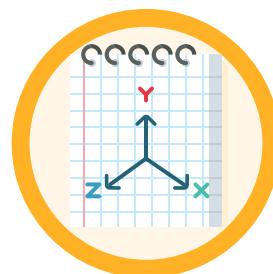


Agregaciones en R



A continuación, se desarrollará un ejemplo de agregaciones y estadísticas descriptivas sin agregar, basado en el análisis de la Gran Encuesta de Hogares.

Para comenzar, lea en R la tabla *cabecera - ocupados.sav* correspondiente a la Gran Encuesta de Hogares. Ubique el archivo en una carpeta y configure el directorio de trabajo:



```
> setwd("D:/Gdrive/Laboral 2016/DANE_INNOVACION/Curso análisis de datos con R/cursor_2016II/Curso/Bases/GEIH/Diciembre")  
> library(haven)  
> cabec_ocup <- read_spss("Cabecera - Ocupados.sav")
```



DANE
Para tomar decisiones

TODOS POR UN
NUEVO PAÍS
PAZ EQUIDAD EDUCACIÓN



Procesamiento y análisis de datos con R

Agregaciones en R

En primer lugar, calcularemos el promedio de los ingresos laborales en el país:



```
> mean(cabec_ocup$Inglabo)  
[1] NA
```

Al existir varios valores faltantes, se debe especificar la opción **na.rm = T**. Como puede observar, hay 3181 faltantes, correspondientes al 11.97%:

```
table(is.na(cabec_ocup$Inglabo))  
  
FALSE TRUE  
23392 3181
```



Utilice el argumento **na.rm = T**, para que se omitan los valores faltantes:

```
mean(cabec_ocup$Inglabo, na.rm = T)  
[1] 948304.7
```

Ahora, se averiguará qué departamento tiene los mayores ingresos laborales en el país. Como hay datos faltantes, al calcular el promedio de ingreso por departamento se obtiene lo siguiente:

```
tapply(cabec_ocup$Inglabo, cabec_ocup$Dpto, FUN = mean)  
05 08 11 13 15 17 18 19 20 23 25 27 41 44 47 50  
52 54 63 66 68 70 73 76  
NA  
NA NA NA NA NA NA NA NA NA NA NA NA NA NA
```

Para poder calcular los promedios omitiendo los datos faltantes, se debe especificar al lado de **mean** la función **na.rm = T**.

```
tapply(cabec_ocup$Inglabo, cabec_ocup$Dpto, FUN = mean, na.rm = T)  
05 08 11 13 15  
1074056.7 794987.2 1297056.7 777045.8 1067789.6  
17 18  
1061845.5 967424.5
```



DANE
Para tomar decisiones

TODOS POR UN
NUEVO PAÍS
PAZ EQUIDAD EDUCACIÓN



Procesamiento y análisis de datos con R

Agregaciones en R

19	20	23	25	27
925361.0	935174.4	826556.7	780000.6	1000083.7
41	44	47	50	52
946041.7	967786.0	806990.9	944762.0	927794.0
54	63	66	68	70
794671.8	922606.3	848579.2	1103797.4	822505.1
73	76			
960987.0	916139.9			



En la segunda posición ahora se toma el valor de 100. Con la función **as_factor** se traen las etiquetas que están en SPSS:

```
cabec_ocup$Dpto <- as_factor(cabec_ocup$Dpto)
> tapply(cabec_ocup$Inglabo, cabec_ocup$Dpto, FUN = mean, na.rm = T)
```

Antioquia	Atlántico	Bogotá, d.c.	Bolívar
1074056.7	794987.2	1297056.7	794987.2
Boyacá	Caldas	Caquetá	Caldas
1067789.6	1061845.5	967424.5	1061845.5
César	Córdoba	Cundinamarca	Córdoba
935174.4	826556.7	780000.6	826556.7

Huila	La guajira	Magdalena	Meta
946041.7	967786.0	806990.9	944762.0
Nariño	Norte de Santander	Quindío	Risaralda
927794.0	794671.8	922606.3	848579.2
Santander	Sucre	Tolima	Valle del Cauca
1103797.4	822505.1	960987.0	916139.9

