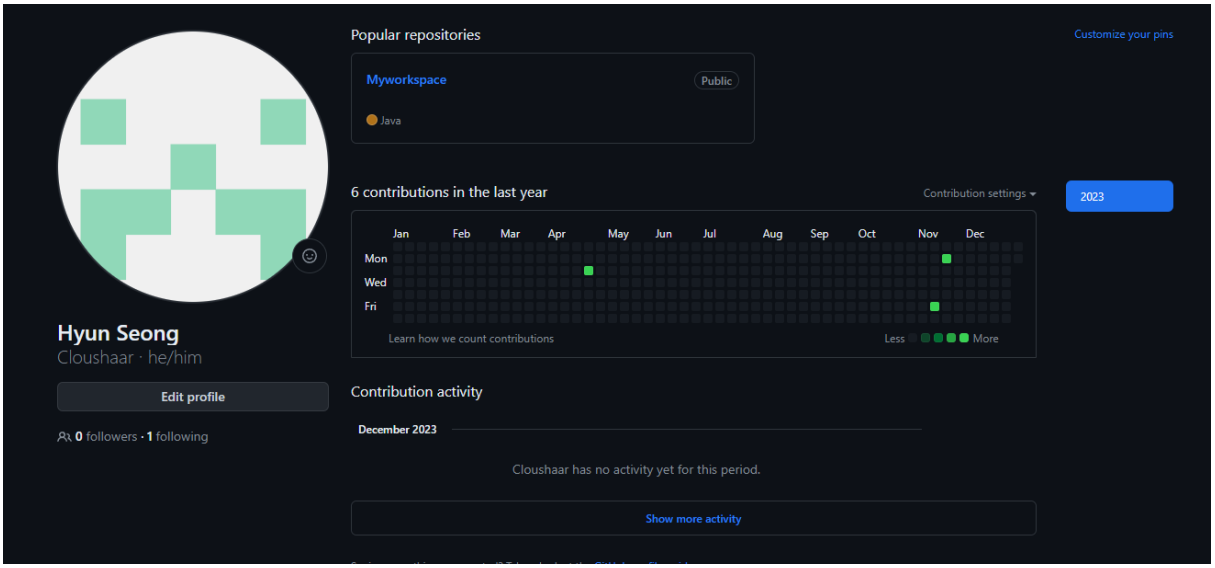


포트폴리오

현성

> Seoul, South Korea

About Me

현재 세명컴퓨터 고등학교에 재학 중인 현성 입니다.

Works

- JAVA
- © C언어
- 범용레지스터 8개
- 리버싱 엔지니어링(기초)
- 웹보안(NATAs)
- bandit
- TypeScript
- 어셈블리어
- Packet Tracer
- 네트워크
- Python
- Python 라이브러리
- 컴퓨터 구조 개론
- 컴퓨터 개론
- DB
- SQL모음

📘 Null4U책 만들기

⛅ 클라우드

🔎 과제

🌏 DNS 작동원리

AWS

Projects

담당한 프로젝트를 최신 순으로 구성했습니다.

프로젝트

Aa Name	# 기여도	작업 영역	작업기간
S3정적 호스팅	100%	AWS CSS HTML JS TS	@2023년 6월 1일 → 2023년 6월 27일
TypeScript로 BlockChain 구현	100%	TS	@2023년 11월 6일 → 2023년 11월 13일
Bastion Hosting	100%	AWS	@2023년 12월 21일 → 2023년 12월 21일

Skills

Linux

🟪🟪🟪⬜⬜

AWS

🟪🟪🟪⬜⬜

Assembly

🟪⬜⬜⬜⬜

C

🟪🟪🟪⬜⬜

Java

Python

JavaScript

TypeScript

Html

CSS

SQL

Thank you

To the Top Contact

네트워크

▼ NAT(Network Address Translation)

네트워크 주소 변환

사설IP를 공인 IP로 변환 하거나 반대로 해줌

NAT를 사용하면 보안성이 증가함

IP주소 부족에 대응가능 사설Ip를 공인 Ip로 바꾸어 외부와 통신함

트래픽관리 가능 하나의 공인Ip

내부 네트워크의 구조를 감추어 외부에서 직접 접근하기 어렵게 만듬

+@ IP주소 “만” 가지고 구분하면 모호성이 증가함

▼ DHCP 설명

DHCP의 주요 기능:

1. **IP 주소 할당:** DHCP는 네트워크에 연결된 기기에 동적으로 IP 주소를 할당합니다. 이는 중복된 IP 주소를 방지하고 네트워크 구성을 단순화합니다.
2. **서브넷 마스크 및 기본 게이트웨이 설정:** DHCP는 IP 주소 외에도 서브넷 마스크, 기본 게이트웨이 및 DNS 서버와 같은 네트워크 설정 정보를 할당합니다.
3. **IP 주소 재사용 및 갱신:** DHCP는 기기가 네트워크를 떠날 때 해당 IP 주소를 풀(Pool)에 반환하여 다른 기기에 할당할 수 있도록 합니다. 또한 기기가 네트워크에 계속해서 남아 있을 때 주기적으로 IP 주소를 갱신합니다.
4. **중앙 관리:** DHCP는 네트워크 관리자가 중앙에서 IP 주소 및 구성 정보를 관리할 수 있도록 합니다.

DHCP 프로세스

1. **DHCP Discover:** 클라이언트는 네트워크에 연결되면서 DHCP 서버를 찾기 위해 DHCP Discover 메시지를 브로드캐스트합니다. 이 메시지에는 클라이언트의 MAC 주소 및 기타 정보가 포함되어 있습니다.
2. **DHCP Offer:** DHCP 서버는 DHCP Discover 메시지를 수신하면 사용 가능한 IP 주소 중 하나를 할당하고 해당 정보를 DHCP Offer 메시지로 클라이언트에게 전송합니다. 이 메시지에는 제안된 IP 주소 및 서브넷 마스크, 기본 게이트웨이 및 DNS 서버 등의 설정이 포함됩니다.
3. **DHCP Request:** 클라이언트는 여러 DHCP Offer 중 하나를 선택하고 해당 서버

에게 DHCP Request 메시지를 보냅니다. 이 메시지에는 클라이언트가 선택한 제안된 IP 주소가 포함됩니다.

4. **DHCP Acknowledgment:** 선택한 DHCP 서버는 DHCP Request를 수신하면 해당 IP 주소를 클라이언트에게 할당하고 DHCP Acknowledgment 메시지를 보냅니다. 이 메시지에는 최종적으로 할당된 IP 주소 및 다른 네트워크 설정이 포함됩니다.

Lease Time

Lease Time은 DHCP서버로 부터 IP주소를 빌려서 쓸 수 있는 임대 기간

▼ TCP(Transmission Control Protocol)

연결 지향 (Connection-Oriented) → 연결할 때 3way HandShake(syn → syn ack → ack)

끊을 때 4way HandShake(fin → ack → fin → ack)

신뢰성이 있는 데이터 수송신 방식

가상회선 방식

순서 보장

흐름 제어 및 혼잡 제어 없음

▼ UDP (User Datagram Protocol)

간단한 데이터 + 빠른 전송속도

비연결성(**Connectionless) 전송방식**

신뢰성이 낮음

순서 보장 없음

흐름 제어 및 혼잡 제어 없음

▼ CIDR SubNetMask 설명

CIDR은 Classless Inter-Domain Routing의 약자

이름 그대로 클래스 없는 도메인간의 라우팅 기법을 뜻함

이는 도메인간의 라우팅에 사용되는 인터넷 주소를 원래 IP주소 클래스 체계를 쓰는 것보다 더욱 능동적의로 할수 잇도록 할당하여 지정하는 방식중 하나임

SubNetMask는 IP주소에서 네트워크ID와 호스트ID를 구분하는 용

▼ CIDR 계산법

ex)

192.168.10.0/24 라는 Ip주소가 있을 때 이 Ip주소에서 사용가능한 IP주소의 범위를 구

▼ 서브넷 서브넷팅 슈퍼넷팅

서브넷(subnet)은 하나의 네트워크가 분할되어 나누어진 작은 네트워크임

서브네팅(Subnetting)

서브넷을 만들기위해 네트워크를 분할하는 것
서브네팅을 하면 IP 할당 범위를 더 작은 단위로 쪼갤수 있게 됨
그래서 만일 ip가 100개만 필요하다면 C클래스(256개)를 더 쪼개서 줄 수 있음

수퍼넷팅(**Supernetting)**

슈퍼넷팅은 서브넷팅의 반대임 작은 네트워크 여러개를 하나의 큰 네트워크로 만드는 것

슈퍼넷팅를 하면 라우팅 테이블에서 다뤄야할 정보를 줄여서 메모리, CPU등의 자원 낭비를

하면 막을 수 있음

/24는 1이 24개라는 뜻 32 - 24 = 8 0은 8개

192 . 168 .10 .0

11111111 11111111 11111111 00000000 임

뒤에 있는 0만 사용가능 하기 때문에 범위를 구하면

192.168.10.0 ~ 192.168.10.255 까지 인데 첫 IP와 마지막 IP는 각각 네트워크 주소와 브로드캐스트 주소 라서 사용 불가 그래서 개수는 2 ^ 8 - 2 = 254개임

근데 aws에서의 사용가능 개수는 다름 이유는 AWS에서는 따로 자체 클라우드에서 설정해 사용하고있는 IP가 있기 때문에 5개를 제외 해야 함

10.0.0.0/24을 예시로 설명

10.0.0.0은 네트워크 주소

10.0.0.1은 AWS에서 VPC 라우터용으로 예약 (Default Gateway)

10.0.0.2는 DNS 서버 주소 DNS 서버의 IP 주소는 기본 VPC 네트워크 범위에 2를 더한 주소이다. CIDR 블록이 여러 개인 VPC의 경우, DNS 서버의 IP 주소가 기본 CIDR에 위치하게 된다.

10.0.0.3은 AWS에서 앞으로 사용하려고 예약한 주소

10.0.0.255는 네트워크 브로드캐스트 주소

그래서 사용가능 개수는 2^8-5=251

▼ CIDR블럭

CIDR블럭은 서브넷 이다.

그냥 AWS에서 서브넷을 CIDR블록이라고 멋들여지게 부르는 것

예를 들어 192.169.0.0/16인 대역망이 있을 때 이를 세개의 네트워크 단위(서브넷)로 쪼갠다.

그럼 쪼개진 서브넷1(192.169.1.0/24), 서브넷2(192.169.2.0/24), 서브넷3(192.169.3.0/24)이 CIDR블럭임

▼ 라우팅

▼ 라우팅 테이블

어디로 보낼지를 결정하는 테이블

▼ 포워딩 테이블

테이터 프레임을 어디로 보내야 할지 정하는 테이블

▼ 지연

D1.전송지연

(패킷이 날아가기 전) 패킷을 만들고, 제조 해서 쏘기 직전까지 걸리는 시간

D2.전파지연

대상 목적지 노드까지 날아가는데 걸리는 시간

D3.처리지연

패킷을 받은 수신자 노드가, 데이터를 처리하는 데 걸리는 시간

D4.큐(Queue)잉지연

대기 큐에서 기다리는 시간 - 입력지연 + 출력지연

Total Delay = (n + 1) * (D1 + D2 + D3) + n*(D4)

▼ 패킷화

패킷으로 만들기 전의 단위: 프레임.
26byte ~ 1528byte (헤더 포함)

▼ IP 주소

전자기기 1개당 IP번호 1개로 만들었었다.

IP주소 → 32bit

약42억개의 전자기기를 구분할 수 있다.

1계층 - 물리 → 비트단위 → 주소없음 L1(리피터, 허브)

2계층 - 데이터링크 → 프레임 → L2스위치 보다 같거나 높은 단위들

3계층 - 네트워크 → 패킷 → IP주소(전자기기 단위 PC/라우터 이상)

4계층 - 전송 → 세그먼트 → Port번호(L4스위치 PC들도 다 역할을 함) → 프로세스 구분(여러 앱을 구분)

클래스 기반의 주소지정의 대시작.

불합리 하고 비효율적인 IP분배

하면서 이후 등장한 것 CIDR

▼ 1.사설 IP 주소

사설IP주소 → NAT(Network Adress Translation)

사설 IP대역

- 10.0.0.0 ~ 10.255.255.255(10.0.0.0/8)
- 172.16.0.0 ~ 172.31.255.255(172.16.0.0/12)
- 192.168.0.0 ~ 192.168.255.255(192.168.0.0/16)

사설 IP주소를 사용함으로써, 외부로 나가는 공인IP 하나로 3*2^24개의 사설 IP를 사용할 수 있게 됨

2.DHCP (IP 동적 할당 프로토콜)

ARP RARP

클래스 없는 주소지정 1996년 인터넷 협회 발표

클래스 없는 인터도메인 라우팅(CIDR Classless Inter Domain Routing)등장

동시에 등장한, 서브넷 마스크, 슬래시 표기법 등장

네트워크 주소가 같은 장치가 몇개까지 있을 수 있는 가

2 ^ 5 → 32개

컴퓨터 구조 개론

▼ 1. CPU 구성요소

CU-Control Unit

ALU-Arithmetic Logical Unit

REG-Register

▼ 2.Virtual Memory (가상 메모리)

일부만 넣고 다 넣은 척, 메모리 있는 척

1. **물리적 메모리 한계 극복:** 컴퓨터의 물리적인 메모리(RAM)는 제한된 크기를 가집니다. 가상 메모리는 이 한계를 극복하기 위해 사용됩니다. 물리적 메모리 공간이 부족한 경우에도 프로그램은 가상 메모리를 활용하여 추가적인 메모리 공간을 확보할 수 있습니다.
2. **주소 공간 확장:** 가상 메모리는 각 프로세스에게 논리적인 주소 공간을 제공합니다. 이 논리적인 주소 공간은 실제 물리적 메모리 주소와는 다릅니다. 프로세스는 이 가상 주소를 사용하여 데이터를 읽고 쓰며, 운영체제가 이를 실제 메모리 주소로 매핑합니다.
3. **페이지 파일:** 운영체제는 물리적 메모리에 현재 실행 중인 프로세스의 필요한 부분만을 유지하고, 나머지는 보조 저장 장치(일반적으로 하드 디스크)에 저장합니다. 이를 페이지 파일 또는 스왑 파일이라고 합니다. 필요한 데이터를 물리 메모리로 로드하고, 필요 없는 데이터는 페이지 파일로 다시 스왑하는 과정을 통해 가상 메모리가 작동합니다.
4. **메모리 관리:** 운영체제는 가상 메모리와 물리 메모리 간의 매핑을 관리하며, 필요에 따라 페이지를 로드하거나 스왑합니다. 이로써 여러 프로세스가 동시에 실행될 때 메모리 공간을 효율적으로 활용할 수 있습니다.

Paging방식 VS Segment방식

Paging 방식

물리적으로 다 잘라놨음. 프로그램을 페이지 단위로 할당함.

- Paging은 메모리를 일정한 크기의 페이지(또는 프레임)로 나누는 방식입니다. 프로세스는 페이지 단위로 분할되며, 물리적 메모리는 같은 크기의 페이지 프레임으로 나눕니다.
- 프로세스의 논리적 주소 공간은 페이지 번호와 오프셋(offset)으로 표시됩니다.

Segment방식

프로그램 단위로 잘라놈.

- Segment는 논리적으로 관련된 데이터 덩어리인 세그먼트로 프로세스를 분할하는 방식입니다. 각 세그먼트는 다른 크기와 형태를 가질 수 있으며, 논리적 주소 공간 내에서 명명된 세그먼트로 참조됩니다.
- 논리적 주소는 세그먼트 번호와 오프셋으로 표시됩니다.

- 주요 장점 중 하나는 메모리 할당과 해제가 간단하다는 것입니다. 논리적 주소 공간의 페이지를 물리적 페이지 프레임에 매핑하는 페이지 테이블을 사용하여 이러한 할당 및 해제가 관리됩니다.
- Paging은 내부 단편화 문제가 발생하지 않습니다.
- 세그먼트 방식은 프로세스의 논리적 구조를 잘 반영하므로 코드, 데이터, 스택 등의 세그먼트를 분리하여 관리할 수 있습니다.
- 하나의 세그먼트의 크기가 다를 수 있기 때문에 외부 단편화 문제가 발생할 수 있습니다. 이를 해결하기 위해 세그먼트 테이블이 사용됩니다.

내부단편화 VS 외부단편화

▼ 내부 단편화(Internal Fragmentation)

- 내부 단편화는 주로 고정 크기의 할당 단위(예: 페이지, 블록)를 사용하는 메모리 할당에서 발생합니다.
- 프로그램이나 데이터가 메모리에 저장될 때, 할당된 메모리 공간이 필요한 프로그램이나 데이터보다 큰 경우에 내부 단편화가 발생합니다.
- 이로 인해 메모리 공간이 낭비되며, 할당된 공간 중에서 사용되지 않는 부분이 발생합니다.
- 내부 단편화를 최소화하기 위해서는 메모리 할당 시에 최대한 큰 메모리 블록을 사용하거나, 동적 크기 조정 알고리즘을 사용하여 메모리를 할당하는 것이 중요합니다.

▼ 외부 단편화(External Fragmentation)

- 외부 단편화는 가변 크기 할당 단위(동적 할당)를 사용하는 메모리 할당에서 발생합니다.
- 프로그램이나 데이터가 해제되면, 해당 공간은 다시 사용 가능한 상태가 됩니다.
- 그러나 해제된 메모리 공간이 작거나 연속적이지 않아 다른 프로그램이나 데이터를 할당하기에 부적합한 크기나 위치에 있을 때 외부 단편화가 발생합니다.
- 외부 단편화를 해결하기 위해서는 메모리 할당과 해제를 효율적으로 관리하고, 조각화된 공간을 다시 하나로 합치는 메커니즘(예: 메모리 압축, 가상 메모리)을 사용할 수 있습니다.

캐시 메모리의 구현 방식

▼ 직접 사상

1. **장점**:
    - 간단하고 저렴한 하드웨어 구조: 직접 사상은 캐시 메모리를 관리하기 위한 하드웨어가 단순하며 비용이 적게 듭니다.
    - 메모리 관리가 효율적: 각 메모리 블록은 캐시의 특정 슬롯에 고정적으로 매핑되므로 메모리 블록을 찾는 데에 걸리는 시간이 예측 가능합니다.

▼ 연관 사상

1. **장점**:
    - 충돌 없음: 연관 사상에서는 메모리 블록이 어떤 캐시 슬롯에든 매핑될 수 있으므로 충돌이 발생하지 않습니다.
    - 높은 효율성: 메모리 블록을 찾는 데에 걸리는 시간이 직접 사상에 비해 예측하기 어렵지만, 대부분의 상황에서 더 빠를 수 있습니다.

2. **단점**:
    - 충돌 발생 가능성: 서로 다른 메모리 블록이 동일한 캐시 슬롯에 매핑될 수 있으므로 충돌이 발생할 가능성이 높습니다.
    - 공간 효율성: 캐시의 일부 슬롯은 항상 사용되지 않을 수 있으며, 이로 인해 공간 효율성이 떨어질 수 있습니다.
    - 공간 활용: 캐시 슬롯 중 사용되지 않는 슬롯이 있어도 메모리 블록을 저장할 수 있어 공간을 더 효과적으로 활용할 수 있습니다.
2. **단점**:
    - 복잡한 하드웨어: 연관 사상을 구현하기 위한 하드웨어가 복잡하며, 이로 인해 더 많은 비용과 전력 소비를 초래할 수 있습니다.

JAVA

업 캐스팅 다운 캐스팅

▼ 업 캐스팅 설명

업 캐스팅은 하위 클래스(자식 클래스) 객체를 상위 클래스(부모 클래스) 타입으로 변환하는 것을 의미합니다.

업 캐스팅은 자동으로 이루어지며, 명시적인 형 변환 연산자가 필요하지 않습니다.

이것은 다형성(polymorphism)을 구현하는 데 사용됩니다. 즉, 상위 클래스의 참조 변수를 사용하여 하위 클래스 객체에 접근할 수 있습니다.

▼ 다운 캐스팅 설명

다운 캐스팅은 상위 클래스(부모 클래스) 타입으로 변환된 객체를 다시 하위 클래스(자식 클래스) 타입으로 변환하는 것을 의미합니다.

다운 캐스팅은 명시적인 형 변환 연산자(캐스트 연산자)를 사용하여 수행해야 합니다.

다운 캐스팅은 주로 업 캐스팅된 객체를 원래 자식 클래스로 되돌리고, 자식 클래스에서 추가된 메서드나 속성에 접근하기 위해 사용됩니다.

▼ 예제

```
class Car{
    String name = "차";
    String print(){
        return "나는" + name + "이다";
    }
}

class Bus extends Car{
    String name = "버스";
    String print(){
        return "차의 종류는" + name + "이다";
    }

    void busMethod(){
        System.out.println("버스에만 있는 메소드 이다.");
    }
}

class Truck extends Car{
    String name = "트럭";
```

```
    String pritnt(){
        return "차의 종류는 "+ name + "이다.";
    }
}

public class CastingExam
{
    public static void main(String[] args) {
        Car c1 = new Car();
        System.out.printf("c1.pritnt(): %S\n", c1.print());
        Car c2 = new Bus(); //upcasting
        System.out.printf("c2.pritnt(): %S\n", c2.print());
        Bus b1 = (Bus)c2; //downcasting
        //c2.busMethod(); 이 부분 에러 이유는 밑에 서술
        b1.busMethod();

        Car[] c = new Car[3];
        c[0] = new Car();
        c[1] = new Bus();
        c[2] = new Truck();
        System.out.println(c[0].print());
        System.out.println(c[1].print());
        System.out.println(c[2].print());
    }
}


//이 예제에서 업캐스팅을 쉽게 설명 하자면 자동차에 버스 모양의 박스를 덮어 놓은거 라서
```

this

▼ this란?

현재 클래스의 인스턴스를 의미함

즉, 현재 클래스의 멤버변수를 지정할때 사용합니다.

멤버변수?

메소드 밖에 선언된 변수

this 사용예제

```
class Person {
    // 인스턴스 변수
    String name;
    int age;

    // 생성자
    public Person(String name, int age) {
        this.name = name; // 현재 객체의 name에 매개변수로 전달된 name 값을 할
        this.age = age;   // 현재 객체의 age에 매개변수로 전달된 age 값을 할당
    }

    // 메서드
    public void introduce() {
        System.out.println("안녕, 나는 " + this.name + "이고, " + this.ag
    }
}

public class Main {
    public static void main(String[] args) {
        // Person 클래스의 객체 생성
        Person person1 = new Person("Alice", 25);

        // introduce 메서드 호출
        person1.introduce();

        // 또 다른 객체 생성
        Person person2 = new Person("Bob", 30);
        person2.introduce();
    }
}
```

▼ this()란?

현재 클래스에 정의된 생성자를 부를때 사용됩니다.

this() 사용예제

```
class Person {
    String name;
    int age;

    // 첫 번째 생성자
    public Person() {
```

```
        this("Unknown", 0); // 다른 생성자 호출
    }

    // 두 번째 생성자
    public Person(String name) {
        this(name, 0); // 다른 생성자 호출
    }

    // 세 번째 생성자
    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }

    public void introduce() {
        System.out.println("안녕, 나는 " + this.name + "이고, " + this.ag
    }
}

public class Main {
    public static void main(String[] args) {
        Person person1 = new Person(); // 첫 번째 생성자 호출
        person1.introduce();

        Person person2 = new Person("Alice"); // 두 번째 생성자 호출
        person2.introduce();

        Person person3 = new Person("Bob", 30); // 세 번째 생성자 호출
        person3.introduce();
    }
}
```

위의 코드에서 `this()`를 사용하여 다른 생성자를 호출하고 있습니다. 이렇게 하면 생성자 간의 중복 코드를 피할 수 있으며, 다양한 매개변수 조합을 가진 객체를 생성할 수 있습니다.

---

super

▼ super란?

자식 클래스에서 상속받은 부모 클래스의 멤버 변수를 참조할 사용합니다.

super 사용예제

```
class Animal {
    String name;
```

```
    public Animal(String name) {
        this.name = name;
    }

    public void makeSound() {
        System.out.println("동물이 소리를 냅니다.");
    }
}

class Dog extends Animal {
    String breed;

    public Dog(String name, String breed) {
        super(name); // 부모 클래스의 생성자 호출
        this.breed = breed;
    }

    @Override
    public void makeSound() {
        System.out.println("개가 짖습니다.");
    }

    public void printInfo() {
        System.out.println("이 개의 이름은 " + super.name + "이고, 종류는 "
    }
}

public class Main {
    public static void main(String[] args) {
        Dog myDog = new Dog("멍멍이", "골든 리트리버");
        myDog.makeSound(); // 하위 클래스의 메서드 호출
        myDog.printInfo(); // 부모 클래스의 멤버에 접근하기 위해 super 사용
    }
}
```

▼ super()란?

자식 클래스가 자신을 생성할 때 부모 클래스의 생성자를 불러 초기화 할때 사용됩니다.

(기본적으로 자식 클래스의 생성자에 추가됩니다.)

오버라이딩(Overriding)

▼ Overriding이란?

객체 지향 프로그래밍에서 자식 클래스가 부모 클래스의 메서드를 다시 정의하고 구현하는 것

Overriding장점

오버라이딩은 다형성(polymorphism)을 구현할 수 있다는 장점이 있다.

*다형성

다형성(Polymorphism)은 객체 지향 프로그래밍(OOP)의 중요한 개념 중 하나로, 한 가지 기능 또는 인터페이스를 여러 다른 방식으로 구현할 수 있는 능력을 나타냅니다.

Overriding을 수행하기 위한 규칙

1. 메서드 이름, 매개변수 목록(시그니처) 및 반환 유형은 오버라이드할 부모 클래스의 메서드와 정확히 일치해야 합니다.
2. 접근 제어자는 부모 클래스의 메서드와 동일하거나 더 넓은 범위(더 접근 가능한)로 변경할 수 있습니다. 예를 들어, 부모 클래스의 메서드가 `public`이면 하위 클래스에서 `public` 또는 `protected`로 오버라이드할 수 있습니다.
3. 예외 처리(throws)는 부모 클래스의 메서드보다 좁은 범위로 변경할 수 있습니다.

Overriding예제

```
class Animal {
    public void makeSound() {
        System.out.println("동물이 소리를 냅니다.");
    }
}

class Dog extends Animal {
    @Override
    public void makeSound() {
        System.out.println("개가 짖습니다.");
    }
}

public class Main {
    public static void main(String[] args) {
        Animal animal = new Dog(); // 다형성을 활용한 객체 생성
        animal.makeSound(); // 하위 클래스의 메서드가 호출됨
    }
}
```

추상 클래스

▼ 추상화란?

프로그래밍에서의 추상화는 복잡한 시스템이나 개념을 간소화하거나 특정 측면에 집중하기 위해 세부 사항을 감추는 프로세스입니다.

추상화는 캡슐화와 비슷해보이지만 둘은 서로 다른 측면을 강조 하고 있습니다.

추상화는 개념을 단순화하고 이해하기 쉽게 만드는 것에 중점을 두며, 캡슐화는 객체의 상태와 행동을 보호하고 외부에 노출되는 것을 제한하여 시스템을 더 안전하게 만드는 데 중점을 둡니다.

▼ 추상 클래스란?

상속관계에 있는 클래스 중에서 상위 클래스에서는 특별한 구현 없이 사용하고자 하는 메소드만 기술하고 구체적인 구현은 하위 클래스에서 하도록 할 수 있다.

이때 사용하는 것이 추상클래스로 구체적인 내용 기술 없이 모양만 갖춘 클래스이다.

추상 클래스는 abstract 키워드를 사용하여 표시하고 추상 클래스 안에는 추상 메소드를 가진다

추상 클래스(abstract class)는 추상 메소드를 가진 클래스로 추상 클래스는 new 연산자로 객체 생성할 수 없다.

따라서 추상 클래스의 추상 메소드는 자신이 직접 이용하지 못하고 반드시 하위 클래스에서 이 메소드를 상속받아 구현하여 사용해야 한다.

추상 클래스의 추상 메소드는 반드시 오버라이딩 되어야 하기 때문에 하위 클래스들이 특정 메소드를 반드시 구현하도록 강제할 수 있고, 만약 추상 메소드를 오버라이딩하지 않으면 상속받는 클래스는 자동으로 추상 클래스가 된다.

▼ 예제

```
abstract class ShapeExam{
    public abstract double getArea();
    public abstract double getCircum();
}

class Rect extends ShapeExam{
    int width;
    int height;
    public double getArea(){ return width * height;}
    public double getCircum(){ return 2*(width+height);}
}

class Circle extends ShapeExam{
    int r;
```

```
    public double getArea(){ return 3.14*r*r;}
    public double getCircum(){ return 3.14*2*r;}
}
public class ShapeUser
{
    public static void main(String[] args) {
        Rect aa = new Rect();
        Circle bb = new Circle();
        aa.width = 10;
        aa.height = 10;
        bb.r = 10;

        System.out.printf("사각형의 넓이는 %.1f입니다.\n", aa.getArea());
        System.out.printf("사각형의 둘레는 %.1f입니다.\n", aa.getCircum())
        System.out.printf("원의 넓이는 %.1f입니다.\n", bb.getArea());
        System.out.printf("원의 둘레는 %.1f입니다.\n", bb.getCircum());
    }
}
```

인터페이스

▼ 인터페이스란?

자바는 다중상속이 불가능 하지만 유사한 기능의 인터페이스가 존재한다.

인터페이스는 모든 메소드가 추상 메소드이며, 데이터는 final static 변수만을 가지는 특별한 형태의 클래스이다.

인터페이스는 클래스가 아니기 때문에 객체가 가질 수 있는 메소드 이름만을 명시하고 실제 기능은인터페이스를 상속받아 구현하는 클래스에서 기술해야 한다.

인터페이스를 적용한 클래스는 인터페이스에서 선언된 메소드를 모두 구현해 주어야 한다. (해당 메소드를 구현하지 않은 클래스는 추상 클래스가 된다.)

인터페이스는 클래스가 아니고 상속받을때는 extends가 아닌 implements를 사용

▼ 예제

```
interface Sound{
    public void soundUp(int level);
    public void soundDown(int level);
}
```

```
class TV implements Sound{
    private int sndLevel;
    public TV(){
        sndLevel = 0;
    }
    public void soundUp(int level){
        sndLevel += level;
        System.out.printf("현재의 TV 소리크기 %d\n", sndLevel);
    }
    public void soundDown(int level){
        sndLevel -= level;
        if(sndLevel < 0){
            sndLevel = 0;
        }
        System.out.printf("현재의 TV 소리크기 %d\n", sndLevel);
    }
}

class Radio implements Sound{
    private int sndLevel;
    public Radio(){
        sndLevel = 0;
    }
        public void soundUp(int level){
        sndLevel += level;
        System.out.printf("현재의 라디오 소리크기 %d\n", sndLevel);
                if(sndLevel > 100){
            sndLevel = 100;
        }
    }
    public void soundDown(int level){
        sndLevel -= level;
        if(sndLevel < 0){
            sndLevel = 0;
        }
        System.out.printf("현재의 라디오 소리크기 %d\n", sndLevel);
    }
}

public class SoundExam
{
    public static void main(String[] args) {
        Sound radio = new Radio();
        Sound tv = new TV();
        radio.soundUp(5);
```

```
        tv.soundUp(5);
    }
}
```

```
interface Sound{
    public void soundUp();
    public void soundDown();
    public void soundZero();
}

interface Channel{
    public void channelUp();
    public void channelDown();
    public void channelGo(int ch);
}

class TV implements Sound, Channel{
    private int sndLevel;
    private int channel;
    public TV(){
        sndLevel = 0;
        channel = 0;
    }
    public void soundUp(){
        sndLevel += 1;
        if(sndLevel > 100){
            sndLevel = 100;
        }
        System.out.printf("현재의 TV 소리크기 %d\n", sndLevel);
    }
    public void soundDown(){
        sndLevel -= 1;
        if(sndLevel < 0){
            sndLevel = 0;
        }
        System.out.printf("현재의 TV 소리크기 %d\n", sndLevel);
    }
    public void soundZero(){
        sndLevel = 0;
        System.out.println("TV 소리의 크기는 0");
    }
    public void channelUp(){
        channel += 1;
        if(channel > 1000){
            channel = 0;
```

```
        }
        System.out.printf("현재의 TV의 채널은 %d\n", channel);

    }
    public void channelDown(){
        channel -= 1;
        if(channel < 0){
            channel = 1000;
        }
        System.out.printf("현재의 TV의 채널은 %d\n", channel);
    }
    public void channelGo(int ch){
        if(ch>1000 || ch<0){
            System.out.printf("현재의 TV의 채널은 %d\n", channel);
        }
        else{
            channel = ch;
            System.out.printf("현재의 TV의 채널은 %d\n", channel);
        }
    }
}

public class TVExam
{
    public static void main(String[] args) {
        TV tv = new TV();
        tv.soundUp();
        tv.soundUp();
        tv.soundDown();
        tv.soundUp();
        tv.soundZero();
        tv.channelUp();
        tv.channelDown();
        tv.channelDown();
    }
}
```

라이브러리

▼ import

```
import java.~.~
//다른언어에서 쓰는 방법이랑 같음 뒤어 ~에는 *이걸 사용하면 해당 패키지의 모든걸 impo
```

▼ java.lang.string

equals()

문자열의 내용이 같은지를 비교하는 메소드

객체 내의 문자열이 같으면 true를 반환하고 거짓이면 false를 반환하는 메소드

```
class EqualExam
{
    public static void main(String[] args) {
        String s1 = "aaa";
        String s2 = "aaa";
        String s3 = new String("bbb");
        String s4 = new String("bbb");

        if(s1==s2) System.out.println("주소가 같다.");
            else System.out.println("주소가 다르다.");
        if(s3==s4) System.out.println("주소가 같다.");
            else System.out.println("주소가 다르다.");
        if(s1.equals(s2)) System.out.println("내용 같다.");
            else System.out.println("내용 다르다.");
        if(s3.equals(s4)) System.out.println("내용 같다.");
            else System.out.println("내용 다르다.");
    }
}
//new String으로 생성한 s3와 s4는 메모리를 따로 할당 받아서 주소가 다름
//반면 s1과 s2는 같은 주소값을 갖고있음
```

trim()

고백제거하는 메소드 DB에서 많이씀 커뮤니티를 만들때 제목 앞뒤로 공백이 있으면 없에고 추가함

substring()

문자열 일부를 반환하는 메소드

valueOf()

매개 변수의 값을 String 형으로 변환하는 메소드

매개 변수에는 boolean, char, char[``], double, float, int, long, Object 등이 올 수 있음

charAt()

문자열의 특정 위치의 문자를 반환하는 메소드

length()

String 클래스의 문자열의 길이를 반환하는 메소드

StringBuffer 클래스의 length() 메소드와 같이 클래스 객체에 저장된 문자열의 길이를 반환하며, 이때 공백도 하나의 문자로 인식함

▼ java.lang.math

Math 클래스의 모든 메소드는 static 형으로 정의되어 있어 상속이나 오버라이딩
할 수 없는 메소드들임

Math 클래스는 생성자를 제공하지 않아 객체를 생성할 수 없음

객체 생성 없이 클래스 이름으로 멤버 변수나 메소드를 접근함

abs()

절대값을 반환하는 메소드

round()

매개 변수 값과 같거나 가장 가까운 정수를 반환하는 메소드 (반올림)

pow()

거듭제곱 값을 반환하는 메소드

sqrt()

제곱근을 반환하는 메소드(루트)

random()

0.0보다 크거나 같고 1.0보다 작은 난수를 반환하는 메소드

Python 라이브러리

list

```
#list 값의 평균 구하기
def list_avg(my_list, a, b):
    return sum(mylist) / len(mylist)
```

```
s#리스트 압축
#짝수 만으로 리스트 만들기
myList = [1,2,3,4,5,6,7,8,9,10]
even = []
for i in myList:
if i&2 ==0:
even.append(i)
print(even)

#한줄로 요약
even = [i for i in myList if i&2==0]
print(even)

#5보다 작은 정수
small = [i for i in myList if i<5]
print(small)

#변수값 가공도 가능
```

```
small = [i+10 for i in myList if i<5]
print(small, "변수 가공")
```

문자열 다루기

1. 필요성

   텍스트 분석

   크롤링

   엑셀파일 및 데이터 다루기

```
#문자열 길이
a = 'apple'
b = 'apple watch!'
print(len(a))
print(len(b))
```

```
#한글
a = '사과'
b = '애플 와치~'
print(len(a))
print(len(b))
```

```
#문자열 쪼개기
a = 'This is an apple'
words = a.split(' ') #list 형태로 반환
print(words)

words = a.split() #기본적으로 공백으로 쪼갬

a 'This-is-an-apple'
ab = a.split('-')
print(ab[0])
print(ab[1])
print(ab[0] + ab[1])
```

```
#대소문자 전환
a = 'Semyeong Computer Highschool'
a.lower #소문자
a.upper #대문자
```

```
#특정 문자열 시작/끝 부분 일치 여부 검색
a = '01-smcimg.jpg'
b = '02-smcimg.jpg'
b = '03-smcimg.gif'
a.startswith('01') #시작하는 부분
a.endswith('png') #끝나는 부분

mylist = [a,b,c]
for file in mylist:
    if file.endswith('jpg'):
        print(file)
```

```
#특정문 자열 교체(replace)
a = '01-smc.jpg'
a.replace('.jpg', '.bmp') #a값이 변경되지 않음 새 변수에 받아줘야 함
```

```
#불필요한 공백 제거(strip)
a = ' kim semyeong'
b = 'kim semyeong'
c = 'kim semyeong '
a.strip()
```

```
#문자열 인덱싱
str = 'abcdef'
print(str[0]) #a
print(str[3]) #d
print(str[-1]) #f
print(str[-2]) #e
print(str[-7]) #error
```

```
#문자열 슬라이싱
str = "Semyeong Computer Highschool"
print(str[:8]) #뒤에 수는 포함 아님
print(str[9:17]) #앞에 수는 포함
print(str[18:])
```

Turtle

1. turtle:화면에 그림을 그리는 파이썬 기본 패키지에 포함된 모듈
2. 속성: 위치, 방향, 펜
3. 사용하기

a. import 필요 import ColabTurtle.Turtle

b. 그림 그릴땐 캔버스 생성

c. 캔버스 초기화 initializeTurtle()

```
#3각형
t.initializeTurtle()
for i in range(3):
  t.forward(100)
  t.left(120)

#4각형
t.initializeTurtle()
for i in range(4):
  t.forward(100)
  t.left(90)

#5각형
t.initializeTurtle()
for i in range(5):
  t.forward(100)
  t.left(72)

#star
t.initializeTurtle()
t.color('yellow')
t.right(90)
for i in range(5):
  t.forward(100)
  t.left(144)

#6각형 여러개를 그려서 360도 채우기 색은 3개를 로테
t.initializeTurtle()
t.speed(10)
for j in range(360):
  if(j%3==0):
    t.color('blue')
  elif(j%3==1):
    t.color('red')
  else:
    t.color('yellow')
  for i in range(6):
    t.forward(100)
    t.left(60)
```

```
  t.left(1)

#이쁜거
import ColabTurtle.Turtle as t
from random import randint
t.initializeTurtle()
t.pensize(1)
x=1
t.speed(10)

while x < 400:
  r = randint(0, 255)
  g = randint(0, 255)
  b = randint(0, 255)

  t.pencolor(r, g, b)
  t.forward(50+x)
  t.right(91)
  x+=1
```

print() 함수

1. print(출력1, 출력2)

```
a = 1
b ='str'
print(a, b)
print('----------')
print(a, b, end = '\n')
print(a, b, end = '\n')
print(a, b, end = '\n')
print('----------')
print(a, b, end = ' ')
print(a, b, end = ' ')
print(a, b, end = ' ')
print('----------')
#sep
print([1,2,3,4,5])
print(a,b)
print(a, b, sep = '|')
#리스트 출력
matrix = [
    [1,2,3],
    [4,5,6],
```

```
    [7,8,9]
]
for i in matrix:
  print(i)
print(*matrix, sep = '\n')
```

변수 출력

1. 형식화된 문자열을 출력하고자 할 때 format 스트링을 사용
2. 중괄호를 사용하면 데이터형에 상관없이 원하는 데이터 전달 가능

```
#format
print("나는 %s고등학교 %d학년 %d반 입니다."&('세명컴퓨터', 2, 5))
print('나는 {}고등학교 {}학년 {}반입니다'.format('세명컴퓨터',2,3))
print("%d는 16진수로 %x 입니다."%(30,30))
a = '세명컴고'
b = '홍길동'
print(f'안녕하세요. {a}에 다니는 {b}입니다.')
```

QRCODE 만들기

1. 필요한 라이브러리
    a. pip install pyqrcode
    b. pip install pypng
2. pyqrcode 사용법
    a. create(contents, error, version, mode)
        i. contents(data) 만 지정해줘도 됨.
    b. eps(file):QR코드 file을 eps 형식으로 작성
    c. svg(file): QR코드 file을 svg 형식으로 작성
    d. png(file, scale, module_color, background):png 형식으로 작성
        i. scale - QR코드 크기
        ii. module_color - QR코드 rgb 색상(일반적으로 검정색), 마지막은 투명도
        iii. background - 배경 rgb 색상(일반적으로 흰색), 마지막 투명도

```
#간단한 qrcode
import pyqrcode
qrcode = pyqrcode.create("https://www.naver.com")
qrcode.svg("qr_naver.svg")
```

```
qrcode.eps("qr_naver.eps")

import pyqrcode
qrcode = pyqrcode.create('https://www.naver.com', error="H", mode='binary
qrcode.png('qr_naver.png', scale=8, module_color=[0,0,0,255], background=
```

```
#wifi 연결
import pyqrcode
data = "WIFI:S:senwifi_Free;T:WPA;P:sen2021!wi;"
qrcode = pyqrcode.create(data, mode='binary')
qrcode.png('qr_wifi.png', scale=8, module_color=[10, 10, 50, 255], backgr
#입력받은 wifi 연결
import pyqrcode
print('#### WIFI 접속 스크립트 ####')
ssid = input('와이파이 SSID를 입력하세요:')
secu = input('암호화 방식을 입력하세요(WPA, WPA2, WAP2-PSK, TKIP 등):')
pw = input('와이파이 PassWord를 입력하세요:')

qrcode = pyqrcode.create(f'WIFI:S:{ssid};T:{secu};p:{pw};')

qrcode.png('qr_wifi2.png', scale=8, module_color= [ 0,0,0,255], backgrou
```

```
#명함용 QR(예시)
import pyqrcode

data = "BEGIN:VCARD\n"
data += "VERSION:3.0\n"
data += "FN:김선생\n"
data += "TEL:TYPE=WORK ;CELL:010 1234 1234\n"
data += "END:VCARD\n"

qrcode = pyqrcode.create(data, mode='binary', encoding='utf-8')
qrcode.png('qr_namecard.png', scale=6, module_color=[10,10,100,255], bac

#명함용QR(+@)
import pyqrcode

data = "BEGIN:VCARD\n"
data += "VERSION:3.0\n"
data += "FN:김선생\n"
data += "ORG:세명컴퓨터고등학교\n"
data += "TITLE:교사\n"
data += "TEL;TYPE=WORK ;CELL:010 1234 1234\n"
data += "TEL;TYPE=HOME ;CELL:010 4444 5555\n"
```

```
data += "EMAIL;TYPE=WORK:abcd@naver.com\n"
data += "URL;TYPE=WORK:https:smc.sen.go.kr\n"
data += "END:VCARD\n"

qrcode = pyqrcode.create(data, mode='binary', encoding='utf-8')
qrcode.png('qr_namecard2.png', scale=6, module_color=[10,10,100,255], ba
```

pip install qrcode

```
import qrcode
img = qrcode.make('나는 문어.')
img.save('dd.png')
```

CV2 라이브러리 사용

```
import cv2

def read_qrcode_cv2(opencv_image):
  detector = cv2.QRCodeDetector()
  data, bbox, _ = detector.detectAndDecode(opencv_image)
  if data:
    print("QRCODE 데이터: {}",format(data))
    print("QRCODE 위치: {}",format(bbox))
    print(_)

filepath = "qr_wifi.png"
cv_image1 = cv2.imread(filepath)
read_qrcode_cv2(cv_image1)
```

1. 이 코드는 qr_wifi.png를 읽고 출력하는 코드
2. 함수 내에서 `cv2.QRCodeDetector()`를 사용하여 QR 코드를 감지하는 객체 `detector`를 생성
3. `detector.detectAndDecode(opencv_image)`를 호출하여 이미지에서 QR 코드를 감지하고 디코딩하고 있어. 이 함수는 세 개의 반환값을 가지는데, 각각은 디코딩된 데이터(`data`), 바운딩 박스(`bbox`), 그리고 추가 정보(`_`)를 나타냄
4. `if data:` 구문은 디코딩된 데이터가 있다면(True), 즉 QR 코드가 감지되고 성공적으로 디코딩 되었다면 해당 블록 안의 코드를 실행
5. `print("QRCODE 데이터: {}".format(data))`는 디코딩된 QR 코드의 데이터를 출력하는 부분이고, `print("QRCODE 위치: {}".format(bbox))`는 바운딩 박스의 좌표를 출력

```
import cv2
from google.colab.patches import cv2_imshow

def read_qrcdoe_cv2(opencv_image):
  detector = cv2.QRCodeDetector()
  data, bbox, _ = detector.detectAndDecode(opencv_image)
  if data:
    print("QRCODE 데이터: {}".format(data))
    print("QRCODE 위치: {}".format(bbox))

    lefttop = (int(bbox[0][0][0]), int(bbox[0][0][1]))
    rightbottom = (int(bbox[0][2][0]), int(bbox[0][2][1]))

    cv2.rectangle(opencv_image, lefttop, rightbottom, (0,0,255), 5)

    cv2_imshow(opencv_image)

filepath = "qr_wifi.png"
cv_image1 = cv2.imread(filepath)
read_qrcdoe_cv2(cv_image1)
```

위에 코드에 테두리 붙이는 것 rgb값 설정이 이상함

QR끝

pyautogui

마우스 키보드 조작하는 라이브러리

```
#이건 걍 모니터 사이즈
import pyautogui as pg
print(pg.size())
#이건 마우스 위치 출력
print(pg.position())
#마우스 정보
pg.mouseInfo()
#마우스 절대 좌표로 이동
pg.moveTo(x=9, y=9)
#마우스 절대 좌표로 이동인데 시간만큼 움직임
pg.moveTo(x=10, y=10, duration=10)
#마우스 상대 좌표로 이동
pg.move(100, 100)
```

```
#오토마우스 clicks는 회수 interval은 간격 button은 버튼
pg.click(clicks=100, interval=0.1, button='left')
#마우스 드래그
pg.moveTo(816,81, 2)
pg.dragTo(539,80, 2)
#그냥 써줌 안에내용
pg.write("Hello World!")
#알림창 띄움
a = pg.alert(text="내용", title='제목',button='ok')
print(a)
#클릭한 버튼값리턴함
a = pg.confirm(text="나나나나나", title='나나난',buttons=['ok', 'cancel'])
print(a)
#입력창이 있는 메세지 박스
a = pg.prompt(text='내용입니다', title='제목입니다', default='입력해라')
print(a)
#password입력창
a = pg.password(text='내용입니다', title='제목입니다', default='입력하세요', ma
print(a)
```

```
#버튼 1개 클릭 3번 10초
import pyautogui as pg
pg.click(397, 590, clicks=3, interval=11)
```

```
#버튼 2개 클릭 3번 1분기다리기
import time
import pyautogui as pg
for i in range(0, 3):
    time.sleep(62)
    pg.click(1358, 1010, duration=1)
    pg.click(1496, 1010, duration=1)
```

opencv-python

이미지 인식하는 라이브러리

```
pip install opencv-python
```

```
#num5.png부분을 화면에서 찾아서 해당 이미지의 좌표의 중간을 누르는 거?
i = pg.locateOnScreen('num5.PNG')
```

```
q = pg.center(i)
pg.click(q)
#걍 이렇게 가능
i = pg.locateCenterOnScreen('num5.PNG')
pg.click(i)
```

```
#지정된 좌표를 스샷함 뒤에 숫자는 스샷찍는 사진의 가로 세로 사이즈
import pyautogui as pg
x, y = pg.position()
pg.screenshot('num2.png', region=(x, y, 40, 40))
```

```
#메모장 열고 hello입력
import pyautogui as pg
import time
time.sleep(1)
pg.hotkey('win', 'r')
time.sleep(0.5)
pg.write('notepad')
pg.write(['enter'])
time.sleep(1)
pg.write('hello')
```

```
#반환값을 이용하는 방법
btn2 = pg.confirm('선택하세요.')
if btn2 == 'OK':
    print("ㅇㅇ")
else:
    print("ㄴㄴ")
```

selenium

웹 크롤링 및 자동화를 위한 툴

```
#네이버 띄우고 검색
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time
```

```
driver = webdriver.Chrome()
driver.get('https://www.naver.com')
time.sleep(2)
driver.find_element(By.CLASS_NAME, "search_input").send_keys("파이썬 seler
time.sleep(2)
```

```
#이렇게 해도될듯 전에했던거 사용해서
from selenium import webdriver
from selenium.webdriver.common.keys import Keys

driver = webdriver.Chrome()
driver.get('https://www.naver.com')
time.sleep(2)
pg.write("python selenium")
time.sleep(0.001)
pg.press('enter')
```

```
#위에거 입력하고 지우고 다른거 검색하고 스크린샷 저장
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time

driver = webdriver.Chrome()
driver.get('https://www.naver.com')
time.sleep(2)

driver.find_element(By.CLASS_NAME, "search_input").send_keys("파이썬 seler
time.sleep(2)

driver.find_element(By.NAME,"query").clear()
driver.find_element(By.NAME,"query").send_keys("세명컴퓨터고등학교"+Keys.RETU
time.sleep(2)

png = driver.get_screenshot_as_png()
open('search_result.png', 'wb').write(png)
```

```
#웹브라우저에서 구글 이미지 검색 사이트 띄우기
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time
```

```
driver = webdriver.Chrome()
driver.get('https://www.google.com/imghp?hl=ko&authuser=0&ogbl')
ele = driver.find_element(By.NAME, "q")
ele.send_keys("세명컴퓨터고등학교")
ele.send_keys(Keys.RETURN)
```

```
#네이버뉴스에서 수학능력시험 검색
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time

driver = webdriver.Chrome()
driver.get('https://news.naver.com')
driver.find_element(By.CLASS_NAME, "Nicon_search").click()
time.sleep(2)
ele = driver.find_element(By.NAME, "query")
ele.send_keys("수학능력시험")
ele.send_keys(Keys.RETURN)



#ele = driver.find_element(By.NAME, "query")
#ele.send_keys("수학능력시험")
#ele.send_keys(Keys.RETURN)
#이부분을 합쳐서
#driver.find_element(By.NAME, "query").send_keys("수학능력시험" +Keys.RETUR
#이렇게 해도됨
```

```
#네이버 뉴스 좌측상단 1번째 뉴스 클릭
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time

driver = webdriver.Chrome()
ele = driver.get('https://news.naver.com')

time.sleep(2)
driver.find_elements(By.CLASS_NAME, "cjs_news_mw")[0].click()
```

```
#css를 이용해서 하는거
from selenium import webdriver
```

```
from selenium.webdriver.common.by import By
import time
driver = webdriver.Chrome()

driver.get("https://www.naver.com")
time.sleep(2)

css_selector = "#shortcutArea > ul"
shortcut = driver.find_element(By.CSS_SELECTOR, css_selector)
print(shortcut.text)
shortcut.click()
```

```
#웹브라우저에서 구글 이미지 검색 사이트 띄우고 첫번째거 누르기
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time

driver = webdriver.Chrome()
driver.get('https://www.google.com/imghp?hl=ko&authuser=0&ogbl')
driver.find_element(By.NAME, "q").send_keys("집에 가고 싶다 짤" +Keys.RETURN
css_selector = "#islrg > div.islrc > div:nth-child(2) > a.FRuiCf.islib.n
driver.find_element(By.CSS_SELECTOR, css_selector).click()
```

selenium 보조도구 (Navigation 관련)

- get: 원하는 페이지로 이동
- back: 뒤로가기
- forward: 앞으로가기
- refresh: 새로고침
- title: 현재탭 타이틀 갖고옴
- current_url: 현재탭 url갖고옴

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time

driver = webdriver.Chrome()

driver.get("https://www.naver.com")
time.sleep(2)
```

```
driver.get("https://www.google.com")

driver.back()
time.sleep(2)

driver.forward()
time.sleep(2)

driver.refresh()
```

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time

driver = webdriver.Chrome()
driver.get("https://mail.naver.com")

title = driver.title
print(title)

url = driver.current_url
print(url)
```

```
#로그인 여부를 알아내고 출력
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time

driver = webdriver.Chrome()
driver.get("https://mail.naver.com")
time.sleep(2)

title = driver.title
print(title)
url = driver.current_url
print(url)

login_url = "nid.naver.com"
if login_url in url:
    print("로그인이 필요함")
```

```
else:
    print("로그인이 불필요함")
```

대기(Driver wait)

- time.sleep(3)는 대기하는 것으로 적절치 않음
- 네트워크 및 시스템 상태에 따라 2초가 적합할지 10초가 적합할지 알 수 없음
- webDriverWait, expected_conditions 필요
- from selenium.webdriver.support.ui import WebDriverWait 필요
- from selenium.webdriver.support import expected_conditions as EC
- 이런 기능으로 인해 Selenium은 동적인 사이트를 다루는 데 유용함
- 개발자 도구에서 Network 속도를 느리게 하여 테스트 가능함

```
#네이버 증시정보 뜰때까지 기다리는 거
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

d = webdriver.Chrome()
d.get("https://naver.com")

css = "#right-content-area > div.Layout-module__right_top___h3g3v > div:n

ele = WebDriverWait(d, 10).until(EC.presence_of_all_elements_located((By
print(d.find_element(By.CSS_SELECTOR , css).text)
#print(ele.text) 이거 왜안됨!?
ele.click()
```

```
#영상 제목 1개 갖고옴
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
import time

d = webdriver.Chrome()
search = '집에갈래'.replace(' ', '+')

url = "https://www.youtube.com/results?search_query=" + search
d.get(url)
time.sleep(2)
```

```
selector = "#video-title"
item = d.find_element(By.CSS_SELECTOR, selector)
print(item.text)
```

```
#제목 url 원하는 만큼 갖고 오기
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
import time

d = webdriver.Chrome()
search = '집에갈래'.replace(' ', '+')

url = "https://www.youtube.com/results?search_query=" + search
d.get(url)
time.sleep(2)

#d.execute_script("window.scrollBy(0, 10000);")

selector = "#video-title"

for i in range(0,10):
    item = d.find_elements(By.CSS_SELECTOR, selector)[i]
    url = item.get_attribute("href")
    print(f"{item.text}:{url}")
```

```
#오류
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time

driver = webdriver.Chrome()
query = "CPU"
page = 1
item_name = ""
url = f"https://msearch.shopping.naver.com/search/all?pagingIndex={page}&
driver.get(url)
time.sleep(2)

for i in range(2):
    driver.execute_script("window.scrollBy(0, 10000);")
```

```
    time.sleep(1)
try:
    target_selector = "#content > div.style_content__xWg5l > div.basicLis
    item = driver.find_element(By.CSS_SELECTOR, target_selector)
    item_name = item.text
    item_code = item.get_attribute('data-i')
    data = item.get_attribute('data-nclick')
    rank = data.split(f"{item_code},r:")

    print(f"item code: {item_code}, data: {data}, rank: {rank}")
    print(f"'{item_name}'은 {query} 키워드로 검색시 {page}페이지에서 검색되었으며
except:
    print(f"{page} 페이지에서 타겟 상품을 못 찾음")
```

```
#제목입력받아 검색 네이버뉴스
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time

driver = webdriver.Chrome()
keyword = input('검색할 제목을 적어주세요')
searchurl = 'https://search.naver.com/search.naver?where=news&ie=utf8&sm=
driver.get(searchurl+keyword)
items = driver.find_elements(By.CLASS_NAME,'news_tit')
for i in range(len(items)):
    url = items[i].get_attribute("href")
    print(f'{items[i].text}:{url}')
```

```
#아이유가 뭐하는 아이유?
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time

driver = webdriver.Chrome()
search = "아이유"
driver.get("https://www.google.com/imghp?hl=ko&authuser=0&ogbl")
time.sleep(1)

selector = "#APjFqb"
```

```
item = driver.find_element(By.CSS_SELECTOR, selector)
item.send_keys(search+Keys.RETURN)

time.sleep(1)
item = driver.find_elements(By.CLASS_NAME, "rg_i.Q4LuWd")

for i in range(len(item)):
    item[i].click()
    time.sleep(1.5)
```

```
#김세정 무한 스크롤 가능
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time

driver = webdriver.Chrome()
search = "김세정"
driver.get("https://www.google.com/imghp?hl=ko&authuser=0&ogbl")
time.sleep(1)

selector = "#APjFqb"

item = driver.find_element(By.CSS_SELECTOR, selector)
item.send_keys(search+Keys.RETURN)

time.sleep(1)
item = driver.find_elements(By.CLASS_NAME, "rg_i.Q4LuWd")

for i in range(len(item)):
    driver.execute_script("arguments[0].scrollIntoView(true);", item[i])
    item[i].click()
    time.sleep(1)
    if i > 100:
        break
```

```
#위에거 제목 갖고 오기
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time

driver = webdriver.Chrome()
search = "김세정"
```

```
driver.get("https://www.google.com/imghp?hl=ko&authuser=0&ogbl")
time.sleep(1)

selector = "#APjFqb"

item = driver.find_element(By.CSS_SELECTOR, selector)
item.send_keys(search+Keys.RETURN)

time.sleep(1)
item = driver.find_elements(By.CLASS_NAME, "rg_i.Q4LuWd")

for i in range(len(item)):
    driver.execute_script("arguments[0].scrollIntoView(true);", item[i])
    item[i].click()
    imgtxt = item[i].get_attribute("alt")
    print(f"{imgtxt}")
    time.sleep(1)
    if i > 100:
        break
```

```
#썸네일 이미지 크롤링
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time
import urllib.request

driver = webdriver.Chrome()
search = "페이커"
driver.get("https://www.google.com/imghp?hl=ko&authuser=0&ogbl")
time.sleep(1)

selector = "#APjFqb"

item = driver.find_element(By.CSS_SELECTOR, selector)
item.send_keys(search+Keys.RETURN)

time.sleep(1)
item = driver.find_elements(By.CLASS_NAME, "rg_i.Q4LuWd")

for i in range(len(item)):
    driver.execute_script("arguments[0].scrollIntoView(true);", item[i])
    imgurl = item[i].get_attribute("src")
```

```
    urllib.request.urlretrieve(imgurl, f"{search}{i}.jpg")
    time.sleep(1)
```

```
#위에거 걍 사진 뜯어오기
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import urllib
import time

driver = webdriver.Chrome()
search = "김세정"
driver.get("https://www.google.com/imghp?hl=ko&authuser=0&ogbl")
time.sleep(1)

selector = "#APjFqb"

item = driver.find_element(By.CSS_SELECTOR, selector)
item.send_keys(search+Keys.RETURN)

time.sleep(1)
item = driver.find_elements(By.CLASS_NAME, "rg_i.Q4LuWd")

for i in range(len(item)):
    driver.execute_script("arguments[0].scrollIntoView(true);", item[i])
    item[i].click()
    imgtxt = item[i].get_attribute("alt")
    time.sleep(1.5)
    imgurl = item[i].get_attribute("src")
    img = driver.find_element(By.XPATH, "//*[@id=\"Sva75c\"]/div[2]/div[
    print(imgurl)

    opener = urllib.request.build_opener()
    opener.addheaders = [('User-Agent', 'Mozila/5.0')]
    urllib.request.install_opener(opener)

    urllib.request.urlretrieve(imgurl, f"{search}{i}{i}.jpg")
    print(f"{imgtxt}")
```

```
#이것은 유튜브다.... 밑에서 이거 활용함
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time
```

```
driver = webdriver.Chrome()
search = input('검색어를 입력하세요: ').replace(' ', '+')
URL = "https://www.youtube.com/results?search_query=" + search
driver.get(URL)
time.sleep(3)
selector = "#video-title"

for i in range(0, 20):
    item = driver.find_element(By.CSS_SELECTOR, selector)
    driver.execute_script("arguments[0].scrollintoView(true):", item)
    item_name = item.text
    url = item.get_attribute("href")
    if url:
        print(f"{item_name} : {url}")
    else:
        continue
    time.sleep(1)
```

pandas

데이터 분석 라이브러리

엑셀파일 저장

- pandas 라이브러리 사용
- 파이썬을 이용하여 데이터 분석 시 필수적인 패키지
- DataFrame 을 이용하면 데이터를 표(table) 형태로 처리 가능

```
#1~100까지 숫자 엑셀에 저장
import pandas as pd

mylist = []
for i in range(100):
    mylist.append(i)

df = pd.DataFrame({"세로줄":mylist})
print(df)

df.to_excel("엑셀연습.xlsx") #이 부분이 엑셀파일 만드는 거임
```

유튜브 크롤링 결과를 엑셀 파일로 저장하기

1. 검색어를 입력받는다.
2. 유튜브에서 검색어로 조회한다.
3. 검색 결과 상위 10개를 검색하는데, URL이 없을 경우는 해당 항목은 생략한다.(광고 또는 쇼츠)
4. 제목, URL을 출력하고, 해당 내용을 엑셀파일로 저장한다.

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
import time
import pandas as pd

driver = webdriver.Chrome()
# 검색 키워드 설정: 키워드 내 띄어쓰기는 URL에서 '+'로 표시되기 때문에 이에 맞게 변환
SEARCH_KEYWORD = input('검색어를 입력하세요: ').replace(' ', '+')
# 스크래핑 할 URL 세팅
URL = "https://www.youtube.com/results?search_query=" + SEARCH_KEYWORD
# 크롤링할 내용을 담아 엑셀로 만들 리스트 생성
data_list = []

# 크롬 드라이버를 통해 지정한 URL의 웹 페이지 오픈
driver.get(URL)
# 웹 페이지 로딩 대기
time.sleep(3)

selector = "#video-title"
for i in range(0,10):
    data = {}
    item = driver.find_elements(By.CSS_SELECTOR, selector)[i]
    item_name = item.text
    url = item.get_attribute("href")
    if url:
        data['title'] = item_name
        data['url'] = url
        data_list.append(data)
        print(f"{item_name} : {url}")
    else:
        continue

df = pd.DataFrame(data_list)
#print(data_list)
print(df)
```

```
# 엑셀로 저장
df.to_excel(f"youtube_{SEARCH_KEYWORD}_result.xlsx")
```

네이버 뉴스 검색 결과 크롤링 및 엑셀 파일 저장

- 원하는 검색어 입력
- 해당 검색어로 네이버 뉴스 사이트에서 검색(URL로 바로 접근)
- 검색 결과 모두 Dataframe에 넣어서 출력
- 결과 저장(엑셀파일로)

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time
import pandas as pd

driver = webdriver.Chrome()
search = input('검색어를 입력하세요: ').replace(' ', '+')

driver.get(f"https://search.naver.com/search.naver?where=news&sm=tab_jum
data_list = []
time.sleep(1)

# 요소를 찾아서 카피 해옴. 웹브라우저 + 개발자 도구
selector = "news_tit"

# 찾아온 요소를 find_element로 가져오기
items = driver.find_elements(By.CLASS_NAME, selector)

time.sleep(1)

for i in range(len(items)):
    data = {}
    item_name = items[i].text
    url = items[i].get_attribute("href")
    if url:
        data['title'] = item_name
        data['url'] = url
        data_list.append(data)
        #print(f"{item_name} : {url}")
    else:
        continue

df = pd.DataFrame(data_list)
```

```
#print(data_list)
print(df)
# 엑셀로 저장
df.to_excel(f"naver_{search}_result.xlsx")
```

파이썬에서 데이터 입력하기(DB)

1. MySQL연결: conn = pymysql.cinnect(연결옵션)
2. 커서 생성하기: cur = conn.cursor()
3. 테이블 만들기: cur.execute(”CREATE TABLE문장”)
4. 데이터 입력하기: cur.execute(”INSERT문장”)
5. 입력 데이터 저장: conn.commit()
6. MySQL 연결 종료: conn.close()

뭉탱이

```
import pymysql
conn = pymysql.connect(host='localhost', user='root', password='root', dl
cur = conn.cursor()

sql = """CREATE TABLE stu (
    uid INT PRIMARY KEY AUTO_INCREMENT,
    nick VARCHAR(20) NOT NULL,
    pwd VARCHAR(20) NOT NULL,
    phone VARCHAR(20) NOT NULL,
    addr VARCHAR(50)
);"""

cur.execute(sql)

sql = "INSERT INTO stu (nick, pwd, phone, addr) VALUES ('smc01', '1234',
cur.execute(sql)

sql = "INSERT INTO stu (nick, pwd, phone, addr) VALUES ('smc02', '5678',
cur.execute(sql)

sql = "INSERT INTO stu (nick, pwd, phone, addr) VALUES ('smc03', '1111',
cur.execute(sql)

nick = "smc04"
pwd = "9999"
phone = "0105555"
addr = "서울시 마포구"
```

```
sql = "INSERT INTO stu (nick, pwd, phone, addr) VALUES (%s, %s, %s, %s)"
vals = (nick, pwd, phone, addr)
cur.execute(sql, vals)

# 위에서 실행한 쿼리를 취소하고 싶다.
conn.rollback()

# 위에서 실행한 쿼리 결과를 반영(저장)하겠다.
conn.commit()

# 연결 종료
conn.close()

# 데이터 검색하기(SELECT)
# * 커넥션 -> 커서 -> 실행 -> fetch


import pymysql
conn = pymysql.connect(host='localhost', user='root', password='root', dI
cur = conn.cursor()

sql = "SELECT * FROM stu"
cur.execute(sql)
data = cur.fetchall()
print(data)

import pandas as pd
data = pd.DataFrame(data)
data

sql= "select * from country limit 10"
cur.execute(sql)
data = cur.fetchall()
cols = [desc[0] for desc in cur.description]
cols

data = pd.DataFrame(data, columns = cols)
data

sql = "select * from city limit %s"
vals = 10
cur.execute(sql, vals)
data = cur.fetchall()
print(data)
```

```
cols = [desc[0] for desc in cur.description]
data = pd.DataFrame(data, columns=cols)
data

def tb_search(tb, num):
    sql = f"select * from {tb} limit {num}"
    cur.execute(sql)
    data = cur.fetchall()
    print(data)

tb_search('city', 10)

# 데이터베이스에서 검색하기
# * 검색어 입력받기
# * 해당 검색어를 city 테이블에서 검색하기
# * pandas의 데이터프레임 형태로 출력하기


search = input('도시명을 검색하세요 :')
sql = f"SELECT * FROM city WHERE Name LIKE '%{search}%'"
cur.execute(sql)
cols = [desc[0] for desc in cur.description]
data = cur.fetchall()
#print(data)

pd.DataFrame(data)
```

SQL 모음(1주차)

설명 by GPT

1일차 문제 1 프로그래머스(조건에 부합하는 중고거래 댓글 조회하기)

```
SELECT
    USED_GOODS_BOARD.TITLE,
    USED_GOODS_BOARD.BOARD_ID,
    USED_GOODS_REPLY.REPLY_ID,
    USED_GOODS_REPLY.WRITER_ID,
    USED_GOODS_REPLY.CONTENTS,
    DATE_FORMAT(USED_GOODS_REPLY.CREATED_DATE, '%Y-%m-%d') AS CREATED_DA
FROM
    USED_GOODS_BOARD
INNER JOIN
    USED_GOODS_REPLY ON USED_GOODS_BOARD.BOARD_ID = USED_GOODS_REPLY.BOA
WHERE
    MONTH(USED_GOODS_BOARD.CREATED_DATE) = 10
ORDER BY
    USED_GOODS_REPLY.CREATED_DATE ASC,
    USED_GOODS_BOARD.TITLE ASC;

-- join문, as문
```

join문 설명

JOIN은 SQL에서 두 개 이상의 테이블을 연결하여 하나의 결과 집합으로 조합하는 작업을 수행하는 키워드입니다. JOIN을 사용하여 데이터베이스에서 여러 테이블 간의 관계를 활용하여 필요한 정보를 효과적으로 검색하거나 조작할 수 있습니다.

JOIN은 다양한 종류가 있으며, 각각의 JOIN 유형은 테이블 간의 관계와 원하는 결과에 따라 선택됩니다. 주요 JOIN 유형은 다음과 같습니다:

1. **INNER JOIN:** 두 테이블 간의 일치하는 행만을 선택합니다. 즉, 공통된 값을 가지는 행들을 연결하여 결과로 얻습니다. 일치하는 데이터만 보여주는 JOIN 유형입니다.
2. **LEFT JOIN (또는 LEFT OUTER JOIN):** 왼쪽(첫 번째) 테이블의 모든 행과 오른쪽(두 번째) 테이블의 일치하는 행을 선택합니다. 오른쪽 테이블에 일치하는 행이 없더라도 왼쪽 테이블의 모든 행은 결과에 포함됩니다.
3. **RIGHT JOIN (또는 RIGHT OUTER JOIN):** LEFT JOIN과 유사하지만, 오른쪽(두 번째) 테이블의 모든 행과 왼쪽(첫 번째) 테이블의 일치하는 행을 선택합니다. 왼쪽 테이블에 일치하는 행이 없더라도 오른쪽 테이블의 모든 행은 결과에 포함됩니다.

4. **FULL JOIN (또는 FULL OUTER JOIN):** 왼쪽과 오른쪽 테이블의 모든 행을 선택하며, 일치하는 행이 없는 경우에도 모든 행이 결과에 포함됩니다.
5. **SELF JOIN:** 하나의 테이블을 여러 번 사용하여 자체적으로 연결하는 것을 말합니다. 이를 통해 같은 테이블 내의 데이터를 비교하거나 관계를 분석할 수 있습니다.
6. **CROSS JOIN:** 두 테이블의 모든 가능한 조합을 생성합니다. 즉, 첫 번째 테이블의 각 행이 두 번째 테이블의 모든 행과 결합됩니다. 큰 결과 집합을 생성할 수 있으므로 사용할 때 주의가 필요합니다.

JOIN을 사용할 때, ON 절을 사용하여 어떤 열을 기준으로 두 테이블을 연결할 것인지를 명시해야 합니다. 이를 통해 데이터베이스 시스템은 해당 열의 값에 따라 행을 조합합니다.

예시:

```
SELECT orders.order_id, customers.customer_name
FROM orders
INNER JOIN customers ON orders.customer_id = customers.customer_id;
```

위 예시는 "orders" 테이블과 "customers" 테이블을 "customer_id" 열을 기준으로 INNER JOIN하여 주문 정보와 고객 이름을 연결하는 예시입니다.

AS문 설명

"AS"는 SQL에서 결과 열에 별칭(alias)을 부여하기 위해 사용되며, 결과 데이터를 더 읽기 쉽게 만들거나 열 이름 충돌을 방지하기 위해 활용됩니다. 아래 예시를 통해 "AS" 문법을 설명하겠습니다.

1. **단순한 열 별칭:**

   ```
   SELECT first_name AS "First Name", last_name AS "Last Name"
   FROM employees;
   ```

   위의 쿼리에서 "first_name" 열은 "First Name", "last_name" 열은 "Last Name"으로 별칭을 지정하여 결과를 읽기 쉽게 만들었습니다.

2. **계산된 열에 별칭 사용:**

   ```
   SELECT salary * 12 AS annual_salary
   FROM employees;
   ```

   위의 쿼리에서 "salary" 값을 12로 곱한 결과에 "annual_salary"라는 별칭을 지정하여 연봉을 계산한 값을 보여주었습니다.

3. **테이블에 별칭 지정:**

   ```
   SELECT *
   FROM orders AS o
   WHERE o.order_date >= '2023-01-01';
   ```

   위의 쿼리에서 "orders" 테이블에 "o"라는 별칭을 지정하여 더 간결하게 테이블에 접근하였습니다.

4. **함수나 표현식에 별칭 사용:**

```
SELECT AVG(price) AS avg_price
FROM products
WHERE category = 'Electronics';
```

위의 쿼리에서 "AVG(price)" 함수의 결과에 "avg_price"라는 별칭을 지정하여, 평균 가격을 읽기 쉽게 표현하였습니다.

"AS"를 사용하면 결과 데이터를 더 명확하고 가독성 있게 만들 수 있으며, 복잡한 쿼리에서도 열 이름 충돌을 방지할 수 있습니다.

2일차 문제 1 프로그래머스(조건에 부합하는 중고거래 상태 조회하기)

```
SELECT BOARD_ID, WRITER_ID, TITLE, PRICE,
case
    when STATUS = 'DONE' then '거래완료'
    when STATUS = 'SALE' then '판매중'
    when STATUS = 'RESERVED' then '예약중'
    end as STATUS
from USED_GOODS_BOARD
where DATE_FORMAT(CREATED_DATE, '%m-%d') = '10-05'
order by BOARD_ID desc;

-- case문
```

CASE문 설명

물론, 아래는 `CASE` 문을 사용한 예시와 함께 설명한 내용입니다.

가상의 'students' 테이블에는 학생들의 성적 정보가 저장되어 있다고 가정해봅시다. 이 성적 정보에 따라 학생들을 성적 등급으로 나누고자 합니다. 성적이 90 이상이면 'A', 80 이상이면 'B', 70 이상이면 'C', 그리고 70 미만이면 'F'라는 등급을 부여하려고 합니다.

이러한 상황에서 `CASE` 문을 사용하여 쿼리를 작성할 수 있습니다:

```
SELECT student_name,
       score,
       CASE
           WHEN score >= 90 THEN 'A'
           WHEN score >= 80 THEN 'B'
           WHEN score >= 70 THEN 'C'
           ELSE 'F'
```

```
        END AS grade
FROM students;
```

위의 쿼리에서:

- `score` 컬럼이 90 이상인 경우 'A'를 할당합니다.
- `score` 컬럼이 80 이상인 경우 'B'를 할당합니다.
- `score` 컬럼이 70 이상인 경우 'C'를 할당합니다.
- 위의 조건에 해당하지 않는 경우 'F'를 할당합니다.

이와 같은 방식으로 `CASE` 문을 사용하여 조건에 따라 학생들을 성적 등급으로 나눌 수 있습니다. 이 결과는 `AS` 문을 사용하여 'grade'라는 별칭을 가진 새로운 열로 반환됩니다.

결과적으로 `CASE` 문은 데이터베이스에서 조건에 따라 값을 가공하거나 변경할 때 매우 유용한 도구입니다.

2일차 문제2 프로그래머스(조건에 맞는 사용자와 총 거래 금액 조회하기)

```
SELECT ugu.USER_ID,
       ugu.NICKNAME,
       SUM(ugb.PRICE) AS TOTAL_SALES
FROM USED_GOODS_BOARD ugb
JOIN USED_GOODS_USER ugu ON ugb.WRITER_ID = ugu.USER_ID
where STATUS = 'DONE'
GROUP BY ugb.WRITER_ID
HAVING TOTAL_SALES >= 700000
order by TOTAL_SALES asc;
--HAVING
```

HAVING문 설명

`HAVING` 절은 SQL에서 `GROUP BY` 절과 함께 사용되어 그룹화된 결과에 대한 필터링을 수행하는 역할을 합니다. `HAVING` 절은 `GROUP BY` 절 다음에 오며, 그룹화된 결과에 대한 조건을 지정하여 원하는 그룹만 선택할 수 있습니다. 아래 예시를 통해 `HAVING` 절을 설명하겠습니다.

가정:

가게에서 판매된 제품에 대한 판매량 데이터가 있는 'sales' 테이블이 있다고 가정합니다. 이 테이블에는 제품 이름(`product_name`)과 해당 제품의 판매량(`quantity`)이 저장되어 있습니다.

목표:

판매량이 50개를 초과하는 제품만 출력하고 싶습니다.

쿼리:

```
SELECT product_name, SUM(quantity) AS total_quantity
FROM sales
GROUP BY product_name
HAVING SUM(quantity) > 50;
```

위의 쿼리에서:

- `SELECT` 절에서 `SUM(quantity)` 함수를 사용하여 각 제품의 총 판매량을 계산하고 `total_quantity` 별칭으로 반환합니다.
- `FROM` 절에서 'sales' 테이블을 선택합니다.
- `GROUP BY` 절에서 `product_name` 열을 기준으로 그룹화합니다. 이로 인해 각 제품별로 그룹이 형성됩니다.
- `HAVING` 절에서 그룹화된 결과 중에서 `SUM(quantity)` 값이 50보다 큰 그룹만 선택합니다.

결과:

쿼리 실행 결과로, 판매량이 50개를 초과하는 제품만 출력됩니다.

`HAVING` 절은 그룹화된 결과에 대한 조건을 지정하여 필터링하는 데 사용되며, 그룹화된 결과 중에서 원하는 그룹만 선택하여 출력할 때 유용합니다.

2일차 문제3 프로그래머스(조회수가 가장 많은 중고거래 게시판의 첨부파일 조회하기)

```
SELECT CONCAT('/home/grep/src/', ugb.BOARD_ID, '/', ugf.FILE_ID, ugf.FILE
from USED_GOODS_BOARD ugb
join USED_GOODS_FILE ugf on ugb.BOARD_ID = ugf.BOARD_ID
where views = (select max(views) from USED_GOODS_BOARD)
order by ugf.FILE_ID desc;
--concat
```

```
--다른 사람의 쿼리 (서브쿼리를 두번사용하여 join문 없이 품)
SELECT CONCAT('/home/grep/src/', BOARD_ID, '/', FILE_ID, FILE_NAME, FILE_
FROM USED_GOODS_FILE
WHERE BOARD_ID =
    (
    SELECT BOARD_ID
    FROM USED_GOODS_BOARD
    WHERE VIEWS =
        (
        SELECT MAX(VIEWS) FROM USED_GOODS_BOARD
```

```
        )
    )
ORDER BY FILE_ID DESC;
--이런식으로 푸는게 시간을 더 아낄거 같음
```

CONCAT문 설명

`CONCAT` 함수는 SQL에서 사용되는 함수 중 하나로, 문자열을 결합하여 하나의 문자열로 만드는데 사용됩니다. 이 함수는 주어진 문자열 인자들을 차례대로 연결하여 새로운 문자열을 생성합니다. 예시를 통해 `CONCAT` 함수의 작동 방식을 이해해 보겠습니다.

가정하자면, 다음과 같은 두 개의 문자열이 있다고 해봅시다:

- 문자열 1: `'Hello, '`
- 문자열 2: `'World!'`

이 두 문자열을 `CONCAT` 함수를 사용하여 연결하면 다음과 같은 결과를 얻을 수 있습니다:

```
SELECT CONCAT('Hello, ', 'World!') AS ConcatenatedString;
```

위의 SQL 쿼리를 실행하면 `'Hello, World!'`라는 새로운 문자열이 반환됩니다. `CONCAT` 함수 내에 여러 개의 인자를 전달하여 순서대로 결합할 수 있습니다.

또한, 변수나 열(column)의 값과 문자열을 결합할 수도 있습니다. 예를 들어, 데이터베이스 테이블에 다음과 같은 레코드가 있다고 가정해 봅시다:

BOARD_ID	FILE_ID	FILE_NAME	FILE_EXT
123	1	image	jpg

다음과 같은 SQL 쿼리를 사용하여 `CONCAT` 함수를 이용해 열들의 값을 결합할 수 있습니다:

```
SELECT CONCAT('/home/grep/src/', BOARD_ID, '/', FILE_ID, '/', FILE_NAME,
FROM USED_GOODS_FILE;
```

위의 쿼리는 결과로 `/home/grep/src/123/1/image.jpg`와 같은 `FILE_PATH` 값을 반환합니다. 이처럼 `CONCAT` 함수를 사용하여 여러 문자열을 결합하여 원하는 형식의 문자열을 생성할 수 있습니다.

3일차 문제1 프로그래머스(저자 별 카테고리 별 매출액 집계하기)

```
SELECT b.AUTHOR_ID, a.AUTHOR_NAME, b.CATEGORY, sum(b.PRICE * s.SALES) as
from BOOK b
join AUTHOR a on b.AUTHOR_ID = a.AUTHOR_ID
join BOOK_SALES s on b.BOOK_ID = s.BOOK_ID
```

```
where s.SALES_DATE like '2022-01%'
group by a.AUTHOR_ID, b.CATEGORY
order by a.AUTHOR_ID, b.CATEGORY desc;
--join문 여러개
```

```
SELECT b.AUTHOR_ID, a.AUTHOR_NAME, b.CATEGORY, sum(b.PRICE * s.SALES) as
from BOOK b
join AUTHOR a on b.AUTHOR_ID = a.AUTHOR_ID
join BOOK_SALES s on b.BOOK_ID = s.BOOK_ID
where b.PUBLISHED_DATE like '2022-01%'
group by a.AUTHOR_ID, b.CATEGORY
order by a.AUTHOR_ID, b.CATEGORY desc;

/*
계속 틀렸던 코드: 왜 안되나 한참 보다가 s.SALES_DATE로
풀어야 하는걸 b.PUBLISHED_DATE 로 함 잘못봐서 1시간 낭비ㅋㅋㅋ
*/
```

JOIN문 여러개 사용

SQL에서 여러 개의 테이블을 조인하는 것은 데이터를 결합하여 더 복잡한 쿼리를 생성하는데 사용됩니다. 다음은 여러 개의 JOIN 절을 사용하는 SQL 쿼리의 예시입니다.

가정:

- "orders" 테이블: 주문 정보를 저장하는 테이블 (order_id, customer_id 등)
- "order_items" 테이블: 주문된 제품 정보를 저장하는 테이블 (order_id, product_id, quantity 등)
- "products" 테이블: 제품 정보를 저장하는 테이블 (product_id, product_name, price 등)
- "customers" 테이블: 고객 정보를 저장하는 테이블 (customer_id, customer_name, email 등)

이제 주문된 제품 정보와 해당 제품의 정보, 그리고 주문한 고객의 정보를 모두 가져오는 쿼리를 작성해보겠습니다.

```
SELECT
    o.order_id,
    c.customer_name,
    p.product_name,
    oi.quantity,
    p.price
FROM
    orders o
JOIN
```

```
    customers c ON o.customer_id = c.customer_id
JOIN
    order_items oi ON o.order_id = oi.order_id
JOIN
    products p ON oi.product_id = p.product_id;
```

이 쿼리는 "orders" 테이블을 기준으로 "customers", "order_items", "products" 테이블과 조인하며, 각 테이블 간의 관계를 활용하여 주문 정보, 고객 정보, 제품 정보, 그리고 주문된 제품의 수량 및 가격을 가져옵니다.

이와 같은 방식으로 JOIN 절을 여러 개 사용하여 데이터를 결합하고 필요한 정보를 가져올 수 있습니다. 조인을 사용할 때는 테이블 간의 관계를 이해하고, 적절한 조인 조건을 사용하여 데이터를 정확하게 가져오는 것이 중요합니다.

4일차 문제 1 프로그래머스(년, 월, 성별 별 상품 구매 회원 수 구하기)

```
SELECT
    YEAR(sale.SALES_DATE) as YEAR,
    MONTH(sale.SALES_DATE) as MONTH,
    user.GENDER,
    count(DISTINCT sale.USER_ID) as USERS
from
    USER_INFO user
join
    ONLINE_SALE sale on user.USER_ID = sale.USER_ID
where
    user.GENDER is not null
group by
    YEAR,
    MONTH,
    user.GENDER
order by
    YEAR,
    MONTH,
    user.GENDER;
--DISTINCT
--문제 풀다 틀렸던 이유 중복 값 제거를 안해서
```

DISTINCT 설명

`DISTINCT`를 예시를 통해 더 자세히 설명해드리겠습니다.

예를 들어, 다음과 같이 `orders` 테이블이 있다고 가정해봅시다. 이 테이블은 주문 정보를 포함하고 있습니다.

```
| order_id | customer_id | product_name |
|----------|-------------|--------------|
| 1        | 101         | Product A    |
| 2        | 102         | Product B    |
| 3        | 101         | Product A    |
| 4        | 103         | Product C    |
| 5        | 101         | Product A    |
```

이제 `product_name` 열에서 중복을 제거하여 고유한 상품 이름만 보여주는 쿼리를 작성해보겠습니다. 이때 `DISTINCT`를 사용합니다.

```
SELECT DISTINCT product_name FROM orders;
```

위 쿼리를 실행하면 다음과 같은 결과가 나옵니다:

```
| product_name |
|--------------|
| Product A    |
| Product B    |
| Product C    |
```

`DISTINCT`를 사용하면 `product_name` 열에서 중복된 값들이 제거되고, 고유한 상품 이름만을 보여줍니다. 즉, "Product A"는 중복되는 값이 있었지만, 결과에선 한 번만 나타납니다.

이렇게 `DISTINCT`를 사용하면 데이터베이스에서 특정 열의 고유한 값을 조회하고 중복을 제거할 수 있습니다.

5일차 문제1 프로그래머스(특정 기간동안 대여 가능한 자동차들의 대여비용 구하기)

```
SELECT
    car.CAR_ID,
    car.CAR_TYPE,
    FLOOR(car.DAILY_FEE * 30 * (1 - plan.DISCOUNT_RATE/100)) as FEE
from
    CAR_RENTAL_COMPANY_CAR car
inner join
    CAR_RENTAL_COMPANY_DISCOUNT_PLAN plan on car.CAR_TYPE = plan.CAR_TYPI
and
    plan.DURATION_TYPE = '30일 이상'
left outer join
    CAR_RENTAL_COMPANY_RENTAL_HISTORY history on history.CAR_ID = car.CAI
```

```
and
    ('2022-11-01' between history.START_DATE and history.END_DATE or '202
where
    plan.CAR_TYPE IN ('세단', 'SUV') and history.CAR_ID is null
and
    500000 <= (car.DAILY_FEE * 30 * (1 - plan.DISCOUNT_RATE/100))
and (car.DAILY_FEE * 30 * (1 - plan.DISCOUNT_RATE/100)) < 2000000

order by
    (car.DAILY_FEE * 30 *(1 - plan.DISCOUNT_RATE/100)) desc, car.CAR_TYPE

--왜 is null이 필요하지?
```

```
SELECT
    car.CAR_ID,
    car.CAR_TYPE,
    FLOOR(car.DAILY_FEE * 30 *(1 - plan.DISCOUNT_RATE/100)) as FEE
from
    CAR_RENTAL_COMPANY_CAR car, CAR_RENTAL_COMPANY_DISCOUNT_PLAN plan
where
    car.CAR_TYPE = plan.CAR_TYPE
and
    plan.CAR_TYPE IN ('세단', 'SUV')
and
    FLOOR(car.DAILY_FEE * 30 *(1 - plan.DISCOUNT_RATE/100)) between 50000
and
    car.CAR_ID = (select CAR_ID from CAR_RENTAL_COMPANY_RENTAL_HISTORY wh
order by
    FEE desc, car.CAR_TYPE, car.CAR_ID desc;SELECT
    car.CAR_ID,
    car.CAR_TYPE,
    FLOOR(car.DAILY_FEE * 30 *(1 - plan.DISCOUNT_RATE/100)) as FEE
from
    CAR_RENTAL_COMPANY_CAR as car
join
    CAR_RENTAL_COMPANY_RENTAL_HISTORY history on history.CAR_ID = car.CAR
and
    ('2022-11-01' between history.START_DATE and history.END_DATE or '202
join
    CAR_RENTAL_COMPANY_DISCOUNT_PLAN plan on car.CAR_TYPE = plan.CAR_TYPE
and
    plan.DURATION_TYPE = '30일 이상'
where
```

```
    plan.CAR_TYPE IN ('세단', 'SUV')
and
    500000 <= FLOOR(car.DAILY_FEE * 30 * (1 - plan.DISCOUNT_RATE/100))
and FLOOR(car.DAILY_FEE * 30 * (1 - plan.DISCOUNT_RATE/100)) < 2000000

order by
    FEE desc, car.CAR_TYPE, car.CAR_ID desc;
--쓰레기 쿼리
```

SQL 모음(2주차)

설명by GPT and 나

1일차 프로그래머스(자동차 대여 기록 별 대여 금액 구하기)

우선 with문을 배우고 알아봅시다…

WITH문

GPT설명

SQL에서 `WITH` 절은 일반적으로 공통 테이블 식(Common Table Expression, CTE)을 정의할 때 사용됩니다. CTE는 하위 쿼리와 비슷한 방식으로 사용되지만, 더 가독성 있고 재사용 가능한 코드 조각을 만들 수 있도록 도와줍니다.

CTE를 사용하여 하위 쿼리의 결과를 임시 테이블로 만들어, 이후의 쿼리에서 그 결과를 사용할 수 있습니다. `WITH` 절의 기본 구문은 다음과 같습니다:

```
WITH cte_name (column1, column2, ...) AS (
    -- CTE 쿼리 정의
    SELECT ...
    FROM ...
    WHERE ...
)
-- CTE 이후에 원하는 SQL 쿼리 작성
SELECT ...
FROM ...
WHERE ...
```

여기서 `cte_name`은 CTE의 이름을 나타내며, `column1`, `column2`, ...은 선택적으로 CTE에서 반환하는 열의 이름을 지정합니다. 그리고 `SELECT` 문 안에서 CTE를 사용할 수 있습니다.

예를 들어, 직원(Employee) 테이블에서 특정 부서(Department)의 평균 급여(Salary)를 계산하는 CTE를 만들어보겠습니다:

```
WITH DeptSalaryAvg AS (
    SELECT Department, AVG(Salary) AS AvgSalary
    FROM Employee
    WHERE Department = 'Sales'
    GROUP BY Department
)
```

```
SELECT Department, AvgSalary
FROM DeptSalaryAvg;
```

이 쿼리에서 `DeptSalaryAvg` CTE는 'Sales' 부서의 평균 급여를 계산하고, 이후의 `SELECT` 문에서 해당 결과를 사용합니다.

CTE를 사용하면 쿼리를 더 가독성 있고 모듈화할 수 있으며, 복잡한 쿼리를 작성하는 데 도움이 됩니다.

내가 보고 안것 에대한 설명

WITH절은 공통 테이블 식인지를 정의 할때 쓴다는데 그냥 쉽게 이해하면 일반적인 프로그래밍언어의 Funtion(함수)같음 내가 하나의 sql을 작성하고 밑에서 그 sql문을 통해 찾은 인자값을 밑에서 호출해서 씀 그냥 서브쿼리를 하나 작성한거에 이름을 정해서 여러번 쓰는 것 같다.

WITH절을 이용한 풀이법

```
WITH final AS (
    SELECT h.history_id,
           h.car_id,
           DATEDIFF(end_date, start_date)+1 AS duration,
           c.car_type,
           c.daily_fee,
           d.discount_rate
    FROM CAR_RENTAL_COMPANY_RENTAL_HISTORY h
    JOIN CAR_RENTAL_COMPANY_CAR c ON c.car_id = h.car_id
    LEFT JOIN CAR_RENTAL_COMPANY_DISCOUNT_PLAN d
        ON d.car_type = c.car_type
           AND d.duration_type =
           CASE
                                WHEN DATEDIFF(end_date, start_date)+1 BE
                THEN '7일 이상'
               WHEN DATEDIFF(end_date, start_date)+1 BETWEEN 30 AND 89
                THEN '30일 이상'
               WHEN DATEDIFF(end_date, start_date)+1 >= 90
                THEN '90일 이상'
                END
    WHERE c.car_type = '트럭')

    SELECT history_id,
           ROUND(daily_fee * duration * (1-COALESCE(discount_rate,0)/100
    FROM final
    ORDER BY fee desc, history_id desc;
--DATEDIFF 설명
#걍 날짜로 덧셈 뺄셈 해줌
```

sql해석 by me

…..with함수를 사용해서 join 문의 case조건이나 car.type등에 일치하는 값들을 받아오고 그값을 밑에 select문을 이용해서 값을 출력함

더 쉽고 맛좋은 풀이

```
SELECT
    history_id,
    round(daily_fee * (timestampdiff(day, start_date, end_date)+1) * (10
    case
        when
            (timestampdiff(day, start_date, end_date)+1) >= 90
        then
            (select
                discount_rate
             from
                car_rental_company_discount_plan
             where
                duration_type = '90일 이상' and car_type = '트럭')
        when
            (timestampdiff(day, start_date, end_date)+1) >= 30
        then
            (select
                discount_rate
             from
                car_rental_company_discount_plan
             where
                duration_type = '30일 이상' and car_type = '트럭')
        when
            (timestampdiff(day, start_date, end_date)+1) >= 7
        then
            (select
                discount_rate
             from
                car_rental_company_discount_plan
             where
                duration_type = '7일 이상' and car_type = '트럭')
        else 0
    end
)) /100) fee
from
    car_rental_company_rental_history history
join
    car_rental_company_car car
where
```

```
    car.CAR_ID = history.CAR_ID and car_type = '트럭'
order by
    fee desc,
    history_id desc;
--내가 푼 코드가 아님
--그래서 설명이라도 함
```

설명 by me

이건 그냥 select문에서 duration_type에 대한 조건을 다해서 다 찾은 뒤에 밑에서 그냥 트럭인 것만 출력한 거

2일차 프로그래머스(우유와 요거트가 담긴 장바구니)

```
with Buy as(
select distinct CART_ID, NAME
from CART_PRODUCTS
where NAME = 'Milk'
union all
select distinct CART_ID, NAME
from CART_PRODUCTS
where NAME = 'Yogurt'
)

select CART_ID from Buy
group by CART_ID
having count(NAME) >= 2
--having
```

HAVING문

SQL의 `HAVING` 절은 데이터베이스에서 그룹화된 결과 집합에 대한 조건을 지정하는 데 사용됩니다. `HAVING` 절은 일반적으로 `GROUP BY` 절과 함께 사용되며, 그룹화된 결과에서 필터링하려는 행을 선택할 때 유용합니다. `HAVING` 절은 집계 함수를 사용하여 그룹화된 데이터의 통계적 특성에 대한 조건을 지정하는 데 주로 사용됩니다.

다음은 `HAVING` 절의 주요 특징과 사용 방법에 대한 설명입니다:

1. 그룹화된 결과에 적용: `HAVING` 절은 `GROUP BY` 절 다음에 나타나며, 그룹화된 결과에 조건을 적용합니다. `GROUP BY` 절은 열을 그룹화하고 각 그룹에 대한 결과 집합을 생성하는 데 사용되고, `HAVING` 절은 이러한 결과 집합에 대한 조건을 지정합니다.
2. 집계 함수와 함께 사용: `HAVING` 절은 주로 집계 함수를 사용하여 그룹화된 데이터의 특성을 평가하는 데 사용됩니다. 일반적인 집계 함수에는 `SUM`, `COUNT`, `AVG`, `MAX`, `MIN` 등이 포함됩니다. `HAVING` 절에서는

집계 함수의 결과를 비교하여 조건을 설정합니다.

3. WHERE 절과의 차이: `WHERE` 절은 데이터를 검색하기 위해 행 수준에서 조건을 지정하는 데 사용되며, `HAVING` 절은 그룹화된 결과 집합에 대한 조건을 지정합니다. 따라서 `HAVING` 절은 그룹화된 데이터에 대한 필터링을 수행하고, `WHERE` 절은 행 수준의 필터링을 수행합니다.
4. 예시: 다음은 `HAVING` 절을 사용한 간단한 SQL 쿼리의 예시입니다.

   ```
   SELECT department, AVG(salary) as avg_salary
   FROM employees
   GROUP BY department
   HAVING AVG(salary) > 50000;
   ```

   위의 쿼리는 'employees' 테이블에서 각 부서별 평균 연봉을 계산하고, 그 평균 연봉이 50,000 이상인 부서만 선택합니다.

`HAVING` 절은 데이터베이스 쿼리에서 그룹화된 데이터에 대한 조건을 지정할 때 매우 유용하며, 데이터 요약 및 필터링에 사용됩니다.

2. 도시 중에 가장 인구가 많은 곳의 인구 수는?

SELECT NAME, max(Population) FROM city;

3. 인구가 가장 많은 도시부터 상위 5개 도시 정보를 조회하시오.

SELECT NAME, Population FROM city ORDER BY Population DESC LIMIT 5;

4. China 의 대통령을 조회하시오.

SELECT HeadOfState FROM country WHERE NAME = 'South Korea';

5. 인구가 가장 적은 국가의 인구수는?

SELECT NAME ,MIN(Population) FROM country;

6. 인구가 가장 적은 국가의 정보를 조회하시오.(서브 쿼리 이용)

SELECT * FROM country WHERE Population = (SELECT MIN(Population) FROM country);

7. 면적이 가장 넓은 국가의 정보를 조회하시오.

```
SELECT * FROM country WHERE SurfaceArea = (SELECT MAX(SurfaceArea) FROM country);
```

8. 면적이 가장 넓은 국가에 속한 도시를 모두 조회하시오.

```
SELECT c.* from country co INNER JOIN city c on co.Code = c.CountryCode WHERE
co.SurfaceArea = (SELECT MAX(SurfaceArea) FROM country);
--다른방법
SELECT * from city WHERE CountryCode = (SELECT CODE FROM country ORDER BY
SurfaceArea DESC LIMIT 1);
#국가 코드별 사용하는 언어의 개수를 많은 수로 출력 만약 같은게 있으면 코드순
SELECT countryCode,COUNT(LANGUAGE) AS l FROM countrylanguage group by CountryCode
ORDER BY l DESC, CountryCode ASC;
#기대 수명이 가장높은 나라 10개
SELECT NAME, LifeExpectancy FROM country WHERE NOT LifeExpectancy is NULL ORDER BY
LifeExpectancy DESC LIMIT 10;
#각대륙별 평균 수명
SELECT NAME FROM country WHERE
UNION all
select Continent, max(LifeExpectancy) FROM country GROUP BY Continent;
#각대륙별 평균 기대 수명을 조회
SELECT Continent, LifeExpectancy from country where NOT LifeExpectancy is NULL GROUP BY
Continent;
```

TypeScript

문법은 js와 거의 동일

TypeScript는 상수나 변수의 type을 추리 할 수도 있고 개발자가 정해줄 수도 있다.

Player라는 Object를 생성할때 age는 있거나 없어도 되고 name은 꼭 필요하다고 할때

```
const player {
    name:string,
    age?:number
} = {
    name: "me"
}
```

이런식으로 구현이 가능하다.

: 을 이용해서 변수나 상수의 type을 정해줄 수 있는데 ?를 추가 하면 선택적 변수 취급한다.

```
const player {
    name:string,
    age?:number
} = {
    name: "me"
}

//if(player.age < 10){
// 이런식으로 하면 에러남 이유는 age가 undifind일 수 있어서임
//}

if(player.age && palyer.age < 10){
//이렇게 하면 age가 존재한다는 것을 확인시켜줌
}
```

```
type player {
    name:string,
    age?:number
}
funtion playerMaker(name:string) : Player{
```

```
    return {
        name
    }
}
```

type 스크립트는 무려 type을 정의할 수 있다!!!!

```
type player {
    name:string,
    age?:number
}
const playerMaker = (name:string) : Player => ({name})
const me = playerMaker("me")
me.age = 18
```

typescript는 readonly로 만들 수 있음

```
const numbers: readonly number[] = [1, 2, 3, 4]
//이렇게 만들면 readonly가 됨
number.push(5) //error readonly이기 때문
```

typescript는 type이 3가지 더 있음

```
never, unknown, void
```

```
type Add = (a : number, b : number) => number
const add : Add =(a, b) => a + b
```

오버로딩 구현 다형성

```
type Add = {
    (a: number, b :number) :number,
    (a: number, b :number, c :number) :number
}

const add:Add =(a,b,c?:number){
    if(c) return a + b + c
    return a + b
}
//이렇게 하면 c는 옵션이라는 것을 TypeScript가 인식을 함
```

<> 제너릭: 제너릭은 프로그램을 추상화 시키기 위한 것입니다. 이는 코드의 재사용성과 유연성을 높일 수 있습니다.

▼ 추상화란?

프로그래밍에서의 추상화는 복잡한 시스템이나 개념을 간소화하거나 특정 측면에 집중하기 위해 세부 사항을 감추는 프로세스입니다.

추상화는 캡슐화와 비슷해보이지만 둘은 서로 다른 측면을 강조 하고 있습니다.

추상화는 개념을 단순화하고 이해하기 쉽게 만드는 것에 중점을 두며, 캡슐화는 객체의 상태와 행동을 보호하고 외부에 노출되는 것을 제한하여 시스템을 더 안전하게 만드는 데 중점을 둡니다.

```
function SuperPrint<T>(a :T[]){
    return a[0]
}

const a = superPrint([1, 2, 3, 4])
const b = superPrint([true, true, false])
const c = superPrint(["a", "b", "c"])
const d = superPrint([1, "s", true])
```

```
type SuperPrint = {
    <T>(a: T[]): T
}

const superPrint: SuperPrint = (a) => a[0]

const a = superPrint([1, 2, 3, 4])
const b = superPrint([true, true, false])
const c = superPrint(["a", "b", "c"])
const d = superPrint([1, "s", true])
```

추상메소드와 접근제어자

```
abstract class User {
    constructor(
        protected lastname : string,
        protected firstname : string,
        public nickname : string
    ){}

    abstract getNickname():void

    getFullName(){
        return `${this.firstname} ${this.lastname}`
    }
```

```
}

class Player extends User{
    getNickname(): void {
        console.log(this.nickname)
    }
}

const park = new Player("Hyun", "seong", "Cloushaar")
```

단어를 카테고리 별로 나눠서 추가하는 사전임

```
type Words = {
    [key: string]: string;
};

class Word {
    constructor(public term: string, public def: string) {}
}

class Dict {
    private words: Words;
    constructor() {
        this.words = {};
    }
    add(word: Word) {
        if (this.words[word.term] === undefined) {
            this.words[word.term] = word.def;
        }
        else {
            console.log("이미 존재하는 단어입니다.");
        }
    }
    def(term: string) {
        return this.words[term];
    }
    update(word: Word) {
        if (this.words[word.term] !== undefined) {
            this.words[word.term] = word.def;
        }
        else{
            console.log("존재하지않는 단어입니다.");
        }
    }
    del(term: string) {
```

```
        if (this.words[term] !== undefined) {
            delete this.words[term];
        }
        else{
            console.log("존재한지않는 단어입니다.")
        }
    }
}

const potato = new Word("potato", "감자");
const dict = new Dict();
dict.add(potato);
```

```
type Team = "red" | "blue" | "green"

interface User{
    Name: string
    team: Team
}

interface User{
    Age: number
}

interface Player extends User{
    nickname: string,
    job: string
}

const player: Player = {
    Name: "hyun",
    Age: 18,
    team: "blue",
    nickname: "cloushaar",
    job: "tanker"
}
```

interface를 사용할 수 도있는데 이게 type이랑 비슷함 차이점은 같은 이름으로 여러개를 만들어도 합쳐줌 그리고 interface는 class와 비슷하게 사용할 수 있음 만약 type을 이용해 상속할려면

보통 object의 모양을 typescript한테 알려줄때 interface를 사용하고 나머지 경우는 type을 사용함

interface끼리 상속할때는 extends를 사용하지만 class한테 상속할때는 일반적인 객체지향 프로그래밍 언어와 똑같이 implements를 사용함

```
type Team = "red" | "blue" | "green"
type User = {
    Name: string
    team: Team
}


type Player = User &{ //이부분에 &가 있는 이유는 Player라는 type이 User와 그뒷부
    nickname: string,
    job: string
}

const player: Player = {
    Name: "hyun",
    team: "blue",
    nickname: "cloushaar",
    job: "tanker"
}
```

이런 식으로 함

다형성 구현

예시(localstrage API를 대충 시뮬레이션 해봄)

```
interface SStorage<T> {
    [key:string]: T
}

class LocalStorage<T>{
    private storage: SStorage<T> ={}
    set(key:string, value:T){
        if(key !== undefined){  //만약 이미 존재하는 키일 경우 작동
            console.log("이미 존재하는 키입니다.");
        }
        else{this.storage[key] = value;}
    }
    remove(key:string){
        delete this.storage[key];
    }
    get(key:string):T {
        return this.storage[key];
    }
    clear(){
        this.storage = {};
```

```
    }
}
```

블록체인 구현

```
import crypto from "crypto";
import { createTextChangeRange } from "typescript";

interface BlockShape{
    hash: string;
    prevHash: string;
    height: number;
    data: string;
}

class Block implements BlockShape{
    public hash: string;
    constructor(
        public prevHash: string,
        public height: number,
        public data: string
    ) {
        this.hash = Block.calculateHash(prevHash, height, data);
    }
    static calculateHash(prevHash: string, height:number, data:string){
        const toHash = `${prevHash}${height}${data}`;
        return crypto.createHash("sha256").update(toHash).digest("hex");
    }
}

class Blockchain{
    private blocks: Block[]
    constructor(){
        this.blocks = [];
    }
    private getPrevHash(){
        if(this.blocks.length === 0) return ""
        return this.blocks[this.blocks.length - 1].hash;
    }
    public addBlock(data:string){
        const newBlock = new Block(this.getPrevHash(), this.blocks.lengt
        this.blocks.push(newBlock);
    }
    public getBlocks(){
```

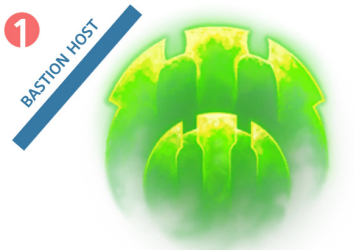
```
        return [...this.blocks];
    }
}
/*
const blockchain = new Blockchain();

blockchain.addBlock("First one");
blockchain.addBlock("Second one");
blockchain.addBlock("Third one");

console.log(blockchain.getBlocks());
*/
```

AWS

▼ Bastion 호스팅

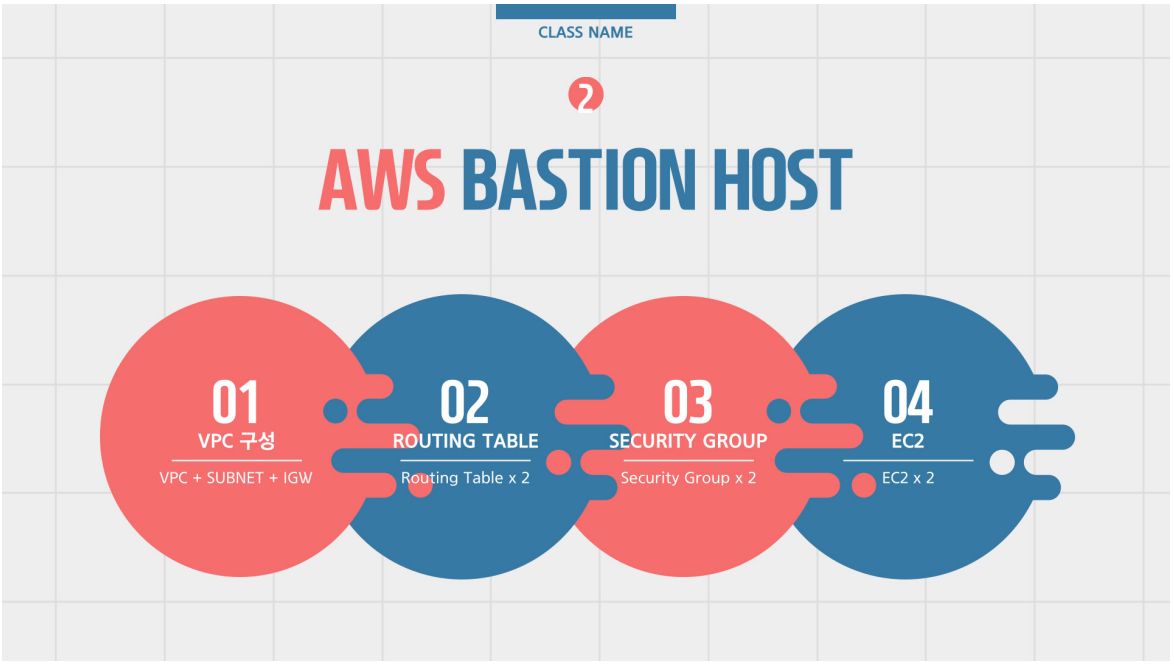


BASTION HOST란?

BASTION HOST는 무엇인가?

Bastion이란? (영어사전)
: 수호자, 보루, 요새

BASTION HOST는 public 네트워크에서 private 네트워크에 대한 엑세스를 제공하기 위한 목적을 가진 서버입니다.
쉽게 말해, 침입 차단 소프트웨어가 설치되어 내부와 외부 네트워크 사이에서
일종의 게이트 역할을 수행하는 호스트를 뜻합니다.

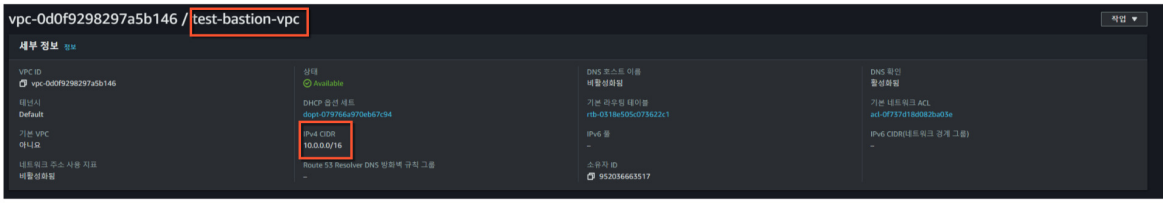


AWS와 BASTION HOST

구 성 하 기

BASTION HOST 3

VPC 구성



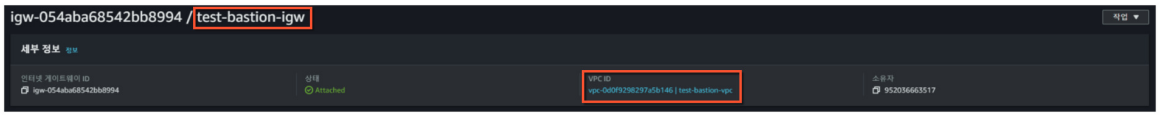
NAME TAG: test-bastion-vpc CIDR: 10.0.0.0/16

AWS와 BASTION HOST

구 성 하 기

BASTION HOST 4

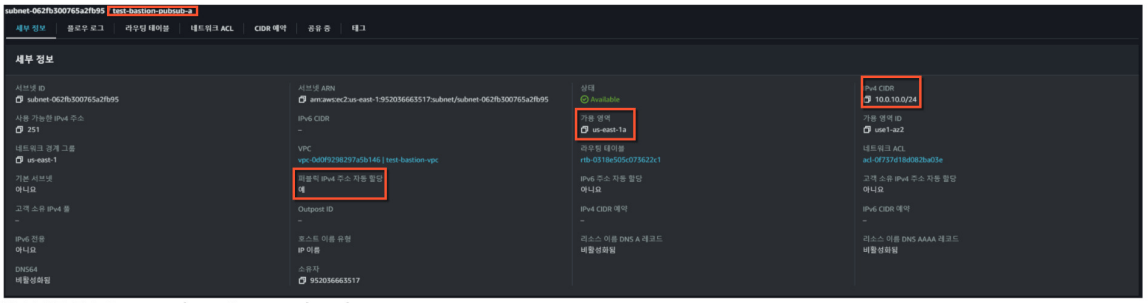
ITERNET GATEWAY 구성



NAME TAG: test-bastion-igw ATTACH: test-bastion-vpc

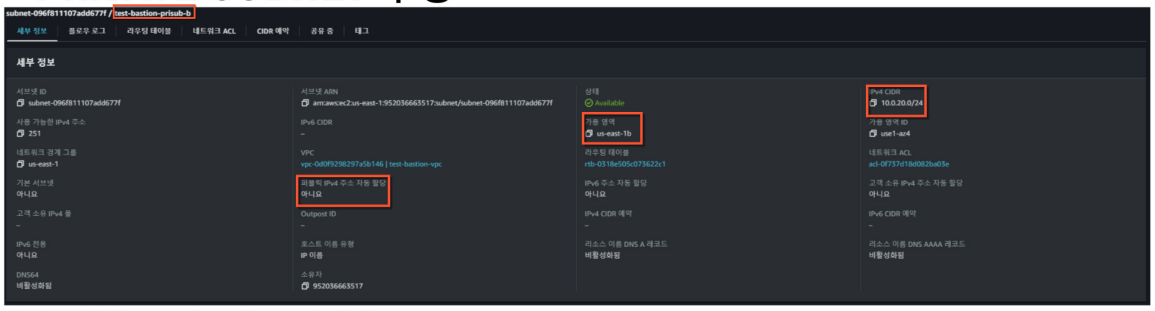
AWS 직접 BASTION HOST 구성하기

PUBLIC SUBNET 구성



NAME TAG: test-bastion-pubsub-a
CIDR : 10.0.10.0/24 AZ : us-east-1a
퍼블릭 IPv4 주소 자동 할당 활성화

PRIVATE SUBNET 구성



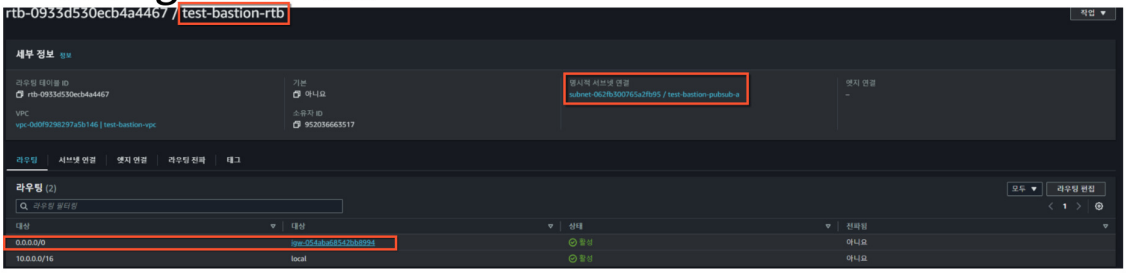
NAME TAG: test-bastion-prisub-b
CIDR : 10.0.20.0/24 AZ : us-east-1b
퍼블릭 IPv4 주소 자동 할당 비활성화

AWS 아키텍처 BASTION HOST

BASTION HOST 7

구 성 하 기

Routing Table 구성



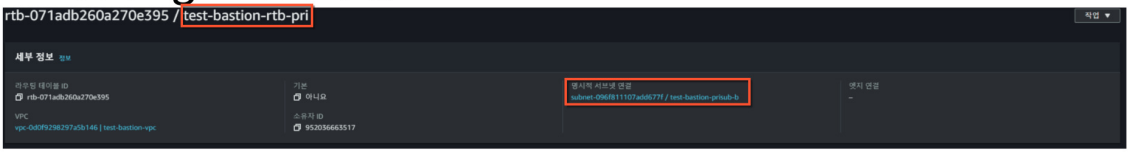
NAME TAG: test-bastion-rtb
Subnet : test-bastion-pubsub-a
Add Routing : 0.0.0.0/0 → test-bastion-igw

AWS 아키텍처 BASTION HOST

BASTION HOST 8

구 성 하 기

Routing Table 구성



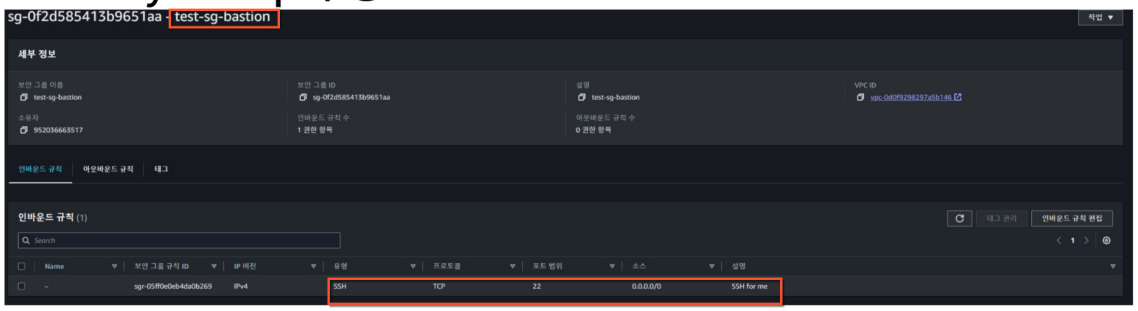
NAME TAG: test-bastion-rtb-pri
Subnet : test-bastion-prisub-b

AWS와 함께 BASTION HOST

9 BASTION HOST

구 성 하 기

Security Group 구성



NAME TAG: test-sg-bastion
VPC : test-bastion-vpc

Inbound Rule
Type : SSH
Source : 내 IP 또는 위치 무관

AWS와 함께 BASTION HOST

10 BASTION HOST

구 성 하 기

Bastion EC2 구성

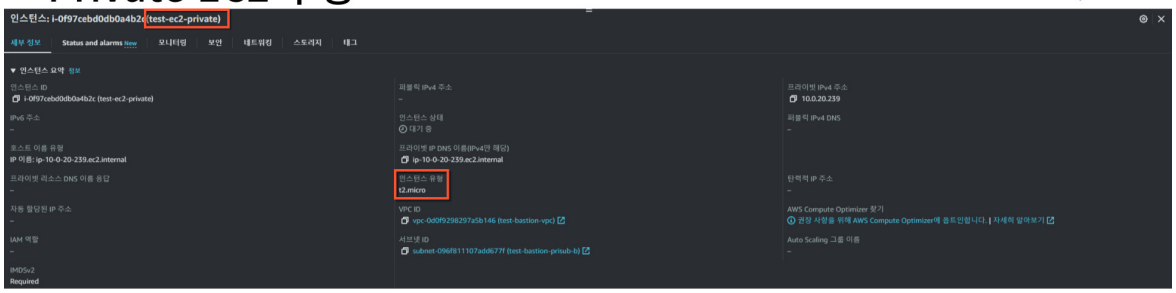
NAME TAG: test-ec2-bastion
보안 그룹 : test-sg-bastion

AMI : Amazon Linux 2 AMI (HVM)
instance type : t2.micro (기본값)
Subnet : test-bastion-pubsub-a

AWS에서 BASTION HOST

구 성 하 기

Private EC2 구성



NAME TAG: test-ec2-private

보안 그룹 : test-sg-private

AMI : Amazon Linux 2 AMI (HVM)
instance type : t2.micro (기본값)
Subnet : test-bastion-prisub-b

결과

Bastion서버에서
private 서버에 핑 가짐 ㅇㅇ

```
[ec2-user@ip-10-0-10-202 ~]$ ping 10.0.20.239
PING 10.0.20.239 (10.0.20.239) 56(84) bytes of data.
64 bytes from 10.0.20.239: icmp_seq=1 ttl=255 time=0.995 ms
64 bytes from 10.0.20.239: icmp_seq=2 ttl=255 time=0.568 ms
64 bytes from 10.0.20.239: icmp_seq=3 ttl=255 time=0.693 ms
64 bytes from 10.0.20.239: icmp_seq=4 ttl=255 time=0.617 ms
64 bytes from 10.0.20.239: icmp_seq=5 ttl=255 time=0.613 ms
64 bytes from 10.0.20.239: icmp_seq=6 ttl=255 time=0.618 ms
64 bytes from 10.0.20.239: icmp_seq=7 ttl=255 time=0.601 ms
64 bytes from 10.0.20.239: icmp_seq=8 ttl=255 time=0.636 ms
64 bytes from 10.0.20.239: icmp_seq=9 ttl=255 time=0.637 ms
64 bytes from 10.0.20.239: icmp_seq=10 ttl=255 time=0.603 ms
64 bytes from 10.0.20.239: icmp_seq=11 ttl=255 time=0.612 ms
64 bytes from 10.0.20.239: icmp_seq=12 ttl=255 time=0.594 ms
64 bytes from 10.0.20.239: icmp_seq=13 ttl=255 time=0.643 ms
64 bytes from 10.0.20.239: icmp_seq=14 ttl=255 time=0.591 ms
^C
--- 10.0.20.239 ping statistics ---
14 packets transmitted, 14 received, 0% packet loss, time 13276ms
rtt min/avg/max/mdev = 0.568/0.644/0.995/0.103 ms
```

질문 받습니다.

▼ S3정적 웹 사이트 호스팅

▼ 정적 웹 사이트 호스팅이란?

