code_lieu est l'identifiant de la table **LIEU**. Il sera défini comme clé primaire indexé sans doublon.

LES REGLES DE GESTION

Ce sont les règles qui régissent notre application. Ici un adhérent peut pratiquer **plusieurs activités**, sur **un seul lieu.** Il suffira qu'il paie le tarif des cotisations correspondant aux équipements utilisés, et le droit d'entrée sur le lieu de pratique.

LE MODELE RELATIONNEL

Nous devons maintenant créer le modèle relationnel.
Les activités

- ***Un adhérent peut pratiquer plusieurs ACTIVITES***
- ***Une activité peut être pratiquée par plusieurs ADHERENTS***

Il y a donc une relation de **plusieurs à plusieurs** entre les tables **ADHERENT** et **ACTIVITE**.
Avec ACCESS, il n'est pas possible de créer un tel type de relation directement entre deux tables. Il faut nécessairement ***transiter par une table intermédiaire***. Pour cela, il faut ***remplacer l'association*** matérialisée par le verbe utiliser par une nouvelle ***table*** qui servira de lien entre les 2 autres tables.
Cette nouvelle table que nous appellerons **PRATIQUE** comprendra donc les champs suivants :

Num_adh
Code_activ

N.B *Ces deux champs correspondent aux clés primaires des deux autres tables.*
*Nous établirons une relation de type **un** à **plusieurs** entre le champ **Num_adh** de la table **ADHERENT** et le champ **Num_adh** de la table **PRATIQUE.***
Nous établirons une relation de type un à plusieurs entre le champ Code_activ de la table ACTIVITE et le champ Code_activ de la table PRATIQUE.

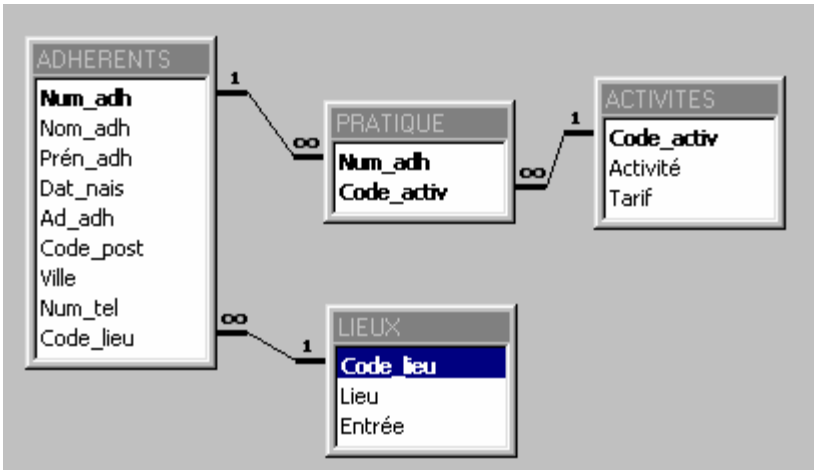
Le lieu de pratique

Un adhérent ne pratique que sur un seul lieu.
Un lieu peut recevoir plusieurs adhérents.

N.B *Nous avons donc une **relation de un à plusieurs** entre la table **LIEU** et la table **ADHERENT**.*

*Pour créer cette relation, nous allons devoir rajouter dans la table **ADHERENT** le code du lieu de pratique, afin d'établir la relation directe entre le champ **Code_lieu** de la table **LIEU** et le champ **Code_lieu** de la table **ADHERENT**.*

Le modèle relationnel sera donc le suivant



III Les redondances

Exemple

Il est nécessaire d'éviter les redondances dans le modèle relationnel. Prenons par exemple la table suivante qui concerne les propriétaires de véhicules :

nom	date	tél	n° immat	marque	type	cv	coul
Durand	10/2/88	23.32.32.23	3344RF45	Renault	R25	9	bleu
Dupont	8/10/88	62.62.52.55	7787FG56	Peugeot	405GR	7	vert
Pagnol	7/7/89	76.45.34.34	554FG22	Volvo	245	8	blanc
Pagnol	21/4/90	76.45.34.34	667TG22	Peugeot	305	6	gris
Duval	15/8/90	78.25.68.52	129DR75	Renault	R25	9	blanc

Elle pose dans son utilisation un certain nombre de problèmes, liés à la redondance des données.
Données redondantes

La table fait apparaître une personne et ses coordonnées autant de fois qu'elle possède un véhicule.

Pagnol	7/7/89	76.45.34.34	554FG22	Volvo	245	8	blanc
Pagnol	21/4/90	76.45.34.34	667TG22	Peugeot	305	6	gris

Si Mr Pagnol change de N° de téléphone, il faut s'assurer que la mise à jour s'effectue bien sur les deux enregistrements le concernant.

Une autre redondance est liée à la correspondance Marque, Type, CV

Durand	10/2/88	23.32.32.23	3344RF45	Renault	R25	9	bleu
Duval	15/8/90	78.25.68.52	129DR75	Renault	R25	9	blanc

Pour chaque propriétaire ayant une R25, il faudra saisir la marque et la puissance.

De plus, un même véhicule peut passer entre les mains de plusieurs propriétaires. Il faudra alors saisir toutes ces caractéristiques lorsqu'il changera de mains.

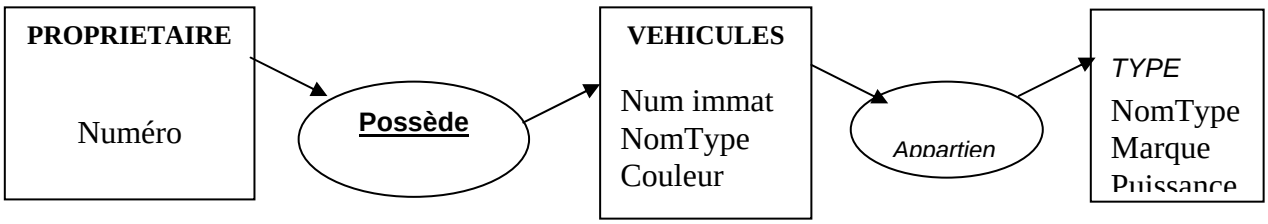
Solution

Les champs que nous trouvons dans cette table sont les attributs d'entités différentes. Nous allons rattacher ses attributs aux entités qu'ils caractérisent

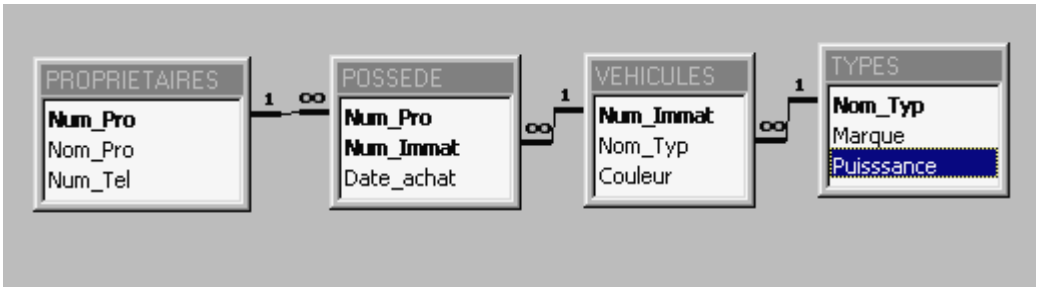
Nom et **Numéro de téléphone** caractérisent l'entité **PROPRIETAIRE**
Numéro d'immatriculation, **Marque**, **Type** et **Couleur** caractérisent l'entité **VEHICULE**
Marque et **Puissance** caractérisent l'entité **TYPE**.

L'entité **PROPRIETAIRE** et l'entité **VEHICULE** sont liées par la notion de **Possession**. La relation est matérialisée par le verbe **Posséder :** En effet, un propriétaire possède un ou plusieurs véhicules. Mais, un même véhicule pourra avoir été possédé par plusieurs propriétaires successifs. Nous avons donc entre ces deux entités une relation de plusieurs à plusieurs.
L'attribut **Date d'achat** ne caractérise pas l'une des entités mises en évidences ci-dessus. Par contre elle caractérise le moment ou le propriétaire va posséder le véhicule.

L'entité **VEHICULE** et l'entité **TYPE** seront liées par la notion d'appartenance. La relation est matérialisée par le verbe **Appartenir**. En effet un véhicule appartient à un **TYPE** et un seul.



Le modèle relationnel aura cette allure :



La relation entre la table **PROPRIETAIRES** et la table **VEHICULES** est matérialisée par la table **POSSEDE**. Celle-ci a comme attributs les deux clés primaires des deux tables et le champ **Date d'achat**, point de départ de la possession du véhicule.

La relation entre la table **VEHICULES** et la table **TYPES** ne nécessite pas la création d'une table intermédiaire puisqu'il s'agit d'une relation de **un à plusieurs** de la table **TYPES** vers la table **VEHICULES**.

√

PARTIE 3 CREATION ET MANIPULATION DES BASES DE DONNEE AVEC MS ACCESS

PARTIE 4 CREATION ET MANIPULATION DES BASES DE DONNEE AVEC MS ACCESS

I Création des tables

La **table** est Le principal organe de stockage de données d'ACCESS.

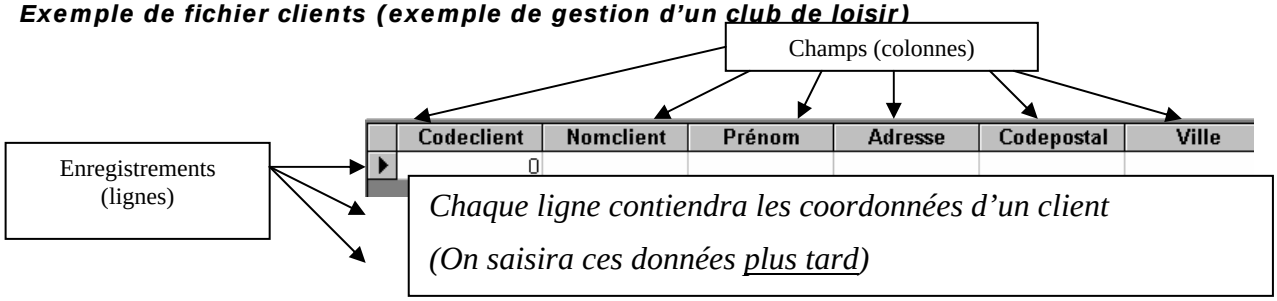
Pour qu'une base de données Access existe, il faut au moins une table. Il peut bien évidemment y en avoir plusieurs.

Les tables servent à emmagasiner les données stables (*quand on dit données stables, cela veut dire que leur **structure** est stable ; par exemple une table **clients** contiendra toujours des noms, des adresses, etc. et ces éléments se retrouveront à un emplacement déterminé)*.

La table ressemble physiquement à une feuille de calcul Excel : il y a des colonnes, qui prennent ici le nom de **champs** et des lignes qu'on appelle **enregistrements**.

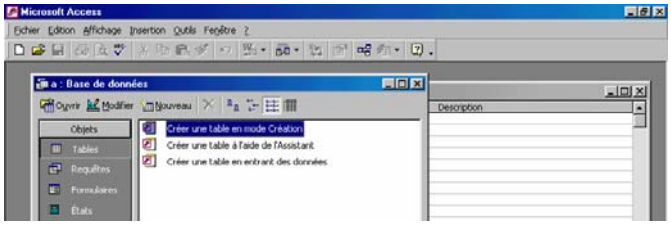
Créer une table, c'est d'abord décider de sa structure c'est-à-dire quels champs il faut créer et quel sera leur type de contenu (alphabétique, numérique, etc.)

Exemple de fichier clients (exemple de gestion d'un club de loisir)

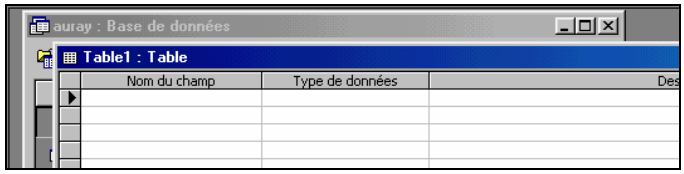


Création de la table ADHÉRENTS

- Double cliquez la première option

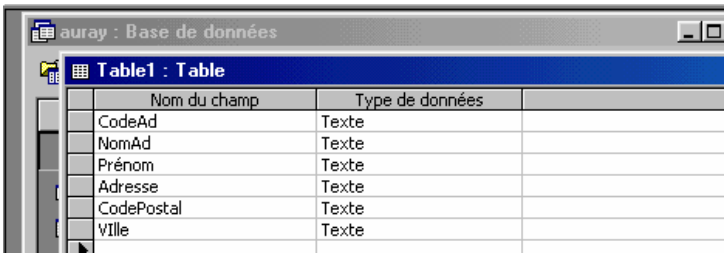


- Vous obtenez le panneau de création de structure de table



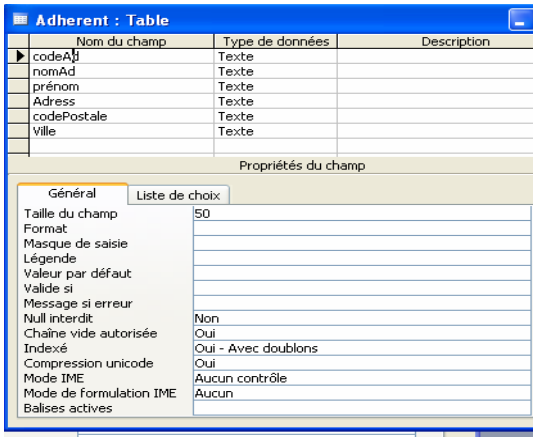
- Le travail va consister à décider les noms des champs puis à choisir le type de données qui figureront dedans. (*Par exemple, le nom de l'adhérent contiendra du texte exclusivement ; ce sera donc un champ de type Texte*).

- Entrez les noms des champs tels qu'ils apparaissent ci-contre ; pour le moment ne touchez pas au type de données.
- Vous obtenez

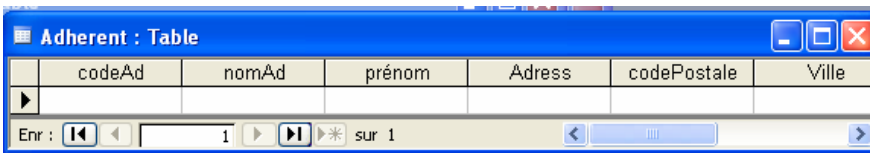


Jusque-là, vous travaillez sur la structure c'est-à-dire *l'envers du décor*. Pour visualiser ce que vous venez de créer, passez en mode feuille de données

Habituez-vous à passer du mode création  au mode feuille de données 



- Répondez **oui** à la question : *la table doit d'abord être enregistrée*, donnez-lui le nom **ADHÉRENTS** et répondez **non** à la question *laisser ACCESS définir une clé primaire*.
- Résultat (*rappel : vous vous trouvez en mode feuille de données*)

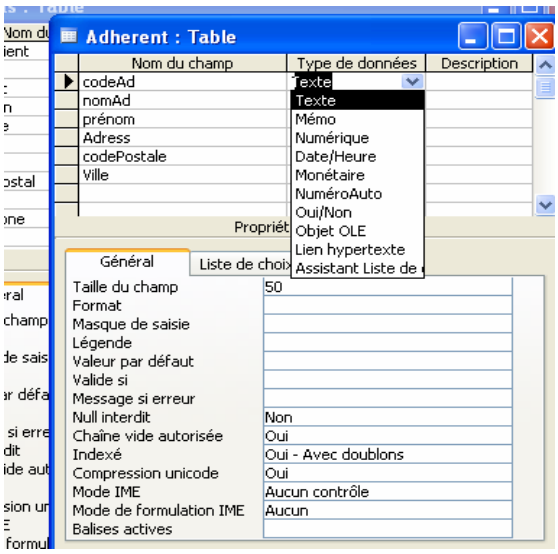


- Pour continuer à travailler sur la structure de la table, il faut repasser en mode création.

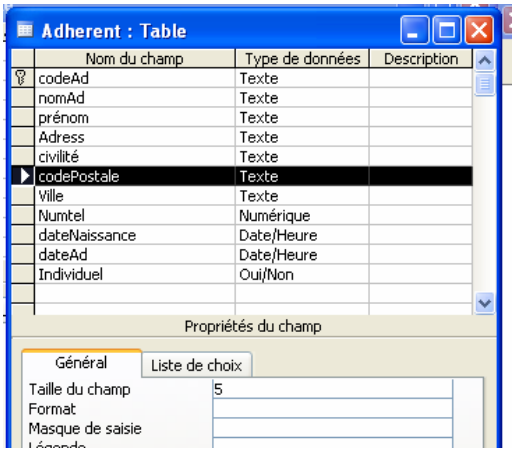
Cliquez sur l'icône 

Définir les caractéristiques des champs

- Vous allez maintenant définir le type de vos différents champs.



- Cliquez après le mot Texte sur la même ligne que CodeAd et choisissez Numérique. Cela veut dire que le code de vos adhérents sera constitué de chiffres. Remarque : un champ Texte peut contenir des lettres ou des chiffres, alors qu'un champ Numérique ne peut contenir que des chiffres.
- Laissez les autres champs en texte.
- Pour le CodePostal ; placez-vous sur sa ligne et limitez-le à 5 caractères en corrigeant les 50 en 5 dans la zone Taille de champ de l'onglet Général en bas

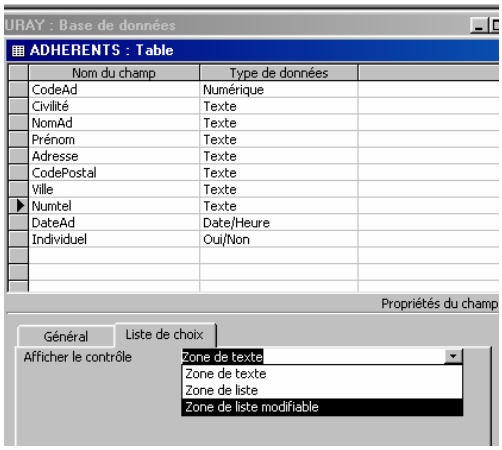


- Passez en mode Feuille de données pour voir, puis repassez en création.
- Vous allez maintenant ajouter d'autres champs à la table ADHÉRENTS.
- Sur la ligne après Ville, ajoutez Numtel. Vous allez le laisser en texte (nous avons vu qu'un champ texte peut contenir des chiffres), mais pour que la saisie soit plus aisée par la suite, vous allez définir un masque de saisie.
- Cela veut dire que la zone à remplir pour le numéro de téléphone se présentera ainsi : __ __ __ __ __ et que vous n'aurez pas à saisir les espaces. Il faut savoir que tout champ contenant un masque de saisie, même s'il doit recevoir des chiffres, doit être obligatoirement un champ Texte.
- Le curseur étant sur la ligne de Numtel, cliquez dans Masque de saisie puis sur les pointillés qui vont lancer l'assistant. Laissez-vous guider par les écrans ; vous obtenez en fin de compte les signes suivants sur la ligne Masque de saisie : 00\ 00\ 00\ 00\ 00;;_ (*remarque : en tapant ces signes à la main, on obtiendrait le même résultat*).
- Ajoutez à la table **ADHÉRENTS** les champs suivants:

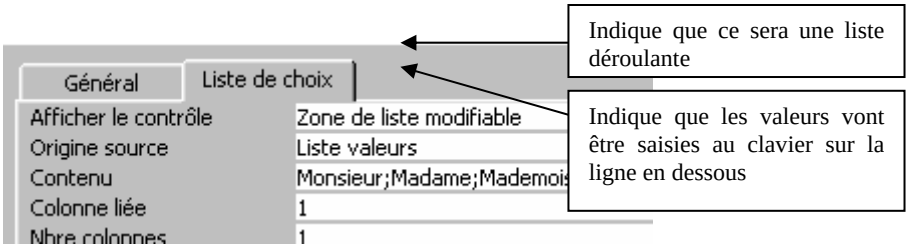
  DateAd, champ de type Date/heure — pour saisir la date d'adhésion du membre
  Individuel, champ de type Oui/Non — pour savoir s'il appartient ou non à un comité d'entreprise

- Passez *en* Feuille de données pour voir (*acceptez l'enregistrement de la table*).
  - Dans ce mode, cliquez dans le champ Numtel pour voir si le masque est actif. Repassez en mode Création.
  - Il manque un champ Civilité ; vous allez insérer une nouvelle ligne.
  - Sélectionnez la ligne NomAd et appuyez sur la touche Inser du clavier. Une nouvelle ligne est créée. Saisissez Civilité comme nom de champ.

- Ce serait intéressant de créer une liste déroulante pour les civilités, c'est à dire une liste dans laquelle il suffirait de choisir au lieu d'avoir à saisir.

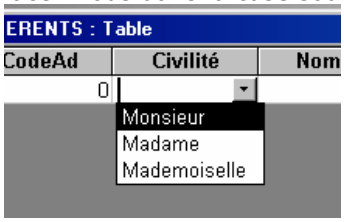


- Le curseur étant sur la ligne de Civilité, cliquez sur l'onglet Liste de choix puis sur la ligne Zone de texte et choisissez Zone de liste modifiable.



- Sur la ligne en-dessous, choisissez Liste de valeurs
- Sur la ligne en-dessous, saisissez : Monsieur;Madame;Mademoiselle (*n'oubliez pas les points virgules et ne faites pas d'espace*)

- Passez en mode visualisation, placez-vous dans la case sous Civilité : vous obtenez



Récapitulons : vous avez créé la trame capable d'accueillir les données des adhérents.

Pour le moment, vous n'avez pas saisi ces données ; il est en effet préférable d'avoir créé toutes les tables utiles avant de les remplir, car s'il y avait une erreur dedans, en la corrigeant on risquerait de perdre tout ou partie de la saisie, c'est-à-dire … d'avoir travaillé pour rien !

Notion de clé primaire

Dans un système de gestion de bases de données comme Access, pour toute table que l'on crée il faut se poser une question : **Quel est le champ qui contiendra pour chaque enregistrement une caractéristique unique** ? (*Vous allez comprendre : s'il s'agit de personnes, la caractéristique que l'on est absolument sûr de ne pas retrouver chez plusieurs personnes différentes, c'est son numéro INSEE !*) Il est indispensable que toute table comporte un tel champ, de façon que le système puissse identifier de manière certaine chaque enregistrement.
Dans la table **ADHÉRENTS**, quel champ va servir d'identifiant ?
C'est bien sûr le Code Adhérent : chaque adhérent aura le sien.
Pour indiquer au système que le champ CodeAd est l'identifiant unique, vous allez poser dessus une clé (*on dit une clé primaire*).

- En mode création, sélectionnez la ligne de CodeAd et cliquez sur l'icône de clé. Un symbole de clé s'insère en début de ligne.

Pour Access, cela signifie qu'il est impossible de donner deux fois le même code dans la table.

***Remarque** : la clé est généralement posée sur un champ numérique mais techniquement rien n'empêche de la poser sur un champ texte, dans la mesure où on est sûr que le contenu sera unique dans la table. Il est possible aussi de poser une clé sur deux champs en même temps : à ce moment-là ce sera l'association des deux contenus qui devra être unique.*

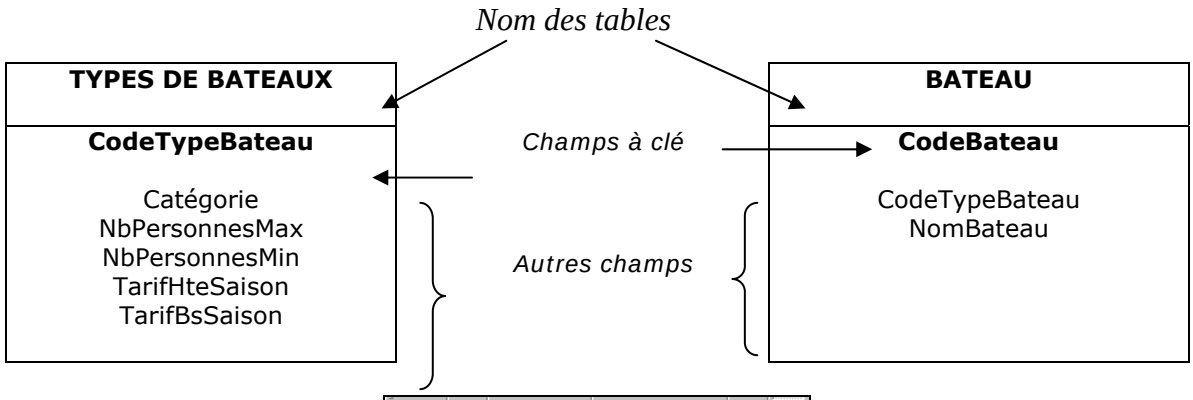
- Refermez la table **ADHÉRENTS** en enregistrant les modifications.

Création des tables pour la location des bateaux

Dans la plaquette d'AURAY PLAISANCE, la page **Tourisme fluvial** décrit la flotte.

On peut louer trois types de bateaux, mais il y a plusieurs bateaux de chaque type ; vous aurez donc besoin de deux tables : la table **TYPES DE BATEAUX**, qui contiendra pour chaque enregistrement les caractéristiques du type de bateau. La table **BATEAUX** contiendra le nom propre de chaque bateau et sera rattachée à la table **TYPES DE BATEAUX** (de cettemanière, vous n'aurez à saisir les caractéristiques qu'une fois par type de bateau). Dans la partie Tables de la fenêtre Base de données, cliquer Créer une table en mode Création.

Réfléchissez maintenant aux champs nécessaires dans chacune de ces deux tables. Il s'agit de chercher dans le descriptif de la flotte ce qui se rapporte au Type de bateau et ce qui se rapporte à un bateau en particulier (de cette analyse dépend le bon fonctionnement de la base de données). On peut ainsi schématiser la structure de nos deux tables :



Nom du champ	Type de données
CodeTypeBateau	Numérique
Catégorie	Texte
NbPersonnesMin	Numérique
NbPersonnesMax	Numérique
TarifHtSaison	Monétaire
TarifBsSaison	Monétaire

- Depuis la fenêtre Base de données, dans l'élément Tables, faites Créer une table en mode création.

- Constituez la table **TYPES DE BATEAUX** comme sur le modèle ci-contre. N'oubliez pas de poser la clé sur le premier champ. Vérifiez que vous ne vous êtes pas trompé(e) de type de données pour chaque champ. Pour les tarifs, mettez en plus de Monétaire un format Euro dans l'onglet du bas.

Nom du champ	Type de données
CodeBateau	Numérique
NomBateau	Texte
CodeTypeBateau	Numérique

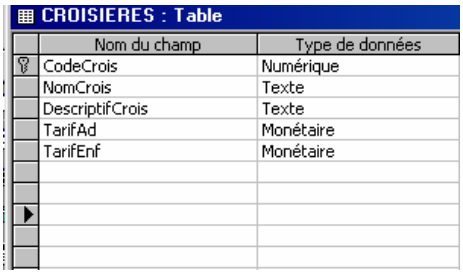
- Refermez et enregistrez la table sous le nom **TYPES DE BATEAUX** (*les données seront saisies plus tard, de même que pour la table **ADHÉRENTS***)

- Créez la table **BATEAUX** comme ci-contre en posant la clé sur ce champ. Refermez et enregistrez la table sous le nom **BATEAUX**.

On pourrait aller chercher le CodeTypeBateau dans la table **TYPE DE BATEAUX.**

Création de la table CROISIÈRES

Vous savez maintenant créer une table.



- Créez la table **CROISIÈRES** comme ci-contre. Pour les champs monétaires, demandez un format euro.
- Refermez la table.

La base **AURAY** comporte maintenant 4 tables : **ADHÉRENTS**, **BATEAUX, TYPES DE BATEAUX** et **CROISIÈRES**.

Ainsi juxtaposées, les 4 tables ne communiquent pas entre elles. Les liaisons que vous allez établir dans le chapitre suivant vont les rendre communicantes.

