LIC
भारतीय जीवन बीमा निगम
LIFE INSURANCE CORPORATION OF INDIA
बिपीनकुमार अेम. पटेल
सदस्य - अधिकारीओ के लिये क्षेत्रिय प्रबंधक क्लब
Bipinkumar M. Patel
Member of the Zonal Manager's Club for Agents
828 83A
• सदस्य • अधिकारीओ के लिये क्षेत्रीय प्रबंधक क्लब • MEMBER • ZONAL MANAGER'S CLUB FOR AGENTS

CH-19 Conti. ①

* Concurrency Control in Distributed Databases

(I) Locking Protocals - lock manager deal with replicate Data

(a) single lock Manager
- One site "S_i" maintain locks.
- Transaction needs to lock data item, first send request to "S_i" and wait until grant request.
- Transaction read data from any sites but write performe on all sites

Adv i) Simple Implimentation - two message for lock Req. one " " lock Release
ii) simple deadlock handling

DisAdv : i) Bottleneck - "S_i" become a boltoleneck
ii) Vulnerability - Concurrency controller is lost if "S_i" is fails.

(b) Distributed lock Manager - lock manager function is distributed over several sites.
- each site maintain lock manager
- when transaction wishes to lock data item φ that is not replicated and resides at site S_i, send lock request, wait until grant request.
→ It is simple in implementation and reduce the degree of bottleneck.
→ However, deadlock handling is much more Complex

(c) Primary Copy.
- Copy of φ must be reside in precisely one site
- when Transaction needs to lock a data item φ, it request a lock at site φ. The primary copy enables concurrency control for replicated data. If φ fails, φ is inaccessible even though it is present on other sites.

(d) Majority Protocol :
- φ is replicated in 'n' different sites, then lock request message must be send to more than one-half of the n site in which φ is stored.
- The transaction does not operate on φ until it has successfully obtained a lock on a majority of the replicas of φ.

Adv - Easy to deal with site failures.

Dis i) Implementation - at least $2\left(\frac{n}{2}+1\right)$ message for lock request
at least $\left(\frac{n}{2}+1\right)$ message for unlock

ii) Deadlock handling - suppost T_1, T_2 transaction lock data item φ. φ is available on S_1, S_2, S_3, S_4. T_1 lock S_1 & S_3
T_2 lock S_2 & S_4
→ now T_1 wait for S_2 & S_4
T_2 wait for S_1 & S_3

कार्यालय : परबडी के पास, पो.समरखा.
निवास : सरदार पार्क सोसायटी, पो.समरखा, ता.जी.आणंद-388360.
फोन : (नि) (02692) 256673 (मो) 98242 32755.
Offi. : Nr. Parabadi, Po. Samarkha.
Resi. : Sardar Park Society, Po. Samarkha, Ta. Dist. Anand-388360.
Ph. : (R) (02692) 256673 (M) 98242 32755. E-mail : bipinpatellic@yahoo.com

(e) Biased Protocol -
- shared locks are given more favorable treatment than requests for exclusive locks.
i) shared locks - request a lock on Q at one site only
ii) exclusive locks - request a lock on Q, at every site
→ less overhead on "read". Working good when "read" is more frequent than "write".

(f) Quorum Consensus Protocol
- generalization of majority Protocol.
- It assigns read and write operations on an item x two integers called read quorum "Q_r" and write Quorum "Q_w".

$$Q_r + Q_w > S \quad \text{and} \quad 2 * Q_w > S$$

- To read and write enough replica must be locked.
- small reads need to obtain fewer locks but write will be higher
- This Protocol can simulate the majority Protocols & biased Protocols

(2) Time Stamping
- Unique timestamp is given to each transaction
Two methods for unique timestamp generation
① Centralized - single site distributes timestamp (logical counter used)
② Distributed - each site generate local timestamp using either logical counter or the local clock.

local unique timestamp	site identifier

- Problem - Some site generate timestamp fast
Sol^n - define within each site S_i a logical clock (LC_i) which generate Unique local timestamp. Logical clock can be implemented as a counter that is incremented after a new local timestamp is generated
- S_i advance its logical clock whenever a transaction T_i with timestamp $\langle x, y \rangle$ visits that site and x is greater than the current value of LC_i. for Synchronization.

LIC
भारतीय जीवन बीमा निगम
LIFE INSURANCE CORPORATION OF INDIA
बिपीनकुमार अेम. पटेल
Bipinkumar M. Patel
Member of the Zonal Manager's Club for Agents
828 83A
MEMBER
ZONAL MANAGER'S CLUB FOR AGENTS

(3) Replication with Weak degrees of Consistency

- master-slave replication - update at a primary site and automatically propagates update to other sites.
- Some transactions do not obtain locks at remote sites
- Replica reflect all update of transactions up to some transaction to the primary data
- This transactions are in serialization order.
- Database configured to propagate update immediately after they occur at the primary or updates only periodically.
- This replication use in central office to branch offices of an organization.
- Another use, creating a copy of the database to run large queries so that queries do not interfere with transactions. Updates should be propagated periodically - every night.
- Multimaster replication updates are permitted at any replica of data item and automatically propagated to all replicas.
- Transaction update local copy & system updates other replicas.
- One way of updating replicas is to apply immediate update with two phase commit, using one of the lock manager.
- generally biased protocol used
- Many database systems provide an alternative form of updating They update at one site, with lazy propagation. - Lazy Propagation allow transaction processing to proceed even if a site is disconnected from the network
- Lazy Propagation
  - (a) Update Primary site and it will update other replica lazily.
    - problem some site have old values, serializability problem can be occurs
  - (b) Update only replica and propagated other sites.

