Subject: sqlite and dropdate / editdate etc
Posted by jimlef on Fri, 25 Sep 2020 05:19:53 GMT
View Forum Message <> Reply to Message

Is there a way to have an sqlctrl dropdate for example that retrieves a field from your db as int and translates that to date? As in
17093 = 01/07/2019

Currently, it only seems to show the integer value...

Thanks!

Subject: Re: sqlite and dropdate / editdate etc
Posted by mirek on Fri, 25 Sep 2020 06:18:53 GMT
View Forum Message <> Reply to Message

jimlef wrote on Fri, 25 September 2020 07:19Is there a way to have an sqlctrl dropdate for example that retrieves a field from your db as int and translates that to date? As in
17093 = 01/07/2019

Currently, it only seems to show the integer value...

Thanks!

Nothing ready made, but easy to implement.

For EditField, you should assign custom made Convert (that is basically a class that converts Value to String and back).

DropDate might be a bit more complicated. I would probably made it a child of some helper Ctrl and would do the conversion in its parent SetData/GetData. Something like

```
struct DropDateInt {
	DropDate dd;
	const Date zero_date = Date(1970, 1, 1);
 virtual void   SetData(const Value& data) { dd <<= zero_date + (int)data; }
 virtual Value  GetData() const            { return (Date)~dd - zero_date; }

	DropDateInt() { Add(dd.SizePos()); }
}
```

(not tested, there might be some caveat I do not see now...)

Why do you need this? Sqlite3 seems to have date type AFAIK...

Mirek

Subject: Re: sqlite and dropdate / editdate etc
Posted by jimlef on Fri, 25 Sep 2020 06:30:49 GMT
View Forum Message <> Reply to Message

Thanks for that Mirek, I was just looking into the option because my original code used datetime.ticks values. I have converted those huge numbers to epoch based ints, but I suppose it would be just as easy for me to convert that to actual sqlite dates... In fact I think I'm going to go that route :)

Jim
Edit: Looking more closely, it seems that sqlite doesn't have an official date type, but can store dates as strings or numbers ... I can convert them to strings however, and formatting will not be difficult.

Subject: Re: sqlite and dropdate / editdate etc
Posted by jimlef on Fri, 25 Sep 2020 21:37:59 GMT
View Forum Message <> Reply to Message

I have converted the int values to string dates, and the controls (editdate, dropdate etc) now accept the field as a date. However, when editing I find that the date format is changed when written back. My files have m/d/yyyy, but when say editdate writes it back out it goes yyyy-mm-dd?

My locale is set systemwide to en_US.UTF-8, which should do month/day/year, or so I thought? Also, localtime -k LC_TIME shows d_fmt="%m/%d/%Y", as expected...

Any ideas?

Subject: Re: sqlite and dropdate / editdate etc
Posted by mr_ped on Sat, 26 Sep 2020 02:28:14 GMT
View Forum Message <> Reply to Message

just a general note: I would rather store epoch seconds or some UTC timestamp than formatted date.
The particular formatting/timezone should be done when viewing the data and adjusted to current user locale/preference/options, but database should have the data itself. (that way if multiple users from different timezones are viewing the same datetime, each user sees their own familiar form of it)

Also with storing the formatted date (in general sense, if you store any kind of date) you may easily run into some peculiarities of certain dates and times (for example there's every now and then a day with 61 seconds in last minute, i.e. 23:59:60, etc), so having it in form of some timestamp supported by some common datetime library (or C++ native support, I think there's some date/time stuff added recently in C++17 or C++20? or was it already C++14?) is usually the safest way, how to avoid all of these complex things around time, and just use the library functions.

Subject: Re: sqlite and dropdate / editdate etc
Posted by jimlef on Sat, 26 Sep 2020 05:43:56 GMT
View Forum Message <> Reply to Message

I personally prefer epoch dates, but with that I was getting invalid info when assigning (automatically) to sqlctrls... Instead of editdate or dropdate showing a date, they'd show the integer value that was stored.

Perhaps that is a weakness of sqlite, but I'm just working things out as I go :) I know I still have a lot to learn as well...

Can always change back to ints (wrote (hacked together) a little c# util that understands epoch and the original datetime contstructs for the conversions).

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SQLite;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace convertdates
{
   public partial class Form1 : Form
   {
      DataTable dt = new DataTable();
      TransactionsTable transactionsTable = new TransactionsTable();
      ProductsTable productsTable = new ProductsTable();
      CustomersTable customersTable = new CustomersTable();

      string myConnection = "Data Source=sample.db;Version=3;";
```

```
        public Form1()
        {
            InitializeComponent();
            DateTime dt = new DateTime(1970, 1, 1);
            long sticks = dt.Ticks;
            label1.Text = sticks.ToString();
        }

        private string correctdate(long baddate)
        {
            DateTime dt = new DateTime(1970, 1, 1);
            long junk = baddate * 86400 * 10000000;
            long sticks = dt.Ticks;
            junk += sticks;
            DateTime retvalue = new DateTime(junk);
            // long datenum = dtp1.Value.Date.Ticks;
            // long dayssince1970 = ((baddate * 86400 * 10000000) - sticks) / 86400 / 10000000;
            return retvalue.ToShortDateString();
        }

        private void btnConvert_Click(object sender, EventArgs e)
        {
            dt = DisplayAllTransactions();
            foreach (DataRow row in dt.Rows)
            {
                transactionsTable.INVOICES_ID = int.Parse(row[0].ToString());
                transactionsTable.TRANSACTIONDATE = transactionsTable.DATEPAID =
correctdate(long.Parse(row[1].ToString()));
                UpdateTransactions(transactionsTable);
            }
            dt = SelectProducts();
            foreach (DataRow row in dt.Rows)
            {
                productsTable.PROD_ID = int.Parse(row[0].ToString());
                productsTable.DATEPURCHASED = correctdate(long.Parse(row[1].ToString()));
                UpdateProducts(productsTable);
            }

        }
        public DataTable DisplayAllTransactions()
        {
            //SQLiteConnection First
            SQLiteConnection conn = new SQLiteConnection(myConnection);

            //Create a DAta Table to hold the datafrom database temporarily
            DataTable dt = new DataTable();

            try
```

```
        {
            //Write the SQL Query to Display all Transactions
            string sql = "SELECT INVOICES_ID, TRANSACTIONDATE, DATEPAID FROM
INVOICES";

            //SQLiteCommand to Execute Query
            SQLiteCommand cmd = new SQLiteCommand(sql, conn);

            //SQLiteDataAdapter to Hold the data from database
            SQLiteDataAdapter adapter = new SQLiteDataAdapter(cmd);

            //Open DAtabase Connection
            conn.Open();

            adapter.Fill(dt);
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
        finally
        {
            if (ConnectionState.Open == conn.State) conn.Close();
        }

        return dt;
    }
    #region Method to Update transaction in Database
    bool UpdateTransactions(TransactionsTable p)
    {
        //create a boolean variable and set its initial value to false
        bool isSuccess = false;

        //Create SQL Connection for DAtabase
        SQLiteConnection conn = new SQLiteConnection(myConnection);

        try
        {
            //SQL Query to Update Data in dAtabase
            String sql = "UPDATE INVOICES SET TRANSACTIONDATE=@transactionDate,
DATEPAID=@datePaid WHERE INVOICES_ID=@id";

            //Create SQL Cmmand to pass the value to query
            SQLiteCommand cmd = new SQLiteCommand(sql, conn);
            //Passing the values using parameters and cmd
            cmd.Parameters.AddWithValue("@transactionDate", p.TRANSACTIONDATE);
            cmd.Parameters.AddWithValue("@datePaid", p.DATEPAID);
            cmd.Parameters.AddWithValue("@id", p.INVOICES_ID);
```

