

Fichier et Disques

Lecture : Transfert des données du disque vers la mémoire centrale.

Ecriture : Transfert de la RAM vers le disque.

Les 2 opérations sont relativement coûteuses par rapport à d'autres opérations qui sont effectuées en RAM.

Pourquoi ne pas tout garder en RAM ?

La RAM est plus chère que les disques

- Volatilité de la RAM

Hiérarchie de stockage

RAM pour les données en cours d'utilisation,

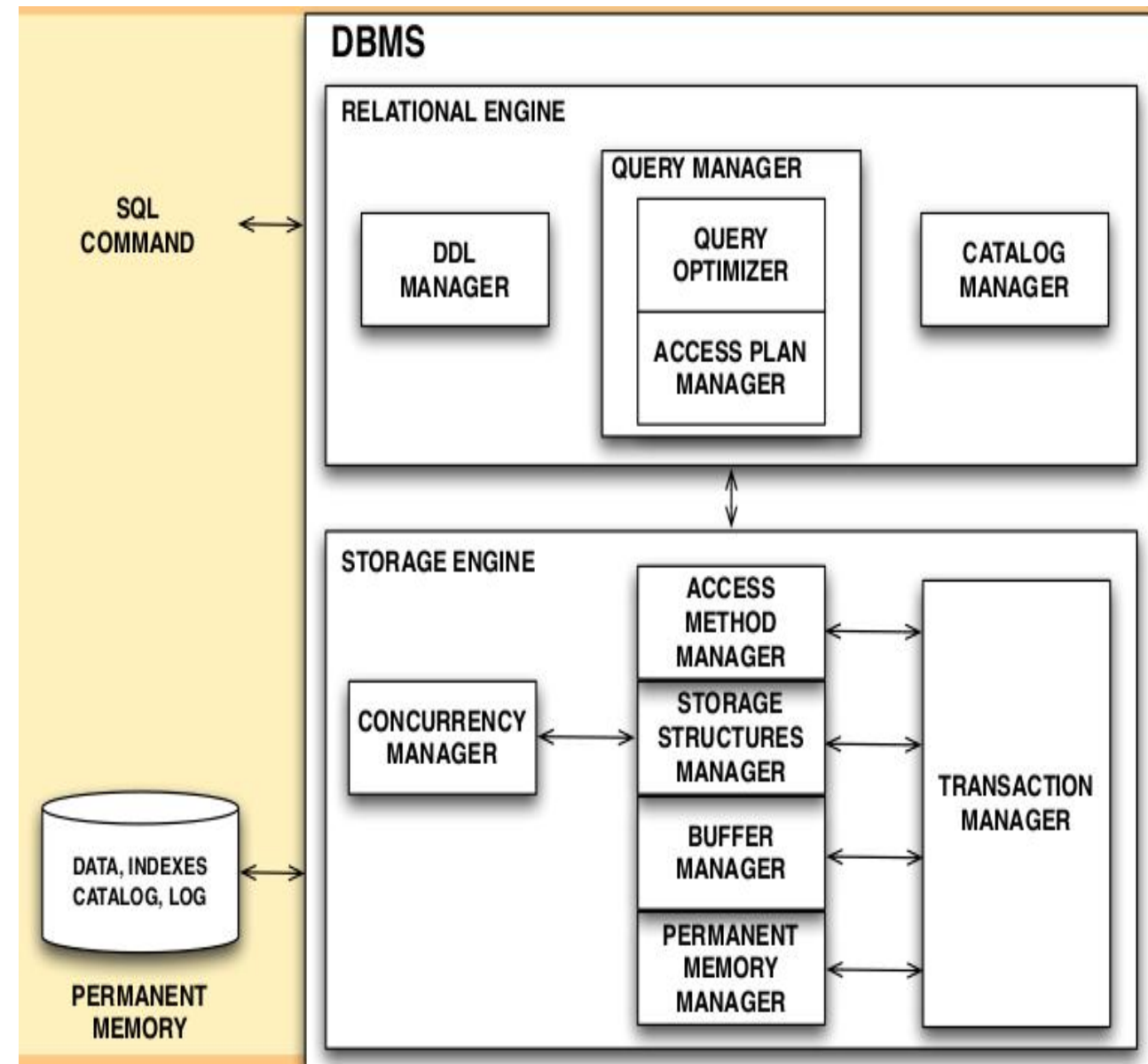
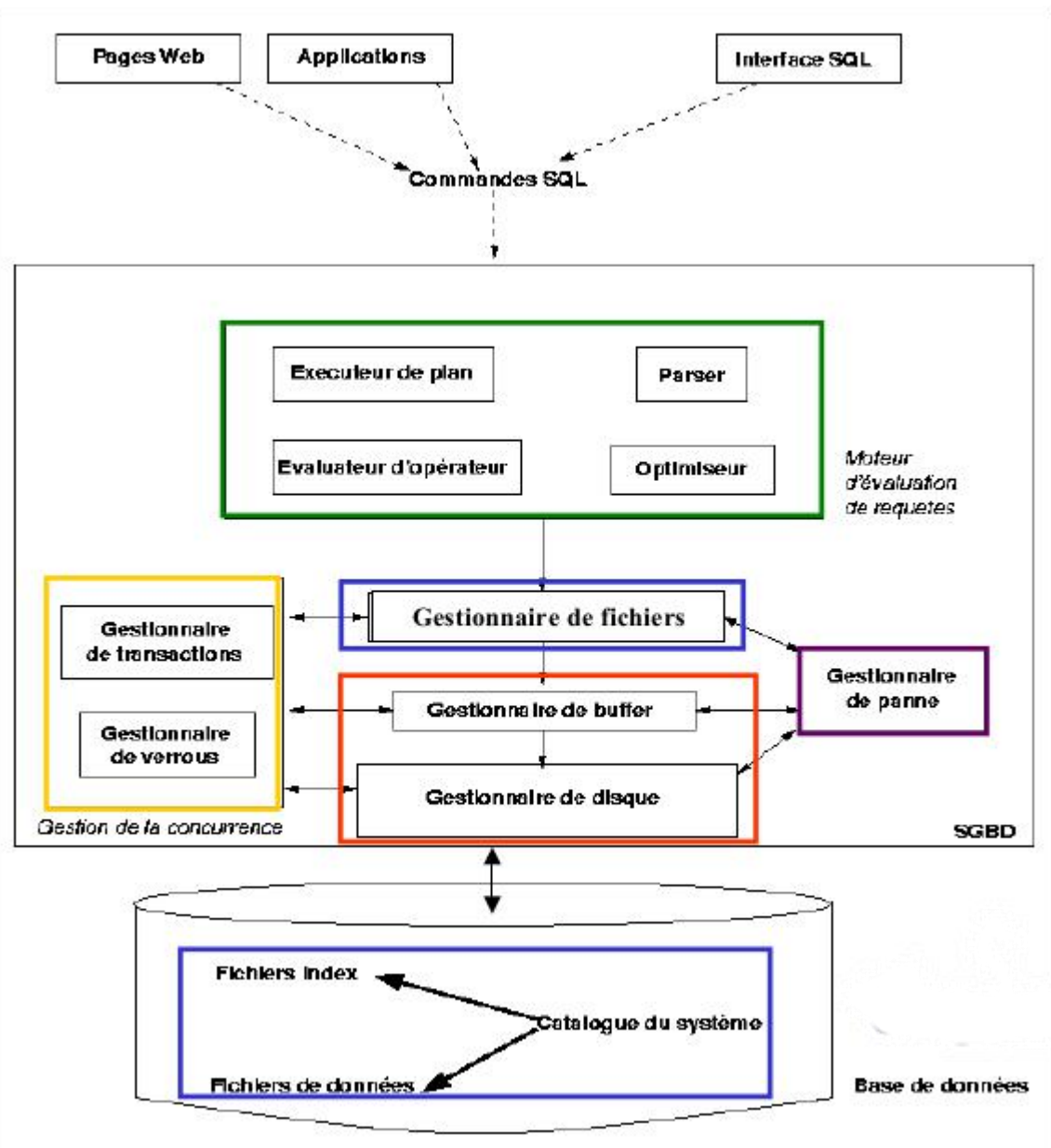
Disques pour toute la BD,

Bandes magnétiques pour les archives.

Les disques

Les données sont stockées et accédées par blocs ou pages entières.

Contrairement aux RAM, le temps d'accès aux données sur les disques dépend de l'emplacement. Il faut donc bien placer les données pour optimiser les transferts.



