SYSTEM DESIGN

1. HW: UBER
2. INTERVIEW QUESTIONS

---

1. Places stationary → Quad Tree
   - Sharded
   - In memory

   Zomato, Swiggy, Gmaps

2. UBER / OLA / LYFT

   Taxis → not stationary.

   Find nearest X taxis?

   1. Simplification → shard by city

      Delhi, Gurgaon, Noida

      Bangalore

      Mumbai | Navi Mumbai

      Pune

      Nagpur

      Hyderabad

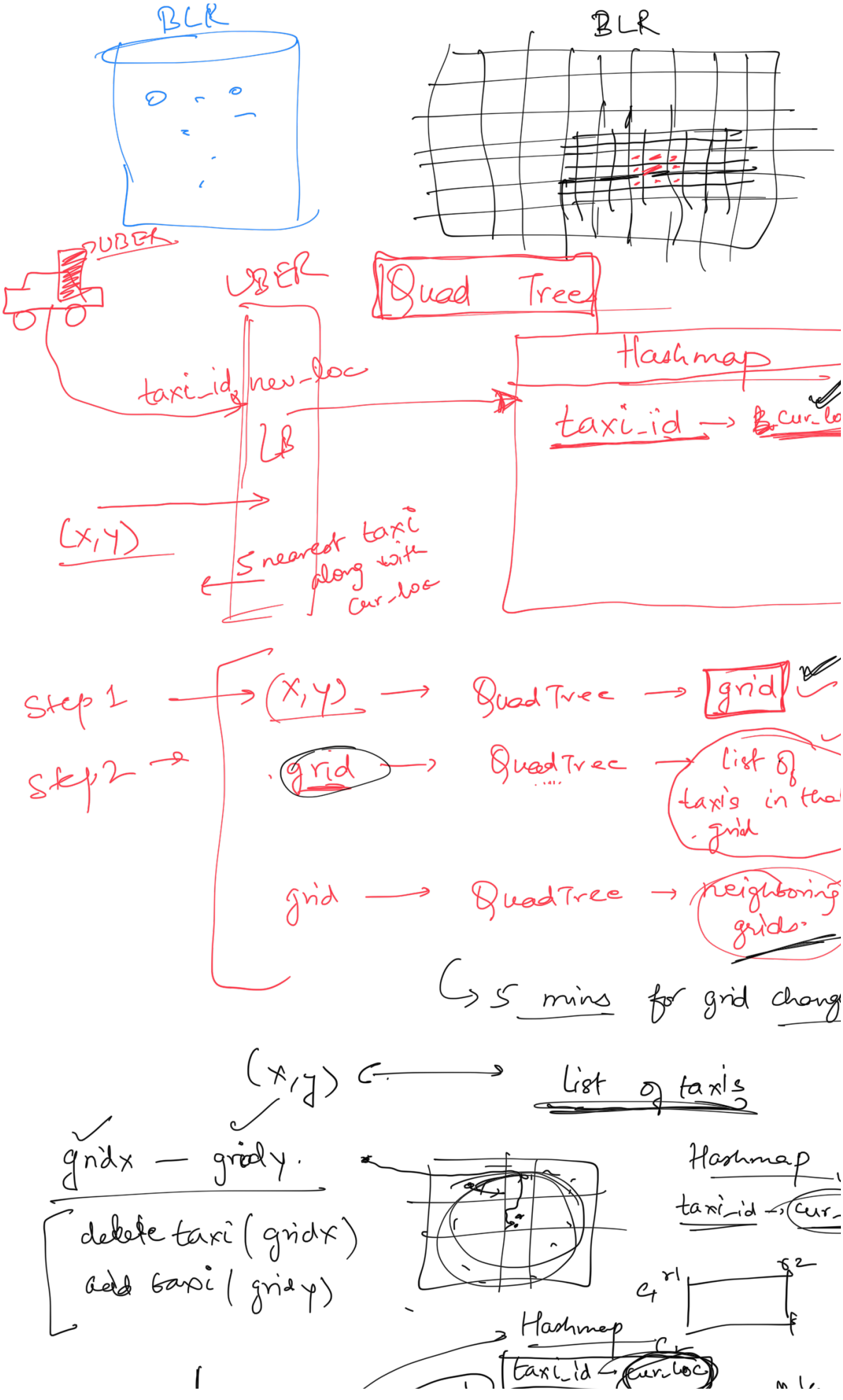
      ~~Goa~~ Vizag

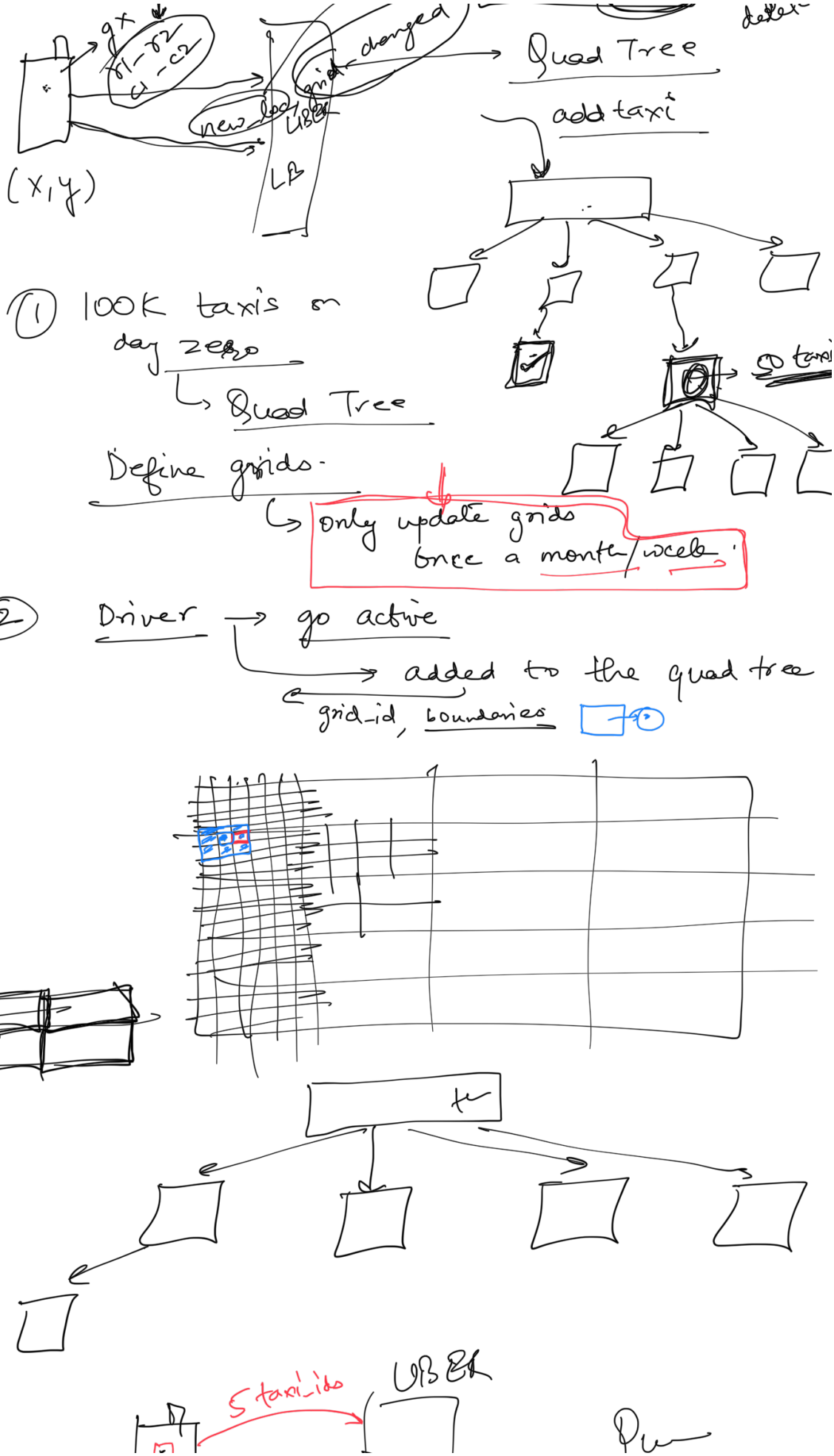
      UBER

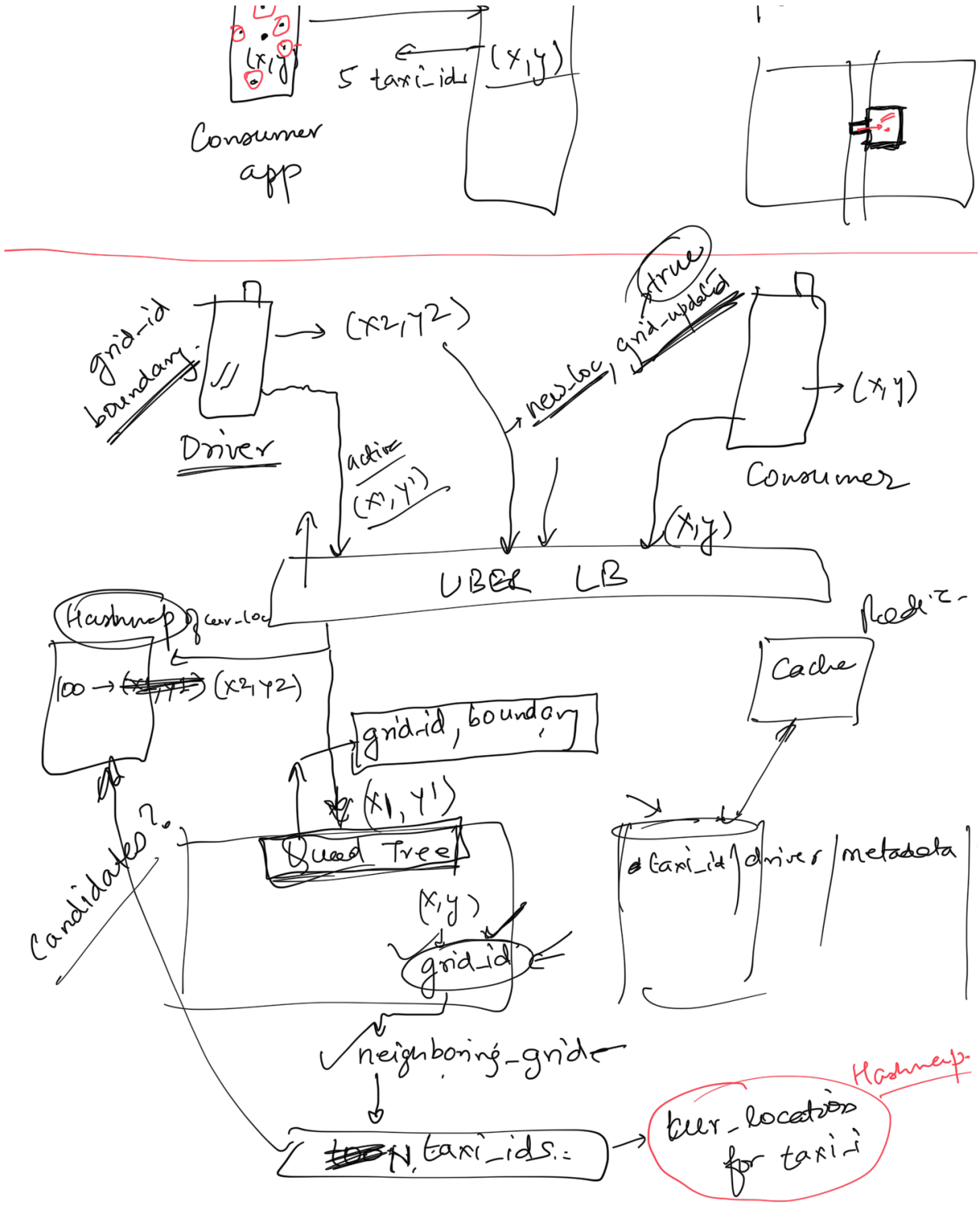
      Intracity | Inter

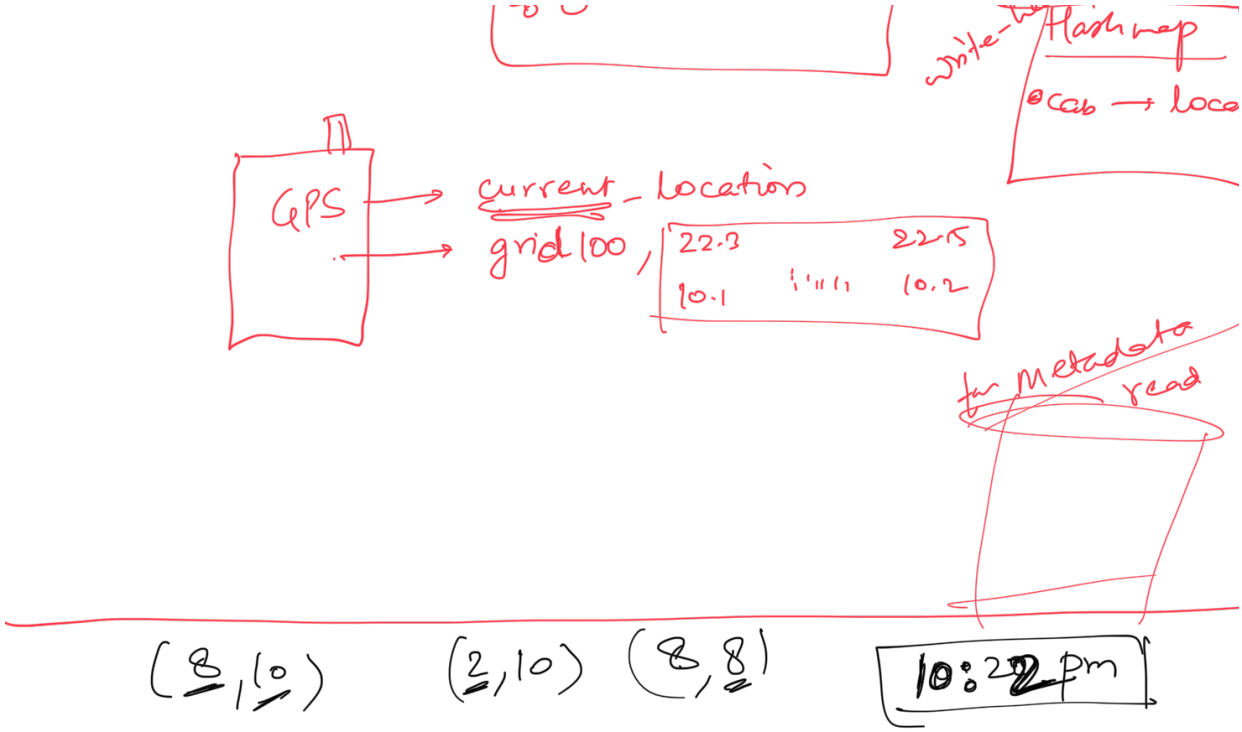
   1. Shard by city.

