BITS Pilani
Pilani Campus

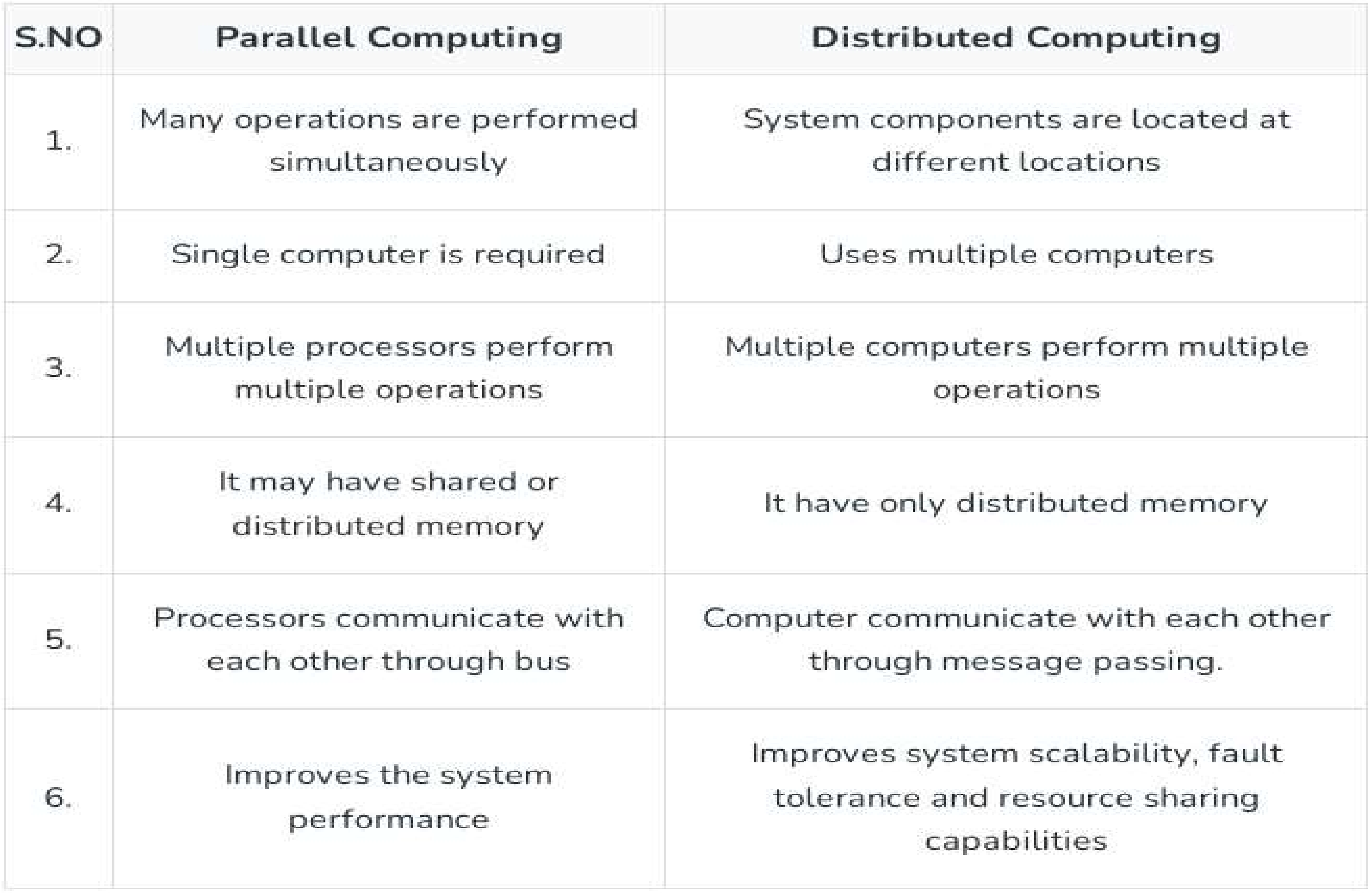
BITS Pilani presentation

Dr.Vivek V. Jog
Department of Computer Engineering

BITS Pilani

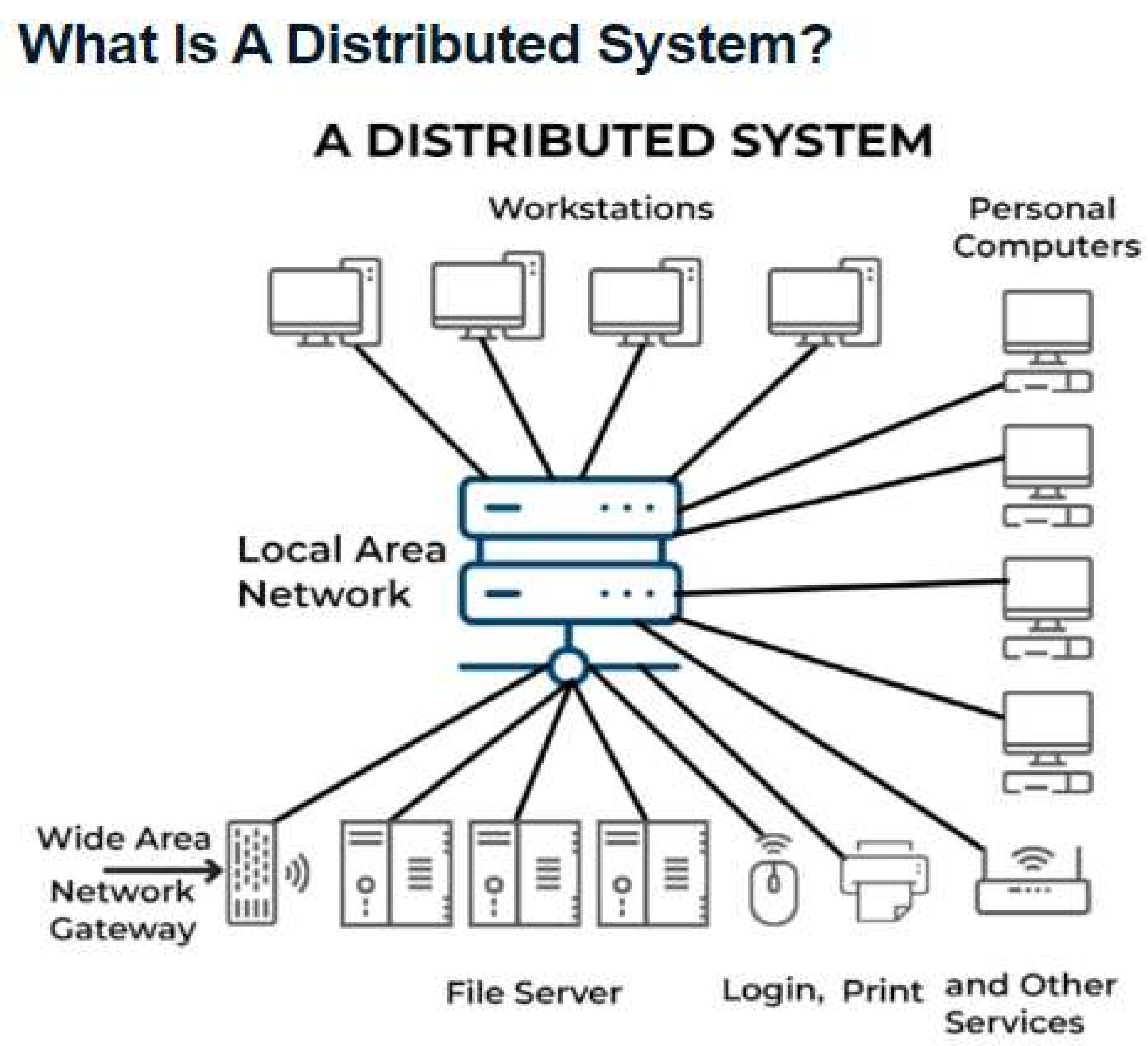
Pilani Campus

Lecture No.2



S.NO	Parallel Computing	Distributed Computing
1.	Many operations are performed simultaneously	System components are located at different locations
2.	Single computer is required	Uses multiple computers
3.	Multiple processors perform multiple operations	Multiple computers perform multiple operations
4.	It may have shared or distributed memory	It have only distributed memory
5.	Processors communicate with each other through bus	Computer communicate with each other through message passing.
6.	Improves the system performance	Improves system scalability, fault tolerance and resource sharing capabilities

Distributed Env


What Is A Distributed System?
A DISTRIBUTED SYSTEM
Workstations
Personal Computers
Local Area Network
Wide Area Network Gateway
File Server
Login, Print and Other Services

A distributed system is a **collection of computer programs spread across multiple computational nodes**. Each node is a separate physical device or software process but works towards a shared objective. This setup is also known as distributed computing systems or distributed databases.

The main goal of a distributed database system is to **avoid bottlenecks and eliminate central points of failure** by allowing the nodes to communicate and coordinate through a shared network.

Benefits Of Distributed Systems

Scalability: Distributed database systems offer improved scalability as they can add more nodes to easily accommodate the increase in workload.

Improved reliability: It eliminates central points of failure and bottlenecks. The redundancy of nodes ensures that even if one node fails, others can take over its tasks.

Enhanced performance: These systems can easily scale horizontally by adding more nodes or vertically by increasing a node's capacity. This scalability results in enhanced performance and optimum output.

Drawbacks & Risks Of Distributed Systems

Requirement for specialized tools: Management of multiple repositories in a distributed system requires the use of specialized tools.

Development sprawl and complexity: As the system's complexity grows, organizing, managing, and improving a distributed system can become challenging.

Security risks: A distributed system is more vulnerable to cyber attacks, as data processing is distributed across multiple nodes that communicate with each other.

Drawbacks & Risks Of Distributed Systems

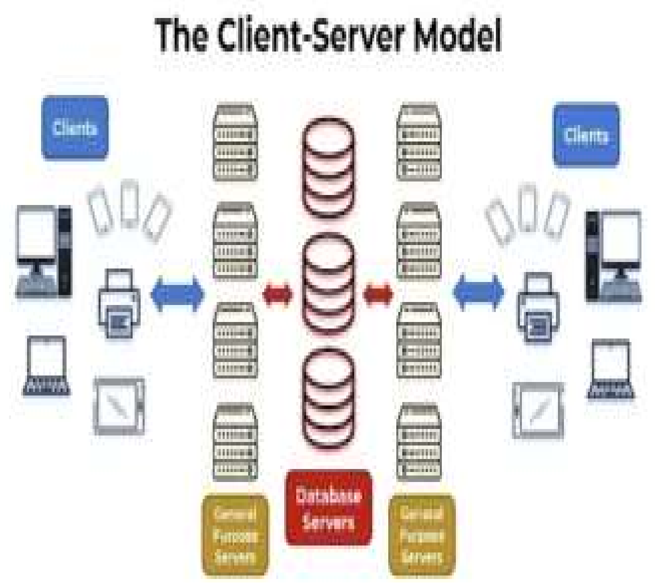
Requirement for specialized tools: Management of multiple repositories in a distributed system requires the use of specialized tools.

Development sprawl and complexity: As the system's complexity grows, organizing, managing, and improving a distributed system can become challenging.

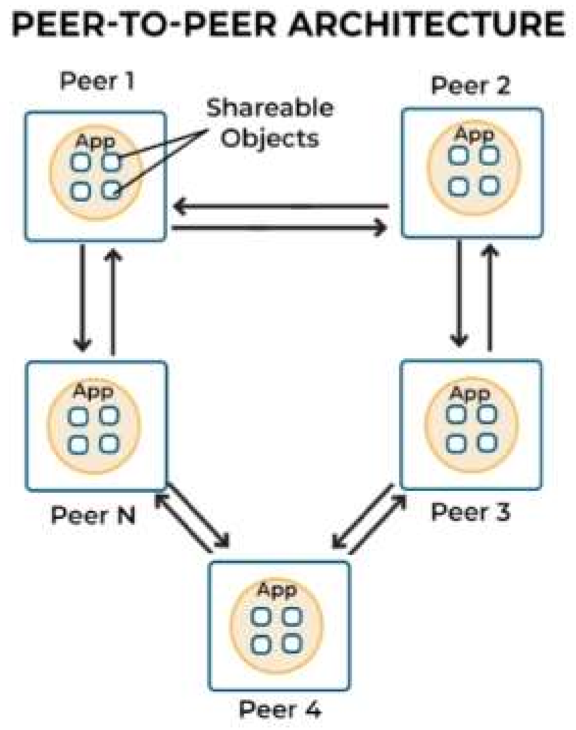
Security risks: A distributed system is more vulnerable to cyber attacks, as data processing is distributed across multiple nodes that communicate with each other

Examples

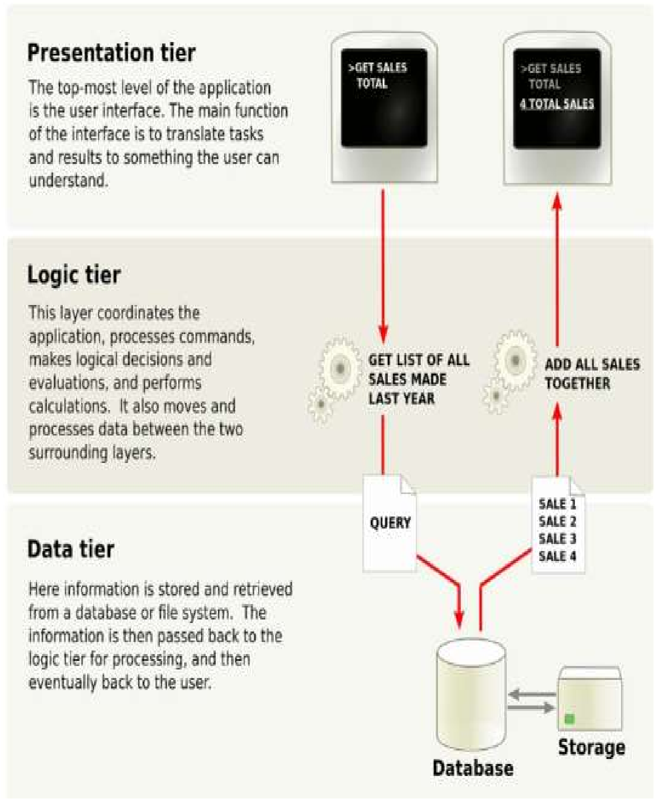
Client-Server Architecture



Peer-To-Peer (P2P) Architecture



Multi-Tier (n-tier) Architecture



Presentation tier

The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.

