

Re: Identifying Duplicate Records

Source:

<http://www.tech-archive.net/Archive/Access/microsoft.public.access.formscoding/2008-02/msg01046.html>

- *From:* "gllincoln" <gllincoln@xxxxxxxx>
 - *Date:* Sun, 17 Feb 2008 14:39:57 -0800
-

Hi Peter,

The first order of business would be to determine what column of data is unique except when it isn't? <grin>

Phone number might be a good one, particularly if you force a uniform format on the phone numbers entered. Address is what you mentioned but nailing down dupe addresses isn't necessarily straightforward. Unless you are running your addresses through an address correction/validating scrub, accurately finding all of the duplicate street addresses can get complicated.

Managing the address data for a couple bulk mail campaigns makes one cynical regarding their fellow man's ability to accurately and/or consistently enter a postal address into a form; maybe I'm a little prejudiced on this subject. However, you might need to concatenate two or more columns into a single entity to get a pseudo primary key (for querying purposes), to double-check things. For example, if first and last names are two columns – you might use `FullName:[firstname] & " " & [lastname]` in the query designer or `' ' & [firstname] & ' ' & [lastname] & ' ' AS FullName` in SQL.

Whatever you use, you then can use the results to flag the dupes for review in one way or another, or bring up the dupe records to look at them.

The traditional dupe finder query looks be something like this:

```
SELECT * FROM mytable WHERE address=IN(SELECT address FROM mytable AS tmp GROUP BY address HAVING Count(*)>1);
```

This is a tricky little query to understand, from a beginner to intermediate point of view. I didn't come up with it on my own – Access 95/97 used to have a find duplicate records wizard. Deduping a set of records is such a common chore that I soon had that criteria phrase memorized.

If you are using the query designer – you would put

```
=IN(SELECT address FROM mytable AS tmp GROUP BY address HAVING Count(*)>1);
```

in the criteria box of the address column.

For those who don't quite get it (how this works) don't feel badly. I struggled with this one. Just had a mental block at first – I saw that it worked, I memorized it character by character and used it. Later, when I started using an occasional aggregating query and got stuck doing some elaborate reports, I began to understand GROUP BY and then 'I got it'.

Re: Identifying Duplicate Records

A quick overview for the slow learners like me.

Whatever you put inside the IN() (and we can have more than one item separated by commas) will be compared to the left-hand side and return true or false if we get a match or not.

5= IN(1,2,3) will be false, no match

5= IN(3,4,5) will be true, we have a match

To find our duplicate records we build a query inside the IN function that will return only those values that occur more than once in the table.

In plain English what our query is saying, give us all the addresses (SELECT address AS tmp) where we get more than one item (HAVING Count(*)>1) that is the same when we group them together (GROUP BY).

This query inside the IN – returns a list similar to our IN example above. When the query is running, each row's address column is being compared using the IN function to an array of dupes only addresses (dupeaddress1, dupeaddress2,dupeaddress3, etc)

Hope this helps...

Gordon

"Peter Hallett" <PeterHallett@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote in message
news:049A1BFD-A581-488C-AE0B-D2636A98AFE7@xxxxxxxxxxxxxxxxxxxx

I have a dynamically created and amended table which occasionally contains duplicate records. These duplicates must be allowed but need to be flagged-up, when they occur, and marked in the table. Setting the No Duplicates property is clearly inappropriate.

I can think of one or two ways of going about the job, including, perhaps

Re: Identifying Duplicate Records

running two nearly identical queries, one with the unique records property set and the other without, then comparing the results but the details of subsequently identifying and marking the appropriate entries have started to suggest some rather prolix VBA and I have a suspicion that I have missed something obvious or, at least, simpler. Has anyone got any thoughts?

—

Peter Hallett