United States
International
University-Africa

DATABASE SYSTEMS

Presented by Prof. Elisha T. O. Omulo

WEEK 9 AGENDA

Transaction Processing Concepts.
Properties of Transactions.
Schedules and Recoverability.
Serializability of Schedules.
Locking Techniques for Concurrency Control.
Types of Locks.
Deadlock Handling.

Course Textbook: **Carlos Coronel, Steven Morris, Peter Rob and Keeley Crockett Database Principles: Fundamentals of Design, Implementation, and Management, 14th Edition, 2022, ISBN-13978-0357673034.**

Transaction Processing Concepts

Transaction

- A transaction is a logical unit of work; that is, it must be entirely completed or aborted—no intermediate ending states are accepted.
- It is realized as a sequence of database requests that access the database.
- All transactions must have the properties of atomicity, consistency, isolation, and durability (Course text).

Transaction processing- all operations and activities undertaken so that transactions are performed subject to the desired propeties - atomicity, consistency, isolation, and durability.

Task: Discuss the notion of transaction.

Transaction Processing Concepts

Basic operations

- read_item() − reads data item from storage to main memory.
- modify_item() − change value of item in the main memory.
- write_item() − write the modified value from main memory to storage.

Transaction Operations-Low level steps:

- begin_transaction − specify start of transaction execution.
- read_item or write_item − Database operations that may be interleaved with main memory operations as a part of transaction.
- end_transaction − A marker that specifies end of transaction.
- commit − specify success of the transaction in its entirety and will not be undone.
- rollback − specify as unsuccessful and all temporary changes in the database are undone. A committed transaction cannot be rolled back.

Task: Describe transaction operations.

Transaction Processing Concepts

Transaction States

There are five states, active, partially committed, committed, failed and aborted.

Active – starts with active state, and it may read, write or do other operations.

Partially Committed – this state occurs after the last statement of the transaction.

Committed – occurs after completion of the transaction and commit signal issued.

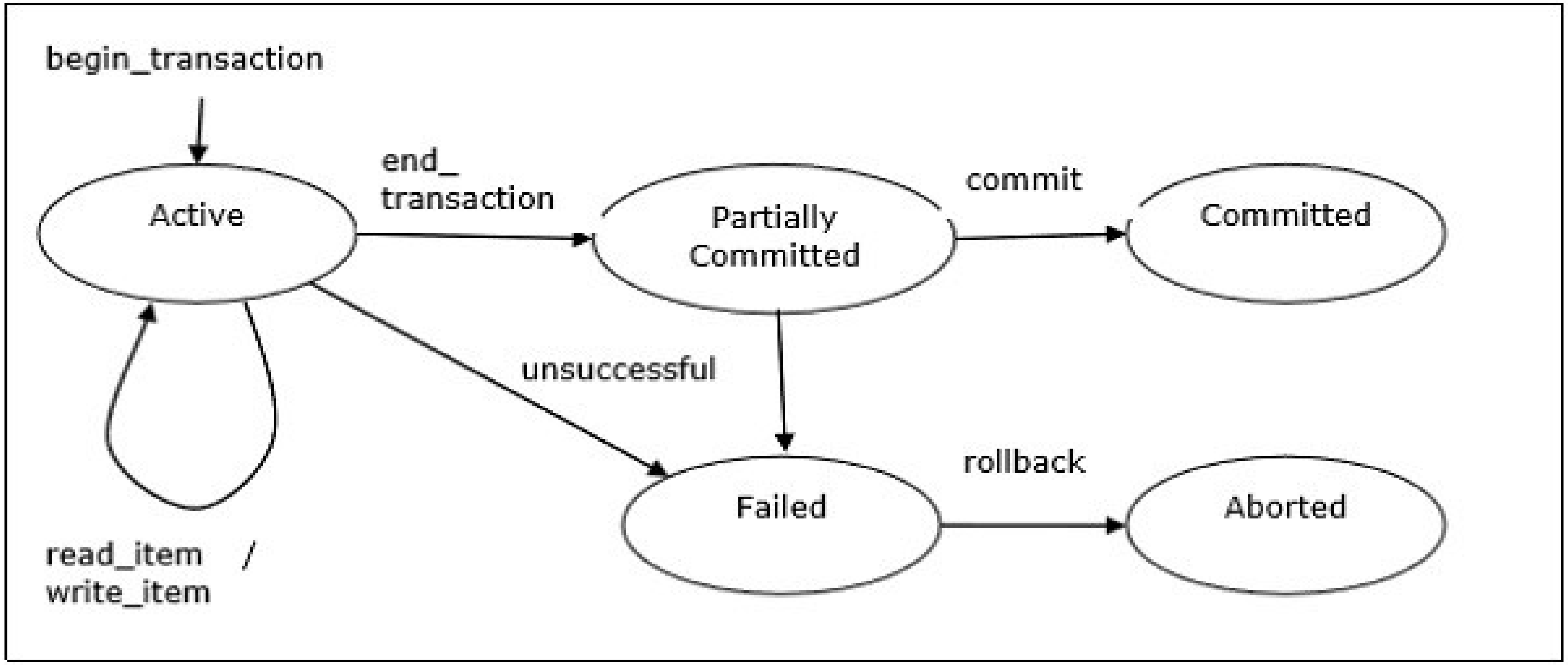
Failed – occurs when normal execution can no longer proceed or system checks fail.

