SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with ‘A’ Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING-IOT Including CS&BCT

COURSE NAME : 19SB504 DATABASE MANAGEMENT SYSTEMS

III YEAR / V SEMESTER

Unit V- CONCURRENCY CONTROL AND RECOVERY SYSTEM

Topic : **BUFFER MANAGEMENT, FAILURE WITH LOSS OF NON-VOLATILE STORAGE**

Database Buffer

A database buffer is a temporary storage area in the main memory.

It allows storing the data temporarily when moving from one place to another.

A database buffer stores a copy of disk blocks.

A Buffer Manager is responsible for allocating space to the buffer in order to store data into the buffer.

If a user request a particular block and the block is available in the buffer, the buffer manager provides the block address in the main memory.

If the block is not available in the buffer, the buffer manager allocates the block in the buffer.

If free space is not available, it throws out some existing blocks from the buffer to allocate the required space for the new block.

The blocks which are thrown are written back to the disk only if they are recently modified when writing on the disk.

If the user requests such thrown-out blocks, the buffer manager reads the requested block from the disk to the buffer and then passes the address of the requested block to the user in the main memory.

However, the internal actions of the buffer manager are not visible to the programs that may create any problem in disk-block requests.

The buffer manager is just like a virtual machine

For serving the database system in the best possible way, the buffer manager uses the following 3 methods:

1.Buffer Replacement Strategy:

If no space is left in the buffer, it is required to remove an existing block from the buffer before allocating the new one.

The various operating system uses the LRU (least recently used) scheme. In LRU, the block that was least recently used is removed from the buffer and written back to the disk.

Such type of replacement strategy is known as Buffer Replacement Strategy.

2. Pinned Blocks:

If the user wants to recover any database system from the crashes, it is essential to restrict the time when a block is written back to the disk.

In fact, most recovery systems do not allow the blocks to be written on the disk if the block updation being in progress.

Such types of blocks that are not allowed to be written on the disk are known as pinned blocks.

Luckily, many operating systems do not support the pinned blocks.

3. Forced Output of Blocks:

In some cases, it becomes necessary to write the block back to the disk even though the space occupied by the block in the buffer is not required.

When such type of write is required, it is known as the forced output of a block.

It is because sometimes the data stored on the buffer may get lost in some system crashes, but the data stored on the disk usually does not get affected due to any disk crash.

Failure with loss of non-volatile storage

To recover from the loss of nonvolatile storage, the system restores the database to disk by using the most recent dump.

Then, it consults the log and redoes all the transactions that have committed since the most recent dump occurred.

Notice that no undo operations need to be executed.

The failure with the loss of non-volatile storage in a Database Management System (DBMS) can have serious consequences and impact the integrity and availability of data.

Non-volatile storage typically refers to the persistent storage devices such as hard drives, solid-state drives, or other long-term storage mediums where data is stored even when the power is turned off.

Data Integrity Loss:

Sudden storage failures can lead to data corruption or loss if the affected storage contains critical database files.
Incomplete write operations or sudden power losses can result in inconsistencies within the data.

Recovery Challenges:

The DBMS may face difficulties in recovering data if the storage failure occurred during a critical operation, such as a transaction commit.
Incomplete transactions could leave the database in an inconsistent state.

Backup and Restore:

Regular backups are crucial for mitigating the impact of storage failures.

If a proper backup strategy is in place, you can restore the database to a point in time before the failure. However, any data changes made after the last backup may be lost.

Transaction Logs:

Transaction logs are essential for maintaining the consistency and recoverability of a database.

If transaction logs are intact, the DBMS may be able to replay the logs to bring the database back to a consistent state.

Redundancy and High Availability:

Implementing redundancy and high availability measures, such as RAID (Redundant Array of Independent Disks) for storage or clustering for the DBMS, can help mitigate the impact of storage failures.

These measures ensure that there are copies of data and services available even if one component fails.

Monitoring and Alerts:

Continuous monitoring of storage health and setting up alerts for potential issues can help identify and address storage failures proactively.

This can minimize the time between the occurrence of an issue and its resolution.

Data Recovery Services:

In extreme cases where data is lost or corrupted, specialized data recovery services may be necessary.

These services use advanced techniques to recover data from damaged storage devices.

Thank you ......