Loading and Reading Flat Files

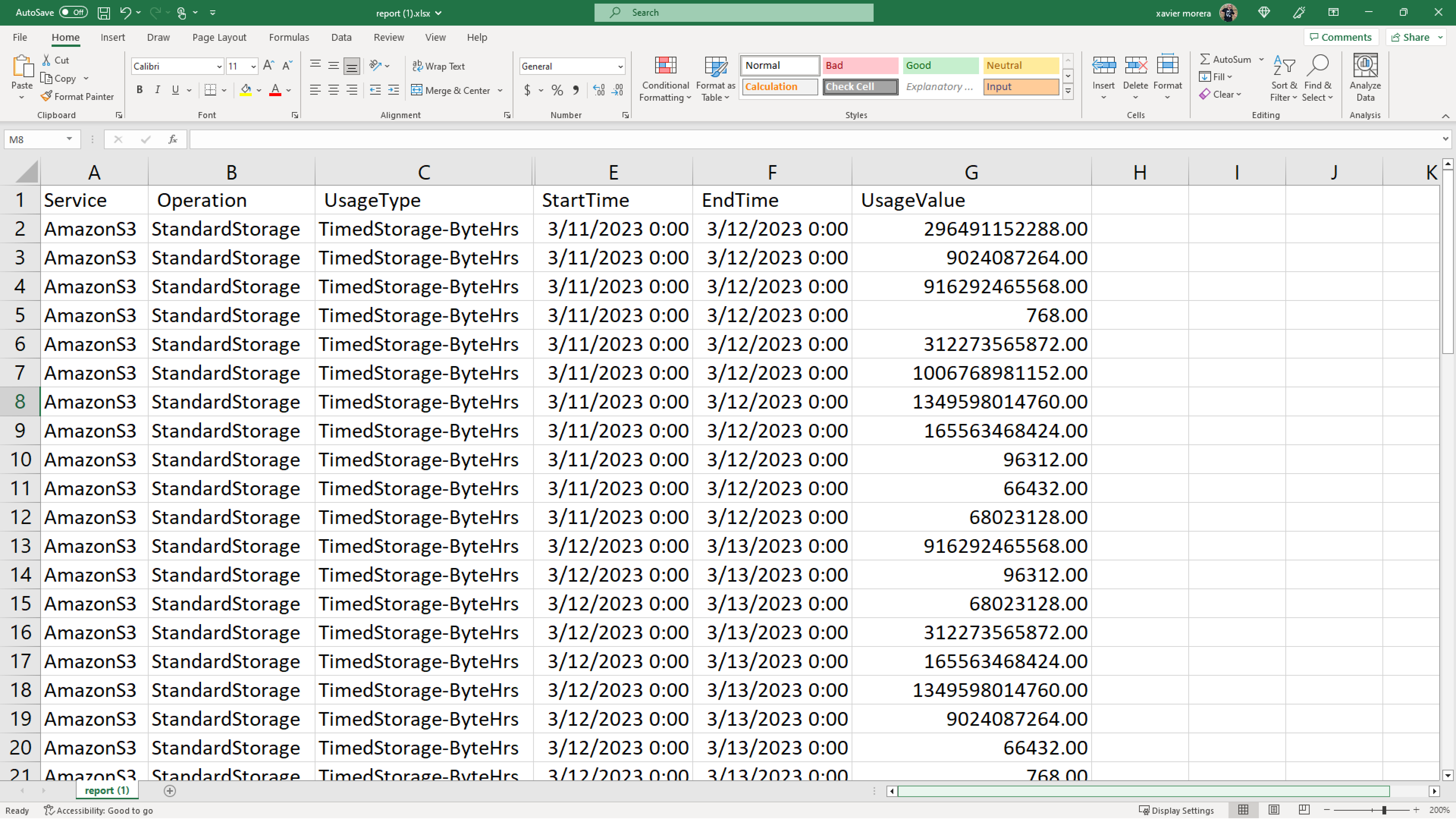
Xavier Morera

Helping developers create amazing applications

@xmorera / www.xaviermorera.com / www.bigdatainc.org

What comes to your mind if I say:

"Data in a file"



