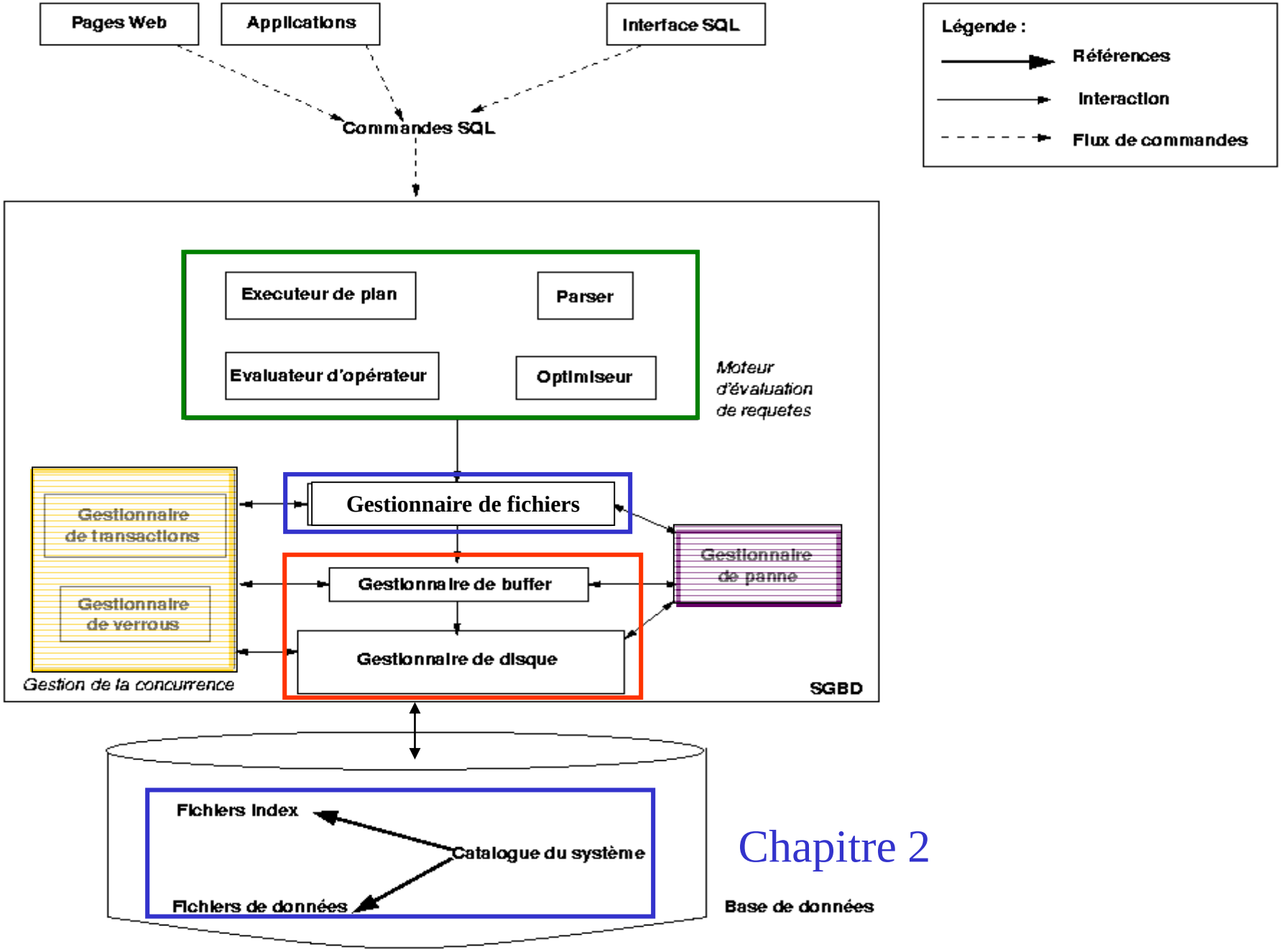
Architecture d'un SGBD
reprise après pannes
Marta Rukoz

Pannes : Gestion des pannes, types de panne dans les SGBD, journaux des mises à jour, procédures de reprise


