MDLD – Modélisation DB - EPIC GAMES

Table of Contents

[Description de l’application Web dans son ensemble (en 15 lignes max) 2](#_Toc197333233)

[Contenu fonctionnel 2](#_Toc197333234)

[Volumétrie approximative (nb utilisateurs simultanés, nombre d’objets gérés, …) 2](#_Toc197333235)

[Utilisateurs cibles (grand public/professionnels, utilisateurs non identifiés/abonnés, etc) 2](#_Toc197333236)

[Définition du périmètre étudié 3](#_Toc197333237)

[Prints screens du site web modélisé avec mise en évidence des principales entités détectées. 3](#_Toc197333238)

[Délimitation du modèle à un sous-ensemble fonctionnel de l’application (si nécessaire) 3](#_Toc197333239)

[Modèle conceptuel des données 3](#_Toc197333240)

[Schéma avec Win’Design 3](#_Toc197333241)

[Description de toutes les entités répertoriées 3](#_Toc197333242)

[Description de l’identifiant 3](#_Toc197333243)

[Description de tous les attributs (limiter à 10 au maximum par entité) 3](#_Toc197333244)

[Description de toutes les associations répertoriées entre entités 3](#_Toc197333245)

[Description précise des cardinalités 3](#_Toc197333246)

[Description des attributs éventuels 3](#_Toc197333247)

[Description des contraintes éventuelles 3](#_Toc197333248)

[Modèle logique des données 3](#_Toc197333249)

[Cas types d’utilisation 3](#_Toc197333250)

[Test des cas types d’utilisation 3](#_Toc197333251)

[Requête SQL 3](#_Toc197333252)

[Implémentation 3](#_Toc197333253)

[Méthodologie de création des données 4](#_Toc197333254)

# Description de l’application Web dans son ensemble (en 15 lignes max)

## La plateforme web EPIC GAMES Store est un système de distribution numérique permettant aux utilisateurs d’acheter, télécharger et gérer leurs jeux. Les développeurs peuvent y publier leurs titres, suivre les ventes et effectuer des mises à jour. L’application gère les amis, les succès, les critiques, les remboursements et les bibliothèques personnelles. Une interface d’administration contrôle les utilisateurs, les jeux, les paiements et la modération. Des fonctionnalités comme les événements, les promotions et les bundles complètent l’écosystème, conçu pour être simple, fluide et évolutif.

## Contenu fonctionnel

La base de données prend en charge les principales fonctionnalités d’une plateforme de distribution numérique de jeux. Elle gère les comptes utilisateurs, les bibliothèques de jeux, les achats, les relations sociales (amis, discussions), les succès en jeu, les critiques et les remboursements. Côté professionnel, les développeurs peuvent publier, mettre à jour et suivre les performances de leurs jeux. Des modules spécifiques permettent d’organiser des promotions, des bundles, et de gérer différents moyens de paiement. Une interface d’administration centralise la gestion des utilisateurs, des contenus, des statistiques et du support.

## Volumétrie approximative (nb utilisateurs simultanés, nombre d’objets gérés, …)

La plateforme vise à gérer **plus de 50 000 utilisateurs enregistrés**, avec **500 à 2 000 connexions simultanées**. Elle référence environ **5 000 jeux**, pour un total de **plus de 000 transactions**. On estime à **300 000+** les relations sociales (amis, invitations), et à **plusieurs millions** les succès déverrouillés par les joueurs. Le système enregistre aussi des **centaines de milliers d’évaluations, tickets et demandes de remboursement**. Ces volumes exigent une base de données robuste, optimisée et évolutive.

## Utilisateurs cibles (grand public/professionnels, utilisateurs non identifiés/abonnés, etc)

La plateforme s’adresse à plusieurs types d’utilisateurs. Le **grand public** peut naviguer dans la boutique et consulter les jeux sans créer de compte. Les **utilisateurs enregistrés** ont accès à l’achat de jeux, leur bibliothèque personnelle, les fonctionnalités sociales (amis, messages) et les succès. Les **développeurs** ou **éditeurs professionnels** peuvent publier des jeux, gérer leurs mises à jour et suivre les statistiques de ventes. Enfin, les **administrateurs** assurent la gestion globale du système (modération, contenu, support, remboursements), tandis que des **modérateurs** peuvent valider les contenus ou gérer les litiges. Cette segmentation permet d’offrir une expérience adaptée à chaque profil.

