*/\*\* Deborah Barndt  
 \* 4-27-17  
 \* SqlHelper.java  
 \* Lab 8  
 \* This program will extend the functionality of SQLiteOpenHelper, which will manage a  
 \* database creation and version management.  
 \* Written by Deborah Barndt.  
 \*/***package** com.example.u2.bookreviews;  
  
**import** android.content.ContentValues;  
**import** android.content.Context;  
**import** android.database.Cursor;  
**import** android.database.sqlite.SQLiteDatabase;  
**import** android.database.sqlite.SQLiteOpenHelper;  
**import** android.util.Log;  
  
**import** java.util.LinkedList;  
**import** java.util.List;  
  
*/\*\*  
 \* Created by U2 on 3/30/2017.  
 \*/***public class** SqlHelper **extends** SQLiteOpenHelper  
{  
 *// The database version.  
 //private static final int DATABASE\_VERSION = 1;  
 //private static final int DATABASE\_VERSION = 2;  
 //private static final int DATABASE\_VERSION = 3;* **private static final int *DATABASE\_VERSION*** = 5;  
  
 *// The database name.* **private static final** String ***DATABASE\_NAME*** = **"BookDB"**;  
  
 *// The Books table name.* **private static final** String ***TABLE\_BOOKS*** = **"books"**;  
  
 *// The Books Table column names.* **private static final** String ***KEY\_ID*** = **"id"**;  
 **private static final** String ***KEY\_TITLE*** = **"title"**;  
 **private static final** String ***KEY\_AUTHOR*** = **"author"**;  
  
 **public** SqlHelper(Context context)  
 {  
 **super**(context, ***DATABASE\_NAME***, **null**, ***DATABASE\_VERSION***);  
 }  
  
 @Override  
 **public void** onCreate(SQLiteDatabase db)  
 {  
 *// The SQL statement to create the book table.* String CREATE\_BOOK\_TABLE = **"CREATE TABLE books ( "** +  
 **"id INTEGER PRIMARY KEY AUTOINCREMENT, "** +  
 **"title TEXT, "** +  
 **"author TEXT )"**;  
  
 *// Create the books table.* db.execSQL(CREATE\_BOOK\_TABLE);  
 }  
  
 @Override  
 **public void** onUpgrade(SQLiteDatabase db, **int** oldVersion, **int** newVersion)  
 {  
 *// Drop the older books table if it already exists.* db.execSQL(**"DROP TABLE IF EXISTS books"**);  
  
 *// Create a new books table after the older version is dropped.* **this**.onCreate(db);  
  
 String upgradeQuery = **"ALTER TABLE books ADD COLUMN rating TEXT"**;  
 **if** (oldVersion == 4 && newVersion == 5)  
 {  
 db.execSQL(upgradeQuery);  
 }  
 }  
  
 *// Crud operations: create "add", read "get", update, and delete* **public void** addBook(Book book)  
 {  
 Log.*d*(**"addBook"**, book.toString());  
  
 *// First get the reference to the writable database.* SQLiteDatabase db = **this**.getWritableDatabase();  
  
 *// Second create the ContentValues to add key "column"/value.* ContentValues values = **new** ContentValues();  
  
 *// Get the title.* values.put(***KEY\_TITLE***, book.getTitle());  
  
 *// Get the author.* values.put(***KEY\_AUTHOR***, book.getAuthor());  
  
 *// Third insert values into the table.* db.insert(***TABLE\_BOOKS***, *// the table* **null**, *//null ColumnHack* values); *// key/value -> keys = column names/values  
  
 // Fourth close the database.* db.close();  
 }  
  
 *// Get all the books.* **public** List<Book> getAllBooks()  
 {  
 List<Book> books = **new** LinkedList<Book>();  
  
 *// First build the query.* String query = **"SELECT \* FROM "** + ***TABLE\_BOOKS***;  
  
 *// Second get the reference to the writable database.* SQLiteDatabase db = **this**.getWritableDatabase();  
 Cursor cursor = db.rawQuery(query, **null**);  
  
 *// Third go over each row, build book, and add it to list.* Book book = **null**;  
  
 **if**(cursor.moveToFirst())  
 {  
 **do** {  
 book = **new** Book();  
 book.setId(Integer.*parseInt*(cursor.getString(0)));  
 book.setTitle(cursor.getString(1));  
 book.setAuthor(cursor.getString(2));  
 book.setRating(cursor.getString(3));  
  
 *// Add a book to books.* books.add(book);  
 }  
  
 **while**(cursor.moveToNext());  
 }  
  
 Log.*d*(**"getAllBooks()"**, books.toString());  
  
