Сидоренко В.А., старший преподаватель
ФГБОУ ВО «КалмГУ»,
г. Элиста, Российская Федерация

Проблемы несогласованности типа DATE в связке СУБД SQLite и среды разработки Lazarus

Аннотация

В статье показана проблема несогласованности связки СУБД SQLite и интегрированной среды разработки Lazarus в части использования типа Date, применяемого в базах данных. Рассмотрен практический пример с пояснениями причин такой проблемы, а также даны рекомендации по работе с указанным типом в SQLite. Данная статья будет полезна для разработчиков баз данных, использующих среду программирования Lazarus.

Ключевые слова: Lazarus и SQLite, тип Date в SQLite, проблемы несогласованности SQLite и Lazarus.

Lazarus – это открытая среда программной разработки на языке Object Pascal [2]. Основная её цель – предоставление кроссплатформенных и свободных средств разработки в Delphi-подобном окружении. То есть, считается, что это бесплатная замена инструменту Delphi. Мы используем Lazarus в учебном процессе, как инструмент для разработки компьютерных приложений с удобным графическим пользовательским интерфейсом. Так же как и Delphi, Lazarus позволяет работать с различными типами баз данных. Для этого имеются соответствующие библиотеки с объектами-компонентами. Многие компоненты, особенно те, что чаще всего используются в создании интерфейса (меню, кнопки, окна ввода, списки и т.д.) по своим характеристикам практически совпадают – это стандартные компоненты. Но, что касается баз данных, то здесь имеются некоторые отличия. В основном отмечается сокращение технических возможностей в работе с базами данных. Набор библиотек для баз данных тоже отличается – то есть, здесь используются другие типы баз данных. Информации по данной теме в Интернете крайне мало. Поэтому приходится много экспериментировать в этом направлении и соответственно сталкиваться с некоторыми проблемами. Одна из них и будет описана в этой статье.

В Lazarus есть возможность работать с SQLite [1]. Это компактная встраиваемая СУБД, так же свободно распространяемая, как и Lazarus. SQLite хранит всю базу данных (включая определения, таблицы, индексы и данные) в единственном стандартном файле на том компьютере, на котором исполняется программа. То есть, по сути она похожа на обычные промышленные СУБД – такие, как MS SQL Server, Oracle, InterBase и т.д. Поэтому есть смысл учиться работать с базами данных на её примере. Сама приставка в названии Lite

намекает на то, что она облегчённая. Так и есть, некоторые характеристики, свойства, возможности упрощены или сокращены. Но для начального знакомства со всем этим серьёзным семейством клиент-серверных СУБД, управляемых с помощью языка SQL, она подходит, и мы её включили в программу. К тому же, как оказалось, у неё имеются некоторые непредсказуемые проблемы, для обхода которых приходится изобретать дополнительные механизмы, что для приобретения опыта тоже полезно.

Итак, начнём с типа Date. Строго говоря, предъявлять претензии к странному поведению этого типа нельзя – по причине того, что никто и не обещал, что в таблицах SQLite можно задавать этот тип. В стандарте SQL-92 его просто нет, а подобные СУБД лишь гарантируют, можно так сказать, выполнение этого стандарта. А далее они могут добавлять что-то своё. Но промышленные СУБД всё-таки гарантируют надёжную работу своих дополнений.

Известно, что официально в SQLite используются 4 основных типа данных, это: INTEGER, TEXT, BLOB и REAL. Обычно предлагается для даты использовать тип TEXT. В крайнем случае, можно и так. Но, во-первых, ввод даты в таком формате слабо контролируется. Во-вторых, плохо обрабатывается, если, например, потребуется сортировка или выборка по диапазону. Конечно, для таких целей можно установить, например, формат YYYY.MM.DD. Но такой непривычный формат плохо воспринимается пользователем. Поэтому желательно всё же применять наш обычный формат DD.MM.YYYY и обойтись без типа TEXT. Кстати, в системе Linux (иногда и в Windows) формат даты может отличаться от привычного, поэтому желательно в начале работы программы задать настройку даты:

```
DefaultFormatSettings.ShortDateFormat := 'dd.mm.yyyy';
```

Оказывается, что всё же SQLite принимает некоторые дополнительные часто используемые типы: DATE, CHAR, VARCHAR. Их можно задавать при создании таблиц, и они работают. Но посмотрим, как работает тип DATE.

Для работы с базой данной удобно использовать менеджер БД, с помощью которого можно управлять базой в визуальном режиме. Для СУБД SQLite самый удобный менеджер – это **SQLiteStudio.** Допустим, мы создали в SQLiteStudio тестовую базу данных, а в ней таблицу, в которой один из столбцов определили как типа DATE. Там же в SQLiteStudio ввели несколько записей.

