

# Rapport SI40

P25

## Plateforme pédagogique “Soodle”

**Créé par :** UZUN Enes-Alperen, VARMAZ

Evren et PATURAL Mathieu

**Date :** 5 mai 2025

### Introduction



Dans le cadre des unités d'enseignement **SI40 (Systèmes d'information)** et **WE4A/WE4B (Technologies et programmation WEB)**, nous avons été amenés à collaborer sur un projet transversal visant à développer une **plateforme pédagogique en ligne**, inspirée de solutions existantes telles que Moodle.

L'objectif principal de ce projet est de **concevoir et implémenter une application web fonctionnelle**, permettant la gestion de cours, d'utilisateurs, d'annonces, d'interactions pédagogiques, et de publications de contenus. Ce projet se divise en deux grandes phases correspondant aux deux UEs impliquées :

- **SI40**, qui se concentre sur la **conception, la modélisation et la gestion de la base de données**.
- **WE4A/WE4B**, qui se concentre sur la **création de l'interface utilisateur (front-end)** et de la **logique métier (back-end)**.

Nous parlerons dans ce rapport surtout de la première partie, où nous avons travaillé sur la **structuration de la base de données**, depuis sa modélisation selon l'approche Merise (MCD et MLD), jusqu'à sa mise en œuvre dans un système de gestion de bases de données relationnel (PostgreSQL ou MySQL). L'objectif est de garantir **l'intégrité, la cohérence et la performance** du système d'information de notre plateforme.

Cette base de données est conçue pour prendre en charge un ensemble de **fonctionnalités essentielles**, telles que l'authentification sécurisée, la gestion des rôles (étudiant, enseignant, administrateur), les inscriptions aux unités d'enseignement (UE) et la publication de contenus et d'annonces.

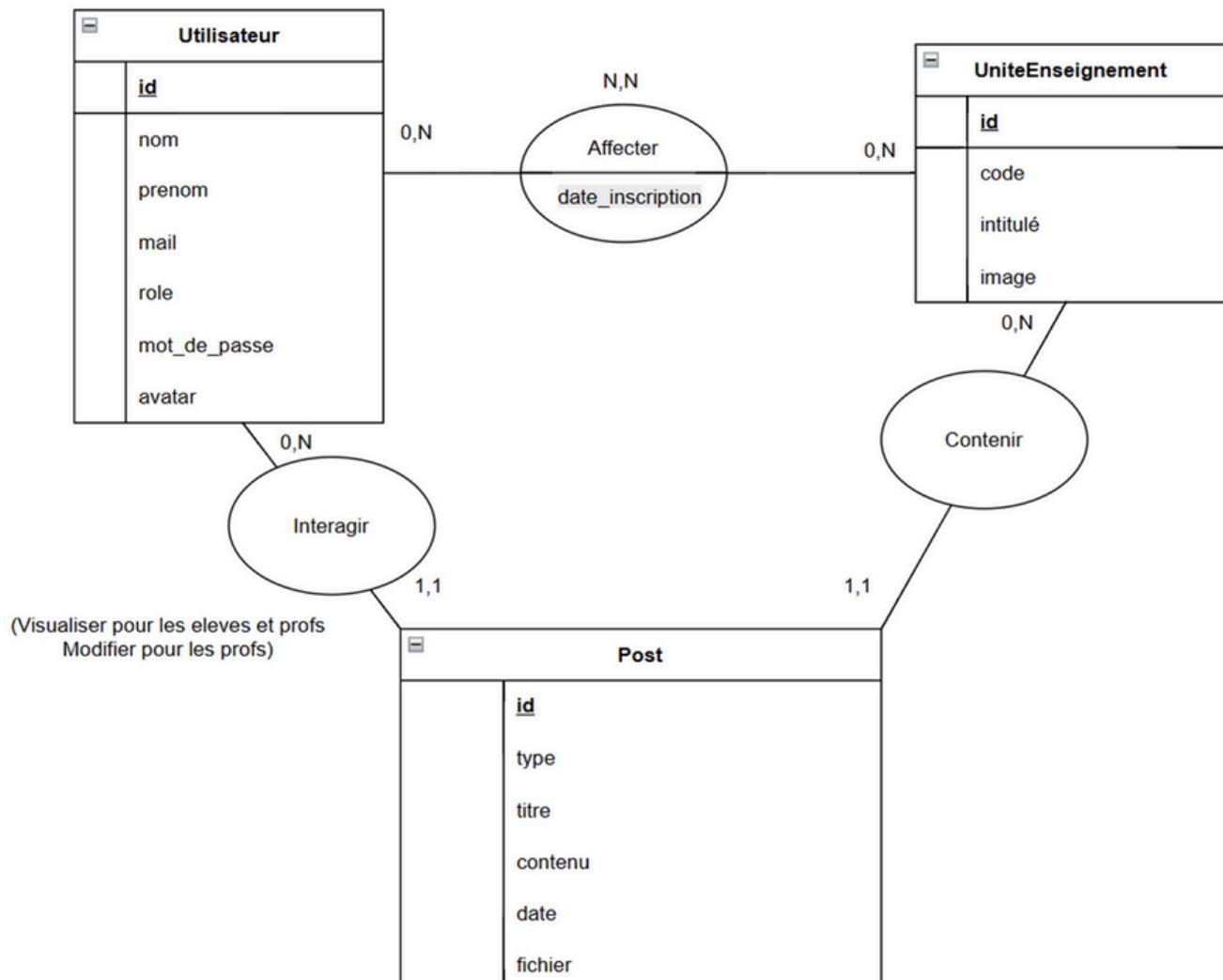
Ce rapport présente dans un premier temps notre démarche de modélisation à travers le MCD et le MLD, puis l'implémentation des tables, des relations et des requêtes SQL

