**ARINDA HILLARY**

**DBMS**

**ASSIGNMENT 9: DB recovery**

**3/15/2024**

1. **Discuss the types of failure that may occur in a database environment.  Explain why it is important for a multi-user DBMS to provide a recovery mechanism.**

**Types of Failures:**

1. Hardware Failures:
   * Disk crashes: Physical damage to storage devices.
   * Power outages: Power loss can interrupt ongoing transactions and corrupt data.
2. Software Failures:
   * Operating system errors: Issues with the underlying operating system can affect database accessibility and stability.
   * Application errors: Faulty code in applications interacting with the database can cause data corruption or inconsistencies.
3. Human Errors:
   * Accidental data deletion: Users might delete critical data unintentionally.
   * Logical errors in SQL statements: Incorrect queries can lead to unintended data modification or loss.
4. Natural Disasters:
   * Floods, fires, earthquakes, or other natural disasters can damage physical infrastructure and lead to data loss.

**Importance of Recovery Mechanisms in a Multi-user DBMS:**

1. Data Consistency**:** Recovery mechanisms ensure that data remains consistent and reliable even in the event of failures which helps restore the database to a known state, minimizing data loss and corruption.
2. Data Availability: After failures, recovery mechanisms enable restoring database access and functionality, allowing users to resume operations since back-ups of the data is made by recovery mechanisms.
3. User Confidence: Robust recovery mechanisms instill confidence in users that their data is protected and can be retrieved in case of unforeseen circumstances.

1. **Discuss how the log file (or journal) is a fundamental feature in any recovery mechanism. Explain what is meant by forward and backward recovery and describe how the log file is used in forward and backward recovery.**

The log file records all the transactions that have occurred in the system, including the details of each operation within a transaction, such as the data items affected and the before and after values. This record-keeping helps with the recovery processes.  
  
**Forward Recovery (Redo):**

Forward recovery is used after a system crash or unexpected shutdown which involves replaying the log file forward from a known consistent state (last checkpoint) up to the point of failure. This redoes the committed changes since the last consistent state, bringing the database to a consistent state.

**Backward Recovery (Undo):**

Used in scenarios like transaction rollback or recovering from logical errors which involves scanning the log file backward from the point of failure. This process undoes the changes made by the uncommitted transaction, reverting the database state to a point before the failure or error.

**How the Log File is used in Redo and Undo Operations:**

* **Forward Recovery:**

The log file stores record of all thedata modifications like new values for the updates. During forward recovery, the system replays these changes to the database, reapplying the committed transactions.

* **Backward Recovery:**

The log file records include information about the previous data state before the modification like old values for the updates. During backward recovery, the system uses this information to revert the database to its state before the uncommitted transaction's changes.

1. **What is the significance of the write-ahead log protocol? How do checkpoints affect the recovery protocol?**

**Significance of Write-Ahead Logging (WAL):**

Atomicity: WAL upholds the concept of atomicity in transactions. All changes associated with a transaction are either entirely written to the database or not written at all. This prevents partial updates and ensures data consistency.

Durability: WAL ensures that data modifications are written to a stable storage medium like a hard disk before they are reflected in the main database file. This guarantees data persistence even in case of system crashes or power outages.

Recoverability: The log file created by WAL serves as a vital component for recovery mechanisms. It allows the system to perform redo and undo operations in case of any errors.

How the checkpoints affect the recovery protocol.

* + 1. **Reducing Recovery Time:**

During normal operation, the database continuously logs all modifications made through transactions. The log file grows in size as transactions occur.With a check pointwhen the system identifies the last completed checkpoint, recovery starts from the checkpoint instead of beginning of the log, this reduces the amount of the data replay that has to be redone.

Only the log entries after the checkpoint are processed, focusing on changes that occurred since the last consistent state.

* + 1. **Faster Forward Recovery:**
* Forward recovery (Redo): This involves replaying the log forward from a known consistent state to bring the database to a consistent state reflecting committed transactions.
* Backward recovery (Undo): Used for transaction rollbacks or recovering from logical errors. It involves scanning the log backward and undoing the changes made by uncommitted transactions.

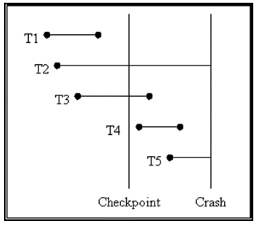
1. **Compare and contrast the deferred update and immediate update recovery protocols.**

Deferred Update (NO-UNDO/REDO). The Changes made by a transaction are not immediately reflected in the database, they are first recorded in the log file and applied only after the transaction reaches its commit point. If a system crash occurs before the commit, no changes need to be undone since nothing was saved to the database. After a crash, the recovery process involves redoing transactions that were committed but not yet applied to the database.

**While**

Immediate Update (UNDO/REDO): As soon as a transaction modifies a data item, the change is immediately reflected in the database, and both the old and new values are recorded in the log file. If a transaction fails or a system crash occurs, the recovery process may involve both undoing the changes of uncommitted transactions and redoing the changes of committed transactions that were not fully written to disk.

1. **Figure out what will happen to each of the transactions under Deferred Update and Immediate Update protocols when a crash occurs.  
   Assume that database buffers are only written to disk at checkpoints.**



|  |  |  |
| --- | --- | --- |
|  | **Deferred Update** | **Immediate Update** |
| T1 | Do nothing | Do nothing |
| T2 | Restart | Undo and Restart |
| T3 | Redo | Redo |
| T4 | Redo | Redo |
| T5 | Restart | Restart |