

응용 SW 기초 기술 활용

1. 네트워크 기초 활용하기

1 - 1 네트워크 프로토콜 활용

① 네트워크 계층 구조

1. OSI 7 계층(Open System Interconnection 7 Layer)

(1) OSI 7 계층의 개념

국제표준기구(ISO: International Organization for Standardization)이 정리한 네트워크 구조 기본 모델

(2) OSI 7 계층의 필요성

데이터 통신을 위해서는 데이터를 전기 신호로 변환하고 복잡한 네트워크 경로를 통해 데이터를 전달해야 하며 송신지와 수신지의 컴퓨터 구조가 상이할 경우 전송한 데이터가 올바르게 인식되지 않을 수도 있어 복잡한 변환 과정을 거쳐야 한다. 이런 복잡한 구성을 유사한 기능별로 계층화하여 처리되는 정보들을 캡슐화하여 구성하고 각 계층에 사용되는 통신 규격을 프로토콜로 표준화함으로써 응용소프트웨어 개발자, 서버/네트워크 엔지니어들이 본인이 연관된 계층만 고려해 업무를 수행할 수 있도록 한다.

* OSI 7 Layer의 응용소프트웨어 개발 활용

응용소프트웨어 개발자는 주로 OSI 7 Layer의 상위 계층(5~7)을 다루고 서버/네트워크 엔지니어는 하위 계층(1~4)을 다룬다. 응용소프트웨어 개발자는 4계층에 지정된 포트 번호를 통해 상위 계층의 프로토콜을 구별할 수 있기 때문에 응용소프트웨어 개발을 위해서는 4계층의 헤더 정보 분석에도 관심을 가지고 학습하도록 한다.

(3) OSI 7 계층의 구성

데이터를 목적지에 정확히 전달하는 것에 초점 - 하위 계층
송수신 데이터의 가공 및 활용에 초점 - 상위 계층

〈표 1-1〉 OSI 7 Layer 구성

구분	계층	역할 및 기능	주요 장비
Application Layer (상위 계층)	7. Application Layer	- 사용자에게 서비스 제공 - 사용자 입출력 정의 - 응용 프로세스 관리	L7 스위치, 웹 방화벽
	6. Presentation Layer	- 송수신자의 다른 데이터 표현 방식을 상호 인식 가능하도록 변환 - 인코딩, 암호화, 압축, 코드 변환	
	5. Session Layer	- 송수신자의 프로세스 간 연결 관리 - 통신 오류 복구 및 재전송	
Data Flow Layer (하위 계층)	4. Transport Layer	- 세그먼트 구성 - 송수신자의 포트(Port) 지정 - 메시지 분할 및 재조립 - 프로세스 간 혼잡제어, 흐름제어 - 오류제어 및 재전송	로드밸런서, 방화벽
	3. Network Layer	- 패킷 구성 - 송수신자의 논리 주소(IP) 지정 - 최적 경로 탐색 및 전송	라우터, L3 스위치
	2. Data Link Layer	- 프레임 구성 - 오류제어, 흐름제어, 접근제어 - 송수신자의 물리주소(MAC) 지정	NIC, L2 스위치
	1. Physical Layer	- 비트 스트림의 전기 신호 전송 - 비트의 부호화 및 복호화 - 물리적 연결 설정 및 해제	허브, 리피터, 케이블

2. TCP/IP(Transmission Control Protocol/Internet Protocol) 프로토콜 스택

(1) TCP/IP 프로토콜 스택의 개념

OSI 7 Layer를 실무에 활용하는 기능 중심으로 4계층으로 구조화하고, 각 그룹에서 활용되는 프로토콜군을 정리한 네트워크 통신 구조 모델이다.

(2) TCP/IP 프로토콜 스택의 구성

OSI 7 Layer의 1, 2계층과 5, 6, 7계층을 통합하여 4계층으로 구성된다

〈표 1-2〉 TCP/IP 프로토콜 스택 구성

OSI 7 Layer	TCP/IP	프로토콜
7. Application	Application	- HTTP(Hypertext Transfer Protocol): 인터넷 브라우저용 - FTP(File Transfer Protocol): 파일 송수신용
6. Presentation		- IMAP(Internet Messaging Access Protocol): 메일 송수신용 - SMTP(Simple Mail Transfer Protocol): 메일 송수신용
5. Session		- TELNET(Telecommunication Network): 터미널 접속용
4. Transport	Transport	- TCP(Transmission Control Protocol): 패킷 송수신용 - UDP(User Datagram Protocol): 패킷 단방향 송신용
3. Network	Internet	- ARP(Address Resolution Protocol): IP의 MAC 주소 변환용 - ICMP(Internet Control Message Protocol): NW 제어용
2. Data Link	Network	- RS-232(Recommended Standard-232): 직렬 포트용 - V.35: 케이블 랜선용
1. Physical	Access	- FDDI(Fiber Distributed Data Interface): 광섬유 케이블용