avancées nécessaires au bon fonctionnement de la plateforme. Enfin, nous reviendrons sur les choix techniques effectués, les difficultés rencontrées et les solutions apportées.

# Conception de la base de données



## Modèle Conceptuel de Données (MCD)



Le **Modèle Conceptuel de Données (MCD)** permet de représenter de manière abstraite l'ensemble des entités, attributs et relations nécessaires au bon fonctionnement de notre plateforme pédagogique. Il a été réalisé selon la méthode Merise et tient compte des différentes interactions entre utilisateurs, contenus pédagogiques et unités d'enseignement.

Notre MCD repose sur trois entités principales :

- **Utilisateur** : représente les personnes accédant à la plateforme (étudiants, enseignants, administrateurs). Chaque utilisateur est identifié par un **id unique** et possède plusieurs attributs : **nom, prénom, mail, mot de passe, rôle** (indiquant le type d'utilisateur), et **avatar** (image de profil).
- **UniteEnseignement** : représente une unité d'enseignement disponible sur la plateforme. Elle est caractérisée par un **id, un code** (identifiant académique), un **intitulé** (nom de l'UE), et une **image** associée à l'UE.
- **Post** : représente tout contenu publié sur la plateforme (annonce, ressource pédagogique, message, etc.). Il est défini par un **id, un type** (déterminant si l'annonce est un message ou un message + un fichier joint), un **titre, un contenu, une date**, et éventuellement un **fichier joint**.

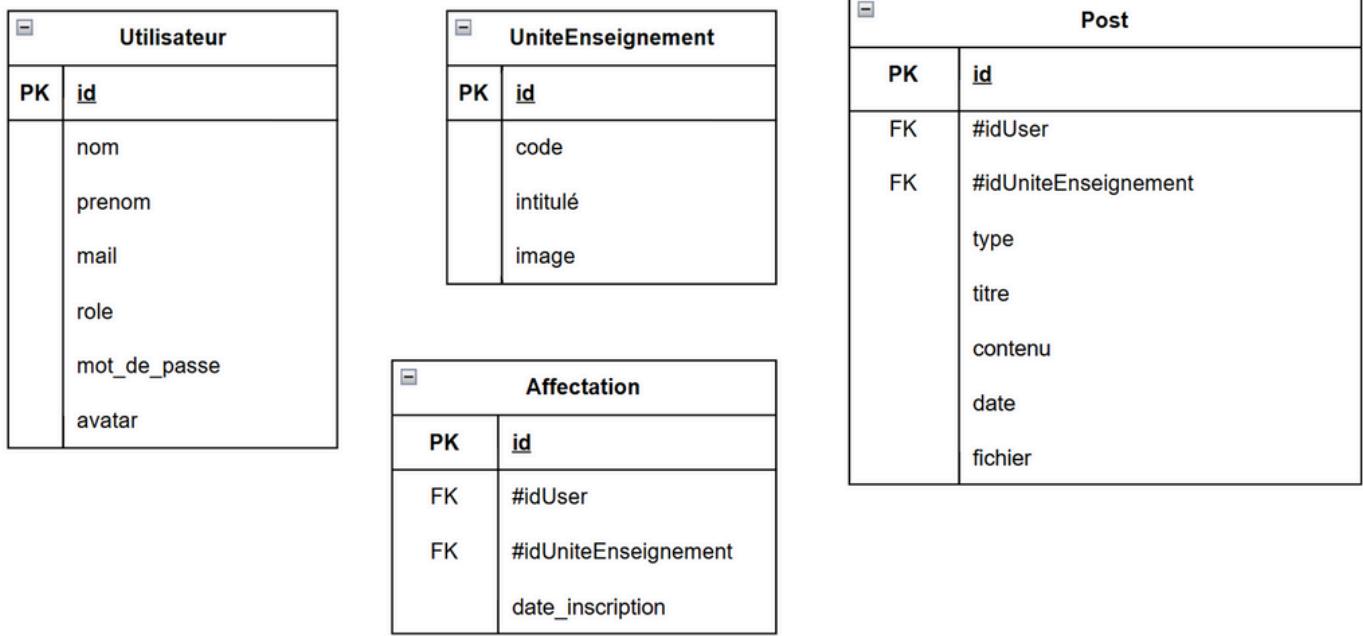
Les relations entre ces entités sont les suivantes :

