README.doc

12191632 윤진용

1. Get the data of the movie identified by a specific 'movie id' from 'u.item'

```
if [ $choicenum -eq 1 ];then
          read -p "Please enter 'movie id' (1~1682):" id
          cat $1 | awk -F'|' -v id=$id '$1==id {print $0}'
```

how to work

첫번째 인자로 준 u.item($1) 파일 출력을 파이프로 연결하여 ->(output 넘겨줘) awk를 이용하여 read로 받은 id를 awk내에 쓸 수 있는 변수로 만들고 u.item의 첫 번째 column(즉 user-id)과 id가 같다면 전체 column을 출력한다. 여기서 -F 옵션을 통해 |를 column delimeter로 사용했다.

2. Get the data of 'action' genre movies from 'u.item'

```
elif [ $choicenum -eq 2 ];then
             read -p "Do you want to get the data of 'action' genre movies
from 'u.item'?(y/n):" answer
             if [ $answer = "y" ];then
               cat $1 | awk -F'|' 'NR <= 35 && $7==1 {print $1 $2}'
             else
               continue
             fi
```

how to work

read -p "Do you want to get the data of 'action' genre movies from 'u.item'?(y/n):" answer

를 통해 y를 입력 받으면,

awk를 사용해 |를 column delimiter로 사용하여 1번 2번 coulmn을 출력한다

단 action 장르만 출력하고 10개만 출력해야 하기에 다음과 같은 조건을 붙여준다.

```
'NR <= 35 && $7==1
```

3. Get the average 'rating' of the movie identified by specific 'movie id' from 'u.data'

```
elif [ $choicenum -eq 3 ];then
             read -p "Please enter the 'movie id'(1~1682):" id
             cat $2 | awk -v id=$id '$2==id {sum+=$3; count++} END {if(count
> 0) printf "average rating of %d : %.5f \n", id, sum/count}'
```

how to work

read -p "Please enter the 'movie id'(1~1682):" id 를 통해 movie id를 입력 받으면 u.data를 통해

해당 영화의 rating을 통해 평균을 구한다.

awk를 사용하여 id를 변수로 받고, 두번째 column, 즉 movie id가 입력해준 id인 것만 골라서 sum에 더한 다음 (count도 평균을 위해 따로 더해 줌) 마지막에 sum/count연산을 printf로 출력해준다. 이때 반올림도 처리해 줌

4. Delete the 'IMDb URL' from 'u.item'

```
elif [ $choicenum -eq 4 ];then
            read -p "Do you want to delete the 'IMDb URL' from
'u.item'?(y/n):" answer
            if [ $answer = "y" ];then
                cat $1 | sed 's|http://[^|]*||' | head -n 10
            else
                continue
            fi
```

how to work

u.item에서 중간에 껴 있는 URL을 삭제하고, 처음 10줄을 출력한다.

read -p "Do you want to delete the 'IMDb URL' from 'u.item'?(y/n):" answer 를 통해 asnwer를 받고 answer가 y일 경우에 동작한다.

's|http://[^|]*||' 이 부분이 특징적인데 s는 배웠듯이 치환 명령이고 |를 delimiter로 http://로 시작하는 모든 문자열을 대체한다. -> 하지만 뒤에 [^|]* 가 나와 | 문자가 나타날 때까지의 모든 문자열로 의미가 바뀌게 된다. ||는 빈 문자열로 대체한다는 의미이다. 따라서 url 문자열이 없어진다.

마지막에 head -n 10으로 출력의 처음 10줄만 보여주도록 설정했다.

5. Get the data about users from 'u.user'

```
elif [ $choicenum -eq 5 ];then
            read -p "Do you want to get the data about users from
'u.user'?(y/n):" answer
            if [ $answer = "y" ];then
               cat $3 | sed 's/F/female/g; s/M/male/g' | awk -F'|' 'NR <=
10 {printf "user %s is %s years old %s %s\n", $1, $2, $3, $4}'
            else
                  continue
            fi
```

how to work

read -p "Do you want to get the data about users from 'u.user'?(y/n):" answer 를 통해 answer를 입력 받고 만약 answer가 y일 경우 동작한다.

u.user로부터의 데이터 출력 형식을 변환하는 코드이다.

조금 돌아간 느낌이 있는데 u.user 파일을 출력한 cat 커맨드 결과를 받아서, sed를 이용해서 F또는 M을 female과 male 표시로 바꿔준다.