Architecture d'un SGBD

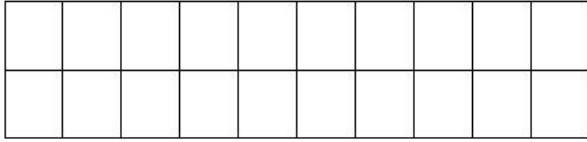
- Vision des données par le SGBD : un ensemble d'enregistrements mémoire
- Vision des données par le gestionnaire de fichiers : un ensemble de pages mémoire
- Vision des données par le gestionnaire de disque : un ensemble de pages disque
- Rôle du gestionnaire de buffer : passage des pages du disque vers la mémoire (et inversement)

Gestionnaire de buffer (Tampon)

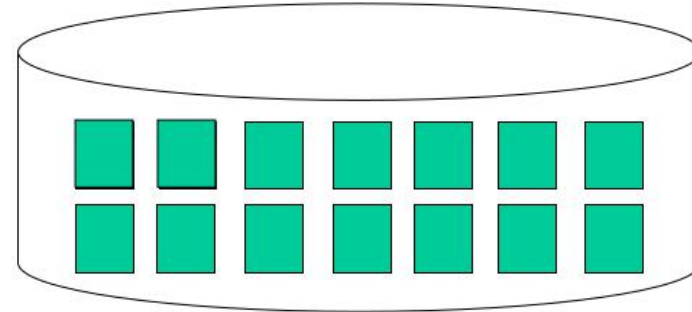
- Rôle : placer, au moment voulu, une page du disque vers la mémoire et inversement
 - Politique de remplacement (ex. LRU :Least Recently Used)
 - Gestion des pages mises à jour
 - Partition de la mémoire
 - Vérification des droits sur les pages




Gestionnaire de buffer

Mémoire



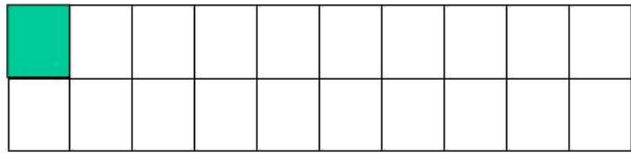
Disque



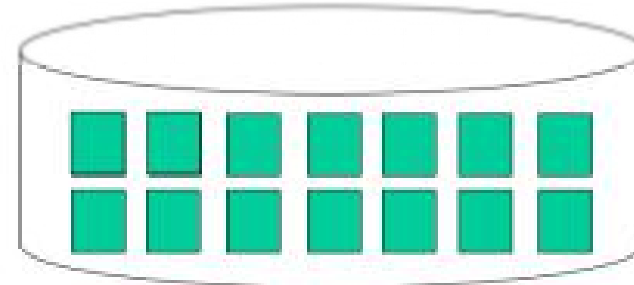
-  Page mémoire libre
-  Page de données
-  Page de données modifiées

Pour que les données puissent être manipulées par le SGBD, il faut qu'elles soient en RAM

