TOPCIT

ESSENCE Ver. 3

Technical Field

02 Understanding and Using Data

TOPCIT ESSENCE is published to provide learning materials for TOPCIT examinees.
The TOPCIT Division desires the TOPCIT examinees who want to acquire the necessary practical competency in the field of ICTto exploit as self - directed learning materials.

For more information about TOPCIT ESSENCE, visit TOPCIT website or send us an e-mail.

As part of the **TOPCIT ESSENCE** contents feed into authors' personal opinions, it is not the TOPCIT Division's official stance.

Ministry of Science, ICT and Future Planning
Institute for Information and Communications Technology Promotion
Korea Productivity Center

Publisher TOPCIT Division
+82-2-398-7649 www.topcit.or.kr helpdesk@topcit.or.kr

Date of Publication 1st **Edition** 2014. 12. 10.
2nd **Edition** 2016. 02. 26.
3rd **Edition** 2020. 02. 26.

Copyright © Ministry of Science, ICT and Future Planning
All rights reserved.
No part of this book may be used or reproduced in any manner whatever without written permission.

TOPCIT

ESSENCE Ver. 3

Technical Field

02 Understanding and Using Data

TOPCIT

ESSENCE Ver. 3

Technical Field

02 Understanding and Using Data

CONTENTS

I. Understanding Data and Databases 18

01 Understanding Data 20

A) Concept and characteristics of data, information, and knowledge 20
B) Concept and characteristics of data processing types 20

02 Understanding the Database 21

A) Concept and characteristics of the file processing system 21
B) Concept and characteristics of the database 22

03 Understanding the Database System 23

A) The concept and components of the database system (DBS) 23
B) Data independence and ANSI–SPARC's 3–level database architecture 23
C) Definition and key roles of the database administrator (DBA) and data architect (DA) 25
D) Concept and functions of the DBMS (Database Management System) 26

II. Understanding Types of Databases 28

01 Types of Databases 30

A) Development process of the database 30
B) Major types of databases 30

02 Object relational Database 32

A) Object relational database (ORDB) 32

03 Understanding XML 32

A) Concept and characteristics of XML (Extensible Markup Language) 32
B) Schematic diagram and components of XML 33

C) XML configuration 33
D) Structure and major components of the XML processor 34
E) XML document creation procedure 35
F) DTD concept and creation procedure 35
G) Concept and characteristics of XML Schema, and comparison with DTD 36
H) Characteristics of XQuery 36
I) Concept and characteristics of XLL 37

04 Understanding Various Database Types 37
A) Multimedia database 37
B) Main memory database (MMDB) 37
C) Embedded database 38
D) Mobile database 38
E) Spatial database 38
F) Column base database 39

III. Database Design and Build Procedure 41

01 Database Design and Build Process 42
A) Requirements collection and analysis 43
B) DB design 43
C) Database building 44
D) Operation and maintenance 44

02 Considerations for Database Design 45
A) Deliverables to be analyzed 45
B) Considerations of the design subject 45

CONTENTS

03 Impact of database design in the project lifecycle 46

IV. Data Modeling 48

01 Concept andProcedure of Data Modeling 51

A) Requirements collection and analysis 51

B) DB design 51

C) Differences in the academic world's and industry's understanding of conceptual/logical modeling 51

D) Concept of process modeling 52

E) Concept of correlation modeling 52

02 Data Modeling Procedure 53

A) Conceptual modeling 53

B) Logical modeling 53

C) Physical model 53

03 Types of ER Notations 54

04 ER Notation Based on the Chen Model 55

A) Entity 55

B) Relationship 55

C) Attribute 56

05 Extended ER (EER: Extended Entity-Relationship) Model 57

A) Generalization/Specialization 57

B) Aggregation 57

06 Another Type of ERD Notation: The Crow's Foot Model 58
A) Entity, attribute, identifier 58
B) Characteristics of the entity 58
C) Classification of entities 58
D) Classification of attributes 59
E) Relationship 59
F) How to read relationships 60
G) Characteristic of the identifier 60
H) Classification of identifiers 61
I) Identifier relationship and non-identifier relationship 61
J) Super-type and sub-type 62

07 Connection Trap 63
A) Fan trap 63
B) Chasm trap 64

08 Object-Relational Mapping (ORM) 64
A) Class conversion 64
B) Conversion of the class relation into a relational relationship 65

09 Integrity and Key 65
A) ntegrity 65
B) Key 66

V. Normalization and Denormalization 67

01 Normalization and Anomalies 69
A) Insertion anomaly 69

CONTENTS

B) Deletion anomaly 69
C) Update anomaly 69

02 Concept of Functional Dependency and Rule of Inference 70
A) Functional dependency 70
B) Armstrong's axioms 71

03 Database Design in Which Normalization is Applied (1st, 2nd, 3rd, Boyce-Codd) 71
A) Normalization process ([Figure 28]) 71
B) Example of normalization 71

04 Database Design in Which Normalization is Applied - (4th normailzation) 73
A) Concept of the fourth normalization 73
B) Characteristics of the fourth normalization 73
C) Targets of the fourth normalization 74
D) Performing the fourth normalization 75

05 Database Design to Which Normalization is Applied - (5thnormalization) 76
A) Concept of the fifth normalization 76
B) Characteristics of the fifth normalization 76
C) Performing the fifth normalization 76
D)Performing the fifth normalization 77

06 Denormalization 77
A) Concept and procedure of denormalization 77
B) Actual denormalization 78

07 Considerations: for Performance Design 82

VI. Physics Design of the Database 84

01 Relational Table Conversion and Table Design 86

A) Differences between the academic world's and industry's understanding of physical modeling 86

B) Relational table conversion 86

C) Table design 87

02 Data Type Design 88

A) Character data type 88

B) Numeric data type 88

C) Binary data type 88

D) Date data type 88

03 Index Design 89

A) Functions of the index 89

B) Index design procedure 89

C) Types of index structures 89

04 View Design 89

A) Characteristics of the database view 89

B) Creation of the view 90

C) Modifying data with a view 90

D) Other considerations 90

05 Distributed Database 91

A) Characteristics of the distributed database 91

B) Data transparency 91

CONTENTS

VII. Database Quality and Standardization 93

01 Data Quality Control Framework 95

A) Data value 95

B) Data structure 96

C)Data management process 96

D) Data quality management maturity model 96

02 Data Standardization 97

A) Overview of data standardization 97

B) Necessity of data standardization 97

C) Components of data standardization 98

D) Defining data standards 98

E) Confirming data standards 99

VIII. Relational Operation (relational algebra) 100

01 Understanding the Concept of Relational Algebra 101

02 Set Operation and Relation Operation 101

A) Set operation 101

B) Relation operation 101

03 Extended Relational Algebra Operations 102

IX. Relational Database Language (SQL) 103

01 Types of Relational Database Languages 105

A) DDL, DCL, DML 105
B) Changes in SQL and the characteristics of SQL3 105

02 Data Definition Language (DDL) 106
A) Types of data definition language 106

03 Data Control Language (DCL) 107
A) Roles of the data control language 107
B) Types of data control languages 107

04 Data Manipulation Language (DML) 108
A) Basic DML operations 108
B) Aggregate DML operations 108
C) Advanced DML join operations 109

X. Database Query Application 111

01 Stored Procedure 113
A) Definition of the stored procedure 113
B) Strengths and weaknesses of the stored procedure 113
C) Strengths of the stored procedure 113
D) Shortcomings of the stored procedure 113

02 Embedded SQL 113
A) Definition of embedded SQL 113
B) Characteristics of embedded SQL 113
C) Embedded SQL cursor 114

03 Dynamic SQL 115

CONTENTS

A) Comparison between dynamic and static SQL 115
B) Processing methods of dynamic SQL and static SQL 116
C) Example of dynamic SQL and static SQL code 116

04 Query Optimization and Optimizer 117
A) Query optimization process 117
B) Optimizer 117
C) Roles of the optimizer in each query processing phase 117
D) Classification of optimizers according to criteria 118

05 Linking the Web and Database 118
A) Server expansion methods 118
B) Browser extension method 119

XI. Concurrency Control 121

01 What is a Transaction 123
A) Concept of a transaction 123
B) ACID properties of the transaction 123
C) when finishing a transaction 123
D) Considerations when executing a transaction 124

02 Concurrency Control 124
A) Definition of the serializable schedule of the transaction 124
B) Definition of concurrency control 124
C) Purposes of concurrency control 124
D) Problems that occur when concurrency is not controlled 125
E) Concurrency control techniques 125

F) What is the 2PL (2-Phase Locking) technique? 125

03 Transaction Isolation Level 126

A) Read uncommitted: Allows other transactions to read the data that are still being processed by the transaction (not yet completed). 126

B) Read committed: Allows other transactions to read the data that are confirmed by the completed transaction only. 126

C) Repeatable read: When a query is executed more than twice in a transaction, the record in the first query does not disappear or the value is not changed. 126

D) Serializable read: When a query is executed more than twice in a transaction, the record in the first query does not disappear or the value is not changed, and no new record appears either. 126

04 Deadlock 126

A) Definition of deadlock 126

B) Causes of deadlock 127

C) Solutions to deadlock 127

XII. Database Recovery 129

01 Concept of Database Failure and Recovery 131

A) Definition of data recovery 131

B) Types of database failures 131

C) Basic principles of database recovery: Principle of information redundancy 131

D) Types of database recovery actions 131

02 Database Troubleshooting Method 132

A) Database recovery technique 132

CONTENTS

B) Recovering from a distributed database 2-phase commit protocol 132

03 Database Backup 133

A) Database backup overview 133

B) Database backup requirements and main tasks 133

C) Types and characteristics of database backup methods 134

XIII. Understanding Database Analytics 136

01 Concept and Characteristics of the Data Warehouse (DW) 138

A) Concept of the data warehouse 138

B) Characteristics of the data warehouse 138

02 Data Warehouse Modeling 138

A) Definition of data warehouse modeling 138

B) Data warehouse modeling technique 139

03 Concept of ETL (Extraction, Transformation, Loading) 140

04 Concept and Search Technique of OLAP (Online Analytical Processing) 141

A) Concept of OLAP 141

B) OLAP search technique 141

05 Concept and Algorithm of Data Mining 141

XIV. Understanding Big Data and NoSQL 143

01 Big Data Overview 145

A) Definition and characteristics of big data 145
B) Detailed technology by big data life cycle 145

02 Technologies Related to Big Data 146
A) Collection technology 146
B) Big data storage/processing technology 146
C) Visualization technology 147
D) Classification of big data analytics 148
E) Main methods of big data analysis 148
F) Data scientist 149

03 NoSQL 150
A) Definition and characteristics of NoSQL 150
B) BASE attributes of NoSQL 150
C) Storage method of NoSQL 150
D) Characteristics of the NoSQL data model 152
E) Characteristics of the NoSQL data model 152

XV. Understanding Artificial intelligence 154

01 Overview of AI 156
A) Definition and classification of AI 156
B) History of AI 156
C) Distinguishing AI 157
D) Machine learning 157
E) Deep learning 160

I. Understanding Data and Databases

▶▶▶ Recent trends and major issues

According to the annual White Paper on the Data Industry published by the Korea Data Agency, the domestic data-related market is showing a steady growth trend. The trend also shows that interest in the diversity and availability of data is increasing significantly. Considering this trend, the fields of data utilization, database construction, and data solutions seem to be maintaining a steady rate of growth in all industries in Korea. It is also obvious that personnel requirements in the related fields will increase significantly in the coming years. Likewise, we can anticipate the main trends of business, such as the spread of the cloud platform to the data environment, increased interest in data quality management, and increased decision-making based on data. Judging from these trends, it is evident that competence related to the processing and utilization of data is becoming increasingly important.

▶▶▶ Learning objectives

1. To be able to explain the concepts and characteristics of data, information, and knowledge in the Information Age.
2. To be able to explain the concept and characteristics of data processing.
3. To be able to explain the concept and characteristics of the file handling system.
4. To be able to explain the concept and characteristics of the database.
5. To be able to explain the concepts and components of the database system.
6. To be able to explain the 3-level database architecture of ANSI-SPARC.
7. To be able to explain data independence.
8. To be able to explain the roles of the database administrator (DBA) and the concept of the data architect (DA).
9. To be able to explain the concept and functions of the DBMS (Database Management System).

▶▶▶ Keywords

Data, information, knowledge, database, batch processing, online processing, distributed processing, DBMS, data independence, ANSI-SPARC 3-level database architecture

✚ Preview for practical business

There are many cases in which a database is used like an existing file system, and tables are created in the database, but each table is actually dependent on individual applications, screens, and reports.
For example, when developing a book management system for each team a booklist table can be created for each team during implementation of the database in the same way as managing books using Excel files. However, if tables are designed in the same way as Excel files, the advantages of the database, including integration, storage, operation, and sharing, cannot be utilized at all.
In this case, the complexity of the application program also increases, causing problems that can lead to serious symptoms, such as an increase in the development cost, problems with data integrity (lack of consistency due to data duplication), and deterioration of data processing performance. Therefore, these problems can be prevented, and the advantages of the database can be maximized and applied to system development only when we understand the definition and characteristics of the database (integration, storage, operation, sharing) and its advantages, and apply them to practical business.

01 Understanding Data

A) Concept and characteristics of data, information, and knowledge

① Data

Data refers to the collected resource itself, which are the basic data discovered, investigated, collected, and created in the real world. That is, data are related to a fact, which refers to a natural state free of the values and judgments of human beings.

② Information

Information refers to various data that are organized, classified, and systematized for certain purposes according to certain rules. When data are processed in a certain form, the information necessary for achieving a specific purpose is created.

③ Knowledge

Knowledge refers to matter that is generalized from several items of concrete information. It is created while analyzing and studying the meaning and relationship of informational data. Correlations between information should be established to convert information into knowledge. Therefore, information can vary significantly depending on how one assigns meaning to informatized data, analyzes the relationship, and judges the values of the human being. Enterprises and institutions manage such information or knowledge in such a way as to make decisions or create added value.

<Table 1> Data, information, and knowledge

Item	Definition and characteristics	Key point
Fact	• A state of disorder in reality. Something that exists without being observed by others.	Phenomenon
Data	• Realistic data that exist widely in the real world. • Several simple facts that have not yet been evaluated for a specific purpose.	Factual data
Information	• When data are organized into meaningful patterns, they become information. • Data are treated and processed by a certain program (form) in order to create information necessary for achieving a specific purpose.	Treatment and processing
Knowledge	• Information of the same type is accumulated and organized into a generalized form. • A state to which human interpretations and meanings are given. • Information is used for decision-making or creation, and added value is created.	Added value, generalization, and decision-making
Wisdom	• A state in which an individual can understand and apply knowledge. • The mental ability to acquire, understand, apply, and develop knowledge.	Internalized ability

B) Concept and characteristics of data processing types

The data processing system, which is the core element of the information system, and is directly related to the computer, can be divided into batch processing, online processing, and distributed processing system depending on the type of data processing, in other words, how the data are organized and accessed.

① Batch processing system

• Data is collected for a certain period or a certain amount to process it at once.

- System-centered processing method. (Low processing cost and high system performance are required.)
- Preparatory work is required. (Collecting, classifying, and organizing raw data -> Writing in a file)
- Standby time is required. (Immediate processing is not supported.)
- The procedure of modifying changes is complex and difficult until files are processed collectively.
- Example: Payroll processing system, grade processing system, utility bill processing system, etc.

② Online processing system

- When data are transferred to a computer, the computer immediately processes the data. (Real-time processing system)
- User-centered processing method. (It incurs a high processing cost and low system performance is required.)
- Preparatory work is not required.
- Data currency is maintained.
- Difficult to maintain, repair, or restore.
- Example: Seat reservation processing systems for airlines and railroads, bank deposit processing systems, stock account systems for securities companies, etc.

③ Distributed processing system

- Data are processed by connecting processors and databases, which are geographically dispersed, over the network.
- Operated in the form of a client/server system.
- Improves the operation speed and reliability.
- Increases the efficient use of resources.
- Software development is difficult, and the security level and degree of design complexity are relatively high.

02 Understanding the Database

A) Concept and characteristics of the file processing system

The term "file processing system" refers to a method of storing and retrieving paper documents that was widely used before the computer was invented. However, the meaning of the term began to include computerized records in the 1960s. Such a file system is designed to arrange and manage the data recorded by the user on a physical disk. In general, a hierarchical file system with a directory structure is used.

Each application program in a file system accesses an individual file to process, in order to search, input, delete, and modify it.

For example, if a file system is used when a certain web service system saves the user's log-in history, the application program that saves the login history has a unique form according to the characteristics of the application program, depending on the sequence in which the information to be saved is arranged, and the unit of classification of such information.

① Characteristics of the file processing system

- The application program should directly implement the logical file structure designed by the application programmer

as a physical file structure.

- Application programmers can implement the data access method in the application program only when they know the physical data structure very well.
- Sharing data naturally becomes difficult in an environment where each application program has its own data. Therefore, one file ultimately exists only for one application only.

② Problems with the file system

- Inability to guarantee data independence - Program dependent.
- Problem of ensuring data consistency - Time dependence of a file (which has a different value depending on the time of retrieval).
- Problem of ensuring data integrity - Values that are semantically the same should remain identical.
- Low sharing and use convenience - Low economic feasibility and poor security management.

B) Concept and characteristics of the database

① Concept of the database

The database has the characteristics of integration, operation, storing, and sharing. In the past, data were inevitably saved in duplicate when recorded on paper, because even the same data could not be shared in real time. Data were also saved in several files in duplicate even when the file system was in use. The database, however, manages duplicated data in one place as intensively as possible to eliminate duplication to the maximum.

<Table 2> Data classification

Classification	Contents
Integrated data	This means that the same data are not duplicated in principle. • Minimal redundancy • Controlled redundancy
Stored data	Data are saved in a storage medium that can be accessed by a computer (tape, disk, etc.).
Operational data	Data which are required to perform the unique functions of an organization. (Temporary data used while performing a task, such as simple input/output, are not operational data.)
Shared data	Data which are jointly owned, maintained, and used by several application programs within an organization.

② Characteristics of the database

The shared database can be accessed in real time using the programming language, and is constantly changing due to the input, modification and deletion of data. The database can also be accessed and shared by multiple users simultaneously, and it has the characteristic of being referable by using the contents of data.

<Table 3> Characteristics of the database

Item	Contents
Real-time accessibility	Real-time response to occasional and informal queries.
Continuous evolution	Update, insert, delete: Dynamic characteristics (keeping the current state accurately during such changes).
Concurrent sharing	Supports the concurrent sharing of the same data in different ways.

Content reference	Refer to processing the contents of data requested by the user, (that is, by values) rather than by location or address.

03 Understanding the Database System

A) The concept and components of the database system (DBS)

① Definition of the database system

The term 'DBS' refers to a computer-centered system that creates necessary information by storing and managing data in a database.

[Figure 1] Database system (DBS)

② Components of the database system

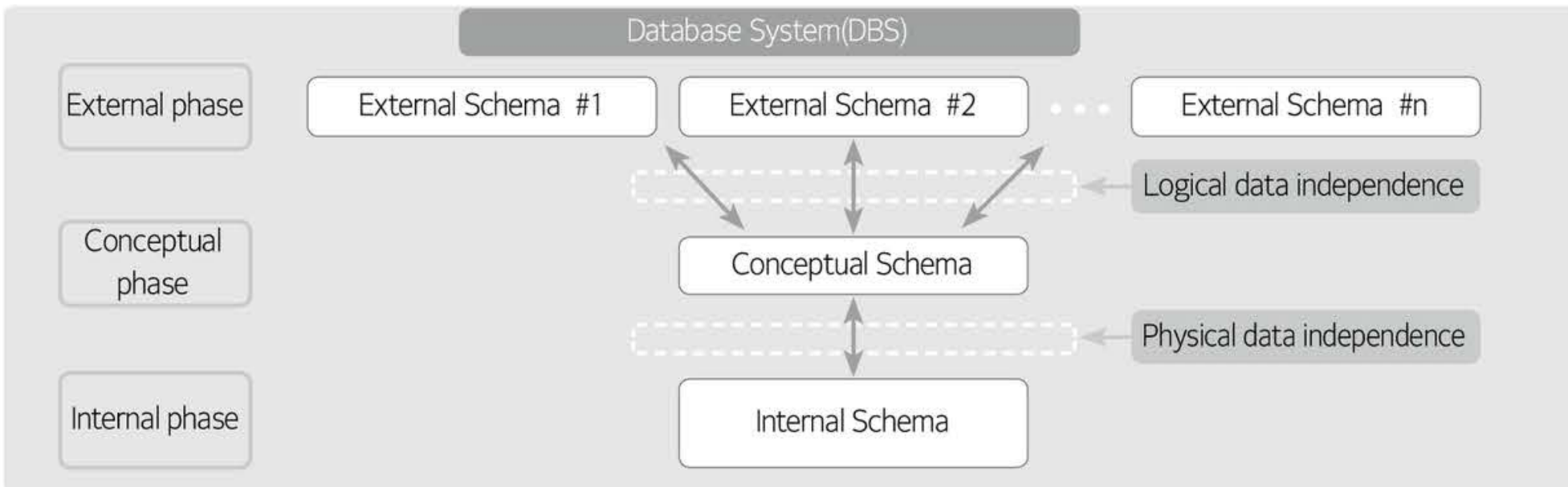
As regards the processing of data stored in a database, there is a user who processes it, a language that can manipulate and read data in the database, and DBMS software that enables the processing of all data.
That is, a database system is composed of four attributes: database, database language, user, and database management system (DBMS).

<Table 4> DBS components

Item	Contents
Database	A set of operational data integrated and stored with minimal redundancy, so that the multiple application systems of an organization can share it.
Database language	A tool that provides an interface between people and system.
Users	Database administrator (DBA), database application programmer, database user.
Database management system (DBMS)	System software that provides database construction and utilization functions.

B) Data independence and ANSI-SPARC's 3-level database architecture

① Background to introduction of the concept of data independence (reason for its necessity)



[Figure 2] 3-level database architecture

To understand data independence, it is helpful to understand the background to the introduction of the concept of data independence. The opposite of data independence can be defined as data dependency. Here, the subject of dependency generally refers to an application program. An application is the interface object which interacts with users and handles the user requirements. It can be said that data dependency has the purpose of reducing continuously increasing maintenance costs, data complexity, and the number of duplicated data. The concept of data independence was also introduced to maintain independence between the screen and the database against the users' constantly evolving requirements.

It can be said that data independence is a three-schema architecture, which was proposed by the special subcommittee of the X3 committee under the American Standards Institute (ANSI) for the DBMS and its interfaces.

The main purpose of this three-schema architecture was to reduce interference in changes, by separating the user's view of the database from the view by which the database is actually expressed. When data independence is secured, the independence of each view can be maintained, data can be changed without affecting the view by layer, and different DDL (data definition language) and DML (data manipulation language) can be provided according to the schema by phase.

② 3-level database architecture of ANSI-SPARC

The data independence model, which is based on the 3-level database architecture proposed by ANSI/SPARC, presents a model composed of an external phase, a conceptual phase, and an internal phase, none of which interferes with the others.

<Table 5> External schema, conceptual schema, and internal schema

Item	Contents	Remarks
External schema	• An external schema is composed of several users' viewpoints in the view phase, that is, a personal database schema viewed by everyone in each user's phase. • Individual database users or the database accessed by the application programmer are defined.	User's viewpoint Schema composition according to accessed characteristics
Conceptual schema	• Composed of one conceptual schema in the conceptual phase. The database of the entire organization is described, which has integrated all its users' perspectives. • The database of the entire organization is described, which has integrated the data needed by all application systems or users, including the data saved in the database and the schema that expresses the relationship between the data.	Integrated perspective

Internal schema	• An internal schema is composed of an internal phase and an internal schema. The format in which the database is physically stored. • A schema that expresses how data are actually saved in a physical device.	Physical storage structure

③ Data independence of two areas

As the concept of data independence can be separated into these three phases, there are two terms that specify the independence of each area, namely 'logical independence' and 'physical independence'.

<Table 6> Types of independence

Type	Contents	Characteristics
Logical independence	• When a concept schema is changed, an external schema is not affected. • An application program is not affected by a change in the logical structure.	• Changeable according to the users' characteristics. • The integrated structure can be changed.
Physical independence	• When an internal schema is changed, an external/conceptual schema is not affected. • An application program and a conceptual schema are not affected by a structural change of the storage device.	• A conceptual structure can be changed without affecting a physical structure. • A physical structure can be changed without affecting a conceptual structure.

④ Relationship between mapping and independence

The English word "mapping" can be translated into "sasang" in Korean, which can be defined as a "bridge that connects independent concepts". Broadly speaking, there are two types of mapping in data independence.

<Table 7> Types of mapping

Finishing	Contents	Example
External/conceptual mapping (logical thought)	The correlation between the external view and the conceptual view is defined.	A view can have different types of fields depending on the user's access type. The field type of a conceptual view is not changed.
Conceptual/internal mapping (physical mapping)	The correlation between the conceptual view and the stored database is defined.	If the structure of the stored database changes, conceptual/internal mapping should change to keep the conceptual schema intact.

C) Definition and key roles of the database administrator (DBA) and data architect (DA)

① Roles of the database administrator (DBA)

The DBA takes responsibility for the configuration and overall administration of the database to ensure that the functions of the database system are performed properly.

<Table 8> Tasks and roles of the DBA

Functional tasks	Technology details	Role
Data modeling	Data modeling according to business. Physical data modeling according to a storage environment. Semi-normalization, performance modeling.	Plays the role of a physical model for analysis/design when implementing a project.

Physical database design	Index design, storage space design. Clustering design, partition design, etc.	Physical design according to the physical space environment, server, DBMS - Plays the role of the designer
Tuning (performance improvement)	Improvement of performance according to the index distribution chart, join relationship, and type and number of transactions.	Plays the role of the tuner.
Database building	Creation of a table space and a data file space. Creation of a database object. Setting of parameters and specifying of a backup structure.	Plays the role of the developer.
Database operation	Backup/restoration. Monitoring of memory and performance on a regular basis.	Plays the role of the operator.
Database standardization	Definition of a glossary and a domain. Management of enterprise meta-data.	Manages the database or data standardization.

② Roles of the data architect (DA)

A data architect establishes, models, and systemizes the policies and standards for data-related elements such as data, database, data standard, and data security.

<Table 9> Roles of the data architect

Role	Details	Key roles
Establishment of a data management system	Establishment of systems for metadata, data distribution/integration, information lifecycle management (ILM), performance and capacity monitoring system, log management system, failure management system, etc.	Establishment of data governance
Establishment of data standards	Establishment of standards related to entire data, such as definitions of terms and domains, data dictionary, metadata standards, etc.	Maintaining consistency is very important.
Performance of data modeling	Conceptual modeling > Logical modeling > Physical modeling	Important roles in the structure of the entire data architecture
Establishment of data security systems	Establishment of systems for controlling user access, tables, views, etc., data encryption, access logs, transaction traceability, etc.	Management of data security

D) Concept and functions of the DBMS (Database Management System)

① Concept of the DBMS

- A system contrived to solve the problem of the file system, dependency and redundancy.
- A software system that manages the database so that it can be shared by all application programs as a mediator between the application program and data.

② Functions of the DBMS

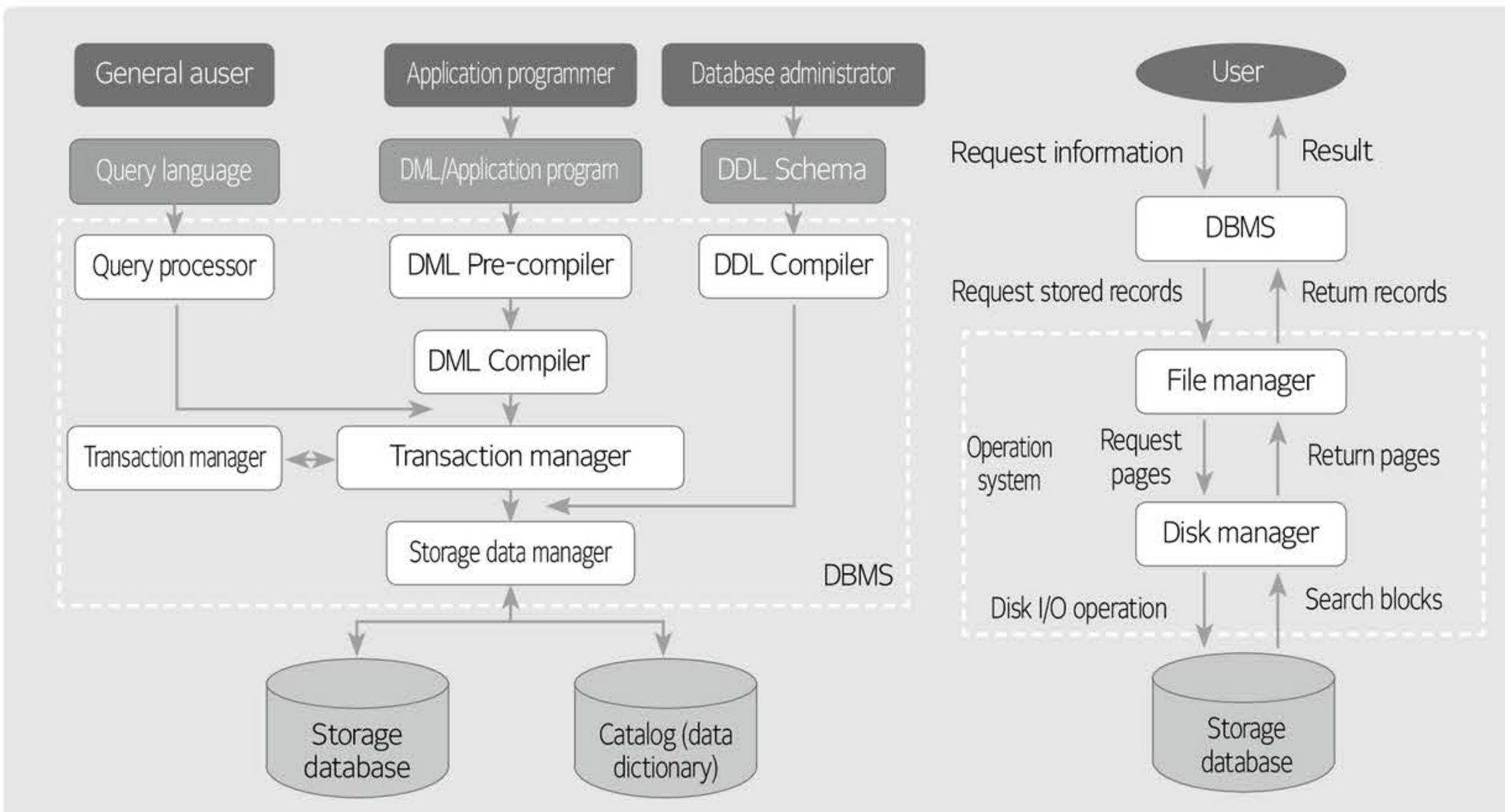
There is a file structure that stores a database, and memory and major processes to process the file structure in DBSM.

- Controls duplication from the aspects of data storage, and development and maintenance.
- Enables data sharing among multiple users.

- Controls access to data by unauthorized users.
- Provides various types of interfaces to diverse users.
- Expresses the complex relationships that exist between data.
- Ensures the integrity of the database.

③ Conceptual diagram and main functions of the DBMS

There is a file structure that stores a database, and memory and major processes to process the file structure in DBSM.



[Figure 3] Conceptual diagram of the DBMS

<Table 10> DBMS components

Component	Description
DDL compiler	• Processes the schema specified in the DDL as an internal schema, i.e. metadata, and stores it in the system catalog. • All DBMS modules should access and use this catalog information when necessary.
Query processor	Processes advanced queries submitted by general users. I.e. queries are analyzed, parsed, and compiled. Then, the database access code is created and sent to the runtime DB handler for execution.
DML pre-compiler	DML commands inserted into an application program are extracted and sent to the DML compiler to be compiled as an object code for database access.
DML compiler	A DML compiler generates an object code by parsing and compiling the received DML command.
Runtime database handler	Database access is managed at runtime. That is, database operations such as search or update are executed in the database using the storage data manager.
Transaction manager	• The transaction manager checks compliance with integrity constraints and the user's rights while accessing the database. • Performs restoration when controlling transactions concurrently or when a failure occurs.
Stored data manager	The stored data manager manages access to the user database and catalog stored in the disk. (Request to the file manager of the OS.)

II. Understanding Types of Databases

Recent trends and major issues

The "database" was created in the late 1960s to manage and use data more efficiently than the computer environment which used the file system in the early days. Since then, the basic model of database has evolved continuously into relational, object-oriented, and object relational types of database, having started as a network model. Currently, the object relational model is the most widely used model in the market.
In addition, the internal data model and the structure of the database were a matter of interest in the past. However, the importance of database processing performance was highlighted with regard to the processing of large-scale data, as the problem of performance degradation became increasingly serious due to the rapid increase in the amount of digital data. Accordingly, the relevant technology has expanded to include new types of databases, such as the column-type database which places emphasis on database performance and scalability rather than data relations, NoSQL, etc.

Learning objectives

1. To be able to explain the data model and the structure of each type of database.
2. To be able to explain the concept and characteristics of the object relational database (ORDB).
3. To be able to understand and write an XML document.
4. To be able to explain various database systems.

Keywords

Hierarchical database, network database, relational database (RDB), object-oriented database (OODB), object relational database (ORDB)
XML, multimedia database, main memory database, embedded database, spatial database, column type database, graph database

✚ Preview for practical business Reasons to understand database types

The relational database and the object relational database, which are based on E.F. Codd's normalization theory of 1970, have mainly been used in the existing workplace, because data consistency was important in workplaces where OLTP (Online Transaction Processing) jobs were dominant. However, the processing of diverse types of data and the rapid processing of large quantities of data are required these days. Therefore, DBMS vendors are developing DBMS functions in such a way that the advantages of each type of database can be introduced and applied.

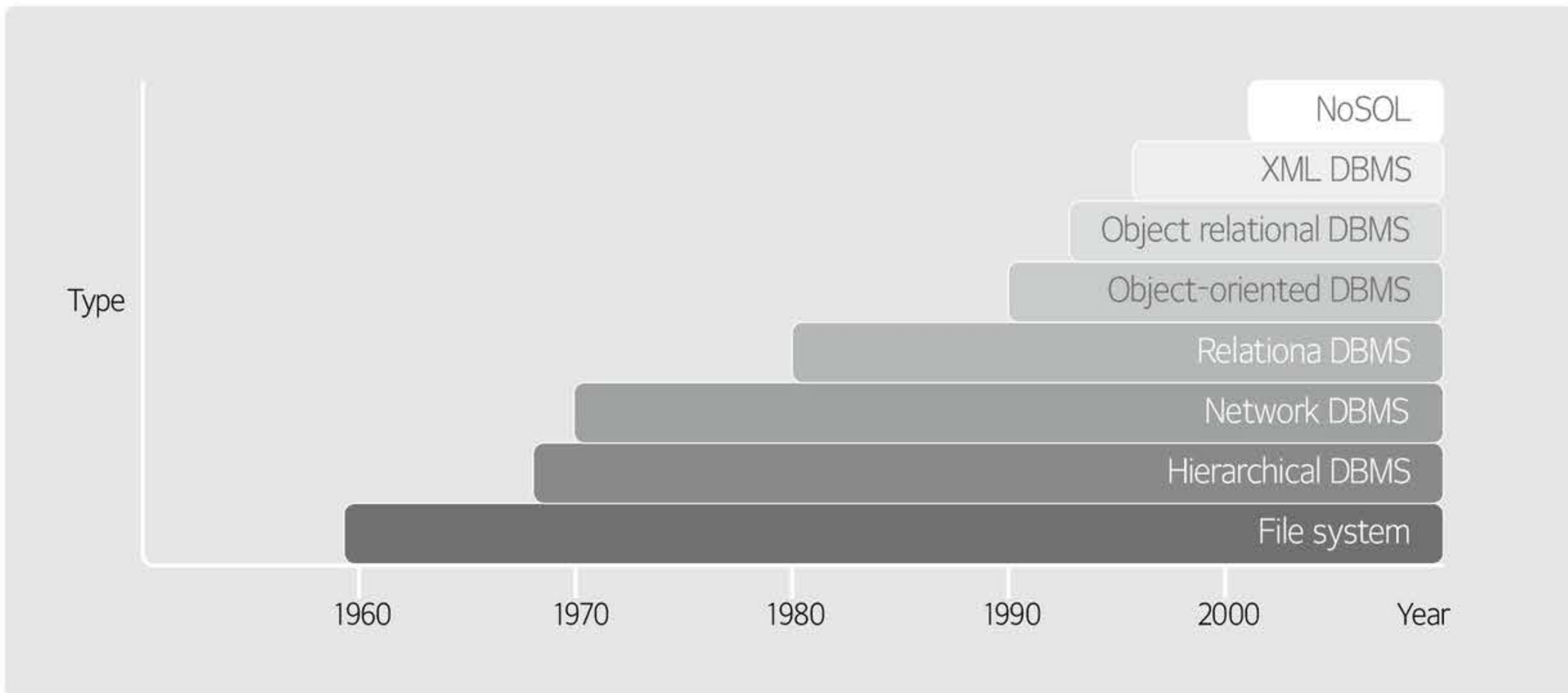
As a result, if we know the background to database development and the basic concepts of the major types of database, we can easily understand the functions and purposes of the various DBMSs used for various purposes in the workplace and get help in strategically selecting a DBMS that is optimized for the characteristics and purposes of the business in question.

01 Types of Databases

A) Development process of the database

The hierarchical and network databases that developed the data structure used in existing applications were the main types of database used in the early days. However, these databases had a big problem in that it was difficult to maintain consistency. To resolve these problems, relational databases appeared in the 1970s. Since then, databases have developed continuously around relational database technology.

In the 1990s, user-defined data and multimedia data needed to be managed, but existing relational data models could not handle such complex data. Accordingly, an object-oriented database combining object-oriented technology, which began to gain attention from the mid-1980s, and a database appeared. Since then, an object-oriented database that combines the advantages of both the relational database and the object-oriented database has appeared and been widely used until now. Databases are evolving from simple business applications to large-capacity and complex business environment supports to respond to changes in the IT market and technology, such as the XML database, which emerged as a response to the development of the Internet environment, and the NoSQL, which was developed to process big data.



[Figure 4] Application area of the database by era

B) Major types of databases

① Hierarchical database

A hierarchical database hierarchically stores data in a tree format of the relationship between subordinates and superiors. Although this database has the advantages of rapid data access and easy prediction of data usage, it cannot adapt to changing business processes easily.

The major characteristics of the hierarchical database are as follows.

- It is the oldest database with a hierarchical structure (1960s).
- Each hierarchical structure maintains a dependent relationship by connecting with physical pointers.
- It is difficult to change the data structure when the business process is changed after building a database.

• It is difficult to perform unexpected random search.

② Network database

A network database stores data by expanding the tree form of a hierarchical database into a network form. Pointers are used to maintain a many-to-many relationship between records and to link data.
The major characteristics of the network database are as follows.

• It was developed in the early 1970s to solve the problems with the hierarchical database.
• Data can be extracted quickly and effectively by adding links between hierarchical structures.
• High maintenance costs and backlog are entailed by the complex type system.
• Programmers must understand the structure to write a program.
• A record can have pointers to children and sibling records and to parent records, which was not possible with the hierarchical database.

③ Relational database (RDB)

The relational database is based on the relational data model proposed by E.F. Codd in the 1970s. Major commercial products include Oracle, SQL Server, DB2, Informix, Sybase, and Ingres.
Characteristics of the relational database

• The model itself is simple. The database stores characters/numbers/date information in a two-dimensional structure (column, row).
• The model is based on mathematical theory - particularly the set theory of mathematics. Therefore, the performance of the system to be developed can be predicted and verified mathematically, and various operations can be mathematically optimized.
• Existence of query language - Anyone can easily search for desired information by learning a simple query language in the 4GL format.
• Contiguous support for technology that keeps pace with the circumstances of the times - Supports the client/server architecture, large-scale parallel processing, etc.

④ Object-oriented database (OODB)

The relational database cannot create new types of data or expand existing types, and has difficulty processing unstructured complex information such as multimedia. Also, the standard query language "SQL" expresses data relationships by values, making it difficult to find and process mutually related entities when expressing a complex object. Accordingly, a database that can search and store information based on an object model emerged, namely, the object-oriented database. Major commercial products include ObjectStore, O2, Objectivity, and Uni-SQL.
The object-oriented database has the following characteristics.

• User-defined data types are supported, and inheritance can be specified.
• Unstructured complex information can be modeled.
• Information can be accessed based on navigation using the reference structure between objects.
• The structure of information in the program is also same as that of the database schema.

However, the object-oriented database was not distributed widely in the market because its basic database management functions were found to be weak, such as transaction processing, number of supported concurrent users, backup and recovery, etc., and system safety and performance were never proven in an enterprise environment.

02 Object Relational Database

A) Object relational database (ORDB)

① Concept of the object relational database

Even though the object-oriented database was introduced to solve the weakness of the relational database regarding new advanced applications, it showed certain limitations in terms of its use in an enterprise environment. To overcome this, the object relational database was introduced by combining the existing relational database with the concept of the object-oriented database and expanding its functions. Currently, most commercial databases are object relational databases, such as Oracle's Oracle9i, IBM's DB2 UDB, and Microsoft's SQL Server.

② Characteristics of the object relational database

The object relational database has the following characteristics.

<Table 11> Characteristics of the object relational database

Characteristic	Description
Supports user-defined data types	Users can define and use new types of data in addition to basic data types.
Supports the reference type	Data can be accessed based on navigation using the reference structure in such a way that one object record refers to another object record.
Supports a nested table	A data model with a complex structure can be designed because one column in a table is composed of another table.
Supports large-scale objects	Supports LOB (Large Object) as a basic data type for large-scale unstructured data such as images, audio, and video.
Supports table inheritance relationships	Accommodates the advantages of the object-oriented database by specifying the inheritance relationship between tables.

03 Understanding XML

A) Concept and characteristics of XML (Extensible Markup Language)

HTML (HyperText Markup Language) is mainly used to create and format web documents in a web environment, but it remains unsuitable for specifying structured data extracted from a database. Therefore, the World Wide Web Consortium (W3C) has developed an extensible markup language (XML) as a standard language that is used to structure and exchange data in a web environment.

XML has the following characteristics.

① Simplicity: XML can simplify SGML (unused functions have been eliminated, while major functions have been retained).

② Openness: XML can be used together with HTML on the web, and can exchange meta-data.

③ Scalability: It can create its own tags and supports self-description.

④ Structure can be understood both by humans and machines: Makes it easy to compare and collect data.

