Bases de données

Stockage

Nadime Francis

Université Gustave Eiffel
LIGM - 4B130 Copernic
nadime.francis@univ-eiffel.fr

Hiérarchie de mémoire

Types de mémoire : compromis entre rapidité et capacité

Mémoire interne : volatile, très limitée

- Registres
- Cache

Mémoire principale : volatile

- RAM

Mémoire secondaire : persistante, accès direct

- Mémoire flash
- Disque magnétique

Mémoire tertiaire : persistante, peu ou pas d'écriture, très lente

- Disque optique
- Bande magnétique

Échelle de temps des opérations du système

Source : Brendan Gregg, Systems performance : Entreprise and the Cloud

Opération	Durée	À échelle humaine
Un cycle du processeur	0.3 ns	1 s
Accès cache de niveau 1	0.9 ns	3 s
Accès cache de niveau 2	2.8 ns	9 s
Accès cache de niveau 3	12.9 ns	43 s
Accès DRAM	120 ns	6 min
Accès mémoire flash	100 µs	4 jours
Accès disque dur	1-10 ms	1-12 mois
Internet (continental)	40 ms	4 ans
Internet (inter-continental)	80 ms	8 ans
Retransmission TCP	1-3 s	100-300 ans
Reboot physique	5 m	32 millénaires

Stockage et accès aux données

Stockage en mémoire secondaire

- Mémoire persistante
- Bon rapport volume/prix
- Accès direct (online)

Accès aux données en mémoire principale

- Copie des données demandées en RAM
- Lecture et écriture efficace
- Recopie en mémoire secondaire pour stockage

Stockage et accès aux données

Stockage en mémoire secondaire

- Mémoire persistante
- Bon rapport volume/prix
- Accès direct (online)

Accès aux données en mémoire principale

- Copie des données demandées en RAM
- Lecture et écriture efficace
- Recopie en mémoire secondaire pour stockage

Question : que faire quand la mémoire secondaire ne suffit pas ?

Stockage et accès aux données

Stockage en mémoire secondaire

- Mémoire persistante
- Bon rapport volume/prix
- Accès direct (online)

Accès aux données en mémoire principale

- Copie des données demandées en RAM
- Lecture et écriture efficace
- Recopie en mémoire secondaire pour stockage

Question : que faire quand la mémoire secondaire ne suffit pas ?

- Archivage en mémoire tertiaire (offline, peu de lecture / écriture)
- Bases de données distribuées sur le réseau (Cloud)

Modèle physique des données

Encodage des données

Champ : séquence d'octets encodant une valeur d'un type de base

- Un entier 32 bits (4 octets)

00101101	00111010	11111111	00001100
45	58	255	12

- Une chaine de 8 caractères

H	a	r	r	y	\0	\0	\0

- Et d'autres : float, numeric, varchar, etc

Enregistrement : séquence de champs

0	1	H	a	r	r	y	\0	\0	\0	22	0	2

- Format : ordres et types des champs qui composent l'enregistrement
  Dans l'exemple : (entier 16 bits, chaine 8 octets, entier 8 bits, entier 16 bits)
- Différences entre format (physique) et schéma (logique)
  Champs de "gestion" : pointeurs, longueur d'un varchar, etc.

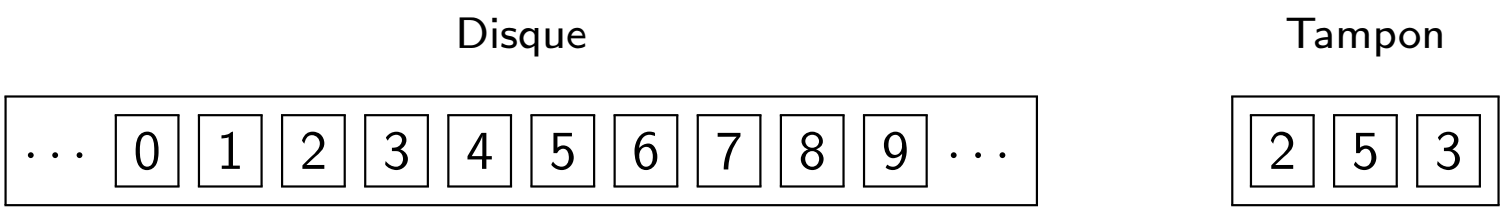
Blocs et mémoire tampon

Bloc (ou page) :

- Séquence d'octets contigus du disque et adressable par le système
- Quelques ko (configurable, doit respecter la géométrie du disque)
- Unité de transfert du disque vers la mémoire principale

Mémoire tampon (buffer) :

- Partie de la mémoire principale où sont copiés les blocs transférés
- Maintient plusieurs blocs pour réduire le nombre d'accès au disque
- Lectures et écritures efficaces dans le tampon



Gestionnaire de tampon

Gestionnaire de tampon (buffer manager) :

- Module du système chargé des transferts entre le disque et le tampon
- Le SGBD implémente généralement son propre buffer manager
  $\rightarrow$ capable de choix plus experts que l'OS

Gestionnaire de tampon

Gestionnaire de tampon (buffer manager) :

- Module du système chargé des transferts entre le disque et le tampon
- Le SGBD implémente généralement son propre buffer manager
  $\rightarrow$ capable de choix plus experts que l'OS

Lors d'une demande de lecture ou d'écriture, le buffer manager :

1. Vérifie si le bloc contenant la donnée demandée est dans le tampon
2. S'il n'y est pas, charge tout le bloc concerné dans le tampon
3. Effectue l'opération demandée dans le tampon

Gestionnaire de tampon

Gestionnaire de tampon (buffer manager) :

- Module du système chargé des transferts entre le disque et le tampon
- Le SGBD implémente généralement son propre buffer manager
  $\rightarrow$ capable de choix plus experts que l'OS

Lors d'une demande de lecture ou d'écriture, le buffer manager :

1. Vérifie si le bloc contenant la donnée demandée est dans le tampon
2. S'il n'y est pas, charge tout le bloc concerné dans le tampon
3. Effectue l'opération demandée dans le tampon

Si le tampon est plein :

1. Recopie si nécessaire (flush) un bloc du tampon vers le disque
2. Libère la mémoire allouée au bloc concerné

Stratégies pour choisir quel bloc libérer : least recently used, FIFO, ...

Organisation d'un bloc de données

Le bloc contient une entête qui stocke :

- Le nombre d'enregistrements dans le bloc
- Un pointeur vers la fin de l'espace inutilisé
- Un pointeur vers chaque enregistrement (avec sa taille)

La structure du bloc est maintenue lors des lectures et écritures :

Organisation d'un bloc de données

Le bloc contient une entête qui stocke :

- Le nombre d'enregistrements dans le bloc
- Un pointeur vers la fin de l'espace inutilisé
- Un pointeur vers chaque enregistrement (avec sa taille)

La structure du bloc est maintenue lors des lectures et écritures :

Ex :

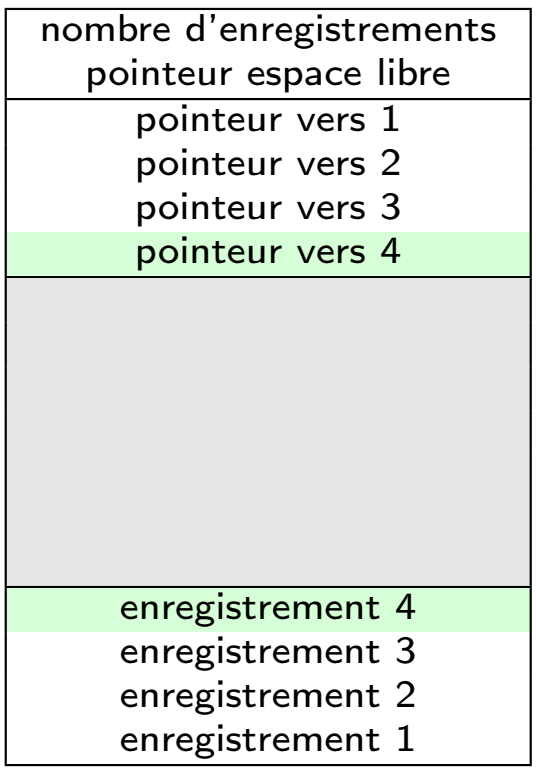
- Insertion de l'enregistrement 4

Organisation d'un bloc de données

Le bloc contient une entête qui stocke :