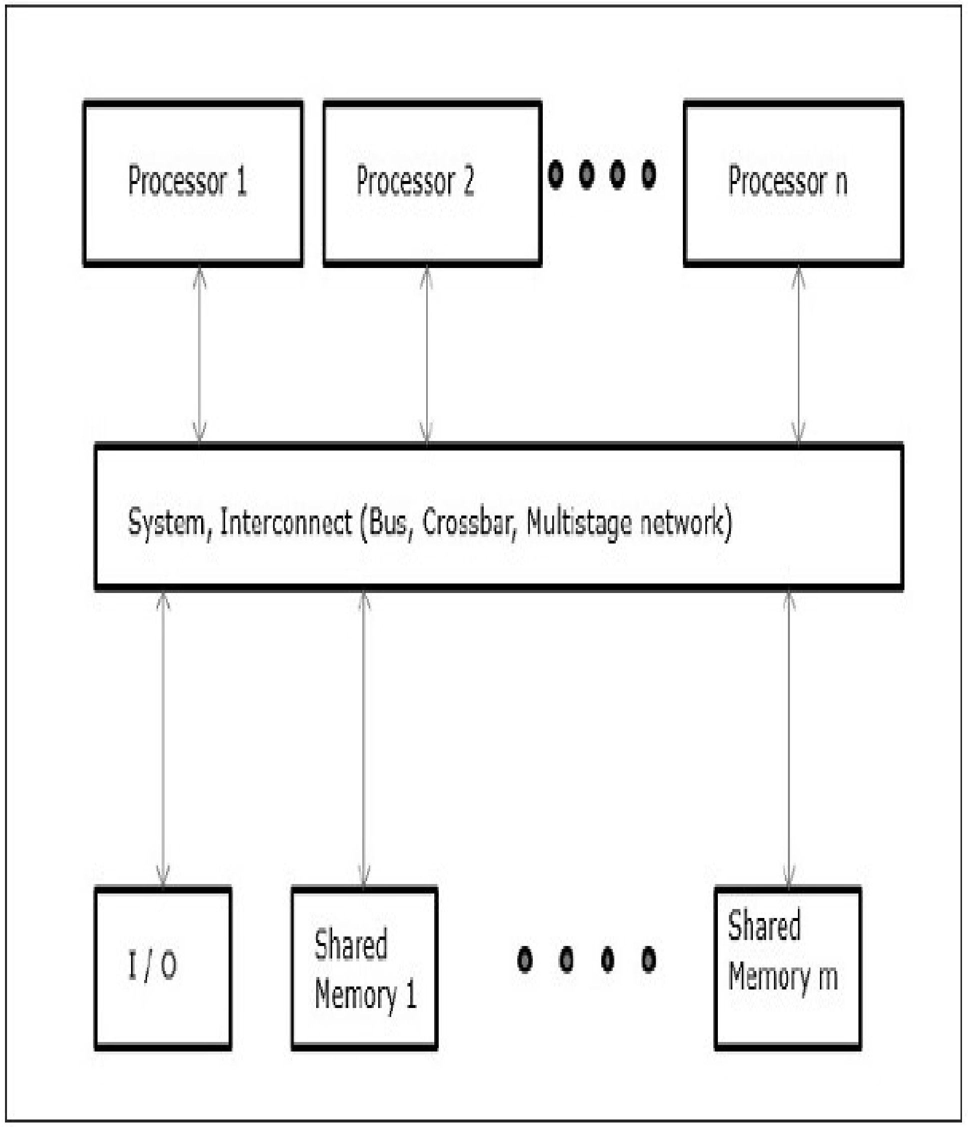
API can also viewed as Distributed Service Flavor

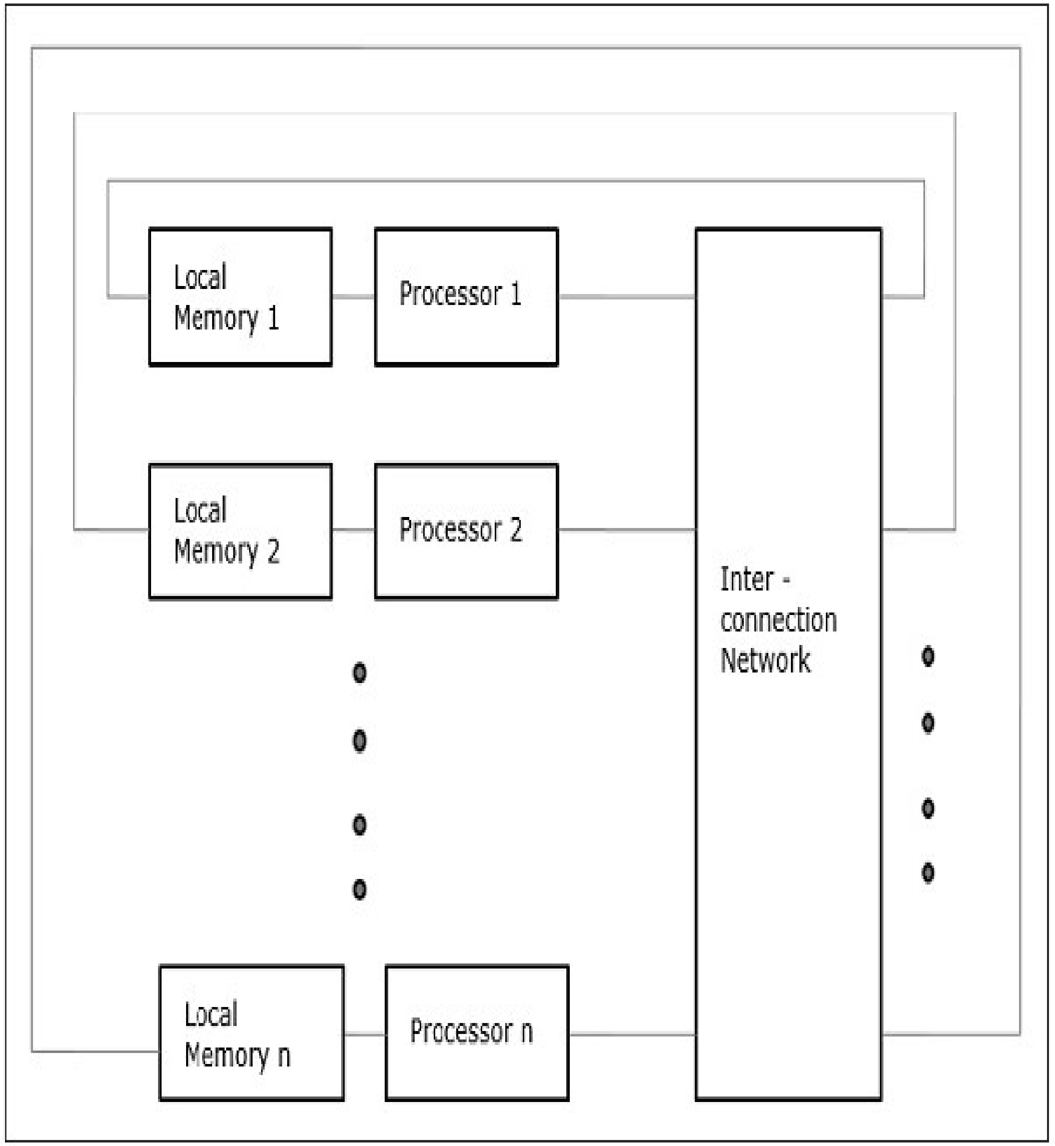
Definition : Parallel Systems

Parallel computing refers to the process of executing several processors an application or computation simultaneously. Generally, it is a kind of computing architecture where the large problems break into independent, smaller, usually similar parts that can be processed in one go.

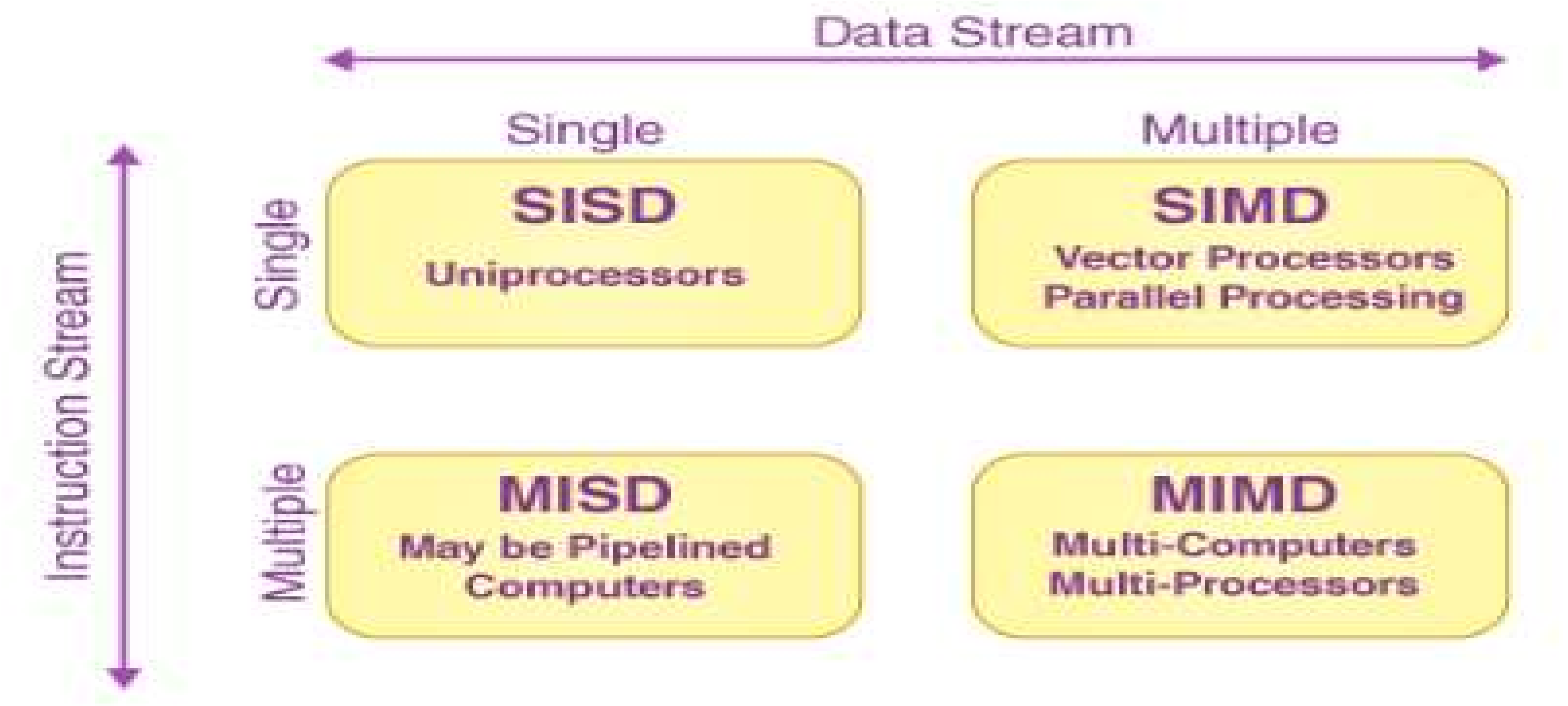
It is done by multiple CPUs communicating via shared memory, which combines results upon completion. It helps in performing large computations as it divides the large problem between more than one processor.

Parallel Systems UMA/NUMA





Flynn's Classification



Parallel processing can happen in the data stream, the instruction stream, or both.

Parallel V/s Distributed Databases

Parallel Database:

A parallel DBMS is a DBMS that runs across multiple processors and is designed to execute operations in parallel, whenever possible. The parallel DBMS link a number of smaller machines to achieve the same throughput as expected from a single large machine.

Features :

There are parallel working of CPUs
It improves performance
It divides large tasks into various other tasks
Completes works very quickly

Distributed Database:

A Distributed database is defined as a logically related collection of data that is shared which is physically distributed over a computer network on different sites. The Distributed DBMS is defined as, the software that allows for the management of the distributed database and makes the distributed data available for the users.

Features :

It is a group of logically related shared data
The data gets split into various fragments
There may be a replication of fragments
The sites are linked by a communication network

Definition Shares/Message Pass Model

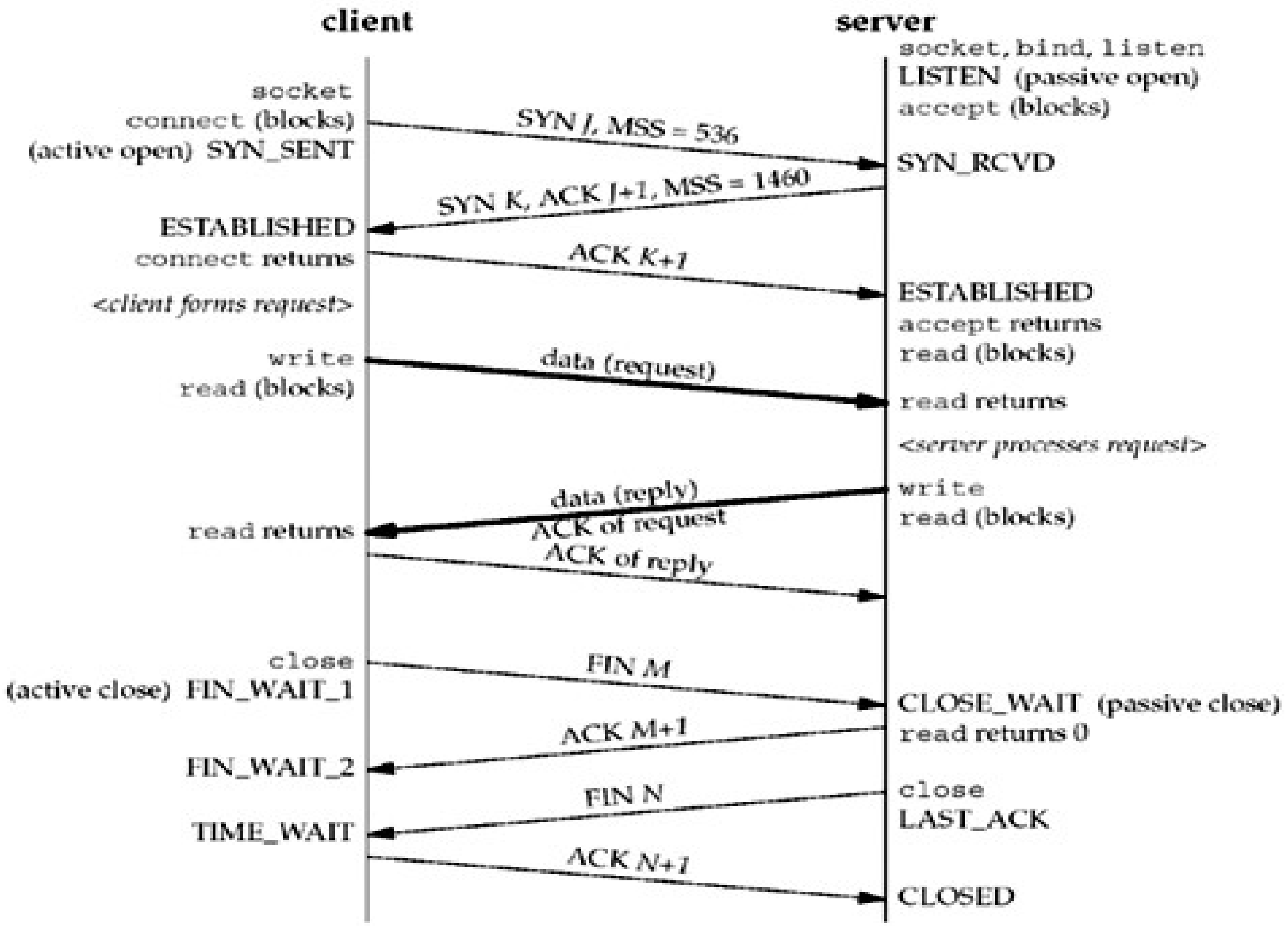
What is Shared Memory?
The fundamental model of inter-process communication is the shared memory system. In a shared memory system, the collaborating communicates with each other by establishing the shared memory region in the address space region.

If the process wishes to initiate communication and has data to share, create a shared memory region in its address space. After that, if another process wishes to communicate and tries to read the shared data, it must attach to the starting process's shared address space.

