

# **Rapport**

## **Base De Données**

### **Bibliothèques**



**ELKRIEFF Benjamin et MUSENGA Christ**

**M1 INFORMATIQUE**

SOMMAIRE

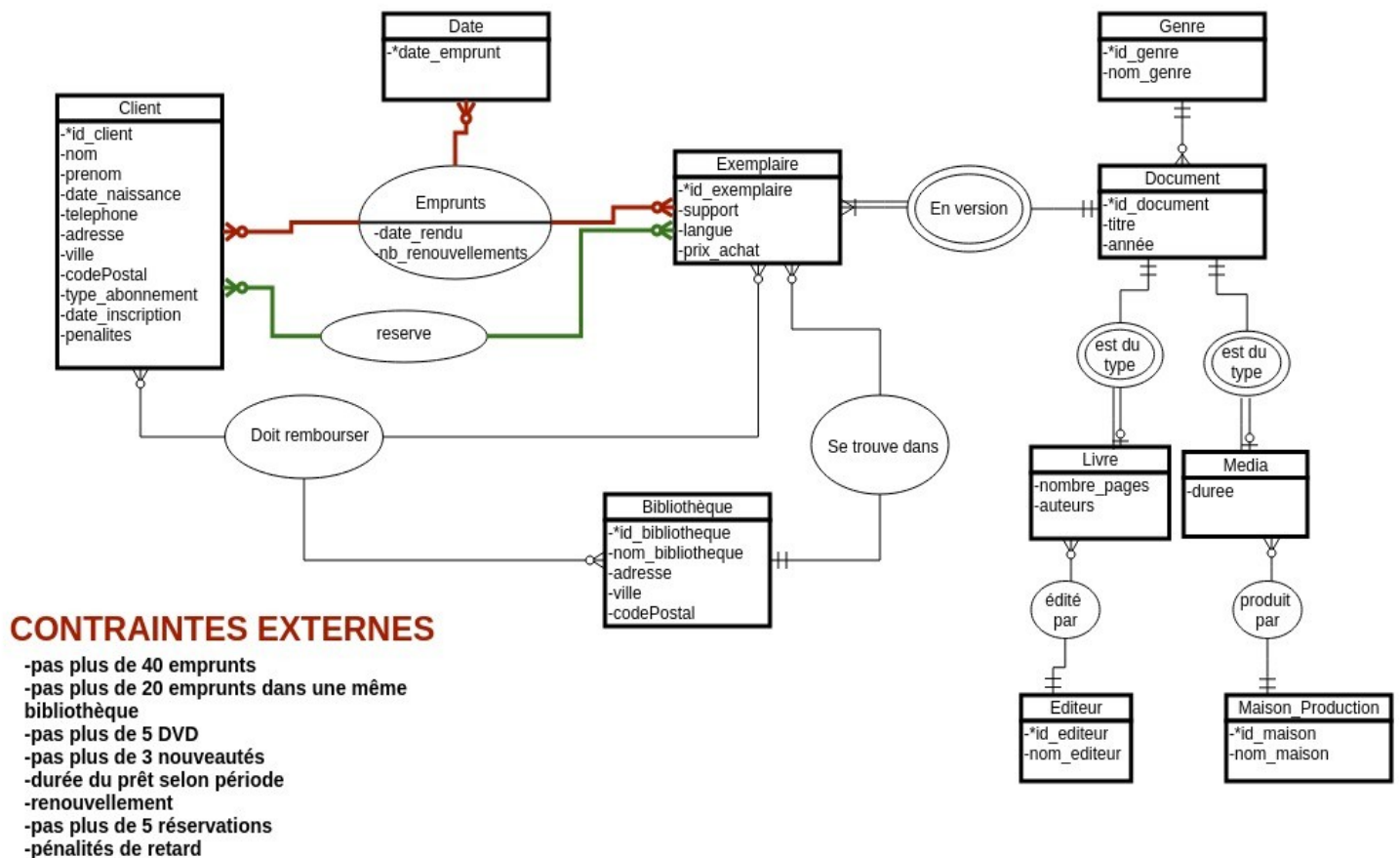
1. Simplifications choisies.....3

2. Listes des fonctions.....4

3. Liste des triggers.....8

4. Règles de gestion qui sont gérés par les triggers.....8

# 1. Simplifications choisies



Dans ce projet, plusieurs simplifications par rapport à notre modélisation a été faite. En effet nous n'avons pas forcément utilisé toutes les tables ou tous les attributs de chaque table pour simuler les actions des bibliothèques par l'utilisation des triggers.

