앱 프로그래밍

Chapter 11

단말에 데이터베이스와 내용 제공자 만들기

이번 장에서는 무엇을 다룰까요?

앱의 데이터를 잘 관리하려면 어떻게 해야 할까요?

- 단말에 저장하는 데이터베이스에 대해 알아볼까요?
- 데이터베이스와 테이블을 만드는 방법을 알아볼까요?
- 데이터베이스에서 데이터를 조회하는 방법을 알아볼까요?
- 내용제공자는 어떻게 만들 수 있는지 알아볼까요?
- 앨범과 연락처에 들어있는 데이터는 어떻게 조회하는지 알아볼까요?

이번 장에서는 무엇을 다룰까요?

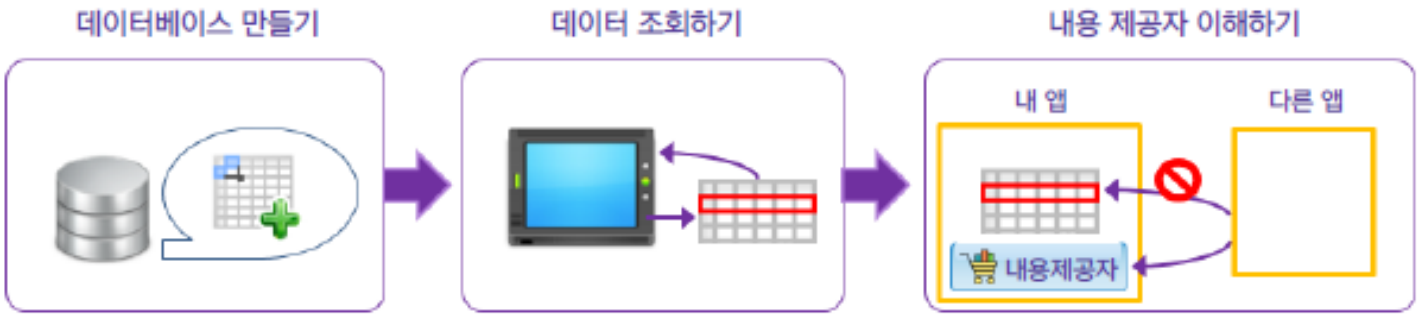
스마트폰에서도 데이터베이스를 사용할 수 있나요?
- 모바일 데이터베이스

▼

데이터베이스는 어떻게 만들면 되나요?
- 데이터베이스 만들기
- 테이블 만들기

▼

테이블의 데이터는 어떻게 가져올 수 있나요?
- 데이터 조회하기



강의 주제 및 목차

강의 주제

데이터베이스에 대한 이해와 데이터 저장 및 조회 실습

1. 모바일 데이터베이스란?

모바일 데이터베이스란?

- 안드로이드에서 데이터를 저장하는 대표적인 방법
  - 설정 정보
  - 파일 사용
  - 데이터베이스 → 많은 데이터를 체계적으로 관리

[데이터베이스 활용 순서]

- 데이터베이스
  - 여러 개의 테이블을 담고 있는 하나의 그릇 역할

- 데이터베이스를 만드는 가장 간단한 방법
  - Context 클래스에 정의된 openOrCreateDatabase() 메소드를 사용
  - 애플리케이션에서 기본적으로 사용하는 Activity 클래스가 Context를 상속한 것이므로 액티비티 안에서 데이터베이스 생성 가능

2. 데이터베이스와 테이블 만들기

데이터베이스와 테이블 만들기

■ 데이터베이스를 열거나 삭제할 수 있는 메소드

```
public abstract SQLiteDatabase openOrCreateDatabase (String name, int mode,
        SQLiteDatabase.CursorFactory factory)
public abstract boolean deleteDatabase (String name)
```

■ SQL을 실행할 수 있는 메소드

- create, insert, delete 등 결과데이터가 없는 SQL문

```
public void execSQL(String sql) throws SQLException
```

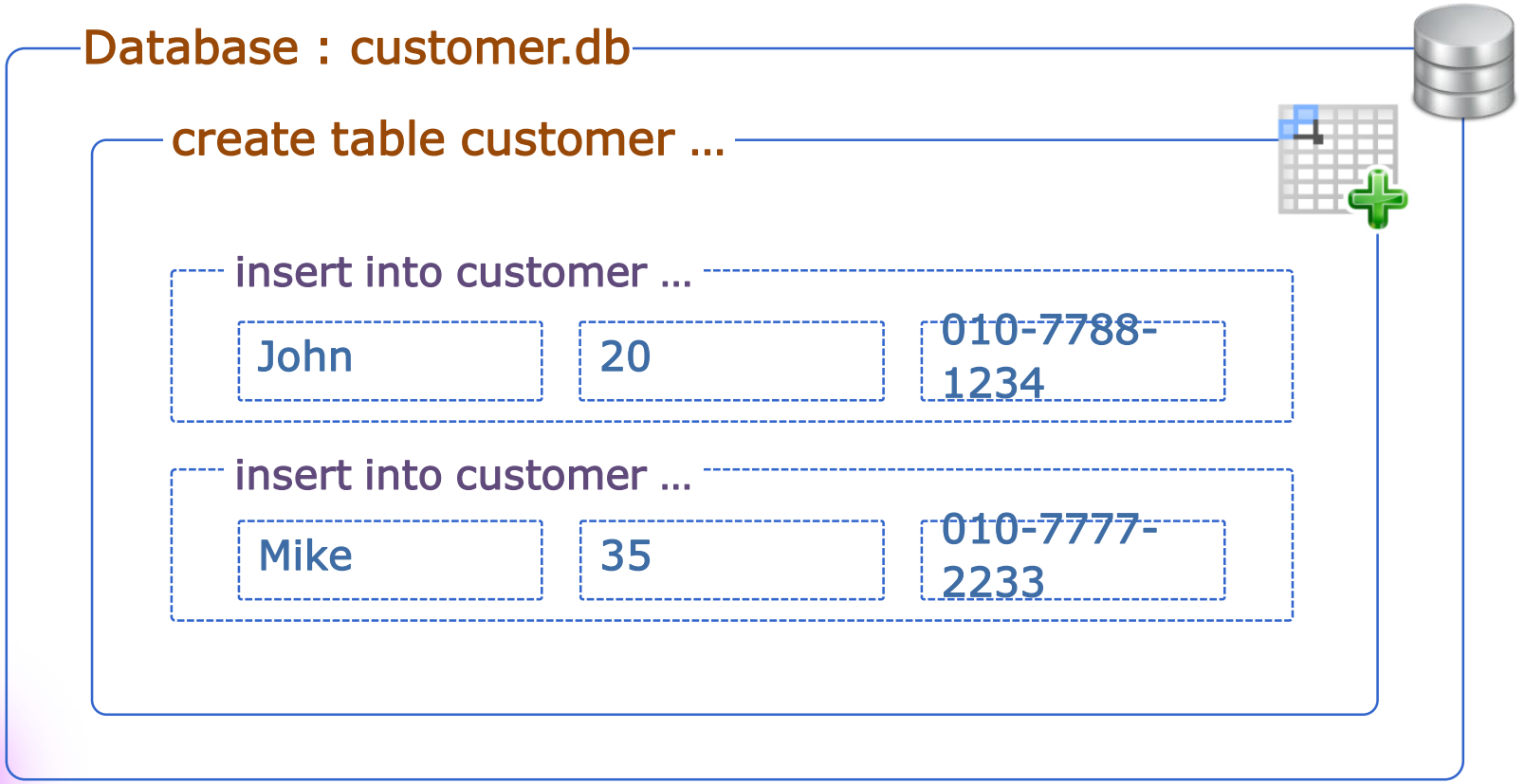
- select 와 같이 조회에 따른 결과 데이터가 있는 SQL문

```
public Cursor rawQuery(String sql) throws SQLException
```

