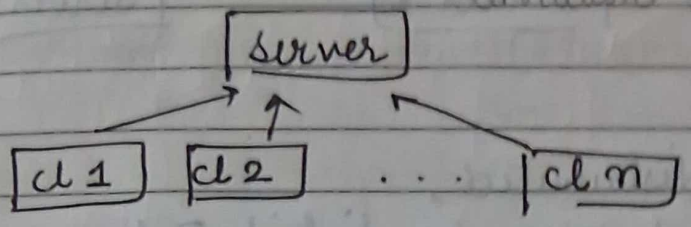
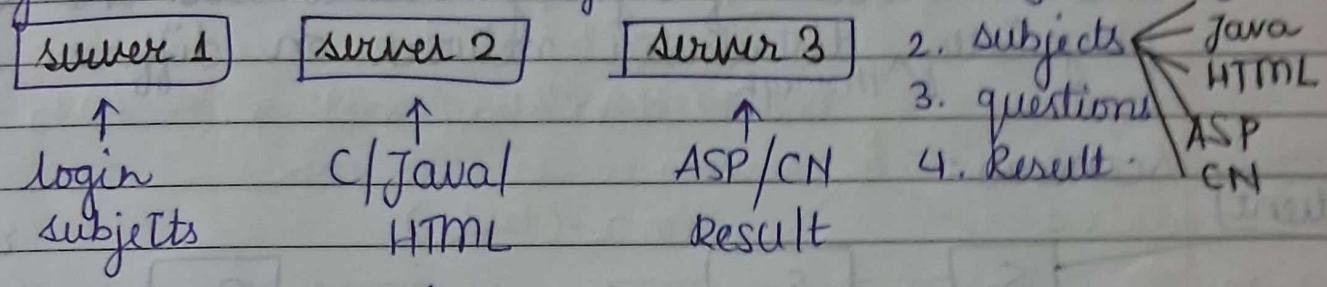
- Centralised database (Handles data at 1 server)
  - client
  - server

  (it has a backup server)
  (If server fails backup server acts as centralised server like archive)

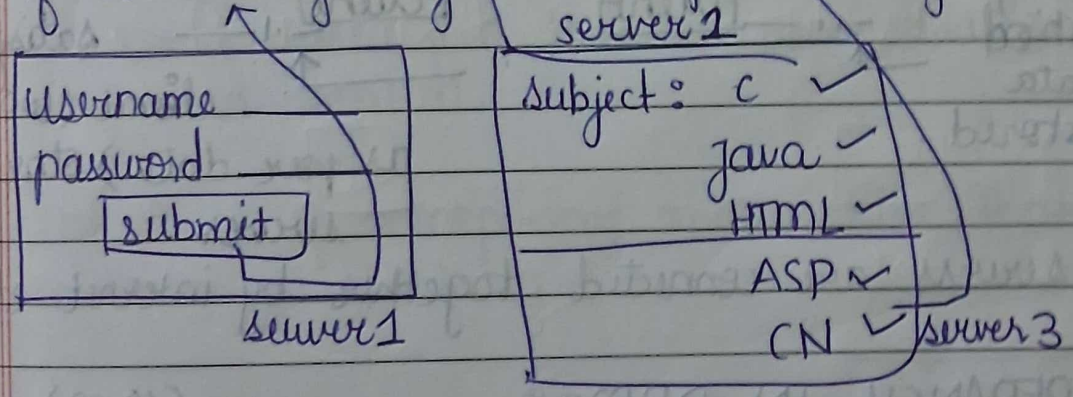


- Distributed database (divided servers for work load balancing)

eg: Online examination system →
1. login
2. subjects
3. questions
4. Result

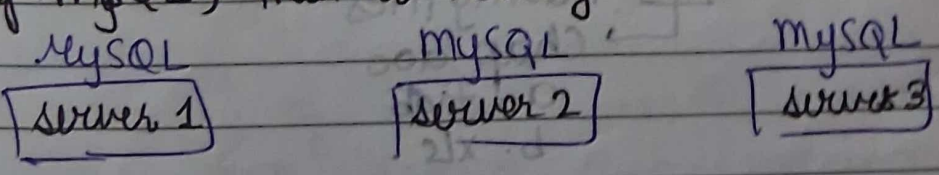


If we are loginning then compiler will go to server 1.



Imp: (Homogeneous & Hetroge) – Types of DDBMS.

If same database (DBMS) is used in all servers that is (eg: MySQL) then it homogeneous DBMS.



- If there is only one server, response time is more. Hence we use distributed database on multiple server.

The DDBMS in which there are different DBMS installed on different servers.

Mongodb

If there is no exchange of data betn the servers then we can use different DBMS called heterogeneous DBMS.

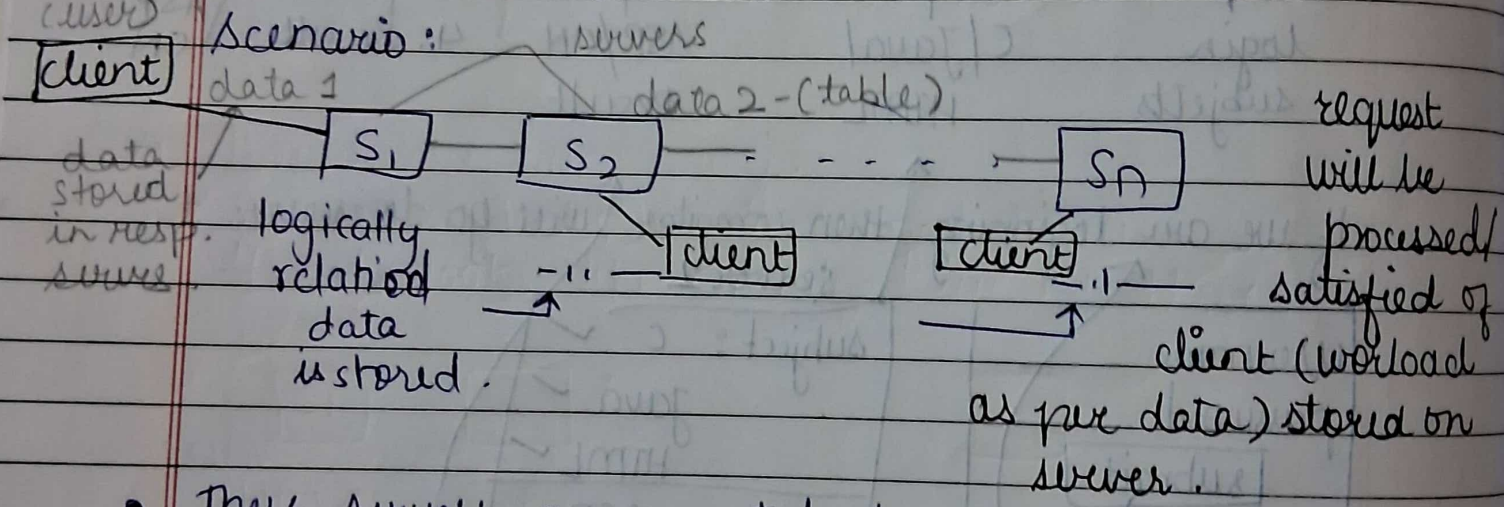
mysQL	oracle	SQL server
server 1	server 2	server 3

- As database is distributed, processing, load will be distributed.

- DDBMS : Distributed DBMS. (2m)

It is the collection of logically related data that is shared physically through distributed network on different servers. (where servers are connected)

Scenario :



- These servers are connected together by internet etc.