Problem since the same data item may be updated concurrently at multiple sites.

④ Deadlock handling

- Deadlock Prevention may result in unnecessary waiting & rollback.

→ Each sites have **local wait-for graph**

- In deadlock detection, the main problem in deciding how to maintain the wait-for graph.

→ These local wait-for graphs are constructed in the usual manner for local transactions and data items. when a transaction T_i on site S_1 needs a resource in site S_2. If resource is held by transaction T_j, the system insert edge from $T_i \rightarrow T_j$ in the local wait-for graph of site S_2.

→ If local wait-for graph has cycle which means deadlock but no cycle which not mean no deadlock.

→ Some time Union of two sites wait-for graph has cycle.

* Centralized deadlock detection, system constructs and maintain global wait-for graph in a single site: deadlock detection coordinator.

- Some time Controller find deadlock but it is not that is because of "false cycles".

The global wait-for graph reconstructed if

(i) new edge is inserted or removed

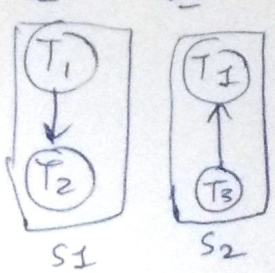
(ii) periodically (iii) Coordinator needs to invoke cycle-detection Algo.

Rollbacks if

(i) False cycle - occur due to message delays.

(ii) Actual Deadlock - find victim and coordinator notify all the sites and roll back that transaction.

false cycle



→ If T_2 request T_3 in site S_2 and $T_I \rightarrow T_2$ is complete.

→ But before delete edge between T_I & T_2, System add edge between T_2 and T_3

→ Coordinator find cycle $T_I \rightarrow T_2 \rightarrow T_3 \rightarrow T_I$

Actually deadlock is not present

LIC
भारतीय जीवन बीमा निगम
LIFE INSURANCE CORPORATION OF INDIA
सदस्य - अभिकर्ताओं के लिये क्षेत्रिय प्रबंधक क्लब
MEMBER - ZONAL MANAGER'S CLUB FOR AGENTS
बिपीनकुमार अेम. पटेल
सदस्य - अभिकर्ताओं के लिये क्षेत्रिय प्रबंधक क्लब
Bipinkumar M. Patel
Member of the Zonal Manager's Club for Agents
828 83A

* Availability :-

- functioning even during failure referred to as robustness.
- All failures handle with different ways
- message loss → Retransmission
- link failure - alternative route
- If system not find other route means network partition
- system find failure but can not find which type of failure

*Suppose that site S1 has discoverd failure

(1) Abort such transaction, locks on data at sites that are still active.
(2) wait until the site to become accessible again
(3) when data objects are replicated then read & updates are possible.
(4) when failed site recovers, update all changes.

(1) If Replicated data are stored at failed site, catalog should be updated, so that queries do not reference the copy at the failed site.
- when site rejoins, care must be taken to ensure data consistent.

(1) If failed site is a central server than election must be held to determine new server.

→ In particular, these situations must be avoided to ensure Consistency.

(1) Two or more central servers are elected in distinct partitions
(2) More than one partition updates a replicated data item.

* Replication Protocol design

(1) Majority-Based Approach

- Each data object store with it a version number to detect latest value
- lock request same as majority Based lock

(a) Read operation look at all replicas and read the value that has the highest version number.
(b) write operation look at all replicas and find highest version number and increment one and store into all replicas. which has obtained locks, and set the new version number.
- If above two rules are violated, the transaction must be aborted. Here two phase commit protocol can be used, on the sites which are available.

कार्यालय : परबडी के पास, पो.समरखा.
निवास : सरदार पार्क सोसायटी, पो.समरखा, ता.जी.आणंद-388360.
फोन : (नि) (02692) 256673 (मो) 98242 32755.
Offi. : Nr. Parabadi, Po. Samarkha.
Resi. : Sardar Park Society, Po. Samarkha, Ta. Dist. Anand-388360.
Ph. : (R) (02692) 256673 (M) 98242 32755. E-mail : bipinpatellic@yahoo.com

- Here reintegration is not require because many replicas are updated so read can be identify latest value easily.

(2) Read One, write all

→ Spacial case of quorum censensus - read quorum = 1
write quorum = n

→ In this case, if single site containing a data item fails, no write to the item can proceed since the write quorum not available.
→ Any available replica can be read and a read lock is obtained. A write operation is shipped to all replicas if a site is down, the transaction manager proceeds without waiting for site recover.
→ Reintegration action is require for down sites.
→ This will work fine for no network partitioning System.

(3) Site Reintegration

- Use for repaire sites
- when failed site recovers, it must reflect changes.
- It is complicated
- entire system halt temporarily while failed site rejoins
- There are techniques have been developed to allow failed sites to reintegrate and concurrent updates to data items.
- Before read or write lock is granted system ensure that it has caught up on all updates to the data item.

