CHAP 12. 데이터베이스

데이터 스토리지

데이터를 저장하는 방법

방법	설명
공유 프레퍼런스(Shared Preferences):	키-값 쌍(key-value pair)으로 사적이고 기초적인 데이터를 저장한다.
내부 저장(Internal Storage)	사적인 데이터를 내부 저장소에 저장한다.
외부 저장(External Storage)	공유 데이터를 공유 외부 저장소에 저장한다.
SQLite 데이터베이스(SQLite Databases)	구조화된 데이터를 사적인 데이터베이스에 저장한다.
네트워크 연결(Network Connection)	데이터를 네트워크 서버에 저장한다.

공유 프레퍼런스

- 애플리케이션의 환경 설정
- 기초적인 자료형을 키-값 쌍으로 저장하고 복원할 수 있는 방법
- 저장된 데이터는 사용자 애플리케이션이 종료되더라도 저장

공유 프레퍼런스를 얻는 방법

- getSharedPreferences(name, mode)
- getPreferences(mode)

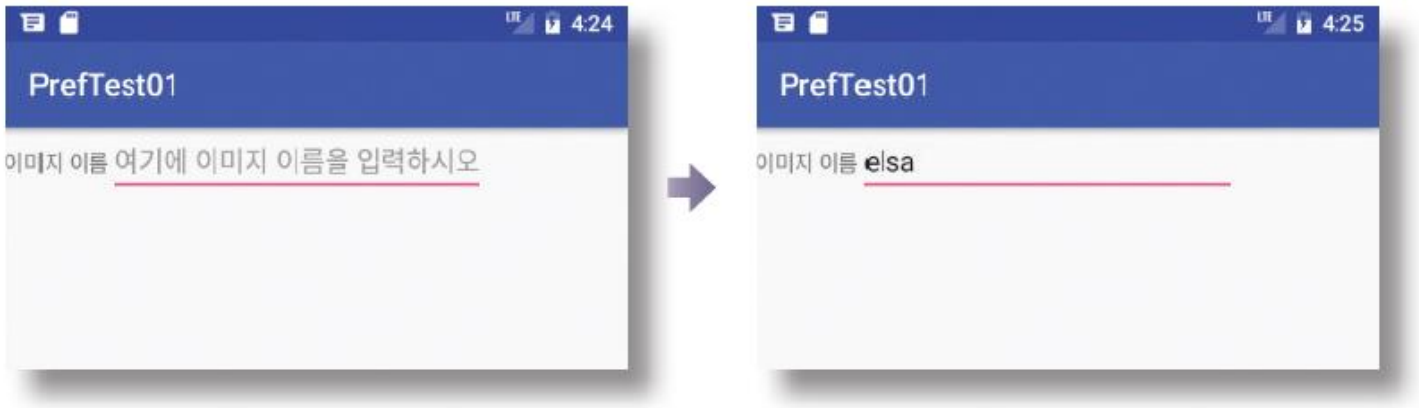
프레퍼런스 파일에서 값을 읽거나 쓰는 메소드

- getBoolean(String key, boolean defValue)
- getInt(String key, int defValue)
- getString(String key, String defValue)

- putBoolean(String key, boolean value)
- putInt(String key, int value)
- putString(String key, String value)

예제: 문자열을 프레퍼런스에 기록하고 읽는다.

- 문자열을 입력하고 BACK를 누르더라도 문자열이 없어지지 않는 예제



예제: 사용자 인터페이스 작성

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal" >

    <TextView
        android:id="@+id/TextView01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="이미지 이름" >
    </TextView>

    <EditText
        android:id="@+id/EditText01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:hint="여기에 이미지 이름을 입력하시오"
        android:text="" >
    </EditText>

</LinearLayout>
```

예제: 액티비티 작성

```
public class MainActivity extends AppCompatActivity {
    public static final String PREFS_NAME = "MyPrefs";
    TextView name;
    EditText value;
    String imageName;
    @Override
    protected void onCreate(Bundle state){
          super.onCreate(state);
          setContentView(R.layout.main);

          name = (TextView)findViewById(R.id.TextView01);
          value = (EditText)findViewById(R.id.EditText01);

          SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
          imageName = settings.getString("imageName", "");
          value.setText(imageName);
    }
    @Override
    protected void onStop(){
          super.onStop();
          SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
          SharedPreferences.Editor editor = settings.edit();
          imageName = value.getText().toString();
          editor.putString("imageName", imageName);
          editor.commit();
    }
}
```

실행 결과


4:24
PrefTest01
이미지 이름 여기에 이미지 이름을 입력하시오
4:25
PrefTest01
이미지 이름 elsa
Back 키
다시 실행

안드로이드에서의 파일 저장 장소

방법	설명
공유 프레퍼런스(Shared Preferences):	키-값 쌍(key-value pair)으로 사적이고 기초적인 데이터를 저장한다.
내부 저장(Internal Storage)	사적인 데이터를 내부 저장소에 저장한다.
외부 저장(External Storage)	공유 데이터를 공유 외부 저장소에 저장한다.
SQLite 데이터베이스(SQLite Databases)	구조화된 데이터를 사적인 데이터베이스에 저장한다.
네트워크 연결(Network Connection)	데이터를 네트워크 서버에 저장한다.

내부 공간에 파일 만들기

- 애플리케이션은 장치의 내부 저장 공간에 파일을 저장할 수 있다.
- 내부 저장 공간에 저장되는 파일은 해당 애플리케이션만 접근이 가능하다.
- 사용자가 애플리케이션을 제거하면 이들 파일들도 제거된다.

내부 공간에 파일 만들기: UI 설계

```
<LinearLayout ...
        <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        android:layout_weight="1"
        <EditText ...
                android:singleLine="false" />
        </LinearLayout>

        <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">
        <Button ... android:text="읽기"/>
        <Button ... android:text="쓰기"/>
        </LinearLayout>
</LinearLayout>
```

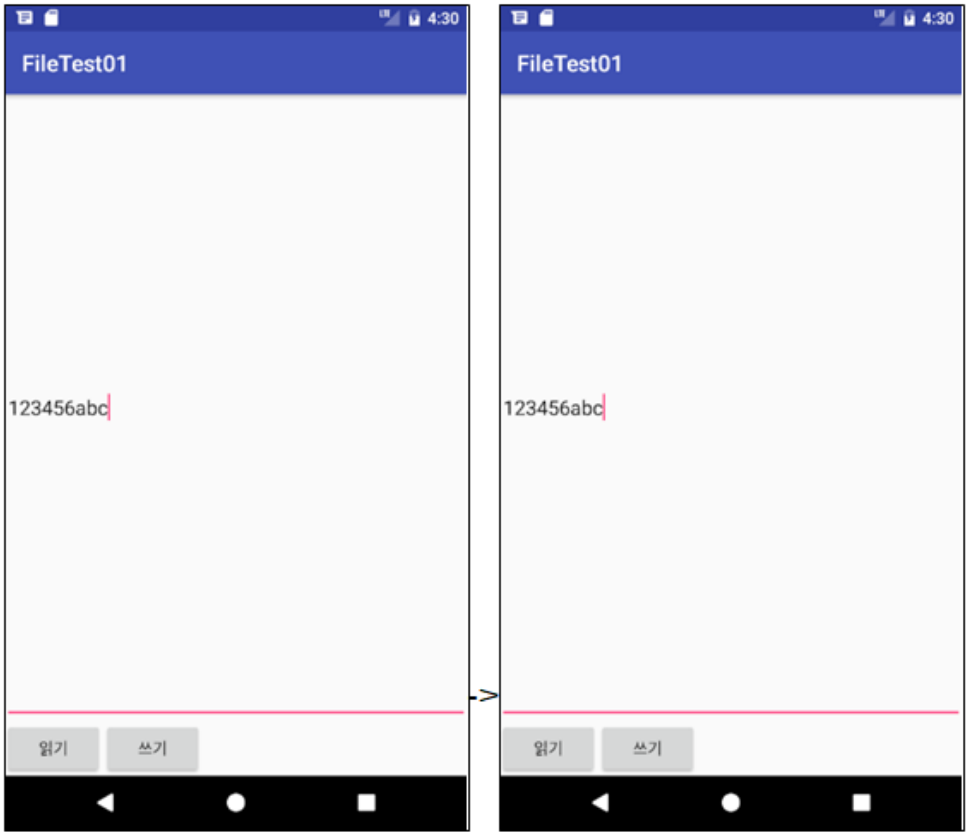
내부 공간에 파일 만들기