# Définition du périmètre étudié

Le périmètre d’étude se base sur le site web et l’application de bureau EPIC GAMES. Les fonctionnalités entre ces 2 derniers ne sont pas exactement les mêmes, sur le navigateur web, il est possible de voir tous les jeux, les infos relatives au compte et plus. Cependant il n’est pas possible de voir la liste des jeux acquis. C’est pourquoi l’application de bureau est importante pour ce cas-là.

## Prints screens du site web modélisé avec mise en évidence des principales entités détectées.

Bibliothèque des jeux :

* Compte utilisateur  
  A screenshot of a computer

  AI-generated content may be incorrect.
* Jeux
* Collections (All, Favorites, Adventure Games) A screenshot of a video game

  AI-generated content may be incorrect.
* Informations sur le jeu  
  A screenshot of a video game company

  AI-generated content may be incorrect.A screenshot of a black screen

  AI-generated content may be incorrect.  
    
    
  A screen shot of a computer

  AI-generated content may be incorrect.A screen shot of a black background

  AI-generated content may be incorrect.  
  A screenshot of a black screen

  AI-generated content may be incorrect.  
    
  A screenshot of a computer

  AI-generated content may be incorrect.
* Type de contenu  
  A screenshot of a black screen

  AI-generated content may be incorrect.
* Succès  
  A screenshot of a video game

  AI-generated content may be incorrect.
* DLC / Add ons (Contenu supplémentaire au jeu de base)  
  A screenshot of a video game

  AI-generated content may be incorrect.
* Panier d’achat  
  A screenshot of a computer

  AI-generated content may be incorrect.
* Liste de souhait  
  A screenshot of a computer

  AI-generated content may be incorrect.
* Liste d’amis  
  A screenshot of a social media account

  AI-generated content may be incorrect.
* News  
  A screenshot of a computer

  AI-generated content may be incorrect.
* Code de réclamation d’un produit  
  A screenshot of a computer

  AI-generated content may be incorrect.
* Assistance   
  A screenshot of a computer screen

  AI-generated content may be incorrect.
* Transactions :   
  A screenshot of a black screen

  AI-generated content may be incorrect.
* Achievements :   
  A screenshot of a video game

  AI-generated content may be incorrect.

## Délimitation du modèle à un sous-ensemble fonctionnel de l’application

Le périmètre étudié couvre les entités principales qui serviront de base à la modélisation complète :

* Les Utilisateurs permettant la création de compte, l’authentification, le profil, les achats dont un panier, liste de souhait, transactions, la bibliothèque, les succès et la liste d’amis, etc. (10-15 tables)
* Les jeux fournissant un catalogue complet avec leur prix, et données associées. Ainsi que la plateforme utilisée, les dlc (Contenu supplémentaire à téléchargé séparément du jeu originel), la configuration optimale pour y jouer, son éditeur/développeur et des images/vidéos (5-10 tables)
* Un système de promotion et de gestion de réduction
* Des genres de jeux pour les classer en catégories
* Les actualités (sortie d’un jeu, promotion, etc…)
* Un système d’assistance

# Modèle conceptuel des données

Tables pouvant être traitées (Ces tables sont un aperçu de base. La version finale peut être différente de cela) :   
A list of tasks with black text

AI-generated content may be incorrect.

## Schéma avec Win’Design

A diagram of a network

AI-generated content may be incorrect.

## Description de toutes les entités répertoriées

### Person

La table person est une entité centrale du système, représentant tous les acteurs humains interagissant avec la plateforme. Elle stocke les informations de base de chaque individu enregistré.  
Cette table est parente d’un héritage pour gérer la base d’autre tables.

Chaque individu enregistré dans le système est identifié de manière unique par cette valeur. Sa longueur est de 7 étant donné que beaucoup de personnes peuvent être enregistrées sur le site.

#### Attributs perUsername: (Alphanumérique (AL) de longueur 100). Représente le nom d'utilisateur unique choisi par la personne pour se connecter et interagir avec le système.

perFirstName: (Alphanumérique (AL) de longueur 100). Correspond au prénom de la personne.

perBirthDate: (Date (D) de longueur 8). Enregistre la date de naissance de la personne.

perCreationDate: (Date (D) de longueur 8). Indique la date à laquelle le compte de la personne a été créé et enregistré dans le système.

#### Associations

Parent d’héritage. Les enfants sont : Staff, User, Publisher, Contributor et developer

Un développeur est non seulement associé à un jeu, mais est surtout définit afin que certaines personnes aient accès à un projet. Il est donc possible en tant que développeur de créer des projets qui ont un moteur de jeu associé (Unreal Engine).

