République Algérienne Démocratique et Populaire

Ministère de l’Enseignement Supérieur et de la Recherche Scientifique

### École Supérieure en Science et Technologie de l’Informatique et du Numérique

### REALISATION ET CONCEPTION D’UN SITE WEB D’ANNONCESIMMOBILIERE

Réaliser par :

AIDDAOUI AMANI

BELAIOUAR ANIS

CHOUAKI INES

OUTERBAH MOHAMED

SEKIOU DOUA

**2024 /2025**

## INTRODUCTION GENERALE

Le secteur immobilier, pilier essentiel de l’économie moderne, a connu une transformation majeure avec l’avènement du numérique. Aujourd’hui, la recherche et la gestion des biens immobiliers ne se limitent plus aux méthodes traditionnelles, mais s’étendent à des plateformes en ligne dynamiques et accessibles. Cette révolution numérique a permis de transcender les frontières géographiques, offrant aux utilisateurs un accès instantané à une multitude de propriétés, tout en simplifiant les processus de transaction et de communication entre les différents acteurs du marché.

Dans ce contexte, la conception d’une plateforme immobilière en ligne comme DZ-Estate représente bien plus qu’un simple outil de mise en relation entre vendeurs et acheteurs. C’est une solution innovante qui répond aux besoins croissants de transparence, d’efficacité et de personnalisation dans le domaine immobilier. Que ce soit pour un agent immobilier cherchant à promouvoir ses listings, un particulier souhaitant vendre ou louer son bien, ou un acheteur à la recherche de la propriété idéale, DZ-Estate se positionne comme un acteur clé dans la modernisation du marché immobilier algérien.

L’immobilier, au-delà de son aspect transactionnel, revêt une dimension émotionnelle et stratégique. Chaque propriété représente un investissement, un projet de vie, ou une opportunité économique. La création d’une plateforme comme DZ-Estate va donc bien au-delà de la simple gestion d’annonces. Elle offre une expérience immersive, permettant aux utilisateurs d’explorer des biens selon leurs critères spécifiques, de visualiser des photos et des vidéos, et de localiser géographiquement chaque propriété. Cette approche enrichie transforme la recherche immobilière en une expérience intuitive et engageante.

L’attrait pour les plateformes immobilières en ligne réside également dans leur capacité à offrir une variété de fonctionnalités avancées, telles que la recherche filtrée, les alertes personnalisées, et une messagerie sécurisée pour faciliter les échanges entre les parties prenantes. Ces fonctionnalités, souvent absentes dans les méthodes traditionnelles, répondent aux attentes des utilisateurs modernes, qui recherchent rapidité, précision et sécurité.

Dans ce paysage numérique en constante évolution, la création d’une base de données robuste et flexible pour soutenir DZ-Estate devient un pilier fondamental. Cette base de données constitue l’infrastructure invisible mais essentielle qui sous-tend la gestion des annonces, la recherche avancée, la sécurisation des transactions, et la personnalisation de l’expérience utilisateur. Elle permet de stocker, organiser et restituer efficacement les informations relatives aux biens immobiliers, aux utilisateurs, et à leurs interactions.

Ainsi, ce rapport vise à explorer en détail la conception, le développement et l’implémentation de cette base de données, mettant en lumière son rôle central dans la réussite de DZ-Estate. En scrutant les mécanismes sous-jacents qui alimentent cette plateforme, nous chercherons à comprendre comment la technologie rencontre les besoins spécifiques du marché immobilier pour offrir une expérience utilisateur fluide, sécurisée et captivante. À travers cette analyse, nous démontrerons comment une base de données bien conçue peut devenir le socle d’une plateforme immobilière innovante, capable de répondre aux défis du marché actuel tout en anticipant les besoins futurs.

## Chapitre I

### PRESENTATION DU PROJET

### I.1 INTRODUCTION

Ce chapitre explore le choix du thème du site web DZ-Estate, une plateforme immobilière en ligne, et définit les objectifs de ce rapport. Nous examinerons les motivations derrière ce choix thématique, son importance dans le domaine de l’immobilier numérique, ainsi que les attentes spécifiques fixées pour ce document. Nous débuterons par une analyse des raisons ayant conduit à opter pour une plateforme dédiée à la gestion et à la recherche de biens immobiliers, mettant en lumière son importance stratégique dans le contexte actuel du marché immobilier en Algérie. En parallèle, nous définirons clairement les objectifs précis que ce rapport vise à réaliser pour une compréhension exhaustive du processus de conception et de mise en œuvre d’une base de données adaptée à cette plateforme.

Le choix de développer DZ-Estate s’inscrit dans une dynamique de modernisation du secteur immobilier, où la digitalisation des services devient une nécessité pour répondre aux attentes des utilisateurs. Cette plateforme vise à combler un manque évident dans le marché algérien, où les méthodes traditionnelles de recherche et de transaction immobilière restent souvent complexes, chronophages et peu transparentes. En offrant une solution en ligne centralisée, DZ-Estate se positionne comme un outil innovant pour simplifier les interactions entre les différents acteurs du marché : agents immobiliers, propriétaires, acheteurs et locataires.

L’objectif de ce rapport est de fournir une analyse détaillée des étapes de conception, de développement et d’implémentation de la base de données qui sous-tend DZ-Estate. Cette base de données joue un rôle central dans la gestion des annonces, la recherche avancée, la sécurisation des échanges et la personnalisation de l’expérience utilisateur. En explorant les spécifications techniques, les choix de modélisation et les fonctionnalités clés, ce document vise à démontrer comment une base de données bien structurée peut devenir le pilier d’une plateforme immobilière performante et évolutive.

À travers cette analyse, nous chercherons également à mettre en lumière les défis techniques et les solutions apportées pour garantir une expérience utilisateur fluide, tout en assurant la sécurité, la scalabilité et la performance du système. Ce rapport servira ainsi de référence pour comprendre comment la technologie peut être mise au service du secteur immobilier, en répondant aux besoins spécifiques du marché tout en anticipant les évolutions futures.

I.2 LE SCÉNARIO CHOISI ET SON IMPORTANCE

L’émergence et la croissance exponentielle des plateformes immobilières en ligne ont révolutionné la manière dont les biens immobiliers sont présentés, recherchés et gérés. Dans ce contexte dynamique, le choix de développer DZ-Estate, une plateforme dédiée à la gestion et à la recherche de propriétés en Algérie, s’avère être une stratégie fondamentale. Cette décision repose sur plusieurs éléments clés, qui soulignent l’importance et la pertinence de ce projet dans le paysage immobilier actuel.

1. Accès National et Marché en Pleine Expansion

Le choix d’une plateforme en ligne comme DZ-Estate s’explique par la portée nationale qu’offre Internet. Contrairement aux méthodes traditionnelles limitées par des contraintes géographiques, DZ-Estate permet d’atteindre un public large et diversifié à travers l’Algérie. Le marché immobilier en ligne est en pleine expansion, offrant ainsi une opportunité stratégique pour répondre aux besoins croissants des utilisateurs, qu’ils soient acheteurs, locataires, vendeurs ou agents immobiliers.

