CHAPTER 23

Distributed Database Concepts
Introduction

- **Distributed computing system**
  - Consists of several processing sites or nodes interconnected by a computer network
  - Nodes cooperate in performing certain tasks
  - Partitions large task into smaller tasks for efficient solving

- **Big data technologies**
  - Combine distributed and database technologies
  - Deal with mining vast amounts of data
What constitutes a distributed database?

- Connection of database nodes over computer network
- Logical interrelation of the connected databases
- Possible absence of homogeneity among connected nodes

Distributed database management system (DDBMS)

- Software system that manages a distributed database
Distributed Database Concepts (cont’d.)

- Local area network
  - Hubs or cables connect sites
- Long-haul or wide area network
  - Telephone lines, cables, wireless, or satellite connections
- Network topology defines communication path
- Transparency
  - Hiding implementation details from the end user
Transparency

- Types of transparency
  - Data organization transparency
    - Location transparency
    - Naming transparency
  - Replication transparency
  - Fragmentation transparency
    - Horizontal fragmentation
    - Vertical fragmentation
  - Design transparency
  - Execution transparency
Figure 23.1 Data distribution and replication among distributed databases
Availability and Reliability

- **Availability**
  - Probability that the system is continuously available during a time interval

- **Reliability**
  - Probability that the system is running (not down) at a certain time point

- Both directly related to faults, errors, and failures
- Fault-tolerant approaches
Scalability and Partition Tolerance

- **Horizontal scalability**
  - Expanding the number of nodes in a distributed system

- **Vertical scalability**
  - Expanding capacity of the individual nodes

- **Partition tolerance**
  - System should have the capacity to continue operating while the network is partitioned
Autonomy

- Determines extent to which individual nodes can operate independently

  - Design autonomy
    - Independence of data model usage and transaction management techniques among nodes

  - Communication autonomy
    - Determines the extent to which each node can decide on sharing information with other nodes

  - Execution autonomy
    - Independence of users to act as they please
Advantages of Distributed Databases

- Improved ease and flexibility of application development
  - Development at geographically dispersed sites
- Increased availability
  - Isolate faults to their site of origin
- Improved performance
  - Data localization
- Easier expansion via scalability
  - Easier than in non-distributed systems
23.2 Data Fragmentation, Replication, and Allocation Techniques for Distributed Database Design

- **Fragments**
  - Logical units of the database

- **Horizontal fragmentation (sharding)**
  - Horizontal fragment or shard of a relation is a subset of the tuples in that relation
  - Can be specified by condition on one or more attributes or by some other method
  - Groups rows to create subsets of tuples
    - Each subset has a certain logical meaning
Data Fragmentation (cont’d.)

- **Vertical fragmentation**
  - Divides a relation vertically by columns
  - Keeps only certain attributes of the relation

- **Complete horizontal fragmentation**
  - Apply UNION operation to the fragments to reconstruct relation

- **Complete vertical fragmentation**
  - Apply OUTER UNION or FULL OUTER JOIN operation to reconstruct relation
Data Fragmentation (cont’d.)

- **Mixed (hybrid) fragmentation**
  - Combination of horizontal and vertical fragmentations

- **Fragmentation schema**
  - Defines a set of fragments that includes all attributes and tuples in the database

- **Allocation schema**
  - Describes the allocation of fragments to nodes of the DDBS
Data Replication and Allocation

- Fully replicated distributed database
  - Replication of whole database at every site in distributed system
  - Improves availability remarkably
  - Update operations can be slow
- Nonredundant allocation (no replication)
  - Each fragment is stored at exactly one site
Partial replication
- Some fragments are replicated and others are not
- Defined by replication schema

Data allocation (data distribution)
- Each fragment assigned to a particular site in the distributed system
- Choices depend on performance and availability goals of the system
Example of Fragmentation, Allocation, and Replication

- **Company with three computer sites**
  - One for each department
  - Expect frequent access by employees working in the department and projects controlled by that department

- See Figures 23.2 and 23.3 in the text for example fragmentation among the three sites
23.3 Overview of Concurrency Control and Recovery in Distributed Databases

- Problems specific to distributed DBMS environment
  - Multiple copies of the data items
  - Failure of individual sites
  - Failure of communication links
  - Distributed commit
  - Distributed deadlock
Distributed Concurrency Control Based on a Distinguished Copy of a Data Item

- Particular copy of each data item designated as distinguished copy
  - Locks are associated with the distinguished copy
- Primary site technique
  - All distinguished copies kept at the same site
- Primary site with backup site
  - Locking information maintained at both sites
- Primary copy method
  - Distributes the load of lock coordination among various sites
Distributed Concurrency Control Based on Voting

- **Voting method**
  - No distinguished copy
  - Lock requests sent to all sites that contain a copy
  - Each copy maintains its own lock
  - If transaction that requests a lock is granted that lock by a majority of the copies, it holds the lock on all copies
  - Time-out period applies
  - Results in higher message traffic among sites
Distributed Recovery

