Bases de données 2 : Projet Partie 2

Alexandre DUBERT et Hayk ZARIKIAN

1 Introduction

Ce rapport est dédié à la seconde partie de notre projet de bases de données 2 dans le cadre de la troisième année de licence d'Informatique à l'UFR de Mathématique et d'Informatique à l'Université de Strasbourg. Pour la réalisation de cette deuxième partie, nous avons décidé de travailler en binôme en répartissant les tâches au préalable. Nous avons utilisé le modèle fournit par défaut afin de mieux s'accorder sur les questions posées.

2 Requêtes SQL

2.1 Introduction

Dans la réalisation des requêtes SQL, nous avons essayé d'être le plus simple possible et d'avoir le code le plus compréhensible possible. Il y a surement des milliers de façon d'écrire une requête mais dès la réalisation d'une requête, nous passions en revue le code afin de le rendre plus simple jusqu'à ne plus avoir d'idée de simplification. Nous avons aussi décidé de ne pas utiliser d'index car les requêtes sont déjà assez minimes et optimisées, l'utilisation d'index dans ce genre de cas ne serait pas judicieuse.

2.2 Requête 3 : Les utilisateurs n'ayant jamais mis une note inférieure à 8 à un objet culturel.

Pour la troisième requête, nous avons pris le problème à l'envers, plutôt que de sélectionner les utilisateurs n'ayant jamais mis de note inférieure à 8, nous avons pris ceux qui n'ont mis que des notes supérieures ou égales à 8, ce qui est équivalent.

2.3 Requête 4 : L'objet culturel le plus commenté au cours de la dernière semaine.

Quant à la quatrième requête, nous avions deux possibilités, d'une part sélectionner l'objet dont le compte est égal au compte maximum de la table, et d'autre part trier en fonction du compte et sélectionner uniquement la première ligne. Ces deux requêtes donnent car l'algorithme de recherche du maximum est moins coûteux que celui du tri.

2.4 Requête 5 : Pour chaque utilisateur ayant au moins noté des objets sur 3 mois consécutifs au cours de l'année précédente (par exemple, une note a été ajoutée en juin, en juillet et en aout), indiquer son nombre d'objets possédés, son nombre d'objets à acheter, la longueur minimale, maximale et moyenne de ses collections.

Enfin, la dernière requête est une jointure de quatre tables : la table temporaire des utilisateurs ayant émis un avis au cours de 3 mois consécutifs, la table temporaire de la taille des listes de chaque utilisateur, la table `possede` et la table `prévoit_achat`. Cette dernière est jointe par un `LEFT OUTER JOIN` car l'intersection avec `possede` enlèverait des objets de la requête.

3 Procédures et fonctions PL/SQL

3.1 Procédure 2 : favoris

Pour la procédure `favoris`, nous avons choisi de créer un curseur paramétré qui sélectionne les 10 œuvres les mieux noté de l'utilisateur pour la catégorie en paramètre. Ensuite pour chaque catégorie, nous vérifions qu'elle contient au moins dix avis, le cas échéant, nous parcourons le curseur avec la catégorie en argument et nous affichons chaque ligne.

3.2 Procédure 3 : suggestion

La procédure `suggestion` prend quatre paramètres ; l'id de l'utilisateur, l'entier x de résultats à afficher, l'entier y d'objets communs aux utilisateurs sélectionnés et z le nombre de notes communes à avoir avec l'utilisateur en paramètre pour être sélectionné. Le curseur est construit en 4 étapes. Premièrement, nous sélectionnons les utilisateurs ayant mis au moins z notes identiques que l'utilisateur id. Deuxièmement, nous sélectionnons parmi ces utilisateurs les objets qu'ils sont au moins y à posséder et que l'utilisateur id ne possède pas. Troisièmement, nous trions ces objets par leur moyenne croissante. Enfin dernièrement, nous sélectionnons les x premiers objets. La suite de la procédure est juste un parcours classique de curseur pour l'affichage des résultats.

4 Déclencheurs

4.1 Déclencheur 1 : liste_du_mois

Le déclencheur `liste_du_mois` doit, lors de la création d'un objet, ajouter cet objet dans une liste des nouveautés du mois courant. Cette liste est créée par un faux utilisateur appelé « Nouveauté ». Il faut donc vérifier si la liste existe, le cas contraire la créer à l'aide de l'utilisateur « Nouveauté », et ensuite insérer l'objet dans la liste. Ce déclencheur a causé des problèmes avec la contrainte dynamique d'insertion dans la table `appartient_a`, il a donc fallu modifier la contrainte pour qu'elle ne tienne pas compte des insertions dans cette table par l'utilisateur « Nouveauté ».

