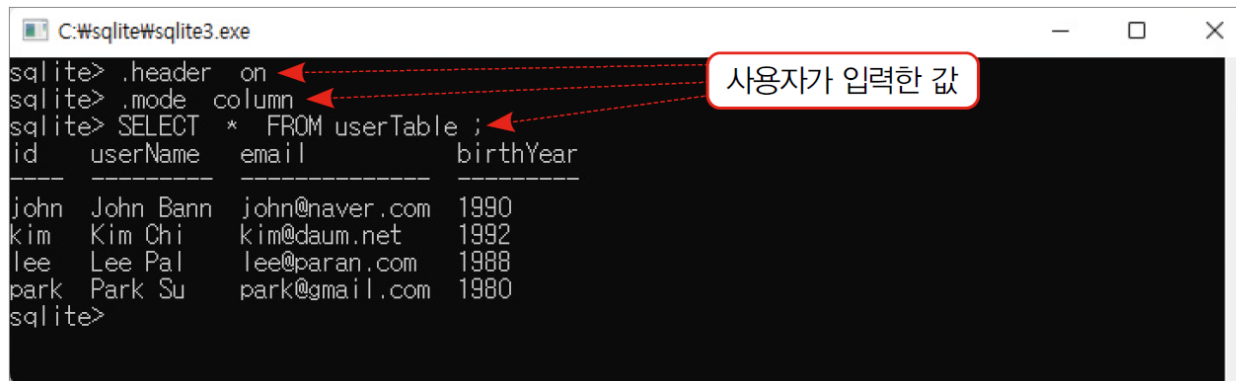


## Section 01 이 장에서 만들 프로그램

### ■ [프로그램1] 데이터베이스 기본

- 파이썬으로 작성하지 않고 SQLite 에서 데이터베이스를 조회하는 프로그램



C:\sqlite\sqlite3.exe

```
sqlite> .header on
sqlite> .mode column
sqlite> SELECT * FROM userTable ;
```

id	userName	email	birthYear
john	John Bann	john@naver.com	1990
kim	Kim Chi	kim@daum.net	1992
lee	Lee Pal	lee@paran.com	1988
park	Park Su	park@gmail.com	1980

sqlite>

사용자가 입력한 값

## Section 01 이 장에서 만들 프로그램

### ■ [프로그램2] 파이썬에서 SQLite

- 파이썬에서 데이터베이스를 입력하고 조회하는 프로그램

사용자ID	사용자이름	이메일	출생연도
john	John Bann	john@naver.com	1990
kim	Kim Chi	kim@daum.net	1992
lee	Lee Pal	lee@paran.com	1988
park	Park Su	park@gmail.com	1980
su	Su Ji	suji@naver.com	1994
woo	Woo Ja	woo@hanbit.co.kr	2002

### ■ 데이터베이스의 개념

- 데이터베이스 : 대량의 데이터를 체계적으로 저장해 대량의 데이터를 처리 할 수 있는 방법
- 파일 처리 : 데이터의 양이 적을 때
- 데이터베이스 소프트웨어 : DBMS(DataBase Management System 또는 Software )
  - 종류 : 오라클( Oracle ), SQL 서버( SQL Server ), MySQL , 액세스 ( Access ), SQLite 등

### ■ 관계형 데이터베이스

- DBMS의 구분 : 계층형( Hierarchical ) , 망형( Network ) , 관계형( Relational ) , 객체지향형 ( Object - Oriented ) , 객체관계형( Object - Relational ) 등
- 관계형 DBMS의 종류 : 오라클, SQL 서버, 액세스, MySQL
- 단점 : 속도가 전반적으로 느림

