군산대학교
KUNSAN NATIONAL UNIVERSITY

Android 데이터베이스(SQLite)

모바일 시스템 SW
남 광 우

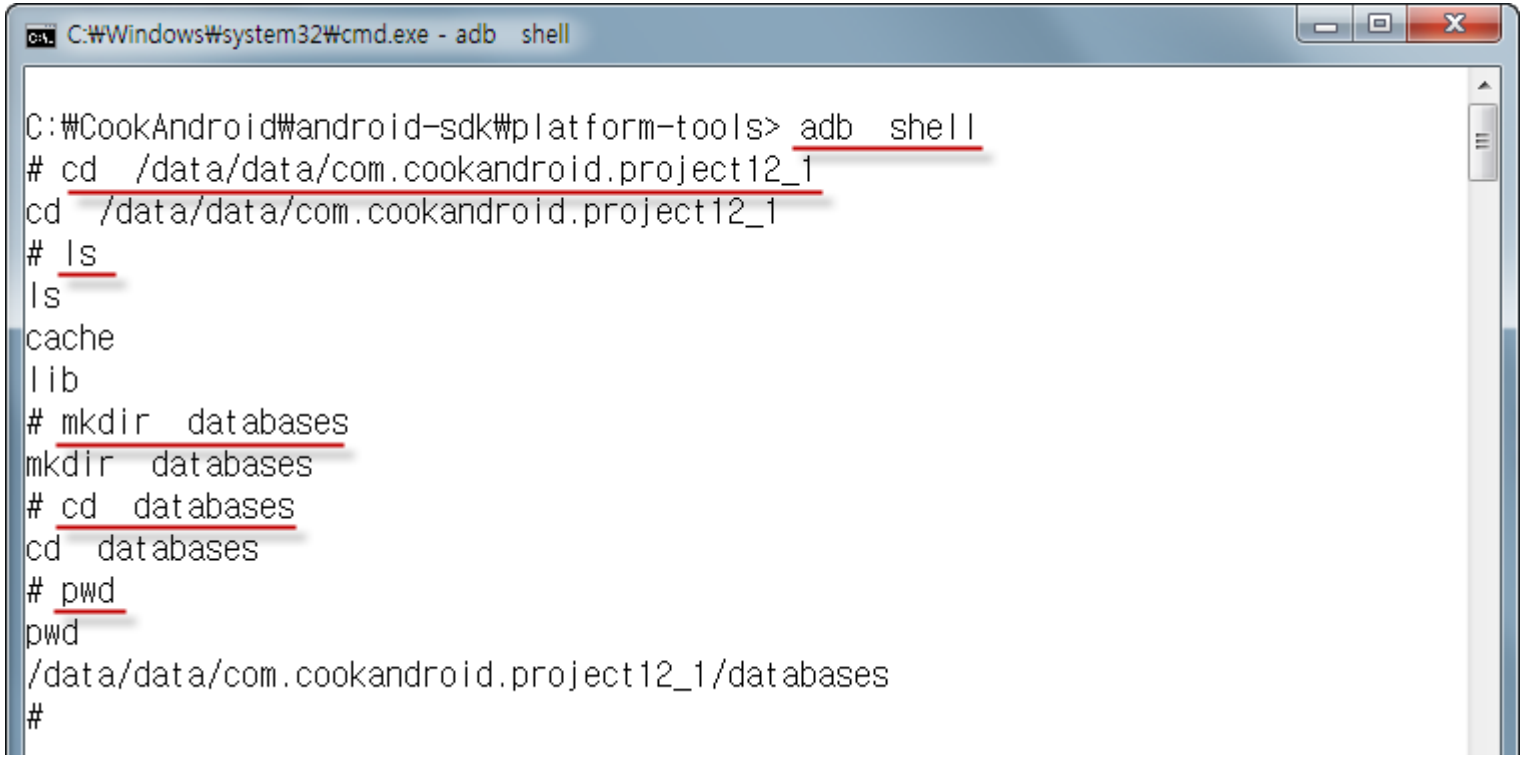
SQLite

❖ 소개

- ✓ SQLite 라이브러리를 통해 완전한 관계형 데이터베이스(RDBMS) 기능 제공
- ✓ 오픈 소스
- ✓ 표준 준수
- ✓ 경량
- ✓ 단일 계층

