CSCI 5408

Dr. Saurabh Dey

saurabh.dey@dal.ca

Content

1. Distributed DBMS

The Evolution of Distributed Database Management Systems

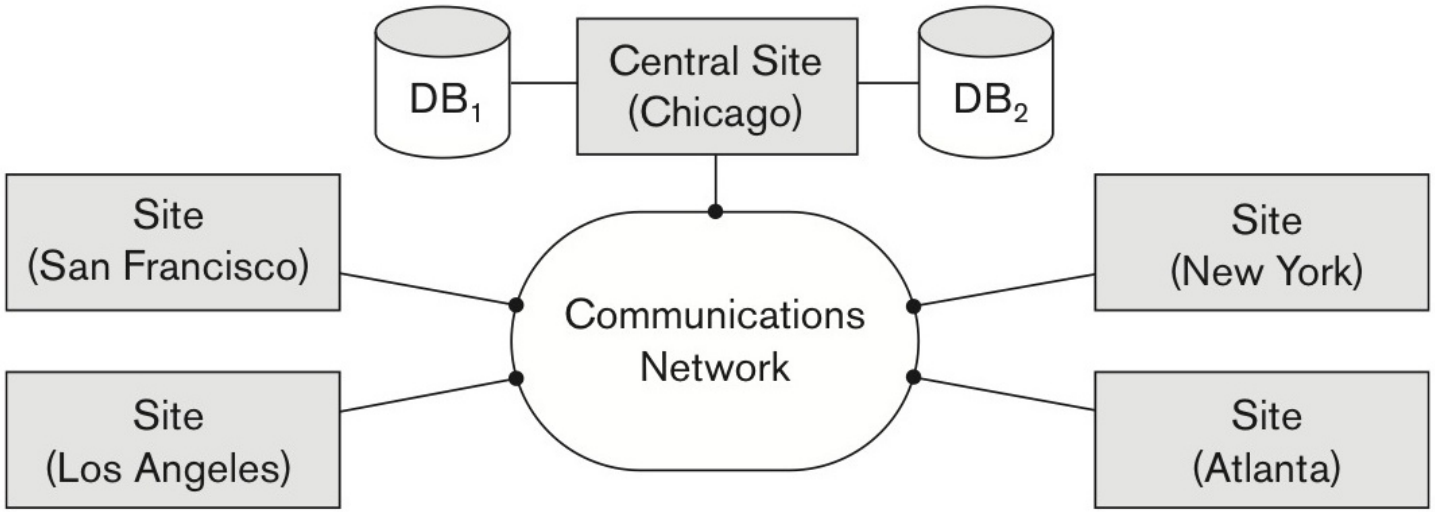
Centralized database management system

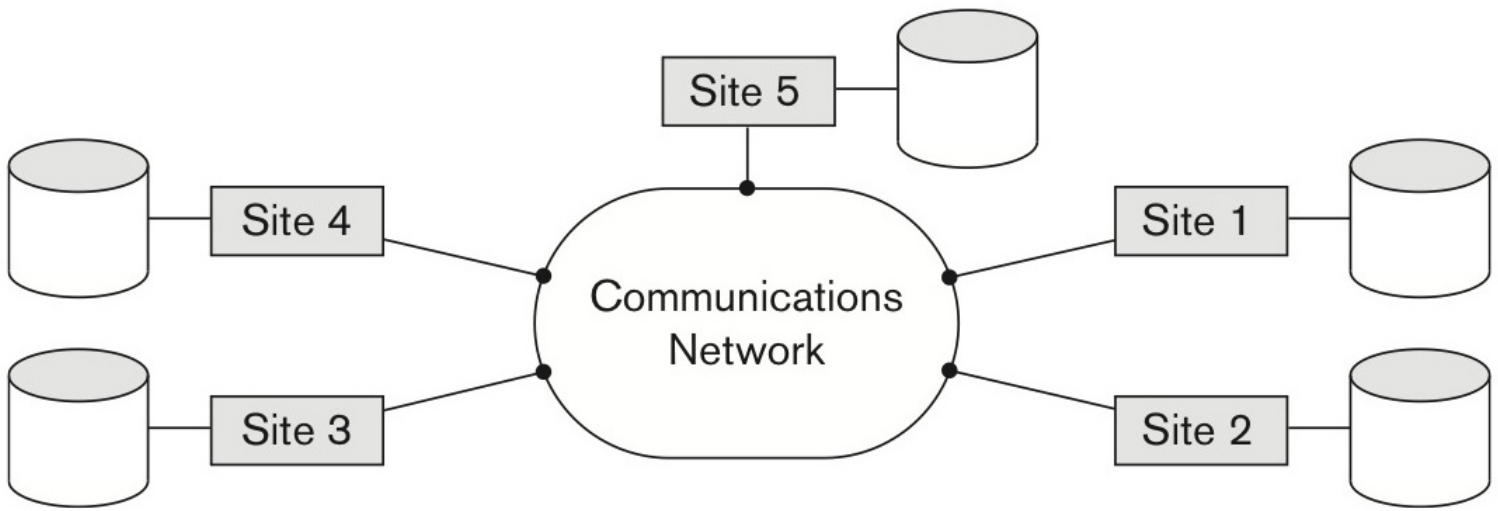
- Required corporate data be stored in a single central site
- Provided data access through dumb terminals
- Not suitable, when quickly moving events required faster response times and equally quick access to information

Distributed Database Management Systems

A distributed database management system (DDBMS)

- Governs storage and processing of logically related data over interconnected computer systems
- Distributes data and processing functions among several sites


DB1
Central Site
(Chicago)
DB2
Site
(San Francisco)
Communications
Network
Site
(New York)
Site
(Los Angeles)
Site
(Atlanta)


Site 5
Site 4
Site 1
Communications
Network
Site 3
Site 2

The Evolution of Distributed Database Management Systems

Database requirements in a dynamic business environment

Rapid ad hoc data access

Crucial in the quick-response decision-making environment

Distributed data access

Needed to support geographically dispersed business units

DDBMS Advantages and Disadvantages

ADVANTAGES	DISADVANTAGES
Data is located near the site of greatest demand. The data in a distributed database system is dispersed to match business requirements.	***Complexity of management and control.*** Applications must recognize data location, and they must be able to stitch together data from various sites. Database administrators must have the ability to coordinate database activities to prevent database degradation due to data anomalies.
Faster data access. End users often work with only the nearest stored subset of the data.	***Technological difficulty.*** Data integrity, transaction management, concurrency control, security, backup, recovery, and query optimization must all be addressed and resolved.
Faster data processing. A distributed database system spreads out the system's workload by processing data at several sites.	***Security.*** The probability of security lapses increases when data is located at multiple sites. The responsibility of data management will be shared by different people at several sites.

DDBMS Advantages and Disadvantages (Contd.)

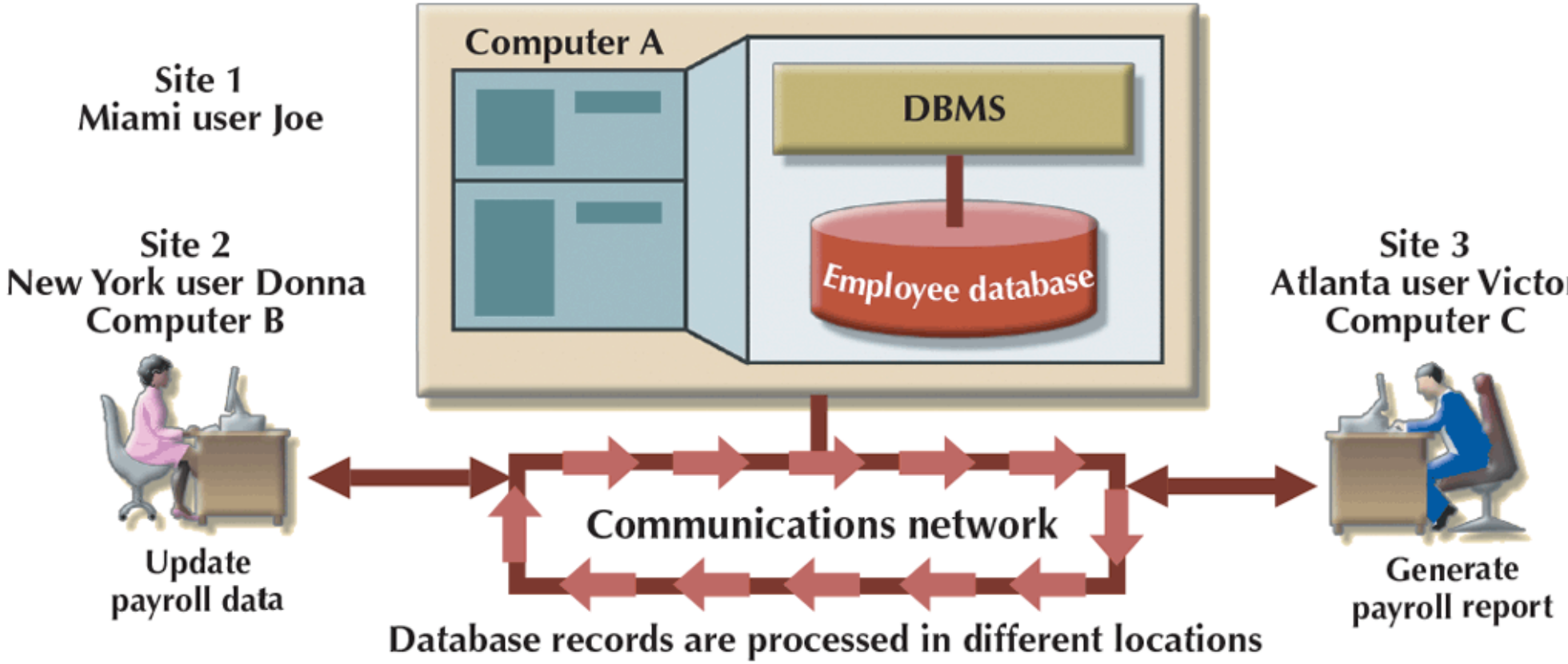
ADVANTAGES	DISADVANTAGES
Growth facilitation. New sites can be added to the network without affecting the operations of other sites.	***Lack of standards.*** There are no standard communication protocols at the database level. For example, different database vendors employ different and often incompatible techniques to manage the distribution of data and processing in a DDBMS environment.
Improved communications. Because local sites are smaller and located closer to customers, local sites foster better communication among departments and between customers and company staff.	***Increased storage and infrastructure requirements.*** Multiple copies of data are required at different sites, thus requiring additional storage space.
Reduced operating costs. It is more cost-effective to add nodes to a network than to update a mainframe system. Development work is done more cheaply and quickly on low-cost PCs and laptops than on mainframes.	***Increased training cost.*** Training costs are generally higher in a distributed model than they would be in a centralized model, sometimes even to the extent of offsetting operational and hardware savings.