### ■ 데이터베이스 관련 용어

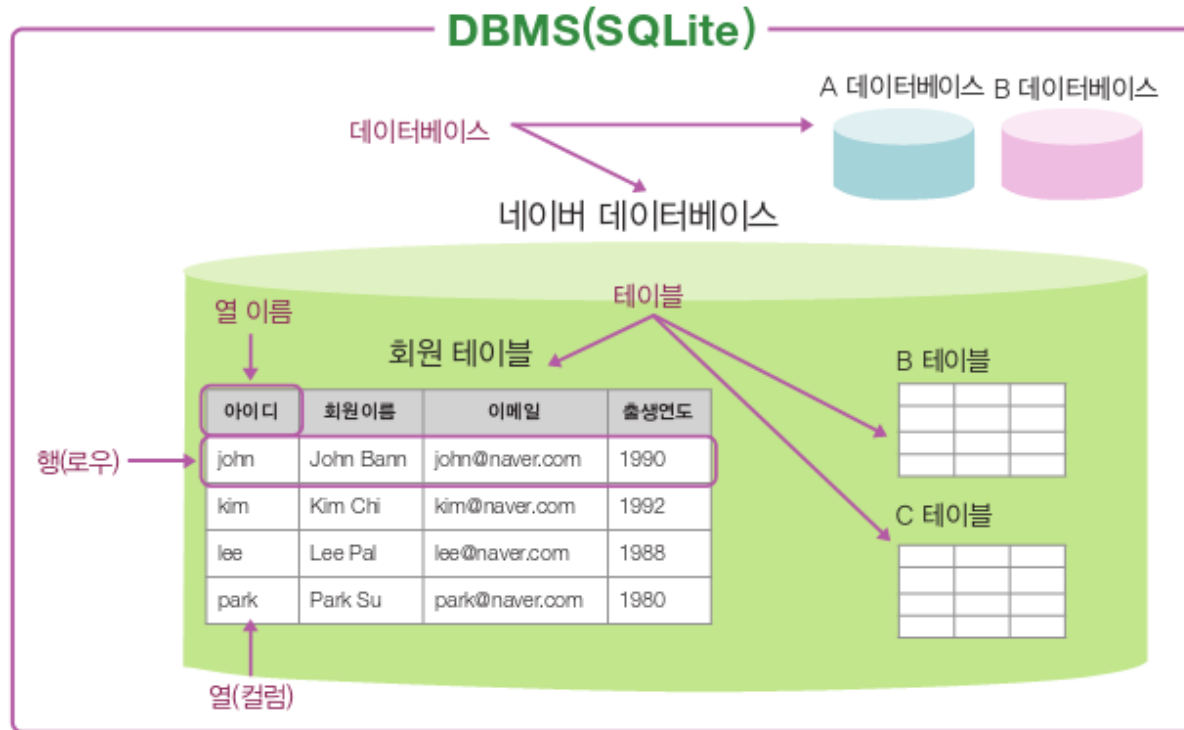


그림 13-1 DBMS 구성도

## Section 02 데이터베이스의 기본

- **데이터** : john , lee @ naver.com , 1980 등 하나하나의 단편적인 정보 의미
- **테이블** : 회원 데이터가 표 형태로 표현된 것
- **데이터베이스(DB)** : 테이블이 저장되는 저장소, 주로 원통 모양으로 표현
- **DBMS(DataBase Management System)** : 데이터베이스 관리 시스템 또는 소프트웨어
- **열(컬럼 또는 필드)** : 각 테이블은 1 개 이상의 열로 구성
- **열 이름** : 각 열을 구분하는 이름. 열 이름은 각 테이블 안에서 중복되지 않아야 함
- **데이터 형식** : 열의 데이터 형식으로 테이블을 생성할 때 열 이름과 함께 지정해야 함
- **행(로우)** : 실질적인 데이터
- **SQL(Structured Query Language : 구조화된 질의 언어)** : 사용자와 DBMS가 소통하는 말

