

Projet Bases de données : Gange

Gander Félix
Bossuet Céline
Zahran Antony
Chahbani Alaeddine
Doan-Van Louis

Lundi 6 Décembre 2021
3MMPDB – Projet BD
ENSIMAG – 2A : Filière ISI

La société de commerce électronique destinée aux particuliers Gange souhaite mettre en place une base de données permettant à ses clients de vendre et d'acheter des produits aux enchères. Notre équipe a été missionnée pour la réalisation de ce projet qui se découpe en plusieurs phases :

- Analyse du problème et de ses contraintes
- Conception et implémentation d'une base de données
- Réalisation des fonctionnalités via des transactions SQL
- Création d'une application Java réalisant les demandes clients

I – Analyse du problème sous forme de contraintes

Dépendances Fonctionnelles	Contraintes de Valeur	Contraintes de Multiplicité	Contraintes contextuelles
ID_prod ⇒ intitulé, prix_produit, texte, URL, nom_cat	prix_produit > 0	ID_prod --> nom_cat	
nom_caract, ID_prod ⇒ valeur			
		nom_cat -/-> nom_pere	
		ID_uti -/-> mail	ID modifié si un utilisateur est supprimé
mail ⇒ mdp, nom, prenom, adresse, ID			
ID_prod, date ⇒ ID_uti, prix	prix > prix_produit (lors de l'insertion)	Achat : ID_prod -/->date Offre : ID_prod -/->>date	ID_uti, ID_prod définis avant

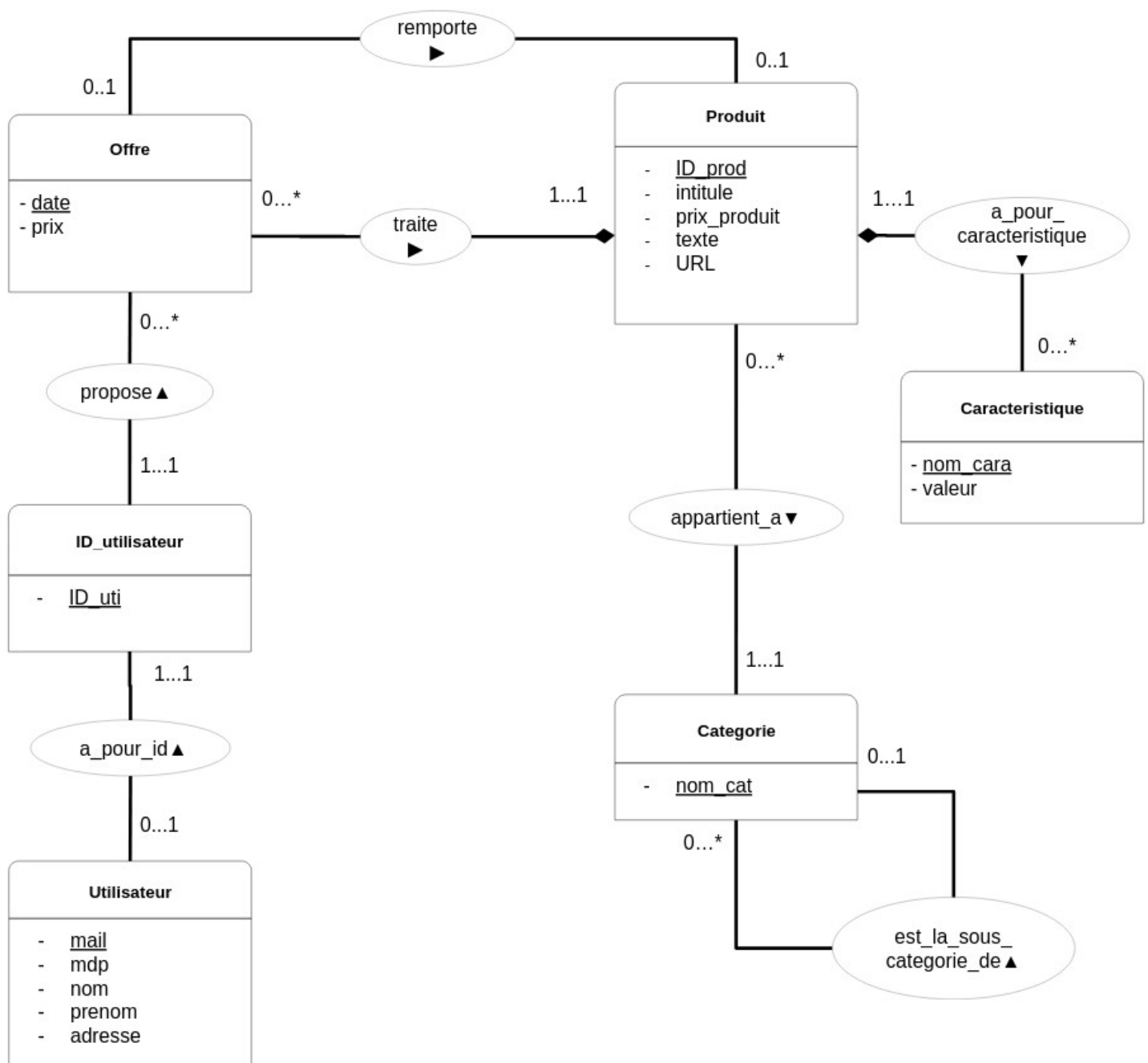
Parmi les contraintes présentes dans ce tableau, certaines nécessitent une justification et d'autres n'ont pas pu être représentées.