DDBMS Advantages and Disadvantages (Contd.)

ADVANTAGES	DISADVANTAGES
User-friendly interface. Client devices are usually equipped with an easy-to-use graphical user interface (GUI). The GUI simplifies training and use for end users.	***Higher costs.*** Distributed databases require duplicated infrastructure to operate, such as physical location, environment, personnel, software, and licensing.
Less danger of a single-point failure. When one of the computers fails, the workload is picked up by other workstations. Data is also distributed at multiple sites.	
Processor independence. The end user can access any available copy of the data, and an end user's request is processed by any processor at the data location.	

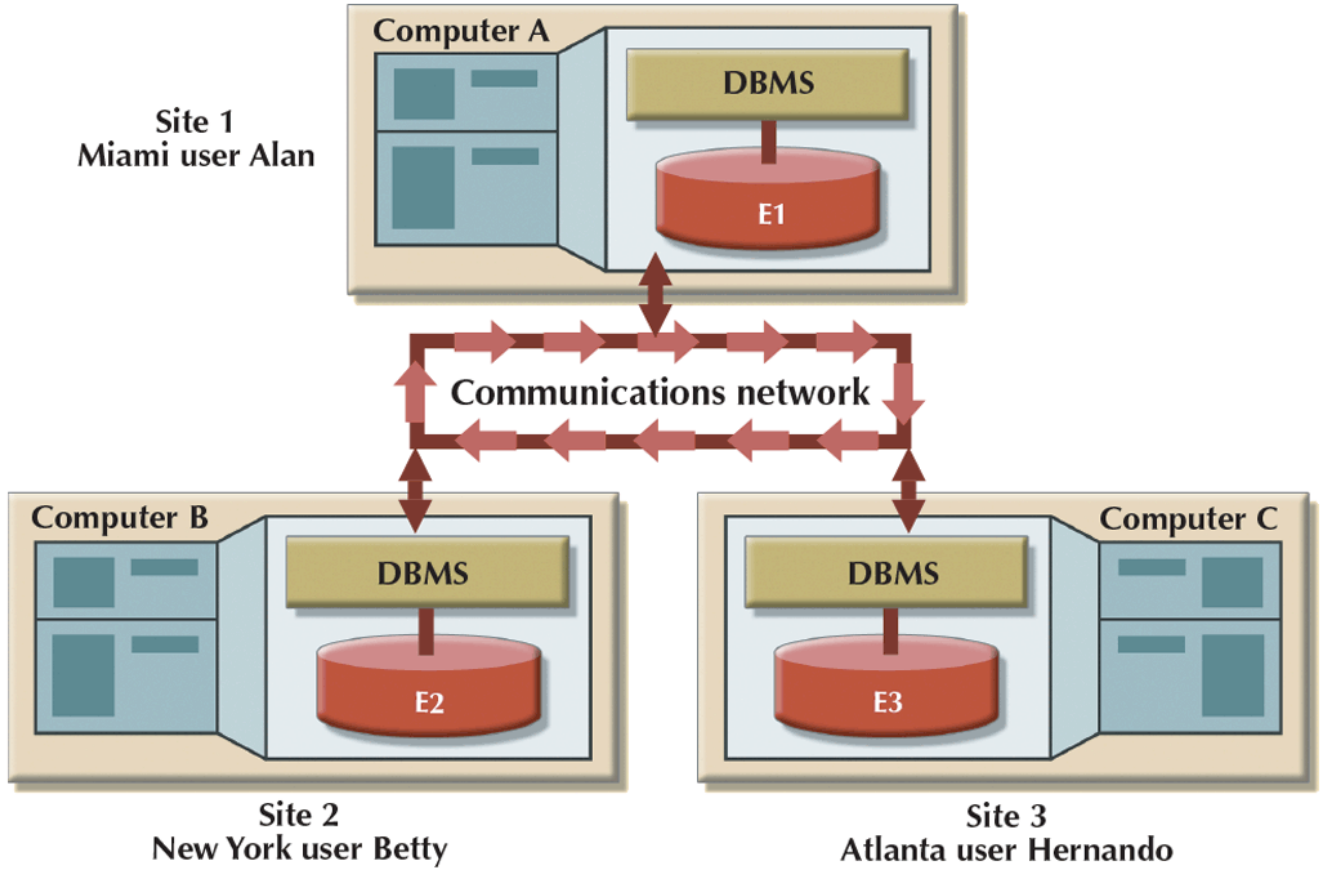
Distributed Processing and Distributed Databases

Distributed processing: database's logical processing is shared among two or more physically independent sites via network



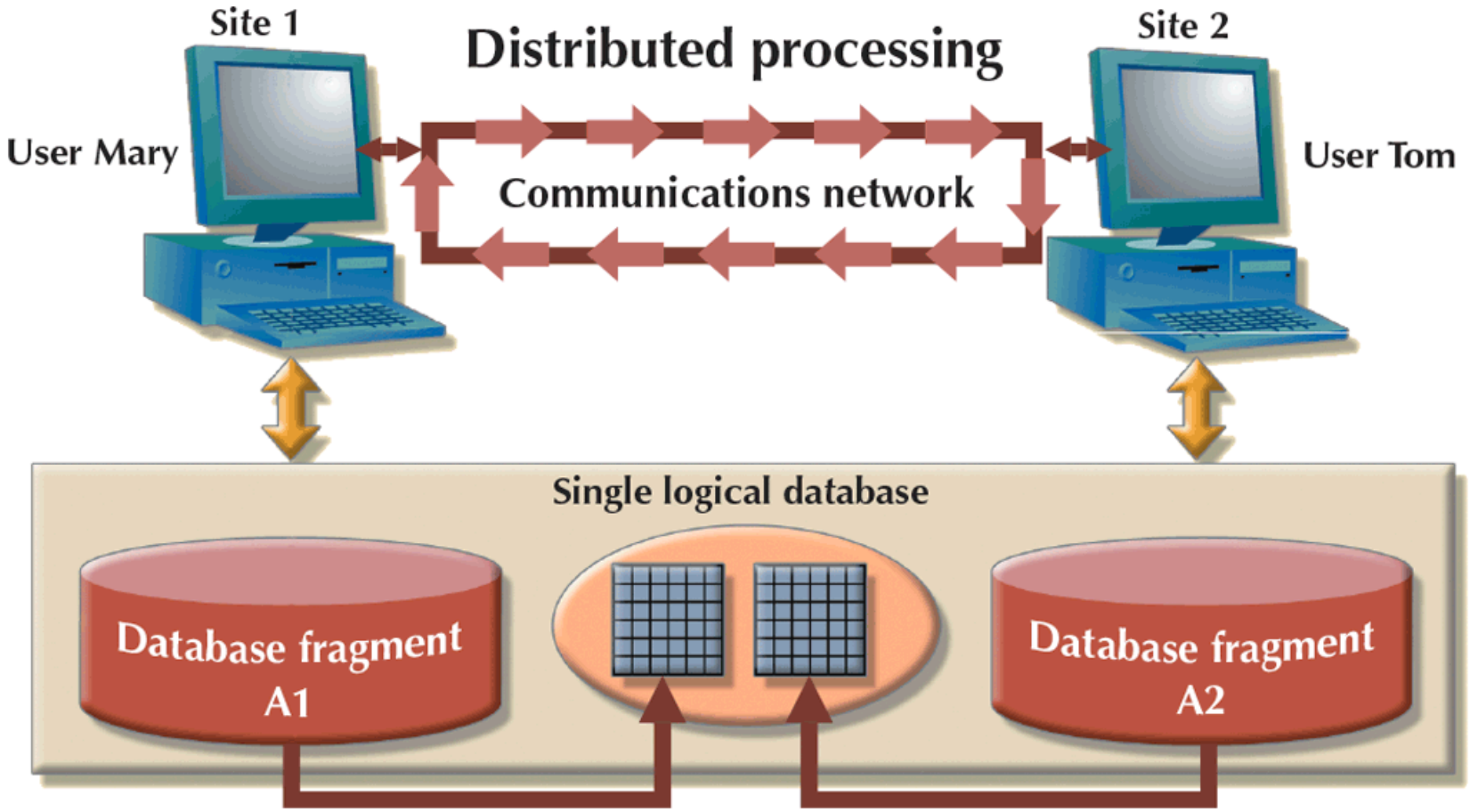
Distributed Processing and Distributed Databases

Distributed database: stores logically related database over two or more physically independent sites via a computer network



