Unit 2

Database System Architechture :-

★ The architechture of a database system is greatly influenced by underlying computer system on which it runs, in particular by such aspects of computer architechture.

→ Networking ⇒ client server system
→ Parallelism ⇒ Parallel database system
→ Distribution ⇒ Distribution database system.

Centralized System :-

★ A modern, general purpose computer system consist of one to a few CPU's a no. of device controllers that are connected through a common through a common bus that provides access to shared memory.

★ CPU's have local cache memories that stores local copies of part of a memory to speed up access to data.

★ Each device controller is in charge of a specific type of device

→ Disk drive
→ Audio device
→ Video device.

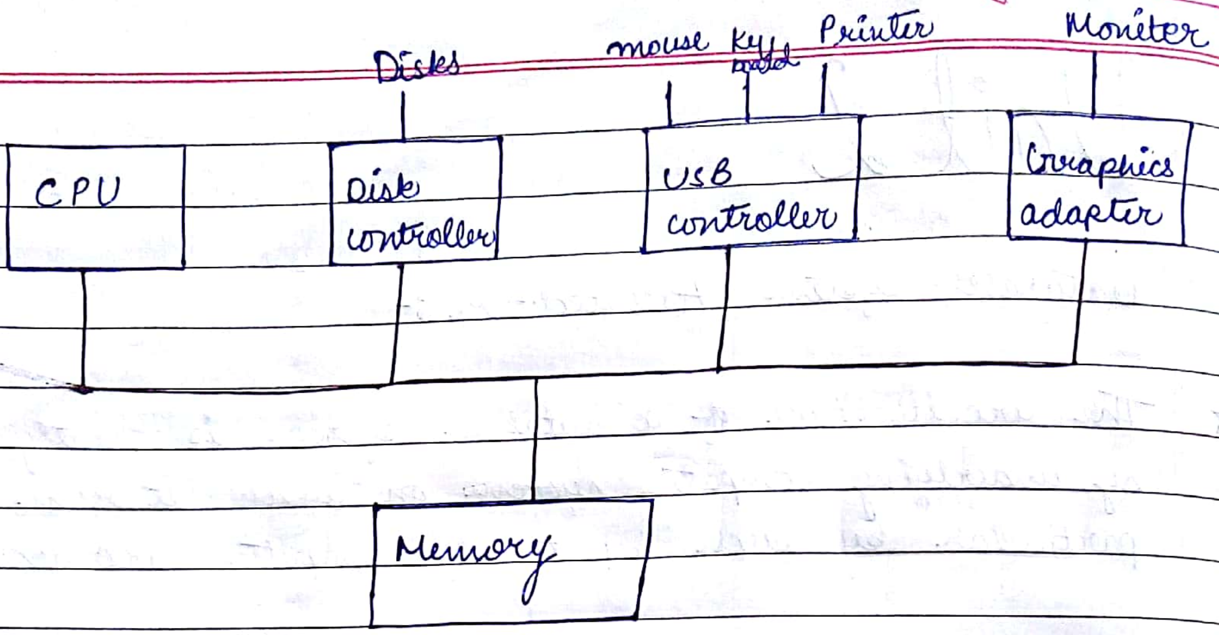


fig. A centralized computer system.

Centralized System :- Two types of parallelism.

Coarse-granularity parallelism :-

★ Database running on such machines usually don't attempt to partition a single query among processors.

★ Each query run on a single processor.

★ System support higher throughput.

Fine Granularity parallalism :-

★ DB system running on such machines attempt to parallilize single tasks (queries) ~~sub~~ submitted by the users.

Client Server system :-

(i) FRONT END

(ii) BASIC END

FRONT END :-

consist of → SQL user interface tools.
→ forms interface
→ Report generation tools.
→ Data mining and analysis tools.

BASIC END manages:-

→ access structure
→ query evaluation
→ optimization
→ concurrancy controls.
→ Recovery.

