Transaction

Dans ce TP, j'ai exploré en profondeur la gestion de la concurrence des transactions dans un système de réservation de vols, en mettant l'accent sur la comparaison entre les niveaux d'isolation REPEATABLE READ et SERIALIZABLE.

1. Objectif du TP:

Analyser et comparer le comportement des transactions concurrentes sous différents niveaux d'isolation, en se concentrant sur les aspects de performance et d'intégrité des données.

2. Scénarios testés:

- Réservations simultanées de billets par deux clients
- Mises à jour concurrentes des places disponibles
- Lectures répétées des données de vol pendant les transactions

3. Comparaison des niveaux d'isolation :

a) REPEATABLE READ:

Avantages observés:

- Permet une meilleure concurrence, améliorant les performances globales
- Évite les lectures non reproductibles au sein d'une même transaction

Limitations constatées :

• Possibilité d'anomalies de phantoms dans certains scénarios

b) **SERIALIZABLE**:

Avantages observés:

- Garantit la plus haute intégrité des données
- Élimine complètement les anomalies de lecture et d'écriture

1. Scénario:

Transaction 1 : Réservation de 2 billets pour le client C1

Transaction 2 : Tentative de réservation de 5 billets pour le client C2

J'ai effectué les transactions concurrentes ci-dessous dans le niveau d'isolation REPEATABLE READ.

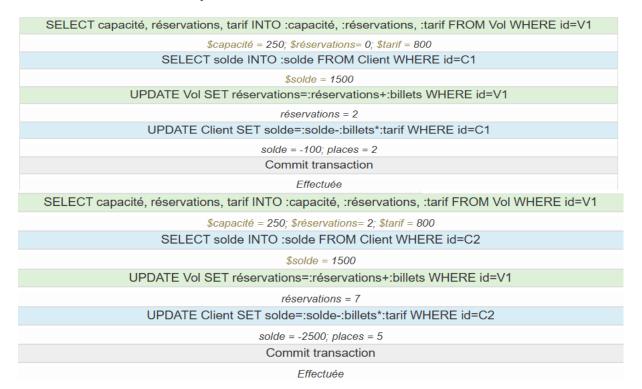
Transaction 1

Transaction 2

	Table Client						Table Client					
	ld	Nom	Solde	Places		ld	N	lom	Solde	Places		
	C1	Serge	-100	2		C1	S	erge	-100	2		
	C2	Philippe	-2500	5		C2	Ph	ilippe	-2500	5		
	Table Vol						Table Vol					
l d	Intitulé	Capacité	Réservati	ons Tarif		l d	Intitulé	Capacité	Réservation	ons Tarif		
V 1	Mexico	250	7	800		V 1	Mexico	250	7	800		
	Variables locales						Variables locales					
9	Scapacité= 2	50 \$réservations=	0 \$tarif= 800	\$solde= 1500		\$ca _j	pacité= 250	\$réservations= 2	\$tarif= 800	\$solde= 1500		
	\$billets= 2						\$billets= 5					
	Requêtes					Requêtes						
	SELECT VI I	NTO SELECT C1 IN	UPDATE VI	UPDATE C1		SE	LECT VI INTO	SELECT C2 INTO	UPDATE V1	UPDATE C2		
	ROLLBACK					ROLLBACK COMMIT						

Historique des requêtes effectuées

dans le niveau d'isolation repeatable read



2. Déroulement :

La Transaction 1 a réussi à réserver 2 places, mettant le solde du client C1 à -100€ (en découvert).

La Transaction 2 a pu lire les données mises à jour par la Transaction 1 (nombre de réservations = 2).

La Transaction 2 a tenté d'ajouter 5 réservations supplémentaires mais a été annulée (rollback).

3. Observations clés :

REPEATABLE READ permet la lecture des modifications commises par d'autres transactions.

Il évite les lectures non reproductibles au sein d'une même transaction.

L'importance des contrôles de cohérence est mise en évidence par l'annulation de la Transaction 2.

4. Implications pratiques:

Ce niveau d'isolation offre un bon équilibre entre la cohérence des données et les performances. Il est crucial de mettre en place des mécanismes de contrôle supplémentaires pour gérer les cas limites (comme le découvert du client C1).

La gestion des rollbacks est essentielle pour maintenir l'intégrité du système de réservation.

niveau d'isolation SERIALIZABLE est susceptible d'entraîner certaines lenteurs.

On peut modifier le niveau d'isolation ci-dessus.



SELECT capacité, réservations,	tarif INTO :capacité, :réservations, :tarif FROM Vol WHERE id=V1					
\$ca	\$capacité = 250; \$réservations= 7; \$tarif = 800					
SELECT so	SELECT solde INTO :solde FROM Client WHERE id=C1					
	\$solde = -100					
UPDATE Vol SE						
	réservations = 9					
	SELECT capacité, réservations, tarif INTO :capacité, :réservations,	:tarif FROM Vol WHERE id=V1				
	en attente 💿 Annuler					

6. Implications pratiques:

Le choix du niveau d'isolation doit être fait en fonction des exigences spécifiques de l'application.

Pour un système de réservation de vols, une approche hybride pourrait être optimale :

- Utiliser REPEATABLE READ pour la majorité des transactions
- Basculer sur SERIALIZABLE pour les opérations critiques ou lors de pics d'activité

Limitations constatées :

• Réduction significative des performances due à un verrouillage plus strict

5. Conclusion:

Ce TP démontre l'importance de choisir le bon niveau d'isolation en fonction des besoins spécifiques de l'application, et l'importance cruciale de comprendre et de gérer efficacement la concurrence des transactions dans les systèmes de bases de données.

REPEATABLE READ s'avère approprié pour un système de réservation, offrant une bonne protection contre les anomalies de lecture tout en permettant un certain niveau de concurrence."

Il a démontré que le choix du niveau d'isolation est un facteur clé pour équilibrer performance et intégrité des données, particulièrement dans des environnements à forte concurrence comme les systèmes de réservation."