Search in Google Sheets in Different Sheets  
With Up to 3 Parameters

1. To See the Generalized template go to: <https://docs.google.com/spreadsheets/d/1vL8FRdpASL1ZZAvaeR1QlVeRyNOIa0TRU3R9psJ76Eg/edit?usp=sharing>
2. This is a view only sharable that you can make a copy to and edit to your own needs
3. The spreadsheet has 2 main parts - Search and Combine:
4. Search: The search sheet (name: "SearchBy3Parameters") offers the user to enter up to 3 parameters to search/retrieve from one table.  
   The results are the intersection of all the parameters that had entered, meaning if only 1 parameter was entered, the results would show all the rows that (in the search column) has the parameter value contained in. If 2 or 3 parameters were entered, the results will be all the rows that contains (in the corresponding search columns) all the parameters.
5. Combine: The combine tables sheets (name: "ComboTable") is using a script to create a formula that stacks tables from different sheets into one table.  
   The assumption is that all the tables in all the sheets are starting in the same row and column, and has the same number of columns. The last row may vary between the table. The formula is dynamic so if values are added/removed from the table it would update automatically. In addition, the formula adds a column to the table columns to document the sheet name origin/source of the table. This is also dynamically updated if the user changes a sheet name.

Deeper Dive into The Combo Script

1. In the head of the script, we can find constants to help us customize the code in accordance to our needs. The constant EVERY\_TABLE\_RANGE holds the range in the format of $FIRST\_COL$FIRST\_ROW:$LAST\_COL.   
   Please note that:
2. There's no last row in order to keep the table dynamic.
3. The FIRST\_ROW is the first row excluding the headers, as we don’t want this to be mixed with the values.
4. I tried using a built in App-Script function (getDataRange()) in order to eliminate the need to keep all the tables in the search sheets in the same range, but that function returned ranges that are always starting from A1, therefore I gave up on that solution.
5. The script will combo all the tables from all the sheets, except from the sheets you explicitly mention (by name) to ignore in "excludeSheetNames" array.   
   Two default sheets to ignore are the search and combo sheets, and their names are entered in the constants at the beginning of the script.
6. The generated formula utilizes Google Sheets' ARRAYFORMULA and QUERY built in functions in order to return the stacked tables. To group the ranges, we utilize Google Sheets' curly brackets with a semi-colon to separate each table range. Also, to each range, there's a comma separator that is utilized to add a column to the table with the sheet name value. In order to make this column with as many rows as the table we utilize the built it ROW function (that returns a sequence array, under ARRAYFORMULA, in the length of the number of the rows).
7. The RIGHT and LEN function are used in order to clean the value in each row in the added column, so it will present only the sheet name value, without the numbers sequence returned from ROW.
8. The assembled function will be entered by default into the A1 cell in the COMBO\_SHEET\_NAME sheet.
9. Note that if a sheet is removed or added, the formula won't be updated automatically, therefore I've added a "Refresh" icon on both sheets to run the script that built the formula, when needed.

Here's an example of how the generated function would look like:

=ARRAYFORMULA(QUERY({  
TheBeatles!$B$5:$E, RIGHT(ROW(TheBeatles!$B$5:$E)&"TheBeatles", LEN("TheBeatles"));  
'Iron Maiden'!$B$5:$E, RIGHT(ROW('Iron Maiden'!$B$5:$E)&"Iron Maiden", LEN("Iron Maiden"));  
Queen!$B$5:$E, RIGHT(ROW(Queen!$B$5:$E)&"Queen", LEN("Queen"))  
}, "SELECT \* WHERE Col1 IS NOT NULL"))

Deeper Dive into The Search Sheet

1. The search sheet is made in order to search specific values, that might be located in multiple rows in a large table. Each parameter acts as a filter to the search results (if a parameter left empty, then it's not effective).
2. The search is done by using this Google Sheets function:

=IFERROR(  
QUERY(INDIRECT($B$1),  
"SELECT Col5, Col1, Col2, Col3, Col4 WHERE "&   
IF(ISBLANK($C$4),"", "LOWER("&$D$4&") LIKE LOWER('%"&$C$4&"%')") & IF(ISBLANK($C$5), "", IF(ISBLANK($C$4), "", " AND ") & "LOWER("&$D$5&") LIKE LOWER('%"&$C$5&"%')") & IF(ISBLANK($C$6), "", IF(AND(ISBLANK($C$4), ISBLANK($C$5)), "", " AND ") & "LOWER("&$D$6&") LIKE LOWER('%"&$C$6&"%')")&" ORDER BY Col1"),  
 "No Search Results")

1. The QUERY function is utilized to query the table and return the search results:
2. The table range is entered by string in cell B1, therefore the INDIRECT function is utilized to treat it as a range.
3. The returned columns and order of the column is dictated by the way they are presented after the SELECT keyword. The reason that in the example Col5 is the first column is because this is the "Source" column in the combo table.
4. At the end of the query, it utilizes the "ORDER BY" keywords to order the results by Col1. Other query keywords can be applied like "LIMIT" to limit the number of rows in the result.
5. The ampersand (&) symbol is used to concatenate strings in the query body.
6. The search sheet in the example is meant to be as generalized and dynamic as possible, so there are few cells that can be adjusted:
7. Cell B1 holds the range of the search table as a string, without the last row, to keep the range dynamic. You can either change the table range in B1 to the desired table you'd like to search in, or you can "hard code" the range in the query function (instead of " INDIRECT($B$1)")
8. The search parameters are in cells C4, C5 and C6. If you would like to use different cells, you'd have to change the formula so it will use the relevant cells accordingly.
9. Each parameter is compared against one column of the table. In D4, D5 and D6 cells the corresponding columns are written in the form of "Col" + <the column number to search>. You can keep it like that to allow the user a dynamic search in different columns, or to "hard code" the search columns in the formula, instead of the cells (so instead of "LOWER("&$D$4&") you will write "LOWER("Col3")).
10. In the example, the search allows partial match with no case-sensitivity. The partial match is allowed by using the query keyword LIKE (instead of the equal sign for exact match) and the wildcard precent symbol (%) to specify prefix and suffix (by using precent sign at the beginning and at the end it means we're looking for results that has the search value contained within).
11. The LOWER query function allows the result to not be case sensitive.   
    In my personal use, I want to also ignore spaces and special signs in the search string/result. So, if a table contains for example "Paul McCartney" and a user searched for "paul-mccartney" it will still return the result in the query. To accomplish that I create a custom/named function in Google Sheets with the name "FIX\_NAME" and it substitutes the signs and spaces with nothing and then it uses UPPER on the string. I apply the function inside the query for each parameter. To apply the function on the table I add 3 columns to the table, one column that is corresponding for each search column. At the first row I enter an ARRAYFORMULA to apply the FIX\_NAME function on all the values in the search column (so if I want to "fix" column D I will enter =ARRAYFORMULA(FIX\_NAME(D1:D)). Then I will change the query accordingly to compare the parameters with the corresponding fixed columns (and also delete the LOWER function from the query).
12. Here's an example of the FIX\_NAME function:

=UPPER(SUBSTITUTE(SUBSTITUTE(name, " ", ""),"-",""))

1. Here's an example of the formula with "hard-coded" columns and using the FIX\_NAME function:

=IFERROR( QUERY(COMBO!$A$1:$K, "SELECT Col8, Col1, Col2, Col3, Col4, Col7 WHERE "& IF(ISBLANK($F$2), "", "Col9 LIKE '%"&FIX\_NAME($F$2)&"%'") &   
IF(ISBLANK($F$3), "", IF(ISBLANK($F$2), "", " AND ")& "Col10 LIKE '%"&FIX\_NAME($F$3)&"%'") & IF(ISBLANK($F$4), "", IF(AND(ISBLANK($F$2), ISBLANK($F$3)), "", " AND ") & "Col11 LIKE '%"&FIX\_NAME($F$4)&"%'")&" ORDER BY Col9"), "No Search Results")