

Extract Your Data

Version: 2019.2 Applies to: Tableau Desktop

Extracts are saved subsets of data that you can use to improve performance or to take advantage of Tableau functionality not available or supported in your original data. When you create an extract of your data, you can reduce the total amount of data by using filters and configuring other limits. After you create an extract, you can refresh it with data from the original data. When refreshing the data, you have the option to either do a full refresh, which replaces all of the contents in the extract, or you can do an incremental refresh, which only adds rows that are new since the previous refresh.

Extracts are advantageous for several reasons:

- **Supports large data sets:** You can create extracts that contain billions of rows of data.
- **Fast to create:** If you're working with large data sets, creating and working with extracts can be faster than working with the original data.
- **Help improve performance:** When you interact with views that use extract data sources, you generally experience better performance than when interacting with views based on connections to the original data.
- **Support additional functionality:** Extracts allow you to take advantage of Tableau functionality that's not available or supported by the original data, such as the ability to compute Count Distinct.
- **Provide offline access to your data:** Extracts allow you to save and work with the data locally when the original data is not available. For example, when you are traveling.

Latest changes to extracts

Beginning in version 10.5, when you create a new extract it uses the .hyper format. Extracts in the .hyper format take advantage of the improved data engine, which supports faster analytical and query performance for larger data sets.

Similarly, when an extract-related task is performed on a .tde extract using version Tableau Desktop 2019.2, the extract is upgraded to a .hyper extract. After a .tde extract is upgraded to a .hyper extract, it can't be reverted back to .tde extract. For more information, see [Extract Upgrade to .hyper Format \(extracting_upgrade.htm\)](#).

Changes to values and marks in the view

To improve extract efficiency and scalability, values in extracts can be computed differently in version 2019.2 compared to versions 10.4 and earlier. Changes to how the values are computed can affect the way marks in your view are populated. In some rare cases, the changes can cause your view to change shape or become blank. These changes can also apply to multi-connection data sources, data sources that use live connections to filed-based data, data sources that connect to Google Sheets data, cloud-based data sources, extract-only data sources, and WDC data sources.

To get an idea of some of the differences you might see in your view using version 2019.2, see the sections below.

Format of date and date time values

In version 2019.2, extracts are subject to more consistent and stricter rules around how date strings are interpreted through the DATE, DATETIME, and DATEPARSE functions. This affects how dates are parsed, or the date formats and patterns that are allowed for these functions. More specifically, the rules introduced in 2019.2 can be generalized as the following:

1. Dates are evaluated and then parsed by column, not by row.
2. Dates are evaluated and then parsed based on the locale of where the workbook was created, not on locale of the computer where the workbook is opened.

These new rules allow extracts to be more efficient and to produce results that are consistent with commercial databases.

However, because of these rules, particularly in international scenarios where the workbook is created in a locale different from the locale that the workbook is opened in or the server that the workbook is published to, you might see that 1.) date and datetime values change to different date and datetime values or 2.) date and datetime values change to *Null*. When your date and datetime values change to different date and datetime values or become *Null*, it's often an indication that there are issues with the underlying data.

Here are some common reasons why you might see changes to your date and datetime values in your extract data source using version 10.5 and later.

