How to Easily Filter Google Sheets Query by Date Range

Authored by

stats writer

December 3, 2025

Filtering data efficiently is critical for analysis, and when working with large datasets in Google Sheets, the built-in QUERY function stands out as an exceptionally powerful tool. This function allows users to manipulate, summarize, and filter data using a syntax similar to standard SQL. When dealing with time-series or transactional data, the ability to accurately filter by a specific date range becomes paramount for conducting timely analysis and generating relevant reports.

Successfully filtering a Google Sheets Query by date range is not merely about selecting a column; it requires precise handling of date formats within the query string. The core mechanism involves utilizing the WHERE clause, combined with specific Google Sheets functions like `TEXT` and `DATEVALUE`, to ensure that the date values are interpreted correctly by the underlying query language parser. This technique ensures that your results are meticulously confined to the desired timeframe. Furthermore, enhancing the query with additional clauses, such as the SORT BY clause, can further refine the output by presenting the filtered data in chronological order, optimizing clarity and readability.

Understanding the Google Sheets QUERY Function

The QUERY function is often described as the most sophisticated and adaptable function in the Google Sheets arsenal. It operates by processing data based on the Google Visualization API Query Language, which mimics SQL syntax. A standard query takes two primary arguments: the range of data to analyze (e.g., A1:Z100) and the query string itself (the selection and filtering criteria). Utilizing this function effectively allows you to perform complex database operations--like pivots, groupings, and sophisticated filtering--directly within a spreadsheet environment without resorting to scripting.

To perform any filtering operation, we rely on the WHERE clause. This clause is fundamental for setting conditions that each row must meet to be included in the final result set. When dealing with numeric or text data, the conditions are relatively straightforward (e.g., **WHERE B > 100** or **WHERE C contains 'Product X'**). However, filtering based on date fields introduces complexities because the query engine requires dates to be presented in a very specific format, often contrasting with how dates are usually displayed within the sheet itself.

Mastering the date filter is essential for analysts who regularly handle time-sensitive metrics. Whether you need to compare sales performance quarter-over-quarter, track project milestones between two specific dates, or simply extract all records created after a certain cutoff, the precision offered by the **QUERY** function's date handling mechanism is unmatched. It provides a dynamic solution far superior to simple static filters, especially when the criteria (the start or end date) might frequently change based on other cell inputs.

The Importance of Date Formatting in Queries

When defining dates within the Google Visualization API Query Language, it is absolutely mandatory to use the date literal format: **date 'YYYY-MM-DD'**. The query language strictly expects this structure, which is derived from the internationally recognized ISO 8601 format. If a date is passed in any other format (e.g., MM/DD/YYYY or DD-MMM-YY), the query parser will fail to recognize it as a valid date object, leading to incorrect or null results.

Since most users input dates into Google Sheets using standard regional formats (like 1/1/2020), we must employ a conversion mechanism to transform these user-friendly dates into the required **YYYY-MM-DD** string format that the **QUERY** function demands. This conversion process is typically handled by nesting two critical Google Sheets functions: the DATEVALUE function and the TEXT function.

The core challenge in date filtering lies in concatenating the query string. Because the date itself is a variable that needs to be calculated and formatted, it must be broken out of the main string literal, converted, formatted, and then reinserted using the ampersand (**&**) operator. This meticulous process ensures that the query receives a string that looks exactly like **select * where A > date '2020-01-01'**, thereby successfully interpreting the date as a comparison value.

Setting Up the Initial Query and Date Conversion

Let us examine the foundational structure required to inject a dynamic date into the query string. We begin by using the DATEVALUE function. This function takes a date string (like "1/1/2020") and converts it into its corresponding serial number, which is how Google Sheets internally stores dates. While the serial number is accurate, it is not the format required by the query language.

Next, we utilize the TEXT function. The **TEXT** function is designed to take a numeric value (in this case, the date serial number returned by **DATEVALUE**) and format it into a specified text string. We specify the format pattern **"yyyy-mm-dd"**. This step is crucial as it creates the necessary ISO 8601 format date string required by the WHERE clause.

Finally, we concatenate all these pieces together. The formula below demonstrates how to select all data from the range **A1:C9** where the date in column A is greater than January 1, 2020. Note how the fixed string components (**"select * where A > date '"** and **"'"**) sandwich the dynamic date conversion logic, ensuring the entire query string is valid.

The standard formula structure for filtering a query by a date greater than a specified value is:

=QUERY(A1:C9,"select * where A > date '"&TEXT(DATEVALUE("1/1/2020"),"yyyy-mm-dd")&"'")

This formula specifically returns all rows within the defined range **A1:C9** where the corresponding date entry in column A is chronologically after the date **1/1/2020**.

