3 Database Management

Chapter Objectives

When actually using a database, administrative processes maintaining data integrity and security, recovery from failures, etc. are required. A database management system (DBMS) is software to perform these processes for the users.
In this chapter, we will learn about the overview, types, characteristics and functions of database management systems.

① Understanding functions and characteristics of database management systems to use databases efficiently.
② Understanding characteristics of various databases (DBMSs) such as RDB, OODB, ORDB and multimedia database.
③ Understanding differences between a centralized database and a distributed database and those functions such as commitment control necessary which are required to run a distributed database.

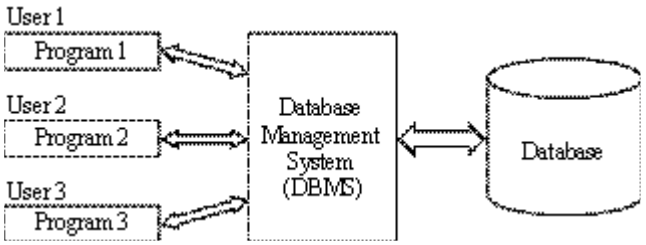
3.1 Functions and Characteristics of Database Management System (DBMS)

Even if data is integrated based on the hierarchical, network, or relational data model and stored in storage media such as magnetic disks as a database, it cannot be operated as a database system. To efficiently operate a database, which has complex data structures, dedicated database management software is needed.

3.1.1 Roles of DBMS

A database management system (DBMS) is software placed between users (programs) and a database to manage data.

Figure 3-1-1
Database Management System



(1) Roles required for a DBMS

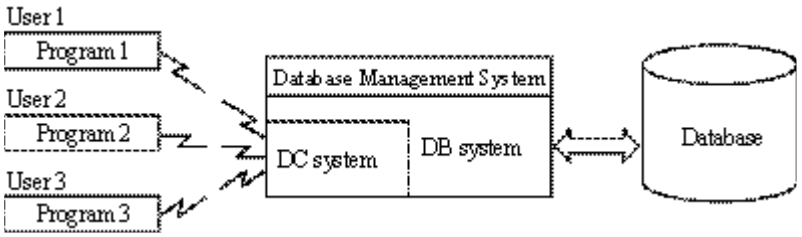
The following roles are required for a DBMS:

- Definition of databases
- Efficient use of data
- Sharing of databases
- Measures against database failures
- Protection of database security
- Provision of languages accessible to a database

(2) DB/DC system (database/data communication system)

Many terminals gain access to a database on a mainframe computer. To operate a database management system on an online system, the database (DB) and data communication (DC) must function in unity. This is called a DB/DC system (Figure 3-1-2). IMS (Information Management System) of IBM is a representative DB/DC system.

Figure 3-1-2 DB/DC System



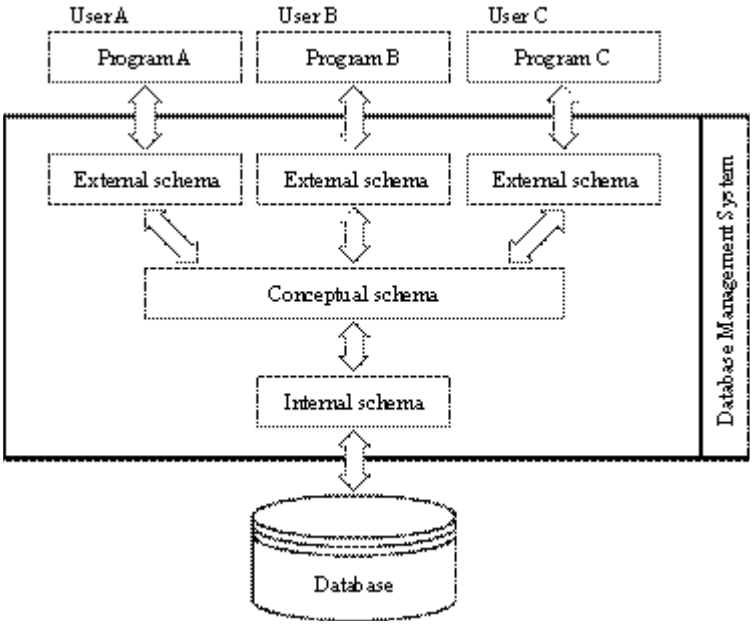
3.1.2 Functions of DBMS

Many DBMSs have been made public so far. In this section, taking a DBMS defined by ANSI-SPARC as an example, its functions are explained.

(1) Database definition functions

For a DBMS, the external schema, the conceptual schema and the internal schema are defined according to the 3-tier schema.

Figure 3-1-3
3-tier Schema of ANSI-SPARC



① Conceptual schema (in CODASYL, called 'schema')

In the conceptual schema, information on records, characteristics of fields, information on keys used to identify records and database names etc. are defined. The logical structure and contents of a database are described in this schema.

② External schema (in CODASYL, called 'subschema')

In the external schema, database information required by an individual user's program is defined. This contains definitions on only those records which are used in the program and their relationships extracted from the database defined in the conceptual schema.

③ Internal schema (in CODASYL, called 'storage schema')

In the internal schema, information concerning storage areas and data organization methods on the storage devices are defined.

Each of these schemata is defined in a database language, DDL (Data Definition Language). Data items such as attributes and names of the described data are called meta-data and meta-data described in each schema is managed by a data dictionary (Data Dictionary/Directory; DD/D). The DD/D consists of a data dictionary in the user-oriented information format and a data directory translated for use by computers.

