Université de Caen Normandie
Département de mathématiques et informatique
Licence informatique L3 - 2019-2020

Bases de données - TP "Gestion de transactions"

Exercice 1 - Accès concurrentiels

Préambule : le département informatique de l'INSA de Rennes est l'auteur d'une part importante de cet exercice.

L'objectif de cet exercice est de se familiariser avec la notion d'accès concurrentiels à une base de données.

Soit le schéma de la base de données BUVEUR :

```
Bar (Bar, Nom, Adresse)
Sert (Bar, Biere, Quantite)
Frequente (Buveur, Bar)
Aime (Buveur, Biere)
```

Ces relations signifient :

- un n-uplet `(ba,n,a)` de la relation `Bar` signifie que le bar `ba` s'appelle `n` et se trouve à l'adresse `a` (on prendra `int` pour le type de `ba`).
- un n-uplet `(ba,bi,q)` de la relation `Sert` signifie que le bar `ba` sert la bière `bi` et possède celle-ci en `q` exemplaires.
- un n-uplet `(bu,ba)` de la relation `Frequente` signifie que le buveur `bu` fréquente le bar `ba`.
- un n-uplet `(bu,bi)` de la relation `Aime` signifie que le buveur `bu` aime la bière `bi`.

Remarque : on ne s'intéresse pas ici aux caractéristiques propres aux buveurs et aux bières, c'est pourquoi le schéma de la base ne comporte pas de relations `Buveur` et `Bière`.

Partie 1 - Création de la base

1. Complétez les schémas des tables de cette base. Incorporez des contraintes d'intégrité dans la définition du schéma (clés primaires et étrangères, la quantité de bière disponible dans un bar est forcément une valeur positive).
2. Chargez les extensions de ces relations qui sont disponibles sur la plate-forme en ligne. Pour cela vous utiliserez la commande `\copy` (les valeurs des attributs d'un n-uplet sont séparées par des virgules). Par exemple :

   ```
   user_bd=> \copy bar FROM 'bar.txt' WITH DELIMITER ','
   ```

Partie 2 - Accès concurrentiels

Dans cette partie, deux utilisateurs appelés Tic et Tac effectuent des requêtes sur la base Buveur de Tac (base nommée `tac_bd`).

Question 1 - Accès concurrents en lecture/écriture.

Exécutez les requêtes suivantes :

- Tic essaye de se connecter [1] à la base de Tac :
  `psql -h postgresql.info.unicaen.fr -d tac_bd -U tic`
  quel message d'erreur s'affiche ?
- Tac : autorise Tic à se connecter à sa base (`tac_bd`).
- Tic se connecte à la base de Tac et exécute `\d` : que constatez-vous ? Pourquoi ?
- Tac : autorise Tic à connaitre le schéma "public" de sa base. "public" est le schéma `Postgres` où sont stockées par défaut les tables.
- Tic exécute `\d sert`
  Remarquez que le nom de ce schéma est affiché avant `sert`
- Tic tape une commande `SQL` comme par exemple :
  ```
  SELECT * FROM sert ;
  ```
  Que constatez-vous ?
- Tac : donne tous les droits à Tic sur les tables `bar`, `sert`, `frequente` et `aime`.
- Tac veut écrire une transaction qui ajoute un buveur avec ses fréquentations et ses bières aimées (respectez les contraintes d'ordre d'ajout). Pour pallier le fait que `PostgreSQL` exécute par défaut un `COMMIT` après chaque instruction d'accès aux tables, débutez la transaction par un `BEGIN` (n'oubliez pas le "`;`" après `BEGIN`) et terminez-là par un `COMMIT`. Une fois Tac connecté à sa base et Tic connecté à la base de Tac, le scénario est le suivant :
  - Tic et Tac : visualisez les tables `bar`, `sert`, `frequente` et `aime` de Tac.
  - Tac : tapez le début de la transaction sans aller jusqu'à l'ordre `COMMIT`
  - Tic : visualisez les tables de Tac : que constatez-vous ?
  - Tac : tapez la fin de la transaction, dont l'ordre `COMMIT`
  - Tic : visualisez les tables de Tac : que constatez-vous ?

