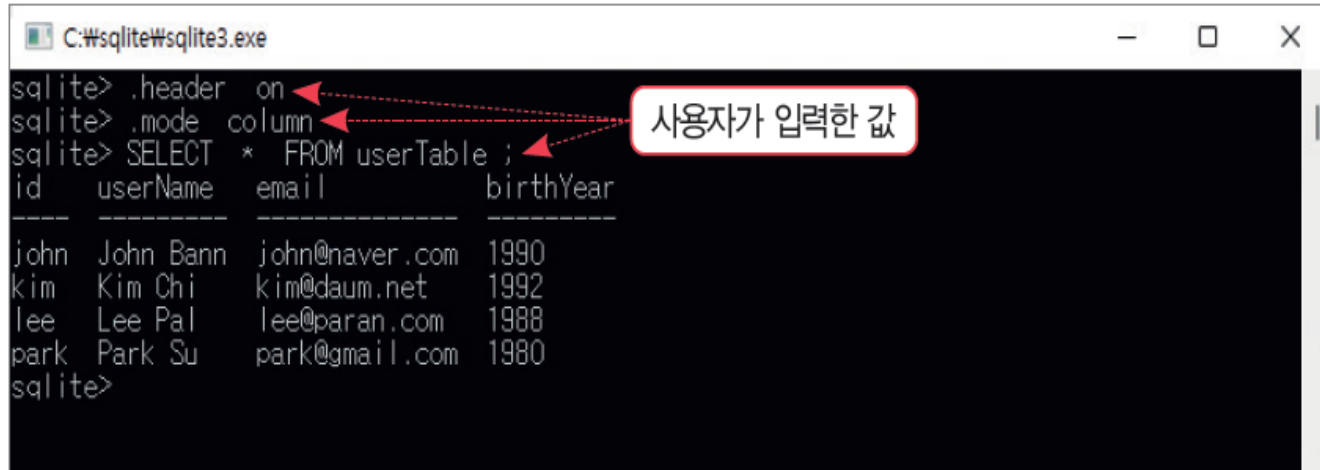


데이터베이스

- 데이터베이스의 개념을 익힌다.
- SQLite의 사용법과 SQL 문을 익힌다.
- 데이터베이스를 연동하는 프로그램을 만든다.

## Section 01 이 장에서 만들 프로그램

- [프로그램 1] 데이터베이스 운영
  - SQLite에서 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

- [프로그램 2] SQLite 활용
  - 파이썬에서 데이터베이스를 입력하고 조회하는 프로그램

## Section 02 데이터베이스 기본

### ■ 데이터베이스 개념

- 대량의 데이터가 발생하는 현대 사회에서 파일 처리로는 한계가 있음
- 데이터베이스는 이를 보완하는 방법

## Section 02 데이터베이스 기본

### ■ 데이터베이스 개념

- 데이터베이스를 익히려면 데이터베이스 소프트웨어를 설치해야 함
- 이 소프트웨어를 DBMS(DataBase Management System 또는 DataBase Management Software)라고 함
- DBMS는 데이터베이스를 관리해 주는 시스템 또는 소프트웨어



그림 8-1 DBMS 제품 종류

## Section 02 데이터베이스 기본

### ■ 데이터베이스 종류

- 계층형(Hierarchical), 망형(Network), 관계형(Relational), 객체지향형(Object -Oriented), 객체관계형(Object-Relational)
- 일반적으로 관계형 DBMS를 가장 많이 사용
- 오라클, SQL 서버, 액세스, MySQL, SQLite 등은 모두 관계형 DBMS, 즉 RDBMS에 속함