- Difficult to determine whether a site is down without exchanging numerous messages with other sites

- Distributed commit
  - When a transaction is updating data at several sites, it cannot commit until certain its effect on every site cannot be lost
  - Two-phase commit protocol often used to ensure correctness
23.4 Overview of Transaction Management in Distributed Databases

- **Global transaction manager**
  - Supports distributed transactions
  - Role temporarily assumed by site at which transaction originated
    - Coordinates execution with transaction managers at multiple sites
  - Passes database operations and associated information to the concurrency controller
    - Controller responsible for acquisition and release of locks
Commit Protocols

- **Two-phase**
  - Coordinator maintains information needed for recovery
    - In addition to local recovery managers

- **Three-phase**
  - Divides second commit phase into two subphases
    - Prepare-to-commit phase communicates result of the vote phase
    - Commit subphase same as two-phase commit counterpart
23.5 Query Processing and Optimization in Distributed Databases

- Stages of a distributed database query
  - Query mapping
    - Refers to global conceptual schema
  - Localization
    - Maps the distributed query to separate queries on individual fragments
  - Global query optimization
    - Strategy selected from list of candidates
  - Local query optimization
    - Common to all sites in the DDB
Query Processing and Optimization in Distributed Databases (cont’d.)

- Data transfer costs of distributed query processing
  - Cost of transferring intermediate and final result files
- Optimization criterion: reducing amount of data transfer
Distributed query processing using semijoin

- Reduces the number of tuples in a relation before transferring it to another site
- Send the joining column of one relation R to one site where the other relation S is located
- Join attributes and result attributes shipped back to original site
- Efficient solution to minimizing data transfer
Query and update decomposition

- User can specify a query as if the DBMS were centralized
  - If full distribution, fragmentation, and replication transparency are supported
- Query decomposition module
  - Breaks up a query into subqueries that can be executed at the individual sites
  - Strategy for combining results must be generated
- Catalog stores attribute list and/or guard condition
23.6 Types of Distributed Database Systems

- Factors that influence types of DDBMSs
  - Degree of homogeneity of DDBMS software
    - Homogeneous
    - Heterogeneous
  - Degree of local autonomy
    - No local autonomy
    - Multidatabase system has full local autonomy
- Federated database system (FDBS)
  - Global view or schema of the federation of databases is shared by the applications
Classification of Distributed Databases

Figure 23.6 Classification of distributed databases

Legend:
- A: Traditional centralized database systems
- B: Pure distributed database systems
- C: Federated database systems
- D: Multidatabase or peer-to-peer database systems
Types of Distributed Database Systems (cont’d.)

- **Federated database management systems issues**
  - Differences in data models
  - Differences in constraints
  - Differences in query languages

- **Semantic heterogeneity**
  - Differences in meaning, interpretation, and intended use of the same or related data
Design autonomy allows definition of the following parameters

- The universe of discourse from which the data is drawn
- Representation and naming
- Understanding, meaning, and subjective interpretation of data
- Transaction and policy constraints
- Derivation of summaries
Communication autonomy
- Decide whether to communicate with another component DBS

Execution autonomy
- Execute local operations without interference from external operations by other component DBSs
- Ability to decide order of execution

Association autonomy
- Decide whether and how much to share its functionality and resources
23.7 Distributed Database Architectures

- Parallel versus distributed architectures
- Types of multiprocessor system architectures
  - Shared memory (tightly coupled)
  - Shared disk (loosely coupled)
  - Shared-nothing
Database System Architectures

Figure 23.7 Some different database system architectures (a) Shared-nothing architecture (b) A networked architecture with a centralized database at one of the sites (c) A truly distributed database architecture
General Architecture of Pure Distributed Databases

- Global query compiler
  - References global conceptual schema from the global system catalog to verify and impose defined constraints

- Global query optimizer
  - Generates optimized local queries from global queries

- Global transaction manager
  - Coordinates the execution across multiple sites with the local transaction managers
Schema Architecture of Distributed Databases

Figure 23.8 Schema architecture of distributed databases
Figure 23.9 The five-level schema architecture in a federated database system (FDBS)
An Overview of Three-Tier Client/Server Architecture

- Division of DBMS functionality among the three tiers can vary

Figure 23.10 The three-tier client/server architecture
23.8 Distributed Catalog Management

- **Centralized catalogs**
  - Entire catalog is stored at one single site
  - Easy to implement

- **Fully replicated catalogs**
  - Identical copies of the complete catalog are present at each site
  - Results in faster reads

- **Partially replicated catalogs**
  - Each site maintains complete catalog information on data stored locally at that site
23.9 Summary

- Distributed database concept
- Distribution transparency
- Fragmentation transparency
- Replication transparency
- Design issues
  - Horizontal and vertical fragmentation
- Concurrency control and recovery techniques
- Query processing
- Categorization of DDBMSs