Gestion des transactions

Année 2023-2024

La notion de transaction

- Unité logique de traitement qui est :
  - soit complètement exécutée
  - soit complètement abandonnée
- Une transaction est une unité atomique de traitement
- Une transaction fait passer la base de données d'un état cohérent à un autre état cohérent
- Si une transaction ne va pas à son terme pour une raison ou pour une autre, la base est restaurée dans l'état où elle se trouvait avant que la transaction ne démarre

Exemple

- le transfert d'une somme S d’un compte C1 vers un compte C2
  - (1) début-transaction
  - (2) lire C1
  - (3) C1 := C1 - S
  - (4) écrire C1
  - (5) lire C2
  - (6) C2 := C2 + S
  - (7) écrire C2
  - (8) fin-transaction
- Cette transaction est constituée d’un ensemble d’actions élémentaires, mais elle doit être traitée comme une seule opération.
- Autrement dit le gestionnaire des transactions doit assurer que **toutes** les actions de la transaction sont exécutées, ou bien qu’**aucune** ne l’est.

La vie d'une transaction

- **COMMIT**
  - La transaction s'exécute normalement jusqu'à la fin. Elle se termine par une instruction de validation **COMMIT** en SQL. Nous dirons que cette transaction est ***validée***.
  - Toutes les modifications faites sur la base par cette transaction sont considérées comme définitives.
- **Assassinat**
  - Un événement extérieur vient interrompre l'exécution de la transaction de façon irrémédiable (panne ou une action délibérée de la part du SGBD qui décide de supprimer telle ou telle transaction)
- **ROLLBACK**
  - Au cours de son exécution la transaction détecte certaines conditions qui font que la poursuite de son exécution s'avère impossible, elle peut se supprimer en exécutant une instruction d'annulation **ROLLBACK** en SQL

Gestion des transactions

- Un système de gestion transactionnel doit garantir les propriétés suivantes (résumées par le vocable **ACID**) :
  - ***Atomicité***
    - Une transaction doit effectuer toutes ses mises à jour ou rien faire du tout
  - ***Cohérence***
    - La transaction doit faire passer la base de données d'un état cohérent à un autre
  - ***Isolation***
    - Les résultats d'une transaction ne doivent être visibles aux autres transactions qu'une fois la transaction validée
  - ***Durabilité***
    - Dès qu'une transaction valide ses modifications, le système doit garantir que ces modifications seront conservées en cas de panne

ACCÈS CONCURRENT AUX DONNÉES

- Plusieurs utilisateurs ou applications peuvent accéder aux données en même temps. De surcroît, la plupart des machines sont connectées à un réseau, ce qui augmente encore le nombre potentiel d'utilisateurs simultanés
- **Lecture(s) étrange(s)**
  - Que se passe-t-il lorsqu'un ou plusieurs utilisateurs décident de modifier les mêmes données au même moment ?
  - L'utilisateur 'Doudou' consulte la liste des voitures non vendues.
  - L'utilisateur 'Modou' enregistre la vente d'une voiture.
  - L'utilisateur 'Doudou' relance la même requête et trouve un résultat différent.
  - L'utilisateur 'Modou' invalide la vente de cette voiture.
  - L'utilisateur 'Doudou' pensant s'être trompé relance la même requête qui aboutit au résultat initial.

Lecture(s) étrange(s)

- Pour trois exécutions de la même requête, l'utilisateur 'Doudou' va obtenir des résultats différents. On peut considérer la même séquence exécutée dans un ordre différent.
  - L'utilisateur 'Doudou' consulte la liste des voitures non vendues.
  - L'utilisateur 'Doudou' relance la requête et trouve le même résultat.
  - L'utilisateur 'Modou' enregistre la vente d'une voiture.
  - L'utilisateur 'Modou' invalide la vente de cette voiture.
  - L'utilisateur 'Doudou' relance la requête précédente et retrouve le même résultat.

Incohérence de résultats

- En raison de l'augmentation des frais de structure, le service comptable a décidé d'une augmentation générale du prix de vente de 5 %.
- Afin que cette dernière passe inaperçue et dans le cadre d'une campagne de communication, le service marketing offre 100 euros de ristourne sur tout le catalogue pendant un mois.
- L'utilisateur 'Doudou' appartient au service comptable et l'utilisateur 'Modou' au service marketing.
- Les vérifications sont effectuées par l'utilisateur 'Omar' de la direction

Incohérence de résultats

- L’utilisateur ‘Doudou’ consulte le prix de vente de la voiture ‘X’.
- L’utilisateur ‘Modou’ effectue la modification de promotion sans vérifier le prix de vente (Prix_vente = Prix_vente – 100).
- L’utilisateur ‘Doudou’ effectue la modification d’augmentation (Prix_vente = Prix_vente × 1,05).
- L’utilisateur ‘Omar’ vérifie le résultat de l’opération de modification du prix de vente et constate une différence avec le résultat attendu :
  - le prix est égal à (Prix_vente – 100) × 1.05,
  - alors qu’il s’attendait à un prix égal à (Prix_vente × 1,05) – 100.