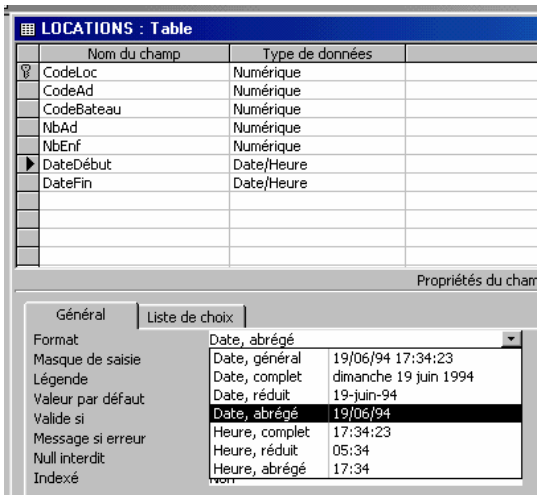
Quelles relations peut-on établir entre ces tables ? Pour trouver la réponse à cette question, on utilise des phrases avec sujet, verbe, complément :

un adhérent **réserve** *une croisière*,

un adhérent **loue** *un bateau*.

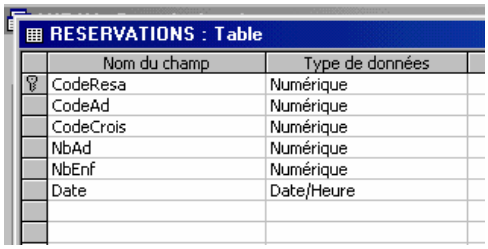
Le fait de réserver ou louer nous amène à créer deux autres tables : une table des réservations et une table des locations dans laquelle seront stockés les éléments propres aux réservations ou aux locations (*exemple : la date ; celle-ci est bien une caractéristique de la réservation et non du bateau).*

Création de la table LOCATIONS



- Créez la table **LOCATIONS** comme ci-contre
- Placez la clé sur le champ CodeLoc
- Pour DateDébut et DateFin, choisissez un format date,abrégé (en vous plaçant sur la ligne concernée, cliquer dans Format, onglet Général).
- Refermez la table **LOCATIONS**.

Création de la table RÉSERVATIONS

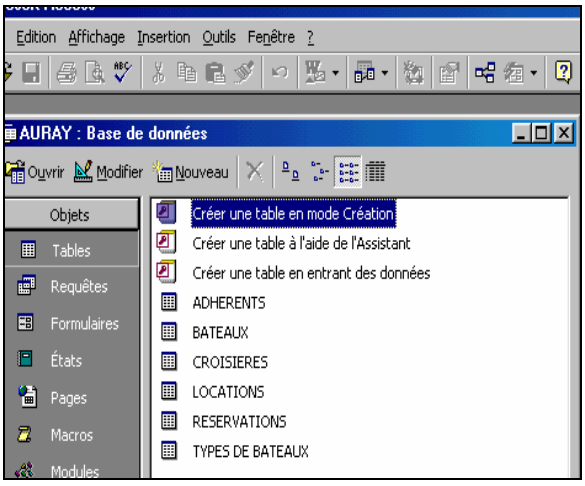


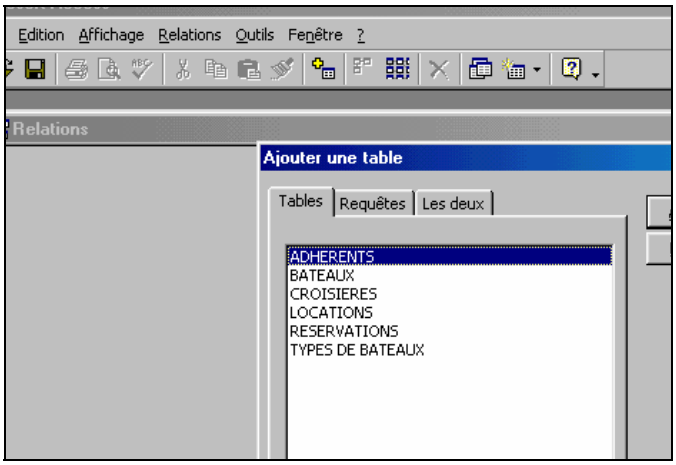
- Créez la table **RÉSERVATIONS** comme ci-contre.
- Placez la clé sur le champ CodeResa
- Pour la date, prenez un format Date, abrégé

Définition des relations

Cette partie du travail est particulièrement importante, car si vous rencontrez des problèmes ils se répercuteront tout au long du travail. Il faut donc impérativement que les relations fonctionnent parfaitement.

- Depuis la fenêtre de la Base de données, cliquez sur l'icône Relations

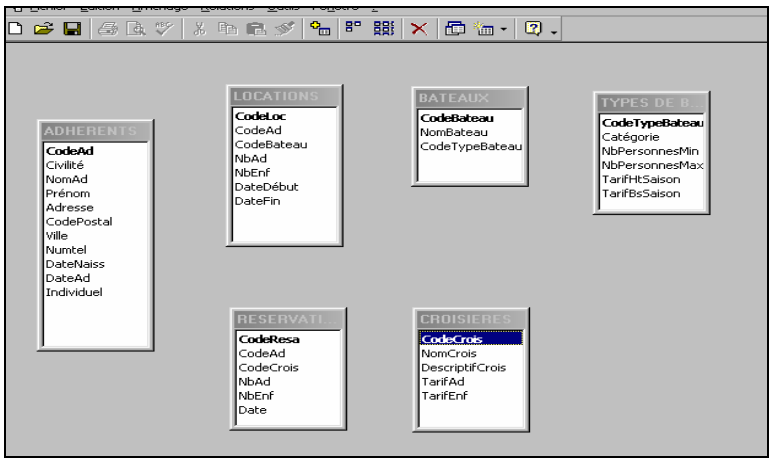




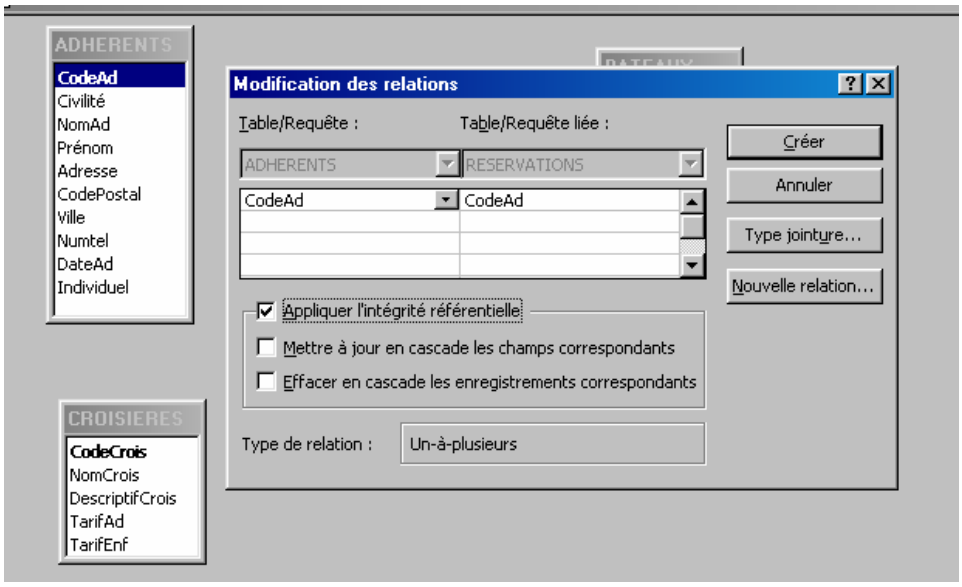
- Vous vous trouvez sur **ADHÉRENTS** ; cliquez Ajouter. Déplacez-vous sur **BATEAUX**, cliquez Ajouter, et ainsi de suite sur **CROISIÈRES** puis **LOCATIONS** puis **RÉSERVATIONS** puis **TYPES DE BATEAUX.**
- Les 6 tables sont affichées. Cliquez Fermer. (*Si par mégarde vous avez ajouté deux fois une table, ôtez-la en appuyant sur la touche Suppr du clavier après l'avoir sélectionnée avec la souris*).

Les tables peuvent être redimensionnées et déplacées comme n'importe quelle fenêtre Windows.

- Aménagez-les pour obtenir ceci :



Remarquez que dans chaque table, le champ à clé se présente en caractères gras.
Il reste à tracer les relations.



- Placez-vous sur CodeAd de la table **ADHÉRENTS** et sans lâcher le clic, tirez-le pour le déposer sur le champ CodeAd de la table **RÉSERVATIONS**.
- Dans la fenêtre qui apparaît, cochez Appliquer l'intégrité référentielle puis cliquez sur Créer.

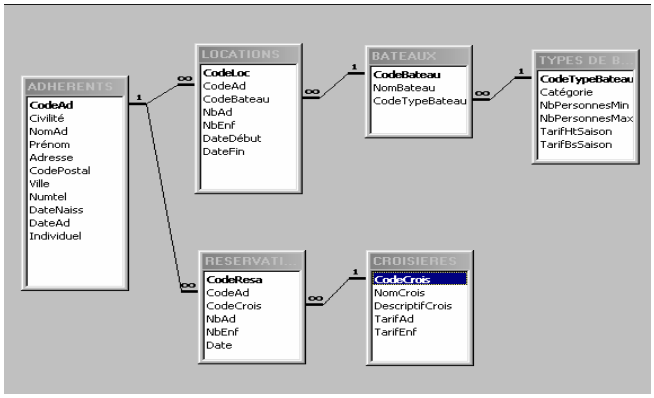
Le résultat est une ligne qui va de CodeAd de la table **ADHÉRENTS** jusqu'à CodeAd de la table **RÉSERVATIONS**.

Signification de cette ligne

Le système retrouve l'adhérent qui a réservé grâce à son code, indiqué dans la table des **RÉSERVATIONS**

1. Le chiffre 1 et le symbole ∞ (infini) signifient que pour 1 adhérent, plusieurs réservations sont possibles. En revanche, une réservation ne peut être attribuée qu'à un et un seul adhérent.
2. Le sens de la relation, de 1 à ∞ indique aussi que lorsque vous saisirez les données, il faudra d'abord saisir celles de la table du côté du 1 (*en clair : il ne sera pas possible d'enregistrer une réservation pour un adhérent qui n'existe pas encore*).

- Définissez les autres relations pour obtenir ceci (*chaque fois il suffit de prendre le champ de démarrage et de le déposer sur le champ d'arrivée, attention de ne pas vous tromper).*
- *Vérifiez bien que votre écran est conforme à cette image*



A partir de maintenant, le modèle est prêt. On sait quelles sont les tables, quels sont les champs qu'elles contiennent avec quel type de données à l'intérieur, on sait où se trouvent les clés, et on sait quelles relations unissent les tables.

- Enregistrez les modifications apportées à la fenêtre des relations.

L'essentiel de la base est constitué, la saisie des données va pouvoir se faire.

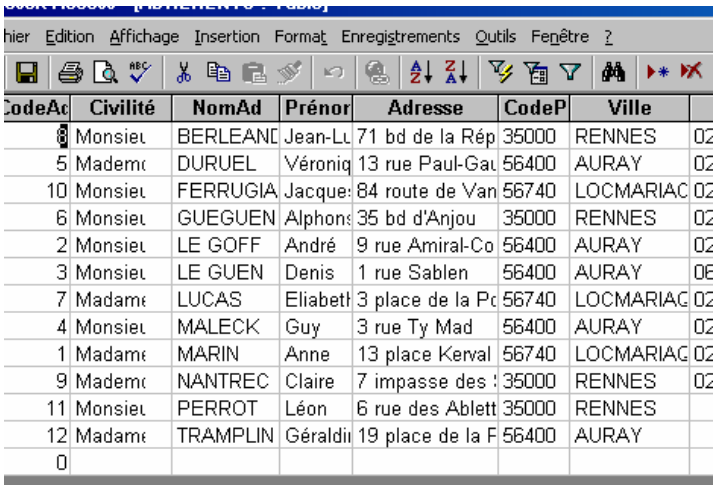
II Saisie de donnée et création de formulaire

Saisie de données en mode table

Rappel : d'après le modèle que vous avez déterminé, la saisie des données ne peut pas se faire dans n'importe quel ordre ; la table des réservations devra être saisie en dernier, puisque les chiffres 1 sont tous du côté des autres tables (voir fenêtre des relations).

Prenez avec vous la page 8 (**fichier adhérents**).

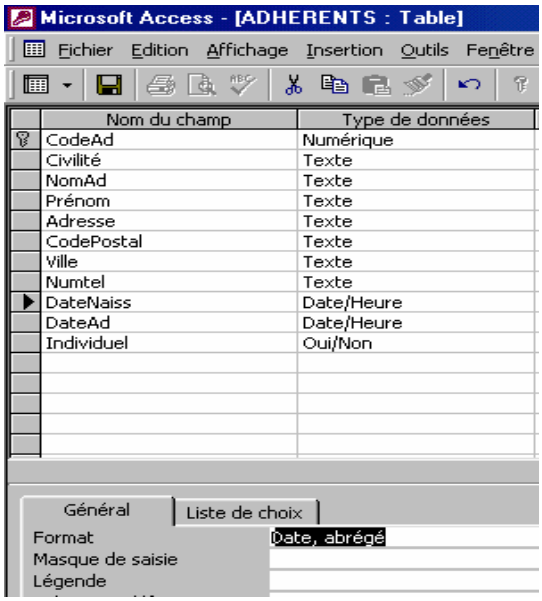
Depuis la fenêtre Base de données, dans les tables, double-cliquez sur **ADHÉRENTS** pour ouvrir la table en mode Feuille de données. Vous obtenez :



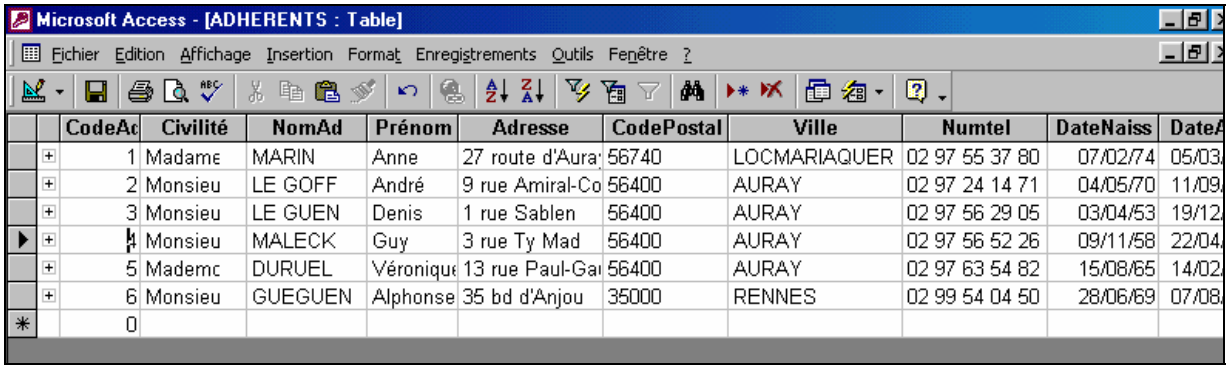
CodeAd	Civilité	NomAd	Prénor	Adresse	CodeP	Ville	
8	Monsieu	BERLEAND	Jean-Lu	71 bd de la Rép	35000	RENNES	02
5	Mademc	DURUEL	Véroniq	13 rue Paul-Gau	56400	AURAY	02
10	Monsieu	FERRUGIA	Jacque:	84 route de Van	56740	LOCMARIAC	02
6	Monsieu	GUEGUEN	Alphons	35 bd d'Anjou	35000	RENNES	02
2	Monsieu	LE GOFF	André	9 rue Amiral-Co	56400	AURAY	02
3	Monsieu	LE GUEN	Denis	1 rue Sablen	56400	AURAY	0E
7	Madame	LUCAS	Eliabeth	3 place de la Pc	56740	LOCMARIAG	02
4	Monsieu	MALECK	Guy	3 rue Ty Mad	56400	AURAY	02
1	Madame	MARIN	Anne	13 place Kerval	56740	LOCMARIAG	02
9	Mademc	NANTREC	Claire	7 impasse des :	35000	RENNES	02
11	Monsieu	PERROT	Léon	6 rue des Ablett	35000	RENNES	
12	Madame	TRAMPLIN	Géraldi	19 place de la F	56400	AURAY	
0							

- Vous allez saisir le premier adhérent de la liste. Cliquez dans le champ CodeAd, 1ère ligne, et saisissez 1 puis tabulateur pour passer au champ Civilité.
- Cliquez sur la petite flèche de la liste déroulante et choisissez Madame.
- Continuez à saisir les données du premier adhérent. Constatez que le masque de saisie pour le téléphone a bien fonctionné. Un petit problème se pose cependant : la date de naissance n'a pas été prévue. Vous allez remedier à cela.
- La structure de la table peut encore être modifiée si cela ne touche pas au champ qui comporte la clé. De plus il ne s'agit que d'ajouter un champ. Cliquez sur l'équerre pour passer en mode création.
- Sélectionnez la ligne DateAd et appuyez sur la touche Inser du clavier pour ajouter une ligne vierge.
- Créez le champ DateNaiss de type Date/Heure et dans l'onglet Général du bas donnez un format Date, abrégé.

- Le champ DateAd n'avait pas jusqu'ici de format Date,abrégé ; profitez de l'occasion pour le faire.
- Vous obtenez le résultat ci-contre



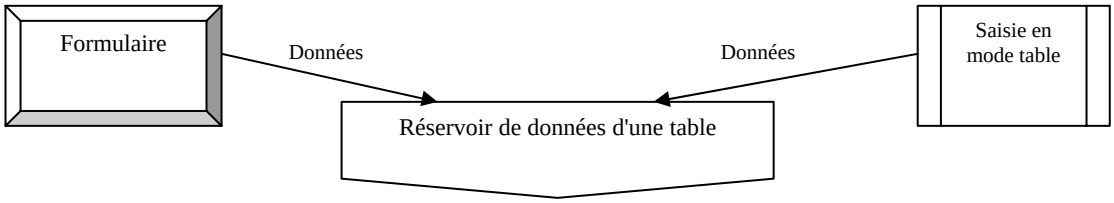
- Repassez en mode Feuille de données
- Continuez votre saisie jusqu'au dernier adhérent.
- Cochez le champ Individuel lorsque l'adhérent n'appartient pas à un comité d'entreprise
- Résultat à obtenir ci-dessous (*en raison de la dimension de notre page, le dernier champ Individuel n'apparaît pas sur l'image*)



La saisie que vous venez d'effectuer s'est faite en mode Table, c'est-à-dire dans un tableau où sont regroupées toutes les données de tous les enregistrements.

Ce n'est pas très convivial !!

Heureusement, on peut se faciliter la vie en créant une sorte de fenêtre de dialogue qui s'appelle *Formulaire*. La différence avec la table, c'est que le formulaire n'affiche qu'un enregistrement à la fois. (*Pour ceux qui connaissent la fonction Base de données d'Excel, cela correspond à l'affichage Grille*).

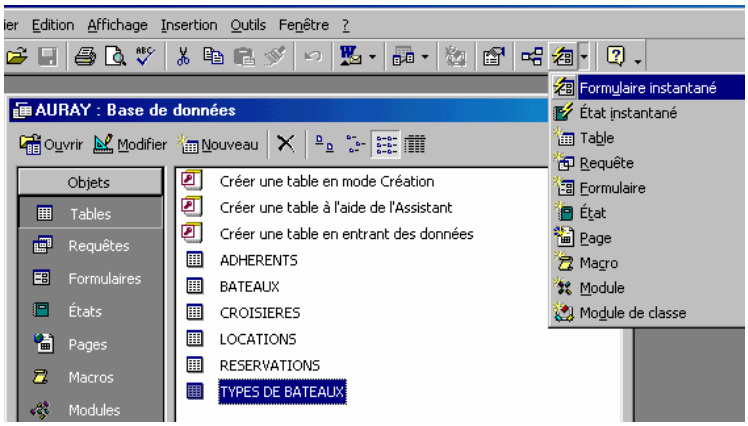


En fait les données sont toujours contenues dans la table ; si vous les saisissez depuis le formulaire, elles vont dans le même "*réservoir*".

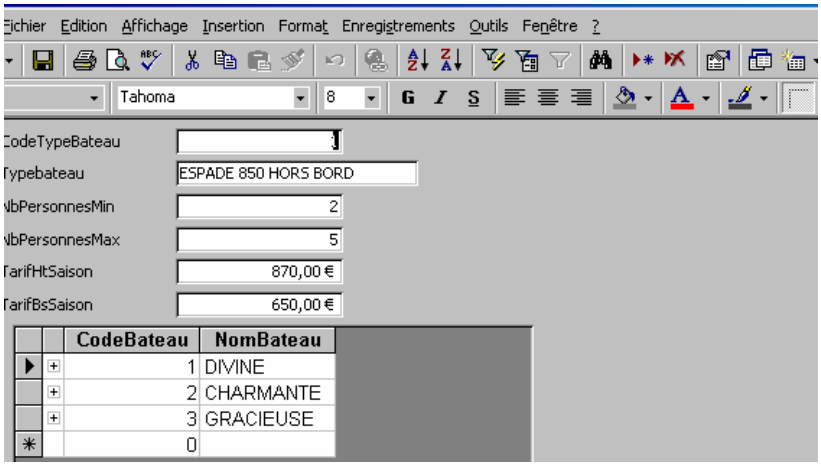
Saisie de données en mode formulaire

Grâce aux assistants d'Access, le travail va être facile.

Depuis la fenêtre Base de données, placez-vous sur la table **TYPES DE BATEAUX** et développez le menu déroulant Nouvel objet, choisissez Formulaire automatique.



C'est vraiment instantané ! Vous avez devant les yeux un formulaire tout prêt. Il n'y a plus qu'à entrer les données dedans et améliorer sa présentation, mais ceci est un détail que nous verrons par la suite.



- Saisissez les données du premier type de bateau en lui attribuant le code 1.
- Le système a prévu (en fonction des liaisons que vous avez déclarées) un sous-formulaire pour enregistrer les différents bateaux. Saisissez les données de chaque bateau comme ci-contre.
- Pour passez au deuxième enregistrement, cliquez tout en bas de la fenêtre sur la flèche de défilement. Pour le moment, cela indique que vous en êtes à la saisie de l'enregistrement 1 sur un total de 1.
- Saisissez les données du second type de bateau avec les trois bateaux concernés (attention dans le sous formulaire, les noms des bateaux seront numérotés 4, 5), puis les données du troisième type de bateau avec ses deux bateaux (à numéroter 6 et 7).

> *Remarque : Il est possible de demander à Access de numéroter lui-même les enregistrements avec un compteur automatique. Toutefois en situation d'apprentissage nous avons remarqué que cela pose beaucoup de problèmes car si l'apprenant se trompe et détruit un enregistrement, ce numéro n'est plus réutilisable. On risque de se retrouver en déphasage avec le support de cours. Mais dans la réalité, on utiliserait le compteur automatique. Il suffit de demander dans la structure de table un type de données NumAuto au lieu de Numérique.*

- Fermez et enregistrez le formulaire sous le nom **TYPES DE BATEAUX**.

Création du formulaire CROISIÈRES

- De la même manière que précédemment, créez à partir de la table **CROISIÈRES** un formulaire instantané et saisissez les données des croisières.
- Pour les descriptifs, utilisez une forme abrégée (*exemple : sortie 2 h passage écluse*)
- Quand il n'y a qu'un tarif, mettez le même pour adultes et enfants.
- Pour la Formule Moussaillons, mettez 0 dans tarif adulte.
- Refermez le formulaire en acceptant le nom **CROISIÈRES** proposé par le système.

Vous allez vérifier que les données saisies dans le formulaire sont bien présentes dans les tables.

- Dans la fenêtre Base de données, cliquez l'élément Tables et double-cliquez **CROISIÈRES** ; constatez que les données sont là. Refermez la table. Faites de même pour la table **TYPES DE BATEAUX**, puis pour la table **BATEAUX**.

Création du formulaire LOCATIONS

- Créez un formulaire instantané à partir de la table **LOCATIONS**. Ne saisissez rien pour le moment.

Gestion des événements courants

Ajouter des enregistrements dans une table

Aujourd'hui, deux nouveaux clients s'inscrivent à AURAY PLAISANCE ; avant de les enregistrer, créez un formulaire instantané à partir de la table **ADHÉRENTS**. Saisissez ensuite les données de :

LUCAS Elizabeth (Madame)	**BERLÉAND Jean-Luc**
3 place de la Poste	71 bd de la République
56740 LOCMARIAQUER	35000 RENNES
02 97 54 87 21	02 99 41 52 63
Née le 03/05/58	Né le 25/6/59
Individuel	Individuel

Modifier des données

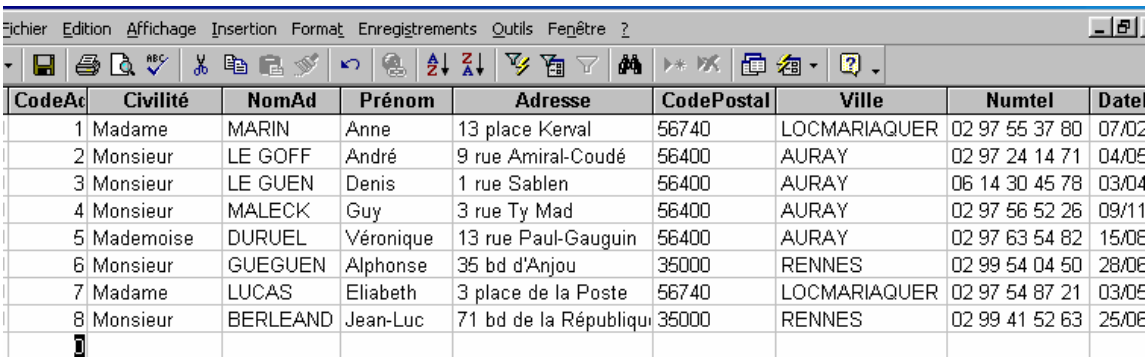
Monsieur LE GUEN Denis fait part de son nouveau numéro de téléphone : 06 14 30 45 78. Faites la modification (*vous pouvez pour cela utiliser le filtre par formulaire*).

Madame MARIN a changé d'adresse : désormais elle habite 13 place Kerval (*même localité*).

Trier, filtrer, rechercher dans une table selon des critères

Tri

- Ouvrez la table **ADHÉRENTS**. Vous voulez obtenir un tri alphabétique par noms d'adhérents.



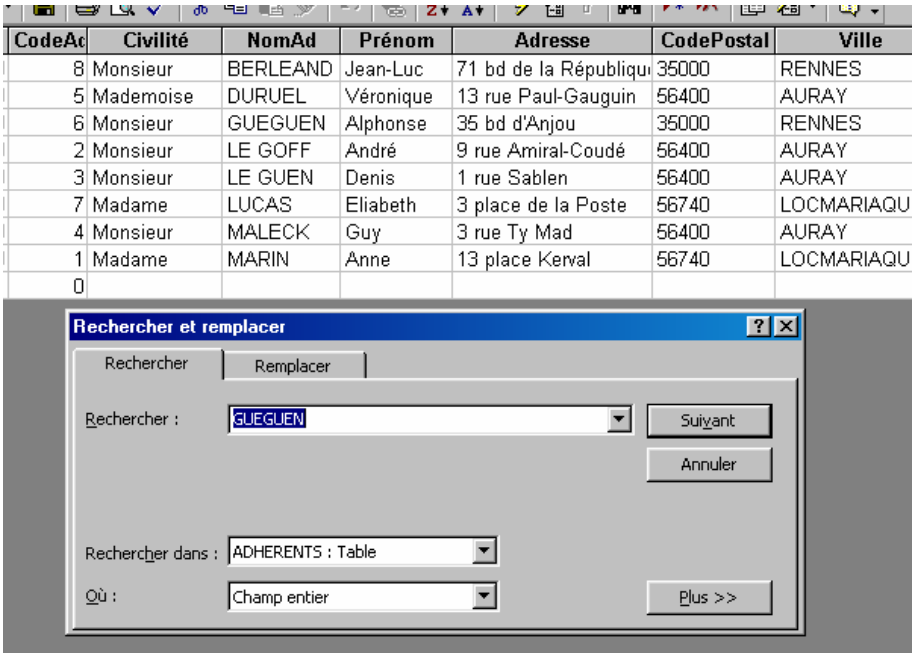
CodeAd	Civilité	NomAd	Prénom	Adresse	CodePostal	Ville	Numtel	Datel
1	Madame	MARIN	Anne	13 place Kerval	56740	LOCMARIAQUER	02 97 55 37 80	07/02
2	Monsieur	LE GOFF	André	9 rue Amiral-Coudé	56400	AURAY	02 97 24 14 71	04/05
3	Monsieur	LE GUEN	Denis	1 rue Sablen	56400	AURAY	06 14 30 45 78	03/04
4	Monsieur	MALECK	Guy	3 rue Ty Mad	56400	AURAY	02 97 56 52 26	09/11
5	Mademoise	DURUEL	Véronique	13 rue Paul-Gauguin	56400	AURAY	02 97 63 54 82	15/08
6	Monsieur	GUEGUEN	Alphonse	35 bd d'Anjou	35000	RENNES	02 99 54 04 50	28/08
7	Madame	LUCAS	Eliabeth	3 place de la Poste	56740	LOCMARIAQUER	02 97 54 87 21	03/05
8	Monsieur	BERLEAND	Jean-Luc	71 bd de la Républiqu	35000	RENNES	02 99 41 52 63	25/08
1								

- Sélectionnez le champ NomAd ; cliquez sur l'icône A/Z. Immédiatement, la table est triée.
- Constatez que les données de la ligne entière ont suivi le nom de l'adhérent (heureusement !)

