Learning to Group Times into Unequal Intervals Using Excel

Authored by

Mohammed looti

November 12, 2025

Understanding the Need for Unequal Time Bucketing

Data analysis frequently requires segmenting continuous time-series information--such as transaction logs, shift schedules, or operational timestamps--into discrete, manageable categories. This essential practice is commonly known as data binning or creating time buckets. While native grouping features in applications like Excel efficiently handle equal intervals (e.g., grouping data every 60 minutes or 4 hours), real-world business intelligence demands greater flexibility. Operational requirements frequently necessitate grouping based on specific, **unequal intervals** that directly correspond to critical business phases or defined activity periods.

Consider a common scenario where a business defines its crucial peak operational window as 10:00 AM to 1:00 PM, while the pre-peak hours span a much longer duration, perhaps 12:00 AM to 10:00 AM. Standard, equal-interval grouping methods are incapable of effectively isolating and analyzing metrics within these precise, business-defined periods. To achieve this level of analytical precision, analysts must implement sophisticated logical functions that sequentially evaluate time data against custom, non-uniform cut-off points, ensuring the resulting data categories are meaningful for strategic reporting and decision-making.

The powerful technique detailed in this guide utilizes a tailored combination of specialized logical and time conversion functions within Excel. This approach successfully bypasses the limitations inherent in standard data binning tools. It grants the analyst complete control over the boundaries and size of each time category, facilitating highly customized and operationally relevant data segmentation without resorting to external programming languages or overly complex pivot table manipulations.

The Core Formula: Leveraging Nested IF and TIMEVALUE

Successfully categorizing time data into non-uniform intervals hinges on the robust evaluation capabilities of nested IF functions, perfectly harmonized with the precision of the TIMEVALUE function. The primary obstacle when comparing times numerically in Excel is ensuring the comparison is mathematical, not textual. The **TIMEVALUE function** is essential because it converts a time represented as a text string (e.g., "10:00:00 AM") into its underlying Excel serial number--a decimal value ranging from 0 (12:00:00 AM) to less than 1 (just before midnight). This critical conversion allows for accurate numerical comparisons necessary for evaluating boundaries.

Because unequal bucketing requires multiple, conditional checks, the logical structure must be built sequentially using nesting. The resulting time categories must be mutually exclusive and non-overlapping. The formula operates by checking the first boundary: if the time falls within that range, the corresponding bucket is assigned, and the evaluation stops. If the condition is false, the formula systematically proceeds to the next nested condition, checking the subsequent boundary. This sequential evaluation is the mechanism that ensures precise categorization into the correct,

non-overlapping time interval.

The following foundational syntax illustrates how to construct this potent formula in Excel. This specific example creates three distinct, unequal time buckets based on the time value retrieved from cell **B2**:

=IF(B2<=TIMEVALUE("10:00:00 AM"),"12AM-10AM",IF(B2<=TIMEVALUE("1:00:00 PM"),"10AM-1PM","1PM-12AM"))

This construction provides the basic framework for any unequal time bucketing exercise, offering robust scalability regardless of the number of intervals required for the final analysis.

Deconstructing the Sequential Logic

Grasping the sequential flow of the nested IF functions is paramount to correctly defining and respecting the custom time boundaries. The formula evaluates the time in cell **B2** against the specified time checkpoints in a strictly linear, top-down fashion. This linear progression ensures that the moment a condition is satisfied, the remaining conditions are bypassed, which is the core mechanism that establishes the unequal limits of the time buckets.

The outermost `IF` statement initiates the process by defining the first category. It checks whether the time in **B2** is less than or equal to the numerical time serial value equivalent of 10:00:00 AM. If this condition evaluates to **true**, the formula immediately returns the label "12AM-10AM." Crucially, this action isolates all times from the start of the day (12:00:00 AM) up to and including the 10:00 AM cut-off point. This initial step effectively segments the earliest operational period before proceeding to the later intervals.

If the time in **B2** is determined to be greater than 10:00:00 AM, the original formula fails the first test and proceeds directly to the nested `IF` function. This second check evaluates the remaining times against the 1:00:00 PM boundary. Because the first condition already filtered out all times up to 10:00 AM, this nested evaluation inherently establishes a lower boundary of 10:00:01 AM. Consequently, if the time is greater than 10:00 AM but less than or equal to 1:00 PM, the formula returns "10AM-1PM." This structured isolation clearly demonstrates how **nested logic** is used to define precise, sequential, and non-equal time intervals.

