École Polytechnique de Montréal
Département Génie Informatique et Génie Logiciel
INF3710 – Fichiers et Bases de données

TP 4 – Applications et Bases de données

Objectif:

1. Informations générales

Pondération	**20%**
Taille de l'équipe	**2 personnes**

Notez bien:
1. Tout retard dans la remise du TP entraîne automatiquement une pénalité comme discuté dans le plan de cours.
2. Aucun TP ne sera corrigé, s'il est soumis par une équipe dont la taille est différente **de deux (2) étudiants** sans l'approbation préalable du chargé de laboratoire. Cette approbation ne sera accordée qu'à une seule équipe et qu'en cas de nombre impair d'étudiants dans le laboratoire. Sinon, la note de zéro sera attribuée aux étudiants concernés.
3. Soumission du TP par **Moodle** uniquement (https://moodle.polymtl.ca). Aucune soumission "hors **Moodle**" ne sera corrigée. La note de zéro) sera attribuée aux étudiants concernés.

2. Évaluation

Rubriques	**Points**
Clarté et présentation du rapport	10
Réponses aux questions (voir le détail)	90
Total	100

Critères d'évaluation:
1. Qualité de la modélisation et de la BD correspondante
2. Qualité du code SQL
3. Qualité du code de l'application, avec des critères de qualité semblables à ceux qui sont requis en projet 2
4. Exécution fluide de l'application avec gestion de toutes les exceptions (on ne devrait pas avoir à redémarrer l'application). Attention, une pénalité allant jusqu'à 15 points pourrait être appliquée si ce n'est pas le cas.
5. Ergonomie de l'interface
6. Professionnalisme du rapport

3. Environnement et outils nécessaires

Vous devez utiliser :
- Le logiciel de votre choix pour le design UML
- Le SGBD PostgreSQL
- Angular (version 4 et +) et NodeJS pour l'application Web.

4. Etude de cas

Considérez une plateforme de location de vidéos en ligne *Netflix_Poly*. Cette plateforme permet à ses membres d'acheter et/ou de télécharger des vidéos.

Pour chaque membre, on veut conserver une adresse courriel, un mot de passe, un nom, une adresse postale ainsi que des informations de carte de crédit (titulaire, numéro, date d'expiration, CCV). Notons qu'on peut avoir plusieurs cartes de crédit pour un membre donné. Il existe deux catégories de membres distinctes: les membres qui possèdent un abonnement mensuel (on conserve alors le prix de l'abonnement et la date de début et d'échéance de l'abonnement) et les membres qui effectuent un paiement à la vue (*pay per view*). Dans ce cas, on veut alors conserver le nombre de films vus dans un attribut *film_payperview*.

Le mot de passe d'un membre ne doit pas être stocké tel quel dans la base de données. *Netflix_Poly* vous demande de l'encrypter avant de l'enregistrer.

Les films sont décrits par un numéro, un titre, un genre, une date de production et une durée totale en minutes.

Pour tout film, on veut savoir quelles sont toutes les personnes qui ont participé au film ainsi que leur rôle (réalisateur, producteur, acteur, maquilleur, metteur en scène, etc.). Notons qu'on veut garder des informations sur ces différentes personnes telles que leur nom, leur âge, leur sexe, ou leur nationalité. On veut aussi savoir le salaire reçu par chaque personne dans un rôle donné. Notons qu'une même personne peut endosser plusieurs rôles dans un même film ou dans des films différents. Notons également qu'une même personne peut endosser plusieurs rôles au cours de sa carrière. Par exemple, Angelina Jolie a été actrice dans « *Salt* » et réalisatrice dans « *D'abord ils ont tué mon père* ».

On veut également savoir si un film a été nominé ou a gagné aux Oscars, et dans quelle catégorie (Exemple : meilleur réalisateur, meilleur acteur masculin, etc.), et pour chaque cérémonie d'oscars, on veut se souvenir de la date, du maitre de cérémonie et du lieu de la cérémonie. Notons qu'un même film peut être nominé ou gagner plusieurs oscars dans différents rôles.

Certains films peuvent avoir des DVD physiques qui leur sont associés. Notons que le numéro d'un DVD n'est pas unique par film et que l'on peut avoir par exemple un Film F1 avec DVD1 et F2 avec DVD1 aussi. Ainsi les membres peuvent choisir d'acheter un DVD qui leur est envoyé à leur adresse postale, soit choisir de regarder un film en ligne. Dans ce dernier cas, on conserve la date de visionnement, et la durée de visionnement. En effet, en cas de visionnage en ligne, on veut pouvoir relancer le film à l'endroit même où il s'est arrêté.