② 인캡슐레이션(Encapsulation)과 디캡슐레이션(Decapsulation)

네트워크는 다수의 기기가 연결되어 통신하기 때문에 하나의 기기가 네트워크를 단독으로 점유하지 않고 다수의 기기가 공유하여 동시에 사용할 수 있어야 한다. 이를 위해 송신지에서는 인캡슐레이션 과정을 통해 전송 데이터를 패킷으로 분할하고, 각 패킷에 송수신지의 IP/Port, 전송 순번 등을 같이 보내고 수신지에서 디캡슐레이션 과정을 통해 분할되어 도착된 패킷들을 재 조합하여 활용한다.

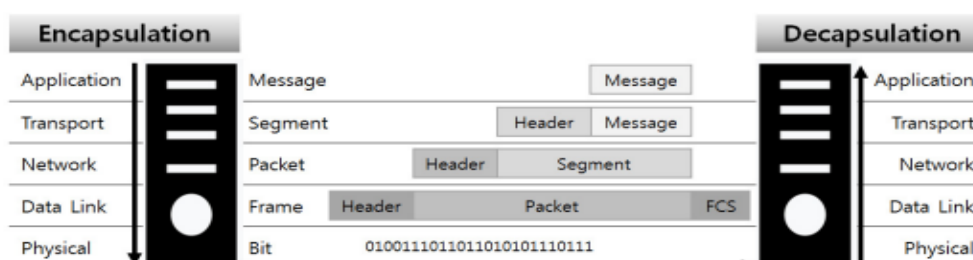
1. 인캡슐레이션과 디캡슐레이션의 개념

(1) 인캡슐레이션의 개념

송신지 Application Layer에서 발생한 데이터를 하위 계층으로 이동시키면서 각 계층에서 처리한 결과를 캡슐화하는 과정

(2) 디캡슐레이션의 개념

수신지의 하위 계층에서 인식한 데이터를 상위 계층으로 이동시키면서 각 네트워크 계층에서 처리 가능한 형태로 디캡슐화하는 과정



2. 인캡슐레이션과 디캡슐레이션의 처리 방법

(1) 송신지에서의 인캡슐레이션 처리

- (가) Transport 계층에서는 상위 Application 계층에서 생성된 메시지를 전송이 용이하도록 분할하여 Body 정보에 추가하고 수신지에서 분할되어 수신된 메시지를 재조립할 수 있도록 전송 순번, 출발지와 도착지의 Port 정보 등을 Header 정보에 추가하여 Segment를 생성한다.
- (나) Network 계층에서는 상위 Transport 계층에서 생성한 Segment를 Body 정보에 추가하고 출발지와 도착지의 IP 정보 등을 Header에 추가하여 Packet을 생성한다.
- (다) Data Link 계층에서는 상위 Network 계층에서 생성한 Packet을 Body 정보에 추가하고 출발지와 도착지의 MAC 정보 등을 Header에 추가하고 수신지에서 수신된 데이터가 정상인지 판단할 수 있도록 FCS(Frame Check Sequence)를 마지막에 추가하여 Frame을 생성한다.
- (라) Physical 계층에서는 상위 Data Link 계층에서 생성한 Frame을 물리적인 전기신호로 부호화하여 수신지에 전송한다.

(2) 수신지에서의 디캡슐레이션 처리

- (가) Physical 계층에서 전기 신호를 비트로 복호화하고 Network 계층에 전달한다.
- (나) Data Link 계층에서 헤더 정보를 체크하여 목적지의 MAC 주소가 자신의 MAC주소와 동일하면 이더 타입(Ether Type)에 정의된 Network계층의 프로토콜로 헤더를 제거한 Packet을 전달한다.
- (다) Network 계층에서 헤더 정보를 체크하여 목적지의 IP 주소가 자신의 IP 주소와 동일하면 프로토콜 번호(Protocol Number)에 정의되어 있는 Transport 계층의 프로토콜로 헤더를 제외한 Segment를 전달한다.
- (라) Transport계층에서 헤더 정보를 체크하여 도착한 Segment들을 재조립하여 Message를 생성하고 포트번호(Port Number)에 정의된 Application계층의 프로토콜로 헤더를 제외한 Message를 전달.

