CS8492: DATABASE MANAGEMENT SYSTEMS

Distributed Database Management Systems

Session Objectives

- Introduction to DDBMS
- Advantages of DDB
- Types of Data store

Session Outcomes

- At the end of this session, participants will be able to
  - Understand the ddbms
  - Advantages of DDB
  - Types of Data store

Distributed Database Management Systems

Dr. P.Mirunalini
SSNCE
May 23rd,2022

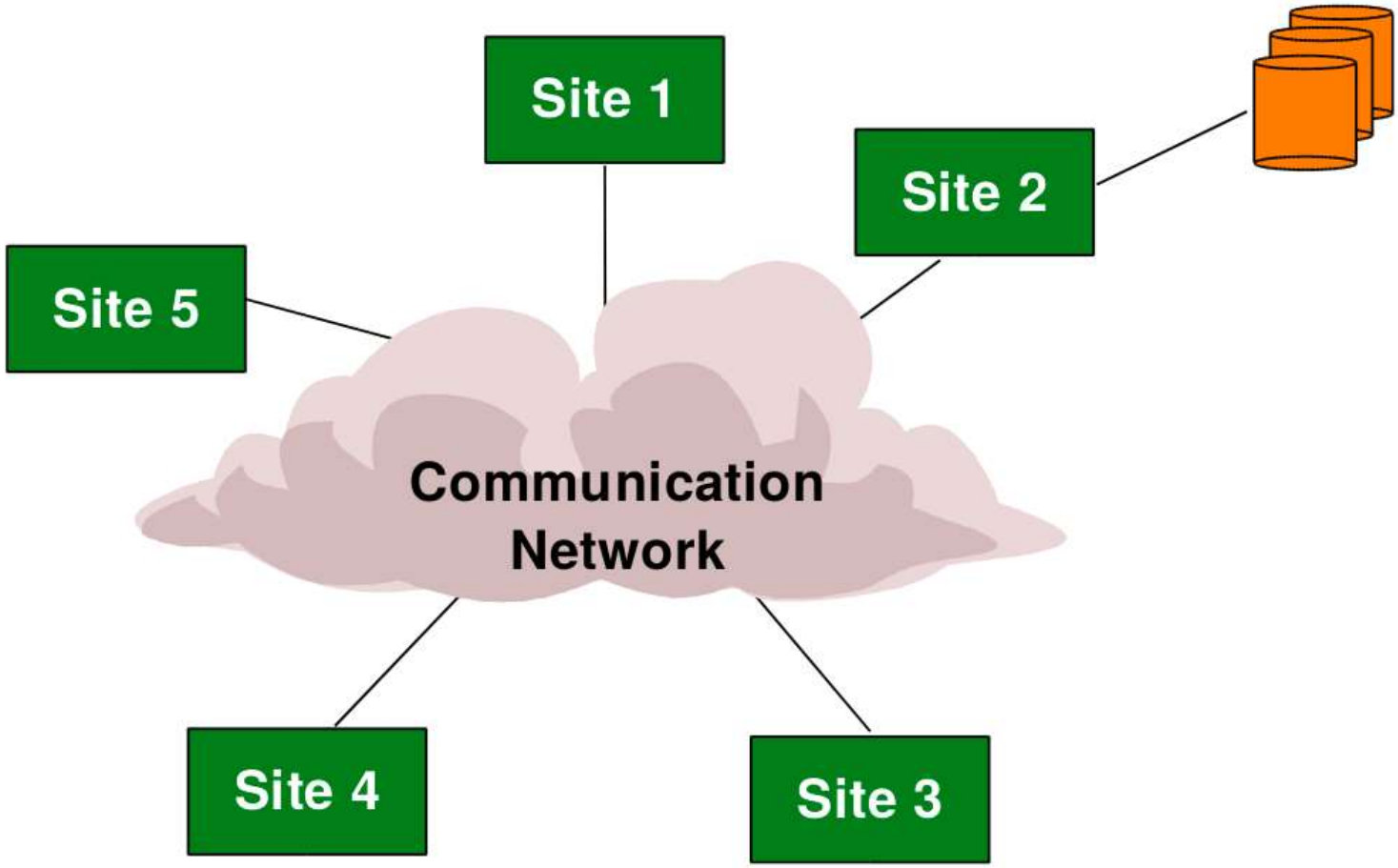
Distributed Database Concepts

- A **distributed database (DDB)** processes **Unit of execution** (a transaction) in a **distributed manner** using multiple networked networks.
- Transactions may access data at one or more sites.
- Database systems that run on each site are independent of each other.
- **DDB**: A distributed database (DDB) is a collection of **multiple logically related database** distributed over a computer network.
- **D-DBMS** - A distributed database management system is a software system that manages a distributed database while making the distribution transparent to the user.

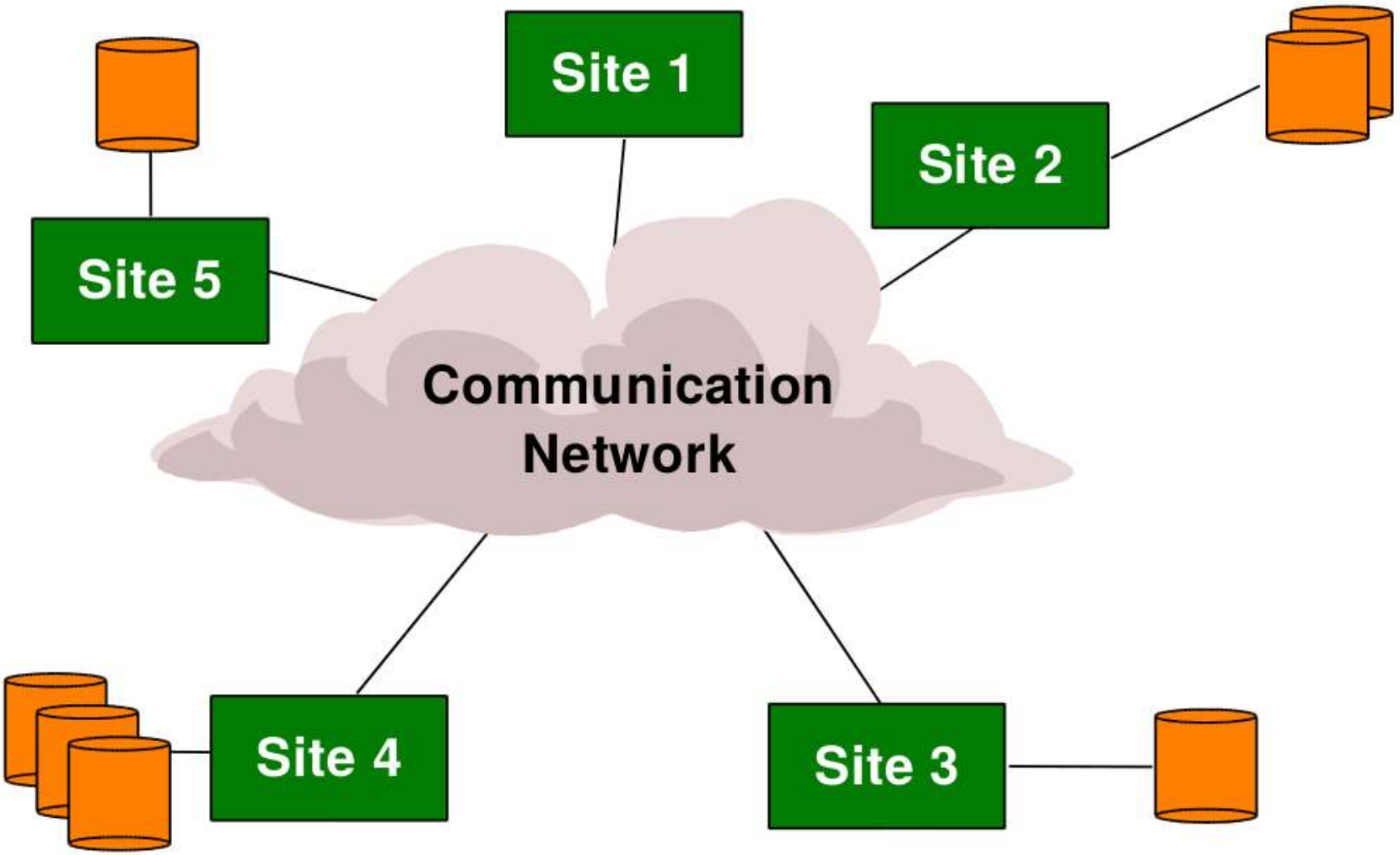
Distributed Database Concepts

- **Connection of database nodes over a computer network:** There are multiple computers, called sites or nodes. These sites must be connected by an underlying network to transmit data and commands among sites.
- **Logical interrelation of the connected databases** :It is essential that the information in the various database nodes be logically related.
- **Possible absence of homogeneity among connected nodes:**
  - It is not necessary that all nodes be identical in terms of data, hardware and software.
  - The sites may all be located in physical proximity - connected via a local area network or geographically distributed over large distances and connected via a long-haul or wide area network.
  - Connected using telephone lines, cables, wireless communication infrastructures, or satellites.