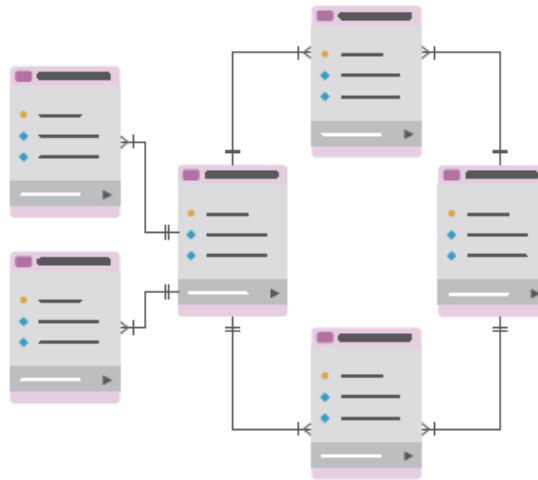


그림 8-2 RDBMS

## Section 02 데이터베이스 기본

### ■ 데이터베이스 관련 용어

- 데이터베이스를 구축하려면 데이터베이스 모델링이라는 작업이 필요함
- 데이터베이스 모델링은 현실에서 사용되는 데이터를 DBMS 안에 어떻게 옮겨 놓을지 결정하는 과정

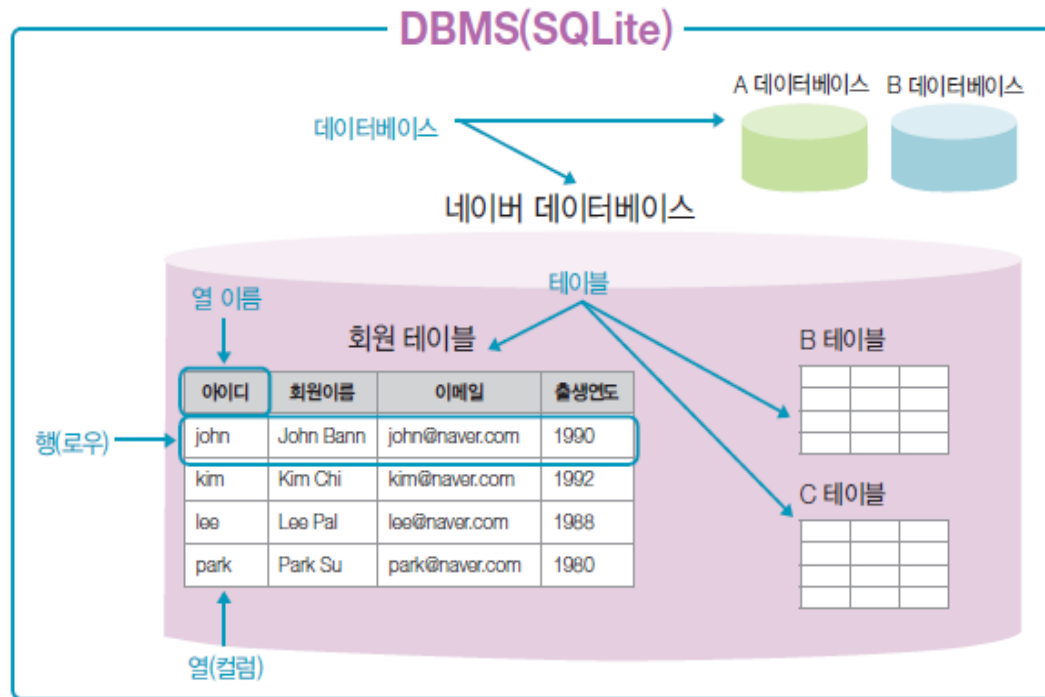


그림 8-3 DBMS 구성도

## Section 02 데이터베이스 기본

### ■ 데이터베이스 관련 용어

- 데이터: 단편적인 정보
- 테이블: 데이터가 표 형태로 표현된 것
- 데이터베이스: 테이블이 저장되는 저장소로, 주로 원통 모양으로 표현
- DBMS: 데이터베이스를 관리하는 시스템 또는 소프트웨어를 의미
- 열: 각 테이블은 1개 이상의 열로 구성
- 열 이름: 각 열을 구분하는 이름으로, 각 테이블 안에서 중복되지 않아야 함
- 데이터 형식: 열의 데이터 형식으로 테이블을 생성할 때 열 이름과 함께 지정
- 행: 실질적인 데이터
- SQL(Structured Query Language): DBMS가 알아듣는 언어



## Section 03 데이터베이스 구축

### ■ DBMS 설치와 실행

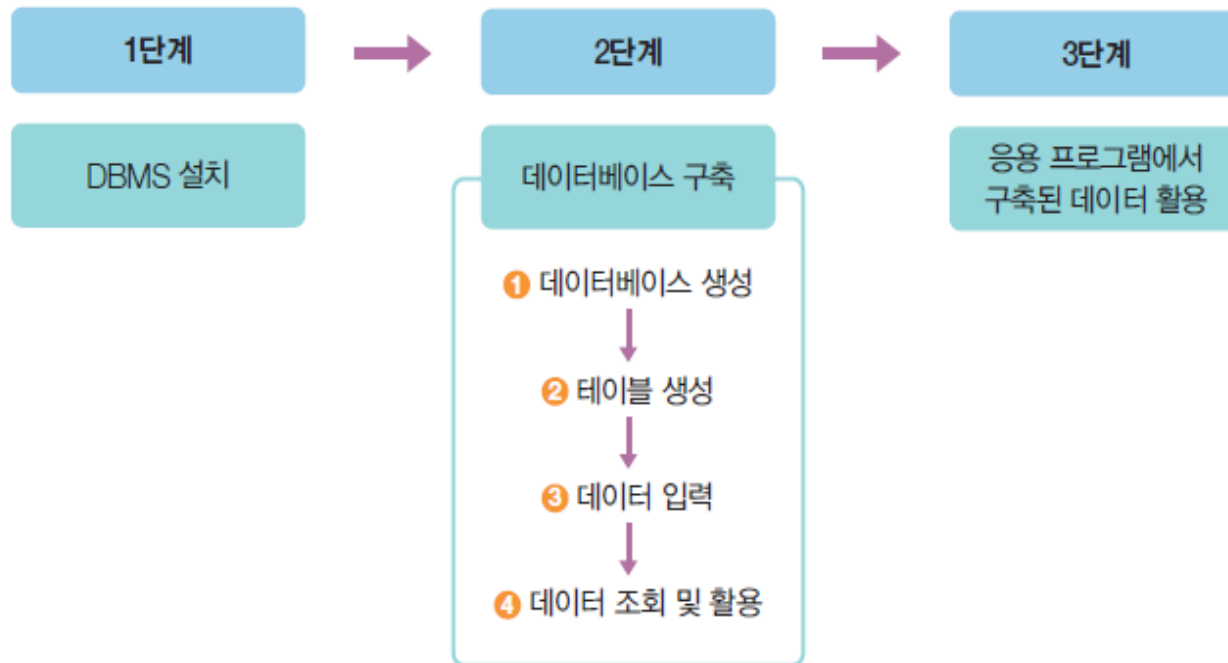


그림 8-4 데이터베이스 구축 및 운영 과정

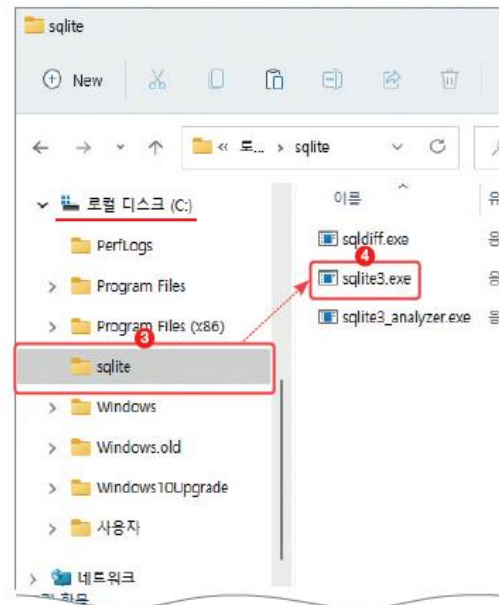
## Section 03 데이터베이스 구축

### ■ DBMS 설치와 실행

- SQLite 설치 페이지(<https://www.sqlite.org/download.html>)에 접속
- 'Precompiled Binaries for Windows' 항목에서 sqlite-tools-win32-x86-버전.zip 파일을 클릭하여 다운로드
- 다운로드한 파일의 압축을 풀고, 폴더명을 sqlite로 변경



(a) SQLite 다운로드



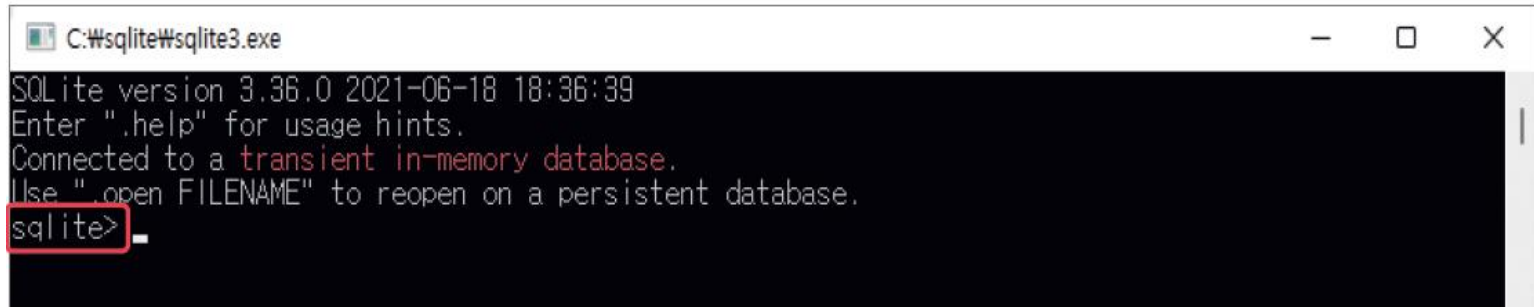
(b) SQLite 폴더

그림 8-5 DBMS 설치

## Section 03 데이터베이스 구축

### ■ DBMS 설치와 실행

- 파일 탐색기에서 sqlite3.exe 파일을 더블클릭해 실행하면 명령 프롬프트창이 열리면서 sqlite>로 표시됨



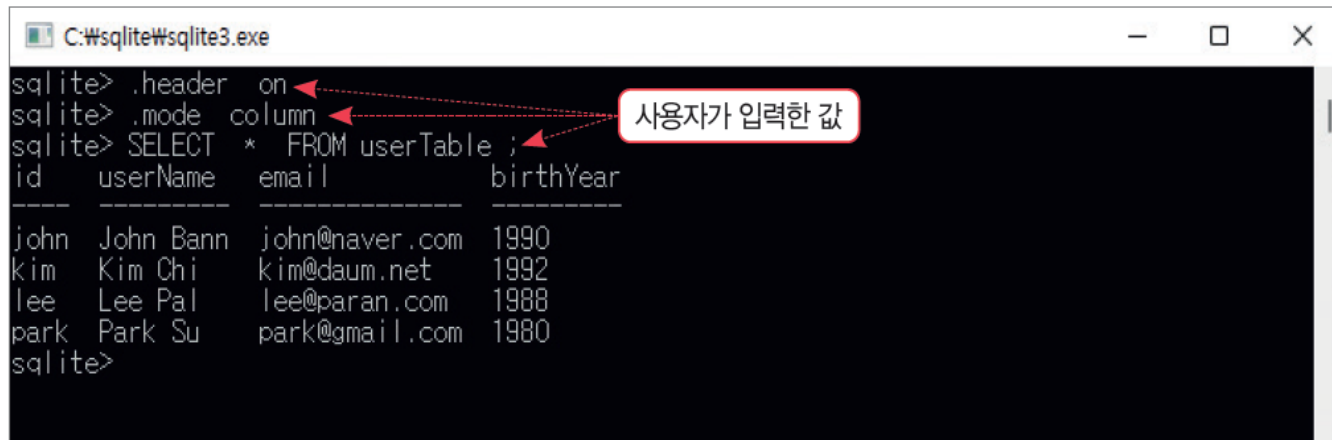
```
C:\sqlite\sqlite3.exe
SQLite version 3.36.0 2021-06-18 18:36:39
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> -
```

그림 8-6 SQLite 실행 화면

## Section 03 데이터베이스 구축

### ■ [프로그램 1] 완성

- SQLite에서 데이터베이스를 완성하는 것



```
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>
```

그림 8-7 완성된 데이터베이스 결과 확인

## Section 03 데이터베이스 구축

### ■ [프로그램 1] 완성

#### ■ 데이터 베이스 생성

- 데이터베이스를 생성하거나 열려면 '.open 데이터베이스이름' 명령어를 실행
- 데이터베이스가 있다면 열어 주고, 없다면 새로 생성하는 명령어
- 다음 명령어를 실행하면 [그림 8-3]의 네이버 데이터베이스(naverDB)가 생성됨



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

그림 8-8 비어 있는 naverDB 내부

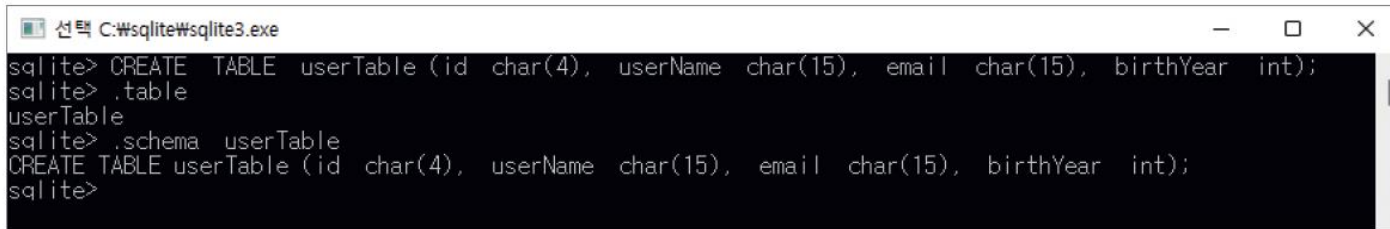
## Section 03 데이터베이스 구축

### ■ [프로그램 1] 완성

- 테이블 생성
  - 테이블을 생성하는 SQL 문의 형식

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

- 앞서 계획한 [그림 8-3]의 회원 테이블을 생성하고 확인하는 코드



```
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>
```

그림 8-9 테이블 생성

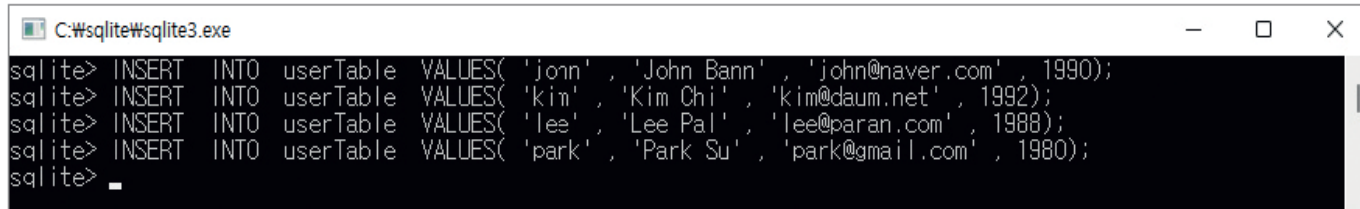
## Section 03 데이터베이스 구축

### ■ [프로그램 1] 완성

- 데이터 입력
  - 행 데이터를 입력하는 SQL 문의 형식

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

- [그림 8-3]의 회원 테이블에 행 4개를 입력하는 SQL 문



```
C:\sqlite\sqlite3.exe
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' , 1980);
sqlite> _
```

그림 8-10 데이터 입력

## Section 03 데이터베이스 구축

### ■ [프로그램 1] 완성

#### ■ 데이터 조회 및 활용

- 데이터를 조회 및 활용하는 SQL 문은 SELECT로 일반적인 형식은 다음과 같음

```
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>
```

사용자가 입력한 값

그림 8-11 데이터 조회

- SELECT 문을 WHERE 조건과 함께 사용할 수도 있음

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



## Section 03 데이터베이스 구축

### ■ [프로그램 1] 완성

#### ■ 데이터 조회 및 활용

```
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
```

1990년 이전에 태어난 사람의  
아이디와 출생연도를 확인

아이디가 'park'인 사람의 모든 정보를 조회  
\*는 모든 열을 의미

조회한 결과를 정렬하기 위해  
ORDER BY 문을 사용

작업이 모두 끝나 SQLite를 종료

## Section 03 데이터베이스 구축

### ■ [프로그램 1] 완성

#### SELF STUDY 8-1

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.

#### 실행 결과

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

## Section 04 데이터 입력과 조회

### ■ 데이터 입력 순서

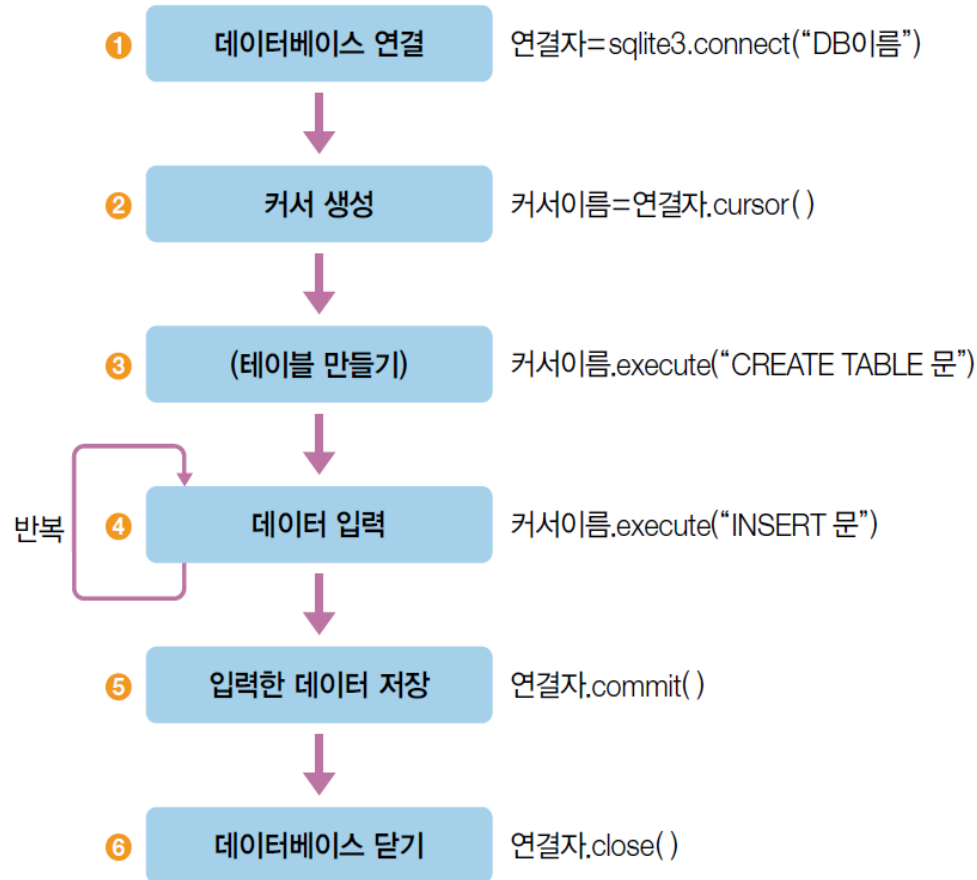


그림 8-12 SQLite의 데이터 입력 순서

## Section 04 데이터 입력과 조회

### ■ 데이터 입력 순서

#### ■ 데이터베이스 연결

- SQLite를 사용하려면 먼저 관련 모듈인 sqlite3를 임포트한 후 sqlite3.connect("DB이름")으로 데이터베이스와 연결

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

#### ■ 커서 생성

- 커서(Cursor): 데이터베이스에 SQL 문을 실행하거나 실행된 결과를 돌려받는 통로

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

## Section 04 데이터 입력과 조회

### ■ 데이터 입력 순서

#### ■ 테이블 만들기

- 테이블을 만드는 SQL 문을 커서이름.execute() 함수의 매개변수로 넘겨주면 SQL 문이 데이터베이스에 실행됨

```
cur.execute("CREATE TABLE userTable (id char(4), userName char(15), email char(20),  
birthYear int)")
```

#### 실행 결과

<sqlite3.Cursor object at 개체번호>

## Section 04 데이터 입력과 조회

### ■ 데이터 입력 순서

- 데이터 입력
  - 데이터는 필요한 만큼 반복해서 입력함

```
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)")
```

#### 실행 결과

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

## Section 04 데이터 입력과 조회

### ■ 데이터 입력 순서

#### ■ 입력한 데이터 저장

- 입력한 데이터 4건은 아직 데이터베이스에 완전히 저장한 것이 아닌 임시로 저장된 상태
- 이를 확실하게 저장하는 것을 커밋(Commit)이라고 함

```
con.commit()
```

#### ■ 데이터 베이스 닫기

- 데이터베이스를 모두 사용했다면 1번에서 연결한 데이터베이스를 닫아야 함

```
con.close()
```

## Section 04 데이터 입력과 조회

### ■ 데이터 입력 프로그램 구현

- 사용자가 반복해서 데이터를 입력하는 코드
- 앞서 생성한 naverDB의 userTable에 Enter를 입력하기 전까지 반복해서 한 행씩 데이터를 입력함



## Section 04 데이터 입력과 조회

### ■ 데이터 입력 프로그램 구현

Code08-01.py

```
01 import sqlite3
02 ## 변수 선언 부분 ##
03
04 con, cur = None, None
05 data1, data2, data3, data4 = "", "", "", ""
06 sql = ""
07
08 ## 메인 코드 부분 ##
09 con = sqlite3.connect("C:/CookAnalysis/naverDB") # DB가 저장된 폴더까지 지정
10 cur = con.cursor()
11
12 while (True) :
13     data1 = input("사용자ID ==> ")
14     if data1 == "" :
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 con.commit()
22 con.close()
```

실행 결과

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

## Section 04 데이터 입력과 조회

### ■ 데이터 입력 프로그램 구현

#### SELF STUDY 8-2

Code08-01.py를 수정해서 [SELF STUDY 8-1]에서 생성한 productTable이 입력되도록 하자.

