오픈소스\_SW\_개론\_Project1

12191627 오영제

0. 전체 코드

#!/bin/bash

fun1() {

    echo -n "Please enter 'movie id'(1~1682) : "

    read movieId

    awk "NR == $movieId" $1

}

fun2() {

    echo -n "Do you want to get the data of ‘action’ genre movies  from '$1’?(y/n) :"

    read confirm

    if [ "$confirm" != 'y' ]; then

        return

    fi

    awk -F'|' '$7==1 {print $1, $2; cnt++} cnt == 10 {exit}' $1 | sort -k 1 -n

}

fun3() {

    echo -n "Please enter the 'movie id’ (1~1682) :"

    read movieId

    awk -v movieId="$movieId" '$2 == movieId { sum += $3; cnt++} END { if (cnt > 0) printf "average rating of %d : %.5f\n", movieId, (sum/cnt) }' $1

}

fun4() {

    echo -n "Do you want to delete the ‘IMDb URL’ from ‘$1’?(y/n):"

    read confirm

    if [ "$confirm" != 'y' ]; then

        return

    fi

    sed -n '1,10 s/|http.\*imdb.com.\*)|/||/p' $1

}

fun5() {

    echo -n "Do you want to get the data about users from ‘$1’?(y/n) :"

    read confirm

    if [ "$confirm" != 'y' ]; then

        return

    fi

    sed -n '1, 10 s/\([0-9]\*\)|\([0-9]\*\)|\([MF]\)|\([a-zA-Z]\*\)|\(.\*\)/user \1 is \2 years old \3 \4/p' $1 | sed 's/M/male/; s/F/female/'

}

fun6() {

    echo -n "Do you want to Modify the format of ‘release data’ in ‘$1’?(y/n) :"

    read confirm

    if [ "$confirm" != 'y' ]; then

        return

    fi

    sed -n '1673, 1682 s/\([0-9]\*\)-\([a-zA-Z]\*\)-\([0-9]\*\)/\3\2\1/p' $1 | sed 's/Jan/01/; s/Feb/02/; s/Mar/03/; s/Apr/04/; s/May/05/; s/Jun/06/; s/Jul/07/; s/Aug/08/; s/Sep/09/; s/Oct/10/; s/Nov/11/; s/Dec/12/'

}

fun7() {

    echo -n "Please enter the ‘user id’(1~943) :"

    read userId

    echo $userId

    awk -v userId="$userId" '{if($1 == userId && cnt < 10) {cnt++; print $2}}' $2 | sort -k 1 -n | awk '{printf "%s|", $0} END {printf "%s\n\n", $0}'

    awk -v userId="$userId" '{if(NR == FNR) {if($1 == userId && cnt < 10) {cnt++; data[$2]} next}} {FS = "|"} $1 in data {print $1,$2}' $2 $1 | sed 's/ /|/'

}

fun8() {

    echo -n "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) :"

    read confirm

    if [ "$confirm" != 'y' ]; then

        return

    fi

    awk -F'|' '{if(NR == FNR) {if($4 == "programmer" && 20 <= $2 && $2 <= 29 ) userId[$1] } else {{FS = " "} for(loop in userId){if($1 in userId){sumArr[$2] += $3; cntArr[$2]++}}}} END{for(key in sumArr){if ((sumArr[key]/cntArr[key]) == int(sumArr[key]/cntArr[key])){printf "%d %d\n", key, (sumArr[key]/cntArr[key])}else{num = sprintf("%.5f", (sumArr[key]/cntArr[key])); printf "%d %g\n", key, num}}}' $2 $1

}

if [ $# -ne 3 ]; then

    echo "Invalid Arguments Number"

    exit -1

fi

echo "User Name : 오영제"

echo "Student Number : 12191627"

echo "[ MENU ]"

echo "1. Get the data of the movie identified by a specific 'movie id' from '$1'"

echo "2. Get the data of action genre movies from '$1’"

echo "3. Get the average 'rating’ of the movie identified by specific 'movie id' from '$2’"

echo "4. Delete the ‘IMDb URL’ from ‘$1"

echo "5. Get the data about users from '$3’"

echo "6. Modify the format of 'release date' in '$1’"

echo "7. Get the data of movies rated by a specific 'user id' from '$2'"

