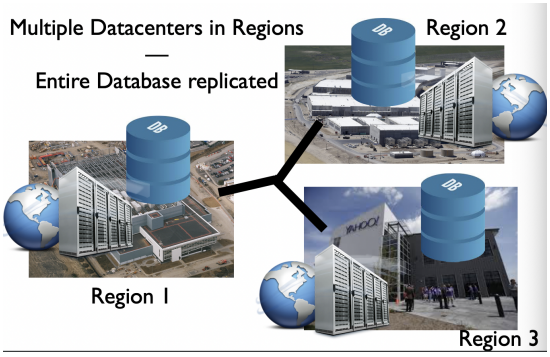
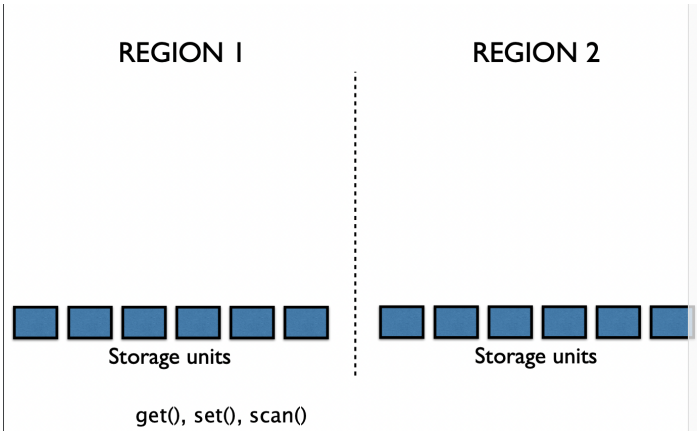
CAS CS 350
Lec 19

PNUTS

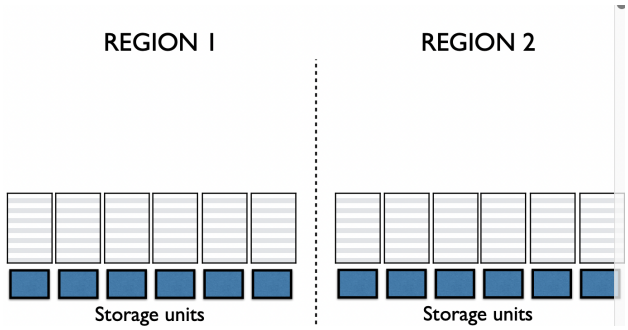
1. Acid
    a. Atomicity: Transaction is committed as a whole or not committed
    b. Consistency: transaction can only bring the database from one consistent state to another
    c. Isolation: transaction should not worry about others (not worry about other transactions)
    d. Durability: once we commit, we cannot undo
    e. Not supported by PNUTS (is very flexible)
2. The Big picture
    a. Multiple datacenters in regions
    b. Entire database is replicated
    c. 

    d. PNUTS → each region stores exactly the same data in all geographic regions (clients view from the replica that is closest to the database that is closest to them)
3. Replication
    a. Goal: make the reads fast
        i. Clients connect to local data center
        ii. Web app in that data center can do the reads fast - no remote communication
        iii. Reads are more frequent than writes
        iv. But reads possible stale (reads here do not go through a leader → if reads are together with writes without locks, clients can see stale data)
    b. Tradeoff: writes slow
    c. Raft prevents stale reads because every client request goes through the leader and there is at most one leader

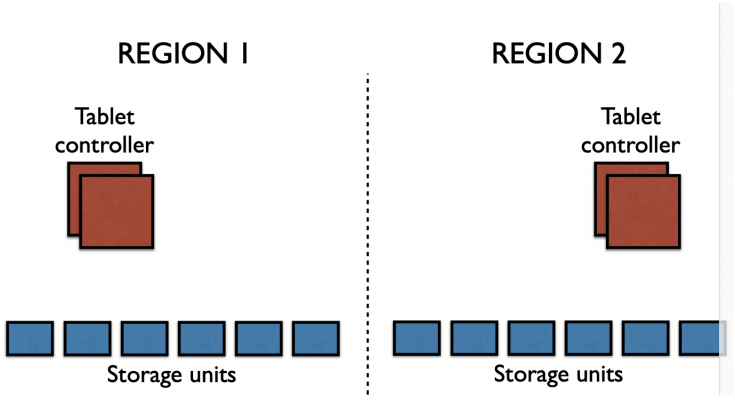
4. PNUTS

a. 