Aborted – the transaction has been rolled back after failure and the database has been restored to its state that was before the transaction began.

Task: Discuss the transaction states

Transaction Processing Concepts

Transaction States



Source: https://www.tutorialspoint.com/distributed_dbms/distributed_dbms_transaction_processing_systems.htm

Task: Discuss the transaction states.

Properties of Transactions

Desired properties of transactions

They must maintain the ACID properties, viz. Atomicity, Consistency, Isolation, and Durability.

Atomicity − it is either it is performed in its entirety or not performed at all. No partial update should exist.

Consistency − it should take the database from one consistent state to another consistent state. It should not adversely affect any data item in the database.

Isolation − it should be executed as if it is the only one in the system. There should not be any interference from the other concurrent transactions that are simultaneously running.

Durability − If a committed transaction brings about a change, that change should be durable in the database and not lost in case of any failure.

Task: Describe the desired properties of transactions.

USIU AFRICA

Schedules and Recoverability.

Schedule

- A schedule S, is the total order of execution of operations.
- Let a schedule S have n transactions, say T_1, T_2, T_3...T_n; any transaction T_i has the operations that execute as laid down in the schedule S.
- It is a sequence of the operations by a set of concurrent transactions that preserves the order of the operations in each of the individual transactions.

Types of Schedules

There are two types of schedules:

Serial Schedules –

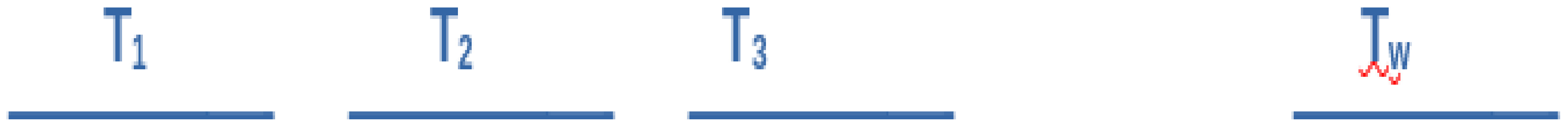
- A schedule where the operations of each transaction are executed consecutively without any interleaved operations from other transactions.
- At any point of time, only one transaction is active, i.e. there is no overlapping of transactions.

Task: Discuss the types of schedules.