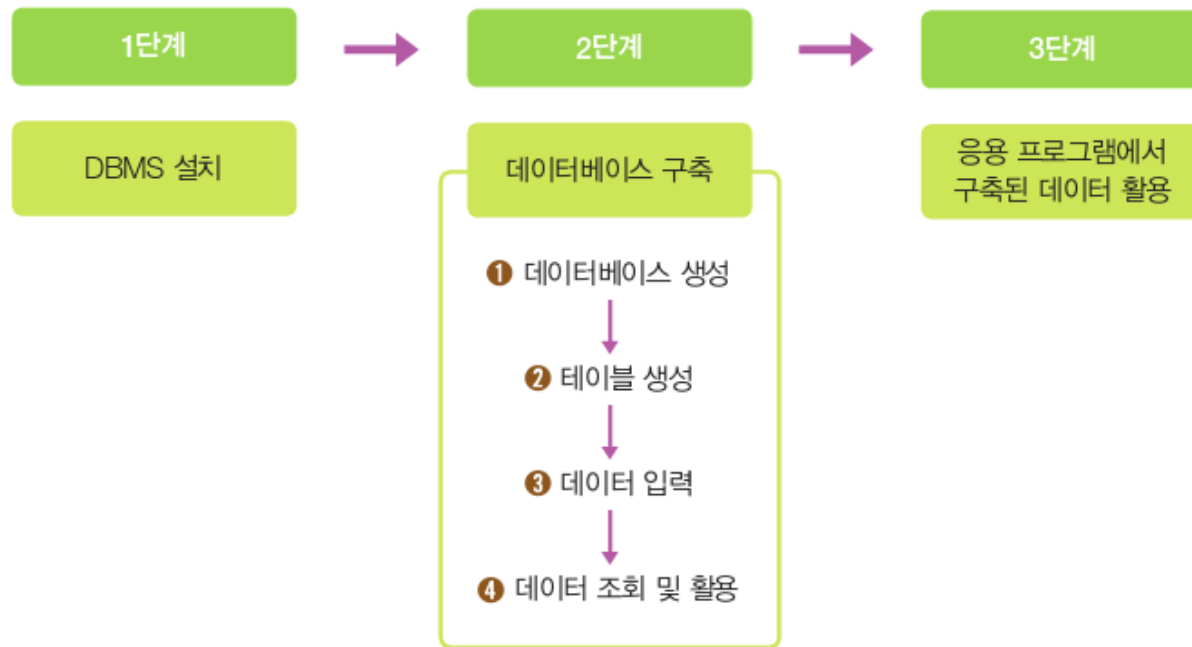
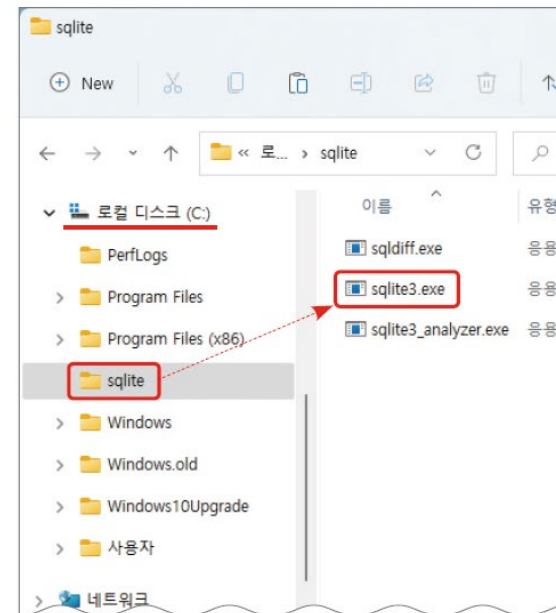
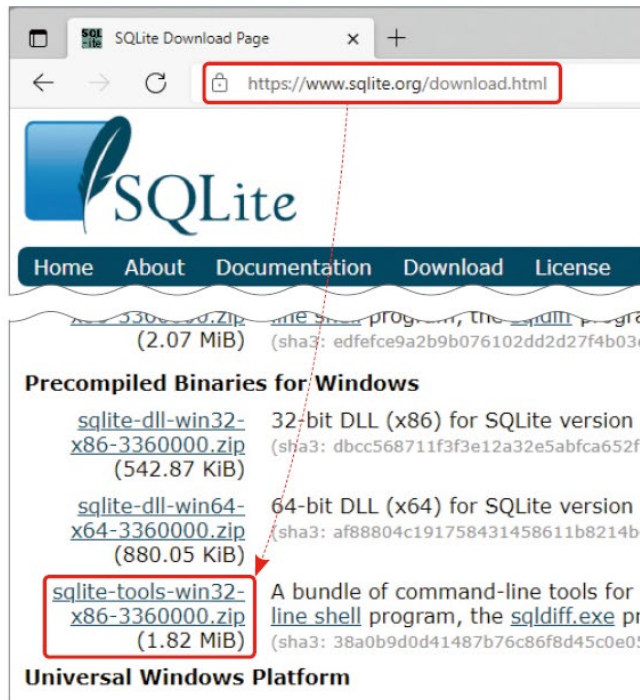


