## Chapter 6.1 Recoverable Operation Problem and Model

*Introduction:*

In this chapter, we discuss the possible problem and how DBMS help solve these problems and start from where, and discuss how to deal with these malfunctions.

*At first, we mainly discuss ‘System Malfunction’ or ‘Crash’, and the design of Logging and Recovery is used to recover these kind of errors. Also we will introduce the Buffer Management Model, it is the fundamental of the DBMS Recovery.* Also in the next chapter, we still need this model for several transactions to access the Database.

### Chapter 6.1.1 Malfunction Model

When the database has been visited or updated, there would has a lot of problems. The problem’s range from the wrong input error data from Keyboard to Explosion which happened in the room where the Disk stored.

*Wrong Data Input*

Some wrong data input would not be detected. What we need to do is to write the constraint and trigger to find out the wrong data input.

Medium Malfunction

* *Partial Malfunction* - one or several bytes malfunction in the disk, and normally *Parity Check* can be used to check this issue.
* *Disk Head Malfunction* - the whole Disk can not be accessed. Then there have one or two methods to solve:
* Using *RAID model, then lost Disk can be recovered*.
* *Maintain one Backup*, which is to say that there would have one copy on tape or disk. *Create Backup periodically*, just completely or incremental, stored in safety distance far from Database.
* We can not use the Backup method, but to *save the redundancy copy online*. These copy can be distributed on several points. Later, we will discuss how to maintain the database consistency.

Disaster Malfunction

This kind of malfunction includes several situations that range from the location explosion or fire or malicious damage to DBMS.

Under such situation, the RAID Model can not provide any help, since all Data Disk and Parity Check would lose all their usage. But, other methods such as *Backup, Redundancy and Distributed Copy can be used to prevent the Disaster Malfunction.*

System Malfunction

*Transaction:*

The processor to Query and Database Modification are called transaction. Transactions can be just as all other function procedures to execute a series of steps, normally, some of which are used to modify the database.

*Each Transaction includes the status which stands for the specific step that the transaction has processed, and there have several things included in the status:*

* *The current location that transaction executes.*
* *All temporary variables which are needed later.*

*System Malfunction is the main reason that causes the loss of status.* Typical system Malfunctions are *Power Loss* and *Software Error*. Since the main memory is Volatile Storage, then Power Loss would cause the loss of all contents in main memory, the result that saved in the main memory for Transaction Steps have all disappeared, however this situation is totally different from the disk.

*However, the similar situation can also happen when there has software error that may cover parts of content in the main memory, and it may include the value of Program Status.*

When the main memory lost, then the status of the transaction is also lost, which means that we can not make sure which part of Transaction has been finished. Also re-do all transactions can not repair the problem.

*Example:*

When the value in the database should add by 1 by using the transaction, but we do not know whether we need to repeat the process.

The solution is to update all things by using the divorced, and non-volatile logging to update the database, and recover the database if necessary. But we need to ensure that the logging would be recorded under the non-distributed method, it is very complex, which would be checked in the future.

### Chapter 6.1.2 Further Discussion about Transaction

### Chapter 6.1.3 Right Execution about Transaction

### Chapter 6.1.4 Primitive Operation about Transaction