Common causes of changes to date/datetime values	Common causes of null values
<ul style="list-style-type: none">When a function has to parse multiple date formats in a single column. Where the date is ambiguous and can be interpreted in several different ways, the date will be interpreted based on the format Tableau has determined for that column. For some examples, see Date scenario 1 and Date scenario 2 below.When a function has to parse a YYYY-MM-DD (ISO) format. For an example, see Date scenario 3.When a function doesn't have enough information to derive the time, it can interpret a value as "00:00:00.0", using "o" for hour, minute, second, and millisecond.When a function doesn't have enough information to derive the day, it can interpret a value as "1" or "January" for month.When a function parses years, it is interpreted as the following:<ul style="list-style-type: none">Year "07" is interpreted as "2007"Year "17" is interpreted as "2017."Year "30" is interpreted as "2030."Year "69" interpreted as "2069."Year "70" is interpreted as "1970."	<ul style="list-style-type: none">When a function has to parse multiple date formats in a single column. After Tableau determines the date format, all other dates in the column that deviate from the format become null values. For some examples, see Date scenario 1 and Date scenario 2 below.When a function has to parse a YYYY-MM-DD (ISO) format. Values that exceed what is allowed for "YYYY," or "MM," or "DD" cause null values. For an example, see Date scenario 3.When a function has to parse date values that contain trailing characters. For example, time zone and daylight savings suffixes and keywords, such as "midnight" cause null values.When a function has to parse an invalid date or time. For example, 32/3/2012 causes a null value. In another example, 25:01:61 causes a null value.When a function has to parse contradicting inputs. For example, suppose the pattern is 'dd.MM (MMMM) y' and the input string is '1.09 (August) 2017', where both "9" and "August" are months. The result is a null value because the month values are not the same.When a function has to parse contradicting patterns. For example, a pattern that specifies a mix of Gregorian year (y) and ISO week (ww) causes null values.

Date scenario 1

Suppose you have a workbook created in an English locale that uses .tde extract data source. The table below shows a column of string data contained in the extract data source.

10/31/2018
31/10/2018
12/10/2018

Based on the particular English locale, the format of the date column was determined to follow the MDY (month, day, and year) format. The following tables show what Tableau displays based on this locale when the DATE function is used to convert string values into date values.

October 31, 2018
October 31, 2018
December 10, 2018

If the extract is opened in a German locale, you see the following:

31 Oktober 2018
31 Oktober 2018
12 Oktober 2018

However, after the extract is opened in a German locale using version 2019.2, the DMY (day, month, and year) format of the German locale is strictly enforced and causes a *Null* value because one of the values doesn't follow DMY format.

Null
October 31, 2018
October 12, 2018

Date scenario 2

Suppose you have another workbook created in an English locale that uses a .tde extract data source. The table below shows a column of numeric date data contained in the extract data source.

1112018
1212018
1312018
1412018

Based on the particular English locale, the format of the date column was determined to follow the MDY (month, day, and year) format. The following tables show what Tableau displays based on this locale when the DATE function is used to convert the numeric values into date values.

11/1/2018
12/1/2018
Null
Null

Date scenario 3

Suppose you have a workbook that uses a .tde extract data source. The table below shows a column of string data contained in the extract data source.

2018-10-31
2018-31-10
2018-12-10
2018-10-12

Because the date uses the ISO format, the date column always follows the YYYY-MM-DD format. The following tables show what Tableau displays when the DATE function is used to convert string values into date values.

October 10, 2018
Null
December 10, 2018
October 12, 2018

Note: In versions 10.4 (and earlier), ISO format and other date formats could have produced differing results depending on the locale of where the workbook was created. In an English locale for example, both 2018-12-10 and 2018/12/10 could produce December 12, 2018. However, in a German locale 2018-12-10 could produce December 12, 2018 and 2018/12/10 could produce October 12, 2018.

Sort order and case sensitivity

In version 2019.2, extracts have collation support and therefore can more appropriately sort string values that have accents or are cased differently.

For example, suppose you have a table of string values. In terms of sort order, this means that a string value like Égypte is now appropriately listed after Estonie and before Fidji.

About Excel data:

With regard to casing, this means that how Tableau stores values have changed between version 10.4 (and earlier) and version 10.5 (and later). However, the rules for sorting and comparing values haven't. In version 10.4 (and earlier), string values like "House," "HOUSE," and "houSe" are treated the same and stored with one representative value. In version 10.5 (and later), the same string values are considered unique and therefore stored as individual values. For more information, see [Changes to the way values are computed \(examples_excel.htm#semantics\)](#).