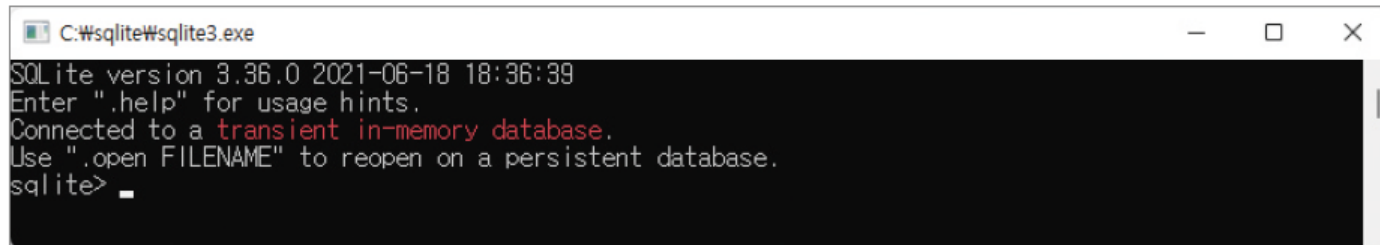
그림 13-2 데이터베이스 구축 및 운영 과정

### ■ 1단계 : DBMS 설치

#### ■ SQLite 설치



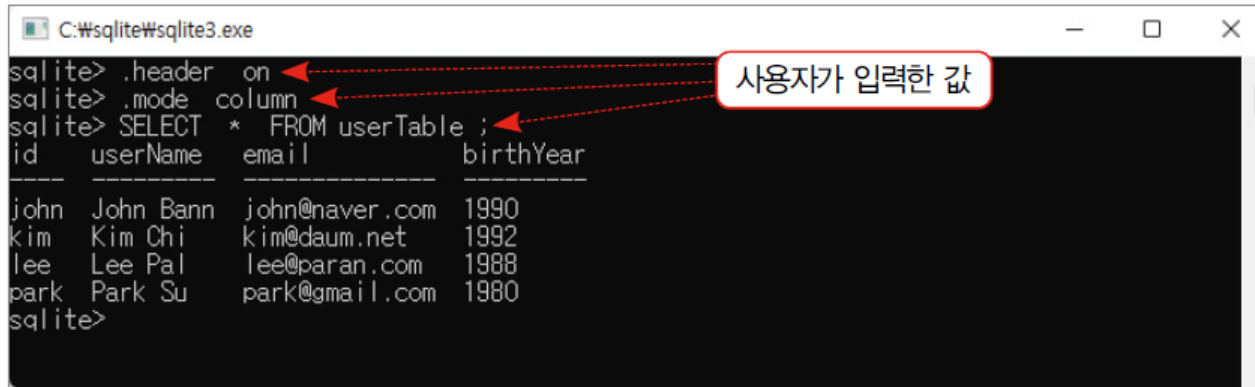
- 2단계 : 데이터베이스 구축
  - SQLite 에 접속



```
C:\sqlite>sqlite3.exe
SQLite version 3.36.0 2021-06-18 18:36:39
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> _
```



### ■ [프로그램 1]의 완성



The screenshot shows a SQLite3 command prompt window titled "C:\sqlite\sqlite3.exe". The user has entered the following commands:

```
sqlite> .header on
sqlite> .mode column
sqlite> SELECT * FROM userTable ;
```

The output of the query is displayed in a columnar format:

id	userName	email	birthYear
john	John Bann	john@naver.com	1990
kim	Kim Chi	kim@daum.net	1992
lee	Lee Pal	lee@paran.com	1988
park	Park Su	park@gmail.com	1980

Red dashed arrows point from a red-bordered box containing the text "사용자가 입력한 값" (Value entered by the user) to the three command lines: ".header on", ".mode column", and "SELECT \* FROM userTable ;".

## Section 03 데이터베이스의 구축

### ■ ❶ 데이터베이스 생성

- '.open 데이터베이스이름' 명령어 실행



```
C:\sqlite\sqlite3.exe
sqlite> .open naverDB
sqlite>
```

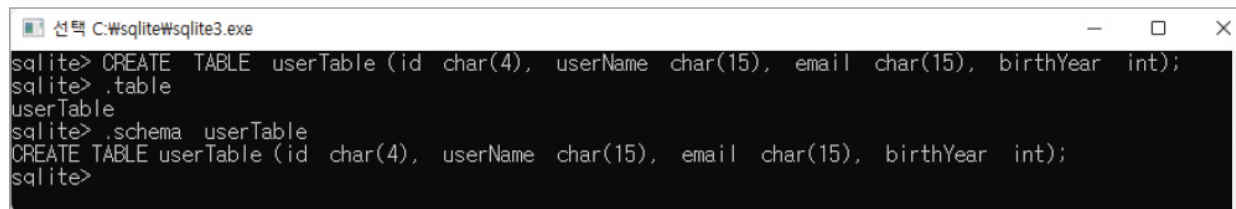
#### Tip 자주 사용하는 SQLite 명령어

- table : 현재 데이터베이스의 테이블 목록을 보여 준다.
- schema 테이블이름 : 테이블의 열 및 데이터 형식 등 정보를 보여 준다.
- header on : SELECT 문으로 출력할 때 헤더를 보여 준다.
- mode column : SELECT 문으로 출력할 때 컬럼 모드로 출력한다.
- quit : SQLite 를 종료한다.
- SELECT 문 사용 전 '. header on ', '. mode column ' 설정하면 결과 화면 보기 좋게 출력

