Extracting the Hour from Datetime in Google Sheets: A Step-by-Step Tutorial

Authored by

Mohammed looti

November 12, 2025

Mastering the manipulation of date and time information is a fundamental requirement for anyone serious about performing rigorous data analysis within spreadsheet environments. In Google Sheets, the ability to isolate specific temporal components from a full datetime stamp--such as extracting the precise hour--is an indispensable skill. This extraction capability enables users to effectively perform time-based aggregation, filtering, and detailed reporting, allowing for a deep dive into cyclical patterns and peak activity periods regardless of the specific calendar day. The primary and most efficient instrument for achieving this task is the built-in HOUR function, a straightforward yet remarkably powerful formula specifically engineered to parse and return the numerical hourly value from a specified cell containing a valid date and time format. This functionality proves absolutely crucial when analyzing high-volume transactional data, interpreting server logs, or managing complex scheduling information where the moment of occurrence, independent of the day, holds significant analytical and operational value.

The underlying mechanics of utilizing the **HOUR** function are exceptionally simple, demanding only a single argument: the cell reference containing the date and time value that you intend to analyze. For example, if your complete timestamp resides in cell **A2**, the formula structure is clean and yields the immediate output of the hour component. It is critically important for analytical purposes to remember that the output generated by the **HOUR** function is always an integer, ranging definitively from 0 (representing the beginning of the day, 12:00 AM midnight) up to 23 (representing 11:00 PM). This adherence to the strict 24-hour clock format is highly advantageous as it facilitates numerical comparison, sorting, and seamless calculation within your spreadsheet environment without requiring complex text parsing or formatting adjustments.

To grasp this concept practically, consider a typical scenario where cell **A2** holds the timestamp string **1/1/2023 10:15**. Applying the designated formula will immediately and reliably isolate the hourly component, providing the integer 10. This inherent simplicity makes the **HOUR** function an indispensable utility tool during time-series data preparation, empowering analysts to rapidly categorize events by the hour they transpired. Such categorization leads directly to improved insights regarding peak operational times, potential bottlenecks, or periods of high user engagement. The following formula perfectly illustrates the basic, fundamental syntax required to execute this essential time extraction task:

=HOUR(A2)

If, as detailed in the scenario, cell **A2** contains the value **1/1/2023 10:15**, executing this calculation will reliably return the integer **10**. Furthermore, this function is designed to handle a wide variety of standardized date formats that are internally recognized by Google Sheets. This robustness ensures that as long as the content of the referenced cell is correctly interpreted internally as a valid date and time serial number, the hour extraction process will invariably succeed. We will now proceed to thoroughly demonstrate a complete, practical application of this function using a typical

business dataset, highlighting its utility in real-world data cleanup and preparation.

Example 1: Extracting the Hour from Datetime in Google Sheets

To effectively illustrate the substantial utility of the **HOUR** function, let us establish a representative sample dataset. This dataset tracks sales activities, recording the exact date and precise time of each transaction for a hypothetical company. This format of raw data, featuring combined date and time stamps, is an industry standard found in numerous transactional databases, customer relationship management (CRM) systems, and financial logs. Our core analytical objective is to transform this highly detailed timestamp information into a more easily digestible and calculable format by isolating solely the hour component of each transaction. This critical transformation will then permit a focused analysis of peak sales periods across the day, abstracting away the specific calendar date and allowing for true hourly trend identification.

Imagine our spreadsheet is structured with two primary columns: Column A, dedicated to holding the full datetime stamp of the sale event, and Column B, detailing the associated monetary sales amount. Our current task necessitates the creation of a new, third column--which we will designate as Column C--dedicated exclusively to housing the extracted hour value. This data transformation is pivotal for subsequent steps, such as building summary tables or generating charts that powerfully visualize hourly activity trends. By initially examining the raw data, we can readily observe that the timestamps exhibit significant variance, underscoring the necessity of employing a robust, systematic method like the **HOUR** function to standardize the time component for reliable aggregated analysis.

The following illustration clearly depicts the initial, raw structure of our dataset before any functions are applied. It is important to note how Column A is meticulously populated with the combined date and time stamp, which serves as the mandatory input for the subsequent function application. Analyzing this input data structure is always the crucial first step in any complex spreadsheet operation, ensuring that all input references are accurate and that the data is recognized correctly before applying the core extraction logic.

