

Langages d'exploitation des bases de données

Projet 1

420-C42

Énoncé

Vous devez concevoir et réaliser une base de données répondant à une problématique donnée.

Ce projet se veut un premier contact avec la conception et la réalisation d'une base de données. Pour y arriver, vous devrez réaliser 4 étapes distinctes, mais interdépendantes :

1. conception de la base de données;
2. réalisation du script permettant de créer les objets fondamentaux de la base de données;
3. réalisation du script permettant de populer la base de données;
4. réalisation du script contenant les requêtes demandées.

Problématique

Mise en situation

Vous travaillez pour une entreprise en démarrage désirant s'établir dans le domaine du jeu vidéo. Vous avez rencontré des investisseurs, mais ces derniers restent frileux à cause de votre manque d'expérience. Pour les convaincre, vous décidez de mettre en place l'essentiel d'une base de données permettant un fonctionnement rudimentaire de l'écosystème lié à votre projet : joueurs – avatars – jeux – activités.

Votre entreprise tente de percer le marché avec une idée originale consistant à pouvoir faire évoluer un avatar dans plusieurs mondes différents où chaque monde est un jeu différent. Ainsi, un avatar peut obtenir des habiletés exclusives à certains mondes, mais les réutiliser dans d'autres mondes. Cet aspect de l'écosystème garantit une expérience unique, car certaines habiletés disponibles dans certains mondes sont essentielles à la réussite d'objectifs présents dans d'autres mondes.

Description sommaire du projet

Voici une description sommaire de la structure du projet :

- l'idée consiste à développer plusieurs jeux de type [MMORPG](#) où vont évoluer les différents avatars, chaque jeu correspond à un monde, n_g le nombre de jeux (de mondes)
chacun des jeux (des mondes) est identifié par g_i $g_0, g_1, g_2, g_3, \dots, g_{n_g-1}$
- dans chaque jeu, il existe quelques portails pour aller vers d'autres jeux (un jeu ne peut permettre d'aller vers tous les autres jeux), n_p le nombre de portails
- chaque joueur peut créer autant d'avatars qu'il le désire, toutefois il ne peut en utiliser qu'un seul à la fois afin de le faire évoluer, n_a le nombre d'avatars
chacun des avatars est identifié par a_i $a_0, a_1, a_2, a_3, \dots, a_{n_a-1}$
chaque joueur possède automatiquement un avatar à la création du compte $a_0, n_a > 0$
- chaque joueur peut avoir l'un des deux statuts : *statut*
 - invité, gratuit
 - payant, paye un tarif fixe mensuel
- selon le statut, les possibilités sont les suivantes :
 - invité :
 - ne peut utiliser qu'un seul avatar, celui créé automatiquement lors de la création du compte du joueur
 - ne peut utiliser qu'un seul monde, g_0
 - les portails du monde g_0 sont non disponibles