Service	Operation	UsageType	StartTime	EndTime	UsageValue
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/11/2023 0:00	3/12/2023 0:00	296491152288.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/11/2023 0:00	3/12/2023 0:00	9024087264.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/11/2023 0:00	3/12/2023 0:00	916292465568.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/11/2023 0:00	3/12/2023 0:00	768.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/11/2023 0:00	3/12/2023 0:00	312273565872.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/11/2023 0:00	3/12/2023 0:00	1006768981152.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/11/2023 0:00	3/12/2023 0:00	1349598014760.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/11/2023 0:00	3/12/2023 0:00	165563468424.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/11/2023 0:00	3/12/2023 0:00	96312.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/11/2023 0:00	3/12/2023 0:00	66432.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/11/2023 0:00	3/12/2023 0:00	68023128.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/12/2023 0:00	3/13/2023 0:00	916292465568.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/12/2023 0:00	3/13/2023 0:00	96312.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/12/2023 0:00	3/13/2023 0:00	68023128.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/12/2023 0:00	3/13/2023 0:00	312273565872.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/12/2023 0:00	3/13/2023 0:00	165563468424.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/12/2023 0:00	3/13/2023 0:00	1349598014760.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/12/2023 0:00	3/13/2023 0:00	9024087264.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/12/2023 0:00	3/13/2023 0:00	66432.00
AmazonS3	StandardStorage	TimedStorage-ByteHrs	3/12/2023 0:00	3/13/2023 0:00	768.00

```json
{
    "Date":"2021-12-01T12:34:00",
    "Temperature":7,
    "feelslike":3.14,
    "Summary":"overcast clouds"
}
```

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<root>
  <Date>2021-12-01T12:34:00</Date>
  <Temperature>7</Temperature>
  <feelslike>3.14</feelslike>
  <Summary>overcast clouds</Summary>
</root>
```

A lot of data is stored in *flat files*

Flat Files

```
Name,Age,Gender,Email
Xavier Smith,40,Male,xavier@lupo.ai
Irene Doe,22,Female,irene@lupo.ai
Mary John,50,Female,mary@lupo.ai
```

Flat Files

```
Name,Age,Gender,Email
Xavier Smith,40,Male,xavier@lupo.ai
Irene Doe,22,Female,irene@lupo.ai
Mary John,50,Female,mary@lupo.ai
```

Flat Files

```
Name,Age,Gender,Email
Xavier Smith,40,Male,xavier@lupo.ai
Irene Doe,22,Female,irene@lupo.ai
Mary John,50,Female,mary@lupo.ai
```

Flat Files

```
Name,Age,Gender,Email
Xavier Smith,[illegible],Male,xavier@lupo.ai
Irene Doe,22,Female,irene@lupo.ai
Mary John,50,Female,mary@lupo.ai
```

Types of Flat Files

Types of Flat Files

CSV

Fixed-width

TSV

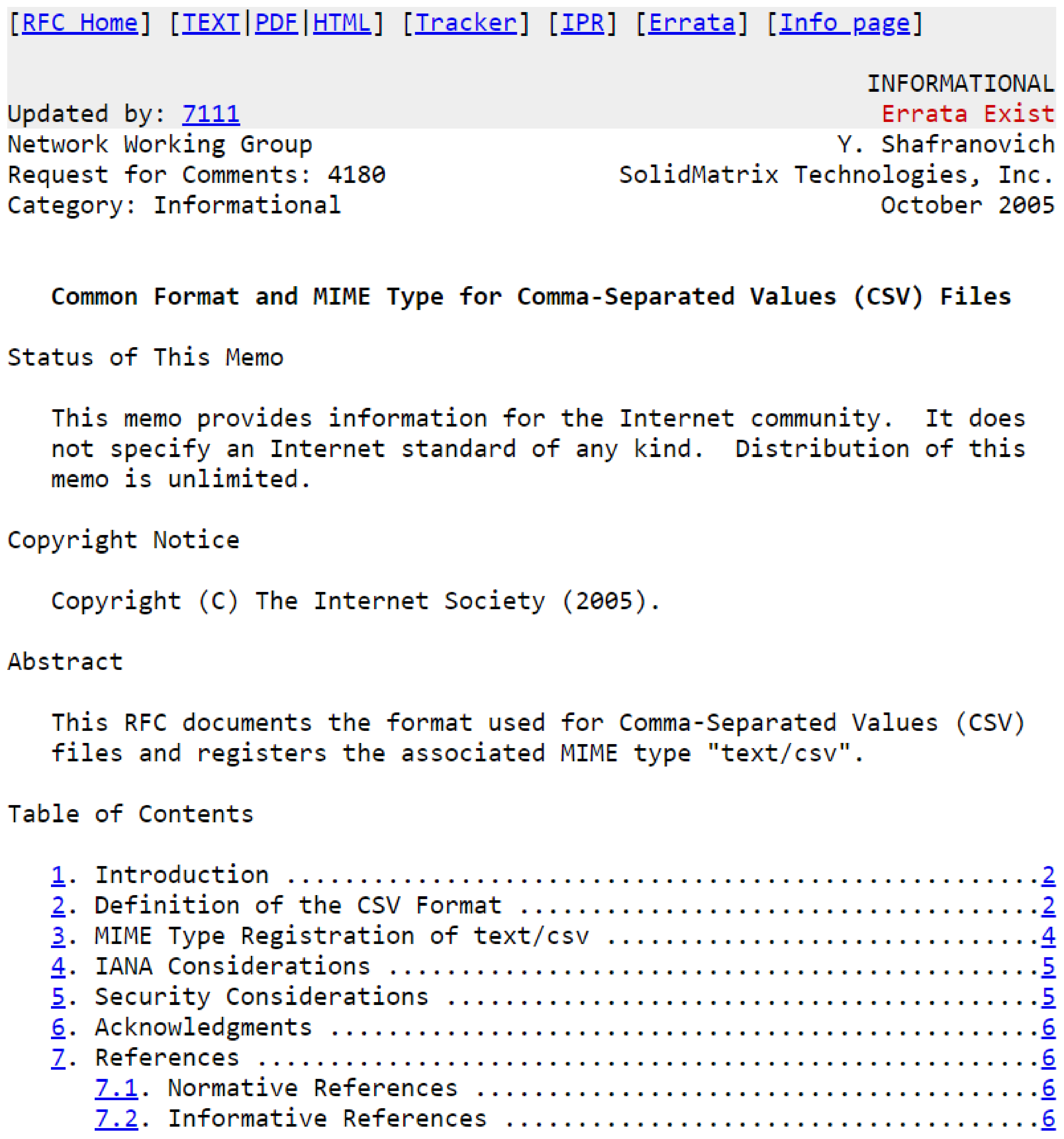
```
Name,Age,Gender,Email
Xavier Smith,40,Male,xavier@lupo.ai
Irene Doe,22,Female,irene@lupo.ai
Mary John,50,Female,mary@lupo.ai
```

CSV

Comma-separated values, which uses

- Commas to separate fields, potentially including a header
- New lines to separate records, can have enclosing character (")

Perhaps the most widely used type of flat file



[RFC Home] [TEXT|PDF|HTML] [Tracker] [IPR] [Errata] [Info page]

```
                                                           INFORMATIONAL
Updated by: 7111                                           Errata Exist
Network Working Group                                    Y. Shafranovich
Request for Comments: 4180           SolidMatrix Technologies, Inc.
Category: Informational                                     October 2005
```

Common Format and MIME Type for Comma-Separated Values (CSV) Files

Status of This Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Abstract

This RFC documents the format used for Comma-Separated Values (CSV) files and registers the associated MIME type "text/csv".

Table of Contents

RFC 4180

```
Name           Age  Gender   Email
Xavier Smith   40   Male     xavier@lupo.ai
Irene Doe      22   Female   irene@lupo.ai
Mary John      50   Female   mary@lupo.ai
```

Fixed-width

Fixed-width files

Uses a fixed number of characters for each field

- Spaces used to pad shorter fields, which might make difficult to read

Data could get truncated

```
Name    ->->-->Age->-->Gender->Email
Xavier Smith->40 ->-->Male->-->xavier@lupo.ai
Irene Doe ->-->22 ->-->Female->irene@lupo.ai
Mary John ->-->50 ->-->Female->mary@lupo.ai
```

TSV

Tab-separated values, which uses

- Tabs to separate fields
- New lines to separate records

Similar to CSV, but uses tabs instead of commas

Flat Files

```
Name,           Age,  Gender,  Email
Xavier Smith,   40,   Male,    xavier@lupo.ai
Irene Doe,      22,   Female,  irene@lupo.ai
Mary John,      50,   Female,  mary@lupo.ai
```

The question is…

How exactly do you process and parse flat files?

Reading manually a flat file

CSV Reader Module

Reading and parsing a flat file, like a CSV...

"Works"

* but some issues may come up and it is a lot of work to do it manually every time

Previous topic
File Formats

Next topic
configparser — Configuration file parser

This Page
Report a Bug
Show Source

«

`csv` — CSV File Reading and Writing

Source code: Lib/csv.py

The so-called CSV (Comma Separated Values) format is the most common import and export format for spreadsheets and databases. CSV format was used for many years prior to attempts to describe the format in a standardized way in **RFC 4180**. The lack of a well-defined standard means that subtle differences often exist in the data produced and consumed by different applications. These differences can make it annoying to process CSV files from multiple sources. Still, while the delimiters and quoting characters vary, the overall format is similar enough that it is possible to write a single module which can efficiently manipulate such data, hiding the details of reading and writing the data from the programmer.

