COSC344
Database Theory and Applications

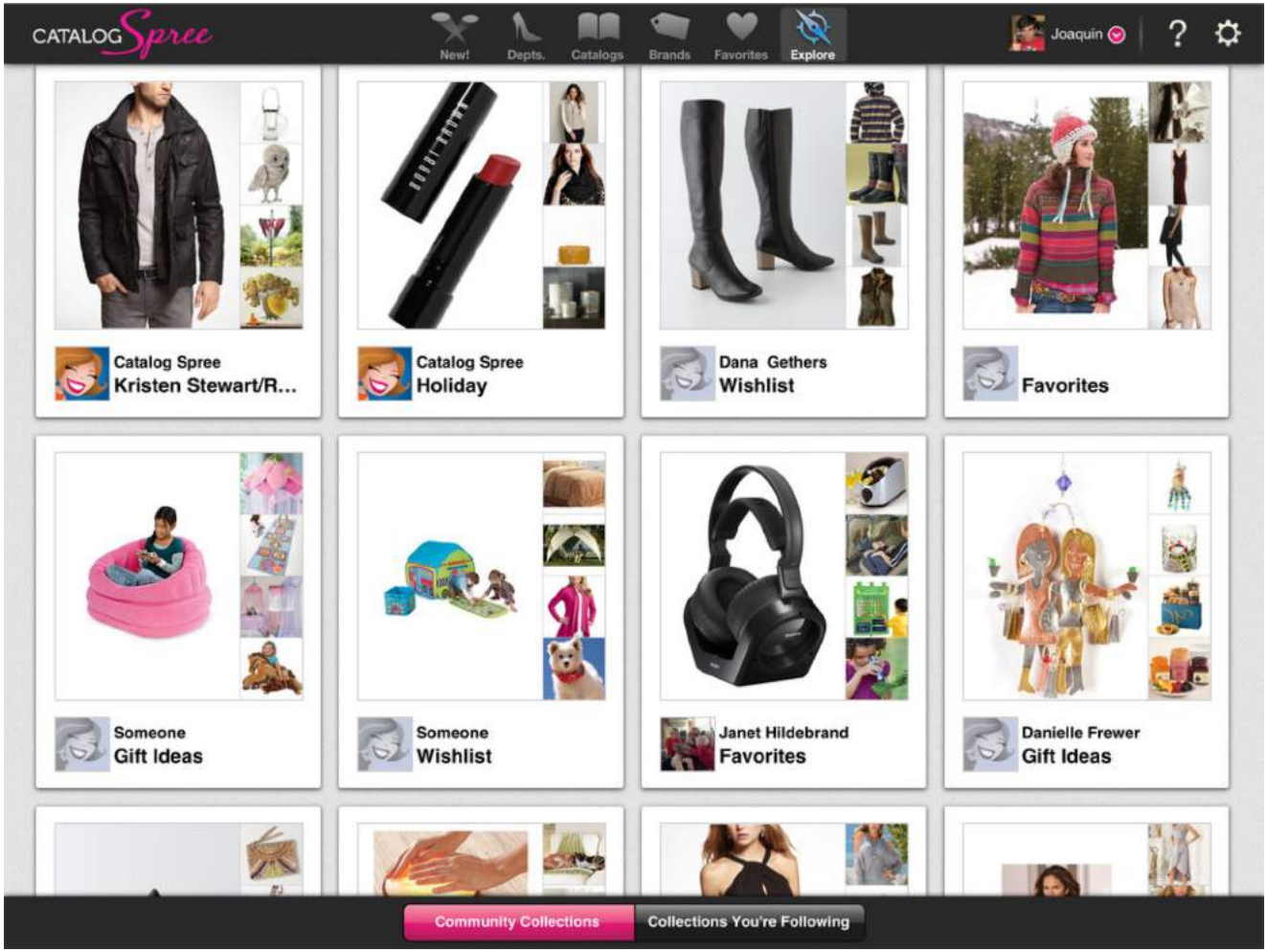
Lecture 17
Database files and storage

Overview

- This Lecture
  - System Catalogs
  - Database files and storage
    - Formatting and storing records of database files on disk
    - 3 primary methods for organizing records of a file on disk
      - unordered records
      - ordered records
      - hashed records
  - Source: Chapter 16
- Next Lecture
  - Database Indexing
  - Source: Chapter 17

Shopping Catalog (“mini-shop”)

“look at the catalog before going to the real shop”

