DBMS

PARTE I

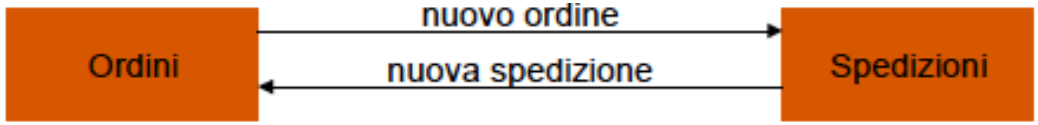
DBMS

- un DBMS è un sistema software che gestisce grandi quantità di dati persistenti e condivisi

- La gestione di grandi quantità di dati richiede particolare attenzione ai problemi di efficienza (ottimizzazione delle richieste, ma non solo!)
- La persistenza e la condivisione richiedono che un DBMS fornisca dei meccanismi per garantire l'affidabilità dei dati (fault tolerance), per il controllo degli accessi e per il controllo della concorrenza
- Diverse altre funzionalità vengono messe a disposizione per motivi di efficacia, ovvero per semplificare la descrizione dei dati, lo sviluppo delle applicazioni, l'amministrazione di un DB, ecc.

Condivisione dei dati

- La gestione integrata e la condivisione dei dati permettono di evitare ripetizioni (ridondanza dovuta a copie multiple dello stesso dato), e quindi un inutile spreco di risorse (memoria)
- Inoltre, la ridondanza può dar luogo a problemi di inconsistenza delle copie e, in ogni caso, comporta la necessità di propagare le modifiche, con un ulteriore spreco di risorse (CPU e rete)
- Esempio: il settore Ordini di un'azienda manifatturiera memorizza i propri dati in un file, non condiviso con gli altri settori aziendali. Ogni volta che arriva un ordine, i dati relativi devono essere trasmessi al settore Spedizioni, affinché l'ordine possa essere evaso. A spedizione eseguita, i dati relativi devono essere ritrasmessi al settore Ordini



Il modello dei dati

- Dal punto di vista utente un DB è visto come una collezione di dati che modellano una certa porzione della realtà di interesse
- L'astrazione logica con cui i dati vengono resi disponibili all'utente definisce un **modello dei dati**; più precisamente:
  - un modello dei dati è una collezione di concetti che vengono utilizzati per descrivere i dati, le loro associazioni/relazioni, e i vincoli che questi devono rispettare
- Un ruolo di primaria importanza nella definizione di un modello dei dati è svolto dai meccanismi che possono essere usati per strutturare i dati

  (cfr. i costruttori di tipo in un linguaggio di programmazione)
- Ad es. esistono modelli in cui i dati sono descritti (solo) sotto forma di alberi (modello gerarchico), di grafi (modello reticolare), di oggetti complessi (modello a oggetti), di relazioni (modello relazionale)

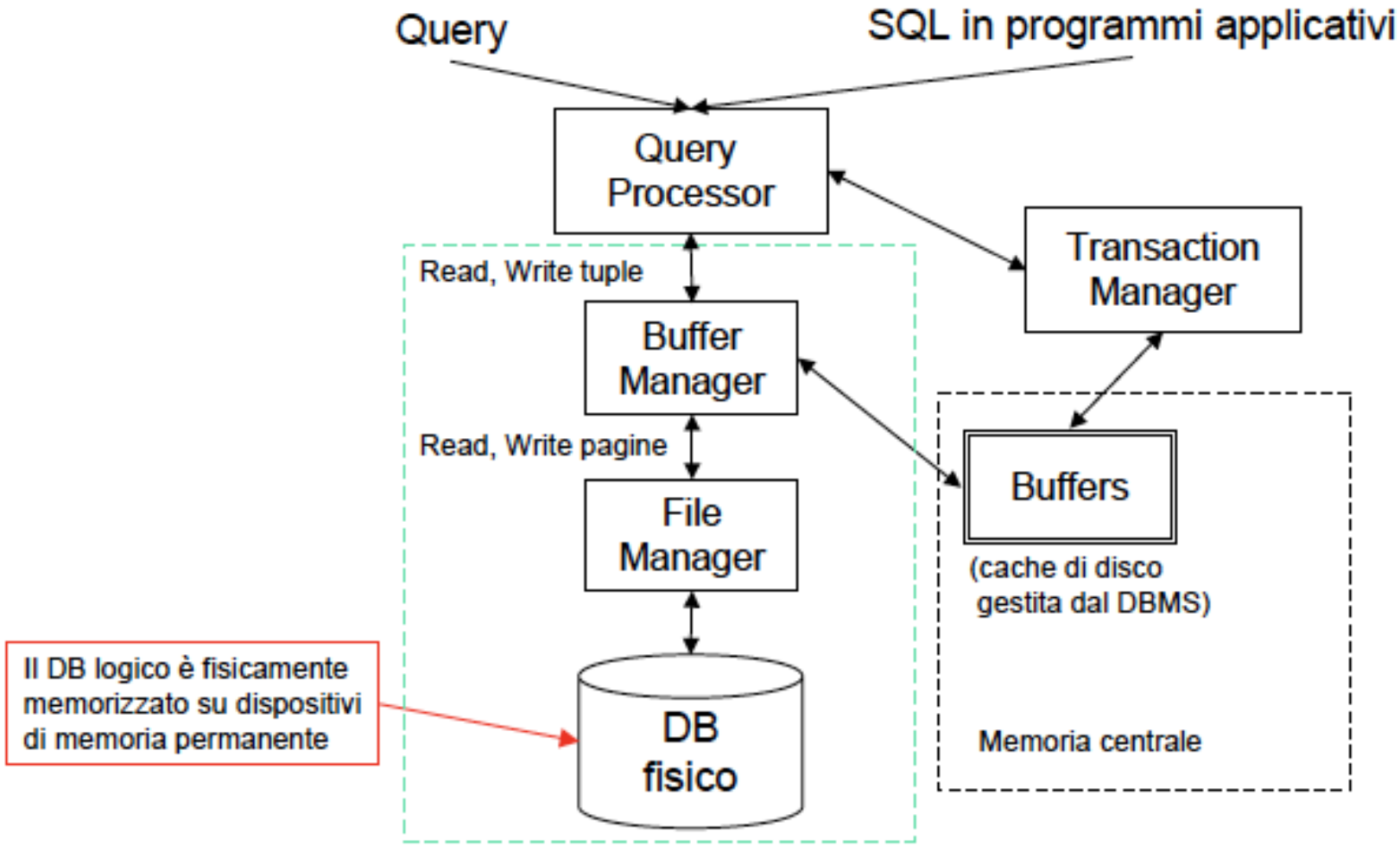
Indipendenza fisica e logica

- Tra gli obiettivi di un DBMS vi sono quelli di fornire caratteristiche di:

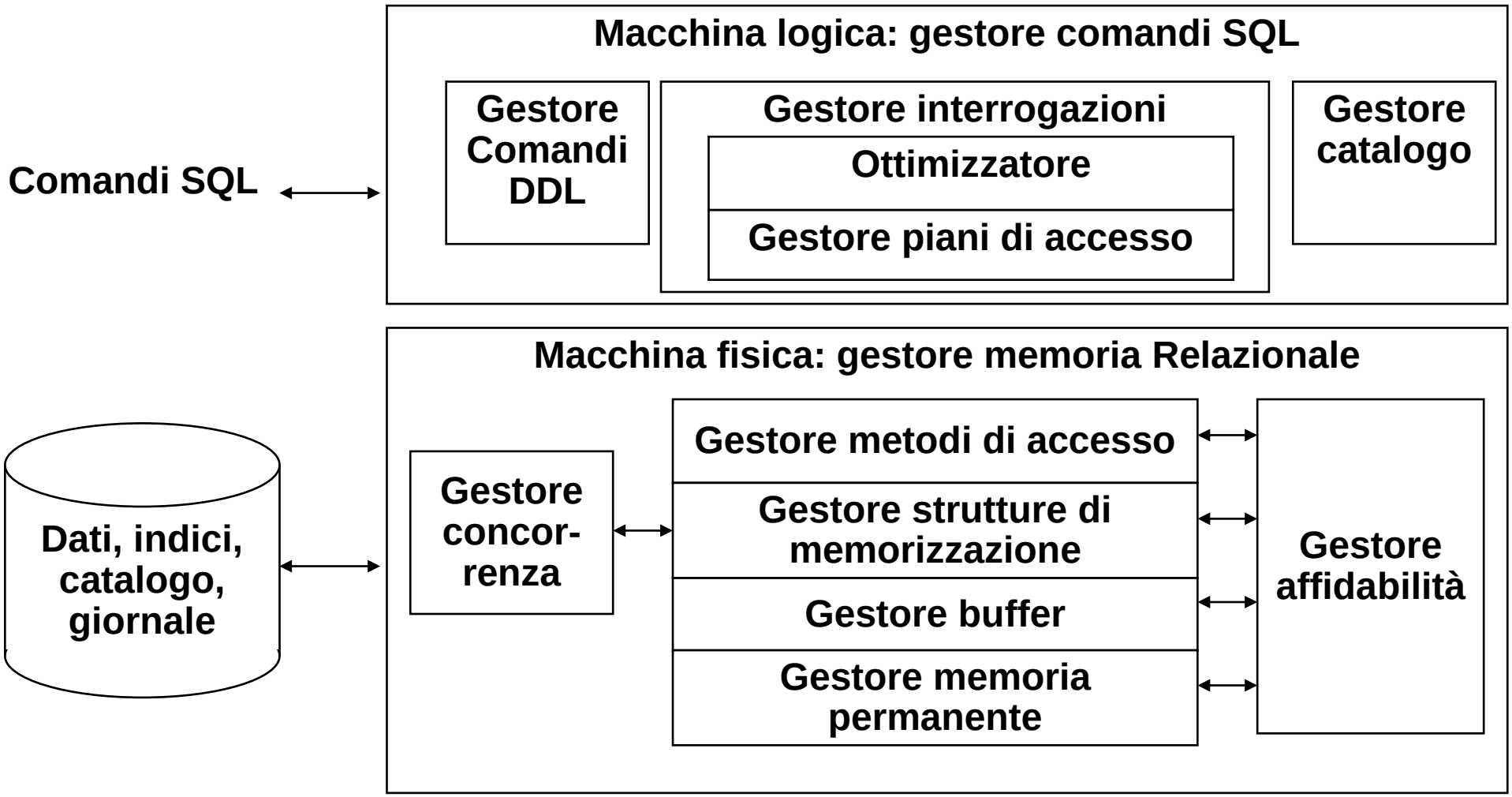
- Indipendenza fisica
  - L'organizzazione fisica dei dati dipende da considerazioni legate all'efficienza delle organizzazioni adottate. La riorganizzazione fisica dei dati **non deve comportare effetti collaterali sui programmi applicativi**

- Indipendenza logica
  - permette di accedere ai dati logici indipendentemente dalla loro rappresentazione fisica