## Section 03 데이터베이스의 구축

- ② 테이블 생성
  - naverDB 안에 테이블을 생성

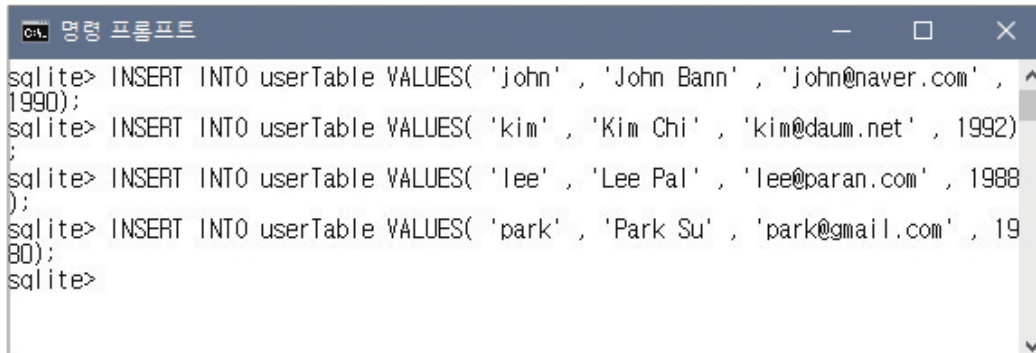
```
CREATE TABLE 테이블이름(열이름1 데이터형식, 열이름2 데이터형식, ...);
```



```
선택 C:\sqlite\sqlite3.exe
sqlite> CREATE TABLE userTable (id char(4), userName char(15), email char(15), birthYear int);
sqlite> .table
userTable
sqlite> .schema userTable
CREATE TABLE userTable (id char(4), userName char(15), email char(15), birthYear int);
sqlite>
```

- ③ 데이터 입력
  - 행 데이터 입력

```
INSERT INTO 테이블이름 VALUES(값1, 값2, ...);
```



```
cmd 명령 프롬프트
sqlite> INSERT INTO userTable VALUES( 'john' , 'John Bann' , 'john@naver.com' ,
1990);
sqlite> INSERT INTO userTable VALUES( 'kim' , 'Kim Chi' , 'kim@daum.net' , 1992)
;
sqlite> INSERT INTO userTable VALUES( 'lee' , 'Lee Pal' , 'lee@paran.com' , 1988
);
sqlite> INSERT INTO userTable VALUES( 'park' , 'Park Su' , 'park@gmail.com' , 19
80);
sqlite>
```

## Section 03 데이터베이스의 구축

### ■ 4 데이터 조회 및 활용

- SELECT 로 일반적인 형식

```
SELECT * FROM 테이블이름;
```

```
C:\sqlite#\sqlite3.exe
sqlite> .header on
sqlite> .mode column
sqlite> SELECT * FROM userTable ;
id      userName  email      birthYear
-----
john    John Bann  john@naver.com  1990
kim     Kim Chi   kim@daum.net   1992
lee     Lee Pal   lee@paran.com   1988
park    Park Su   park@gmail.com  1980
sqlite>
```

- SELECT 문을 WHERE 조건과 함께 사용

```
SELECT 열이름1, 열이름2, ... FROM 테이블이름 WHERE 조건;
```

```
sqlite> SELECT id, birthYear FROM userTable WHERE birthYear<=1990; ❶  
john|1990  
lee|1988  
park|1980  
sqlite> SELECT * FROM userTable WHERE id = 'park'; ❷  
park|Park Su|park@gmail.com|1980  
sqlite> SELECT * FROM userTable ORDER BY birthYear; ❸  
park|Park Su|park@gmail.com|1980  
lee|Lee Pal|lee@paran.com|1988  
john|John Bann|john@naver.com|1990  
kim|Kim Chi|kim@daum.net|1992  
sqlite>.quit ❹
```

### SELF STUDY 13-1

SQLite에 다시 접속해서 naverDB에 다음 테이블(productTable)을 구축해 보자.

제품코드(pCode)	제품명(pName)	가격(price)	재고수량(amount)
p0001	노트북	110	5
p0002	마우스	3	22
p0003	키보드	2	11