Pages Web
Applications
Interface SQL
Commandes SQL
Légende :
Références
Interaction
Flux de commandes
Executeur de plan
Parser
Evaluateur d'opérateur
Optimiseur
Moteur d'évaluation de requetes
Gestionnaire de transactions
Gestionnaire de verrous
Gestionnaire de fichiers
Gestionnaire de buffer
Gestionnaire de disque
Gestionnaire de panne
Gestion de la concurrence
SGBD
Fichiers Index
Catalogue du système
Fichiers de données
Base de données

Chapitre 2

Les transactions fournissent

l'exécution *atomique* et *fiable* en présence de pannes

l'exécution *correcte* en présence d'utilisateurs concurrents

Problème:

Comment maintenir

atomicité

durabilité

des transactions

Pannes

- Fonctions du **gestionnaire de pannes**
  - **Atomicité :** en défaisant les T qui ne se valident pas
  - **Durabilité :** en vérifiant que les opérations effectuées sur la base par les T validées « survivent » aux pannes
- Différents types de panne
  - Panne de transaction *(abandon normal ou du à un deadlock)*
  - Panne du système *(panne de processeur, mémoire, alimentation…)*
  - Panne de la mémoire secondaire *(les données sur disque sont perdues, panne de tête de lecture ou du contrôleur disque)*