- Le nombre d'enregistrements dans le bloc
- Un pointeur vers la fin de l'espace inutilisé
- Un pointeur vers chaque enregistrement (avec sa taille)

La structure du bloc est maintenue lors des lectures et écritures :



Ex :

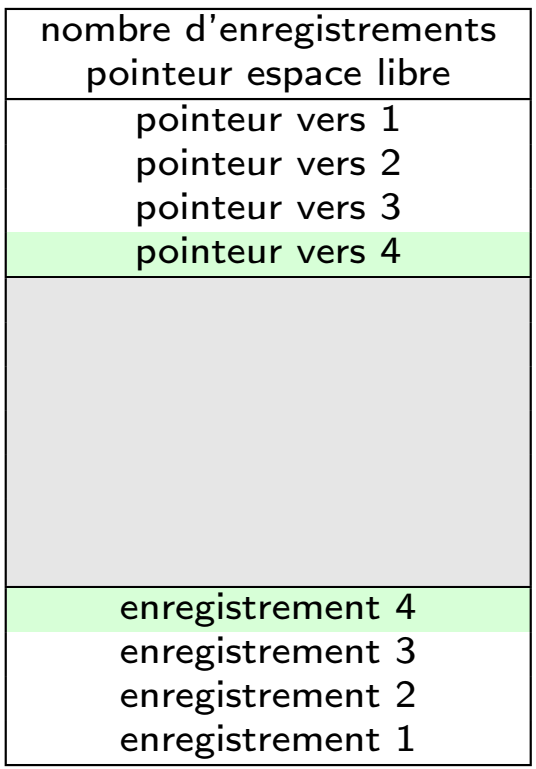
- Insertion de l'enregistrement 4

Organisation d'un bloc de données

Le bloc contient une entête qui stocke :

- Le nombre d'enregistrements dans le bloc
- Un pointeur vers la fin de l'espace inutilisé
- Un pointeur vers chaque enregistrement (avec sa taille)

La structure du bloc est maintenue lors des lectures et écritures :



Ex :

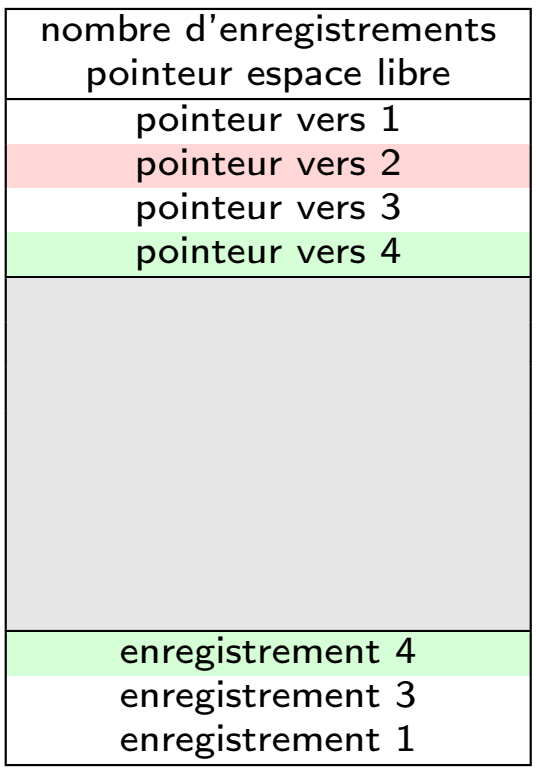
- Insertion de l'enregistrement 4
- Suppression de l'enregistrement 2

Organisation d'un bloc de données

Le bloc contient une entête qui stocke :

- Le nombre d'enregistrements dans le bloc
- Un pointeur vers la fin de l'espace inutilisé
- Un pointeur vers chaque enregistrement (avec sa taille)

La structure du bloc est maintenue lors des lectures et écritures :



Ex :

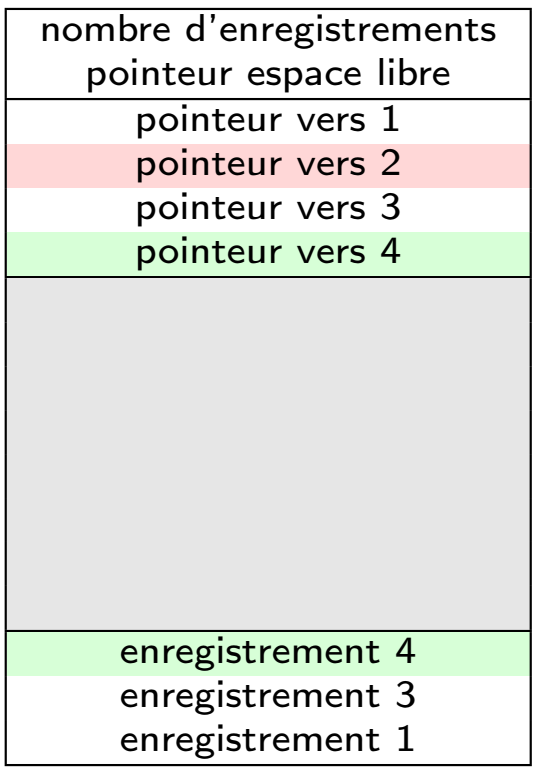
- Insertion de l'enregistrement 4
- Suppression de l'enregistrement 2

Organisation d'un bloc de données

Le bloc contient une entête qui stocke :

- Le nombre d'enregistrements dans le bloc
- Un pointeur vers la fin de l'espace inutilisé
- Un pointeur vers chaque enregistrement (avec sa taille)

La structure du bloc est maintenue lors des lectures et écritures :



Ex :

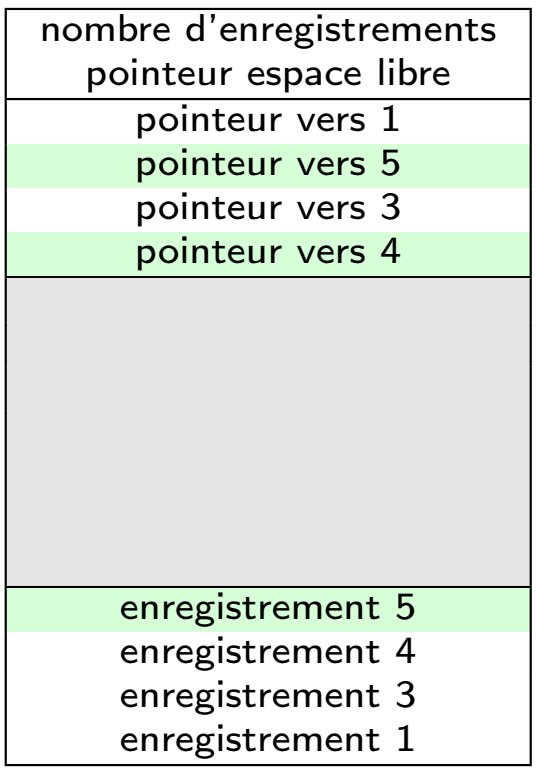
- Insertion de l'enregistrement 4
- Suppression de l'enregistrement 2
- Insertion de l'enregistrement 5

Organisation d'un bloc de données

Le bloc contient une entête qui stocke :

- Le nombre d'enregistrements dans le bloc
- Un pointeur vers la fin de l'espace inutilisé
- Un pointeur vers chaque enregistrement (avec sa taille)

La structure du bloc est maintenue lors des lectures et écritures :



Ex :

- Insertion de l'enregistrement 4
- Suppression de l'enregistrement 2
- Insertion de l'enregistrement 5

Entête : remarques

L'entête permet une gestion d'adresses logiques des enregistrements :

- Les pointeurs extérieurs au bloc pointent dans l'entête plutôt que directement sur les enregistrements
- Les réorganisations du bloc sont invisibles de l'extérieur, seule l'entête doit être tenue à jour

Entête : remarques

L'entête permet une gestion d'adresses logiques des enregistrements :

- Les pointeurs extérieurs au bloc pointent dans l'entête plutôt que directement sur les enregistrements
- Les réorganisations du bloc sont invisibles de l'extérieur, seule l'entête doit être tenue à jour

Les enregistrements sont compactés à la fin du bloc pour permettre à l'entête de grandir avec l'ajout de nouveaux enregistrements

Entête : remarques

L'entête permet une gestion d'adresses logiques des enregistrements :

