# INFORMATIONS GENERALES

|  |  |  |
| --- | --- | --- |
| Apprenti : | Nom: Zhuravel | Prénom :Mykola |
| Lieu de travail : | ETML / VENNES / 1010 LAUSANNE | |
| Chef de projet : | Nom: | Prénom: - |
| Du lundi xx.xx.202x au vendredi xx.xx.202x | |
| Dates de réalisation : | x périodes par semaine, x semaines (moins les jours fériés !) | |
| Horaire de travail : | 4 périodes par semaine | |
| Temps total : | 32 périodes | |

# PROCÉDURE

* Tous les apprentis réalisent le projet sur la base d'un cahier des charges.
* Le cahier des charges est présenté, commenté et discuté en classe.
* Les apprentis sont entièrement responsables de la sécurité et sauvegarde de leurs données.
* En cas de problèmes graves, les apprentis avertissent le client au plus vite.
* Les apprentis ont la possibilité d’obtenir de l’aide externe, mais ils doivent le mentionner.
* Les informations utiles à l’évaluation de ce projet sont disponibles au chapitre 8.

# TITRE

***Thématique « SpicyInvader »***

# SUJET

Travailler avec une base de données sur la thématique « SpaceInvader. »

# MATÉRIEL ET LOGICIEL À DISPOSITION

|  |
| --- |
| * Un PC ETML * Accès à Internet * Docker * MySQL * Looping * Git / Github |

# PRÉREQUIS

|  |
| --- |
| Modules de bases de données |

# CAHIER DES CHARGES

## Réaliser le MCD

Un studio de développement de jeux vidéo souhaite mettre en place une base de données pour gérer les informations relatives aux joueurs, leurs armes, leurs commandes d’achats d’armes, et un suivi des armes possédées par chaque joueur. En utilisant la méthode MERISE, créez le Modèle Conceptuel des Données (MCD) correspondant aux besoins suivants.

Dans le jeu, plusieurs types d'armes sont disponibles. Chaque arme est identifiée de façon unique et possède un nom, une description, un prix (en points), une force (indique le dégât que peut faire une arme offensive, 0 pour une arme défensive), et un nombre maximum de coups utilisables (indique combien de « cartouches » on dispose avec cette arme offensive ou défensive).

Les joueurs sont également identifiés de manière unique. Chacun d’eux possède un pseudonyme et un nombre de points accumulés dans le jeu en détruisant ses vaisseaux ennemis.

Les joueurs peuvent passer des commandes pour acheter des armes. Chaque commande est définie par une date et possède un numéro unique. Une commande ne peut être passée que par un et un seul joueur. Une commande peut inclure plusieurs armes, et pour chaque arme commandée, une quantité est précisée.

Le jeu conserve également un suivi des armes possédées par chaque joueur dans leur arsenal personnel. Ainsi, chaque joueur peut posséder plusieurs armes, et chaque arme peut être détenue par plusieurs joueurs. Pour chaque arme détenue par un joueur, le nombre de coups restants est suivi.

**Vous devez réaliser le MCD correspondant avec le logiciel looping en respectant les nouvelles conventions de nommage de L’ETML.**

Remarque : Pour compléter ou éclaircir le texte ci-dessus, vous devez aller poser des questions à votre enseignant !

**Attention ! Avant de passer à l’étape suivante, vous devez faire valider votre MCD par votre enseignant.**

## Création de la base de données et importation des données

Dans cette étape, vous devez créer la base de données ayant pour nom **db\_space\_invaders**, puis créer des données pour chacun des tables.

Chaque table doit contenir au moins 50 enregistrements.

## Gestions des utilisateurs

Différents utilisateurs vont devoir interagir avec la base de données.

Pour ce faire, ces utilisateurs ont des missions différentes :

1. Administrateur du jeu :

- Peut créer, lire, mettre à jour et supprimer (CRUD) n'importe quelle table.

- Gérer les utilisateurs et leurs privilèges.

2. Joueur :

- Lire les informations des armes (pour voir quelles armes il peut acheter).

- Créer une commande.

- Lire toutes les commandes.

3. Gestionnaire de la boutique :

- Lire les informations sur tous les joueurs (pour savoir qui a passé une commande).

- Mettre à jour, lire et supprimer des armes (ajout de nouvelles armes, modification des prix, etc.).

- Lire toutes les commandes.

Pour chaque « catégorie d’utilisateurs », on pourrait être amener à créer plusieurs voire même un grand nombre d’utilisateurs.

Travail demandé :

Vous devez expliquer, comment mettre en place ces différents « types d’utilisateurs » afin de rendre la gestion des utilisateurs la plus simple possible.

Pour cela, vous devez définir un ensemble de requêtes SQL.

Vous devez créer au moins un utilisateur par « type d’utilisateurs ».

## Requêtes de sélection

Voici une liste de requêtes SQL que vous devez réaliser et expliquer en détail.

Requête n°1 :

La première requête que l’on vous demande de réaliser est de sélectionner les 5 joueurs qui ont le meilleur score c’est-à-dire qui ont le nombre de points le plus élevé. Les joueurs doivent être classés dans l’ordre décroissant

**SELECT joueur\_id, pseudonyme, points**

**FROM t\_joueur**

**ORDER BY points DESC**

**LIMIT 5;**

Requête n°2 :

Trouver le prix maximum, minimum et moyen des armes.

Les colonnes doivent avoir pour nom « PrixMaximum », « PrixMinimum » et « PrixMoyen)

**SELECT**

**MAX(prix) AS PrixMaximum,**

**MIN(prix) AS PrixMinimum,**

**AVG(prix) AS PrixMoyen**

**FROM t\_arme;**

Requête n°3 :

Trouver le nombre total de commandes par joueur et trier du plus grand nombre au plus petit.

La 1ère colonne aura pour nom "IdJoueur", la 2ème colonne aura pour nom "NombreCommandes"

**SELECT**

**joueur\_id AS IdJoueur,**

**COUNT(\*) AS NombreCommandes**

**FROM t\_commande**

**GROUP BY joueur\_id**

**ORDER BY NombreCommandes DESC;**

Requête n°4 :

Trouver les joueurs qui ont passé plus de 2 commandes.

La 1ère colonne aura pour nom "IdJoueur", la 2ème colonne aura pour nom "NombreCommandes"

**SELECT**

**joueur\_id AS IdJoueur,**

**COUNT(\*) AS NombreCommandes**

**FROM t\_commande**

**GROUP BY joueur\_id**

**HAVING COUNT(\*) > 2;**

Requête n°5 :

Trouver le pseudo du joueur et le nom de l'arme pour chaque commande.

**SELECT**

**t\_joueur.pseudonyme,**

**t\_arme.nom**

**FROM t\_commande**

**JOIN concerne ON t\_commande.commande\_id = concerne.commande\_id**

**JOIN t\_arme ON concerne.arme\_id = t\_arme.arme\_id**

**JOIN t\_joueur ON t\_commande.joueur\_id = t\_joueur.joueur\_id;**

Requête n°6 :

Trouver le total dépensé par chaque joueur en ordonnant par le montant le plus élevé en premier, et limiter aux 10 premiers joueurs.

La 1ère colonne doit avoir pour nom "IdJoueur" et la 2ème colonne "TotalDepense"

**SELECT**

**t\_joueur.joueur\_id AS IdJoueur,**

**SUM(t\_arme.prix \* concerne.quantité) AS TotalDepense**

**FROM t\_commande**

**JOIN concerne ON t\_commande.commande\_id = concerne.commande\_id**

**JOIN t\_arme ON concerne.arme\_id = t\_arme.arme\_id**

**JOIN t\_joueur ON t\_commande.joueur\_id = t\_joueur.joueur\_id**

**GROUP BY t\_joueur.joueur\_id**

**ORDER BY TotalDepense DESC**

**LIMIT 10;**

Requête n°7 :

Récupérez tous les joueurs et leurs commandes, même s'ils n'ont pas passé de commande.

Dans cet exemple, même si un joueur n'a jamais passé de commande, il sera quand même listé, avec des valeurs `NULL` pour les champs de la table `t\_commande`.

**SELECT**

**t\_joueur.joueur\_id,**

**t\_joueur.pseudonyme,**

**t\_commande.commande\_id,**

**t\_commande.datedecommande**

**FROM t\_joueur**

**LEFT JOIN t\_commande ON t\_joueur.joueur\_id = t\_commande.joueur\_id;**

Requête n°8 :

Récupérer toutes les commandes et afficher le pseudo du joueur s’il existe, sinon montrer `NULL` pour le pseudo.

**SELECT**

**t\_commande.commande\_id,**

**IFNULL(t\_joueur.pseudonyme, NULL) AS pseudonyme**

**FROM t\_commande**

**LEFT JOIN t\_joueur ON t\_commande.joueur\_id = t\_joueur.joueur\_id;**

Requête n°9 :

Trouver le nombre total d'armes achetées par chaque joueur (même si ce joueur n'a acheté aucune Arme).

**SELECT**

**t\_joueur.joueur\_id,**

**IFNULL(SUM(concerne.quantité), 0) AS NombreTotalArmesAchetees**

**FROM t\_joueur**

**LEFT JOIN t\_commande ON t\_joueur.joueur\_id = t\_commande.joueur\_id**

**LEFT JOIN concerne ON t\_commande.commande\_id = concerne.commande\_id**

**GROUP BY t\_joueur.joueur\_id;**

Requête n°10 :

Trouver les joueurs qui ont acheté plus de 3 types d'armes différentes

**SELECT**

**t\_joueur.joueur\_id,**

**COUNT(DISTINCT concerne.arme\_id) AS TypesArmesAchetees**

**FROM t\_joueur**

**JOIN t\_commande ON t\_joueur.joueur\_id = t\_commande.joueur\_id**

**JOIN concerne ON t\_commande.commande\_id = concerne.commande\_id**

**GROUP BY t\_joueur.joueur\_id**

**HAVING COUNT(DISTINCT concerne.arme\_id) > 3;**

## Création des index

En étudiant le dump MySQL **db\_space\_invaders.sql** vous constaterez que vous ne trouvez pas le mot clé **INDEX**.

1. Pourtant certains index existent déjà. Pourquoi ?

**Certains index existent déjà parce que les clés primaires et les clés étrangères**

**créent automatiquement des index pour garantir l'unicité et améliorer la performance**

**des opérations de jointure et de recherche.**

1. Quels sont les avantages et les inconvénients des index ?

**Avantages :**

**Accélèrent les requêtes SELECT et les recherches.**

**Facilitent le tri et le filtrage des données.**

**Inconvénients :**

**Occupent de l'espace supplémentaire dans la base de données.**

**Ralentissent les opérations d'INSERT, UPDATE, et DELETE car l'index doit être mis à jour.**

1. Sur quel champ (de quelle table), cela pourrait être pertinent d’ajouter un index ? Justifier votre réponse.

**Pseudonyme, prix et l’id de joueur**

## Backup / Restore

Nous souhaitons réaliser une sauvegarde (Backup) de la base de données db\_space\_invaders.

Ensuite, nous souhaitons nous assurer que cette sauvegarde est correcte en la rechargeant dans MySQL (opération de restauration).

Donner la commande permettant de faire :

- Un backup de la base de données db\_space\_invaders

**mysqldump -u root -p root > backup\_db\_space\_invaders.sql**

- Un restore de la base de données db\_space\_invaders

**mysql -u root-p root < backup\_db\_space\_invaders.sql**

En expliquant en détail chaque commande utilisée.

## Livrables

* Un rapport répondant aux différentes questions posées dans ce document.
* Un journal de travail
* Les fichiers SQL
* Le fichier Looping de la modélisation
* Dépôt github

# Évaluation

1. Votre enseignant peut effectuer une ou plusieurs évaluations impromptues à l’oral tout au long du projet.
2. Vous devez être en capacité d’expliquer à votre enseignant chaque requête SQL ou commande que vous avez écrites.
3. Le recours à des outils en ligne d’intelligence artificielle (ex. : Chat GPT) doit être mentionné dans le rapport de manière détaillée et explicite et ne peut servir que d’inspiration à la réalisation ou d’aide à l’apprentissage (montée en compétence en requêtes SQL par exemple). En cas d’abus, l’évaluation du projet en tiendra compte.