Sujet de TP n°1 – JDBC & MySQL : Gestion d'une compagnie aérienne

Objectifs

- Manipuler plusieurs tables en JDBC avec Java.
- Réaliser les opérations CRUD (Create, Read, Update, Delete).
- Apprendre à travailler avec des **relations entre tables** (clés étrangères, jointures).

Contexte

Vous devez développer une mini-application console en Java permettant de gérer une **base de données de vols** pour une compagnie aérienne.
Cette application utilisera **JDBC** pour se connecter à une base MySQL `vols`.

Schéma de la base de données

- **avion** : les avions de la compagnie.
- **pilote** : les pilotes de la compagnie.
- **vol** : les vols programmés, associés à un pilote et un avion.
- **passager** : les passagers inscrits.
- **reservation** : les réservations des passagers sur les vols (relation n-n).

Organisation du code (obligatoire) :

- Chaque entité de la base doit être représentée par une **classe Java modèle** (par ex. `Avion`, `Pilote`, `Passager`, `Vol`).
- Chaque table doit avoir une **classe DAO**(Data Access Object) dédiée pour les opérations CRUD en JDBC (par ex. `AvionDAO`, `PiloteDAO`, `PassagerDAO`, `VolDAO`).
- Les méthodes CRUD doivent être implémentées dans ces classes (ex. `create()`, `readAll()`, `update()`, `delete()`).
- Une classe principale `Main` servira à tester et exécuter les différentes opérations.

Consigne de structuration du code :
Vous devez séparer les responsabilités :

- 1 classe modèle par table (`Avion`, `Pilote`, `Vol`, `Passager`, `Reservation`).
- 1 classe DAO par table (`AvionDAO`, `PiloteDAO`, etc.) contenant les méthodes CRUD.
- 1 classe principale `Main` pour exécuter et tester le code.

Arborescence demandée :

```
tp-jdbc-vols/
│
├── src/
│   ├── model/
│   │   ├── Avion.java
│   │   ├── Pilote.java
│   │   ├── Passager.java
│   │   ├── Vol.java
│   │   └── Reservation.java
│   │
│   ├── dao/
│   │   ├── AvionDAO.java
│   │   ├── PiloteDAO.java
│   │   ├── PassagerDAO.java
│   │   ├── VolDAO.java
│   │   └── ReservationDAO.java
│   │
│   ├── util/
│   │   └── DatabaseConnection.java   // pour gérer la connexion JDBC
│   │
│   └── Main.java   // programme principal avec un menu console
│
├── lib/
│   └── mysql-connector-j-x.x.x.jar   // driver JDBC MySQL
│
├── sql/
│   └── volsdb.sql   // script de création et données fictives
│
└── README.md        // consignes ou rappel d'utilisation
```

Travail demandé

Étape 1 : CRUD de base

1. Implémentez en Java les opérations suivantes :
    - Ajouter / afficher / modifier / supprimer un **avion**.
    - Ajouter / afficher / modifier / supprimer un **pilote**.
    - Ajouter / afficher / modifier / supprimer un **passager**.

Étape 2 : Gestion des vols

1. Créer un vol en sélectionnant un **pilote** et un **avion** existants.
2. Lister tous les vols avec :
    - le numéro du vol,
    - la ville de départ et d'arrivée,

    - le nom du pilote,
    - le modèle de l'avion.
3. Modifier le statut d'un vol (`prévu`, `retardé`, `annulé`).
4. Supprimer un vol.

Étape 3 : Gestion des réservations

1. Créer une réservation pour un passager sur un vol donné.
2. Lister tous les passagers d'un vol donné.
3. Lister tous les vols réservés par un passager donné.
4. Supprimer une réservation.

Bonus (si vous avez fini)

- Ajouter une contrainte qui empêche de dépasser la capacité d'un avion lors des réservations.
- Implémenter un petit menu console permettant de naviguer entre les différentes opérations CRUD.