

CS 245: Database System Principles

Notes 03: Disk Organization

Hector Garcia-Molina

CS 245

Notes 3

1

Topics for today

- How to lay out data on disk
- How to move it to memory

CS 245

Notes 3

2

What are the data items we want to store?

- a salary
- a name
- a date
- a picture

⇒ What we have available: Bytes



← 8 →
bits

CS 245

Notes 3

3

To represent:

- Integer (short): 2 bytes
e.g., 35 is

00000000 00100011

- Real, floating point
 n bits for mantissa, m for exponent...

CS 245

Notes 3

4

To represent:

- Characters
→ various coding schemes suggested,
most popular is ascii

Example:

A: 1000001
a: 1100001
5: 0110101
LF: 0001010

CS 245

Notes 3

5

To represent:

- Boolean
e.g., TRUE 1111 1111
FALSE 0000 0000
- Application specific
e.g., RED → 1 GREEN → 3
BLUE → 2 YELLOW → 4 ...

⇒ Can we use less than 1 byte/code?

Yes, but only if desperate...

CS 245

Notes 3

6

To represent:

- Dates
e.g.: - Integer, # days since Jan 1, 1900
- 8 characters, YYYYMMDD
- 7 characters, YYYYDDD
(not YYMMDD! Why?)
- Time
e.g. - Integer, seconds since midnight
- characters, HHMMSSFF

CS 245

Notes 3

7

To represent:

- String of characters
 - Null terminated
e.g.,

c	a	t	⊗		
---	---	---	---	--	--
 - Length given
e.g.,

3	c	a	t	⊗	
---	---	---	---	---	--
 - Fixed length

CS 245

Notes 3

8

To represent:

- Bag of bits



CS 245

Notes 3

9

Key Point

- Fixed length items
- Variable length items
 - usually length given at beginning

CS 245

Notes 3

10

Also

- Type of an item: Tells us how to interpret
(plus size if fixed)

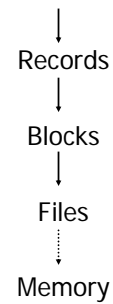
CS 245

Notes 3

11

Overview

Data Items



CS 245

Notes 3

12

Record - Collection of related data items (called FIELDS)

E.g.: Employee record:
name field,
salary field,
date-of-hire field, ...

CS 245

Notes 3

13

Types of records:

- Main choices:
 - FIXED vs VARIABLE FORMAT
 - FIXED vs VARIABLE LENGTH

CS 245

Notes 3

14

Fixed format

A SCHEMA (not record) contains following information

- # fields
- type of each field
- order in record
- meaning of each field

CS 245

Notes 3

15

Example: fixed format and length

Employee record

- (1) E#, 2 byte integer
- (2) E.name, 10 char.
- (3) Dept, 2 byte code

} Schema

55 s m i t h 02

83 j o n e s 01

} Records

CS 245

Notes 3

16

Variable format

- Record itself contains format "Self Describing"

CS 245

Notes 3

17

Example: variable format and length

2 | 5 | I | 46 | 4 | S | 4 | F O R D

Fields
↑
Code identifying field as E#
↑
Integer type
↑
Code for Ename
↑
String type
↑
Length of str.

Field name codes could also be strings, i.e. TAGS

CS 245

Notes 3

18

Variable format useful for:

- “sparse” records
- repeating fields
- evolving formats

.....→ But may waste space...

- EXAMPLE: var format record with repeating fields
Employee → one or more → children

3	E_name: Fred	Child: Sally	Child: Tom
---	--------------	--------------	------------

Note: Repeating fields does not imply
- variable format, nor
- variable size

John	Sailing	Chess	--
------	---------	-------	----

- Key is to allocate maximum number of repeating fields (if not used → null)

☆ Many variants between fixed - variable format:

Example: Include record type in record

5	27
---	----	-------

↑ record type tells me what to expect (i.e. points to schema)
← record length

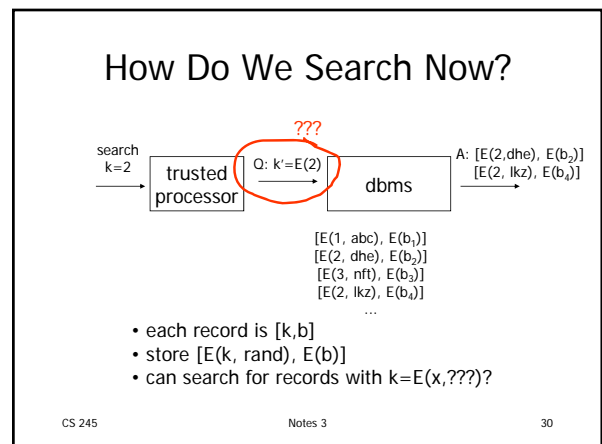
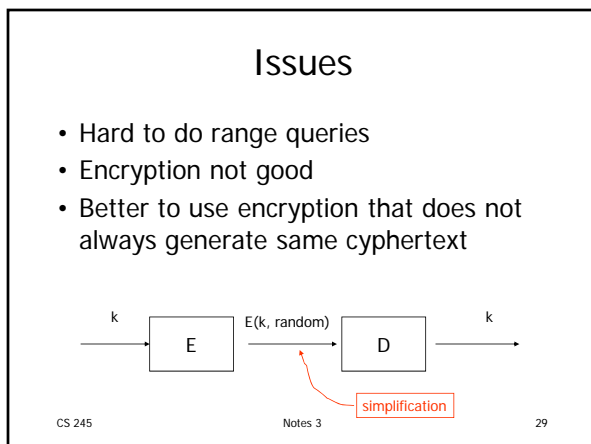
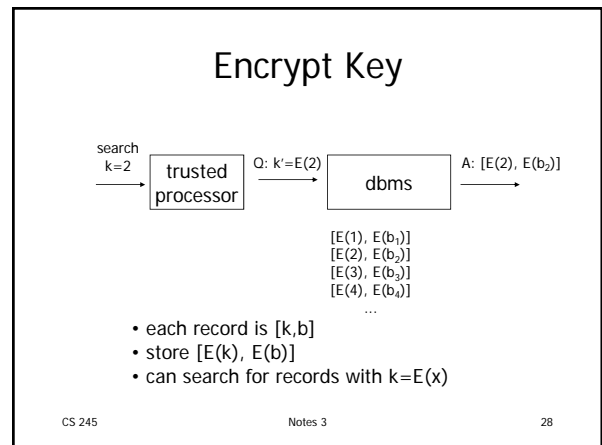
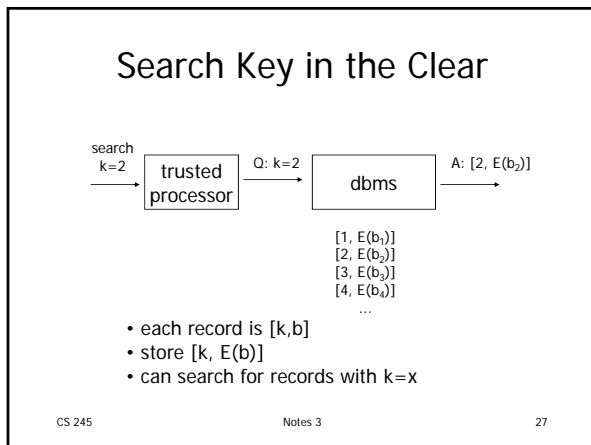
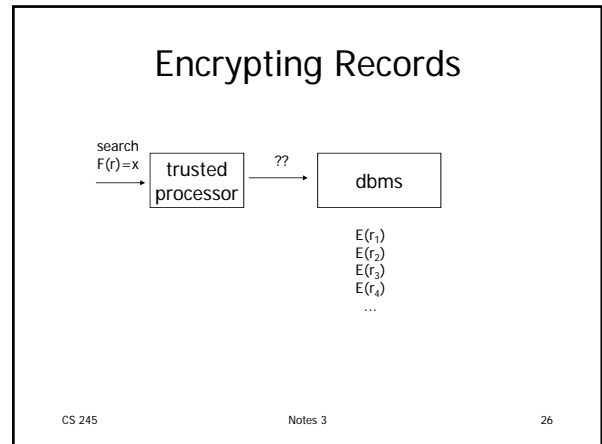
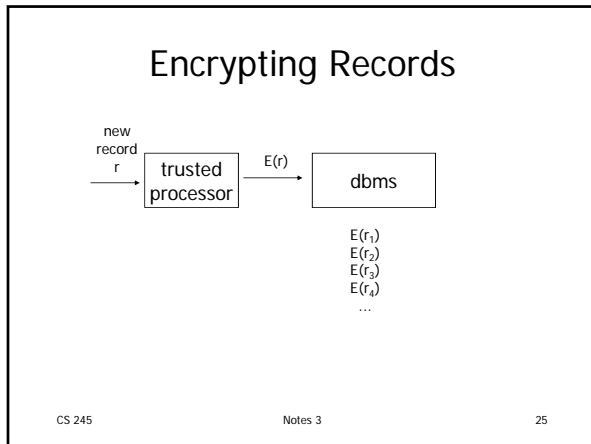
Record header - data at beginning that describes record