**힌트** 제품코드와 제품명은 char형으로 지정하고, 가격과 재고수량은 int형으로 지정한다.

#### 출력 결과

pCode	pName	price	amount
p0001	노트북	110	5
p0002	마우스	3	22
p0003	키보드	2	11

### ■ 데이터의 입력과 조회

#### ■ 파이썬에서 데이터 입력하는 코딩 순서

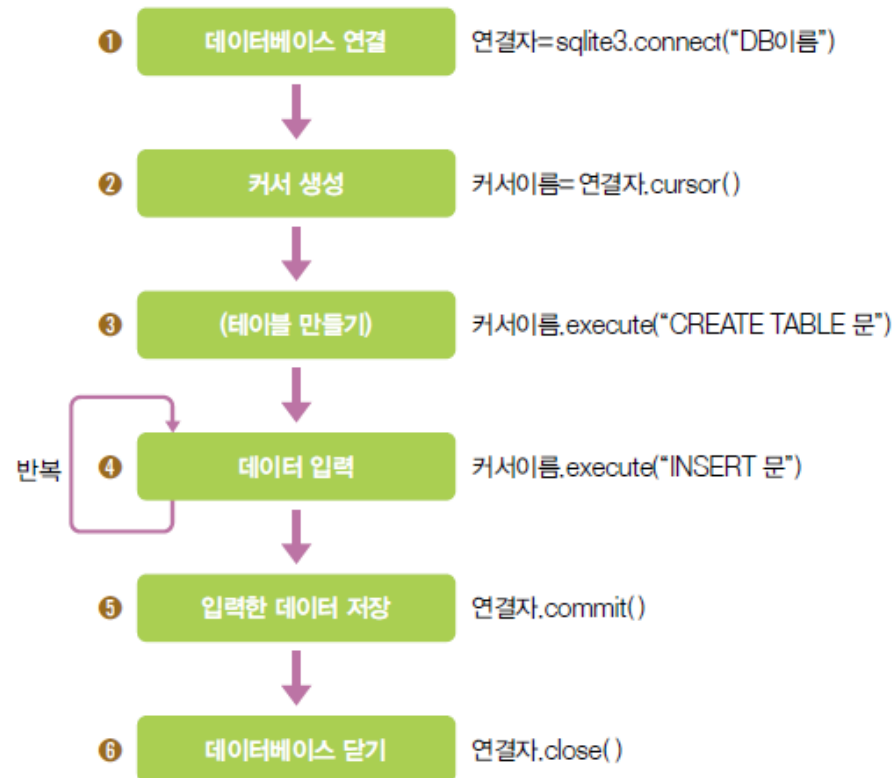


그림 13-7 SQLite의 데이터 입력 순서



## Section 04 데이터의 입력과 조회

### ▪ ❶ 데이터베이스 연결

- sqlite3 임포트 후 sqlite3 . connect (" DB 이름")으로 데이터베이스와 연결

```
import sqlite3
con = sqlite3.connect("C:/CookPython/naverDB")    # 소스 코드가 저장된 폴더에 생성
```

출력 결과

아무것도 나오지 않음

### ▪ ❷ 커서 생성

- 커서( Cursor ) : 데이터베이스에 SQL 문을 실행 또는 실행된 결과를 돌려받는 통로
- ❶ 에서 연결한 연결자에 커서 만듦

```
cur = con.cursor()
```

출력 결과

아무것도 나오지 않음

## Section 04 데이터의 입력과 조회

### ■ ③ 테이블 만들기

- 테이블 만드는 SQL 문을 커서이름. execute ( ) 함수의 매개변수로 넘겨주면 SQL 문이 데이터베이스에 실행

```
cur.execute("CREATE TABLE userTable (id char(4), userName char(15), email char(15), birthYear int)")
```

출력 결과

<sqlite3.Cursor object at 개체번호>

### ■ ④ 데이터 입력

```
cur.execute("INSERT INTO userTable VALUES('john', 'John Bann', 'john@naver.com', 1990)")
cur.execute("INSERT INTO userTable VALUES('kim', 'Kim Chi', 'kim@daum.net', 1992)")
cur.execute("INSERT INTO userTable VALUES('lee', 'Lee Pal', 'lee@paran.com', 1988)")
cur.execute("INSERT INTO userTable VALUES('park', 'Park Su', 'park@gmail.com', 1980)")
```

출력 결과

<sqlite3.Cursor object at 개체번호>가 각각 4회 출력됨

- ⑤ 입력한 데이터 저장-커밋( Commit )

```
con.commit()
```

출력 결과

아무것도 나오지 않음

- ⑥ 데이터베이스 닫기

```
con.close()
```

출력 결과

아무것도 나오지 않음

### ■ 데이터 입력 프로그램의 구현

Code13-01.py

```
1  import sqlite3
2
3  ## 변수 선언 부분 ##
4  con, cur = None, None
5  data1, data2, data3, data4 = "", "", "", ""
6  sql = ""
7
8  ## 메인 코드 부분 ##
9  con = sqlite3.connect("C:/CookPython/naverDB")      # DB가 저장된 폴더까지 지정
10 cur = con.cursor()
11
12 while (True) :
13     data1 = input("사용자ID ==> ")
14     if data1 == "" :
```

## Section 04 데이터의 입력과 조회

```
15         break;
16         data2 = input("사용자이름 ==> ")
17         data3 = input("이메일 ==> ")
18         data4 = input("출생연도 ==> ")
19         sql = "INSERT INTO userTable VALUES('" + data1 + "', '" + data2 + "', '" + data3 + "',
            " + data4 + "')"
20         cur.execute(sql)
21
22     con.commit()
23     con.close()
```

### 출력 결과

사용자ID ==> su  
사용자이름 ==> Su Ji  
이메일 ==> suji@naver.com  
출생연도 ==> 1994  
... 반복해서 입력 ...  
사용자 ID ==>  ← 종료됨

### SELF STUDY 13-2

Code13-01.py를 수정해서 [SELF STUDY 13-1]에서 생성한 productTable이 입력되도록 하자.