- payant :
 - le joueur peut créer autant d'avatars qu'il le désire, à n'importe quel moment il peut changer d'avatar
 - le joueur peut, à n'importe quel moment, utiliser tous les portails pour aller dans tous les mondes
- ancien joueur payant qui annule son inscription et qui devient un invité :
 - ne peut utiliser qu'un seul avatar, celui qui a été le plus utilisé
 - ne peut utiliser qu'un seul monde, celui dans lequel l'avatar principal a passé le plus de temps
 - le joueur ne perd pas ses autres avatars et leur avancement
 - à n'importe quel moment il peut se réinscrire et réactiver tous ses avatars dans leur état respectif
- chaque monde offre un nombre varié d'habiletés exclusives n_h le nombre d'habiletés
 - lorsqu'une habileté est acquise, l'avatar peut l'utiliser dans tous les autres mondes
 - une habileté peut évoluer au fil du temps et devenir de plus en plus puissante, ce qui permet aux avatars d'obtenir un niveau de compétence lié indirectement à son temps de jeu.
 - par exemple, les mondes suivants permettent les habiletés suivantes :
 - dans le jeu *SpaceX™* :
 - ralentir le temps (amélioration possible : durée du ralentissement);
 - bouclier de proximité (amélioration possible: durée du bouclier);
 - dans le jeu *DeepHorizonX™* :
 - auto guérison (amélioration possible : capacité d'auto guérison plus fréquente);
 - implant d'un zoom oculaire (amélioration possible : niveau du zoom);
 - dans le jeu *FreeZoneX™* :
 - visée automatique (amélioration possible : précision);
 - ubiquité (amélioration possible: durée de la *multiprésence*).
- chaque monde présente des items rares et exclusifs n_i le nombre d'items
 - il est possible de les obtenir à travers le jeu via diverses situations ou d'en acheter
 - un avatar peut posséder plusieurs quantités du même item
 - les items peuvent utilisés dans n'importe quel monde.
- pour l'écosystème, il existe une unité universelle d'échange utilisée pour différentes situations: le **moX™**
 - les **moX** sont importants dans les jeux, car ils peuvent être utilisés pour améliorer une habileté, acheter des items ou tout simplement faire du troc entre ses avatars, mais aussi entre les avatars des autres joueurs.

Données à traiter

Voici les données à manipuler :

- joueur : les n_p joueurs sont les clients inscrits à notre service
 - alias 32 caractères max, unique, obligatoire
 - courriel 128 caractères max, unique, obligatoire
 - mot de passe 32 caractères max, obligatoire
 - genre une énumération : [femme cisgenre, homme cisgenre, femme transgenre, homme transgenre, non binaire, genre fluide, agenré, autre] optionnel, une valeur nulle indique que le genre est inconnu, défaut NULL
 - dates d'inscription date, obligatoire, doit être après le 1^{er} janvier 2023
 - date de naissance date, optionnel, après le 1^{er} janvier 1900 & doit avoir au moins 13 ans lors de l'inscription, défaut NULL
 - statut une énumération : [invité, payant], obligatoire
- jeu : les n_g jeux sont les différents logiciels supportant chacun des mondes
 - nom 16 caractères max, unique, obligatoire, au moins 3 caractères, doit débuter par une lettre
 - sigle 6 caractères fixes, unique, obligatoire, doit débuter par une lettre
 - description 2048 caractères max, optionnel
 - une liste de n_h habiletés que peut développer un avatar : $n_h \geq 0$
 - nom 32 caractères max, unique, obligatoire
 - sigle 3 caractères fixes terminant toujours par 'P' majuscule, unique, obligatoire
 - énergie maximum nombre variant de [1.000, 1000.000] (3 décimales), obligatoire, défaut 100.0
 - coef₁, coef₂, coef₃ nombre à double précision, obligatoire, ces trois paramètres déterminent la structure de coût selon le niveau :
coût, en **moX**, pour un niveau donné = coef₁ × niveau² + coef₂ × niveau + coef₃
 - description 1024 caractères max, obligatoire
 - une liste de n_i items rares et exclusifs contenus dans le jeu : $n_i \geq 0$
 - nom 32 caractères max, unique, obligatoire
 - sigle 4 caractères fixe débutant toujours par 'I' majuscule, unique, obligatoire
 - probabilité nombre variant de]0.000, 1.000[(3 décimales), défaut 0.025, obligatoire
 - description 1024 caractères max, obligatoire
 - valeur en **moX** nombre variant de [1.00, 1.00e6] (2 décimales), obligatoire, défaut 100.0