(4) Comparison with remote Backup

- actions such as Concurrency control and recovery are perform at the single site, only data and log records are replicated at the other site.
- Remote Backup avaid Two phase commite and its overheads.
- It provides high availability at lower cost

(5) Coordinator selection

- Some time Coordinator fails and execution stops. One way to Continue execution is by maintaing a backup to the coordinator, which is ready to assume responsibility if the coordinator fails.
- Back up coordinator has all information which is require in execution but it can not take action that affects sites.
- If coordinator fail, back up coordinator take charge. Prime advantage is continue processing without any interruption.
- If backup is not present, then newly appointed coordinator would have to seek information from all sites.
- If it get only few information then it may be abort several active transactions, and to restart them under the control of the new coordinator

19(5)

- backup coordinator avoid substantial amount of delay in new coordinator schema.
- Disadvantage is the overhead of duplication of tasks. Furthermore Coordinator and backup need to communicate regularly for synchronization

LIC
भारतीय जीवन बीमा निगम
LIFE INSURANCE CORPORATION OF INDIA

बिपीनकुमार अेम. पटेल
सदस्य - अधिकारीओ के लिये क्षेत्रिय प्रबंधक क्लब
Bipinkumar M. Patel
Member of the Zonal Manager's Club for Agents
828 83A

सदस्य अधिकारीओं के लिये क्षेत्रीय प्रबंधक क्लब
MEMBER ZONAL MANAGER'S CLUB FOR AGENTS

- In the absence of backup coordinator, Election algorithms enable new coordinator
- This algo. require unique identification number be associated with each active site in the system.

Bully algo. for election

- identification number of site "S_i" is "i" and site is active with the largest identification number. Whenever coordinator fails, the algo. Choose active site which have largest identification number
- In this case, S_i sends request to coordinator within a prespecified time interval T coordinator should responce if not S_i will take charge and send election message to every active site that has a higher identification number. If sites not respond, for specific time T then it will elect itself. and inform all sites [redacted] with identification number less than i. If S_i does receive an answer, it begin a time interval T', to receive a message informing it that a site with a higher identification number has been elected.
- If S_i receives no message within T', then it assumes the site with a higher number has failed and restart algo.

→ After a failed site recovers, it immediately forces all sites to change coordinator if on higher number present. That's why it's called bully algo.

⑥ Trading off Consistency for Availability.

- If a network failure results in more than two partitions, no partition may have a majority of sites. The write all protocol provide availability but not consistency.

CAP theorum: Distributed Database can have at most two of the following properties (i) Consistency (ii) Availability (iii) Partition tolerance

→ Consistent Data if result is the same as if the operation were executed on single site.

→ In any large scale distributed system partition can not be prevented and as a result either availability or consistency has to be secrificed.

कार्यालय : परबडी के पास, पो.समरखा.
निवास : सरदार पार्क सोसायटी, पो.समरखा, ता.जी.आणंद-388360.
फोन : (नि) (02692) 256673 (मो) 98242 32755.

Offi. : Nr. Parabadi, Po. Samarkha.
Resi. : Sardar Park Society, Po. Samarkha, Ta. Dist. Anand-388360.
Ph. : (R) (02692) 256673 (M) 98242 32755. E-mail : bipinpatellic@yahoo.com

→ Bank system require ACID properties
→ Social-networking system require BASE properties:
- Basically available
- Soft state
- Eventually consistent

→ Primary requirement is availability, updates should be allowed, even in the event of partitioning. Soft state refers to the property that the state of database may not be precisely defined, with each replica have different state. Eventually consistent is requirement that once partitioning is resolved - last step, identified inconsistent copies of data items and updates to new version of data that is meaningful to the application. Application decides how to resolve the inconsistency not database.

* Distributed Query Processing

→ Centralized system - criterion of cost is disk accesses while Distributed system - Data transmission over the network and potential gain in performance from having several sites process parts of the query in parallel.

* Query Transformation :-
- If no replicas are fragmented, we choose the replica which transmission cost is lowest. However, if replicas is fragmented the choice is not so easy to make, since several joins or Unions Require to reconstruct the "account" relation

Ex $\sigma_{branch\_name = "Hillside"}(account)$, Here account = account1 ∪ account2

$\sigma_{branchname = "Hillside"}(account1) \cup \sigma_{branchname = "Hillside"}(account2)$

* Simple join Processing
- account ⋈ depositor ⋈ branch
  S1 S2 S3

strategies (i) ship copies of all three relations to site S1 ← drawback: more computation, more Network traffic
(ii) S1 to S2 compute account ⋈ depositor → temp
S2 to S3 compute temp ⋈ branch

* Join strategies that exploit Parallelism

r1 ⋈ r2 ⋈ r3 ⋈ r4 _output at S1

① ship r1 from S1 to S2 ② Compute r1 ⋈ r2 at S2 = temp1
③ Simultenously ship r3 from S3 to S4 ④ compute r3 ⋈ r4 at S4 = temp2
⑤ ship temp1 from S2 to S1 ⑥ ship temp2 from S4 to S1
⑦ temp1 ⋈ temp2 at S1

(19) 6

* Semijoin strategy ($r_1 \ltimes r_2$)

→ Suppose expression $r_1 \bowtie r_2$ where r_1 is stored at S_1 and r_2 is stored at S_2.