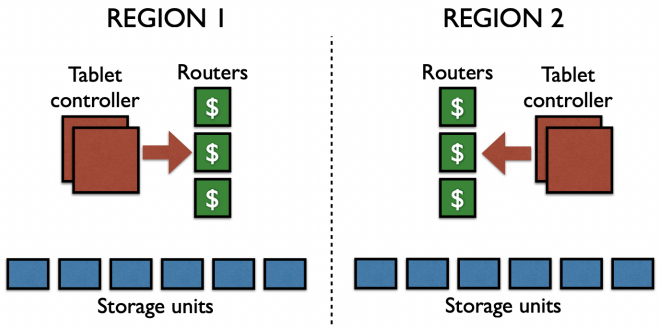
b. Each region has storage units (harddisks that store units)
c. Tablets are ordered with key or can be hashed using hash function
d. All data replicas have the same setup/layout (all have the same tablets with same data)

e. 

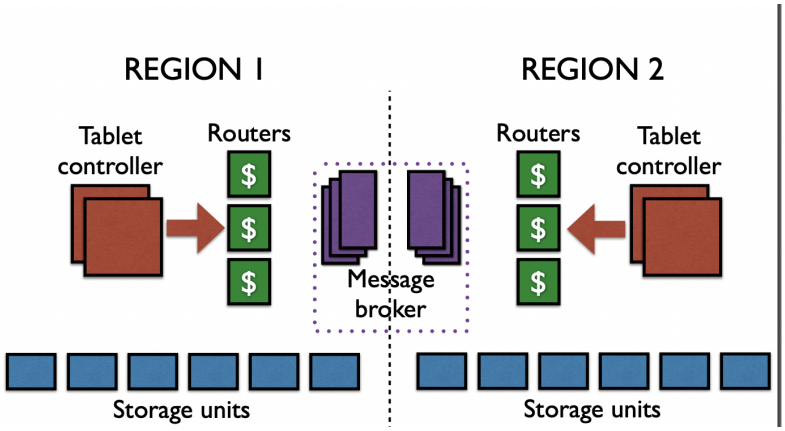
f. On top of the units, system uses a controller (tablet controller) that maintains mapping of rows/data rows/tablets
g. Controller acts as a memory index to the data
h. Controller can reorganize mapping of tablets of units as well

i. 

j. Controller uses primary backup and once it fails, there is a backup that takes place

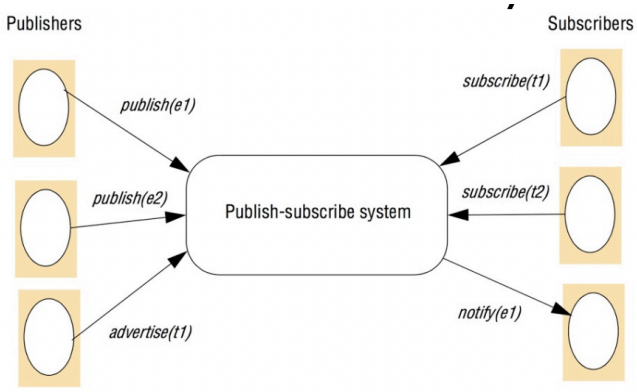
k. 