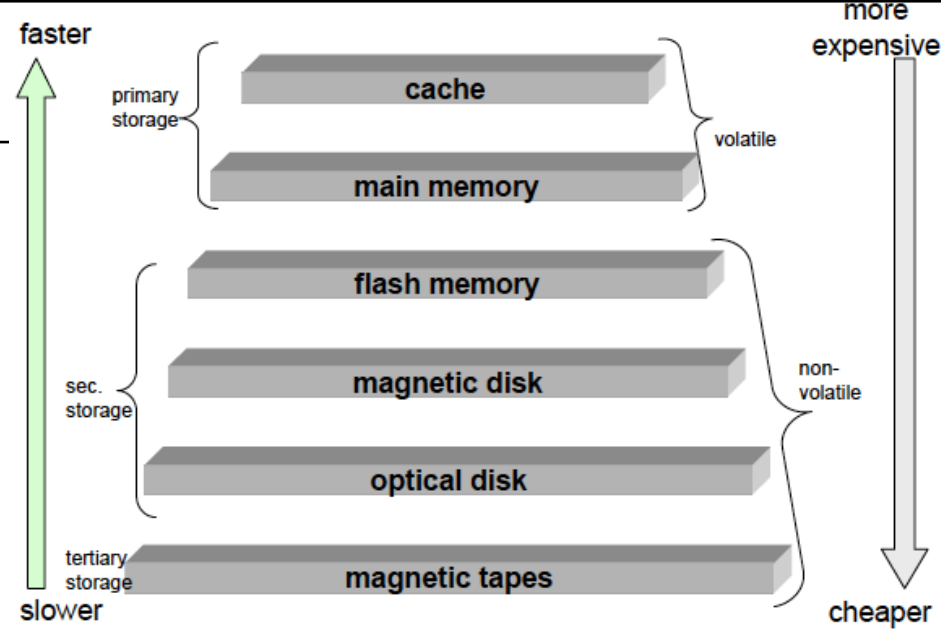
L'architettura (semplificata) di un DBMS



ARCHITETTURA DEI DBMS



Gerarchia delle memorie



- La memoria di un sistema di calcolo è organizzata in una gerarchia.
- Al livello più alto memorie di piccola dimensione, molto veloci, costose; scendendo lungo la gerarchia la dimensione aumenta, diminuiscono la velocità e il costo
- Prestazioni di una memoria:
  - dato un indirizzo di memoria, le prestazioni si misurano in termini di tempo di accesso, determinato dalla somma della
    - latenza (tempo necessario per accedere al primo byte), e del
    - tempo di trasferimento (tempo necessario per muovere i dati)

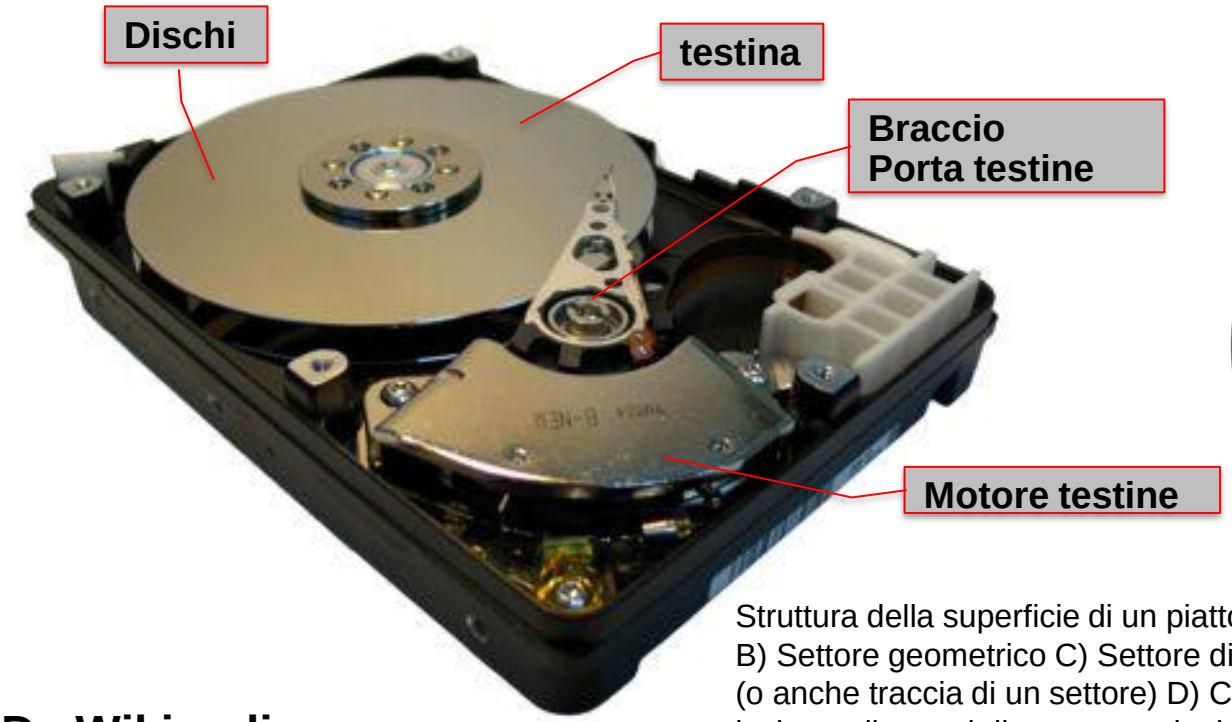
$$\textbf{Tempo di accesso} = \textbf{latenza} + \frac{\textit{dimensione dati da trasferire}}{\textit{velocità di trasferimento}}$$