Distributed Processing and Distributed Databases

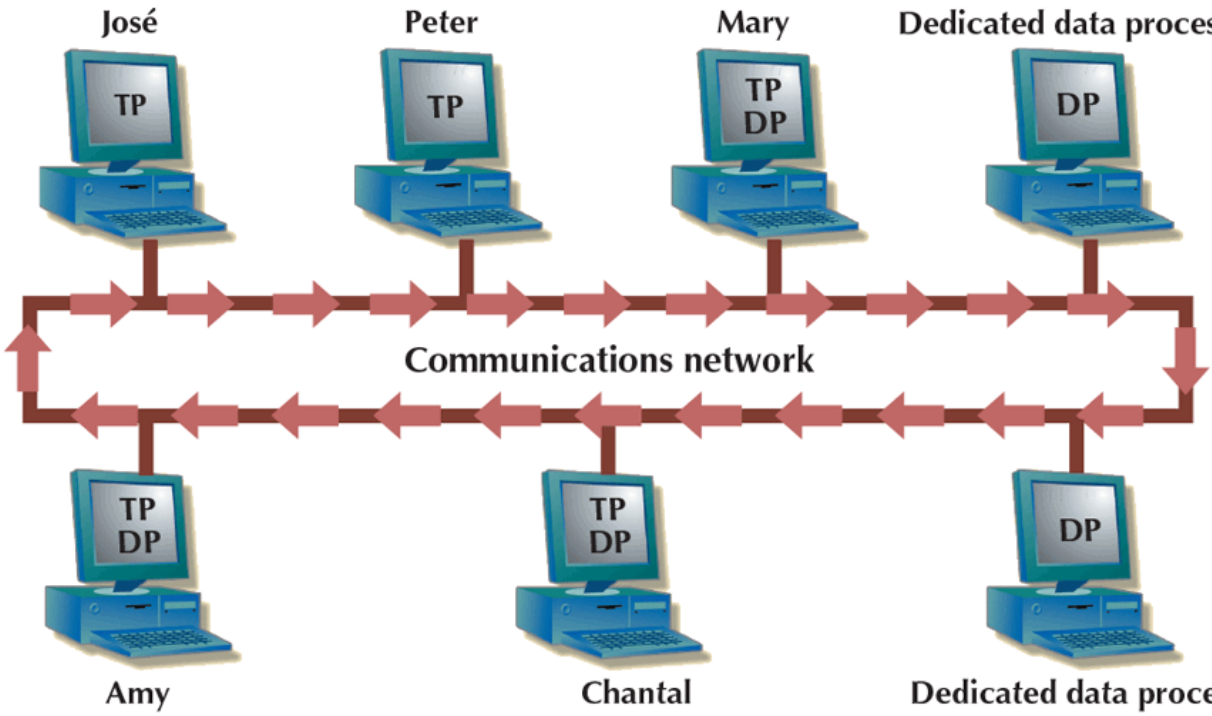
Database fragments: database composed of many parts in distributed database system



DDBMS Components

The DDBMS must include at least the following components

- Computer workstations or remote devices
- Network hardware and software components
- Communications media
- Transaction processor (TP)
- Data processor (DP) or data manager (DM)



Note: Each TP can access data on any DP, and each DP handles all requests for local data from any TP.

Levels of Data and Process Distribution

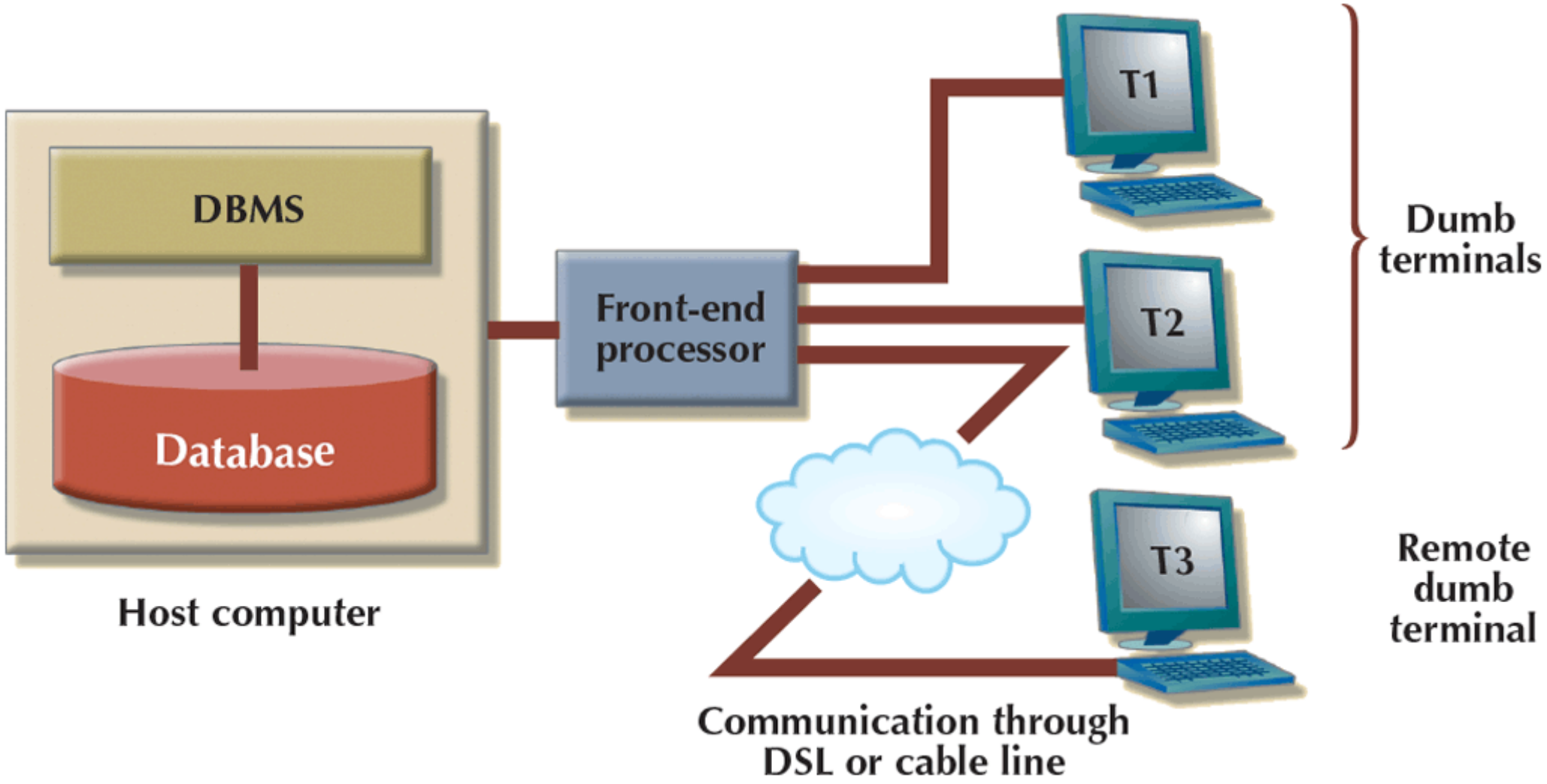
DATABASE SYSTEMS: LEVELS OF DATA AND PROCESS DISTRIBUTION		
ADVANTAGES	SINGLE-SITE DATA	MULTIPLE-SITE DATA
Single-site process	Host DBMS	Not applicable (Requires multiple processes)
Multiple-site process	File server Client/server DBMS (LAN DBMS)	Fully distributed Client/server DDBMS

Single-site Processing, Single-site Data

Characteristics

- Processing is done on a single host computer
- Data stored on host computer's local disk
- Processing cannot be done on end user's side
- DBMS is accessed by connected terminals

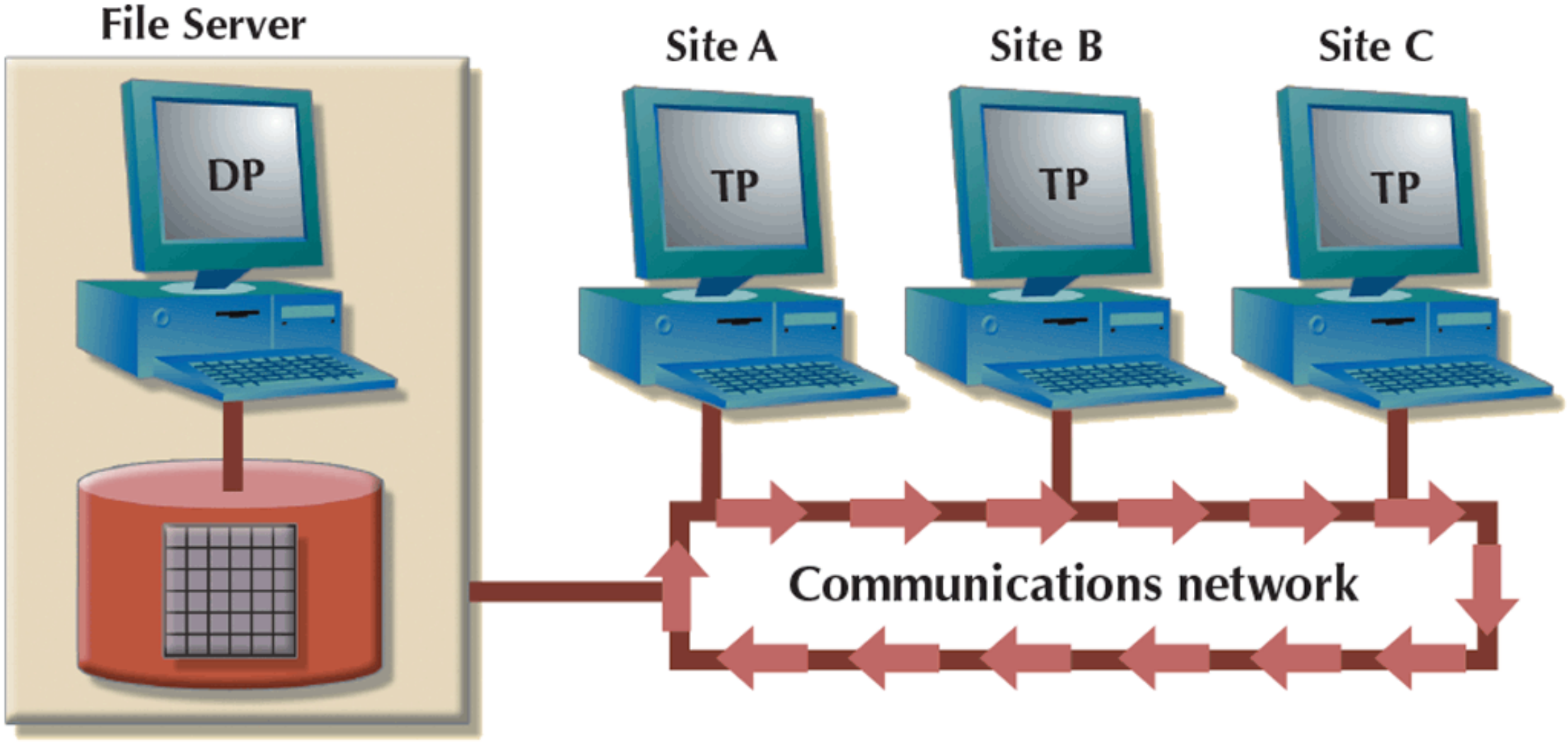
Single-site Processing



Multiple-site Processing, Single-site Data

- Multiple processes run on different computers sharing a single data repository
- Typically requires network file server running conventional applications
- Accessed through LAN
- Client/server architecture
  - Reduces network traffic
  - Distributes processing
  - Supports data at multiple sites

Multiple-site Processing, Single-site Data



Multiple-site Processing, Multiple-site Data

Fully distributed database management system

Support multiple data processors and transaction processors at multiple sites

Classifications

Homogeneous: integrate multiple instances of same DBMS over a network

Heterogeneous: integrate different types of DBMSs over a network

Fully heterogeneous: support different DBMSs, each supporting different data model running under different computer systems

Multiple-site Processing, Multiple-site Data

Restrictions of DDBMS

- Remote access is provided on a read-only basis
- Restrictions on the number of remote tables that may be accessed in a single transaction
- Restrictions on the number of distinct databases that may be accessed
- Restrictions on the database model that may be accessed

Distributed Database Transparency Features

Minimum desirable DDBMS transparency features

1. **Distribution transparency**
   - Treated as a single logical database

2. **Transaction transparency**
   - allows a transaction to update data at more than one network site.

