université PARIS-SACLAY

Bases de Données & Optimisation : Les Index

Thomas Gerald

October 2, 2026

Laboratoire Interdisciplinaire des Sciences du Numérique – LISN, CNRS
thomas.gerald@lisn.upsaclay.fr

Les index : introduction

Rule-based query optimization, 507
Rules in Datalog, 819
Running information for aggregation, 470
Runs in sorting, 423
R* trees, 985
SABRE, 6
Safe queries, 125
  in Datalog, 826
Safety, 826
Sampling
  real systems, 485
Savepoints, 535
Scalability, 890
Scale-up, 728
Scan, 744
Schedule, 523
  avoid cascading abort, 530
  conflict equivalence, 550
  conflict serializable, 550
  recoverable, 530, 571
  serial, 524
  serializable, 525, 529
  strict, 552
  view serializable, 553
Schema, 11, 59, 61
Schema decomposition, 609
  time for disks, 284, 308
Selection condition
  conjunct, 445
  conjunctive normal form, 445
  term, 444
Selection pushing, 409
Selections, 744
  definition, 103
Selectivity
  of an access path, 399
Semantic data model, 10, 27
Semantic integration, 995
Semijoin, 747
Semijoin reduction, 747
Serninaive fixpoint evaluation, 836
Semistructured data, 946, 1001
Sequence data, 913
Sequence of itemsets, 902
Sequence set in a B+ tree, 345
Sequential flooding, 321, 472
Sequential patterns, 901
Serial schedule, 524
Serializability, 525, 529, 550, 553, 561
Serializability graph, 551
Serializable schedule, 529
Server-side processing, 254

Comment retrouver efficacement l'information ?

Dans un livre ?

Les index : introduction

Rule-based query optimization, 507
Rules in Datalog, 819
Running information for aggregation, 470
Runs in sorting, 423
R* trees, 985
SABRE, 6
Safe queries, 125
  in Datalog, 826
Safety, 826
Sampling
  real systems, 485
Savepoints, 535
Scalability, 890
Scale-up, 728
Scan, 744
Schedule, 523
  avoid cascading abort, 530
  conflict equivalence, 550
  conflict serializable, 550
  recoverable, 530, 571
  serial, 524
  serializable, 525, 529
  strict, 552
  view serializable, 553
Schema, 11, 59, 61
Schema decomposition, 609
  time for disks, 284, 308
Selection condition
  conjunct, 445
  conjunctive normal form, 445
  term, 444
Selection pushing, 409
Selections, 744
  definition, 103
Selectivity
  of an access path, 399
Semantic data model, 10, 27
Semantic integration, 995
Semijoin, 747
Semijoin reduction, 747
Serninaive fixpoint evaluation, 836
Semistructured data, 946, 1001
Sequence data, 913
Sequence of itemsets, 902
Sequence set in a B+ tree, 345
Sequential flooding, 321, 472
Sequential patterns, 901
Serial schedule, 524
Serializability, 525, 529, 550, 553, 561
Serializability graph, 551
Serializable schedule, 529
Server-side processing, 254

Comment retrouver efficacement l'information ?

Dans un livre ?

- Index par ordre alphabétique/thèmatique
- Chaque entrée de l'index pointe sur une page

Les index : introduction

Rule-based query optimization, 507
Rules in Datalog, 819
Running information for aggregation, 470
Runs in sorting, 423
R* trees, 985
SABRE, 6
Safe queries, 125
  in Datalog, 826
Safety, 826
Sampling
  real systems, 485
Savepoints, 535
Scalability, 890
Scale-up, 728
Scan, 744
Schedule, 523
  avoid cascading abort, 530
  conflict equivalence, 550
  conflict serializable, 550
  recoverable, 530, 571
  serial, 524
  serializable, 525, 529
  strict, 552
  view serializable, 553
Schema, 11, 59, 61
Schema decomposition, 609
  time for disks, 284, 308
Selection condition
  conjunct, 445
  conjunctive normal form, 445
  term, 444
Selection pushing, 409
Selections, 744
  definition, 103
Selectivity
  of an access path, 399
Semantic data model, 10, 27
Semantic integration, 995
Semijoin, 747
Semijoin reduction, 747
Serninaive fixpoint evaluation, 836
Semistructured data, 946, 1001
Sequence data, 913
Sequence of itemsets, 902
Sequence set in a B+ tree, 345
Sequential flooding, 321, 472
Sequential patterns, 901
Serial schedule, 524
Serializability, 525, 529, 550, 553, 561
Serializability graph, 551
Serializable schedule, 529
Server-side processing, 254

Comment retrouver efficacement l'information ?