정의:

정적 웹 사이트 호스팅은 웹 사이트를 이루는 파일과 콘텐츠가 고정되어 있는 경우에 사용되는 호스팅 방식입니다. 이는 주로 HTML, CSS, JavaScript 등의 정적 파일들로 이루어진 웹 사이트에 적합하며, 서버 측에서 동적으로 변경되지 않는 특징을 가지고 있습니다. 정적 웹 사이트는 사용자에게 항상 동일한 콘텐츠를 제공합니다.

장점:

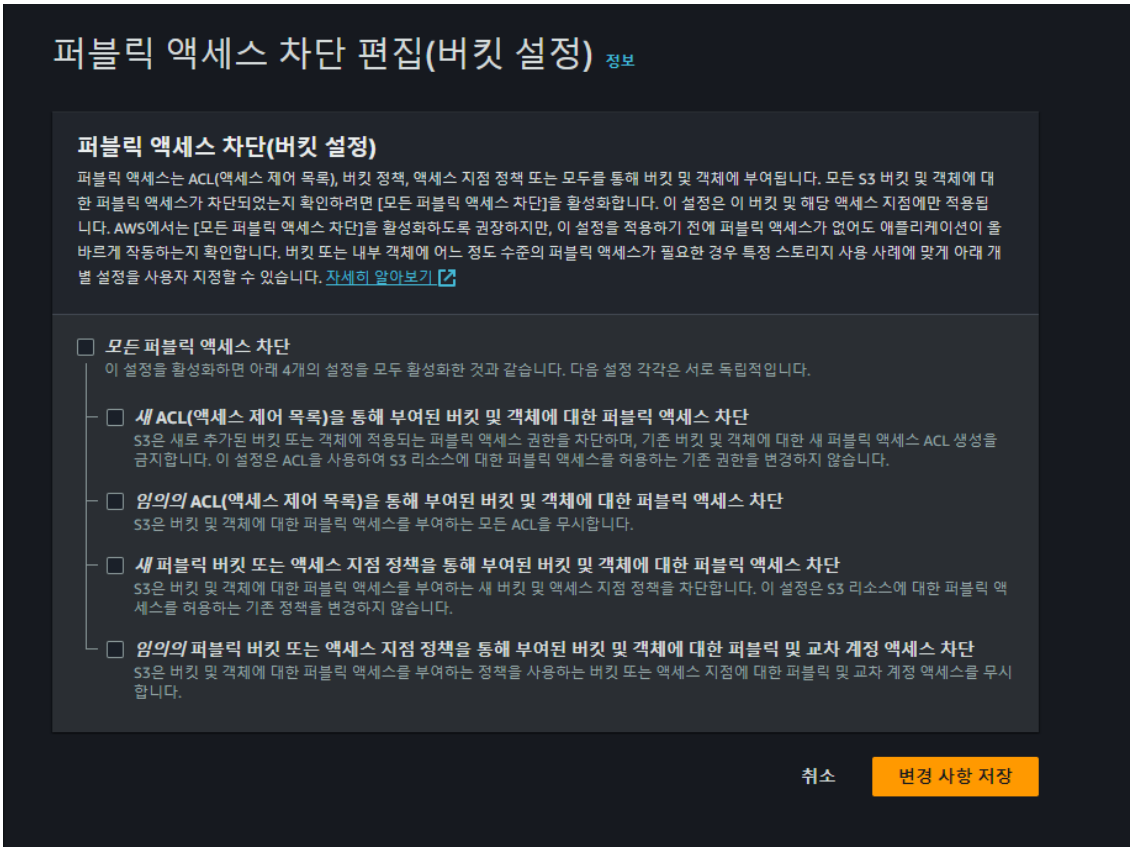
1. **빠른 로딩 속도:** 정적 파일을 사용하기 때문에 서버에서 동적으로 생성할 필요가 없어 빠른 로딩 속도를 제공합니다.
2. **간편한 관리:** 정적 파일이 고정되어 있기 때문에 관리가 간편하며, 호스팅 서비스들은 파일을 저장하고 전달하는 역할만 수행합니다.
3. **저렴한 비용:** 동적 웹 사이트에 비해 서버 측에서의 데이터 처리가 적기 때문에 호스팅 비용이 저렴합니다.
4. **강력한 보안:** 정적 파일만을 다루므로 보안 측면에서 강점을 가집니다.

단점:

1. **동적 기능 부재:** 서버 측에서 동적으로 데이터를 처리할 수 없어, 일부 고급 기능이 불가능할 수 있습니다.
2. **콘텐츠 업데이트 어려움:** 웹 사이트 내용을 업데이트하려면 파일을 직접 수정하고 업로드해야 하므로 동적인 업데이트가 어려울 수 있습니다.
3. **대규모 트래픽 처리 어려움:** 매번 요청마다 파일을 전달하기 때문에 대규모 트래픽 처리에 한계가 있을 수 있습니다.

▼ S3정적 웹사이트 호스팅 구성

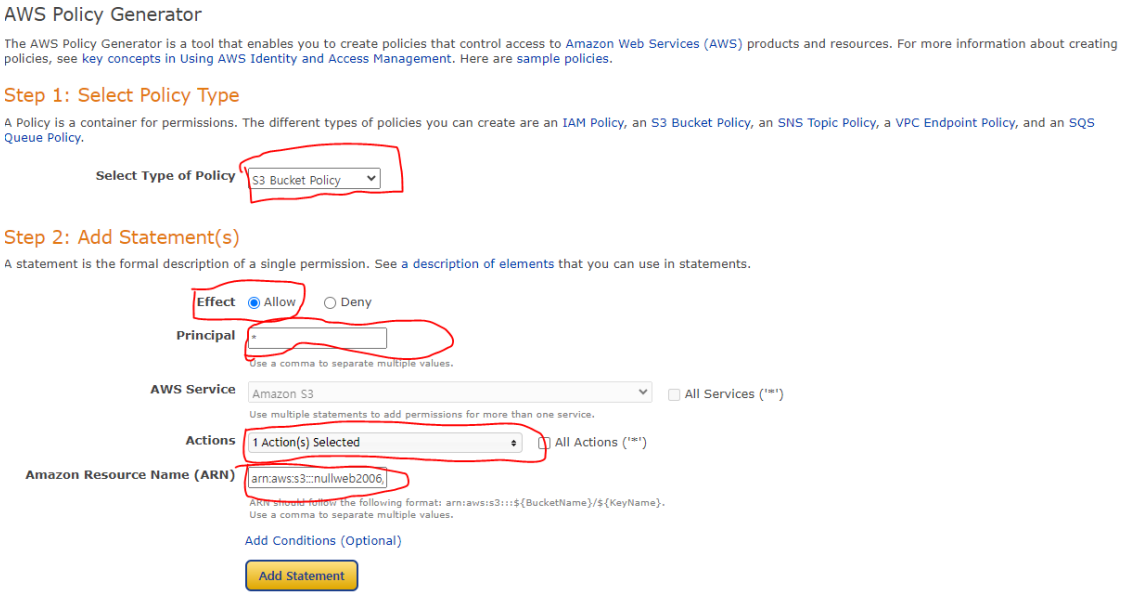
버킷의 퍼블릭 엑세스 차단 설정



모든 퍼블릭 엑세스를 허용 이유는 불특정 다수의 사용자가 웹에 엑세스해야 하기 때문

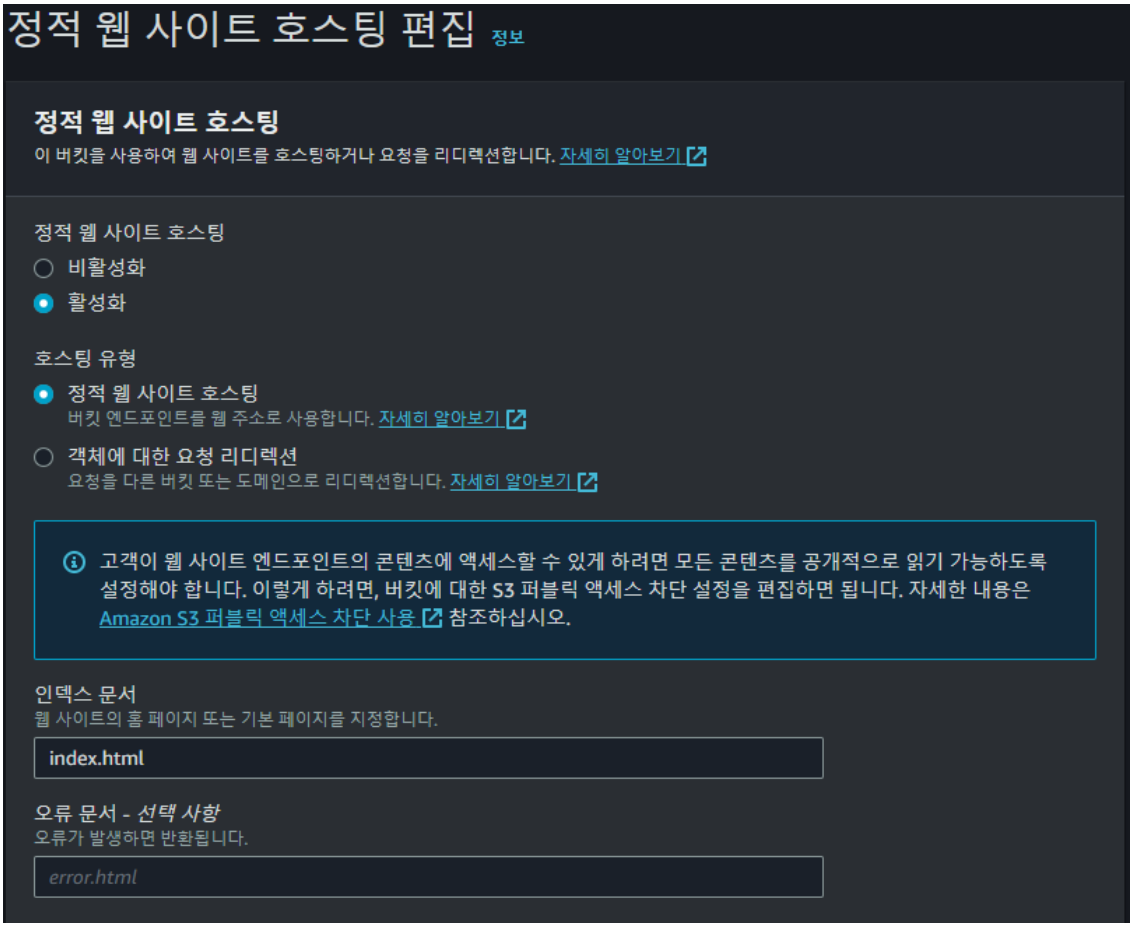
정책 생성

정책 편집에서 정책 생성기 페이지로 넘어갑니다.



빨간색 위주로 설정 Actions는 Get Object만 선택 ARN은 자신의 버킷을 ARN을 복사하고 뒤에 /*을 넣어줍니다. 이것은 버킷안의 모든 파일에 적용하겠다는 것

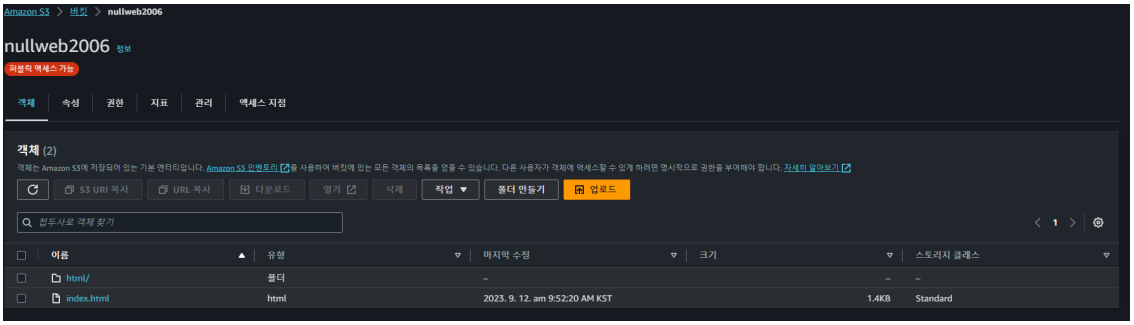
정적웹 호스팅 활성화



정적 웹 호스팅을 활성화한 뒤 호스팅 유형을 선택합니다. 만약 개인이 소유한 도메인을 사용하고자 하는 경우에는 S3버킷의 이름을 해당 도메인의 이름과 동일학게 해야 합니다.

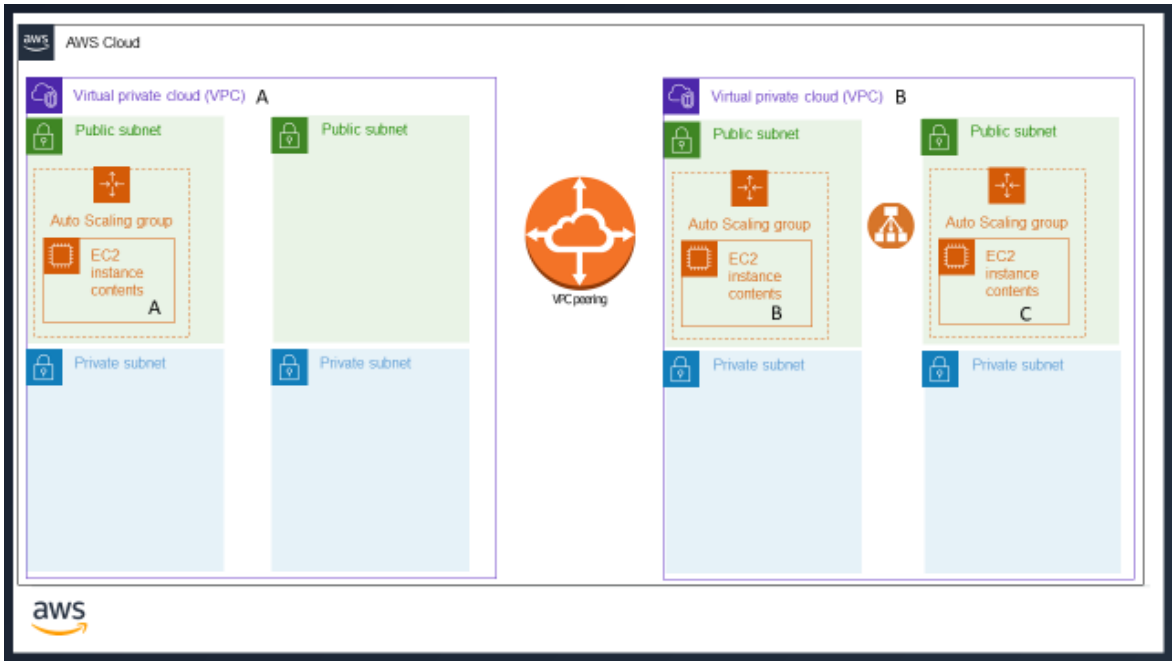
그후 인덱스 문서를 지정해주고 오류 문서가 있는 경우 설정해주면 됩니다.

객체 업로드



객체 업로드를 해준뒤 엔드포인트나 도메인으로 접속하면 됩니다.

▼ VPC peering + Load Balancer



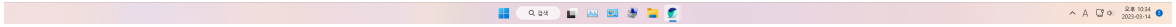
대충 만들어본 아키택처

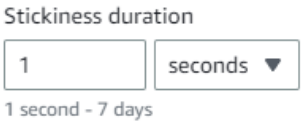
```
64 bytes from 44.201.241.156: icmp_seq=1 ttl=254 time=0.557 ms
64 bytes from 44.201.241.156: icmp_seq=2 ttl=254 time=0.579 ms
64 bytes from 44.201.241.156: icmp_seq=3 ttl=254 time=0.619 ms
64 bytes from 44.201.241.156: icmp_seq=4 ttl=254 time=0.561 ms
64 bytes from 44.201.241.156: icmp_seq=5 ttl=254 time=0.610 ms
64 bytes from 44.201.241.156: icmp_seq=6 ttl=254 time=0.501 ms
64 bytes from 44.201.241.156: icmp_seq=7 ttl=254 time=0.608 ms
64 bytes from 44.201.241.156: icmp_seq=8 ttl=254 time=0.554 ms
64 bytes from 44.201.241.156: icmp_seq=9 ttl=254 time=0.603 ms
64 bytes from 44.201.241.156: icmp_seq=10 ttl=254 time=0.602 ms
64 bytes from 44.201.241.156: icmp_seq=11 ttl=254 time=1.00 ms
64 bytes from 44.201.241.156: icmp_seq=12 ttl=254 time=0.526 ms
64 bytes from 44.201.241.156: icmp_seq=13 ttl=254 time=0.573 ms
64 bytes from 44.201.241.156: icmp_seq=14 ttl=254 time=0.552 ms
64 bytes from 44.201.241.156: icmp_seq=15 ttl=254 time=0.545 ms
64 bytes from 44.201.241.156: icmp_seq=16 ttl=254 time=0.688 ms
64 bytes from 44.201.241.156: icmp_seq=17 ttl=254 time=0.514 ms
64 bytes from 44.201.241.156: icmp_seq=18 ttl=254 time=0.630 ms
^C
--- 44.201.241.156 ping statistics ---
18 packets transmitted, 18 received, 0% packet loss, time 17353ms
rtt min/avg/max/mdev = 0.501/0.601/1.008/0.111 ms
```

VPC 2개를 Peering한후 연결됬는지 확인하기 위해 핑을 보내보았다

처음 보냈을때 안되었었는데 이유는 ICMP포트를 안열어놔서 였다

Web_Service_B





Web_Service_C