3. **Failure transparency**
   - ensures that the system will continue to operate in the event of a node or network failure.

Distributed Database Transparency Features

Minimum desirable DDBMS transparency features

4. **Performance transparency**
   - allows the system to perform as if it were a centralized DBMS.

5. **Heterogeneity transparency**
   - allows the integration of several different local DBMSs (relational, network, and hierarchical) under a common, or global, schema.

NOTE: These features are explained in detail later in this presentation

DDBMS

- Distribution Transparency
- Transaction Transparency
- Failure Transparency
- Performance Transparency
- Heterogeneity Transparency

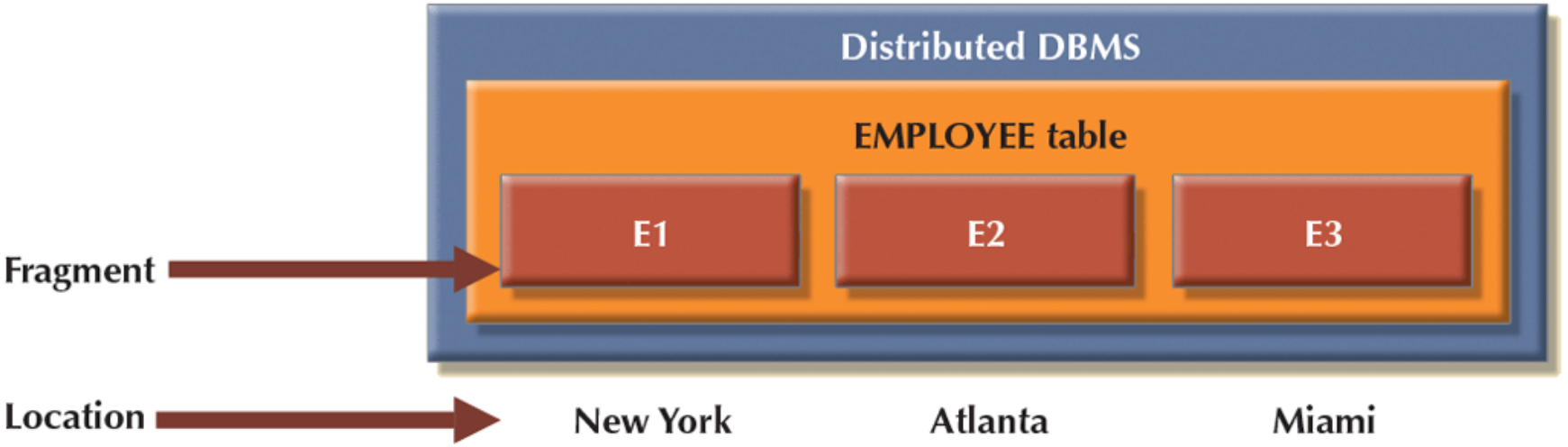
1. Distribution Transparency

Allows management of physically dispersed database as if centralized

Levels: fragmentation, location, and local mapping

SUMMARY OF TRANSPARENCY FEATURES			
IF THE SQL STATEMENT REQUIRES:			
FRAGMENT NAME?	LOCATION NAME?	THEN THE DBMS SUPPORTS	LEVEL OF DISTRIBUTON TRANSPARENCY
Yes	Yes	Local mapping transparency	Low
Yes	No	Location transparency	Medium
No	No	Fragmentation transparency	High

1. Distribution Transparency (Fragmentation Transparency)



Case 1: The Database Supports Fragmentation Transparency

The query conforms to a nondistributed database query format; that is, it does not specify fragment names or locations. The query reads:

```
SELECT      *
FROM        EMPLOYEE
WHERE       EMP_DOB < '01-JAN-1979';
```

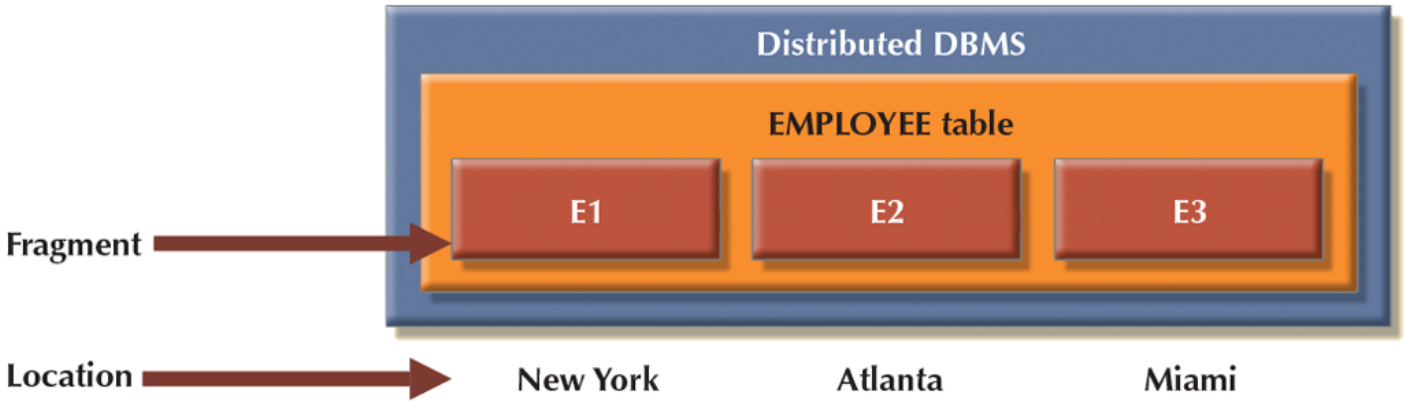
1. Distribution Transparency (Location Transparency)

Case 2: The Database Supports Location Transparency

Fragment names must be specified in the query, but the fragment's location is not specified. The query reads:

```
SELECT      *
FROM        E1

WHERE       EMP_DOB < '01-JAN-1979'
UNION
SELECT      *
FROM        E2
WHERE       EMP_DOB < '01-JAN-1979'
UNION
SELECT      *
FROM        E3
WHERE       EMP_DOB < '01-JAN-1979'
```

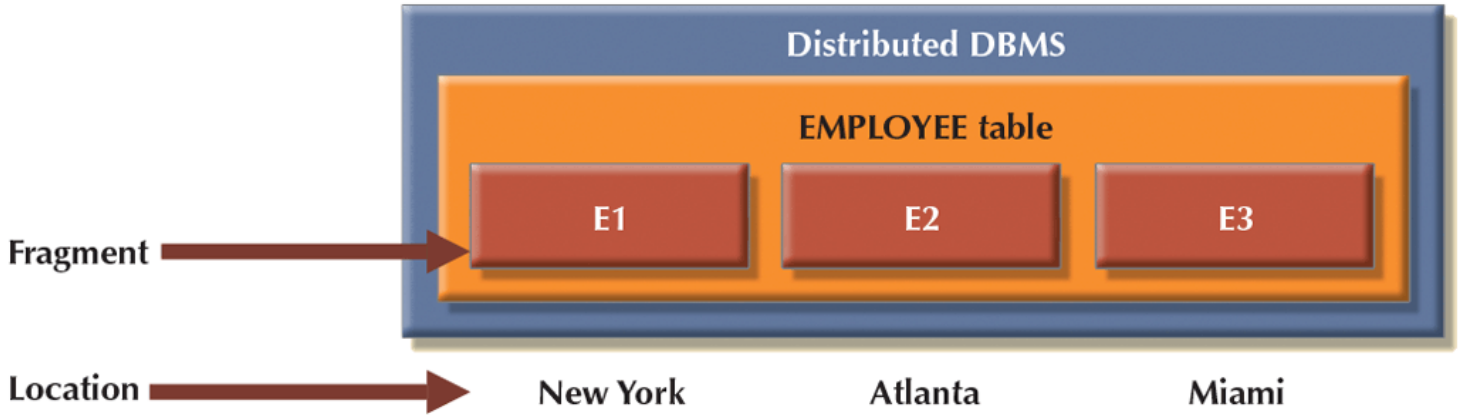


1. Distribution Transparency (Local Mapping Transparency)

Case 3: The Database Supports Local Mapping Transparency

Both the fragment name and its location must be specified in the query. Using pseudo-SQL:

```
SELECT      *
FROM        El NODE NY
WHERE       EMP_DOB < '01-JAN-1979';
UNION
SELECT      *
FROM        E2 NODE ATL
WHERE       EMP_DOB < '01-JAN-1979';
UNION
SELECT      *
FROM        E3 NODE MIA
WHERE       EMP_DOB < '01-JAN-1979';
```



1. Distribution Transparency (Contd.)

Unique fragment: each row is unique, regardless of the fragment in which it is located

Supported by distributed data dictionary (DDD) or distributed data catalog (DDC)

DDC contains the description of the entire database as seen by the database administrator

Distributed global schema: common database schema to translate user requests into subqueries

** Current implementations impose limitation on distribution transparency

DDBMS

- Distribution Transparency
- Transaction Transparency
- Failure Transparency
- Performance Transparency
- Heterogeneity Transparency

2. Transaction Transparency

Transaction transparency: DDBMS property that ensures database transactions will maintain the distributed database's integrity and consistency

- Ensures transaction completed only when all database sites involved complete their part

