운영체제 기초 활용하기

1-1 운영체제의 특징 파악

1. 운영체제의 특징

운영체제는 사용자로 하여금 컴퓨터의 하드웨어를 보다 쉽게 사용할 수 있도록 인터페이스를 제공해 주는 소프트웨어이다.

하드웨어는 컴퓨터의 장치를 제어하고 데이터를 처리하는 **중앙 처리 장치**, 데이터를 저장하는 **기억 장치**, 외부와의 통신을 담당하는 **통신 장치** 그리고 데이터 입력과 출력을 담당하는 **입출력 장치** 등으로 구분될 수 있다.

즉, 운영체제는 사용자 편의성을 위한 인터페이스인 동시에 다양한 자원을 관리하는 자원 관리자이다.

2. 윈도즈 계열 운영체제의 특징

마이크로소프트사에서 제작

사용자가 컨트롤하는 **마우스의 아이콘을 이용**하여 소프트웨어를 실행시키는 편리한 인터페이스를 지원하는 것이 특징이다.

마이크로소프트사만이 수정 및 배포할 수 있으며 고객 지원이 체계적이라는 장점을 가지고 있지만, 문제점(버그 등)이 발견되었을 시 수정에 시간이 걸린다는 단점이 있고, 이런 시간적인 차이를 이용하는 악성 해커들로 인하여 유닉스 계열의 운영체제에 비하여 **보안에 취약**하다는 문제점이 지속적으로 제기되고 있다.

사용자용 윈도즈와 서버용 윈도즈의 차이점
- 서버 관리자 기능이 있다.
- Microsoft 연동 계정을 사용할 수 없다.
- Active Directory Domain Controller 기능이 있다.
- 윈도즈 Server 백업 기능 외 서버에 필요로 하는 기능이 있다.
- 윈도즈 Server 전용 보안 업데이트를 추가로 받을 수 있다.

3. 리눅스/유닉스 계열 운영체제의 특징

유닉스는 1960년대 AT&T Bell 연구소, MIT 그리고 General Electric이 공동 연구로 개발에 착수하여 개발한 운영체제이다. **멀티태스크 기능에 초점**을 맞추었으며 초기 운영체제 Multics를 만들었다.

C 언어로 재이식되어 대중화의 기반을 마련하였고, 1970년대 AT&T가 본격적으로 유닉스 시스템을 판매하게 되었다.

현재 SYSTEM V 계열과 BSD(Berkely Software Distribution) 계열 이 둘의 장점을 통합한 버전의 유닉스가 배포되고 있다.

IBM의 사용운영체제인 AIX, 오라클의 솔라리스(Solaris), HP의 UX가 그 예이다.

리눅스는 유닉스의 호환 커널이다. 1991년 리누스 토발즈(Linus Tovalds)는 `**자유 소프트웨어(Free Software)**' 정책 하에서 완전히 자유롭고 재배포가 가능한 운영체제인 유닉스를 만들었다.

`자유 소프트웨어'란 금전적 무료가 아닌 `원하는 대로의 실행', `무료나 유료로 복제물 재배포', `필요에 따른 개작' 등 포괄적인 `자유'를 부여하는 것을 의미한다.

리눅스는 **수천 명 이상의 개발자들이 코드를 보고 update**를 하고 있다. 따라서 버그 발생 시 **다수의 개발자가 수정에 참여하여 빠른 업데이트가 가능**하지만, 윈도즈와 같은 체계적인 지원이 상대적으로 부족하여 일반인들보다는 전문가들이 사용하고 있다. 유닉스는 현재 **서버 시장과 슈퍼컴퓨터시장**에서 매우 높은 점유율을 가지고 있다.

분류	리눅스	유닉스
비용	• 대부분 무료이며 지원 정책에 따라 일부 유료 서비스 제품도 있음.	• 대부분 유료
주 사용자	• 개발자, 일반 사용자	• 메인프레임, 워크스테이션 등 대형 시스템 관리자
개발사	• 커뮤니티	• IBM, HP 등
개발 배포	• 오픈소스 개발	• 대부분 사업자에 의해 배포
사용량	• 모바일폰, 태블릿 등 다양하게 사용	• 인터넷 서버, 워크스테이션 등 대형 서비스에 주로 사용
사용자 편의	• GUI 제공, 파일시스템 지원, BASH 셸 사용	• 커맨드 기반이 주였으나 GUI도 제공하는 추세, 파일시스템 제공 • 기본은 Bourne Shell, 현재는 많은 Shell과 호환 가능

4. 기타 운영체제

매킨토시 운영체제 OS X

유닉스 기반으로 만들어져 **애플사의 제품군**에서만 사용이 가능한 그래픽 기반 운영체제이다.

매킨토시 OS는 프로그램을 카피하고 삭제함으로써 install과 uninstall의 과정을 단순화하였으며, 드라이버 설치 또한 OS의 확장 폴더에 넣고 재부팅을 하면 인식되어 매우 간단하다.

1-2 운영체제 기본 명령어 활용

1. 운영체제 기본 명령어

운영체제를 제어하기 위한 방법은 CLI(Command Line Interface)와 GUI(Graphic User Interface)가 있다.
CLI는 사용자가 **직접 명령어를 입력**하여 컴퓨터에게 명령을 내리는 방식이며, GUI는 **마우스로 화면을 클릭**하여 컴퓨터를 제어하는 방식이다.
CLI 증가 이유 : 오픈소스 기반의 개발환경이 급격히 늘어나며 GitHub등의 사용이 중요해지게 되었다.

2. 윈도즈 운영체제의 기본 명령어

1) CLI(Command Line Interface) 기본 명령어
CLI 명령어를 입력하기 위해서는 Command 창이 필요하다. 프로그램 및 파일 검색에서 `CMD'를 입력하거나 윈도즈 보조 프로그램에서 **`명령 프롬프트'**를 선택하여 Command 창을 호출할 수 있다.
명령어는 시스템 제어, 파일 제어, 실행, I/O, 실행 중 프로그램 중지, 진단 및 검증과 같이 다양하다. 명령어는 `Help'를 command 창에 입력함으로써 검색이 가능하다.

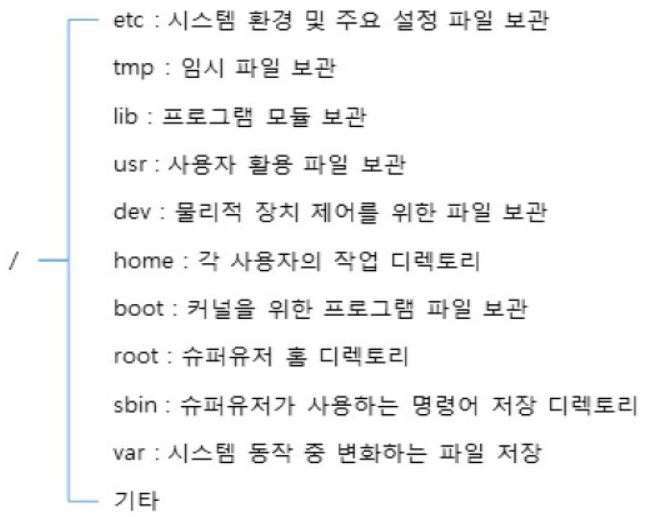
2) GUI(Graphic User Interface) 기본 명령어
윈도즈 내에서 파일을 이동하고 프로그램을 실행하는 것 등 모든 것이 GUI 명령에 해당한다. 메모리나 디스크 제어 등이 필요할 경우에는 **제어판**에서 필요 기능을 선택하여 명령을 내릴 수 있다.

3. 리눅스/유닉스 계열 운영체제의 기본 명령어

리눅스와 유닉스 명령어는 Shell에서 입력할 수 있다. Shell이란 컴퓨터 내부를 관리하는 Kernel과 사용자 간을 연결하는 Command 창이다.
- 세션별 변수를 설정, 운영체제를 사용자가 원하는 상태로 설정하도록 지원
- 사용자 요청에 기반한 명령렬 작성
- 백그라운드 처리, 서브 셸 생성
- 일련의 명령어를 묶어 처리하는 스크립트 기능 지원 등

1) CLI(Command Line Interface) 기본 명령어
리눅스는 **최상위 유저를 CLI 환경에서 #**으로 표시하며 **일반 유저를 $**로 표시한다. 명령어에 대한 도움말은 **-help, -h, #man**을 명령어 뒤에 붙임으로써 확인할 수 있다.
명령어는 파일 디렉터리 관리, 유저 관리, 권한 관리, 프로세스관리, 통신 관련 등으로 구분될 수 있다. **최상위 디렉터리는 /**이며 root는 최상위 디렉터리 아래의 **root 계정의 홈 디렉터리임**을 의미한다. 타 운영체제에서 root를 최상위 디렉터리라고 하는 것과 다르다.

