**DATABASE RECOVERY**

**Purpose of Data Recovery**

Data recovery is the process of restoring data that has been lost, accidentally deleted, corrupted or made inaccessible for any reason. Data recovery typically refers to the restoration of data to a desktop, laptop, server, or external storage system from a backup. The purposes of Data Recovery are:

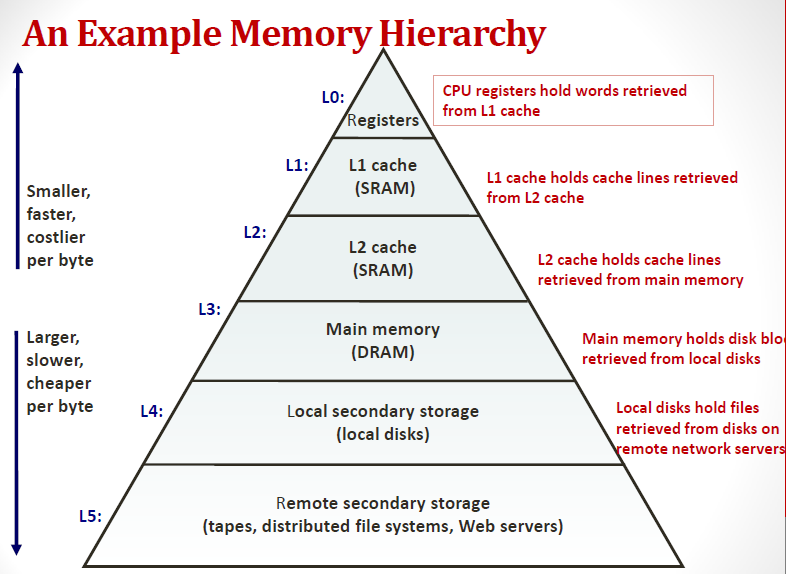
* Planning and testing responses to different kinds of failures
* Configuring the database environment for backup and recovery
* Setting up a backup schedule
* Monitoring the backup and recovery environment
* Troubleshooting backup problems
* Recovering from data loss if the need arises

**Types of Failure**

|  |  |
| --- | --- |
| Transaction | Caused by errors within the transaction processes. |
| System | Caused by failure of network or operating system or physical threats to the system as a whole. |
| Media | Failure of hard disk, out of memory errors, out of disk space errors. |
| Reasons for failure: |  |

* A System Crash
* Transaction or System Error
* Local errors or exception
* Concurrency Control
* Enforcement
* Disk Failures
* Physical Problems

**The storage Hierarchy**



**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. 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.

**Buffer Management**

Buffer management is a key component in achieving this efficiency. The buffer management component consists of two mechanisms: the buffer manager to access and update database pages, and the buffer cache (also called the buffer pool), to reduce database file I/O.

A buffer is an 8-KB page in memory, the same size as a data or index page. Thus, the buffer cache is divided into 8-KB pages. The buffer manager manages the functions for reading data or index pages from the database disk files into the buffer cache and writing modified pages back to disk. A page remains in the buffer cache until the buffer manager needs the buffer area to read in more data. Data is written back to disk only if it is modified. Data in the buffer cache can be modified multiple times before being written back to disk. For more information, see Reading Pages and Writing Pages.

When SQL Server starts, it computes the size of virtual address space for the buffer cache based on a number of parameters such as the amount of physical memory on the system, the configured number of maximum server threads, and various startup parameters. SQL Server reserves this computed amount of its process virtual address space (called the memory target) for the buffer cache, but it acquires (commits) only the required amount of physical memory for the current load.

The interval between SQL Server startup and when the buffer cache obtains its memory target is called ramp-up. During this time, read requests fill the buffers as needed. For example, a single-page read request fills a single buffer page. This means the ramp-up depends on the number and type of client requests. Ramp-up is expedited by transforming single-page read requests into aligned eight-page requests. This allows the ramp-up to finish much faster, especially on machines with a lot of memory.

**Transaction Log**

A transaction log is a file – integral part of every SQL Server database. It contains log records produced during the logging process in a SQL Server database. The transaction log is the most important component of a SQL Server database when it comes to the disaster recovery – however, it must be uncorrupted. After each database modification – transaction occurrence, a log record is written to the transaction log. All the changes are written sequentially.A transaction log stores every transaction made to a SQL Server database, except some which are minimally logged like BULK IMPORT or SELECT INTO. Internally it is split into the smaller parts called Virtual Log Files (VLFs). When one VLF becomes full logging continue to write into the next available in the transaction log. The transaction log file can be represented as a circular file. When the logging reaches the end of the file it starts again from the beginning, but only if all the requirements has been met and the inactive parts has been truncated. The truncation process is necessary to mark all inactive parts so they can be used again and overwritten

A log record is no longer needed in the transaction log if all of the following are true:

* The transaction of which it is part has committed
* The database pages it changed have all been written to disk by a checkpoint
* The log record is not needed for a backup (full, differential, or log)
* The log record is not needed for any feature that reads the log (such as database mirroring or replication)

Logical log is an active part of the transaction log. A Log Sequence Number (LSN) identifies every transaction in the transaction log. The MinLSN is the starting point of the oldest active transaction in the online transaction log.

**Data Updates**

**Data Caching**

**Transaction Rollback and Roll forward**

Rollback: The Rollback transaction is a transaction which rolls back the transaction to the beginning of the transaction. The transaction can be rolled back completely by specifying the transaction name in the Rollback statement or to cancel any changes to a database during current transaction. It is permissible to use before Commit transaction.

Roll forward: Recovering a database by applying different transactions that recorded in the database log files. It is nothing but re-doing the changes made by a transaction i.e. after the committed transaction and to over write the changed value again to ensure consistency.

**Check Pointing**

Checkpoint-Recovery is a common technique for imbuing a program or system with fault tolerant qualities, and grew from the ideas used in systems which employ transaction processing. It allows systems to recover after some fault interrupts the system, and causes the task to fail, or be aborted in some way. While many systems employ the technique to minimize lost processing time, it can be used more broadly to tolerate and recover from faults in a critical application or task. The basic idea behind checkpoint-recover is the saving and restoration of system state. By saving the current state of the system periodically or before critical code sections, it provides the baseline information needed for the restoration of lost state in the event of a system failure. While the cost of checkpoint-recovery can be high, by using techniques like memory exclusion, and by designing a system to have as small a critical state as possible may minimize the cost of check pointing enough to be useful in even cost sensitive embedded applications. This technique provides protection against the transient fault model. Unfortunately, it has only limited utility in the presence of a software design fault. Consider for instance a system which performs control calculations, one of which is to divide a temperature reading into some value.

**Recovery Schemes (WAL: Write Ahead Logging Protocol)**

**Failure with loss of Non-volatile storage (General Concepts)**

**Recovery in Multi-database System**