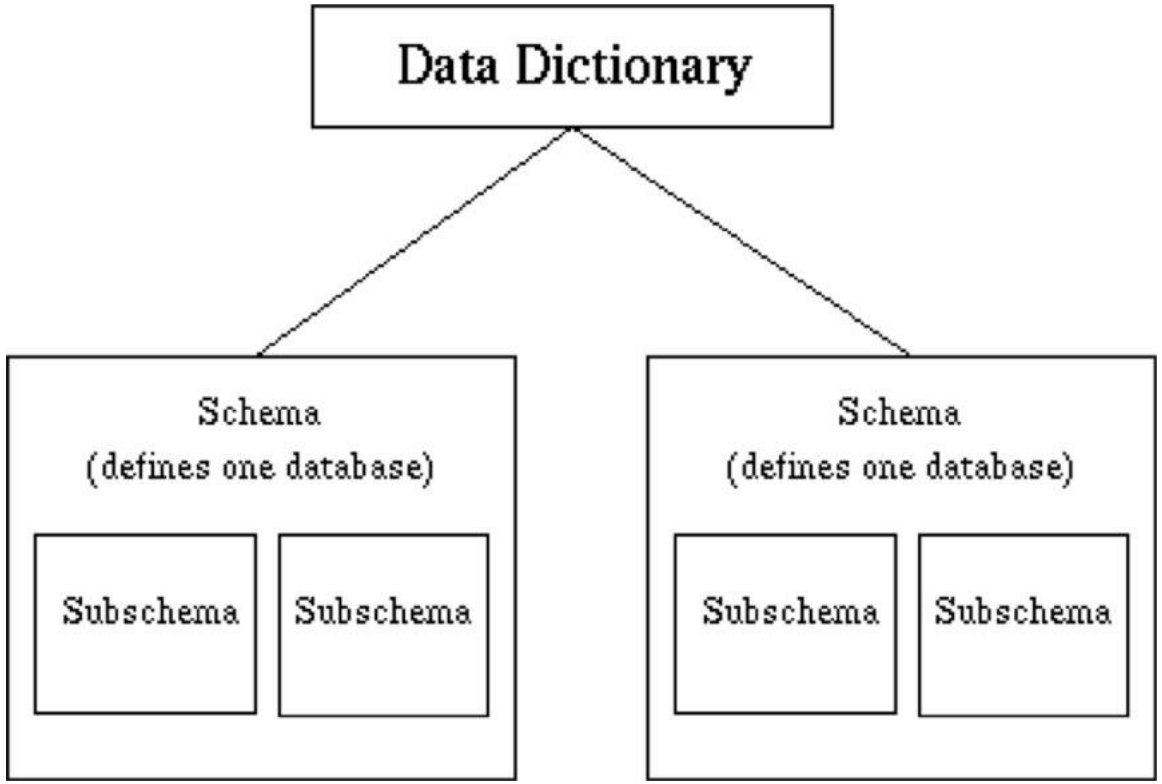


Dictionary

"provide descriptions, references"

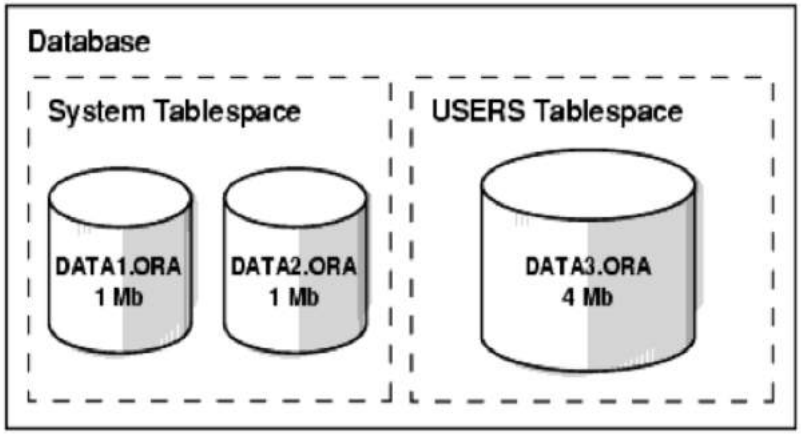
Data Dictionary (System Catalog)

"provide descriptions, references to everything in the database"



System Catalog

- It is a “minidatabase” itself.
- It stores metadata that describes the schemas.
- System Catalogs for relational DBMS provide descriptions
  - Tables
  - Attributes
  - Data types
  - Constraints
  - Owner
  - Primary and foreign keys
  - Views
  - . . .



System Catalogs in Oracle

https://docs.oracle.com/cd/B10501_01/nav/catalog_views.htm

- Called data dictionary
- Access is allowed through views
- Categories (used as a prefix, search “oracle catalog”)
  - USER: user’s view (what is in the user’s schema)
  - ALL: expanded user’s view (what the user can access)
  - DBA: database administrator’s view (what is in all users’ schemas)
- Examples:
  - ALL_CATALOG: the owner, table_name, and table_type of all tables.
  - _TAB_COLUMNS: information about tables and their attributes
  - _TABLES: information about tables
  - _VIEWS: information about views

Oracle System Catalog Examples

```
SQL> DESCRIBE all_catalog;

 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 OWNER                                     NOT NULL VARCHAR2(30)
 TABLE_NAME                                NOT NULL VARCHAR2(30)
 TABLE_TYPE                                         VARCHAR2(11)
```

