Création de la structure d'une base de données

Créer une base de données

- Pour créer une nouvelle base de données on utilise la commande suivante:

```
CREATE DATABASE nomdelabase ;
```

Exemple:

Create database gestionCommandes;

Supprimer une base de données

- Pour supprimer une base déjà existante on utilise la commande **drop** de la manière suivante:

```
DROP  Database nomdebase ;
```

Exemple:

Drop database gestionCommandes;

Utilisation d'une base de données

- Pour sélectionner la base de données convenable à votre travail on se sert de la commande USE comme suit:

```
USE nomdebase;
```

Exemple:

Use gestionCommandes;

Création des tables

```
CREATE TABLE nomTABLE(
Colonne1        type   [DEFAULT   valeur],
Colonne2        type   [DEFAULT  valeur],
.....
)
```

- Colonne 1 c'est le nom du champ et type est son type
- L'instruction **[DEFAULT valeur]** permet de préciser une valeur par défaut de ce champ. Cette instruction n'est pas obligatoire.

Exemple de création de table

```
Create table client(
        Codeclient      varchar(5),
        NomClient       varchar(20),
        Adresse         varchar(40),
        Ville           varchar(20) DEFAULT  'Oujda',
        CP              int)
```

Les contraintes d'intégrité

- Tout SGBD fait appel à des contraintes (conditions) pour **empêcher l'entrée de données incorrectes dans les tables.**
  - C'est-à-dire que les contraintes d'intégrité, contrôlent les règles de gestion au niveau de la table
  - Empêche la suppression d'un enregistrement s'il y'a une dépendance avec une autre table
  - Eviter de supprimer une table qui dépend d'une autre table.

NULL/NOT NULL

- Cette contrainte autorise ou non, la valeur NULL dans une colonne.
- ***Exemple:***

```
Create table client (
        Idc   int NOT NULL,
        Nom varchar(20),  …)
```

UNIQUE (1)

- Cette contrainte exige que chaque valeur dans une colonne ou dans un ensemble de colonnes soient unique. C'est-à-dire qu'il n'existe pas dans plusieurs lignes, pour la colonne concernée, la même valeur.
  - Une table peut comporter plusieurs colonnes qui acceptent la contrainte unique.
  - Un champ unique accepte la valeur **null** par default.

UNIQUE (2)

- Exemple 1:

```
Create table client(
Idc int NOT NULL,
Nom varchar(20)  unique,.....)
```

- Exemple 2:

```
Create table client(
Idc int NOT NULL,
Nom varchar(20),
Prenom varchar(20),
Unique(Nom,Prenom) )
```

Primary key

- Cette contrainte permet de créer une clé primaire pour une table. Une seule clé primaire peut être crée par table.

- Exemple 1: **la clé primaire porte sur un seul champ**

```
Create table client(
        Idc int primary key,
        Nom varchar(20),…)
```

Primary key

- Exemple 2: **la clé primaire porte sur deux champs ou plus**

```
Create table client(
            Idc int,
            Nom varchar(20),…
            Primary key(idc,Nom)
)
```

La clause AUTO_INCREMENT

- Elle permet **d'incrémenter automatiquement** la valeur de la clé primaire.
- Cette clause est utilisée pour les clés primaires qui sont de type **int**.

- Exemple:

```
Create table client(
Idc int  primary key AUTO_INCREMENT,
Nom varchar(20),…)
```

Foreign key

- Cette contrainte désigne une colonne qui est **clé étrangère.**
- Exemple:

```
Create table filiere(
            numf  int  primary key,
            Nomfiliere  varchar(20) );

Create table stagiaire (
            Numisnc  int  primary key,
            Nom  varchar(20),..
            Idf  int,
            Foreign key (idf)    references    filiere(numf)
)
```

Foreign key

- Autre méthode:

```
Create table stagiaire(
Numisnc  int  primary key,
Nom  varchar(20),..
Idf  int,
Constraint    fk_s    foreign key (idf)    references    filiere(numf)
)
```

Où fk_s est le nom de la contrainte.

check

- Définit une **condition** sur une colonne que chaque ligne doit vérifier.

- La condition peut inclure plusieurs ***expressions logiques*** combinées par **AND** et **OR**. S'il existe plusieurs contraintes Check pour la même colonne, elles sont validées dans l'ordre de leur création.

check

- Dans la condition de **check** on peut utiliser:
  - Les opérateurs de comparaison (=, >,>=,<,<=,!=)
  - Les opérateurs logiques (**AND** et **OR**)
- Exemple 1:

```
Create table personne(
idp  int primary key,
Nom varchar(20) not null,
Age  int check(age>=18 AND age<=30)
)
```

check

- Dans la condition de la contrainte check, on peut utiliser la clause **IN** et **NOT IN**: avec cette clause on spécifie **une liste de valeurs.**
- Exemple :

```
Create table personne(
idp  int primary key,
Nom varchar(20) not null,
Age  int,
Ville varchar(20)  check(ville IN('oujda','berkane','ahfir') )
)
```

check

- Dans la condition de la contrainte check, on peut utiliser la clause **between** et **NOT BETWEEN**: permet de définir un **intervalle de valeurs**
- Exemple:

```
Create table personne(
idp  int primary key,
Nom varchar(20) not null,
Age  int check(age between 18 and 30)
)
```

check

- Dans la condition de la contrainte check, on peut utiliser La clause **LIKE** et **NOT LIKE**:
- Cette clause signifie que le champ doit être **comme …**
  - % remplace plusieurs caractères
  - _ remplace un seul caractère
  - [a-g]: tout caractère entre a et g
  - [ag]: soit a soit g
  - [^a-g]:ne doit pas être entre a et g.

check

- Exemple:

```
Create table materiel(
CodeMat  varchar(10)  primary key
check(CodeMat  like 'Ser%'),
Nom varchar(20) not null,
)
```

Cela veut dire que le code du matériel doit **commencer** par **Ser,** suivi de n'importe quel caractères.

check

- Exemple 2:

```
Create table materiel(
CodeMat  varchar(10)  primary key
check(CodeMat like 'M%2018'),
Nom varchar(20) not null,
)
```

Cela veut dire que le code du matériel doit **commencer** par **M** et **se terminer par 2018**, peu importe ce qu'il a entre M et 2018.

check

- Exemple :

```
Create table materiel(
CodeMat  varchar(10)  primary key
check(CodeMat  like 'Ser__),
Nom varchar(20) not null,
)
```

Cela veut dire que le code du matériel doit **commencer** par **'Ser'**, suivi de **2 caractères**.

check

- Exemple :

```
Create table materiel(
CodeMat  varchar(10)  primary key check (CodeMat like '%2018'),
Nom varchar(20) not null,
)
```

Le code du matériel doit se **terminer** par **2018**, et peut commencer avec n'importe quels caractères

check

- Exemple:

```
Create table materiel(
CodeMat  varchar(10)  primary key  check(CodeMat
like '[A-D]%'),
Nom varchar(20) not null,
)
```

Le code du matériel doit commencer par A, ou B, ou C, ou D, et suivi de n'importe quel caractère

check

- Exemple:

```
Create table materiel(
CodeMat  varchar(10)  primary key  check(CodeMat like '[AB]%'),
Nom varchar(20) not null,
)
```

Le code du matériel doit commencer soit par A ou par D, et suivi de n'importe quel caractère.

check

- Exemple:

```
Create table materiel(
CodeMat  varchar(10)  primary key  check(CodeMat like '[^AD]%'),
Nom varchar(20) not null,
)
```

Le code du matériel ne doit pas commencer ni par A ni par D.