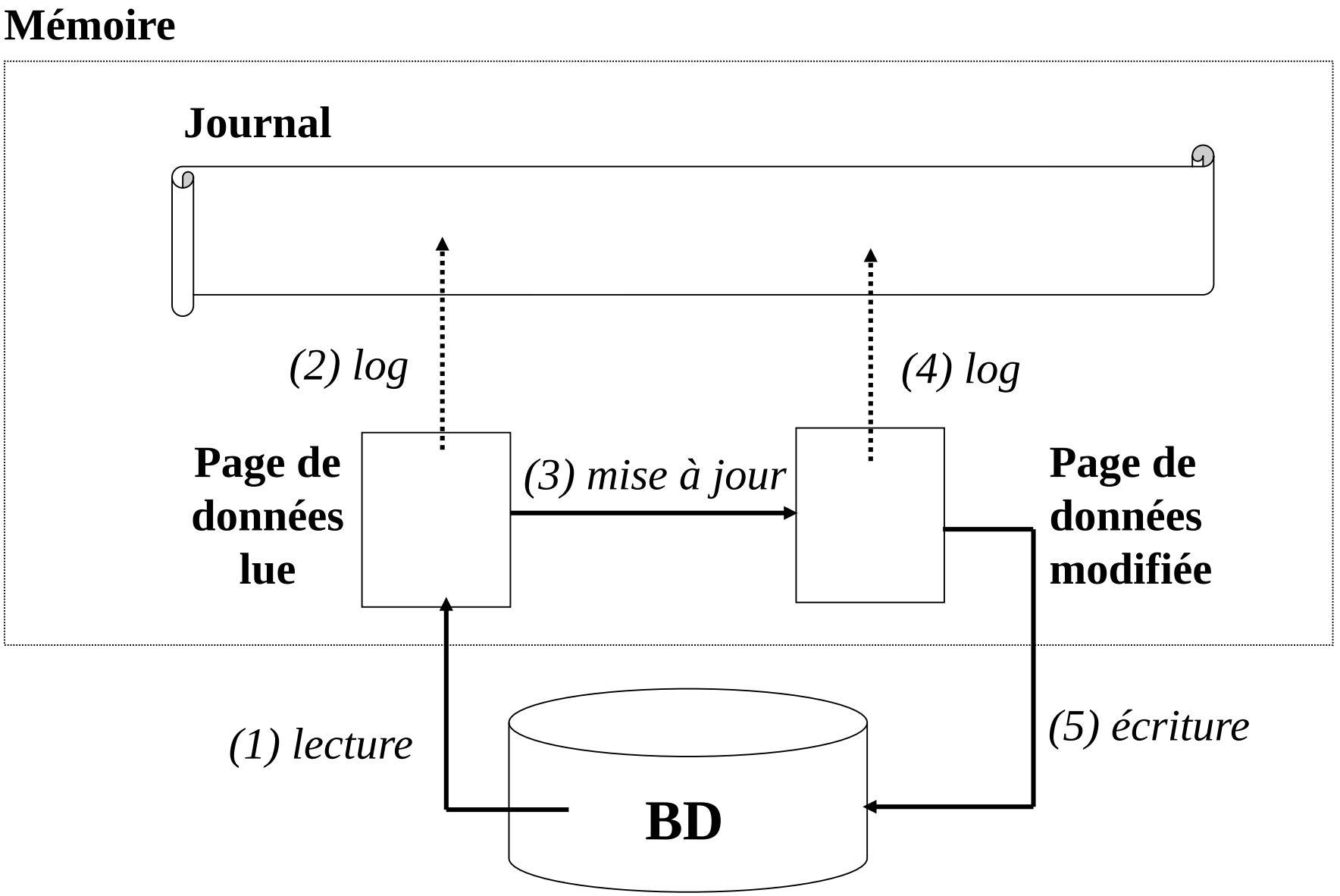
Journaux

- **Journal** ou ***log***

  Historique des modifications effectuées sur la base

- **Journal des images avant (*rollback segment*)**
  - Valeurs des pages avant modifications
  - Pour défaire (*undo*) les mises à jour d'une transaction
- **Journal des images après (*redo log*)**
  - Valeurs des pages après modifications
  - Pour refaire (*redo*) les mises à jour d'une transaction
- **Points de reprise**

Processus de journalisation



Gestion du journal

- Écriture des pages du journal dans un *buffer* en mémoire
- Sauvegarde du journal lorsque le *buffer* est plein
- Sauvegarde du journal lorsqu'il y a validation d'une transaction ou d'un groupe de transactions
- **Ecriture du journal sur le disque avant l'écriture des pages de données modifiées**
- **Structure des enregistrements**
  - Numéro de transaction
  - Type d'enregistrement (*start*, *update*, *commit*, *abort* …)
  - Adresse de la page modifiée
  - Image avant
  - Image après