2) GUI(Graphic User Interface) 기본 명령어
리눅스의 GUI는 윈도즈와 같이 기본 설정이 아닌 경우가 많아 버전별로 별도의 설치방법에 따라 GUI 환경을 설치해야 한다.



1-3 운영체제 핵심 기능 파악

1. 운영체제 핵심 기능

운영체제는 중앙 처리 장치, 메모리, 스토리지, 주변 기기 등을 적절히 관리한다. 특히 **주기억 장치와 메모리, 메모리와 스토리지 사이의 속도차**로 인해 여러 가지 관리 기법들이 개발되었다.

초기에는 메모리 용량에 제한이 많아 소프트웨어 개발 시 메모리 관리가 매우 중요했으나, 최근에 들어서는 운영체제에서 대부분 자동으로 관리해 주므로 사용이 편리해졌다. 또 개발 및 시스템 환경이 **클라우드화** 되면서 자원에 대한 관리 노력이 많이 줄었다.

1) 메모리관리

프로그램의 실행이 종료될 때까지 메모리를 가용한 상태로 유지 및 관리하는 것을 **메모리관리**라고 한다.

메모리에 있는 프로그램은 중앙 처리 장치인 CPU로 이동하여 처리된다. CPU는 Virtual or logical address를, 메모리는 physical address를 사용하는데 이를 매핑하는 것은 MMU(Memory Management Unit)가 담당한다.

즉, CPU와 지속적으로 데이터를 송수신하는 상황에서 어떤 부분의 메모리가 현재 사용되는지, 어떤 순서로 메모리에 입출력되어야 하는지, 메모리 공간이 필요할 경우 어떻게 확보 및 제거할지에 대한 종합적인 관리가 메모리관리이다.

리눅스/유닉스 메모리관리 기능을 사용한다.

- meminfo 명령어를 이용하여 메모리 상태를 점검한다.
- 메모리가 부족하면 swapping(프로그램에 할당된 메모리 일부를 디스크에 저장) 기법을 이용한다.
- 메모리가 부족하면 /proc/sys/vm/min_free_kbytes(최소 가용 메모리 확보 기준 해제)를 통해 추가적인 메모리를 확보한다.

2) 프로세스관리

프로그램은 파일 형태로 저장하여 관리되다가 실행을 시키면 동작을 하게 된다. 이때 실행 중인 프로그램을 **프로세스**(Process)라고 한다.

프로세스관리 기법에는 **`일시 중지 및 재실행', `동기화', `통신', `교착상태 처리', `프로세스 생성 삭제'** 등이 있다.

윈도즈에서는 **작업관리자의 프로세스 탭**에서 다양한 프로세스를 조회할 수 있고 프로그램이 정상 동작하지 않을 때 프로그램 끝내기를 통해 프로세스를 중단시킬 수도 있다.

리눅스/유닉스 프로세스관리 기능을 실행한다.

- /proc 디렉터리 아래서 현재 실행되는 프로세스를 확인한다.
- 세부적인 프로세스를 확인한다. 프로세스의 PID를 확인한 뒤 해당 디렉터리로 이동하여 ls 명령어를 사용하여 조회한다. 해당 PID에 해당하는 프로세스가 종료되면 임시로 생성되었던 디렉터리는 사라지는 것을 확인한다.
- TOP 툴을 사용하여 현재 시스템의 CPU와 RAM 사용률을 모니터링한다.

1. 가상화, 클라우드

1) 가상화

가상화는 물리적인 리소스들을 사용자에게 하나로 보이게 하거나, 반대로 하나의 물리적인 리소스를 여러 개로 보이게 하는 것을 의미한다. 가상화는 서버의 가동률을 60~70% 이상으로 올릴 수 있다.

가상화를 통해 사용자는 하나의 PC에 여러 개의 운영체제(윈도즈, 유닉스/리눅스 등)를 설치하여 개발되는 프로그램을 **다양한 환경에서 테스트**할 수 있도록 하며, 서로 다른 운영체제에서만 구동되는 프로그램을 실행시키도록 지원하기도 한다.

가상화는 크게 플랫폼 가상화와 리소스 가상화로 구분될 수 있다.

플랫폼 가상화는 하드웨어 플랫폼 위에서 실행되는 호스트 프로그램이 게스트 프로그램을 만들어 마치 독립된 환경을 만들어 낸 것처럼 보여 주는 것이다.

리소스 가상화는 메모리, 저장 장치, 네트워크 등을 결합하거나 나누는 것

2) 클라우드
인터넷 기반에서 구동되는 컴퓨팅 기술을 의미한다.
클라우드 컴퓨팅을 이용하면 응용 프로그램을 필요에 따라 불러 사용하고, 데이터를 손쉽게 저장 및 추출할 수 있다.
클라우드 서비스는 IaaS, PaaS, SaaS 등으로 구분된다.

- IaaS(Infrastructure as a Service): 웹상에서 구글, 마이크로소프트, 아마존 등에서 제공하는 환경의 네트워크, 보안, 데이터 저장소, 콘텐츠 딜리버리 서비스를 포함한 다양한 **인프라를 임대**하여 이용할 수 있는 서비스

- PaaS(Platform as a Service): 운영체제가 이미 구성되어 있는 상태에서 사용자는 **데이터와 애플리케이션만 직접 관리**할 수 있는 서비스

- SaaS(Software as a Service): **인프라와 운영체제뿐만 아니라 사용할 수 있는 소프트웨어**까지 갖추어져 웹상의 로그인만으로 다양한 소프트웨어를 사용한 만큼 비용을 지불해가며 사용할 수 있는 서비스

데이터베이스 기초 활용하기

2-1. 데이터베이스 종류 및 선정

1. 데이터베이스 종류

1) 데이터베이스 개요
데이터베이스는 다수의 인원, 시스템 또는 프로그램이 사용할 목적으로 **통합하여 관리되는 데이터의 집합**이다. **자료의 중복성 제거, 무결성 확보, 일관성 유지, 유용성 보장**은 데이터베이스 관리의 핵심이다.
DBMS(Database Management System)는 위와 같은 데이터 **관리의 복잡성을 해결**하는 동시에 **데이터 추가, 변경, 검색, 삭제 및 백업, 복구, 보안 등의 기능**을 지원하는 소프트웨어이다. 저장되는 정보는 **텍스트, 이미지, 음악 파일, 지도 데이터 등** 매우 다양하며, SNS의 발달과 빅데이터의 폭넓은 활용으로 인해 데이터의 종류와 양은 급격히 증가 중이다.

2) 데이터베이스 종류
(1) 파일시스템
파일에 이름을 부여하고 저장이나 검색을 위하여 논리적으로 그것들을 어디에 위치시켜야 하는지 등을 정의한 뒤 관리하는 데이터베이스 전 단계의 데이터 관리 방식이다.
- ISAM: 자료 내용은 주 저장부, 자료의 색인은 자료가 기록된 위치와 함께 색인부에 기록되는 시스템
- VSAM: 대형 운영 체계에서 사용되는 파일 관리시스템

(2) 계층형 데이터베이스 관리시스템(HDBMS: Hierarchical Database Management System)
데이터를 상하 종속적인 관계로 계층화하여 관리하는 데이터베이스이다. 데이터에 대한 접근 속도가 빠르지만, 종속족인 구조로 인하여 변화하는 데이터 구조에 유연하게 대응하기가 쉽지 않다.
계층형 데이터베이스는 **하위 속성이 중복된 상위 속성에 포함될 수 없다.**

(3) 망형 데이터베이스 관리시스템(NDBMS: Network Database Management System)
데이터의 구조를 **네트워크상의 망상 형태**로 논리적으로 표현한 데이터 모델이다. 트리구조나 계층형 데이터베이스보다는 유연하지만 설계가 복잡한 단점이 있다.
망형은 하위 속성이 다수의 상위 속성에 포함되어 정의될 수 있다.