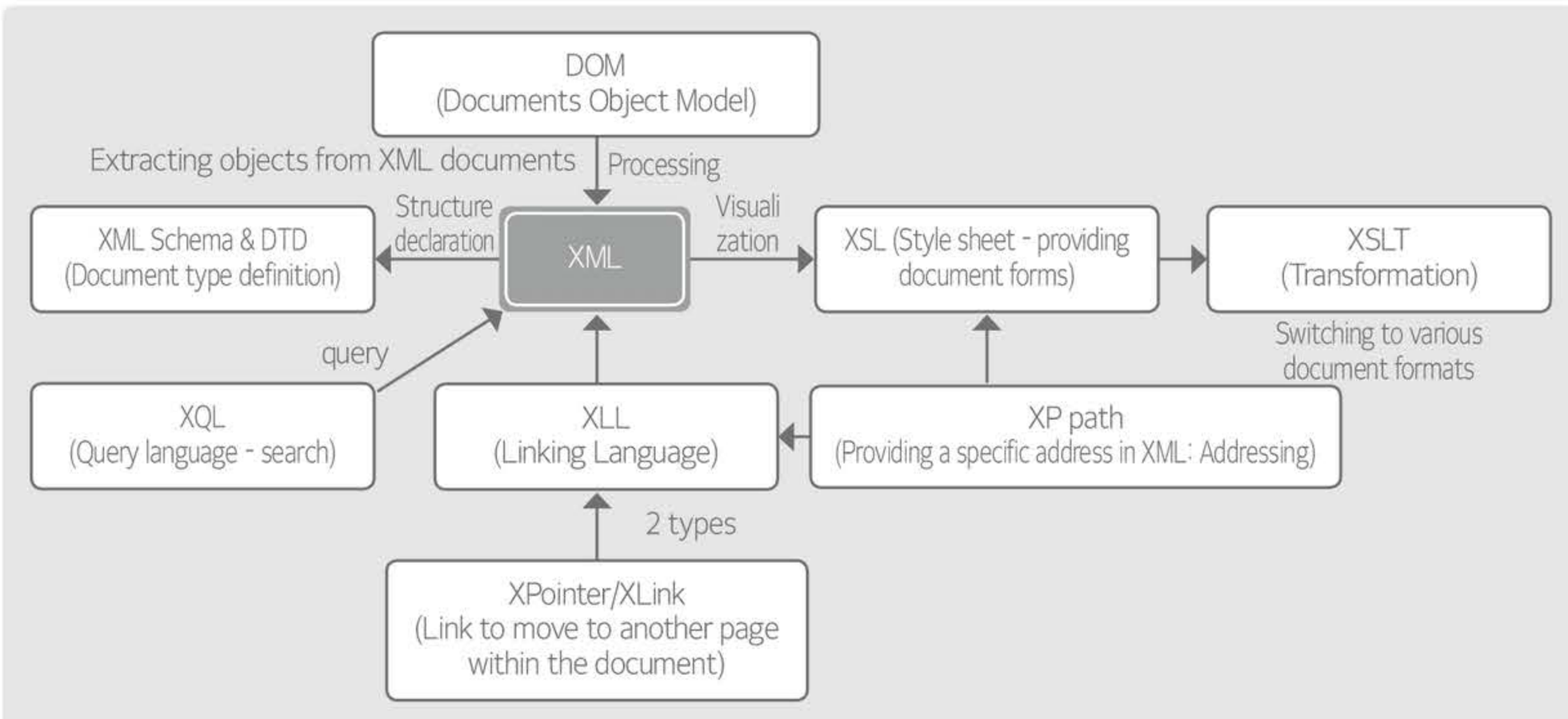
⑤ Separation of content and expression: The format can be changed according to the user's preferences (increased reusability).

⑥ Hierarchical structure: Supports structure search and full text search.

⑦ Unicode: Supports multiple languages.

B) Schematic diagram and components of XML

XML is composed of XML Document Type Definition (DTD) and the XML schema for specifying XML; the XPath for processing XML documents; and XQuery, Extensible Stylesheet Language (XSL), XML Linking Language (XLL), etc. To create an XML document, it is necessary to fully understand the XML components and the basic syntax.



[Figure 5] Diagram of XML configuration

C) XML configuration

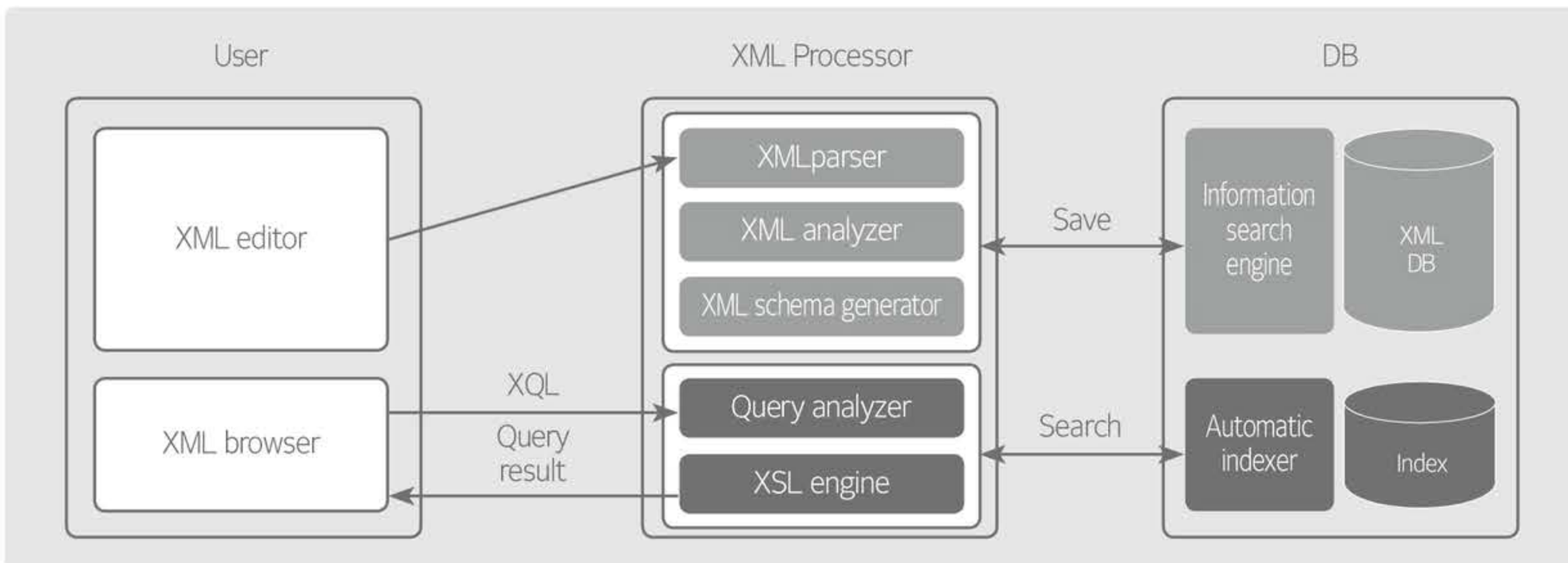
① XML components

<Table 12> XML components

Component	Description
XML DTD	• XML Document Type Definition • A document that defines the form of an XML document in a consistent structure. XML DTD is used to validate XML.
XML Schema	• High-level data definition language used to specify the structure of an XML document by replacing XML DTD. • More complex data types than XML DTD can be declared.
XPath	• A language that expresses extended queries so that the search condition can be included in the XML path expression.
XQuery	• A standard XML query language that can extract intended information from an XML file as if using a database.

XSL/XSLT	• XSL (Extensible Stylesheet Language) is a language that specifies the style sheet, which is used to express XML data in various different forms. • XSLT (Extensible Stylesheet Language Transformation) is a part of XSL that converts an XML document into a different type of document (such as HTML) so that it can be displayed on general browsers.
XLL	• The extensible Linking Language (XLL) displays the connection and relationship between XML elements. • XLink: Recognizes and processes hyperlinks. • XPointer: An address of the element in an XML document.

D) Structure and major components of the XML processor

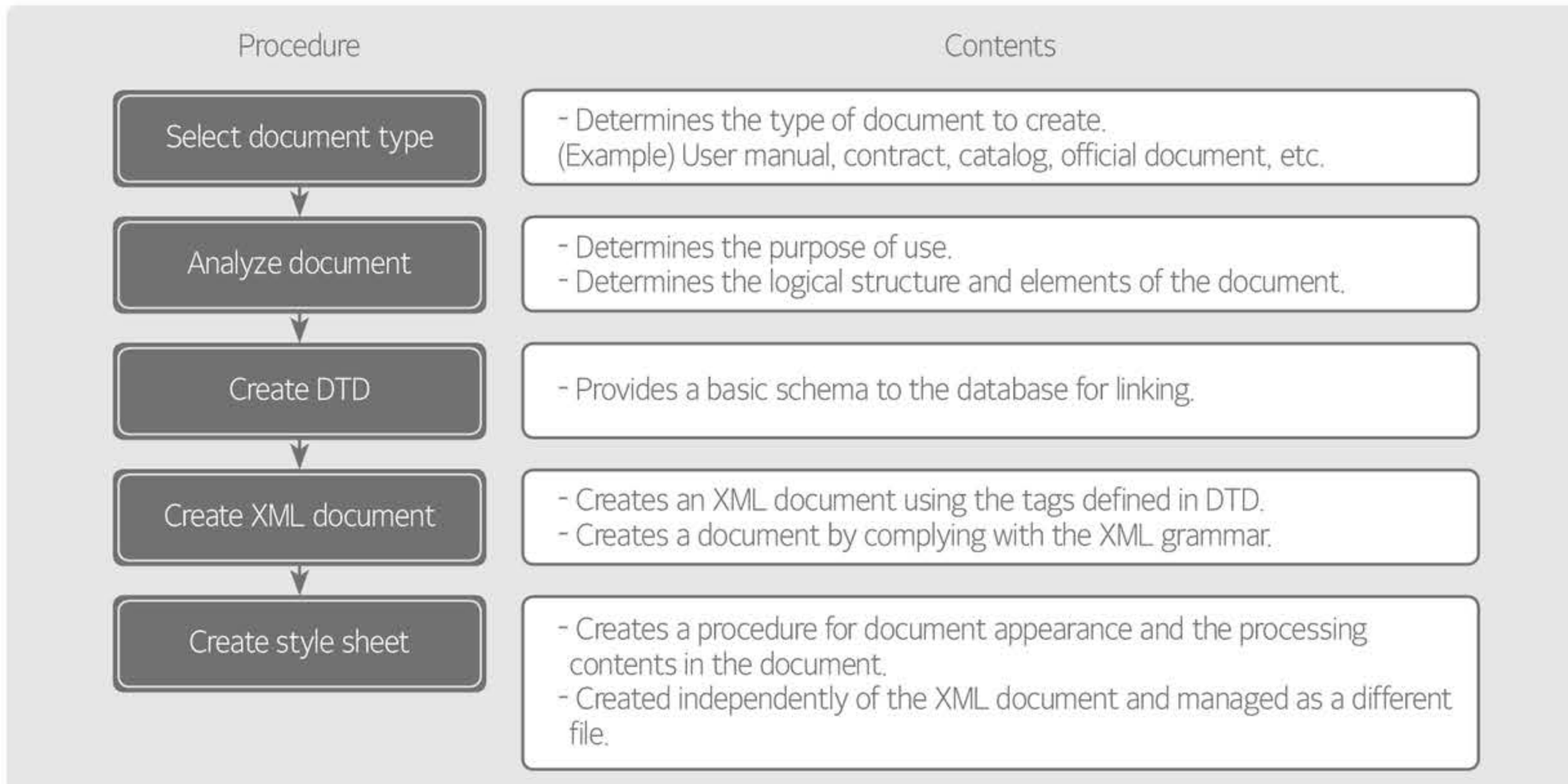


[Figure 6] Structure and components of the XML processor

<Table 13> XML components

Component	Function
XML parser	Checks and inspects the grammar and syntax structure of an XML document (validation check).
XML syntax analyzer	Analyzes the syntax structure of an XML document (SAX, DOM).
XSL engine	Converts an XML document to a document format with expression information.

E) XML document creation procedure



[Figure 7] XML creation procedure

F) DTD concept and creation procedure

DTD is a file that explicitly declares the structure of an XML document by defining the structure and contents of that document. There are several types of DTD declaration, as shown in the table below.

<Table 14> Types of DTD declaration

Definition	Declaration	Yes
Element type declaration	- element type declaration	<!ELEMENT element name~>
Attribute list declaration	- attribute type declaration	<!ATTLIST element name~>
Entity declaration	- entity declaration	<!ENTITY ~>
Notation declaration	-notation declaration -non-xml data processing: Images, etc.	<!NOTATION name~>

• DTD creation procedure

① Step 1: DTD declaration - Describing the process of declaring DTD

<! DOCTYPE Root_Element[<! ELEMENT Root_Element(..)> <...> <...>]>	<! DOCTYPE books[<! ELEMENT book(title, author>)> <! ELEMENT title(#PCDATA>)>]>

② Step 2: Declaring an element type

<!ELEMENT element_name(con tent_model)>	<! ELEMENT books(book*)>
[Note] The asterisk * symbol indicates a case in which an element is omitted or may appear multiple times.	

③ Step 3: Combining XML and DTD

- Determines whether the DTD declaration and definition part will be created inside XML or saved as an external file.
- Internal declaration: Defines the DTD in an XML document.
- External declaration: Applies the DTD (Document type declaration) part to an XML document.

G) Concept and characteristics of XML Schema, and comparison with DTD

The existing DTD cannot limit or extend the data type or range of certain information, and the DTD description grammar differs from the XML description grammar, which requires understanding of both the DTD and XML grammars. As a result, the XML schema was introduced to replace the DTD and to enable the creation of a data type that can process a document more easily.

Supporting data types: More complex types of data than the existing DTD can be created, and new types can be created and used.

- Supporting complex structure definition: Another schema document can be included in a schema document using the Schema location directive.
- Supporting a namespace: XML Schema supports a namespace.

* Namespace: An abstract entity that can distinguish elements when extracting elements from an XML document type and combining them with other documents, and when processing multiple documents at the same time.

<Table 15> Comparison of the XML Schema and DTD

Item	XML Schema	DTD
Writing grammar	Complies with XML 1.0.	EBNF + pseudo
Structure	Complex	Relatively simple
Namespace	Supported (Multiple namespaces can be used in a document.)	Not supported (Used once in a document.)
DOM support	DOM is supported and can be used because it is XML.	Not supported.
Dynamic schema support	Supported (Selecting at runtime. Subject to change due to an interaction.)	Not supported (DTD can actually read only.)
Data type	Extensive data type	Very limited data type
Scalability	Fully object-oriented scalability	Extension using string substitution
Openness	Content model that supports open and closed modification.	Closed structure

H) Characteristics of XQuery

XQuery is a query language that is used to search an XML-based database, and can extract information from XML files as if using a database.

<Table 16> Characteristics of XQuery

Characteristics	Description
Technology neutral	A technology that is standardized based on W3C-based XQuery1.0.
XML-based query language	- DA data search and storage technology using XML. - A language that is originated from an XML query language called Quilt and which includes the XPath expression. - The result of the query statement written in XQuery is the list of nodes indicating a tree structure, not an XML document.

Simple and easy to implement	Easy to implement using syntax similar to SQL, such as for, let, where, and return (FLWR).

I) Concept and characteristics of XLL

XLL is a standard language that performs link functions in an XML document, such as linking XML documents and setting specific locations within an XML document. XLL was introduced to overcome the limitations of HTML-based simple hyperlinks and to provide various approaches to using resources on the web. XLL supports various hyperlink methods and two-way links between link resources. XLL can use some of the required web resources by using XPointer or XPath, and is used as a core technology in X-Internet, etc.

- Supports two types, i.e. Xpointer when moving within the same document, and Xlnik when moving between different pages, etc.
- Provides a two-way link between resources.
- Provides an extended pointer (XPointer) for resource locating.
- Link type: Simple, Extended, Locator, Group, Document.

04 Understanding Various Database Types

A) Multimedia database

The multimedia database was developed to efficiently search and manage unstructured multimedia data characterized by their large capacity and complexity, such as text, image, audio, video, etc. Existing databases face limitations due to the exponential increase of unstructured/multimedia data, and an object-oriented approach, the synchronized representation of multimedia, and the concept of time-dependent modeling were used to model various types of unstructured data.

<Table 17> Building a multimedia database

How to build	Main content
File based	Used for simple search-oriented VOD (Video On Demand). It is difficult to support the function for restoring concurrent data access rights (DBMS functions are not used.)
RDBMS based	Stores ASCII text data in the CLOB (Character Large Object) field. Stores the image/video/audio in the BLOB (Binary Large Object) field. It is difficult to build a complete multimedia database.
OODBMS based	Defines the classes for each media using the user-defined class and user-defined method function. It is incompatible with existing databases
ORDBMS based	Supports the CLOB and BLOB field to store mono media. Defines the types for each media using user-defined types and functions.

B) Main memory database (MMDB)

Unlike general commercial databases in which a database is stored on a disk, the main memory database manages and manipulates a database by keeping it in the memory. The current trend is that the main memory database is actively used due to business demands, such as securing the competitiveness of an enterprise through quick decision-making, and the revolutionary development of related technologies such as the release of 64-bit OS, the

reduction of memory prices, etc.

The MMDB has the following characteristics due to the fact that the database runs in the main memory.

- Disk I/O does not occur because all operations are performed in the main memory.
- Performance is good because disk I/O, which degrades performance in the disk-based database, is rare.
- Hardware-based error recovery techniques are used due to the volatility of the main memory.
- A disk is used for backup and logging.
- Hashing and T-tree indexing algorithm optimized for a memory environment is used.
- Types of commercial main memory databases can be updated at [https://en.wikipedia.org/wiki/List_of_in-memory_databases].

C) Embedded database

General commercial databases are not suitable for the embedded system, which has limited memory and special performance goals. The embedded database was developed for the embedded system, so as to allow specific functions to be used in a limited embedded environment.

As a result, the embedded database has the following technical characteristics.

- It provides essential functions only after reducing overheads to use minimum RAM and disk.
- It supports communication between different devices to allow communication with the database in a central server.
- It supports porting to the various platforms of the embedded system.
- It satisfies the performance requirements of the real-time OS of the embedded system.
- The list of commercial embedded databases can be updated at [http://embedded-databases.com].

D) Mobile database

The mobile database is exclusively used by mobile devices. The mobile database, which is installed in a mobile device, processes the data generated during field work and sends them to the central server for synchronization. Due to the distinct characteristics of installation in a mobile device, a mobile database should be independent from various platforms and operation systems, should recover quickly from failures, and should be optimized for a limited mobile environment.

As a result, the mobile database has the following characteristics.

- The database can be installed in a small capacity device equipped with a limited CPU and memory.
- It is provided as an embedded form that integrates data and applications.
- It has the function of data replication and synchronization with the database of the existing server.
- Commercial mobile databases include Sybase SQL Anywhere, Oracle Lite, Microsoft SQL Server Compact, SQLite, and IBM DB2 Everyplace.

E) Spatial database

A spatial database is a set of non-spatial data represented by letters and numbers and spatial data represented by coordinate values. The spatial database was initially developed because a technology for processing "unstructured data", such as the geographic information system, was needed as a core technology for a type of guided missile that strikes a preset target by tracking the target's location using geographical information. Thereafter, the spatial database was introduced full scale, as the market was expanded to include services that store and manage values for specific locations using geographical information, such as GIS and location information check.

The spatial database has the following characteristics.

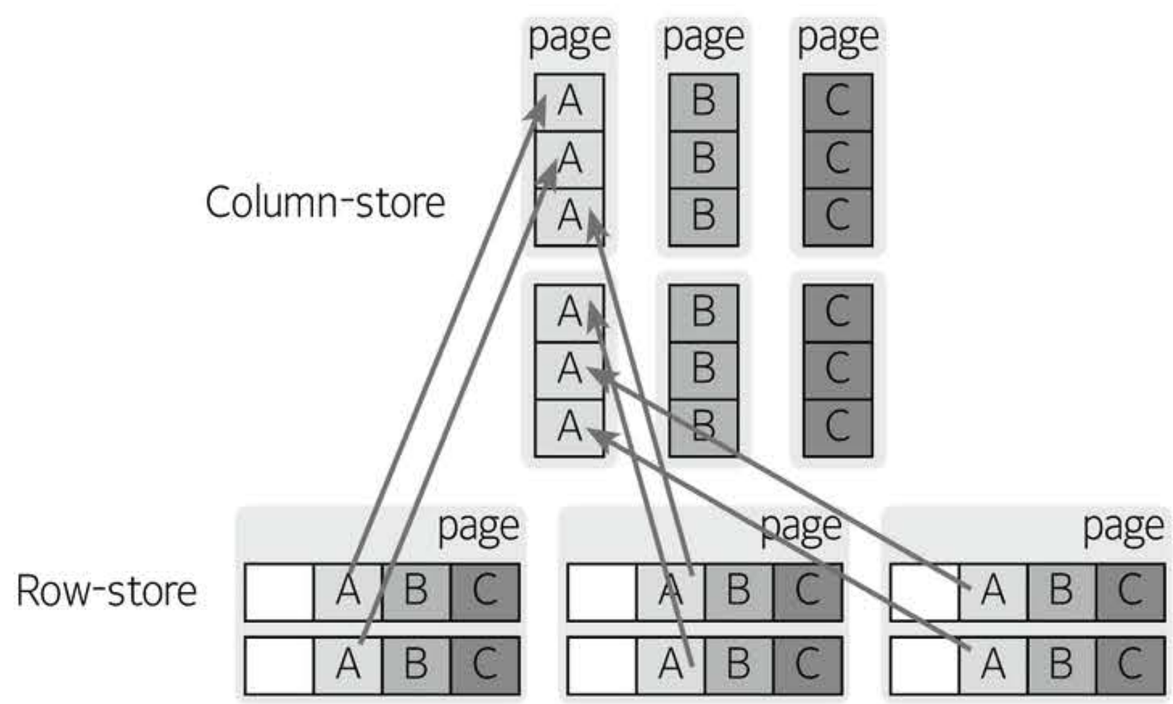
- Geometry, a geographical object, and topology for the spatial relationship between objects are included.
- Unstructured data can be processed; large amounts of data can be processed quickly.
- Spatial (topological, geometric) characteristics are reflected.
- New index and operation (R-tree index) are used to sort data that cannot be sorted.
- Expressive data model that can express complex information.
- Supports the combination of spatial data and non-spatial data.

F) Column base database

A column base database physically stores data based on columns. The data storage method of the relational database is not defined as row base or column base, but general relational databases use a physical storage structure based on rows. However, a way to store data with this structure reached the limit in analyzing large-scale data at high speed because its physical structure required that unnecessary data should be also read. The concept of column storage did not suddenly appear recently. Rather, this concept was studied relatively early on, including TAXIR, which was developed in 1969 for searching biological information. Recently, as the demand for the rapid processing of large amounts of data increased, a function that partially borrowed the concept of the column store such as the column store index, column store base DBMS, and hybrid DBMS supporting both structures were developed from the mid-2000s. Column base database technology is now being actively used in the DBMS market, together with the main memory database.

The column base database has the following structure and characteristics compared to the row-based database.

<Table 18> Comparison of the column base database and row base database

Item	Row-based Database	Column-based Database
Physical storage structure		

<table>
<tr><th>Item</th><th>Row-based Database</th><th>Column-based Database</th></tr>
<tr><td>Conceptual diagram</td><td><table><tr><td>Employee number</td><td>1001</td></tr><tr><td>Name</td><td>Hong Gildong</td></tr><tr><td>Payroll</td><td>1000</td></tr><tr><td>Phone</td><td>09) 123- 2134</td></tr><tr><td>Employee number</td><td>1002</td></tr><tr><td>Name</td><td>Lim Ggoekjeong</td></tr><tr><td>Payroll</td><td>2000</td></tr><tr><td>Phone</td><td>09) 253- 9378</td></tr><tr><td></td><td>...</td></tr></table></td><td><table><tr><td>Employee number</td><td>1001</td></tr><tr><td>Employee number</td><td>1002</td></tr><tr><td>Employee number</td><td>1003</td></tr><tr><td>Employee number</td><td>1004</td></tr><tr><td>Name</td><td>Hong Gildong</td></tr><tr><td>Name</td><td>Lim Ggoekjeong</td></tr><tr><td>Name</td><td>Lee Dongcheol</td></tr><tr><td>Name</td><td>Lee Cheolsu</td></tr><tr><td></td><td>...</td></tr></table></td></tr>
<tr><td>Characteristics</td><td>- Data are stored in row units.
- Several records are stored on one disk page.</td><td>- Data are stored in column units.
- The same column values are stored continuously on one disk page.</td></tr>
<tr><td>Transaction characteristics</td><td>- Suitable for adding, modifying and deleting data by record.</td><td>- Suitable for processing large amounts of data for the same column.</td></tr>
<tr><td>Data compression efficiency</td><td>- Compression efficiency is relatively low because records are unique and without duplication.</td><td>- Compression efficiency is high when there are many duplicate values for each column.</td></tr>
<tr><td>Example of frequently used SQL patterns</td><td>SELECT * (or a large number of columns)
FROM Table</td><td>SELECT AVG(COL1) (or column operation)
FROM Table</td></tr>
<tr><td>Applied DBMS</td><td>RDBMS for general OLTP
Oracle, DB2, Sybase ASE, etc</td><td>RDBMS for analysis
Vectorwise, Sybase IQ, SAP HANA, etc.</td></tr>
</table>

• The list of commercial column-oriented databases is updated at [https://en.wikipedia.org/wiki/List_of_column-oriented_DBMSes].

III. Database Design and Build Procedure

Recent trends and major issues

In the past, many databases were built to create a "paperless" environment in which data recorded on paper are digitalized and stored in a database. Recently, there have been many cases in which databases have been elaborately built from a digital environment to a digital environment. Instead of simply managing data with a fixed format or length, there is an increasing number of cases in which data are stored and processed as various types of unstructured data regardless of the data format or size.

Learning objectives

1. To be able to explain the database design and build process.
2. To be able to explain the matters to take into consideration when designing a database

Keywords

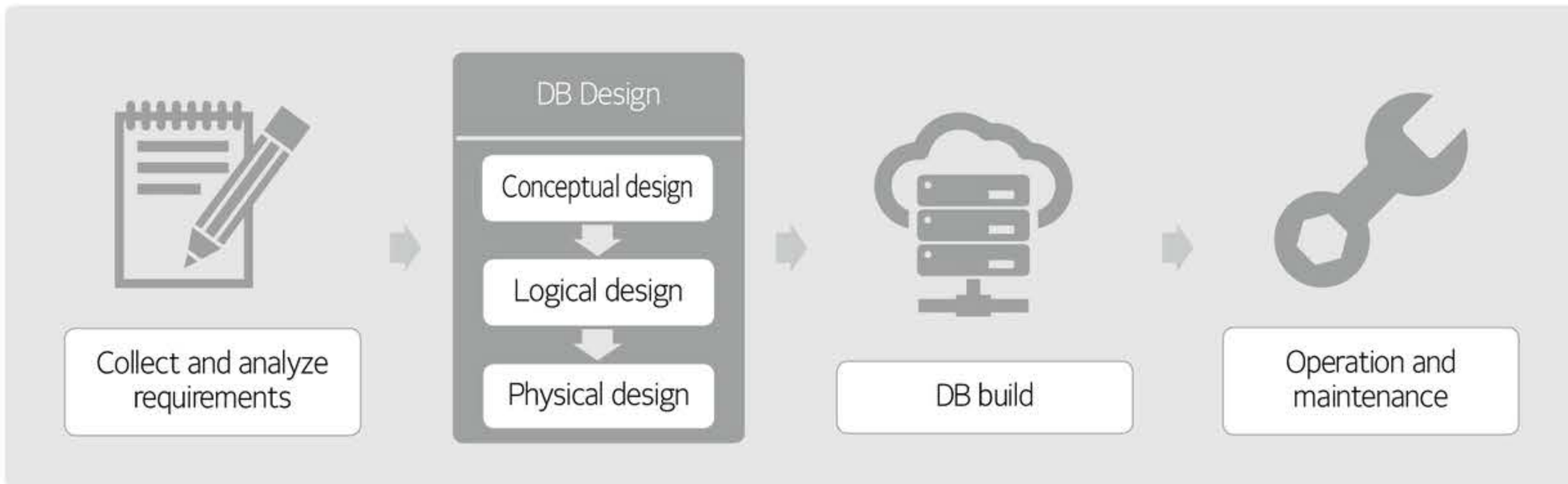
Requirements analysis, conceptual data modeling, logical data modeling, physical data modeling, data modeling subject, deliverables for analysis, impact of design

✚ Preview for practical business

There are many cases in which a database is used like an existing file system, and tables are created in the database, but each table is actually dependent on individual applications, screens, and reports.
For example, when developing a book management system for each team, a booklist table can be created for each team during implementation of database in the same way as books are managed using Excel files. If tables are designed in the same way as Excel files, the advantages of the database, including integration, storage, operation, and sharing, cannot be utilized at all.
In this case, the complexity of the application program also increases, causing symptoms that can lead to serious problems, such as an increase in development costs, problems with data integrity (lack of consistency due to data duplication), and the deterioration of data processing performance. Therefore, these problems should be prevented and the advantages of the database can be maximized and applied to the development of a system only when we understand the definition and characteristics of the database (integration, storage, operation, sharing) and the advantages of the database, and then apply them to practical business.

01 Database Design and Build Process

In general, the database design and build process is performed in the phases of collecting and analyzing the requirements, designing the database, building the database, and operating and maintaining it. In particular, the analysis/design phase is performed in detailed stages, such as conceptual design, logical design, and physical design. The details of the three design phases will be described in greater detail in the next topic [Data modeling].



[Figure 8] Database design and build process

A) Requirements collection and analysis

① Collect and analyze the business requirements of the user.

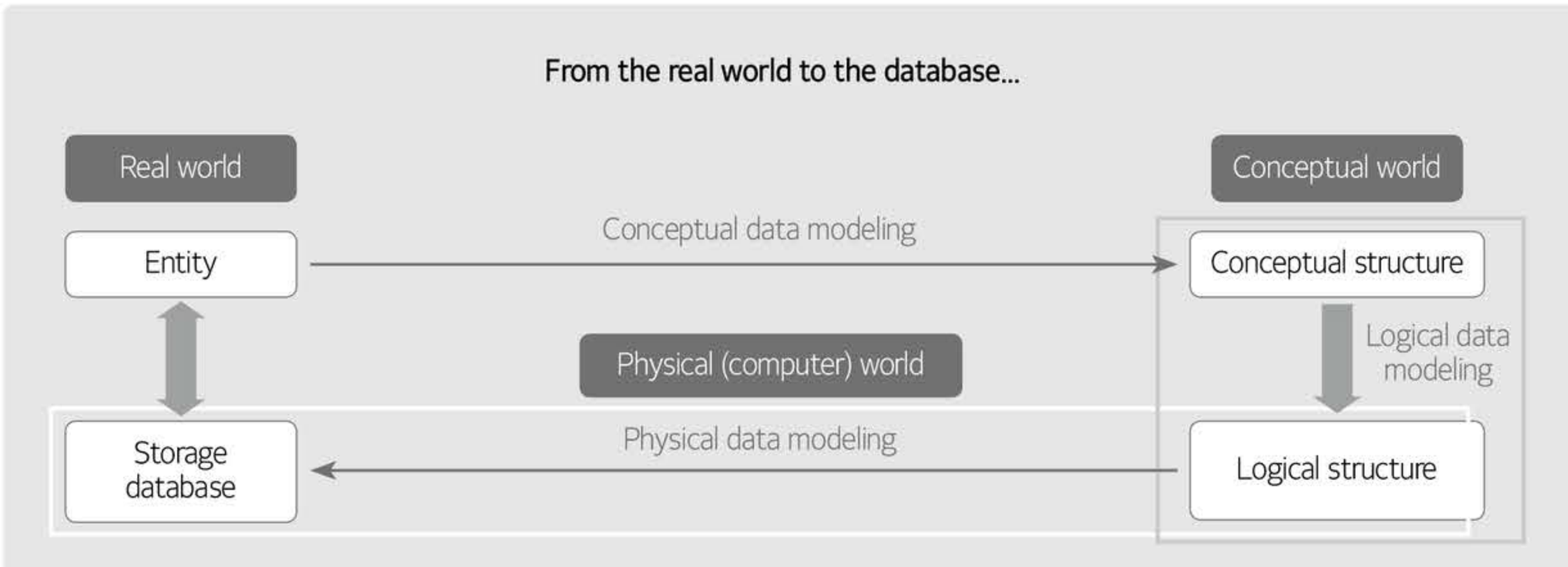
② Create a requirement statement as a phase deliverable.

③ Identify both static and dynamic structural requirements.

- Static structure requirements: Entity, property, relationship, constraint, etc.
- Dynamic structure requirements: Transaction type, frequency, etc.

B) DB design

A data model is a design document for building a database that has a clear goal. In the real world, a database is created over time, as shown below. Depending on the level of abstraction, the design can be divided into a conceptual data model, a logical data model, and physical data.



[Figure 9] Process of creating a database in the real world

First, conceptual data modeling is performed to embody the database at the higher level that has a high level of abstraction in the real world. The conceptual data model has a high level of abstraction, focuses on business, and performs modeling at the comprehensive level. This model is frequently used for enterprise data modeling and EA establishment. For reference, when deploying EA-based enterprise data modeling, general data modeling, which is a higher level, is performed first, after which conceptual data modeling is performed for each business area. When an entity-centered data model is completed at the higher level, a data model with a focus on concrete business can be created according to a concrete business aspect and flow, which is called "logical data modeling". In addition, a method that considers the tablespace according to the storage structure of the database is called "physical data modeling".

<Table 19> Types of data modeling

Data modeling	Contents	Level
A. Conceptual data modeling	Has a high level of abstraction, focuses on business, and performs modeling at the comprehensive level. Often used for enterprise data modeling, and establishing the enterprise architecture	Abstract
B. Logical data modeling	Accurately expresses keys, attributes, relationships, etc. for the work to develop a system, and the data model is highly reusable.	↕
C. Physical data modeling	Design in which physical characteristics are considered in order to improve performance and storage efficiency, so that the data can be actually ported to a database.	Concrete

① Conceptual data modeling

- Abstract conceptualization of the information structure appearing in the real world.
- The entity, entity identifier, relationship between entities, number and sequence of corresponding relationships, entity properties, etc. can be identified.
- Generally represented by an entity-relationship (ER) model.

② Logical data modeling

- Conversion of the result of the conceptual design, which is created for human understanding, into a logical structure that can be easily stored in a database.
- There are the relational, network, hierarchical and object-oriented models, of which the relational model is most generally used.
- In case of the relational model, the table name, primary key (PK), foreign key (FK), etc. are determined in the logical design phase.
- A logical schema is created as a phase deliverable.
- Normalization is performed according to the requirement level.

③ Physical data modeling

- Determination of the physical storage structure of the database created using logical structure design.
- Definition of the data type of the column, constraints, and access method and path for specific data.
- A technology for converting the structure according to the performance requirements is needed.
- Specifically, transaction analysis, view design, index design, capacity design, and approach design are performed.
- A physical schema is created as a phase deliverable.

C) Database building

① Also called the "database implementation" or "database development" phase.

Creation of a database that can accommodate the user requirements by writing and executing SQL statements needed to configure a database.

② Collection and processing of data to be built.

- Classification, index, abstract, etc. are created.

③ Collected and processed data are entered and saved.

D) Operation and maintenance

① Management and monitoring of database quality.

② Establishment of a strategy for database recovery and restoration.

③ Establishment of security policies

④ Continuous maintenance and follow-up evaluation: Deliverables to be analyzed, design subject, impact of design.

02 Considerations for Database Design

Database design is the key area in the process of database build. The possibility of a problem arising can be minimized when proceeding with the next phase only when each phase is clearly arranged and the deliverables are accurately produced.

A) Deliverables to be analyzed

① Considerations in the conceptual model design phase

The meaning of data and the rules of the correlation between data are defined in this phase. A conceptual data model should be created as a deliverable by defining the subject area, key entity, and relationships in this phase.

② Considerations in the logical model design phase

Attributes and keys should be defined based on the previously created conceptual data model; entities should be specified accordingly; and a final entity-relationship diagram should be created after defining history management.

③ Considerations in the physical model design phase

A physical schema is implemented in a physical data model to implement a database. Semi-normalization can be performed in consideration of performance in this phase. A table definition document reflecting these requirements is created.

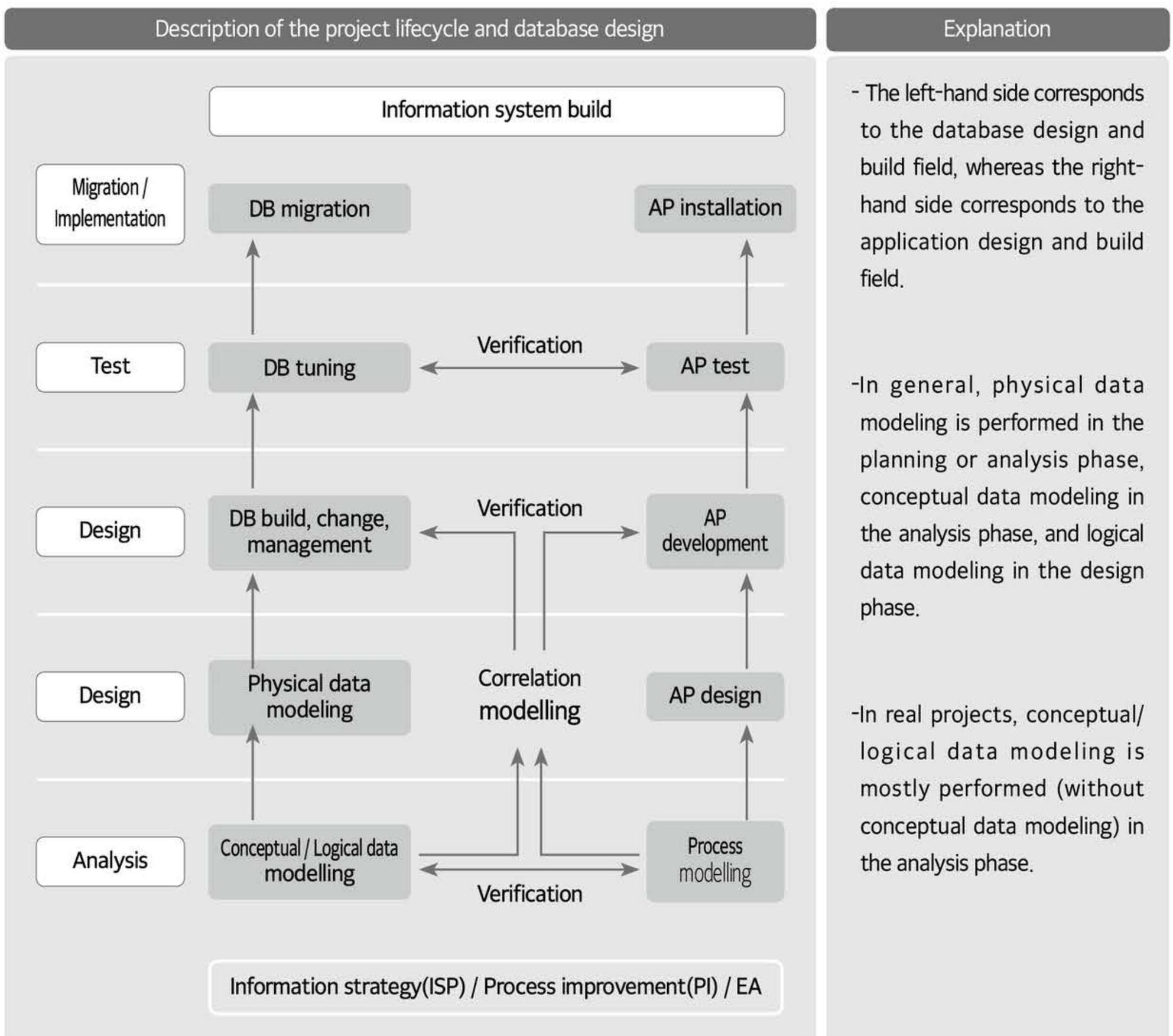
B) Considerations of the design subject

A data architect (DA) plays the most important role in the database design phase. The data architect, who is directly involved from the phase of requirement collection, establishes a data management system and defines the data standards. In addition, the data architect manages the overall data models that are created by conceptual, logical, and physical modeling. However, the performance aspect should be considered in the physical modeling phase, to ensure that the database concerned will work well regardless of whether the user requirements are implemented. At this time, the database administrator (DBA) may review and reflect the physical structure area, such as the indexes, partitions, as well as the clustering storage space, and denormalization.

That is, the data architect and the database administrator should discuss and determine the final physical model which they regard as the best from their own perspectives in the physical modeling phase, so as to achieve efficient and stable performance.

03 Impact of Database Design in the Project Lifecycle

Data modeling can be clearly defined in the project lifecycle because it is separated into the analysis phase and the design phase. Usually, business-oriented logical data modeling is performed in the analysis phase, and physical data modeling is performed by considering hardware and performance in the methodology, based on information engineering or structural models. In the spiral model, the logical data model and the physical data model are performed on both sides according to the size of the task in the analysis and design phase. The logical data model is performed more frequently in the analysis phase.



[Figure 10] Project lifecycle and database design

As a project is implemented by separating it into the data axis and the application axis, matters identified by both sides are mutually verified in order to increase the completeness by phase. However, the object-oriented concept is performed in an integrated way without separating data modeling and processing modeling, because modeling is performed by

considering data and process at the same time. (The representative example is a class that has both data (attributes) and process (method).)

IV. Data Modeling

▶▶▶ Recent trends and major issues

The data modeling technique represented by individual entities and relationships has been used widely. In recent times, however, there has been an increase in practical business techniques that try to express the composition or workflow of a business more elaborately in terms of the data modeling expression method. Among these, the extended data model supplements the shortcomings of the traditional data model which expresses the composition with simple entities and relationships only and has the strengths of the object-oriented data model. This model expresses the object-oriented inheritance structure as a super-type and a sub-type of data modeling, and designates a grouping type that has multiple attribute values in a single entity. An analysis of commonly implemented projects showed that about 70% of them use the super-type and sub-type method. As a result, even though data modeling is deployed, there are many cases in which the physical data model is created without a clear standard. Therefore, concrete data modeling techniques, which developer can understand accurately and apply to practical business, need to be acquired and applied in the field.

Actually, if the following knowledge and techniques, which are frequently mentioned in actual projects, are known or learned, they can be applied usefully.

- It is essential to fully know and apply the normalization rules, which are the starting point of creating a database.
- Do not casually think that inquiry performance is slow in normalized models.
- Denormalization technology should be applied based on a proper understanding.
- A user-independent conceptual schema should be applied to data modeling from the viewpoint of data independence.
- The meaning and effect of identifier relationships and non-identifier relationships should be understood before applying them.
- Conversion should be performed after understanding the concepts of super-type and sub-type and after considering performance.
- Do not create a data model with a missing relationship.
- History data modeling (occurrence, change, progress) in consideration of performance should be available.
- When setting primary keys, the performance and index characteristics should be considered.

