UNIVERSITÉ DE MONTRÉAL

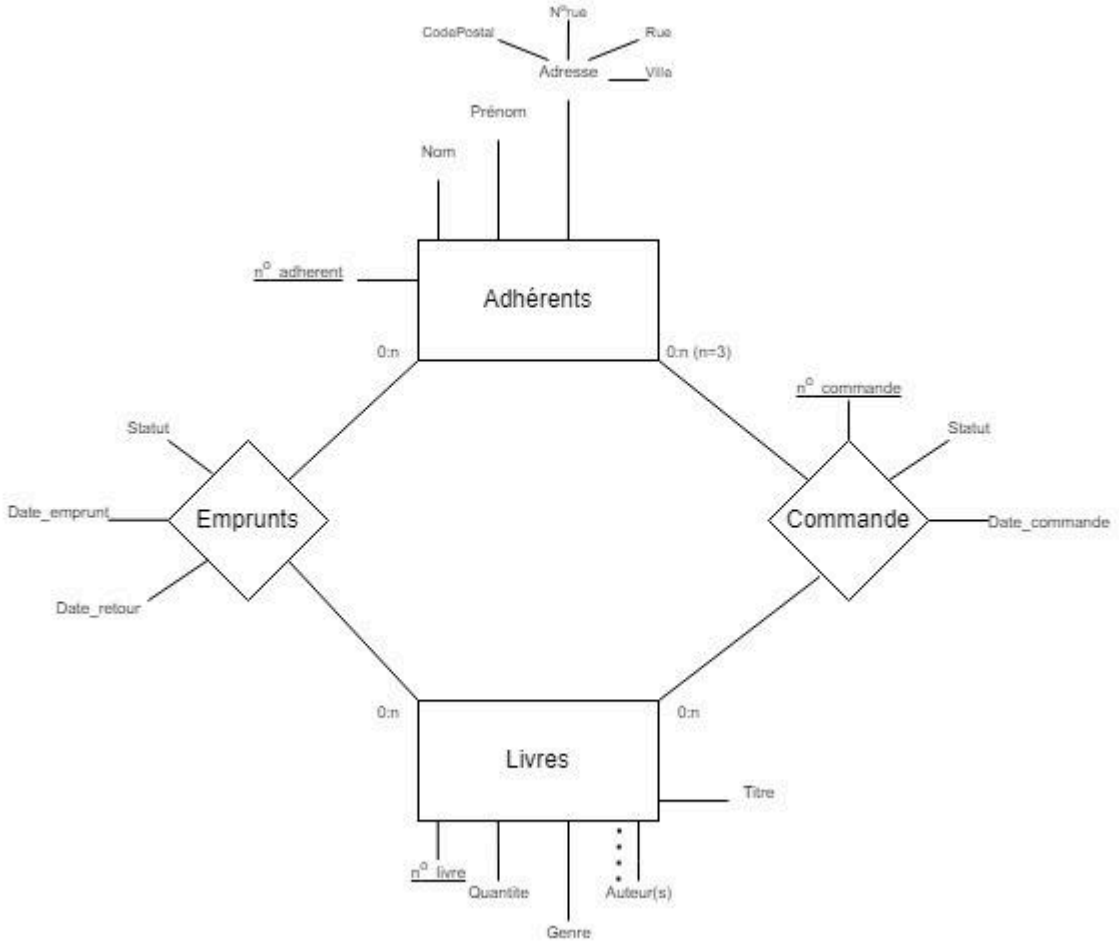
PROJET H24

PAR
Félix Carpentier - 20188286
Mahesh Roy Khulputeea - p1034741
Laurent-Philippe Roy-Lemaire - 20074007
Alex Rompré Beaulieu - 20230614

IFT2935 - BASES DE DONNÉES

AVRIL 2024

1 - Modélisation



Adhérents

Pour les adhérents, les attributs que nous avons choisis permettent d'identifier facilement l'utilisateur de la bibliothèque avec son nom et son prénom. On a comme clé primaire le numéro de l'adhérent qui est associé à son dossier pour la bibliothèque. On a aussi besoin de son adresse pour les commandes alors on a l'attribut adresse qui se décompose en code postal, numéro de rue, nom de rue et la ville pour avoir toutes les informations nécessaires aux commandes.

