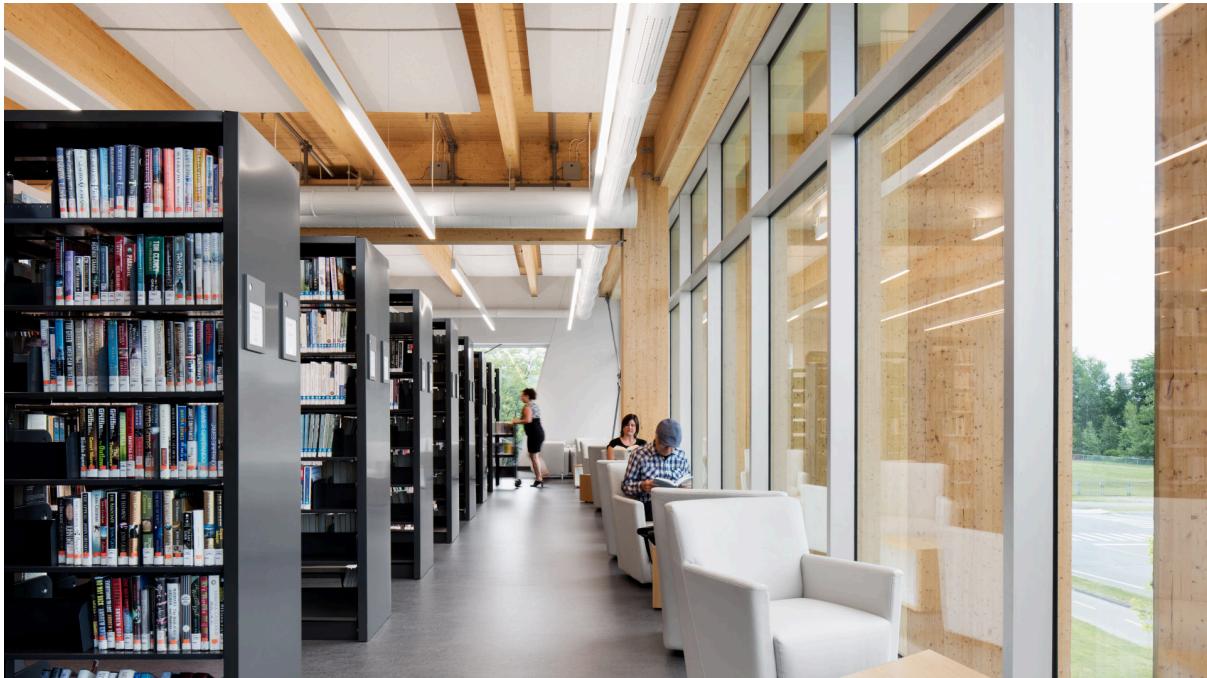


Semestre : A22

# NF18 - Projet Bibliothèque

## TD 1 - Groupe 3

### Rapport final



#### *Auteurs*

HASSAN Yousra, HONG Julie,  
PAUVAREL Alexandre, ROULLET Augustin

*Professeur référent : VICTORINO Alessandro*

## SOMMAIRE

<b>I. Contexte du projet</b>	<b>3</b>
<b>II. Présentation du projet</b>	<b>4</b>
<b>III. Fonctionnement de la base de données</b>	<b>6</b>
A. Ressources	6
1. Les 3 différents types ressources et leurs contributeurs	6
2. Ajouter et modifier des contributeurs, des ressources, les exemplaires	6
3. Consulter des ressources	6
B. Utilisateurs	6
1. L'implémentation des utilisateurs	6
2. Gestion des utilisateurs	7
3. Requêtes notables concernant la gestion des utilisateurs	7
C. Prêts	9
1. Emprunts d'exemplaire	9
2. Conditions de prêt	9
3. Gestion des sanctions	10
D. Statistiques	12

## I. Contexte du projet

Dans le cadre de l'UV NF18 à l'UTC, durant le semestre A22, nous sommes un groupe de 4 étudiants à avoir créé un système informatique de gestion pour une bibliothèque municipale. Ce projet a fait l'objet de 6 séances de travail en TD de 3h chacune, et d'heures supplémentaires en groupe et individuelles.