▶▶▶ Learning objectives

1. To be able to explain the concept of data modeling.
2. To be able to design a database according to the data modeling procedure.
3. To be able to design entities, attributes, and identifiers in data modeling according to the business requirements.
4. To be able to design relationships in data modeling according to the business requirements.
5. To be able to explain and solve the problem of the connection trap.
6. To be able to explain how to convert object-relational mapping (ORM).
7. To be able to explain how to express the Entity Relationship Diagram (ERD).
8. To be able to explain the extended entity-relationship (ER) type.
9. To be able to explain data integrity.
10. To be able to designate primary keys and foreign keys.

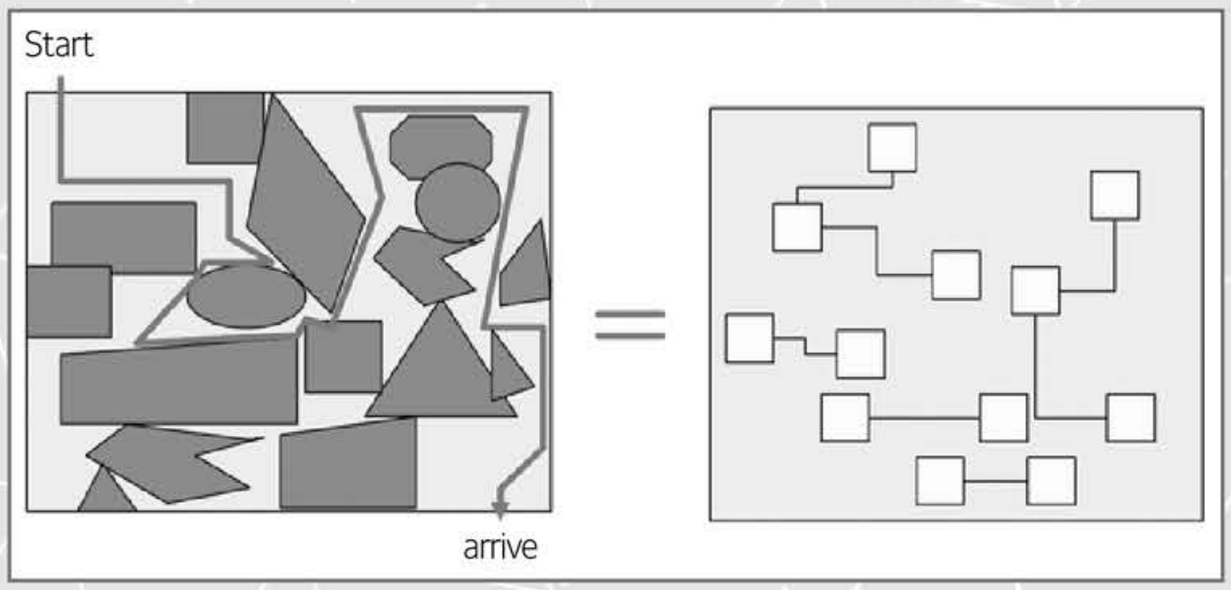
▶▶▶ Keywords

- Data modeling, entity, property, relationship, identifier, connection trap, ERD, extended ER

✚ Preview for practical business

A data model that is as complex as a maze can ruin a project. A complex maze, like the one shown below left, means that countless trials and errors will be required to find a fast path. Such a data model does not consider business characteristics and physical characteristics. As a result, the model has to undergo countless trials and errors to obtain the desired result because the path to the data is not clear. Also, even if the same number of paths is passed, join operations are generated by checking additional conditions only, without passing through the optimized path. In the end, multiple join operations further degrade performance.

In addition, if an entity is expressed unnecessarily in large numbers, data are processed inefficiently because an SQL statement should read several tables, while the intended result can be obtained by reading and processing data from one table alone.



If we check the data only, the two kinds of problems that arise due to increased data can be summarized as follows:

- Increased data → Data duplication → Problems occur in data consistency.
- Increased data → Fall in SQL response speed → Performance declines.

These two problems become more important as the number of data grows, and can be solved by optimizing data modeling, which is the ideal way to fundamentally resolve the problem. However, data modeling cannot be performed at any time. Therefore, when data modeling is performed again, it should be optimized for efficiently processing data.

01 Concept and Procedure of Data Modeling

"Data modeling" refers to the process of abstracting the real world to create a database and is performed according to the following procedure: requirements analysis, conceptual modeling, logical modeling, and physical modeling. That is, modeling refers to a technique of expression that uses a certain notation to abstract, simplify, and clarify the real world.

<Table 20> Characteristics of modeling

Item	Contents
Abstraction	Expresses the real world according to a certain format.
Simplification	Expresses the real world using agreed protocols, or limited notation and language.
Clarification	Removes ambiguity and accurately describes a phenomenon so that everyone can understand it easily.

A) Requirements collection and analysis

- A phase in which ambiguity is removed from the requirements by collecting and analyzing the business requirements.
- A required specification (or job description) is produced as a deliverable.

B) DB design

Design in the order of conceptual modeling -> logical modeling -> physical modeling.

C) Differences in the academic world's and industry's understanding of conceptual/logical modeling

① Understanding the academic world

- Conceptual modeling is the phase in which an ERD is produced, whereas logical modeling is the phase in which the structural drawing of the table is produced. These two phases are clearly separated.
- Therefore, the entities and relationships of the ERD are identified in the conceptual modeling phase, and the primary keys and foreign keys of the table are identified in the logical modeling phase.
- Normalization that handles table structure should be performed in the logical modeling phase.
- For reference, "physical modeling" refers to the process of creating tables suitable for an actual DBMS.
- The Chen type of ERD is mainly used.
- The relationship itself can have its own properties in the ERD.
- All information shown in the table structure drawing is also shown in the ERD.

② Understanding of the industry

- An ERD is produced during logical modeling.
- Conceptual modeling further abstracts the ERD and summarizes it at a comprehensive level including the subject area, core entity, core relationship, and core attribute. Conceptual modeling is not studied in the academic world.
- The Crow's Foot type of ERD is mainly used.

- If the relationship has its own attributes, they are included in the connected entity or included by identifying a relationship entity.
- For your reference, "physical modeling" refers to the process of converting an ERD into a table structure.
- In general, normalization is performed in the logical data modeling phase, and denormalization is performed in the physical data modeling phase.

<Table 21> Differences in the understanding of data modeling

	Academic world	Industry
The time of ERD production	Conceptual modeling	Logical modeling
Main notation	Chen type	Crow`s Foot
Normalization phase	Performed in the logical modeling phase.	Generally, normalization is performed in the logical modeling phase, while denormalization is performed in the physical modeling phase for the purpose of improving performance.
When the relationship has attributes	The relationship itself can have attributes.	Not allowed. Separating into relationship entities or switching to the existing entity attribute.
Physical modeling	The process of creating tables that are suitable for the actual DBMS.	The process of converting an ERD into a table structure.

D) Concept of process modeling

"Process modeling" refers to the task of finding missing or unnecessary tasks to satisfy the functions required by the target system, by analyzing a given task efficiently within a specified period of time, when analyzing large and complex tasks. In addition, it is regarded as most important to develop a high-quality system by classifying the analyzed tasks systematically and by designing the menu of the target system and program system.

- When a task is analyzed using a methodology, the task can be arranged systematically in order to find most of missing or unnecessary tasks, which is helpful when creating a data model.
- The process hierarchy diagram is a frequently used process model; and the task process is subdivided into the function area, function, process, and unit process according to the size of the task.

E) Concept of correlation modeling

Correlation modeling is a method of analyzing what is happening (process) to something (data) that exists within the task concerned and what is affected by which event, when developing an information system.

- Correlation modeling is performed by using the entity type identified from data modeling and the unit process identified from process modeling.
- Whether or not data are generated by the entity type and the unit process can be verified using the CRUD matrix, which enables the developer to verify the task for the data model and the process model in the analysis phase, and which can be used to understand the test and task data exchange in the future development phase.

02 Data Modeling Procedure

A) Conceptual modeling

- The phase of abstractly conceptualizing and expressing the information structure appearing in the real world.
- Conceptual modeling begins with analyzing and finding the data requirements of the organization and user, and includes the process of determining which data are important and the form of maintaining that data.
- Such data requirements are used to describe the phases to be performed in subsequent phases, to give shape to the conceptual data model, which is a set of low relevance, and to discuss the initial requirements with business stakeholders.
- Finding the core entities and the relationships between them is most important, and the relationships should be expressed as an entity relationship diagram.
- The entity, entity identifier, relationship between entities, number and sequence of corresponding relationships, entity properties, etc. can be identified.

B) Logical modeling

- Logical modeling is the key part of the data modeling process, which is a technique or process that clearly expresses the logical structure and rules of business information.
- In this phase, the result of conceptual design, which is created for human understanding, is converted into a logical structure that can be easily stored in a database.
- The phase in which normalization is performed to store data efficiently.
- The relational model is mainly used, and the deliverables of conceptual modeling are converted into the table structure drawing. Table name, primary key, foreign key, etc. are defined.

C) Physical model

- The phase that plans how to express logical data modeling in a computer hardware system as a data repository.
- The structure of physical storage that is expressed in tables and columns, the storage device to be used, and the approach used to extract data are determined in the physical modeling phase.
- Performance is one of the most important factors to consider in this phase, and a denormalization review is considered.
- The data type, constraint, and index of the column are defined.

03 Types of ER Notations

In 1976, Peter Chen created a form of notation for data models called an entity-relationship model (ER model). This notation uses a rectangle, a diamond, and an ellipse to represent an entity, a relationship, and an attribute respectively. This notation is widely used when learning theories about data modeling. The main college courses in Korea use Chen's model of notation when teaching database design. The following section describes the various notations used to denote entities, properties, and relationships.

Notation	Description
Chen Team — 1 — Include — N — Employee	- Used with many college teaching materials. - Hardly used for practical business
IDEF1X Team — Employee	- Used by the US Ministry of National Defense. - Applied by ERWin.
IE / Crow's Foot Team — Include / Belongs to — Employee	- Most frequently used notation that uses the Crow's Foot shape. - Widely used, including ERWin, RStudio, etc.
Min-Max / ISO Team — (0,N) Include / (1,1)Belongs to — Employee	- A description is written for the relationshiap name. - Rarely used.
UML <<Entity>> Team — 1 — <<Relationship>> Include employees — 0..N — <<Entity>> Employee	- Entities are expressed using stereotypes. - Used for data modeling by expressing it in UML.
Case * Method Team — 1 — 0..N — Employee	- Partly different (relationship description, etc.) from Crow's Foot when it is applied (Barker's Notation). - DA#

[Figure 11] Data model notation

04 ER Notation Based on the Chen Model

The ER model presented by Chen was the first data modeling method to represent data configuration and correlation. This model is useful for understanding the theoretical concepts of data modeling.

Element	Symbol	Element	Symbol	Element	Symbol
Entity		Attribute		Derived attribute	
Weak entity		Key attribute		Complex attribute	
Relationship		Partial key attribute			
Identifying relationship		Multiple value attribute			

[Figure 12] ER model notation

A) Entity

① An entity refers to a meaningful unit of information in the real world.

② An entity includes both physical objects and conceptual objects.

- Physical object: student, car, classroom
- Conceptual object: project, job, subject

③ In general, an entity has its own identifier (key attribute) and is represented using a rectangle.

④ An entity without its own identifier is called a "weak entity", which is represented using a two-line rectangle.

B) Relationship

① A relationship refers to the correlation between entities and is represented using a diamond

Example) The relationship between [student] and [course] is represented as follows.

[Figure 13] Relationship notation

② The degree of the relationship represents the number of entities participating in the relationship.

- Unary, binary, ternary, ⋯, N-ary

③ The cardinality of the relationship represents the maximum number of relationship instances that can participate in the relationship.

- There are one-to-one, one-to-many and many-to-many relationships.

Cardinality	Expression example
One-to-one (1:1)	Student — 1 — Department representative — 1 — Department
One-to-many (1:N or N:1)	Student — N — Major — 1 — Department
Many to many (M:N)	Student — M — Taking a class — N — Subject

[Figure 14] Cardinality of the relationship

④ The relationship connecting the weak entity and the identifying entity is called an "identifying relationship" and is distinguished from the general relationship by marking it with a two-line diamond.

C) Attribute

① An attribute represents the intrinsic nature of an entity or relationship and is represented using an ellipse.

② Identifier (key attribute)

- An identifier is represented by underlining the attribute's name.
- An attribute or a set of attributes that always has a unique value in all entity sets (e.g. student ID, vehicle registration number, etc.)
- When an identifier is made by combining several attributes, the attributes concerned are combined first to make a complex attribute, and are then marked by underlining it.
- If there are multiple attributes that can be an identifier, mark all of them by underlining them.

③ Weak entity separator (partial key attribute)

As a weak entity does not have an identifier, it should be connected to another entity that plays the role of an identifying entity, and the identifier of the identifying entity and some attributes of the weak entity are combined to be used as the identifier. The attributes of the weak entity used at this time are called separator or partial key attributes.

④ Multi-valued attribute

A multi-valued attribute can have multiple values at the same time, which are distinguished from general attributes (single-valued attribute) by marking them with a two-line ellipse

⑤ Derived attribute

A derived attribute refers to an attribute that can be derived from other stored data and is distinguished from general attributes (storage attribute) by marking it with a dotted ellipse.
Example) The number of people in each department can be derived by counting the number of employees in the department.

⑥ Composite attribute

A composite attribute refers to an attribute that can be decomposed into two or more elements, and is distinguished from general attributes (simple attribute) by displaying the links between the attributes.

05 Extended ER (EER: Extended Entity-Relationship) Model

The EER model adds several useful concepts to the basic ER model.

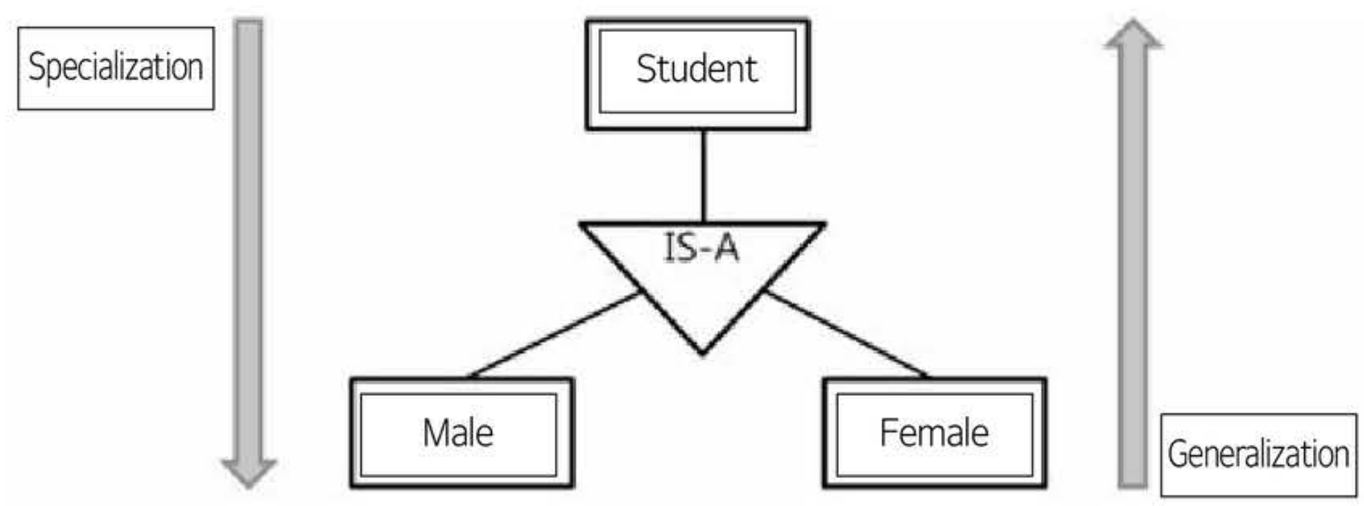
A) Generalization/Specialization

① Generalization: Separation of one entity type (super type) into multiple sub-level entity types (sub-type).

② Specialization: Integration of multiple entity types (sub type) into one upper-level entity type (super-type).

③ Generalization and specialization have an inverse relationship. The generalization/specialization relationship is called the IS-A relationship.

④ The concept of inheritance applies.

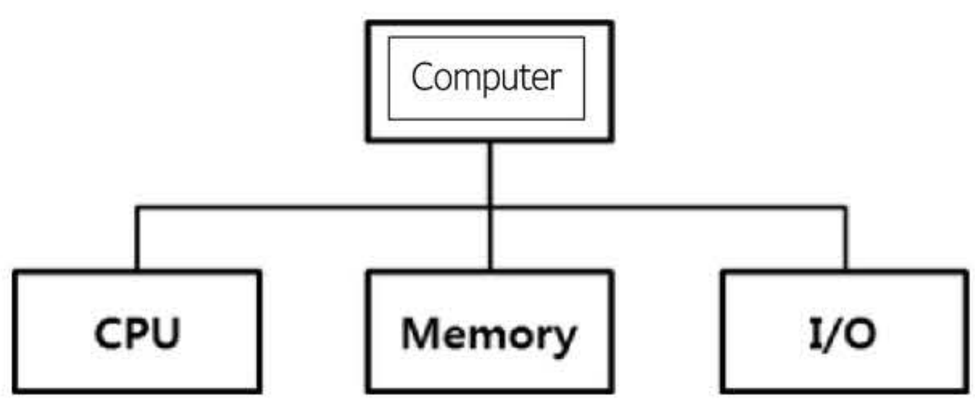


[Figure 15] Generalization/Specialization

B) Aggregation

① Aggregation defines a new entity with a set of multiple entities. It is called an IS-PART-OF relationship.

② The concept of inheritance does not apply.



[Figure 16] Aggregation

06 Another Type of ERD Notation: The Crow's Foot Model

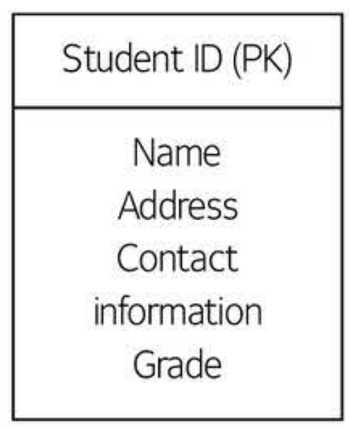
Chen's ERD (described above) is mainly used in the academic world, whereas information engineering notation, also known as Crow's Foot notation, is mainly used in industry.

A) Entity, attribute, identifier

① Specify the entity name outside the rectangle.

② Specify the attribute name inside the rectangle.

③ The rectangle is divided into top/bottom. Specify the identifier attribute at the top and the general attributes at the bottom.



[Figure 17] Crow's Foot notation of entity, property, and identifier

B) Characteristics of the entity

① Information must be required by the work concerned and should be the information to be managed. (Example: Patient, number of times TOEIC exams are taken, etc.)

② An entity should be identifiable by a unique identifier.

③ An entity should be a set of instances that exist permanently.
(Not "one", but "two or more")

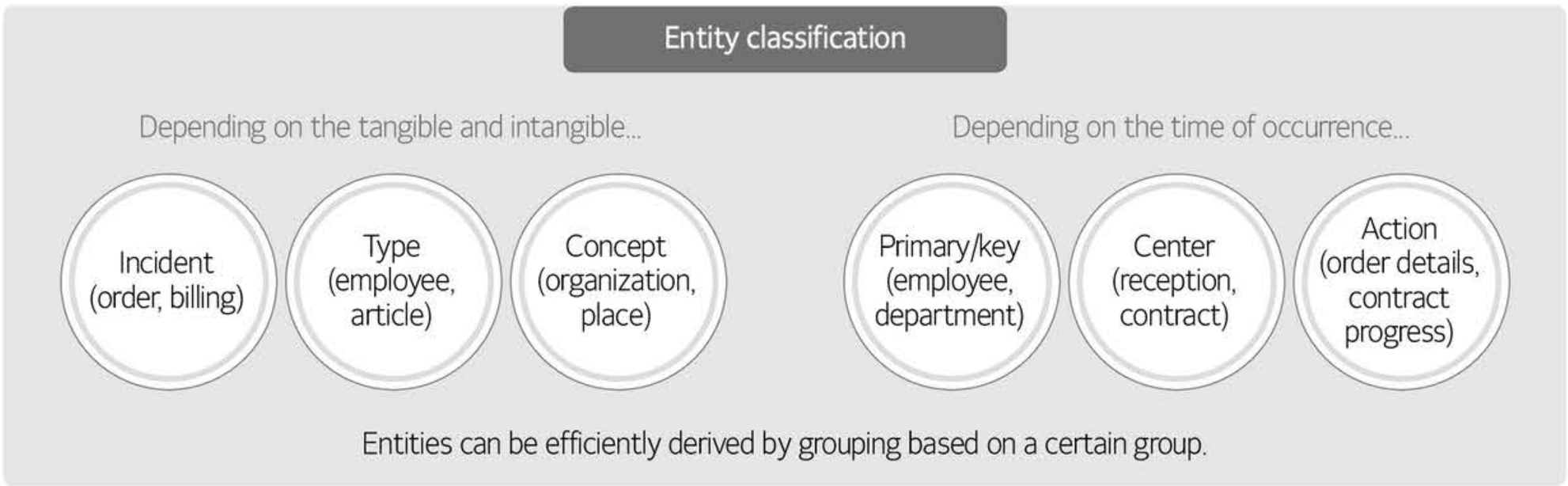
④ An entity should be used by the work process.

⑤ An entity must have an attribute.

⑥ An entity should have at least one relationship with other entities.

C) Classification of entities

① Classification by tangible and intangible type - Generally, entities can be classified into tangible entities, concept entities, and event entities depending on whether they are tangible or intangible.
Classification by the time of occurrence - Entities can be classified into fundamental entities/key entities, main entities, and active entities depending on the time of occurrence.



[Figure 18] Data model expression method

D) Classification of attributes

Attributes can be classified into the basic attribute which is directly defined by analyzing the work, the designed attribute which is identified during design even though it does not exist in work, and the derived attribute which is calculated or transformed from other attributes.

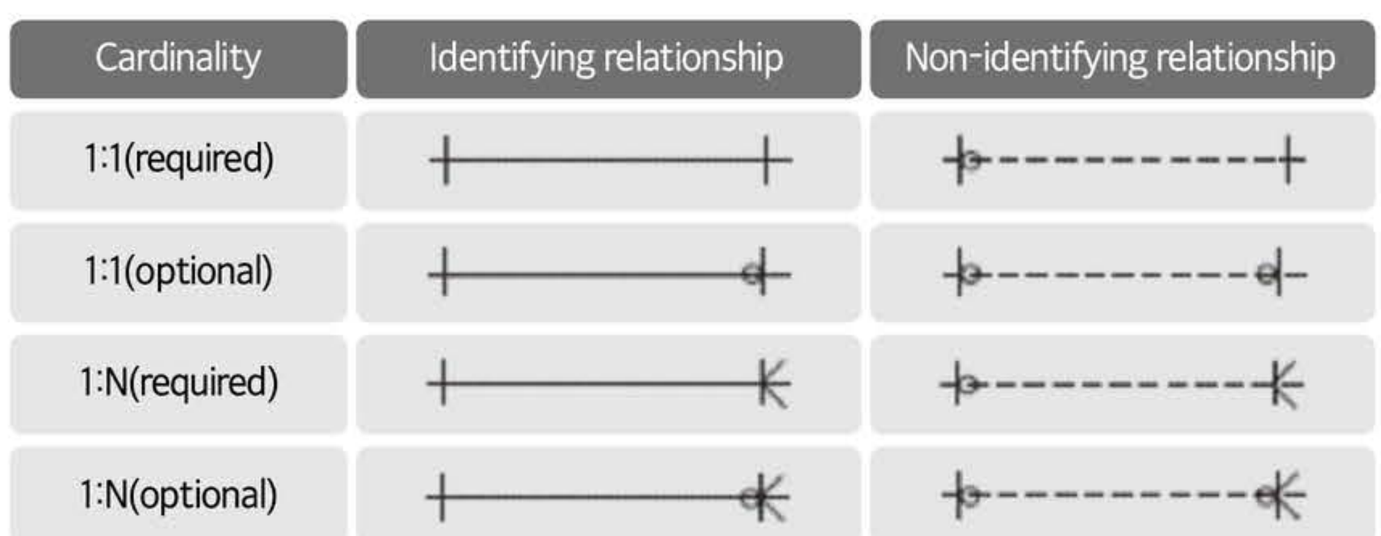
[Figure 19] Classification of attributes

E) Relationship

① The identifying relationship and non-identifying relationship are marked clearly.

② The required relationship and the optional relationship are marked clearly.

③ Unlike Chen notation, the attribute of the relationship is included in an entity or by identifying a separate entity.

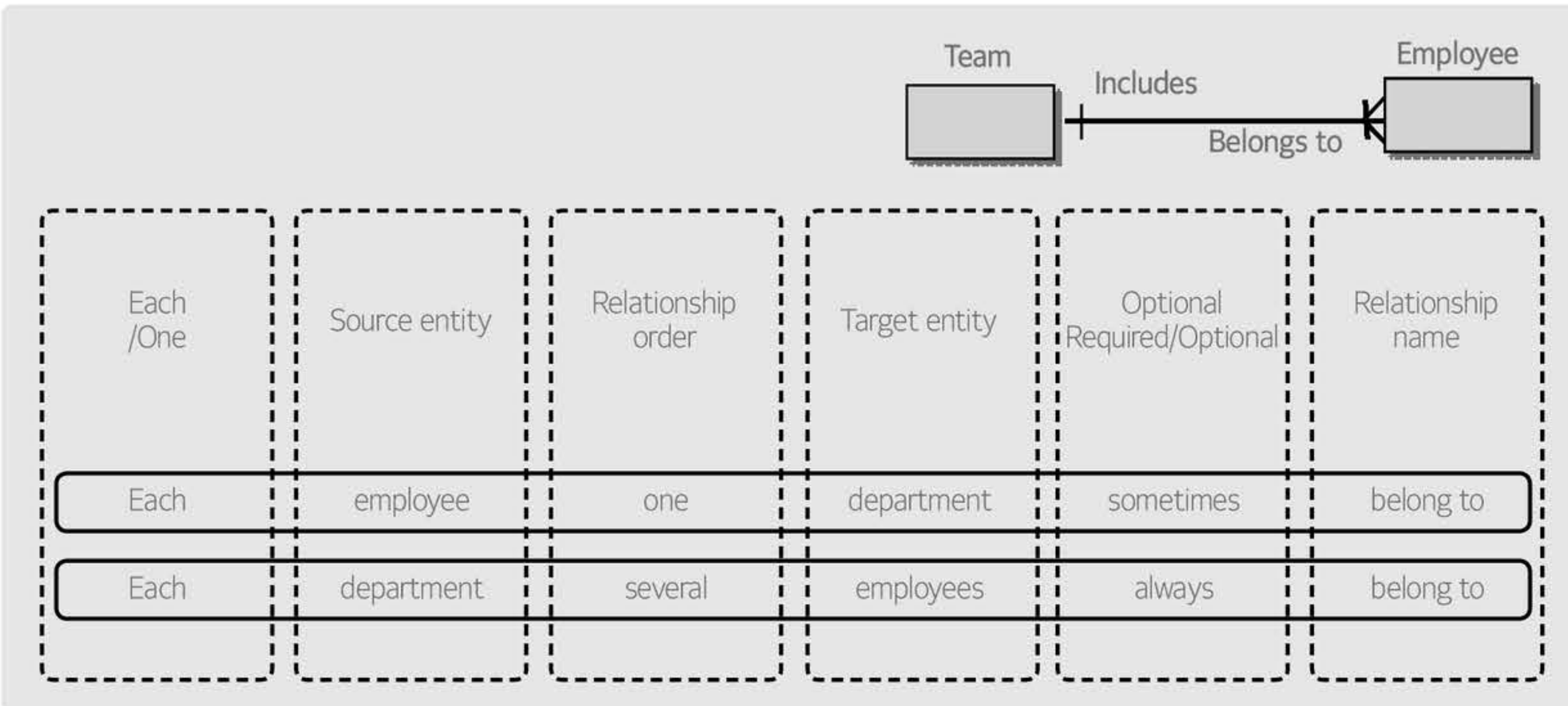


[Figure 20] Relationship expression in Crow's Foot notation

F) How to read relationships

To read a data model, the identifier reads one or each source entity participating in the relationship, followed by the number of target entities (one, more than one), relationship options, and relationship names.

- The identifier reads one or each source entity.
- The identifier reads the degree of the target entity's participation in relationships, that is, the number (one, more than one).
- The identifier reads the relationship option and the relationship name.



[Figure21] Questioning method for identifying the relationship

G) Characteristic of the identifier

An identifier will have different characteristics depending on whether it is a "primary identifier" or a "foreign identifier". First, if it is a "primary identifier", it has the following characteristics.

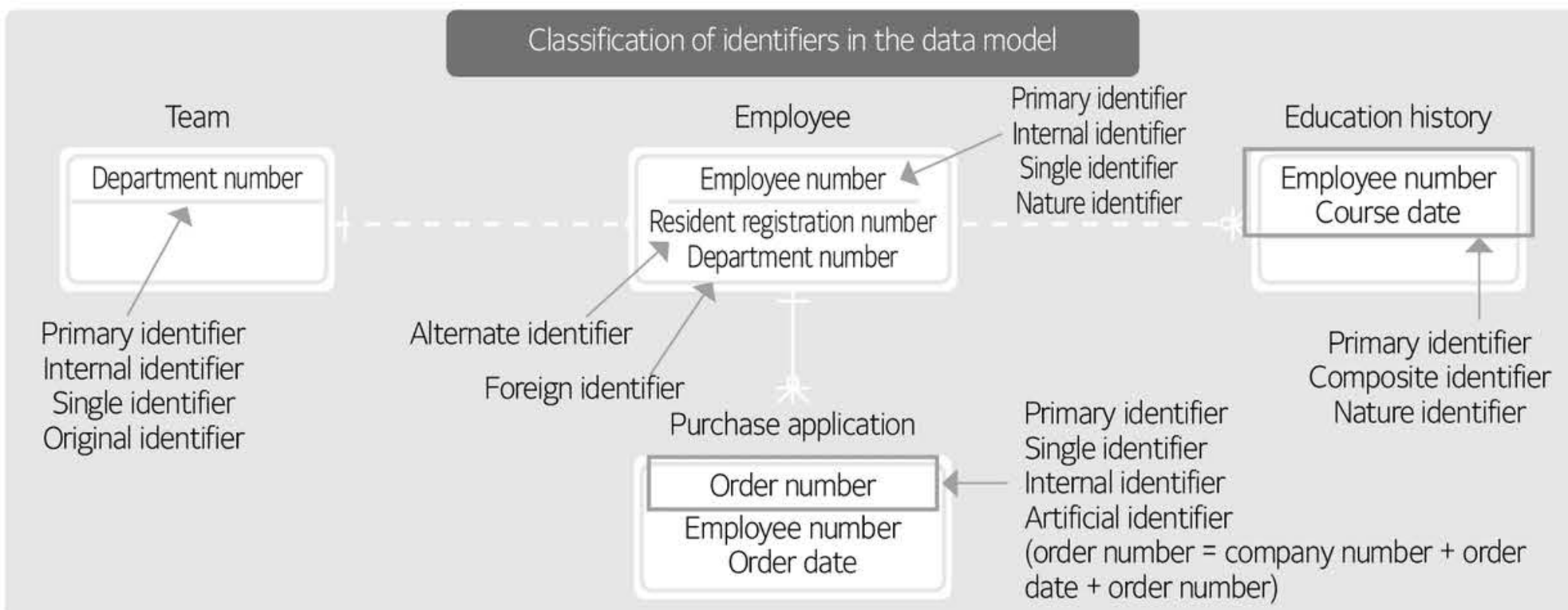
<Table 22> Characteristics of the primary identifier

Characteristics	Contents	Remark
Uniqueness	A primary identifier uniquely distinguishes all instances within an entity.	Example) An employee number (primary identifier) is uniquely assigned to each employee.
Minimality	The number of attributes constituting a primary identifier should be the minimum number that satisfies uniqueness.	Example) If an identifier is composed of the employee classification code + the employee number while only the employee number has a unique structure, the primary identifier has an inappropriate structure.
Invariability	If a primary identifier has been assigned to a specific entity, the value of the identifier should not be changed.	Example) If the value of an employee number has changed, it means that the previous record has been erased and a new record will be created.
Presence	If a primary identifier has been designated, there must be a data value. (Null is not allowed.)	Example) A company employee must have an employee number.

H) Classification of identifiers

<Table 23> Classification of identifiers

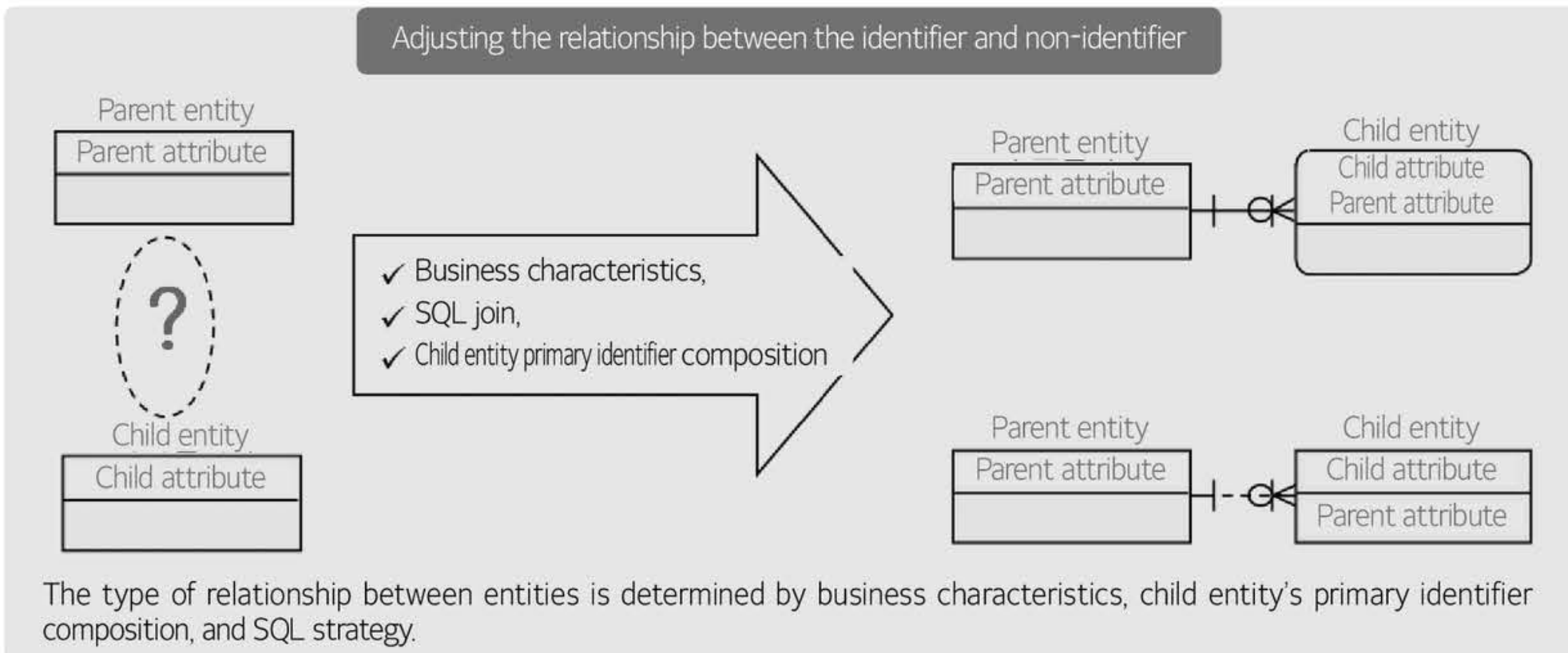
Classification	Identifier	Description
Representa-tiveness	Primary identifier	A separator that can distinguish each occurrence within an entity, and an identifier that can connect a reference relationship with other entities.
	Alternate identifier	A separator that can distinguish each occurrence within an entity, but which cannot connect the reference relationship because it has no representativeness.
Self-generation	Internal identifier	An identifier that is created by itself inside an entity.
	Foreign identifier	An identifier that is received from other entities based on the relationship with other entities.
Number of attributes	Single identifier	An identifier consisting of one attribute.
	Composite identifier	An identifier consisting of more than two attributes.
Replacement status	Natural identifier	An identifier created by work.
	Artificial identifier	An identifier that is not created for work, but which is artificially created because the original identifier has a complex configuration.



[Figure 22] Classification of identifiers in the data model

I) Identifier relationship and non-identifier relationship

A foreign identifier is an attribute created in a child entity through its relationship with other entities, not an attribute needed by a given entity itself. The foreign identifier acts as a foreign key when creating a database. The foreign identifier can be naturally identified by a logical relationship when the relationship and attribute, as well as the primary identifier, are defined. When a primary identifier is assigned to an entity and a relationship between the entities is established, the child entity's attributes are inherited from the parent's primary identifier. At this time, it should be determined whether the child entity will use the foreign identifier received from the parent identity as its own primary identifier or not, or whether it will use it only as a property that is connected to the parent.



[Figure 23] Adjustment of the identifier/non-identifier relationship

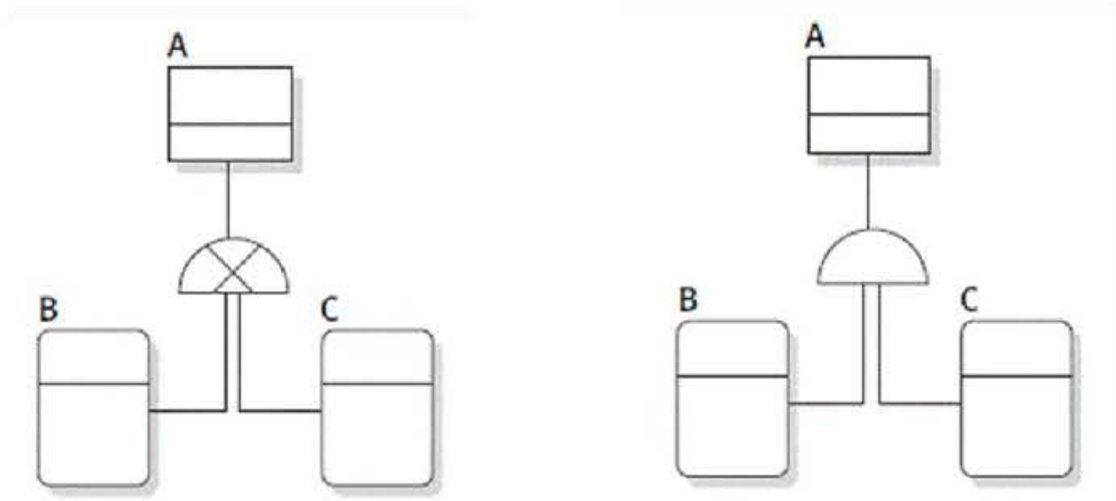
The following table compares the identifier relationship and the non-identifier relationship..

<Table 24> Identifier relationship and non-identifier relationship

Item	Identifier relationship	Non-identifier relationship
Purpose	To express a strong relationship.	To express a weak relationship.
Impact on the child's primary identifier	Included in the composition of the child's primary identifier.	Included in the child's general attributes.
Notation	Indicated by a solid line.	Indicated by a dotted line.
Considerations for connection	- Must be dependent on the parent entity. - The parent's primary identifier needs to be included in the composition of the child's primary identifier. - The inherited primary identifier's attributes need to be transferred to another entity.	- Weak subordinate relationship. - Configures the child's primary identifier independently. - Composition of the parent's primary identifier is needed in the composition of the child's primary identifier. - The inherited primary identifier's attributes need to be blocked from other entities. - Parents' participation in the relationship is optional.

J) Super-type and sub-type

① Exclusive sub-type: A super-type is associated with one sub-type only. ([Figure 26(a)])
② Comprehensive sub-type: A super-type can be associated with more than two sub-types. ([Figure 26(b)])



(a) Exclusive subtype **(b)** Comprehensive subtype

[Figure 24] Sub-type expression in Crow's Foot notation

07 Connection Trap

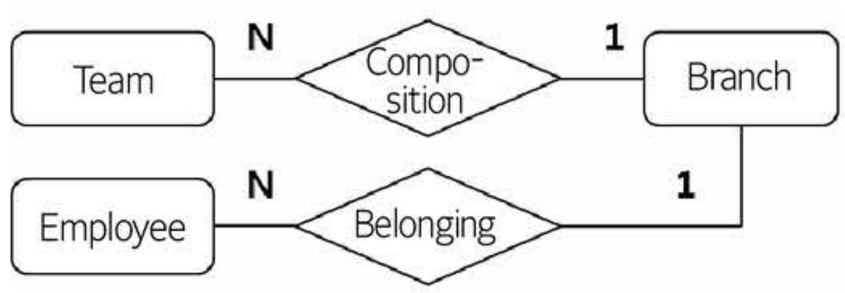
The connection trap means that desired information cannot be accurately found even though a relationship has been established in a data model. Representative types include the fan trap and the chasm trap.

A) Fan trap

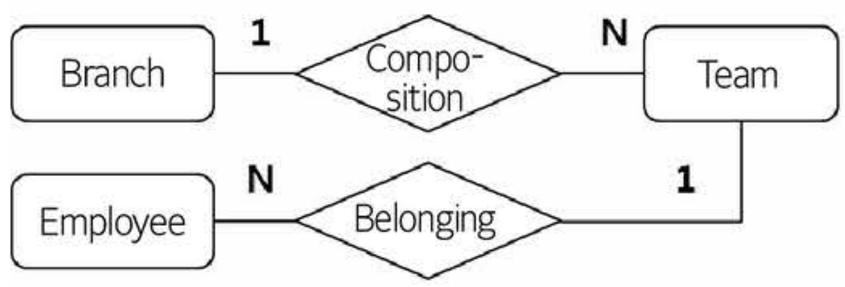
① The fan trap can occur when there are entity types A, B, and C, and when there is an N:1 relationship between entities A and B, and a 1:N relationship between entities B and C.

② An example of the fan trap is as follows.

- [Figure 25(a)] shows a case in which each branch is composed of several departments, with several employees belonging to each branch.
- The fan trap is found in this case because the department to which each employee belongs cannot be known.
- The fan trap can be removed by modifying it, as shown in [Figure 25(b)].



