Rapport Projet Base de données avancées.
Bibliothèques

KERRIS Abderrahmane
TOUNSI Abdelkader

Introduction:

Le projet consiste à créer une base de données qui va être similaire à celle des ensembles des bibliothèques de la ville de Paris. Comme c'est mentionné dans l'énonce; on va essayer de gérer le plus d'événements et de contraintes possible: inscription d'un utilisateur, faire un prêt, faire une réservation d'un document etc. On va décrire brièvement dans ce rapport ce qu'on a fait et on va mentionner les simplifications prises en compte durant la réalisation.

Inscription d'un utilisateur:

Une inscription dans le réseau fait en sorte que l'utilisateur soit inscrit dans tout l'ensemble des bibliothèques du réseau. Ajouter un nouvel utilisateur est équivalent à ajouter une nouvelle ligne dans la table Utilisateur avec tous les informations nécessaires: Nom, prénom, date de naissance, adresse, type de l'inscription (Gratuite, cd, DVD). Le reste est généré automatiquement avec l'ajout de la ligne:

- Date de l'inscription = date courante (aujourd'hui).
- Date d'expiration = date courante + 1an. On a choisi de faire en sorte que l'abonnement dure 1 an.
- État de l'abonnement (Normal, Suspendu, Bloqué) = Normal (Évidemment).
- Pénalité (amende) = 0.

L'initialisation de ces attributs ce fait à travers un trigger qui se déclenche à l'ajout de la nouvelle ligne désignant le nouvel utilisateur.

Triggers associé à la création d'un utilisateur (selon l'ordre d'exécution):

1. Vérification de la cohérence de la date de naissance avec la date courante (Ne pas accepté une date de naissance ≥ date d'aujourd'hui).
2. Triggers qui fait l'ensemble des initialisations décrites ci-dessus.

La table Utilisateur a un autre trigger associé au faite du paiement de la pénalité (voir section Pénalité ci-dessous).

Effectuer un prêt depuis une bibliothèque:

Un utilisateur à jour peut emprunter un document depuis une bibliothèque. Le faite de faire un prêt consiste à ajouter une nouvelle ligne dans la table Emprunt, en donnant l'identifiant de l'utilisateur et l'identifiant de l'exemplaire du document souhaité. Chaque exemplaire est associé à une bibliothèque, donc inutile d'ajouter l'information Bibliothèque dans la table Emprunt.
La date du prêt est automatique la date courante (le prêt est effectué aujourd'hui), et la date de retour prévu est calculé selon la saison:

- Été → date retour prévu = date emprunt + 6 semaines.
- Hiver → date retour prévu = date emprunt + 3 semaines.

Bien évidement, avant de valider l'ajout du prêt, on a une liste des vérifications à faire, elles sont tous faite avec des triggers déclenché juste avant l'ajout de l'emprunt:

1. Vérifier l'état de l'abonnement de l'utilisateur: Un utilisateur suspendu ou bloqué n'a pas le droit d'emprunter des documents.
2. Vérifier qu'il n'y a pas de duplication dans la table Emprunt: Un utilisateur ne peut pas emprunter deux fois le même exemplaire (au même temps).
3. Vérifier que l'exemplaire demandé est bien disponible dans la bibliothèque et qu'il n'est pas déjà emprunté par un autre utilisateur.
4. Vérifier que le format du document (livre, cd, DVD) correspond au type d'abonnement de l'utilisateur (Gratuit, CD, CD/DVD).
5. Vérifier l'age de l'utilisateur: Un enfant ne peut pas emprunter un document pour adultes.
6. Vérifier la limite du format: S'assurer que l'utilisateur n'a pas emprunté plus que la limite autorisé pour le format du document (Exemple si l'utilisateur souhaite emprunter un DVD, il faut qu'il respecte le nombre de prêt autorisé pour les DVD).

7. Vérifier la limite dans la bibliothèques:
    a) Limite documents: Chaque bibliothèque a sa propre limite pour les document type 'livre'.
    b) Limite nouveauté: Chaque bibliothèque a sa propre limite pour les nouveaux exemplaires.

Après avoir assurer que tout est à jour et Ok, le calcule des dates (date emprunt et date retour prévu) seront maintenant calculer, avec un trigger qui s'exécute après tous les triggers listés ci-dessus.
La table Emprunt contient un attribut: nbRenouvellement, qui désigne le nombre de renouvellements effectués sur le prêt. L'attribut est initialisé à 0.

