Travail final d'intégration

SimbaPlus pour aller loin …et bien

- Ce travail sera réalisé **individuellement**
- À terminer pour le mardi 14 décembre 2020, 23h59;
- Le travail compte pour 25 % de la note finale;

Objectifs

Ce travail vise principalement à vous faire expérimenter les aspects suivants:

- Concevoir un modèle de base de données;
- Écrire des procédures stockées;
- Écrire des déclencheurs;
- Expérimenter l'aspect sécurité des données;
- Intégration avec un langage de programmation;
- Veiller à ce que les données soient cohérentes en tout temps (transactions)

Mise en situation

La compagnie SimbaPlus est une PME qui œuvre dans le développement de jeux vidéo. Sa base de données est une base de données MS SQL Sever

Cette compagnie compte deux départements distincts, « Ressources humaines **(RSH) »** et « Développement (**DEV**) ». Chaque département a sa propre base de données.

Le département RSH compte 3 employés : 1 technicien (Martin St-Louis), un commis (Hugo Lapointe) et le responsable du département (***Saturne Lune***).

Le département DEV compte 5 employés : 4 développeurs (Guy Tares, Bien Thierry, Martin Lejeune, et Lyna Pilon) et le responsable du département (***Alain Patoche***).

Authentification au serveur MS SQL

Les tâches spécifiques à la gestion de « MS SQL Server » telles que créer et supprimer une base de données, gérer les rôles et autorisations, créer les connexions et les usagers des bases de données ainsi que de créer les tables dans la base de données **dbrsh** seront réservées uniquement à l'administrateur du serveur qui est **Mathieu Powers**

Pour vaquer à ses activités, l'**administrateur** s'authentifiera au serveur avec la connexion **vos_initiales_dba** (par exemple : **EF_dba** – vous pouvez utiliser une authentification SQL Server pour **vos_initiales_dba).**

Chaque employé de l'entreprise aura sa propre connexion pour s'authentifier au SGBDR. Ces connexions n'auront aucun autre rôle serveur que **public**.

Finalement, le jeu **Trivial Pursuit** aura aussi sa propre connexion, **trivial,** qui sera utilisée par l'interface graphique pour se connecter au SGBDR. Cette connexion n'aura aucun autre rôle serveur que **public**.

Les bases de données de l'entreprise

Chacun des départements possède sa propre base de données. Les données propres au département des « Ressources humaines **(RSH) »** sont conservées dans la base de données **dbrsh** et celles propres au département « Développement (**DEV**) » dans la base de données **dbdev**.

À leur création, le nom de ces bases de données doit-être précédé de vos initiales (exemple : **SYdbrsh**)

Base de données dbrsh

La base de données **dbrsh** contiendra les 4 tables décrites ci-dessous.

Departements	Employes	Employe_formation	Formations
deptno (pk)	empno (pk)	empno (fk)	idFormation (pk)
nomDepartement	nom	idFormation (fk)	contenu
	prenom	dateDebut	coutMinimum
	adresse	coutReel	coutMaximum
	salaire		
	deptno (fk)	**(empno idFormation) forment une PK**	

Toutes les primary key ont un type **int**.

Les coûts et le salaire ont un type **money.**

La dateDebut a un type **date.**

Les autres colonnes sont des **varchar.**

Le coutReel est contrôlé par un trigger ***CTRLcoutFormation***. Lors de l'inscription d'un employé à une formation, si le coutReel de la formation n'est pas compris entre le coutMinimum et coutMaximum, l'insertion est annulée.

Le salaire des employés est contrôlé par un trigger ***CTRLSalaire*** qui garantit que le salaire ne doit jamais être révisé à la baisse.

Initialement, la table Employes contient tous les employés de l'entreprise.

Initialement, la table Departements contient au moins les départements **RSH** pour ressources humaines et **DEV** pour Développement.