Remarque : sous Excel, il peut arriver que les données soient destructurées si vous avez fait une sélection malencontreuse ; avec Access, cela ne peut pas arriver.

Recherche

- Vous recherchez l'adhérent GUEGUEN. Cliquez sur l'icône qui représente des jumelles. Remplissez la fenêtre dialogue comme suit.

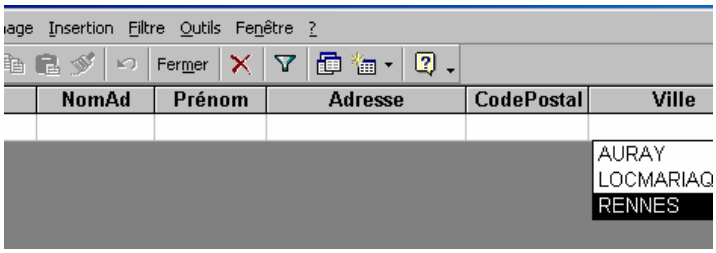


- Cliquez Suivant : le nom de GUEGUEN apparaît en surbrillance dans la liste.

Filtre

Vous voulez ressortir uniquement les adhérents qui habitent RENNES.

- Cliquez l'icône Filtrer par formulaire. Placez-vous dans le champ Ville et choisissez RENNES dans la petite liste. Ensuite, cliquez sur Appliquer le filtre.

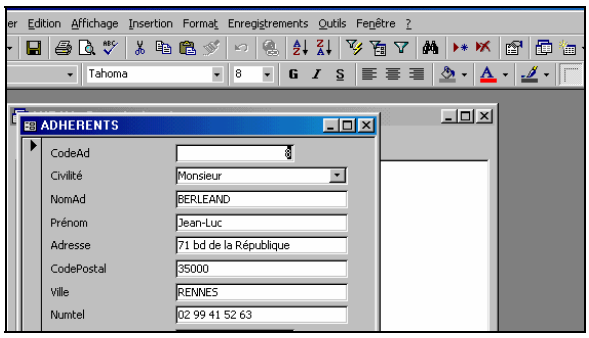


- Vous avez devant les yeux la liste des adhérents de Rennes. Pour annuler le filtre, désactivez l'icône (qui s'appelle maintenant Supprimer le filtre).
- Supprimer le critère Rennes.
- Filtrez les adhérents nés après le 1er janvier 1965 (*saisissez > 01/01/65 dans le champ DateNaiss*).

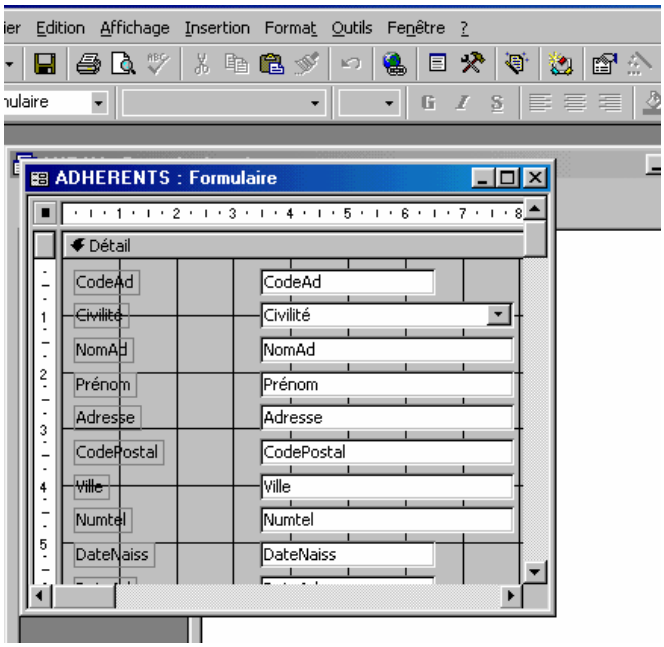
Faites d'autres essais à votre guise (*attention, n'oubliez pas d'ôter les critères entre deux essais, sinon le système va chercher des enregistrements qui correspondent à la fois à plusieurs critères et n'aura peut-être pas de réponse*).

Améliorer la présentation d'un formulaire

Jusqu'ici nous ne nous sommes pas intéressés à la présentation car d'autres choses étaient plus importantes… toutefois il est toujours plus agréable de travailler sur un écran convivial et c'est pourquoi vous allez consacrer un peu de temps à améliorer votre cadre de saisie. Ceci est dans votre intérêt personnel, mais il faut penser que, dans l'entreprise, ce sont peut-être d'autres personnes qui devront saisir des données et qu'il est bon de leur faciliter aussi le travail.

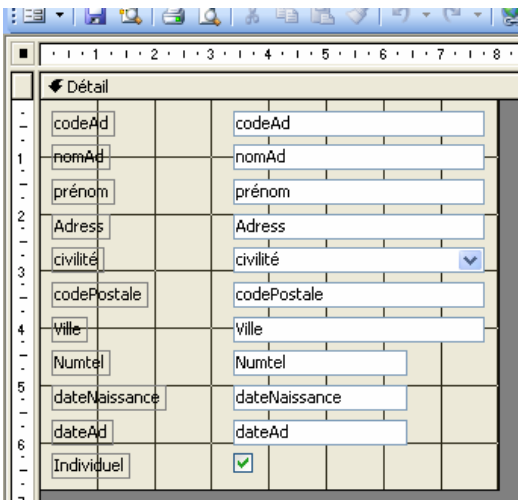


- Depuis la fenêtre Base de données, dans l'élément Tables, placez-vous sur **ADHÉRENTS** et créez un formulaire instantané comme d'habitude.
- Vous obtenez :
- Passez en mode Création

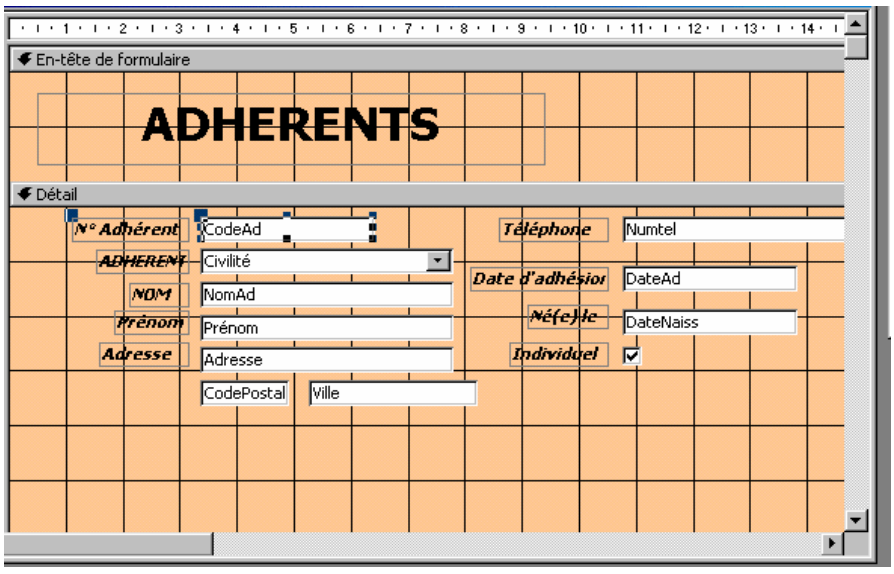


Vous allez travailler sur « l'envers » du décor. A tout moment, vous pouvez passer en mode Visualisation pour voir l'effet produit par vos manipulations.

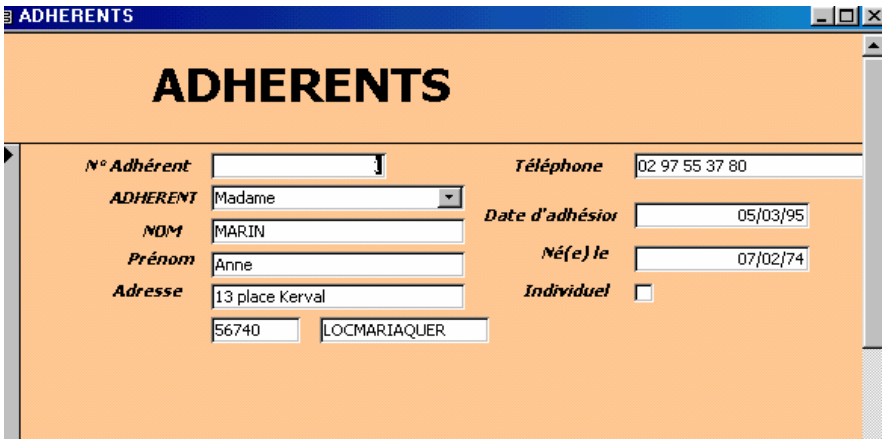
- Cliquez sur la zone de texte CodeAd : l'ensemble du contrôle est sélectionné.
- Déplacez-le vers la droite ; vous pouvez constater que l'ensemble du contrôle se déplace.
- Maintenant vous allez déplacer uniquement l'étiquette. Pour cela, cliquez précisément sur le petit carré noir en haut à gauche de l'étiquette. Le curseur prend la forme d'une main doigt levé.
- Rapprochez l'étiquette de la zone de texte. Observez bien de quelle manière apparaissent les marques de sélection dans les différents cas.

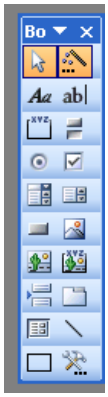


- Les étiquettes (ou intitulés) sont des emplacements où on peut écrire ce qu'on veut, alors que les zones de textes correspondent pour le système à des champs qui ont été définis dans la base de données. Ainsi vous ne pouvez changer le mot CodeAd de la zone de texte à fond blanc sans perturber le fonctionnement. En revanche, le mot CodeAd sur fond gris de l'étiquette peut être modifié.
- Sélectionnez le mot CodeAd ainsi
- Tapez à la place N° Adhérent
- Agrandissez la fenêtre au maximum
- Ecrivez Adhérent à la place de Civilité dans l'étiquette
- Ecrivez NOM à la place de NomAd
- Supprimez l'étiquette CodePostal et l'étiquette Ville
- Vous obtenez
- Rapprochez les étiquettes des zones de texte concernées
- Déplacez les contrôles pour arriver à une présentation ressemblant à celle-ci. Renommez les étiquettes Numtel en Téléphone, DateAd en Date d'adhésion, DateNaiss en Né(e) le.
- Cliquez Affichage En-tête/Pied de formulaire.
- Ouvrez la boîte à outils
- Cliquez l'outil Aa
- Créez une étiquette dans l'en-tête de formulaire (il faut la dessiner c'est-à-dire cliquer glisser depuis le coin gauche supérieur jusqu'au coin droit inférieur).
- Tapez ADHERENTS dans l'étiquette. Sélectionnez le cadre pour mettre l'étiquette en 24 gras.
- Cliquez avec le bouton droit sur le fond de l'en-tête de formulaire et dans l'option Couleur d'arrière-plan remplissage, choisissez une couleur orange clair. Faites la même chose pour la partie Détail du formulaire.
- Vous allez formater en une fois toutes les étiquettes. Pour les sélectionner, cliquez sur la première (N° Adhérent) puis appuyez sur la touche MAJ du clavier et maintenez-la enfoncée puis cliquez sur chacune des autres étiquettes une par une. Lorsque la sélection est faite, mettez les étiquettes en gras italique. Peut-être certaines d'entre elles seront trop étroites… il vous suffira de les agrandir légèrement comme n'importe quel objet de dessin.
- Vous allez améliorer l'alignement des zones de texte et des étiquettes. Sélectionnez les zones de texte (fond blanc) de la première partie, depuis CodeAd jusqu'à CodePostal. Cliquez avec le bouton droit dans cette sélection et choisissez Alignement puis Gauche.
- Faites la même chose pour les zones de texte de la deuxième partie.
- Alignez les étiquettes de la première partie sur la droite puis celle de la deuxième partie. Page suivante, voir résultat à obtenir.

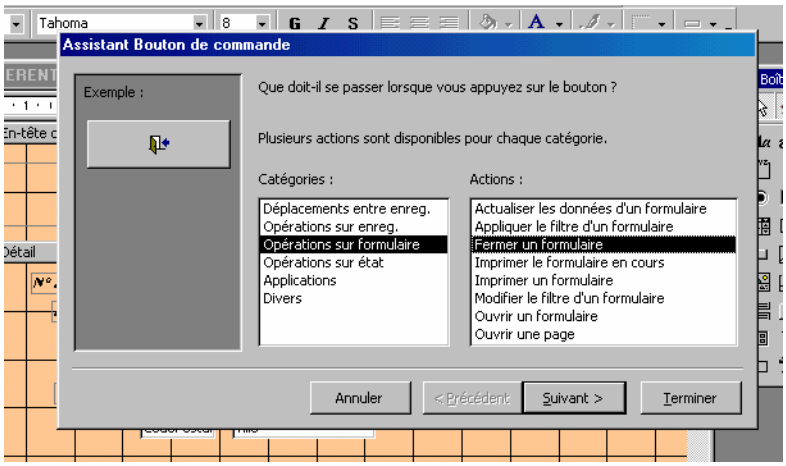


- Passez en mode Feuille de données

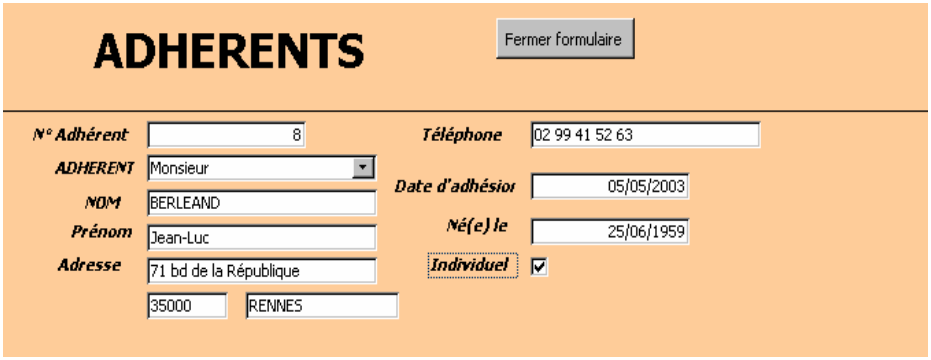




- Repassez en mode Création. Vous allez créer un bouton de commande pour fermer le formulaire. Assurez-vous que l'outil Assistant est enclenché.
- Dans la boîte à outils, (affichez-la si nécessaire) choisissez l'outil Bouton de commande.
- Dessinez un petit rectangle avec cet outil dans la zone en-tête du formulaire, à droite de l'étiquette ADHERENTS.
- Choisissez les paramètres suivants : Opérations sur formulaire et Fermer formulaire



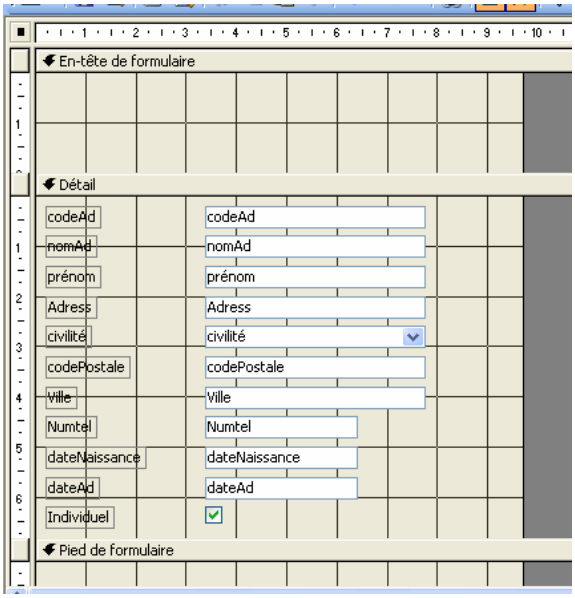
- Dans la boîte suivante, choisissez Texte Fermer Formulaire, Ok, puis donnez comme nom à votre bouton **Fermer Adhérents**.
- Passez en mode Affichage pour juger du résultat. Testez le fonctionnement de votre bouton.



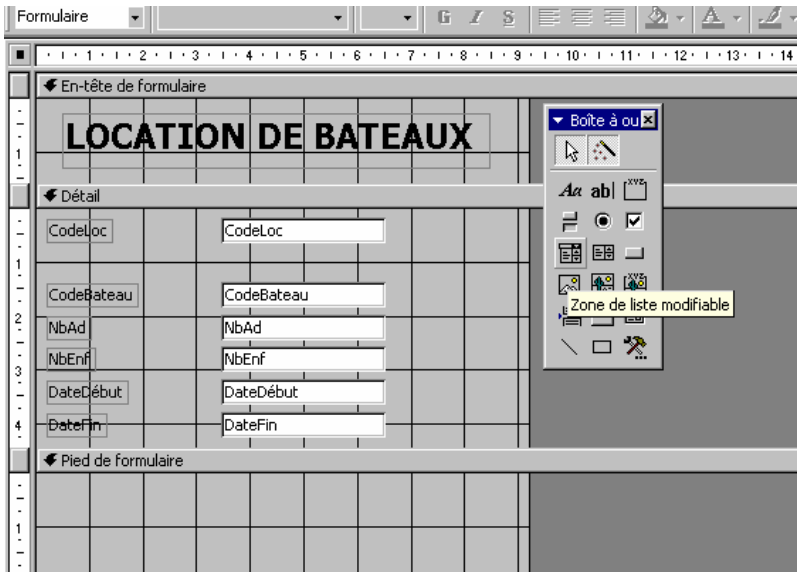
- Allez au dernier enregistrement pour inscrire un nouvel adhérent
- Enregistrez le nouvel adhérent : Mademoiselle Claire NANTREC, 7 impasse des Sternes, RENNES, téléphone 02 99 65 41 89, date de naissance 5/6/71, individuel.
- Fermez le formulaire.
- Ouvrez le formulaire **LOCATIONS** en mode Création.

- Affichez En-tête et pied de formulaire. Créez une étiquette LOCATION DE BATEAUX dans l'en-tête. Formatez-la en gras 18.

- En mode Affichage vous obtenez:



- Le problème c'est que pour enregistrer une location, il va falloir aller chercher le code de l'adhérent dans la table ADHÉRENTS et le code du bateau dans la table BATEAUX. Heureusement, vous allez pouvoir automatiser tout cela grâce aux outils assistés.
- La technique va consister à supprimer le champ et le remplacer par une liste déroulante, laquelle ira chercher les données dans une table. Repassez en Création.
- Cliquez sur le contrôle CodeAd et supprimez-le (touche Suppr).



- Dans la boîte à outils prenez l'outil Zone de liste modifiable (l'icône Assistant doit être enclenchée) et dessinez le nouveau contrôle à la même place que précédemment.
- Cliquez deux fois Suivant (vous voulez que le système aille chercher les données dans la table ADHÉRENTS).
- Dans le panneau suivant, prenez les trois premiers champs (CodeAd, NomAd et Prénom). Ce sont ceux qui s'afficheront pour que vous puissiez choisir l'adhérent. Cliquez Suivant.
- Dans le panneau suivant, gardez l'option Colonne clé cachée (*en fait, dans le formulaire, peu vous importe de connaître le numéro de l'adhérent, pourvu que la machine s'y retrouve*).
- Dans le panneau suivant, cochez l'option Stocker la valeur dans le champ et choisissez CodeAd. Ceci est une nécessité de la base de données pour qu'elle s'y retrouve. Suivant.
- Donnez comme nom à l'étiquette : Adhérent. Terminé.
- Passez en mode Affichage pour voir ce que ça fait.
- Revenez en mode Création pour créer la deuxième liste modifiable.
- Supprimez le contrôle CodeBateau. Avec l'outil zone de liste modifiable, dessinez-en un autre. Avec l'assistant, laissez-vous guider par les écrans comme précédemment (en choisissant la table **BATEAUX)**, gardez les trois colonnes, et n'oubliez pas de stocker la valeur dans le champ CodeBateau. Donnez le nom Bateau retenu à votre étiquette.
- Passez en mode Affichage pour voir.
- Repassez en Création, mettez votre formulaire dans une couleur de votre choix.
- Saisissez dans votre formulaire les locations suivantes :
- Monsieur LE GOFF André loue un EAU CLAIRE (bateau de type 2) pour une semaine à compter du 20/07. Il y aura 3 adultes et 5 enfants.
- Mademoiselle DURUEL Véronique loue un ESPADE 850 (type 1) pour 3 adultes pour deux semaines à compter du 09/09.

Monsieur BERLÉAND loue un CAT CAMP (type 3) pour la semaine prochaine pour 6 personnes

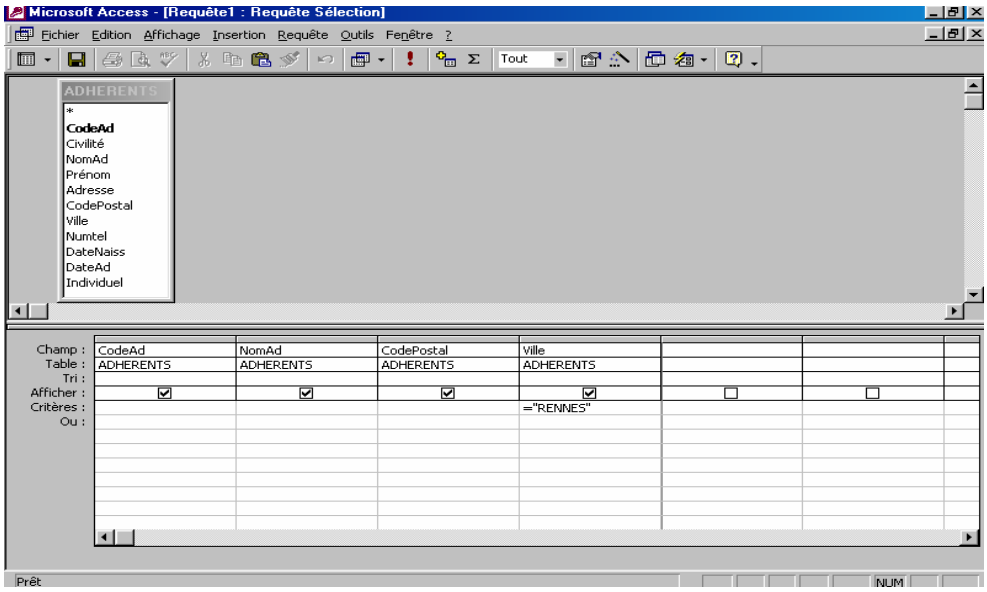
III Manipulation des donnée et création des requête

En matière de bases de données, la requête est quelque chose d'important car cela sert à beaucoup de choses.

C'est d'abord une question qu'on pose au système (exemple : quels sont les clients qui habitent RENNES ? ou Quels sont les bateaux retenus pour telle période ?). Le système fournit une réponse sous forme de liste. L'avantage de définir une requête (par rapport à une simple interrogation) est que vous pouvez l'enregistrer et vous en resservir plus tard. La réponse ultérieure du système tiendra compte, bien entendu, des mises à jour qui auraient eu lieu entretemps.
Avec une requête on peut aussi faire des calculs, des regroupements,etc.
Dans le cas qui nous intéresse, vous allez utiliser la requête pour regrouper des tables et ainsi créer un formulaire qui affiche les données venant de ces tables.

Requête sélection

- Dans la fenêtre Base de données, placez-vous dans l'onglet Requêtes, puis cliquez Créer une requête en mode Création.
- Double-cliquez **ADHÉRENTS** puis fermer.

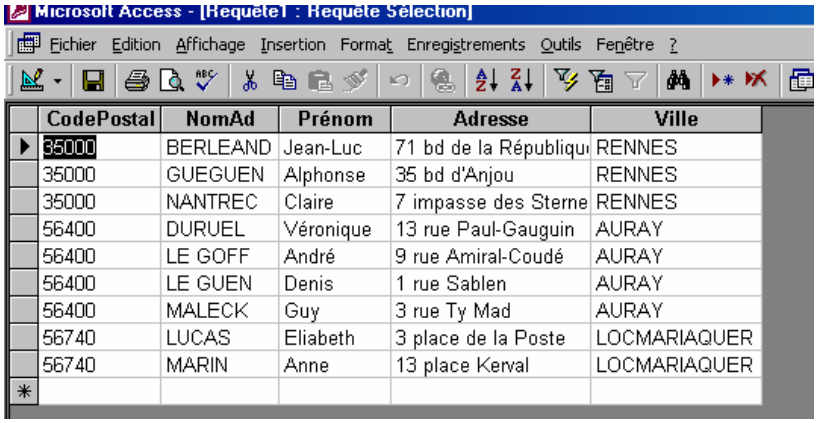


- Double-cliquez les champs suivants : CodeAd, NomAd, Prénom, Ville
- Vous obtenez
- Sur la ligne Critères du champ Ville saisissez RENNES. Vous obtenez :
- Exécutez la requête en appuyant sur l'icône point d'exclamation de la barre d'outils.
- Le résultat s'affiche : il y a 3 clients à RENNES..
- Refermez la requête ; donnez-lui le nom
- ADHÉRENTS RENNES.

Requête Tri

Vous voulez disposer à tout moment d'une liste des clients triée par codes postaux puis alphabétiquement à l'intérieur de ce classement.

- Requêtes, Création d'une requête en mode Création
- Ajoutez la table **ADHÉRENTS.** Fermez**.**
- Ajoutez les champs CodePostal, NomAd, Prénom, Adresse, Ville (dans cet ordre-là)
- Dans la ligne Tri des champs CodePostal et NomAd, choisissez tri Croissant. Un premier tri va s'effectuer sur le code postal, premier champ rencontré par le système puis un deuxième tri sur le champ Nom.
- Exécutez la requête puis refermez-la en lui donnant le nom Liste des clients par localités.



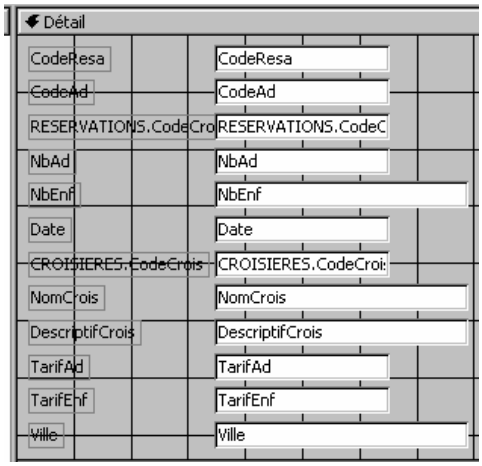
CodePostal	NomAd	Prénom	Adresse	Ville
35000	BERLEAND	Jean-Luc	71 bd de la Républiqu	RENNES
35000	GUEGUEN	Alphonse	35 bd d'Anjou	RENNES
35000	NANTREC	Claire	7 impasse des Sterne	RENNES
56400	DURUEL	Véronique	13 rue Paul-Gauguin	AURAY
56400	LE GOFF	André	9 rue Amiral-Coudé	AURAY
56400	LE GUEN	Denis	1 rue Sablen	AURAY
56400	MALECK	Guy	3 rue Ty Mad	AURAY
56740	LUCAS	Eliabeth	3 place de la Poste	LOCMARIAQUER
56740	MARIN	Anne	13 place Kerval	LOCMARIAQUER

Requête regroupement de tables

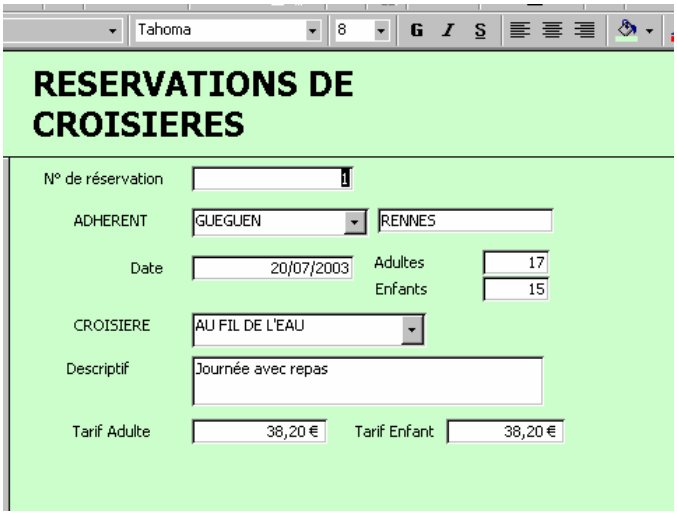
Cette requête a pour but de créer un formulaire regroupant les données de plusieurs tables.

