[INFO-H-303] Bases de données - Rapport Projet: "Ebay"

Alexandre Heneffe - 000440761
Nicolas Jonas - 000442112
Evgueniy Starygin - 000443325

Mars 2018

1 Introduction

Ceci est le rapport du projet de base de données. Celui-ci contiendra le schéma entité-relationnel (et ses contraintes) de la future base de données ainsi que sa traduction en un modèle relationnel.

1.1 Hypothèses

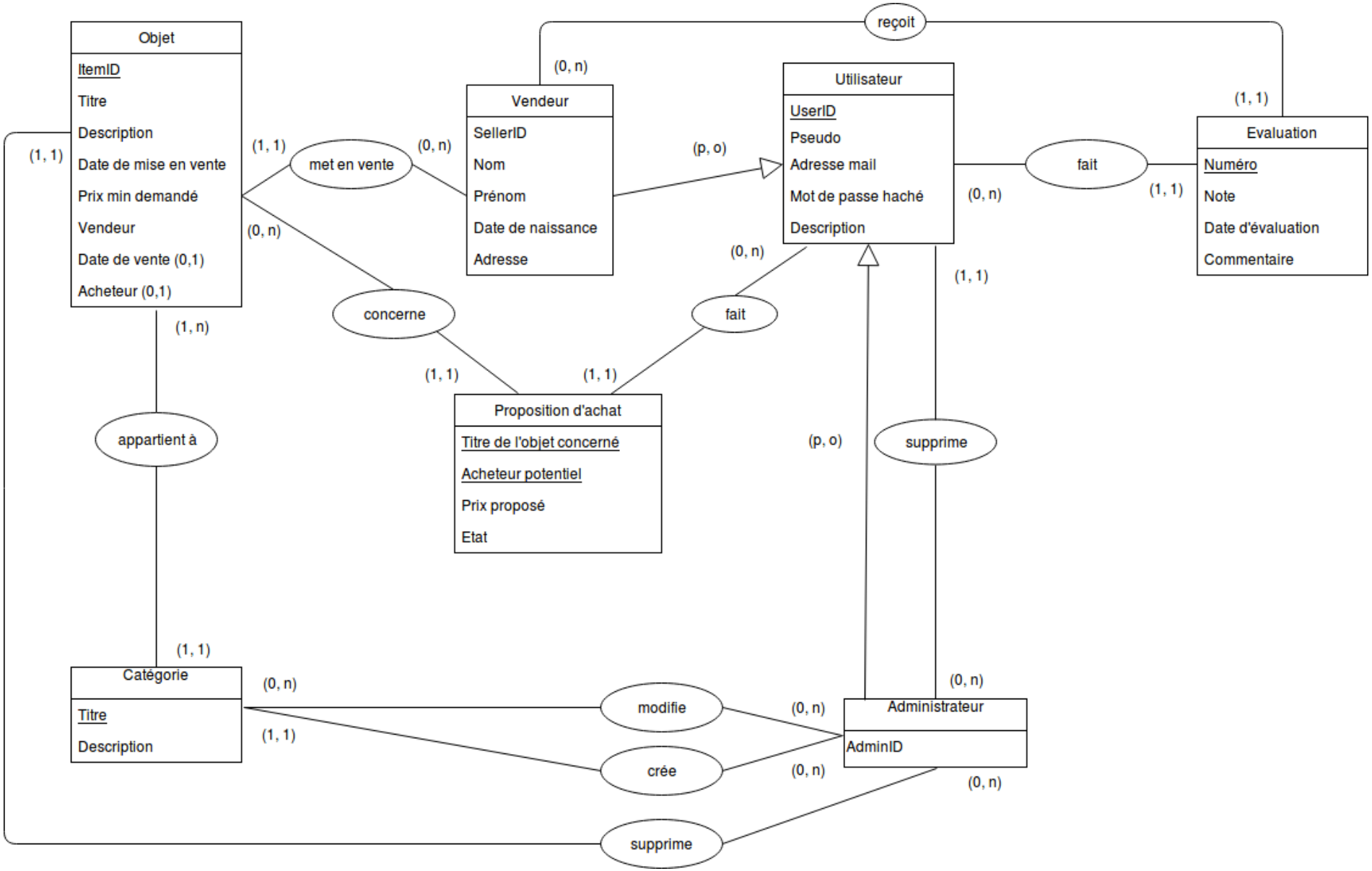
- Un objet appartient à 1 catégorie
- Une catégorie peut exister sans objet y étant associé
- Chaque objet est unique, il n'y a pas de notion de quantité
- Un utilisateur doit être majeur afin d'acheter des objets