La table **formations** contient les informations suivantes.

idFormation	contenu	coutMinimum	coutMaximum
1	Administration SQL Server	1 200	30 000
2	Introduction à SQL	1 000	12 000
3	PL/SQL	2 000	25 000
4	Transact-SQL	1 800	20 000

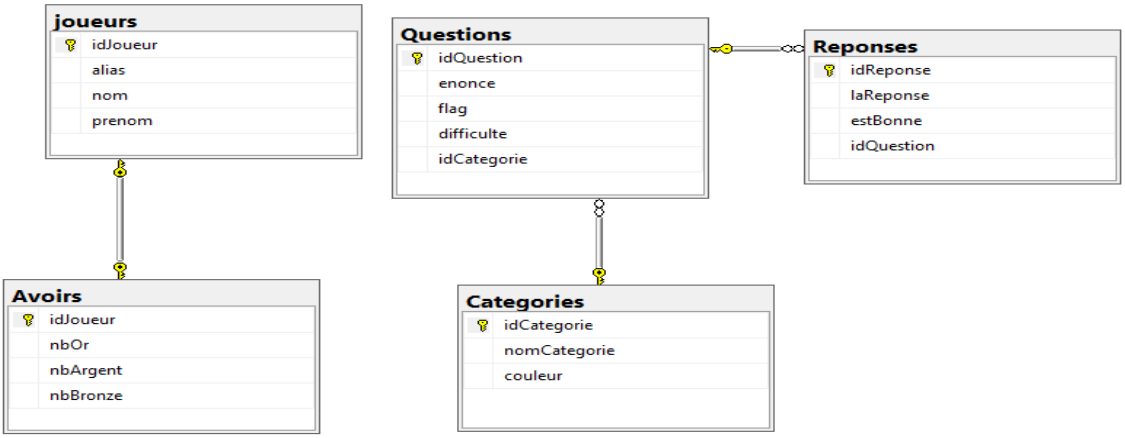
Base de données dbdev

La base de données **dbdev** contiendra les tables du jeu Trival Pursuit (**voir le script)** et ainsi que la table **Avoirs** décrite ci-dessous.

Les joueurs dans la table joueurs sont les employés du département DEV.

Colonnes	Description
idJoueur (pk) et fk	Référence à un joueur de la table joueur
nbOr	Nombre de pièces d'or
nbArgent	Nombre de pièces d'Argent
nbBronze	Nombre de pièces de bronze

Voici la partie du modèle de du jeu Trivial Pursuit qui nous intéresse (la table joueurs est liée à d'autres tables qui ne sont pas sur le modèle)


joueurs
idJoueur
alias
nom
prenom
Questions
idQuestion
enonce
flag
difficulte
idCategorie
Reponses
idReponse
laReponse
estBonne
idQuestion
Avoirs
idJoueur
nbOr
nbArgent
nbBronze
Categories
idCategorie
nomCategorie
couleur

Sécurité et intégrité des données au RSH

Pour respecter les règles de sécurité, l'administrateur (DBA) Mathieu Powers décide de ne pas donner tous les droits à tous les employés. Il décide plutôt d'y aller par attribution de rôles. Pour chaque type d'emploi, un rôle est défini.

Voici un tableau qui résume un peu la structure de la PME concernant les droits et les privilèges.

Roles	Privilèges	Membres
RoleTrivial	SELECT, UPDATE, INSERT sur toutes les tables de la base de données **devdb**.	trivial
RespRSH	SELECT, UPDATE, INSERT sur toutes les tables du département RSH. Également le privilège EXECUTE.	Responsable des RSH
RoleTech	SELECT sur toutes les tables du département RSH, UPDATE de la colonne adresse de la table Employés, UPDATE de description de Formations, INSERT et UPDATE de la table Employe_formation	Techniciens
RoleCommis	SELECT sur toutes les tables du département RSH	Les commis
db_owner	Propriétaire de la base de données du département DEV	Responsable du département Dev
db_datawriter	Peuvent écrire sur la BD du département DEV	Les développeurs
db_datareader	Peuvent lire sur la BD du département DEV	Les développeurs
db_ddladmin	Peuvent exécuter les commandes DDL su la bd du département DEV	Les développeurs
En plus, Les développeurs doivent pouvoir exécuter les procédures stockées de la base de données de leur département		

Tableau 1:Tableau 1: Rôles et autorisations

Ce qu'on vous demande :

