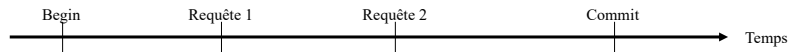


- **Séquence d'opérations de lecture et mise à jour des données**
  - ◆ Exemple instructions SQL (*select, insert, update, delete*)
- **Unité logique de traitement**
  - ◆ Fait passer la BD d'un état cohérent à un autre état cohérent
  - ◆ État cohérent :
    - ★ Schéma conceptuel global
    - ★ Contraintes d'intégrité fonctionnelles
- **Propriétés ACID**
  - ◆ Atomicité : tout ou rien
  - ◆ Cohérence : contraintes d'intégrité (différées)
  - ◆ Isolation : vue mono-utilisateur
  - ◆ Durabilité : résultats validés jamais perdus
- **Instructions SQL**
  - ◆ (**begin;**), **commit;**, **rollback;**

## Déroulement d'une transaction

### ■ Tout se passe bien



### ■ **Commit**

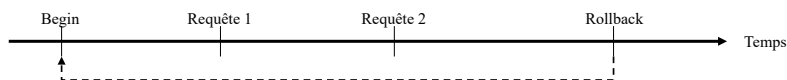
- ◆ Validation de la transaction
- ◆ Les modifications de la transaction deviennent permanentes

## Déroulement d'une transaction (2)

### ■ Annulation - **Rollback**

- ◆ Pas d'effet dans la base de données

### ■ Abandon explicite



### ■ Erreur dans l'exécution

- ◆ Norme : Abandon automatique
- ◆ En pratique : le SGBD rend la main



### ■ Principe

- ◆ Soit toutes les requêtes s'exécutent correctement soit aucune

### ■ En pratique

- ◆ Utilisation d'un journal d'image avant
  - ★ Sauvegarde de l'état de la base de données avant la modification
- ◆ Opérations sur une copie des données
  - ★ Remplacement en cas de validation

### ■ Principe

- ◆ Toutes les contraintes d'intégrité doivent être vérifiées
  - ★ Vérification au fur et à mesure de l'exécution
  - ★ Vérification différée à la validation

### ■ Syntaxe SQL

- ◆ Ajout du mot clef **DEFERRABLE** après la définition d'une contrainte d'intégrité
- ◆ Dans une transaction :
  - ★ **SET CONSTRAINT [S]**  
(**ALL** | <constraint\_name>)  
(**IMMEDIATE** | **DEFERRED**) ;

## Isolation

### ■ Principe

- ◆ L'utilisateur doit avoir l'impression d'être le seul connecté à la BD
- ◆ Une transaction ne doit pas pouvoir voir les résultats intermédiaires des autres transactions

### ■ En pratique

- ◆ Exécution concurrente de plusieurs transactions  
Entrelacement des opérations des différentes transactions
- ◆ Ordonnancement *sérialisable*  
Ordonnancement produisant un résultat équivalent à celui obtenu par une exécution des transactions en série

## Ordonnancements, serializabilité?

### Ordonnancement 1

	T1	T2	A	B
Temps			25	25
	lire(A,t)			
	t:=t+100			
	écrire(A,t)		125	
	lire(B,t)			
	t:=t+100			
	écrire(B,t)			125
		lire(A,s)		
		s:=s*2		
		écrire(A,s)	250	
		lire(B,s)		
		s:=s*2		
		écrire(B,s)		250

## Ordonnancements, serializabilité?

Ordonnancement 1				Ordonnancement 2			
T1	T2	A	B	T1	T2	A	B
		25	25			25	25
lire(A,t)				lire(A,s)			
t:=t+100				s:=s*2			
écrire(A,t)		125		écrire(A,s)			
lire(B,t)				lire(B,s)			
t:=t+100				s:=s*2			
écrire(B,t)			125	écrire(B,s)			
	lire(A,s)			lire(A,t)			
	s:=s*2			t:=t+100			
	écrire(A,s)	250		écrire(A,t)			
	lire(B,s)			lire(B,t)			
	s:=s*2			t:=t+100			
	écrire(B,s)		250	écrire(B,t)			

Equipe pédagogique BD Ensimag

11

## Ordonnancements, serializabilité?

Ordonnancement 3				Ordonnancement 4			
T1	T2	A	B	T1	T2	A	B
		25	25			25	25
lire(A,t)				lire(A,s)			
t:=t+100				s:=s*2			
écrire(A,t)				écrire(A,s)			
	lire(A,s)			lire(A,t)			
	s:=s*2			t:=t+100			
	écrire(A,s)			écrire(A,t)			
lire(B,t)				lire(B,s)			
t:=t+100				s:=s*2			
écrire(B,t)				écrire(B,s)			
	lire(B,s)			lire(B,t)			
	s:=s*2			t:=t+100			
	écrire(B,s)			écrire(B,t)			