	ID	Name	DR
1	1	Васильев А.А.	31.10.2000
2	2	Кузьмин В.У.	23.12.1910
3	3	Брежнев Л.И.	19.12.1906
4	4	Ленин В.И.	10.04.1970
5	5	Сталин И.В.	21.12.1879

Рис.1. Таблица Proba1 в SQLiteStudio

Проверим реакцию на запрос с условием:

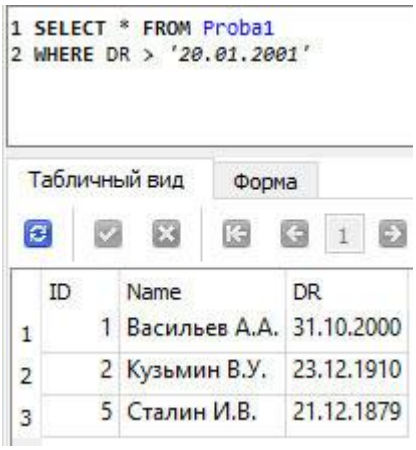

Рис.2. Запрос к таблице Proba1

Возникает подозрение, что тип DATE в SQLite трактуется как обычный текст. Убедимся в этом, вводя текст в это поле. При этом никаких замечаний по поводу формата даты от SQLite не поступает:

	ID	Name	DR
1	1	Васильев А.А.	31.10.2000
2	2	Кузьмин В.У.	23.12.1910
3	3	Брежнев Л.И.	не знаю
4	4	Ленин В.И.	1870 10 апреля
5	5	Сталин И.В.	

Рис.3. Тип DATE – это просто текст

А в итоге вот что мы видим в Lazarus:

ID	Name	DR
1	Васильев А.А.	30.12.1899
2	Кузьмин В.У.	30.12.1899
3	Брежнев Л.И.	30.12.1899
4	Ленин В.И.	30.12.1899
5	Сталин И.В.	30.12.1899

Рис.4. Таблица Proba1 в Lazarus

Следовательно, поле DR здесь воспринимается как пустое. Тип DATE в языке Pascal трактуется как число с началом отсчёта от даты 30.12.1899. Поэтому здесь это «пустое» значение принято за 0, откуда и появилась эта странная дата. И то, что Lazarus это поле считает датой, доказывается именно тем, что он переводит всё же значение в дату.

Обычно с базой данных работают не только в менеджере БД (в данном случае SQLiteStudio), всё-таки приложение создаётся в среде разработки – в данном случае Lazarus на языке Object Pascal. Поэтому есть интерес научиться в этой среде работать с датой в SQLite. Тем более, что Lazarus не отказывается от типа DATE и даже по-своему пытается нам дату показать.

Попробуем здесь же в DBGrid исправить записи таблицы:

	ID	Name	DR
	1	Васильев А.А.	31.10.2000
	2	Кузьмин В.У.	23.12.1910
	3	Брежнев Л.И.	19.12.1906
	4	Ленин В.И.	22.04.1870
	5	Сталин И.В.	21.12.1879
▶	6	Пушкин А.С.	06.06.1799

Рис.5.Попытка исправления записей в Lazarus

Попутно выяснилось, что корректировать таблицу через DBGrid, представленную компонентом SQLQuery только тогда можно, если в таблице задан главный ключ – без него изменения в таблице не сохраняются. И естественно, чтобы набор данных мог в принципе корректироваться, следует соблюсти условия для SELECT-запроса (таблица одна, нет ORDER и GROUP и т.д.), а также обязательно при этом для SQLQuery задать свойство Options = [sqoKeepOpenOnCommit, sqoAutoApplyUpdates, sqoAutoCommit]. Если главный ключ не задан, то изменять таблицу можно только через SQL-запросы (UPDATE, INSERT), без компонентов типа DBGrid или DBEdit.

Теперь посмотрим эти записи снова в SQLiteStudio.

	ID	Name	DR
1	1	Васильев А.А.	2451848.5
2	2	Кузьмин В.У.	2419028.5
3	3	Брежнев Л.И.	2417563.5
4	4	Ленин В.И.	2404174.5
5	5	Сталин И.В.	2407704.5
6	6	Пушкин А.С.	2378287.5

Рис.6. SQLiteSudio: результат после исправления записей в Lazarus

Понятно, что SQLite уже не может сопоставить какой-то текст заданным датам. Получились какие-то странные числа.

Вывод следующий. Задавать тип DATE можно, но при этом следует понимать, что Lazarus с ним будет работать так, как принято в Pascal – через число дней от даты 30.12.1899, а сам SQLite – как с текстом. И эти действия несовместимы.

Список использованной литературы

1. Википедия: SQLite https://ru.wikipedia.org/wiki/SQLite
2. Википедия: Lazarus https://ru.wikipedia.org/wiki/Lazarus
3. Алексеев Е. Р., Чеснокова О. В., Кучер Т. В. Free Pascal и Lazarus: Учебник по программированию. – М.: Альт Линукс, ДМК Пресс, 2010. 440 с.
4. M. van Canneyt, M. Gartner, S.Heinig, F.Monteiro de Cavalho, I.Ouedraogo. Lazarus, the Complete Guide. Blaise Pascal Magazine, 2011. 735 с.

Сидоренко В.А., 2023