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

## Section 04 데이터 입력과 조회

### ■ 데이터 조회 순서

- 파이썬에서 데이터를 조회하려면 아래 단계를 거쳐야 함

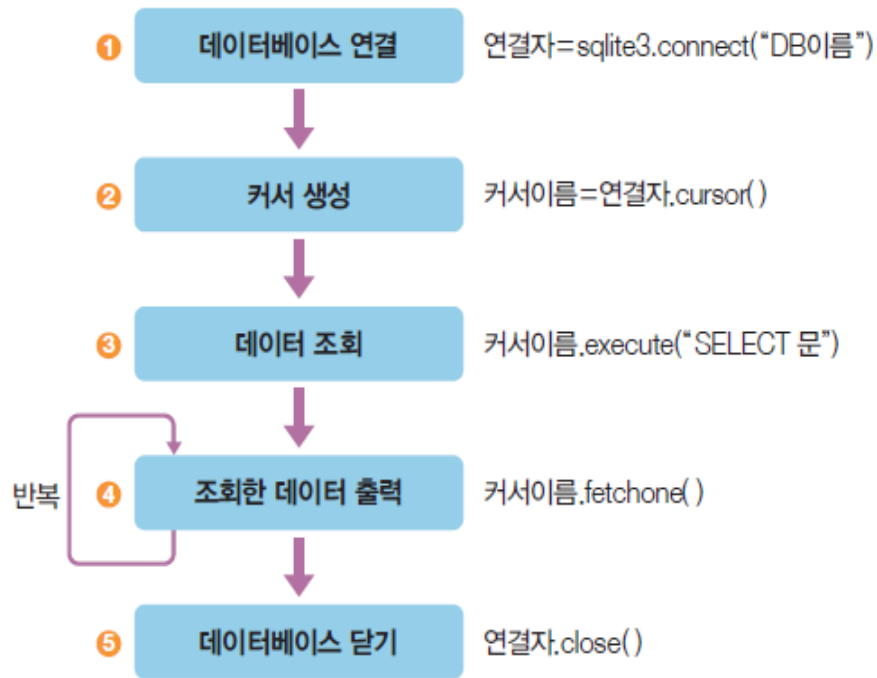


그림 8-13 SQLite의 데이터 조회 순서

## Section 04 데이터 입력과 조회

### ■ 데이터 조회 프로그램 구현

Code08-02.py

```
01  import sqlite3
02
03  ## 변수 선언 부분 ##
04  con, cur = None, None
05  data1, data2, data3, data4 = "", "", "", ""
06  row = None
07
08  ## 메인 코드 부분 ##
09  con = sqlite3.connect("C:/CookPython/naverDB")
10  cur = con.cursor()
11
12  cur.execute("SELECT * FROM userTable")
13
14  print("사용자ID 사용자이름 이메일 출생연도")
15  print("-----")
16
17  while (True) :
18      row = cur.fetchone()
19      if row == None :
20          break
```

## Section 04 데이터 입력과 조회

### ■ 데이터 조회 프로그램 구현

Code08-02.py

```
21     data1 = row[0]
22     data2 = row[1]
23     data3 = row[2]
24     data4 = row[3]
25     print("%5s %15s %20s %d" % (data1, data2, data3, data4))
26
27     con.close()
```

#### 실행 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 데이터 입력과 조회

### ■ 데이터 조회 프로그램 구현

#### SELF STUDY 8-3

Code08-02.py를 수정해서 [SELF STUDY 8-2]에서 입력한 productTable의 내용이 출력되도록 해 보자.

##### 실행 결과

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

## Section 04 데이터 입력과 조회

### ■ [프로그램 2] 완성

- GUI 화면에서 데이터를 입력하고 수정할 수 있도록 함

Code08-03.py

```
01  import sqlite3
02  from tkinter import *
03  from tkinter import messagebox
04
05  ## 함수 선언 부분 ##
06  def insertData() :
07      con, cur = None, None
08      data1, data2, data3, data4 = "", "", "", ""
09      sql = ""
10
11  con = sqlite3.connect("C:/CookAnalysis/naverDB") # DB가 저장된 폴더까지 지정
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 + \
```

## Section 04 데이터 입력과 조회

### ■ [프로그램 2] 완성

Code08-03.py

```
17         "','" + data3 + "','" + 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:/CookAnalysis/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("-----")
```



## Section 04 데이터 입력과 조회

### ■ [프로그램 2] 완성

Code08-03.py

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

## Section 04 데이터 입력과 조회

### ■ [프로그램 2] 완성

Code08-03.py

```
56  lis tF rame = F rame(w indow)56  lis tF rame.pack(s ide = BOTTOM, fill = BOTH, expand = 1)
57
58  ed t1 = Entry(ed tF rame, w id th = 10); ed t1.pack(s ide = LEFT, padx = 10, pady = 10)
59  ed t2 = Entry(ed tF rame, w id th = 10); ed t2.pack(s ide = LEFT, padx = 10, pady = 10)
60  ed t3 = Entry(ed tF rame, w id th = 10); ed t3.pack(s ide = LEFT, padx = 10, pady = 10)
61  ed t4 = Entry(ed tF rame, w id th = 10); ed t4.pack(s ide = LEFT, padx = 10, pady = 10)
62
63  btnInsert = Button(edtFrame, text = "입력", command = insertData)
64  btnInsert.pack(s ide = LEFT, padx = 10, pady = 10)
65  btnSelect = Button(edtFrame, text = "조회", command = selectData)
66  btnSelect.pack(s ide = LEFT, padx = 10, pady = 10)
67
68  lis tData1 = L is tbox(lis tF rame, bg = 'yellow');
69  lis tData1.pack(s ide = LEFT, fill = BOTH, expand = 1)
70  lis tData2 = L is tbox(lis tF rame, bg = 'yellow')
71  lis tData2.pack(s ide = LEFT, fill = BOTH, expand = 1)
72  lis tData3 = L is tbox(lis tF rame, bg = 'yellow')
73  lis tData3.pack(s ide = LEFT, fill = BOTH, expand = 1)
74  lis tData4 = L is tbox(lis tF rame, bg = 'yellow')
75  lis tData4.pack(s ide = LEFT, fill = BOTH, expand = 1)
76
77  w indow m a in loop ()
```