	A	B	C	D
1	**Datetime**	**Sales**		
2	1/1/2023 10:15	22		
3	1/4/2023 11:59	13		
4	1/19/2023 22:40	19		
5	2/4/2023 15:45	40		
6	2/17/2023 5:16	23		
7	3/1/2023 12:34	28		
8	3/5/2023 17:01	24		
9	3/7/2023 18:20	17		
10	4/1/2023 2:19	10		
11	4/5/2023 20:04	19		
12				
13				
14				
15				
16				
17				

Applying the HOUR Function to a Range

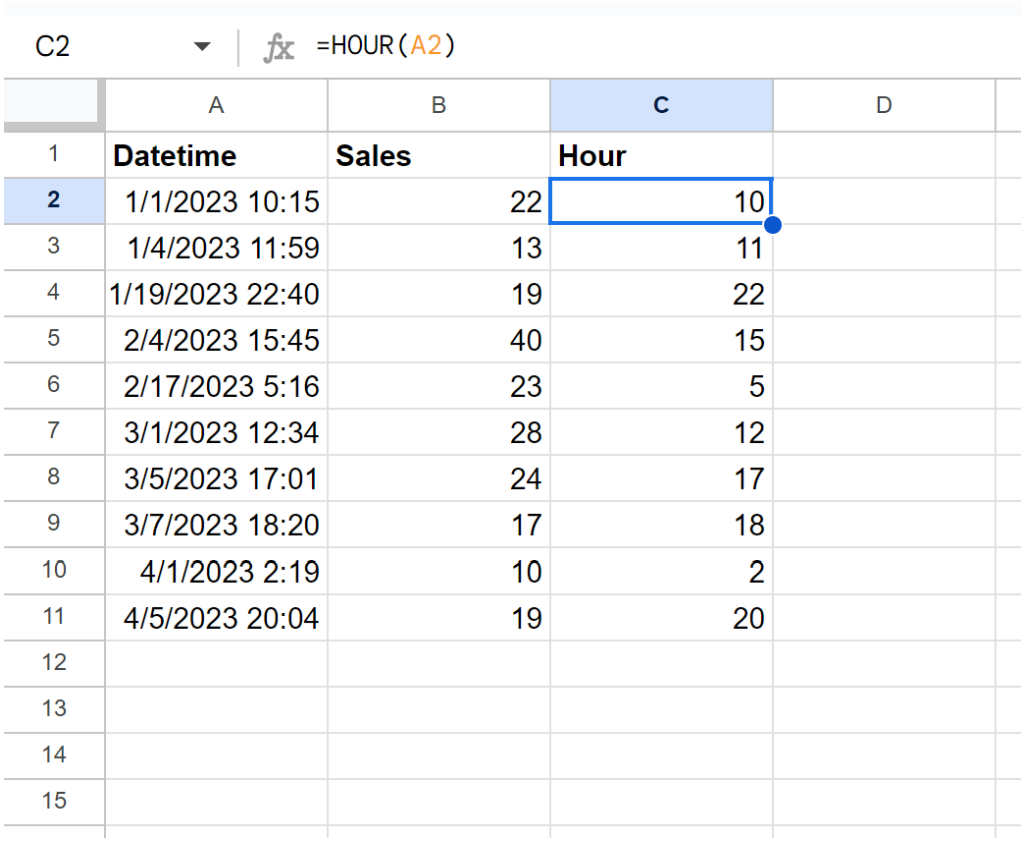
Our immediate objective is to systematically extract the hour from every single entry listed in Column A. To effectively initiate this process, we begin by targeting cell **C2**, as this cell directly corresponds to the first data entry located in **A2**. In cell **C2**, we accurately input the simple, yet highly effective, formula that references the adjacent cell containing the full timestamp. This precise setup ensures that the calculation is executed row-by-row, diligently maintaining the correct correlation between the original timestamp and the resulting hour value across the entire range.

The formula utilized for this range application remains entirely consistent with the basic syntax that was previously introduced. By specifying the cell **A2** as the argument, we issue a clear instruction to Google Sheets: evaluate the content of that specific cell and return the corresponding numerical hour. This foundational step must be executed with precision, paying close attention to accurate cell referencing. This careful attention is especially vital when handling extensive datasets, as any minute error in the initial formula setup can rapidly propagate down the entire column, leading to widespread data inaccuracies.

=HOUR(A2)

Once the formula is correctly entered into **C2**, the real efficiency and power of modern spreadsheets become apparent. Instead of tediously typing or copying the formula for every single row individually, we leverage the highly efficient autofill feature. This is accomplished by clicking and dragging the formula handle (the small square located at the bottom right corner of the selected cell **C2**) down to encompass all the remaining cells in Column C. As this action is performed, Google Sheets intelligently and automatically adjusts the relative cell reference (for example, changing **A2** to **A3**, then **A4**, and so on) for each subsequent row. This method of bulk application is standard, highly recommended practice for efficiently analyzing entire columns of time-series data, saving significant time and reducing manual error potential.

