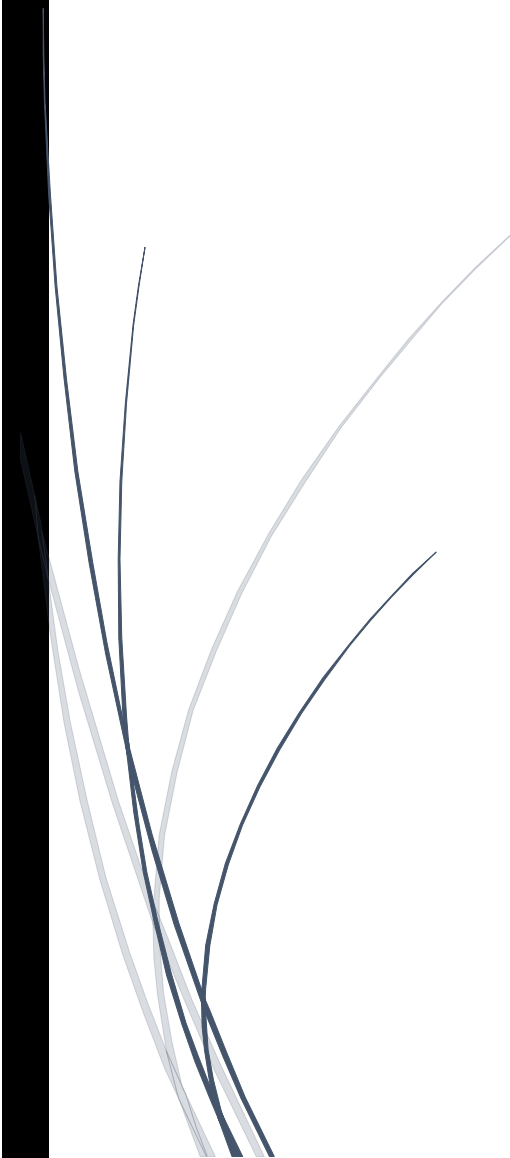




MAJ le : 13/03/2022

BASE DE DONNEES

Support de cours



Enseignant : ZINSOU Mensah
Contact : 97478114 / 95714195
E-mail : zinsmensah@yahoo.fr

CHAPITRE 1 : Généralités sur les bases de données

INTRODUCTION

Dans les formes traditionnelles de l'informatique, les données sont stockées sur des supports constituant les fichiers. Ces fichiers sont construits pour répondre aux besoins des applications informatiques. Leur exploitation connaît des difficultés liées au fait qu'ils sont utilisés isolément et indépendamment les uns des autres ; ceci est d'une part coûteux en moyen de stockage de données et d'autre part pose le problème de cohérence entre les données car la mise à jour d'une même donnée localisée dans plusieurs fichiers utilisés, par des applications différentes, peut ne pas être simultanée.

Aujourd'hui, la plupart des utilisateurs ne veulent plus d'un tel système d'information constitué d'un ensemble de fichiers inflexibles et de données inaccessibles. Il existe aujourd'hui des systèmes de gestion de base de données basés sur des langages d'accès puissants.

Dans le déroulement de ce chapitre, nous essayerons de définir ce que c'est qu'une base de données et les différentes étapes à suivre pour sa mise en œuvre. On cherchera ensuite à connaître l'utilité des bases de données, ce que c'est qu'un système de gestion d'une base de données, ses principes et ses objectifs.

I- LES BASES DE DONNÉES

A- Qu'est-ce qu'une base de données ?

Une base de données (Data-base, en Anglais) est un ensemble structuré avec le moins de redondance possible des données organisées de façon à permettre leur exploitation.

Ces données doivent pouvoir être utilisées par des programmes, par des utilisateurs différents.

On parle généralement de système d'information pour désigner toute la structure regroupant les moyens mis en place pour pouvoir partager les données.

B- Utilité d'une base de données

Une base de données permet de mettre les données à la disposition des utilisateurs pour une consultation, une saisie ou bien une mise à jour tout en s'assurant des droits accordés à ce dernier. Cela est d'autant plus utile lorsque les données informatiques sont de plus en plus nombreuses.

L'avantage majeur de l'utilisation des bases de données est la possibilité de pouvoir être utilisée par plusieurs utilisateurs simultanément.

C- Système de gestion de base de données (SGBD)

Afin de pouvoir contrôler les données ainsi que les utilisateurs, le besoin d'un système de gestion de base de données s'est fait vite ressentir. La gestion se fait grâce à un système appelé Système de Gestion de Base de Données (SGBD). Une fois la base de données spécifiée, on peut y insérer les données, les récupérer, les modifier ou les détruire : c'est ce qu'on appelle manipuler les données.

D- Principaux objectifs d'un système de gestion de données

- Permettre l'accès efficace aux données de façon simple.
- Autoriser un accès aux données à plusieurs utilisateurs de manière simultanée.
- Manipuler les données présentes dans la base.
- Permettre le partage des données entre les programmes.
- Assurer la sécurité des données.
- Faciliter une administration centralisée des données.

II- MISE EN ŒUVRE D'UNE BASE DE DONNÉES

La mise en œuvre d'une base de données fiable et cohérente passe d'abord par l'une des phases importantes qu'est l'analyse des données (Confère cours de MERISE).

En effet, cette phase est indispensable car les résultats qui seront obtenus serviront de base pour la conception proprement dite de la base de données à mettre en place. Il faut donc éviter de bâcler cette étape !

Pour bien réussir cette phase, il existe une démarche classique à suivre. Cette démarche permettra d'aboutir à deux schémas importants : Le schéma conceptuel des données (SCD) et le schéma logique relationnel (SLR).

La démarche à suivre est la suivante :

- Élaborer le dictionnaire des données
- Élaborer la liste des dépendances fonctionnelles élémentaires et directes encore appelée couverture minimale.
- Représenter le schéma conceptuel des données correspondant.
- Dédire le Schéma Logique Relationnel (SLR) à partir de Schéma Conceptuel des Données (SCD).
- Présenter un Schéma Physique des Relationnel (SCD) correspondant.

CHAPITRE 2 : Base de données relationnelle : Etude du langage algébrique

INTRODUCTION

Un modèle de données est un ensemble de concepts et de règles permettant de décrire les données. Il existe généralement trois modèles de données à savoir : le modèle hiérarchique, le modèle réseau et le modèle rationnel.

C'est l'étude de ce dernier qui fera l'objet de ce chapitre. Il a été (modèle rationnel) inventé par **CODD** en **1970** et est basé sur de concepts très simples. Il lui est associé une théorie qui ne peut être séparée du modèle : c'est la théorie de la normalisation des relations. Cette théorie a pour but d'éliminer les données redondantes et de mieux comprendre la structure des données.

I- Les différents modèles de données

Le principal objectif est de rendre indépendant les données vis-à-vis des applications. L'accès à ces données par ces applications pose souvent un certain nombre de problèmes. Pour simplifier ou réduire ces problèmes d'accès, plusieurs modèles logiques de données ont vu le jour. On distingue par ordre chronologique :

A- Le Modèle Hiérarchique

Historiquement premier, il est constitué d'une structure simple à gérer. Dans cette structure, chaque élément n'a qu'un seul supérieur ; un tel modèle à un nombre de communication limitée. Il n'y a aucune connexion.

B- Le Modèle Réseau

Le modèle réseau est une extension du modèle précédent. Il permet d'établir les connexions entre les différents éléments. De cette manière, on dispose d'un plus grand nombre d'interrogations possible mais elles doivent être toujours prévues lors de la construction de la base de données.

C- Le Modèle Relationnel

Le modèle relationnel permet de se libérer de la contrainte suivante : connaître à l'avance les interrogations que l'on effectuera. Ainsi, les données sont stockées sous forme de relation dans les "tables" qui deviendront par la suite les fichiers. Ce type de structure permet d'établir des connexions au moment de l'exécution des requêtes. On pourra donc effectuer toute sorte d'interrogation plus ou moins complexe. L'accès aux bases de données relationnelles s'effectue en appliquant les trois opérations de base suivante : **la sélection, la jointure et la projection.**

II- Etude des dépendances fonctionnelles

Une dépendance fonctionnelle est un outil clé qui permet de déterminer les liens possibles qui existent entre les propriétés, de nature élémentaire du dictionnaire des données, afin de les structurer et d'éviter ainsi les redondances qui compliquent la mise à jour des bases de données.

On dit qu'une propriété **B** dépend fonctionnellement d'une autre propriété ou groupe de propriétés **S** si et seulement si, à une valeur de S correspond une seule valeur possible de B. On note :

S \rightarrow **B** et on lit : "S détermine B" ou bien "B dépend de S"
(Source) (But)

Exemples

L'expression **numcde** \rightarrow **datecde** signifie qu'à une valeur du numéro d'un bon de commande correspond toujours une seule valeur de la date de commande ; l'inverse n'est pas vrai car à une date donnée, on peut passer plusieurs commandes.