2. Variété de Biens et Personnalisation de l’Expérience Utilisateur

DZ-Estate offre une vitrine numérique pour exposer une variété étendue de biens immobiliers, allant des appartements modernes aux villas luxueuses, en passant par les terrains, les locaux commerciaux et les propriétés rurales. Cette diversité, difficile à présenter efficacement dans un contexte physique, devient accessible à un large public grâce à une interface intuitive et des fonctionnalités de recherche avancée. Les utilisateurs peuvent filtrer les annonces selon des critères spécifiques (prix, surface, localisation, type de bien, etc.), offrant ainsi une expérience personnalisée et adaptée à leurs besoins.

3. Expérience Immersive et Sécurisée

L’un des piliers de DZ-Estate est la création d’une expérience immersive et sécurisée pour ses utilisateurs. Grâce à des fonctionnalités telles que la géolocalisation, les galeries photo et vidéo, et les visites virtuelles, les utilisateurs peuvent explorer les propriétés en détail, comme s’ils y étaient. La plateforme intègre également une messagerie sécurisée pour faciliter les échanges entre acheteurs, vendeurs et agents, tout en garantissant la confidentialité des informations partagées.

4. Suivi des Tendances et Mise à Jour en Temps Réel

Le marché immobilier est en constante évolution, avec des tendances qui varient selon les régions, les types de biens et les préférences des utilisateurs. DZ-Estate permet de suivre ces tendances en temps réel et de mettre à jour les annonces facilement. Cette flexibilité est essentielle pour rester compétitif dans un secteur où les opportunités peuvent apparaître et disparaître rapidement. Les utilisateurs bénéficient ainsi d’informations actualisées et pertinentes.

5. Interaction Clientèle et Marketing Ciblé

DZ-Estate favorise une interaction directe et fluide entre les différents acteurs du marché immobilier. Les utilisateurs peuvent laisser des commentaires, poser des questions et recevoir des réponses rapides, créant ainsi un environnement de confiance et de transparence. De plus, la plateforme utilise des données analytiques pour proposer des annonces pertinentes et des offres spéciales, optimisant ainsi l’expérience utilisateur et augmentant les chances de transactions réussies.

6. Réduction des Coûts Opérationnels et Optimisation des Ressources

Un site web bien conçu comme DZ-Estate permet de réduire considérablement les coûts opérationnels liés à la gestion traditionnelle des biens immobiliers. Les agents et les propriétaires peuvent gérer leurs annonces en ligne, sans avoir besoin de supports physiques coûteux ou de déplacements fréquents. Cela permet de maintenir une présence constante sur le marché tout en optimisant les ressources et en maximisant la rentabilité.

7. Transparence et Confiance, Pilier du Succès

Dans un secteur où la confiance est primordiale, DZ-Estate s’engage à offrir une plateforme transparente et fiable. Les utilisateurs peuvent accéder à des informations détaillées sur chaque bien, vérifier la crédibilité des annonceurs grâce à des systèmes de notation et de commentaires, et suivre l’historique des transactions. Cette transparence renforce la confiance des utilisateurs et favorise des échanges plus sécurisés et équitables.

8. Modernisation du Secteur Immobilier Algérien

En Algérie, le secteur immobilier reste encore largement dominé par des méthodes traditionnelles, souvent complexes, chronophages et peu transparentes. DZ-Estate se positionne comme un acteur clé dans la modernisation de ce secteur, en offrant une solution numérique centralisée qui simplifie les processus de recherche, de gestion et de transaction. Cette innovation répond aux attentes des utilisateurs modernes, qui recherchent rapidité, précision et sécurité.

Conclusion

Le choix de développer DZ-Estate s’appuie sur la portée nationale du numérique, l’expérience utilisateur enrichissante qu’il offre, et les opportunités de croissance qu’il présente pour les professionnels et les particuliers. En combinant accessibilité, personnalisation, sécurité et innovation, DZ-Estate se positionne comme une solution clé pour moderniser le marché immobilier algérien. Cette plateforme ne se contente pas de répondre aux besoins actuels du marché ; elle anticipe également les évolutions futures, en s’adaptant aux nouvelles tendances et aux attentes des utilisateurs dans un monde de plus en plus connecté.

Ainsi, DZ-Estate incarne bien plus qu’une simple plateforme immobilière : c’est un outil stratégique qui transforme la manière dont les biens immobiliers sont recherchés, gérés et échangés, tout en renforçant la confiance et la transparence dans un secteur en pleine mutation.

I.3 L’OBJECTIF DU RAPPORT

L’objectif de ce rapport est de détailler la conception et la mise en œuvre de la base de données de DZ-Estate, une plateforme immobilière en ligne. Il se concentre sur les étapes clés de la création de la base de données, depuis la modélisation avec un diagramme entité-relation (ERD) jusqu’à l’implémentation des contraintes d’intégrité et des requêtes SQL. Ce document vise à fournir une analyse approfondie des exigences techniques, des relations entre les entités, et des outils utilisés pour garantir une base de données robuste, sécurisée et performante. Il servira de guide pratique pour les projets similaires dans le domaine de l’immobilier numérique.

# Chapitre II

## ANALYSE DES EXIGENCES

### II.1 INTRODUCTION

Ce chapitre se consacre à une analyse approfondie des exigences, tant fonctionnelles que non fonctionnelles, pour la base de données de DZ-Estate, une plateforme immobilière en ligne. Nous examinerons en détail les fonctionnalités attendues, les caractéristiques techniques et les contraintes indispensables pour garantir le bon fonctionnement, la performance et la fiabilité de cette application. Cette analyse vise à identifier les besoins des utilisateurs, les objectifs techniques et les critères de qualité essentiels à la conception d’une base de données robuste et adaptée aux spécificités du marché immobilier.

II.2 ANALYSE DES BESOINS FONCTIONNELS

La plateforme DZ-Estate est conçue pour répondre aux besoins fonctionnels des utilisateurs, en offrant des fonctionnalités essentielles pour une expérience utilisateur optimale. Les principales fonctionnalités requises sont les suivantes :

Fonctionnalités pour les Clients :

1**. Consultation des Annonces** :

Les clients peuvent parcourir l’ensemble des biens immobiliers disponibles sur la plateforme, afficher les détails des annonces (photos, descriptions, prix, localisation, etc.) et consulter les différentes catégories de propriétés (appartements, villas, terrains, etc.).

2. **Recherche Avancée et Filtres :**

Les utilisateurs peuvent effectuer des recherches avancées en filtrant les annonces par critères spécifiques (prix, surface, localisation, type de bien, etc.) pour trouver rapidement des propriétés correspondant à leurs besoins.

3. **Gestion des Favoris :**

Les clients peuvent marquer des annonces comme favoris pour les consulter ultérieurement, ou les retirer de leur liste de favoris. Cette fonctionnalité permet de personnaliser l’expérience de recherche.

4**. Prise de Contact :**

Les clients peuvent contacter les vendeurs ou les agents immobiliers via une messagerie sécurisée intégrée à la plateforme pour poser des questions ou demander des informations supplémentaires.

5. **Commentaires et Feedback :**

Les clients peuvent laisser des commentaires ou des avis sur les annonces, partageant ainsi leur expérience avec d’autres utilisateurs. Ces feedbacks contribuent à renforcer la transparence et la confiance sur la plateforme.