* TRANSPERANCY IN DBMS : (10m)

Consider folder abc

having files
- Chap.doc
- a.txt
- b.xls.

- Copying all the folder is called - duplication
- — " — some part of the folder, some file - replication

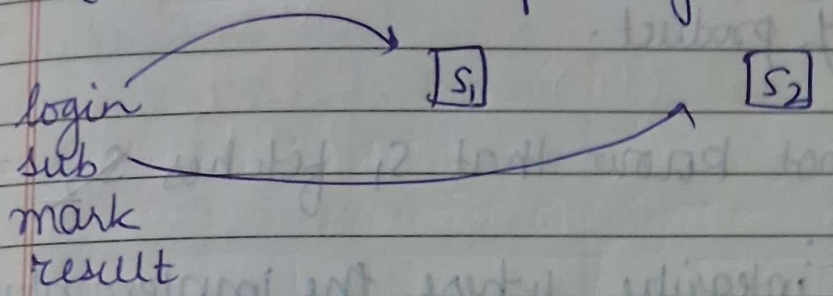
Explain eg.

2. Explain eg.

- It is hiding implementation details from users of an application of an project.

Types:

1. Distribution transparency: (Location)
User will not know where which data is stored called distributed transparency.

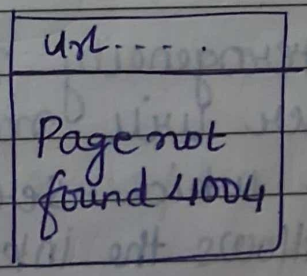


Here, user doesnot know where login is present in which server which is called distribution transpara

2. Failure transperancy:
Hiding the details of replacement system.
If one system fails then the user will not ~~the~~ about the replaced system.

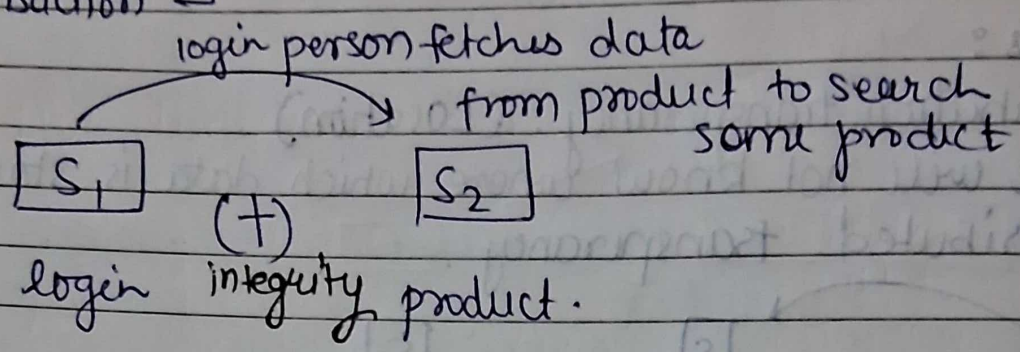
This type of transparency makes sure that system is continuosly working.

eg: webpage →



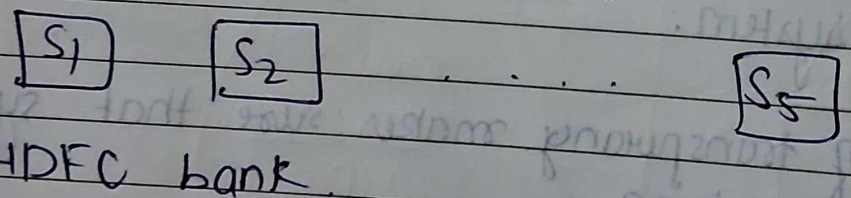
The ~~recovery~~ details of recovery server is hidden.
Hidding imp. details of recovering failded server.

3. Transaction transperancy:
→ (no. of programs that will be executed by GUI's.)
Transaction →



The user will not know that S_1 fetches S_2 product from S_2.
Hence this is integrity where the implementation details of transaction are not known to the user.

4. Performance transperancy:
The system looks like centralized DBMS.
It only looks like centralized but not there.



5. Heterogeneity transperancy:
User will not know which data is stored in which server.
It allows the integrity of several different local DBMS under common schema.

6. Replication
Copying of some data on the servers.
We dont know which data is copied where.
User doesn't

. 7 Fragementation Transperancy:
Division of data on no. of servers.
Fragementation types:
1. Horizontal
2. Vertical.

1. Horizontal : There are more than 1 horizontal fragments depending on conditions.

Student.

Rollno.	Name	Add.	DIV
1			A
2			.
3			.
.			
.			A
.			
.			B
.			A.

DIV A.

Serv. 1 | serv2

Division of data on logicall inp. div-A.

select * from stu.
where DIV = A.

only divA will be selected and placed in server A.

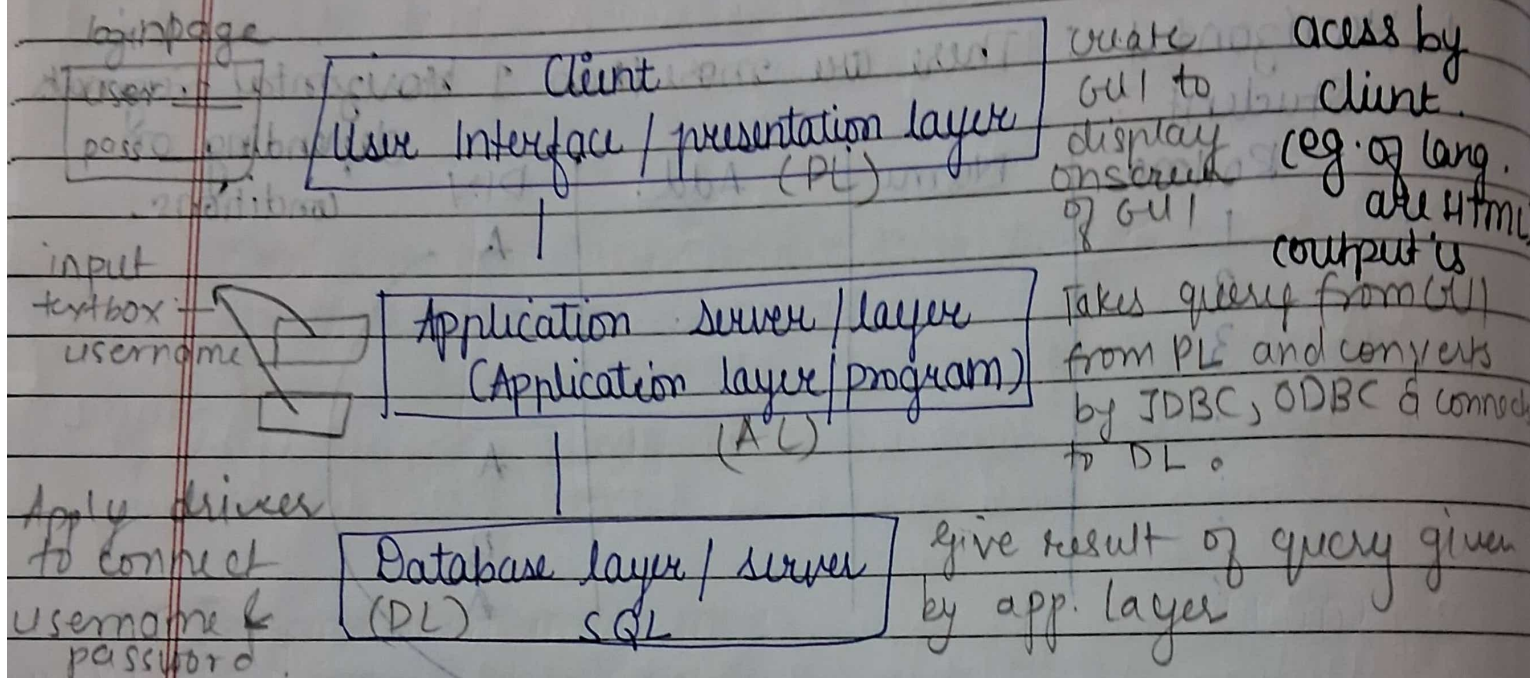
2. Vertical :
Division of data based on columns/attribute.