Oracle System Catalog Examples

```
SQL> SELECT * FROM all_catalog WHERE owner='YAWEN'; Note names in upper case

OWNER                  TABLE_NAME                TABLE_TYPE
-------------------------------------------------------
YAWEN                  CUSTOMERS                   TABLE
YAWEN                  D1                          TABLE
YAWEN                  DATEANDTIME                 TABLE
YAWEN                  DEPARTMENT                  TABLE
YAWEN                  DEPENDENT                   TABLE
YAWEN                  DEPT                        TABLE
YAWEN                  DEPT_INFO                   VIEW
YAWEN                  DEPT_LOCATIONS              TABLE
YAWEN                  E1                          TABLE
YAWEN                  EMPLOYEE                    TABLE
YAWEN                  EMP_DEP                     TABLE
YAWEN                  ORDERS                      TABLE
YAWEN                  PROJECT                     TABLE
YAWEN                  SALESPEOPLE                 TABLE
YAWEN                  TEST_X                      TABLE
YAWEN                  WORKS_ON                    TABLE
YAWEN                  XYZ                         TABLE
```

Oracle System Catalog Examples

```
SQL> SELECT table_name, column_name
     FROM user_tab_columns
     WHERE table_name = 'EMPLOYEE';

TABLE_NAME                     COLUMN_NAME
------------------------------------------------
EMPLOYEE                       FNAME
EMPLOYEE                       MINIT
EMPLOYEE                       LNAME
EMPLOYEE                       SSN
EMPLOYEE                       BDATE
EMPLOYEE                       ADDRESS
EMPLOYEE                       SEX
EMPLOYEE                       SALARY
EMPLOYEE                       SUPERSSN
EMPLOYEE                       DNO
```

Oracle System Catalog Examples

```
SQL> SELECT num_rows, blocks, empty_blocks
     FROM user_tables
     Where table_name = 'EMPLOYEE';

 NUM_ROWS     BLOCKS  EMPTY_BLOCKS
---------- ---------- ------------
    8           5           0
```

```
SQL> SELECT view_name, text
     FROM user_views;

VIEW_NAME
------------------------------
TEXT
--------------------------------------------------
DEPT_INFO
select dname, count(*), sum(salary)
from department, employee
where dnumber=dno
```