L'expression **numcde** \rightarrow **numcli** signifie qu'à une valeur du numéro de commande correspond une seule valeur du numéro de client ; l'inverse n'est pas toujours vrai car un client peut passer plusieurs commandes.

Dans l'expression *numcde, codprod* \rightarrow *qtecde*, la seule valeur du numéro d'un bon de commande ne peut déterminer la quantité de chaque produit figurant sur ce bon ; il en est de même pour celle du code d'un produit.

En effet, si la propriété *numcde* déterminait *qtecde* alors quel que soit le produit concerné par ce bon de commande, sa quantité aurait une même valeur ; ce qui n'est pas toujours vrai.

De même, si la propriété *codprod* déterminait *qtecde* alors quel que soit le bon de commande sur lequel se trouverait un produit quelconque, ce dernier aura une même quantité ; ce qui n'est pas toujours vrai.

Ainsi donc, la connaissance d'une quantité commandée de chaque produit figurant sur un bon de commande dépend à la fois du numéro de ce bon et du code dudit produit.

A- Quelques propriétés des dépendances fonctionnelles

Dans ce paragraphe, les lettres S, S1, S2, B, B1 et B2 désignent des propriétés.

1- La réflexivité

On a : $S \rightarrow S$ ou $S1, S2 \rightarrow S1$ ou $S1, S2 \rightarrow S2$

Exemple : $\text{numcli} \rightarrow \text{numcli}$

2- L'additivité

Si $S \rightarrow B1$ et $S \rightarrow B2$ alors $S \rightarrow B1, B2$

Exemple : $\text{numcli} \rightarrow \text{nomcli}$ et $\text{numcli} \rightarrow \text{prencli}$

alors $\text{numcli} \rightarrow \text{nomcli, prencli}$

3- La projection

Si $S \rightarrow B1, B2$ alors $S \rightarrow B1$ et $S \rightarrow B2$

4- La transitivité

Si $S \rightarrow S1$ et $S1 \rightarrow B$ alors $S \rightarrow B$

Exemple : numcde \rightarrow numcli or numcli \rightarrow nomcli

donc numcde \rightarrow nomcli

Remarque : dans le schéma modélisant la dépendance fonctionnelle entre deux ou plusieurs propriétés, la propriété ou groupe de propriétés "source" est toujours un identifiant. Il est soit simple soit composé.

B- Types de dépendances fonctionnelles

- Dépendance fonctionnelle élémentaire (dfe)

Considérons deux propriétés S et B.

La propriété B est en dépendance fonctionnelle élémentaire de S si B dépend de la totalité de S et non d'une partie de S.

On note : $S \xrightarrow{\text{dfe}} B$

Remarque : lorsqu'il s'agit d'une dépendance fonctionnelle telle que la source est constituée d'une seule propriété, cette dépendance fonctionnelle est forcément élémentaire.

- Dépendance fonctionnelle élémentaire et directe (dfed)

Considérons trois propriétés S, B1 et B2.

La propriété **B2** est en dépendance fonctionnelle élémentaire et directe de **S** si elle est d'abord en dépendance fonctionnelle élémentaire de S et s'il n'existe pas B2 en dépendance fonctionnelle de B1 et B1 en dépendance fonctionnelle de S ; en d'autres termes, on élimine la transitivité.

On note : $S \xrightarrow{\text{dfed}} B2$

Exemple : numcde $\xrightarrow{\text{dfed}}$ numcli

C- Formes normales

Toute relation issue d'une dépendance fonctionnelle doit respecter trois conditions pour être juste : *on parle de formes normales*. Elles ont pour objectif de décomposer des relations sans perdre d'information.

1- La première forme normale

Une relation R est en première forme normale si tous ses attributs (propriétés) sont élémentaires et si cette relation admet une clé (identifiant).

Exemple : soit la relation R1 (numcli, nomcli, prenci, sexcli) dont les éléments sont tous élémentaires.

La propriété numcli est la clé de cette relation telle que numcli \rightarrow nomcli, prenci, sexcli ; on peut donc dire que la relation R1 est en première forme normale.

2- La deuxième forme normale

Elle permet d'assurer l'élimination de certaines redondances en garantissant qu'aucun attribut n'est déterminé seulement que par une partie de sa clé.

Une relation R est en deuxième forme normale si et seulement si :

- Elle est d'abord en première forme normale.
- Tout attribut "non clé" ne dépend d'une partie de la clé mais plutôt de sa totalité.

Remarque : Toute relation qui est en 1^{ère} forme normale et dont la clé est simple est considérée comme une relation en 2^{ème} forme normale.

On peut donc dire que la relation R1 est en 2^{ème} forme normale.

3- Troisième forme normale

La 3^{ème} forme normale permet d'assurer l'élimination des redondances dues aux dépendances transitives.

Une relation R est en 3^{ème} forme normale si et seulement si elle est d'abord en 2^{ème} forme normale et tout attribut non clé ne dépend d'un autre attribut non clé.

Applications : (confère planche d'exercices)

III- ÉTUDE DU LANGAGE ALGEBRIQUE

Le langage algébrique est un outil préalable au niveau opérationnel des traitements permettant de formaliser les opérations qui peuvent être réalisées sur des relations définies au niveau logique.

Bien formuler l'enchaînement des opérations à effectuer sur les relations, permet de préparer la manipulation des données présentes dans la base relationnelle.

A- Qu'est-ce qu'une relation ?

Une relation permet de décrire des entités et des associations par un ensemble d'attributs. Sa syntaxe est :

Nom_RELATION (Clé primaire, Attribut(s), #clé(s) Etrangère(s))

Remarque : une relation peut être présentée sous forme d'une table contenant un ensemble d'enregistrements. Dans cette table, chaque colonne correspond à un attribut de l'objet décrit ; chaque ligne de cette table correspond à un enregistrement (ou tuple).

Exemple : Considérons la table COMMANDE relative à la relation COMMANDE du SLR précédent.

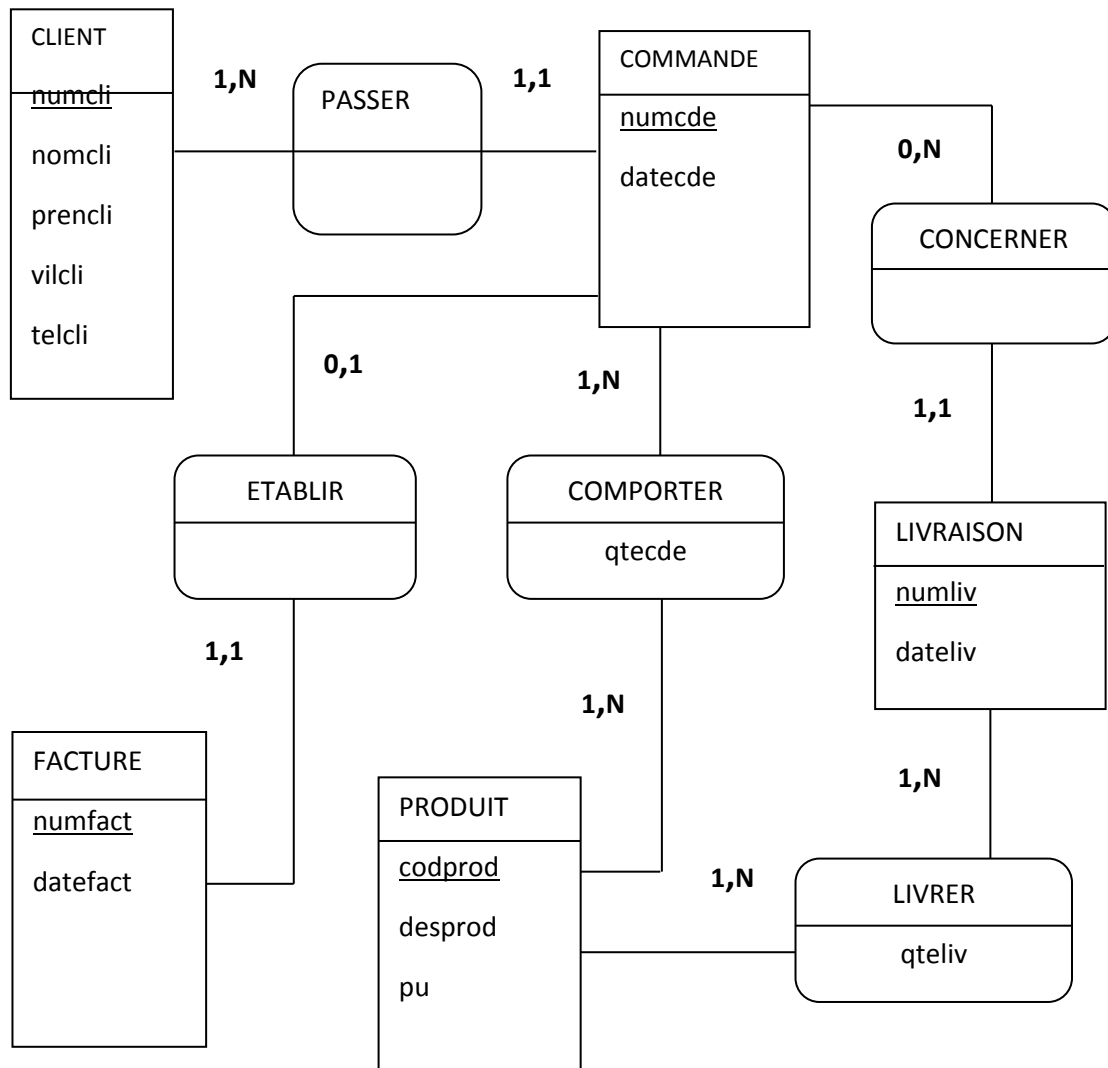
On a :

