# Overview of File Organizations and Indexing

SUN YAT-SEN UNIVERSITY

# Alternative File Organizations(文件组织)

- Many alternatives exist, *each good for some situations, and not so good in others:*
  - Heap files(堆文件): Suitable when typical access is a file scan retrieving all records.
  - Sorted Files(排序文件): Best for retrieval in *search key* order, or only a "range" of records is needed.
  - Clustered Files (with Indexes) (聚簇文件): Coming soon…

# Cost Model(代价模型) for Analysis

- **B:** The number of data blocks
- **R:** Number of records per block
- **D:** (Average) time to read or write a disk block
- *Average-case* analyses for *uniform random* workloads
- We will ignore:
  - Sequential vs. Random I/O
  - Pre-fetching
  - Any in-memory costs

# More Assumptions

- Single record insert and delete.
- Equality selection(等值选择)
  - exactly one match
- For Heap Files:
  - Insert always appends to end of file.
- For Sorted Files:
  - Files compacted after deletions.
  - Selections on search key.

# Cost of Operations

**B:** The number of data pages
**R:** Number of records per page
**D:** (Average) time to read or write disk page

| | Heap File | Sorted File | Clustered File |
|---|---|---|---|
| Scan all records | | | |
| Equality Search | | | |
| Range Search | | | |
| Insert | | | |
| Delete | | | |

# Cost of Operations

**B:** The number of data pages
**R:** Number of records per page
**D:** (Average) time to read or write disk page

| | Heap File | Sorted File | Clustered File |
|---|---|---|---|
| Scan all records | BD | BD | |
| Equality Search | | | |
| Range Search | | | |
| Insert | | | |
| Delete | | | |

# Cost of Operations

**B:** The number of data pages
**R:** Number of records per page
**D:** (Average) time to read or write disk page

| | Heap File | Sorted File | Clustered File |
|---|---|---|---|
| Scan all records | BD | BD | |
| Equality Search | 0.5 BD | $(\log_2 B) * D$ | |
| Range Search | | | |
| Insert | | | |
| Delete | | | |

# Cost of Operations

**B:** The number of data pages
**R:** Number of records per page
**D:** (Average) time to read or write disk page

| | Heap File | Sorted File | Clustered File |
|---|---|---|---|
| Scan all records | BD | BD | |
| Equality Search | 0.5 BD | $(\log_2 B) * D$ | |
| Range(范围) Search | BD | $[(\log_2 B) +$ #match pg$]*D$ | |
| Insert | | | |
| Delete | | | |

# Cost of Operations

**B:** The number of data pages
**R:** Number of records per page
**D:** (Average) time to read or write disk page

| | Heap File | Sorted File | Clustered File |
|---|---|---|---|
| Scan all records | BD | BD | |
| Equality Search | 0.5 BD | $(\log_2 B) * D$ | |
| Range Search | BD | $[(\log_2 B) +$ #match pg$]*D$ | |
| Insert | 2D | $((\log_2 B)+B)D$ | |
| Delete | | | |

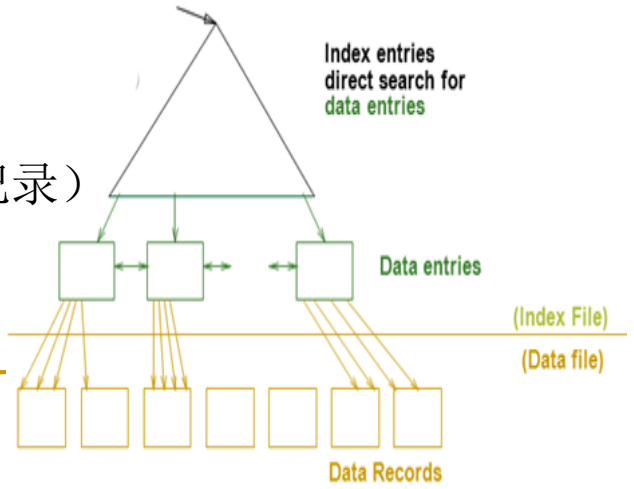
# Cost of Operations

**B:** The number of data pages
**R:** Number of records per page
**D:** (Average) time to read or write disk page

| | Heap File | Sorted File | Clustered File |
|---|---|---|---|
| Scan all records | BD | BD | |
| Equality Search | 0.5 BD | $(\log_2 B) * D$ | |
| Range Search | BD | $[(\log_2 B) +$ #match pg$]*D$ | |
| Insert | 2D | $((\log_2 B)+B)D$ | |
| Delete | 0.5BD + D | $((\log_2 B)+B)D$ *(because R,W 0.5)* | |

# Indexes – 索引

- 用途: Allow record retrieval *by value* in one or more fields
  - Find all students in the “CS” department
  - Find all students with a gpa > 3
- *Index* : disk-based data structure for fast lookup by value
  - *Search key(搜索键):* any subset of columns in the relation.
  - *Search key* need not be a *key* of the relation
    - Can have multiple items matching a lookup
  - 索引是为关系文件建立的索引文件
  - 索引文件由两部份组成
    1. 数据项部分
       - Data Entry(数据项) ⟺ data record（数据记录）
    2. 引导部份
       - 树索引技术
       - Hash索引