- Les pointeurs extérieurs au bloc pointent dans l'entête plutôt que directement sur les enregistrements
- Les réorganisations du bloc sont invisibles de l'extérieur, seule l'entête doit être tenue à jour

Les enregistrements sont compactés à la fin du bloc pour permettre à l'entête de grandir avec l'ajout de nouveaux enregistrements

Lorsqu'un bloc contient uniquement des enregistrements de taille fixe :

- L'entête n'est pas nécessaire : on peut stocker les enregistrements séquentiellement, autant que le bloc peut en contenir
- Elle est malgré tout utilisée pour faciliter les réorganisations du bloc

Fichier de données

Quelques ordres de grandeur :

- Un enregistrement :
- Un bloc :
- Une table :

Fichier de données

Quelques ordres de grandeur :

- Un enregistrement : quelques dizaines à quelques centaines d'octets
- Un bloc : quelques kilo-octets
- Une table : pas de limite ! peut se compter en tera-octets

Fichier de données

Quelques ordres de grandeur :

- Un enregistrement : quelques dizaines à quelques centaines d'octets
- Un bloc : quelques kilo-octets
- Une table : pas de limite ! peut se compter en tera-octets

Conclusion :

- Un bloc contient généralement de nombreux d'enregistrements
- Une table peut nécessiter beaucoup de blocs

Fichier de données

Quelques ordres de grandeur :

- Un enregistrement : quelques dizaines à quelques centaines d'octets
- Un bloc : quelques kilo-octets
- Une table : pas de limite ! peut se compter en tera-octets

Conclusion :

- Un bloc contient généralement de nombreux d'enregistrements
- Une table peut nécessiter beaucoup de blocs

Fichier de données : encodage physique d'une table

- Ensemble des blocs qui stockent les enregistrements de la table
- Les blocs ne sont généralement pas contigus en mémoire
- En général, les blocs ne mélangent pas de données de plusieurs tables

Organisation d'un fichier de données

Choix d'organisation du fichier

Un fichier de données contient généralelement de nombreux blocs
Plusieurs approches possibles pour organiser les blocs :

- Organisation en tas de données (heap file)
- Organisation séquentielle
  (sequential file, ISAM : indexed sequential access method)
- Organisation en grappe (cluster file)

Choix d'organisation du fichier

Un fichier de données contient généralelement de nombreux blocs
Plusieurs approches possibles pour organiser les blocs :

- Organisation en tas de données (heap file)
- Organisation séquentielle
  (sequential file, ISAM : indexed sequential access method)
- Organisation en grappe (cluster file)

D'autres possibilités, sur mesure selon l'usage ou les ordres de grandeurs :

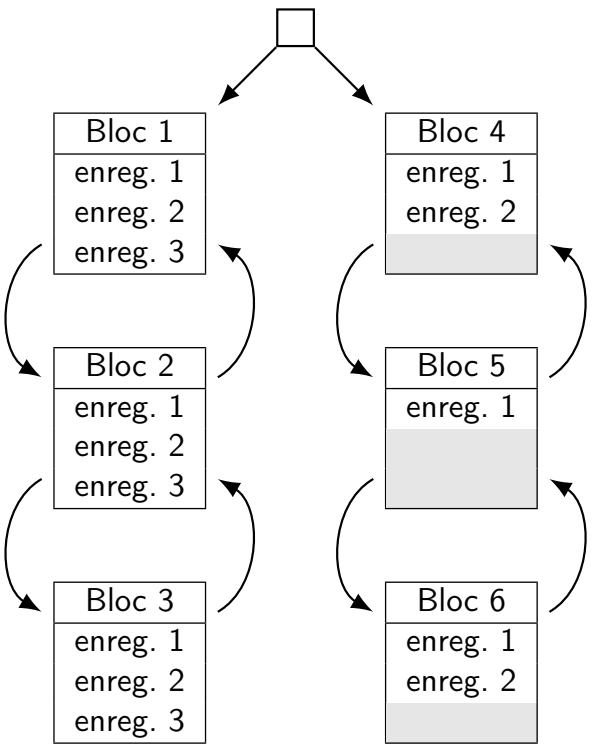
- Enregistrements très volumineux (plus qu'un bloc)
- Enregistrements très courts, jointures très fréquentes
- Nombreuses tables, peu d'enregistrement par table
- ...

Organisation en tas

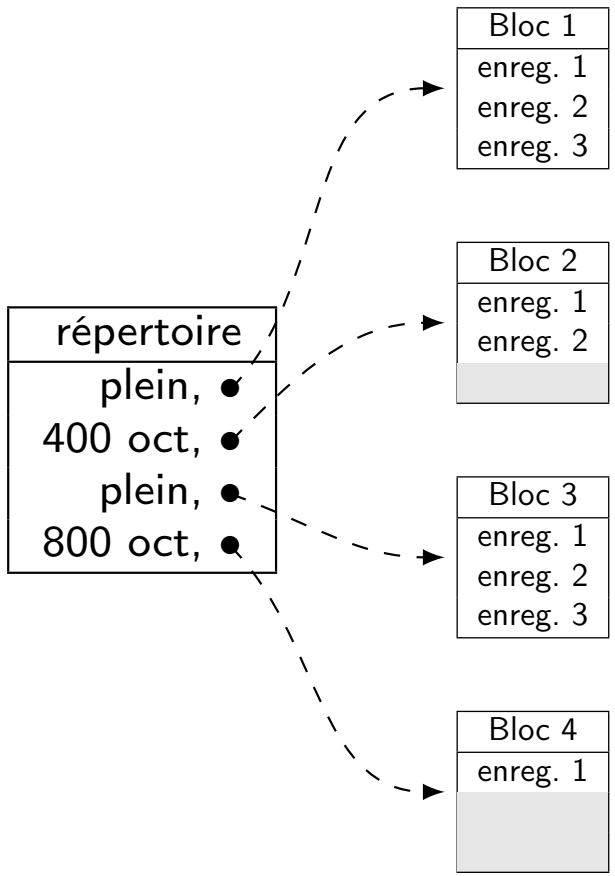
Les enregistrements sont répartis dans les blocs sans ordre particulier

Deux approches pour organiser les blocs entre eux :

Liste doublement chaînée



Répertoire (directory)



Organisation en tas : insertion

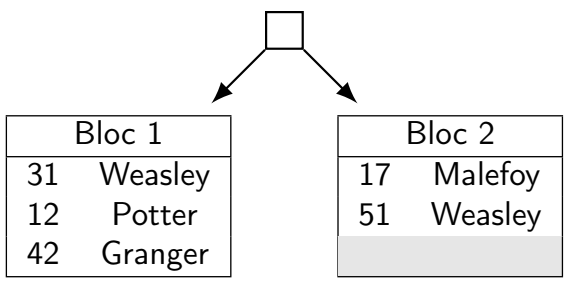
- Recherche d'un bloc avec suffisamment de place
- Si tous les blocs sont pleins, création d'un nouveau bloc
- Mise à jour de la structure (liste chaînée ou répertoire)

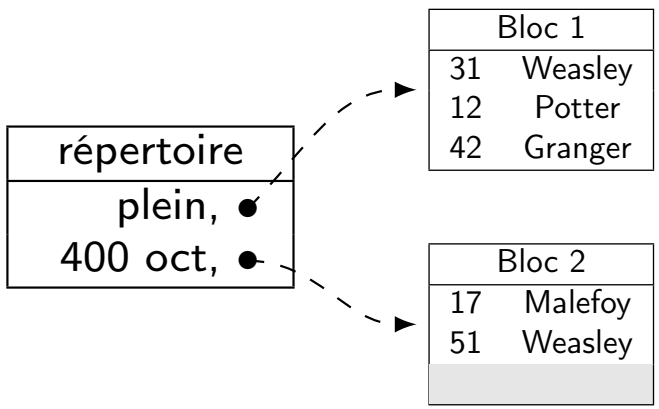
Organisation en tas : insertion

- Recherche d'un bloc avec suffisamment de place
- Si tous les blocs sont pleins, création d'un nouveau bloc
- Mise à jour de la structure (liste chaînée ou répertoire)

Ex :

- Insertion de (43, Crabbe)