Question 2 - Accès concurrents en écriture/écriture.

Exécutez les requêtes suivantes (toujours sous la base de Tac) :

- Tac : lancez une transaction commençant par `BEGIN`, puis modifiez l'adresse du bar numéro 3.
- Tic : lancez une transaction commençant par `BEGIN`, puis modifiez l'adresse du bar numéro 2.
- Tac : continuez la même transaction en modifiant l'adresse du bar numéro 1.
- Tic : continuez votre propre transaction en modifiant aussi l'adresse du bar numéro 1.
- Tac : validez votre transaction par `COMMIT`
- Tic : validez votre transaction par `COMMIT`

Que constatez-vous ?

1. le paramètre `-h postgresql.info.unicaen.fr` est ici facultatif

Question 3 - Interblocage avec verrous implicites.

Exécutez les requêtes suivantes (toujours sous la base de Tac) :

- Tac : lancez une transaction commençant par `BEGIN` et supprimez l'information que le buveur `Dupin` aime la bière `Kro`.
- Tic : lancez une transaction commençant par `BEGIN` et supprimez l'information que le buveur `Dupin` fréquente le bar numéro 1.
- Tac : continuez la même transaction en supprimant l'information que le buveur `Dupin` fréquente le bar numéro 1.
- Tic : continuez la même transaction en supprimant l'information que le buveur `Dupin` aime la bière `Kro`.
- Tac : validez votre transaction par `COMMIT`
- Tic : visualisez la relation `frequente` pour voir les modifications

Que constatez-vous ? Quelle est la raison de ce comportement et comment retrouver un fonctionnement "normal" pour Tic ?

Question 4 - Interblocage avec verrous explicites.

Provoquez un interblocage en exécutant les requêtes suivantes (toujours sous la base de Tac) :

- Tac :
  ```
  BEGIN ;
  LOCK TABLE sert IN EXCLUSIVE MODE ;
  ```
- Tic :
  ```
  BEGIN ;
  LOCK TABLE aime IN EXCLUSIVE MODE ;
  ```
- Tac :
  ```
  LOCK TABLE aime IN EXCLUSIVE MODE ;
  ```
- Tic :
  ```
  LOCK TABLE sert IN EXCLUSIVE MODE ;
  ```
- Tac : validez votre transaction par `COMMIT`
- Tic : visualisez la relation `aime` pour voir si vous pouvez y accéder.

Que constatez-vous ? Quelle est la raison de ce comportement et comment retrouver un fonctionnement "normal" pour Tic ?

Question 5 - Retirer les droits.

- Tac : retirez les droits pour Tic de connaitre le schéma "public" de la base de Tac
- Tac : retirez les droits pour Tic de se connecter à la base de Tac

Exercice 2 - Requêtes, mises à jour et suppressions

Exprimez chacune des requêtes suivantes par une instruction SQL.

1. Buveurs qui fréquentent au moins un bar où l'on sert une bière qu'ils aiment.
2. Buveurs qui ne fréquentent aucun bar où l'ont sert une bière qu'ils aiment. Remarque : si un n-uplet `(bu,bi)` n'est pas présent dans la table `Aime`, cela signifie implicitement que le buveur `bu` n'aime pas la bière `bi`.
3. Buveurs fréquentant uniquement les bars qui servent une bière qu'ils aiment.
4. Les buveurs qui fréquentent au moins 2 bars où l'on sert une bière qu'ils aiment.
5. Diminuez de 50 la quantité de bière "Grimbergen" pour le bar numéro 3. Que constatez-vous ?
6. Changez le nom du bar numéro 3.
7. Pour ce bar, doublez toutes les quantités en stock.
8. Supprimez, pour le bar numéro 3 les bières que Dupin n'aime pas.