ESIR1 BD
Bases de données

mise à jour

Olivier Ridoux

Mise à jour de BD

Plan

- Insertion de données
- Transactions

Insertion de données

Ajout de lignes (1)

- INSERT INTO Table ($attr_1$, $attr_2$, ... , $attr_n$)
  VALUES (v_1, v_2, ... , v_n) ;

- INSERT INTO Table ($attr_1$, $attr_2$, ... , $attr_n$)
  SELECT $attr_1$, $attr_2$, ... , $attr_n$ FROM ... ;

Ajout de lignes (2)

- INSERT INTO Table ($attr_1$, $attr_2$, ... , $attr_n$)

  SELECT $attr_1$, $attr_2$, ..., **$constante_i$**, ... , $attr_n$

  FROM ... ;

Exemple - Ajout de lignes

- INSERT INTO produits
  VALUES (8, ecrou, 5, 12, vert) ;

- INSERT INTO braderie(pno, prix)
  SELECT pno, 100 FROM produits
  WHERE produit.prix < 500
  AND produit.prix > 100 ;

Modification de lignes

- UPDATE Table

  SET $attr_i$ = ...

  WHERE ... ;

Exemple - Modification de lignes

- UPDATE produits
  SET prix = prix * 1.05
  WHERE couleur = 'rouge' ;

- UPDATE produits
  SET prix = prix * 0.9
  WHERE prix = (SELECT MAX(prix)
  FROM produits) ;

Suppression de lignes

- DELETE FROM Table
  WHERE … ;

Exemple - Suppression de lignes

- DELETE FROM produits
  WHERE prix > 100 ;
- DELETE FROM produits
  WHERE pno IN (SELECT pno
  FROM Temp) ;
- DELETE FROM produits ;

Notion de transaction

Transaction

- Un traitement constitué de plusieurs requêtes...

...qu'il faut considérer comme un tout

Exemple - dépot / retrait

- Retrait

UPDATE compte

SET solde = solde – S

WHERE num-compte = A ;

- Dépot

UPDATE compte

SET solde = solde + S

WHERE num-compte = A ;

Exemple - transfert

- Transfert S de A vers B

```
UPDATE compte SET solde = solde - S
WHERE num-compte = A ;

% « milieu du gué »

UPDATE compte SET solde = solde + S
WHERE num-compte = B ;
```

- Invariant : **solde de A + solde de B = k**

Invariant

- Respecté au début et à la fin
- Mais pas **au « milieu du gué »**
- Personne ne doit voir le **« milieu du gué »**
- Sûr de rien si multi-processus

Transaction

- Séquence de requêtes **ACID**
  - **atomicité** : toute la séquence est **exécutée ou rien**
  - **cohérence** : les contraintes sont **respectées**
  - **isolation** : les états intermédiaires ne sont **pas visibles**
  - **durabilité** : les actions de la transaction sont **effectives**

Programmation des transactions

- BEGIN TRANSACTION
  ...séquence de requêtes...
  END TRANSACTION
- COMMIT
- ROLLBACK

Journalisation (1)

- À chaque début de transaction **T**

  Journal += **(T : Start)**

- À chaque modification

  Journal += **(T : endroit, état avant, état après)**

- En fin de transaction

  Journal += **(T : Commit)**

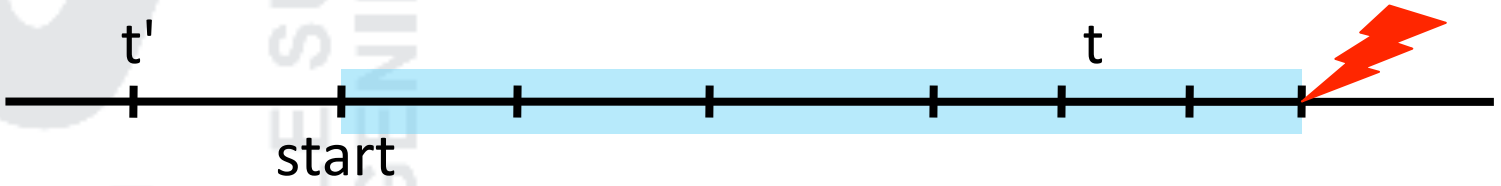
Journalisation (2)

- Si accident au temps t

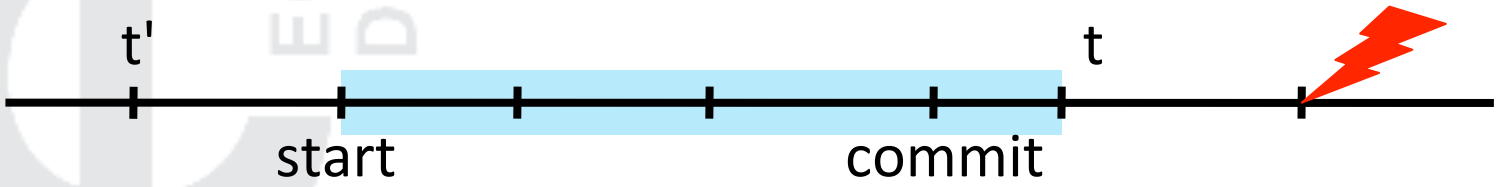
  et sauvegarde au temps t' < t

...lire le journal de t à t'

– si start **sans commit**, rétablir les **états avant**



– si start **avec commit**, rétablir les **états après**



Bilan journal

- Critères AC.D respectés
- Journalisation du journal ?
- Pb requêtes concurrentes

à voir plus tard