Rapport de Projet : Base de Données des PNJ d'un Jeu de Fantasy

YAO Paul-Alex
SAVATIER Arthur
SIVAHARAN Kevin

15 mars 2025

Calendrier du Projet

Étape	Date	Durée	Description
Conception de la base de données	6 mars 2025	1h	Modélisation des tables,des relations avec looping
Implémentation SQL	6 mars 2025	2h	création des contraintes et insertion des données et écriture des requêtes SQL.
Tests et validation	15 mars 2025	2h	Vérification des contraintes, des relations et des requêtes SQL.
Rédaction du rapport	15 mars 2025	1h	Synthèse du travail réalisé et rédaction du rapport final.

TABLE 1 – Calendrier du projet

Introduction

Dans le cadre de notre projet de base de données, nous avons conçu un système pour gérer les **Personnages Non-Joueurs (PNJ)** d'un jeu de fantasy. Ce projet vise à structurer et organiser les informations relatives aux PNJ, leurs caractéristiques, leurs localisations, ainsi que les quêtes auxquelles ils sont associés. L'objectif est de fournir une base de données optimisée pour faciliter la gestion des données dans un environnement de jeu .

1 Description des Étapes du Projet

1.1 Conception de la Base de Données

La première étape a consisté à modéliser la base de données en identifiant les **tables principales** et les **relations** entre elles. Voici les tables principales que nous avons conçues :

- **PNJ** : Contient les informations des Personnages Non-Joueurs (nom, race, classe, niveau, métier, statistiques).
- **Quête** : Stocke les quêtes du jeu (nom, type, difficulté, extension).
- **Localisation** : Définit les lieux où se trouvent les PNJ (région, contrée, lieu).
- **Extension** : Représente les extensions du jeu auxquelles les quêtes appartiennent.
- **Intervenir** : Table associative qui définit le rôle d'un PNJ dans une quête.

Les **relations** entre les tables sont les suivantes :

- Un PNJ est associé à une **localisation** (relation 1 :1).
- Une quête peut appartenir à une **extension** (relation 0 :1).
- Un PNJ peut intervenir dans plusieurs quêtes avec des rôles différents (relation 1 :N).

1.2 Implémentation SQL

Une fois la conception terminée, nous avons implémenté la base de données en SQL directement à l'aide de looping. Voici les principales étapes :

1. **Création des tables** : Nous avons créé les tables en respectant les contraintes d'intégrité (clés primaires, clés étrangères, contraintes de domaine).
2. **Insertion des données** : Nous avons peuplé les tables avec des données fictives pour simuler un environnement de jeu réaliste. (à l'aide de chat gpt)

3. **Écriture des requêtes SQL** : Nous avons écrit des requêtes pour explorer les données, comme la recherche de PNJ par race, la liste des quêtes par difficulté, ou les PNJ intervenant dans une quête spécifique.

1.3 Contraintes et Règles de Gestion

Pour garantir l'intégrité des données, nous avons appliqué des **contraintes** conformes aux règles normales et assez strictes pour pouvoir gérer optimalement la base de donnée :

- **Table PNJ** :
  - Les statistiques (force, sagesse, vitalité, agilité) sont limitées à des intervalles définis.
  - La classe et le métier sont restreints à des valeurs prédéfinies.
  - Les métiers sont limités en fonction de la race du PNJ (ex : un Elfe ne peut pas être Forgeron).
  - Un PNJ ne peut pas être associé à plusieurs localisations en même temps.
- **Table Quête** :
  - Le type de quête est limité à "Principale" ou "Annexe".
  - La difficulté est comprise entre 1 et 5.
- **Table Intervenir** :
  - Le rôle d'un PNJ dans une quête est limité à des valeurs prédéfinies (Donneur de quête, Ennemi, Boss, Allié, Marchand).

1.4 Tests et Validation

Nous avons testé la base de données en vérifiant que les contraintes étaient bien respectées et que les requêtes SQL retournaient les résultats attendus. Par exemple :

- Vérification que les PNJ de métier "Divinité" ont bien un niveau d'au moins 80.
- Vérification que les quêtes de type "Annexe" ont bien une difficulté comprise entre 1 et 5.

2 Conclusion

2.1 Forces de la Solution

- **Structuration claire** : Les tables et les relations sont bien définies, ce qui facilite la gestion des données.
- **Intégrité des données** : Les contraintes garantissent que les données respectent les règles métier du jeu pour ne pas avoir des données dans le désordre ce qui n'est pas optimal.
- **Flexibilité** : La base de données permet d'ajouter facilement de nouveaux PNJ, quêtes ou localisations respectant les contraintes déjà définies.

2.2 Faiblesses et Améliorations Possibles

- **Complexité des requêtes** : Certaines requêtes, notamment celles impliquant des jointures multiples, peuvent être complexes à écrire et à optimiser.
- **Évolution** : Pour un jeu plus complexe, il pourrait être nécessaire d'ajouter des tables supplémentaires (ex : objets, compétences) qui nous permettraient l'ajout de nouvelles contraintes et une meilleure gestion des données.
- **Interface utilisateur** : Actuellement, la base de données est manipulée via SQL. Une interface graphique simplifierait son utilisation.

2.3 Apprentissages

Ce projet nous a permis de :
— Approfondir nos connaissances en modélisation de bases de données relationnelles.
— Maîtriser les contraintes SQL pour garantir l'intégrité des données.
— Développer des compétences en écriture de requêtes SQL complexes.

Annexes

— **Fichier SQL** : Contient le script de création des tables, les contraintes et les requêtes SQL.
— **Analyse du Cahier des charges** : Document détaillant les objectifs, les contraintes et les fonctionnalités du projet.
— **PS** : correction des fautes d'orthographe avec IA