- **Affecter** : relation **n-n** entre Utilisateur et UniteEnseignement, représentant l'inscription d'un utilisateur à une UE. Cette relation est porteuse de l'attribut **date\_inscription**, qui précise la date à laquelle l'inscription a eu lieu.
- **Contenir** : relation **1-n** entre UniteEnseignement et Post, signifiant qu'un post appartient à une seule UE, tandis qu'une UE peut contenir plusieurs posts.
- **Interagir** : relation **1-n** entre Utilisateur et Post, indiquant qu'un post est toujours publié (ou modifié) par un seul utilisateur, mais qu'un utilisateur peut interagir avec plusieurs posts. L'interaction dépend du rôle : les enseignants peuvent publier et modifier, tandis que les étudiants ont un accès en lecture seule.

Ce MCD pose ainsi les bases de notre modélisation logique et physique, en garantissant une structure relationnelle claire et normalisée, adaptée aux besoins fonctionnels de la plateforme.

## **Modèle Logique de Données (MLD)**

# MLD



Le Modèle Logique de Données (MLD) représente la traduction du MCD dans un formalisme relationnel adapté à la modélisation de la base de données relationnelle. Il formalise les tables, les attributs, les clés primaires (PK) et les clés étrangères (FK) qui assureront l'intégrité référentielle entre les différentes entités.

Notre MLD se compose des quatre tables suivantes :

- **Utilisateur** : cette table stocke les informations personnelles des utilisateurs de la plateforme. L'attribut **id** constitue la **clé primaire**. Les autres colonnes sont **nom**, **prénom**, **mail**, **mot de passe**, **rôle** (étudiant, enseignant, administrateur) et **avatar** (chemin vers l'image de profil).
- **UniteEnseignement** : elle contient les unités d'enseignement disponibles. Chaque enregistrement est identifié par un **id** (**clé primaire**) et possède les attributs **code**, **intitulé** et **image**.
- **Post** : cette table regroupe les contenus partagés sur la plateforme, tels que les annonces, ressources ou messages. Elle contient une **clé primaire id** et **deux clés étrangères** : **#idUser** (référence à l'auteur du post) et **#idUniteEnseignement** (UE à laquelle le post est associé). Les autres colonnes sont **type** (nature du post), **titre**, **contenu**, **date**, et **fichier** (fichier joint si applicable).
- **Affectation** : cette table fait le lien entre les utilisateurs et les unités d'enseignement, modélisant une relation plusieurs-à-plusieurs. Elle possède une **clé primaire id** et **deux clés étrangères** : **#idUser** et **#idUniteEnseignement**. Elle contient également un attribut **date\_inscription**, indiquant la date à laquelle l'utilisateur a été affecté à l'UE.

Ce MLD garantit l'intégrité des données tout en facilitant les jointures nécessaires aux différentes fonctionnalités prévues sur la plateforme (affichage des posts par UE, gestion des utilisateurs, affectations, etc.).

# Explication des requêtes SQL



Nous allons lister les requêtes qui nous ont été utiles lors de notre projet, et leurs utilités concrètes.