(a) Fan traps exist



(b) Fan traps are removed

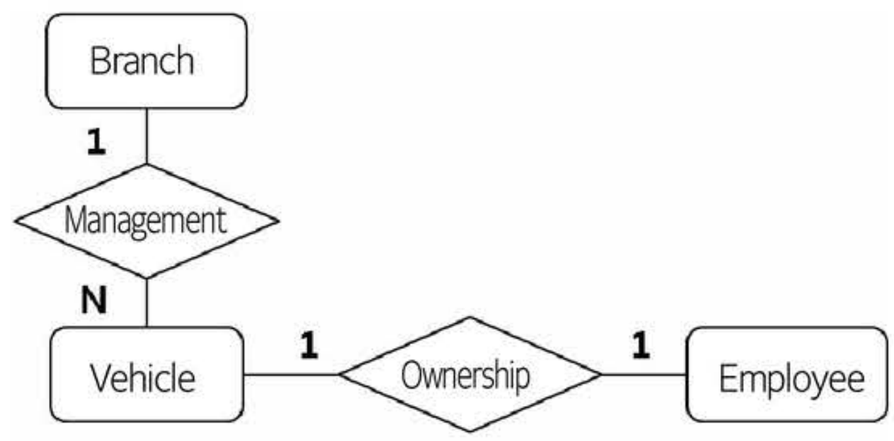
[Figure 25] Fan trap

B) Chasm trap

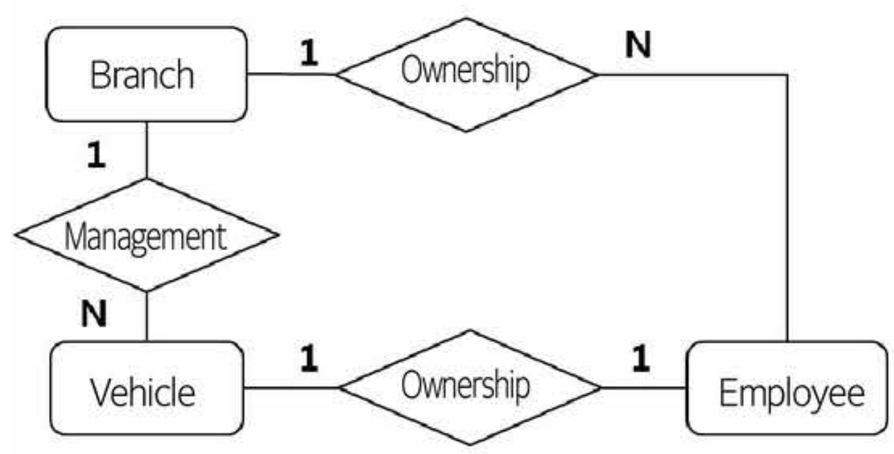
① The chasm trap means that desired information cannot be found due to the disconnection of the information flow when there is an optional relationship rather than a mandatory relationship.

② ([Figure 26]) shows an example of the chasm trap.

- [Figure 26(a)] shows a case in which several branches are registered in one branch, with each vehicle being owned by one employee.
- In this case, we can know to which branch the employee who owns a vehicle belongs, but we cannot know to which branch the employee who does not own a vehicle belongs.
- To eliminate the chasm trap, a relationship that is missing, even though it is required in terms of meaning, should be explicitly linked again ([Figure 26(b)]).



(a) Chasm traps exist



(b) Chasm traps are removed

[Figure 26] Chasm trap

* However, in [Figure 26] some employees own a vehicle, and some do not.

08 Object-Relational Mapping (ORM)

Object-relational mapping means that the table of the relational database corresponds to the class used in the object-oriented design.

A) Class conversion

① Each class corresponds to a table.

② The instance object of a class corresponds to the record of a table.

③ The attribute of a class corresponds to the column of a table.

④ Class operation has no corresponding element and is implemented in the application program.

B) Conversion of the class relation into a relational relationship

① Association

- 1: M relationship - the primary key on side 1 is mapped on to the foreign key on side M.
- 1:1 relationship - the primary key of the side that is less frequently accessed is mapped on to the foreign key of the side that is most frequently accessed, or the two classes are combined into one class.
- M: N relationship - a new class corresponding to the relationship is created.

② Aggregation

The table concerned is referenced using the foreign key.

③ Composition

Implementing using a constraint

④ Generalization

- The super class and each sub-class are mapped on to separate tables.
- (Or) A super-class has all attributes of the sub-class.
- (Or) A sub-class has the attribute of the super-class.

09 Integrity and Key

Data integrity refers to the need to protect data from lost updates of data to maintain their accuracy, validity, consistency, and stability. Three to five types of integrity are defined depending on the viewpoint. Three types of integrity include entity, referential, and domain integrity. Five types of integrity are also defined by separating entity integrity into entity integrity and key integrity, and by including user-defined integrity. To main integrity in this way, the primary and foreign keys should be carefully selected and used.

A) Integrity

① Domain integrity: Domain integrity means to ensure the integrity of the field in a table. Data types, allowing a null value, etc. can be defined and used. The attribute value has atomicity and should be a value defined in the domain.

② Key integrity: All records in a table should be identifiable from each other.

③ Entity integrity: All tables must have a primary key, and each primary key must have a unique value (but not null).

④ Referential integrity: Data in two tables in a reference relationship must always have consistent values. A foreign key must be a null value or a value that exists in the primary key of the table referenced by the foreign key. If there is a

foreign key that refers to a primary key, the data cannot be deleted or changed.

[Figure 27] Referential integrity

⑤ User-defined integrity: A specific work rule, which does not fall into other integrity categories, is defined.

B) Key

① Super key: A sub-set of all fields in a table. It is an attribute or set of attributes that can uniquely identify a record.

② Candidate key: A candidate for the primary key. It is a super key that satisfies uniqueness and minimality. That is, if any attribute is removed from a super key, the super key loses its nature and becomes a candidate key.

③ Primary key: A unique identifier selected from among candidate keys to distinguish a specific record in the table. Like the candidate key, the primary key has the attribute of uniqueness and minimality, but a null value is not allowed. If there are multiple candidate keys, a primary key is selected by considering the following two points.

- Selecting a field in which values are not frequently changed: Each time the value of the field selected as the primary is changed, it should be checked whether the value concerned is a unique value or not. Therefore, a column without frequent change is more suitable.
- Selecting a field in which values are not relatively complex: If a field selected as a primary key has a complex data domain, the cost of comparing the primary key increases. Therefore, it is more suitable to select a domain with a relatively short and simple form.

④ Foreign key: When Table A refers to Table B, the foreign key of Table A refers to a key that can uniquely identify a record of Table B. A foreign key consists of one field or a subset of all fields, and cannot have duplicate values or a null value. Also, you must refer to a field that has a unique value in Table B.

V. Normalization and Denormalization

Recent trends and major issues

The "normalization" theory is the foundation of completing a system and is one of the most important theories that need to be understood when building a system in the field. The theory of first normalization is the starting point for normalization, which removes duplication from a file system in order that data can be processed stably. This situation occurs frequently in the actual project site, so it is necessary to learn the details in depth. The first, second, and third Boyce-Codd normalizations are performed based on function dependency, whereas the fourth and fifth normalizations are performed by the semantic association between attributes. Data should be managed stably when building a system. Normalization is an important theory that everyone should know when developing an information system, because normalization is the most important and basic theory for ensuring stable data management.

Learning objectives

1. To be able to explain the concept and characteristics of file dependency.
2. To be able to explain abnormal normalization.
3. To be able to design a database by applying normalization.
4. To be able to explain denormalization.
5. To be able to explain the points to be taken into consideration for performance design.

Keywords

- Normalization removes anomalies by removing data redundancy.
- Denormalization is a design technique for improving query performance.

✚ Preview for practical business How should normalization be utilized in an actual project?

It is quite rare that data modeling is performed as a series of tasks, such as identifying entities through normalization, and setting and diversifying the relationship. It is also rare to prove an anomaly by assigning functional dependency to each table, as we did when learning the theory of normalization. Why, then, is it so important that we should learn the theory of normalization and remember the theory of normalization?

Learning and utilizing normalization for practical business can be summarized into two points.
The first point is that those who learn the theory of normalization accurately can identify objects and establish a relationship based on the idea of normalization, even though normalization is not performed according to the normalization procedure.
That is, even though data modeling is not performed according to the process of normalization, those who perform data modeling are clearly aware of the theory and are naturally aware of functional dependency. Therefore, they perform data modeling from that perspective. Conversely, those who do not know the concept of normalization frequently perform data modeling that goes against the first, second, and third normalizations because they lack the concept of normalization. It can be understood that normalization is utilized by so-called "internalized inner forces".

The second point is that the completeness of the completed data model is verified using the theory of normalization. It is really important to test whether the entire data model has stability for anywhere from 50 to 2,000 entities and relationship attributes. At this time, criteria are needed to test this, and it becomes the most important theory of tool normalization for the criteria in the data model. When we diagnose the data models of a project that has approximately 500 entities, 10 to 40 normalization violations can be found. As such, many unknowingly made normalization violations are found in many projects. Besides artificial denormalization, there are many models that are not normalized without forecasting the result. Data integrity is lost because such a model is applied to table creation and then to the operating environment. As a result, the quality of the business data is deteriorated.
Therefore, it should be remembered that it is also used as a tool to verify the completeness of the data model that has been designed based on the theory of normalization.

01 Normalization and Anomalies

Various anomalies can occur when performing data processing operations in a database that has been poorly designed without strictly considering the relationship between data attributes, particularly dependencies between attributes. Representative anomalies include an insertion anomaly, deletion anomaly, and update anomaly. These anomalies are explained in <Table 25>.

<Table 25> Table showing mixed student information and department information (primary key = "student ID")

Student ID	Name	Gender	Department name	Department office
1111	Hong Gildong	Male	Computer engineering	Engineering building
2222	Ryu Gwansun	Female	Computer engineering	Engineering building
3333	Gang Gamchan	Male	Physics	Natural science building

A) Insertion anomaly

① Definition: An anomaly in which unintended information is also inserted when inserting certain information.

(Example) When inserting the information (Department name = "Management information", Department office = "Management Building") into the table in <Table 25>.

- Entity integrity is lost if only the department name and department office values are inserted, because the primary key ("Student ID") becomes null.
- To maintain entity integrity, the department name and department office information must be inserted together with the student ID of the student belonging to the Department of Management Information Technology.

B) Deletion anomaly

① Definition: An anomaly in which necessary information is also deleted when deleting certain information.

(Example) When deleting the information of (Student ID="3333") in <Table 25>.

- If this student is the only student belonging to the "Physics Department", the information that the department office of the "Physics Department" is in the "Natural Science Building" is also deleted.

C) Update anomaly

① Definition: An anomaly in which the same content should be updated for several data repeatedly when modifying certain information.

(Example) When the office of the "Department of Computer Science and Engineering" is changed to the "Future Building" in <Table 25>.

- Modification operations should be performed as many times as the number of students belonging to the "Department of Computer Engineering".

02 Concept of Functional Dependency and Rule of Inference

The anomalies described above can be prevented by decomposing each table based on specific criteria. Functional dependency is one of the most useful criteria that can be applied when decomposing a table in this way and is expressed as the relationship between a determinant and a dependent. Additional functional dependency can be inferred based on the previously discovered functional dependency. At this time, Armstrong's Axioms are used.

A) Functional dependency

① Definition: There are the subsets X and Y of the field defined in the table R. At this time, if the values of X of the arbitrary record pair t1 and t2 are the same, and the Y values of those two records are always the same, Y is functionally dependent on X.

② Expression: X →Y (Dependent Y is functionally dependent on determinant X.)

- (Example) (Resident registration number) → (Name)
- If the resident registration numbers of two records are the same in the table about people, it means that the names of these two records are always the same.

<Table 26> Types of functional dependency

Type	Concept
Full functional dependency	Full functional dependency There is no attribute X that satisfies X'⊂X and X'→Y.
Partial functional dependency (2NF)	Full functional dependency There is no attribute X that satisfies X'⊂X and X'→Y. Employee number / Employee name / Building number / Start date 1789 / Park Jiseong / 15 / 2000 / 07 / 12 1412 / Lee Youngpyo / 23 / 1999 / 02 / 19 1789 / Park Jiseong / 19 / 2000 / 05 / 24 1412 / Lee Youngpyo / 32 / 2001 / 04 / 21 Employee name ← Employee number, Building number → Start date Partial functional dependency Complete functional dependency <Complete/Partial functional dependency - Used in 2NF>
Transitive dependence (3NF)	Transitive Dependence If attributes A→X and X→Y in relation R, then A→Y. Supplier number / Place of location / Transport distance S1 / Suwon / 46 S2 / Daejeon / 164 S3 / Daegu / 302 S4 / Gwangju / 329 Supplier number → Place of location → Transport distance; Supplier number → Transport distance <Transitive functional dependency - Used in 3NF>>
Boyce-Codd Normal Form (BCNF)	Boyce-Codd normalization If the determinant that is functionally dependent is not a candidate key. That is, X is not a candidate key in X→Y.

B) Armstrong's axioms

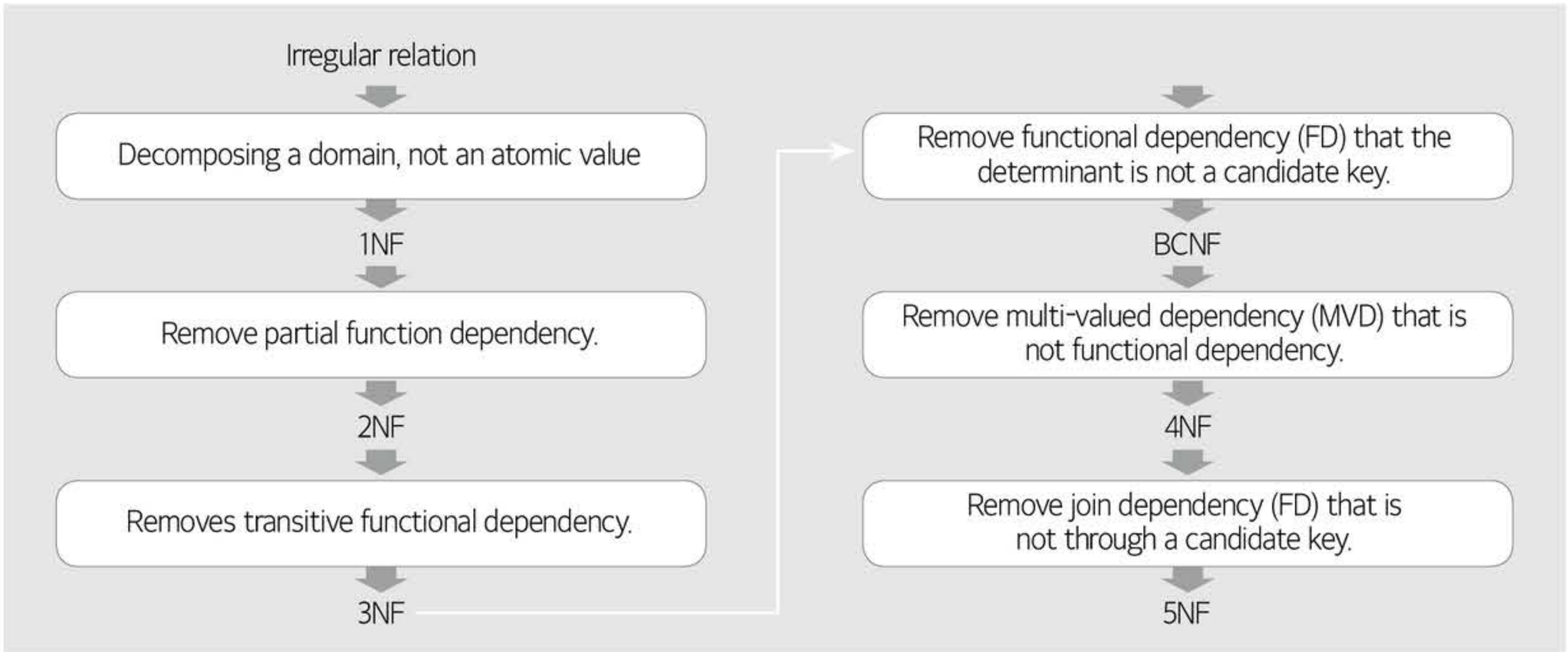
<Table 27> Armstrong's axioms

Item	Inference	Contents
Armstrong's axioms	Reflexivity rule	If $Y \subseteq X$, then $X \rightarrow Y$.
	Augmentation rule	If $X \rightarrow Y$, then $XZ \rightarrow YZ$.
	Transitivity rule	If $X \rightarrow Y$, and $Y \rightarrow Z$, then $X \rightarrow Z$.
Other derived rules	Union rule	If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow YZ$.
	Decomposition rule	If $X \rightarrow YZ$, then $X \rightarrow Y$, and $X \rightarrow Z$.
	Pseudo-transitivity rule	If $X \rightarrow Y$ and $WY \rightarrow Z$, then $XW \rightarrow Z$.

03 Database Design in Which Normalization is Applied (1st, 2nd, 3rd, Boyce-Codd)

The table derived by normalization is called the "normal form" table. The first normal form (1NF), second normal form (2NF), third normal form (3NF), Boyce-Codd normal form (BCNF), fourth normal form (4NF), and fifth normal form (5NF) have been introduced in both the academic world and practical business. In particular, it is known that the third normal form and the Boyce-Codd normal form are frequently utilized.

A) Normalization process ([Figure 28])



[Figure 28] Normalization process (1st, 2nd, 3rd, Boyce-Codd normal form)

B) Example of normalization

① First normalization ([Figure 29])

- The table on the left is an un-normalized table, because the "Courses" field does not have atomic values.
- The first normal form table, shown in the table on the right, can be derived by decomposing the attribute "Courses"

and designating the composite key of "Student ID" and "Courses" as the primary key.

Student ID	Name	Subject name
1111	Hong Gildong	Database, operating system
2222	Gang Gamchan	Operating system, network, data structure

Student ID	Name	Subject name
1111	Hong Gildong	Database
1111	Hong Gildong	Operation system
2222	Gang Gamchan	Operation system
2222	Gang Gamchan	Network
2222	Gang Gamchan	Data structure

[Figure 29] Example of the first normalization

② Second normalization ([Figure 30])

- The table on the left is not the second normal form.
- The reason is that there is a partial functional dependency of the (course code) → (course name) in the function dependency of the (student ID) → (course name).
- The second normal form table is derived by dividing the un-normalized table into two tables, as shown on the right.

Student ID	Name	Subject code	Subject name	Grade
1111	Hong Gildong	D11	Database	A
1111	Hong Gildong	O22	Operation system	B
2222	Gang Gamchan	O22	Operation system	A
2222	Gang Gamchan	N33	Network	A
2222	Gang Gamchan	D44	Data structure	B

Student ID	Name	Subject code	Grade
1111	Hong Gildong	D11	A
1111	Hong Gildong	O22	B
2222	Gang Gamchan	O22	A
2222	Gang Gamchan	N33	A
2222	Gang Gamchan	D44	B

Subject code	Subject name
D11	Database
O22	Operation system
O22	Operation system
N33	Network
D44	Data structure

[Figure 30] Example of the second normalization

③ Third normalization ([Figure 31])

- The table on the left is not the third normal form.
- The reason is that there is transitive dependency of the (student ID) → (department), (department) → (department office).
- The third normal form table is derived by dividing the un-normalized table into two tables, as shown on the right.

Student ID	Name	Department	Department office
1111	Hong Gildong	Computer engineering	Engineering building
2222	Gang Gamchan	Computer engineering	Engineering building
3333	Ryu Gwansun	Physics	Natural science building

Student ID	Name	Department
1111	Hong Gildong	Computer engineering
2222	Gang Gamchan	Computer engineering
3333	Ryu Gwansun	Physics

Department	Department office
Computer engineering	Engineering building
Computer engineering	Engineering building
Physics	Natural science building

[Figure 31] Example of the third normalization

④ Boyce-Codd normalization ([Figure 32])

- It is assumed that there is a functional dependency relationship between (professor, course name) → (text material name), (text material name) → (course name) in the table on the left.
- In this case, the table on the left is not the Boyce-Codd normal form.
- The reason is that there is functional dependency (text material name) → (course name) wherein the determinant is not a candidate key.
- The Boyce-Codd normal form table is derived by dividing the un-normalized table into two tables, as shown on the right.

Professor	Subject name
P1	Data structure
P1	Network
P2	Network
P2	Programming
P3	Programming

Textbook name	Professor	Textbook name
Book1	P1	Book1
Book2	P1	Book2
Book3	P2	Book3
Book4	P2	Book4
Book4	P3	Book4

Textbook name	Subject name
Book1	Data structure
Book2	Network
Book3	Network
Book4	Programming

[Figure32] Example of Boyce-Codd normalization

04 Database Design in Which Normalization is Applied - (4th Normalization)

A) Concept of the fourth normalization

The fourth normalization is the process of removing more than two multi-valued dependencies (MVD) if they occur in one relation.

B) Characteristics of the fourth normalization

① The fourth normalization is possible only when there is a semantic association constraint between attributes.

② It is established when all attributes are primary identifiers. (Primary key)

③ Multi-valued dependency can be established when there are three or more associated attributes.

④ It corresponds to normalization by semantic association, not by the functional dependency relationship.

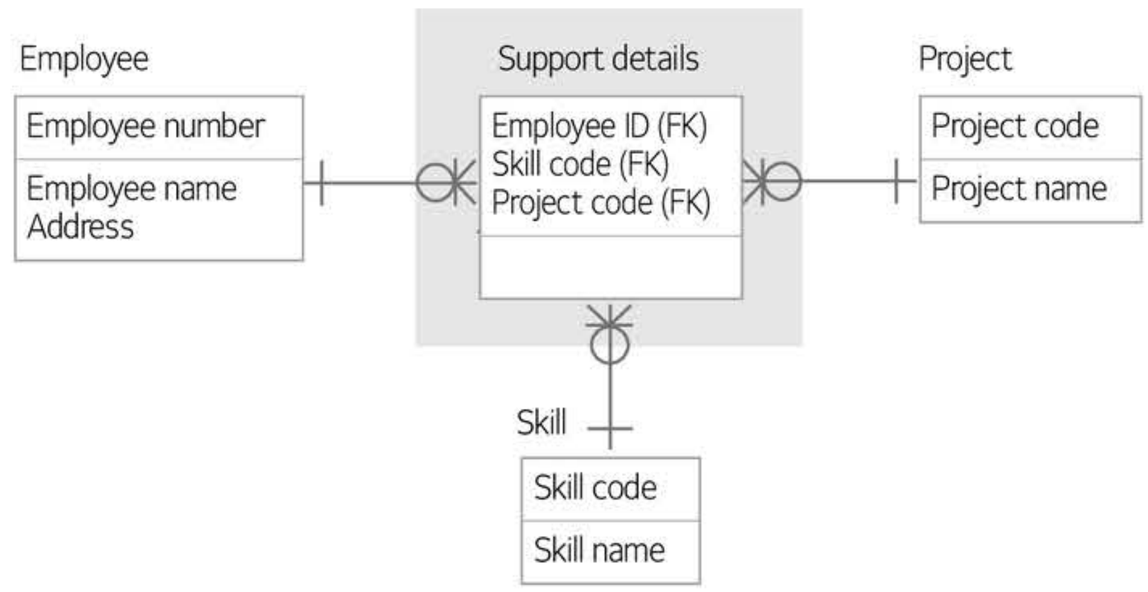
⑤ When there are three attributes, A, B, and C, there is a semantic relationship as follows. However, an anomaly occurs when creating a table using these attributes.

- Relationship between A and B: A and B are related.
- Relationship between A and C: A and C are related.
- Relationship between C and B: C and B are not related.

C) Targets of the fourth normalization

Preconditions for table design
- If you want to manage the holding technology and the support project of the employee, but the holding technology and support project are not related.
That is, the employee and the holding technology are related (O). The employee and the project are related (O). The holding technology and the project are not related (X).

- If only one table is designed despite these business rules, the following data model and structure of data values are identified.

<table>
<tr><th>Item</th><th>Structure of the table's data values</th></tr>
<tr><td>Data model</td><td> 
</td></tr>
<tr><td>Structure of the table's data values</td><td><table>
<tr><th>Employee number</th><th>Technology code</th><th>Project code</th></tr>
<tr><td>10</td><td>{ MODELING
DBA }</td><td>{ SI
OO
PA }</td></tr>
<tr><td>20</td><td>{ DBA
XML }</td><td>{ PA }</td></tr>
</table></td></tr>
</table>

[Figure 33] Example of the fourth normalization

- The focus of this example is placed on the relationship, but on the multi-valued dependency in which each data value in the support detail table produces several data values due to the previous value. It can be expressed as follows, if it is expressed as a table in a database.

Employee number	Technology code	Project code
10	MODELING	AND
10	MODELING	OO
10	DBA	PA
20	DBA	PA
20	XML	PA

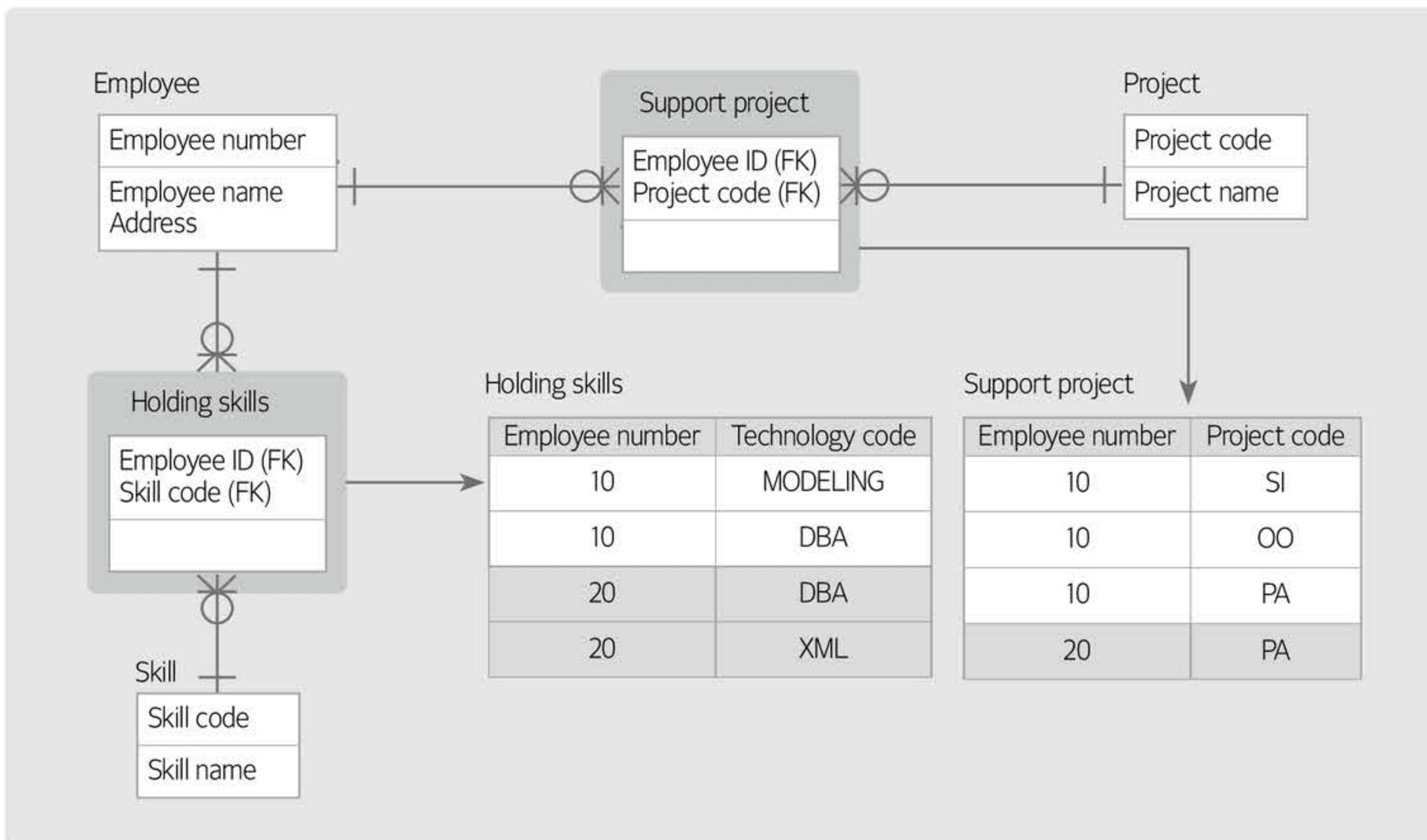
• Input error, modification error, and deletion error can occur even without performing the fourth normalization.

<Table 28> Classification of anomalies

Type	Anomaly
Input error	When an employee is assigned to a new project, the technical code should be entered regardless of the project. At this time, if the employee has supported several projects, the technology code should be entered as many times as the number.
Modification error	If the project code experienced by an employee needs to be changed, it should be changed as many times as the number of holding technologies.
Delete error	When the technology held by an employee is deleted, the project to which the employee was assigned is also deleted, making the project history unknown. Also, when deleting the holding technology, the modification should be made as many times as the number of technologies used for the project.

D) Performing the fourth normalization

The fourth normalization can be performed as follows.



[Figure 34] Performing the fourth normalization

<Table 29> Association rules

Rule	Association rule	Table
1	An employee supports a project. (A and B are related.)	Created
2	An employee has a certain technology. (A and C are related.)	Created
3	The technology used in the project is not set. (B and C are not related.)	Not created

05 Database Design to Which Normalization is Applied - (5th Normalization)

A) Concept of the fifth normalization

- If all join dependencies that exist in relation R can be satisfied by the candidate key of relation R, relation R is called the fifth normal form.

B) Characteristics of the fifth normalization

① Normalization occurs in the primary key.

② There should be a join relationship between independent attributes.

③ Created after being separated into associative relations.

④ When there are three attributes, A, B, and C, there is a semantic relationship as follows. However, an anomaly occurs when creating a table using these attributes.

- Relationship between A and B: A and B are related.
- Relationship between A and C: A and C are related.
- Relationship between C and B: C and B are related.
- Relationship between A, B, and C: There is no relationship in which A, B, and C are all related.

C) Performing the fifth normalization

① When there is an attribute that has the primary of SN, CM, and PN, it can be created as a table, as follows.

- SN: ID of the supplier (PK)
- CN: ID of the part (PK)
- PN: ID of the project (PK)

② The association preconditions of each identifier are as follows.

- A supplier has an association in which a certain part is supplied. (A and B are related.)
- There is an association in which the supplier has supported a certain project. (A and C are related.)
- There is an association in which a certain part has been used for a certain project. (B and C are related.)
- However, there is no association about how a certain supplier has used a certain part for a certain project. (A, B, and C are not related.)
- Despite the above constraints, it is designed as a single table, as follows.

SN	CN	PN
S1	C1	P2
S1	P2	P1
S2	C1	P1
S1	C1	P1

- If it is designed as a single table despite these preconditions, input error, modification error, and deletion error can occur even without performing the fifth normalization.

<Table 30> Classification of anomalies

Type	Anomaly
Input error	If a supplier wants to input new association information for a part, the supplier should consider the association between the supplier and project and the association between the part and project.
Modification error	To modify the association information for a supplier and a part, multiple records should be modified because other associations should also be considered.
Delete error	If the association information for a supplier and a part is deleted, the association information for a supplier and a project, as well as the association information for a part and a project, are also deleted.

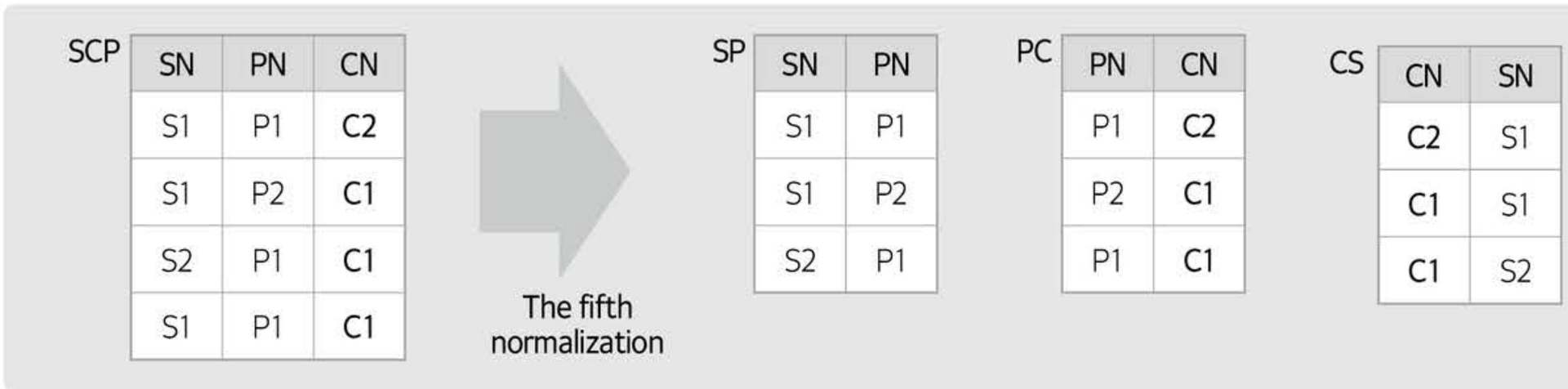
D) Performing the fifth normalization

- As there is a precondition of the four same semantic associations, it can be analyzed so as to create the following table.

<Table 31> Association rule table

Rule	Association rule	Table
1	A supplier handles a certain part. (A and B are related.)	Created
2	It can be known whether a part has been used for a certain project. (A and C are related.)	Created
3	A supplier supports a certain project. (B and C are related.)	Created
4	It cannot be known whether a supplier supports a certain project using a certain part. (A, B, and C are not related.)	Not created

- The fifth normalization based on the above prerequisites means creating a join table (relational table) as follows.



[Figure36] Performing the fifth normalization

06 Denormalization

A) Concept and procedure of denormalization

① Definition: The process of integrating data models for normalized entities, attributes, and relationships in order to improve system performance and simplify development and operations.

② Considerations: Data consistency and integrity → System performance and data model simplicity

③ Procedure of denormalization

- Investigating denormalization targets: Range processing frequency, large range processing, aggregation process,

and number of table joins.

- Considering inducing other methods: View creation, clustering application, index adjustment, and application development.
- Applying denormalization: Denormalization of tables, columns, and relationships.

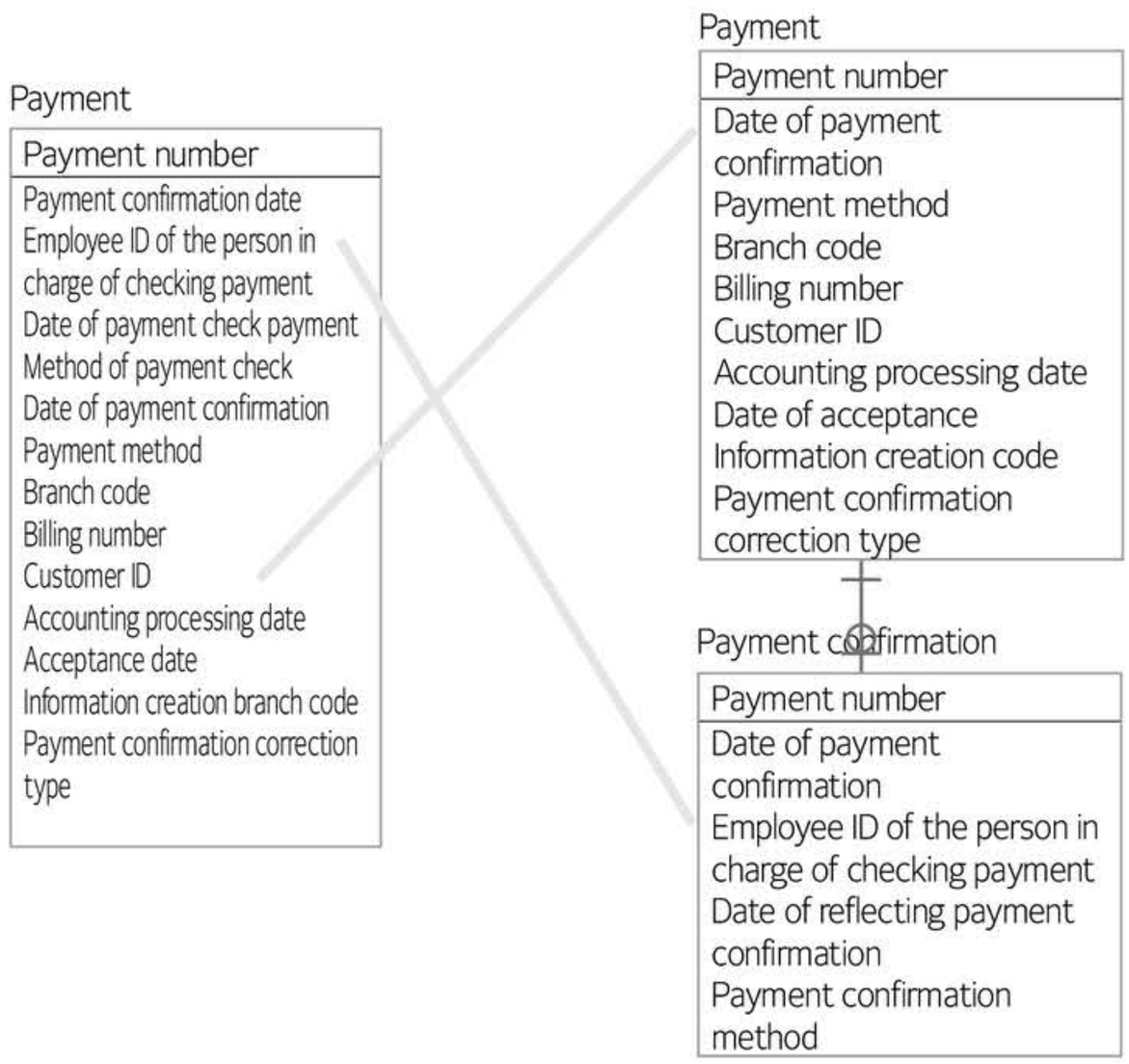
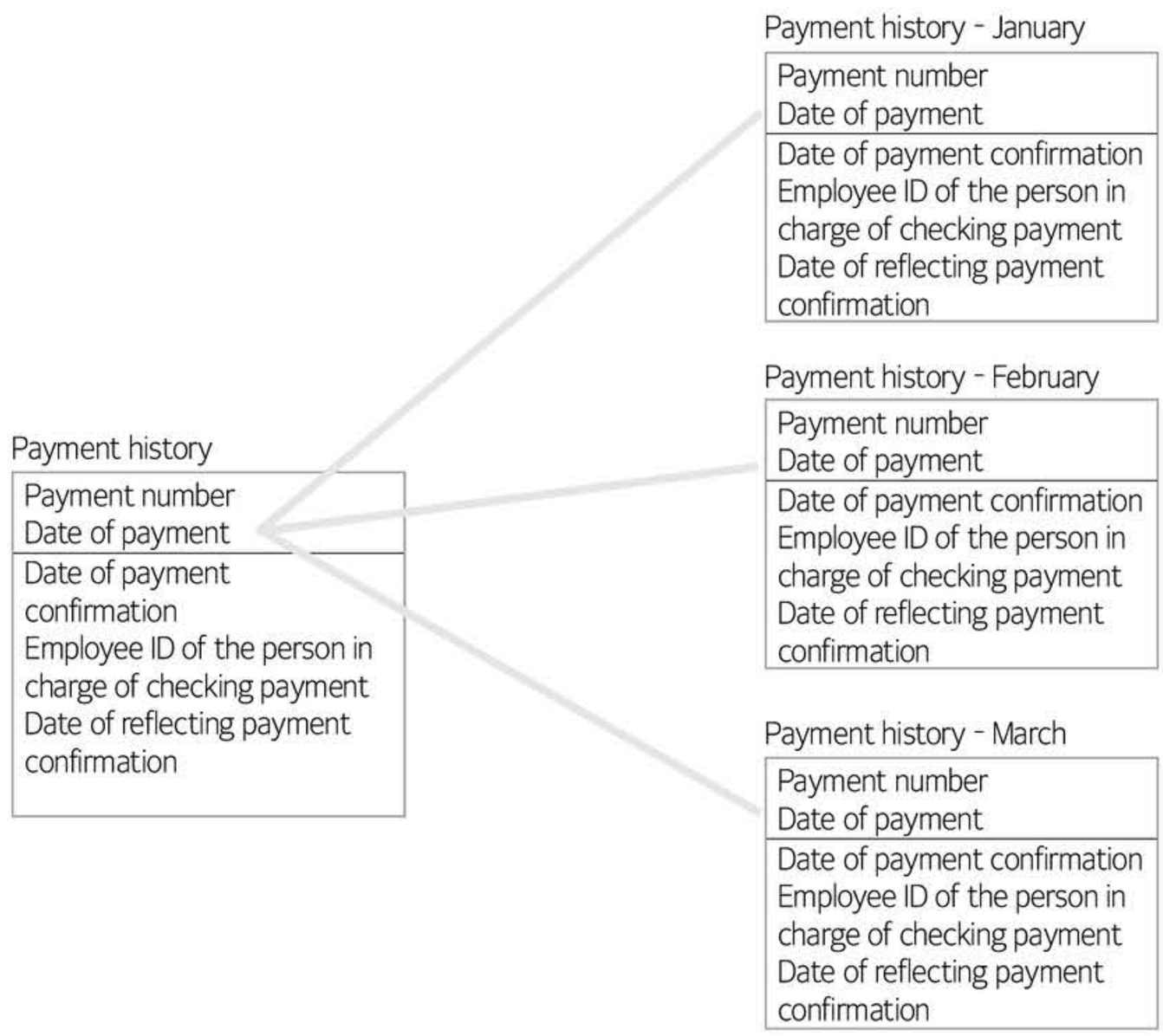
<Table 32> Denormalization procedure and method

