# Exercice – MCD : Gestion des Livres en Bibliothèque

## Objectif

Modéliser un **Modèle Conceptuel de Données (MCD)** pour un système de **gestion de prêts** dans une bibliothèque municipale.

---

## Contexte

Une bibliothèque souhaite informatiser la gestion des **emprunts de livres** par ses **adhérents**. Chaque **livre** peut être emprunté plusieurs fois, **mais pas simultanément**. Chaque **adhérent** peut emprunter **plusieurs livres à la fois**, à condition de les **rendre dans les délais**. L'historique complet des emprunts doit être **conservé**.

---

## Données métier

### Livre

- Code ISBN
- Titre
- Auteur
- Éditeur
- Date de publication

### Adhérent

- Numéro d'adhérent
- Nom
- Prénom
- Date d'inscription
- Email

### Emprunt

- Date d'emprunt
- Date de retour prévue
- Date de retour effective *(si le livre a été rendu)*

---

## Contraintes

- Un **livre** ne peut être **emprunté que par un seul adhérent à la fois**.
- Un **adhérent** peut avoir **plusieurs emprunts simultanés**.
- Chaque **emprunt** concerne **un seul livre et un seul adhérent**.
- L'**historique des emprunts** doit être **intégralement conservé**.

---

## Travail demandé

1. Identifier les **entités** et leurs **attributs**.
2. Définir les **relations** entre les entités.
3. Spécifier les **cardinalités**.
4. Justifier l'usage éventuel d'une **entité-association**.
5. Dessiner un **MCD simplifié**.

---

## Aide-mémoire

| Élément | Exemple |
|---|---|
| Entités | `LIVRE`, `ADHERENT` |
| Association | `EMPRUNT` |
| Cardinalité | 1,n – 0,n |
| Attributs | `email`, `date_retour`, `titre`, etc. |

---

# Corrigé

## 1. Entités et attributs

### LIVRE

- **Identifiant** : `isbn`
- **Attributs** :
  - `titre`
  - `auteur`
  - `editeur`
  - `date_publication`

### ADHERENT

- **Identifiant** : `id_adherent`
- **Attributs** :
  - `nom`
  - `prenom`
  - `email`
  - `date_inscription`

### EMPRUNT *(entité-association)*

- **Identifiant** : `id_emprunt` (ou clé composite, mais un identifiant simple est recommandé)

- **Attributs** :
  - `date_emprunt`
  - `date_retour_prevue`
  - `date_retour_effective`

> **EMPRUNT est une entité-association**, car elle relie deux entités **avec ses propres attributs**.

---

## 2. Relations et cardinalités

| Relation | Description | Cardinalité |
|---|---|---|
| `ADHERENT` → `EMPRUNT` | Un adhérent peut effectuer plusieurs emprunts | 1,n |
| `LIVRE` → `EMPRUNT` | Un livre peut faire l'objet de plusieurs emprunts | 1,n |
| `EMPRUNT` → `ADHERENT` | Chaque emprunt est lié à un seul adhérent | 1,1 |
| `EMPRUNT` → `LIVRE` | Chaque emprunt concerne un seul livre | 1,1 |

---

## 3. Justification de l'entité-association

**EMPRUNT** est une **entité-association nécessaire** car :

- Elle **relie deux entités** (`ADHERENT`, `LIVRE`),
- Elle comporte **des attributs spécifiques à la relation**,
- Elle représente un **événement métier** avec sa propre **identité**.

---

## 4. MCD simplifié (texte et graphique)

### Version texte (compatible merise.fr)

```
ENTITE ADHERENT (#id_adherent: INT, nom: VARCHAR, prenom: VARCHAR, email:
VARCHAR, date_inscription: DATE)
ENTITE LIVRE (#isbn: VARCHAR, titre: VARCHAR, auteur: VARCHAR, editeur:
VARCHAR, date_publication: DATE)
ENTITE EMPRUNT (#id_emprunt: INT, date_emprunt: DATE, date_retour_prevue:
DATE, date_retour_effective: DATE)

ASSOCIATION emprunte (ADHERENT 1,n --- 1,1 EMPRUNT)
ASSOCIATION concerne (LIVRE 1,n --- 1,1 EMPRUNT)
```