→ Problem: all tuples from r_2 moved to S_1 and then performe join, Network cost is too much

LIC
भारतीय जीवन बीमा निगम
LIFE INSURANCE CORPORATION OF INDIA
MEMBER ZONAL MANAGER'S CLUB FOR AGENTS
बिपीनकुमार अेम. पटेल
सदस्य - अधिकारीओ के लिये क्षेत्रिय प्रबंधक क्लब
Bipinkumar M. Patel
Member of the Zonal Manager's Club for Agents
828 83A

Let the schemas of r_1 & r_2 is R_1 & R_2

* possible strategy

① Compute $temp_1 \leftarrow \pi_{R_1 \cap R_2}(r_1)$ at S_1

② Ship $temp_1$ from S_1 to S_2

③ Compute $temp_2 \leftarrow r_2 \bowtie temp_1$ at S_2

④ Ship $temp_2$ from S_2 to S_1

⑤ Compute $r_1 \bowtie temp_2$ at S_1

$$r_1 \bowtie r_2 \bowtie \pi_{R_1 \cap R_2}(r_1) = (r_1 \bowtie \pi_{R_1 \cap R_2}(r_1)) \bowtie r_2 = r_1 \bowtie r_2$$

→ Overhead of shipping $temp_1$ will be dominated by the saving of shipping only a fraction of the tuples in r_2.

→ $temp_2$ should have less tuples then and then it strategy is useful.

* Cloud Based Databases

① Data Representation

- data-storage support XML & JSON for representing such complex data.
- Both data structure is flexible in set of attributes that a record contains
- store data with associated key and retrive data with that key.
- In social networking application when someone post photo then it will send to his friends so whenever their friends open social media they received posts. "no delay"
- Cloud data-storage systems are based on two primitive functions put(key, value) and get(key)
- In Bigtable, a record is not stored single value but is instead split into component attributes that stored separately.
- Each attribute value is just a string as far as Bigtable concerned.
- Record identifier is used to fetch the record.
- single instance of Bigtable can store data for multiple applications with multiple table per application, by simply prefixing the application name and table name to the record identifier.
- Data storage system typically allow multiple version of data items to be stored. Versions are often identified by timestamp, or integer value.
- Lookups can simply pick data which has highest version.

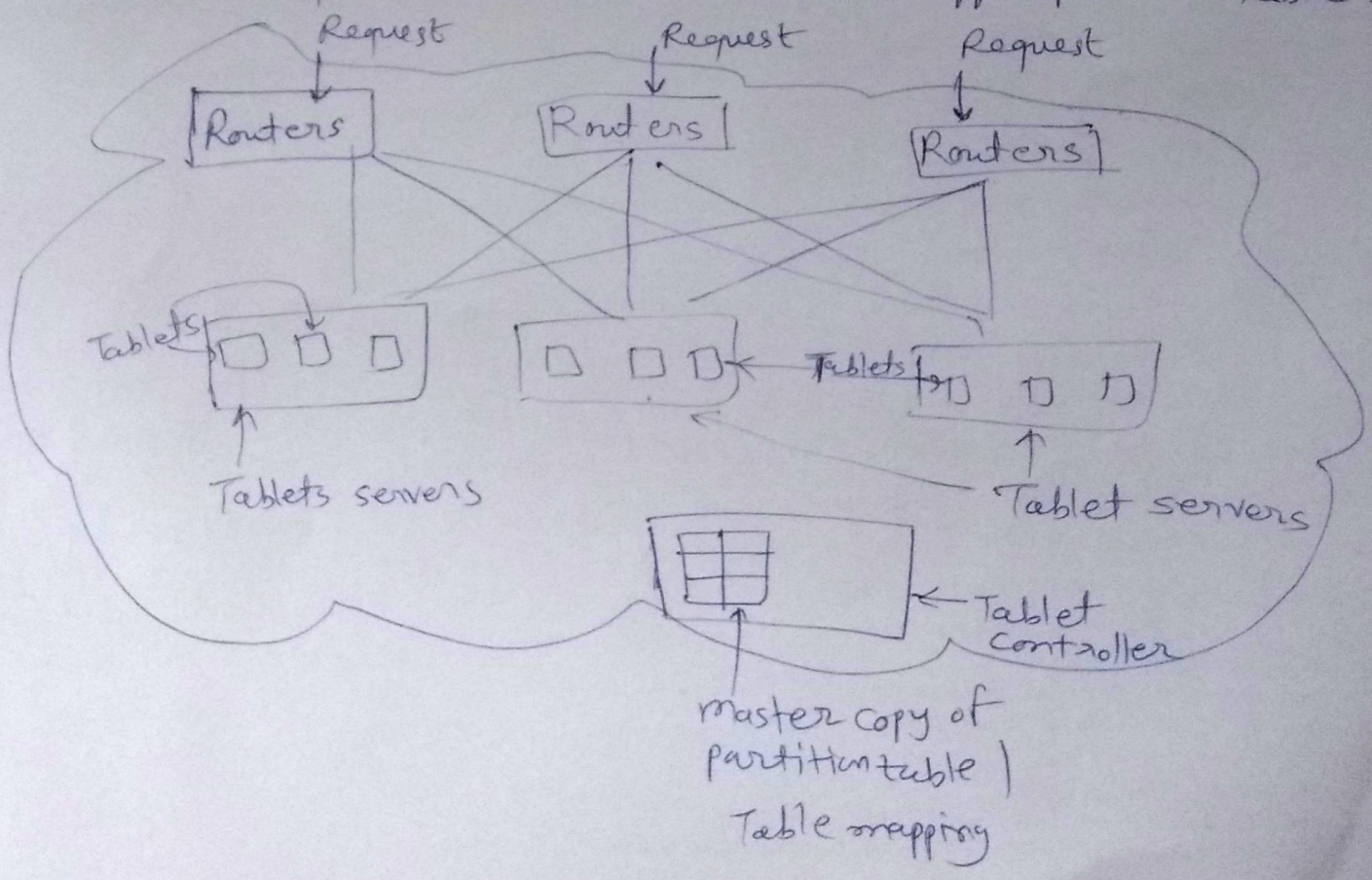
कार्यालय : परबडी के पास, पो.समरखा.
निवास : सरदार पार्क सोसायटी, पो.समरखा, ता.जी.आणंद-388360.
फोन : (नि) (02692) 256673 (मो) 98242 32755.
Offi. : Nr. Parabadi, Po. Samarkha.
Resi. : Sardar Park Society, Po. Samarkha, Ta. Dist. Anand-388360.
Ph. : (R) (02692) 256673 (M) 98242 32755. E-mail : bipinpatellic@yahoo.com

