

# Sort IP Addresses in Excel (With Example)

Authored by  
**stats writer**

November 17, 2025

## RECOMMENDED CITATION

stats writer (2025). *Sort IP Addresses in Excel (With Example)*. PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=94514>

## The Challenge of Sorting IP Addresses in Excel

Sorting lists of data is one of the most fundamental operations performed in a spreadsheet program like Excel (1/5). When dealing with standard numerical or text data, the built-in sorting mechanisms work flawlessly. However, difficulties arise when attempting to organize data formats that Excel does not recognize as standard numeric values, such as IP addresses (1/5). An IP address (2/5) is technically composed of four octets, separated by periods, and although it represents a numeric hierarchy, Excel treats it merely as a string of text.

This discrepancy between how the data is structured (hierarchical numbers) and how Excel (2/5) interprets it (text string) leads to incorrect sorting results. When the default **Sort** function is applied, it performs an alphabetical or lexicographical sort (1/5), comparing character by character from left to right. This means that an address starting with "192." might incorrectly be placed after an address starting with "2." simply because the character '1' precedes '2' in numerical sorting, but not necessarily in the context of the full numeric value of the octet.

To achieve a true, numerically hierarchical sort (2/5) for these network identifiers, we must bypass the default text-based sorting mechanism. This requires transforming the dot-separated string into a single, massive integer that accurately reflects the hierarchical order of the four octets. This transformation involves advanced string manipulation and numeric conversion functions within Excel (3/5), culminating in the use of a specialized formula and a temporary **helper column** to facilitate the correct ordering.

## Why Standard Excel Sorting Fails

When you instruct Excel (4/5) to sort (3/5) a list of IP addresses (3/5), it treats the entry--for example, "192.168.1.100"--as four separate components delimited by periods. However, because it is recognized primarily as text, the comparison starts from the first character. This is known as lexicographical sorting.

Consider two different IP addresses (4/5): 10.1.1.100 and 2.255.255.255. Numerically, 2.255.255.255 is clearly smaller than 10.1.1.100. However, in a standard text sort (4/5), Excel compares '1' (from 10) and '2' (from 2). Since '1' comes before '2', it would incorrectly place 10.1.1.100 before 2.255.255.255, completely violating the logical numerical hierarchy inherent in the addressing scheme.

This limitation forces advanced users to employ mathematical manipulation to force a correct numerical interpretation. The core issue lies in variable length: the octets can range from 0 to 255. An octet of '5' is shorter than '150', yet both represent distinct numerical values. Without standardizing the length of these components, Excel's (5/5) default algorithm cannot determine the true numerical magnitude of the full address string. This is why we need techniques like zero

padding to ensure every octet is represented by the same number of digits.

### Example Scenario: Incorrect Default Sort

Suppose we have the following column of IP addresses (5/5) in Excel, located in Column A. This list is intentionally unordered to demonstrate the need for a specialized sorting mechanism:

|    | A                 | B | C | D |
|----|-------------------|---|---|---|
| 1  | <b>IP Address</b> |   |   |   |
| 2  | 122.45.65.90      |   |   |   |
| 3  | 122.108.99.155    |   |   |   |
| 4  | 122.230.33.192    |   |   |   |
| 5  | 138.192.34.93     |   |   |   |
| 6  | 184.234.223.94    |   |   |   |
| 7  | 200.89.43.80      |   |   |   |
| 8  | 213.45.67.1       |   |   |   |
| 9  |                   |   |   |   |
| 10 |                   |   |   |   |
| 11 |                   |   |   |   |
| 12 |                   |   |   |   |
| 13 |                   |   |   |   |
| 14 |                   |   |   |   |

If we were to highlight the cell range **A1:A8** and then click the standard **Sort** button found in the **Sort & Filter** group on the **Data** tab, instructing Excel to sort (5/5) these addresses from smallest to largest, the results would be misleading and functionally useless for network management or data analysis.