### Schéma Merise (simplifié)

```
erDiagram
    ADHERENT ||--o{ EMPRUNT : "emprunte"
    LIVRE ||--o{ EMPRUNT : "concerne"
```

```
    ADHERENT {
        int id_adherent PK
        string nom
        string prenom
        string email
        date date_inscription
    }

    LIVRE {
        string isbn PK
        string titre
        string auteur
        string editeur
        date date_publication
    }

    EMPRUNT {
        int id_emprunt PK
        date date_emprunt
        date date_retour_prevue
        date date_retour_effective
    }
```

---

## 5. Transformation MCD → MPD

### Règles de transformation

- Chaque **entité** devient une **table**.
- Une **entité-association** avec attributs devient aussi une **table**.
- Les **relations** sont traduites en **clés étrangères**.
- Les **cardinalités** orientent les **contraintes** (`NOT NULL`, unicité…).

---

## 6. Script SQL

### Création des tables

```sql
-- Suppression des tables si elles existent
DROP TABLE IF EXISTS emprunts;
DROP TABLE IF EXISTS livres;
DROP TABLE IF EXISTS adherents;

-- Table des adhérents
CREATE TABLE adherents (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nom VARCHAR(255) NOT NULL,
    prenom VARCHAR(255) NOT NULL,
    email VARCHAR(255),
    date_inscription DATE
```

```
);

-- Table des livres
CREATE TABLE livres (
    isbn VARCHAR(20) PRIMARY KEY,
    titre VARCHAR(255) NOT NULL,
    auteur VARCHAR(255) NOT NULL,
    editeur VARCHAR(255),
    date_publication DATE
);

-- Table des emprunts
CREATE TABLE emprunts (
    id INT AUTO_INCREMENT PRIMARY KEY,
    adherent_id INT NOT NULL,
    isbn VARCHAR(20) NOT NULL,
    date_emprunt DATE NOT NULL,
    date_retour_prevue DATE,
    date_retour_effective DATE,
    FOREIGN KEY (adherent_id) REFERENCES adherents(id),
    FOREIGN KEY (isbn) REFERENCES livres(isbn)
);
```

---

Jeu de données

```
-- Insertion : adhérents
INSERT INTO adherents (nom, prenom, email, date_inscription) VALUES
('Dupont', 'Jean', 'jean.dupont@example.com', '2023-01-10'),
('Martin', 'Claire', 'claire.martin@example.com', '2024-03-15'),
('Nguyen', 'Paul', 'paul.nguyen@example.com', '2022-09-20');

-- Insertion : livres
INSERT INTO livres (isbn, titre, auteur, editeur, date_publication) VALUES
('9782070612751', '1984', 'George Orwell', 'Gallimard', '1949-06-08'),
('9782253006329', 'L'Étranger', 'Albert Camus', 'Folio', '1942-05-19'),
('9782266285989', 'Le Petit Prince', 'Antoine de Saint-Exupéry', 'Reynal &
Hitchcock', '1943-04-06');

-- Insertion : emprunts
INSERT INTO emprunts (adherent_id, isbn, date_emprunt, date_retour_prevue,
date_retour_effective) VALUES
(1, '9782070612751', '2025-08-10', '2025-08-20', '2025-08-18'),
(2, '9782253006329', '2025-08-15', '2025-08-25', NULL),
(3, '9782266285989', '2025-08-20', '2025-08-30', NULL);
```

---

Exemple de requête : **livres actuellement empruntés**

Cette requête liste les livres **non encore rendus**, en indiquant l'adhérent et la date d'emprunt :

```sql
SELECT
    l.titre,
    a.nom,
    a.prenom,
    e.date_emprunt
FROM
    emprunts e
JOIN
    livres l ON e.isbn = l.isbn
JOIN
    adherents a ON e.adherent_id = a.id
WHERE
    e.date_retour_effective IS NULL;
```

**Résultat attendu (exemple)**

| titre | nom | prenom | date_emprunt |
|---|---|---|---|
| L'Étranger | Martin | Claire | 2025-08-15 |
| Le Petit Prince | Nguyen | Paul | 2025-08-20 |

---

## Résumé des éléments livrables

| Élément | Contenu fourni |
|---|---|
| MCD | Modèle texte + Merise + Mermaid |
| MPD | Tables SQL avec clés et contraintes |
| Jeu de données | INSERTs pour 3 adhérents, 3 livres, 3 emprunts |
| Requêtes d'exemple | Liste des livres empruntés |
| Bonus | Propositions d'évolution du MCD |

---

# TP Modélisation & SQL – Parc Informatique

### Objectifs

Ce TP vous permettra de :

- Comprendre et manipuler un **Modèle Conceptuel de Données (MCD)**,
- Passer du **MCD** au **MLD**, puis aux requêtes **SQL**,
- Renforcer vos compétences en **relations entre tables**, **jointures**, et **cardinalités**,
- Apprendre à créer des **requêtes métiers utiles** dans un environnement informatique réel (réseau d'entreprise).

