Le M/09/2012

SGBD

PLAN

I. INTRODUCTION

II. STRUCTURES DE STOCKAGES SUR DISQUE
   A. ORGANISATION DES FICHIERS.
   B. ORACLE

III. ALGORITHMES DE L'ALGEBRE RELATIONNELLE

IV. OPTIMISATION DES REQUÊTES.

V. CONCURRENCE

VI. JCDBC.

ZD

II. STRUCTURE DE STOCKAGE SUR DISQUE

≠ ⇒ Relation ou table ou tuple, C'est au niveau conceptuel.
⇒ Fichier : Au niveau physique.

Une relation peut être stocké sur plusieurs fichiers et plusieurs relations peuvent être stocké sur un seul fichier.

Pour minimiser le temps de déplacement de tête de lecture (Car le coût est cher), l'idéal serait de regrouper les tables ou relations qu'on utilise souvent ensemble sous un même fichier.
Exemple

La table client et Commande dans un seul fichier pour la rédaction d'une facture.

A. Les différentes organisations de fichier

1. L'organisation séquentielle

Définition

Les enregistrements sont placés les uns à la suite des autres. Mais rien n'interdit de placer l'enregistrement là où il y a de la place (Pour).
→ C'est l'organisation utilisé par Oracle par défaut
→ Elle est moins couteuse.
→ Elle répond à plusieurs besoins.

Le 18/09/2012.

- Caractéristiques générales des index :

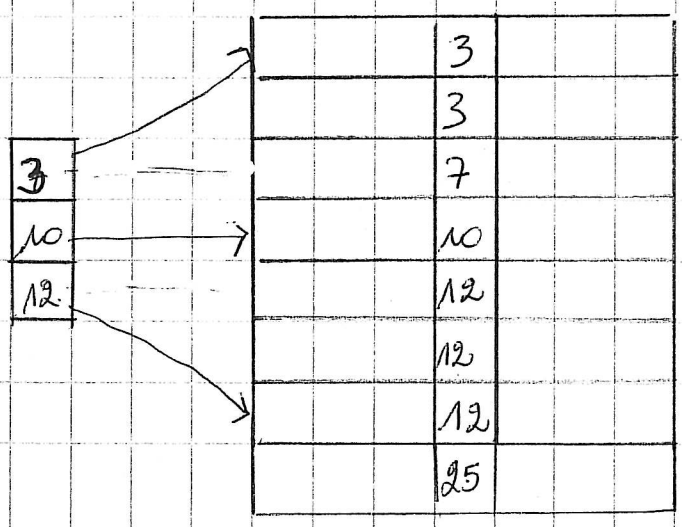
1. Il est toujours trié.
2. Parfois le fichier est lui même trié selon l'index.
   ↳ dans ce cas on parle d'un index plaçant.
   Sinon l'index est non plaçant ↳ Primaire
   ↳ Secondaire.

⇒ la notion d'index est différente de la notion de clé.
On peut avoir créer un index primaire sur un autre attribut (la ville par exemple), autre que la clé primaire.

3. index hiérarchisé : sous forme d'un arbre, pour permettre une recherche plus facile à partir de la racine.

4. Index dense ou non dense.
   ↳ Un index où toutes les valeurs des rubriques indexées sont dans l'index.
⇒ non dense : Seules 1% de l'ensemble des valeurs des rubriques indexées sont dans l'index.
   ↳ Ne marche que dans le cas où les index sont Primaires.

Ex :



On peut avoir plusieurs index sur un fichier.

Exp :

- Sur la table Client, On peut créer un index sur la clé, sur la ville, sur le nom.
- F(a, b, c, d)

index sur a — ac
b — ad
c — bc
d — bdc
ab — ca, ---
b ≠ a

3. Les index uniques ou non.

Plusieurs clients peuvent avoir la même valeur d'index sur ville : "DIJON".

2.1. Organisation ISAM

↳ Index Sequential Access Method

- Index Primaire ou plaçant.
- Index trié
- Index est non dense.
- Index hiérarchisé.

⇒ L'organisation ISAM : implique une réorganisation périodique des fichiers.

⇒ Dans ISAM, les indexes sont figés, ils évoluent pas.

→ Avec l'organisation séquentielle, on peut faire du balayage séquentiel.

Ce qu'on attend de l'organisation séquentielle.

- Mettre les enregistrements et les récupérer un par un.

⇒ On attends 2 opérations.

a) Ecr. seq ⇒ Ecriture séquentielle.

Elle porte sur un fichier, elle écrit un enregistrement et nous donne en retour un fichier modifié ou un code d'erreur.

Ecr. seq (fichier, enregistrement) → Fichier, C. err

b) lec. seq ⇒ La lecture séquentielle

On lit dans un fichier, soit le 1er enregistrement soit le suivant (paramètre d'ordre) et on attends en sortie un enregistrement ou un code d'erreur.

Lec. seq (fichier, {1ere / suivant}) → enregistrement, C. err

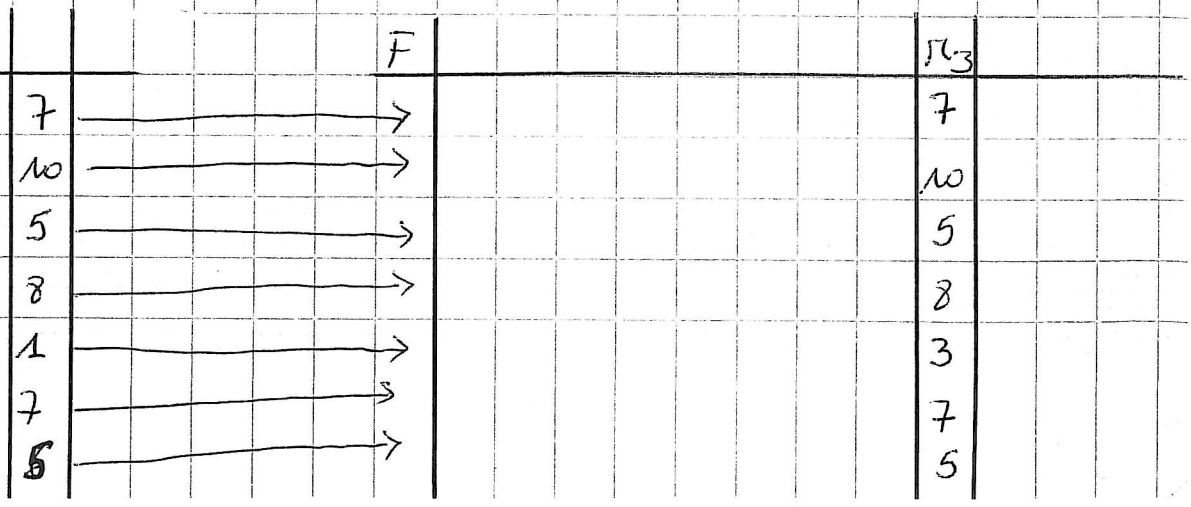
2. Organisation indexée

*) Définition

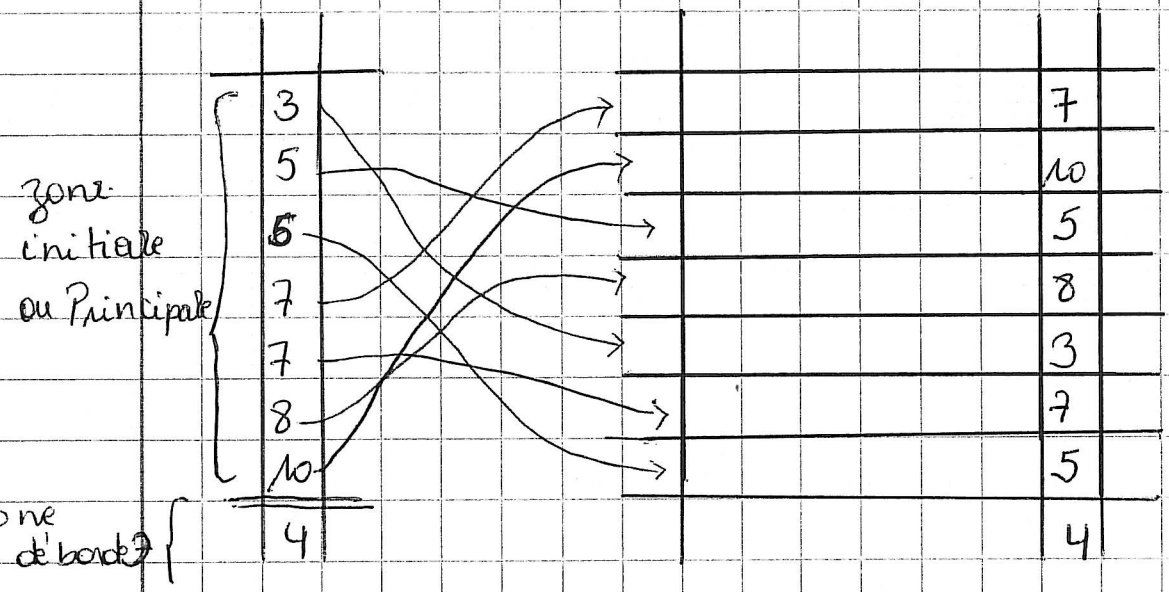
Dans l'organisation indexée, l'adressage des enregistrements se fait via un index.

index

Fichiers.



⇒ Fichier indexé = index + fichier.



Si l'index est trié on peut faire de la recherche dychotomique.