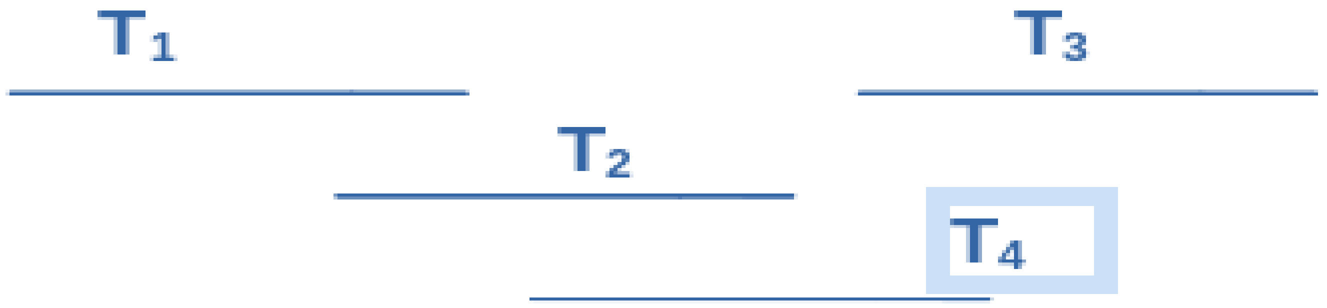
Schedules and Recoverability.

Types of Schedules

Serial Schedules – **at any point of time, only one transaction is active, i.e. there is no overlapping of transactions.**



Parallel Schedules –more than one transactions are active simultaneously, i.e. the transactions contain operations that overlap at time.



Task: Discuss the types of schedules.

Schedules and Recoverability.

Conflicts in Schedules- several transactions occur

A conflict occurs when two active transactions perform non-compatible operations.

Conflict operations occur if:

Two operations are parts of different transactions.
The operations access the same data item.
One of the operations is a write_item() trying to modify the data item.

Recoverable schedule

A schedule where, for each pair of transactions T_i and T_j, if T_j reads a data item previously written by T_i , then the commit operation of T_i precedes the commit operation of T_j.

Task: Discuss the conflicts of operations and recoverability

Serializability of Schedules

- A serializable schedule of ‘n’ transactions is a parallel schedule which is equivalent to a serial schedule comprising of the same ‘n’ transactions.
- A serializable schedule contains the correctness of serial schedule while ascertaining better CPU utilization of parallel schedule.

Equivalence of Schedules

Result equivalence – producing identical results are said to be result equivalent.

View equivalence – that perform similar action in a similar manner are said to be view equivalent.

Conflict equivalence – contain the same set of transactions and have the same order of conflicting pairs of operations.

Task: Describe serializability of schedules.

USIU AFRICA

Serializability of Schedules

Locking Based Concurrency Control Protocols

A lock is a variable associated with a data item that determines whether read/write operations can be performed on that data item.

One-phase Locking Protocol

- **Each transaction locks an item before use and releases the lock as soon as it has finished using it.**
- **This locking method provides for maximum concurrency but does not always enforce serializability.**

Task: Describe one phase locking protocol.

USIU AFRICA

Serializability of Schedules

Locking Based Concurrency Control Protocols

Two-phase Locking Protocol

- **All locking operations precede the first lock-release or unlock operation.**
- **The two phases:**
    - **First phase(growing): a transaction only acquires all the locks it needs and does not release any lock. Also called the expanding or the growing phase.**
    - **Second phase(shrinking): the transaction releases the locks and cannot request any new locks. Also called the shrinking phase.**
- **Every transaction that follows two-phase locking protocol is guaranteed to be serializable. This approach gives low parallelism between two conflicting transactions.**

Task: Describe how to realize serializability.

Locking Techniques for Concurrency Control

Concurrency control methods allow several transactions to be executed simultaneously while maintaining the ACID properties of the transactions and serializability in the schedules.

Locking

- **A procedure used to control concurrent access to data. When one**
- **transaction is accessing the database, a lock may deny access to other transactions to prevent incorrect results.**
- **A lock guarantees exclusive use of a data item to a current transaction.**

Task: Describe the locks

Types of Locks

Shared lock

If a transaction has a shared lock on a data item, it can read the item but not update it.

Exclusive lock

If a transaction has an exclusive lock on a data item, it can both read and update the item.

Task: Describe the different types of locks.

Types of Locks

- Read operations cannot conflict so, it is permissible for more than one transaction to hold shared locks simultaneously on the same item.
- Exclusive lock gives a transaction exclusive access to that item.
- Thus, as long as a transaction holds the exclusive lock on the item, no other transactions can read or update that data item.