```
            //Open the Database connection
            conn.Open();

            //Create Int Variable to check if the query is executed successfully or not
            int rows = cmd.ExecuteNonQuery();

            //if the query is executed successfully then the value of rows will be greater than 0 else
it will be less than zero
            if (rows > 0)
            {
                //Query ExecutedSuccessfully
                isSuccess = true;
            }
            else
            {
                //Failed to Execute Query
                isSuccess = false;
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
        finally
        {
            conn.Close();
        }

        return isSuccess;
    }
    #endregion
    #region Select method for Product Module
    public DataTable SelectProducts()
    {
        //Create Sql Connection to connect Databaes
        SQLiteConnection conn = new SQLiteConnection(myConnection);

        //DAtaTable to hold the data from database
        DataTable dt = new DataTable();

        try
        {
            //Writing the Query to Select all the products from database
            String sql = "SELECT PROD_ID, DATEPURCHASED FROM PRODUCTS";

            //Creating SQL Command to Execute Query
            SQLiteCommand cmd = new SQLiteCommand(sql, conn);
```

```
            //SQL Data Adapter to hold the value from database temporarily
            SQLiteDataAdapter adapter = new SQLiteDataAdapter(cmd);

            //Open DAtabase Connection
            conn.Open();

            adapter.Fill(dt);
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
        finally
        {
            conn.Close();
        }

        return dt;
    }
    #endregion
    #region Method to Update Product in Database
    bool UpdateProducts(ProductsTable p)
    {
        //create a boolean variable and set its initial value to false
        bool isSuccess = false;

        //Create SQL Connection for DAtabase
        SQLiteConnection conn = new SQLiteConnection(myConnection);

        try
        {
            //SQL Query to Update Data in dAtabase
            String sql = "UPDATE PRODUCTS SET DATEPURCHASED=@datepurchased
WHERE PROD_ID=@id";

            //Create SQL Cmmand to pass the value to query
            SQLiteCommand cmd = new SQLiteCommand(sql, conn);
            //Passing the values using parameters and cmd
            cmd.Parameters.AddWithValue("@datepurchased", p.DATEPURCHASED);
            cmd.Parameters.AddWithValue("@id", p.PROD_ID);

            //Open the Database connection
            conn.Open();

            //Create Int Variable to check if the query is executed successfully or not
            int rows = cmd.ExecuteNonQuery();
```

```
            //if the query is executed successfully then the value of rows will be greater than 0 else
it will be less than zero
            if (rows > 0)
            {
                //Query ExecutedSuccessfully
                isSuccess = true;
            }
            else
            {
                //Failed to Execute Query
                isSuccess = false;
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
        finally
        {
            conn.Close();
        }

        return isSuccess;
    }
    #endregion

    private void btnSelectDB_Click(object sender, EventArgs e)
    {
        string ProgramDataDir =
Path.Combine(Environment.ExpandEnvironmentVariables("%userprofile%"), "Documents") + "\\";
        Directory.CreateDirectory(ProgramDataDir);
        ofDialog.DefaultExt = "jts|.db";
        ofDialog.CheckFileExists = true;
        ofDialog.InitialDirectory = ProgramDataDir;
        if (ofDialog.ShowDialog() == DialogResult.OK)
        {
            myConnection = "Data Source=" + ofDialog.FileName + ";Version=3;";
        }
    }
    public DataTable SelectCustomers()
    {
        //SQL Connection for Database Connection
        SQLiteConnection conn = new SQLiteConnection(myConnection);

        //DataTble to hold the value from database and return it
        DataTable dt = new DataTable();

        try
```

```
        {
            //Write SQL Query t Select all the DAta from dAtabase
            string sql = "SELECT * FROM CUSTOMERS";

            //Creating SQL Command to execute Query
            SQLiteCommand cmd = new SQLiteCommand(sql, conn);

            //Creting SQL Data Adapter to Store Data From Database Temporarily
            SQLiteDataAdapter adapter = new SQLiteDataAdapter(cmd);

