

Bases de Données & Optimisation : Les Index

Thomas Gerald

September 23, 2025

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

- 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
- Seminaive 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 ?

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
Semi-naïve 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/thématique
- 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
Semi-naïve 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/thématique
- 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
Semi-naïve 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/thématique
- 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 : Concrètement ?

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

Les index : Concrètement ?

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

Un index

→ Organise les données pour un accès rapide

Les index : Concrètement ?

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 : Concrètement ?

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 : Concrètement ?

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

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

Les index : Concrètement ?

Analogie d'un fichier trié

Un dictionnaire

- Les pages d'un dictionnaire sont triées par ordre lexical
- Chaque entrée contient les données (définition, traduction)

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

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

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 ?

Comment encoder le lien vers la données ?

Comment encoder le lien vers la données ?

Un index de type 1

(v, e) où e est l'enregistrement

Comment encoder le lien vers la données ?

Un index de type 1

(v, e) où e est l'enregistrement

→ le lien vers la données est la donnée elle-même

Comment encoder le lien vers la données ?

Un index de type 1

(v, e) où e est l'enregistrement

→ 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

→ le lien vers la données est la donnée elle-même

Par exemple ? → Fichier de données trié/haché ...

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

Page 1

Page 2

Page 3

Comment encoder le lien vers la données ?

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

Page 1

Page 2

Page 3

Un index de type 1 : problèmes

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

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 ?~~

Comment encoder le lien vers la données ?

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

→ le lien vers la données est un pointeur

Avantages ?

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

	AGE	RID
	19	3-2
	21	2-3
	22	2-2
Page 1	22	3-3
	23	1-3
	23	2-1
	23	3-1
	24	1-1
Page 2	26	1-2

Comment encoder le lien vers la données ?

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

Comment encoder le lien vers la données ?

Comment encoder le lien vers la données ?

Un index de type 3

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

Comment encoder le lien vers la données ?

Un index de type 3

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

→ 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

AGE	RID
19	3-2
21	2-3
22	2-2
23	3-3
	1-3
	2-1
24	3-1
	1-1
26	1-2

Page 1

Page 2

Comment encoder le lien vers la données ?

Comment encoder le lien vers la données ?

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

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 spécifique au cours (type 1, 2 et 3)

Index de type 2 (non groupant)

L'index et fichier triés différemment

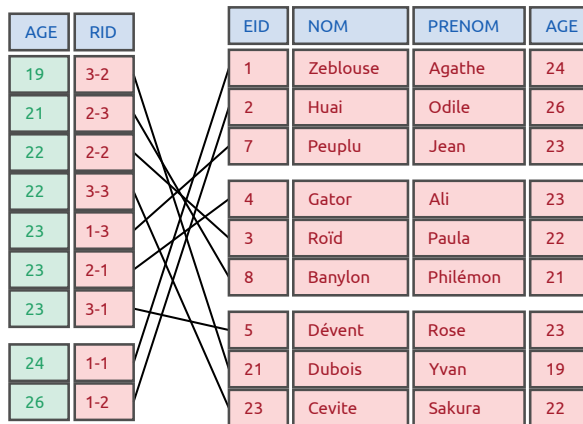
- Recherche par égalité/intervalle
- Plus de pages que nécessaire pour les enregistrements

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



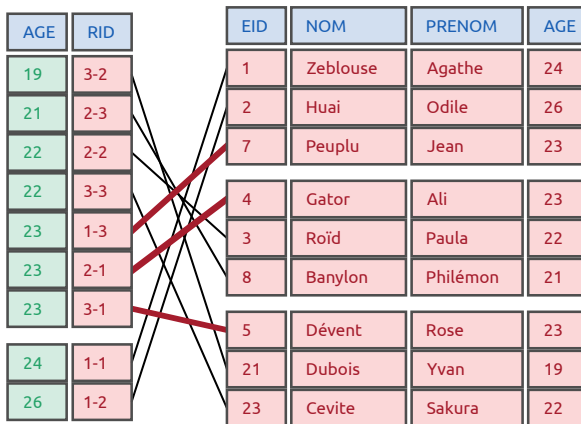
Les index : Groupant versus non-groupant

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

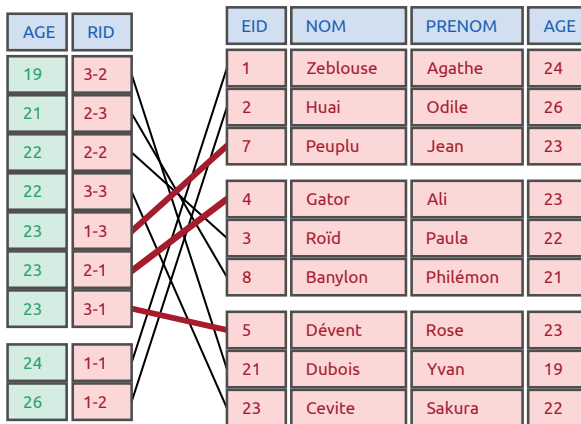
Index de type 2 (non groupant)

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...