Server 1		Server 2	
Rollno.	Name	Add	DIV

Q 3 Tier Arch.

This type of transperancy is which where user does'nt know where fragments are created on table and where they are stored.

• 3 tier client server architecture:



1. Client
2. Server
3. Dia
4. eg. of each layer

Output is generation of presentation by app. layer & Application layer takes input from presentation layer.

(Relation, exp., eg., app of each layer & explain client & server) (10m)

- Data replication & allocation : (5M)

REPLICATION:

→ It is a technique of copying of some part of data or fragments.

ALLOCATION:

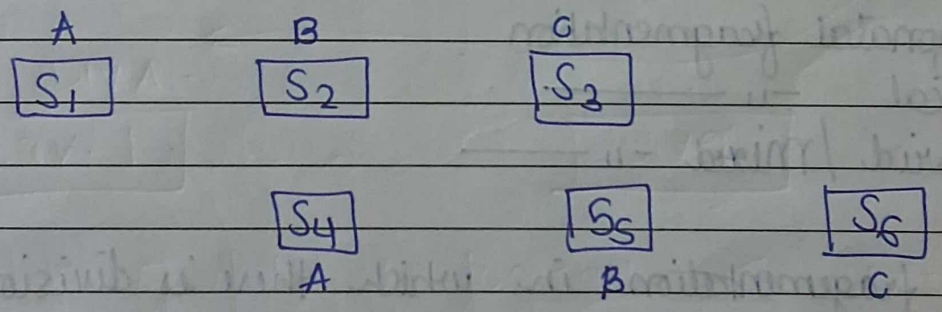
→ It is the way of storing fragments or replicas in server.

- Fully replicated distributed database:

It is db is which there will copy of each data in different servers.

This is the most extreme option in which replication of whole database will at every server in case of distributed system (distributed database).

Assume A, B, C as fragments that are stored on S_1, S_2 & S_3 resp., then in fully distributed database A, B, C will have a replication S_4, S_5, S_6.



- No replication:

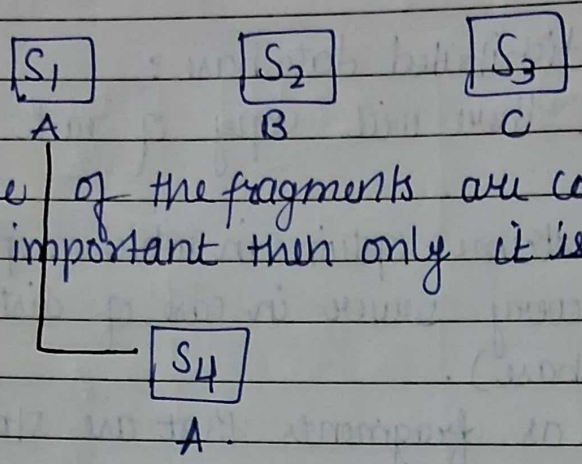
In this case, there will no copies of the fragements on the available servers.

These fragements will not have copies in another servers.

Above case there were copies of A, B, C in S_4, S_5 & S_6 but in this case there will be no copies in S_4, S_5 & S_6.

- Partial replication:
In this type of data allocation there will be copying / replication of some parts of ~~sv~~ fragments.
In this case important data will be copied on server.

assume A, B, C on S_1, S_2, S_3.

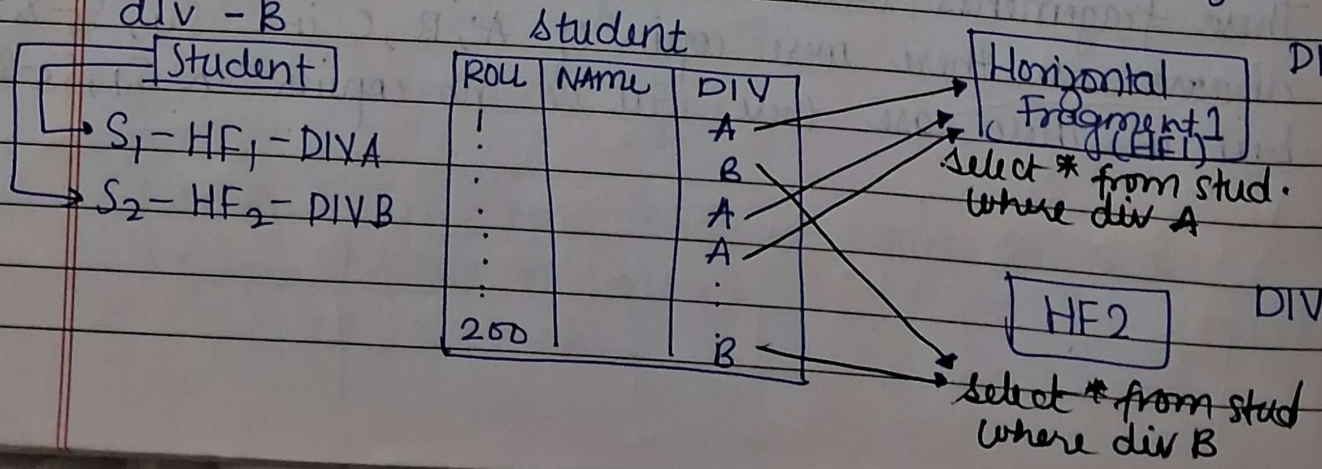


Only some of the fragments are copied.
If A is important then only it is copied.

IMP. Fragementation in DDBMS: (10M)
1. Horizonatal fragmentation
2. Vertical —"—
4. Hybrid / Mixed —"—