DBMS Component Modules

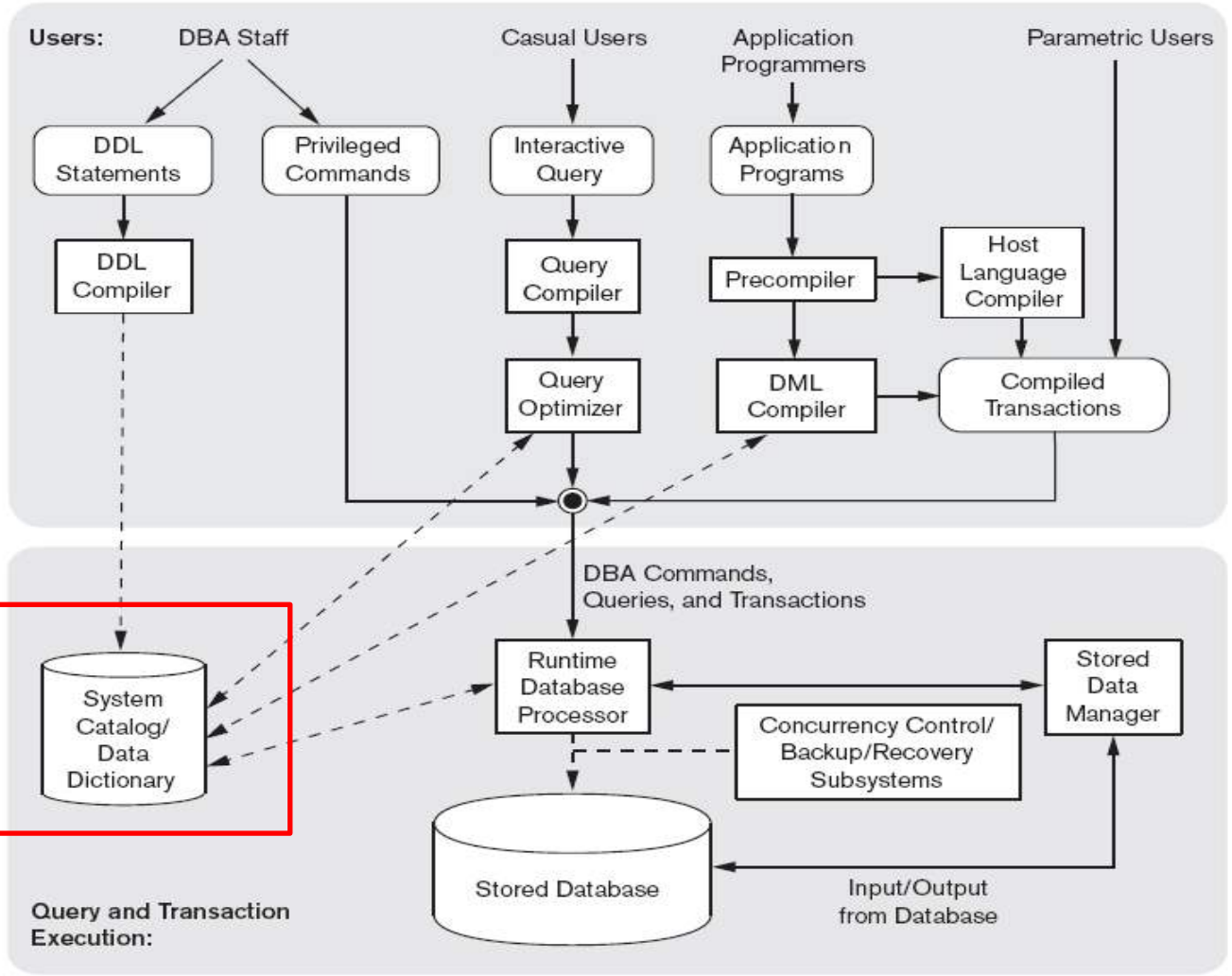


Figure 2.3
Component modules of a DBMS and their interactions.

