BashQL
Manual de Usuario

Esteban Rodríguez Betancourt - B15512
Marie Barquero Rojas - B00799

Lunes 8 de diciembre del 2014

Resumen

BashQL es un conjunto de herramientas que permiten hacer consultas sobre archivos CSV. Cada herramienta está diseñada para que tenga una sola función específica, y al usarlas en conjunto se pueden lograr consultas más complejas.

Las herramientas están programadas en Google Go, y pueden ser utilizadas en cualquier sistema tipo UNIX, como Linux, BSD o OS X.

1. Instalación

La forma más sencilla de instalar BashQL es utilizando el gestor de paquetes de Google Go. Esto además permite recompilar las herramientas para que incorporen futuras mejoras en el compilador de Go. También es posible copiar los ejecutables precompilados para la plataforma correcta: BashQL no tiene dependencias dinámicas aparte de las bibliotecas específicas del sistema.

1.1. Instalación de Google Go

Las siguientes instrucciones son para Go 1.3.3. Es posible que las instrucciones cambien para futuras versiones. Se recomienda revisar el sitio web http://golang.org/ para tener la información más actualizada.

1.1.1. Instalación precompilada

1. Descargar la versión correspondiente a su sistema operativo y arquitectura desde https://golang.org/dl/.
2. Descomprimir el archivo (puede ser en $HOME).
3. Crear una carpeta aparte para los paquetes de Go (ejm: $HOME/gopacks).
4. Modificar las variables de entorno para agregar las herramientas de Go:

a) Agregar en ~/.profile o ~/.bashrc las siguientes líneas, al final. Modificar según sea el caso.

```
export GOROOT=$HOME/go
export GOPACKS=$HOME/gopacks
export PATH=$PATH:$GOROOT/bin:$GOPACKS/bin
```

b) Cargar las variables de entorno en la sesión actual:

```
$ . ~/.profile
```

5. En este punto al correr el comando «go» se deberían desplegar las instrucciones para correr el programa.

1.1.2. Instalación desde código fuente

1. Asegurarse de tener instalado un compilador de C (se recomienda gcc) y mercurial.
2. Clonar el respostorio:

```
$ hg clone -u release https://code.google.com/p/go
```

3. Construir la distribución de Go:

```
$ cd go/src
$ ./all.bash
```

4. Seguir las instrucciones para actualizar las variables de entorno, tal como se indica en los pasos del 3 al 5 de Instalación precompilada.

1.2. Instalación de BashQL

Para descargar BashQL es necesario tener GIT y Go instalados. Luego solamente hay que correr los comandos:

```
$ go get github.com/estebarb/bashql/...
$ go install github.com/estebarb/bashql/...
```

Con esto se instalan las herramientas que forman parte de BashQL.

2. Uso de BashQL

BashQL está formado por varios comandos, que cumplen con funciones muy específicas. Prácticamente todos los comandos utilizan la entrada y la salida estándar, como método de comunicación entre las herramientas, por lo que es muy sencillo crear nuevos componentes interoperables con BashQL.

2.1. bqlfrom

Nombre bqlfrom - Lee un archivo CSV, lo normaliza y lo escribe en stdout.

Sinopsis bqlfrom [opciones] archivo

Descripción Lee un archivo CSV según el formato especificado en las opciones (o bien el formato estandar), lo transforma al formato estandar y lo escribe en stdout.

-d Indica el caracter que delimita las columnas del archivo CSV. Ejm: «-d ';'» o «-d=';'». Por defecto se utiliza una coma («,») como separador.

-c Indica el caracter de inicio de un comentario. Las líneas comentadas son ignoradas por el lector de CSV. Ejm: «-c '%'» o «-c='%'». Por defecto los CSV no tienen comentarios.

Ejemplo:

```
$ cat personal.csv
% Personal de la tienda
id;nombre;edad;puesto
1;Antonio;25;Cajas
2;María;26;Cajas
$ bqlfrom -d ';' -c='%' personal.csv
id,nombre,edad,puesto
1,Antonio,25,Cajas
2,María,26,Cajas
```

2.2. bqlselect

Nombre bqlselect - Selecciona columnas específicas de un archivo CSV.

Sinopsis bqlselect columnas...