```
public class FileTest01 extends AppCompatActivity {
    String FILENAME = "test.txt";
    EditText edit;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        edit = (EditText) findViewById(R.id.EditText01);
        Button readButton = (Button) findViewById(R.id.read);
        readButton.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                try {
                    FileInputStream fis = openFileInput(FILENAME);
                    byte[] buffer = new byte[fis.available()];
                    fis.read(buffer);
                    edit.setText(new String(buffer));
                    fis.close();
                } catch (IOException e) {
                }
            }
        });
```

내부 공간에 파일 만들기

```
        Button writeButton = (Button) findViewById(R.id.write);
        writeButton.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                try {
                    FileOutputStream fos = openFileOutput(FILENAME,
                            Context.MODE_PRIVATE);
                    fos.write(edit.getText().toString().getBytes());
                    fos.close();
                } catch (IOException e) {
                }
            }
        });
    }
}
```

실행 결과


4:30
FileTest01
123456abc
읽기
쓰기
4:30
FileTest01
123456abc
읽기
쓰기

외부 공간에 파일 만들기

- 외부 저장 공간은 착탈이 가능한 SD 카드
- 외부 저장 공간에 저장된 파일들은 누구나 읽을 수 있으며 사용자에 의해서 변경될 수 있다.
- 데이터를 읽기 전에 외부 미디어가 장착되어 있는지를 확인

내부 공간과 외부 공간 비교

내부 공간	외부 공간
항상 사용 가능하다.	항상 사용이 가능한 것은 아니다. 사용자가 SD 카드를 제거할 수도 있다.
여기에 저장되는 파일은 해당되는 앱만 사용이 가능하다.	누구나 읽을 수 있다.
사용자가 앱을 제거하면 시스템이 앱이 사용하였던 공간도 삭제한다.	사용자가 앱을 제거할 때, 공용 디렉터리에 저장된 파일은 삭제되지 않는다. 다만 getExternalFilesDir()가 반환하는 디렉터리에 파일을 저장한 경우만 시스템이 삭제한다.

외부 미디어 장착 여부 검사

```
String state = Environment.getExternalStorageState();
if(state.equals(Environment.MEDIA_MOUNTED))
        // 미디어에 쓰고 읽을 수 있다
 else if(state.equals(Environment.MEDIA_MOUNTED_READ_ONLY))
        // 미디어를 읽을 수 있다
 else
        // 쓰거나 읽을 수 없다
```

매니페스트 파일

```
<manifest ...>
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE"
                    android:maxSdkVersion="18" />
    ...
</manifest>
```

외부 파일 생성


```
void createExternalStoragePrivateFile() {
    // 외부 저장 공간의 디렉터리를 얻어서 파일 객체를 생성한다.
    File file = new File(getExternalFilesDir(null), "DemoFile.jpg");
    try {
        ...
    } catch (IOException e) {
        // 오류: 외부 미디어가 마운트되어 있지 않음
        ...
    }
}
```

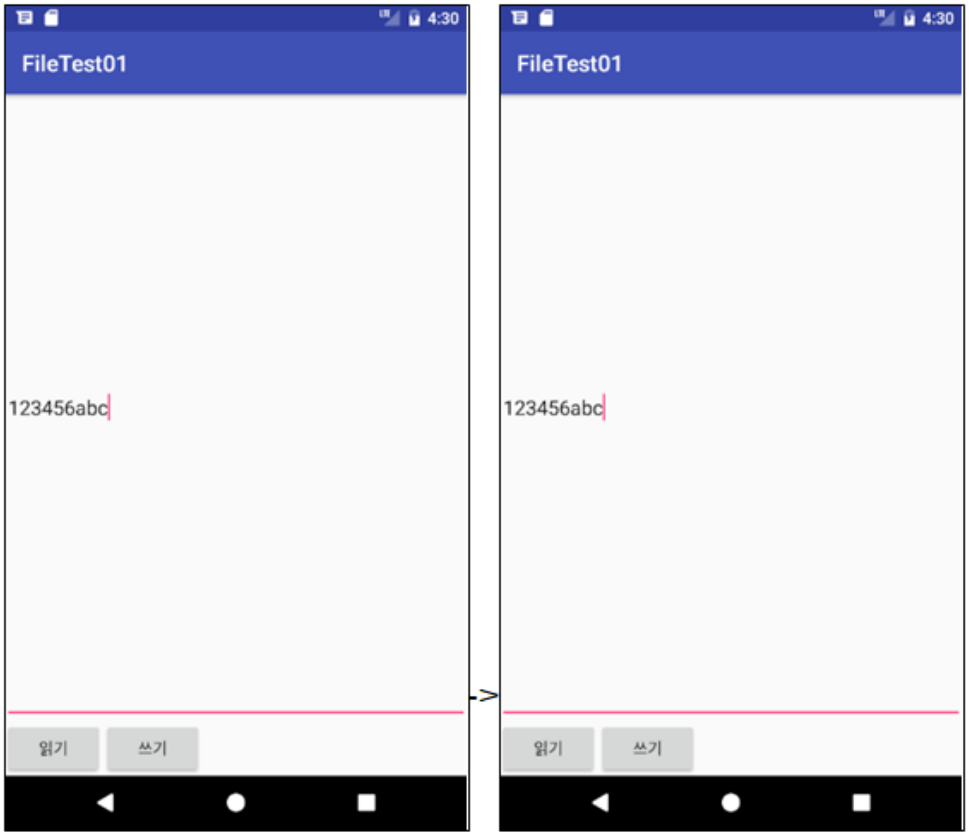
예제: 외부 공간에 파일 만들기

```
public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        String state = Environment.getExternalStorageState();
        if(state.equals(Environment.MEDIA_MOUNTED)==false){
              Toast.makeText(this,"외부 스토리지 실패
",Toast.LENGTH_SHORT).show();
        }
```

예제: 외부 공간에 파일 만들기