**힌트** 테이블을 생성해야 하므로 11행에서 cur.execute("CREATE TABLE 문")도 수행해야 한다.

### ■ 파이썬에서 데이터를 조회하는 코딩 순서

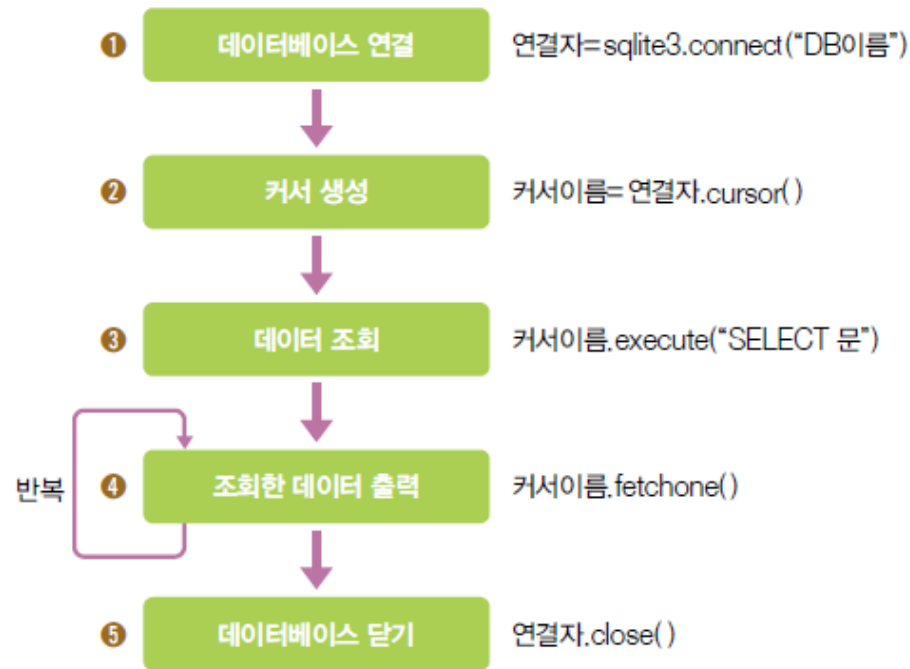


그림 13-8 SQLite의 데이터 조회 순서

### ■ 데이터 조회 프로그램의 구현

Code13-02.py

```
1  import sqlite3
2
3  ## 변수 선언 부분 ##
4  con, cur = None, None
5  data1, data2, data3, data4 = "", "", "", ""
6  row = None
7
8  ## 메인 코드 부분 ##
9  con = sqlite3.connect("C:/CookPython/naverDB")
10 cur = con.cursor()
11
12 cur.execute("SELECT * FROM userTable")
13
```

## Section 04 데이터의 입력과 조회

```
14 print("사용자ID      사용자이름      이메일      출생연도")
15 print("-----")
16
17 while (True) :
18     row = cur.fetchone()
19     if row == None :
20         break;
21     data1 = row[0]
22     data2 = row[1]
23     data3 = row[2]
24     data4 = row[3]
25     print("%5s  %15s  %15s  %d" % (data1, data2, data3, data4))
26
27 con.close()
```

### 출력 결과

사용자ID	사용자이름	이메일	출생연도
john	John Bann	john@naver.com	1990
kim	Kim Chi	kim@daum.net	1992
lee	Lee Pal	lee@paran.com	1988
park	Park Su	park@gmail.com	1980
su	Su Ji	suji@naver.com	1994



## Section 04 데이터의 입력과 조회

### ▪ rows 저장 형태

```
[("john", "John Bann", "john@naver.com", 1990), ('kim', 'Kim Chi', 'kim@daum.net', 1992), ...]
```

#### SELF STUDY 13-3

Code13-02.py를 수정해서 [SELF STUDY 13-2]에서 입력한 productTable의 내용이 출력되도록 해 보자.

##### 출력 결과

제품코드	제품명	가격	재고수량
-----			
p0001	노트북	110	5
p0002	마우스	3	22
p0003	키보드	2	11

## Section 04 데이터의 입력과 조회

### ■ [프로그램 2]의 완성

- 데이터 입력, 수정

The screenshot shows a GUI application window titled "GUI 데이터 입력". At the top, there is a form with four input fields: "woo", "Woo Ja", "@hanbit.co.kr", and "2002". These fields are grouped by a red dashed line. To the right of the fields are two buttons: "입력" (Add) and "조회" (Search). A red dashed arrow points from the "입력" button to the "조회" button. Below the form is a table with four columns: "사용자ID", "사용자이름", "이메일", and "출생연도". The table contains six rows of data, with the last row highlighted in red. A red solid line is drawn under the last row of the table.