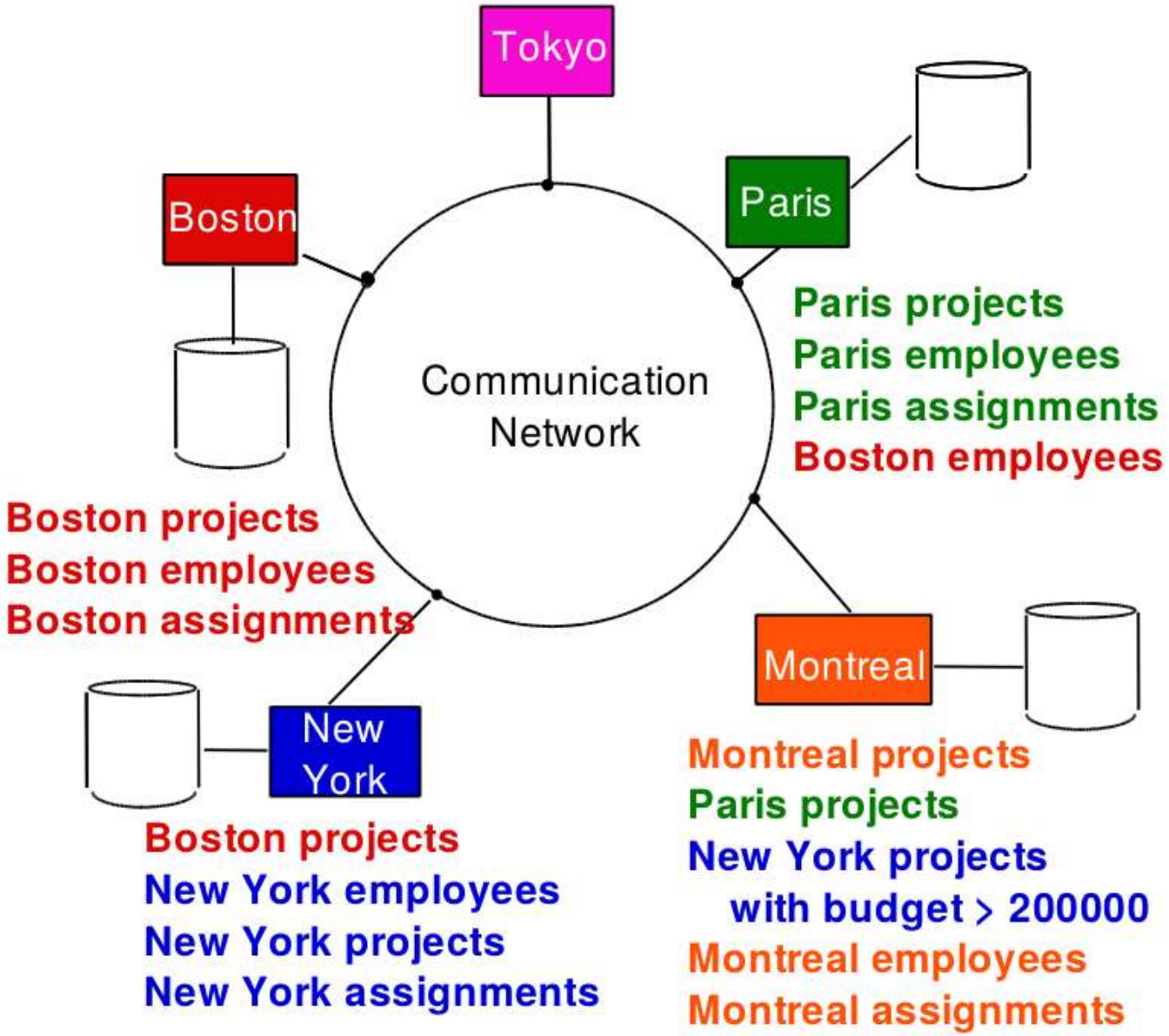
A Centralized DBMS



A Distributed DBMS



Example-DDB


Tokyo
Boston
Paris
Communication Network
Paris projects
Paris employees
Paris assignments
Boston employees
Boston projects
Boston employees
Boston assignments
Montreal
New York
Montreal projects
Paris projects
New York projects
with budget > 200000
Montreal employees
Montreal assignments
Boston projects
New York employees
New York projects
New York assignments

Implicit Assumptions

Data stored at a number of sites ⇨ each site logically consists of a single processor.

Processors at different sites are interconnected by a computer network

- no multiprocessors
- parallel database systems

Distributed database is a database, not a collection of files

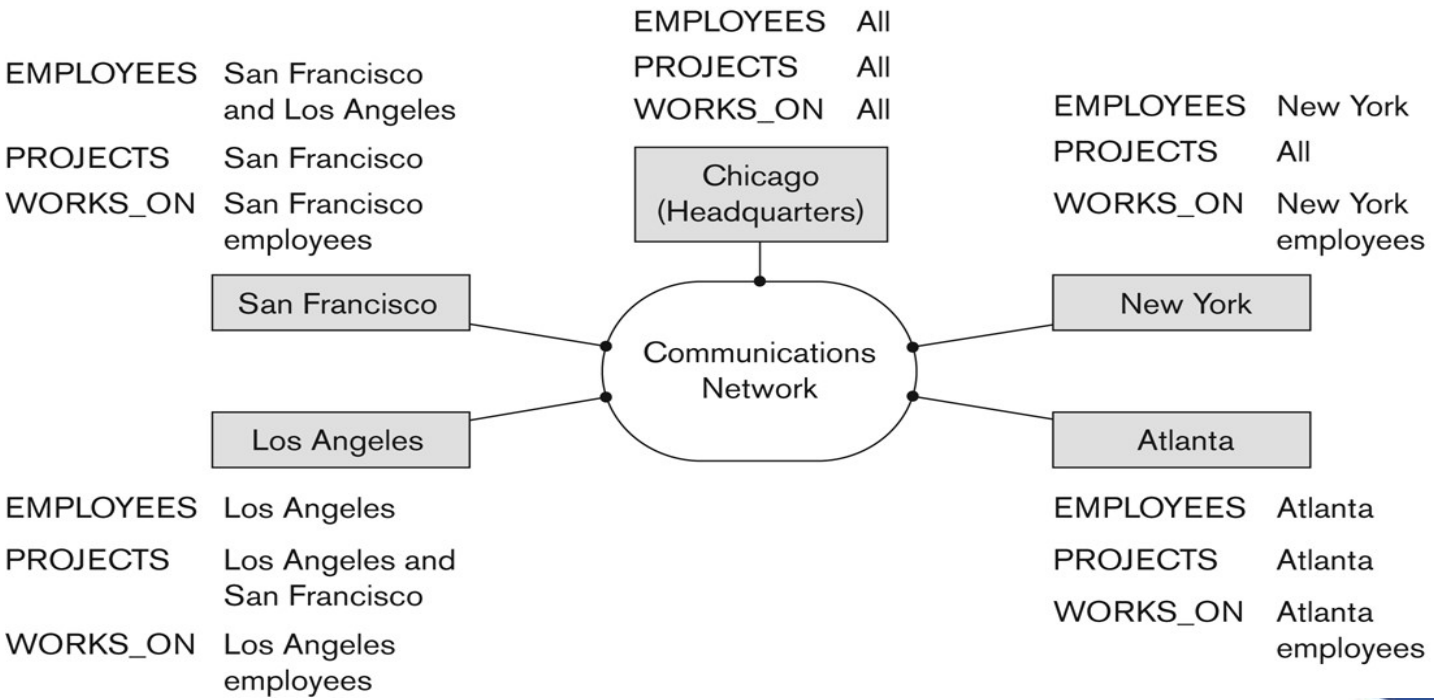
- All data are logically related

Distributed Database System

Advantages:

- Management of distributed data with different **levels of transparency .**
  - This refers to the physical placement of data (files, relations, etc.) which is not known to the user (**distribution transparency**).

Figure 25.2
Data distribution and replication among distributed databases.



Distributed Database System

Data organization transparency: This refers to freedom for the user from the **operational details of the network and the placement of the data** in the distributed system.

Location transparency : Refers to the fact that the command used to perform a task is **independent of the location of the data and the location of the node** Where the command was issued.

Naming transparency : Implies that once a name is associated with an object, the named objects can be **accessed unambiguously** without additional specification as to where the data is located.

Replication transparency : Copies of the same data objects may be stored at multiple sites for better **availability, performance, and reliability.** Replication transparency makes the user unaware of the existence of these copies.

Distributed Database System

Fragmentation transparency: Two types of fragmentation are possible.

Horizontal fragmentation distributes a relation (table) into subrelations that are subsets of the tuples (rows) in the original relation; this is also known as **sharding.**

Vertical fragmentation : distributes a relation into subrelations where each subrelation is defined by a subset of the columns of the original relation.

Distributed Database System

Increased reliability and availability:

- **Reliability:** **R**eliability is the probability a system will fail in a given period. In simple terms, a distributed system is considered reliable if it keeps delivering its services even when one or several of its software or hardware components fail.
- EX: In the large electronic commerce store (like Amazon), any user transaction should never be canceled due to a failure of the machine running that transaction. A reliable distributed system achieves this through the redundancy of both the software components and data.
- **Availability** is the probability that the **system is continuously available** (usable or accessible) during a time interval.
- Ex: An aircraft that can be flown for many hours a month without much downtime has high availability. A distributed database system has **multiple nodes** (computers) and if one fails then others are available to do the job.

Distributed Database System

Scalability and Partition Tolerance

- **Scalability:** determines the extent to which the system can expand its capacity while continuing to operate without interruption.
  - **Horizontal scalability:** This refers to expanding the number of nodes in the distributed system. As nodes are added to the system, it should be possible to distribute some of the data and processing loads from existing nodes to the new nodes.
  - **Vertical scalability:** This refers to expanding the capacity of the individual nodes in the system, such as expanding the storage capacity or the processing power of a node

- **Partition Tolerance:**
  - Faults cause the nodes to be partitioned into groups of nodes. The nodes within each partition are still connected by a subnetwork, but communication among the partitions is lost.
  - The concept of partition tolerance states that the system should have the capacity to continue operating while the network is partitioned.