D'après les spécifications du client, il n'y a qu'une seule valeur associée à une caractéristique pour un certain produit. D'où la dépendance à la seconde ligne du tableau. Un utilisateur est identifié par son mail et non son ID, il est important de les dissocier. Lorsqu'il est supprimé, son ID correspondant est modifié et ses informations retirées.

Un utilisateur peut enchérir sur sa propre offre d'après le client. Un achat correspond à une seule offre gagnante, il peut n'y avoir aucune offre ou plusieurs. Le nombre d'offres n'est pas dans les dépendances de manière à pouvoir modifier les conditions de ventes, il sera calculable.

II – Conception du schéma Entités/Associations

Le schéma suivant représente les dépendances mises en lumière dans la première partie. Il ne contient cependant pas les contraintes de valeurs.



Dans ce schéma, nous avons pris plusieurs décisions clés. Tout d'abord, la relation entre catégorie et sous-catégorie est représentée par une association sur une même entité. Cela permettra à l'avenir un parcours récursif des catégories.

Ensuite, l'utilisateur est lié à un ID mais peut être retiré sans impacter les autres entités qui ne traitent qu'avec des ID. Ceci permet de respecter le droit à l'oubli présenté dans le cahier des charges.

Le système d'offres proposées et remportées est représenté par 2 associations ne permettant à un produit de n'être qu'une seule fois acheté suite à une offre proposée. De plus, ces produits possèdent un nombre variable de caractéristiques propres à celui-ci, d'où le choix de le représenter par une entité faible.

III – Passage au Relationnel

La réalisation des tables présentes dans la base de données s'appuie sur le schéma présenté précédemment. Nous l'avons ainsi traduit par les tables suivantes :

PRODUIT (ID_prod, intitulé, prix_prod, texte, URL, nom_cat)

CARACTERISTIQUE (ID_prod, nom_cara, valeur)

ID_UTILISATEUR (ID_uti)

UTILISATEUR (mail, ID_uti, mdp, nom, prénom, adresse)

OFFRE (ID_prod, date_heure, prix, ID_uti)

REMORTE (ID_prod, date_heure)

CATEGORIE (nom_cat)

EST_LA_SOUS_CATEGORIE_DE (nom_cat, nom_pere)

Tout attribut présent dans les différentes tables ne contient que des valeurs atomiques non nulles. Afin de respecter cette règle et d'obtenir une forme normale n°1, nous avons dû séparer les caractéristiques en une table disjointe de celle des produits car d'après le client, un produit peut posséder plusieurs caractéristiques différentes.

Tous les attributs non clés sont pleinement dépendants de chacune des clés de la table associée. La forme normale n°2 est alors respectée pour toutes les tables.