Dans un livre ?

- Index par ordre alphabétique/thèmatique
- Chaque entrée de l'index pointe sur une page

Dans un SGBD

Les index : introduction

Rule-based query optimization, 507
Rules in Datalog, 819
Running information for aggregation, 470
Runs in sorting, 423
R* trees, 985
SABRE, 6
Safe queries, 125
  in Datalog, 826
Safety, 826
Sampling
  real systems, 485
Savepoints, 535
Scalability, 890
Scale-up, 728
Scan, 744
Schedule, 523
  avoid cascading abort, 530
  conflict equivalence, 550
  conflict serializable, 550
  recoverable, 530, 571
  serial, 524
  serializable, 525, 529
  strict, 552
  view serializable, 553
Schema, 11, 59, 61
Schema decomposition, 609
  time for disks, 284, 308
Selection condition
  conjunct, 445
  conjunctive normal form, 445
  term, 444
Selection pushing, 409
Selections, 744
  definition, 103
Selectivity
  of an access path, 399
Semantic data model, 10, 27
Semantic integration, 995
Semijoin, 747
Semijoin reduction, 747
Serninaive fixpoint evaluation, 836
Semistructured data, 946, 1001
Sequence data, 913
Sequence of itemsets, 902
Sequence set in a B+ tree, 345
Sequential flooding, 321, 472
Sequential patterns, 901
Serial schedule, 524
Serializability, 525, 529, 550, 553, 561
Serializability graph, 551
Serializable schedule, 529
Server-side processing, 254

Comment retrouver efficacement l'information ?

Dans un livre ?

- Index par ordre alphabétique/thèmatique
- Chaque entrée de l'index pointe sur une page

Dans un SGBD

- Index selon un certain ordre
- Chaque entrée de l'index pointe sur une page

Les index : Concrétemment ?

Dans le cours précédent nous avons déjà observé des index !!!

Les index : Concrétemment ?

Dans le cours précédent nous avons déjà observé des index !!!

Un index

→ Organise les données pour un accès rapide

Les index : Concrétemment ?

Dans le cours précédent nous avons déjà observé des index !!!

Un index

→ Organise les données pour un accès rapide

→ Pointe sur les données (pages)

Les index : Concrétemment ?

Dans le cours précédent nous avons déjà observé des index !!!

Un index

→ Organise les données pour un accès rapide

→ Pointe sur les données (pages)

→ Un fichier trié/haché est un index

Les index : Concrétemment ?

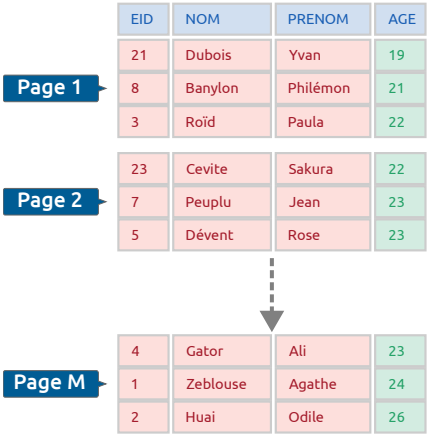
Dans le cours précédent nous avons déjà observé des index !!!

Un index

→ Organise les données pour un accès rapide

→ Pointe sur les données (pages)

→ **Un fichier trié/haché est un index**



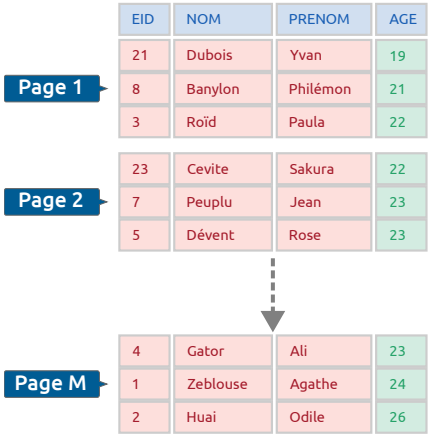
Les index : Concrétemment ?