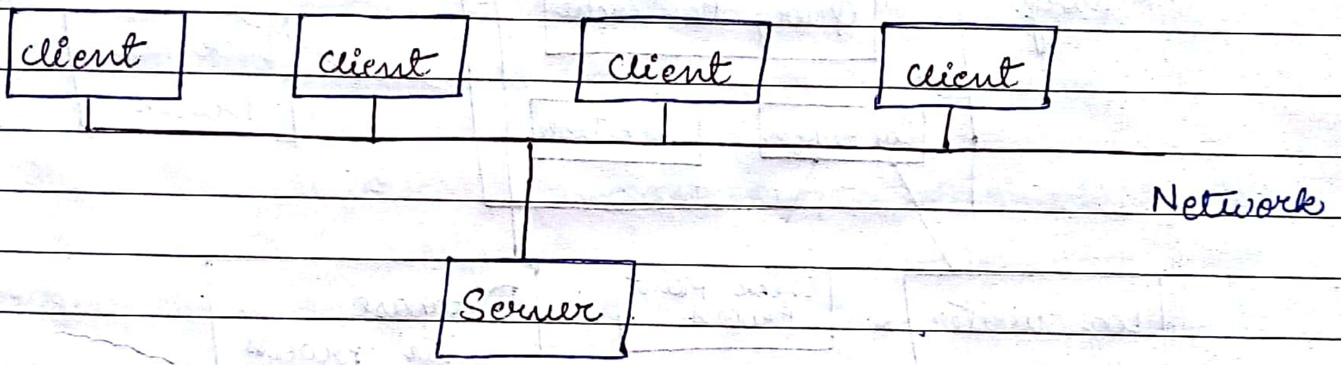


fig. general structure of client - server system.

Server system architechture :-

1. Transaction server system
2. Data server system

1. Transaction server system :-

★ Query server system

★ It provides an interface to which clients can send request to perform an action in response to which they execute action and send back results to the clients.

★ Request may specified by using SQL or through any specialized application program interface.