The following examples illustrate the versatility of this approach using a shared sample dataset:

	A	B	C	D	E
1	**Date**	**Product**	**Revenue**		
2	1/1/2020	A	10		
3	1/3/2020	A	6		
4	1/3/2020	B	8		
5	1/2/2020	C	14		
6	1/5/2020	A	10		
7	1/9/2020	B	19		
8	1/23/2020	B	22		
9	1/19/2020	C	14		
10	1/14/2020	A	18		
11	1/9/2020	B	8		
12	1/19/2020	C	4		
13	1/22/2020	C	7		
14	1/25/2020	A	7		
15	1/4/2020	B	11		
16	1/3/2020	B	13		
17	1/13/2020	C	8		
18					
19					
20					

Example 1: Filtering Records Before a Specific Date

A common requirement is to isolate historical data, meaning records that occurred before a specified cut-off date. This operation is easily achieved by modifying the relational operator within the WHERE clause from **>** (greater than) to **<** (less than). This reversal instructs the QUERY function to only include rows where the date value is strictly earlier than the date provided in the comparison string.

For instance, if we are analyzing data up to the end of the first week of January 2020, we would use a formula that specifies a date limit of 1/10/2020. It is crucial to remember that the less-than operator (**<**) is exclusive; it will not include data recorded exactly on 1/10/2020. If you require inclusive filtering, you would use the less-than-or-equal-to operator (**<=**).

We use the following formula structure to filter for rows where the date falls before 1/10/2020.

Notice the change in the comparison operator within the quoted string:

We can use the following formula to filter for rows where the date is before 1/10/2020:

=QUERY(A1:C17,"select * where A < date '"&TEXT(DATEVALUE("1/10/2020"),"yyyy-mm-dd")&"'")

E1 fx =QUERY(A1:C17,"select * where A < date '"&TEXT(DATEVALUE("1/10/2020"),"yyyy-mm-dd")&"'")

	A	B	C	D	E	F	G
1	**Date**	**Product**	**Revenue**		Date	Product	Revenue
2	1/1/2020	A	10		1/1/2020	A	10
3	1/3/2020	A	6		1/3/2020	A	6
4	1/3/2020	B	8		1/3/2020	B	8
5	1/2/2020	C	14		1/2/2020	C	14
6	1/5/2020	A	10		1/5/2020	A	10
7	1/9/2020	B	19		1/9/2020	B	19
8	1/23/2020	B	22		1/9/2020	B	8
9	1/19/2020	C	14		1/4/2020	B	11
10	1/14/2020	A	18		1/3/2020	B	13
11	1/9/2020	B	8				
12	1/19/2020	C	4				
13	1/22/2020	C	7				
14	1/25/2020	A	7				
15	1/4/2020	B	11				
16	1/3/2020	B	13				
17	1/13/2020	C	8				
18							
19							
20							
21							
22							

Upon execution, the results confirm that only rows where the date is chronologically before 1/10/2020 are successfully returned by the query.

Example 2: Filtering Records After a Specific Date

Conversely, if the analysis requires focusing only on recent or forward-looking data--for example, isolating all transactions that occurred after a specific policy change or project start date--we use the greater-than operator (**>**). This operation effectively creates a lower boundary, ensuring that only data points meeting or exceeding that time threshold are included. This is particularly useful for tracking recent trends or post-implementation performance.

As discussed in the previous example, the strict greater-than operator (**>**) excludes the specified date itself. If 1/10/2020 is the date specified, any record dated exactly 1/10/2020 will not be

included. To include the start date in your result set, always use the greater-than-or-equal-to operator (**>=**). Understanding the difference between these two operators is vital for precise data extraction.

To demonstrate, the following formula isolates all records from the dataset that occurred after 1/10/2020. This utilizes the same date conversion logic, ensuring the date string is correctly parsed by the query language:

We can use the following formula to filter for rows where the date is after 1/10/2020:

=QUERY(A1:C17,"select * where A > date '"&TEXT(DATEVALUE("1/10/2020"),"yyyy-mm-dd")&"'")

E1 fx =QUERY(A1:C17,"select * where A > date '"&TEXT(DATEVALUE("1/10/2020"),"yyyy-mm-dd")&"'")

	A	B	C	D	E	F	G
1	**Date**	**Product**	**Revenue**		Date	Product	Revenue
2	1/1/2020	A	10		1/23/2020	B	22
3	1/3/2020	A	6		1/19/2020	C	14
4	1/3/2020	B	8		1/14/2020	A	18
5	1/2/2020	C	14		1/19/2020	C	4
6	1/5/2020	A	10		1/22/2020	C	7
7	1/9/2020	B	19		1/25/2020	A	7
8	1/23/2020	B	22		1/13/2020	C	8
9	1/19/2020	C	14				
10	1/14/2020	A	18				
11	1/9/2020	B	8				
12	1/19/2020	C	4				
13	1/22/2020	C	7				
14	1/25/2020	A	7				
15	1/4/2020	B	11				
16	1/3/2020	B	13				
17	1/13/2020	C	8				
18							
19							
20							
21							

The visual output confirms that only the rows recorded after the specified date of 1/10/2020 are successfully included in the final result set.