* PDU(Protocol Data Unit)

PDU는 각 네트워크 계층에서 사용하는 데이터 단위를 의미하는 용어로 통신에 필요한 헤더 정보(제어 정보)와 데이터를 캡슐화하여 구성하며 Frame, Packet, Segment 등 계층별로 다른 단위를 사용한다. 예를 들어 4계층에서는 Segment단위를 사용하며 송신지와 수신지의 4계층에서는 Segment를 이용해 통신을 수행하게 된다

③ 네트워크 계층별 헤더 정보

송신지와 수신지의 각 계층에서는 독립적으로 통신을 수행하기 위해 PDU의 헤더 정보에 계층별 주요한 제어 정보를 포함한다.

〈표 1-3〉 네트워크 계층별 헤더 정보

OSI 7 Layer	헤더 정보
4. Transport	- Sort Port, Destination Port
	- Sequence Number, Acknowledgement Number
	- Data Offset, Res, Flags, Window Size, Header and Data Checksum
	- Urgent Pointer, Options
3. Network	- Source IP, Destination IP, Protocol
	- Version, IHL, DSCP, ECN, Total Length, Identification
	- Flags, Fragment Offset, Time to Live, Header Checksum, Options
2. Data Link	- Destination MAC, Source MAC, Ether Type

〈표 1-4〉 네트워크 상위 프로토콜 지시자

OSI 7 Layer	지시자	주요 상위 프로토콜 값
4. Transport	Port Number	- TCP 20, 21: FTP(File Transfer Protocol) - TCP 22: SSH(Secure Shell) - TCP 23: TELNET(Telnet Terminal) - TCP 80, UDP 80: HTTP(Hypertext Transfer Protocol) - TCP 443: HTTPS(Hypertext Transfer Protocol Secure)
3. Network	Protocol Number	- 1: ICMP(Internet Control Message) - 2: IGMP(Internet Group Management) - 6: TCP(Transmission Control) - 17: UDP(User Datagram)
2. Data Link	Ether Type	- 0x0800: IPv4(Internet Protocol Version 4) - 0x86DD: IPv6(Internet Protocol Version 6) - 0x8035: RARP(Reverse ARP)

① 라우팅 유형

데이터가 송신지에서 수신지까지 이동하는 경로를 형성하는 것을 라우팅이라고 하며, 크게 정적 라우팅과 동적 라우팅으로 분류한다.

1. 정적 라우팅(Static Routing)

네트워크 관리자가 직접 라우팅 테이블에 라우팅 경로를 입력하여 관리하는 방법을 정적 라우팅.

2. 동적 라우팅(Dynamic Routing)

네트워크 관리자가 직접 개입하지 않고 라우터 간에 정보를 교환하면서 라우팅 경로를 관리하는 방법을 동적 라우팅. 동적 라우팅은 거리 벡터 라우팅 알고리즘과 링크 상태 라우팅 알고리즘을 사용한다.

(1) 거리 벡터 라우팅 알고리즘(Distance Vector Routing Algorithm)

인접 라우터와 라우팅 테이블 정보를 교환하여 패킷 전송을 위해 거쳐야 하는 라우터의 개수로 거리를 산정하여 최적의 라우팅 경로를 수립하는 방법으로 RIP 등의 프로토콜을 이용해 라우터 간 정보를 교환한다

(가) RIP(Routing Information Protocol)

라우팅 테이블에 동일 네트워크에 포함된 각 라우터에 도달하기 위해 거쳐야 하는 라우터들의 최대 수와 각 라우터에 도달하기 위해 이동해야 하는 다음 라우터 정보를 관리하고, 정기적으로 라우팅 테이블 정보를 인접 라우터와 교환하여 자신의 라우팅 테이블을 갱신하는 방법이다

(2) 링크 상태 라우팅 알고리즘(Link State Routing Algorithm)

연결된 모든 라우터로부터 연결 상태 정보를 수신하여 각 라우터까지 최단 경로를 라우팅 테이블로 만드는 방법으로, OSPF 등의 프로토콜을 이용해 라우터 간 정보를 교환한다.