Analogie d'un fichier trié

Un dictionnaire

→ Les pages d'un dictionnaire sont triés par ordre lexical

→ Chaque entrée contient les données (définition, traduction)



Les index : 3 types d'index

Un index, structure

- Un index est un fichier contenant des enregistrements appelés entrées de l'index
- Une entrée de l'index est un couple clef/reference

Une entrée de l'index ?

Une entrée est un couple (**entrée de l'index**, **référence**)

- Une **clef de l'index** est la valeur selon laquelle sont triés les enregistrements dans l'index
- La **référence** est un lien (record-id) vers l'enregistrement

Les index : différents types

On considère un fichier correspondant à une relation R, on souhaite indexer la relation selon le champs $R.a$, un enregistrement de l'index est

- Une valeur v de l'attribut $R.a$
- Un lien vers la donnée où l'attribut a est égale à v dans R

Comment encoder le lien vers la données ?

Les index : Type 1

Comment encoder le lien vers la données ?

Les index : Type 1

Comment encoder le lien vers la données ?

Un index de type 1

(v, e) où e est l'enregistrement

Les index : Type 1

Comment encoder le lien vers la données ?

Un index de type 1

(v, e) où e est l'enregistrement
$\rightarrow$ **le lien vers la données est la donnée elle-même**

Les index : Type 1

Comment encoder le lien vers la données ?

Un index de type 1

(v, e) où e est l'enregistrement
$\rightarrow$ **le lien vers la données est la donnée elle-même**

Par exemple ?

Comment encoder le lien vers la données ?

Un index de type 1

(v, e) où e est l'enregistrement
$\rightarrow$ **le lien vers la données est la donnée elle-même**

Par exemple ? $\rightarrow$ Fichier de données trié/haché ...

EID	NOM	PRENOM	AGE	
21	Dubois	Yvan	19	
8	Banylon	Philémon	21	Page 1
3	Roïd	Paula	22	
23	Cevite	Sakura	22	
7	Peuplu	Jean	23	Page 2
5	Dévent	Rose	23	
4	Gator	Ali	23	
1	Zeblouse	Agathe	24	Page 3
2	Huai	Odile	26	

Les index : Type 1

Comment encoder le lien vers la données ?

EID	NOM	PRENOM	AGE
21	Dubois	Yvan	19
8	Banylon	Philémon	21
3	Roïd	Paula	22

Page 1

23	Cevite	Sakura	22
7	Peuplu	Jean	23
5	Dévent	Rose	23

Page 2

4	Gator	Ali	23
1	Zeblouse	Agathe	24
2	Huai	Odile	26

Page 3

Un index de type 1 : problèmes

- **Un seul** index par relation ou . . .
- **Une copie** de *R* pour chaque index ?

Les index : Type 1

Comment encoder le lien vers la données ?

EID	NOM	PRENOM	AGE
21	Dubois	Yvan	19
8	Banylon	Philémon	21
3	Roïd	Paula	22

Page 1

23	Cevite	Sakura	22
7	Peuplu	Jean	23
5	Dévent	Rose	23

Page 2

4	Gator	Ali	23
1	Zeblouse	Agathe	24
2	Huai	Odile	26

Page 3

Un index de type 1 : problèmes

- **Un seul** index par relation ou . . .
- ~~**Une copie** de *R* pour chaque index ?~~

Les index : Type 2

Comment encoder le lien vers la données ?

Les index : Type 2

Comment encoder le lien vers la données ?

Un index de type 2

(v, rid) où *rid* est un pointeur sur l'enregistrement

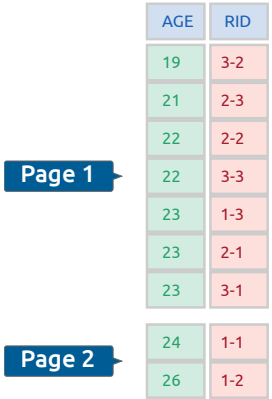
Comment encoder le lien vers la données ?

