Práctica Big Data con Mongo DB y R

Jorge Colomer

2023-10-06

Contents

Cargamos los paquetes que vamos a usar.

```
library(tidyverse) # manipulación de data frames
library(mongolite) # para interactuar con MongoDB
library(ggeasy) # facilita la personalización de objetos ggplot
library(gt) # para generar tablas formateadas en el informe html
```

Tarea 1

*Exploremos otro diagrama de barras con una colección diferente - **`inspections`**. Esta recopilación contiene datos sobre las inspecciones de edificios de la ciudad de Nueva York y si pasan o no. Recupere dicha colección en R.*

Primeramente, procederemos a establecer la cadena y opciones de conexión a la base de datos en MongoDB.

```
cadena_conexion <- "mongodb+srv://user01:user01@cluster0.mcblc3z.mongodb.net/test"
opciones_conexion <- ssl_options(weak_cert_validation = FALSE)
```

Conectamos a MongoDB y recuperamos la colección `inspections` dentro de `sample_training`.

```
inspections <- mongo(
  collection = "inspections",
  db = "sample_training",
  url = cadena_conexion,
  options = opciones_conexion
  )
```

Tarea 2

Suponga que desea verificar el número de empresas que no aprobaron las inspecciones en 2015 en comparación con 2016.

Si ve los datos obtenidos de la colección, notará que el campo de fecha es una Cadena. Convertirlo en tipo de fecha y luego extraer el año requerirá algún procesamiento. Pero, con la canalización de agregación

de Mongodb, puede hacer todo en una sola consulta. Para manipular el campo de fecha, use el operador $addFields.

Además, agregue las deficiencias encontradas en las inspecciones por año.

En primer lugar, examinamos la estructura del primer documento para inferir la del resto.

```
num_docs <- inspections$count() # por si interesara conocer el número de registros
inspections$iterate()$one()
```

```
## $id
## [1] "10172-2015-CMPL"
##
## $certificate_number
## [1] 9304489
##
## $business_name
## [1] "UNNAMED HOT DOG VENDOR LICENSE NUMBER TA01158"
##
## $date
## [1] "Aug 21 2015"
##
## $result
## [1] "No Violation Issued"
##
## $sector
## [1] "Mobile Food Vendor - 881"
##
## $address
## $address$city
## [1] ""
##
## $address$zip
## [1] ""
##
## $address$street
## [1] ""
##
## $address$number
## [1] ""
```

La colección tiene 80047 registros.

Para conocer más en profundidad la base de datos de trabajo, podemos importar los 3 primeros registros (aunque la base de datos es relativamente pequeña y no habría problemas en cargarla en *R* en su totalidad) y aplicar la función `dplyr::glimpse`.

```
inspecciones <- inspections$find(limit = 3) %>%
  as_tibble()

glimpse(inspecciones)
```

```
## Rows: 3
## Columns: 7
## $ id                 <chr> "10172-2015-CMPL", "10712-2015-ENFO", "108-2015-UNI~
## $ certificate_number <int> 9304489, 3019428, 10003472
## $ business_name      <chr> "UNNAMED HOT DOG VENDOR LICENSE NUMBER TA01158", "T~
```

```
## $ date                <chr> "Aug 21 2015", "Feb 11 2015", "May 15 2015"
## $ result              <chr> "No Violation Issued", "Pass", "Pass"
## $ sector              <chr> "Mobile Food Vendor - 881", "Scale Dealer/Repairer ~
## $ address             <df[,4]> <data.frame[3 x 4]>
```

Vemos que el campo **address** es en realidad un data frame con 4 variables. Usamos la función **unnest** para "aplanar" esta columna.

```
inspecciones <- inspecciones %>%
  unnest(cols = "address") # para "aplanar" la columna address, ya que contiene un df anidado con 4 cam

glimpse(inspecciones)
```

```
## Rows: 3
## Columns: 10
## $ id                  <chr> "10172-2015-CMPL", "10712-2015-ENFO", "108-2015-UNI~
## $ certificate_number  <int> 9304489, 3019428, 10003472
## $ business_name       <chr> "UNNAMED HOT DOG VENDOR LICENSE NUMBER TA01158", "T~
## $ date                <chr> "Aug 21 2015", "Feb 11 2015", "May 15 2015"
## $ result              <chr> "No Violation Issued", "Pass", "Pass"
## $ sector              <chr> "Mobile Food Vendor - 881", "Scale Dealer/Repairer ~
## $ city                <chr> "", "DURHAM", ""
## $ zip                 <chr> "", "27709", ""
## $ street              <chr> "", "CORNWALLIS RD", ""
## $ number              <chr> "", "3039", ""
```

Es decir, el campo **result** es el que nos interesa para conocer el tipo y número de inspecciones. Veamos los valores que puede tomar esta variable.

```
# Encontrar los valores únicos en result
pipeline01 <- '[
  { "$group": { "_id": "$result" } },
  { "$sort": { "_id": 1 } }
]'

inspections$aggregate(pipeline01) %>%
  as_tibble() %>%
  rename("Type of inspection" = `_id`) %>%
  gt()
```

Type of inspection
Business Padlocked
Closed
Completed
Condemned
Confiscated
ECB Summons Issued
ECB Warning Issued
Fail
License Confiscated
Licensed
NOH Withdrawn
No Evidence of Activity
No Violation Issued
Out of Business
Pass

Posting Order Served
Re-inspection
Samples Obtained
Unable to Complete Inspection
Unable to Locate
Unable to Seize Vehicle
Violation Issued
Warning

El número de categorías en la variable `result` es de 23.[1]

Asumiendo que `Fail` indica que una empresa no aprueba la inspección, veamos el número total de `Fail` en la columna `result`:

```
inspections$count('{"result": "Fail"}')
```

```
## [1] 1100
```

Para conocer el número de empresas que no aprobaron las inspecciones en 2015 y 2016 diseñamos la siguiente consulta (desglose por año):

- `addFields`: Extraemos el año de la cadena `date` utilizando `substr` y lo almacenamos en un nuevo campo llamado `year`. *date* es el nombre del campo que contiene la fecha como una cadena y 7, 4 especifica que queremos extraer 4 caracteres comenzando desde el carácter 7 (considerando que los índices comienzan desde 0).
- `match`: Filtramos los documentos para considerar sólo aquellos que tienen `result` igual a `Fail`.
- `group`: Agrupamos los documentos por año y contamos el número de documentos en cada grupo utilizando `sum`.
- `match`: Filtramos los resultados para incluir solo los años 2015 y 2016.

```
# Definir el pipeline de agregación
pipeline02 <- '[
  { "$addFields": { "year": { "$substr": [ "$date", 7, 4 ] } } },
  { "$match": { "result": "Fail" } },
  { "$group": { "_id": "$year", "count": { "$sum": 1 } } },
  { "$match": { "_id": { "$in": ["2015", "2016"] } } }
]'

# Ejecutar la agregación
fail_year <- inspections$aggregate(pipeline02) %>%
  as_tibble() %>%
  rename("year" = `_id`,
         "# fails" = count)

fail_year %>%
  gt()
```

year	# fails
2015	1042
2016	58

[1] Podríamos haber obtenido un resultado similar utilizando la función `unique()` de *R* sobre la columna `result` del data frame completo.

Es decir, hay 1042 fallos en 2015 y 58 en 2016.

Para agregar las deficiencias encontradas por año, definimos, ejecutamos e imprimimos el siguiente pipeline:

```
# Definir el pipeline de agregación
pipeline03 <- '[
  { "$addFields": { "year": { "$substr": [ "$date", 7, 4 ] } } },
  { "$match": { "year": { "$in": ["2015", "2016"] } } },
  { "$group": { "_id": { "year": "$year", "result": "$result" }, "count": { "$sum": 1 } } },
  { "$sort": { "_id.year": 1, "_id.result": 1 } }
]'

# Ejecutar y mostrar la agregación
inspections$aggregate(pipeline03) %>%
  as_tibble() %>%
  unnest(cols = `_id`) %>% # para "aplanar" el df de la columna _id
  gt()
```

year	result	count
2015	Business Padlocked	1
2015	Closed	860
2015	Completed	14
2015	Condemned	2
2015	Confiscated	4
2015	ECB Summons Issued	18
2015	ECB Warning Issued	196
2015	Fail	1042
2015	License Confiscated	14
2015	Licensed	139
2015	NOH Withdrawn	101
2015	No Evidence of Activity	2164
2015	No Violation Issued	35359
2015	Out of Business	6673
2015	Pass	14405
2015	Posting Order Served	305
2015	Re-inspection	115
2015	Samples Obtained	44
2015	Unable to Complete Inspection	39
2015	Unable to Locate	196
2015	Unable to Seize Vehicle	1
2015	Violation Issued	12773
2015	Warning	930
2016	Closed	41
2016	Completed	6
2016	Fail	58
2016	Licensed	2
2016	NOH Withdrawn	2
2016	No Evidence of Activity	103
2016	No Violation Issued	1951
2016	Out of Business	365
2016	Pass	1104
2016	Posting Order Served	19
2016	Re-inspection	62
2016	Samples Obtained	13

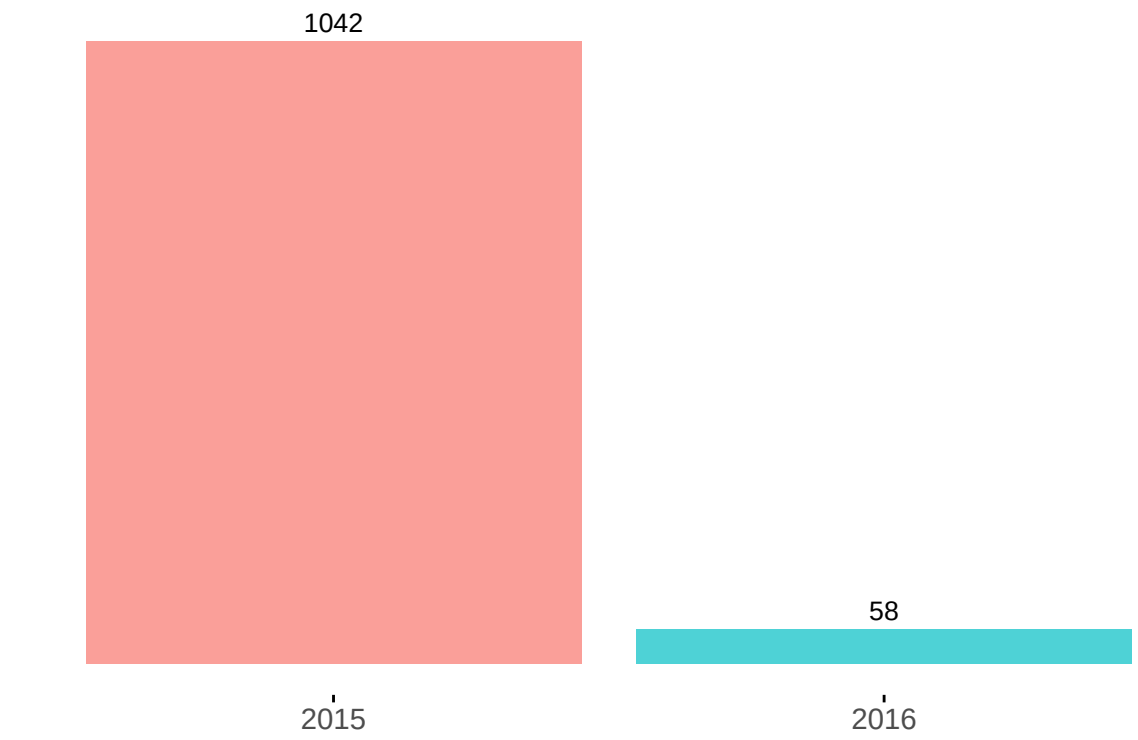
2016	Unable to Complete Inspection	2
2016	Unable to Locate	17
2016	Violation Issued	865
2016	Warning	42

Tarea 3

Teniendo en cuenta que el resultado de la tarea anterior está agrupando los resultados por año, cree un gráfico de barras.

Aquí podemos usar el paquete `ggplot` para realizar el gráfico solicitado, utilizando el tema `theme_minimal` y la librería `ggeasy`, la cual facilita la personalización de un objeto `ggplot`.

```
fail_year %>%
ggplot(aes(x = year,
           y = `# fails`,
           fill = year)
       ) +
 geom_col(alpha = 0.7) +
 theme_minimal() +
  easy_plot_title_size(size = 18) +
  easy_plot_subtitle_size(size = 14) +
 easy_remove_legend() +
 easy_remove_axes(which = "both", what = "title") +
 easy_remove_y_axis(what = "text") +
 easy_remove_gridlines() +
 geom_text(aes(label = `# fails`),
           color = "black",
           vjust = -0.5,
           size = 3.4
           ) +
 easy_x_axis_labels_size(size = 11) +
 easy_change_text(which = "plot.title",
                  what = "face",
                  to = "bold"
                  ) +
 theme(axis.ticks.length = unit(0.10, "cm"),
        axis.ticks.x = element_line(colour = "black"),
        axis.ticks.y = element_blank()
       ) +
  easy_labs(title = "Número de fallos por año",
            subtitle = "Base de datos: inspections"
            )
```



Tarea 4

A continuación, se utilizará la colección 'companies', que contiene información sobre empresas, como su año de fundación y la dirección de su sede.

Supongamos que desea conocer la tendencia del número de empresas de publicidad (category_code = 'advertising') fundadas a partir de 2000 en adelante. Para ello, utilice el operador relacional $gt, agrupe los resultados por año de creación ('founded_year') y ordénelos para que se muestren posterioremente en un gráfico de líneas por año.

En primer lugar, generemos el puntero a la base de datos y exploremos el primer registro con `dplyr::glimpse`:

```
companies <- mongo(collection = "companies",
                   db = "sample_training",
                   url = cadena_conexion,
                   options = opciones_conexion
                  )

glimpse(companies$find(limit = 1))
```

```
## Rows: 1
## Columns: 41
## $ name                <chr> "Facebook"
## $ permalink           <chr> "facebook"
## $ crunchbase_url      <chr> "http://www.crunchbase.com/company/facebook"
## $ homepage_url        <chr> "http://facebook.com"
## $ blog_url            <chr> "http://blog.facebook.com"
## $ blog_feed_url       <chr> "http://blog.facebook.com/atom.php"
```

```
## $ twitter_username    <chr> "facebook"
## $ category_code       <chr> "social"
## $ number_of_employees <int> 5299
## $ founded_year        <int> 2004
## $ founded_month       <int> 2
## $ founded_day         <int> 1
## $ deadpooled_year     <lgl> NA
## $ deadpooled_month    <lgl> NA
## $ deadpooled_day      <lgl> NA
## $ deadpooled_url      <chr> ""
## $ tag_list            <chr> "facebook, college, students, profiles, network, o~
## $ alias_list          <chr> ""
## $ email_address       <chr> ""
## $ phone_number        <chr> ""
## $ description         <chr> "Social network"
## $ created_at          <chr> "Fri May 25 21:22:15 UTC 2007"
## $ updated_at          <chr> "Thu Nov 21 19:40:55 UTC 2013"
## $ overview            <chr> "<p>Facebook is the world&#8217;s largest social n~
## $ image               <df[,2]> <data.frame[1 x 2]>
## $ products            <list> [<data.frame[8 x 2]>]
## $ relationships       <list> [<data.frame[273 x 3]>]
## $ competitions        <list> [<data.frame[14 x 1]>]
## $ providerships       <list> [<data.frame[10 x 3]>]
## $ total_money_raised  <chr> "$2.43B"
## $ funding_rounds      <list> [<data.frame[11 x 10]>]
## $ investments         <list> [<data.frame[3 x 1]>]
## $ acquisition         <lgl> NA
## $ acquisitions        <list> [<data.frame[41 x 9]>]
## $ offices             <list> [<data.frame[3 x 9]>]
## $ milestones          <list> [<data.frame[26 x 13]>]
## $ ipo                 <df[,6]> <data.frame[1 x 6]>
## $ video_embeds        <list> []
## $ screenshots         <list> [<data.frame[1 x 2]>]
## $ external_links      <list> [<data.frame[4 x 2]>]
## $ partners            <list> []
```

Ahora generemos la consulta requerida:

```
# Pipeline de agregación
pipeline04 <- '[
  { "$match": { "category_code": "advertising", "founded_year": { "$gt": 1999 } } },
  { "$group": { "_id": "$founded_year", "count": { "$sum": 1 } } },
  { "$sort": { "_id": 1 } }
]'

# Ejecutar la agregación
companies_year <- companies$aggregate(pipeline04) %>%
  as_tibble() %>%
  rename(year = `_id`)

companies_year %>%
  gt()
```

year	count

2000	16
2001	13
2002	15
2003	17
2004	24
2005	32
2006	49
2007	80
2008	77
2009	27
2012	1
2013	1

Ya podemos dibujar el gráfico de líneas solicitado.

```
companies_year %>%
  mutate(year = make_date(year)) %>%
  ggplot(aes(year, count)) +
  geom_line(linewidth = 1, color = "orange") +
  geom_point(color = "orange",
             size = 3) +
  theme_light() +
  easy_labs(title = "Número de empresas fundadas por año",
            subtitle = "Base de datos: companies"
            ) +
  easy_plot_title_size(size = 18) +
  easy_plot_subtitle_size(size = 14) +
  easy_remove_legend() +
  easy_remove_gridlines() +
  easy_remove_axes(which = "both",
                   what = "title"
                   ) +
  easy_remove_y_axis(what = c("ticks",
                              "text"
                              )
                     ) +
  easy_x_axis_labels_size(size = 10) +
  easy_change_text(which = "plot.title",
                   what = "face",
                   to = "bold"
                   ) +
  scale_x_date(date_breaks = "1 year",
               date_labels = "%Y"
               ) +
  theme(axis.ticks.length = unit(0.10, "cm"),
        axis.ticks = element_line(color = "black")
        ) +
  geom_text(aes(label = count),
            color = "black",
            vjust = -0.5,
            hjust = 1.5,
            size = 3.4
            )
```

Número de empresas fundadas por año

Base de datos: companies