Creating a submission tracker with OpenOffice.org Base

TRACK YOUR PAPER TRAIL

Writers who submit papers to publications can create a database to track and report the status of their projects. **BY DMITRI POPOV**

For a writing professional, a tool that keeps tabs on submissions is as important as a time-tracking utility. If you write for a living, you might want to build a custom solution with OpenOffice.org Base. In this way, you can learn a few useful database techniques in the process, create an application that suits your specific needs, and re-purpose it easily for other duties, such as a simple document archiving tool or a document management solution.

To start, you want to create a new *.odb* database. As with any database, the first order of business is to create a table to store all the required data. First, switch to the *Tables* section and click on the *Create Table in Design View* link. Which fields you add to the table depends largely on what kind of data you want to store in the database (Figure 1). In this case, I want to add the following fields: *ID* (a primary key), *Title* (text field for storing article titles), *Publication* (text field to store publication names), *Notes* (text field to store miscellaneous notes), *DateSubmitted* (date field to store submission dates), *DatePublished* (date field to store publication dates), *FileName* (text field to store file names), and *FileObj* (binary field to store files). Once the table is ready, give it a name (e.g., *submissions*) and save it.

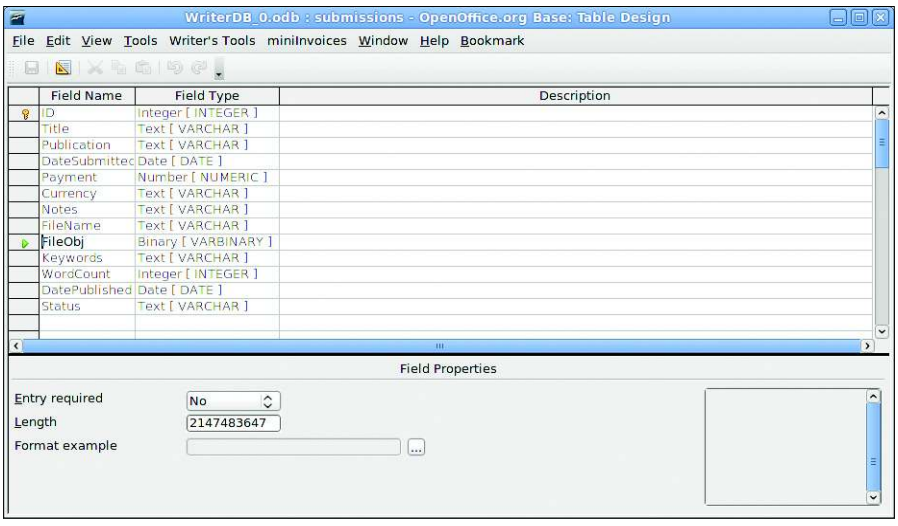

Figure 1: Designing the submissions table.

THE AUTHOR

Dmitri Popov holds a degree in Russian language and computer linguistics. He has been writing exclusively about Linux and open source software for several years, and his articles have appeared in Danish, British, North American, German, and Russian magazines and websites.

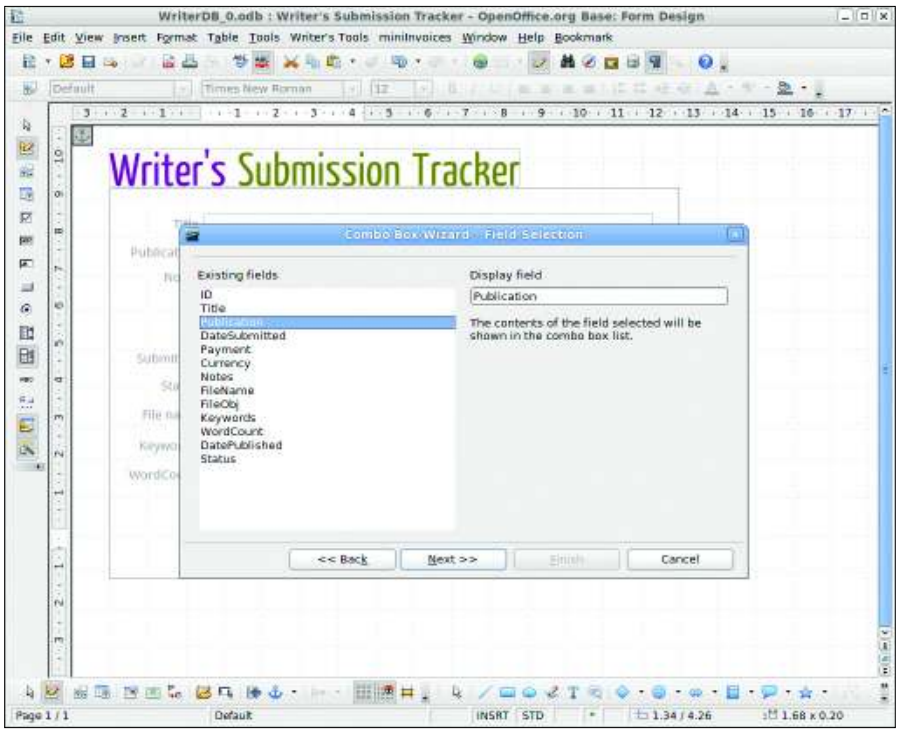

Figure 2: Using the Combo Box Wizard.