Un index de type 2

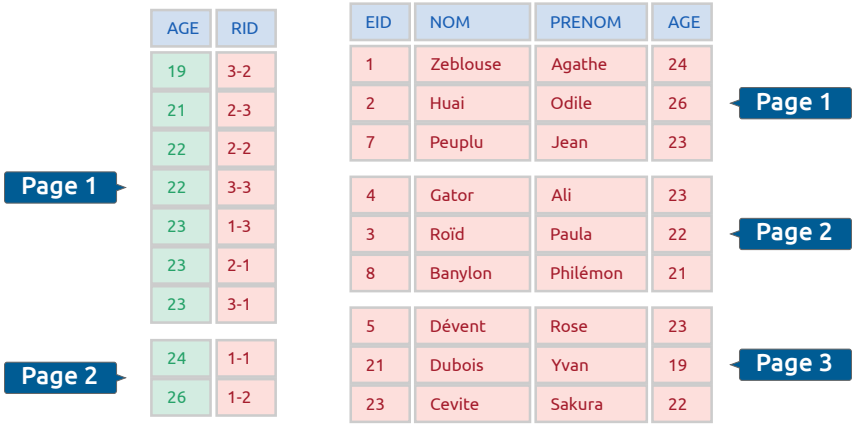
(v, rid) où rid est un pointeur sur l'enregistrement
$\rightarrow$ le lien vers la données est un pointeur

Avantages ?

- nombre pages index $<$ nombre pages de R
- Faciliter la mise à jour !!!



Comment encoder le lien vers la données ?



Les index : Type 3

Comment encoder le lien vers la données ?

Les index : Type 3

Comment encoder le lien vers la données ?

Un index de type 3

$(v, [rid_1, rid_2, \ldots, rid_n])$ où rid_i est un pointeur sur l'enregistrement

Comment encoder le lien vers la données ?

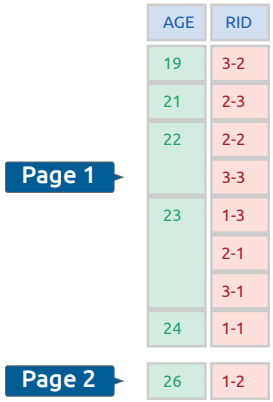
Un index de type 3

$(v, [rid_1, rid_2, \ldots, rid_n])$ où rid_i est un pointeur sur l'enregistrement

$\rightarrow$ **le lien vers la données est une liste de pointeurs**

Avantages ?

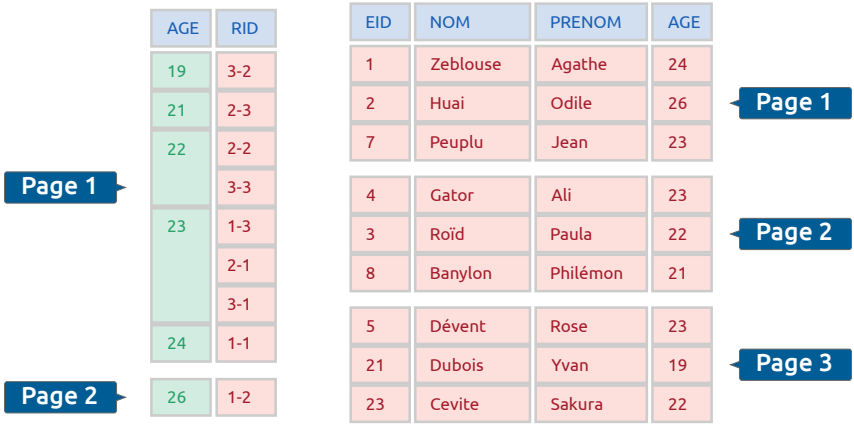
- Nombre pages index $\leq$ nombre pages de R
- Nombre pages index $\leq$ nombre pages index type 2



Les index : Type 3

Comment encoder le lien vers la données ?

Comment encoder le lien vers la données ?