해당 결과값에서 awk를 이용해서 |를 delimiter로 column을 구분하고

printf "user %s is %s years old %s %s₩n", $1, $2, $3, $4 를 통해 형식을 변환한다.

이때 10행까지만 출력하기 위해 NR <= 10 조건을 붙였다.

6. Modify the format of 'release date' in 'u.item'

```
elif [ $choicenum -eq 6 ];then
            read -p "Do you want to Modify the format of 'release data' in
'u.item'?(y/n):" answer
            if [ $answer = "y" ];then
                  cat  $1  |  sed  -n  's/\([0-9]\{2\}\)-\([A-Za-z]\{3\}\)-
\([0-9]\{4\}\)/\3\2\1/p'|   sed   's/Jan/01/g;   s/Feb/02/g;   s/Mar/03/g;
s/Apr/04/g;  s/May/05/g;  s/Jun/06/g;  s/Jul/07/g;  s/Aug/08/g;  s/Sep/09/g;
s/Oct/10/g; s/Nov/11/g; s/Dec/12/g' | awk -F'|' '$1>1672 && $1<1683 {print
$0}'
            else
                  continue
            fi
```

how to work

u.item의 개봉일자 형식을 변환해주는 코드이다.

read -p "Do you want to Modify the format of 'release data' in 'u.item'?(y/n):" answer 를 통해

answer 값이 y이면 동작한다.

전체적인 변화를 보자면

sed -n 's/₩([0-9]₩{2₩}₩)-₩([A-Za-z]₩{3₩}₩)-₩([0-9]₩{4₩}₩)/₩3₩2₩1/p'

를 통해 개봉일자 column에서 -를 제거해준다. (1996-Feb-03 -> 1996Feb03)

sed 's/Jan/01/g; s/Feb/02/g; s/Mar/03/g; s/Apr/04/g; s/May/05/g; s/Jun/06/g; s/Jul/07/g; s/Aug/08/g; s/Sep/09/g; s/Oct/10/g; s/Nov/11/g; s/Dec/12/g'

를 이용하여 문자열 형식의 월 표시를 숫자로 바꿔준다. (무식한 방법이지만 효과가 좋다)

awk -F'|' '$1>1672 && $1<1683 {print $0}'

마지막 awk 구문을 통해 1672년부터 1682년에 속하는 행만 출력해준다.

7. Get the data of movies rated by a specific 'user id' from 'u.data'

```
elif [ $choicenum -eq 7 ];then
            read -p "Please enter the 'user id'(1~943):" id
            tempfile=$(mktemp)
            cat $2 | sort -k2,2n | awk -v id=$id '$1==id {movieIds =
movieIds $2 "|"} END {print substr(movieIds, 1, length(movieIds)-1)}'
            cat $2 | sort -k2,2n | awk -v id=$id '$1==id {movieIds =
movieIds $2 "|"} END {print substr(movieIds, 1, length(movieIds)-1)}' | tr
'|' '\n' > $tempfile
            echo -e "\n"
            awk -F'|' 'NR < 11 && NR==FNR{a[$1]++;next}{if($1 in a){printf
"%d | %s\n", $1, $2}}' $tempfile $1
            rm $tempfile
```

how to work

전체 흐름부터 살펴보자, read -p "Please enter the 'user id'(1~943):" id 를 통해 원하는 user id를 입력 받으면 u.data 파일에서 매칭되는 movie id를 찾아 구분자를 덧붙이며 하나의 문자열로 만든다.

```
cat $2 | sort -k2,2n | awk -v id=$id '$1==id {movieIds = movieIds $2 "|"}
END {print substr(movieIds, 1, length(movieIds)-1)}'
```

u.data 파일을 출력하고 -> (결과 넘겨) u.data 출력 결과값을 두 번째 column인 movie id로 정렬한다 -> (결과 넘겨) 입력 받은 id와 같은 user id를 갖는 행을 movieIds = movieIds $2 "|"

movieIds는 두 번째 열(movie id) 와 | 로 구성되는데 이를 movieIds 문자열에 더한다.

print substr(movieIds, 1, length(movieIds)-1)

마지막에는 마지막 |를 뺀 다음 출력해준다.

이러면 첫 번째 조건인 첫 번째 출력은 완성된다.