Ce projet a été découpé en plusieurs livrables par jalon tout au long du semestre :

- 12/10/2022 : NDC + MCD v1
- 19/10/2022 : MCD v2 + MLD v1
- 09/11/2022 : MLD v2 + SQL CREATE et INSERT
- 16/11/2022 : SQL SELECT et GROUP BY, application Python
- En décembre et janvier, nous y avons ajouté les finalisation de l'application Python et un fichier NoSQL R-JSON.

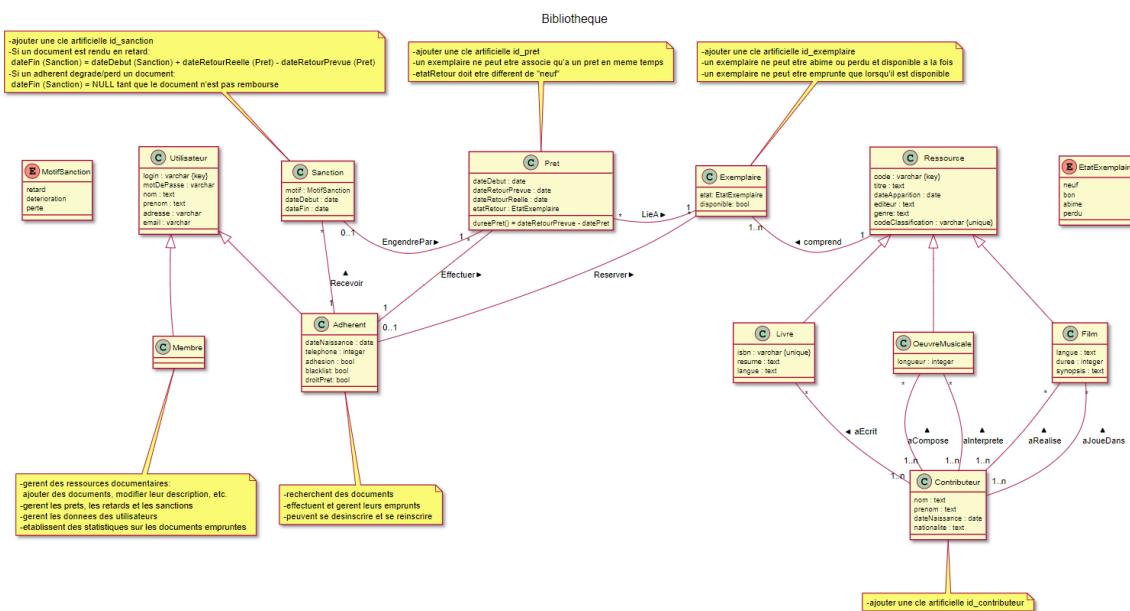
Ce compte rendu résume donc le travail réalisé pendant le semestre. Tout cela est accessible sur notre serveur Gitlab de l'UTC avec le lien :  
[https://gitlab.utc.fr/apauvare/nf18\\_p22\\_projet.git](https://gitlab.utc.fr/apauvare/nf18_p22_projet.git)

## II. Présentation du projet

Le projet Bibliothèque consiste à gérer la base de données d'une bibliothèque municipale. Pour cela, notre projet se présente sous la forme d'une interface entre les utilisateurs de la bases de données et les informations enregistrées (exemplaires des ressources, prêts, sanctions). Elle sert à :

- Classifier les ressources pour connaître le stock (catalogage, consultation)
- Gérer les adhésions et les comptes des utilisateurs
- Enregistrer des prêts et les suivre dans le temps
- Sanctionner des utilisateurs en cas de retard ou de dégradation des emprunts
- Pouvoir blacklister des utilisateurs

La structure de la base de données peut s'illustrer sous la forme de l'UML suivant :



Ce sont les membres qui gèrent les emprunts et les sanctions des adhérents.

