

Databases Illuminated

Chapter 12

Distributed Databases

Distributed Database System

- Multiple sites connected by a communications system
- Data at any site available to users at other sites
- Sites may be far apart; linked by telecommunications lines
- May be close together; linked by a local area network

Advantages of Distribution

- Compared to a single, centralized system that provides remote access, distributed system advantages are
 - Local autonomy
 - Improved reliability
 - Better data availability
 - Increased performance
 - Reduced response time
 - Lower communications costs

Architecture Design Considerations

- Factors the designer considers in choosing an architecture for a distributed system
 - Type of communications system
 - Data models supported
 - Types of applications
 - Data placement alternatives

Architecture Alternatives

- Centralized database with distributed processing
- Client-server system
- Parallel databases
 - Shared memory
 - Shared disk
 - Shared nothing
 - Cluster
- true distributed database-data and processing shared among autonomous sites

Types of Distributed Systems

- **Homogeneous**

- All nodes use the same hardware and software

- **Heterogeneous**

- Nodes have different hardware or software
- Require translations of codes and word lengths due to hardware differences
- Translation of data models and data structures due to software differences

Software Components of DDBMS

- Data communications component (DC)
- Local database management system (DBMS)
- Global data dictionary (GDD)
- Distributed database management system component (DDBMS)
- Not all sites have all these components

DDBMS Functions

- Provides the user interface
 - Needed for location transparency
- Locates the data
 - Directs queries to proper site(s)
- Processes queries
 - Local, remote, compound (global)
- Provides network-wide concurrency control and recovery procedures
- Provides translation in heterogeneous systems

Data Placement Alternatives

- Centralized
 - All data at one site only
- Replicated
 - All data duplicated at all sites
- Partitioned
 - Data divided among sites
 - Fragmentation scheme: horizontal, vertical, mixed fragments
 - Each item appears only once
- Hybrid
 - Combination of the others

Factors in Data Placement Decision

- Locality of reference
- Reliability of data
- Availability of data
- Storage capacities and costs
- Distribution of processing load
- Communications costs

Types of Transparency

- Data distribution transparency
 - Fragmentation transparency
 - Location transparency
 - Replication transparency
- DBMS heterogeneity transparency
- Transaction transparency
 - Concurrency transparency
 - Recovery transparency
- Performance transparency

Transaction Management for DDBMS

- Each site that initiates transactions has a **transaction coordinator** to manage transactions that originate there
 - For local or remote transactions, transaction manager at data site takes over
 - For global transactions, originating site coordinator
 - Starts execution
 - Uses GDD to form sub-transactions
 - Directs sub-transactions to appropriate sites
 - Receives sub-transaction results
 - Controls transaction end-either commit or abort at all sites
- Additional concurrency control problem
 - **Multiple-copy inconsistency** problem
- Solutions use **locking** and **timestamping**

Locking Protocols

- Extension of two-phase locking protocol
 - Single-site lock manager
 - May use Read-One-Write-All replica handling
 - Site may become a bottleneck
 - Distributed lock manager
 - Can use Read-One-Write-All method
 - Deadlock difficult to determine
 - Primary copy
 - Dominant node for each data item
 - Majority locking

Global Deadlock Detection

- Each site has local wait-for graph-detects only local deadlock
- Need global wait-for graph
 - Single site can be global deadlock detection coordinator
 - Constructs global graph and checks for cycles
 - Responsibility could be shared among sites

Timestamping Protocols

- One site could issue all timestamps
- Instead, multiple sites could issue them
 - Each timestamp has two parts-the time and the node identifier
 - Guarantees uniqueness of timestamps
 - Difficult to synchronize clocks-to control divergence, can advance clock reading if later timestamp received
 - Can apply basic timestamping, Thomas' Write Rule, multi-version timestamp protocols using unique timestamps

Recovery-Failures

- Must guarantee atomicity and durability of transactions
- Failures include usual types, plus loss of messages, site failure, link failure
- Network partitioning
 - Failure where network splits into groups of nodes that are isolated from other groups, but can communicate with one another

Handling Node Failure

- System flags node as failed
- System aborts and rolls back affected transactions
- System checks periodically to see if node has recovered, or node self-reports
- After restart, failed node does local recovery
- Failed node catches up to current state of DB, using system log of changes made while it was unavailable

Commit Protocols

- Two-phase commit protocol
 - Phase 1-voting phase
 - Coordinator writes <begin commit T> to its log, writes log to disk, sends <prepare T> msg to all participants. Each site either does a <ready T> or <abort T> and sends its vote to coordinator
 - Phase 2-resolution phase
 - Coordinator resolves fate of transaction
 - If any abort msg received, makes all sites abort
 - Failure of any site to vote generates global abort
 - If all sites voted ready, coordinator tells all to commit
 - Handles various types of failure
- Three-phase commit protocol – non-blocking, assumes no more than k sites fail, for some k

Distributed Query Processing

- Queries can be
 - local: done at originating site only
 - Remote: done at different site
 - Compound: requires multiple sites
- Must consider cost of transferring data between sites
- Semijoin operation is sometimes used when a join of data stored at different sites is required

Steps in Distributed Query-1

1. Accept user's request
2. Check request's validity
3. Check user's authorization
4. Map external to logical level
5. Determine request processing strategy
6. For heterogeneous system, translate query to DML of data node(s)
7. Encrypt request
8. Determine routing (job of DC component)

Steps in Distributed Query-2

9. Transmit messages (job of DC component)
10. Each node decrypts its request
11. Do update synchronization
12. Each data node's local DBMS does its processing
13. Nodes translate data, if heterogeneous system
14. Nodes send results to requesting node or other destination node
15. Consolidate results, edit, format results
16. Return results to user