Partitioning and Retrieving Data

→ Problems - extremely large scale in data storage system, load ↑ on one site

Soln - Partition data relatively small units called tablets
- each tablet is a fragment of a table

- The partitioning of data should be done on the search key, so that a request for specific key value is directed to a single tablet; otherwise each request would require processing at multiple sites, increasing the load on the system.
- Range or hash partitioning is used
- A site to which a tablet is assigned acts as the master site for that tablet All updates are routed through this site, and updates are then propogated to replicas of the tablet.
- Partitioning of data into tablets is dynamic. As data are inserted, if a tablet grows too big, it is broken into smaller parts.
- even tablet is not large enough to merit being broken into smaller tablets and distributes over sites.
- Controller site - Tracks the partition function to map a "get()" request to one or more tablets, and a mapping function from tablets to sites., to find which site were responsible for which tablet.
- single tablet Controller site would soon get overloaded. Instead, the mapping information replicated on a set of router sites, which route requests to the site with the appropriate tablet.



19 (6)

LIC
भारतीय जीवन बीमा निगम
LIFE INSURANCE CORPORATION OF INDIA
सदस्य - अधिकारीओ के लिये क्षेत्रिय प्रबंधक क्लब
MEMBER ZONAL MANAGER'S CLUB FOR AGENTS

* Transactions and Replication
- Such system support transactions on data within a single tablet, which controlled by a single master site.
- It update data item to be conditional on the current version number of the data item is more recent than the specified version number.

बिपीनकुमार अेम. पटेल
सदस्य - अधिकारीओ के लिये क्षेत्रिय प्रबंधक क्लब
Bipinkumar M. Patel
Member of the Zonal Manager's Club for Agents
828 83A

→ In system thousands of sites, at any time it is almost guaranteed that several of the sites will be down. A data storage system on the Cloud must be able to continue normal processing even with many sites down
- since each table is controlled by a single master site, if the site fails the tablet should be reassigned to a different site that has a copy of the tablet, which becomes the new master site for tablet
- when site fail new controller use Log and updates, lookups can be performed on the tablet.
- Replication at a remote site is for high availability
- Propogating updates to secondary indices. Secondary indices are basically tables, partitioned just like regular tables, based on the index search key, an update of record in a table can be mapped to updates of one or more tablets in a secondary index on the table.

* Traditional DataBases on Cloud
Virtual machines - user was given the illusion of having private Computer
- It provides great flexibility. User can use their own O.S. & softwares
- Entire computer can be allocated to each virtual machine of a client whose virtual machine have a high load.
- Easy to add or subtract computing power as workloads grow & shrink
- Applications are easily parallelized.

* Challanges with Cloud-Based Databases
- distributed database system require frequent communication and coordination among sites for
(1) access to data on another machine
(2) obtaining lock on remote data
(3) ensuring atomic transaction commit via 2-phase Commit.
- Effective query optimization requires that the optimizer have accurate cost measures for operations.
- Lacking knowledge of physical placement of data, the optimizer has to rely on estimates that may be inaccurate, this issue can have significant impact of performance
- Replication is done by vendor without knowledge of application e.g. It is controlled by cloud not by database.

कार्यालय : परबडी के पास, पो.समरखा.
निवास : सरदार पार्क सोसायटी, पो.समरखा, ता.जी.आणंद-388360.
फोन : (नि) (02692) 256673 (मो) 98242 32755.
Offi. : Nr. Parabadi, Po. Samarkha.
Resi. : Sardar Park Society, Po. Samarkha, Ta. Dist. Anand-388360.
Ph. : (R) (02692) 256673 (M) 98242 32755. E-mail : bipinpatellic@yahoo.com

- Now a days, application itself need to take responsibility for consistency
- Users of cloud computing have security and legal liability problem because data held by another organization
- main problem, when cloud vendor store data in foreign country e.g. if a german company store data in "NY" then U.S. government can see that data without client permission.