Distributed database systems require complex mechanisms to manage transactions

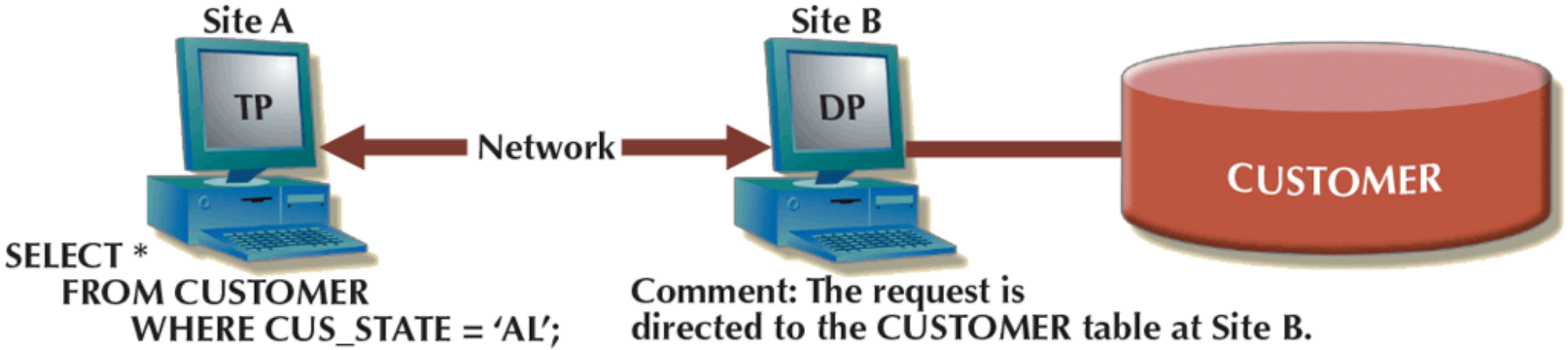
- Ensure the database's consistency and integrity

2. Transaction Transparency

Distributed Requests and Distributed Transactions

Remote request

Single SQL statement accesses data processed by a single remote database processor

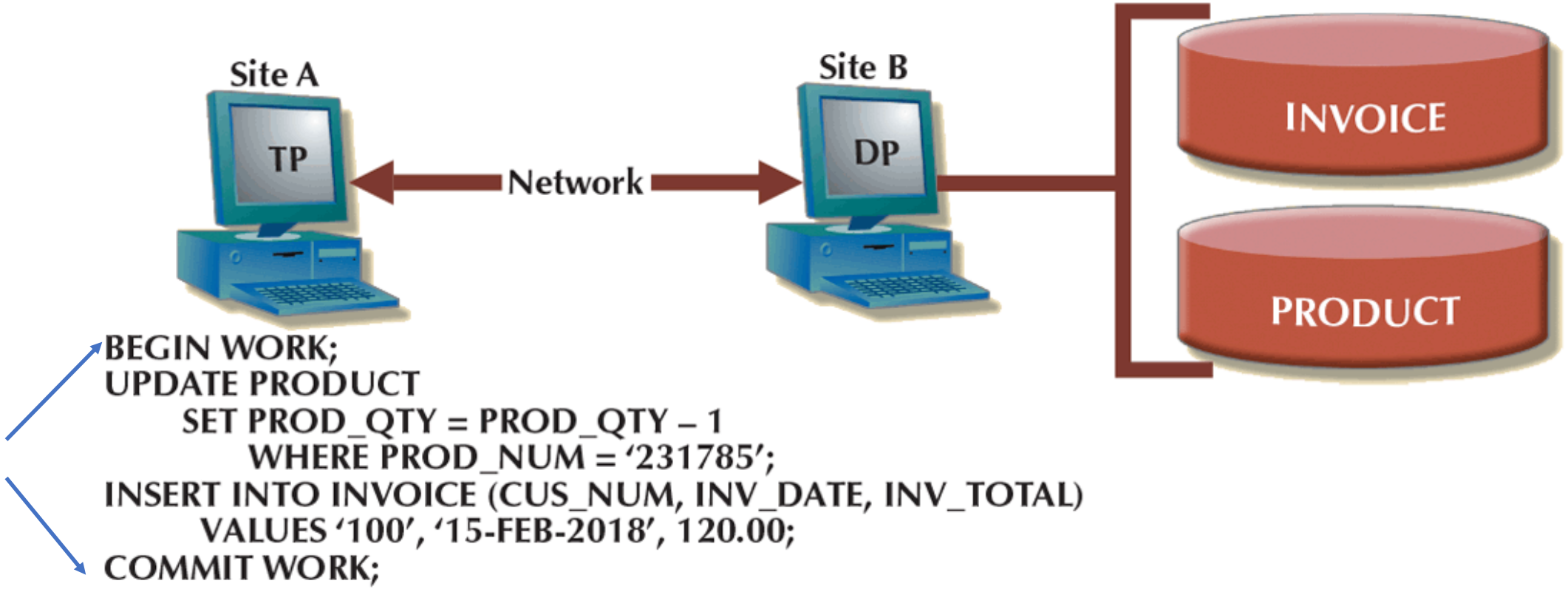


2. Transaction Transparency

Distributed Requests and Distributed Transactions

Remote transaction

Accesses data at single remote site composed of several requests

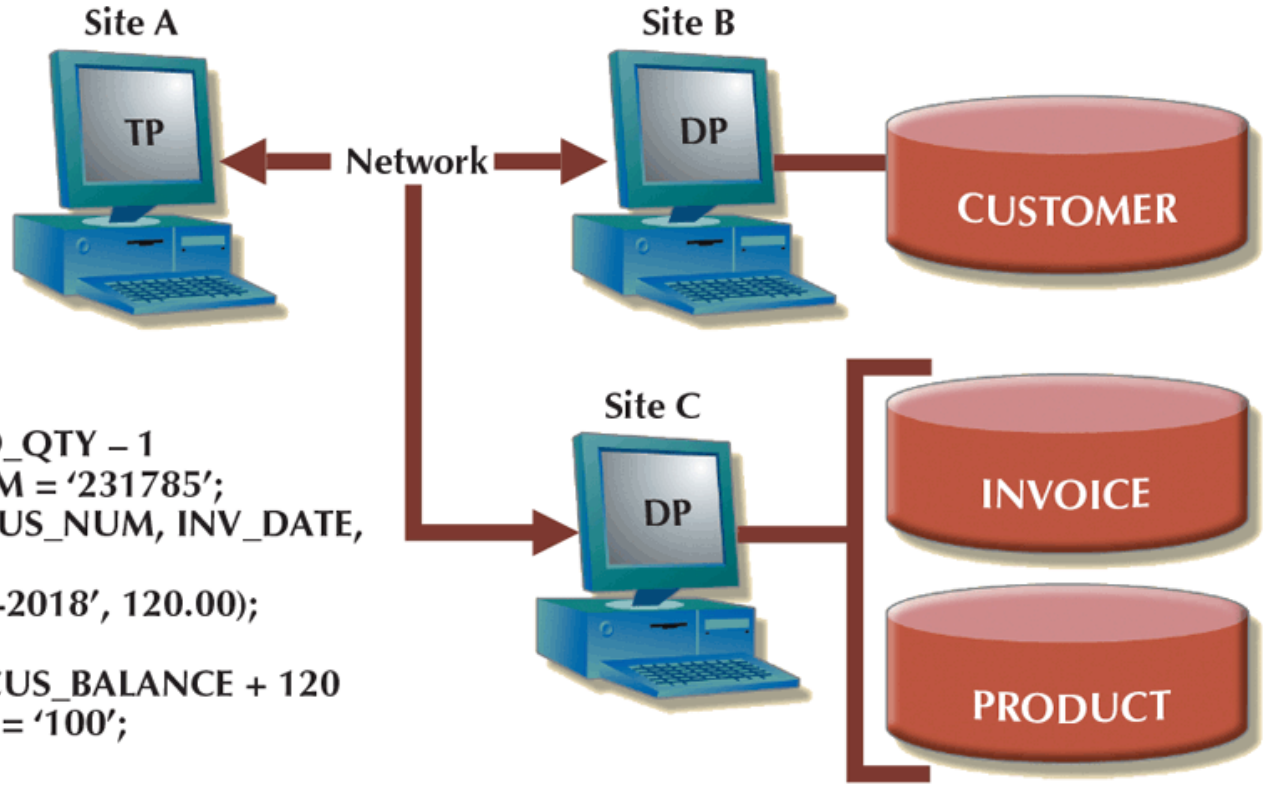


2. Transaction Transparency

Distributed Requests and Distributed Transactions

Distributed transaction

Requests data from several different remote sites on network



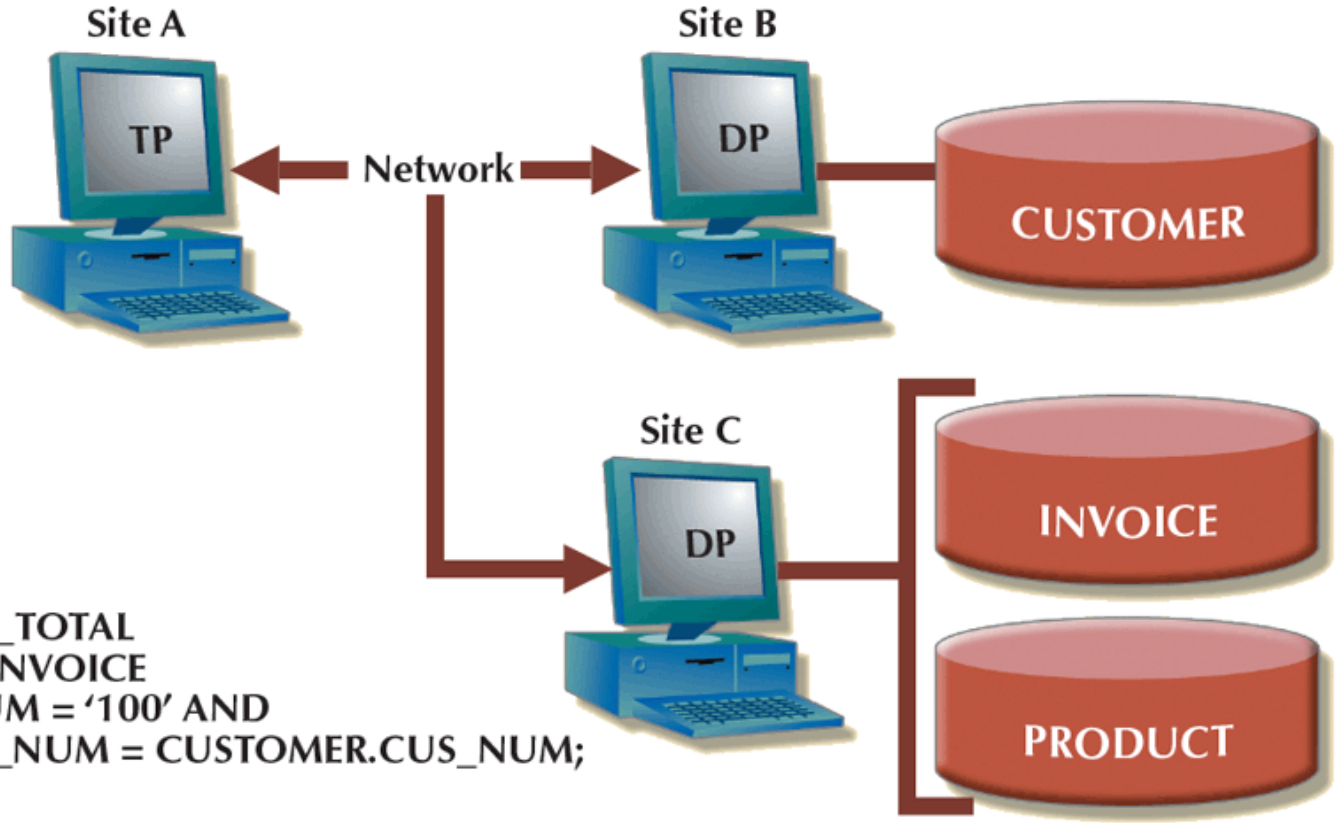

```
BEGIN WORK;
UPDATE PRODUCT
    SET PROD_QTY=PROD_QTY – 1
        WHERE PROD_NUM = '231785';
INSERT INTO INVOICE (CUS_NUM, INV_DATE,
  INV_TOTAL)
    VALUES ('100', '15-FEB-2018', 120.00);
UPDATE CUSTOMER
    SET CUS_BALANCE = CUS_BALANCE + 120
        WHERE CUS_NUM = '100';
COMMIT WORK;
```

2. Transaction Transparency

Distributed Requests and Distributed Transactions

Distributed request

Single SQL statement references data at several DP sites




```
BEGIN WORK;
   SELECT CUS_NUM, INV_TOTAL
      FROM CUSTOMER, INVOICE
         WHERE CUS_NUM = '100' AND
            INVOICE.CUS_NUM = CUSTOMER.CUS_NUM;
COMMIT WORK;
```

2. Transaction Transparency

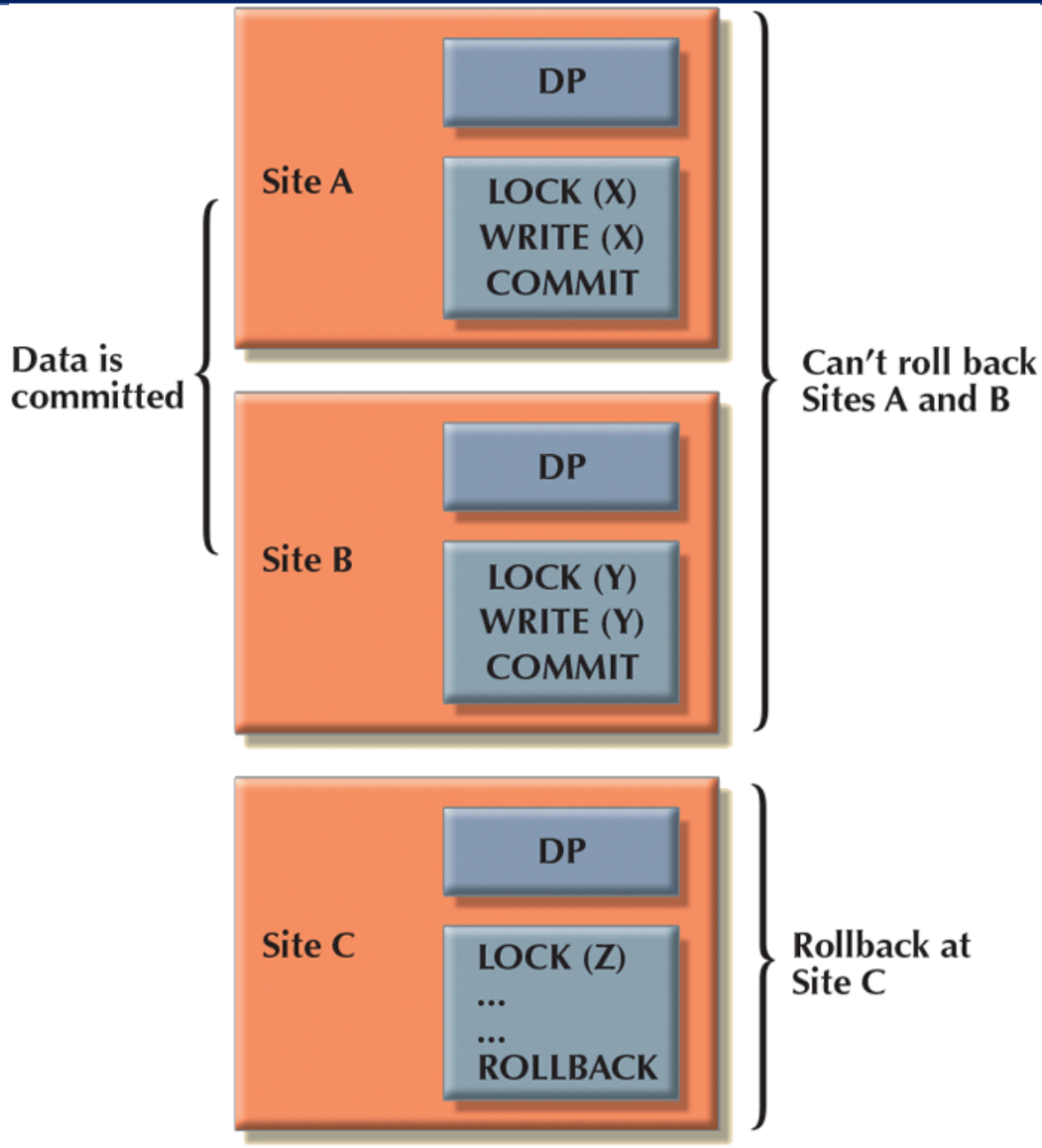
Distributed Concurrency Control

Concurrency control is especially important in distributed databases environment

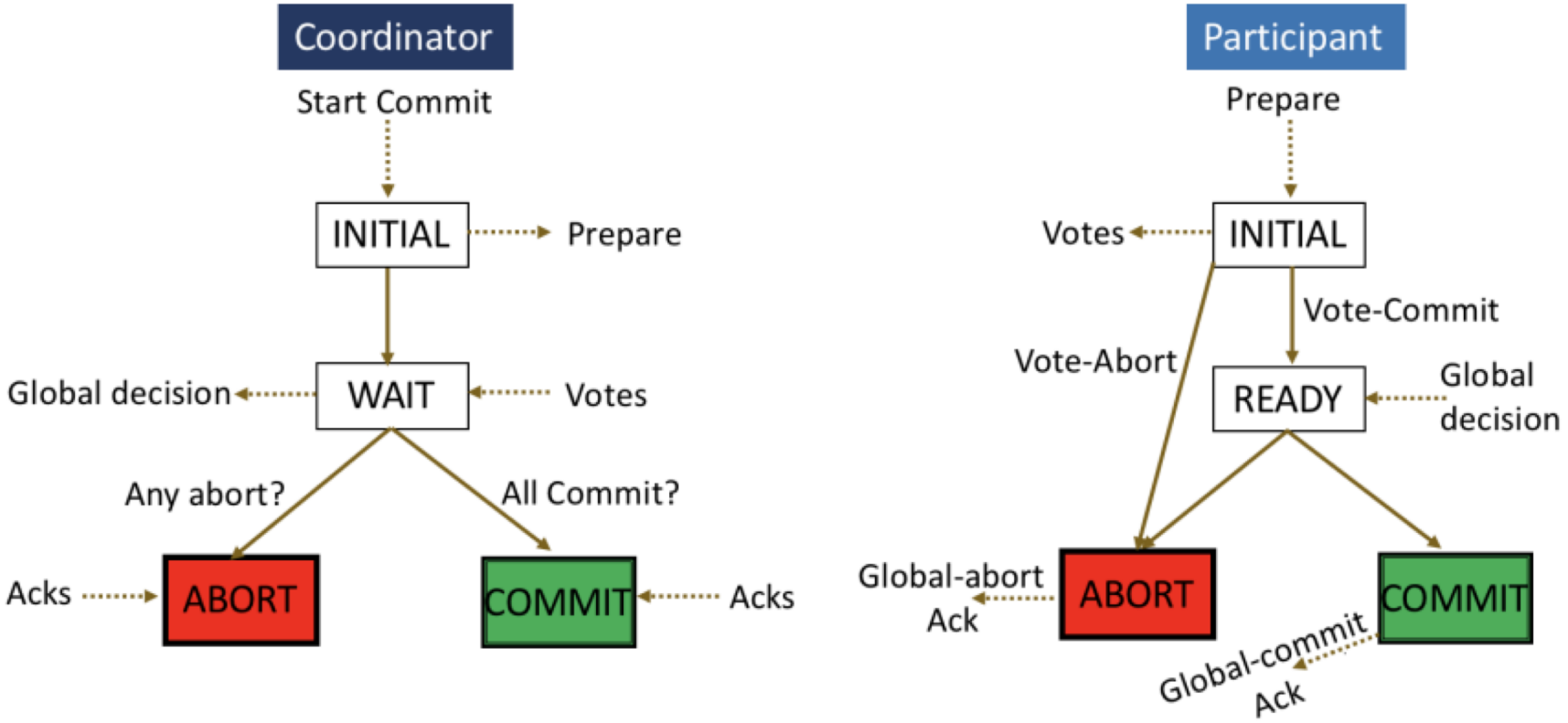
- Multi-site, multiple-process operations are more likely to create inconsistencies and deadlocked transactions
- Solution to inconsistent database is a two-phase commit protocol