Les index : Récapitulatif des types

- **Type 1 :** Fichier de Données
- **Type 2 :** Référence de l'entrée de l'index sur un enregistrement
- **Type 3 :** Référence de l'entrée de l'index sur une liste d'enregistrements

- Un index de type 1 par relations
- Plusieurs index de type 2 & 3 possibles

⚠ la nomenclature est specifique au cours (type 1, 2 et 3)

Les index : Groupant versus non-groupant

Index de type 2 (non groupant)

L'index et fichier triés différemment

- Recherche par égalité/intervalle

→ Plus de pages que nécessaire pour les enregistrements

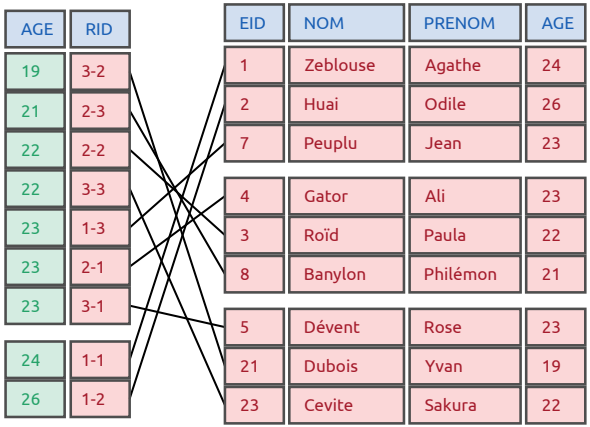
Les index : Groupant versus non-groupant

Index de type 2 (non groupant)

L'index et fichier triés différemment

- Recherche par égalité/intervalle

→ Plus de pages que nécessaire pour les enregistrements



Les index : Groupant versus non-groupant

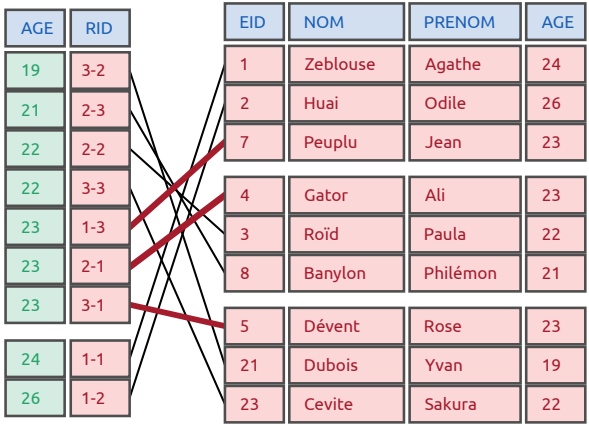
Index de type 2 (non groupant)

L'index et fichier triés différemment

- Recherche par égalité/intervalle

→ Plus de pages que nécessaire pour les enregistrements

Recherche étudiants ayant 23 ans ?



Les index : Groupant versus non-groupant

Index de type 2 (non groupant)

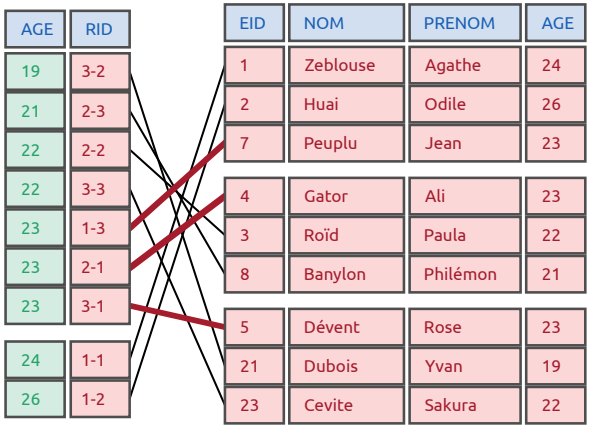
L'index et fichier triés différemment

- Recherche par égalité/intervalle

→ Plus de pages que nécessaire pour les enregistrements

Recherche étudiants ayant 23 ans ?

⚠ 3 pages pour 3 enregistrements...



Les index : Groupant versus non-groupant

Index de type 2 (Groupant)

Index et fichier triés dans le même ordre

- Recherche par égalité/intervalle

→ ≈ nombre de pages nécessaires pour les enregistrements

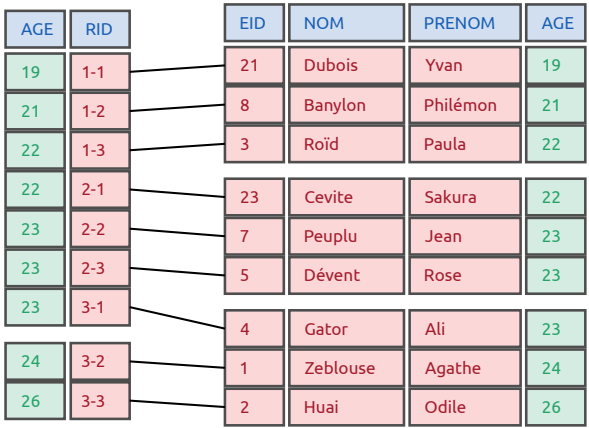
Les index : Groupant versus non-groupant