Renouvellement d'un prêt:

Renouvellement un prêt est fait en appelant la fonction *Renouveler(idUser, idExemplaire)*, la fonction va faire une mise à jours de la ligne de l'emprunt correspondante en incrémentant le nombre de renouvellement (+1). Ceci a comme effet le lancement un trigger qui lance une suite de vérifications avant de validé le renouvellement:

1. Vérifier que l'utilisateur n'a pas dépassé le nombre de renouvellement autorisé (2).
2. Vérifier que le prêt n'est pas en retard.
3. Vérifier que le document emprunté n'est pas réservé par un autre utilisateur (dans la bibliothèque).

Si tout est Ok, le trigger va finir par mettre à jours l'attribut nbRenouvellement, et la date de retour prévu pour l'emprunt (en ajoutant 3 semaines à la date de retour d'avant).

Retourner un prêt:

Après avoir emprunté un document, l'utilisateur va le retourner à la bibliothèque d'où il l'a emprunté. On retourne un document par l'ajout d'une ligne dans la table Retour; qui va nous servir comme historique sur les prêts.
En retournant un document, la ligne de la table Emprunt correspondante va être «transférée» vers la table Retour, avec quelques informations supplémentaires. C'est pourquoi après chaque retour, une ligne de la table Emprunt va être supprimée (la ligne qui correspond au prêt bien sur…) et une nouvelle ligne dans la table Retour va être insérée.
Avant de valider le retour, voici la liste des actions qu'on doit faire (des triggers qui s'exécutent avec l'ajout de la nouvelle ligne):

1. Générer la date de retour (qui est égale à la date courante) et récupérer la date d'emprunt: d'habitude; cet manipulation est faite après avoir fait toutes les vérifications, mais puisque l'ajout de la ligne signifie la suppression de la ligne depuis la table Emprunt, on doit d'abord récupérer la date d'emprunt pour la sauvegarder dans la nouvelles ligne. Cet attribut est très utile pour faire les statistiques après.
2. Supprimer la ligne de l'emprunt depuis la table Emprunt, après avoir testé l'identifiant de l'utilisateur et de l'exemplaire, pour s'assurer que le prêt a vraiment eu lieu.
3. Faire la mise à jour de l'utilisateur: Plusieurs scénarios:
    3.1. S'il avait du retard pour rendre ce document:
        a) Si la pénalité n'a pas dépassé 15€, sont état redevient Normal (Il était suspendu avant).
        b) Si la pénalité a dépassé 15€, sont état reste toujours bloqué (tant qu'il n'a pas payer l'amende).
    3.2) S'il n'avait pas du retard, on vérifie le reste de ses prêts et on met à jour son état selon le résultat de la vérification.
4. Voir si l'article est réservé par un autre utilisateur, si c'est le cas; un message de notification est affiché.

Si tout est OK, la nouvelle ligne dans la table retour est insérée, avec l'id de l'utilisateur, l'id de l'exemplaire, la date de l'emprunt et la date de retour.

Faire une Réservation:

Un utilisateur peut faire une réservation d'un document dans une bibliothèque, cette opération est effectuée avec une nouvelle insertion dans la table Réservation, avec l'identifiant de l'utilisateur, l'identifiant du document ainsi que celui de la bibliothèque souhaitée.

Pour valider une réservation, tous les exemplaires du document dans la bibliothèque doivent etres empruntés, autrement dit; on ne peut pas emprunter un document qui est disponible dans la bibliothèque.

L'opération se fait après avoir assuré que certaines conditions sont vérifié, tout comme le reste des tables, ces vérification sont faites avec des triggers qui se déclenchent à l'ajout d'une nouvelle ligne dans la table Réservation:

1. Vérifier que la date réservé est cohérente avec la date courante: On ne peut pas réservé un document pour une date dans le passée.
2. Vérifier que l'utilisateur n'a pas fait la réservation auparavant (Duplication).
3. Vérifier que le nombre de réservation n'a pas atteint la limite max (5 réservations par utilisateur).
4. Voir si l'utilisateur a un prêt en retard.
5. Voir si le document est disponible dans la bibliothèque (si c'est le cas, annulé la réservation).
6. Générer la date de réservation (le jour où on a fait la réservation, et non pas la date réservée) = date Courante.

Après tous ces opération, on valide la réservation si tout est OK.

Quelques mots sur Document, Bibliothèque et Exemplaire:

La table Document contient la liste de tous les documents disponibles dans le réseau des bibliothèques, la relation entre un document et une bibliothèque et représenté par Exemplaire: Un bibliothèque ait une liste d'exemplaires d'un document. C'est pourquoi donc, le prêt et le retour sont reliés directement aux exemplaires et non pas au documents, comme ça on évite de chercher à chaque fois un exemplaire du document dans une bibliothèque.

Les document sont en plusieurs formats: Livre, CD ou DVD. Néanmoins; on a fait une table Format qui contient les différents formats possibles, ceci est pour donner la main à l'utilisateur d'ajouter des formats au futur, et bien sur; chaque format a sa propre limite dans l'ensemble des bibliothèques. Même remarque pour Genre; chaque document a son propre genre (Math, Informatique, Bande dessinée, Policier …), et on les a stockés dans une table à part pouvoir en ajouté de nouveaux genres dans le futur tout comme format.

Pour les auteurs, on a choisi de faire une simplification et de mettre qu'un seul auteur par document. Les auteurs dans une table Auteur, et on l'a associé avec Document 1-1 (Voir schéma).

Listes des fonction créées:

- Vérifier les retards: *verifierRetard(idUtilisateur):* Comme son nom l'indique, la fonction prend en paramètre un utilisateur et vérifie si ce derniers a de prêts en retard ou pas. Elle retourne Vrai si c'est le cas, non Faux. Elle est très utilisée dans les triggers mentionnées plus haut dans ce rapport.
- Trouver des items dans le réseau: *rechercherLivre(motCle, idBibliothèque):* Celle-ci fait la recherche d'un document dans une des bibliothèques du réseau en donnant la bibliothèque souhaitée. Si ce paramètre vaut 0; elle s'exécute sur tout l'ensemble des bibliothèque. Le mot clé MotCle peut désigner soit le titre, soit une partie du titre, soit l'auteur, soit une partie du nom de l'auteur. La fonction ne retourne rien, elle affiche la liste des documents trouvée avec quelques informations supplémentaire: T*itre du document, Nom de la bibliothèque, nombre d'exemplaire qui existe ainsi que si le document est disponible/emprunté ou inexistant.*
- Supprimer les données trop vieux: *supprimer_les_donnes_trop_vieux():* afin d'éviter le stockage des informations inutiles, nous avons ajouté une nouvelle fonction, elle supprimer

touts les donnée qui sont trop vieux pour la base. Elle se base surtout sur l'historique des prêts (table Retour), tout les retours qui date de plus de deux mois seront supprimés de la table.

- Renouveler une inscription d'un utilisateur: *renouvler_inscription_utilisateur(Utilisateur)*: La durée d'abonnement d'un utilisateur est de 1 an. Sont compte sera bloqué dés que la date sera égale à la date d'expiration, et par conséquence; il ne pourra plus faire des actions sur l'ensemble des bibliothèques. Le renouvellement doit être fait au dernier mois de l'abonnement. À la fin de l'opération, l'état d'abonnement de l'utilisateur devient comme il était avant, et sa date d'expiration sera mise à jour; en ajoutant 1 an à cette dernière.
- Renouveler la durée d'un prêt: *renouveler(Utilisateur, exemplaire)*: La fonction est décrite dans partie Emprunt ci-dessus.

Fonctions sur les statistiques:

- Pourcentage des prêts dans le mois dernier: pourcentageEmprunt (*Bibliothèque*): Calculer les prets effectué durant le mois derniers (rendu ou pas), et calculer leur pourcentage par rapport aux exemplaires disponibles dans la bibliothèque.
- Pourcentage des prêts dans le mois dernier: pourcentageEmprunt (): Cette fonction fait le même traitement que la dernière; pour toutes les bibliothèques du réseau.
- Nombre de retard dans une bibliothèque: *nbRetard(bibliothèque):* Donne le nombre de prêts qui sont en retard (dans la bibliothèque données).

Testes:

Le rendu du projet contient des fichiers avec des scriptes de testes sur notre implémentation. On a essayé de faire des testes pour touts les scénarios possible, chaque scénario est décri en commentaire juste au dessus des commandes à exécuter.

NOTE: La date utilisée pour l'ensemble des testes est: '2016-04-25'. Il est important de la prendre en considération durant l'exécution de ces testes.