*Afficher tout les posts du plus récent au moins récent.*

“ SELECT \* FROM Post

ORDER BY date DESC;

”

Cette requête permet de **récupérer tous les posts** de la table Post, **classés du plus récent au plus ancien** selon la colonne date.

Elle est utile par exemple pour afficher les publications récentes en premier sur une page d'actualités ou un fil d'actualité.

*Afficher les posts d'UEs selon une liste d'UE du plus récent au plus moins.*

“ SELECT \* FROM Post

WHERE idUniteEnseignement IN (...)

ORDER BY date DESC;

”

Il faut remplacer (...) par la liste des identifiants d'unités d'enseignement (ex : 1, 2, 3).

Cette requête permet de **récupérer tous les posts** qui appartiennent à **certaines unités d'enseignement** (passées en paramètre) et de **les trier par date décroissante** (les plus récents en premier).

Utile, par exemple, pour afficher uniquement les publications d'UE auxquelles un utilisateur est inscrit.

## *Trouver toutes les affectations correspondantes à une UE*

```
“ SELECT * FROM Affectation a  
      INNER JOIN UniteEnseignement ue ON  
          a.idUniteEnseignement = ue.id  
      WHERE ue.id = :id; ”
```

Cette requête permet de **récupérer toutes les affectations** liées à **une unité d'enseignement précise** (celle dont l'identifiant est donné en paramètre).

Autrement dit, elle renvoie la liste des utilisateurs affectés (inscrits) à une certaine UE, qu'on à utilisé dans la page des participants à une UE pour notre application.

## *Trouver tout les cours d'un utilisateur*

```
“ SELECT ue.*  
      FROM affectation a  
      INNER JOIN unite_enseignement ue ON  
          a.unite_enseignement_id = ue.id  
      WHERE a.utilisateur_id = :userId; ”
```

Cette requête permet de **récupérer toutes les unités d'enseignement (UE)** auxquelles un **utilisateur spécifique est inscrit**, en se basant sur son identifiant userId.

Autrement dit : on récupère les UEs associées à un utilisateur donné via la table d'affectation.



# Difficultés rencontrées et solutions mises en place

---

Lors de la conception et de la mise en œuvre de la base de données, plusieurs difficultés ont été rencontrées.

L'une des principales concernait **la modélisation des relations complexes entre les entités**, notamment la **gestion des rôles utilisateurs et des inscriptions aux unités d'enseignement**. Pour résoudre cela, nous avons pris soin de bien normaliser notre MCD et d'**utiliser des associations n-aires** (comme pour l'affectation des utilisateurs aux UEs), puis de les transformer correctement dans le MLD avec des **clés étrangères** et des **contraintes d'intégrité**.

Une autre difficulté importante a été la gestion de la **cohérence des données lors de l'utilisation de requêtes SQL avancées**, en particulier celles impliquant des **jointures multiples**. Pour garantir de bonnes performances et une structure cohérente, nous avons **ajouté des index sur les colonnes clés** (comme les identifiants des UEs ou des utilisateurs) et vérifié la validité de nos requêtes via des jeux de données de test.

Enfin, lors du **développement avec Doctrine (ORM)**, nous avons dû **adapter notre logique aux méthodes de récupération de données orientées objet**. Certaines requêtes complexes ont donc été traduites en SQL natif lorsque cela facilitait la compréhension et l'efficacité, comme dans la récupération des UEs liées à un utilisateur.

---

## Conclusion

---



Ce projet a été une opportunité enrichissante pour mettre en pratique les concepts de modélisation de base de données vus en SI40, en lien direct avec un développement web complet réalisé dans le cadre de WE4A. À travers la création d'un MCD puis d'un MLD cohérent, nous avons pu structurer efficacement les données de notre plateforme pédagogique. L'implémentation du SGBD, la mise en place des contraintes d'intégrité, des jointures et des requêtes optimisées ont permis de garantir la fiabilité, la cohérence et la performance du système.

Par ailleurs, la collaboration entre la base de données et les fonctionnalités applicatives a mis en lumière l'importance d'une conception bien pensée dès les premières étapes. Ce travail interdisciplinaire nous a permis de développer des compétences techniques solides, tant en SQL qu'en intégration avec un framework back-end moderne.

Nous avons maintenant une base de données fonctionnelle, évolutive, et adaptée aux besoins d'une plateforme pédagogique, prête à accueillir de futures améliorations dans la seconde partie du projet avec WE4B.