Les bases de données

Séance 10

Les transactions

Sommaire

- Notion de transaction
- ACID : les quatre propriétés d'une transaction
- Mise en œuvre d'une transaction
- Gestion d'accès concurrentiels

Notion de transaction

- Une transaction est ensemble ordonné d'opérations modifiant des données (LMD) qu'un SGBD effectuera parfaitement et complètement ou pas du tout.
- Une opération d’une transaction est donc au choix un ordre de type INSERT, UPDATE ou DELETE

Exemple

On considère le MLD suivant :

Facture(id,#*idProduit*,#*IdClient*,Qte,DateFact,montant)

Produit(id, stock, prixUnitaire)

Client(id, nom, prenom, caCumule)

Une vente d'un produit nécessite :

- Une insertion dans la table Facture
- Une mise à jour de la table Produit
- Une mise à jour de la table Client

Une vente est cohérente si les trois opérations sont effectuées.

ACID

Une transaction vérifie les quatre propriétés suivantes :

- **A**tomicité : une transaction doit soit être complètement validée ou complètement annulée.
- **C**ohérence : une transaction ne peut pas laisser la base de données dans un état incohérent.
- **I**solation : une transaction ne peut voir aucune autre transaction en cours d'exécution.
- **D**urabilité : après que le client a été informé du succès de la transaction, les résultats de celle-ci ne disparaîtront pas.

Tous les SGBD Relationnels possèdent ces quatre propriétés de façon native.

SGBD, Instance et Session

- Un **SGBD** est un logiciel permettant de gérer des bases de données.
- Une **instance** d'une BDD est un ensemble de ressources mémoires et logicielles qui permet de rendre accessible une BDD à plusieurs utilisateurs.
- Une **session** permet à un utilisateur de se connecter à une instance d'une BDD pour exécuter des instructions SQL

Détail d'une transaction

- Une transaction commence :
  - Soit au début d'une session
  - Soit à la fin d'une transaction
- (**Atomicité**) Une transaction se termine :
  - Soit avec l'ordre **commit** qui valide définitivement tous les opérations de la transaction
  - Soit avec l'ordre **rollback** qui annule tous les opérations de la transaction

Fin d’une session

- La fin d’une session provoque la fin de la transaction courante.
- Il faut se conférer à la documentation de l’utilitaire pour savoir lequel des deux ordre commit ou rollback est exécuté.
- En général, c’est l’ordre commit.

SGBD et Cohérence

- Le SGBD garantit la cohérence de la base de données même en cas de panne du serveur.
- Si la panne se produit pendant la validation ou l'annulation d'une transaction, lors du redémarrage la fin de la transaction est effectuée si cela est possible ou la BDD est remis dans l'état avant la transaction.

SGBD et Isolation

Lors de l'exécution d'un ordre SELECT

- Le SGBD affiche :
  - Toutes les données déjà validées lors de transactions précédentes
  - Les données créées ou mises à jour dans la transaction courante
  - Les données **détruites** lors d'**autres transactions en cours**

SGBD et Isolation

Lors de l'exécution d'un ordre SELECT

- Le SGBD **n'affiche pas**
  - Les données détruites définitivement lors de transactions précédentes et validées
  - Les données détruites lors de la transaction courante
  - Les données créées ou mises à jour dans les autres transactions en cours.

SGBD et Isolation renforcée

- On peut créer une transaction en lecture seule et consistante
- Cela signifie deux choses :
  - On ne peut faire que des lectures
  - Toute validation d'autres transactions pendant cette transaction n'a aucun effet sur les lectures de la transaction courante.
- Pour donner cette propriété à une transaction, il faut exécuter au début de la transaction **SET TRANSACTION READ ONLY**

Découpage d'une transaction

- On peut poser des jalons dans une transaction avec l'ordre **savepoint** nom_point
- On peut annuler un sous ensemble d'opérations avec l'ordre **rollback to savepoint** nom_point. Toutes les opérations créées depuis le jalon sont annulées.
- Un jalon ne disparaît qu'à la fin de la transaction

Gestion des accès concurrentiels

- Le principe de la transaction crée un nouveau problème :

  Comment gérer la mise à jour d'une donnée dans deux transactions en cours ?

- Tant qu'une transaction n'est pas terminée, l'instance du SGBD ne sait pas si elle sera validée ou annulée.

Exemple

On considère la table suivante :

```
CREATE TABLE nombre (n number);
INSERT INTO nombre VALUES (1); COMMIT;
```

```
Transaction 1 : T1 -- au temps t1
UPDATE nombre
SET n = 2
WHERE n = 1;
```

```
Transaction 2 : T2 --au temps t2
UPDATE nombre
SET n = 4
WHERE n = 1;
```

L'ordre de la transaction 2 n'a pas le même sens suivant que la transaction 1 est validée ou annulée. Si T1 est validée, l'ordre de la transaction 2 ne concerne aucune ligne et l'unique élément de la table vaut 2 sinon il vaut 4

Verrouillage et blocage

- Chaque mise à jour d'une ligne d'une table provoque le verrouillage de la ligne.
- Le verrou est annulé à la fin de la transaction qui a posé le verrou.
- Chaque mise à jour d'une ligne verrouillée provoque le blocage temporaire de la transaction.
- Le blocage d'une transaction est annulé à la fin de la transaction qui a verrouillé la ligne

Interblocage : Problème

T1 met à jour L1 : L1 est verrouillée
T2 met à jour L2 : L2 est verrouillée
T1 met à jour L2 : T1 est bloquée
T2 met à jour L1 : T2 est bloquée

Nous sommes en présence d'un deadlock (étreinte fatale).

T1 attend la fin de T2 et T2 attend la fin de T1

Interblocage : Solution

T1 met à jour L1 : L1 est verrouillée

T2 met à jour L2 : L2 est verrouillée

T1 met à jour L2 : T1 est bloquée

T2 met à jour L1 :

1. T2 est bloquée
2. La mise à jour de L2 par T1 est annulée et T1 est débloquée

Le problème est résolu mais au prix d'une annulation d'une opération.

OLTP

- **OLTP** : *Online Transaction Processing*
- **OLTP** est une architecture de programme permettant de gérer des transactions en temps réel et leurs problèmes associés.
- Des applications de réservation de billets de train ou de bourse nécessitent ce genre d'architecture.
- Tous les SGBD Relationnels dignes de ce nom ont une architecture **OLTP**.

FIN DU COURS