Les indexes possibles:

Utilisateur: On remarque qu'on accède à la table utilisateur presque dans chaque requête, l'attribut le plus recherché est évidement IdUser (l'identifiant). Il est très utile de mettre un index sur cet attribut, un index de type Table De Hashage, avec une fonction de hashage optimale à fin d'accéder directement à la ligne souhaité juste depuis l'identifiant de l'utilisateur (Exécution en O(1)).

Un autre index peut optimisé les accès à cette table: Un index type Arbre B+ sur l'attribut DateExpiration. La date d'expiration est souvent utilisée pour faire des comparaison avec la date Courante (<, >), dans ce style de teste; il est recommencé d'utilisé un index type Arbre B+ (Exécution en O(log n)).

Exemplaire: On accède souvent aussi à la table Exemplaire, pour ajouter des prêts;rendre un document ou faire une réservation. L'attribut utilisé est IdExemplaire (identifiant). Pareil que l'utilisateur, on ajoute un index type Table De Hashage avec une fonction de hashage adéquate. Le style de teste qu'on fait sur cet attribut sont des testes d'égalité directe. Un index Table de Hashage est la meilleur solution pour optimiser les accès.

Document: Même remarque que Exemplaire, on ajoute un index Table de Hashage sur l'identifiant.

Emprunt/Retour: Même chose pour les identifiants (idEmprunt et idRetour)-> Un index table de hashage.

Pour la table emprunt, on ajoute un index supplémentaire sur la date de retour prévu (DateRetourPrevu). Cet attribut est souvent utilisée dans des testes pour voir si on a dépasser la date courante par exemple (testes < >). C'est pourquoi il est recommander d'ajouter un index type Arbre B+ dans la table Emprunt sur DateRetourPRevu.

Simplification:

PART 1: Simplifications pour la première partie du projet: Modélisation

- Un utilisateur peut réserver un item dans plusieurs bibliothèque; pour prendre celui qui sera disponible au premier par exemple.
- L'historique d'un utilisateur est stocké dans les tables retour et emprunt:
  - Retour contiendra tout les items empruntés et retournés par l'utilisateur.
  - Emprunt contiendra que les items qui sont toujours en emprunt par l'utilisateur.
- L'éditeur est stocké comme attribut dans la table item. On a trouvé que ce n'est pas nécessaire d'attribuer une table spécialement pour Editeur. (Même remarque pour l'information concernant Adresse)
- 'Format' et 'Genre' d'un item sont stocké dans des tables spécifiques. Pour donner la possibilité d'ajouter de nouveaux Format/Genre, ainsi que d'éviter des problèmes d'incohérence (Par exemple Genre = 'BD' et 'Bande dessinée').

PART 2: Simplification durant la réalisation:

- 1 Auteur pour chaque document.
- Un exemplaire est considéré 'nouveau' si la date d'achat est < 1 mois de la date courante.
- Les limites sont initialisées au début, chaque bibliotheque/format a ses propres limites (qui peuvent être modifiés).

Schéma Relationnel:

Utilisateur(idUser, nom, prenom, dateNaissane, Email, adresse, typeInscription, dateInscription, dateExpiration, etatAbonnement, Penalite);

Auteur(idAuteur, nom, prenom);

Bibliotheque(idBibliotheque, nom, adresse, email, telephone, limitDoc, limitNouveaute);

Document(idDocument, titre, idAuteur*, format*, genre*, langue, isbn, annee, categorie, editeur, edition);

Exemplaire(idExemplaire, idDocument*, idBibliotheque*, prixAchat, dateAchat);

Emprunt(idEmprunt, idUser*, idExemplaire*, dateEmprunt, dateRetourPrevu, nbRenouvellement);