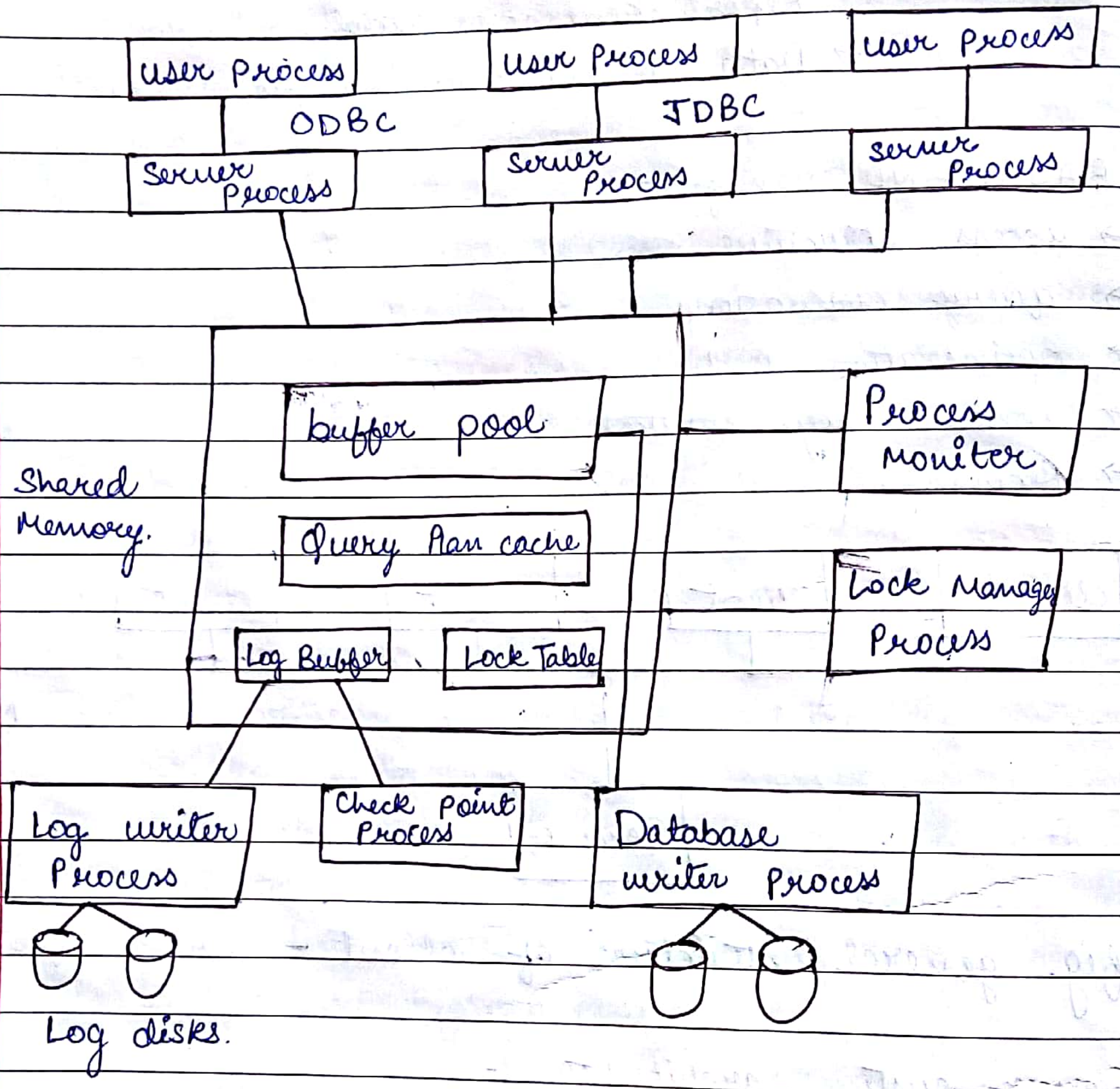


fig. Shared Memory and process structure.

Transaction server process structures :-

★ consist of multiple process.

(a) ~~consist~~ of server process

(b) lock Managers

(c) Database writer

(d) check point process

(e) process moniter process.

—X—

1. Server process:- These are the processes that recieve user queries, execute them and send the result back. The queries may be submitted to the server process from a user interface or from a user process running

2. Lock managers:- This process impliment lock managers functionally, which includes lock grant, lock release and deadlock dedection.

3. Database writer process:- There are one or more processes that output modify buffer blocks back to the disk on a continious basis.

4. Log writer process:- This process output log record from the log record buffer to the stable storage. The server process simply add log record to the log record buffer in shared memory and if a log force is required, they request the log writter processes to output the log records.

5. Check point process:- This process performed ~~prod~~ periodic check point.

6. Process Moniter:- This process moniter other processes and if any one of them fail it take recovery action for the process, such as aborting any transaction being executed by the failed process and then restarting the process.

Note:→ Shared Memory:- The shared memory contains all shared data such as:-

★ buffer pools.

★ Lock table

★ log buffer, containing log records waiting to be output to the log on stable storage.

★ cached query plans which can be reused if the same query is submitted again.

—X—

Data Server:-

① Data server system are use in local area network, where there is a high speed connection between the client and the server the client machines are comparable in processing power to the server machine.

② In such an environment it make sense to ship the data to the client machine to perform all processing at the client machine and then to ship the data back to the server machine.

★ Issue arise in such an architechture, since ~~time~~ time cost of communication between client & server is high compared to local memory refrence.

(a) Page shipping versus item shipping:-

★ The unit of communication for data.

★ can be

→ coarse granularity (such as page)

→ fine granularity (such as tuple or object).

★ If unit of communication is single item, the overhead of message passing is high compared to the amount of data transmitted.

★ prefetching → ⊙ Fetching items ever before they are requested is called as prefetching.
⊙ For eg. page shipping.

(b) Locking :-

★ Locks are usually granted by the server for the data item it ships to the client machine.

★ A disadvantages of page shipping is that client machine may be granted locks of too coarse of granularity a lock on page implicity lock all items contained in page.

★ Technique for lock $\boxed{\text{de - escalation}}$:- The server can request its clients to transfer back locks on prefetched items.

(c) Data caching :-

★ Data that are shipped to a client on behalf of a T can be cached at the client even after the T completes if sufficient storage space is available.

★ cache coherency is an issue.

(d) Lock caching :- ~~If~~

★ If the use of data is mostly partitioned among the client with clients rarely requesting data that are also requested by other clients locks can also cached at the client machine.

—X

Parallel System :-

Parallel system are required to query extremely large database order of terabyte or 10^{-12} that have process are extremely large no. of transaction per second of the order of 1000 of transaction per second.

In parallel processing, many operations are performed simultaneously.

1. A course grain parallel machine consist of a small no. of powerful processors.

2. A massively parallel or fine - grain parallel machine use thousands of smaller processors.

Interconnection Network in parallel system :-

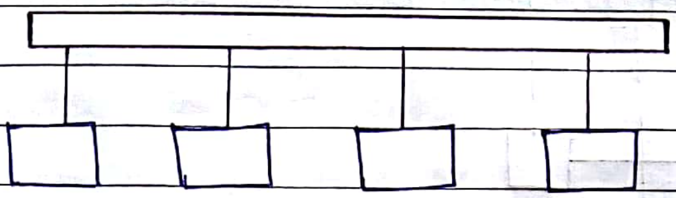
Parallel system consist a set of components (Processors memory and disks) that can communicate with each other through interconnection network.

