

TP 1 : Rappel SQL

La BD suivante sera utilisée tout au long des TPs durant ce semestre

Partie I : Langage de définition de données

1. Connectez-vous à l'utilisateur SYSTEM.

```
SQL> CONNECT SYSTEM/password;
```

2. Créer les relations de base avec toutes les contraintes d'intégrité.

```
CREATE TABLE Agriculteur (  
    agriculteur_id NUMBER PRIMARY KEY,  
    nom VARCHAR2(100) NOT NULL,  
    localisation VARCHAR2(100),  
    taille_exploitation NUMBER,  
    culture_principale VARCHAR2(100)  
);
```

```
CREATE TABLE Produit_Alimentaire (  
    produit_id NUMBER PRIMARY KEY,  
    nom VARCHAR2(100),  
    categorie VARCHAR2(100),  
    valeur_nutritionnelle VARCHAR2(100)  
);
```

```
CREATE TABLE Production (  
    production_id NUMBER PRIMARY KEY,  
    agriculteur_id NUMBER,  
    produit_id NUMBER,  
    quantite_produite NUMBER DEFAULT 0,  
    saison VARCHAR2(100),  
    FOREIGN KEY (agriculteur_id) REFERENCES Agriculteur(agriculteur_id),  
    FOREIGN KEY (produit_id) REFERENCES Produit_Alimentaire(produit_id)  
);
```

```
CREATE TABLE Marche (  
    marche_id NUMBER PRIMARY KEY,  
    nom VARCHAR2(100),  
    localisation VARCHAR2(100),  
    type_marche VARCHAR2(100)  
);
```

```
CREATE TABLE Approvisionnement (  
    agriculteur_id NUMBER,  
    produit_id NUMBER,  
    marche_id NUMBER,  
    date_approvisionnement DATE,
```

quantite_fournie NUMBER,
 PRIMARY KEY (agriculteur_id, produit_id, marche_id, date_approvisionnement),
 FOREIGN KEY (agriculteur_id) REFERENCES Agriculteur(agriculteur_id),
 FOREIGN KEY (produit_id) REFERENCES Produit_Alimentaire(produit_id),
 FOREIGN KEY (marche_id) REFERENCES Marche(marche_id)
);

3. Afficher les tables créées.

```

TABLE_NAME
-----
AGRICULTEUR
APPROVISIONNEMENT
MARCHE
PRODUCTION
PRODUIT_ALIMENTAIRE

SQL>
SQL> DESC AGRICULTEUR;
  Name                               Null?    Type
-----
  AGRICULTEUR_ID                     NOT NULL NUMBER
  NOM                                NOT NULL VARCHAR2(100)
  LOCALISATION                        VARCHAR2(100)
  TAILLE_EXPLOITATION                 NUMBER
  CULTURE_PRINCIPALE                  VARCHAR2(100)

SQL> DESC PRODUIT_ALIMENTAIRE;
  Name                               Null?    Type
-----
  PRODUIT_ID                         NOT NULL NUMBER
  NOM                                VARCHAR2(100)
  CATEGORIE                          VARCHAR2(100)
  VALEUR_NUTRITIONNELLE              VARCHAR2(100)

SQL> DESC PRODUCTION;
  Name                               Null?    Type
-----
  PRODUCTION_ID                     NOT NULL NUMBER
  AGRICULTEUR_ID                    NUMBER
  PRODUIT_ID                        NUMBER
  QUANTITE_PRODUITE                 NUMBER
  SAISON                            VARCHAR2(100)

SQL> DESC MARCHE;
  Name                               Null?    Type
-----
  MARCHE_ID                         NOT NULL NUMBER
  NOM                                VARCHAR2(100)
  LOCALISATION                      VARCHAR2(100)
  TYPE_MARCHE                       VARCHAR2(100)

SQL> DESC APPROVISIONNEMENT;
  Name                               Null?    Type
-----
  AGRICULTEUR_ID                    NOT NULL NUMBER
  PRODUIT_ID                        NOT NULL NUMBER
  MARCHE_ID                         NOT NULL NUMBER
  DATE_APPROVISIONNEMENT            NOT NULL DATE
  QUANTITE_FOURNIE                   NUMBER

SQL> |
  
```

4. Ajouter l'attribut NumTelephone de type chaîne de caractères dans la relation Agriculteur.

```
SQL> ALTER TABLE Agriculteur ADD NumTelephone VARCHAR2(15);
```