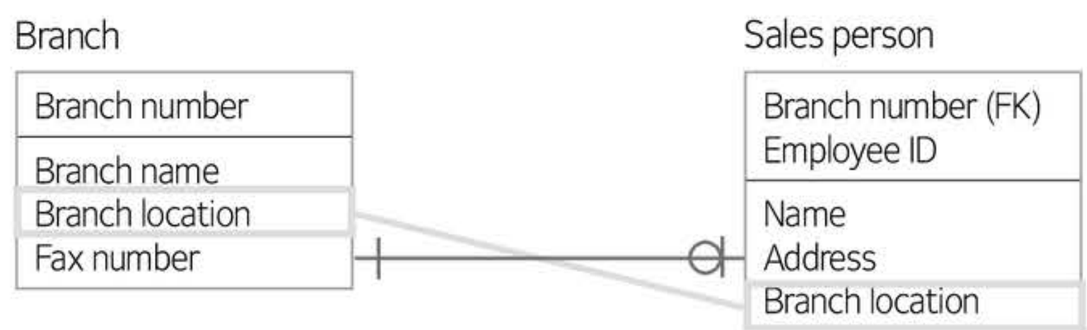
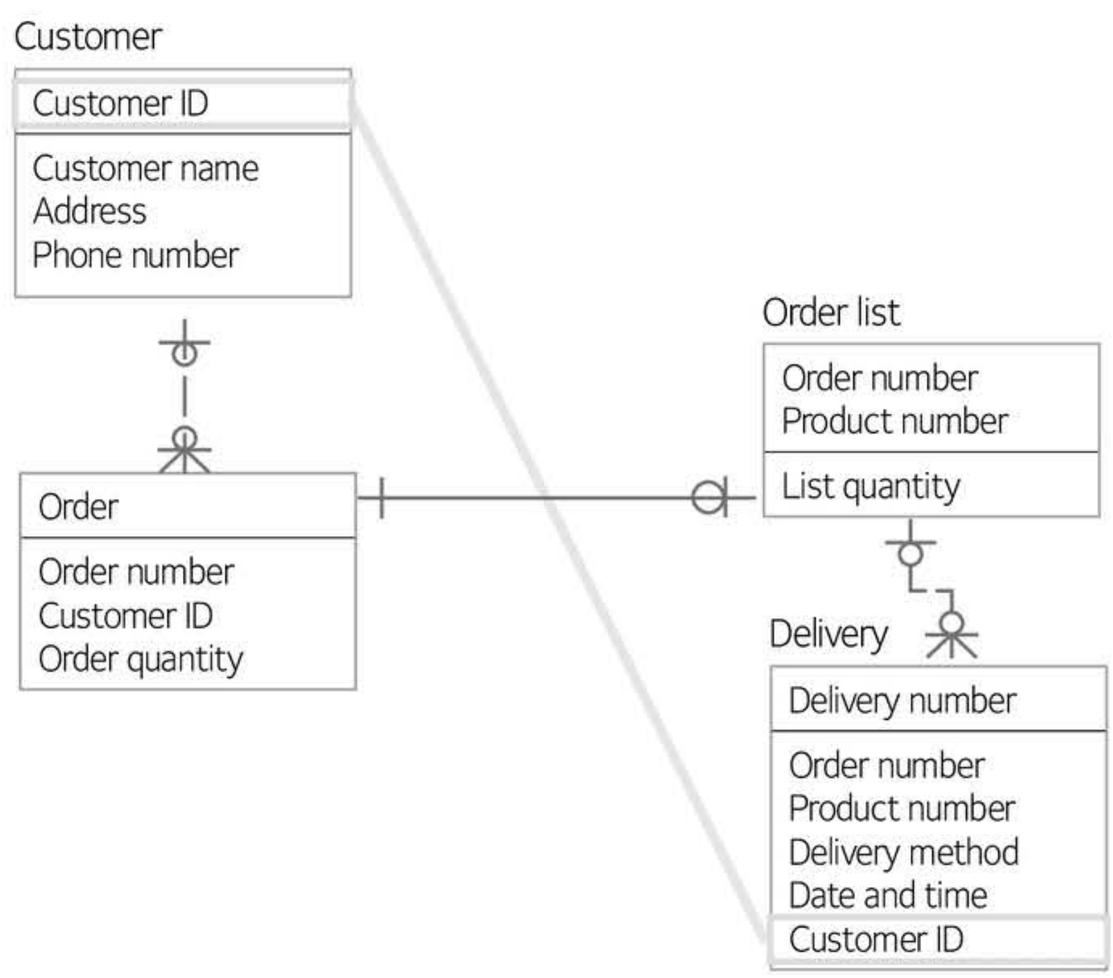
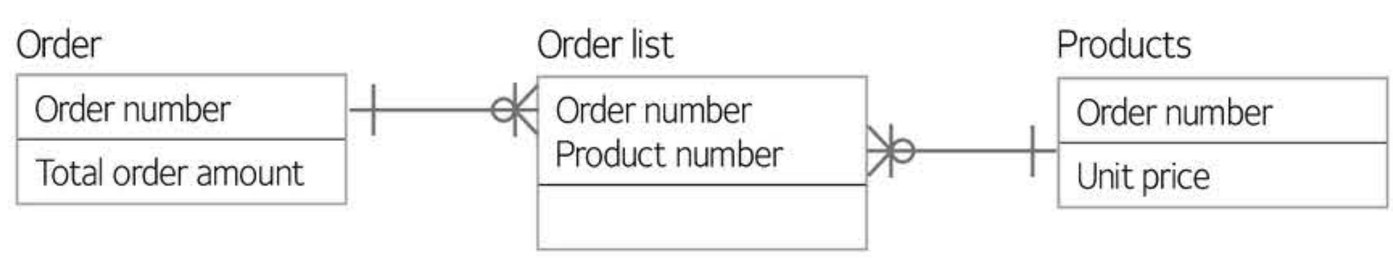
Procedure	Method	Method
1. Investigating denormalization targets	Investigating the range processing frequency	When there are many processes that access frequently used tables, and only a certain range is always searched.
	Investigating wide range processing	When there are large amounts of data in a table and a wide data range is frequently accessed, and performance cannot be guaranteed unless the processing range is reduced to a certain extent.
	Investigating the statistical process	A separate statistics table (denormalization) is created when statistical information is required by a statistics process.
	Investigating the number of table joins	Consider denormalization if the table query job is technically difficult because too many joins are applied to the tables.
2. Considering inducing other methods	View table	A view is used when the data query job is technically difficult because too many joins are applied to the tables.
	Applying clustering	A method of changing the method of storing large amounts of data based on a specific clustering fact (applicable to query-oriented tables only).
	Applying the index	If sufficient performance can be ensured using the index, avoid denormalization by adjusting the index.
	Application	Improving performance by changing the method of applying logic to the application.
3. Applying denormalization	- Table denormalization - Attribute denormalization - Relationship denormalization	

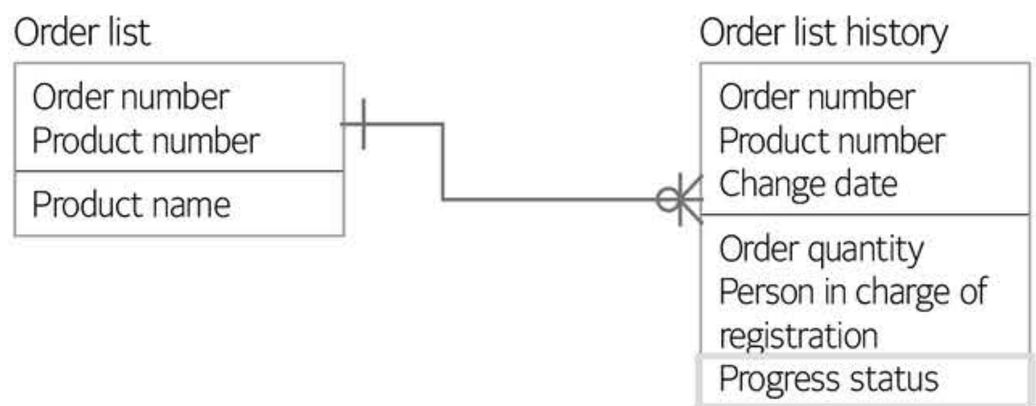
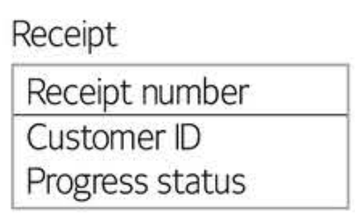
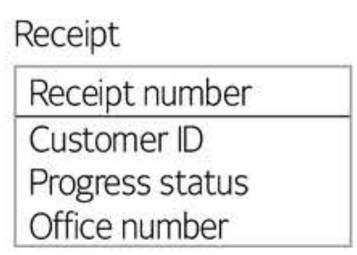
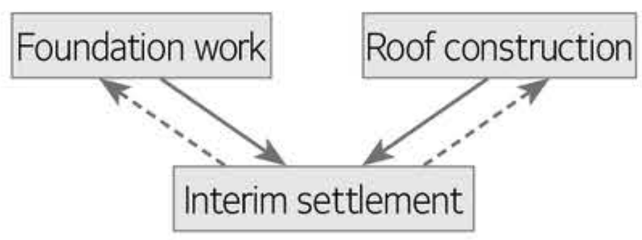
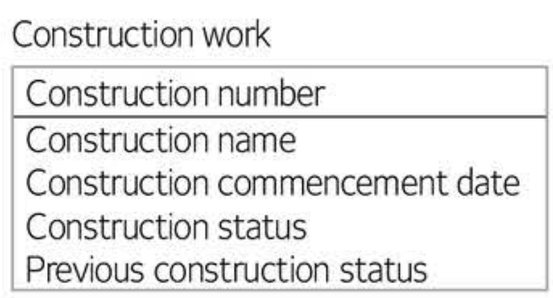
B) Actual denormalization

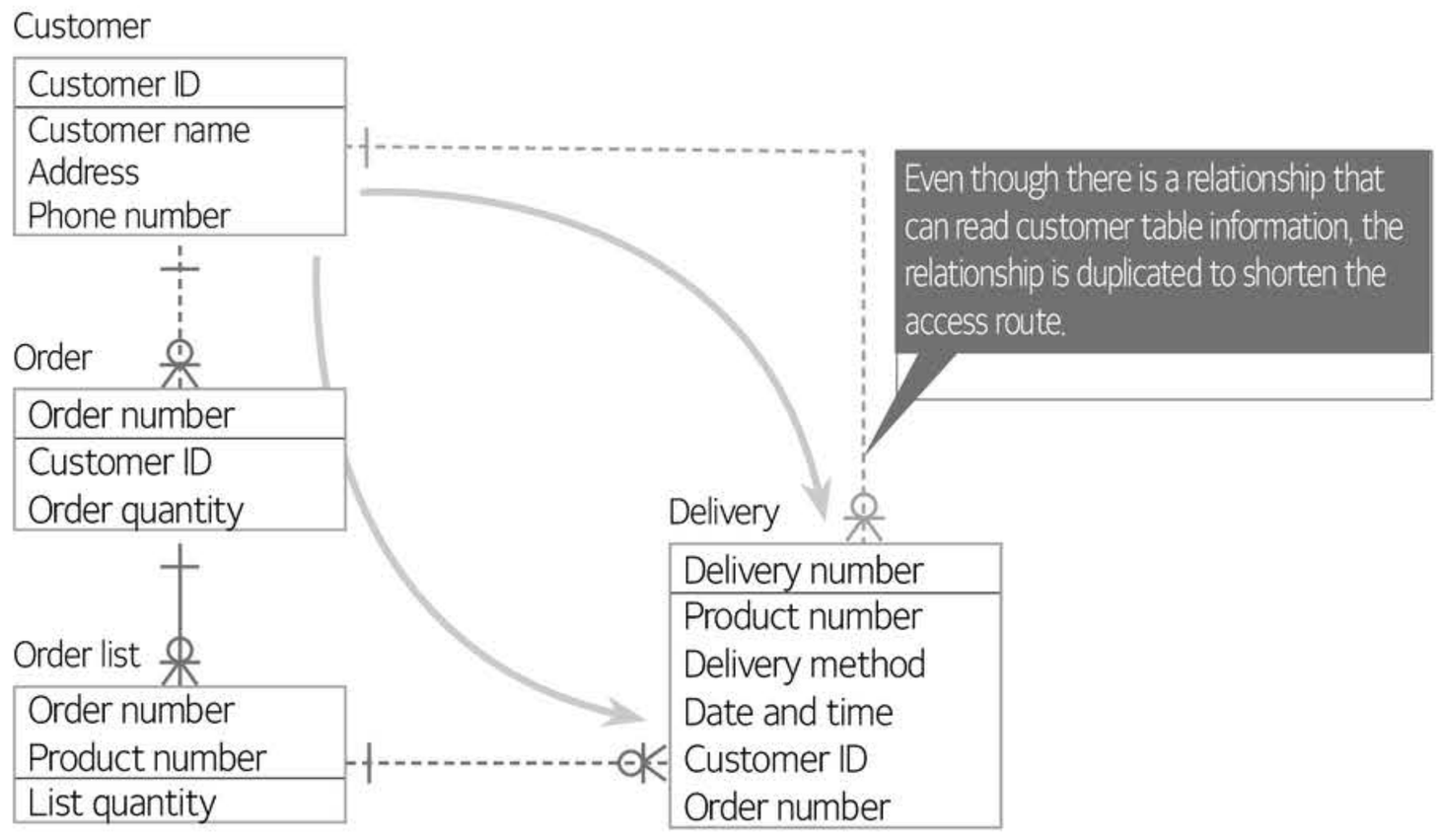
<Table 33> Denormalization technique

Type	Denormalization technique	Description
Table denormali-zation	Merging tables	• Table is merged when it is more efficient in improving performance because there are many join operations. • Merging 1:1 relationship tables • Merging 1:M relationship tables • Merging super/sub-type relationship tables
	Splitting the table	• Table is split when accessing specific attributes only in the table. • When split, access frequency, lockup, and contention are reduced, but performance is degraded because the union operation should be performed when searching the entire split table. • Vertical split: When only specific attributes are frequently accessed, a table is created by splitting the columns.

Type	Denormalization technique	Description
Table denormalization	Splitting the table	 • Horizontal split: When the schema is the same, but the method of using the data value is different for each row (querying history by year, etc.) 

Type	Denormalization technique	Description
Table denormalization	Adding a table	Adding a duplicate table: Performance is improved by removing remote joins, and by duplicating the same table structure, if the task or server is different. Adding a statistics table: Performance is improved by automatically performing SUM, AVG, etc. in advance. Adding a history table: The record in the master table is placed in the history table in duplicate. Adding a partial table: A partial table is a separate denormalized table that collects frequently used and concentrated columns frim among all the columns of one table, in order to reduce disk I/O.
Column denormalization	Adding a duplicate column	A duplicate column is added to prevent performance deterioration due to join operations, that is, to reduce join operations. Duplicating the frequently used column.  • Shortening the query path. 
	Adding a derived column	Values in are calculated advance and stored in a column to prevent performance degradation caused by calculation when a transaction is processed. 

Type	Denormalization technique	Description
Column denormalization	Adding a history table column	A functional column (recent value status, start and end date, etc.) is added to the history table to prevent performance degradation that may occur when querying an unspecified date or a recent value when processing large amounts of data. 
	Adding a column by PK	Occurs when a primary key with multiple meanings is configured with a single attribute. Performance may deteriorate if a specific value is separately queried in a single primary key. "Adding a column by PK" is the method of including data as general attributes for improving performance even though data already exist in the primary key.  [Figure 7-34] Composite PK  [Figure 7-34] Composite PK - Adding a column
	Adding a column to prevent application malfunction	A technique of temporarily storing previous data in duplicate, if the user wants to restore the original value after having processed data incorrectly, even though it is meaningless for the work.  [Figure 7-37] Adding a column to process incorrect input  [Figure 7-38] Adding a column to process incorrect input

Type	Denormalization technique	Description
Relation-ship denor-malization	Adding a duplicate relationship	Data can be processed by passing via several paths. However, an additional relationship is established to prevent performance deterioration. 

07 Considerations: for Performance Design

① Performance improvement goals

- Throughput: The workload performed per unit time.
- Throughput time: Time required to perform a unit work.
- Response time: Time required from the moment when the user presses a key for input until the time when the system responds.
- Load time: Time required to load data into a database.

② Integrating and splitting an entity: Integrating similar entities into one.

<Table 34> Advantages and disadvantages of entity integration

Key benefits of entity integration	Key shortcomings of entity integration
It is easy to query comprehensive information. Improves performance by removing unnecessary joins. Removes duplicates between entities caused by similar attribute integration. The ERD is simplified. The number of physical tables is reduced.	The scalability of the data model is reduced due to changes to the work. It is difficult to understand the workflow using the data model. Performance may deteriorate due to the concentration of large amounts of data.

③ Adjusting the primary key

- The business key is replaced with a system key if it is complex.
- Business key: A key has business meaning such as the resident registration number, student ID, etc., which is helpful in understanding the work.
- System key: A key given by the system according to the predefined rules set in the order in which instances are occurred.

④ Changing the data model structure

- Performance improvement through normalization
- Performance improvement through denormalization
- Performance improvement through data model simplification

⑤ Improving performance related to the index

- The designer efficiently specifies the order of attributes constituting the primary key.
- The designer generates indexes for foreign key columns.
- The designer uses function-based indexes.
- Proper selection of the numbering method: The designer uses the numbering table, the maximum value of table and the system object.

VI. Physics Design of the Database

Recent trends and major issues

Physical design consists of physical modeling in which a logical data model is converted into a physical structure (schema), and a database design process in which an optimal object is created by considering the actual DBMS based on the physical model. Many people misunderstand physical modeling as a simple process of defining data type, length, and domain value, etc., but the data storage space, distributed database build, etc. are actually decided. Therefore, physical design is directly related to database operation performance.

Learning objectives

1. To be able to explain the concept and process of relational table conversion.
2. To be able to design a table.
3. To be able to design a data type.
4. To be able to design an index.
5. To be able to design a database view.
6. To be able to design a distributed database.

Keywords

- B*Tree, index, distributed database, relational table

✚ Preview for practical business

Physical data modeling and database design are performed in the database physics design phase. Physical data modeling consists in converting the logical data model into a database storage structure (physical data model) by considering the characteristics of each DBMS to be used. Here, the concept of physical data modeling and database design can be summarized as follows: Physical data modeling consists in designing matters related to the structure of data including the physical appearance, whereas database design consists in optimally designing the physical model (design drawing) as objects from the DBMS perspective. Examples of database design include the plan for using the storage space for each object efficiently, object partitioning design, and optimal index design, etc. In terms of practical business, physical design consists in understanding the denormalization and performance design for physical data modeling, and in understanding and applying relational table conversion, table, data type, index, database view, and distributed databases for database design.

01 Relational Table Conversion and Table design

Like logical modeling, the academic world and the industrial world have a quite different understanding of the scope of physical modeling. In particular, the process of creating a table structure diagram corresponding to the ERD, that is, the process of relational table conversion, is quite different. The academic world understands this phase as logical modeling, whereas industry understands this process as physical modeling. It is recommended to understand this difference and check the things to be done in each phase according to the two perspectives.

A) Differences between the academic world's and industry's understanding of physical modeling

① Understanding of the academic world

- Physical modeling refers to the process of implementing a table structure diagram, which is an output of logical modeling, according to an actual DBMS.
- Major tasks of physical modeling: Data type definition, index design, etc.
- Normalization and denormalization, which handle table structures, are performed in the logical modeling phase. However, denormalization is performed in logical modeling due to the requirements of physical modeling, because denormalization is performed in consideration of performance.

② Understanding of industry

- Physical modeling refers to the process of converting the ERD, which is an output of logical modeling, into a table structure drawing.
- The major tasks of physical modeling are more diverse than those of the academic world, including table and column definition, primary key and foreign key definition, normalization and denormalization, data type definition, index design, view design, and distributed design.

B) Relational table conversion

① Entity type conversion

- the relation R is created that corresponds to each entity type E and all general attributes of E are included as the columns of R.
- Each sub-component is included only for the composite attribute.
- After selecting one of the key attributes of E, it is selected as the primary key of R.

② Conversion of the weak entity

- The relation R is created that corresponds to each weak entity type W, and including all general attributes of W as the columns of R.
- The primary key of E is included as the foreign key of R, regarding the identifying entity type E of W.
- The primary key of R is composed with a combination of the primary key of E and the partial key of W.

③ Conversion of the 1:1 relationship

- It is assumed that S and T are two entity types participating in the relationship.
- The primary key of T is included as the foreign key of S.

- All general attributes belonging to the relationship is included in S.

④ Conversion of the 1:N relationship

- It is assumed that S (N-side) and T (1-side) are two entity types participating in the relationship.
- The primary key of T is included as the foreign key of S.
- All general attributes belonging to the relationship is included in S.

⑤ Conversion of the M:N relationship

- It is assumed that S and T are two entity types participating in the relationship.
- A new relation R is created that corresponds to the relationship.
- All general attributes belonging to the relationship in R.
- The primary key of S and T is included as the foreign key of R.
- The primary key of R is composed of a combination of a foreign key from S and a foreign key from T.

⑥ Conversion of multi-valued attributes

- The relation R for the multi-valued attribute MA is created that belongs to the entity type E.
- MA is created as the column of R.
- K, the primary key of E, is created as the foreign key of R.
- The primary key of R is composed of a combination of the above columns.

⑦ Conversion of N-ary relationship (N> 2)

- A new relation R is created for each N-ary relationship.
- All general attributes belonging to the relationship is included as the column of R.
- The primary keys of all entities participating in the relationship are included as the foreign keys of R.
- The primary key of S is composed with a combination of all foreign keys. (However, the foreign key of which cardinality is 1 will be excluded from a combination of the primary key.)

⑧ Conversion of the generalization relationship

- Tables are created that correspond to the upper entity type and the lower entity type respectively.
- The primary key of the table is included that corresponds to the upper entity type in the table of the lower entity type.

C) Table design

① Table types

- Heap-organized table: This table type is used as a standard table in most commercial DBMSs. The record storage location in a table is determined when the record is inserted, instead of the value of a specific attribute.
- Clustered index table: A table in which data is stored in the order of the primary key value or index key value.
- Partitioned table: A table designed to prevent performance degradation and make management easy, by storing a physical table that has been split based on specific criteria (range, value, hash, etc.), even though large-scale data is logically one table.
- External table: A database object that enables the user to use external files as a general table existing in the database

- Temporary table: A temporary table that can store and process data for each transaction or session.

② Considerations when designing a table

- If the sum of the column data length is more than 1 block size, vertical split needs to be considered.
- If the use frequency of a specific column is extraordinarily high, vertical split needs to be considered.
- If different user groups use a specific column only, vertical split needs to be considered.
- When considering vertical partitioning, there should be no case in which a split table is processed simultaneously by one transaction or join operations occur frequently.

02 Data Type Design

Data are stored in a database in various forms such as characters, numbers, dates, and images. If a data type is incorrectly selected when designing a database, it makes the development of an application more difficult or deteriorates performance. Therefore, it is very important to carefully determine the data type or size in the database design phase.

A) Character data type

① Fixed-length character type: A type of data for which as much space is allocated and used as data are declared.

② Variable-length character type: A type of data for which as much space is allocated and used as data are actually stored.

③ Character large object: A type of data that is designed to be stored as large character type data. As it is too big, the actual type column is not stored in a database, but stored outside the table, and the table column stores the address that references the data location only. This type of data can be useful when storing large amounts of text, such as a book or publication.

B) Numeric data type

① Real type: A type of data that represents real numbers in floating point and variable point formats.

② Integer type: A type of data that represents an integer within the range of the maximum size.

C) Binary data type

① Fixed-length binary data type: A type of data that can store as much binary data as has been defined.

② Variable-length binary data type: A type of data that can be stored in the form of binary data by allocating as much space as the actual storage size within the defined size.

③ Binary large object: A type of data that can be stored in the form of one large binary set of data such as image, video, and sound. Like CLOB, it is stored outside the table, and a reference address is stored in the actual CLOB column.

D) Date data type

Date and time data type: A type of data that can be stored in the form of time, date, and time + date.

03 Index Design

A) Functions of the index

① An index is a data structure that organizes database record information to perform search operations quickly.

② Desired information can be searched quickly without searching all data if an index is used. The search speed does not change even if the number of records increases.

③ The index is sorted by the column value that has generated the index and has the pointer of the location where the actual values are stored in the table.

④ The most important function of the index is to speed up data search by reducing the access path.

B) Index design procedure

① An index should be selected based on a comprehensive judgment after collecting all access paths to the table and analyzing the collected results.

② In general, an index is designed in the following order.

- Collection of access paths.
- Selection of candidate columns based on the distribution survey.
- Decision on the access path.
- Decision on the column combination and order.

C) Types of index structures

① Tree-based index

② Function-based index

③ Bitmap join index

④ Domain index

04 View Design

A) Characteristics of the database view

① A view is a table that is virtually created by collecting the desired data only from one or more tables.

② Repetitive data manipulation can be performed efficiently by defining frequently used queries in advance.

③ Users can focus exclusively on the data they are interested in.

④ A view can show calculated or derived information.

⑤ A view can limit the data that users can see.

B) Creation of the view

① A view is created from an existing table using the CREATE VIEW statement ([Figure 37]).

```
CREATE VIEW View_name
[(column_name [,column_name…])]
[WITH ENCRYPTION]
AS select_statement [WITH CHECK OPTION]
```

[Figure 37] View creation syntax

② WITH CHECK OPTION

- This option is used to make all data modification statements executed on the view comply with the conditions of the SELECT statement.

③ WITH ENCRYPTION

- This option prevents users from viewing the text that creates a view.
- To decrypt, the view must be deleted, and a new view should be created.

④ Constraints

- The user of the view should have the SELECT privilege for the entity used in the view definition.
- The SELECT INTO statement cannot be used.
- A view on a temporary table cannot be created.
- A trigger or an index cannot be created for the view.
- Up to 250 columns can be referred to.

C) Modifying data with a view

① A view does not have a separate data copy.

② An update of the view always affects the original table.

③ Constraints

- An update of the view cannot affect more than one original table, and it can affect only one table.
- A column that has a calculated value, built-in function, or calculation function cannot be updated using a view.
- An error will occur if a view affects a table that has a NOT NULL column.
- A default value should have been defined or Null should be allowed for the column without an input value.

D) Other considerations

① If a new column is added to the original table after defining the view, this column will not appear in the view.

② To change the definition of a view itself, the existing view should be deleted, and the view should be defined again.

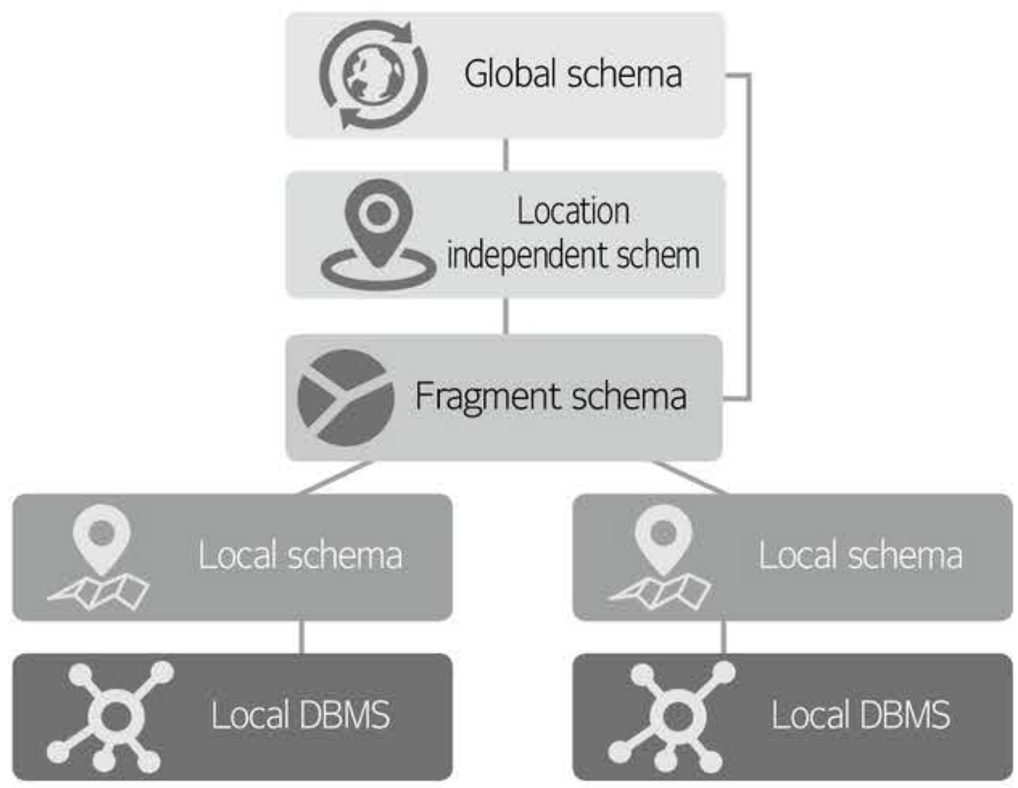
③ The view is not automatically deleted even if the original entity is deleted. Therefore, the view should be manually deleted.

05 Distributed Database

A) Characteristics of the distributed database

① A distributed database refers to a database that is logically integrated and shared so that users can recognize it as a single database even though it is physically distributed among multiple computers on a network.

② The distributed database management system refers to a system that exchanges and manages the information of each local database so that multiple databases can be recognized as one logical database. ([Figure 38])



[Figure 38] Structure of the distributed database

③ Strengths of the distributed database

- Reduces dependence on remote data by controlling its data locally.
- Large volumes of data can be processed, which is impossible in a single server.
- Scale-up is supported by adding a server to the existing system.
- Reliability and availability are improved because even if one site fails, only the data in that site cannot be used.

④ Shortcomings of the distributed database

- Distributed processing increases complexity, which in turn increases software development costs.
- Weak control functions.
- Increased possibility of error due to distributed processing.
- The response speed may vary depending on the condition of the system in which data is physically stored.
- It is difficult to completely ensure data integrity.

B) Data transparency

① To recognize multiple physical databases as a single logical database, users do not need to know where the data are physically located or how to access it.

② This characteristic is called “data transparency”, and the distributed database management system provides partitioning, location, replication, failure, and concurrency transparency.

③ Partitioning transparency

- Partitioning transparency informs users how the global schema has been split.
- Users do not have to know how the global schema has been split because the global query entered by the user is converted into several fragment queries.
- Vertical partitioning splits a relation into relations made up of a subset of attributes.
- Horizontal partitioning splits a relation into relations made up of a subset of tuples.

④ Location transparency

- Location transparency means that users or applications do not need to know the physical location of data in a distributed database.
- That is, users should be able to access data using the same command regardless of the location of data or the input system.

⑤ Replication transparency

- Replication transparency means that users should not need to know which data are redundant or where the redundant data are stored.
- That is, users should be able to think that they are using logically unique data.

⑥ Failure transparency

- Failure transparency means that data integrity should be guaranteed even if a failure occurs in each distributed computer system or network.
- The distributed database should maintain the atomicity of transactions regardless of a component failure.

⑦ Concurrency transparency

- Concurrency transparency means that the consistency of the results is maintained even though multiple transactions are executed at the same time.
- To maintain the consistency of distributed transactions, the distributed database management system mainly uses locking and timestamp methods.

VII. Database Quality and Standardization

Recent trends and major issues

Alan Greenspan, the former chair of the Federal Reserve (FRB), said that "Bad data is invalidating the IT systems of Wall Street. One of the causes of the financial crisis is inadequate data". As the scope of the information system operated by institutions and corporations expands and its complexity increases, data quality - including accuracy, consistency, and timeliness - is becoming a very important issue. That is, data quality control has become an essential requirement for the operation and management of information systems, and each institution is making efforts to present a management framework for data quality and perform data quality control based on that framework. Therefore, it is very important to accurately understand a data quality control framework that can be universally applied to multiple organizations and extended according to the characteristics of a given organization.

Learning objectives

1. To be able to explain the concept of data quality.
2. To be able to explain the concept of data standardization.
3. To be able to design a standardized database using a dictionary.

Keywords

- Data quality: Data value, data structure, data process quality
- Data standardization: Word, term, code, and domain standardization
- Dictionary: Word dictionary, glossary, domain dictionary

✚ Preview for practical business

Problems with practical business that occur when data quality deteriorates

Deterioration in data quality brings about confusion and distrust, as follows.

- The result value becomes different depending on the requesting department (supervisory authority, etc.).
- The information provided by IT cannot be fully trusted.
- It takes time and money to improve or adjust inaccurate data.
- Data reliability cannot be guaranteed even with an additional budget.
- Decisions cannot be made quickly due to inaccurate data.
- It is difficult to understand the exact number for a specific situation.

Effects of data quality control

If data quality is managed well, it can positively affect an entire business including its information system.

- Sales increase, productivity improves, and the market share increases.
- Customer satisfaction increases, service quality improves, and product competitiveness increases.
- Business costs decrease, IT costs are reduced, and rework costs are reduced.
- Accident prevention, compliance with government regulations, maintenance of market confidence.

What if database quality control and standardization are not understood?

The importance of quality control and standardization, which constitute the basis of information systems, is being increasingly emphasized as information systems become larger and more complex. If database quality control is not clearly understood, the information provided by IT cannot be fully trusted because the result value will differ depending on the requesting department. In addition, decisions cannot be made quickly because it takes a lot of time and money to improve and adjust inaccurate data. On the other hand, an inappropriate understanding of data standardization can cause data duplication and data inconsistency for each organization, work, and system. In addition, information cannot be provided in time as it takes more time to understand the meaning of data. Also, data integration, information system change, and maintenance become very difficult if data are not properly standardized. Therefore, it is very important to understand the concept and process of database quality control and standardization.

01 Data Quality Control Framework

"Data quality control" refers to a series of activities undertaken to improve the quality of data. Data quality can be controlled over the data value, data structure, and data management process.

<Table 35> Data quality control framework

Organization \ Target	Data value	Data structure	Data management process
CIO / EDA (overview perspective)	Data management policy		
DA (conceptual perspective)	Standard data	Conceptual data model Data reference model	Data standards management Requirements management
Modeler (logical perspective)	Model data	Logical data model	Data model management Data flow management
DBA (physical perspective)	Management data	Physical data model Database	Database management Database security management
User (operational perspective)	Business data	User view	Data utilization management

A) Data value

① Standard data: Data related to standards, which define and use the terms, domains, codes, and other data-related elements used in the information system based on the common format and content.

- Standard word dictionary: A dictionary that defines the basic unit of the world that is used for business by enterprises or institutions and which has a certain meaning.
- Standard domain dictionary: A range of values that satisfy the condition defined in the attribute.
- Standard term dictionary: A term that is frequently used for work and is defined by combining standard words.
- Standard code: Data are replaced with a symbol within a limited range according to the defined criteria, in order to standardize the data value that can be expressed in various ways.
- Data standard element: A standard for the data-related elements that are needed to design and build a system.

② Model data: Data required to operate and manage the data model.

- Model data include metadata and DBMS object information, etc., regarding the data reference model, conceptual data model, and logical data model.
- Completeness, consistency, traceability, interrelation, up-to-dateness, and compatibility are managed.

③ Management data: Data required to operate and manage a database effectively.

- Usage management data, failure and security management data, performance management data, flow management data, quality control data, etc. are included.

④ Business data: Data needed to conduct the work or business of institutions or companies.

- Business data are classified into source data, operational data, and analysis data according to the data flow.

B) Data structure

① Conceptual data model: A model that defines the subject area, key entity, and key relationship of the data that meet business requirements.

② Data reference model: A model that defines a set of standard data and control items by business area and subject area for the purpose of establishing and maintaining the data architecture, of interoperating the data models used by organizations, and of referring to and reusing other organization data models.

③ Logical data model: A model that defines the logical data set, control item, and relationship by specifying the conceptual data model.

④ Physical data model: A model that actualizes the logical data model by considering the characteristics and performance of the DBMS.

⑤ Database: A result of implementing a physical model and a data repository that stores actual build data.

⑥ User view: A screen or output of the information system that provides data.

C) Data management process

① Data standard management: An activity that cleans and improves data by defining, changing, and managing the standard word dictionary, domain dictionary, standard term dictionary, standard code, data-related element standard, etc., which are defined according to the data standardization principle, into the form that is most suitable for the institution, and by checking compliance with data standards.

② Requirements management: The task of collecting, classifying, and reflecting user requirements in the related applications and the overall system as well as data.

③ Data model management: The task of reflecting the data structure, which is changed by data requirements management, in a model; and the task of maintaining a data model in the same way as the database system structure.

④ Data flow management: The task of managing the process of manually creating source data, storing the data or processing the data through extraction, conversion, and loading.

⑤ Database management: The task of stably operating and managing a database.

- This includes such tasks as backup, security, tuning, and monitoring, etc.

⑥ Database security management: The task of protecting a database and loaded data from a series of unauthorized actions.

- This includes such tasks as authentication, access control, log analysis, and auditing.

⑦ Data utilization management: The task of measuring quality by selecting the data to be measured and the quality index and by analyzing the cause of a failure to comply with the quality requirements.

D) Data quality management maturity model

To access data quality control systematically, the level of data quality control should be measured based on objective

criteria, and a guideline on what to prepare to secure a better level is urgently required. The data quality management maturity model, which was developed to satisfy these requirements, is composed of three key elements, namely, the data quality standard, the data quality management process, and the data quality management maturity level.

① Data quality standard

- Data validation aspect: Accuracy, consistency
- Data utilization aspect: Usability, accessibility, timeliness, security

② Data quality control process

- Processes that are needed to improve the quality of accuracy, consistency, usability, accessibility, timeliness, and security are identified.
- Sometimes, there is a one-to-one correspondence between the quality standard and the data quality management process. There are also processes that are related to several quality standards at the same time.

③ Data quality control maturity level

- The level is defined from 1 to 5.
- The higher the maturity level, the more the data is managed systematically and elaborately.

02 Data Standardization

Data standardization means that principles for the name, definition, format, and rule of data information elements are established and applied throughout an organization. It is indispensable for improving the clarity of communication and data quality and for reducing the cost of data conversion and cleansing.

A) Overview of data standardization

① Recently, interest in data integration and data quality has been increasing because data have emerged as a key element of a corporation's strategic decision-making process.

② Data standardization means that the principles for the name, definition, format, and rule of data information elements, which are dispersed in each system, are established and applied throughout an organization.

③ The exact meaning of data can be understood, and opposite opinions about data can be adjusted through data standardization.

B) Necessity of data standardization

① Improves clarity of communication by unifying names.

② Reduces the time and effort spent on finding the location of the necessary data.

③ Improves data quality by applying consistent data formats and rules.

④ Reduces the cost of data conversion and cleansing when designing a data interface between information systems.

C) Components of data standardization

① Standardization components: Data standard, data management organization, data standardization procedure

② Target for data standard management: Terms, words, domains, and codes

③ Management functions

- Data standard management function: Word management, term management, domain management, standard code management, multi-standard management
- Data structure management function: ER model structure management, database schema management, variable attribute management, history management, model comparison management
- Process management function: Standard registration, model registration

D) Defining data standards

① Overview of deriving standard words

- Standard words are derived from all terms that are dispersed in each information system by dividing them into word units.
- Standard words are extracted from each of the entities and attributes that are managed throughout the organization.
- The extracted words are managed along with physical names and similar terms based on logical names after reorganizing homophones and synonyms.
- The standard word dictionary should also define the structure between such words as synonyms, antonyms, etc. in addition to individual words.

② Word split

- Splitting the collected terms into the basic words that are used for business and have a certain meaning.
  E.g. Resident registration number → resident + registration + number

③ Processing synonyms

- Selecting one representative word for words with the same meaning as a standard word.
- All words that are finally derived should be unique in both their Korean and English abbreviations.
  E.g. Unification of the secret number, passcode, password into the password.

④ Deriving a standard domain dictionary

- Each attribute should be assigned to a certain domain, and cannot be assigned to more than one domain.
- When a new attribute is added, the domain of that attribute should be selected and registered.
  E.g. (Term) Resident registration number - (Domain name) Registration number - (Type) Char(13)

⑤ Deriving a standard code dictionary

- Standard codes are defined by checking and integrating the same codes after collecting all of the codes used throughout the organization.
- Derived codes can be defined from the standard codes (e.g. Color code- wrapping paper color).

⑥ Deriving a standard term dictionary

- Standard terms are created by combining the standard words that are extracted from the entities and attributes held throughout the organization.
- Standard terms for current terms are derived by referring to the standard word dictionary, standard domain dictionary, and standard code dictionary.
- The standard term dictionary is divided into an entity term dictionary and a property term dictionary.
  E.g. (Standard word) Customer, classification, code, ID → (Standard term) Customer classification code, customer ID

E) Confirming data standards

① Target for data standard review

- Standard word dictionary, standard domain dictionary, standard code dictionary, standard term dictionary, etc.

② Major verification criteria for data standards

- Uniqueness: Check whether each data standard is physically and semantically unique.
- Completeness: Check whether all required inputs for each data standard object are defined.
- Accuracy: Check whether the input items of each data standard target are completely entered.
- Generality: Check whether the defined data standard can be applicable to multiple systems.

VIII. Relational Operation (Relational Algebra)

Recent trends and major issues

Relational algebra is a set of operations for processing relations in a relational database and has the characteristic that all operands of each operation are relations, and the result of the operation is also relations. Basic relational algebra operations are usually divided into set operations and relation operations. Exact understanding of relational algebra operations is required to use the relational database efficiently using a language such as SQL.

Learning objectives

1. To be able to explain the concept of relational algebra.
2. To be able to utilize set operations and relation operations.
3. To be able to utilize extended relational algebra operations.

Keywords

- Relational operation, relational algebra, relational calculus, set operation, relational operation, join, extended relational algebra

Preview for practical business

Most relational databases used by the information systems of enterprises and organizations use SQL as the standard query language. If an SQL statement is written without understanding relational algebra, which is the theoretical basis of SQL, the SQL statement cannot retrieve the result intended by the user, and may degrade performance. Therefore, users should fully understand relational algebra before writing a more optimized SQL statement.

01 Understanding the Concept of Relational Algebra

Relational algebra is largely divided into relational algebra and relational calculus. Unlike relational analysis, which is a non-procedural language that defines desired information, a relation operation has the characteristic of a procedural language because it specifies the order of operations to be performed in order to obtain desired information. In particular, as relational algebra becomes the theoretical basis for SQL, the standard query language of relational databases, the concept and principle of relational algebra need to be clearly understood. Therefore, this module introduces set operation and relation operation, which constitute the basic relational algebra, and the extended relational algebra operation.

02 Set Operation and Relation Operation

Relational algebra is a set of operations for processing relations in a relational database and all operands of each operation, and the result of the operation is relations. Basic relational algebra operations are usually divided into set operations and relation operations.

A) Set operation

Set operators include Union, Intersect, Difference, and Cartesian Product.

<Table 36> Set operators

① Operator	② Sign	③ Description
Union	∪	• A binary operation to create a relation that is composed of tuples, which belong to more than one relation among two relations that can be merged.
Intersect	∩	• A binary operation to create a relation that is composed of tuples, which belong to both of the two relations that can be merged.
Difference	-	• A binary operation to create a relation that is composed of tuples, which exist in one relation but not in the other when there are two relations that can be merged.
Cartesian product	×	• A binary operation to create a relation that is composed of a combination of tuples, which can be configured with two relations.

B) Relation operation

Relation operators include Select, Project, Join, and Division.

<Table 37> Relation operators

Operator	Sign	Description
Select	σ	• A unary operation in which a specific tuple which satisfies the condition is separated from a relation. • A horizontal subset is taken from one relation.
Project	π	• A unary operation in which a specific attribute is elected and separated from a relation. • A vertical subset is taken from one relation.

Operator	Sign	Description
Join	⋈	• An operation to create a new relation that is composed of tuples, which is created by combining the tuples of two relations that meet the join condition in two or more relations. • Theta join: A join operation with general join conditions (=, <, ≤, >, ≥, ≠). • Equi-join: A join that uses the equal condition (=) only. The common attribute used in the join appear in duplicate. • Natural join: A join that uses the equal condition (=) only. The common attribute used in the join appear only once without duplicates.
Division	÷	• An operation that separates and projects the tuple of R where all tuples of the relation S exist, when there are two relations R and S.