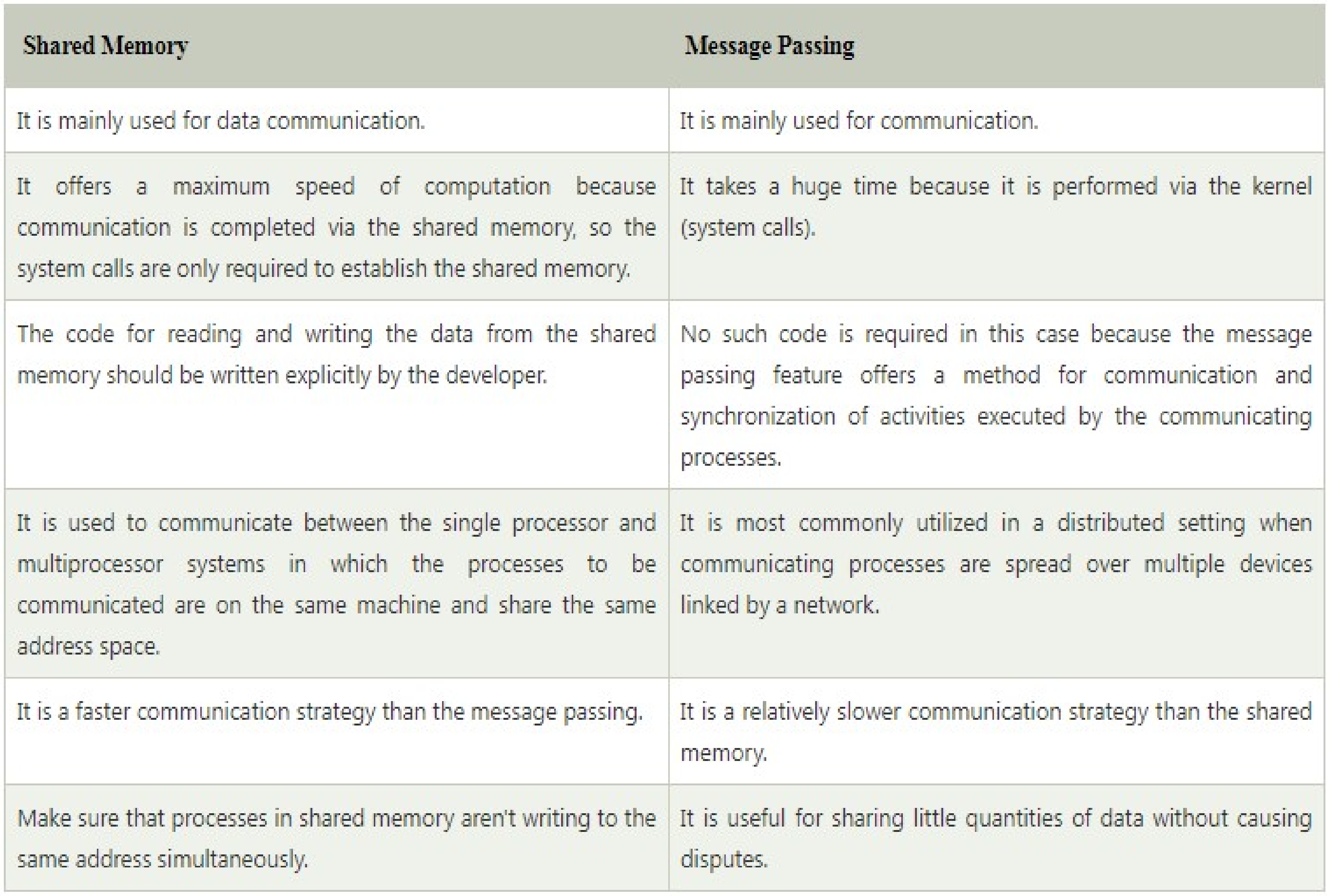
What is Message Passing?
In this message passing process model, the processes communicate with others by exchanging messages. A communication link between the processes is required for this purpose, and it must provide at least two operations:
transmit (message) and ***receive (message)***.

Message sizes might be flexible or fixed.

Message Passing Scenario



Shared V/s Message Passing

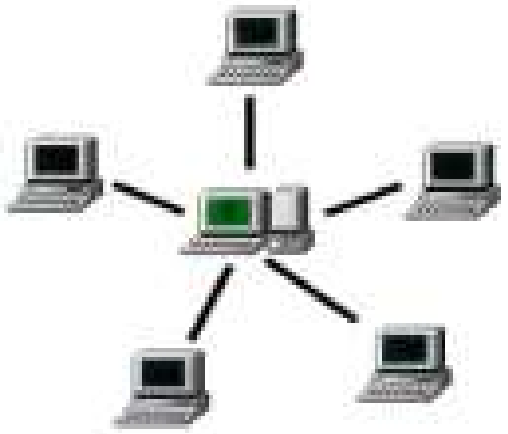


Shared Memory	Message Passing
It is mainly used for data communication.	It is mainly used for communication.
It offers a maximum speed of computation because communication is completed via the shared memory, so the system calls are only required to establish the shared memory.	It takes a huge time because it is performed via the kernel (system calls).
The code for reading and writing the data from the shared memory should be written explicitly by the developer.	No such code is required in this case because the message passing feature offers a method for communication and synchronization of activities executed by the communicating processes.
It is used to communicate between the single processor and multiprocessor systems in which the processes to be communicated are on the same machine and share the same address space.	It is most commonly utilized in a distributed setting when communicating processes are spread over multiple devices linked by a network.
It is a faster communication strategy than the message passing.	It is a relatively slower communication strategy than the shared memory.
Make sure that processes in shared memory aren't writing to the same address simultaneously.	It is useful for sharing little quantities of data without causing disputes.

Networking

Two Networking Models

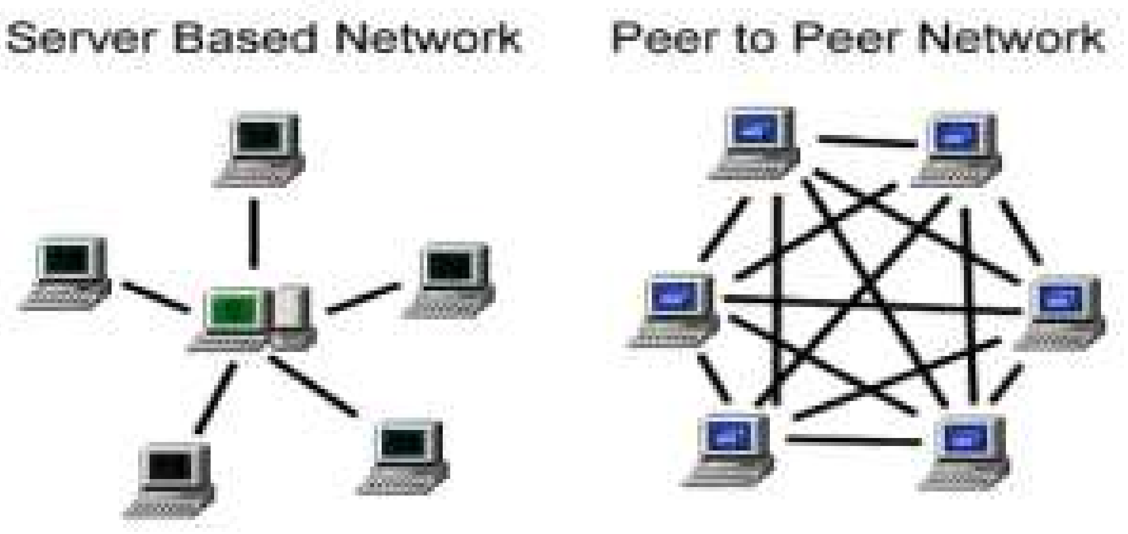
Server Based Network



- Networked computers take on different roles or functions in relation to each other.
  - **Client-Server** network:
    - Requires central servers responding to client requests

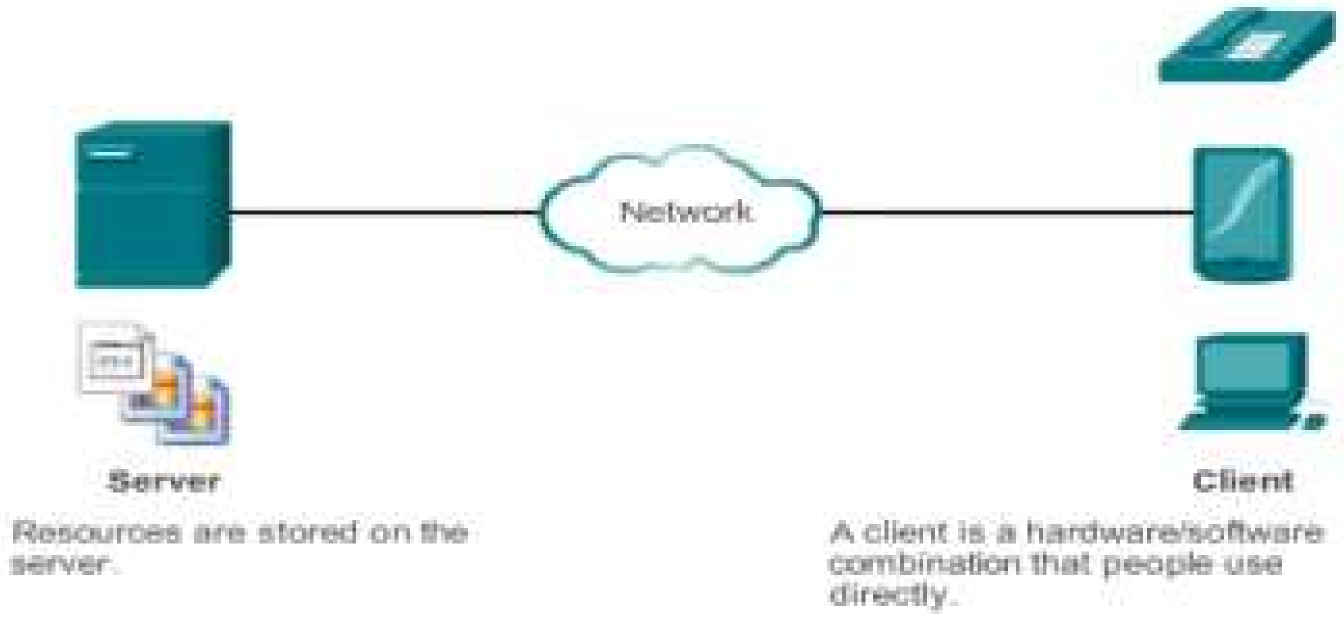
Networking Models

Two Networking Models



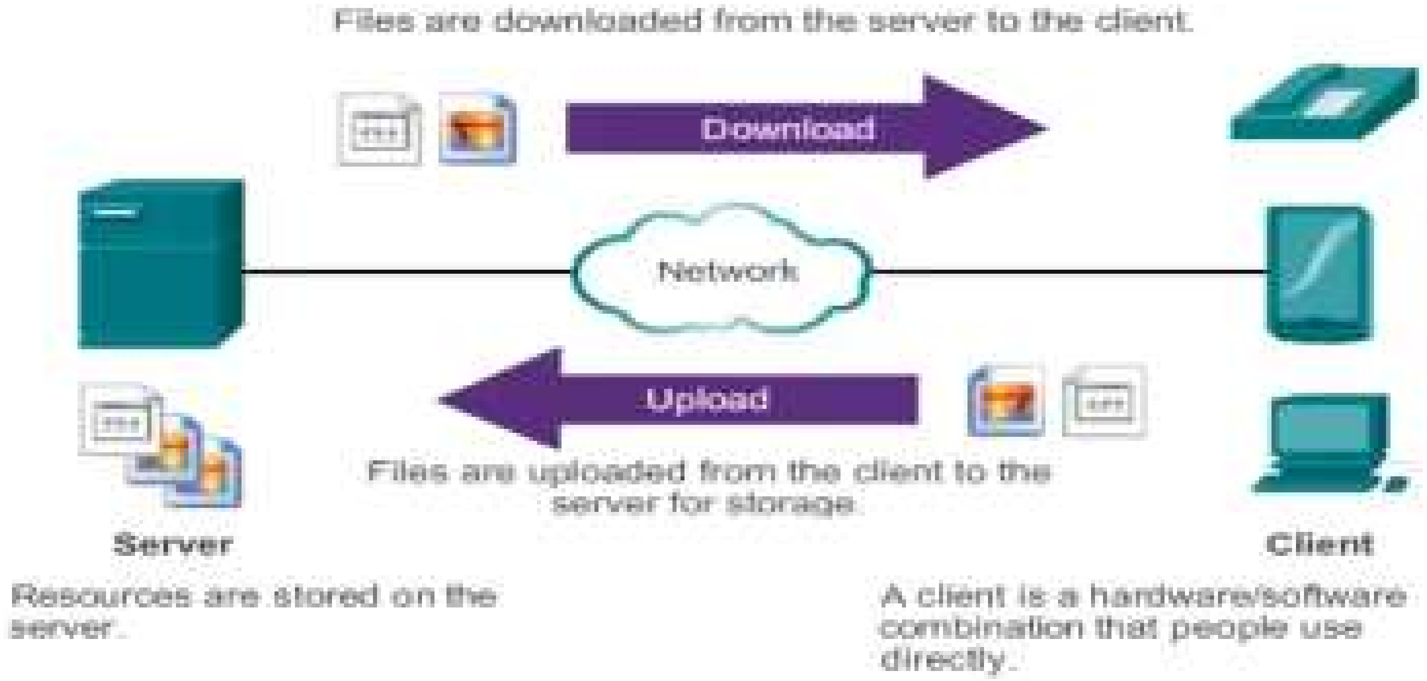
- Networked computers take on different roles or functions in relation to each other.
  - **Client-Server** network:
    - Requires central servers responding to client requests
  - **Peer-to-Peer (P2P)** network.
    - Variations: P2P networks and P2P applications.

Contd


Network
Server
Resources are stored on the server.
Client
A client is a hardware/software combination that people use directly.

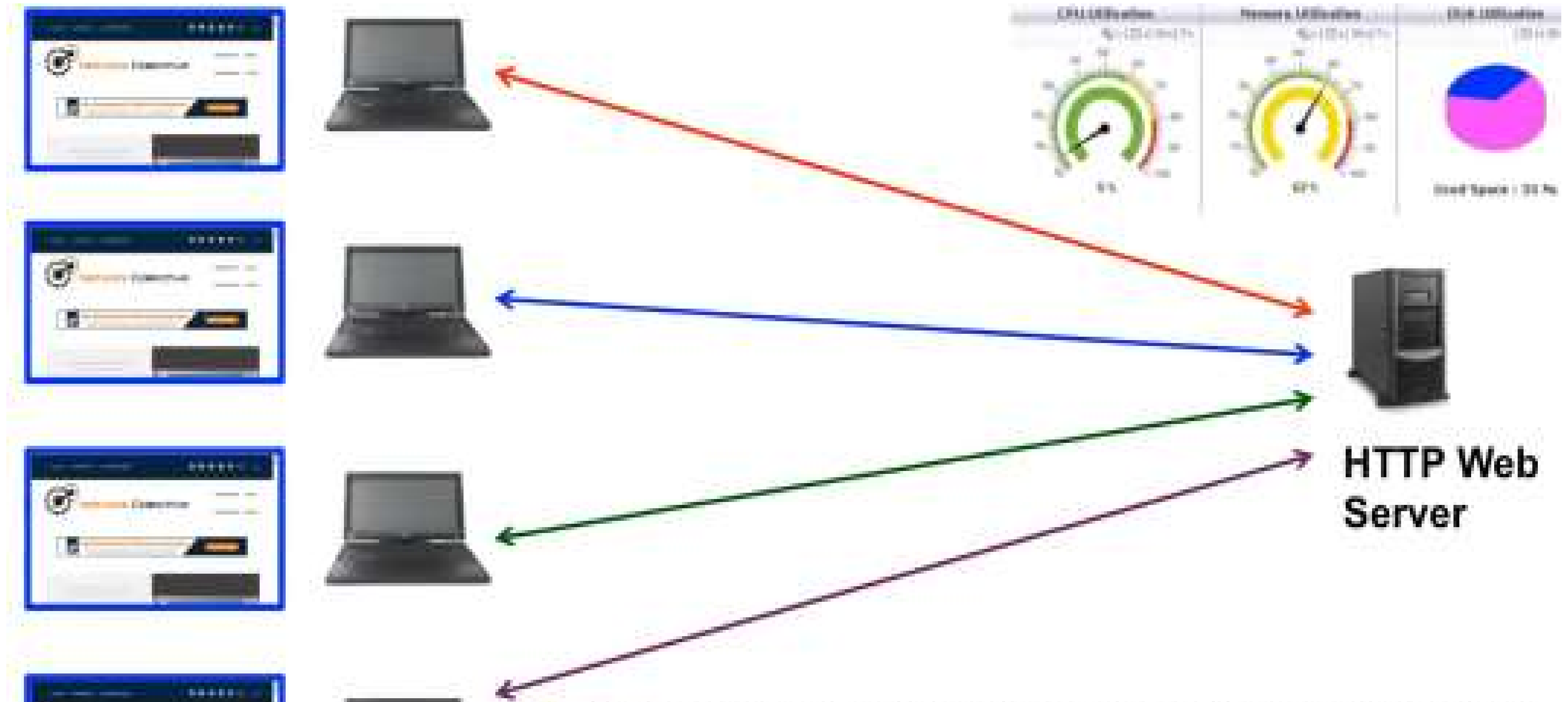
- In a client server model, 2 computers typically communicate with each other by using request/response protocols.
  - The requestor takes on the role of a **client**.
  - The responder takes on the role of a **server**.

