ⵜⴰⵖⵣⵓⴰⵜ ⵏ ⵜⵙⴰⴷⴰⵡⵉⵏ
FACULTÉ DES SCIENCES

MARRAKECH
جامعة القاضي عياض
UNIVERSITÉ CADI AYYAD

كليـــــــــة العلوم
FACULTY OF SCIENCE

Semlalia

Projet de Fin de Module : Bases de Données Avancées & PL/SQL

Titre du Projet : Conception et Réalisation d'une Marketplace

E-commerce Multi-vendeurs

Réalisé par :

El Houcine Rahmouni, Zakaria Oulamine

Encadré par :

Hamza Er-rahmouny

Filière :

S5 Sciences de l'Informatique

Année universitaire : 2025/2026

Remerciements

Avant d'entamer la présentation de ce travail, nous tenons à exprimer notre profonde gratitude et nos plus sincères remerciements à notre professeur, M. Hamza Er-rahmouny, pour ses conseils précieux, sa disponibilité et son encadrement rigoureux tout au long de la réalisation de ce projet.

Nos remerciements s'étendent également à l'Université Cadi Ayyad et au corps professoral de la Faculté des Sciences Semlalia, qui nous ont fourni un environnement d'apprentissage stimulant et les ressources nécessaires à notre formation.

Enfin, nous souhaitons souligner l'excellente collaboration et l'esprit d'équipe qui ont animé notre groupe de travail. Ce projet est le fruit d'un effort commun et d'un engagement partagé.

Introduction Générale

1. Contexte Général du Projet

Le présent rapport est élaboré dans le cadre du projet de fin de module de "Bases de Données Avancées & PL/SQL", une composante essentielle du cinquième semestre de la filière Sciences de l'Informatique. Ce projet représente une opportunité fondamentale de mettre en application les concepts théoriques et les compétences pratiques que nous avons acquis en matière de modélisation, de conception de bases de données relationnelles et de programmation avancée.

2. Objectifs du Projet

L'objectif principal qui nous a été confié est la conception et la réalisation d'une plateforme de commerce électronique de type marketplace multi-vendeurs. Le système doit permettre à divers vendeurs de gérer leurs propres boutiques virtuelles, d'ajouter et de suivre leurs produits, tout en offrant aux clients une expérience d'achat fluide. Le cœur technique du projet repose sur une base de données Oracle robuste et une logique métier complexe qui sera implémentée en PL/SQL, conformément aux exigences du cahier des charges.

Chapitre 1 : Étude Préalable et Analyse des Besoins

Introduction du chapitre

Ce premier chapitre est consacré à l'étude préalable du projet. Il a pour objectif de définir le cadre général, de présenter le contexte métier et d'identifier de manière claire et précise les besoins fonctionnels et techniques auxquels notre solution devra répondre. Cette phase est fondamentale car elle conditionne la pertinence et le succès des étapes de conception et de développement qui suivront.

1.1. Contexte Général du Projet

Le projet s'inscrit dans un contexte économique moderne où la digitalisation des activités commerciales est devenue une nécessité stratégique. Il nous est demandé de réaliser une plateforme de type **marketplace e-commerce multi-vendeurs**. Contrairement à un site e-commerce classique qui ne présente les produits que d'une seule entité, une marketplace est un espace en ligne qui met en relation une multitude de vendeurs indépendants avec une large base de clients. Cette approche permet d'offrir une gamme de produits très variée et de créer un écosystème commercial dynamique.

Sur le plan académique, ce projet constitue une application pratique et intégrale des concepts étudiés dans le module "Bases de Données Avancées & PL/SQL". Il nous permettra de mobiliser nos compétences en modélisation de données via la méthode MERISE, en programmation de la logique métier avec PL/SQL (triggers, procédures, fonctions), et en développement d'interfaces web riches avec Oracle APEX.