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

echo "9. Exit"

echo "--------------------------"

while true;

do

    echo -n "Enter your choice [ 1-9 ] : "

    read flag

    case $flag in

        1)

            fun1 "$1"

            ;;

        2)

            fun2 "$1"

            ;;

        3)

            fun3 "$2"

            ;;

        4)

            fun4 "$1"

            ;;

        5)

            fun5 "$3"

            ;;

        6)

            fun6 "$1"

            ;;

        7)

            fun7 "$1" "$2"

            ;;

        8)

            fun8 "$2" "$3"

            ;;

        9)

            echo "Bye!"

            exit 0

            ;;

        \*)

            echo "Invalid Number. Try Again"

            ;;

    esac

done

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

fun1() {

    echo -n "Please enter 'movie id'(1~1682) : "

    read movieId

    awk "NR == $movieId" $1

}

1) NR은 현재 탐색중인 행 번호를 저장하는 내부 변수이다

moviedId와 일치하면 출력한다

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

fun2() {

    echo -n "Do you want to get the data of ‘action’ genre movies  from '$1’?(y/n) :"

    read confirm

    if [ "$confirm" != 'y' ]; then

        return

    fi

    awk -F'|' '$7==1 {print $1, $2; cnt++} cnt == 10 {exit}' $1 | sort -k 1 -n

}

1) $7은 7번째 열로써 Genre의 액션장르 여부를 가리킨다.

2) 일치할 때마다 cnt를 1씩 증가시키고 10일 때 탐색을 종료한다.

3) sort -k 1 -n은 1번째 열을 기준(-k 1)으로 정수형(-n)으로 정렬한다는 코드이다.

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

fun3() {

    echo -n "Please enter the 'movie id’ (1~1682) :"

    read movieId

    awk -v movieId="$movieId" '$2 == movieId { sum += $3; cnt++} END { if (cnt > 0) printf "average rating of %d : %.5f\n", movieId, (sum/cnt) }' $1

}

1) -v movieId는 외부변수인 movieId를 내부 변수 movieId로 가져온다는 코드이다.

2) 두번째 열이 movieId 변수와 일치할 때마다 rate인 $3을 더한다. 또한 cnt를 1씩 증가시켜 개수도 카운팅한다.

3) END는 탐색이 모두 종료후 실행하는 코드를 작성하는 영역이다.

4) %.5f를 사용하여 소수점 여섯번째자리에서 반올림한다고 지정한다.

4. Delete the ‘IMDb URL’ from ‘u.item’

fun4() {

    echo -n "Do you want to delete the ‘IMDb URL’ from ‘$1’?(y/n):"

    read confirm

    if [ "$confirm" != 'y' ]; then

        return

    fi

    sed -n '1,10 s/|http.\*imdb.com.\*)|/||/p' $1

}

1) -n 1, 10/~/p는 1~10번째 행만 출력한다는 코드이다.

2) 정규표현식 |http.\*imdb.com.\*)|은 ‘|’ ‘|’과 그 사이에 문자열 http.\*imdb.com.\*)이 들어가는 문자열을 나타내는 정규표현식이다. 이때 \*은 와일드카드로써 어떤 문자든지 어떤 개수만큼 올수 있음을 나타낸다.

3) /||/은 앞선 정규표현식과 일치하는 문자열을 ||로 치환함을 나타낸다.

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

fun5() {

    echo -n "Do you want to get the data about users from ‘$1’?(y/n) :"

    read confirm

    if [ "$confirm" != 'y' ]; then

        return

    fi

    sed -n '1, 10 s/\([0-9]\*\)|\([0-9]\*\)|\([MF]\)|\([a-zA-Z]\*\)|\(.\*\)/user \1 is \2 years old \3 \4/p' $1 | sed 's/M/male/; s/F/female/'

}

1) -n 1, 10/~/p는 1~10번째 행만 출력한다는 코드이다.

