Date ___ / ___ / _____

Saathi

Transactions:

A Automicity
C Consistency
I Isolation
D Durability

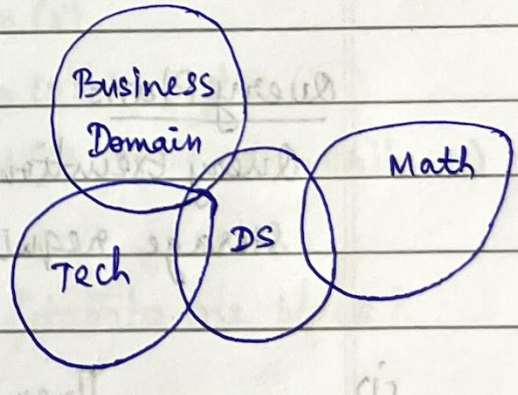
Acid properties some times fails in case of Big Data. So we use Base properties which is a flexible property to accept some kind of inconsistencies unlike ACID.

Database

Structured
→ Defined Schema
→ ACID properties
Ex: CSV

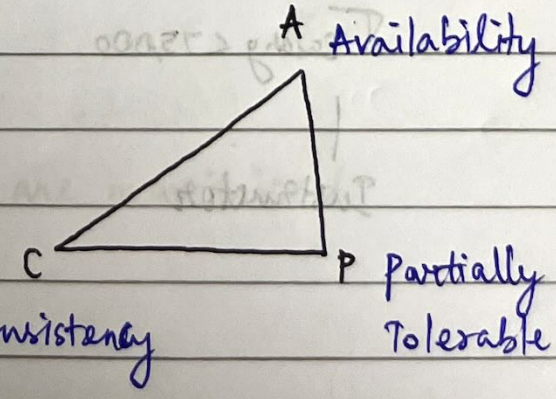
Unstructured
→ Schema may be there
→ BASE properties
Ex: DM, NLP
Health

Semi Structured.




BA: Basically Available
S: Soft State
E: Eventual Consistency → not up to date.

Important Vs:
Volume, Variety, Velocity,
Veracity, Visualisation,
Value, Vagueness.



Date ___/___/_____

Consider an employee table. Find 3 maximum salary in employee table. Select the department name & no. from department table in which there is no employee in the department.

Construct an instructor table with ID, name, dept, salary & course table with course id, name, department, credits; Teaches with id, course id, semester, year.

(i) Find the names of instructors whose salary < 75,000.

(ii) Find the names of instructors in the music dept. together with course title that the instructors teach.

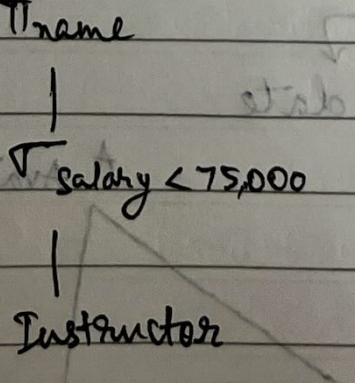
→ SELECT name FROM INSTRUCTORS where salary < 75000.

→ SELECT instructor.name, course.title FROM INSTRUCTORS NATURAL JOIN TEACHES

Query Plan:

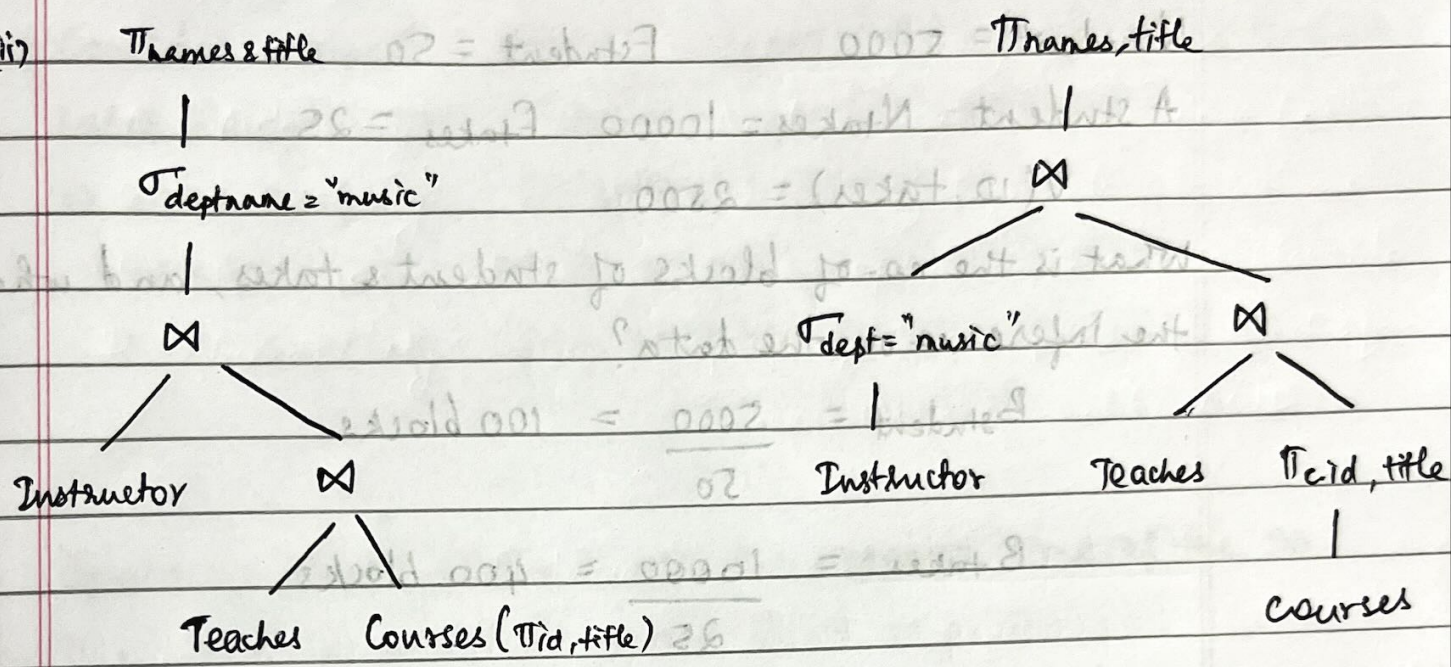
Query Execution Plan is useful for size estimation for storage requirements.

(i)



Date ___ / ___ / _____

(ii)




## Estimating Statistics of Estimating Result.

$n_R$ – no. of tuples in a relation (r)

$b_r$ – no. of blocks containing tuples of relation (r).

$f_r$ – blocking factor of relation (r).

$l_r$ – size of tuple of a relation (r) in bytes.

V – no. of distinct tuples. V(A, r) (A – attribute).

Blocking Factor – How many records fit into one block?

$$f_r = \frac{n_r}{b_r} \quad \text{(or)} \quad b_r = \frac{n_r}{f_r}$$

All these estimations are maximised.

Date ___/___/_____

Saathi

$N_{student} = 5000$ $F_{student} = 50$

A student $N_{takes} = 10000$ $F_{takes} = 25$.

$V(ID, takes) = 2500$.

What is the no. of blocks of student & takes, and what is the inference on the data?

$$B_{student} = \frac{5000}{50} = 100 \text{ blocks}$$

$$B_{takes} = \frac{10000}{25} = 400 \text{ blocks}.$$

1) The paging system access time is 20ns. Memory access is 18ns Hit ratio is 80%. Find effective memory access.

⇒ Hit ratio (TLB + MM) + miss (TLB + MM).

2) A non pipeline system takes 50ns to process a task. Same task can be processed in 6 segment pipelines with a clock cycle of 10ns. Determine approximately speed up ratio of pipeline for 500 tasks.

⇒ Speed up = $\frac{\text{without pipeline}}{\text{with}} = \frac{n(T_n)}{(n+k-1)\,T_p}$

$$= \frac{500 \times 50}{(500+6-1)\,10}$$

Date ___/___/_____

Saathi

3) Consider a pipeline having 4 faces with duration 60, 50, 90 & 80. Latch delay is 10ns. calculate speed up ratio.

⇒ without pipeline : 60+50+90+80 = 280

with pipeline : 90 + 10 = 100.

Speed up : $\frac{280}{100}$ = 2.8.

Speed up in Big Data to see the response time of the DB.

4) Calculate throughput for 1000 tasks with pipeline of 4 segments & time taken for 1 cycle is 100ns, for sequential 280ns.

⇒ Pipeline – 4×100 + 99900. = 100300.

Sequential – 1000/280 = 3.

Throughput = 100/100300 = 0.00099.

5) What will be the blocks req. if a search query is given, if a binary search is performed?

⇒ Binary Search = $\log_2 N = \log_2(1000) = 10$.

No. of blocks (as a whole) = $\frac{10000}{10}$

6) Dept. no 20 to be searched :

⇒ $\log_2(20)$ . 20 blocks should be searched.

Page No.

Date ___ / ___ / _____

## Range Partitioning:

↳ dept no = 15

3rd disk. can be directly searched for. Searching is efficient. if not queries may be overloaded.

## Hash partition:

Depending on hash values, we move to that value.