- Créez une nouvelle requête en mode Création.
- Ajoutez les tables **RÉSERVATIONS, CROISIÈRES** et **ADHÉRENTS.** Fermez**.**
- Ajoutez tous les champs de **RÉSERVATIONS** et **CROISIÈRES** et le champ VILLE de la table **ADHERENTS**

- Pour l'instant, inutile d'exécuter la requête, car il n'y a pas de données saisies dans les tables. Refermez-la et donnez-lui comme nom **Pour formulaire réservations**.
- Dans la fenêtre Base de données, placez-vous sur la requête que vous venez de créer et cliquez sur Formulaire instantané de façon à créer votre formulaire automatiquement en se basant sur les tables regroupées dans la requête.
- Supprimez le contrôle RESERVATIONS.CodeC



- Comme vous l'avez fait dans le formulaire LOCATIONS, vous allez remplacer CodeAd par une liste déroulante qui affichera les noms en clair. Supprimez le contrôle puis dessinez avec la boîte à outils une zone de liste modifiable puis laissez-vous guider par les écrans en choisissant la talbe ADHÉRENTS et en demandant l'affichage des champs CodeAd, NomAd, Prénom. N'oubliez pas de stocker la valeur dans le champ CodeAd.
- Remplacez aussi le contrôle CROISIERES.CodeCrois par une liste modifiable (choisissez la talbe **CROISIÈRES** et demandez l'affichage des 2 premiers champs) et stockez la valeur dans le champ CodCrois. Supprimez le contrôle NomCrois car cela ferait double emploi.
- Passez en mode Affichage et remplissez votre formulaire pour tester son fonctionnement avec cette réservation : Monsieur GUEGUEN Alphonse réserve une croisière AU FIL DE L'EAU pour le 20/07 pour un groupe de 32 personnes (17 adultes et 15 enfants)
- Constatez que lorsque vous choisissez une croisière dans la liste déroulante, ses caractéristiques se reportent automatiquement dans les autres champs. Ceci se fait en fonction de tout ce que vous avez déterminé dans le modèle de base. De même, la ville où habite l'adhérent se reporte automatiquement.
- Améliorez la présentation du formulaire en le mettant en couleur,en alignant mieux les différentes étiquettes et zones de texte. Insérer un en-tête de formulaire avec une étiquette de titre **RÉSERVATIONS DE CROISIÈRES**, comme ci-dessous.

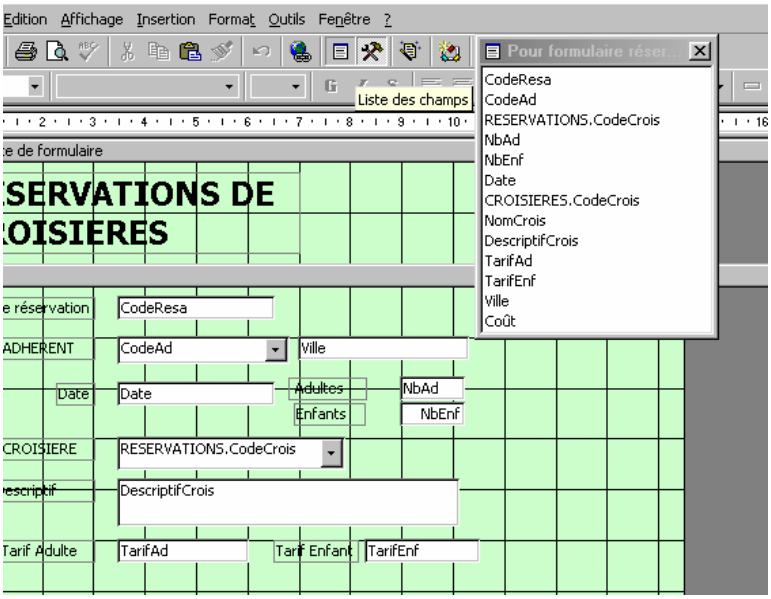


- Le formulaire commence à prendre tournure.Toutefois vous pensez peut-être avec raison qu'il serait intéressant de pouvoir calculer directement le coût de cette réservation pour le client. Le nombre d'adultes et d'enfants est connu, de même que les tarifs. Il faudrait pouvoir créer un champ qui calcule. C'est possible ! Mais rappelez-vous, le formulaire est basé sur une requête. C'est dans celle-ci qu'il faut préparer le champ calculé ; ensuite il suffira d'insérer ce nouveau champ dans le formulaire.

- Fermez le formulaire, ouvrez la requête Pour formulaire RÉSERVATIONS en mode Création.
- Placez-vous sur le premier champ libre (tout-à-fait à droite de tous les autres) et saisissez très exactement l'expression de calcul suivante *(attention, toute erreur de crochet ou de majuscules provoquera un dysfonctionnement)*. N'oubliez pas le signe deux-points après Coût. Rappel : le crochet s'obtient en actionnant la touche AltGr et la touche 5 ou °.

Coût:[NbAd]*[TarifAd]+[NbEnf]*[TarifEnf]

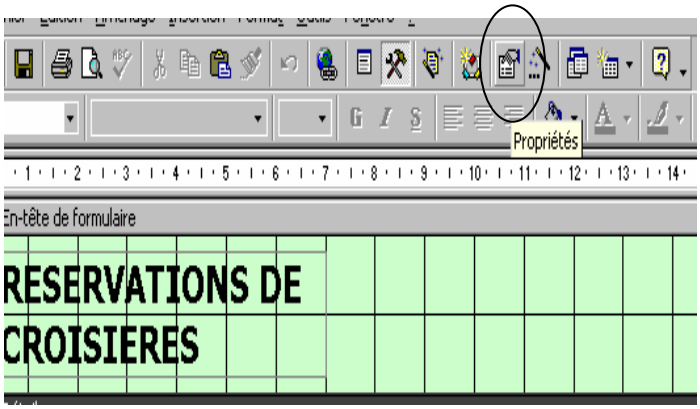
- Exécutez la requête pour voir si le calcul se fait bien.
- Il reste à insérer le champ calculé dans votre formulaire.
- Fermez la requête en l'enregistrant. Ouvrez le formulaire **RÉSERVATIONS** en mode Création.
- Cliquez l'icône Liste des champs. Celle-ci a l'avantage de se tenir toujours à jour des modifications que l'on pourrait apporter au support de notre formulaire.



- En effet, dans la liste qui apparaît vous voyez le champ Coût que vous venez de créer. Prenez-le avec la souris et placez-le dans le formulaire.
- Passez en mode Affichage pour voir.
- Enregistrez les deux réservations suivantes dans votre formulaire.

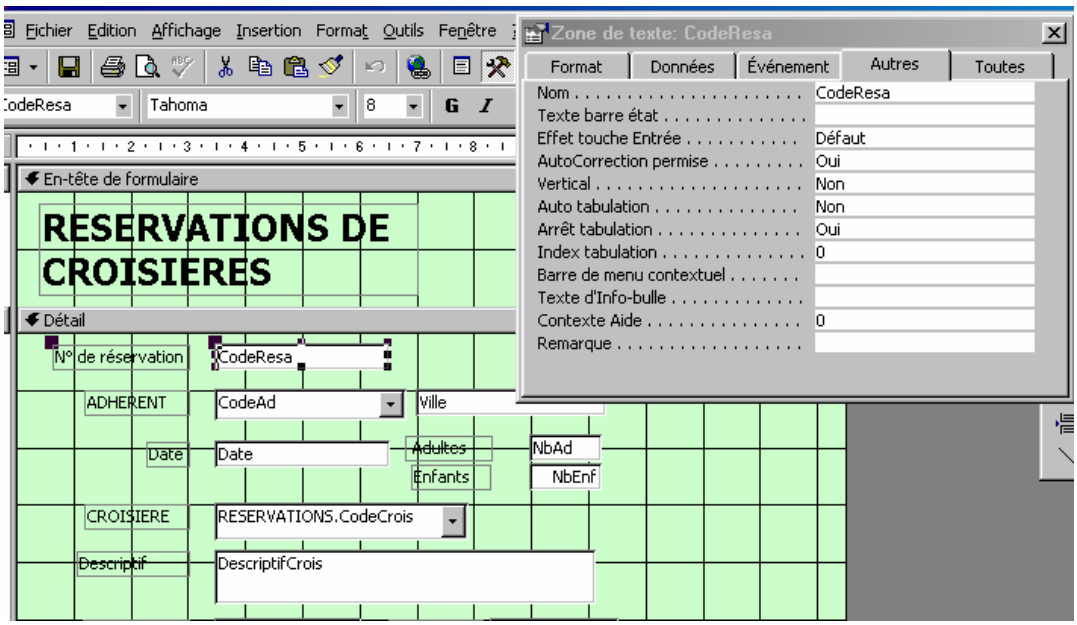
Monsieur MALECK Guy réserve une croisière DETENTE pour le 23/06 pour 40 adultes et 11 enfants entre 3 et 12 ans	Madame LUCAS réserve une croisière EVASION pour le 7 juillet pour 52 personnes (35 adultes et 17 enfants)

Propriétés des champs



Vous pouvez constater que dès que vous positionnez le nom de la croisière, le reste s'affiche de lui-même. Toutefois vous avez remarqué que la touche tabulation qui vous permet de passer d'un champ à l'autre passe un peu du coq à l'âne car les champs ont été déplacés ou rajoutés et cela ne suit pas l'ordre qui vous faciliterait la saisie. Il y a heureusement un remède.

- En mode Création, cliquez l'icône Propriétés de la barre d'outils.
- Cette fenêtre vous montre toutes les propriétés de tous les éléments de votre écran. Ici vous êtes dans le formulaire Réservations. Cliquez sur le champ CodeResa et dans la fenêtre des propriétés, prenez l'onglet Autres. Regardez la ligne Index Tabulation : il y 0, ce qui veut dire que c'est la première zone à remplir.



- Sans fermer la fenêtre des propriétés, cliquez sur CodeAd ; mettez 1 à la place de l'index tab. Continuez de la même manière pour tous les autres champs, jusqu'au Coût qui sera le 10e. Dorénavant, la touche tabulation vous emmènera du début à la fin selon votre paramètrage !

Un nouvel adhérent s'inscrit (le 3 juin) :

Jacques FERRUGIA 84 route de Vannes 56740 LOCMARIACQUER Tél. 02 97 45 78 12 Né le 7/3/51 Individuel
Loue un CAT CAMP pour 3 semaines pour 4 personnes à compter du 10 août

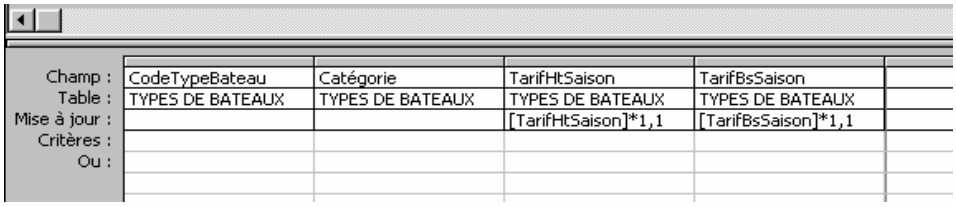
- Vous allez l'enregistrer dans le formulaire **ADHÉRENTS** ; toutefois vous avez remarqué que vous êtes obligé de taper en majuscules le nom de famille ; il serait préférable de disposer d'un système où la saisie serait faite en majuscules même si on tape en minuscules. Il y a une solution !
- Ouvrez le formulaire en mode Création. Placez-vous sur le champ NomAd, ouvrez la fenêtre des propriétés, onglet Format. Sur la ligne Format, saisissez simplement le caractère > (*il forcera l'affichage en majuscules dans ce champ*).
- Repassez en mode Affichage pour saisir votre nouvel adhérent et constatez que votre manipulation a porté ses fruits.
- Enregistrez la location dans le formulaire **LOCATIONS**.

Requête Mise à jour

Malheureusement, les prix des locations de bateaux augmentent de 10 %. Vous allez devoir changer les tarifs de la table **TYPES DE BATEAUX**. *La mauvaise solution serait de le faire manuellement ; bien entendu le cas de notre exemple est tellement petit que ce ne serait pas gênant de faire les opérations à la main. Mais il faut toujours penser « gros volumes » avec une base de données*.

Vous allez donc créer une requête qui procédera automatiquement à l'augmentation des prix.

- Dans l'élément Requêtes de la fenêtre Base de données, créez une requête en mode Création.
- Ajoutez la table **TYPES DE BATEAUX**, affichez tous les champs sauf les nombres de personnes.
- Cliquez sur le menu Requête et choisissez Requête Mise à jour, ce qui aura pour effet d'ajouter une ligne Mise à jour dans vos champs.
- Saisissez comme ci-dessous les expressions de calcul dans les champs des tarifs (attention, pas d'erreur de saisie, sans quoi cela ne marchera pas !)



	CodeTypeBateau	Catégorie	TarifHtSaison	TarifBsSaison
Champ :				
Table :	TYPES DE BATEAUX	TYPES DE BATEAUX	TYPES DE BATEAUX	TYPES DE BATEAUX
Mise à jour :			[TarifHtSaison]*1,1	[TarifBsSaison]*1,1
Critères :				
Ou :				

- Exécutez la requête. Attention, ne faites la manœuvre qu'une seule fois ! Si vous la lancez plusieurs fois, les prix seront augmentés plusieurs fois de 10 %, et à chaque fois… sur la base déjà augmentée… Il faudra alors calculer le coefficient capable de ramener les choses à leur état initial. Donc vous l'exécutez une seule fois et vous cliquez sur Feuille de données pour voir le résultat.

Catégorie	NomBateau	NomAd	Prénom	Ville	NbAd	NbEnf	Nb To	DateDébut	DateFin	Durée
CAT CAMP	MALOUIN	BERLEAND	Jean-Luc	RENNES	6	0	6	15/06/2003	22/06/2003	7
CAT CAMP	REDOUTABLE	FERRUGIA	Jacques	LOCMARIAC	2	2	4	10/08/2003	29/08/2003	19
EAU CLAIRE 930	CORSAIRE	LE GOFF	André	AURAY	3	5	8	20/07/2003	27/07/2003	7
ESPADE 850 HORS	DIVINE	DURUEL	Véronique	AURAY	3	0	3	07/09/2003	21/09/2003	14

Requête Somme/Regroupement

Vous aimeriez disposer d'un moyen permanent de connaître ce qu'a rapporté globalement chaque type de croisière

- Créez une requête en mode Création, ajoutez la requête Pour formulaire **RÉSERVATIONS**, affichez les champs Coût et CROISIERES.CodeCrois.
- Cliquez l'icône Σ de la barre d'outils qui aura pour effet d'ajouter une ligne **Regroupement** dans vos champs.
- Dans cette ligne pour le champ Coût, déroulez la liste et choisissez Somme. Pour le deuxième champ, laissez Regroupement.
- Exécutez la requête. Fermez la requête et enregistrez-la sous le nom Chiffre d'affaires croisières. Vous pourrez à tout moment lancer cette requête et connaître ainsi votre chiffre d'affaires croisières.

Requête sélection (autre genre)

Vous souhaiteriez disposer à moments réguliers de la liste des clients ayant loué un bateau et de celle des clients ayant réservé une croisière. Cela pourra être édité sous forme d'état, ce que vous allez étudier dans le chapitre suivant. Mais il faut déjà disposer de la requête qui fait l'extraction de données.

- Créez une requête en mode Création, ajoutez les tables **ADHÉRENTS**, **RÉSERVATIONS**, **CROISIÈRES**.
- Affichez les champs :Nom, Prénom, Ville, Individuel de la table **ADHÉRENTS**, NbAd et NbEnf de la table **RÉSERVATIONS**, NomCrois de la table **CROISIERES**.
- Pour connaître le nombre total de personnes de chaque croisière, créez un champ calculé sur le premier champ vierge, ainsi paramétré :

Nb Total :[NbAd]+[NbEnf]

- Exécutez la requête.

Vous voudriez que la liste soit triée par ordre alphabétique des clients. Que pouvez-vous modifier dans la requête pour l'obtenir ?

- Fermez et enregistrez la requête sous le nom **CROISIÈRES PAR CLIENT**.

IV Langage SQL

INTRODUCTION

Qu'appelle t'on SQL?

SQL (*Structured Query Language*, traduisez *Langage de requêtes structuré*) est un langage de définition de données (LDD, ou en anglais DDL *Data Definition Language*), un langage de manipulation de données (LMD, ou en anglais DML, *Data Manipulation Language*), et un langage de contrôle de données (LCD, ou en anglais DCL, *Data Control Language*), pour les *bases de données relationnelles*.
Le modèle relationnel a été inventé par E.F. Codd (Directeur de recherche du centre IBM de San José) en 1970, suite à quoi de nombreux langages ont fait leur apparition:

- IBM Sequel (Structured English Query Language) en 1977
- IBM Sequel/2
- IBM System/R
- IBM DB2

Ce sont ces langages qui ont donné naissance au standard SQL, normalisé en 1986 par l'ANSI pour donner SQL/86. Puis en 1989 la version SQL/89 a été approuvée. La norme SQL/92 a désormais pour nom SQL 2.

SQL est un langage de définition de données

SQL est un langage de définition de données (**LDD**), c'est-à-dire qu'il permet de créer des tables dans une base de données relationnelle, ainsi que d'en modifier ou en supprimer.

SQL est un langage de manipulation de données

SQL est un langage de manipulation de données (**LMD**), cela signifie qu'il permet de sélectionner, insérer, modifier ou supprimer des données dans une table d'une base de données relationnelle.

SQL est un langage de protections d'accès

Il est possible avec SQL de définir des permissions au niveau des utilisateurs d'une base de données. On parle de **DCL** (Data Control Language).

Typologie du langage

Il est possible d'inclure des requêtes SQL dans un programme écrit dans un autre langage (en langage C par exemple), ainsi que d'envoyer directement les requêtes SQL telles quelles au SGBD.
Il est possible d'ajouter des commentaires grâce:

- au caractère `%`. Tous les caractères situés après celui-ci sur la même ligne ne seront pas interprétés
- aux délimiteurs `/*` et `*/`. Tous les caractères compris entre les délimiteurs sont considérés comme des commentaires

Le langage SQL n'est pas sensible à la casse (en anglais *case sensitive*), cela signifie que l'on peut aussi bien écrire les instructions en minuscules qu'en majuscule. Toutefois, cette insensibilité à la casse n'est que partielle dans la mesure où la différenciation entre minuscules et majuscules existe au niveau des identificateurs d'objets.

SELECTION DES DONNEES

Le langage de manipulation de données

Le SQL est à la fois un langage de manipulation de données et un langage de définition de données. Toutefois, la définition de données est l'oeuvre de l'administrateur de la base de données, c'est pourquoi la plupart des personnes qui utilisent le langage SQL ne se servent que du langage de manipulation de données, permettant de sélectionner les données qui les intéressent.

La principale commande du *langage de manipulation de données* est la commande *SELECT*.

Syntaxe de la commande SELECT

La commande *SELECT* est basée sur l'algèbre relationnelle, en effectuant des opérations de sélection de données sur plusieurs tables relationnelles par projection. Sa syntaxe est la suivante:

SELECT [ALL] | [DISTINCT] <liste des noms de colonnes> | *
FROM <Liste des tables>
[WHERE <condition logique>]
Il existe d'autres options pour la commande *SELECT*:

GROUP BY
HAVING
ORDER BY

- L'option **ALL** est, par opposition à l'option *DISTINCT*, l'option par défaut. Elle permet de sélectionner l'ensemble des lignes satisfaisant à la condition logique
- L'option **DISTINCT** permet de ne conserver que des lignes distinctes, en éliminant les doublons
- La **liste des noms de colonnes** indique la liste des colonnes choisies, séparées par des virgules. Lorsque l'on désire sélectionner l'ensemble des colonnes d'une table il n'est pas nécessaire de saisir la liste de ses colonnes, l'option ***** permet de réaliser cette tâche
- La **liste des tables** indique l'ensemble des tables (séparées par des virgules) sur lesquelles on opère
- La **condition logique** permet d'exprimer des qualifications complexes à l'aide d'opérateurs logiques et de comparateurs arithmétiques

III- EXPRESSION DES PROJECTIONS

Expression d'une projection

Une projection est une instruction permettant de sélectionner un ensemble de colonnes dans une table. Soit la table *VOITURE* suivante:

TABLE VOITURE

Marque	Modèle	série	Numéro
Renault	18	RL	469sj45
Renault	Kango	RL	4568hd16
Renault	kango	RL	6576VE38
Peugeot	106	KID	7845ZS83
Peugeot	309	Chorus	5647ABY82
Ford	Escort	Match	8562EV23

La sélection de toutes les colonnes de la table se fait par l'instruction:

SELECT * FROM VOITURE

Résultat	Marque	Modèle	série	Numéro
	Renault	18	RL	469sj45
	Renault	Kango	RL	4568hd16
	Renault	kango	RL	6576VE38
	Peugeot	106	KID	7845ZS83
	Peugeot	309	Chorus	5647ABY82
	Ford	Escort	Match	8562EV23

La sélection des colonnes *Modèle* et *Série* de la table se fait par l'instruction:

SELECT Modèle, Série FROM VOITURE

Résultat	Modèle	série
	18	RL
	Kango	RL
	kango	RL
	106	KID
	309	Chorus
	Escort	Match

La sélection des colonnes *Modèle* et *Série* en éliminant les doublons se fait par l'instruction:

SELECT DISTINCT Modèle, Série FROM VOITURE

Résultat	Modèle	série
	18	RL
	kango	RL
	106	KID

309	Chorus
Escort	Match

L'EXPRESSION DES RESTRICTIONS

Une restriction consiste à sélectionner les lignes satisfaisant à une condition logique effectuée sur leurs attributs.

En SQL, les restrictions s'expriment à l'aide de la clause *WHERE* suivie d'une condition logique exprimée à l'aide d'opérateurs logiques

AND
OR
NOT

De comparateurs de chaîne:

IN
BETWEEN
LIKE

D'opérateurs arithmétiques:

+, -, *, /, %, &, |, ^, ~

Et de comparateurs arithmétiques:

=, !=, >, <, >=, <=, <>, !>, !<

Restrictions simples

Soit la table suivante, présentant des voitures d'occasion:

TABLE OCCAZ

Marque	Modèle	série	Numéro	Compteur
Renault	18	RL	469sj45	123450
Renault	Kango	RL	4568hd16	56000
Renault	kango	RL	6576VE38	12000
Peugeot	106	KID	7845ZS83	75600
Peugeot	309	Chorus	5647ABY82	189500
Ford	Escort	Match	8562EV23	

Le champ présentant la valeur du kilométrage au compteur de la *Ford Escort* est délibérément non renseigné.

La sélection de toutes les voitures d'occasion ayant un kilométrage inférieur à 100 000 Km se fait par l'instruction:

```
SELECT * FROM OCCAZ
WHERE (Compteur < 100000)
```

Résultat	Marque	Modèle	série	Numéro	Compteur
	Renault	Kango	RL	4568hd16	56000
	Renault	kango	RL	6576VE38	12000
	Peugeot	106	KID	7845ZS83	75600

La sélection des colonnes *Marque* et *Compteur* des voitures ayant un kilométrage inférieur à 100 000 Km se fait par l'instruction:

```
SELECT Marque, Compteur FROM OCCAZ
WHERE (Compteur < 100000)
```

Résultat	Marque	Compteur
	Renault	56000
	Renault	12000
	Peugeot	75600

La sélection de toutes les voitures d'occasion ayant un kilométrage inférieur ou égal à 100 000 Km, mais supérieur ou égal à 30000Km, se fait par l'instruction:

```
SELECT * FROM OCCAZ
WHERE (Compteur <= 100000) AND (Compteur >= 30000)
```

Résultat	Marque	Modèle	série	Numéro	Compteur
	Renault	Kango	RL	4568hd16	56000
	Peugeot	106	KID	7845ZS83	75600

Restriction sur une comparaison de chaîne

Le prédicat *LIKE* permet de faire des comparaisons sur des chaînes grâce à des caractères, appelés caractères *jokers*:

Le caractère **%** permet de remplacer une séquence de caractères (éventuellement nulle)
Le caractère **_** permet de remplacer un caractère (l'équivalent du "blanc" au scrabble...)
Les caractères **[-]** permettent de définir un intervalle de caractères (par exemple [J-M])
La sélection des voitures dont la marque a un E en deuxième position se fait par l'instruction:

```
SELECT * FROM OCCAZ
WHERE Marque LIKE _E%
```

Résultat	Marque	Modèle	série	Numéro	Compteur
	Renault	18	RL	469sj45	123450
	Renault	Kango	RL	4568hd16	56000
	Renault	kango	RL	6576VE38	12000
	Peugeot	106	KID	7845ZS83	75600
	Peugeot	309	Chorus	5647ABY82	189500

Restriction sur un ensemble

Les prédicats *BETWEEN* et *IN* permettent de vérifier respectivement qu'une valeur se trouve dans un intervalle ou qu'une valeur appartient à une liste de valeurs:

La sélection de toutes les voitures d'occasion ayant un kilométrage inférieur ou égal à 100 000 Km, mais supérieur ou égal à 30000Km, (effectuée plus haut avec des comparateurs arithmétiques) peut se faire par l'instruction:

```
SELECT * FROM OCCAZ
WHERE Compteur BETWEEN 100000 AND 30000
```

Résultat	Marque	Modèle	série	Numéro	Compteur
	Renault	Kango	RL	4568hd16	56000
	Peugeot	106	KID	7845ZS83	75600

La sélection des voitures d'occasion dont la marque est Peugeot ou Ford se fait grâce à l'instruction:

```
SELECT * FROM OCCAZ
WHERE Marque IN (Peugeot, Ford)
```

Résultat	Marque	Modèle	série	Numéro	Compteur
	Peugeot	106	KID	7845ZS83	75600
	Peugeot	309	Chorus	5647ABY82	189500
	Ford	Escort	Match	8562EV23	

Restriction sur les valeurs manquantes

Lorsqu'un champ n'est pas renseigné, le SGBD lui attribue une valeur spéciale que l'on note *NULL*. La recherche de cette valeur ne peut pas se faire à l'aide des opérateurs standard, il faut utiliser les prédicats *IS NULL* ou bien *IS NOT NULL*.

La sélection de toutes les voitures d'occasion dont le kilométrage n'est pas renseigné se fait par l'instruction:

```
SELECT * FROM OCCAZ
WHERE Compteur IS NULL
```

Résultat	Marque	Modèle	série	Numéro	Compteur

	Ford	Escort	Match	8562EV23	

TRI DES RESULTATS

Il est possible en SQL d'organiser les tuples fournis en résultat grâce à la clause *ORDER BY*. La clause *ORDER BY* est suivie des mots clés *ASC* ou *DESC*, qui précisent respectivement si le tri se fait de manière croissante (par défaut) ou décroissante. Le classement se fait sur des nombres ou des chaînes de caractères.