Index de type 2 (Groupant)

Index et fichier triés dans le même ordre

- Recherche par égalité/intervalle

→ ≈ nombre de pages nécessaires pour les enregistrements



Les index : Groupant versus non-groupant

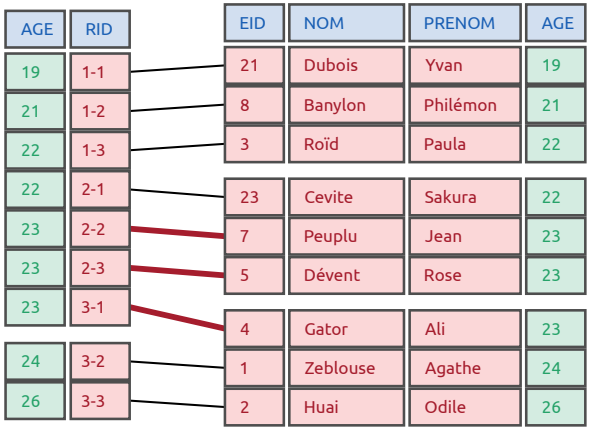
Index de type 2 (Groupant)

Index et fichier triés dans le même ordre

- Recherche par égalité/intervalle

→ ≈ nombre de pages nécessaires pour les enregistrements

Recherche étudiants ayant 23 ans ?



Les index : Groupant versus non-groupant

Index de type 2 (Groupant)

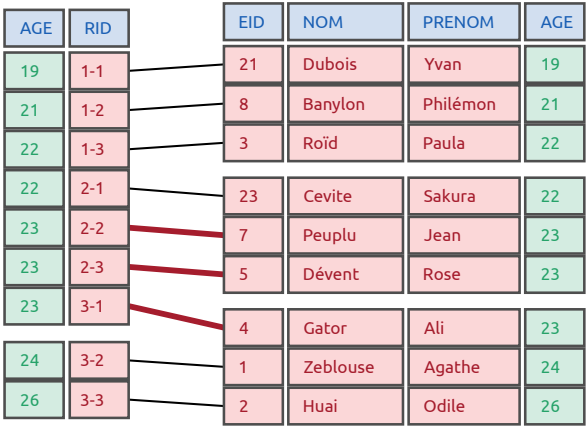
Index et fichier triés dans le même ordre

- Recherche par égalité/intervalle

→ ≈ nombre de pages nécessaires pour les enregistrements

Recherche étudiants ayant 23 ans ?

✅ 2 pages pour 3 enregistrements...



Les index : Groupant versus non-groupant

Index de type 2 (Groupant)