Example 3: Filtering Records Within a Date Range

The most powerful application of date filtering is defining a precise range, allowing the extraction of data bounded by both a start date and an end date. This is achieved by combining two conditions

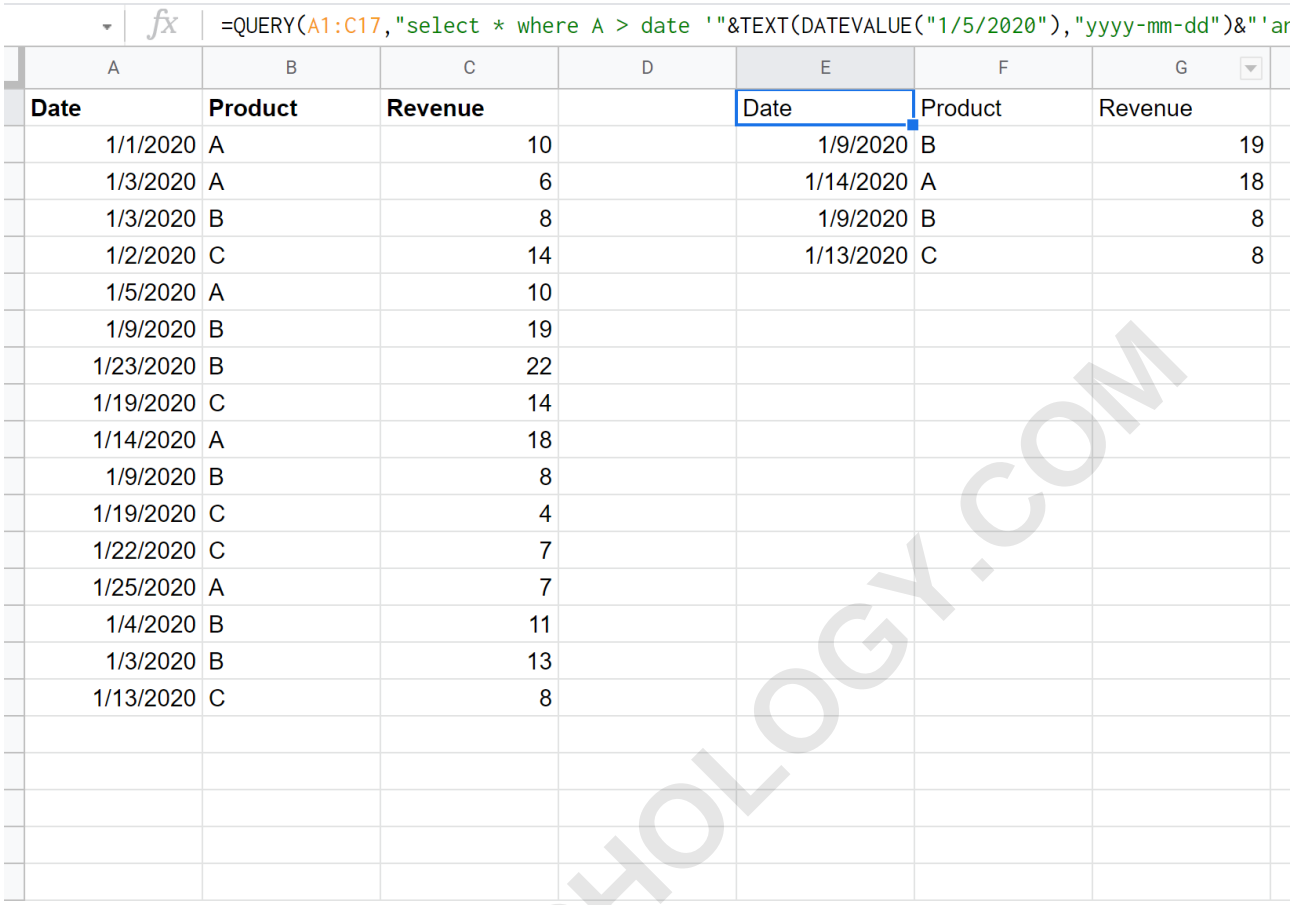
within the WHERE clause, joined by the logical operator **AND**. We must specify that the date column must be greater than the start date AND less than the end date simultaneously.