How locks are used

- A transaction that needs access to data item first locks the item, requests a shared lock for read only access or an exclusive lock for both read and write access.
- If the item is not locked by another transaction, the lock will be granted.
- If the item is currently locked, the DBMS checks compatibility with the existing lock; a shared lock request on a shared lock is granted; otherwise, the transaction must wait until the existing lock is released.

Task: Describe the locks are used.

USIU AFRICA

Types of Locks

How locks are used

- A transaction holds a lock until it explicitly releases it either during execution or when it terminates (aborts or commits).

- It is only when the exclusive lock has been released that the effects of the write operation will be made visible to other transactions.

- Some systems allow a transaction to issue a shared lock on an item and then later to upgrade the lock to an exclusive lock. The transaction can examine the data first and then decide whether it wishes to update its lock.

Task: Describe how the locks are used.

USIU AFRICA

Types of Locks

Lock granularity

Database-level lock: the entire database is locked, preventing the use of any tables in the database by transaction T2 while transaction T1 is being executed. This good for batch processes, but it is unsuitable for multiuser DBMSs as it is s-l-o-w.

Table-level lock: the entire table is locked, blocking access to any row by transaction T2 while transaction T1 is using the table. If a transaction requires access to several tables, each table may be locked. However, two transactions can access the same database as long as they access different tables. It causes traffic jams when many transactions are waiting to access the same table. They are not suitable for multiuser DBMSs.

Task: Describe how the locks are used.

Types of Locks

Lock granularity

Page-level lock:

- Involve entire diskpage. A diskpage, or page, is the equivalent of a diskblock, which can be described as a directly addressable section of a disk. A page has a fixed size, such as 4K, 8K, or 16K. For example, if you want to write only 100 bytes to a 4K page, the entire 4K page must be read from disk, updated in memory, and written back to disk.
- A table can span several pages, and a page can contain several rows of one or more tables.
- Page-level locks are currently the most frequently used locking method for multiuser DBMSs.

Task: Describe how the lpage level ocks.

USIU AFRICA

Types of Locks

Lock granularity

Row-level lock:

- Much less restrictive than the locks discussed earlier. The DBMS allows concurrent transactions to access different rows of the same table even when the rows are located on the same page.
- It has a high overhead as a lock exists for each row in a table of the database involved in a conflicting transaction.
- Modern DBMSs automatically escalate a lock from a row level to a page level when the application session requests multiple locks on the same page.

Task: Describe how the row level locks.

Types of Locks

Lock granularity

Field-level lock

- Allows concurrent transactions to access the same row as long as they require the use of different fields (attributes) within that row.
- Yields the most flexible multiuser data access,
- Rarely implemented in a DBMS because it requires an extremely high level of computer overhead and because the row-level lock is much more useful in practice.

Task: Describe how the row level locks.

Deadlock Handling

Deadlock

- An impasse that may result when two (or more) transactions are each waiting for locks to be released that are held by the other.

- To break a deadlock: abort one or more of the transactions.
- Undo all the changes made by the aborted transaction(s).
- Once this is complete,the locks held by transaction are released and other transaction can continue again.
- Deadlock handling should be transparent to the user, so the DBMS should automatically restart the aborted transaction(s).

There are deadlock prevention methods that are left as an exercise.

Task: Investigate the deadlock prevention methods.

USIU AFRICA

Week 9 exercises

1)Discuss Transaction Processing Concepts.
2)Desirable Properties of Transactions.
3)Describe Schedules and Recoverability.
4)Describe Serializability of Schedules.
5)Describe Locking Techniques for Concurrency Control.
6)Describe Types of Locks.
7)Discuss Deadlock Handling.

Week 9 Session References

- **[Course Text] Carlos Coronel, Steven Morris, Peter Rob and Keeley Crockett Database Principles: Fundamentals of Design, Implementation, and Management, 14th Edition, 2022, ISBN-13978-0357673034.**

- **Thomas M. Connolly, Carolyn E. Begg (2021). Database Systems: A Practical Approach to Design, Implementation, and Management. Published by Pearson (July 14th 2021). ISBN-13: 9780137517053**

Thank You