Лабораторная работа №4. Использование БД в Android приложениях

1. **Перечислите типы данных использующиеся в SQLite.**
   - INTEGER: представляет целое число, аналог типу Int в kotlin
   - REAL: представляет число с плавающей точкой, аналог float и Double в kotlin
   - TEXT: представляет набор символов, аналог String в kotlin
   - BLOB: представляет массив бинарных данных, например, изображение
2. **Опишите процесс создания базы данных SQLite.**
   Для создания или открытия новой базы данных из кода Activity в Android можно вызвать метод openOrCreateDatabase(). Этот метод может принимать три параметра:
   - название для базы данных;
   - числовое значение, которое определяет режим работы (как правило, в виде константы MODE_PRIVATE);
   - необязательный параметр в виде объекта SQLiteDatabase.CursorFactory, который представляет фабрику создания курсора для работы с БД.

   Например, создание базы данных app.db:
   var db = baseContext.openOrCreateDatabase("app.db", Context.MODE_PRIVATE, null)
3. **Опишите процесс открытия базы данных SQLite.**
   Смотри второй вопрос
4. **Перечислите операции с данными таблицы.**
   К основным операциям с данными относятся вставка, обновление и удаление данных. Для выполнения этих SQLiteDatabase имеет методы insert(), update() и delete().
   - Метод insert(): long insert (String table, String nullColumnHack, ContentValues values);
     ContentValues cv = new ContentValues();
     cv.put("name", name);
     cv.put("age", age);
     long rowID = db.insert("mytable", null, cv);
   - Метод update(): int update (String table, ContentValues values, String whereClause, String[] whereArgs)
     int updCount = db.update("mytable", cv, "id = ?", new String[] { id });
   - Метод delete(): int delete (String table, String whereClause, String[] whereArgs)
     int delCount = db.delete("mytable", "id = " + id, null);
5. **Опишите механизм работы со списками.**
   При работе со списками имеют дело с тремя компонентами. Во-первых, это сами элементы списков (ListView, GridView), которые отображают данные. Во-вторых, это источник данных - массив, объект ArrayList, база данных и т.д., в котором находятся сами отображаемые данные. И в-третьих, это адаптеры – специальные компоненты, которые связывают источник данных с элементом списка.
   ArrayAdapter<String>(this,android.R.layout.simple_list_item_1, Array<String>)
   - *this* – текущий объект activity,
   - *android.R.layout.simple_list_item_1* - файл разметки списка,
   - *Array<String>* - массив данных. Здесь необязательно указывать именно массив, это может быть список ArrayList<T>.

   В конце необходимо установить для ListView адаптер с помощью метода *setAdapter()*.
6. **Опишите механизм работы с файлами?**
   Приложение Android сохраняет свои данные в каталоге /data/data/<название_пакета>/ и, как правило, относительно этого каталога будет идти работа.
   Для работы с файлами абстрактный класс android.content.Context определяет ряд методов:
   - deleteFile(String name): удаляет определенный файл

- fileList(): получает все файлы, которые содержатся в подкаталоге /files в каталоге приложения
- getDir(String dirName, int mode): получает ссылку на подкаталог в каталоге приложения, если такого подкаталога нет, то он создается
- getFileStreamPath(String filename): возвращает абсолютный путь к файлу в файловой системе
- openFileInput(String filename): открывает файл для чтения
- openFileOutput (String name, int mode): открывает файл для записи

Все файлы, которые создаются и редактируются в приложении, как правило, хранятся в подкаталоге /files в каталоге приложения.

При создании файла можно оперировать двумя разными режимами:

- MODE_PRIVATE: файлы могут быть доступны только владельцу приложения (режим по умолчанию)
- MODE_APPEND: данные могут быть добавлены в конец файла

BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(openFileOutput(FILENAME, MODE_PRIVATE)));
bw.write("Содержимое файла");
bw.close();
BufferedReader br = new BufferedReader(new InputStreamReader(openFileInput(FILENAME)));
str = br.readLine()