Basically 3 types of Interconnection are as follows:-

1. Bus
2. Mesh
3. Hyper cube.

1. Bus :- All the system components can send the data or recieve the data from a single communication bus. It work well for small no. of processors but can n't handle those cases in which parallism increses. Since, bus can handle communication

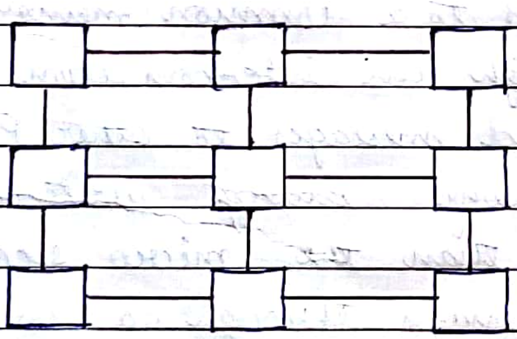
only one component at a time.



(a) bus.

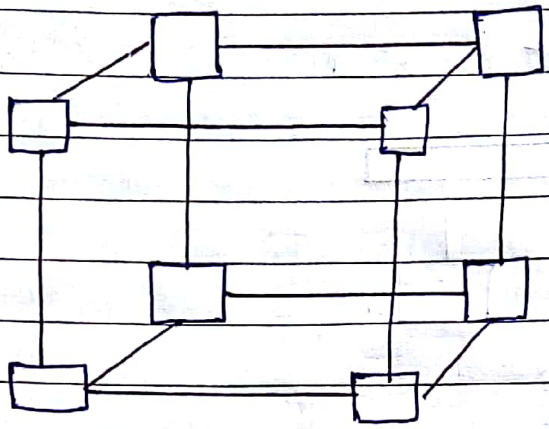
2. Mesh :- The components are nodes in a grid and each component connect to all its adjecents components in a grid. In 2-D mash each node connect to 4 adjecent node while in 3-D mash each nodes connect to 6 adjecent nodes.

Nodes that are not directly connected can communication with one ~~an~~ another by routing massages. i.e. a sequence of intermediate node that are directly connected to one another.



(b) Mesh.

3. Hyper cube :- The components are numbered in binary and a component is connected to another if the binary representation of there no. differs in exactly one bit. In given diagram shows a hypercube with 8 nodes.



(c) Hypercube.

—x—

Parallel Database Architechture :-

There are several Architechture models for parallel machines.

1. Shared Memory :-

i> All the processors shares a common memory.

ii> In a shared memory architechtureprocessor and the disk have access to a common memory, typically by a bus or through an interconnection network.

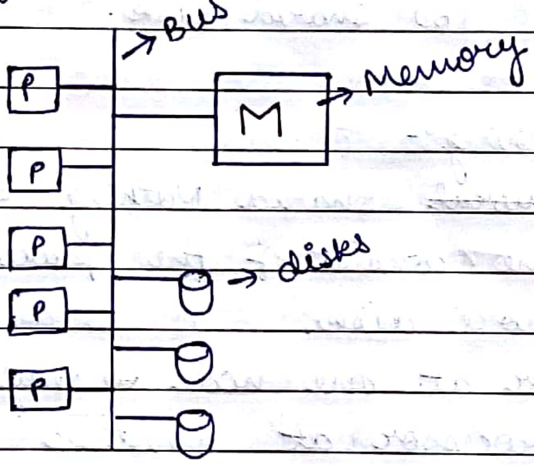
iii> A processor can send messages to other Processor much faster by using memory write which usually take less than the micro second then by sending the messages through a communication mechanism.

iv> The drawback of this architechture is that it is not scaleable beyond 32 or 64 preprocessors b/c the bus and interconnection network becomes a bottle neck.

v> Adding more processors does not help after a point since the processor will spend most of the time waiting for there turn on the bus to access the memory.

vi) Shared memory architechture have large memory cache at each processor, so that refering of the shared memory is avoided whenever possible.

vii) The cache memory data need to be kept coherent i.e. if a processor perform a right to a memory location, the data in that memory location should be either updated or removed from any processor where the data are cache. Maintaining cache coherence becomes and incresing overhead with incresing no. of processors.