Distributed Database System

Improved performance:

- A distributed DBMS fragments the database to **keep data closer to where it is needed most.**
- This reduces data management (**access and modification**) time significantly.
- Easier expansion (**scalability**):
- Allows new nodes (computers) to be added anytime without chaining the entire configuration.

Distributed Data Storage

A relation r can be stored in distributed database by two approaches:

- **Replication:**
  - System maintains multiple copies of data, stored in different sites, for faster retrieval and fault tolerance.
- **Fragmentation:**
  - Relation is partitioned into several fragments and stored in distinct sites
- **Replication and fragmentation can be combined:**
  - Relation is partitioned into several fragments: system maintains several identical replicas of each such fragment.

Data Replication

- **Data Replication**
  - Database is replicated to all sites.
  - In full replication the entire database is replicated and in partial replication some selected part is replicated to some of the sites.
  - Data replication is achieved through a **replication schema.**
- **Data Allocation**
  - Each fragment—or each copy of a fragment—must be assigned to a particular site in the distributed system. This process is called **data distribution.**
  - The choice of sites and the degree of replication depend on the **performance and availability goals** of the system and on the **types and frequencies of transactions** submitted at each site.

Data Replication

Advantages of Replication:

- **Availability:** Failure of site containing relation r does not result in unavailability of r is replicas exist.
- **Parallelism:** Queries on r may be processed by several nodes in Parallel.
- **Reduced data transfer: R**elation r is available locally at each site containing a replica of r.

Disadvantages of Replication:

- **Increased cost of updates**: Each replica of relation r must be updated.
- **Increased complexity of concurrency control**: Concurrent updates to distinct replicas may lead to inconsistent data unless special concurrency control mechanisms are implemented.

Data Fragmentation, Replication and Allocation

Data Fragmentation:

- Split a relation into logically related and correct parts. A relation can be fragmented in two ways:
  - **Horizontal Fragmentation**
  - **Vertical Fragmentation**

Reasons for Fragmentation

- Entire relation is not a suitable unit for distribution because, each application only **views a subset of relations**.
- If the applications refers the relations which resides at different sites then two methods can be followed:
  - Either the relation is stored at **only one site**(problem of remote data access) or it is **replicated** (problem in updates).
- Hence decomposition of relations into **fragments results in the parallel execution** of a single query by dividing it into a set of subqueries that operate on fragments.
- Increases the level of concurrency (**intraquery concurrerncy**) and hence **system throughput.**

Horizontal fragmentation

- It is a horizontal subset of a relation which contain those of tuples which satisfy selection conditions.
- Consider the Employee relation with selection condition (DNO = 5). All tuples satisfy this condition will create a subset which will be a horizontal fragment of Employee relation.
- A selection condition may be composed of several conditions connected by AND or OR.
- **Derived horizontal fragmentation:** It is the partitioning of a primary relation to other secondary relations which are related with Foreign keys.

Horizontal fragmentation

- **Representation**
  - **Horizontal fragmentation**
    - Each horizontal fragment on a relation can be specified by a sigma (σ_{Ci} (R)) operation in the relational algebra.
    - Complete horizontal fragmentation
    - A set of horizontal fragments whose conditions C1, C2, …, Cn include all the tuples in R- that is, every tuple in R satisfies (C1 OR C2 OR … OR Cn).