Si se presentan problemas de etiquetas, se puede convertir el factor a carácter y cambiar el tipo de codificación a "latin1". De esta forma se logran resolver los inconvenientes con las tildes, las ñ, y demás símbolos propios del español.

```
cabec_ocup$Dpto <- as.character(cabec_ocup$Dpto)
> Encoding(cabec_ocup$Dpto) <- "latin1"
> tapply(cabec_ocup$Inglabo, cabec_ocup$Dpto, FUN = mean, na.rm = T)
```

Antioquia	Atlántico	Bogotá, d.c.	Bolívar
1074056.7	794987.2	1297056.7	794987.2
Boyacá	Caldas	Caquetá	Cauca
1067789.6	1061845.5	967424.5	925361.0
César	Chocó	Córdoba	Cundinamarca
935174.4	1000083.7	826556.7	780000.6





Procesamiento y análisis de datos con R

Agregaciones en R

Huila	La guajira	Magdalena	Meta
946041.7	967786.0	806990.9	944762.0
Nariño	Norte de santander	Quindío	Risaralda
927794.0	794671.8	922606.3	848579.2
Santander	Sucre	Tolima	Valle del cauca
1103797.4		960987.0	916139.9
		822505.1	

Se puede observar que Bogotá y Antioquia presentan los mayores promedios de ingresos laborales.

El resultado al utilizar la función **tapply** es un vector, para que este sea un *Dataframe* se puede utilizar la función **aggregate**:

```
> aggregate(Inglabo ~ Dpto, data = cabec_ocup,  
FUN = mean, na.rm = T)
```

Dpto	Inglabo
1 Antioquia	1074056.7
2 Atlántico	794987.2
3 Bogotá, d.c.	1297056.7
4 Bolívar	777045.8
5 Boyacá	1067789.6
6 Caldas	1061845.5

7 Caquetá	1074056.7
8 Cauca	794987.2
9 César	1297056.7
10 Chocó	777045.8
11 Córdoba	1067789.6
12 Cundinamarca	1061845.5
13 Huila	946041.7
14 La guajira	967786.0
15 Magdalena	806990.9
16 Meta	944762.0
17 Nariño	927794.0
18 Norte de santander	794671.8
19 Quindío	922606.3
20 Risaralda	848579.2
21 Santander	1103797.4
22 Sucre	822505.1
23 Tolima	960987.0
24 Valle del cauca	916139.9

Es posible utilizar cualquier función que arroje un valor sobre las funciones de agregación.





Procesamiento y análisis de datos con R

Agregaciones en R

Si se requiere por ejemplo calcular una agregación de una función que no existe en R, puede construirse previamente la función. Si esta se encuentra en un paquete, se puede invocar este.

En el siguiente ejemplo se calcula el promedio, la desviación estándar, el coeficiente de variación, la asimetría y la curtosis del ingreso por departamento.



```
datos <- select(cabec_ocup, Dpto, Inglabo)
datos <- datos[!is.na(datos$Inglabo), ]

cv <- function(x){
  100 * sd(x) / mean(x)
}

library(moments) #Para asimetría (skewness) y curtosis (kurtosis)
summarize(group_by(datos, Dpto), prom_inglaboral = mean(Inglabo),
de_inglaboral = sd(Inglabo), cv_inglaboral = cv(Inglabo),
asimetria_inglaboral = skewness(Inglabo),
curtosis_inglaboral = kurtosis(Inglabo))
```

A tibble: 24 x 6

Dpto	prom_inglaboral	cv_inglaboral	cv_inglaboral	asimetria_inglaboral	curtosis_inglaboral
<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>





