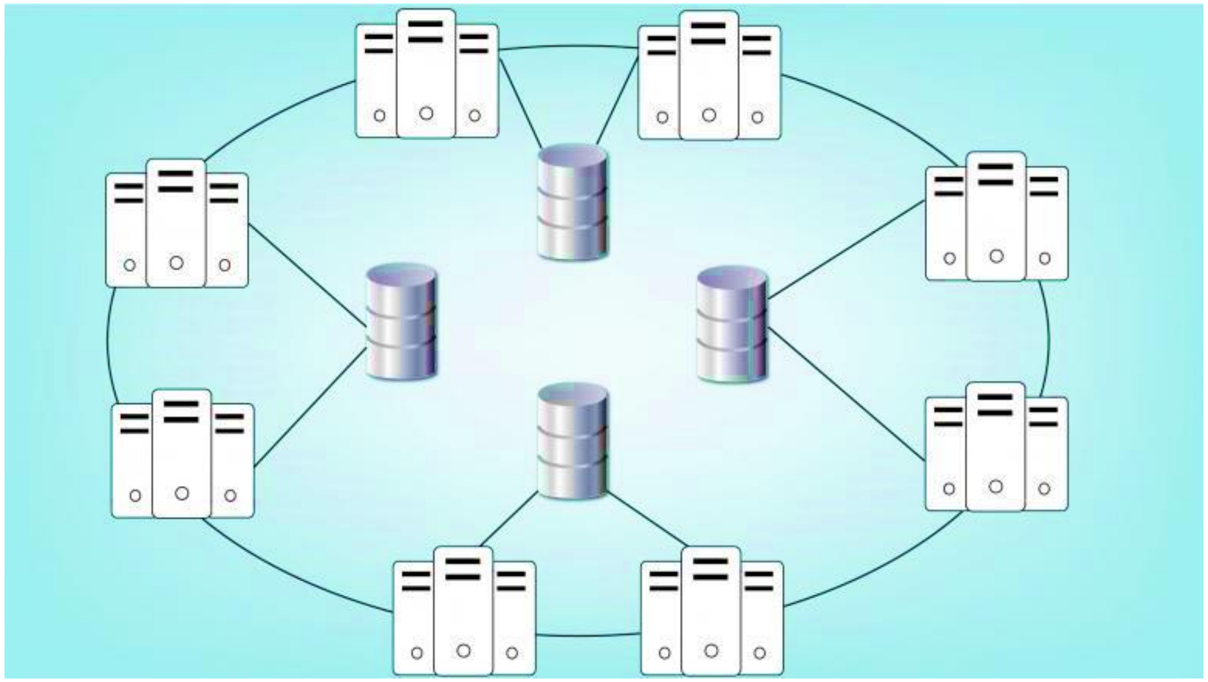
Distributed System

Moumita Asad
IIT, DU



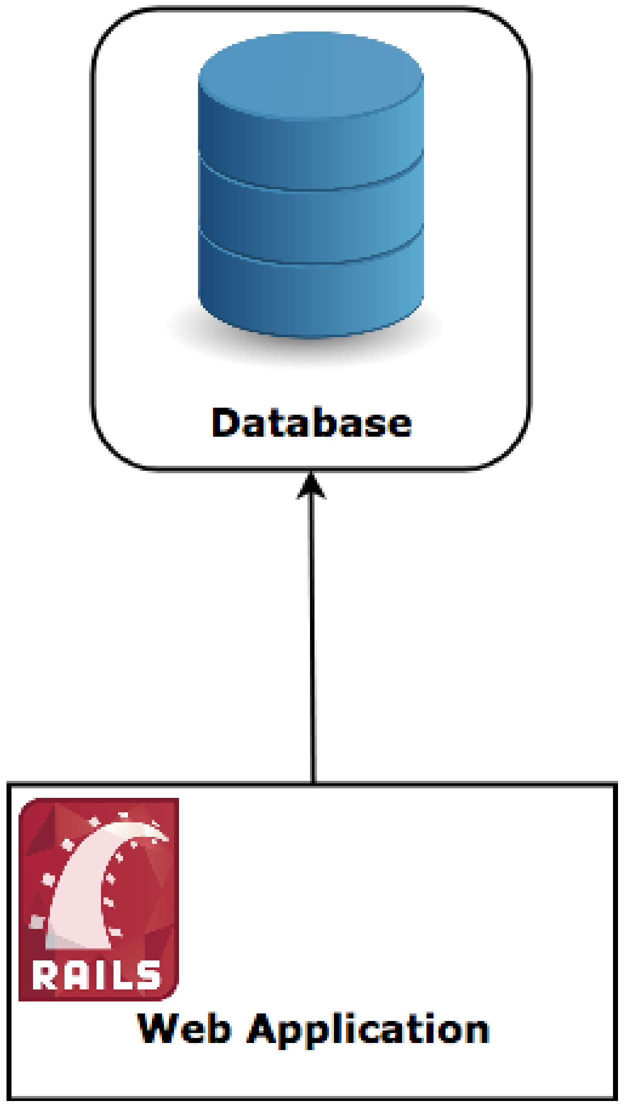
Distributed System

A collection of independent computers
that appears to its users
as a single coherent system

