5. **ANNEXE – SQL et Modélisation des Données**

5.1 Rappels sur les requêtes SQL

Principales actions SQL

Action SQL	Description	Syntaxe générique	Exemple générique
Sélectionner des données	Affiche des données d'une table	`SELECT colonnes FROM table;`	`SELECT nom FROM UTILISATEUR;`
Sélectionner toutes les colonnes	Affiche toutes les colonnes d'une table	`SELECT * FROM table;`	`SELECT * FROM PRODUIT;`
Filtrer les résultats	Limite les résultats selon une condition	`WHERE condition`	`WHERE quantite > 10;`
Jointure entre tables	Associe les données de plusieurs tables	`JOIN table2 ON condition`	`JOIN COMMANDE ON LIGNE.id_commande = COMMANDE.id_commande;`
Insérer des données	Ajoute une nouvelle ligne	`INSERT INTO table (colonnes) VALUES (valeurs);`	`INSERT INTO UTILISATEUR VALUES (1, 'Nom', 'Email');`
Modifier des données	Met à jour une ou plusieurs lignes	`UPDATE table SET colonne=valeur WHERE condition;`	`UPDATE PRODUIT SET prix=12.5 WHERE id_produit=3;`
Supprimer des données	Supprime une ou plusieurs lignes	`DELETE FROM table WHERE condition;`	`DELETE FROM COMMANDE WHERE id_commande=5;`

> **Note** : l'absence de clause `WHERE` dans une requête `UPDATE` ou `DELETE` entraîne la modification ou la suppression de **toutes les lignes** de la table.

5.2 Modélisation des données (MCD / MLD)

a. Entité

Une **entité** représente un objet du monde réel que l'on souhaite stocker dans la base de données.

Exemples :

- Client
- Technicien
- Produit

> En base de données relationnelle, une entité devient une **table**.

b. Attribut

Un **attribut** est une information qui décrit une entité.

Exemples :

- nom
- adresse
- date
- quantité

c. Clé primaire (Primary Key – PK)

La **clé primaire** permet d'identifier **de manière unique** chaque enregistrement d'une table.

Caractéristiques :

- valeur **unique**
- **non NULL**
- une seule clé primaire par table

d. Relation et cardinalités

Une **relation** relie deux entités entre elles. Les **cardinalités** indiquent le nombre minimum et maximum de liens possibles.

Cardinalité	Signification
0..1	zéro ou un
1..1	un et un seul
0..N	zéro, un ou plusieurs
1..N	au moins un

- Question à se poser :

> *« Pour une occurrence de l'entité A, combien d'occurrences de l'entité B ? »*

e. Clé étrangère (Foreign Key – FK)

Une **clé étrangère** est un attribut :

- présent dans une table
- qui référence la **clé primaire d'une autre table**

Elle permet de :

- créer des **liens entre les tables**
- assurer la **cohérence des données**

f. Intégrité référentielle

Les règles d'intégrité référentielle permettent :

- d'empêcher l'ajout de données incohérentes
- de garantir l'existence des données liées

Exemples :

- impossible d'ajouter une ligne avec une clé étrangère inexistante
- la suppression d'une ligne référencée peut être refusée

5.3 Fiche méthode – Lire un diagramme MCD

Étape 1 – Repérer les entités

- Les entités sont représentées par des **rectangles**
- Elles correspondent aux futures **tables** de la base de données

Étape 2 – Identifier la clé primaire

- Attribut :
  - souligné
  - ou clairement indiqué comme identifiant
- Il permet d'identifier chaque occurrence de façon unique

Étape 3 – Lire les relations

- Les relations relient les entités entre elles
- Elles traduisent un lien logique (ex. : *possède*, *réalise*, *concerne*)

Étape 4 – Interpréter les cardinalités

- Les cardinalités indiquent :
  - le minimum
  - le maximum de liens possibles

Règle essentielle :

> **Relation 1,N → clé étrangère du côté N**

Étape 5 – Faire le lien avec les tables SQL

- Les entités deviennent des tables
- Les relations 1,N se traduisent par des clés étrangères
- Les clés étrangères permettent les **jointures SQL**

5.4 Passer du MCD au MLD

1. Transformer chaque entité en table
2. Reporter les attributs en colonnes
3. Identifier la clé primaire
4. Ajouter les clés étrangères pour les relations 1,N
5. Vérifier la cohérence globale

5.5 Écrire une jointure SQL

Principe

Une **jointure** permet d'associer des données provenant de plusieurs tables grâce aux clés étrangères.

Syntaxe générale

```
SELECT colonnes
FROM table1
JOIN table2
ON table1.cle_etrangere = table2.cle_primaire;
```

Avec condition

```
WHERE condition;
```

Erreurs fréquentes à éviter

- oublier la clause `ON`
- confondre clé primaire et clé étrangère
- joindre des colonnes sans lien logique
- utiliser `SELECT *` sans justification

Résumé

- Une entité → une table
- Une table → une clé primaire
- Une relation 1,N → clé étrangère côté N
- Les clés étrangères assurent la cohérence
- Les jointures permettent d'exploiter les relations