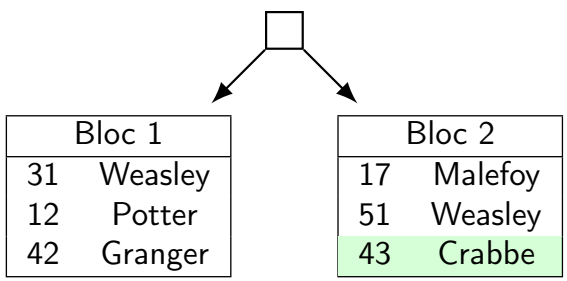


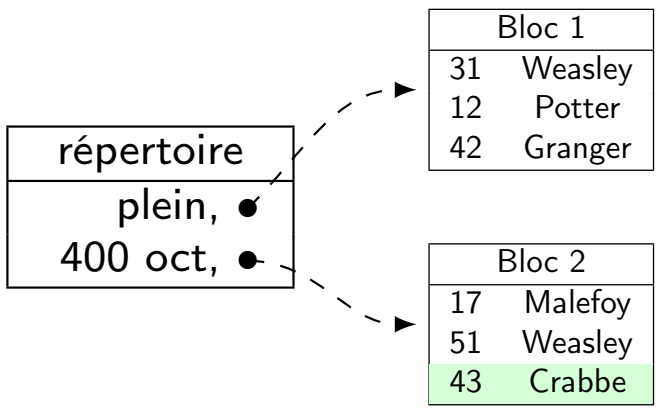
Organisation en tas : insertion

- Recherche d'un bloc avec suffisamment de place
- Si tous les blocs sont pleins, création d'un nouveau bloc
- Mise à jour de la structure (liste chaînée ou répertoire)

Ex :

- Insertion de (43, Crabbe)



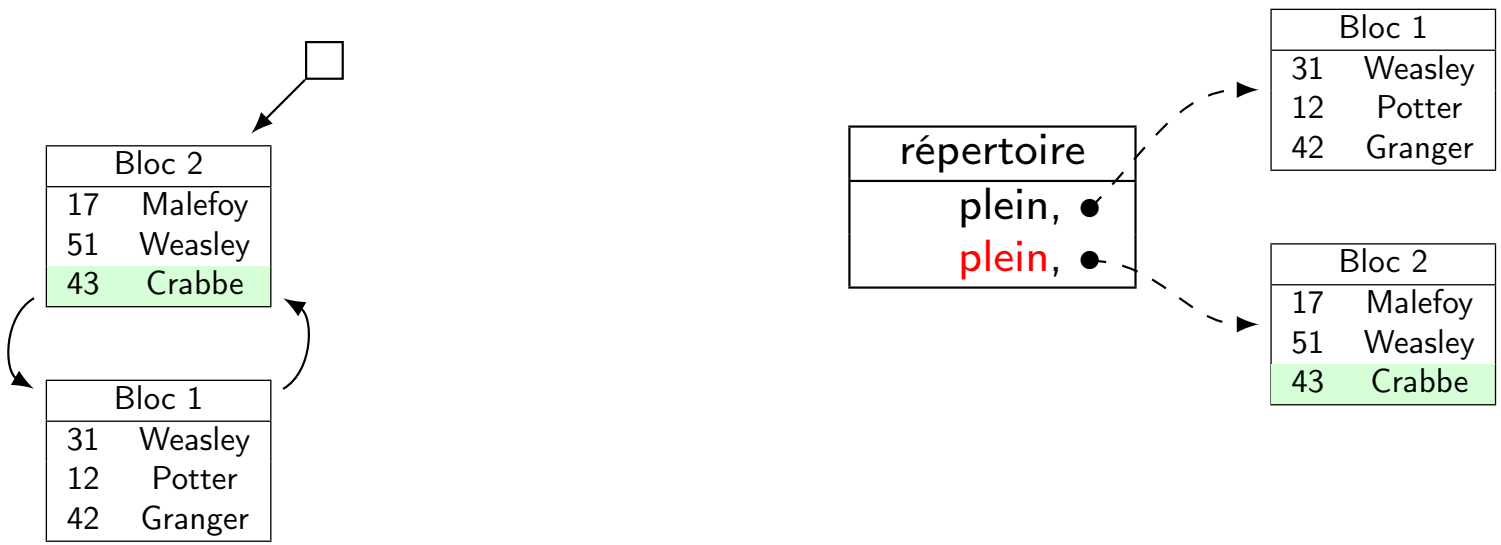


Organisation en tas : insertion

- Recherche d'un bloc avec suffisamment de place
- Si tous les blocs sont pleins, création d'un nouveau bloc
- Mise à jour de la structure (liste chaînée ou répertoire)

Ex :

- Insertion de (43, Crabbe)

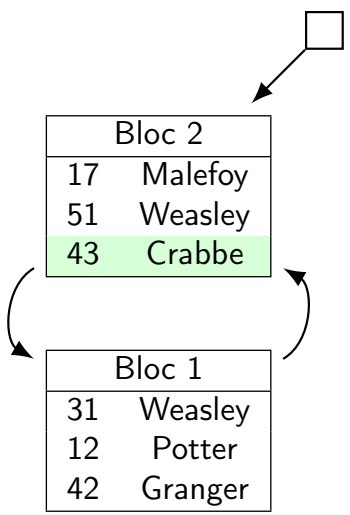


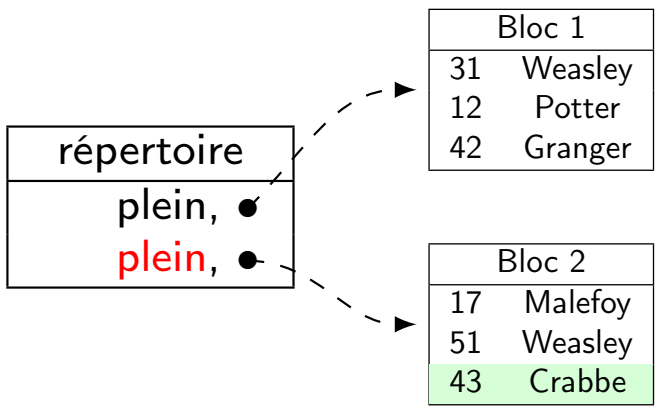
Organisation en tas : insertion

- Recherche d'un bloc avec suffisamment de place
- Si tous les blocs sont pleins, création d'un nouveau bloc
- Mise à jour de la structure (liste chaînée ou répertoire)

Ex :

- Insertion de (43, Crabbe)
- Insertion de (16, Goyle)



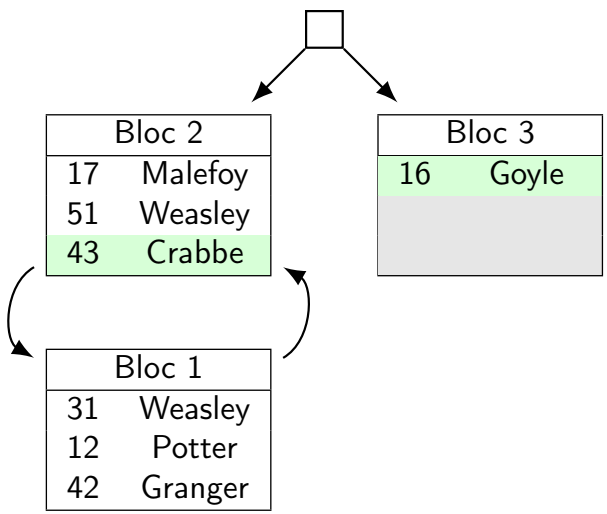


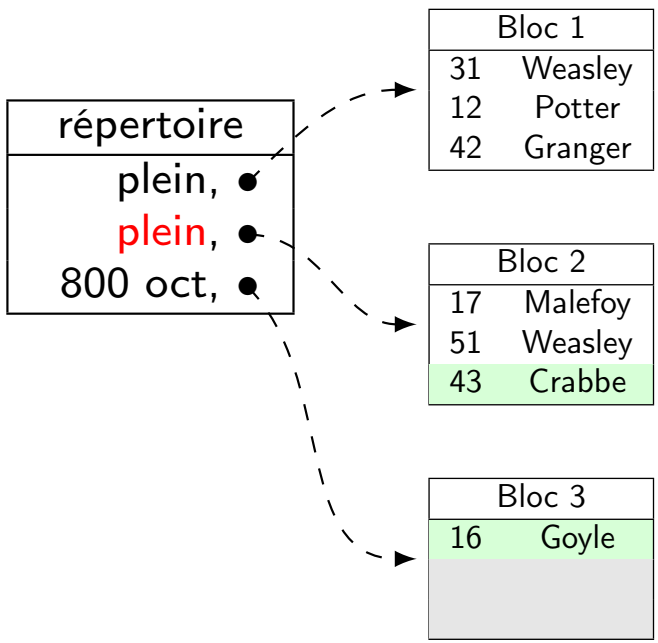
Organisation en tas : insertion

- Recherche d'un bloc avec suffisamment de place
- Si tous les blocs sont pleins, création d'un nouveau bloc
- Mise à jour de la structure (liste chaînée ou répertoire)