After applying the default sort, the data might rearrange into an order that looks alphabetically correct but numerically wrong:

|    | A                 | B | C | D |  |
|----|-------------------|---|---|---|--|
| 1  | <b>IP Address</b> |   |   |   |  |
| 2  | 122.108.99.155    |   |   |   |  |
| 3  | 122.230.33.192    |   |   |   |  |
| 4  | 122.45.65.90      |   |   |   |  |
| 5  | 138.192.34.93     |   |   |   |  |
| 6  | 184.234.223.94    |   |   |   |  |
| 7  | 200.89.43.80      |   |   |   |  |
| 8  | 213.45.67.1       |   |   |   |  |
| 9  |                   |   |   |   |  |
| 10 |                   |   |   |   |  |
| 11 |                   |   |   |   |  |
| 12 |                   |   |   |   |  |
| 13 |                   |   |   |   |  |
| 14 |                   |   |   |   |  |

As demonstrated in the resulting image, the IP addresses are not sorted correctly from smallest to largest based on their true numerical value. For instance, the IP address **122.45.65.90**, which has a smaller numerical value than 192.168.1.100, should logically appear much earlier in the list. The failure of the built-in sort tool confirms that a robust solution involving formula-based data transformation is necessary to handle this specific data type.

## The Concept of Zero Padding and Helper Columns

The key to resolving the sorting issue is standardizing the length of each octet. Since the maximum value an octet can hold is 255 (a three-digit number), we must ensure that every single octet in every IP address is treated as a three-digit number. This process is called **zero padding**. For example, if an octet is '5', we convert it to '005'. If it is '45', we convert it to '045'. If it is '150', it remains '150'.

Once each of the four octets has been zero-padded, we concatenate them (join them end-to-end) without the periods. The address 122.45.65.90 would become the 12-digit string "122045065090". This 12-digit string now represents the true numerical order of the original IP address, allowing for accurate sorting.

We implement this transformation using a helper column (1/5). A **helper column** is a temporary column added next to the dataset specifically to perform complex calculations or intermediate data transformations that cannot be handled directly by the sorting feature. The formula we construct will live in this helper column (2/5), converting the text IP string into a sortable numeric value. This value is then used as the primary key when executing the multi-column sort.

## Deconstructing the IP Sorting Formula

The complete formula used to create the sortable numeric key is complex and leverages several powerful string and number manipulation functions available in newer versions of Excel (those supporting dynamic arrays and modern text functions). We need to break down the IP string (e.g., A2) into its four constituent octets using functions that search for and extract text based on the delimiter (the period ".").

The primary functions utilized are **TEXTBEFORE**, **TEXTAFTER**, **TEXT** (for zero padding), **&** (for concatenation), and **NUMBERVALUE** (for final conversion).

The formula, intended for cell **B2** (assuming A2 contains the first IP address), is as follows:

```
=NUMBERVALUE(TEXT(TEXTBEFORE(A2,".",1),"000")&TEXT(TEXTBEFORE(TEXTAFTER(A2,".",1),".",1),"000")&TEXT(TEXTBEFORE(TEXTAFTER(A2,".",2),".",1),"000")&TEXT(TEXTAFTER(A2,".",3),"000"))
```

Let's break down how this formula handles the four octets:

**First Octet Extraction and Padding:** **TEXT(TEXTBEFORE(A2,".",1),"000")** extracts everything before the first period (.), and the **TEXT** function formats this extracted number to ensure it has three digits (e.g., 122 -> 122, 5 -> 005).

**Second Octet Extraction and Padding:** This is more complex, requiring nested functions: **TEXTAFTER(A2,".",1)** first strips off the first octet and the period, leaving "45.65.90". Then, **TEXTBEFORE(...,".",1)** extracts the '45' from the remainder. Finally, **TEXT(...,"000")** pads it.

