Appendix H: Writing to the Database

```
public void writetoDB(String pathtofiles, String tableName) {

	// Declare the JDBC objects.
	con = null;
	stmt = null;
	rs = null;

	try {
		// Establish the connection - sqllite server:
		Class.forName("org.sqlite.JDBC");
		con = DriverManager.getConnection("jdbc:sqlite:" + connectionUrl);

		File child = new File(pathtofiles);

		FileReader reader = new FileReader(child);
		Scanner in = new Scanner(reader);

		while (in.hasNextLine()) {
			String Num = in.nextLine().replace("<num>", "");
			Num = Num.replace("</num>", "");

			String Query = in.nextLine().replace("<query>", "");
			Query = Query.replace("</query>", "");

			// write details to DB:
			String SQL1 = "INSERT INTO " + tableName + " (queryNum,
queryTerms) VALUES(?, ?);";

			PreparedStatement pstmt1 = con.prepareStatement(SQL1);
			// create statement

			// a statement
			pstmt1.setString(1, Num);
			pstmt1.setString(2, Query);
			pstmt1.executeUpdate();
			// execute insert statement
			pstmt1.close();
			}
```

Synopsis of what happens:

- Read in the path to the Query files, specified else ware.
- Use replace functions to remove the XML content, <Query> and <Num> tags.
- Create a string for SQL statement, with '?' meaning wildcard, so can expect anything.
- Create a statement for this query string, using PreparedStatement.
- Set where things are to be aligned, eg. Num in first column.
- Execute the statement and close the statement to remove leaked resources.