Ex :

- Insertion de (43, Crabbe)
- Insertion de (16, Goyle)





Organisation en tas : suppression

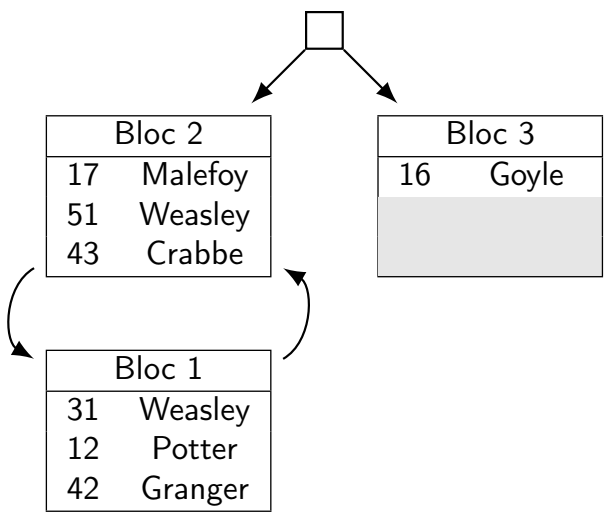
- Recherche exhaustive pour trouver l'enregistrement
- Suppression de l'enregistrement et des blocs vides
- Mise à jour de la structure (liste chaînée ou répertoire)

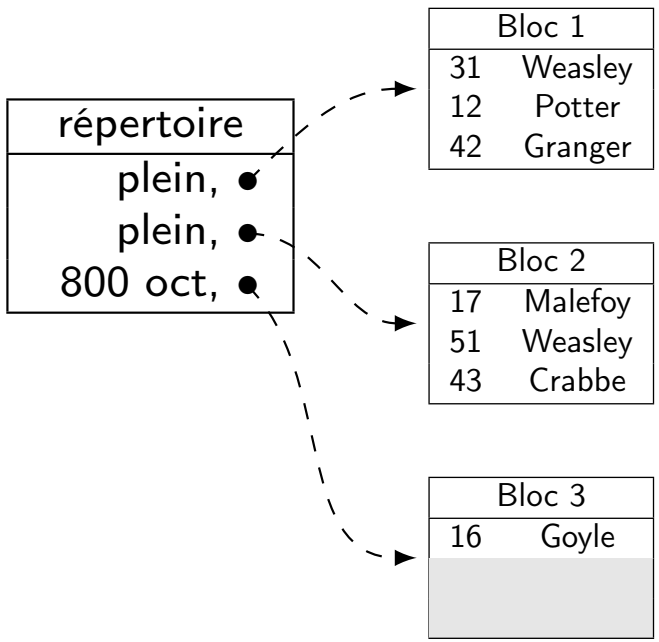
Organisation en tas : suppression

- Recherche exhaustive pour trouver l'enregistrement
- Suppression de l'enregistrement et des blocs vides
- Mise à jour de la structure (liste chaînée ou répertoire)

Ex :

- Suppression de (12, Potter)



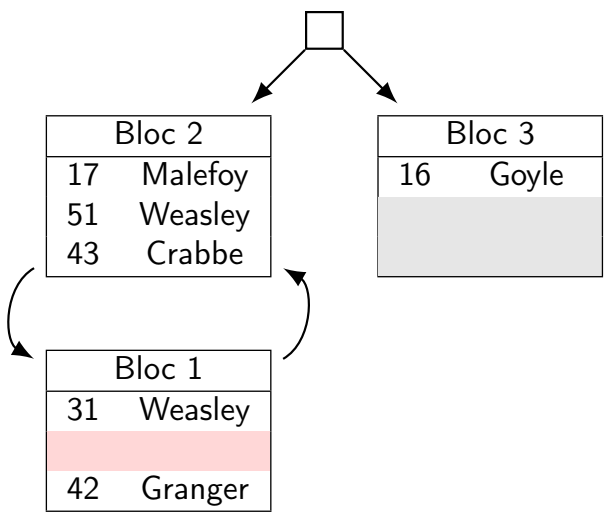


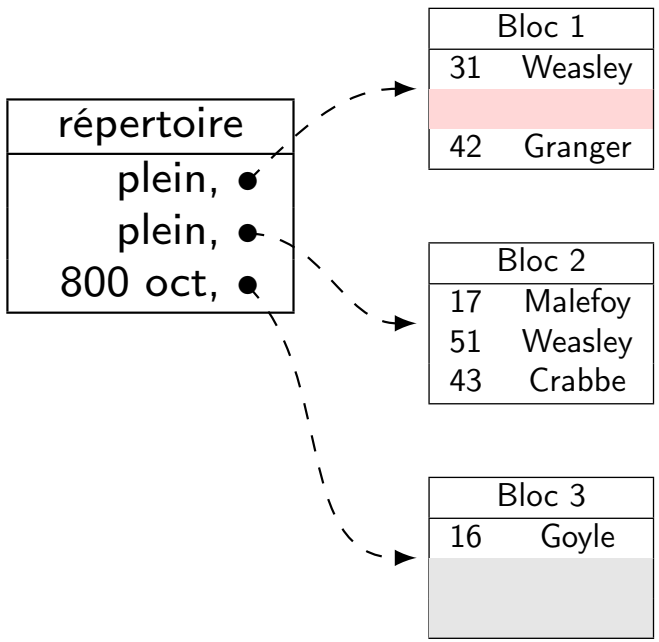
Organisation en tas : suppression

- Recherche exhaustive pour trouver l'enregistrement
- Suppression de l'enregistrement et des blocs vides
- Mise à jour de la structure (liste chaînée ou répertoire)

Ex :

- Suppression de (12, Potter)



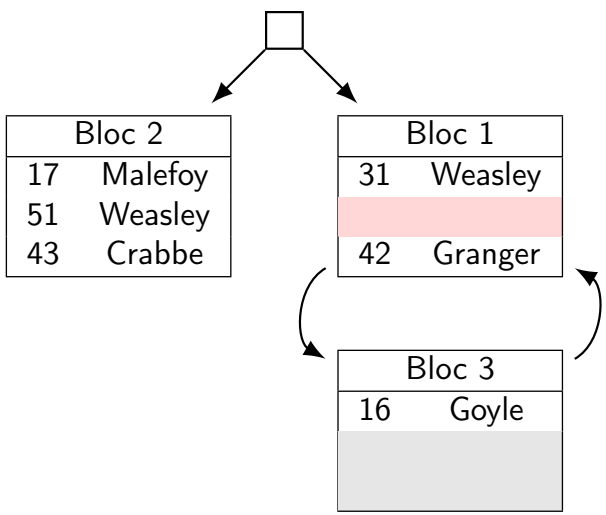


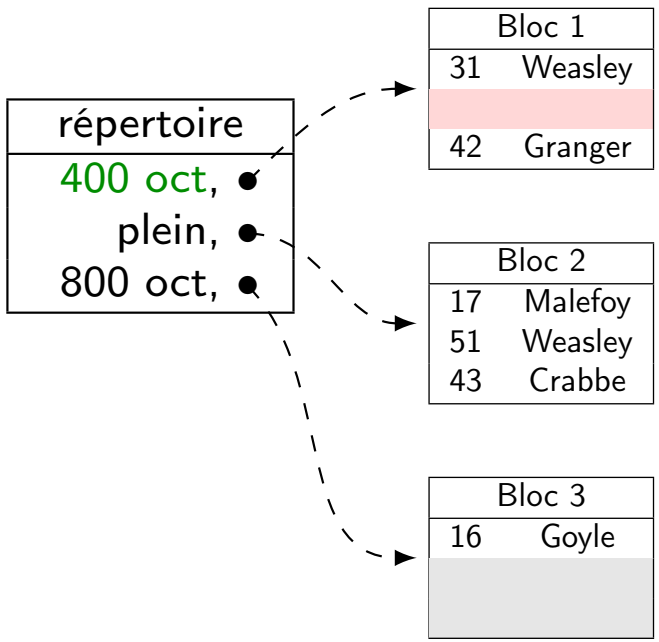
Organisation en tas : suppression

- Recherche exhaustive pour trouver l'enregistrement
- Suppression de l'enregistrement et des blocs vides
- Mise à jour de la structure (liste chaînée ou répertoire)

