# ST.XAVIER’S COLLEGE

# MAITIGHAR, KATHMANDU

****

**ASSIGNMENT #9**

**Database Management System**

**Submitted By:**

Pratibha Panta

013BSCCSIT028

4th Sem, 2nd Year

**Submitted To:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Er. Sanjay Kumar Yadav

Department of Computer Science

Lecturer

**#Database Recovery**

Data recovery is the process of restoring data that has been lost, accidentally deleted, corrupted or made inaccessible for any reason.[1]

**1) Purpose of data 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.

So to remove this all, database recovery is needed.[2]

**2) Types of failure**

Transaction failure: Individual transactions fail

Logical error: Internal problem within the transaction

System error: External problem during transaction execution (e.g., deadlock)

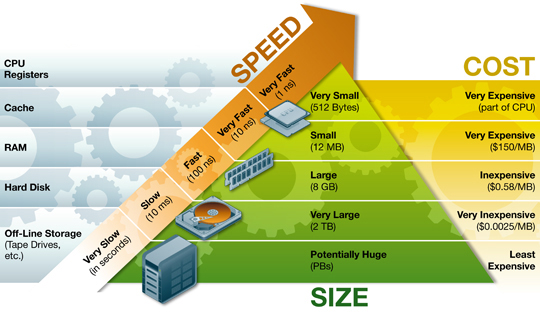
System crash: Problem with overall database server execution; terminates the current process

Fail-stop assumption: Data in non-volatile storage is unharmed in the event of a system crash

Disk failure: Problem with storage media

**3) The storage hierarchy**

The range of memory and storage within and attached to a computer system is known as the Storage Hierarchy and to help understand this further can be categorised into 4 segments. As memory and storage devices move down the hierarchy they reduce in performance and cost/MB or GB but tend to rise in capacity through to the last category which includes removable media which in effect has no restriction on capacity a device can store.[4]



Primary Storage is the top level and is made up of CPU registers, CPU cache and memory which are the only components that are directly accessible to the systems CPU. The CPU can continuously read data stored in these areas and execute all instructions as required quickly in a uniform manner. Secondary Storage differs from primary storage in that it is not directly accessible by the CPU. A system uses input/output (I/O) channels to connect to the secondary storage which control the data flow through a system when required and on request

Secondary storage is non-volatile so does not lose data when it is powered down so consequently modern computer systems tend to have a more secondary storage than primary storage. All secondary storage today consist of hard disk drives (HDD), usually set up in a RAID configuration, however older installations also included removable media such us magneto optical or MO

Tertiary Storage is mainly used as backup and archival of data and although based on the slowest devices can be classed as the most important in terms of data protection against a variety of disasters that can affect an IT infrastructure. Most devices in this segment are automated via robotics and software to reduce management costs and risk of human error and consist primarily of disk & tape based back up devices

Offline Storage is the final category and is where removable types of storage media sit such as tape cartridges and optical disc such as CD and DVD. Offline storage is can be used to transfer data between systems but also allow for data to be secured offsite to ensure companies always have a copy of valuable data in the event of a disaster.

**4) Buffer Management**

A DBMS must manage a huge amount of data, and in the course of processing the required space for the blocks of data will often be greater than the memory space available. For this there is the need to manage a memory in which to load and unload the blocks. The buffer manager is responsible primarily for managing the operations inherent saving and loading of the blocks. In fact, the operations that provide the buffer manager are these:

\* FIX: This command tells the operator of the buffer to load a block from disk and return the pointer to the memory where it is loaded. If the block was already in memory, the buffer manager needs only to return the pointer, otherwise he must load from disk and bring it into memory.[5]

\* SET DIRTY: invoking this command, you mark a block of memory as amended.[5]

\* Force: This command will cause the operator of the buffer to make the writing in synchronously with the completion (commit) the transaction\* FLUSH: This command will cause the operator of the buffer to perform the rescue, when in how NOT FORCE.[5]

Log: Sequence of records (sequential file)

Modified by appending (no updating)

• Contains information from which database can be restored

• Log and database stored on different mass storage devices

• Often replicated to survive single media failure • Contains valuable historical data not in database

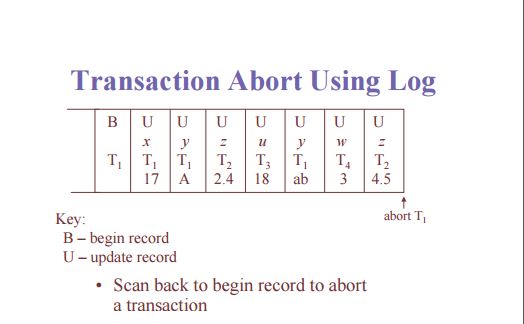
**5) Transaction Log**

Scan log backwards using tid to identify transaction’s update records

• Reverse each update using before image

• In a strict system, new values are unavailable to concurrent transactions (as a result of long term exclusive locks); hence rollback makes transaction atomic

• Problem: terminating scan (log can be long) • Solution: append begin record containing tid prior to first update record[3]



**6) Data Updates**

The Update and Update All methods allows the updating of data already in your database. The return value of an Update method is the number of rows that were just updated.

You can Update using two forms, Named parameters and by object. The object can be a POCO or a dynamic (i.e.ExpandoObject).

\_db.Users.UpdateById(Id: 1, Name: "Steve", Age: 50);

Note: There are differences in the generated SQL when using Named Parameters or object, where the object includes all properties including on the object, but the Named Parameters only includes those you specify

**7) Data Caching**

The Database Management System (DBMS) is a memory buffer which stores copies of portions of the database that the DBMS is currently using. Reading from memory is much faster than reading from the disk. The DBMS therefore returns a record more quickly if it is already stored in cache. As long as the required data is stored in cache, the data is immediately available. When the required data is not stored in cache, it must be copied from the disk and then stored in cache.[6]

**8) Transaction Roll back (Undo) and Role Foreword**

**9) Check pointing, shadow paging**

**10) Recovery Schemes (WAL: write ahead logging protocol)**

**11) Failure with loss of non-volatile storage (General Concept)**

**12) Recovery in database system**

**Reference**

[1] <http://searchdisasterrecovery.techtarget.com/definition/data-recovery>

[2] <http://holowczak.com/database-recovery/>

[3] <https://webdocs.cs.ualberta.ca/~zaiane/courses/cmput391/slides/L9-391-04.pdf>

[4] <http://www.ts.avnet.com/uk/products_and_solutions/storage/hierarchy.html>

[5] <http://www.scribd.com/doc/49586450/The-Buffer-Manager-of-a-DBMS#scribd>

[6] <https://msdn.microsoft.com/en-us/library/dd355169.aspx>