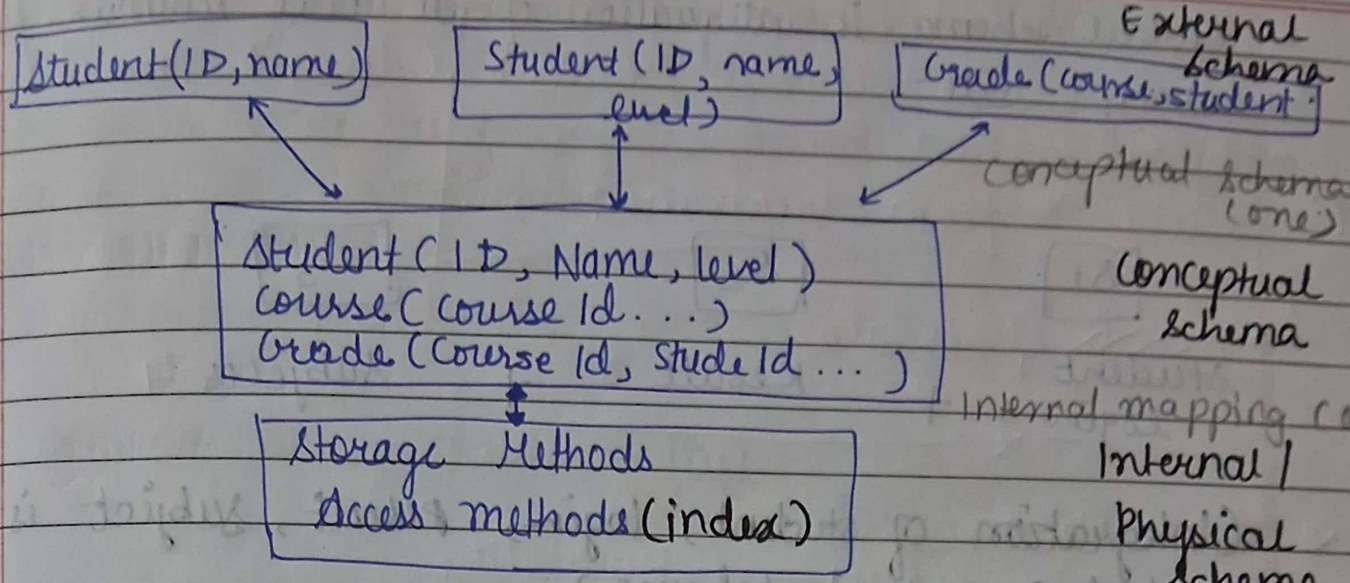
1. This is fragementation in which there is division / distribution of tables on rows / records horizontally by applying some condition of the attributes.

Representation:
Condition on division attribute there are two fragementation
div - A
div - B



10:30am
Friday
28/07/2023

GENERAL ARCHITECTURE OF DDBMS:



eg:

DBMS
Student table
- Roll no.
- name
- add
- email.Id

} sequence should be same.

GUI
Rollno.
name
email.Id

there is no need for all attributes from DBMS.
add in DBMS will have 'NULL value'
If required 'add' is added.

- Conceptual schema types:
  - Global schema:
    It is the general architecture of distributed DBMS or a global logical veiw of data / project / app.
  - Local conceptual schema:
    It is the logical data representation at every server / site.

- Global schema is integration of all local schemas from every site.

eg:

S_1 : student 1, login 2 +

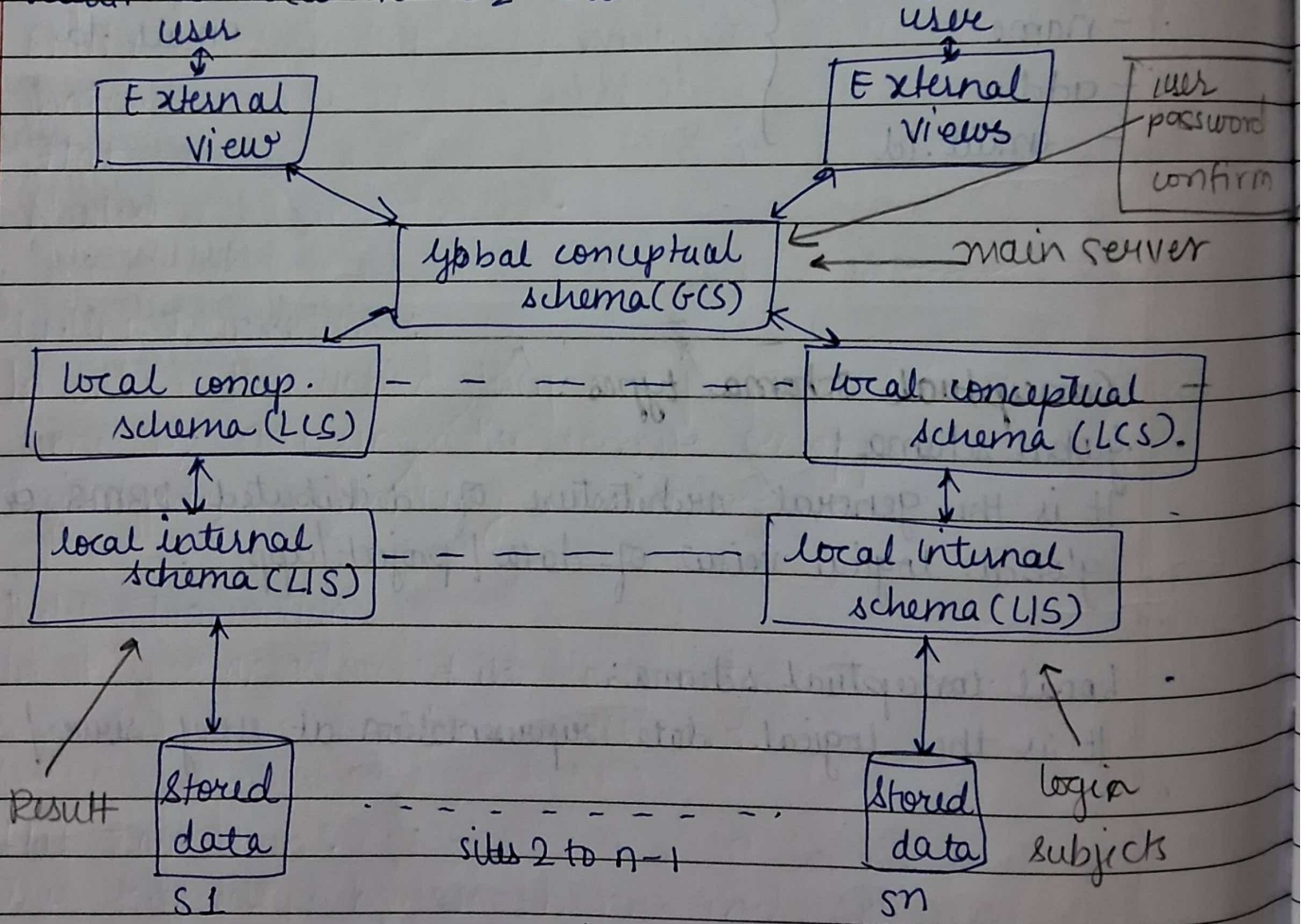
S_2 : Result 3 +

S_3 : subjects 4

(Integration of student, login, result, subject is global conceptual schema).

[Global conceptual schema - student + login + result + subject]

- login is local host to S1 but remote to S_2 & S_3.
Result is local to S_2 but remote S_1 & S_3.



Logical architecture model

2/08/23

MODULE: 2

Derive horizontal fragmentation (5m)

Fragmentation

In which table is fragmented based on constraints on another table (owner table)

owner table → PK

member table → FK

1. Consider owner table = Pay
   (title, salary) → Attribute

2. member table = emp
   (child)
   ↓
   (eno., ename, title) Attributes

3. HF Fragmentation of owner table / parent table:

- There are two groups of engineers that are divided as per salary ranges as sal > 30,000 and sal < 30,000.

3.

Fragments on Owner Table

$Pay_1 = \sigma sal > 30000$ & $Pay_2 = \sigma sal < 30000$

4. EMP_1 & EMP_2 are fragments on emp table (member table) with HF of owner table

semijoin

fragments of owner table

$EMP1 = EMP \ltimes Pay1$

$EMP2 = EMP \ltimes Pay2$

2HF on Pay

semijoin

- Semijoin:

It is the type of join in which result will contain the columns from one of the join table.

