Cours de bases de données, aspects systèmes, http://sys.bdpedia.fr

Le hachage statique

Philippe Rigaux

le cnam

Le hachage

Très utilisé comme structure en mémoire RAM.

Pour les bases de données, un concurrent de l'arbre-B

- **Meilleur** (un peu, et en théorie) pour les recherches par clé
- **N'occupe aucune place**

Mais

- Se réorganise difficilement
- Ne supporte pas les recherches par intervalle

Principe du hachage

Le stockage est organisé en N **fragments** (*buckets*) constitués de séquences de blocs.

La répartition des enregistrements se fait par un **calcul**. Une *fonction de hachage* h prend une valeur de clé en entrée et retourne une adresse de fragment en sortie

- **Stockage** : on calcule l'adresse du fragment d'après la clé, on y stocke l'enregistrement
- **Recherche (par clé)** : on calcule l'adresse du fragment d'après la clé, on y cherche l'enregistrement

Simple et efficace !

Exemple

On veut créer une structure de hachage pour nos 16 films

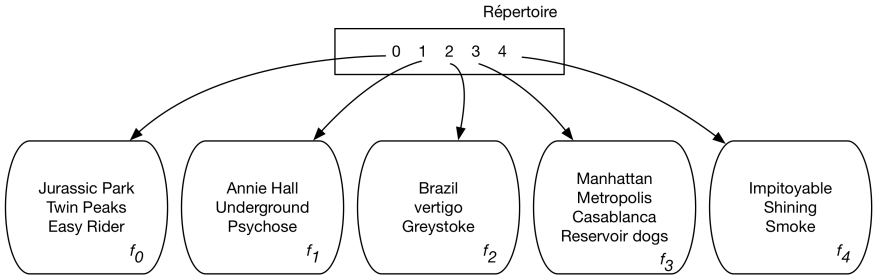
Simplifions : chaque fragment fait un bloc et contient 4 enregistrements au plus

- on alloue 5 fragments (pour garder une marge de manœuvre)
- un **répertoire** à 5 entrées (0 à 4) pointe vers les fragments
- On définit la fonction $h(titre) = rang(titre[0]) \; mod \; 5$

Donc on prend la première lettre du titre (par exemple 'I' pour 'Impitoyable'), on prend sot rang dans l'alphabet (ici 9) et on garde le reste de la division par 5, le nombre de fragments.

$$h('Impitoyable') = 4$$

Le résultat



La seule structure additionnelle est le répertoire, qui **doit** tenir en RAM

Recherches

- Par clé , Oui

```
SELECT * FROM  Film WHERE titre = 'Impitoyable'
```

- Par préfixe ? Non

```
SELECT * FROM  Film WHERE titre LIKE 'Mat%'
```

- Par intervalle : non !

```
SELECT *
FROM  Film
WHERE titre BETWEEN 'Annie Hall' AND 'Easy Rider'
```

Les difficultés

Très important : h doit répartir uniformément les enregistrements dans les n fragments

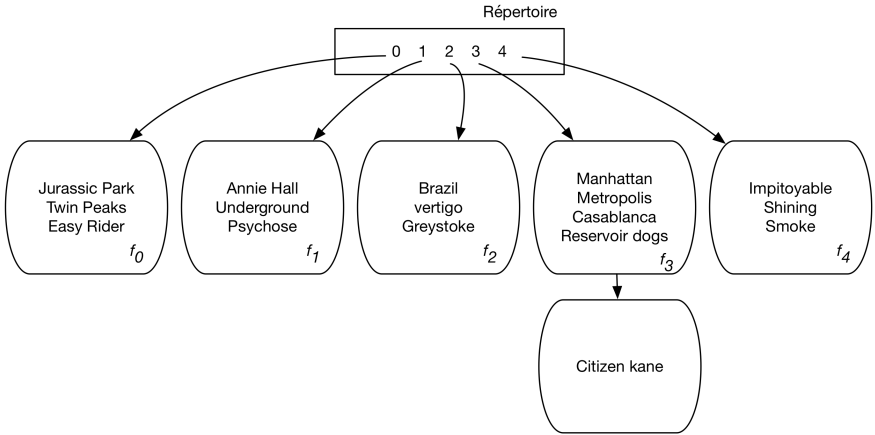
Notre fonction est un contre-exemple : si une majorité de films commence par une même lettre ('L' par exemple) la répartition va être déséquilibrée.

Plus grave : La structure simple décrite précédemment n'est pas **dynamique**.

- On ne peut pas changer un enregistrement de place
- Donc il faut créer un chaînage de blocs quand un fragment déborde
- Et donc les performances se dégradent...

Notez que l'on ne peut pas changer le nombre de fragments
car cela implique le changement de la fonction de répartition.

Exemple : insertion de Citizen Kane



Il faut maintenant deux lectures pour accéder à Citizen Cane.

Résumé : avantages et inconvénients du hachage

En terme d'efficacité, il paraît optimal

- La structure (le répertoire) prend très peu de place
- Le coût d'une recherche par clé est constant
- De plus il est particulièrement simple à implanter.

Mais

- Dans sa version de base, il se dégrade quand la situation évolue
- Pas de recherche pas intervalle ou préfixe
- Structure plaçante : il ne peut y avoir qu'une seule structure de hachage par table.