Index et fichier triés dans le même ordre

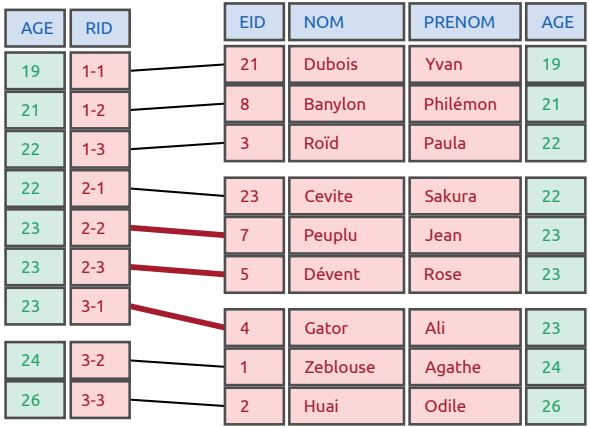
- Recherche par égalité/intervalle

→ ≈ nombre de pages nécessaires pour les enregistrements

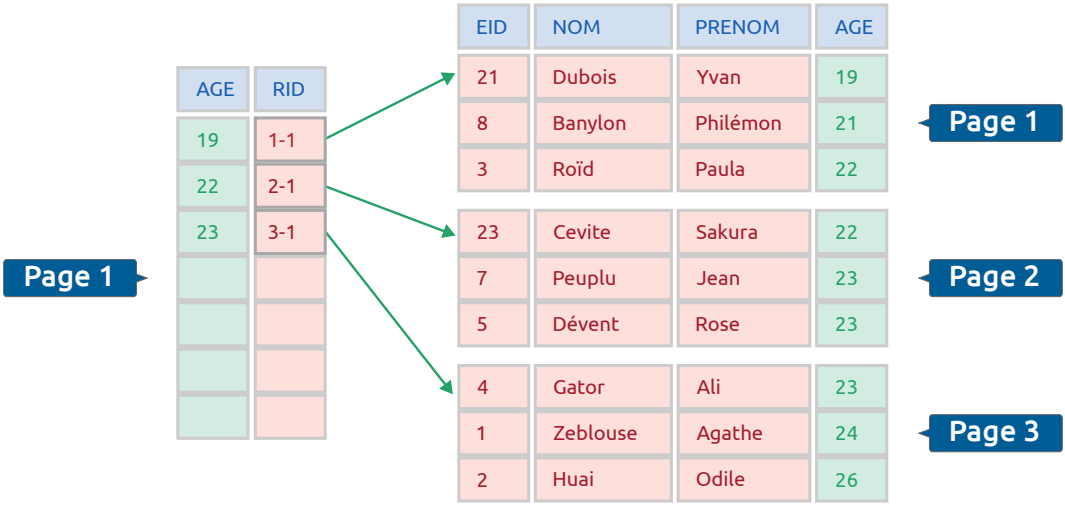
Recherche étudiants ayant 23 ans ?

✅ 2 pages pour 3 enregistrements...

⚠ Un index (max) groupant par relation



Les index : Dense versus Non-dense (clairssemé)



Les index : Dense versus Non-dense (clairssemé)

Quel est la taille du fichier d'un index dense sur R avec la clef $R.a$? D'un index creux ?

Example

- R comporte 10.000 pages de $8Ko$
- 100 enregistrements par page
- $R.a$ est stocké sur $4octets$ (32 bits)
- Un rid est stocké sur $8octets$

Index dense

On considère (pour simplifier) que toutes les pages sont remplies

- $10.000 \times 100 = 10^6$ enregistrements
  $\rightarrow 10^6$ entrées de l'index
- $(8 + 4) \times 10^6$ donc 12Mo
- $\frac{12000}{8} = 1500$

Les index : Dense versus Non-dense (clairssemé)

Quel est la taille du fichier d'un index dense sur R avec la clef $R.a$? D'un index creux ?

Example

- R comporte 10.000 pages de $8Ko$
- 100 enregistrements par page
- $R.a$ est stocké sur $4octets$ (32 bits)
- Un rid est stocké sur $8octets$

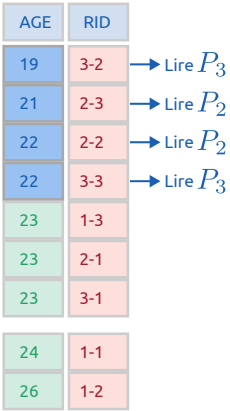
Index creux

On considère (pour simplifier) que toutes les pages sont remplies

- 10.000 pages
  $\rightarrow 10^4$ entrées de l'index
- $(8 + 4) \times 10.000$ donc 120Ko
- $\frac{120}{8} = 15$ pages

Les index : Un problème d'ouverture de page

On se place dans le cas d'une recherche des étudiants ayants moins de 23 ans



Recherche des valeurs

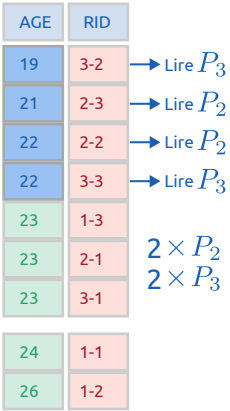
- On trouve dans l'index 4 enregistrements satisfaisant la condition
- pour chaque enregistrement on ouvre la page associée

→ ⚠ Ouvrir plus de pages que nécessaire !!

[0] NB : Notons que normalement le cache permet de ne pas recharger les pages. Ce problème apparaît dès lors que l'on considère de relations de taille importante où la taille du cache n'est pas suffisante.