1.2. Objectifs du Projet

À partir de l'analyse du cahier des charges, nous avons identifié un ensemble d'objectifs principaux que notre application doit atteindre pour être considérée comme fonctionnelle et complète. Ces objectifs se décomposent en exigences fonctionnelles et non fonctionnelles.

1.2.1. Exigences Fonctionnelles

Le système doit impérativement prendre en charge les fonctionnalités suivantes :

- **Gestion des Vendeurs :** Permettre aux vendeurs de créer et gérer leur profil, ainsi que de mettre en ligne et suivre leurs propres produits.
- **Gestion des Produits et Catégories :** Organiser les produits dans un catalogue structuré par catégories pour faciliter la navigation des clients.
- **Gestion des Clients :** Offrir aux clients la possibilité de créer un compte, de parcourir les produits, de les ajouter à un panier et de passer une commande.
- **Gestion des Commandes :** Assurer le traitement complet d'une commande, de sa création à sa validation, même si elle contient des produits de plusieurs vendeurs différents.
- **Paiement et Facturation :** Calculer automatiquement le montant total d'une commande, en appliquant les remises éventuelles, et permettre au client d'effectuer le paiement, générant ainsi une facture.
- **Gestion de l'Expédition :** Permettre le suivi de l'état d'avancement de l'expédition de chaque commande (en attente, en cours, livrée) en association avec un transporteur.
- **Gestion des Avis Clients :** Donner la possibilité aux clients de noter et de laisser des commentaires sur les produits qu'ils ont achetés.

1.2.2. Exigences Techniques

Le cahier des charges impose l'utilisation exclusive de la stack technologique Oracle :

- Système de Gestion de Base de Données : **Oracle Database XE**.
- Langage de la logique métier : **PL/SQL**.
- Framework de développement de l'interface : **Oracle APEX**.
- Outil de modélisation : **Draw.io** pour les diagrammes MERISE ou UML.

Conclusion du chapitre

Cette étude préalable nous a permis de cerner avec précision le périmètre du projet et de définir

une liste claire d'objectifs. Le chapitre suivant sera consacré à la conception détaillée du système d'information en réponse à ces exigences.

Chapitre 2 : Conception du Système d'Information

Introduction du chapitre

Après avoir défini le périmètre et les objectifs de notre projet dans le chapitre précédent, nous abordons maintenant la phase de conception. Ce chapitre est le cœur de notre travail de modélisation. Il a pour but de traduire les besoins fonctionnels identifiés en une structure de données formelle et robuste. Pour ce faire, nous allons justifier le choix de notre méthodologie de conception, puis présenter en détail les modèles conceptuel et logique des données qui serviront de fondation à notre base de données Oracle.

2.1. Choix de la Méthodologie de Conception

2.1.1. Présentation de la méthode MERISE

1. **Séparation claire des niveaux d'abstraction :**
    - Le **MCD (Modèle Conceptuel de Données)** décrit les informations du système sans se préoccuper de la technique (c'est le "quoi").
    - Le **MLD (Modèle Logique de Données)** adapte le modèle à un SGBD précis (le "comment").
    - Cette séparation permet de **mieux comprendre le système avant de le coder**.
2. **Cohérence et fiabilité :**

    La méthode Merise permet de **détecter les incohérences et les redondances** dans les données avant la création de la base.

    Ainsi, les erreurs sont corrigées très tôt, ce qui améliore la **qualité et la fiabilité du modèle**.

3. **Communication facilitée :**
   Les diagrammes (MCD, MLD) sont **visuels et faciles à comprendre**, ce qui facilite la communication entre les développeurs, les analystes et les utilisateurs.
4. **Gestion des dépendances fonctionnelles :**
   Merise permet d'identifier clairement les **liens logiques entre les entités**, les **contraintes d'intégrité** et les **cardinalités**.
5. **Documentation complète :**
   Le **dictionnaire de données** fournit une description précise de chaque donnée (nom, type, taille, signification, contraintes...), ce qui constitue une **base de documentation fiable** pour tout le projet.

