COURSE NAME: DBMS
COURSE CODE:23AD2102A

TOPIC:

RECOVERY CONCEPTS

Session - 7

AIM OF THE SESSION

To familiarize students with the basic concept of Recovery in DBMS.

INSTRUCTIONAL OBJECTIVES

This Session is designed to:

Their applications in ensuring data consistency, durability, and availability.

LEARNING OUTCOMES

At the end of this session, you should be able to:

- Gain knowledge about the different types of transaction failures and how they can be recovered.

RECOVERY CONCEPTS

Definition of recovery concepts

- Recovery concepts in DBMS refer to a set of techniques used to ensure that a database can recover from various types of failures, such as hardware failures, software errors, and other system crashes.
- To ensure that the database remains consistent and durable, and that all transactions are processed correctly.

Importance of recovery concepts

- This will ensure that the database can recover from various types of failures and that it can do so quickly and reliably.
- They help to ensure that the database remains available to users and that it can continue to operate even in the event of a failure.

RECOVERY CONCEPTS

There are several key recovery concepts in DBMS,

- Logging
- Checkpointing
- Redo and Undo
- Recovery manager, and
- Transaction management.

RECOVERY CONCEPTS

Logging

Definition:

- Logging is a recovery technique used in DBMS to record all modifications made to a database during a transaction**.**
- It’s a sequential record of all database modifications.
- Used to reconstruct the state of the database in the event of a failure.
- The log is typically stored on a separate disk from the database to ensure that it is not lost in the event of a disk failure.

Importance of logging in recovery

- Without proper logging, a DBMS cannot recover from various types of failures, such as system crashes and power failures.
- It identify which changes were made to the database but were not written to disk, and which changes were made to the database and were written to disk.
- By reading the log and replaying the changes that were not written to disk, the DBMS can ensure that the database is consistent and durable.

RECOVERY CONCEPTS

There are several types of logs that can be used in DBMS, including:

- **Transaction log:** This log records all modifications made to a database during a transaction, including updates, inserts, and deletes.
- **System log:** This log records all system-level events, such as system startup and shutdown, database backup and recovery, and other administrative tasks.
- **Archive log:** This log contains a historical record of all transactions processed by the DBMS.
- **Redo log:** This log contains a record of all changes made to the database that have not been written to disk.

RECOVERY CONCEPTS

EXAMPLE:

- Timestamp: 2023-05-05 14:30:00 Transaction ID: 1234 Operation: Update Table: Employees Record ID: 5678 New Value: {Name: John Doe, Salary: $50,000}
- This log record indicates that at 2:30 pm on May 5, 2023, transaction ID 1234 updated the record for employee ID 5678 in the Employees table, changing the employee's name to John Doe and salary to $50,000.
- By replaying this log record in the event of a failure, the DBMS can ensure that the database remains consistent and durable.

Check Pointing

Definition:

- Checkpointing is a recovery technique used in DBMS.
- To reduce the amount of work needed to recover the database after a failure.
- It involves writing a copy of the database to disk at certain intervals, called checkpoints, to provide a stable starting point for recovery.
- Checkpointing is typically performed in combination with logging, where the checkpoint record is also logged to ensure durability.

RECOVERY CONCEPTS

Importance of checkpointing in recovery

- Without proper checkpointing, the recovery process can be slow and time-consuming, requiring the DBMS to replay all transactions since the last checkpoint.
- By reducing the amount of work required to recover the database, checkpointing can help to minimize downtime and ensure that the database remains available to users.

RECOVERY CONCEPTS

There are several types of checkpoints that can be used in DBMS, including:

- **Time-based checkpoints:** These checkpoints are triggered at regular intervals, such as every hour or every day.
- **Threshold-based checkpoints:** These checkpoints are triggered when a certain amount of log data has been generated since the last checkpoint.
- **Hybrid checkpoints:** These checkpoints are triggered based on a combination of time and threshold-based criteria.

RECOVERY CONCEPTS

An example of a checkpoint :

- Timestamp: 2023-05-05 23:00:00 Checkpoint ID: 1234 Log sequence number: 5678 Database size: 1 GB Status: Completed
- This checkpoint record indicates that a checkpoint was taken at 11 pm on May 5, 2023, with ID 1234. The checkpoint includes all log records up to sequence number 5678 and a database size of 1 GB.
- The status of the checkpoint is marked as completed, indicating that it was successfully written to disk.
- By using this checkpoint as a starting point for recovery, the DBMS can minimize the amount of work required to recover the database.

RECOVERY CONCEPTS

Redo:

Definition:

- Redo and Undo are two important recovery techniques used in DBMS to ensure database consistency and durability in the event of a failure.
- Redo refers to the process of reapplying changes made to the database from the log during the recovery process.
- Redo ensures that all committed changes are applied to the database, even if they were not written to disk at the time of the failure.

RECOVERY CONCEPTS

Undo:

Definition:

- Undo refers to the process of reversing changes made to the database during an incomplete transaction.
- Undo ensures that changes made by transactions that were not yet committed at the time of the failure are removed from the database, leaving the database in a consistent state.