(a) Shared memory.

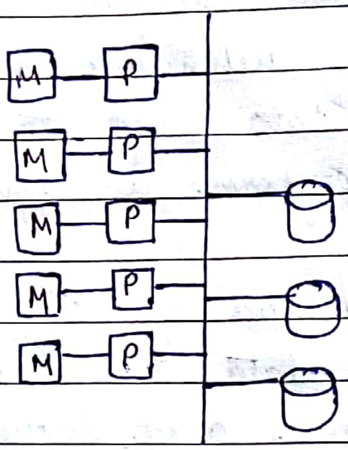
X

2. Shared disk :-

i) In the shared disk model, all processors can access all the disk directly through an interconnection network, and each processor also have its own private memory.

ii) Since each processor has its own memory, the memory bus is not a bottle neck here. Secondly it also ~~off~~ offers away to provide a of fault tolerance i.e. a processor or its memory fail the other processor can take over its task.

since the database is available on the disk that are accessible from all the processors.



(b) Shared Disk

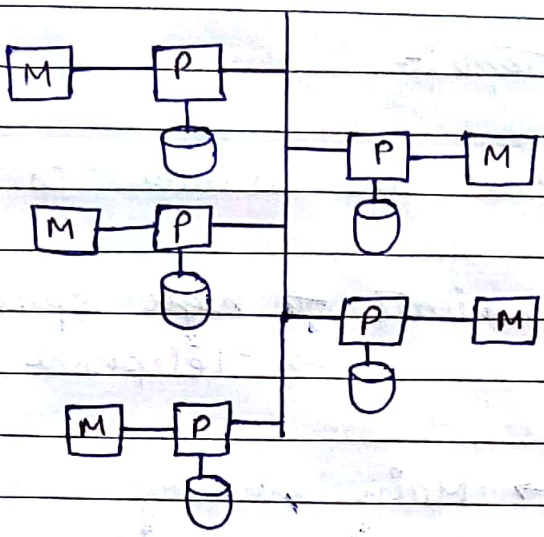
3. Shared Nothing:-

i> In the ~~several~~ shared Nothing system each node of the machine consist of the processor, memory and one or more disk.

ii> The processor at one node may communicate with another processor at another node by a high speed interconnection network.

iii> A node will function as a for the data on the disk which its owns.

iv> Since the local disk referance are serviced by local disk at each processor, this model overcome the disadvantages of requiring all Input / output to go through a single interconnection network and only the queries and the result relation pass through the network.

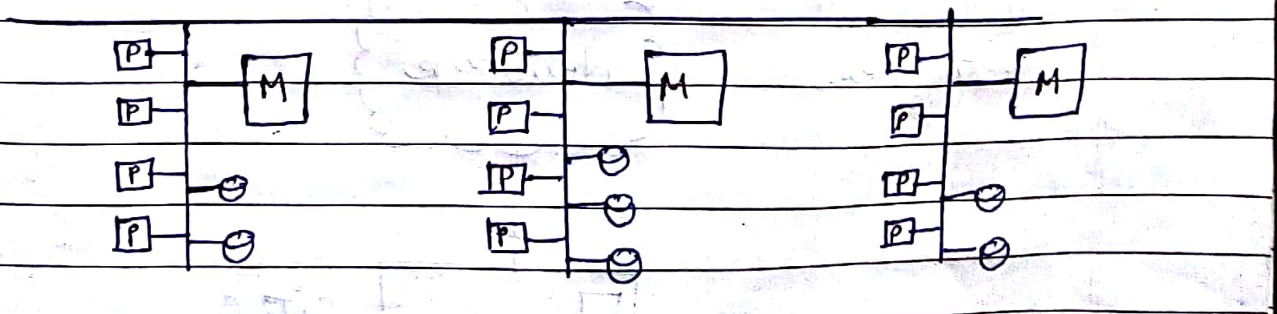


(c) Shared Nothing.

4. Hierarchical :-

i) This architechture combine the characterstics of shared memory shared disk and shared Nothing architechture at the top level the system consist of nodes that are connected by an interconnection network and do not share disk or memory with so top level is shared Nothing architechture.

ii) Each node could be a share disk system & each of the system sharing a set of disk could be a shared memory system. Therefore, a system could be built with shared memory architechture with a few processors at the base and a shared Nothing architechture at the top with possibility a shared architechture in the middle.