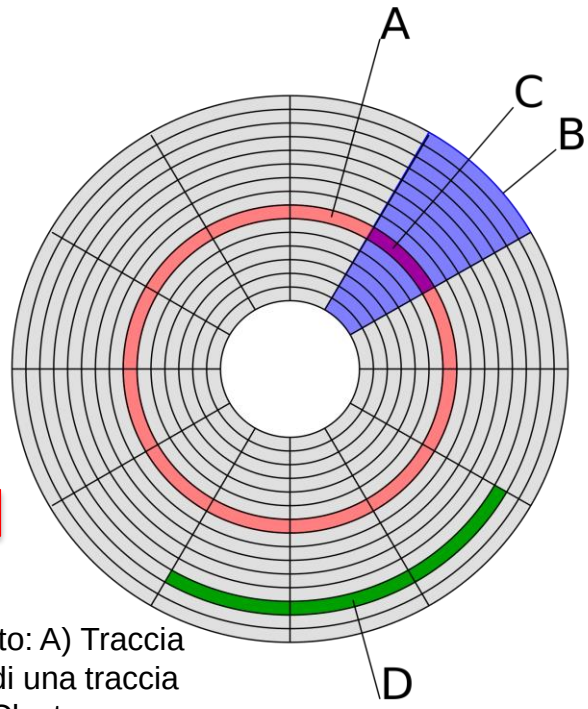
Implicazioni per i DBMS

- Un DB, a causa della sua dimensione, risiede normalmente su dischi (ed eventualmente anche su altri tipi di dispositivi)
- I dati devono essere trasferiti in memoria centrale per essere elaborati dal DBMS
- Il trasferimento non avviene in termini di singole tuple, bensì di blocchi (o **pagine**, termine comunemente usato quando i dati sono in memoria)
- Poiché spesso le operazioni di I/O costituiscono il collo di bottiglia del sistema, si rende necessario ottimizzare l'implementazione fisica del DB, attraverso:
  - Organizzazione efficiente delle tuple su disco
  - Strutture di accesso efficienti
  - Gestione efficiente dei buffer in memoria
  - Strategie di esecuzione efficienti per le query

Gli hard disk

Un hard disk (HD) è un dispositivo elettro-meccanico per la conservazione di informazioni sotto forma magnetica, su supporto rotante a forma di piatto su cui agiscono delle testine di lettura/scrittura





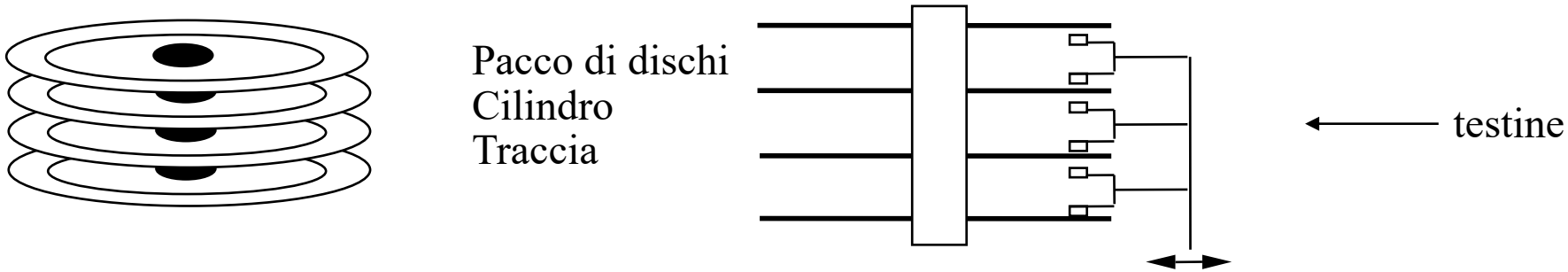
Struttura della superficie di un piatto: A) Traccia B) Settore geometrico C) Settore di una traccia (o anche traccia di un settore) D) Cluster, insieme di settori di tracce contigui

Da Wikipedia

Il meccanismo del disk drive

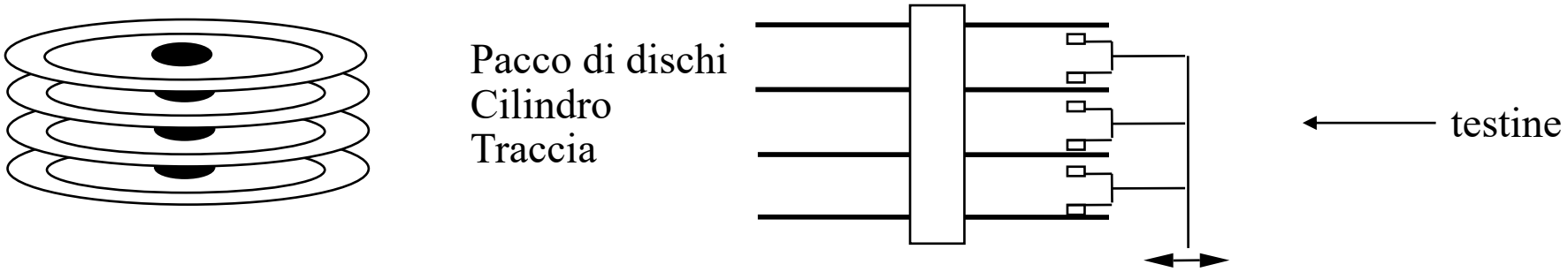
Include organi di registrazione, di posizionamento e di rotazione.

Un'unità a dischi contiene una pila di dischi metallici che ruota a velocità costante ed alcune testine di lettura che si muovono radialmente al disco



MEMORIE A DISCO

- Un'unità a dischi contiene una pila di dischi metallici che ruota a velocità costante ed alcune testine di lettura che si muovono radialmente al disco



- Una traccia è organizzata in settori di dimensione fissa; i settori sono raggruppati logicamente in blocchi, che sono l'unità di trasferimento.
- Trasferire un blocco richiede un tempo di posizionamento delle testine, un tempo di latenza rotazionale e tempo per il trasferimento (trascurabile)
  - IBM 3380 (1980) :2Gb, 16 ms, 8.3 ms, 0.8 ms/2.4Kb
  - IBM Ultrastar 36Z15 (2001): 36GB, 4.2 ms, 2 ms, 0.02 ms/Kb