DBMS Component Modules

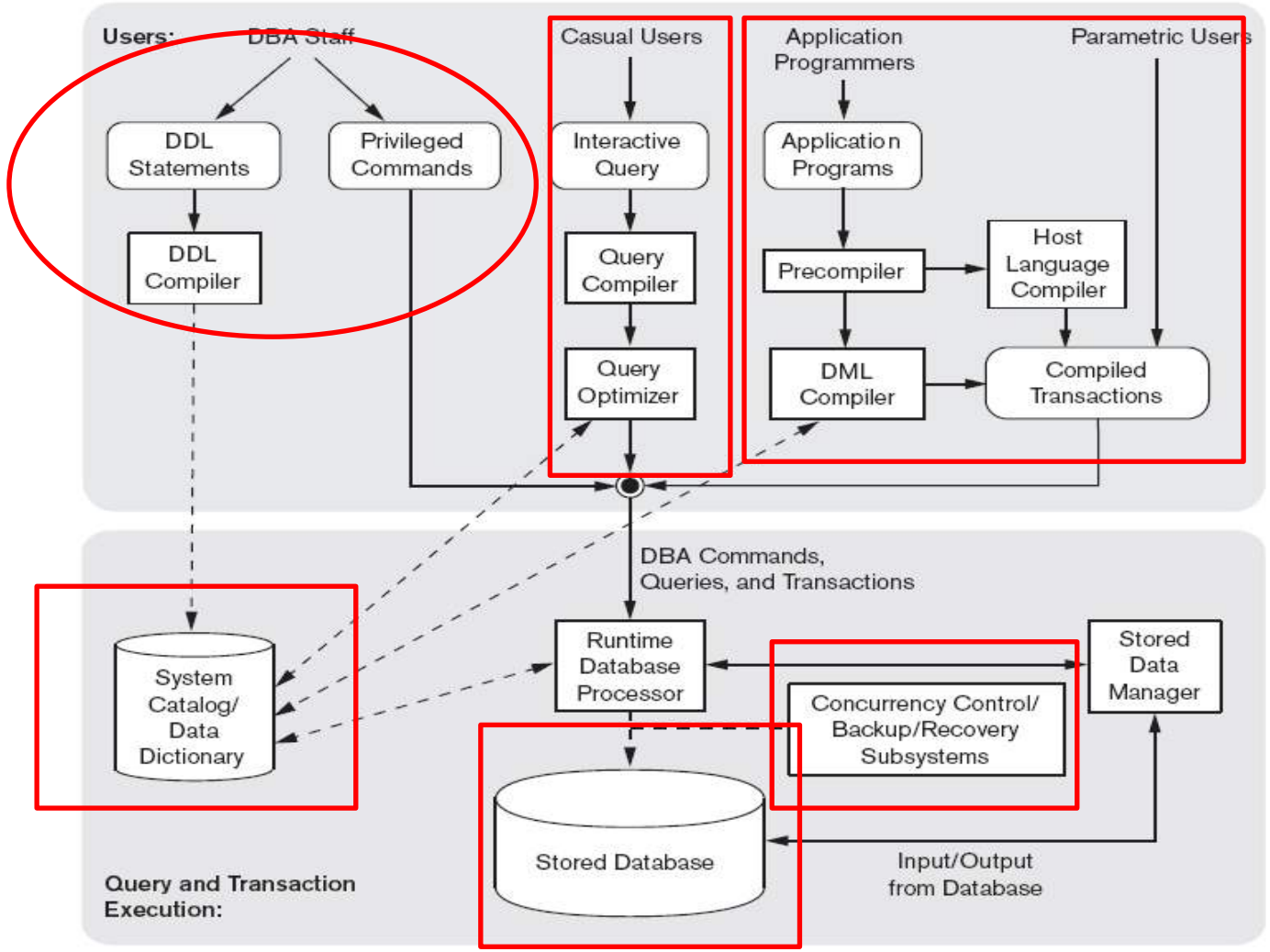


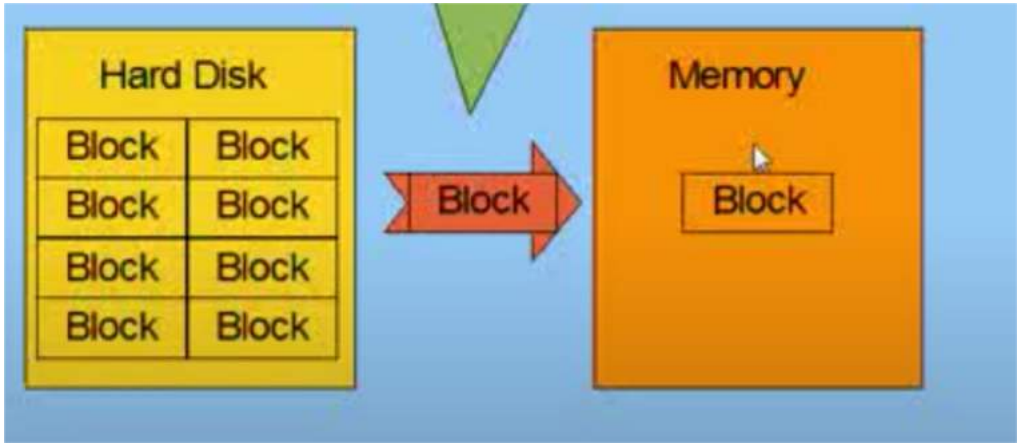
Figure 2.3
Component modules of a DBMS and their interactions.

Storage Hierarchy

- Primary storage
  - Data can be directly processed by CPU, e.g. cache memory, computer main memory
  - Fast access to data
  - Limited storage capacity
- Secondary storage
  - e.g. Hard-disk
  - Large capacity
  - Slower access than primary storage
- Tertiary storage
  - e.g. CDs, DVDs, tapes
  - Slower access than second storage

Data blocks

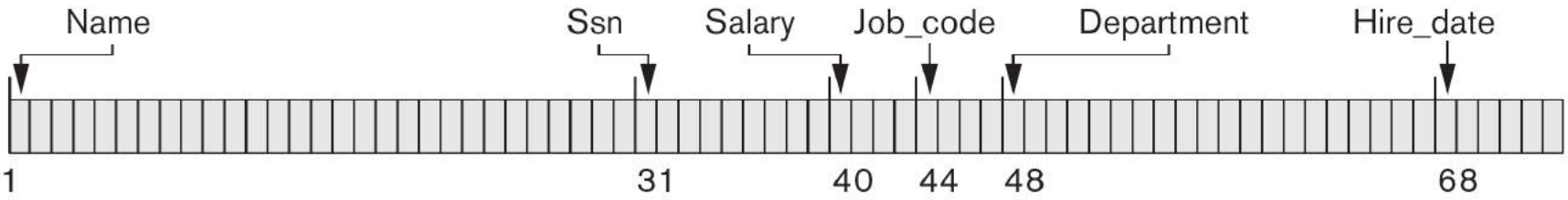
- Data in secondary or tertiary storage needs to be transferred to main memory, and then processed by CPU.
- Data is stored in data blocks. One data block corresponds to a specific number of bytes on disk. (e.g. 8k)
- Block is the unit of data transfer between disk and memory
- Database systems normally only read/write a block or a few blocks at a time
- Transfer time is very large compared to memory access time



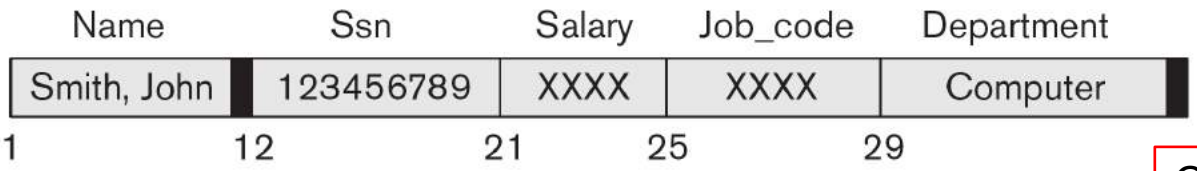
Formatting File Records on Disk

- Records
  - A record is usually a tuple or row for relational databases
- Files
  - A sequence of records
  - In many case all records in a file are of the same record type
- Fixed-length records
  - Every record has exactly the same size
- Variable-length records
  - Different records have different sizes (fields with varying size, optional fields and etc.)