제일 처음 mktemp 커멘드를 통해 만든 tempfile 임시 파일에다 위의 출력값을 담는데

이때 하나의 열로 나오도록 파일을 구성한다. (| 제거하고 movie id만 하나의 열로 나오도록)

awk -F'|' 'NR < 11 && NR==FNR{a[$1]++;next}{if($1 in a){printf "%d | %s₩n", $1, $2}}' $tempfile $1

awk에서 두 파일을 합치는 방식을 이용해 tempfile의 첫번째 열과 u.item의 첫번째 열을 비교하여 같다면 printf "%d | %s₩n", $1, $2 출력한다.

(a[$1]은 $tempfile의 첫 번째 열을 a로 지정해준 것!)

8. Get the average 'rating' of movies rated by users with 'ag e' between 20 and 29 and 'occupation' as 'programmer'

```
elif [ $choicenum -eq 8 ];then
            read -p "Do you want to get the average 'rating' of movies
rated by users with 'age' between 20 and 29 and 'occupation' as
'programmer'?(y/n):" answer
            if [ $answer = "y" ];then
                  tempfile=$(mktemp)
                   cat $3 | awk -F'|' '$2>19 && $2<30 && $4=="programmer"
{printf "%d\n",$1}'>$tempfile
                   tempfile2=$(mktemp)
                   awk    'NR==FNR{a[$1]++;next}{if($1    in    a){printf
"%d  %d\n",$2,$3}}' $tempfile $2 | sort -k1 >$tempfile2
                   awk '{sum[$1] += $2;count[$1]++} END { for (key in sum)
{ printf "%d %.5f\n", key, sum[key]/count[key]}}' $tempfile2
             else
                  continue
            fi
```

read -p "Do you want to get the average 'rating' of movies rated by users with 'age' between 20 and 29 and 'occupation' as 'programmer'?(y/n):" answer 를 통해 answer 값이 y이면 동작한다.

먼저 u.user 파일에서 20~29살 사이이고, 직업이 programmer인 user를 걸러준다.

```
tempfile=$(mktemp)
                   cat $3 | awk -F'|' '$2>19 && $2<30 && $4=="programmer"
{printf "%d\n",$1}'>$tempfile
```

(임시파일 tempfile로 결과값 넣어주기, 그럼 tempfile에는 현재 조건을 만족하는 user id만 한 열로 나와있다.)

```
tempfile2=$(mktemp)
                   awk    'NR==FNR{a[$1]++;next}{if($1    in    a){printf
"%d  %d\n",$2,$3}}' $tempfile $2 | sort -k1 >$tempfile2
                   awk '{sum[$1] += $2;count[$1]++} END { for (key in sum)
{ printf "%d %.5f\n", key, sum[key]/count[key]}}' $tempfile2
```

7번에서 했던 것처럼

awk 'NR==FNR{a[$1]++;next}{if($1 in a){printf "%d %d₩n",$2,$3}}' $tempfile $2

를 통해 tempfile의 첫번째 열과 u.data의 첫번째 열(user id)을 비교하여 같은 것만 골라

Movie_id rating

구조로 출력해준다. sort -k1 >$tempfile2 를 통해 같은 movie_id끼리 모아 정렬하고 tempfile2 임시 파일에 저장해준다.

awk '{sum[$1] += $2;count[$1]++} END { for (key in sum) { printf "%d %.5f₩n", key, sum[key]/count[key]}}' $tempfile2

tempfile2의 첫번째 열을 sum으로 지정하고(배열 고유 매칭), 두 번째 열인 rating을 더해준다.

결국 평균을 구해야 하니 동일한 movie id 개수를 구하기 위해 count도 증가시켜주고, 첫 번째 열(movie id)의 고유 값을 키로 루프를 도는데, 각 고유 값의 첫 번째 열 값(key)과 해당 고유 값 그룹의 두 번째 열 값 합계를 카운트 값으로 나눈 평균을 출력한다. 정확히는 key즉 movie id와 평균을 출력한다.

9. Exit

```
elif [ $choicenum -eq 9 ];then
            stop="Y"
            echo "Bye!"
            break
```

while문이 stop="Y"이면 멈추기 때문에 다음 반복 때 멈춤 -> 혹시 몰라서 break 설정

각 결과에 대한 동작은 github에서 확인 바람!

https://github.com/jinyongyun/OSS_shell_script_made