(2) Database manipulation functions

The functions for users' manipulating databases are written in a DML (Data Manipulation Language), a database language. Concrete contents of database manipulation by users are described in DML and there are three description methods as follows:

① Host language system

The host language system is a system to describe and manipulate a database in a procedural programming language. In the host language system, by extending functions by adding database manipulation commands to the languages such as COBOL, FORTRAN, and PL/I, databases can be processed in the same system as by traditional programming. To operate databases in the host language system, comprehensive knowledge and engineering skill of programming languages and databases are required.

② Self-contained system

The self-contained system is a system using a language uniquely prepared for a specific DBMS. In this system, interactive database operations with the DBMS are performed. While procedures inherent in the system can be easily described, non-routine procedures cannot be described.

③ Query system

The query system is also called a command system and commands are inputted in this case. This system is designed for the non-procedural use of a database by end users.

(3) Database control functions

Among DBMS functions, aforementioned database definition functions and database manipulation functions are basic functions for application programs (as users of a database) to gain access to data and schemata. Furthermore, the following functions are required for a DBMS:

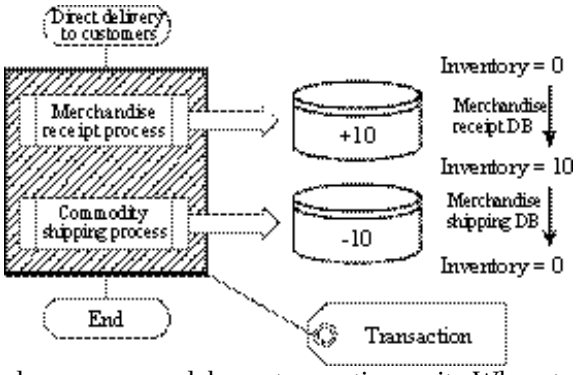
- A function to facilitate the development and maintenance of application programs
- A function to maintain data integrity
- A function to improve data reliability, availability, and security
- A function to maintain appropriate efficiency of processing

More specifically, the following functions are used to realize the above functions:

① Transaction management

A unit of processing from a user's point of view, including database reference and update processing is called a transaction. For example, some trading firms directly deliver some merchandise from suppliers to customers, without keeping in-house inventories. In this case, the receipt and the shipping of merchandise occur at the same time and the same operations are performed also in the inventory management system. If only one of the receipt/shipping operations is performed by a failure in the inventory management database, the actual number of merchandises and the number in the inventory management system will be inconsistent. The correct result can be gained only when both receipt/shipping processes are normally performed. Therefore, in this case, a combination of receipt and shipping processes is considered as a meaningful process, that is, a transaction.

Figure 3-1-4
Transaction Management



The update of a database is always managed by a transaction unit. When transaction processing is normally completed, receipt/shipping processing is also regarded as having been normally completed and the database update is executed. But, if transaction processing stops abnormally, it is not regarded as having been normally completed and the state before processing is restored. Ensuring update is called 'commit process,' and restoring the original state is called 'rollback process. '

② User view function

The external schema is also called a view. Therefore, as previously mentioned, a view is created by extracting a part of the conceptual schema. In a relational database, a view is defined by the SQL statement.
A table is an actual table, and it is stored in the auxiliary storage device. A view, however, is a virtual table created from the actual source table on a case-by-case basis by the execution of the SQL statement and is an abstract entity. Views, generally created by join operations, cannot be updated.
A view has the following roles in database control:

- To achieve logical data independence
- To improve security
- To increase efficiency in application program development

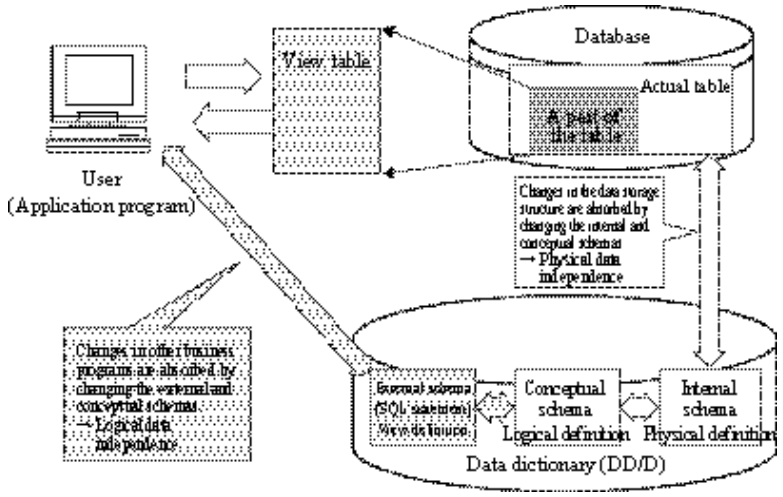
3.1.3 Characteristics of DBMS

By using a DBMS, users can use a database without paying much attention to its structure.
In this section, the characteristics of a DBMS are explained.

(1) Achievement of data independence

One of the purposes of using a database is "independence of data from a program." This is achieved by the 3-tier schema. Data independence is classified into the physical data independence and the logical data independence.

Figure 3-1-5
Data independence



① Physical data independence

When data is not affected by changes of physical data structure and magnetic disk devices, this characteristic is called the physical data independence. In this case, even if the internal and conceptual schemata are modified, the modification of application programs is not required.

