UNISTUDY

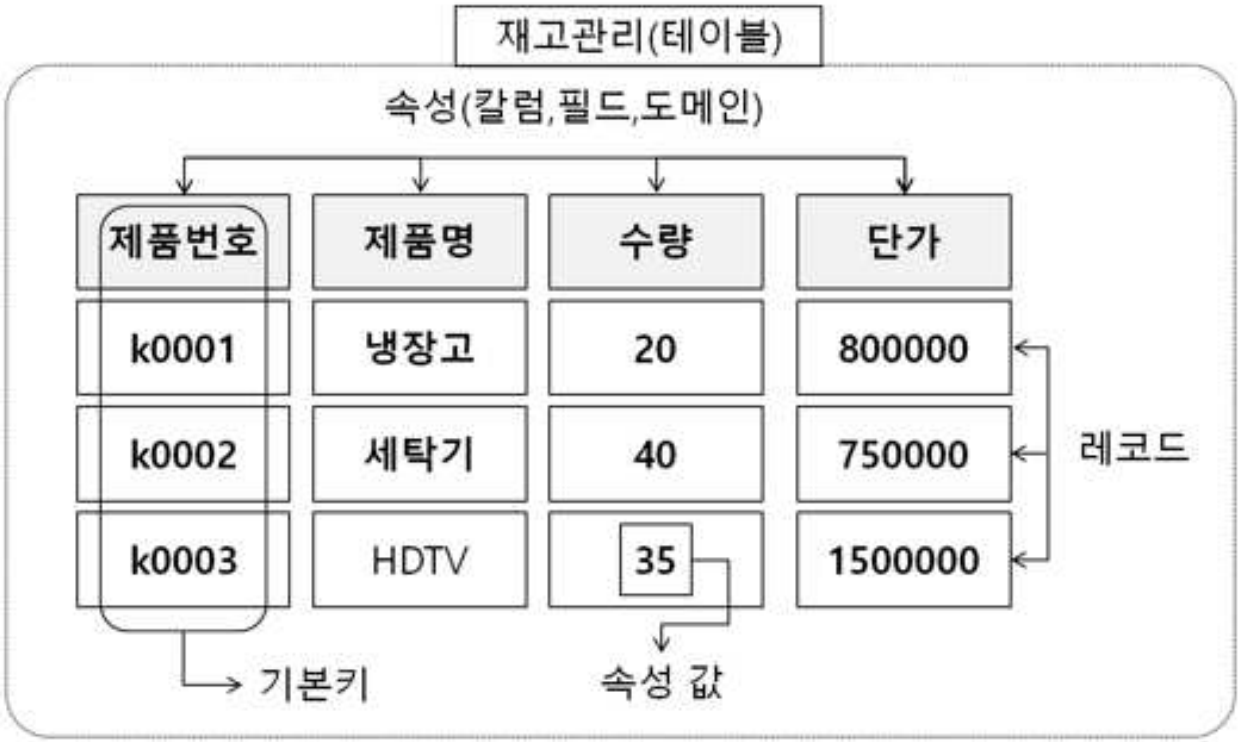
파이썬

37강. 관계형데이터베이스

1. 관계형데이터베이스

- 관계형데이터베이스는 자료와 자료 사이의 관계를 2차원의 테이블(table) 형태로 제공하는 데이터베 이스를 말한다. 테이블의 각 행(Row)은 하나의 객체로 표현되고, 각 열(Column)은 객체의 속성으로 표현된다. 다음은 재고관리 테이블의 예를 보여주고 있다.
- 관계형데이터베이스를 시스템에 구축하기 위해서는 데이터베이스관리 시스템(DBMS)라는 소프트웨어 를 이용해야 한다.

1. 관계형데이터베이스


재고관리(테이블)
속성(칼럼,필드,도메인)
제품번호
제품명
수량
단가
k0001
냉장고
20
800000
k0002
세탁기
40
750000
k0003
HDTV
35
1500000
레코드
기본키
속성 값

2. DBMS

- 방대한 양의 자료를 효율적으로 저장하고, 편리하게 관리 및 검색할 수 있는 환경을 제공하는 시스템 소 프트웨어를 말한다. 데이터베이스에 대한 사용자의 요구를 수행할 수 있도록 사용자 인터페이스와 데이 터 베이스언어(SQL)로 구성된다. 현재 대표적인 DBMS는 Oracle, MySQL(MariaDB), MS-SQL, Infomix, MongoDB, SQLite3 등이 있다

3. DBMS 기능

- 데이터베이스관리시스템에서 제공하는 기능은 다음과 같다.
- 정의 기능(Definition)
- 자료의 형태, 구조, 데이터베이스 저장에 관한 내용을 정의하는 기능
- 조작 기능(Manipulation)
- 사용자의 요구에 따라 검색, 수정, 삽입, 삭제 등을 지원하는 기능 사용자와 데이터베이스 사이의 인터페이스를 위한 수단을 제공
- 제어 기능(Control)
- 데이터베이스의 내용에 대해 정확성과 안전성을 유지하는 기능 무결성 유지, 자료 보안, 병행 수행 제어 등의 기능 제공

4. SQLite3

- SQLite3는 기기 내부에서만 사용할 수 있는 DBMS이다. 주로 모바일이나 소형 기기에서 관계형데이 터베이스를 생성하는 소프트웨어이다. 생성된 데이터베이스는 외부에서의 접근은 허용하지 않는다.
- 파이썬에서는 SQLite3를 기본으로 제공하기 때문에 특별한 제약 없이 import를 통해서 바로 사용할 수 있다.

5. 테이블 만들기

- 데이터베이스 안에 테이블을 만들기 위해서는 다음과 같이 Create table 명령어 형식 을 이용한다. [if not exists]는 생략할 수 있는 문장으로 해당 테이블이 존재하지 않는 경우에만 테이블을 생성한다는 의미이다, table-name은 만들고자할 테이블명이다. ()안에는 테이블을 구성하 는 칼럼명, 자료형, 제약조건 순서로 작성하고 각 칼럼은 콤마(,)로 구분한다.

5. 테이블 만들기

- 칼럼명의 자료형은 문자형 : text, 숫자형 : numeric, 정수형 : integer, 실수형 : real, 대용량 : blob 등을 사용할 수 있고, 칼럼의 제약조건은 기본키 : primary key, 중복불가 : unique, 생략 불 가 : not null, 제한 조건 : check, 기본값 : default 등을 적용할 수 있다.

[형식]

```
Create table  [ if not exists ] table-name (
   column-name  type-name  column-constraint,
)
```

[예]

```
Create table  if not exists test(
   name text(10) primary key,
   phone text(15) not null,
   addr text(50)
)
```

[type-name]

TEXT : 문자형
NUMERIC : 숫자형
INTEGER : 정수형
REAL : 실수형
BLOB : 대용량자료

[column-constraint]

PRIMARY KEY : 기본키
UNIQUE : 중복불가
NOT NULL : 생략 불가
CHECK : 제한 조건
DEFAULT : 기본값

6. CRUD 만들기

- CRUD는 대부분의 컴퓨터 소프트웨어가 가지는 기본적인 데이터 처리 기능인 Create(생성), Read (읽기), Update(갱신), Delete(삭제)를 묶어서 일컫는 말이다. DBMS의 기능 중에서 조작 기능 (Manipulation)에 해당된다. 사용자 인터페이스가 갖추어야 할 기능(정보의 참조/검색/갱신)을 가리 키는 용어로서도 사용된다.
- CRUD는 다음과 같이 표준 SQL 명령어로 나타낸다.

6. CRUD 만들기

CRUD	기능	SQL 명령어
CREATE	생성	INSERT
READ	읽기/조회	SELECT
UPDATE	수정/갱신	UPDATE
DELETE	삭제/제거	DELETE

6. CRUD 만들기

- 실제 SQLite3를 이용하여 CRUD를 구현하기 위해서 각각의 표준 SQL 명 령문의 형식을 제공하고 있다.

[Insert 형식]

Insert into table-name (column-name) values(값1, 값2,…);

[Select 형식]

Select column-name or * from table-name where 조건식;

[Update 형식]

Update table-name set column-name = 수정값 where 조건식;

[Delete 형식]

Delete from table-name where 조건식 ;

6. CRUD 만들기

- SQLite3의 테이블 형식과 CRUD 표준 SQL 명령문의 형식을 이용하여 테이블을 생 성하고, 레코드를 조작하는 과정은 다음 예문에서 알아보다.

6. CRUD 만들기

chapter10.lecture.step01_sqlite_basic.py

```
'''sqlite3 - 내장형 DBMS'''
# 패키지 import
import sqlite3
print(sqlite3.sqlite_version_info) # (3, 21, 0)


try :
    # (1) db 연동 객체
    conn = sqlite3.connect("chapter10/data/sqlite db") # db 생성 -> 연결 ob
ject
    # (2) sql 실행 객체
    cursor = conn.cursor()


    # (3) table 생성
    sql='create table if not exists test_table(name text(10),phone text
(15),addr text(50))'
    cursor.execute(sql) # sql문 실행

    # (4) 레코드 추가
    cursor.execute("insert into test table values('홍길동', '010-1111-1111
', '서울시')")
    cursor.execute("insert into test table values('이순신', '010-2222-2222
', '해남시')")
    cursor.execute("insert into test table values('강감찬', '010-1111-1111
', '평양시')")
    conn.commit() # db 반영
```

6. CRUD 만들기

```
    # (5) 레코드 조회..
    cursor.execute("select * from test_table")
    rows = cursor.fetchall() # 조회 레코드 가져오기..

    for row in rows :  # (6) 레코드 출력1
        print(row)

    print('이름 \t전화번호    \t주소') # (7) 레코드 출력2
    for row in rows :
        print(row[0],'\t',row[1],'\t',row[2])

except Exception as e :
    print('db 연동 실패 : ', e)
    conn.rollback() # 실행 취소..


finally:
    cursor.close() # cursor 객체 닫기..
    conn.close() # conn 객체 닫기..
```

6. CRUD 만들기

- 해설 SQLite3 데이터베이스 만들기 예
- # (1) db 연동 객체
- sqlite3 패키지에서 제공하는 connect() 함수를 이용하여 지정한 경로에 데이터베이스 파일이 생성되고, 해당 데이터베이스로 연결할 수 있는 객체(conn)가 생성된다.
- # (2) sql 실행 객체
- 데이터베이스 연동 객체에서 cursor() 메서드를 호출하여 SQL문을 실행할 수 있는 객체를 생성한다.
- # (3) table 생성
- 테이블 생성 형식을 바탕으로 'test_table'을 생성한다. 테이블은 이름, 전화번호, 주소를 저장할 수 있는 3개의 칼럼으로 만들어진다. 테이블을 생성하는 sql문은 cursor 객체에서 호출할 수 있는 execute()함수를 이용하여 실행해야 한다.

6. CRUD 만들기

- # (4) 레코드 추가
- CRUD의 Insert문의 형식을 바탕으로 테이블에 레코드를 추가한다. 예문에서는 3개의 레코드를 추가 하고 있다. 테이블에 레코드를 추가한 이후에 이를 데이터베이스에 반영하기 위해서 데이터베이스 연동 객체를 이용하여 commit()함수를 호출한다.
- # (5) 레코드 조회
- CRUD의 Select문의 형식을 바탕으로 테이블의 레코드를 조회한다. 조회된 전체 레코드를 가져오기 위해서 cursor 객체를 이용하여 fetchall() 함수를 호출한다.

6. CRUD 만들기

- # (6) 레코드 출력1
- 조회된 전체 레코드를 저장한 rows를 for문으로 반복하여 한 줄의 레코드 단위로 출력한다. 한 줄의 레코드는 튜플 단위로 반환된다. 출력한 결과는 다음과 같다.
- ('홍길동', '010-1111-1111', '서울시')
- ('이순신', '010-2222-2222', '해남시')
- ('강감찬', '010-1111-1111', '평양시')
- # (7) 레코드 출력2
- 조회된 전체 레코드를 저장한 rows를 for문으로 반복하여 한 줄의 레코드 단위로 가져와서 색인을 이용하여 칼럼 단위로 출력한다. 칼럼 단위로 출력한 결과는 다음과 같다

이름	전화번호	주소
홍길동	010-1111-1111	서울시
이순신	010-2222-2222	해남시
강감찬	010-1111-1111	평양시

6. CRUD 만들기

- SQLite3에서 제공하는 CRUD 표준 SQL문의 형식을 이용하여 테이블을 조작하는 과정 이다. 테이블 생성, 레코드 추가, 레코드 조회에 대한 예문을 먼저 살펴보자.

6. CRUD 만들기

chapter10.lecture.step02_sqlite_crud.py

```
import sqlite3

try :
    # db 연동 객체
    conn = sqlite3.connect("chapter10/data/sqlite db") # db 생성 -> 연결 ob
ject
    # sql 실행 객체
    cursor = conn.cursor()

    # (1) table 생성
    sql = """create table if not exists
    goods(code integer primary key,
    name text(30) unique not null,
    su integer default 0,
    dan real default 0.0
    )"""
    cursor.execute(sql) # sql 실행

    # (2) 레코드 추가
    cursor.execute("insert into goods values(1, '냉장고', 2, 8500000)")
    cursor.execute("insert into goods values(2, '세탁기', 3, 5500000)")
    cursor.execute("insert into goods(code, name) values(3, '전자레인지')")
    cursor.execute("insert into goods(code, name, dan) values(4, 'HDTV',
15000000)")
    conn.commit() # db 반영

    # (3) 레코드 조회
    sql = "select * from goods"
    cursor.execute(sql)
    rows = cursor.fetchall() # 레코드 가져오기

    for row in rows :
        print(row[0], row[1], row[2], row[3])
```

6. CRUD 만들기

```
    print('검색된 레코드 수 : ', len(rows))

    # (4) 상품명 조회.
    name = input("상품명 입력 : ")
    sql = f"select * from goods where name like '%{name}%'"
    cursor.execute(sql) # 조회.
    rows = cursor.fetchall()

    if rows : # null = false
        for row in rows :
            print(row)
    else :
        print('검색된 레코드 없음')

except Exception as e :
    print('db 연동 error :', e)
    conn.rollback()

finally:
    cursor.close()
    conn.close()
```

UNISTUDY

6. CRUD 만들기

- 해설 SQLite3 CRUD 예
- # (1) table 생성
- 테이블 생성 형식을 바탕으로 'goods' 테이블을 생성한다. 테이블은 상품코드, 상품명, 수량, 단가를 저장할 수 있는 4개의 칼럼으로 만들어진다. 상품코드(code)는 기본키로 지정되고, 상품명(name)은 중복불가와 생략 불가의 제약조건을 가지며, 수량과 단가는 default 제약조건에 의해서 생략하면 기본값 0이 삽입된다.
- # (2) 레코드 삽입
- 4개의 칼럼 모두 값을 입력하는 방법과 default 값을 갖는 칼럼에는 입력 값을 생략하는 방법으로 레코드 4개를 삽입한다.
- # (3) 레코드 조회
- CRUD의 Select문의 형식을 바탕으로 goods 테이블의 전체 레코드를 조회한다. 조회된 전체 레코드를 가져와서 for문으로 각 레코드를 색인을 이용하여 칼럼 단위로 출력한다.

6. CRUD 만들기

- (4) 상품명 조회
- 키보드로 상품명을 입력받아서 해당 상품의 레코드를 조회하는 조건 검색 과정이다. 예문의 sql문에서 like '%{name}%'" 명령문 형식은 키보드로 입력한 상품명을 포함하는 name 칼럼의 레코드를 검색하 기 위한 sql문 이다.
- 다음은 키보드로 입력받은 값을 레코드로 삽입하는 SQL문과 상품코드(code)를 기준으로 수량과 단가 를 수정하는 SQL문, 상품코드(code)를 기준으로 레코드를 삭제하는 SQL문을 추가한 예문이다.
- 실습은 레코드 추가, 레코드 수정, 레코드 삭제를 순서대로 한 번씩 실행한다. 예를 들면 키보드로 상품 코드, 상품명, 수량, 단가를 입력받아서 레코드를 추가하는 코드를 작성하고, 실행해서 레코드 조회를 통해서 확인한다. 이때 레코드 수정과 레코드 삭제 코드는 작성하지 않는다. 레코드 수정을 위해서는 레코드 추가 코드는 주석처리하고 레코드 삭제는 작성하지 않는다. 끝으로 레코드 삭제를 위해서는 레 코드 추가와 레코드 수정 코드는 주석처리한다.

6. CRUD 만들기

chapter10.lecture.step02_sqlite_crud.py

```
import sqlite3

try :
    # db 연동 객체
    conn = sqlite3.connect("chapter10/data/sqlite db") # db 생성 -> 연결 ob
ject
    # sql 실행 객체
    cursor = conn.cursor()

    # (1) table 생성
    sql = """create table if not exists
    goods(code integer primary key,
    name text(30) unique not null,
    su integer default 0,
    dan real default 0.0
    )"""
    cursor.execute(sql) # sql 실행

    # (2) 레코드 추가
    '''
    cursor.execute("insert into goods values(1, '냉장고', 2, 8500000)")
    cursor.execute("insert into goods values(2, '세탁기', 3, 5500000)")
    cursor.execute("insert into goods(code, name) values(3, '전자레인지')")
    cursor.execute("insert into goods(code, name, dan) values(4, 'HDTV',
15000000)")
    conn.commit() # db 반영
    '''
    # (5) 레코드 추가 : 2차
    code = int(input('code 입력 : '))
    name = input('name 입력 : ') # 문자
    su = int(input('su 입력 : '))
    dan = int(input('dan 입력 : '))
```

6. CRUD 만들기

```
        sql = f"insert into goods values({code},'{name}',{su},{dan})"
        cursor.execute(sql) # 레코드 추가.
        conn.commit()

        # (6) 레코드 수정 : code -> su, dan 수정.
        ''' 주석처리.
        code = int(input('수정 code 입력 : '))
        su = int(input('수정 su 입력 : '))
        dan = int(input('수정 dan 입력 : '))

        sql = f"update goods set su = {su}, dan = {dan} where code = {code}"
        cursor.execute(sql) # 수정.
        conn.commit() # db 반영.
        '''
        # (7) 레코드 삭제 : code -> 삭제.
        ''' 주석처리.
        code = int(input('삭제 code 입력 : '))
        sql = f"delete from goods where code = {code}"
        cursor.execute(sql) # 삭제.
        conn.commit() # db 반영.
        '''
        # (3) 레코드 조회.
        sql = "select * from goods"
        cursor.execute(sql)
        rows = cursor.fetchall() # 레코드 가져오기.

        for row in rows :
            print(row[0], row[1], row[2], row[3])

        print('검색된 레코드 수 : ', len(rows))

        # (4) 상품명 조회.
        name = input("상품명 입력 : ")
        sql = f"select * from goods where name like '%{name}%'"
        cursor.execute(sql) # 조회.
        rows = cursor.fetchall()

        if rows : # null = false
            for row in rows :
                print(row)
        else :
            print('검색된 레코드 없음')

    except Exception as e :
        print('db 연동 error :', e)
        conn.rollback()

    finally:


       cursor.close()
       conn.close()
```

6. CRUD 만들기

- 해설 SQLite3 CRUD 예

- # (5) 레코드 추가 : 2차
- 4개의 칼럼에 해당하는 값을 키보드로 입력 받아서 values()함수 안에서 {변수} 형식으로 insert문 을 작성한다. 자료형이 문자형인 경우에는 '{변수}' 형식으로 지정한다.
- # (6) 레코드 수정 : code -> su, dan 수정
- 키보드로 수정할 상품코드, 수량, 단가를 입력 받아서 update문으로 상품코드의 레코드를 찾아서 해당 레코드의 수량과 단가를 수정한다.
- # (7) 레코드 삭제 : code -> 삭제
- 키보드로 상품코드를 입력받아서 delete문으로 해당 상품코드의 레코드를 삭제한다.

UNISTUDY

THANK YOU