(4) 관계형 데이터베이스 관리시스템(RDBMS: Relational Database Management System)
가장 보편화된 데이터베이스 관리시스템이다. 데이터를 저장하는 테이블의 일부를 다른 테이블과 상하 관계로 표시하며 **상관관계**를 정리한다. 변화하는 업무나 데이터 구조에 대한 유연성이 좋아 유지 관리가 용이하다.
- Oracle: **미국 오라클사**에서 개발한 데이터베이스 관리시스템으로 유료이다. 리눅스/유닉스/윈도 모두를 지원하며 대형 시스템에서 많이 사용한다.
- SQL Server: **마이크로소프트사**에서 개발한 관계형 데이터베이스 시스템이다. 마이크로소프트사 제품이기 때문에 윈도즈 서버에서만 구동이 되며, 마이크로소프트사의 개발언어인 C# 등과 가장 잘 호환된다.
- MySQL: 썬 마이크로시스템에서 소유했던 관계형 데이터베이스 시스템이었으나 오라클에서 인수하였다. 리눅스, 유닉스, 윈도에서 모두 사용이 가능하고 **오픈소스 기반**으로 개발되었다.
- Maria DB: MySQL 출신 개발자가 만든 데이터베이스로 **MySQL과 완벽히 호환**된다.

3) 데이터베이스 관리시스템(DBMS) 특징
- **데이터 무결성**: 부적절한 자료가 입력되어 동일한 내용에 대하여 서로 다른 데이터가 저장되는 것을 허용하지 않는 성질
- **데이터 일관성**: 삽입, 삭제, 갱신, 생성 후에도 저장된 데이터가 변함없이 일정
- **데이터 회복성**: 장애가 발생하였을 시 특정 상태로 복구되어야 하는 성질
- **데이터 보안성**: 불법적인 노출, 변경, 손실로부터 보호되어야 하는 성질
- **데이터 효율성**: 응답 시간, 저장 공간 활용 등이 최적화되어 사용자, 소프트웨어, 시스템 등의 요구 조건을 만족 시켜야 하는 성질

4) 상용 데이터베이스 관리시스템 및 오픈소스 기반 데이터베이스 관리시스템
- **상용 데이터베이스** 관리시스템은 특정 회사에서 **유료**로 판매하는 시스템이다. **유지 보수와 지원이 원활**한 장점이 있다.
- **오픈소스 기반 데이터베이스 시스템**은 오픈소스 라이선스 정책을 준용하는 범위 내에서 사용이 **자유**롭다.

<표 2-2> 데이터베이스 유형 및 특징

구분	설명	예제
관계형 DBMS (Relational DBMS)	- 테이블의 구조(스키마)를 정의하고 테이블 간의 관계를 정의하여 데이터를 관리 - 가장 광범위 하게 쓰이는 DBMS - 비관계형 부분까지 확장하여 관리 범위를 넓힌 형태의 관계형 DBMS 출시	Oracle MySQL MS SQL Server PostgreSQL DB2 Maria DB
문서 저장 DBMS (Document store)	- 관계형 DBMS와는 달리 스키마 구조 필요 없음 - 일관된 구조가 필요 없음 - 칼럼은 하나 이상의 값을 가질 수 있음 - Client단에서 후처리 필요	Mongo DB Amazon Dynamo DB Couchbase MS Azure cosmos DB
그래프 DBMS (Graph DBMS)	- 노드와 에지로 특징되는 요소 특화 - 노드 간 관계를 구조화하여 저장	Neo4j MS Azure cosmos DB OrientDB ArangoDB
키값 DBMS (Key-value DBMS)	- 가장 간단한 형태의 DBMS - 임베디드 시스템과 같은 간단한 시스템에 적합	Redis Amazon Dynamo DB Memcached

2.오픈소스 DBMS와 상용 DBMS의 특징과 주요 제품

1). 데이터베이스 관리시스템 제품 중 오픈소스 기반의 제품이 차지하는 비중이 높아지는 이유
- **기업들의 원가 절감 노력과의 상관관계**
- 인공 지능, 클라우드, 빅데이터 등 새로운 기술의 증가에 따른 **오픈소스 데이터베이스 관리시스템**의 대응 동향
- **보안, 안정성에 대한 우려를 오픈소스 진영에서 해결하고 있는 노력**
- **정책적으로 오픈소스 데이터베이스 관리시스템 활성화를 추진**

<표 2-3> 상용화 및 오픈소스 DBMS

Top 5 상용화 DBMS	Top 5 오픈소스 기반 DBMS
Oracle	MySQL
MS SQL Server	PostgreSQL
DB2	Mongo DB
Microsoft Access	Redis
Teradata	Elasticsearch

2-2. 관계형 데이터베이스 활용

1. ERD(E-R Diagram)

1) ERD 개요
ERD는 업무 분석 결과로 도출된 **실체(엔티티)와 엔티티 간의 관계를 도식화**한 것이다.
ERD로 요소들 간 연관성을 도식화하여 데이터베이스 관리자, 개발자, 사용자 모두 데이터의 흐름과 연관성을 공통적으로 쉽게 확인할 수 있다.

2) ER Model
ERD로 도식화하기 전 각 개체를 **사각형, 화살표, 마름모로 표기한 형태**를 ER 모델이라고 한다.

(1) 엔티티(Entity)
사물 또는 사건으로 정의되며 개체라고도 한다. ERD에서 엔티티는 사각형으로 나타내고 사격형 안에는 엔티티의 이름을 넣는다.
- 가능한 한 대문자로 엔티티 이름을 써 주며 **단수형**으로 명명한다.
- **유일한 단어**로 정한다.

(2) 속성(Attribute)
엔티티가 가지고 있는 요소 또는 성질을 속성이라 부른다. 속성은 선으로 연결된 동그라미로 표기하거나 표 형식으로 표기하기도 한다.
- 속성명은 **단수형**으로 명명한다.
- **엔티티명을 사용하지 않는다.**
- 속성이 필수 사항(Not Null)인지, 필수 사항이 아닌지(Null) 고려하여 작성한다.

(3) 관계(Relationship)
두 엔티티 간의 관계를 정의한다. 개체는 사각형(□), 속성은 타원형(○)을 이용하여 표시한다.

2. ERD(E-R Diagram) 작성
1) ERD 작성 표기
ERD는 종이, 화이트보드, 포스트잇, 전문 소프트웨어 등을 사용하여 **작성자가 원하는 도구**를 사용하여 작성하면 된다.

2) ERD 최적화

(1) 테이블 정의
업무나 시스템을 분석하여 엔티티, 속성을 추출한 뒤 테이블을 작성한다.

(2) 정규화 수행
데이터베이스 정규화는 **무결성을 확보**하고 **중복성을 배제**하여 테이블에 정확한 데이터가 들어가도록 하는 데 목적이 있다. 데이터의 중복성을 없애면 저장 공간을 최소화하고 시스템의 속도 또한 빠르게 할 수 있다.
- **1차 정규화**: 반복되는 그룹의 속성을 별도로 추출한다.
- **2차 정규화**: 부분 함수적 종속성을 제거한다.
- **3차 정규화**: 키에 종속되지 않은 칼럼을 제거한다.

(3) ERD 관계형 스키마(논리 모델, 물리 모델) 작성
사용자가 식별하기 쉬운 한글 또는 영어 단어로 작성된 **논리 ERD**를 작성한다. 시스템이 식별하기 쉽도록 코드화된 물리 ERD를 작성한다.

3. 관계형 데이터베이스 테이블 생성

1) 생성 언어
데이터베이스의 종류와 상관없이 명령어는 국제 표준으로 제정된 SQL를 사용한다. 미국 표준 연구소인 **ANSI**와 국제 표준화 기구인 ISO에서 **SQL**을 관계 데이터베이스의 표준 질의어로 채택하고 표준화하고 있다.

2) 테이블 관리
- 쿼리 명령어 “Create”를 사용하면 테이블 및 속성을 생성할 수 있다.
- 쿼리 명령어 “Drop”을 사용하면 테이블을 삭제할 수 있다.
- 쿼리 명령어 “Show tables”를 사용하면 테이블의 내용을 확인할 수 있다.

4. 관계형 스키마 작성을 위해 테이블 형태로 표시
- 테이블의 가장 위 칸은 유일하게 필드를 구분할 수 있는 **구분자(유일키)**가 되어야 한다.

5. 테이블 간 관계를 설정한다.
- 테이블 간 관계를 표시한다. 표기 시에는 **1:1, 1:N, N:1, N:N의 관계를 고려**해야 한다.
- 테이블 간 **연결할 수 있는 속성을 부여**하여 하나의 테이블에서 다른 테이블의 값을 찾아갈 수 있어야 한다.

6. 관계형 데이터베이스 테이블을 생성한다.

1) 관계형 테이블 생성을 위한 물리 스키마을 작성
SQL 표준 언어는 **한글 입력이 허용되지 않는다.**

2) 테이블 생성을 위한 SQL 명령어를 작성
작성 시 필요한 datatype은 **정수형, 실수형, 날짜형** 등 다양하다.