② Logical data independence

When logically extraneous data is not affected even if other application programs are changed, the characteristic is called the logical data independence. In this case, even if the external and conceptual schemata are modified, the modification of data is not required.

Thus, the independence of the data shared by users' application programs enables users to create programs without paying much attention to the data storage structures and increases flexibility in programming. Database administrators can also modify databases flexibly without taking users' programs into account.

(2) Database access

In a database system, programs do not directly gain access to the data, but all access operations are performed through a DBMS. In a relational database, for example, data access is performed by the execution of the SQL statement. A database system must respond to access from multiple users, including permission and denial of access. Because such actions are complicated, when a failure occurs, many users can be affected. Therefore, fast failure recovery is essential.
To satisfy these requirements, a DBMS provides the concurrent execution control for simultaneous access from multiple users, the failure recovery and the access privilege control for security.

① Concurrent execution control (exclusive lock management)

To respond to access from multiple users, simultaneous writing to and reading from the same database by multiple users must be reflected in the database without contradiction. The function to realize this is called the concurrent execution control or the exclusive control.

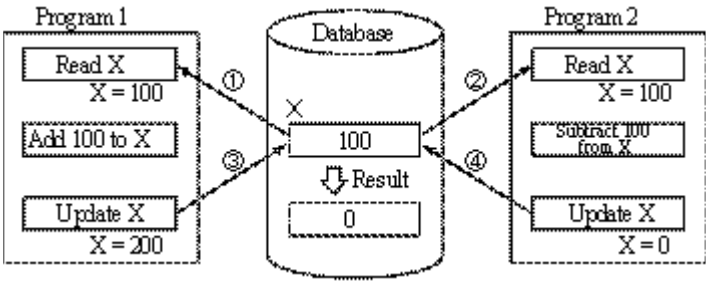
a. Mechanism of concurrent execution control (exclusive lock management)

Figure 3-1-6 shows the simultaneous access to the same data X in a database by programs 1 and 2.

① Program 1 reads data X in the database. The value of X is 100.
② Program 2 reads data X in the database. The value of X is also 100.
③ Program 1 adds 100 to the value of data X and writes the result 200 in the database.
④ Program 2 subtracts 100 from the value of data X, and writes the result 0 in the database.

If the processing is performed in the order of ①, ②, ③, and ④, the value of data X in the database becomes 0.

Figure 3-1-6
When the database does not have the concurrent execution control (exclusive control):



As stated above, when multiple programs gain access to one data item almost at the same time and try to update its contents, they may not be able to gain the correct results. The mechanism to prevent this phenomenon is the concurrent execution control (exclusive control).
In a DBMS, "lock" is used to perform this concurrent execution control (exclusive control). When multiple users gain access to the same data, the concurrent execution control (exclusive control) is performed in a DBMS as follows:

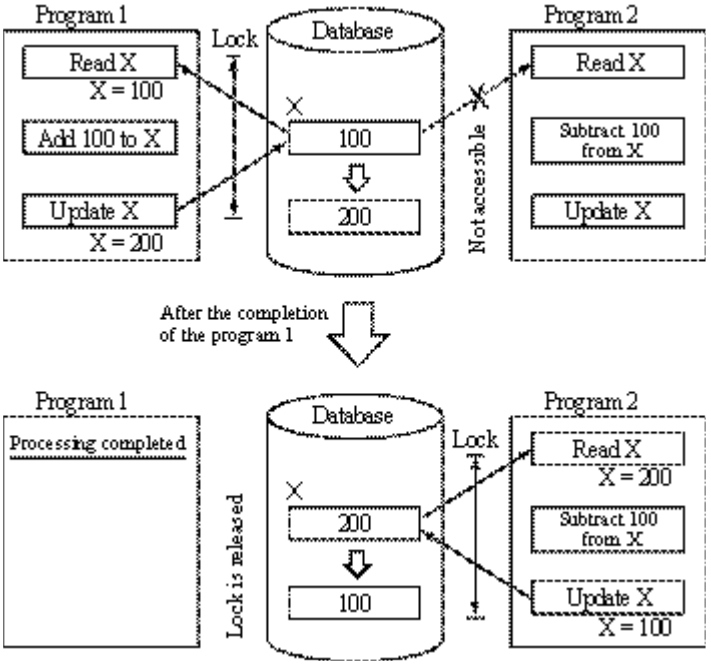
- Until the processing of the user who accessed the database first has been finished, hold the next user's access (this is called the lock).

- When the processing of the first user has been completed, release the lock.
- After confirming the release of the lock, accept the access from the next user.

Figure 3-1-7 shows an example of the concurrent execution control (exclusive lock management) function in a DBMS. The procedures are as follows:

① Program 1 gains access to data X and locks it at the same time to prevent access from program 2.
② After program 1 has completed its processing, program 2 gains access to data X to perform processing.
③ After the execution of programs, the result becomes 100.

Figure 3-1-7
When the database has the concurrent execution control (exclusive control):



This concurrent execution control (exclusive lock management), however, might produce another problem. That is the deadlock explained below.

b. Deadlock

In most DBMSs, the concurrent execution control (exclusive lock management) is performed for simultaneous access to a database. However, by using the lock of this control execution control (exclusive control), the phenomenon shown in Figure 3-1-8 may occur.