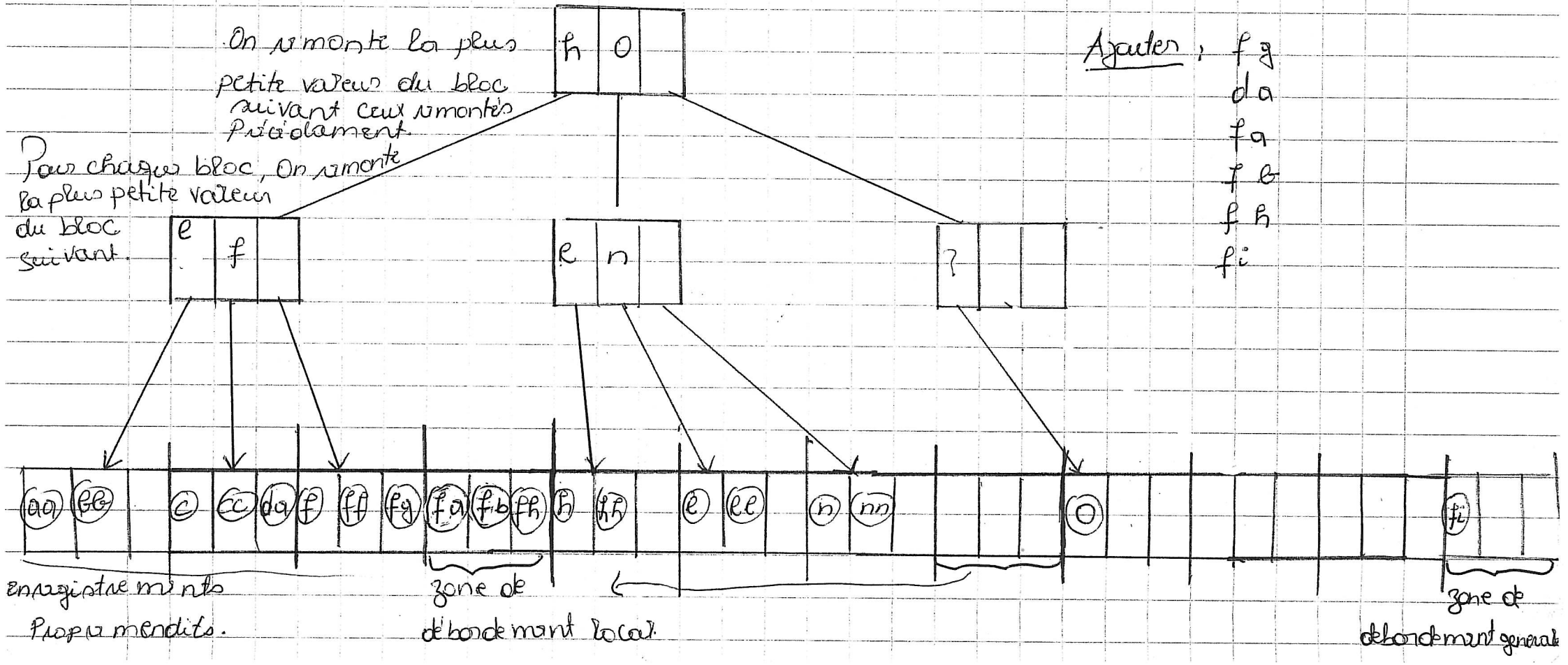
Si le fichier est trié peut minimiser la tête de lecture.

Bloc : Unité de transfert mémoire secondaire vers mémoire centrale

Taille d'un bloc : 2046 octets, 8192, 4096.

Taille de débordement locale après 3 blocs.

Taille "zone de débordement générale" = taille du fichier - taille effectivement occupée par les données à la création du fichier.



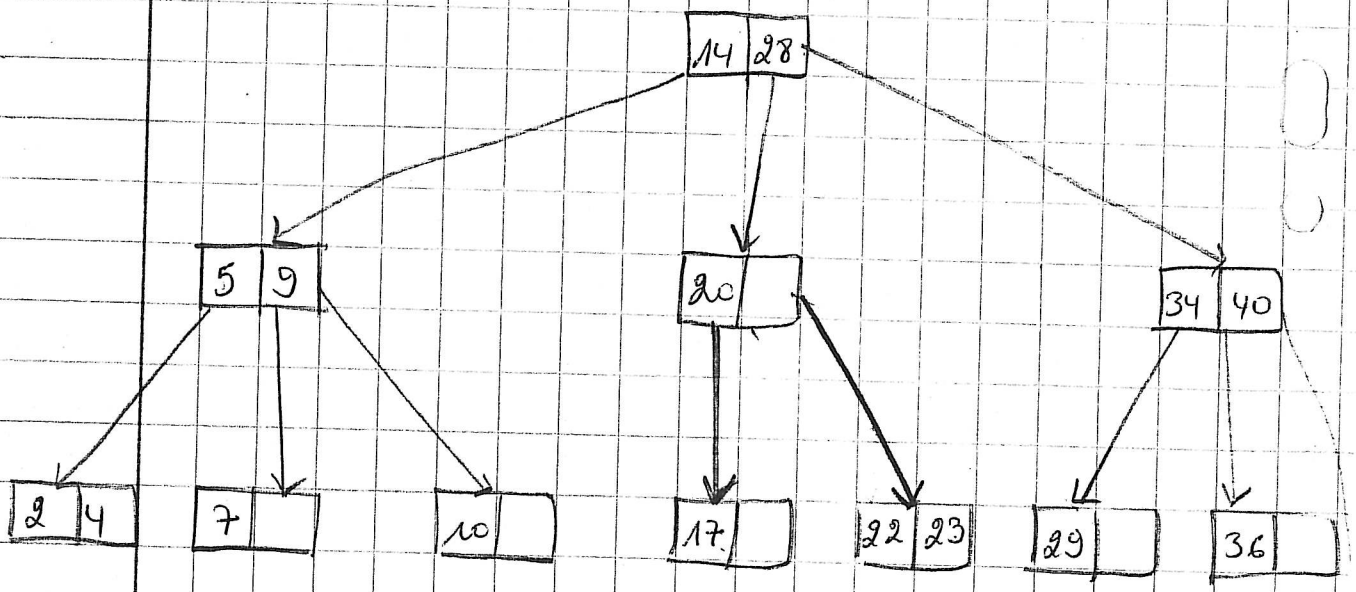
2.2. Notion de B-Tree

Balanced_Tree = arbre équilibré.

Définition

Un B-TREE d'ordre n est un arbre où :

- Toutes les branches sont strictement de même longueur.
- Chaque noeud (sauf racine) contient entre n et 2n valeurs. (Pour la racine, entre 1 et 2n valeurs).
- 1 noeud non terminal a 1 fils de plus que de valeurs. (contient autant de fils que de valeurs).
- Le $i^{ième}$ fils d'un noeud N contient des valeurs compris entre le $(i-1)^{ième}$ valeur et le $i^{ième}$ valeur N.

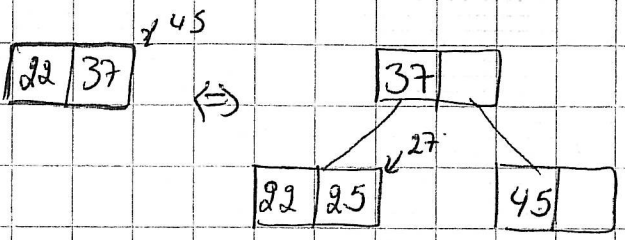


Ordre de l'arbre = 1.

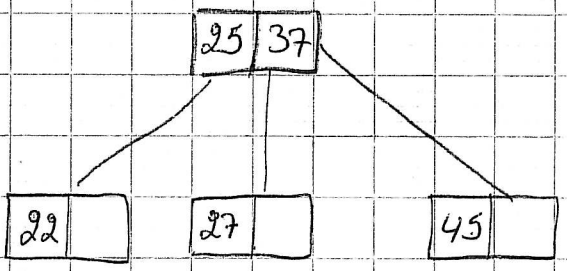
Chaque noeud doit être au moins à moitié rempli.

⇒ Petite recette de construction des B_Tree.

22, 37, 45, 25, 27



⇐



⇒ On peut utiliser les B_Tree, on peut stocker les index et les enregistrement.

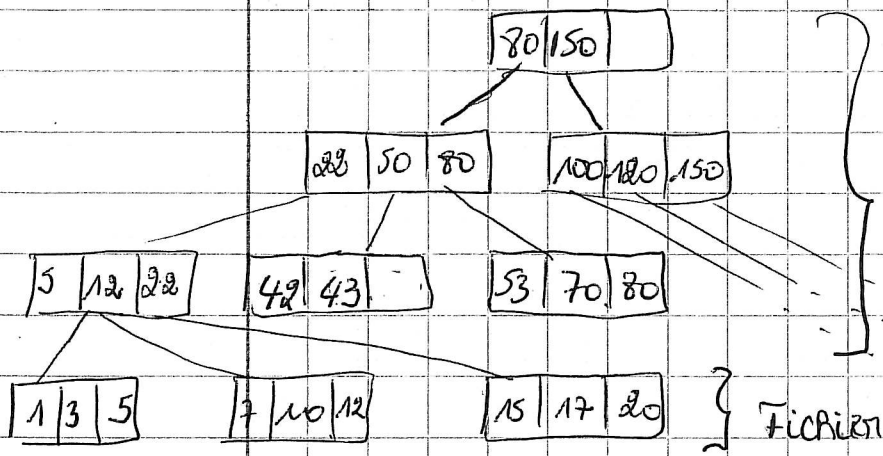
2.3. Organisation VSAM

↳ Virtual Storage Access Method

⇒ Les enregistrements plus l'index sont organisé dans un unique B_Tree.