RECOVERY CONCEPTS

Importance of Redo and Undo:

- The importance of redo and undo in recovery cannot be overstated. These techniques ensure that the database remains consistent and durable, even in the event of a failure.
- By applying redo changes and undoing uncommitted changes, the DBMS can bring the database back to a consistent state.
- Redo aims to apply committed changes to the database, while undo aims to remove uncommitted changes from the database.
- Redo is used during recovery to apply changes made since the last checkpoint or backup.
- Undo is used to rollback changes made by transactions that were not yet committed at the time of the failure.

An example of a redo operation :

Timestamp: 2023-05-05 15:00:00

Transaction ID: 1234

Operation: Update

Table: Employees

Record ID: 5678

New Value: {Name: Jane Doe, Salary: $60,000}

- This redo record indicates that at 3 pm on May 5, 2023, transaction ID 1234 updated the record for employee ID 5678 in the Employees table, changing the employee's name to Jane Doe and salary to $60,000.
- By applying this redo record during recovery, the DBMS can ensure that the change is applied to the database, even if it was not yet written to disk at the time of the failure.

RECOVERY CONCEPTS

An example of an undo operation:

Timestamp: 2023-05-05 15:30:00

Transaction ID: 5678

Operation: Rollback

- This undo record indicates that at 3:30 pm on May 5, 2023, transaction ID 5678 was rolled back, meaning that any changes made by this transaction were removed from the database.
- By undoing the changes made by the rolled-back transaction during recovery, the DBMS can ensure that the database remains in a consistent state.

RECOVERY CONCEPTS

Recovery Manager

Definition:

- A recovery manager is a component of a DBMS that is responsible for managing the recovery process in the event of a failure.
- The recovery manager is designed to ensure that the database remains consistent and durable, even in the event of a failure.

Importance of a recovery manager in recovery:

- The recovery manager provides a centralized mechanism for managing the recovery process.
- Which can help to minimize downtime and ensure that the database remains available to users.

RECOVERY CONCEPTS

The components of a recovery manager typically include:

- **Logging component:** Responsible for writing log records to disk to ensure durability.
- **Checkpointing component:** Responsible for triggering and managing checkpoints to reduce the amount of work required during recovery.
- **Recovery component:** Responsible for applying redo and undo operations to bring the database back to a consistent state.

RECOVERY CONCEPTS

An example of a recovery manager in action might look like this:

- The DBMS detects a failure, such as a power outage.
- The recovery manager is activated and begins the recovery process.
- The logging component reads the log records written before the failure to determine the current state of the database.
- The checkpointing component is used to identify a stable starting point for the recovery process.
- The recovery component applies redo and undo operations to bring the database back to a consistent state.
- The recovery manager signals that the recovery process is complete, and the database is made available to users.

Transaction Management

Definition:

- Transaction management is a key component of a DBMS that is responsible for ensuring that database transactions are executed in a reliable and consistent manner.
- Transaction management is essential for maintaining the integrity of the database and ensuring that it remains consistent in the event of a failure.

Importance of Transaction Management:

- Transactions are the basic unit of work in a DBMS, and the transaction management component ensures that all transactions are executed in an all-or-nothing manner.
- In other words, a transaction is either executed completely, or not at all.

RECOVERY CONCEPTS

There are two types of transactions: Atomic and Composite.

- An atomic transaction is a single, indivisible unit of work that is executed as a single entity.
- A composite transaction is composed of multiple atomic transactions that are executed as a single unit of work.

RECOVERY CONCEPTS

An example of transaction management in action might look like this:

- A user initiates a transaction to transfer funds from one account to another.
- The DBMS begins the transaction and creates a transaction log to track the progress of the transaction.
- The DBMS executes the necessary database operations to transfer the funds.
- The DBMS writes the changes to the transaction log to ensure durability.
- If the transaction is successful, the DBMS commits the changes to the database and completes the transaction.
- If the transaction fails, the DBMS rolls back the changes and aborts the transaction.

DATABASE RECOVERY

The ARIES Recovery Algorithm

- The ARIES Recovery Algorithm is based on:
  - **WAL** (Write Ahead Logging)
  - **Repeating history during redo**:
    - ARIES will retrace all actions of the database system prior to the crash to reconstruct the database state when the crash occurred.
  - **Logging changes during undo**:
    - It will prevent ARIES from repeating the completed undo operations if a failure occurs during recovery, which causes a restart of the recovery process.

DATABASE RECOVERY

The ARIES Recovery Algorithm (contd.)

- The ARIES recovery algorithm consists of three steps:
  1. **Analysis**: step identifies the dirty (updated) pages in the buffer and the set of transactions active at the time of crash. The appropriate point in the log where redo is to start is also determined.
  2. **Redo**: necessary redo operations are applied.
  3. **Undo**: log is scanned backwards and the operations of transactions active at the time of crash are undone in reverse order.

DATABASE RECOVERY

The ARIES Recovery Algorithm (contd.)

- **The Log and Log Sequence Number (LSN)**
  - A log record is written for:
    - (a) data update
    - (b) transaction commit
    - (c) transaction abort
    - (d) undo
    - (e) transaction end
  - In the case of undo a compensating log record is written.

DATABASE RECOVERY

The ARIES Recovery Algorithm (contd.)

