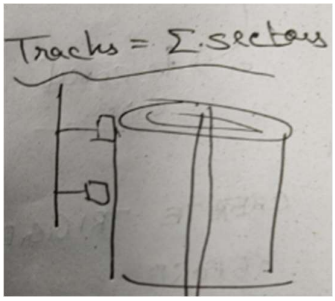
FILES AND INDEXES →



- Files is made up of blocks …
- **Hard Disk:** Seek Time due to track seeking, HDD made up of tracks and sectors , inner track has less data , [Platters → Cylinder → Tracks → Physical Sector → Logical Block]
- **Seek Time (t_s):** For positioning on tracks , time consuming process ,
- **Rotational Delay or Latency (t_r):** Time for bringing head under correct sector (or block) under the R/W Head …
- ***ts >> tr*** [tr is associated with RPM (revolutions per minute)]
- On average ½ revolution is needed , so ROT DELAT = 1 / (2 * p) minutes [p is RPM]
- **Block Transfer Time:** tr bytes / sec → Block Size 'B' → tr = 1s → 1 = 1/tr s → B = ***B / tr seconds*** … [Here , tr is not rotational delay → this is transfer rate …]
- ---B--- | G | ---B--- → G is inter block separation → To read B , ***'B + G'*** has to be read … → (B + G) / tr seconds BT Time

- **File:** Collection of Records **Records:** Collection of related fields , maybe of fixed or variable length (***variable length*** when → storing records of different types in same file /OR/ same type records but fields of variable length (varchar [eg:- name]) /OR/ Optional Fields (Superclass - Subclass) /OR/ Multi-valued attribute or field)
- **Fixed Length Records:** Jumping to record easier (O(1)) , Modification easier (for variable length , don't know length , so can it fit in its current space surrounded by other records ???) , Wastage of space (not necessary fields always take space , for variable length field allot max fields)
- **How to handle variable length field:** Store max length [fixed size record] , store length then data [variable sized record]
- **How to handle Optional Fields:** If NULL , store all bits '0' → A value cannot be 0 (in binary form) here …
- **Multiple Values:** Keep provision of max cnt of values [fixed len] ,

- **How to have fast access in variable sized records ???**
  - Store sizes of records → O(n) to sum up …
  - Store cumulative sum of records (in other words positions) → O(1) retrieve → O(n) update when modification of record
- **Record Separator:** To separate variable sized records [------$---------$----------$---$--- …]
- **Field Separator:** To separate variable sized fields
- **How to handle Optional Fields here:** <FIELD_NAME , VALUE> pairs (Also , keep a field name separator)
- **How to handle Multivalued Attribute (Value separator):** $v1.v2. --- . vn$ ---------- [separators must be differentiable among themselves]
- Separators must be chosen such that they can be differentiated from data (Keep this info in file header) …

SPANNED VS UNSPANNED ORGANISATION →

- **Spanned Org:** If a record is allowed to cross the block boundary and continue into another block …
- **Unspanned Org:** Record cannot cross block boundary , residual part is wasted (this can be used for linking the next block)
  [In case of Record Size (R) > Block Size(B) → Only option is Spanned Org]

- **Blocking Factor:** #Records per block **[BFR = floor(B / R)]** (in case of fixed length record in unspanned org)
  → Average #Records per block **[b = ceil(r / bfr)]** (in case of variable length record in unspanned org) [A file has 'b' #blocks , 'r' is the total #records …]

ALLOCATION OF BLOCKS →

- **Candidate Block:** Logically deleted records where new inserting records can be overwriting here (new space not needed)
- **Contiguous Allocation:**
  - **Adv:** Sequential Access is faster (No Seek Time or Rotational Delay) [Complete Transfer Time - $t_s + t_r + k * btt$] (k -> #Blocks , btt -> Block Transfer Time)
  - **Disadv:** Growth of the file is difficult (no guarantee that next block is free) (may need complete relocation) (Internal Fragmentation)
- **Linked Allocation:** Linked List of Blocks
  - **Adv:** Growth Easier
  - **Disadv:** Accessing all blocks is time consuming
- **Cluster Allocation:** Linked Collection of contiguous blocks [Cluster is collection , Clusters are linked]
  - **A combination of contiguous and linked ameliorating their adv and disadv …**
- **Indexed Allocation:** Table of Pointers to block addresses