1.2 Contraintes d'intégrités

- La date de vente d'un objet doit être > à la date de mise en vente de cet objet
- Le prix proposé pour une proposition d'achat doit être >= au prix minimum de l'objet
- Un administrateur ne peut pas se supprimer lui-même
- L'évaluation d'un objet est faite que si cet objet est acheté depuis moins de 10 jours, c'est-à-dire que la date d'évaluation doit être < à la date de vente de l'objet + 10
- La date d'évaluation d'un objet doit être > à la date de vente d'un objet
- Le pseudo d'un utilisateur est unique (c'est ce qui permet de différencier les utilisateurs)
- Le vendeur d'un objet ne peut pas être l'acheteur de cet objet
- la date de naissance d'un vendeur doit être > à la date de mise en vente d'un objet qu'il vend
- la note d'une évaluation est une valeur entre 1 et 10
- le numéro d'une évaluation doit être unique

2 Modèle Entité-Association

2.1 Schéma


Objet
ItemID
Titre
Description
Date de mise en vente
Prix min demandé
Vendeur
Date de vente (0,1)
Acheteur (0,1)
Vendeur
SellerID
Nom
Prénom
Date de naissance
Adresse
Utilisateur
UserID
Pseudo
Adresse mail
Mot de passe haché
Description
Evaluation
Numéro
Note
Date d'évaluation
Commentaire
Proposition d'achat
Titre de l'objet concerné
Acheteur potentiel
Prix proposé
Etat
Catégorie
Titre
Description
Administrateur
AdminID
met en vente
reçoit
fait
concerne
fait
appartient à
modifie
crée
supprime
supprime
(1, 1)
(0, n)
(p, o)
(1, n)

2.2 Justifications

La généralisation (Utilisateur, vendeur et administrateur) dans le schéma, est partiel et couvrant.

- Partiel: Les vendeurs et administrateurs sont des types d'utilisateurs. Mais il y a aussi les utilisateurs classiques.
- Couvrant: Un utilisateur peut être à la fois vendeur et administrateur par exemple.

Evaluations : une évaluation a pour clef un Numéro (qui doit donc être unique) car ses autres attributs ne permettent pas de la distinguer de manière unique (ex : deux évaluations peuvent avoir la même note et la même date).

Utilisateur : un utilisateur quelque soit son statut (simple utilisateur, vendeur et/ou administrateur) reste toujours identifié par son Pseudo.

Proposition d'achat : un proposition d'achat est identifiée à la fois par le Titre de l'objet concerné et par l'Acheteur potentiel, car on ne peut se baser uniquement sur le Titre de l'objet (il peut y avoir plusieurs propositions d'achat pour un même objet) ou sur le Pseudo de l'acheteur potentiel (un acheteur peut faire plusieurs propositions d'achats sur différenets Objets).

3 Modèle Relationnel

Dans cette section, nous traduisons le modèle entité-association de la section précédente en un modèle relationnel et en ajoutant ses contraintes.

3.1 Modèle

3.1.1 Utilisateur

UserID	MotDePasse	Pseudo	AddresseMail	Description

3.1.2 Administrateur

AdminID

- Administrateur.AdminID référence Utilisateur.UserID
  → L'administrateur est une généralisation de l'utilisateur, il en hérite

3.1.3 Vendeur

SellerID	Nom	Prénom	DateNaissance	Adresse

- Vendeur.SellerID référence Utilisateur.UserID
  → Le vendeur est une généralisation de l'utilisateur, il en hérite

3.1.4 Catégorie

Titre	Description_cat	AdminID

- Catégorie.AdminID référence Administrateur.AdminID
  → Une catégorie est créée par (1,1) administrateur

3.1.5 Objet

ItemID	Titre	Description_obj	DateMiseEnVente	PrixMin	DateVente	Acheteur	SellerID	Categorie

- Objet.SellerID référence Vendeur.SellerID
  → Un objet est mis en vente par (1,1) vendeur

- Objet.Categorie référence Categorie.Titre
  → Un objet appartient à (1,n) catégories

3.1.6 Evaluation

Numéro	Buyer	Seller	Time	Rate	Commentaire

- Evaluation.Seller référence Vendeur.SellerID
  → une evaluation est reçue par (1,1) vendeur
- Evaluation.Buyer référence Utilisateur.UserID
  → une evaluation est faite par (1,1) utilisateur