La forme normale n°3 de Boyce-Codd-Kent est respectée car les attributs non clés dépendent uniquement des clés et en aucun cas des autres attributs. Les dépendances fonctionnelles non triviales de la forme $X \rightarrow Y$ sont telles que X contient une clé de la table.

Il subsiste des contraintes non implémentables dans la définition des tables comme le nombre d'offres limité qui doit être modifiable. Celui-ci sera calculé à l'aide de transactions SQL lorsque nécessaire (demande d'achats, recommandations, etc). Les contraintes de valeur seront également vérifiées par la suite lors des insertions d'offres. De plus, la contrainte qu'une offre proposée ait un prix supérieur au produit ciblé n'est pas non plus implémentable à la création de ses tables.

IV – Analyse des fonctionnalités : Transactions

Le script nécessaire à la création des tables est joint à ce rapport. Il est également accompagné d'un script de remplissage de la base afin de pouvoir tester directement les transactions suivantes.

1. Parcours du catalogue

Un utilisateur peut entrer son mail et mot de passe pour se connecter à son compte. Il doit être inscrit dans la base préalablement et saisir les bons mail et mot de passe.

```
SELECT * FROM UTILISATEUR WHERE mail LIKE ? AND mdp LIKE ?;
```

Désormais, l'utilisateur est capable de parcourir le catalogue des produits en fonction des recommandations qui lui sont proposées. Les catégories qui ont, par le passé, fait l'objet d'offres de la part de l'utilisateur sont mis en avant. Les offres ayant débouchées sur un achat ne sont pas prises en considération.

```
SELECT nom_cat, COUNT(*) AS nb_offres FROM OFFRE
JOIN PRODUIT ON PRODUIT.ID_prod = OFFRE.ID_prod
JOIN UTILISATEUR ON UTILISATEUR.ID_uti = OFFRE.ID_uti
WHERE UTILISATEUR.mail = ? AND
      NOT EXISTS (SELECT * FROM REMPORTE JOIN OFFRE OFR ON
                  REMPORTE.ID_prod = OFR.ID_prod AND
                  REMPORTE.date_heure = OFR.date_heure
                  WHERE OFR.ID_uti = OFFRE.ID_uti AND
                  OFFRE.ID_prod = REMPORTE.ID_prod)
GROUP BY nom_cat ORDER BY nb_offres DESC, nom_cat;
```

De plus, les catégories possédant le plus d'offres en moyenne par produit sont aussi mises en avant. Tout cela est ordonnée par nombre d'offre décroissant. On veille à ne pas afficher de doublons avec les catégories déjà prises en compte dans les recommandations spécifiques au client.

```
SELECT nom_cat, AVG(nb_offres) AS moyenne_offre FROM
  (SELECT prod.ID_prod, prod.nom_cat, COUNT(*) AS nb_offres
   FROM PRODUIT prod JOIN OFFRE ON OFFRE.ID_prod = prod.ID_prod
   GROUP BY prod.ID_prod, prod.nom_cat
   UNION SELECT prod1.ID_prod, prod1.nom_cat, 0 as nb_offres
    FROM PRODUIT prod1 WHERE prod1.ID_prod
    NOT IN (SELECT OFFRE.ID_prod from OFFRE)
    GROUP BY prod1.ID_prod, prod1.nom_cat)
WHERE nom_cat NOT IN
  (SELECT DISTINCT PRODUIT.nom_cat FROM OFFRE
   JOIN PRODUIT ON PRODUIT.ID_prod = OFFRE.ID_prod
   JOIN UTILISATEUR ON UTILISATEUR.ID_uti = OFFRE.ID_uti
   WHERE UTILISATEUR.mail = ? AND
    NOT EXISTS (SELECT * FROM REMPORTE
     JOIN OFFRE OFR ON REMPORTE.ID_prod = OFR.ID_prod
     AND REMPORTE.date_heure = OFR.date_heure
     WHERE OFR.ID_uti = OFFRE.ID_uti AND
      OFFRE.ID_prod = REMPORTE.ID_prod))
GROUP BY nom_cat" ORDER BY moyenne_offre DESC, nom_cat;
```

Concernant la requête SQL, nous avons fait une requête dans la section FROM mais nous aurions très bien pu le faire en deux temps via la création d'une vue temporaire contenant la somme des produits.