2. Transaction Transparency

Database is inconsistent

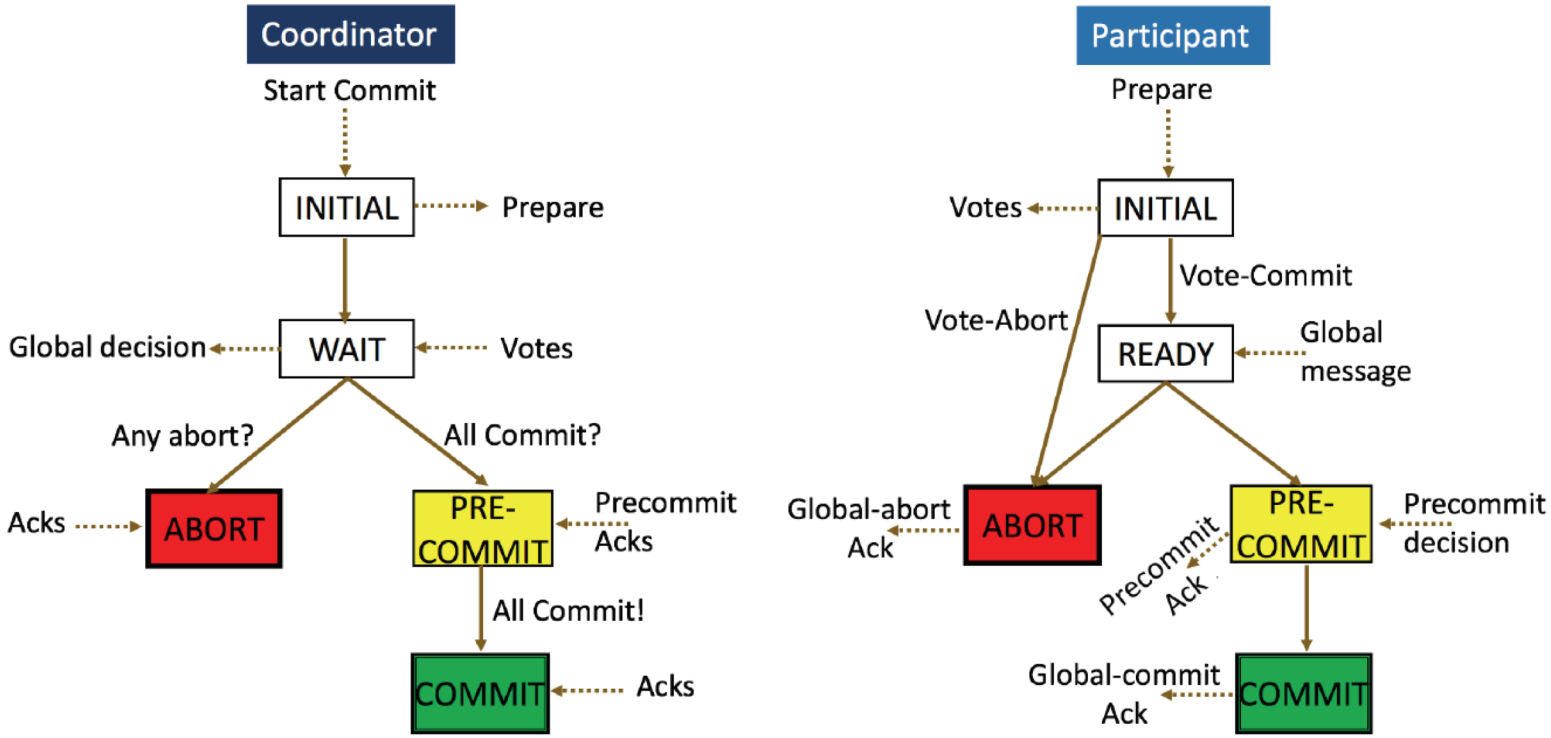


Two-phase Commit Protocol



Citation: Gupta, S., & Sadoghi, M. (2018). EasyCommit: A Non-blocking Two-phase Commit Protocol. *EDBT*.

Three-phase Commit Protocol



Citation: Gupta, S., & Sadoghi, M. (2018). EasyCommit: A Non-blocking Two-phase Commit Protocol. *EDBT*.

DDBMS

- Distribution Transparency
- Transaction Transparency
- Failure Transparency
- Performance Transparency
- Heterogeneity Transparency

3 & 4. Performance and Failure Transparency

Performance transparency allows a DDBMS to perform as if it were a centralized database

- Failure transparency ensures the system will operate in case of network failure

Objective of query optimization is to minimize total costs

- Access time (I/O) cost involved in accessing data from multiple remote sites
- Communication costs associated with data transmission
- CPU time cost associated with the processing overhead

3 & 4. Performance and Failure Transparency

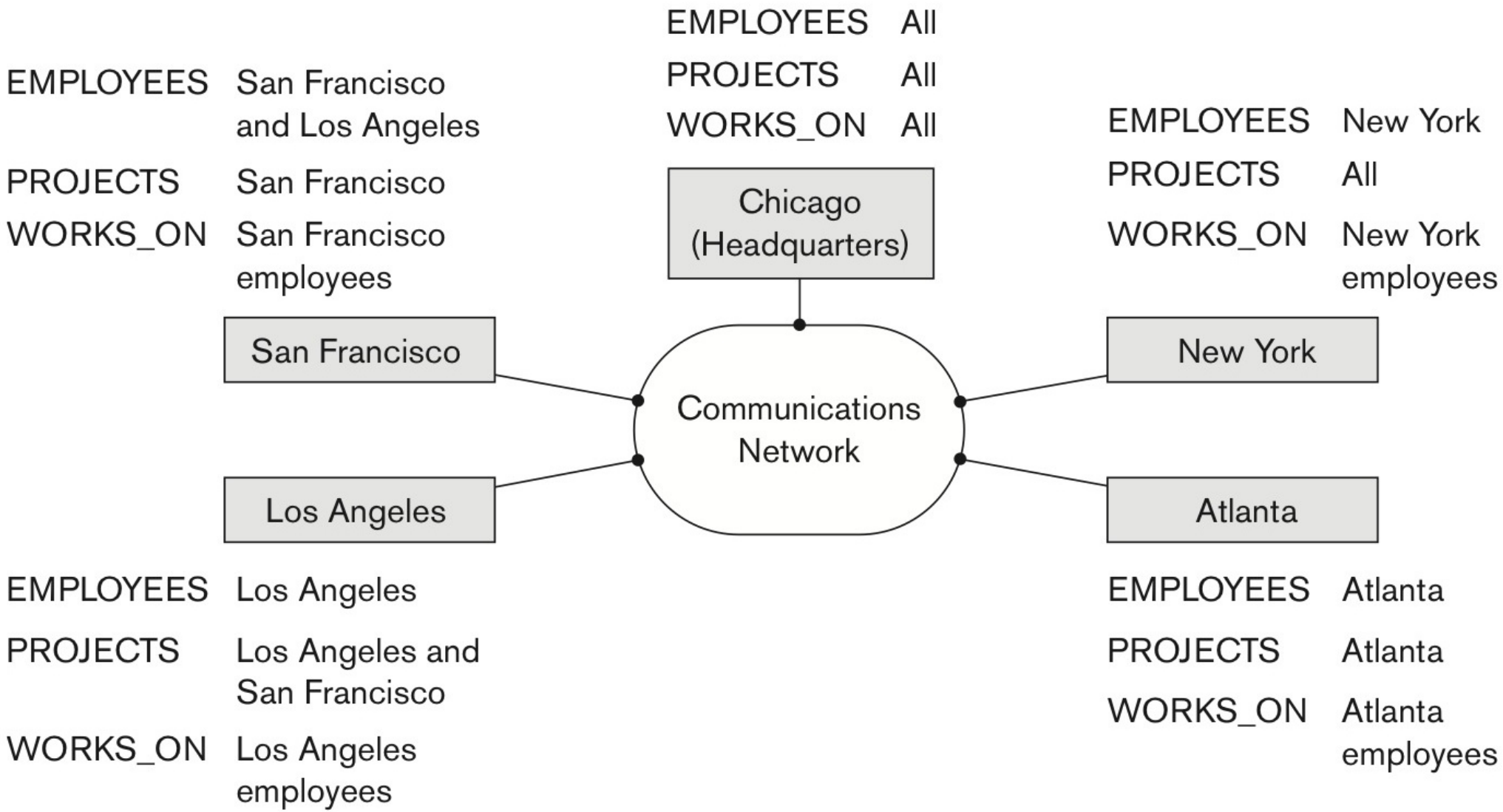
Considerations for resolving data requests in a distributed data environment

Data distribution and data replication

- **Replica transparency:** DDBMS's ability to hide multiple copies of data from the user

Network and node availability

- **Network latency:** delay imposed by the amount of time required for a data packet to make a round trip
- **Network partitioning:** delay imposed when nodes become suddenly unavailable due to a network failure


EMPLOYEES All
PROJECTS All
WORKS_ON All
EMPLOYEES San Francisco and Los Angeles
PROJECTS San Francisco
WORKS_ON San Francisco employees
Chicago (Headquarters)
EMPLOYEES New York
PROJECTS All
WORKS_ON New York employees
San Francisco
Communications Network
New York
Los Angeles
Atlanta
EMPLOYEES Los Angeles
PROJECTS Los Angeles and San Francisco
WORKS_ON Los Angeles employees
EMPLOYEES Atlanta
PROJECTS Atlanta
WORKS_ON Atlanta employees

Distributed Database Design

Data fragmentation

How to partition database into fragments

Data replication

Which fragments to replicate

Data allocation

Where to locate those fragments and replicas

Data Fragmentation

Breaks a single object into two or more segments

- Information is stored in distributed data catalog (DDC)
- Each fragment can be stored at any site over a computer network.