Breaking ties in top N queries

When a top N query in your extract produces duplicate values for a specific position in a rank, the position that breaks the tie can be different when using version 2019.2. For example, suppose you create a top 3 filter. Positions 3, 4, and 5 have the same values. When using version 10.4 and earlier, the top filter can return 1, 2, and 3 positions. However, when using version 2019.2, the top filter can return 1, 2, and 5 positions.

Precision of floating-point values

In version 2019.2, extracts are better at taking advantage of the available hardware resources on a computer and therefore able to perform mathematical operations in a highly parallel way. Because of this, real numbers can be aggregated by .hyper extracts in different order. When numbers are aggregated in different order, you might see different values in your view after the decimal point each time the aggregation is computed. This is because floating-point addition and multiplication is not necessarily associative. That is, $(a + b) + c$ is not necessarily the same as $a + (b + c)$. Also, real numbers can be aggregated in different order because floating-point multiplication is not necessarily distributive. That is, $(a \times b) \times c$ is not necessarily the same as $a \times (b \times c)$. This type of floating-point rounding behavior in .hyper extracts resemble that of floating-point rounding behavior in commercial databases.

For example, suppose your workbook contains a slider filter on an aggregated field comprised of floating point values. Because the precision of floating-point values have changed, the filter might now exclude a mark that defines the upper or lower bound of the filter range. The absence of these numbers could cause a blank view. To resolve this issue, move the slider on the filter or remove and add the filter again.

Accuracy of aggregations

In version 2019.2, extracts optimize for large data sets by taking better advantage of the available hardware resources on a computer and therefore able to compute aggregations in a highly parallel way. Because of this, aggregations performed by .hyper extracts can resemble the results from commercial databases more than the results from software that specializes in statistical computations. If you're working with a small data set or need a higher level of accuracy, consider performing aggregations through reference lines, summary card statistics, or table calculation functions like variance, standard deviation, correlation, or covariance.

About the Compute Calculations Now option for extracts

If the **Compute Calculations Now** option was used in a .tde extract using an earlier version of Tableau Desktop, certain calculated fields were materialized and therefore computed in advance and stored in the extract. If you upgrade the extract from a .tde extract to a .hyper extract, the previously materialized calculations in your extract are not included. You must use the **Compute Calculations Now** option again to ensure that materialized calculations are a part of the extract after the extract upgrade. For more information, see [Materialize Calculations in Your Extracts \(extracting_optimize.htm\)](#).

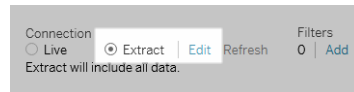
New Extract API

You can use the Extract API 2.0 to create .hyper extracts. For tasks that you previously performed using the Tableau SDK, such as publishing extracts, you can use the Tableau Server REST API or the Tableau Server Client (Python) library. For refresh tasks, you can use the Tableau Server REST API as well. For more information, see [Tableau Extract API \(extracting TDE API.htm\)](#).

Create an extract

Though there are number of places in your Tableau work flow where you can create an extract, the primary method is described below.

1. After you connect to your data and set up the data source on the Data Source page, in the upper-right corner, select **Extract**, and then click the **Edit** link to open the Extract Data dialog box.



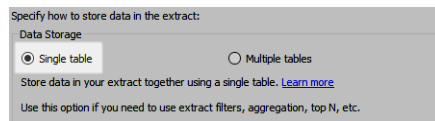
2. (Optional) Configure one or more of the following options to tell Tableau how to store, define filters for, and limit the amount of data in your extract:

- **Decide how the extract data should be stored**

You can choose to have Tableau store the data in your extract using one of two structures (schemas): single table (denormalized schema) or multiple tables (normalized schema). The option you choose depends on what you need.

- **Single table**

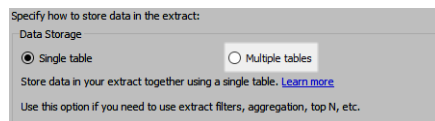
Select **Single table** when you want to limit the amount of data in your extract with additional extract properties like extract filters, aggregation, etc.; or when your data uses pass-through functions (RAWSQL). This is the default structure Tableau uses to store extract data. If you use this option when your extract contains joins, the joins are applied when the extract is created.



- **Multiple tables**

Select **Multiple tables** if your extract is comprised of tables combined with one or more equality joins and meets the [Conditions for using the "Multiple tables" option](#) listed below. If you use this option, joins are performed at query time.

This option can potentially improve performance and help reduce the size of the extract file. For more information about how Tableau recommends you use the "Multiple tables" option, see [Tips for using the "Multiple tables" option](#). In some cases, you can also use this option as a workaround for row-level security. For more information about row-level security using Tableau, see [Restrict Access at the Data Row Level \(publish_userfilters.htm\)](#).



Conditions for using the "Multiple tables" option

To store your extract using the "Multiple tables" option, the data in your extract must meet all of the conditions listed below.

- All joins between tables are equality (=) joins
- Data types of the join columns are identical
- No pass-through functions (RAWSQL) used
- No incremental refresh configured
- No extract filters configured
- No top N or sampling configured

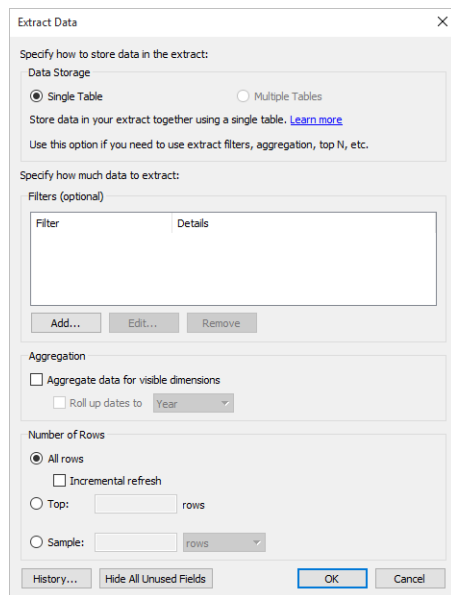
When the extract is stored as "Multiple tables," you cannot append data to it.

Note: Both the "Single table" and "Multiple tables" options only affect how the data in your extract is stored. The options do not affect how tables in your extract are displayed on the Data Source page.

For example, suppose your extract is comprised of three tables. If you directly open the extract (.hyper) file that has been configured to use the default option, "Single table," you see one table listed on the Data Source page. However, if you open the extract using the packaged data source (.tdsx) file or the data source (.tdsx) file with its corresponding extract (.hyper) file, you see all three tables that comprise the extract on the Data Source page.

◦ Determine how much data to extract

Click **Add** to define one or more filters to limit how much data gets extracted based on fields and their values.



The "Extract Data" dialog box is used to configure how data is stored and extracted. It contains several sections:

- Specify how to store data in the extract:**
 - Data Storage:** Two radio buttons are present: "Single Table" (selected) and "Multiple Tables". Below them is a note: "Store data in your extract together using a single table. [Learn more](#)" and "Use this option if you need to use extract filters, aggregation, top N, etc."
- Specify how much data to extract:**
 - Filters (optional):** A table with two columns, "Filter" and "Details", is shown. Below the table are buttons for "Add...", "Edit...", and "Remove".
- Aggregation:**
 - A checkbox labeled "Aggregate data for visible dimensions" is unchecked.
 - Below it, a checkbox labeled "Roll up dates to" is unchecked, followed by a dropdown menu currently showing "Year".
- Number of Rows:**
 - A radio button labeled "All rows" is selected.
 - Below it, an unchecked checkbox labeled "Incremental refresh" is shown.
 - Two other options are available: "Top:" followed by a text input field and the word "rows", and "Sample:" followed by a text input field and a dropdown menu currently showing "rows".

At the bottom of the dialog are buttons for "History...", "Hide All Unused Fields", "OK", and "Cancel".

◦ Aggregate the data in the extract

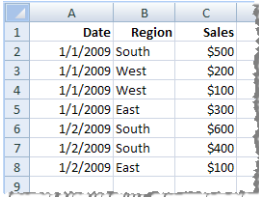
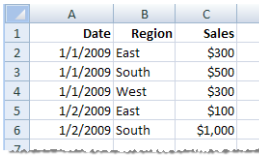
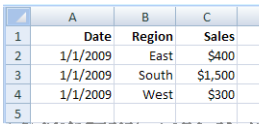
Select **Aggregate data for visible dimensions** to aggregate the measures using their default aggregation. Aggregating the data consolidates rows, can minimize the size of the extract file, and increase performance.

When you choose to aggregate the data, you can also select **Roll up dates** to a specified date level such as Year, Month, etc. The examples below show how the data will be extracted for each aggregation option you can choose.

Aggregation

☐ Aggregate data for visible dimensions

☐ Roll up dates to
 Year

Original data		Each record is shown as a separate row. There are seven rows in your data.
Aggregate data for visible dimensions (no roll up)		Records with the same date and region have been aggregated into a single row. There are five rows in the extract.
Aggregate data for visible dimensions (roll up dates to Month)		Dates have been rolled up to the Month level and records with the same region have been aggregated into a single row. There are three rows in the extract.

◦ **Choose the rows to extract**

Select the number of rows you want to extract.

You can extract **All rows** or the **TopN** rows. Tableau first applies any filters and aggregation and then extracts the number of rows from the filtered and aggregated results. The number of rows options depend on the type of data source you are extracting from.

Notes:

- Not all data sources support sampling. Therefore, you might not see the **Sampling** option in the Extract Data dialog box.
- Any fields that you hide first in the Data Source page or on the sheet tab will be excluded from the extract. Click the **Hide All Unused Fields** button to remove these hidden fields from the extract.

3. When finished, click **OK**.

4. Click the sheet tab. Clicking the sheet tab initiates the creating of the extract.

5. In the subsequent dialog box, select a location to save the extract, give the extract file a name, and then click **Save**.

If the Save dialog box does not display, see the [Troubleshoot extracts](#) section, below.

General tips for working with extracts

Save your workbook to preserve the connection to the extract

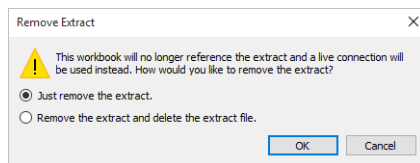
After you create an extract, the workbook begins to use the extract version of your data. However, the connection to the extract version of your data is not preserved until you save the workbook. This means if you close the workbook without saving the workbook first, the workbook will connect to the original data source the next time you open it.

Toggle between sampled data and entire extract

When you're working with a large extract, you might want to create an extract with a sample of the data so you can set up the view while avoiding long queries every time you place a field on a shelf on the sheet tab. You can then toggle between using the extract (with sample data) and using the entire data source by selecting a data source on the **Data** menu and then selecting **Use Extract**.

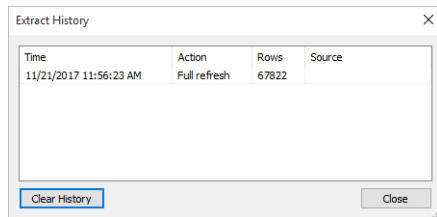
Remove the extract from the workbook

You can remove an extract at anytime by selecting the extract data source on the **Data** menu and then selecting **Extract > Remove**. When you remove an extract, you can choose to **Remove the extract from the workbook only** or **Remove and delete the extract file**. The latter option will delete the extract from your hard drive.



See extract history

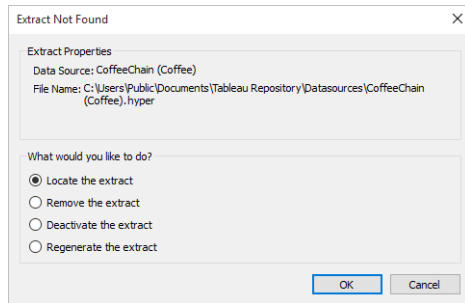
You can see when the extract was last updated and other details by selecting a data source on the **Data** menu and then selecting **Extract > History**.