Prenons l'exemple de la table voiture:

La sélection de toutes les colonnes de la table triées par ordre croissant de l'attribut *Marque* se fait par l'instruction:

```
SELECT * FROM VOITURE
ORDER BY Marque ASC
```

Résultat	Marque	Modèle	série	Numéro	Compteur
	Ford	Escort	Match	8562EV23	
	Peugeot	106	KID	7845ZS83	75600
	Peugeot	309	Chorus	5647ABY82	189500
	Renault	18	RL	469sj45	123450
	Renault	Kango	RL	4568hd16	56000
	Renault	kango	RL	6576VE38	12000

La sélection de toutes les colonnes de la table triées par ordre croissant de l'attribut *Marque*, puis par ordre décroissant du compteur, se fait par l'instruction:

```
SELECT * FROM VOITURE
ORDER BY Marque ASC, Compteur DESC
```

Résultat	Marque	Modèle	série	Numéro	Compteur
	Ford	Escort	Match	8562EV23	
	Peugeot	106	KID	7845ZS83	75600
	Peugeot	309	Chorus	5647ABY82	189500
	Renault	18	RL	469sj45	123450
	Renault	Kango	RL	4568hd16	56000
	Renault	kango	RL	6576VE38	12000

Regroupement de résultats

Il peut être intéressant de regrouper des résultats afin de faire des opérations par groupe (opérations statistiques par exemple). Cette opération se réalise à l'aide de la clause *GROUP BY*, suivie du nom de chaque colonne sur laquelle on veut effectuer des regroupements.

Les principales fonctions pouvant être effectuées par groupe sont:

AVG: Calcule la moyenne d'une colonne (ou de chaque regroupement si elle est couplée à la clause *GROUP BY*)
COUNT: Calcule le nombre de lignes d'une table (ou de chaque regroupement ...)
MAX: Calcule la valeur maximale d'une colonne (ou de chaque regroupement ...)
MIN: Calcule la valeur minimale colonne (ou de chaque regroupement ...)
SUM: Effectue la somme des valeurs d'une colonne (ou de chaque regroupement ...)

Soit la table VOITURE ci-dessus:

L'affichage des moyennes des compteurs par marque se fait par l'instruction:

```
SELECT AVG(Compteur) FROM VOITURE
GROUP BY Marque
```

Résultat	Marque	Modèle	série	Numéro	Compteur
	Renault	kango	RL	4568hd16	70350
	Peugeot	309	Chorus	5647ABY82	63816.6
	Ford	Escort	Match	8562EV23	

La clause *HAVING* va de pair avec la clause *GROUP BY*, elle permet d'appliquer une restriction sur les groupes créés grâce à la clause *GROUP BY*.

L'affichage des moyennes des compteurs non nulles regroupées par marque se fait par l'instruction:

SELECT AVG(Compteur) FROM VOITURE
GROUP BY Marque
HAVING Compteur IS NOT NULL

Résultat	Marque	Modèle	série	Numéro	Compteur
	Peugeot Ford	309 Escort	Chorus Match	5647ABY82 8562EV23	70350 63816.6

LES JOINTURES

Expression des jointures

Une jointure (ou θ-jointure) est un produit cartésien de deux tables. On appelle équijointure une θ-jointure dont la qualification est une égalité entre deux colonnes.
En SQL, l'expression d'une jointure se fait en précisant le nom des colonnes des tables sur lesquelles on fait la jointure, en désignant les colonnes des différentes tables en écrivant le nom de la table, suivie d'un point puis du nom de la colonne. La clause *WHERE* permet de préciser la qualification de la jointure.

Soit les deux tables suivantes:

TABLE OCCAZ

Marque	Modèle	série	Numéro	Compteur
Renault	18	RL	469sj45	123450
Renault	Kango	RL	4568hd16	56000
Renault	kango	RL	6576VE38	12000
Peugeot	106	KID	7845ZS83	75600
Peugeot	309	Chorus	5647ABY82	189500
Ford	Escort	Match	8562EV23	

TABLE SOCIETE

NOM	PAYS
Renault	France
Peugeot	France
Ford	US

L'affichage des pays d'origine des voitures par marque/modèle se fait par l'instruction:

SELECT Occaz.Marque, Occaz.Modele, Societe.Nom, Societe.Pays FROM OCCAZ,SOCIETE
WHERE Occaz.Marque = Societe.Nom

Résultat	Marque	Modèle	Pays
	Renault	18	France
	Renault	Kango	France
	Renault	kango	France
	Peugeot	106	France
	Peugeot	309	France
	Ford	Escort	US

LES SOUS-REQUETES

Expression des sous requêtes

Effectuer une sous requête consiste à effectuer une requête à l'intérieur d'une autre, ou en d'autres termes d'utiliser une requête afin d'en réaliser une autre.

Une sous-requête doit être placée à la suite d'une clause *WHERE* ou *HAVING*, et doit remplacer une constante ou un groupe de constantes qui permettraient en temps normal d'exprimer la qualification.

Lorsque la sous-requête remplace une constante utilisée avec des opérateurs classique, elle doit obligatoirement renvoyer une seule réponse (une table d'une ligne et une colonne). Par exemple:

SELECT ---- FROM ----
WHERE ---- < (SELECT ---- FROM ----)

Lorsque la sous-requête remplace une constante utilisée dans une expression mettant en jeu les opérateurs *IN*, *EXISTS*, *ALL* ou *ANY*, elle doit obligatoirement renvoyer une seule ligne.

SELECT ---- FROM ----
WHERE ---- IN (SELECT ---- FROM ----)

Revenons a la table voiture:

La sélection des voitures dont le compteur est inférieur à la moyenne se fait par l'instruction:

SELECT * FROM OCCAZ
WHERE Compteur < (SELECT AVG(Compteur) FROM OCCAZ)

Résultat	Marque	Modèle	série	Numéro	Compteur
	Renault	Kango	RL	4568hd16	56000
	Renault	Kango	RL	6576VE38	12000
	Peugeot	106	KID	7845ZS83	75600

OPERATIONS ENSEMBLISTES

Les opérations ensemblistes

Les opérations ensemblistes en SQL, sont ceux définis dans l'algèbre relationnelle. Elles sont réalisées grâce aux opérateurs:

UNION
INTERSECT (ne fait pas partie de la norme SQL et n'est donc pas implémenté dans tous les SGBD)
EXCEPT (ne fait pas partie de la norme SQL et n'est donc pas implémenté dans tous les SGBD)

Ces opérateurs s'utilisent entre deux clauses *SELECT*.

L'opérateur *UNION*

Cet opérateur permet d'effectuer une UNION des tuples sélectionnés par deux clauses *SELECT* (les deux tables sur lesquelles on travaille devant avoir le même schéma).

SELECT ---- FROM ---- WHERE ------
UNION
SELECT ---- FROM ---- WHERE ------

Par défaut les doublons sont automatiquement éliminés. Pour conserver les doublons, il est possible d'utiliser une clause *UNION ALL*.

L'opérateur *INTERSECT*

Cet opérateur permet d'effectuer une INTERSECTION des tuples sélectionnés par deux clauses *SELECT* (les deux tables sur lesquelles on travaille devant avoir le même schéma).

SELECT ---- FROM ---- WHERE ------
INTERSECT
SELECT ---- FROM ---- WHERE ------

L'opérateur *INTERSECT* n'étant pas implémenté dans tous les SGBD, il est possible de le remplacer par des commandes usuelles:

```
SELECT a,b FROM table1
WHERE EXISTS (      SELECT c,d FROM table2
                WHERE a=c AND b=d )
```

L'opérateur *EXCEPT*

Cet opérateur permet d'effectuer une DIFFERENCE entre les tuples sélectionnés par deux clauses *SELECT*, c'est-à-dire sélectionner les tuples de la seconde table n'appartenant pas à la première (les deux tables devant avoir le même schéma).

SELECT a,b FROM table1 WHERE ------
INTERSECT
SELECT c,d FROM table2 WHERE ------

L'opérateur *EXCEPT* n'étant pas implémenté dans tous les SGBD, il est possible de le remplacer par des commandes usuelles:

```
SELECT a,b FROM table1
WHERE NOT EXISTS (  SELECT c,d FROM table2
                WHERE a=c AND b=d)
```

MODIFICATION DES DONNEES

Le SQL permet la modification d'une table par un utilisateur (pour peu qu'il ait les droits suffisants...). La modification d'une table consiste à:

Ajouter des tuples
Modifier des tuples existants
Ou bien supprimer des tuples

Insertion de données

L'insertion de nouvelles données dans une table se fait grâce à l'ordre INSERT, qui permet d'insérer de nouvelles lignes dans la table.

L'ordre INSERT attend la clause *INTO*, suivie du nom de la table, ainsi que du nom de chacune des colonnes entre parenthèses (les colonnes omises prendront la valeur NULL par défaut).

Les valeurs à insérer peuvent être précisées de deux façons:

avec la clause *VALUES*: une seule ligne est insérée, elle contient comme valeurs, l'ensemble des valeurs passées en paramètre dans la parenthèse qui suit la clause *VALUES*. Les données sont affectées aux colonnes dans l'ordre dans lequel les colonnes ont été déclarées dans la clause *INTO*

INSERT INTO Nom_de_la_table(colonne1,colonne2,colonne3,...)
VALUES (Valeur1, Valeur2, Valeur3,...)

Lorsque chaque colonne de la table est modifiée, l'énumération de l'ensemble des colonnes est facultative

avec la clause *SELECT*: plusieurs lignes peuvent être insérées, elle contiennent comme valeurs, l'ensemble des valeurs découlant de la sélection. Les données sont affectées aux colonnes dans l'ordre dans lequel les colonnes ont été déclarées dans la clause *INTO*

INSERT INTO Nom_de_la_table(colonne1,colonne2,...)
SELECT colonne1, colonne2,... FROM Nom_de_la_table2
WHERE qualification

Lorsque l'on remplace un nom de colonne suivant la clause *SELECT* par une constante, sa valeur est affectée par défaut aux tuples. Il n'est pas possible de sélectionner des tuples dans la table dans laquelle on insère des lignes (en d'autres termes *Nom_de_la_table* doit être différent de *Nom_de_la_table2*)

Modification de données

La modification de données (aussi appelée *mise à jour*) consiste à modifier des tuples (des lignes) dans une table grâce à l'ordre *UPDATE*. La modification à effectuer est précisé après la clause *SET*. Il s'agit d'une affectation d'une valeur à une colonne grâce à l'opérateur = suivi d'une expression algébrique, d'une constante ou du résultat provenant d'une clause *SELECT*. La clause *WHERE* permet de préciser les tuples sur lesquels la mises à jour aura lieu

UPDATE Nom_de_la_table
SET Colonne = Valeur_Ou_Expression
WHERE qualification

Suppression de données

La suppression de données dans une table se fait grâce à l'ordre *DELETE*. Celui-ci est suivi de la clause *FROM*, précisant la table sur laquelle la suppression s'effectue, puis d'une clause *WHERE* qui décrit la qualification, c'est-à-dire l'ensemble des lignes qui seront supprimées.

L'ordre *DELETE* est à utiliser avec précaution car l'opération de suppression est irréversible. Il faudra donc s'assurer dans un premier temps que les lignes sélectionnées sont bien les lignes que l'on désire supprimer!

CREATION DES TABLES

Le SQL, comportant un langage de définition de données (LDD), permet de créer des tables. Pour cela, il utilise le couple de mots clés *CREATE TABLE*.

La création de tables

Le création de tables se fait à l'aide du couple de mots-clés *CREATE TABLE*. La syntaxe de définition simplifiée d'une table est la suivante:

CREATE TABLE *Nom_de_la_table* (*Nom_de_colonne1 Type_de_donnée*,

Nom_de_colonne2 Type_de_donnée,
...);

Le nom donné à la table doit généralement (sur la plupart des SGBD) commencer par une lettre, et le nombre de colonnes maximum par table est de 254.

Les types de données

Pour chaque colonne que l'on crée, il faut préciser le type de données que le champ va contenir. Celui-ci peut être un des types suivants:

Type de donnée	Syntaxe	Description
Type alphanumérique	CHAR(n)	Chaîne de caractères de longueur fixe *n* (n<16383)
Type alphanumérique	VARCHAR(n)	Chaîne de caractères de n caractères maximum (n<16383)
Type numérique	NUMBER(n,[d])	Nombre de n chiffres [optionnellement *d* après la virgule]
Type numérique	SMALLINT	Entier signé de 16 bits (-32768 à 32757)
Type numérique	INTEGER	Entier signé de 32 bits (-2E31 à 2E31-1)
Type numérique	FLOAT	Nombre à virgule flottante
Type horaire	DATE	Date sous la forme 16/07/99
Type horaire	TIME	Heure sous la forme 12:54:24.85
Type horaire	TIMESTAMP	Date et Heure

L'option *NOT NULL*, placée immédiatement après la type de donnée permet de préciser au système que la saisie de ce champ est obligatoire.

Insertion de lignes à la création

Il est possible de créer une table en insérant directement des lignes lors de la création. Les lignes à insérer peuvent être alors récupérées d'une table existante grâce au prédicat *AS SELECT*. La syntaxe d'une telle expression est la suivante:

CREATE TABLE *Nom_de_la_table* (*Nom_de_colonne1 Type_de_donnée*,
Nom_de_colonne2 Type_de_donnée,
...)
AS SELECT *Nom_du_champ1*,
Nom_du_champ2,
...

FROM *Nom_de_la_table2*
WHERE *Prédicat*;

LES CONTRAINTES D'INTEGRITE

Expression de contraintes d'intégrité

Une contrainte d'intégrité est une clause permettant de contraindre la modification de tables, faite par l'intermédiaire de requêtes d'utilisateurs, afin que les données saisies dans la base soient conformes aux données attendues. Ces contraintes doivent être exprimées dès la création de la table grâce aux mots clés suivants:

CONSTRAINT
DEFAULT
NOT NULL
UNIQUE
CHECK

Définir une valeur par défaut

Le langage SQL permet de définir une valeur par défaut lorsqu'un champ de la base n'est pas renseigné grâce à la clause *DEFAULT*. Cela permet notamment de faciliter la création de tables, ainsi que de garantir qu'un champ ne sera pas vide.
La clause *DEFAULT* doit être suivie par la valeur à affecter. Cette valeur peut être un des types suivants:
Constante numérique
Constante alphanumérique (chaîne de caractères)
Le mot clé **USER** (nom de l'utilisateur)
Le mot clé **NULL**
Le mot clé **CURRENT_DATE** (date de saisie)

Le mot clé **CURRENT_TIME** (heure de saisie)
Le mot clé **CURRENT_TIMESTAMP** (date et heure de saisie)

Forcer la saisie d'un champ

Le mot clé *NOT NULL* permet de spécifier qu'un champ doit être saisi, c'est-à-dire que le SGBD refusera d'insérer des tuples dont un champ comportant la clause *NOT NULL* n'est pas renseigné.

Emettre une condition sur un champ

Il est possible de faire un test sur un champ grâce à la clause *CHECK()* comportant une condition logique portant sur une valeur entre les parenthèses. Si la valeur saisie est différente de *NULL*, le SGBD va effectuer un test grâce à la condition logique. Celui-ci peut éventuellement être une condition avec des ordres *SELECT*...

Tester l'unicité d'une valeur

La clause *UNIQUE* permet de vérifier que la valeur saisie pour un champ n'existe pas déjà dans la table. Cela permet de garantir que toutes les valeurs d'une colonne d'une table seront différentes.

Nommer une contrainte

Il est possible de donner un nom à une contrainte grâce au mot clé *CONSTRAINT* suivi du nom que l'on donne à la contrainte, de telle manière à ce que le nom donné s'affiche en cas de non respect de l'intégrité, c'est-à-dire lorsque la clause que l'on a spécifiée n'est pas validée.

Si la clause *CONSTRAINT* n'est pas spécifiée, un nom sera donné arbitrairement par le SGBD. Toutefois, le nom donné par le SGBD risque fortement de ne pas être compréhensible, et ne sera vraisemblablement pas compris lorsqu'il y aura une erreur d'intégrité. La stipulation de cette clause est donc fortement conseillée.

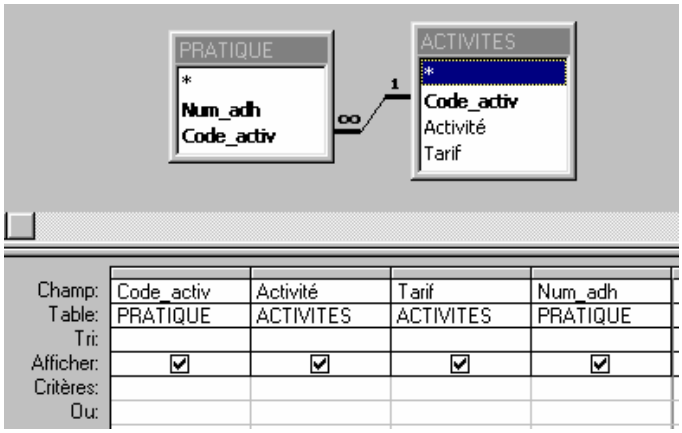
Exemple de création de table avec contrainte

Voici un exemple permettant de voir la syntaxe d'une instruction de création de table avec contraintes:

```
create TABLE clients(
Nom    char(30) NOT NULL,
Prenom char(30) NOT NULL,
Age    integer, check (age < 100),
Email  char(50) NOT NULL, check (Email LIKE "%@%"))
```

Définition de clés

Grâce à SQL, il est possible de définir des clés, c'est-à-dire spécifier la (ou les) colonne(s) dont la connaissance permet de désigner précisément un et un seul tuple (une ligne).

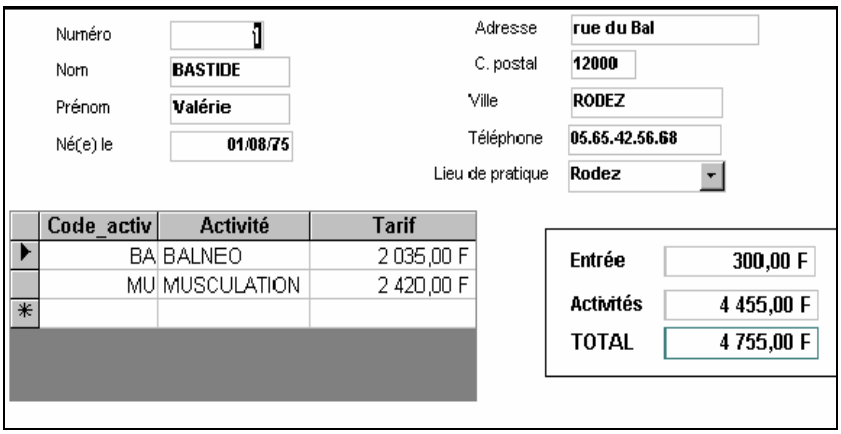


L'ensemble des colonnes faisant partie de la table en cours permettant de désigner de façon unique un tuple est appelé **clé primaire** et se définit grâce à la clause *PRIMARY KEY* suivie de la liste de colonnes, séparées par des virgules, entre parenthèses. Ces colonnes ne peuvent alors plus prendre la valeur *NULL* et doivent être telles que deux lignes ne puissent avoir simultanément la même combinaison de valeurs pour ces colonnes.

PRIMARY KEY (colonne1, colonne2, ...)

Lorsqu'une liste de colonnes de la table en cours de définition permet de définir la clé primaire d'une table étrangère, on parle alors de **clé étrangère**, et on utilise la clause *FOREIGN KEY* suivie de la liste de colonnes de la table en cours de définition, séparées par des virgules, entre parenthèses, puis de la clause *REFERENCES* suivie du nom de la table étrangère et de la liste de ses colonnes correspondantes, séparées par des virgules, entre parenthèses.

FOREIGN KEY (colonne1, colonne2, ...)
REFERENCES Nom_de_la_table_etrangere(colonne1,colonne2,...)



V LES FORMULAIRES et sous- FORMULAIRES

Reprenons l'exemple de départ. Le directeur du centre souhaite que lorsque l'adhérent vient s'inscrire, la saisie soit facile à faire et que le montant à payer apparaisse directement sur le formulaire de saisie. Le reçu des cotisations devra alors pouvoir être imprimé facilement.

Nous allons donc créer un formulaire de saisie qui nous permettra d'entrer les renseignement concernant l'adhérent et les activités qu'il souhaite pratiquer, et visualiser le montant total à payer. La saisie doit pouvoir se faire dans les deux tables **ADHERENT** et **PRATIQUE**, au moyen d'un seul formulaire, avec sous formulaire, qui se présentera ainsi :
Nous devons baser le formulaire principal sur une requête à partir des tables **ADHERENT** et **LIEU**, de façon à faire apparaître tous les champs concernant l'adhérent, son lieu de pratique et le montant du droit d'entrée qu'il aura à payer. Cette requête est triée sur le numéro de l'adhérent, et se nommera ***R_entrée par adhérent.***
Pour le sous formulaire nous créons une requête basée sur la table **PRATIQUE** et la table **ACTIVITE**, en filtrant :

- Code_activ de la table **PRATIQUE**
- Activités de la table **ACTIVITES**
- Tarif de la table **ACTIVITES**
- Num_adh de la table **PRATIQUE**

Elle se nommera ***R_activités par adhérent*** et aura l'allure suivante :
Le formulaire est créé à l'aide de l'assistant à partir de tous les champs de la première requête (pour le formulaire principal) , et des champs **Code_activ**, **Activité** et **Tarif** de la deuxième requête (pour le sous formulaire).
Le formulaire principal est nommé **SAISIE DES ADHERENTS** et le sous formulaire **SF activités par adhérent**.
Il reste maintenant à modifier le formulaire principal de façon à faire apparaître le montant global des cotisations concernant les activités et le montant total à payer par l'adhérent.
Le droit d'entrée figure déjà dans le formulaire principal dans un contrôle nommé **Entrée**. On peut le déplacer en bas du formulaire de façon à le regrouper avec les autres calculs.
Calculs dans le sous formulaire

Il faut modifier le sous formulaire **SF activités par adhérent** de façon à rajouter le calcul du total des cotisations, dans le pied de formulaire. Ce calcul est introduit dans un contrôle de type zone de texte par la formule suivante :

=SOMME([tarif])

Ce contrôle est nommé **Total**.

Calculs dans le formulaire principal

Dans le formulaire principal, il suffit maintenant de créer un contrôle de type Zone de texte dans lequel on va générer la formule suivante :

=[SF activités par adhérent].[Formulaire]![Total]

Celle-ci récupère le total des cotisations dans le sous-formulaire. Ce contrôle se nommera **Cotis**.

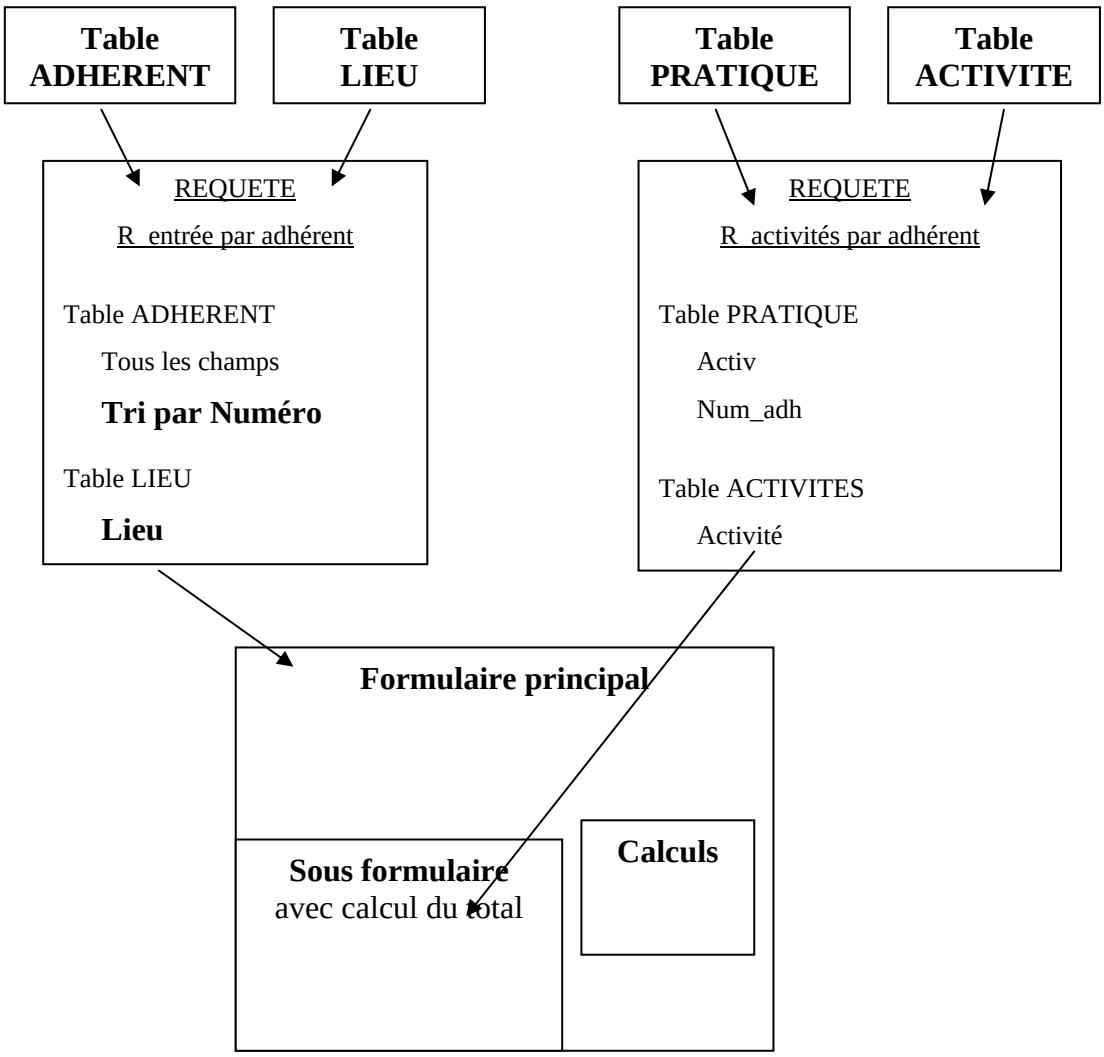
Dans un autre contrôle de type zone de texte, il suffit maintenant d'entrer la formule suivante :

=[Entrée]+[Cotis]

Celle-ci renvoie le total dû par l'adhérent.

Nous pouvons aussi ajouter un contrôle de type Zone de liste pour faciliter le choix du lieu de pratique.

Nous pouvons schématiser l'enchainement des éléments de la façon suivante :

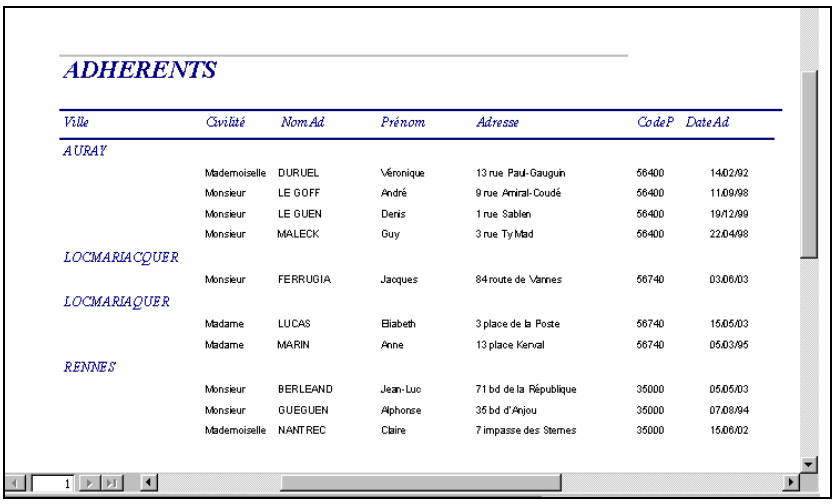


VI LES ETATS

L'état, dans un système de bases de données, est l'élément qui permet d'obtenir une présentation des données sur papier. Bien sûr vous pouvez imprimer n'importe quel autre objet, table, formulaire, etc. Mais vous aurez une liste exhaustive de toutes les données et de tous les champs, alors qu'un état donne la possibilité de n'imprimer que certains champs, avec une mise en forme choisie, et de plus on peut obtenir des calculs et des regroupements. Il est aussi possible par le biais de l'état de créer des étiquettes pour envoyer des courriers.