The next step is to create a form that will act as a graphical front end to the data stored in the *submissions* table. To create a new form, switch to the *Forms* section and click on the *Create Form in Design View* link. Creating a form with the Form Designer is rather straightforward. All you have to do is to add the fields to the form and then specify their properties. However, a couple of things require some additional work.

Because the *Publication* field is used to store the names of the publications you write for, it is most likely to contain only a handful of unique names. So instead of typing the same name every time you create a new record, you can add a combo box that lets you simply select the publication name from the list. To do this, choose the *Combo Box* control in the Form Controls toolbar, and draw a box in the form (see Figure 2). This opens the Combo Box Wizard, which helps you configure the created combo box.

By default, the created combo box is empty, so you have to enter the name of the publication you want. Once you've done that, the entry is saved in the database and it appears in the combo box.

The key feature of the submission tracker is the ability to store documents and files in the database. This feature requires two form elements: a file selection field and buttons to import and export the selected file. The buttons are used to trigger OpenOffice.org Basic macros that insert the selected file into the *FileObj* field and save the file in the specified directory on the hard disk. Both macros and related functions are written by Drew Jensen.

The code itself is too long to list here, but it's available as a neatly packaged OpenOffice.org Basic library [1]. To import the library, download and unpack the *AddSaveFile.zip* archive. Then, switch to the submission tracker form and choose *Tools | Macros | Organize Macros | OpenOffice.org Basic*. Now press the *Organizer* button, switch to the *Libraries* section and select the submission tracker form from the *Location* drop-down list.

Next, press the *Import* button and select the *AddSaveFile* library, then go back to the submission form and add two buttons to it: *Add File* and *Save File*. Now you have to assign the appropriate macro to each button.

Double-clicking on the *Add File* button opens the Properties window so you can switch to the *Events* section and assign the *onClickAddFile* macro to the *When initiating* action. In a similar manner, assign the *onClickWriteFile* macro to the *Save File* button.

Automating Data Entry

Your submission tracker is ready to go, but you still have room for improvement. For example, instead of manually entering information such as article title, description, notes, word count, and so on, you can create a simple macro that pulls this data from the currently active document and inserts it into the *submissions* table. This, of course, requires that the document contains the required information in the appropriate fields in *File | Properties*.

Also, you have to register the submission tracker database with OpenOffice.org. To do this, choose *Tools | Options* in the main toolbar, select *OpenOffice.org Base | Databases*, and press the *New* button. Select the submission tracker database and give the new connection a name (e.g., "SubTracker"). Press *OK* to save the settings and close the window. Now you can add the InsertMetadata macro (Listing 1). It starts by establishing a connection to the *SubTracker* database. The macro then uses an SQL query to insert data retrieved from the document properties and the document word count into the appropriate fields.

To make it easier to identify the status of the currently viewed submission quickly, you might want to add a feature that displays the "NOT YET PUBLISHED" marker on the records with an empty *DateSubmitted* field. To do this, you need to create an SQL query containing a calculation field whose value depends on a certain condition. In the submission tracker database, switch to the *Queries* section and click on the *Create Query in SQL View* link. Enter the following SQL query in the Query Design window (see Figure 3):

```
SELECT "ID", "DatePublished", ➥
CASEWHEN( LENGTH( "DatePublished" ) ➥
> 0, ' ', 'NOT YET PUBLISHED' ) ➥
AS "Status" FROM "submissions"
```

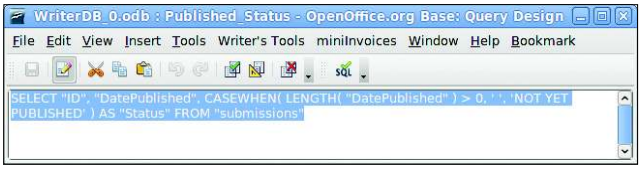

Figure 3: Writing the SQL query.

This query selects the *ID* and *DatePublished* fields from the *submission* table and sets the value of the *Status* field to *NOT YET PUBLISHED* if the length of the *DatePublished* field equals *0*. Now save and close the query, then open the submission tracker form for editing. Now you have to add a subform that is based on the created SQL query. To do this, turn the Form Navigator palette on, right-click on the main form, and choose *New | Form*.

Next, right-click on the inserted subform and choose *Properties*. Switch to the *Data* section, select *Query* from the *Content type* drop-down list, then select the created query from the *Content* drop-down list. The next step is to link the main form and its subform via the *ID* field. Click on the selection button next to the *Link master fields* entry and choose *ID* field in both columns. Do the same for the *Link slave fields* entry. Now you can add the *Status* text field to the subform. Make sure that the subform is selected in the Form Navigator and drag a text field control onto the form in the Form Designer window.