5. O/P: DHF's from HF's of owner table

View & Fragments: (doesn't store data)
(It is virtual network / table)
Syntax:
select * from viewname;

doesn't stores data

---

* Transaction:
Properties: ACID — Durability
Atomicity consistency — Isolation

Types of transaction:
1. Online (Performed on daily basis in short time) eg: Gpay, ecommerce site
2. Batch
3. Flat
4. Nested

1. Online Transaction:
In online transaction they are performed in a very short time by accessing the website or apps. (Performed on daily basis)
eg: Gpay, netbanking, ecommerce site

2. Batch Transaction:
In this, transactions take longer time to execute.
eg: Report generation, credit card (monthly bills of customers → with dates will be shown)
(bills will be generated in a batch)

3. Flat transaction:
This transaction has one begin and one end that is only one ~~trasn~~ transaction at a time.
eg: bank (verification by OTP)

Q Explain 2 phase locking protocol with types. (10M)

4. Nested Transaction :

Nested is opposite of flat transaction. Hence there will be more than one begin & one end. Here than one transaction at a time is performed.

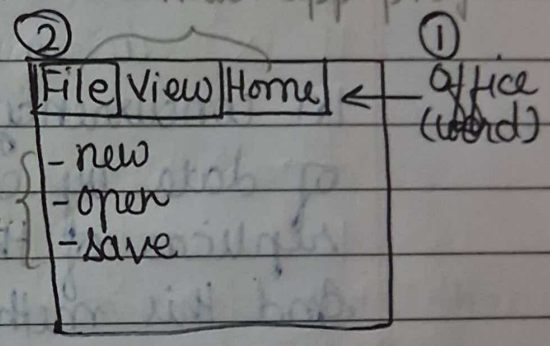
Syntax :

```
Begin Transactions
  Begin
    .
    .
    .
  end
end
```

- No. of transaction will be included to stisfy the requirement of user & the client.

eg : word document

(New → open → save)



* CONCURRENCY CONTROL :

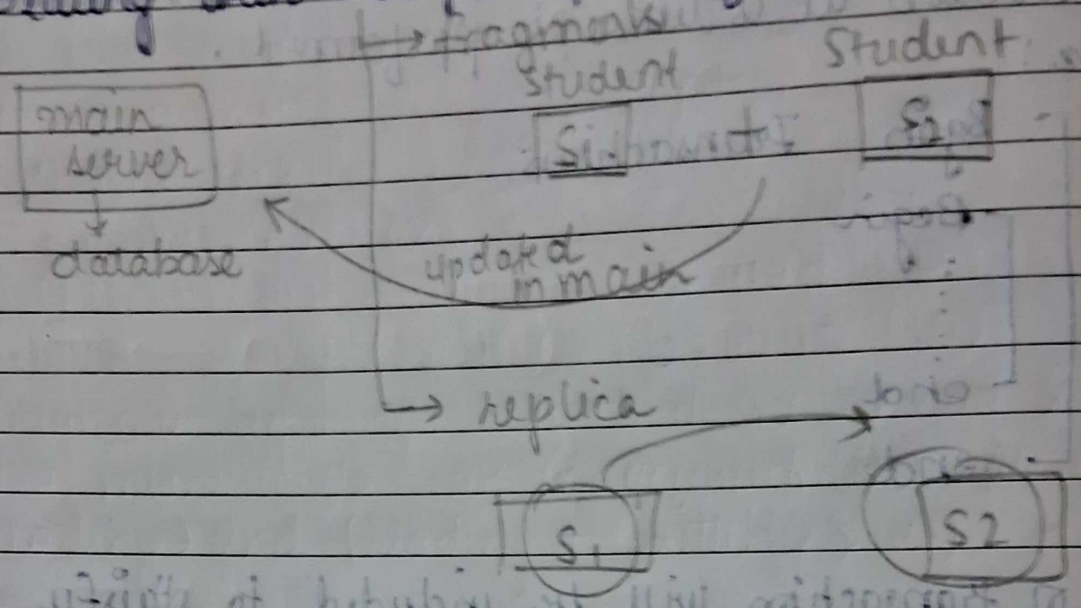
When no. of users are increased for same project / application. Concurrency control is mechanism to make the application consistent.

explain plan for select * from a join b on a.id = b.id ; } inner join.

Vertical = same
Horizontal = column = same
row = union

Q. Discuss issue in case of distributed concurrency control

1. → Dealing with multiple copies of data:



If S1 is updated S2 will also be updated

In concurrency control will deal with multiple copies of data by checking with whether it is fragments or replicas of the tables.
And this method insures that updations are done at every copies of replica as well as in main database.

2. → Failure of individual sites:

In failure of fragments,
it will be switched to main server.

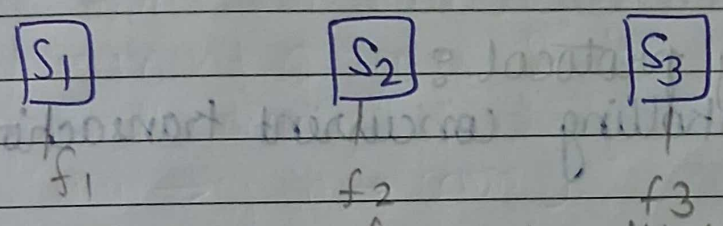
In case of replicas,
then it is switched to next working replica.
If R_1 fails
R_2 is activated.

In case of failure of individual sites the distributed dbms will manage continuous execution of the project with the help of replicas or main database.

3. → Failure of communication links:

In case of failure of communication links it should deal with failure - " _____ by using network partioning. (Alternate options will be used in failure of comm. link of networks).
network issue

4. → Distributed commit:



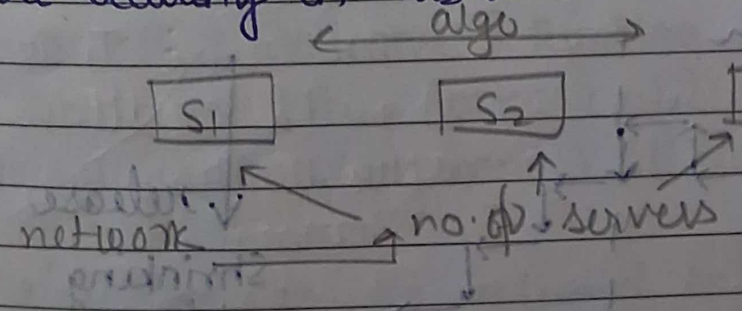
Each & every fragment will be updated due to distributed commit.

Distributed commit will be followed to make sure that each and every copy of fragments or replicas will be updated with current status of data.

* If fail → rollback
* If not → process ahead.

5. → Distributed deadlock: to solve

Concurrency control should be able the deadlocks that are occuring in distributed network of deadlock.



04/08/2023

* 2 phase locking protocol & its types:

- A lock is process / mechanism to deal with concurrent execution or parallel execution of data.