Verrous

- On bloque une ressource pour effectuer les opérations et on la libère dès que les opérations sont effectuées.
- Cette méthode présente cependant quelques pièges et doit être utilisée avec prudence.
- En effet, on peut rapidement parvenir à une situation de blocage que l'on nomme **étreinte fatale** (*deadlock* en anglais) ou parfois **interblocage**.

Exemple

- L'utilisateur 'Doudou' veut insérer une vente de voiture dans la table 'vente' et constate à cette occasion une erreur sur la couleur dans la table 'voiture' qu'il veut modifier.
- L'utilisateur 'Omar' veut insérer à la fois une nouvelle voiture dans la table 'voiture' et la mention de sa vente dans la table 'vente'.
- La séquence d'instruction peut être la suivante :
  - L'utilisateur 'Doudou' pose un verrou sur la table 'vente'.
  - L'utilisateur 'Omar' pose un verrou sur la table 'voiture'.
  - L'utilisateur 'Doudou' a terminé la mise à jour de 'vente' et veut réaliser celle de 'voiture'…mais la table 'voiture' est verrouillée par 'Omar'.
  - L'utilisateur 'Omar' a terminé la mise à jour de 'voiture' et veut procéder à celle de 'vente'… mais la table 'vente' est verrouillée par 'Doudou.
- On parvient à une situation où les deux utilisateurs attendent que l'autre libère la ressource pour continuer. Évidemment, c'est une attente infinie qui se profile pour chacun.

Algorithme des « lecteurs-rédacteurs »)

- L'utilisation classique des verrous pour protéger l'accès à une ressource est la suivante (algorithme des « lecteurs-rédacteurs ») :
  - Tous les lecteurs peuvent lire en même temps.
  - Si un rédacteur veut écrire, aucun lecteur ne doit plus utiliser la donnée.
  - Si un rédacteur est en train d'écrire, aucun lecteur ne peut lire la donnée et aucun autre rédacteur ne peut y accéder (accès exclusif).
  - S'il existe un grand nombre de lecteurs qui se succèdent en lecture sur la donnée, l'attente pour un rédacteur qui voudrait la modifier peut alors devenir quasi infinie.

- Laisser la gestion des verrous à un utilisateur est donc délicat et peut conduire à des blocages du système. Les SGBD performants disposent d'outils capables de détecter et résoudre ces phénomènes, voire de les prévenir le cas échéant.

Réalisation des transactions dans les SGBD

- Pour mettre en œuvre les transactions, le SGBD doit offrir **l'exclusivité d'accès** aux données ainsi que la possibilité **d'annuler** des modifications effectuées.
- La possibilité de revenir en arrière repose sur l'utilisation d'un journal de transactions
- On conserve donc un journal de l'ensemble des lectures et écritures afin de pouvoir reconstituer un système cohérent.
- En utilisant ce journal, à partir d'un état courant du système, on est capable de réexécuter les instructions qui n'ont pu l'être effectivement ou on revient en arrière si ce n'est pas possible

Verrouillage

- Il repose sur les deux actions :
  - **verrouiller (A)** : acquérir un contrôle de l'objet A
  - **libérer (A)** : libérer l'objet A
- Un objet A est typiquement un n-uplet de la BD
- **Il y a deux types de verrous :**
  - Verrous **exclusifs** (X locks) ou verrous d’écriture
  - Verrous **partagés** (S locks) ou verrous de lecture

Protocole d’accès aux données

- 1. Aucune transaction ne peut effectuer une lecture ou une mise à jour d'un objet si elle n'a pas acquis au préalable un verrou S ou X sur cet objet
- 2. Si une transaction ne peut obtenir un verrou déjà détenu par une autre transaction T2, alors elle doit attendre jusqu’à ce que le verrou soit libéré par T2
- 3. Les verrous X sont conservés jusqu’à la fin de la transaction (COMMIT ou ROLLBACK)
- 4. En général les verrous S sont également conservés jusqu’à cette date