(나) OSPF(Open Shortest Path First))

라우터의 연결 상태가 변경된 경우 동일 네트워크에 포함된 모든 라우터에 자신의 변경 정보를 전달하고, 정보를 수신한 라우터들이 각 라우터에 접근하기 위한 최적의 네트워크 경로를 갱신하는 방법이다

2. 미들웨어 기초 활용하기

2 - 1 미들웨어 파악

① 미들웨어(Middleware)

1. 미들웨어 개념

하나의 시스템에서 다양한 목적의 응용소프트웨어가 동시 수행되거나 복수 시스템의 응용소프트웨어가 서로 연계되어 수행되는 경우에도 안정적으로 실행될 수 있도록 운영체제와 응용소프트웨어 사이에서 다양한 기능을 지원하는 소프트웨어이다.

2. 미들웨어 주요 기능

〈표 2-1〉 미들웨어 주요 기능

주요기능	설명
분산 시스템 SW	- 물리적으로 분산되어 구축되어 있는 다수의 컴퓨팅 환경에서 사용자가 하나의 시스템처럼 사용할 수 있도록 구성된 소프트웨어
IT 자원 관리	- IT 자원에 대한 관리 정책을 기반으로 지속적으로 모니터링하고 성능과 가용성을 관리하는 기능을 제공하는 소프트웨어
서비스 플랫폼	- 서로 다른 서비스들을 하나의 통합 환경에서 인터랙티브하게 사용할 수 있도록 해주는 인터넷 기반 환경 구성 기술
네트워크 보안	- 네트워크에 연결된 호스트들의 송수신 정보 탈취 및 변조를 통한 불법적인 서비스 이용을 방지하는 기술

3. 미들웨어 주요 기능별 분류 체계

(1) 분산 시스템 SW

〈표 2-2〉 분산 시스템 SW 미들웨어 분류

분류	설명
웹 애플리케이션 서버	- Web Application Server - 웹 시스템에서 전달된 Request를 처리하기 위해 트랜잭션 관리, 세션 유지, 부하 분산 등의 역할을 하는 서버의 소프트웨어
연계 통합 솔루션	- 시스템 간 표준화된 데이터 송수신 처리를 통해 통합 환경 구성 지원 - EAI(Enterprise Application Integration, 각 애플리케이션에 어댑터를 설치하고 중앙 허브를 통해 연결하는 방식), ESB(Enterprise Service Bus, 시스템 간 연계 및 메시징 변환, 라우팅 등의 작업을 표준 기반 인터페이스를 통해 연결하는 방식) 등 솔루션 존재
실시간 데이터 처리	- 지속적으로 발생하는 데이터를 실시간 분석하고 반응하는 시스템 - CEP(Complex Event Processing, 실시간 발생하는 다양한 데이터를 조합하여 복잡한 상황에 대한 이벤트 또는 유형을 추론하는 방식) 등 존재
분산 병렬 처리	- 대규모 데이터를 실시간 처리, 분석하기 위해 다수의 노드에서 분할 처리 - DDS(Data Distribution Service, 각 시스템에서 교환 데이터 유형을 정의하면 해당 데이터 교환 대상 시스템 검색, 교환 등 처리 수행) 등 존재
TP 모니터	- Transaction Processing Monitor - 트랜잭션 처리 모니터링 및 제어 시스템

(2) IT 자원 관리

〈표 2-3〉 IT 자원 관리 미들웨어 분류

분류	설명
시스템 관리	<ul style="list-style-type: none"> - 서버의 리소스 및 프로세스 관리 시스템 - 하드웨어 자원 관리, 서버 가용성 모니터링 및 측정, SW 자원 관리, 사용 자별 권한 및 사용현황 관리, 서버 보안 관리 등 수행
SW 실행 관리	<ul style="list-style-type: none"> - 시스템 소프트웨어의 실행 상태 관리 시스템 - 실행 소프트웨어의 아키텍처 분석, 사용자 체감 성능 모니터링, 트랜잭션 자동 분석 및 보고서 생성 등 수행
네트워크 관리	<ul style="list-style-type: none"> - 네트워크 장비, 회선, 트래픽 등 관리 시스템 - 네트워크 장애 검출 및 대응, 네트워크 구성 관리, 네트워크 성능 측정 및 분석/보고, 네트워크 보안 예방/탐지/억제/복구 관리 등 수행
IT 서비스 운영 관리	<ul style="list-style-type: none"> - IT 시스템의 운영 과정 모니터링 및 관리 시스템 - 헬프 데스크(Help Desk), 구성 관리, 형상 관리, 문제 관리, 장애 관리, 이행 관리 등 수행

