// Begin with a data file that has been uploaded to the BLOB repository.

// Pull data from the uploaded Blob.

// Rapture keeps track of the types of blobs.

// If the type is defined as CSV then the Pull operation will automatically break it into its component values.

data <-- "blob://demo/dayOneDemo.csv";

// Or the blob can be opened with the file command and treated as a string

stringdata <-- file("blob://demo/dayOneDemo.csv");

data = split(stringdata, "\n", true);

i = 0;

for datum in data do

data[i]= split(datum, ",", true);

i = i+1;

end

println(data[1]);// The data can now be stored as one or more Series documents.

// A series is a set of key, value pairs.

// The series path is based on the first (N-2) columns.

// So if the data is A,B,C,D,E it would store the values {D : E} as series://demo/A/B/C

data --> “series://demo”;

// This is the equivalent of the following:

for line in data do

location = "series://demo";

columns = size(line);

if (columns >= 2) do

for field in line do

if (columns == 1) do

value = field;

else if (columns == 2) do

key = field;

else do

location = location+"/"+field;

end

columns = columns - 1;

end

#series.addDoubleToSeries(location, key, value);

else do

println("Line has less than 2 values: it will be skipped");

end

end// The Document repository stores JSON format documents.

// The CSV can be written as a Document and is automatically converted to a JSON Map

data --> “document://demo/doc”;

// Data pulled from a Document can be accessed as a map:

docmap <-- “document://demo/doc”;

println(docu.HIST.Provider\_1a.USGG2YR\_Index\_Dummy.DAILY.LOW);

// Data can be read as a String using the #doc.getDoc API call

// It can then be expanded to a Map using fromjson

docu = #doc.getDoc("document://demo/doc");

documap = fromjson(docu);

println(documap.HIST.Provider\_1a.USGG2YR\_Index\_Dummy.DAILY.LOW);

// If we decide to store the CSV in the document repository as a list of records then here is how to read the records and convert them back to a list.

// Set up some dummy data

j1 = "{ \"series\": \"HIST\", \"provider\": \"Provider\_1a\", \"index\_id\": \"AUDUSD\_CURNCY\_Dummy\", \"frequency\": \"DAILY\", \"date\": \"20141028\", \"price\_type\": \"PX\_BID\", \"index\_price\": 0.91 }";

j2 = "{ \"series\": \"HIST\", \"provider\": \"Provider\_1a\", \"index\_id\": \"AUDUSD\_CURNCY\_Dummy\", \"frequency\": \"DAILY\", \"date\": \"20141029\", \"price\_type\": \"PX\_BID\", \"index\_price\": 0.92 }";

j3 = "{ \"series\": \"HIST\", \"provider\": \"Provider\_1a\", \"index\_id\": \"AUDUSD\_CURNCY\_Dummy\", \"frequency\": \"DAILY\", \"date\": \"20141030\", \"price\_type\": \"PX\_BID\", \"index\_price\": 0.93 }";

s1 = fromjson(j1);

s2 = fromjson(j2);

s3 = fromjson(j3);

s = [ s1, s2, s2 ];

// Note that Rapture requires Documents to be Maps, so although a list of records is valid JSON it won’t be accepted by the DocApi.

// #doc.putDoc("document://demo/flat", json(s));

// The above won’t work; it needs to be made into a map

m = { };

m.values = s;

#doc.putDoc("document://demo/flat", json(m));

// Read the records and convert back into lists

flatDoc = #doc.getDoc("document://demo/flat");

records = fromjson(flatDoc);

csv = [ ] ;

keySet = keys(records.values[0]);

csv.add(keySet);

for record in records.values do

row = [ ] ;

for key in keySet do

row.add(record[key]);

end

csv.add(row);

end