Emprunts

Comme on a besoin d'une trace de tous les emprunts effectués à la bibliothèque, on doit avoir des attributs liés à cette association. On a besoin des dates d'emprunt et date de retour afin de pouvoir savoir si un livre est déjà emprunté pour les commandes et pour pouvoir savoir le statut d'un emprunts (en retard ou non)

Livres

Pour les livres, notre clé primaire permet d'identifier facilement un livre en prenant son numéro. On doit aussi savoir la quantité d'exemplaires d'un livre avec la quantité et nous avons aussi les informations générales en lien avec des livres tels que les auteur(s), le genre et le titre.

Commande

On a aussi besoin d'une trace pour les commandes passées à la bibliothèque, on a donc comme clé primaire le numéro de la commande. On a aussi besoin de savoir la date de la commande et son statut qui sera déterminé selon la disponibilité des livres commandés (Commande annulée ou honorée).

2 - Transformation

Les entités deviennent des relations avec leurs attributs. Chaque composante d'un attribut composite devient un attribut. On obtient donc

Adhérents (noAdherent, nom, prenom, CodePostal, noRue, Rue, Ville)

Pour l'entité Livres on a aussi un attribut multivalué. On doit donc créer une relation propre à lui avec en clé étrangère la clé primaire de l'entité qu'il vient et sa clé primaire est la combinaison des attributs. (diapo 27, PDF-Le modèle relationnel). On a donc

Livres(noLivre, quantite, genre, titre)

Auteurs(#noLivre,auteur)

On ne doit pas ajouter de clé étrangère à ces relations, car les liens ne sont pas 0:1 ou 1:1. Par contre, comme on a des liens 0:n ou 1:n de chaque côté de nos associations, on doit alors les ajouter en tant que relations. L'association Commande a déjà une clé primaire, les clés étrangères ajoutées ne seront pas la clé primaire. Par contre pour Emprunts la clé primaire sera une clé composée des clés étrangères ajoutées. On obtient donc ceci:

Emprunts(#noAdherent, #noLivre, statut, dateEmprunt, dateRetour)

Commande(noCommande, statut, dateCommande, #noAdherent, #noLivre)

On obtient donc ces relations:

Adhérents (noAdherent, nom, prenom, CodePostal, noRue, Rue, Ville)

Livres(noLivre, quantite, genre, titre)

Auteurs(#noLivre,auteur)

Emprunts(#noAdherent, #noLivre, statut, dateEmprunt, dateRetour)

Commande(noCommande, statut, dateCommande, #noAdherent, #noLivre)

3 - Normalisation

Adhérents
F = {noAdherent→nom,prenom,CodePostal,noRue,Rue,Ville}
Ici tous nos attributs dépendent du noAdherent et directement de celui-ci.

- 1NF

Ici tous nos attributs sont atomiques donc on est bien en 1NF

- 2NF

Tous les attributs qui n'appartiennent pas à la clé dépendent entièrement de la clé, car c'est une clé simple. On est donc 2NF

- 3NF

Elle est 3NF, car tous les attributs dépendent directement de la clé.

- BCNF

La relation est aussi en BCNF, car tous les attributs de gauche dans nos dépendances sont clés.

Livres
F = {noLivre→quantite,genre,titre}
Ici tous nos attributs dépendent du noLivre qui est la clé primaire.

- 1NF

Ici tous nos attributs sont atomiques donc on est bien en 1NF

- 2NF

Tous les attributs qui n'appartiennent pas à la clé dépendent entièrement de la clé, car c'est une clé simple. On est donc 2NF

- 3NF

Elle est 3NF, car tous les attributs dépendent directement de la clé.

- BCNF

La relation est aussi en BCNF, car tous les attributs de gauche dans nos dépendances sont clés.

Auteurs

Les dépendances fonctionnelles sont triviales ici, car tous nos attributs sont dans la clé.
On sait aussi de façon triviale que cette relation est BCNF.

Emprunts

F:{#noAdherent,#noLivre→dateEmprunt,dateRetour,statut}
Ici dateEmprunt, dateRetour et statut dépendent de la clé primaire entière, car chaque livre emprunté par une personne peut être retourné à différentes dates et donc avoir des statuts différents.

- 1NF

Ici tous nos attributs sont atomiques donc on est bien en 1NF

- 2NF

Tous les attributs qui n'appartiennent pas à la clé dépendent entièrement de la clé, car c'est une clé simple. On est donc 2NF

- 3NF

Elle est 3NF, car tous les attributs dépendent directement de la clé.