Each boundary condition (the start and end date) requires its own full date conversion sequence using DATEVALUE and TEXT to ensure both date parameters are correctly formatted as ISO 8601 format strings before being passed to the query engine. It is essential to manage the quotation marks meticulously; each date literal must be wrapped by **date 'YYYY-MM-DD'**, meaning the **DATEVALUE/TEXT** function output is sandwiched between the necessary single quotes and the keyword **date**.

For our example, we aim to retrieve records between 1/5/2020 and 1/15/2020. Notice that we use **>** for the start date (1/5/2020) and **<** for the end date (1/15/2020), making both boundaries exclusive. If the requirement was to include the boundary dates, the operators would be adjusted to **>=** and **<=**. The resulting formula is significantly longer due to the required duplication of the date formatting logic for the second date boundary:

We can use the following comprehensive formula to filter for rows where the date is strictly between 1/5/2020 and 1/15/2020:

=QUERY(A1:C17,"select * where A > date '"&TEXT(DATEVALUE("1/5/2020"),"yyyy-mm-dd")&"'and A < date '"&TEXT(DATEVALUE("1/15/2020"),"yyyy-mm-dd")&"'")



=QUERY(A1:C17,"select * where A > date '"&TEXT(DATEVALUE("1/5/2020"),"yyyy-mm-dd")&"'ar

A	B	C	D	E	F	G
Date	**Product**	**Revenue**		Date	Product	Revenue
1/1/2020	A	10		1/9/2020	B	19
1/3/2020	A	6		1/14/2020	A	18
1/3/2020	B	8		1/9/2020	B	8
1/2/2020	C	14		1/13/2020	C	8
1/5/2020	A	10				
1/9/2020	B	19				
1/23/2020	B	22				
1/19/2020	C	14				
1/14/2020	A	18				
1/9/2020	B	8				
1/19/2020	C	4				
1/22/2020	C	7				
1/25/2020	A	7				
1/4/2020	B	11				
1/3/2020	B	13				
1/13/2020	C	8				

The filtered output confirms that only the rows where the date falls strictly between 1/5/2020 and 1/15/2020 are returned.

Advanced Considerations for Date Range Filtering

While using hardcoded date strings (e.g., **"1/5/2020"**) within the DATEVALUE function is effective for fixed reports, the true power of this technique lies in making the date range dynamic. Instead of referencing a fixed date string, reference a cell containing the desired start or end date (e.g., **A2** or **B2**). If cell B1 contains the start date and cell B2 contains the end date, the formula segment for the start date would become **&TEXT(B1,"yyyy-mm-dd")&**. This methodology makes the query highly adaptable; changing the date in B1 or B2 instantly updates the filtered results.

A significant challenge when working with dates in Google Sheets, especially with the WHERE clause, is handling time components. If your date column (Column A in our examples) actually contains full date-time stamps (e.g., 1/15/2020 10:30:00 AM), a simple date filter might yield unexpected results. When you use **WHERE A > date '2020-01-15'**, it is implicitly comparing against 2020-01-15 00:00:00. To accurately filter the entire day inclusive of time, you often need to adjust the comparison logic or use the **DATETIME** keyword instead of **DATE** if dealing specifically

with time components in the query string.

Another advanced technique involves integrating other filtering criteria alongside the date range. For example, you might want all sales transactions between 1/1/2020 and 1/31/2020 for a specific region. You simply extend the logical conditions in the WHERE clause: **... where A > date '...' AND A < date '...' AND C = 'North Region'**. This ability to layer filters, combined with the robustness of the date conversion process, makes the **QUERY** function an indispensable tool for complex data analysis in Google Sheets.

Further Exploration of Date Operations

The methods described above lay the groundwork for basic range filtering. However, the query language supports various other date-related comparisons and manipulations. Users can leverage functions like **YEAR()**, **MONTH()**, and **DAY()** directly within the WHERE clause to filter based on specific components of the date, regardless of the full date value. For instance, filtering all records that occurred in January of any year might look like **... where month(A) = 0**. (Note: Query language months are zero-indexed, meaning January is 0).

For those needing to group or aggregate data based on time periods, the **GROUP BY** clause can be combined with date components. Grouping by **year(A)** or **quarter(A)** allows for quick summaries of yearly or quarterly metrics, respectively, greatly enhancing the reporting capabilities beyond simple row extraction.

Finally, always ensure that the column referenced in the query (Column A in our examples) is formatted correctly as a date field within Google Sheets. If the column data is stored as plain text, even the most meticulously constructed formula using DATEVALUE and TEXT will fail, as the underlying data type must be recognizable as a numerical date serial for accurate comparison within the **QUERY** engine.

The following tutorials explain how to perform other common operations with dates in Google Sheets: