

Intro

Create a spark application which will :

- Read a csv file from local filesystem
- Remove rows where any string column is an empty string or just spaces (" ")
Note : empty string is not same as null
- Convert data-type and names of the columns as per user's choice
- For a given column, provide profiling information: total number of unique values, count of each unique value. Exclude nulls

The application should be generic, so that it can be used for any csv file, on the basis of arguments provided by the user. There should not be any hard-coding of column names. The sample.csv provided below is just an example.

Source code should be placed in an open repository (e.g. github.com), no zip archives. Prepare this task as you do with features for your production (abstractions, code style, etc).

Focus on business logic, not on parsing command-line arguments. If input params can be set only via modifying several first lines of the source code of your app - it will be fine.

Step 1

Load a csv file from the local filesystem to spark dataframe. First row in the file is header information, which should become the column's for the dataframe.

Sample.csv

```
"name","age","birthday","gender"
"John","26","26-01-1995","male"
"Lisa","xyz","26-01-1996","female"
,"26","26-01-1995","male"
"Julia"," ","26-01-1995","female"
" ","26-01-1995",
"Pete"," ","26-01-1995"," "
```

Dataframe

Columns - name (string), age (string), birthday (string), gender (string)

name	+++	age	+++	birthday	+++	gender
"John"	+++	"26"	+++	"26-01-1995"	+++	"male"
"Lisa"	+++	"xyz"	+++	"26-01-1995"	+++	"female"

null	+++	"26"	+++	"26-01-1995"	+++	"male"
"Julia"	+++	" "	+++	"26-01-1995"	+++	"female"
" "	+++	null	+++	"26-01-1995"	+++	"male"
"Pete"	+++	" "	+++	"26-01-1995"	+++	" "

Step 2

Filter data and remove rows from the dataframe, where the column is of string type and has empty value or just spaces. Do not remove rows where the value for the column is null

Dataframe

Columns - name (string), age (string), birthday (string), gender (string)

name	+++	age	+++	birthday	+++	gender
"John"	+++	"26"	+++	"26-01-1995"	+++	"male"
"Lisa"	+++	"xyz"	+++	"26-01-1995"	+++	"female"
null	+++	"26"	+++	"26-01-1995"	+++	"male"

Step 3

The system should take a input's from the user which is a list of :

- existing_col_name, new_col_name, new_data_type, date_expression (required only if the new_data_type is of DateType

On the basis of the above inputs, the system should convert existing columns in the dataframe to new columns with the new data type. If there are column values in a row which cannot be converted, we should replace it with null.

Example : 2nd row, 2nd column i.e. age = "xyz". This cannot be converted to integer. So it should be replaced with null

If an existing column in dataframe is not mentioned in the list, it should be omitted from the dataframe

Example : gender hasn't been mentioned. So it's removed from the dataframe generated.

The valid data_types supported for this task are string, integer, date, boolean. However please note - Ensure code modularity so that the system can handle new types like decimal, float, etc.

Arguments:

[

```
{
  "existing_col_name": "name", "new_col_name": "first_name", "new_data_type": "string"},
  {"existing_col_name": "age", "new_col_name": "total_years", "new_data_type": "integer"},
  {"existing_col_name": "birthday", "new_col_name": "d_o_b", "new_data_type": "date",
  "date_expression": "dd-MM-yyyy"}
}
```

Dataframe

Columns - first_name (string), total_years (integer), d_o_b (date)

first_name	+++	total_years	+++	d_o_b
"John"	+++	26	+++	26-01-1995
"Lisa"	+++	null	+++	26-01-1995
null	+++	26	+++	26-01-1995

Step 4

Provide profiling on all columns.

For the column provided, the system should give a count of the total number of unique values and count of each unique value. null values should be ignored.

Output :

```
[
  {
    "Column": "name",
    "Unique_values": 2,
    "Values": [
      { "John": 1 }, { "Lisa": 1 }
    ]
  },
  {
    "Column": "age",
    "Unique_values": 1,
    "Values": [
      { "26": 2 }
    ]
  },
  {
    "Column": "birthday",
    "Unique_values": 1,
    "Values": [
      { "26-01-1995": 3 }
    ]
  }
]
```

1
}
1