État-liste

Pour obtenir une état ainsi présenté, vous allez utiliser l'assistant-état d'Access.



ADHERENTS

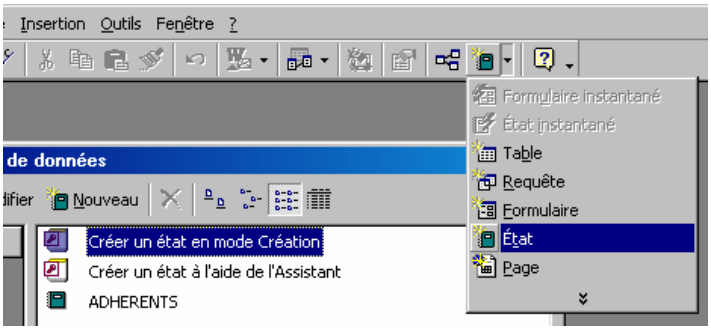
Ville	Civilité	NomAd	Prénom	Adresse	CodeP	DateAd
AURAY						
	Mademoiselle	DURUEL	Véronique	13 rue Paul-Gauguin	56400	14/02/92
	Monsieur	LE GOFF	André	9 rue Amiral-Coudé	56400	11/09/98
	Monsieur	LE GUEN	Denis	1 rue Sablen	56400	19/12/99
	Monsieur	MALECK	Guy	3 rue Ty Mad	56400	22/04/98
LOCMARIACQUER						
	Monsieur	FERRUGIA	Jacques	84 route de Vannes	56740	03/06/03
LOCMARIAQUER						
	Madame	LUCAS	Eliabeth	3 place de la Poste	56740	15/05/03
	Madame	MARIN	Anne	13 place Kerval	56740	05/03/95
RENNES						
	Monsieur	BERLEAND	Jean-Luc	71 bd de la République	35000	05/05/03
	Monsieur	GUEGUEN	Alphonse	35 bd d'Anjou	35000	07/08/94
	Mademoiselle	NANTREC	Claire	7 impasse des Sternes	35000	15/06/02

Cet état, comme vous le voyez, affiche les adhérents et les classe par ville. Tri alphabétique à l'intérieur de chaque ville.

- Dans le fenêtre Bases de données, placez-vous dans l'onglet Etats et cliquez créer un Etat à l'aide de l'assistant.
- Double-cliquez les champs suivants : Civilité, NomAd, Prénom, Adresse, CodePostal, Ville, DateAd. Cliquez Suivant.
- Dans le panneau suivant, choisissez le niveau de regroupement : Ville (double-cliquez dessus). Vous voyez se dessiner l'état. Cliquez Suivant.
- Choisissez NomAd dans le panneau qui suit. Cliquez Suivant.
- Cliquez encore deux fois Suivant et à la fin acceptez le nom **Adhérents** pour votre état.
- L'état se présente à l'écran en mode Affichage ; mais le fait d'avoir utilisé l'assistant ne vous oblige pas à accepter la présentation exactement telle qu'il la propose ! Vous préféreriez à juste titre avoir une présentation en Paysage pour ne pas tronquer certains champs.
- Passez en mode Création. Vous êtes à l'envers du décor.
- Regardez comment se présente la structure de cet état ; dans la section En-tête de page, vous avez les étiquettes (les titres des colonnes) dans la section En-tête de groupe vous avez le champ de regroupement (Ville) et dans la section Détail, ce sont les zones de texte (là où s'affichent les données).
- Commencez par mettre l'état en Paysage (Fichier Mise en page, Page, Paysage).
- Ensuite arrangez, déplacez, agrandissez les étiquettes pour qu'elles ne soient pas tronquées ; vous pouvez changer leur contenu si vous le souhaitez (*par exemple, au lieu de **DateAd** saisissez **Date d'adhésion***). Puis modifiez la place des zones de texte de la section Détail pour qu'elles soient conformes au étiquettes. *Ne changez rien au contenu des zones de texte, car comme vous le savez, ces contenus correspondent à des champs bien précis de la base.*
- Passez en Affichage pour voir (*pour les états, le mode Affichage est symbolisé par l'icône Aperçu*). Les traits bleus sont trop courts maintenant que vous avez mis la page en Paysage.
- Repassez en Création, repérez les traits (*ils sont dans la section En-tête de page au ras en haut et en bas de cette zone*) et agrandissez-les avec la souris.
- Changez le titre de l'état : juste après **ADHÉRENTS**, saisissez **PAR VILLE**.
- Jugez de l'effet en mode Affichage. Imprimez votre état.

Planche d'étiquettes

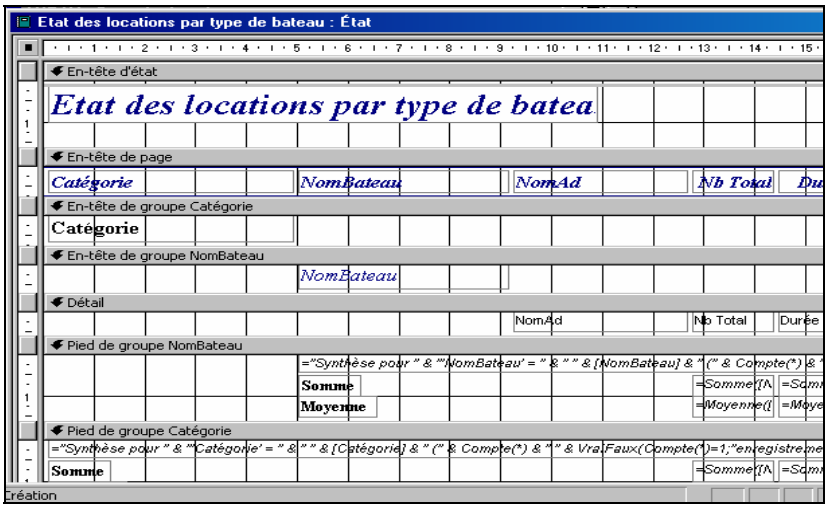
- Depuis l'élément Etats de la fenêtre Base de données, cliquez sur l'icône Nouvel Objet, choisissez Etat, puis Assistant Etiquette, et choisissez ADHÉRENTS comme table.



- Cliquez deux fois sur Suivant.
- Double-cliquez sur les champs à insérer dans l'étiquette en plaçant des espaces entre eux. Si vous ne mettez pas d'espaces, ne vous plaignez pas ensuite que la présentation est vilaine.
- Dans le panneau suivant, choisissez le tri sur le NomAd pour éditer vos étiquettes en ordre alphabétique. Suivant. Terminer. Vos 10 étiquettes doivent tenir sur une page A4.

État de regroupement et de synthèse

- Vous avez créé précédemment une requête qui présente les locations par types de bateaux. Vous souhaiteriez disposer d'un Etat basé sur cette requête qui effectuerait des calculs : d'abord le nombre total et le nombre moyen de participants à un séjour pour un type de bateau, ensuite la durée moyenne d'un séjour ainsi que la durée totale par type de bateau.
- Dans Etats, cliquez Créer un état à l'aide de l'assistant. Basez-vous sur la requête **LOCATIONS PAR TYPE DE BATEAU**.
- Ajoutez les champs Catégorie, NomBateau, NomAd, Nb total, Durée. Cliquez 3 fois Suivant.
- Cliquez Options de synthèse.
- Cochez Somme et Moyenne pour les deux lignes.
- Laissez Détail et Total, cliquez Ok. Choisissez NomAd comme critère de tri, cliquez 3 fois suivant. Donnez comme nom à l'état : **ÉTAT DES CROISIÈRES PAR CATÉGORIE DE BATEAUX**.
- Passez en création ; supprimez les zones en italiques « synthèse….. ; » <



- Revenez en Affichage pour voir si cela vous convient. Si vous le souhaitez, vous pouvez apporter des améliorations à la présentation.

Vous allez enregistrer quelques réservations supplémentaires et ensuite vous réaliserez un autre type d'état, visant à calculer le chiffre d'affaires des croisières.

Madame MARIN
AU FIL DE L'EAU
24 adultes et 7 enfants 24 juillet

Monsieur LE GUEN
AU FIL DE L'EAU
15 adultes et 18 enfants 26 juillet

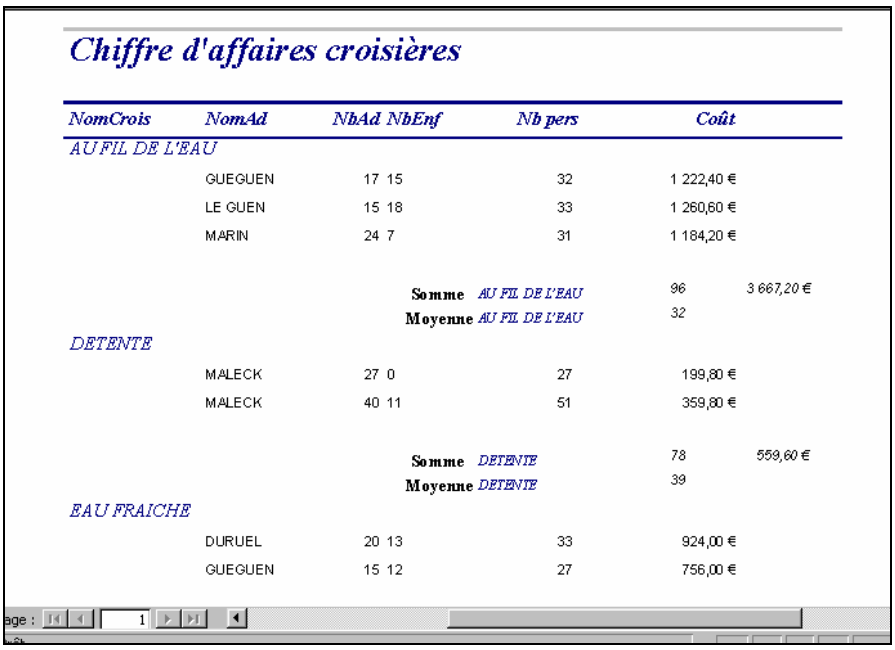
Monsieur MALECK
DETENTE
27 adultes 0 enfants 2 août

Madame MARIN
EVASION
34 adultes 5 août

Monsieur GUEGUEN
EAU FRAICHE
15 adultes 12 enfants 7 août

Mademoiselle DURUEL
EAU FRAICHE
20 adultes 13 enfants 10 août

Vous allez créer un état qui permettra de connaître le chiffre d'affaires des croisières, avec le détail par croisière. Il aura cette allure :



Chiffre d'affaires croisières

NomCrois	NomAd	NbAd	NbEnf	Nb pers	Coût	
AU FIL DE L'EAU						
	GUEGUEN	17	15	32	1 222,40 €	
	LE GUEN	15	18	33	1 260,60 €	
	MARIN	24	7	31	1 184,20 €	
			Somme AU FIL DE L'EAU	96		3 667,20 €
			Moyenne AU FIL DE L'EAU	32		
DETENTE						
	MALECK	27	0	27	199,80 €	
	MALECK	40	11	51	359,80 €	
			Somme DETENTE	78		559,60 €
			Moyenne DETENTE	39		
EAU FRAICHE						
	DURUEL	20	13	33	924,00 €	
	GUEGUEN	15	12	27	756,00 €	

Vous allez d'abord pour plus de facilité créer une requête qui regroupe les tables utiles, d'où vous pourrez extraire les champs que vous voulez utiliser.

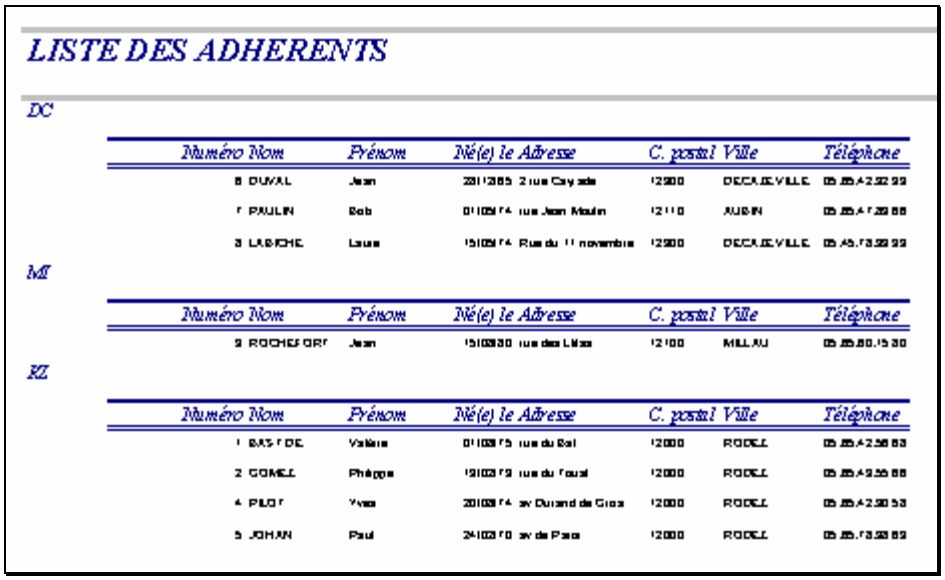
- Créez une nouvelle requête en mode Création. Ajoutez les tables **ADHÉRENTS**, **RÉSERVATIONS** et **CROISIÈRES**.
- Ajoutez les champs NomCrois, NomAd, NbAd, NbEnf.
- Créez à la suite des champs affichés deux champs calculés comme suit :

Nb Pers : [NbAd]+[NbEnf]
Coût total : [TarifAd]*[NbAd]+[TarifEnf]*[NbEnf]

- Exécutez la requête pour voir le résultat puis fermez-la et enregistrez-la sous le nom **POUR ÉTAT CROISIÈRES**.
- Dans l'élément Etats, cliquez Créer un Etat avec l'aide de l'assistant.
- Choisissez la requête **POUR ÉTAT CROISIÈRES**, ajoutez tous les champs en cliquant sur le double chevron.
- Cliquez Suivant. Regroupez sur le NomCrois. Cliquez Options de synthèse, cochez Somme pour Nb Pers et pour Coût et Moyenne pour Nb Pers. Ok.
- Triez selon les NomAd. Suivant. Allez jusqu'au bout, donnez comme nom **CHIFFRE D'AFFAIRES CROISIÈRES**.
- Passez en Création, agrandissez les étiquettes et/ou les zones qui pourraient être tronquées (*faites notamment attention aux zones qui contiennent des chiffres, car si elles sont tronquées, on peut croire que les calculs sont faux alors qu'ils sont cachés*).
- Supprimez la mention en italique "*Synthèse;……*" qui n'est pas utile.
- Pour peaufiner, rapprochez les mots Somme et Moyenne des chiffres concernés. Ecrivez en clair les intitulés.

ETAT BASE SUR UNE TABLE

Nous voulons éditer la liste des adhérents avec un regroupement par lieu.
Il suffit d'utiliser l'assistant état, en sélectionnant la table **ADHERENT**. Nous obtiendrons l'état suivant :



LISTE DES ADHERENTS

DC

Numéro	Nom	Prénom	Né(e) le	Adresse	C. postal	Ville	Téléphone
6	DUVAL	Jean	23/12/65	2 rue Cayade	12500	DECAZEVILLE	05.65.42.92.93
7	PAULIN	Bob	01/05/74	rue Jean Moulin	12110	AUBIN	05.65.47.29.88
8	LABICHE	Laura	15/05/74	Rue du 11 novembre	12500	DECAZEVILLE	05.65.73.92.93

MI

Numéro	Nom	Prénom	Né(e) le	Adresse	C. postal	Ville	Téléphone
9	ROCHEFORT	Jean	15/08/80	rue des Lilas	12100	MILLAU	05.65.60.15.30

RZ

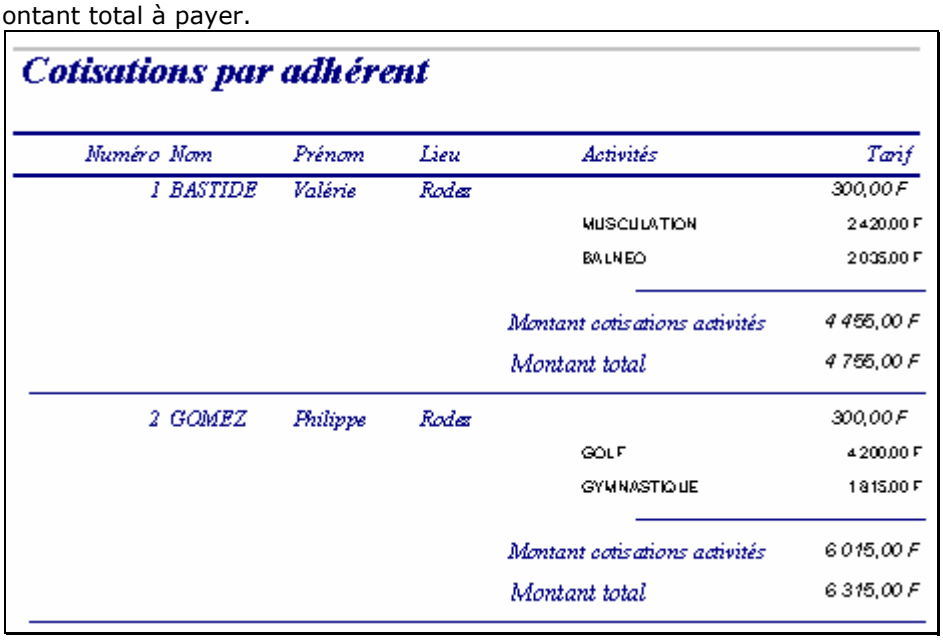
Numéro	Nom	Prénom	Né(e) le	Adresse	C. postal	Ville	Téléphone
1	BASTIDE	Valérie	01/08/75	rue du Bal	12000	RODEZ	05.65.42.56.83
2	GOMEZ	Philippe	15/08/72	rue du Faoul	12000	RODEZ	05.65.49.56.88
4	PILOT	Yves	20/08/74	av Durand de Gros	12000	RODEZ	05.65.42.30.53
5	JOHAN	Paul	24/08/70	av de Paris	12000	RODEZ	05.65.73.93.83

État basé sur une requête

Nous voulons maintenant éditer un état général des cotisations des adhérents.
Il suffit de créer une requête filtrant les renseignements suivants :

Table **ADHERENT** : Numéro adhérent, Nom, Prénom, Entrée,
Table **LIEU** : Lieu de pratique,
Table **ACTIVITE** : Activité, Tarif

Nous créons ensuite un état avec regroupement sur les caractéristiques de l'adhérent.
Nous rajoutons le calcul du cumul des cotisations des activités en pied de groupe, ainsi que le calcul du Montant total à payer.



Cotisations par adhérent

Numéro	Nom	Prénom	Lieu	Activités	Tarif
1	BASTIDE	Valérie	Rodez		300,00 F
				MUSCULATION	2 420,00 F
				BALNEO	2 035,00 F
				Montant cotisations activités	4 455,00 F
				Montant total	4 755,00 F
2	GOMEZ	Philippe	Rodez		300,00 F
				GOLF	4 200,00 F
				GYMNASTIQUE	1 815,00 F
				Montant cotisations activités	6 015,00 F
				Montant total	6 315,00 F

Après mise en forme, il se présentera ainsi :

ETAT BASE SUR UNE REQUETE PARAMETREE

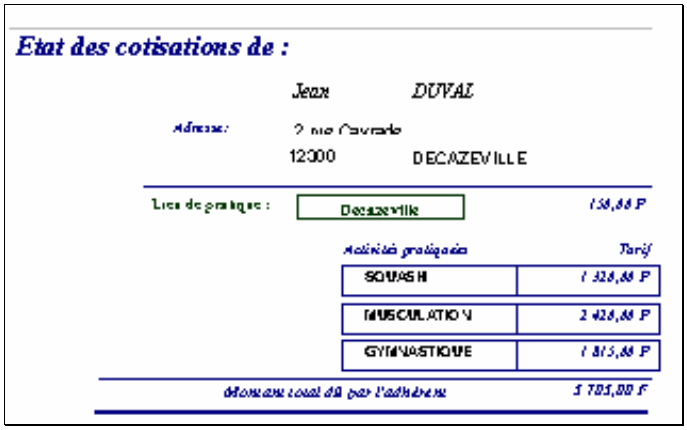
De la même façon il est possible de créer un état des cotisations d'un seul adhérent.
Nous créons une requête avec les renseignements suivants :

- Table **ADHERENT** : Nom, prénom, adresse, code postal, ville et entrée
- Table **LIEU** : Lieu
- Table **ACTIVITE** : Activité et tarif

Dans la colonne **Num_adh** de la requête, il suffit d'ajouter le paramètre suivant sur la ligne critère :

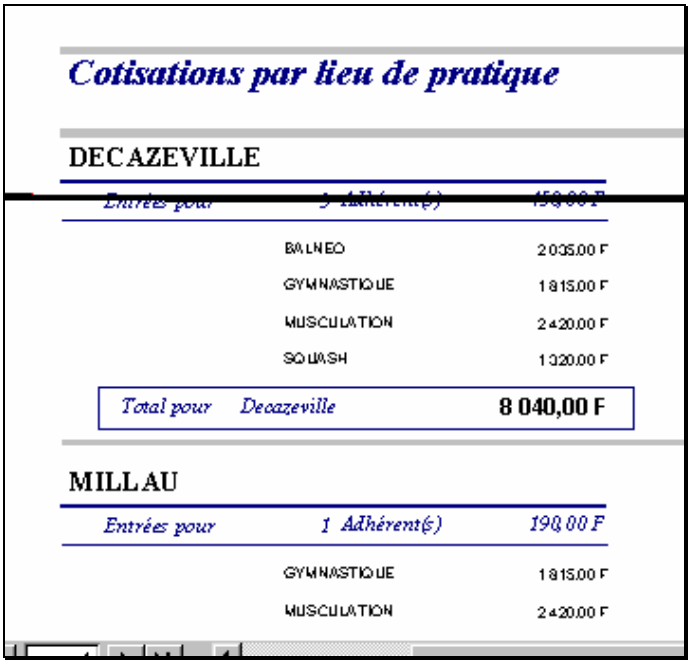
[Saisir le numéro de l'adhérent].

De cette façon, lorsqu'il est fait appel à cette requête, un message demande à l'utilisateur de saisir le numéro de l'adhérent. Les données sont alors filtrées sur ce numéro.
Nous créons ensuite un état des cotisations pour un adhérent. Avec tous les renseignements ci-dessus. Nous rajoutons le calcul du montant dû par l'adhérent.



L'état peut se présenter ainsi après mise en forme :

AUTRES ETATS



On pourra ainsi créer d'autres états en fonction des besoins de l'application. Par exemple on peut créer l'état des cotisations par lieu de pratique :

PARTIE 4 TRAVAUX D'APPLICATION

PARTIE 5 Travaux d'application

Introduction

Les travaux d'applications sont conçu pour être une synthèse des acquis théorique vue dans le modules 5 « manipulation de base de données », ils sont poses sous forme d'atelier enchaîner qui abouti a l'élaboration d'une application de base de donnée touchant la totalités des fonctionnalités d'un SGBDR Tel que MS Access. Le projet est portés sur **la GESTION D'UN FOURNISSEUR DE PHARMACIES**

ATELIER N° 1 Création de tables

Objectifs :
- Créer une base de donnée sous MS Access
- Créer des table sous MS Access
- Définir des Clés

Travail effectué avec:
- Un micro-ordinateur par 2 stagiaires
- Un logiciel de base de données MS ACCESS
- Une imprimante

Un laboratoire de médicaments désire informatiser la gestion de stock des médicaments ainsi que la gestion des commandes, et les médicaments périmés et d'autres opérations manuelles.

1) Créer une nouvelle base de données sur le disque dur, en précisant le nom de la base « gestion d'un Laboratoire de médicaments ».
2) Créer la table « médicaments » pour le stockage des informations concernant tous les médicaments
   - Sur la 1ère colonne, vous préciserez les noms des champs,
   - Sur la 2ème colonne, vous préciserez les types des champs,
   - Sur la 3ème colonne, vous donnez les descriptions des champs.

La liste des champs de la table « **médicaments** » est la suivante :

Nom du champ	**Type de données**
Référence médicament	Numérique
Libelle médicament	Texte
Date de péremption	Date
Prix unitaire	Numérique
Quantité en Stock	Numérique
Quantité min	Numérique

4) Une fois les noms des champs spécifiés, vous devez définir la clé primaire pour la table en utilisant l'icône **.**
5) Créer la table Catégorie de médicament nommée (Catégorie)

La liste des champs de la table « **Catégorie** » est la suivante :

Nom du champ	**Type de données**
Numéro Catégorie	Numérique
Libelle Catégorie	Texte

6) Créer la table pharmacie

La liste des champs de la table « Pharmacie » est la suivante :

Nom du champ	**Type de données**
Numéro pharmacie	Numérique
Nom pharmacie	Texte
Adresse	Texte
Tel	Texte

6) Fermez la base de données « **Fournisseur de pharmacies** » avec **Fichier** / **Fermer**.

ATELIER N° 2 Remplissage des tables

Objectifs :
- Remplir et saisir les donnée table sous MS Access
- Mettre en forme des tables et Champs
- Gérer les tables sous MS Access

Travail effectué avec:
- Un micro-ordinateur par 2 stagiaires
- Un logiciel de base de données MS ACCESS
- Une imprimante

1) Entrez les données des médicaments dans la table « **médicaments** », en cliquant sur le bouton table puis ouvrir.
- Informations à saisir dans la table médicaments :

Référence Médicament	Libelle Médicament	Date Péremption	Prix unitaire	Quantité en Stock	Quantité en Seuil
0001	Doliprane	01/01/2004	12	100	20
0002	Aspégic	12/03/2004	50	200	50
….	….				
….	….				

2) Triez les enregistrements
3) Recherchez l'enregistrement n° 0007
4) Supprimer l'enregistrement n° 0007
5) Modifier la largeur des colonnes
6) Modifier la hauteur des lignes
7) Masquer la colonne « **Quantité min** »
8) Réafficher la colonne « **Quantité min** ».
9) Figez la colonne « **Libelle médicament** »
10) Changer la couleur de l'arrière plan de la table
11) Faites une copie de la table (Données et structure)
12) Renommez la table « Médicament »
13) Supprimez la table que vous venez de renommer.
14) Entrez les données des Catégories dans la table « **catégorie** »

Numéro catégorie	Libellé catégorie
01	Antibiotique
02	Antiinflamatoire
03	Antalgique
….	….

15) Entrez les données des pharmacies dans la table « pharmacie »

Numéro pharmacie	Nom pharmacie	Adresse	Tel
001	Pharma-Maroc	36 Av Toubkal Fès	56-89-75-55
002	MédiAgdal	25 Av mohamed V Casablanca	75-48-89-36
….	……		

ATELIER N° 3 Propriété des tables

Objectifs :
- Définir et configurer les propriétés des tables
- Définir et configurer les propriétés des champs

Travail effectué avec:
- Un micro-ordinateur par 2 stagiaires
- Un logiciel de base de données MS ACCESS
- Une imprimante

A) Copier la table « médicament » sous un autre nom « médicament1 » (la structure seulement).
B) Ouvrez la table médicament1 et définissez les propriétés des champs.

Après avoir défini chaque nouveau format, basculez du mode création vers le mode feuille de données pour tester le format que vous venez de créer

1. Référence médicament doit être numérique
2. Le Libellé du médicament doit avoir une entrée indispensable, sachant qu'une chaîne vide est interdite
3. La quantité min doit avoir une valeur par défaut « 10 ».
4. Le libellé du médicament doit être saisi en majuscule et affiché en bleu
5. Le tel domicile doit comporter uniquement des chiffres
6. La date de péremption doit être supérieur à la date du système
7. La Quantité min doit être comprise entre 10 et 100
8. Pour la date de péremption, elle doit être comprise entre date système +1jour et la date système +5ans.
9. Lorsque vous entrez une date qui ne respecte pas la règle de validité, Access ouvre une boîte de dialogue affichant un message d'erreur. Modifiez le message de cette boîte de dialogue.
10. Créer une règle de validité qui permet de contrôler si la quantité en stock est supérieure à la quantité min. (**Propriété de la table**). Afficher un message d'erreur personnalisé.