Figure 3-1-8
Deadlock

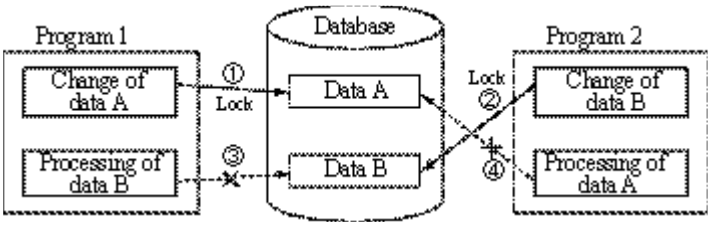


Figure 3-1-8 shows the simultaneous access to data A and B by programs 1 and 2.

① Program 1 gains access to data A.
② Program 2 gains access to data B.
③ Program 1 tried to access data B after accessing data A. But, data B is locked because it has already been accessed by program 2.
④ Program 2 tried to access data A after accessing data B. But, data A is locked because it has already been accessed by program 1.

Thus, the state in which both programs 1 and 2 cannot perform their processing and are locked in a waiting state of the completion of each other's processing is called the deadlock.

To prevent the deadlock, the following controls are performed in a DBMS:

- Regular monitoring of the occurrence of the waiting state of programs.
- When programs are in the deadlock state, the program that started processing later is forced to suspend its processing so that the program that first started processing can continue its processing by priority.
- After the program that first started processing has completed its processing, allow the program that started processing later to perform its processing.

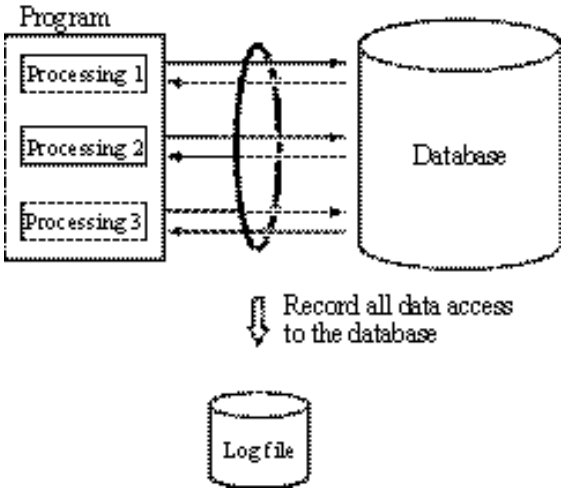
② Failure recovery

When a failure occurs in a database, the computer stops its processing and online transaction processing stops. Because important data indispensable to business activities are recorded in a database, failure prevention and fast failure recovery are essential for database availability.

a. Log file

A database management system prepares a log file to record processes including errors and each update of data in a time series. When a failure occurs in a database, the log file is used (Figure 3-1-9). A log file is also called a journal file or a journal log.

Figure 3-1-9
Log File



b. Rollback processing and roll forward processing

When a failure occurs in a database, there are two recovery methods: the rollback processing and the rollforward processing.

- Rollback processing

When a failure occurs in an operating system or a DBMS, restructure the database in the most recent recoverable state and restore the database before the point of failure by rewriting the contents using the images of the log file. Generally, this processing is automatically performed by the DBMS.

- Rollforward processing

If the disk storing the database is physically damaged, restore the contents of the database at the point of failure by reading the updated process images in the log file sequentially from the backup file.

③ Security

A database storing important and confidential data is accessed by many programs and interactive data manipulations, security to protect information is important.
Actually, security protection is performed not only by a DBMS, but also by software, hardware, and human efforts.
To protect disks on which a database is stored, a DBMS performs file access control and prevents unauthorized access to specific databases by users. It controls access privileges using user IDs, passwords, and their combinations, and encrypts data against data leakage to third parties.

(3) ACID characteristics

To protect a database, all database operations during transaction processing must have the following characteristics:

① Atomicity

A transaction must have the following characteristics:
- Normally complete all data operations included within a transaction processing.
- If only part of a transaction has been completed, the whole transaction processes have to be cancelled.

That means, a transaction has no option other than commit or rollback, and termination in the halfway state is not permitted.
The characteristic satisfying these requirements is the atomicity.

② Consistency

A transaction must be processed by the reliable program. Data manipulation by a transaction must be correctly performed without contradiction. After starting a transaction, the system must be maintained in the normal state.
The characteristic satisfying these requirements is the consistency.

③ Isolation

A transaction must not be affected by the processing results of other transactions. Even when being processed in parallel, transactions must not interfere with each other. In other words, the results of parallel processing and individual processing must be the same.
The characteristic satisfying these requirements is the isolation. The isolation is also called the independence.

④ Durability

When a transaction is normally completed, the state of the transaction must be maintained even if a failure occurs afterwards. That means, once a transaction has successfully ended, the state must be by all means maintained.
The characteristic satisfying these requirements is the durability. The durability is also called 'persistence.'

3.1.4 Types of DBMS

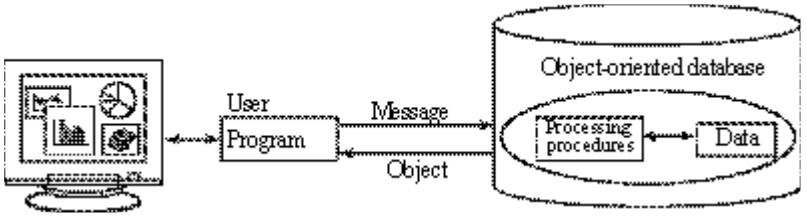
(1) RDB (Relational Database)

The database mentioned in 1.2 is called the relational database (RDB). Since the user of an RDB does not require a knowledge of specific computers, an RDB is employed for most of the current database software for personal computers.
The RDB is built on a mathematical foundation and its data structure, semantic constraints and data manipulation are logically systematized. An RDB consists of a set of simple two-dimensional tables and its smallest data unit is a character or a numeric value. Therefore, its structure is very simple and easy to understand. In addition, because its data manipulation is performed based on declarative manipulation using relational algebra, instead of the path-tracking method, it can provide high-level data control languages.