May contain:

- record type
- record length
- time stamp
- other stuff ...

Other interesting issues:

- Compression
 - within record - e.g. code selection
 - collection of records - e.g. find common patterns
- Encryption



Solution?

- Develop new decryption function:
 $D(f(k_1), E(k_2, \text{rand}))$ is true if $k_1=k_2$

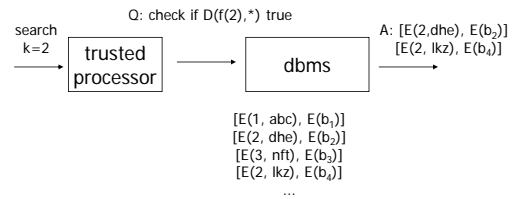
CS 245

Notes 3

31

Solution?

- Develop new decryption function:
 $D(f(k_1), E(k_2, \text{rand}))$ is true if $k_1=k_2$



CS 245

Notes 3

32

Issues?

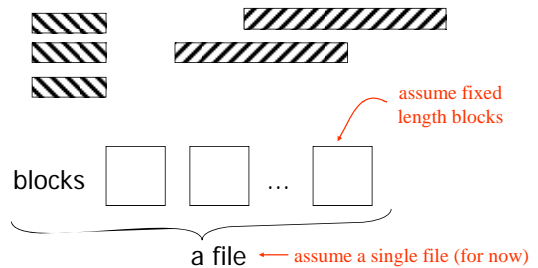
- Cannot do non-equality predicates
- Hard to build indexes

CS 245

Notes 3

33

Next: placing records into blocks



CS 245

Notes 3

34

Options for storing records in blocks:

- (1) separating records
- (2) spanned vs. unspanned
- (4) sequencing
- (5) indirection

CS 245

Notes 3

35

(1) Separating records



- no need to separate - fixed size recs.
- special marker
- give record lengths (or offsets)
 - within each record
 - in block header

CS 245

Notes 3

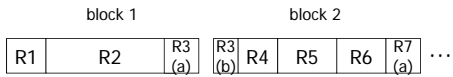
36

(2) Spanned vs. Unspanned

- Unspanned: records must be within one block



- Spanned

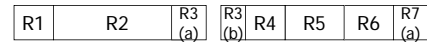


CS 245

Notes 3

37

With spanned records:



need indication
of partial record
"pointer" to rest

need indication
of continuation
(+ from where?)

CS 245

Notes 3

38

Spanned vs. unspanned:

- Unspanned is much simpler, but may waste space...
- Spanned essential if
 record size > block size

CS 245

Notes 3

39

(3) Sequencing

- Ordering records in file (and block) by some key value

Sequential file (\Rightarrow sequenced)

CS 245

Notes 3

40

Why sequencing?

Typically to make it possible to efficiently read records in order
(e.g., to do a merge-join — discussed later)

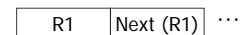
CS 245

Notes 3

41

Sequencing Options

- (a) Next record physically contiguous



- (b) Linked



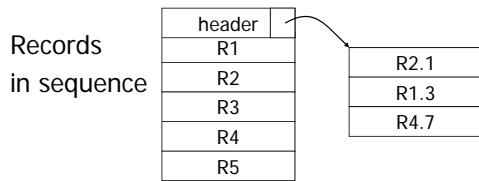
CS 245

Notes 3

42

Sequencing Options

(c) Overflow area



CS 245

Notes 3

43

(4) Indirection

- How does one refer to records?