2.1.2.Présentation de la méthode UML

Objectifs différents :

- La méthode UML est principalement conçue pour modéliser l'ensemble d'un système logiciel, incluant les comportements, les interactions et la logique orientée objet.
- La méthode Merise, quant à elle, est spécialisée dans la modélisation des données et des traitements d'un système d'information.

Ainsi, pour un projet centré sur la conception d'une base de données, Merise s'avère plus pertinente et mieux adaptée.

Adaptation au modèle relationnel

- UML décrit des classes et des objets, ce qui correspond davantage à la programmation orientée objet (Java, C++, etc.).
- Merise, en revanche, décrit des entités, des associations et des attributs, ce qui correspond directement à un modèle relationnel (tables, clés, relations).

Par conséquent, la transformation du MCD → MLD → SQL est plus simple, logique et claire avec Merise.

Simplicité et clarté pour la modélisation de données

- Merise propose une notation simple et intuitive basée sur les entités, les associations et les cardinalités.

- UML, en comparaison, peut devenir plus complexe et verbeuse pour les utilisateurs qui souhaitent uniquement concevoir une base de données sans programmation orientée objet.

Pertinence pédagogique et méthodologique

- Dans les contextes éducatifs ou administratifs, comme les projets d'analyse, de gestion ou de comptabilité, Merise offre une démarche méthodologique claire et structurée.
- UML, de son côté, est davantage utilisée dans les projets de développement orienté objet ou dans les architectures logicielles complexes.

2.1.3. Justification du choix de MERISE

Dans le cadre de notre projet, nous avons choisi d'utiliser la méthode Merise plutôt que la méthode UML.

Ce choix s'explique par la nature de notre travail, qui porte principalement sur la modélisation et la conception d'une base de données relationnelle.

La méthode Merise est particulièrement adaptée à ce type de projet, car elle permet de décrire de manière claire et progressive les données (MCD, MLD, MPD) et leurs relations.

À l'inverse, UML est davantage orientée vers la programmation orientée objet et la modélisation de systèmes logiciels complets, ce qui dépasse les besoins de notre projet.

Ainsi, l'utilisation de Merise nous a permis d'obtenir une modélisation plus simple, cohérente et directement exploitable pour la création de la base de données.

2.2. Modèle Conceptuel de Données (MCD)

Le MCD est une représentation abstraite de la structure des données du système, indépendante de toute technologie. Il identifie les objets d'information principaux (entités) et les liens sémantiques qui les unissent (relations).

2.2.1. Description et justification des entités du système e-commerce multi-vendeurs

1. VENDEUR

Justificatin :

Cette entité représente les vendeurs de la platforme .chaque vendeur dispose d'un compte personnel lei permettant de gérer sa boutique et ses produits.

Role :

Enregistrer ses informations personnelles et commerciales (nom,boutique,email,t éléphone …)

Ajouter,modifier ou supprimer ses produits .

Consulter et suivre les commandes associéees à ses produits.

2. PRODUIT

Justification :

Le produit constitue l'élément central de la platforme. Chaque produit possède des caractérisique propres (nom,prix,description,quantité en stock) et appartient à un seul vendeur .

Role :

Être affiché dans la boutique du vendeur .

Être inclus dans les commandes des clients .

Être classé dans une catégorie donnée .

3. CATEGORIE

Justification :

Les produits sont regroupés par catégories pour faciliter la navigation el la recherche.

Role :

Organiser les produits selon leur type (informatique,mode,livres …).

Permettre une hiérarchisation (catégories principales et sous-catégories).

4. CLIENT

Justification :

Le client est l'utilisateur final de la platforme .Il consulte les produits et effectue des achats.

Role :

Créer un compte personnel.

Passer des commandes et laisser des avis.

5. PANIER

Justification :

Avant de valider une commande, le client ajoute les produits souhaités à son panier .

Role :

Représenter la sélection ajoute ou suppression de produit.

Etre mis à jour à chaque ajoute ou suppression de produit.

