Excel: Find Unique Values in Non-Adjacent Columns

Authored by

Mohammed looti

November 14, 2025

In the realm of data analysis, identifying and extracting unique values is a fundamental task, especially when working with extensive Excel spreadsheets. While Excel provides a straightforward UNIQUE function for analyzing contiguous data ranges, the challenge intensifies significantly when these required values are spread across non-adjacent columns. This complex scenario frequently arises in large, unstructured datasets where critical information is separated by irrelevant or intermediate columns that must be ignored.

Fortunately, modern Excel versions offer a highly sophisticated and elegant solution by combining several powerful Dynamic Array Formulas. This article provides a precise, step-by-step guide to constructing a formula that efficiently tackles this specific challenge, enabling you to extract unique combinations from discontiguous ranges with exceptional ease and accuracy. By leveraging the synergistic power of the FILTER function, the UNIQUE function, and the SORT function, we can build a robust framework that bypasses the limitations of traditional, simpler approaches.

Understanding the Core Functions: FILTER, UNIQUE, and SORT

Before implementing the complex nested formula, it is crucial to understand the individual capabilities and roles of the constituent functions, as their combined synergy is what delivers this advanced solution. The FILTER function, a revolutionary tool introduced in modern Excel versions (Excel 365), is essential for sophisticated data manipulation. It permits the user to filter a range based on specific criteria, returning only the rows or, in our case, the specific columns that meet the defined conditions. Its standard syntax requires specifying the target array, the criteria (often a Boolean array), and an optional **if_empty** argument. For our purpose, FILTER's primary role is to isolate only the necessary columns from the overall range.

The UNIQUE function, another vital element of Excel's Dynamic Array suite, is engineered to return a distinct list of values from a specified range or array. This function is particularly powerful because it can identify unique items based on combinations of values across multiple columns simultaneously. Critically, UNIQUE will process the temporary output generated by the FILTER function--which has already isolated the desired non-adjacent columns--to accurately pinpoint only the distinct entries. This makes UNIQUE the cornerstone of our objective, ensuring we capture only the truly unique combinations.

Lastly, the SORT function serves to organize the final calculated output in a clear, logical, and highly readable format. While sorting is not strictly necessary for the technical identification of unique values, it drastically enhances the clarity and usability of the results, making subsequent analysis much easier. SORT accepts the processed unique array as input and arranges it based on specified column indices and sort orders (either ascending or descending). When these three functions are nested together, they construct a flexible, powerful, and complete framework for advanced data extraction tasks in Excel.

Deconstructing the Formula: Combining FILTER, UNIQUE, and SORT

To successfully extract unique values that are spread across non-adjacent columns, you must employ the following powerful, nested Dynamic Array formula. This concise expression executes a precise sequence of operations to achieve the desired outcome efficiently.

=SORT(UNIQUE(FILTER(A1:C11,{1,0,1})))

The logical flow of this calculation begins with the innermost function: the FILTER function. FILTER is responsible for isolating the specific columns required for the uniqueness check. In this example, **FILTER(A1:C11,{1,0,1})** operates on the data range from **A1** to **C11**. The critical binary array **{1,0,1}** dictates that only the first (Column A) and third (Column C) columns are retained, while the second column (B) is discarded. The result of this step is a new, temporary array containing only the selected data.

This filtered array is then passed directly to the UNIQUE function. UNIQUE processes this newly constructed array and returns a list where every row represents a distinct combination of values from the isolated columns A and C. This step is pivotal, as it successfully eliminates all row duplicates based solely on the criteria we defined, thereby identifying the unique pairs across the non-adjacent data.

Finally, the entire calculation is encased within the SORT function. SORT accepts the definitive list of unique combinations generated by the nested functions and arranges them in an orderly fashion. By default, SORT organizes the results starting with the first column in the filtered output in ascending order, ensuring the final presentation is clean, professional, and easy to interpret. This layered approach guarantees both accuracy in data isolation and optimization for visual clarity.

The Power of the FILTER Function's Binary Selector Array

A key element that grants this formula its flexibility is the second argument within the FILTER function: the binary selector array, represented in our example as **{1,0,1}**. This array acts as a customizable mask or boolean filter, precisely controlling which columns from your initial range are included in the subsequent uniqueness check.

The logic guiding this selector is remarkably straightforward yet powerful: each number within the curly brackets corresponds sequentially to a column in your original range, from left to right. A value of **1** instructs Excel to **include** the corresponding column in the filtered output, while a value of **0** dictates that the column must be **excluded**. This mechanism is what allows for the precise selection of non-adjacent columns. For instance, if your data spans columns A, B, and C, the array

{1,0,1} ensures that only columns A and C contribute to the identification of unique combinations, completely bypassing the data in column B.

To further clarify the structural mechanism using our example formula:

The first **1** in **{1,0,1}** compels the inclusion of the first column of the range **A1:C11**, which is Column A.
The subsequent **0** signals the mandatory exclusion of the second column in the range, Column B.
Finally, the second **1** mandates the inclusion of the third column, Column C.