2-3. 데이터베이스 관리

1. 데이터베이스 기본 연산

CRUD는 데이터베이스가 가지는 기본적인 데이터 처리 기능인 Create(생성), Read(읽기), Update(갱신), Delete(삭제)를 말한다.

1) 기본 처리 SQL 설명
Create Insert 테이블 내 칼럼에 데이터를 **추가**한다.
Read Select 테이블 내 칼럼에 저장된 데이터를 **불러**온다.
Update Update 테이블 내 칼럼에 저장된 데이터를 **수정**한다.
Delete Delete 테이블 내 칼럼에 저장된 데이터를 **삭제**한다.

2) SQL 기타 명령어 사용
- ALTER DATABASE 명령어를 사용하여 **데이터베이스를 수정**한다.
- ALTER TABLE 명령어를 사용하여 **테이블 구조를 수정**한다.
- DROP TABLE 명령어를 사용하여 **테이블을 전체 삭제**한다.
- CREATE INDEX 명령어를 사용하여 테이블 내 데이터의 검색 속도를 향상시킬 수 있는 **인덱스를 생성**한다.
- DROP INDEX 명령어를 사용하여 **인덱스를 삭제**한다.
- JOINT 기능을 사용하여 복수의 테이블로부터 **데이터를 조합**하여 가져온다.
- 그 외 UNION, GROUP BY 등의 다양한 명령어를 활용한다.

2. 데이터베이스 관리 툴

Oracle, DB2, SAP, SQL Server, Edge 등의 상용 데이터베이스를 편리하게 관리하기 위하여 다양한 관리 툴을 사용한다.
데이터베이스 관리 툴을 사용하면 데이터베이스의 생성, 관리, SQL 문장의 작성, 모니터링등 다양한 기능을 손쉽게 활용할 수 있다. **오픈소스 기반으로 무료로 사용할 수 있는 툴과 상용화로 비용을 지불해야 사용할 수 있는 툴**이 있으며, 지속적으로 새로운 기능을 가진 프로그램들이 나오고 있으므로 자신에게 적합한 툴을 선정하도록 한다.
- 데이터베이스 생성, 삭제
- SQL 명령어 작성 및 실행
- 상태 모니터링: 받은 데이터양, 보낸 데이터양, 동시 연결 수, 실패한 시도 등
- 사용자 계정 관리
- 데이터베이스 내보내기/가져오기
- 환경 설정

1)툴 선정 시 고려 사항
- **협업**: 개발자, 품질 담당자, 필요에 따라 경영층까지 협업이 용이한 툴 선정
- **지원 깊이**: SQL에서 제공하는 명령어에 대한 지원 커버리지
- **시각화**: 데이터의 흐름, 테이블 가시화 등 그래픽 요소 지원 여부
- **이기종 데이터베이스 지원**: 하나의 툴로 다양한 복수의 데이터베이스 지원
- **비용**: 라이선스 비용, 운용 인력 급여 등 유지하는 데 들어가는 총비용 고려
- **편의 기능**: 관리자, 사용자를 구분한 접근 권한 설정 등 편의 기능 고려

2) 데이터베이스 관리 툴의 주요 기능
- 논리적/물리적 모델 구축
- 스크립트를 사용한 자동 생성 및 수정
- SQL 관련 최적화 수행
- 벤치마크 시나리오 생성
- 가상 유저를 가정한 다양한 테스트 환경 단순화
- 부하 테스트 수행
- 데이터베이스 상태 모니터링 및 문제 해결 방안 제시

3. 오픈소스 기반 DBMS 및 DBMS 관리 툴

정보통신산업진흥원에서는 공개SW포털이라는 웹 사이트를 운영하여 다양한 오픈소스 정보를 정기적으로 업데이트 하고 있다(https://www.oss.kr).

3. 네트워크 기초 활용하기

3-1. 네트워크 계층 구조 파악

1. 네트워크 개요

원하는 정보를 원하는 수신자 또는 기기에 정확하게 전송하기 위한 기반 인프라를 **네트워크**라고 한다. 정보 전달 시에는 약속한 규칙에 따라야 하는데 이를 **프로토콜**이라고 한다.

WAN 광대역 네트워크
- LAN에 비해 전송 거리가 넓음. 라우팅 알고리즘이 필요함
- LAN 대비 에러율이 높고 전송 지연이 큼

LAN 근거리 네트워크
- 한 건물 또는 작은 지역을 커버하는 네트워크임

1) WAN(Wide Area Network)
국가, 대륙과 같이 **광범위한 지역을 연결하는 네트워크**이다. 거리에 제약이 없으나 다양한 경로를 지나 정보가 전달되므로 **LAN보다 속도가 느리고 에러율도 높다.** 전용 회선 방식은 통신 사업자가 사전에 계약을 체결한 송신자와 수신자끼리만 데이터를 교환하는 방식이며, 교환 회선 방식은 공중망을 활용하여 다수의 사용자가 선로를 공유하는 방식이다.

(1) 회선 교환 방식
물리적 전용선을 활용하여 데이터 전달 경로가 정해진 후 동일 경로로만 전달이 된다.
데이터를 동시에 전송할 수 있는 양을 의미하는 대역폭이 **고정되고 안정적인 전송률을 확보**할 수 있다.

(2) 패킷 교환 방식
패킷이라는 단위를 사용하여 데이터를 송신하고 수신한다. 패킷이란 정보를 일정한 크기로 분할한 뒤 각각의 패킷에 송수신 주소 및 부가 정보를 입력한 것이다. 현재 컴퓨터 네트워크에서 주로 사용하는 방식이다.

2. OSI(Open System Interconnection) 7계층

국제 표준화 기구인 ISO(International Standardization Organization)에서 개발한 **네트워크계층 표현 모델**이다. 각 계층은 서로 독립적으로 구성되어 있고, 각 계층은 하위 계층의 기능을 이용하여 상위 계층에 기능을 제공한다. 1계층인 물리 계층부터 7계층인 애플리케이션 계층으로 정의되어 있다.

네트워크 관리 기술의 발달로 인해 최근에는 5, 6계층 레이어는 7계층 레이어로 합쳐 통칭하기도 한다.

1) 계층 계층 이름 설명 주요 장비 및 기술
1계층 : 물리 계층 실제 장비들을 연결하기 위한 연결 장치 - 허브, 리피터
2계층 : 데이터 링크 계층 오류와 흐름을 제거하여 신뢰성 있는 데이터를 전송 - 브리지, 스위치
3계층 : 네트워크 계층 다수의 중개 시스템 중 올바를 경로를 선택하도록 지원 -라우터
4계층 : 전송 계층 송신, 수신 프로세스 간의 연결 - TCP/IP UDP
5계층 : 세션 계층 송신, 수신 간의 논리적 연결 - 호스트(PC 등)
6계층 : 표현 계층 코드 문자 등을 번역하여 일관되게 전송하고 압축, 해제, 보안 기능도 담당 - 호스트(PC 등)
7계층 : 응용 계층 사용자 친화 환경 제공(이메일, 웹 등) - 호스트(PC 등)

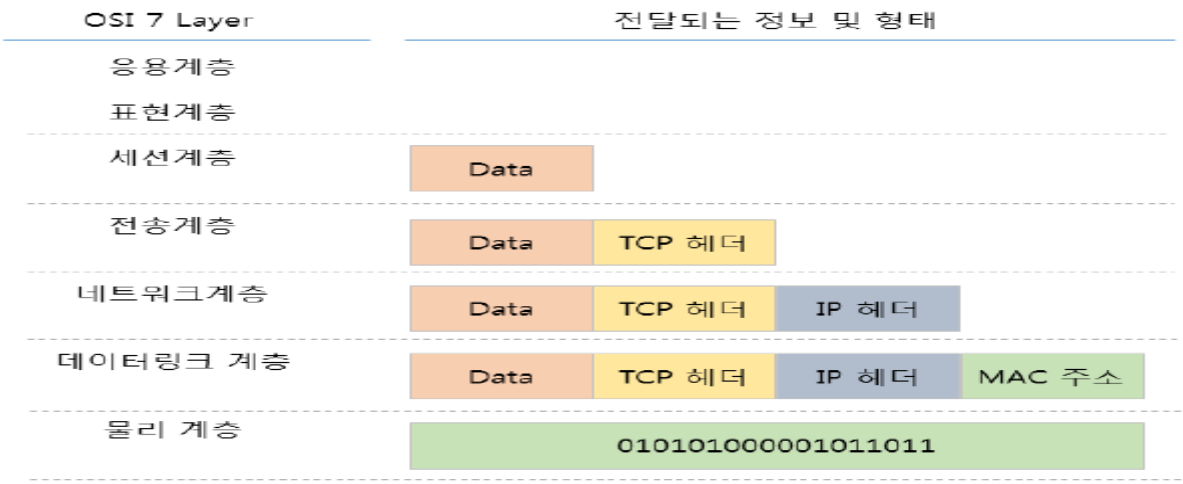
3. 네트워크 주요 장비

1) 허브, 리피터
허브는 **여러 대의 컴퓨터를 연결하여 네트워크로 보내거나 하나의 네트워크로 수신된 정보를 여러 대의 컴퓨터로 송신하기 위한 장비**이다.
리피터는 **디지털 신호를 증폭시켜주는 역할**을 하여 신호가 약해지지 않고 컴퓨터로 수신되도록 한다.