Ex :

- Suppression de (12, Potter)



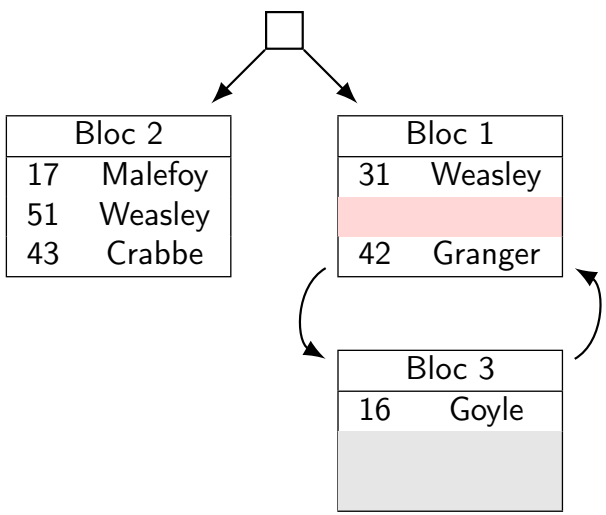


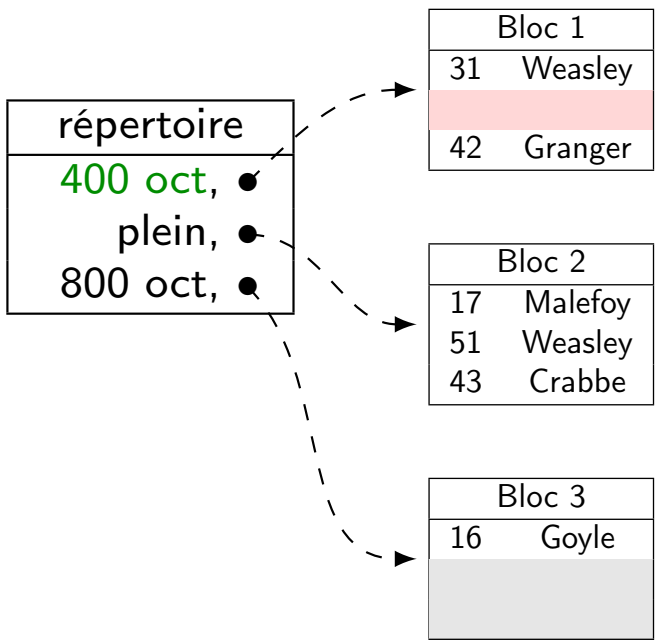
Organisation en tas : suppression

- Recherche exhaustive pour trouver l'enregistrement
- Suppression de l'enregistrement et des blocs vides
- Mise à jour de la structure (liste chaînée ou répertoire)

Ex :

- Suppression de (12, Potter)
- Suppression de (16, Goyle)



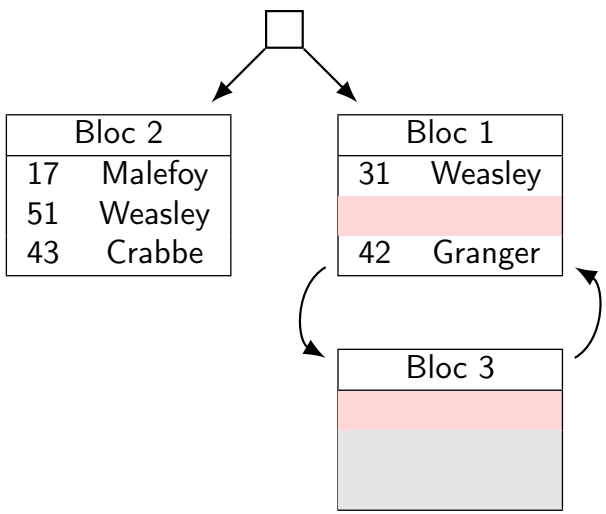


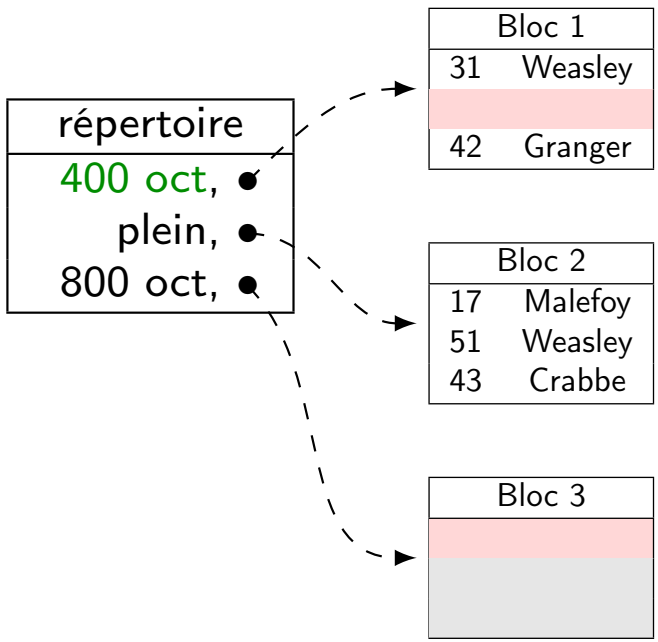
Organisation en tas : suppression

- Recherche exhaustive pour trouver l'enregistrement
- Suppression de l'enregistrement et des blocs vides
- Mise à jour de la structure (liste chaînée ou répertoire)

Ex :

- Suppression de (12, Potter)
- Suppression de (16, Goyle)



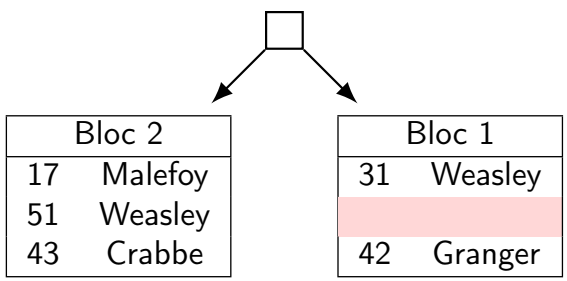


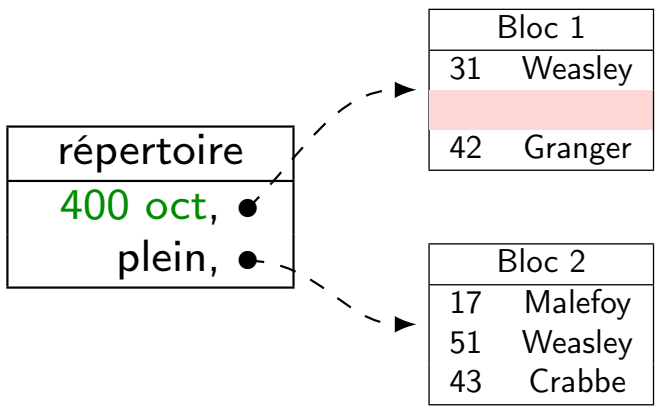
Organisation en tas : suppression

- Recherche exhaustive pour trouver l'enregistrement
- Suppression de l'enregistrement et des blocs vides
- Mise à jour de la structure (liste chaînée ou répertoire)

Ex :

- Suppression de (12, Potter)
- Suppression de (16, Goyle)





Organisation séquentielle

Les enregistrements sont répartis dans les blocs de façon ordonnée

Clef de recherche :

- Ensemble de champs choisi comme critère de tri
- Usuellement, mais pas nécessairement, la clef primaire de la table

Index :

- Structure auxiliaire utilisée pour maintenir l'ordre des blocs
- Mémorise les plages de clefs pouvant apparaître dans chaque bloc

Organisation séquentielle

Les enregistrements sont répartis dans les blocs de façon ordonnée

Clef de recherche :

- Ensemble de champs choisi comme critère de tri
- Usuellement, mais pas nécessairement, la clef primaire de la table

Index :

- Structure auxiliaire utilisée pour maintenir l'ordre des blocs
- Mémorise les plages de clefs pouvant apparaître dans chaque bloc

Avantages :

