09. 데이터 저장

시작하면서

- **목차**
  - 프레퍼런스
  - 데이터베이스 저장
  - 컨텐트 제공자

프레퍼런스

프레퍼런스(Preference)

- 제일 단순한 저장 형태
- 각 애플리케이션에 고유한 설정값을 지정
- <키, 값>의 조합으로 데이터 저장
  - 값에 이름을 부여하여 저장
  - 환경설정에 유용
- 주요 메소드
  - SharedPreference 인터페이스
    - getSharedPreference()에 의해 반환된 프레퍼런스 객체를 접근/수정 제공
    - SharedPreference.Editor editor = pref.edit()
      - 프레퍼런스 객체를 수정 후, commit() 연산으로 배치로 처리

프레퍼런스

- **<실습 9-1> 공유 프레퍼런스 설정**
  - 프로젝트 SharedPreference를 생성
    - <코드 9-1>를 main.xml에 복사
    - <코드 9-2>를 SharedPreference.java에 복사
    - 특기사항: 편집 텍스트 입력 및 체크박스 표시 후 DDMS에서 애플리케이션 종료 후 재실행을 통해 프레퍼런스 값이 영속적임을 확인

    - **<SharedPreference.java>**
      - **public void onCreate(Bundle savedInstanceState) {**

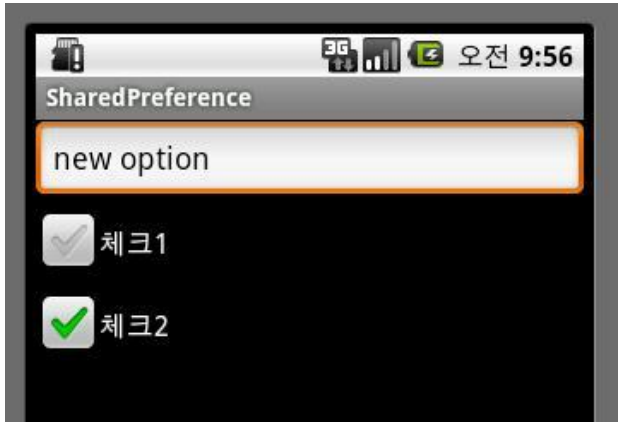
```
SharedPreferences pref = getSharedPreferences("pref",
  Activity.MODE_PRIVATE);
String text = pref.getString("editText", "");
edit1.setText(text);
}
```

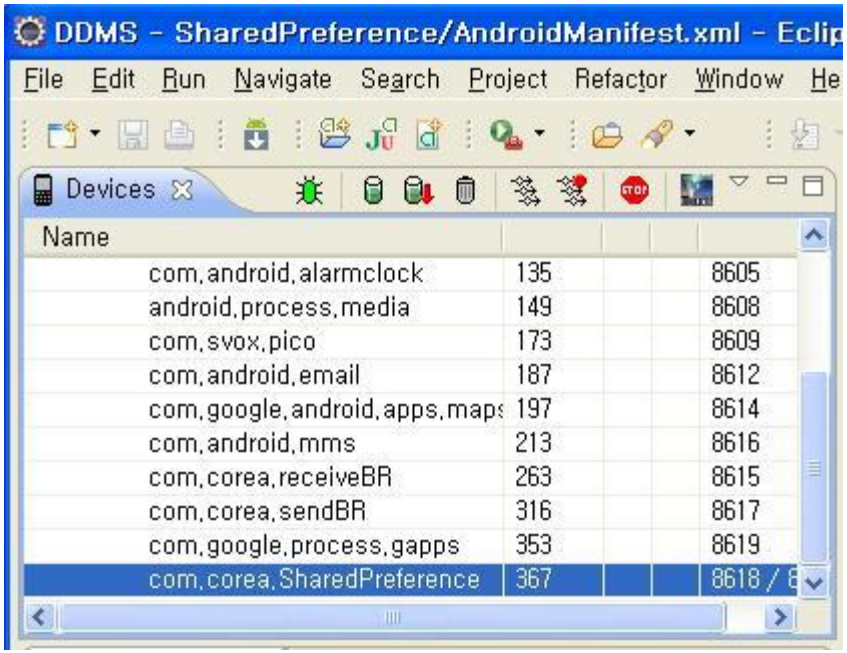
      - **public void onStop(){**

```
SharedPreferences pref = getSharedPreferences("pref",
  Activity.MODE_PRIVATE);
SharedPreferences.Editor editor = pref.edit();
editor.putString("editText", edit1.getText().toString());
editor.commit();
}
```

프레퍼런스


오전 9:51
SharedPreference
글자 입력!
체크1
체크2


오전 9:56
SharedPreference
new option
체크1
체크2


DDMS - SharedPreference/AndroidManifest.xml - Eclip
File Edit Run Navigate Search Project Refactor Window He
Devices
Name
com.android.alarmclock 135 8605
android.process.media 149 8608
com.svox.pico 173 8609
com.android.email 187 8612
com.google.android.apps.maps 197 8614
com.android.mms 213 8616
com.corea.receiveBR 263 8615
com.corea.sendBR 316 8617
com.google.process.gapps 353 8619
com.corea.SharedPreference 367 8618

데이터베이스 저장

데이터베이스 저장

- **관계형 데이터베이스relational database를 이용**
  - 표 형태의 데이터 관리
  - 안드로이드에서 내장형 데이터베이스 SQLite 제공
  - 파일과 데이터베이스는 자신이 생성된 애플리케이션에 종속
  - 주요 기능
    - 데이터베이스 생성과 버전 부여
    - 데이터베이스 관리를 위한 클래스
    - 적절한 SQL 명령문과 질의문 작성을 보조하는 클래스
    - 질의 결과를 탐색하기 위한 커서 클래스
  - 커서Cursor 클래스
    - SQLite 데이터베이스의 데이터나 콘텐츠 제공자가 제공하는 데이터에 접근
    - 복수의 데이터들을 순차적으로 처리하는 경우 주로 사용
  - SQLite 데이터베이스 접근 방식
    1. 코드에서 생성/접근하는 방식
    2. ADB 쉘 인터페이스에서 명령행 도구 sqlite3를 이용해 데이터베이스를 직접 조작하는 방식

데이터베이스 실습

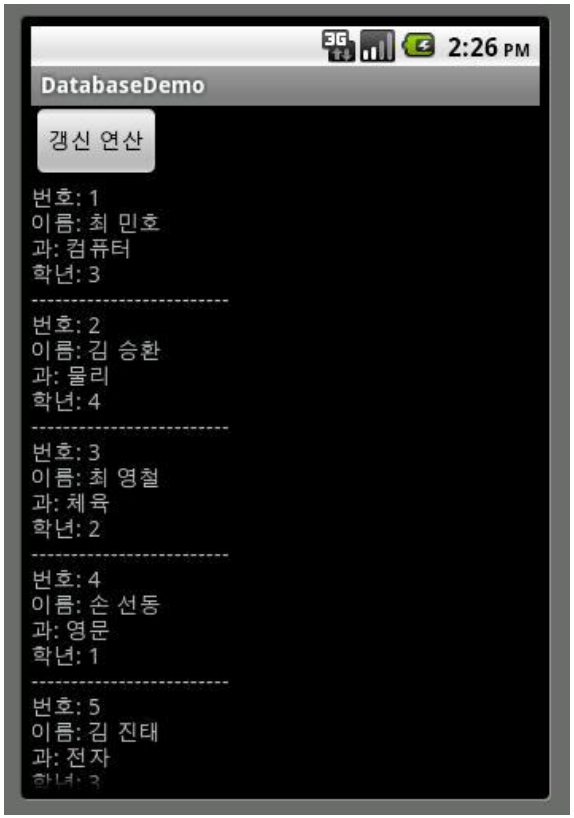
- 실습 내용
  - 데이터베이스 생성
    - SQLiteDatabase.create()을 호출
    - SQLiteOpenHelper를 호출
  - 데이터베이스에 레코드를 삽입insert, 갱신update, 삭제delete하는 실습
  - 데이터베이스 이름: school
  - 테이블 이름: students
- 실습 순서
  - 삽입 연산: students에 5개 항목을 삽입
  - 변경 연산: students의 2번 항목을 변경
  - 삭제 연산: students의 1번 항목을 삭제
- 3개 액티비티로 구성
  - <activity android:name=".InsertActivity"> … </activity>
  - <activity android:name="UpdateActivity"></activity>
  - <activity android:name="DeleteActivity"></activity>

데이터베이스 실습

- **<실습 9-2> 데이터베이스 실습**
  - DatabaseDemo 프로젝트 생성
    - <코드 9-3>을 AndroidManifest.xml에 복사
    - <코드 9-4>를 insert.xml에 복사
    - <코드 9-5>를 InsertActivity.java에 복사
    - <코드 9-6>을 InsertActivity.java의 끝에 추가
    - <코드 9-7>을 update.xml에 복사
    - <코드 9-8>을 UpdateActivity.java에 복사
    - <코드 9-9>를 delete.xml에 복사
    - <코드 9-10>을 DeleteActivity.java에 복사
    - <코드 9-11>을 DBAdapter.java에 복사
    - <코드 9-12>를 DBAdapter.java의 끝에 추가
  - 삽입(레코드 5개), 변경(레코드 1개), 삭제(레코드 1개) 연산을 차례로 수행

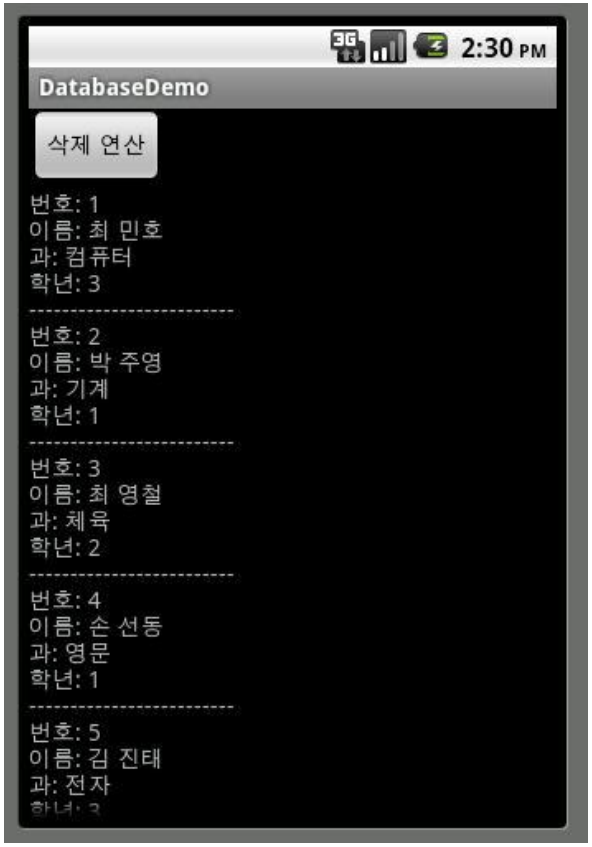
데이터베이스 실습: 삽입


2:24 PM
DatabaseDemo
갱신 연산
번호: 1
이름: 최 민호
과: 컴퓨터
학년: 3
번호: 2
이름: 김 승환
과: 물리
레코드 5개 삽입 성공
OK
번호: 4
이름: 손 선동
과: 영문
학년: 1
번호: 5
이름: 김 진태
과: 전자


2:26 PM
DatabaseDemo
갱신 연산
번호: 1
이름: 최 민호
과: 컴퓨터
학년: 3
번호: 2
이름: 김 승환
과: 물리
학년: 4
번호: 3
이름: 최 영철
과: 체육
학년: 2
번호: 4
이름: 손 선동
과: 영문
학년: 1
번호: 5
이름: 김 진태
과: 전자

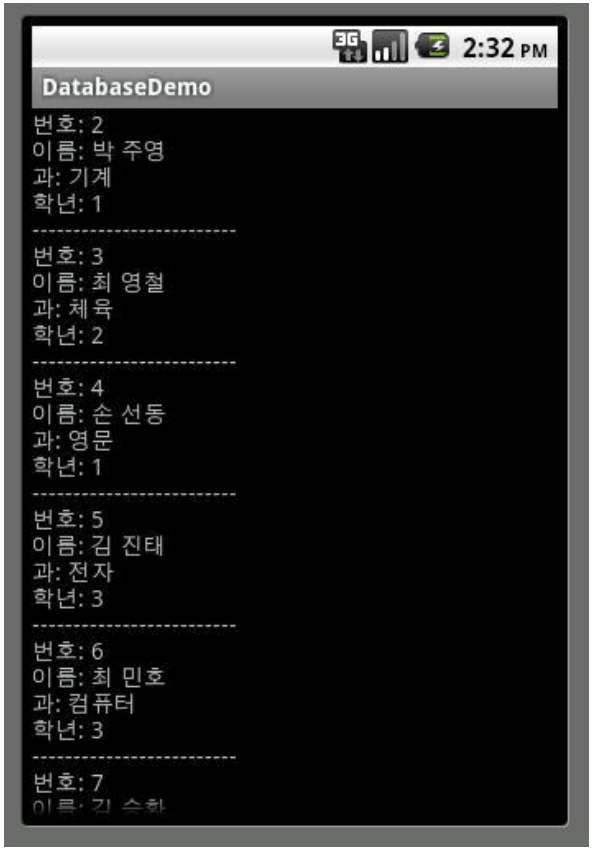
데이터베이스 실습: 변경


2:29 PM
DatabaseDemo
삭제 연산
번호: 1
이름: 최 민호
과: 컴퓨터
학년: 3
번호: 2
이름: 박 주영
과: 기계
레코드 2번 갱신 성공
OK
번호: 4
이름: 손 선동
과: 영문
학년: 1
번호: 5
이름: 김 진태
과: 전자


2:30 PM
DatabaseDemo
삭제 연산
번호: 1
이름: 최 민호
과: 컴퓨터
학년: 3
번호: 2
이름: 박 주영
과: 기계
학년: 1
번호: 3
이름: 최 영철
과: 체육
학년: 2
번호: 4
이름: 손 선동
과: 영문
학년: 1
번호: 5
이름: 김 진태
과: 전자

데이터베이스 실습: 삭제


2:31 PM
DatabaseDemo
번호: 2
이름: 박 주영
과: 기계
학년: 1
번호: 3
이름: 최 영철
과: 체육
학년: 2
번호: 4
레코드 1번 삭제 성공
OK
학년: 3
번호: 6
이름: 최 민호
과: 컴퓨터
학년: 3
번호: 7


2:32 PM
DatabaseDemo
번호: 2
이름: 박 주영
과: 기계
학년: 1
번호: 3
이름: 최 영철
과: 체육
학년: 2
번호: 4
이름: 손 선동
과: 영문
학년: 1
번호: 5
이름: 김 진태
과: 전자
학년: 3
번호: 6
이름: 최 민호
과: 컴퓨터
학년: 3
번호: 7

데이터베이스 실습: 코드 부분

데이터베이스 실습: 삽입부

- **InsertActivitiy**

```
DBAdapter db = new DBAdapter(this); …
db.open();
id = db.insertTitle("홍 길동","컴퓨터","2");
    …
Cursor c = db.getAllTitles();
If ( c.moveToFirst() ) {
    alltables = alltables.concat(…);
    while ( c.moveToNext() );
    …
db.close();
   …
    startActivity(new Intent(InsertActivity.this, UpdateActivity.class));
```

데이터베이스 실습: 변경부

- **UpdateActivitiy**

```
DBAdapter db = new DBAdapter(this); …
db.open();
id = db.updateTitle(2, “박 주영”,“기계”,”1”);
    …
Cursor c = db.getAllTitles();
If ( c.moveToFirst() ) {
    alltables = alltables.concat(…);
    while ( c.moveToNext() );
    …
db.close();
   …
    startActivity(new Intent(UpdateActivity.this, DeleteActivity.class));
```

데이터베이스 실습: 삭제부

- **DeleteActivitiy**

```
DBAdapter db = new DBAdapter(this); …
db.open();
id = db.deleteTitle(1);
    …
Cursor c = db.getAllTitles();
If ( c.moveToFirst() ) {
    alltables = alltables.concat(…);
    while ( c.moveToNext() );
    …
db.close();
deleteDatabase(DBAdapter.getDatabaseName());
```

데이터베이스 실습: DB, Table 생성

- **DBAdapter**
  - 항목명(주키), 데이터베이스명, 테이블명

```
public static final String KEY_ROWID = "_id";  ...
private static final String DATABASE_NAME = "school";
private static final String DATABASE_TABLE = "students";
```

  - // SQL문: create a table

```
private static final String TABLE_CREATE =
        "create table students (_id integer primary key autoincrement, "
        + "name text not null, dept text not null, "
        + "grade text not null);";
```

  - // create a database if it doesn't exist, or open if it exists

```
db = DBHelper.getWritableDatabase();
```

  - // create a table

```
db.execSQL(TABLE_CREATE);
```

데이터베이스 실습: helpers

- **DBAdapter**
  - // insertTitle

    ContentValues initialValues = new ContentValues();
    initialValues.put(*KEY_NAME, name);*
    return db.insert(*DATABASE_TABLE, null, initialValues);*

  - // deleteTitle

    db.delete(*DATABASE_TABLE, KEY_ROWID* + "=" + rowId, null) > 0

  - // updateTitle

    db.update(*DATABASE_TABLE, args, KEY_ROWID + "=" + rowId, null) > 0*

데이터베이스 실습: 질의부

- **DBAdapter**
  - // getAllTitles
    - db.query(*DATABASE_TABLE, new String[] {*
    - *KEY_ROWID, KEY_NAME, KEY_DEPARTMENT, KEY_GRADE},*
    - null, null, null, null, null);
  - // getTitle
    - Cursor mCursor = db.query(true, *DATABASE_TABLE, new String[] { KEY_ROWID, ...*},
    - *KEY_ROWID + "=" + rowId,* null, ...);
    - if (mCursor != null)
    - mCursor.moveToFirst();
    - return mCursor;

컨텐트 제공자

컨텐트 제공자

- 애플리케이션이 저장한 데이터를 다른 애플리케이션들이 간편하게 접근할 수 있는 방법
- 가장 자연스러운 애플리케이션간 데이터 공유 방법
  예)연락처 리스트, 브라우저의 북 마크, 음악이나 사전 등의 미디어 저장media store 접근시 적합
- 안드로이드에서는 데이터 저장작업과 활용 작업을 분리
  - 컨텐트 제공자는 데이터를 저장 및 관리하는 역할을 수행
  - 애플리케이션은 컨텐트 제공자가 제공하는 데이터를 활용
  - 애플리케이션은 ContentResolver 객체를 사용하여 컨텐트 제공자에게 요청
- 자료 공유 방법
  - 사용자는 컨텐트 리졸버와 URI를 통해 컨텐트 제공자에게 원하는 데이터를 요청

  →컨텐트 제공자는 사용자에게 요청받은 데이터를 제공

◉ **URI** **Uniform Resource Identifier**

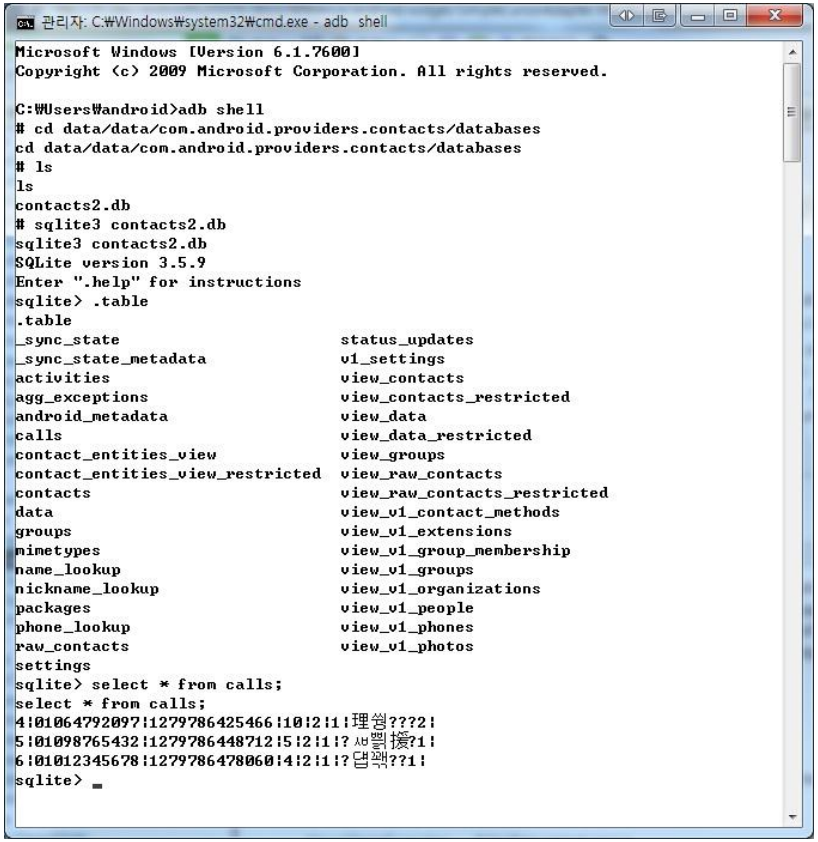
- 데이터에 대한 접근 형식
- scheme://authority/path/id
- 주요 컨텐트 제공자: <표 9-1>

내부 데이터베이스 접근

- 내부 데이터베이스에 저장된 전화통화 기록 접근

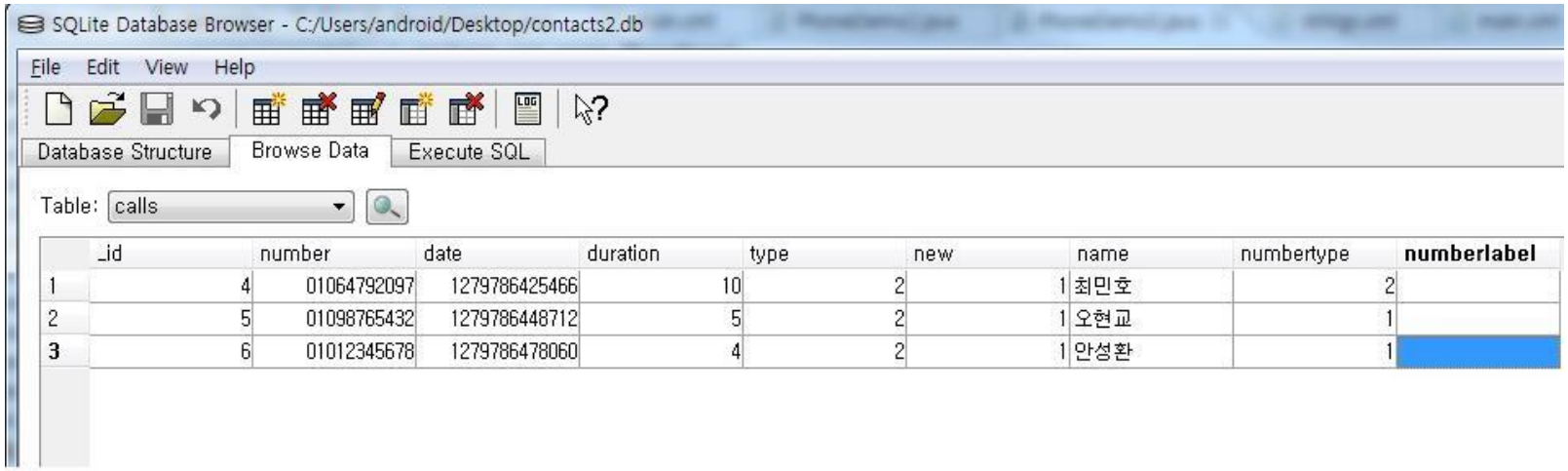
1)ADB를 통한 접근: [그림 9-5]

  - Adb shell 명령으로 sqlite3을 실행하여 전화 통화 기록 조회



내부 데이터베이스 접근

2)DDMS를 통한 접근: [그림 9-6]



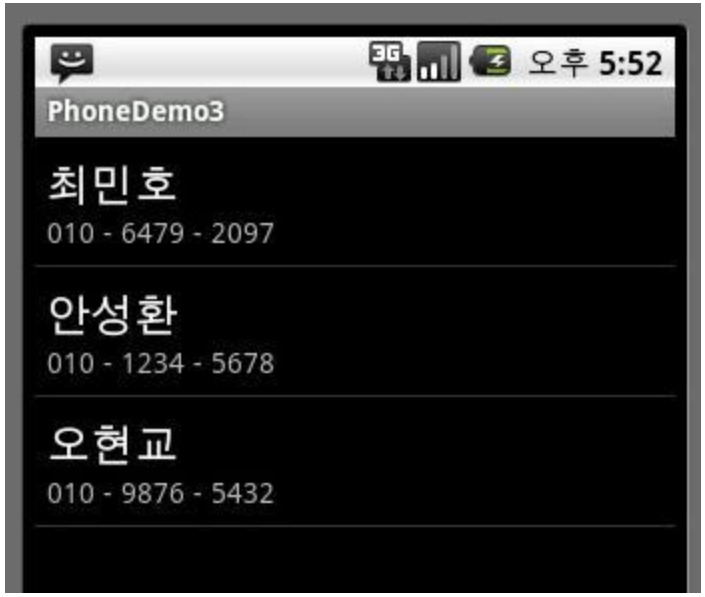
	_id	number	date	duration	type	new	name	numbertype	numberlabel
1	4	01064792097	1279786425466	10	2	1	최민호	2	
2	5	01098765432	1279786448712	5	2	1	오현교	1	
3	6	01012345678	1279786478060	4	2	1	안성환	1	

컨텐트 제공자 실습

- 이름과 전화번호로 구성된 전화 통화 기록을 검색
- 데이터에 대한 접근 권한 명시 필요

<실습 9-3> 콘텐트 제공자 실습

- PhoneHistoryDemo 프로젝트 생성
  - <코드 9-13>을 main.xml에 복사
  - <코드 9-14>로 AndroidManifest.xml을 수정
  - <코드 9-15>를 CallLogDemo.java에 복사



컨텐트 제공자 실습

- **<AndroidManifest.xml>**

```
<uses-permission android:name="android.permission.READ_CONTACTS">
</uses-permission>
```

- **<PhoneHistoryDemo.java>**

```
private static String[] PROJECTION = new String[] {
        Phone._ID, Phone.DISPLAY_NAME, Phone.NUMBER };
Cursor c = getContentResolver().query(Phone.CONTENT_URI,
            PROJECTION, null, null, null);
startManagingCursor(c);
ListAdapter adapter = new SimpleCursorAdapter(this
        android.R.layout.simple_list_item_2, c,
        new String[] {Phone.DISPLAY_NAME, Phone.NUMBER},
        new int[] {android.R.id.text1,android.R.id.text2});
setListAdapter(adapter);
```