Les index : Un problème d'ouverture de page

On se place dans le cas d'une recherche des étudiants ayants moins de 23 ans



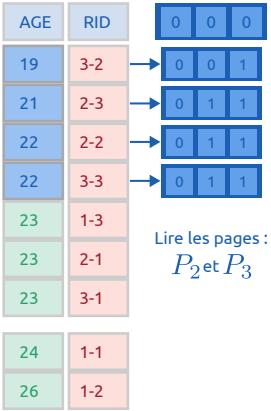
Recherche des valeurs

- On trouve dans l'index, 4 enregistrements satisfaisant la condition
- pour chaque enregistrement on ouvre la page associée

→ ⚠ Ouvrir plus de pages que nécessaire !!

[0] NB : Notons que normalement le cache permet de ne pas recharger les pages. Ce problème apparaît dès lors que l'on considère de relations de taille importante où la taille du cache n'est pas suffisante.

On se place dans le cas d'une recherche des étudiants ayants moins de 23 ans



BitMap

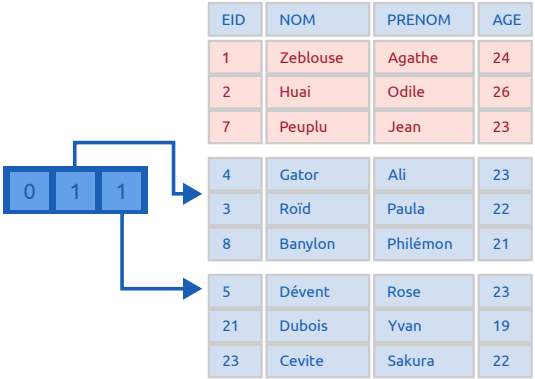
On peut découper en deux temps notre recherche :

- Retrouver toutes les pages contenant au moins un enregistrement satisfaisant la condition - **bitmap index scan**
- Ouvrir les pages présentes dans le bitmap et rechercher dans les pages - **bitmap heap scan**

→ On peut utiliser un tableau de bits (bitmap) ayant la taille du nombre de pages

Les index : Bitmap index scan

On se place dans le cas d'une recherche des étudiants ayants moins de 23 ans



Bitmap heap scan

- Ouvrir les pages pointées par le Bitmap
- Retrouver les enregistrements validant la condition

$\rightarrow$ 2 pages lus au lieu de 4

Les index : clefs composites

Page 1

AGE	NOM	RID
19	Dubois	3-2
21	Banylon	2-3
22	Cevite	3-3
22	Roid	2-2
23	Dévent	1-3
23	Gator	2-1
23	Peuplu	3-1

Page 2

AGE	NOM	RID
24	Zeblouse	1-1
26	Huai	1-2

Page 1

EID	NOM	PRENOM	AGE
1	Zeblouse	Agathe	24
2	Huai	Odile	26
7	Peuplu	Jean	23

Page 2

EID	NOM	PRENOM	AGE
4	Gator	Ali	23
3	Roïd	Paula	22
8	Banylon	Philémon	21

Page 3

EID	NOM	PRENOM	AGE
5	Dévent	Rose	23
21	Dubois	Yvan	19
23	Cevite	Sakura	22

Comme pour les fichiers

- Des index triés (recherche rapide, mais ajout/suppression coûteuse)
- Des index hachés (recherche sur égalité rapide, mais recherche sur intervalle long)

✅ On peut créer plusieurs index pour une relation !!
⚠️ Mais...Si I le nombre de pages dans l'index

	Scan	Rech (égalité)	Rech (intervalle)	Insert	Suppr
Trié (Sorted Index)	I	$log(I)$	$log(I)$	I	I
Haché (Hashed Index)	I	1	I	1	1

Bonne performance en recherche et en insertion/suppression ? → ✅ B-TREE

Les index : Conclusion

Introduction aux types d'index

- Les 3 types d'index
  - Type 1 : le fichier d'enregistrement est un index
  - Type 2 : Une clef et un **record-id**
  - Type 3 : Une clef et une liste de **record-id**
- Index groupant/non groupant (index trié/organisé ≈ fichier d'enregistrement)
- Index dense/non-dense (index contenant une clef par enregistrement vs par page)
- Limiter l'ouverture multiple d'une page avec les BitMaps
- Des index composites (plusieurs clefs)
- Quelles structures sont possibles pour les index

Prochain cours : Les index **B+**