$i \rightarrow hash(i) = k_1$ $\quad K = 1, 2, 3$

$j \rightarrow hash(j) = k_2$

leads to collision.

Avoid: by increasing k,

selection of best hashing method (imod2, imod5, ..)

↳ Universal hashing (more than 1 hash func.)

stream data handling.

Partitioning → Parallelism.

→ Round Robin partitioning.

Processing < Parallel

Sequential

Date ___/___/_____

Consider join operation for customers & order. No. of cust=5000 No. of orders tuples = 10000. The unique customers in customer table = 5000. Order table = 2500. No of tuples in join operation? What type of join?

⇒ Normal join : $n_c \times n_o = 5000 \times 10000$

distinct : $\frac{n_c \times n_o}{V(\text{cust}, \text{order})} = \frac{10000 \times 5000}{5000} = 10000.$

Parallelism
- Intra query
- Inter query.

1) Consider a sailor's table that is tree index. Select info for boat id > 5 & lowest boat id = 1 & highest = 100.

⇒ If key is greater than a given value,

$$\frac{\text{high} - \text{value}}{\text{high} - \text{low}}.$$

key < value : $\frac{\text{value} - \text{low}}{\text{high} - \text{low}}.$

Date ___/___/_____

Saathi

2) Consider nested loop join where 1000 pages are required to store R.

for every tuple or in R
  for every tuple s in S 'r.id' = 'sid' do
    add <r,s>

M = 1000 pages. Consider 100 tuples per page ($P_r$)
Calculate no. of tuples, N = 500 pages $P_s$ = 80 tuples per page.

Apply outer relation R, perform join operation with S, how many I/o operations are required?

⇒ cost model is $M + (P_r * m) * n$ ⇒ Page by tuple comparison.

$= 1000 + (100 * 1000) * 500$

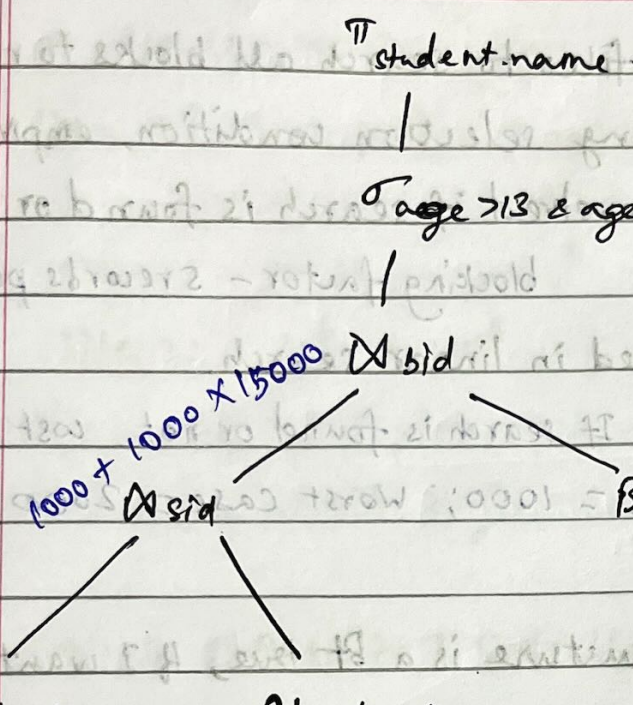
$= 1000 + 50000000 = 50001000$.

Page by page comparison → $M + M * n$

3) Consider student S with field sid, name, age, address, Book b with bid, title, author, Checkout C with sid, bid, date of checkout. Consider 10000 student records are stored on 1000 pages. 50,000 book records stored on 5000 pages. 300000 checkout record on 15000 pages. 500 different authors are existing. Student age ranges from 7-24. Write a query to filter students under age of 13-19, who has taken author 'XYZ'. What is the total cost of I/o accessing?

Page No.

Date ___/___/_____




Use maximum query for estimation purposes. Highly estimated queries won't given s/o total cost.

If index is done on Book with authors, what will be the cost involved for full table scan for the query ⇒ "select from Books where author = 'JK'." what is the cost involved for index scan?

⇒ B(S) = 1000 B(C) = 15000. = 1000 + 1000 X 15000.

Table scan = $\frac{50000}{500}$ = 100.

4) Consider block nested loop, the no. of pages in movie data file is 800, no. of pages for director is 200. Using the algorithm, if accessing time is given for 1 block is 5ms, what is the time required to execute the algorithm.

Date __/__/____ Saathi

5) In a linear search, the files to search all blocks to retreive all the records satisfying selection condition, empNo = b what will be the cost involved if search is found or not?
employee - 10000 records blocking factor - 5 records per block.