3.1.7 PropositionAchat

ItemID	Time	Buyer	price	accepted

- PropositionAchat.ItemID référence Objet.ItemID
  → Une proposition d'achat est concernée par (1,1) objet
- PropositionAchat.Buyer référence Utilisateur.UserID
  → Une proposition d'achat est faite par (1,1) utilisateur

3.2 Contraintes d'intégrité

- Pour tout Objet.Titre, il existe au moins un Appartenance.TitreObj
- Pour un Objet, Objet.DateVente > Objet.DateMiseEnVente
- Pour un PropositionAchat, PropositionAchat.Prix >= Objet.PrixMin
- Pour un Utilisateur, Administrateur.Pseudo != Utilisateur.Pseudo (un Administrateur ne peut pas se supprimer)
- Pour un Evaluation, Evaluation.Date < Objet.DateVente + 10
- Pour un Evaluation, Evaluation.Date > Objet.DateVente
- Pour un Vendeur, Vendeur.DateNaissance >= 18
- Pour un Vendeur, Vendeur.DateNaissance > Objet.DateMiseEnVente
- Pour un Objet, Objet.Vendeur != Objet.Acheteur

4 Gestion des données

Explications des méthodes utilisées pour l'extraction des données

Pour insérer les données des fichiers dans la base de données, nous avons utilisé la commande "LOAD DATA LOCAL INFILE" pour les fichiers .txt et la commande "LOAD DATA LOCAL INFILE" pour les fichiers .xml.

5 Requêtes demandées

5.1 R1

→ Les vendeurs qui sont appréciés par les utilisateurs ayant les memes gouts que l'utilisateur 'Jules'. Les goûts d'un utilisateur' sont définis par les vendeurs auxquels cet utilisateur a donné une évaluation supérieure (ou égale) à 3 sur 5.

5.1.1 SQL

```
SELECT DISTINCT v1.SellerID FROM Vendeur v1, Evaluation e1, Utilisateur u1
WHERE e1.Seller = v1.SellerID AND u1.UserID = e1.Buyer AND e1.Rate >= 3 AND u1.UserID in
        SELECT u2.UserID
        FROM Vendeur v2, Evaluation e2, Utilisateur u2
        WHERE u2.UserID = e2.Buyer AND v2.SellerID = e2.Seller
        AND e2.Rate > 3 AND v2.SellerID in
                SELECT v3.SellerID
                FROM Vendeur v3, Utilisateur u3, Evaluation e3
                WHERE e3.Seller = v3.SellerID AND e3.Buyer = u3.UserID
                AND u3.UserID = 41 AND e3.Rate > 3))
```

5.1.2 Algèbre relationnel

$JulesID \leftarrow \pi_{SellerID}(\sigma_{Prenom=\text{“}Jules''}(Vendeur))$
$SellerIDLikedByJules \leftarrow \pi_{SellerID}(\sigma_{Buyer=JulesID \wedge Rate \geq 3}(Evaluation))$
$BuyerWhoLikeAsJules \leftarrow \pi_{Buyer}(\sigma_{Rate \geq 3}(Evaluation *_{Seller=SellerID} LikedByJules))$
$R1 \leftarrow \pi_{SellerID}(\sigma_{Rate \geq 3}(BuyerWhoLikeAsJules * Evaluation))$

5.1.3 Calcul tuple

{v2.SellerID | Vendeur(v2) ∧ Evaluation(e3) ∧ Vendeur(v) ∧ v.Prenom = "Jules" ∧
∀u (Utilisateur(u) ∧
∃e1 ∃e2 ∃v1 (Evaluation(e1) ∧ Evaluation(e2) ∧ Vendeur(v1) ∧ v1.SellerID = e1.Seller ∧
v1.SellerID = e2.Seller ∧ e1.Buyer = u.UserID ∧ e2.Buyer = v.SellerID ∧ e1.Rate >= 3 ∧
e2.Rate >= 3) → e3.Buyer = u.UserID ∧ e3.Seller = v2.SellerID) }

5.2 R2

→ Les vendeurs ayant vendu et acheté à la meme personne

5.2.1 SQL

```
SELECT DISTINCT v.SellerID
FROM Vendeur v, Objet o1
WHERE v.SellerID = o1.SellerID AND o1.Acheteur IN
        SELECT DISTINCT v2.SellerID
        FROM Vendeur v2, Objet o2
        WHERE v2.SellerID = o2.SellerID AND v.SellerID = o2.Acheteur)
```

5.2.2 Algèbre relationnel

$R2 \leftarrow \pi_{ItemID}((Objet \bowtie_{Buyer=Seller} \pi_{Seller,Buyer}(Objet)) \cap (Objet \bowtie_{Seller=Buyer} \pi_{Seller,Buyer}(Objet)))$

