Класс **DBHelper**является **вложенным**в **MainActivity,**  этот класс должен наследовать класс **SQLiteOpenHelper**.

В **конструкторе**мы вызываем конструктор суперкласса и передаем ему

**class DBHelper extends SQLiteOpenHelper {**

**public DBHelper(Context context) {**

**// конструктор суперкласса**

**super(context, "myDB", null, 1);**

**}**

**public void onCreate(SQLiteDatabase db){. . . }**

**public void onUpgrade(SQLiteDatabase db,   
 int oldVersion, int newVersion) {…}**

**}**

**context**- контекст  
*mydb*- название базы данных  
**null**– объект для работы с курсорами, нам пока не нужен, поэтому null  
*1* – версия базы данных

// создаем объект для создания и управления версиями БД

**dbHelper = new DBHelper(this);**

// подключаемся к БД

**SQLiteDatabase db = dbHelper.getWritableDatabase();**

    // создаем объект для работы с данными

**ContentValues cv = new ContentValues();**

**name = "Петя";**

**email= "petr@mail.ru";**

**cv.put("name", name);**

**cv.put("email", email);**

      // вставляем запись и получаем ее ID

**long rowID = db.insert("mytable", null, cv);**

метод [insert](http://developer.android.com/reference/android/database/sqlite/SQLiteDatabase.html" \l "insert(java.lang.String, java.lang.String, android.content.ContentValues)" \t "_blank) – передаем ему **имя таблицы**и объект **cv**с вставляемыми значениями. Второй аргумент метода используется, при вставке в таблицу пустой строки. Нам это сейчас не нужно, поэтому передаем null. Метод **insert**возвращает **ID**вставленной строки.

**insert**

added in [API level 1](https://developer.android.com/guide/topics/manifest/uses-sdk-element.html#ApiLevels)

public long insert ([String](https://developer.android.com/reference/java/lang/String.html) table,

[String](https://developer.android.com/reference/java/lang/String.html) nullColumnHack,

[ContentValues](https://developer.android.com/reference/android/content/ContentValues.html) values)

|  |  |
| --- | --- |
| table | String: the table to insert the row into |
| nullColumnHack | String: необязательно; может быть null. SQL не позволяет вставлять пустые строки без указания хотя бы одного имени столбца. Если указанные значения пусты, имена столбцов неизвестны и пустая строка не может быть вставлена. Если не NULL, то параметр nullColumnHack содержит имя столбца значение null явно вставить значение null в случае, когда ваши поля пусты. |
| values | ContentValues: эта карта содержит начальные значения столбцов для строки. Ключами должны быть имена столбцов, а значениями-значения столбцов |
| RETURN | |
| long | the row ID of the newly inserted row, or -1 if an error occurred |

// делаем запрос всех данных из таблицы mytable, получаем Cursor

**query**

**Cursor c = db.query("mytable", null, null, null, null, null, null);**

public [Cursor](https://developer.android.com/reference/android/database/Cursor.html) **query** ([String](https://developer.android.com/reference/java/lang/String.html) table, имя таблицы

[String[]](https://developer.android.com/reference/java/lang/String.html) columns, Список возвращаемых столбцов. Передача null вернет все столбцы, что не рекомендуется для предотвращения чтения данных из хранилища, которое не будет использоваться.

[String](https://developer.android.com/reference/java/lang/String.html) selection, Фильтр, объявляющий возвращаемые строки, отформатированный как предложение SQL WHERE (исключая само WHERE). Передача null вернет все строки для данной таблицы.

[String[]](https://developer.android.com/reference/java/lang/String.html) selectionArgs, Вы можете включить ?s в предыдущем фильтре, которое будет заменено значениями из selectionArgs, для того, чтобы они появились в выделении. Значения будут связаны как строки

[String](https://developer.android.com/reference/java/lang/String.html) groupBy, Фильтр, объявляющий, как группировать строки, отформатированные как предложение SQL GROUP BY (исключая саму группу). Передача null приведет к строки не группируются.

[String](https://developer.android.com/reference/java/lang/String.html) having, Фильтр объявляет, какие группы строк включать в курсор, если используется группировка строк, отформатированные как предложение SQL HAVING (за исключением самого HAVING). Передача null приведет к включению всех групп строк и является обязательной, если группировка строк не используется.

[String](https://developer.android.com/reference/java/lang/String.html) orderBy)

ПРИМЕРЫ:

1. // проверка существования записей

**Cursor c = db.query("mytable", null, null, null, null, null, null);**

**if (c.getCount() == 0)** {

1. **columns = new String[] { sFunc };**

**c = db.query("mytable", columns, null, null, null, null, null);**

1. // Население больше, чем

**sPeople = "120000";**

**selection = "people > ?";**

**selectionArgs = new String[] { sPeople };**

**c = db.query("mytable", null, selection, selectionArgs, null, null,null);**

// Население по региону

**columns = new String[] { "region", "sum(people) as people" };**

**groupBy = "region";**

**c = db.query("mytable", columns, null, null, groupBy, null, null);**

// Население по региону больше чем

**columns = new String[] { "region", "sum(people) as people" };**

**groupBy = "region";**

**having = "sum(people) > " + sRegionPeople;**

**c = db.query("mytable", columns, null, null, groupBy, having, null);**

**update**

cv.put("name", name);

cv.put("email", email);