03 Extended Relational Algebra Operations

Useful operations can be defined by extending the basic relational algebra operations. Extended relational algebra operations such as outer join, semi join, outer union, etc. are generally used.

<Table 38> Extended relational algebra operations

Operator	Sign	Description
Semi join	⋉	• An operation that performs natural join for the result of projecting the relation after the operator with join attributes on to the relation before the operator. • It has the same effect as projecting the result of performing natural join on two relations using the attribute of the relation before the operator.
Outer join	$⋈^+$	• In this operation, if a tuple corresponding to a tuple in one relation does not exist in the other relation, it is not excluded, but the other is changed to a Null tuple to include it in the result. • All tuples of the two join relations are included in the result relation.
Outer union	$\cup^+$	• An operation that creates a union by expanding the sequence number of two relations that cannot be completely merged.

IX. Relational Database Language (SQL)

Recent trends and major issues

Even though the types of databases have been diversified recently, most of the information systems used by corporations and organizations are still running on relational databases. Therefore, software developers should be familiar with relational database language and query processing methods and should be able to use them proficiently in order to process data efficiently.
SQL was developed by IBM researchers Donald D. Chamberlain and F. Raymond Boyce in the early 1970s. SQL began with the name of SEQUEL (Structured English Query Language) and was devised to manipulate and receive the data stored in System R, which was the semi-relational database management system of IBM. Recently, interest in acquiring SQL-related certifications has been increasing because the importance of SQL has been highlighted.

Learning objectives

1. To be able to utilize a Data Definition Language (DDL).
2. To be able to utilize a Data Control Language (DCL).
3. To be able to utilize the basic operation, group function, and advanced join of a Data Manipulation Language (DML).

Keywords

• SQL, Data Definition Language (DDL), Data Control Language (DCL), Data Manipulation Language (DML)

✚ Preview for practical business

Since most of the recent information systems store data in a database, an adequate database processing language is needed to process values by linking a program to a database. In particular, as more than 90% of the databases adopted by a project are mostly relational databases in the majority of cases, developers should be familiar with the relational database language. So how well should developers familiarize themselves with the relational database language? Objects such as tables and views are managed by the data architect (DA) or the database administrator (DBA) in some large-scale projects; however, developers manage everything from table creation/change/delete to data processing in most projects. Therefore, it is desirable for the developer to know all of the data definition language (DDL), data control language (DCL), and data manipulation language (DML) of the relational database, in order to perform those tasks smoothly. That is, relational database languages are one of the key database build and operation areas that developers should know well, and it is recommended for developers to learn those languages to such a level that they can use the syntax quite freely.

01 Types of Relational Database Languages

A) DDL, DCL, DML

The relational database language processes the structure and value stored in a relational database and can be classified into the Data Definition Language (DDL), Data Control Language (DCL), and Data Manipulation Language (DML).

<Table 39> Types of relational database languages

Classification	Description	Main commands
DDL (Data Definition Language)	A language that defines the relationship between data and is used to create, change, and delete a data structure like a table in a database.	CREATE ALTER DROP RENAME
DCL (Data Control Language)	A language that controls access to data in a database and is used to perform data security, integrity, and concurrency control Sometimes, commit and rollback, which controls nested transactions, are separated into TCL (Transaction Control Language).	GRANT REVOKE COMMIT ROLLBACK
DML (Data Manipulation Language)	A language that searches, registers, deletes, updates, etc. the data of the database user or application program.	SELECT FROM WHERE INSERT UPDATE, DELETE

B) Changes in SQL and the characteristics of SQL3

① Changes in SQL

SQL was developed by IBM researchers Donald D. Chamberlain and F. Raymond Boyce in the early 1970s, and its standards have been changed periodically by the ANSI, as follows.

<Table 40> Changes in SQL

Year	Name	Alias	Characteristics
1986	SQL	SQL-86	First SQL standard
1992	SQL2	SQL-92	Adopted a major revision (ISO9075) FIPS 127-2. Relational database
1999	SQL3	SQL-1999	Object-oriented database Added the regular expression, recursive query, trigger, etc.
2003	SQL4	SQL-2003 SQL-2006	XML-related characteristics, Window functions, standardized sequences, and a column that has an automatically generated value
In progress	SQL/NM		Consistent access to the metadata registry

② Characteristics of SQL3

The object-oriented characteristics of SQL3 can be seen as the common characteristics of the object relational DBMS..

<Table 41> Characteristics of SQL3

Characteristics	Contents
Relational characteristics	Newly introduced data types and conditional expressions, and a strengthened type system.
Object-oriented characteristics	Supports the user-defined type, object, object identifier, reference type, function, and method.
Other additional functions	Newly introduced concepts such as the recursive query. Provides an active database function that supports a trigger. Supports the client-server environment. Improved security and view update functions.

02 Data Definition Language (DDL)

A data definition language is used to describe the database schema in such a way that computers can understand it. Specifically, it manages the creation, change, and deletion of a table structure. The main statements of the data definition language include CREATE, DROP, ALTER, and RENAME.

A) Types of data definition language

① CREATE: Defining and creating an object

A table, view, index, schema, function, etc. can be defined using the CREATE statement. A primary key and foreign key can be defined when creating a table, and the object should not be the same as other created objects.

② ALTER: Changing an object

Once a table has been created, it is best to keep its structure intact. However, it may be necessary to add or delete columns and constraints while using the table due to changes in the work requirements or the need for system operation. At this time, the structure of the table can be changed using the ALTER statement.

③ DROP: Deleting an object

This command is used to delete all data and structures of the table when it has been created incorrectly or is no longer needed. However, the referenced constraint related to the table concerned should be deleted first before deleting a table. In Oracle, the CASCADE CONSTRAINT option can be used to delete it at the same time.

④ RENAME: Changing an object name

This command is used to change the table name.

03 Data Control Language (DCL)

A data control language is used to define and describe data controls, in order to use the database in common. The main statements include GRANT, REVOKE, COMMIT, and ROLLBACK. Sometimes, COMMIT and ROLLBACK, which are used to control nested transactions, are separated and called TCL (Transaction Control Language).

A) Roles of the data control language

<Table 42> Roles of the data control language

Item	Roles
Data security	Protect data from unauthorized users.
Integrity	Maintain the accuracy of data.
Recovery	Cope with system failure.
Concurrency control	Enable multiple users to access (use) a database at the same time.

B) Types of data control languages

① GRANT

This statement is used to grant privileges related to an object to a user.

- Example: When granting the privilege of searching the customer table to user ABC.
  -> GRANT SELECT ON customer TO ABC ;

② REVOKE

This statement is used to cancel a user's privileges related to an object.

- Example: When withdrawing the privilege of searching the customer table from user ABC.
  -> REVOKE SELECT ON customer FROM ABC ;

③ DENY

This statement is used to deny the user's privilege related to an object. If both GRANT and DENY are assigned at the same time, DENY takes precedence.

- Example: When denying the user ABC's privilege of searching the customer table.
  -> DENY SELECT ON customer TO ABC;

④ COMMIT

This statement is used to complete a transaction and confirm a data change.

⑤ ROLLBACK

This statement is used to cancel a transaction and restore data to their original state before the change.

04 Data Manipulation Language (DML)

A data manipulation language is used to input, modify, delete, and search data in a database. This section introduces the data manipulation language by subdividing it into basic operations, aggregate operations, and advanced join operations.

A) Basic DML operations

DML basic operations consist of statements such as INSERT, UPDATE, DELETE, SELECT, FROM, and WHERE, and are used for input, modification, deletion, and search operations. Among them, FROM and WHERE are used together with the INSERT, UPDATE, DELETE, and SELECT statements. FROM is used to designate the target table and WHERE is used to set the conditions.

<Table 43> Basic DML operations

Statement	Description and examples of usage
SELECT	Used to search data values stored in a table.
	SELECT column name 1, column name 2 FROM table name WHERE conditional clause;
INSERT	Used to input values in all or some columns of a table.
	(Entering all columns) INSERT INTO table name VALUES (input value 1, input value 2, input value 3); (Entering some columns) INSERT INTO table name (column name 1, column name 3) VALUES (input value 1, input value 3); (Entering from an existing table) INSERT INTO table name SELECT column name 1, column name 2 FROM table name WHERE conditional clause;
UPDATE	Used to change the data stored in a table. UPDATE is performed for all rows if a conditional clause is not specified.
	UPDATE table name SET column name 1 = input value 1, column name 2 = input value 2 WHERE conditional clause;
DELETE	Used to delete the data stored in a table. DELETE is performed for all rows if a conditional clause is not specified.
	DELETE FROM table name WHERE conditional clause;

B) Aggregate DML operations

Aggregate operations can be performed by using the AVE, SUM, and COUNT statements of the DML. The GROUP BY clause is specified after the FROM clause and the WHERE clause in the SQL statement. This clause is used to group data and to obtain statistical information on each group item. Aggregate operations are generally used with the GROUP BY clause. However, if an entire table is a group, an aggregate operation can be performed without the GROUP BY clause. On the other hand, the HAVING clause can be used to set the condition for the GROUP BY clause. In addition, the DISTINCT option can be used to consider the same value as the same data.

<Table 44> Aggregate DML operations

Statement	Description and examples of usage
COUNT	Used to output the number of rows.
	SELECT COUNT(*) FROM table name
SUM	Used to output the total excluding the NULL value.
	SELECT column name 1, SUM (column name 2) AS alias FROM table name GROUP BY column name 1
AVG	Used to output the average excluding the NULL value.
	SELECT column name 1, AVG (column name 2) AS alias FROM table name GROUP BY column name 1

Example of calculation of no. of employees by department/total employee salary/average salary from the [Employee] table below.

Department	Employee number	Name	Agreed annual salary	Job title	Contact information
Finance	01144	Hong Gildong	6000	Team leader	010-XXXX-XXXX
Legal affairs	04489	Lee Mongryong	5400	Assistant manager	010-XXXX-XXXX
HR	01122	Maeng Saseong	5700	General manager	010-XXXX-XXXX
...					
General affairs	07876	Ha Wongil	7700	General manager	010-XXXX-XXXX

SELECT department, COUNT(*) AS number of employees, SUM (agreed annual salary) AS total salary, AVG (agreed annual salary) AS average salary

[Figure 39] Utilization of aggregate operations

C) Advanced DML join operations

① NL (Nested Loop) Join

NL Join performs join operations in a similar way to the nested loop statement used in programming. The table outside the loop is called the "preceding table" or the "outer table", and the table inside the loop is called the "following table" or the "inner table".

NL Join first extracts the rows that meet the conditions of the preceding table and performs a join operation while reading the following table. This operation is repeated as many times as the number of rows that meets the conditions of the preceding table. As the join operation of the following table is repeated as many times as the number of rows that meet the conditions of the preceding table, it is recommended to select a table with fewer result rows as the preceding table in consideration of performance. It is advantageous to NL Join to have a narrow processing range because data are accessed randomly.

② Sort Merge Join

Sort Merge Join performs a join operation by sorting data based on the join column. Sort Merge Join is advantageous when processing a wide range of data as it mainly reads data using the scan method, unlike NL Join. However,

performance deteriorates if all sorting operations cannot be performed in the memory due to there being too many data to be sorted.

③ Hash Join

Hash Join performs a join operation using the hashing technique. The hash function is executed using the join column of the table to perform a join operation. The join operation is performed while checking whether the data with the same hash value actually have the same value. Hash Join was introduced as an alternative for solving the random-access problem of NL Join and sorting the workload of Sort Merge Join.

X. Database Query Application

Recent trends and major issues

SQL is the most important and powerful tool in a relational database. It is not an exaggeration to say that the strengths of the relational database are being increasingly highlighted thanks to the strong function of SQL, which prevents the relational database from being changed to the object-oriented DBMS or another DBMS easily. Prior to the introduction of SQL, the procedural programming language had to be used for simple data processing only by writing a long programming code. Since the introduction of SQL, however, the productivity of database programming has improved significantly, because the DBMS manages everything when the desired data are specified.

Learning objectives

1. To be able to explain the concept and characteristics of the stored procedure.
2 To be able to explain the concept and characteristics of the embedded SQL.
3. To be able to explain the concept and characteristics of the dynamic SQL.
4. To be able to explain the characteristics of each query optimization phase.
5. To be able to explain the characteristics of the optimizers by type.
6 To be able to explain the technique of linking the web and the database.

Keywords

• Stored procedure, embedded SQL, dynamic SQL, query optimization, optimizer, web database linkage

✚ Preview for practical business

SQL is the language that must be used to process data when using a relational database. In the past, SQL statements were simply used to input, modify, delete, or retrieve data. Now, the composition of a statement is becoming too complex to process complex calculations, aggregations, and processing according to the conditions. In fact, SQL statements processing such a complex procedure are written as long as several A4 pages. SQL statements that are not properly written may cause a deterioration of performance, thus creating a problem in the entire project. For example, the performance of the entire database system can be degraded if SQL statements are called repetitively while processing a loop in a program, or if the characteristics of the index are not considered when writing SQL statements, or if SQL statements are excessively long due to the indiscriminate use of subqueries. A full understanding of the database system, data model, index structure, as well as SQL syntax, is required to write correct SQL statements according to the characteristics required by the business.

01 Stored Procedure

A) Definition of the stored procedure

A stored procedure is a set of queries for executing a series of queries like a single function. The stored procedure organizes a series of operations in a database and stores the result in a relational database management system, and is also called a "persistent storage module".

B) Strengths and weaknesses of the stored procedure

Most DBMS vendors provide a function that can describe a procedure using an extended language of the specifications for static and dynamic SQL, including cursor handling and control statements, exception handling, etc. The external procedure function is also implemented by calling a function or a class method from an external module (shared libraries) compiled based on C language and Java class libraries.

C) Strengths of the stored procedure

- Reduces the network load by executing several SQL statements using a single request.
- Reduces the processing time as syntax analysis and internal intermediate code conversion are performed in advance.
- Maintains the referential integrity of data with complex rules by combining it with a trigger.
- Improves the readability of the source code as the host language like Java and SQL statements are clearly distinguished.
- Easy to maintain as the website can be modified during operation by replacing the stored procedures.

D) Shortcomings of the stored procedure

- Reusability is low as a code asset because products are less compatible.
- The definition of the stored procedure needs to be changed together with the external application program when work is changed, which is used as a part of the business logic. That is, additional efforts are needed when a change occurs, and a failure may occur due to mistakes in the change process.

02 Embedded SQL

A) Definition of embedded SQL

When an SQL statement that defines or accesses data is included in an application program code such as PASCAL, COBOL, C, C++, Visual Basic, Java, etc. and executed together at the time of program execution, the inserted SQL statement is called an "embedded SQL", and the application program is called a "host program".

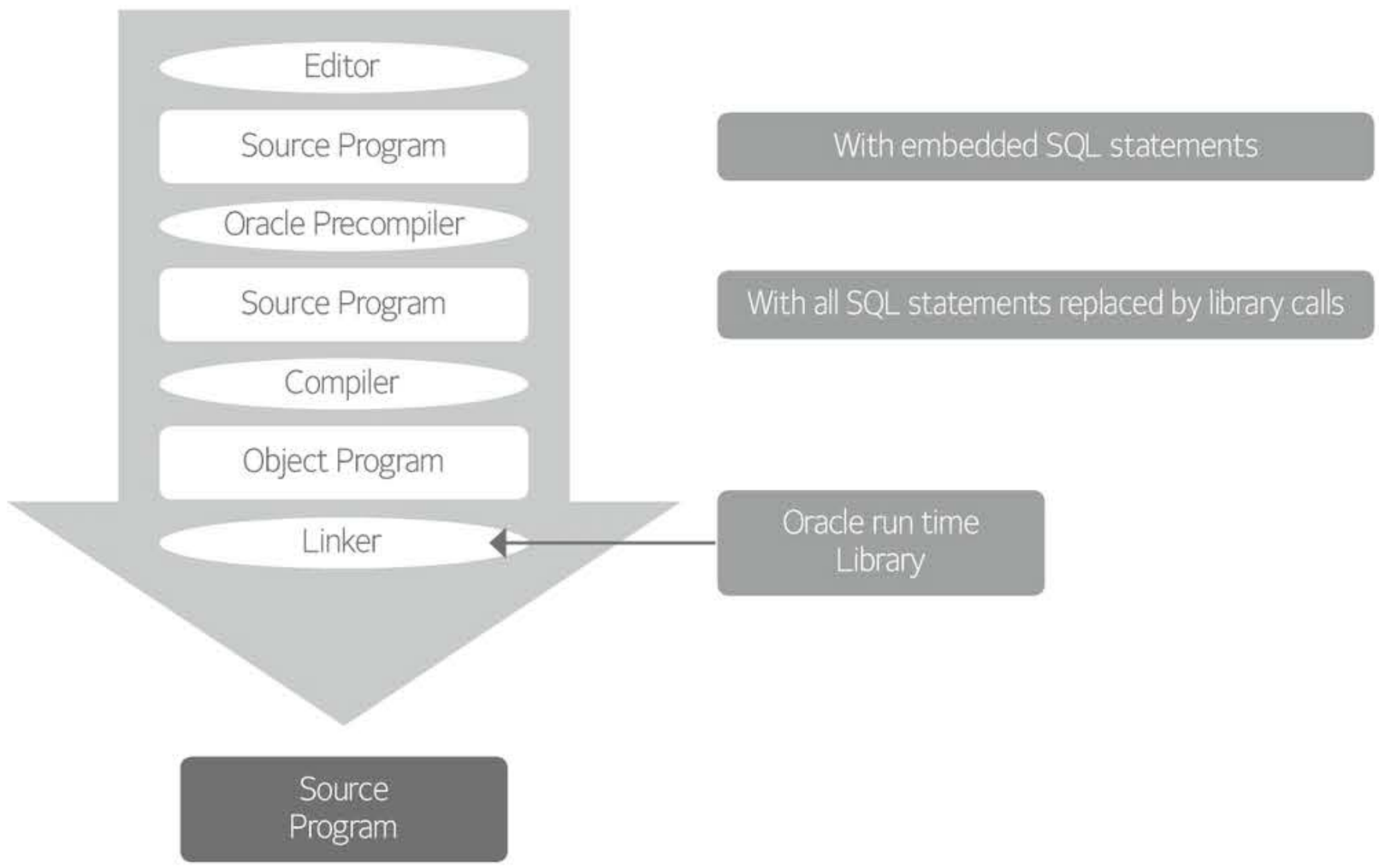
B) Characteristics of embedded SQL

① Embedded SQL statements can be used anywhere in the host program code if there is an execute statement. A

general SQL statement can return multiple tuples as an execution result, whereas an embedded SQL statement can return only one tuple.

② The tuple returned by the embedded SQL statement can be stored using a common variable.

③ The embedded SQL statement is separated and compiled by a preprocessor when the host program is compiled.



[Figure40] Conceptual diagram of embedded SQL compilation

④ The host program variable and the database field can have the same name.

⑤ The data type of the host program variable should match that of the corresponding database field.

⑥ The execution result, such as success, failure, or error, is passed as a string (or integer) value using an implicit SQL state variable called SQLCODE.

C) Embedded SQL cursor

Cursor indicates a pointer to the tuples of the table that are returned as the result of executing an embedded SQL statement. Multiple tuples, which can be returned as the result of executing an embedded SQL statement, can be accessed using cursor. The main statements used to operate cursor are as follows.

① Declare: A statement for declaration related to cursor, such as cursor definition.

② OPEN: A statement used to set cursor to point to the first tuple of the query result.

③ FETCH: A statement used to move cursor to the next tuple among the tuples of the query result.

④ CLOSE: A statement used to close cursor upon completing the processing of the query execution result.

03 Dynamic SQL

Unlike static SQL statements, dynamic SQL statements are dynamically processed according to program logic. As dynamic SQL and static SQL are in a trade-off relationship in terms of performance and flexibility, an appropriate method should be selected according to the development environment and purpose.

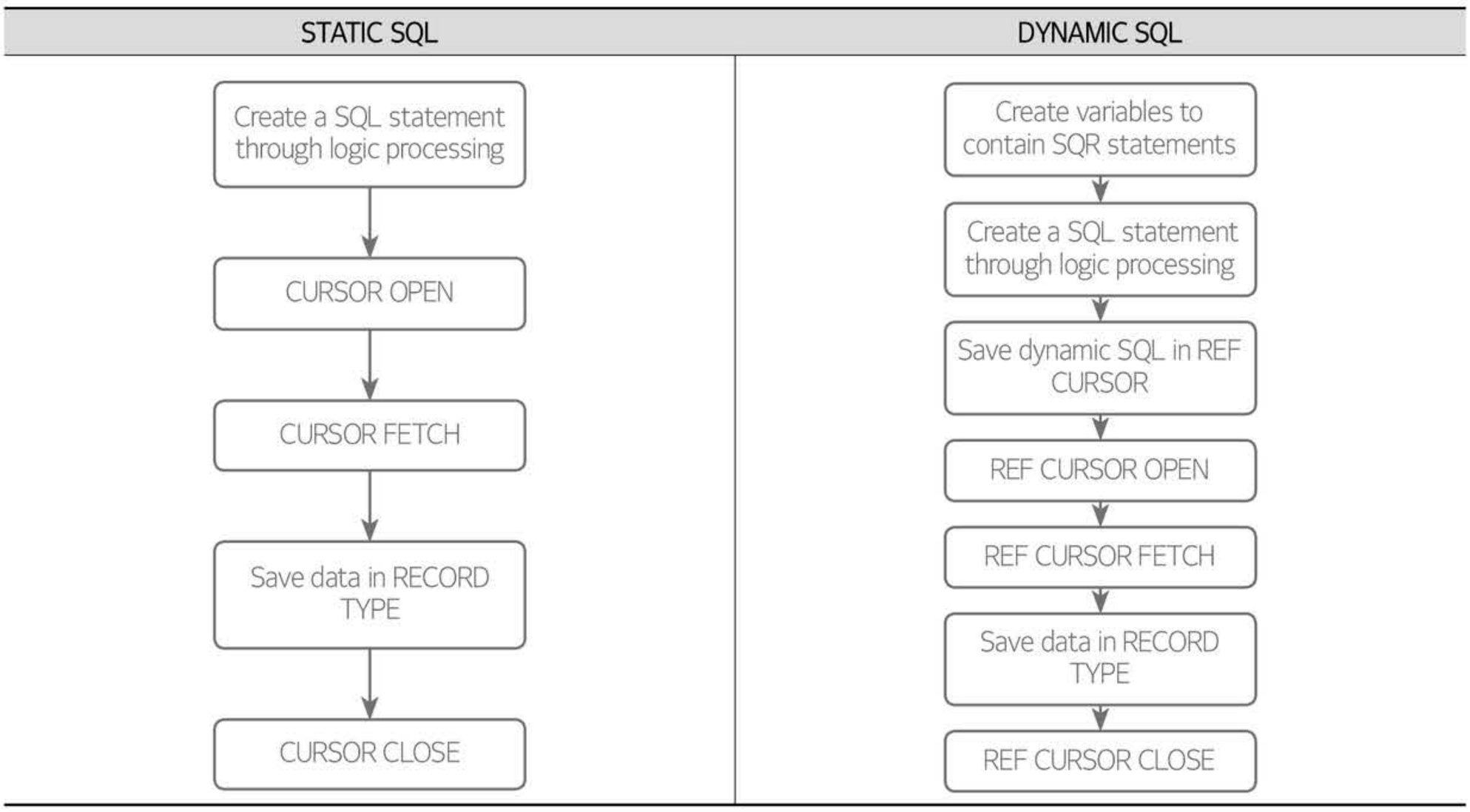
A) Comparison between dynamic and static SQL

<Table 45> Comparison of the characteristics of dynamic SQL and static SQL

Item	Static SQL	Dynamic SQL
Overview	An SQL statement is not assigned to a variable but is mixed in the code for description. An SQL statement is processed statically by declaring CURSOR.	An SQL statement is assigned to a string-type variable and processed dynamically. The SQL statement can be dynamically changed according to the condition. Part or all of the SQL statement can be received from the user at runtime.
Development pattern	Static SQL is declared in the CURSOR clause and then the data is processed in a looping structure between the BEGIN and END clause.	NVL () processing is not needed as the syntax of dynamic SQL can be changed.
Column composition	The column and WHERE clause cannot be changed.	As a DBMS is called after assigning an SQL statement in a variable, all SQL statements such as variables and columns can be freely processed with logic.
Execution plan	The optimizer sets up an execution plan for IS NULL and IS NOT NULL to process the condition wherein NVL () is processed. That is, if there are 6 conditions, the execution plan is divided by 12 CONCATENATION. As a result, a long hard parsing time is required to set up an execution plan.	The optimizer sets up an execution plan for pure access paths only because the execution plan does not have to be divided, since there is no WHERE condition whereby NVL() is processed. Therefore, the hard parsing time can be minimized.
Strengths	The execution speed is faster than dynamic SQL. SQL statements can be checked in advance during development. Code readability is high because the code is more intuitive than dynamic SQL.	An access plan for each SQL statement in the application is available based on the most recent statistical information. More diverse and flexible applications can be developed because the SQL statement is determined at runtime, not during development.
Shortcomings	The SQL statements should all be defined during development. The pre-compilation and bind processes are required.	The processing speed is slower than that of static SQL. The type, syntax, and privilege of the SQL statement cannot be checked before execution. The level of development difficulty is high, and development takes a remarkably long time.

B) Processing methods of dynamic SQL and static SQL

<Table 46> Comparison of the processing methods of dynamic SQL and static SQL



C) Example of dynamic SQL and static SQL code

① Example of static SQL

```
Init main()
\{
   Printf(" Input your employee ID :  ");
   Scanf("%d",&empno);
   EXEC SQL WHENEVER NOT FOUND GOTO notfound;
   EXEC SQL SELECT ENAME INTO :ename
            FROM EMP
            WHERE EMPNO = :empno;
   Printf(" Employee name: %s.\n",ename);
Notfound:
   Printf("%d is an invalid employee ID. .\n",empno);
\}
```

② Example of dynamic SQL

```
Int main()
\{
Char select_stmt\[50\] = "SELECT ENAME FROM EMP WHERE EMPNO = :empno";
// scanf("c", &select_stmt) ; ? SQL The SQL statement can be entered dynamically.
EXEC SQL PREPARE sql_stmt FROM :select_stmt ;
EXEC SQL DECLARE emp_cursor CURSOR SQL sql_stmt ;
EXEC OPEN emp_cursor USING :empno ;
EXEC FETCH emp_cursor INTO :ename ;
EXEC CLOSE emp_cursor ;
Printf(" Employee name: %s.\n", ename);
\}
```

04 Query Optimization and Optimizer

Query optimization refers to the process of making a DBMS select one optimal strategy by systematically evaluating several strategies for executing query statements. It is used by systems that support advanced relational query languages.

A) Query optimization process

① For query optimization, the intermediate query statement created by the parser is converted into a certain form of sub-expression and represented as a query tree.

② Then, an internal expression is rewritten into an efficient query statement using logical conversion rules. Multiple conditions are combined into one condition, and projection is performed as early as possible to remove any unnecessary attributes. Selection is also performed as early as possible.

③ Access plans are created and specified by the join procedure, selection procedure, etc.

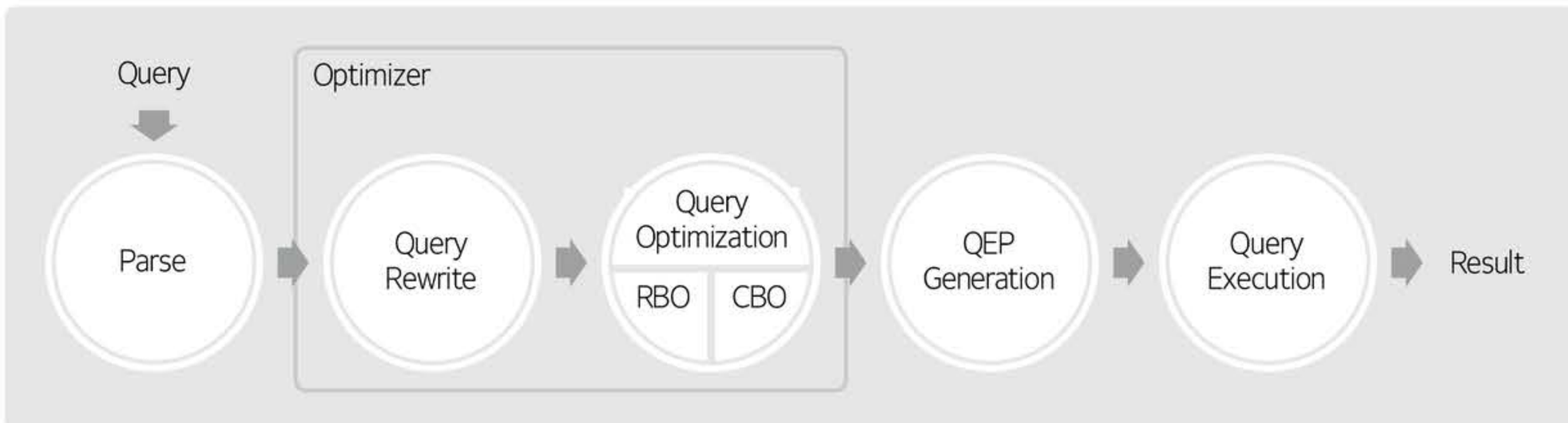
④ The most efficient access plan is selected by evaluating them. The evaluation criteria to consider include the disk access cost, storage cost, calculation cost, and communication cost.

B) Optimizer

An optimizer, as the core engine of the DBMS, checks for grammatical errors in SQL statements according to the various requests of the user, and then finds an optimal path or a processing procedure by creating and adopting the fastest data access path. When SQL specifies the desired data (what) in a relational database, the DBMS optimizer automatically decides and handles how to get the data. As such, the optimizer, which guarantees the physical data independence of the database and finds the method of executing the user's SQL queries efficiently, is making a significant contribution to the commercial success of the relational database.

C) Roles of the optimizer in each query processing phase

Query processing is divided into 5 phases - parsing, query rewriting, query optimization, QEP generation, and query execution. The optimizer acts in the query rewriting and query optimization phase. QEP (Query Execution Plan) refers to a query execution plan, which is the detailed information necessary to execute a query.



[Figure 34] Five phases of handling RDBMS queries

<Table 47> Roles of the optimizer in each query processing phase

Query processing phase	Processing details
Query rewrite	Checks whether a more effective query execution plan is available or not, merges the sub-query and view, and performs OR expansion.
Query optimization	Determines the access path for the query.

D) Classification of optimizers according to criteria

① Rule-based optimizer (RBO)

The rule-based optimizer sets the optimal path based on the ranking according to the index structure or comparison operator. It enables the user to accurately predict the path because the optimizer's decision is very regular and clear. However, it has the limitation that a serious judgment error may occur in terms of execution performance because the realistic factor (statistical information) is not considered.

② Cost-based optimizer (CBO)

The cost-based optimizer calculates the cost of each processing method and selects the method that costs the least. Even though there is the advantage of guaranteeing minimum performance without having to have a deep understanding of the optimizer through optimization control based on statistical information management and decision-making in consideration of reality, it has a limitation in that an execution plan cannot be forecast and controlled easily.

05 Linking the Web and Database

Web browser components and a gateway that connects the DBMS are required to link the web and database. Depending on the method of implementing the gateway, the methods of linking the web and database are divided into server extension methods and browser extension methods.

A) Server expansion methods

The server extension method is also called a stateless method. A database is connected through the web server, and a connection is established each time a request is made. These methods include the CGI executable file method, CGI application server method, extended API method, and servlet method.

① CGI executable file method

- The CGI executable file method executes the database application program as a CGI (Common Gateway Interface) executable file. The CGI directly accesses the DBMS to retrieve data. One application process is created for each request.
- This method has the advantage that the structure is simple, and the method can be implemented using most programming languages. In addition, users can select one of the connection methods provided by various database vendors, and it is very convenient because the user only has to modify the CGI program when expanding the system.
- However, this method has the disadvantages that the cost of content replacement increases exponentially when

the web client makes requests frequently or when the application program to link with the web is large, and that performance deteriorates significantly when there are too many connections.

② CGI application server method

- The CGI application server method separates and executes the application program accessing the database as a daemon and transfers the user's requests by application processor. The CGI module is composed of a small executable file and transfers the request received from the web server to the application server only.
- The application process includes the fork, pre-fork, and multithread methods. In particular, the multithread method has the advantage of a fast response speed by using system resources most efficiently.
- The CGI application server method has the advantage of easy linking with middleware such as CORBA and TP Monitor, while reducing the size of the process and keeping the existing system intact, but it also has the limitation of being difficult to implement.

③ Extended API method

- When the extended API method is used, a database application program is executed as a process by linking it with the web server. As the web server provides APIs with extended functions, it solves the problem of the existing CGI method wherein a new process should be created each time a request is received. Microsoft's ISAPI, Netscape's NSAPI, and Oracle's Web Request Broker (WRB) belong to this method.
- The extended API method has the advantage of saving more system resources, reducing more system load, and providing faster speed than the CGI method, because the routine created with the extended API of the web server is used as part of the server program. However, this method has the problem that it is difficult to create a routine using an extended API and becomes dependent on a specific web server or browser.

④ Servlet method

- The servlet method is similar to the extended API method but is platform independent. JDBC (Java DataBase Connectivity) or another database access class is used to connect a database.
- The servlet method has attracted the attention of many Java developers because it provides continuous processing and a fast response speed using Java's multi-thread features.

B) Browser extension method

The browser extension method processes data while maintaining the connection by directly connecting a database from a browser. It is also called the direct (state) method. JDBC, ADO (ActiveX Data Object), and OLE DB belong to this method.

① JDBC

- A method of directly connecting a database server from a Java applet or ActiveX control. Specifically, some methods use Java solutions such as JDBC and JavaBeans. When this method is used, the limitations of the web that occur when linking with mission-critical systems can be overcome by connecting a database server continuously.
- JDBC is an API for the programming language Java, which is designed to implement an independent Java application program. JDBC uses an integrated SQL server, and another database can be connected without modifying the source by replacing the driver or a module that can be linked to it only.

② ADO

- ADO, a Microsoft solution, is an ActiveX control that provides a database interface and manages connection with a database in the Inter or client/server environment.

③ OLE DB

- Although OLE DB provides access to all types of data including relational data (ISAM, VSAM, Excel Data, etc.) using a consistent API, it is limited to MS operating systems only.

XI. Concurrency Control

▶▶▶ Recent trends and major issues

Most of the databases we experience in the field are running in a multi-user environment. Therefore, the atomicity of transactions should be guaranteed by controlling interactions among transactions when multiple transactions are executed in a database simultaneously. To this end, concurrency control methods are used. Most of the DBMSs we use have these methods. Therefore, if transactions are wrongly designed due to a lack of understanding of the concurrency control technique, such as the locking mechanism applied to the DBMS, problems may occur due to a transaction delay, such as processing performance degradation, deadlock, and so on. And users who are accustomed to the fast processing of information systems may raise complaints about the service delay.

The recent DBMS has several advanced features such as a function that internally ensures the isolation of transactions and considers concurrent processing performance or the inclusion of an algorithm that detects and handles deadlocks quickly. However, those features are minimal and provisional measures, not a fundamental solution. Therefore, the concepts of transaction, concurrency control, and deadlock need to be clearly understood in order to develop a system that delivers high performance in a multi-user environment.

▶▶▶ Learning objectives

1. To be able to explain the concept and characteristics of a transaction.
2. To be able to explain the transaction-commit operation and the rollback operation for aborting a transaction.
3. To be able to explain the purposes, necessity, and techniques of concurrency control.
4. To be able to explain 2-phase locking (2-PL).
5. To be able to explain and apply a transaction isolation level.
6. To be able to explain the causes of and solutions to transaction deadlocks.

▶▶▶ Keywords

- Transaction
- ACID properties: Atomicity, consistency, isolation, durability
- Commit operation, rollback operation
- Lost update, dirty read, unrepeatable read, inconsistency, cascade rollback
- Serializable schedule
- Concurrency control techniques, lock-based technique, optimistic or validation-based technique, timestamp-based technique, multi-version technique
- Transaction isolation level: Read uncommitted, read committed, repeatable read, serialization
- Causes of deadlock occurrence: Mutual exclusion, block & wait, non-preemption, circular wait
- Solutions for breaking deadlock, deadlock avoidance, deadlock prevention, deadlock detection & recovery

✚ Preview for practical business Problems that arise when a transaction and concurrency control are not properly understood

There are some cases in which a set of operations executed in a database is viewed is viewed as a unit from the viewpoint of the database user. For example, transferring money from account A to account B may look like a single operation at first glance. However, the transaction is internally composed of two update operations, i.e. changing the balance of account A and changing that of account B, respectively. What if this operation fails in the middle of processing, and the balance of account A has been changed while that of account B remains unchanged? If any operation fails, all operations relating to the account transfer task should be either performed normally as a whole or returned to the original state prior to the operations. As explained previously, a set of operations that constitute a logical work unit is called a "transaction". The important thing here is that all operations of the transaction should be completely executed or returned to the original state before any transaction if an error occurs.

What if the update of the data shared between transactions cannot be controlled in a database system that executes several transactions simultaneously? For example, what happens if you transfer KRW 1,000 online from an account with a balance of KRW 1,000 and withdraw KRW 1,000 from an ATM at the same time? In reality, it is naturally impossible, because the database system controls these transactions. As explained so far, the database system provides a concurrency control method to secure the isolation of a transaction, which ensures that the transaction is not affected by the change made by another transaction at the same time. However, if the level of transaction isolation is high, simultaneous access to the database (concurrency) becomes difficult and performance is negatively affected. Therefore, the isolation level should be selected appropriately by considering the nature of the transaction.

In fact, service failure occurs frequently in the field due to the problem with concurrency control, including the case of the college course registration crisis that occurred due to the application of the isolation level of the banking system, which requires strict transaction isolation, to the college course registration system. Therefore, the concept of transaction and concurrency control should be correctly understood to determine the "appropriate" level of the system to be developed and transaction isolation.

01 What is a Transaction

A) Concept of a transaction

① A transaction is a set of several operations constituting a logical task unit, which converts a consistent state of the database to another consistent state.

② A transaction is also a unit of performing a series of read and write operations in a database, which should be performed at the same time.

[Figure 42] Conceptual diagram of a transaction

B) ACID properties of the transaction

<Table 48> Transaction properties

Characteristics	Cases of violating ACID
Atomicity	• Cheolsu has to process the task of entering 10 data as one transaction. However, he successfully enters 9 data but fails to input one datum. →All data should be successfully entered or failed in one transaction.
Consistency	• Ganghee performs an SQL operation with the logic of column 1 + column 2 + column 3 - Column 4, but the result varies each time he performs the operation. →The transaction result should always be the same expected value. A transaction should be controlled using Lock, Timestamp, Validation, etc. to prevent other transactions from intervening in the middle of processing the transaction.
Isolation	• Younghee is performing the task of modifying 100 data and the 35th datum is being updated in a database, but Cheolsu accesses the first modification datum and modifies it again. →Another transaction should not intervene and process the data until transaction commit or rollback is issued.
Durability	• Cheolsu has entered 10 items of member information and successfully committed the transaction, but the data are lost due to a temporary power failure. →The persistence of a successful transaction should be guaranteed. In this case, the data should be restored using the forward recovery algorithm.

C) when finishing a transaction

① Transaction commit operation: An operation that completes a transaction after permanently reflecting the results of all SQL statements in the transaction to the database.

② Transaction rollback operation: An operation that restores all data values that have been modified by a transaction so far to the state prior to executing the transaction (that is, aborting the transaction), when the transaction cannot

continue due to a fatal error in the middle of the transaction or because the user issues a rollback command.

D) Considerations when executing a transaction

① Implementation of transaction concurrency: Simultaneous execution of transactions can increase concurrency by increasing the transaction throughput and system utilization ratio and by reducing latency.

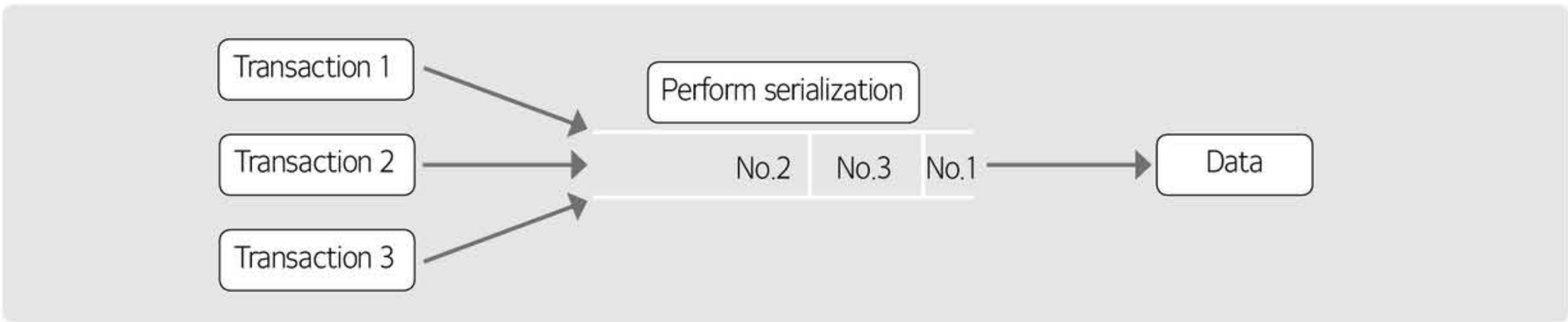
② Executing a transaction as quickly as possible: If a transaction takes too long, there is a high possibility that conflicts and deadlocks will occur due to the long locking time. Locking should be maintained long enough to ensure transaction serialization, but should be executed quickly enough so as not to deteriorate performance at the same time.

02 Concurrency Control

A) Definition of the serializable schedule of the transaction

① A serializable schedule means that even if each transaction is executed at the same time, the result is the same as the sequential execution of the transaction.

② Guaranteeing the serialization of transactions.



[Figure 43] Serialization of transactions

B) Definition of concurrency control

① A function that enables the successful execution of several transactions at the same time in a database system that supports a multi-user environment.

② A database system that supports a multi-user environment must have this function, which is also called concurrency control.

C) Purposes of concurrency control

① To create a serializable schedule or ensure the possibility of transaction serialization.

② To guarantee the maximum sharing level, minimum response time, and maximum system activity.

③ To ensure data integrity and consistency.

D) Problems that occur when concurrency is not controlled

<Table 49> Problems that occur when concurrency is not controlled

Item	Contents
Lost update	• Occurs when transactions update the same data simultaneously. • Occurs when the later transaction overwrites the update value before the previous transaction has completed the transaction after updating the data.
Dirty read	• Other transactions refer to the intermediate execution result of a transaction.
Inconsistency	• A database remains inconsistent when two transactions are executed at the same time.
Cascading rollback	• When a specific transaction cancels processing if data is shared by multiple transactions, the part processed by other transactions cannot be canceled.
Unrepeatable read	• When a transaction executes the same query twice, the query results differ because other transactions have modified or deleted the value in the middle.

