Exemplo Prático de uma Solução em Bash

Guilherme Bittencourt Bueno da Silva

Universidade Federal do Paraná

gbbs14@inf.ufpr.br

27 de Agosto de 2018

Resumo

O exercício a seguir propõe ao leitor encontrar uma solução automatizada para filtrar e separar dados específicos de um arquivo, usando apenas o bash (e auxiliares, invocados pelo bash). A solução inclui a pesquisa por ferramentas que possam ser de auxílio na solução, como sed, awk, cut, entre outros (para usar o bash da melhor forma é preciso conhecer, ou ser capaz de pesquisar por, ferramentas que auxiliam as soluções).

Introdução

As expansões do Bash, juntamente com os redirecionamentos, são muito práticos, porém, para qualquer programador, é difícil automatizar tarefa sem utilizar estruturas de controle como 'if' e 'for' (while), por isso é difícil encontrar exemplos reais de uso para automatização que usem apenas as ferramentas apresentadas até o momento. Para o exercício seguinte não é necessário o uso de 'if', mas é recomendado o uso do 'for' para obter uma solução ideal. O material do prodessor Roberto Hexsel [Hexsel, 2018] apresenta com exemplos práticos e exercícios sobre o conteúdo necessário para a solução do exercício deste capítulo, incluíndo comando for, quotes e variáveis.

Exercício

Para um arquivo patrimonio.csv no seguinte formato:

```
...
"447676";"ARMARIO";"ALTO␣COM␣DUAS␣PORTAS";"";"2100.13.03.31";
"447677";"ARMARIO";"ALTO␣COM␣DUAS␣PORTAS";"";"2100.13.03.29";
"447678";"ARMARIO";"ALTO␣COM␣DUAS␣PORTAS";"";"2100.13.03.30";
"447679";"ARMARIO";"ALTO␣COM␣DUAS␣PORTAS";"";"2100.13.03.28";
"452782";"CADEIRA␣GIRATORIA";"BASE␣A␣GAS";"FUNPAR";"2100.13.07";
"460805";"FORNO␣DE␣MICROONDAS";"30L";"CONSUL";"2100.13.02.24";
"460806";"VENTILADOR␣DE␣TETO␣(5234)";"127V␣";"";"2100.13.02.21";
"460807";"VENTILADOR␣DE␣TETO␣(5234)";"127V␣";"";"2100.13.02.21";
"460808";"VENTILADOR␣DE␣TETO␣(5234)";"127V␣";"";"2100.13.02.21";
...
```

Em cada linha, o conteúdo das colunas está entre aspas (" "), separados por ponto e vírgula (;). Porém, o conteúdo de alguns campos pode conter o caractere (;). Exemplo: Original:

```
"460808";"VENTILADOR␣DE␣TETO;␣CINZA";"127V␣";"";"2100.13.02.21";
```

Filtrado:

```
"460808";"VENTILADOR␣DE␣TETO␣CINZA";"127V␣";"";"2100.13.02.21";
```

A primeira etapa do exercício é filtrar todo o conteúdo do arquivo e remover essas ocorrências. A segunda etapa, é, a partir do resultado filtrado, obter em um arquivo, todos os locais (conteúdo da quinta coluna), um em cada linha, ordenados e sem repetição. Exemplo:

```
...
2100.13.02.21
2100.13.02.23
2100.13.02.24
2100.13.02.25
2100.13.02.29
2100.13.03.01
2100.13.03.02
2100.13.03.03
2100.13.03.04
...
```

Por fim, deve-se criar um arquivo para cada local, com o nome sendo o código do local com a extensão ".csv". O conteúdo de cada arquivo deve ser as linhas do arquivo original em que o campo de local corresponde ao nome do arquivo. Exemplo: 2100.13.csv:

```
...
"480170";"NO-BREAK␣(5230)";"";"";"2100.13";
"480171";"MICROCOMPUTADOR␣";"";"";"2100.13";
"480172";"NOTEBOOK";"";"";"2100.13";
"480173";"MONITOR␣DE␣VIDEO␣(5235)";"";"";"2100.13";
"480174";"NOTEBOOK";"";"";"2100.13";
"480175";"MICROCOMPUTADOR␣";"";"";"2100.13";
...
```

Solução

```
#!/bin/sh

# remover ";" e gerar locais.txt
cat patrimonio.csv | sed -E 's/(([^"])(;)|(;)([^"]))*/\2/g' | grep
   -oE '"([0-9]+(\.[0-9]+)+)";' | grep -oE '[0-9]+(\.[0-9]+)+' |
   sort -u > locais.txt;
# se não existir, cria diretorio locais
mkdir -p locais
# para cada local, preencher os dados
for i in $(cat locais.txt); do
  cat patrimonio.csv | grep "\"$i\"\;" >> locais/$i.csv;
done
```

Eliminar ;

A maneira mais intuitiva de solucionar o problema é um editor de fluxo de texto (não acessa o arquivo por posições, e sim, acessa o arquivo linearmente através de um buffer). Assim, basta utilizar uma expressão regular [Wikipédia, 2018] para identificar o padrão e eliminar o ; indesejado. A atividade não é trivial. Ao buscar por soluções parecidas em sites de perguntas e respostas e tutoriais, é comum encontrar soluções quem eliminam todo o conteúdo dentro de delimitadores, mas não de eliminar caracteres específicos dentro de delimitadores. O sed tem uma solução necessária, porém, pode ser difícil encontrar, visto que a maioria dos tutoriais, apesar de bem completos, são bem antigos e o uso do sed pode variar entre diferentes shells. Uma solução possível com o sed é:

```
sed -E 's/(([^"])(;)|(;)([^"]))*/\2/g'
```

A opção -E permite expressões regulares extendidas, que são necessárias para que o sed remova o ; mas mantenha o resto do conteúdo. A expressão regular busca por ; que não sejam seguidos de " ou que não aparecem logo após um ". Dessa maneira, os ponto e vírgula de separação, estão entre " de ambos os lados e não são detectados pela expressão, sobrando apenas os ; que devem ser isolados. A opção s/// substitui o padrão encontrado por outra coisa, o \2 é o segundo padrão identificado, (os padrões são identificados por "(" e ")") na ordem de associação. Com isso, é trocado o padrão casado pelo segundo elemento encontrado (no primeiro caso do or, a parte que não é o ";", e no segundo caso, nada, pois ele não casa nada com o padrão da primeira parte do or). Essa solução não é perfeita em casos onde o ; seja o primeiro caracter, pois ele é encontrado no segundo caso do or, e nesse caso, a parte do padrão a ser mantida não é mais o segundo padrão encontrado. Para corrigir isso é possível utilizar uma expressão regular diferente ou executar 2 seds em seguida, um para cada caso.

Gerar locais.txt

A partir do retorno do sed anterior, é possível usar o grep para selecionar apenas os locais. a opção -E permite o uso de expressões regulares e -o para retornar apenas o padrão

encontrado, e não as linhas que contém o padrão. No primeiro grep, são selecionados apenas os campos cujo conteúdo completo seja apenas o código de local e em seguida o próximo grep busca apenas o código, sem os " e ; para isolar o conteúdo. Dessa maneira, mesmo que outro campo cite um código de local, ele não será detectado pelo grep pois o conteúdo da célula não é somente o código de local. Outro modo de isolar apenas a coluna de locais, é utilizar um editor de fluxo para separar os campos por ; (após resolver o problema de ; dentro do conteúdo), porém, caso uma coluna seja adicionada antes dessa, o script deve ser alterado para selecionar a coluna correta. O ponto negativo da primeira solução é que se for criada uma outra coluna de referência a locais, 2 colunas serão selecionadas. Independente da maneira escolhida, com os códigos de locais isolados, basta ordenar removendo repetições e redirecionar a saída.

```
grep -oE '"([0-9]+(\.[0-9]+)+)";' | grep -oE '[0-9]+(\.[0-9]+)+' |
    sort -u > locais.txt;
```

Preencher arquivos de locais

Por fim, para cada linha do arquivo locais.txt (ou seja, para cada local), são selecionadas as linhas do arquivo original que contém esse local. O (») redireciona concatenando ao que já foi escrito, assim, cada linha é adicionada sem substituir o conteúdo anterior.

```
for i in $(cat locais.txt); do
  cat patrimonio.csv | grep "\"$i\";" >> locais/$i.csv;
done
```

Conclusão

Existem várias soluções possíveis, usando várias ferramentas, existem diferentes maniras de encontrar partes do texto. A solução deve levar em consideração o contexto dos dados, por exemplo, se for comum alterar a estrutura do arquivo com mais colunas, é ideal uma solução fácil de alterar ou até mesmo que receba nomes de arquivos, caracteres separadores ou números de coluna como parâmetros do script.

Referências

[Hexsel, 2018] Hexsel, R. A. (2018). Como computeiros ganham mais tempo com bash. http://www.inf.ufpr.br/roberto/ci064/labBash-2.html. Acessado em 27-08-2018.

[Wikipédia, 2018] Wikipédia (2018). Expressão regular. https://pt.wikipedia.org/wiki/Express%C3%A3o_regular. Acessado em 27-08-2018.