SQLite 기본

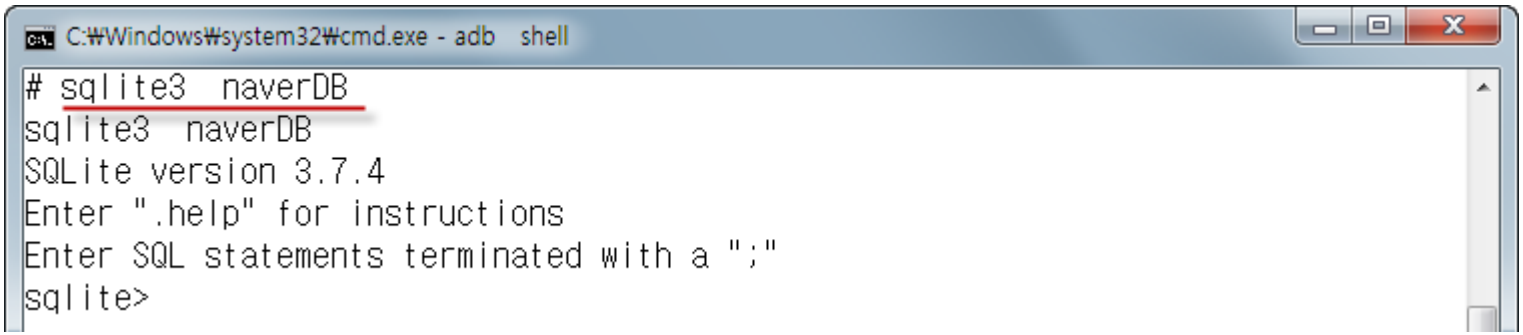
❖ SQLite 직접 사용해보기

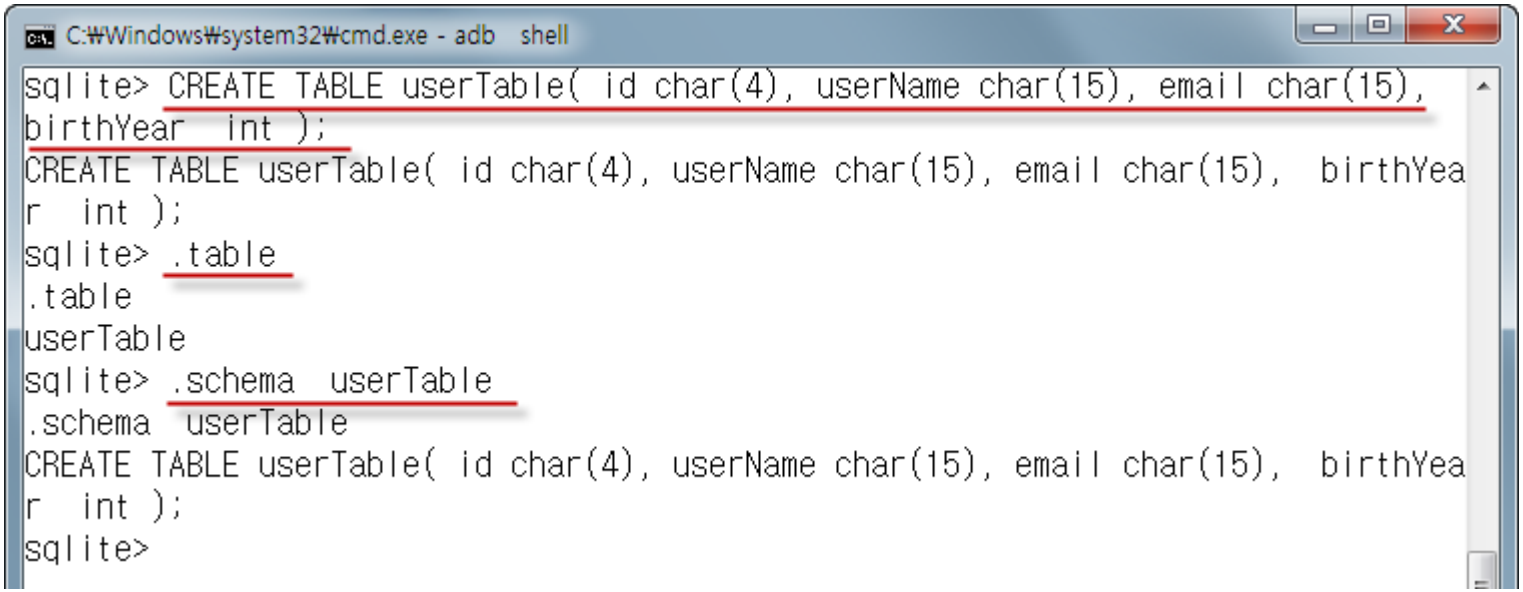


자료 : 한빛미디어 안드로이드 프로그래밍

SQLite 기본

❖ SQLite 직접 사용해보기 : 테이블의 생성

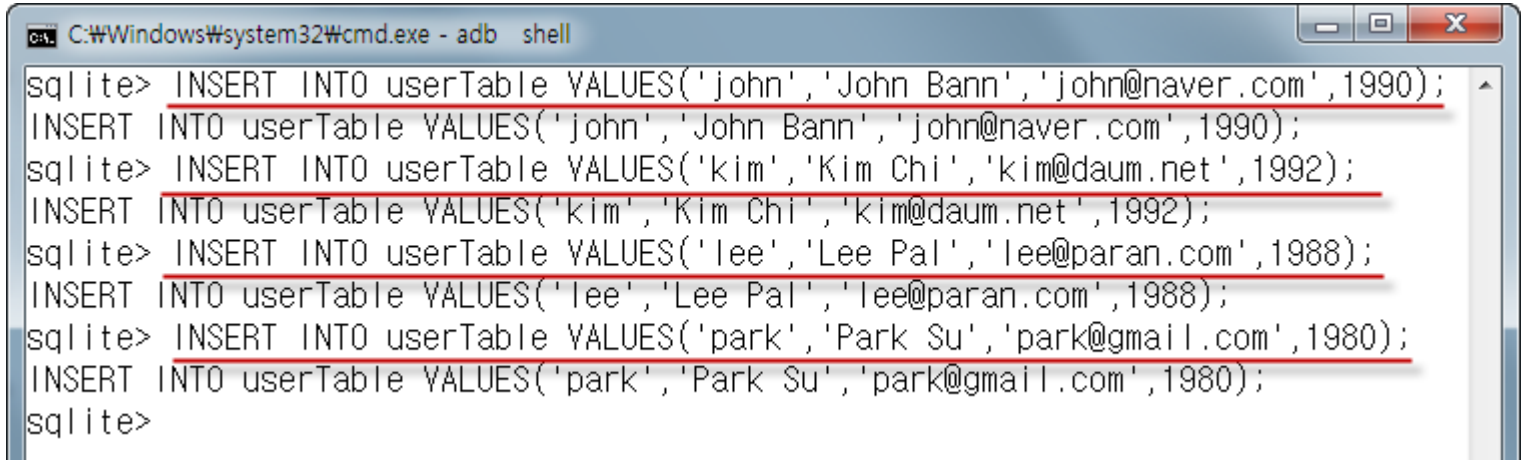




자료 : 한빛미디어 안드로이드 프로그래밍

SQLite 기본

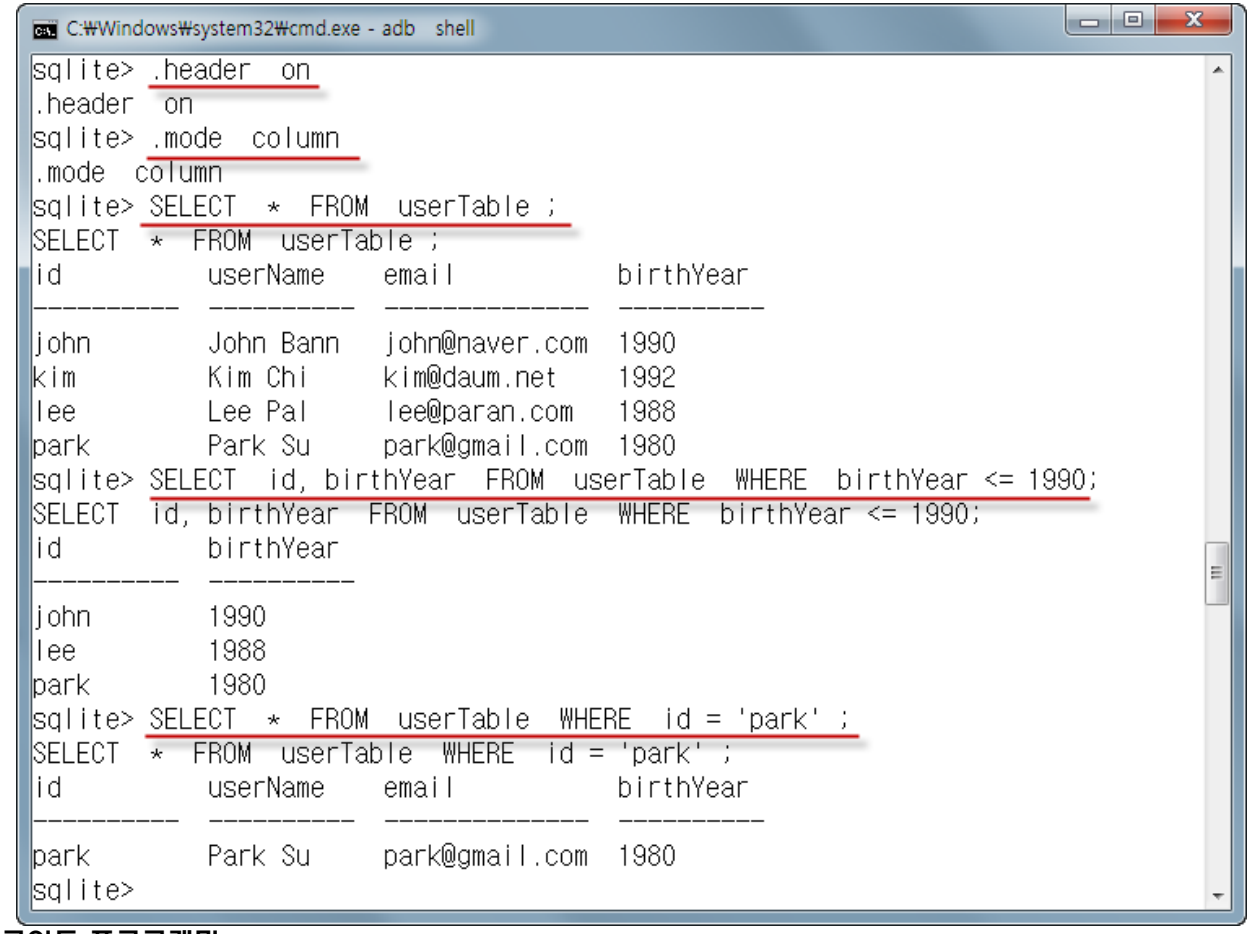
❖ SQLite 직접 사용해보기 : 데이터 입력



자료 : 한빛미디어 안드로이드 프로그래밍

SQLite 기본

❖ SQLite 직접 사용해보기 : 데이터 SELECT

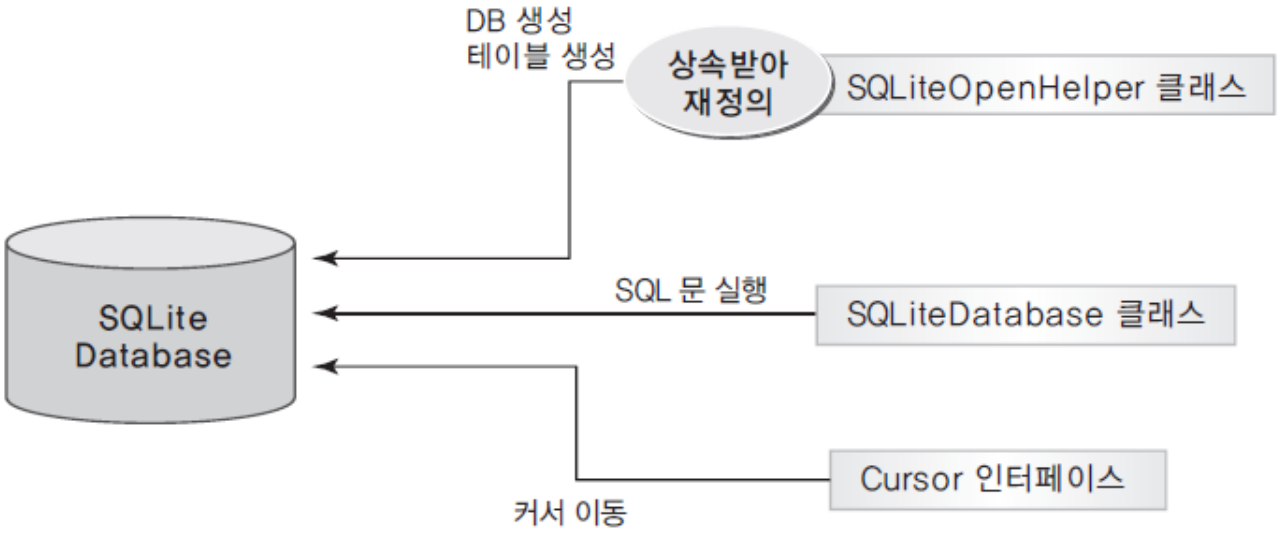


자료 : 한빛미디어 안드로이드 프로그래밍

SQLite 데이터베이스 라이브러리

- **SQLiteDatabase**
  - ✓ 추가 라이브러리(android.database.sqlite.SQLiteDatabase)
  - ✓ 데이터베이스를 다루는 작업(추가, 삭제 , 수정, 질의)를 담당

- **SQLiteOpenHelper**
  - ✓ 추가 라이브러리(android.database.sqlite.SQLiteOpenHelper)
  - ✓ 데이터베이스의 생성, 열기, 업그레이드를 담당



[그림 12-9] SQLite 관련 클래스 동작

SQLite 데이터베이스 라이브러리

[표 12-1] SQLite 관련 클래스 · 인터페이스 및 메소드

클래스 또는 인터페이스	메소드	주 용도
SQLiteOpenHelper 클래스	생성자	DB 생성
	onCreate()	테이블 생성
	onUpgrade()	테이블 삭제 후 다시 생성
	getReadableDatabase()	읽기 전용 DB 열기, SQLiteDatabase 반환
