ST. XAVIER’S COLLEGE

**(Affiliated to Tribhuvan University)**

Maitighar, Kathmandu



**Database Management System Assignment #9**

**Submitted by:**

Bishal Pandey  
013BSCCSIT016

**Submitted to:**

|  |  |
| --- | --- |
| Er. Sanjay Kumar Yadav Lecturer, St. Xavier’s College |  |

**# Database recovery**

* There are many situations in which a transaction may not reach a commit or abort point.
  + - An operating system crash can terminate the DBMS processes
    - The DBMS can crash
    - The system might lose power
    - A disk may fail or other hardware may fail.
    - Human error can result in deletion of critical data.
* In any of these situations, data in the database may become inconsistent or lost.
* For example, if a transaction has completed 30 out of 40 scheduled writes to the database when the DBMS crashes, then the database may be in an inconsistent state as only part of the transaction’s work was completed.
* Database Recovery is the process of restoring the database and the data to a consistent state. This may include restoring lost data up to the point of the event (e.g. system crash).
* Two approaches are discussed here: Manual Reprocessing and Automated Recovery.

**Purpose of data recovery**

As a backup administrator, your principal duty is to devise, implement, and manage a backup and recovery strategy. In general, the purpose of a backup and recovery strategy is to protect the database against data loss and reconstruct the database after data loss. Typically, backup administration tasks include the following:

• Planning and testing responses to different kinds of failures

• Configuring the database environment for backup and recovery

• Setting up a backup schedule

• Monitoring the backup and recovery environment

• Troubleshooting backup problems

• Recovering from data loss if the need arisesa

**Types of failure**

To see where the problem has occurred, we generalize a failure into various categories, as follows −

**Transaction failure**

A transaction has to abort when it fails to execute or when it reaches a point from where it can’t go any further. This is called transaction failure where only a few transactions or processes are hurt.

Reasons for a transaction failure could be −

* **Logical errors** − Where a transaction cannot complete because it has some code error or any internal error condition.
* **System errors** − Where the database system itself terminates an active transaction because the DBMS is not able to execute it, or it has to stop because of some system condition. For example, in case of deadlock or resource unavailability, the system aborts an active transaction.

**System Crash**

There are problems − external to the system − that may cause the system to stop abruptly and cause the system to crash. For example, interruptions in power supply may cause the failure of underlying hardware or software failure.

Examples may include operating system errors.

**Disk Failure**

In early days of technology evolution, it was a common problem where hard-disk drives or storage drives used to fail frequently.

Disk failures include formation of bad sectors, unreachability to the disk, disk head crash or any other failure, which destroys all or a part of disk storage.

**The storage hierarchy**

You can doubtless think of many examples of storage hierarchies in ordinary life. For example, people live in neighborhoods, which are in towns, which are in regions, countries, continents, and so on up the line. The relations are generally many-to-one, although there are occasional one-to-one correspondences (e.g., Australia is both a country and a continent), and occasional exceptions (e.g., a person can straddle a city boundary).

**Buffer Management**

Data must be in RAM for DBMS to operate on it!

• Buffer Mgr hides the fact that not all data is in RAM DB MAIN MEMORY DISK disk page free frame Page Requests from Higher Levels BUFFER POOL choice of frame dictated by replacement policy When a Page is Requested ...

• Buffer pool information table contains:

• If requested page is not in pool: – Choose a frame for replacement (only un-pinned pages are candidates) – If frame is “dirty”, write it to disk – Read requested page into chosen frame

• Pin the page and return its address. If requests can be predicted (e.g., sequential scans) pages can be pre-fetched several pages at a time! More on Buffer Management

• Requestor of page must eventually unpin it, and indicate whether page has been modified: – dirty bit is used for this.

• Page in pool may be requested many times, – a pin count is used. – To pin a page, pin\_count++ – A page is a candidate for replacement iff pin\_count == 0 (“unpinned”)

• CC & recovery may entail additional I/O when a frame is chosen for replacement. (Write A head Log protocol; more later.)

**Transaction log**

Every SQL Server database has a transaction log that records all transactions and the database modifications made by each transaction. The transaction log must be truncated on a regular basis to keep it from filling up. However, some factors can delay log truncation, so monitoring log size is important. Some operations can be minimally logged to reduce their impact on transaction log size.

The transaction log is a critical component of the database and, if there is a system failure, the transaction log might be required to bring your database back to a consistent state. The transaction log should never be deleted or moved unless you fully understand the ramifications of doing this.

**Data updates**

**Immediate Update:** As soon as a data item is modified in cache, the disk copy is updated.

**Deferred Update:** All modified data items in the cache is written either after a transaction ends its execution or after a fixed number of transactions have completed their execution.

**Shadow update:** The modified version of a data item does not overwrite its disk copy but is written at a separate disk location.

**In-place update:** The disk version of the data item is overwritten by the cache version.

**Data caching**

A cache, in computing, is a data storing technique that provides the ability to access data or files at a higher speed.

A cache works in both hardware and software to provide similar functionality. In its physical or hardware form, it is a small form factor of internal memory that stores instances of the most frequently executed programs in the main memory to enable faster access when they are requested by the CPU.

A very commmon example of caching is in a Web browser, where a website's HTML, images, CSS, Javascript, etc is cached locally so that a page will load faster after its first hit.

**Transaction Roll back(undo) and Roll forward**

**Check Pointing, shadow paging**

Keeping and maintaining logs in real time and in real environment may fill out all the memory space available in the system. As time passes, the log file may grow too big to be handled at all. Checkpoint is a mechanism where all the previous logs are removed from the system and stored permanently in a storage disk. Checkpoint declares a point before which the DBMS was in consistent state, and all the transactions were committed.

**Recovery schemes (WAL: Write Ahead logging Protous)**

**Failure with loss of Non-volatile storage (general concepts)**

**Recovery in multidatabase system.**