2) 브리지, 스위치
브리지와 스위치는 두 시스템을 연결하는 네트워킹 장치이며 두 개의 LAN을 연결하여 훨씬 더 큰 LAN을 만들어 준다.
스위치는 하드웨어 기반으로 처리하기 때문에 속도가 빠르며, **브리지는 소프트웨어 방식으로 처리하기 때문에 속도가 느리다**.
브리지는 포트들이 같은 속도를 지원하는 반면, 스위치는 각기 다른 속도를 지원하도록 제어할 수 있다.
스위치는 제공하는 포트 수가 수십, 수백 개로 2~3개의 포트를 제공하는 브리지보다 많다.
브리지는 Store and Forwarding 전송 방식만을 사용하나, 스위치는 Cut Through와 Fragment Free 방식을 같이 사용한다.
- Store and Forwarding: 데이터를 전부 받은 후 다음 처리를 하는 방식
- Cut Through: 데이터의 목적지 주소만 확인 후 바로 전송 처리하는 방식
- Fragment Free: 위 두 방식의 장점을 결합한 방식

3) 라우터
라우터는 **망 연동 장비**이다. PC 등의 로컬 호스트가 LAN에 접근할 수 있도록 하며, WAN 인터페이스를 사용하여 WAN에 접근하도록 한다. 라우팅 프로토콜은 경로 설정을 하여 원하는 목적지까지 지정된 데이터가 안전하게 전달되도록 한다.

4. OSI 7계층 구성에 따라 전달되는 데이터의 형식



1) Data 구조의 특징을 파악한다.
데이터는 단말기에서 구동되는 응용 프로그램과 기기의 종류에 따라 다른 형태로 전달되며 16bit, 24bit, 36bit, ASCII, BCDIC, Binary 등으로 구분된다.

2) TCP 헤더
TCP 헤더는 **송수산자의 포트번호, 순서 번호, 응답 번호 등을 전달되는 정보**

3) IP 헤더
IP 헤더는 **IP 버전과 전송지 IP, 목적지 IP뿐만 아니라 프로토콜의 종류, 서비스 타입 등이 표준화되어 있다.**
IP 헤더는 IPv4와 IPv6 IP 체계에 따라 다른 표준을 따른다.

4) MAC 주소의 특징을 파악한다.
네트워크 세그먼트의 데이터 링크 계층에서 통신을 위한 네트워크 인터페이스에 할당된 고유 식별자이다.
IEEE 802 네트워크 기술에 네트워크 주소로 사용되며, 이더넷과 와이파이를 포함한 대부분의 기기들이 주소를 가지고 있다.

5. 인터넷망을 통해 전달되는 각 장비의 특성

1) 장비
- **허브**: 다수의 오프라인, 온라인 접속 기기들을 로컬 네트워크(LAN)에 연결하기 위한 장비이다. 수신한 프레임을 수신 포트를 제외한 모든 포트로 전송한다.
- **스위치**: MAC 주소 테이블을 이용하여 목적지 MAC 주소를 가진 장비 측 포트로만 프레임을 전송하는 역할을 한다.
- **스위칭허브**: 스위치 기능을 가진 허브를 의미하며 요즘 사용되는 대부분의 허브가 스위칭허브이다.
- **망(백본) 스위칭허브**: 광역 네트워크를 커버하는 스위칭허브이다.
- **라우터**: LAN과 LAN을 연결하거나 LAN과 WAN을 연결하기 위한 인터넷 네트워킹 장비이다.
- **유무선 인터넷 공유기**: 외부로 부터 들어오는 인터넷 라인을 연결하여 유선으로 여러대의 기계를 연결하거나 무선 신호로 송출여 여러 대의 컴퓨터가 하나의 인터넷라인을 공유할 수 있도록 하는 네트워크 기기이다.
- **브리지**: 두 개의 근거리 통신망(LAN)을 서로 연결해 주는 통신망 연결 장치이다.
- NIC(Network Interface Card): 외부 네트워크와 접속하여 가장 빠른 속도로 데이터를 주고받을 수 있게 컴퓨터 내에 설치되는 장치이다.
- **리피터**: 감쇠된 전송 신호를 새롭게 재생하여 다시 전달하는 재생 중계 장치이다.
- **게이트웨이**: 프로토콜을 서로 다른 통신망에 접속할 수 있게 해 주는 장치이다.

2) L2 스위치와 L3, L4 스위치의 기능상 역할

구분	특징	한계
L2 스위치	- 가장 원초적인 스위치	- 상위 레이어에서 동작하는 IP 이해 불가 - IP 이해 불가에 따른 라우팅도 불가
L3 스위치	- IP 레이에서의 스위칭을 수행하여 외부로 전송	- 라우터와의 경계가 모호함. - FTP, HTTP 등 우선 스위칭 불가
L4 스위치	- TCP/UDP 등 스위칭 수행 - FTP, HTTP 등을 구분하여 스위칭하는 로드 밸런싱 가능	- 애플리케이션 레이어에서 파악이 가능한 이메일 내용 등 정교한 로드 밸런싱 수행 불가

3) 스위치와 라우터의 기능상 차이
라우터는 서로 다른 네트워크 간의 데이터를 전송하고 가장 이상적인 데이터 전달 경로를 설정하여 주는 역할을 수행한다.

3-2. 네트워크 프로토콜 파악

1. 네트워크 프로토콜 개요

네트워크 프로토콜은 컴퓨터나 원거리 통신 장비 사이에서 **메시지를 주고받는 양식과 규칙의 체계**이다. 통신 규약 또는 규칙에는 **전달 방식, 통신 방식, 자료의 형식, 오류 검증 방식, 코드 변환 규칙, 전송 속도** 등을 정하게 된다. 다른 기종의 장비는 각기 다른 통신 규약을 사용하는데 프로토콜을 사용하면 다른 기기 간 정보의 전달을 **표준화**할 수 있다.

프로토콜은 다음과 같은 특징이 있다.
- **단편화:** 전송이 가능한 작은 블록으로 나누어지는 것
- **재조립:** 단편화되어 온 조각들을 원래 데이터로 복원하는 것
- **캡슐화:** 상위 계측의 데이터에 각종 정보를 추가하여 하위 계층으로 보내는 것
- **연결 제어:** 데이터의 전송량이나 속도를 제어하는 것
- **오류 제어:** 전송 중 잃어버리는 데이터나 오류가 발생한 데이터를 검증하는 것
- **동기화:** 송신과 수신 측의 시점을 맞추는 것
- **다중화:** 하나의 통신 회선에 여러 기기들이 접속할 수 있는 기술
- **주소 지정:** 송신과 수신지의 주소를 부여하여 정확한 데이터 전송을 보장하는 것

IP(Internet Protocol) 주소는 전 세계 컴퓨터에 부여되는 **유일한 식별자**이다. IP는 각 나라의공인 기관에서 할당하고 관리하는데, 우리나라의 경우에는 **한국인터넷진흥원**(www.krnic.or.kr)에서 관리한다.
IPv4는 인터넷 초기부터 현재까지 쓰고 있는 주소 체계이며 000.000.000.000과 같이 **12자리로 표시하며 약 43억 개를 부여**할 수 있다.
2018년 현재는 **IPv4와 IPv6가 공존하며 두 개의 주소 체계를 변환하여 사용하고 있으며 이를 담당하는 것을 NAT**(Network Address Translator)라고 한다. IPv6는 이전 버전에 비하여 효율적인 패킷을 처리하고 보안이 강화된 특징이 있다.