	getWritableDatabase()	읽고 쓰기용 DB 열기, SQLiteDatabase 반환
SQLiteDatabase 클래스	execSQL()	SQL 문(Insert/Update/Delete) 실행
	close()	DB 닫기
	query(), rawQuery()	Select를 실행 후 커서 반환
Cursor 인터페이스	moveToFirst()	커서의 제일 첫 행으로 이동
	moveToLast()	커서의 제일 마지막 행으로 이동
	moveToNext()	현재 커서의 다음 행으로 이동

SQLite 데이터베이스 라이브러리

❖ 데이터베이스 설계 고려사항

- ✓ 파일은 보통 데이터베이스 테이블로 저장하지 않음
  - 대신 문자열을 이용해 해당 파일의 경로를 저장한다.
  - 정규화된 콘텐트 공급자 URI 이용

- ✓ 거의 모든 테이블에 자동증가 키(auto-increment key)필드를 포함시켜, 각행에 대한 인덱스 값으로 기능하도록 하는 것을 적극 권장.

SQLiteDatabase

❖ SQLiteDatabase 이용하여 데이터 베이스 열고 생성하기

- ✓ DDL구문을 사용하기 위해 SQLiteDatabase의 ExecSQL 메소드 호출
- ✓ SQLiteDatabase의 ExecSQL 메소드를 이용한 테이블 생성
  - 예제

```
private static final String DATABASE_NAME = "myDatabase.db";
private static final String TABLE_NAME = "mainTable";

private static final String dbCreateString =
  "create table " + TABLE_NAME + " ( _id integer primary key autoincrement," +
  "column_one text not null);";

SQLiteDatabase myDatabase;

private void createDatabase() {