Notre menu se présente ainsi :

```

Bienvenue au service de gestion de la bibliothèque NF18

Veuillez vous connecter en tant que membre ou adhérent

(0) Quitter
(1) Membre
(2) Adhérent

Choisissez 0, 1 ou 2: 1
login pour membre: fontmar
mot de passe pour fontmar: 1893FMart

***** Authentification Membre *****

Login: fontmar
Mot de passe: 1893FMart

```

```
***** Authentification réussie *****
```

```
***** Menu *****
```

```
Choisissez une action:
```

- (0) Déconnexion
- (1) Gestion des membres
- (2) Gestion des adhérents
- (3) Gestion des ressources
  
- (4) Gestion des exemplaires
- (5) Gestion des emprunts
- (6) Gestion des sanctions
- (7) Statistiques

```
Choisissez entre 0 et 7: 2
```

Les actions possibles sont :

- Ajouter des ressources
- Ajouter des utilisateurs
- Ajouter un prêt
- Enregistrer un retour de prêt
- Sanctionner en cas de retard

### III. Fonctionnement de la base de données

#### A. Ressources

##### 1. Les 3 différents types ressources et leurs contributeurs

Notre base de données accepte seulement 3 types de ressources : les livres, les œuvres musicales et les films. Chacune des ressources ont des caractéristiques propres ; elles sont liées à des types de contributeurs spécifiques. Les livres sont liés à des auteurs. Les œuvres musicales sont liées à un compositeur et un interprète. Les films sont liés à un réalisateur et à un acteur.

##### 2. Ajouter et modifier des contributeurs, des ressources, les exemplaires

Lors de l'acquisition de nouveaux ouvrages, les membres peuvent ajouter des ressources dans la base de données ainsi que les exemplaires correspondants.

Pour commencer, il faut ajouter les contributeurs selon le type de ressource (décrit dans la partie précédente).

Ensuite, il faut ajouter la ressource de la même manière dans la base de données. Enfin, il faut créer autant d'exemplaires qu'il y en a d'acquis dans la réalité. On peut vérifier l'ajout de ces exemplaires en affichant la liste des exemplaires.

Lors de ces enregistrements, si une erreur s'est glissée, il est possible de modifier les valeurs des attributs. Il y avait par exemple une erreur dans le titre "Star W ars : Un nouvel espoire" qu'il faut corriger en "Star Wars : Un nouvel espoir".

##### 3. Consulter des ressources

Il est intéressant pour les membres de la bibliothèque de pouvoir renseigner les adhérents sur le catalogue des ressources disponibles. Ici un exemple pour savoir quel exemplaire est disponible à l'emprunt.

#### B. Utilisateurs

##### 1. L'implémentation des utilisateurs

Dans notre base de données, le terme 'utilisateur' qualifie soit un membre soit un adhérent. Ainsi, au moment de la connexion à la base de données, l'utilisateur doit s'identifier selon sa catégorie et aura en conséquence un menu adapté. Notons que les adhérents ont un menu plus restreint et ont accès à moins de fonctionnalités.

Au niveau de l'implémentation, les tables Adherent et Membre partagent les mêmes attributs (issus d'une transformation par héritage à partir de l'UML) avec trois attributs supplémentaires pour Adherent qui caractérisent son adhésion et sa capacité à emprunter. Ces derniers sont :

- adhesion (configuré à true par défaut)
- blackList (configuré à false par défaut)
- et droitPret (configuré à true par défaut).

Ces attributs sont étroitement liés : si l'adhérent est blacklisté (blackList = true), il n'est plus adhérent (adherent = false). De plus, droitPret devient false en cas de sanctions à répétitions ou en cas d'erreurs graves (nous expliciterons la gestion des prêts dans la partie dédiée). Enfin, nous avons choisi comme convention que droitPret est réversible contrairement à blackList.

## 2. Gestion des utilisateurs

Au moment de la connexion à la base de données, on obtient deux menus différents :

- Le menu membre permet de gérer les membres, les adhérents, les ressources, les exemplaires, les prêts, les sanctions, et les statistiques. (gérer = ajouter, modifier et afficher)
- Le menu adhérent ne permet que d'afficher des ressources (en cas de recherche de documents), d'en emprunter si l'adhérent y a droit ou d'afficher ses coordonnées (pour pouvoir demander une modification à un membre du personnel en cas d'erreur).