* Directory Systems

- Directory is a listing of information about some class of objects (person)
- It can be used to find information about spacific object
- Directories [lookups in forward direction - white pages
  lookups in reverse direction - yellow pages

* Directory Access Protocols

- several protocols are developed to provide a standardized way of accessing data in directory. Most used, is "Lightweight Directory Access Protocal"

why JDBC/ODBC not used ?

① This protocols is simplified, for a limited type of data access, evolved parallel to ODBC/JDBC.

② Directory system provides nice hierarchical naming mechanism
- Data can be partitioned amongst multiple server for different parts of hierarchy.
- directory system can be setup to automatically forward queries made at one site to other site, without user intervention.

* LDAP Data model

→ Directory store entries. Each entry must have distinguished Name (DN)
- A DN is in turn made up of sequence of Relative distinguished names (RDNs)
- DN is combination of a attributes
- The set of RDNs for a DN is difined by schema of the directory system.
- It provides string, binary, time types, tel ← for telephone
- all attributes are multivalued by default.
- LDAP provides object classes with names and types attribute
- Entries are organized into a directory information tree, according to their distinguished names. Entries at leaf level represent specific objects. Internal nodes represent - organization Unit, country. Children of a node inherit the DN of the parent, and add RDNs e.g. internal node with DN C=USA
  Child node have DN starting with C=USA & further RDNs such as "O" or "ou"
- DN of an entry can be generated by traversing path from root.
- leaf level can be pointing another entry
- Entries can have more than one DN

[read from PPT]

LIC
भारतीय जीवन बीमा निगम
LIFE INSURANCE CORPORATION OF INDIA

- Recently, complex data types such as addresses work fine because address could be view as an atomic data item and it is easy to store data.

Complex type system - composite attributes, multivalued
- generalization, specialization

बिपीनकुमार अेम. पटेल
→ Rather than view a database as a set of records,
Bipinkumar M. Patel
Member of the Zonal Manager's Club for Agents
828 83A

Users of applications view it as a set of objects without complex data types translation to relational model.
- These objects require set of records for representation
- Easy-to-use interface requires a one to one correspondence between the user's intuitive notion of an object and the database system's notion of a data item.

Ex Library application
Authors - book may have list of authors, which can represent as an array
Keywords - keywords include one or more specified keywords
Publisher - Publisher consisting name and branch.
- Above all three fields are nonatomic which mean they are no-1NF.
- same system can implement in different way and have - 4NF schema but that design requires queries to join multiple relations, whereas non-1NF design makes many types of queries easier.

Structured Types and Inheritance in SQL
* Structured Types:- user define type

Create type Name as (firstname varchar(20), lastname varchar(20)) final;
create type Address as (street varchar(20), city varchar(20)) nonfinal;
we can now create composite attribute in relation
create table person(name Name, address Address, Dob date);
Components of composite attribute can accessed using "."
e.g. name.firstname
- We can also create a table whose rows are of user define type

Ex Create type personType as (name Name, address Address, dob date);
Create table person of personType;

*[In other way]

Create table person (name row (firstname varchar(20), lastname varchar(20)), address row (street varchar(20), city varchar(20)), dob date);

कार्यालय : परबडी के पास, पो.समरखा.
निवास : सरदार पार्क सोसायटी, पो.समरखा, ता.जी.आणंद-388360.
फोन : (नि) (02692) 256673 (मो) 98242 32755.

Offi. : Nr. Parabadi, Po. Samarkha.
Resi. : Sardar Park Society, Po. Samarkha, Ta. Dist. Anand-388360.
Ph. : (R) (02692) 256673 (M) 98242 32755. E-mail : bipinpatellic@yahoo.com

- Here name, and address are unnamed Type and row of table also unnam

structured type can have methods define on it.

```
create type PersonType as (name Name, address Address, dOB date)
                    notfinal
                    method age(Date date)
                                returns interval year;
```

* method

```
create instance method age(Date date) returns interval year
                                for PersonType
begin
    return Date - self.dOB;
end
```

→ This method executes on an instance of the PersonType.
→ The variable "self" refers to the person instance on which the method is invoked.

* After Person table created, fetch age and last name

```
Select name.lastname, age(current_date) from person;
```

* create Constructor for structured Type

```
create function Name(firstname varchar(20), lastname varchar(20))
returns Name
begin
    set self.firstname = firstname
    set self.lastname = lastname
end
```

→ By default every structured type has a constructor with no arguments
- There can be more than one constructor for same structured type although they distinguishable by the number of arguments and types of their arguments

```
insert into person values (new Name('mit', 'Patel'), Address("main",
                                'anand'), date '1999-08-12');
```

(called function)

* Type Inheritance

- student and teacher also persons so both have same attribute hence person type can be inherited

```
create type Student under Person (degree Varchar(20))
create type Teacher under Person (salary integer);
```

- Here methods of structured type are inherited by its subtypes.
- declaring method can be override using "overriding method"
- Final says that subtype may not be created
  nonfinal subtype may be created

* Here we can inherit student and teacher both simultaneously

```
create type TechingAssistant under Student, Teacher
```

* If student & Teacher both have department attribute than problem may arise. Though name, address are also available in both but they are not cause any problem because of both are inherited so

```
create type TechnigAssistant under Student with (department as stu-depart),
Teacher with (department as Tea-depart)
```

- each value must be associated with one specific type called its most-specific type.