Descripción Recibe por stdin un archivo CSV normalizado y devuelve por stdout los datos de las columnas seleccionadas.

Ejemplo:

```
$ bqlfrom -d ';' -c='%' personal.csv \
| bqlselect nombre puesto
nombre,puesto
Antonio,Cajas
María,Cajas
```

2.3. bqlwhere

Nombre bqlwhere - Selecciona las filas que cumplen con los criterios dados.

Sinopsis bqlwhere {columna operador argumento}...

Descripción Recibe por stdin un archivo CSV normalizado y devuelve por stdout las filas que cumplen con TODOS los criterios solicitados. Los argumentos van en trios, donde se indica la columna, el operador y el argumento del operador (valor o columna).

Operadores: El comando acepta los siguientes operadores:

- **=** Igualdad entre el valor de la columna y el valor pasado como argumento (número o cadena).
- **<** El valor de la columna es menor que el valor pasado como argumento (número o cadena). El texto se compara por orden alfabético.
- **>** El valor de la columna es mayor que el valor pasado como argumento (número o cadena). El texto se compara por orden alfabético.
- **!=** Desigualdad entre el valor de la columna y el valor pasado como argumento (número o cadena).
- **<=** El valor de la columna es menor o igual que el valor pasado como argumento (número o cadena). El texto se compara por orden alfabético.
- **>=** El valor de la columna es mayor o igual que el valor pasado como argumento (número o cadena). El texto se compara por orden alfabético.
- **c=** Igualdad entre el valor de la columna y el valor en la columna pasada como argumento (número o cadena).
- **c<** El valor de la columna es menor que el valor en la columna pasada como argumento (número o cadena). El texto se compara por orden alfabético.
- **c>** El valor de la columna es mayor que el valor en la columna pasada como argumento (número o cadena). El texto se compara por orden alfabético.
- **c!=** Desigualdad entre el valor de la columna y el valor en la columna pasada como argumento (número o cadena).
- **c<=** El valor de la columna es menor o igual que el valor en la columna pasada como argumento (número o cadena). El texto se compara por orden alfabético.
- **c>=** El valor de la columna es mayor o igual que el valor en la columna pasada como argumento (número o cadena). El texto se compara por orden alfabético.

like El texto en la columna hace match contra una expresión regular pasada como argumento.

unlike El texto en la columna NO hace match contra una expresión regular pasada como argumento.

Ejemplo:

```
$ bqlfrom poblacion.csv \
| bqlwhere Tipo = 'Provincia' \
Total '<' 500000 \
Nombre '<=' 'Limón'
Nombre,Total,Hombre,Mujer,Tipo
Cartago,491425,241121,250304,Provincia
Heredia,433975,211417,222558,Provincia
Limón,386954,193673,193281,Provincia
Guanacaste,326821,161932,164889,Provincia
```

2.4. bqlwhenever

Nombre bqlwhenever - Selecciona las filas que cumplen con alguno de los criterios dado.

Sinopsis bqlwhenever {columna operador argumento}...

Descripción Recibe por stdin un archivo CSV normalizado y devuelve por stdout las filas que cumplen con AL MENOS UNO de los criterios solicitados. Los argumentos van en trios, donde se indica la columna, el operador y el argumento del operador (valor o columna).

Operadores: El comando acepta los siguientes operadores:

= Igualdad entre el valor de la columna y el valor pasado como argumento (número o cadena).

< El valor de la columna es menor que el valor pasado como argumento (número o cadena). El texto se compara por orden alfabético.

> El valor de la columna es mayor que el valor pasado como argumento (número o cadena). El texto se compara por orden alfabético.

!= Desigualdad entre el valor de la columna y el valor pasado como argumento (número o cadena).

<= El valor de la columna es menor o igual que el valor pasado como argumento (número o cadena). El texto se compara por orden alfabético.