⇒ All blocks are searched in linear search.

cost = 2000. If search is found or not, cost = 2000.

Best case = 1 ; Avg case = 1000; Worst case = 2000.

Consider the data structure is a Bt tree, If I want to select the empNo > 5.

a = select * from R,S,T where R.B = S.B & S.C = T.C and RA<40 (1/3, 1/10, 1/2 ← selection factor)

T(R) = 30K T(S) = 200K T(T) = 10K.

What is the estimate of the query?

⇒ $30000 \times 200000 \times 10000 \times \frac{1}{3} \times \frac{1}{10} \times \frac{1}{2}$

Selection factor: How many queries turn up for the condition.

6) Consider a site 1 contains employee of 10,000 rows, rowsize = 100. dept at site 2 with 100 rows, rowsize = 35 byte. Consider the query to filter firstname, last name & dept. name that matches with employee & dept. & approx. 10,000 tuples are returned, 1 tuple size is 40 bytes. If employee & dept query is posted in site 3, what is the cost involved? (even for site 1 & 2).

Page No.

Date ___/___/_____

Saathi

⇒ Result = 40x10000 bytes. + fetching cost.

For site 3, transfer from site 1 & 2. transfer cost is more here

$10000 \times 100 + 35 \times 100 + 40 \times 10000$ → total cost.

site 1: $35 \times 100 + 40 \times 10^4$ site 1 is better. ← transfer cost.

site 2: $10^6 + 4 \times 10^5$

For an efficient plan, process all the queries in site 1 itself, to display in site 3, just send results to site 3.

7) 3 4 6 2 9 4 8 7 5 6 3 1 2, Apply a merge sort to sort this

⇒

Date ___/___/_____ external sorting method

One page contains 2 data; 6 pages; 3 buffers.

| | |
|---|---|
| 40 | 3 |
| 8 | 34 |
| 23 | 12 |
| 2 | 13 |
| 5 | 17 |
| 25 | 15 |

Consider 3 buffer page, size it can hold = 2 records, find the cost of input & output operations.

External merge sort.

Consider 2 files: File1 with values (44, 10), (33, 12), (55, 31)

File2 with values (18, 22), (27, 94), (3, 1)

6 pages, 3 buffers

24 ip, op [M=3, N=3]

For sorting $2(M+N) = 12$

merging $2(M+N) = 12$

$\underline{24}$

$2n \log n$

Date ___/___/_____

Saathi

db.inventory.insert([
{ item: "Journal", qty: 25, tags: ["blank", "red"], dim_cm: [14, 21] })

→ Find the documents whose qty is greater than 20
ii) dim_cm >15 & <20.

→ Insert in the inventory : item & instock field with array of properties included: warehouse = A qty = 5, warehouse = C qty = 15.

→ Insert with item no, qty, size as a property : height, weight, status => A, D.

i) Select the query whose status = A or D.
ii) Select the status = A & qty < 30.

$gt $lt.

filter field condition

i) db.inventory.find( qty: {$gt: 25})

end -,
or : $or ( )

$in
$all.
$lt
$gte.

Date ___/___/_____

Saathi

→ checks for order of the values.

→ db.inventory.find({tags: ["red", "blank"]})

→ db.inventory.find({tags: {$all: ["red", "blank"]}})

→ db.inventory.find({dim_cm: {$gt: 15, $lt: 20}})

→ db.inventory.insert({item: "journal",
instock: [{warehouse: "A", qty: 5},
{warehouse: "C", qty: 15}]});

(1) Find the document that contains instock qty <= 20.
(2) Find: qty = 5, warehouse = A.

→ db.inventory.find({"instock.qty": 5,
"instock.warehouse": "A"});

Consider transaction 1 & 2:

| T1 | T2 | Serialisable, Recovable |
|---|---|---|
| R(X) | | |
| W(X) | | |
| | R(X) | |
| commit | | |
| | commit | |

Page No.

Date ___ / ___ / _____

Saathi

| $T_1$ | $T_2$ |
|---|---|
| R(A) | |
| W(A) | |
| | R(A) |
| | C = C + A |
| | commit |
| Abort ~~commit~~ | |

Non recovable.
(because of abort)

| $T_1$ | $T_2$ | $T_3$ | $T_4$ |
|---|---|---|---|
| R(A) | | | |
| W(A) | | | |
| | R(A) | | |
| | W(A) | | |
| | | R(A) | |
| | | W(A) | |
| | | | R(A) |
| | | | W(A) |

If $T_1$ fails, rollback should occur for other transaction
⇒ because of no commit.

serialisable.