(a)



(b)



Separator Characters

Characters that do not appear in any field value

(c)

Separator Characters

= Separates field name from field value

Separates fields

Terminates record

Figure 17.5
Three record storage formats. (a) A fixed-length record with six fields and size of 71 bytes. (b) A record with two variable-length fields and three fixed-length fields. (c) A variable-field record with three types of separator characters.

Unspanned vs. Spanned Records

- Records must be allocated to disk blocks
- Unspanned record
  - Records are not allowed to cross block boundaries
- Spanned records
  - A record can span more than one block

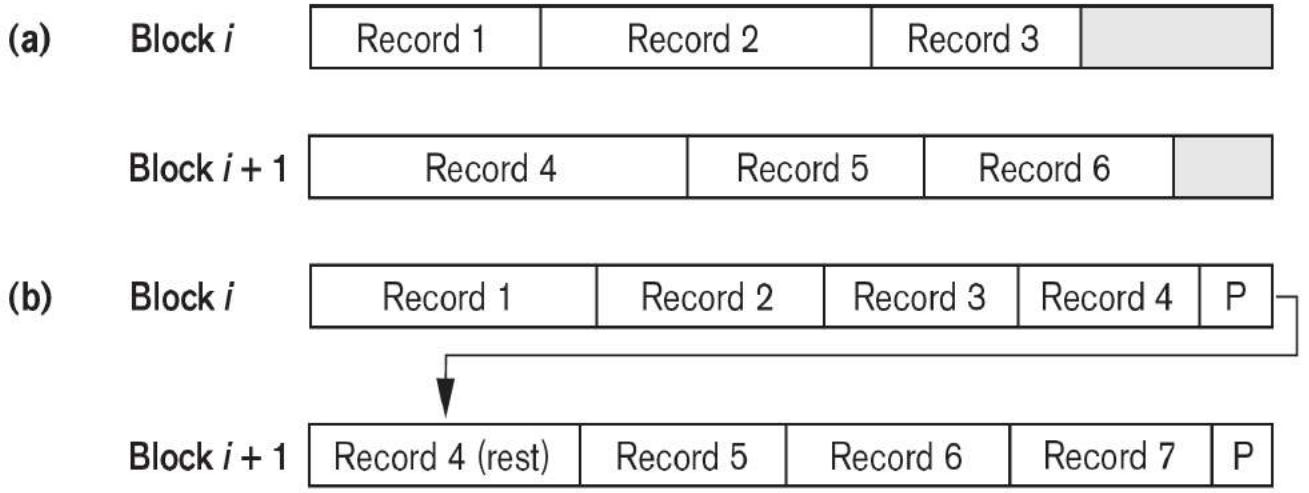


Figure 17.6
Types of record organization.
(a) Unspanned.
(b) Spanned.

Record Blocking

- How many records can be allocated in one disk block?
- Assume
  - block size B
  - Record size R<=B
- Fixed-length records
  - Blocking factor (bfr)

    bfr = $\lfloor B/R \rfloor$ $\lfloor x \rfloor$ floor function (rounds down/truncates)
  - Unused space using unspanned organization

    B-(bfr*R) bytes
- Variable-length records
  - bfr represents the average number of records per block
  - Number of blocks for r records using spanned organization

    b = $\lceil r/bfr \rceil$ blocks $\lceil x \rceil$ ceiling function

Block Allocation

How to allocate blocks in the disk?

- As needed
  - Results in data being scattered overall the disk
- Contiguous allocation
  - File blocks are allocated to consecutive disk blocks
  - Fast read, expansion is a problem
- Linked allocation
  - Each block contains a pointer to the next block
  - Slower read, easier expansion
- Clusters
  - Cluster of consecutive blocks and clusters are linked
- Indexed allocation
  - Index blocks with pointers to actual data blocks

Types of Database File Organizations

How to organize the sequence of the records in the file?

- Files of unordered records
  - Heap files
- Files of ordered records
  - Sorted Files
- Files of hashed records

Files of Unordered Records

- Heap files
  - Records are placed in the order in which they are inserted
  - New records are inserted at the end of the file
  - Efficient insertion
- Expensive searching
  - Linear search – ($b/2$) blocks in average, b blocks in worst
- Records deletion
  - Find the block, copy the block to the buffer, delete the record, and rewrite the block back to the disk
  - Wasted storage space
  - Periodic reorganization

Files of Ordered Records

- Sorted Files
  - Ordering field (Key field)
- Advantages
  - Reading in the order of ordering field is very efficient
  - Finding next record often needs no disk access (same block)
  - Search on the ordering field results in fast access
    - Binary search – O(log2(b))
  - Search involving >, >=, <, <= on the ordering field is efficient
- Disadvantages
  - Insertion records are expensive
  - Modify the ordering field may change its position, requires a delete followed by an insert
  - No help for searches on non-ordering field