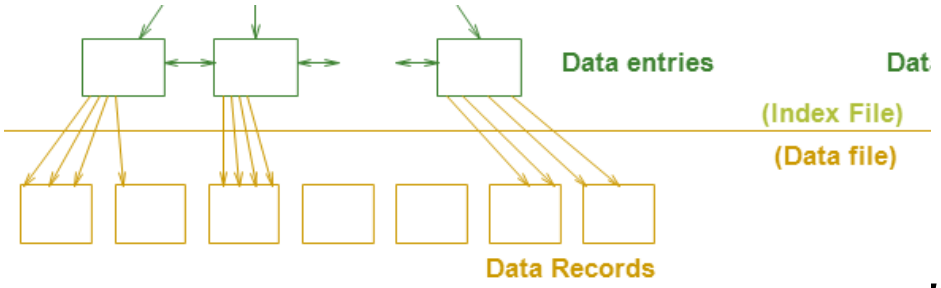


# Indexes – 索引(Contd.)

索引是文件

- Index contains a collection of *data entries (数据项)*

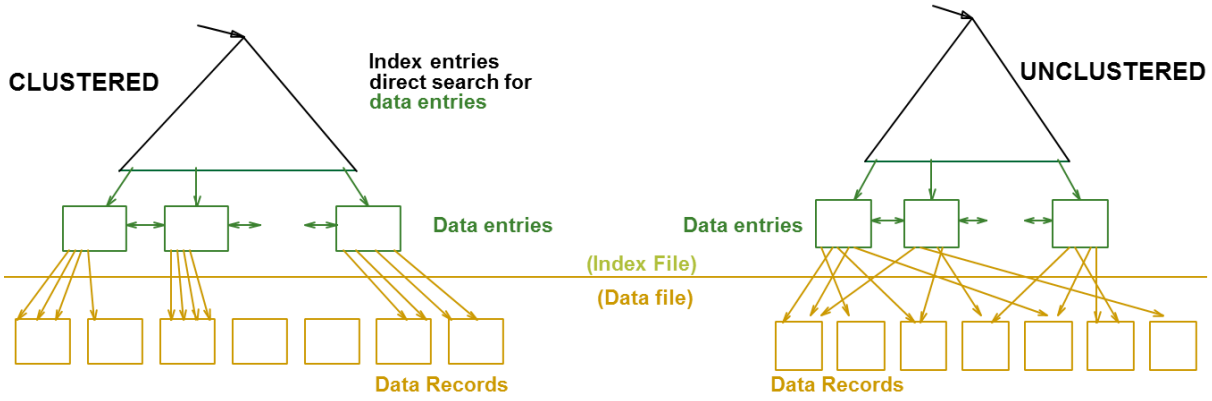
Data Entry(数据项) ⟺ data record（数据记录）




  - Items associated with each search key value ***k*** -- k*
- Three alternatives for Data Entry k*:
  1. Actual data record (with key value **k**) -- 数据记录
  2. <**k**, rid>, rid is record id of matching data record
  3. <**k**, rid-list>, rid-list is list of rids of matching data records

# Index Classification –索引分类

- *Clustered* vs. *Unclustered:*
  - Cost of retrieving data records through index varies *greatly* based on whether index is clustered or not!
- Clustered index – 聚簇索引:
  - order of data records the same as, or `close to', order of index data entries

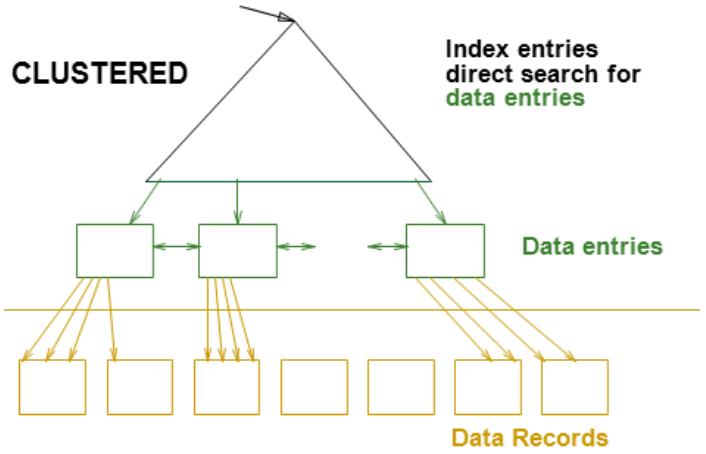



  - Alternative 1 implies clustered, *but not vice-versa*.

Data Entry(数据项)⟺ data record（数据记录）

# Unclustered vs. Clustered Indexes

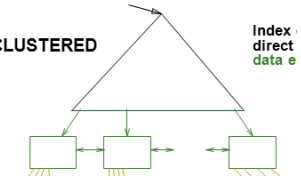
- Clustered Pros –优点
  - Efficient for range searches
  - Possible locality benefits
    - Disk scheduling, prefetching, etc.




- Clustered Cons-缺点
  - More expensive to maintain
    - on the fly or “sloppily” via reorganizations
    - Heap file usually only packed to 2/3 to accommodate inserts

# Cost of Operations

**B:** The number of data pages
**R:** Number of records per page
**D:** (Average) time to read or write disk page

| | Heap File | Sorted File | Clustered File<br>Alternative 1 |
|---|---|---|---|
| Scan all records | BD | BD | 1.5 BD   |
| Equality Search | 0.5 BD | $(\log_2 B) * D$ | $(\log_F 1.5B) * D$ |
| Range Search | BD | $[(\log_2 B) +$ #match pg$]*D$ | $[(\log_F 1.5B) +$ *#match pg*$]*D$ |
| Insert | 2D | $((\log_2 B)+B)D$ | $((\log_F 1.5B)+1) * D$ |
| Delete | 0.5BD + D | $((\log_2 B)+B)D$ *(because R,W 0.5)* | $((\log_F 1.5B)+1) * D$ |