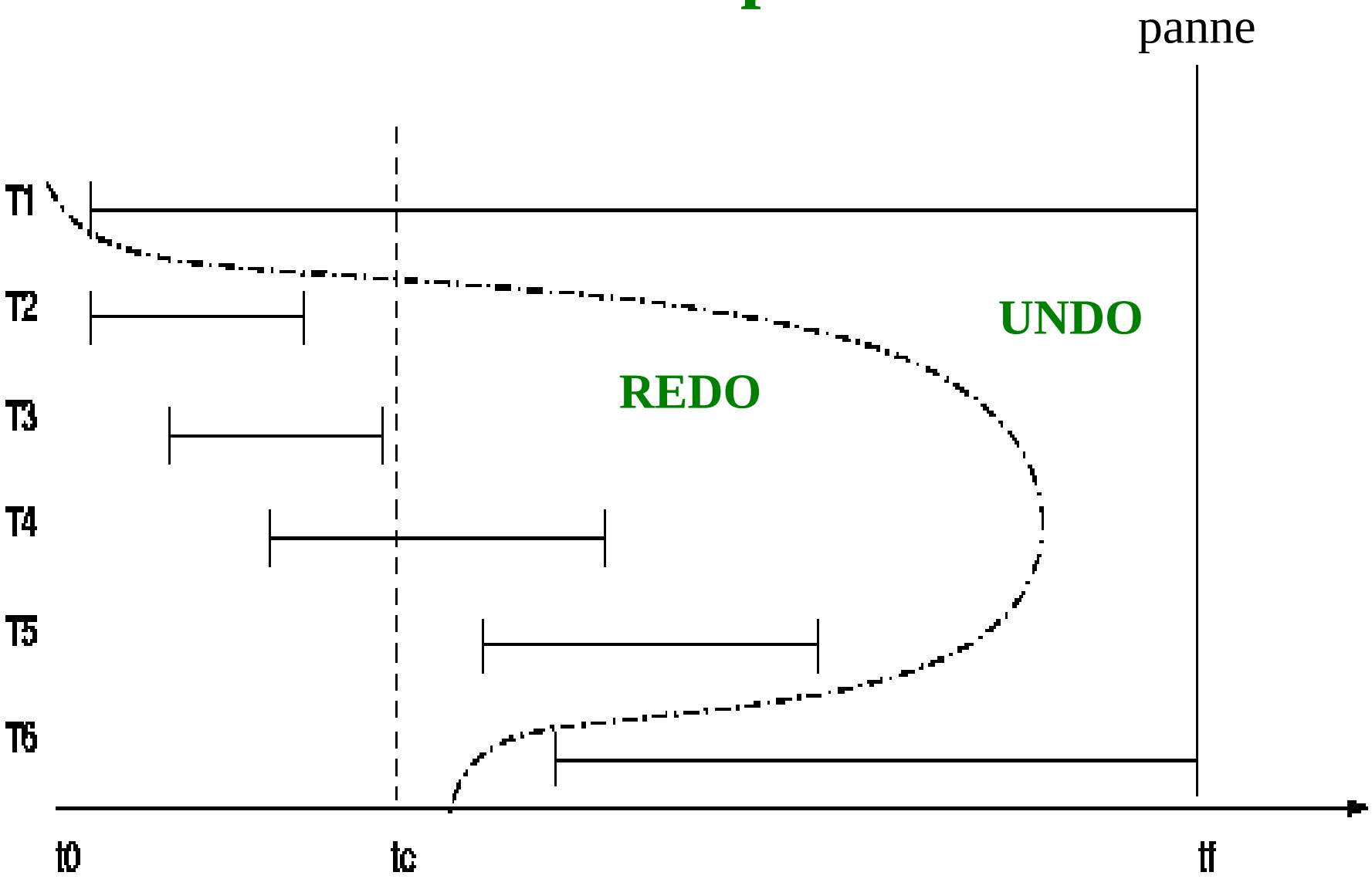
Exemple de journal

Début du journal ⟶

*T*1, begin
*T*1, *x*, 99, 100
*T*2, begin
*T*2, *y*, 199, 200
*T*3, begin
*T*3, *z*, 51, 50
*T*2, *w*, 1000, 10
*T*2, commit
*T*4, begin
*T*3, abort
*T*4, *y*, 200, 50
*T*5, begin
*T*5, *w*, 10, 100

Fin du journal ⟶ *T*4, commit

Exemple



Techniques de validation d'une transaction

- Modification immédiate de la base :
  - Chaque mise-à-jour cause la modification dans des pages dans le cache de la BD
  - L'ancienne valeur est écrasée par la nouvelle
- Modification différée de la base
  - Les nouvelles valeurs de données sont écrites séparément des anciennes dans des pages ombres
  - Mises-a-jour des index
  - Peu utilisé en pratique car très cher
- Basculement des tables des pages

Journalisation : modification immédiate de la base

- **Au début de la T**, insérer un enregistrement de début de transaction dans le journal
- **Quand une opération d'écriture modifie la base**, des enregistrements indiquant les mises à jour sont insérées dans le journal

  → modifications sont effectuées dans les pages du buffer

  → les mises à jour sont reportées sur les pages physiques de la base
- **Lorsque la transaction valide**, un enregistrement indique cette validation dans le journal

Modification immédiate de la base : cas de panne

•Si aucun enregistrement de validation n'est trouvé dans le journal pour une T

$\Rightarrow$ T était active au moment de la panne, il faut l'annuler

$\rightarrow$ le journal est relu dans le sens inverse pour défaire les opérations de T dans l'ordre

•S'il existe un enregistrement de validation pour T

$\rightarrow$ T doit être rejouée

Exemple 1

Soit l'ordonnancement suivant :

<T0, debut>, <T0, ecrire(A,95)>, <T0, ecrire(B,205)>, <T0, valider>, <T1,debut>, <T1,ecrire(C,60)>, <T1,valider>

Avant le lancement de l'ordonnancement la base contient : A = 100, B=200, C=700.