  myDatabase = openOrCreateDatabase(DATABASE_NAME, Context.MODE_PRIVATE, null);
  myDatabase.execSQL( dbCreateString  );
}
```

데이터베이스 이름 및 테이블명 전역변수 선언

데이터베이스 생성에 대한 쿼리문 전역변수

openOrCreateDatabase를 호출하여 새로운 데이터베이스 생성 후 인스턴스의 exeSQL을 호출하여 SQL명령을 실행한다.

SQLite 데이터베이스

❖ 데이터베이스 행 추가, 업데이트, 제거

- ✓ SQLiteDatabase 클래스
  - 삽입 삭제 업데이트 SQL문을 메서드로 가지고 있음
  - 캡슐화하도록 특화된 메서드를 제공
  - contentValues 사용
    - insert()
    - delete()
    - update()

- ✓ execSQL 메서드
  - 유효한 SQL이라면 어떠한 것이든 데이터베이스 테이블상에 실행가능함

SQLiteDatabase

❖ 데이터 삽입

✓ SQL을 이용한 삽입

```
strSQL = "INSERT INTO member ( c_name, c_alias, c_age )“
            + " VALUES ( 'code_" + Integer.toString( i ) + "'," + " 'test', " + Integer.toString( i + 2 ) + " );";
m_db.execSQL( strSQL );
```

✓ ContentValues를 이용한 삽입

```
// ContentValues 를 이용한 데이터 삽입
ContentValues cvInsert = new ContentValues();
cvInsert.put( "c_name", "neo" );
cvInsert.put( "c_alias", "dreamer" );
cvInsert.put( "c_age", "20" );
m_db.insert( "member", null, cvInsert );
```

