**□ 모듈프로젝트 산출물**

클라우드 보안 융합 전문가 양성 과정

|  |
| --- |
| 모듈프로젝트 1 |

**2021년 10월 13일**

이름 : 5조(양인모, 배정원, 김수찬, 주지현)

목 차

1. 개요

#1. 문서 개요

#2. 프로그램 개요

1. 프로그램 설명

#1. 프로그램 요구사항

#2. 일정

#3. Dockerfile & Docker-compose

#4. 프로그램 동작 원리

Ⅰ. Main

Ⅱ. DDL

Ⅲ. DML

#5. 프로그램 기능

개요

#1. 문서 개요

• 본 문서는 회원관리 프로그램 개발을 위한 요구사항을 작성한다.

• 본 문서는 회원관리 프로그램의 기능과 프로그램 실행 결과를 간략하게 설명한다.

#2. 프로그램 개요

○ 프로그램의 목적

-이 프로그램은 컨테이너 내부 테이블에 있는 데이터를 이용하여 CLI 기반의 회원서비스를 구성한다.

-이 프로그램은 회원의 개인정보(이름, 나이, 전화번호, 이메일)데이터를 입력 받아 이를 저장하는 프로그램이다.

-이 프로그램은 사용자의 개인정보를 입력 받아, 이를 실시간으로 갱신하고, 이를 검색하면 그 결과를 화면에 보여준다.

-이 프로그램은 관리자가 지정한 기능을 수행하여, 회원 관리를 원활하게 할 수 있다.

○ 일반 제약 사항

- 이 프로그램은 MYSQL 이미지를 컨테이너로 실행해서 컨테이너 내부의 PYTHON 애플리케이션을 실행하기 위한 프로그램이다.

프로그램 설명

#1. 프로그램 요구사항

○ 고려사항

데이터 관련 사항

- PYTHON 컨테이너 내부에 있는 데이터베이스에 접근하여 회원정보를 입력하여 내부 테이블에 저장한다.

- PYTHON 컨테이너 내부 데이터에 접근해 데이터를 검색한다.

- PYTHON 컨테이너 내부 데이터에 접근해 데이터를 삭제한다.

- PYTHON 컨테이너 내부 데이터 상태를 파악해 변경 사항을 갱신한다.

시스템 관련 사항

- Docker의 MYSQL 이미지와 새로 생성한 python-mysql이미지를 compose작업을 통해 함께 호스트 PC에 띄워 작업한다.

- python-mysql 이미지를 Docker Hub에 올려 이용자들이 자유롭게 다운로드 받아 사용한다.

#3. Dockerfile & Docker-compose file

Ⅰ. Dockerfile

FROM **python**

COPY . .

RUN pip install pymysql && pip install cryptography

ADD https://raw.githubusercontent.com/vishnubob/wait-for-it/master/wait-for-it.sh /

RUN chmod +x /wait-for-it.sh

PYTHON 컨테이너 내부의 PYTHON 애플리케이션을 실행시키기 위한 프로그램.

1. python 이미지를 사용하여 컨테이너를 실행시킨다.
2. 현재 존재하는 파일들을 python 이미지를 실행할 컨테이너의 루트 디렉토리에 복사한다.
3. Pip는 python으로 작성된 패키지 소프트웨어를 설치하고 관리하는 패키지 관리 시스템이다. Pip를 이용해 pymysql을 설치한다. pymysql을 이용해 python 파일을 mysql에 연결하고, cryptography라는 프로그램의 대칭키를 이용하여 암호화 및 복호화를 수행한다.
4. mysql 컨테이너가 실행된 후에 python의 컨테이너에 의존하여 실행할 때, python 컨테이너가 mysql 내부의 데이터베이스에 있는 내용들을 담지 못하고 먼저 설치되는 에러가 발생하여 사용자 입장에선 한번 컨테이너를 내리고 다시 올려야 하는 불편함이 있다. 이를 피하기 위해 데이터베이스 파일들이 만들어지는 동안 지연을 시켜주기 위해 쉘 스크립트 명령어를 루트 디렉토리에 다운받는다.
5. 위에서 다운받은 지연시키는 파일에 실행 권한을 부여하고 실행하기 위함 => 실행은 docker-compose.yaml파일에 정의된 명령을 실행
6. 빌드: docker build -t sql\_python .
7. 컨테이너에서 python파일 실행(docker-compose) docker compose run python

Ⅱ. Docker-compose file

version: "3"

services:

  db:

    image: mysql

    restart: always

    environment:

      - MYSQL\_ROOT\_PASSWORD=t123456789

      - MYSQL\_PASSWORD=t123456789

      - MYSQL\_DATABASE=memlist

    ports:

      - "3306:3306"

    volumes:

      - C:\mysql\team-5\mysql:/var/lib/mysql/

  python:

       image: sql\_python

    restart: always

    command:

      [

        "/wait-for-it.sh",

        "tcp://db:3306",

        "-t",

        "20",

        "--",

        "python",

        "main.py",

      ]

    depends\_on:

      - db

MYSQL 컨테이너와 python-mysql 컨테이너를 함께 호스트 PC에서 작동시키기 위한 프로그램

1. 이 파일의 버전은 3이다.