>= El valor de la columna es mayor o igual que el valor pasado como argumento (número o cadena). El texto se compara por orden alfabético.

c= Igualdad entre el valor de la columna y el valor en la columna pasada como argumento (número o cadena).

c< El valor de la columna es menor que el valor en la columna pasada como argumento (número o cadena). El texto se compara por orden alfabético.

c> El valor de la columna es mayor que el valor en la columna pasada como argumento (número o cadena). El texto se compara por orden alfabético.

c!= Desigualdad entre el valor de la columna y el valor en la columna pasada como argumento (número o cadena).

c<= El valor de la columna es menor o igual que el valor en la columna pasada como argumento (número o cadena). El texto se compara por orden alfabético.

c>= El valor de la columna es mayor o igual que el valor en la columna pasada como argumento (número o cadena). El texto se compara por orden alfabético.

like El texto en la columna hace match contra una expresión regular pasada como argumento.

unlike El texto en la columna NO hace match contra una expresión regular pasada como argumento.

Ejemplo:

```
$ bqlfrom poblacion.csv \
| bqlwhere Tipo = Provincia \
| bqlwhenever Nombre '<' 'Cartago' Nombre '>' 'Cartago' \
| bqlselect Nombre
Nombre
San José
Alajuela
Heredia
Puntarenas
Limón
Guanacaste
```

2.5. bqljoin

Nombre bqljoin - Toma dos archivos CSV y hace un join entre ellos.

Sinopsis bqljoin [opciones] csv1 columna1 csv2 columna2
bqljoin [opciones] columnaSTDIN csv2 columna2

Descripción Lee dos archivos CSV y hace un join entre ellos según la columna especificada por archivo. Por defecto se hace un join interno, pero con los parámetros se puede modificar para que realice un left join, right join, inner join o outer join / full join. Escribe la salida en stdout.

-d1 Indica el delimitador del primer archivo. Ejm: «-d1=';'». Por defecto es una coma.

-d2 Indica el delimitador del segundo archivo. Ejm: «-d2=';'». Por defecto es una coma.

-d Indica el delimitador de la entrada estandar. Ejm: «-d=';'». Por defecto es una coma.

-type Indica el tipo de join que se realizará. El tipo puede ser «inner», «left», «right», «outer» o «full».

Ejemplo:

```
$ cat english_numbers.csv
number,value
zero,0
one,1
two,2
three,3
four,4
five,5
ten,10
"one hundred",100
"two thousand",2000
million,1000000
$ cat spanish_numbers.csv
numero,value
cero,0
cuatro,4
cinco,5
seis,6
siete,7
ocho,8
nueve,9
cien,100
mil,1000
millón,1000000
$ bqljoin english_numbers.csv value spanish_numbers.csv value
number,value,numero,value
zero,0,cero,0
one hundred,100,cien,100
million,1000000,millón,1000000
four,4,cuatro,4
five,5,cinco,5
$ bqljoin -type left english_numbers.csv value spanish_numbers.csv
    value
number,value,numero,value
zero,0,cero,0
one,1,"",""
ten,10,"",""
one hundred,100,cien,100
million,1000000,millón,1000000
two,2,"",""
two thousand,2000,"",""
three,3,"",""
four,4,cuatro,4
```

```
five,5,cinco,5
$ bqljoin -type right english_numbers.csv value spanish_numbers.csv
    value
number,value,numero,value
zero,0,cero,0
one hundred,100,cien,100
"","",mil,1000
million,1000000,millón,1000000
four,4,cuatro,4
five,5,cinco,5
"","",seis,6
"","",siete,7
"","",ocho,8
"","",nueve,9
$ bqljoin -type full english_numbers.csv value spanish_numbers.csv
    value
number,value,numero,value
zero,0,cero,0
one,1,"",""
ten,10,"",""
one hundred,100,cien,100
"","",mil,1000
million,1000000,millón,1000000
two,2,"",""
two thousand,2000,"",""
three,3,"",""
four,4,cuatro,4
five,5,cinco,5
"","",seis,6
"","",siete,7
"","",ocho,8
"","",nueve,9
```

2.6. bqlgroupby

Nombre bqlgroupby - Agrupa una tabla según las columnas especificadas, usando los acumuladores especificados para las demás columnas.

Sinopsis bqlgroupby {-g columnaGrupo}... {-c columnaReducir -f funcionReduccion}

Descripción Lee una tabla por stdin y la agrupa usando las columnas seleccionadas. En las columnas seleccionadas para reducción aplica una función de reducción sobre los valores de dichas columnas.