- **The Log and Log Sequence Number (LSN) (contd.)**
  - A unique LSN is associated with every log record.
    - LSN increases monotonically and indicates the disk address of the log record it is associated with.
    - In addition, each data page stores the LSN of the latest log record corresponding to a change for that page.
  - A log record stores
    - (a) the previous LSN of that transaction
    - (b) the transaction ID
    - (c) the type of log record.

DATABASE RECOVERY

The ARIES Recovery Algorithm (contd.)

- **The Log and Log Sequence Number (LSN) (contd.)**
- A log record stores:
  1. Previous LSN of that transaction: It links the log record of each transaction. It is like a back pointer points to the previous record of the same transaction
  2. Transaction ID
  3. Type of log record
- For a write operation the following additional information is logged:
  1. Page ID for the page that includes the item
  2. Length of the updated item
  3. Its offset from the beginning of the page
  4. BFIM of the item
  5. AFIM of the item

DATABASE RECOVERY

The ARIES Recovery Algorithm (contd.)

- The **Transaction table** and the **Dirty Page table**
  - For efficient recovery following tables are also stored in the log during checkpointing:
    - **Transaction table**: Contains an entry for each active transaction, with information such as transaction ID, transaction status and the LSN of the most recent log record for the transaction.
    - **Dirty Page table**: Contains an entry for each dirty page in the buffer, which includes the page ID and the LSN corresponding to the earliest update to that page.

DATABASE RECOVERY

The ARIES Recovery Algorithm (contd.)

- Checkpointing
  - A checkpointing does the following:
    - Writes a begin_checkpoint record in the log
    - Writes an end_checkpoint record in the log. With this record the contents of transaction table and dirty page table are appended to the end of the log.
    - Writes the LSN of the begin_checkpoint record to a special file. This special file is accessed during recovery to locate the last checkpoint information.
  - To reduce the cost of checkpointing and allow the system to continue to execute transactions, ARIES uses "fuzzy checkpointing".

DATABASE RECOVERY

The ARIES Recovery Algorithm (contd.)

- The following steps are performed for recovery
  - **Analysis phase**: Start at the begin_checkpoint record and proceed to the end_checkpoint record. Access transaction table and dirty page table are appended to the end of the log. Note that during this phase some other log records may be written to the log and transaction table may be modified. The analysis phase compiles the set of redo and undo to be performed and ends.
  - **Redo phase**: Starts from the point in the log up to where all dirty pages have been flushed, and move forward to the end of the log. Any change that appears in the dirty page table is redone.
  - **Undo phase**: Starts from the end of the log and proceeds backward while performing appropriate undo. For each undo it writes a compensating record in the log.
- The recovery completes at the end of undo phase.

DATABASE RECOVERY

(a)

Lsn	Last_lsn	Tran_id	Type	Page_id	Other_information
1	0	T_1	update	C	...
2	0	T_2	update	B	...
3	1	T_1	commit		...
4	begin checkpoint				
5	end checkpoint				
6	0	T_3	update	A	...
7	2	T_2	update	C	...
8	7	T_2	commit		...

(b)

TRANSACTION TABLE

Transaction_id	Last_lsn	Status
T_1	3	commit
T_2	2	in progress

DIRTY PAGE TABLE

Page_id	Lsn
C	1
B	2

(c)

TRANSACTION TABLE

Transaction_id	Last_lsn	Status
T_1	3	commit
T_2	8	commit
T_3	6	in progress

DIRTY PAGE TABLE

Page_id	Lsn
C	1
B	2
A	6

Figure 19.6

An example of recovery in ARIES. (a) The log at point of crash. (b) The Transaction and Dirty Page Tables at time of checkpoint. (c) The Transaction and Dirty Page Tables after the analysis phase.

DATABASE RECOVERY

10 Recovery in multidatabase system

- A multidatabase system is a special distributed database system where one node may be running relational database system under UNIX, another may be running object-oriented system under Windows and so on.
- A transaction may run in a distributed fashion at multiple nodes.
- In this execution scenario the transaction commits only when all these multiple nodes agree to commit individually the part of the transaction they were executing.
- This commit scheme is referred to as "**two-phase commit**" (**2PC**).
  - If any one of these nodes fails or cannot commit the part of the transaction, then the transaction is aborted.
- Each node recovers the transaction under its own recovery protocol.

SUMMARY OF RECOVERY CONCEPTS

- Recovery concepts in database management systems (DBMS) refer to the methods and techniques used to restore data to a consistent state after a failure or error.
- Some of the main recovery concepts include checkpoints, transaction logging, rollbacks, and forward recovery.
- Checkpoints are markers that indicate a specific point in time when the system was in a consistent state.
- Transaction logging involves recording all changes made to the database during transactions in a log file, which can be used to undo or redo transactions if necessary.
- Rollbacks involve undoing a transaction that was not completed successfully, while forward recovery involves redoing transactions that were not completed due to a failure.

SELF-ASSESSMENT QUESTIONS

1. In the _____ phase, the system replays updates of all transactions by scanning the log forward from the last checkpoint.

a) Repeating

b) Redo

c) Replay

d) Undo

2. The actions which are played in the order while recording it is called _____ history.

a) **Repeating**

b) Redo

c) Replay

d) Undo

SELF-ASSESSMENT QUESTIONS

3. Logs consist of a ___ of records.

(a) Compilations

(b) Series

(c) Data

(d) Table

4. ___ is used to keep track of all database operations.