            //Open Database Connection
            conn.Open();
            //Passign the value from SQL Data Adapter to DAta table
            adapter.Fill(dt);
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
        finally
        {
            conn.Close();
        }

        return dt;
    }
    bool UpdateCustomers(CustomersTable dc)
    {
        //SQL Connection for Database Connection
        SQLiteConnection conn = new SQLiteConnection(myConnection);
        //Create Boolean variable and set its default value to false
        bool isSuccess = false;

        try
        {
            //SQL Query to update data in database
            string sql = "UPDATE CUSTOMERS SET CUSTNAME=@name, EMAIL=@email,
CONTACT=@contact, ADDRESS=@address WHERE CUST_ID=@id";
            //Create SQL Command to pass the value in sql
            SQLiteCommand cmd = new SQLiteCommand(sql, conn);

            //Passing the values through parameters
            cmd.Parameters.AddWithValue("@name", dc.CUSTNAME);
            cmd.Parameters.AddWithValue("@email", dc.EMAIL);
            cmd.Parameters.AddWithValue("@contact", dc.CONTACT);
            cmd.Parameters.AddWithValue("@address", dc.ADDRESS);
            cmd.Parameters.AddWithValue("@id", dc.CUST_ID);
```

```
            //open the Database Connection
            conn.Open();

            //Int varialbe to check if the query executed successfully or not
            int rows = cmd.ExecuteNonQuery();
            if (rows > 0)
            {
                //Query Executed Successfully
                isSuccess = true;
            }
            else
            {
                //Failed to Execute Query
                isSuccess = false;
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
        finally
        {
            conn.Close();
        }

        return isSuccess;
    }
    public string GenerateNumber()
    {
        Random random = new Random();
        string r = "";
        int i;
        for (i = 1; i < 11; i++)
        {
            r += random.Next(0, 9).ToString();
        }
        return r;
    }
    private void btnGeneralize_Click(object sender, EventArgs e)
    {
        var rand = new Random();
        string[] firstNames = new string[] { "Jim", "Mary", "Catherine", "John", "Mark", "Luke",
"Lucy", "Abi", "Roger", "Ruth" };
        string[] lastNames = new string[] { "Jones", "Smith", "Johnson", "Yancovich",
"Steemburgin", "Carter", "Reagan", "Trump", "Jefferson", "Franklin" };
        dt = SelectCustomers();
        foreach (DataRow row in dt.Rows)
        {
```

```
            string CustFirstName = firstNames[rand.Next(10)];
            string CustLastName = lastNames[rand.Next(10)];
            string Email = CustFirstName + "." + CustLastName + "@gmail.com";
            string Contact = GenerateNumber();
            customersTable.CUST_ID = int.Parse(row[0].ToString());
            customersTable.CUSTNAME = CustFirstName + " " + CustLastName;
            customersTable.CONTACT = Contact;
            customersTable.EMAIL = Email;
            customersTable.ADDRESS = "0 Main Way";
            UpdateCustomers(customersTable);
        }

    }

    private void dtp1_ValueChanged(object sender, EventArgs e)
    {
        DateTime dt = new DateTime(1970, 1, 1);
        long sticks = dt.Ticks;
        long datenum = dtp1.Value.Date.Ticks;
        long dayssince1970 = (datenum - sticks) / 86400 / 10000000;
        label1.Text = dayssince1970.ToString();
        Clipboard.SetText(label1.Text);
    }

}
class ProductsTable
    {
        //Getters and Setters for Product Module
        public int PROD_ID { get; set; }
        public string PRODNAME { get; set; }
        public string PRODDESCRIPTION { get; set; }
        public string DATEPURCHASED { get; set; }
        public decimal COST { get; set; }
        public int INVOICEID { get; set; }

    }
public enum status
{
    notpaid = 0,
    voided,
    partialpayment,
    paidinfull
};
class TransactionsTable
{
    public int INVOICES_ID { get; set; }
    public int INVOICENUMBER { get; set; }
    public int CUSTOMERID { get; set; }
```