Mémoire

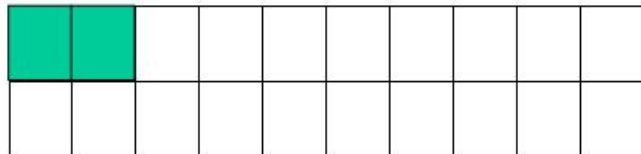


Disque

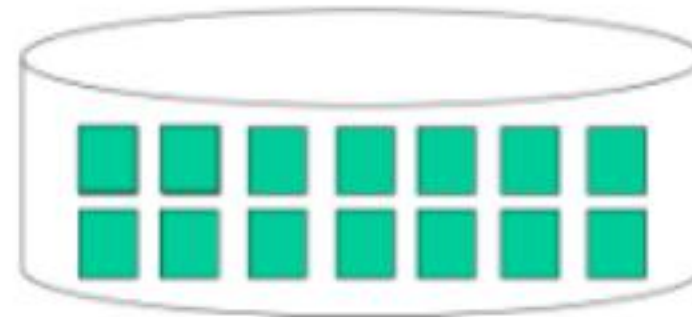


-  Page mémoire libre
-  Page de données
-  Page de données modifiées

Mémoire

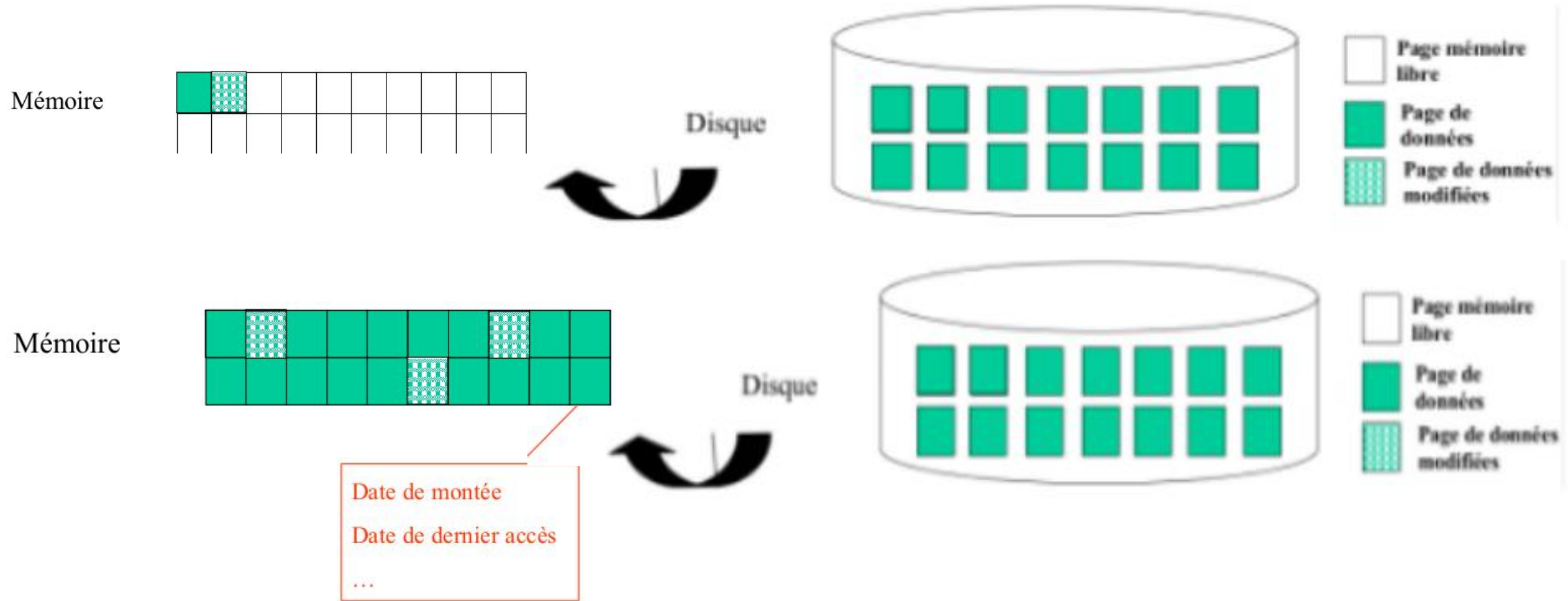


Disque



-  Page mémoire libre
-  Page de données
-  Page de données modifiées

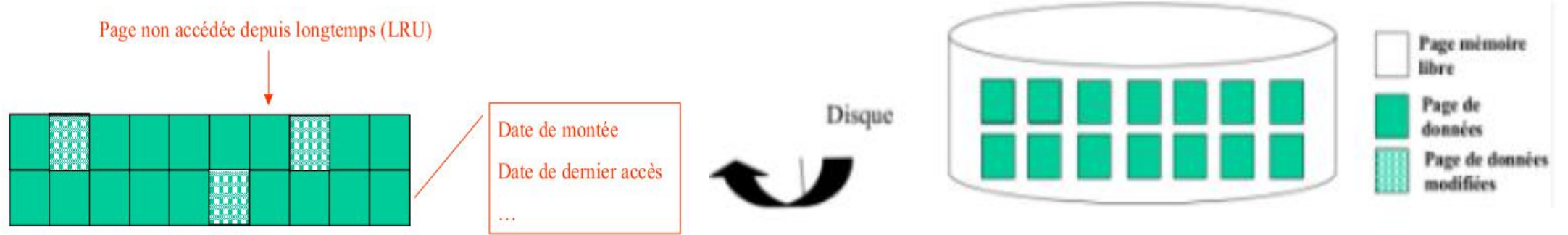
Gestionnaire de buffer



On a une table de $\langle \text{ZoneId}, \text{pageId} \rangle$: A chaque page en RAM est associé un bit qui indique si elle a été modifiée et un compteur des transactions qui sont en train de l'utiliser