- avatar : les n_a avatars sont les personnages virtuels créés par les joueurs
 - nom 32 caractères max, unique, obligatoire, au moins 3 caractères
 - n phrases personnelles (genre leitmotiv) 64 caractères max chacune où n peut être 0
par exemple : *I'm the boss – How do you feel now? – Don't cry!*
 - 3 couleurs préférées encodé par un entier 32 bits, 1 obligatoire & 2 optionnelles, la couleur obligatoire est blanche par défaut
 - date de création date, défaut date courante, obligatoire
 - quantité de **moX** nombre variant de [1.00, 1.00e6] (2 décimales), obligatoire
 - pour chaque habileté développée :
 - date d'obtention de l'habileté date et heure, défaut date et heure courante, obligatoire
 - niveau actuel nombre variant de [1, 100], défaut 1, obligatoire
 - pour chaque item rare obtenu :
 - date d'obtention de l'item date et heure, défaut date et heure courante, obligatoire
 - quantité nombre variant de [1, 1e3], défaut 1, obligatoire
- activité : représente les informations générales de toutes les périodes d'activité des joueurs – par exemple, si le joueur se connecte et joue pendant 45 min le matin puis se reconnecte et joue pendant 3 heures le soir, il faudra enregistrer les informations liées à ces deux périodes d'activité
 - date et heure du début de l'activité date et heure, obligatoire
 - durée de l'activité en seconde, obligatoire, doit être supérieure à 0
 - pour la durée de l'activité, on désire enregistrer toutes les « capsules d'activité » - une « capsule d'activité » correspond à la liste de temps passé dans chaque monde pour chaque avatar joué pendant la durée de l'activité
int, en seconde, obligatoire, doit être supérieure à 0

Note importante

Les données présentées plus haut sont celles à considérer pour le projet. Vous remarquerez que plusieurs aspects de la description du projet donné en introduction sont absentes des données. Sachez que vous **ne devez pas** compléter les données pour représenter toutes les nuances du projet présenté. L'introduction est faite dans l'objectif de mieux comprendre le contexte général.

Par exemple, vous n'avez pas à considérer :

- l'aspect financier du projet;
- les données stockées pour construire les mondes eux-mêmes ou les structures des avatars;
- la programmation des jeux eux-mêmes;
- ...

À considérer

Lorsque vous serez en industrie, ce genre de spécification correspondre à ce qu'un analyste logiciel peut produire.

Les données qui vous sont présentées sont considérées comme potentiellement incomplètes, fragmentées et non structurées pour une implémentation directe. Il est possible que l'ordre de présentation ne corresponde aucunement à la structure de votre base de données. C'est à vous d'interpréter et de transformer ces informations en modèle relationnel utilisable pour un vrai projet informatique de base de données.

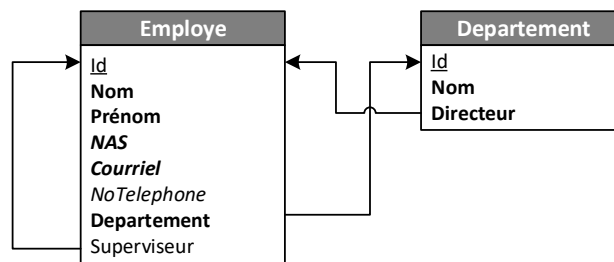
Finalement, votre enseignant a le rôle de client. Si vous avez des questions sur le projet et surtout pour clarifier des détails du mandat, c'est à lui qu'il faut s'adresser comme s'il était votre client. Cet énoncé présente plusieurs informations incomplètes et subtilités qu'il est possible d'interpréter différemment. Si vous ne le faites pas, il est probable que vous ayez une mauvaise interprétation de la situation et que vous perdiez des points en conséquence. Il est donc attendu que vous posiez des questions sur le projet.

Les quatre tâches à réaliser

1. Conception

Vous devez prendre le temps de réfléchir à ce projet. Le défi est de le faire en équipe et de trouver une solution adéquate qui assure l'intégrité des données. Vous devez vous assurer de n'avoir aucune redondance structurelle d'information, c'est la partie la plus importante de tout le projet. Vous devez considérer qu'aucune optimisation n'est souhaitée au détriment de la répétition structurelle.

On vous demande de créer un graphique typique de base de données. Vous pouvez inventer votre propre notation pourvu que vous ajoutiez une légende explicative. Voici un exemple typique :



Exemple de schéma relationnel

Ce schéma est un exemple minimal, car on vous demande d'ajouter les contraintes suivantes sur le graphique : clé primaire (soulignée), clé étrangère (illustré par une flèche directionnelle), unicité (en caractères italiques) et obligatoire (en caractères gras). Dans tous les cas, vous devez ajouter une légende explicative de votre notation.

Cette partie du travail, souvent sous-estimée et bâclée, est probablement la plus importante de tout le projet. Vous pouvez remettre un document réalisé avec un programme de dessin vectoriel comme Visio ou produit à la main (à condition qu'il soit propre, numérisé et intégré dans votre remise en version électronique – le document papier n'est pas à remettre). Vous pouvez aussi considérer utiliser le logiciel en ligne [dbdiagram](#) (attention, il ne faut pas utiliser cette plateforme pour générer le code SQL automatiquement).

2. Création des tables et contraintes

Vous devez créer un script (en format `pgsql`) permettant de créer automatiquement tous les objets fondamentaux de la base de données :

- types énumérés,
- tables,
- attributs,
- contraintes.

Votre script doit permettre de détruire les objets s'ils existent. N'oubliez pas qu'on vous demande d'ajouter toutes les contraintes de clé étrangères à la fin du script.

3. Insérer des enregistrements dans la base de données

On vous demande de rédiger un script permettant de populer les différentes tables avec des données pertinentes et cohérentes les unes avec les autres. Vous devez respecter ces nombreuses contraintes :

- $m^{[1]}$ joueurs différents (les joueurs doivent avoir vos noms) – un des joueurs sera désigné joueur principal (identifié par un * à la fin du nom) et contiendra plus de données que les autres pour les parties décrites plus bas;
- 1 avatar pour chaque joueur, le joueur principal doit en posséder $n^{[2]}$ dont un sera considéré principal (identifié par un * à la fin du nom de l'avatar);
- $n^{[2]}$ jeux ayant chacun $n^{[2]}$ habileté pour les avatars et un total de $1.5n^{[2]}$ items rares;
- pour les capsules d'activités :
 - le joueur principal doit avoir $2m^{[1]}$ capsules d'activité dont $1.5m^{[1]}$ sont liés à l'avatar principal - les capsules d'activité doivent être distribuées en $m^{[1]}$ activités différentes
 - tous les autres joueurs doivent avoir $n^{[2]}$ capsules d'activité distribuées sur un nombre d'activités laissé à votre discrétion;
- considérant toutes les autres tables :
 - sauf pour les tables de liaison, vous devez insérer au moins $n^{[1]}$ entrées pertinentes et cohérentes avec le reste des données,
 - pour les tables de liaison, vous devez avoir les entrées nécessaires à la consistance de votre base de données considérant toutes les données présentes.

4. Rédaction de requêtes spécifiques

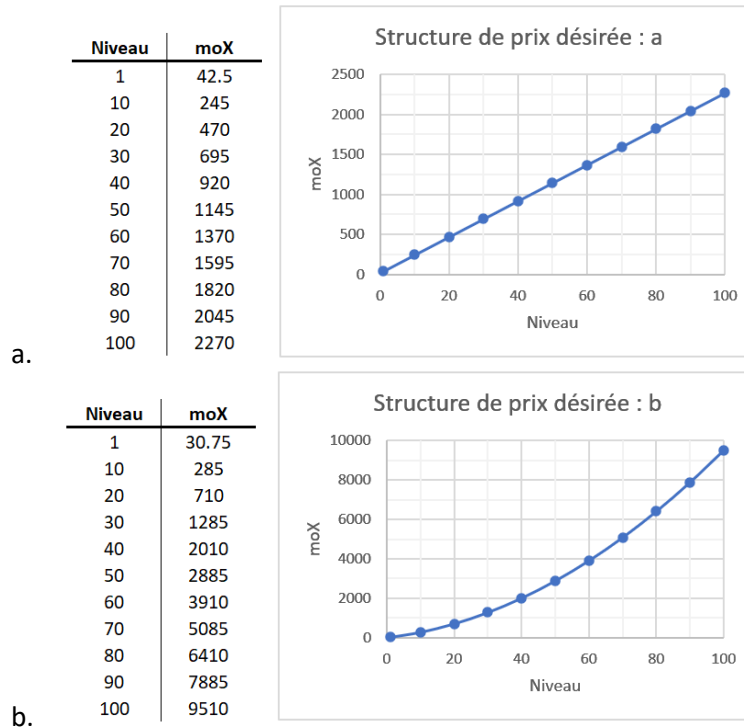
On vous demande de créer un script avec les requêtes répondant aux questions suivantes :

1. Donnez la liste de tous les joueurs, on désire : alias, courriel, date d'inscription. Le tout trié par date d'inscription (croissant).
2. Donnez la liste des avatars d'un joueur quelconque (pour l'exemple, prendre le joueur principal), en donnant : nom, la couleur préférée transformée en trois composantes rouge, vert, bleu (dans une seule colonne selon ce format : « (127, 0, 255) »), date de création suivant le format 2000 | 12 | 25, le nombre de **moX**.
3. Pour l'avatar principal, donnez toutes les habiletés qu'il possède en présentant : le sigle et le nom entre crochets dans la même colonne, la date d'obtention, le niveau courant, la valeur en **moX** du niveau courant et le coût en **moX** pour acheter le prochain niveau.
4. Pour l'avatar principal, donnez sa valeur totale : pour les habiletés on considère le niveau et pour les items on considère la quantité.
5. Pour le joueur principal, donnez le nombre total d'heures passées dans chaque jeu joué.
6. Donnez la liste de tous les avatars qui possèdent plus de 1 item : nom du joueur, nom de l'avatar et nombre d'items.
7. Chaque membre de l'équipe doit réaliser une requête **pertinente** de son cru. Considérant toutes les requêtes produites par l'équipe, vous devez respecter ces contraintes :

¹ m représente le nombre d'étudiants dans l'équipe de travail.

² $n = m - 1$

- a. vous devez avoir deux requêtes qui ont au minimum 4 et 5 tables différentes,
 - b. vous devez utiliser au moins une fois ces clauses: LIMIT, GROUP BY, ORDER BY, HAVING.
- À même le script et avant ces requêtes, vous devez ajouter un commentaire expliquant précisément ce que fait la requête. Le pointage associé à chacune de ces requêtes est autant basé sur la pertinence que la qualité de la réalisation technique.
8. [Boni] On désire modifier le prix pour deux des items de façon à avoir les montants indiqués plus bas selon le niveau associé :



Faites deux requêtes permettant de modifier les structures de prix des items 1 et 2 que vous avez créées initialement selon les données des tableaux ci-haut.

Toutes vos requêtes doivent afficher un nom de colonne adéquat.

Finalement, toutes vos requêtes doivent avoir un commentaire d'en-tête présentant :

- No de la requête
- Énoncé de la requête
- Une ligne intitulée *Fonctionnelle* et indiquant :
 - oui (si la requête est complètement fonctionnelle)
 - aucunement (si la requête ne fonctionne pas du tout)
 - partiellement, *en ajoutant une description de ce qui ne fonctionne pas* (si la requête est partiellement fonctionnelle)

Contraintes

Plusieurs contraintes sont à respecter pour ce travail :

1. Vous devez travailler en équipe de 3 ou de 4 étudiants.

2. Il est très important que tous les membres de l'équipe travaillent équitablement, car l'évaluation finale tient compte de plusieurs critères, dont la répartition de la tâche de travail. À ce sujet, vous devez remplir une fiche d'auto-évaluation influençant la note de chaque étudiant en fonction de son apport et de la note d'équipe.
3. Vous devez travailler avec le SGBDR PostgreSQL utilisé tout au long de la session.
4. Vous devez respecter la norme de codage.
5. La date de remise est non négociable.

