CSLR51 – Database Management Systems Laboratory

#Session: 08 || Date: 03/10/2024

Viva Due: Q.No. 2 (03/10/2024) Moodle Due: 09/10/2024 at 11 PM

Develop C code to simulate the following tasks:

1. Simulate Select and Project commands using the command prompt with necessary arguments in a menu driven fashion. For integer attributes, choices are: greater, greater than equal to, less than, lesser than equal to, equals For string attributes, choices are: starting with, ending with, length of the characters, equals to, substring matching Input: Select: Filename.txt, A condition(s) to retrieve a tuple(s). Project: Filename.txt, A condition to retrieve a column. Note: You may use the code you developed in Q.No.1 in Session 2.

Ans:

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

#define max_count 10

typedef struct {
    char name[20];
    int age;
    char b_g[3];
    char phone_no[10];
} donor;

donor Donor[max_count];
int count=0;

void read_file(char* file_name)
{
    FILE* fp=fopen(file_name,"r");
    if(fp==NULL)
    {
        printf("Could not open file for reading");
        return;
    }
    fclose(fp);
}

void write_file(char * file_name)
```

```
{
   FILE * fp=fopen(file_name,"w");
   if(fp==NULL)
   {
      printf("Cant open");
   }
      for(int i=0;i<count;i++)
      {
         fprintf(fp,"%s,%d,%s,%s",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
         printf("%s,%d,%s,%s",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
         fprintf(fp,"\n");
      }

   fclose(fp);
}

void insert(char * file_name)
{
   if(count>max_count)
   {
      printf("Limit reached ");
      return;
   }
   printf("Enter name : ");
   scanf("%s",Donor[count].name);
   printf("Enter age : ");
   scanf("%d",&Donor[count].age);
   printf("Enter b_g : ");
   scanf("%s",Donor[count].b_g);
   printf("Enter ph_no : ");
   scanf("%s",Donor[count].phone_no);
   count++;
   write_file(file_name);
   return;
}

void proj(char * file_name, int column[4],int record[5])
{
   for(int i=0;i<count;i++)
   {
      for(int k=0;k<(count);k++)
      {
         if(i==record[k])
         {
```

```
            for(int j=0;j<4;j++)
            {
                if(column[j]==1)
                printf("%s  ",Donor[i].name);
                else if(column[j]==2)
                printf("%d  ",Donor[i].age);
                else if(column[j]==3)
                printf("%s  ",Donor[i].b_g);
                else if(column[j]==4)
                printf("%s  ",Donor[i].phone_no);
            }
            printf("\n");
            break;
          }
       }
   }
}

void start(char * file_name,char * col_name)
{
   char a[1];
   int j=strcmp(col_name,"name");
   if (j==0)
   {printf("Enter the first alphabet : ");
   scanf("%s",a);
   for(int i=0;i<count;i++)
   {
      if(Donor[i].name[0]==a[0])
      printf("%s,%d,%s,%s\n",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
   }
   }
   else
   {printf("Enter the first alphabet : ");
   scanf("%s",a);
   for(int i=0;i<count;i++)
   {
      if(Donor[i].b_g[0]==a[0])
      printf("%s,%d,%s,%s\n",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
   }
   }
}

void len(char * file_name,char * col_name)
{
```

```
    int a;
    int j=strcmp(col_name,"name");
    if(j==0)
    {
    printf("Enter the length : ");
    scanf("%d",&a);
    for(int i=0;i<count;i++)
    {
        if(strlen(Donor[i].name)==a)
        printf("%s,%d,%s,%s\n",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
    }
    }
    else
    {
    printf("Enter the length : ");
    scanf("%d",&a);
    for(int i=0;i<count;i++)
    {
        if(strlen(Donor[i].b_g)==a)
        printf("%s,%d,%s,%s\n",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
    }
    }
}


void end(char * file_name,char * col_name)
{
    char a[1];
    int j=strcmp(col_name,"name");
    if(j==0)
    {printf("Enter the last alphabet : ");
    scanf("%s",a);
    for(int i=0;i<count;i++)
    {
        int k=strlen(Donor[i].name)-1;
        if(Donor[i].name[k]==a[0])
        printf("%s,%d,%s,%s\n",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
    }
    }
    else
    {printf("Enter the last alphabet : ");
    scanf("%s",a);
    for(int i=0;i<count;i++)
    {
```

```
        int k=strlen(Donor[i].b_g)-1;
        if(Donor[i].b_g[k]==a[0])
        printf("%s,%d,%s,%s\n",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
    }
    }