## 3. Requêtes notables concernant la gestion des utilisateurs

- Ajouter un membre

```
#Ajouter un membre dans la base de données
def ajouterMembre(connexion):
    print("Veuillez saisir les informations ci-dessous:")
    login = input("Login: ")
    motDePasse = input("Mot de passe: ")
    nom = input("Nom: ")
    prenom = input("Prénom: ")
    adresse = input("Adresse: ")
    email = input("Email: ")
    cur = connexion.cursor()
    cur.execute("SELECT login FROM Membre WHERE login=%s", (login,))
    conditionCheck = cur.fetchall()
    if len(conditionCheck) != 0:
        print("Le membre existe déjà")
    else:
        cur.execute("INSERT INTO Membre VALUES (%s, %s, %s, %s, %s, %s)", (login, motDePasse, nom, prenom, adresse, email))
        connexion.commit()
    print("Voulez-vous ajouter un autre membre?")
    choixAjout = int(input("(0) Non\n(1) Oui\nChoisissez 0 ou 1: "))
    while choixAjout < 0 and choixAjout > 1:
        choixAjout = int(input("Saisie invalide, choisissez 0 ou 1: "))
    if choixAjout == 1:
        ajouterMembre(connexion)
```

- Modifier le droit d'emprunt d'un adhérent

```
def changerDroitEmprunt(connexion, adherent, nouveauDroit):
    cur = connexion.cursor()
    cur.execute("update Adherent set droitPret = %s where login=%s", (nouveauDroit, adherent))
    connexion.commit()
```

- Afficher les coordonnées d'un utilisateur

```
# Afficher les informations sur les adhérents
def afficherAdherents(connexion):
    cur = connexion.cursor()
    cur.execute("SELECT nom, prenom, adresse, email, dateNaissance, telephone, adhesion, blacklist, droitPret FROM Adherent")
    resultat = cur.fetchall()
    if len(resultat) > 0:
        for i in resultat:
            print("*****")
            print("Nom: ", i[0])
            print("Prénom: ", i[1])
            print("Adresse: ", i[2])
            print("Email: ", i[3])
            print("Date de naissance: ", i[4])
            print("Numéro de téléphone: ", str(i[5]))
            print("Est adhérent: ", i[6])
            print("Est blackisé: ", i[7])
            print("A le droit d'emprunt: ", i[8], "\n")
    else:
        print("Aucun adhérent enregistré")
```

## C. Prêts

### 1. Emprunts d'exemplaire

Pour pouvoir gérer les prêts, il faut préalablement s'assurer que des exemplaires des ressources soient disponibles et en mesure d'être prêtés. On s'intéressera aux conditions de prêts dans la section suivante.

C'est les membres du personnel qui sont chargés d'ajouter des exemplaires (option de leur menu). Au moment de l'ajout, l'exemplaire est considéré comme disponible et neuf, par conséquent ses attributs par défaut prennent la valeur 'disponible = true' et 'etat = neuf'.

Tous les utilisateurs peuvent afficher les exemplaires disponibles d'une ressource donnée.

*Afficher la disponibilité d'un exemplaire d'une ressource donnée :*

```
--Afficher la disponibilité d'un titre de ressource, si pas disponible afficher sa date de retour prévue
SELECT R.titre, E.disponible, P.dateRetourPrevue F
FROM (Exemplaire E INNER JOIN Ressource R ON E.ressource = R.code) LEFT OUTER JOIN Pret P
ON E.idExemplaire = P.exemplaire
WHERE R.titre = 'Transformer'
ORDER BY idExemplaire;
```

*Afficher les exemplaires disponibles pour une ressource donnée*

```
--Afficher les exemplaires empruntables de Transformer
SELECT idExemplaire, titre, codeClassification
FROM Exemplaire
WHERE ressource='Transformer'
AND disponible=TRUE
AND (etat = 'neuf' or etat = 'bon')
AND reserve IS NULL;
```

### 2. Conditions de prêt

Pour pouvoir emprunter un exemplaire, l'adhérent doit satisfaire un certain nombre de conditions :

*Qui dépendent de lui :*

- Il doit être adhérent ('adherent = true' et 'blackList = false')
- Il n'a le droit d'emprunter qu'un nombre limité d'exemplaires
- Il n'a pas le droit d'emprunter s'il est soumis à une sanction :
  - > S'il n'a pas rendu un livre après la date de retour, il ne peut plus emprunter pendant une durée égale au nombre de jours de retard
  - > En cas de perte ou de détérioration, il ne peut emprunter tant qu'il n'a pas remboursé le document.

*Qui dépendent des exemplaires qu'il souhaite emprunter :*

- L'exemplaire doit être disponible (existant, non réservé, non emprunté).
- L'exemplaire doit être neuf ou en bon état.

