**How to delete or add column in SQLITE?**

|  |
| --- |
| I want to delete or add column in sqlite database  I am using following query to delete column: ALTER TABLE TABLENAME DROP COLUMN DIST\_TYPE  But it gives error System.Data.SQLite.SQLiteException: SQLite error near "DROP": syntax error |

|  |
| --- |
| [ALTER TABLE SQLite](http://www.sqlite.org/lang_altertable.html)  SQLite supports a limited subset of ALTER TABLE. The ALTER TABLE command in SQLite allows the user to rename a table or to add a new column to an existing table. It is not possible to rename a column, remove a column, or add or remove constraints from a table.  You can:   1. create new table as the one you are trying to change, 2. copy all data, 3. drop old table, 4. rename the new one. |
| **I've wrote a Java implementation based on the Sqlite's recommended way to do this:**  private void dropColumn(SQLiteDatabase db,  ConnectionSource connectionSource,  String createTableCmd,  String tableName,  String[] colsToRemove) throws java.sql.SQLException {  List<String> updatedTableColumns = getTableColumns(tableName);  // Remove the columns we don't want anymore from the table's list of columns  updatedTableColumns.removeAll(Arrays.asList(colsToRemove));  String columnsSeperated = TextUtils.join(",", updatedTableColumns);  db.execSQL("ALTER TABLE " + tableName + " RENAME TO " + tableName + "\_old;");  // Creating the table on its new format (no redundant columns)  db.execSQL(createTableCmd);  // Populating the table with the data  db.execSQL("INSERT INTO " + tableName + "(" + columnsSeperated + ") SELECT "  + columnsSeperated + " FROM " + tableName + "\_old;");  db.execSQL("DROP TABLE " + tableName + "\_old;");  }  To get the table's column, I used the "PRAGMA table\_info":  public List<String> getTableColumns(String tableName) {  ArrayList<String> columns = new ArrayList<String>();  String cmd = "pragma table\_info(" + tableName + ");";  Cursor cur = getDB().rawQuery(cmd, null);  while (cur.moveToNext()) {  columns.add(cur.getString(cur.getColumnIndex("name")));  }  cur.close();  return columns;  }  I actually wrote about it on my blog, you can see more explanations there:  <http://udinic.wordpress.com/2012/05/09/sqlite-drop-column-support/> | |