Contd


Files are downloaded from the server to the client.
Download
Network
Upload
Files are uploaded from the client to the server for storage.
Server
Resources are stored on the server.
Client
A client is a hardware/software combination that people use directly.

- Files are downloaded from a **server**.
- Files / requests are uploaded from a **client**.

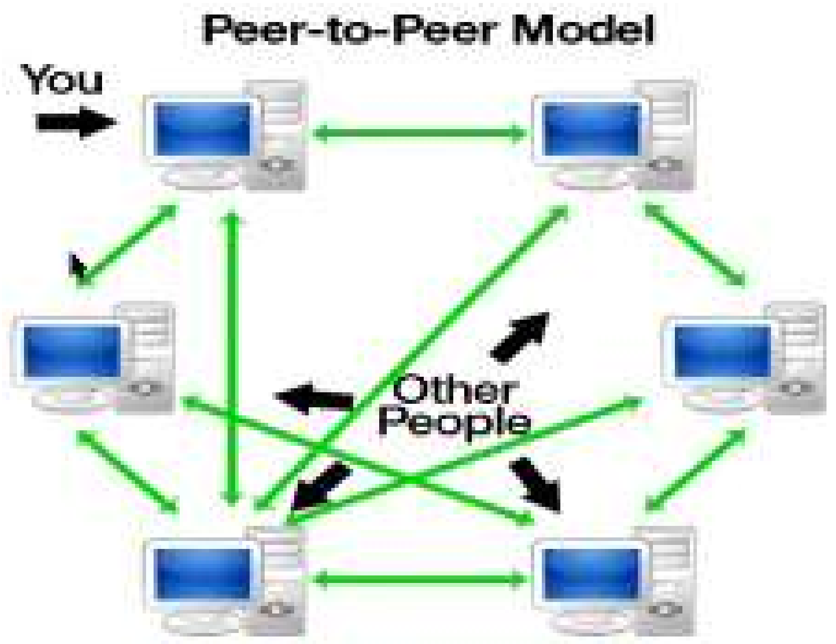
Contd



- Servers typically have multiple clients requesting information at the same time.
- Servers have specialized software and typically require more processing power, memory, and storage.

Peer to Peer

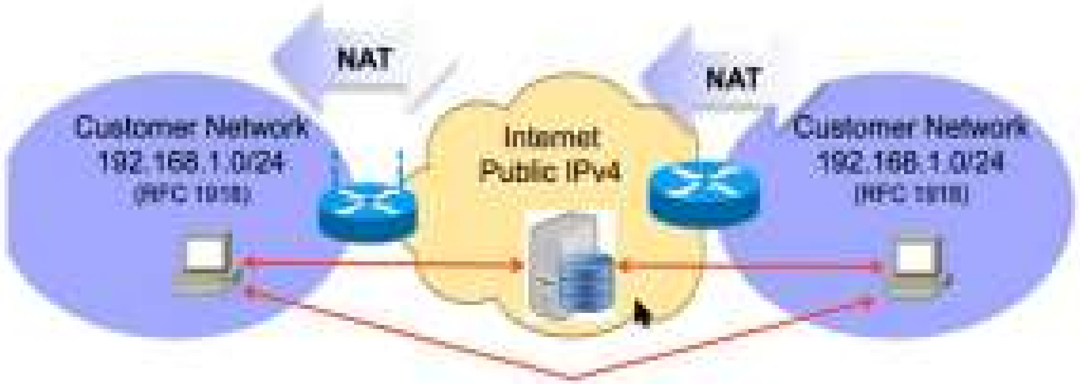
Peer-to-Peer Networking Model Concerns

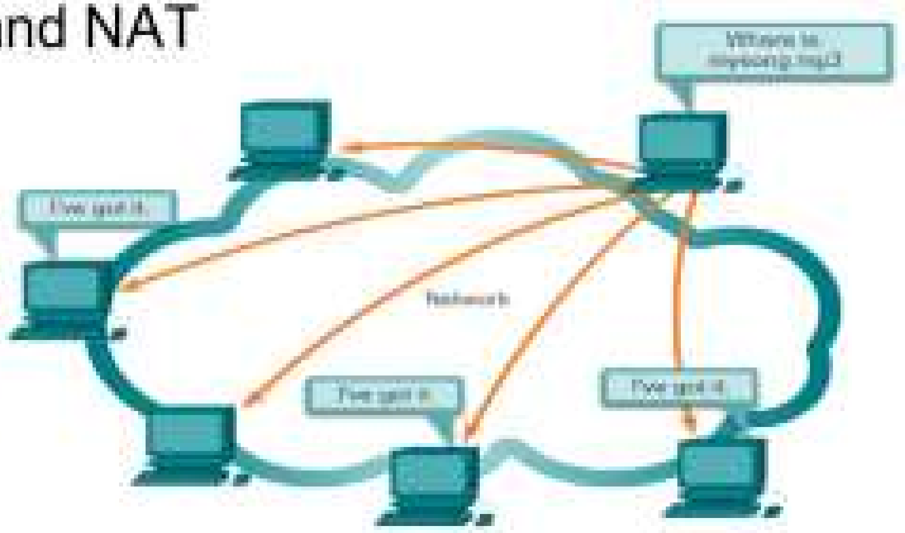


- P2P networks **decentralize the resources** on a network.
  - Data can be located anywhere and on any connected device.
- In a "pure P2P" architecture, data is accessed from a peer device without the use of a dedicated server.
- Each device (known as a peer) can function as both a server and a client.

NAT-Nating

Problem with Private IPv4 Addresses and NAT

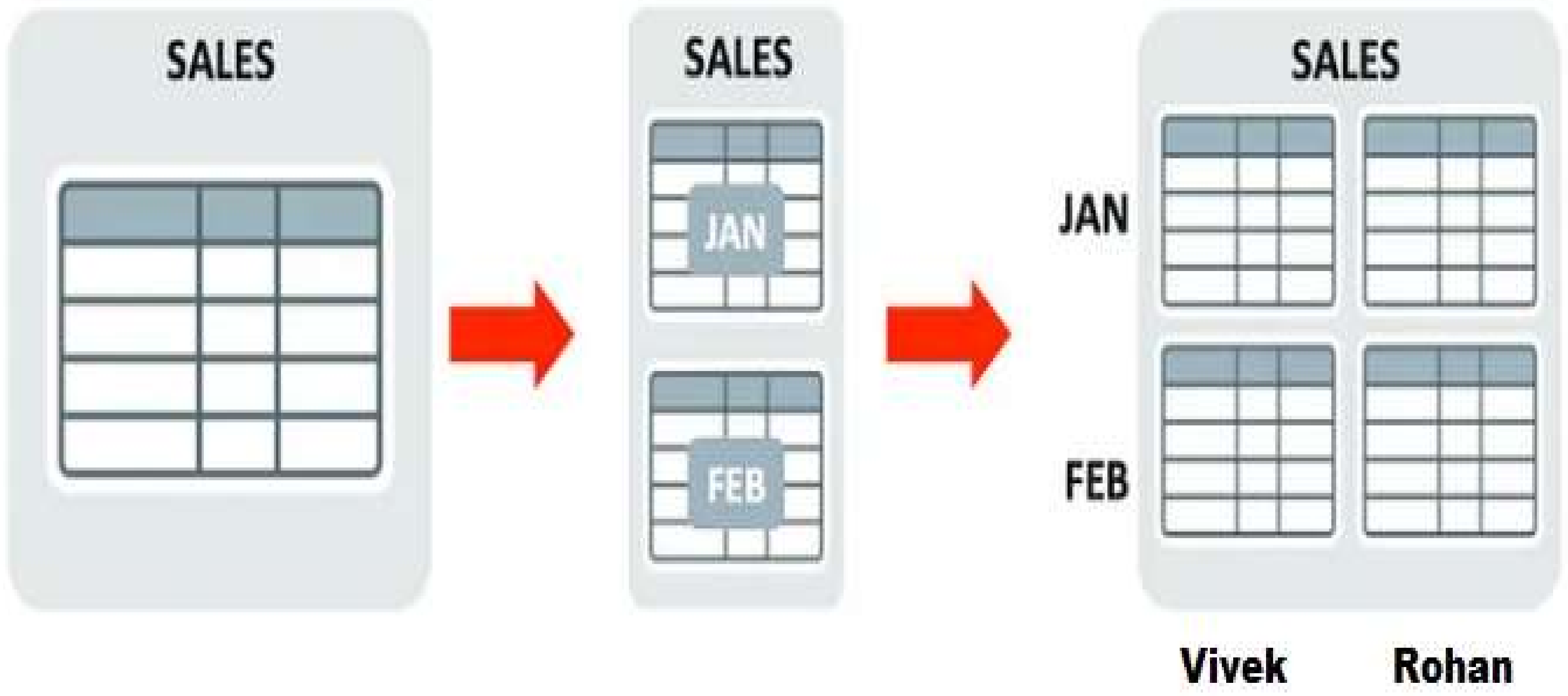




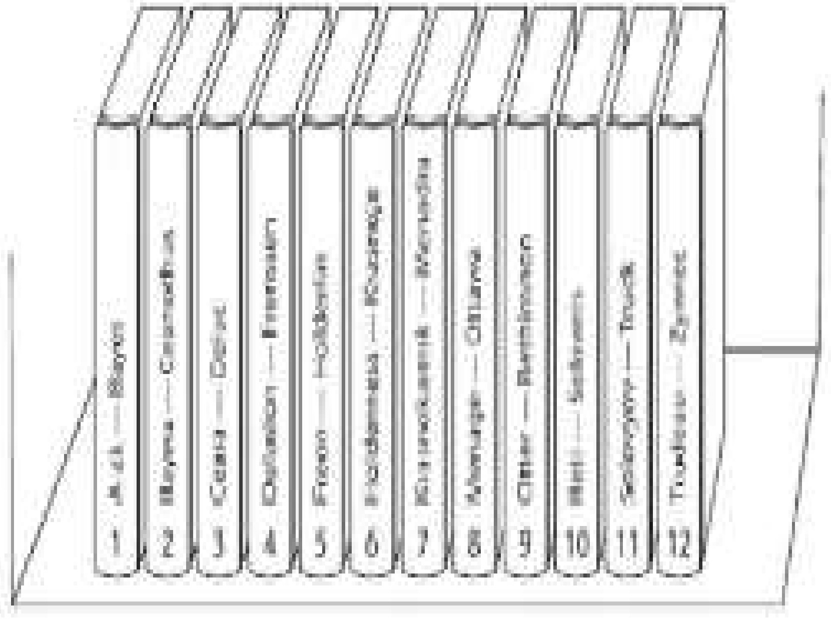
Gnutella allows P2P applications to search for shared resources on peers.

- Because most user have a private IPv4 address and NAT is required for Internet access, each peer must access **a central index server** to get the location of a resource stored on another peer.
  - The index server can also **help connect two peers**, but after connected, the **communication "may" take place between the two peers without additional communication to the index server – (TCP/UDP hole punching, NAT Transversal.**
  - Real solution: IPv6