Afin d'obtenir plus de détails sur un produit, il est possible d'afficher toute sa fiche produit.

```
SELECT * FROM PRODUIT WHERE ID_prod=?;
SELECT * FROM CARACTERISTIQUE WHERE ID_prod=?;
```

2. Enchères

On récupère les identifiants de l'utilisateur ainsi que du produit souhaité, si celui-ci est toujours disponible à la vente (c'est à dire qu'il n'est pas encore remporté). On vérifie également que le prix proposé soit supérieur au prix actuel du produit.

```
SELECT ID_uti FROM UTILISATEUR WHERE mail=?;  
SELECT ID_prod FROM PRODUIT WHERE ID_prod=? AND prix_produit < ? AND ID_prod  
NOT IN (SELECT ID_prod FROM REMPORTE);
```

On insère l'offre proposée et on met à jour le prix du produit.

```
INSERT INTO OFFRE VALUES(?, (SELECT LOCALTIMESTAMP FROM dual), ?, ?);  
UPDATE PRODUIT SET prix_produit=? WHERE ID_prod=?;
```

Il faut vérifier si l'offre remporte le produit. Pour cela, on compte le nombre d'offres faites sur ce même produit. La cinquième offre remporte le produit et l'insère dans la table remporte car le produit n'est plus en vente.

```
SELECT COUNT(*) FROM OFFRE WHERE ID_prod = ?;  
INSERT INTO REMPORTE VALUES  
  (?, ((SELECT date_heure FROM OFFRE WHERE ID_prod = ?) MINUS  
    (SELECT DISTINCT O1.date_heure FROM OFFRE O1  
      JOIN OFFRE O2 ON O1.ID_prod = O2.ID_prod AND O1.ID_prod = ?  
        WHERE O1.date_heure < O2.date_heure))));
```

3. Droit à l'oubli

On génère un nouvel ID à l'utilisateur et on l'ajoute dans la table. L'ID généré n'appartient à aucun utilisateur et n'est pas déjà présent dans la base.

```
SELECT max(id_uti) FROM ID_UTILISATEUR;  
INSERT INTO ID_UTILISATEUR VALUES (?);
```

Ensuite, on cherche l'ID de l'utilisateur pour trouver toutes les offres et achats qu'il a pu réaliser avec l'application.

```
SELECT id_uti FROM UTILISATEUR WHERE mail LIKE ?;  
SELECT ID_prod, date_heure, ID_uti FROM OFFRE WHERE ID_uti = ?;
```

Afin de conserver l'anonymat, on remplace l'ID de toutes ces offres par celui généré auparavant.

```
UPDATE OFFRE SET id_uti = ? WHERE id_uti = ? AND date_heure = ? AND ID_prod = ?;
```

L'utilisateur peut désormais être supprimé de la base sans impacter les transactions qu'il a déjà effectuées qui sont maintenant anonymisées.

```
DELETE FROM UTILISATEUR WHERE mail LIKE ?;  
DELETE FROM ID_UTILISATEUR WHERE id_uti = ?;
```

V – Bilan du projet

Finalement, il est intéressant de faire une rétrospective du projet vis à vis de l'équipe et du sujet en lui-même, afin de comprendre comment nous en sommes arrivés là.

1. Répartition des tâches

Au lancement du projet, il était très difficile de paralléliser les tâches car les dépendances entre l'analyse fonctionnelle, le schéma entité/association et le passage au relationnel étaient bien trop fortes. Nous avons donc commencé par travailler tous ensemble sur l'analyse du problème pour avoir rapidement une base pour la suite. À partir de ce point, nous avons construit un premier schéma entité/association qui a évolué par la suite.

Ensuite, nous avons pu diviser le travail par fonctionnalités à implémenter. Les transactions étaient assez indépendantes pour être traitées en même temps. En parallèle, le code Java était mis en place pour implémenter directement les transactions SQL dans l'application. Au fur et à mesure que les transactions étaient validées, nous pouvions les insérer et les tester via l'application.

La préparation des rendus (code source, script, slides et rapport) a aussi pu être découpée et répartie au sein de l'équipe.

