

Table 6-2. Some `read_csv/read_table` function arguments

Argument	Description
<code>path</code>	String indicating filesystem location, URL, or file-like object
<code>sep</code> or <code>delimiter</code>	Character sequence or regular expression to use to split fields in each row
<code>header</code>	Row number to use as column names; defaults to 0 (first row), but should be <code>None</code> if there is no header row
<code>index_col</code>	Column numbers or names to use as the row index in the result; can be a single name/number or a list of them for a hierarchical index
<code>names</code>	List of column names for result, combine with <code>header=None</code>
<code>skiprows</code>	Number of rows at beginning of file to ignore or list of row numbers (starting from 0) to skip.
<code>na_values</code>	Sequence of values to replace with NA.
<code>comment</code>	Character(s) to split comments off the end of lines.
<code>parse_dates</code>	Attempt to parse data to <code>datetime</code> ; <code>False</code> by default. If <code>True</code> , will attempt to parse all columns. Otherwise can specify a list of column numbers or name to parse. If element of list is tuple or list, will combine multiple columns together and parse to date (e.g., if date/time split across two columns).
<code>keep_date_col</code>	If joining columns to parse date, keep the joined columns; <code>False</code> by default.
<code>converters</code>	Dict containing column number or name mapping to functions (e.g., <code>{ 'foo' : f }</code> would apply the function <code>f</code> to all values in the 'foo' column).
<code>dayfirst</code>	When parsing potentially ambiguous dates, treat as international format (e.g., 7/6/2012 -> June 7, 2012); <code>False</code> by default.
<code>date_parser</code>	Function to use to parse dates.
<code>nrows</code>	Number of rows to read from beginning of file.
<code>iterator</code>	Return a <code>TextParser</code> object for reading file piecemeal.
<code>chunksize</code>	For iteration, size of file chunks.
<code>skip_footer</code>	Number of lines to ignore at end of file.
<code>verbose</code>	Print various parser output information, like the number of missing values placed in non-numeric columns.
<code>encoding</code>	Text encoding for Unicode (e.g., 'utf-8' for UTF-8 encoded text).
<code>squeeze</code>	If the parsed data only contains one column, return a <code>Series</code> .
<code>thousands</code>	Separator for thousands (e.g., ',' or '.').