데이터베이스 만들기 구조

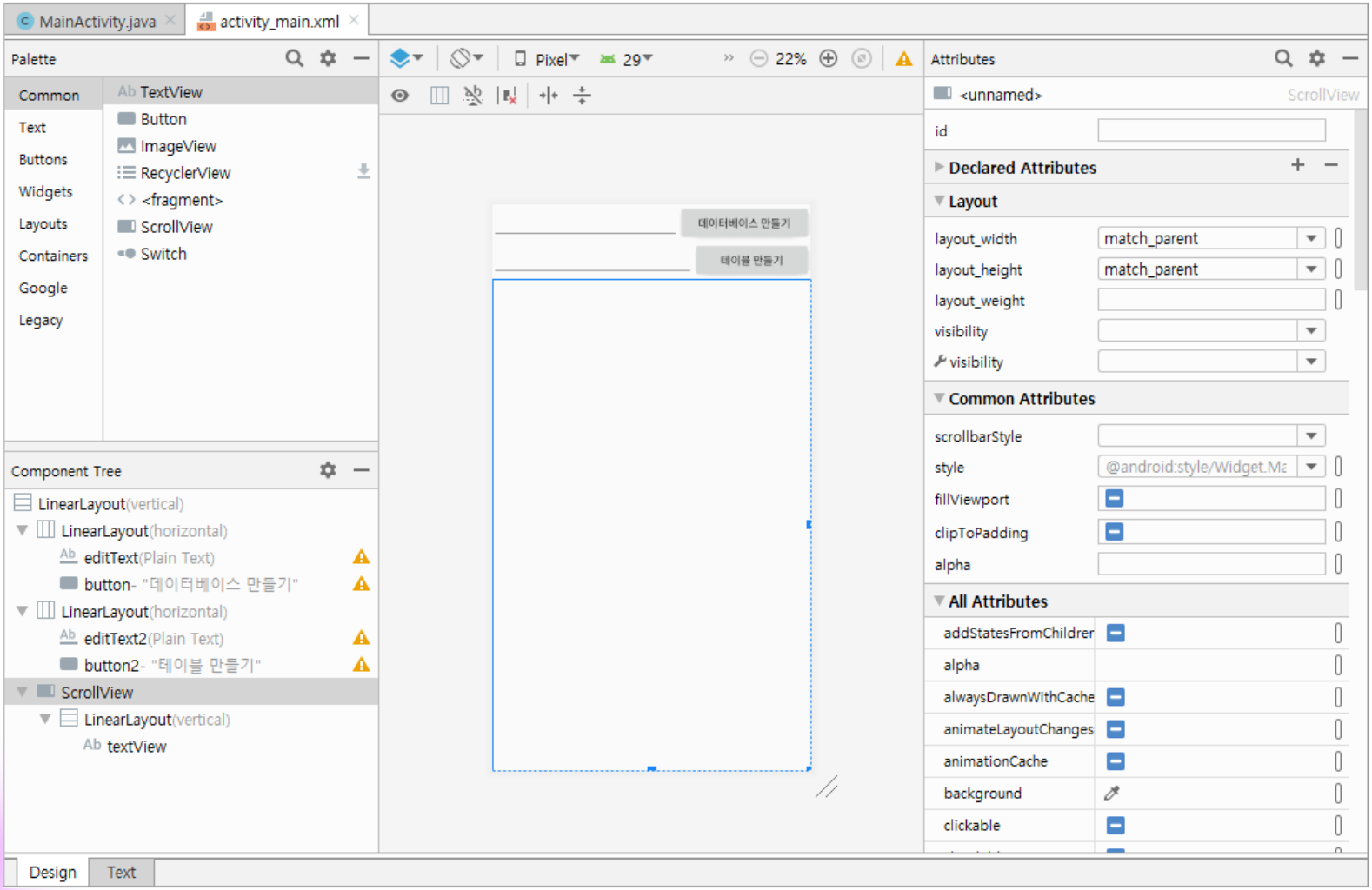
▶ 1단계 : 데이터베이스 생성 → 2단계 : 테이블 생성 → 3단계 : 레코드 추가

▶ 테이블 생성과 레코드 추가는 SQL문을 만들어 실행 (create table … & insert into …)



XML 레이아웃 만들기

▶ SampleDatabase 프로젝트 만들고 activity_main.xml 파일 열어 화면 구성하기



메인 액티비티 코드 만들기 (계속)

```
private void createDatabase(String name) {
  println("createDatabase 호출됨.");

  database = openOrCreateDatabase(name, MODE_PRIVATE, null);

  println("데이터베이스 생성함: " + name);
}
private void createTable(String name) {
  println("createTable 호출됨.");

  if (database == null) {
    println("데이터베이스를 먼저 생성하세요.");
    return;
  }

  database.execSQL("create table if not exists " + name + "("
              + " _id integer PRIMARY KEY autoincrement, "
              + " name text, "
              + " age integer, "
              + " mobile text)");

              println("테이블 생성함: " + name);
}
```

❶ 데이터베이스를 만들기 위한 메서드 실행하기

❷ 테이블을 만들기 위한 SQL문 실행하기

메인 액티비티 코드 만들기 (계속)

```
private void insertRecord() {

  println("insertRecord 호출됨.");
  if (database == null) {
    println("데이터베이스를 먼저 생성하세요.");
    return;
  }

  if (tableName == null) {
    println("테이블을 먼저 생성하세요.");
    return;
  }

  database.execSQL("insert into " + tableName
            + "(name, age, mobile) "
            + " values "
            + "( ' John ' , 20,  ' 010-1000-1000 ' )");

  println("레코드 추가함.");
}
```

칼럼 참조용 데이터 타입

칼럼 타입	설 명
text, varchar	문자열
smallint, integer	정수 (2바이트 또는 4바이트)
real, float, double	부동소수 (4바이트 또는 8바이트)
boolean	true 또는 false
date, time, timestamp	시간 (날짜, 시간, 날짜+시간)
blob, binary	바이너리

[표] SQLite에서 지원하는 칼럼 타입

테이블 생성과 레코드 추가를 위한 SQL 문법

■ 테이블을 만들기 위한 SQL문

```
CREATE TABLE [IF NOT EXISTS] table_name(col_name column_definition, ...)
                [table_option] ...
```