(3) 서비스 플랫폼

〈표 2-4〉 IT 자원 관리 미들웨어 분류

분류	설명
IoT 플랫폼	<ul style="list-style-type: none"> - IoT 장치를 연결하고 응용 서비스의 설치, 구동, 정지, 해제 등을 제어하고 관리하는 플랫폼
클라우드 서비스 플랫폼	<ul style="list-style-type: none"> - 클라우드 서버 기반 서비스 제공 플랫폼
UI/UX 프레임워크	<ul style="list-style-type: none"> - 사용자와 소프트웨어 간 소통 기능 구성을 위해 필요한 라이브러리 및 응용소프트웨어 집합
CDN	<ul style="list-style-type: none"> - Content Delivery Network - 다수의 노드에 콘텐츠를 복제 저장하여 사용자가 인접 노드에서 빠르게 콘텐츠를 받을 수 있도록 지원하는 시스템

(4) 네트워크 보안

〈표 2-5〉 네트워크 보안 미들웨어 분류

분류	설명
네트워크 접근 제어	<ul style="list-style-type: none"> - 인가된 사용자만 네트워크에 접근할 수 있도록 검사 및 차단 관리 기술
보안 통신	<ul style="list-style-type: none"> - 네트워크의 모든 통신 데이터를 암호화하여 데이터의 유출 및 변조를 방지하는 기술
침입 방지/사고 대응	<ul style="list-style-type: none"> - 네트워크의 비정상적 트래픽을 탐지하고 대응하는 기술 - SIEM(Security Information and Event Management) 솔루션 등
보안 관리	<ul style="list-style-type: none"> - 다양한 보안 정책을 통합 관리하고 적용 현황을 분석하여 보안위협을 탐지하고 관리하는 시스템

2 - 2 미들웨어 운용

① 전자정부 표준 프레임워크

1. 전자정부 표준 프레임워크의 개념

정부 부처, 지자체, 공공기관 등의 공공정보화사업에서 JAVA 기반 웹/모바일 시스템 구축 시 활용되는 개발 프레임워크로 한국지능정보사회진흥원(NIA)의 표준 프레임워크 센터에서 Apache 2.0 라이선스로 공개하고 있어 일반 기업 및 학습용으로도 활용 가능하다.

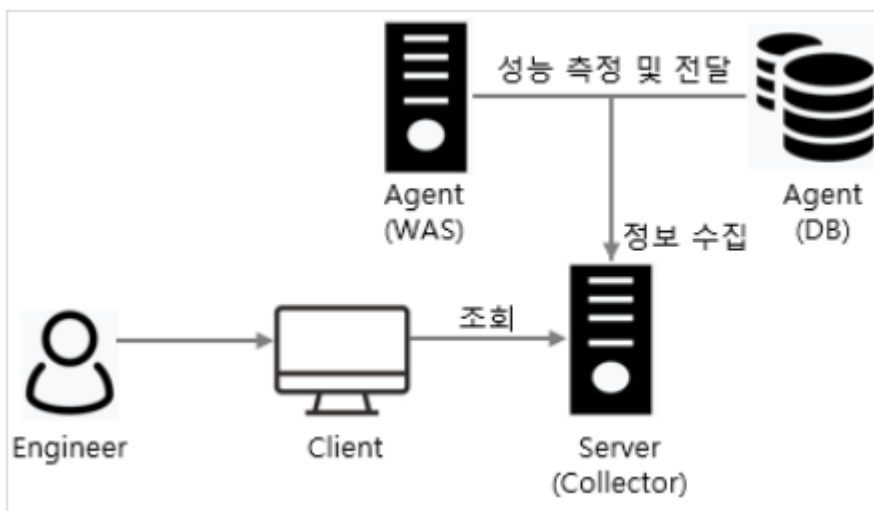
② Scouter

1. Scouter 개념

Scouter는 오픈소스 소프트웨어(Apache-2.0 License)로 공개된 애플리케이션 성능 관리(APM: Application Performance Management) 도구로서 응용소프트웨어 엔지니어가 분산 환경에서 안정적으로 시스템을 운영 및 관리하도록 지원하는 미들웨어이다. C/S(Client/Server) 형태가 일반적이지만 플러그인을 이용하면 Web 기반으로도 이용이 가능하며 오픈소스 소프트웨어이므로 학습용으로도 적합하다

