Subject: BOOL in SQLite
Posted by borbek on Wed, 17 Jul 2019 12:00:53 GMT
View Forum Message <> Reply to Message

Hi!

In Sqlite3Schema.h BOOL replaces by TEXT:

```
#define BOOL(x)                 COLUMN("text", bool, x, 0, 0)
#define BOOL_ARRAY(x, items)       COLUMN_ARRAY("text", bool, x, 0, 0, items)
#define BOOL_(x)                 COLUMN_("text", bool, x, 0, 0)
#define BOOL_ARRAY_(x, items)      COLUMN_ARRAY_("text", bool, x, 0, 0, items)
```

but in sqlite, the boolean is an integer 0 or 1, so, IMHO, it is right:

```
#define BOOL(x)                 COLUMN("integer", bool, x, 0, 0)
#define BOOL_ARRAY(x, items)       COLUMN_ARRAY("integer", bool, x, 0, 0, items)
#define BOOL_(x)                 COLUMN_("integer", bool, x, 0, 0)
#define BOOL_ARRAY_(x, items)      COLUMN_ARRAY_("integer", bool, x, 0, 0, items)
```

"
3.1. No Separate BOOLEAN Datatype
Unlike most other SQL implementations, SQLite does not have a separate BOOLEAN data type. Instead, TRUE and FALSE are (normally) represented as integers 1 and 0, respectively.
"

Subject: Re: BOOL in SQLite
Posted by mirek on Fri, 09 Aug 2019 07:32:56 GMT
View Forum Message <> Reply to Message

borbek wrote on Wed, 17 July 2019 14:00Hi!

In Sqlite3Schema.h BOOL replaces by TEXT:

```
#define BOOL(x)                 COLUMN("text", bool, x, 0, 0)
#define BOOL_ARRAY(x, items)       COLUMN_ARRAY("text", bool, x, 0, 0, items)
#define BOOL_(x)                 COLUMN_("text", bool, x, 0, 0)
#define BOOL_ARRAY_(x, items)      COLUMN_ARRAY_("text", bool, x, 0, 0, items)
```

but in sqlite, the boolean is an integer 0 or 1, so, IMHO, it is right:

```
#define BOOL(x)                 COLUMN("integer", bool, x, 0, 0)
#define BOOL_ARRAY(x, items)       COLUMN_ARRAY("integer", bool, x, 0, 0, items)
#define BOOL_(x)                 COLUMN_("integer", bool, x, 0, 0)
#define BOOL_ARRAY_(x, items)      COLUMN_ARRAY_("integer", bool, x, 0, 0, items)
```

"

3.1. No Separate BOOLEAN Datatype
Unlike most other SQL implementations, SQLite does not have a separate BOOLEAN data type. Instead, TRUE and FALSE are (normally) represented as integers 1 and 0, respectively.
"

Well, this is more about cross-db compatibility. SQLs often lack(ed) BOOL type, but often it can be represented by single character, which can often be stored as single byte.

As it is better to have single model for all databases, we had to choose between numerical and character. Choosed character.

Note: The reason why it is better to have single model is that at some point, you want to have uniform way how to deal with it in widgets.