Horizontal fragmentation

PROJ

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York
P3	CAD/CAM	250000	New York
P4	Maintenance	310000	Paris
P5	CAD/CAM	500000	Boston

$PROJ_1 = \sigma_{budget < 200000}(PROJ)$

$PROJ_1$

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York

$PROJ_2 = \sigma_{budget \geqslant 200000}(PROJ)$

$PROJ_2$

PNO	PNAME	BUDGET	LOC
P3	CAD/CAM	250000	New York
P4	Maintenance	310000	Paris
P5	CAD/CAM	500000	Boston

Vertical fragmentation

- It is a subset of a relation which is created by a subset of columns.
- Thus a vertical fragment of a relation will contain values of **selected columns.**
- There is **no selection condition** used in vertical fragmentation.
- Consider the Employee relation. A vertical fragment can be created by keeping the values of Name, Bdate, Sex, and Address.
- Because there is no condition for creating a vertical fragment, each fragment must include the **primary key attribute** of the parent relation Employee.
- In this way all vertical fragments of a relation are connected.

Vertical fragmentation - Representation

Vertical fragmentation

- A vertical fragment on a relation can be specified by a $\Pi_{Li}(R)$ operation in the relational algebra.
- Complete vertical fragmentation
- A set of vertical fragments whose projection lists L1, L2, …, Ln include all the attributes in R but share only the primary key of R.
- In this case the projection lists satisfy the following two conditions:
  - $L1 \cup L2 \cup ... \cup Ln = ATTRS(R)$
  - $Li \cap Lj = PK(R)$ for any i , j, where ATTRS (R) is the set of attributes of R and PK(R) is the primary key of R.

Reconstruct the relation r by taking the natural join of all fragments:

$$r = r_1 \bowtie r_2 \bowtie .... \bowtie r_n$$

Vertical fragmentation - Eample

PROJ

PNO	PNAME	BUDGET	LOC
P1	Instrumentation	150000	Montreal
P2	Database Develop.	135000	New York
P3	CAD/CAM	250000	New York
P4	Maintenance	310000	Paris
P5	CAD/CAM	500000	Boston

$PROJ_1 = \Pi_{PNO,BUDGET}(PROJ)$

$PROJ_2 = \Pi_{PNO,PNAME,LOC}(PROJ)$

$PROJ_1$

PNO	BUDGET
P1	150000
P2	135000
P3	250000
P4	310000
P5	500000

$PROJ_2$

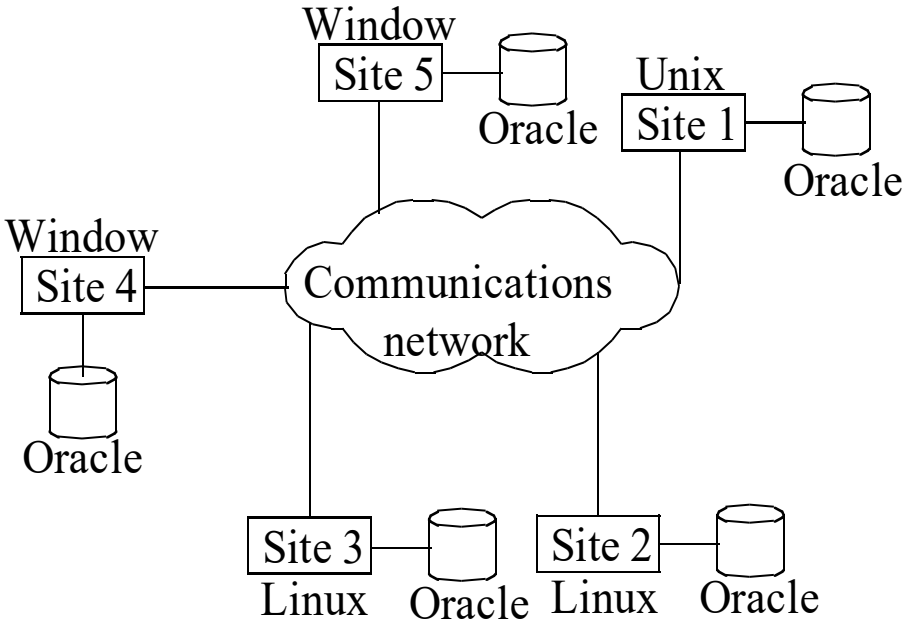
PNO	PNAME	LOC
P1	Instrumentation	Montreal
P2	Database Develop.	New York
P3	CAD/CAM	New York
P4	Maintenance	Paris
P5	CAD/CAM	Boston