      Bangalore → Book taxis

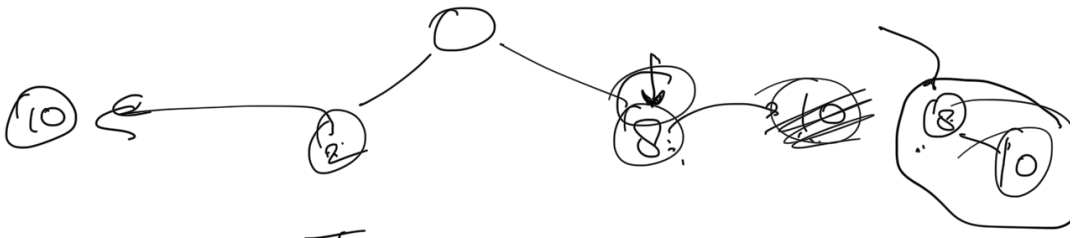

BLR
BLR
UBER
UBER
Quad Tree
taxi_id
new_loc
LB
Hashmap
taxi_id →
(x, y)
5 nearest taxi along with cur_loc
Step 1
(x, y)
QuadTree
grid
Step 2
grid
QuadTree
list of taxis in the grid
grid
QuadTree
neighboring grids
5 mins for grid chang
(x, y)
List of taxis
gridx — gridy
delete taxi (gridx)
add taxi (gridy)
Hashmap
taxi_id
cur_
Hashmap
taxi_id
cur_loc


Quad Tree
add taxi
(x, y)
LB
grid changed
(1) 100K taxis on day zero
Quad Tree
Define grids.
only update grids once a month/week
(2) Driver → go active
added to the quad tree
grid_id, boundaries
taxi_ids


(x,y)
5 taxi_ids
(x,y)
Consumer app
grid-id boundary
(x2,y2)
Driver
active (x',y')
new_loc
grid_updated
true
(x,y)
Consumer
(x,y)
UBER LB
Hashmap
cur_loc
100 → (x2,y2)
Cache
grid_id, boundary
(x1,y1)
Quad Tree
candidates
(x,y)
grid_id
taxi_id | driver | metadata
neighboring_grids
N taxi_ids
curr_location for taxi i
Hashmap
Leaf
- boundary
- all places in this grid
100K
taxi_id
prev
curr



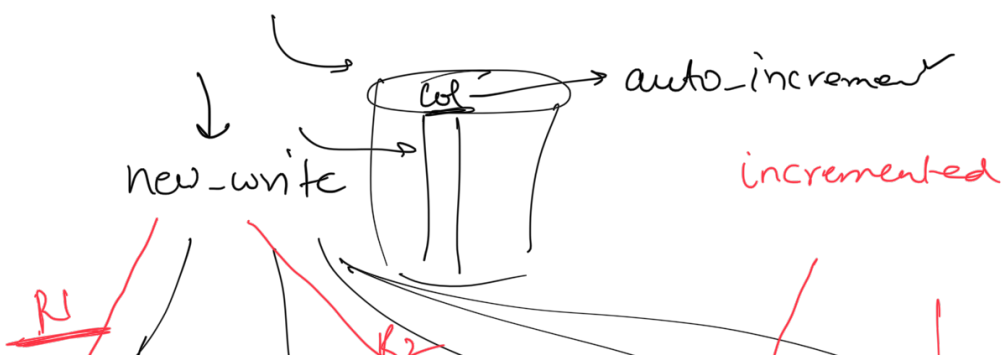
(8,10) (2,10) (8,8) 10:22 pm



Tunable consistency

$R + W > X$

unique_id generator



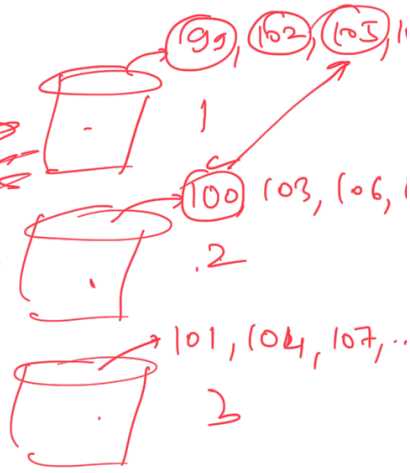
① Single machine → SPOF ✗

② timestamp → ✗ (de-duplicate acr machines)

