Bases de données
Transactions - ACID

Bertrand LIAUDET

SOMMAIRE

Mise à jour Novembre 2016

LES TRANSACTIONS

Présentation des transactions

ACID

Une transaction est un ensemble de requêtes élémentaires du DML (insert, update, delete) qui doivent être exécutées ensemble pour maintenir la cohérence de la base.

Une transaction doit être **ACID** : Atomique, Cohérente, Isolée, Durable

Atomique

Tout ou rien. Soit toutes les opérations de la transaction sont validées, soit aucune ne l'est.

Il ne faut donc pas valider une partie d'une transaction.

Cohérente

La transaction doit maintenir la cohérence de la BD.

La cohérence est une conséquence des contraintes d'intégrité (mais aussi de de l'atomicité et de l'isolation). Là aussi, c'est un principe de base des SGBD. Les différentes contraintes d'intégrés permettent de gérer cela.

Isolée !!!

Une transaction doit être indépendantes des autres transactions qui se déroulent en même temps. C'est un problème compliqué. Quand on utilise des données dans une transaction il faut avoir lu ces données dans cette transaction. Et dans certains cas il faut spécifier en plus 'FOR UPDATE' pour dire qu'on va y toucher. A partir de là, le risque est de lire des données qui seront modifiées pendant qu'on les utilisent.

L'isolation est le problème le plus compliqué à gérer.

Durable

Les effets d'une transaction doivent être persistants.

La durabilité est un principe de base des SGBD. Il est lié à la question des pannes. Toute donnée enregistrée doit l'être de façon durable.

On va donc surtout s'intéresser aux problèmes de l'atomicité et de l'isolation.

MySQL

Le moteur MyISAM ne gère pas l'isolation. Les transactions ont forcément lieu l'une après l'autre.

Les moteurs InnoDB, Berkeley DB, NDB Cluster gère l'isolation.

Atomicité : le problème des pannes

Principe : Commit et Rollback

Pour être atomique, une transaction doit valider toutes ses requêtes élémentaires ou aucunes.

START TRANSACTION : permet de marquer le début de la transaction.

COMMIT : permet de valider la transaction : toutes les requêtes élémentaires sont désormais validées durablement.

ROLLBACK : tant que le COMMIT n'est pas exécuté, un ROLLBACK permet de revenir en arrière. C'est ce que fera automatiquement le système en cas de panne au milieu de la transaction.

Mode Autocommit

En mode Autocommit, chaque requête est validée individuellement validée implicitement (commit).

C'est le fonctionnement par défaut de MySQL.

Une transaction peut s'ouvrir explicitement avec un START TRANSACTION qui stoppe alors l'autocommit jusqu'au prochain COMMIT explicite.

Mode sans Autocommit

En mode sans Autocommit, la première requête lancée ouvre implicitement une transaction qui s'achèvera au prochain COMMIT.

La requête suivante s'ouvrira alors de nouveau implicitement une transaction.

Un START TRANSACTION en mode sans Autocommit est équivalent à un COMMIT.

Gestion de l'Autocommit

Select @@autocommit

Set @@aucommit=0 ; // mode sans autocommit

COMMIT implicite

De nombreuses commandes implique un COMMIT implicite

START TRANSACTION

Le passage en mode autocommit

Le DDL : Create, alter, drop, rename, pour tout objet.

Truncate (drop+create)

Lock et unlock

Suspendre l'intégrité référentielle

Set @@foreign_key_cheks = 0 ; // suspend l'intégrité référentielle.

En suspendant l'intégrité référentielle, on peut saisir les données dans n'importe quel ordre. C'est à manier avec prudence et toujours au sein d'une transaction !

Isolation

https://makina-corpus.com/blog/metier/2015/bien-debuter-avec-les-transactions-sql