How to Generate Unique IDs in Google Sheets

Authored by

stats writer

January 15, 2026

Generating unique identifiers (UIs) for data entries is a fundamental requirement in data management, crucial for tracking records, establishing relationships between datasets, and ensuring data integrity. In the dynamic environment of Google Sheets, users often need robust methods to assign these UIs, whether they are based on specific groupings or simply sequential row numbering.

When working with large or complex spreadsheets, the need to assign unique identifiers to rows or categories frequently arises. This process allows for simplified data querying and prevents potential ambiguities when dealing with duplicate entry names. Fortunately, Google Sheets offers several powerful built-in functions that make this task manageable and highly customizable.

Overview of Unique Identifier Generation Methods

There are several effective strategies for creating unique identifiers, depending on whether you need a purely random ID, an ID based on row position, or an ID grouped by existing category values. Understanding these core methods is key to choosing the most appropriate solution for your specific dataset requirements.

One primary method involves using the RAND function. This function generates a random decimal number between 0 and 1. While a random number alone might not guarantee absolute uniqueness, combining it with other functions--such as CONCATENATE (or the ampersand operator &) or the TEXT function to format timestamps--can create a highly unique string. For instance, concatenating a random number with a precise timestamp often results in a nearly collision-free identifier suitable for logging purposes.

Alternatively, if your goal is simply to ensure that a list of values contains no duplicates, you can utilize the UNIQUE function. This formula processes a range of values and returns only the distinct items, effectively creating a list of unique identifiers derived directly from your source data. For structured sequencing, the ROW function is invaluable. By assigning a unique number based on its position in the sheet, the row number can be combined with prefixes or other static information to generate clear, sequential identifiers for each entry.

Method 1: Generating Grouped Unique IDs with Advanced Logic

A common requirement is assigning a consistent unique identifier to all instances of a specific category within a column. For example, if you have multiple entries for "Lakers," you want them all to share ID 2, while "Hawks" might all share ID 3. This approach requires a complex logical formula that checks if the current category name has already been assigned an ID higher up in the sheet. If it has, the formula retrieves that existing ID; if it hasn't, it assigns the next sequential ID available.

This method leverages several powerful lookup and conditional functions, specifically IF function,

MATCH function, MAX, and VLOOKUP function, working in tandem to maintain ID consistency across duplicate entries. This provides a clean, self-contained solution within the spreadsheet, eliminating the need for external scripts or manual assignments.

Example: Generate Unique Identifiers in Google Sheets

Setting Up the Data and Initial Value

To illustrate the generation of grouped unique identifiers, let us consider a dataset containing a list of basketball team names in column A. Notice that many of these teams are listed multiple times, which necessitates a system to assign the same ID to all matching team names.

	A	B	C
1	**Team**		
2	Mavs		
3	Lakers		
4	Mavs		
5	Hawks		
6	Hawks		
7	Warriors		
8	Mavs		
9	Lakers		
10	Nets		
11	Spurs		
12	Lakers		
13			
14			
15			
16			
17			

Our objective is to create a singular, consistent unique identifier value for each distinct team name. To initiate this sequence, we must manually define the starting point. We select the first entry in our ID column (B2, assuming data starts in A2) and assign it a static starting value. For this demonstration, we will use the value **1** for the first team listed in cell A2.

	A	B	C	D
1	**Team**	**Unique ID**		
2	Mavs	1		
3	Lakers			
4	Mavs			
5	Hawks			
6	Hawks			
7	Warriors			
8	Mavs			
9	Lakers			
10	Nets			
11	Spurs			
12	Lakers			
13				
14				
15				
16				

Implementing the Advanced Grouping Formula

The complexity of grouping identifiers requires a logical formula that dynamically checks history. We must determine if the team name in the current row has appeared previously in the list. If it has, we retrieve the ID that was already assigned to it. If it is a completely new team name, we assign it the next highest sequential identifier available.

We input the core formula into cell **B3** (the first cell that requires calculation based on previous rows). This formula performs a two-part check:

=IF(ISNA(MATCH(A3,A2:$A$2,0)),MAX(B2:$B$2)+1,VLOOKUP(A3,A2:$B$2,2,FALSE))

