


Physical Data Organization

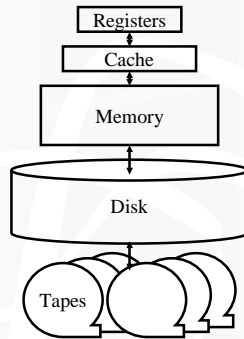
CPS 216
Advanced Database Systems

Outline

- It's all about disks
 - That's why you always draw a database as 
- Record layout
- Block layout

2

Storage hierarchy



3

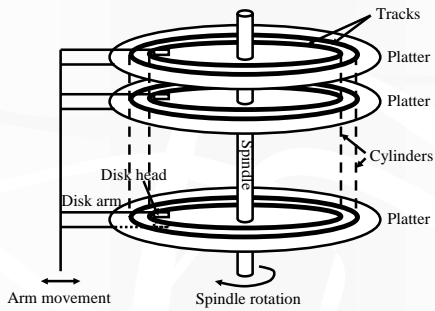
How far away is data?

Location	Cycles
Registers	1
On-chip cache	2
On-board cache	10
Memory	100
Disk	10^6
Tape	10^9

(Source: AlphaSort paper, 1995)

4

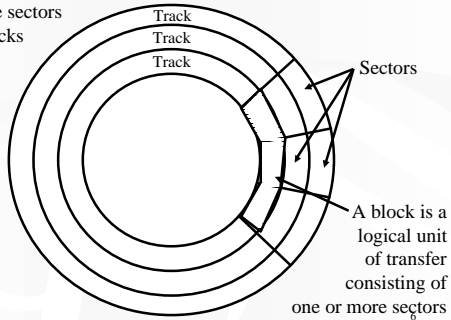
A typical disk



5

Top view

Higher-density sectors on inner tracks and/or more sectors on outer tracks



Disk access time

Sum of:

- Seek time: time for disk heads to move to the correct cylinder
- Rotational delay: time for the desired block to rotate under the disk head
- Transfer time: time to read/write data in the block (= time for disk to rotate over the block)

7

Random disk access

Seek time + rotational delay + transfer time

- Average seek time
 - “Typical” value: 5 ms
- Average rotational delay
 - “Typical” value: 4.2 ms (7200 RPM)

8

Sequential disk access

Seek time + rotational delay + transfer time

- Seek time
 - (assuming data is on the same track)
- Rotational delay
 - (assuming data is in the next block on the track)
- Easily an order of magnitude faster than random disk access!

9

Data layout strategy

Keep related things close together!

- Same sector/block
- Same track
- Same cylinder
- Adjacent cylinder

10

More performance tricks

- Disk scheduling algorithm
 - Example: “elevator” algorithm
- Track buffer
 - Read/write one entire track at a time
- Double buffering
 - While processing the current block in memory, prefetch the next block from disk
- Parallel I/O
 - More disk heads working at the same time

11

Record layout

Record = row in a table

- Variable-format records
 - Rare in DBMS—table schema dictates the format
 - Maybe relevant for semi-structured data such as XML
- Focus on fixed-format records
 - With fixed-length fields only, or
 - With possible variable-length fields

12

Fixed-length fields

- All field lengths and offsets are constant
 - Computed from schema, stored in the system catalog
- Example:** create table Student(SID integer, name CHAR(20), age integer, GPA float)

0	4	24	28	36
142	Bart (padded with '0')		10	2.3
- Watch out for alignment!
- What about NULL?

13

Variable-length fields

- Example:** create table Student (SID integer, name VARCHAR(20), age integer, GPA float, comment VARCHAR(100))
- Approach 1: use field delimiters**

0	4	8	16
142	10	2.3	Bart!0 Weird kid!0
- Approach 2: use an offset array**

0	4	8	16	18	22	32
142	10	2.3			Bart	Weird kid!
- Update is messy if it changes the length of a field

LOB fields

- Example:** create table Student(SID integer, name CHAR(20), age integer, GPA float, picture BLOB(32000))
- Student records get “de-clustered”
 - Bad because most queries do not involve picture
- Decompose (automatically done by DBMS)
 - Student(SID, name, age, GPA)
 - StudentPicture(SID, picture)

15

Block layout

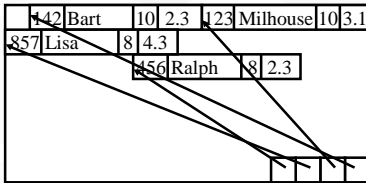
How do you organize records in a block?

- NSM (N-ary Storage Model)
 - Most commercial DBMS
- DSM (Decomposition Storage Model)
- PAX (Partition Attributes Across)
 - Recent work (Ailamaki et al., VLDB 2001)

16

NSM

- Store records from the beginning of each block
- Use a directory at the end of each block
 - To locate records and manage free space
 - Necessary for variable-length records



17

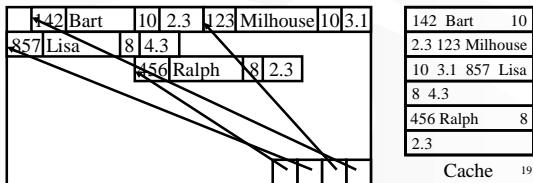
Options

- Reorganize after every update/delete to avoid fragmentation
- What if records are fixed-length?
 - Reorganize after delete
 - Do not reorganize after update

18

Cache behavior of NSM

- Query: `SELECT SID FROM Student WHERE GPA > 2.0;`
- Assumption: cache block size < record size
- Lots of cache misses!
 - Things are not close enough (by memory standard)



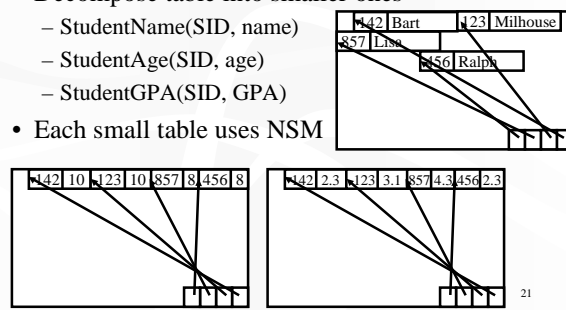
Do cache misses matter in DBMS?

- Yes? Percentage of memory-related stall time due to data cache misses:
 - 90% for OLAP workloads
(lots of large, complex queries; few updates)
 - 50-70% for OLTP workloads
(lots of small queries and updates)
- No? Compared to disk I/Os, memory-related stall time is nothing

20

DSM

- Decompose table into smaller ones
 - StudentName(SID, name)
 - StudentAge(SID, age)
 - StudentGPA(SID, GPA)
- Each small table uses NSM



Pros and cons of DSM

Pros

-
-

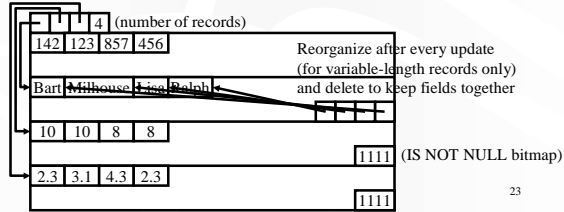
Cons

-
-

22

PAX

- Keep entire rows in a block
 -
- Within a block, cluster columns
 -



PAX versus NSM

- Space requirement
 - Roughly the same
- Cache performance
 - PAX incurs 75% less data cache misses than NSM
- Overall performance
 - For OLAP, PAX is 11-48% faster
 - For OLTP
 - Updates: PAX is 10%-16% faster (assuming NSM reorganizes as well)
 - Queries (typically very selective): I/O still dominates? ²⁴