■ 레코드를 추가하기 위한 SQL문

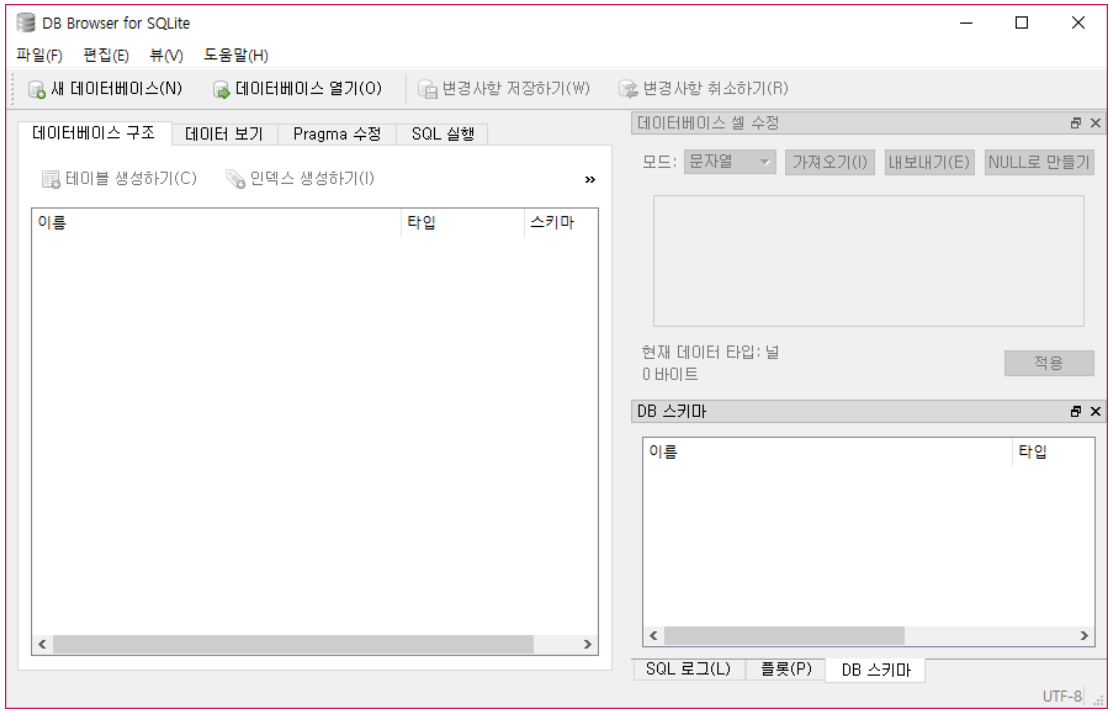
```
INSERT INTO table_name<(column list)> VALUES (value, ...)
```

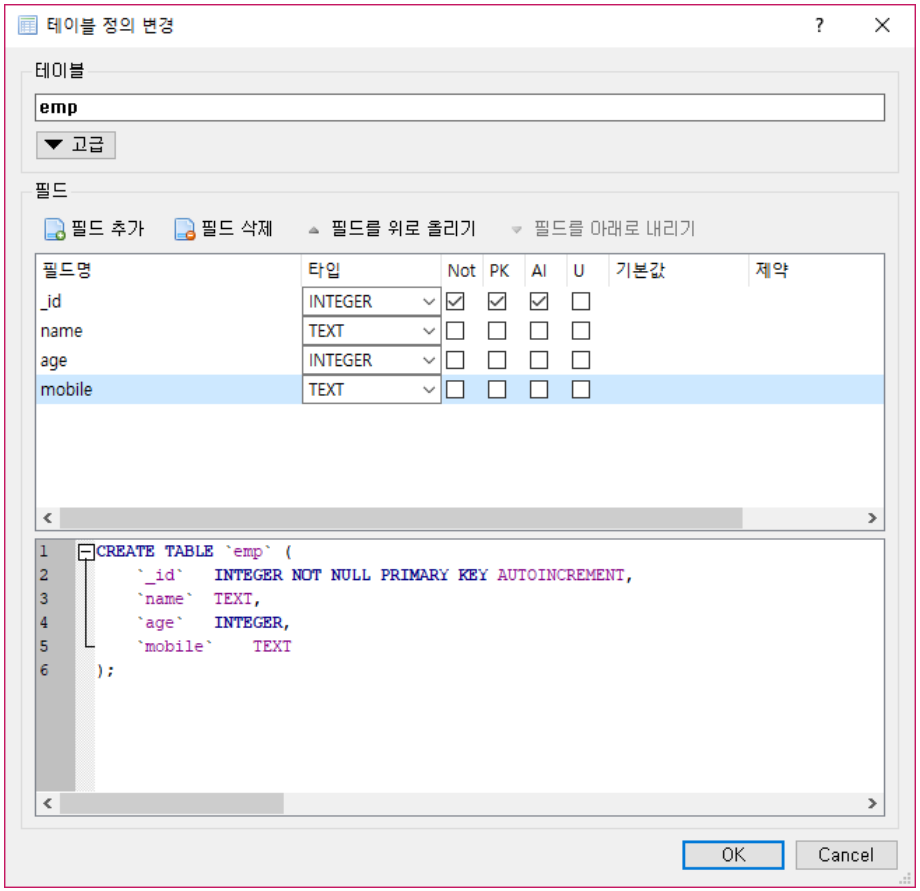
실행 화면

데이터베이스 생성, 테이블 생성 그리고 레코드 추가

데이터베이스 관리도구

http://sqlitebrowser.org/ 참조





3. 헬퍼 클래스로 업그레이드 지원하기

헬퍼 클래스를 이용해 업그레이드 지원하기

■ SQLiteOpenHelper 클래스

- 데이터베이스를 만들거나 열기 위해 필요한 일들을 도와주는 역할을 함

■ SQLiteOpenHelper 클래스

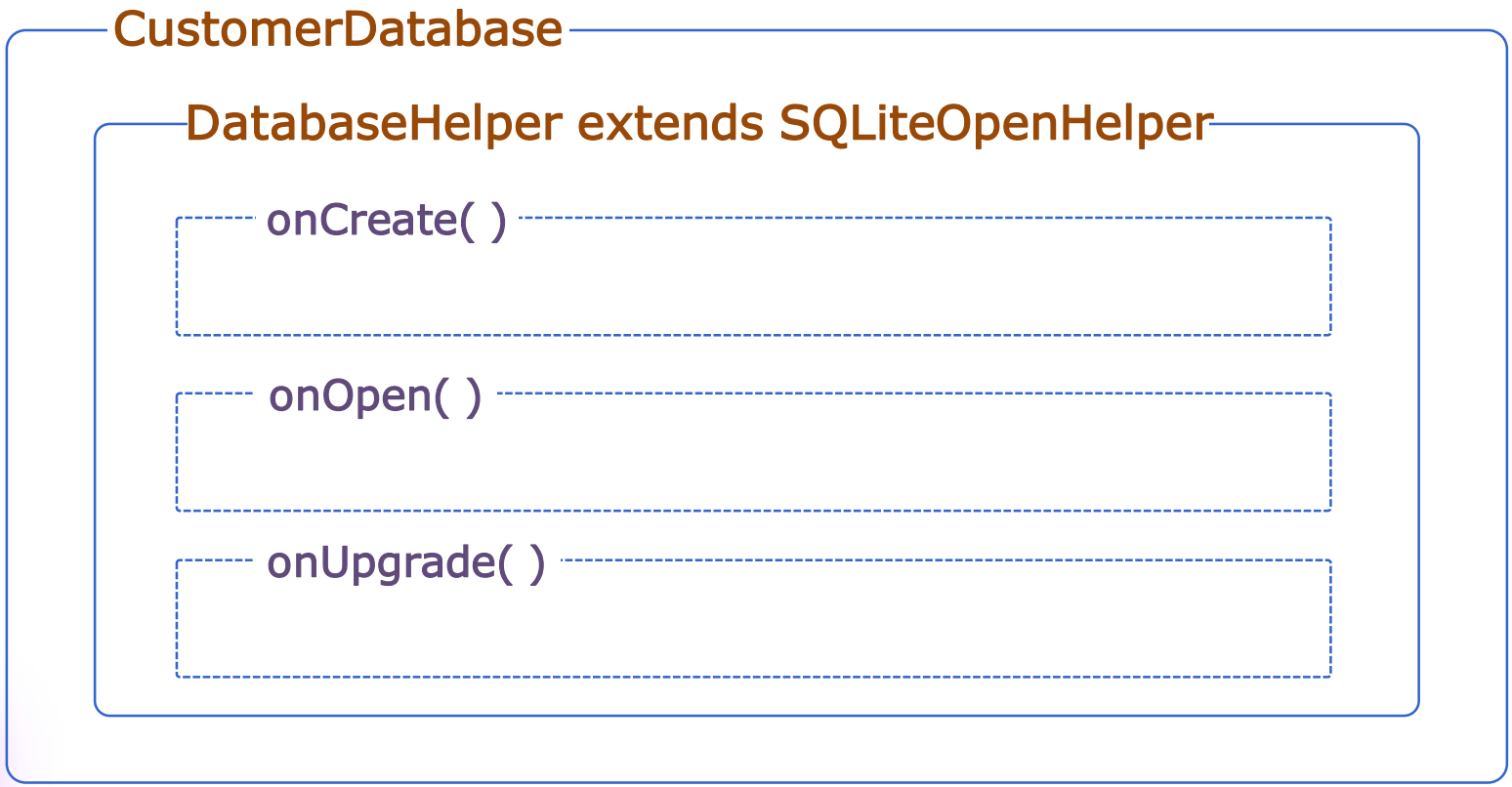
```
public SQLiteOpenHelper (Context context, String name,
                         SQLiteDatabase.CursorFactory factory, int version)

public abstract void onCreate (SQLiteDatabase db)
public abstract void onOpen (SQLiteDatabase db)
public abstract void onUpgrade (SQLiteDatabase db, int oldVersion, int
newVersion)
```

