

# Advanced Transaction Processing

Ram Datta Bhatta

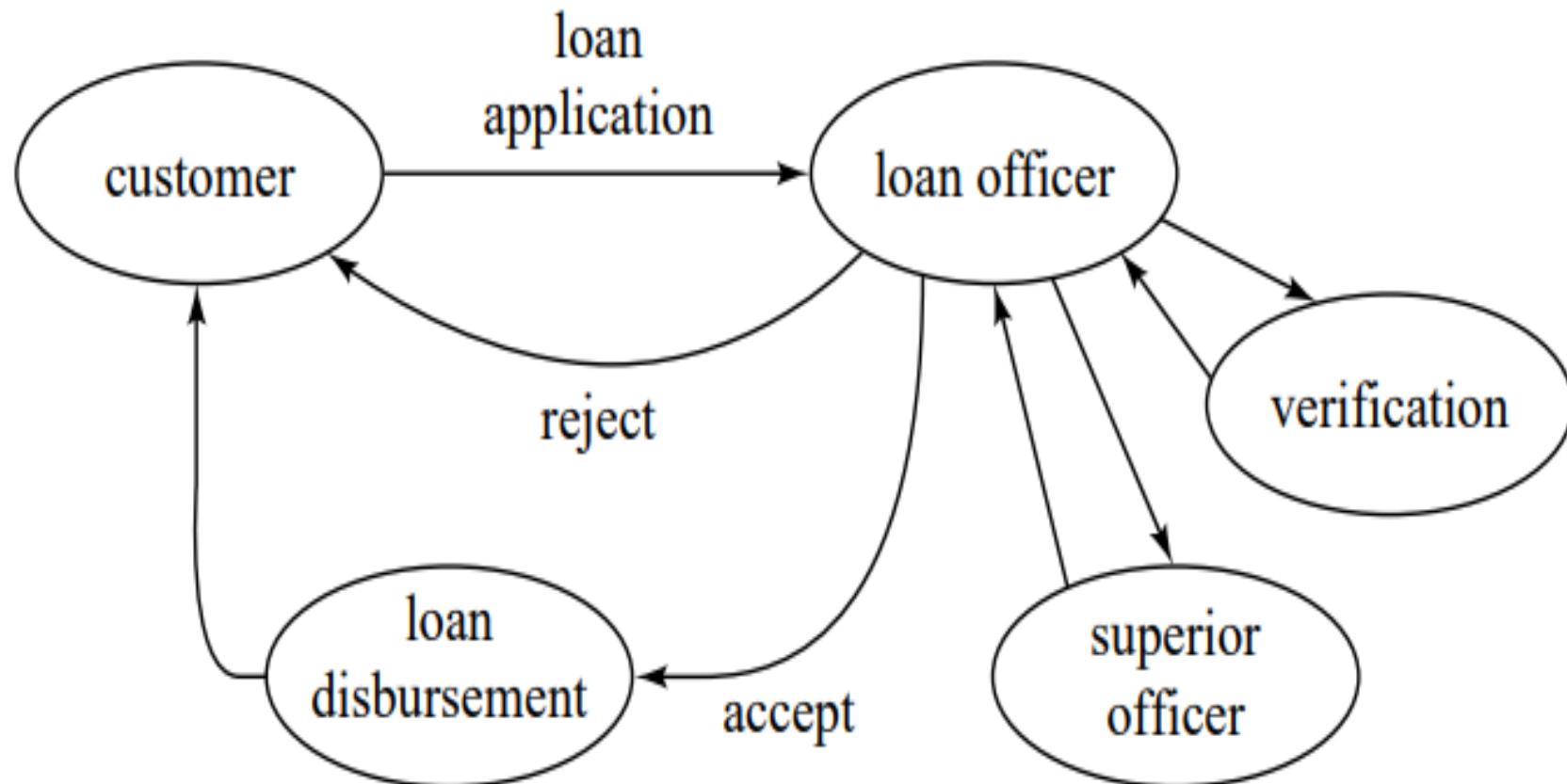
# Advanced Transaction Processing (ATP)

- ✓ The techniques and features that enhance the processing and management of database transactions in complex and demanding environments
- ✓ It is designed to handle complex and demanding transactional scenarios with optimal performance, concurrency control, data consistency, and fault tolerance.
- ✓ Offers a comprehensive set of techniques, mechanisms, and optimizations that ensure efficient execution of transactions in various challenging environments, including high-concurrency, distributed, and mission-critical scenarios