⇒ Dans VSAM, c'est un peu comme ISAM sauf que l'index évolue, elle n'est pas figé. Si on supprime un enregistrement on ne met pas une étoile ds la case vide mais on réorganise comme l'exige la règle des B_TREE.

* L'index est éparçant.
* L'index est non dense.



index.

} Fichier Proprement dit.

→ Ajouter M.

Le 18/09/2012

TD1 : SGBD

Primitives sur les fichiers à Organisation indexée

* L'adressage des enregistrements se fait via un index.

Deux Primitives.

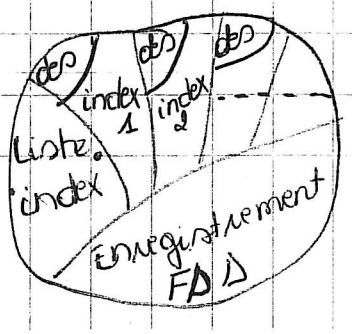
* Lec_ind (fichier, index, val_index {1er / suivant ?} ordre.
  → Enregistrement (en sortie) ou code erreur.
* Ecr_ind (fichier, enregistrement)
  → fichier modifier (en sortie), code erreur.

Ecriture des algo

1) Ecr_ind (fichier, enr) → fichier, C_err
   Ecr_enr (fichier.FPD, enr) → fichier.FPD, C_err, adr_enr.

   Pour tout index dans fichier.liste_index
   extr_val_index (index.description, enr) → val_ind
   MAJ_index (index, adr_enr, val_int) → index, C_err
   finPour

   → Récupérer la valeur d'index pour pouvoir faire la MAJ de tous les indexes.

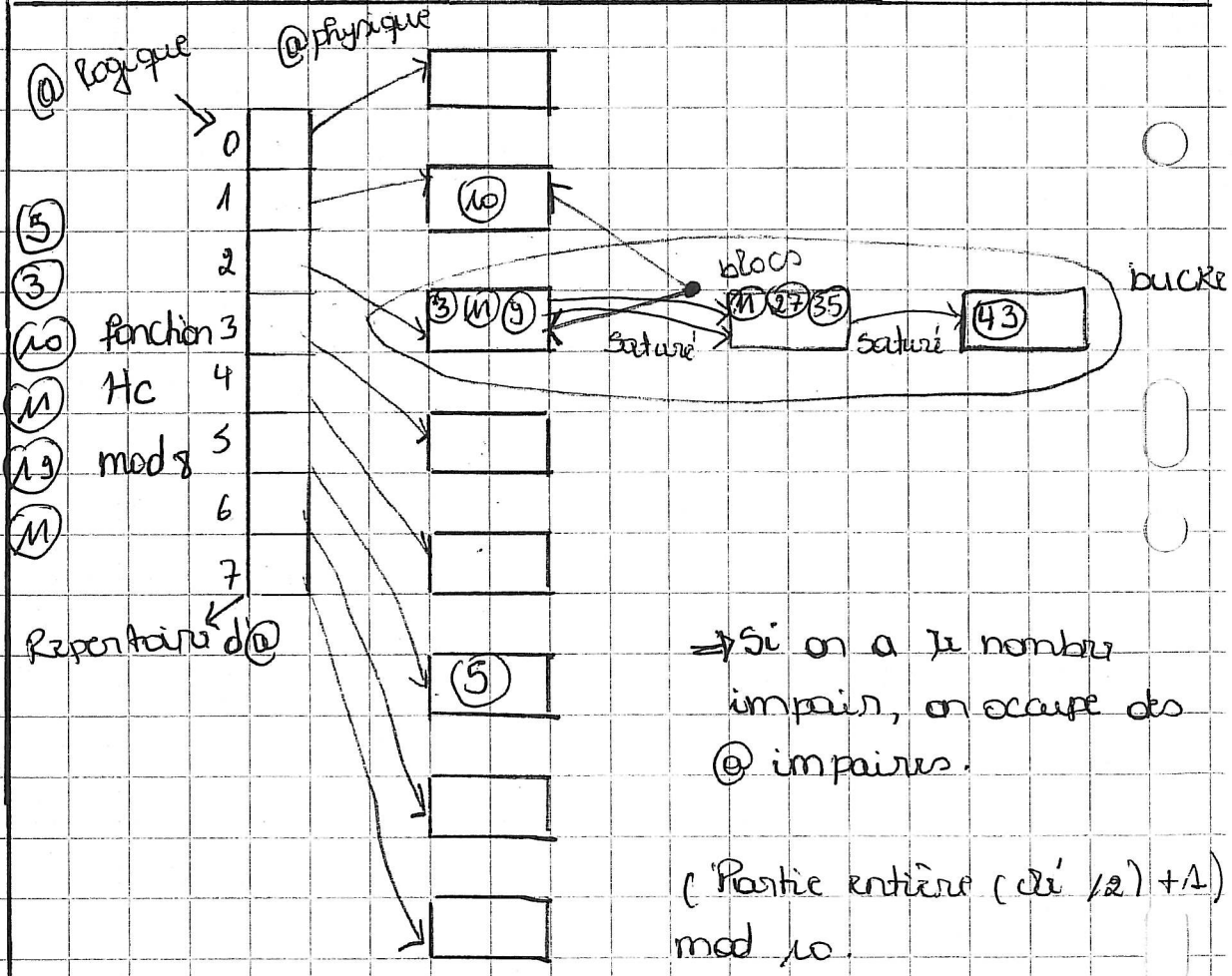


2) lec_ind (fichier, index, val_ind, ordre) → err, c_err
lec_ind (index, val_index, ordre) → err, c_err
lec_val_ind (indexe, val_index, ordre) → c_err, adr, err

Si c_err ≠ pas cette val_index
Alors lec_adr (fichier.FPS, adr_err) → err, c_err
finsi

→ On lit dans l'index, la valeur d'index et on récupère l'adresse et l'err.

lec_ind (FCli, 'Ville', 'Paris', '1er') → err, c_err
tantque c_err ≠ "pas d'enregistrement"
lec_ind (FCli, 'Ville', 'Paris', suivant) → err, c_err.



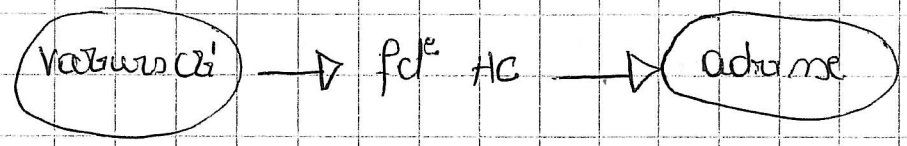
→ Si on a le nombre impair, on occupe des @ impaires.

(Partie entière (cli /2) +1) mod 10.

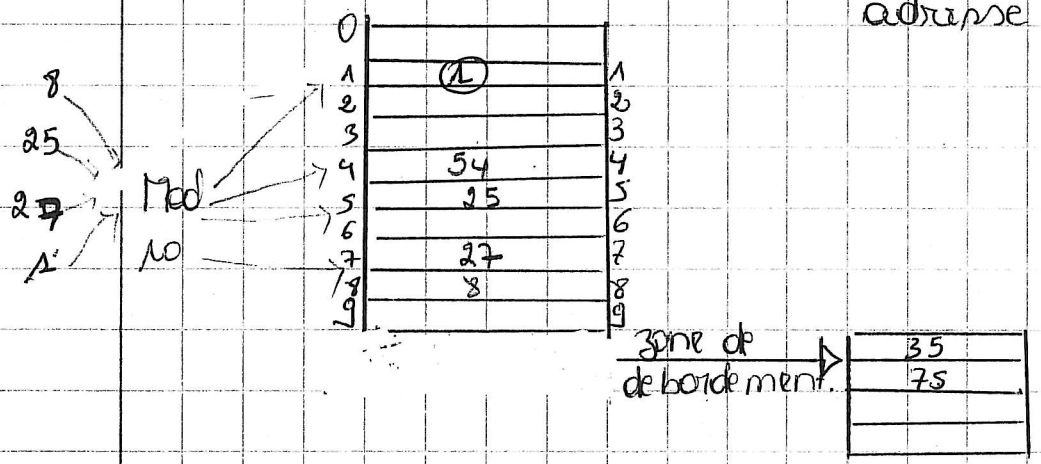
Ch3 SGBD

3. Organisation aléatoire : 3.1 Organisation aléatoire statique

Cette Organisation se définit par un mode d'adressage d'enregistrement se fait au moyen d'une fonction aléatoire.



On soumet la valeur de clé à 1 fct° hash code et on obtient une adresse.



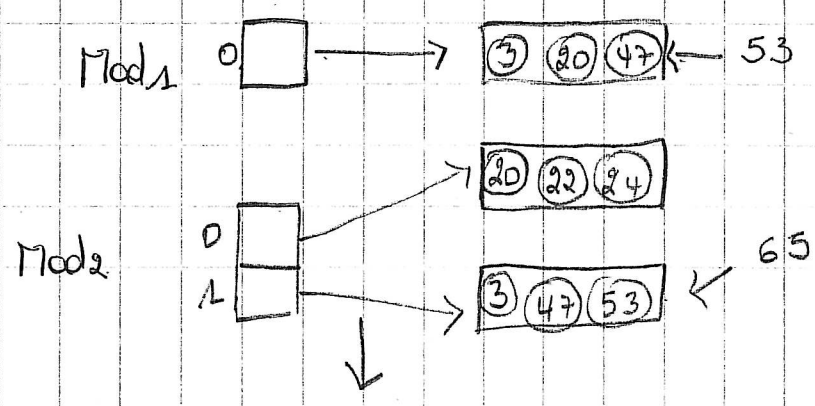
Collision : deux argument peuvent avoir la même valeur au emplacement.