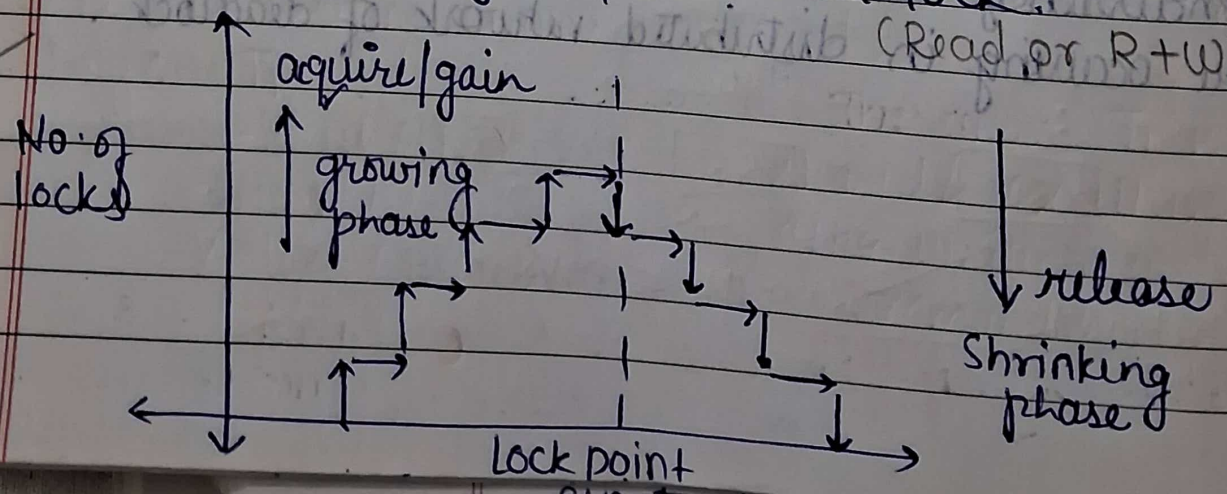
- Types of lock:
  1. Shared lock (Read only) (S) (Read)
  2. Exclusive lock (X) (Read + Write)

  ways of accessing data or table (options)
  1. read (R)
  2. write (W)

There are types of lock that are shared & exclusive that means data can be accessed in 2 ways for (reading) and (reading + writting) respectively.

- Two phase locking protocol:
  It is way of controlling concurrent transaction.
  → 2 Phase:
  1. Growing phase:
     The phase in which transactions are allowed to obtain or gain the locks.
  2. Shrinking phase:
     Transactions will released of the locks. (unlock)

- LOCK POINT:
  The point where growing phase stops. No transactions are allowed to gain / obtain a lock. (Read or R+W)



2 Phase lock:

eg:

Sr no.	T_1	T_2
1	Lock S(A)	
2	.....	lock S(A)
3	..... Lock X(B)	
4		
5	..... Unlock (A)	
6	.....	lock X(C)
7	..... Unlock(B)	
8	.....	Unlock (A)

growing phase for T_1

shrinking phase for T_1

Shared lock (Reading only)

Exclusive lock (Read + Write)

growing phase for T_2

lock point

shrinking for T_2

For T_1: Consider transaction T_1; lock type (data)

T_1 is accessing data in shared lock A.

Srno. (1,2,3) is growing phase of T_1.

For T_1, lock point is at sr. no 3.

Sr. no. (5,6,7) is a shrinking phase for T_1.

For T_2:

Sr. no. (2 to 6) is growing phase for T_2.

Sr. no. 6 is lock point.

Sr. no. 8 is shrinking phase/point.

- Types of 2PL: Types (explain with example)

1. ~~Straight 2PL~~ Strict 2PL:
2. Rigorous 2PL:
3. Conservative 2PL.

1. Strict 2PL: Exclusive & Reading.

It holds all the locks until the commit point and release (unlock/uncommit) all locks at a time.

eg: (exclusive locks)

Read lock can be released before commit. S(A)

Read & write lock only is realiased after commit. X(A)

A = exclusive
B = shared

eg:	T_1	T_2	T_3
1.	lock X(A)		
2.	Read (A)		
3.	lock S(B)		
4.	Read (B)		
5.	Write (A)		
6.	Unlock (A)	lock X(A)	
7.	Unlock (B)	Read (A)	
8.		Write (A)	
9.		Unlock (A)	lock S(A)
10.			Read (A)
11.			Unlock (A)

commit (for rigorous)

eg: Strict PL:

lock S(A)
:
lock X(B)
unlock (A)
commit
unlock (B)

reading is released before commit.

only after commit

2. Rigorous PL:

In this type of 2PL are locks are held (shared + exclusive) till commit. That means if transaction is holding locks in shared and exclusive mode then lock will not be released till the moment transaction is not writing commit command.

eg: T_1

S-lock (A)
read (A)
X-lock (B)
read (B)
write (B)
commit
unlock (A)
unlock (B)

The above T_1 is realeasing shared & exclusive locks after writing commit command. (read) & (read + write locks.)

Q Explain types of 2PL with example & explain example.
Q Compare two PL & rigorous & strict 2PL with eg.
Q Explain timestamp ordering protocol. (10m)
for concurrency control method →

3. Conservative 2PL:
In this, all locks that Transaction will need will be aquired before start of execution. There will be starvation for other transaction.

eg:
T_1
S-lock(A)
read(A)
X-lock (B)
read (B)
write (B)
commit.

predict locks or no. of tables & will hold till commit.
Thefore, there is starvation for next transaction.

The lock will not be released ~~unit~~ until required.

* Types of failure in distributed dbms:

1. Transaction failure:
This is due to
- deadlock
- unavailability of system resources (data, resources)
- logical errors.

This failure is happening due to the application not going ahead.

2. System crash:
Reson
- no power supply
- network traffic

3. Disk Failure:
- bad sector (corrupted)
- unavailability of disk.

/08/2023

→ If there is conflict

- TIMESTAMP ORDERING PROTOCOL:

It is time associated with transaction (TS).
Time can be coded starting with '01' or systime.

This is represented as TS (transaction).

$$TS(T_1) < TS(T_2) < TS(T_3) < \ldots\ldots TS(T_n)$$

Rule 1: If transaction T_i issues / performs Read (X) operation:

(newer) ↑ ith transaction

a) If $W_TS(X) > TS(T_i)$ (older) → this means $TS(T_i)$ is a older transaction but this is not true.
(write / updation) (performing transaction)

It is a timestamp of transaction when updation is done by that transaction at a particular time

This operation will rejected that is there will be rejection of read operation issued by transaction.

b) If $W_TS(X) \leq TS(T_i)$
~~(newer~~ (older) newer (This is to be performed so it should be new)

This will be accepted.
There will be acceptance of read / write operation issued by Transaction.

Rule 2: If transaction T_i issues write (X) operation:

9:00am 9:03am
older

a) If $TS(T_i) < R_TS(X)$ → already read so no updation to the old transaction.
then operation is rejected or not updated.

10:40am W → (last write to X) 11:00am

b) If $TS(T_i) <\cancel{>} \cancel{R}_TS(X)$
then operation accepted.

29/08/2023

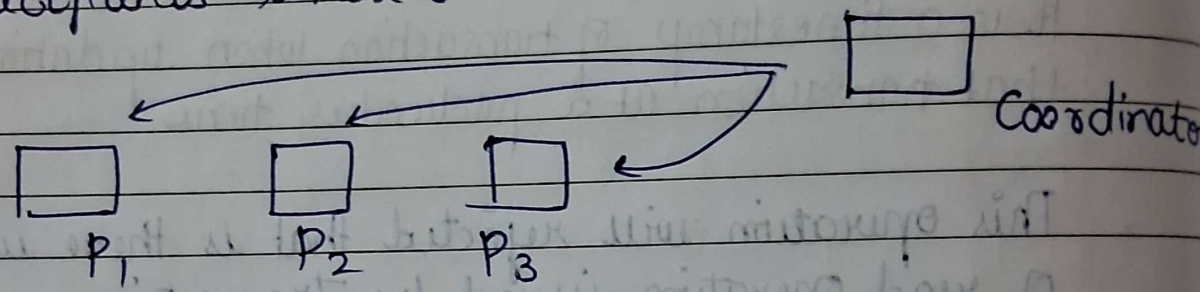
- 2PC - 2 phase commit protocol:

In this protocol, there are two phases to execute, commit command by the number of the sites.