```
        public string TRANSACTIONDATE { get; set; }
        // public long due_date { get; set; }
        public string TERMS { get; set; }
        public decimal NONTAXABLESUB { get; set; }
        public decimal TAXABLESUB { get; set; }
        public decimal TAX { get; set; }
        public decimal GRANDTOTAL { get; set; }
        public decimal AMTPAID { get; set; }
        public string DATEPAID { get; set; }
        public int STATUS { get; set; }

    }
    class CustomersTable
    {
        public int CUST_ID { get; set; }
        public string CUSTNAME { get; set; }
        public string EMAIL { get; set; }
        public string CONTACT { get; set; }
        public string ADDRESS { get; set; }
        public string CITY { get; set; }
        public string STATE { get; set; }
        public string ZIP { get; set; }
        public int TAXABLE { get; set; }

    }
}
```

Subject: Re: sqlite and dropdate / editdate etc
Posted by mirek on Sat, 26 Sep 2020 13:10:31 GMT
View Forum Message <> Reply to Message

jimlef wrote on Fri, 25 September 2020 08:30Thanks for that Mirek, I was just looking into the option because my original code used datetime.ticks values. I have converted those huge numbers to epoch based ints, but I suppose it would be just as easy for me to convert that to actual sqlite dates... In fact I think I'm going to go that route :)

Jim
Edit: Looking more closely, it seems that sqlite doesn't have an official date type, but can store dates as strings or numbers ... I can convert them to strings however, and formatting will not be difficult.

Sorry, missed the "Edit:" :)

You are overthinking it. Sqlite3 U++ module does this conversion for you...

Mirek

Subject: Re: sqlite and dropdate / editdate etc
Posted by jimlef on Sat, 26 Sep 2020 16:16:29 GMT
View Forum Message <> Reply to Message

I do that a lot Mirek :lol:

I am testing things out, I added a field TESTING to my products table. I added a Testing editdate to the layout for editing prodcuts.
When I save a testing date to my database, it's stored as yyyy-mm-dd, while my region is supposed to put year last. I'd like to use my region to manage the format, as that would make the code more generally usable (for later sharing). When I reload the info into my editdate field, the field is colored light red... doesn't that indicate invalid value?

File Attachments
1) example.png, downloaded 578 times

Subject: Re: sqlite and dropdate / editdate etc
Posted by mirek on Mon, 28 Sep 2020 12:19:13 GMT
View Forum Message <> Reply to Message

jimlef wrote on Sat, 26 September 2020 18:16I do that a lot Mirek :lol:

I am testing things out, I added a field TESTING to my products table. I added a Testing editdate to the layout for editing prodcuts.
When I save a testing date to my database, it's stored as yyyy-mm-dd, while my region is supposed to put year last. I'd like to use my region to manage the format, as that would make the code more generally usable (for later sharing).

OK, that one is unexpected :) This would require changing Sqlite3 code module. I can do that, but I am really not sure this is a good idea; as these text are basically internal format, maybe the most logical internal format is indeed yyyy-mm-dd. Apart from being ISO 8601 conformant, it at least can be sorted correctly... From broader perspective, this is not what user will see, this is basically only issue for admin and I think most admins would be more comfortable with ISO...

Quote:
When I reload the info into my editdate field, the field is colored light red... doesn't that indicate invalid value?

Yes. If you would call Accept for that dialog (that basically happens on normal OK button, more generaly on all "acceptor" buttons), it would display error and return false.

Mirek

Subject: Re: sqlite and dropdate / editdate etc
Posted by jimlef on Mon, 28 Sep 2020 13:59:55 GMT
View Forum Message <> Reply to Message

The internally stored format is irrelevant to me, as whatever format the program expects can be used (and converted to)...

Then only 1 issue remains. I have here two dropdate ctrls, auto loading from sqlctrls functions.