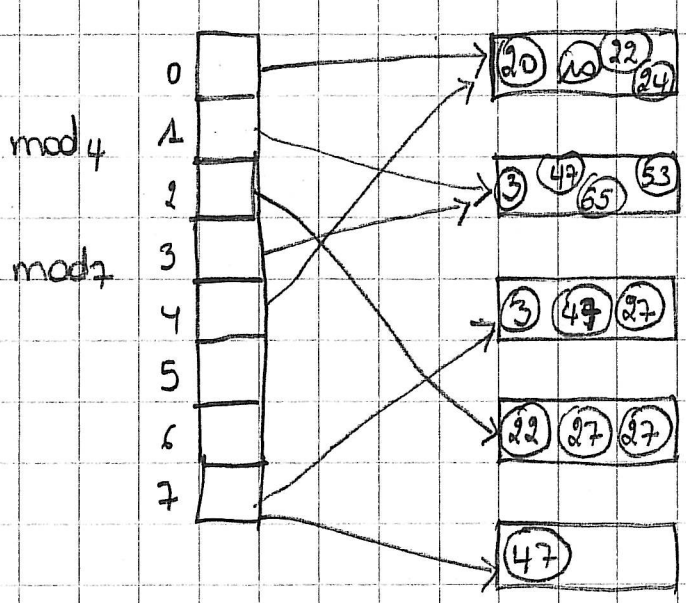
NB

On retrouve des blocs d'enregistrements sur le disque.

3.2. Organisation aléatoire dynamique,

Exple





(dictionnaire de donnée ORACLE).

B_ORACLE / 1- Tables et domaines.

CREATE TABLE < table >
(< Attribut > < domaine > CI, - - -)

Number (< Precision >, < decimals >)
⇒ Stocké sur la forme signe + mantisse + exposa
2 chiffres par octets (exp 123
[1] [23]

Varchar (< longueur >)
⇒ Stockage en longueur variable 1 octet par caractère.

Char (< longueur >)
⇒ Stockage en longueur fixe.

Date 7 octets.

⇒ Varchar prend 30% plus de place que Char

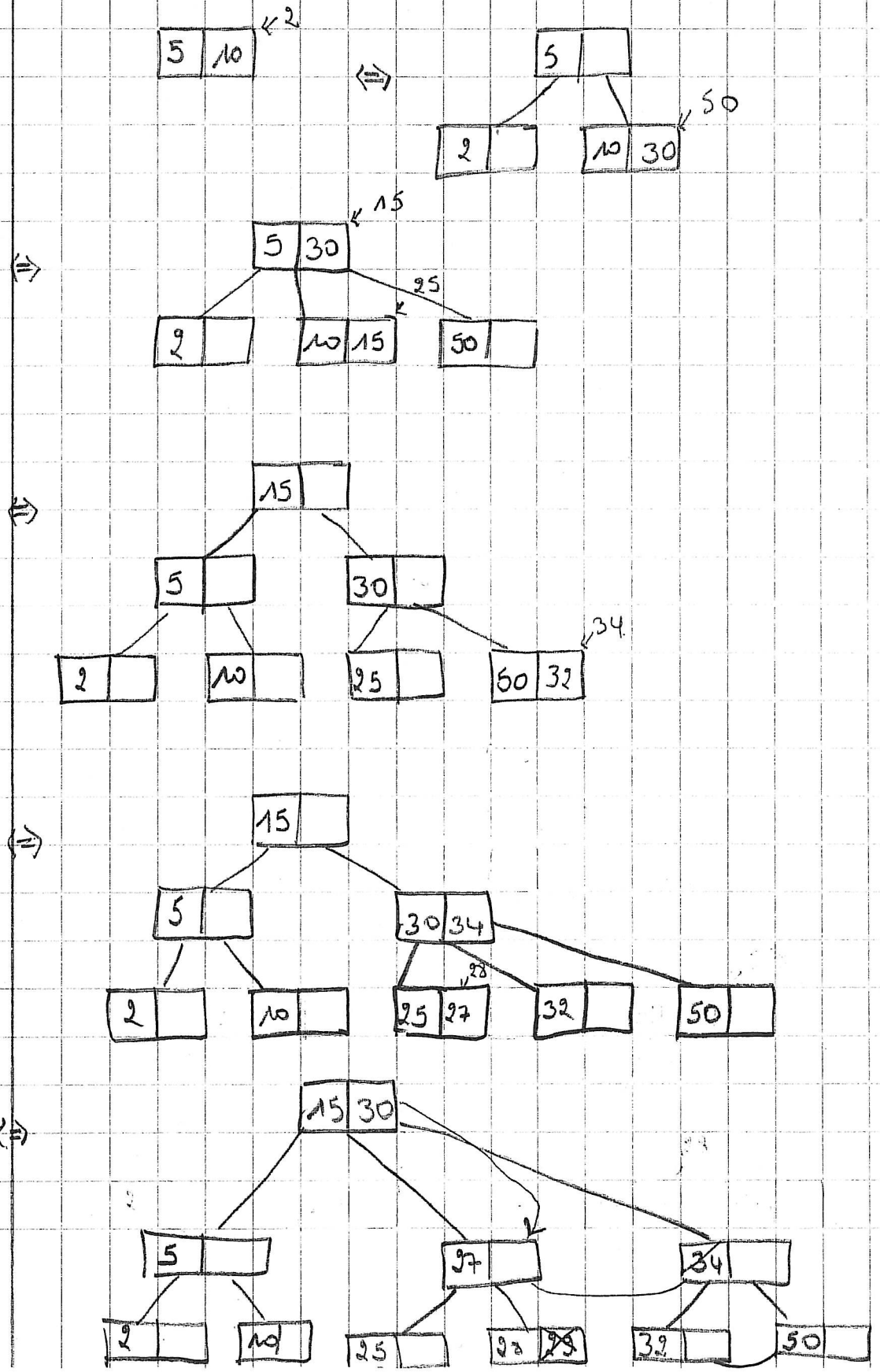
blob

Rowid : @ d'1 ligne d'1 table.

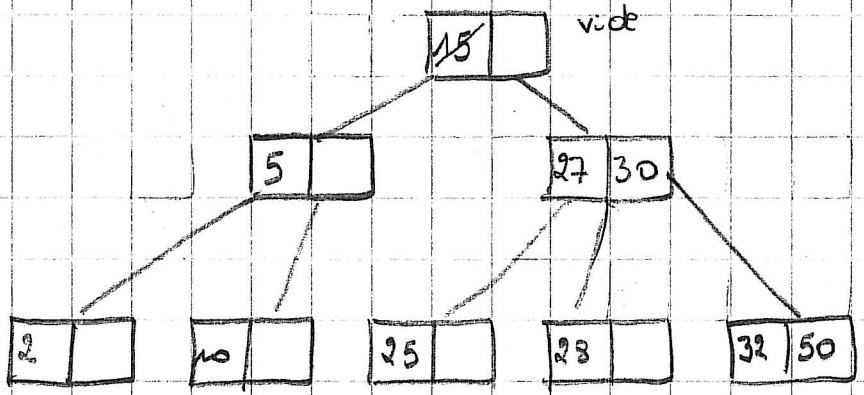
I. D2 B_TREE

Construire un B-tree d'ordre 1 qui contient les valeurs suivantes:

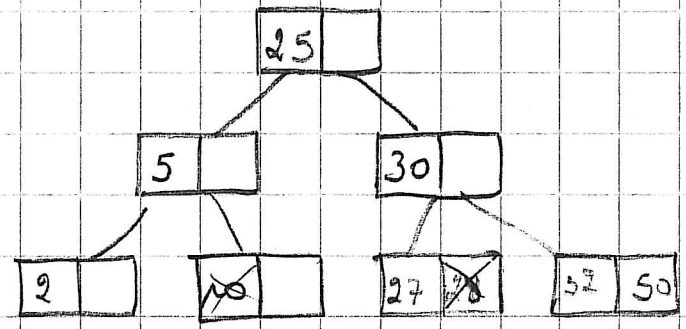
5 - 10 - 2 - 30 - 50 - 15 - 25 - 32 - 34 - 27 - 28 - 29



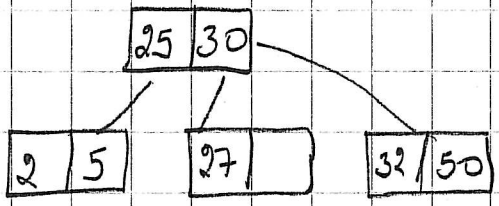
Supprimer 29 - 34 - 15 - 28 - 10



(=)