- There are two phases:
  1. Prepare
  2. Commit

- There are two sites: 1. Coordinator & 2. Participant sites.

1. Coordinate: It will take the decision for no. of participants where data is stored.



Coordinator will make sure that none of the sites participants fail in working.

It is the site that will coordinate with all the participants to check whether to commit or abort.

2. Participants sites are the sites that are working on data, that will be having the no. of fragments of data & will respond to the query.

1. Prepare: After each participants will completing the transactions will send done message to co-ordinator site / controlling site.
All participants are sending done message. Therefore, it is preparation of participants site.

2. Commit:
In this phase, after the coordinator site recieves a message as ready from all the participants sites ~~with~~, where coordinator site will now send global commit to all participants. After completing commit command by the participants will send 'commit Ack' to controlling sites.

If any one participant doesnot send commit or done message the abort command is used without change & have a rollback. Also 'abort Ack' will be send.

- 3PC : 3 phase commit protocol:

This protocol consists of three phases to execute global commit or global abort by coordination of participants & coordinator sites. The commit phase is divided into two phases to improve the performance. In this phase

Phase 2 { the coordinator site will ask all the participants whether to abort or commit.
pre commit phase: ~~Decision making phase~~

Phase 3: Decision making phase: In this phase the if coordinator recieves 'YES' from all the participants site that are agreeing to commit then ~~coordinator~~ coordinator will send global commit message to all the participants.

04/10/2023

Short note (10m)

* SPATIAL DATA:
It is a data that stores information about physical location & geometrical objects. eg: maps & pics
geography

* NON SPATIAL DATA:
It stores data with some values.
eg: name, address, number.

- The data of images or the picture that are released by NASA satellite is eg. of spatial data.

- Data types used in spatial data:

1. Map data:
It includes different types of spatial features of objects in maps.
(building, roads)
3 types used for using data in map:
1. point (eg. buildings, vehicles)
2. lines (eg. roads)
3. Polygons (eg. sea, lakes, mountains)

2. Attribute data:
Data stored in map having features.
eg: largest city (in context to area), population, traffic
In this type of data, the info. ~~of~~ on the map is stored on basis of some features like largest city will be shown with the help of 'area' feature ~~as~~ of that city.

3. Image data:
It is stored data that, the photographs or the pictures that are taken by camera & objects of interest are shown such as buildings or roads by satellite.

10/10/2023

* Models of spatial data & information :

Spatial info is the info that consists that the objects, images, points, lines etc.

1. Feild
2. Object

1. **Feild:**
These models are used to store spatial data that is having continuous in nature data.
eg: temperature data, traffic data (using colors (red, yell.))

2. **Object:**
This model will be used for storing the data of an application which consist of the objects like roads, buildings.
eg: (roads - lines)

- Spatial query:
1. Range queries - Finds all object within given range of values within given spatial data.
eg: hospitals, colleges in ~~andheri~~ range (from Andheri to Bandra).

1. Find all the engg. colleges that are in Andheri at a distance of 10kms.

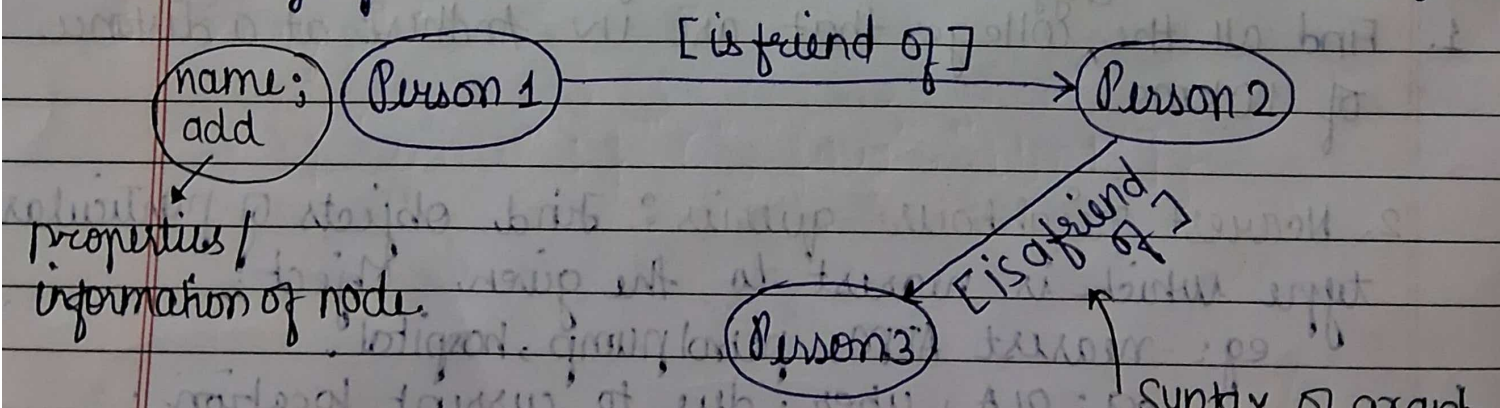
2. Nearest neighbour queries: Find objects of particular type which is nearest to the given object.
eg: nearest ATM, petrol pump, hospital.
eg: OLA, uber. due to current location.

- spatial join or overlays:
It joins the objects of two types based on spatial condition or geographical condition such as objects that are intersecting or overlapping.
eg: when physical location is same.
flyovers & roads, national highways betn 2 cit

05/10/2023

- GRAPH DATABASE:
The software or database that is going to store data in form of the graph instead of tabular form.
It consists of nodes and relationships or links between or among the nodes. The relationship in the graph gives the meaning to the nodes.
eg: Neo4j having cypher.
In Neo4j having cypher which is similar to mySQL.
social networking ~~sy~~ sites use graph database & not mySQL like facebook, insta, twitter etc.

eg → Facebook:
Consider facebook social networking site.
In graph database, relation betn the nodes;



There is friend suggestion of P_3 for P_1 due to graph database. This is used for recommendation in facebook.

Note: Nodes will be objects

Variable in graph database will be used to access the information from the graph that is stored in database.

eg:

p : person

variable used to fetch data of person.

match:
used for searching node, relationship that exists already.

- syntax: MATCH (p:person)
  RETURN p
  LIMIT 1

return:
What values you might ~~to~~ want to return.

limit:
used to display no. of nodes.

- syntax: MATCH (tom : PERSON {name : 'Tom Hanks'})
  RETURN tom

  (tom → variable, {name : 'Tom Hanks'} → information)

This syntax is used to find specific thing/person.

- To find movie directed by Tom Hanks:
  Syntax: MATCH (: Person {name : 'Tom Hanks'})-
  [: DIRECTED] → (movie : Movie)
  RETURN movie.

JSON : Java Script Object Notation

- Data types
- Retrival using K value pair
- query

Imp Q. What is JSON? And its data types. -5m
Q which are features & purpose of JSON. -5m

- It is open source text formatted language.
- Java script Object Notation.
- Similar to Java Script objects.
- It consists of name & value pair. (It make one object)

Purpose of JSON:
Used for transmitting data in web application.

- Syntax rule:
Data is in name/value pairs.
Data is separated by commas
curly braces to hold objects { }
square brackets to hold arrays. [ ]

Objects:
{"firstname": "John", "lastname": "Doe"}
value & name is separated.

- JSON array: → collection of similar datatype of objects
can contain the objects.

diff both objects

"employee": [{"firstname": "John", "Lastname": "Doe"},
{"firstname": "A", "lastname": "B"}]

Q → xml & json

What is XQuery & XPath.