* Table inheritance

```
create table people of Person;
create table student of Student under people;
create table teacher of Teacher under people;
```

→ all person's attributes inherited into teacher and student table.

```
delete from people where name = "mit"
```

→ This will affect all three table because people is supertable and Student and teacher are subtable - all tuples will be deleted

```
delete from only people where name = "mit"
```

- only people will alter

→ multiple inheritance of table is not support by SQL

e.g. create table x of x1 under P1, P2

बिपीनकुमार अेम. पटेल
सदस्य - अधिकर्ताओ के लिये क्षेत्रिय प्रबंधक क्लब
Bipinkumar M. Patel
Member of the Zonal Manager's Club for Agents
828 83A

कार्यालय : परबडी के पास, पो.समरखा.
निवास : सरदार पार्क सोसायटी, पो.समरखा, ता.जी.आणंद-388360.
फोन : (नि) (02692) 256673 (मो) 98242 32755.

Offi. : Nr. Parabadi, Po. Samarkha.
Resi. : Sardar Park Society, Po. Samarkha, Ta. Dist. Anand-388360.
Ph. : (R) (02692) 256673 (M) 98242 32755. E-mail : bipinpatellic@yahoo.com

- tuples in a subtable and parent table correspond if they have the same values for all inherited attributes. Thus, corresponding tuples represent the same entity.

* The consistency require for subtables are
① Each tuple of supertable can correspond to at most one tuple in each of its immediate subtables.
② all the tuples corresponding to each other must be derived from one tuple.

→ without first condition two tuples in students
- since sql does not support multiple inheritance, it prevents a person from being both a teacher and a student.

* Array and Multiset Types in SQL

- Create type Publisher as (name varchar(20), branch varchar(20));
Create type Book as (title varchar(20), author varchar(20) array[10], publish Publisher, keyword varchar(20) multiset);
Create table b of Book;

*Creating and accessing Collection Values

insert into b values ('compiler', array['mit','smith'], new Publisher('mc', 'NY'), multiset['parsing', 'analysis']);
→ Assume Publisher constructor (created explicitly)

* Querying Collection-valued Attributes

select title from b where 'Parsing' in (unnest(keyword));
Select title, A.author, A.position from b, unnest(b.author) with ordinality as A (author, position);
→ "with ordinality" extra attribute return - position of the element in the array

* Nesting and Unnesting

select title, A.author, publisher.name as pub-name, publisher.branch as pub-branch, K.keyword from b, unnest(b.author) as A(author), unnest(b.keyword) as K(keyword);

output is 1NF
↓nest

select title, author, Publisher(pub_name, pub-branch), collect(keyword)
from books groupby title, author, publisher

(02)
select title, array(select author as A where A.title = B.title), Publisher(pub-name, pub-branch), multiset(select keyword from keyword as k where k.title = B.title)
from books as B

LIC
भारतीय जीवन बीमा निगम
LIFE INSURANCE CORPORATION OF INDIA
MEMBER ZONAL MANAGER'S CLUB FOR AGENTS
बिपीनकुमार अेम. पटेल
सदस्य - अभिकर्ताओ के लिये क्षेत्रिय प्रबंधक क्लब
Bipinkumar M. Patel
Member of the Zonal Manager's Club for Agents
828 83A

Create type Department (name varchar(20), head ref(Person) scope people);
(datatype, table)
- Here scope work like foreign keys. We can omit scope people;
Create table department of Department (head with options scope people);

* self-referential attribute

create table people of person
ref is personid system generated;

In order to initialize a reference attribute, we need to get the identifier of the tuple that is to be refered. we may first create the tuple with a null reference and then set the reference separately:

insert into department value ('cs', null);
update department set head = (select p.personid from people as p where name = 'mit')
where name = 'cs';

Alternative to system generated identifiers is to allow users to generate

create type Person (name varchar(20), address varchar(20)) ref using varchar(20);
Create table people of Person ref is personid user generated;

insert into people ~~(ref, address)~~ (personid, name, address)
values ('123', 'mit', 'NY'));
insert into department values ('cs', '123');

→ It is possible to use an existing primary key value as the identifier, by including the ref from clause in the type definition:

create type Person (name varchar(20) primary key, address varchar(20))
ref from (name);
create table people of person ref is personid derived;
insert into department values ('cs', 'mit');

→ Reference are dereferenced by the "→" symbol

select head → name, head → address from department;

कार्यालय : परबडी के पास, पो.समरखा.
निवास : सरदार पार्क सोसायटी, पो.समरखा, ता.जी.आणंद-388360.
फोन : (नि) (02692) 256673 (मो) 98242 32755.
Offi. : Nr. Parabadi, Po. Samarkha.
Resi. : Sardar Park Society, Po. Samarkha, Ta. Dist. Anand-388360.
Ph. : (R) (02692) 256673 (M) 98242 32755. E-mail : bipinpatellic@yahoo.com