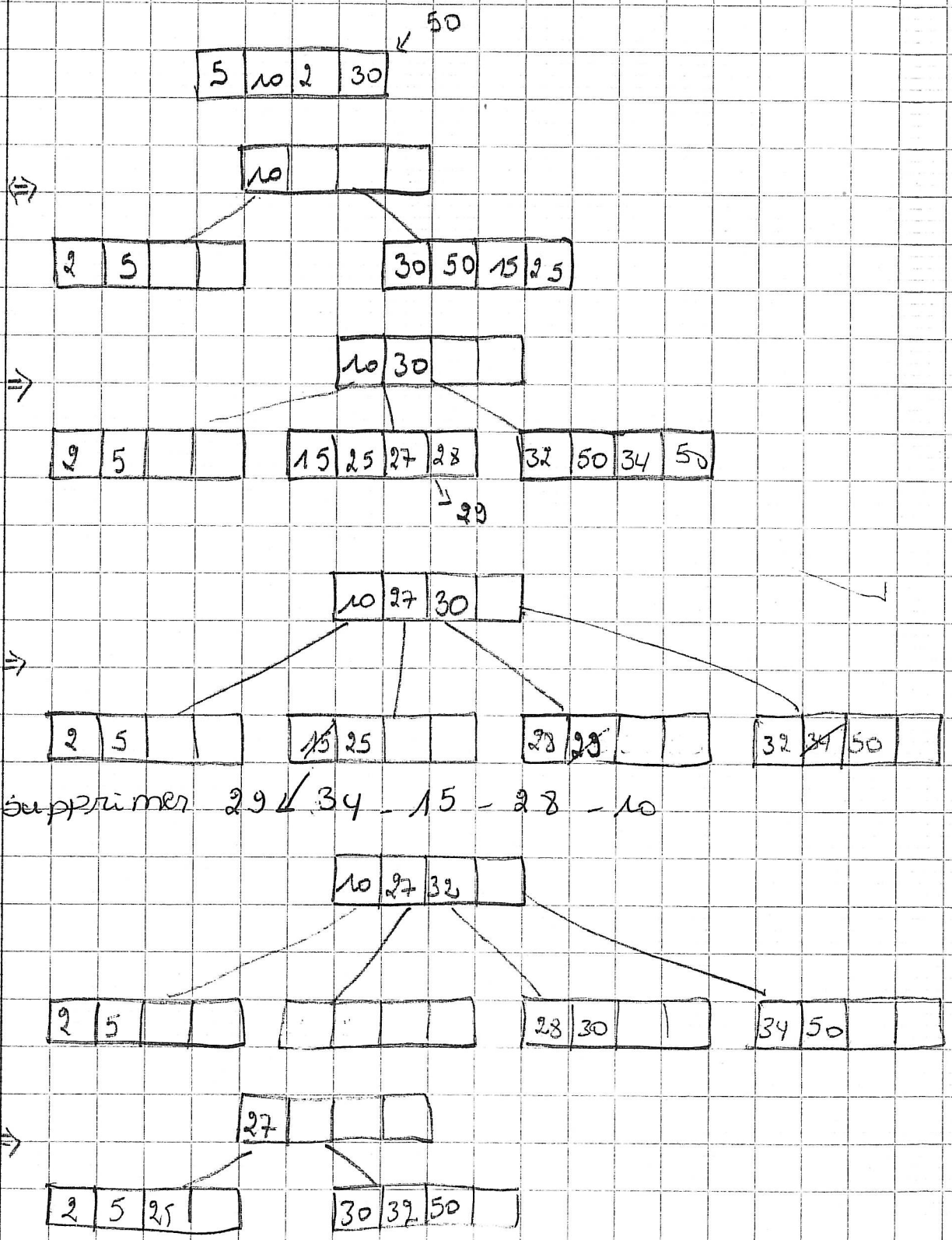


on le supprime



Exo 2 :

Construire un B-TREE d'ordre 2 qui contient les valeurs suivantes, 5 - 10 - 2 - 30 - 50 - 15 - 25 - 32 - 34 - 27 - 28 et 29. Et supprimer 29 - 34 - 15 - 28 - 10



Supprimer 29 // 34 - 15 - 28 - 10

CREATE INDEX <index>
ON <table>
PCT FREE <%>

Cette commande ORACLE permet de créer un index secondaire.

PCT FREE renvoie à l'espace qu'on laisse dans les nœu
(exple PCT FREE (1 (75%) ⇒ tout les nœuds sont remplis à
(75% (3 valeurs par nœud) mais à la création.

T.P Oracle

Se connecter à Oracle
Ouvrir terminal
ssh -x <login>@kutor
- export / home / oracle / oravr.sh .
- oravr.sh
  set
  sqlplus
  Login
  motd passe

Étudiant (n° number(5), nom_ varchar(30), prenom_ varchar(30), Ville varchar(50), Promotion varchar(20), Annee Number(4), Date_nai Date).

⇒ Remplir aléatoirement, la table Etudiant avec 100 occurences.

20% Dijon
30% Lyon
50% Paris.

TD 3.

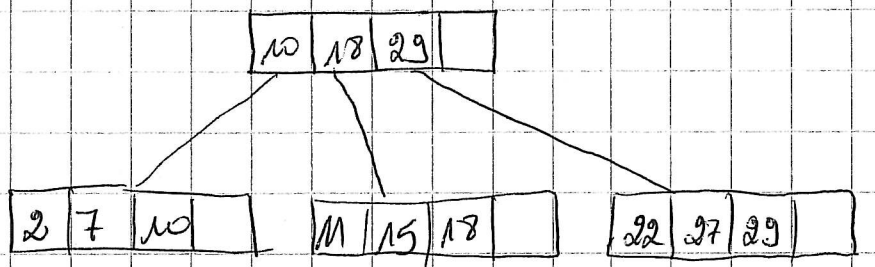
CREATE INDEX <index>
ON(TABLE> (<attributs>, ...
<fonction>)
PCT FREE <%>
⇒ Index secondaire
sous forme de B_TREE dont les nauds sont remplis. à (A_PCTFREE %).

⇒ À la création, les noeuds sont remplis à 75%. Comment Oracle atteint-il l'objectif de pct FREE.

EXP :

Nbre de valeurs à insérer = 9.

La forme de l'arbre est :



Nbre de niveaux = 2
Taux de remplissage = 75%

⇒ Ce qui nous donne 3 valeurs par feuille et on. enter la plus grande valeur de chaque feuille. mais avant on trie les valeurs à insérer.

2 - 7 - 10 - 11 - 15 - 18 - 22 - 27 - 29

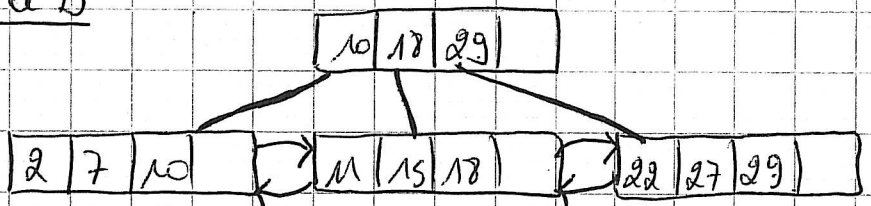
B^+ TREE

Intérêt : Lire les valeurs en séquence triée par le seul parcours des feuilles.

→ Toutes les valeurs d'index sont dans les feuilles.
→ Les feuilles sont doublement chaînés.

Les index index secondaires sous Oracle sont sous formes de B^+_tree.

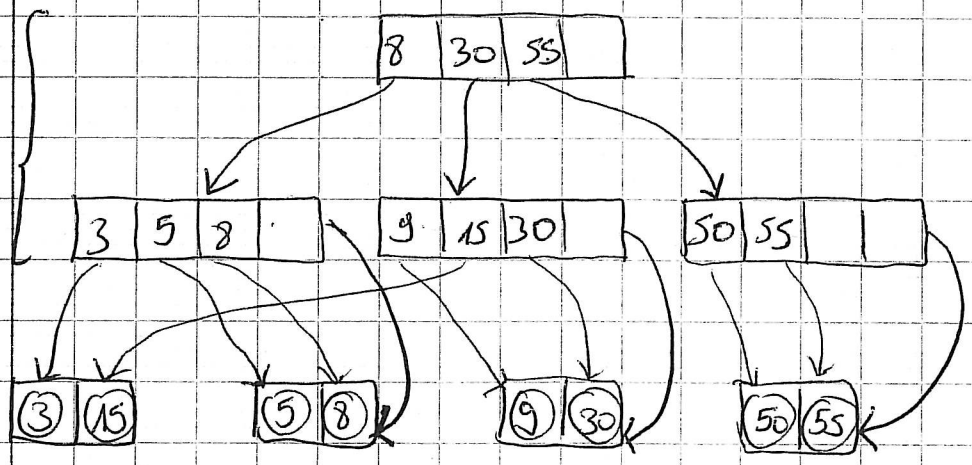
Exp de B^+



⇒ Organisation de fichier indexé sous forme de B-TREE (INGRES).

