**cols Property**

cols is an array of objects describing the ID and type of each column. Each property is an object with the following properties (case-sensitive):

* type [*Required*] Data type of the data in the column. Supports the following string values (examples include the v: property, described later):
  1. 'boolean' - JavaScript boolean value ('true' or 'false'). *Example value*: v:'true'
  2. 'number' - JavaScript number value. *Example values*: v:7 , v:3.14, v:-55
  3. 'string' - JavaScript string value. *Example value*: v:'hello'
  4. 'date' - JavaScript Date object (zero-based month), with the time truncated. *Example value*: v:new Date(2008, 0, 15)
  5. 'datetime' - JavaScript Date object including the time. *Example value*: v:new Date(2008, 0, 15, 14, 30, 45)
  6. 'timeofday' - Array of three numbers and an optional fourth, representing hour (0 indicates midnight), minute, second, and optional millisecond. *Example values*: v:[8, 15, 0], v: [6, 12, 1, 144]
* id [*Optional*] String ID of the column. Must be unique in the table. Use basic alphanumeric characters, so the host page does not require fancy escapes to access the column in JavaScript. Be careful not to choose a JavaScript keyword. *Example*: id:'col\_1'
* label [*Optional*] String value that some visualizations display for this column. *Example*: label:'Height'
* pattern [*Optional*] String pattern that was used by a data source to format numeric, date, or time column values. This is for reference only; you probably won't need to read the pattern, and it isn't required to exist. The Google Visualization client does not use this value (it reads the cell's formatted value). If the DataTable has come from a data source in response to a query with a [format](https://developers.google.com/chart/interactive/docs/querylanguage#Format) clause, the pattern you specified in that clause will probably be returned in this value. The recommended pattern standards are the ICU [DecimalFormat](http://icu-project.org/apiref/icu4j/com/ibm/icu/text/DecimalFormat.html)and [SimpleDateFormat](http://icu-project.org/apiref/icu4j/com/ibm/icu/text/SimpleDateFormat.html).
* p [*Optional*] An object that is a map of custom values applied to the cell. These values can be of any JavaScript type. If your visualization supports any cell-level properties, it will describe them; otherwise, this property will be ignored.**Example:** p:{style: 'border: 1px solid green;'}.

**cols Example**

cols: [{id: 'A', label: 'NEW A', type: 'string'},

{id: 'B', label: 'B-label', type: 'number'},

{id: 'C', label: 'C-label', type: 'date'}]

**rows Property**

The rows property holds an array of row objects.

Each row object has one required property called c, which is an array of cells in that row. It also has an optional p property that defines a map of arbitrary custom values to assign to the whole row. If your visualization supports any row-level properties it will describe them; otherwise, this property will be ignored.

**Cell Objects**

Each cell in the table is described by an object with the following properties:

* v [*Optional*] The cell value. The data type should match the column data type. If the cell is null, the vproperty should be null, though it can still have f and p properties.
* f [*Optional*] A string version of the v value, formatted for display. Typically the values will match, though they do not need to, so if you specify Date(2008, 0, 1) for v, you should specify "January 1, 2008" or some such string for this property. This value is not checked against the v value. The visualization will not use this value for calculation, only as a label for display. If omitted, a string version of v will be automatically generated using the default formatter. The f values can be modified using your own formatter, or set with setFormattedValue() or setCell(), or retrieved with getFormattedValue()