6. **PANIER_ITEM**

Justification :

Cette entité décrit les détails des produits présents dans un panier (quantité, prix unitaire …).

Role :

Lier un produit à un panier .

Permettre de connaitre le contenu exact u panier du client .

7. **COMMANDE**

Justification :

Une commande enregistre chaque achat effectué par un client .

Role :

Contenir plusieurs produits provenant de différents vendeurs.

Indiquer l'état de la commande (en attent, payée, expédiée) .

Inclure les informations du client et les adresses associées .

8. **COMMANDE_ITEM**

Justification :

Cette entité représente un produit spécifique au sein d'une commande.

Role :

Lier chaque produit et vendeur à une commande donnée.

Enregistrer la quantité et le prix appliqué pour chaque ligne de commande.

9. **PAIMENT**

Justification :

Permet d'assurer le suivi des transactions financière liées aux commandes.

Role :

Enregistrer chaque opération de paiment (carte, paypal, etc.).

Suivre l'état du paiment (en attente, réussi, échoué).

10. **EXPEDITION**

Justification :

Une fois la commande payée, le processus d'expédition permet d'assurer la livraison au client.

Role :

Contenir les informations de livraison (date, numéro de suivi, status) .

Etre associée aux éléments de commande (COMMANDE_IEM).

11. **TRANSPORTEUR**

Justification :

Cette entité représente les sociétés chargées de la livraison (DHL, Arames, etc.) .

Role :

Enregistrer les informations des transporteurs (nom, contact, délai moyen) .

Etre liée à chaque expédition réalisée.

12. **AVIS**

Justification :

Les clients peuvent évaluer les produits achetés pour aider les autres utilisateurs.

Role :

Enregistrer les notes et commentaires laissés par les clients.

Améliorer la fiabilité et la réputation des produits et vendeurs .

2.2.2. Relations & cardinalités (MCD Merise - description)

- **VENDEUR ---- PRODUIT**

Relation : un vendeur possède plusieurs produits .

Cardinalités :

VENDEUR (1,1) ---> (0,N) PRODUIT

Un vendeur peut proposer 0 à N produits .

Un produit appartient à un seul vendeur .

- **CATEGORIE ---- PRODUIT**

Relation : Une catégorie regroupe plusieurs produits .

Cardinalités :

CATEGORIE (1,1) ---> (0,N) PRODUIT

Un produit appartient à une seule catégorie (ou plusieurs si on utilise un table d'association).

- **CLIENT ---- PANIER**

  Relation : Un client possède un ou plusieurs paniers (souvent un seul panier actif).

  Cardinalités :

  CLIENT (1,1) ----> (0,N) PANIER

  Un panier appartient à un seul client .

- **PANIER ---- PANIER_ITEM ----- PRODUIT**

  Relation :

  - Un panier contient plusieurs paniers_item .
  - Chaque panier_item correspond à un produit .

  Cardinalités :

  PANIER (1,1) ---> (1,N) PANIER_ITEM

  PANIER_ITEM (1,1) ---> (1,1) PRODUIT

- **CLIENT ---- COMMANDE**

  Relation : Un client passe plusieurs commandes .

  Cardinalités :

  CLIENT (1,1) ---> (0,N) COMMANDE

  Une commande est passée par un seul client .

- **COMMANDE ---- COMMANDE_ITEM ---- PRODUIT**

  Relation :

  - Une commande contient plusieurs commande_item .
  - Chaque commande_item correspond à un produit vendu par un vendeur .

  Cardinalités :

  COMMANDE (1,1) ---> (1,N) COMMANDE_ITEM

  COMMANDE_ITEM(1,1) ---> (1,1) PRODUIT

  COMMANDE_ITEM(1,1) ---> (1,1) CLIENT

COMMANDE ---- PAIMENT

Relation : Une commande peut avoir un ou plusieurs paiments (si paiments partiels) .

Cardinalités :

COMMANDE (1,1) ---> (0,N) PAIMENT

Un paiment est associé à une seule commande .