This highly adaptable binary array empowers analysts to dynamically select virtually any configuration of non-adjacent columns simply by adjusting the sequence of **1**s and **0**s. This flexibility makes the formula an essential and adaptable tool for diverse data structures and analytical requirements.

Practical Application: Identifying Unique Combinations in a Dataset

Let us apply this powerful technique to a concrete, real-world example. Imagine you are tasked with managing a dataset containing information about a roster of basketball players. Each row includes the player's name (Column A), their team affiliation (Column B), and their playing position (Column C). The specific business requirement is to generate a list of all unique combinations of Player Name and Position, meaning the intermediate Team column must be ignored when determining uniqueness.

Suppose your initial dataset looks like the illustration below, showing repeated pairings of names and positions across various rows:

	A	B	C	D	E
1	**Team**	**Points**	**Position**		
2	Mavs	24	Guard		
3	Mavs	33	Center		
4	Mavs	39	Forward		
5	Mavs	40	Forward		
6	Rockets	25	Center		
7	Rockets	20	Guard		
8	Rockets	24	Guard		
9	Spurs	18	Forward		
10	Spurs	14	Center		
11	Spurs	16	Center		
12					
13					
14					
15					
16					

The critical challenge is that since we are defining uniqueness based on two columns that are not physically contiguous (A and C, skipping B), a simple application of the standalone UNIQUE function will not suffice. UNIQUE typically requires a single, unified range for its operation. Therefore, we must first utilize the FILTER function to dynamically construct a temporary array containing only columns A and C. This step effectively brings the non-adjacent data together, enabling the subsequent uniqueness check and demonstrating why the combined power of FILTER, UNIQUE, and SORT is indispensable for complex data analysis.

Implementing the Formula and Interpreting Results

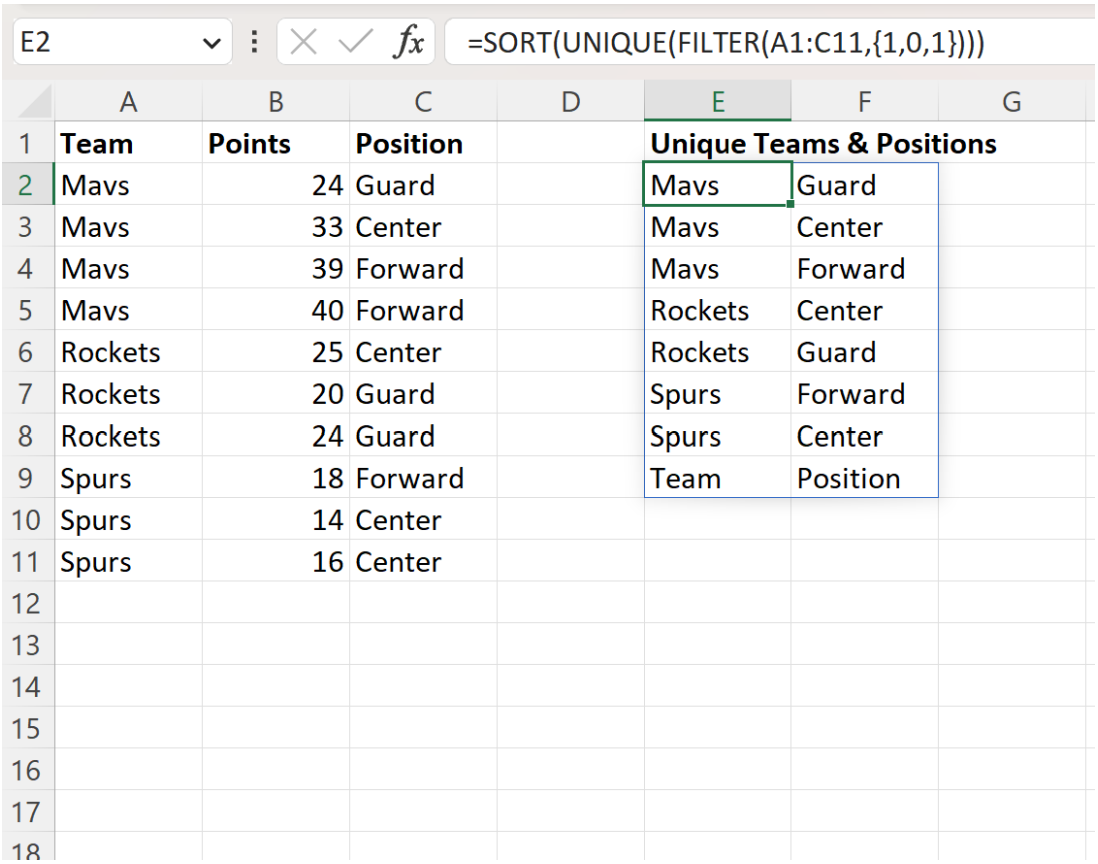
To achieve our specific goal--finding unique combinations of 'Player Name' (Column A) and 'Position' (Column C) from the sample data--we meticulously apply the nested formula discussed in the preceding sections. Assuming the player data occupies the range **A1:C11**, the formula, designed to select the first and third columns, is:

=SORT(UNIQUE(FILTER(A1:C11,{1,0,1})))

To implement this powerful solution, simply select an empty cell where you wish the unique results to begin spilling, such as cell **D2**. Type the formula exactly as written into the formula bar and press **Enter**. As this is a Dynamic Array Formula, the resulting unique list will automatically "spill" into the

adjacent cells below and to the right, instantly generating a distinct list of the unique player name and position pairings from your dataset.