db만들 때. 만들어져있을시 ㄴㄴ

헬퍼클래스의 구조

▶ 새로 만드는 CustomerDatabase 클래스는 DatabaseHelper 객체와 버전 정보 관리

▶ Helper 클래스를 상속한 DatabaseHelper 클래스 안에서는 처음 데이터베이스가 만들어질 때는 onCreate(), 버전이 바뀌어 업그레이드될 때는 onUpgrade() 메소드가 호출됨



헬퍼클래스 만들기

참조파일 SampleDatabase2>/app/java/org.techtown.database/DatabaseHelper.java

```
public class DatabaseHelper extends SQLiteOpenHelper {
  public static String NAME = "employee.db";
  public static int VERSION = 1;

  public DatabaseHelper(Context context) {
    super(context, NAME, null, VERSION);
  }

  public void onCreate(SQLiteDatabase db) {
    println("onCreate 호출됨");

    String sql = "create table if not exists emp("
                 + " _id integer PRIMARY KEY autoincrement, "
                 + " name text, "
                 + " age integer, "
                 + " mobile text)";

    db.execSQL(sql);
  }
```

❶ SQLiteOpenHelper 클래스를 상속하여 새로운 클래스 정의하기

❷ onCreate() 메서드 안에서 SQL문 실행하기

헬퍼클래스 만들기 (계속)

```
public void onOpen(SQLiteDatabase db) {
  println("onOpen 호출됨");
}

public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
  println("onUpgrade 호출됨: " + oldVersion + " -> " + newVersion);

  if (newVersion > 1) {
    db.execSQL("DROP TABLE IF EXISTS emp");
  }
}
```

헬퍼클래스 사용

참조파일 SampleDatabase2>/app/java/org.techtown.database/MainActivity.java

```
public class MainActivity extends AppCompatActivity {
  EditText editText;
  EditText editText2;
  TextView textView;

  DatabaseHelper dbHelper;
  SQLiteDatabase database;
  String tableName;

  중략…

   private void createDatabase(String name) {
    println("createDatabase 호출됨.");

    dbHelper = new DatabaseHelper(this);
    database = dbHelper.getWritableDatabase();

    println("데이터베이스 생성함: " + name);
  }

중략…
```

DatabaseHelper 객체 생성하고
SQLiteDatabase 객체 참조하기

4. 데이터 조회하기

데이터 조회하기

데이터 조회하기 예제

-데이터베이스에서 SQL로 데이터 조회하기

메인 액티비티의 XML 레이아웃 정의

-메인 액티비티 레이아웃 정의

메인 액티비티 코드 작성

-커서를 이용해 데이터 조회하기

메인 액티비티 만들기

```
public void executeQuery() {
    println("executeQuery 호출됨.");

    Cursor cursor = database.rawQuery("select _id, name, age, mobile from emp", null);
    int recordCount = cursor.getCount();
    println("레코드 개수: " + recordCount);

    for (int i = 0; i < recordCount; i++) {
      cursor.moveToNext();

      int id = cursor.getInt(0);
      String name = cursor.getString(1);
      int age = cursor.getInt(2);
      String mobile = cursor.getString(3);

      println("레코드#" + i + " : " + id + ", " + name + ", " + age + ", " + mobile);
    }
    cursor.close();
  }
```

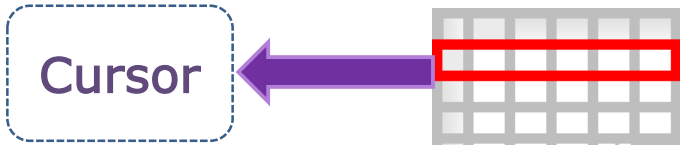
❶ SQL 실행하고 Cursor 객체 반환받기

❷ 다음 결과 레코드로 넘어가기

중략…

커서의 메소드

- 데이터베이스 조회를 위해 SELECT문 실행
- 결과값으로 Cursor 객체 리턴
- getCount() 메소드로 레코드 개수 확인
- moveToNext() 메소드로 하나씩 진행
- getXXX() 메소드로 값 확인



rawQuery("select ...")

```
public abstract int getColumnCount ()
public abstract int getColumnIndex (String columnName)
public abstract String getColumnName (int columnIndex)
public abstract String[] getColumnNames ()
public abstract int getCount ()
public abstract boolean moveToNext ()
public abstract boolean moveToPrevious ()
public abstract boolean moveToFirst ()
public abstract boolean moveToLast ()
public abstract boolean move (int offset)
public abstract String getString (int columnIndex)
public abstract short getShort (int columnIndex)
public abstract int getInt (int columnIndex)
public abstract long getLong (int columnIndex)
public abstract float getFloat (int columnIndex)
public abstract double getDouble (int columnIndex)
public abstract byte[] getBlob (int columnIndex)
```

데이터 조회 – SELECT SQL

SELECT [* | DISTINCT] column_name [,columnname2]

FROM tablename1 [,tablename2]

WHERE [condition and|or condition...]