Les coût d'envoi d'un DVD sont calculés en fonction de la distance de l'adresse du client. *Netflix_poly* calcule donc, à partir de l'adresse du Pavillon Lassonde, le nombre de Kms à parcourir par le DVD pour se rendre au domicile du membre. Chaque km coûte 25 cents. On peut donc ensuite connaitre le coût d'envoi d'un DVD en dollars et le consigner dans la commande du client, ainsi que la date d'envoi du DVD. Il faut ainsi être capable de retrouver toutes les commandes d'un membre, qu'elles concernent l'envoi d'un DVD ou le visionnement d'un film.

5. Travail à faire (100 points)

1) Lisez attentivement l'étude de cas. Etablissez vos hypothèses. Discutez-en avec votre chargé de laboratoire. Assurez-vous de bien modéliser ce qui est demandé.
2) Proposez un modèle entités-associations (ou entités associations étendu) permettant de répondre aux besoins exprimés ci-dessus et en tenant compte des requêtes auxquelles vous devez pouvoir répondre. N'oubliez aucun composant du modèle. Utilisez le logiciel de votre choix pour créer le modèle en notation UML. **(10 points)**
3) Créez la base de données PostgreSQL correspondante. Indiquez vos clés primaires et étrangères. N'oubliez aucune contrainte nécessaire dans votre modèle (exemple : intégrité référentielle, valeurs non nulles, etc.). Notez que votre script doit être fait à la main et non généré avec PGADMIN. Enregistrez votre code SQL dans **bdschema.sql. (10 points)**
4) Entrez des données dans la base de données et enregistrez vos données dans **data.sql.** Notez que votre script doit être fait à la main et non généré avec PGADMIN. Assurez-vous d'avoir des données qui permettent de répondre aux requêtes tel qu'indiqué dans la liste des requêtes à implanter. **(5 points)**
5) Créez les requêtes ci-dessous et enregistrez-les dans un fichier **query.sql.** Notez que chaque requête SQL doit être précédée par un commentaire contenant la requête en Français. **(20 points)**
6) Exprimez en algèbre relationnelle les requêtes 1, 2 et 5. **(5 points)**
7) Créez un trigger dans un fichier *trigger.sql* qui calcule le coût d'envoi d'un DVD à une adresse donnée. Ce trigger doit automatiquement insérer le coût adéquat lors de l'envoi du DVD et le stockage de l'information correspondante dans la base de données. Ici vous avez le choix d'intégrer une api telle que Google maps (c'est la solution idéale) ou de calculer un nombre de kms « standards » en fonction de codes postaux prédéterminés. **(5 points)**
8) Créez une application Web pour que l'usager puisse directement interroger la base de données (voir la section Application Web). **(35 points)**
9) Un rapport nommé *matricule1_matricule2_TP4.pdf*. Voir la section *Rapport*. **(10 points)**

Liste des requêtes à implanter

1) Affichez toutes les informations sur un film spécifié par l'utilisateur (selon le titre)
2) Pour chaque genre de film, listez tous les titres de films ainsi que la dernière date à laquelle un film a été acheté (DVD) ou visionné
3) Pour chaque genre de film, trouvez les noms et courriels des membres qui les ont visionnés le plus souvent. Par exemple, Amal Z est le membre qui a visionné le plus de documentaires animaliers
4) Trouvez le nombre total de films groupés par réalisateur

5) Trouvez les noms des membres dont le coût total d'achat de DVD est plus élevé que la moyenne
6) Ordonnez et retournez les films en termes de quantité totale vendue (DVD) et en nombre de visionnements
7) Trouvez le titre et le prix des films qui n'ont jamais été commandés sous forme de DVD mais qui ont été visionnés plus de 10 fois
8) Trouvez le nom et date de naissance des acteurs qui jouent dans les films qui sont visionnés le plus souvent (soit plus que la moyenne)
9) Trouvez le nom du ou des réalisateurs qui ont réalisé les films qui ont le plus grand nombre de nominations aux oscars. Par exemple, Woody Allen et Steven Spielberg ont réalisé 10 films qui ont été nominés aux oscars.
10) Trouvez le nom des réalisateurs qui ont été le plus souvent nominés aux oscars mais qui n'ont jamais gagné d'oscar
11) Trouvez les films (titre, année) qui ont gagné le plus d'oscars. Listez également leur réalisateurs et leurs acteurs ;
12) Quelles paires de femmes québécoises ont le plus souvent travaillé ensemble dans différents films ?
13) Comment a évolué la carrière de *Woody Allen* ? (On veut connaitre tous ses rôles dans un film (réalisateur, acteur, etc.) du plus ancien au plus récent)

