Core Technology for Handling Large Amounts of Data

Hong Taeha
SungKyunKwan University
Software department
2014313303
010-8011-4229
xogk369@hanmail.net

ABSTRACT
This paper describes what is needed to process large amounts of data and describes the features of the Database Management System that satisfy those requirements. It also discusses the various core technologies that are required to create DBMS.

Keywords
Requirements in large data, DBMS, data Independence, query language, transaction, concurrency control, recovery

1. INTRODUCTION
In our society, information has become a tremendous asset as the amount of available information has been overflowing. Accordingly, there is a need for technologies that collect a large amount of information, find meaningful value in the information, and manage large amounts of information.

However, in order to manage and analyze large amounts of data, some problems must be overcome. As more data grows, more and more users are using the data, and these users have to solve problems that occur when accessing data at the same time. Also, security issues must be solved to handle large amounts of data.

So how can you manage large amounts of data? You may use file system, but it has many problems. We will discuss this issue later. On the other hand, DBMS is effective in overcoming problems of file system and managing large amount of data. Therefore, this paper focuses on the points and limitations for managing large volumes of data, and the characteristics of DBMSs that overcome these limitations.

2. REQUIREMENTS IN LARGE DATA
There are several necessary conditions for handling large amounts of data.

Usually, large amounts of data are stored on storage devices such as disks or tapes. And fetches the relevant part needed for processing from that storage device to main memory. However, 32-bit computers can retrieve up to 4 gigabytes of data, so it takes a lot of time to access the data. Therefore, there is a need for a technique for accessing and referring to necessary data at a time.

It is also important to solve the data concurrency problem. Various users access data at the same time. If data concurrency is not maintained, there is a problem that invalid data generated during modification of data is used. To handle large amounts of data, you should be able to avoid this problem.

Security issues are also important. Some information should be restricted and protected by users. Therefore, access rights management and security problems must be managed flexibly.

As a result, it should be easy to access and use large amounts of data quickly, and be able to control the simultaneous access of many users. In addition, security issues and user access rights should be managed.

Table 1. Difference of File System and DBMS.

Requirements	File System	DBMS
Data access	It is slow because it has to scan several times.	It is fast because you need to access and reference the data you need.
Data concurrency	Implementation is very complex.	Maintain data independence
Security	Operating systems provide only a password mechanism for security. So it is not suitable.	It protects the security by checking in advance before the user accesses the data.
Recovery	weak.	Protect users from the effects of system failure.

2.1 The limitation of file system
Table 1 shows the difference between file system and DBMS. This paper will focus on these two differences in the future.

We can try to store and manage data in an operate system file. However, this approach has many problems.

The file system scans a limited amount of data stored on storage devices such as disks and tapes. For 32-bit computers, 4 gigabytes is the maximum. If you scan the data in this way, you will not be able to scan the data at once, and it takes longer than the method to scan and access the data with one step at a time.

And in the file system, you need to write a special program that processes the data that users request. But it is very complicated to implement this program because the amount of data is very large. It also becomes difficult to solve the data concurrency problem.

There is also a problem with security. Because the operating system only provides a password mechanism, it is not flexible enough to manage the security of the data and the users' access rights.

The file system is also vulnerable to the recovery of corrupted data when a system crash occurs.

3. OVERCOMING LIMITATIONS
As you can see in 2.1, the file system is vulnerable to handling large amounts of data. I will describe the features I need to

overcome this vulnerability and meet the requirements of handling large amounts of data.

3.1 DBMS concept

There are many benefits to handling data using a DBMS.

Data Independence ensures that applications do not expose detailed information about data representation and storage. So the DBMS hides the details of the data and provides an abstract look. This means that even if certain data is modified, it does not affect other data or other schemes.

It also uses sophisticated technologies to efficiently retrieve and store data. Therefore, it can reduce the data processing time by allowing the user to access the data and refer to the data instead of reading only a part of the data in the file system.