The final outcome of this seamless operation is a newly populated Column C, where every cell now contains a clean, standardized integer representing the hour of the corresponding sale event. This successful transformation is visually confirmed in the following screenshot, clearly demonstrating how the extracted data aligns perfectly with our initial analytical requirements. With the data now prepared, it is immediately ready for advanced statistical functions such as `COUNTIF` or `SUMIF`, which allows for the rapid generation of summary statistics based purely on the hour of the day, isolating true temporal patterns.



	A	B	C	D
1	**Datetime**	**Sales**	**Hour**	
2	1/1/2023 10:15	22	10	
3	1/4/2023 11:59	13	11	
4	1/19/2023 22:40	19	22	
5	2/4/2023 15:45	40	15	
6	2/17/2023 5:16	23	5	
7	3/1/2023 12:34	28	12	
8	3/5/2023 17:01	24	17	
9	3/7/2023 18:20	17	18	
10	4/1/2023 2:19	10	2	
11	4/5/2023 20:04	19	20	
12				
13				
14				
15				

As is vividly confirmed above, Column C now provides only the standardized hourly value derived accurately from the original combined datetime stamps that were situated in Column A. This

extraction process has successfully and robustly isolated the necessary time component required for initiating deeper, more insightful time-based analysis.

Advanced Extraction: Combining Hour and Minute Components

While extracting solely the hour component is often sufficient for high-level analysis, many scenarios require a much finer granularity of detail, necessitating the inclusion of the minute component alongside the hour. For instance, if a data analysis project mandates grouping events into precise 15-minute intervals, simply relying on the hour extraction alone is clearly insufficient. To address this need, Google Sheets provides the MINUTE function, which operates identically to the **HOUR** function but specifically extracts the minute value (an integer ranging between 0 and 59). To present a combined hour and minute output in a standard time display format (HH:MM), we must implement a technique known as string Concatenation.

Concatenation is the critical process of joining multiple distinct text strings or numerical results together to form a single, unified output string. Within Google Sheets, the ampersand symbol (**&**) serves as the dedicated operator for this combining purpose. To accurately construct the desired HH:MM format, we follow a precise sequence: first, we extract the hour using the **HOUR** function; then, we join that numerical result with a colon separator (which must be represented as a text string: **":"**); and finally, we join that intermediate result with the extracted minutes using the MINUTE function. This specific structural layout ensures that the final output is a clearly formatted, human-readable time string, rather than a single ambiguous numerical value.

The comprehensive formula provided below clearly demonstrates this combined approach, effectively utilizing both the **HOUR** and MINUTE functions, linked together seamlessly by the **&** operator and the necessary colon separator. It is important to acknowledge that the resulting output of this specific concatenation formula is inherently a text string, not a numerical time value, which is a crucial distinction when planning for any subsequent mathematical operations. If true numerical time values are required for further calculation, alternative formatting methods, such as employing the dedicated `TEXT` function, would be necessary. However, purely for visual display and readability, this concatenation method is highly effective and widely adopted.

=HOUR(A2)&":"&MINUTE(A2)

When this powerful formula is applied across the dataset, it flawlessly transforms the raw datetime stamp into a standardized time display that includes both hour and minute components. The screenshot below unequivocally confirms the successful application, illustrating Column C now being meticulously populated with the combined hour and minute components, correctly separated by a colon. This enhanced detail provides a significantly more granular snapshot of the transaction time compared to the simple hour extraction alone.

C2 =HOUR(A2)&":"&MINUTE(A2)

	A	B	C	D
1	**Datetime**	**Sales**	**Hour & Minute**	
2	1/1/2023 10:15	22	10:15	
3	1/4/2023 11:59	13	11:59	
4	1/19/2023 22:40	19	22:40	
5	2/4/2023 15:45	40	15:45	
6	2/17/2023 5:16	23	5:16	
7	3/1/2023 12:34	28	12:34	
8	3/5/2023 17:01	24	17:1	
9	3/7/2023 18:20	17	18:20	
10	4/1/2023 2:19	10	2:19	
11	4/5/2023 20:04	19	20:4	
12				
13				
14				
15				

In this final result, Column C accurately displays both the hours and minutes for every corresponding datetime entry found in Column A. This successful implementation is entirely reliant on the correct usage of the MINUTE function to reliably retrieve the minute value and the accurate application of the **&** symbol to properly concatenate the resulting time string into the desired HH:MM format.

