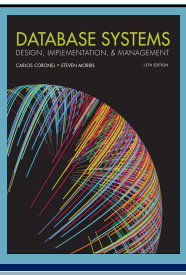


Chapter 12

Distributed Database Management Systems

Learning Objectives

- After completing this chapter, you will be able to:
  - Explain the purpose and function of distributed database management systems (DDBMSs)
  - Summarize the advantages and disadvantages of DDBMSs
  - Describe the characteristics and components of DDBMSs
  - Explain how database implementation is affected by different levels of data and process distribution
  - Understand how transactions are managed in a distributed database environment
  - Describe how distributed database design balances performance, scalability, and availability
  - Explain the trade-offs of implementing a distributed data system

CENGAGE

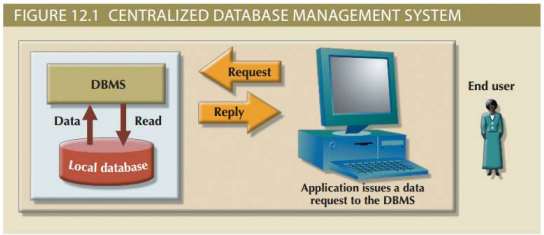
The Evolution of Distributed Database Management Systems (1 of 5)

- 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
- Centralized database management system
  - Required corporate data be stored in a single central site
  - Provided data access through dumb terminals
  - Filled structured information needs of corporations; fell short when quickly moving events required faster response times and equally quick access to information

CENGAGE

The Evolution of Distributed Database Management Systems (2 of 5)

FIGURE 12.1 CENTRALIZED DATABASE MANAGEMENT SYSTEM



CENGAGE

The Evolution of Distributed Database Management Systems (3 of 5)

- Changes that affected the nature of systems
  - Globalization of business operation
  - Increased market needs for an on-demand transaction style, based on web-based services
  - Rapid social and technological changes fueled by low-cost smart mobile devices
  - Converging data realms in the digital world
  - Advent of social media to reach new customers and markets

CENGAGE

The Evolution of Distributed Database Management Systems (4 of 5)

- 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

CENGAGE

The Evolution of Distributed Database Management Systems (5 of 5)

- Factors that influenced DDBMS
  - Acceptance of Internet as a platform for business
  - Mobile wireless revolution
  - Growth of use of "application as a service"
  - Focus on mobile business intelligence
  - Emphasis on Big Data analytics
- Potential centralized DBMS problems
  - Performance degradation
  - High costs
  - Reliability problems
  - Scalability problems
  - Organizational rigidity

CENGAGE

DDBMS Advantages and Disadvantages

Table 12.1: Distributed DBMS 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 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. Probability of security lapses increases when data is located at multiple sites. Responsibility of data management will be shared by different people at several sites.
Growth facilitation. New sites can be added to the network without affecting the operations of other sites.	Lack of standards. No standard communication protocols at the database level. 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's 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.
User-friendly interface. Client devices are usually equipped with an easy-to-use graphical user interface (GUI). The G UI 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.	

CENGAGE

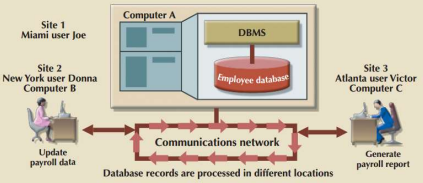
Distributed Processing and Distributed Databases (1 of 3)

- Distributed processing: database's logical processing is shared among two or more physically independent sites via network
  - Distributed database: stores logically related database over two or more physically independent sites via a computer network
  - Database fragments: database composed of many parts in distributed database system

CENGAGE

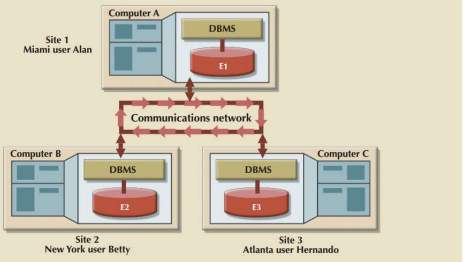
Distributed Processing and Distributed Databases (2 of 3)

FIGURE 12.2 DISTRIBUTED PROCESSING ENVIRONMENT



Distributed Processing and Distributed Databases (3 of 3)

FIGURE 12.3 DISTRIBUTED DATABASE ENVIRONMENT



Characteristics of Distributed Management Systems (1 of 3)

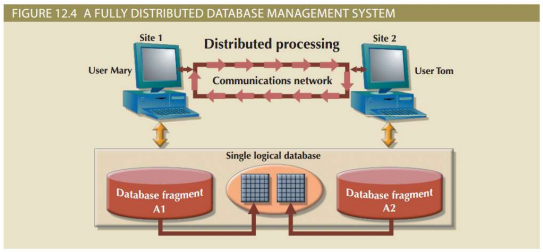
- A DBMS must have several functions to be classified as distributed
  - Application interface
  - Validation
  - Transformation
  - Query optimization
  - Mapping
  - I/O interface
  - Formatting
  - Security
  - Backup and recovery
  - DB administration
  - Concurrency control
  - Transaction management

Characteristics of Distributed Management Systems (2 of 3)

- Functions of fully distributed DBMS
  - Receive the request of an application or end user
  - Validate, analyze, and decompose the request
  - Map request's logical-to-physical data components
  - Decompose request into several I/O operations
  - Search, locate, read and validate data
  - Ensure database consistency, security, and integrity
  - Validate data for conditions specified by request
  - Present data in required format
  - Handle all necessary functions transparently to user

Characteristics of Distributed Management Systems (3 of 3)