Nous avons en majorité utilisé les tables :

- *client*
- *bibliothèque*
- *exemplaire*
- *emprunts*

- *reserve*
- *date\_courante*

De plus, pour les emprunts en été nous avons plutôt préféré faire en sorte que les emprunts en été soient faites entre le mois de Juin et le mois de Septembre plutôt qu'ils soient faite plus précisément entre le 21 Juin et le 20 Septembre.

Aussi, nous avons fait en sorte que les clients ne peuvent pas emprunter ou réserver pendant la période des 3 semaines précédent la fin de leur inscription car ils doivent d'abord renouveler leur abonnement (sinon certains pourraient profiter de faire exprès d'emprunter la veille de leur fin d'inscription).

Enfin, pour savoir si un exemplaire n'est pas encore rendu, nous avons préféré mettre la colonne *date\_rendu* à **NULL**. Si cet exemplaire est rendu alors *date\_rendu* vaudra la date où on veut rendre l'exemplaire.

## ***2. Listes des fonctions***

Pour ce projet, nous avons implémenté plusieurs fonctions afin que la majorité soit utilisé dans les triggers :

- ***nb\_emprunts\_biblio(idBiblio INTEGER)*** : prend en argument l'id d'une bibliothèque et retourne son nombre d'emprunts dans celle-ci.
- ***nb\_emprunts\_total()*** : retourne le nombre d'emprunt total dans toutes bibliothèques.
- ***abonnement\_ok(idClient INTEGER, idExemplaire INTEGER)*** : prend en argument l'id d'un client et l'id d'un exemplaire puis vérifie si ce client peut emprunter cet exemplaire

par rapport à son abonnement(normale,CD ou DVD).

- **is\_late(idClient INTEGER)** : prend en argument l'id d'un client et vérifie si ce client est en retard pour ses rendus d'emprunts.
- **nb\_jours\_en\_retard(idExemplaire INTEGER)** : prend en argument l'id d'un exemplaire et retourne le nombre de jours de retard de cet exemplaire(si il est en retard bien évidemment) depuis la date prévu de retour de cet exemplaire.
- **est\_a\_3semaines\_finAbo(idClient INTEGER)** : prend en argument l'id d'un client et vérifie si ce client est dans la période des 3 semaines précédent la fin de son inscription.
- **max\_limite\_emprunts\_40(idClient INTEGER)** : prend en argument l'id d'un client et vérifie si ce client peut encore emprunter par rapport au nombre d'emprunts max(40 en tout).
- **max\_limite\_emprunts\_biblio(idClient INTEGER, idBiblio INTEGER)** : prend en argument l'id d'un client et l'id d'une bibliothèque et vérifie si ce client peut encore emprunter dans cette même bibliothèque par rapport au nombre d'emprunts max(20 en tout).
- **max\_limite\_emprunts\_DVD(idClient INTEGER)** : prend en argument l'id d'un client et vérifie si ce client peut encore emprunter un nouveau DVD par rapport au nombre max d'emprunts de DVD(5 en tout).
- **max\_limite\_emprunts\_nouveaute(idClient INTEGER)** : prend en argument l'id d'un client et vérifie si ce client peut encore emprunter une nouveauté par rapport au nombre max d'emprunts

de nouveautés(3 en tout).