→ "dref" return the tuple pointed by a reference and then access it's attributes

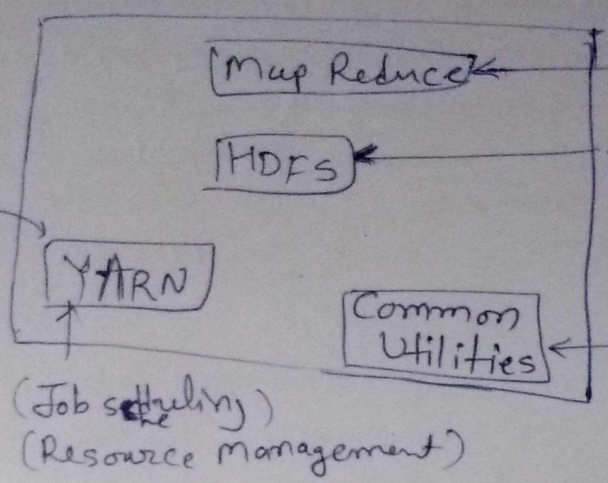
select dref(head).name
from department;

Object-Relational Mapping

- It is built on top of a traditional relational database, and allow a programmer to define a mapping between tuples in database relations and objects in the programming language.
- An object, or set of objects, can be retrieved based on a selection condition on its attributes; relevant data are retrieved from the underlying database based on the selection conditions.
- Mapping from objects to relations is used to correspondingly update, insert or delete tuples in the database
- main goal of mapping is proving object model, benefits of using a robust relational database underneath.
- This system provide significant performance gains over direct access to the underlying database
- This mapping also provide query languages that allow to write query for direct object model; such queries are translated into SQL and result object created from SQL query results.
- It suffer from significant overheads for bulk database updates, and may provide only limited querying capability.
- Benefits of mapping exceed the drawbacks for many applications.

* Hadoop

LIC
भारतीय जीवन बीमा निगम
LIFE INSURANCE CORPORATION OF INDIA
बिपीनकुमार अेम. पटेल
सदस्य - अभिकर्ताओं के लिये क्षेत्रिय प्रवंधक क्लब
Bipinkumar M. Patel
Member of the Zonal Manager's Club for Agents
828 83A



• सदस्य • अभिकर्ताओं के लिये क्षेत्रीय प्रबंधक क्लब • MEMBER • ZONAL MANAGER'S CLUB FOR AGENTS

- Hadoop follows a master-slave architecture using HDFS and Map-Reduce respectively
- A file on HDFS is split into multiple blocks and each is replicated within the hadoop cluster. A default block size is 64 MB. It can be extended up to 256 MB
- Namenode and DataNode are two critical components of the Hadoop HDFS architecture.
- Application data is stored on DataNodes & File system metadata is stored on NameNode.
- The NameNode and DataNode communicate with each other using TCP based protocols.

HDFS must satisfy pereequisities-

1. All hard drives should have a high throughput
2. Good network speed to manage intermediate data transfer and block replications.

* NameNode

- all the files and directories in namespace are represented by Inodes that contain various attributes like permissions, modification timestamp, disk space quota and access times.
- Two files fsimage and edits are used for persistence

Fsimage file contains the Inodes and list of block which define the metadata. It has complete snapshot of the file systems.

edit file contains any modifications that have been performed on the content of the fsimage file.

- Create new fsimage snapshot everytime the namespace is being altered.
- when NameNode starts, fsimage file is loaded and then the contents of the edit file are applied to recover the latest state of the file system.

Problem is file grows and consume more space, slow down running.

कार्यालय : परबडी के पास, पो.समरखा.
निवास : सरदार पार्क सोसायटी, पो.समरखा, ता.जी.आणंद-388360.
फोन : (नि) (02692) 256673 (मो) 98242 32755.
Offi. : Nr. Parabadi, Po. Samarkha.
Resi. : Sardar Park Society, Po. Samarkha, Ta. Dist. Anand-388360.
Ph. : (R) (02692) 256673 (M) 98242 32755. E-mail : bipinpatellic@yahoo.com

- If the hadoop cluster has not been restarted for months together then there will be a huge downtime as the size of the edits file will be increase. This is when Secondary NameNode comes to the rescue. Secondary NameNode gets the fsimage and edits log from the primary NameNode at regular intervals and loads both the fsimage and edit logs file to the main memory by applying each operation from edits log file to fsimage.

* DataNode

- A DataNode can perform CPU intensive jobs like semantic and language analysis, statistics and machine learning tasks and indexing. A DataNode needs bt of I/o for data processing and transfer.
- on startup every DataNode connects to the NameNode and performs a handshake to verify the Namespace ID and software version of the DataNode. If either of them does not match dataNode shutdown.
- DataNode sends heartbeat to the NameNode every 3 seconds to conform that the dataNode is available

* MapReduce

- The execution of a MapReduce job begins when the client submits the job configuration to the job Tracker that specifies the map, combine and reduce functions along with the location for input and output data.
- Job tracker identifies the number of splits based on the input path and selected task trackers based on their network vicinity to the data sources. Job tracker send signal to selected Task trakers.
- After the complition of work task trackers notify job tracker. when all task trackers are done, job tracker collect all work

Hadoop	RDBMS
→ Used for structured, semistructured and Unstructured data	→ mainly for structured data
→ Use for large dataset	→ Average size dataset
→ Hive query Language	→ Structured query language
→ Required schema on read (dynamic)	→ required schema on write (static)
→ Reads writes are fast	→ Reads are fast
→ free	→ License
→ Throughput High	→ Throughput low
→ work on key/pair value (Data objects)	→ works on Relational Tables
→ Horizontal scalability	→ Vertical scalability
→ Integrity low	→ Integrity High
→ Commodity/Utility Hardware	→ High end servers
→ Analytics, Data Discovery	→ online transactin processing