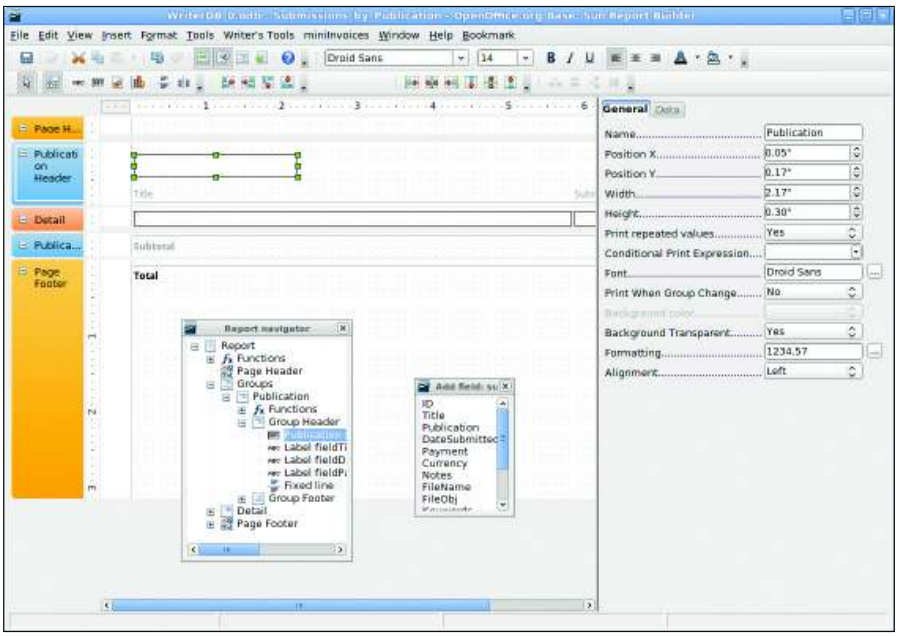

Figure 4: Creating the report with Sun Report Builder.

Adding a Report

If you have the Sun Report Builder extension [2] installed on your system, you can use it to create a report that can help you analyze data stored in the submission tracker database. To create the report, switch to the *Reports* section and click on the *Create Report in Design View* link. The blank report page is divided into three default sections: *Page Header*, *Detail*, and *Page Footer*. To create a report that groups all submissions by publication, you have to add a new group header page section by selecting the *Publication* field from the *Sorting and Grouping* palette. Specify the desired sorting option (*Ascending* or *Descending*) and select *Present* from the *Group Header* drop-down list.

Use the *Add Field* palette to place the *Publication* field inside the *Publication Header* section. Then you can use the available formatting options under the *General* tab in the *Properties* pane to format the field to your liking. The *Detail* section of the page is designed to generate a list of database records. In this case, the list will contain records related to a particular publication. Using the *Add Field* palette, add the desired fields (e.g., *Title*, *DateSubmitted*, and *DatePublished*) to the *Details* section. Pressing the *Execute Report* button in the main toolbar generates a list of all submissions grouped by publication, which you can use to preview the report.

Although you can use the created report as it is, you can do a few other things to make it even more useful. For example, you might want to add a field that calculates the sum of payments for each publication. To do this, click on the *Text Box* button in the *Report Controls* toolbar and draw a field in the *Page Footer* section.

In The *Properties pane*, switch to the *Data* section, and select the options as follows:

```
Data Field Type: Function
Data Field: Payment
Function: Accumulation
Scope: Report
```

Final Word

Although the described submission tracker is designed with writers in mind, you can use it as a foundation for creating other solutions. For example, turning the submission tracker into a document management application requires only a few simple tweaks.

Also, you can integrate the submission tracker into other databases. For instance, you can use it as a part of an invoicing solution or a project management system. ■

Listing 1: InsertMetadata macro

```
01 Sub InsertMetadata()
02
03 DBContext=createUnoService("com.sun.
   star.sdb.DatabaseContext")
04 DataSource=DBContext.
   getByName("SubTracker")
05 ConnectToDB=DataSource.GetConnection
   ("","")
06
07 SQLQuery="INSERT INTO
   ""submissions"" " + "(""Title"",
   ""Notes"", ""WordCount"") VALUES "_
08 + "('" + ThisComponent.DocumentInfo.
   Title + "','" + ThisComponent.
   DocumentInfo.Description + "','" +_
09 ThisComponent.WordCount + "')"
10
11 SQLStatement=Database.
   createStatement
12 Result=SQLStatement.executeQuery
   (SQLQuery)
13
14 Database.close
15 Database.dispose()
16 End Sub
```

INFO

[1] AddSaveFile: *bit.ly/zAl7e*

[2] Sun Report Builder: *extensions.services.openoffice.org/project/reportdesign*