(d) Hierarchical.

—X—

Distributed System :-

★ Database is stored on several computers

★ Computers communicate → high speed Network
→ Telephone Lines

★ Don't share → Main memory.
→ Disks.

★ Nodes or sites (Physical distribution of computer).

★ Transaction → Local Transaction (is one that access data only from sites where T was initiated)
→ Global Transaction (is one that either access data in a site different from on at which T initiated or access data from several sites).

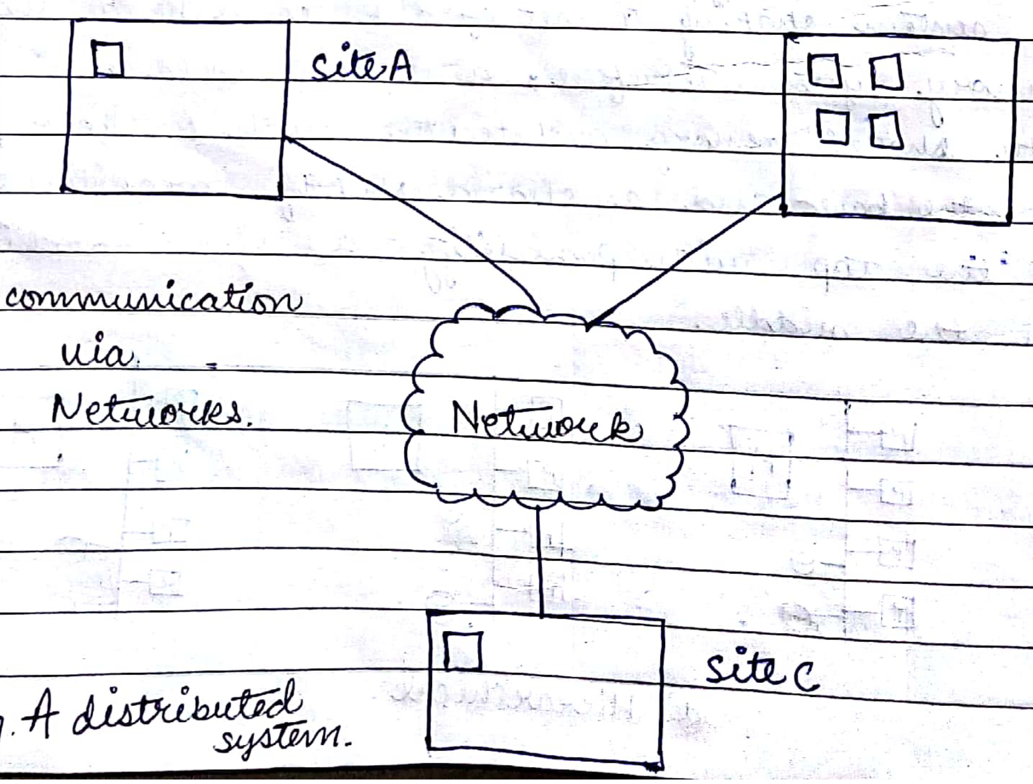


fig. A distributed system.

Advantages of Distreibuted system:-

1. Sharing Data:- User at one site may be able to access the data residing at other sites.
 for eg:- Distributed Banking system.

2. Autonomy:- There is a global database administrator responsible for the entire system. A part of these responsibilities is delegated to local database admistrator for each site.

3. Availability:- If one site fails, Remaining sites may be able to continue operating. ~~system~~ The failure of one site must be detected by the system and appropriate action may be needed to recover from the failure. The system must no longer use the service of the fail site recover or is repair, mechanism must be available to integrated it smoothly back into the system.

Object Based DataBase:-

Object Relational data model:- It extends the relational data model by providing a rich type system including complex data type and Object Orientation.

Object Relational database system:- It is a Database System i.e. Based on the object Relational model and provide a convinient Database who wish to object oriented features.

—x—

Complex Data types:-

★ Consider a Library application and suppose we wish to store the following information for each book.

- ★ book title
- ★ List of authors
- ★ Publishers
- ★ set of Keywords.

⇒ If we define a relation for this them, several domains will be non-atomic.

★ Authors:- A book may have a list of authors which we can represent as an array.

★ Keywords:- If we store a set of Keywords for a book we expect to be able to retrieve all books whose keywords include one or more specific keywords.

★ Publisher :- consist of subfield

→ Name

→ Branch.

Title	Author_array	Publisher	Keyword_set
		(name, branch)	
compilers	[smith, Jones]	[Mc-Grow-Hill New york)	{Parsing, analysis)
Network	[Jones, Frick]	(oxford, London)	{Internet, web}

fig. Non-1NF book relation.

★ we assume that title of a book uniquely identifiers the book.

★ we can them represent the same information using following schema.

- Authors (title, author, position)
- Keywords (title, keyword)
- book (title, pub_name, pub_branch).

Title	Author	Position
Compilers	Smith	1
Compilers	Jones	2
Networks	Jones	1
Networks	Frick	2

Authors.

Title	Keyword
Compiler	Passing
Compiler	analysis
Networks	Internet
Networks	web.

Keywords.

Title	Pub_name	Pub_branch
Compiler	Mc Graw_Hill	Netyork
Network	Oxford	London

Books

Imp The 4NF design requires queries to join multiple relation where as the Non-1NF design makes many types of queries easier.

Structure types:-

★ It allows composite attributes of E-R diagrams to be represented directly.

★ We can define following structured types to represent a composite attributes name with component attributes firstname and lastname.

Syntax:-

```
Create type Name as
( firstname varchar (20);
  lastname varchar (20));
  final
```

★ To represent a composite attributes address:-

Syntax:-

```
Create type Address as
(  street   varchar (20),
   city     varchar ( 20),
   Zipcode  varchar (9));
  not final.
```

★ Now use these types to create composite attributes in a relation by declaring an attributes to be of one of these types.

Scanned by CamScanner

For eg:- Create a table customer as follows:-

```
create table customer
( name     Name,
  address  Address,
  date of Birth  date);
```

FINAL:- The final specification for Name indicate that we can't create sub type for name whereas that not final specification for Address indicate that we can create sub types of address.

★ The component of composite attributes can be accessed using a dot (.) operator.
for example:- name.firstname
Return the firstname component of the name attributes.

★ we define a type CustomerType and create the table customer as follows:-

Syntax:-

```
Createtype CustomerType as
( name    Name,
  address Address,
  date of Birth  date);
 not final
create table customer of CustomerType.
```

—X—

Type Inheritance:-

★ Suppose that we have following types definition for people.

eg:-

```
Create type Person
   ( name      varchar (20)
      address   varchar (20));
```

★ To add information in DB about people (Student and teachers).

eg:-

```
Create type Student
under person
( degree varchar (20),
  department varchar (20));

Create type teacher
 under person
( salary integer,
  deparment varchar(20));
```

★ Both student and teacher inherit the properties of person.

★ student and teacher are subtypes of person or person is a subtype of them (S, T).

★ A subtype can redefine a method by declaring a method again using overridding method.

★ If a system support Multiple inheritance, we can define.

eg:-

Create type TeachingAssistant
under Student , Teacher.

OR

Create type TeachingAssistant
under Student with (department as student_dept),
Teacher with (department as teacher_dept);

Table Inheritance:-

★ Suppose we define the people table as follows:-

eg:-

create table people of person.

★ Define tables student and teachers as subtables of people , as follows:-

eg:-

create table student of student
under people
create table teacher of teacher
under people.

★ The Types of subtables must be subtypes of the type of the parent table. Every attributes present in table people is also in subtables.

★ we declare S & T as subtables of people , every tuple present in student and teacher becomes also implicity present in people.

ONLY KEYWORD:- The ONLY Keyword is used delete and update statement of a query it is use to find the tuples that are in people but not in its subtable by using "only people" in the query.

eg:- delete from people where P.

The following statement will delete all the tuple from the table people as well as it subtables student and teacher that satisfied P. If the "ONLY" Keyword is added to the ~~above~~ above statement, tuple that were inserted in the subtables are not ~~satisfied~~ affected, even if they satify the where clause condition.