```
edit = (EditText) findViewById(R.id.EditText01);
Button readButton = (Button) findViewById(R.id.read);
readButton.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        File file = new File(getExternalFilesDir(null), FILENAME);
            try {
                InputStream is;
                is = new FileInputStream(file);
                byte[] buffer = new byte[is.available()];
                is.read(buffer);
                edit.setText(new String(buffer));
                is.close();
            } catch (Exception e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
    }
});
```

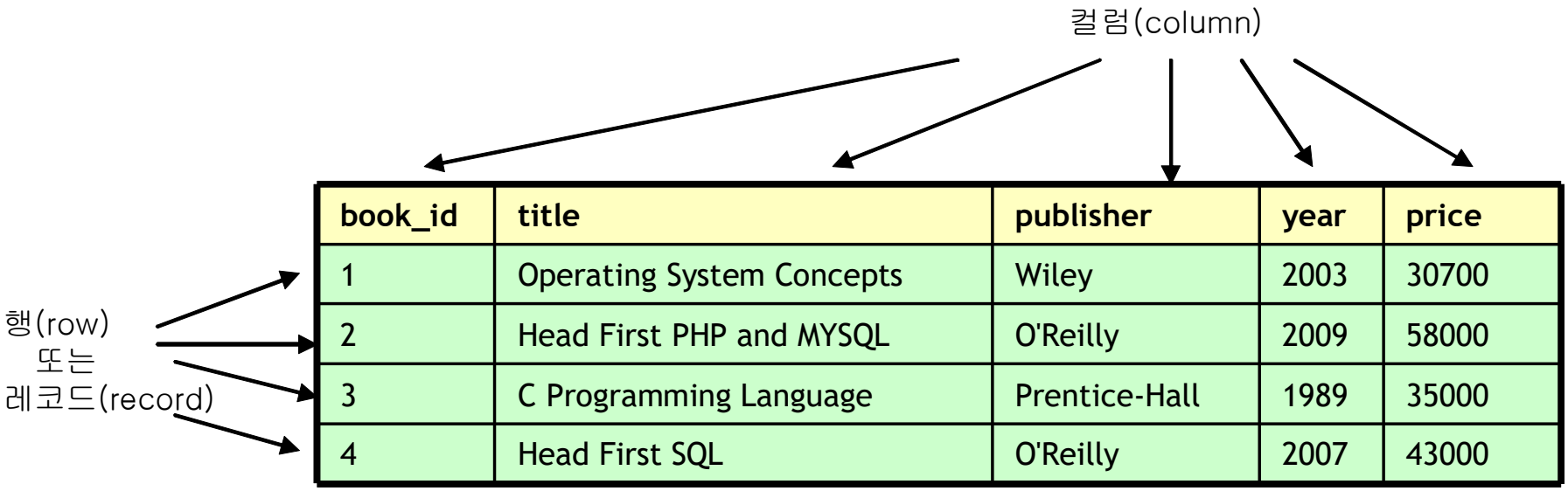
실행 결과


4:30
FileTest01
123456abc
읽기
쓰기
4:30
FileTest01
123456abc
읽기
쓰기

안드로이드에서의 데이터베이스

- **SQLite**
- www.sqlite.org
- SQLite는 안드로이드 와 아이폰을 비롯한 많은 모바일 장치에서 사용되는 데이터베이스
- SQL을 거의 완전하게 지원

데이터베이스



book_id	title	publisher	year	price
1	Operating System Concepts	Wiley	2003	30700
2	Head First PHP and MYSQL	O'Reilly	2009	58000
3	C Programming Language	Prentice-Hall	1989	35000
4	Head First SQL	O'Reilly	2007	43000

SQL

구분	명령어	설명
데이터 정의 명령어 (Data Definition Language)	CREATE	사용자가 제공하는 컬럼 이름을 가지고 테이블을 생성한다. 사용자는 컬럼의 데이터 타입도 지정해야 한다. 데이터 타입은 데이터베이스에 따라 달라진다. 이미 테이블이 만들어져 있는 경우가 많기 때문에 CREATE TABLE은 통상적으로 DML 보다 적게 사용된다.
	ALTER	테이블에서 컬럼을 추가하거나 삭제한다.
	DROP	데이블의 모든 레코드를 제거하고 테이블의 정의 자체를 데이터베이스로부터 삭제하는 명령어이다.
	USE	어떤 데이터베이스를 사용하는지 지정한다.
데이터 조작 명령어 (Data Manipulation Language)	SELECT	데이터베이스로부터 데이터를 쿼리하고 출력한다. SELECT 명령어들은 결과 집합에 포함시킬 컬럼을 지정한다. SQL 명령어 중에서 가장 자주 사용된다.
	INSERT	새로운 레코드를 테이블에 추가한다. INSERT는 새롭게 생성된 테이블을 채우거나 새로운 레코들를 이미 존재하는 테이블에 추가할 때 사용된다.
	DELETE	지정된 레코드를 테이블로부터 삭제한다.
	UPDATE	테이블에서 레코드에 존재하는 값을 변경한다.

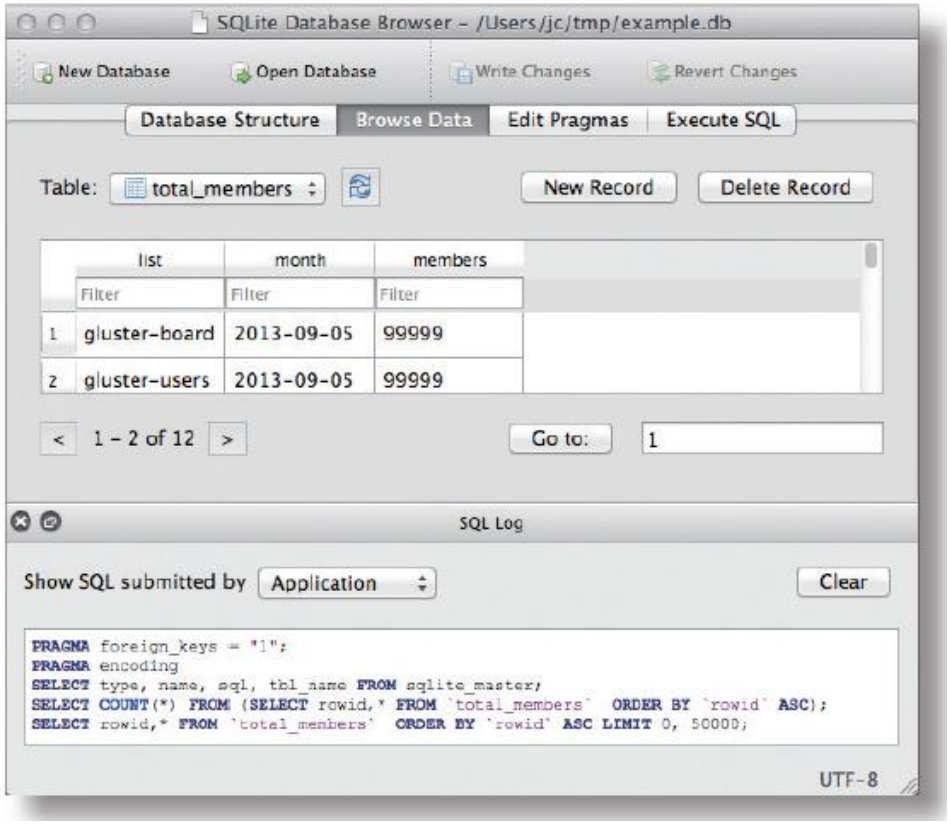
SQL의 예

