

Concept de transaction :

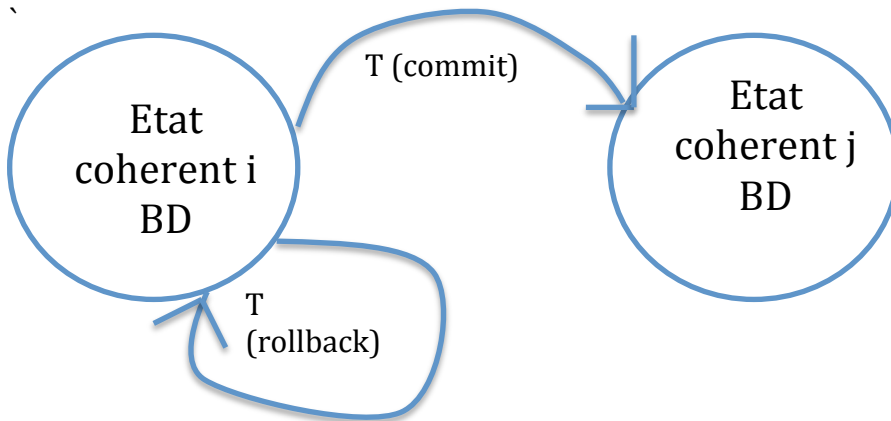
Pb1 Cohérence bd face à la concurrence

Pb2 Cohérence bd face aux pannes



TRANSACTION

Definition de la transaction :



$T = \{\text{maj dans BD}\}$

Les actions possibles :

COMMIT : sure de la réalisation

ROLLBACK : retour à l'opération précédente, la version antérieure

BEGIN/END : lancement de la maj

(SAVEPOINT) : point de sauvegarde lors d'une opération (pannes)

LOCK/UNLOCK : les verous

➔ UPDATE, INSERT, DELETE, SET ...

Propriétés SQL2 transaction

Panne

« A ».Atomicité (atomicity) < TOUT ou RIEN >
 → write a head og protocol (LOG) -> journal
 → shadow mechanims (miroir)

T transfert

```
BEGIN UPDATE COMPTE
  SET SOLDE = SOLDE + 1
  WHERE # COUNT = '100'
  UPDATE COMPTE
  SET SOLDE = SOLDE - 1
  WHERE # COUNT = '150'
END Transaction transfert
```

« C ». COHERENCE(consistency)

Propriété ACID

Concurrence :

« I ».ISOLATION ->

LOCK | READ(partage)

UPDATE(exclusif) → risque interbloquage (LIVE LOCK, DEAD LOCK) → selection { Prevention, detection, resolution

UNLOCK

« D ».DURABILITE (durability)

Pb :

Panne
interférence
concurrency

TRANSACTION

LIVELOCK
DEADLOCK

Detection par
graphe d'état
resolution

Transaction (AC)

T1. Augmentation de 10 de la capacité des AIRBUS

T2. Reduction de 5 de la capacité des avions localisé à Paris

Begin T1

LOCK AVION IN EXCLUSIVE MODE

UPDATE AVION

SET CAPT = CAP + 10

Where AVION = « AIRBUS »

UNLOCK AVION

End T1

Begin T2

LOCK AVION IN EXCLUSIVE MODE

UPDATE AVION

SET CAPT = CAP - 5

Where LOC = 'Paris'

UNLOCK AVION

End T2

wait

Resultat de bd coherent pour airbus localisé à paris :

$$CAP_F = CAP_I + 10 - 5 = CAP_I + 5$$

SERIALISIBILITY : transaction cohort

Lock (verrou) <-> Isolation



Extend BD : $\left\{ \begin{array}{l} \text{partagé /shared lock /read lock} \\ \text{exclusif /updatelock} \end{array} \right.$

P1 : Transaction « bien fournie » : Ecriture d'un verrou lecture avant de lire et d'un verrou ecriture avant d'ecrire

P2 : verouillage à 2 phases $\left\{ \begin{array}{l} 1. \text{acquisition des verrou (SEIZE BLOCK)} \\ 2. \text{liberation des verrou} \end{array} \right.$

Th de JIM GRAY : si P1 et P2 alors SERIALISABILITY

-> execution des transaction maintient la coherence des BD

1. Seize block : lock , unlock
2. Unlock

Grain de verouillahe : A tuple, B table -> data space, C page

Verrou A -> concurrence la plus forte -> uniquement bloque les acyions lié a cette tuple :
OLTP : on line transaction processing

T3 : verrouillage AVION puis VOL

T4 verouillage VOL puis AVION

Begin

LOCK AVION IN EXCLUSIVE MODE
LOCK AVION IN SHARED MODE
UPDATE AVION

Begin

LOCK VOL IN EXCLUSIVE MODE
UPDATE AVION

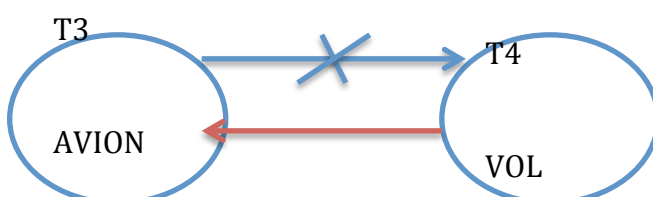
LOCK VOL IN EXCLUSIVE MODE
LOCK VOL IN SHARED MODE
UPDATE VOL

LOCK AVION IN EXCLUSIVE MODE
UPDATE VOL

End

End

DEADLOCK : pour le débloquent, on doit changer les mots en rouge par ceux en vert



Dead lock <-> cycle dans gaph
d'attente

ROLLBACK

Create view AVION_NICE

As (select * from AVION where LOC = 'Nice')

Begin T5 lock avion in share mode

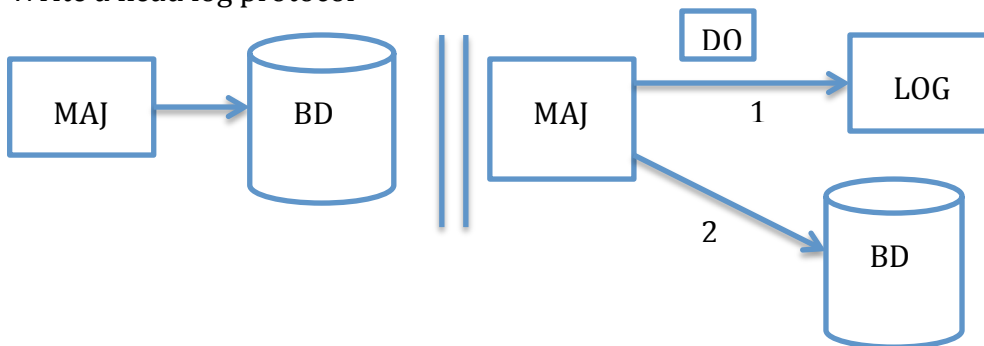
1. Requête (definit) en view
2. Table (resultat) shapshot

Journal de MAJ (LOG)

| {MAJ} | Table | Op | TID | Etat de T | Mode view | Image avan | Image apres |
|--------|-------|-------------------|-----|-----------|-----------|------------|-------------|
| UPDATE | AVION | CAP : CAP + 10 | T3 | COMMIT | | Ptr page | Ptr page |

Panne <-> par def [DO] / REDO / UNDO

Write a head log protocol



MAJ page :