### 3. Gestion des sanctions

Les sanctions opèrent lorsqu'un adhérent ne respecte pas les règles concernant les prêts. Il se doit de rendre l'exemplaire :

- avant la date de retour,
- en bon état.

Si l'une de ses trois règles n'est pas respectée, des sanctions sont prises à son encontre :

- S'il ne rend pas l'exemplaire (en cas de perte par exemple) ou s'il le détériore, il n'a plus le droit d'emprunter jusqu'au remboursement de ce dernier ("droitPret = false").
- S'il rend l'exemplaire en retard, son droit de prêt est suspendu d'une durée du nombre de jours de retard.
- Si les sanctions sont répétées, un membre du personnel peut le blacklister, à savoir lui retirer (de manière réversible cependant) son statut d'adhérent.

La sanction prise à l'encontre d'un adhérent commence après la date de retour de l'exemplaire.

Enfin, avant de pouvoir emprunter une ressource, l'implémentation de la base de données couplée à la supervision d'un membre du personnel permet de vérifier les sanctions d'un adhérent donné afin de vérifier s'il est éligible au prêt. De la même manière, le retour d'une ressource se fait auprès d'un membre, ce qui permet notamment de juger l'état de la ressource rendue.

*Afficher le nombre de prêts actuel d'un adhérent :*

```
--Afficher le nombre de prêts actuels de l'utilisateur 'pauale'  
SELECT count (*)  
FROM Adherent INNER JOIN Pret  
ON Adherent.login = Pret.adherent  
WHERE dateRetourReelle > current_date AND login='pauale';
```

*Afficher les sanctions d'un adhérent :*

```

def afficherSanctions(connexion):
    cur = connexion.cursor()
    login = input("Veuillez saisir le login de l'adhérent: ")
    cur.execute("SELECT login FROM Adherent WHERE login = %s", (login,))
    result = cur.fetchall()
    if len(result) == 0:
        print("\nCet adhérent n'existe pas. ")
    else:
        cur.execute(
            "SELECT * FROM Sanction WHERE adherent = %s",
            (login,))
        result = cur.fetchall()
        for i in result:
            print("***** Id de la sanction: ", i[0], " *****")
            print("Motif: ", i[1])
            print("Début de sanction: ", i[2])
            print("Fin de sanction: ", i[3])
            print("Id de l'adhérent: ", i[4])
            print("Id du prêt: ", i[5], "\n")
  
```

### Retourner un prêt :

```

# Retourner un prêt
def retournerPret(connexion):
    cur = connexion.cursor()
    idPret = input("Id du prêt: ")
    cur.execute("SELECT * FROM Pret WHERE idPret = %s", (idPret,))
    resultat = cur.fetchall()
    if len(resultat) == 0:
        print("Prêt non trouvé")
        return
    exemplaire = resultat[0][5]
    adherent = resultat[0][6]
    etatretour = input("État de retour de l'exemplaire (neuf, bon, abîmé, perdu): ")
    while etatretour not in ['neuf', 'bon', 'abîme', 'perdu']:
        etatretour = input("État non reconnu; états possibles: neuf, bon, abîme, perdu\nVotre choix: ")
    if etatretour == 'abîme' or etatretour == 'perdu':
        print("L'utilisateur doit rembourser l'exemplaire pour pouvoir à nouveau emprunter.")
        cur.execute("UPDATE Exemplaire SET disponible = False where idExemplaire = %s", (exemplaire,))
        cur.execute("UPDATE Exemplaire SET etat = %s where idExemplaire = %s", (etatretour, exemplaire))
        cur.execute("UPDATE Adherent SET droitPret = False WHERE login = %s", (adherent,))
        idsanction = input("Id de la sanction: ")
        if etatretour == 'abîme':
            cur.execute("INSERT INTO Sanction VALUES (%s, 'deterioration', current_date, NULL, %s, %s)", (idsanction, adherent, re
        else:
            cur.execute("INSERT INTO Sanction VALUES (%s, 'perte', current_date, NULL, %s, %s)",
                        (idsanction, adherent, resultat[0][0]))
    connexion.commit()
else:
  
```

## D. Statistiques

Nous avons également proposé des fonctions qui permettent de récupérer des données afin d'afficher des statistiques. Ci-dessous un exemple de visualisation du nombre d'emprunts par livre (notamment pour mettre en évidence les livres les plus empruntés) :

Graphique illustrant le nombre d'emprunts par livre