사용자ID	사용자이름	이메일	출생연도
john	John Bann	john@naver.com	1990
kim	Kim Chi	kim@daum.net	1992
lee	Lee Pal	lee@paran.com	1988
park	Park Su	park@gmail.com	1980
su	Su Ji	suji@naver.com	1994
woo	Woo Ja	woo@hanbit.co.kr	2002

Code13-03.py

```
1 import sqlite3
2 from tkinter import *
3 from tkinter import messagebox
4
5 ## 함수 선언 부분 ##
6 def insertData() :
7     con, cur = None, None
8     data1, data2, data3, data4 = "", "", "", ""
9     sql = ""
10
11     con = sqlite3.connect("C:/CookPython/naverDB")      # DB가 저장된 폴더까지 지정
```

## Section 04 데이터의 입력과 조회

```
12     cur = con.cursor()
13
14     data1 = edt1.get(); data2 = edt2.get(); data3 = edt3.get(); data4 = edt4.get()
15     try :
16         sql = "INSERT INTO userTable VALUES('" + data1 + "', '" + data2 + "', '" + data3 +
17             "'" + data4 + "')"
18         cur.execute(sql)
19     except :
20         messagebox.showerror('오류', '데이터 입력 오류가 발생함')
21     else :
22         messagebox.showinfo('성공', '데이터 입력 성공')
23     con.commit()
24     con.close()
25
26 def selectData() :
27     strData1, strData2, strData3, strData4 = [], [], [], []
28     con = sqlite3.connect("C:/CookPython/naverDB")      # DB가 저장된 폴더까지 지정
29     cur = con.cursor()
30     cur.execute("SELECT * FROM userTable")
31     strData1.append("사용자ID"); strData2.append("사용자이름")
32     strData3.append("이메일"); strData4.append("출생연도")
33     strData1.append("-----"); strData2.append("-----")
34     strData3.append("-----"); strData4.append("-----")
35     while (True) :
```

## Section 04 데이터의 입력과 조회

```
35         row = cur.fetchone()
36         if row == None :
37             break;
38         strData1.append(row[0]); strData2.append(row[1])
39         strData3.append(row[2]); strData4.append(row[3])
40
41         listData1.delete(0, listData1.size() - 1); listData2.delete(0, listData2.size() - 1)
42         listData3.delete(0, listData3.size() - 1); listData4.delete(0, listData4.size() - 1)
43         for item1, item2, item3, item4 in zip(strData1, strData2, strData3, strData4) :
44             listData1.insert(END, item1); listData2.insert(END, item2)
45             listData3.insert(END, item3); listData4.insert(END, item4)
46         con.close()
47
48     ## 메인 코드 부분 ##
49     window = Tk()
50     window.geometry("600x300")
51     window.title("GUI 데이터 입력")
52
53     edtFrame = Frame(window);
54     edtFrame.pack()
55     listFrame = Frame(window)
```

```
56 listFrame.pack(side = BOTTOM, fill = BOTH, expand = 1)
57
58 edt1 = Entry(edtFrame, width = 10); edt1.pack(side = LEFT, padx = 10, pady = 10)
59 edt2 = Entry(edtFrame, width = 10); edt2.pack(side = LEFT, padx = 10, pady = 10)
60 edt3 = Entry(edtFrame, width = 10); edt3.pack(side = LEFT, padx = 10, pady = 10)
61 edt4 = Entry(edtFrame, width = 10); edt4.pack(side = LEFT, padx = 10, pady = 10)
62
63 btnInsert = Button(edtFrame, text = "입력", command = insertData)
64 btnInsert.pack(side = LEFT, padx = 10, pady = 10)
65 btnSelect = Button(edtFrame, text = "조회", command = selectData)
66 btnSelect.pack(side = LEFT, padx = 10, pady = 10)
67
68 listData1 = Listbox(listFrame, bg = 'yellow');
69 listData1.pack(side = LEFT, fill = BOTH, expand = 1)
70 listData2 = Listbox(listFrame, bg = 'yellow')
71 listData2.pack(side = LEFT, fill = BOTH, expand = 1)
72 listData3 = Listbox(listFrame, bg = 'yellow')
73 listData3.pack(side = LEFT, fill = BOTH, expand = 1)
74 listData4 = Listbox(listFrame, bg = 'yellow')
75 listData4.pack(side = LEFT, fill = BOTH, expand = 1)
76
77 window.mainloop()
```