### **Relational Databases & ACID Properties**

* **Relational Model Benefits:**
  + Standard data model & query language
  + ACID compliance (Atomicity, Consistency, Isolation, Durability)
  + Handles structured data efficiently
  + Supports indexing, query optimization, and replication
* **Transaction Processing:**
  + CRUD operations performed as a single unit (COMMIT or ROLLBACK)
  + Ensures data integrity, error recovery, and concurrency control
* **ACID Properties:**
  + **Atomicity:** Transactions execute fully or not at all
  + **Consistency:** Database remains in a valid state after a transaction
  + **Isolation:** Prevents interference between concurrent transactions
    - Dirty Reads: Uncommitted changes are read by another transaction
    - Non-repeatable Reads: Data changes between two reads in a transaction
    - Phantom Reads: New rows appear due to another transaction
  + **Durability:** Committed transactions are permanent even after system failure
* **SQL Transaction Example (Money Transfer):**
  + Uses START TRANSACTION, checks balance, and either commits or rolls back

### **Limitations of Relational Databases**

* Schema rigidity—may not adapt to evolving applications
* Expensive joins and query execution
* Difficulty handling semi-structured/unstructured data (JSON, XML)
* Horizontal scaling is challenging

### **Scaling & Distributed Systems**

* **Scaling Strategies:**
  + **Vertical Scaling:** Upgrading hardware (simpler but costly)
  + **Horizontal Scaling:** Distributing data across multiple machines
  + Distributed databases make horizontal scaling more manageable
* **Distributed System Characteristics:**
  + Multiple computers function together as one
  + Failures occur independently
  + No shared global clock
* **Distributed Storage Models:**
  + Data replication across multiple nodes
  + Relational & NoSQL databases support replication and sharding
  + Examples: MySQL, PostgreSQL, CockroachDB

### **CAP Theorem (Consistency, Availability, Partition Tolerance)**

* **Consistency:** Every read gets the latest write or an error
* **Availability:** Every request gets a response (but may be outdated)
* **Partition Tolerance:** System continues operating despite network failures
* **Tradeoffs:**
  + **CP (Consistency + Partition Tolerance):** Latest data but may drop requests
  + **AP (Availability + Partition Tolerance):** Always responds but may return stale data
  + **CA (Consistency + Availability):** No tolerance to network failures