---

## Contexte

Vous êtes en charge de la base de données d'un service informatique. Cette base permet de **suivre les connexions des utilisateurs** aux différents ordinateurs du réseau d'entreprise.

Chaque utilisateur peut se connecter à plusieurs ordinateurs, et chaque ordinateur peut être utilisé par plusieurs utilisateurs. Chaque **connexion** est enregistrée avec une **date**, une **heure de début**, et une **heure de fin**.

---

## Partie 1 – MCD et compréhension du modèle

1. **Donner le nom des trois entités principales du MCD.**
2. **Indiquer les attributs principaux de chaque entité.**
3. **Quelles sont les cardinalités entre `UTILISATEUR` et `CONNEXION` ? Et entre `ORDINATEUR` et `CONNEXION` ?**
4. **Quel est le rôle métier de l'entité `CONNEXION` ? Justifie-t-elle sa propre table ou pourrait-elle être une simple association sans attributs ?**

---

## Partie 2 – MLD & Script de Création SQL

5. En vous appuyant sur le MLD suivant, écrivez les requêtes **SQL de création** des tables :

```
UTILISATEUR (
  id_utilisateur INT PRIMARY KEY,
  nom VARCHAR(50),
  prenom VARCHAR(50),
  service VARCHAR(50),
  email VARCHAR(100)
)

ORDINATEUR (
  id_ordinateur INT PRIMARY KEY,
  nom_ordinateur VARCHAR(50),
  adresse_ip VARCHAR(15),
  systeme_exploitation VARCHAR(50),
  date_installation DATE
)

CONNEXION (
  id_connexion INT PRIMARY KEY,
  id_utilisateur INT,
  id_ordinateur INT,
  date_connexion DATE,
  heure_debut TIME,
  heure_fin TIME,
  FOREIGN KEY (id_utilisateur) REFERENCES UTILISATEUR(id_utilisateur),
  FOREIGN KEY (id_ordinateur) REFERENCES ORDINATEUR(id_ordinateur)
)
```

## Partie 3 – Insertion de données fictives

6. Écrivez les requêtes **INSERT INTO** pour insérer les données suivantes :

**Table `UTILISATEUR`**

| id_utilisateur | nom | prenom | service | email |
|---|---|---|---|---|
| 1 | Dupont | Alice | RH | alice.dupont@entreprise.com |
| 2 | Martin | Jean | Informatique | jean.martin@entreprise.com |
| 3 | Lopez | Maria | Compta | maria.lopez@entreprise.com |

**Table `ORDINATEUR`**

| id_ordinateur | nom_ordinateur | adresse_ip | systeme_exploitation | date_installation |
|---|---|---|---|---|
| 101 | PC-RH-01 | 192.168.1.10 | Windows 10 Pro | 2022-03-15 |
| 102 | PC-IT-01 | 192.168.1.20 | Ubuntu 22.04 | 2023-01-10 |
| 103 | PC-COMPTA-01 | 192.168.1.30 | Windows 11 | 2024-09-05 |

**Table `CONNEXION`**

| id_connexion | id_utilisateur | id_ordinateur | date_connexion | heure_debut | heure_fin |
|---|---|---|---|---|---|
| 1001 | 1 | 101 | 2025-08-28 | 08:45:00 | 12:15:00 |
| 1002 | 2 | 102 | 2025-08-28 | 09:00:00 | 17:30:00 |
| 1003 | 3 | 103 | 2025-08-30 | 08:30:00 | 12:00:00 |
| 1004 | 2 | 101 | 2025-08-30 | 14:00:00 | 15:00:00 |

## Partie 4 – Requêtes SQL

7. Afficher **la liste des utilisateurs** avec le nom de l'ordinateur sur lequel ils se sont connectés.
8. Afficher les **connexions du 30 août 2025**, avec les noms des utilisateurs et ordinateurs.
9. Afficher la **durée (en heures)** de chaque connexion (indice : utiliser TIMESTAMPDIFF ou équivalent).
10. Afficher les **noms des ordinateurs** qui ont été utilisés par **plus d'un utilisateur**.
11. Afficher le **temps total passé** par chaque utilisateur **sur tous les ordinateurs**.
12. Lister les utilisateurs qui **ne se sont jamais connectés**.