- BCNF

La relation est aussi en BCNF, car tous les attributs de gauche dans nos dépendances sont clés.

Commande

F = {noCommande→statut,dateCommande,#noAdherent,#noLivre}
Ici tous nos attributs dépendent directement de la clé primaire noCommande qui permet d'identifier une commande.

- 1NF

On ne respecte pas 1NF, car une commande peut contenir plusieurs livres (jusqu'à 3) et donc avoir plusieurs attributs noLivre différents. Pour corriger ce problème, on va décomposer la relation en deux relations (comme démontré dans la diapo 4 du PDF - Les formes normales) :
Commande(<u>noCommande</u>, statut, dateCommande, #noAdherent)
LivresCommande(<u>noCommande, #noLivre</u>)

Ces deux relations respectent maintenant 1NF

- 2NF

Pour la relation Commande on respecte bien le 2NF, car les attributs non clés dépendent entièrement de la clé noCommande qui est clé simple aussi.
LivresCommandes est aussi 2NF et on peut le voir de façon triviale.

- 3NF

Les deux relations sont 3NF, car tous les attributs dépendent directement de la clé de la relation.

- BCNF

Les relations sont aussi en BCNF, car tous les attributs de gauche dans nos dépendances sont clés des relations.

On obtient donc ceci comme relations :
Adherents (noAdherent, nom, prenom, CodePostal, noRue, Rue, Ville)
Livres(noLivre, quantite, genre, titre)
Auteurs(#noLivre,auteur)
Emprunts(#noAdherent, #noLivre, statut, dateEmprunt, dateRetour)
Commande(noCommande, statut, dateCommande, #noAdherent)
LivresCommande(noCommande, #noLivre)

4 - Implémentation

Voir fichier sql

5 - Questions/réponses

- Question 1

Question : Nom et prénom des adhérents qui ont emprunté plus d'un livre depuis 2023 et qui n'ont aucun retard

Algèbre relationnel :

$r1 \leftarrow \sigma_{"DATEEMPRUNT >= 2023-01-01" \ AND \ "STATUT != En \ retard"}(EMPRUNTS)$

$r2 \leftarrow NOAHDHERENT\lambda_{COUNT(NOLIVRE)}(r1)$

$r3 \leftarrow \rho_{"NOADHERENT, NBEMPRUNTS"}(r2)$

$r4 \leftarrow \sigma_{"NBEMPRUNTS > 1"}(r3)$

$r5 \leftarrow \pi_{"NOADHERENT"}(r4)$

$r6 \leftarrow r5 \bowtie (ADHERENTS)$

$r7 \leftarrow \pi_{"NOM, PRENOM"}(r6)$

SQL :

WITH r1 AS (SELECT * FROM EMPRUNTS WHERE DATEEMPRUNT >= '2023-01-01' AND STATUT != 'En retard'),

r2 AS (SELECT NOADHERENT,COUNT(NOLIVRE) AS NBLIVRES FROM r1 GROUP BY NOADHERENT HAVING COUNT(NOLIVRE) > 1)

SELECT NOM,PRENOM FROM r2 NATURAL JOIN ADHERENTS

Optimisation :

Pour l'optimisation, l'important ici est de faire la jointure naturelle avec le moins d'éléments et donc d'enlever le plus d'éléments possibles avant les opérations coûteuses. C'est pour ça qu'on commence par une sélection avant de faire l'agrégation et on s'assure de garder seulement NOADHERENT qui satisfait toutes les conditions pour la jointure. De cette façon, notre requête est plus optimale.

- Question 2

Question : Tous les livres empruntés depuis 2020 par des adhérents dont le prénom commence par la lettre `M`.

Algèbre relationnel :