Upon execution, you will observe that Excel instantly generates a new table containing only the unique pairs from the selected non-adjacent columns. This immediate visual feedback confirms that the formula has successfully isolated the specified columns, identified all distinct entries, and then presented them in a sorted order for optimal readability. The elegance of Excel's dynamic array capabilities shines through, streamlining what would traditionally be a multi-step, complex process into a single, efficient calculation.



	A	B	C	D	E	F	G
1	**Team**	**Points**	**Position**		**Unique Teams & Positions**		
2	Mavs	24	Guard		Mavs	Guard	
3	Mavs	33	Center		Mavs	Center	
4	Mavs	39	Forward		Mavs	Forward	
5	Mavs	40	Forward		Rockets	Center	
6	Rockets	25	Center		Rockets	Guard	
7	Rockets	20	Guard		Spurs	Forward	
8	Rockets	24	Guard		Spurs	Center	
9	Spurs	18	Forward		Team	Position	
10	Spurs	14	Center				
11	Spurs	16	Center				
12							
13							
14							
15							
16							
17							
18							

Extending the Solution: Handling Multiple Non-Adjacent Columns

The versatility of this formula is not limited to isolating just two non-adjacent columns. Its true strength lies in the highly customizable binary array argument within the FILTER function. While our primary example utilized **{1,0,1}** to select the first and third columns, this structure is fully adaptable. You can easily modify this array to include or exclude as many columns as your analysis requires, provided the array corresponds correctly to the overall range specified.

For example, if your dataset spans five columns (A, B, C, D, E) and your objective is to find unique combinations based on the values located in columns A, C, and E, your binary selector array would

simply become **{1,0,1,0,1}**. The fundamental rule is to ensure that the number of **1**s and **0**s in your array exactly matches the total number of columns in your specified data range (in this case, five). A **1** includes the column, and a **0** excludes it.

This adaptable and systematic approach is particularly useful in large, complex spreadsheets where relevant identifying attributes might be scattered across various sections. By mastering the construction of this flexible binary array, you unlock the full analytical potential of this formula, establishing it as an indispensable tool for advanced data cleaning and targeted selection within Excel.

Enhancing Your Workflow: Tips and Best Practices

While the formula for identifying unique values in non-adjacent columns is exceptionally powerful, adhering to certain best practices will further enhance your workflow and minimize potential errors. Always ensure that your primary data range (e.g., **A1:C11**) is specified correctly. An inaccurate range definition can lead to either critical data omission or the inclusion of irrelevant columns. Furthermore, meticulously verify the binary array used within the FILTER function; the count of **1**s and **0**s must correspond precisely to the total number of columns in your specified range to prevent formula calculation errors.

For managing very large datasets, continuous recalculation inherent in Dynamic Array Formulas can sometimes affect spreadsheet performance. If you experience performance degradation, consider converting the resulting unique list to static values by copying the output and pasting it back as values. This simple action can free up significant calculation resources. Additionally, it is important to remember that these dynamic array functions are features of modern Excel for Microsoft 365. Users operating older Excel versions will need to rely on traditional methods, such as utilizing helper columns, complex **INDEX/MATCH** combinations, or Advanced Filter tools to achieve similar results.

Finally, always prioritize data integrity. Before implementing any complex formula, it is strongly recommended practice to back up your original dataset or work exclusively on a duplicate copy. A deep understanding of the nuances of each function and their interaction is essential for effective troubleshooting and ensuring the accuracy of your extracted data.

Conclusion and Further Exploration

Mastering the methodology for finding unique values in non-adjacent columns represents a significant elevation of your data manipulation skills in Excel. The synergistic combination of the FILTER function, UNIQUE function, and SORT function provides an efficient, elegant, and reliable solution to a pervasive data challenge, enabling you to derive precise insights from even the most complex and unstructured datasets.

By understanding how to correctly construct and modify the binary array within the FILTER function, you gain unparalleled flexibility in selecting specific columns, regardless of their location within the spreadsheet. This technique is a powerful testament to the continually evolving capabilities of modern Excel and empowers users to perform sophisticated data transformations with remarkable ease. We strongly encourage readers to experiment with different column selections and larger datasets to fully grasp the efficiency and capability of this powerful combined formula.

Continue to explore the vast array of functions Excel offers. The deeper you delve into its capabilities, particularly its dynamic array functions, the more streamlined and effective your data analysis tasks will become. Continuous professional learning and dedicated practice are the definitive keys to becoming proficient in leveraging Excel for complex data challenges.

Additional Resources

The following tutorials explain how to perform other common tasks in Excel: