# Forms, Datasets, Looping and Updating - a Simple Example

Posted At : January 5, 2011 2:20 PM | Posted By : Mark Kruger
Related Categories: ColdFusion

Recently my good friend and colleague Mike Klostermeyer - who everyone would recognize as a brilliant programmer and guru if he would just learn to blog - suggested that I include some simpler posts among my obscure troubleshooting play-by-plays. Here's one that most CF programmers have had to overcome at some point. Now before we go on I have to point out that there are 4 or 5 ways to do this - not counting things like Hibernate and the "black box" stuff that ships with many frameworks. What I'm illustrating here is the capabilities of the language. Moreover, if you have to support any legacy code (as 95% of us do) then you don't always have other options. You have to find a solution that works in context. With that in mind let's proceed.

Our problem stems from a typical form you might create as a part of an admin toolkit. Let's suppose you select multiple records from the database and load them into the form. When you submit the form you want all the values of the form fields to "update" the values in the table. Simple right? To start with we have 2 assumptions:

- Each row has a primary key - let's make it an "identity" field called "id" for simplicity sake.
- On submission we are simply going to update every record with the values. It's a small data set so I'm not optimizing here. I don't care if a value has changed or not - I'm just going to update it.

Now let's say we have the following data from the table "famousLargeMouths":

| ID | FirstName | LastName |
|---|---|---|
| 1 | Mick | Jagger |
| 1 | Angelina | Jolie |

The first task is to select all these records, load them into a form, and update them in a single action on submit.

## Beginning Attempt

Now a beginning programmer might do something like this:

```
<input type="hidden"
    name="totRecords"
    value="#qryLips.recordcount#"/>

<cfoutput query="qryLips">
ID: <input type="text" name="id" value="#id#"/>
firstname: <input type="text" name="firstname" value="#firstname#"/>
firstname: <input type="text" name="lastname" value="#lastname#"/><br>
</cfoutput>
<input type="submit" name="go"/>
```

That would give us a form that looks like this:

| | | |
|---|---|---|
| ID: 1 | firstname: Mick | lastname: Jagger |
| ID: 2 | firstname: Angelina | lastname: Jolie |

go

Submitting such a form would give you the following output:

| struct | |
|---|---|
| FIRSTNAME | Mick,Angelina |
| ID | 1,2 |
| LASTNAME | Jagger,Jolie |

To handle the values submitted we would do the following:

```
<cfloop from="1" to="#form.totRecords#" index="ct">
<cfquery name="update" datasource="mydsn">
    UDPATE      famousLargeMouths
    SET         firstname = '#listgetat(form.firstname,ct)#',
            lastname = '#listgetat(form.lastname,ct)#'
    WHERE     id = #listgetat(form.id,ct)#'>
</cfquery>
</cfloop>
```

That's pretty easy right? Anyone see an issue? Let's suppose we add an additional record for "Robert Downey, Jr.". Now our form looks like this:

| | | |
|---|---|---|
| ID: 1 | firstname: Mick | lastname: Jagger |
| ID: 2 | firstname: Angelina | lastname: Jolie |
| ID: 3 | firstname: Robert | lastname: Downey, |

go

Perhaps you see the problem right away. Robert Downey, Jr. has a comma in his name (We probably don't "need" the comma, but let's just say...). That means when we submit the results are going to look like this:

| struct | |
|---|---|
| FIRSTNAME | Mick,Angelina,Robert |
| ID | 1,2,3 |
| LASTNAME | Jagger,Jolie,Downey, Jr. |

Anyone? Bueller? The addition of the comma after "Downey" means the list length for "lastname" is now 4 and not 3. The code we have created is going to truncate Mr. Downey's name (though not of course his ego). We could add some JavaScript to tease out commas and replace them with pipe symbols before submission and then add a replace function to put them back in for our update query. That would work I suppose, but I suggest that we get away from the this *listGetAt()* approach altogether. It seems error prone.

**The Fix**

Here's my approach to this simple problem. Note, as I said before this is just *one way* to solve this issue. The point here is to get you thinking and add another arrow to your CF quiver. My approach would be to use independent form names. Instead of trusting a single form name and field to carry all the data for a database column, I'm going to create separate names that represent the "cell" (both the row and the column). Here's my new sample code:

```
<input type="hidden"
    name="allIds"
    value="#Valuelist(qryLips.id)#"/>

<cfoutput query="qryLips">
firstname: <input type="text" name="firstname_#id#" value="#firstname#"/>
firstname: <input type="text" name="lastname_#id#" value="#lastname#"/><br>
</cfoutput>
<input type="submit" name="go"/>
```

Now, when I submit my form the form values look like this:

| struct | |
|---|---|
| allIds | 1,2,3 |
| Firstname_1 | Mick |
| Firstname_2 | Angelina |
| Firstname_3 | Robert |
| Lastname_1 | Jagger |
| Lastname_2 | Jolie |
| lastName_3 | Downey, Jr. |

I've managed to encapsulate each value in its own form field which makes the use of commas a non issue. Now all I need to do is loop through it and update like so:

```
<cfloop list="#form.allids#" index="id">
<cfquery name="update" datasource="mydsn">
    UDPATE      famousLargeMouths
    SET         firstname = '#form["firstname_" & id]#',
            lastname = '#form["lastname_" & id]#'
    WHERE      id = #id#
</cfquery>
</cfloop>
```

This keeps me from various search and replace expressions. It also kind of reflects more clearly the "Row and column" organization of a table in the DB - so I personally find it easier to understand from the outset.

Ok, Muse readers, I know you have your keyboard at the ready. Let's hear how you solve this particular problem and why you think it's a better approach. I look forward to comments as always.