How can I export a dataframe to a text file in R?

Authored by

stats writer

June 24, 2024

Exporting a dataframe to a text file in R refers to the process of saving the contents of a dataframe, a tabular data structure, to a text file. This can be done using the "write.table" function, which converts the dataframe into a delimited text format and saves it as a text file on the local system. This allows for easy sharing and manipulation of data with other software or users. The exported text file can also be imported back into R for further analysis or processing.

Use R base function wirte.table() to export the data from R DataFrame to a text file. Besides this, R also provides a third-party package to write the text file. In this article, I will explain different ways to export data from a DataFrame to a text file.

1. Quick Examples of Export to Text File

Following are quick examples.

```
# Quick Examples

# Write tab delimiter text file
write.table(df,"/Users/admin/fileOut.txt")

# Write comma delimiter text file
write.table(df,"/Users/admin/fileOut.txt", sep=",")

# Ignore row names/numbers
write.table(df,"/Users/admin/fileOut.txt", row.names=FALSE)

# Replaces NA with specified string.
write.table(df,"/Users/admin/fileOut.txt", na='')

# Write with UTF-8 encoding
write.table(df,"/Users/admin/fileOut.txt", fileEncoding='UTF-8')
```

Let's create an R DataFrame and run these examples.

```
# Create DataFrame
df <- data.frame(
id = c(10,11,12,13),
name = c('sai','ram','deepika','sahithi'),
gender = c('M','M',NA,'F'),
dob = as.Date(c('1990-10-02','1981-3-24','1987-6-14','1985-8-16')),
state = c('CA','NY',NA,NA)
```

```
)
df
```

Yields below output.

```
# Output
id name gender dob state
1 10 sai M 1990-10-02 CA
2 11 ram M 1981-03-24 NY
3 12 deepika <NA> 1987-06-14 <NA>
4 13 sahithi F 1985-08-16 <NA>
```

2. Usage of write.table()

R base function has a `write.table()` function which is used to export the data frame data into a text file. This function takes several arguments that can be used to specify what and how you wanted to write. For example use `sep` to specify the delimiter, use `quote=FALSE` to ignore quotes for the values, use `row.names=FALSE` to ignore row names.

Note that using write.table() you can't ignore column names while writing.

2.1 Syntax of write.table()

The following is the syntax of `write.table()` function.

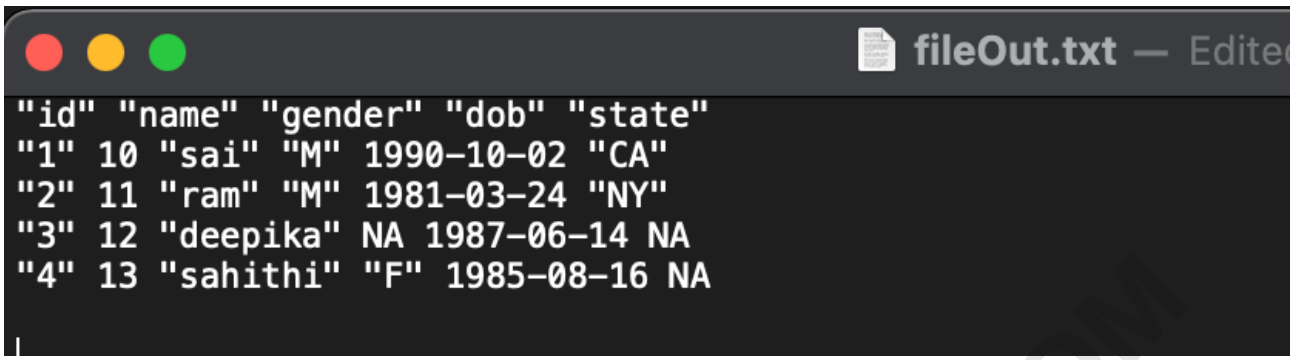
```
# Syntax of write.table()
write.table(x, file = "", append = FALSE, quote = TRUE, sep = " ",
eol = "n", na = "NA", dec = ".", row.names = TRUE,
col.names = TRUE, qmethod = c("escape", "double"),
fileEncoding = "")
```

2.2 Export to Text File in R using write.table()

By using the above specified `write.table()` syntax export the data from data frame to a text file. This by default exports tab delimiter fields.

```
# Write tab delimiter text file
write.table(df,"/Users/admin/fileOut.txt")
```

This writes the fileOut.txt at the specified path.