Procesamiento y análisis de datos con R

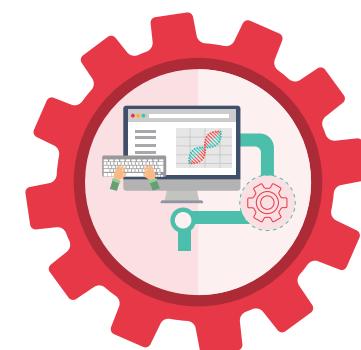
Agregaciones en R

1	Antioquia	1074056.7	1529405.8	142.39526	11.829214	214.01035
2	Atlántico	794987.2	920327.1	115.76628	7.984431	121.36489
3	Bogotá, d.C.	1297056.7	1930271.2	148.81934	8.736833	127.86595
4	Bolívar	777045.8	740703.0	95.32295	3.577556	22.66840
5	Boyacá	1067789.6	1039431.8	97.34425	2.672144	14.83881
6	Caldas	1061845.5	1431883.4	134.84856	6.972453	70.38464
7	Caquetá	967424.5	1384731.6	143.13588	8.493091	118.51507
8	Cauca	925361.0	991314.0	107.12727	3.071810	17.71751
9	César	935174.4	1106214.9	118.28969	8.511327	123.17480
10	Chocó	1000083.7	107.85833	2.782161	2.782161	13.37217

... with 14 more rows

La manera más fácil de realizar agregaciones es con el paquete de **Hadley Wickham dplyr**. Si se quiere sacar una tabla con las medidas resumen se puede utilizar la función **summarise** de ese paquete.

Con la función **summarise** se calculan las estadísticas resumen. En el primer argumento se especifica el Dataframe con el cual se quiere trabajar:





Procesamiento y análisis de datos con R

Agregaciones en R

```
library(dplyr)
summarise(cabec_ocup, prom_inglaboral = mean(Ingla-
    bo))
# A tibble: 1 x 1
  prom_inglaboral
  <dbl>
1 NA
summarise(cabec_ocup, prom_inglaboral = mean(Ingla-
    bo, na.rm = T))
# A tibble: 1 x 1
  prom_inglaboral
  <dbl>
1 948304.7
```

Para sacar la información agregada a nivel departamento, basta con utilizar previamente la función **group_by**. En el primer argumento se indica la tabla con la cual se requiere realizar el agrupamiento y en el segundo se señala que la columna con la cual se realiza es Dpto.

Posteriormente se utiliza la función **summarise**. En el primer argumento como ya se mencionó se debe colocar una tabla o el resultado de la función **group_by**:

```
por_dpto <- group_by(cabec_ocup, Dpto)
summarise(por_dpto, prom_inglaboral = mean(Ingla-
    bo, na.rm = T))
# A tibble: 24 x 2
```

Dpto	prom_inglaboral
<chr>	<dbl>
1 Antioquia	1074056.7
2 Atlántico	794987.2
3 Bogotá, d.c.	1297056.7
4 Bolívar	777045.8
5 Boyacá	1067789.6
6 Caldas	1061845.5
7 Caquetá	967424.5
8 Cauca	925361.0
9 César	935174.4
10 Chocó	1000083.7
# ... with 14 more rows	

La anterior consulta se puede asignar a resumen:



Procesamiento y análisis de datos con R

Agregaciones en R

```
resumen <- summarise(por_dpto, prom_inglaboral =  
mean(IngLabo, na.rm = T))  
resumen  
# A tibble: 24 x 2  
  Dpto      prom_inglaboral  
  <chr>        <dbl>  
1 Antioquia    1074056.7  
2 Atlántico     794987.2  
3 Bogotá, d.C. 1297056.7  
4 Bolívar       777045.8  
5 Boyacá        1067789.6  
6 Caldas         1061845.5  
7 Caquetá       967424.5  
8 Cauca          925361.0  
9 César          935174.4  
10 Chocó         1000083.7  
# ... with 14 more rows
```

