Le journal des transactions (*log*)

Le fichier des transactions, ou *log*, un fichier complémentaire à la base de données.

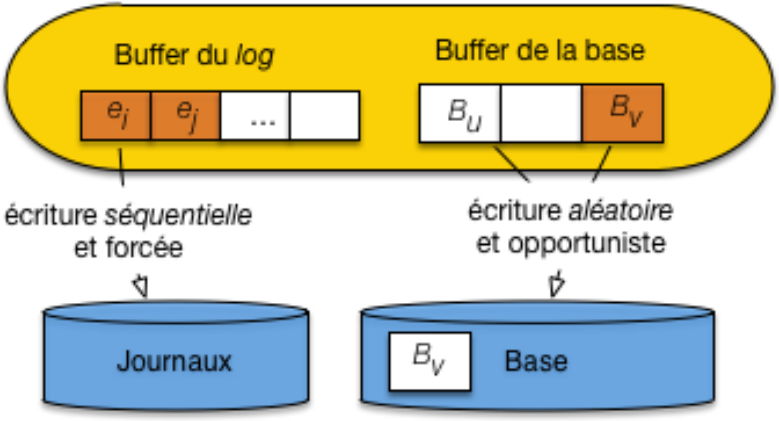
- le système y écrit **séquentiellement**
- le *log* dispose de son propre buffer
- on n'efface **jamais** un enregistrement du *log*

La reprise sur panne repose sur la stratégie suivannte :

- **la base et son buffer** sont en écriture **opportuniste**
- **le *log* et son buffer** sont en écriture **immédiate**

Le *log* contient l'historique des mises à jour.

Le système d'entrées/sorties avec *log*



Contenu du *log*

Toutes les instructions de mise à jour.

Chaque ligne contient un enregistrement d'une des formes suivantes.

- `start(T)`
- `write(T, x, old_val, new_val)`
- `commit(T)`
- `rollback(T)`
- `checkpoint`

Le `write` peut être représenté sous forme **logique** (instructions) ou **physique** (enregistrements).

Le *log* et la gestion des `commit`

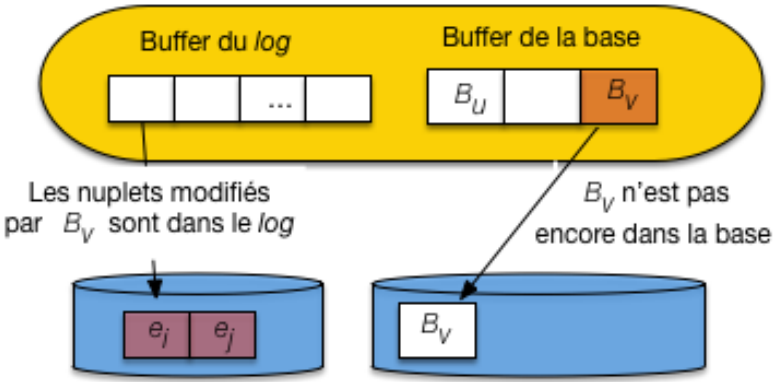
La règle suivante est dite **point de commit** :

- au `commit`, on force l'écriture des modifications effectuées dans le *log*

Conséquences

- **L'état de la base** peut être reconstitué à partir du *log*
- Une transaction est validée quand l'instruction `commit` est écrite dans le *log*

Illustration : après un `commit`



Le *log* et la gestion des `rollback`

Un bloc **modifié** mais pas encore **validé** peut être écrit dans la base.

En cas de panne, il faudra reprendre l'image avant.

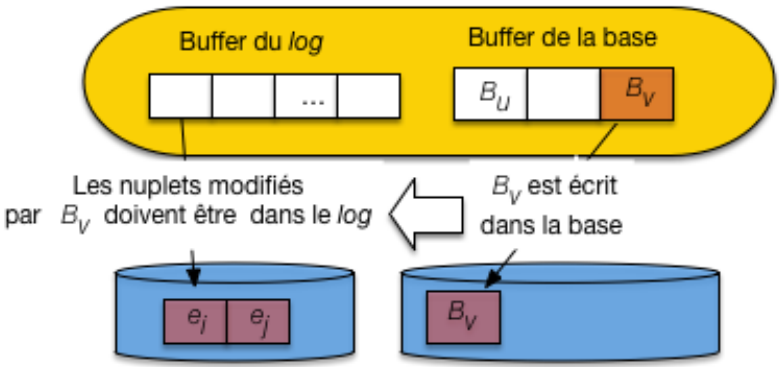
La règle suivante est dite *write-ahead logging*

- quand un bloc modifié est écrit dans la base, il faut **d'abord** écrire les modifications dans le *log*.

Conséquence

- Le *log* contient des modifications qui n'ont pas été validées.

Illustration du *write-ahead*



Résumé : le *log*

Retenir :

- Le *log* est un fichier séquentiel dans lequel on enregistre toutes les mises à jour
- un `commit` est effectif quand l'instruction est écrite dans le *log*
  $\Rightarrow$ l'état de la base est dans le *log*
- un `rollback` peut s'effectuer en reprenant l'image avant dans le *log*
- performances : on laisse la base et son buffer en mode d'écriture opportuniste

Prochaine étape : l'algorithme de reprise sur panne

Résumé : le *log*

Retenir :

- Le *log* est un fichier séquentiel dans lequel on enregistre toutes les mises à jour
- un `commit` est effectif quand l'instruction est écrite dans le *log*
  $\Rightarrow$ l'état de la base est dans le *log*
- un `rollback` peut s'effectuer en reprenant l'image avant dans le *log*
- performances : on laisse la base et son buffer en mode d'écriture opportuniste

Prochaine étape : l'algorithme de reprise sur panne

Merci !