E) Concurrency control techniques

① Lock-based technique

② Timestamp-based technique

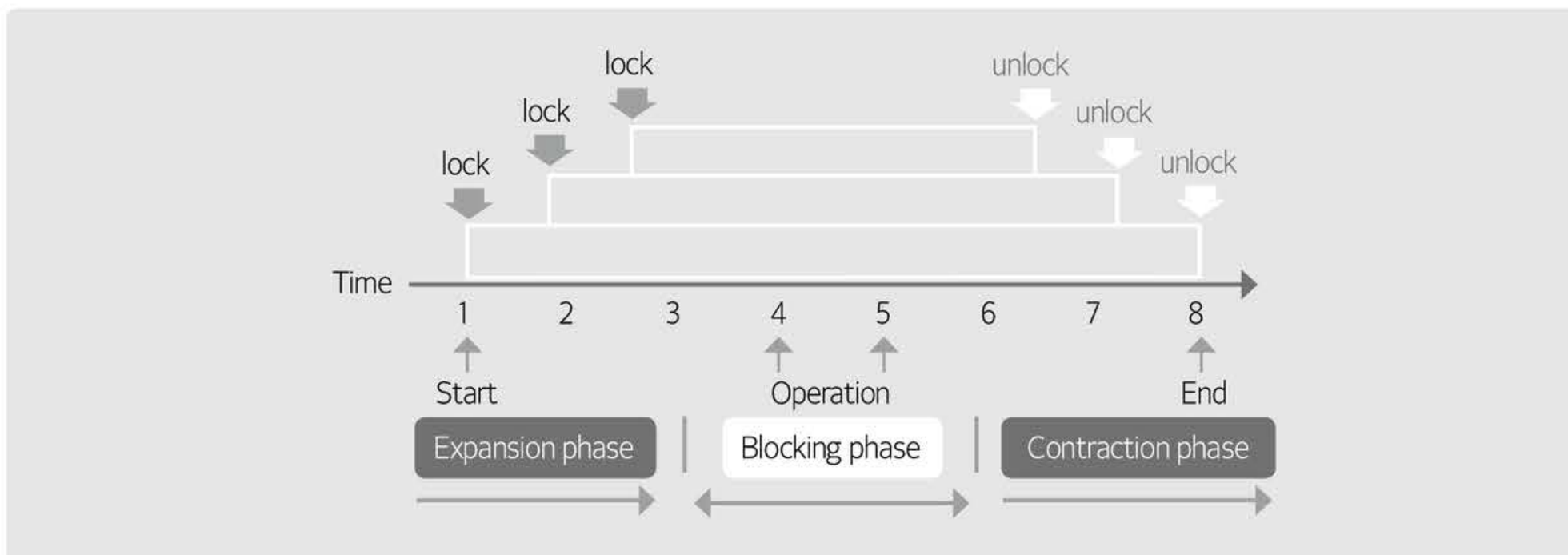
③ Multi-version technique

④ Verification (optimistic) based technique

F) What is the 2PL (2-Phase Locking) technique?

① The 2-phase locking technique controls concurrency by guaranteeing serialization and follows the 2PL protocol, which performs jobs by dividing 'lock' and 'unlock' operations into the expansion and contraction phases.

- Expansion phase: A phase in which a transaction can perform locking only, and cannot perform unlocking.
- Contraction phase: A phase in which a transaction can perform unlocking only, and cannot perform locking.



[Figure 44] Conceptual diagram of 2PL

② If all lock operations (Read_lock, Write_lock) appear before the first unlock operation, the transaction conforms to the 2PL protocol.

③ If all transactions comply with the 2PL protocol, the serialization of all transactions is guaranteed, but the reverse does not hold.

④ The 2PL protocol is most widely used as a protocol that can guarantee serialization.

03 Transaction Isolation Level

The following are the four transaction isolation levels defined in the ANSI/ISO SQL standard (SQL92).

A) Read uncommitted: Allows other transactions to read the data that are still being processed by the transaction (not yet completed).

B) Read committed: Allows other transactions to read the data that are confirmed by the completed transaction only.

C) Repeatable read: When a query is executed more than twice in a transaction, the record in the first query does not disappear or the value is not changed.

D) Serializable read: When a query is executed more than twice in a transaction, the record in the first query does not disappear or the value is not changed, and no new record appears either.

<Table 50> Concurrency according to the transaction isolation level

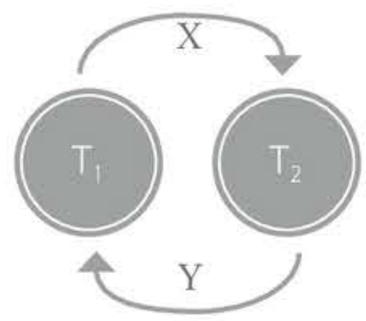
Isolation Level	Dirty Read	Non-repeatable Read	Phantom Read
Leve l 0 Read Uncommited	Occurs	Occurs	Occurs
Level 1 Read Committed	X	Occurs	Occurs
Level 2 Repeatable Read	X	X	Occurs
Level 3 Serializable	X	X	X

04 Deadlock

A) Definition of deadlock

① Multiple processes or transactions are waiting for the allocation of a specific resource infinitely in a multi-processing environment or a multi-transaction database system.

② Transactions in a deadlock can never end execution and cannot start another task because system resources are locked. Therefore, when a deadlock occurs, the system must cancel one of the two transactions.



[Figure 45] Conceptual diagram of a deadlock

③ All transactions are waiting indefinitely without making any progress at all.

- T1 waits for T2 to unlock data x.
- T2 is locking data x.
- T2 waits for T1 to unlock data y.
- T1 is locking data y.

<Table 51> Comparison of deadlock and infinity

Item	Deadlock	Starvation (infinite waiting)
Definition	Multiple processes are waiting for a certain event indefinitely without doing anything.	A specific process is waiting indefinitely for resource allocation.
Causes of occurrence	Mutual exclusion, hold and wait, non-preemption, circular wait.	Weighted resource allocation policy.
Solution	Prevention, avoidance, detection, recovery.	Aging technique.

B) Causes of deadlock

<Table 52> Deadlock occurrence conditions within a system (Deadlock occurs when certain conditions are met at the same time.)

Cause	Details
Mutual exclusion	As processes occupy a certain resource exclusively, other processes cannot use that resource.
Block & wait	A process has been allocated and is occupying a certain resource but requests another resource.
Non-preemption	The resource allocated to the process cannot be taken by force, and the resource can be released by the occupying process itself only.
Circular wait	The resource requests between processes form a circular shape.

C) Solutions to deadlock

<Table 53> Solutions to deadlock

Solution	Details
Deadlock prevention	• Use a protocol that ensures no deadlock will occur in a system. • Use this when there is a relatively high possibility that the system is likely to be in a deadlock. • Prevent mutual exclusion, partial allocation, non-preemption, and circular wait. • Avoidance techniques: Use the Wait-Die, Wound-Wait technique that uses a timestamp.

Deadlock detection and recovery	• Let the system remain in deadlock and solve it using the deadlock detection and recovery technique. • Detection: Check the system's status using a monitoring algorithm (Wait for graph reduction, cycle detection, and knot detection). • Recovery: Remove the deadlock by killing processes sequentially until the deadlock is broken. (Victim selection - Calculate the minimum cost of the process, rollback, starvation)

XII. Database Recovery

Recent trends and major issues

A database has the property that once it has been built, it should process actual business data stably. However, data can be corrupted due to a certain problem when they are digitalized and saved. In this case, it is vital to have a technology that can restore the database stably. This chapter explains how to process a database stably and the concept and techniques of backing up and restoring data.
The recent DBMS essentially operates database backup policies for stable data processing. In addition, backup methods are also evolving continuously to enable rapid backup, such as using the VTL (Virtual Tape Library).

Learning objectives

1. To be able to explain the concept and types of database failures.
2. To be able to explain the concept and methods of database recovery.
3. To be able to explain how to recover a distributed database.
4. To be able to explain and apply the types and characteristics of database backup methods.

Keywords

- Database failure: Transaction failure, media failure, system failure
- Recovery, redundancy, redo , undo, log
- Database recovery techniques: Log-based recovery technique (deferred update, immediate update), checkpoint recovery technique, shadow paging recovery technique
- 2-Phase commit protocol
- Database backup methods: Full backup, differential backup, incremental backup, archive log backup, hot backup, cold backup

✚ Preview for practical business Why database recovery is necessary

It has been reported that IT-related losses accounted for 16% (approximately USD 7 billion) of the total financial losses sustained during the 09.11 WTC attack in New York, even though there were also numerous casualties. Some enterprises suffered serious damages due to the loss of data and subsequently experienced difficulty resuming their online work because it took a couple of days to recover the lost data. As such, if the duplicated data and archives of a business system are not available, damages can be caused not only by simple system shutdown but also by the loss of the most important information essential to the survival of the company in an instant. In fact, the incident served as a timely reminder for all enterprises of the overriding importance of database backup and recovery, and led many enterprises to adopt a disaster recovery system, which is a non-disruptive recovery system.

Although they may not be the result of a terrorist attack as in the above example, failures frequently occur in the field, which cannot ensure transaction atomicity and durability due to various reasons such as hardware failure (e.g. broken hard drive), power outage, DBMS software defects, and so forth. Therefore, all database systems need a method of maintaining the atomicity and persistence of transactions. The recovery techniques used to restore a system to its previous state after the occurrence of a database failure (consistent state) should ensure high availability so that service downtime caused by the failure of the database can be minimized.

01 Concept of Database Failure and Recovery

A) Definition of data recovery

Data recovery refers to the restoration of the consistency and integrity of a database before the occurrence of an unexpected (database) failure while operating the database.

B) Types of database failures

<Table 54> Types of database failures

Type	Main content
Transaction failure	• Logical error: A transaction cannot be completed due to an internal error. • System error: An active transaction is terminated by force due to error conditions such as deadlock.
System failure	• Failure of power, hardware, software, etc. • Checks integrity to ensure that the stored contents are not affected by system failure.
Disk failure	• When a part or all of the disk storage is broken. • Re-executes transactions (redo) that were completed after dumping, using the most recent dump and log.
User failure	• Occurs due to the users' lack of understanding of the database. • DBA's mistakes made while managing the database.

C) Basic principles of database recovery: Principle of information redundancy

① Archive or dump: Copy and store a database to another storage device.

② Log or journal: Record the old values and new values of changing data each time a database is changed.

D) Types of database recovery actions

① Redo: Archive copy + log => Status after commit

- If the contents of the database itself are damaged, the most recent copy is loaded, and any changes made after making the copy are read from the log and executed again so that the contents of the database can be restored.
- This corresponds to the completed transaction at the time of failure occurrence.
- Changed values are stored in a disk (forward recovery) by executing the write operation again.

② Undo: Log + backward cancellation operation => Start status

- If contents that are being changed or have been changed cannot be trusted even though the contents of the database itself have not been damaged, the database is restored to its original state by cancelling all changes.
- This corresponds to the transaction in progress at the time of failure occurrence (incomplete transaction).
- Previous values are restored by cancelling the write operation (backward recovery).

02 Database Troubleshooting Method

A) Database recovery technique

[Figure 46] Types of database recovery techniques

<Table 55> Comparison of database recovery techniques

Item	Concept	Characteristics	Recovery process	Recovery speed
Log-based technique	Recovery using log files.	Checks all logs to determine Redo and Undo. Time delay. Redo of the transaction that does not need Redo.	Uses Redo and Undo.	Slow
Checkpoint recovery technique	Recovery using log files and checkpoints.	Relatively faster recovery speed than log-based recovery.	Uses Undo.	Faster than logs
Shadow paging technique	Recovery using shadow page tables.	Simple Undo and no need for Redo. Fast execution speed and simple. Difficulty in using it alone when several transactions are executed at the same time. Used along with log-based or checkpoint techniques. Overhead occurs while copying and recording shadow page tables.	Replaces shadow tables.	Faster recovery from copies and backups

B) Recovering from a distributed database 2-phase commit protocol

① Transaction of the distributed database

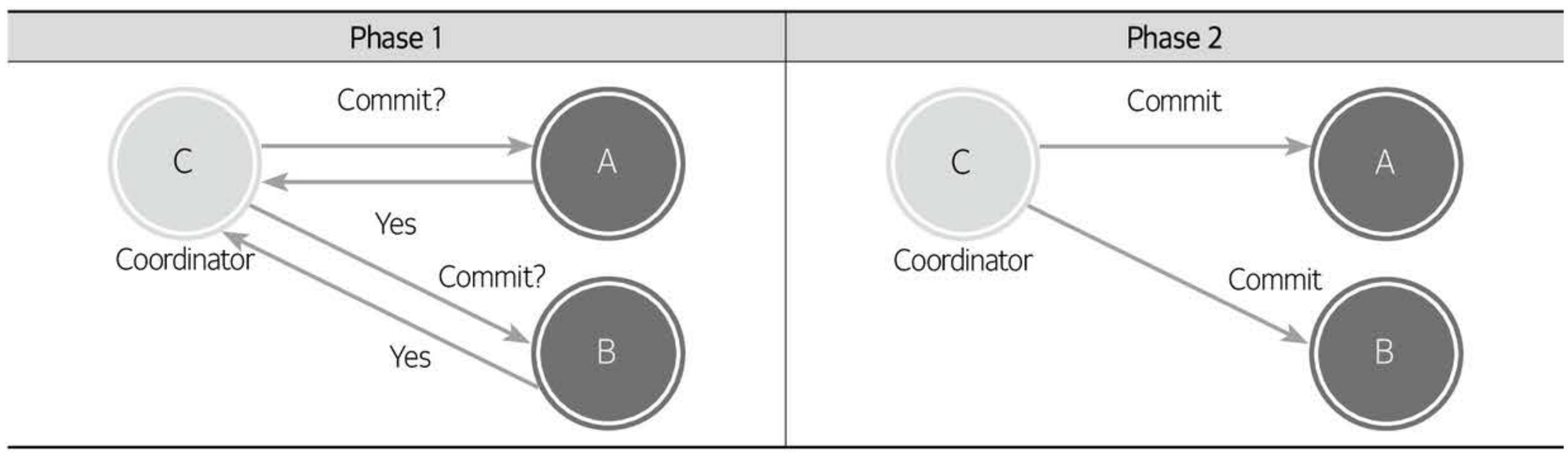
- This transaction accesses more than two databases, and each DBMS can use different recovery techniques and transaction managers.
- The atomicity of transactions is an important issue when building multiple database systems.
- If a certain transaction is executed across two sites and the transaction is not professionally designed, an inconsistent state can be caused when one site completes the transaction while the other site cancels the transaction.
- The 2-phase commit protocol should be used to maintain the atomicity of multiple database transactions.

✓ Either all participating databases should complete the transaction, or none of them should complete it.
✓ Even though a failure occurs in any participating database, recovery to the transaction completion state or cancellation state is possible.

② 2-Phase commit protocol

This is the transaction completion protocol that is most widely used to maintain the atomicity of transactions in a distributed database.

<Table 56> Execution of the 2-phase commit protocol



- Step 1: When a transaction is started by transaction coordinator C, coordinator C sends a message asking whether the transaction can be completed to databases A and B, and database A and B reply with the "ready" status.
- Step 2: If all participating databases are "ready", the coordinator sends a "completion (commit)" signal. If one or more participating databases signals a "not ready (abort)" signal, the coordinator sends a "cancellation (abort)" signal.

03 Database Backup

A) Database backup overview

① Definition of database backup

- The task of storing all or a part of a database in duplicate in order to recover the database in the event of a failure.
- In general, the database is backed up every day, week, or month in units of incremental, differential, or full.

② Purposes of backup

- Prevent data loss: Recovers a database using duplicated data in the event of a failure.
- Secure business continuity: Minimizes business disruption caused by IT failure while processing business.

B) Database backup requirements and main tasks

① Database backup requirements

<Table 57> Requirements by item

Item	Requirement
Business requirements	• Mean-Time-To-Recovery (MTTR) • Mean-Time-Between-Failures (MTBF)
Operational requirements	• Service operation requirements (e.g. 24x365). • Backup requirements at the test and verification level.
Technical requirements	• Physical copy of the OS image. • Logical copy of database objects. • Database configuration backup. • Considers how frequently the database should be backed up depending on the number of transactions that occur.
Matters related to disaster recovery	• Analyzes how much a business is affected by a disaster. • When computer equipment is completely lost due to earthquake, flood, fire, etc.

② Contents of database backup management

<Table 58> Contents of backup management

Item	Contents	Example
Backup policy	• The backup policy describes the backup cycle (daily, weekly, or monthly) and scope of backup (e.g. considers scattering to the outside). • The policy mentions concrete backup strategies, technologies, and equipment. • The policy describes the backup strategy for minimizing recovery time.	Daily: Database archive information Weekly: OS/user data backup Full backup: DB/AP/CM full backup₩
Backup configuration	• A technical backup architecture that can provide effective backup according to the backup policy is configured.	Technologies apply to disk, tape, VTI backup method, internal, LAN, SAN, mirror backup, etc.
Performing backup and managing the history	• Performs backup. • The backup result is recorded in the backup log and analyzed on a regular basis.	
Estimating backup capacity	• The amount of backup data is estimated to determine the number of disks or tapes to be introduced. • As backup has the characteristic of data duplication, the volume of data can increase by anywhere from two to six times its original size.	If the amount of original data is 1,000 GB, 2,000 GB is expected for mirror backup, 3,000 GB for disk backup, and 5,000 GB for disaster recovery.

C) Types and characteristics of database backup methods

<Table 59> Types and characteristics of database methods

Item	Backup method	Description	Advantages/disadvantages
Classification by service interruption status	Offline backup	• Backup after stopping a database.	• Simple. • Database is unavailable during backup (service interruption).
	Online backup	• Backup while operating a database.	• No service disruption. • Increased CPU/memory usage

Item	Backup method	Description	Advantages/disadvantages
Classification by backup range	Full backup	• Full backup is the process of making a copy of the entire database.	• Full backup is the process of making a copy of the entire database. • Backup takes a long time, and costs are high due to data duplication.
	Differential Backup	• A differential backup is a cumulative backup of all changes made since the last full backup. E.g. If full backup is set for Sunday and differential backup for Thursday, all changes after the Sunday backup will be backed up.	• The amount of data backup can be reduced compared to a full backup. • There must be a full backup. • More data are backed up in duplicate compared to an incremental backup. • The recovery time can be reduced because only the latest full backup and the last differential backup copy needs to be applied when restoring.
Classification by backup range	Incremental Backup	• An incremental backup is a type of backup that only copies data that have been changed or created since the conduct of the previous backup activity. E.g. If full backup is set for Sunday and differential backup for every day, only data accumulated after the Wednesday incremental backup will be backed up on Thursday.	• The amount of data backup can be reduced compared to a full backup. • Fewer data are backed up in duplicate compared to a differential backup. • There must a full backup. • The recovery time increases because the latest full backup and all incremental backups should be applied when restoring. • Recovery fails if any backup files are missing.
	Archive Log Backup	• A log file backup that records the transaction execution details.	• There is an advantage in recovering to a specific point in time because the transaction record in the log file can be read and executed again. • Data loss can be minimized because data can be restored to their original state immediately before the failure. • As database data are not backed up, there must be a data copy when the database itself is damaged.

XIII. Understanding Database Analytics

Recent trends and major issues

As the business environment has switched from a seller-centered model to a buyer-centered model, ever faster and more accurate decision-making is directly related to the survival of enterprises. Accordingly, analytical systems are needed that support a series of analysis processes, including the collection, storage, processing, presentation and utilization of data, in order to analyze enterprise information including customers more quickly and accurately. In particular, the data warehouse (hereafter referred to as a "DW") for storing large amounts of corporate information accumulated over time, the OLAP for diversely analyzing and utilizing data from the user's perspective, and data mining for finding new business insights have been introduced.

On the other hand, data warehouses that were previously available for each area are now being replaced by the EDW (Enterprise DW) due to the development of large-scale data storage technology, and semi-real-time data warehouses have also been introduced due to the development of data processing technology. In addition, as enterprises are adopting the Hadoop system for big data analytics, its coexistence with existing analytics data warehouses is required.

Learning objectives

1. To be able to explain the concept and characteristics of the data warehouse (DW).
2. To be able to explain data warehouse modeling.
3. To be able to explain ETT.
4. To be able to explain the concept and search technique of OLAP (Online analysis processing).
5. To be able to explain the concept and algorithm of data mining.

Keywords

- Data warehouse (DW), star schema, snowflake, ETL, ETT, OLAP, search technique, data mining, association analysis, sequential analysis, classification, clustering

✚ Preview for practical business Why do we need to understand data analysis?

Recently, enterprises in most industries have begun using analysis systems for handling business and diverse other various purposes, such as CRM, SCM, etc., in order to analyze accumulated data from various perspectives, and to use the information derived from such analysis directly in their business operations, including management decisions, sales, marketing, new product releases, etc. In fact, data analysis systems are becoming increasingly important as the key system of enterprises' business. For example, Mart A analyzes the shopping details of individual customers and sends them personalized coupons to induce them to make additional purchases. Meanwhile, Investment & securities company B analyzes the wealth, age, residential area, and investment product purchase history of its customers and discovers that "People in their thirties who have financial assets of about KRW 30 million and live in rural areas are the most active purchasers of investment products." Therefore, the company uses that information to develop company marketing strategies and investment products.

In addition, markets are calling for more and more experts who are capable of building and operating data analysis systems.

As explained, data is very important in these days. Therefore, it is necessary to understand data processing such as pre-processing, loading, analysis, etc., and to understand technology about ETL, DW, and OLAP.

01 Concept and Characteristics of the Data Warehouse (DW).

A) Concept of the data warehouse

A data warehouse is an integrated system or database that enables the user to instantly analyze internal data and external data generated by the operation system of an enterprise over time, without the need for separate programming from multiple points of view, by integrating data by subject.

B) Characteristics of the data warehouse

Unlike the data repository of OLTP for OLTP systems, the data warehouse generally has the following characteristics.

<Table 60> Characteristics of the data warehouse

Characteristics	Description
Subject-oriented	• Among the multiple types of operation system data that are managed by data business functions, the data of a specific subject needed for decision-making activities from an enterprise perspective is saved, while other data are not included.
Integrated	• The structure of a data warehouse is characterized by data consistency and physical unity through company-wide data standardization. • When obtaining data from the operation system, a series of data conversion tasks are performed to integrate the data.
Time variant	• To analyze past and present trends and forecast the future, a data warehouse retains data for a long time in the form of a series of snapshots. • Users can understand the process of data change over time using the data history.
Non-volatile	• A data warehouse is a read-only database that cannot be deleted or modified once it has been loaded from the operation system database. • When a modification occurs in the operation system data, existing data are deleted. The data in the data warehouse stores the history of data at each point in time.

02 Data Warehouse Modeling

A) Definition of data warehouse modeling

Unlike general ER modeling for OLTP systems, data warehouse modeling refers to a data modeling technique that, from a data analysis perspective, enables users to analyze large-scale data from various viewpoints. Data are generally organized into fact tables and dimension tables so that end users or analysts can easily analyze information. The table below shows the main components of data warehouse modeling.

<Table 61> Components of data warehouse modeling

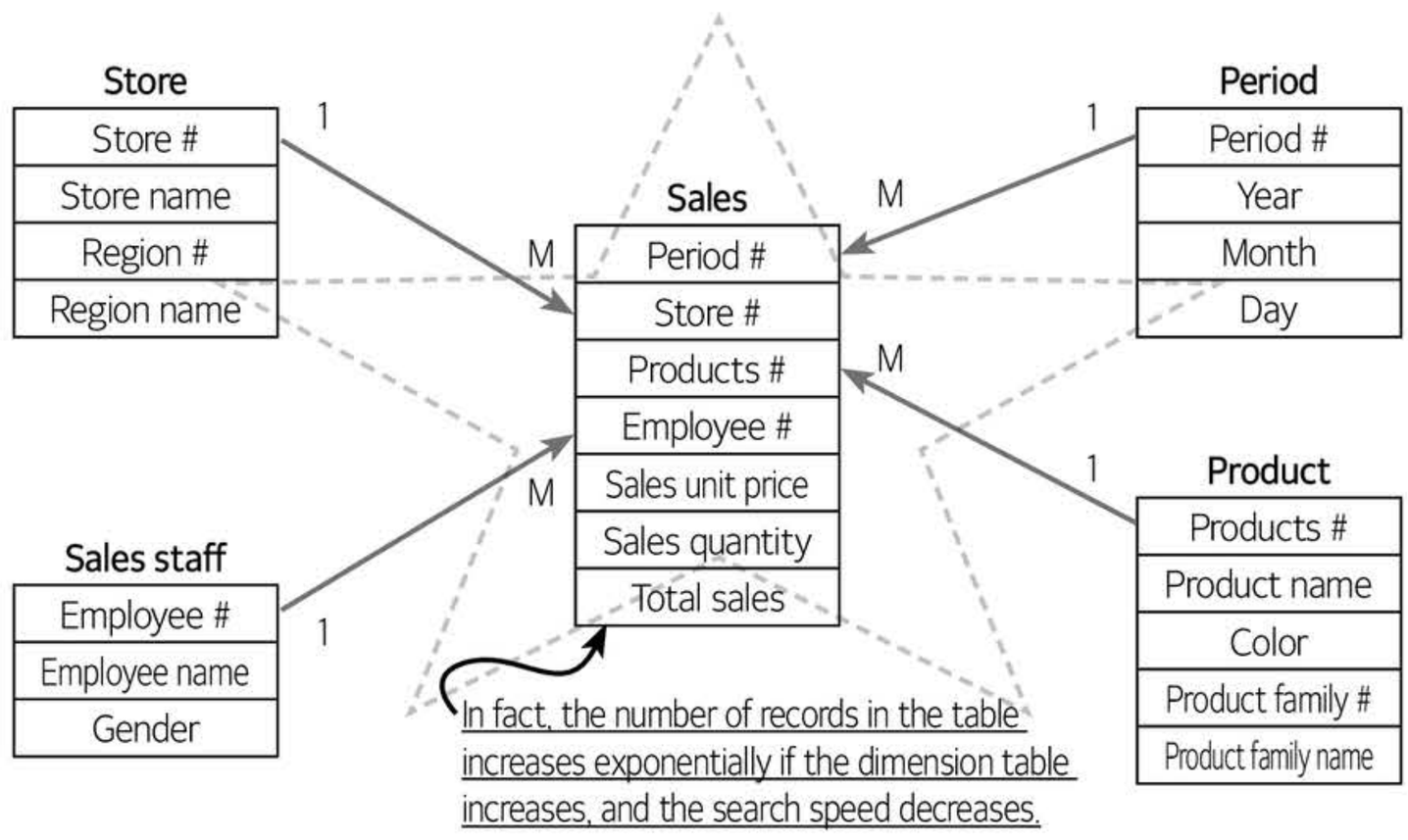
Item	Description
Fact table	• A core table composed of a set of highly relevant measures. • As measurement data, a Measure can observe the goal of information analysis, such as the amount, number, time, etc.

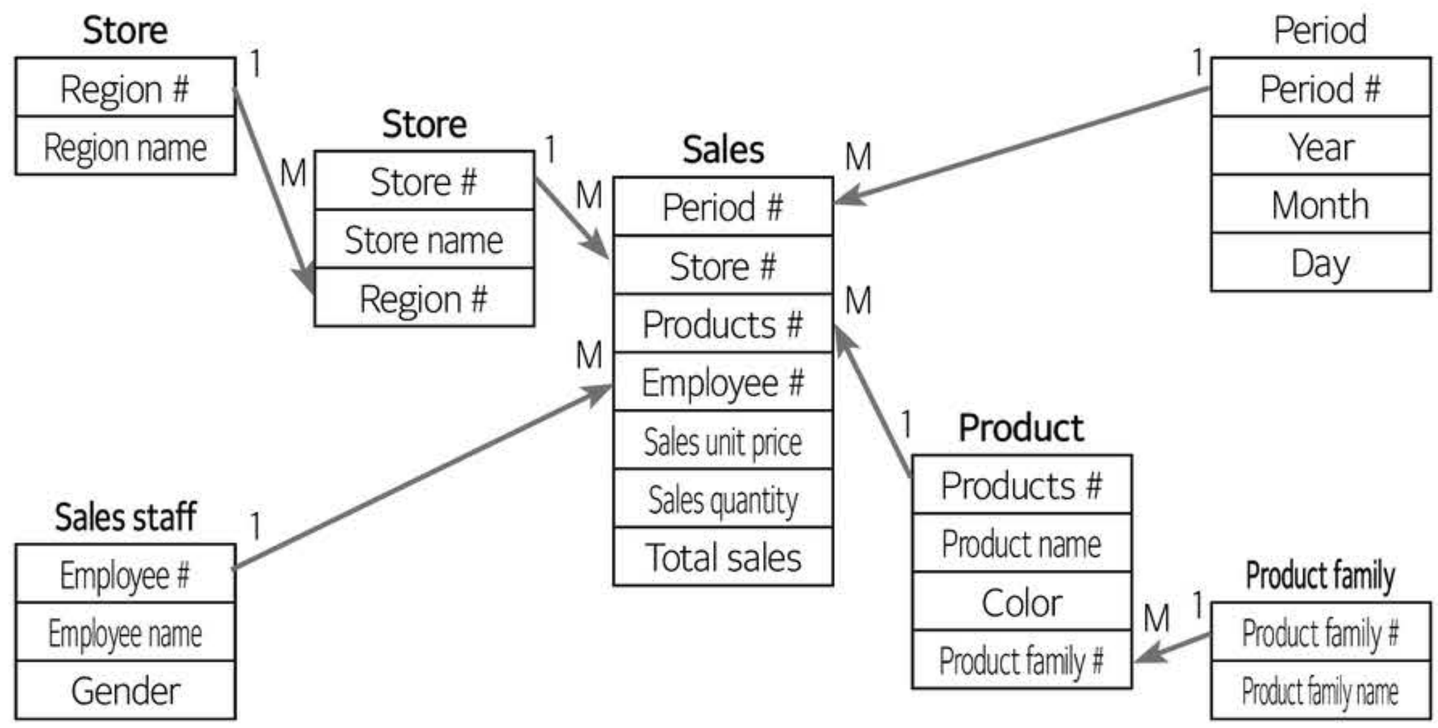
Dimension table	• A sub-table and a perspective of analyzing each fact. • A dimension table has multiple attributes, thus allowing data analysis from diverse perspectives.

B) Data warehouse modeling technique

A data warehouse model organizes data using fact tables and dimension tables to facilitate the analysis of information. The technique can be divided into the 'star schema' and the 'snowflake' depending on the dimension table normalization status.

<Table 62> Data warehouse techniques

Operator	Description
Star schema	• A modeling technique for designing data by separating it into fact tables and dimension tables. • Data duplication occurs because dimension table data are not normalized. • The schema is easy to understand and has few joins, thus improving query performance, but data consistency problems may occur. 

Snowflake schema	• A modeling technique for completely normalizing the dimension table of the star schema. • Data duplication is rare, and few storage spaces are used owing to the normalization of the dimensional table, but there is some concern about performance degradation due to the greater number of joins compared to the star schema. 

03 Concept of ETL (Extraction, Transformation, Loading)

ETL refers to the entire process by which data are extracted from the source system and stored in the data warehouse after cleansing and conversion. It plays the role of maintaining data consistency and integrity among the components of the data warehouse, and is also called ETT (Extraction Transformation, Transportation).

<Table 63> Work of each phase in ETL

Phase	Main task
Extraction	• The phase in which data are extracted from the original file or operating system database and stored in the data warehouse. • In the past, data were extracted on a daily or monthly basis, but in some recent cases data were extracted in real time using database logs according to the business requirements.
Transformation	• A phase in which extracted data are cleaned and converted into a data format suitable for the data warehouse. • In the event of data quality problems, data are cleansed according to the reference data or business rules. • The original data format is converted into a data format suitable for the data warehouse.
Loading	• A phase in which converted data are sent to the warehouse for storage and the necessary indexes are generated. • Full and partial update techniques are available.

04 Concept and Search Technique of OLAP (Online Analytical Processing)

A) Concept of OLAP

OLAP refers to the process by which the end user accesses multi-dimensional information without an intermediary or medium, and then analyzes the information interactively and uses it for decision making. That is, when the operational data extracted and converted by ETL are stored in the data warehouse or data mart, the end user analyzes them using OLAP.

B) OLAP search technique

OLAP provides various search techniques that allow end users to analyze data from diverse perspectives and summary levels. The table below shows the main search techniques of OLAP.

<Table 64> Main OLP search techniques

Search technique	Description
Drill Down	• A search technique that approaches a specific analysis topic in phases from a high summary level to a low (detail) summary level. E.g. Time dimension: Year → monthr → day
Roll Up	• Concept opposite to Drill Down • A search technique that approaches a specific analysis topic in phases from a low summary level to a high summary level. E.g. Time dimension: Day r → month r → year
Drill Across	• A search technique that uses a certain analysis viewpoint on one analysis topic to approach another analysis topic.
Pivot	• A search technique that changes the axis of the analysis perspective on a specific analysis topic.
Slice	• A search technique that creates subsets by selecting specific values for the members at one level or the members above that level.
Dice	• A search technique that creates subsets by slicing more than two dimensions.

05 Concept and Algorithm of Data Mining

Data mining refers to a series of processes that identify a systematic statistical rule or pattern among a large amount of data, convert it into meaningful information, and apply it to corporate decision-making. The table below shows the major data mining algorithms.

<Table 65> Data mining algorithms

Algorithm	Description
Association	• An analysis algorithm that discovers a pattern using a combination of highly relevant data in transaction data, etc. • Apriori algorithm, etc. • This algorithm is mainly used to place products by analyzing offline stores, and to recommend related products automatically at online shopping malls, etc.

Sequence	• An analysis algorithm that searches the correlation of items over time by adding the concept of time to association analysis. • The possibility of a given transaction occurring in the future is forecast by performing time series analysis on transaction history data. • Apriori algorithm, Generalized Sequential Patterns (GSP), etc.
Classification	• An analysis algorithm that creates a tree-type model, which classifies the values (category values) of a specific attribute (category type) by analyzing a dataset when it is given. • Decision tree algorithm, etc.
Clustering	• An analysis algorithm that groups records with similar attributes, by considering several attributes of given records (customers, products). • K-Means algorithm, EM algorithm, etc.

XIV. Understanding Big Data and NoSQL

Recent trends and major issues

It is essential to understand technologies related to big data analytics because the social demand for big data analytics and big data technicians (engineers, data scientists, etc.) is increasing significantly. In addition, NoSQL, which has the characteristics of a BASE focusing on the fast-processing speed for unstructured data and large-scale data, is increasingly being adopted in enterprises' operating environment. Therefore, it is necessary to understand NoSQL (Not Only SQL) as it is based on a different concept to existing relational databases.

Learning objectives

1. To be able to explain the concept of big data and related technologies.
2. To be able to explain the concept and characteristics of NoSQL.

Keywords

- Big data, 3V, unstructured data, Distributed File System (DFS), NoSQL, CAP theory

+ Preview for practical business Why do we need to understand big data technology?

Although interest in data analysis has increased, the existing system architecture and DBMS face limitations in terms of processing speed and performance when an existing analysis system processes tens of PBs of unstructured data generated by multimedia, SNS, sensors, IoT, etc., due to the advancement of IT technologies. As a result, solutions are being developed that are suitable for analyzing the large volume of unstructured data (variety) generated at high speed (velocity). Technologies related to big data, which was still in its infancy only a few years ago, have developed rapidly and are now being used directly in our real life.

There are many examples of this. For instance, the US movie Minority Report, which was released in 2002, portrayed futuristic crime prediction by "precogs" in the year 2054, yet something akin to this technology was actually realized as a crime prevention system in San Francisco as early as 2009. (However, the concept of "big data" had only been introduced at that time for the study of genomes, and prediction based on big data analytics had not yet been thought of.) Other commonly known examples include Google's flu map, the US presidential election case, ZARA, the DHL case, and distribution demand forecasting.

Furthermore big data technology was unfamiliar to IT workers until relatively recently to the extent that they only needed to know the definition of big data. However, with the rapid development of big data technology, terms such as crawler, Hadoop, MapReduce, R, and NoSQL, as well as the 3V characteristics of big data, have become familiar technical terms, and IT workers now experience big data systems more frequently in the workplace. Therefore, IT workers need to understand at least the concepts of the technical terms for each phase of big data analytics, if not the detailed technical principles, in order to quickly adapt to the changed workplace.

01 Big Data Overview

A) Definition and characteristics of big data

① Definition of big data

Big data generally refers to either data that exceed the ability of database management tools used to capture, store, and analyze data (McKinsey, 2011), or to next-generation technologies and architectures designed to extract value from large-scale data at low cost and support the rapid collection, discovery, and analysis of data (IDC, 2011).

② Characteristics of big data (3V)

The characteristics of big data can be explained by the three elements (3V) of big data, namely, volume, velocity, and variety. Each element has the following characteristics.

<Table 66> Characteristics of the three elements of big data (3 Vs)

Three elements	Description
Volume	- Refers to a volume of data of tens of terabytes, petabytes or more, thus exceeding the processing limit of commonly used software when collecting, storing, and processing data.
Velocity	- 'Big data' is created very quickly. - Data collection, processing, storage, and analysis need to be processed in real time.
Variety	- Diverse kinds of data. - Big data can be classified into structured, semi-structured, and unstructured data.

※ 6V of big data = Volume, Variety, Velocity, Veracity, Visualization, Value.

③ Structured data vs. unstructured data

<Table 67> Description of structured, semi-structured, and unstructured data

Data type	Description
Structured	- Data to be stored in a fixed field.
Semi-structured	- Data not stored in a fixed field, but which contain metadata or schema, such as XML or HTML. - XML, CVS, XLS, RDF, etc.
Unstructured	- Data not to be stored in a fixed field. - Document, picture, video, and audio data, etc.

B) Detailed technology by big data life cycle

<Table 68> Detailed technology by big data life cycle

Item	Description	Detailed technology
Collection	- A technology that can collect data from all devices and systems.	- Crawling (web robot), ETL, CEP (Complex Event Processing), etc.
Storage/ processing	- A technology that can store and process collected large-scale data using a distributed processing system.	- Distributed file system, NoSQL, MapReduce, processing
Analysis	- A method of analysis that can assist companies and the public with using big data in business and daily life.	- Natural language processing, machine learning, data mining algorithms, etc.
Visualization	- A technology that can visualize analyzed results effectively.	- Visualization such as R, graphs, drawings, etc.

02 Technologies Related to Big Data

A) Collection technology

Various technologies can be used for data collection, such as ETL, web crawling, RSS Feeding, Open API, and CEP (Complex Event Processing). Among them, the web crawling technology automatically collects various documents and data generated on the web, and is used to collect data such as SNS, blogs, and news. Web crawling copies the entire web page after collecting the URLs to be collected, or collects data with a specific tag only after analyzing the HTML code.

<Table 69> Examples of big data collection solutions

Technology	Description	Solutions
Collection using the DBMS	Collects data using the SQL function of the DBMS.	Oracle, MariaDB, MS SQL, Tibero, etc.
Collection using sensors	Collects data when a certain condition is met.	CQL, Kafka
FTP collection	Collects data using a port that can transfer files.	
HTTP collection	Collects data by reading HTML tags.	Scraper

B) Big data storage/processing technology

The Distributed File System (DFS), NoSQL, MapReduce, and other technologies are used to store and process large amounts of data and unstructured data (that is, big data) generated at high speed in an efficient, cost-effective manner. Recently, the distributed file system, which is based on the cloud and uses virtualization technology in the cloud computing environment, was introduced.

<Table 70> Example of big data storage/processing technology

Technology	Description	Solutions
Distributed File System (DFS)	A file system that allows access to files on multiple host computers which are shared over a computer network.	GFS (Google File System), HDFS (Hadoop Distributed File system), etc.
NoSQL (Not Only SQL)	A new type of data storage/retrieval system that uses a less restrictive consistency model (BASE characteristics) than the traditional relational database.	HBase, Cassandra, Mongodb, CouchBase, Redis, Neo4J, etc.
Distributed parallel processing	A technology that processes a large amount of data in a distributed parallel computing environment.	MapReduce

① Distributed File System (DFS)

The DFS is a file system architecture for storing and processing large-scale and unstructured data in a distributed environment. It has the following characteristics.

- It is composed of inexpensive servers.
- Scale-out: Its entire available capacity and performance increase almost linearly each time equipment is added.
- High availability: Even if some servers fail, the usability of the entire system is not affected very much.
- Optimized for throughput: It is suitable for the batch processing of large-scale data.

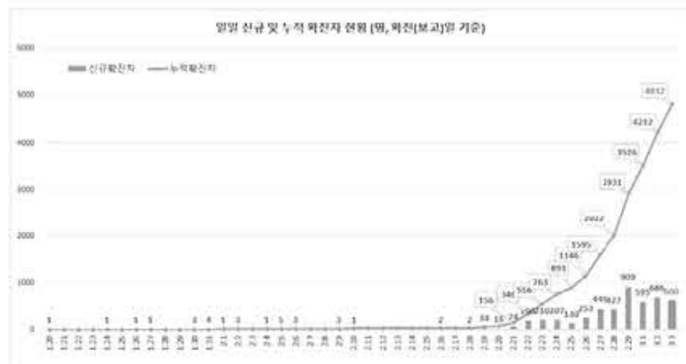
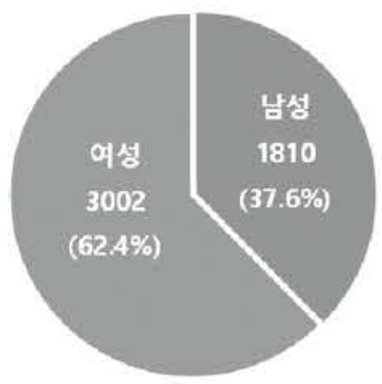
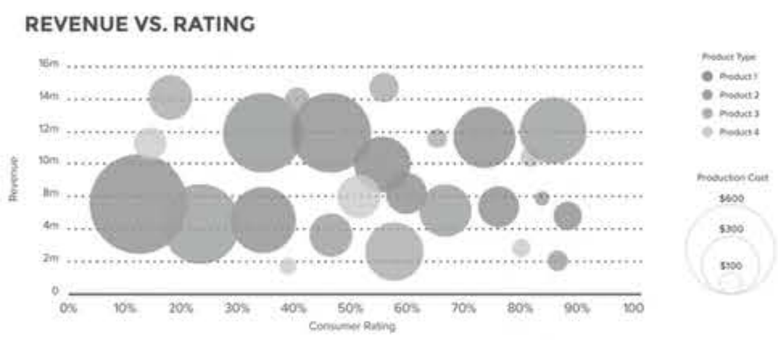
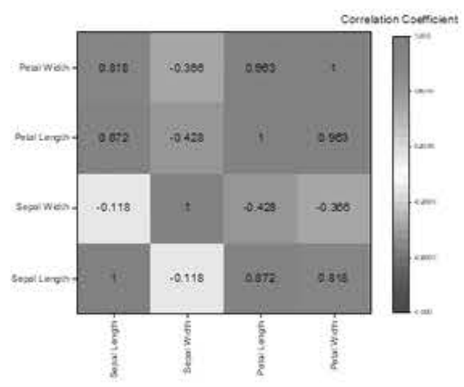
② MapReduce

MapReduce is a programming model designed for the parallel distributed processing of big data using inexpensive machines. This model can process large amounts of data in parallel using a program composed of a map procedure and a reduce method. MapReduce allows the analysis of large-scale data by processing data that has been distributed and stored in multiple machines. Basically, MapReduce performs batch-based processing and can handle large-scale data conveniently. The data of the execution result is copied, distributed, and stored safely in consideration of the failure of the physical device.