(2) OODB (Object Oriented Database)

While the relational database handles character data and numeric data, the object-oriented database (OODB) enables the efficient processing of complex data such as multimedia data (Figure 3-1-10). An integrated (encapsulated) set of data and processing procedures is called an object. In the OODB, objects are recorded and managed in magnetic disks.

Figure 3-1-10 OODB



In addition to basic manipulations such as query and update, persistent data integrity and failure recovery capabilities are included in processing procedures. Since objects are highly independent of each other, application programs can be built by assembling objects. User access to the object data is performed by sending messages in the predefined format.

(3) ORDB (Object Relational Database)

The object relational database (ORDB) is a database inheriting the data model and the data manipulation method of the RDB and including object-oriented features. An ORDB can handle abstract data type as well as numeric values and character strings handled in an RDB. The ORDB is a database adopting object-oriented features and inheriting the advantages of database management functions of the traditional RDB.
The ORDB employs SQL3, currently being standardized by ISO as the next version of SQL, as its database language. Some RDB products already put into practical use had begun to adopt object-oriented features before the announcement of SQL3.

(4) NDB (Network Database)

The network database mentioned in Section 1.2 is called NDB. Since knowledge about specific computers is required to use an NDB, it is mainly used for operational systems handling routine works. Compared to the hierarchical database, the NDB can create flexible structures such as cycles (closed paths) and loops (by setting itself as its parent) without being limited to vertical relations. However, the difficulty of having access beyond processing paths have been the challenging issue.

(5) Multimedia database

So far, the data mainly handled by databases are characters and numeric values. However, in response to the multimedia era, the multimedia database is designed to handle such data as video and audio in addition to characters and numeric values.
A multimedia database generally uses an object-oriented approach to provide a uniform user interface without making users conscious of the data structure of the media.
The following features are required for the multimedia database management system:

- Handling of a complex large data structure
  A DBMS can define the data structure by itself, and can perform queries and partial changes according to the structure.
- Time-related data operations and search
  A DBMS achieves such variable speed controls as fast-forwarding, slow-motion, and stop-motion in reproduction of video and audio data.

(6) Hypertext database

The hypertext database can handle complex data structures that cannot be expressed by the traditional structural databases and relational databases. A hypertext is a group of nodes that are linked together to express a set of related pieces of information. The hypertext database is designed by fitting these hypertexts into a database in the network data model structure.

The hypertext database enables the successive use of related databases such as searching for a new data item based on a search result. For example, it is suitable for the search of a homepage on the Internet.

In contrast to the hypertext database that can only search character information, the database that can search data including audio and video as well as characters is called the hypermedia database.

3.2 Distributed Database

3.2.1 Characteristics of Distributed Database

Originally, the purpose of a database was to achieve a central control by centralizing data. Although the idea of distributed database seems to conflict with this original purpose, it is not true. Even when physically (geographically) distributed, if the data are logically centralized and under centralized control, the original purpose can be accomplished. Network technology has enabled this centralization. Using networks, a company headquarters can do centralized control of databases distributed to its branch offices. Therefore, network technology is indispensable to realize a distributed database. In this section, the advantages and problems of a distributed database are explained.

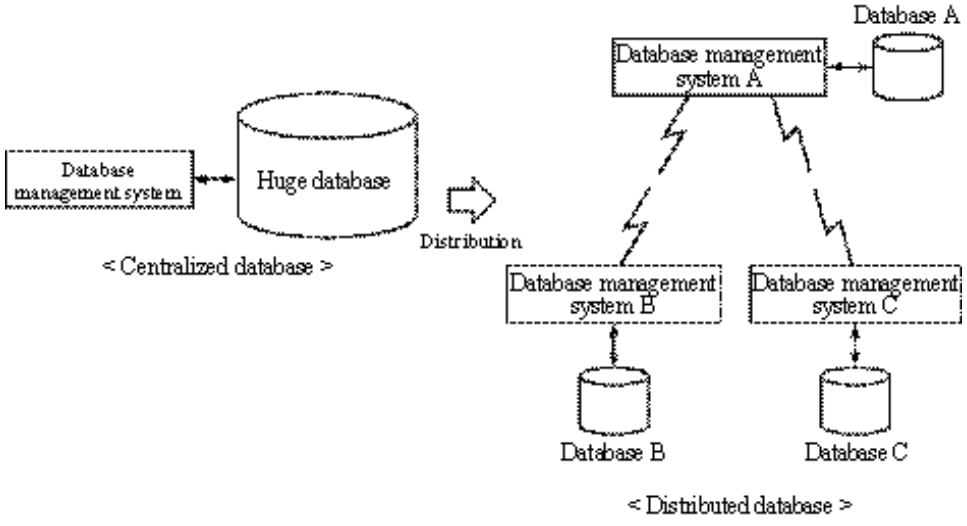
The centralized database created by gathering data used to be the major traditional database because it reduced the costs of system development, maintenance, and operation management.