}

void sub(char * file_name,char * col_name)
{
    char a[10];
    printf("Enter the substring : ");
    scanf("%s",a);
    int k=strlen(a);
    if(!strcmp(col_name,"name"))
    {
    for(int i=0;i<count;i++)
    {
        int l=strlen(Donor[i].name);
        for(int m=0;m<l-k+1;m++)
        {
            int y=0;
            for(int g=0;g<k;g++)
            {
                if(Donor[i].name[m+g]==a[g])
                y++;
            }
            if(y==k)
            printf("%s,%d,%s,%s\n",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
        }
    }
    }
    else
    {
    for(int i=0;i<count;i++)
    {
        int l=strlen(Donor[i].b_g);
        for(int m=0;m<l-k+1;m++)
        {
            int y=0;
            for(int g=0;g<k;g++)
            {
                if(Donor[i].b_g[m+g]==a[g])
                y++;
```

```
            }
            if(y==k)
            printf("%s,%d,%s,%s\n",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
        }
    }
    }

}

void gte(char * file_name,char * col_name)
{
    int a;
    printf("Enter the value: ");
    scanf("%d",&a);
    if(strcmp(col_name,"age")==0)
    {
        for(int i=0;i<count;i++)
        {
            if(Donor[i].age>=a)
            printf("%s,%d,%s,%s\n",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
        }
    }
}

void gt(char * file_name,char * col_name)
{
    int a;
    printf("Enter the value: ");
    scanf("%d",&a);
    if(strcmp(col_name,"age")==0)
    {
        for(int i=0;i<count;i++)
        {
            if(Donor[i].age>a)
            printf("%s,%d,%s,%s\n",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
        }
    }
}

void lte(char * file_name,char * col_name)
{
    int a;
    printf("Enter the value: ");
    scanf("%d",&a);
```

```
    if(strcmp(col_name,"age")==0)
    {
        for(int i=0;i<count;i++)
        {
            if(Donor[i].age<=a)
            printf("%s,%d,%s,%s\n",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
        }
    }
}

void lt(char * file_name,char * col_name)
{
    int a;
    printf("Enter the value: ");
    scanf("%d",&a);
    if(strcmp(col_name,"age")==0)
    {
        for(int i=0;i<count;i++)
        {
            if(Donor[i].age<a)
            printf("%s,%d,%s,%s\n",Donor[i].name,Donor[i].age,Donor[i].b_g,Donor[i].phone_no);
        }
    }
}

void eq(char * file_name,char * col_name)
{
    int a;
    printf("Enter the value: ");
    scanf("%d",&a);
    int b;
    printf("\nDo you want to project specfic columns?? ");
    scanf("%d",&b);
    int arr[4];
    int arr1[count];
    int j=0;
    if(strcmp(col_name,"age")==0)
    {
        for(int i=0;i<count;i++)
        {
            if(Donor[i].age==a)
            {
                arr1[j++]=i;
            }
```

```
        }
    }
    if(b)
    {
        printf("\nEnter column numbers:\n");
        for(int i=0;i<4;i++)
        {
            printf("Enter %d:",i);
            scanf("%d",&arr[i]);
        }
        proj(file_name,arr,arr1);
    }
    return;
}

void int_op(char * file_name,char * col_name)
{
    int choice;
    printf("Menu:\n");
    printf("1.>=\n");
    printf("2.>\n");
    printf("3.<=\n");
    printf("4.<\n");
    printf("5.=\n");
    scanf("%d",&choice);

    switch(choice)
    {
        case 1:
        {
            gte(file_name,col_name);
            break;
        }
        case 2:
        {
            gt(file_name,col_name);
            break;
        }
        case 3:
        {
            lte(file_name,col_name);
            break;
        }
        case 4:
```

```
        {
            lt(file_name,col_name);
            break;
        }
        case 5:
        {
            eq(file_name,col_name);
            break;
        }
    }
    return;
}

void char_op(char * file_name,char * col_name)
{
    int choice;
    printf("Menu:\n");
    printf("1.Start with\n");
    printf("2.End with\n");
    printf("3.With length\n");
    printf("4.With substring\n");
    scanf("%d",&choice);

    switch(choice)
    {
        case 1:
        {
            start(file_name,col_name);
            break;
        }
        case 2:
        {
            end(file_name,col_name);
            break;
        }
        case 3:
        {
            len(file_name,col_name);
            break;
        }
        case 4:
        {
            sub(file_name,col_name);
            break;
```