Correctness Rule for Fragmentation

- **Reconstruction :**
  - » If a relation R is decomposed into fragments R1, R2,..., Rn, it should be possible to define a relational operator Δ such that **R** = Δ **Ri**
- **Disjointness :**
  - » Horizontal Fragmentation – If a relation R is *horizontally* decomposed into fragments R1, R2,..., Rn, and data item di is in Rj, it is not in any other fragment Rk.
  - » If a relation R is *vertically* decomposed, its **primary key** attributes are typically repeated in all its fragments.

Types of Distributed Database Systems

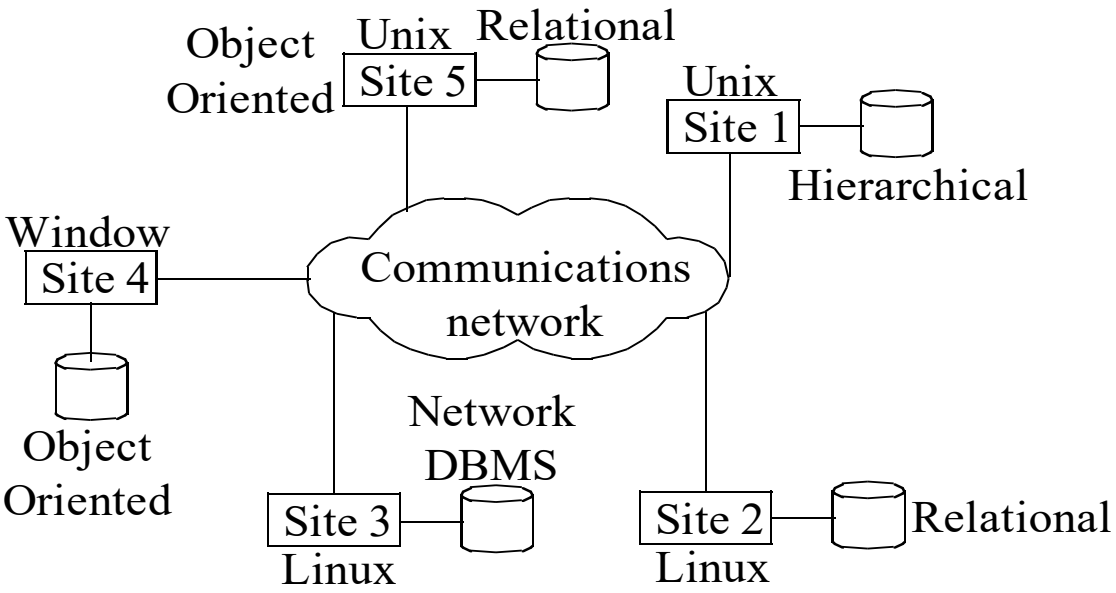
- Homogeneous
  - All sites of the database system have identical setup, i.e., same database system software.
  - The underlying operating system may be different.
    - For example, all sites run Oracle or DB2, or Sybase or some other database system.
  - The underlying operating systems can be a mixture of Linux, Window, Unix, etc.



Types of Distributed Database Systems

Heterogeneous- Federated:

- Each site may run **different database** system but the data access is managed through a **single conceptual schema**.
- Each site must adhere to a centralized access policy. There may be a global schema.
- **Multidatabase:** There is no one conceptual global schema. For data access a schema is constructed dynamically as needed by the application software.



Summary

- DDBMS
- Advantages of DDB
- Types of data storage
  - Replication
  - Fragmentation