2. Scouter 구성

Scouter는 Server, Client, Agent로 구성된다



출처: 집필진 제작(2021)
[그림 2-7] Scouter 구성도

〈표 2-11〉 Scouter 구성 요소

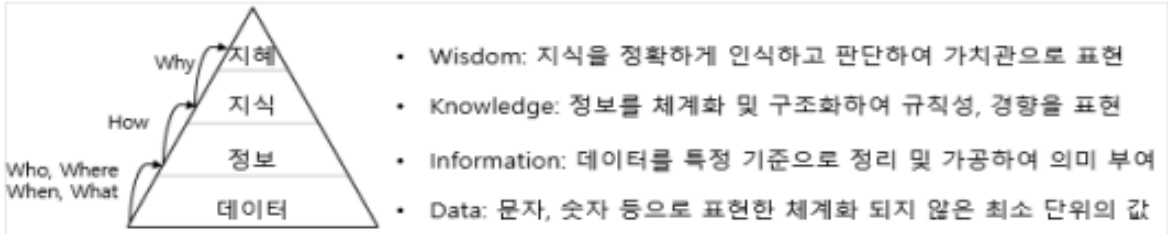
구분	설명
Agent	- WAS(Web Application Server), Database 등 모니터링 대상 시스템들에 설치하여 성능을 측정하고 그 결과를 Server로 전송
Server	- 각 Agent에서 측정된 결과 수집 및 저장
Client	- 서버 요청 및 응답 현황, 응답 평균속도, CPU 사용량, JVM 메모리 사용량, GC 시간 등 Server에 수집된 정보 확인

3 - 1 데이터베이스 특징 식별

① 데이터베이스(Database)

1. 데이터(Data)의 개념

데이터는 관찰이나 측정으로 수집한 사실(Fact)을 수치 또는 문자 형태로 표현한 최소 단위의 값이다. DIKW 피라미드와 같이 정보, 지식, 지혜로의 변환을 통한 창조(Creation)적 사고의 근간이 된다.



출처: 집필진 제작(2021)
[그림 3-1] DIKW 피라미드

2. 데이터베이스의 개념

데이터베이스는 공용으로 활용하기 위해 통합하여 저장한 운영 데이터의 집합이다.

〈표 3-1〉 데이터베이스 특성

특성	설명
실시간 접근성	- 요청받은 데이터 처리는 실시간으로 처리되고 결과를 반환해야 함
계속적 변화	- 저장된 데이터는 입력, 수정, 삭제에 의해 지속적으로 변화함
동시 공유	- 서로 다른 목적의 응용 SW 및 사용자에게 의해 동시 공유 가능
내용에 의한 참조	- 데이터의 참조는 데이터의 주소가 아닌 저장된 값에 의해 처리됨

3. 데이터베이스 관리시스템(DBMS: Database Management System)의 개념

다수의 응용소프트웨어 및 사용자가 데이터베이스에 접근하여 원활하게 사용할 수 있도록 중간에서 관리해주는 시스템

〈표 3-2〉 데이터베이스 관리시스템 기능

기능	설명
동시성 제어	- 다수 트랜잭션의 동시 처리로부터 데이터 무결성 확보를 위한 제어 수행
회복 관리	- 시스템 오류 및 장애로 인한 데이터 손실 및 결함의 대응
성능 관리	- 데이터 처리 속도 확보를 위한 실행 계획의 최적화
보안 관리	- 비인가 사용자의 접근 제어 및 중요 정보의 암호화