Table : COMMANDE

numcmde	datecmde	numcli
1425	04/11/2020	250
1426	04/11/2020	145
4241	07/12/2020	167

EXERCICE

On vous donne le Schéma Conceptuel des Données suivant :



Travail à faire

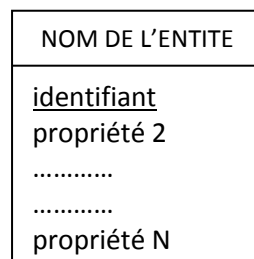
- 1- Indiquer le nombre d'entité, le nombre d'association, d'association binaire, d'association ternaire, d'association porteuse, d'association hiérarchique, association non-hiérarchique, CIF et CIM que comporte le schéma.
- 2- Présentons le tableau de justification des cardinalités.
- 3- Dédurre le Modèle Logique Relationnel des Données correspondant.
- 4- Présenter, dans un tableau, le degré, la clé primaire et la (les) clé(s) étrangère(s) de chacune des relations du MLRD obtenu

RESOLUTION :

Rappel 1 :

Sur un schéma conceptuel des données (SCD) on observe deux types d'objets : les entités et les associations.

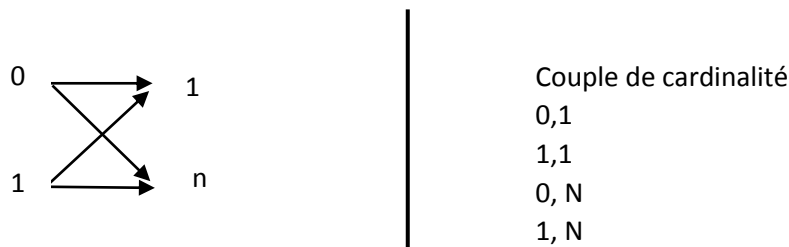
Une entité est un objet concret ou abstrait décrit par un ensemble de propriétés dont l'une est son identifiant qui peut être soit un numéro, un code ou une référence. Son formalisme est le suivant :



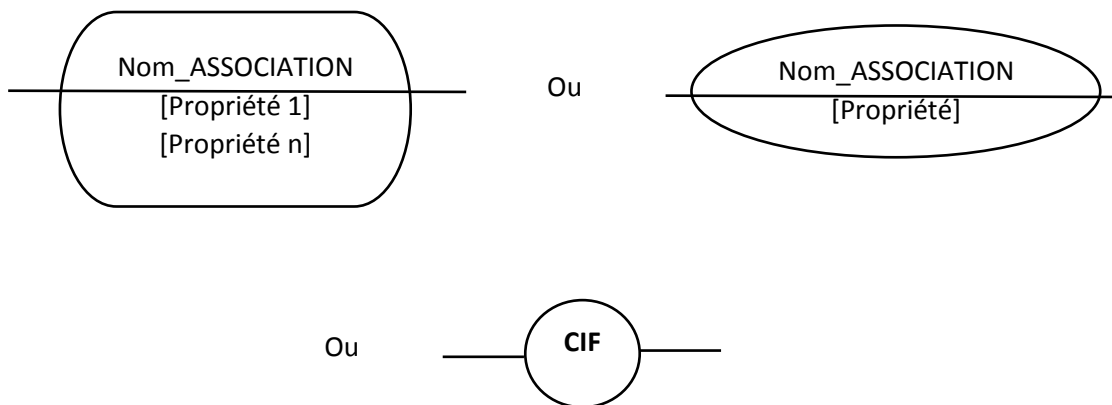
Une association est un lien sémantique défini entre deux ou plusieurs entités. Son nom est généralement un verbe à l'infinitif. Elle est dite binaire lorsque relie exactement deux entités ; ternaire lorsque relie exactement trois entités, N-aires lorsqu'elle relie plus de trois entités etc. Elle peut être porteuse de propriété(s) ou non.

Sur les branches reliant une entité à une association on observe des cardinalités. Il existe deux types de cardinalité : la cardinalité minimum (0 ou 1) et la cardinalité maximum (1 ou n).

Les différents types de cardinalités (CardMin, CardMax) possibles sont :



Le formalisme d'une association est le suivant :



Une association est dite hiérarchique lorsqu'elle porte sur l'une de ses branches le type de cardinalité 1,1 ou 0,1. Elle n'est jamais porteuse de propriété. On la considère comme une contrainte d'intégrité fonctionnelle (CIF).

Une CIM est une association non hiérarchique. Elle peut être porteuse de propriété(s) ou non.

1- Dénombrement

On a :

Élément	Nombre
Entité	05
Association	05
Association binaire	05
Association ternaire	00
Association porteuse	02
Association hiérarchique	03
Association non hiérarchique	02
CIF	03
CIM	02

Rappel 2 :

2- Justification des cardinalités

Association	Entité	Cardi- nalité	Justification
PASSER	CLIENT	1,N	Un client passe au moins une commande
	COMMANDE	1,1	Une commande est passée par un seul client
CONCERNER	COMMANDE	0,N	Une commande peut faire l'objet d'aucune ou de plusieurs livraisons
	LIVRAISON	1,1	Une livraison ne concerne qu'une seule commande
ETABLIR	COMMANDE	0,1	Une commande fait l'objet d'au plus une facture
	FACTURE	1,1	Une facture est établie pour une seule commande
COMPORTER	COMMANDE	1,N	Une commande comporte au moins un produit
	PRODUIT	1,N	Un même produit, d'une quantité donnée, figure sur au moins un bon de commande
LIVRER	LIVRAISON	1,N	une livraison peut faire l'objet d'au moins un produit
	PRODUIT	1,N	Un même produit, d'une quantité donnée, figure sur au moins un bon de livraison

Rappel 3 :

Règles de passage du MCD au MLDR

R1 : Toute entité du MCD devient, donne son nom à une relation du MLDR. L'identifiant de cette entité devient la clé primaire et les propriétés de l'entité deviennent les attributs.

Remarque : les entités n'ayant que leur identifiant comme élément ne sont pas concernées par cette règle.

R2 : Toute association hiérarchique ou CIF ou association Père-Fils induit la création d'une clé étrangère dans la relation de "l'objet fils". Cette clé étrangère, qui n'est rien d'autre que l'identifiant de "l'objet père", sera précédée du symbole dièse (#).

Remarque : dans le cas d'une association hiérarchique portant sur ses deux branches les types de cardinalités 1,1 et 0,1, la clé étrangère sera créée dans la relation de l'entité ayant pour cardinalité 1,1.

R3 : Toute association non hiérarchique du MCD donne son nom à une relation du MLDR. La clé primaire de cette relation sera formée par concaténation (c'est-à-dire par juxtaposition) des identifiants des entités reliées à cette association. Chaque composant de cette clé formée sera précédé du symbole dièse (#), sauf dans le cas où ce composant est issu d'une entité n'ayant que son identifiant comme élément. Les propriétés éventuellement portées par cette association deviennent les attributs de la relation obtenue.

3- Déduisons le MLDR correspondant au MCD

En appliquant de façon cumulative les règles de passage du MCD au MLDR,

On a :

CLIENT (numcli, nomcli, prencli, vilcli, telcli)

COMMANDE (numcde, datecde, #numcli)

LIVRAISON (numliv, dateliv, #numcde)

FACTURE (numfact, datefact, #numcde)

PRODUIT (codprod, desprod, pu)

COMPORTER (#numcde, #codprod, qtecde)

LIVRER (#numliv, #codprod, qteliv)

4- Caractéristiques des relations

Relation	Degré	Clé primaire	Clé étrangère
CLIENT	5	numcli	-
COMMANDE	3	numcde	numcli
LIVRAISON	3	numliv	numcde
FACTURE	3	numfact	numcde
PRODUIT	3	codprod	-
COMPORTER	3	numcde + codprod	numcde, codprod
LIVRER	3	numliv + codprod	numliv, codprod

B- Les opérateurs algébriques

1- Opérateurs algébriques spécifiques

Ils travaillent sur les attributs ou sur les valeurs des attributs des relations. On distingue deux types d'opérateurs spécifiques :

- les opérateurs portant sur une relation (*SELECTION, PROJECTION*)
- les opérateurs portant sur deux relations (*JOINTURE, PROJECTION*)

2- Opérateurs ensemblistes classiques

Ils travaillent sur des relations considérées comme des ensembles.