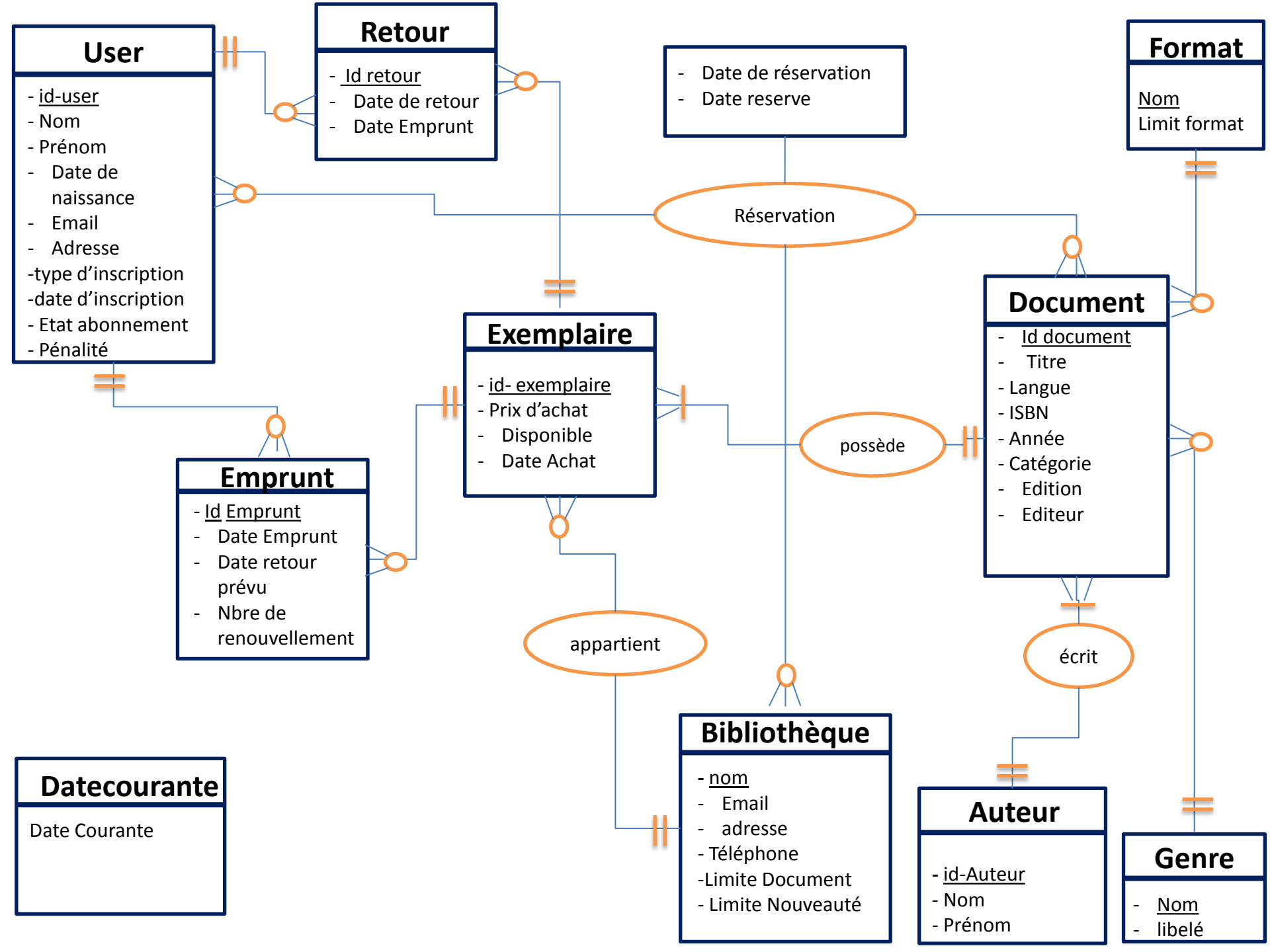
Retour(idRetour, idUser*, idExemplaire*, dateRetour, dateEmprunt);

Reservation(idReservation, idUser*, idDocument*, idBibliotheque*, dateReserve, dateReservation);

Genre(nom, libelle);

Format(nom, limitFormat);

DateCourante(dateCourante);


User
- id-user
- Nom
- Prénom
- Date de naissance
- Email
- Adresse
-type d'inscription
-date d'inscription
- Etat abonnement
- Pénalité
Retour
- Id retour
- Date de retour
- Date Emprunt
- Date de réservation
- Date reserve
Format
Nom
Limit format
Réservation
Exemplaire
- id- exemplaire
- Prix d'achat
- Disponible
- Date Achat
Document
- Id document
- Titre
- Langue
- ISBN
- Année
- Catégorie
- Edition
- Editeur
possède
Emprunt
- Id Emprunt
- Date Emprunt
- Date retour prévu
- Nbre de renouvellement
appartient
écrit
Bibliothèque
- nom
- Email
- adresse
- Téléphone
-Limite Document
- Limite Nouveauté
Auteur
- id-Auteur
- Nom
- Prénom
Genre
- Nom
- libelé
Datecourante
Date Courante