6**. Authentification et Sécurité :**

Les fonctionnalités telles que la prise de contact, l’ajout de commentaires ou la gestion des favoris ne sont accessibles qu’aux utilisateurs authentifiés, garantissant ainsi la sécurité des interactions.

Fonctionnalités pour l’Administrateur :

**1. Gestion des Annonces :**

L’administrateur peut ajouter de nouvelles annonces, les modifier ou les supprimer, assurant ainsi une gestion complète et dynamique du catalogue de biens immobiliers.

**2. Consultation des Demandes :**

L’administrateur a accès à la liste complète des demandes de contact ou des messages envoyés par les clients, permettant un suivi efficace des interactions et une réponse rapide aux questions.

**3. Intégration des Réseaux Sociaux :**

L’administrateur peut intégrer les liens vers les réseaux sociaux de l’entreprise pour favoriser l’interaction avec les clients et accroître la visibilité de la plateforme.

**4. Gestion des Comptes :**

L’administrateur peut modifier les informations personnelles de son compte (email, mot de passe, etc.) et gérer les accès des autres utilisateurs si nécessaire.

Remarque :

Les fonctionnalités telles que la prise de contact, l’ajout de commentaires ou la gestion des favoris ne sont accessibles qu’aux utilisateurs authentifiés. Cette restriction garantit la sécurité des données et la fiabilité des interactions sur la plateforme.

En résumé, DZ-Estate offre une gamme complète de fonctionnalités pour répondre aux besoins des clients et des administrateurs, tout en garantissant une expérience utilisateur fluide, sécurisée et personnalisée. Ces fonctionnalités sont essentielles pour assurer le succès de la plateforme dans un marché immobilier en pleine évolution.

II.3 ANALYSE DES BESOINS NON FONCTIONNELS

Les exigences non fonctionnelles sont des éléments essentiels qui influent sur la qualité, la performance et l’expérience globale de DZ-Estate, notre plateforme immobilière en ligne. Bien que ces exigences ne soient pas directement liées aux fonctionnalités de l’application, elles sont cruciales pour garantir un fonctionnement optimal, une utilisation aisée et une satisfaction maximale des utilisateurs. Voici les principales exigences non fonctionnelles identifiées pour DZ-Estate :

**. Performance**

- Objectif : Le site doit offrir une expérience rapide et réactive, avec des temps de chargement minimaux, même en période de trafic intense.

- Enjeu : Une performance optimale est essentielle pour maintenir l’engagement des utilisateurs et éviter les abandons dus à des temps d’attente excessifs.

**. Sécurité**

- Objectif : Garantir la sécurité des données des utilisateurs en protégeant les informations sensibles (mots de passe, coordonnées, etc.) et en prévenant les accès non autorisés.

- Enjeu : La sécurité est primordiale pour instaurer la confiance des utilisateurs et se conformer aux réglementations en matière de protection des données.

---

**. Fiabilité**

- Objectif : La base de données doit être fiable et stable, minimisant les interruptions de service et les pertes de données.

- Enjeu : Une plateforme fiable assure une disponibilité continue, ce qui est essentiel pour maintenir la crédibilité et la satisfaction des utilisateurs.

**. Convivialité**

- Objectif : L’interface utilisateur doit être intuitive, accessible et facile à utiliser pour un large éventail d’utilisateurs, y compris ceux ayant des compétences techniques limitées.

- Enjeu : Une interface conviviale améliore l’expérience utilisateur et encourage l’adoption de la plateforme.

**. Scalabilité**

- Objectif : La base de données doit être capable d’évoluer pour gérer efficacement un volume croissant de données et d’utilisateurs sans compromettre les performances.

- Enjeu : La scalabilité est essentielle pour anticiper la croissance de la plateforme et répondre aux besoins futurs.

**. Maintenance**

- Objectif : Le site doit être facile à maintenir et à mettre à jour, avec des processus clairs pour garantir sa stabilité et son efficacité continue.

- Enjeu : Une maintenance simplifiée réduit les coûts opérationnels et minimise les perturbations pour les utilisateurs.

**. Compatibilité**

- Objectif : La base de données doit être compatible avec les technologies et les outils utilisés dans le cadre du projet, assurant une intégration harmonieuse avec d’autres systèmes.

- Enjeu : La compatibilité garantit une accessibilité optimale pour tous les utilisateurs et facilite les évolutions futures de la plateforme.

**. Accessibilité**

- Objectif : La plateforme doit être accessible à tous les utilisateurs, y compris ceux ayant des besoins spécifiques (par exemple, des fonctionnalités adaptées aux personnes handicapées).

- Enjeu : L’accessibilité élargit la portée de la plateforme et renforce son inclusion sociale.

**Conclusion**

Les exigences non fonctionnelles de DZ-Estate jouent un rôle clé dans la réussite de la plateforme. Elles garantissent non seulement une expérience utilisateur fluide et sécurisée, mais aussi une infrastructure robuste et évolutive capable de s’adapter aux besoins futurs. En répondant à ces exigences, DZ-Estate se positionne comme une solution immobilière en ligne fiable, performante et adaptée aux attentes des utilisateurs modernes.

# Chapitre III

### CONCEPTION DE LA BASE DE DONNEES

### III.1 MODELE ENTITE-ASSOCIATION

III.1.1 Introduction

Le modèle entité-association (EA) est un outil de conception utilisé en ingénierie des systèmes d'information pour représenter de manière abstraite la structure des données d'un système, en modélisant les entités(objets ou concepts du monde réel), leurs attributs (propriétés descriptives) et les associations (relations entre entités). Il est souvent matérialisé sous forme de diagramme visuel, facilitant la communication entre les parties prenantes et servant de base à la conception de bases de données relationnelles. Son utilité réside dans sa capacité à structurer les données, éviter les redondances, clarifier les besoins et documenter la logique métier. Il est couramment utilisé dans des cas tels que la conception de bases de données, l'analyse des besoins, la migration de systèmes ou la gestion de projets complexes, comme dans un système de bibliothèque où l'on modélise les relations entre "Livre", "Auteur" et "Emprunteur".