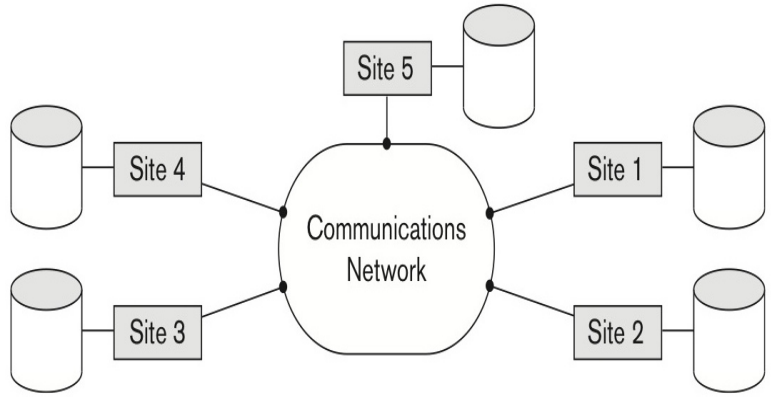
Strategies

- Horizontal fragmentation: division of a relation into subsets (fragments) of tuples (rows)
- Vertical fragmentation: division of a relation into attribute (column) subsets
- Mixed fragmentation: combination of horizontal and vertical strategies

Horizontal fragmentation: division of a relation into subsets (fragments) of tuples (rows)

Table name: CUSTOMER **Database name: Ch12_Text**

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_STATE	CUS_LIMIT	CUS_BAL	CUS_RATING	CUS_DUE
10	Sinex, Inc.	12 Main St.	TN	3500.00	2700.00	3	1245.00
11	Martin Corp.	321 Sunset Blvd.	FL	6000.00	1200.00	1	0.00
12	Mynux Corp.	910 Eagle St.	TN	4000.00	3500.00	3	3400.00
13	BTBC, Inc.	Rue du Monde	FL	6000.00	5890.00	3	1090.00
14	Victory, Inc.	123 Maple St.	FL	1200.00	550.00	1	0.00
15	NBCC Corp.	909 High Ave.	GA	2000.00	350.00	2	50.00



Round-robin partitioning [X]

Range partitioning based on a partition key [✓]

Database name: Ch12_Text

Table name: CUST_H1 **Location: Tennessee** **Node: NAS**

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_STATE	CUS_LIMIT	CUS_BAL	CUS_RATING	CUS_DUE
10	Sinex, Inc.	12 Main St.	TN	3500.00	2700.00	3	1245.00
12	Mynux Corp.	910 Eagle St.	TN	4000.00	3500.00	3	3400.00

Table name: CUST_H2 **Location: Georgia** **Node: ATL**

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_STATE	CUS_LIMIT	CUS_BAL	CUS_RATING	CUS_DUE
15	NBCC Corp.	909 High Ave.	GA	2000.00	350.00	2	50.00

Table name: CUST_H3 **Location: Florida** **Node: TAM**

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_STATE	CUS_LIMIT	CUS_BAL	CUS_RATING	CUS_DUE
11	Martin Corp.	321 Sunset Blvd.	FL	6000.00	1200.00	1	0.00
13	BTBC, Inc.	Rue du Monde	FL	6000.00	5890.00	3	1090.00
14	Victory, Inc.	123 Maple St.	FL	1200.00	550.00	1	0.00

Vertical fragmentation: division of a relation into attribute (column) subsets

Table name: CUSTOMER **Database name: Ch12_Text**

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_STATE	CUS_LIMIT	CUS_BAL	CUS_RATING	CUS_DUE
10	Sinex, Inc.	12 Main St.	TN	3500.00	2700.00	3	1245.00
11	Martin Corp.	321 Sunset Blvd.	FL	6000.00	1200.00	1	0.00
12	Mynux Corp.	910 Eagle St.	TN	4000.00	3500.00	3	3400.00
13	BTBC, Inc.	Rue du Monde	FL	6000.00	5890.00	3	1090.00
14	Victory, Inc.	123 Maple St.	FL	1200.00	550.00	1	0.00
15	NBCC Corp.	909 High Ave.	GA	2000.00	350.00	2	50.00

Table name: CUST_V1 **Location: Service Building** **Node: SVC**

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_STATE
10	Sinex, Inc.	12 Main St.	TN
11	Martin Corp.	321 Sunset Blvd.	FL
12	Mynux Corp.	910 Eagle St.	TN
13	BTBC, Inc.	Rue du Monde	FL
14	Victory, Inc.	123 Maple St.	FL
15	NBCC Corp.	909 High Ave.	GA

Table name: CUST_V2 **Location: Collection Building** **Node: ARC**

CUS_NUM	CUS_LIMIT	CUS_BAL	CUS_RATING	CUS_DUE
10	3500.00	2700.00	3	1245.00
11	6000.00	1200.00	1	0.00
12	4000.00	3500.00	3	3400.00
13	6000.00	5890.00	3	1090.00
14	1200.00	550.00	1	0.00
15	2000.00	350.00	2	50.00

Suppose that the company is divided into two departments: the service department and the collections department.

Mixed fragmentation: combination of horizontal and vertical strategies

The XYZ Company's structure requires that the CUSTOMER data be fragmented horizontally to accommodate the various company locations; within the locations, the data must be fragmented vertically to accommodate the two departments (service and collection). In short, the CUSTOMER table requires mixed fragmentation.

Table name: CUST_M1 **Location: TN-Service** **Node: NAS-S**

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_STATE
10	Sinex, Inc.	12 Main St.	TN
12	Mynux Corp.	910 Eagle St.	TN

Table name: CUST_M2 **Location: TN-Collection** **Node: NAS-C**

CUS_NUM	CUS_LIMIT	CUS_BAL	CUS_RATING	CUS_DUE
10	3500.00	2700.00	3	1245.00
12	4000.00	3500.00	3	3400.00

Table name: CUST_M3 **Location: GA-Service** **Node: ATL-S**

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_STATE
15	NBCC Corp.	909 High Ave.	GA

Table name: CUST_M4 **Location: GA-Collection** **Node: ATL-C**

CUS_NUM	CUS_LIMIT	CUS_BAL	CUS_RATING	CUS_DUE
15	2000.00	350.00	2	50.00

Table name: CUST_M5 **Location: FL-Service** **Node: TAM-S**

CUS_NUM	CUS_NAME	CUS_ADDRESS	CUS_STATE
11	Martin Corp.	321 Sunset Blvd.	FL
13	BTBC, Inc.	Rue du Monde	FL
14	Victory, Inc.	123 Maple St.	FL

Table name: CUST_M6 **Location: FL-Collection** **Node: TAM-C**

CUS_NUM	CUS_LIMIT	CUS_BAL	CUS_RATING	CUS_DUE
11	6000.00	1200.00	1	0.00
13	6000.00	5890.00	3	1090.00
14	1200.00	550.00	1	0.00

Data Replication

Storage of data copies at multiple sites served by a computer network

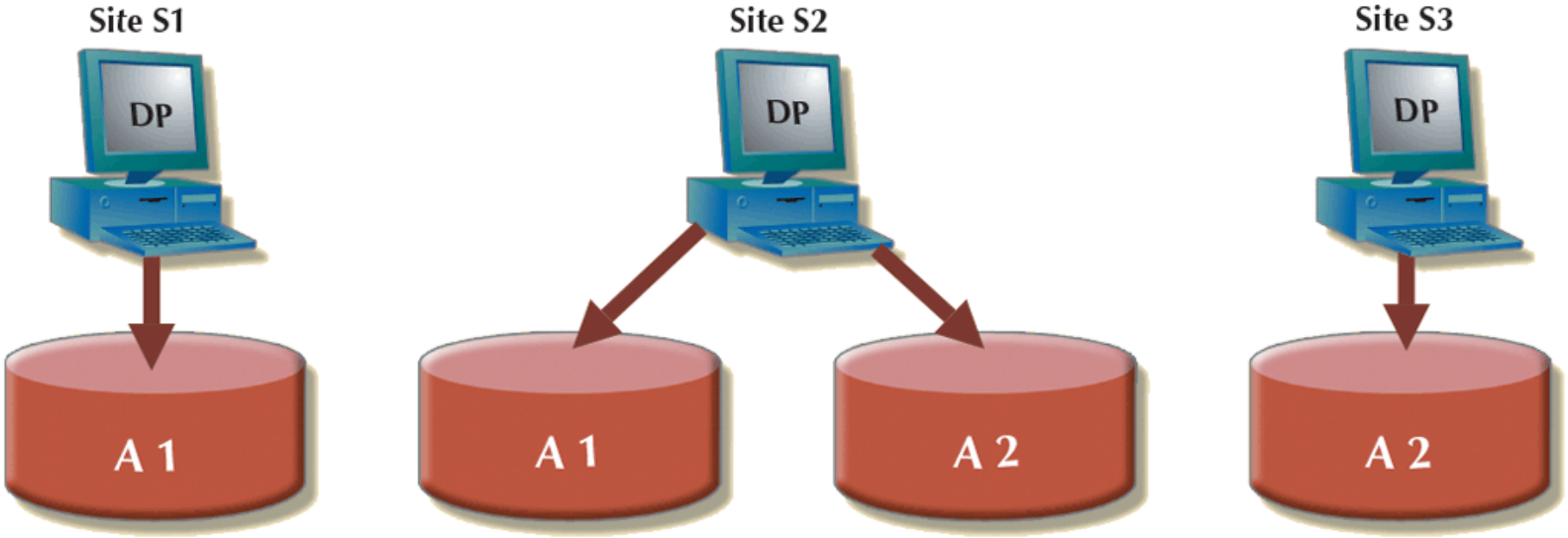
Mutual consistency rule requires all copies of data fragments be identical

Styles of replication

- Push replication focuses on maintaining data consistency
- Pull replication focuses on maintaining data availability and allows for temporary data inconsistencies

Data Replication

Suppose database A is divided into two fragments, A1 and A2. Fragment A1 is stored at Sites S1 and S2, while fragment A2 is stored at Sites S2 and S3.



Data Replication – Push replication

- After a data update, the originating DP node sends the changes to the replica nodes to ensure that data is immediately updated.
- This type of replication focuses on maintaining data consistency.
- Decreases data availability due to the latency involved in ensuring data consistency at all nodes.

Data Replication – pull replication

- After a data update, the originating DP node sends “messages” to the replica nodes to notify them of the update.
- The replica nodes decide when to apply the updates to their local fragment.
- In this type of replication, data updates propagate more slowly to the replicas. The focus is on maintaining data availability.
- This style of replication allows for temporary data inconsistencies.

Data Replication – pull replication

Data replication scenarios

- **Fully replicated database:** stores multiple copies of each database fragment at multiple sites
- **Partially replicated database:** stores multiple copies of some database fragments at multiple sites
- **Unreplicated database:** stores each database fragment at a single site

DDBMS

- Distribution Transparency
- Transaction Transparency
- Failure Transparency
- Performance Transparency
- Heterogeneity Transparency

End of Lecture Questions

1. For an organization with 5 branches, when will centralized DBMS be a good choice over DDBMS?

2. Is implementing DDBMS always expensive than centralized DBMS?