## Partie 5 – Requêtes bonus (facultatives)

13. Lister les ordinateurs **non utilisés depuis plus de 30 jours**.

14. Lister les utilisateurs ayant utilisé **au moins 2 ordinateurs différents**.
15. Proposer une vue `vue_utilisation` qui affiche pour chaque connexion :
    - nom utilisateur,
    - nom ordinateur,
    - durée de la connexion (en minutes)

---

Consignes

- Indenter et commenter votre code SQL.
- Sauvegarder vos requêtes dans un fichier `.sql` ou `.txt`.
- Testez vos requêtes avec les données fournies.

---

# Correctionn – TP Parc Informatique

Partie 1 – MCD et compréhension du modèle

1. **Les trois entités principales du MCD :**
    - **UTILISATEUR**
    - **ORDINATEUR**
    - **CONNEXION**
2. **Attributs principaux de chaque entité :**
    - **UTILISATEUR** :
        - `id_utilisateur` (clé primaire)
        - `nom`
        - `prenom`
        - `service`
        - `email`
    - **ORDINATEUR** :
        - `id_ordinateur` (clé primaire)
        - `nom_ordinateur`
        - `adresse_ip`
        - `systeme_exploitation`
        - `date_installation`
    - **CONNEXION** :
        - `id_connexion` (clé primaire)
        - `id_utilisateur` (clé étrangère)
        - `id_ordinateur` (clé étrangère)
        - `date_connexion`
        - `heure_debut`

- heure_fin

3. **Cardinalités entre `UTILISATEUR` et `CONNEXION` :**
   - Un **utilisateur** peut avoir plusieurs **connexions** (cardinalité 1,N).
   - Une **connexion** est associée à un seul **utilisateur** (cardinalité N,1).

   **Cardinalités entre `ORDINATEUR` et `CONNEXION` :**
   - Un **ordinateur** peut être utilisé par plusieurs **utilisateurs** (cardinalité 1,N).
   - Une **connexion** concerne un seul **ordinateur** (cardinalité N,1).

4. **Rôle métier de l'entité `CONNEXION` :**
   - L'entité `CONNEXION` représente l'enregistrement d'une utilisation d'un ordinateur par un utilisateur à un moment donné. Elle contient des informations critiques telles que la date et l'heure de la connexion et de la déconnexion. Elle justifie donc une table propre plutôt que d'être une simple relation sans attributs, car elle contient des informations spécifiques sur chaque connexion.

---

## Partie 2 – MLD & Script de Création SQL

5. **Requêtes SQL pour la création des tables :**

```sql
CREATE TABLE UTILISATEUR (
    id_utilisateur INT PRIMARY KEY,
    nom VARCHAR(50),
    prenom VARCHAR(50),
    service VARCHAR(50),
    email VARCHAR(100)
);

CREATE TABLE ORDINATEUR (
    id_ordinateur INT PRIMARY KEY,
    nom_ordinateur VARCHAR(50),
    adresse_ip VARCHAR(15),
    systeme_exploitation VARCHAR(50),
    date_installation DATE
);

CREATE TABLE CONNEXION (
    id_connexion INT PRIMARY KEY,
    id_utilisateur INT,
    id_ordinateur INT,
    date_connexion DATE,
    heure_debut TIME,
    heure_fin TIME,
    FOREIGN KEY (id_utilisateur) REFERENCES UTILISATEUR(id_utilisateur),
    FOREIGN KEY (id_ordinateur) REFERENCES ORDINATEUR(id_ordinateur)
);
```

## Partie 3 – Insertion de Données Fictives

6. **Requêtes `INSERT INTO` pour insérer les données :**

```sql
-- Données UTILISATEUR
INSERT INTO UTILISATEUR VALUES
(1, 'Dupont', 'Alice', 'RH', 'alice.dupont@entreprise.com'),
(2, 'Martin', 'Jean', 'Informatique', 'jean.martin@entreprise.com'),
(3, 'Lopez', 'Maria', 'Compta', 'maria.lopez@entreprise.com');

-- Données ORDINATEUR
INSERT INTO ORDINATEUR VALUES
(101, 'PC-RH-01', '192.168.1.10', 'Windows 10 Pro', '2022-03-15'),
(102, 'PC-IT-01', '192.168.1.20', 'Ubuntu 22.04', '2023-01-10'),
(103, 'PC-COMPTA-01', '192.168.1.30', 'Windows 11', '2024-09-05');

-- Données CONNEXION
INSERT INTO CONNEXION VALUES
(1001, 1, 101, '2025-08-28', '08:45:00', '12:15:00'),
(1002, 2, 102, '2025-08-28', '09:00:00', '17:30:00'),
(1003, 3, 103, '2025-08-30', '08:30:00', '12:00:00'),
(1004, 2, 101, '2025-08-30', '14:00:00', '15:00:00');
```