Person 0-N -> 1-1 Adress : Une personne peut avoir plusieurs adresses et une adresse n’appartient qu’à une personne.

### Product

La table product est la deuxième entité centrale du projet. Elle est parente de tous les produits disponibles sur Epic Games.

Chaque produit est répertorié par un identifiant unique numérique de longueur 7 afin de répertorier une énorme quantité de produits.

#### Attributs

proTitle: (Alphanumérique Variable (AV) de longueur 150). Représente le titre ou le nom du produit.

proDescription: (Texte (TXT) de longueur 1000). Fournit une description détaillée du produit.

proReleaseDate: (Date (D) de longueur 8). Indique la date de sortie ou de publication du produit.

proPrice: (Décimal (DEC) avec 10 chiffres au total et 2 chiffres après la virgule). Représente le prix standard du produit.

proDiscount: (Décimal (DEC) avec 3 chiffres au total et 2 chiffres après la virgule). Indique le pourcentage de réduction applicable sur le prix du produit (par exemple, 0.15 pour 15%).

proRewardPercentage: (Numérique Signé (NS) de longueur 3). Représente le pourcentage de récompense ou de points de fidélité que l'achat de ce produit génère (par exemple, 5 pour 5%).

proRefundable: (Booléen (BOO) de longueur 1). Un indicateur booléen (vrai/faux) qui détermine si le produit est remboursable ou non.

#### Assotiations

Un produit est parent d’un héritage de tout type de produits qui peuvent être proposés sur la plateforme : Bundle, Subscription, Game et Dlc.

Product -> RedeemCode : Il est possible de récupérer un produit grâce a une clef pouvant être généré par le commerçant. Un produit peut avoir plusieurs clefs générées.

Product -> User : Il y a plusieurs liaisons pour cela : La liste des produits dans le panier, dans la wishlist ou même dans la librairie. Ils sont les 3 en 0-N de chaque côté car chaque produit peut être acquiert par plusieurs utilisateurs.

Product -> Purchase : Entre ces 2 tables, il y aura une liste de produits acheté par un utilisateur. Chaque utilisateur peut donc effectuer plusieurs achats et chaque achat peut contenir plusieurs produits achetés. Chaque achat à été effectué par une méthode de paiement spécifique (Carte de crédit/débit, paypal, etc…) caractérisé par la table PaymentMethod.

### SupportTicket

Dans le système de support, un utilisateur peut créer des tickets afin de les envoyer à des membres du staff afin qu’ils puissent répondre.

#### Attributs

id\_supportTicket: (Numérique (N) de longueur 7). Clé primaire unique de la table supportTicket, identifiant chaque ticket de support de manière non ambiguë.

supSubject: (Alphanumérique Variable (AV) de longueur 150). Représente le sujet ou le titre bref du ticket de support.

supDescription: (Texte (TXT) de longueur 1000). Fournit une description détaillée du problème ou de la requête du support.

supCreatedAt: (Date (D) de longueur 8). Indique la date et l'heure de création du ticket de support.

supClosedAt: (Date (D) de longueur 8). Enregistre la date et l'heure de fermeture du ticket de support, si applicable.

#### Assotiations

SupportTicket 1-1 -> 0-N SupportCategory : Permet d’avoir une catégorie unique par ticket

SupportTicket 1-1 -> 0-N User : Un utilisateur peut créer plusieurs tickets de supports et ce ticket n’appartient qu’à lui seul.

SupportTicket 0-N -> 1-1 TicketMessage : Un ticket de support peut contenir plusieurs messages afin de communiquer avec les staffs, cependant, un message ne peut être .

TicketMessage 1-1 -> 0-N Staff/User : Un message peut être écrit soit par un staff, soit par un utilisateur sachant que chacun d’eux peuvent avoir plusieurs messages à leur actif.

#### Contraintes

#### Le diagramme de base de données indique une contrainte XT (Exclusive, Total) sur l'association "write" entre l'entité person et l'entité ticketMessage. Cette contrainte est spécifiquement placée sur le rôle des personnes impliquées dans l'écriture, distinguant le staff et l’utilisateur.

"Exclusive" : Cette partie de la contrainte stipule qu'une personne qui écrit un ticketMessage doit être soit un membre du staff, soit un user, mais pas les deux simultanément pour un même message. En d'autres termes, un ticketMessage donné est écrit par une personne qui est clairement identifiée comme appartenant à l'une de ces deux catégories mutuellement exclusives.