(a) Dragging

(b) Logging

(c) Receiving

(d) Recovery

TERMINAL QUESTIONS

1. What is a database recovery concept, and why is it important in DBMS?
2. What are the different types of database failures that can occur, and how can they be prevented or resolved?
3. What are the different phases of database recovery, and how does each phase contribute to the overall recovery process?
4. What is the role of the transaction log in database recovery, and how does it help to ensure data consistency and durability?

REFERENCES FOR FURTHER LEARNING OF THE SESSION

Reference Books:

1. "Database Management Systems" by Raghu Ramakrishnan and Johannes Gehrke - This is a comprehensive textbook that covers both theoretical and practical aspects of DBMS, including data modeling, relational algebra, SQL, query optimization, transaction management, and distributed databases.

2. "Fundamentals of Database Systems" by Ramez Elmasri and Shamkant Navathe - This is another popular textbook that covers the fundamental concepts of DBMS, including database design, normalization, SQL programming, transaction management, and concurrency control.

Sites and Web links:

1. https://www.geeksforgeeks.org/database-recovery-techniques-in-dbms/
2. https://www.tutorialspoint.com/dbms/dbms_data_recovery.htm
3. https://www.javatpoint.com/dbms-log-based-recovery

THANK YOU

Team – DBMS

COURSE NAME: DBMS
COURSE CODE:22CS2110 RA

TOPIC: SHADOW PAGING

AIM OF THE SESSION

To familiarize students about the shadow paging concept in DBMS.

INSTRUCTIONAL OBJECTIVES

This Session is designed to:

- Their applications in ensuring data consistency, durability, and availability.

LEARNING OUTCOMES

At the end of this session, you should be able to:

- Understand about the benefits and limitations of the shadow paging technique in DBMS.

SHADOW PAGING

Definition:

- Shadow paging is a recovery technique used in database management systems to ensure data consistency and integrity in the event of a system failure.
- In shadow paging, the database system maintains two page tables: a current page table and a shadow page table.
- It is a recovery technique where the transactions are performed in the main memory.
- After completion of transactions, they get updated in the database.
- However, it didn’t show up in the database if failures happen during transaction execution.
- The main use of this technique is maintaining the consistency in data if failure happens in any case.
- The technique is also known with the name of Cut of Place Updating.

SHADOW PAGING

- It offers durability and atomicity to the system in the entire process.
- This concept utilizes quite a few disks for completing the operation.
- It is known to give power to manipulating pages in a database that is a very important step to perform.

SHADOW PAGING

Kinds of Functions do Shadow Paging Perform in different environments

- **Single User Environment**

If we talk about the single-user environment, the recovery scheme never asks for the log. The entire operation can be easily completed without the use of a log in any manner.

- **Multi-User Environment**

Things are entirely different when we talk about performing shadow paging in a multiuser environment. There will be strongly a requirement for a log as it involves a concurrency control procedure.

SHADOW PAGING

Example of Shadow Paging:

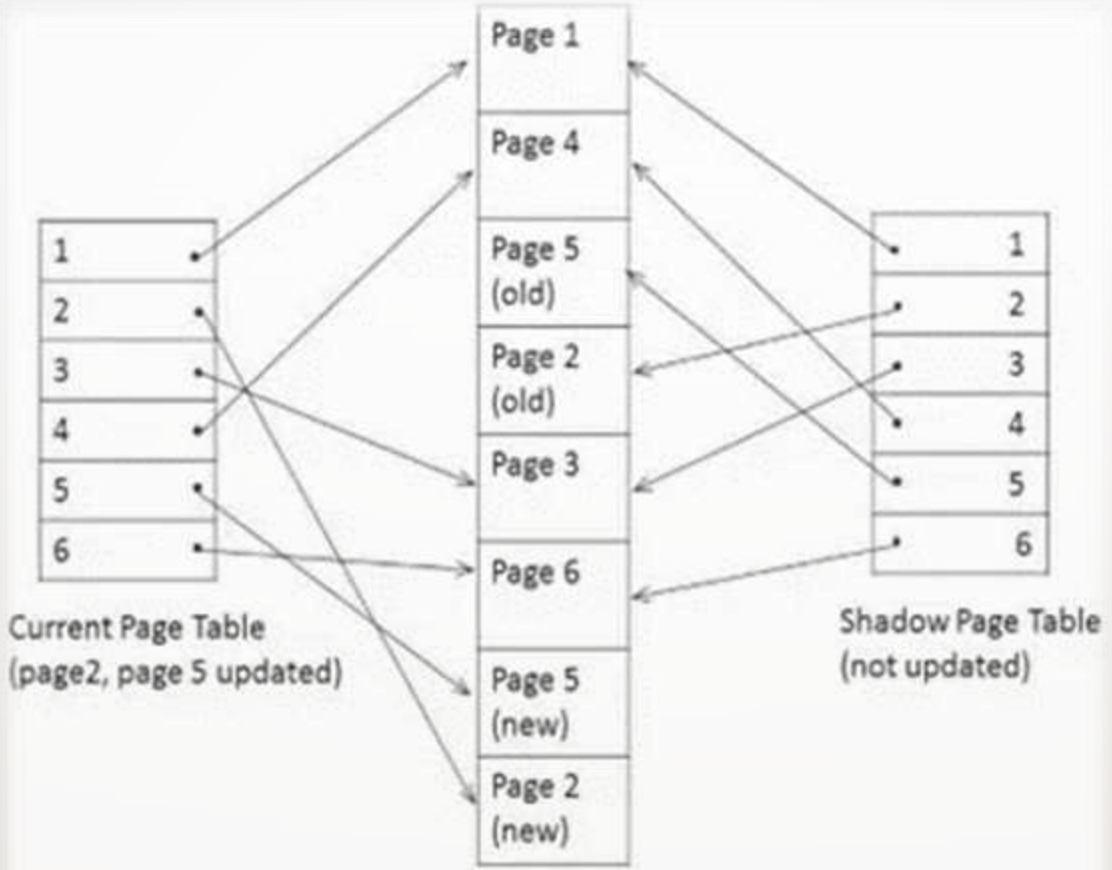


Figure 1

SHADOW PAGING

Advantages of Shadow Paging in DBMS

- There is overhead removed in log record output with the help of shadow paging.
- It makes sure that the recovery process is performed in a much better way from the crash.
- The shadow paging asks for a very less number of disks for performing the entire operation.
- The operations like Undo and Redo don’t need to be performed during shadow paging.
- It is an inexpensive and faster method to perform recovery after a crash.

SHADOW PAGING

Disadvantages of shadow paging in DBMS

- It becomes tough to maintain pages that are closer to the disk due to location changes.
- There is a chance of data fragmentation while performing shadow paging.
- The system needs multiple blocks for performing a single transaction which reduces the execution speed.
- Extending the algorithm becomes difficult for letting transactions operate concurrently.

SHADOW PAGING

Is Shadow paging different from log-based recovery?

Log-based recovery is a very important used structure to record database modifications. Many people inquire how they are different from the shadow paging. Have a look at the major differences between the two below:

- The log record is made using different fields like data item identifier, transaction identifier, new value, and old value.
- On the other hand, shadow paging is made from a certain number of fixed-size disk pages.

SHADOW PAGING

Is Shadow paging different from log-based recovery?

- The search process in the log-based recovery is quite long, while things become faster in shadow paging thanks to the elimination of overhead of log record output.
- There is a need for only one block to complete a single transaction in log-based recovery, but the commit of one transaction asks for multiple blocks in shadow paging.
- The locality property of pages can be lost in shadow paging, while it never happens in log-based recovery.

SUMMARY OF SHADOW PAGING

- Shadow paging is a technique for providing atomicity and durability (two of the ACID properties) in database systems.
- A page in this context refers to a unit of physical storage (probably on a hard disk), typically of the order of 1 to 64 KiB.
- Shadow paging is a copy-on-write technique for avoiding in-place updates of pages.
- Instead, when a page is to be modified, a shadow page is allocated.
- Since the shadow page has no references (from other pages on disk), it can be modified liberally, without concern for consistency constraints, etc.
- When the page is ready to become durable, all pages that referred to the original are updated to refer to the new replacement page instead.
- Because the page is "activated" only when it is ready, it is atomic.

SELF-ASSESSMENT QUESTIONS

1. **Shadow paging has**

(a) No redo

(b) No Undo

(c) Redo but no undo

(d) Neither redo or undo

2. In the _____ scheme, a transaction that wants to update the database first creates a complete copy of the database.

(a) Shadow copy

(b) Shadow Paging

(c) Update log record

(d) All of the above

SELF-ASSESSMENT QUESTIONS

3. In the _____ phase, the system replays updates of all transactions by scanning the log forward from the last checkpoint.

(a) Repeating

(b) Redo

(c) Replay

(d) undo

4. The actions which are played in the order while recording it is called _____ history.

(a) Repeating

(b) Redo

(c) Replay

(d) Undo

TERMINAL QUESTIONS

1. What is shadow paging method used for?
2. What is the difference between shadow paging and log based recovery?
3. Describe shadow paging recovery technique.
4. Explain current page table and shadow page table?

REFERENCES FOR FURTHER LEARNING OF THE SESSION

Reference Books:

1. "Database Management Systems" by Raghu Ramakrishnan and Johannes Gehrke - This is a comprehensive textbook that covers both theoretical and practical aspects of DBMS, including data modeling, relational algebra, SQL, query optimization, transaction management, and distributed databases.

2. "Fundamentals of Database Systems" by Ramez Elmasri and Shamkant Navathe - This is another popular textbook that covers the fundamental concepts of DBMS, including database design, normalization, SQL programming, transaction management, and concurrency control.

Sites and Web links:

1. https://www.geeksforgeeks.org/shadow-paging-dbms/
2. https://www.tutorialspoint.com/what-is-shadow-paging-in-dbms
3. https://whatisdbms.com/shadow-paging-in-dbms/

THANK YOU

Team – DBMS

COURSE NAME: DBMS
COURSE CODE:22CS2110 RA

TOPIC: ARIES

AIM OF THE SESSION

To familiarize students with the ARIES concept in DBMS..

INSTRUCTIONAL OBJECTIVES

This Session is designed to:

- Their applications in ensuring data consistency, durability, and availability.