Gestionnaire de buffer

Mémoire



Mémoire

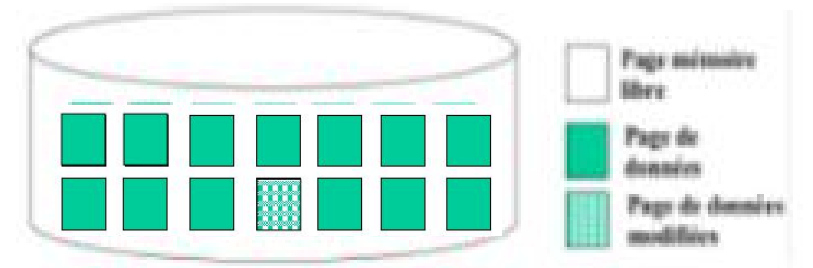
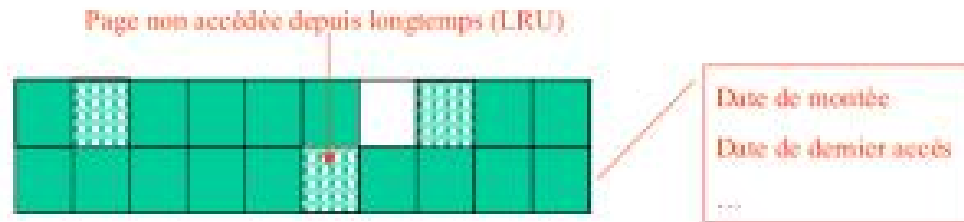


Mémoire

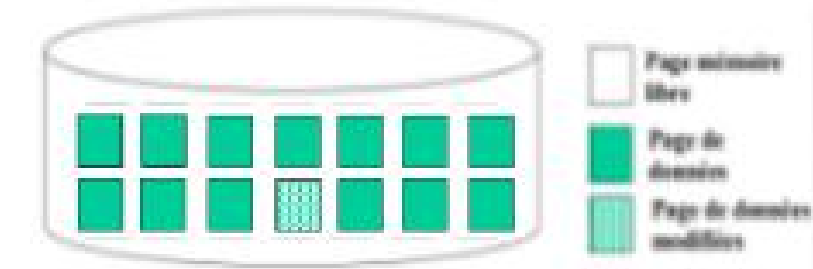
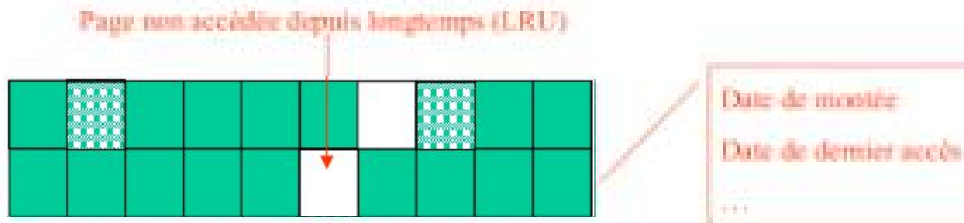


Gestionnaire de buffer

Mémoire



Mémoire



Quand une page est demandée :

Si la page n'est pas disponible, le gestionnaire du buffer émet une demande au gestionnaire de disque:

- Choisir une page (Zone) du Buffer à remplacer.
- Si la page à remplacer a été modifiée alors la réécriture sur le disque.
- Lire la page demandée, i.e la transférer du disque vers la zone libérée.

2. Incrémenter son compteur et retourner son adresse.

La zone choisie pour être remplacée doit avoir son compteur à Zéro, i.e aucune transaction n'est en train de l'utiliser

Choix de remplacement :

LRU (Least Recently used) : La moins récemment utilisée.

MRU (Most Recently Used) : La plus récemment utilisée.

FIFO (First In First Out), **LIFO** (Last In First Out) ...

