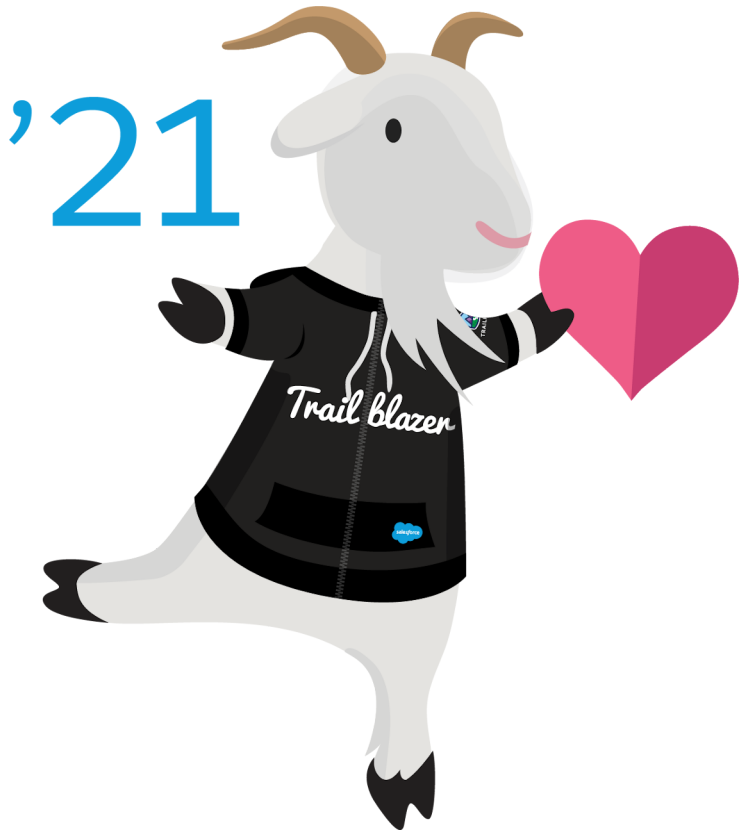
Tableau CRM External Data Format Developer Guide

Salesforce, Summer '21



@salesforcedocs

Last updated: May 20, 2021

CONTENTS

EXTERNAL DATA METADATA OVERVIEW

To upload external data into an Tableau CRM dataset, make sure that you have prepared your data and metadata files.

You can load external data into a dataset by preparing two files.

- A data file, which contains the external data, in comma-separated value (CSV) format
- An optional metadata file, which describes the structure of the data file in JSON format

Note: Providing a metadata file is recommended. Otherwise, every field is treated as text.

The data and metadata files are used to populate a dataset with the external data.

CSV Example

The following CSV example contains data that conforms to the .json metadata file that's described next.

```
Name,Amount,CloseDate
opportunityA,100.99,6/30/2014
opportunityB,99.01,1/31/2012
```

The first row in the CSV file lists the field names for your dataset. Each subsequent row corresponds to a record of data. A record consists of a series of fields delimited by commas. For information on creating valid field names, see Field Names.

JSON Example

The following JSON example represents a SalesData object with three fields: Name, Amount, and CloseDate. The example corresponds to the preceding CSV example.

```
{
    "fileFormat": {
    "charsetName": "UTF-8",
    "fieldsEnclosedBy": "\"",
    "fieldsDelimitedBy": ",",
    "numberOfLinesToIgnore": 1
  },
  "objects": [
  {
        "connector": "AcmeCSVConnector",
        "description": "",
        "fullyQualifiedName": "SalesData",
        "label": "Sales Data",
        "name": "SalesData",
        "fields": [
        {
            "description": "",
            "fullyQualifiedName": "SalesData.Name",
            "label": "Account Name",
            "name": "Name",
```

```
            "isSystemField": false,
            "isUniqueId": false,
            "isMultiValue": false,
            "type": "Text"
        },
        {
            "description": "",
            "fullyQualifiedName": "SalesData.Amount",
            "label": "Opportunity Amount",
            "name": "Amount",
            "isSystemField": false,
            "defaultValue": "0",
            "isUniqueId": false,
            "type": "Numeric",
            "precision": 10,
            "scale": 2,
        },
        {
            "description": "",
            "fullyQualifiedName": "SalesData.CloseDate",
            "label": "Opportunity Close Date",
            "name": "CloseDate",
            "isSystemField": false,
            "isUniqueId": false,
            "type": "Date",
            "format": "MM/dd/yyyy",
            "fiscalMonthOffset": 0
        }
    ]
 }
  ]
}
```

CSV Format

The External Data API uses a strict format for field values to optimize processing for large sets of data. Note the following when generating .csv files.