III.1.2 Diagramme entité-association

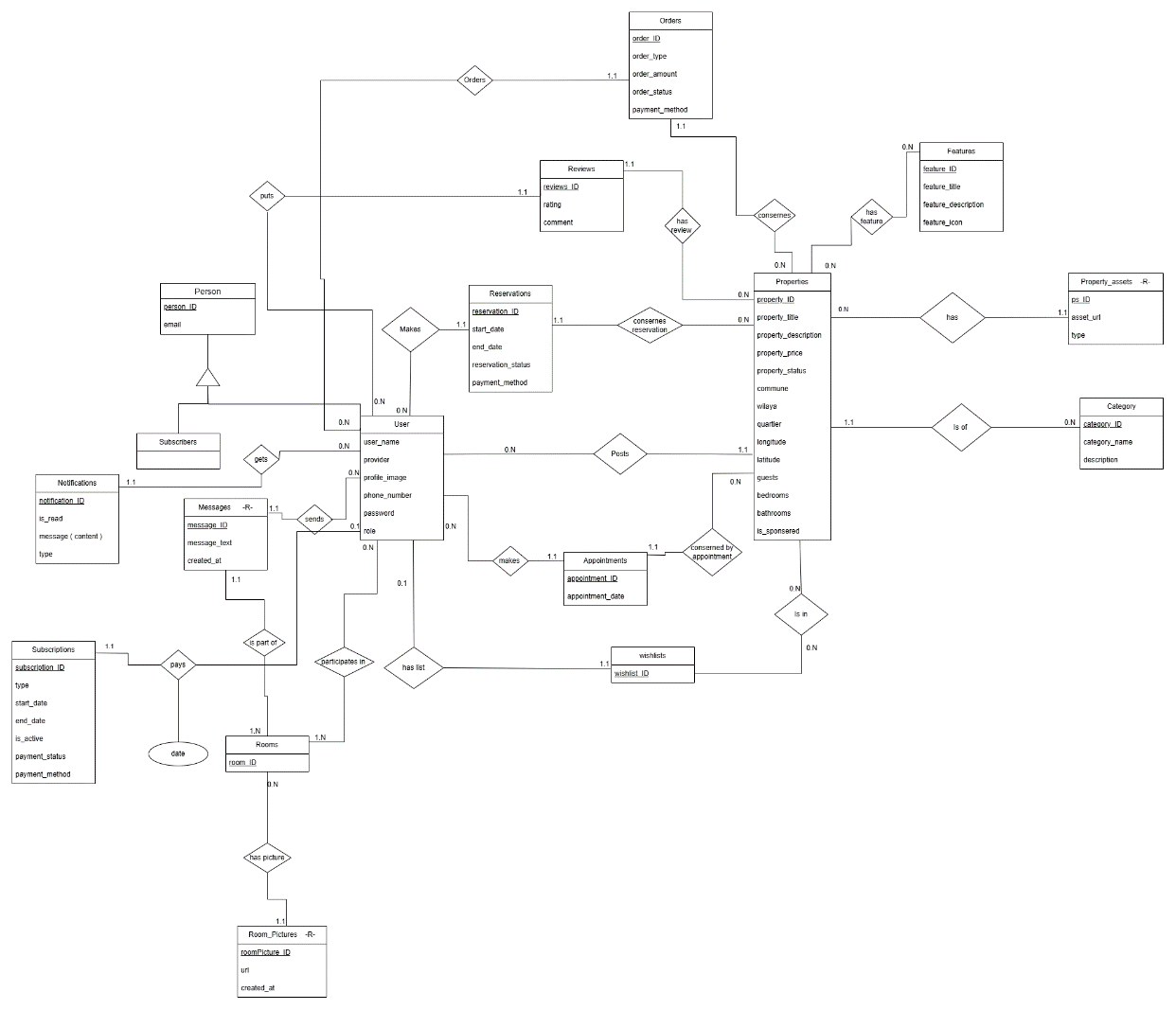


Figure III.1 – Diagramme entité-association

III.1.3 Description récapulative du modéle en générale :

La base de données définie dans ce fichier SQL est conçue pour gérer une plateforme immobilière, avec des tables dédiées à différentes fonctionnalités. La table **Users** stocke les informations des utilisateurs, y compris leurs rôles et méthodes d'authentification, tandis que **categories** organise les propriétés par type. Les **subscriptions** gèrent les abonnements des utilisateurs ce qui leur donne accès à plus fonctionnalités, et **properties** enregistre les annonces immobilières avec des détails tels que le prix, l'emplacement et les caractéristiques. Les **features** et **property\_features** permettent d'ajouter des commodités aux propriétés, et stocke les médias associés. Les **reviews** contiennent les avis des utilisateurs, la **wishlist** permet de sauvegarder les propriétés préférées. Les **reservations** et **orders** gèrent les transactions, tandis que **appointments** organise les visites. Les **notifications** informent les utilisateurs, et les tables **rooms**, **room\_pictures**, et **messages** facilitent la communication entre utilisateurs. Enfin, **subscribers** gère les abonnés sans compte . Ensemble, ces tables offrent une structure complète pour gérer les utilisateurs, les propriétés, les transactions et les interactions sur la plateforme.

III.1.4 Description des entités

D'après l'image du diagramme du le projet "DZ Estate", voici une description des :

Le fichier SQL définit une structure de base de données pour une plateforme immobilière, avec des tables conçues pour gérer efficacement les utilisateurs, les propriétés, les transactions et les interactions. Voici une description plus détaillée du rôle de chaque table :

**1. Users**

- Rôle: Stocke les informations des utilisateurs de la plateforme.

- Détails : Inclut des attributs comme l'identifiant unique (`id`), le nom d'utilisateur (`username`), l'email, le mot de passe (`password`), le rôle (`role` : admin ou user)

**2. Categories**

- Rôle : Organise les propriétés en catégories pour faciliter la navigation.

- Détails : Contient un identifiant unique (`id`), un nom de catégorie (`name`), et une description optionnelle (`description`). Cela permet de classer les propriétés par type (exemple : appartements, villas).

3. **Subscriptions**

- Rôle: Gère les abonnements des utilisateurs à des services premium ou gratuits.

- Détails : Inclut son identifiant , le type d'abonnement (`type` : free ou premium), les dates de début et de fin (`start\_date`, `end\_date`), et le statut de paiement (`payment\_status` : pending, confirmed, canceled).

4. **Properties**

- Rôle: Enregistre les propriétés disponibles à la vente ou à la location.

- Détails : Contient des informations détaillées comme le titre ,la description (`description`), le prix (`price`), le type de transaction (`status` : sell ou rent), l'emplacement (`commune`, `quartier`, `wilaya`), les coordonnées géographiques (`longitude`, `latitude`), et les caractéristiques du logement (`guests`, `bedrooms`, `bathrooms`).

5. **Features**

- Rôle: Liste les caractéristiques ou commodités disponibles pour les propriétés.

- Détails : Contient un identifiant unique (`id`), un titre (`title`), une description (`description`).

6. **Property\_Assets**

- Rôle : Stocke les médias associés aux propriétés, comme les images ou les vidéos.

- Détails : Contient un identifiant unique (`id`), l'URL du média (`asset\_url`), et le type de média (`type` : image ou video).

7. **Reviews**

- Rôle: Contient les avis et les évaluations des utilisateurs sur les propriétés.

- Détails : Inclut un idantifiant unique (‘id ‘) ,une note (`rating`), un commentaire (`comment`), et la date de création.

8. **Wishlists**

- Rôle : Permet aux utilisateurs de sauvegarder leurs propriétés préférées.

- Détails : Caracterisée par son idantifiant (‘id’).

9. **Réservations**

- Rôle: Gère les réservations de propriétés par les utilisateurs.

- Détails: Contient son l'identifiant , les dates de début et de fin (`start\_date`, `end\_date`), le statut de paiement (`status`), et la méthode de paiement (`payment\_method`).

10**. Appointments**

- Rôle : Enregistre les rendez-vous pour visiter les propriétés.

- Détails : Contient un identifiant et la date du rendez-vous (`date`).