**Third Octet Extraction and Padding:** This follows a similar nested pattern, using **TEXTAFTER(A2,".",2)** to isolate the last two octets ("65.90") before extracting the third octet ('65') using **TEXTBEFORE** and padding it.

**Fourth Octet Extraction and Padding:** **TEXT(TEXTAFTER(A2,".",3),"000")** is the simplest, extracting everything after the third period and padding it.

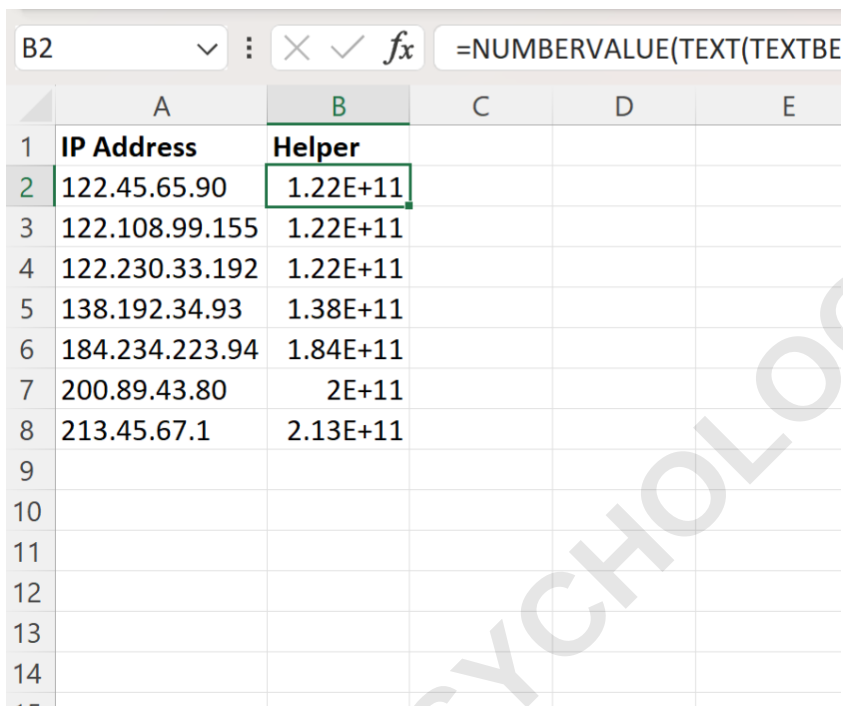
**Final Conversion:** The four padded strings are concatenated using the **&** operator, resulting in the 12-digit text string (e.g., "122045065090"). Finally, the entire concatenated text string is wrapped in the **NUMBERVALUE** (1/5) function, converting the 12-digit string into a true numeric value that can be correctly sorted by Excel.

## Implementing the Helper Column and Formula Application

After entering the comprehensive formula into cell **B2**, we must apply this logic consistently across

the entire dataset. Since we used relative referencing (A2), we can simply click and drag the fill handle (the small square at the bottom right corner of the cell) down to each remaining cell in Column B corresponding to our list of IP addresses in Column A.

This action populates the **Helper** column (B) with the large, sortable 12-digit integer representation for every helper column (3/5) row. This intermediate step is critical because Excel will now use these generated numbers, rather than the original text strings, when determining the sort order.



|    | A                 | B             | C | D | E |
|----|-------------------|---------------|---|---|---|
| 1  | <b>IP Address</b> | <b>Helper</b> |   |   |   |
| 2  | 122.45.65.90      | 1.22E+11      |   |   |   |
| 3  | 122.108.99.155    | 1.22E+11      |   |   |   |
| 4  | 122.230.33.192    | 1.22E+11      |   |   |   |
| 5  | 138.192.34.93     | 1.38E+11      |   |   |   |
| 6  | 184.234.223.94    | 1.84E+11      |   |   |   |
| 7  | 200.89.43.80      | 2E+11         |   |   |   |
| 8  | 213.45.67.1       | 2.13E+11      |   |   |   |
| 9  |                   |               |   |   |   |
| 10 |                   |               |   |   |   |
| 11 |                   |               |   |   |   |
| 12 |                   |               |   |   |   |
| 13 |                   |               |   |   |   |
| 14 |                   |               |   |   |   |