Equipe pédagogique BD Ensimag

13

## Ordonnancements, serializabilité?

### Ordonnancement 5

T1	T2	A	B
		25	25
lire(A,t)			
t:=t+100			
écrire(A,t)			
	lire(A,s)		
	s:=s*2		
	écrire(A,s)		
	lire(B,s)		
	s:=s*2		
	écrire(B,s)		
lire(B,t)			
t:=t+100			
écrire(B,t)			

## Techniques

### ■ Pessimistes

- ◆ Interdiction des séquences potentiellement non sérialisables
- ◆ Exemples
  - ★ Verrouillage
  - ★ Estampillage

### ■ Optimistes

- ◆ Exécution libre et vérification à la validation
- ◆ Exemple
  - ★ Certification

## Niveaux d'isolation

- La *sérialisabilité* peut être trop contraignante...
- **Alternative : relâcher l'isolation**
  - ◆ 4 niveaux d'isolation
    - ★ Read uncommitted
    - ★ Read committed
    - ★ Repeatable Read
    - ★ Serializable

## Niveaux d'isolation (1)

- **Read uncommitted**
  - ◆ Possible lecture de valeurs non validées (*lectures sales*)
    1.  $T_1$  lit le n-uplet A
    2.  $T_1$  écrit le n-uplet A
    3.  $T_2$  lit le n-uplet A (valeur au temps 2)
    4.  $T_1$  rollback
    5.  $T_2$  exploite une valeur de A qui n'est pas validée

## Niveaux d'isolation (2)

### ■ Read committed

- ◆ Lecture des valeurs validées uniquement  
(*pas de lectures sales !*)
- ◆ Lectures non reproductibles possibles
  1.  $T_1$  lit A
  2.  $T_1$  écrit A
  3.  $T_2$  lit A (valeur au temps 1)
  4.  $T_1$  commit
  5.  $T_2$  relit A (valeur validée par  $T_1$  au temps 4)
- ◆ Contrôle de concurrence
  - ★ Séquences écriture→écriture et  
écriture→lecture

## Niveaux d'isolation (3)

### ■ Repeatable Read

Read committed + si des données sont relues dans une tx, les données vues lors de la 1ère lecture, sont vues la 2ème fois aussi mais des données supplémentaires peuvent apparaître

### ■ Possibilité de fantômes

1.  $T_1$  calcule la moyenne de R.A
2.  $T_2$  insert un nouveau n-uplet dans R
3.  $T_2$  commit
4.  $T_1$  utilise un résultat « faux »...

### ■ Contrôle de concurrence

- ◆ Séquences écriture→écriture, écriture→lecture et  
lecture→écriture



## Niveaux d'isolation (4)

### ■ Serializable

- ◆ Niveau le plus fort
- ◆ Pas d'anomalies, pas de fantômes
- ◆ Contrôle de concurrence
  - ★ Séquences écriture→écriture, écriture→lecture, lecture→écriture et lecture→insertion

Isolation level	Lectures sales	Lectures non reproductibles	Fantômes
Read Uncommitted	oui	oui	oui
Read Committed	non	oui	oui
Repeatable reads	non	non	oui
Serializable	non	non	non

### ■ Principe

- ◆ Les effets d'une transaction validée ne doivent **jamais** être perdus

### ■ En pratique

- ◆ Journaux
  - ★ Opérations exécutées au fur et à mesure
  - ★ Terminaison de transaction
  - ★ (Images après)
- ◆ Sauvegardes externalisées

### ■ Utilité

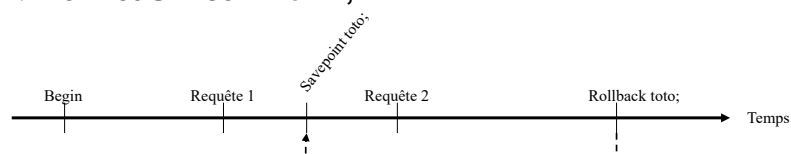
- ◆ Abandon partiel

### ■ Propriété

- ◆ L'exécution reste linéaire

### ■ Syntaxe SQL

- ◆ **Savepoint** <nom>;
- ◆ **Rollback to** <nom>;



## Isolation dans Oracle

- **Verrouillage multi-versions**

- ◆ Chaque utilisateur (session) travaille sur une copie des données

- **Principe**

- ◆ Verrouillage en lecture sur la copie locale
- ◆ Verrouillage en écriture sur toutes les copies

- **Modes d'isolation**

- ◆ Read committed (par défaut)
  - ★ Synchronisation de la copie locale dès qu'une transaction valide, quelle qu'elle soit (mode *push*)
- ◆ Serializable
  - ★ Synchronisation de la copie locale uniquement à la terminaison de la transaction locale (mode *pull*)

## En pratique...

- Au début de la transaction

### SET TRANSACTION ISOLATION LEVEL X

X peut être

1. SERIALIZABLE
2. READ COMMITTED

- Ne pas oublier

- ◆ set autocommit off / on