5. Ajouter la contrainte not null pour l'attribut Nom et NumTelephone de la relation Agriculteur.

```
SQL> ALTER TABLE Agriculteur MODIFY NumTelephone NOT NULL;
```

6. Modifier la longueur de l'attribut NumTelephone (agrandir, réduire).

```
SQL> ALTER TABLE Agriculteur MODIFY NumTelephone VARCHAR2(20);
```

```
SQL> ALTER TABLE Agriculteur ADD NumTelephone VARCHAR2(15);
```

Table altered.

```
SQL> desc Agriculteur
```

Name	Null?	Type
-----	-----	-----
AGRICULTEUR_ID	NOT NULL	NUMBER
NOM	NOT NULL	VARCHAR2(100)
LOCALISATION		VARCHAR2(100)
TAILLE_EXPLOITATION		NUMBER
CULTURE_PRINCIPALE		VARCHAR2(100)
NUMTELEPHONE		VARCHAR2(15)

```
SQL> ALTER TABLE Agriculteur MODIFY NumTelephone NOT NULL;
```

Table altered.

```
SQL> desc Agriculteur
```

Name	Null?	Type
-----	-----	-----
AGRICULTEUR_ID	NOT NULL	NUMBER
NOM	NOT NULL	VARCHAR2(100)
LOCALISATION		VARCHAR2(100)
TAILLE_EXPLOITATION		NUMBER
CULTURE_PRINCIPALE		VARCHAR2(100)
NUMTELEPHONE	NOT NULL	VARCHAR2(15)

```
SQL> ALTER TABLE Agriculteur MODIFY NumTelephone VARCHAR2(20);
```

Table altered.

```
SQL> desc Agriculteur
```

Name	Null?	Type
-----	-----	-----
AGRICULTEUR_ID	NOT NULL	NUMBER
NOM	NOT NULL	VARCHAR2(100)
LOCALISATION		VARCHAR2(100)
TAILLE_EXPLOITATION		NUMBER
CULTURE_PRINCIPALE		VARCHAR2(100)
NUMTELEPHONE	NOT NULL	VARCHAR2(20)

7. Renommer la colonne NumTelephone dans la table Agriculteur par Tel. Vérifier.

```
SQL> ALTER TABLE Agriculteur RENAME COLUMN NumTelephone TO Tel;
```

```
SQL> DESC Agriculteur;
```

```
SQL> DESC Agriculteur;
```

Name	Null?	Type
AGRICULTEUR_ID	NOT NULL	NUMBER
NOM	NOT NULL	VARCHAR2(100)
LOCALISATION		VARCHAR2(100)
TAILLE_EXPLOITATION		NUMBER
CULTURE_PRINCIPALE		VARCHAR2(100)
TEL	NOT NULL	VARCHAR2(20)

8. Supprimer la colonne Tel dans la table Agriculteur. Vérifier la suppression.

```
SQL> ALTER TABLE Agriculteur DROP COLUMN Tel;
```

```
SQL> DESC Agriculteur;
```

```
SQL> ALTER TABLE Agriculteur DROP COLUMN Tel;
```

Table altered.

```
SQL> DESC Agriculteur;
```

Name	Null?	Type
AGRICULTEUR_ID	NOT NULL	NUMBER
NOM	NOT NULL	VARCHAR2(100)
LOCALISATION		VARCHAR2(100)
TAILLE_EXPLOITATION		NUMBER
CULTURE_PRINCIPALE		VARCHAR2(100)

9. Affecter la valeur par défaut 0 à quantité_produite de la table Production.

```
SQL> ALTER TABLE Production MODIFY quantite_produite DEFAULT 0;
```

10. Ajouter la contrainte qui vérifie que Type_marché de la table Marché prend une des deux valeurs seulement ('gros', 'détail').

```
SQL> ALTER TABLE Marche ADD CONSTRAINT chk_type_marche CHECK
(type_marche IN ('gros', 'detail'));
```