The DBMS can check in advance whether the information is valid before several users access the information at the same time. DBMS offers methods to impose constraints while entering data into the database and retrieving the same at a later stage. DBMS offers many different levels of security features, which enables multiple users to have different views with different features. Pre-validation ensures data integrity and security.

And the DBMS solves the problems that occur when multiple users access the information at the same time through the transaction and also handles the recovery part. Transaction will be discussed in more detail later.

3.2 Data Independence and Query language

Physical schema pertains to the actual storage of data and its form of storage like files, indices, etc. It defines how the data will be stored in a secondary storage. Logical schema defines all the logical constraints that need to be applied on the data stored. It defines tables, views, and integrity constraints. Figure 1 shows examples of physical and logical schema structures.

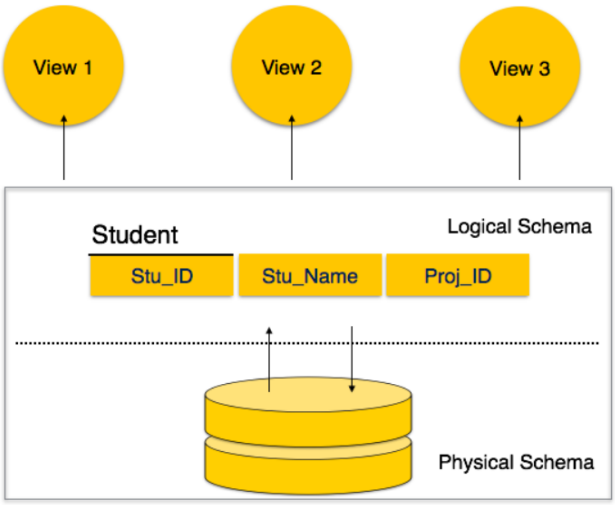

Figure 1. Examples of physical and logical schema structures

Data independence is a technique for separating user schema, logical schema, and physical schema and managing each schema independently, so that even if a particular schema is modified, it does not affect other schemas. This avoids the problem of other data being damaged due to modification of specific data, and is advantageous in terms of performance, which is an essential feature of a database that manages a large amount of data.

DBMS is equipped with query language, which makes it more efficient to retrieve and manipulate data. A user can apply as many and as different filtering options as required to retrieve a set of data. Traditionally it was not possible where file-processing system was used. The query is simply sending a request. The query statement is a kind of high-level language that passes these requests. The query is easy to learn because it has a format similar to a normal English sentence. In addition, through the structure of the query, it is possible to efficiently manage a large amount of data, thereby making it possible to easily access a large amount of data.

That is, the independence of the data and the query statement have a common advantage in that they improve the performance by efficiently managing the data.

3.3 Transaction, Concurrency Control, Recovery

A transaction is a unit of work that is a collection of statements that must be processed as a group within the database. A unit of work enclosed in a transaction must be completely executed, and if none of them is done, and if any of them fails, the whole is canceled. The database system guarantees atomicity, consistency, isolation, and durability for each transaction.

Atomicity is all done, or if one fails, all should fail. Therefore, if a failure occurs during the execution of a transaction, the operation must be canceled.

Consistency means that a consistent database state must be maintained when a transaction is successfully completed.

Isolation helps transactions to be performed independently of each other. When multiple users request access to the same data at the same time, order the transaction so that the result is processed sequentially. Another transaction waits until the end of a transaction.

Durability means that the result of a transaction must be permanent if the transaction completes successfully. If the result is successfully processed, the resultant value should not be carried.

These features of the transaction help manage large amounts of data and allow you to solve problems that arise from managing large amounts of data.

Concurrency control is a control to overcome the problems that arise when two or more transactions are executed. This is essential for the database to maintain the consistency of data and to efficiently manage the value of erroneous data resulting from various user accesses.

Recovery refers to recovering data corruption that occurs when a problem such as a system crash occurs Recovery techniques are essential because unexpected data corruption occurs where critical data is managed.

4. REFERENCES

Ramakrishnan, Gehrke, 2003, Database Management System