Pagine

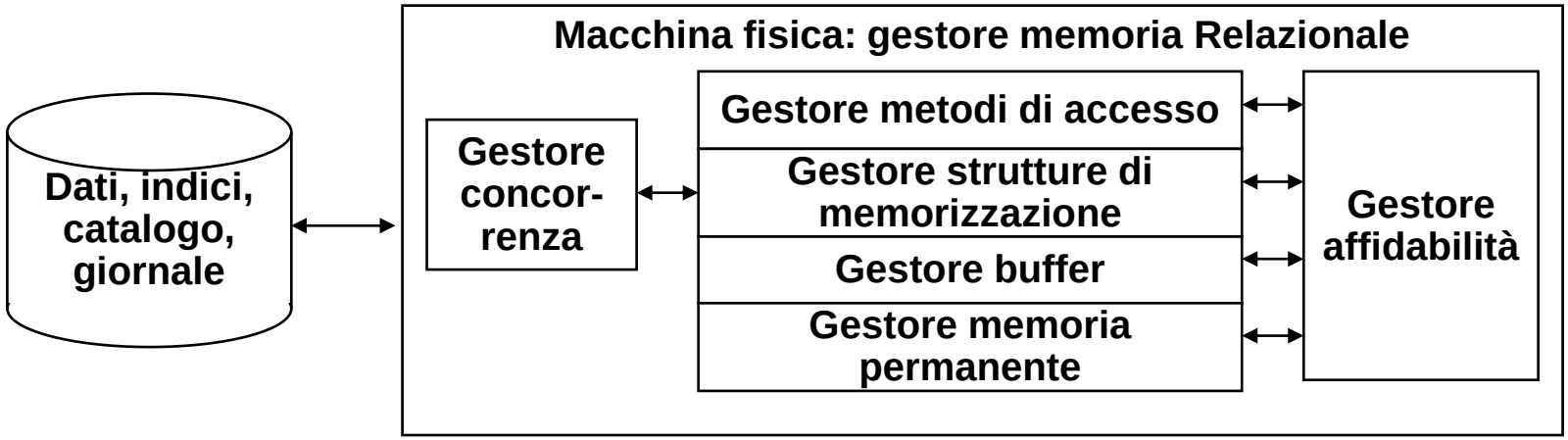
- Un blocco (o **pagina**) è una sequenza contigua di settori su una traccia, e costituisce l'unità di I/O per il trasferimento di dati da/per la memoria principale
- La dimensione tipica di una pagina è di qualche KB (4 -64 KB)
- Pagine piccole comportano un maggior numero di operazioni di I/O
- Pagine grandi tendono ad aumentare la frammentazione interna (pagine parzialmente riempite) e richiedono più spazio in memoria per essere caricate
- Il tempo di trasferimento di una pagina (**Tt**) da disco a memoria centrale dipende dalla dimensione della pagina (**P**) e dal transfer rate (**Tr**)
- Esempio: con un transfer rate di 21.56 MB/sec e P = 4 KB si ha

  Tt= 0.18 ms, con P = 64 KB si ha Tt= 2.9 ms

L'IMPORTANZA DELLA LOCALITÀ

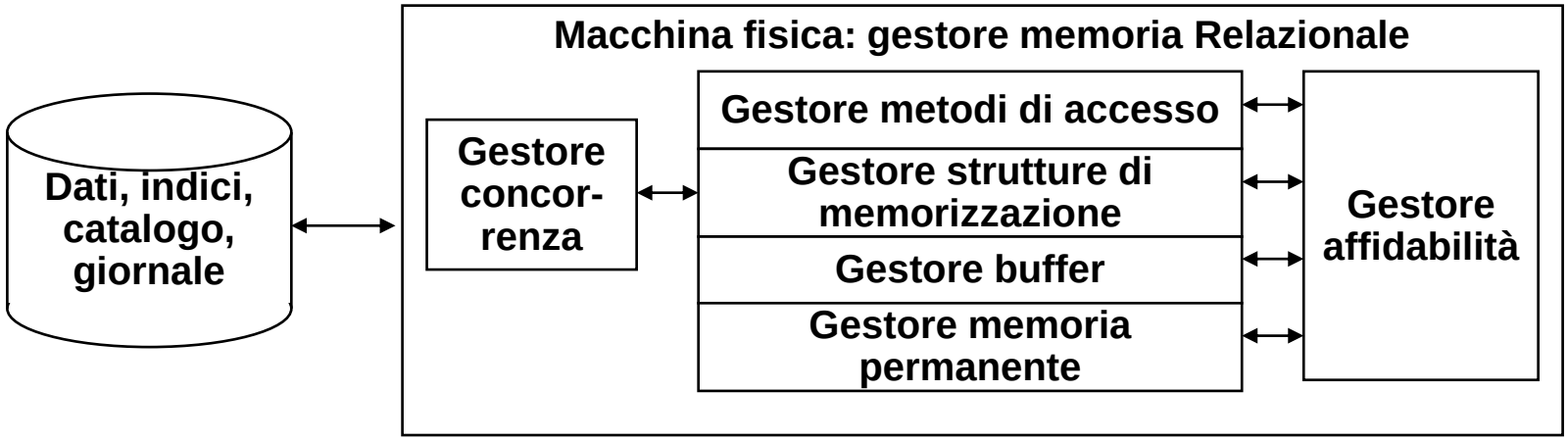
- Per leggere un file da un MB servono (IBM Ultrastar 36Z15):
  - 0,027 secondi se memorizzato in settori consecutivi
  - 0,8 secondi se memorizzato in blocchi da 16 settori ciascuno (8KB) distribuiti a caso (128*(4,2+2+0,16))
  - 12,7 secondi se memorizzato in blocchi da 1 settore ciascuno, distribuiti a caso (2048*(4,2+2+0,01))

GESTORE MEMORIA PERMANENTE E GESTORE DEL BUFFER



- GESTORE MEMORIA PERMANENTE
  - Fornisce un'astrazione della memoria permanente in termini di insiemi di file logici di pagine fisiche di registrazioni (blocchi), nascondendo le caratteristiche dei dischi e del sistema operativo.