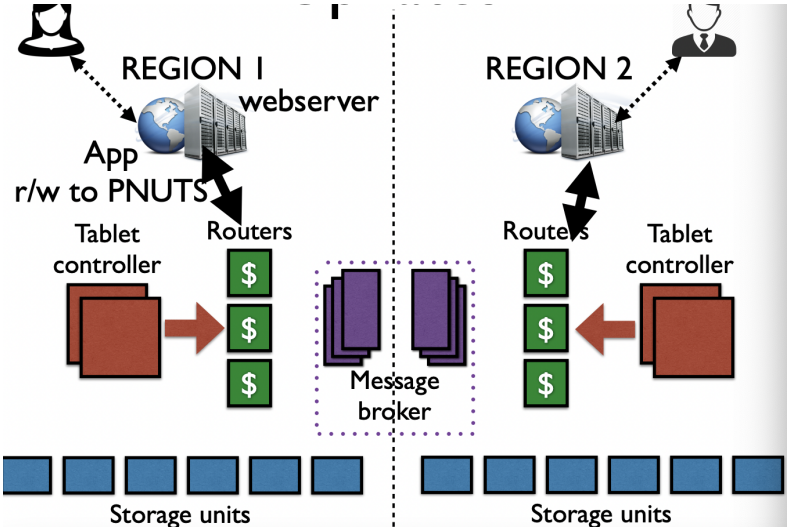
l. Problem: concurrency (if many users request at the same time, there can be problems)
m. PNUTS use routers in each region

n. 

o. Each region, there is set of servers that provide service
p. Pub-sub acts a log and delivering message across servers for PNUTS

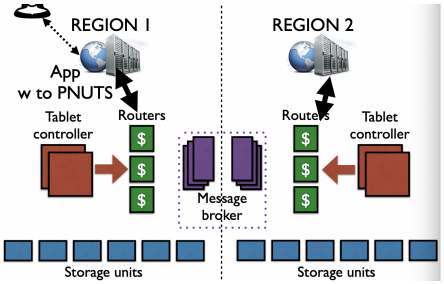
5. Publish-subscribe
    a. “Decoupled and reactive style of programming”
    b. Publishers publish structured events to an event service
    c. Subscribers subscribe to particular events through subscriptions
    d. System must match subscriptions against published events and ensure the correct delivery of event notifications
    e. One-to-many paradigm of communications
    f. Asynchronous communication, event based
    g. Delivery guarantees: reliability, agreement, latency, etc.

h. 

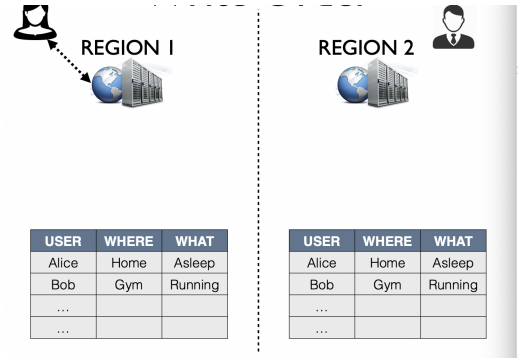
6. Updates
   a. 

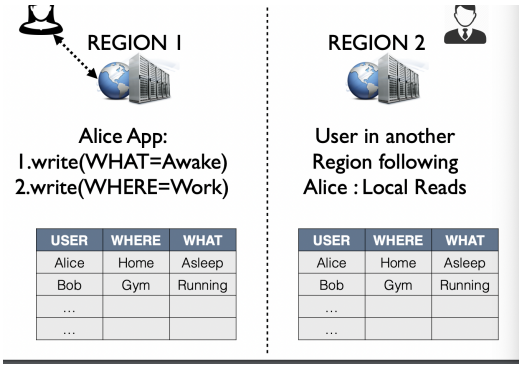
   b. PNUTS can communicate with storage units to read/write data
   c. If users write, it must be propagated to all regions (there can be delays) so that other clients can view the updated data some point
   d. Why not just have app logic send update to every region and use pub-sub system?
      i. Easier abstraction
      ii. Make system fault tolerant → not easy task to do
      iii. Prefer to push all complexity to the system and make applications light-weighted (simple as possible)
   e. What if app crashes after updating only some regions? What if concurrent updates to same record?
7. Record based Mastering
   a. Pnuts has a “record master” (Region) for each record/row
      i. All updates are directed to master
      ii. All writes go through the same master
      iii. Master imposes order on writes for each record (master chooses the order)
      iv. Responsible storage unit executes updates one at a time per record
      v. Tells MB to broadcast updates in order to all regions