11. **Orders**

- Rôle: Gère les commandes de location ou d'achat de propriétés.

- Détails : Contient un identifiant, le type de commande (`order\_type` : rent ou purchase), le montant (`amount`), le statut de la commande (`status`), et la méthode de paiement (`payment\_method`).

12. **Subscribers**

- Rôle: Stocke les emails des utilisateurs sans compte abonnés aux notifications .

- Détails : Contient un identifiant unique (`id`), l'email de l'utilisateur (`email`), et la date d'abonnement (`subscribed\_at`).

13. **Notifications**

- Rôle: Envoie des notifications aux utilisateurs.

- Détails: Inclue un identifiant, et un indicateur de lecture (`is\_read`).

14. **Rooms**

- Rôle : Crée des salles de discussion entre deux utilisateurs.

- Détails: Contient un identifiant unique et une date de création(‘created\_at)’.

15. **Room\_pictures**

- Rôle : Stocke les images associées aux salles de discussion.

- Détails : Contient un identifiant et l'URL de l'image (`url`).

16. **Messages**

- Rôle : Enregistre les messages échangés dans les salles de discussion.

- Détails : Inclut un identifiant , le contenu du message (`message`), et la date de création (`created\_at`).

Conclusion

Ces entités permettent de gérer efficacement les utilisateurs, les propriétés, les transactions et les interactions sur la plateforme immobilière. Chaque table est conçue pour répondre à un besoin spécifique, assurant une gestion structurée et cohérente des données.

### III.2 MODELE RELATIONNELLE

**Le modèle de données relationnel**, qui organise les informations sous forme de tables ou relations, est l’un des modèles les plus importants et les plus utilisés en informatique. Sa simplicité et sa structure intuitive en font un outil essentiel pour la gestion des données de manière organisée et facilement accessible, ce qui permet une manipulation efficace via des opérations telles que les requêtes, les jointures et les filtres. **Les Systèmes de Gestion de Bases de Données** (SGBD) relationnels, basés sur ce modèle, sont largement répandus car ils offrent une grande flexibilité, une intégrité des données et une capacité à gérer des volumes importants d’informations. Le modèle relationnel est donc fondamental pour la conception de bases de données robustes, scalables et adaptées à des applications variées, allant des systèmes transactionnels aux analyses complexes. Son utilité réside dans sa capacité à simplifier la gestion des données tout en garantissant leur cohérence et leur sécurité.

Les règles de transformation du modèle entité-association au modèle relationnel sont les suivantes :

→ Une entité devient une relation.

→ Une propriété devient un attribut.

→ Un identifiant devient une clé primaire (souligné).

→ une association devient une relation, si les cardinalités maximales sont n.n .

→ Une association transmet l’identifiant de l’entité ayant pour cardinalité maximale n vers l’entité ayant la cardinalité maximale 1 (l’identifiant est ainsi précédé d’un #).

→ Une entité marquée -R- est faible et a comme clé primaire son identifiant et l’identifiant de l’entité forte (de laquelle elle dépend)

Pour notre modèle, on aura :

**Remarque :** Pour les entités Person, Users et Subscribers on traite l’héritage avec la méthode push down puisque l’entité parent ‘Person’ n’a que deux attributs et que l’entité Users a beaucoup d’associations, ce qui donne les tables suivantes :

1. **Users** ( user\_ID , user\_name , email\_client , user\_phone, provider, password\_client, user rôle, user\_profileImage)
2. **Subscribers** (id\_subscriber , email , date )
3. **Categories** ( categorie\_ID , name , description )
   1. **Remarque :** Pour la table Subscriptions nous allons utiliser l’identifiant de l’utilisateur et la date de payement comme clés primaires pour des raisons d’optimisation et pour faciliter la recherche , vu que un abonnement n’appartient qu’à un seul et unique user et chaque renouvellement crée une nouvelle instance de la table ; donc ces deux attribues sont suffisants pour identifier les tuples de la table.
4. **Subscriptions** (#user\_id , updated\_at, type , start\_date , end\_date , is\_active , payment\_status , payment\_method , created\_at)
5. **Property** (id\_Propriété, Propriété\_title , Propriété\_description , Propriété\_price
   1. , Propriété\_status, Propriété\_loscation , commune, wilaya, quartier ,longitude,numguests,bedrooms ,bathrooms, is\_sponsered ,#category\_id, #Owner\_id )
6. **Features** (feature ID , title , description )
7. **Property \_ Features** (#proprety\_id , #feature\_id)
   1. **Remarque :** L’entité propertyAsset est faible et depend de l’entité Property donc lors du passage au modèle relationnel la table Property\_assets prend son identifiant et celui de la table Property comme cles primaires
8. **Proprety\_assets** (propertyAsset\_id,#proprety\_id, asset\_url, type)
9. **Reviews** ( id\_Review , #id\_user , rating , comment , date, #id\_propriété )
   1. **Remarque :** pour l’entité Wishlists on obtient la table :
      1. Wishlists ( wishlists\_ID , #user\_ID , #property\_ID)
      2. Or dans ce cas les clés étrangères sont suffisantes pour identifier une instance de la table , donc pour une meilleure optimisation de l’espace mémoire on préfère ne pas utiliser l’attribut wishlists\_ID ce qui donne la table suivante :
10. **Wishlists** (#user\_id, #property\_id )
11. **Reservation**( id\_reseration,#id\_user ,#id\_proprety, start\_date ,end\_date ,status\_payment )
12. **Appointment** ( appointement\_id,#user\_id , #id\_proprety , date )
13. **Rooms** (id\_room , #id\_user1 ,#id\_user2 )
    1. **Remarque :** L’entité Messages est faible et dépend de l’entité Rooms donc lors du passage au modèle relationnel la table Messages prend son identifiant et celui de la table Rooms comme clés primaires
14. **Messages** (id\_message , #id\_room ,#id\_user ,message )
    1. **Remarque :** L’entité RoomPictures est faible et dépend de l’entité Rooms donc lors du passage au modèle relationnel la table RoomPictures prend son identifiant et celui de la table Rooms comme clés primaires
15. **RoomPictures** (id\_picture ,#id\_room , picture , created\_at)
16. **Notifications** (id\_notification , # id\_user ,is\_read , created\_at)
17. **Orders** ( id\_order ,# id\_user ,# id\_propriété , type , amount , status , payment\_method , created\_date)

Chapitre IV

## DICTIONNAIRE DE DONNEES

### IV.1 INTRODUCTION

Ce chapitre explore en détail la structure de la plateforme **DZ Estate**, en présentant chaque table de la base de données, ses attributs spécifiques et les relations entre les entités. Il explique comment les données sont organisées pour gérer les utilisateurs, les logements, les réservations, les avis et d'autres éléments essentiels, assurant une expérience utilisateur fluide et efficace

### IV.2 DESCRIPTION DES TABLES

Dans ce qui suit nous allons fournir une description qui offre une vue détaillée de la structure de la base de données utilisée pour la plateforme de vente de vente et de reservation de biens immobiliers.