They retrieve the stored iso date and show it as stored ie 2020-03-15. They then mark it as invalid, and pop an
error if you try to store a new date in that format. The same holds true for EditDate, naturally.

Shouldn't they show dates in the current locale format? When I drop the calendar and click on a date,
the new date is shown correctly, at least for my locale. With EditDate, when I type in manually 03-15-2020 or
even using / instead, it is accepted. But why doesn't it show in that mode initially? I think it is failing to
parse the date info, and will try stepping through to verify.

File Attachments

1) example.png, downloaded 523 times

Subject: Re: sqlite and dropdate / editdate etc
Posted by jimlef on Mon, 28 Sep 2020 15:41:08 GMT
View Forum Message <> Reply to Message

Upon further inspection, it seems that the ctrls Convert functions scan the field data, and interpret as string or int, never DATE_V (since sqlite doesn't have an official 'date_v' type). So, date formatting isn't applied. I guess what I need is a concise way to tell the function that this field should be interpreted as qtype date_v? Then make sure that the data in that field is appropriate for that type.

Subject: Re: sqlite and dropdate / editdate etc
Posted by mirek on Mon, 28 Sep 2020 15:49:50 GMT
View Forum Message <> Reply to Message

jimlef wrote on Mon, 28 September 2020 17:41Upon further inspection, it seems that the ctrls Convert functions scan the field data, and interpret as string or int, never DATE_V (since sqlite doesn't have an official 'date_v' type). So, date formatting isn't applied. I guess what I need is a concise way to tell the function that this field should be interpreted as qtype date_v? Then make sure that the data in that field is appropriate for that type.

Uhmm I bet it is something else. Convert for EditDate definitely convertes between text and Date...

Just my wild guess: Is not it possible that your SQL type was originally STRING and you have changed to DATE after schema was already created? If yes, then the problem is that schema update script only adds columns (and most of those adds simply fail because the column is already there, that is by design), it does not change the type of column.... (fix is to remove the column manually or perhaps drop whole table or even database).

Mirek

Subject: Re: sqlite and dropdate / editdate etc
Posted by mirek on Mon, 28 Sep 2020 16:06:38 GMT
View Forum Message <> Reply to Message

Just to be sure, for a quick test I have modified Sqlite3 reference example and everything seems to work as expected. Attached.

File Attachments
1) SQL_Sqlite3.zip, downloaded 180 times

Subject: Re: sqlite and dropdate / editdate etc
Posted by jimlef on Mon, 28 Sep 2020 16:51:45 GMT
View Forum Message <> Reply to Message

My original sqlite database had a long int, as in 636824160000000000 - as I was writing in C# on dotnet, and DateTime.Ticks was a convenient storage unit there. Let me see what happens with that...

Subject: Re: sqlite and dropdate / editdate etc
Posted by jimlef on Mon, 28 Sep 2020 18:33:47 GMT
View Forum Message <> Reply to Message

Ok, ever onward :)

I've created an sql file (attached) to generate a new database file.

Using this new db, and with dates from 1970 forward (epoch dates I believe), 5 digit ints.

Have dateconvert "borrowed" from sqlapp example "borrow.cpp". Dates show up perfectly in both tables and editdate controls.

Problem is, when I go to edit, I get assertion failure :(I delete contents and start with a number and ...

File Attachments

1) example.png, downloaded 477 times
2) sample.sql, downloaded 183 times

Subject: Re: sqlite and dropdate / editdate etc
Posted by mirek on Mon, 28 Sep 2020 20:51:02 GMT
View Forum Message <> Reply to Message

I would look for a bug in your Scan routine...

Subject: Re: sqlite and dropdate / editdate etc
Posted by jimlef on Mon, 28 Sep 2020 22:51:42 GMT
View Forum Message <> Reply to Message

Yes, Mirek, you nailed it, naturally :) And I found a bug in related code elsewhere... Now dates are working as intended, both loading and saving.

Thank you again!