Many options:

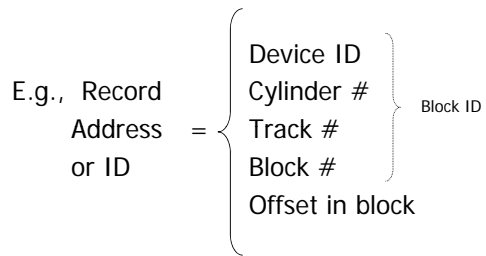
Physical \longleftrightarrow Indirect

CS 245

Notes 3

44

☆ Purely Physical



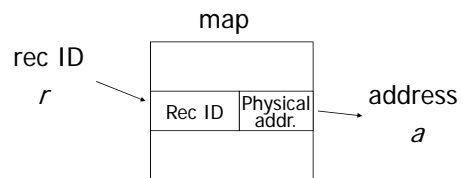
CS 245

Notes 3

45

☆ Fully Indirect

E.g., Record ID is arbitrary bit string



CS 245

Notes 3

46

Tradeoff

Flexibility \longleftrightarrow Cost
to move records of indirection
(for deletions, insertions)

CS 245

Notes 3

47

Physical \longleftrightarrow Indirect

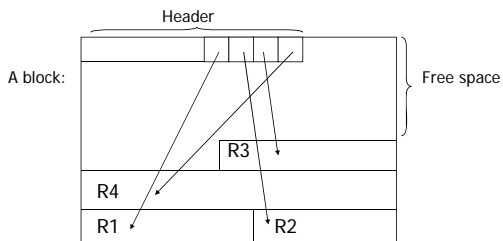
↑
Many options
in between ...

CS 245

Notes 3

48

Example: Indirection in block



CS 245

Notes 3

49

Block header - data at beginning that describes block

May contain:

- File ID (or RELATION or DB ID)
- This block ID
- Record directory
- Pointer to free space
- Type of block (e.g. contains recs type 4; is overflow, ...)
- Pointer to other blocks "like it"
- Timestamp ...

CS 245

Notes 3

50

Options for storing records in blocks:

- (1) separating records
- (2) spanned vs. unspanned
- (4) sequencing
- (5) indirection

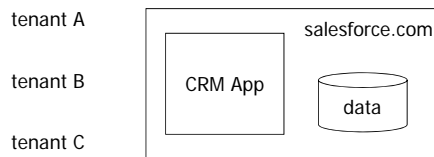
CS 245

Notes 3

51

Case Study: salesforce.com

- salesforce.com provides CRM services
- salesforce customers are *tenants*
- Tenants run apps and DBMS as service



CS 245

Notes 3

52

Options for Hosting

- Separate DBMS per tenant
- One DBMS, separate tables per tenant
- One DBMS, shared tables

CS 245

Notes 3

53

salesforce.com solution

customer	tenant	A	B	C
	1	a1	b1	c1
	1	a2	b2	c2
	2	a3	b3	c2
	2	a1	b1	c1

← fixed schema for all tenants

cust-other	tenant	A	f1	v1	f2	v2	...
	1	a1	D	d1	E	e1	
	1	a2	E	e2	F	f2	
	2	a1	G	g1			
	3	a4	D	d1			

← var schema for all tenants

CS 245

Notes 3

54

Other Topics

- (1) Insertion/Deletion
- (2) Buffer Management
- (3) Comparison of Schemes

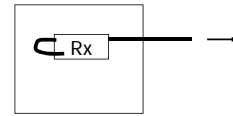
CS 245

Notes 3

55

Deletion

Block



CS 245

Notes 3

56

Options:

- (a) Immediately reclaim space
- (b) Mark deleted
 - May need chain of deleted records (for re-use)
 - Need a way to mark:
 - special characters
 - delete field
 - in map

CS 245

Notes 3

57

☆ As usual, many tradeoffs...

- How expensive is to move valid record to free space for immediate reclaim?
- How much space is wasted?
 - e.g., deleted records, delete fields, free space chains,...

CS 245

Notes 3

58

Concern with deletions

Dangling pointers



CS 245

Notes 3

59

Solution #1: Do not worry

CS 245

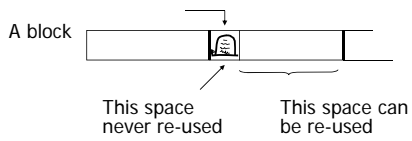
Notes 3

60

Solution #2: Tombstones

E.g., Leave "MARK" in map or old location

- Physical IDs



CS 245

Notes 3

61

Solution #2: Tombstones

E.g., Leave "MARK" in map or old location

- Logical IDs

map

ID	LOC
7788	

Never reuse ID 7788 nor space in map...

