

Mavzu: KONTENT PROVAYDERLAR

Bizning ilovamiz turli xil foydalanuvchi ma'lumotlarini, jumladan, fayllar yoki sozlamalardagi tegishli ma'lumotlarni saqlashi mumkin. Biroq, Android OS allaqachon biz kirishimiz va foydalanishimiz mumkin bo'lgan bir qator muhim foydalanuvchi ma'lumotlarini saqlaydi. Bunga kontaktlar ro'yxati, saqlangan rasm va video fayllar, qo'ng'iroqlar jurnallari va boshqalar kiradi - boshqacha aytganda, kontent. Ushbu kontentga kirish uchun Android **kontent provayderlarini belgilaydi.**

android.content paketida belgilangan quyidagi o'rnatilgan provayderlar mavjud :

AlarmClock : budilnikni boshqarish

Brauzer : Brauzer tarixi va xatcho'plar

CalendarContract : taqvim va tadbir haqida ma'lumot

Qo'ng'iroqlar jurnali : Qo'ng'iroqlar haqida ma'lumot

KontaktlarShartnoma : kontaktlar

MediaStore : media fayllar

SearchRecentSuggestions : qidiruv takliflari

Sozlamalar : tizim sozlamalari

UserDictionary : Tez terish uchun ishlatiladigan so'zlar lug'ati.

Ovozli pochta shartnomasi : ovozli pochta yozuvlari

Kontaktlar bilan ishlash

Android Contacts ilovasida kontaktlar ro'yxatini olish va o'zgartirish imkonini beruvchi o'rnatilgan API mavjud. Barcha kontaktlar SQLite ma'lumotlar bazasida saqlanadi, lekin ular bitta jadval hosil qilmaydi. Kontaktlar uchta jadvalga bo'lingan, ularning har biri bittadan ko'pga munosabatda bog'langan: odamlar ma'lumotlarini saqlash uchun jadval, ularning telefon raqamlari uchun jadval va elektron pochta manzillari uchun jadval. Biroq, Android API tufayli biz bu jadvallar orasidagi munosabatlarni mavhumlashtirishimiz mumkin.

Kontaktlarni qabul qilishning umumiy shakli quyidagicha:

```
ArrayList<String> kontaktlar = yangi ArrayList<String>();
ContentResolver contentResolver = getContentResolver();
Kursor kursor =
contentResolver.query(ContactsContract.Contacts.CONTENT_URI
, null, null, null, null);
agar (kursor! = null) {
shu bilan birga (kursor.moveKeyingi()) {

// biz olamiz har bir aloqa
Satr aloqasi =
cursor.getString(cursor.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME_PRIMARY));
        // ro'yxatga kontakt qo'shish
contacts.add(aloqa);
}
kursor.close();
}
```

Barcha kontaktlar va tegishli funksiyalar maxsus SQLite ma'lumotlar bazalarida saqlanadi. Biroq, biz ular bilan to'g'ridan-to'g'ri ishlashimiz shart emas. Biz **Cursor** **sinf obyektidan foydalanishimiz mumkin. Uni olish uchun avval** `getContentResolver()` usulini chaqiramiz, bu esa **ContentResolver** **obyektini qaytaradi**. Keyin, zanjirli tarzda, biz `query()` usulini chaqiramiz. Bu usul bir qator parametrlarni oladi, ularning birinchisi biz olishni istagan URI - resursni ifodalaydi. Kontaktlar ma'lumotlar bazasiga kirish uchun biz **ContactsContract.Contacts.CONTENT_URI** doimiysidan foydalanamiz.

`contactsCursor.moveToNext()` usuli sizga `contactsCursor.getString()` ni chaqirib, bir vaqtning o'zida bitta kontaktni o'qib, kontakt yozuvlari bo'ylab ketma-ket harakatlanish imkonini beradi.