Un opérateur ensembliste peut porter sur des relations (*UNION, INTERSECTION, DIFFERENCE*) ou non (*PRODUIT CARTESIEN*)

3- Requête

Pour manipuler les données présentes dans une base, les utilisateurs doivent formuler des requêtes ou demandes pour avoir les résultats escomptés. L'ensemble des résultats est présenté dans un tableau appelé **table réponse**. Chaque colonne de cette table a pour nom un attribut de la relation projetée ou une fonction de calcul. Une ligne de données de cette table correspond à un résultat de la requête.

C- Mise en œuvre des concepts

Exercice introductif

Considérons les tables de données suivantes :

Table « DEVIS »

coddev	datdev	codcli
D1	12/12/2010	C01
D2	12/12/2010	C02
D3	12/12/2010	C03

Table « TYPE »

codtype	nomtype
E	Entreprise
O	ONG
P	Particulier

Table « FAMILLE »

codfam	nomfam
La	Laitier
Fr	Fromagerie
Ch	Charcuterie
Al	Alcool
Pd	Petit déjeuner
Go	Gourmandise
Ep	Epices

Table « CONTENIR »

codprod	coddev	qtecmde
P21	D1	2
P30	D1	5
P24	D1	3
P22	D2	2
P34	D2	1
P23	D2	10
P29	D2	1
P21	D3	1
P26	D3	1
P30	D3	2

Table « CLIENT »

codcli	nomcli	adrcli	villecli	codtype
C01	Maquis « La belle gueule »	03 BP 2020	Porto-Novo	E
C02	Hôtel « L'Oriental »	08 BP 1200	Calavi	E
C03	Hôtel « L'Amazone »	Rue Saint Georges	Parakou	E
C04	100% Alphonbétisation	Rue de l'apprenant	Cotonou	O
C05	M. Adogonon Georges	04 BP 4004	Malanville	P
C06	Sauver la Démocratie	Boulevard Obama	Lokossa	O

Table « PRODUIT »

codprod	nomprod	puprod	qtestock	codfam
P21	Lait entier	800	40	La
P22	Vin de table rosé	6 700	14	Al
P23	Gruyère	5 400	23	Fr
P24	Biscuit	975	42	Go
P25	Pâte à tartiner	4 850	50	Pd
P26	Chocolat	2 175	35	Go
P27	Sucre roux	1 250	50	Pd
P28	Jambon	1 000	19	Ch
P29	Bière blonde	1 600	30	Al
P30	Jus de fruit	2 500	32	Pd
P31	Yaourt	275	126	La
P32	Fromage	800	54	Fr
P33	Confiture	1 850	34	Pd
P34	Viande de bœuf	2 500	21	Ch
P35	Huile d'arachide	2 175	87	Ep
P36	Raisin	4 500	32	Go
P37	Moutarde	1 200	76	Ep
P38	Nescafé	800	25	Pd
P39	Tomate en boîte	250	33	Ep
P40	Beurre pasteurisé	875	100	Pd

C-1. La projection

Elle consiste à choisir les noms des colonnes représentant chacun un attribut exprimant ce que l'on désire obtenir.

Remarque :

La projection consiste en un découpage vertical d'une relation à projeter.

La syntaxe est la suivante :

Relation_RESULTAT = PROJECTION RELATION_Projetée (Attribut projeté(s)).

Exemple 1 :

Ecrire en langage algébrique puis illustrer les requêtes suivantes.

- a- Afficher les noms et adresse des clients.
- b- Afficher la liste des devis.

RESOLUTION

- a- Affichage des noms et adresse des clients

- **Ecriture algébrique**

Résultat = PROJECTION CLIENT (nomcli, adrcli)

- **Illustration**

nomcli	adrcli
Maquis « La belle gueule »	03 BP 2020
Hôtel « L'Oriental »	08 BP 1200
Hôtel « L'Amazone »	Rue Saint Georges
100% Alphabétisation	Rue de l'apprenant
M. Adogonon Georges	04 BP 4004
Sauver la Démocratie	Boulevard Obama

- b- Affichage de la liste des devis.

- **Ecriture algébrique**

Résultat = PROJECTION DEVIS (CodDev, DatDev, CodCli)

- **Illustration**

coddev	datdev	codcli
D1	12/12/2010	C01
D2	12/12/2010	C02
D3	12/12/2010	C03

C-2. La sélection

Elle consiste à sélectionner les lignes d'une ou de plusieurs tables qui vérifient simultanément toutes les conditions à préciser. Sa syntaxe est la suivante :

Relation_Résultat = SELECTION RELATION (Condition(s))

Avec condition = Attribut/comparateur/Valeur.

On note généralement deux types de comparateurs : **les comparateurs relationnels (=, >, >=, <, <=, <>)** et **les comparateurs non relationnels (LIKE, IN, BETWEEN, etc.)**

Remarque : la sélection consiste en un découpage horizontal d'une relation.

Exemple 1 (suite)

c- Afficher les clients de code type O

Résolution

- **Ecriture algébrique**

Résultat = SELECTION CLIENT (codtype = "O")

- **Illustration**

codcli	nomcli	adrcli	villecli	codtype
C04	100% Alfabétisation	Rue de l'apprenant	Cotonou	O
C06	Sauver la Démocratie	Boulevard Obama	Lokossa	O

C-3. La jointure

C'est une opération fondamentale qui permet de réunir deux par deux des tables.

En effet, cette opération permet de relier avec une relation d'égalité (on parle d'équi-jointure) des tables qui ont au moins un attribut en commun.

La syntaxe est la suivante :

Relation_Résultat = JOINTURE RELATION 1, RELATION 2 (condition de jointure) avec condition : RELATION 1.AttributClé=RELATION 2.AttributClé

Exemple 1 (suite)

d- Afficher la liste des devis avec les informations des clients.

Résolution

- **Ecriture algébrique**

Résultat = JOINTURE DEVIS, CLIENT (DEVIS.codcli = CLIENT.codcli)

- **Illustration**

coddev	datdev	codcli	codcli	nomcli	adrcli	villecli	codtype
D1	12/12/2010	C01	C01	Maquis « La belle gueule »	03 BP 2020	Porto- Novo	E
D2	12/12/2010	C02	C02	Hôtel « L'Oriental »	08 BP 1200	Calavi	E
D3	12/12/2010	C03	C03	Hôtel « L'Amazone »	Rue Saint Georges	Parakou	E

Remarque :

On peut également effectuer la combinaison des opérateurs précédemment étudiés.

Exemple 1 (suite)

e- Afficher les noms et prix des produits coûtant au moins 4000

Résolution

- **Ecriture algébrique**

R1 = SELECTION PRODUIT (puprod >= 4000)

Résultat = PROJECTION R1 (nomprod, puprod)

- **Illustration**

nomprod	puprod
Vin de table rosé	6700
Gruyère	5400
Pâte à tartiner	4800
Raisin	4500

Exemple 1 (suite)

f- Afficher les noms et les villes des clients habitant à Cotonou.

Résolution

- Ecriture algébrique

R1 = SELECTION CLIENT (villecli = "Cotonou")

Res = PROJECTION R1 (nomcli, villecli)

- Illustration

nomcli	villecli
100% Alphasabétisation	Cotonou

Exemple 1 (suite)

g- La liste des produits (nom et quantité en stock) dont la quantité en stock est d'au moins 50.

Résolution

- Ecriture algébrique

R1 = SELECTION PRODUIT (qtestock >= 50)

Res = PROJECTION R1 (nomprod, qtestock)

- Illustration

nomprod	qtestock
Pâte à tartiner	50
Sucre roux	50
Yaourt	126
Fromage	54
Huile d'arachide	87
Moutarde	76
Beurre pasteurisé	100

Exemple 1 (suite)

h- Afficher les codes et quantités des produits figurant sur le devis de code D3.

Résolution

- Ecriture algébrique

R1 = SELECTION CONTENIR (coddev = "D3")

Res = PROJECTION R1 (coddev, qtecmde)

- Illustration

coddev	qtecmde
P21	1
P26	1
P30	2

CHAPITRE 3 : Etude du langage SQL

I- Généralités

Une requête (query, en anglais) est une question posée à une base de données concernant les informations qu'elle contient, par exemple une recherche d'informations dans la base ou une action exécutée sur ces informations (ajout, suppression, modification).

Toute requête se comprend sur la base d'un raisonnement en trois temps : exprimer un besoin d'information ; programmer grâce au langage SQL le moyen de satisfaire ce besoin ; obtenir la réponse formelle.

Le langage structuré de requêtes (SQL : Structured Query Language, en Anglais) est un langage informatique (de type requête) standard et normalisé, destiné à manipuler une base de données relationnelle à l'aide des commandes.

Le but ce chapitre est d'étudier les différentes commandes du langage SQL et de les mettre en œuvre, par l'écriture des requêtes et leur exécution, en utilisant Microsoft Access, un logiciel de type Système de Gestion de Base de Données (SGBD).