Sorted Files (continued)

	NAME	SSN	BIRTHDATE	JOB	SALARY	SEX
block 1	Aaron, Ed					
	Abbott, Diane					
	⋮					
	Acosta, Marc					
block 2	Adams, John					
	Adams, Robin					
	⋮					
	Akers, Jan					
block 3	Alexander, Ed					
	Alfred, Bob					
	⋮					
	Allen, Sam					
block 4	Allen, Troy					
	Anders, Keith					
	⋮					
	Anderson, Rob					
block 5	Anderson, Zach					
	Angeli, Joe					
	⋮					
	Archer, Sue					
block 6	Arnold, Mack					
	Arnold, Steven					
	⋮					
	Atkins, Timothy					
	⋮					
block n −1	Wong, James					
	Wood, Donald					
	⋮					
	Woods, Manny					
block n	Wright, Pam					
	Wyatt, Charles					
	⋮					
	Zimmer, Byron					

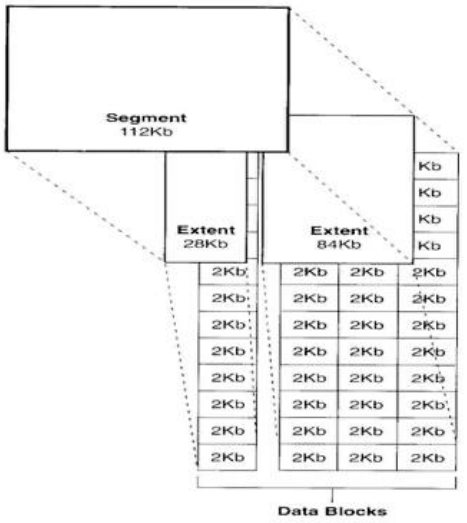
Figure 5.9 Some blocks of an ordered (sequential) file of EMPLOYEE records with NAME as the ordering key field.

Example:

20 records per block

If there are
10,000 records,
500 blocks,
So n=500

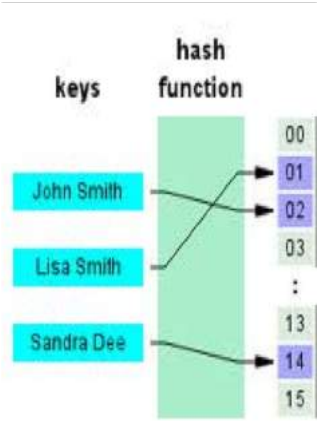
Needs 1MB storage

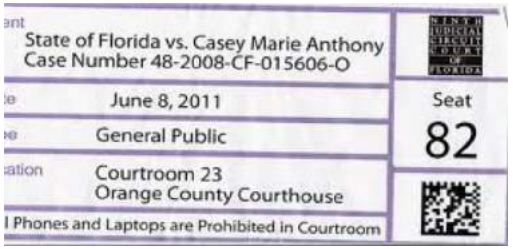


Files of Hashed Records

- Hash Files
  - Very fast access to the hash field
  - Hash field is commonly a key field of the file
  - Hash function or randomising function
    - Map a hash field value to the address of the disk block

      h(K)= K mod M, where M is usually prime
  - Easier insertion and deletion than sorted files
  - Efficient search for equality on the hashed field





Hashing for disk files

- Disk space divided into buckets
  - Each bucket is one block or a cluster of contiguous blocks
- Hashing function maps a key into a relative bucket number
- File header maintains a table that maps bucket number into a disk block address