Chacune des stratégies peut être efficace dans une situation mais pas dans une autre.

Exemple : Lire plusieurs fois un fichier de 10 pages avec un tampon de 9 zones. Dans ce cas, MRU est meilleur que LRU.

Gestionnaire de disque / Permanent Memory Manager

Les pistes de même diamètre forment un cylindre.

La taille d'un bloc est un multiple de la taille d'un secteur.

Temps d'accès à un bloc (TAB) : Optimisation sur TP et LR

Temps de positionnement (TP) : Le temps moyen qu'il faut pour positionner la tête de L/E sur la bonne piste.

Latence de rotation (LR) : Le temps moyen qu'il faut pour positionner la tête de L/E au bon endroit sur la piste.

Taux de transfert (TT) : La quantité d'info transférée par unité de temps (du disque à la RAM ou inversement).

Placement des pages sur disque

Le concept du bloc "Suivant" :

1. Mettre les blocs (successifs) sur la même piste,
2. Blocs sur le même cylindre, ensuite
3. Blocks sur le cylindre adjacent.

Les blocs d'un fichier doivent être stockés d'une manière séquentielle sur le disque pour minimiser TP et LR.

Lorsque l'on veut faire **un balayage séquentiel** d'un fichier, le fait de pouvoir préparer (pre-fetch) plusieurs pages à la fois représente un gain considérable)

Formats d'enregistrements

Notons au préalable que le SGBD gère un catalogue qui permet de stocker des informations sur les fichiers (Exemple : Le nom des champs, leurs tailles, ...)

Taille fixe VS Taille variable :

Taille fixe : étant donnée **B** l'adresse d'un enregistrement, on peut accéder directement à un champ particulier



Adresse de base (B)

Adresse(C3)=B+T1+T2

Formats d'enregistrements

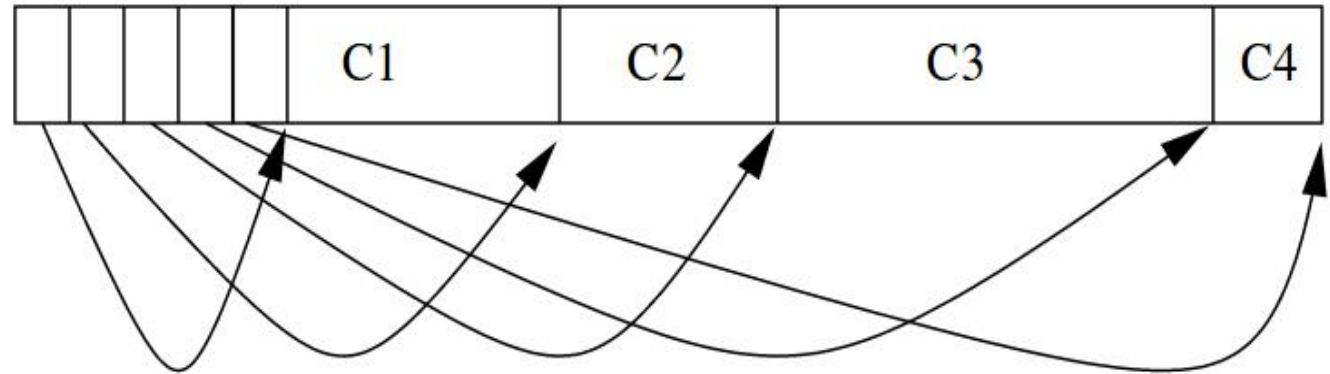
Taille variable : (le nombre de champs est fixe)

Deux alternatives :