En una sola consulta pueden calcularse el promedio, el mínimo, el percentil 25, la mediana, el percentil 75 y el máximo:

```
resumen2 <- summarise(por_dpto, num_colegios =  
n(),  
  prom_inglaboral = mean(IngLabo, na.rm =  
T),  
  min_inglaboral = min(IngLabo, na.rm =  
T),  
  p25_inglaboral = quantile(IngLabo,  
0.25, na.rm = T),  
  mediana_inglaboral = median(IngLabo,  
na.rm = T),  
  p75_inglaboral = quantile(IngLabo,  
0.75, na.rm = T),  
  max_inglaboral = max(IngLabo, na.rm =  
T))  
  
resumen2  
# A tibble: 24 x 8
```



DANE
Para tomar decisiones

TODOS POR UN
PAZ EQUIDAD EDUCACIÓN
NUEVO PAÍS



Procesamiento y análisis de datos con R

Agregaciones en R

Dpto	num_colegios	prom_inglaboral	min_inglaboral	p25_inglaboral	mediana_inglaboral	p75_inglaboral
<chr>	<int>	<dbl>	<dbl>	<dbl>	glaboral <dbl>	<dbl>
1 Antioquia	1726	1074056.7	0	640000	800000	1200000
2 Atlántico	1564	794987.2	0	360000	644336	900000
3 Bogotá, d.C.	1509	1297056.7	0	644350	800000	1300000
4 Bolívar	1244	777045.8	0	400000	640000	900000
5 Boyacá	801	1067789.6	0	500000	700000	1300000
6 Caldas	1072	1061845.5	0	600000	700000	1100000
7 Caquetá	914	967424.5	0	400000	644350	1015000
8 Cauca	1046	925361.0	0	400000	644350	1015000
9 César	974	935174.4	0	530000	700000	985000
10 Chocó	789	1000083.7	0	400000	650000	1120000

... with 14 more rows, and 1 more variables: max_inglaboral <dbl>

Observe que los departamentos con mayores ingresos laborales promedio son Bogotá, Santander, Antioquia, Boyacá y Caldas.





Procesamiento y análisis de datos con R

Agregaciones en R

Agregaciones utilizando el lenguaje sql

Una alternativa para realizar agregaciones es mediante **sql** (lenguaje estructurado de consultas) el cual sirve para la realización de consultas en bases de datos. sql se encuentra incorporado en diferentes programas como Access, Oracle, Mysql, Posgtres y SAS. R dispone de diferentes alternativas para manejar este lenguaje.

La estructura para manejar el lenguaje sql es la siguiente:

- **select**: permite calcular estadísticas resúmenes, promedios, conteos, mínimo, máximo, etc. Funciona igual que el summarize del dplyr.
- **from**: tabla de datos.
- **where**: filtros, lo mismo que el subset o el filter del paquete dplyr.
- **group by**: variables categóricas de agrupamiento. Tiene la misma función que el group_by del paquete dplyr.

Advertencia: debe evitarse colocar punto (.) en los nombres de las variables del Dataframe a trabajar pues este símbolo en sql tiene una función muy similar al signo pesos de R (\$).

Se calcula a continuación el total de evaluados por departamento en las pruebas saber:



```
library(sqldf)
library(haven)
saber_2015II<-read_spss("saber11_2015II.sav")
cons1 <- sqldf("select sum(EVALUADOS) as 'total_eval' from saber_2015II")
cons1
total_eval
1      546655
```

Note que en sql la consulta se realiza dentro sql(" ") .

Observe a continuación que para generar la columna total de evaluados (total_eval), se usa la función **sum**. Después del as se escribe el nombre se le quiere asignar en la tabla generada. Con **from** se especifica el Dataframe a utilizar.

Algunas funciones útiles en sql son :

- avg(): promedio
- count(): conteo
- max(): máximo
- min(): mínimo
- sum(): suma
- avg: promedio
- variance(): varianza
- #sqrt(variance()): desviación estándar





Procesamiento y análisis de datos con R

Agregaciones en R

Agregaciones utilizando el lenguaje sql

Una alternativa para realizar agregaciones es mediante **sql** (lenguaje estructurado de consultas) el cual sirve para la realización de consultas en bases de datos. sql se encuentra incorporado en diferentes programas como Access, Oracle, Mysql, Posgtres y SAS. R dispone de diferentes alternativas para manejar este lenguaje.