The `csv` module implements classes to read and write tabular data in CSV format. It allows programmers to say, “write this data in the format preferred by Excel,” or “read data from this file which was generated by Excel,” without knowing the precise details of the CSV format used by Excel. Programmers can also describe the CSV formats understood by other applications or define their own special-purpose CSV formats.

The `csv` module’s `reader` and `writer` objects read and write sequences. Programmers can also read and write data in dictionary form using the `DictReader` and `DictWriter` classes.

See also:

PEP 305 – CSV File API
The Python Enhancement Proposal which proposed this addition to Python.

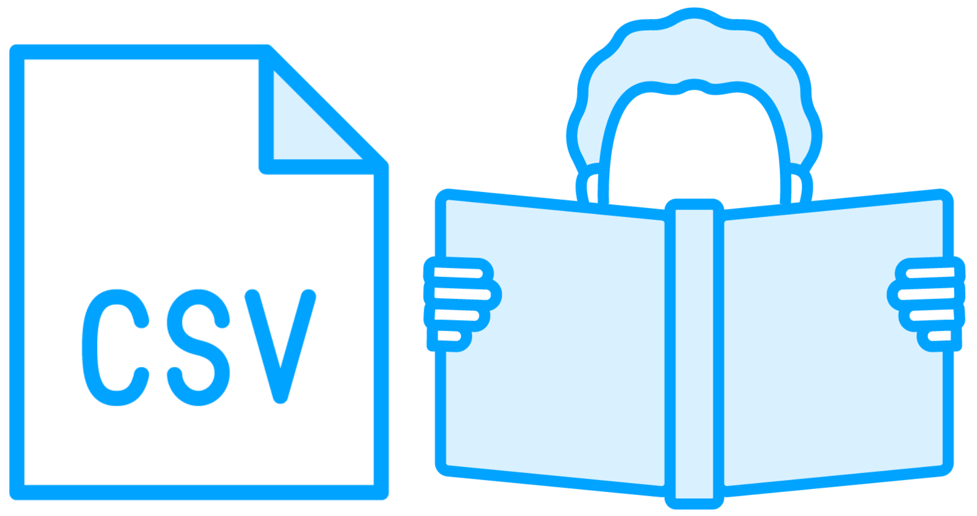
Module Contents

The `csv` module defines the following functions:

`csv.`**`reader`**`(`*`csvfile, dialect='excel', **fmtparams`*`)`

Return a reader object which will iterate over lines in the given *csvfile*. *csvfile* can be any object which supports the iterator protocol and returns a string each time its `__next__()` method is

The CSV Reader Module



Read tabular data

- CSV files

Set different delimiters

- TSV files

Read (and write) to preferred formats

- Excel-style flat file

Define your own special-purpose CSV formats

The CSV Reader module

Libraries for Working with Flat Files

Don't reinvent the wheel

A Flat File

```
Name,Age,Gender,Email
Xavier Smith,40,Male,xavier@lupo.ai
Irene Doe,22,Female,irene@lupo.ai
Mary John,50,Female,mary@lupo.ai
```

Another Very Numerical Flat File

```
1.00  2.00  3.00
4.00  5.00  6.00
7.00  8.00  9.00
```

A Slightly More Complex Flat File

```
name,age,gender,city,income
John,25,M,New York,50000
Jane,30,F,,60000
Bob,35,M,Seattle,70000
Alice,28,F,,55000
Charlie,42,M,Los Angeles,80000
David,29,M,Boston,65000
Eve,31,F,Miami,75000
Frank,45,M,,90000
Grace,27,F,Atlanta,55000
Henry,38,M,Denver,85000
```

A Slightly More Complex Flat File

```
name,age,gender,city,income
John,25,M,New York,50000
Jane,30,F,,60000
Bob,35,M,Seattle,70000
Alice,28,F,,55000
Charlie,42,M,Los Angeles,80000
David,29,M,Boston,65000
Eve,31,F,Miami,75000
Frank,45,M,,90000
Grace,27,F,Atlanta,fifty five thousand
Henry,38,M,Denver,85000
```

A Slightly More Complex Flat File

```
name,age,gender,city,income
John,25,M,New York,50000
"Jane, from CR",30,F,,60000
Bob,35,M,Seattle,70000
Alice,28,F,,55000
Charlie,42,M,Los Angeles,80000
David,29,M,Boston,65000
Eve,31,F,Miami,75000
Frank,45,M,,90000
Grace,27,F,Atlanta,55000
Henry,38,M,Denver,85000
```

Libraries for Loading and Reading Flat Files

Pandas

Numpy

Takeaway

Many ways to store data in a file

- You may think "Excel"
- XML and JSON
- Flat files are very commonly used

Flat file

- Contains data in plain format
- Simple, two-dimensional structure

Takeaway

Different types of flat files

- CSV, TSV, fixed-width…

Parse flat files

- Manually
- CSV Reader module
- Pandas and Numpy

Up Next:

Import Text Files with Numpy