- The delimiter for field values in a row must be a comma.
- If a field value contains a comma, a new line, or a double quote, the field value must be contained within double quotes: for example, `"Director of Operations, Western Region"`.
- If a field value contains a double quote, escape the double quote by preceding it with another double quote: for example, `"This is the ""gold"" standard"`.
- Field values aren't trimmed. A space before or after a delimiting comma is included in the field value. A space before or after a double quote generates an error for the row. For example, `John,Smith` is valid. `John, Smith` is valid, but the second value is `" Smith"`. `"John", "Smith"` isn't valid.
- Numeric values in the CSV file can't contain any formatting (such as currency symbols or grouping separators). For example, $1,000.00 is not a valid numeric value; the correct value is 1000.00.
- The maximum numeric value is 36,028,797,018,963,967 and the minimum is -36,028,797,018,963,968.
- Dates must conform to specific formats, and they must match the formats exactly. For more information, see Date Formats.

- At least one column in the CSV file must contain dimension values.
- If column headers are specified, the number of column headers must equal the number of columns in each record.

For more information about field names, see Field Name Restrictions.

External Data Limits

The following limits apply to all supported editions.

Limit	Value
Maximum file size per external data upload	40 GB
Maximum file size for all external data uploads in a rolling 24-hour period	50 GB
Maximum number of external data jobs per dataset that can be run in a rolling 24-hour period	50
Maximum number of characters in a field	32,000
Maximum number of fields in a record	5,000 (including up to 1,000 date fields)
Maximum number of characters for all fields in a record	400,000

EXTERNAL DATA METADATA FORMAT REFERENCE

The metadata describes the structure of external data files. The metadata file is in JSON format. The .json file consists of 3 main sections: file format, object information, and field information. Include all required fields when you create a record, but you can leave out optional fields.

The File Format Section

The file format section of the metadata file specifies information about the format of the data file, including the character set and delimiter character.

Field Name	Type	Required?	Description
`charsetName`	String	No	The character set of the .csv file. If this field is included, it must be set to UTF-8. Example: `"charsetName": "UTF-8"`
`fieldsEnclosedBy`	String	No	The character that can be used to enclose fields in the .csv file. Only double quotes are supported. If a double quote is used within a field, escape it by preceding it with another double quote. If this field is included, it must be set to "\"". Example: `"fieldsEnclosedBy": "\""`
`fieldsDelimitedBy`	String	No	The character that separates field values in the .csv file. If this field is included, it must be set to ",". Example: `"fieldsDelimitedBy": ","`
`linesTerminatedBy`	String	No	*Deprecated. Do not use.*
`numberOfLinesToIgnore`	Number	No	The number of lines for the parser to ignore. (Allows you to specify a header.) • When the .csv file doesn't have a header, set to 0. • When the .csv file has a header, set to 1. Example: `"numberOfLinesToIgnore": 1`

The Objects Section

The objects section of the metadata file specifies information about the top-level database object, including object-level security information, display name, and API name.

Note: The metadata file can contain only 1 object definition.

Field Name	Type	Required?	Description
`rowLevelSecurityFilter`	String	No	The predicate that's used to apply row-level security on the dataset. When entering the predicate in the metadata file JSON, escape double quotes around string values. Example: `"rowLevelSecurityFilter": "'OwnerId' == \"$User.Id\""` For more information about creating the predicate, see the *Tableau CRM Security Implementation Guide*.
`connector`	String	Yes	The string that uniquely identifies the client application. Example: `"connector": "AcmeCSVConnector"`
`description`	String	No	The description of the object. Must be less than 1,000 characters. Example: `"description": "The SalesData object tracks basic sales data."`
`fullyQualifiedName`	String	Yes	The full path that uniquely identifies the record. Must be less than 1,000 characters. Example: `"fullyQualifiedName": "CRM.SalesData"` For information on creating valid field names, see Field Names.
`label`	String	Yes	The display name for the object. Can be up to 40 characters. Example: `"label": "Sales Data"`
`name`	String	Yes	The unique API name for the object. Can be up to 255 characters. Example: `"name": "SalesData"` For information on creating valid field names, see Field Names.

Field Name	Type	Required?	Description
`fields`	Array	Yes	The array of fields for this object.

The Fields Section

The fields section of the metadata file specifies information about each field in the record, including data type and formatting information.

Note: The fields must be in the same order as the CSV columns are in.

Field Name	Type	Required?	Description
`fullyQualifiedName`	String	Yes	The full path that uniquely identifies the field (object.field). Must be less than 1,000 characters. Example: `"fullyQualifiedName": "SalesData.Amount"` For information on creating valid field names, see Field Names.
`label`	String	Yes	The display name for the field. Can be up to 255 characters. Example: `"label": "Opportunity Amount"`
`name`	String	Yes	The unique API name for the field. Can be up to 255 characters. Example: `"name": "Amount"` Note: With Summer '15 Plus, we recommend that field names in datasets use no more than 40 characters. Long field names increase the likelihood of exceeding character limits when you augment datasets, because names are appended. Field names longer than 40 characters are currently supported, but we recommend shortening them to avoid issues. For information on creating valid field names, see Field Names.
`description`	String	No	The description of the field. Must be less than 1,000 characters. Example: `"description": "The Amount field contains the opportunity amount."`
`isSystemField`	Boolean	No	Indicates whether this field is a system field to be excluded from query results. Example: `"isSystemField": false`

Field Name	Type	Required?	Description
`type`	String	Yes	The type of the field. Can be Text, Numeric, or Date. Example: `"type": "Numeric"`
`defaultValue`	String	No	The default value of the field, if any. All numeric types require a default value.
`isUniqueId`	Boolean	No	Indicates whether this field is the primary key for the record. This field is required for incremental extract. Only 1 field can be set to be the unique ID. Note: Only text fields can be unique IDs. Numeric, date, and multivalue fields can't be unique IDs. Example: `"isUniqueId": false`
`isMultiValue`	Boolean	No	For text fields only. Indicates whether the field has multiple values. Applies only to Text fields. Example: `"isMultiValue": false`
`multiValueSeparator`	String	No	For text fields only. The character that separates multiple values. The default is ";". • If `isMultiValue` equals true, specify a value. • If `isMultiValue` equals false, this field can be set to null. Example: `"multiValueSeparator": ";"`
`format`	String	Yes (for Date values only)	The format of the date value. Deprecated for numeric values. See also: Date Formats. Example: `"format": "dd-MM-yy HH:mm:ss" (Date)`
`precision`	Number	Yes (for Numeric values)	The maximum number of digits in a numeric value, or the length of a text value. For numeric values: Includes all numbers to the left and to the right of the decimal point (but excludes the decimal point character). Value can be up to 18. For text values: Value defaults to 255 characters, but can be up to 32,000 characters. Example: `"precision": 10`

Field Name	Type	Required?	Description
`scale`	Number	Yes (for Numeric values)	The number of digits to the right of the decimal point in a numeric value. Must be less than the precision value. Example: `"scale": 2`
`canTruncateValue`	Boolean	No	Indicates whether to truncate a value when the value exceeds the precision. The default is true. • If true, truncates the value. • If false, the row is rejected. Example: `"canTruncateValue": true`
`decimalSeparator`	String	No	For numeric fields only. The character that separates digits in a decimal number. Can be used to handle international number formats where the decimal separator is ",". The default is ".". Example: `"decimalSeparator": ","`
`fiscalMonthOffset`	Number	No	For date fields only. The difference, in months, between the fiscal year and the calendar year. For example, if the fiscal year starts in January, the offset is 0. If the fiscal year starts in October, the offset is 9. Example: `"fiscalMonthOffset": 9` Note: This attribute also controls whether Tableau CRM generates fiscal date fields. To generate fiscal date fields, set `fiscalMonthOffset` to a value other than 0. See also Date Handling in Datasets.
`isYearEndFiscalYear`	Boolean	No	For date fields only. Indicates whether the fiscal year is the year in which the fiscal year ends or begins. Because the fiscal year can start in one calendar year and end in another, specify which year to use for the fiscal year. • If true, then the fiscal year is the year in which the fiscal year ends. The default is true. • If false, then the fiscal year is the year in which the fiscal year begins. Example: `"isYearEndFiscalYear": true` This field is relevant only when `fiscalMonthOffset` is greater than 0. See also Date Handling in Datasets.