Finally, any time entry in cell **B2** that fails to meet both the 10:00 AM and the 1:00 PM conditions must logically occur after 1:00 PM. In this scenario, the final argument of the innermost `IF` function functions as a default, or "catch-all," bucket. It assigns the value "1PM-12AM" to all remaining times between 1:00 PM and midnight. This comprehensive structure guarantees that every entry within the source dataset receives a definitive classification, preventing any unassigned data points.

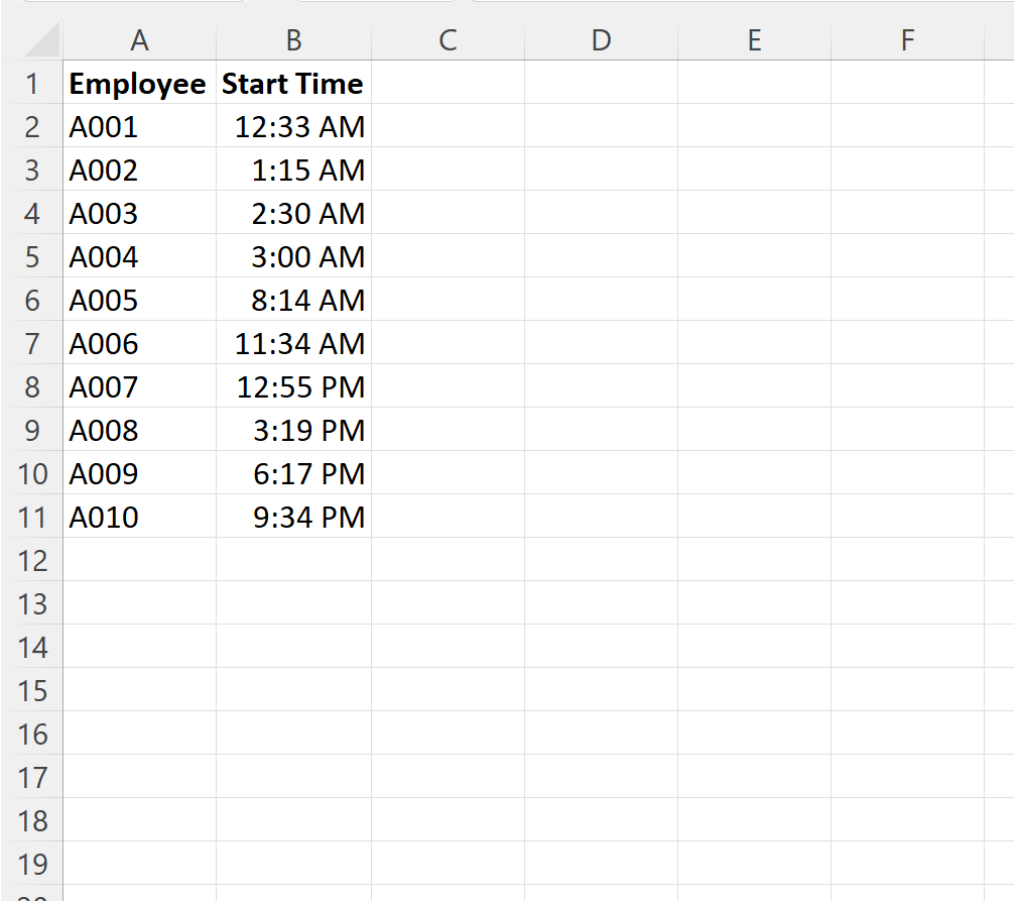
This robust, structured evaluation ensures the following definitive classification criteria are applied:

12AM-10AM: If the time value in cell **B2** is less than or equal to the TIMEVALUE equivalent of 10:00:00 AM.
10AM-1PM: If the time value is greater than 10:00:00 AM but less than or equal to the TIMEVALUE equivalent of 1:00:00 PM.
1PM-12AM: If the time value is greater than 1:00:00 PM (the default classification).

Practical Implementation: Categorizing Employee Shift Data

To clearly demonstrate the practical utility of this formula, let us apply it to a common human resources scenario. Suppose an organization needs to meticulously analyze resource utilization or staffing levels based on the precise start times of employee shifts. Management requires these start times to be categorized into predefined operational windows that specifically deviate from standard, equal hourly groupings.