Chaque table de la base de données est présentée avec une description détaillée de ses attributs, de leur signification et de leur utilité dans le contexte de l’application. Cette section permet de mieux appréhender les entités clés, leurs relations, ainsi que les éléments essentiels qui constituent le fondement de la plateforme. En comprenant la composition et la signification de chaque table, il devient possible d’appréhender plus clairement la logique sous-jacente à la gestion des données au sein du site web de vente de bijoux.

|  |  |  |
| --- | --- | --- |
| Table | Description | Clé primaire/étrangère |
| Users | Table contenant les informations des utilisateurs inscrits sur la plateforme de DZ Estate. Elle comprend des données telles que le nom, le prénom, l’adresse, le numéro de téléphone, l’email et le mot de passe du user. | Primary Key : id\_user |
| Propreties | Table stockant les détails des maison disponibles à la vente. Elle contient des informations telles que le nom, la description, l’adresse , la wilaya , la ville ,le prix et d’autre champs permettant a l’utilisateur de bien faire son choix | Primary Key : id\_Propriété  Foreign Key : id\_owner  Id\_category |
| Categories | La table **catégories** organise les propriétés en catégories pour faciliter la navigation et la recherche. | Primary Key : id\_category |
| Subscriptions | La table **subscriptions** gère les abonnements des utilisateurs à des services premium ou gratuit . | Primary Key : id\_user , updated\_at  Foreign Key : id\_user |
| Features | La table **features**liste les caractéristiques ou commodités disponibles pour les propriétés, comme une piscine, un garage, ou un jardin. | Primary Key : id\_feature |
| Property\_feature | La table associe les caractéristiques au maison en listant les features qui y sont disponibles. | Primary Key : ( Id\_property , Id\_feature )  Foreign Key : Id\_property , Id\_feature |
| Property-asstes | La table **property\_assets** stocke les médias associés aux propriétés, comme les images ou les vidéos | Primary Key : ( id\_asset , property\_id)  Foreign Key : property\_id |
| Reveiws | Table contenant les avis et commentaire des utilisateurs vis-à-vis de leur expérience . | Primary Key : id\_Review  Foreigh Key : user\_id , property\_id |
| Wishlists | Table recensant les différents maison marquées comme favoris par un utilisateur | Primary Key : ( id\_User , Id\_Propriété )  Foreign Key : Id\_propriété , User\_id |
| Rooms | Table stockant les différentes discutions entre les utilisateurs. | Primary Key : id\_rooms  Foreign key : Id\_user1 , Id\_user2 |
| Room-pictures | La table **room\_pictures** stocke les images associées aux salles de discussion | Primary Key : ( roomPicture\_id , room\_id )  Foreign Key : room\_id |
| Messages | Table des messages d’une discution entre deux utilisateurs. | Primary Key : (id\_message, id\_room)  Foreign key : id\_room ,id\_user |
| Réservations | Table stockant les réservations d’une maison. | Primary Key : id\_reservation  Foreign Key : id\_user , id\_property |
| Appointments | La table**appointments** enregistre les rendez-vous pour visiter les propriétés. | Primary Key : id\_appointemnt  Foreign Key : id\_user , property\_id |
| Orders | Table reliant les maison et les utilisateurs par les offres faites | Primary Key : id\_order  Foreign Key : Id\_user , Id\_propriété |
| Subscribers | La table **subscribers** stocke les emails des utilisateurs sans compte abonnés aux notifications | Primary Key : id\_subscriber |
| Notification | La table **notifications** envoie des notifications aux utilisateurs. | Primary Key : id\_notification  Foreign Key : id\_user |

Table IV.1 – DESCRIPTION DES TABLES

### IV.3 DESCRIPTION DES ATTRIBUTS ET LEURS TYPES

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| id\_user | INT | Identifiant unique du user |
| nom\_user | VARCHAR  VARCHAR  VARCHAR  VARCHAR  TEXT | Nom d’utilisateur  Adresse email du user  Rôle de l’utilisateur  Le numéro de téléphone de l’utilisateur  Mot de passe de l’utilisateur |
| email\_user |
| rôle  Phone number  password\_user |
| provider | User\_provider | La manière avec laquelle l’utilisateur s’authentifie |
| Profile\_image | TEXT | URL de la photo de profile de l’utilisateur |

Table IV.2 – Data Dictionary - Table User

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| Id\_category  Category\_name  Category\_description | INT  VARCHAR  TEXT | Identifiant unique  Nom de la categorie  Une description de la catégorie |

Table IV.3 – Data Dictionary - Table Categories

|  |  |  |
| --- | --- | --- |
| Attribut | Type de Données | Description |
| User\_id | INT | Identifiant unique de l’utilisateur associé |
| Updated\_at | TIMESTAMP | La date du payement |
| Type | Subscription\_type | Décris si l’abonnement est payant ou gratuit |
| Start\_date | DATE | Représente la date du début de l’abonnement |
| End\_date | DATE | Date de la fin de l’abonnement |
| Is\_active | BOOLEAN | Décris si l’abonnement est actif ou pas |
| Payment\_status | Payment\_status | Le status du payement ( revlé / non reglé) |
| Payment\_method | Payment\_method | La methode du payement |

Table IV.4 – Data Dictionary - Table Subscriptions

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| id\_propiété | INT | Identifiant unique de la maison |
| title | VARCHAR | Nom de la propriété |
| description\_propriété | Text | Description de la propriété |
| price | DECIMAL | Le prix de la propriété |
| status | Property\_sell\_type (sell / rent) | Caractéristique de la maison : à vendre ou à louer |
| id\_owner | INT | ID de l’utilisateur(agent / particulier ) associé à l’article |
| commune | VARCHAR | La commune ou se trouve la maison |
| quartier | VARCHAR | Le quartier de la maison |
| wilaya | INT | La wilaya ou se trouve la maison |
| longitude | FLOAT | Localisation sur Google maps en longueur |
| latitude | FLOAT | Localisation sur Google maps en larguer |
| guests | INT | Nombre de personne qui peuvent heberger dans la maison |
| bedrooms | INT | Nombre de chambres dans la maison |
| Is\_sponsered | BOOLEAN | Precise si le post est sponsorisé |
| Category\_id | INT | ID de la category à la quelle la maison appartient |

Table IV.5 – Data Dictionary - Table Property

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| id\_feature | INT | Identifiant unique de la caracteristique |
| Title\_feature | VARCHAR | Nom de la caractéristique |
| Description | TEXT | Déscription de la caracteristique |
| icon | TEXT | Image de la caractéristique |

Table IV.6 – Data Dictionary - Table Features

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| Property\_id | INT | Identifiant unique de la maison associée |
| Feature\_id | VINT | Identifiant unique de la particularité associée |

Table IV.7 – Data Dictionary - Table Property\_features

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| Id\_asset | INT | Identifiant unique |
| Property\_id | INT | Identifiant unique de la maison associée |
| Asset\_url | TEXT | L’url de l’image de maison |
| type | Property\_asset\_type | Type de de la ressource (vidéo / image) |

Table IV.8 – Data Dictionary - Table Property\_assets

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| id\_review | INT | Identifiant unique de la critique |
| Property\_id | INT | Identifiant unique de la maison associée |
| User\_id | INT | Identifiant unique de l’utilisateur associée |
| rating | INT | Note donnée par l’utilisateur à la propriété |
| comment | TEXT | Commentaire laissé par l’utilisateur sur la maison |
| date | DATE | Date de l’ajout de la critique |