③ random-hash + timestamp ✗ (incremental random host on still col

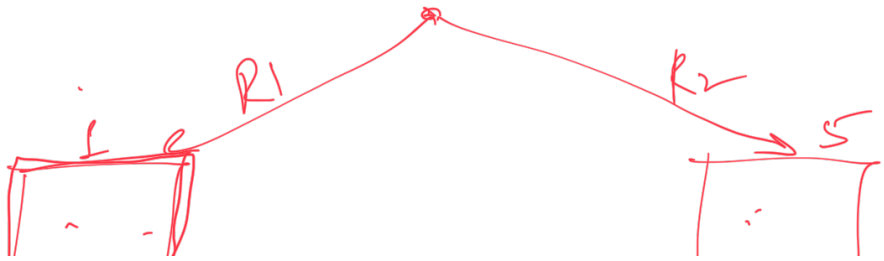
④ uuid → unique, incremental ✗

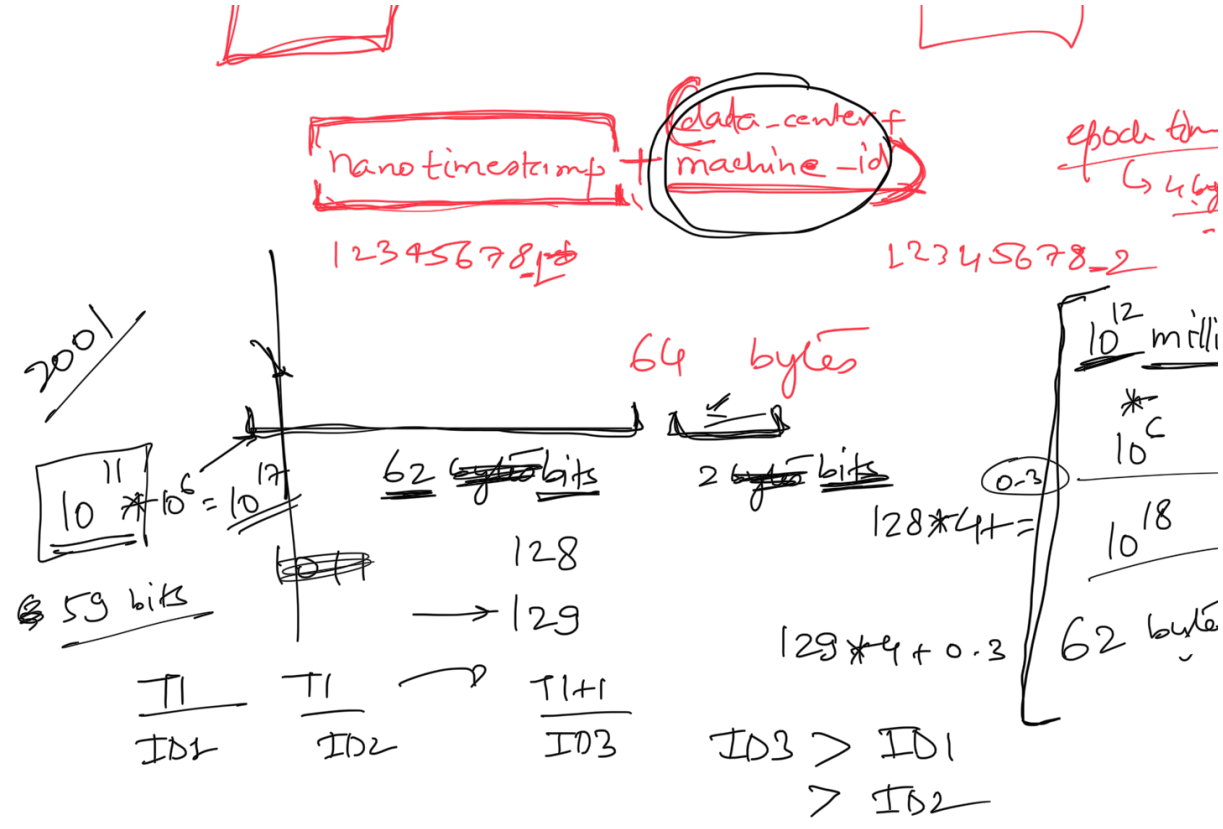
K machine → unique, incremental ✗

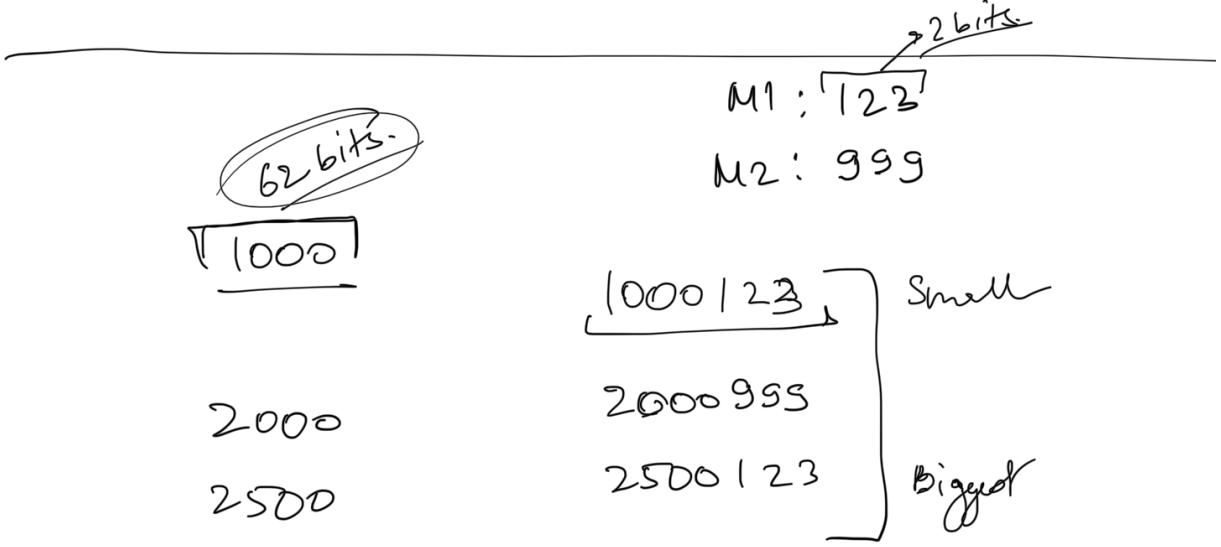


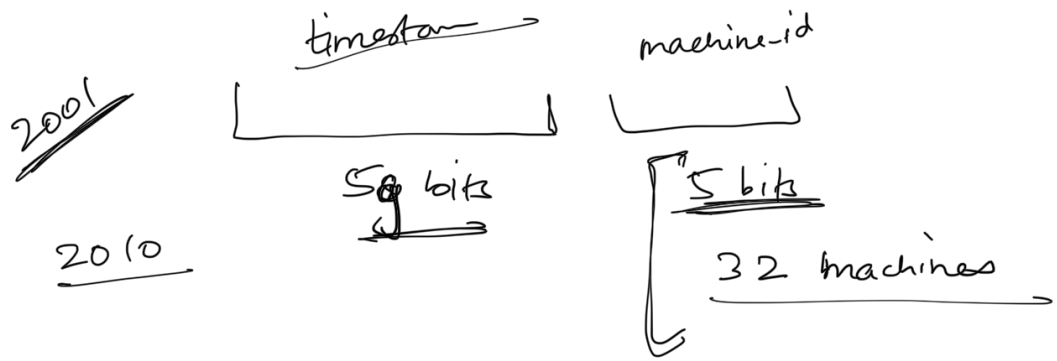
---

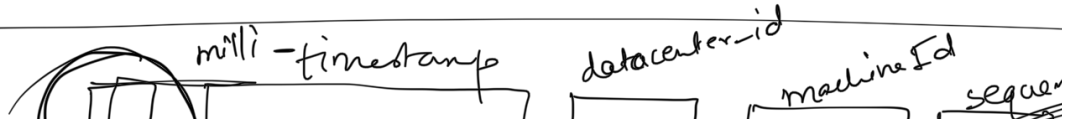
timestamp