[GROUP BY column-list]

[HAVING conditions]

[ORDER BY "column-list" [ASC | DESC]]

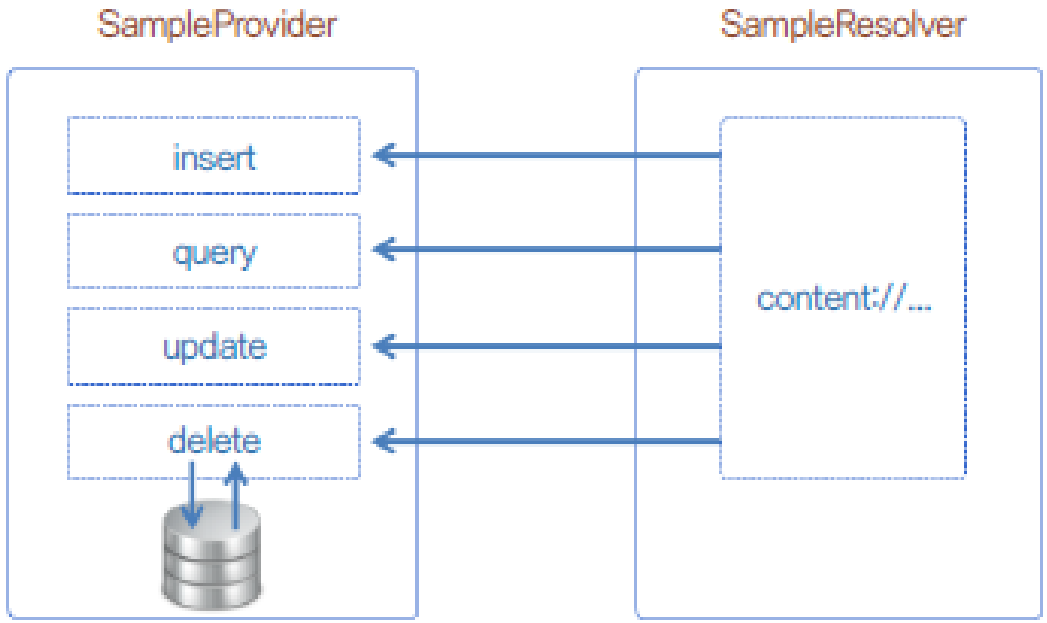
실행 화면

SQL을 이용한 테이블 레코드 조회

5. 내용 제공자 이해하기

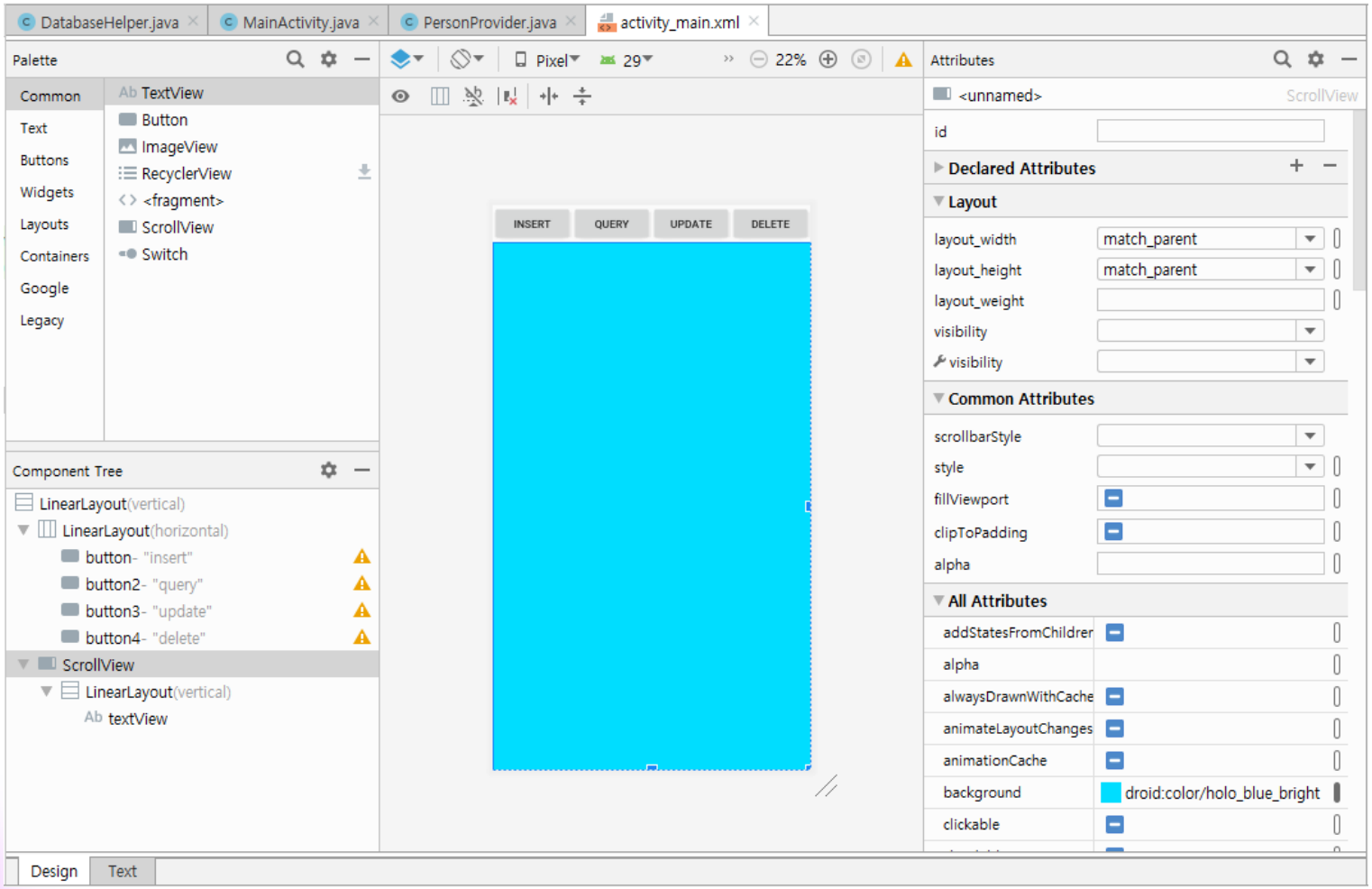
내용 제공자

- 내용 제공자는 콘텐트 프로바이더(Content Provider) 라고도 불림
- 다른 앱에서 데이터를 접근할 수 있도록 함
- Provider에서 제공하는 데이터는 Resolver를 이용해 접근함



화면 레이아웃 만들기

- 상단에 버튼 배치하고 하단에는 스크롤뷰 안에 텍스트뷰 배치



데이터베이스 코드 만들기

- SQLiteHelper를 이용해 데이터베이스를 위한 코드 추가

```
    private static final String CREATE_TABLE =
        "CREATE TABLE " + TABLE_NAME + " (" +
              PERSON_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
              PERSON_NAME + " TEXT, " +
              PERSON_AGE + " INTEGER, " +
              PERSON_MOBILE + " TEXT" +
        ")";

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

    @Override
    public void onCreate(SQLiteDatabase db) {
      db.execSQL(CREATE_TABLE);
    }

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

Provider 클래스 만들기

- ContentProvider를 상속해 새로운 Provider 클래스 정의

참조파일 SampleProvider>/app/java/org.techtown.provider/PersonProvider.java

```
public class PersonProvider extends ContentProvider {

    private static final String AUTHORITY = "org.techtown.provider";
    private static final String BASE_PATH = "person";
    public static final Uri CONTENT_URI = Uri.parse("content://" + AUTHORITY + "/" + BASE_PATH );

    private static final int PERSONS = 1;
    private static final int PERSON_ID = 2;
    private static final UriMatcher uriMatcher = new UriMatcher(UriMatcher.NO_MATCH);
    static {
        uriMatcher.addURI(AUTHORITY, BASE_PATH, PERSONS);
        uriMatcher.addURI(AUTHORITY, BASE_PATH + "/#", PERSON_ID);
    }

    private SQLiteDatabase database;