2) 정규표현식 \([0-9]\*\)|\([0-9]\*\)|\([MF]\)|\([a-zA-Z]\*\)|\(.\*\)

(1) \([0-9]\*\) : 첫 번째 그룹을 설정한다. 이 그룹은 숫자 0~9가 임의의 수 만큼 나타나는 그룹이다.

(2) \([0-9]\*\) : 두 번째 그룹을 설정한다. 이 그룹은 숫자 0~9가 임의의 수 만큼 나타나는 그룹이다.

(3) \([MF]\) : 세 번째 그룹을 설정한다. 이 그룹은 알파벳 M또는 F가 한번 나타나는 그룹이다.

(4) \([a-zA-Z]\*\) : 네 번째 그룹을 설정한다. 이 그룹은 알파벳 대소문자가 임의의 수 만큼 나타나는 그룹이다.

(5) \(.\*\) : 다섯 번째 그룹을 설정한다. 이 그룹은 어떤 문자던지 임의의 수만큼 나타나는 그룹이다.

3) /user \1 is \2 years old \3 \4/ : 스트링과 더불어 \n 자리에 n번째 그룹을 넣는다.

4) | sed 's/M/male/; s/F/female/' : 파이프라이닝하여 앞선 결과 값을 전달하고, M은 male로 F는 female로 치환한다.

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

fun6() {

    echo -n "Do you want to Modify the format of ‘release data’ in ‘$1’?(y/n) :"

    read confirm

    if [ "$confirm" != 'y' ]; then

        return

    fi

    tail -n 10 $1 | sed 's/\([0-9]\*\)-\([a-zA-Z]\*\)-\([0-9]\*\)/\3\2\1/' | sed 's/Jan/01/; s/Feb/02/; s/Mar/03/; s/Apr/04/; s/May/05/; s/Jun/06/; s/Jul/07/; s/Aug/08/; s/Sep/09/; s/Oct/10/; s/Nov/11/; s/Dec/12/'

}

1) tail -n 10 : 마지막 10행만 파이프라이닝을 통하여 넘겨준다.

2) 정규표현식 \([0-9]\*\)-\([a-zA-Z]\*\)-\([0-9]\*\)

(1) \([0-9]\*\) : 첫 번째 그룹을 설정한다. 이 그룹은 숫자 0~9가 임의의 수 만큼 나타나는 그룹이다.

(2) \([a-zA-Z]\*\) : 두 번째 그룹을 설정한다. 이 그룹은 알파벳 대소문자가 임의의 수 만큼 나타나는 그룹이다.

(3) \([0-9]\*\) : 세 번째 그룹을 설정한다. 이 그룹은 숫자 0~9가 임의의 수 만큼 나타나는 그룹이다.

3) \3\2\1 : 3, 2, 1그룹 순서대로 출력한다.

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

: 파이프라이닝하여 앞선 결과값을 전달, 각 월 영단어에 대응하는 숫자로 치환한다.

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

fun7() {

    echo -n "Please enter the ‘user id’(1~943) :"

    read userId

    echo $userId

    awk -v userId="$userId" '{if($1 == userId && cnt < 10) {cnt++; print $2}}' $2 | sort -k 1 -n | awk '{cnt ++; elem[cnt] = $0} END {for(key in elem){if(key != cnt){printf "%s|", elem[key]} else{printf "%s\n\n", elem[key]}}}'

    awk -v userId="$userId" '{if(NR == FNR) {if($1 == userId && cnt < 10) {cnt++; data[$2]} next}} {FS = "|"} $1 in data {print $1,$2}' $2 $1 | sed 's/ /|/'

}

1) if($1 == userId && cnt < 10) {cnt++; print $2} : cnt를 증가시켜 행 10개만 출력한다.

2) | sort -k 1 -n : 파이프라이닝하여 앞선 결과를 전달, 첫번째 열 기준으로 정수 정렬한다.

3) | awk '{cnt ++; elem[cnt] = $0} ' 파이프라이닝하여 앞선 결과를 전달,

행 개수 카운팅을 위한 cnt를 1증가, elem배열에 현재 행을 추가한다.

4) END {for(key in elem){if(key != cnt){printf "%s|", elem[key]} else{printf "%s\n\n", elem[key]}}

(1) key의 값이 마지막 행 번호(cnt)와 일치하지않는다면 문자열| 형식으로 출력하고

(2)key의 값이 마지막 행 번호(cnt)와 일치하면 문자열\n\n형식으로 출력한다.

