ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR

Membre de HONORIS UNITED UNIVERSITIES

École Marocaine des Sciences de l'Ingénieur

Filière: IIR

Les Bases de Données

Prof. Zakaria KHATAR

LANGAGE SQL

Introduction au langage SQL

SQL (Structured Query Language), ou langage de requêtes structurées, est un langage informatique destiné à la manipulation de bases de données au sein des SGBD et plus particulièrement des SGBDR.

SQL est composé trois sous-langages :

- **Le Langage de Définition de Données (LDD) :** Ce langage permet la définition et la mise à jour de la structure de la base de données (relations, attributs, ...).
- **Le Langage d'Interrogation de Données (LID) :** Ce langage permet de rechercher des informations utiles en interrogeant la base de données.
- **Le Langage de Manipulation de Données (LMD) :** Ce langage permet de manipuler les données de la base et de les mettre à jour.

LANGAGE SQL

Langage de Définition des Données : LDD

Les Types de données :

Type de Donnée	Description
CHAR(n)	Chaîne de caractères de longueur fixe n.
VARCHAR2(n)	Chaîne de caractères de longueur variable avec une longueur maximale de n caractères.
CLOB	Character Large Object, pour stocker des chaînes de caractères jusqu'à 4 Go.
NUMBER(p,s)	Nombre décimal avec p précision et s échelle.
FLOAT(p)	Nombre à virgule flottante de précision p.
DATE	Date (et heure à la seconde).
TIMESTAMP	Date et heure avec précision en fraction de seconde.
BLOB	Binary Large Object, pour stocker des données binaires jusqu'à 4 Go.
RAW(n)	Données binaires de longueur fixe jusqu'à 2000 octets.

Les Contraintes d'intégrité :

Contrainte	Description
PRIMARY KEY	Garantit que les valeurs d'une colonne (ou d'un ensemble de colonnes) sont uniques et non nulles. Chaque table ne peut avoir qu'une seule clé primaire.
UNIQUE	Garantit que les valeurs d'une colonne (ou d'un ensemble de colonnes) sont uniques. Une table peut avoir plusieurs contraintes uniques.
FOREIGN KEY - REFERENCES	La clé étrangère garantit que les valeurs d'une colonne correspondent aux valeurs d'une clé primaire d'une autre table. Le mot-clé **REFERENCES** est utilisé avec **FOREIGN KEY** pour définir la table et la colonne référencées.
CHECK	Garantit que les valeurs d'une colonne (ou d'un ensemble de colonnes) satisfont une condition spécifiée.
NOT NULL	Garantit qu'une colonne ne peut pas contenir de valeur NULL.
DEFAULT	Attribue une valeur par défaut à la colonne si aucune valeur n'est spécifiée lors de l'insertion d'un enregistrement.

Concept de la base de données et Schéma sous Oracle

Concept de la base de données et Schéma sous Oracle

Dans Oracle, une base de données est un ensemble structuré de données. Un schéma représente la collection d'objets d'un utilisateur. L'utilisateur est un compte avec des droits spécifiques pour accéder à la base de données. Donc on peut dire :

- **Base de données** : L'endroit physique où les données sont stockées, composé de fichiers.
- **Schéma** : Collection logique d'objets (tables, vues, etc.), nommé d'après un utilisateur.
- **Utilisateur** : Compte avec des droits pour accéder et manipuler les données.

Création des Tables

Création de tables

Syntaxe de base pour créer une table :

```
CREATE TABLE nom_de_la_table (
    nom_attribut_1 type_donnees contrainte_1,
    nom_attribut_2 type_donnees contrainte_2,
    nom_attribut_3 type_donnees,
    nom_attribut_4 type_donnees
);
```

Création de tables

Exemple de création de table :

```
CREATE TABLE employes (
    Code_etudiant NUMBER PRIMARY KEY,
    nom_etudiant VARCHAR2(100),
    age_etudiant NUMBER
);
```

Création de tables

Syntaxe de base pour créer une table :

```
CREATE TABLE nom_de_la_table (
    nom_attribut_1 type_donnees CONSTRAINT nom_contrainte1 type_contrainte,
    nom_attribut_2 type_donnees CONSTRAINT nom_contrainte2 type_contrainte
    (nom_colonne),
    nom_attribut_3 type_donnees,
    nom_attribut_4 type_donnees
);
```