Data Partitioning

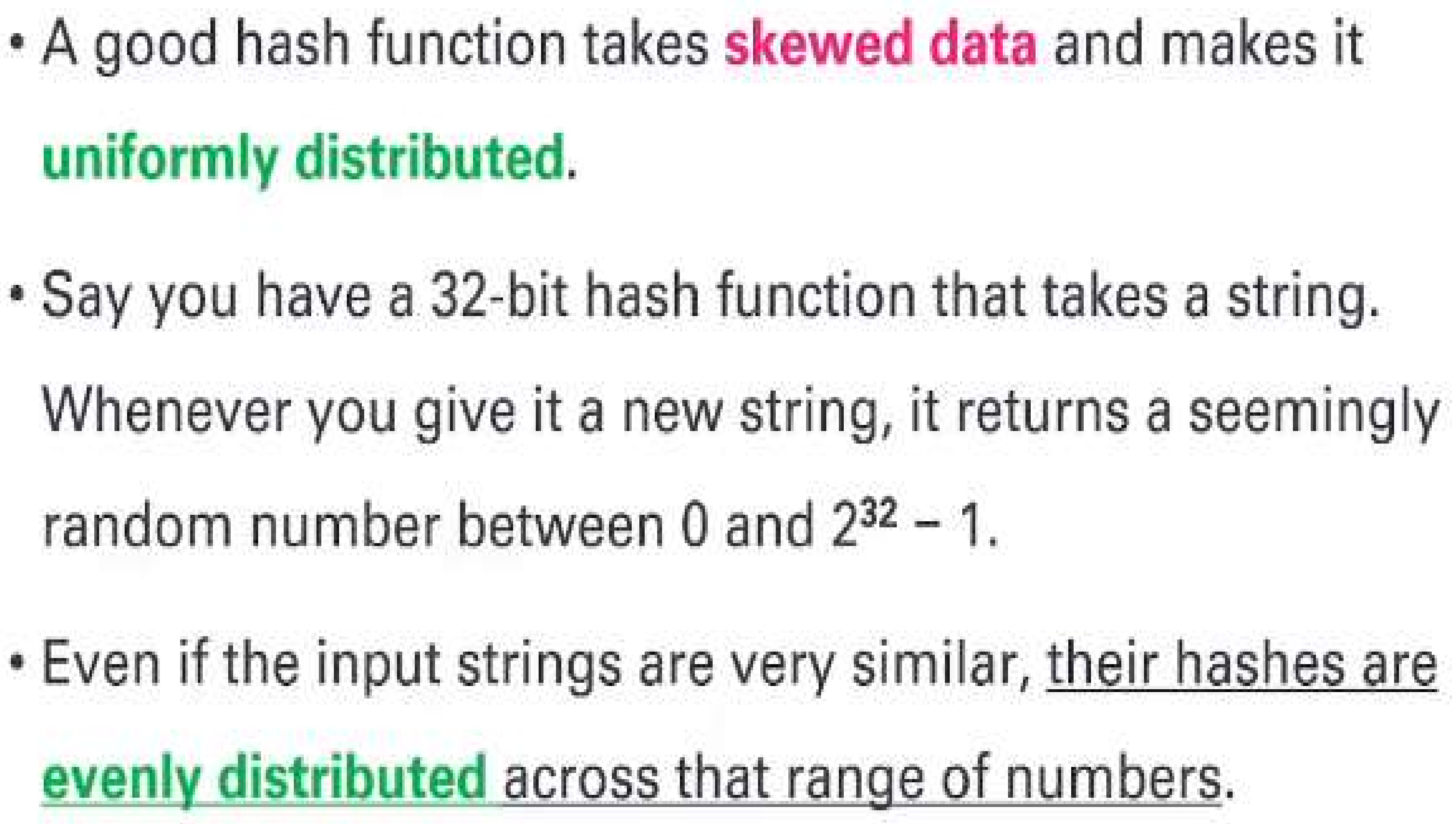


Key Value Based



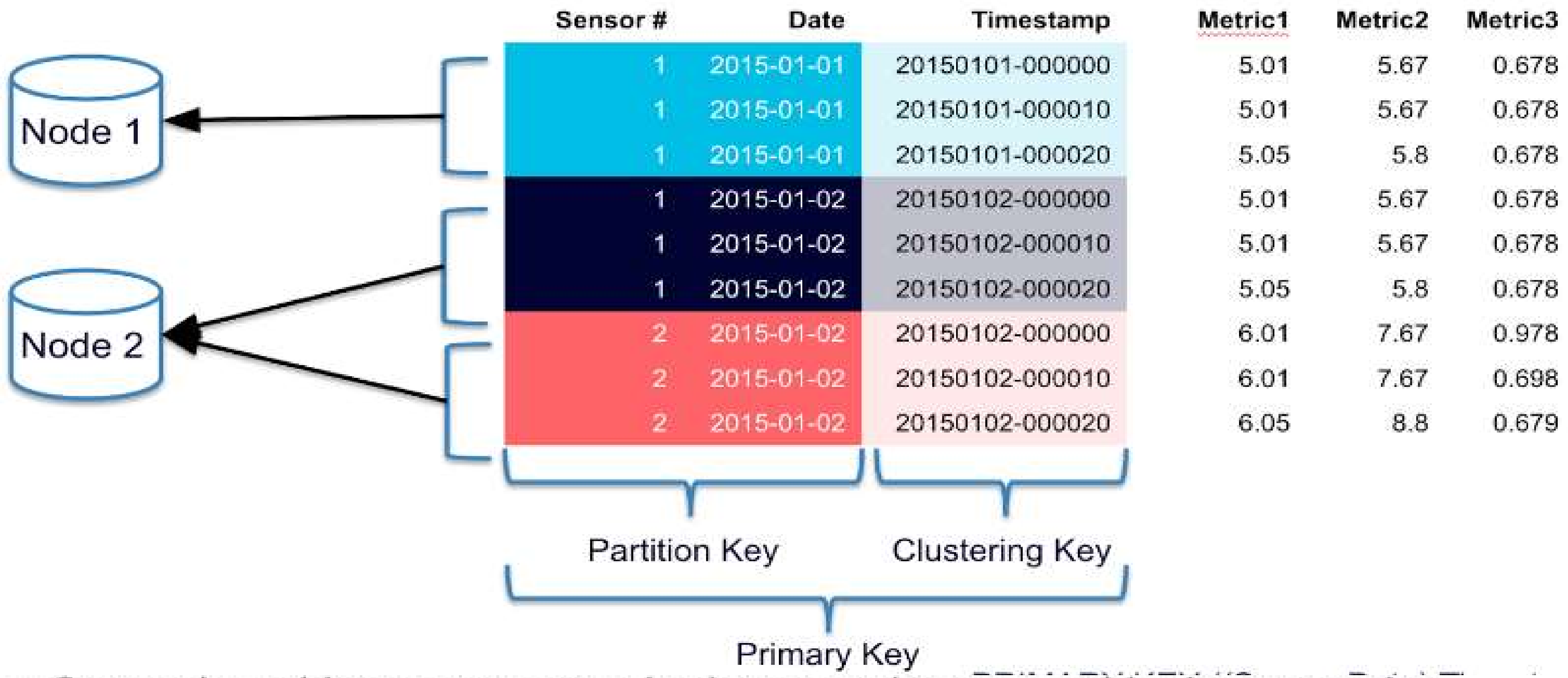
- Volume 1 contains words starting with **A and B**, but volume 12 contains words starting with **T, U, V, W, X, Y, and Z**.
- Simply having one volume per two letters of the alphabet would lead to some volumes being much bigger than others.
- In order to distribute the data evenly, the partition boundaries need to adapt to the data.

Hash Key Value Based



- A good hash function takes skewed data and makes it uniformly distributed.
- Say you have a 32-bit hash function that takes a string. Whenever you give it a new string, it returns a seemingly random number between 0 and $2^{32} - 1$.
- Even if the input strings are very similar, their hashes are evenly distributed across that range of numbers.

Cassandra Strategy



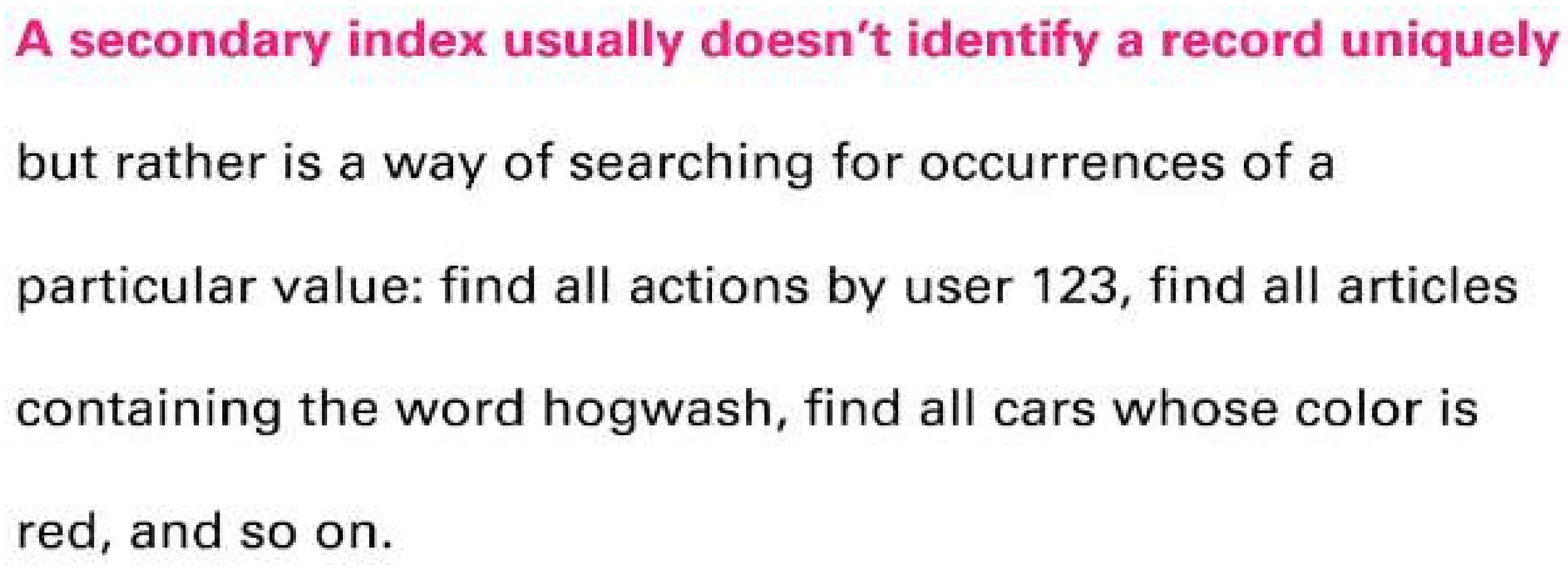
Sensor #	Date	Timestamp	Metric1	Metric2	Metric3
1	2015-01-01	20150101-000000	5.01	5.67	0.678
1	2015-01-01	20150101-000010	5.01	5.67	0.678
1	2015-01-01	20150101-000020	5.05	5.8	0.678
1	2015-01-02	20150102-000000	5.01	5.67	0.678
1	2015-01-02	20150102-000010	5.01	5.67	0.678
1	2015-01-02	20150102-000020	5.05	5.8	0.678
2	2015-01-02	20150102-000000	6.01	7.67	0.978
2	2015-01-02	20150102-000010	6.01	7.67	0.698
2	2015-01-02	20150102-000020	6.05	8.8	0.679

- Cassandra achieves a compromise between the two partitioning strategies .
- A table in Cassandra can be declared with a *compound primary key* consisting of several columns. Only the first part of that key is hashed to determine the partition, but the other columns are used as a concatenated index for sorting the data in Cassandra's SSTables.

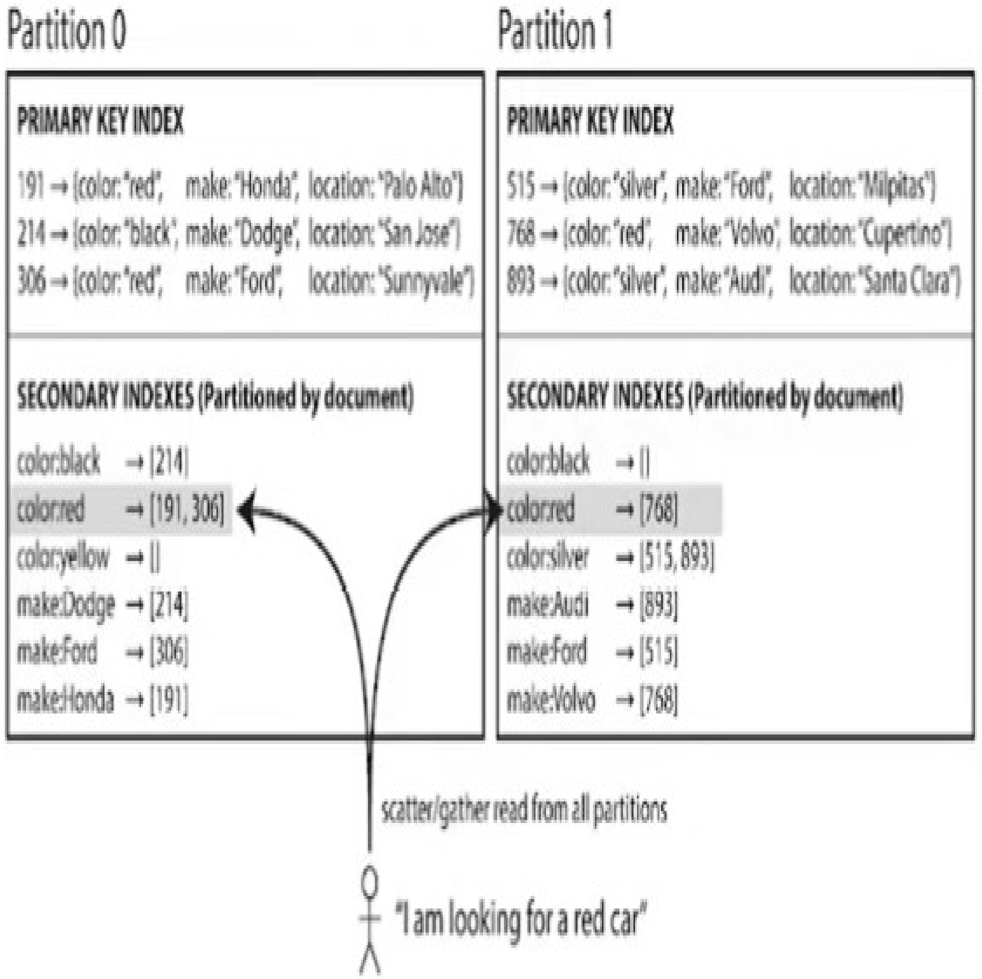
Partition key. The partition key is part of the **primary key and defines which node will store the data based on its hash value and distributes data across the nodes** in the Cassandra cluster. All rows with the same partition key will be stored together on the same node.

Clustering key(s). The clustering key(s) follow the partition key in the definition of the primary key in Cassandra. **The clustering key sorts the data within a partition**, allowing rows with the same partition key to be ordered based on their different clustering key values.

Secondary Index Approach

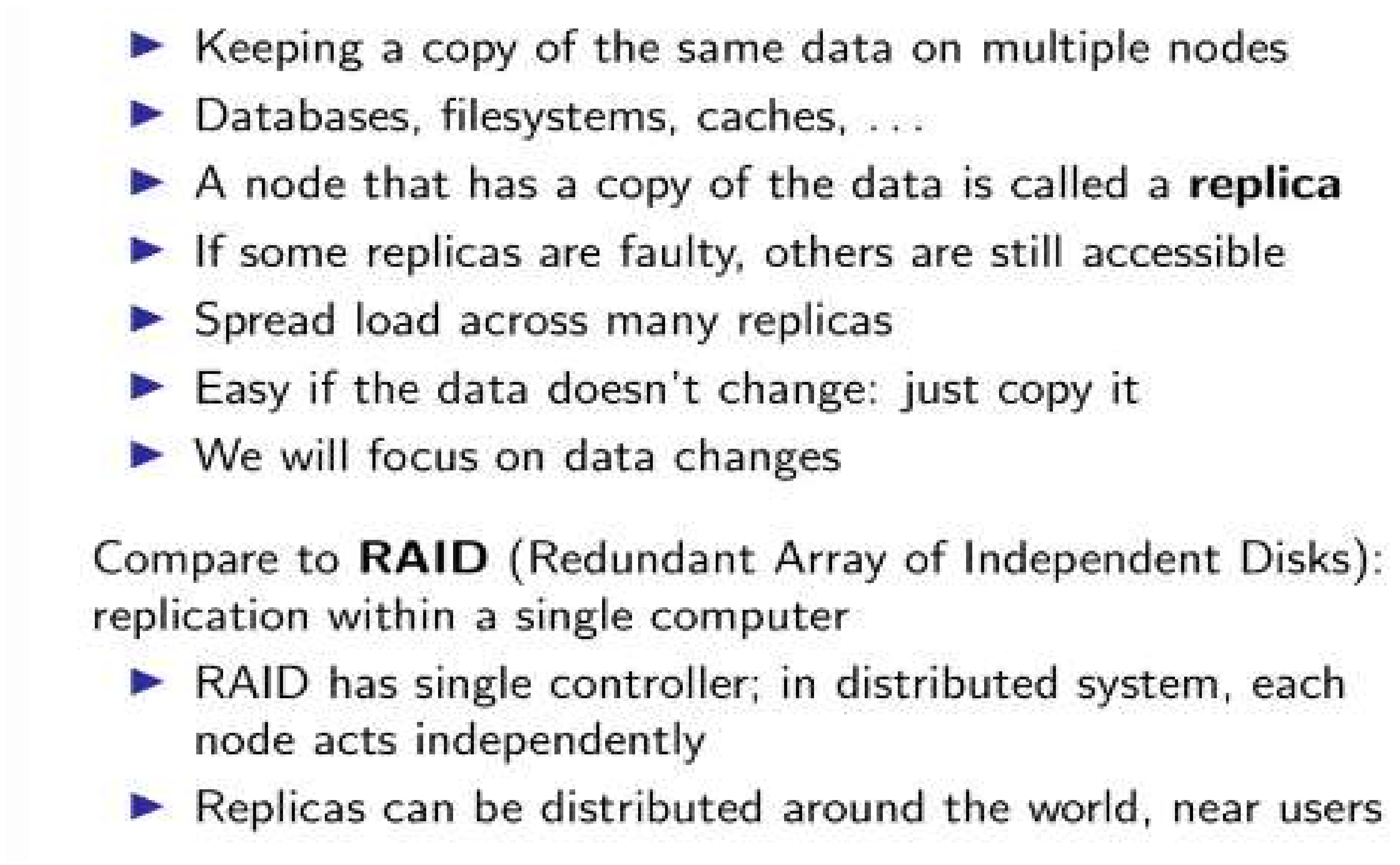


A secondary index usually doesn't identify a record uniquely but rather is a way of searching for occurrences of a particular value: find all actions by user 123, find all articles containing the word hogwash, find all cars whose color is red, and so on.


Partition 0
PRIMARY KEY INDEX
191 → {color:"red", make:"Honda", location:"Palo Alto"}
214 → {color:"black", make:"Dodge", location:"San Jose"}
306 → {color:"red", make:"Ford", location:"Sunnyvale"}
SECONDARY INDEXES (Partitioned by document)
color:black → [214]
color:red → [191, 306]
color:yellow → []
make:Dodge → [214]
make:Ford → [306]
make:Honda → [191]
Partition 1
PRIMARY KEY INDEX
515 → {color:"silver", make:"Ford", location:"Milpitas"}
768 → {color:"red", make:"Volvo", location:"Cupertino"}
893 → {color:"silver", make:"Audi", location:"Santa Clara"}
SECONDARY INDEXES (Partitioned by document)
color:black → []
color:red → [768]
color:silver → [515, 893]
make:Audi → [893]
make:Ford → [515]
make:Volvo → [768]
scatter/gather read from all partitions
"I am looking for a red car"

- In this indexing approach, **each partition is completely separate: each partition maintains its own secondary indexes**, covering only the documents in that partition.
- It doesn't care what data is stored in other partitions.
- Whenever you write to the database—to add, remove, or update a document—you only need to deal with the partition that contains the document ID that you are writing.
- For that reason, a document-partitioned index is also known as a local index (as opposed to a global index, described in the next section).