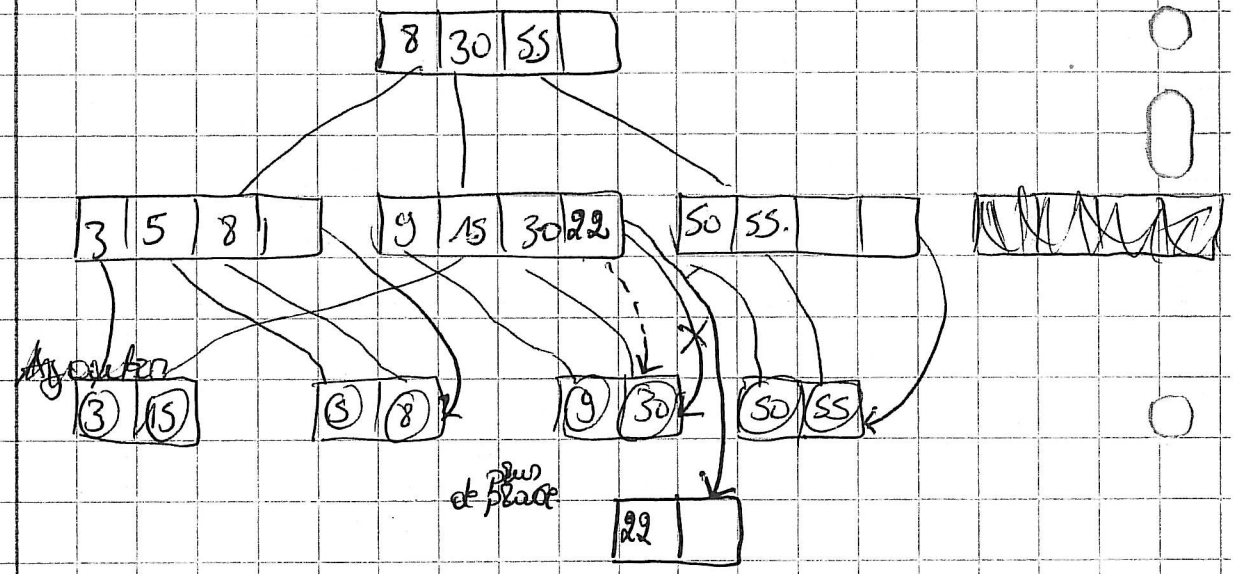
CREATE TABLE

→ Org. Seq.
→ " VSAM
→ ISAM
→ B-TREE des B-TREES Propre à INGRES

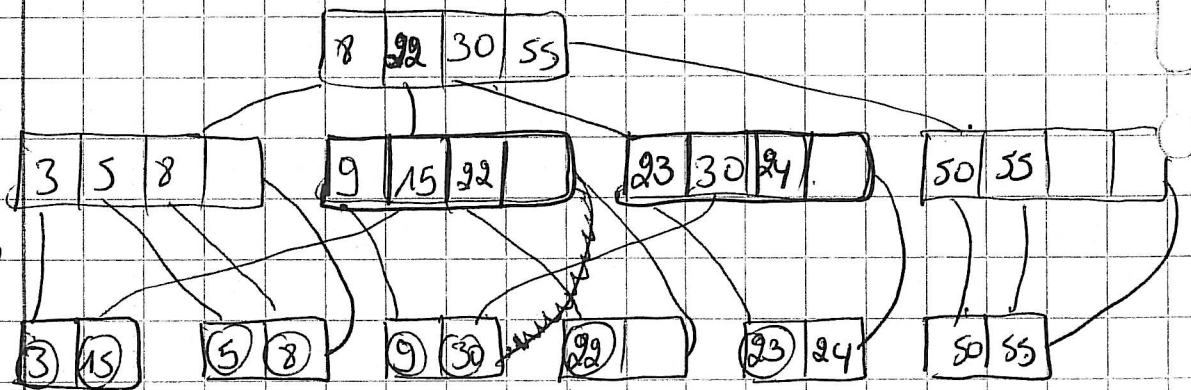


→ Branche d'index
→ Pointeur vers un enregistrement
→ Pointeur vers un bloc d'enregistrement

A chaque feuille de l'index est associé 1 bloc d'enregistrement (flèche verte) qui indique où doit être un enregistrement dont la valeur d'index est insérée dans la feuille en question.



Ajouter
→ 22
→ 23
→ 24



NB : On crée un autre bloc pour mettre 23 alors le bloc de 22 a + d'espace. Parce qu'on ne peut qu'avoir un seul pointeur (vert) ⇒ vers un bloc d'enregistrement.

- Les blocs d'enregistrement n'éclate pas, quand un un bloc est rempli, on crée un autre, on ne déplace pas les enregistrements. ⇒ moins couteux que sur VSAM.

- On recherche avec les flèches avec les rouges et on place avec les verts.
- les blocs d'enregistrement sont partiellement triés. (inconvénient).

III.B Sous Oracle, les flèches rouges sont des ROWID qui sont les adresses physiques des fichiers.
qui sont composés de : - n° de fichier
- n° de bloc
- n° de ligne.
- n° d'objet.

⇒ Réécrire lec_ind et ecr_ind en utilisant la structure d'index 'B_TREE sous INGRES'.

Est-ce l'algorithme marche avec cette structure de B-TREE ??

Lec_ind (fichier, index, val_index $\{^{1er}_{suivant}\}$) → Enreg, C.rend

Lec_val_index (index, val_index, ordre) → addr, enr, C.rend

choix_bon_fils (index, racine, val_index) → fils.

fils = index . racine
Tantque non feuille (fils)
choix_bon_fils (fils, val_index) → fils.
fin tantque

tant qu'on ne trouve pas la valeur d'index descend d's niveau

· Recherche_addr_enr (fils, val_index)
→ addr . enr, C . rend.

finir cet algo écrire ecr_ind.

Cr. SGBD — le 02/10/2012

STRUCTURE DE STOCKAGE

B. Oracle

1. Tables et domaines.

2. Les différentes organisations de fichiers utilisées.

ROWID : → Rowid "réduit" (objet)
→ Rowid "étendu" (n° de l'objet, n° fichier, n° bloc, n° de ligne)
Codifier en base 64, 10 octets.
→ Rowid logique. (couple de valeurs constitué de Rowid étendu, valeur d'index).

→ Pour manipuler les ROWID ETENDU, DBMS_ROWID.ROWID_BLOCK_NUMBER (rowid).

2.1. Organisation séquentielle

L'organisation par défaut qu'utilise oracle.

2.2. Organisation indexée

Apparue à la version 7.3.

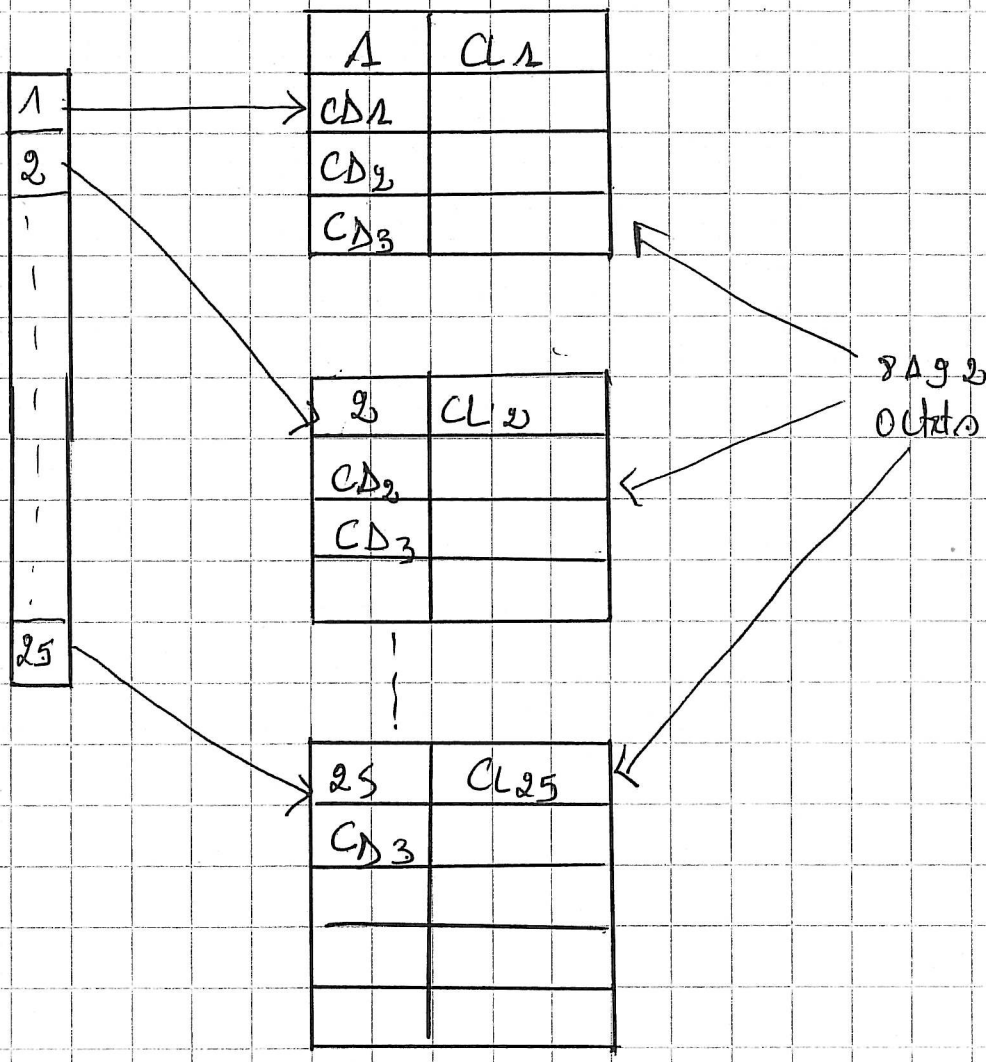
CREATE TABLE <Table> (<attributs>, ... ⊕
déclaration d'1 Primary Key) ORGANIZATION INDEX.
⇒ Organisation indexée primaire.
↳ On retrouve un B_TREE qui contient (Index et les tuples)
⇒ DATA INDEPENDANCE
↳ L'indépendance entre le schéma conceptuel et son implémentation.

2.3. Organisation en cluster

Spécifique à Oracle.
⇒ Un cluster permet de regrouper physiquement (ds la même bloc) des lignes de tables différentes.

CLIENT (n°client, Nom, adresse, ...)
CDE (n° cde, n°client, n° Prod, Qte).

CREATE CLUSTER CLICOM
(<colonne virtuelle>)
⇓ définition de la clé du cluster.
(n°clé number (3))
{SIZE {<nombre d'octets>
↙
Nbre d'octets attendus en moyenne par chaque valeur de clé.
SIZE 900
(permet de stocker plusieurs valeurs de clés sous_Oracle).
INDEX / [HASH is <colonne>]
HASH KEYS <nombre de buckets>.



