Segundo Exercício prático de Bash

Guilherme Bittencourt Bueno da Silva

Universidade Federal do Paraná

gbbs14@inf.ufpr.br

10 de Setembro de 2018

Resumo

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

Introdução

É comum que dados importantes estejam divididos em vários diretórios e que seja necessário percorrer por vários arquivos diferentes para obter um resultado. O exercício a seguir exige a junção de dados de arquivos de diretórios diferentes. A solução apresentada nesse material utiliza estruturas de controle como loops e branches, além de pipeline e expansões do bash.

Exercício

Os dados estão organizados da seguinte forma: Cada diretório representa uma disciplina, dentro de cada um deles, existem arquivos semestrais, do primeiro semestre de 1988 até o segundo semestre de 2002. O nome desses arquivos é o ano, seguido do semestre, terminado em .dados. Por exemplo: 19942.dados é o arquivo de dados do ano 1994 no segundo semestre. Em cada arquivo .dados começa com uma linha contendo "curso:grr", e as linhas seguintes contém o curso e grr dos alunos matriculados na disciplina nesse semestre (seguindo o formato descrito na primeira linha, "curso:grr").

A primeira etapa do exercício é listar a quantidade de alunos matriculados em todas as disciplinas para cada semestre. O resultado deve ser no formato do exemplo:

```
...
19892 : 414
19901 : 421
19902 : 493
19911 : 760
19912 : 764
...
```

A segunda etapa é listar, para cada disciplina, a relação de matriculados de cada curso em cada semestre, um semestre por linha.

Solução

Parte 1

```
#!/bin/sh

path="/nobackup/bcc/$USER/DadosMatricula"

# escolhe uma das disciplinas e lista os semestres
for i in $(ls -1 \$path/$(ls -1 $path | tail -1) | cut -c-5); do
  # para cada semestre, conta as linhas de todos os arquivos de dados
  # daquele semestre (dentro de todos os diretorios de disciplinas),
  # removendo linhas repetidas
  echo "$i : $(cat $path/*/$i.dados | sort -du | head -n -1 | wc
     -l)";
done
```

Primeiramente, o diretório dos dados é definido em uma varável para evitar repetição e facilitar modificações. O laço serve para percorrer todos os semestres de um disciplina. Como todas as disciplinas devem possuir arquivos de dados para o mesmo período de tempo, não importa qual disciplina é utilizada para listar os semestres existentes.

Nesse caso, ls -1 lista 1 arquivo por linha, então, o ls -1 de dentro lista todas as disciplinas do diretório dos dados, e o tail -1 pega apenas a última disciplina, e essa, será a disciplina usada para o primeiro ls -1 listar os semestres. O cut -c-5 serve para manter apenas o ano e semestre e remover o ".dados"dos nomes.

Dentro do laço, o echo imprime o ano-semestre, seguido da quantidade de matrículas. O cat está sendo aplicado para todos os arquivos de semestre de todas as disciplinas com a expansão *. O sort -u serve para remover linhas duplicadas (já considerando as matrículas de todas as disciplinas naquele semestre) e o sort -d apenas ordena para que seja possível remover a última linha, que será a linha que não começa com um número, ou seja, o cabeçalho (que só aparece uma vez, devido ao sort -u remover linhas duplicadas). Por fim o wc -l conta a quantidade de linhas, que após o tratamento anterior, é a quantidade de matrículas daquele semestre.

Parte 2

```
#!/bin/sh

path="$HOME/DadosMatricula"

# escolhe uma das disciplinas e lista os semestres
for disc in $(ls -1 $path); do
  echo "$disc:␣--------------------";
  for sem in $(ls -1 $path/$disc | cut -c-5); do
    cod=0;
    count=0;
    print=0;
    for lin in $(cat $path/$disc/$sem.dados | tail -n +2); do
      if [ $(echo $lin | cut -c-2) -eq $cod ]; then
        count=$(($count + 1));
      else
        if [ "$cod" -ne 0 ]; then
          if [ $print -eq 0 ]; then
            print=1;
            echo -n "$sem␣␣␣␣-";
          fi
        echo -n "␣$cod␣:␣$count␣/";
        fi
        cod=$(echo $lin | cut -c-2);
        count=$((1));
      fi
    done
    if [ "$cod" -ne 0 ]; then
      if [ $print -eq 0 ]; then
        print=1;
        echo -n "$sem␣␣␣␣-";
      fi
      echo -n "␣$cod␣:␣$count␣/";
    fi
    if [ $print -eq 1 ]; then
      echo "␣";
    fi
  done
done
```

Essa solução percorre, para cada disciplina, todos os semestres, e, para cada um deles, conta as linhas que começam com o mesmo código. A parte mais difícil desse exercício é ignorar os semestres sem matrículas na disciplina. Uma solução possível é contar a quantidade de códigos de cursos diferentes (salvando os códigos), e contar a quantidade de matrículas de cada um deles e assim, no final, se a quantidade de códigos de curso for 0, não é necessário imprimir a linha daquele semestre, para a disciplina atual. Entretanto, essa solução resolve o problema sem guardar a quantidade de cursos diferentes encontrados, aproveitando-se de que as matrículas de cada curso sempre aparecem em sequência (e caso

isso seja alterado, é possível as ordenar).

O laço externo é o mais simples, apenas percorre as disciplinas, imprimindo o nome da disciplina. Dentro desse, o segundo laço percorre pelos semestres dessa disciplina, novamente removendo o ".dados"para melhorar a impressão. Em seguida são inicializadas váriáveis para:

- Guardar o código encontrado mais recente
- Contar em quantas linhas esse código aparece
- Guardar se o cabeçalho do semestre já foi impresso

Essa última variável é necessária pois a solução imprime a quantidade de matrículas de um curso assim que encontra um novo código de curso ou o fim do arquivo. Porém, se o cabeçalho do semestre fosse impresso antes do laço, ele seria impresso até para os semestres sem nenhuma matrícula.

Dentro do segundo laço, o terceiro laço percorre as linhas do arquivo de semestre (exceto a linha de cabeçalho, removida pelo tail -n +2, que inverte o parâmetro do tail, e, ao invés de manter uma quantidade de fixa de linhas no final do arquivo, remove uma quantidade fixa de linhas no começo do arquivo). Para cada linha, ao encontrar um código novo, o resultado do código antigo é impresso, o código é salvo e a contagem se inicia. (exceto para o código 0, que é o número de inicialização da variável). No caso de o código ser igual ao código salvo, o contador aumenta. Ao terminar o laço, imprime o resultado do último código de curso encontrado, pois a impressão acontece dentro do laço ao encontrar um novo código. Antes de qualquer impressão, é verificado se o cabeçalho do semestre já foi impresso, caso não tenha sido, ele é imprimido e a variável que guarda a confirmação dessa impressão é atualizada, de modo a imprimir o cabeçalho do semestre apenas uma vez por semestre. Por fim, caso um ou mais cursos tenham sido impressos, o echo ” ” apenas pula uma linha antes de iniciar o próximo laço.

Conclusão

Devido a quantidade de branches para tratar o caso de arquivos sem matrícula, essa solução tem um desempenho pior que a solução que guarda os códigos e a quantidade de ocorrências em vetores. Outra solução possível, com menos branches, é, realizar esse tratamento antes de entrar no laço, verificando se o arquivo do semestre possui apenas uma linha (de cabeçalho), e nesse caso, ignorar esse semestre.