1-Les champs séparés par un caractère spécial

2-L'entête de l'enregistrement indique le début de chaque champs

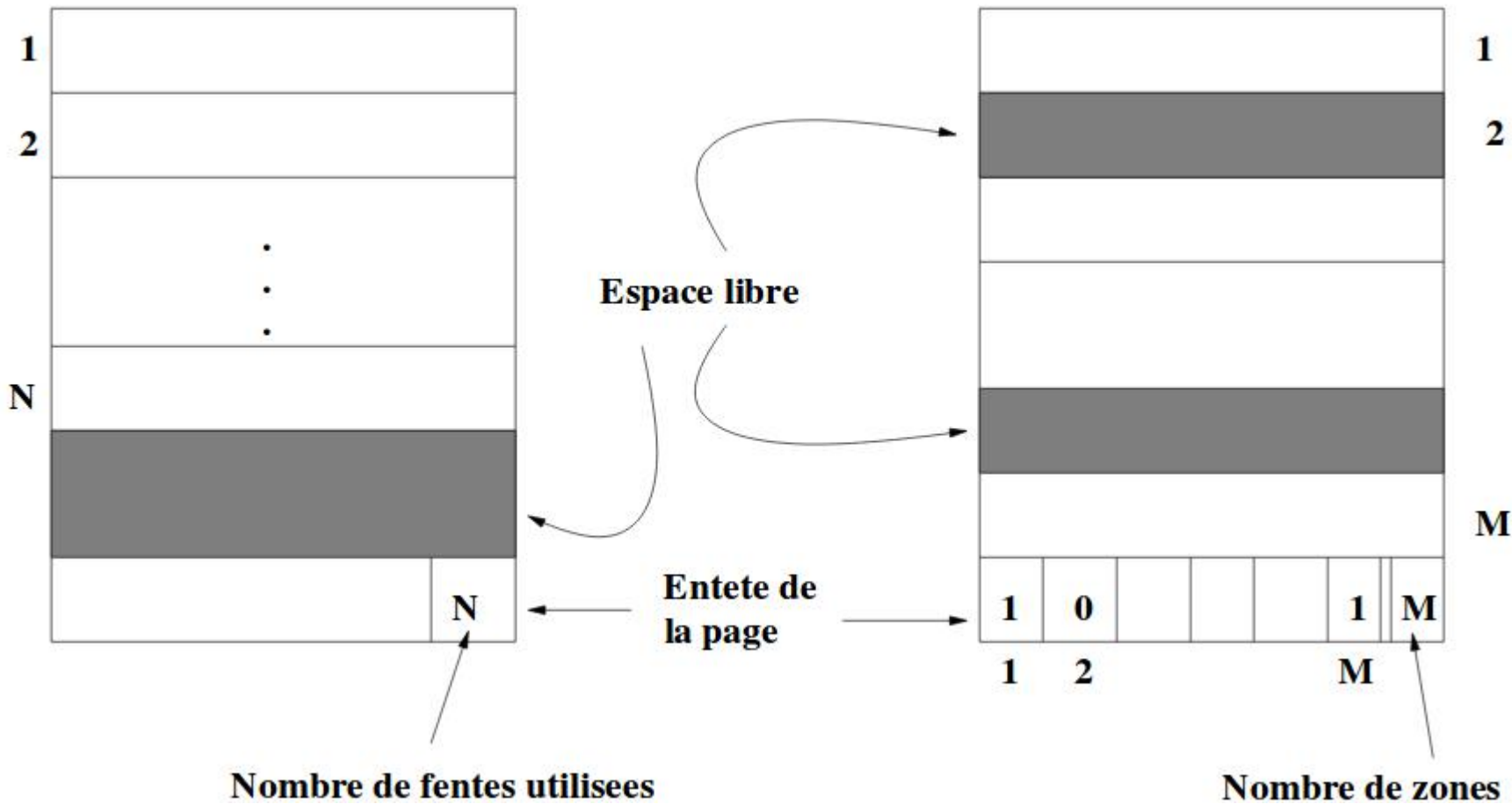


La (2) est plus intéressante surtout pour la gestion des valeurs nulles.
Un champ NULL alors les pointeurs DEB et FIN associées à ce champ sont égaux

Formats des pages

Enregistrements de taille fixe :

Deux variantes :