Application Web

Vous devez programmer une application Web en utilisant le stack Node et Angular. Votre application Web doit permettre, au moyen d'une interface, d'insérer, supprimer, modifier et d'interroger les données de votre base de données. En particulier, vous devez permettre :

1) Des fonctions d'administration **(20 points)** :
   a. L'enregistrement d'un nouveau membre (**5 points**)
   b. L'ajout, suppression et modification d'un nouveau film (et les informations associées) (**15 points**)
2) Des fonctions pour un membre donné (le membre doit pouvoir se loguer dans le système) (**15 points**)
   a. L'affichage de toutes les informations sur un film spécifié par l'utilisateur. Autrement dit, l'utilisateur doit sélectionner le titre d'un film dans une liste, et les informations doivent ensuite être affichées à l'écran; (**5 points**)
   b. De visionner un film en tant que membre, soit à partir du début soit en reprenant à l'endroit où le membre s'est arrêté. Cela implique de se loguer, d'avoir une interface de choix de films et ensuite de visionner le film du début ou de le continuer (**10 points**).

Rapport

Votre rapport doit contenir les informations suivantes :
- La page de garde
- Une brève introduction résumant le projet
- Le modèle conceptuel UML incluant les hypothèses et commentaires si nécessaire
- Le modèle relationnel en syntaxe abstraite

- Les requêtes en algèbre relationnelle
- La présentation de l'application développée avec des copies d'écran permettant de démontrer toutes ses fonctionnalités. N'hésitez pas à soulignez les aspects novateurs de votre application tel que l'utilisation de certains patrons de conceptions, ou un effort d'ergonomie, etc.
- Un guide d'installation et de configuration qui permette au chargé de laboratoire d'installer et d'exécuter votre application

Informations supplémentaires

- Notez que les projets doivent être faits en groupes de deux. Les projets ne respectant pas cette condition ne seront pas acceptés et mèneront à une note de 0.
- Il est fortement conseillé de compléter les étapes de la base de données et des requêtes le plus tôt possible. L'application **vous prendra du temps** et vous devrez compter sur votre effort personnel et votre expérience en programmation pour la compléter.
- Notez bien que le chargé de laboratoire doit être capable d'installer et d'exécuter votre application sans problèmes en suivant le guide d'installation et d'utilisation que vous préparerez (voir section Rapport).
- Notez également que votre application doit aller ajouter/modifier/chercher les données via des requêtes SQL appropriées. En aucun cas vous ne devez effectuer des traitements sur les données au niveau de l'application (exemple en utilisant des filtres Javascript).

Description des livrables à la fin de la session

1) Un modèle conceptuel fait avec un logiciel de votre choix et présenté sous forme d'image jpeg présentée dans le rapport
2) Un modèle relationnel dans un fichier bdschema.sql qui permet de créer votre base de données
3) Un fichier data.sql qui ajoute des données à votre BD avec des instructions INSERT (suffisamment pour qu'il y ait au moins deux tuples dans les réponses aux requêtes)
4) Un fichier query.sql qui rassemble le code SQL de l'ensemble des requêtes. Notez que chaque requête doit être précédée par un commentaire indiquant le texte de la requête (référez-vous à la liste des requêtes)
5) Un fichier trigger.sql
6) Le code de votre application Web, soit :
   a. Le code de votre serveur Node dans un répertoire server
   b. Le code de votre client Angular dans un répertoire client
   c. Un ReadMe pour installer et lancer votre application Web
7) Le rapport matricule1_matricule2_TP4.pdf

Modalités de remise

Vous devez soumettre sur Moodle à la fin du cours (la date exacte sera spécifiée sur Moodle) un fichier nommé `matricule1_matricule2_TP4.zip` qui contient tous les livrables demandés.