- **COMMANDE_ITEM ---- EXPEDITION ---- TRANSPORTEUR**

Relation :

- Chaque commande_item peut avoir une expédition .
- Une expédition est effectuée par un transporteur .

Cardinalités :

COMMANDE_ITEM (1,1) ---> (0,1) EXPEDITION

EXPEDITION (1,1) ---> (1,1) TRANSPORTEUR

- **PRODUIT ---- AVIS ---- CLIENT**

Relation :

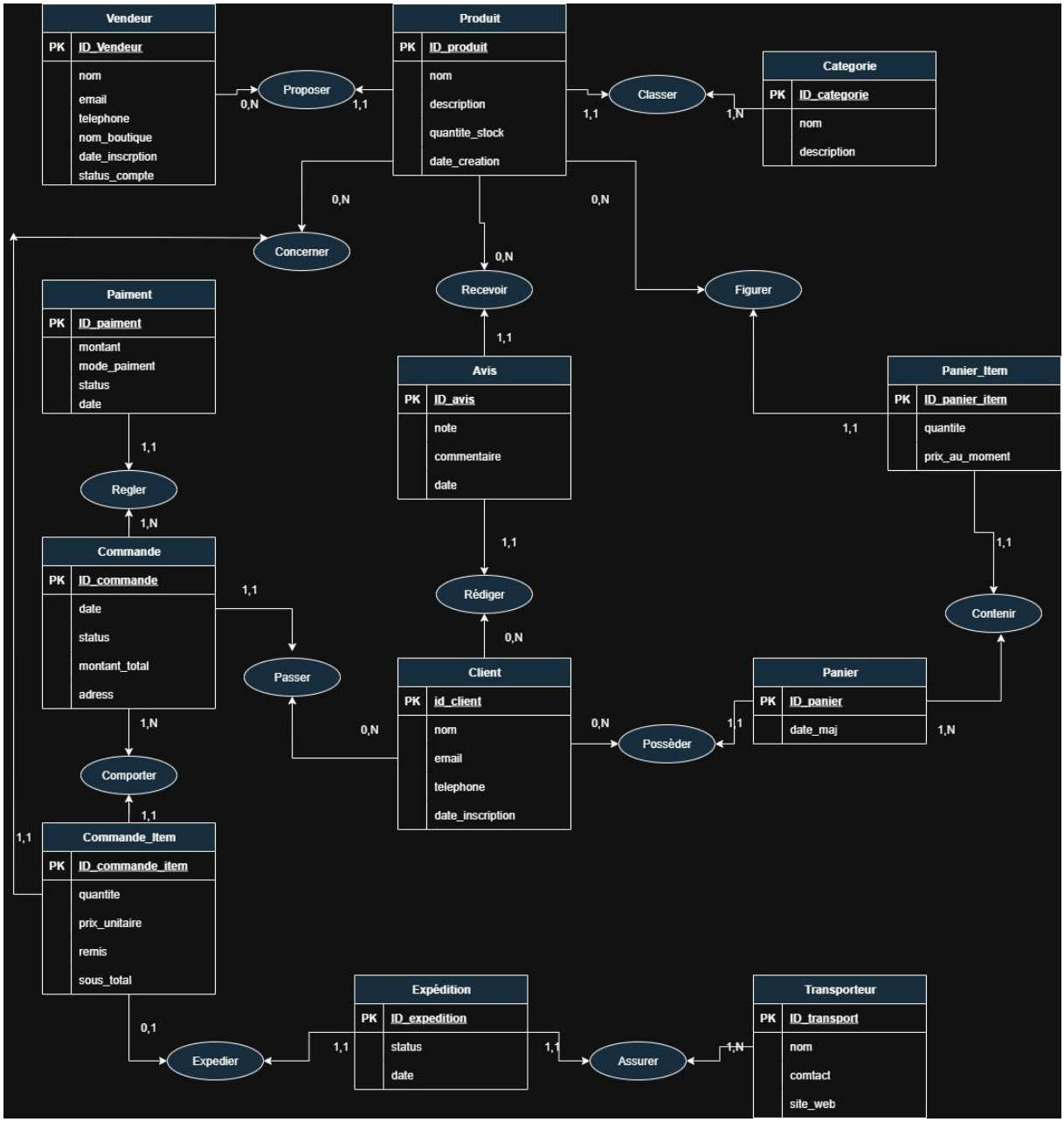
- Un client peut laisser plusieurs avis sur différents produits.
- Un avis est toujours lié à un seul produit et un seul client .