    @Override
    public boolean onCreate() {
        DatabaseHelper helper = new DatabaseHelper(getContext());
        database = helper.getWritableDatabase();

        return true;
    }
```

Provider 클래스 만들기

- 조회를 위한 query 메서드 정의
- UriMatcher를 사용하고 커서를 생성해 반환

```
@Nullable
@Override
public Cursor query(Uri uri, String[] strings, String s, String[] strings1, String s1) {
  Cursor cursor;
  switch (uriMatcher.match(uri)) {
    case PERSONS:
      cursor = database.query(DatabaseHelper.TABLE_NAME,
                  DatabaseHelper.ALL_COLUMNS,
                  s,null,null,null,DatabaseHelper.PERSON_NAME +" ASC");
      break;
    default:
      throw new IllegalArgumentException("알 수 없는 URI " + uri);
  }
  cursor.setNotificationUri(getContext().getContentResolver(), uri);

  return cursor;
}
```

Provider 클래스 만들기

- 추가를 위한 insert 등의 메서드 추가

```
@Nullable
@Override
public Uri insert(Uri uri, ContentValues contentValues) {
  long id = database.insert(DatabaseHelper.TABLE_NAME, null, contentValues);

  if (id > 0) {
    Uri _uri = ContentUris.withAppendedId(CONTENT_URI, id);
    getContext().getContentResolver().notifyChange(_uri, null);
    return _uri;
  }

  throw new SQLException("추가 실패-> URI :" + uri);
}
```

Provider 클래스 만들기

- Content URI 정의 형식

content://org.techtown.provider/person/1

content:// → 내용 제공자에 의해 제어되는 데이터라는 의미로 항상 content:// 로 시작함

Authority → org.techtown.provider 부분을 가리키며 특정 내용 제공자를 구분하는 고유한 값

Base Path → person 부분을 가리키며 요청할 데이터의 자료형을 결정함 (여기에서는 테이블 이름)

ID → 맨 뒤의 1과 같은 숫자를 가리키며 요청할 데이터 레코드를 지정함

Provider 클래스 만들기

- query 메서드

```
Cursor query (
                Uri uri,
                String[] projection,
                String selection,
                String[] selectionArgs,
                String sortOrder
            )
```

메인 액티비티에서 Provider의 메서드 호출

- insertPerson 메서드 정의

```
public void insertPerson() {
  println("insertPerson 호출됨");

  String uriString = "content://org.techtown.provider/person";
  Uri uri = new Uri.Builder().build().parse(uriString);

  Cursor cursor = getContentResolver().query(uri, null, null, null, null);
  String[] columns = cursor.getColumnNames();
  println("columns count -> " + columns.length);

  for (int i = 0; i < columns.length; i++) {
    println("#" + i + " : " + columns[i]);
  }

  ContentValues values = new ContentValues();
  values.put("name", "john");
  values.put("age", 20);
  values.put("mobile", "010-1000-1000");

  uri = getContentResolver().insert(uri, values);
  println("insert 결과-> " + uri.toString());
}

중략…
```

메인 액티비티에서 Provider의 메서드 호출

- queryPerson 메서드 정의

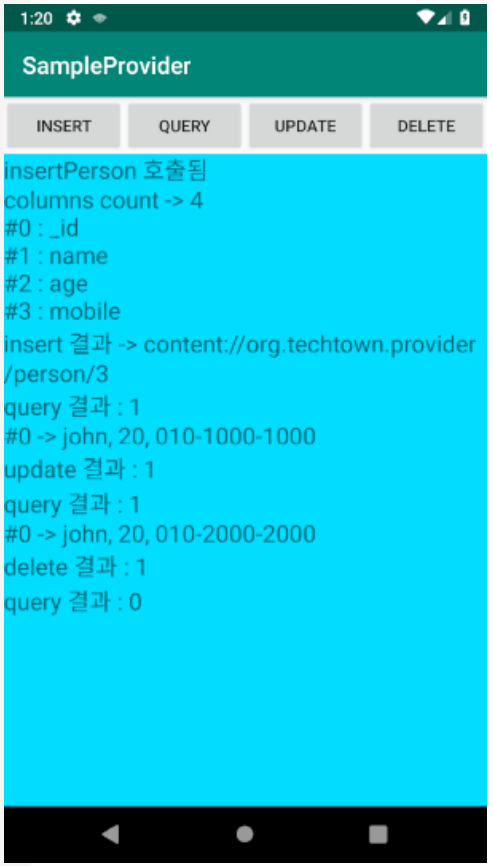
```
public void queryPerson() {
  try {
    String uriString = "content://org.techtown.provider/person";
    Uri uri = new Uri.Builder().build().parse(uriString);

    String[] columns = new String[] {"name", "age", "mobile"};
    Cursor cursor = getContentResolver().query(uri, columns, null, null, "name ASC");
    println("query 결과: " + cursor.getCount());

    int index = 0;
    while(cursor.moveToNext()) {
      String name = cursor.getString(cursor.getColumnIndex(columns[0]));
      int age = cursor.getInt(cursor.getColumnIndex(columns[1]));
      String mobile = cursor.getString(cursor.getColumnIndex(columns[2]));

      println("#" + index + " -> " + name + ", " + age + ", " + mobile);
      index += 1;
    }
```

매니페스트에 내용 제공자 등록

- SD 카드 접근 권한과 provider 태그 추가

참조파일 SampleProvider>/app/manifests/AndroidManifest.xml

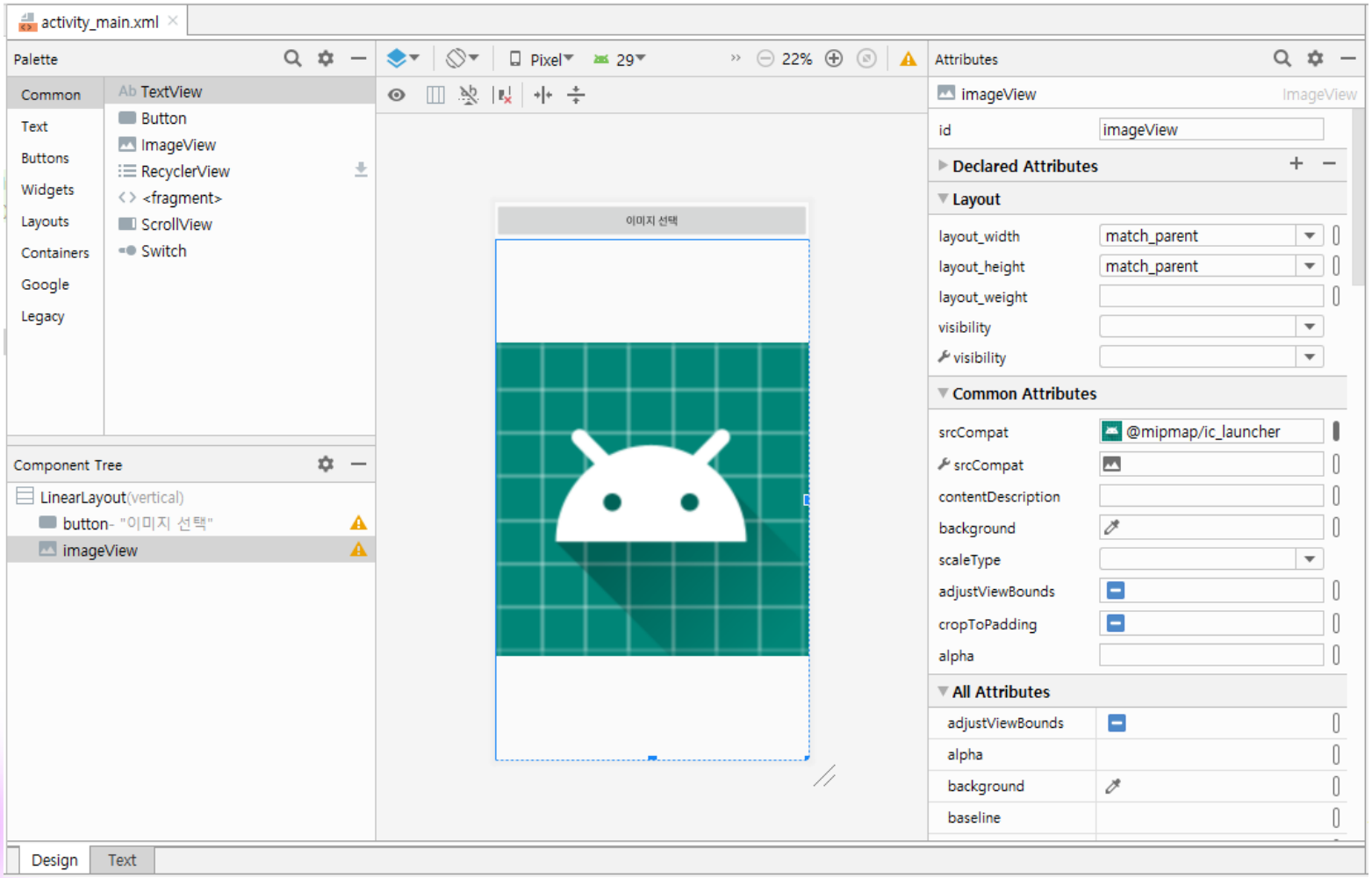
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          package="org.techtown.provider">

  <permission android:name="org.techtown.provider.READ_DATABASE" android:protectionLevel="normal" />
  <permission android:name="org.techtown.provider.WRITE_DATABASE" android:protectionLevel="normal" />

  <application

중략…

    <provider
      android:authorities="org.techtown.provider"
      android:name=".PersonProvider"
      android:exported="true"
      android:readPermission="org.techtown.provider.READ_DATABASE"
      android:writePermission="org.techtown.provider.WRITE_DATABASE"
    />

  </application>
</manifest>
```



