**Database Recovery**

A major responsibility of the database administrator is to prepare for the possibility of hardware, software, network, process, or system failure. If such a failure affects the operation of a database system, you must usually recover the database and return to normal operation as quickly as possible. Recovery should protect the database and associated users from unnecessary problems and avoid or reduce the possibility of having to duplicate work manually.

Recovery processes vary depending on the type of failure that occurred, the structures affected, and the type of recovery that you perform. If no files are lost or damaged, recovery may amount to no more than restarting an instance. If data has been lost, recovery requires additional steps.

**Purpose of data recovery**

* 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 failures
  + overflow, interrupt, data not available, explicit rollback, concurrency enforcement, programming errors
  + no memory loss.
* Process Failure
  + failure in a user, server, or background process of a database instance such as an abnormal disconnect or process termination
* Network Failure
  + disconnected from client workstations to database servers, or to several database servers to form a distributed database system
  + network failures such as aborted phone connections or network communication software failures can interrupt the normal operation of a database system
* System crashes
  + due to hardware or software errors
  + main memory content is lost
* Media failures
  + problems with disk head, unreadable media surface (parts of ) information on secondary storage may be lost
* Natural disasters
  + fire, flood, earthquakes, theft, etc.
  + physical loss of all information on all media

The storage hierarchy

Buffer management

**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

**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.

**Transaction rollback (Undo) and Roll forward**

The process of applying logged changes to data in a database to bring the data forward in time is known as rolling forward. The set of all data restored is called the roll forward set. A roll forward set is defined by restoring one or more full backups, such as a database or partial backup or a set of file backups. If a RESTORE statement specifies filegroups, files, or pages, only these items are included in the roll forward set. Otherwise, all files in the backup being restored are included in the roll forward set. If the full backup contains log records, the restored data will be rolled forward using this log.

After the redo phase has rolled forward all the log transactions, a database typically contains changes made by transactions that are uncommitted at the recovery point. This makes the rolled forward data transactionally inconsistent. The recovery process opens the transaction log to identify uncommitted transactions. Uncommitted transactions are undone by being rolled back, unless they hold locks that prevent other transactions from viewing transactionally inconsistent data. This step, is called the undo (or roll back) phase. If the data is already transactionally consistent at the start of the recovery process, the undo phase is skipped. After the database is transactionally consistent, recovery brings the database online.

After one or more backups have been restored, recovery typically includes both the redo and undo phases. Every full and differential backup contains enough transaction log records to allow for the data in that backup to be recovered to a self-consistent state.

Check pointing, shadow paging

Recovery schemes (WAL: Write Ahead Logging Protocol)

Failure with loss of non-volatile storage (General concept)

Recovery multidatabase system