II- Etude de quelques commandes

A- Création d'une table

La commande **CREATE TABLE** permet de créer une nouvelle table avec ses éléments caractéristiques.

Exemple :

Créer la structure de la table FAMILLE comportant deux colonnes : la première nommée codfam de type alphanumérique et de longueur 2 ; la deuxième nommée nomfam de type alphanumérique et de longueur 15.

On aura :

CREATE TABLE FAMILLE (codfam CHAR (02) NOT NULL PRIMARY KEY, nomfam CHAR (15) NOT NULL) ;

B- Modification de table

La commande **ALTER TABLE** permet d'ajouter une nouvelle modification à la structure d'une table.

Supposons qu'on veut ajouter une nouvelle colonne nommée col3 de type entier et de valeur nulle à la table FAMILLE.

On aura : ALTER TABLE FAMILLE ADD col3 INTEGER NULL ;

C- Suppression de table

La commande **DROP TABLE** permet de supprimer définitivement une table avec son contenu.

Pour supprimer la table FAMILLE, on aura DROP TABLE FAMILLE ;

D- Insertion des données dans une table

La commande **INSERT INTO** permet d'ajouter un ou plusieurs enregistrements à une table. Il existe deux types d'insertion :

- l'insertion mono ligne permettant d'ajouter un enregistrement à une table ;
- l'insertion multi lignes permettant d'ajouter plusieurs enregistrements dans une table par une requête.

La syntaxe d'une insertion mono ligne est la suivante

INSERT INTO NomTABLE (Attribut1, Attribut2,...)

VALUES (valeur1, Valeur2,...) ;

Exemple : Ajouter à la table FAMILLE un nouvel enregistrement tel que codfam est ag et nomfam est Agrume.

INSERT INTO FAMILLE (codfam, nomfam)

```
VALUES ("ag", "Agrume") ;
```

La syntaxe d'une insertion multi lignes est la suivante :

```
INSERT INTO NomTABLE (Attribut1, Attribut2,...)
```

```
SELECT (..... );
```

E- Mise à jour des données d'une table

La commande **UPDATE** permet de modifier une ou plusieurs lignes d'une table.

La syntaxe est la suivante :

```
UPDATE NomTABLE
```

```
SET Attribut1=Valeur1, Attribut2=Valeur2, ..., Attribut=ValeurN
```

```
WHERE condition ;
```

Exemple :

- Augmenter le prix unitaire des produits de 2%.

```
UPDATE PRODUIT
```

```
SET puprod=puprod * 1,02 ;
```

- Remplacer la ville du client dont le code est C02 par DASSA

On aura : UPDATE CLIENT

```
SET villecli="Dassa"
```

```
WHERE CodCli="C02".
```

F- Suppression des données d'une table

La commande **DELETE** permet de supprimer un ou plusieurs enregistrements d'une table. La syntaxe est la suivante :

```
DELETE FROM NomTABLE
```

```
WHERE condition ;
```

Exemple : Supprimer le client dont le code est C02.

On aura :

```
DELETE FROM CLIENT
```

```
WHERE codcli="C02" ;
```

G- Interrogation de données : La commande SELECT

1. Syntaxe générale

La syntaxe de rédaction d'une requête SQL d'interrogation de données est :

SELECT *<Liste des attributs projetés ou fonctions >*

FROM *<Liste des tables des attributs concernés par la requête>*

WHERE *<Liste des critères de jointure et de sélection>*

GROUP BY *<Attribut(s) de groupage>*

HAVING *<Critère de sélection portant sur un groupe d'enregistrements>*

ORDER BY *<Critère de tri ou d'ordre>* ;

2. Le tableau suivant regroupe les différentes fonctions statistiques et fonctions de temps utilisées en langage SQL

Fonctions statistiques	Rôle
COUNT ([DISTINCT] attribut)	Renvoie le nombre de valeurs de la colonne (ou attribut) spécifié(e). Si DISTINCT est précisé, les doublons sont éliminés.
COUNT (*)	Compte toutes les lignes d'une table.
AVG (attribut)	Renvoie la moyenne des valeurs de la colonne (ou attribut) spécifié(e).
MIN (attribut)	Renvoie la plus petite valeur de la colonne (ou attribut) spécifié(e).
MAX (attribut)	Renvoie la plus grande valeur de la colonne (ou attribut) spécifié(e).
SUM (attribut)	Renvoie la somme des valeurs de la colonne (ou attribut) spécifié(e).

Fonctions de temps	Rôle
YEAR (« Expression de date »)	Extrait l'année de l'année spécifiée.
MONTH (« Expression de date »)	Extrait le mois de l'année spécifiée.
DAY (« Expression de date »)	Extrait le jour de l'année spécifiée.

3. Etude de quelques prédicats utilisés en langage SQL

a- Le prédicat **NULL**

Un attribut (une colonne, un champ...) peut avoir la valeur "NULL" soit en raison d'information incomplète (la valeur n'était pas connue au moment de la saisie des données), soit parce que la donnée n'est pas pertinente. La valeur "NULL" est différente de la valeur par défaut d'un l'attribut : zéro pour un attribut de type numérique et espace pour un attribut de type caractère.

La syntaxe est : **IS NULL** et sa négation **IS NOT NULL**.

b- Le prédicat **IN**

Il comporte une liste de valeurs et vérifie si une valeur particulière apparaît sur cette liste.

La syntaxe est : **IN** (val1, val2, ...) et sa négation **NOT IN** (val1, val2, ...).

Lorsque la liste des valeurs est connue et fixe, le prédicat **IN** peut être remplacé par une suite d'opérateurs logiques **OR**.

NB: lorsque la liste des valeurs n'est pas connue à priori, comme dans certaines sous-interrogations (ou sous-requêtes), ce prédicat est nécessaire.

c- Le prédicat **LIKE**

Ce prédicat permet de faire des recherches à l'intérieur d'une chaîne de caractères lorsque l'on dispose d'informations incomplètes.

Il utilise deux (2) caractères génériques :

% : utilisé pour représenter une chaîne de caractère de longueur quelconque.

? : utilisé pour représenter un caractère unique.

La syntaxe est : **LIKE** '*Chaîne de recherche*' et sa négation **NOT LIKE** '*Chaîne de recherche*'

d- Le prédicat **BETWEEN**

Il permet de comparer la valeur d'un champ par rapport à une borne inférieure et une borne supérieure (bornes incluses).

NB : en fonction des conditions que l'on veut exprimer au niveau des valeurs renvoyées par une requête, on peut utiliser les prédicats IN, ANY, ALL ou EXISTS.

e- Le prédicat **ANY**

Il permet de vérifier si au moins une valeur de la liste spécifiée (ou renvoyée par la sous-requête) satisfait la condition.

f- Le prédicat **ALL**

Permet de vérifier si la condition est réalisée pour toutes les valeurs de la liste spécifiée (ou renvoyée par la sous-requête).

g- Le prédicat **EXISTS**

Si la sous-interrogation renvoie un résultat, la valeur retournée est "Vrai" sinon la valeur "Faux" est retournée.

4. Notion d'alias

Un alias permet de renommer une colonne ou une table. Cette opération est utile lorsque l'on veut qu'une colonne ait un nom plus "parlant" ou qu'une table puisse être manipulée plus facilement notamment quand il existe différentes conditions de jointure.

- Alias de colonne (ou d'attribut)

Syntaxe : Col1 **AS** "Nom de la colonne"

Dans le résultat de la requête correspondante, Col1 sera remplacée par "Nom de la colonne". Les griffes ne sont obligatoires que si le nom d'alias comporte des espaces

- **Alias de table** Syntaxe : Table1 Alias1

Dans une requête, Alias1 sera identique à Table1

5. Notion de sous-requête

Les sous-interrogations, ou SELECT imbriqués, permettent de réaliser des interrogations complexes qui nécessiteraient normalement plusieurs interrogations avec stockage des résultats intermédiaires. Elles se caractérisent par une interrogation (clause SELECT) incluse dans la clause WHERE, la clause HAVING ou la clause SELECT d'une autre interrogation.

Remarque : dans le cas de sous-requêtes, seuls les attributs du premier SELECT sont projetés. La sous-requête doit être placée entre parenthèses. Elle ne doit pas comporter de clause ORDER BY, mais peut inclure les clauses GROUP BY et HAVING.

Le résultat d'une sous-requête est utilisé par la requête du niveau supérieur.

Une sous- requête est exécutée avant la requête de niveau supérieur.