The centralized database, however, has the following problems:

- A database failure affects the whole system
- Slow response to demands from a specific department
- High data communication costs due to central processing of data through communication lines
- Increase in costs and personnel to maintain a huge database

To solve these problems, a distributed database that enables the use of multiple databases as one database has been developed.

Figure 3-2-1
Distributed Database



<Advantages of a distributed database>

- Users in each department can perform query and editing of necessary information by themselves with simple operations.
- Better adaptability to changing business environments
- Due to independent processing by each department, the requirements of each department can be directly reflected into the system.
- Because databases are located in each work place, a quick response is possible.
- Even if a failure occurs in a database, other databases are available and the risks can be distributed.
- Users can access other databases without having to consider the location of the databases.

<Problems of a distributed database>

- Administrative management such as security and password controls is difficult.

- Because databases are distributed, duplicate data cannot be completely eliminated and databases can contradict each other.
- Due to the data distribution, programs can also be distributed.
- Due to the addition of department-specific functions, the version control of all the database programs becomes difficult.
- Because programs are developed on a department or individual basis, similar programs can be redundantly created.
- When company-wide processing is performed, larger amounts of time and cost are required for data communication.
- Batch processing is difficult.

In spite of the advantages and disadvantages mentioned above, the distributed database is rapidly becoming prevalent due to the increased performance and lower pricing of personal computers and development of communication networks.

3.2.2 Structure of Distributed Database

Figures 3-2-2 and 3-2-3 show the structures of a traditional centralized database and a general distributed database.

Figure 3-2-2 Centralized database

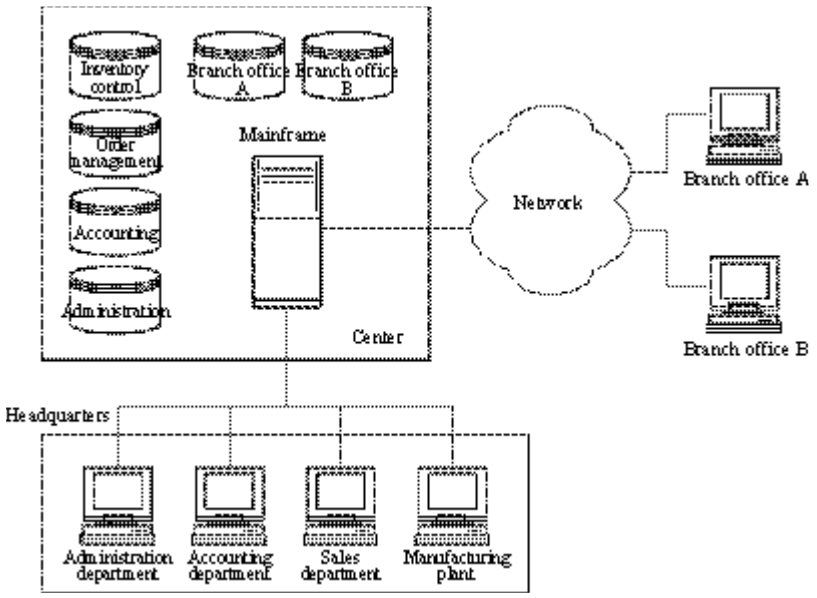
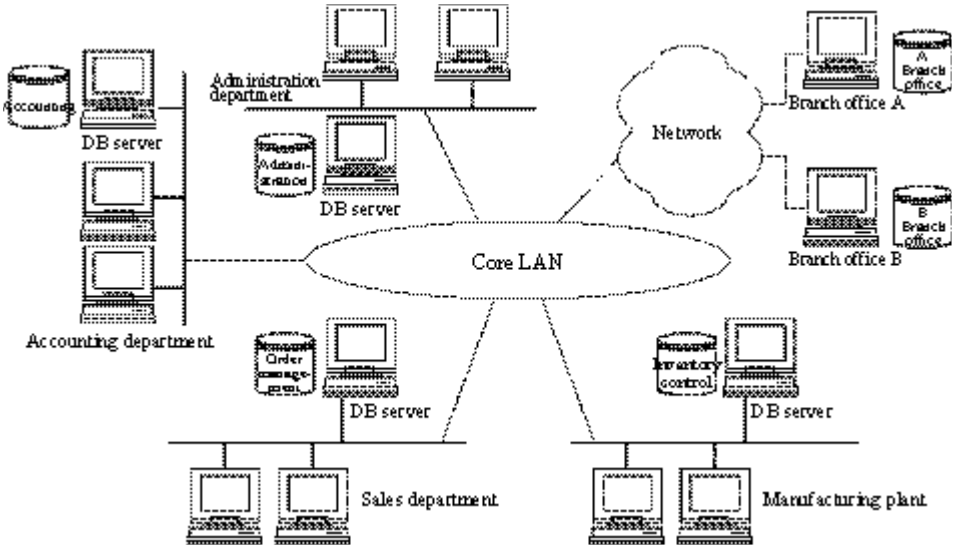


Figure 3-2-3 Distributed Database



These figures are examples using database servers (DB servers). The DB server is a computer that provides database functions for multiple clients (users). Due to the centralized control of database operations, it is possible to maintain the confidentiality of data.