(1) IPv4 vs IPv6의 차이점
IPv4 주소는 전 세계적으로 약 43억 개로 제한되어 관리되는 유한한 주소 체계이다. 일부는 특수한 목적으로 예약되어 있다. IPv6 주소는 IPv4의 주소 개수가 고갈되는 문제를 해결하기 위하여 기존의 IPv4 주소 체계를 **128비트 크기로 확장**한 인터넷 프로토콜로 충분한 수의 디바이스에 IP주소를 부여할 수 있다.

구분	IPv4	IPv6
주소 길이	32비트	128비트
표시 방법	8비트씩 4부분으로 10진수 예) 202.30.64.22	16비트씩 8부분으로 16진수 예)2001:0230:abcd:ffff:0000:0000:ffff:1111
주소 개수	약 43억 개	약 43억×43억×43억×43억 개
주소 할당	A, B, C 등 클래스 단위의 비순차적 할당	네트워크 규모 및 단말기 수에 따른 순차적 할당
품질 제어	지원 수단 없음	등급별, 서비스별로 패킷을 구분할 수 있어 품질 보장이 용이
보안 기능	IPsec 프로토콜 별도 설치	확장 기능에서 기본으로 제공
플러그 앤드 플레이	지원 수단 없음	지원 수단 있음
모바일 IP	상당히 곤란	용이
웹 캐스팅	곤란	용이

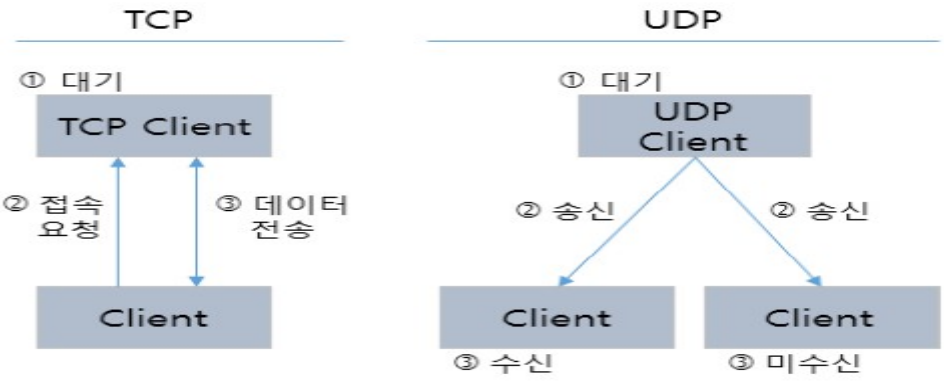
2. TCP/IP 프로토콜

TCP/IP이란 TCP와 IP 프로토콜만을 지칭하는 것이 아니라 UDP(User Datagram Protocol), ICMP(Internet Control Message Protocol), ARP(Address Resolution Protocol), RARP(Reverse ARP) 등 관련된 **프로토콜을 통칭**한다. TCP와 UDP로 구분되는 프로토콜은 트랜스포트 계층에서 응용 계층과 인터넷 계층 사이의 통신을 담당한다.

TCP와 UDP의 가장 큰 차이점은 데이터 전송의 **신뢰성**에 있다. **TCP는 수신 측의 수신가능 상태, 수신 여부 등을 단계별로 체크해 가며 데이터를 전송하는 반면, UDP는 망으로 데이터를 송신할 뿐 확인 작업을 수행하지 않는다.**

1) TCP(Transmission Control Protocol)
- CRC 체크와 재전송 기능을 통해 신뢰성 있는 전송을 확보한다.
- Flow Control 기능을 수행하여 단계별 데이터 전송 상황을 체크한다.
- 논리적인 1:1 가상 회선을 지원하여 해당 경로로만 데이터가 전달되도록 한다.
- 대표 서비스: FTP, Telnet, Http, SMTP, POP, IMAP 등

2) UDP(User Datagram Protocol)
- 연결되어 있어도 데이터를 송신할 수 있다. 단, 수신 측의 수신 여부는 확인하기 어렵다.
- Flow Control, Error Control을 하지 않아 신뢰성 있는 데이터 전송에는 부적합하다.
- 하나의 송신 정보를 다수의 인원이 수신해야 할 경우 UDP를 사용한다.
- 대표 서비스: SNMP, DNS, TFTP, NFS, NETBIOS, 인터넷 게임/방송/증권 등



3) TCP/UCP의 헤더 구조

- **송/수신자 포트번호**: 송신-수신 프로세스에 할당되는 포트 주소
- **순서 번호**: 송신자가 전하는 데이터 전송 순서
- **응답 번호**: 제대로 수신했는지 여부를 수신자 측으로부터 전달받음.
- **데이터 오프셋**: 헤더의 크기
- **예약 필드**: 다른 사용 목적으로 확보된 필드로 실제 사용 안 함.
- **윈도 크기**: 수신 윈도의 버퍼 크기 지정
- Checksum: 헤더와 데이터의 오류 검출
- **긴급 위치**: 긴급 데이터 처리용
- **제어 비트**: 긴급 필드 설정, 응답 번호 유효 여부 등 체크

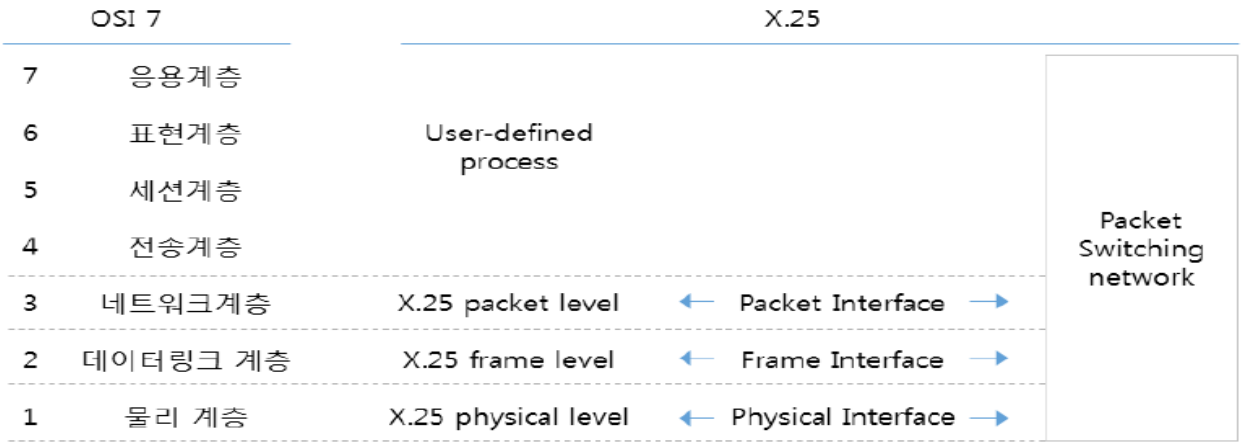
3-3. 네트워크 핵심 알고리즘 파악

1. 패킷 스위칭

WAN을 통해 데이터를 원격지로 송부하기 위해 **X.25, 프레임릴레이 및 ATM**과 같은 다양한 기술들을 필요로 하게 되었다.

1) X.25

전기 통신 국제기구인 ITU-T에서 관리 감독하는 프로토콜이다. X.25는 패킷이라고 불리는 데이터 블록을 사용하여 **대용량의 데이터를 다수의 패킷으로 분리하여 송신하며, 수신 측에서는 다수의 패킷을 결합하여 원래의 데이터로 복원한다.** X.25는 OSI 7계층상의 레이어 중 **1~3계층**까지를 담당하고 있다.



X.25는 데이터 송수신의 신뢰성을 확보하기 위해 **양자 간 통신 연결**을 확립해 나가는 프로세스를 거친다.

2) 프레임릴레이

프레임릴레이는 ISDN을 사용하기 위한 프로토콜로서 ITU-T에 의해 표준으로 작성되었고 다음과 같은 특징이 있다.

- X.25가 고정된 대역폭을 갖는 반면, 프레임릴레이는 사용자의 요청에 따라 **유연한 대역폭**을 할당한다.
- 망의 성능 향상을 위해 **에러 제어 기능과 흐름 제어 기능을 단순화**시켰다.
- X.25가 OSI 7계층 중 1~3계층까지를 담당하는 반면, 프레임릴레이는 **1~2계층**만을 담당한다.
- 전용선을 사용하는 것보다 **가격이 저렴**하며 기술적으로는 X.25에 비해 우위에 있다.