- **-g** Indica según qué columna se hará el agrupamiento. Ejm: «-g year -g month».
- **-c** Indica de qué columna se tomarán los datos para la función de reducción asociada.
- **-f** Indica qué comando se va a correr como función de reducción. Se puede utilizar cualquier comando que tome sus entradas por stdin (una

por línea) y devuelva el resultado por stdout cuando reciba EOF. BashQL contiene las siguientes funciones de reducción:

bqlsum Suma los valores recibidos.

bqlcount Cuenta los valores recibidos.

bqldistinct Cuenta los valores únicos recibidos.

bqlavg Calcula el promedio de los valores recibidos.

bqlmax Retorna el mayor valor recibido.

bqlmin Retorna el menor valor recibido.

Ejemplo:

```
$ head visitas.csv
year,month,day,visits
2014,12,06,4253
2014,12,05,26286
2014,12,04,11716
2014,12,03,10206
2014,12,02,26719
2014,12,01,20816
2014,11,30,24176
2014,11,29,17463
2014,11,28,9194
$ bqlfrom visitas.csv \
| bqlgroupby -g month \
-c visits -f bqlsum \
-c visits -f bqlavg \
-c visits -f bqlmin \
-c visits -f bqlmax
month,visits,visits,visits,visits
01,6.50238e+06,16134.937965260546,62,32765
02,5.976605e+06,16285.027247956403,37,32703
03,6.772536e+06,16722.31111111111,8,32713
04,6.802518e+06,16196.471428571429,132,32722
05,7.27781e+06,16769.14746543779,22,32717
06,6.784116e+06,16152.657142857142,78,32581
07,7.227396e+06,16652.986175115206,116,32626
08,7.131601e+06,16432.260368663596,53,32675
09,6.857604e+06,16327.628571428571,30,32419
10,6.989063e+06,16103.831797235023,67,32767
11,6.852773e+06,16316.12619047619,80,32708
12,6.672185e+06,16313.41075794621,21,32743
```

2.7. bqlheader

Nombre bqlheader - Cambia el nombre de las columnas.

Sinopsis bqlheader [opciones] nombres...

Descripción Cambia o agrega el nombre de las columnas del CSV pasado por stdin y lo devuelve por stdout.

-i Sinónimo de insert.

-insert Inserta el encabezado, se usa si el archivo CSV no trae encabezado.

Ejemplo:

```
$ bqlfrom visitas.csv \
| bqlgroupby -g month \
-c visits -f bqlsum \
-c visits -f bqlavg \
-c visits -f bqlmin \
-c visits -f bqlmax \
| bqlheader mes total promedio_diario minimo maximo
mes,total,promedio_diario,minimo,maximo
01,6.50238e+06,16134.937965260546,62,32765
02,5.976605e+06,16285.027247956403,37,32703
03,6.772536e+06,16722.31111111111,8,32713
04,6.802518e+06,16196.471428571429,132,32722
05,7.27781e+06,16769.14746543779,22,32717
06,6.784116e+06,16152.657142857142,78,32581
07,7.227396e+06,16652.986175115206,116,32626
08,7.131601e+06,16432.260368663596,53,32675
09,6.857604e+06,16327.628571428571,30,32419
10,6.989063e+06,16103.831797235023,67,32767
11,6.852773e+06,16316.12619047619,80,32708
12,6.672185e+06,16313.41075794621,21,32743
```

2.8. bqlsort

Nombre bqlsort - Ordena las filas según las columnas pasadas como argumento.

Sinopsis bqlsort [opciones] columnas...

Descripción Recibe por stdin un archivo CSV normalizado y devuelve por stdout el archivo con las filas ordenadas de forma ascendente según las columnas pasadas como parámetro.

Opciones: El comando acepta las siguientes opciones:

-r Ordena de forma descendente

-reverse Sinónimo de -r.

Ejemplo:

```
$ bqlfrom poblacion.csv \
| bqlwhere Tipo = Provincia \
| bqlwhenever Nombre '<' 'Cartago' \
             Nombre '>' 'Cartago' \
| bqlsort Nombre
Nombre,Total,Hombre,Mujer,Tipo
```

```
Alajuela,847660,420636,427024,Provincia
Guanacaste,326821,161932,164889,Provincia
Heredia,433975,211417,222558,Provincia
Limón,386954,193673,193281,Provincia
Puntarenas,410914,205975,204939,Provincia
San José,1403963,671434,732529,Provincia
```