If you open a workbook that is saved with an extract and Tableau cannot locate the extract, select one of the following options in the Extract Not Found dialog box when prompted:

- **Locate the extract:** Select this option if the extract exists but not in the location where Tableau originally saved it. Click **OK** to open an Open File dialog box where you can specify the new location for the extract file.
- **Remove the extract:** Select this option if you have no further need for the extract. This is equivalent to closing the data source. All open worksheets that reference the data source are deleted.
- **Deactivate the extract:** Use the original data source from which the extract was created, instead of the extract.

- **Regenerate the extract:** Recreates the extract. All filters and other customizations you specified when you originally created the extract are automatically applied.



Tips for using the "Multiple tables" option

Tableau generally recommends that you use the default data storage option, "Single table," when setting up and working with extracts. In many cases, some of the features you need for your extract, like extract filters, are only available to you if you use the "Single table" option.

"Multiple tables" option for extracts that are larger than expected

The "Multiple tables" option should be used sparingly to help with specific situations such as when your data source meets the [Conditions for using the "Multiple tables" option](#) and the size of your extract is larger than expected. To determine if the extract is larger than it should be, the sum of rows in the extract using the "Single table" option must be higher than the sum of rows of all the combined tables before the extract has been created. If you encounter this scenario, try using the "Multiple tables" option instead.

Alternative filtering suggestions when using the "Multiple tables" option

When using the "Multiple tables" option, other options to help reduce the data in your extract, like extract filters, aggregation, Top N and Sampling are disabled. If you need to reduce the data in an extract that uses the "Multiple tables" option, consider filtering the data before it is brought into Tableau Desktop using one of the following suggestions:

- **Connect to your data and define filters using custom SQL:** Instead of connecting to a database table, connect to your data using custom SQL instead. When creating your custom SQL query, make sure that it contains the appropriate level of filtering that you need to reduce the data in your extract. For more information about custom SQL in Tableau Desktop, see [Connect to a Custom SQL Query \(customsql.htm\)](#).
- **Define a view in the database:** If you have write access to your database, consider defining a database view that contains just the data you need for your extract and then connect to the database view from Tableau Desktop.

Row-level security with extracts

If you want to secure extract data at the row level, using the "Multiple tables" option is the recommended way to achieve this scenario. For more information about row-level security in Tableau, see [Restrict Access at the Data Row Level \(publish_userfilters.htm\)](#).

Troubleshoot extracts

- **Creating an extract takes a long time:** Depending on the size of your data set, creating an extract can take a long time. However, after you have extracted the data and saved it to your computer, performance can improve.
- **Extract is not created:** If your data set contains a really large number of columns (e.g., in the thousands), in some cases Tableau might not be able to create the extract. If you encounter problems, consider extracting fewer columns or restructuring the underlying data.
- **Save dialog does not display or extract is not created from a .twbx:** If you follow the above procedure to extract data from a packaged workbook, the Save dialog does not display. When an extract is created from a packaged workbook (.twbx), the extract file is automatically stored in the package of files associated with the packaged workbook. To access the extract file that you created from the packaged workbook, you must unpackage the workbook. For more information, see [Packaged Workbooks \(save_savework_packagedworkbooks.htm\)](#).

Other articles in this section

- [Extract Upgrade to .hyper Format \(extracting_upgrade.htm\)](#)
- [Refresh Extracts \(extracting_refresh.htm\)](#)
- [Add Data to Extracts \(extracting_addfromfile.htm\)](#)
- [Materialize Calculations in Your Extracts \(extracting_optimize.htm\)](#)
- [Update Server Data Sources That Are Using Extracts \(extracting_push.htm\)](#)
- [Tableau Data Extract Command-Line Utility \(extracting_TDE.htm\)](#)
- [Tableau Data Extract API \(extracting_TDE_API.htm\)](#)