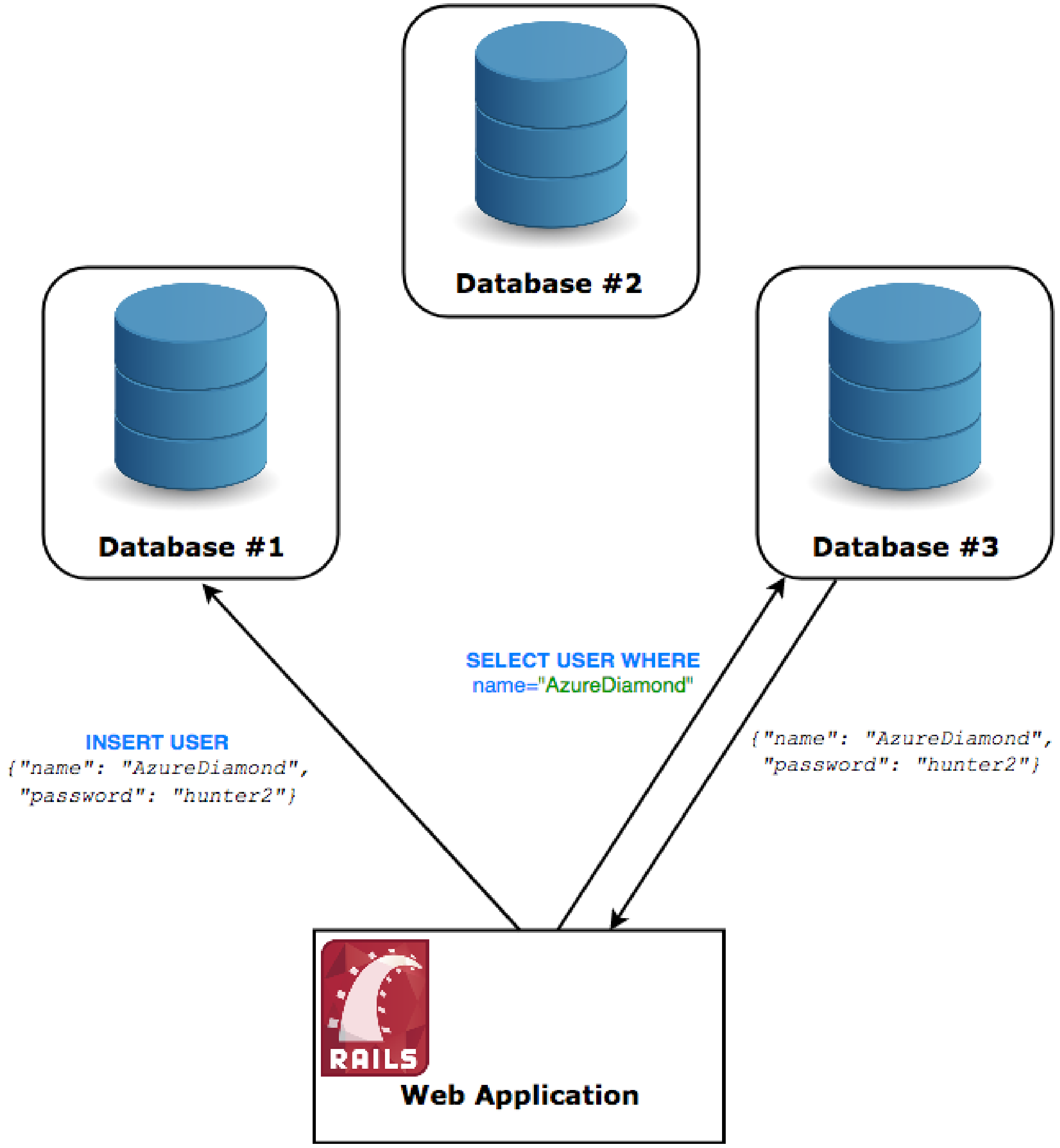
Distributed System

- Distributed system applications are made up of
  - multiple different applications running on different machines, or
  - many replicas running across different machines, all communicating together to implement a system

Traditional System



Distributed System


Database #2
Database #1
Database #3
SELECT USER WHERE name="AzureDiamond"
INSERT USER {"name": "AzureDiamond", "password": "hunter2"}
{"name": "AzureDiamond", "password": "hunter2"}
RAILS
Web Application

Advantages of Distributed System

- **Scalability:** Distributed systems are made on default to be scalable. Whenever there is an increase in workload, users can add more workstations. There is no need to upgrade a single system. Moreover, no any restrictions are placed on the number of machines.
- **Reliability:** Distributed systems are far more reliable than single systems in terms of failures. Even in the case of a single node malfunctioning, it does not pose problems to the remaining servers. Other nodes can continue to function fine.
- **Low Latency:** Since users can have a node in multiple geographical locations, distributed systems allow the traffic to hit a node that's closest, resulting in low latency and better performance.
- **Efficiency:** Distributed systems allow breaking complex problems/data into smaller pieces and have multiple computers work on them in parallel, which can help cut down on the time needed to solve/compute those problems.

Challenges of Distributed System

- **Heterogeneity** (variety and difference): the differences that arise in networks, programming languages, hardware, operating systems and differences in software implementation.
- **Openness**: the system's extensibility and ability to be reimplemented. It can be measured by 3 characteristics:
  1. **interoperability:** system's ability to effectively interchange information between computers by standardization
  2. **portability:** system's ability to properly function on different operating systems
  3. **extensibility:** allowing developers to freely add new features or easily reimplement existing ones without impairing functionality