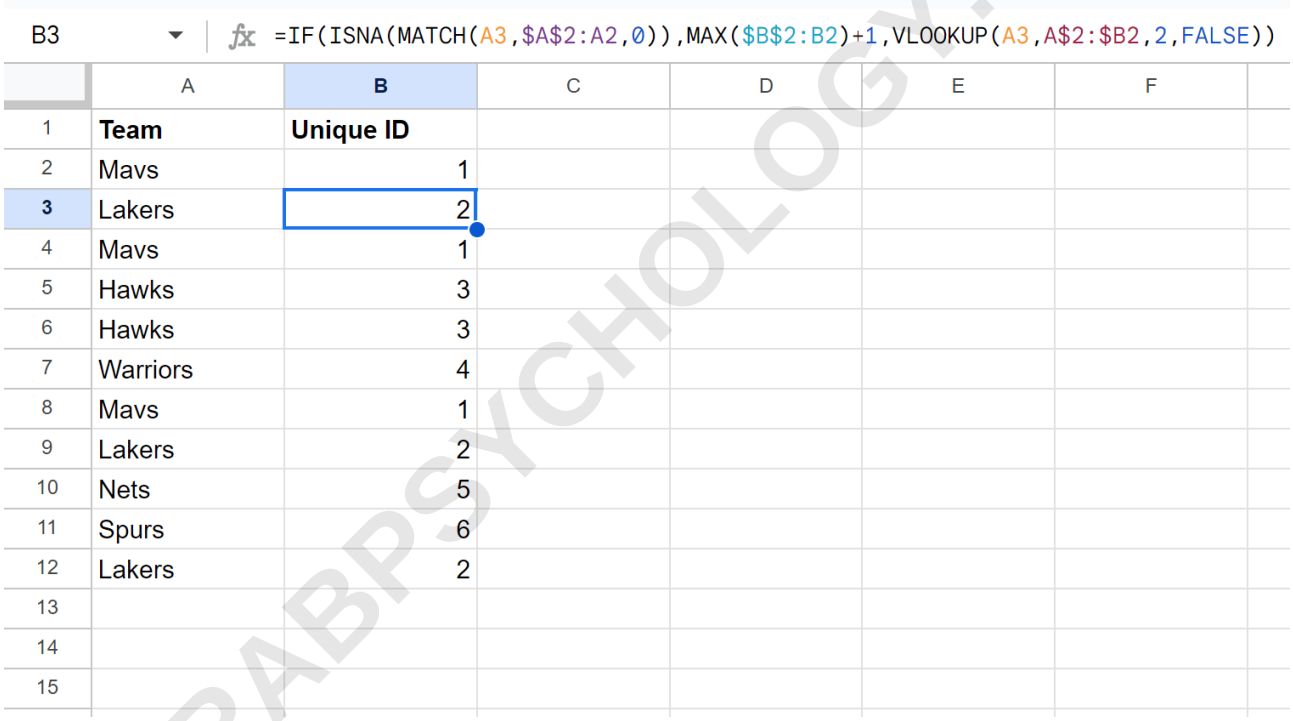
Let's break down the logic of this powerful expression. The MATCH function attempts to locate the value in cell A3 within the range A2:$A$2. Notice that the range expands as the formula is dragged down (A2:$A$3, A2:$A$4, etc.), searching all preceding rows. If the team name is found, MATCH function returns its relative position. If it is not found (meaning it's a new, unique team), MATCH function returns the error value #N/A.

The IF function uses ISNA (Is Not Available) to test the result of the MATCH function. If MATCH function returns #N/A (meaning ISNA is TRUE), the team is new. In this case, the formula executes

the TRUE condition: **MAX(B2:$B$2)+1**. This takes the highest existing ID number in column B above the current row and increments it by 1, assigning a new unique ID.

Conversely, if the team name is found (meaning ISNA is FALSE), the formula executes the FALSE condition: **VLOOKUP(A3, A2:$B$2, 2, FALSE)**. The VLOOKUP function then searches the historical range (A2:$B$2) for the team name in A3 and retrieves the corresponding ID from the second column (Column B), thereby reusing the previously assigned ID for that team.

Once the formula is correctly entered into B3, we can click and drag this formula down to each remaining cell in column B. The relative referencing ensures that the lookup range expands correctly to include all preceding data, while the absolute references ($A$2, $B$2) lock the starting row.



	A	B	C	D	E	F
1	**Team**	**Unique ID**				
2	Mavs	1				
3	Lakers	2				
4	Mavs	1				
5	Hawks	3				
6	Hawks	3				
7	Warriors	4				
8	Mavs	1				
9	Lakers	2				
10	Nets	5				
11	Spurs	6				
12	Lakers	2				
13						
14						
15						

Analyzing the Resulting Grouped IDs

Upon propagation, column B successfully contains a consistent, grouped unique ID value for each team name. This structure is ideal for analyses where grouping metrics by category identifier is necessary, rather than tracking individual rows.

We can clearly observe the grouping logic applied across the dataset:

Each team with the name "Mavs" has an ID value of **1**, reflecting the initial assignment.
Each team with the name "Lakers" has an ID value of **2**, as it was the next unique team

encountered.
Each team with the name "Hawks" has an ID value of **3**, following the sequential assignment pattern.

And so forth, ensuring that every occurrence of a particular team name shares the exact same unique numerical designation across the spreadsheet.

Method 2: Creating Truly Row-Specific Unique Identifiers

While the grouped ID method is useful for categorization, sometimes you require a unique identifier for every single row, even if the primary data column (e.g., team name) contains duplicates. For instance, if you track individual transactions or events, each entry must have its own distinct and non-repeating identifier. This can be achieved by combining the primary data with a dynamically generated sequence number using the COUNTIF function.

This approach guarantees that even if the team name repeats, the resulting combined identifier will be unique. For example, the first instance of "Mavs" might be "Mavs-1," the second "Mavs-2," and so on. This concatenation creates a hybrid identifier that is both descriptive (containing the team name) and truly unique identifier (containing the sequence number).