Si choix d'un cluster indexé, il faut créer la table d'index

Suite syntaxe Cluster.

```
CREATE INDEX <index>
        ON CLUSTER.
CREATE TABLE CLIENT (Attributs)
    Cluster clicon (numcli)
CREATE TABLE COMMANDE (~~Attributs~~ nº Sclient)
    Cluster clicon (NºSclient)
```

⇒ Pourquoi il faut dire explicitement à Oracle de créer un index dans la synthaxe du cluster?
C'est pour définir nous même les paramètres selon nos besoins.

2.4 Index secondaire

CREATE INDEX <index>
ON <table> (<Attributs 1>,<...>)
ou (<Fonctions>).
PCTFREE <%>
NOSORT

NB
Les valeurs null (qui n'existe pas) ne sont pas stocké dans l'index.

2.5 Les index BITMAP

Ex: PERSONNE(nom, Prenom, statutmarital)
(C, M, D, V, P)

CREATE BITMAP INDEX ON <table> (statutmarital).

	C	M	D	V	P
Marié	0	1	0	0	0
Divorcé	0	0	1	0	0
Marié	0	1	0	0	0
Pacsé	0	0	0	0	1.

NB Les ROWID ne sont pas stocké dans cette structure, Parceque la la 1ère ligne de la table correspond à la première ligne d'index, la 2ème ligne de la table correspond à la deuxième ligne de l'index.

C. Notion de tablespace

définit 1 espace de stockage.

⇒ Un tablespace est réalisé physiquement par 1 ou plusieurs fichiers.

CREATE TABLESPACE < tablespace > SIZE
DATAFILE < chemin de fichier > < taille >
[ONLINE / OFFLINE]

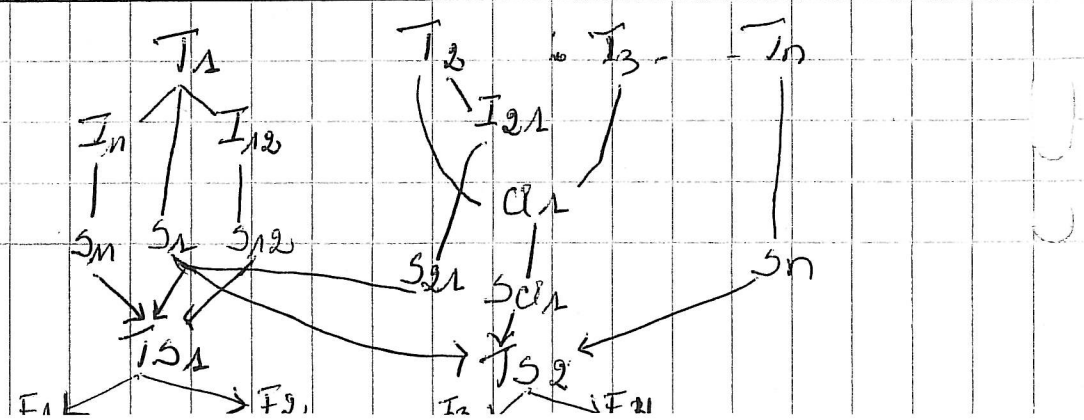
⇒ 1 tablespace permet de regrouper des objets Oracle selon des critères :
- Par type d'objet (table, index ...)
- Par application.

ALTER TABLESPACE < tablespace > { OFFLINE } Pour déconnecter.

Par exemple la table PAYE qui gère les paies des salariés n'est utilisé qu'une seule fois dans l'année. On peut le stocker sur un autre disque dur et le mettre (ONLINE) qu'en cas de besoin. Une fois on finit on le déconnecte et remet à sa place. (OFFLINE).

D. Segments, extents.

T : table.
I : index
C : cluster
S : segments.
TS : tablespace.
F : fichier



TRUNCATE

III.3 : Sous oracle un segment porte le nom de l'objet qui implémente.

Un segment est un ensemble de blocs :

- 1 segment = ensemble de bloc.
- 1 segment = 1 ensemble d'extents.
- 1 extents = 1 suite de blocs consécutifs.
  - ↳ sont décrit dans USER_FREE_SPACE.

⇒ Politique d'allocation d'espace

STORAGE (INITIAL <nbre d'octets>, // taille de l'extent initial
NEXT <nbre d'octets> // 2ème Extents.
PCT_INCREASE <%>,
MAX EXTENTS <nbre> // nbre d'extents Max.
MIN EXTENTS <nbre> // " " min.
OPTIMAL <nbre d'octets>

LLBACK SEGMENT :
n met d'annuler
pération effectuée
ltérieurement.

⇒ Taille de INITIAL ≥ Taille de NEXT ≥ Taille de PCT_INCREASE.

PCT_INCREASE :

⇒ Mode dictionnary : Définit, à la création du Tablespace.

↳ Remplacé par le mode Local, Espace libre décrit dans 1 Bitmap en entête de chaque tablespace index binaire.

E. STRUCTURE DES BLOCS DE DONNÉES.

Blocs de données (8192 octets)

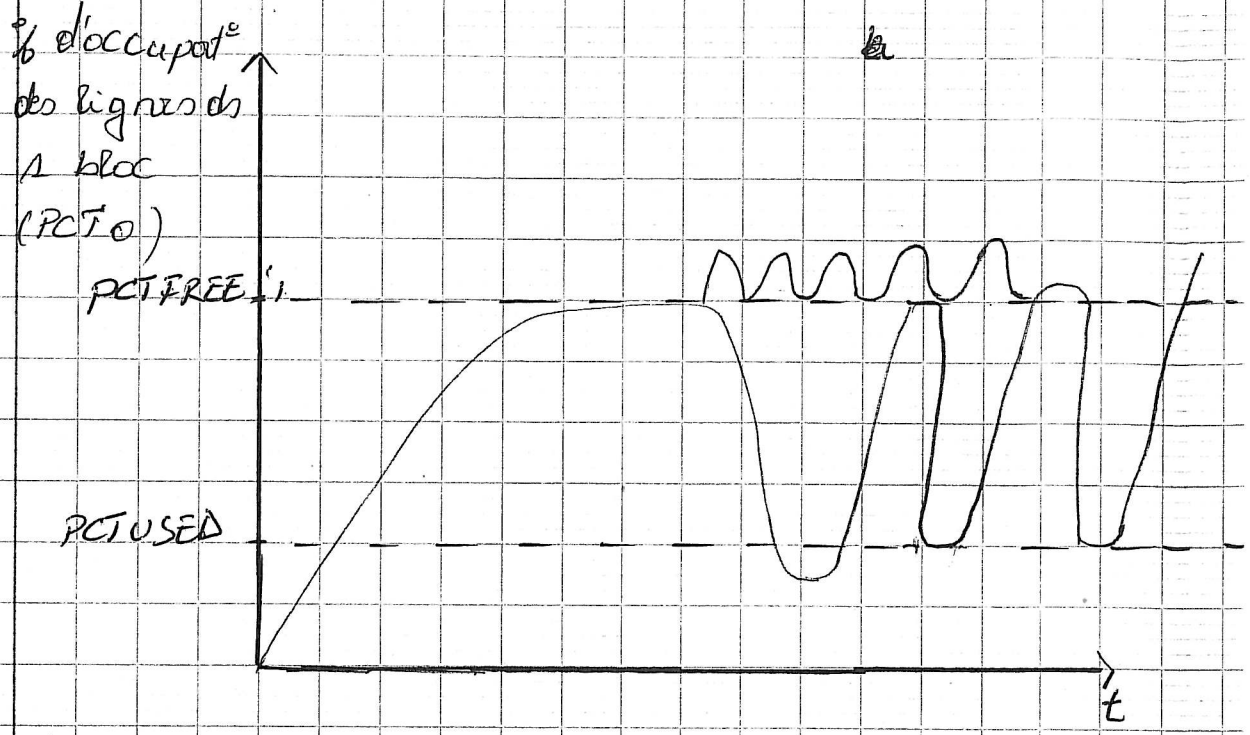
Entêtes du Bloc.	50 octets (@ de Blocs.
Repertoire des tables Ajout des lignes dans le bloc	2 octets / table
Repertoire des lignes du bloc	2 octets / ligne
Repertoire de transaction.	23 octet / transaction (INITRANS / MAXTRANS).
Espace libre	
Lignes des tables	

F. Gestion des lignes dans 1 bloc de données.

1. Paramètres PCTFREE & PCTUSED (CREATE TABLE ---- PCTFREE 10)

PCTFREE spécifie le % du disque laissé libre lors de l'insertion de lignes dans un bloc.

2. PCTUSED spécifie le % d'occupation 1 Bloc à partir duquel le bloc peut à nouveau accepter des lignes.



⇒ Free. list. Blocs : liste des blocs pouvant accepter des nouvelles lignes.

⇒ PCTUSED remplacé par 1 Bitmap stocké en tête de chaque segment.

G. Structure d'une ligne de données.

Entête	Référence de la table	Rowid du morceau suivant	Valeurs des colonnes

3 ø

1 Valeur de colonne : <longueur> <valeur>

↓

Stockée sur 1 à 3 ø

III. ALGORITHMES DES OPERATIONS DE L'ALGEBRE RELATIONNELLES.

A. SELECTION

Il y a au plus 2 algo possibles.

⇒ Balayage séquentiel

⇒ Accès direct (si il y a un index ou fonction hash_code)

→ Coût = Nbre de Blocs qui contient au moins 1 ligne à parcourir (BR).

→ Coût = $\frac{TR}{Ia}$.