SQLiteDatabase

❖ 데이터 검색 : rawQuery와 cursor의 이용

```
// rawQuery 함수를 이용한 데이터 질의
Cursor m_cursor = m_db.rawQuery( "SELECT * FROM member", null );
if ( m_cursor != null )
{
        if ( m_cursor.moveToFirst() )
        {
                String strRow = "--------------------------------\n";
                for(int i = 0 ; i < m_cursor.getColumnCount() ; i++ )
                {
                        strRow += m_cursor.getColumnName(i) + " | ";
                }
                strRow += "\n";
                txtResult.setText( strRow );

                do
                {
                        strRow = "";
                        for(int i = 0 ; i < m_cursor.getColumnCount() ; i++ )
                        {
                                strRow += m_cursor.getString(i) + " | ";
                        }
                        strRow += "\n";
                        txtResult.setText( txtResult.getText() + strRow );
                } while ( m_cursor.moveToNext() );
        }
}
m_cursor.close(); // 커서 닫기
```

SQLiteOpenHelper

❖ 데이터의 삽입, 삭제, 수정

✓ long insert(String table, String nullColumnhack, ContentValues values)

table	DB의 table 이름.
nullcolumnHack	DB에서는 완전히 비어있는 행을 삽입하는 것을 허용하지 않음. 대신 비어있다는 것을 표현하기 위해서 NULL 사용. 따라서 만약 NULL이 들어가야 할 곳에 들어가게 될 것을 말함.
values	생성해 놓은 데이터 맵을 말함

```
mDbHelper = new DatabaseHelper(mCtx);
 mDb = mDbHelper.getWritableDatabase();

 public long createBook(String name, String phone){
    ContentValues initialValues = new ContentValues();
    initialValues.put(KEY_NAME, name);
    initialValues.put(KEY_PHONE, phone);

    return mDb.insert( “book”, null, initialValues);
 }
```

SQLiteOpenHelper

❖ 데이터의 삽입, 삭제, 수정

- ✓ Int delete(String table, String whereClause, Stirng[] whereArgs)

table	DB의 table 이름
whereClause	조건항으로 해당 조건에 맞는 항을 삭제하라는 뜻. Null을 넣으되면 모든 행 삭제.
whereArgs	조건을 규정하는 argument.

```
public boolean deleteBook(long rowID){
    return mDb.delete(DATABASE_TABLE, KEY_ROWID + "=" + rowID, null) > 0;
}
```

SQLiteOpenHelper

❖ 데이터의 삽입, 삭제, 수정

- ✓ Int update(String table, ContentValues values, String whereClause, String[] where Args)

table	DB의 table 이름
values	갱신 내용을 말함
whereClause	조건항으로 해당 조건에 맞는 항을 삭제하라는 뜻. Null을 넣으되면 모든 행 삭제.
whereArgs	조건을 규정하는 argument.

```
public boolean updateBook(long rowID, String name, String phone){
    ContentValues args = new ContentValues();
    args.put(KEY_NAME, name);
    args.put(KEY_PHONE, phone);

    return mDb.update(DATABASE_TABLE, args, KEY_ROWID + "=" + rowID, null) > 0;
}
```

SQLiteDatabase

❖ 데이터 삽입하기

- ✓ ContentValues객체를 이용하여 데이터베이스 테이블에 새로운 행을 삽입가능
- ✓ ContentValues의 값에 대한 열 이름과 맵으로 서 하나의 행을 나타냄
- ✓ ContentValues객체에 데이터베이스 테이블에 맞는 자료를 입력 후 SQLiteDatabase클래스의 insert()메소드를 사용하여 새로운 레코드 추가

- ✓ 예제

```
ContentValues values = new ContentValues();
values.put("country_name", "US");
long countryId = db.insert("tbl_countries", null, values);

ContentValues stateValues = new ContentValues();
stateValues.put("state_name", "Texas");
stateValues.put("country_id", Long.toString(countryId));
try {
    db.insertOrThrow("tbl_states", null, stateValues);
} catch (Exception e) {
    //catch code
}
```

ContentValues객체에
put메소드를 사용하여 데이터 입력

Insert 메소드를 사용하여
ContentValues객체 데이터 입력

insertOrThrow : SQLiteException Throw
insertWithOnConflict : No Throw
insert : 내부적으로 insertWithOnConflict

SQLiteDatabase

❖ 테이블 update 하기

```
ContentValues updateCountry = new ContentValues();
updateCountry.put("country_name", "United States");
db.update("tbl_countries", updateCountry, "id=?", new String[] {Long.toString(countryId)});
```

❖ 테이블에서 ROW 삭제하기

```
db.delete("tbl_states", "id=?", new String[] {Long.toString(countryId)});
```

데이터베이스 쿼리(질의)

❖ Query(질의)

- ✓ 질의를 통해 데이터베이스를 접근
- ✓ 질의 결과는 Cursor객체 형태로 반환
- ✓ 메소드 형태

  public Cursor query (String table, String[] columns, String selection, String[] selectionArgs, String groupBy, String having, String orderBy, String limit)

  - Table - 질의를 수행할 테이블 이름입니다.
  - columns - 자료를 받아올 필드들입니다. null을 입력하면 모든 필드를 반환합니다.
  - selection - SQL의 "where" 구문에 해당되는 조건을 입력합니다. 조건이 많을 경우, ?로 대체합니다.
  - selectionArgs - selection을 ?로 지정하였을 경우, 그 조건들을 입력합니다.
  - groupBy - SQL의 "group by"구문에 해당합니다.
  - Having - groupBy를 지정했을 경우, 그 조건을 넣어줍니다.
  - orderBy - 결과값 정렬 방식을 지정합니다. null을 입력하면 기본 정렬을 수행합니다.
  - limit 결과값의 개수를 제한합니다.