LEARNING OUTCOMES

At the end of this session, you should be able to:

- Different use cases where the ARIES protocol can be applied, including transaction processing systems, real-time systems, and distributed databases.

ARIES

Recovery is one of the most complicated part of DBMS, just because it has to deal with multitude of failure scenarios.

ARIES is one of few algorithm that is documented and explain how to deal with recovery for databases that utilize STEAL and NO-FORCE policies.

The failure scenario we are concerned with are:

1.Transaction failure: When a transaction is aborted, changes made by it need to be rolled back.

2.RDBMS process restart/crash: In this case all data that was in memory is lost and in progress transactions need to be rolled back. Committed transaction need to be reapplied.

ARIES

- RDBMS maintains data in memory and that data is persisted to durable storage depending on policies of database.
- The unit of data that can be written atomically to disk is called a “page”. Persisting a transaction commit may require multiple “page” writes.
- If RDMBS process is restarted, the in-memory data is lost.
- On restart, in-progress transactions need to be rolled back to preserve “atomicity” property of database, and this is known as UNDO.
- Similarly, if a committed transaction was not written to disk before crash, it need to be reapplied to preserve “durability” property of database. This is called REDO.

ARIES

Algorithm for Recovery and Isolation Exploiting Semantics (ARIES) is based on the Write Ahead Log (WAL) protocol.

ARIES starts by finding log records for operations that were not written to disk on crash and then replays all of them. This even includes transactions that need to rolled back. It brings the database to the same state as it was before crash. This process is called “repeating history”. Once database is brought up to date, all the transactions that need to be rolled back are undone.

ARIES

ARIES is a three phase algorithm:

- **Analysis phase:** This phase reads the last checkpoint record in the log to figure out active transactions and dirty pages at point of crash/restart. A page is considered dirty if it was modified in memory but was not written to disk. This information is used by next two phases.
- **REDO phase:** In this phase operations in log are reapplied to bring the state of the database to current.
- **UNDO phase:** This phase proceeds from the end of log reverting back operations for uncommitted transactions. This has impact of rolling them back. Note that this is same procedure that database performs when rolling back a transaction in normal mode of operation for STEAL policy.

ARIES

ARIES maintains two data structures and adds one more field to log record:

- **Transaction table:** It contains all the transactions that are active at any point of time (i.e. are started but not committed/aborted). The table also stores the LSN of last log record written by the transaction in “lastLSN” field.
- **Dirty page table:** Contains an entry for each page that has been modified but not written to disk. The table also stores the LSN of the first log record that made the associated page dirty in a field called “recoveryLSN” (also called “firstLSN”). This is the log record from which REDO need to restart for this page.

ARIES

- In addition log records are also updated to contain a field called “prevLSN” which points to previous log record for the same transaction. This creates a linked list of all log records for a transaction. When a new log record is created, “lastLSN” from transaction table is filled into its “prevLSN” field. And the LSN of current log record becomes the “lastLSN” in transaction table.
- Here is updated log record table with prevLSN filled in:

LSN	Prev LSN	Transaction ID	Type	Page ID
1	NIL	T1	UPDATE	P3
2	NIL	T2	UPDATE	P2
3	1	T1	COMMIT	
4	CHECKPOINT			
5	NIL	T3	UPDATE	P1
6	2	T2	UPDATE	P3
7	6	T2	COMMIT	

ARIES

- During checkpointing, a checkpoint log record is created. This log record contains the content of both "Transaction table" and "Dirty page table". "Analysis" phase starts by reading last checkpoint log record to get the information about active transactions and dirty pages. Here is content of "Transaction table" and "Dirty page table" at the checkpoint stage in above table at LSN 4:

Transaction Table

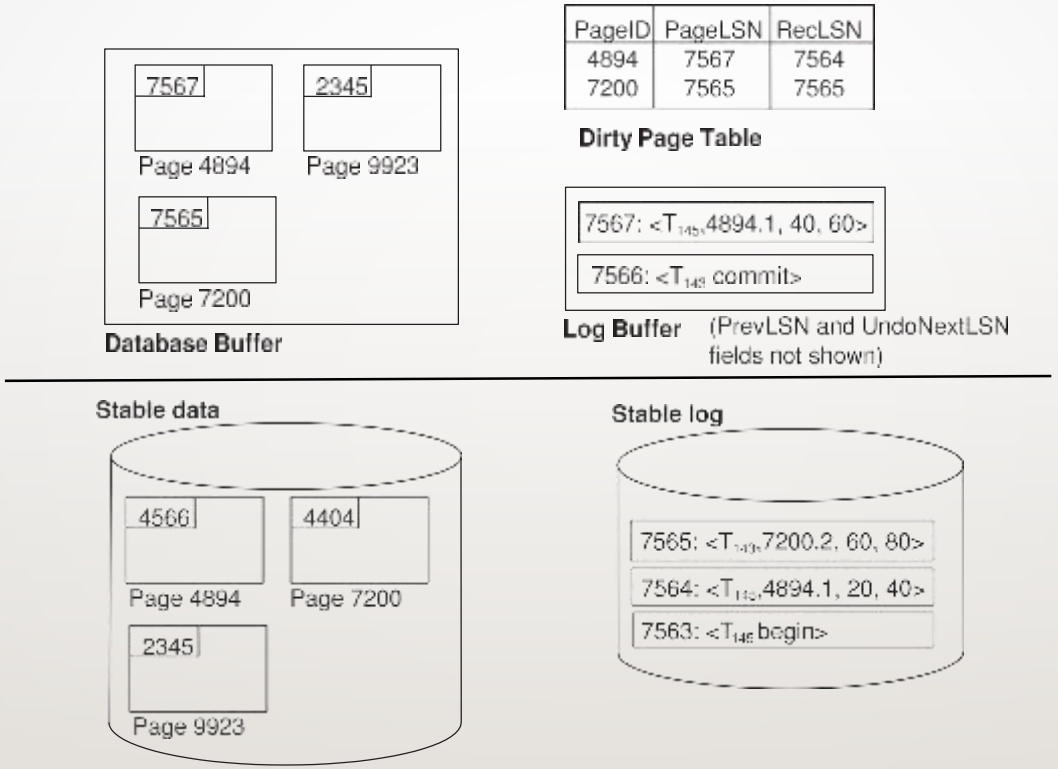
Transaction ID	Last LSN	Status
T1	3	Commit
T2	2	In Progress