5) if(NR == FNR) {if($1 == userId && cnt < 10) {cnt++; data[$2]} next}}

(1) NR == FNR : NR은 전체기준 레코드 번호, FNR은 현재 파일 기준 레코드번호를 나타낸다.

따라서 NR == FNR이라는 것은 첫번째 파일만을 처리할 수 있는 조건이된다.

(2) if($1 == userId && cnt < 10) {cnt++; data[$2]} next} : 첫번째 행이 외부 변수 userId와 일치하고 아직 조건을 만족한지 10번째에 다다르지 않았다면 data배열에 두번째 열(movieId)을 기록한다. 조건에 만족하지 않는다면 next를 실행하여 현재 행을 건너뛰고 다음 행을 실행한다.

6) {FS = "|"} $1 in data {print $1,$2}' $2 $1

(1) NR != FNR 즉, 2번째 파일을 탐색할 때 실행되는 구문이다.

(2) FS = "|" : 파일 구분자를 ‘|’로 새롭게 지정한다.

(3) data배열에 첫번째 열(movieId)가 있다면, 1열(movied)와 2열(movieTitle)을 출력한다.

6) | sed 's/ /|/' : 파이프라이닝을 통해 앞선 결과값을 입력, 공백문자를 |로 치환한다.

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

fun8() {

    echo -n "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) :"

    read confirm

    if [ "$confirm" != 'y' ]; then

        return

    fi

    awk -F'|' '{if(NR == FNR) {if($4 == "programmer" && 20 <= $2 && $2 <= 29 ) userId[$1] } else {{FS = " "} for(loop in userId){if($1 in userId){sumArr[$2] += $3; cntArr[$2]++}}}} END{for(key in sumArr){if ((sumArr[key]/cntArr[key]) == int(sumArr[key]/cntArr[key])){printf "%d %d\n", key, (sumArr[key]/cntArr[key])}else{num = sprintf("%.5f", (sumArr[key]/cntArr[key])); printf "%d %g\n", key, num}}}' $2 $1

}

1) if($4 == "programmer" && 20 <= $2 && $2 <= 29 ) userId[$1] : 4열(occupation)이 “programmer”이고 2열(age)가 20 이상 29 이하라면 userId 배열에 1열(userId)를 기록한다.

2) else {{FS = " "} for(loop in userId){if($1 in userId){sumArr[$2] += $3; cntArr[$2]++}}}

(1) FS = " " : 2번째 파일에서는 구분자를 공백문자로 지정한다.

(2) for(loop in userId) : userId의 모든 원소들에 대해 반복문을 수행한다.

(3) if($1 in userId){sumArr[$2] += $3; cntArr[$2]++} : 1열(userId)가 userId배열에 존재한다면 해당하는 movie id를 key로하는 sumArr배열에 3열(rating)을 저장한다.

또한, movie id를 key로하는 cntArr배열 값을 1 증가시킨다.

3) END{for(key in sumArr){if((sumArr[key]/cntArr[key]) == int(sumArr[key]/cntArr[key])){printf "%d %d\n", key, (sumArr[key]/cntArr[key])}else{num = sprintf("%.5f", (sumArr[key]/cntArr[key])); printf "%d %g\n", key, num}}}

(1) if((sumArr[key]/cntArr[key]) == int(sumArr[key]/cntArr[key]) : 나눗셈의 몫과 그 나눗셈을 정수형으로 캐스팅했을 때 값이 같다면 이 몫은 나누어떨어진 것이므로 %d 형식으로 출력한다.

(2) else{num = sprintf("%.5f", (sumArr[key]/cntArr[key])); printf "%d %g\n", key, num} : 아니라면 소수점아래 5번째수까지 출력하는 float형식으로 변수 num에 저장하고 자리수에 맞게 알맞게 소수점아래 0을 절삭해주는 %g형태로 출력한다.

9. Exit

9)

    echo "Bye!"

    exit 0

    ;;

\*)

flag가 9인 case에는 종료한다.