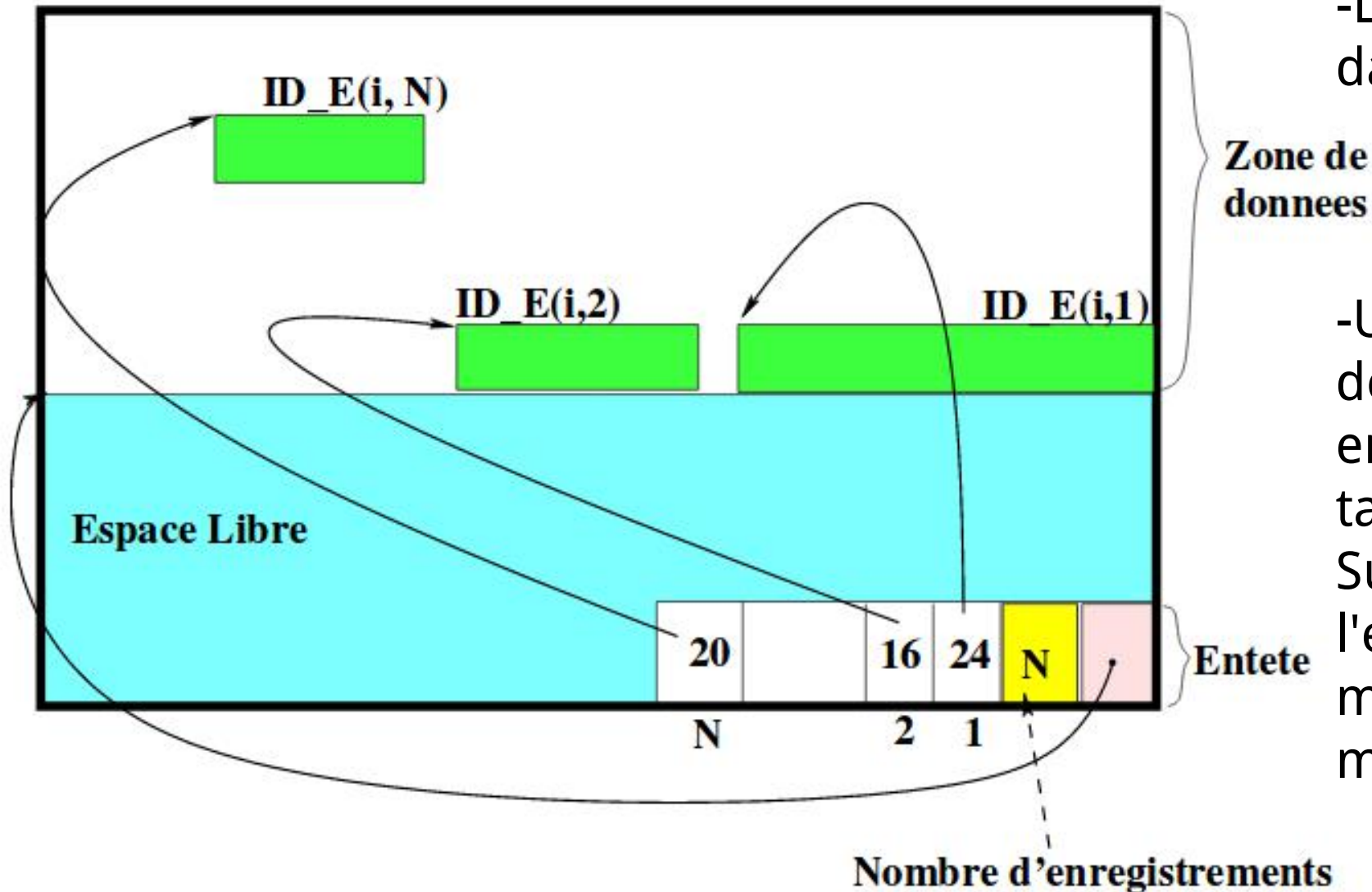


La page est composée de fentes (Slots) (Segments) chacune pouvant accueillir un enregistrement
 $ID_{Enr} = \langle Id_{page}, N^{\circ}_{fente} \rangle$

IBM DB2, Informix, SQL Server, Oracle utilisent la (2)

Formats des pages

Enregistrements de taille variable:



L'entête contient :

- Un pointeur vers le début de la zone libre.
- Le nombre d'enregistrements dans la page.

-Une liste de pointeurs vers le début de chaque enregistrement ainsi que sa taille.

Suite à une suppression, l'enregistrement peut bouger mais son ID_Eng n'est pas modifié.

A RETENIR ...

Le gestionnaire de buffer charge les pages dans la RAM.

Les pages restent dans le tampon jusqu'à ce qu'il n'y ait aucun processus qui ne les utilise.

Elles sont réécrites sur le disque après qu'elles soient libérées et quand la zone qu'elles occupent est choisie pour un remplacement.

Le choix de la zone à libérer est dictée par différentes techniques possibles (LRU ...)