CS 245

Notes 3

62

Insert

Easy case: records not in sequence

- Insert new record at end of file or in deleted slot
- If records are variable size, not as easy...

CS 245

Notes 3

63

Insert

Hard case: records in sequence

- If free space "close by", not too bad...
- Or use overflow idea...

CS 245

Notes 3

64

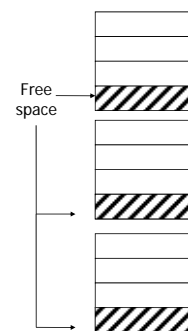
Interesting problems:

- How much free space to leave in each block, track, cylinder?
- How often do I reorganize file + overflow?

CS 245

Notes 3

65



CS 245

Notes 3

66

Buffer Management

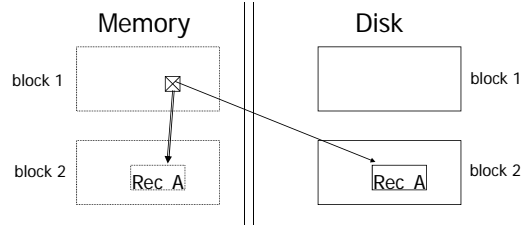
- DB features needed
 - Why LRU may be bad
 - Pinned blocks
 - Forced output
 - Double buffering
 - Swizzling
- Read
Textbook!
in Notes02

CS 245

Notes 3

67

Swizzling



CS 245

Notes 3

68

Row vs Column Store

- So far we assumed that fields of a record are stored contiguously (row store)...
- Another option is to store like fields together (column store)

CS 245

Notes 3

69

Row Store

- Example: Order consists of
 - id, cust, prod, store, price, date, qty

id1	cust1	prod1	store1	price1	date1	qty1
id2	cust2	prod2	store2	price2	date2	qty2
id3	cust3	prod3	store3	price3	date3	qty3

CS 245

Notes 3

70

Column Store

- Example: Order consists of
 - id, cust, prod, store, price, date, qty

id1	cust1	id1	prod1	id1	price1	qty1
id2	cust2	id2	prod2	id2	price2	qty2
id3	cust3	id3	prod3	id3	price3	qty3
id4	cust4	id4	prod4	id4	price4	qty4
...

ids may or may not be stored explicitly

CS 245

Notes 3

71

Row vs Column Store

- Advantages of Column Store
 - more compact storage (fields need not start at byte boundaries)
 - efficient reads on data mining operations
- Advantages of Row Store
 - writes (multiple fields of one record) more efficient
 - efficient reads for record access (OLTP)

CS 245

Notes 3

72

Interesting paper to read:

- Mike Stonebreaker, Elizabeth (Betty) O'Neil, Pat O'Neil, Xuedong Chen, et al. "C-Store: A Column-oriented DBMS," Presented at the 31st VLDB Conference, September 2005.
- http://www.cs.umb.edu/%7Eponeil/vldb05_cstore.pdf

CS 245

Notes 3

73

Comparison

- There are 10,000,000 ways to organize my data on disk...

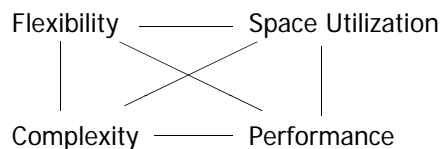
Which is right for me?

CS 245

Notes 3

74

Issues:



CS 245

Notes 3

75

☆ To evaluate a given strategy, compute following parameters:

- > space used for expected data
- > expected time to
 - fetch record given key
 - fetch record with next key
 - insert record
 - append record
 - delete record
 - update record
 - read all file
 - reorganize file

CS 245

Notes 3

76

Example

How would you design Megatron 3000 storage system? (for a relational DB, low end)

- Variable length records?
- Spanned?
- What data types?
- Fixed format?
- Record IDs ?
- Sequencing?
- How to handle deletions?

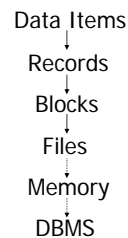
CS 245

Notes 3

77

Summary

- How to lay out data on disk



CS 245

Notes 3

78

Next

How to find a record quickly,
given a key