2. 이 프로그램으로 실행할 프로그램 중 서비스에 대해 설명한다.

3. 데이터베이스는 MYSQL 이미지를 이용해 실행된다. 그리고 언제든지 재가동 할 수 있다. 이 데이터베이스에서 쓸 암호는 t123456789이고 데이터베이스의 이름은 memlist이다. 포트는 mysql 지정 포트인 3306:3306을 사용하며 C:\mysql\team-5\에 있는 파일을mysql을 이용하여 :/var/lib/mysql/로 전송한다.

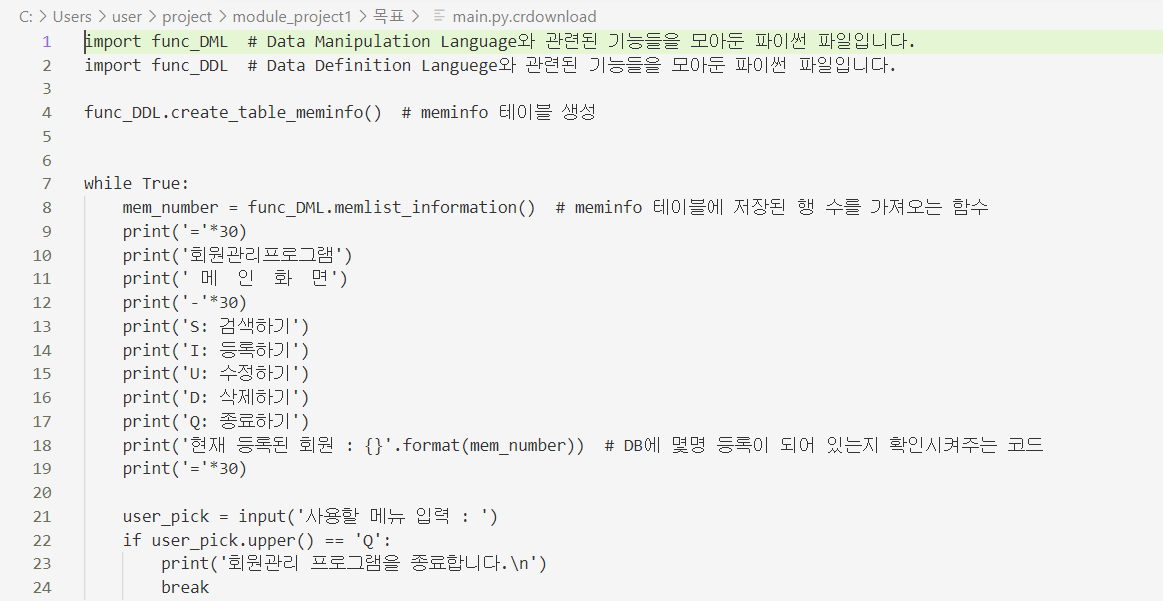
4. 이 프로그램으로 실행할 python에 대해 설명한다. 이때python은 Dockerfile을 통해 이미지화한 sql python이미지를 사용한다. 이 프로그램도 언제든지 재가동할 수 있다.

5. 위 Dockerfile에서 설명한 python실행을 지연시키기 위한 명령어다. wait-for-it.sh라는 명령어를 tcp://db:3306에 있는 main.py로 보내 python을 20초간 지연시킨다.

6. 이 프로그램은 DB에 의존해 실행된다.

#4. 프로그램 동작 원리

Ⅰ. Main



1. func\_DML이라는 함수 모듈을 불러서 이 프로그램에서 활용한다. 이때 func\_DML은 데이터베이스 조작어(DML)에 관련된 함수 모듈이다.