3.2.3 Client Cache

In a distributed database, the amount of data transferred between DB servers and clients could be a problem. To solve this problem, the client cache is used.
In this system, when a client gains access to the database, the cache is used. If necessary data exist in the cache, data transfer from the DB server is not necessary and can reduce the amount of data traffic.
When using the client cache, note the following points:

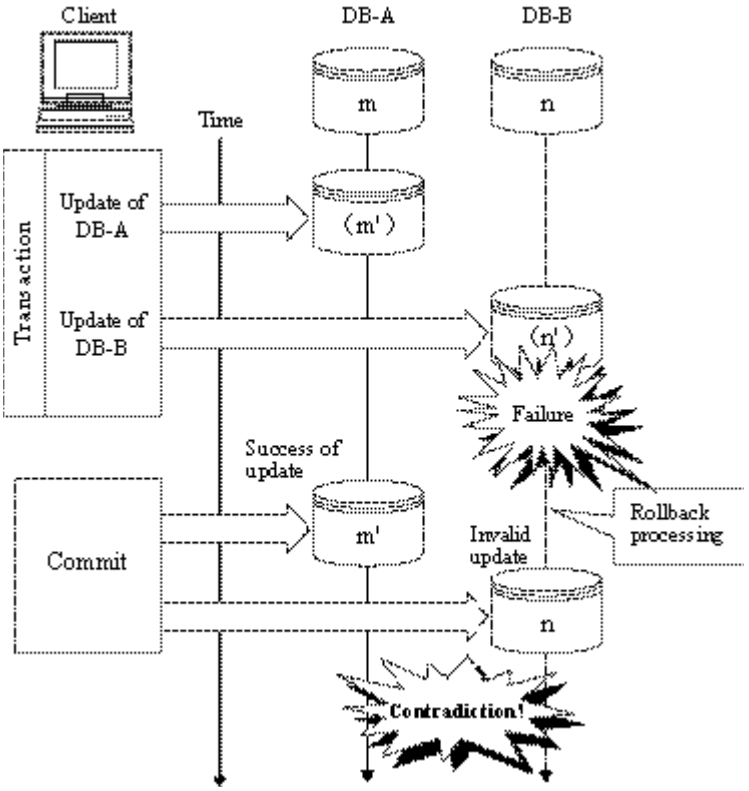
- Contents of the cache among multiple clients and DB servers must be automatically managed to maintain coherency.
- Concurrent execution control between transactions executed on different clients must be performed.

3.2.4 Commitment

(1) 2-phase commitment control

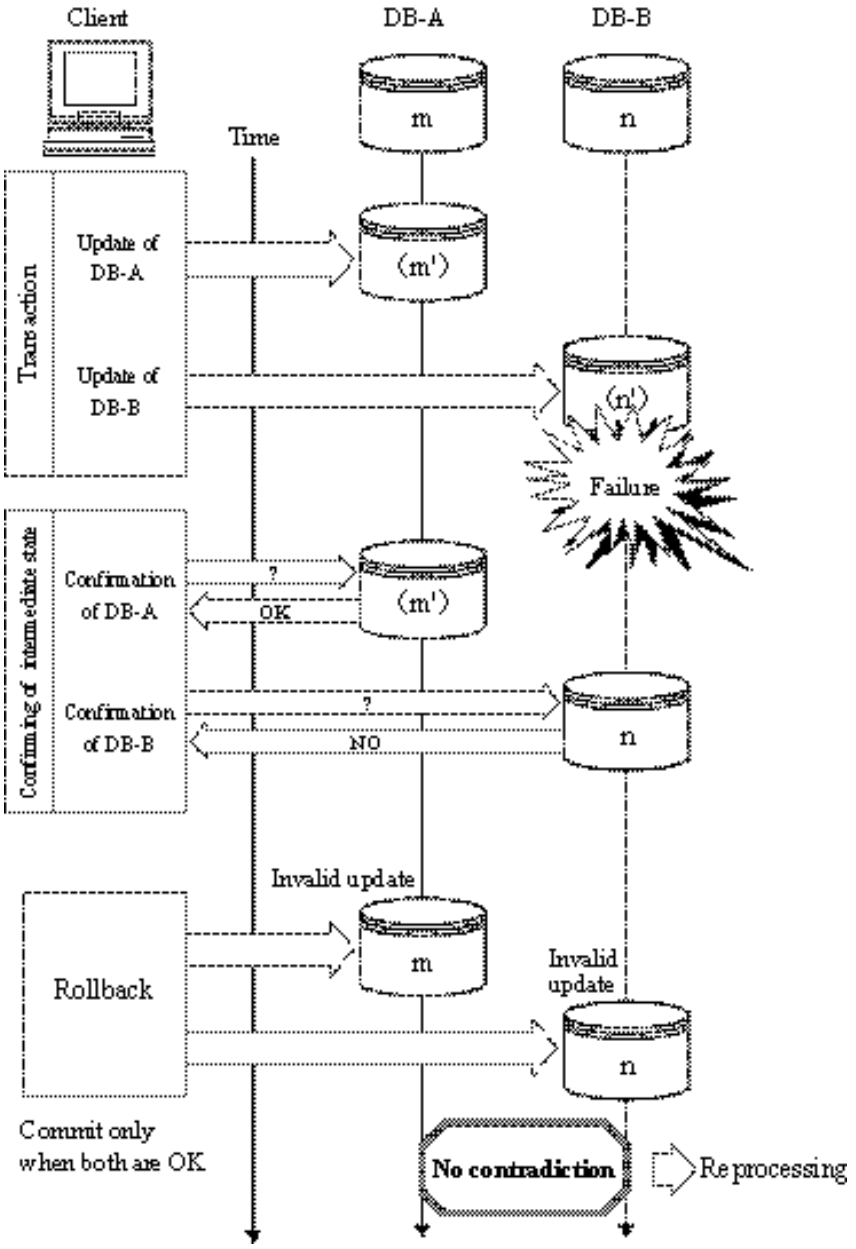
In a centralized database, the data integrity during transaction processing is maintained by controlling commitment and rollback. On the other hand, in a distributed database, because multiple databases are updated by transaction processing from the client, the following problems occur.
As Figure 3-2-4 shows, as a result of transaction processing from the client, commitment processing is performed against DB-A and DB-B based on the commitment request. When processing in DB-A is normally completed and processing in DB-B is abnormally terminated, the integrity of update processing is lost and the contents of the databases contradict each other.