"Total" : Cette partie de la contrainte signifie que toute personne qui écrit un ticketMessage doit obligatoirement appartenir à l'une de ces deux catégories (staff ou user). Il n'est pas possible qu'un message soit écrit par une personne qui ne soit ni staff ni user.

### Game

Un jeu est un enfant d’un produit, cependant il utilise des relations spécifiques à lui-même afin d’avoir des précisions que d’autres produits n’auraient pas

#### Attributs

isDemo: (Booléen (BOO) de longueur 1). Un indicateur booléen (vrai/faux) qui détermine si le jeu est une version de démonstration (demo) ou la version complète.

#### Assotiations

Un jeu a plusieurs associations qui sont en 0-N -> 0-N concernant plusieurs tables. Un jeu est définit par plusieurs spécificités : Une ou plusieurs plateformes, fonctionnalités, genres et même différentes spécifications concernant le pc du joueur (le cpu requis, la mémoire, l’os, etc…)

D’autres spécifications sont : Le développeur et le studio le publiant. Ces 2 relations sont Game 1-1 -> 0-N, car un jeu n’a qu’une seule équipe de développement et un seul studio de publication. Cependant les studios eux peuvent travailler sur plusieurs jeux.

Game 0-N -> 0-N Collection 1-1 -> 0-N User : Un jeu peut faire partie d’une ou plusieurs collections créées par un utilisateur. Un utilisateur peut avoir plusieurs collections afin de ranger ses jeux selon sa volonté.

Game 0-N -> 1-1 Dlc : Un Dlc est un contenu additionnel au jeu de base et celui-ci contient un attribut booléen définissant si le jeu de base est requis afin d’y jouer. Cette relation permet donc de dire qu’un jeu peut avoir zéro ou plusieurs dlc. Ces dlc étant des produits eux aussi sont donc potentiellement gratuits ou achetables.

Game 0-N -> 1-1 Achievement 0-N -> 0-N User : Cette longue relation permet de définir qu’un jeu peut avoir des achèvements. Ce sont des objectifs spécifiques à atteindre dans le jeu. Un achèvement donné est toujours lié à un seul et unique jeu. D'autre part, un achèvement peut être obtenu par plusieurs utilisateurs, et un utilisateur peut débloquer plusieurs achèvements sur différents jeux.

Game 0-N -> 1-1 version : Un jeu peut avoir plusieurs versions le concernant, il contient aussi le changeLog et la date de sortie

#### Contraintes

Un jeu a une ou plusieurs versions, et une version peut aussi être associée à un moteur de jeu, par exemple Unreal Engine. Cependant, il y a une contrainte d'intégrité XT (Exclusive, Totale) qui oblige une version à être **soit** une version **pour un jeu**, **soit** une version **pour un moteur de jeu**, mais **pas les deux** simultanément, garantissant ainsi que chaque entrée de version est clairement et exclusivement assignée à l'un de ces deux types.

### User

Un utilisateur est un enfant d’une personne, cependant il utilise des relations spécifiques à lui-même afin d’avoir des précisions que d’autres produits n’auraient pas. Beaucoup des relation d’utilisateur ont été couvertes précédemment.

#### Attributs

usePassword: (Alphanumérique Variable (AV) de longueur 128). Cet attribut stocke le mot de passe de l'utilisateur. Sa longueur suggère qu'il est destiné à contenir un hachage sécurisé du mot de passe plutôt que le mot de passe en clair.

useDisplayName: (Alphanumérique Variable (AV) de longueur 100). Représente le nom d'affichage ou le pseudonyme de l'utilisateur, qui peut être visible par d'autres utilisateurs ou dans l'interface du système.

#### Assotiations

User 0-N -> 0-N User : Cette relation peut commune permet ici de définir les amis d’un utilisateur. Cela va donc créer une nouvelle table contenant 2 fois des identifiants représentants un utilisateur (l’utilisateur et son ami).

User 1-1 -> 0-N Language : Un utilisateur parle une seule langue dans le cadre de l’application. Cela va aussi changer la langue visible de l’application. Cependant, une langue peut être parlée par plusieurs personnes.

### Resource

Toute ressource est répértoriée dans cette table. Cela permet d’associer une ressource a n’importe quelle autre table et de l’afficher quand il y a besoin.