|  |  |  |  |
| --- | --- | --- | --- |
|  | <http://www.sqlite.org/lang_altertable.html>  As you can see in the diagram, only ADD COLUMN is supported. There is a (kinda heavy) workaround, though: <http://www.sqlite.org/faq.html#q11> | | |
| As others have pointed out It is not possible to rename a column, remove a column, or add or remove constraints from a table.  source : <http://www.sqlite.org/lang_altertable.html>  While you can always create a new table and then drop the older one. I will try to explain [this workaround](http://www.sqlite.org/faq.html#q11) with an example.  sqlite> .schema  CREATE TABLE person(  id INTEGER PRIMARY KEY,  first\_name TEXT,  last\_name TEXT,  age INTEGER,  height INTEGER  );  sqlite> select \* from person ;  id first\_name last\_name age height  ---------- ---------- ---------- ---------- ----------  0 john doe 20 170  1 foo bar 25 171  Now you want to remove the column height from this table.  Create another table called new\_person  sqlite> CREATE TABLE new\_person(  ...> id INTEGER PRIMARY KEY,  ...> first\_name TEXT,  ...> last\_name TEXT,  ...> age INTEGER  ...> ) ;  sqlite>  Now copy the data from the old table  sqlite> INSERT INTO new\_person  ...> SELECT id, first\_name, last\_name, age FROM person ;  sqlite> select \* from new\_person ;  id first\_name last\_name age  ---------- ---------- ---------- ----------  0 john doe 20  1 foo bar 25  sqlite>  Now Drop the person table and rename new\_person to person  sqlite> DROP TABLE IF EXISTS person ;  sqlite> ALTER TABLE new\_person RENAME TO person ;  sqlite>  So now if you do a .schema, you will see  sqlite>.schema  CREATE TABLE "person"(  id INTEGER PRIMARY KEY,  first\_name TEXT,  last\_name TEXT,  age INTEGER  ); | |
| As others have pointed out, sqlite's ALTER TABLE statement does *not* support DROP COLUMN, and the standard recipe to do this does not preserve constraints & indices.  Here's some python code to do this generically, while *maintaining* all the key constraints and indices.  **Please back-up your database before using!** This function relies on doctoring the original CREATE TABLE statement and is potentially a bit unsafe - for instance it will do the wrong thing if an identifier contains an embedded comma or parenthesis.  If anyone would care to contribute a better way to parse the SQL, that would be great!  ***UPDATE*** I found a better way to parse using the open-source sqlparse package. If there is any interest I will post it here, just leave a comment asking for it ...  import re  import random  def DROP\_COLUMN(db, table, column):  columns = [ c[1] for c in db.execute("PRAGMA table\_info(%s)" % table) ]  columns = [ c for c in columns if c != column ]  sql = db.execute("SELECT sql from sqlite\_master where name = '%s'"  % table).fetchone()[0]  sql = format(sql)  lines = sql.splitlines()  findcol = r'\b%s\b' % column  keeplines = [ line for line in lines if not re.search(findcol, line) ]  create = '\n'.join(keeplines)  create = re.sub(r',(\s\*\))', r'\1', create)  temp = 'tmp%d' % random.randint(1e8, 1e9)  db.execute("ALTER TABLE %(old)s RENAME TO %(new)s" % {  'old': table, 'new': temp })  db.execute(create)  db.execute("""  INSERT INTO %(new)s ( %(columns)s )  SELECT %(columns)s FROM %(old)s  """ % {  'old': temp,  'new': table,  'columns': ', '.join(columns)  })  db.execute("DROP TABLE %s" % temp)  def format(sql):  sql = sql.replace(",", ",\n")  sql = sql.replace("(", "(\n")  sql = sql.replace(")", "\n)")  return sql | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| As SQLite has limited support to ALTER TABLE so you can only ADD column at end of the table OR CHANGE TABLE\_NAME in SQLite.  Here is the Best Answer of HOW TO DELETE COLUMN FROM SQLITE? | | | | |
| You can use the SQlite Administrator for changing the column names. Right Click on Table name and select Edit Table.Here you will find the table structure and you can easily rename it. | | | |
| As an alternative:  If you have a table with schema  CREATE TABLE person(  id INTEGER PRIMARY KEY,  first\_name TEXT,  last\_name TEXT,  age INTEGER,  height INTEGER  );  you can use a CREATE TABLE...AS statement like CREATE TABLE person2 AS SELECT id, first\_name, last\_name, age FROM person;, i.e. leave out the columns you don't want. Then drop the original person table and rename the new one.  Note this method produces a table has no PRIMARY KEY and no constraints. To preserve those, utilize the methods others described to create a new table, or use a [temporary table](http://sqlite.org/faq.html#q11) as an intermediate. | |
|  | add a comment | | | |
| This answer to a different question is oriented toward *modifying* a column, but I believe a portion of the answer could also yield a useful approach if you have lots of columns and don't want to retype most of them by hand for your INSERT statement:  <http://stackoverflow.com/a/10385666>  You could dump your database as described in the link above, then grab the "create table" statement and an "insert" template from that dump, then follow the instructions in the SQLite FAQ entry "How do I add or delete columns from an existing table in SQLite." (FAQ is linked elsewhere on this page.) | | |

|  |
| --- |
| you can use Sqlitebrowser. In the browser mode, for the respective database and the table, under the tab -database structure,following the option Modify Table, respective column could be removed. |