As visible in the image, the helper values are generated. Notice how 122.45.65.90 yields a smaller number (122045065090) than 192.168.1.100 (192168001100), accurately reflecting the numerical hierarchy that Excel previously failed to recognize. The consistency introduced by the zero padding ('001' instead of '1') is what makes the subsequent sorting possible and accurate.

## Executing the Custom Sort Operation

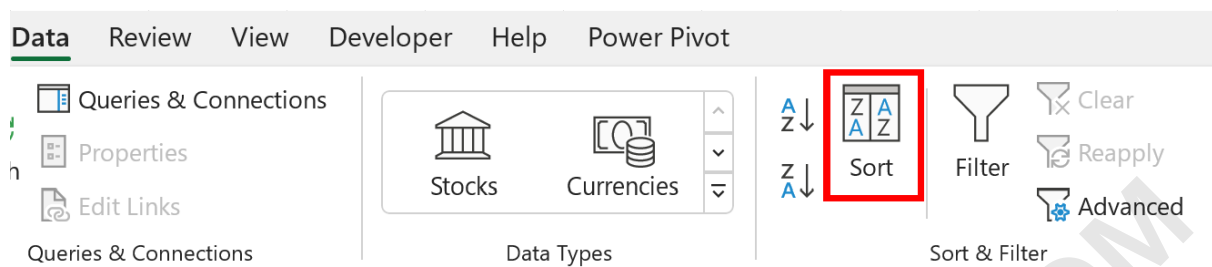
With the sortable keys generated in the helper column (4/5), we are ready to perform the final sort. The key difference here is that we must include both columns (A and B) in our selection range, as the sort relies on the values in Column B to correctly order the corresponding entries in Column A.

Follow these steps precisely:

Highlight the entire cell range containing both the original IP addresses and the helper values--in our example, this is **A1:B8** (including the headers).

Navigate to the **Data** tab along the top ribbon.

Click the **Sort** button located within the **Sort & Filter** group.



A new 'Sort' dialogue box will appear, allowing us to define the sorting criteria based on the available columns.

## Finalizing the Sort Criteria and Results

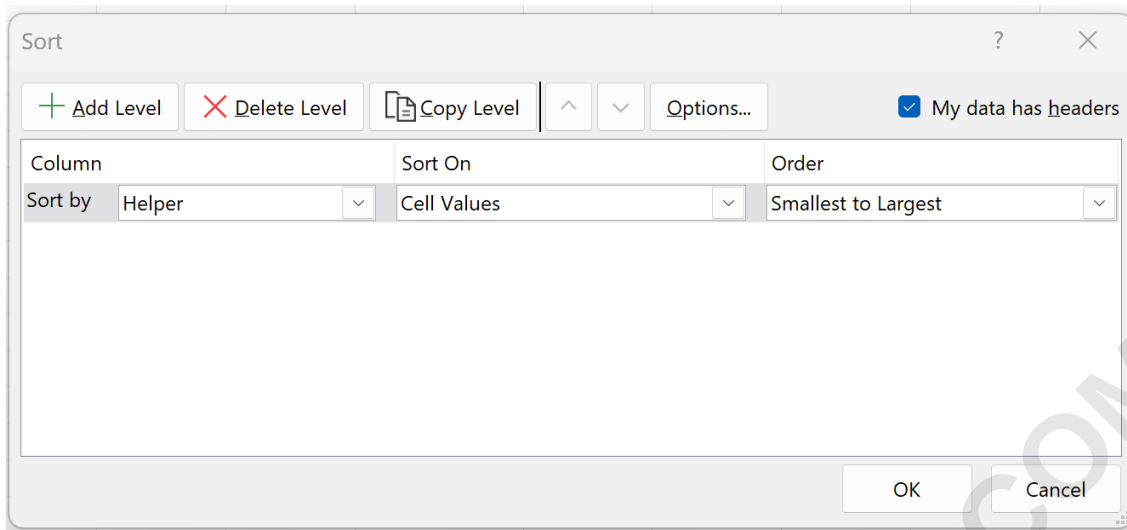
In the new 'Sort' window, we must specify that the sorting operation should be driven by the values in the **Helper** column, not the IP Address column itself. This ensures the 12-digit numeric key dictates the final arrangement.