5.2.3 Calcul tuple

{v.SellerID | Vendeur(v) ∧ Objet(o1) ∧ v.SellerID = o1.SellerID ∧ ∃ v2 ∃ o2 (Vendeur(v2) ∧ Objet(o2)
∧ v2.SellerID = o2.SellerID ∧ v.SellerID = o2.Acheteur ∧ o1.Acheteur = v2.SellerID)}

5.3 R3

→ Les objets vendus à un prix inférieur au montant d'une proposition d'achat de cet objet

5.3.1 SQL

```
SELECT DISTINCT o.ItemID
FROM Objet o, PropositionAchat p
WHERE o.ItemID = p.ItemID AND p.accepted = 'True' AND
        EXISTS (SELECT *
        FROM PropositionAchat p1
        WHERE p1.ItemID = p.ItemID AND p1.price > p.price AND p1.accepted = 'False')
```

5.3.2 Algèbre relationnel

$Vendus \leftarrow \pi_{ItemID,price}(\sigma_{accepted='True'}(PropositionAchat))$
$\alpha_{price:PrixVendu}(Vendus)$
$\alpha_{ItemID:ItemVenduID}(Vendus)$
$R3 \leftarrow \pi_{ItemID}(\sigma_{price \ln PrixVendu}(Vendus *_{ItemVenduID=ItemID} PropositionAchat))$

5.3.3 Calcul tuple

{o.ItemID | Objet(o) ∧ PropositionAchat(p) ∧ o.ItemID = p.ItemID ∧ p.accepted = "True" ∧ ∃p1 (PropositionAchat(p1) ∧ p1.ItemID = p.ItemID ∧ p1.price > p.price ∧ p1.accepted = "False")}

5.4 R4

→ Le/Les vendeurs ayant vendu le plus d'objets dans la meme catégorie

5.4.1 SQL

```
SELECT v.SellerID AS ID
FROM Vendeur v, Categorie c
WHERE ( SELECT COUNT(*)
        FROM Objet o
        WHERE v.SellerID = o.SellerID AND o.Categorie = c.Titre)

        >=

        (SELECT MAX(tot_count.sold)
        FROM (SELECT COUNT(*) as sold, c1.Titre as cat
                FROM Objet o1, Vendeur v1, Categorie c1
                WHERE v1.SellerID = o1.SellerID AND o1.Categorie = c1.Titre
        GROUP BY v1.SellerID ORDER BY COUNT(*) DESC) tot_count
        WHERE cat = c.Titre)
```

5.5 R5

→ Les objets pour lesquels au moins dix propositions d'achats ont été refusées

5.5.1 SQL

```
SELECT p.ItemID
FROM PropositionAchat p
WHERE p.accepted = "False"
GROUP BY p.ItemID
HAVING COUNT(*) >= 10
```

5.6 R6

→ Les vendeurs avec le nombres moyen d'objets vendus, le nombre moyen d'objet achetés, la note qu'ils ont reçue, et la moyenne des notes qu'ils ont donné, et ce pour tous les vendeurs ayant vendu et acheté au moins 10 objets.

5.6.1 SQL

```
SELECT v.SellerID AS ID,
(SELECT COUNT(o1.Titre)
FROM Objet o1
WHERE o1.SellerID = v.SellerID) AS Moy_Nb_obj_vendus,

    (SELECT AVG(o2.Titre)
FROM Objet o2
WHERE o2.Acheteur = u.Pseudo) AS nb_moy_obj_achetés,

    (SELECT MAX(e.rate)
FROM Evaluation e
WHERE e.Seller = v.SellerID) AS notes_recues,

    (SELECT AVG(e1.rate)
FROM Evaluation e1
WHERE e1.Buyer = v.SellerID) AS moy_notes_données

    FROM Vendeur v WHERE (SELECT COUNT(*)
FROM Objet o WHERE v.SellerID = o.Acheteur) >= 10
AND (SELECT COUNT(*)
FROM Objet o WHERE v.SellerID = o.SellerID) >= 10
```

6 Choix et hypothèses

Explications et justifications des choix et hypothèses qu'on a pris

- Pour la requête 1, on part du principe que si il y a ne serait ce qu'un meme élément en commun (vendeur apprécié), alors les 2 utilisateurs ont les memes gouts.
- Dans la table Administrateur, nous avons inséré un administrateur par défaut "Master" qui se chargera de créer les autres administrateurs.
- Nous avons décidé de ne pas stocker les suppressions et modifications faites par les administrateurs dans des tables dédiées.