3) ATM(Asynchronous Transfer Mode)

ATM은 **비동기 전송모드**라고 하는 광대역 전송에 쓰이는 스위칭 기법이다. 동기화를 맞추지 않아 보낼 데이터가 없는 사용자의 슬롯은 다른 사람이 사용할 수 있도록 하여 네트워크상의 효율성을 높였다. ATM망은 **연결형 회선이기 때문에 하나의 패킷을 보내 연결을 설정**하게 되고 이후 실데이터 전송이 이루어진다. ATM은 **OSI 7계층과는 다른 고유한 참조 모델**을 가지고 있다.

- **물리 계층**(Physical Layer): 물리적 전송 매체를 다룬다.
- **ATM 계층**: 셀과 셀 전송을 담당한다. 셀의 레이아웃을 정의하고 헤더 필드가 의미하는 것을 알려 준다. 가상 회선의 연결 및 해제, 혼잡 제어도 다룬다.
- **AAL(ATM Adaptation Layer)**: 패킷을 작은 조각인 셀로 전송한 후 다시 조립하여 원래의 데이터로 복원하는 역할을 한다.

2. 서킷 스위칭

패킷 스위칭과 달리 **네트워크 리소스를 특정 사용층이 독점**하도록 하는 것을 서킷 스위칭이라 부른다. 네트워크를 독점적으로 사용하기 때문에 **전송이 보장(Guaranteed)된다는 특징**이 있다. 서킷 스위칭은 서킷을 확보하기 위한 작업을 진행하고 실데이터를 전송하며 서킷을 닫는 프로세스로 진행된다. 이러한 작업이 일어나는 동안 다른 기기들은 해당 경로를 사용할 수 없다.

3. 라우팅 알고리즘

데이터는 송신 측으로부터 수신 측까지 데이터를 전달하는 과정에서 다양한 물리적인 장치들을 거쳐 간다. **목적지까지의 최적 경로를 산출하기 위한 법칙**이 라우팅 알고리즘이다.

1) **거리 벡터 알고리즘(Distance vector algorithm)**
라우터와 라우터 간의 최단 경로 스패닝 트리를 찾고 그 최적 경로를 이용할 수 없을 경우에 다른 경로를 찾는 방식. 각 라우터가 업데이트될 경우마다 전체 라우팅 테이블을 보내라고 요청하지만 수신된 경로 비용 정보는 이웃 라우터에게만 보내진다. 링크 상태 라우팅 알고리즘보다 계산 면에서 단순하다.

2) **링크 상태 알고리즘(Link state algorithm)**
라우터와 라우터 간의 모든 경로를 파악한 뒤 대체 경로를 사전에 마련해 두는 방식이다. 링크 상태 알고리즘을 사용하면 네트워크를 일관성 있게 파악할 수 있으나 거리 벡터 알고리즘에 비하여 계산이 더 복잡하고 트래픽을 광범위한 범위까지 전달해야 한다.

3) 라우팅 프로토콜의 종류

프로토콜	설명
RIP	- 최초의 라우팅 프로토콜 - 거리 벡터 알고리즘 활용 - 30초 주기로 전체 라우팅 정보 갱신 - 변화 업데이트 시 많은 시간 소요 - 라우팅 루프 발생 가능
IGRP	- RIP의 문제점 개선을 위해 시스코에서 개발 - 네트워크 상태를 고려하여 라우팅(대역폭, 속도 등)
OSPF	- 링크 상태 알고리즘 사용 - 발생한 변경 정보에 대해 RIP보다 빠른 업데이트 - 토폴로지에 대한 정보가 전체 라우터에 동일하게 유지
BGP	- 규모가 큰 네트워크의 상호 연결 - 대형 사업자(ISP) 간의 상호 라우팅

4. 서킷 스위칭과 패킷 스위칭 차이

구분	서킷 교환 방식	패킷 교환 방식
의미	- 전송 경로를 설정한 뒤 데이터를 송수신하는 방식	- 데이터의 단위를 보내는 방식
장점	- 경로 접속 시간은 1초 내외로 매우 빠름 - 전송 제어 절차와 형식에 제약을 받지 않음.	- 회선 효율이 우수 - 비동기 전송이 가능 - 연결 설정이 필요 없고 다중 전달이 용이
단점	- 송수신 측 모두 데이터 교환 준비가 완료되어야 함. - 회선이 독점되어 있음.	- 실시간 전송에 부적합 - 네트워크 지연이 발생
활용	- 영상, 비디오 등	- 이메일, 메시지

4. 기본 개발환경 구축하기

4-1. 운영체제 설치 및 운용

1. 운영체제 선택 및 설치

1) 윈도즈 계열 운영체제 선택
윈도즈 계열 운영체제는 개인용, 기업용, 워크스테이션용으로 출시된다.
- Windows Home: 개인 사용자에게 최적화된 운영체제이다.
- Windows Pro: 소규모 기업용으로 최적화된 운영체제이다. Pro는 Home에 비해 관리 및 배포, 도메인가입, 엔터프라이즈 모드, 원격 데스크톱 지원, Hyper V와 같이 향상된 기능을 제공한다.
- Windows Pro for Workstation: 트랜잭션이 몰리거나 복구 기능을 필요로 하는 소규모 기업용 운영체제이다. Pro에 비하여 CPU를 4개까지 동시 지원하여 동시 처리 성능이 뛰어나며 메모리도 6테라바이트까지 지원한다. 또 비휘발성 메모리 모듈을 지원하여 전력이 공급되지 않아도 데이터를 유지할 수 있어 안정성이 향상되었으며 파일 복원을 위해 향상된 ReFs를 지원한다.

2. 리눅스/유닉스 계열 운영체제 선택

오픈소스 기반의 리눅스와 유닉스는 **개발사 및 제공 업체가 다양**하며 리눅스는 Redhat, 페도라, 센트OS와 같은 Redhat 계열과 데비안, 우분투, 칼리, 라즈비안과 같은 데비안 계열 그리고 기타 리눅스 등으로 구분된다.

종류 특징

종류	특징
Debian GNU/Linux	- 개발자 패키지와 매뉴얼이 활성화되어 있어 개발자에게 최적화 되어 있음. - 다운로드: https://www.debian.org/distrib
Ubuntu	- 가장 광범위하게 쓰이는 Linux 운영체제 - 다양한 개발자용 패키지 제공 - Software Center를 통해 응용소프트웨어 공급 - 다운로드: https://www.ubuntu.com/download
openSUSE	- 안정화된 버전(openSUSE)과 테스트 중인 버전(Tumbleweed)을 동시에 공급 - YaST 패키지를 통해 태스크 자동화 지원 - 다운로드: https://software.opensuse.org
Fedora	- 스마트 설정과 업데이트로 사용자 편의성 제공 - 안정화된 운영과 다양한 하드웨어 지원 - 다운로드: https://getfedora.org/en/workstation
CentOS	- 프로그래밍에 최적화된 환경 제공 - RHEL 소스로 컴파일되어 해당 계열의 프로그램 대다수 사용 가능 - 다운로드: https://www.centos.org/download
Slackware	- 다양한 소프트웨어와 그래픽 유저 인터페이스가 미리 설치 - 시스템 관리자를 위한 복구 툴 내장 - 다운로드: http://www.slackware.com

3. 운영체제 운용

외부의 침입이나 바이러스로 인해 시스템이 통제 불능의 상태가 되어 불필요한 리소스를 낭비하거나 중요한 데이터의 유실을 방지하기 위해 다음 사항을 지속 점검하여 운용한다.

1) 서버 운용 기준

- 운용 아키텍처 및 기능 파악
- 네트워크 구성 현황 및 장비 매뉴얼 확보
- 장비 가동 및 중지 매뉴얼 확인
- 백업 주기, 보안 업데이트 주기 설정 및 점검
- 트러블 발생 시 대처 방안 마련

2) 개별 PC용 운영체제 운용 기준

- 정기적인 데이터 백업
- 주기적 보안 업데이트
- 시스템 백업 정례화
- 트러블 발생 시 문의처 정보 확인

4. 윈도 운영체제를 사용하기 위한 목적을 점검한다.

- 개발환경을 구축하기 위해서는 최소 Pro 또는 Workstation급으로 설치해야 한다.
- 마이크로소프트 공식 홈페이지에 접속하여 최신 윈도즈 정보를 파악한 뒤 설치 여부를 판단한다.
- 라이선스 정책을 확인한다.
- 윈도즈 설치 시 이전 데이터가 삭제될 수 있으므로 백업을 하여 데이터 유실을 방지한다.

