UNIVERSITE DE L'ASSOMPTION DU CONGO DE BENI

UAC/BENI

REPUBLIQUE DEMOCRATIQUE DU CONGO

FACULTE D'INFORMATIQUE APPLIQUEE A LA GESTION DES ENTREPRISES

TRAVAIL PRATIQUE DE BASES DE DONNEES

Par MUYISA VAGHENI EXAUCE

2024-2025

A.

1. **FOREIN KEY (Clé étrangère)** : Supposons qu'on a deux tables, **commandes** et **clients**, où **commandes** contient des colonnes telles que id_commande, id_client, date_commande, etc., et **clients** contient des colonnes comme id_client, nom_client, email, etc. Pour garantir que chaque id_client dans la table **commandes** corresponde à un id_client existant dans la table **clients**, vous définiriez une contrainte de clé étrangère sur la colonne id_client de la table **commandes**, qui fait référence à la colonne id_client de la table **clients**.
   Exemple de la requête :

   ➔ **CREATE TABLE** commandes (id_commande **INT(50) PRIMARY KEY**, id_client **INT(20)**, date_commande **DATE**, **CONSTRAINT** fk_id_client **FOREIGN KEY** (id_client) **REFERENCES** clients(id_client));

2. **NOT NULL (non nul)** : Supposons que vous ayez une table **employes** avec des colonnes telles que id_employe, nom, prenom , email, etc., et vous souhaitez vous assurer que le email de chaque employé est obligatoire (non nul).
   Exemple de la requête :

   ➔ **CREATE TABLE** employes (id_employe **INT(11) PRIMARY KEY**, nom **VARCHAR(255) NOT NULL**, postnom **VARCHAR(255) NOT NULL**, email **VARCHAR(255) NOT NULL**);

3. UNIQUE (unique) : Supposons que vous ayez une table **etudiants** avec des colonnes telles que id_etudiant, nom, postnom, etc., et vous voulez vous assurer qu'aucun nom d'étudiant n'est répété.
Exemple de la requête :

➔ **CREATE TABLE** etudiants (id_etudiant **INT(30) PRIMARY KEY**, nom **VARCHAR(255) NOT NULL**, last_name **VARCHAR(255) NOT NULL, UNIQUE(**nom, postnom**)**);

4. CHECK (Vérifier) : Supposons que vous ayez une table **employes** avec une colonne salaire où vous voulez vous assurer que le salaire de chaque employé est supérieur à 20000.

➔ **CREATE TABLE** employes (id_employe **INT(30) PRIMARY KEY**, nom **VARCHAR(255) NOT NULL,** postnom **VARCHAR(255) NOT NULL**, salaire **DECIMAL(10, 2) CHECK (**salary > 20000) **)**;

B. Liste des instructions :

- Pour créer un synonyme :
  ➔ **CREATE SYNONYM** nom_synonyme **FOR** nom_objet;

- Pour créer un indexe:
  ➔ **CREATE INDEX** nom_index **ON** nom_table (nom_colonne);

- Pour créer une sequence:
  ➔ **CREATE SEQUENCE** nom_sequence **START WITH** valeur_depart **INCREMENT BY** valeur_incrementation;

C. Quelques fonctions de groupes de SQL :

1. **COUNT() :** Cette fonction compte le nombre de lignes dans un groupe.
   Exemple : Compter le nombre d'étudiants par classe.

2. **SUM() :** Cette fonction calcule la somme des valeurs d'une colonne dans un groupe.
   Exemple : Calculer le total des ventes par produit.

3. **AVG() :** Cette fonction calcule la moyenne des valeurs d'une colonne dans un groupe.
   Exemple : Calculer la moyenne des notes par matière.

4. **MIN() :** Cette fonction retourne la valeur minimale d'une colonne dans un groupe.
   Exemple : Trouver la note minimale par étudiant.

5. **MAX() :** Cette fonction retourne la valeur maximale d'une colonne dans un groupe.
   Exemple : Trouver la note maximale par étudiant.

6. **GROUP_CONCAT() :** Cette fonction concatène les valeurs d'une colonne dans un groupe en une seule chaîne de caractères.
   Exemple : Obtenir la liste des hobbies séparés par des virgules par étudiant.