4. 데이터베이스 종류

데이터베이스는 데이터를 관리하는 형태에 따라 계층형, 네트워크형, 관계형, 객체지향형, 객체관계형 데이터베이스가 존재한다

〈표 3-3〉 데이터베이스 종류

종류	설명	특징
계층형 데이터베이스	<ul style="list-style-type: none"> - HDB: Hierarchical DB - 데이터(Record)를 상하 종속적 관계로 계층화하여 관리하는 데이터베이스 	<ul style="list-style-type: none"> - 한 레코드는 필드로 구성되며 다른 레코드들의 포인터로 구성 - 빠른 접근 속도 - 데이터 변화에 대한 유연성 낮음
망형 데이터베이스	<ul style="list-style-type: none"> - NDB: Network DB - 망형의 네트워크 구조로 데이터를 관리하는 데이터베이스 	<ul style="list-style-type: none"> - 레코드에 부모 레코드의 포인터도 관리 가능 - 데이터 변화에 대한 유연성 좋음 - 데이터 모델링 복잡
관계형 데이터베이스	<ul style="list-style-type: none"> - RDB: Relational Database - 데이터 간의 관계 구조를 관리하는 데이터베이스 	<ul style="list-style-type: none"> - 레코드의 집합을 테이블로 구성 - 반복 그룹, 자료형 한계 - 동적 변화로 유연성 높음
객체지향형 데이터베이스	<ul style="list-style-type: none"> - OODB: Object Oriented DB - 데이터를 객체화하여 관리하는 데이터베이스 	<ul style="list-style-type: none"> - 객체 재사용, 캡슐화, 상속 가능 - 멀티미디어 지원 가능 - 데이터 모델링 복잡
객체관계형 데이터베이스	<ul style="list-style-type: none"> - ORDB: Object Relational DB - 기존의 RDB에 객체의 개념을 적용한 데이터베이스 	<ul style="list-style-type: none"> - 개발자가 데이터형 정의 가능 - 반복 그룹, 자료형 한계 극복 - 데이터 모델링 용이

② 트랜잭션(Transaction)

1. 트랜잭션의 개념

트랜잭션은 데이터베이스의 상태를 변화시키기 위한 최소 작업 단위로 한 번에 처리되어야 하는 질의어(SQL)의 묶음이다

2. 트랜잭션의 특징

트랜잭션이 원활하게 처리되기 위해서는 ACID가 준수되어야 한다

〈표 3-4〉 트랜잭션의 ACID 특징

특징	설명
Atomicity(원자성)	- 데이터베이스에 트랜잭션은 모두 반영되거나 전혀 반영되지 않아야 함
Consistency(일관성)	- 트랜잭션 시작 시점에 참조한 데이터는 종료까지 일관성을 유지해야 함
Isolation(고립성)	- 동시에 다수 트랜잭션이 처리되는 경우 서로의 연산에 개입하면 안 됨
Durability(지속성)	- 트랜잭션이 성공적으로 완료되면 처리 결과는 영속적으로 반영되어야 함

3. 트랜잭션의 고립화 수준(Transaction Isolation Level)

트랜잭션 동시에 처리되는 과정에서 트랜잭션 간에 Read, Write에 대해 허용하는 수준을 SQL 표준에서 4가지로 정의한다. 데이터베이스 관리시스템들은 이러한 수준을 제어할 수 있는 기능을 제공한다

〈표 3-5〉 트랜잭션 고립화 수준

Isolation Level	설명
Read Uncommitted	- Commit되지 않은 데이터의 Read 허용
Read Committed	- 질의 시작 전 Commit된 데이터의 Read만 허용
Repeatable Read	- 트랜잭션 시작 전 Commit 된 데이터의 Read만 허용
Serializable	- 병행 처리되지 않고 순차적으로 처리되는 것과 동일한 수준

3 - 2 관계형 데이터베이스 테이블 정의

① 데이터 모델링

1. 데이터 모델링 개념

시스템으로 구성하기 위한 데이터의 집합(실체)을 도출한 후 각 집합을 구성하는 세부 속성과 식별자를 정의하고 각 데이터 집합 간의 관계를 정해진 표기법(Notation)으로 시각화 하는 과정이다.

2. 데이터 모델링 프로세스

데이터 모델링은 요구사항 수집 및 분석을 통해 도출된 데이터 집합을 이용해 개념, 논리, 물리 모델링 과정을 통해 데이터베이스를 구현한다

〈표 3-9〉 데이터 모델링 유형

프로세스	설명
1. 요구사항 수집/분석	- 사용자 및 데이터베이스 용도를 식별하고 사용자 요구사항을 수집 및 분석하여 요구사항 정의서를 작성하는 과정
개념 모델링	- 현실세계의 정보를 추상화하여 주제 영역을 정의하고 식별자/관계/속성을 도출해 개념 ERD(Entity-Relationship Diagram) 작성
2. 설계	논리 모델링
	- 개념 데이터 모델을 특정 데이터베이스(계층형, 망형, 관계형)에 적합하도록 구조화하여 논리 ERD를 작성하는 과정
	물리 모델링
	- 특정 DBMS(Database Management System)에서 활용 가능하도록 물리 ERD 및 테이블 정의서 작성 과정
3. 데이터베이스 구현	- 물리 ERD 및 테이블 정의서를 이용해 특정 DBMS에 데이터베이스를 구축하는 과정