# Transactional workflows

- ✓ A sequence of interrelated and coordinated steps or activities that are performed as part of a larger transaction.
- ✓ A transactional workflow represents a set of tasks, actions, or operations that need to be executed in a specific order to achieve a particular goal or outcome.
- ✓ Workflows are activities that involve the coordinated execution of multiple tasks performed by different processing entities.
- ✓ With the growth of networks, and the existence of multiple autonomous database systems, workflows provide a convenient way of carrying out tasks that involve multiple systems.
- ✓ Example of a workflow: delivery of an email message, which goes through several mailer systems to reach destination.
  - Each mailer performs a task: forwarding of the mail to the next mailer
  - If a mailer cannot deliver mail, failure must be handled semantically (delivery failure message)

# Loan Processing Workflow- Example



# Transactional Workflow

- ✓ Must address following issues to computerize a workflow:
  - **Specification of workflows:** detailing the tasks that must be carried out and defining the execution requirements.
  - **Execution of workflows:** execute transactions specified in the workflow while also providing traditional database safeguards related to the correctness of computations, data integrity, and durability.  
E.g.: Loan application should not get lost even if system fails
- ✓ State of a workflow – consists of the collection of states of its constituent tasks, and the states (i.e., values) of all variables in the execution plan

# E-Commerce

- ✓ the buying and selling of goods and services over the internet.
- ✓ involves the electronic exchange of products, services, information, and payments between businesses, consumers, and other entities.
- ✓ commercial transactions conducted over the internet” or internet commerce

# How does it work?



# Types of E-Commerce

- ✓ **Business-to-Consumer (B2C):** In B2C e-commerce, businesses sell products or services directly to individual consumers. Examples include online retailers like Amazon, Walmart, and fashion brands' websites.
- ✓ **Business-to-Business (B2B):** B2B e-commerce involves transactions between businesses. Companies purchase products, services, or raw materials from other businesses to support their operations. Examples include suppliers providing parts to manufacturers or software companies offering services to other businesses.
- ✓ **Consumer-to-Consumer (C2C):** C2C e-commerce facilitates transactions between individual consumers. Online marketplaces or platforms enable individuals to sell products to each other. Popular examples include platforms like eBay and Craigslist, where individuals can list items for sale to other consumers.

# Types.....

- ✓ Business-to-Government (B2G): B2G e-commerce involves businesses providing products or services to government entities. This can include contracts for supplies, services, or even technology solutions.
- ✓ Government-to-Business (G2B): In G2B e-commerce, government agencies offer products or services to businesses. This might include licenses, permits, or other regulatory requirements.
- ✓ Government-to-Consumer (G2C): G2C e-commerce involves government agencies providing services, information, or transactions directly to consumers. This could include online tax filing, applying for government benefits, or obtaining official documents.

# E-Business Vs E-Commerce



e-business encompasses all the business services and activities operated utilising the web

# Real Time Transaction

- ✓ A type of transaction that is processed instantly or nearly instantly, with minimal delay between the initiation of the transaction and its completion.
- ✓ A real-time transactions offer immediate feedback and outcomes, providing users with timely and up-to-date information.



Debit card transactions must run through the debit card network, issuing bank, an authorization process from a PIN or require a signature. This is all done in just a few seconds. In terms of payment processing, debit cards are closer to accepting a check than a credit card.

# Key characteristics of real-time transactions

- ✓ **Instant Processing:** Real-time transactions are executed and processed immediately upon initiation, without any noticeable delay
- ✓ **Immediate Confirmation:** Users receive immediate confirmation of the transaction's success or failure.
- ✓ **Data Consistency:** Real-time transactions ensure that data is consistent across different systems and databases immediately after the transaction is completed.
- ✓ **Low Latency:** Real-time transactions are designed to have minimal latency or delay, allowing users to experience smooth and rapid interactions.

It is used in financial transactions, e-commerce, supply chain, health care, gaming and entertainment etc.