Strategies for Data Access Replication

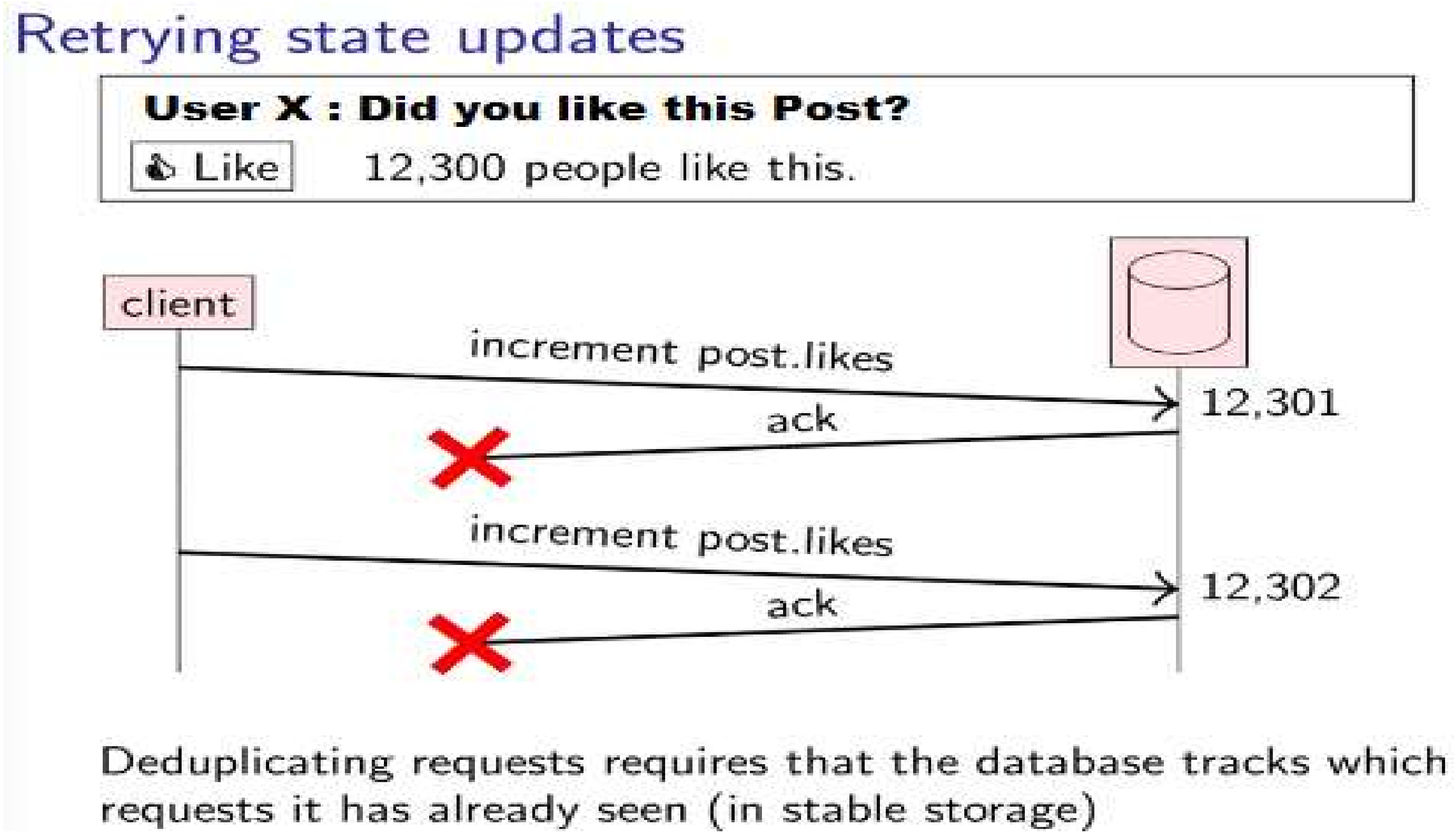


- Keeping a copy of the same data on multiple nodes
- Databases, filesystems, caches, . . .
- A node that has a copy of the data is called a **replica**
- If some replicas are faulty, others are still accessible
- Spread load across many replicas
- Easy if the data doesn't change: just copy it
- We will focus on data changes

Compare to **RAID** (Redundant Array of Independent Disks): replication within a single computer

- RAID has single controller; in distributed system, each node acts independently
- Replicas can be distributed around the world, near users

Updates

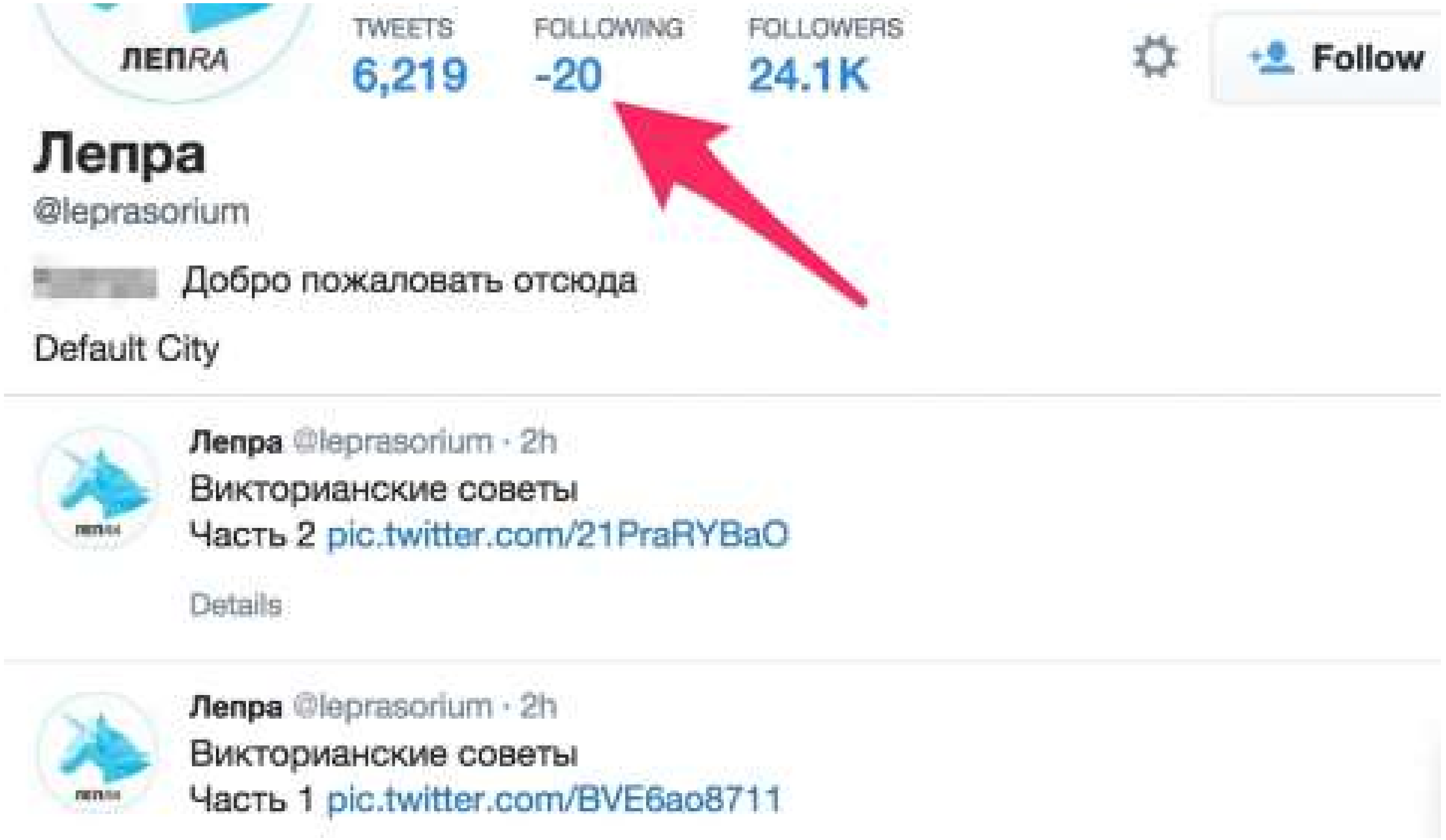


Retrying state updates

User X : Did you like this Post?

Like 12,300 people like this.

Deduplicating requests requires that the database tracks which requests it has already seen (in stable storage)


TWEETS 6,219 FOLLOWING -20 FOLLOWERS 24.1K

Follow

Лепра
@leprasorium

[redacted] Добро пожаловать отсюда

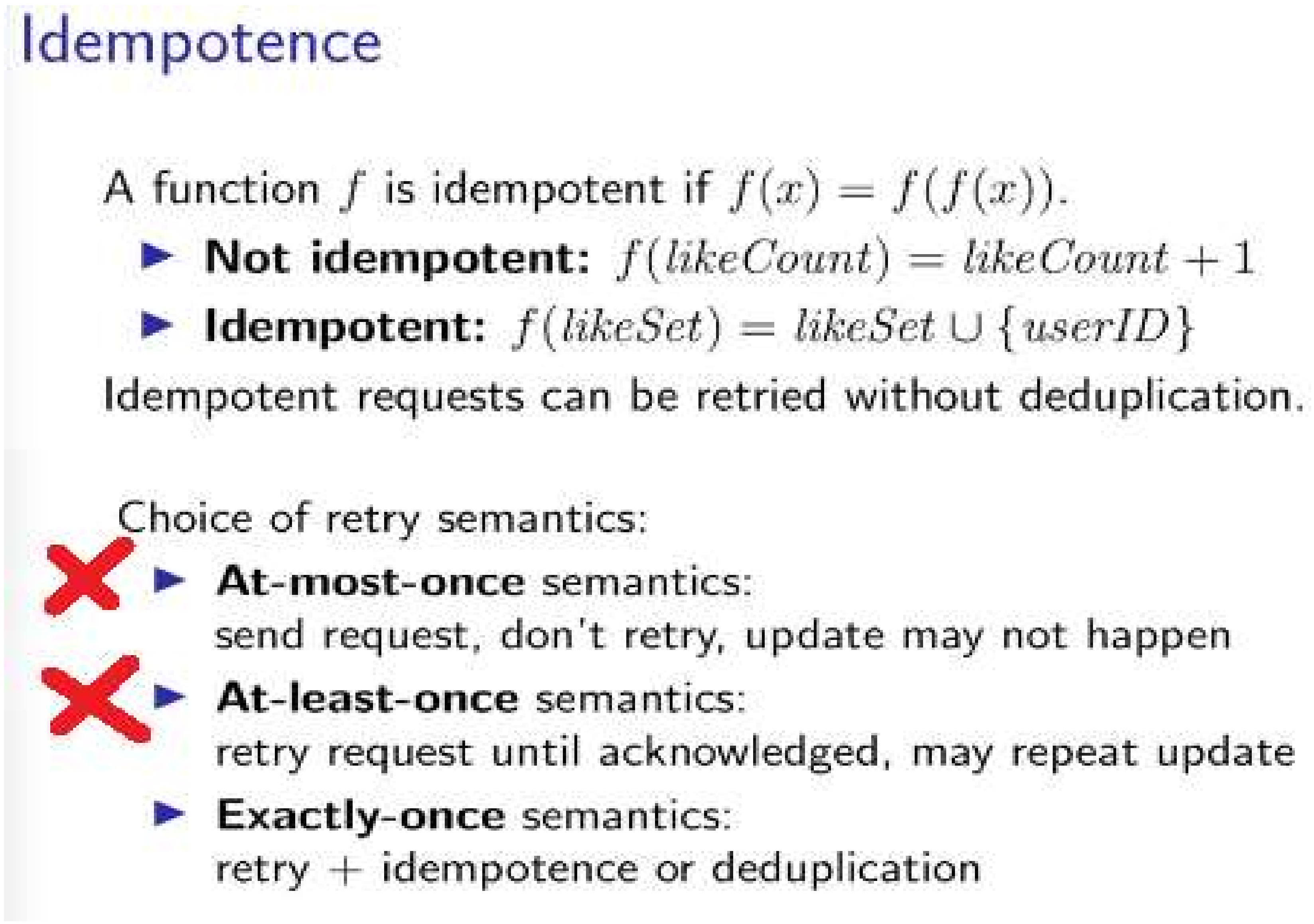
Default City

Лепра @leprasorium · 2h
Викторианские советы
Часть 2 pic.twitter.com/21PraRYBaO

Details

Лепра @leprasorium · 2h
Викторианские советы
Часть 1 pic.twitter.com/BVE6ao8711

Idempotence



Idempotence

A function f is idempotent if $f(x) = f(f(x))$.

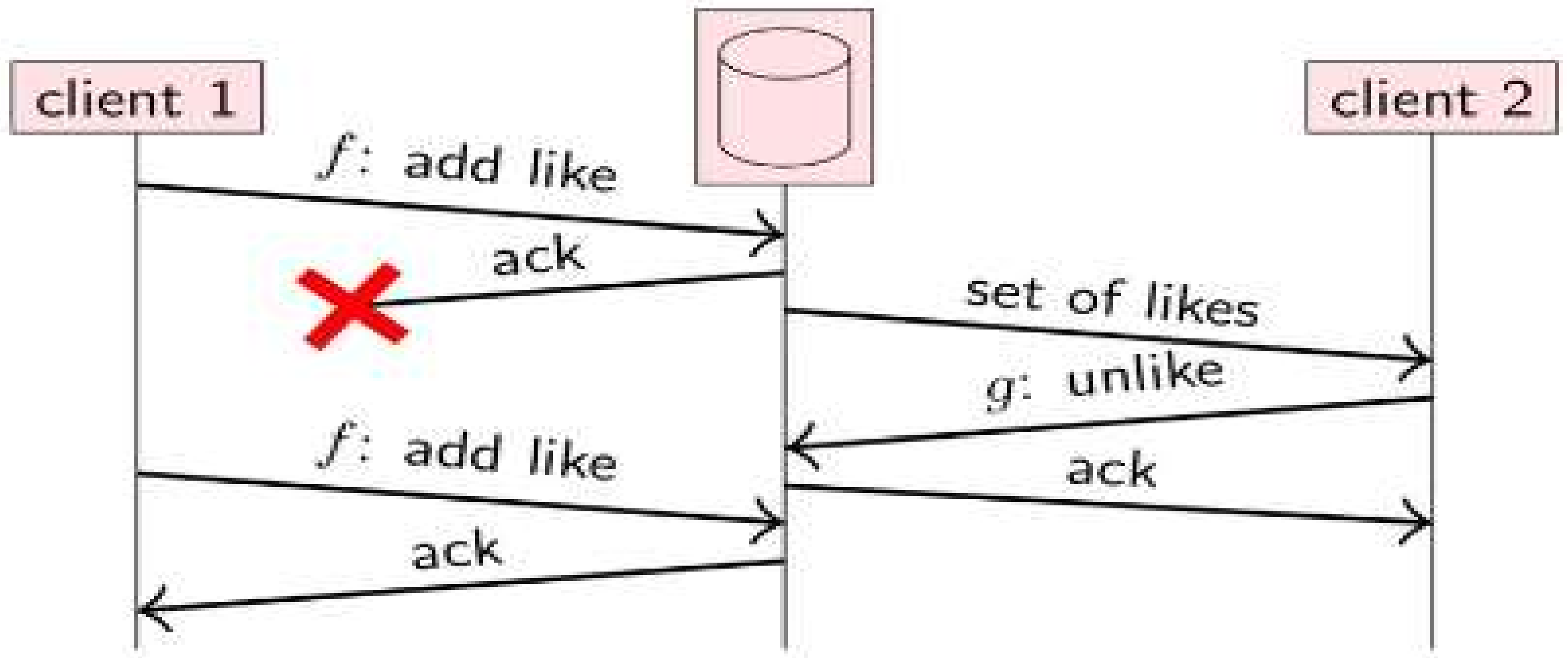
- **Not idempotent:** $f(likeCount) = likeCount + 1$
- **Idempotent:** $f(likeSet) = likeSet \cup \{userID\}$

Idempotent requests can be retried without deduplication.