#### Attributs

id\_resource: (Numérique (N) de longueur 7). Clé primaire unique de la table resource, identifiant chaque ressource de manière non ambiguë.

resName: (Alphanumérique Variable (AV) de longueur 128). Représente le nom de la ressource.

resPath: (Alphanumérique Variable Long (ALV) de longueur 255). Indique le chemin d'accès ou l'emplacement de la ressource, par exemple un chemin de fichier ou une URL.

#### Assotiations

Resource 1-1 -> 0-N ResourceType : Chaque ressource est obligatoirement et uniquement associée à un type de ressource. Un type de ressource, quant à lui, peut être utilisé par zéro ou plusieurs ressources.

Resource 0-1 -> 0-N Product : Un produit peut avoir plusieurs ressources comme des images ou vidéos pour représenter les jeux, ou par exemple afficher des images du jeu ou un trailer afin de donner envie aux utilisateurs d’en acheter. Une ressource peut être liée à un seul produit ou à aucun.

Resource 0-1 -> 0-N Achievement : Une ressource peut être associée à un succès, par exemple pour fournir une icône ou une image visuelle de l'achèvement. Inversement, un achèvement peut être représenté par zéro ou plusieurs ressources.

Resource 0-1 -> 0-N New : Une ressource peut être liée à une nouveauté, servant par exemple d'illustration ou de média accompagnant une annonce. Une nouveauté, de son côté, peut être associée à aucune ou plusieurs ressources.

#### Contraintes

Il y a aussi une contrainte d'intégrité XT (Exclusive, Totale) sur les associations de resource vers achievement, product et new. Cela permet de dire qu'une ressource n'appartient qu'à un seul d'entre eux : soit elle est liée à un achievement, soit à un product, soit à une new (nouveauté), mais elle ne peut pas être liée à plusieurs de ces entités simultanément. De plus, cette contrainte implique qu'une ressource doit obligatoirement être associée à l'une de ces trois entités.

## Modèle logique des données

A diagram of a computer program

AI-generated content may be incorrect.

# Cas types d’utilisation

# Recherche et Achat de Jeu avec Réduction et Récompense : Un utilisateur recherche un jeu en utilisant des critères (par exemple, titre, genre, éditeur). Il trouve un jeu qui lui plaît, vérifie sa description, sa date de sortie et son prix. Le système applique une éventuelle réduction et calcule le pourcentage de récompense associé à l'achat. L'utilisateur procède ensuite à l'achat du produit via une méthode de paiement enregistrée, ce qui déclenche l'enregistrement de la transaction et la mise à jour de ses succès éventuels liés à l'achat. Entités et Relations Impliquées: personne (user), product, game, purchase, paymentMethod, publisher, game\_genre.

# Gestion des Tickets de Support et Communication : Un utilisateur rencontre un problème et crée un nouveau ticket de support, en décrivant le sujet et la description du problème. Un membre du personnel (staff) prend en charge le ticket et communique avec l'utilisateur en ajoutant des messages au ticket. Le système assure que la personne écrivant le message est bien identifiée comme étant l'utilisateur ou le staff. Une fois le problème résolu, le ticket est marqué comme fermé. Entités et Relations Impliquées: personne (user, staff), supportTicket, ticketMessage.

# Exploration de Contenu et Déverrouillage d'Achèvements : Un utilisateur explore des ressources (images, vidéos) liées à un jeu ou à des nouveautés pour en apprendre davantage. En jouant au jeu, il atteint des objectifs spécifiques et débloque des achèvements. Le système enregistre ces succès pour l'utilisateur, et affiche les ressources associées à ces achèvements. Entités et Relations Impliquées: personne (user), game, resource, achievement, new, resourceType.

# Gestion et Utilisation de Codes de Réduction / Échange : Un utilisateur possède un code de réduction ou d'échange. Il utilise ce code pour obtenir un produit ou un avantage. Le système valide le code, applique l'avantage (par exemple, un prix réduit sur un produit ou l'ajout d'un produit gratuit), et marque le code comme utilisé. Entités et Relations Impliquées: personne (user), redeemCode, product, redeem.

# Test des cas types d’utilisation

## Requête SQL

* 1. Pour chacun des cas d’utilisation, écrire la requête SQL correspondante (recherche avec jointure, enregistrement de données, modification de données, …)

## Implémentation

* 1. Vous devez implémenter votre modèle dans une base MySQL ou MariaDB avec un jeu de données exemples, vous permettant de tester vos requêtes et de valider votre modèle.

## Méthodologie de création des données

* 1. Description de la méthodologie de création des données de test. Les 10 fichiers CSV contenant les données sont documentés. Le générateur de données utilisé est cité et présenté.