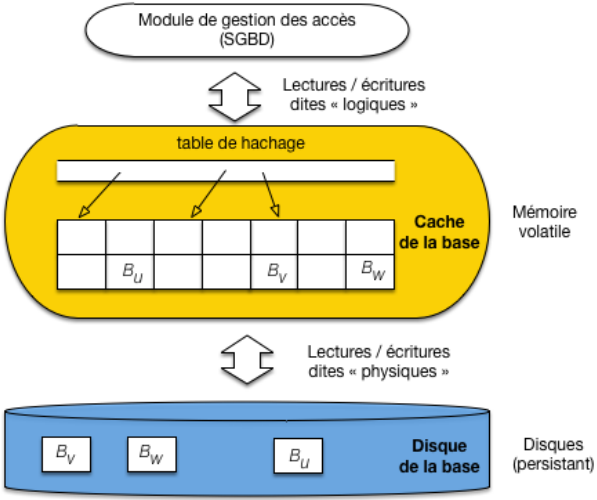
Le buffer et le disque



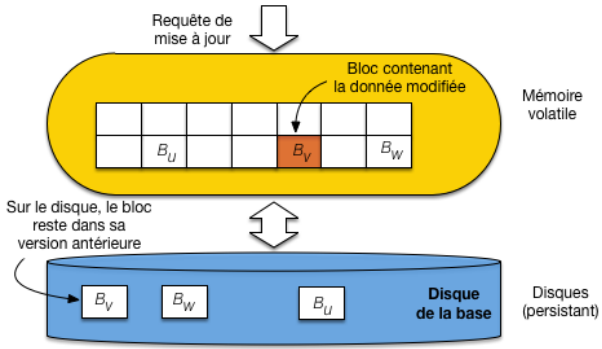
Stratégie de remplacement

Que faire quand la mémoire est pleine ?

L'algorithme le plus courant est dit Least Recently Used (LRU).

- La "victime" est le bloc dont la dernière date de lecture logique est la plus ancienne.
- Ce bloc est alors soustrait de la mémoire centrale.
- Le nouveau bloc vient le remplacer.

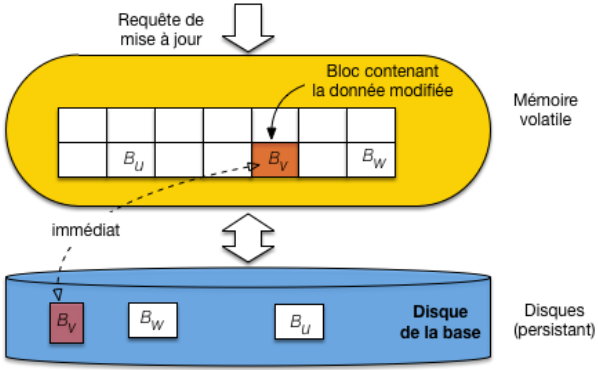
Opération de mise à jour



Faut-il écrire le bloc ou non ?

On trouve le bloc (comme pour une lecture), on modifie la donnée.

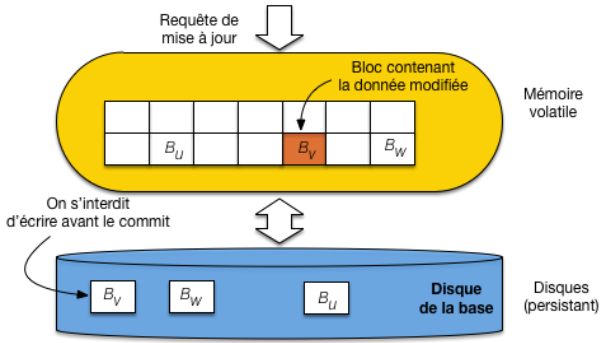
Mise à jour immédiate



Dès qu'un bloc est modifié, on écrit sur le disque pour synchroniser.

- Peu performant : **écritures aléatoires**
- **On écrase l'image avant**

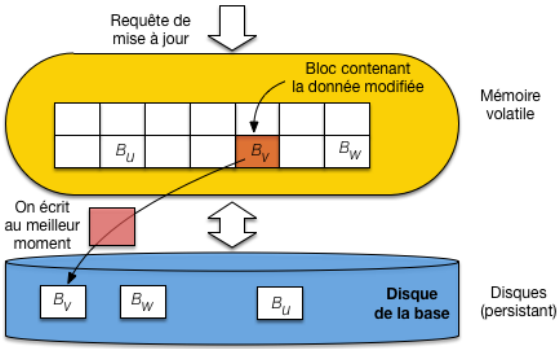
Mise à jour différée



On **interdit** l'écriture d'un bloc modifié avant le `commit`

- Risque de surcharger le buffer
- Préserve l'image avant

Mise à jour opportuniste



C'est la méthode la plus souple : on attend la meilleure opportunité pour synchroniser le buffer et le disque.

Le moins pénalisant ; permet des écritures successives dans le buffer.

Résumé : lecture/écriture, buffer et disque

Retenir :

1. Une partie de la base est copiée dans le buffer.
2. Tout mise à jour s'effectue **d'abord dans le buffer**.
3. La synchronisation avec le disque peut s'effectuer de manière **immédiate**, **différée**, ou **opportuniste**.

La méthode de synchronisation a un impact fort sur la reprise sur panne.

Résumé : lecture/écriture, buffer et disque

Retenir :

1. Une partie de la base est copiée dans le buffer.
2. Tout mise à jour s'effectue **d'abord dans le buffer**.
3. La synchronisation avec le disque peut s'effectuer de manière **immédiate**, **différée**, ou **opportuniste**.

La méthode de synchronisation a un impact fort sur la reprise sur panne.

Merci !