7. **Раскройте понятие LogCat.**
   Самый важный инструмент при отладке – это LogCat. Он отображает сообщения логов (журнал логов), рассылаемые при помощи различных методов.
8. **Перечислите категории сообщений LogCat.**
   - Log.e() - ошибки (error)
   - Log.w() - предупреждения (warning)
   - Log.i() - информация (info)
   - Log.d() - отладка (debug)
   - Log.v() - подробности (verbose)
9. **Раскройте понятие ArrayAdapter.**
   Класс ArrayAdapter представляет собой простейший адаптер, который связывает массив данных с набором элементов TextView, из которых, к примеру, может состоять ListView. То есть в данном случае источником данных выступает массив объектов. ArrayAdapter вызывает у каждого объекта метод toString() для приведения к строковому виду и полученную строку устанавливает в элемент TextView.
   ArrayAdapter<String>(this,android.R.layout.simple_list_item_1, Array<String>)
   - *this* – текущий объект activity,
   - *android.R.layout.simple_list_item_1* - файл разметки списка,
   - *Array<String>* - массив данных. Здесь необязательно указывать именно массив, это может быть список ArrayList<T>.

Лабораторная работа №5. Использование списков и компонента GridView

1. **Раскройте понятие компонент в Android приложении.**
   Компоненты являются основными строительными блоками Android приложений. Каждый компонент является отдельной точкой входа в разработанное приложение. Не все компоненты являются фактически точками входа для пользователя, некоторые из них зависят друг от друга, но каждый из компонентов представляет из себя уникальный строительный блок, который помогает определить поведение приложения в системе.
2. **Перечислите виды компонентов.**

- Активности (Activities) – представляют собой единый экран с пользовательским интерфейсом
- Сервисы (Services) – компонент, работающий в фоновом режиме и предназначенный для выполнения длительных операций или удаленных процессов. Сервис не предоставляет пользовательский интерфейс
- Поставщики содержимого (Content Provider) – управляют общими данными в приложении. Через поставщики содержимого, другие приложения могут получить или модифицировать данные вашего приложения (если это разрешено)
- Широковещательные приемники (Broadcast Reciever) – компонент, который реагирует на общесистемное оповещение

3. **Опишите механизм запуска одним приложением компонентов другого**

   Когда система запускает компонент, запускается процесс для приложения (если оно не было уже запущено) и создаются экземпляры классов, необходимые для работы компонента. Например, если приложение запустило активность другого приложения, это активность будет работать в процессе другого приложения. Поэтому, в отличие от приложений в большинстве других систем, Android приложения не имеют единой точки входа в программу (например, нет основной функции main()).

   Поскольку система запускает каждое приложение в отдельном процессе с разрешением для файлов, которое ограничивает доступ других приложений, приложение не может напрямую активировать компоненты другого приложения. Но это может сделать система Android. Таким образом, чтобы запустить компонент другого приложения, следует передать в систему сообщение, которое определяет намерение пользователя для запуска конкретного компонента.

4. **Опишите роль компонента ExpandableListView.**

   Двухуровневый список ExpandableListView представляет собой объект view, в котором каждый элемент списка содержит вложенный список.

5. **Опишите реализацию ExpandableListView.**

   За работу списка отвечает адаптер ExpandableListViewAdapter, который загружает данные по каждому пункту и вложенному списку.

   Рассмотрим наиболее полезные методы, которые используются при работе с классом ExpandableListView:

   - Метод setChildIndicator(Drawable) используется для отображения индикатора возле каждого элемента списка.
   - Метод setGroupIndicator(Drawable) устанавливает индикатор возле каждого элемента, обозначая развернут он или свернут. Если группа пуста, то будет установлено состояние state_empty. Если группа развернута, будет установлено состояние state_expanded.
   - Метод getGroupView() возвращает view для группы элементов списка
   - Метод getChildView() возвращает view дочернего элемента списка

   Для каждого элемента используется свой тип слушателя:

   - Интерфейс ExpandableListView.OnChildClickListener переопределяется для отслеживания нажатий на дочерние элементы раскрывающегося списка
   - Интерфейс ExpandableListView.OnGroupClickListener переопределяется для отслеживания нажатий на группу элементов
   - Интерфейс ExpandableListView.OnGroupCollapseListener используется для уведомления о том, что группа элементов была свернута
   - Интерфейс ExpandableListView.OnGroupExpandListener используется для уведомления о том, что группа элементов была развернута

6. **Опишите роль компонента Spinner.**

Компонент Spinner из раздела Widgets похож на выпадающий список (ComboBox), используемый в ОС Windows. В закрытом состоянии компонент показывает одну строчку, при раскрытии выводит список в виде диалогового окна с переключателями

7. **Опишите реализацию Spinner.**

var cities = arrayOf("Москва", "Самара", "Вологда", "Волгоград", "Саратов", "Воронеж")
val spinner = findViewById(R.id.cities) as Spinner
val adapter = ArrayAdapter(this, android.R.layout.simple_spinner_item, cities)
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item)
spinner.adapter = adapter

8. **Опишите роль компонента GridView.**

Компонет GridView представляет собой плоскую таблицу. Для GridView можно использовать собственные поля для отображения элементов данных, создав класс, производный от класса ArrayAdapter или BaseAdapter и т.п, и переопределив его метод getView().

9. **Опишите реализацию GridView.**

Число столбцов для GridView чаще задается статически. Число строк в элементе определяется динамически на основании числа элементов, которые предоставляет адаптер.

Существует следующий набор свойств:

- android:numColumns — определяет количество столбцов. Если поставлено значение auto_fit, то система вычислит количество столбцов, основанное на доступном пространстве
- android:verticalSpacing — устанавливает размер пустого пространства между ячейками таблицы
- android:columnWidth — устанавливает ширину столбцов
- android:stretchMode — указывает, куда распределяется остаток свободного пространства для таблицы с установленным значением android:numColumns="auto_fit". Принимает значения columnWidth для распределения остатка свободного пространства между ячейками столбцов для их увеличения или spacingWidth — для увеличения пространства между ячейками

```
var countries = arrayOf("Бразилия", "Аргентина", "Чили", "Колумбия", "Уругвай",
"Парагвай")
val countriesList = findViewById<GridView>(R.id.gridview)
val adapter = ArrayAdapter(this, android.R.layout.simple_list_item_1, countries)
countriesList.adapter = adapter
gridView.numColumns = numColumns
 gridView.horizontalSpacing = horizontalSpacing
val itemListener = AdapterView.OnItemClickListener {
        parent, v, position, id ->
                Toast.makeText(
                        applicationContext, "Вы выбрали " +
                                parent.getItemAtPosition(position).toString(),
                        Toast.LENGTH_SHORT
                ).show()
 }
 countriesList.onItemClickListener = itemListener
```

Лабораторная работа №6. Создание и использование собственных источников данных. Работа со стандартными источниками данных

1. **Раскройте значение термина ContentProvider.**

Контент-провайдер или "Поставщик содержимого" (ContentProvider) - это оболочка (wrapper), в которую заключены данные. В Android существует возможность выражения источников данных (или поставщиков данных) при помощи передачи состояния представления - REST, в виде абстракций, называемых поставщиками содержимого. Базу данных SQLite можно заключить в поставщик содержимого. Чтобы получить данные из поставщика содержимого или сохранить в нём новую информацию, нужно использовать набор REST-подобных идентификаторов URI.

2. **Дайте определение URI.**
   URI (Uniform Resource Identifier) — унифицированный (единообразный) идентификатор ресурса.
   URI – символьная строка, позволяющая идентифицировать какой-либо ресурс: документ, изображение, файл, службу, ящик электронной почты и т. д.
3. **Опишите механизм создания собственного контент-провайдера.**
   1) Для создания собственного контент-провайдера нужно унаследоваться от абстрактного класса ContentProvider.
   2) В классе необходимо реализовать абстрактные методы query(), insert(), update(), delete(), getType(), onCreate().
   3) Следует зарегистрировать контент-провайдер в манифесте с помощью тега provider с атрибутами name и authorities. Тег authorities служит для описания базового пути URI, по которому ContentResolver может найти базу данных для взаимодействия. Данный тег должен быть уникальным, поэтому рекомендуется использовать имя вашего пакета, чтобы не произошло путаницы с другими приложениями, например:
      <provider android:name:".MyContentProvider"
      android:authorities="ru.anything.provider.notepad"/>
4. **Перечислите методы, которые необходимо реализовать в классе, унаследованном от ContentProvider.**
   В классе необходимо реализовать абстрактные методы query(), insert(), update(), delete(), getType(), onCreate().
5. **Приведите пример встроенных контент-провайдеров.**
   - Browser. Используйте этот Источник данных для чтения или изменения закладок, истории посещений или использования поиска в обозревателе.
   - CallLog. ыводит или обновляет историю звонков (входящие, исходящие, пропущенные), открывает доступ к детальной информации, такой как номер звонившего и продолжительность разговора.
   - ContactsContract. Используйте этот Источник данных для получения, изменения или хранения информации о контактах. Он заменяет старый класс Contacts.
   - MediaStore. Этот Источник данных предоставляет централизованный, управляемый доступ к файлам мультимедиа на устройстве, включая аудио, видео и изображения.
   - Settings. Вы можете получить доступ к настройкам устройства с помощью этого Источника данных.
   - UserDictionary. Обеспечивает доступ к пользовательским наборам слов, добавленных в словарь для интеллектуального ввода текста.
6. **Опишите использование источника данных MediaStore.**
   Чтобы получить доступ к медиафайлам из MediaStore, нужно запрашивать их через Источники данных. Класс MediaStore включает подклассы Audio, Video и Images, которые, в свою очередь, содержат подклассы, обеспечивающие доступ к именам столбцов и путям URI, ссылающимся на содержимое каждого Источника данных.
   var cursor = getContentResolver().query(

```
        MediaStore.Audio
                .Media.EXTERNAL_CONTENT_URI, null, null, null
)
val albumIdx = cursor.getColumnIndexOrThrow(MediaStore.Audio.Media. ALBUM)
val titleIdx = cursor. getColumnIndexOrThrow(MediaStore.Audio.Media. TITLE)
String[] result = new String[cursor.getCount()];
if (cursor.moveToFirst())
        do {
                String title = cursor.getString(titleIdx);
                String album = cursor.getString(albumIdx);
                result[cursor.getPosition()] = title + " (" + album + ")";
} while(cursor.moveToNext());
cursor.close();
```

7. **Опишите принцип работы с контактами.**

   Контакты в Android обладают встроенным API, который позволяет получать и изменять список контактов. Все контакты хранятся в базе данных SQLite, однако они не представляют единой таблицы. Для контактов отведено три таблицы, связанных отношением один-ко-многим: таблица для хранения информации о людях, таблица их телефонов и таблица адресов их электронных почт. Но благодаря Android API мы можем абстрагироваться от связей между таблицами.

```
ArrayList<String> contacts = new ArrayList<String>();
ContentResolvercontentResolver = getContentResolver();
Cursor cursor = contentResolver.query(ContactsContract.Contacts
        .CONTENT_URI, null, null, null, null);
if(cursor!=null) {
        while (cursor.moveToNext()) {
                String contact = cursor.getString(cursor.getColumnIndex(
                        ContactsContract.Contacts.DISPLAY_NAME_PRIMARY));
                contacts.add(contact);
}
cursor.close();
```

8. **Приведите примеры основных операций с контактами.**

   - Чтение контактов

     Чтобы получить доступ к любой контактной информации, необходимо добавить полномочие READ_CONTACTS к манифесту вашего приложения:

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

     Пример кода для чтения контактов смотри в седьмом вопросе.

   - Добавление контактов

     Кроме создания запросов к базе данных контактов можно использовать эти Источники данных для изменения, удаления или вставки записей о контактах, добавив полномочие android.permission.WRITE_CONTACTS в манифест приложения.

```
CONTACT_URI = ContactsContract.Contacts.CONTENT_URI
ContentValues cv = new ContentValues();
cv.put(CONTACT_NAME, "name 4");
cv.put(CONTACT_EMAIL, "email 4");
Uri newUri = getContentResolver().insert(CONTACT_URI, cv);
```

   - Обновление

```
CONTACT_URI = ContactsContract.Contacts.CONTENT_URI
ContentValues cv = new ContentValues();
```

cv.put(CONTACT_NAME, "name 5");
cv.put(CONTACT_EMAIL, "email 5");
Uri uri = ContentUris.withAppendedId(CONTACT_URI, 2);
int cnt = getContentResolver().update(uri, cv, null, null);

- Удаление
  CONTACT_URI = ContactsContract.Contacts.CONTENT_URI
  Uri uri = ContentUris.withAppendedId(CONTACT_URI, 3);
  int cnt = getContentResolver().delete(uri, null, null);