Update queries in Access

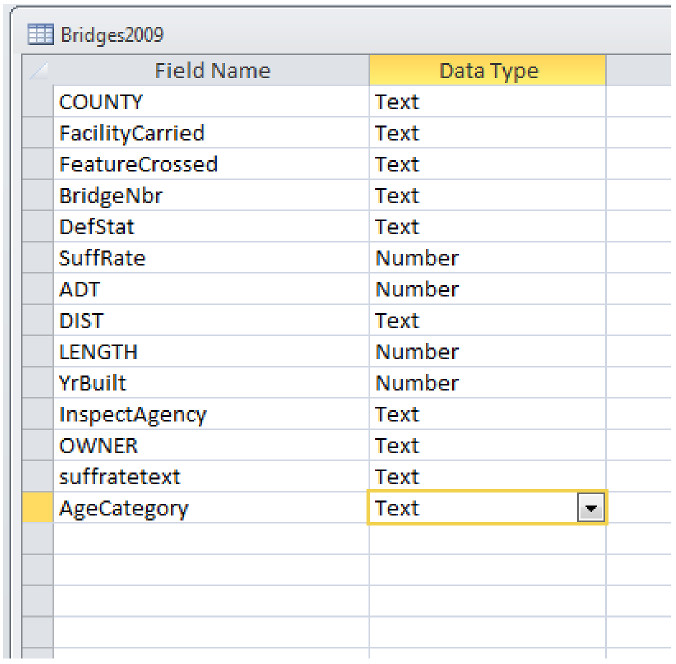
Use the Bridge Inspection Access database to follow along.

When you want to add a new column to your data table and populate those records with something, you need to use an Update query in Access. These allow you to tell Access to populate that new column with a particular value – either for all the records or for a subset or only those that meet a certain criteria. You can transfer a value from another field into this new one. You can do a math calculation and drop the answer into this new field. The possibilities are just about endless.

Step 1:

Go to the Design View of you table and add a new column. Be sure to set the "data type" (text, numeric, date, etc) accordingly. Then Save and close your table. (Note: If you want a field with decimals, choose "Number" and then go down to the bottom and change "long integer" to "double")

In the Bridges data, I've added "AgeCategory" as a text field. We're going to categorize the bridges into groupings based on the year they were built.

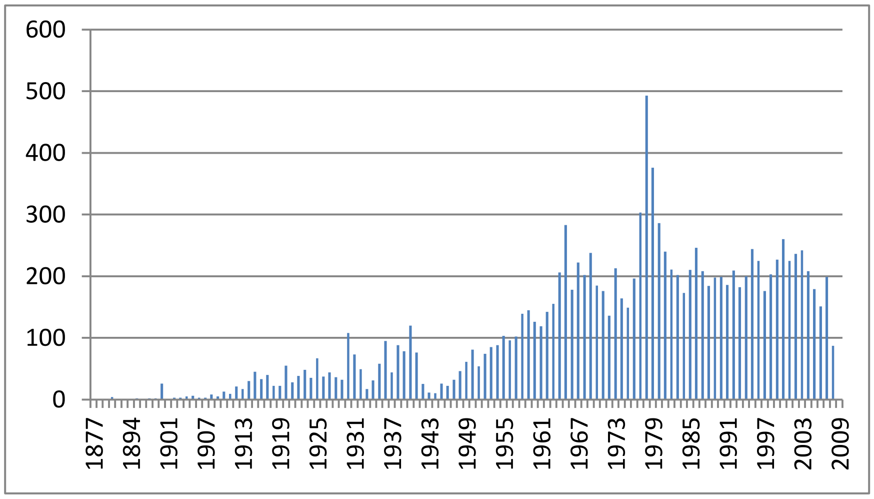

Bridges2009

Field Name	Data Type
COUNTY	Text
FacilityCarried	Text
FeatureCrossed	Text
BridgeNbr	Text
DefStat	Text
SuffRate	Number
ADT	Number
DIST	Text
LENGTH	Number
YrBuilt	Number
InspectAgency	Text
OWNER	Text
suffratetext	Text
AgeCategory	Text

Step 2:

Let's figure out our game plan.

In this case, I want to set the new column – AgeCategory – based on groupings of the years. To figure out the best way to group my records, I did a Group By query on the YrBuilt field, counting up how many bridges were built each year. I put that in



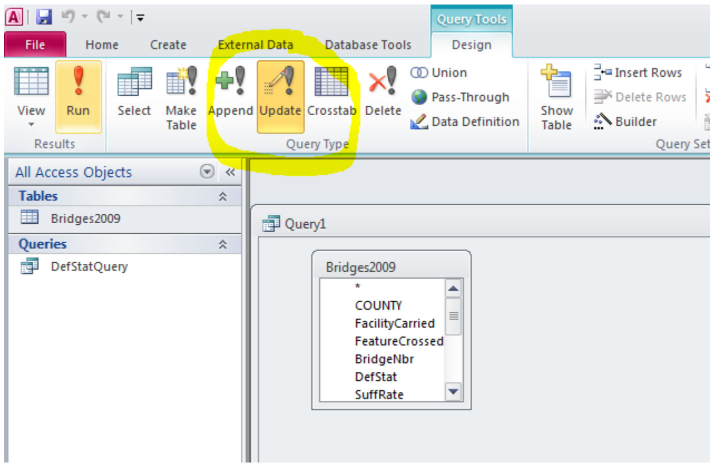
Excel and created this column chart to see the pattern.

We can see that the bulk of the bridges were built since the mid 1960s. So I'm going to make my groupings like this:

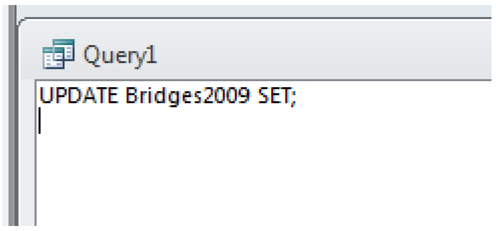
1877-1949 – Pre 1950
1950-1969 – 1950s-1960s
1970-1979 – 1970s
1981-1989 – 1980s
1990-1999 – 1990s
2000 to present – 2000s

Step 3:

Launch a new query based on your table. Stay in the Grid View/Design View window and go up to the menu bar and push the Update button to change this to an Update query.



Then you can flip over to SQL view.



```
UPDATE Bridges2009 SET;
```

You'll see that it has started to write the SQL for you.

Remove the semi-colon.

The syntax for a basic Update query that will do the same thing on all the records in the table follow this pattern:

UPDATE mytablename SET mynewfieldname = x

If you want the Update to run on only certain records – that meet a certain criteria – then you add a WHERE statement to the end, like this:

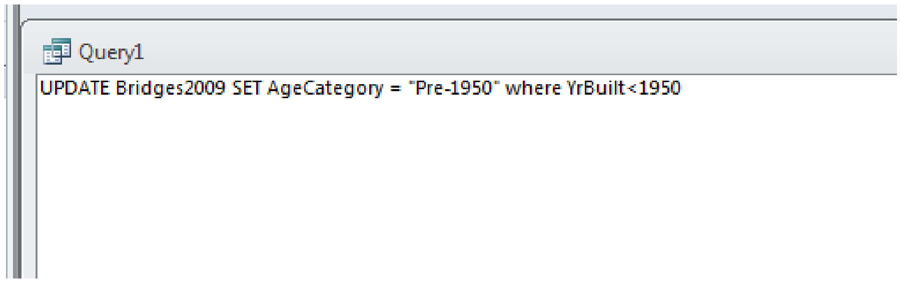
UPDATE mytablename SET mynewfieldname = X   WHERE ….

In this case our "mytablename" is Bridges2009

Our "mynewfieldname" is AgeCategory

And we're going to need the WHERE filter. In this example we're going to run separate queries for each age category that we're going to create. (Later in this handout, it shows you how to create a lookup table and then run just one update query)

So the first query will look like this:



```
UPDATE Bridges2009 SET AgeCategory = "Pre-1950" where YrBuilt<1950
```

This is setting that first age category. It's saying to populate our new field – AgeCategory – with the text "Pre-1950" (the quotes are needed to show Access that we are dropping a string of text in this field, not referring to a field name) for only the records where the YrBuilt is less than 1950.

Run this query. It should tell you that it's updating 1822 records. Hit Yes. You'll notice that it doesn't do anything after that. It's kind of disconcerting for beginners. But if you open your table and go look at the new field, you'll see that some of the records are populated. If you did it right, it should just be the oldest bridges. The others should still have blanks in that new column.

So now we'll go back to our SQL and change the syntax to populate the other categories. We'll need to expand our WHERE segment a bit to grab only those bridges that fall between two start and end periods.

UPDATE Bridges2009 SET AgeCategory = "1950s" where YrBuilt>=1950 and YrBuilt<1960

Here are the remaining queries, run each separately:

UPDATE Bridges2009 SET AgeCategory = "1960s" where YrBuilt>=1960 and YrBuilt<1970

UPDATE Bridges2009 SET AgeCategory = "1980s" where YrBuilt>=1980 and YrBuilt<1990

```
UPDATE Bridges2009 SET AgeCategory = "1990s" where YrBuilt>=1990 and YrBuilt<2000

UPDATE Bridges2009 SET AgeCategory = "2000s" where YrBuilt>=2000
```

UPDATE QUERIES WITH LOOKUP TABLES:

It's common to get a database that has lookup tables with it. The main data table might have one or more fields that have codes and the code descriptions are stored in the lookup table(s). To make things easier on you, there are times you might want to transfer the descriptions into the big table so you don't always have to join tables when you want to run queries.

Alternatively, there are situations like the one we just dealt with in bridges, where you want to make your own categories. In that case, you can make your own lookup table so that you just have to run one update query. This is especially useful if you have lots of categories.

Let's use the bridge example and do the same update, but with a lookup table.

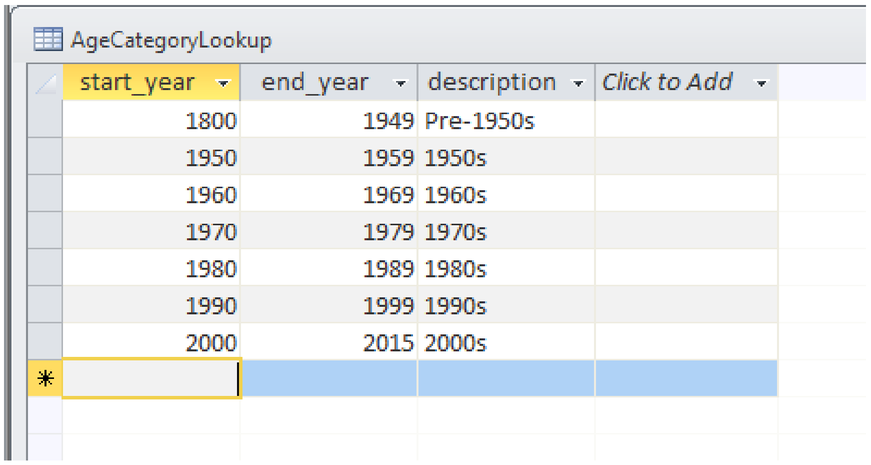
Step 1:

Build your lookup table. Go to the Create tab and choose "Table" to create a new table. You can flip over to design view to set your field names. (alternatively, you can build a lookup table in Excel and import it into your database)

I'm going to name my lookup table "AgeCategoryLookup". It's going to have three fields:

Start_year (set this as number)
end_year (set this as number)
Description (set this as text)

Save the table and return to the datasheet view. Then you can start typing in the table. You should end up with a table like this:


AgeCategoryLookup

start_year	end_year	description	Click to Add
1800	1949	Pre-1950s	
1950	1959	1950s	
1960	1969	1960s	
1970	1979	1970s	
1980	1989	1980s	
1990	1999	1990s	
2000	2015	2000s	

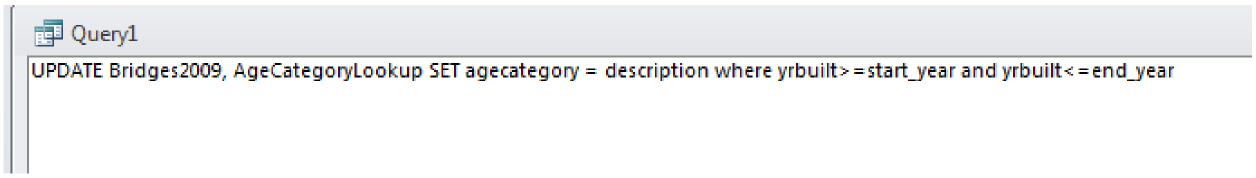
Step 2:

Launch a new query. First, add your Bridges2009 table. Then add your lookup table. (Note: The order that you add them is crucial. You must add the table you want to update first)

Change your query to an Update query (using the button in the Ribbon)

Flip to the SQL View.

We are going to "join" the tables for this one by linking them in the WHERE clause because we are matching the years with ranges in our lookup table.



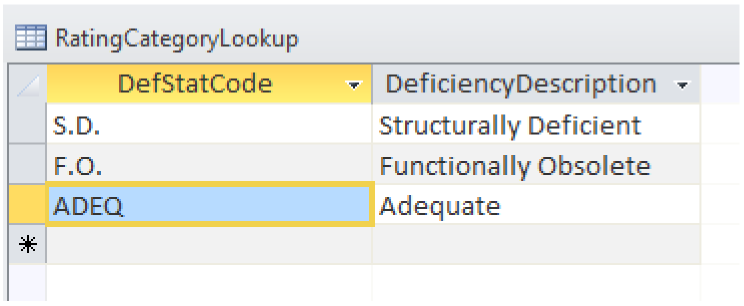
```
UPDATE Bridges2009, AgeCategoryLookup SET agecategory = description where yrbuilt>=start_year and yrbuilt<=end_year
```

Run this query and then go open your table. You should see that all the fields are now populated.

Note: If you have a situation where your table has a specific value (i.e. a county FIPS code) and your lookup table contains one record to match each possible code – i.e. you're matching exact value to exact value – then you can just join the table and the lookup table in the Grid View like you normally would when joining two tables.

In that situation your SQL update syntax would not need a WHERE clause (unless there is some other reason that you want to limit which records are affected by the Update).

I've created another lookup table that translates the DefStat (deficiency status) codes into phrases.

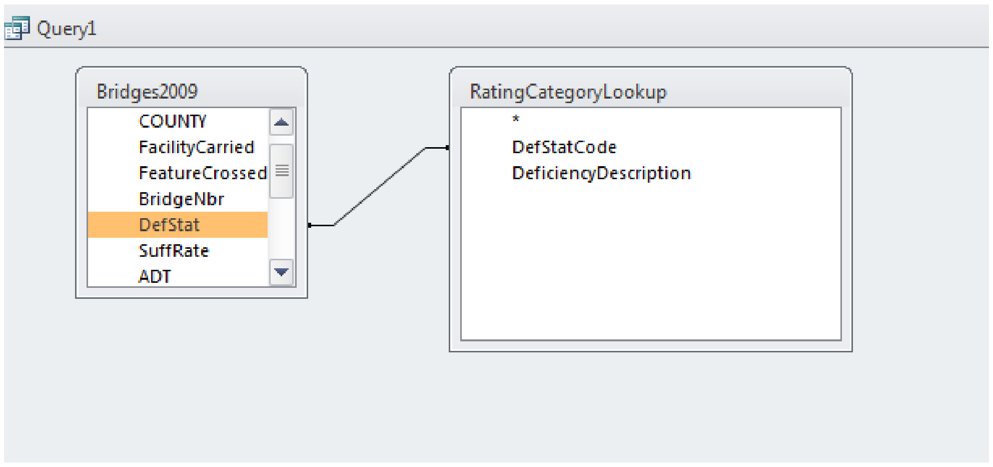


DefStatCode	DeficiencyDescription
S.D.	Structurally Deficient
F.O.	Functionally Obsolete
ADEQ	Adequate

Create a new field in your Bridges2009 table called "DefCategory". Set it as text.

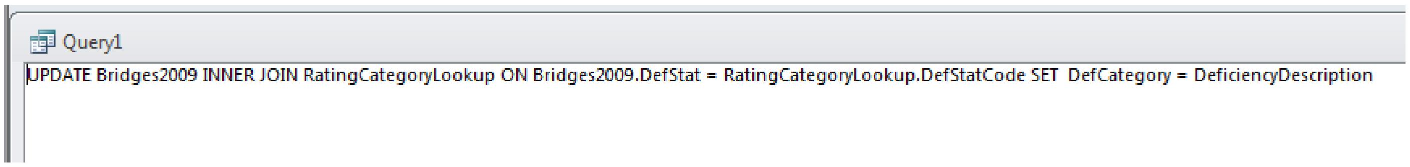
Start a new query and first put in your Bridges2009 table. Then put in this new lookup table – RatingCategoryLookup

In the Grid View, join the two tables by drawing a line between "DefStat" in the Bridges2009 table and "DefStatCode" in the lookup table.



Switch the query to an Update query.

Flip over to SQL. You'll see that it's written a lot of the SQL for you. You just need to remove the semi-colon and then finish the "SET" part of the SQL.



```
UPDATE Bridges2009 INNER JOIN RatingCategoryLookup ON Bridges2009.DefStat = RatingCategoryLookup.DefStatCode SET  DefCategory = DeficiencyDescription
```

Note that I'm telling it to set the "DefCategory" field (in my Bridges2009 table) with the field called "DeficiencyDescription"

Access is OK with that syntax because my tables have different field names (note that it's DefStat in the bridges table and DefStatCode in the lookup table). However, if you have field names that are the same in your tables, then you would need to beef up your SQL syntax, by putting the tablename in front of the fieldanames.

The previous query would look like this if you did that (I've highlighted all the spots where there is a table name as a prefix to a field name)

UPDATE bridges2009 INNER JOIN RatingCategoryLookup SET bridges2009.DefStat = RatingCategoryLookup.DefStatCode SET bridges2009.DefCategory = RatingCategoryLookup.DeficiencyDescription.

One last note: You can use update queries to change values in an existing column/field. However, I would instead recommend only doing this on new fields that you create – so that you maintain the original values. What if you make a mistake and have to undo it? Once you overwrite values, you can't undo. Having that original field makes it possible.

Created by MaryJo Webster
@MaryJoWebster
mjwebster71@gmail.com
January 2016