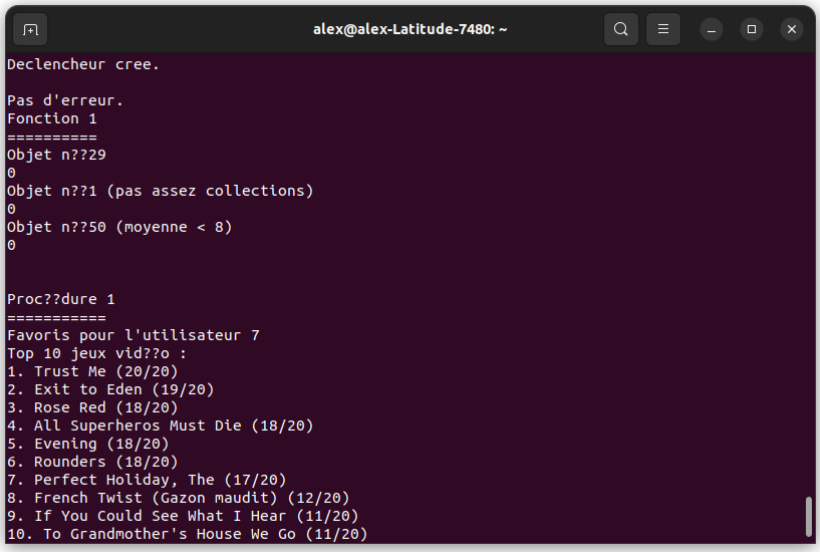
4.2 Déclencheur 2 : archivage

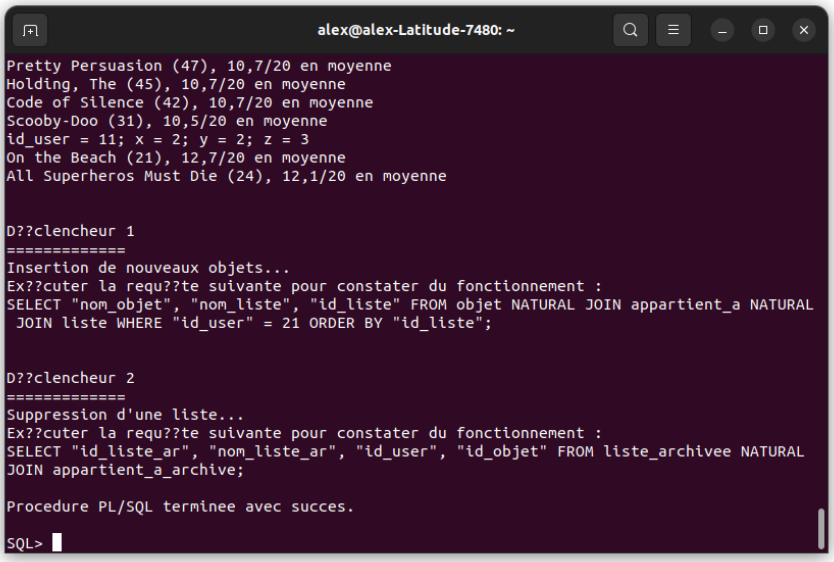
Le déclencheur `archivage` insère dans la `liste_archivee` les lignes supprimées lors de la suppression dans `liste`. Un deuxième déclencheur se déclenche à l'insertion dans `liste_archivee`, un curseur parcours tous les objets appartenant à liste et les insère dans `appartient_a_archive`. Après le parcours de toutes les lignes correspondant à la liste supprimée dans la table `appartient_a` sont supprimées. Nous avons utilisé deux déclencheurs car nous pensions, à tort, que faire le tout dans un seul déclencheur provoquait une erreur ORA-04091 (erreur de la table mutante). En fait celle-ci était provoqué par le `ON DELETE CASCADE` dans la définition de table. Mais cette façon de faire est finalement mieux car si l'on supprime une ligne de `appartient_a` car elle a été insérée par erreur, cette ligne n'est pas archivée. Ne sont archivée que les objets appartenant à une liste supprimée.

5 Scripts et Tests

Pour avoir une bonne vue sur les erreurs et résultats, nous avons utilisé le fichier « Utilisateurs_oracle.sql » accompagné du SGBD « SQL Server » ou bien la connexion à « sqlplus » via « turing » afin de créer les tables correspondant au modèle EA. Nous avons également inséré un peu plus de 1500 lignes dans les tables à l'aide de l'outil Mockaroo. Chaque requêtes, fonctions, procédures, déclencheurs ont été testé avec soin avant de valider le code. Nous fournirons également un fichier de test nommé « test.sql » qui permettra de créer les tables, de faire les insertions puis de tester les fonctions, procédures et déclencheurs PL/SQL.

Vous trouverez ci-dessous, les captures d'écrans du résultat du fichier « test.sql »





6 Conclusion

Pour en conclure, ce projet nous a beaucoup apporté sur deux plans. Premièrement sur le plan de la pratique dans la réalisation de bases de données en langage PL/SQL (Oracle), la compréhension d'un modèle entité-association, l'écriture de requêtes, déclencheurs, … Deuxièmement sur le plan du travail d'équipe. La communication été très bonne. Nous nous sommes mutuellement données des idées et conseillés sur des solutions, difficultés ce qui nous a permit d'être efficace. Finalement, ce projet a été très plaisant à mener à bien et nous avons beaucoup progressé dans le domaine des bases de données.