We begin with a straightforward dataset within Excel. Column A contains the unique Employee ID, and Column B holds the exact time an employee began their shift. It is essential that the data in Column B is stored in a valid time or date-time format, which ensures the underlying serial number comparison functions correctly when evaluated by the formula:



	A	B	C	D	E	F
1	**Employee**	**Start Time**				
2	A001	12:33 AM				
3	A002	1:15 AM				
4	A003	2:30 AM				
5	A004	3:00 AM				
6	A005	8:14 AM				
7	A006	11:34 AM				
8	A007	12:55 PM				
9	A008	3:19 PM				
10	A009	6:17 PM				
11	A010	9:34 PM				
12						
13						
14						
15						
16						
17						
18						
19						

For subsequent reporting and deep analysis, every start time must be classified into one of the following three unequal operational time buckets. These categories are strategically defined by management to align with periods of low, moderate, and high activity:

12AM-10AM (Extended Pre-Peak Hours)
10AM-1PM (Core Peak Activity Window)
1PM-12AM (Post-Peak and Overnight Hours)

The automated application of the nested IF function allows us to assign these shift times rapidly and accurately to their respective organizational periods. This categorization then facilitates crucial downstream analysis, such as calculating headcounts within each defined window or correlating staffing levels with external metrics like customer traffic volume or server load.

Step-by-Step Guide to Applying the Formula

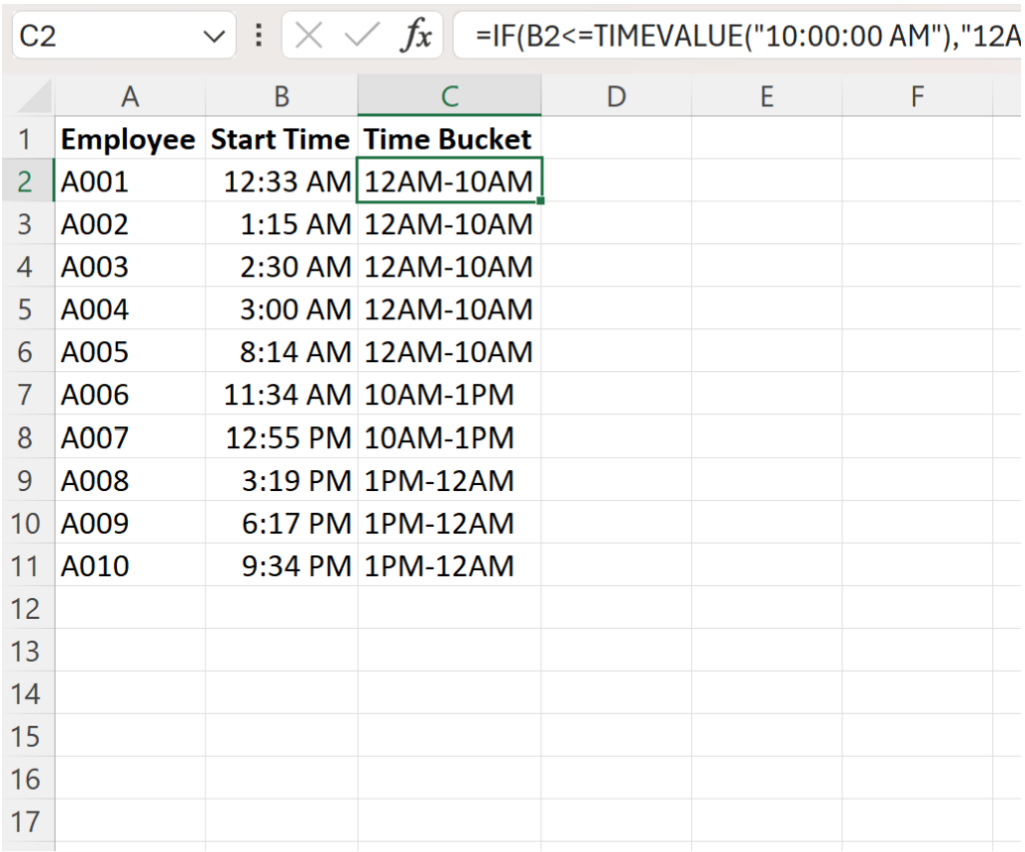
To initiate the custom bucketing process on the employee shift data, we must begin in the first available cell for categorization, typically cell **C2**, which sits immediately adjacent to the first time entry in **B2**. Column C will be designated as the "Time Bucket" classification field. Entering the formula here establishes the crucial relative reference linkage to cell **B2** for the initial calculation.