III- Mise en œuvre d'une base de données avec Microsoft Access

Travaux Pratiques N°1 : création d'une base de données avec Microsoft Access

On vous fournit les informations des annexes ci-après :

Annexe 1 : Tables de données

Table : CLIENT

numcli	nomcli	prenccli	adrcli	villecli
4010	AZONDE	Jean	01 BP 4023	Cotonou
4011	VIVIME	Léopold	Tél. 95 06 11 11	Parakou
4012	LAWANI	Johan	06 BP 004	Porto-Novo
4013	BALOGOUN	Didier	04 BP 2024	Parakou
4014	ABALO	Jérôme	Tél. 97 06 38 18	Bohicon
4015	COCOU	Yves	Tél. 93 34 41 11	Porto-Novo
4016	DANSOU	Côme	Tél. 90 90 04 40	Cotonou
4017	AFANVI	David	06 BP 3434	Porto-Novo
4018	FANOU	Rolland	04 BP 444	Cotonou
4019	EFIOU	Afi	Tél. 95 90 01 01	Cotonou
4020	GANDONOU	Rock	Tél. 90 06 38 41	Bohicon
4021	GUEDE	Zinsou	03 BP 2711	Cotonou
4022	ZANNOU	Magloire	05 BP 3618	Bohicon
4023	MAMADOU	Issifou	02 BP 4400	Natitingou
4024	PADONOU	Irène	Tél. 21 01 44 20	Lokossa
4025	OUINSOU	Clotilde	01 BP 2100	Cotonou
4026	ALLE	Constance	13, rue des Amazones	Cotonou
4027	BEHANZIN	Didier	13, rue des Amazones	Cotonou
4028	EGNON	Clémence	Boulevard des Oliviers	Parakou
4029	JOHNSON	Matthieu	Boulevard Saint Michel	Cotonou

Table : REPRESENTANT

numrep	nomrep	salbase	tauxcom
1010	Zannou Dieudonné	50 000	0,15
1011	Kinto Robert	75 000	0,10
1012	Bamikpo Grégoire	60 000	0,20
1013	Ali Traoré	100 000	0,25
1014	Poupo Alagnon	80 000	0,15
1015	Bakary Séibou	60 000	0,20

Table : COMMANDE

numcmde	datecmde	adrliv	numcli	numrep
550	13/06/2012	Grand Marché	4011	1010
551	13/06/2012	St Michel	4020	1010
552	13/06/2012	Tokpa Hoho	4013	1011
553	15/06/2012	Ganhito	4010	1015
554	15/06/2012	Godomey	4025	1012
555	16/06/2012	Wologuèdè	4021	1014
556	17/06/2012	Wologuèdè	4029	1014
557	17/06/2012	Grand Marché	4010	1013
558	18/06/2012	St Michel	4015	1010
559	18/06/2012	St Michel	4010	1012
560	19/06/2012	Grand Marché	4018	1010
561	13/06/2012	Grand Marché	4011	1010

Table : PRODUIT

codprod	libprod	prixunit
P01	Pâte dentifrice	700
P02	Lait en poudre	2 800
P03	Fromage	600
P04	Lait en boîte	1 500
P05	Suze	5 800
P06	Pastis	3 500
P07	Rhum St James	6 000
P08	William Lawson's	6 000
P09	Coca Cola	1 025
P10	Sucre roux	1 350

Table : DEPARTEMENT

coddep	nomdep	superficie
D01	Alibori	7 000
D02	Borgou	15 000
D03	Couffo	8 500
D04	Mono	9 000
D05	Collines	6 500
D06	Zou	12 000
D07	Atlantique	10 000
D08	Ouémé	8 500
D09	Plateau	6 000
D10	Donga	14 000
D11	Atacora	13 000
D12	Littoral	7 000

Table : PROSPECTER

numrep	coddep
1010	D08
1010	D09
1010	D11
1011	D03
1011	D11
1012	D04
1012	D10
1013	D01
1013	D02
1014	D11
1015	D06
1015	D07

Table : CONCERNER

numcmde	codprod	qtecmde
550	P03	50
550	P06	25
550	P09	28
551	P03	31
551	P08	13
552	P01	13
552	P03	18
552	P09	24
553	P02	25
554	P03	21
554	P08	9
555	P01	25

555	P04	18
555	P06	18
555	P09	31
556	P02	10
556	P07	12
556	P01	30
557	P01	23
557	P05	8
557	P08	29
558	P06	8
558	P10	11
559	P02	12
559	P05	16
559	P07	8
560	P03	9
560	P04	12
560	P09	24
560	P10	11

Annexe 2 : Dictionnaire des données

Propriété	Description	Type	Longueur
numcli	Numéro du client	N	04
nomcli	Nom du client	A	15
prencli	Prénom du client	A	20
adrcli	Adresse du client	AN	25
villecli	Ville du client	A	15
numrep	Numéro du représentant	N	04
nomrep	Nom du représentant	A	20
salbase	Salaire de base du représentant	N	06
tauxcom	Taux de commission du représentant	N	04
numcmde	Numéro de la commande	N	03
datecmde	Date de la commande	D	10
adrliv	Adresse de livraison de la commande	AN	15
codprod	Code du produit	AN	03
libprod	Libellé du produit	A	20
prixunit	Prix unitaire du produit	N	04
coddep	Code du département	AN	03
nomdep	Nom du département	A	10
superficie	Superficie du département	N	06
qtecmde	Quantité commandée pour chaque produit	N	03

Travail à Faire :

- 1- Utiliser le logiciel Microsoft Access pour créer une base de données nommée "TP ACCESS" en se servant des tables de données ci-dessus.
- 2- Créer la structure des différentes tables.
- 3- Afficher toutes les tables précédemment créées.
- 4- Créer des liens possibles entre les tables affichées.
- 5- Insérer les données dans les tables.

Travaux Pratiques N°2 : création et exécution des requêtes sous SQL

Créer et Exécuter chacune des requêtes suivantes

Requête 1

- Affichage des codes et noms de tous les produits.
- Ecriture de la requête en SQL

SELECT codprod, libprod

FROM PRODUIT;

- Réponse formelle

Requête1	
codprod	libprod
P01	Pâte dentifrice
P02	Lait en poudre
P03	Fromage
P04	Lait en boîte
P05	Suze
P06	Pastis
P07	Rhum St James
P08	William Lawson's
P09	Coca Cola
P10	Sucre roux

Requête 2

- Affichage des noms et adresses des clients.
- Ecriture de la requête en SQL

SELECT nomcli, adrcli

FROM CLIENT;

- Réponse formelle

Requête2	
nomcli	adrcli
AZONDE	01 BP 4023
VIVIME	Tél. 95 06 11 11
LAWANI	06 BP 004
BALOGOUN	04 BP 2024
ABALO	Tél. 97 06 38 18
COCOU	Tél. 93 34 41 11
DANSOU	Tél. 90 90 04 40
AFANVI	06 BP 3434
FANOU	04 BP 444
EFIYOU	Tél. 95 90 01 01
GANDONOU	Tél. 90 06 38 41
GUEDE	03 BP 2711
ZANNOU	05 BP 3618
MAMADOU	02 BP 4400
PADONOU	Tél. 21 01 44 20
OUIINSOU	01 BP 2100
ALLE	13, rue des Amazones
BEHANZIN	13, rue des Amazones
EGNON	Boulevard des Oliviers
JOHNSON	Boulevard Saint Michel

Requête 3

- Liste de tous les produits.
- Ecriture de la requête en SQL
SELECT codprod, libprod, prixunit
FROM PRODUIT;
- Réponse formelle

OU
SELECT *
FROM PRODUIT;

Requête3		
codprod	libprod	prixunit
P01	Pâte dentifrice	700
P02	Lait en poudre	2800
P03	Fromage	600
P04	Lait en boîte	1500
P05	Suze	5800
P06	Pastis	3500
P07	Rhum St James	6000
P08	William Lawson's	6000
P09	Coca Cola	1025
P10	Sucre roux	1350

*NB : le symbole * placé au niveau de la clause SELECT remplace tous les attributs de la table se trouvant au niveau de la clause FROM*

Requête 4

- Afficher sans répétition les villes d'habitation des clients.
- Ecriture de la requête en SQL
SELECT DISTINCT villecli
FROM CLIENT;

NB : le mot clé DISTINCT placé avant un attribut se trouvant au niveau de la clause SELECT, empêche la répétition des valeurs de cet attribut lors de l'affichage

- Réponse formelle

Requête4
villecli
Bohicon
Cotonou
Lokossa
Natitingou
Parakou
Porto-Novo

Requête 5

- Liste des clients habitant Cotonou.
- Ecriture de la requête en SQL

SELECT *

FROM CLIENT

WHERE villecli = "Cotonou";

NB : on met la valeur d'un attribut de sélection entre griffes lorsque celle-ci comporte au moins un caractère non numérique ; c'est le cas ici de la valeur Cotonou prise par l'attribut de sélection villecli.