Dirty Page Table

Page ID	Recovery LSN
P3	1
P2	2

ARIES

- This whole setup can be visualized in following picture, pay attention to LSN for P2 in the "pages" list and in dirty table (dirty page table has the first LSN, whereas the P2 page has the last LSN):



ARIES Data Structures

Analysis Phase

- Analysis phase reads the “checkpoint log record” for the latest checkpoint and then scans the log forward to create list of all active transactions and dirty pages. When scanning log forward:
- If a new transaction is started, it is added to the transaction table.
- If a commit/abort is seen for a transaction, its entry is removed from transaction table.
- If a page is updated, a new entry is added to dirty page table and “recoveryLSN” is set to the LSN of the log record.

Here is how the Transaction table and Dirty page table will look like after analysis is done:

Transaction Table (Rows with [x] are deleted, but are left here to show full process)

	Transaction ID	Last LSN	Status
[x]	*T1*	*3*	*Commit*
[x]	*T2*	*7*	*Commit*
	T3	5	In Progress

Dirty Page Table

Page ID	Recovery LSN
P1	5
P2	2
P3	6

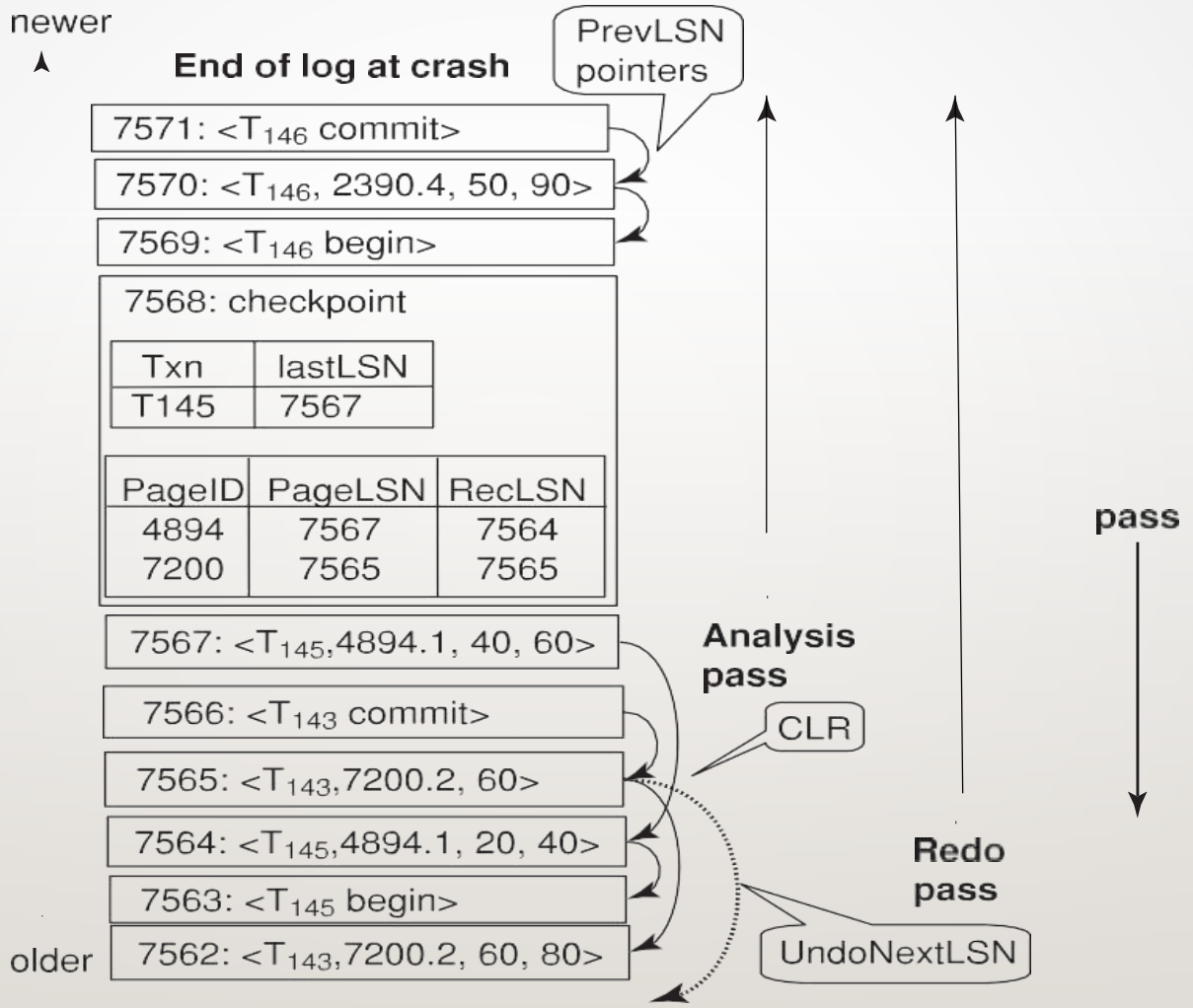
REDO phase

- At the end of analysis phase, the first LSN that caused a page to become dirty is known (its the lowest LSN value in dirty page table).
- REDO phase starts at this point in the log and processes all log entries from that point
- If the log record point to a page that is not dirty, ignore that log record.
- If the log record point to a dirty page, but its “recoveryLSN” is more than current record’s LSN, it means that the page was committed/aborted and a later transaction then updated the page again. So ignore this log record.
- Otherwise check the LSN on the page (as discussed before, each page contains LSN that last updated the page).

ARIES

- If the LSN of page is greater than this logs record's LSN, again ignore the record as this situation indicates that page made to disk successfully for a later transaction.
- Otherwise, redo the action specified in the log record.
- At the end of this phase database is in same state as it was before crash. Here is a timeline showing how three phases make a pass through the log.

Recovery actions in ARIES



UNDO phase

- This phase starts from end of the log and goes backward.
- This phase undoes all transactions that were active at time of restart/crash.
- Information about these transactions are available in the transaction table.
- The lastLSN field in transaction table also tells the first record that need to be undone.
- And as we saw before, “prevLSN” field in log record creates a linked list of operations in the transaction. So all log records in this linked list need to be undone to rollback the transaction.
- Another good property of UNDO is that unlike REDO it doesn’t need to check any state, it need to unconditionally undo log records of all active transactions.

ARIES

- To avoid repeating undo during repeated restart, each time a log record is undone, a “Compensation Log Record” is added to the log.