 *// Display the books.* **return** books;  
 }  
  
 *// Updating a single book.* **public int** updateBook(Book book, String newTitle, String newAuthor)  
 {  
 *// First get the reference to the writable database.* SQLiteDatabase db = **this**.getWritableDatabase();  
  
 *// Second create the ContentValues to add the key "column"/value.* ContentValues values = **new** ContentValues();  
  
 *// Get the title  
 //values.put("title", book.getTitle());* values.put(**"title"**, newTitle);  
  
 *// Get the author.  
 //values.put("author", book.getAuthor());* values.put(**"author"**, newAuthor);  
  
 *// Third update the table row.* **int** i = db.update(***TABLE\_BOOKS***, *// the table* values, *// column/value* ***KEY\_ID***+**" = ?"**, *// selections* **new** String[] { String.*valueOf*(book.getId()) }); *// selection args  
  
 // Fourth close the database.* db.close();  
 Log.*d*(**"UpdateBook"**, book.toString());  
 **return** i;  
 }  
  
 *// Deleting a single book.* **public void** deleteBook(Book book)  
 {  
 *// First get the reference to the writable database.* SQLiteDatabase db = **this**.getWritableDatabase();  
  
 *// Second delete the book.* db.delete(***TABLE\_BOOKS***,  
 ***KEY\_ID***+**" = ?"**,  
 **new** String[] { String.*valueOf*(book.getId()) });  
  
 *// Third close the database.* db.close();  
  
 Log.*d*(**"deleteBook"**, book.toString());  
 }  
  
 **public int** getIds(**int** book)  
 {  
 String selectQuery = **"SELECT id FROM books"**;  
 SQLiteDatabase database = **this**.getReadableDatabase();  
 Cursor c = database.rawQuery(selectQuery, **null**);  
 c.moveToFirst();  
 **int** total = c.getCount();  
  
 *// Print total number of books in the Logcat file.* Log.*d*(**"Books"**, **"Total Books: "** + total);  
  
 **return** total;  
 }  
  
 **public** String getRatingMax()  
 {  
 StringBuilder s = **new** StringBuilder();  
 String selectQuery = **"SELECT MAX(title) FROM books"**;  
 SQLiteDatabase db = **this**.getReadableDatabase();  
 Cursor cursor = db.rawQuery(selectQuery,**null**);  
  
 **if**(cursor.moveToFirst())  
 {  
 **do** {  
 *// Get the author's value.* s.append(cursor.getString(0));  
 }  
  
 **while**(cursor.moveToNext());  
 }  
  
 cursor.close();  
 db.close();  
 **return** s.toString();  
 }  
  
 **public** String getRatingMin()  
 {  
 StringBuilder s = **new** StringBuilder();  
 String selectQuery = **"SELECT MIN(title) FROM books"**;  
 SQLiteDatabase db = **this**.getReadableDatabase();  
 Cursor cursor = db.rawQuery(selectQuery,**null**);  
  
 **if**(cursor.moveToFirst())  
 {  
 **do** {  
 *// Get the author's value.* s.append(cursor.getString(0));  
 }  
  
 **while**(cursor.moveToNext());  
 }  
  
 cursor.close();  
 db.close();  
 **return** s.toString();  
 }  
  
  
 **public** String getTotal()  
 {  
 StringBuilder s = **new** StringBuilder();  
 String selectQuery = **"SELECT COUNT(\*) FROM books"**;  
 SQLiteDatabase db = **this**.getReadableDatabase();  
 Cursor cursor = db.rawQuery(selectQuery,**null**);  
 cursor.moveToFirst();  
 **int** total = cursor.getCount();  
  
 **if**(cursor.moveToFirst())  
 {  
 **do** {  
 *// Get the author's value.* s.append(cursor.getString(0));  
 }  
  
 **while**(cursor.moveToNext());  
 }  
  
 cursor.close();  
 db.close();  
 **return** s.toString();  
 }  
  
  
 **public** String getBooks()  
 {  
 StringBuilder s = **new** StringBuilder();  
 String selectQuery = **"SELECT title FROM books"**;  
 SQLiteDatabase db = **this**.getReadableDatabase();  
 Cursor cursor = db.rawQuery(selectQuery,**null**);  
  
 **if**(cursor.moveToFirst())  
 {  
 **do** {  
 *// Get the author's value.* s.append(cursor.getString(0));  
 }  
  
 **while**(cursor.moveToNext());  
 }  
 cursor.close();  
 db.close();  
 **return** s.toString();  
 }  
  
}