- Rapidité d'accès aux enregistrements par clef de recherche
- Seul l'index et le bloc concerné sont chargés en mémoire principale

Organisation séquentielle

Les enregistrements sont répartis dans les blocs de façon ordonnée

Clef de recherche :

- Ensemble de champs choisi comme critère de tri
- Usuellement, mais pas nécessairement, la clef primaire de la table

Index :

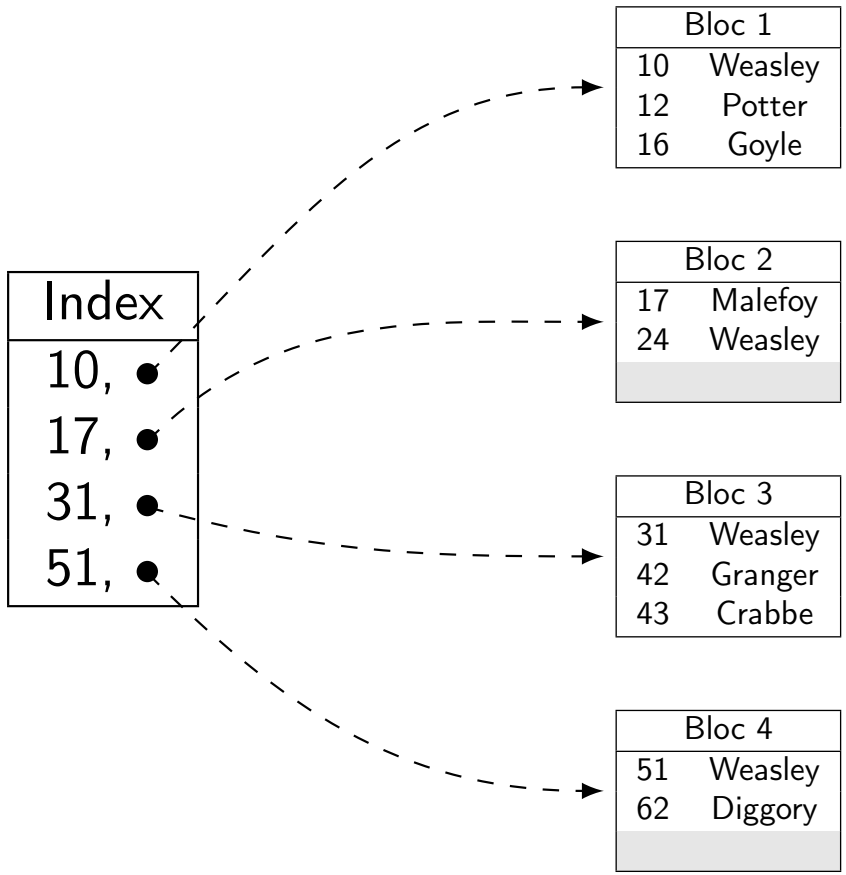
- Structure auxiliaire utilisée pour maintenir l'ordre des blocs
- Mémorise les plages de clefs pouvant apparaître dans chaque bloc

Avantages :

- Rapidité d'accès aux enregistrements par clef de recherche
- Seul l'index et le bloc concerné sont chargés en mémoire principale

Remarque : le fonctionnement des index est l'objet du prochain chapitre !

Exemple : organisation séquentielle et index non-dense



Remarque : les blocs 1 à 4 peuvent ne pas être contigus en mémoire

Organisation séquentielle : insertion

Insertion

- L'index fournit le bloc où faire l'insertion
- L'enregistrement est inséré s'il y a assez de place libre (avec déplacements éventuels vers les blocs voisins)
- Sinon, un nouveau bloc est inséré au bon endroit

Bloc 1	
10	Weasley
12	Potter
16	Goyle

Bloc 2	
17	Malefoy
24	Weasley

Bloc 3	
31	Weasley
42	Granger
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Organisation séquentielle : insertion

Bloc 1	
10	Weasley
12	Potter
16	Goyle

Bloc 2	
17	Malefoy
24	Weasley

Bloc 3	
31	Weasley
42	Granger
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Insertion

- L'index fournit le bloc où faire l'insertion
- L'enregistrement est inséré s'il y a assez de place libre (avec déplacements éventuels vers les blocs voisins)
- Sinon, un nouveau bloc est inséré au bon endroit

Ex :

- Insertion de (11, Patil)

Organisation séquentielle : insertion

Insertion

- L'index fournit le bloc où faire l'insertion
- L'enregistrement est inséré s'il y a assez de place libre (avec déplacements éventuels vers les blocs voisins)
- Sinon, un nouveau bloc est inséré au bon endroit

Ex :

- Insertion de (11, Patil)

Bloc 1	
10	Weasley
12	Potter
16	Goyle

Bloc 2	
17	Malefoy
24	Weasley

Bloc 3	
31	Weasley
42	Granger
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Organisation séquentielle : insertion

Bloc 1	
10	Weasley
12	Potter

Bloc 2	
16	Goyle
17	Malefoy
24	Weasley

Bloc 3	
31	Weasley
42	Granger
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Insertion

- L'index fournit le bloc où faire l'insertion
- L'enregistrement est inséré s'il y a assez de place libre (avec déplacements éventuels vers les blocs voisins)
- Sinon, un nouveau bloc est inséré au bon endroit

Ex :

- Insertion de (11, Patil)

Organisation séquentielle : insertion

Insertion

- L'index fournit le bloc où faire l'insertion
- L'enregistrement est inséré s'il y a assez de place libre (avec déplacements éventuels vers les blocs voisins)
- Sinon, un nouveau bloc est inséré au bon endroit

Ex :

- Insertion de (11, Patil)

Bloc 1	
10	Weasley
11	Patil
12	Potter

Bloc 2	
16	Goyle
17	Malefoy
24	Weasley

Bloc 3	
31	Weasley
42	Granger
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Organisation séquentielle : insertion

Bloc 1	
10	Weasley
11	Patil
12	Potter

Bloc 2	
16	Goyle
17	Malefoy
24	Weasley

Bloc 3	
31	Weasley
42	Granger
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Insertion

- L'index fournit le bloc où faire l'insertion
- L'enregistrement est inséré s'il y a assez de place libre (avec déplacements éventuels vers les blocs voisins)
- Sinon, un nouveau bloc est inséré au bon endroit

Ex :

- Insertion de (11, Patil)
- Insertion de (18, Patil)

Organisation séquentielle : insertion

Insertion

- L'index fournit le bloc où faire l'insertion
- L'enregistrement est inséré s'il y a assez de place libre (avec déplacements éventuels vers les blocs voisins)
- Sinon, un nouveau bloc est inséré au bon endroit

Ex :

- Insertion de (11, Patil)
- Insertion de (18, Patil)

Bloc 1	
10	Weasley
11	Patil
12	Potter

Bloc 2	
16	Goyle
17	Malefoy
24	Weasley

Bloc 5

Bloc 3	
31	Weasley
42	Granger
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Organisation séquentielle : insertion

Insertion

- L'index fournit le bloc où faire l'insertion
- L'enregistrement est inséré s'il y a assez de place libre (avec déplacements éventuels vers les blocs voisins)
- Sinon, un nouveau bloc est inséré au bon endroit

Ex :

- Insertion de (11, Patil)
- Insertion de (18, Patil)

Bloc 1	
10	Weasley
11	Patil
12	Potter

Bloc 2	
16	Goyle
17	Malefoy

Bloc 5	
24	Weasley

Bloc 3	
31	Weasley
42	Granger
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Organisation séquentielle : insertion

Insertion

- L'index fournit le bloc où faire l'insertion
- L'enregistrement est inséré s'il y a assez de place libre (avec déplacements éventuels vers les blocs voisins)
- Sinon, un nouveau bloc est inséré au bon endroit

Ex :

- Insertion de (11, Patil)
- Insertion de (18, Patil)

Bloc 1	
10	Weasley
11	Patil
12	Potter

Bloc 2	
16	Goyle
17	Malefoy
18	Patil

Bloc 5	
24	Weasley

Bloc 3	
31	Weasley
42	Granger
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Organisation séquentielle : suppression

Suppression

- L'index fournit le bloc où faire la suppression
- L'enregistrement est supprimé et le bloc est compacté
- Les blocs vides sont supprimés