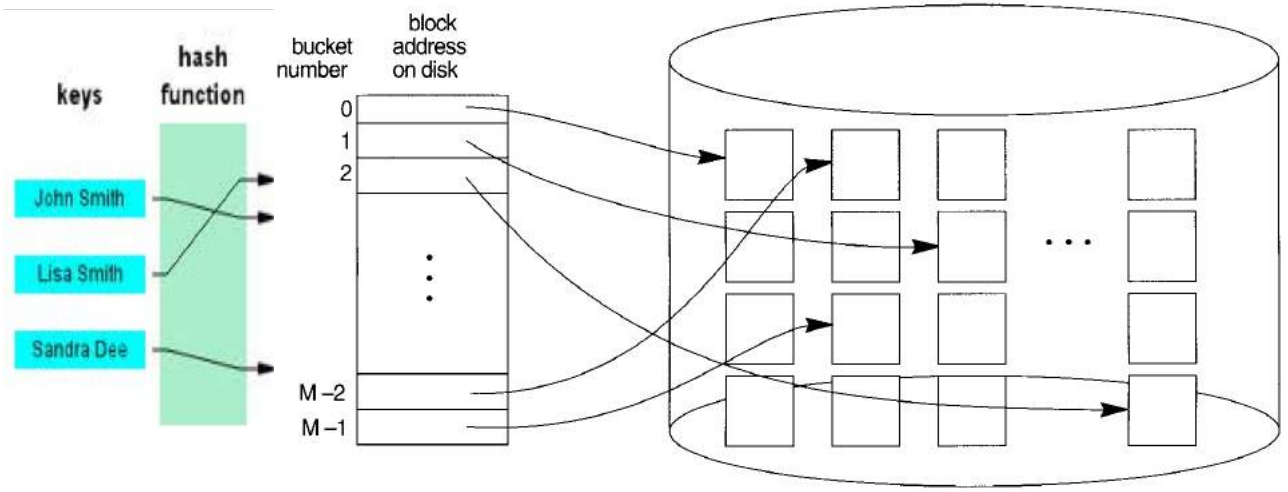


Figure 5.11 Matching bucket numbers to disk block addresses.

Collision Resolution in Hashing

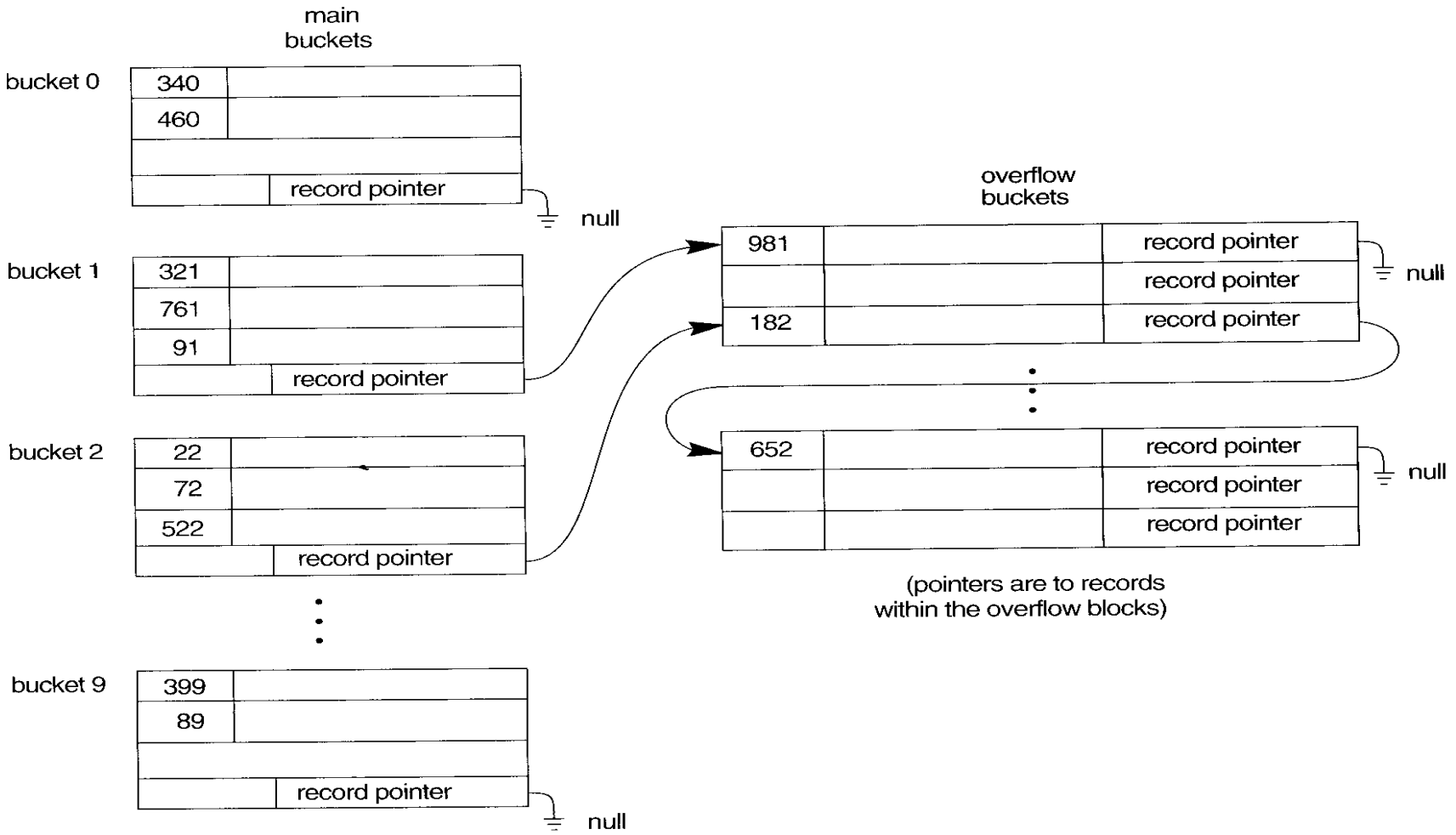


Figure 5.12 Handling overflow for buckets by chaining.

Question to Ponder

- If we have a few million records, how do we find the one(s) we want?

- Consider searching on any attribute.