ATELIER N° 4 Les Requêtes

Objectifs :
- Créer des requêtes a l'aide d'interface graphique de MS Access
- Créer des requêtes a l'aide du code SQL

Travail effectué avec:
- Un micro-ordinateur par 2 stagiaires
- Un logiciel de base de données MS ACCESS
- Une imprimante

N.B *Les stagiaires doivent réaliser l'atelier en utilisant les deux modes d'interrogation des bases de données (graphique et code SQL).*

1. Créez une requête n'affichant que les médicament(s) dont la date de péremption est inférieure à 2003.

Procédez comme suit :

- Appuyez sur le bouton **Requêtes** puis **Nouveau**
- **Mode création** puis **Ok**
- Sélectionnez la **table médicaments** et appuyer sur **Ajouter,** puis **Fermer**
- Glissez les champs que vous voulez afficher dans la requête
- Placez le critère **<=31/12/2003** dans le champ date de péremption.

2. Enregistrez la requête sous le nom « liste 2003 » avec **Fichier** / **Enregistrer**.
3. Ajoutez le champ « code catégorie » dans la table « médicament» (en utilisant la commande Insérer Liste de choix) afin de pouvoir créer une requête qui regroupe les deux tables « médicament » et « catégorie ».
4. Modifier la requête « liste des médicaments par catégorie », en affichant que les médicaments ayant la catégorie « Antibiotique ».
5. Modifier la requête « liste des médicaments par catégorie », en affichant que les médicaments ayant la catégorie « Antibiotique » et la date de péremption supérieure à 2003.
6. Créer une nouvelle table « Achat », pour chaque Achat on veut savoir le Nom pharmacie, le médicament, la quantité acheté, la date d'achat.
7. Créer la requête qui affiche la liste des achats effectués.
8. Modifier la requête en affichant que les achats effectués par une pharmacie spécifiée comme paramètre.
9. Faites une copie de la requête.
10. Supprimez la requête.
11. Modifiez la requête « Liste des médicaments par catégorie » par « liste des médicament par catégorie et par date » en paramétrant la catégorie et la date.
12. Créer une requête sous le nom « Liste des médicaments commençant par C » permettant d'afficher l'ensemble des médicaments dont le nom commence par la lettre C.
13. Modifier la requête précédente de telle manière à pouvoir afficher toutes les médicaments dont le nom commence avec le caractère spécifié en tant que paramètres.
14. Créer une nouvelle requête nommée « liste des médicaments achetés pendant une période » permettant d'afficher la liste des médicaments achetés pendant une date début et une date fin.
15. Créer une requête de la table médicament qui permet de calculer automatiquement le nombre d'années restant de chaque médicament pour être périmé.

16. Créer une requête permettant l'affichage de la liste des pharmacies se trouvant à rabat.
17. Créer une requête permettant l'affichage de la liste des pharmacies se trouvant à rabat et dont le numéro de téléphone commence par le 77.
18. Créer une requête permettant l'affichage du nombre total d'achat par pharmacie pendant une année.
19. Créer une requête permettant l'affichage du nombre total de médicaments par catégorie donnée comme paramètre.
20. Créer une requête paramétré permettant d'afficher le montant minimum d'achat d'une pharmacie.
21. Afficher le prix de médicament le plus cher d'une catégorie donnée.

22. Créer la requête qui affiche les nom des médicaments dont la date de péremption est aujourd'hui.

23. Augmenter de 10% le prix des médicaments dont le prix est supérieur à 100.

24. Requête Création de table

- Requête création de table pour copier tous les médicaments dont la quantité est inférieure à 30, dans une table **Rupture.**
- Requête création de table pour copier juste les codes et les désignations des médicaments dans une table médicament.
- Requête création de table pour copier la table pharmacie dans une nouvelle table **pharmacie sauve.**

25. Requête d'ajout

- Requête ajout permettant d'ajouter des pharmacies dans la table pharmacie à partir de la table pharmacie sauve.

26. Requête Suppression

- Requête suppression permettant de supprimer un médicament quelconque dans la table médicament.
- Requête suppression permettant de supprimer les achats passés avant le (01/01/2001).
- Requête suppression permettant de supprimer les achats d'une pharmacie donnée.

27. Requête mise à jour

- Requête mise à jour permettant d'effectuer une réduction de 10% sur les prix unitaire de tous les médicaments.
- Requête mise à jour permettant d'augmenter la quantité disponible de 20% pour tous les médicaments.

ATELIER N° 5 Les Formulaires

Objectifs :
- Créer des Formulaires de saisie en mode création
- Mettre en forme les formulaires

Travail effectué avec:
- Un micro-ordinateur par 2 stagiaires
- Un logiciel de base de données MS ACCESS
- Une imprimante

1. Créez un nouveau formulaire de la table « **médicament** » en « **mode création** ».

- Appuyez sur le bouton **Formulaire**, puis **Nouveau**.
- Choisissez la table **médicament**.
- Mode création.
- Changez la couleur d'arrière plan.
- Insérez le titre « gestion du personnel 
- Modifier le format du titre « 3D enfoncé ». 
- Insérez les champs de la table médicament. 
- Insérez un champ affichant la date du jour.
- Modifiez le format des champs (taille, police, couleur, style).
- Insérez une image. 
- Basculez en mode formulaire pour visualiser les modifications. 

2. Utilisez les boutons de navigation, situés en bas du formulaire, pour passer d'un enregistrement à un autre.

3. Créez un deuxième formulaire mode création de la table Catégorie.

4. Modifiez le format de votre formulaire en basculant en mode création.

5. Créez un formulaire de la table « **médicament**» dans le forma  **laire** (représentant les valeurs en lignes et colonnes comme un tableau d'une feuille de calcul).

6. Insérez dans votre formulaire un graphique Excel représentant les données suivantes :

Année	Antibiotique	Antinflamatoire
2000	1012	1520
2001	1520	2006
2002	1789	2500

7. Créez un formulaire de type **graphique** représentant les données de la requête « **Liste des médicaments par catégorie** ».

8. Le formulaire suivant sera de type **sous-formulaire**. Etant entendu que dans une catégorie donné, il y a plusieurs médicament, il est très utile pour vous d'obtenir dans un tableau tout les médicaments d'une catégorie.

9. Jusqu'à présent, vous avez enregistré dans les tables des données rédigées par écrit. Le champ de type objet OLE permet d'enregistrer dans la table des images ou des photos. Voilà pas à pas la méthode pour l'insertion d'un champ de type OLE dans une table.
- Ouvrez la table « **pharmacie** ».
- Ajoutez un champ de type **OLE**, nommé « **logo** ».
- Basculez en mode **feuille de données**.
- Positionnez le curseur sur le champ « **logo** » du 1er enregistrement.
- Choisissez le menu **Insertion / Objet**.
- Cherchez le nom du fichier à insérer à partir du bouton Parcourir.
- Spécifier le chemin de recherche et appuyer sur Ouvrir.
- Créer un nouveau formulaire de la table Pharmacie en insérant le nouveau champ « logo ».

ATELIER N° 6 Les Formulaires (suite)

Objectifs :

- Créer des Formulaires de saisie en mode Assistant
- Créer et manipuler les contrôles

Travail effectué avec:

- Un micro-ordinateur par 2 stagiaires
- Un logiciel de base de données MS ACCESS
- Une imprimante

1. Créez un nouveau formulaire de la table « **médicament** » en « **mode assistant** » pour ajouter de nouveau enregistrements sans modifier les anciens.
2. Créez un autre formulaire de la table médicament permettant de modifier les données des champs à part celui du code médicament.
3. Copier le formulaire et modifiez dans les propriétés de telle façon à pouvoir consulter les enregistrements sans mise à jour.
4. Créez un formulaire principal reliant l'ensemble des formulaires précédents. (Utilisation des boutons).
5. Ajoutez dans les formulaires (ajout modification et consultation) un bouton de fermeture ; pour pouvoir revenir au formulaire principal.
6. Créer un formulaire Création et mise à jour de la table « Pharmacie » en utilisant des boutons de commandes.
   - Bouton Création
   - Bouton Modification
   - Bouton Suppression
   - Bouton Consultation
   - Boutons de déplacements
   - Bouton Quitter qui revient au menu principal
7. Créer un formulaire qui permet l'affichage des informations de la table Achat en accordant une remise de prix pour chaque pharmacie de 0,75.

ATELIER N° 7 Les Etats

Objectifs :
- Créer et imprimer les états basant sur des tables et des requêtes
- Créer des de regroupement
- Mettre en forme les états

Travail effectué avec:
- Un micro-ordinateur par 2 stagiaires
- Un logiciel de base de données MS ACCESS
- Une imprimante

1. Créez un Etat représentant la liste des médicaments
- Appuyez sur Etat puis Nouveau
- Sélectionnez la table et le type d'état que vous souhaitez créer, cliquez sur OK.
- Sélectionnez les champs à insérer dans l'état (de la même façon que l'insertion des champs dans un formulaire).
- Cliquez sur Suivant.
- Indiquez comment doit s'effectuer le regroupement des enregistrements et Cliquez sur Suivant.
- Sélectionnez le ou les champs en fonction desquels les enregistrements doivent être triés lors de l'impression de l'état et cliquez sur suivant.
- Choisissez la présentation souhaitée et cliquez sur suivant.
- Tapez le titre de l'état.
- Activez l'option Placer tous les champs sur une seule page pour que chaque enregistrement soit imprimé sur une seule ligne.
- Cliquez sur le bouton Aperçu pour visualiser le résultat.
- Basculez en mode création afin de modifier la présentation de votre Etat.

2. Créez un nouvel Etat permettant le regroupement des médicaments par catégorie.

3. Créez un Etat style tabulaire permettant d'éditer la liste des pharmacies.

4. Créez un Etat basé sur la table pharmacie pour laquelle il faut générer des étiquettes en vue d'un publipostage.
- Cliquez sur Etat puis sur le bouton Nouveau ;
- Cliquez sur Assistant Etiquette ;
- Choisissez la table pharmacie ;
- Choisissez la mise en forme de vos étiquettes et le format de vos étiquettes ;
- Placez les champs
- Cliquez sur le bouton Terminer pour visualiser le résultat.

5. On vous propose la réalisation d'un état où les médicaments seront regroupés par ordre alphabétique.
- Créez un état l'Assistant Etat en vous basant sur la table médicaments ;
- Placez les champs nécessaires pour obtenir un état;
- Choisissez le champ Nom médicament comme champ de regroupement ;
- Choisissez un type de présentation, puis un style et cliquez sur le bouton Terminer.

6. *Créer un état de la table médicament basé sur une mise en forme conditionnelle permettant de sélectionner les prix de médicament >50 Dh.*

7. Créer un état basé sur une mise en forme conditionnelle permettant de sélectionner les médicaments dont la quantité stock < à la quantité minimal.

8. Créer l'état qui permet d'afficher la somme des achats de chaque pharmacie par page.

9. Créer l'état qui permet de regrouper les médicaments par ordre alphabétique (Utiliser l'option trier et regrouper) et changer la propriété de la zone (Gauche$([libelle médicament] ; 1]

10. Créer l'état qui affiche à partir d'une requête croisé dynamique le nombre de médicaments par catégorie et par date de péremption.

ATELIER N° 8 Les Macros

Objectifs :

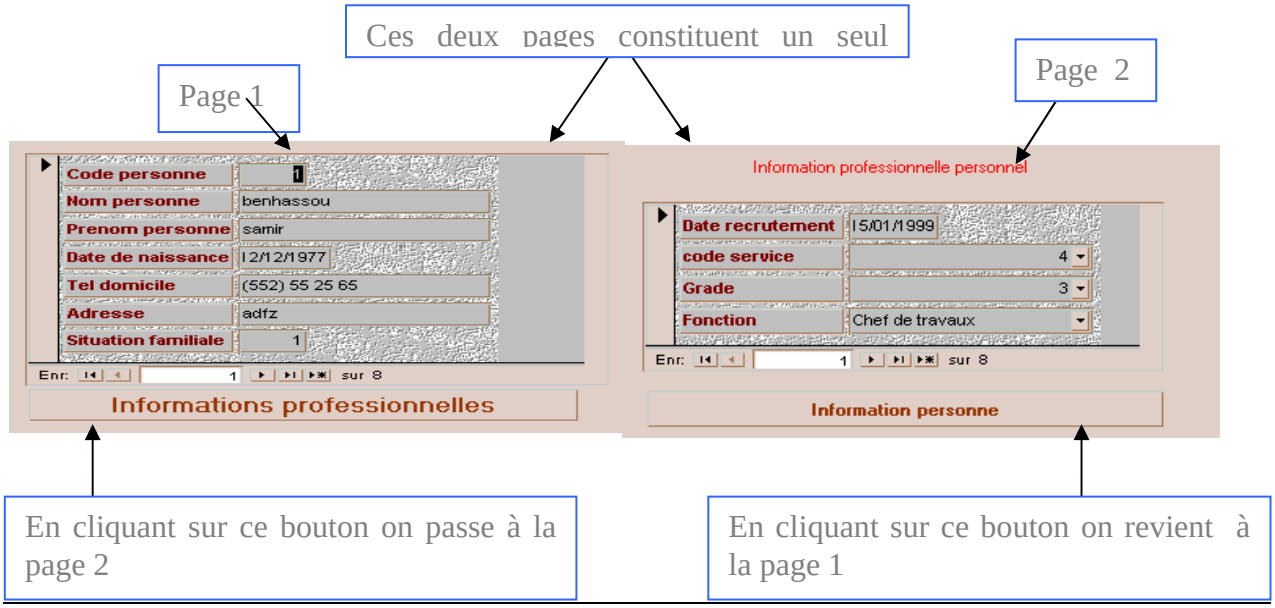
- Consulter des données dépendants dans deux formulaires, en Utilisan les macros
- Gérer et naviguer entre les différents objets de MS Access en utilisant les macros

Travail effectué avec:

- Un micro-ordinateur par 2 stagiaires
- Un logiciel de base de données MS ACCESS
- Une imprimante

1. Comment synchroniser l'affichage de deux formulaires après ouverture via une macro Exemple (Liste des médicaments par catégorie).

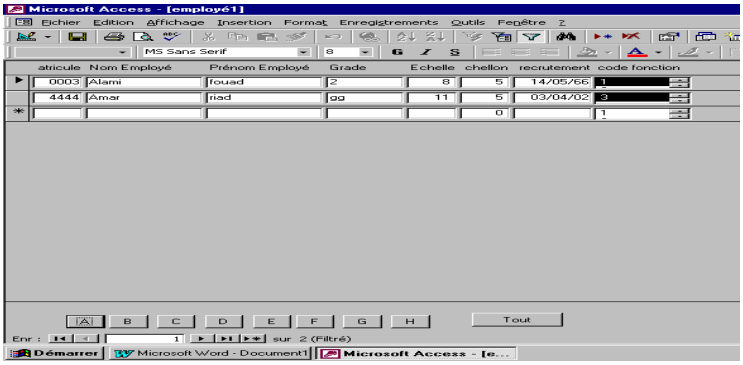
2. Créer des formulaires de la table médicament à plusieurs pages via le contrôle saut de page et d'une Macro ?



Exemple

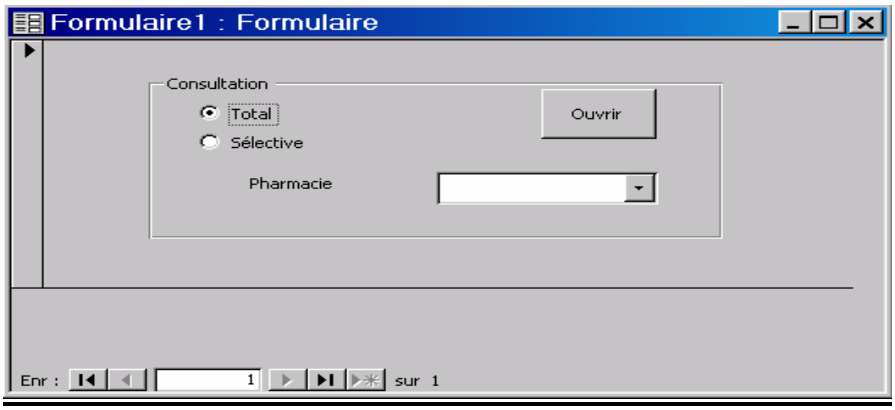
3. Créer la macro qui permet d'extraire des enregistrements de la table médicaments à condition d'avoir la possibilité de limiter des enregistrements dont le nom de médicament commence par une lettre.

Exemple



1. Créer la macro qui permet d'agrandir un formulaire.
2. Créer le formulaire qui permet de consulter selon des critères

- La liste des achats de toutes les pharmacies
  OU
- La liste des achats d'une pharmacie choisie dans la liste de choix (avec un contrôle si la liste est vide lancez un message)



3. Créer la macro qui permet de créer une barre de menu

ATELIER N° 9 Gestion des commandes (Atelier Récapitulatif)

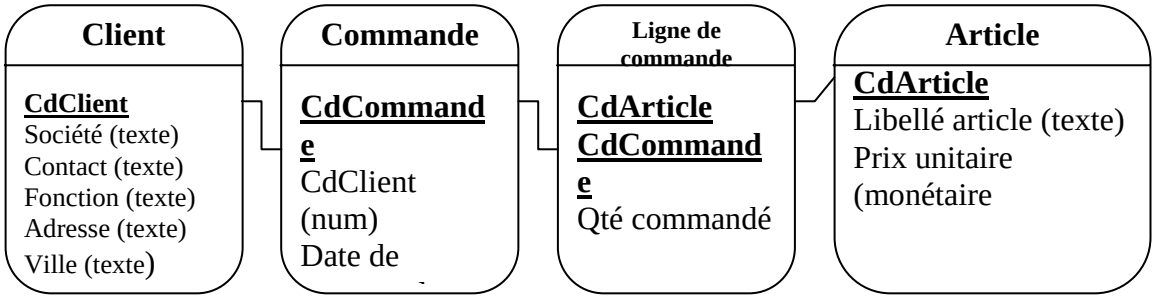
Objectifs :

- Créer une application de base de donnée sous Access
- Créer un formulaires menu

Travail effectué avec:

- Un micro-ordinateur par 2 stagiaires
- Un logiciel de base de données MS ACCESS
- Une imprimante

1. Créez une nouvelle base de données « gestion des commandes ».
2. Créez les tables (Client, Commandes, Lignes-commande et article) pour le stockage des informations de la base de données.



3. Définissez la clé primaire de chaque table.
4. Entrez les données dans les tables pour pouvoir effectuer des tests d'extractions au niveau des requêtes.

Tables Clients

Code client	Société	Contact	Fonction	Adresse	Ville	Pays	Tel	Fax
1	SOMACA	ALAMI Samir	Représentant	45 Av des fars	Casablanca	Maroc	77-77-89	77-78-79
2	EXOTIC LIQUIDE	Charlote cooper	Assistant Export	66 Belveder	Paris	France	77-89-45-88	56-89-78-45
3	SPECIALITE BISCUITS	Tijani Mohamed	Chef de produit	Av Mohamed V	Rabat	Maroc	56-89-45	56-56-89
4	COSUMAR	Salmi brahim	Directeur de marketing	56 IN SABAA	Casablanca	Maroc	75-58-89	45-55-55
5	Blondel père et fils	Frédérique Citeaux	Directeur marketing	24, place klèber	London	Royaume-Uni	12-56-89-56	44-44-77-44

Tables Commandes

Code commande	Code client	Date commande	Date livraisonn
B01	1	05/01/1998	05/02/1998
B02	1	10/12/1997	11/12/1997
B03	2	25/04/1998	25/05/1998
B04	1	23/05/1997	12/06/1997

Tables Lignes commandes

Code ligne	Code commande	Code article	Qté commandée
1	B01	0001	12
2	B01	0002	13
3	B01	0003	20
1	B02	0001	40
1	B03	0003	16
1	B04	0001	20
2	B04	0004	10
3	B04	0002	5
4	B04	0005	12

Table Article

Code article	Libelle article	Prix unitaire
0001	Chaise	200
0002	Table	1000
0003	Bureau	2000
0004	Tableau blanc	1500
0005	Photocopieur	10000

5. Créez :
- Une requête permettant l'affichage des commandes passées entre le 01/1997 et le 12/1997.
- Une requête paramétrée permettant l'affichage des commandes entre une date de début et une date de fin.

6. Créez une requête permettant l'affichage du nombre de commandes par client au Maroc.

7. Créer une requête permettant l'affichage de tous les clients étrangers.

8. Créer la requête qui permet de sélectionner les clients dont le pays est le Royaume-Uni ou dont la ville est Paris.

9. Créez une requête permettant l'affichage de l'entête d'une commande c'est-à-dire (l'ensemble des informations de la commande et le client qui a passé la commande).

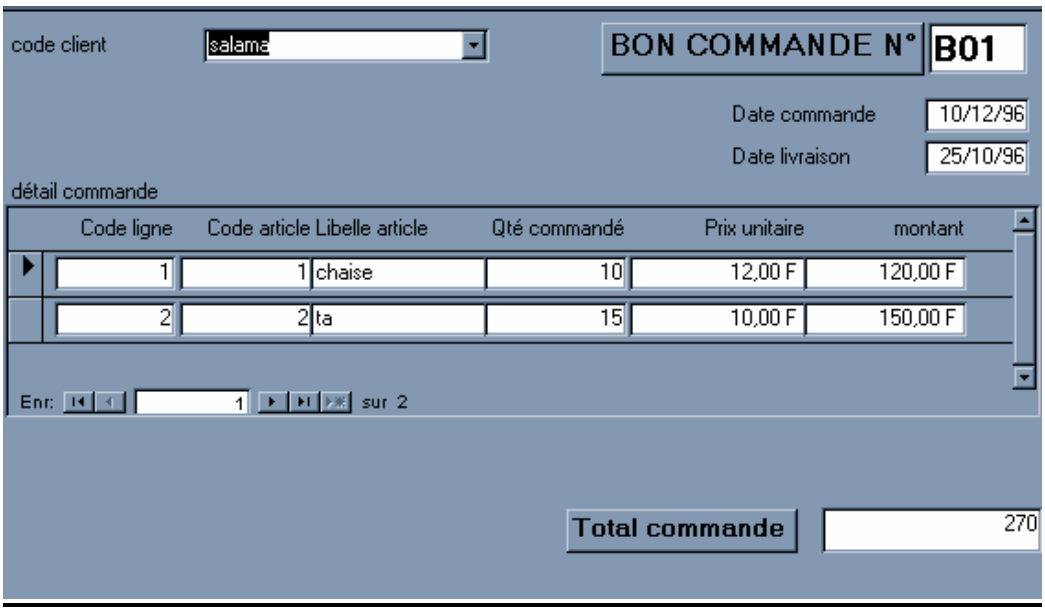
10. Créez une requête permettant l'affichage du détail de la commande C'est-à-dire pour chaque ligne commande (code ligne commande, libellé article, prix unitaire, quantité commandée et une expression de calcul permettant de calculer le montant de ligne commande).

11. Créez une requête permettant l'affichage du pied de commande c'est-à-dire pour chaque commande, le montant total des lignes commande.

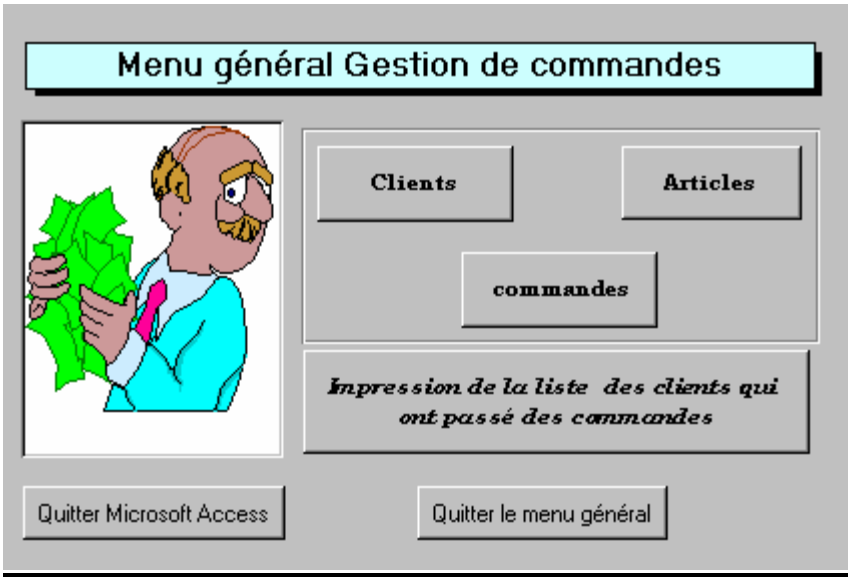
12. Créer un formulaire qui permet l'affichage des informations de la table article en accordant une remise de prix pour chaque article de 0,75.