Figure 3-2-4
1-Phase Commitment



Consequently, processing should be performed by the following two steps so as not to accept the results of transaction processing immediately. In the first step, secure an intermediate state (secure state) where both completion of process and rollback can be carried out and in the second step, perform commitment processing. This is called the 2-phase commitment control (Figure 3-2-5).

Figure 3-2-5
2-Phase Commitment



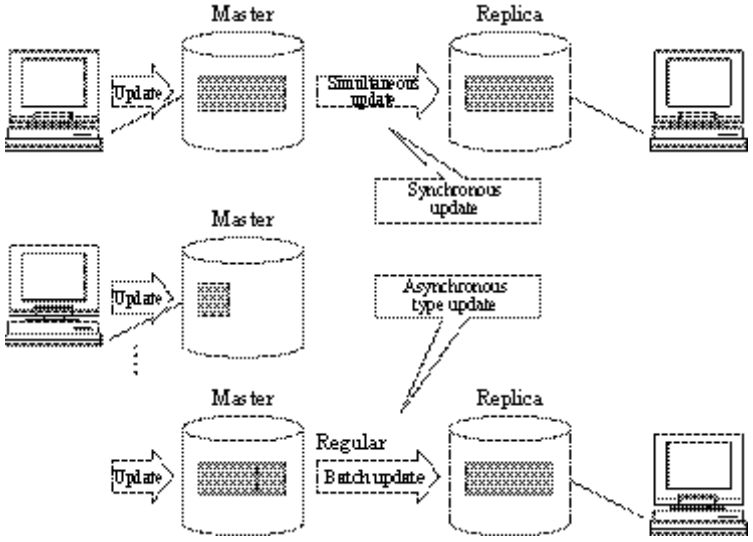
(2) 3-phase commitment control

In the case of 2-phase commitment control, failures are dealt with by having a secure state before commitment processing. However, this is not a complete measure because it cannot deal with failures that occurred during commitment processing.
In the 3-phase commitment control, another processing called pre-commitment processing is set between the secure and commitment states. If either of the databases fail in pre-commitment, rollback processing is conducted against all databases to maintain data integrity. Therefore, the 3-phase commitment control provides higher reliability than the 2-phase commitment control.

3.2.5 Replication

In a distributed database, transaction processing is performed by regarding multiple databases as one database. In the systems in which immediacy is required, real-time processing is performed by the above-mentioned 2-phase commitment control and 3-phase commitment control. On the contrary, in the systems in which immediacy is not so much required, replications of the database are made in the local servers at branch offices, departments, etc., and the burden of data traffic is lowered by using them. The replicated table is called a replica (duplicate table) and creation of a replica is called replication.
In replication, it is necessary to synchronize the contents of the master and those of the replica because the contents of the database are occasionally renewed. There are two methods of synchronization: the synchronization for real-time update and the asynchronous update based on periodical access to the master database.

Figure 3-2-6 Synchronization of Replication



3.3 Measures for Database Integrity

In the database system, processed results of multiple transactions are reflected in the database, and if necessary, the results are shown to users, or printed out. In this process, naturally, transactions themselves must be correct. In addition, in all manipulations such as requests for transaction processing, data manipulation, and result output, consistency of data and processing without contradiction are necessary. The feature is called integrity. As measures for database integrity, the previously mentioned items can be summarized as follows.

- Duplicate data → Data normalization
- Parallel processing of transactions → Concurrent execution control (Exclusive control)
- Update processing of distributed database → 2-phase commitment control
  → 3-phase commitment control

To achieve the database integrity, above all, correctness of the data is the most important factor.

Exercises

Q1 **Which of the DBMS features decides the schema?**

a) Security protection
b) Failure recovery
c) Definition
d) Maintenance

Q2 **In a database system, when multiple transaction processing programs simultaneously update the same database, which method is used to prevent logical contradiction?**

a) Normalization
b) Integrity constraints
c) Data-centric design
d) Exclusive control
e) Rollback

Q3 **There are mainly two files to be used for recovery of the database when a failure occurs in the media. One is a back-up file, and what is the other file?**

a) Transaction file
b) Master file
c) Rollback file
d) Log file

Q4 **Which is the correct data recovery procedure when the transaction processing program against the database has abnormally terminated while updating the data?**

a) Perform rollback processing using the information in the journal after update.
b) Perform rollforward processing using the information in the journal after update.
c) Perform rollback processing using the information in the journal before update.
d) Perform rollforward processing using the information in the journal before update.

Q5 **The ACID characteristic is required for application in the transaction processing. Which of the following features of ACID represents "the nature not producing contradiction by transaction processing?"**

a) Atomicity
b) Consistency
c) Isolation
d) Durability