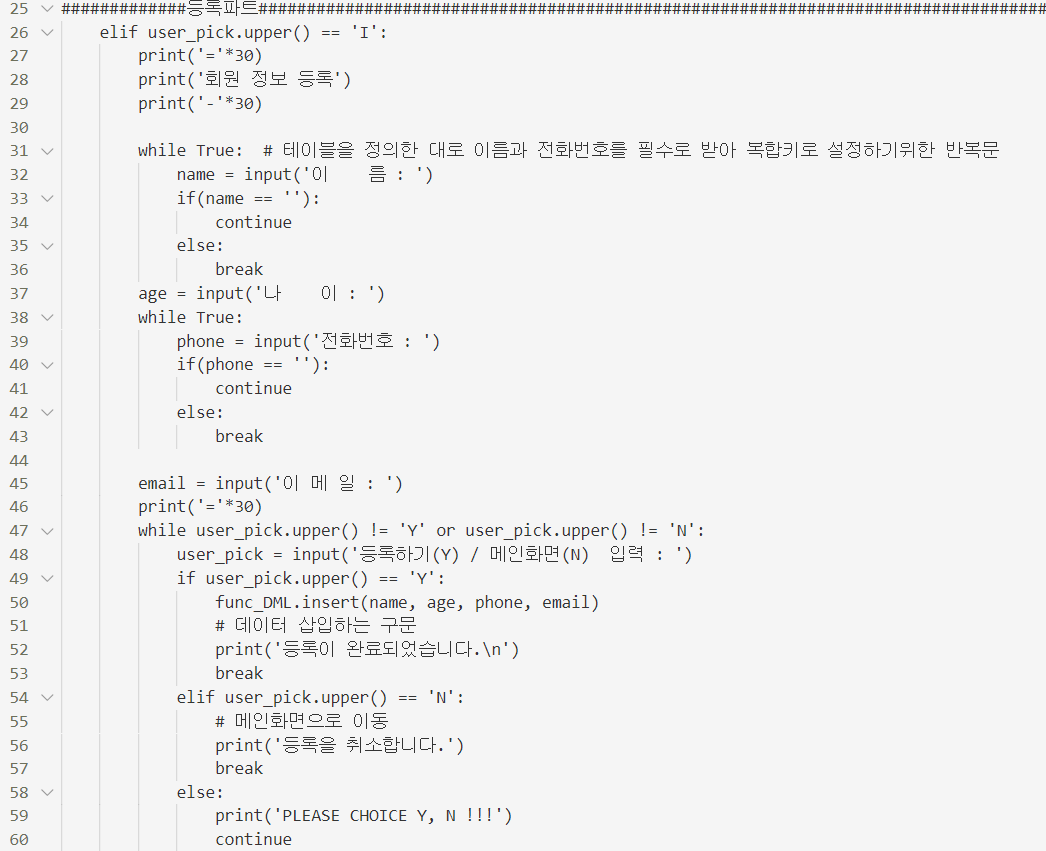
2. func\_DDL이라는 함수 모듈을 불러서 이 프로그램에서 활용한다. 이때 func\_DDL은 데이터베이스 정의어(DDL)에 관련된 함수 모듈이다.

3. func\_DDL 함수모듈에서 create\_table\_meminfo()라는 함수를 불러 meminfo라는 테이블을 생성한다.

4. 반복문이 시작된다. 반복문의 결과가 True가 나오면 func\_DML 함수 모듈에서 memlist\_infomation()라는 함수를 불러내 mem\_number에 대응시킨다. 앞으로 mem\_number가 나오면 memlist\_infomation() 함수를 실행한다.

5. memlist\_infomation() 함수를 실행시킨 결과이다. 메뉴 화면을 출력하고 DB에 몇 명의 회원이 등록되어 있는지 출력해준다.

6.user\_pick은 사용자가 사용할 기능을 입력 받는 창이다. 여기서 종료 키(Q)를 누르면 회원관리 프로그램이 종료된다.

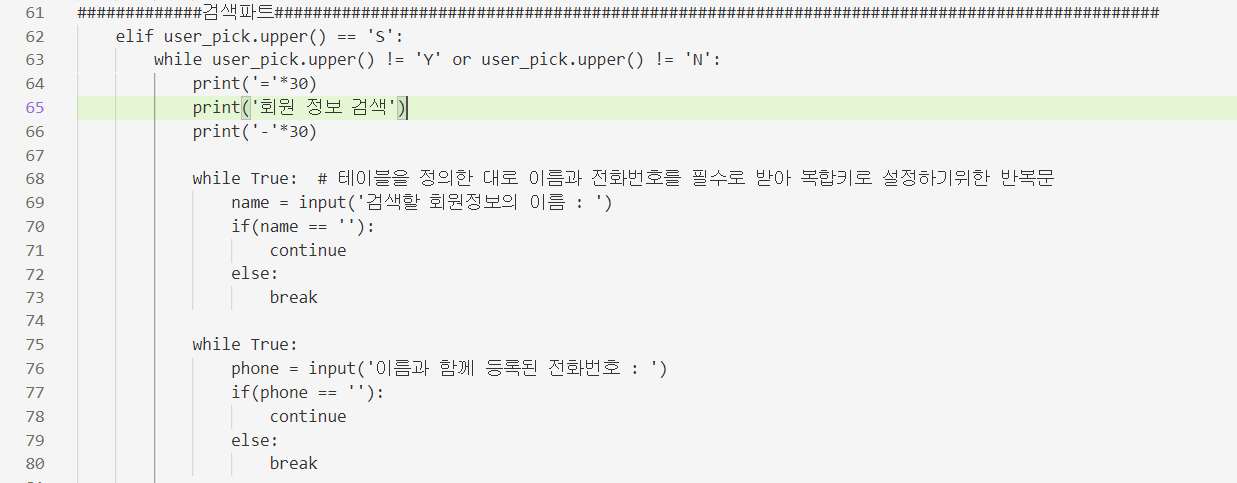


회원정보를 등록하기 위한 프로그램 코드이다.

1. 등록(I)키를 누르면 회원정보를 등록한다.