<표 4-2> 윈도즈 운영체제 선택을 위한 기준

구분	주요 기능
생산성 및 사용자 환경	- 코타나, 잉크, 태블릿 모드, 펜, 터치 등의 주요 기능이 필요한가?
보안	- 윈도즈 정보 보호 기능과 BitLocker 기능이 필요한가?
관리 및 배포	- 그룹 정책이 지원되어야 하는가? - Azure activity directory로 엔터프라이즈 상태 로밍이 돼야 하는가? - 비즈니스용 윈도즈 스토어가 지원되어야 하는가? - 할당된 액세스 지원을 해야 하는가? - 다이내믹 권한 설정이 지원되어야 하는가? - 비지니스용 업데이트가 필요한가? - 공유되는 PC 구성이 지원되어야 하는가?
기본 사항	- 도메인 가입 지원이 필요한가? - Single Sign On 기능 연동이 필요한가? - 엔터프라이즈 모드 지원이 필요한가? - 원격 데스크톱을 통해 지원을 해야 하는가? - 클라이언트 Hype-V 기능이 필요한가?
서버 환경 지원	- 다중의 CPU를 지원해야 하는가? - 대용량 메모리를 지원해야 하는가? - 데이터 안정성이 중요하고 비휘발성 메모리 지원이 필요한가? - 복원 기능이 강력해야 하는가?

4-2. 개발도구 설치 및 운용

1. 프로그래밍 언어

다양한 프로그래밍 언어를 선택하는 것은 시스템 개발 및 운영에 매우 중요하다. 언어를 선택함에 있어 다음 사항 등을 고려한다.
- **언어의 타입**: 정적 개발언어, 동적 개발언어
- **목표 시스템의 특징**: 일반 시스템 또는 도메인 특화 시스템
- **언어 특징**: 객체 지향, 명령형, 서술형, 순서형, 선언형
- **지원**: 관리도구 지원형, 언어 독립형

종류	타입	목적	언어 특징	지원 도구
JAVA	정적	일반	객체 지향, 명령형	존재
C#	정적	일반	객체 지향, 명령형	존재
VB.NET	정적	일반	객체 지향, 명령형	존재
C++	정적	일반	순차적, 명령형	존재하지 않음
Perl	동적	일반	순차적, 명령형	존재
COBOL	정적	일반	순차적, 명령형	존재
SQL	동적	데이터 처리	선언형	존재
ABAP	정적	일반	객체 지향, 명령형	존재
PHP	동적	일반	순차적, 명령형	존재
Python	동적	일반	순차적, 명령형	존재

2. 개발 지원 도구

1) 개발 지원 도구의 종류
SW 개발 과정을 관리하는 오픈소스 기반의 도구를 활용하여 개발 작업의 생산성을 높일 수 있다.

2) 개발환경에 맞는 지원 도구 선정
개발 지원 도구는 개발자가 선택한 개발언어별로 차이를 두고 선택할 수 있다. 오픈소스기반 툴마다 강점과 약점이 명확하며 도구 간 호환성에 차이가 있다. 따라서 필요한 공정을 정의한 뒤 각 툴에 대한 특징을 파악하여 선택하도록 한다.

3. 전자 정부 표준 프레임워크는 다양한 장점을 가지고 있다.

- 공통컴포넌트의 재사용으로 중복 예산 절감
- 표준화된 개발 기반으로 사업자 종속성 해소
- 무상 제공으로 비용 경쟁력 향상
- 표준화된 연계 모듈 활용으로 상호 운영성 향상
- 개발 표준에 의한 모듈화로 유지 보수 용이

4. 버전관리 시스템을 구축하고 활용한다.

형상관리도구는 CVS, SubversionOPEN CM 등이 있다.

이름	설명	라이선스
CVS	버전관리의 대상으로서 각각의 파일을 관리하지만, 프로젝트 전체는 관리할 수 없는 등 초창기 버전관리 도구로서의 여러 제약 사항 내포	Non-GNU
Subversion	제한이 있던 CVS를 대체하기 위해 구현	Subversion License
OpenCM	제한이 있던 CVS를 대체하기 위한 안전하고 완전성 높은 형상관리도구를 표방	GPL

4-3. 응용 시스템 개발 인프라 구축

1. 개발환경 인프라 구축 개요

IT 환경이 급변한 이후 모바일, IOT, 가상현실, 증강 현실, 인공 지능, 블록체인과 같은 다양한 기술들로 인해 개발환경 구성은 매우 복잡해졌다. 따라서 **개발하려는 전체 시스템에 필요로 하는 서비스를 효율적으로 선택하여 개발환경을 구축**해야 한다.

1) 개발환경 인프라 구성 방식
- On-Premise 방식: 외부 인터넷망이 차단된 상태에서 인트라넷 망만을 활용하여 개발환경을 구축하는 방식이다.
- 클라우드 방식: 아마존, 구글, 마이크로소프트 등 클라우드 공급 서비스를 하는 회사들의 서비스를 임대하여 개발환경을 구축하는 방식이다.
- Hybrid 방식: On-Premise와 클라우드 방식을 혼용하는 방식이다.

2) 개발 인프라 환경 고려 사항
- 개발하려는 목표 시스템을 완벽히 이해하고 있는가?
- 로컬 개발환경과 운영환경이 명확히 구분되어 개발 소스가 충돌 나지 않는가?
- 개발 서버는 계층화(Staging)되어 있어 검증되어 안정화된 소스와 개발 중인 불안전한소스가 구분되도록 고려되었는가?
- 서비스의 안정적인 운영을 위해 지속적인 테스트와 신속한 배포가 가능하도록 설계되었는가?
- 운영비가 급격히 상승하여 비용적인 부담이 될 수 있는 가능성이 있는가?
- 서비스의 규모가 커질 경우 확장성이 충분히 고려되어 있는가?
- 오픈소스 활용 시 커뮤니티를 활용하기 위한 환경이 구축되어 있는가?
- 개발에 참여하기 위한 다수의 인원이 참조할 수 있는 표준화가 마련되었는가?

2. 클라우드 기반 개발 인프라 구축

장비 임대, 스토리지 대여뿐만 아니라 개발자 도구 및 생산성 향상을 위한 각종 유틸리티까지 지원한다. 사용자는 서비스의 범위와 사용량에 따른 비용만을 지불하고 서비스 제공자는 이중화된 클라우드 센터를 운영하여 안정적 서비스를 공급한다.

구분	설명
컴퓨팅 환경	프로그램을 설치하고 운용할 하드웨어 장비를 세팅 - 웹 기반 서비스 운용을 위한 웹 서버 - 데이터 관리와 백업 등을 위한 DBMS 서버
스토리지	대규모 데이터를 저장, 관리, 전송, 이중화하기 위한 저장 장치 세팅 - 클라우드 기반 스토리지 세팅 - 데이터 안정성 향상을 위한 아카이브 스토리지 세팅 - 페타 또는 엑사바이트 단위 전송이 필요한 경우 전송 스토리지 세팅
데이터베이스	실데이터를 저장하고 관리하기 위한 데이터베이스 세팅 - 고성능 관계형 데이터베이스 세팅 - 인메모리 데이터베이스 세팅 - 대규모 데이터 운영을 위한 웨어하우징 세팅 - 그래픽, 음성 등 멀티미디어 데이터 처리를 위한 환경 세팅
네트워킹 전송	구성된 서비스, 프로그램, 콘텐츠를 효율적으로 전달하기 위한 환경 세팅 - 콘텐츠 전송용 CDN(Contents delivery network) 구축 - API 제공용 환경 구축 - 대규모 로드 밸런싱 환경 구축
개발자 도구	프로그램을 실제 개발하기 위한 제반 환경 구축 - 애플리케이션 개발 및 배포환경 구축 - 소스코드를 손실 없이 관리하기 위한 repository 구축 - 코드 개발, 테스트를 위한 환경 구축 - 코드 자동 배포, 형상관리를 위한 환경 구축
보안 환경 구축	외부의 침입으로부터 시스템과 데이터를 보호하기 위한 환경 구축 - 사용자 엑세스 및 암호화 관리 - 앱 자격 증명 환경 구축 - 각종 인증서 프로비저닝, 관리 및 배포 - 악성 트래픽 필터링 서비스 관리 - 해킹 공격(DDos 등)으로부터의 방어 환경 구축
응용 기술 세팅	증강 현실, 가상현실 개발을 위한 환경 세팅 머신러닝, 딥러닝 등 AI 개발환경을 위한 환경 세팅 사물 인터넷, 게임 등 개발을 위한 환경 세팅
생산성 향상	시스템 볼륨 자동 확장 환경 구축 실시간 스트리밍 서비스 환경 구축 비즈니스, 운영 상태 분석 서비스 구축