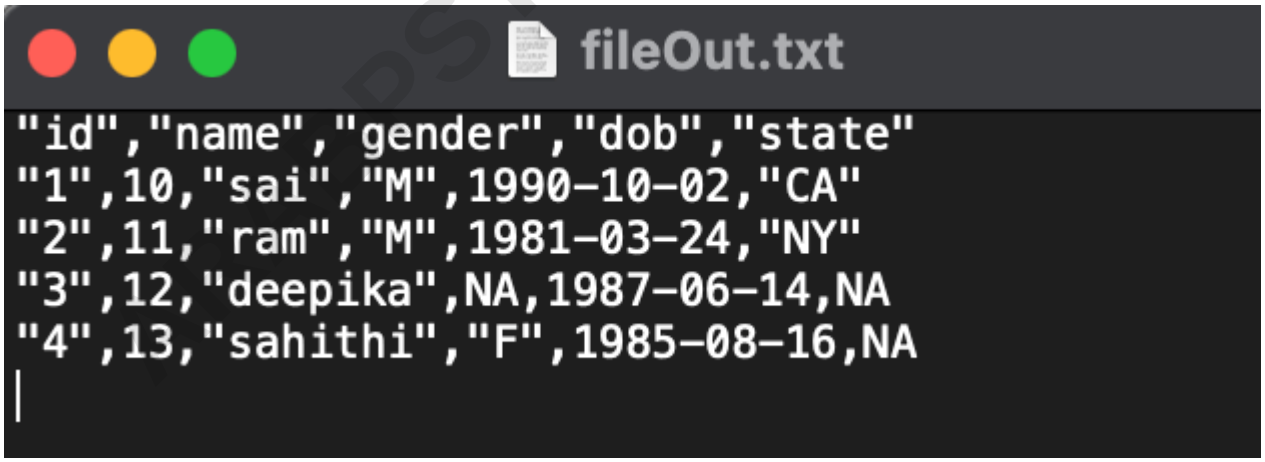


2.3 Export with comma Separator

By default `write.table()` writes data with tab-separated fields, if you wanted to use a custom delimiter use `sep` argument.

```
# Write comma delimiter text file
write.table(df,"/Users/admin/fileOut.txt", sep=",")
```

This creates a text file with comma-separated columns.



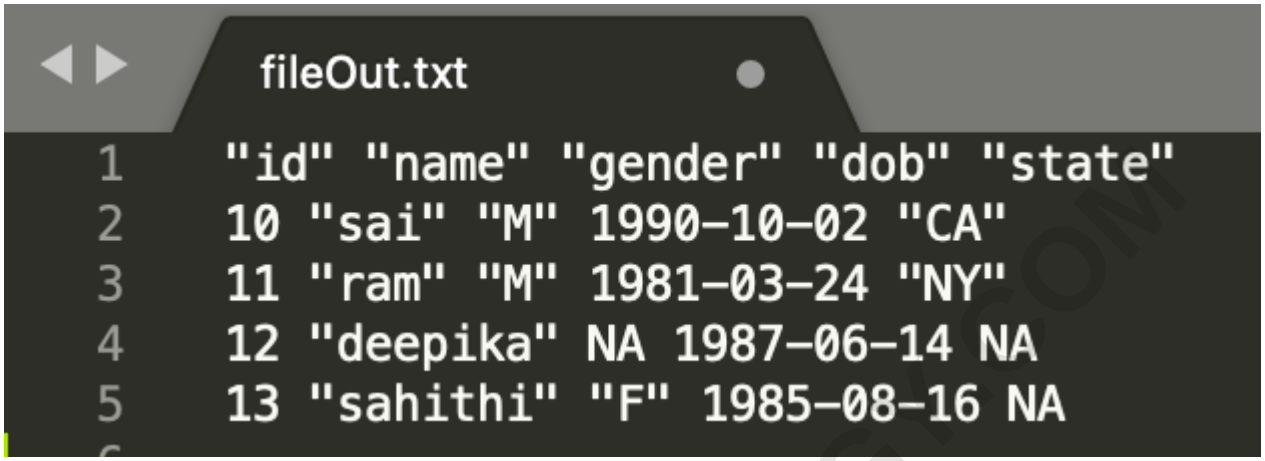
2.4 Ignore Row Names/Numbers

Use `row.names=FALSE` argument to not to export row names/numbers to a file.

```
# Ignore row names/numbers
```

```
write.table(df,"/Users/admin/fileOut.txt", row.names=FALSE)
```

Writes below content.



2.5 Replace NA Missing Values

If you have NA values on DataFrame, you can specify how you wanted to write these NA values on text file by using `na` argument. The following example replaces all NA values with empty string.

```
# Replaces NA with specified string.
write.table(df,"/Users/admin/fileOut.txt", na='')
```

3. Use readr Package to Export Text File

If you are dealing with larger datasets, consider using the `write_csv()` function from the `readr` package. Since `readr` is a third-party library, you need to install it first by running `install.packages('readr')`. After the installation is complete, load the `readr` library by using `library('readr')` to access the `write_csv()` function.

This by default exports fields separated by comma delimiter and you can't change it.

```
# Load readr package
library(readr)
```

# Write to CSV file
write_csv(df, "/Users/admin/new_file.csv")

4. Conclusion

In this article, you have learned write.table() is used to export the text file in R. This function by default exports fields in the tab separator, you can change this by using `sep` argument.

Related Articles

References