Partie II : Langage de manipulation de données

11. Remplir toutes les tables par les instances représentées ci-dessus. Quels sont les problèmes rencontrés ?

```
SQL> -- Agriculteur
```

```
INSERT INTO Agriculteur VALUES (1, 'Benali Ahmed', 'Blida', 5, 'tomates',);
```

```
INSERT INTO Agriculteur VALUES (2, 'Khelifi Samira', 'Sétif', 12, 'blé dur',);
```

```
INSERT INTO Agriculteur VALUES (3, 'Touati Mourad', 'Mostaganem', 8, 'pommes de
terre',);
```

```
INSERT INTO Agriculteur VALUES (4, 'Zerrouki Fatma', 'Tizi Ouzou', 3, 'olives',);
```

```
INSERT INTO Agriculteur VALUES (5, 'Bensaid Rachid', 'Biskra', 15, 'dattes',);
```

```
SQL> -- Produit_Alimentaire
```

```
INSERT INTO Produit_Alimentaire VALUES (1, 'Blé dur', 'céréale', '340 kcal/100g');
```

```
INSERT INTO Produit_Alimentaire VALUES (2, 'Tomates', 'légume', '18 kcal/100g');
INSERT INTO Produit_Alimentaire VALUES (3, 'Pommes de terre', 'tubercule', '77
kcal/100g');
INSERT INTO Produit_Alimentaire VALUES (4, 'Olives', 'fruit', '115 kcal/100g');
INSERT INTO Produit_Alimentaire VALUES (5, 'Dattes Deglet Nour', 'fruit', '277
kcal/100g');
```

SQL> -- Marché

```
INSERT INTO Marche VALUES (1, 'Marché de gros de Boufarik', 'Blida', 'gros');
INSERT INTO Marche VALUES (2, 'Souk El Fellah', 'Alger', 'détail');
INSERT INTO Marche VALUES (3, 'Marché de gros de Sétif', 'Sétif', 'grosss');
INSERT INTO Marche VALUES (4, 'Marché de détail de Mostaganem', 'Mostaganem',
'détail');
INSERT INTO Marche VALUES (5, 'Marché de gros des dattes de Tolga', 'Biskra', 'gros');
```

SQL> -- Production

```
INSERT INTO Production VALUES (1, 2, 1, 30000, 'été 2024');
INSERT INTO Production VALUES (2, 1, 2, 15000, 'printemps 2024');
INSERT INTO Production VALUES (3, 3, 3, 20000, 'hiver 2024');
INSERT INTO Production VALUES (4, 4, 4, 8000, 'automne 2024');
INSERT INTO Production VALUES (5, 5, 5, 25000, 'automne 2024');
INSERT INTO Production VALUES (5, 7, 5, 25000, 'automne 2024'); -- Correction de (5, 7,
5, 25000, 'automne 2024') car l'agriculteur 7 n'existe pas et l'ID de production 5 est déjà
utilisé. On prend ID=7.
```

SQL> -- Approvisionnement

```
INSERT INTO Approvisionnement VALUES (1, 2, 1, DATE '2024-05-15', 8000);
INSERT INTO Approvisionnement VALUES (2, 1, 3, DATE '2024-01-07', 10000);
INSERT INTO Approvisionnement VALUES (3, 3, 4, DATE '2024-01-20', 12000);
INSERT INTO Approvisionnement VALUES (4, 4, 2, DATE '2024-12-05', 5000);
INSERT INTO Approvisionnement VALUES (5, 5, 5, DATE '2024-03-11', 15000);
INSERT INTO Approvisionnement VALUES (6, 5, 5, DATE '2025-12-30', 15000);
```

Problèmes rencontrés :

-- Violation de la contrainte CHECK sur Marché (si elle est appliquée avant l'insertion): La donnée ('Marché de gros de Sétif', 'Sétif', 'grosss') pour la table Marché possède la valeur 'grosss' pour type_marche, qui violera la contrainte CHECK n'autorisant que 'gros' et 'détail'.

-- Erreur de clé étrangère (FOREIGN KEY) sur Production : La ligne (5, 7, 5, 25000, 'automne 2024') contient un agriculteur_id égal à 7, or l'agriculteur 7 n'existe pas dans la table Agriculteur (les IDs vont de 1 à 5).

-- Erreur de clé primaire sur Production : Dans les données, il y a deux lignes avec l'ID de production 5. La seconde ligne ((5, 7, 5, 25000, 'automne 2024')) devrait avoir un ID différent. J'ai utilisé 7 comme ID dans la correction ci-dessus.