$r1 \leftarrow \sigma_{"DATEEMPRUNT >= 2019-12-31"}(EMPRUNTS)$

$r2 \leftarrow \pi_{"NOADHERENT.NOLIVRE, DATEEMPRUNT"}(r1)$

$r3 \leftarrow \sigma_{"PRENOM\ LIKE\ 'M\%'"}(ADHERENTS)$

$r4 \leftarrow \pi_{"NOADHERENT.NOM, PRENOM"}(r3)$

$r5 \leftarrow r2 \bowtie_{"r2.NOLIVRE = livres.NOLIVRE"} (LIVRES)$

$r6 \leftarrow r5 \bowtie_{"r2.NOADHERENT = r4.NOADHERENT"} (r4)$

$r7 \leftarrow \pi_{"L.TITRE, A.NOM, A.PRENOM, E.DATEEMPRUNT"}(r6)$

SQL :

```
WITH r2 AS (
  SELECT NOADHERENT, NOLIVRE, DATEEMPRUNT
  FROM biblio.emprunts
  WHERE DATEEMPRUNT > '2019-12-31'
```

```
),
    r4 AS (
        SELECT NOADHERENT, NOM, PRENOM
        FROM biblio.adherents
        WHERE PRENOM LIKE 'M%'
    )
SELECT L.TITRE, A.NOM, A.PRENOM, E.DATEEMPRUNT
FROM r1 E
        JOIN livres L ON E.NOLIVRE = L.NOLIVRE
        JOIN r2 A ON E.NOADHERENT = A.NOADHERENT
```

Optimisation:

Dans le même esprit que la requête précédente, nous utilisons la syntaxe WITH pour diminuer la taille de nos tables avant de procéder avec l'opération coûteuse qu'est le JOIN.

- Question 3

Question :Le nombre total de livres emprunté par chaque adhérent depuis son premier emprunt.

Algèbre relationnel :

$$r1 \leftarrow \pi_{"NOADHERENT, NOM, PRENOM"}(ADHERENT)$$

$$r2 \leftarrow NOADHERENT\lambda_{COUNT(NOLIVRE)}(EMPRUNT)$$

$$r3 \leftarrow \rho_{"NOADHERENT,NBEMPRUNT"}(r2)$$

$$r4 \leftarrow \sigma_{"NBEMPRUNT > 0"}(r3)$$

$$r5 \leftarrow r1 \bowtie_{"a.NOADHERENT = e.NOADHERENT"}(r3)$$

SQL :

```
SELECT a.NOADHERENT, a.NOM, a.PRENOM, COUNT(EMPRUNTS.NOLIVRE) AS r1
        FROM biblio.adherents a
            JOIN biblio.emprunts ON a.NOADHERENT = EMPRUNTS.NOADHERENT
        GROUP BY a.NOADHERENT
        HAVING COUNT(EMPRUNTS.NOLIVRE) > 0;
```

Optimisation :

Nous avons mis la jointure à la fin pour que l'opération soit la moins couteuse possible.

- Question 4

Question : L'adhérent et le titre du livre de chaque commande marquée comme Honorée.

Algèbre relationnel :

$r1 \leftarrow ADHERENTS \bowtie_{"a.noadherent = c.noadherent"} (COMMANDES)$

$r2 \leftarrow r1 \bowtie_{"c.nocommande = lc.nocommande"} (LIVRESCOMMANDE)$

$r3 \leftarrow r2 \bowtie_{"lc.nolivre = l.nolivre"} (LIVRES)$

$r4 \leftarrow \sigma_{"COMMANDE.STATUT = \ 'Honoree'"}(r3)$

$r5 \leftarrow \pi_{"NOM, PRENOM, TITRE, STATUT"}(r6)$

SQL :

```
SELECT
   a.NOM, a.PRENOM, l.TITRE, c.STATUT
FROM biblio.adherents a
      JOIN biblio.commandes c ON a.NOADHERENT = c.NOADHERENT
      JOIN biblio.livrescommande lc ON c.NOCOMMANDE = lc.NOCOMMANDE
      JOIN biblio.livres l ON lc.NOLIVRE = l.NOLIVRE
WHERE c.STATUT = 'Honoree'
```

Optimisation:

La nature de cette requête demande de créer un amalgame des données qui appartiennent à toutes les tables de la base de données. De fait, nous ne pouvons pas vraiment faire mieux sur le plan de l'optimisation.

ANNEXE: GUIDE POUR L'APPLICATION

—

db.url=jdbc:postgresql://localhost:5432/postgres

db.username=postgres

db.password=root

—

1. Se rendre sur https://github.com/Mastalp/IFT2935_projet ou bien utiliser les fichiers de la remise pour se procurer le fichier zip du projet.
2. Décompresser le fichier zip contenant le code source inclus dans le répertoire du projet.
3. S'assurer d'avoir un serveur postgres qui roule sur le port 5432, user = postgres, mot de passe = root
4. Aller dans le répertoire `\IFT2935_projet-main` et exécuter la commande suivante :

```
java -jar .\ift2935_projet.jar
```