FIGURE 12.4 A FULLY DISTRIBUTED DATABASE MANAGEMENT SYSTEM



DDBMS Components

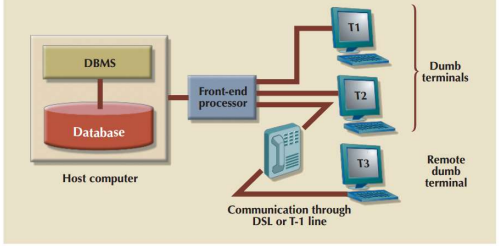
- The DDBMS must include at least the following components:
  - Computer workstations or remote devices
  - Network hardware and software components
  - Communications media
  - Transaction processor (TP)
  - Data processor (DP) or data manager (DM)

Single-Site Processing, Single-Site Data (SPSD) (1 of 2)

- 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, Single-Site Data (SPSD) (2 of 2)

FIGURE 12.6 SINGLE-SITE PROCESSING, SINGLE-SITE DATA (CENTRALIZED)

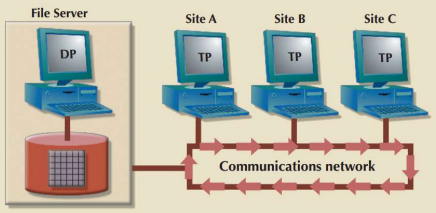


Multiple-Site Processing, Single-Site Data (MPSD) (1 of 2)

- 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 (MPSD) (2 of 2)

FIGURE 12.7 MULTIPLE-SITE PROCESSING, SINGLE-SITE DATA



Multiple-Site Processing, Multiple-Site Data (MPMD) (1 of 2)

- 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 (MPMD) (2 of 2)

- 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
  - Distribution transparency
  - Transaction transparency
  - Failure transparency
  - Performance transparency
  - Heterogeneity transparency

Distribution Transparency

- Allows management of physically dispersed database as if centralized
  - Levels: fragmentation, location, and local mapping
    - 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

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

Distributed Requests and Distributed Transactions

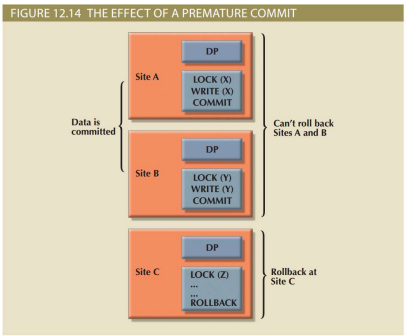
- Remote request
  - Single SQL statement accesses data processed by a single remote database processor
- Remote transaction
  - Accesses data at single remote site composed of several requests
- Distributed transaction
  - Requests data from several different remote sites on network
- Distributed request
  - Single SQL statement references data at several DP sites

Distributed Concurrency Control (1 of 2)

- 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

Distributed Concurrency Control (2 of 2)

FIGURE 12.14 THE EFFECT OF A PREMATURE COMMIT



Distributed Database Design

- Data fragmentation
  - How to partition database into fragments
  - Breaks a single object into two or more segments
- Data replication
  - Which fragments to replicate
  - Storage of data copies at multiple sites served by a computer network
  - Mutual consistency rule requires all copies of data fragments be identical
- Data allocation
  - Where to locate those fragments and replicas
  - 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

The CAP Theorem (1 of 2)

- CAP stands for:
  - Consistency
  - Availability
  - Partition tolerance
- Trade-off between consistency and availability generated in a new system in which data is basically available, soft state, eventually consistent (BASE)
  - Data changes are not immediate but propagate slowly through the system until all replicas are consistent

The CAP Theorem (2 of 2)

Table 12.8
Distributed Database Spectrum

DBMS Type	Consistency	Availability	Partition Tolerance	Transaction Model	Trade-Off
Centralized DBMS	High	High	N/A	ACID	No distributed data processing
Relational DBMS	High	Relaxed	High	ACID (2PC)	Sacrifices availability to ensure consistency and isolation
NoSQL DDBMS	Relaxed	High	High	BASE	Sacrifices consistency to ensure availability
NewSQL DDBMS	High	High	Relaxed	ACID	Sacrifices partition tolerance to ensure transaction consistency and availability

C. J. Date's 12 Commandments for Distributed Databases

Table 12.9	C. J. Date 's 12 Commandments for Distributed Databases	
Rule Number	**Rule Name**	**Rule Explanation**
1	Local-site independence	Each local site can act as an independent, autonomous, centralized DBMS. Each site is responsible for security, concurrency control, backup, and recovery.
2	Central-site independence	Each local site can act as an independent, autonomous, centralized DBMS. Each site is responsible for security, concurrency control, backup, and recovery.
3	Failure independence	The system is not affected by node failures. The system is in continuous operation even in the case of a node failure or an expansion of the network.
4	Failure Independence	The user does not need to know the location of data to retrieve that data.
5	Fragmentation transparency	Data fragmentation is transparent to the user, who sees only one logical database. The user does not need to know the name of the database fragments to retrieve them.
6	Replication transparency	The user sees only one logical database. The DDBMS transparently selects the database fragment to access. To the user, the DDBMS manages all fragments transparently.
7	Distributed query processing	A distributed query may be executed at several different DP sites. Query optimization is performed transparently by the DDBMS.
8	Distributed transaction processing	A transaction may update data at several different sites, and the transaction is executed transparently.
9	Hardware independence	The system must run on any hardware platform.
10	Operating system independence	The system must run on any operating system platform.
11	Network independence	The system must run on any network platform.
12	Database independence	The system must support any vendor's database product.

CENGAGE