## Partie 4 – Requêtes SQL

**7. Liste des utilisateurs avec le nom de l'ordinateur utilisé**

```sql
SELECT u.nom, u.prenom, o.nom_ordinateur
FROM CONNEXION c
JOIN UTILISATEUR u ON c.id_utilisateur = u.id_utilisateur
JOIN ORDINATEUR o ON c.id_ordinateur = o.id_ordinateur;
```

**8. Connexions du 30 août 2025**

```sql
SELECT u.nom, o.nom_ordinateur, c.heure_debut, c.heure_fin
FROM CONNEXION c
JOIN UTILISATEUR u ON c.id_utilisateur = u.id_utilisateur
JOIN ORDINATEUR o ON c.id_ordinateur = o.id_ordinateur
WHERE c.date_connexion = '2025-08-30';
```

**9. Durée de chaque connexion (en heures)**

```
SELECT c.id_connexion,
       u.nom,
       o.nom_ordinateur,
       TIMESTAMPDIFF(HOUR, c.heure_debut, c.heure_fin) AS duree_heures
FROM CONNEXION c
JOIN UTILISATEUR u ON c.id_utilisateur = u.id_utilisateur
JOIN ORDINATEUR o ON c.id_ordinateur = o.id_ordinateur;
```

#### 10. Ordinateurs utilisés par plus d'un utilisateur

```
SELECT o.nom_ordinateur
FROM CONNEXION c
JOIN ORDINATEUR o ON c.id_ordinateur = o.id_ordinateur
GROUP BY c.id_ordinateur
HAVING COUNT(DISTINCT c.id_utilisateur) > 1;
```

#### 11. Temps total passé par utilisateur

```
SELECT u.nom, u.prenom,
       SUM(TIMESTAMPDIFF(MINUTE, c.heure_debut, c.heure_fin)) AS
temps_total_minutes
FROM CONNEXION c
JOIN UTILISATEUR u ON c.id_utilisateur = u.id_utilisateur
GROUP BY u.id_utilisateur;
```

#### 12. Utilisateurs jamais connectés

```
SELECT u.nom, u.prenom
FROM UTILISATEUR u
LEFT JOIN CONNEXION c ON u.id_utilisateur = c.id_utilisateur
WHERE c.id_connexion IS NULL;
```

---

## Partie 5 – Requêtes bonus (facultatives)

#### 13. Lister les ordinateurs non utilisés depuis plus de 30 jours

```
SELECT o.nom_ordinateur
FROM ORDINATEUR o
LEFT JOIN CONNEXION c ON o.id_ordinateur = c.id_ordinateur
WHERE c.date_connexion < CURDATE() - INTERVAL 30 DAY OR c.date_connexion
IS NULL;
```

**14. Lister les utilisateurs ayant utilisé au moins 2 ordinateurs différents**

```sql
SELECT u.nom, u.prenom
FROM CONNEXION c
JOIN UTILISATEUR u ON c.id_utilisateur = u.id_utilisateur
GROUP BY u.id_utilisateur
HAVING COUNT(DISTINCT c.id_ordinateur) >= 2;
```

**15. Créer une vue `vue_utilisation` qui affiche pour chaque connexion :**

```sql
CREATE VIEW vue_utilisation AS
SELECT
  u.nom AS nom_utilisateur,
  o.nom_ordinateur,
  TIMESTAMPDIFF(MINUTE, c.heure_debut, c.heure_fin) AS duree_minutes
FROM CONNEXION c
JOIN UTILISATEUR u ON c.id_utilisateur = u.id_utilisateur
JOIN ORDINATEUR o ON c.id_ordinateur = o.id_ordinateur;
```

---

## Consignes

- **Indenter et commenter votre code SQL.**
- **Sauvegarder vos requêtes dans un fichier `.sql` ou `.txt`.**
- **Tester vos requêtes avec les données fournies.**