Challenges of Distributed System

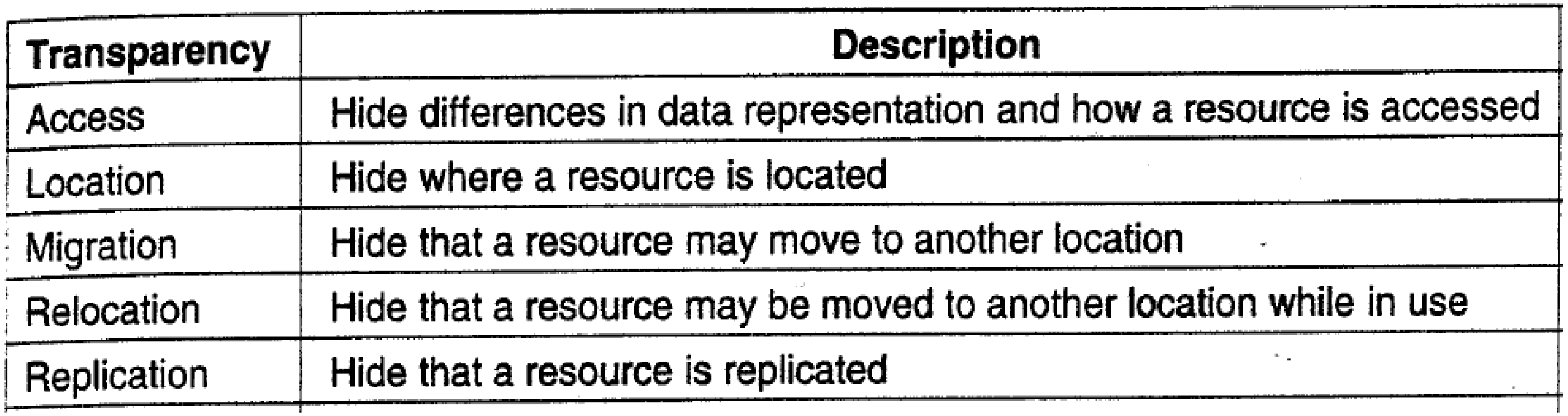
- **Security:** consists of 3 components:
  1. **confidentiality:** protection against disclosure to unauthorized individuals
  2. **integrity:** protection against alteration or corruption
  3. **availability:** protection against interference with the means to access the resources.
- **Concurrency:** multiple clients can access a shared resource at the same time. Thus, steps need to be taken to ensure that any manipulation in the system remains in a stable state. However, the illusion of simultaneous execution should be preserved.

Challenges of Distributed System

- **Scalability:** the measure of a system's ability to increase or decrease in performance and cost in response to changes in application and system processing demands.
  - Example: how well a hardware system performs when the number of users is increased, how well a database withstands growing numbers of queries, or how well an operating system performs on different classes of hardware.
- **Failure handling:** When faults occur, programs may produce incorrect results or may stop before they have completed the intended computation. Any process, computer or network may fail independently of the others. Therefore each component needs to be aware of the possible ways in which the components it depends on may fail and be designed to deal with each of those failures appropriately.

Challenges of Distributed System

- **Transparency:** system that is able to present itself to users and applications as if it were only a single computer system is said to be transparent.



Transparency	Description
Access	Hide differences in data representation and how a resource is accessed
Location	Hide where a resource is located
Migration	Hide that a resource may move to another location
Relocation	Hide that a resource may be moved to another location while in use
Replication	Hide that a resource is replicated

CAP Theorem

- **Consistency:** all clients see the same data at the same time, no matter which node they connect to. For this to happen, whenever data is written to one node, it must be instantly forwarded or replicated to all the other nodes in the system before the write is deemed 'successful.'
- **Availability:** any client making a request for data gets a response, even if one or more nodes are down. Another way to state this—all working nodes in the distributed system return a valid response for any request, without exception.
- **Partition tolerance:** A partition is a communications break within a distributed system—a lost or temporarily delayed connection between two nodes. Partition tolerance means that the cluster must continue to work despite any number of communication breakdowns between nodes in the system.

CAP Theorem

- The theorem formalizes the tradeoff between consistency and availability when there's a network partition.

CAP Theorem

- **AP (Availability + Partition tolerance)**
  - systems that decide to sacrifice consistency and choose to provide higher availability.
  - data conflicts will arise, data versions need to be maintained if data conflicts occur
- **CP (Consistency + partition tolerance)**
  - systems that decide to sacrifice availability in the name of providing stronger consistency guarantees.
  - the system will ensure that most of the nodes have the same data, and a few nodes will become unavailable when they are not synchronized to the latest version of the data.

Resources

- https://www.freecodecamp.org/news/a-thorough-introduction-to-distributed-systems-3b91562c9b3c/
- https://opencirrus.org/distributed-computing-pros-cons/
- https://iq.opengenus.org/challenges-failures-in-distributed-systems/
- https://www.youtube.com/watch?v=k-Yaq8AHlFA