Création de tables

Exemple de création de table :

```
CREATE TABLE employes (
   id_etudiant NUMBER CONSTRAINT pk_etudiant PRIMARY KEY,
   nom_etudiant VARCHAR2(100) NOT NULL,
   age NUMBER CONSTRAINT verification_age CHECK (age >= 18)
    );

CREATE TABLE employes (
   id_poste NUMBER CONSTRAINT emp_pk PRIMARY KEY,
   dept_id NUMBER,
   nom VARCHAR2(100)
   CONSTRAINT emp_fk_dept FOREIGN KEY (dept_id) REFERENCES
    departements(dept_id),
     );
```

Création de tables

Exemple de création de table :

```
CREATE TABLE Etudiant (
Code_Etu Number CONSTRAINT clé_primaire PRIMARY KEY,
Age Number CONSTRAINT verify_age CHECK(Age >= 0 AND Age < 130),
Filière Varchar2(20) CONSTRAINT choix_filière CHECK (Filière IN (‘Informatique’, ‘Automatique’, ‘Audit'))
);
```

Modification des Tables

Modification des tables

1. Ajouter une nouvelle colonne:

Vous pouvez utiliser la commande ALTER TABLE avec l'option **ADD** pour ajouter une nouvelle colonne à une table existante.

```
ALTER TABLE nom_table ADD (nouvelle_colonne type_donnees contrainte);
```

Exemple : ALTER TABLE employes **ADD (**adresse VARCHAR2(255)**);**

2. Supprimer une colonne:

La commande **ALTER TABLE** avec l'option **DROP COLUMN** permet de supprimer une colonne.

```
ALTER TABLE nom_table DROP COLUMN nom_colonne;
```

Exemple : ALTER TABLE employes **DROP COLUMN** adresse;

Modification des tables

3. Renommer une colonne:

Utilisez **ALTER TABLE** avec l'option **RENAME COLUMN** pour renommer une colonne.

```
ALTER TABLE nom_table RENAME COLUMN ancien_nom TO nouveau_nom;
```

Exemple : ALTER TABLE employes **RENAME COLUMN** adresse **TO** adresse_complete**;**

4. Modifier le type ou la taille d'une colonne:

La commande **ALTER TABLE** avec l'option **MODIFY** vous permet de changer le type de données ou la taille d'une colonne.

```
ALTER TABLE nom_table MODIFY (nom_colonne nouveau_type_donnees);
```

Exemple : ALTER TABLE employes **MODIFY (**adresse VARCHAR2(500)**)**;

Modification des Contraintes d'Intégrité

Modification des contraintes d'intégrité

1. Ajouter une contrainte:

Utilisez **ALTER TABLE** avec l'option **ADD CONSTRAINT** pour ajouter une nouvelle contrainte à une colonne.

```
ALTER TABLE nom_table ADD CONSTRAINT nom_contrainte TYPE_CONTRAINTE (nom_colonne);
```

Exemple : ALTER TABLE employes **ADD CONSTRAINT** emp_email **UNIQUE (email);**

2. Supprimer une contrainte:

La commande **ALTER TABLE** avec l'option **DROP CONSTRAINT** permet de supprimer une contrainte existante.

```
ALTER TABLE nom_table DROP CONSTRAINT nom_contrainte;
```

Exemple : ALTER TABLE employes **DROP CONSTRAINT** emp_email;

Modification des contraintes d'intégrité

3. Activer ou désactiver une contrainte:

Vous pouvez **ACTIVER** ou **DÉSACTIVER** des contraintes sans les supprimer.

```
ALTER TABLE nom_table ENABLE CONSTRAINT nom_contrainte;
ALTER TABLE nom_table DISABLE CONSTRAINT nom_contrainte;
```

Dans Oracle, les contraintes garantissent l'intégrité des données. Les commandes ENABLE et DISABLE permettent de gérer ces contraintes d'intégrité, offrant flexibilité lors d'opérations spécifiques tout en veillant à la qualité des données.