② E-R(Entity-Relationship) 데이터 모델

1. E-R 데이터 모델의 개념

현실세계의 구성요소들을 데이터베이스로 관리하기 위해 유형화(Classification), 집단화(Aggregation), 일반화(Generalization) 과정을 통해 추상화(Abstraction)하여 개체(Entity)와 관계(Relationship)로 구조화한 데이터 모델이다.

* 추상화(Abstraction)

현실세계에 존재하는 개체(Entity)들의 특징을 유형화(Classification), 집단화(Aggregation), 일반화(Generalization)하여 추상적 개념으로 표현하는 과정

- 유형화(Classification): 동일한 특성을 하나의 유형(Class)로 분류하여 속성으로 정의

[예시] 인사담당자, 영업담당자, 구매담당자를 담당자구분으로 유형화

- 집단화(Aggregation): 연관된 속성을 하나의 집단으로 분류하여 새로운 속성으로 정의

[예시] 담당자구분, 부서, 이름, 전화번호를 담당자로 집단화

- 일반화(Generalization): 공통 속성을 파악하여 전체집합과 부분집합으로 분류

[예시] 이름, 전화번호를 사람으로 일반화

2. E-R(Entity-Relationship) 데이터 모델의 구성요소

〈표 3-10〉 E-R(Entity-Relationship) 데이터 모델 구성요소

구성요소	설명
개체(Entity)	<ul style="list-style-type: none">- 사람, 사물, 사건, 개념 등의 유무형의 특성을 공유하는 독립적인 실체로 인스턴스(Instance, 값)의 집합- 유일한 식별자에 의해 식별 가능해야 하며 반드시 하나 이상의 속성(Attribute), 하나 이상의 관계(Relation)로 구성되어야 함
속성(Attribute)	<ul style="list-style-type: none">- 개체(Entity)를 구성하는 특성
관계(Relationship)	<ul style="list-style-type: none">- 개체 간의 상호 연관성을 표현하는 페어링(Paring)의 집합

3. E-R(Entity-Relationship) 데이터 모델 표기법

(1) E-R 데이터 모델 표기법 유형

〈표 3-11〉 E-R(Entity-Relationship) 데이터 모델 표기법 유형

유형	설명
Chen 표기법	<ul style="list-style-type: none">- 1976년 Peter Chen에 의해 만들어진 최초의 표기법- 교육용으로 주로 사용되며 실무적으로는 사용하지 않음
정보공학(IE) 표기법	<ul style="list-style-type: none">- 1981년 Clive Finkelstein과 James Martin에 의해 발표되고 1980년 중반 James Martin에 의해 체계가 정리된 표기법
바커(Baker) 표기법	<ul style="list-style-type: none">- 1986년 Richard Barker, Ian Palmer, Harry Ellis 등에 의해 개발되고 이후 Richard Barker에 의해 지속적으로 개선된 표기법

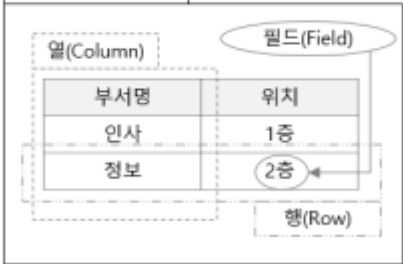
③ 관계형 데이터베이스 테이블

1. 관계형 데이터베이스 테이블 개념

데이터 모델링을 통해 도출된 개체(Entity)와 관계(Relationship)를 데이터베이스에서 관리하기 위한 2차원 표 형태의 저장 공간

2. 관계형 데이터베이스 테이블 구성

〈표 3-15〉 속성(Attribute) 표기법

테이블(Table)		구성	설명
		필드	열과 행이 교차하는 지점 하나의 값
		열	개체를 구성하는 속성(Attribute)
		행	하나의 인스턴스(Instance)를 구성하는 속성 값의 집합으로 튜플(Tuple), 레코드(Record)라고도 함