Index de type 2 (Groupant)

Index et fichier triés dans le même ordre

- Recherche par égalité/intervalle
- \approx 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
- \approx nombre de pages nécessaires pour les enregistrements

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

Les index : Groupant versus non-groupant

Index de type 2 (Groupant)

Index et fichier triés dans le même ordre

- Recherche par égalité/intervalle
→ \approx nombre de pages nécessaires pour les enregistrements

Recherche étudiants ayant 23 ans ?

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

Les index : Groupant versus non-groupant

Index de type 2 (Groupant)

Index et fichier triés dans le même ordre

- Recherche par égalité/intervalle
→ \approx nombre de pages nécessaires pour les enregistrements

Recherche étudiants ayant 23 ans ?

✔ 2 pages pour 3 enregistrements...

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

Les index : Groupant versus non-groupant

Index de type 2 (Groupant)

Index et fichier triés dans le même ordre

- Recherche par égalité/intervalle
→ \approx nombre de pages nécessaires pour les enregistrements

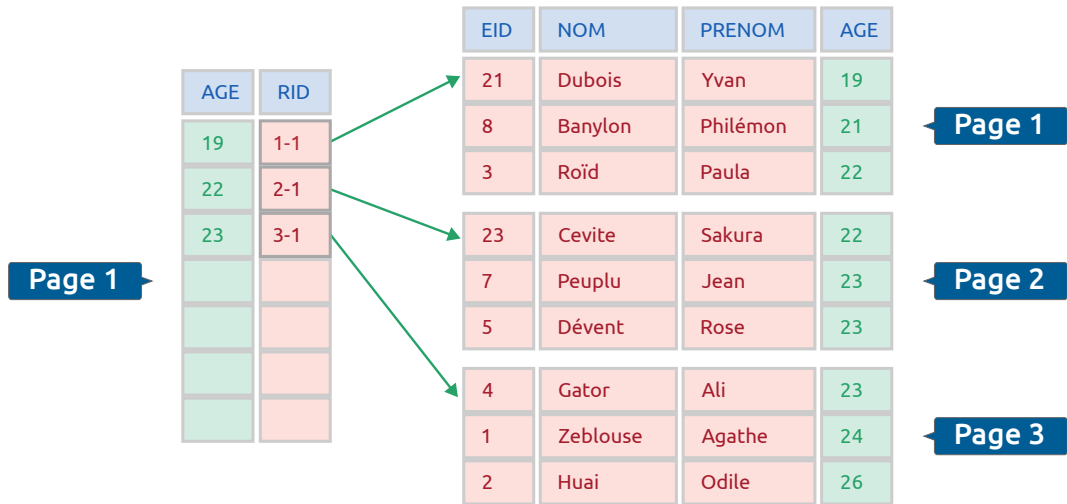
Recherche étudiants ayant 23 ans ?

✔ 2 pages pour 3 enregistrements...

⚠ Un index (max) groupant par relation

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

Les index : Dense versus Non-dense (clairsemé)



Les index : Dense versus Non-dense (clairsemé)

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

Exemple

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

Les index : Dense versus Non-dense (clairsemé)

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

Exemple

- 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
→ 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 ayant moins de 23 ans

AGE	RID
19	3-2
21	2-3
22	2-2
22	3-3
23	1-3
23	2-1
23	3-1
24	1-1
26	1-2

→ Lire P_3

→ Lire P_2

→ Lire P_2

→ Lire P_3

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 !!**

⁰NB : 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

AGE	RID	
19	3-2	→ Lire P_3
21	2-3	→ Lire P_2
22	2-2	→ Lire P_2
22	3-3	→ Lire P_3
23	1-3	$2 \times P_2$ $2 \times P_3$
23	2-1	
23	3-1	
24	1-1	
26	1-2	

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 !!**

⁰NB : 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 : Bitmap index scan

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

AGE	RID	
		0 0 0
19	3-2	0 0 1
21	2-3	0 1 1
22	2-2	0 1 1
22	3-3	0 1 1
23	1-3	
23	2-1	
23	3-1	
24	1-1	
26	1-2	

Lire les pages :
 P_2 et P_3

BitMap

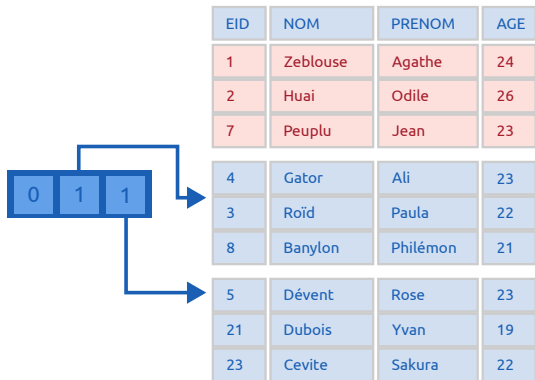
On peut découper en deux temps notre recherche :

- 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 ayant moins de 23 ans



Bitmap heap scan

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

→ 2 pages lus au lieu de 4

Les index : clefs composites

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

	EID	NOM	PRENOM	AGE	
Page 1	1	Zeblouse	Agathe	24	
	2	Huai	Odile	26	
	7	Peuplu	Jean	23	
Page 2	4	Gator	Ali	23	
	3	Roïd	Paula	22	
	8	Banylon	Philémon	21	
Page 3	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

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 \approx 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+