Shunday qilib, kontaktlarni olish oson. Boshqa har qanday kontent provayderi singari, kontaktlar bilan ishlashdagi asosiy qiyinchilik ruxsatnomalarni o'rnatishdir. Android API 23 dan oldin, ilovaning manifest faylida tegishli ruxsatnomani o'rnatish kifoya edi. API 23 (Android Marshmallow) dan boshlab, Google

ruxsatnomalar jarayonini o'zgartirdi. Endi foydalanuvchi ilovaga ruxsatnoma berish-bermaslikni o'zi hal qilishi kerak. Bu ishlab chiquvchilardan qo'shimcha kod qo'shishni talab qiladi.

dastur manifest faylida **android.permission.READ_CONTACTS** ruxsatini o'rnatishimiz kerak :

```
< foydalanish ruxsati android:name =  
"android.permission.READ_CONTACTS" />
```

activity_main.xml faylida kontaktlar ro'yxatini ko'rsatish uchun biz quyidagi interfeys belgisini aniqlaymiz:

```
xml versiya="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout  
xmlns:android="  
http://schemas.android.com/apk/res/android "  
xmlns:app=" http://schemas.android.com/apk/res-auto  
"
```

```
android:layout_width="match_parent"
android:layout_height="match_parent">

<MatnKo'rinishi
android:id="@+id/sarlavha"
.....

<ListView
android:id="@+id/contactList"
.....

</androidx.constraintlayout.widget.ConstraintLayout>
```

Kontaklar ro'yxatini ko'rsatish uchun biz ListView elementidan foydalanamiz. Va **MainActivity sinfida** biz kontaktlarni olamiz.

```
xususiy bo'shliq loadContacts() {
ContentResolver contentResolver = getContentResolver();
```

```
Kursor kursor =
contentResolver.query(ContactsContract.Contacts.CONTENT_URI
, null, null, null, null);
ArrayList<String> kontaktlar = yangi ArrayList<String>();

agar (kursor! = null) {
shu bilan birga (kursor.moveKeyingi()) {

// biz olamiz har bir aloqa
Satr aloqasi = cursor.getString()
cursor.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME_PRIMARY));
// ro'yxatga kontakt qo'shish
contacts.add(aloqa);
}
kursor.close();
}

// yaratish adapter
```

```
ArrayAdapter<String> adapteri = yangi ArrayAdapter<>(bu,  
android.R.layout.simple_list_item_1, kontaktlar);  
  
ListView contactList = findViewById(R.id.contactList);
```

Ma'lumotlarni qabul qilish

Ma'lumotlarni olish uchun **query() usuli provayderda aniqlangan** .

```
ommaviy Kursor so'rovi(@NonNull Uri uri, @Nullable  
String[] proyeksiyasi, @Nullable Satr tanlash,  
@Nullable String[] selectionArgs, @Nullable Satr  
saralash() {  
yakuniy int match = sUriMatcher.match(uri);  
SQLiteQueryBuilder queryBuilder = new  
SQLiteQueryBuilder();  
switch(mos kelish) {  
ish DO'STLAR:  
queryBuilder.setTables(DO'stlarKontrakt.TABLE_NAME);
```

```
tanaffus;
ish DO'STLAR_ID:
queryBuilder.setTables(DO'stlarKontrakt.TABLE_NAME);
uzoq vazifaId = FriendsContract.getFriendId(uri);
queryBuilder.appendWhere(FriendsContract.Columns._ID
+ " = " + vazifa identifikatori);
tanaffus;
standart:
otish yangi IllegalArgumentException("Noma'lum URI:
"+ uri);"
}
SQLiteDatabase ma'lumotlar bazasi =
mOpenHelper.getReadableDatabase();
qaytish queryBuilder.query(db, proyeksiya, tanlash,
selectionArgs, null, null, sortOrder);
}
```

Bu usul beshta parametрни olishi kerak:

- **uri** : so'rov yo'li
- **proyeksiya** : ma'lumotlar olinishi kerak bo'lgan ustunlar to'plami
- **tanlash** : "WHERE Name = ?" turini tanlash uchun ifoda.
- **selectionArgs** : tanlov parametrlari uchun qiymatlar to'plami (savol belgilari o'rniga qo'shilgan)
- **sortOrder** : saralash mezon, ya'ni ustun nomi

Ma'lumotlar qo'shilmoqda

Ma'lumotlarni qo'shish uchun **insert()** usulidan foydalaning :

```

ommaviy Uri qo'shish(@NonNull Uri uri,
@Nullable ContentValues qiymatlari) {

yakuniy int match =
sUriMatcher.match(uri);
yakuniy SQLiteDatabase ma'lumotlar
bazasi;
Uri returnUri;
uzoq yozuv identifikatori;

```

```
agar (mos kelish == DO'STLAR) {  
db = mOpenHelper.getWritableDatabase();  
recordId =  
db.insert(FriendsContract.TABLE_NAME,  
null, qiymatlar);  
agar (yozuv identifikatori > 0) {  
returnUri =  
FriendsContract.buildFriendUri(recordId);  
} boshqa {  
otish yangi SQLException("Qo'shish amalga  
oshmadi:" + uri.toString());  
}  
} boshqa {  
otish yangi  
IllegalArgumentException("Noma'lum URI:"  
+ uri);  
}  
qaytish qaytishUri;
```

```
}
```

Usul ikkita parametрни oladi:

- **uri** : so'rov yo'li
- **qiymatlar** : qo'shilgan ma'lumotlar uzatiladigan ContentValues ob'ekti

Ma'lumotlarni o'chirish

```
ommaviy int o'chirish(@NonNull Uri  
uri, @Nullable Satr tanlash, @Nullable  
String[] selectionArgs) {  
yakuniy int match =  
sUriMatcher.match(uri);  
yakuniy SQLiteDatabase ma'lumotlar  
bazasi =  
mOpenHelper.getWritableDatabase();  
  
Satr tanlashMezonlar = tanlash;
```

```
agar (mos kelsa != DO'STLAR && mos  
kelsa != DO'STLAR_ID)  
otish yangi  
IllegalArgumentException("Noma'lum  
URI: "+ uri);"
```

```
agar (mos kelish == DO'STLAR_ID) {  
uzoq vazifaId =  
FriendsContract.getFriendId(uri);  
tanlashCriteria =  
FriendsContract.Columns._ID + " = " +  
vazifa identifikatori;  
agar ((tanlov != null) &&  
(tanlov.length() > 0)) {  
tanlash mezonlari += "VA (" + tanlash  
+ ")";  
}  
}
```

```
qaytish
db.delete(Do'stlarKontrakt.TABLE_NAME,
tanlashKriteriyalari, tanlashArgs);
}
```

Bu usul uchta parametрни olishi kerak:

- **uri** : so'rov yo'li
- **tanlash** : "WHERE Name = ?" turini tanlash uchun ifoda.
- **selectionArgs** : tanlov parametrlari uchun qiymatlar to'plami (savol belgilari o'rniga qo'shilgan)

Ma'lumotlar yangilanmoqda

Ma'lumotlarni yangilash uchun **update()** usulidan foydalaning :

```
ommaviy int yangilash(@NonNull Uri
uri, @Nullable ContentValues
```

```
qiymatlari, @Nullable Satr tanlash,  
@Nullable String[] selectionArgs) {  
  
yakuniy int match =  
sUriMatcher.match(uri);  
yakuniy SQLiteDatabase ma'lumotlar  
bazasi =  
mOpenHelper.getWritableDatabase();  
Satr tanlashMezonlar = tanlash;  
  
agar (mos kelsa != DO'STLAR && mos  
kelsa != DO'STLAR_ID)  
otish yangi  
IllegalArgumentException("Noma'lum  
URI: "+ uri);"  
  
agar (mos kelish == DO'STLAR_ID) {  
uzoq vazifaId =  
FriendsContract.getFriendId(uri);
```

```
tanlashCriteria =  
FriendsContract.Columns._ID + " = " +  
vazifa identifikatori;  
agar ((tanlov != null) &&  
(tanlov.length() > 0)) {  
tanlash mezonlari += "VA (" + tanlash  
+ ")";  
}  
}  
qaytish  
db.update(FriendsContract.TABLE_NAME,  
qiyamatlar, tanlash mezonlari,  
tanlashArgs);  
}
```

Bu usul to'rtta parametрни olishi kerak:

- **uri** : so'rov yo'li

- **qiymatlar** : yangi qiymatlarni belgilaydigan ContentValues obyekti
- **tanlash** : "WHERE Name = ?" turini tanlash uchun ifoda.
- **selectionArgs** : tanlov parametrlari uchun qiymatlar to'plami (savol belgilari o'rniga qo'shilgan)

Misol

activity_main.xml faylida provayder imkoniyatlarini sinab ko'rish uchun oddiy vizual interfeysni aniqlaylik :

```
1  xml versiya="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout
3  xmlns:android=" http://schemas.android.com/apk/res/android "
4  xmlns:app=" http://schemas.android.com/apk/res-auto "
5  android:layout_width="match_parent"
6  android:layout_height="match_parent" >
7
8  <Tugma
9  android:id="@+id/getButton"
10 android:layout_width="0dp"
11 android:layout_height="wrap_content"
```

```
12 android:text="Olish"
13 android:onClick="All ni olish"
14 ilova: layout_constraintBottom_toTopOf="@id/addButton"
15 ilova: layout_constraintLeft_toLeftOf="ota-ona"
16 ilova:layout_constraintRight_toRightOf="ota-ona"
17 ilova: layout_constraintTop_toTopOf="ota-ona" />
18
19 <Tugma
20 android:id="@+id/addButton"
21 android:layout_width="0dp"
22 android:layout_height="wrap_content"
23 android:text="Qo'shish"
24 android:onClick="qo'shish"
25 ilova: layout_constraintBottom_toTopOf="@id/updateButton"
```

```
26 ilova: layout_constraintLeft_toLeftOf="ota-ona"
27 ilova:layout_constraintRight_toRightOf="ota-ona"
28 ilova: layout_constraintTop_toBottomOf="@id/getButton" />
29
30 <Tugma
31 android:id="@+id/yangilashtugmasi"
32 android:layout_width="0dp"
33 android:layout_height="wrap_content"
34 android:text="Yangilash"
35 android:onClick="yangilash"
36 ilova: layout_constraintBottom_toTopOf="@id/deleteButton"
37 ilova: layout_constraintLeft_toLeftOf="ota-ona"
38 ilova:layout_constraintRight_toRightOf="ota-ona"
39 ilova: layout_constraintTop_toBottomOf="@id/addButton" />
```

```
40
41 <Tugma
42 android:id="@+id/o'chirishTugmasi"
43 android:layout_width="0dp"
44 android:layout_height="wrap_content"
45 android:text="O'chirish"
46 android:onClick="o'chirish"
47 ilova: layout_constraintLeft_toLeftOf="ota-ona"
48 ilova:layout_constraintRight_toRightOf="ota-ona"
49 ilova: layout_constraintTop_toBottomOf="@id/updateButton" />
50
51 </androidx.constraintlayout.widget.ConstraintLayout>
```

Bu yerda do'stlar ro'yxatini ko'rsatish, shuningdek, ularni qo'shish, yangilash va o'chirish uchun tugmalar to'plami belgilangan. Har bir tugma MainActivity sinfidagi mos keladigan usulni chaqiradi.

MainActivity klassi uchun kodni o'zgartiraylik . Soddalashtirish uchun natijalarni **Logcat oynasida Log.d() usuli** yordamida ko'rsatamiz :

```
1  paket com.example.friendsproviderapp;
2
3  import androidx.appcompat.app.AppCompatActivity;
4
5  import android.content.ContentResolver;
6  import android.content.ContentValues;
7  import android.ma'lumotlar bazasi.kursor;
8  import android.net.Uri;
9  import android.os.Bundle;
```

```
10 import android.util.Log;
11 import android.view.View;
12
13 ommaviy sinf MainFaoliyat kengayadi AppCompatActivity {
14
15 xususiyl statik yakuniyl Satr TAG = "Asosiy faoliyat";
16
17 @Override
18 himoyalangan bo'shliql onCreate(saqlanganInstanceState to'plami)
19 super.onCreate(saqlanganInstanceState);
20 setContentView(R.layout.activity_main);
21 }
22 // qabul qilish hamma
23 ommaviyl bo'shliql getAll(Ko'rish ko'rinishi) {
```

```
24 String[] proyeksiyasi = {
25 Do'stlarKontrakt.Ustunlar._ID,
26 Do'stlarKontrakt.Ustunlar.NAME,
27 Do'stlarKontrakt.Ustunlar.EMAIL,
28 Do'stlarKontrakt.Ustunlar.TELEFON
29 };
30 ContentResolver contentResolver = getContentResolver();
31 Cursor kursori =
32 contentResolver.query(FriendsContract.CONTENT_URI, {
33 proyeksiya,
34 nol,
35 nol,
36 Do'stlarKontrakt.Ustunlar.NAME);
37 agar (kursor != null) {
```

```
38 Log.d(TAG, "count:" + cursor.getCount());
39 // sanab o'tish elementlar
40 while(kursor.moveKeyingi()) {
41 uchun(int) i=0; i < cursor.getColumnCount(); i++){
42 Log.d(TAG, cursor.getColumnName(i) + " : " +
43 cursor.getString(i));
44 }
45 Log.d(TAG, "= ...");
46 }
47 kursor.close();
48 }
49 boshqa {
50 Log.d(TAG, "Kursor nolga teng");
51 }
```

```
52 }
53 // Qo'shilmoqda
54 ommaviy bo'shliq qo'shish(Ko'rish ko'rinishi) {
55 ContentResolver contentResolver = getContentResolver();
56 ContentValues qiymatlari = yangi Kontent qiymatlari();
57 values.put(FriendsContract.Columns.NAME, "Sam");
58 values.put(FriendsContract.Columns.EMAIL, "sam@gmail.com");
59 values.put(Do'stlarKontrakt.Ustunlar.TELEFON, "+13676254985");
60 Uri uri = contentResolver.insert(FriendsContract.CONTENT_URI,
61   qiymatlar);
62 Log.d(TAG, "Do'st qo'shildi");
63 }
64
65 // Yangilash
```

```

66 ommaviy bo'shliq yangilash (Ko'rish ko'rinishi) {
67 ContentResolver contentResolver = getContentResolver();
68 ContentValues qiymatlari = yangi Kontent qiymatlari();
69 values.put(FriendsContract.Columns.EMAIL, "sammy@gmail.com");
70 values.put(FriendsContract.Columns.PHONE, "+55555555555");
71 Satr tanlash = FriendsContract.Columns.NAME + " = 'Sam'";
72 int count = contentResolver.update(FriendsContract.CONTENT_URI,
73 qiymatlar, tanlov, null);
74 Log.d(TAG, "Do'st yangilandi");
75 }
76 // O'chirish
77 ommaviy bo'shliq o'chirish (Ko'rish ko'rinishi) {
78 ContentResolver contentResolver = getContentResolver();
79 Satr tanlash = FriendsContract.Columns.NAME + " = ?";

```

```
80 String[] args = {"Sam"};
    int count = contentResolver.delete(FriendsContract.CONTENT_URI,
    tanlash, args);
    Log.d(TAG, "Do'st o'chirildi");
    }
}
```

Keling, ushbu kodda bajarilgan individual harakatlarni ko'rib chiqaylik.



javarush.com

REKLAMA • 16+

Bepul Python kursi

Sizga 5 kun ichida noldan boshlab to'liq Python loyihasini qanday yaratishni bepul o'rgatamiz.

Ma'lumotlarni qabul qilish

```
1 ommaviy bo'shliq getAll(Ko'rish ko'rinishi) {  
2 String[] proyeksiyasi = {  
3 Do'stlarKontrakt.Ustunlar._ID,
```

```
4 Do'stlarKontrakt.Ustunlar.NAME,
5 Do'stlarKontrakt.Ustunlar.EMAIL,
6 Do'stlarKontrakt.Ustunlar.TELEFON
7 };
8 ContentResolver contentResolver = getContentResolver();
9 Cursor kursori =
10 contentResolver.query(FriendsContract.CONTENT_URI, {
11     proyeksiya,
12     nol,
13     nol,
14     Do'stlarKontrakt.Ustunlar.NAME);
15 agar (kursor != null) {
16     Log.d(TAG, "count:" + cursor.getCount());
17     // sanab o'tish elementlar
```

```
18 while(kursor.moveKeyingi()) {
19     uchun(int) i=0; i < kursor.getColumnCount(); i++){
20     Log.d(TAG, kursor.getColumnNames(i) + " : " +
21     kursor.getString(i));
22     }
23     Log.d(TAG, "= ...");
24     }
25     kursor.close();
26     }
27     boshqa {
28     Log.d(TAG, "Kursor nolga teng");
29     }
30 }
```

Kontent provayderi bilan o'zaro aloqa **ContentResolver** obyekti orqali amalga oshiriladi . Ma'lumotlarni olish uchun **query()** usuli chaqiriladi , bu asosan kontent provayderining so'rov usuliga chaqiruvdir. So'rov usuliga uri (ma'lumotlarga yo'l), proyeksiya (olinadigan ustunlar to'plami), tanlov ifodasi va uning parametrlari hamda saralash uchun ustun nomi uzatiladi.

`moveToNext()` usuli yordamida alohida ma'lumotlarni olish uchun takrorlash mumkin . `getColumnName()` usuli ustun nomini qaytaradi va `getString()` ushbu ustunning haqiqiy qiymatini qaytaradi:

Id orqali bitta obyektini olish:

```
1 String[] proyeksiyasi = {  
2 Do'stlarKontrakt.Ustunlar._ID,  
3 Do'stlarKontrakt.Ustunlar.NAME,  
4 Do'stlarKontrakt.Ustunlar.EMAIL,
```

```
5  Do'stlarKontrakt.Ustunlar.TELEFON
6  };
7  ContentResolver contentResolver = getContentResolver();
8  Kursor kursori =
9  contentResolver.query(FriendsContract.buildFriendUri(2),
10 proyeksiya, nol, nol, FriendsContract.Columns.NAME);
11 agar (kursor != null) {
12 while(kursor.moveKeyingi()) {
13 uchun(int) i=0; i < cursor.getColumnCount(); i++){
14 Log.d(TAG, cursor.getColumnName(i) + " : " +
15 cursor.getString(i));
16 }
17 kursor.close();
```

```
}
```

Bu holda biz `_id=2` ga ega obyektни olamiz.

Ma'lumotlar qo'shilmoqda

Ma'lumotlar qo'shilmoqda:

```
1 ContentResolver contentResolver = getContentResolver();  
2 ContentValues qiymatlari = yangi Kontent qiymatlari();  
3 values.put(FriendsContract.Columns.NAME, "Sam");  
4 values.put(FriendsContract.Columns.EMAIL, "sam@gmail.com");  
5 values.put(Do'stlarKontrakt.Ustunlar.TELEFON, "+13676254985");  
6 Uri uri = contentResolver.insert(FriendsContract.CONTENT_URI,  
  qiymatlar);
```

Qo'shish uchun URI yo'lini va ContentValues shaklida qo'shiladigan ma'lumotlarni qabul qiladigan `insert` usulidan foydalaning.

Ma'lumotlar yangilanmoqda

Ma'lumotlarni yangilash:

```
1 ContentResolver contentResolver = getContentResolver();
2 ContentValues qiymatlari = yangi Kontent qiymatlari();
3 values.put(FriendsContract.Columns.EMAIL, "sammy@gmail.com");
4 values.put(FriendsContract.Columns.PHONE, "+555555555555");
5 Satr tanlash = FriendsContract.Columns.NAME + " = 'Sam'";
6 int count = contentResolver.update(FriendsContract.CONTENT_URI,
  qiymatlar, tanlov, null);
```

Bu holda, "Name=Sam" ga ega barcha obyektlar uchun ma'lumotlar yangilanadi. Yangilash mezonlari uchinchi parametr orqali uzatiladi.

Tabiiyki, SQL ifodasidan foydalanib, biz yangilash uchun obyektlarni tanlash uchun istalgan mantiqni belgilashimiz mumkin. Va qulayroq bo'lishi uchun biz

savol belgisi bilan ko'rsatilgan parametrlar yordamida ma'lumotlarni kiritishimiz mumkin:

```
1 ContentResolver contentResolver = getContentResolver();
2 ContentValues qiymatlari = yangi Kontent qiymatlari();
3 values.put(FriendsContract.Columns.NAME, "Sam");
4 Satr tanlash = FriendsContract.Columns.NAME + " = ?";
5 Satr args[] = {"Sam Scromby"};
6 int count = contentResolver.update(FriendsContract.CONTENT_URI,
  qiymatlar, tanlov, argumentlar);
```

Bu holda, to'rtinchi parametr tanlov ifodasi parametrlari uchun qiymatlar massivini uzatish uchun ishlatiladi.

Lekin yuqoridagi misollarda, masalan, "Sam" nomiga ega bo'lgan ma'lumotlar bazasidagi barcha qatorlar yangilandi. Biroq, bitta obyektни ID orqali yangilash ham mumkin. Masalan, qatorni `_id=3` bilan yangilaylik:

```
ContentResolver contentResolver = getContentResolver();
1 ContentValues qiymatlari = yangi Kontent qiymatlari();
2 values.put(FriendsContract.Columns.NAME, "Sam");
3 values.put(FriendsContract.Columns.EMAIL, "sam@gmail.com");
4 int count =
5 contentResolver.update(FriendsContract.buildFriendUri(3),
    qiymatlar, null, null);
```

Ma'lumotlarni o'chirish

Umumiy shartlarga muvofiq ma'lumotlarni o'chirish:

```
1 ContentResolver contentResolver = getContentResolver();
2 Satr tanlash = FriendsContract.Columns.NAME + " = ?";
3 String[] args = {"Sam"};
4 int count = contentResolver.delete(FriendsContract.CONTENT_URI,
    tanlash, args);
```

Bu holda, Name=Sam bilan belgilangan barcha qatorlar o'chiriladi.

ID bo'yicha o'chirish:

```
ContentResolver contentResolver = getContentResolver();  
1 int count =  
2 contentResolver.delete(FriendsContract.buildFriendUri(2), null,  
    null);
```

Bu holda, _id=2 bo'lgan qator o'chiriladi.