ARCHIVI B-TREE

L'organizzazione B-TREE (Balanced Tree) utilizza una struttura ad **albero bilanciat**o contenente le chiavi disposte in modo **ordinato**.

E' in pratica una generalizzazione degli alberi binari di ricerca (ABR) bilanciati.

L'archivio prevede due file:

- un file **dati** di tipo sequenziale ad accesso diretto con tutti i dati;

- un file **indice** contenente le chiavi, disposte ordinatamente in una struttura ad albero, ed i riferimenti ai record del file dati.

L'albero non è binario, l'ordine (o grado) M è di solito elevato (centinaia); i suoi nodi, detti "pagine", contengono le seguenti informazioni:

P1 | K1 | P2 | K2 | P3 |.....| Pm-1 | Km-1 | Pm

In ogni pagina vi sono:

- fino a M-1 chiavi ordinate con i puntatori ai record dell'archivio dati;

- fino a M puntatori ai sottoalberi; nei nodi di un sottoalbero sono contenute tutte le chiavi comprese tra le due chiavi adiacenti al puntatore del sottoalbero.

In una pagina il primo puntatore P1 indirizza il sottoalbero contenente tutte le chiavi minori della prima chiave K1, il secondo puntatore P2 indirizza il sottoalbero contenente le chiavi maggiori di K1 e minori di K2, l'ultimo puntatore Pm punta al sottoalbero contenente le chiavi maggiori di Km-1.

Se l'albero è di ordine M ogni nodo deve contenere almeno N chiavi con $(M-1)/2<N<M-1$ ossia sia occupato almeno al 50%; anche i sottoalberi sono almeno M/2.

Solo la radice può avere un numero minore di chiavi, tra 1 e M-1.

Si può dimostrare che il numero di accessi L, pari al numero di livelli dell'albero, è:

$$\log_M(NR+1) < L < \log_{M/2}(NR+1)/2$$

con NR=n.dei records e M=grado dell'albero

se NR=1.048.576 e M=32 si ha $4<L<4.75$

se NR=1.048.576 e M=256 " $2.5<L<2.6$

Gli inserimenti vengono effettuati, dopo aver eseguito la ricerca senza successo, in una pagina foglia; ciò risulta semplice se la pagina non è piena.

Nel caso la pagina risulti piena occorre procedere ad uno sdoppiamento (splitting) delle chiavi in due pagine distinte ed all'inserimento nella pagina padre di una ulteriore chiave e di un nuovo puntatore; il riempimento delle pagine si può ripercuotere ai livelli sovrastanti fino ad interessare la radice, in questo caso lo sdoppiamento della radice comporta l'inserimento di un nuovo livello nell'albero.

Le cancellazioni risultano semplici nel caso di pagine terminali contenente un numero di chiavi maggiore del valore minimo ((M-1)/2).

Negli altri casi la cancellazione risulta complessa e richiede lo spostamento di chiavi da una pagina ad un'altra (underflow) oppure la fusione tra due pagine con eliminazione di una delle due; occorre spesso modificare anche i livelli precedenti.

E' utilile aggiungere in ogni pagina il puntatore al padre per agevolare le modifiche su più livelli.

Sia negli inserimenti che nelle cancellazioni è necessario che la struttura mantenga le caratteristiche originali: l'albero deve sempre essere bilanciato, ordinato e con occupazione delle pagine superiore al 50%.

Sono state sviluppate molte **varianti** della organizzazione B-TREE, le più note sono:

B*-TREE : E' composto da un file indice contenente le chiavi organizzato come nel B-albero, ma con le chiavi contenute nella radice e nei primi livelli ripetute anche nei livelli sottostanti.
Le pagine dell'ultimo livello contengono tutte le chiavi e l'indirizzo dei records dell'archivio dati, oppure sono direttamente memorizzate nell'archivio dati.

B+-TREE : (usati nei File VSAM della IBM) prevedono un'occupazione delle pagine superiore ai 2/3.
L'ultimo livello è contenuto nel file dati con tutte le informazioni; si ha in pratica un archivio ISAM con indice dinamico (struttura dinamica ad albero).

In entrambi i casi le pagine terminali sono linkate tra di loro per consentire un veloce accesso sequenziale ordinato.

L'organizzazione B-TREE viene anche detta ad "**indice dinamico**" mentre l'organizzazione ISAM dispone di un indice statico (array ordinato); la B-TREE garantisce quindi la possibilità di modificare l'indice con minime variazioni al file ed è quindi migliore nei casi di frequenti inserimenti e cancellazioni.

A differenza dell'organizzazione HASH consente la scansione ordinata e la lunghezza massima della ricerca è limitata al numero dei livelli dell'albero; se il grado dell'albero è elevato anche il tempo medio per la ricerca risulta ridotto, di poco superiore a quello hash.

Vantaggi :

- Se M (ordine dei nodi) è elevato (centinaio) si ha un basso numero di accessi, simile all'organizzazione hash.

- Il numeri di accessi massimo è pari al numero di livelli (più l'accesso all'archivio dati).

- Si può eseguire una scansione ordinata su tutto l'archivio o su una parte.

- Non occorre nessuna riorganizzazione dell'archivio dati principale.

Svantaggi :

- Occupazione di memoria non ottimale.

- Procedure di inserimento e cancellazione complessa per mantenere l'albero bilanciato.

Conclusioni :

Questa organizzazione risulta in generale la migliore in quanto è efficiente in tutte le operazioni da svolgere normalmente in un archivio.

E' quindi largamente utilizzata nelle più moderne applicazioni ed in particolare nei data-base per velocizzare le operazioni di ricerca ed ordinamento.