2. Points difficiles et solutions apportées

Durant ce projet, deux points ont majoritairement posé problème pour la réalisation de l'application.

Le premier point vise la compréhension exacte des spécifications du client, à la fois à partir du sujet mais aussi des demandes de précisions pendant les séances. Des incompréhensions nous ont à plusieurs moments conduits vers de mauvaises décisions d'implémentation. Par exemple, le fait qu'une sous-catégorie pouvait découler de 2 catégories pères impliquait l'ajout d'une table dédiée, cela impactait les cardinalités du schéma entité/association, les tables SQL et les transactions. Pour éviter au maximum ces incompréhensions, nous n'avons pas hésité à demander à plusieurs reprises confirmation auprès du client.

Ensuite, le second point se concentre sur la relation entre achats et offres. En effet, un achat représente une offre qui a abouti, par conséquent il était envisageable de créer un sous-typage. Cette idée avait été mise de côté mais il fallait tout de même éviter de créer une table Achat similaire à Offre à cause de la redondance. La solution que nous avons alors proposée est une deuxième association entre Offre et Produit permettant de savoir si l'offre est celle qui a remporté le produit voulu. Celle-ci est unique avec les contraintes de clé primaire.

Annexe : Mode d'emploi de l'application

Menu principal

```
Bienvenue dans le Gange
Connexion ici
-----
Que voulez-vous faire ?
1 - Connexion utilisateur
2 - Afficher un produit
3 - Afficher les catégories
4 - Afficher les produits d'une catégorie
5 - Afficher les utilisateurs
0 - Quitter l'application
```

Connexion utilisateur

```
Saisir les informations de connexion :
Mail :
tina@gmail.com
Mot de passe :
password
Tina Tasser connecté !
-----
Session de tina@gmail.com, que souhaitez-vous faire ?
1 - Proposer une enchère sur un produit
2 - Afficher les catégories & sous-catégories
3 - Afficher les catégories & sous-catégories recommandés
4 - Afficher les produits d'une catégorie
5 - Voir la fiche d'un produit
6 - Afficher les utilisateurs
7 - Supprimer votre compte (irréversible)
0 - Déconnexion
```

```
Saisir les informations de connexion :
Mail :
tina@gmail.fr
Mot de passe :
1234
Mauvais mot de passe/mail
```

Afficher les recommandations pour l'utilisateur

```
-> chaussures_de_ville  
-> CD  
-> chaussures_randonnee  
-> manteau  
-> pantalons  
-> tennis
```

Afficher la fiche produit

```
Voici la fiche complète du produit numéro 12  
intitulé : chaussures de ville  
prix : 21 euros  
description du produit :  
chaussures GEOX, qualité sans précédent  
URL : https://chaussure_de_ville  
présent dans la catégorie : chaussures_de_ville  
Caractéristique pointure : 38  
Caractéristique couleur : noir
```

Proposer une offre

```
Saisir le numéro du produit :  
12  
Saisir le prix proposé :  
25  
Votre enchère a bien été prise en compte !
```

Supprimer un utilisateur

```
Toutes vos données personnelles vont être supprimées et vous ne pourrez plus vous connecter à ce compte  
Etes-vous certain(e) que vous voulez supprimer votre compte?  
1 - Confirmation  
2 - Annulation  
1  
Suppression terminée
```

Certaines fonctionnalités jugées utiles ont été ajoutées. Elles peuvent être facilement masquer si le client souhaite les retirer. Elles sont cependant utiles pour la démonstration.

Affiche les utilisateurs

```
-> ID 1 : Tim Desmoulin  
-> ID 2 : Tina Tasser  
-> ID 3 : John Doe  
-> ID 4 : Jane Doe  
-> ID 5 : Socrate Petteng
```

Affiche toutes les catégories et sous-catégories

```
-> CD  
-> vêtements  
--> manteau  
--> chaussures  
---> tennis  
---> chaussures_randonnee  
---> chaussures_de_ville  
--> pantalons
```

Affiche les produits à vendre appartenant à une certaine catégorie

```
Saisir la catégorie recherchée :  
chaussures_randonnee  
-> ID 11 : chaussures de rando 18  
-> ID 14 : chaussures de rando 18
```