- Réponse formelle

Requête5				
numcli	nomcli	prenccli	adrcli	villecli
4010	AZONDE	Jean	01 BP 4023	Cotonou
4016	DANSOU	Côme	Tél. 90 90 04 40	Cotonou
4018	FANOU	Rolland	04 BP 444	Cotonou
4019	EFIOU	Afi	Tél. 95 90 01 01	Cotonou
4021	GUEDE	Zinsou	03 BP 2711	Cotonou
4025	OUINSOU	Clotilde	01 BP 2100	Cotonou
4026	ALLE	Constance	13, rue des Amazones	Cotonou
4027	BEHANZIN	Didier	13, rue des Amazones	Cotonou
4029	JOHNSON	Matthieu	Boulevard Saint Michel	Cotonou

Requête 6

- Les numéros et dates des commandes livrées au Grand Marché.
- Ecriture de la requête en SQL
SELECT numcmde, datecmde
FROM COMMANDE
WHERE adrliv = "Grand Marché";
- Réponse formelle

Requête6	
numcmde	datecmde
550	13/06/2012
557	17/06/2012
560	19/06/2012
561	13/06/2012

Requête 7

- Nom et prénom des clients habitant 13, rue des Amazones.
- Ecriture de la requête en SQL
SELECT nomcli, prenci
FROM CLIENT
WHERE adrcli = "13, rue des Amazones";
- Réponse formelle

Requête7	
nomcli	prenci
ALLE	Constance
BEHANZIN	Didier

Requête 8

- Liste des produits dont le nom commence par P.
- Ecriture de la requête en SQL

SELECT *
FROM PRODUIT
WHERE libprod LIKE "P*";

- Réponse formelle

Requête8		
codprod	libprod	prixunit
P01	Pâte dentifrice	700
P06	Pastis	3500

*NB : Le prédicat **LIKE** est utilisé, comme un comparateur pour formuler un critère de sélection, lorsque la valeur de l'attribut de sélection en question comporte des informations incomplètes. La négation de ce prédicat est **NOT LIKE**.*

*A cet effet, on utilise le symbole * pour remplacer un groupe de caractères inconnus dans cette valeur de l'attribut de sélection.*

Requête 9

- Nom et superficie des départements dont le 2^{ème} caractère du nom est O.
- Ecriture de la requête en SQL

SELECT nomdep, superficie
FROM DEPARTEMENT
WHERE nomdep LIKE "?o*";

- Réponse formelle

Requête9	
nomdep	superficie
Borgou	15000
Couffo	8500
Mono	9000
Collines	6500
Zou	12000
Donga	14000

Requête 10

- Noms et adresses des clients dont l'adresse comporte le sigle "BP"
- Ecriture de la requête en SQL

SELECT nomcli, adrcli
FROM CLIENT
WHERE adrcli LIKE "*BP*";

- Réponse formelle

Requête10	
nomcli	adrcli
AZONDE	01 BP 4023
LAWANI	06 BP 004
BALOGOUN	04 BP 2024
AFANVI	06 BP 3434
FANOU	04 BP 444
GUEDE	03 BP 2711
ZANNOU	05 BP 3618
MAMADOU	02 BP 4400
OUINSOU	01 BP 2100

Requête 11

- Liste des clients dont l'adresse ne comporte pas le sigle "BP"
- Ecriture de la requête en SQL

SELECT *

FROM CLIENT

WHERE adrcli NOT LIKE "*BP*";

- Réponse formelle

Requête11				
numcli	nomcli	prencli	adrcli	villecli
4011	VIVIME	Léopold	Tél. 95 06 11 11	Parakou
4014	ABALO	Jérôme	Tél. 97 06 38 18	Bohicon
4015	COCOU	Yves	Tél. 93 34 41 11	Porto-Novo
4016	DANSOU	Côme	Tél. 90 90 04 40	Cotonou
4019	EFIU	Afi	Tél. 95 90 01 01	Cotonou
4020	GANDONOU	Rock	Tél. 90 06 38 41	Bohicon
4024	PADONOU	Irène	Tél. 21 01 44 20	Lokossa
4026	ALLE	Constance	13, rue des Amazones	Cotonou

Requête11				
numcli	nomcli	prencli	adrcli	villecli
4027	BEHANZIN	Didier	13, rue des Amazones	Cotonou
4028	EGNON	Clémence	Boulevard des Oliviers	Parakou
4029	JOHNSON	Matthieu	Boulevard Saint Michel	Cotonou

Requête 12

- Les dates des commandes passées par le client de numéro 4010
- Ecriture de la requête en SQL

```
SELECT datecmde  
FROM COMMANDE  
WHERE numcli = 4010;
```

- Réponse formelle

Requête12
datecmde
15/06/2012
17/06/2012
18/06/2012

Requête 13

- Affichage des noms et prix des produits coûtant moins de 3000
- Ecriture de la requête en SQL

```
SELECT libprod, prixunit  
FROM PRODUIT  
WHERE prixunit < 3000;
```

- Réponse formelle

Requête13	
libprod	prixunit
Pâte dentifrice	700
Lait en poudre	2800
Fromage	600
Lait en boîte	1500
Coca Cola	1025

Requête13	
libprod	prixunit
Sucre roux	1350

Requête 14

- Les représentants dont le salaire de base est 60000
- Ecriture de la requête en SQL

SELECT *

FROM REPRESENTANT

WHERE salbase = 60000;

- Réponse formelle

Requête14			
numrep	Nomrep	salbase	tauxcom
1012	Bamikpo Grégoire	60000	0,2
1015	Bakary Séïbou	60000	0,2

Requête 15

- Les produits dont le prix unitaire est différent de 5000
- Ecriture de la requête en SQL

SELECT *

FROM PRODUIT

WHERE prixunit <>5000;

- Réponse formelle

Requête15		
codprod	Libprod	prixunit
P01	Pâte dentifrice	700
P02	Lait en poudre	2800
P03	Fromage	600
P04	Lait en boîte	1500
P05	Suze	5800
P06	Pastis	3500
P07	Rhum St James	6000

Requête15		
codprod	Libprod	prixunit
P08	William Lawson's	6000
P09	Coca Cola	1025
P10	Sucre roux	1350

Requête 16

- Liste des produits dont le prix unitaire est compris entre 2000 et 5000
 - Ecriture de la requête en SQL
- ```

SELECT *
FROM PRODUIT
WHERE prixunit BETWEEN 2000 And 5000;

```

*NB : Le prédicat **BETWEEN** est utilisé, comme un comparateur pour formuler un critère de sélection, lorsque l'attribut de sélection est susceptible de prendre des valeurs consécutives définies dans un intervalle dont le début et la fin sont précisés. La négation de ce prédicat est **NOT BETWEEN**.*

- Réponse formelle

| Requête16 |                |          |
|-----------|----------------|----------|
| codprod   | libprod        | prixunit |
| P02       | Lait en poudre | 2800     |
| P06       | Pastis         | 3500     |

### Requête 17

- La liste des représentants dont le salaire de base est compris entre 75000 et 100000 avec indication du numéro, du nom et du salaire de base.
  - Ecriture de la requête en SQL
- ```

SELECT numrep, nomrep, salbase
FROM REPRESENTANT
WHERE salbase BETWEEN 75000 And 100000;

```
- Réponse formelle

Requête17		
numrep	nomrep	salbase
1011	Kinto Robert	75 000
1013	Ali Traoré	100000

Requête17		
numrep	nomrep	salbase
1014	Poupo Alagnon	80000

Requête 18

- Numéros des commandes passées le 15 juin 2012

- Ecriture de la requête en SQL

SELECT numcmde

FROM COMMANDE

WHERE datecmde = #15/06/2012#;

NB : toute valeur d'un attribut de type Date se présente sous le format JJ/MM/AAAA. Elle est encadrée de part et d'autre par le symbole #.

- Réponse formelle

Requête18
numcmde
553
554

Requête 19

- Les adresses de livraison des commandes passées dans la 2^{ème} quinzaine de juin 2012.

- Ecriture de la requête en SQL

SELECT DISTINCT adrliv

FROM COMMANDE

WHERE datecmde BETWEEN #16/06/2012# And #30/06/2012#;

NB : il est nécessaire ici de précéder l'attribut adrliv du mot clé DISTINCT pour éviter la répétition d'une même valeur de cet attribut car plusieurs commandes de même adresse de livraison peuvent être livrées dans la période indiquée.

- Réponse formelle

Requête19
adrliv
Grand Marché
St Michel
Wologuèdè

Requête 20

- Les numéros des commandes sur lesquelles la quantité commandée de certains produits est comprise entre 10 et 20.

- Ecriture de la requête en SQL

SELECT DISTINCT numcmde

FROM CONCERNER

WHERE qtecnde BETWEEN 10 And 20;

- Réponse formelle

Requête20	
numcmde	
551	
552	
555	
556	
558	
559	
560	

NB : il est nécessaire ici de précéder l'attribut numcmde du mot clé DISTINCT pour éviter la répétition d'une même valeur de cet attribut car plusieurs produits inscrits sur un même bon de commande peuvent avoir des quantités comprises entre 10 et 20.

Requête 21

- Le nom et le prénom du client qui a passé la commande de numéro 555

- Ecriture de la requête en SQL

SELECT nomcli, prencli

FROM CLIENT, COMMANDE

WHERE CLIENT.numcli = COMMANDE.numcli

AND numcmde = 555;

- Réponse formelle

Requête21	
nomcli	prencli
GUEDE	Zinsou

Requête 22

- Les dates de commandes du produit "Fromage" avec indication du libellé du produit

- Ecriture de la requête en SQL

SELECT DISTINCT datecmde, libprod

FROM COMMANDE, PRODUIT, CONCERNER

WHERE COMMANDE.numcmde = CONCERNER.numcmde

AND CONCERNER.codprod = PRODUIT.codprod

AND libprod = "Fromage" ;

- Réponse formelle

Requête22	
datecmde	libprod
13/06/2012	Fromage
15/06/2012	Fromage
19/06/2012	Fromage

NB : il peut arriver que le produit dont le nom est Fromage soit commandé sur plusieurs commandes le même jour.