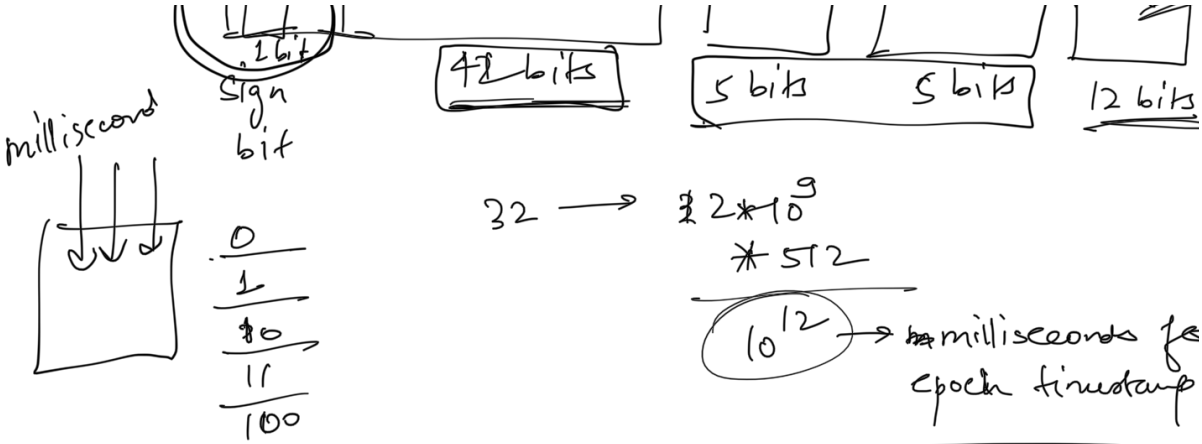



nano timestamp
data_center_id
machine_id
epoch time
12345678_1
12345678_2
2001
64 bytes
62 bits
2 bits
59 bits
128
129
T1
T1
T1+1
ID1
ID2
ID3
ID3 > ID1
> ID2
0-3
128*4+=
129*4+0.3
62 byte


2 bits
M1: 123
M2: 999
62 bits
1000
1000123
2000999
2500123
Small
Biggest
2000
2500


timestamp
machine_id
2001
59 bits
5 bits
32 machines
2010

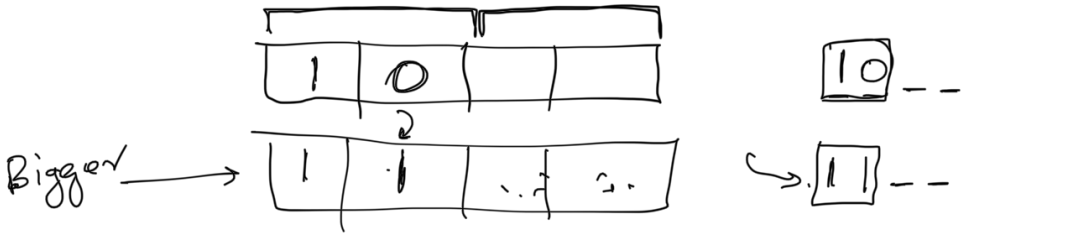

milli-timestamp
datacenter_id
machineId



Unique_id → incremental

12
4000

4000 ids in a ms
on every machine



Rate-limiting → DDOS Attack

- for IP Addr.
- for userId

- 5 requests per second allowed per IP Add
- 2 requests per second allow for user

(1) LN=2

(1) where:

(2) what algo? How to implement?