- Quering on xml documents.
- Path that is written on xml documents.
- Written to fetch data.
- xml stores data with extension .xml.

Xquery follows syntax/keywords

⇓

FLWOR — return

For let where order by } meaning of each word.

address

eg: for $book in /bookstore/book
where $x/price >30
return $x/title

for $book in /bookstore/book
where $x/price >30
order by $x/title
Return $x/title.

Syntax:

For let : let x = books , let $x

for $x in /bookstore/book
where book/price >30
order by book/title
return $x.

1. JSON is Java script Object notation. It is open source text formatted language. It is similar to java script objects. It consists of name and value pair which makes one object.

Purpose of JSON:
JSON is used to transmit data in web application.

Syntax rule:
Data is in value/name pair.
Data is separated by commas.
Curly braces are used to hold objects { }.
Square brackets are " " arrays [ ].

Object:
{"firstname": "John", "lastname": "Doe"}
→ value & name separated

Array:
Collection of similar datatype of objects.

"employee": [{"firstname": "John", "lastname": "Doe"},
{"firstname": "A", "lastname": "B"} ...]

JSON types:
String = {"name": "John"}
number {"age": 30}
object
array
boolean {"sale": true}
null {"middlename": null}

Example :

{"employee": [{"name": "Shyam", "email": "shyam@gmail.com"}
, {"name": "Bob", "email": "bob32@gmail.com"},
{"name": "Jai", "email": jai4050@gmail.com"}]}

2. CAP theorem:

The CAP theorem, can be used to explain some of competing requirements in a distributed system with replication.

CAP represents
C- consistency (among replicated copies)
A- Availability (of the read & write operations)
P - Partition tolerance (in the face of the nodes in the system being partitioned by a network failure).

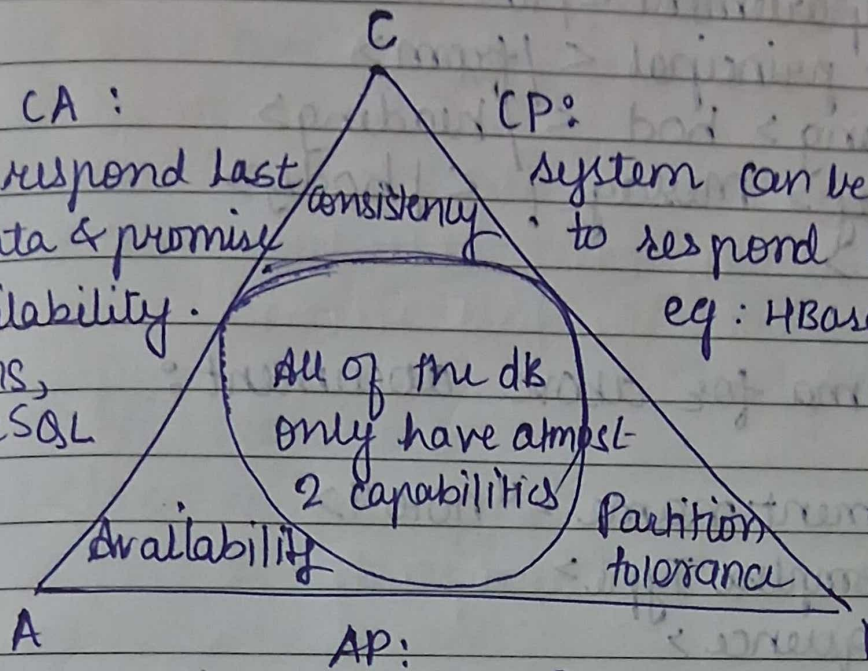
The CAP theorem states that it is not possible to guarantee all three of the desirable properties at the same time in a distributed system with data replication.

Consistency:
It means that the nodes will have the same copies of a replicated data item visible for various transactions. A guarantee that every node in a distributed cluster returns same, most recent & a successful write.

Availability:
It means that each read or write requests for a data item will either be processed successfully or will receive a message that the operation cannot be completed.

Partition tolerance:
It means that the system can continue operating even if the network connecting the nodes has a fault that results in 2 or more partitions, where the nodes in each partition can only communicate among each other.

Following gives which DB systems prioritize specific properties at a given time:



3. XML schema:

XML schema is a language which is used for expressing constraints in XML documents. There are schema language like RelaxNG and XSD present now-a-day.

A XML schema defines the structure of an XML document.

An XML with correct syntax is called "well formed".

simple XML document called "notes.xml":

```
<? xml version="1.0" ?>
<note>
<to> departement </to>
<from> principal </from>
<heading> hod </heading>
<body> message! </body>
</note>
```

XML schema for above document:

```
<xs: element name="note">
<xs : complex Type>
<xs : sequence>
<xs: element name="to" type="xs: string"/>
<xs: element name="from" type="xs: string"/>
<xs: element name="heading" type= — " —>
<— — — = " body " — " —>
</xs: sequence>
</xs: complex type>
</xs: element>
```

4. MongoDB

create, insert one, insert many
find, update, drop, delete
($and, $or, $not, $nor)

1. Create database:
MongoDB created DB & collection automatically when you insert the first record:

Create:
>use my_company
switched to db my_company

list all db:
>show dbs
admin 0.000 GB
config 0.000 GB
local 0.000 GB

copy

2. Insert Record:
db.collection.insert() is used to insert new record into the collection. It can be used to insert a single as well as multiple records.

Insert Single Record
>db.employee.insert({name:"Gargi", email:"gargihawalder@gmail.com", age:21})
copy.

Insert multiple records:

```
> db.employee.insert(
    [
      { name:"Anushka", email:"anushka12@gmail.com",
        age: 20 },
      { name:"Aditi", email:"aditi24@...";
        age: 21 } ]
```

~~copy~~

3 Select:

db.collection.find() method is used to fetch the data from the database.

Fetch all:

```
> db.employee.find()
```

Fetch with condition:

```
> db.employee.find({email:"aditi24@gmail.com"})
```

Fetch multiple condition:

```
> db.employee.find({age: 21, name:"anushka"})
```

4. Update:

db.collection.update({criteria}, {$set: {new value}})
method is used to update the existing document of the collection.

```
> db.employee.update({email:"gargihawaldar@gmail.com"
  , {$set: { age: 22 }})
WriteResult({"nMatched":1,
```

5. logical operators in MongoDB :

($and, $or, $not, $nor)

MongoDB supports logical query operators. These operators are used for filtering the data & getting precise result based on the given condition.

1. $and - It is used to join query clauses with logical AND & return all documents that match the given condition of both clauses.

2. $or - It is used to join query clauses with logical OR & return all documents that match the given condition of ~~both~~ either clauses.

3. $not - It is used to invert the effect of query expression & return document that does not match the query expression.

4. $nor - It is used to join query clauses with logical NOR & return all documents that fail to match both clauses.

Difference betn SQL & NoSQL

6.

	SQL	NoSQL
1	Relational database management system (RDBMS)	Non relational or distributed db system.
2.	These db have fixed schema.	They have dynamic schema.
3.	These db are not suited for heirarchical data storage.	Best suited for heirarchical data storage.
4.	Best suited for complex queries.	Not so good for complex queries.
5.	Vertically scalable	Horizontally scalable.
6.	ACID transaction supported	Follows CAP.
7.	Table based databases.	Document based, key-value pairs, graph databases.
8.	eg: MySQL, PostgreSQL, Oracle, MS-SQL server.	eg: MongoDB, HBase, Neo4j, Cassandra.