A cet effet, il est nécessaire d'utiliser dans la clause SELECT, le mot clé DISTINCT pour éviter, lors de l'affichage des résultats,

Requête 23

- Les noms et prénoms des clients qui ont commandé le produit "Pâte dentifrice"

- Ecriture de la requête en SQL

SELECT DISTINCT nomcli, prenci

FROM CLIENT, PRODUIT, COMMANDE, CONCERNER

WHERE CLIENT.numcli = COMMANDE.numcli

AND COMMANDE.numcmde = CONCERNER.numcmde

AND CONCERNER.codprod = PRODUIT.codprod

AND libprod = "Pâte dentifrice";

- Réponse formelle

Requête23	
nomcli	prencli
BALOGOUN	Didier
GUEDE	Zinsou
JOHNSON	Matthieu
AZONDE	Jean

NB : il peut arriver que le produit dont le nom est Pâte dentifrice soit commandé sur plusieurs commandes par un même client.

A cet effet, il est nécessaire d'utiliser dans la clause SELECT, le mot clé DISTINCT pour éviter, lors de l'affichage des résultats, la répétition d'une même valeur des attributs nomcli et

Requête 24

- La liste des produits figurant sur la commande n° 550
- Ecriture de la requête en SQL
SELECT PRODUIT.*

FROM PRODUIT, CONCERNER

WHERE PRODUIT.codprod = CONCERNER.codprod

AND numcmde = 550;

- Réponse formelle

Requête24		
codprod	libprod	prixunit
P03	Fromage	600
P06	Pastis	3500
P09	Coca Cola	1025

*NB : le symbole * se place seul devant SELECT lorsqu'il y a une seule table au niveau de FROM.*

Dans le cas où on a plusieurs tables au niveau de FROM, il est nécessaire de précéder ce symbole de la table dont il remplace les attributs.

Requête 25

- Les numéros et noms des représentants intervenant dans le département ATACORA
- Ecriture de la requête en SQL

SELECT REPRESENTANT.numrep, nomrep

FROM REPRESENTANT, DEPARTEMENT, PROSPECTER

WHERE REPRESENTANT.numrep = PROSPECTER.numrep

AND PROSPECTER.coddep = DEPARTEMENT.coddep

AND nomdep = "Atacora";

- Réponse formelle

Requête25	
numrep	nomrep
1010	Zannou Dieudonné
1011	Kinto Robert
1014	Poupo Alagnon

NB : dans une requête SQL, tout attribut ayant servi à formuler un critère de jointure doit être précédé, partout où on le trouve, de la table dans laquelle il est considéré comme une clé primaire.

Requête 26

- Total des salaires de base des représentants prospectant dans le département ATACORA.

- Ecriture de la requête en SQL

```
SELECT SUM(salbase) As Total_Salaire  
FROM REPRESENTANT, DEPARTEMENT, PROSPECTER  
WHERE REPRESENTANT.numrep = PROSPECTER.numrep  
AND PROSPECTER.coddep = DEPARTEMENT.coddep  
AND nomdep = "Atacora";
```

- Réponse formelle

Requête26
Total_Salaire
205 000

Requête 27

- Total de la commande 555.

- Ecriture de la requête en SQL

```
SELECT SUM(qtecnde*prixunit) As Total_Com  
FROM CONCERNER, PRODUIT  
WHERE CONCERNER.codprod = PRODUIT.codprod  
AND numcmde = 555;
```

- Réponse formelle

Requête27
Total_Com
139 275

Requête 28

- Noms et prénoms des clients par ville
- Ecriture de la requête en SQL
SELECT villecli, nomcli, prenci
FROM CLIENT
GROUP BY villecli, nomcli, prenci;
- Réponse formelle

Requête28		
villecli	nomcli	prenci
Bohicon	ABALO	Jérôme
Bohicon	GANDONOU	Rock
Bohicon	ZANNOU	Magloire
Cotonou	ALLE	Constance
Cotonou	AZONDE	Jean
Cotonou	BEHANZIN	Didier
Cotonou	DANSOU	Côme
Cotonou	EFIYOU	Afi
Cotonou	FANOU	Rolland
Cotonou	GUEDE	Zinsou
Cotonou	JOHNSON	Matthieu
Cotonou	OUINSOU	Clotilde
Lokossa	PADONOU	Irène
Natitingou	MAMADOU	Issifou
Parakou	BALOGOUN	Didier
Parakou	EGNON	Clémence
Parakou	VIVIME	Léopold
Porto-Novo	AFANVI	David
Porto-Novo	COCOU	Yves
Porto-Novo	LAWANI	Johan

*NB : pour cette requête, l'utilisation du mot « **par** » suivi de « **ville** », une information de la base de données représentée par l'attribut **villecli**, sans qu'il y ait un article (le, la, un, une,...) les séparant, entraîne un regroupement des données des attributs **nomcli** et **prenci** autour d'une même valeur de l'attribut **villecli** ; d'où l'utilisation de la clause **GROUP BY**.*

*Tout attribut de la clause **GROUP BY** est également attribut de la clause **SELECT** et vice-versa.*

Cet attribut doit être placé au début des autres.

Requête 29

- Les numéros des commandes dont la quantité totale des produits dépasse 50 au cours du 2^{ème} trimestre 2012
- Ecriture de la requête en SQL

```
SELECT  COMMANDE.numcmde, SUM(qtecmde) As Qte_Tot
FROM    COMMANDE, CONCERNER
WHERE   COMMANDE.numcmde=CONCERNER.numcmde
AND     datecmde BETWEEN #01/04/2012# And #30/06/2012#
GROUP BY COMMANDE.numcmde
HAVING  SUM(qtecmde)>50 ;
```

- Réponse formelle

Requête29	
numcmde	Qte_Tot
550	103
552	55
555	92
556	52
557	60
560	56

NB : tout critère de sélection formulé avec une fonction de calcul (COUNT, SUM,...) ne peut figurer au niveau de la clause WHERE mais plutôt au niveau de la clause HAVING qui nécessite préalablement l'usage de la clause GROUP BY.

Il est recommandé de compléter aux éléments de SELECT, la fonction de calcul utilisée pour formuler le critère de sélection se trouvant au niveau de la clause HAVING.

Requête 30

- La liste des représentants par salaire de base dont le salaire de base dépasse 50000 avec indication du salaire de base, du numéro et du nom des représentants.

- Ecriture de la requête en SQL

```
SELECT  salbase, numrep, nomrep
FROM    REPRESENTANT
GROUP BY salbase, numrep, nomrep
HAVING  salbase >50000;
```

- Réponse formelle

Requête30		
salbase	numrep	nomrep
60000	1012	Bamikpo Grégoire
60000	1015	Bakary Séïbou
75000	1011	Kinto Robert
80000	1014	Poupo Alagnon
100000	1013	Ali Traoré

*NB : pour cette requête, l'utilisation du mot « **par** » suivi de « **salaire** », une information de la base de données représentée par l'attribut **salbase**, entraine l'utilisation de la clause GROUP BY.*

Puisque cet attribut de groupage est utilisé pour formuler le critère de sélection, ce critère doit être placé au niveau de la clause HAVING.

Requête 31

- Nom des représentants du moins payé au plus payé
- Ecriture de la requête en SQL

SELECT nomrep, salbase

FROM REPRESENTANT

ORDER BY salbase ASC;

OU

SELECT nomrep, salbase

FROM REPRESENTANT

ORDER BY salbase;

- Réponse formelle

Requête31	
nomrep	salbase
Zannou Dieudonné	50000
Bakary Séïbou	60000
Bamikpo Grégoire	60000
Kinto Robert	75000
Poupo Alagnon	80000
Ali Traoré	100000

NB : il est recommandé de compléter tout attribut ou fonction de la clause ORDER BY aux éléments de la clause SELECT.

Cet élément complété se place après les attributs de SELECT.

Le mot clé ASC (Ascendant) indique l'ordre ascendant des données de l'attribut salbase lors de l'affichage. Si l'ordre d'affichage était décroissant, on aurait utilisé le mot clé DESC (Descendant)

Requête 32

- Liste des noms (dans l'ordre alphabétique inverse) des produits figurant sur la commande 555.
- Ecriture de la requête en SQL

```
SELECT libprod
FROM PRODUIT, CONCERNER
WHERE PRODUIT.codprod = CONCERNER.codprod
AND numcmde = 555
ORDER BY libprod DESC;
```

- Réponse formelle

Requête32
libprod
Pâte dentifrice
Pastis
Lait en boîte
Coca Cola