C) Visualization technology

This technique provides insights by effectively transferring numbers, statistics, and valuable meanings, by classifying data for the user's easy understanding, and by analyzing large-scale data. It allows the exact and effective transfer of information without actually requiring the user to look at the data.

<Table 719> Big data visualization method

Technology	Description	Example
Time visualization	Shows the passage of time. Continuous, segmented.	
Distribution visualization	Shows the relationship between the whole and the part, and the ratio. Pie, Treemap	
Relationship visualization	Shows the relationship between two or more variables. Bubble chart, histogram	
Comparison visualization	Shows spaces and shadows intuitively. Heatmap, Stars	

Spatial visualization	Shows information by mapping it on the map. Including POI data	

D) Classification of big data analytics

Big data analytics refers to the process of "discovering meaningful patterns from big data."

<Table 72> Classification of data analysis

Classification criteria	Modeling	Contents	Applied technique
Purpose of use	Descriptive modeling	The primary purpose is to find patterns that describe the given data.	Association rule, clustering, database, segmentation, visualization, etc.
	Predictive modeling	A model is created based on the given data, and is used to predict new input data.	Classification Regression, time series analysis Neural network SVM
Presence of target variable	Supervised data	When the target is determined.	Decision Tree Neural network Case-based reasoning
	Unsupervised data	When there is no target. The correlation or similarity between data is analyzed with the focus on input variables.	Association rule discovery, market basket K-means clustering

E) Main methods of big data analysis

<Table 73> Methods of big data analysis

Concept	Description
Logistic regression	• A statistical technique used to predict the possibility of an event's occurrence (probability of occurrence) using a linear combination of independent variables.
Decision tree analysis	• A method of quantitative analysis that classifies an interested group into several subgroups or performs Prediction by drawing a decision tree chart.
Neural network analysis	• A method of analysis that handles a problem with parallel/distributed/probabilistic calculation based on the idea that digital information is a network of nerve cells, rather than a method of processing digital information based on a deterministic binary computational model using the human brain itself as the model.
Text mining	• A technology that extracts and processes useful information by applying natural language processing technology and document processing technology to unstructured/semi-structured data. • The core technologies of text mining include document summarization, document classification, document clustering, and feature extraction.

SNA (Social Network Analysis)	• A social network refers to the network between the components constituting a given society. • An analysis methodology that analyzes and visualizes the relationship between objects - such as people, groups, organizations, computers and data - and the characteristics and structure of the network.
Opinion mining	• A technology that quickly analyzes the information the user wants and intelligently infers meaningful information from a large number of unstructured reviews such as SNS and replies. • It is used effectively for corporate marketing policies or public opinion analysis by extracting the hot topics of the social network service and analyzing the flow in real time.
Natural Language Processing (NLP)	• An artificial intelligence technology that understands, creates, and analyzes human language using computers. • The process of understanding natural language by analyzing human language mechanically to convert it into a form that can be understood by computers. • Or various technologies that express such a form in language that humans can understand. • Natural language is processed in the following order: preprocessing →morpheme analysis →syntax analysis →semantic analysis →dialog analysis.

F) Data scientist

A data scientist is an expert who can collect, organize, investigate, analyze, and visualize data. The data scientist provides information necessary for corporate/organizational decision-making by collecting, analyzing, and discovering the value of data, using various platform foundations and analysis infrastructures related to big data.

<Table 74> Capabilities of the data scientist

Point of view	Capability	Details
Mana-gement	Business	Understanding the business of the company concerned and expressing it as a business model.
	Data management	Exploring and integrating the internal/external data of a corporation, and manipulating structured/unstructured data.
	Data analysis	Predictive analysis based on data mining/statistics, and analysis based on cognitive psychology, R, and visualization techniques.
	Change management	Establishment of the data strategy, communication skills.
Technology-capability	Understanding statistical analysis tools	• Experience and training are needed to understand statistical analysis tools. • R, SAS, SPSS
	Programming language	• Overall knowledge and experience in programming are needed. • Understanding of various languages such as C, C, Java, Ruby, and Perl.
	RDBMS technology	• Ability to design keys, indexes, queries, normalization, and constraints based on SQL.
	Distributed computing	• Hadoop (MapReduce, HDFS), NoSQL(Cassandra, BigTable, MongoDB)
	Mathematical knowledge	• Understanding of matrix operation and numerical analysis.

03 NoSQL

A) Definition and characteristics of NoSQL

A NoSQL database provides other means of processing than the tabular relations used in relational databases. NoSQL is a non-relational distributed data repository that can be expanded horizontally, such as data replication and distributed storage on multiple servers focusing on the write speed for processing unstructured and ultra-high-capacity data.

<Table 75> Characteristics of NoSQL

Characteristic	Description
Processing of large-scale data	- Provides a loose data structure that allows data to be processed at the petabyte level.
Use of flexible schemas	- Saves data relatively freely without a predefined schema. - Saves (data) in a simplified form such as the key value, graph, and document structure.
Inexpensive cluster configuration	- Supports scale-out, data replication, and distributed storage using multiple servers composed of Pc-level commercial hardware.
Simple CLI (Call Level Interface)	- No query language like SQL in existing relational databases is provided. - A simple interface is provided by calling a simple API or HTTP.
High availability	- NoSQL loads data by automatically dividing data items into the cluster environment.
Allows as much integrity as needed	- While the relational DBMS focuses on ensuring logical structure and ACID, NoSQL makes the application process some of the integrity works instead of assigning them all to the DBMS.
Schema-less	The methods of saving data are largely divided into column, value, document, and chart, using a function that allows data storage and access using the key values, without the fixed data schema for data modeling.
Elasticity	NoSQL has a structure that allows expansion of the system's scale and performance and distribution of the I/O load more easily, so that large-scale data can be created, updated, and queried, while not causing downtime for any clients and application systems that access the system, even if the system fails partially.
Query	NoSQL provides query language, related processing technology, and API that can efficiently search and process data according to the characteristics of data even in a system composed of tens or thousands of servers.
Caching	NoSQL has a structure in which memory-based caching technology is very important, and which can provide a high-performance response speed even for large-scale queries and be consistently applied to development and operation.
High scalability	Partitioning allows a gradual node increase.
High availability	There is no single point of failure, and data are available even though a certain node is down because they are replicated.
High performance	The result should be quickly returned based on memory instead of disk, which can be achieved by using the non-blocking write and low complexity algorithm.
Atomicity	Each write operation needs to be atomic.
Consistency	Strong consistency is not needed, but the resulting consistency is sufficient (Read-Your-Writes).
Persistence	Data should be kept on a disk, not just in a volatile memory only.
Deployment flexibility	When a node is added or deleted, data should be automatically loaded without the need for data distribution or manual mediation, and there should be no constraints, such as distributed file system or shared storage, or any need for special hardware. Hardware should be operable in heterogeneous hardware.
Modeling flexibility	Data of various types such as key-value pairs, hierarchical data, and graphs should be modeled conveniently.
Query flexibility	Multiple GET that obtains a set of values for the provided key from a query, and queries that obtain data based on a specific range of keys, are needed.

B) BASE attributes of NoSQL

① BASE attributes of NoSQL

<Table 76> Description of NoSQL's BASE properties

Item	Contents
Basically Available	- Emphasis is placed on availability and the use of optimistic locking and queue. - Availability is ensured even with multiple failures and copies are stored in multiple storages.
Soft-State	- Node status is determined by the information transmitted from outside. - Updates between distributed nodes are updated when data reach the node.
Eventually Consistent	- The property of maintaining consistency optimally even though consistency is lost temporarily.

② Comparison of BASE attributes and ACID attributes

<Table 77> Comparison of BASE attributes and ACID attributes

Attribute	BASE	ACID
Application field	- NoSQL	- RDBMS
Scope	- Characteristics of the entire system.	- Limited to transactions.
Consistency aspect	- Weak consistency.	- Strong consistency.
Main points	- Focused on "availability".	- Focused on "commit".
System aspect	- Focused on "performance".	- Focused on "strict data management".
Efficiency	- "Query design" is important.	- "Table design" is important.

C) Storage method of NoSQL

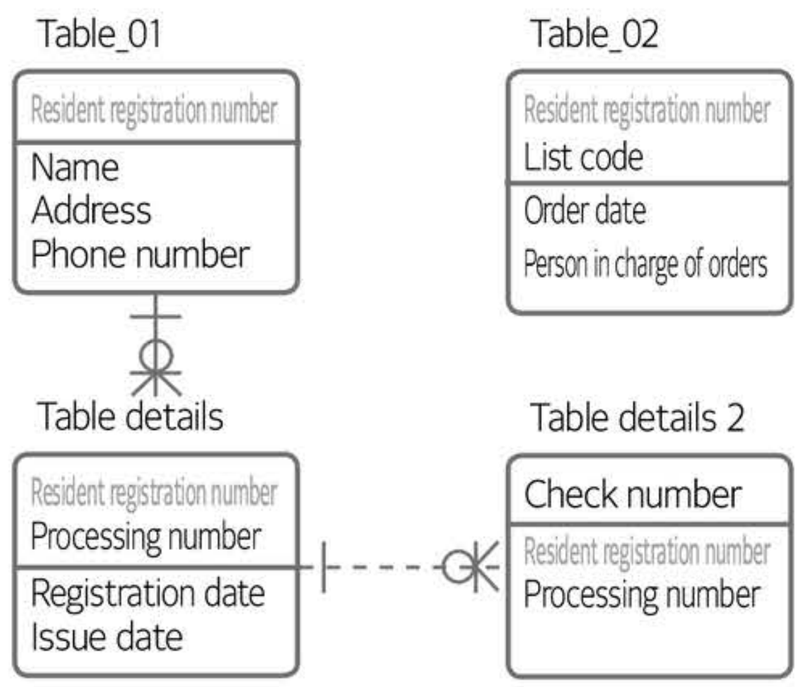
NoSQL databases can be divided as follows from the viewpoint of the data model that stores data.

<Table 78> Types of NoSQL

Type	Description
Key-value based	• The most basic NoSQL database that provides simple and fast Get, Put, and Delete functions based on the key-value. • Dynamo, Redis, MemcacheDB, etc.
Column family based	• An NoSQL database that stores data in rows in the column family, which corresponds to the table in the relational database. • Cassandra, HBase, SimpleDB, etc.
Document based	• An NoSQL database that stores documents such as XML, JSON, BSON, etc. in the value part of the key-value database, such as NoSQL DB, MongoDB, CouchDB, etc
Graph based	• An NoSQL database that expresses the entry attribute of the relational database as a node and the relationship as the edge between nodes, such as Neo4J, Flock DB, etc.

D) Characteristics of the NoSQL data model

<Table 79> Characteristics of NoSQL data modeling compared to relational DB data modeling

Type	Relational DB data modeling	NoSQL data modeling
Conceptual diagram		User 123 | UserInfo: Name, Email | Likes: 111, 222, … | Jay, jp@ebay.com, iphone, ipad ⋮ Item 111 | UserInfo: Title, Desc | LikedBy: 123, 4556, … | iphone, it's a phone, jay, john ⋮
Key point	• ACID-based data modeling	• BASE-based data modeling
Type	• Secures data consistency using minimal redundancy.	• Enables fast performance through data replication.
Phase	• Development executed after data modeling (design) according to business characteristics.	• The data set is designed in consideration of the screen and development logic.
Independence	• Independent program design (data independence is important).	• Dependent program design (avoids data independence).
Characte-ristics	• A data modeling technique that derives logical connection points using the data composition method with generalized notation and execution procedures.	• Data modeling is closer to file structural design, rather than to general data modeling in which a data set that can be easily processed by the program is created.

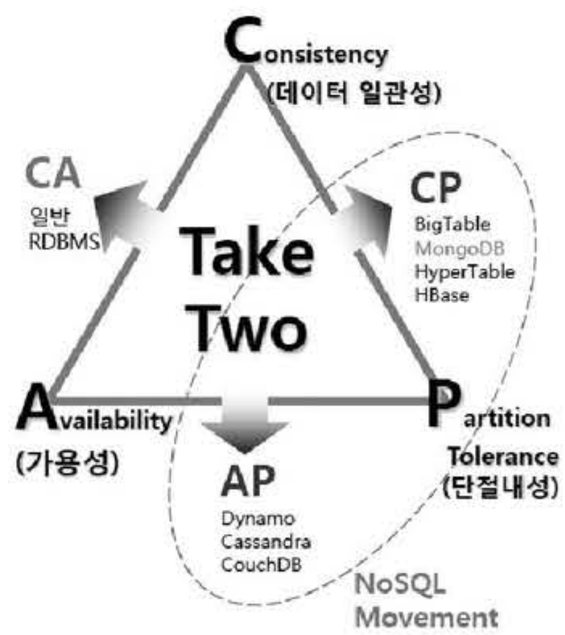
E) Characteristics of the NoSQL data model

① Definition of the CAP theorem

- This theory asserts that as it is impossible for a distributed data store to simultaneously satisfy all of data consistency, availability, and partition tolerance, only two should be strategically selected.

② Composition and management strategy of the CAP theorem

<Table 80> Composition and management strategy of the CAP theorem

Characteristics	Description
Configuration diagram	

Consistency	• All nodes should show the same data at the same time. (Each user should always view the same data.)
Availability	• Even if some nodes are down, it should not affect the other nodes. (All users should always be able to read and write data.)
Partition tolerance	• Even if some messages are lost, the system should operate normally. (The system should work properly in a physically distributed network environment.)
C + A	• An exceptionally reliable type in which message loss can be prevented even if the system is down. • Essential when a transaction is required. • General RDBMS.
C + P	• A performance type in which all nodes must perform well together. • Google's BigTable, HyperTable, HBase.
A + P	• Essential for asynchronous store operations. • Dynamo, Apache Cassandra, CouchDB, Oracle Coherence

• NoSQL systems are composed of distributed systems that have the characteristics of A+P and C+P.

XV. Understanding Artificial Intelligence

Recent trends and major issues

Artificial intelligence (AI) refers to a technology that studies how computers can think, learn, and develop by themselves like a human intelligence. Previously, AI imitated human beings as a computer technology, but it now is being applied to most industrial sectors including the military, aviation, medical, manufacturing, communication, public administration, and life services fields. We are now living in an era in which the ethical and institutional issues of AI must be considered, not to mention the technical issues.

Learning objectives

1. To be able to explain the concepts of AI and machine learning.
2. To be able to explain the concept and algorithm types of deep learning.

Keywords

- Artificial intelligence, Turing test, Asilomar principles, machine learning, supervised learning, unsupervised learning, reinforcement learning, deep learning, DNN, RNN, RBN, DBN

✚ Preview for practical business Why AI must understand both technology and ethics

Although interest in data analysis has increased, the existing system architecture and DBMS face limitations in terms of processing speed and performance when an existing analysis system processes tens of PBs of unstructured data generated by multimedia, SNS, sensors, IoT, etc., due to the advancement of IT technologies. As a result, solutions are being developed that are suitable for analyzing the large volume of unstructured data (variety) generated at high speed (velocity). Technologies related to big data, which was still in its infancy only a few years ago, have developed rapidly and are now being used directly in our real life.

The AI field needs to learn the Asilomar principles which are created by considering not only the understanding of a given technology but also the potential side effects of applying that technology.

01 Overview of AI

A) Definition and classification of AI

① Definition of AI

AI is a technology that realizes human learning ability, reasoning ability, perception ability, and natural language understanding ability by using computer programs. The dictionary meaning of AI is based on a philosophical concept, which is created by humans or intelligent beings or systems. In the past, AI had the narrow meaning of "software that enables computers to imitate human intelligence behaviors". More recently, however, the meaning of AI has come to include a technology that studies how to enable computers to perform the kinds of thinking, learning, and self-development that can be done with human intelligence.

② Classification of AI

<Table 81> Classification of AI

Three elements	Description	Case
Weak AI/Artificial Narrow Intelligence: ANI	Operable only under given conditions.	Recommendation by Google Maps, Autonomous Car, Google Translation, and Facebook
Strong AI/Artificial General Intelligence: AGI	AI that can think like humans.	Terminator, secretary robot, factory robot, etc.
Artificial Super Intelligence: ASI	AI that surpasses humans in all areas.	A high-level command is possible, such as "Create a new energy source that can be used by people for the next 1,000 years."

B) History of AI

<Table 82> History of AI

Item	Description	Detailed technology
1950	The beginnings of AI	Turing defined the method of testing whether a machine can think, the possibility of developing an intelligent machine, and a learning machine, etc. in his paper "Computing Machinery and Intelligence".
1956	Composition of a data-based analysis system	Ten scientists gathered at Dartmouth College, UK and held a 6-week workshop, which became the beginnings of AI study.
1970's	AI Winter (Dark Age)	AI Winter began as funding was cut due to the perception that AI systems would never meet the original expectations.
1980's	Expert system	A rule-based system created by manually entering the knowledge of human experts. Although hundreds of systems were built, small-sized systems were not useful, and large-sized systems were expensive to develop and maintain, and difficult to use. Second "AI Winter" in the late 1980s.
1990's	Imitation of nature Neural network, genetic algorithm	A new technology appeared that focused on the manipulation of high-level symbols by replacing the traditional "logicist" paradigm. Optimism about AI

2010's	Present and future of AI Machine learning, deep learning	Computers design by creating their own learning. They reuse data to be learned for other problems. Big data that processes a large amount of data is spread, and hardware performance is improved.

C) Distinguishing AI

① Definition of the Turing test

- A test of a machine's ability to exhibit intelligent behavior equivalent to, or indistinguishable from, that of a human.
- In a given problem-solving thinking situation, if the evaluator cannot reliably tell the machine from the human, the machine is said to have thought in the same way as a human being.

② Principles of the Turing test

- The Turing test is a kind of game designed to test whether a machine can have intelligence.
- In the 'Imitation Game', people and machines are isolated in different rooms and communicate by teletype.
- The people try to confirm that the other party is a machine by asking it questions, and the machine tries to lead the people to believe that it is a real human through its answers.
- If the machine wins the game, the machine can be said to have AI.

D) Machine learning

① Definition of machine learning

- A program that can solve problems in a new situation while learning new knowledge.
- Science and technology study a system that improves its performance by itself using empirical data based on interactions with the environment.

② Classification of machine learning

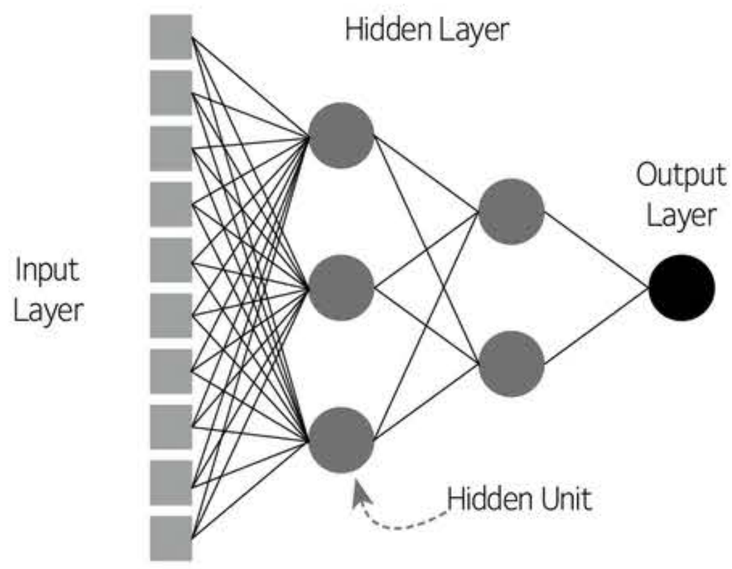
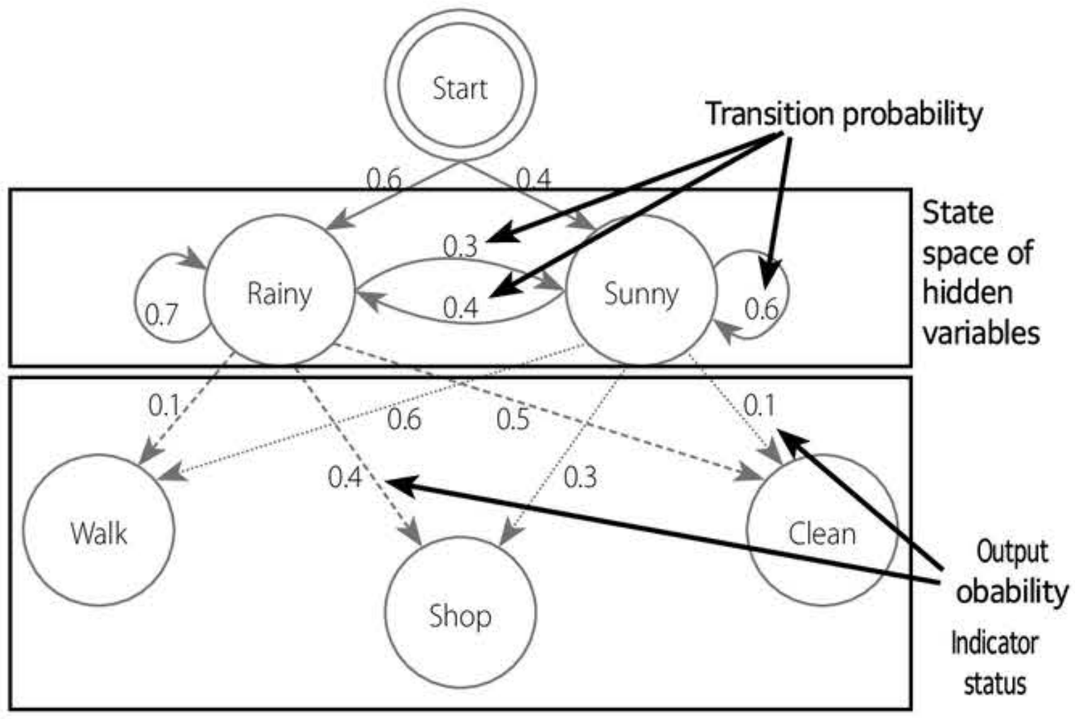
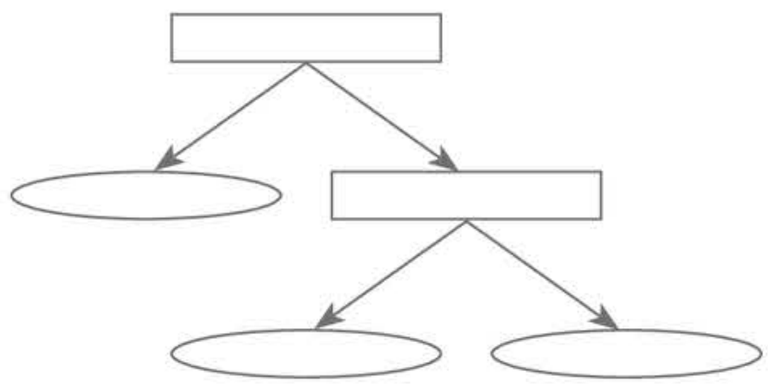
<Table 83> Supervised learning, unsupervised learning, reinforcement learning

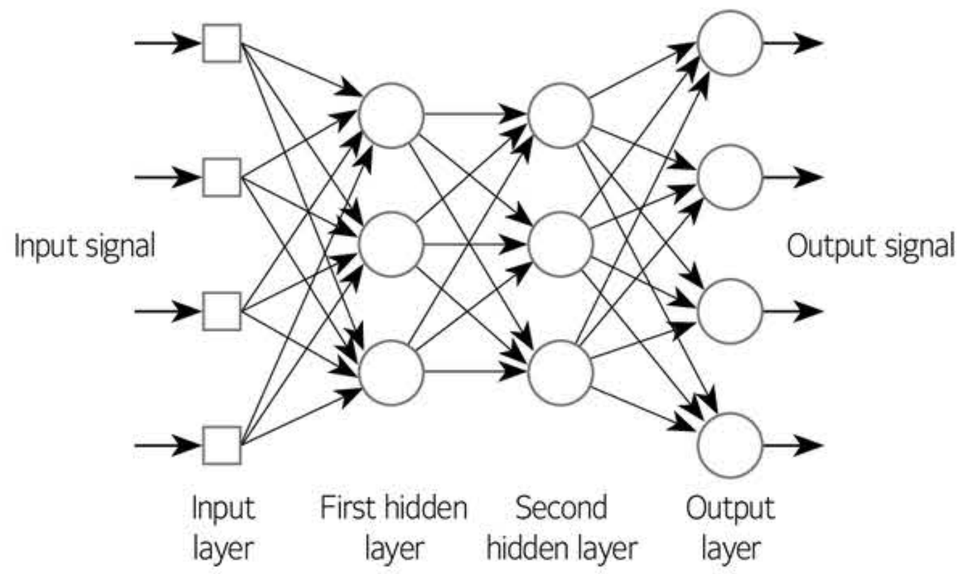
Classification	Description	Learning problem
Supervised learning	A machine learning method that uses learning data, which describes the intended result. Learning a function that maps the input from learning examples, which are composed of a pair of I/O, and the output. A learning method that presents input x and its target output y.	Classification, regression, recognition, diagnosis, prediction Neural network model, SVM
Unsupervised learning	A machine learning method that uses learning data, which does not describe the intended result. Applicable to the clustering problem because learning data can be built easily even though performance is poor. For example, input is given, but there is no corresponding output. The purpose of learning is to understand the common characteristics of input patterns. A learning method in which only input x is given and target output y for it is not presented.	Clustering, density function estimation, dimensional reduction, feature extraction, K-means

Reinforcement learning	A method of learning in which learners increases their own knowledge through the experience of being praised for good behavior and punished for wrongdoing. An intermediate form between supervised and unsupervised learning. A learner selects behavior for the input and learns from the reward that the teacher provides for the action. A method that informs output evaluation value c only, instead of input x and target output y for input x.	Trial and error, reward function, dynamic programming, Q-Learning, DQN

③ Supervised learning

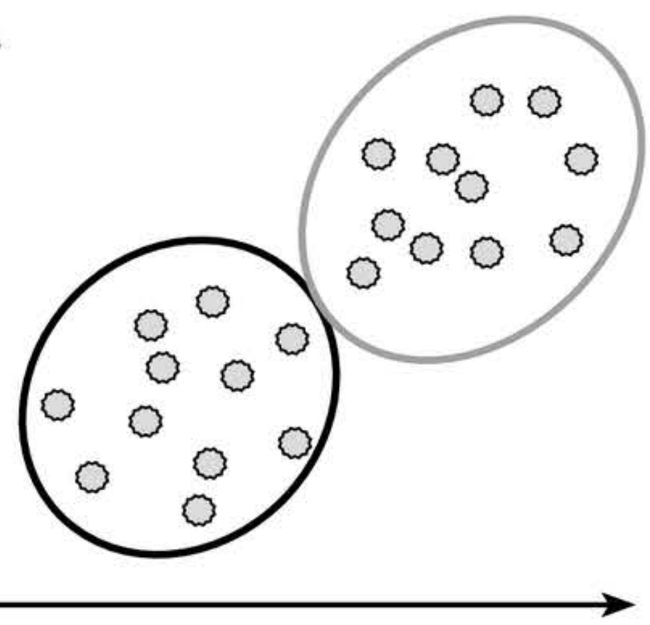
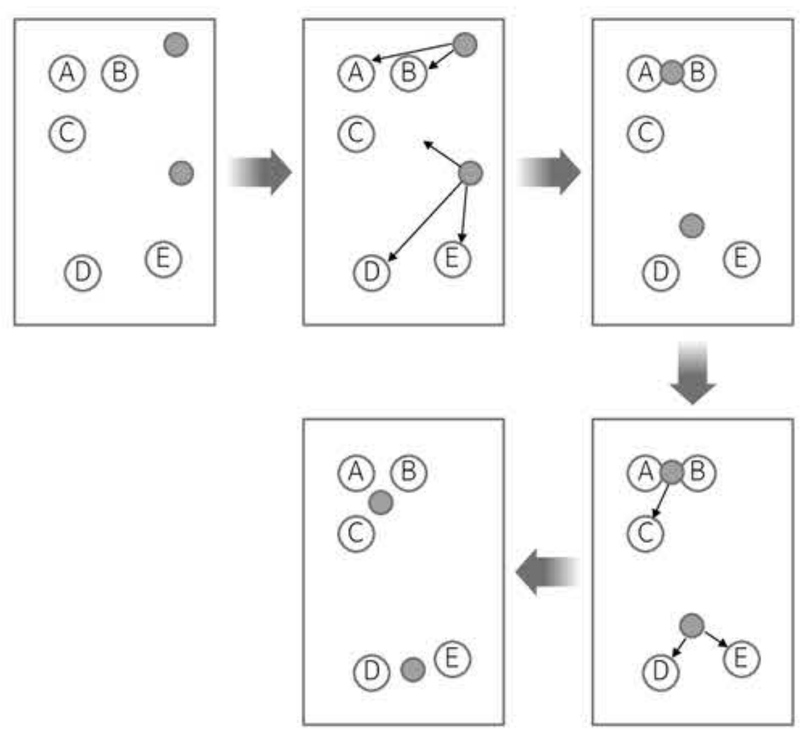
<Table 84> Supervised Study Table

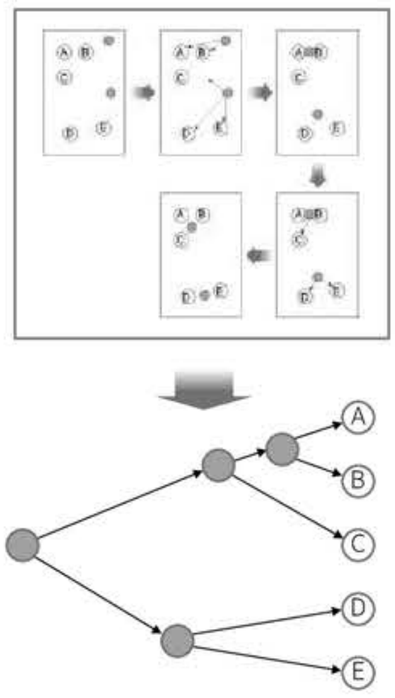
Classification	Description	Case of learning problem
Neural network	 Structure of the neural network model	An algorithm (neural network) that processes diverse information in a way that is similar to the brain, using the human brain as a model.
Hidden Markov Model (HMM)		A model that is made of a multiplication of transition probability and emission probability. - Directed graph, sequential data modeling, probabilistic model, generative model.
Decision tree		Tree structure, sequential division of data space, entropy-based information acquisition, convertible to a rule.

Multi-Layer Perceptron (MLP)	 [figure6-8] Multi-Layer Perceptron with two hidden layers	Layer structure, sigmoid neuron, error correction, strong against noise, discrete/consecutive I/O mapping.

④ Unsupervised learning

<Table 85> Unsupervised Study Table

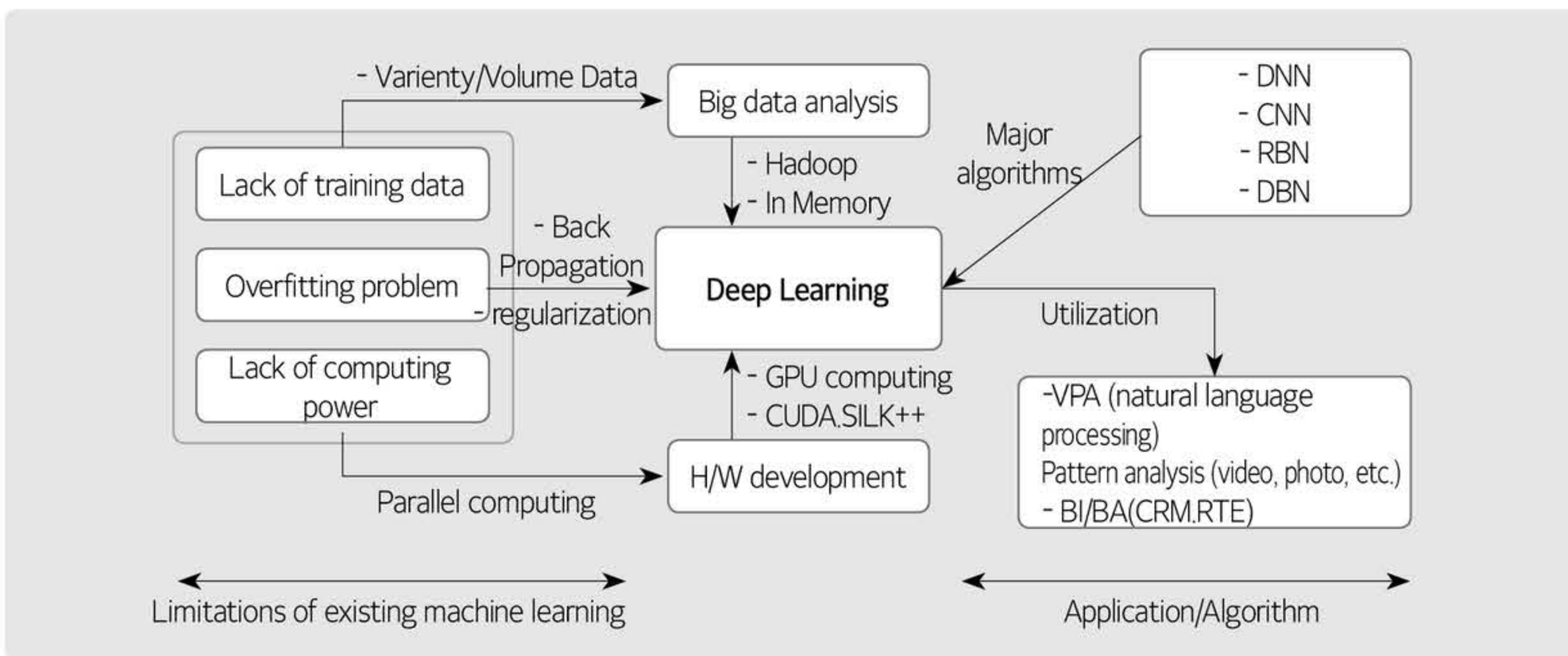
Classification	Conceptual diagram	Description
Clustering		Clustering is a technique for finding a group of closely related data. A data set is classified into a subset or a cluster.
K-means		A clustering algorithm that classifies the data of similar groups by measuring the distance of the data from the center of each cluster.

Hierarchical Clustering (HC)	 Flow of hierarchical clustering execution Hierarchical drawing of hierarchical clustering results	HC divides all data into a hierarchical structure. It is useful for the visualization of multidimensional data.

E) Deep learning

① Background and definition of deep learning

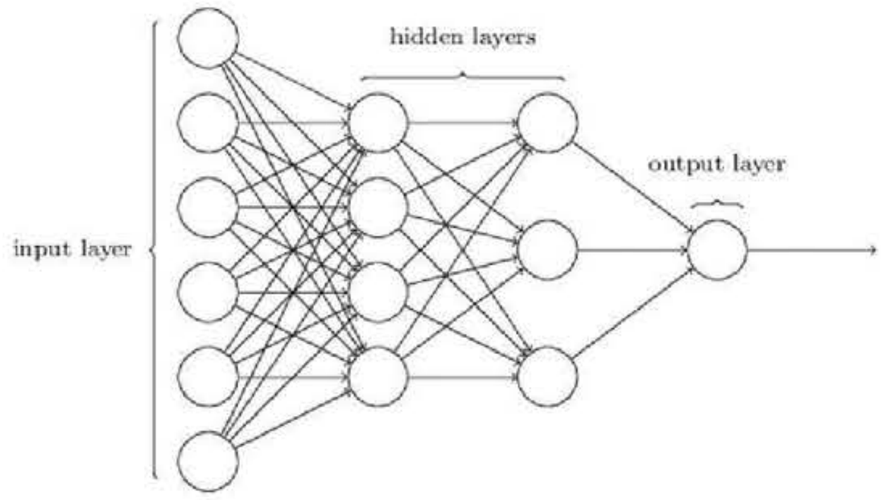
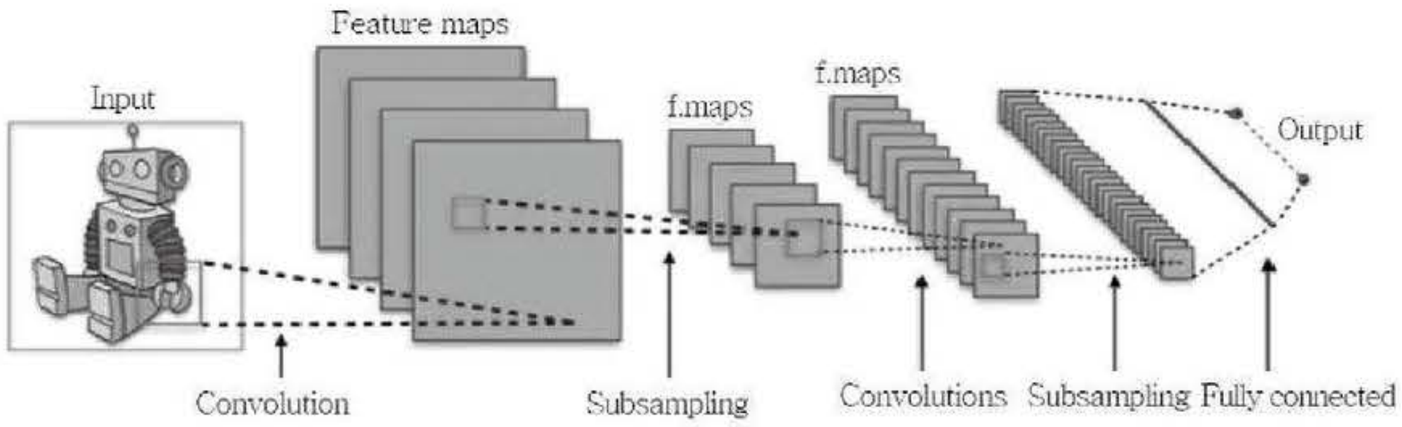
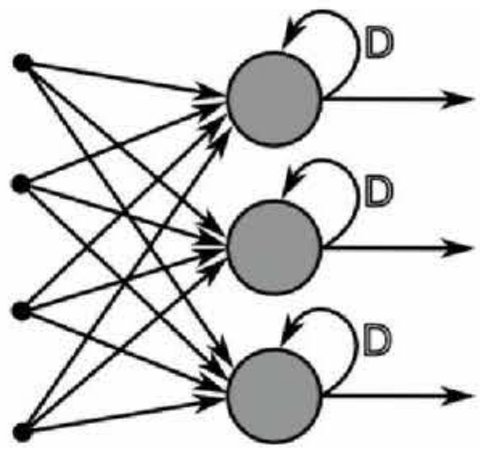
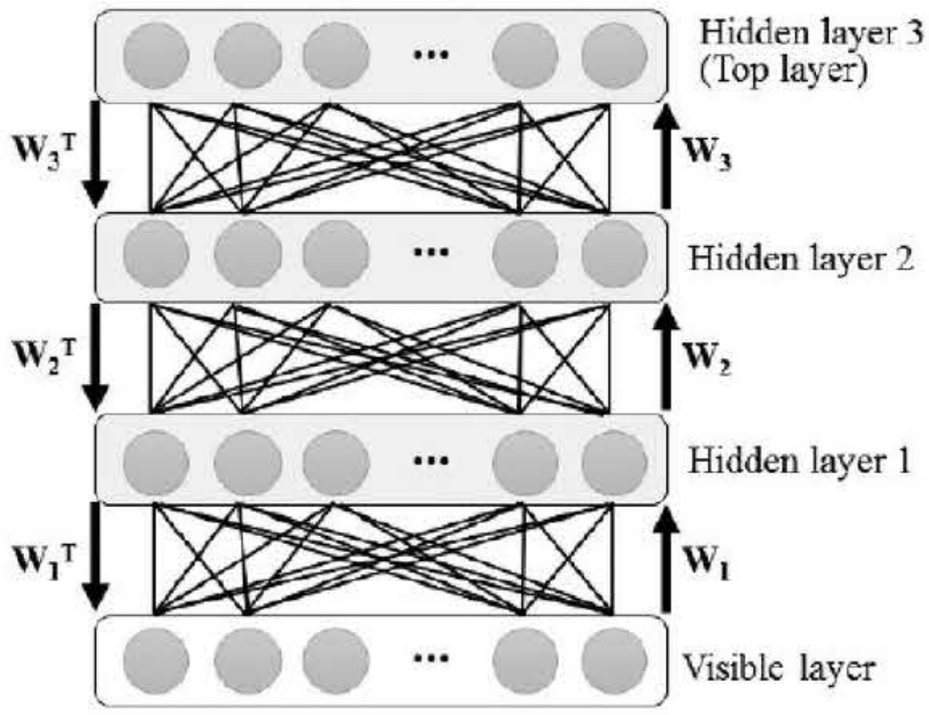
- AI technology that allows computers to learn on their own, like humans, by combining existing supervised learning, which requires human intervention, with more active unsupervised learning.
- A series of machine learning based on the deep neural networks (DNN) theory and an algorithm that teaches computers how to think like a human.



[Figure 46] Deep Learning Background

② Deep learning algorithm

<Table 86> DNN, CNN, RNN, RBM, DBN

Characteristics	Description
DNN	• An artificial neural network that is composed of multiple hidden layers between the input layer and the output layer in order to build a complex and expressive model by overcoming the limitations of ANN. • An artificial neural network structure composed of multiple hidden layers. 
CNN	• Convolutional Neural Networks • A neural network algorithm with a structure suitable for image data processing, using the mechanical principle of the organism's optic nerve. • It is composed of the convolution layer > pooling layer > convolution layer > pooling layer > feed forward layer. 
RNN	• Recurrent Neural Network • A neural network in which the connections between units, which constitute an artificial neural network, compose a directed cycle. • An algorithm that creates a relationship which produces multiple values when one value is entered. • It connects the past neural network (T-1) and the current neural network (T), and learns using an algorithm called BPTT (Back-Propagation Through Time), which is an extension of error back-propagation, a learning algorithm of the existing artificial neural network. 

Characteristics	Description
DBN	• A generative graphical model used for machine learning. • A deep neural network that is composed of the multiple layers of the latent variable. • A deep learning technique that stacks the Restricted Boltzmann Machine (RBM), a type of Boltzmann machine, for learning. • A structure that learns by stacking RMB, which has a separate visible layer and hidden layer.