- ✓ 예제(위 메소드와 다른형태)

```
// 모든 레코드를 반환하는 쿼리를 실행합니다.
Cursor all = myDB.query("data", null, null, null, null, null, null, null);
// 이름이 google인 레코드를 반환하는 쿼리를 실행합니다.
Cursor sel = myDB.query("data", "name = google", null, null, null, null, null, null);
```

Cursor와 ContentValues

❖ Cursor

- ✓ 질의(Queries)결과를 Cursor객체로 반환 받음
- ✓ Cursor객체는 결과 값의 사본이 아닌 실제 데이터(레코드)를 가리키는 역활
- ✓ Cursor 클래스는 질의 결과 탐색을 위한 여러 함수를 포함함
  - moveToFirst – 커서를 질의 결과 내의 첫번째 행으로 옮긴다.
  - moveToNext – 커서를 다음 행으로 옮긴다.
  - moveToPrevious – 커서를 이전 행으로 옮긴다.
  - getCount – 결과셋에 있는 행의 수를 리턴
  - getCloumnIndexOrThrow – 지정된 이름을 가진 열에 대한 인덱스를 리턴
  - getCloumnName – 지정된 열 인덱스의 이름을 리턴
  - getColumnNames – 현재커서에 있는 모든 열 이름을 가진 String 배열 하나를 리턴
  - moveToPosition – 현재 커서 위치를 리턴
  - getPosition – 커서가 현재 가리키고 있는 위치를 반환
  - get<데이터타입>(필드 인덱스) – 커서에서 데이터를 받아올 때 사용

    예제) 레코드로부터 이름을 받아옵니다.

    • String name = result.getString(1);

    • 다른 데이터형식 http://developer.android.com/reference/android/database/Cursor.html

SQLite 데이터베이스

❖ 데이터베이스 질의하기

✓ 일반적인 질의 예제)

```
String[] result_columns = new String[] {KEY_ID, KEY_COL1, KEY_COL3};

Cursor allRows = myDatabase.query(true, DATABASE_TABLE, result_columns,
                                  null, null, null, null, null, null);

// Return all columns for rows where column 3 equals a set value
// and the rows are ordered by column 5.
String where = KEY_COL3 + "=" + requiredValue;
String order = KEY_COL5;
Cursor myResult = myDatabase.query(DATABASE_TABLE, null, where,
                                   null, null, null, order);
```

모든 행의 1열과 3열을 중복 없이 리턴 하는 질의

3열의 값과 설정된 값과 같은 행의 모든 열을 리턴하고 행들을 5열 기준으로 정렬하는 질의

SQLite 데이터베이스

❖ 데이터베이스 질의하기

✓ 커서에서 결과 얻어오는 예제)

```
int GOLD_HOARDED_COLUMN = 2;

Cursor myGold = myDatabase.query("GoldHoards", null, null, null, null, null, null);

float totalHoard = 0f;

// Make sure there is at least one row.
if (myGold.moveToFirst()) {
  // Iterate over each cursor.
  do {
    float hoard = myGold.getFloat(GOLD_HOARDED_COLUMN);
    totalHoard += hoard;
  } while(myGold.moveToNext());
}

Float averageHoard = totalHoard / myGold.getCount();
```

GlodHoards란 문자를 가진 행들을 커서에서 지원한 메소드를 통해 커서를 이동하며 커서가 위치한 데이터를 안전하게 가져와 부동소수열을 추출하여 더하는 작업을 검색된 수만큼 반복하고 있다.

SQLite 데이터베이스

❖ 새로운 행 삽입하기

✓ 예제)

```
// Create a new row of values to insert.
ContentValues newValues = new ContentValues();

// Assign values for each row.
newValues.put(COLUMN_NAME, newValue);
[ ... 각 열마다 반복... ]

// Insert the row into your table
myDatabase.insert(DATABASE_TABLE, null, newValues);
```

1. contentValues객체를 통해 삽입할 새로운 행 하나를 생성
2. 각 열에 값을 할당
3. 값을 할당 받은 행을 테이블에 삽입

SQLite 데이터베이스

❖ 데이터베이스에 있는 행 업데이트 하기

✓ 예제)

```
// Define the updated row content.
ContentValues newValues = new ContentValues();

// Assign values for each row.
newValues.put(COLUMN_NAME, newValue);
[ ... 각 열마다 반복... ]

String where = KEY_ID + "=" + rowId;

// Update the row with the specified index with the new values.
myDatabase.update(DATABASE_TABLE, newValues, where, null);
```

1. contentValues객체를 통해 업데이드할 행 하나를 생성
2. 각 열에 값을 할당
3. 업데이트할 행 인덱스 값 지정
4. 지정된 인덱스의 행을 새로운 값으로 업데이트