```
INSERT INTO books (title, publisher, year, price)
      VALUES('Operating System Concepts', 'Wiley', '2003', 30700);
INSERT INTO books (title, publisher, year, price)
      VALUES('Head First PHP and MYSQL', 'OReilly', '2009', 58000);
INSERT INTO books (title, publisher, year, price)
      VALUES('C Programming Language ', 'Prentice-Hall', '1989', 35000);
INSERT INTO books (title, publisher, year, price)
      VALUES('Head First SQL', 'OReilly', '2007', 43000);
```

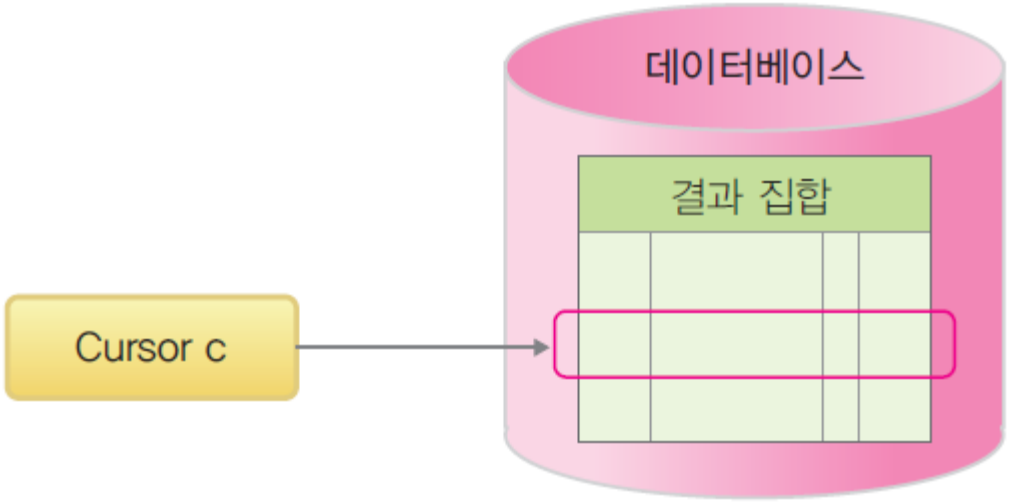
```
mysql> SELECT title, publisher, price FROM books;
+--------------------------+--------------+-------+
| title                    | publisher    | price |
+--------------------------+--------------+-------+
| Operating System Concepts | Wiley        | 30700 |
| Head First PHP and MYSQL | OReilly      | 58000 |
| C Programming Language   | Prentice-Hall | 35000 |
| Head First SQL           | OReilly      | 43000 |
+--------------------------+--------------+-------+
```

SQLite 브라우저

- SQLite를 사용하여서 테이블을 정의하고 데이터를 입력할 때 브라우저를 사용하면 편리하다.



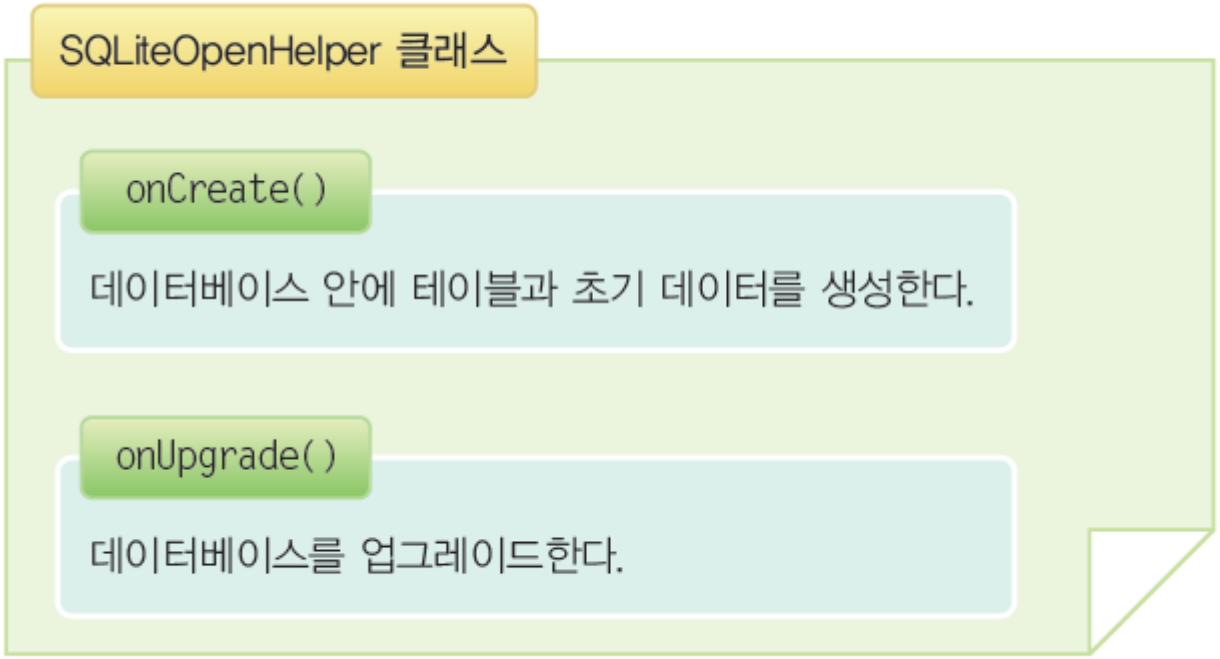
결과 집합(Result Sets)과 커서(Cursors)



안드로이드에서 데이터베이스 사용하는 2가지 방법

- SQLiteOpenHelper를 사용하는 방법이다.
- openOrCreateDatabase( ) 메소드로 데이터베이스 객체를 직접 생성하는 방법

SqLiteOpenHelper 클래스



SQLiteOpenHelper 클래스 이용하여 데이터베이스 생성하기

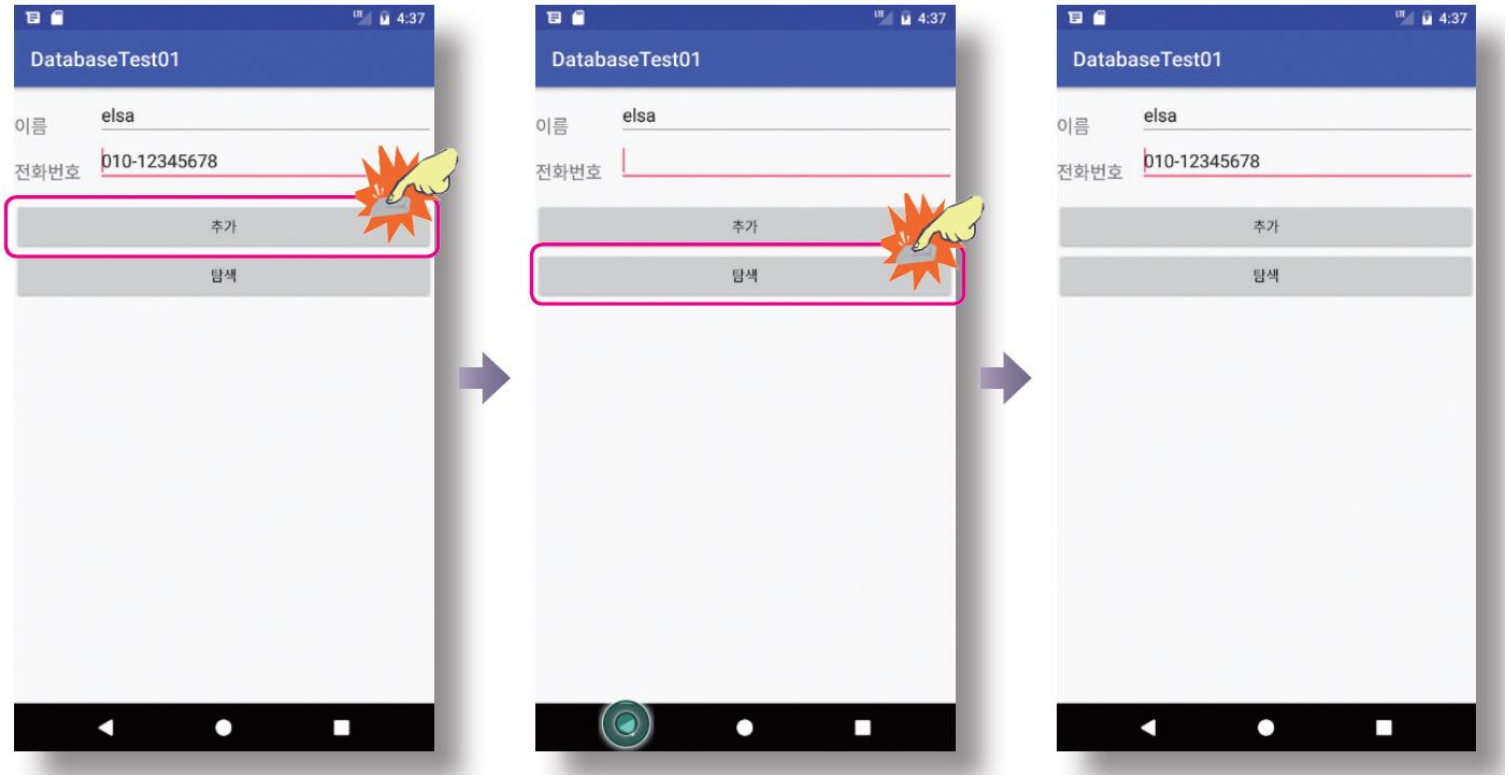
```
class dbHelper extends SQLiteOpenHelper {
        private static final String DATABASE_NAME = "mycontacts.db";
        private static final int DATABASE_VERSION = 2;