Table IV.9 – Data Dictionary - Table Reviews

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| id\_wishlist | INT | Identifiant de la liste |
| Property\_id | INT | Identifiant unique de la maison associée |

Table IV.10 – Data Dictionary - Table Whishlists

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| Reservation\_ID | INT | Identifiant unique de la reservation |
| User\_ID | INT | Identifiant de l’utilisateur associé |
| Property\_ID | INT | Indentifiant de la propriété associée |
| Start\_date | Date | Date du début de la réservation |
| End\_date | Date | Date de la fin de la reservation |
| Reservation\_status | Payement\_status | Payée ou non payée |
| Payement\_method | Payement\_methd | La méthode de payement |

Table IV.11 – Data Dictionary - Table Favoris

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| Appointement\_ID | INT | Identifiant unique du rendez-vous |
| User\_ID | INT | Identifiant de l’utilisateur associé |
| Property\_ID | INT | Identifiant de la maison associée |
| date | DATE | Date du rendez-vous |

Table IV.12 – Data Dictionary - Table Appointments

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| Order\_ID | INT | Identifiant unique de l’offre |
| User\_ID | INT | Identifiant de l’utilisateur associé |
| Property\_ID | INT | Identifiant de la maison associée |
| Order\_type | Order\_type | Type de l’offre ( sell / rent ) |
| amount | Descimal | Le montant de l’offre |
| Order\_status | Order\_status | L’etat de l’offre ( pending / paid / canceled ) |
| Payement\_method | Payment\_method | La method du payement |

Table IV.13 – Data Dictionary - Table Order

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| Subscribers\_ID | INT | Identifiant unique du subscriber |
| Subscriber\_email | VARCHAR | Email de l’utilisateur |

Table IV.14 – Data Dictionary - Table Subscribers

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| notification\_ID | INT | Identifiant unique de la notification |
| User\_ID | INT | L’identifiant de l’utilisateur qui reçoit la notification |
| message | TEXT | Le contenue de la notification |
| Is\_read | BOOLEAN | Indique si la notification est lue ou pas |

Table IV.15 – Data Dictionary – Table Notifications

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| Room\_ID | INT | Identifiant unique de la conversation |
| User1\_ID | INT | L’identifiant de l’un des deux utilisateurs dans la conversation |
| User2\_ID | INT | L’identifiant de l’autre utilisateur dans la conversation |

Table IV.16 – Data Dictionary – Table Rooms

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| Room\_picture\_ID | INT | Identifiant unique de la photo |
| Room\_ID | INT | L’identifiant de la conversation |
| url | TEXT | Liens de la photo |
| Created\_at | TIMESTAMP | Date et heure d’envoie |

Table IV.17 – Data Dictionary – Table Room\_Pictures

|  |  |  |
| --- | --- | --- |
| Attribut | Type de données | Description |
| Message\_ID | INT | Identifiant unique du message |
| Room\_ID | INT | L’identifiant de la conversation |
| Sender\_ID | INT | L’identifiant de l’utilisateur qui envoie |
| message | TEXT | Contenue du message |
| Created\_at | TIMESTAMP | Date et heure d’envoie |

Table IV.18 – Data Dictionary – Table Messages

# Chapitre V

## LES CONTRAINTES D’INTEGRITE

### V.1 INTRODUCTION

Les contraintes d’intégrité jouent un rôle essentiel dans le maintien de la précision et de la cohérence des données au sein de notre base de données. Notre schéma de base de données est structuré autour de plusieurs de ces contraintes, chacune ayant une fonction spécifique pour assurer la qualité des informations stockées. Ces règles sont mises en place pour garantir que seules des données valides et pertinentes sont enregistrées, évitant ainsi les incohérences ou les erreurs potentielles. Cette section détaille les différentes contraintes d’intégrité appliquées dans notre schéma, en soulignant leur importance cruciale pour préserver l’exactitude et la fiabilité des données.

### V.2 **Contraintes de clé primaire (PRIMARY KEY)**

* **Clé primaire** : Une contrainte de clé primaire garantit que chaque enregistrement dans une table est unique et non nul. Elle est définie sur les colonnes suivantes :
  + users(id)
  + categories(id)
  + properties(id)
  + features(id)
  + reviews(id)
  + reservations(id)
  + appointments(id)
  + orders(id)
  + subscribers(id)
  + notifications(id)
  + rooms(id)
* **Clé primaire composite** : Certaines tables utilisent des clés primaires composites pour garantir l'unicité sur plusieurs colonnes :
  + subscriptions(user\_id, updated\_at)
  + property\_features(property\_id, feature\_id)
  + wishlists(user\_id, property\_id)
  + property\_assets(property\_asset\_ID, property\_id)
  + room\_pictures(picture\_ID, room\_id)
  + messages(message\_ID, room\_id)

### **V.3Contraintes de clé étrangère (FOREIGN KEY)**

* **Clé étrangère** : Une contrainte de clé étrangère garantit que les valeurs d'une colonne (ou d'un ensemble de colonnes) correspondent à une valeur existante dans une autre table. Cela maintient l'intégrité référentielle entre les tables. Les clés étrangères sont définies comme suit :
  + subscriptions(user\_id) référence users(id)
  + properties(owner\_id) référence users(id)
  + properties(category\_id) référence categories(id)
  + property\_features(property\_id) référence properties(id)
  + property\_features(feature\_id) référence features(id)
  + property\_assets(property\_id) référence properties(id)
  + reviews(property\_id) référence properties(id)
  + reviews(user\_id) référence users(id)
  + wishlists(user\_id) référence users(id)
  + wishlists(property\_id) référence properties(id)
  + reservations(user\_id) référence users(id)
  + reservations(property\_id) référence properties(id)
  + appointments(user\_id) référence users(id)
  + appointments(property\_id) référence properties(id)
  + orders(user\_id) référence users(id)
  + orders(property\_id) référence properties(id)
  + notifications(user\_id) référence users(id)
  + rooms(user1\_id) référence users(id)
  + rooms(user2\_id) référence users(id)
  + room\_pictures(room\_id) référence rooms(id)
  + messages(room\_id) référence rooms(id)
  + messages(sender\_id) référence users(id)
* **Actions sur suppression (ON DELETE)** :
  + **CASCADE** : Supprime les enregistrements liés lorsque l'enregistrement parent est supprimé .

Exemple :

-- Subscriptions Table

    user\_id INT REFERENCES users(id) ON DELETE CASCADE,

-- Property Features Table (Many-to-Many Relationship)

    property\_id INT REFERENCES properties(id) ON DELETE CASCADE,

    feature\_id INT REFERENCES features(id) ON DELETE CASCADE,

* + **SET NULL** : Met à jour la clé étrangère à NULL lorsque l'enregistrement parent est supprimé .