❖ 행 삭제하기

✓ 예제)

```
myDatabase.delete(DATABASE_TABLE, KEY_ID + "=" + rowId, null);
```

1. 테이블 이름과 지우고자 하는 행을 리턴하는 where절을 지정하여 삭제

SQLite 데이터베이스

❖ 트랜잭션 지원

```
db.beginTransaction();
        Cursor cur = null;
        try {
          cur = db.query("tbl_countries",
                        null, null, null, null, null, null);
          cur.moveToPosition(0);
          ContentValues values = new ContentValues();
          values.put("state_name", "Georgia");
          values.put("country_id", cur.getString(0));
          long stateId = db.insert("tbl_states", null, values);
          db.setTransactionSuccessful();
          view.append("n" + Long.toString(stateId));
        } catch (Exception e) {
          Log.e("Error in transaction", e.toString());
        } finally {
          db.endTransaction();
          cur.close();
        }
```

여기까지는 성공적이었음을 mark함

SQLite 데이터베이스

❖ 예 : DB insert와 query

```
public class DBTest extends Activity {

public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        SQLiteDatabase db = null;
        try {
                db = db.openOrCreateDatabase("sample.db",SQLiteDatabase.CREATE_IF_NECESSARY, null);
                        db.execSQL("CREATE TABLE IF NOT EXISTS fruit“
                                + " (id integer primary key autoincrement, name varchar(30));");
                db.execSQL("insert into fruit(name) values('grape');");
                db.execSQL("insert into fruit(name) values('apple');");
        } catch (Exception e) {
                // TODO: handle exception
                 Log.e("ERROR","ERROR IN CODE:"+e.toString());
        }
        TextView tv = (TextView) findViewById(R.id.TextView01);
        Cursor c = db.query("fruit", new String[] {"name"}, null, null, null, null, null);
        int rowcount = c.getCount();
        c.moveToFirst();
        String str = "";
        for (int i = 0; i < rowcount; i++) {
                str += c.getString(0)+"\n";
                c.moveToNext();
        }
        tv.setText(str);
        if (db != null) {
                db.close();
        }
}

}
```

SQLiteOpenHelper

- **SQLiteOpenHelper**
  - ✓ DB를 생성 및 오픈하는 처리를 담당
  - ✓ onCreate , onUpgrade 등의 메서드에 @Overide 하면 나머지를 담당

- **주요 메소드**
  - ✓ SQLiteOpenHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, int version)

  - ✓ Overide 메소드

onCreate	**DB가 처음 만들어질 때 호출된다 여기서 테이블을 만들고 초기 레코드를 삽입(CREATE TABLE)**
onUpgrade	DB를 업그레이드할 때 호출된다. 기존 테이블을 삭제하고 새로 만들거나 ALTER TABLE로 스키마를 수정한다.
onOpen	DB를 열 때 호출된다.

SQLiteOpenHelper

❖ 간단한 예제

```
public class DbAdapter {
    private DatabaseHelper mDbHelper;
    private SQLiteDatabase mDb; // 데이터베이스를 저장

    private static final String DATABASE_CREATE =
        "create table data (_id integer primary key autoincrement,"+
        "name text not null, phone text not null);";

    private static final String DATABASE_NAME = "datum.db";
    private static final String DATABASE_TABLE = "data";
    private static final int DATABASE_VERSION = 1;

    private final Context mCtx;

    private class DatabaseHelper extends SQLiteOpenHelper{

        public DatabaseHelper(Context context) {
            super(context, DATABASE_NAME, null, DATABASE_VERSION);
            // TODO Auto-generated constructor stub
        }

        public void onCreate(SQLiteDatabase db){
            db.execSQL(DATABASE_CREATE);
        }

        public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion){
            Log.w(TAG, "Upgrading db from version" + oldVersion + " to" +
                    newVersion + ", which will destroy all old data");
            db.execSQL("DROP TABLE IF EXISTS data");
            onCreate(db);
        }
    }
```

SQLiteOpenHelper

❖ **SQLiteOpenHelper의 주요 메소드**

getReadableDatabase	**읽기 위해 DB 를 연다. 이 단계에서 DB 가 존재하지 않으면 onCreate가 호출되며 버전이 바뀌었으면 onUpgrade 가 호출된다.**
getWritableDatabase	읽고 쓰기 위해 DB 를 연다. 권한이 없다거나 디스크가 가득차면 실패한다.
close	DB 를 닫는다.

SQLiteOpenHelper

❖ SQLiteOpenHelper확장하기 – 상속 및 데이터 베이스 생성

- ✓ SQLiteOpenHelper객체를 사용하는게 가장 쉬운 일반적인 생성 방법
  - SQLiteOpenHelper클래스를 사용할 인스턴스 생성
  - 읽기/쓰기 여부에 따라 getReadableDatabase()/getWriteableDatabase() 메소드를 호출
  - 예제

```
private static class myDBHelper extends SQLIteOpenHelper {
        public myDBHelper(Context context, String name,
                                        CursorFactory factory, imt version) {
        super(context, name, factory, version);
        }
……./*생성 및 업그레이드에 대한 내용*/……..
 }

dbHelper = new myDBHelper(context, DATABASE_NAME, null, DATABASEVERSION);
SQLiterDatabase db;
try {
        db = dbHelper.getWriteableDatabase();
}
Catch(SQLiteException ex){
        db = dbHelper.getReadableDatabase();
}
```