Ia : Nombre des valeurs différentes prise par l'attribut 'a' sur lequel porte le critère de sélection.

BR : Bloc de R.

TR : Tuple de R.

Exple1

(Selection(R, a = v)).

Coûts selon le type de critère de sélection.

→ a = v → Coût = TR / Ia

→ a > v → Coût ⇒ TR / 2 ou

⇒ $\frac{(B_{sup} - v)}{(B_{sup} - B_{INF})}$ ou

Cf. histogramme de valeurs s'il existe.

Exple 2

$T_{Client} = 10.000$

B_{Client} = 10 clients par blocs = 1000

→ $a \neq v \Rightarrow$ balayage séquentiel.

→ like 'A%' $\Rightarrow \hat{C} = \frac{1}{2^3} \cdot (TR)$ ⇒ Ça n'existe pas, Pas encore implémenter.

Exple

$$\left(\underset{I_s}{a = v_1} \text{ et } \underset{I_s}{b = v_2} \right)$$

I : indexes.

⇒ Si utilisation des 2 indexes, lire les ROWID des lignes qui vérifient $a = v_s \Rightarrow ER_1$.

Lire les ROWID des lignes qui vérifient $b = v_2 \Rightarrow ER_2$

$ER = ER_1 + ER_2$

Lire les lignes dont le ROWID est dans ER

$\hat{C} = TR / I_a / I_b$.

0646 43 47 83

Loc_ind (fichier

TD4

le 09/10/2012

(Suite de l'exercice du TD3).

Ecrind (fichier, enregistrement) → Fichier, C-err
Ecr_err (fichier.FPD, enregistrement) → Fichier.FPD, adr err, C_err

Si C_err = Pas d'erreur alors
  Pour H index dans fichier.liste_index faire
    → extraire.val_ind (enrgt, index.description) → Valind
    Maj_index (index, val_ind, adr.enrgt) → index.

Extraire la valeur d'index grâce à la description.

⇒ Adapter cet algorithme sous ingres.

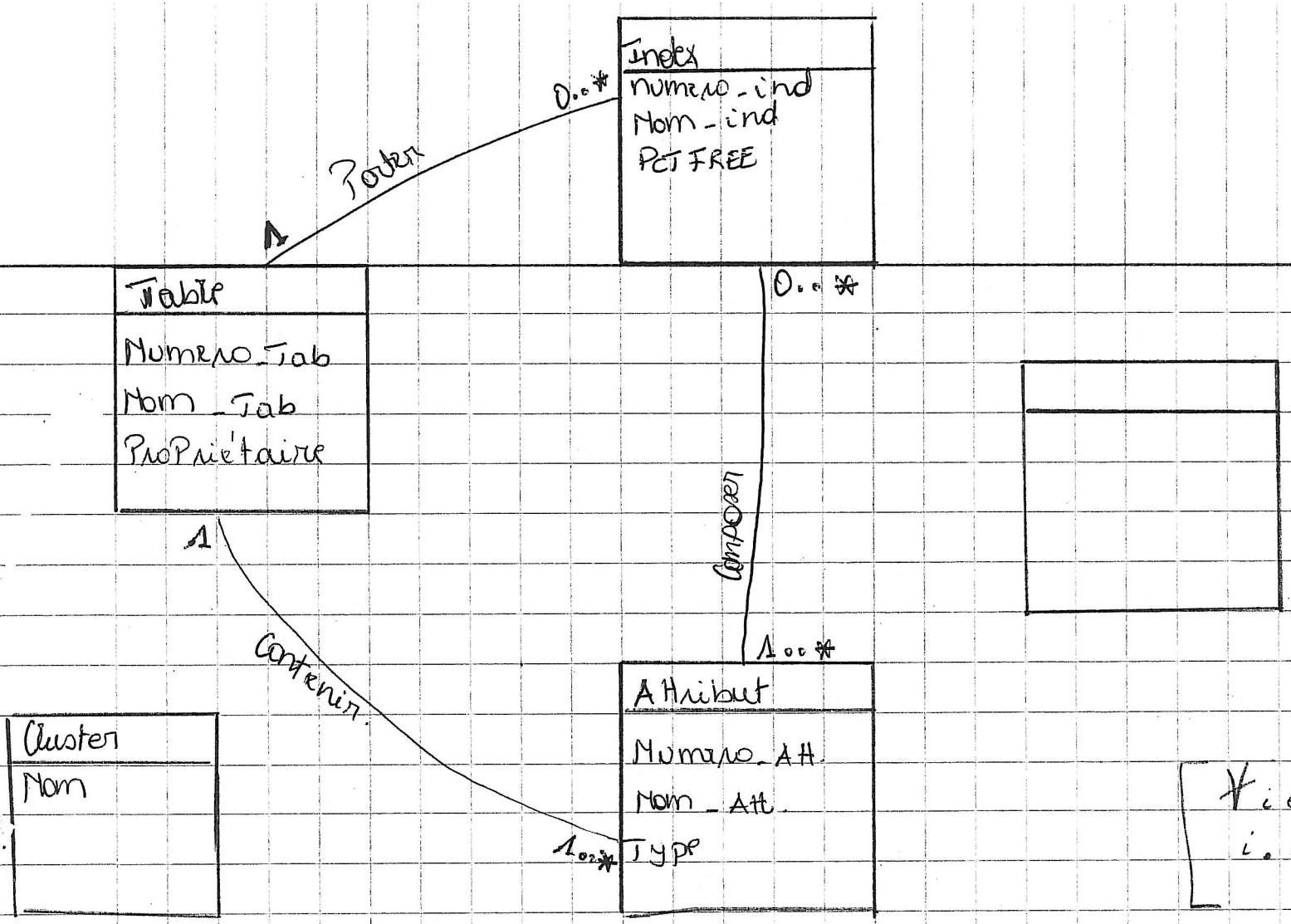
B; Ici Oracle va accéder à la nième valeur il faut donc connaître le nombre d'enregt.

SCHEMA UML du dictionnaire de données Oracle

Dictionnaire de données; Base des données relationnelles où on trouve la structure des bases des données.
décrit. de la

NOCLI	3.	3.
SFI	Nom	Prom.

⇒ Construire le schéma UML qui décrit les notions.
- Attributs
- Table
- index
- Contrainte d'intégrité
- Cluster.



$\forall i \in index$
$i.attribut \subseteq i.table.Attribut$

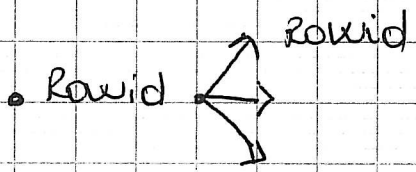
=> Quand on crée une table avec un Primary Key, l'index secondaire est créé sur la clé, par contre si on on précise ORGANIZATION INDEX, sa clé l'index est alors primaire.

=> Pourquoi oracle le fait-il ? Pour question d'efficacité

Select Rowid
from client } se pointe sur la ligne de la table.

Rowid, → n° de fichier
→ n° de bloc
→ n° de ligne
→ n° d'objet

- Rowid étendu



—

9ème tp.

le 16/10/2013

T.D4.

Ecrire les requêtes suivantes sur le dictionnaire de données

1) Afficher les tablespaces dans lequel l'utilisateur 'x' a au moins une table ou un index.

=> Un tablespace permet de regrouper par thèmes les objets sous Oracle.

```
SELECT TABLESPACE_NAME FROM USER_TABLESPACES
  WHERE TABLESPACE_NAME IN (SELECT TABLESPACE_NAME
  FROM ALL_INDEXES WHERE OWNER='X') OR
  (SELECT TABLESPACE_NAME FROM ALL_TABLE WHERE
   OWNER='X').
```

Ou

```
SELECT DISTINCT TABLESPACE_NAME FROM DBA_TABLES
 WHERE OWNER='X'
UNION
SELECT TABLESPACE_NAME FROM DBA_INDEXES
 WHERE OWNER='X'.
```

Ou

```
SELECT TABLESPACE SEGMENT_NAME FROM DBA_SEGMENTS WHERE
 OWNER='X'.
    AND (SEGMENT_TYPE='INDEX' OR
         SEGMENT_TYPE='TABLE')
```

2) Afficher pour chaque table sa taille en nombre de Blocs.

⇒ Quand on crée un segment, il prend le même nom que la table.

```
SELECT BLOCKS, SEGMENT_NAME FROM
    USER_SEGMENT WHERE SEGMENT_TYPE = 'TABLE'.
```

Ou.

```
SELECT TABLE_NAME, BLOCKS
    FROM USER_TABLES, USER_SEGMENT
    WHERE USER_TABLE.TABLE_NAME = USER_SEGMENTS.
            SEGMENT_NAME.
```

3) Afficher pour chaque tablespace le nombre d'extents libres ainsi que leurs tailles moyenne, minimale, maximale et totale.

```
SELECT COUNT(*), AVG(BLOCKS), MIN(BLOCKS),
  MAX(BLOCKS), SUM(BLOCKS), TABLESPACE_NAME
    FROM USER_FREE_SPACE GROUPE BY TABLESPACE_NAME.
```

4) Afficher par tablespace et par intervalle de taille le nombre d'extents libres.

[0 - 10.000 [

[10.000 - 20.000 [

[20.000 - 30.000 [.

donne un numéro

```
SELECT TABLESPACE_NAME, '['||TRUNC(BLOCKS/10000)*10||
TRUNC(BLOCK/10000)+1)*10||']', COUNT(*)
FROM USE_FREE_SPACE
GROUP BY TABLESPACE_NAME, BINF, BSUP;
```

Alias

Alias

```
)*10 BINF;
```