[excelheroacademy]

Module 5 - VBA Active Filter and EHDropLists

TOPICS COVERED:

1) Inspiration for the Advanced Filter Technique (0:07)
2) Worksheet_Change Event (3:32)
3) Advanced Filter Code (4:43)
4) Useless Named Ranges (8:45)
5) Filter Criteria - Wild Card Behavior (9:46)
6) Filter Criteria - OR logic (13:35)
7) Filter Criteria - BETWEEN Logic (14:49)
8) EHDropLists (16:34)
9) EHDropLists - CALL (21:40)
10) EHDropLists Routine (23:10)

Inspiration for the Advanced Filter Technique (0:07)

If VBA is not forbidden, one of the best techniques to use for filtering large datasets is the *AdvancedFilter* method of the *Range* object.
While the Advanced Filter is clumsy when accessed from the Excel controls, it is extremely elegant when accessed from VBA.

Credit goes to **SAM** of EHA1 who inspired Daniel to look further into developing this method.

We will discover in this module that with merely one line of code we can elegantly filter through hundreds of thousands of records instantly. In comparison, the UDF filter that we studied in this module requires several dozen lines of code, has less features and capabilities, and is several orders of magnitudes slower than the Advanced Filter.

Accessing these pre-built Excel features from VBA is, perhaps, the ultimate solution!

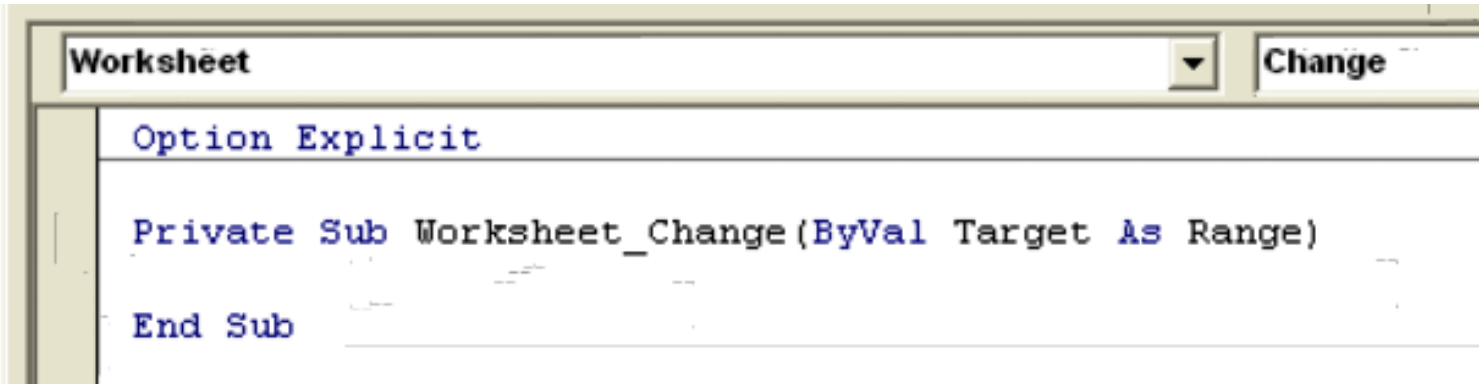
Setting Up Named Formulas

d: a dynamic range for the dataset on the "data" worksheet (28 columns X 65500 rows)

criteria_anchor: the first cell of the criteria range

filtered: the headers of the filtered result

Worksheet_Change Event (3:32)

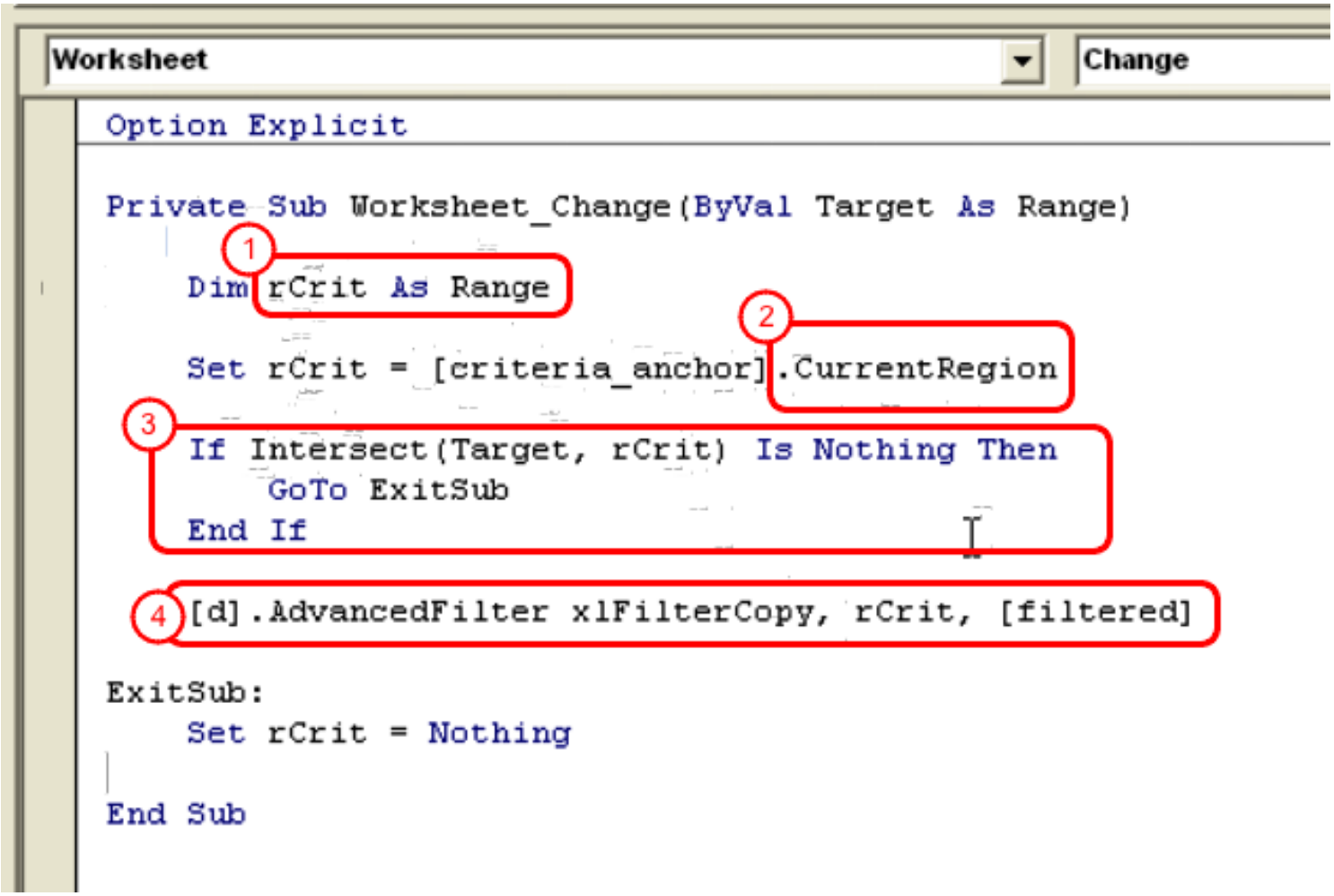


Code placed in the Worksheet_Change event will fire whenever anything changes on the worksheet, e.g. entering a value into a cell.

This event will house the Advanced Filter code.

[excelheroacademy]

Advanced Filter Code (4:43)



Stepping through the procedure, we are able to see how simple and elegant the code really is...

(1) The **rCrit** variable represents the criteria range

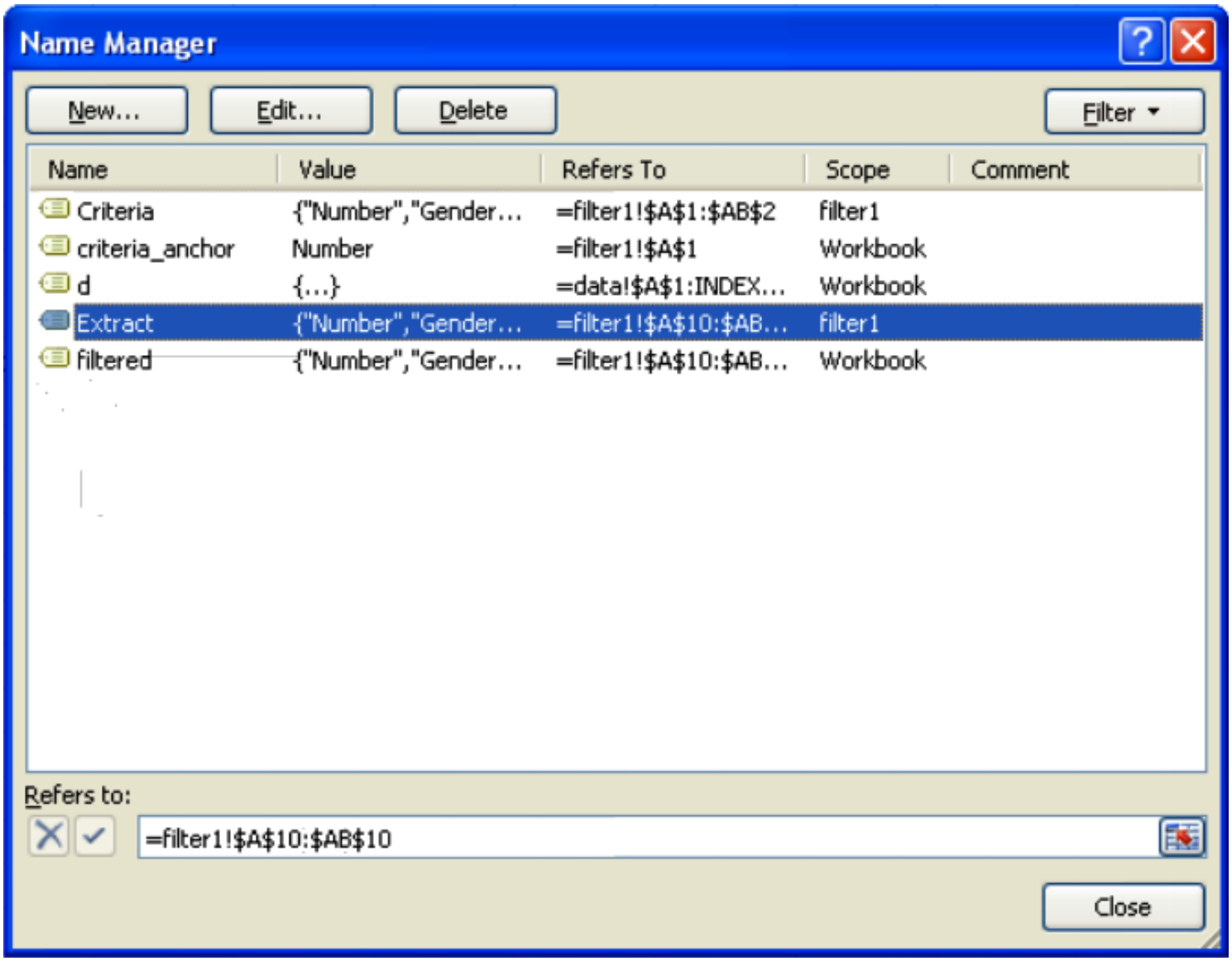
(2) rCrit will then be set to the full criteria range, defined by the *.CurrentRegion* property.
The CurrentRegion expands the initial range to include all the cells in use around it (bounded by blank rows/columns)

(3) The ***Intersect*** function allows us to determine if any changes have been made to the worksheet in the intersection of cells based on the parameters **rCrit** and **Target**, where Target represents the cell changed by the user in the worksheet.

(4) Run the **AdvancedFilter** method on the range, ***d***, with this one line of code.

The **xlFilterCopy** attribute allows filtered data to be copied to another area of the worksheet
The **rCrit** variable holds the CriteriaRange.
And the **[filtered]** property represents the CopyToRange, or the destination range for the copied filter data.

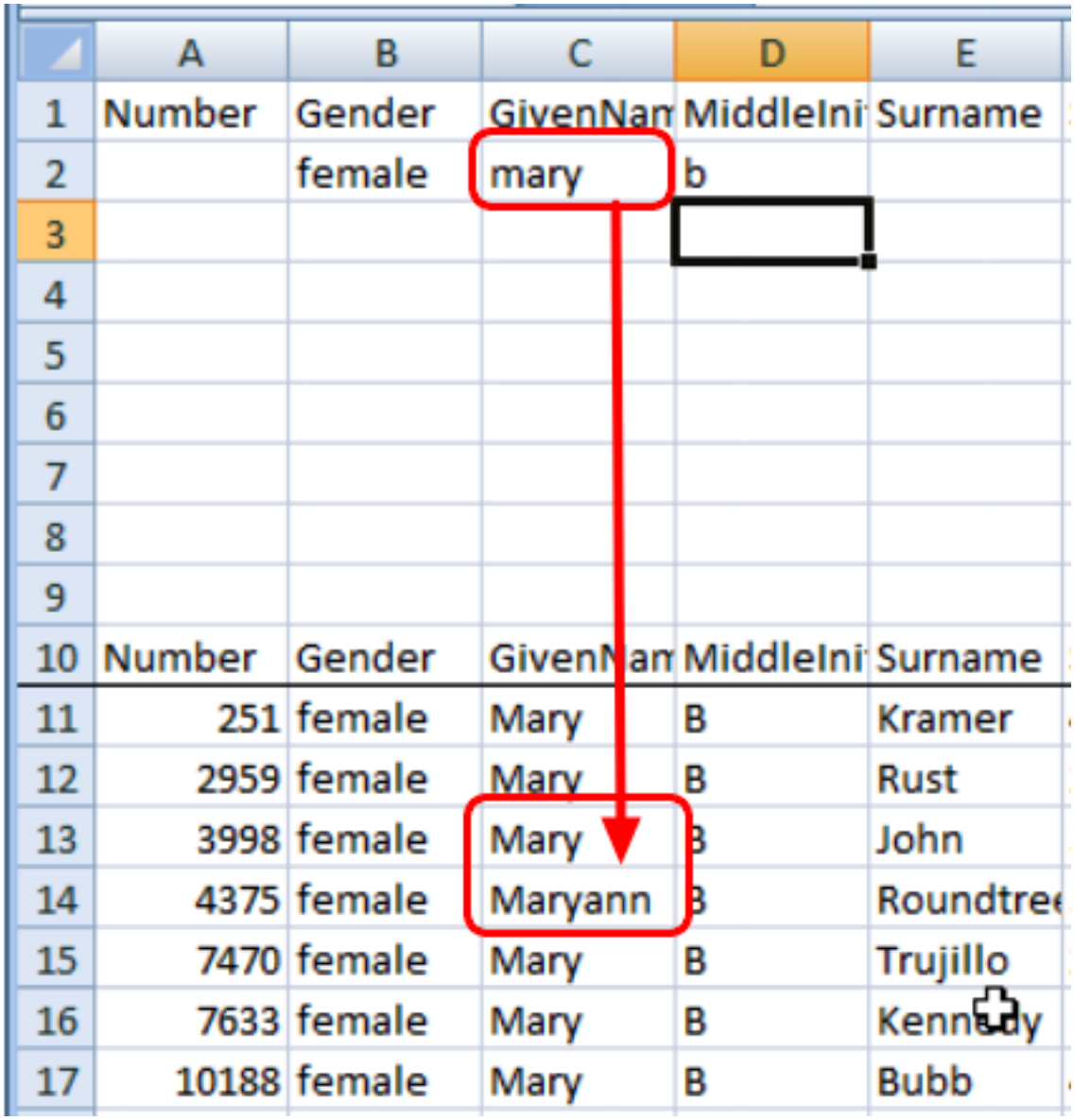
Useless Named Ranges (8:45)



When running the Advanced Filter method from VBA, Excel will always create two named formulas (Criteria and Extract). These are superfluous and are not needed.

[excelheroacademy]

Filter Criteria - Wild Card Behavior (9:46)



	A	B	C	D	E
1	Number	Gender	GivenNam	MiddleIni	Surname
2		female	mary	b	
3					
4					
5					
6					
7					
8					
9					
10	Number	Gender	GivenNam	MiddleIni	Surname
11	251	female	Mary	B	Kramer
12	2959	female	Mary	B	Rust
13	3998	female	Mary	B	John
14	4375	female	Maryann	B	Roundtree
15	7470	female	Mary	B	Trujillo
16	7633	female	Mary	B	Kennedy
17	10188	female	Mary	B	Bubb

By default, all advanced filter criteria are treated as if an asterisk (*), a wild card character, is appended to the end of the filter criteria. Note that both Mary and Maryann are returned for the search criteria **mary**. To Excel, there is no difference between searching for the term **mary** or **mary***.

So to search for a specific string of text like **mary**, you have to use an equals (=) sign. And so that Excel doesn't confuse this as a formula, you must precede the search string with an apostrophe ('). For example, **'=mary**.

**The advanced filter does work correctly on numerical quantities, though.

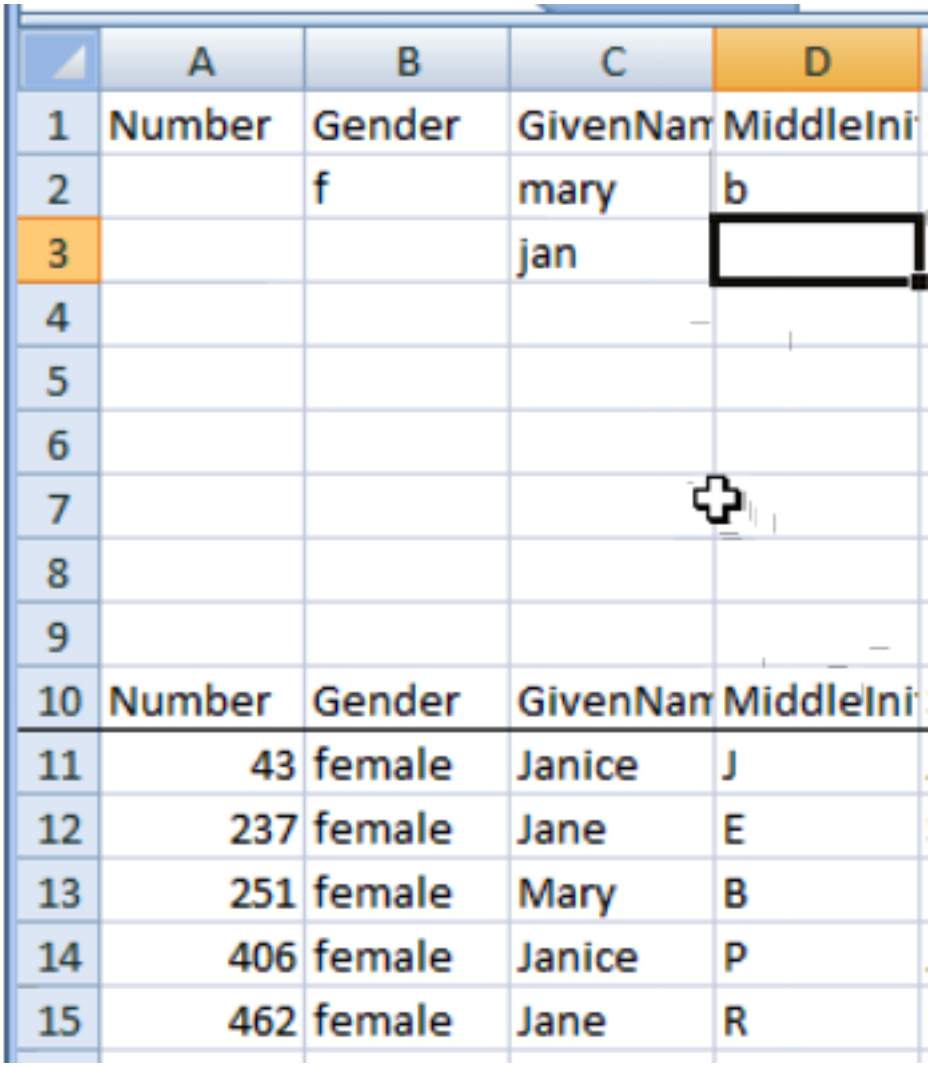
Other Examples:

? is the character length wildcard. So to find "Rosemary" for example, you could write **????mary**

* is the variable length wildcard. So **'=m*y** would return all names that begin with **m** and end in **y**.

~ is an escaping character which allows searching for the wildcard characters themselves, e.g. **~?** or **~***.

Filter Criteria - OR Logic (13:35)



	A	B	C	D
1	Number	Gender	GivenNam	MiddleIni
2		f	mary	b
3			jan	
4				
5				
6				
7				
8				
9				
10	Number	Gender	GivenNam	MiddleIni
11	43	female	Janice	J
12	237	female	Jane	E
13	251	female	Mary	B
14	406	female	Janice	P
15	462	female	Jane	R

To filter for records using **OR** logic, that is, search for **mary OR jan**, just use additional rows in the criteria range. Simply stack the search criteria as seen above.

Notice that all the Mary's in the search result have the middle initial of **B** but the **jan** results do not. This is because each row of search criteria is discrete. In order to view **jan** who also have the middle initial of **B** would require entering **B** as a search term in the **jan** row.

In case it was not obvious, **AND** logic spans *across* the columns, but always within the same row.

Filter Criteria - BETWEEN Logic (14:49)

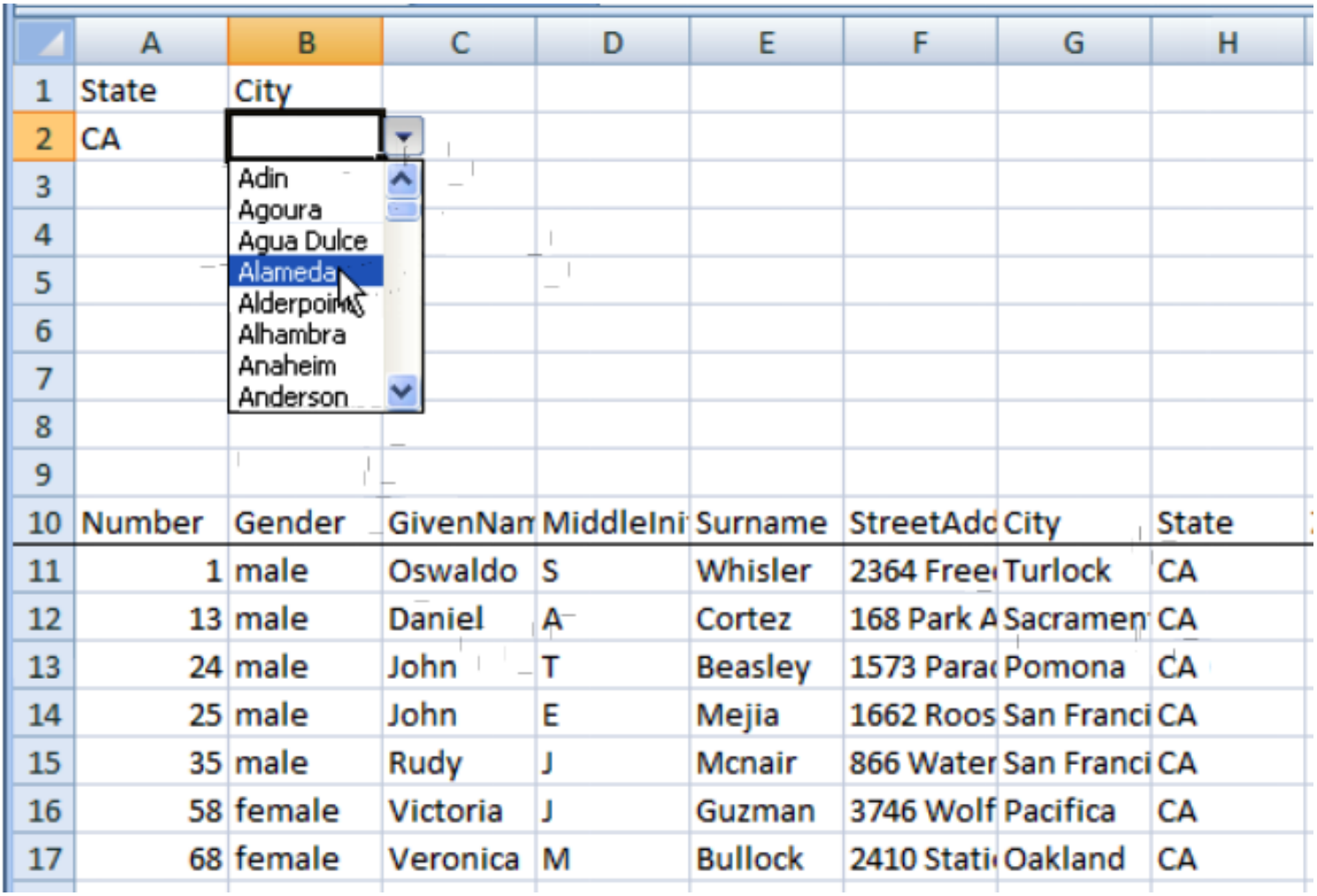
To produce a **BETWEEN** filter, we simply include the criteria column twice!

This second column can be placed anywhere within the criteria range, including at the end. Simply add one condition to the first criteria column and the other condition to the second criteria column.

*Note that if you are using **OR** logic in your filter, you must add the **BETWEEN** filter criteria to both rows of filter criteria to get the expected results.

[excelheroacademy]

EHDropLists (16:34)



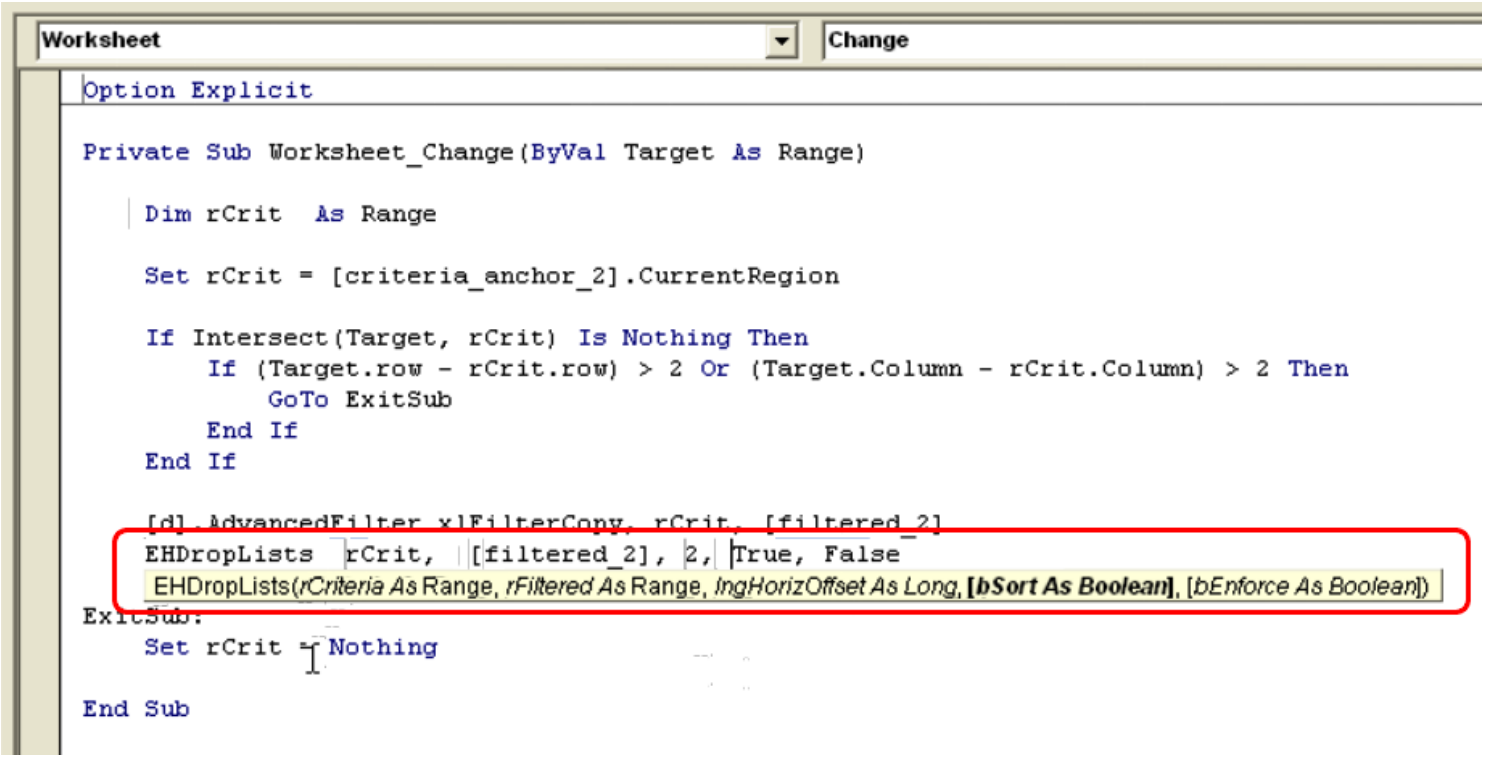
Number	Gender	GivenNam	MiddleIni	Surname	StreetAdd	City	State
1	male	Oswaldo	S	Whisler	2364 Free	Turlock	CA
13	male	Daniel	A	Cortez	168 Park A	Sacramen	CA
24	male	John	T	Beasley	1573 Para	Pomona	CA
25	male	John	E	Mejia	1662 Roos	San Franci	CA
35	male	Rudy	J	Mcnair	866 Water	San Franci	CA
58	female	Victoria	J	Guzman	3746 Wolf	Pacifica	CA
68	female	Veronica	M	Bullock	2410 Stati	Oakland	CA

The AdvancedFilter routine is very easy to use, extraordinarily powerful, and QUICK! And it makes a great case for creating dynamic, cascading dropdown lists. We will review the code module EHDropLists and add it to our growing code library.

As seen above, using EHDropLists, we can add any criteria we'd like (from the dataset, of course!). It allows the user to select ANY of the source columns in ANY order and in ANY number (for example with BETWEEN filters).

In the example illustration above, we have used *State* and *City* for our criteria. The *State* chosen was "CA." Therefore, when looking at the *City* criteria, the cascading functionality of the EHDropList shows only those cities represented in the dataset that are in California. The same will hold true for any additional columns that are used for filter criteria.

EHDropLists - CALL (21:40)

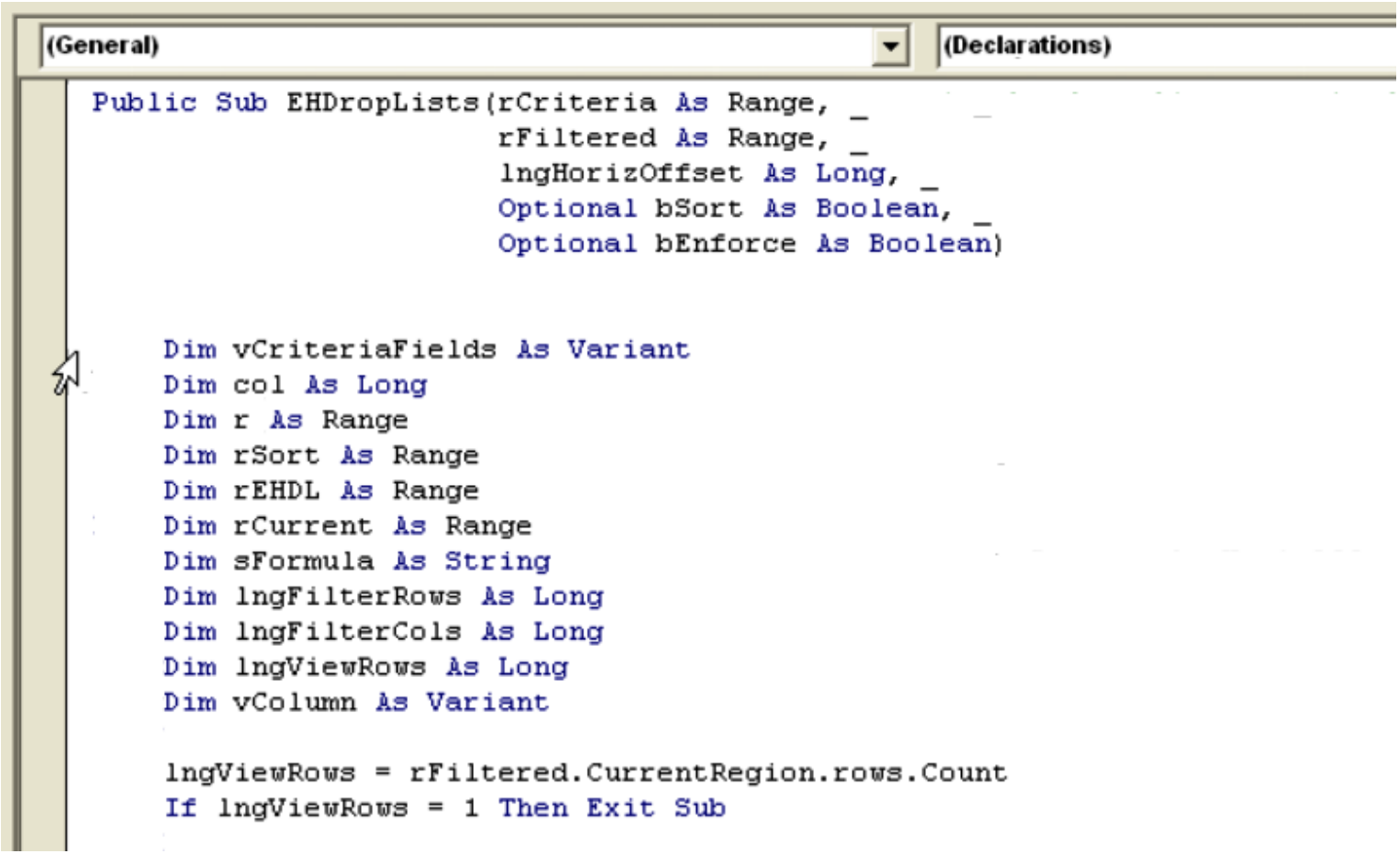


The call to the EHDropLists code is placed in the **Worksheet_Change** event.

Note that the function call requires several parameters. The criteria and filtered result ranges are both required. And two optional parameters, **bSort** for sorting the lists (default is TRUE), and **bEnforce** for determining whether values in the dropdown lists are required or not (default is FALSE).

[excelheroacademy]

EHDropLists Routine (23:10)



The EHDropLists code is worth stepping through to understand how it works to continue advancing your VBA skills.