La estructura para manejar el lenguaje sql es la siguiente:

- **select:** permite calcular estadísticas resúmenes, promedios, conteos, mínimo, máximo, etc. Funciona igual que el summarize del dplyr.
- **from:** tabla de datos.
- **where:** filtros, lo mismo que el subset o el filter del paquete dplyr.
- **group by:** variables categóricas de agrupamiento. Tiene la misma función que el group_by del paquete dplyr.

Advertencia: debe evitarse colocar punto (.) en los nombres de las variables del Dataframe a trabajar pues este símbolo en sql tiene una función muy similar al signo pesos de R (\$).

Se calcula a continuación el total de evaluados por departamento en las pruebas saber:



```
library(sqlite)
library(haven)
saber_2015II<-read_spss("saber11_2015II.sav")
cons1 <- sqldf("select sum(EVALUADOS) as 'total_eval' from saber_2015II")
cons1
total_eval
1      546655
```

Note que en sql la consulta se realiza dentro sql(" ") .

Observe a continuación que para generar la columna total de evaluados (total_eval), se usa la función **sum**. Después del as se escribe el nombre se le quiere asignar en la tabla generada. Con **from** se especifica el Dataframe a utilizar.

Algunas funciones útiles en sql son :

- avg(): promedio
- count(): conteo
- max(): máximo
- min(): mínimo
- sum(): suma
- avg: promedio
- variance(): varianza
- #sqrt(variance()): desviación estándar





Procesamiento y análisis de datos con R

Agregaciones en R

Es posible realizar varias agregaciones que involucren diferentes estadísticas. En el siguiente ejemplo se calcula el conteo, el promedio de evaluados y el puntaje, así como la mediana y la desviación estándar, del puntaje de los colegios por departamento.

```
cons6 <- sqldf("select DEPARTAMENTO, count(*) as  
'num_colegios', avg(EVALUADOS) as 'prom_mat', me-  
dian(EVALUADOS) as 'mediana_eval',  
stdev(PROMMATEMATICA) as 'de_eval' from  
saber_2015II group by DEPARTAMENTO")
```

```
head(cons6)
```

	DEPARTAMENTO	num_cole- gios	prom_eval	mediana_ mat	de_eval
1	AMAZONAS	19	44.31579	28	6.624333
2	ANTIOQUIA	1411	52.74061	37	7.423588
3	ARAUCA	71	41.30986	33	6.686052
4	ATLANTICO	589	49.96944	37	7.397740
5	BOGOTÁ	1459	62.12886	50	7.200925
6	BOLIVAR	559	44.17889	34	7.682582



Usando sql es posible realizar filtros con el argumento **where** (que haría las veces de subset o filter). Por ejemplo, para realizar un análisis del número de evaluados en los colegios oficiales, se puede especificar en where que solo tenga en cuenta los colegios de naturaleza oficial.

```
saber_2015II$NATURALEZA<-as_factor(saber_2015II$-  
NATURALEZA)  
cons7 <- sqldf("select DEPARTAMENTO, count(*) as  
'num_colegios', avg(EVALUADOS) as 'prom_eval',  
median(EVALUADOS) as 'mediana_eval', stdev(EVALU-  
ADOS) as 'de_eval'  
from saber_2015II where NATURALEZA = 'Oficial'  
group by DEPARTAMENTO")
```

```
head(cons7)
```

	DEPARTAMENTO	num_cole- gios	prom_eval	mediana_ mat	de_eval
1	AMAZONAS	17	47.88235	31.0	38.05076
2	ANTIOQUIA	1018	52.37623	40.0	56.55249
3	ARAUCA	60	45.35000	35.5	34.27497
4	ATLANTICO	358	58.29050	48.0	41.05744
5	BOGOTÁ	638	77.11599	70.0	44.38453
6	BOLIVAR	433	45.69515	37.0	34.33056