Exemple :

-- Rooms Table

   user1\_id INT REFERENCES users(id) ON DELETE SET NULL,

   user2\_id INT REFERENCES users(id) ON DELETE SET NULL,

-- Messages Table

    sender\_id INT REFERENCES users(id) ON DELETE SET NULL,

### V.4. **Contraintes d'unicité (UNIQUE)**

* **Contrainte d'unicité** : Garantit que les valeurs dans une colonne (ou un ensemble de colonnes) sont uniques dans la table.
  + users(email) : L'adresse e-mail doit être unique.

email VARCHAR(100) UNIQUE NOT NULL,

* + categories(name) : Le nom de la catégorie doit être unique.

 name VARCHAR(100) NOT NULL UNIQUE,

* + features(title) : Le titre de la fonctionnalité doit être unique.

 title VARCHAR(255) NOT NULL UNIQUE,

* + subscribers(email) : L'adresse e-mail de l'abonné doit être unique

  email VARCHAR(100) UNIQUE NOT NULL,

### V.5 **Contraintes de vérification (CHECK)**

* **Contrainte de vérification** : Garantit que les valeurs d'une colonne respectent une condition spécifique.
  + properties(price) : Le prix doit être supérieur à 0.

 price DECIMAL(15, 2) NOT NULL CHECK (price > 0),

* + properties(wilaya) : La wilaya doit être comprise entre 1 et 58.

 wilaya INT CHECK (wilaya BETWEEN 1 AND 58),

* + properties(guests) : Le nombre d'invités doit être supérieur à 0.

guests INT NOT NULL CHECK (guests > 0),

* + properties(bedrooms) : Le nombre de chambres doit être supérieur à 0.
  + properties(bathrooms) : Le nombre de salles de bain doit être supérieur à 0.

bedrooms INT NOT NULL CHECK (bedrooms > 0),

bathrooms INT NOT NULL CHECK (bathrooms > 0),

* + reviews(rating) : La note doit être comprise entre 1 et 5.

rating INT CHECK (rating >= 1 AND rating <= 5),

* + orders(amount) : Le montant doit être supérieur à 0

amount DECIMAL(15, 2) NOT NULL CHECK (amount > 0),

### V.6. **Contraintes de type (ENUM)**

* **Types ENUM** : Les colonnes de type ENUM limitent les valeurs possibles à un ensemble prédéfini.
  + user\_role : Les rôles possibles sont admin ou user.

CREATE TYPE user\_role AS ENUM ('admin', 'user');

role user\_role DEFAULT 'user',

* + user\_provider : Les fournisseurs possibles sont credentials ou google.

CREATE TYPE user\_provider AS ENUM ('credentials', 'google');

provider user\_provider DEFAULT 'credentials',

* + subscription\_type : Les types d'abonnement possibles sont free ou premium.

CREATE TYPE subscription\_type AS ENUM ('free', 'premium');

type subscription\_type NOT NULL DEFAULT 'free',

* + property\_sell\_type : Les types de vente possibles sont sell ou rent.

CREATE TYPE property\_sell\_type AS ENUM ('sell', 'rent');

status property\_sell\_type NOT NULL,

* + payment\_status : Les statuts de paiement possibles sont pending, confirmed ou canceled.

CREATE TYPE payment\_status AS ENUM ('pending', 'confirmed', 'canceled');

payment\_status payment\_status DEFAULT 'pending',

* + payment\_method : Les méthodes de paiement possibles sont stripe ou chargily.

CREATE TYPE payment\_method AS ENUM ('stripe', 'chargily');

--Subscription Table

payment\_method payment\_method NOT NULL,

* + order\_type : Les types de commande possibles sont rent ou purchase.

CREATE TYPE order\_type AS ENUM ('rent', 'purchase');

--Orders Table

order\_type order\_type NOT NULL,

* + order\_status : Les statuts de commande possibles sont pending, paid ou canceled.

CREATE TYPE order\_status AS ENUM ('pending', 'paid', 'canceled');

--Orders Table

status order\_status DEFAULT 'pending',

* + property\_asset\_type : Les types d'actifs possibles sont image ou video.

CREATE TYPE property\_asset\_type AS ENUM ('image', 'video');

--Property\_assets Table

type property\_asset\_type DEFAULT 'image',

### V.7. **Index de performance**

* **Index** : Les index sont créés pour améliorer les performances des requêtes sur certaines colonnes fréquemment interrogées en permettant de localiser rapidement les informations sans avoir à parcourir l'intégralité de la table
  + idx\_users\_email : Index sur users(email).

CREATE INDEX idx\_users\_email ON users(email);

* + idx\_subscriptions\_user\_active : Index sur subscriptions(user\_id, is\_active).

CREATE INDEX idx\_subscriptions\_user\_active ON subscriptions(user\_id, is\_active);

* + idx\_properties\_owner\_sponsored : Index sur properties(owner\_id, is\_sponsored).

CREATE INDEX idx\_properties\_owner\_sponsored ON properties(owner\_id, is\_sponsored);

* + idx\_reservations\_user\_property : Index sur reservations(user\_id, property\_id).

CREATE INDEX idx\_reservations\_user\_property ON reservations(user\_id, property\_id);

* + idx\_orders\_user\_property : Index sur orders(user\_id, property\_id).

CREATE INDEX idx\_orders\_user\_property ON orders(user\_id, property\_id);

Chapitre VI

## REQUETES SQL ET EXEMPLES

### VI.1 INTRODUCTION

La section "Requêtes SQL et Exemples" met en lumière l’utilisation pratique de requêtes SQL pour interagir avec la base de données dédiée à notre plateforme **DZ Estate**, spécialisée dans la gestion immobilière. Elle propose une série d’exemples concrets illustrant l’utilisation des requêtes pour des opérations telles que la récupération, l’insertion, la modification et la suppression de données liées aux propriétés, aux clients et aux transactions immobilières. Chaque requête est accompagnée d’une explication détaillée décrivant son objectif, son utilité dans le contexte de l’application, ainsi que sa contribution à la fonctionnalité globale du système. Cette section vise à offrir une compréhension pratique des interactions SQL nécessaires pour gérer efficacement les données au sein de **DZ Estate**, en mettant l’accent sur l’optimisation des processus immobiliers et l’amélioration de l’expérience utilisateur.

### VI.2 TYPES DE REQUETES SQL

VI.2.1 Insertion des données :

→ Requête SQL :

// Insert the message into the database

      "INSERT INTO messages (chat\_id, user\_id, message) VALUES ($1, $2, $3) RETURNING \*",

// Insertion d’un nouvel utilisateur

"INSERT INTO users (name, email, password) VALUES ($1, $2, $3) RETURNING \*"

VI.2.2 Mise à jour :

// mise à jour d’un tuple dans la table user

"UPDATE users SET name = $1, email = $2, password = $3 WHERE id = $4 RETURNING \*"

// mise à jour de l’état d’une réservation

UPDATE reservations SET status = $1 WHERE id = $2

VI.2.3 Suppression:

// Suppression d'un utilisateur

DELETE FROM users WHERE id = $1 RETURNING \*", [id]

La variable id est recuperer avec la methode get et correspond a l’identifiant de l’utilisateur.

VI.2.4 Recherche des données :

//Recherche d’une notification

SELECT \* FROM notifications WHERE user\_id = $1, [id]

// Recherche d’un utilisateur en utilisant son email

SELECT \* FROM users WHERE email = $1", [email]