We accurately input the complete formula into cell **C2**, paying meticulous attention to syntax and ensuring the correct deployment of the TIMEVALUE function for defining the precise cut-off points:

=IF(B2<=TIMEVALUE("10:00:00 AM"),"12AM-10AM",IF(B2<=TIMEVALUE("1:00:00 PM"),"10AM-1PM","1PM-12AM"))

After the formula is calculated for the first row, the efficient autofill feature of Excel is employed. By clicking and dragging the fill handle (the small green square located at the bottom-right corner of cell **C2**) down to the final row of the data range, the formula is automatically replicated across the entire list. Excel dynamically adjusts the relative cell reference (changing **B2** to **B3**, **B4**, and so on) for each subsequent row, allowing the rapid categorization of the entire dataset.

The result is a fully categorized table where Column C shows the specific time bucket for each employee's start time:



	A	B	C	D	E	F
1	**Employee**	**Start Time**	**Time Bucket**			
2	A001	12:33 AM	12AM-10AM			
3	A002	1:15 AM	12AM-10AM			
4	A003	2:30 AM	12AM-10AM			
5	A004	3:00 AM	12AM-10AM			
6	A005	8:14 AM	12AM-10AM			
7	A006	11:34 AM	10AM-1PM			
8	A007	12:55 PM	10AM-1PM			
9	A008	3:19 PM	1PM-12AM			
10	A009	6:17 PM	1PM-12AM			
11	A010	9:34 PM	1PM-12AM			
12						
13						
14						
15						
16						
17						

A review of specific data points confirms the flawless accuracy of the applied logic. For example, Employee A001, whose shift commenced at **12:33 AM**, is correctly assigned to the initial **12AM-10AM** bucket. Employee A006, starting at **11:34 AM**, is classified into the **10AM-1PM** category because their time exceeded the 10:00 AM checkpoint but remained within the 1:00 PM

limit. Lastly, Employee A008, starting at **3:19 PM**, falls into the final default bucket, **1PM-12AM**, as their time surpassed both explicitly defined limits.

Extending Functionality: Creating More Granular Buckets

While the preceding example successfully demonstrated a solution for three time buckets, many demanding analytical scenarios require a much finer level of granularity, potentially requiring four, five, or even more unequal intervals. Fortunately, the fundamental structure of nested IF functions is inherently scalable, enabling analysts to define additional layers of conditions to create more precise checkpoints.

To incorporate an additional bucket--for instance, to isolate a "Late Morning" window spanning from 8:00 AM to 10:00 AM--the original formula requires insertion of a new `IF` condition into the existing structure. Each subsequent bucket necessitates embedding a new `IF` statement into the "value_if_false" argument of the function immediately preceding it, thereby constructing a longer, sequential chain of checks. Although older versions of Excel imposed strict limitations on nesting depth, modern versions handle deep nesting capably, though maintaining formula readability can quickly become challenging.

For complex scenarios involving five or more unequal buckets, experienced Excel users often seek out alternatives to mitigate the complexity of deeply nested `IF` structures. The powerful `IFS` function (available in Excel 2019 and Microsoft 365) provides an excellent solution, evaluating multiple conditions without demanding repetitive nesting, which significantly enhances clarity and maintainability. Alternatively, the most scalable and robust solution for highly complex bucketing requirements involves creating a separate lookup table containing the lower bound of each time bucket and utilizing the **VLOOKUP** or **XLOOKUP** functions with the approximate match mode (range lookup) enabled.

Summary and Best Practices

The ability to accurately group time data into custom, unequal time buckets using Excel's logical functions represents a fundamental skill set. This technique is crucial for transforming raw, time-stamped data into actionable business intelligence. The combined strength of the sequential evaluation provided by **nested IF functions** and the numerical accuracy guaranteed by the **TIMEVALUE function** ensures categorization is precise and perfectly aligned with unique operational rules.

A vital consideration for successfully implementing and troubleshooting this technique is **data integrity**. The entire formula relies on the source column (e.g., Column B) containing valid time data that is stored internally as a numerical serial number. If the time data is mistakenly stored as pure text, the necessary TIMEVALUE comparison will fail, leading to incorrect classifications.

Analysts should always make it a best practice to verify that the source data is correctly formatted as Time or a Custom Date/Time before attempting to apply this complex logical formula.

Mastering these logical structures empowers analysts to transition beyond simple data aggregation and execute sophisticated data segmentation. Whether the task involves analyzing shift overlaps, monitoring service response times, or tracking system usage during non-standard peak periods, utilizing custom time bucketing ensures that the resulting analysis is both accurate, insightful, and perfectly tailored to the unique analytical demands of the business environment.

Additional Resources

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