Stratégie d'évaluation

L'évaluation se fera en 2 parties. D'abord, l'enseignant évaluera le projet remis et assignera une note de groupe pour le travail. Ensuite, chaque équipe devra remettre un fichier Excel dans lequel sera soigneusement reportée une cote représentant la participation de chaque étudiant dans la réalisation du projet. Cette évaluation est faite en équipe et un consensus doit être trouvé.

Une pondération appliquée sur ces deux évaluations permettra d'assigner les notes finales individuelles.

Ce projet est long et beaucoup plus difficile que les exercices. Il est conçu pour être réalisé en équipe. L'objectif est que chacun prenne sa place et que chacun laisse de la place aux autres.

Ainsi, trois critères sont évalués :

- **participation** (présence en classe, participation active, laisse participer les autres, pas toujours en train d'être sur Facebook ou sur son téléphone, concentré sur le projet, pas en train de faire des travaux pour d'autres cours ...)
- **réalisation** (répartition du travail réalisé : conception, modélisation, rédaction de script, tests, documentation ...)
- **impact** (débrouillardise, initiative, amène des solutions pertinentes, motivation d'équipe ...)

Remise

Vous devez créer un fichier de format `zip` dans lequel vous insérez :

- | | |
|---|------------------------------|
| • votre schéma de conception | <code>schema.pdf</code> |
| • le script de création de la base de données | <code>creation.pgsql</code> |
| • le script permettant d'ajouter des données | <code>insertion.pgsql</code> |
| • le script des requêtes demandées | <code>requetes.pgsql</code> |
| • le fichier Excel rempli sur la participation active des membres du groupe | <code>evaluation.xlsx</code> |

Au final, vous devez remettre votre projet une seule fois sur Lea après avoir nommé votre fichier :

`NomEtudiant1_NomEtudiant2[_NomEtudiantN].zip`

Critères d'évaluation

1. Conception (30 points)
 - La conception répond à tous les critères demandés 26 points
 - données bien structurées
 - contraintes
 - aucune redondance
 - répond à l'énoncé
 - Le schéma est propre et lisible 2 points
 - Le schéma présente une légende 2 points
2. Script de création (20 points)
 - La création des tables correspond au schéma 7 points
 - Les types sont définis adéquatement 4 points
 - Les contraintes sont pertinentes et répondent à l'énoncé 5 points
 - Les contraintes de clé étrangère sont toutes ajoutées à la fin 2 points
 - Les objets sont effacés au début du script 2 points
3. Script d'insertion (20 points)
 - 4 [3] joueurs dont un joueur principal 3 points
 - 4 [3] avatars dont un avatar principal 3 points
 - 3 [2] jeux avec 3 [2] habiletés et 6 [4] items 4 points
 - 19 [12] capsules d'activités 4 points
 - autres tables (sauf n vers n) 3 points
 - tables n vers n 3 points
4. Requêtes (22 points)
 - Requête 1 1 point .
 - Requête 2 2 points
 - Requête 3 2 points
 - Requête 4 2 points
 - Requête 5 2 points
 - Requête 6 3 points
 - Requête 7a (3 tables) 2 points
 - Requête 7b (4 tables) 2 points
 - Requête 7c (5 tables) 3 points
 - Requête 7d (5 tables) – pour les équipes de 3, le pointage est redistribué 3 points
 - Requête 8 [boni] *2 points
5. Qualité générale des scripts (8 points)
 - nom des tables, des colonnes et contraintes 1 point .
 - les en-têtes des scripts sont bien documentés 2 points
 - les commentaires généraux sont adéquats 1 point .
 - respect général de la norme de codage 4 points