Field Name	Type	Required?	Description
`firstDayOfWeek`	Number	No	For date fields only. The first day of the week for the calendar year and, if applicable, fiscal year. Use 0 to set the first day to be Sunday, 1 to set the first day to be Monday, and so on. Use -1 to set the first day to be January 1. The default is -1. Example: `"firstDayOfWeek": 0` See also Date Handling in Datasets.
`isSkipped`	Boolean	No	Indicates whether to skip the field when the data is uploaded. Example: `"isSkipped": true`

Field Name Restrictions

Field names in the .csv file and the metadata file:

- Can contain only alphanumeric and underscore characters
- Must begin with a letter
- Can't end with an underscore
- Can't contain 2 consecutive underscore characters, except when ending with "__c" (case-sensitive)
- Must be unique across all fields of the object

Numeric Formats

An example of a typical numeric value is $1,000,000.99, which is represented as $#,##0.00. You are required to specify the precision and scale of the number. The format is specified by using the following symbols:

Symbol	Meaning
0	One digit. Use to add leading or trailing 0s, like `#,###.00` for $56,375.00.
#	Adds zero or one digit
.	Default symbol used as the decimal separator. Use the `decimalSeparator` field to set the decimal separator to a different symbol.
-	Minus sign
,	Grouping separator
$	Currency sign

Note: The format for numeric values when displayed in the UI defaults to No Format. Existing formatting is removed. For data ingestion, numeric values can't contain any formatting (such as currency symbols or grouping separators). For example, $1,000.00 is not a valid numeric value; the correct value is 1000.00.

Date Formats

For Date fields, specify the format of the date by using one of the following supported formats. Dates must match the format exactly and can't have any extra text. For example, if the date format is "MM-dd-yyyy hh:mm:ss" and the value is "12-31-2015 12:00:00.0000", the upload fails because the value has extra milliseconds.

Note: The date formats listed here are the two-digit versions for date fields that use leading zeros; for example, **03/06/14 09:01:06 AM**. If a date field doesn't have leading zeros, use the one-digit version of the format. For example, use the format `M/d/yy h:m:s a` for date values such as **3/6/14 9:1:26 AM**. If you use a two-digit format for a field, rows containing values with one-digit date parts will fail.

The timestamp part of each date format is optional.

Format	Sample Value
yyyy-MM-dd'T'HH:mm:ss.SSS'Z'	2014-04-29T16:53:34.000Z
yy-MM-dd'T'HH:mm:ss.SSS'Z'	14-04-29T16:53:34.000Z
yyyy-MM-dd'T'HH:mm:ss'Z'	2014-04-29T16:53:34Z
yy-MM-dd'T'HH:mm:ss'Z'	14-04-29T16:53:34Z
yyyy-MM-dd HH:mm:ss	2014-06-03 11:31:45
yy-MM-dd HH:mm:ss	14-06-03 11:31:45
dd.MM.yyyy HH:mm:ss	03.06.2014 11:31:45
dd.MM.yy HH:mm:ss	03.06.14 11:31:45
dd/MM/yyyy HH:mm:ss	03/06/2014 11:31:45
dd/MM/yy HH:mm:ss	03/06/14 11:31:45
dd/MM/yyyy hh:mm:ss a	03/06/2014 11:31:45 AM
dd/MM/yy hh:mm:ss a	03/06/14 11:31:45 AM
dd-MM-yyyy HH:mm:ss	03-06-2014 11:31:45
dd-MM-yy HH:mm:ss	03-06-14 11:31:45
dd-MM-yyyy hh:mm:ss a	03-06-2014 11:31:45 AM
dd-MM-yy hh:mm:ss a	03-06-14 11:31:45 AM
MM/dd/yyyy hh:mm:ss a	06/03/2014 11:31:45 AM
MM/dd/yy hh:mm:ss a	06/03/14 11:31:45 AM
MM-dd-yyyy hh:mm:ss a	06-03-2014 11:31:45 AM

Format	Sample Value
MM-dd-yy hh:mm:ss a	06-03-14 11:31:45 AM
HH:mm:ss dd/MM/yyyy	11:31:45 03/06/2014
HH:mm:ss dd/MM/yy	11:31:45 03/06/14

These formats use the following symbols:

Symbol	Meaning
yyyy or yy	Four-digit year (yyyy) or two-digit year (yy)
MM	Two-digit month (01–12)
M	One-digit month when month less than 10 (1–12)
dd	Two-digit day (01–31)
d	One-digit day when day less than 10 (1–31)
'T'	A separator that indicates that time of day follows
HH	Two-digit hour (00–23)
H	One-digit hour when hour less than 10 (0–23)
mm	Two-digit minute (00–59)
m	One-digit minute when minute less than 10 (0–59)
ss	Two-digit second (00–59)
s	One-digit second when second less than 10 (0–59)
SSS	Optional three-digit milliseconds (000–999)
'Z'	The reference UTC time zone