     // обновляем по id

**int updCount = db.update("mytable", cv, "id = ?",   
 new String[] { id });**

public int update ([String](https://developer.android.com/reference/java/lang/String.html) table,

[ContentValues](https://developer.android.com/reference/android/content/ContentValues.html) values,

[String](https://developer.android.com/reference/java/lang/String.html) whereClause,

[String[]](https://developer.android.com/reference/java/lang/String.html) whereArgs)

|  |  |
| --- | --- |
| table | String: the table to update in |
| values | ContentValues: сопоставление имен столбцов с новыми значениями столбцов. null-допустимое значение, которое будет преобразовано в NULL. |
| whereClause | String: необязательное предложение WHERE для применения при обновлении. Передача null обновит все строки. |
| whereArgs | String: Вы можете включить ?s в предложении where, которое будет заменено значениями из whereArgs. Значения будут связаны как строки. |
| Returns | |
| int | the number of rows affected |

**Перебор значений в базе**

// ставим позицию курсора на первую строку выборки

      // если в выборке нет строк, вернется false

      if (c.moveToFirst()) {

        // определяем номера столбцов по имени в выборке

        int idColIndex = c.getColumnIndex("id");

        int nameColIndex = c.getColumnIndex("name");

        int emailColIndex = c.getColumnIndex("email");

        do {

          // получаем значения по номерам столбцов и пишем все в лог

          Log.d(LOG\_TAG,

              "ID = " + c.getInt(idColIndex) +

              ", name = " + c.getString(nameColIndex) +

              ", email = " + c.getString(emailColIndex));

          // переход на следующую строку

   // а если следующей нет (текущая - последняя), то false - выходим из цикла

        } while (c.moveToNext());

      // удаляем все записи

      int clearCount = db.delete("mytable", null, null);

или

int delCount = db.delete("mytable", "id = " + id, null);

**delete**

added in [API level 1](https://developer.android.com/guide/topics/manifest/uses-sdk-element.html#ApiLevels)

public int delete ([String](https://developer.android.com/reference/java/lang/String.html) table,

[String](https://developer.android.com/reference/java/lang/String.html) whereClause,

[String[]](https://developer.android.com/reference/java/lang/String.html) whereArgs)

|  |  |
| --- | --- |
| table | String: the table to delete from |
| whereClause | String: the optional WHERE clause to apply when deleting. Passing null will delete all rows. |
| whereArgs | String: You may include ?s in the where clause, which will be replaced by the values from whereArgs. The values will be bound as Strings. |
| Returns | |
| int | the number of rows affected if a whereClause is passed in, 0 otherwise. To remove all rows and get a count pass "1" as the whereClause. |

**Запросы из связанных таблиц**

Запрос на выборку ОДНОСТРОЧНЫЙ

// выводим результат объединения

    // используем rawQuery

**String sqlQuery = "select PL.name as Name, PS.name as Position, salary as Salary "**

**+ "from people as PL "**

**+ "inner join position as PS "**

**+ "on PL.posid = PS.id "**

**+ "where salary > ?";**

**c = db.rawQuery(sqlQuery, new String[] {"12000"});**

**c.close();**

**rawQuery**

public [Cursor](https://developer.android.com/reference/android/database/Cursor.html) rawQuery ([String](https://developer.android.com/reference/java/lang/String.html) sql,

[String[]](https://developer.android.com/reference/java/lang/String.html) selectionArgs)

Runs the provided SQL and returns a [Cursor](https://developer.android.com/reference/android/database/Cursor.html) over the result set.

|  |  |
| --- | --- |
| sql | String: the SQL query. The SQL string must not be ; terminated |
| selectionArgs | String: You may include ?s in where clause in the query, which will be replaced by the values from selectionArgs. The values will be bound as Strings. |
| Returns | |
| [Cursor](https://developer.android.com/reference/android/database/Cursor.html) | A [Cursor](https://developer.android.com/reference/android/database/Cursor.html) object, which is positioned before the first entry. Note that [Cursor](https://developer.android.com/reference/android/database/Cursor.html)s are not synchronized, see the documentation for more details. |

Тот же запрос на выборку, но функция **query**

**// используем query**

**String table = "people as PL inner join position as PS on PL.posid = PS.id";**

**String columns[] = { "PL.name as Name", "PS.name as Position",   
 "salary as Salary" };**

**String selection = "salary < ?";**

**String[] selectionArgs = {"12000"};**

**c = db.query(table, columns, selection, selectionArgs, null, null, null);**

**c.close();**

**// закрываем БД**

**dbh.close();**

**Вывод данных из курсора**

  // вывод в лог данных из курсора

**void logCursor(Cursor c) {**

**if (c != null) {**

**if (c.moveToFirst()) {**

**String str;**

**do {**

**str = "";**

**//** **for** ( x=0; x< c.getColumnCount(); x++)

**for (String cn : c.getColumnNames())//**это аналог foreach для java

**{**

**str = str.concat(cn + " = " + c.getString(c.getColumnIndex(cn)) + "; ");**

**}**

**Log.d(LOG\_TAG, str);**

**} while (c.moveToNext());**

**}**

**} else**

**Log.d(LOG\_TAG, "Cursor is null");**

**}**