데이터베이스의 쓰기 가능한 인스턴스 얻기

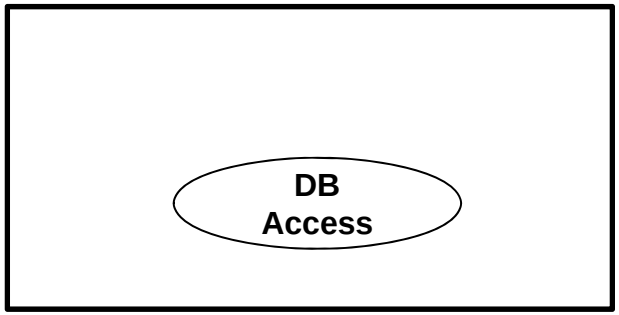
데이터베이스의 읽기 가능한 인스턴스 얻기

DB Adapter 형태로 만들어보기

❖ 전화번호부 DB Adapter 만들어보기

✓ 메소드 설계

- Class PhoneDBAdapter
  - open()
  - close()
  - public long createBook(String name, String phone)
  - public boolean deleteBook(long rowID)
  - public Cursor fetchAllBooks()
  - public Cursor fetchBook(long rowID)
  - public boolean updateBook(long rowID, String name, String phone)



Adapter
(Application Logic)

DB Adapter 형태로 만들어보기

❖ 전화번호부 DBAdapter 구현

```
public class DbAdapter {

    public static final String KEY_NAME = "name";
    public static final String KEY_PHONE = "phone";
    public static final String KEY_ROWID = "_id";

    public static final int FIND_BY_NAME = 0;
    public static final int FIND_BY_PHONE = 1;

    private static final String TAG = "DbAdapter";
    private DatabaseHelper mDbHelper;
    private SQLiteDatabase mDb; // 데이터베이스를 저장

    private static final String DATABASE_CREATE =
        "create table data (_id integer primary key autoincrement,"+
        "name text not null, phone text not null);";

    private static final String DATABASE_NAME = "datum.db";
    private static final String DATABASE_TABLE = "data";
    private static final int DATABASE_VERSION = 1;

    private final Context mCtx;

    private class DatabaseHelper extends SQLiteOpenHelper{

        public DatabaseHelper(Context context) {
            super(context, DATABASE_NAME, null, DATABASE_VERSION);
            // TODO Auto-generated constructor stub
        }

        public void onCreate(SQLiteDatabase db){
            db.execSQL(DATABASE_CREATE);
        }

        public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)
            Log.w(TAG, "Upgrading db from version" + oldVersion + " to" +
                    newVersion + ", which will destroy all old data");
            db.execSQL("DROP TABLE IF EXISTS data");
            onCreate(db);
        }

    }
```

```
    public DbAdapter(Context ctx){
        this.mCtx = ctx;
    }

    public DbAdapter open() throws SQLException{
        mDbHelper = new DatabaseHelper(mCtx);
        mDb = mDbHelper.getWritableDatabase();
        return this;
    }

    public void close(){
        mDbHelper.close();
    }

    public long createBook(String name, String phone){
        ContentValues initialValues = new ContentValues();
        initialValues.put(KEY_NAME, name);
        initialValues.put(KEY_PHONE, phone);

        return mDb.insert(DATABASE_TABLE, null, initialValues);
    }

    public boolean deleteBook(long rowID){
        return mDb.delete(DATABASE_TABLE, KEY_ROWID + "=" + rowID, null) > 0;
    }
    public Cursor fetchBook(long rowID) throws SQLException{
        Cursor mCursor =
            mDb.query(true, DATABASE_TABLE,
               new String[]{KEY_ROWID, KEY_NAME, KEY_PHONE},
               KEY_ROWID + "=" + rowID, null, null, null, null, null);
        if(mCursor != null)
            mCursor.moveToFirst();
        return mCursor;
    }

    public boolean updateBook(long rowID, String name, String phone){
        ContentValues args = new ContentValues();
        args.put(KEY_NAME, name);
        args.put(KEY_PHONE, phone);

        return mDb.update(DATABASE_TABLE, args, KEY_ROWID + "=" + rowID, null) > 0;
    }
```

SQLite 데이터베이스

❖ 예 : Android폰의 Bookmark 보기

```
import android.app.Activity;
import android.os.Bundle;
import android.provider.Browser;
import android.widget.TextView;
import android.database.Cursor;

public class TestingData extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        TextView view = (TextView) findViewById(R.id.hello);
        String[] projection = new String[] {
                                Browser.BookmarkColumns.TITLE
                                , Browser.BookmarkColumns.URL
        };
        Cursor mCur = managedQuery(android.provider.Browser.BOOKMARKS_URI,
                                projection, null, null, null
                                );
        mCur.moveToFirst();
        int titleIdx = mCur.getColumnIndex(Browser.BookmarkColumns.TITLE);
        int urlIdx = mCur.getColumnIndex(Browser.BookmarkColumns.URL);
        while (mCur.isAfterLast() == false) {
            view.append("n" + mCur.getString(titleIdx));
            view.append("n" + mCur.getString(urlIdx));
            mCur.moveToNext();
        }

    }
}
```

SQLite 데이터베이스

❖ 예 : Mediaplayer 파일 정보 보기

```
import android.app.Activity;
import android.os.Bundle;
import android.provider.MediaStore;
import android.provider.MediaStore.Audio.Media;
import android.widget.TextView;
import android.database.Cursor;

public class TestingData extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        TextView view = (TextView) findViewById(R.id.hello);
        String[] projection = new String[] {
                                    MediaStore.MediaColumns.DISPLAY_NAME
                                    , MediaStore.MediaColumns.DATE_ADDED
                                    , MediaStore.MediaColumns.MIME_TYPE
        };
        Cursor mCur = managedQuery(Media.EXTERNAL_CONTENT_URI,
                                    projection, null, null, null
                                    );
        mCur.moveToFirst();

        while (mCur.isAfterLast() == false) {
                for (int i=0; i<mCur.getColumnCount(); i++) {
            view.append("n" + mCur.getString(i));
            }
            mCur.moveToNext();
        }

    }
}
```