- **max\_limite\_emprunts\_ok(idClient INTEGER, idExemplaire INTEGER, idBibliotheque INTEGER)** : vérifie si un client donné respecte bien toutes les différentes limites d'emprunt(4 fonctions précédentes ci-dessus).
- **max\_limite\_reservation(idClient INTEGER)** : prend en argument l'id d'un client puis vérifie si ce client peut encore emprunter par rapport au nombre de réservations max(5 en tout)
- **actualisationPenalites(idClient INTEGER)** : prend en argument l'id d'un client et fait la mise à jour des pénalités de ce client.
- **actualisationPenalitesDefault()** : mise a jour des jour des pénalités des clients en retard, par défaut.
- **alerteRendreDans2jours(idClient INTEGER)** : prend en argument l'id d'un client et va avertir que ce client doit rendre le livre dans 2 jours.
- **penalites\_accumulees(idClient INTEGER, nb\_jours\_retard\_rajoute INTEGER)** : prend en argument l'id d'un client et un nombre de jours puis retourne les pénalités accumulées de ce client, en fonction de ce nombre de jours de retard rajouté.
- **fin\_abonnement(idClient INTEGER)** : prend en argument l'id d'un client et vérifie si ce client est en fin d'abonnement.
- **renouvellement\_inscription(idClient INTEGER)** : prend en argument l'id d'un client et renouvelle son abonnement si celui-ci n'est pas en retard sur ses exemplaires et qu'il est dans la

période des 3 semaines précédant la fin de son inscription (grâce à la fonction **est\_a\_3semaines\_finAbo()**).

- **avancer\_date(nb\_jours INTEGER)** : prend en argument un nombre de jours et fait avancer la date courante de **nb\_jours** jours.
- **renouvellement\_exemplaire(idExemplaire INTEGER)** : prend en argument un exemplaire et renouvelle son emprunt en updatant la date d'emprunt à la date courante si cet exemplaire n'est pas en retard et n'est pas réservé.
- **a\_fait\_reservation(idClient INTEGER, idExemplaire INTEGER)** : prend en argument l'id d'un client et l'id d'un exemplaire puis va vérifier si ce client donné a fait une réservation sur cet exemplaire.

### 3. Liste des triggers

Pour ce projet, nous avons utilisé différents triggers afin de gérer différentes actions dans la bibliothèque (emprunts, réservations etc..) :

- **emprunte()**
- **reservation()**
- **updateDateCourante()**
- **alerte\_rendu\_exemplaire()**

### 4. Règles de gestion qui sont gérés par les triggers

- **emprunte()** : Lors de l'emprunt d'un client(insertion dans la table **emprunts**), vérifie si ce client peut emprunter un exemplaire d'un document selon les différentes conditions (pas plus de 15€ de pénalité, exemplaire pas emprunté, pas réservé etc..). Elle vérifie d'abord si le client n'est pas bloqué, si le client peut emprunter l'exemplaire par rapport à son abonnement grâce à la fonction **abonnement\_ok()**, si il n'a pas de retard grâce à la fonction **is\_late()**, si l'exemplaire n'est pas déjà emprunté ou réservé et si ce client respecte bien toutes les limites d'emprunts grâce à la fonction **max\_limite\_emprunts\_ok()**.
- **reservation()** : Lors de la réservation d'un client(insertion dans la table **reserve**), elle vérifie d'abord si le client n'est pas bloqué, si le client peut emprunter l'exemplaire par rapport à son abonnement grâce à la fonction **abonnement\_ok()**, si le livre est déjà emprunté, si le client n'a pas de retard grâce à la fonction **is\_late()** et si le client respecte bien les limites du nombre de réservations grâce à la



fonction ***max\_limite\_reservation()***.

- **updateDateCourante()** : Lors de la mise à jour de la date courante(update dans la table ***date\_courante***), il y a une mise à jour de des pénalités des clients si ils sont en retard en le vérifiant grâce à la fonction ***is\_late()***. Les pénalités sont modifiées grâce à la fonction ***penalites\_accumulees()***.
- **alerte\_rendu\_exemplaire()** : Lors d'un rendu d'un exemplaire(update de la colonne ***date\_rendu*** de la table ***emprunts***), va lancer une alerte comme quoi un client vient de rendre un exemplaire et si un client avait fait une réservation sur l'exemplaire rendu, ce client l'empruntera directement(insertion dans la table ***emprunts***).