Understanding Google Sheets Datetime Serial Numbers

To achieve true mastery over date and time manipulation in spreadsheets, it is fundamentally important to grasp the underlying mechanism by which Google Sheets (and spreadsheet programs universally) store these complex values internally. Dates and times are fundamentally not stored as human-readable text strings; rather, they are maintained as highly precise **serial numbers**. This serial number system is split into two distinct parts: the date component is represented by the integer part of the serial number, which counts the total number of days elapsed since a fixed epoch date (which, in Sheets, is typically December 30, 1899).

Conversely, the crucial time component--which encompasses the hour, minute, and second--is stored entirely as the fractional, or decimal, part of that exact same serial number. For a clear example, a numerical value of 1.5 would precisely represent January 1, 1900, at 12:00 PM (noon). Here, the integer 1 signifies the date, and the decimal 0.5 accurately represents half a day, or 12 hours. The HOUR function essentially performs the mathematical operation of taking this fractional

part of the serial number and calculating precisely which of the 24 intervals it falls into, consequently returning the corresponding integer value (0 through 23). This sophisticated underlying mechanism is why these extraction functions are so inherently robust; they operate directly on the numeric representation rather than attempting to parse complex, variable text formats, making them highly resilient to minor formatting inconsistencies.

A deep understanding of this serial number system is absolutely key to effectively troubleshooting any date and time issues that may arise. If, for instance, the **HOUR** function returns an unexpected error or an incorrect value, it typically signals that the cell input is not being recognized internally as a valid serial date number, often because the format is purely text-based or ambiguous. A simple and effective way to test the validity of the input is to quickly change the cell format from Date/Time to Number. If the cell then accurately displays a large integer followed by a decimal, the value is correctly stored as a datetime serial number, ensuring that the function will execute successfully.

Related Time Extraction Functions and Documentation

The HOUR function is an integral component of a comprehensive suite of time-related functions that collectively allow for the thorough dissection of date and time data. Utilizing these related functions in conjunction with one another allows analysts to construct remarkably powerful, multidimensional time-based aggregations. Beyond the foundational **HOUR** and **MINUTE** functions, other essential tools in this suite include **SECOND**, **DAY**, **MONTH**, and **YEAR**. Crucially, each of these functions accepts a valid date/time serial number as its required argument and returns the corresponding integer value for its specific component.

For instance, if your analytical requirement is to track sales activities not only by the hour but also correlated with the day of the week, you can seamlessly integrate the **WEEKDAY** function alongside the **HOUR** function. This combined analytical approach facilitates truly multidimensional analysis, allowing you to accurately determine if certain specific hours on particular days (such as late Friday evenings or early Monday mornings) exhibit statistically higher transaction volumes or activity rates. It is therefore highly recommended that all users familiarize themselves extensively with the official documentation for these functions to maximize their utility and ensure correct implementation within complex reporting structures.

For users who require highly detailed technical specifications, advanced implementation examples, and a deeper understanding of potential limitations, the complete documentation for the **HOUR** function in Google Sheets, along with all related date and time functions, is readily accessible through the official Google support pages. Consulting these authoritative resources ensures that all formulas are implemented strictly according to current best practices and helps proactively prevent common errors related to function arguments or incorrect input formatting.

Additional Resources for Spreadsheet Mastery

Beyond the crucial skill of extracting individual temporal components, true spreadsheet mastery involves a holistic understanding of how to efficiently handle, accurately calculate, and appropriately format date and time data to meet diverse analytical requirements. Utilizing the core extraction functions discussed throughout this guide is foundational for advancing to more complex data manipulation tasks, such as precisely calculating time differences, accurately determining elapsed time, or scheduling conditional tasks based on specific time criteria.

The following tutorials explain how to perform other common and closely related tasks in Google Sheets, building directly upon the essential skill of isolating time components:

Tutorial on calculating the exact time duration between two distinct timestamps.

A comprehensive guide to formatting date and time outputs using the `TEXT` function for achieving custom display formats.

Instructions on how to use conditional formatting based dynamically on the extracted hour (e.g., automatically highlighting data that falls within identified peak hours).

Effective methods for using the powerful `QUERY` function to group and summarize data based on the extracted hour component.

Developing robust proficiency in these specialized functions--particularly the foundational **HOUR** function--is an absolutely critical step toward becoming a highly efficient and effective data analyst capable of confidently deriving meaningful, actionable insights from complex time-stamped datasets.