GESTORE MEMORIA PERMANENTE E GESTORE DEL BUFFER

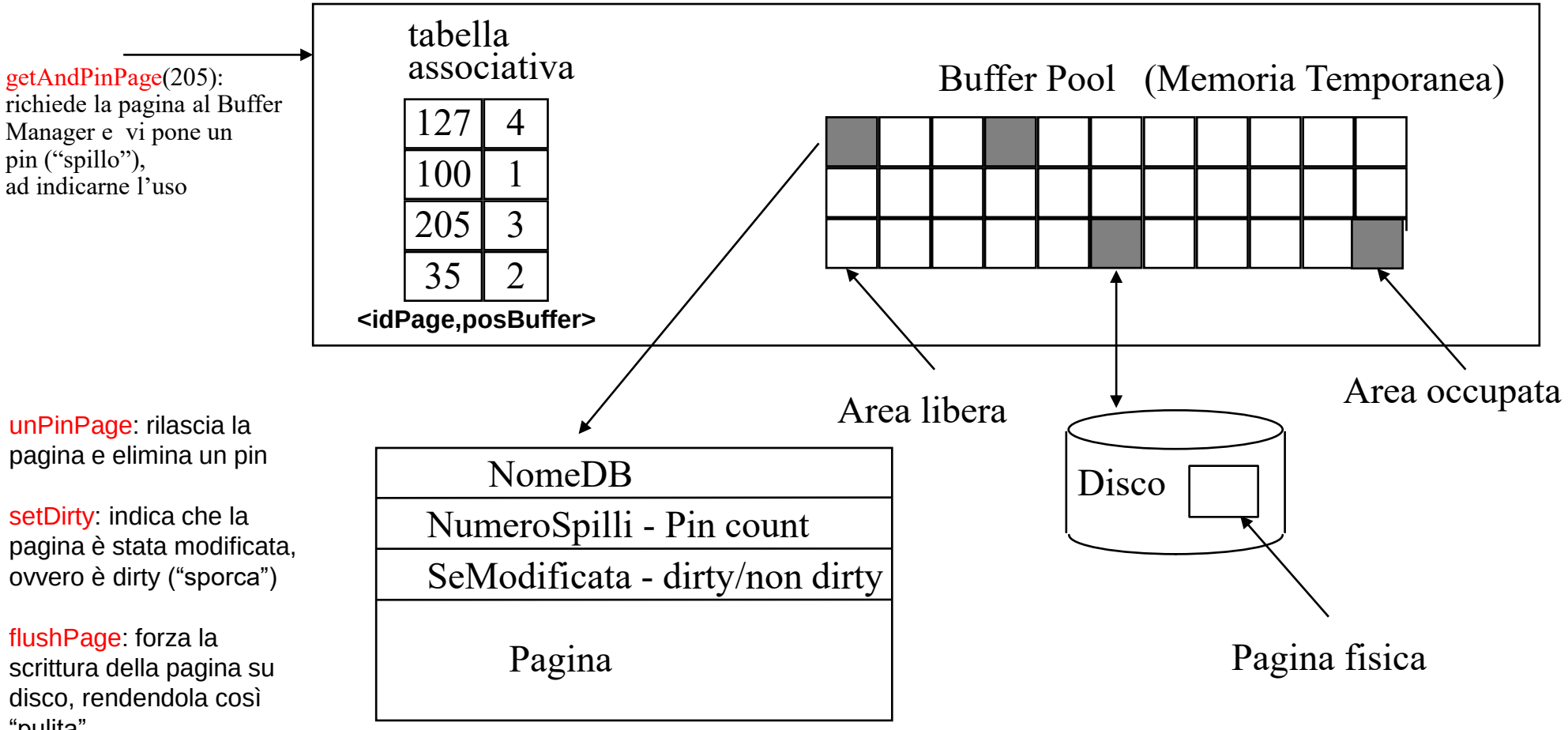


GESTORE DEL BUFFER

- Si preoccupa del trasferimento delle pagine tra la memoria temporanea e la memoria permanente, offrendo agli altri livelli una visione della memoria permanente come un insieme di pagine utilizzabili in memoria temporanea, astraendo da quando esse vengano trasferite dalla memoria permanente al buffer e viceversa

GESTORE DEL BUFFER: AREA DELLE PAGINE

- Pin count: quando una pagina viene richiesta/rilasciata il contatore viene incrementato/decrementato,
- dirty/non dirty: la componente che ha eseguito la modifica chiede al buffer di aggiornare la proprietà dirty/nondirty della pagina modificata



unPinPage: rilascia la pagina e elimina un pin

setDirty: indica che la pagina è stata modificata, ovvero è dirty ("sporca")

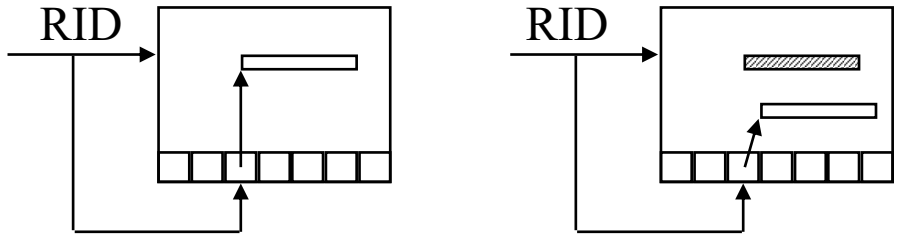
flushPage: forza la scrittura della pagina su disco, rendendola così "pulita"

Politiche di rimpiazzamento

- Nei sistemi operativi una comune politica adottata per decidere quale pagina rimpiazzare è la LRU (Least Recently Used), ovvero si rimpiazza la pagina che da più tempo non è in uso
- Nei DBMS, LRU non è sempre una buona scelta, in quanto per alcune query il "pattern di accesso" ai dati è noto, e può quindi essere utilizzato per operare scelte più accurate, in grado di migliorare anche di molto le prestazioni
- L'hit ratio, ovvero la frazione di richieste che non provocano una operazione di I/O, indica sinteticamente quanto buona è una politica di rimpiazzamento
- Esempio: esistono algoritmi di join che scandiscono N volte le tuple di una relazione. In questo caso la politica migliore sarebbe la MRU (Most Recently Used), ovvero rimpiazzare la pagina usata più di recente!
- … altro motivo per cui i DBMS non usano (tutti) i servizi offerti dai sistemi operativi…

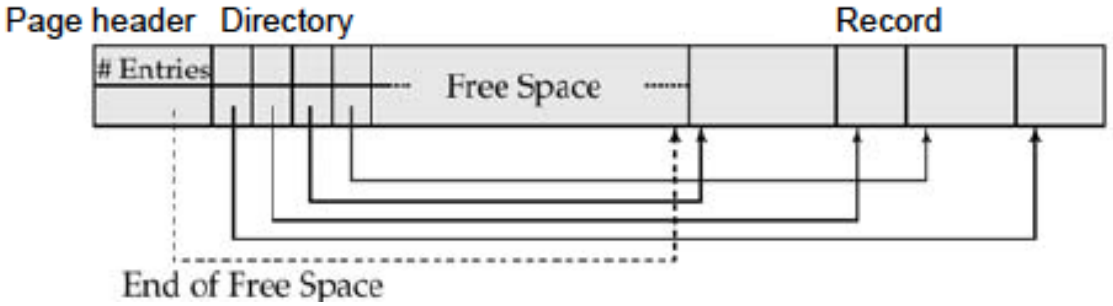
STRUTTURA DI UNA PAGINA

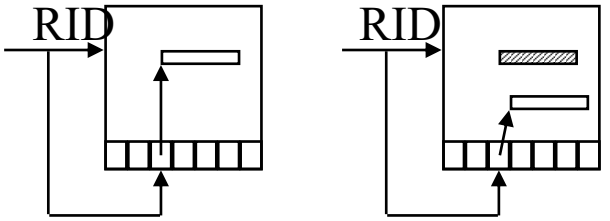
- Struttura fisica: un insieme, di dimensione fissa, di caratteri.
- Struttura logica:
  - informazioni di servizio;
  - un'area che contiene le stringhe che rappresentano i record;
- Il problema dei riferimenti ai record: coppia (PID della pagina, posizione nella pagina) (RID).



Organizzazione a slot delle pagine

- Il formato tipico di una pagina in un DBMS è descritto in figura



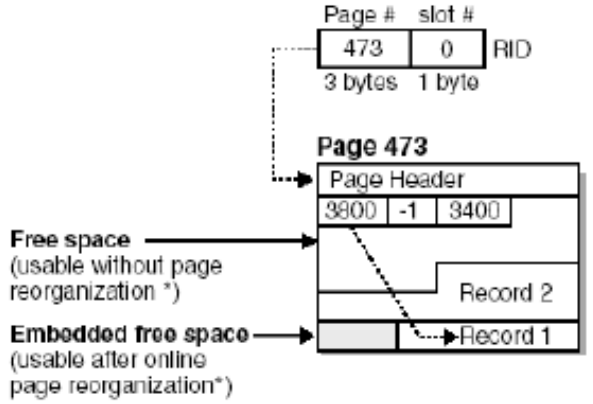


La directory contiene un puntatore per ogni record nella pagina

Con questa soluzione l'identificatore di un record (RID) nel DB è formato da una coppia:

- PID: identificatore della pagina
- Slot: posizione all'interno della pagina

È possibile sia individuare velocemente un record, sia permettere la sua riallocazione nella pagina senza modificare il RID



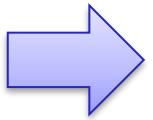
PARTE II

GESTORE STRUTTURE DI MEMORIZZAZIONE

- Tipi di organizzazioni:
  - Seriali o Sequenziali
  - Per chiave
  - Per attributi non chiave
- Parametri che caratterizzano un'organizzazione:
  - Occupazione di memoria
  - Costo delle operazioni di:
    - Ricerca per valore o intervallo
    - Modifica
    - Inserzione
    - Cancellazione

ORGANIZZAZIONI SERIALE E SEQUENZIALE

- Organizzazione seriale (heap file): i dati sono memorizzati in modo disordinato uno dopo l'altro:
  - Semplice e a basso costo di memoria
  - Poco efficiente
  - Va bene per pochi dati
- E' l'organizzazione standard di ogni DBMS.



ORGANIZZAZIONI SERIALE E SEQUENZIALE

- Organizzazione sequenziale: i dati sono ordinati sul valore di uno o più attributi:
  - Ricerche più veloci
  - Nuove inserzioni fanno perdere l'ordinamento

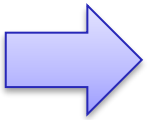
Costo di ricerca

Ricerca binaria su file di dati ordinato richiede $\log_2 b$ accessi a blocco/pagina

Se il file contiene b_i blocchi, la localizzazione di un record richiede:

- ricerca binaria nel file
- accesso al blocco che contiene il record

- Quindi richiede $\log_2 b_i + 1$ accessi a blocco/pagina



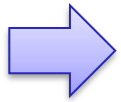
Esempio

- File ordinato con r = 30.000 record
- Dimensioni dei blocchi/pagine su disco B = 1024 byte
- I record hanno dimensione fissa e sono indivisibili, lunghezza R = 100 byte
- Il fattore di blocco/pagine è

  bfr = ⌊ (B/R) ⌋ = ⌊ (1024/100) ⌋ = 10 record per blocco/pagine
- Numero blocchi necessari b = ⌈(r/bfr) ⌉ = ⌈(30000/10) ⌉ = 3000 blocchi
- Una ricerca binaria su file dati richiede circa ⌈ $\log_2 b$ ⌉ = ⌈$\log_2 3000$ ⌉ = 12 accessi ai blocchi/pagine

ORGANIZZAZIONI PER CHIAVE

- **Obiettivo**: noto il valore di una chiave, trovare il record di una tabella con qualche accesso al disco (ideale: 1 accesso).
- **Alternative**:
  - Metodo procedurale (hash) o tabellare (indice)
  - Organizzazione statica o dinamica.



HASH FILE

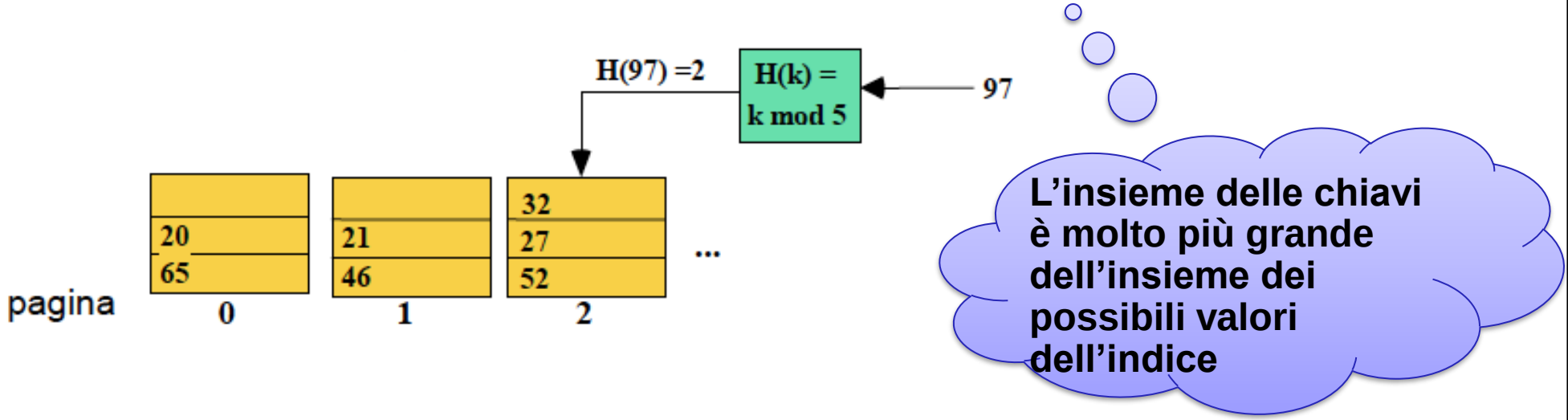
- In un file hash i record vengono allocati in una pagina il cui indirizzo dipende dal valore di chiave del record:

  key → H(key) → page address

- Una comune funzione hash è il resto della divisione intera:

  H(k) = k mod NP

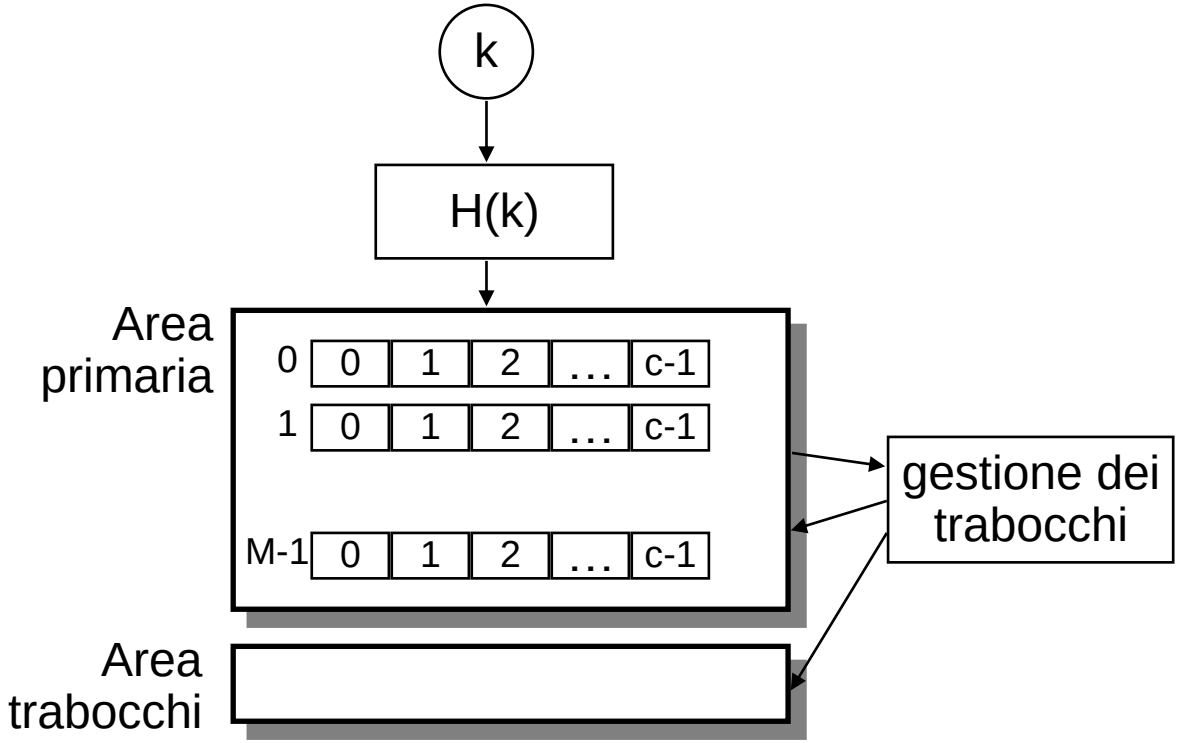
- Si può applicare anche a chiavi alfanumeri che dopo averle convertite



ORGANIZZAZIONE PER CHIAVE: METODO PROCEDURALE STATICO

Parametri di progetto:

- La funzione per la trasformazione della chiave
- Il fattore di caricamento d=N/(M*c)
- La capacità c delle pagine
- Il metodo per la gestione dei trabocchi



Frazione dello spazio fisico disponibile mediamente utilizzata.
Se N è il numero di tuple previsto per il file, M il fattore di pagine e c il fattore di caricamento, il file può prevedere un numero di blocchi B pari al numero intero immediatamente superiore a d.

Strutture Hash

- Le collisioni (overflow) sono di solito gestite con liste linked.

- E' l'organizzazione più efficiente per l'accesso diretto basato su valori della chiave con condizioni di uguaglianza (accesso puntuale)
- Non è efficiente per ricerche basate su intervalli (n. per ricerche basate su altri attributi)
- Funzionano solo con file la cui dimensione non varia molto nel tempo (PROCEDURALE STATICO)