- In addition to the information about the undo, the record also contains the next log record for the transaction that need to be undone.
- And the next log record that needs to be undone is stored in the “prevLSN” of the current log being undone.
- After that UNDO becomes a simple operation of going in reverse through the linked list of log records for an active transaction and undoing it.
- It need to do this for all active transactions in the transaction table.

SUMMARY OF RECOVERY CONCEPTS

- The ARIES (Algorithm for Recovery and Isolation Exploiting Semantics) concept is a database recovery and concurrency control protocol that was proposed by Mohan et al. in 1992.
- ARIES is widely used in modern database management systems (DBMS) due to its efficiency and reliability.
- The ARIES protocol is based on the Write Ahead Logging (WAL) technique, which requires all changes to be logged in a sequential manner before they are made to the database.
- The WAL enables ARIES to support transaction rollback and recovery in case of system failures.
- ARIES provides both deferred and immediate update techniques for database recovery.
- In ARIES, a transaction log is maintained to record all changes made to the database.
- The log is used during the recovery process to undo or redo the changes made to the database by the transactions.
- The log also provides a way to ensure atomicity and durability of transactions.

SELF-ASSESSMENT QUESTIONS

1. ARIES uses a _____ to identify log records, and stores it in database.

(a) Log sequence number

(b) Log number

(c) Lock number

(d) sequence

2. _____ is used to minimize unnecessary redos during recovery.

(a) Dirty page table

(b) Page table

(c) Dirty redo

(d) All of the mentioned

SELF-ASSESSMENT QUESTIONS

3. The _____ contains a list of pages that have been updated in the database buffer.

(a) Dirty page table

(b) Page table

(c) Dirty redo

(d) All of the mentioned

4. _____ rolls back all transactions that were incomplete at the time of crash.

(a) Analysis pass

(b) Redo pass

(c) Undo pass

(d) None of the mentioned

TERMINAL QUESTIONS

1. What are the three principles behind ARIES?
2. What is the data structure of ARIES?
3. What is the concept of ARIES algorithm in DBMS?
4. What are the advantages of ARIES recovery algorithm?

REFERENCES FOR FURTHER LEARNING OF THE SESSION

Reference Books:

1. "Database Management Systems" by Raghu Ramakrishnan and Johannes Gehrke - This is a comprehensive textbook that covers both theoretical and practical aspects of DBMS, including data modeling, relational algebra, SQL, query optimization, transaction management, and distributed databases.

2. "Fundamentals of Database Systems" by Ramez Elmasri and Shamkant Navathe - This is another popular textbook that covers the fundamental concepts of DBMS, including database design, normalization, SQL programming, transaction management, and concurrency control.

Sites and Web links:

https://www.geeksforgeeks.org/algorithm-for-recovery-and-isolation-exploiting-semantics-aries/

https://blog.acolyer.org/2016/01/08/aries/

https://www.shreya-shankar.com/aries/

THANK YOU

Team – DBMS