Exécutions :

Journal	Base de donnés
<T0, debut>	
<T0, A,100,95>	
<T0, B,200,205>	A = 95
	A = 95 B=205

Panne ⇒ il faut défaire T0. Le journal est lu à l'envers pour faire un *undo* ⇒ les anciennes valeurs de A et B sont restaurées

Exemple 2

Exécutions :

Journal	Base de donnés
<T0, debut>	
<T0, A,100,95>	
<T0, B,200,205>	**A = 95**
	A = 95 B=205
<T0, valider>	
<T1,debut>	
<T1,C,700,60>	
	C = 60

Panne ⇒ il faut défaire (*undo*) T1 et rejouer (*redo*) T0

Exemple 3

Exécutions :

Journal	Base de donnés
<T0, debut>	
<T0, A,100,95>	
<T0, B,200,205>	**A = 95**
	A = 95 B=205
<T0, valider>	
<T1,debut>	
<T1,C,700,60>	
	C = 60
<T1,valider>	

Panne ⇒ les deux transactions doivent être rejouées (*redo*)

Quand écrire le journal sur disque?

Supposons une transaction *T* qui modifie la page *P*

Cas chanceux

- le système écrit *P* dans la BD sur disque
- le système écrit le journal sur disque pour cette opération
- PANNE!... (avant la validation de *T*)

Nous pouvons reprendre (undo) en restaurant *P* à son ancien état grâce au journal

Cas malchanceux

- le système écrit *P* dans la BD sur disque
- PANNE!... (avant l'écriture du journal)

Nous ne pouvons pas récupérer car il n'y a pas d'enreg. avec l'ancienne valeur dans le journal

Solution: le protocole Write-Ahead Log (WAL)

Protocole WAL

Observation:

- si la panne précède la validation de transaction, alors toutes ses opérations doivent être défaites, en restaurant les images avant (*partie undo* du journal)
- dès qu'une transaction a été validée, certaines de ses actions doivent pouvoir être refaites, en utilisant les images après (*partie redo* du journal)

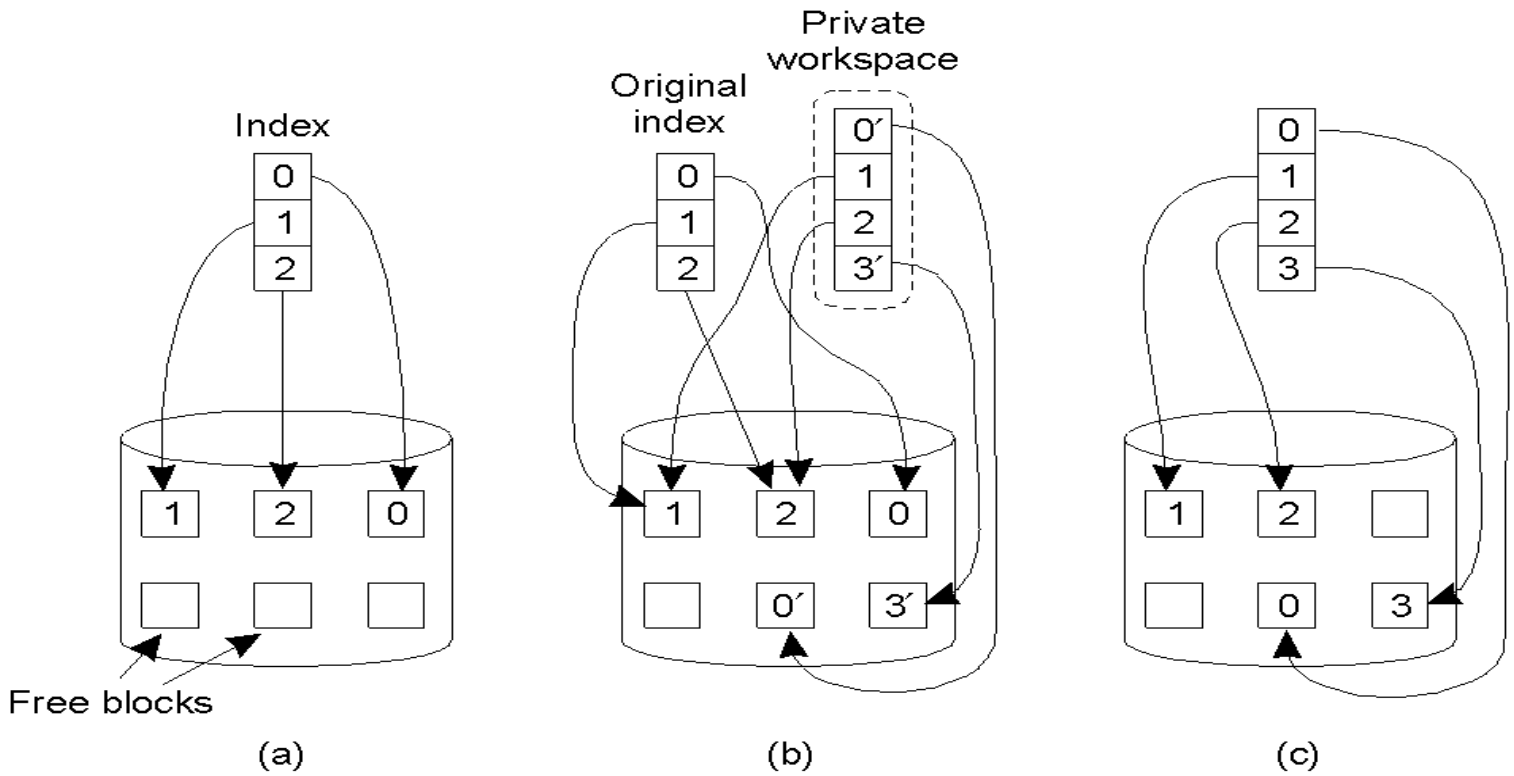
Protocole WAL:

- avant d'écrire dans la BD sur disque, la partie undo du journal doit être écrite sur disque
- lors de la validation de transaction, la partie redo du journal doit être écrite sur disque avant la mise-à-jour de la BD sur disque

Modification différée de la base

- **Au début de la T**, insérer un enregistrement de début de transaction dans le journal
- **Quand une opération d'écriture modifie la base**, des enregistrements indiquant les mises à jour sont insérés dans le journal. La base de données n'est pas modifiée. Il n'est pas nécessaire de stocker l'image avant dans le journal.
- **Lorsque la transaction valide**, un enregistrement indique cette validation dans le journal, et les mises à jour sont effectuées sur la base en utilisant les enregistrements contenus dans le journal

Basculement des tables de pages



a) **Initialement**

b) **Après mise à jour du block 0 et ajout du block 3.**

c) **Après validation**

Panne : rien à faire. Cependant, il y fragmentation du disque et il est nécessaire de nettoyer régulièrement le disque pour supprimer les pages non référencées

Points de reprise

Réduit la quantité de travail à refaire ou défaire lors d'une panne

Un point de reprise enregistre une liste de transactions actives

Pose d'un point de reprise:

- écrire un enreg. begin_checkpoint dans le journal
- écrire les buffers du journal et de la BD sur disque
- écrire un enreg. end_checkpoint dans le journal

Procédures de reprise

- **Objectif**

  Reconstruire, à partir du journal et éventuellement de sauvegarde**, un état proche de l'état cohérent de la base avant la panne,** en perdant le minimum de travail

- **Reprise à chaud**

  Perte de données en mémoire mais pas sur disque

  - ***No Undo, Redo*** ***(utilisée en cas de modification différées de la base)***
  - ***Undo, Redo*** ***(le plus classique et le plus fréquemment utilisé)***
  - ***Undo, No Redo*** ***(applicable dans le cas où on est certain que toutes les transactions validées ont ces modifications de reporter sur la base, en mémoire secondaire)***

Procédures de reprise

- **Reprise à froid**

  Perte de tout ou partie de données sur disque

  Pour reconstruire la base, on utilise une sauvegarde de la base ainsi que le journal (dernier point de reprise) :

  - **REDO des transactions validées**
  - **UNDO inutile**

  Une panne est catastrophique si tout ou partie du journal sur mémoire secondaire est perdu