        public dbHelper(Context context) {
                super(context, DATABASE_NAME, null, DATABASE_VERSION);
        }

        public void onCreate(SQLiteDatabase db) {
                db.execSQL("CREATE TABLE contact ( _id INTEGER PRIMARY KEY
                        AUTOINCREMENT, " + "name TEXT, tel TEXT);");
        }

        public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
                db.execSQL("DROP TABLE IF EXISTS contact");
                onCreate(db);
        }
}
```

SQLiteOpenHelper 클래스 이용하여 데이터베이스 생성하기

```
public class MainActivity extends AppCompatActivity {
    dbHelper helper;
    SQLiteDatabase db;

    public void onCreate(Bundle savedInstanceState) {
        ...
        helper = new dbHelper(this);
        try {
            db = helper.getWritableDatabase();
        } catch (SQLiteException ex) {
            db = helper.getReadableDatabase();
        }
        // 이제 필요할 때마다 db를 통하여서 SQL문장을 실행하면 된다.
        db.execSQL("INSERT INTO contact VALUES (null, '" + name
                + "', '" + tel + "');");
    }
}
```

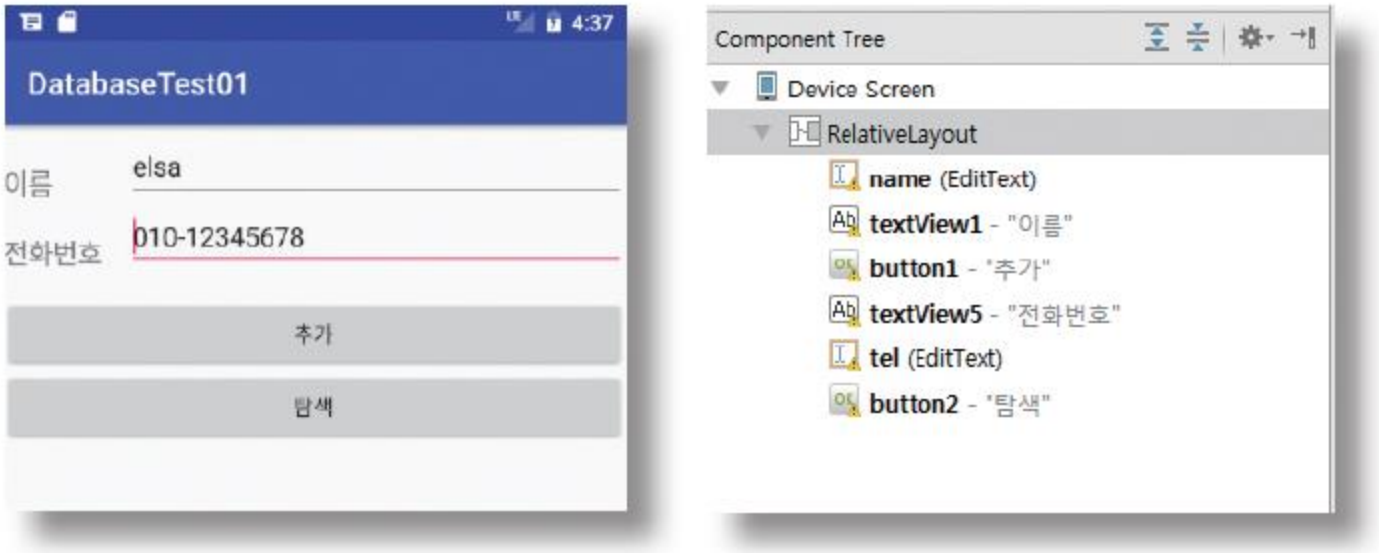
예제

- 연락처를 저장하고 검색하는 간단한 애플리케이션를 작성하여 보자.



사용자 인터베이스 작성

- 다음과 같은 사용자 인터페이스를 작성한다.



코드 작성

```
package com.example.hello;

class DBHelper extends SQLiteOpenHelper {
	private static final String DATABASE_NAME = "mycontacts.db";

	private static final int DATABASE_VERSION = 2;

	public DBHelper(Context context) {
		super(context, DATABASE_NAME, null, DATABASE_VERSION);
	}

	public void onCreate(SQLiteDatabase db) {
		db.execSQL("CREATE TABLE contacts ( _id INTEGER PRIMARY KEY"+
				" AUTOINCREMENT, name TEXT, tel TEXT);");
	}

	public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
		db.execSQL("DROP TABLE IF EXISTS contacts");
		onCreate(db);
	}
}
```

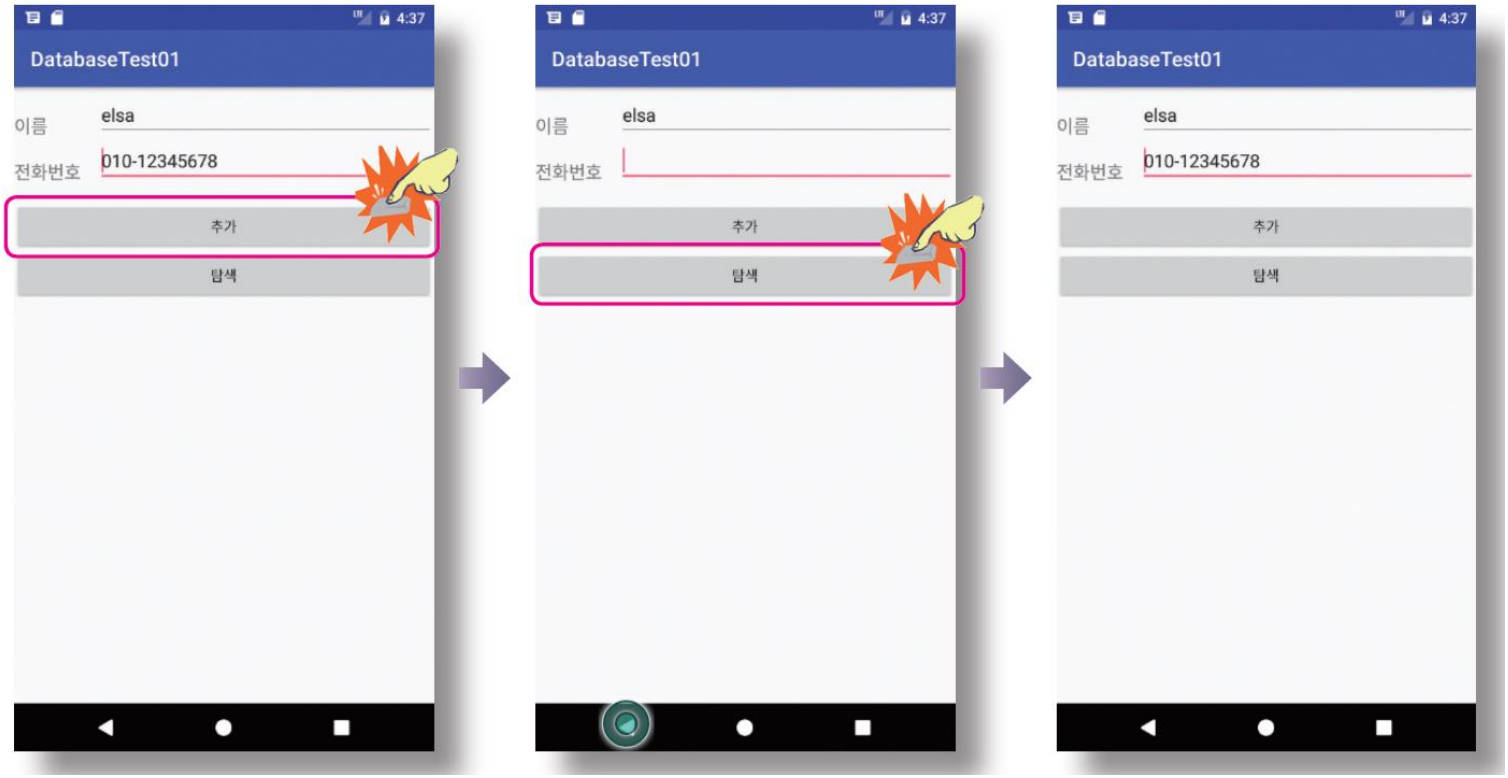
코드 작성

```
public class DatabaseTest01Activity extends AppCompatActivity {
	DBHelper helper;
	SQLiteDatabase db;
	EditText edit_name, edit_tel;

	public void onCreate(Bundle savedInstanceState) {

		super.onCreate(savedInstanceState);
		setContentView(R.layout.activity_main);
		helper = new DBHelper(this);
		try {
			db = helper.getWritableDatabase();
		} catch (SQLiteException ex) {
			db = helper.getReadableDatabase();
		}
		edit_name = (EditText) findViewById(R.id.name);
		edit_tel = (EditText) findViewById(R.id.tel);
	}
```

코드 작성

```
    public void insert(View target) {
        String name = edit_name.getText().toString();
        String tel = edit_tel.getText().toString();
        db.execSQL("INSERT INTO contacts VALUES (null, '" + name + "', '" + tel
                + "');");

        Toast.makeText(getApplicationContext(), "성공적으로 추가되었음",
                Toast.LENGTH_SHORT).show();
        edit_name.setText("");
        edit_tel.setText("");
    }

    public void search(View target) {
        String name = edit_name.getText().toString();
        Cursor cursor;
        cursor = db.rawQuery("SELECT name, tel FROM contacts WHERE name='"
                + name + "';", null);

        while (cursor.moveToNext()) {
            String tel = cursor.getString(1);
            edit_tel.setText(tel);
        }

    }

}
```

실행 결과

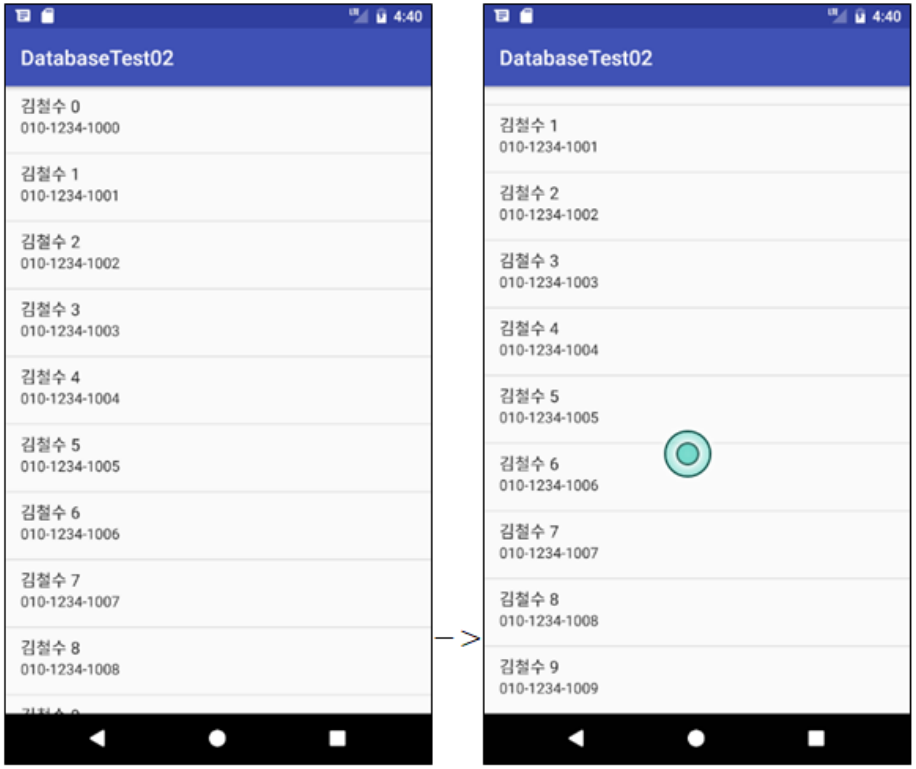

4:37
DatabaseTest01
이름
elsa
전화번호
010-12345678
추가
탐색
4:37
DatabaseTest01
이름
elsa
전화번호
추가
탐색
4:37
DatabaseTest01
이름
elsa
전화번호
010-12345678
추가
탐색

SQLiteOpenHelper 없이 데이터베이스 사용하기

```
db = openOrCreateDatabase(DATABASE_NAME, null);
db.execSQL("CREATE TABLE contact ( _id INTEGER PRIMARY KEY
          AUTOINCREMENT, "+ "name TEXT, tel TEXT);");
…
```

데이터베이스와 어댑터

- **SimpleCursorAdapter** 객체는 데이터베이스와 화면을 연결한다.
- 데이터베이스에서 데이터를 읽어서 정해진 레이아웃으로 화면에 표시한다.



코드 작성

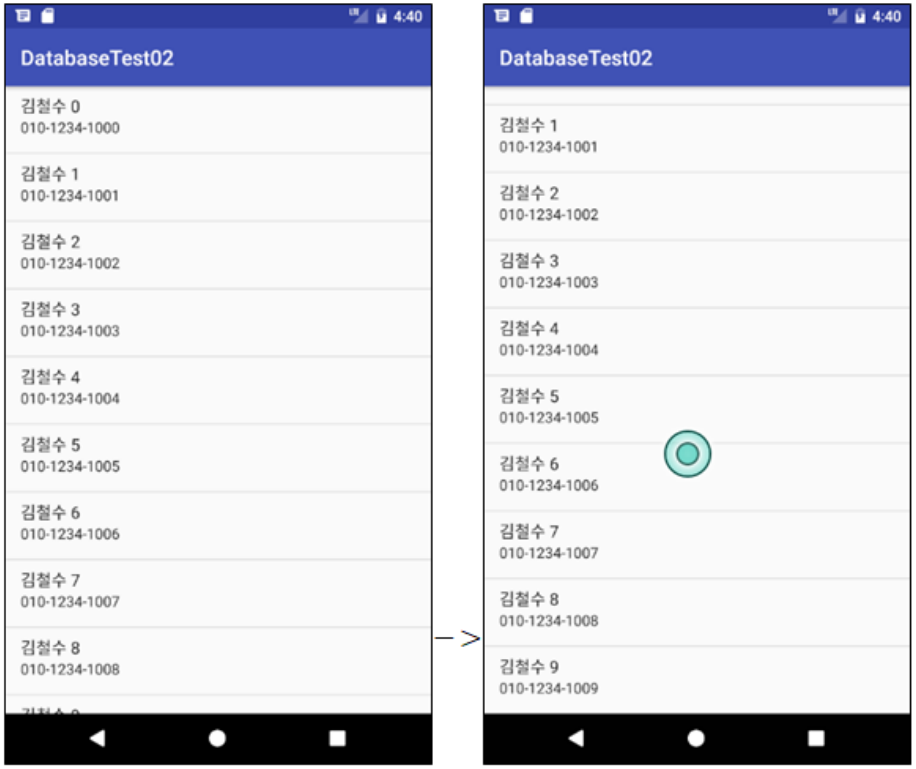
```
public class MainActivity extends AppCompatActivity {
          dbHelper helper;
          SQLiteDatabase db;
          EditText edit_name, edit_tel;