Applying the COUNTIF Sequence Generator

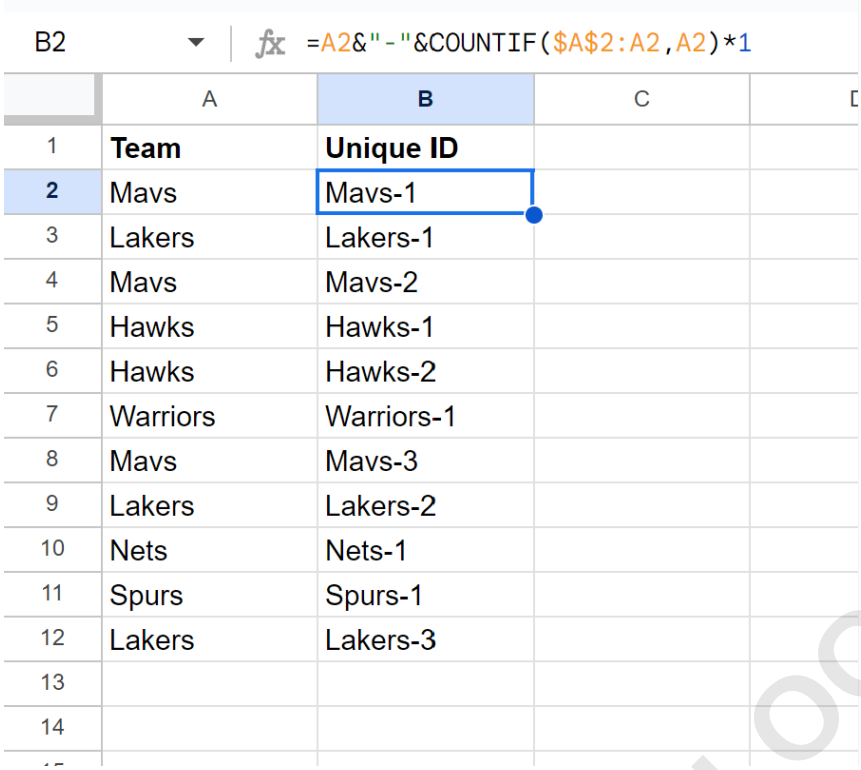
The formula for generating a row-specific, descriptive unique ID relies on the power of relative and absolute referencing within the COUNTIF function. We use the cell containing the team name, append a hyphen separator, and then append the count of how many times that team name has appeared up to the current row.

The formula to be entered into cell B2 (or the first data row) is:

=A2&"-"&COUNTIF($A$2:A2,A2)*1

In this formula, the first part, **A2&"-"**, simply takes the team name and attaches a hyphen. The critical component is the COUNTIF function: **COUNTIF($A$2:A2, A2)**. The range $A$2:A2 starts with an absolute reference ($A$2) and ends with a relative reference (A2). As this formula is dragged down, the range expands (e.g., $A$2:A3, $A$2:A4), counting how many times the criteria (the value in the current row, A2) has appeared in the range from the start ($A$2) up to the current row.

This progressive counting ensures a sequential suffix (1, 2, 3...) for every repetition of the team name. You can then click and drag this formula down to each remaining cell in column B:



	A	B	C
1	**Team**	**Unique ID**	
2	Mavs	Mavs-1	
3	Lakers	Lakers-1	
4	Mavs	Mavs-2	
5	Hawks	Hawks-1	
6	Hawks	Hawks-2	
7	Warriors	Warriors-1	
8	Mavs	Mavs-3	
9	Lakers	Lakers-2	
10	Nets	Nets-1	
11	Spurs	Spurs-1	
12	Lakers	Lakers-3	
13			
14			

As demonstrated in the resulting table, Column B now contains a distinct unique ID for each row. This methodology is highly scalable and ensures non-repeating identifiers for every record in your Google Sheets dataset.

Alternative Strategies and Data Preservation

While the grouping and sequencing methods discussed are robust, other scenarios may call for alternatives, particularly those leveraging randomization or temporal data to achieve maximum uniqueness.

For high-volume data entry where the possibility of duplication must be minimized, combining the RAND function with a timestamp (using the NOW function or similar) provides excellent entropy. By formatting the timestamp to include milliseconds and concatenating it with a randomized component, the chance of generating identical identifiers simultaneously becomes negligible. This approach is often favored in technical logging or database key generation.

The implementation of unique identifiers is paramount for maintaining data integrity. When identifiers are generated using automated formulas, they inherently update if the source data changes. However, if the generated IDs are intended to be permanent, it is essential to convert the formula results to static values.

To ensure permanence, users should select the entire column containing the generated IDs, copy

the cells, and then use the "Paste Special > Paste values only" option. This action replaces the dynamic formulas (like COUNTIF function or VLOOKUP function) with the final calculated numerical or text values. Once converted to static data, the unique identifiers are fixed and will not change, regardless of subsequent modifications to the sorting or content of the source columns.

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