Set the criteria as follows:

Choose **Helper** from the **Sort by** dropdown menu. (Note: Excel recognizes the header row we included in the selection).

Ensure **Sort On** is set to **Values**.

Select the desired order, typically **Smallest to Largest**, as this corresponds to ascending IP address order.



Once you click **OK**, the sort operation is executed. Excel rearranges the rows based on the calculated numeric values in the Helper column (5/5). The IP addresses in Column A, which are linked directly to the helper values, are moved accordingly, resulting in a dataset that is mathematically and hierarchically correct:

|    | A                 | B             | C | D |
|----|-------------------|---------------|---|---|
| 1  | <b>IP Address</b> | <b>Helper</b> |   |   |
| 2  | 122.45.65.90      | 1.22E+11      |   |   |
| 3  | 122.108.99.155    | 1.22E+11      |   |   |
| 4  | 122.230.33.192    | 1.22E+11      |   |   |
| 5  | 138.192.34.93     | 1.38E+11      |   |   |
| 6  | 184.234.223.94    | 1.84E+11      |   |   |
| 7  | 200.89.43.80      | 2E+11         |   |   |
| 8  | 213.45.67.1       | 2.13E+11      |   |   |
| 9  |                   |               |   |   |
| 10 |                   |               |   |   |
| 11 |                   |               |   |   |
| 12 |                   |               |   |   |
| 13 |                   |               |   |   |
| 14 |                   |               |   |   |
| 15 |                   |               |   |   |

We can clearly observe that the IP address **122.45.65.90** is now correctly positioned as the first element in the list, followed by 122.45.65.200, and so forth, confirming the successful use of zero padding and the NUMBERVALUE (2/5) function to force a numerical sort.

## Conclusion and Final Steps

Sorting non-standard numerical formats like IP addresses in Excel demands moving beyond the default textual sorting capabilities. By leveraging powerful text manipulation functions--specifically TEXTBEFORE (2/5), TEXTAFTER (2/5), and NUMBERVALUE (3/5)--and employing the technique of zero padding, we successfully transformed a text string into a reliable numerical key. This robust methodology ensures that even complex data hierarchies are accurately organized.

Once the rows have been correctly sorted using the custom NUMBERVALUE (4/5) generated key, the purpose of the **Helper** column is fulfilled. Since the rows themselves are now physically rearranged, the helper values are no longer needed for subsequent analysis. Users can safely delete Column B, returning the dataset to a clean format containing only the correctly sorted IP addresses.

This approach provides a scalable and dependable method for handling IP address sorting, a frequent requirement in network administration and data auditing tasks.

### Summary of Key Functions Used:

TEXTBEFORE (3/5): Extracts text before a specified delimiter (the period).

TEXTAFTER (3/5): Extracts text after a specified delimiter, useful for isolating later octets.

TEXT: Crucial for applying the '000' format mask, ensuring zero padding.

& (Ampersand): Used to concatenate the four padded octets into a single 12-character string.

NUMBERVALUE (5/5): Converts the concatenated string into a large integer that Excel can numerically sort.

**Note:** Feel free to delete the **Helper** column once you've sorted the rows now that you no longer need that column.