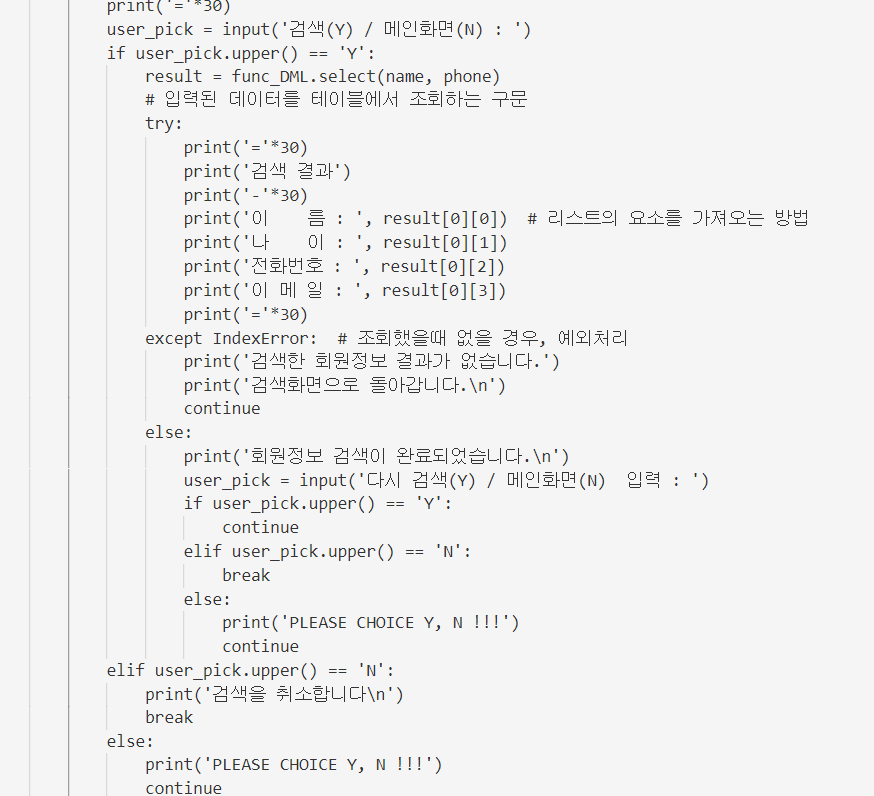
2. 반복문. 만약 반환 값이 True이면 이름, 나이, 전화번호, 이메일을 입력 받는다. 여기서 이름과 전화번호는 각 데이터를 구분하기 위해 필수로 입력해야 하는 복합 키이다. 이름과 전화번호를 입력하지 않았을 땐 continue를 이용해 등록 화면으로 다시 돌아간다. 다 입력하면 break로 루프를 빠져나간다.

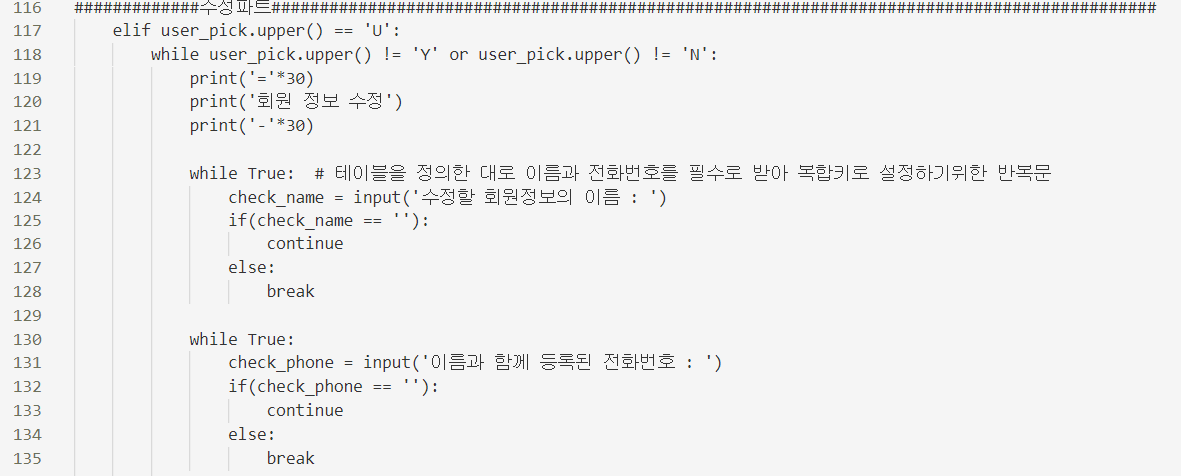
3. 예(Y)/아니오(N)을 입력 받는다. 여기서 대소문자를 통일시키기 위해 upper함수를 쓴다. Y를 누르면 테이블에 데이터를 입력해 저장하고 N을 누르면 등록을 취소하고 메인 화면으로 다시 이동한다. 만약 Y/N이 아닌 다른 키를 입력하면 Y와 N 중 하나의 키를 선택하라는 메시지가 뜬다.



↑↓회원정보를 검색하기 위한 프로그램 코드이다.

1. 검색(S)키를 누르면 회원정보를 검색한다.
2. 검색할 대상의 이름을 입력한다. 만약 이름이 공백이면 다시 이름을 입력하라고 메시지가 뜨며 이름을 입력하는 명령어로 돌아간다.
3. 이름과 함께 등록된 전화번호를 입력한다. 만약 전화번호가 공백이면 다시 전화번호를 입력하는 명령어로 돌아간다.
4. Y를 누르면 검색을 수행하고 N을 누르면 메인 화면으로 돌아간다. Y를 누르면 테이블에 저장된 데이터(리스트)를 가져와 그 리스트에서 해당되는 데이터를 순서대로 가져온다. Ex) 저장된 정보가 김한국, 23세, 011 – 666 – 4034, [qawsed@rftg.com](mailto:qawsed@rftg.com)일 때 이름 [0][0], 나이[0][1], 전화번호[0][2], 이메일[0][3] 이런 식으로 불러온다.
5. 조회했을 때 결과가 없으면 IndexError가 발생한다. 이럴 땐 회원정보가 없다는 결과를 보여주고 검색화면으로 돌아간다.
6. 조회가 완료되면 회원정보 검색이 완료되었다는 문장이 뜬다. 여기서 다시 검색을 할 것인지 메인화면으로 돌아갈 것인지 물어본다. Y를 누르면 다시 데이터를 입력하고 N을 입력하면 다시 메인 화면으로 돌아간다. 다른 키를 눌렀을 시 Y/N 키를 누르라는 메시지가 뜬다.
7. 검색창에서 N을 입력했을 경우 검색을 취소한다는 결과를 반환한다.
8. 검색 메인에서 Y/N이 아닌 다른 키를 입력하면 Y/N키를 입력해달라는 메시지를 띄운다.





회원정보를 찾아 수정하는 프로그램 코드이다.

1. U를 입력하면 회원 정보를 수정한다.

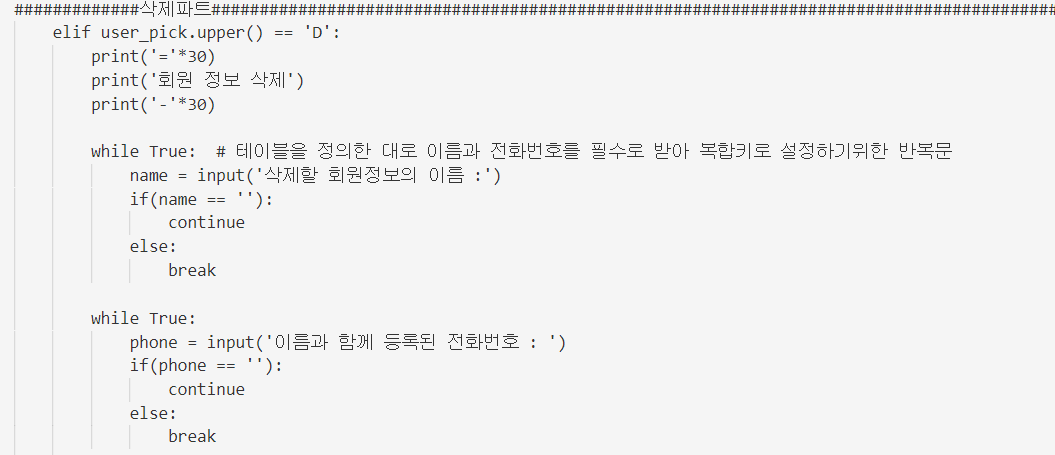
2. 반복문으로 들어간다. 수정할 회원정보의 이름과 전화번호를 입력한다. 이때 이름과 전화번호가 빈칸이면 입력으로 다시 돌아간다. 입력이 되었으면 루프를 종료하고 다음으로 넘어간다.

3. Y를 입력하면 회원정보를 수정하는 명령이 뜬다. 줄바꿈을 하지 않고 데이터를 수정하기 위해 줄바꿈을 넣지 않는다. 이때, 기존의 데이터 주소 값은 유지한다.

4. 데이터 수정이 완료되면 루프를 종료한다. 만약 각 데이터 양식에 맞지 않게 값을 입력하거나 빈칸으로 두면 올바른 양식이 아니라는 문장이 뜬다.

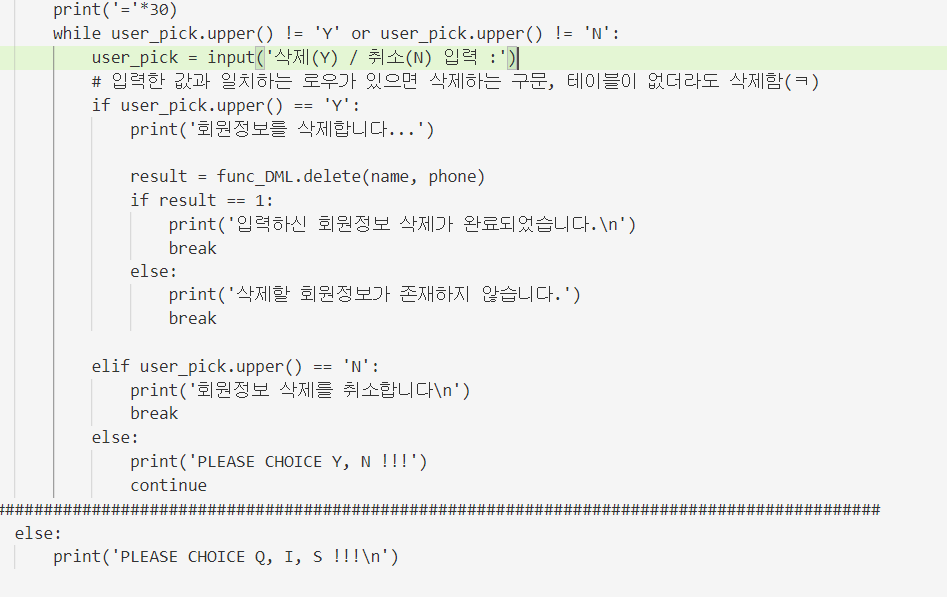
5. 수정화면에서 N을 입력 받았다면 회원정보를 취소하고 메인화면으로 돌아간다. Y/N이 아닌 다른 값을 입력했다면 Y/N을 입력하라는 메시지가 뜬다.



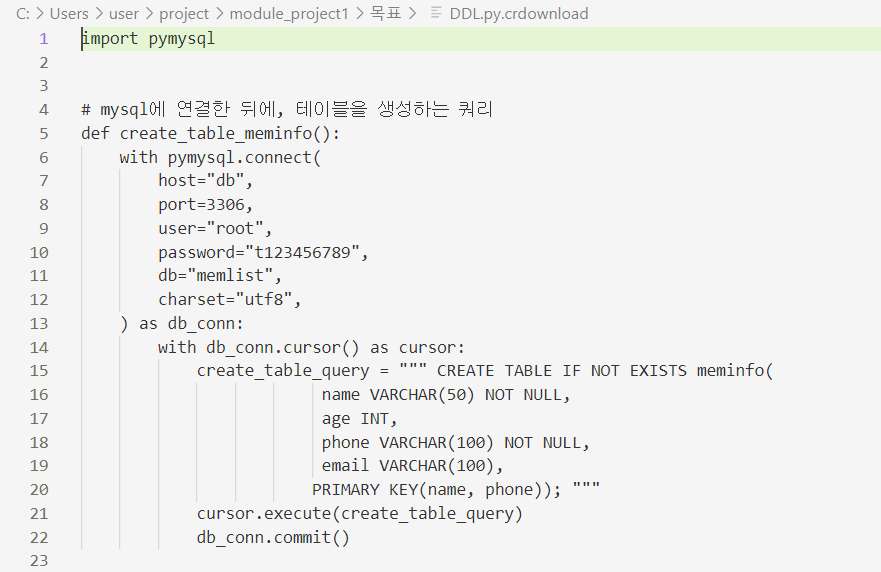


회원정보를 삭제하기 위한 프로그램 코드이다.

1. D를 입력할 시 회원정보 삭제 메뉴를 실행시킨다. 역시 이름과 전화번호를 입력 받아 테이블 내에서 삭제할 데이터를 찾는다. 이름과 전화번호가 빈칸이면 다시 입력으로 돌아간다.
2. Y를 누르면 해당 데이터를 삭제한다. 이때 이 데이터가 테이블에 없어도 삭제할 수 있다. 만약 삭제할 데이터가 테이블에 없다면 삭제할 회원정보가 없다는 결과가 나온다. 그렇지 않다면 해당 데이터를 삭제하고 루프를 빠져 나온다.
3. N을 누르면 삭제를 취소한다는 메시지가 뜬다. Y/N이 아닌 다른 값을 입력 받으면 Y/N을 입력하라는 메시지가 출력된다.



Ⅳ. DDL



데이터베이스의 기능을 정의하는 프로그램 코드이다.

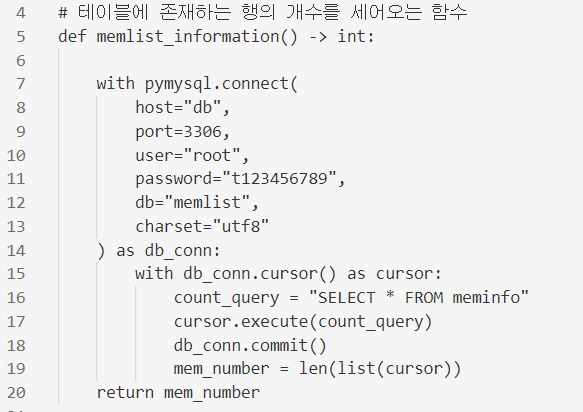
1. pysql 모듈을 가져온다.

2. 테이블을 생성하는 쿼리를 정의한다.

3. 만약 meminfo라는 테이블이 존재하지 않으면, meminfo라는 테이블을 만들고, 테이블에 들어갈 이름, 나이, 전화번호, 이메일의 입력 값을 정해준다. 이 중에서 필수적인 키는 이름과 전화번호이다.

4. 테이블 생성을 수행한다.

Ⅴ. DML

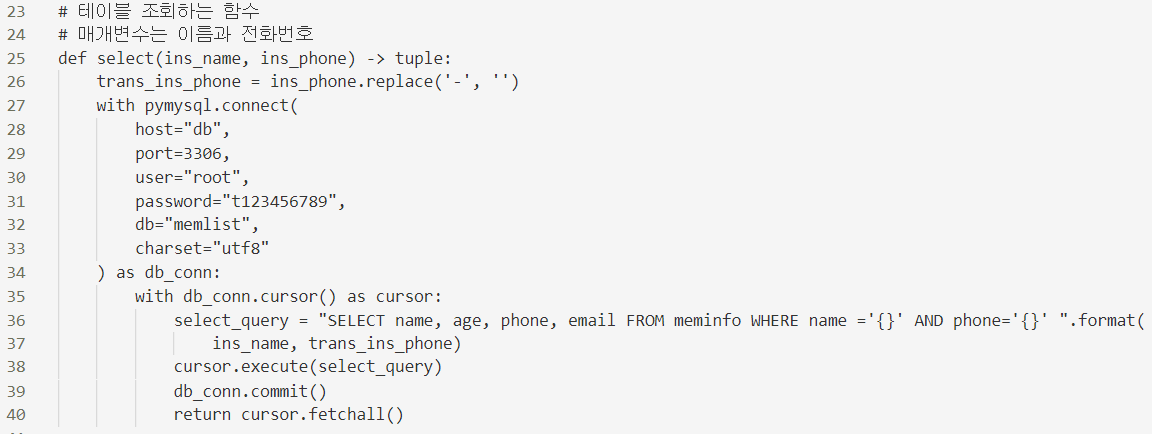


프로그램 실행에 필요한 함수들을 정의하는 프로그램 코드이다. 이 함수는 테이블에 존재하는 행의 개수를 세는 함수이다.

1. memlist\_information함수를 정의한다. 이때 함수의 반환 값은 int(정수형)이다.

2. meminfo에서 행의 개수를 세서 그 값을 count\_query에 대응시킨다. 이 쿼리는 행의 개수를 출력한다.

3. mem\_number는 데이터가 있는 리스트들의 길이를 재서 그 길이를 숫자로 반환 시켜 회원의 수를 구한다. 이 쿼리는 회원 수를 반환한다,

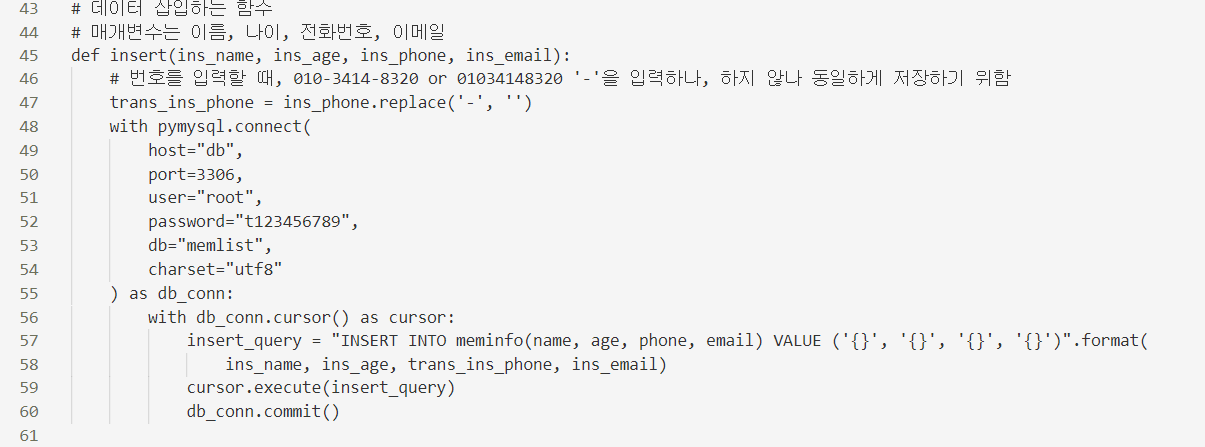


이름과 전화번호를 매개변수로 해서 테이블을 조회하는 함수다.

1. 이름과 전화번호를 선택하는 select함수를 정의한다. 반환 값은 튜플로 정의한다.

2. 전화번호에 있는 ‘-‘을 공백으로 바꾸기 위해 trans\_ins\_phone을 정의한다.

3. meminfo에서 입력한 이름과 전화번호가 포함된 데이터를 찾는 작업을 수행한다. 이 결과를 모두 보여주기 위해 fetchall을 사용한다.

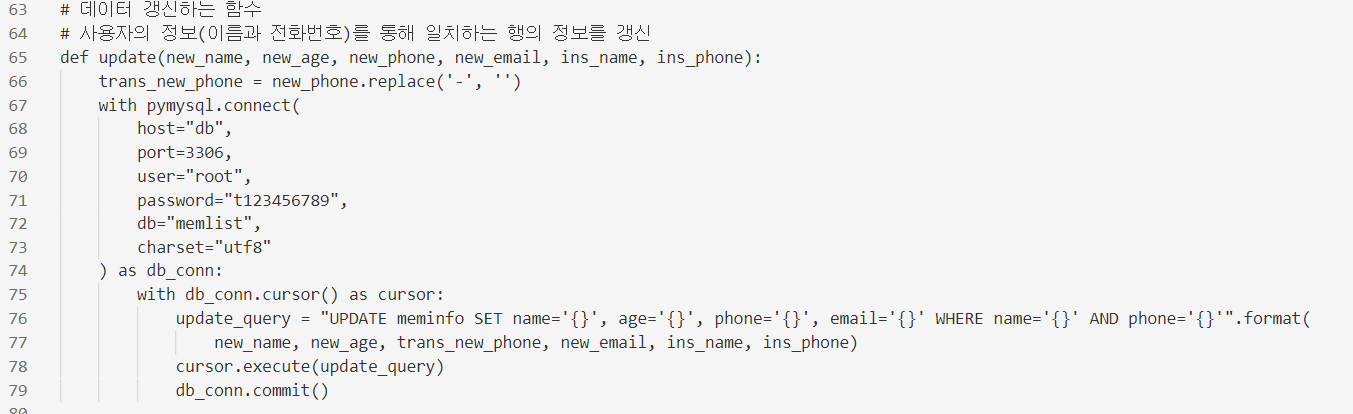


데이터를 삽입하는 함수이다.

1. 입력해야 할 값은 이름, 나이, 전화번호, 이메일이다.

2. 전화번호를 입력할 때 ‘-‘를 입력하든 입력하지 않든 같은 결과값을 출력하기 위해 trans\_ins\_phone을 사용한다.

3. meminfo로 들어가 이름, 나이, 전화번호, 이메일을 입력하는 insert\_query를 실행한다.

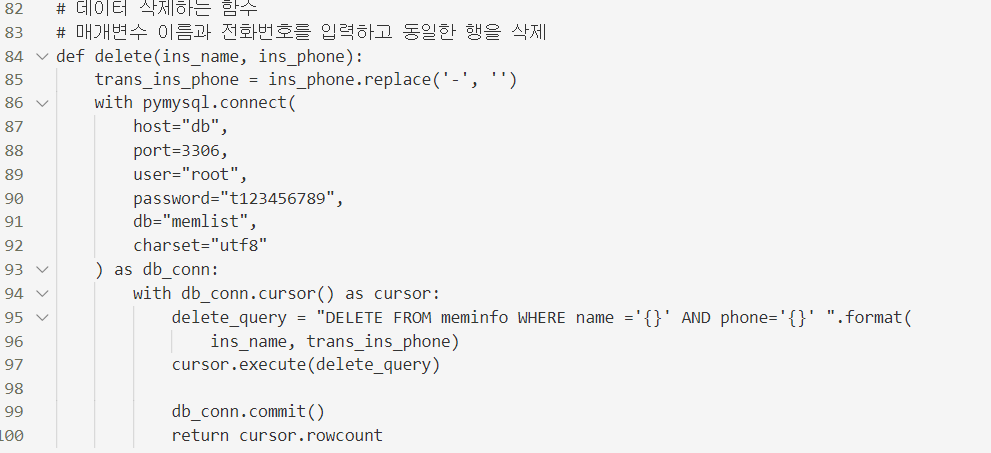


데이터를 갱신하는 함수이다.

1. 입력한 이름과 전화번호를 이용해 일치하는 행을 찾아 새로운 정보를 입력할 수 있게 하는 update함수를 정의한다.

2. 전화번호를 바꾸는 trans\_new\_phone이다. 이때도 ‘-‘을 공백으로 바꿔준다.

3. meminfo로 들어가 데이터를 갱신하는 update\_query를 실행한다.



1. 이름과 전화번호를 받아 해당되는 데이터를 삭제하는 delete를 정의한다.

2. meminfo로 들어가 이름과 전화번호를 찾아 해당되는 데이터 행을 삭제하는 쿼리를 수행한다.

3. 삭제한 후 삭제한 행의 개수를 세서 반환한다.

#5. 프로그램 기능

|  |  |  |
| --- | --- | --- |
| 기능 | 조작 | 설명 |
| 메뉴 출력 | 실행 | 메인 화면을 출력한다. |
| 등록 | I | 데이터베이스에 회원정보를 등록한다. |
| 검색 | S | 데이터베이스에 있는 회원정보를 검색한다. |
| 갱신 | U | 데이터베이스에 있는 회원정보를 수정한다. |
| 삭제 | D | 데이터베이스에 있는 회원정보를 삭제한다. |
| 종료 | Q | 프로그램을 종료한다. |