Choice of retry semantics:

- ✗ **At-most-once** semantics:
  send request, don't retry, update may not happen
- ✗ **At-least-once** semantics:
  retry request until acknowledged, may repeat update
- **Exactly-once** semantics:
  retry + idempotence or deduplication

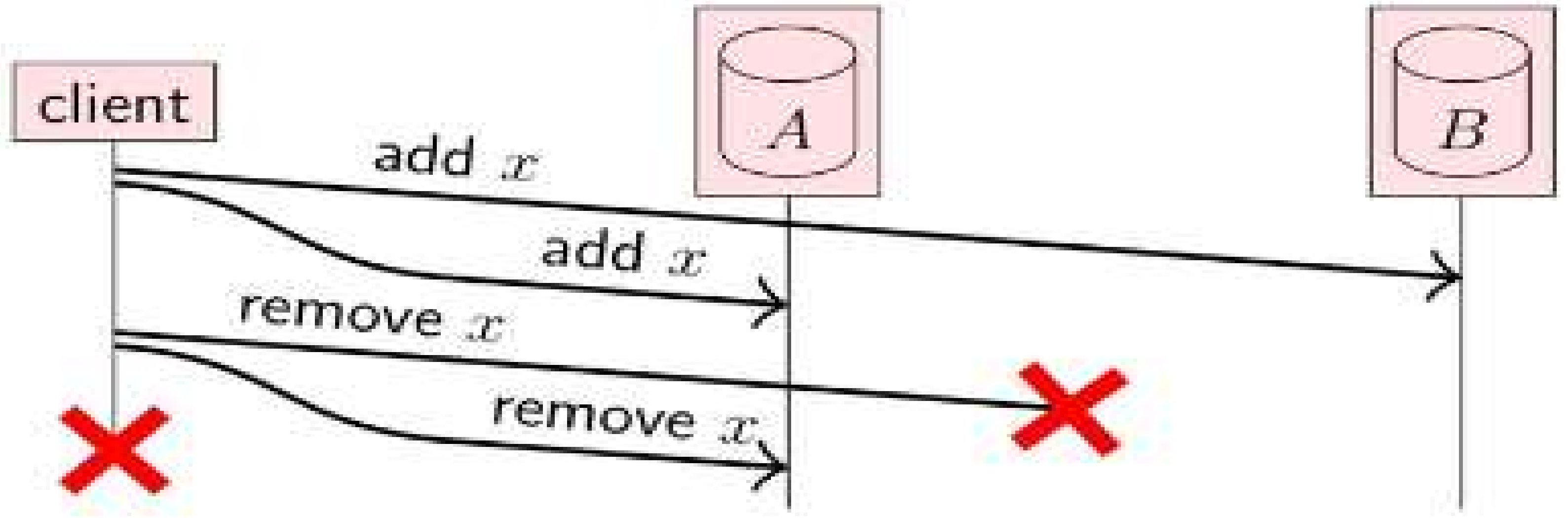
Idempotent : Does not work here



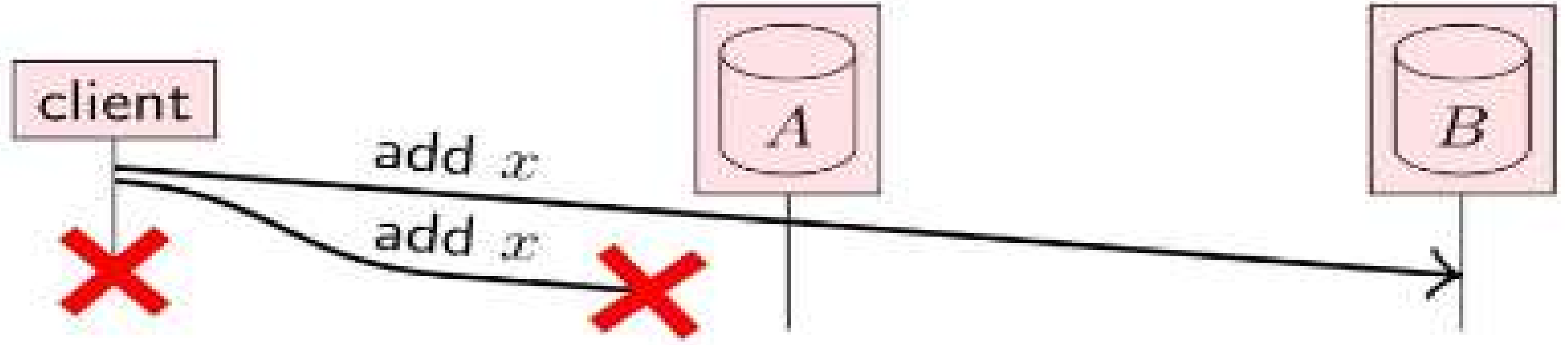
$$f(likes) = likes \cup \{userID\}$$
$$g(likes) = likes \setminus \{userID\}$$

Idempotent? $f(f(x)) = f(x)$ but $f(g(f(x)) \neq g(f(x))$

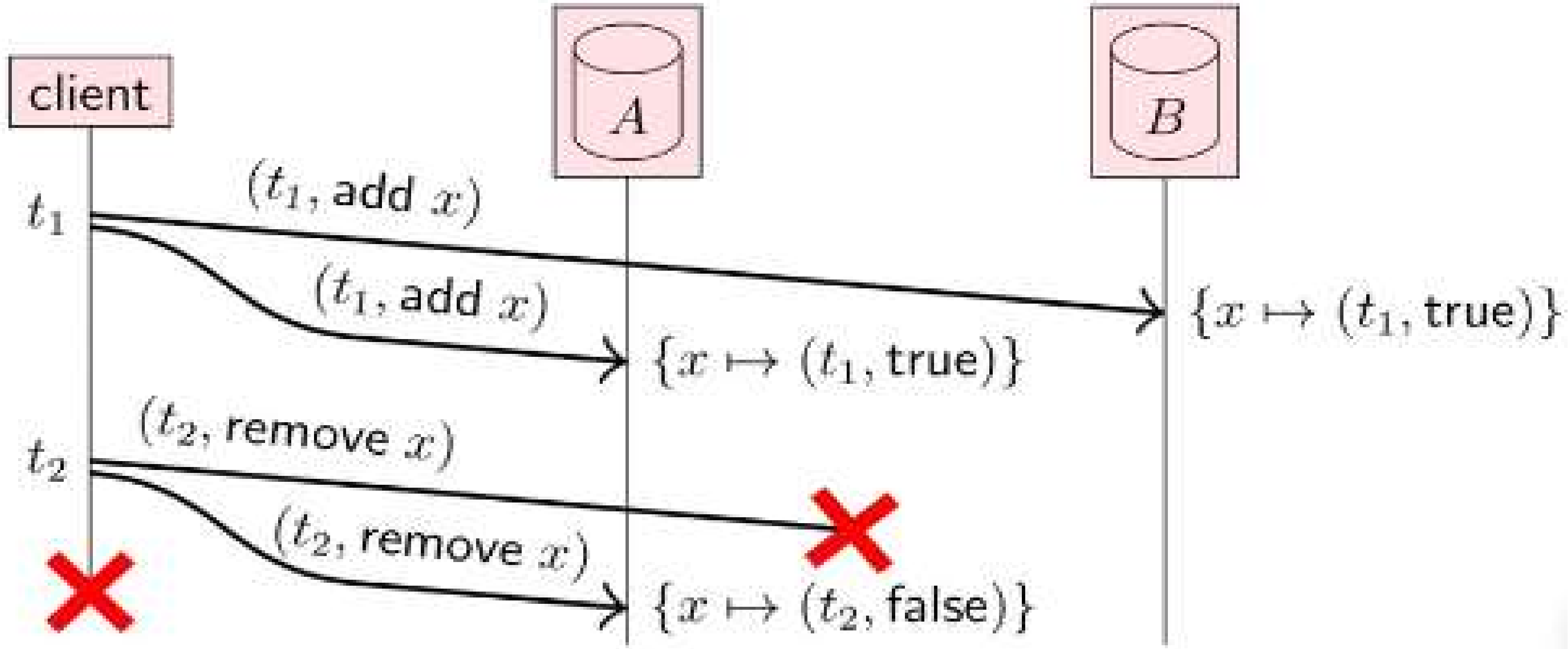
Add and Remove



Final state ($x \notin A$, $x \in B$) is the same as in this case:



Tombstone

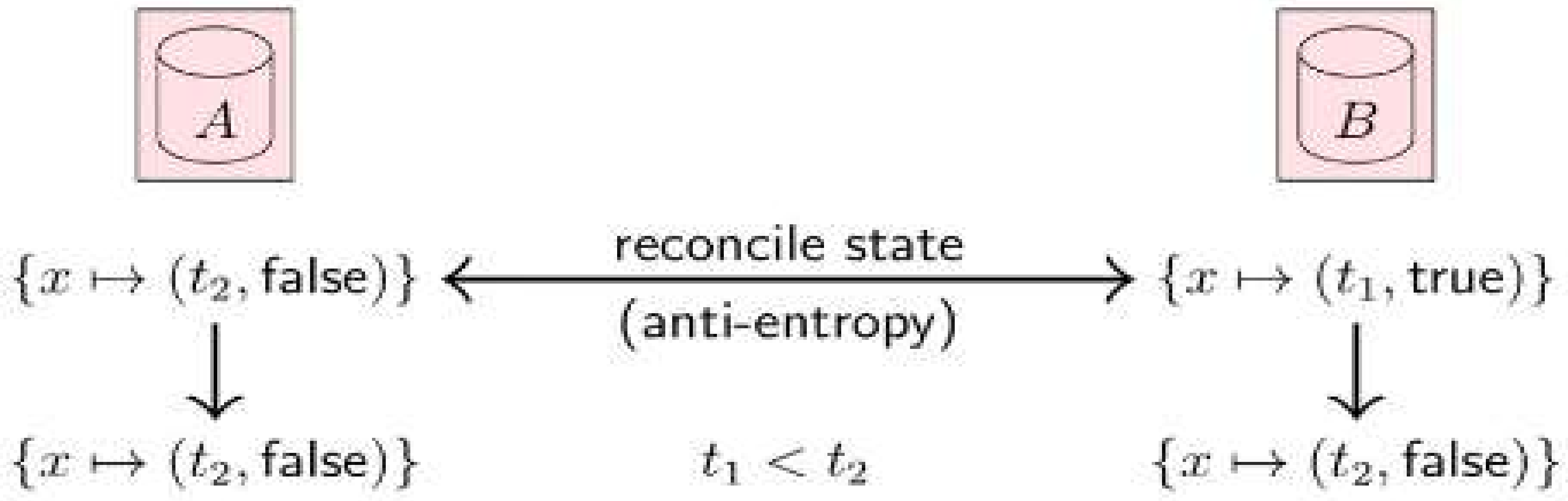

client
A
B
t_1
$(t_1, \text{add } x)$
$(t_1, \text{add } x)$
$\{x \mapsto (t_1, \text{true})\}$
$\{x \mapsto (t_1, \text{true})\}$
t_2
$(t_2, \text{remove } x)$
$(t_2, \text{remove } x)$
$\{x \mapsto (t_2, \text{false})\}$

"remove x" doesn't actually remove x: it labels x with "false" to indicate it is invisible (a **tombstone**)

Every record has **logical timestamp** of last write

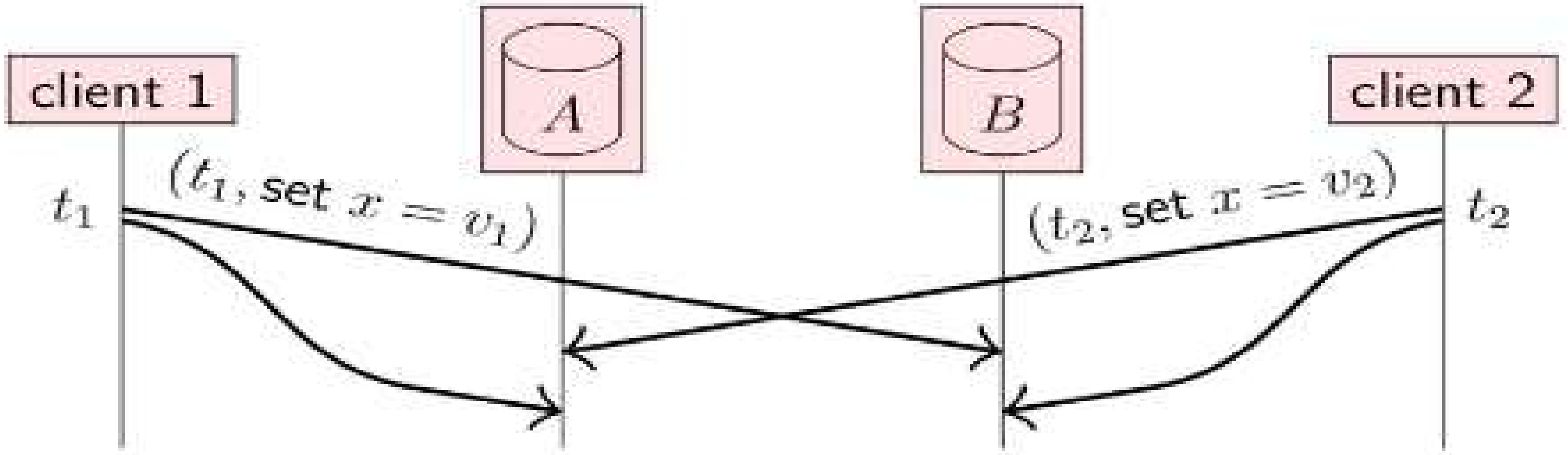
Reconciling Replicas

Replicas periodically communicate among themselves to check for any inconsistencies.


A
B
$\{x \mapsto (t_2, \text{false})\}$
reconcile state
(anti-entropy)
$\{x \mapsto (t_1, \text{true})\}$
$\{x \mapsto (t_2, \text{false})\}$
$t_1 < t_2$
$\{x \mapsto (t_2, \text{false})\}$

Propagate the record with the latest timestamp, discard the records with earlier timestamps (for a given key).