6. 앨범과 연락처 조회하기

화면 레이아웃 만들기

- 상단에 버튼 배치하고 하단에는 이미지뷰 배치



메인 액티비티에서 앨범 띄우기

- 인텐트를 이용해 앨범 화면 띄우기

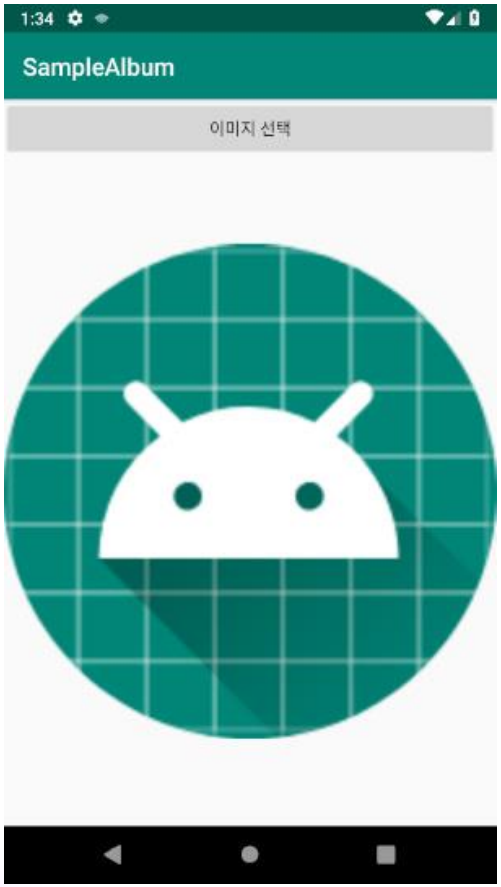
```
public void openGallery() {
  Intent intent = new Intent();
  intent.setType( " image/* " );
  intent.setAction(Intent.ACTION_GET_CONTENT);

  startActivityForResult(intent, 101);
}
```

선택된 사진 보여주기

- onActivityResult 메서드 안에서 선택된 사진 보여주기

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
  if(requestCode == 101) {
    if(resultCode == RESULT_OK) {
      Uri fileUri = data.getData();

      ContentResolver resolver = getContentResolver();  ❶ ContentResolver 객체 참조하기

      try {

        InputStream instream = resolver.openInputStream(fileUri);  ❷ ContentResolver 객체의 openInput-Stream() 메서드로 파일 읽어 들이기
        Bitmap imgBitmap = BitmapFactory.decodeStream(instream);
        imageView.setImageBitmap(imgBitmap);
        instream.close();
      } catch(Exception e) {
        e.printStackTrace();
      }
    }
  }
}
```

앱 실행

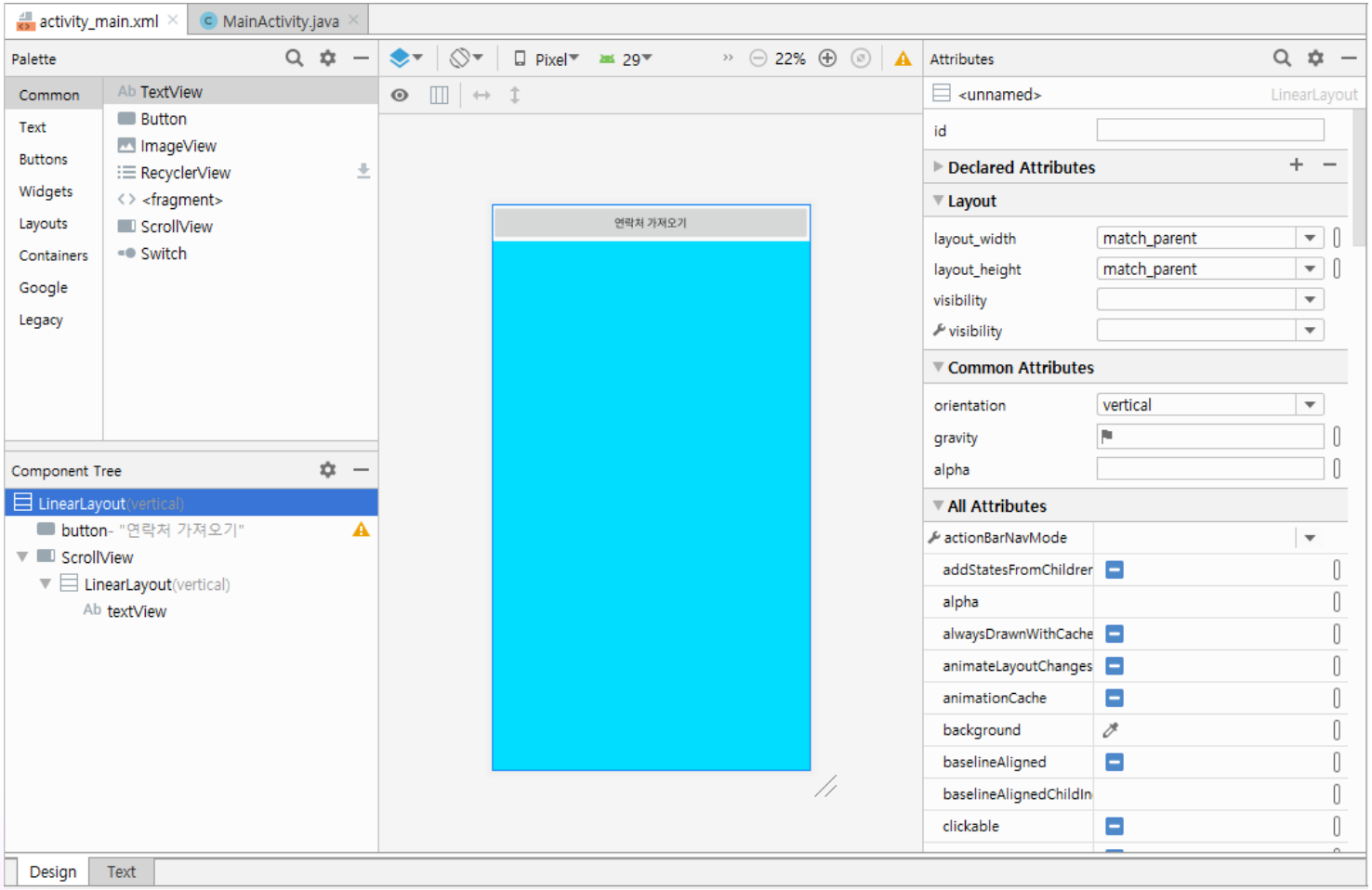

1:34
SampleAlbum
이미지 선택


1:35
Recent
Modified
IMG_20190112...
199 kB 4:19 AM
IMG_20190112...
200 kB 4:19 AM
IMG_20190112...
200 kB 4:19 AM


1:35
SampleAlbum
이미지 선택

연락처 선택을 위한 화면 레이아웃 만들기

- 상단에 버튼 배치하고 하단에는 스크롤뷰 안에 텍스트뷰 배치하기



메인 액티비티에서 연락처 화면 띄우기

- 인텐트로 연락처 선택 화면 띄우고 결과 받아 표시하기

```
public void chooseContacts() {
  Intent contactPickerIntent = new Intent(Intent.ACTION_PICK,
  ContactsContract.Contacts.CONTENT_URI); ⟶ ❶ 연락처 화면을 띄우기 위한 인텐트 만들기
  startActivityForResult(contactPickerIntent,101);
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
  if (resultCode == RESULT_OK) {
    if (requestCode == 101) {
      try {
        Uri contactsUri = data.getData();
        String id = contactsUri.getLastPathSegment(); ⟶ ❷ 선택한 연락처의 id 값 확인하기

        getContacts(id);
      } catch (Exception e) {
        e.printStackTrace();
      }
    }
  }
}
```

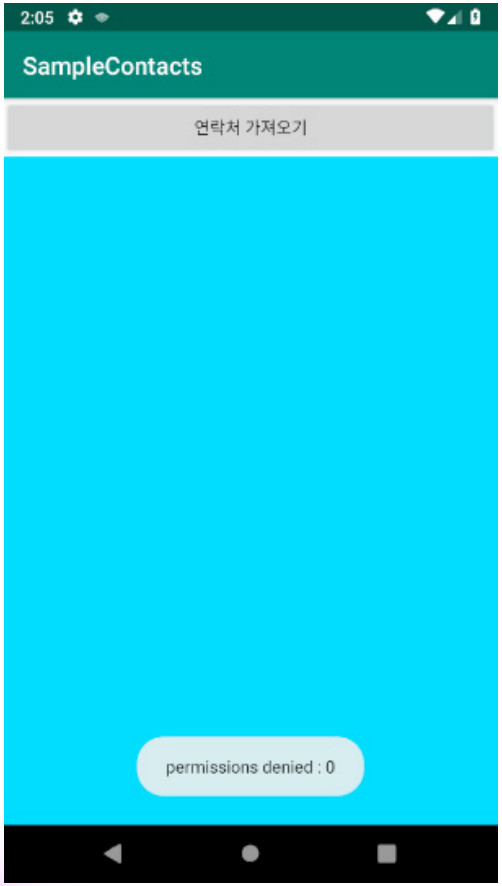
메인 액티비티에서 연락처 화면 띄우기

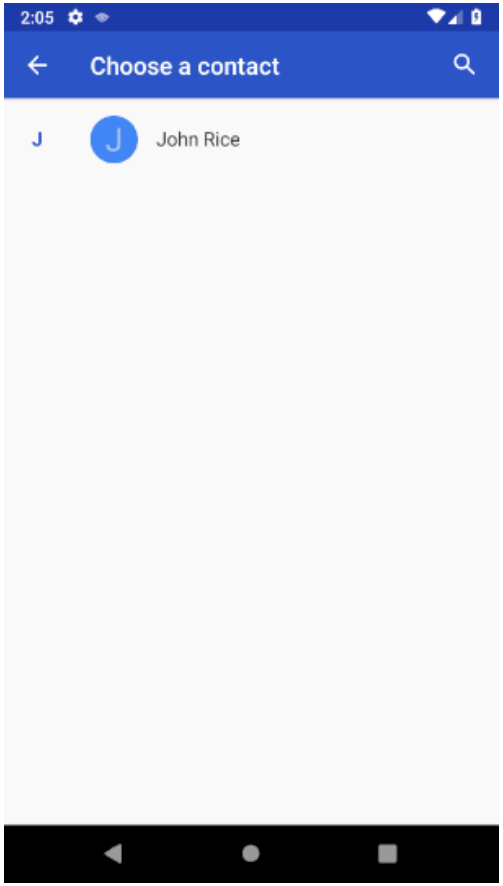
- getContacts 메서드에서 연락처 정보 접근

```
public void getContacts(String id) {
  Cursor cursor = null;
  String name = "";

  try {
    cursor = getContentResolver().query(ContactsContract.Data.CONTENT_URI,
                    null,
                    ContactsContract.Data.CONTACT_ID + "=?",
                    new String[] { id },
                    null);

    if (cursor.moveToFirst()) {
      name = cursor.getString(cursor.getColumnIndex(ContactsContract.Data.DISPLAY_NAME));
      println("Name : " + name);

      String columns[] = cursor.getColumnNames();
      for (String column : columns) {
        int index = cursor.getColumnIndex(column);
        String columnOutput = ("#" + index + " -> [" + column + "] " + cursor.getString(index));
        println(columnOutput);
      }
      cursor.close();
    }
```

ContentResolver 객체의 query() 메서드 호출하기

앱 실행


2:05
SampleContacts
연락처 가져오기
permissions denied : 0


2:05
Choose a contact
J
J
John Rice


2:05
SampleContacts
연락처 가져오기
Name : John Rice
#0 -> [phonetic_name] null
#1 -> [status_res_package] null
#2 -> [custom_ringtone] null
#3 -> [contact_status_ts] null
#4 -> [account_type] null
#5 -> [data_version] 0
#6 -> [photo_file_id] null
#7 -> [contact_status_res_package] null
#8 -> [group_sourceid] null
#9 -> [display_name_alt] Rice, John
#10 -> [sort_key_alt] Rice, John
#11 -> [mode] null
#12 -> [last_time_used] null
#13 -> [starred] 0
#14 -> [contact_status_label] null
#15 -> [has_phone_number] 1
#16 -> [chat_capability] null
#17 -> [raw_contact_id] 1
#18 -> [carrier_presence] 0
#19 -> [contact_last_updated_timestamp]
1547278988058
#20 -> [res_package] null