1. Créer les bases de données des deux départements, RSH et DEV
2. Créer les tables pour le département RSH, puis insérer les données.
3. Exécuter le script TrivialPursuit.sql pour créer les tables du département DEV et y insérer les données
4. Créer les logins avec les utilisateurs mappés sur les logins dans la base de données correspondante
5. Attribuer les rôles serveur et bases de données s'il y a lieu
6. Créer les RÔLES non prédéfinis du tableau 1
7. Ajouter les membres aux rôles en tenant compte des autorisations.
8. Faire en sorte que les développeurs puissent exécuter les procédures stockées.
9. Programmer le trigger qui contrôle le salaire des employés.
10. Programmer le trigger qui contrôle le coût réel d'une formation.

Développement et sécurité des données au département DEV

Les employés du département DEV voulant se détendre durant leurs heures libres jouent au jeu **Trivial Pursuit**, jeu de questions et réponses que nous avons utilisé pour quelques démonstrations du cours.

A- Système de récompense, et questions réponses

Dans le jeu Trivial Pursuit, chaque question a un niveau de difficulté de 1 à 3 : 1 pour facile, 2 pour moyen et 3 pour difficile. La quantité de pièces obtenues en fonction du niveau de difficulté de la question est affichée dans le tableau ci-dessous.

Niveau de difficulté	Nombre de pièces obtenues
1	5 pièces de bronze
2	10 pièces d'argent
3	15 pièces d'or

Les récompenses ainsi obtenues seront comptabilisées dans la table **Avoirs** (ci-dessus) par la procédure **validerRéponse**.

Ce qu'on vous demande

1. Écrire la procédure stockée **PigerQuestion** qui permet de piger une question de manière aléatoire à partir de la table questions et de mettre le flag de la question à « o » (comme quoi la question a été pigée).

2. Écrire une procédure **ValiderRéponse** qui comptabilise les pièces d'or, d'argent ou de bronze associées à la question. Les récompenses sont comptabilisées dans la table **Avoirs**. En d'autres mots, lorsqu'un joueur (alias) répond correctement à une question (a la bonne réponse) alors la table Avoirs est mise à jour selon le degré de difficulté de la question.

3. Programmer la procédure **AjouterQuestionRéponses** qui permet d'ajouter une question avec ses réponses associées;

4. Pour garantir l'intégrité des données de la table **réponses** et s'assurer qu'il n'existe en tout temps qu'une seule bonne réponse par question, vous devez programmer le trigger **CTRLRéponse.**

5. L'information de la table A**voirs** est confidentielle. Il est donc d'une importance capitale qu'un usager de la base de données du département DEV ait accès uniquement aux données qui le concernent. Ici on suppose qu'un joueur correspond à un usager. Vous devez mettre en place un mécanisme qui garantira:
    a. Qu'un joueur puisse consulter ses avoirs et uniquement les siens
    b. Qu'un joueur puisse mettre à jour (INSERT et UPDATE) ses avoirs et uniquement les siens.

B- Authentification des joueurs et information de crédit

Des projets sont en discussion pour implémenter un système d'achats en ligne pour les joueurs inscrits au jeu. L'aspect sécurité des données devient donc un enjeu important.

Dans un premier temps, vous devez implémenter un mécanisme d'authentification des joueurs à l'aide d'un mot de passe. Le mot de passe des joueurs sera conservé dans la table joueurs et sera chiffré à l'aide d'une fonction de hachage. Vous aurez donc à ajouter deux nouvelles colonnes à la table **joueurs** pour conserver ce mot de passe et les informations de crédit.

De plus, les joueurs auront la possibilité d'associer un numéro de carte de crédit à leur profil dans la table **joueurs**. Votre travail est d'ajouter la fonctionnalité pour pouvoir conserver en toute sécurité le numéro de carte de crédit des joueurs.

Le numéro de carte de crédit sera crypté à l'aide d'un algorithme de chiffrement symétrique. La clé utilisée pour crypter le numéro sera le mot de passe du joueur. Cette façon de procéder garantit qu'uniquement le joueur en possession du mot de passe aura accès à l'information de crédit.

Ce qu'on vous demande

Pour mettre en place cette nouvelle fonctionnalité, vous devez programmer les procédures et fonctions suivantes.