Conflict Due to Concurrent Write



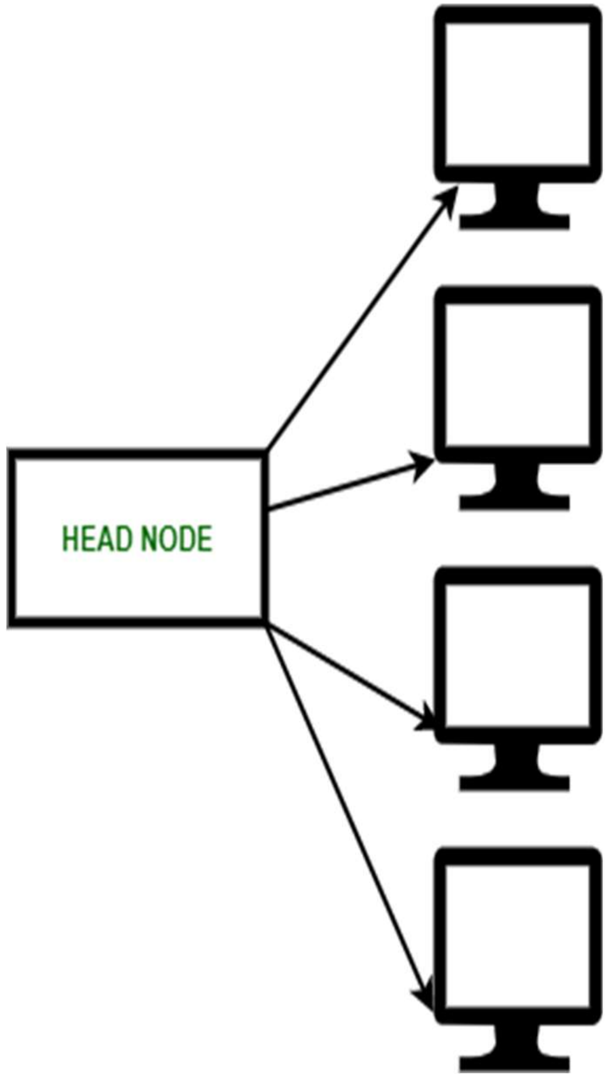
Two common approaches:

- **Last writer wins** (LWW):
  Use timestamps with total order (e.g. Lamport clock)
  Keep v_2 and discard v_1 if $t_2 > t_1$. Note: **data loss**!
- **Multi-value register**:
  Use timestamps with partial order (e.g. vector clock)
  v_2 replaces v_1 if $t_2 > t_1$; preserve both $\{v_1, v_2\}$ if $t_1 \parallel t_2$

Cluster Computing

Introduction :

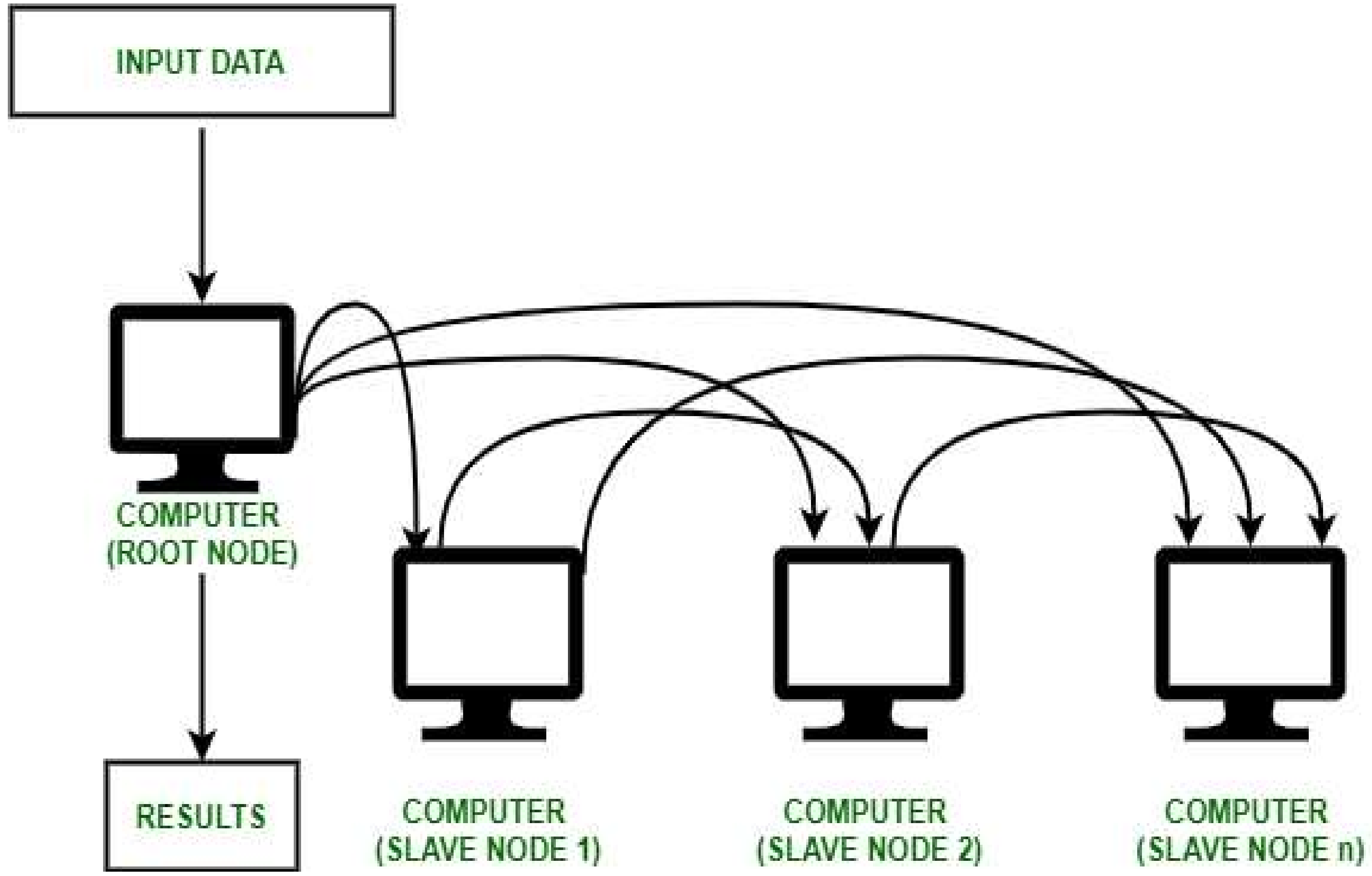
- Cluster computing is a collection of tightly or loosely connected computers that work together so that they act as a single entity.
- The connected computers execute operations all together thus creating the idea of a single system.
- The clusters are generally connected through fast local area networks (LANs)



Why is Cluster Computing important?

1. Cluster computing gives a relatively inexpensive, unconventional to the large server or mainframe computer solutions.
2. It resolves the demand for content criticality and process services in a faster way.
3. Many organizations and IT companies are implementing cluster computing to augment their scalability, availability, processing speed and resource management at economic prices.
4. It ensures that computational power is always available.
5. It provides a single general strategy for the implementation and application of parallel high-performance systems independent of certain hardware vendors and their product decisions.

Cluster computing layout



Types of Cluster computing :

- **High performance (HP) clusters :**
  - HP clusters use computer clusters and supercomputers to solve advanced computational problems.
  - These are used to perform functions that need nodes to communicate as they perform their jobs.
  - These are designed to take benefit of the parallel processing power of several nodes.

Cont.…

- **Load-balancing clusters :**
  - Incoming requests are distributed for resources among several nodes running similar programs or having similar content.
  - This prevents any single node from receiving a disproportionate amount of tasks.
  - This type of distribution is generally used in a web-hosting environment.

Cont.…

- **High Availability (HA) Clusters :**
  - HA clusters are designed to maintain redundant nodes that can act as backup systems in case any failure occurs.
  - Consistent computing services like business activities, complicated databases, customer services like e-websites and network file distribution are provided.
  - They are designed to give uninterrupted data availability to the customers.

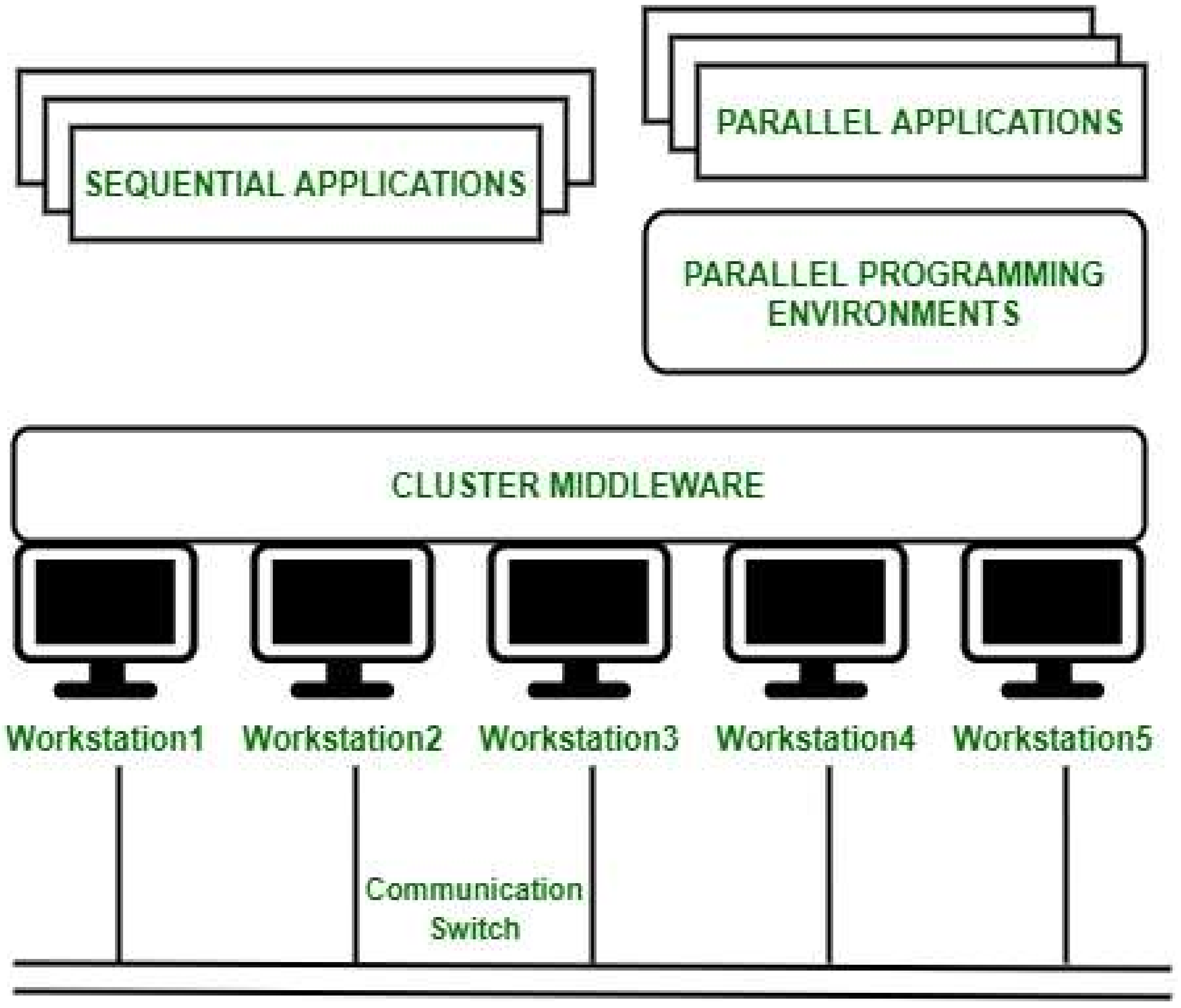
Classification of Cluster :

- **Open Cluster :**
  - IPs are needed by every node and those are accessed only through the internet or the web. This type of cluster causes enhanced security concerns.
- **Close Cluster :**
  - The nodes are hidden behind the gateway node, and they provide increased protection. They need fewer IP addresses and are good for computational tasks.

Cluster Computing Architecture :

- It is designed with an array of interconnected individual computers and computer systems operating collectively as a single standalone system.
- It is a group of workstations or computers working together as a single, integrated computing resource connected via high-speed interconnects.
- A node – Either a single or multiprocessor network having memory, input and output functions and an operating system.
- Two or more nodes are connected on a single line or every node might be connected individually through a LAN connection.

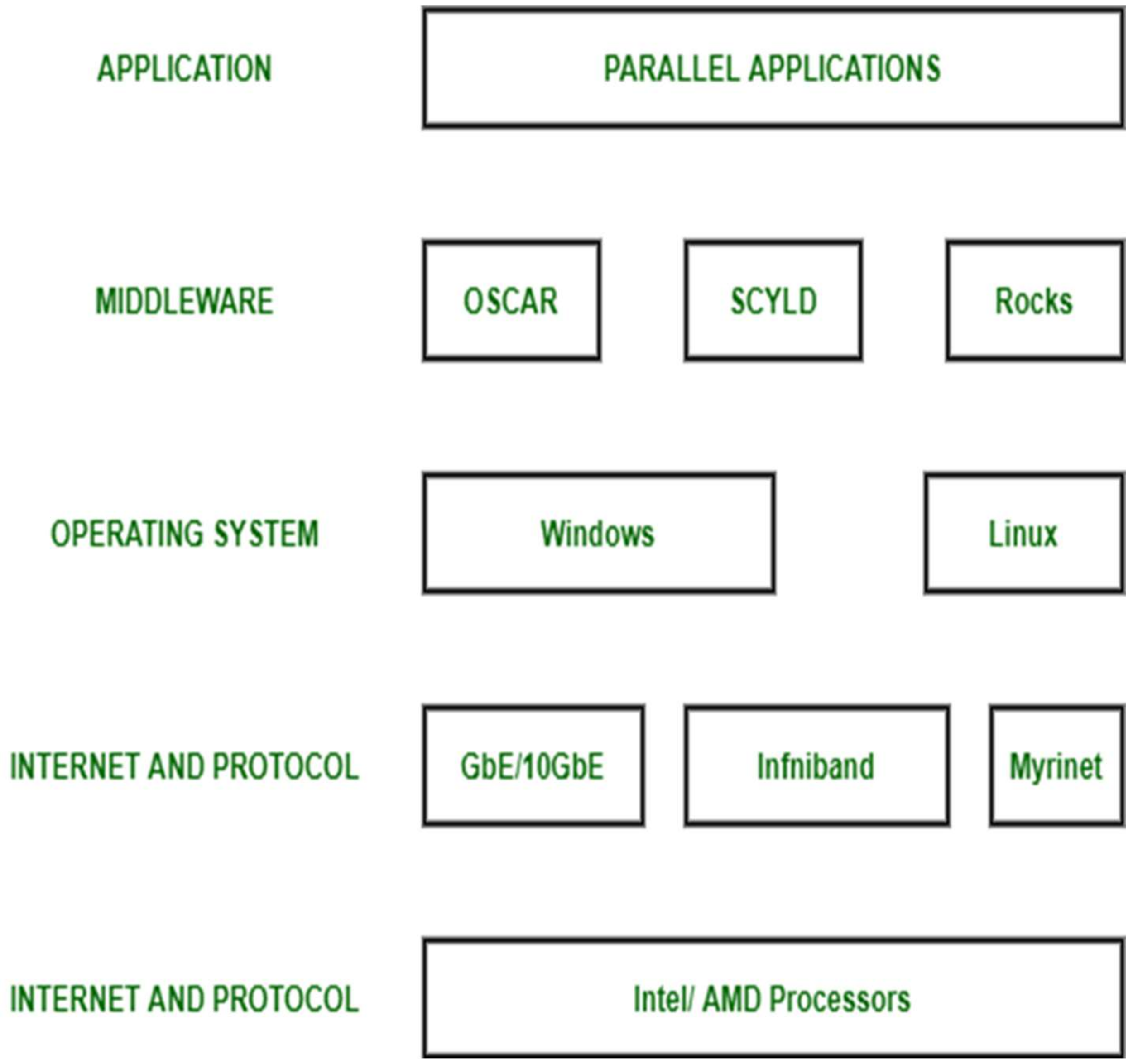
Cluster Computing Architecture Diagram



Components of a Cluster Computer :

1. Cluster Nodes
2. Cluster Operating System
3. The switch or node interconnect
4. Network switching hardware

Component Diagram



Advantages of Cluster Computing :

- **High Performance :**
  - The systems offer better and enhanced performance than that of mainframe computer networks.
- **Easy to manage :**
  - Cluster Computing is manageable and easy to implement.
- **Scalable :**
  - Resources can be added to the clusters accordingly.
- **Expandability :**
  - Computer clusters can be expanded easily by adding additional computers to the network. Cluster computing is capable of combining several additional resources or networks to the existing computer system.
- **Availability :**
  - The other nodes will be active when one node gets failed and will function as a proxy for the failed node. This makes sure for enhanced availability.
- **Flexibility :**
  - It can be upgraded to the superior specification or additional nodes can be added.

Disadvantages of Cluster Computing :

- **High cost :**
  - It is not so much cost-effective due to its high hardware and its design.
- **Problem in finding fault :**
  - It is difficult to find which component has a fault.
- **More space is needed :**
  - Infrastructure may increase as more servers are needed to manage and monitor.

Applications of Cluster Computing :

- Various complex computational problems can be solved.
- It can be used in the applications of aerodynamics, astrophysics and in data mining.
- Weather forecasting.
- Image Rendering.
- Various e-commerce applications.
- Earthquake Simulation.
- Petroleum reservoir simulation.

That is all

Thank you