Local 표준 환경 안내 : Server Local, PC Local 환경 설정 설명

- **Python 은 PC Local 동작 시 격리된 가상 환경이 필요**합니다. (Server Local 도 동일)
  향후 상용 서비스 구성 시에는, 특히 PC 또는 Server Local 에서 Python 가상환경별 **버전관리 그리고 이에 따른 패키지별 종속성 등이 체계적으로 관리**되어야 합니다.
- Server Local 환경에서는 "Container + Python 가상 환경 또는 선택적으로 + Poetry 추가" 적용이 표준입니다.
- 일반적으로 컨테이너 기반 기술을 적용하기 위해서는 도커(Docker)를 사용합니다. 도커 (Docker)를 사용하면 실행하고자 하는 프로세스만 격리된 환경에서 실행하는 것이 가능합니다. 이를 이용해 손쉽게 프로세스를 격리할 수 있을 뿐만 아니라, 격리된 환경을 이미지로 만들어서 어디서든 똑같이 동작하는 컨테이너를 생성할 수 있습니다.
- 도커 (Dokcer)는 리눅스 OS 에서 동작하는 것이며, 그 외의 OS 에서 동작시키려면 Docker Desktop 을 설치해야 합니다. 이번 과정에서는 Python 가상환경으로 처리할 것이므로, 생략합니다. 도커(Docker)는 경량 가상화 기술 리눅스 컨테이너를 구현하는 애플리케이션입니다. 참고로 여러 개의 컨테이너를 관리하기 위한 기술로 도커 컴포우즈 (Docker Compose) 또는 쿠버네티스 (Kubernetes) 기술이 있습니다. 쿠버네티스는 대규모 분산 환경에서 사용합니다.
- 참고로, PC Local 에서 Container 와 Poetry 는 적용 여부는 사용자의 선택에 따릅니다. (일부 환경 편향을 해결하는 셋팅 필요 → 예, 맥북 O 및 윈도우 X 발생 사례 등)
- 실제 프로젝트에서 배포 및 동작 환경은, 컨테이너 기반 리눅스 환경이 많이 사용됩니다. 현장에서는 자체 검증한 SBOM(Software Bill of Materials) 구성으로 패키징한 컨테이너 이미지 기반으로 빌드 배포를 진행하기 때문입니다.
- 본 환경 설정 가이드에서는 PC Local 구성 환경을 안내하며, 향후 모델 서빙 등 과정으로 Server Local 환경으로 들어가게 되면, 컨테이너 기반 또는 Poetry 를 이용하여 Server Local 환경에서는 Container 이미지 기반으로 각 프로젝트별 패키지 및 버전 종속성 관리를 진행하는 것을 추천합니다.

Visual Studio Code 설치

https://code.visualstudio.com/download

VS Code 추가 확장자 설치

윈도우: Ctl + Shift + X 또는 Mac : Cmd + Shift + X

- (필수) Python, Jupyter Notebook
- (선택) Pylance, Python Indent, Python Docstring Generator

VS Code 필수 패키지 추가 설치 (★ 부록 **필수 패키지 설치 가이드**: 2.**practice_dev-env-package-add 참조**)

▪ C++ build tools
▪ Windows 10 SDK 또는 Windows 11 SDK
▪ MSVC v143 - VS 2022 C++ x64/x86 build tools
▪ C++ CMake Tools for Windows

Mac 사용자 :
Homebrew 를 이용하여 필요한 도구 설치, 다음 각 라인에서 해당 명령어를 Copy & Paste 합니다.

> brew --version → 이 결과 값이 나오면, 바로 6 번째 > 라인으로 넘어갑니다.
> xcode-select --install
> /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
> echo 'eval "$(/opt/homebrew/bin/brew shellenv)"' >> ~/.zshrc
> source ~/.zshrc

> brew install cmake → brew 가 이미 설치되어 있는 경우, 여기부터 시작
> brew install gcc
> brew install llvm

Python 설치 (Python ~ VS Code 연결)

https://www.python.org/downloads ▪ **3.11** 이상 설치 권고, 윈도우 설치 시에는 반드시 경로 자동 설정 옵션 체크
> python --version

▪ Mac 은 설치 후 경로 설정 필수

맥 사용자 :
> brew --version
> brew install python

Mac 이나 별도 Linux 서버에서, /config/.local/bin 과 같이 로컬 또는 새로운 폴더에 설치한 경우, /config 폴더 밑에 경로를 직접 잡아 줘야 합니다.
> vi ~/.zshrc (~/ 는 현재 사용자의 홈 디렉토리를 나타내는 특수 기호임)
> export PATH=$PATH:/config/.local/bin
> source ~/.zshrc

> python3 –version

Python 가상환경 및 VS Code 연결 준비

Ctrl + Shift + X (Windows) 또는 Cmd + Shift + X (Mac)
검색창에 Python 입력 후 확장 프로그램 설치

윈도우 사용자 :

1. 프로젝트 디렉토리 생성 및 이동
> mkdir my_project
> cd my_project

2. 가상환경 생성
> python -m venv 가상환경이름

3. 가상환경 활성화
> 가상환경이름₩Scripts₩Activate

Mac 사용자 :

1. 프로젝트 디렉토리 생성 및 이동
> mkdir my_project
> cd my_projects

2. 가상환경 생성
> python3 -m venv 가상환경이름

3. 가상환경 활성화 (방금 만든 가상환경 내에 bin/activate 실행)
> source 가상환경이름/bin/activate

(가상환경이름) 표시되면 완료
* 가상환경 해제는, 가상환경에서 > deactivate 입력

VS Code에서 Python 가상환경 연결

1. VS Code 에서 프로젝트 폴더 열기 (선택, 경로 설정 시 동작)
> code .

2. VS Code 에서 가상환경 설정
Ctrl + Shift + P (Windows) 또는 Cmd + Shift + P (Mac)
Python : Select Interpreter 입력 후 실행
./가상환경이름/bin/python 또는 ./가상환경이름/Scripts/python.exe 선택

3. VS Code 에서 터미널 열기 (Ctrl + ~)
> source 가상환경이름/bin/activate # Mac/Linux
> 가상환경이름₩Scripts₩Activate # Windows

여러 프로젝트 환경 유지

1. 새로운 프로젝트 디렉토리 생성
> mkdir another_project
> cd another_project

2. 필요한 새 가상환경 개별 생성
> python -m venv 가상환경이름

3. VS Code 에서 Python : Select Interpreter 로 프로젝트별 가상환경 선택 가능

다음은 PC 에서 DBMS 구성 시 선택합니다. (선택 사항)

DBMS ~ Client 설치 및 접속

엔터프라이즈(기업 또는 조직)에서 비즈니스(업무)를 데이터 모델링으로 설계하고, 이후 이 구조에 따라서 대상 DBMS 에 테이블들과 레코드들이 생성하게 됩니다.
SQL 은 DBMS 에서 동작시키는 언어입니다. 따라서 DBMS 의 선택에 영향을 받습니다.
전통적으로는 Oracle (1977)이라는 회사 제품이 80 년대부터 많이 사용되어 시장 점유율이 높습니다. 전통적으로 대량의 복잡한 업무 트랜잭션(제조, 금융, 공공 등)에 대한 처리가 필요한 경우에 상용 DBMS 가 권장됩니다.
하지만 온라인 쇼핑물, 거래 플랫폼 등과 같이 각 사용자들이 단위 업무들이 개별적으로 처리되는 경우에는 아무리 접속자와 사용량이 많더라도 복잡한 트랜잭션들에 대한 처리가 발생하지 않기 때문에, 무료인 오픈소스 DBMS 를 활용하기도 합니다. 또는 시계열성 로그나 메타정보를 기록하고 관리하는 목적으로도 사용합니다.

Oracle 과 MariaDB 간 SQL 실습 차이점은, 내장함수와 메타정보 구성이 일부 다른 점이 있습니다. 또한 DBMS 에게 지시하는 힌트문의 활용방법, SQL 실행계획과 실행결과를 보는 법에서 차이가 있습니다. 내장함수와 메타정보 구성이 다른 점은 과정내에서 같이 설명을 하고 있습니다.
예를 들어 SQL 실행 계획은, Maria DB 는 SQL 앞에 explain 을 붙여서 실행합니다.
Oracle 은 explain plan for 를 붙여서 실행하는 차이가 있습니다.

본 과정에서는 실습 환경은, 오픈소스 DBMS 인 MariaDB 를 사용하며 자체 검증한 SBOM (Software Bill of Materials) 구성으로 패키징한 컨테이너 이미지 기반으로 진행합니다.

MariaDB 설치

https://mariadb.org/download/

수동 설치 (서버에 컨테이너 기반으로 상용 서비스 구성에서 선택, 별도 yaml 구성하여 진행, 파일 이용은 필요 시 별도 안내) :
C:₩Windows₩System32₩drivers₩etc₩localhost 확인
my.cnf 파일 read-only 체크, yaml 파일 위치로 이동
docker-compose –f maria_db.yaml up
(-d 옵션 백그라운드 띄우기, 컨테이너 로그파일 확인 목적 시 불필요)

MySQL Workbench 설치 : 데이터 모델링 도구 및 DBMS 접속 도구 사용

https://dev.mysql.com/downloads/workbench/

다음은 PC에서 컨테이너 구성 시 적용합니다. (선택 사항)

윈도우 커맨드 (파워 쉘) 실행 : 윈도우 PC에서 컨테이너 구성 시 진행, Mac은 불필요

- wsl 설치 (Windows Subsystem for Linux) 2 : 윈도우에서 리눅스기반 실행을 지원하는 기능

> wsl –install
> wsl –set-default-version 2

Docker Desktop 설치 : PC에서 컨테이너 구성 시 진행

- 윈도우 및 맥 환경에서 Docker Engine 을 실행하는 GUI 도구로, 컨테이너 관리, 가상화 환경 제공, Docker Compose, 저장소(Volume) 및 네트워크 관리 등 역할 수행 (참고로, 리눅스 서버 환경에서는 Docker Desktop 없이 Dockerd 로 Docker Engine 을 바로 사용 가능)

https://www.docker.com/products/docker-desktop

- 향후 상용 프로젝트 시에는, 도커기반으로 컨테이너 이미지를 생성해서 사용 권고
- PC 에 직접 특정 프로그램을 설치하지 않고, 상용 서비스를 고려한 도커 컨테이너 기반 구성
- 참고로 Mac 은 수동 설치도 가능

> brew install --cask docker
> open /Applications/Docker.app

Docker Desktop 설치 후 WSL2 백엔드 활성화 : 윈도우 PC에서 컨테이너 구성 시 진행, Mac은 불필요

- Docker Desktop → Settings → General → "Use the WSL 2 based engine" 체크
- 파워 쉘 또는 CMD 에서

> docker ps → 실행중인 컨테이너 목록 확인₩

> docker ps -a → 전체 컨테이너 목록 확인

> docker exec –it 3de86e71b201 /bin/bash (예 3de86e71b201 는 컨테이너명, 윈도우의 경우 /bin/bash → sh 로 대체)
: 전체 컨테이너 접속하여 확인

> docker start 컨테이너 ID → 컨테이너 시작

```
> docker stop 컨테이너 ID → 컨테이너 멈춤

> docker rm 컨테이너 ID → 컨테이너 삭제
```