# Data Queries

This page describes how to send a query to a data source that supports the Chart Tools Datasource protocol.

## Overview

A Datasource is a web service that supports the Chart Tools Datasource protocol. You can send a SQL query to a Datasource, and in response you will receive a DataTable populated with the appropriate information. Some examples of Datasources include Google Spreadsheets (/chart/interactive/docs/spreadsheets) and SalesForce.

## Sending a request

**To send a request:**

1. Instantiate a Query (/chart/interactive/docs/reference#Query) object with the URL of your Datasource. The URL should indicate what data is being requested, in a syntax understood by that data source.
2. Optionally specify request options such as sending method as an optional second parameter in the `Query` object constructor (see the Query constructor's `opt_options` (/chart/interactive/docs/reference#Query) parameter for details):
3. Optionally add a query language string (/chart/interactive/docs/querylanguage) to sort or filter the results, and then send the request. Datasources are not required to support the Chart Tools Datasource query language. If the Datasource does not support the query language, it will ignore the SQL query string, but still return a `DataTable`. The query language is a SQL language variant; read the full query language syntax here (/chart/interactive/docs/querylanguage).
4. Send the query, specifying a callback handler that will be called when the response is received: see next section for details.

Here's an example of sending a request for data in a Google Spreadsheet cell range; to learn how to get the URL for a Google Spreadsheet, see here

(/chart/interactive/docs/spreadsheets#Google_Spreadsheets_as_a_Data_Source):

```
function initialize() {
  var opts = {sendMethod: 'auto'};
  // Replace the data source URL on next line with your data source URL.
  var query = new google.visualization.Query('http://spreadsheets.google.com?key

  // Optional request to return only column C and the sum of column B, grouped b
  query.setQuery('select C, sum(B) group by C');

  // Send the query with a callback function.
  query.send(handleQueryResponse);
}

function handleQueryResponse(response) {
  // Called when the query response is returned.
  ...
}
```

If you are sending your query from within Apps Script, be sure to use `IFRAME` mode (/apps-script/reference/html/sandbox-mode).

## Processing the response

Your response handler function will be called when the request returns. The parameter passed in to your response handler function is of type google.visualization.QueryResponse (/chart/interactive/docs/reference#QueryResponse). If the request was successful, the response contains a data table (class `google.visualization.DataTable`). If the request failed, the response contains information about the error, and no `DataTable`.

**Your response handler should do the following:**

1. Check whether the request succeeded or failed by calling `response.isError()`. You shouldn't need to display any error messages to the user; the Visualization library will display an error message for you in your container `<div>`. However, if you do want to handle errors manually, you can use the `goog.visualization.errors`

(/chart/interactive/docs/reference#errordisplay) class to display custom messages (see the Query Wrapper Example (/chart/interactive/docs/examples#querywrapper) for an example of custom error handling).

2. If the request succeeded, the response will include a **`DataTable`** that you can retrieve by calling **`getDataTable()`**. Pass it to your chart.

The following code demonstrates handling the previous request to draw a pie chart:

```
function handleQueryResponse(response) {

  if (response.isError()) {
    alert('Error in query: ' + response.getMessage() + ' ' + response.getDetaile
    return;
  }

  var data = response.getDataTable();
  var chart = new google.visualization.PieChart(document.getElementById('chart_d
  chart.draw(data, {width: 400, height: 240, is3D: true});
}
```

## Reading CSV files

If you want to build a chart out of CSV (comma-separated values) data, you have two choices. Either manually convert the CSV data into the Google Charts datatable format (/chart/interactive/docs/datatables_dataviews#creatingpopulating), or place the CSV file on the web server serving the chart, and query it using the technique on this page.

## More information

- Query Language Syntax (/chart/interactive/docs/querylanguage) - Describes the syntax of the language used to make data queries.
- Query Class (/chart/interactive/docs/reference#Query) - Reference page for the class that wraps a query.

- QueryResponse Class (/chart/interactive/docs/reference#QueryResponse) - Reference page for the class that wraps the response to a query.