          public void onCreate(Bundle savedInstanceState) {
                    super.onCreate(savedInstanceState);
                    setContentView(R.layout.activity_main);

                    helper = new dbHelper(this);
                    db = helper.getWritableDatabase();
                    Cursor cursor = db.rawQuery("SELECT * FROM contacts", null);

                    startManagingCursor(cursor);
                    String[] from = {"name", "tel" };
                    int[] to = { android.R.id.text1, android.R.id.text2 };
                    SimpleCursorAdapter adapter = new SimpleCursorAdapter(this,
                                        android.R.layout.simple_list_item_2, cursor,
from, to);
                    ListView list = (ListView) findViewById(R.id.list);
                    list.setAdapter(adapter);

          }

}
```

코드 작성

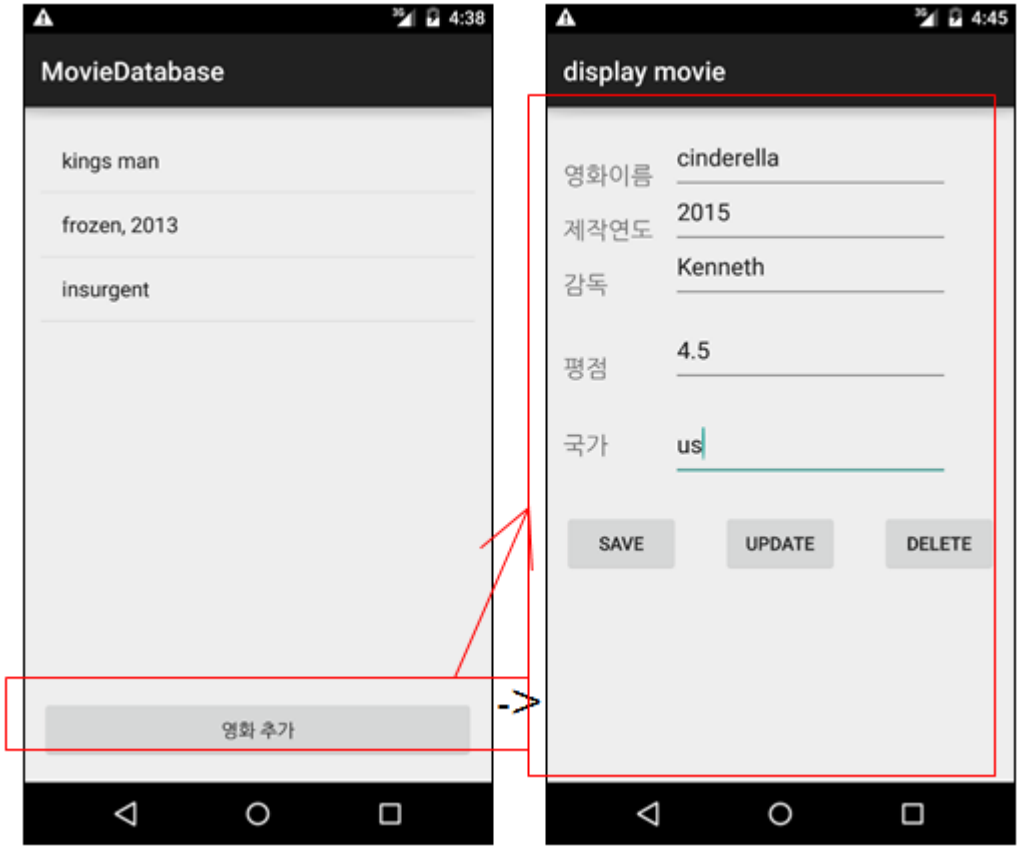
```
class dbHelper extends SQLiteOpenHelper {
        private static final String DATABASE_NAME = "mycontacts.db";
        private static final int DATABASE_VERSION = 2;

        public dbHelper(Context context) {
                super(context, DATABASE_NAME, null, DATABASE_VERSION);
        }
        public void onCreate(SQLiteDatabase db) {
                db.execSQL("CREATE TABLE contacts ( _id INTEGER PRIMARY KEY
AUTOINCREMENT, “ + "name TEXT, tel TEXT);");
                for (int i = 0; i < 10; i++) {
                        db.execSQL("INSERT INTO contacts VALUES (null, " +
"'김철수 " + i+ "'" + ", '010-1234-100" + i + "');");
                }
        }
        public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)
{
                db.execSQL("DROP TABLE IF EXISTS contacts");
                onCreate(db);
        }
}
```

데이터베이스와 어댑터

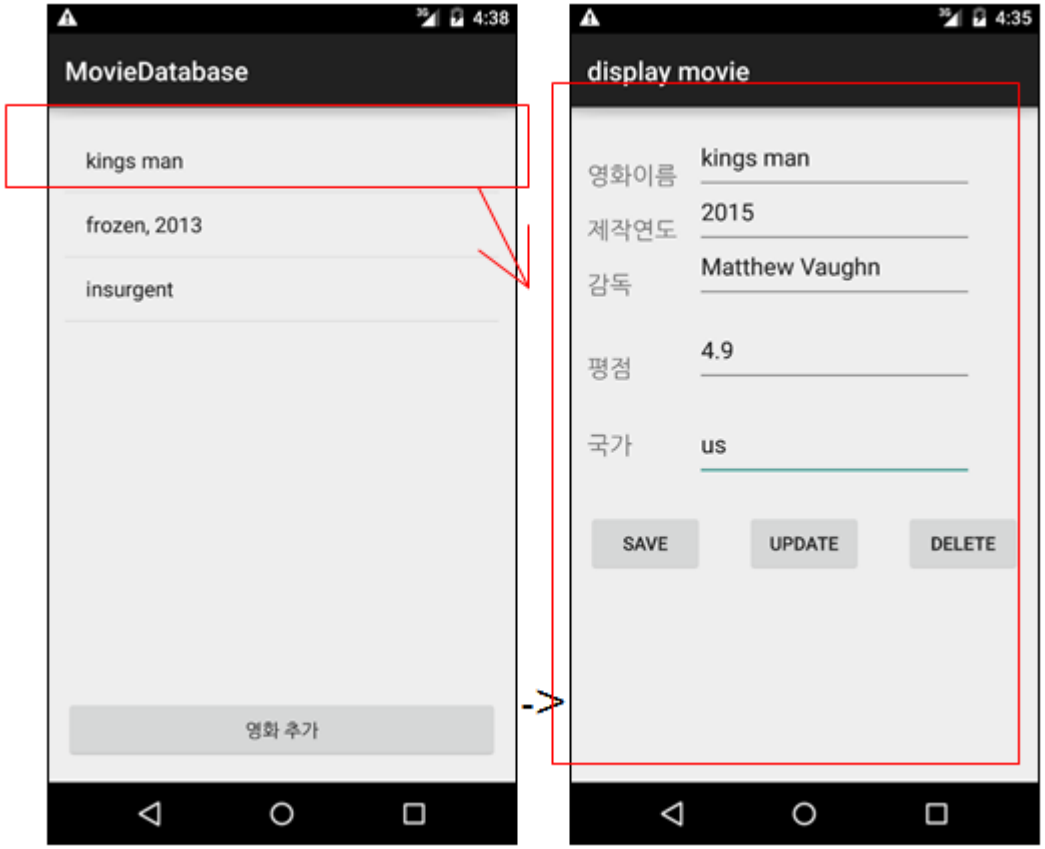
- **SimpleCursorAdapter** 객체는 데이터베이스와 화면을 연결한다.
- 데이터베이스에서 데이터를 읽어서 정해진 레이아웃으로 화면에 표시한다.



Lab: 영화 데이터베이스 만들기



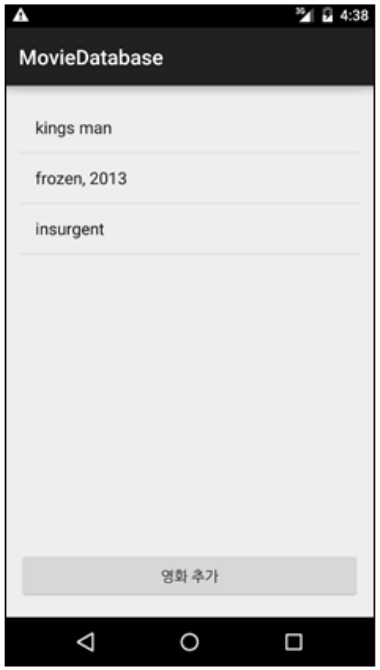
Lab: 영화 데이터베이스 만들기



사용자 인터페이스

main.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/btnAdd"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:onClick="onClick"
        android:text="영화 추가" />

    <ListView
        android:id="@+id/listView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/btnAdd"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true" />
</RelativeLayout>
```