13. Créer le formulaire qui permet d'afficher la liste des bons de commandes par client.

14. Créez un formulaire permettant l'affichage d'une commande.



15. Créer un état qui permet l'affichage de la liste alphabétique des clients.
16. Créer l'état qui permet de regrouper les commandes livrées par semaine.
17. Créer l'état qui permet l'affichage des informations de la tables article.
18. Créer l'état qui permet d'afficher la liste des clients qui ont passé une commande



19. Créer le formulaire suivant qui permet la gestion des commandes.

PARTIE 6 EXERCICES D'APPLICATIONS (LANGAGE SQL)

Exercice 1 :

Select nom_éditeur, ville, région
from éditeurs

```
nom_éditeur                            ville                région
-------------------------------------- -------------------- ------
New Moon Books                         Boston               MA
Binnet & Hardley                       Washington           DC
Algodata Infosystems                   Bruxelles            (null)
Five Lakes Publishing                  Chicago              IL
Ramona, éditeur                        Lausanne             (null)
GGG&G                                  Munich               (null)
Scootney Books                         New york             NY
Editions Lucerne                       Paris                (null)
(8 ligne(s) affectée(s))
```

Exercice 2 :

SELECT nom_employé, pn_employé, date_embauche, position_employé
FROM employé
WHERE (nom_employé LIKE 'L%') AND (date_embauche LIKE '%1990%') AND (position_employé BETWEEN 10 AND 100)

```
nom_employé                        pn_employé           date_embauche     position_employé
---------------------------------- -------------------- ----------------- ----------------
Lincoln                            Elizabeth            7/déc./1990  0:00           35
(1 ligne(s) affectée(s))
```

Exercice 3 :

SELECT nom_employé, id_éditeur
FROM employé
ORDER BY id_éditeur, nom_employé

```
nom_employé                    id_éditeur
------------------------------ ----------
Brognon                        0736
Ibsen                          0736
Josephs                        0736
Karttunen                      0736
Lebihan                        0736
Mendel                         0736
O'Rourke                       0736
Saveley                        0736
Snyder                         0736
…etc … (43 ligne(s) affectée(s))
```

Exercice 4 :

SELECT nom_auteur, pays, adresse
FROM auteurs
WHERE pays IN ('FR','CH', 'BE')

```
nom_auteur                             pays adresse
-------------------------------------- ---- ----------------------------------------
Lorense                                BE   14, impasse Lacarte
Schildwachter                          BE   11, rue Buffon
Bec                                    BE   4, chemin de la Tour de Campel
```

```
Posey                    CH   57, avenue des tapis
Sorense                  CH   55, rue Pierre-Louis Coll
Chevalier                CH   48, rue de Valmy
Chevalier                CH   48, rue de Valmy
…etc… (20 ligne(s) affectée(s))
```

Exercice 5 :

```
SELECT position_employé, count(*), MIN(date_embauche),MAX(date_embauche)
FROM employé
GROUP BY position_employé
```

```
position_employé
--------------- ----------- ------------------------- -------------------------
32              1           6/nov./1989  0:00         6/nov./1989  0:00
35              3           9/nov./1989  0:00         8/déc./1992  0:00
64              1           7/juil./1992  0:00        7/juil./1992  0:00
75              3           1/janv./1990  0:00        5/déc./1992  0:00
78              1           9/déc./1988  0:00         9/déc./1988  0:00
80              1           11/nov./1993  0:00        11/nov./1993  0:00
..etc…  (31 ligne(s) affectée(s))
```

Exercice 6 :

```
SELECT id_titre, MAX(droits)
FROM droits_prévus
GROUP BY id_titre
```

```
id_titre
-------- -----------
BU1032   12
BU1111   24
BU2075   24
BU7832   24
MC2222   20
MC3021   24
..etc… (16 ligne(s) affectée(s))
```

Exercice 7 :

```
select ville, nom_éditeur, count(*)
from éditeurs
GROUP BY ville, nom_éditeur
HAVING ville LIKE '%L%' OR  ville LIKE '%B%'
```

```
ville              nom_éditeur
------------------ ---------------------------------------- -----------
Boston             New Moon Books                           1
Bruxelles          Algodata Infosystems                     1
Lausanne           Ramona, éditeur                          1
(3 ligne(s) affectée(s))
```

Exercice 8 :

```
SELECT droits, id_titre,  minimum, maximum
FROM  droits_prévus
ORDER BY droits
```

```
droits      id_titre minimum     maximum
----------- -------- ----------- -----------
10          BU1032   0           5000
10          PC1035   0           2000
10          BU2075   0           1000
10          PS2091   0           1000
….etc…
           count
            ========
            16
```

```
droits     id_titre minimum     maximum
---------- -------- ----------- ----------
12         BU1032   5001        50000
12         PC1035   2001        3000
12         BU2075   1001        3000
12         PS2091   1001        5000
..etc…
           count
           =========
           16
..etc… (94 ligne(s) affectée(s))
```

Exercice 9 :

```
SELECT nom_auteur, titre, prix
FROM  auteurs a, titres t, titreauteur ta
WHERE (ville = 'PARIS') AND (a.id_auteur = ta.id_auteur) AND (ta.id_titre = t.id_titre)
```

```
nom_auteur           titre                          prix
------------------------------------ ------------------------------------
Mathieu    Guide des bases de données du gestionnaire pressé          140,00
Mathieu    Le stress en informatique n'est pas une fatalité !         24,00
Merrell    Toute la vérité sur les ordinateurs                        136,00
Jalabert   Phobie et passion informatique : éventail de comportements 147,00
(4 ligne(s) affectée(s))
```

Exercice 10

```
SELECT  nom_éditeur, nom_auteur, titre, qt
FROM  auteurs a,titreauteur ta, titres t, éditeurs e, ventes v
WHERE (a.id_auteur = ta.id_auteur) AND (ta.id_titre = t.id_titre) AND (t.id_éditeur = e.id_éditeur)
     AND (t.id_titre = v.id_titre)
ORDER BY nom_éditeur
COMPUTE SUM(qt) BY nom_éditeur
```

Exercice 11

```
SELECT nom_auteur, SUM(qt)
FROM  auteurs a,titreauteur ta, titres t, ventes v
WHERE (a.id_auteur = ta.id_auteur) AND (ta.id_titre = t.id_titre) AND (t.id_titre = v.id_titre)
GROUP BY nom_auteur
HAVING SUM(qt) > 20
```

Exercice 12

```
SELECT nom_auteur, pn_auteur
FROM   auteurs a
WHERE  id_auteur IN (SELECT id_auteur FROM titreauteur)    /* l'auteur doit avoir écrit au moins
un livre */
AND    100 = ALL (SELECT droits_pourcent
             FROM titreauteur
             WHERE a.id_auteur = id_auteur)
ORDER BY nom_auteur
```

```
nom_auteur                              pn_auteur
--------------------------------------- --------------------
Bec                                     Arthur
Bourne                                  Stéphanie
Chartier                                Laurent
Merrell                                 Patricia
Sorense                                 Christophe
Vilc                                    Benjamin
Vue                                     Jessica
(7 ligne(s) affectée(s))
```

Exercice 13 :

```
SELECT titre, prix
FROM   titres
WHERE  prix =  (SELECT MAX (prix)
                FROM titres )
```

```
titre                                                                prix
-------------------------------------------------------------------- ------
Est-ce vraiment convivial ?                                          156,00
(1 ligne(s) affectée(s))
```

Exercice 14 :

Afficher la liste des titres et le cumul de leurs ventes, tous magasins confondus, classés par ordre croissant des ventes

```
SELECT titre, somme = (SELECT SUM (qt)
                       FROM ventes v
                       WHERE t.id_titre = v.id_titre)
FROM titres t
ORDER BY somme
```

```
Titre                                                                                somme
-------------------------------------------------------------------------------- -----------
La psychologie des ordinateurs de cuisine                                        (null)
Guide des bonnes manières sur un réseau                                          (null)
Les festins de Parly 2                                                           10
Guide des bases de données du gestionnaire pressé                                15
Toute la vérité sur les ordinateurs                                              15
Privation durable d'informations : étude de quatre cas représentatifs            15
Phobie et passion informatique : éventail de comportements                       20
Cinquante ans dans les cuisines de l'Elysée                                      20
La cuisine japonaise - la portée de tous                                         20
La cuisine - l'ordinateur : bilans clandestins                                   25
Vivre sans crainte                                                               25
Equilibre émotionnel : un nouvel algorithme                                      25
Est-ce vraiment convivial ?                                                      30
Le stress en informatique n'est pas une fatalité !                               35
Les micro-ondes par gourmandise                                                  40
Oignon, poireau et ail : les secrets de la cuisine méditerranéenne               40
Les secrets de la Silicon Valley                                                 50
La colère : notre ennemie ?                                                      108
(18 ligne(s) affectée(s))
```

Exercice 15 :

Afficher le titre du livre le plus vendu de tous les magasins, et le nom de ce magasin

```
SELECT titre, nom_mag
FROM titres t, ventes v, magasins m
WHERE t.id_titre = v.id_titre
AND   v.qt =  ( SELECT MAX(qt)
                FROM ventes)
AND   v.id_mag = m.id_mag
```

```
titre                                                nom_mag
-------------------------------------------------- ----------------------------------------
```

La colère : notre ennemie ? Librairie spécialisée

Exercice 16 *:*
Rentrez vos noms, prénoms, dans la table auteurs, avec un identificateur qui n'existe pas déjà
insert auteurs (id_auteur, nom_auteur, pn_auteur, contrat)
values ('100-00-1020', 'Lécu', 'Régis', 1)

Exercice 17 :

Recopier toutes les caractéristiques d'un auteur en lui donnant un nouvel identificateur, et un nouveau nom

INSERT auteurs (id_auteur, nom_auteur, pn_auteur, téléphone, adresse, ville, pays, code_postal, contrat)
SELECT '100-00-1200', 'toto', pn_auteur,téléphone, adresse, ville, pays, code_postal, contrat
FROM auteurs
WHERE nom_auteur = 'Bourne'

Exercice 18 :

Augmenter de 10% tous les prix des livres de l'éditeur « Algodata Infosystems». Vérifier l'opération par une commande Select adéquate avant et après l'augmentation.

```
SELECT titre, prix
FROM   titres t, éditeurs e
WHERE  t.id_éditeur = e.id_éditeur
AND    nom_éditeur = "Algodata Infosystems"
AND    prix is not NULL

UPDATE titres
SET prix = 1.10 * prix
WHERE prix is not NULL
AND id_éditeur = (SELECT id_éditeur
            FROM  éditeurs
            WHERE  nom_éditeur = "Algodata Infosystems")

SELECT titre, prix
FROM   titres t, éditeurs e
WHERE  t.id_éditeur = e.id_éditeur
AND    nom_éditeur = "Algodata Infosystems"
AND    prix is not NULL
```

Exercice 19 *: Détruire les lignes crées dans la tables auteur, dans les exercices 16 et 17*

```
DELETE auteurs
FROM auteurs
where nom_auteur ='Lécu'
```

PARTIE 7 EVALUATION

Epreuve pratique

Durée 4 heures
Barème : /40

La compagnie BALADI de transport envisage d'intégrer dans son site web la gestion des réservations et paiement en ligne des voyages planifiés. A cet effet, le directeur de la compagnie veut implanter au préalable la nouvelle gestion des réservations et paiement en ligne sous Access et par la suite la migrer vers SQL Server

Le MLD relationnel de la base de donnée se présente ainsi :

Client (***CINClient***,NomClient, PrénomClient, TelClien)t
Billet (***NumBillet***, CINClient, CodeVoyage, DateBillet, NumPlace, Reéglé, EtatPlace)
Voyage (***CodeVoyage***,HeureDepartVoyage, HeureArrivéVoyage , VilleArrivéVoyage, PrixVoyage)
Place (***NumPlace***)

Structure de la base de donnée

Table Client

Nom de champ	Signification	Type	Taille/Format
CINClient	***CIN de Client***	Texte	10
NomClient	Nom de Client	Texte	20
PrénomClient	Prénom de Client	Texte	15
TelClien	Téléphone de Client	Texte	12

Table Billet

Nom de champ	Signification	Type	Taille/Format
NumBillet	Numéro de Billet	Numérique	Entier long
CINClient	CIN Client	Texte	10
CodeVoyage	Code Voyage	Numérique	Entier
DateBillet	Date de Billet	Date et heure	Abrégé
NumPlace	Numéro de Place	Numérique	Octet
Réglé	Reglé	Oui/Non	
EtatPlace	Etat de Place	Texte	7

Table Voyage

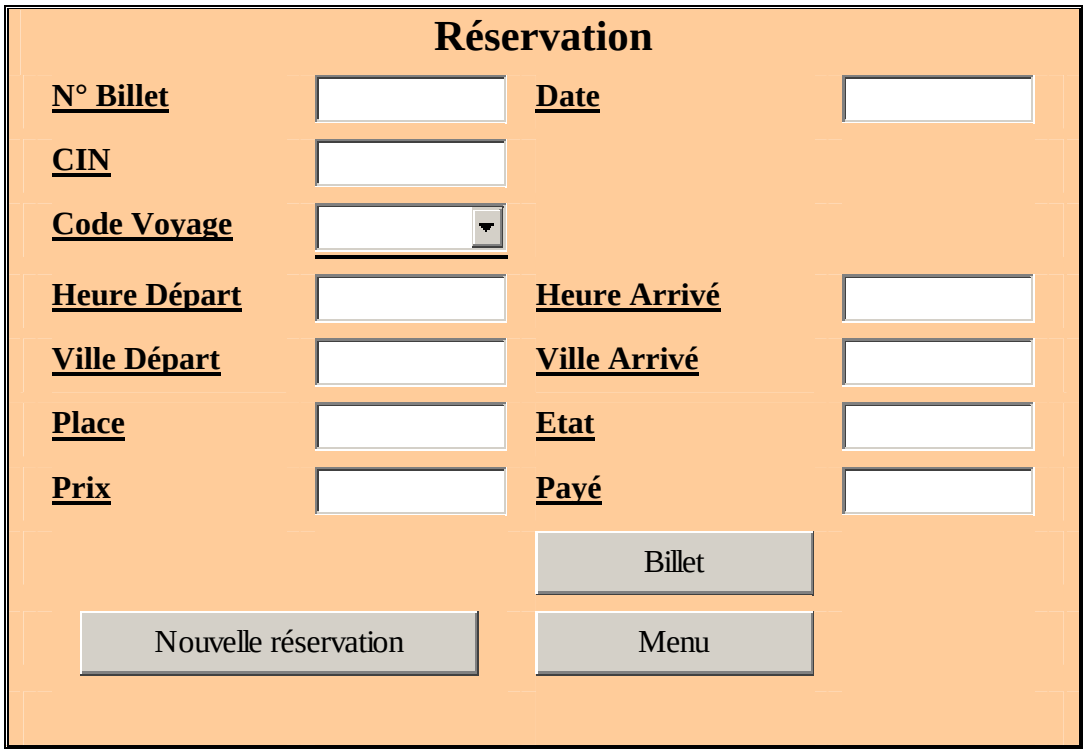
Nom de champ	Signification	Type	Taille/Format
CodeVoyage	***Code de Voyage***	Numerique	Entier
HeureDepartVoyage	Heure de Départ de Voyage	Date et heure	Heure Abrégé
Ville_Départ	Ville de départ de voyage	texte	15
HeureArrivéVoyage	Heure d'Arriver de Voyage	Date et heure	Heure Abrégé
VilleArrivéVoyage	Ville d'Arriver de Voyage	Texte	15
PrixVoyage	Prix de Voyage	Monetaire	Dh

Table Place

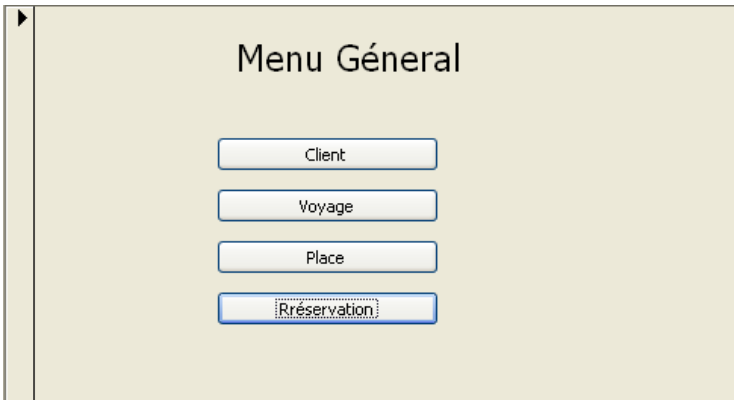
Nom de champ	Signification	Type	Taille/Format
NumPlace	Numéro de la place	Numérique	octet

Travail demandé

1. Pour chaque table ci-dessus, créer sa structure. Utiliser les mêmes clés primaires indiquées dans le MLD ci-dessus et créer les relations entrer ces tables (4 Pts)
2. Saisie les enregistrements donnés dans l'annexe ci-dessus pour chaque table de la base donnée. (2 Pts)
3. Créer une requête qui affiche les informations sur un client en introduisant son CIN et l'entreprise sous R_CIN_client (3 Pts)
4. Créer une requête qui affiche les informations sur un voyage planifié en introduisant la ville de départ et la ville d'arriver puis l'enregistrer sous R_Client_CIN (3 Pts)
5. Créer un formulaire base sur la requête R_Client_CIN et l'enregistrer sous F_Client_CIN (3 Pts)
6. Créer une requête qui affiche les informations sur le billet + voyage et l'enregistrer sous R_Réservation (3 Pts)
7. Créer un formulaire base sur la requête R_voyage et l'enregistrer sous F_Voyage (3 Pts)
8. Créer un formulaire base sur la table voyage (Ajouter/modifier/Supprimer) et l'enregistrer sous F_Voyage (3 Pts)
9. Créer un formulaire base sur la requête R_Réservation et l'enregistrer sous F_Réservation comme est montre dans la figure suivante :(10 Pts)



- Le bouton affiche les information sur un voyage a choisir et remplir les champ de voyage dans ce formulaire
- Le bouton nouvelle réservation vide tous le champs
- Le bouton billet imprime le billet de la place réservé
- Le bouton menu retourne vers le formulaire ci-dessus



10. Créer l'état qui fait sortir des billes payées par un client donné dans une date de voyage comme le montre la figure suivante et l'enregistrer sous E_billets et le lier au bouton « Billet » du formulaire du Réservation. (6 Pts)

BALADY TRANSPORT

N° : 0012500254 Date :01/01/06

Heure Départ	**Ville Départ**	**Heure Arrivé**	**Ville Arrivé**	**Place**	**Prix**
4 :00	Casa	10 :00	Agadi	3	350.00

Annexe

Formalisme MERISE

Concevoir une base de données relationnelle, c'est établir pour le système d'information étudié, les relations entités et les relations associations en troisième forme normale.

1ère étape :

Etablir les schémas externes, c'est-à-dire lister les données nécessaires à chaque utilisateur de la future base.

2ème étape :

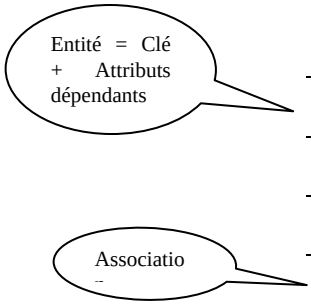
Etablir le dictionnaire de données en regroupant les schémas externes, en supprimant les redondances et en ne conservant que les informations élémentaires (non déduites). Ceci revient à lister les attributs de la base.

Dictionnaire de données du système d'informations relatif aux tests sur les matériels de production :

code matériel, *libellé matériel*
code marque, *libellé marque*
code type de test, *libellé du test*, *date du test*, *résultat du test*

3ème étape :

Etablir les contraintes d'intégrité fonctionnelle (ou dépendances fonctionnelles) entre attributs.



Attributs	**En dépendance fonctionnelle avec**
Code matériel	
Libellé matériel…..……	*code matériel*
Code marque	
Libellé marque……	*code marque*
Code type de test	
Libellé du test……..	*code type de test*
Date du test……..	*code matériel + type de test*
Résultat du test……..	*code matériel + type de test*

4ème étape :

En déduire les relations "entités" et les relations "associations avec attributs" :

- Les entités sont constituées d'une clé primaire et d'un ou plusieurs attributs qui ne dépendent fonctionnellement que de cette clé
- Les associations sont constituées d'une liste d'au moins deux clés représentant des entités, et d'attributs qui dépendent de ces clés

Entités : *Matériel*, *Marque*, *Type de test*
Association avec attributs *: Test*

5ème étape :

Etablir les relations "associations sans d'attributs" en considérant deux cas :

- Il existe un lien fonctionnel N : 1 entre les entités : la clé primaire de l'entité mère devient clé étrangère dans l'entité fille

Exemple: matériel-marque. L'entité "Matériel" dépend (est fille) de l'entité "Marque" : la clé étrangère "code marque" dans "Matériel" pointe sur la clé primaire "code marque" dans "Marque" .

- Le lien entre les deux entités est de type N:M : il faut créer une nouvelle relation association sans attributs, qui contient seulement les clés primaires des deux relations associées.

6ème étape:

Représenter le schéma de la base

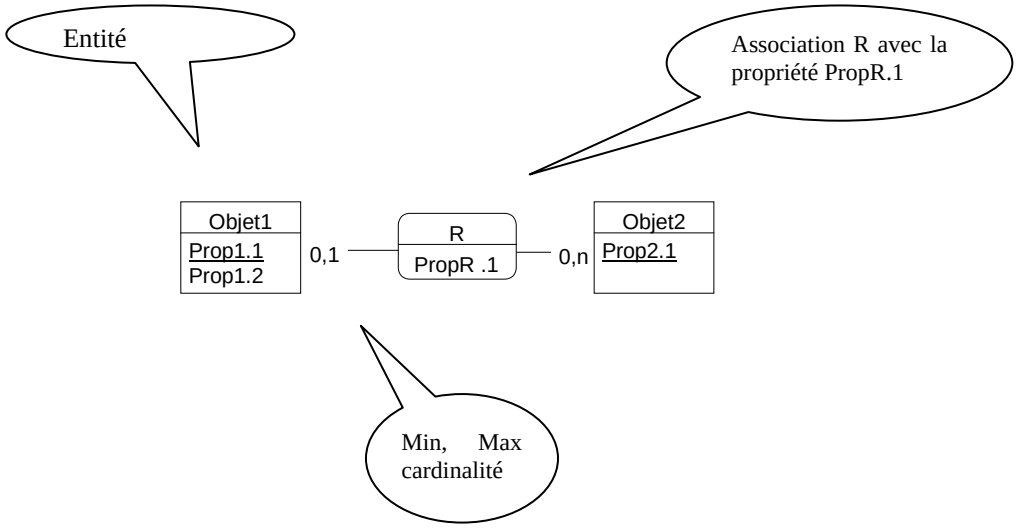
7ème étape:

S'assurer que les relations sont en troisième forme normale.

Les phases de la conception avec un symbolisme de type « merise »

Présentation

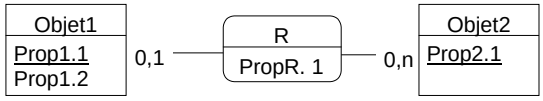
- Sans faire un exposé sur la méthode Merise, cet Annexe chapitre voudrait présenter succinctement les différents modèles des données (MCD Conceptuel, MLD Logique, MPD Physique), à titre de comparaison avec la méthode « maison » qui vient d'être exposée.
- Après avoir recueilli les données auprès des clients (étape 1), supprimé les redondances (étapes 2), classé les données selon les dépendances fonctionnelles (étape 3), on construit le modèle conceptuel « entités/associations » (étape 4), où les associations sont des relations valuées, comportant un ou plusieurs attributs :



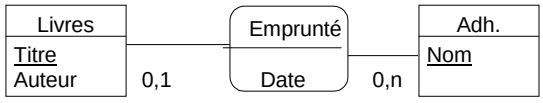
- A partir du modèle conceptuel, on peut déduire le modèle logique et physique par des opérations systématiques (étape 5 et 6) : l'application complète de la méthode Merise garantit l'obtention des formes normales. Le passage du MCD au MLD, puis au MPD dépend de la cardinalité des relations.

Du MCD au MPD, pour un lien fonctionnel (N :1) ou hiérarchique (1 :N)

M.C.D



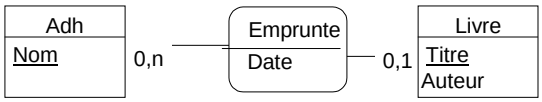
- ***Exemple :***



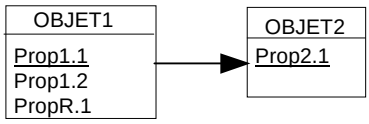
- *L'association* ***« est emprunté le »*** *contient la propriété* ***« Date »***
- ***Cardinalité :*** *Un livre peut être emprunté 0 ou une fois (min = 0, max = 1 dans la notation Merise). Un adhérent peut emprunter de 0 à N livres (min= 0, max= N)*

⇨ *lien fonctionnel N : 1 dans la notation ANSI-SPARC*

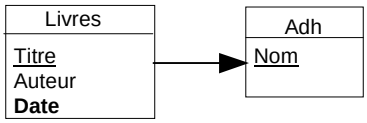
- La relation inverse **« emprunte le »** est une relation hiérarchique 1 :N



M.L.D



Règle **:** les propriétés de l'association glissent du côté 0-1, la flèche pointe vers le côté 0,n



Dans la table Livres, on ajoute la date de l'emprunt, et une flèche vers l'adhérent emprunteur

MPD

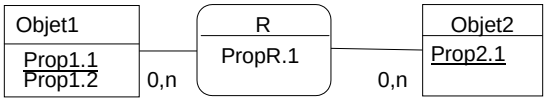
Objet1 (Prop1.1, Prop1.2, PropR.1, Prop2.1)
Objet2 (Prop2.1)

Règle : une clé étrangère Prop2.1 pointant sur objet2.Prop2.1 est ajoutée à objet1

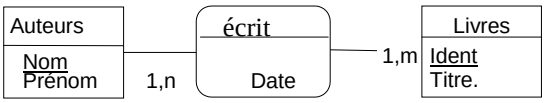
Livres *(Titre, Auteur, Date, Nom)*
Adhérents *(Nom)*
Dans la relation Livres, on ajoute la clé étrangère « Nom », pointant sur « Adherents.Nom »

Du MCD au MPD pour une Relation 0-N : 0-N

M.C.D

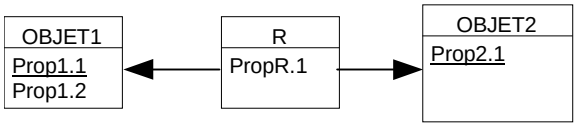


- ***Exemple :***

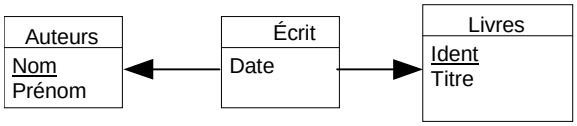


- *L'association* ***"écrit"*** *contient la propriété* ***"Date"*** *(de parution)*
- ***Cardinalité de la relation :*** *Un auteur peut écrire de 1 à n livres. Un livre peut avoir de 1 à m auteurs => lien maillé N : M dans la notation ANSI-SPARC*

M.L.D



Règle **:** l'association devient une nouvelle table et les flèches pointent vers les tables liées



On crée une nouvelle table « écrit », avec la propriété « Date », et des liens vers les clés des entités « Auteurs » et « Livres »

MPD

Objet1 (Prop1.1, Prop1.2)
R1 (Prop1.1, Prop2.1, PropR.1)
Objet2 (Prop2.1)

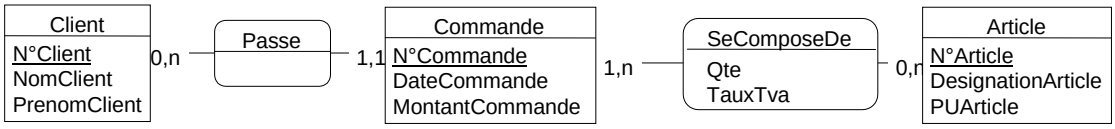
Règle **:** la relation devient une table dont la clé est la concaténation des clés des deux objets liés.

Exemple :

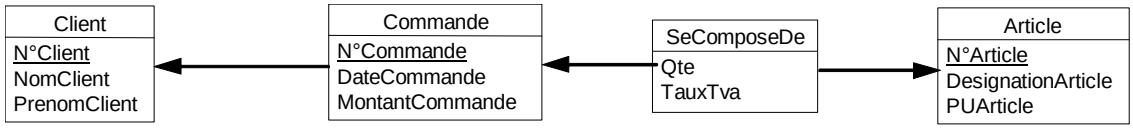
Livres *(Ident, Titre)*
Ecrit (Nom, Ident, Date)
Auteurs *(Nom, Prénom)*

Exemple récapitulatif

MCD



MLD

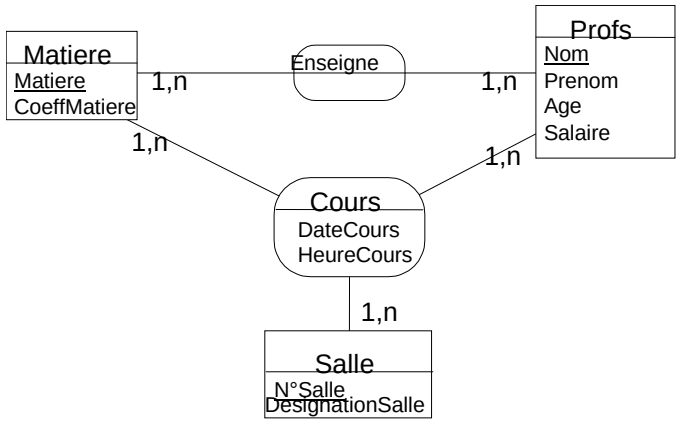


MPD

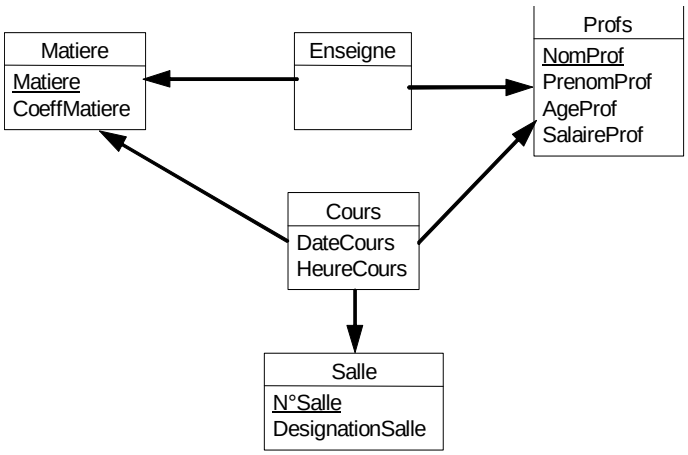
Client (N°Client, NomClient, PrenomClient).
Commande (N°Commande, DateCommande, MontantCommande, N°Client).
SeComposeDe (N°Commande, N°Article, Qte, TauxTva).
Article (N°Article, DesignationArticle, PUArticle).

Du MCD au MPD, dans une Relation ternaire

MCD



MLD



MPD

Matiere (Matiere, CoeffMatiere).
Enseigne (NomProf, Matiere).
Profs (NomProf, PrenomProf, AgeProf, SalaireProf).
Cours (NomProf, Matiere, N°Salle, DateCours, HeureCours).
Salles (N°Salle, DesignationSalle)

Règle : une relation ternaire devient une table dont la clé est la concaténation des clés des trois objets liés.