Cardinalités :

PRODUIT (1,1) ---> (0,N) AVIS

CLIENT (1,1) ---> (0,N) AVIS

2.2.3. MCD



2.3. Modèle Logique de Données (MLD)

Le schéma relationnel pour notre base de données Oracle sera le suivant (PK : Clé Primaire,

FK : Clé Étrangère) :

1. **VENDEUR**

VENDEUR (id_vendeur pk , nom, email, téléphone, nom_boutique, date_inscription)

- Clé etrangère : aucune
- Justification : chaque vendeur est un compte autonome, indépendant des autres entités. Il devient clé étrangère seulemnt dens PRODUIT .

2. **CLIENT**

CLIENT (id_client PK, nom, prénom, email, téléphone)

- Clé etrangère : aucune
- Justification : Chaque client est indépendant . Il devient clé étrangère dans PANIER, COMMANDE, et AVIS .

3. **CATEGORIE**

CATEGORIE (id_categorie PK, nom_categorie, description)

- Clé etrangère : aucune
- Justification : permet de créer une hiérarchie de catégories (sous-catégories) .

4. **PRODUIT**

PRODUIT (id_produit PK, nom_produit, description, prix, quantite_stock, #id_vendeur FK, #id_categorie FK)

- Clé etrangère : id_vendeur, id_categorie
- Justification : chaque produit appartient à un vendeur et une catégorie.

5. **PANIER**

PANIER (id_panier PK, date_creation, #id_client FK)

- Clé etrangère : id_client
- Justification : 1 client peut avoir plusieurs paniers.

6. **PANIER_ITEM**

PANIER_ITEM (#id_panier FK, #id_produit FK, quantite, prix_unitaire)

- Clé etrangère : id_panier, id_produit
- Justification : table associative entre PANIER et PRODUIT.

7. **COMMANDE**

COMMANDE (id_commande PK, date_commande, statut, #id_client FK)

- Clé etrangère : id_client
- Justification : relie chaque commande à un client et aux adresses concernées.

8. COMMANDE_ITEM

COMMANDE_ITEM (#id_commande FK, #id_produit FK, #id_vendeur FK, quantite, prix_unitaire)

- Clé etrangère : id_commande, id_produit, id_vendeur
- Justification : détaille chaque produit acheté dans une commande multi-vendeur.

9. PAIMENT

PAIMENT (id_paiement PK, montant, methode, statut, date_paiement, #id_commande FK)

- Clé etrangère : id_commande
- Justification : 1 commande → 0,N paiements possibles, permet de suivre l'état du paiement.

10. TRANSPORTEUR

TRANSPORTEUR(id_transporteur PK, nom_transporteur, telephone, email)

- Clé etrangère : aucune
- Justification : chaque transporteur est indépendant. Il devient clé étrangère uniquement dans EXPEDITION.

11. EXPEDITION

EXPEDITION(id_expedition PK, date_expedition, numero_suivi, statut_expedition, #id_transporteur FK, #id_commande_item FK)

- Clé etrangère : id_transporteur, id_commande_item
- Justification : relie chaque ligne de commande à un transporteur.

12. AVIS

AVIS (id_avis PK, note, commentaire, date_avis, #id_client FK, #id_produit FK)

- Clé etrangère : id_client, id_produit
- Justification : permet de stocker les évaluations des clients pour chaque produit.