Design of a real-time system involves ensuring that enough processing power exists to meet deadlines without requiring excessive hardware resources

# Long Duration Transaction

- ✓ A long duration transaction also known as Batch transaction requires a longer execution/response time and generally accesses larger portion of the database.
- ✓ It involves human intervention, while short duration transactions are more or less Non-Interactive.
- ✓ [https://www.cukashmir.ac.in/cukashmir/User\\_Files/imagefile/DIT/StudyMaterial/AdvancedDBMS/Advanced%20Transaction%20Processing2.pdf](https://www.cukashmir.ac.in/cukashmir/User_Files/imagefile/DIT/StudyMaterial/AdvancedDBMS/Advanced%20Transaction%20Processing2.pdf)

# Characteristics

- ✓ **Long Duration:** When interacting with Humans, it is quite natural the response will be very slow relative to computer speed. In some applications, the human activity may involve hours, days, even longer periods of time. So overall the transactions will be of longer duration.
- ✓ **Exposure to Uncommitted Data:** In many cases of long duration transactions, the other transactions may be forced to read uncommitted data. If several users are cooperating on a project, user transactions may need to exchange data prior to transaction commit.

# Characteristics..

- ✓ **Subtasks:** Interactive transaction will consist of set of subtasks initiated by the user. At some point of time, user may wish to abort a subtask, without necessarily causing the entire transaction to Abort.
- ✓ **Recoverability:** It is un-acceptable to abort a Long-duration interactive transaction given a system crash. Even if we have a system crash, the active transaction must be recovered to a state that existed shortly before the crash, so that relatively human work is lost.
- ✓ **Performance:** Good performance in Long Duration Transaction is defined as how fast the response has been generated for a particular transaction. However, in case of non-interactive system, performance is measured as HIGH THROUGHPUT

# Concurrency problem in Long Duration Transaction

Given the properties of Long Duration Transactions, various Concurrency protocols are very difficult to implement:

**2 Phase Locks:** If a lock isn't granted, the transaction requesting the lock is forced to wait for the data item to be unlocked. If the transaction holding the lock on data item happens to be a Long Duration Transaction, Response Time increases leading to increased chances of Deadlock.

**Time-Stamp-Based Protocols:** Although timestamp based protocols never force a transaction to wait; however they require transaction to abort under certain circumstances. If a long duration transaction is aborted a substantial amount of work is lost.

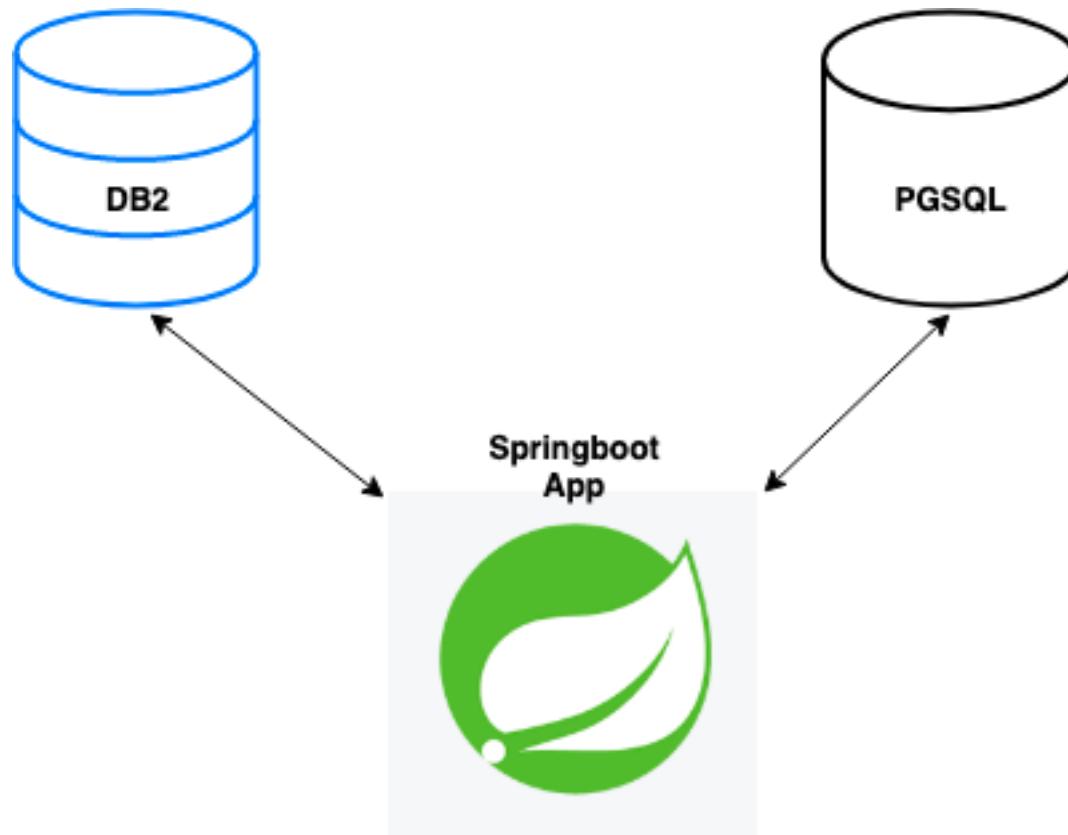
Snapshot Isolation is a solution for achieving Concurrency when dealing with Long Duration Transactions. It works as follow:

- **Read Transactions and Snapshot:** When a transaction begins, it takes a snapshot of the database as it existed at the start of the transaction.
- **Read Operations:** Throughout the transaction's duration, any read operations performed by the transaction retrieve data from the snapshot rather than directly from the current database state. This ensures that the transaction's view remains consistent throughout its execution.
- **Write Operations and Versioning:** When the transaction performs write operations (e.g., inserts, updates, deletes), the DBMS creates new versions of the affected data items rather than modifying the existing data directly. These new versions are associated with the transaction and are not visible to other transactions until the current transaction is committed.
- **Isolation:** Snapshot Isolation ensures that while a transaction is executing, it operates in isolation from other concurrently executing transactions. Other transactions see the database state as it existed at the start of their respective transactions.
- **Commit:** When the long-duration transaction is ready to commit, the DBMS checks if any conflicts have arisen due to the changes made by the transaction. If conflicts are detected, the transaction may need to be rolled back or rescheduled to maintain data consistency.

# Transaction Management in Multidatabase

- ✓ A multidatabase system (MDBS) is a facility that allows users access to data located in multiple autonomous database management systems (DBMSs). It integrates multiple separate databases into a unified environment, allowing users and applications to access and manipulate data across different database management systems (DBMS) as if they were part of a single database.
- ✓ Transaction management in multidatabase systems refers to the techniques and mechanisms used to ensure the proper execution, consistency, isolation, and durability of transactions that span multiple independent and distributed databases.
- ✓ Transaction management in multidatabase systems involves addressing challenges that arise due to the distributed and heterogeneous nature of the participating databases

# Transactions for multiple databases



Some strategies and techniques for effective transaction management in a multidatabase system:

✓ **Global Transaction Coordinator (GTC):**

Implement a central component, the Global Transaction Coordinator, responsible for coordinating and managing distributed transactions across multiple databases. The GTC tracks the progress of transactions, communicates with local transaction managers in each database, and ensures that all participating databases reach a consensus to commit or abort the transaction

✓ **Isolation and Concurrency Control:**

Implement appropriate isolation levels (e.g., Serializable, Repeatable Read) to control the visibility of data changes during transaction execution. Also, utilize concurrency control mechanisms (e.g., locking, timestamp ordering) to manage concurrent access to data across participating databases

## ✓ **Deadlock Detection and Resolution:**

Deploy deadlock detection and resolution mechanisms to handle potential deadlocks that may arise due to concurrent transactions accessing shared resources.

## ✓ **Transaction Recovery and Durability:**

Implement logging and transaction history mechanisms to ensure transaction durability and support recovery from failures. Maintain transaction logs to record changes made by transactions, allowing for recovery and rollback if necessary.

## ✓ **Heterogeneous Data Models:**

Accommodate different data models by providing a uniform interface and translation layer to ensure that transactions can interact with databases of varying data types.

## ✓ **Consistency and Integrity Constraints:**

Enforce data consistency and integrity constraints across participating databases to maintain the integrity of the distributed data

[https://dl.acm.org/doi/pdf/10.1007/BF01231700#:~:text=A%20multidatabase%20system%20\(MDBS\)%20is,control%20of%20the%20local%20DBMSs.](https://dl.acm.org/doi/pdf/10.1007/BF01231700#:~:text=A%20multidatabase%20system%20(MDBS)%20is,control%20of%20the%20local%20DBMSs.)

**Thank you.**