데이터베이스 구조

필드이름	id	name	director	year	nation	rating
형식	integer	text	text	text	text	text
	,,,,					

소스 일부(다운로드 소스 참조)

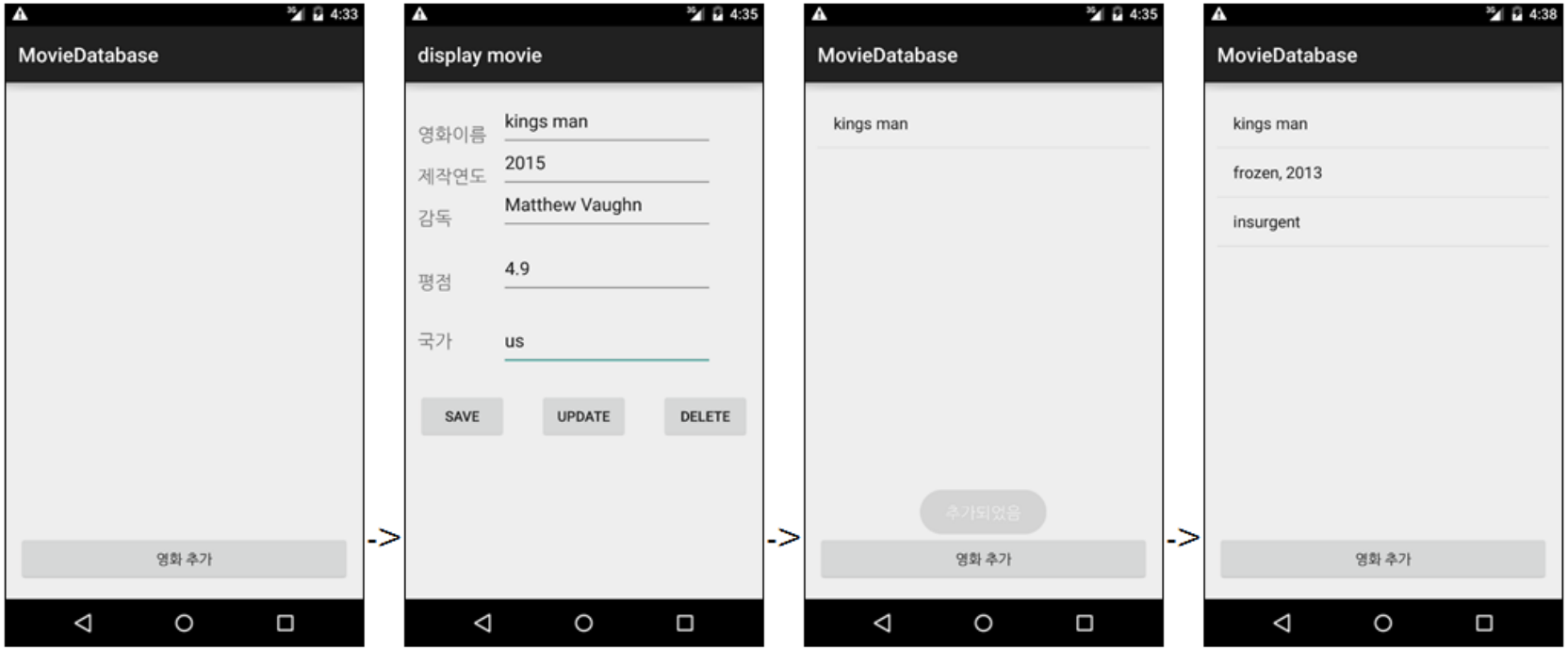
DBHelper.java

```
package kr.co.company.moviedatabase;
// 소스만 입력하고 Alt+Enter를 눌러서 import 문장을 자동으로 생성한다.

public class DBHelper extends SQLiteOpenHelper {

    public static final String DATABASE_NAME = "MyMovies.db";
    public static final String MOVIES_TABLE_NAME = "movies";
    public static final String MOVIES_COLUMN_ID = "id";
    public static final String MOVIES_COLUMN_NAME = "name";
    public static final String MOVIES_COLUMN_DIRECTOR = "director";
    public static final String MOVIES_COLUMN_YEAR = "year";
    public static final String MOVIES_COLUMN_NATION = "nation";
    public static final String MOVIES_COLUMN_RATING = "rating";

    public DBHelper(Context context) {
        super(context, DATABASE_NAME, null, 1);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("create table movies "
                + "(id integer primary key,name text, director text, year text, nation
text, rating text)");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS movies");
        onCreate(db);
    }
```

소스 일부(다운로드 소스 참조)

```
public boolean insertMovie(String name, String director, String year,
        String nation, String rating) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();

    contentValues.put("name", name);
    contentValues.put("director", director);
    contentValues.put("year", year);
    contentValues.put("nation", nation);
    contentValues.put("rating", rating);

    db.insert("movies", null, contentValues);
    return true;
}

public Cursor getData(int id) {
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor res = db.rawQuery("select * from movies where id=" + id + "",
            null);
    return res;
}

public int numberOfRows() {
    SQLiteDatabase db = this.getReadableDatabase();
    int numRows = (int) DatabaseUtils
            .queryNumEntries(db, MOVIES_TABLE_NAME);
    return numRows;
}

public boolean updateMovie(Integer id, String name, String director,
        String year, String nation, String rating) {
    SQLiteDatabase db = this.getWritableDatabase();
```

실행 결과


4:33
MovieDatabase
영화 추가
->
4:35
display movie
영화이름
kings man
제작연도
2015
감독
Matthew Vaughn
평점
4.9
국가
us
SAVE
UPDATE
DELETE
->
4:35
MovieDatabase
kings man
추가되었음
영화 추가
->
4:38
MovieDatabase
kings man
frozen, 2013
insurgent
영화 추가