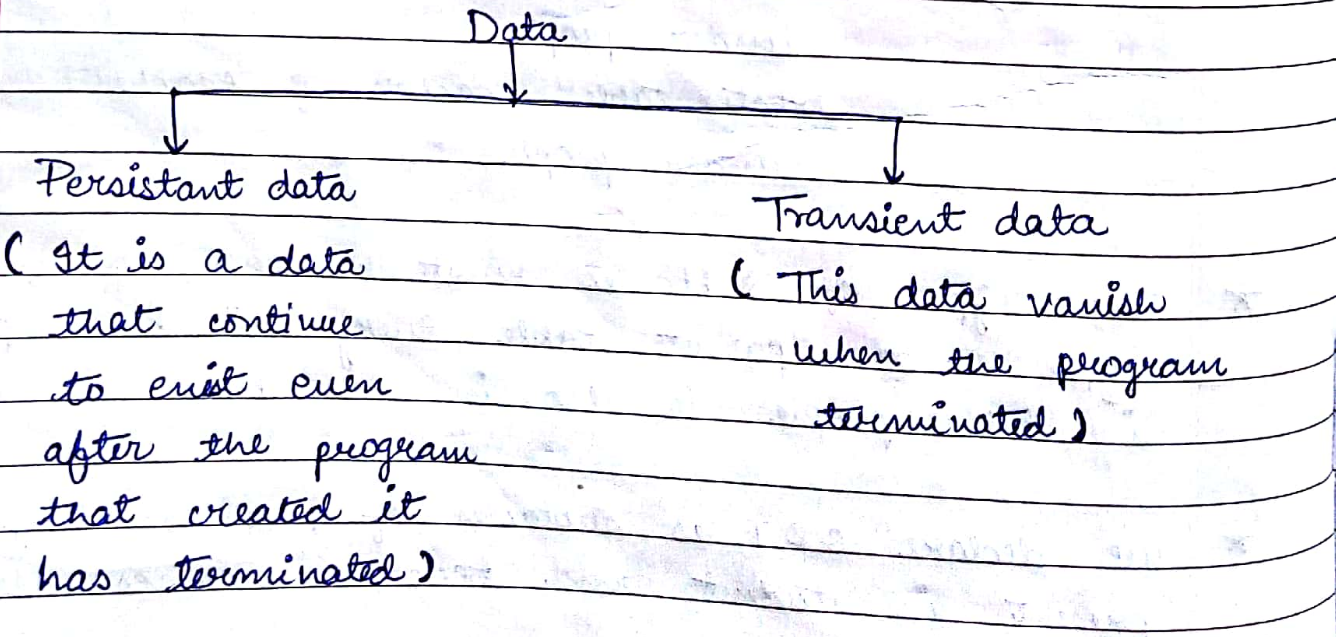
X

Object Based Database:-

★ Persistent Programming language:-

★ These languages manipulate data that are persistant.

★



Data

Persistant data
(It is a data that continue to exist even after the program that created it has terminated)

Transient data
(This data vanish when the program terminated)

★ Approaches proposed for persistent object

① Presistence by creation.	② Persistence by class.	③ Persistent by marking	④ Persistence by Reachability
★ New syntax is introduced to create persistent objects ★ Extending the syntax for creating transient objects.	★ Simple Method ★ Declaee the class as persistent ★ Then by default all the object of that class will be Persistent.	★ Mark the object as persistent after they are created.	★ If any objects is reachable from the root objects through a sequence of one or more refrences.

★ Object Identity and Pointers :-

1. Transient object identifires are valid only when program that created then is executing as program terminates, object are deleted and identifies is meaning less.

2. when a persistent object is created, it assigned a persistent object identifires.

★ Several Degree of ~~pert~~ permanence of Identity.

~~(a) / Intraprocedure~~

(a) Intraprocedure :-

(Identity persists only during execution of a single procedure).

eg:- Local Variable.

(b) Intraprogram :-
(only during execution of a single program or query)
eg:- Global Variable.

(c) Interprogram :-
(Identify persist from one program execution to another.)
eg:- File Management.

(d) Persistent :-
(Among structured re- organisation of the data).
Identify persist not only among program execution, but also among straptural reorganisation of the data. It is the persisting form of identity that is required for object oriented system.

—x—

Storage and Access of Persistent objects :-

Several ways :-

1. give names to object

2. Expose object Identifiers or persistent pointers to the objects which can be stored externally.

3. Store collection of objects → Set
→ Multisets
→ list etc.

Persistent C++ system :-

★ Persistent pointers:- A new datatypes has to be defined to represent persistent objects.

★ Creation of persistent object :- C++ new operator is used.

★ Class extents :- They are created & maintained automatically for each class.

★ Relationship :- It is represented by string painters for each object to the object that it is related to.

★ Trasaction :- Persistent C++ system provide support for
→ Storing
→ committed
→ rollback

X