Bloc 1	
10	Weasley
11	Patil
12	Potter

Bloc 2	
16	Goyle
17	Malefoy
18	Patil

Bloc 5	
24	Weasley

Bloc 3	
31	Weasley
42	Granger
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Organisation séquentielle : suppression

Suppression

- L'index fournit le bloc où faire la suppression
- L'enregistrement est supprimé et le bloc est compacté
- Les blocs vides sont supprimés

Ex :

- Suppression de (42, Granger)

Bloc 1	
10	Weasley
11	Patil
12	Potter

Bloc 2	
16	Goyle
17	Malefoy
18	Patil

Bloc 5	
24	Weasley

Bloc 3	
31	Weasley
42	Granger
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Organisation séquentielle : suppression

Suppression

- L'index fournit le bloc où faire la suppression
- L'enregistrement est supprimé et le bloc est compacté
- Les blocs vides sont supprimés

Ex :

- Suppression de (42, Granger)

Bloc 1	
10	Weasley
11	Patil
12	Potter

Bloc 2	
16	Goyle
17	Malefoy
18	Patil

Bloc 5	
24	Weasley

Bloc 3	
31	Weasley
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Organisation séquentielle : suppression

Suppression

- L'index fournit le bloc où faire la suppression
- L'enregistrement est supprimé et le bloc est compacté
- Les blocs vides sont supprimés

Ex :

- Suppression de (42, Granger)

Bloc 1	
10	Weasley
11	Patil
12	Potter

Bloc 2	
16	Goyle
17	Malefoy
18	Patil

Bloc 5	
24	Weasley

Bloc 3	
31	Weasley
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Organisation séquentielle : suppression

Suppression

- L'index fournit le bloc où faire la suppression
- L'enregistrement est supprimé et le bloc est compacté
- Les blocs vides sont supprimés

Ex :

- Suppression de (42, Granger)
- Suppression de (24, Weasley)

Bloc 1	
10	Weasley
11	Patil
12	Potter

Bloc 2	
16	Goyle
17	Malefoy
18	Patil

Bloc 5	
24	Weasley

Bloc 3	
31	Weasley
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Organisation séquentielle : suppression

Suppression

- L'index fournit le bloc où faire la suppression
- L'enregistrement est supprimé et le bloc est compacté
- Les blocs vides sont supprimés

Ex :

- Suppression de (42, Granger)
- Suppression de (24, Weasley)

Bloc 1	
10	Weasley
11	Patil
12	Potter

Bloc 2	
16	Goyle
17	Malefoy
18	Patil

Bloc 5

Bloc 3	
31	Weasley
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Organisation séquentielle : suppression

Bloc 1	
10	Weasley
11	Patil
12	Potter

Bloc 2	
16	Goyle
17	Malefoy
18	Patil

Bloc 3	
31	Weasley
43	Crabbe

Bloc 4	
51	Weasley
62	Diggory

Suppression

- L'index fournit le bloc où faire la suppression
- L'enregistrement est supprimé et le bloc est compacté
- Les blocs vides sont supprimés

Ex :

- Suppression de (42, Granger)
- Suppression de (24, Weasley)

Organisation séquentielle : réorganisation différée

L'index doit être mis-à-jour à chaque création ou suppression de bloc

Réorganisation différée :

- Stratégie pour remettre à plus tard les réorganisations du fichier
- Les insertions dans des blocs pleins sont temporairement stockées dans un bloc de débordement (overflow)
- Les suppressions sont simplement marquées et remises à plus tard

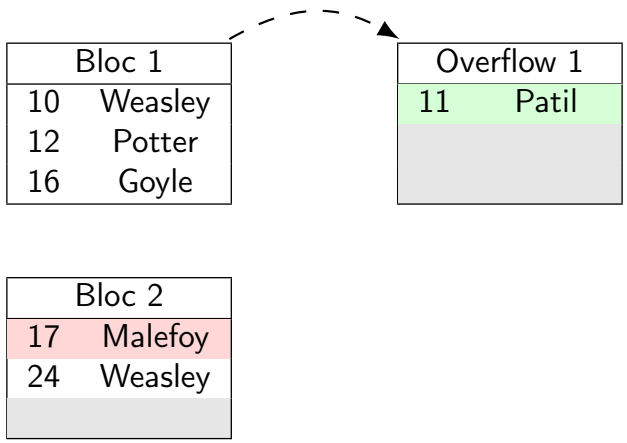
Organisation séquentielle : réorganisation différée

L'index doit être mis-à-jour à chaque création ou suppression de bloc

Réorganisation différée :

- Stratégie pour remettre à plus tard les réorganisations du fichier
- Les insertions dans des blocs pleins sont temporairement stockées dans un bloc de débordement (overflow)
- Les suppressions sont simplement marquées et remises à plus tard

Ex : insertion de (11, Patil) et suppression de (17, Malefoy)



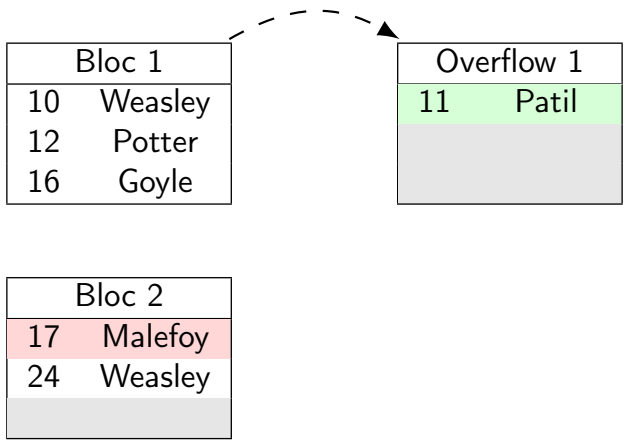
Organisation séquentielle : réorganisation différée

L'index doit être mis-à-jour à chaque création ou suppression de bloc

Réorganisation différée :

- Stratégie pour remettre à plus tard les réorganisations du fichier
- Les insertions dans des blocs pleins sont temporairement stockées dans un bloc de débordement (overflow)
- Les suppressions sont simplement marquées et remises à plus tard

Ex : insertion de (11, Patil) et suppression de (17, Malefoy)



Entretien périodique : résorber les débordements et reconstruire l'ordre

Organisation en grappe

Organisation particulière pour accélerer une jointure $n \rightarrow 1$ très fréquente

- Les deux tables à joindre sont stockées dans le même fichier
- Les enregistrements de la deuxième table sont intercalés avec les enregistrements auxquels ils font référence

Organisation en grappe

Organisation particulière pour accélerer une jointure $n \to 1$ très fréquente

- Les deux tables à joindre sont stockées dans le même fichier
- Les enregistrements de la deuxième table sont intercalés avec les enregistrements auxquels ils font référence

Ex : *departement*(`id, nom`) *etudiant*(`id, nom, prenom, id_dep`)

Bloc 1		
1	Gryffondor	
12	Potter	Harry
24	Weasley	Ron
42	Granger	Hermione

Bloc 2		
10	Weasley	Fred
31	Weasley	George
11	Patil	Parvati
51	Weasley	Ginny

Bloc 3		
73	Weasley	Percy
2	Serpentard	
17	Malefoy	Drago
16	Goyle	Gregory

Bloc 4		
43	Crabbe	Vincent
3	Serdaigle	
66	Lovegood	Luna
4	Poufsouffle	

Organisation en grappe

Organisation particulière pour accélerer une jointure $n \rightarrow 1$ très fréquente

- Les deux tables à joindre sont stockées dans le même fichier
- Les enregistrements de la deuxième table sont intercalés avec les enregistrements auxquels ils font référence

Ex : *departement*(`id, nom`) *etudiant*(`id, nom, prenom, id_dep`)

Bloc 1		
1	Gryffondor	
12	Potter	Harry
24	Weasley	Ron
42	Granger	Hermione

Bloc 2		
10	Weasley	Fred
31	Weasley	George
11	Patil	Parvati
51	Weasley	Ginny

Bloc 3		
73	Weasley	Percy
2	Serpentard	
17	Malefoy	Drago
16	Goyle	Gregory

Bloc 4		
43	Crabbe	Vincent
3	Serdaigle	
66	Lovegood	Luna
4	Poufsouffle	

Remarques :

- Moins de calculs et moins d'accès disque pour la jointure :

```
SELECT * FROM departement NATURAL JOIN etudiant
WHERE departement.id = 1;
```

- Les performances des accès séquentiels sont fortement dégradées :

```
SELECT * FROM departement;
```