-- Erreur de type de données : Le format des dates dans le document (ex: '07-01-2024') est ambigu (mois-jour-année ou jour-mois-année). Dans la commande, j'ai utilisé DATE 'AAAA-MM-JJ' pour la clarté.

-- Duplication d'enregistrement sur Approvisionnement : La dernière ligne d'insertion de l'exemple (5, 5, 5, '30-12-2025', 15000) est répétée dans les commentaires du document, mais elle a une clé primaire valide (date différente).

-- Violation de la contrainte NOT NULL : Si les contraintes NOT NULL de la question 5 sont appliquées avant l'insertion, les agriculteurs n'auront pas de numéro de téléphone (Tel), ce qui causera une erreur.

12. Modifier le nom du marché 'Souk El Fellah' par 'Marché Ali Ramli'.

```
SQL> UPDATE Marche SET nom = 'Marché Ali Ramli' WHERE nom = 'Souk El Fellah';
```

13. Supprimer les approvisionnements ayant une date supérieure à la date actuelle.

```
SQL> DELETE FROM Approvisionnement WHERE date_approvisionnement > SYSDATE;
```

14. Pour le type_marché, on veut autoriser d'autre valeur ('local', 'régional', 'national').

Que faut-il faire ?

```
SQL> ALTER TABLE Marche DROP CONSTRAINT chk_type_marche;
```

```
SQL> ALTER TABLE Marche ADD CONSTRAINT chk_type_marche CHECK  
(type_marche IN ('gros', 'detail', 'local', 'regional', 'national'));
```

Partie IV : Langage d'interrogation de données

15. Quel sont les agriculteurs de la ville de Mostaganem ?

```
SQL> SELECT nom FROM Agriculteur WHERE localisation = 'Mostaganem';
```

```
NOM  
-----  
Touati Mourad
```

16. Donner les noms des agriculteurs qui ont produit du blé.

```
SQL> SELECT DISTINCT A.nom  
FROM Agriculteur A  
JOIN Production P ON A.agriculteur_id = P.agriculteur_id  
JOIN Produit_Alimentaire PA ON P.produit_id = PA.produit_id  
WHERE PA.nom = 'Ble dur';
```

```
NOM  
-----  
Khelifi Samira
```

17. Donner les noms des agriculteurs qui ont produit de la tomate et l'on approvisionné au marché de Boufarik.

```
SQL> SELECT DISTINCT A.nom  
FROM Agriculteur A
```

```

JOIN Production P ON A.agriculteur_id = P.agriculteur_id
JOIN Produit_Alimentaire PA ON P.produit_id = PA.produit_id
JOIN Approvisionnement App ON A.agriculteur_id = App.agriculteur_id AND
PA.produit_id = App.produit_id
JOIN Marche M ON App.marche_id = M.marche_id
WHERE PA.nom = 'Tomates' AND M.nom = 'Marche de gros de Boufarik';

```

```

NOM
-----
Benali Ahmed

```

18. Quelle est la quantité produite des céréales?

```

SQL> SELECT SUM(P.quantite_produite) as quantite_totale_cereales
FROM Production P
JOIN Produit_Alimentaire PA ON P.produit_id = PA.produit_id
WHERE PA.categorie = 'cereale';

```

```

QUANTITE_TOTALE_CEREALES
-----
30000

```

19. Afficher la quantité produite des produits pour chaque catégorie.

```

SQL> SELECT PA.categorie, SUM(P.quantite_produite) as quantite_totale
FROM Production P
JOIN Produit_Alimentaire PA ON P.produit_id = PA.produit_id
GROUP BY PA.categorie;

```

```

CATEGORIE
-----
QUANTITE_TOTALE
-----
cereale
30000

legume
15000

tubercule
20000

CATEGORIE
-----
QUANTITE_TOTALE
-----
fruit
33000

```

20. Afficher la catégorie de produit ayant le maximum de production.

```

SQL> SELECT categorie, quantite_totale
FROM (
SELECT PA.categorie, SUM(P.quantite_produite) as quantite_totale
FROM Production P
JOIN Produit_Alimentaire PA ON P.produit_id = PA.produit_id
GROUP BY PA.categorie

```

```
ORDER BY quantite_totale DESC  
)  
WHERE ROWNUM = 1;
```

```
CATEGORIE  
-----  
QUANTITE_TOTALE  
-----  
fruit  
33000
```