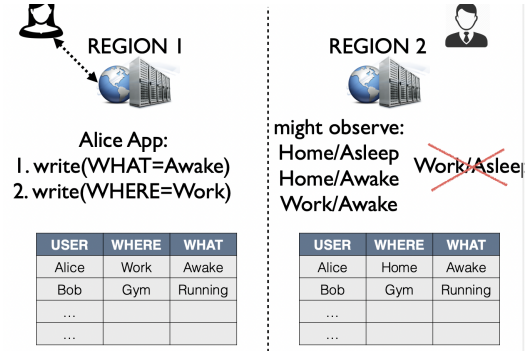
b. Two cases:
    i. Update occurs at master region
    ii. Update occurs at non-master region
c. Difference between the two cases
    i. Performance
    ii. Updating at master region is faster because it is local
    iii. Updating at non-master region is slower

8. So what is going on
    a. Why per-record master?
        i. Each record has its own master region (it can change dynamically based on the workload)
        ii. PNUTS master is dynamic each row has hidden column that indicates master region - can migrate
    b. Hopefully get best of both worlds
        i. Fast reads of local data
        ii. Fast writes because master is often the local region
9. Update walk through
    a. 

    b. Try to update in region I although the master-region is region 2
    c. Application sends update to its local router (router contains some version of the index that is stored in the controller– memory to figure out where the data is)
    d. Router forwards it to the appropriate region
    e. Storage units send update request through the message broker to the router in region 2
    f. Router 2 updates to its local storage unit
    g. Send the message to backup and then commits (message is updated and we won't lose it)
    h. The system replies to the client and application continues
    i. How do we update all replicas (to all datacenters)?
        i. Publish the update (transmitting the update to all regions by the server)
        ii. But can take some time to update (asynchronous)
    j. But for this to be useful what do we need from the message brokers
        i. Fault tolerance (reliable communication)
        ii. Communication must be asynchronous (if it is synchronous (first wait to propagate to all regions and return to the client))

10. Message broker: cool idea
    a. Reliable: logs to disk, keeps trying, to cope with various failures
    b. Ordered: record's replicas eventually identical even with multiple writers
    c. Async: apps don't wait once write committed at MB
    d. Encapsulates all kinds of the tricky DS stuff - App's and DB don't have to deal with it
11. Writer order
    a. 

    b. Assuming there is only region, these are two identical databases
    c. 

    d. Client tries to update the data of Alice in region 1
    e. Region 1 is the master region for Alice record
    f. Writes will be updated in region 1 and will be updated in the same order in region 2 as well
    g. 

    h. Per-record time consistency is the name and can do this without global master but with master region that is very flexible (can change anytime)

    i. Only single record ordering of updates and transactions
12. Why is it OK for writes to take effect slowly?
    a. Fundamental benefit: reads are then very fast, since local and reads usually greatly outnumber writes
    b. PNUTS mitigates write cost:
        i. Application waits for MB commit but not propagation (“asynchronous”)
        ii. Master likely to be local (they claim 80% of the time)
        iii. So MB commit will often be quick
    c. Down side: readers at non-master regions may see stale data
13. How stale might a non-master record be?
    a. Depends on how quickly MB sends updates to regions
    b. Guess: less than a second usually
    c. Longer if network slow/flaky, or MB busy
14.