1. Programmez la procédure **AjouterJoueur** qui ajoute un joueur à la table joueurs avec toutes ses informations, incluant son mot de passe.
2. Programmez la procédure **ModifierMotPasse** qui permet à un joueur de modifier son mot de passe. L'ancien mot de passe doit être fourni pour valider son identité.
3. Programmez la fonction scalaire **ValiderIdentité** pour valider l'identité d'un joueur basé sur son mot de passe. La fonction retourne 0 (succès) ou 1 (échec).
4. Programmez la procédure **AjouterInfoCrédit** qui permet d'ajouter un numéro de carte de crédit à un joueur donné. Le mot de passe du joueur doit être fourni.
5. Programmez la fonction table **ObtenirInfoCrédit** qui retourne toute l'information d'un joueur, incluant son numéro de carte de crédit, mais pas son mot de passe.

C- ADO.Net

Vous devez concevoir une petite application (C#/ADO.NET/Form) pour permettre l'ajout de questions et réponses dans la base de données du jeu Trivial Pursuit.

L'interface utilisateur doit permettre de saisir un **alias** et un mot de passe. Cette information servira à authentifier le joueur. Pour authentifier le joueur, vous utiliserez les services que vous avez implémentés dans la section précédente

Seulement après avoir été authentifié avec succès, la fonctionnalité d'ajout de questions/réponses sera activée.

Le logiciel devra utiliser la connexion **trivial** pour s'authentifier au SGBDR.

Archéologues en herbe

Comme petite PME de développement de jeu vidéo, vous avez reçu le mandat de développer le une application Android « Archéologues en herbe ». Le but du jeu est de trouver le maximum d'objets rares sur des sites archéologiques et d'obtenir le classement des joueurs. le meilleur joueur étant celui qui a le plus de points. Les objets d'importance 5 ont 5 points, les objets d'importance 4 ont 4 points, les objets d'importance 3 ont 3 points …

Voici la description du jeu.

Les joueurs doivent trouver des objets rares, archéologiques, dispersés de manière aléatoire sur des sites différents.

Chaque site a numéro, un nom, une brève description, une adresse.

Chaque joueur peut fouiller plusieurs sites. Un site est fouillé par plusieurs joueurs. Un joueur a un alias, un nom et un prénom.

Un objet est trouvé sur un seul site, par un joueur. Évidemment, sur un site, il y a plusieurs objets, et un joueur peut trouver plusieurs objets.

Un objet a un numéro, un nom, un état (bon, à restaurer), type (squelette ou accessoire), importance de la découverte (1 à 5). 5 veut dire découverte majeure.

Ce qu'on vous demande

Donner le modèle relationnel de votre jeu « Archéologues en herbe », en format pdf, jpg ou png. Peu importe le format que vous choisissez, assurez-vous que tout est lisible (c'est correct qu'on doive zoomer), et que tout est sur une seule page (on ne veut pas devoir recoller des pages côte à côte pour recréer le schéma complet).

Ce que vous devez remettre :

1. Un fichier SQL portant votre nom, identifié à l'intérieur par votre nom et contenant :
   - La création des BD
   - La création des tables
   - Les insertions initiales
   - Les requêtes SQL bien identifiées par leurs numéros et la section.
   - Les triggers identifiés incluant les tests d'exécution
   - Les procédures (fonctions) stockées identifiées par leurs noms et la section. Incluant les tests d'exécution

2. Un projet Visual Studio (ADO.NET/Form)
3. Le modèle de la base de données : « Archéologues en herbe »

Modalité de remise : Boite de remise

Vous allez remettre trois fichiers clairement identifiés par VOS NOMS

- Le modèle de la BD de « Archéologues en herbe » en pdf ou image.
- Le projet ADO.NET, zippé
- Un fichier SQL.

Barème :

Éléments évalués	Sur
Sécurité et intégrité des données au RSH	30
Développement et sécurité des données au département DEV	50 (total = 20+20+10)
A- Système de récompenses et questions réponses	20
B- Authentification des joueurs et information de crédit	20
C- ADO.NET	10
Le modèle « Archéologues en herbe »	10
Respect des consignes de remise	10
Total	100