```
        }
    }
    return;
}

void selproj(char * file_name)
{
    int que;
    printf("Do you want to select?(1)");
    scanf("%d",&que);

    if(que==0)
    {
        printf("Enter the column number:\n");
        int arr[4];
        for(int i=0;i<4;i++)
        {
            printf("Enter %d: ",i);
            scanf("%d",&arr[i]);
        }
        int  arr1[count];
        for(int i=0;i<count;i++)
        {
            arr1[i]=i;
        }
        proj(file_name,arr,arr1);
    }

    else
    {
    char col_name[10];
    int type;
    printf("Enter column name : ");
    scanf("%s",col_name);
    printf("Enter column_type: 2 for int 1 for char: ");
    scanf("%d",&type);
    if(type==1)
    {
        char_op(file_name,col_name);
    }
    else if(type==2)
    {
        int_op(file_name,col_name);
    }
```

```
    }
}

int main()
{
    int choice;
    char file_name[50];
    printf("Enter file name: ");
    scanf("%s",file_name);
    read_file(file_name);
    while(1)
    {
        printf("\nMenu\n");
        printf("1.Insert donor record\n");
        printf("2.Project and select operations\n");
        printf("3.Exit\n");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1:
            {
                insert(file_name);
                break;
            }
            case 2:
            {
                selproj(file_name);
                break;
            }
            case 3:
            {
                exit(0);
            }
            default:
            printf("Invalid choice");
        }
    }
    return 0;
}
```

Output:

```
reethi@DESKTOP-8744EFO:~/dir1/dbms$ ./a.out
Enter file name: t1.txt
```

Menu
1.Insert donor record
2.Project and select operations
3.Exit
1
Enter name : Reethikka
Enter age : 20
Enter b_g : o+
Enter ph_no : 12345
Reethikka,20,o+,12345
Menu
1.Insert donor record
2.Project and select operations
3.Exit
1
Enter name : SriGanesh
Enter age : 17
Enter b_g : o+
Enter ph_no : 67890
Reethikka,20,o+,12345SriGanesh,17,o+,67890
Menu
1.Insert donor record
2.Project and select operations
3.Exit
1
Enter name : Priya
Enter age : 45
Enter b_g : o-
Enter ph_no : 234567
Reethikka,20,o+,12345SriGanesh,17,o+,67890Priya,45,o-,234567
Menu
1.Insert donor record
2.Project and select operations
3.Exit
2
Do you want to select?(1)0
Enter the column number:
Enter 0: 1
Enter 1: 3
Enter 2: 0
Enter 3: 0
Reethikka o+
SriGanesh o+

```
Priya  o-

Menu
1.Insert donor record
2.Project and select operations
3.Exit
2
Do you want to select?(1)1
Enter column name : age
Enter column_type: 2 for int 1 for char: 2
Menu:
1.>=
2.>
3.<=
4.<
5.=
5
Enter the value: 20

Do you want to project specfic columns?? 1

Enter column numbers:
Enter 0:1
Enter 1:3
Enter 2:2
Enter 3:0
Reethikka  o+  20

Menu
1.Insert donor record
2.Project and select operations
3.Exit
1
Enter name : Rakshana
Enter age : 21
Enter b_g : ab+
Enter ph_no : 890
Reethikka,20,o+,12345SriGanesh,17,o+,67890Priya,45,o-,234567Rakshana,21,ab+890,890
Menu
1.Insert donor record
2.Project and select operations
3.Exit
2
Do you want to select?(1)1
```

```
Enter column name : age
Enter column_type: 2 for int 1 for char: 2
Menu:
1.>=
2.>
3.<=
4.<
5.=
5
Enter the value: 20

Do you want to project specfic columns?? 1

Enter column numbers:
Enter 0:1
Enter 1:3
Enter 2:0
Enter 3:0
Reethikka  o+

Menu
1.Insert donor record
2.Project and select operations
3.Exit
1
Enter name : Sheetal
Enter age : 20
Enter b_g : o
Enter ph_no : 123
Reethikka,20,o+,12345SriGanesh,17,o+,67890Priya,45,o-,234567Rakshana,21,ab+890,890Sh
eetal,20,o,123
Menu
1.Insert donor record
2.Project and select operations
3.Exit
2
Do you want to select?(1)1
Enter column name : age
Enter column_type: 2 for int 1 for char: 2
Menu:
1.>=
2.>
3.<=
4.<
```

5.=
5
Enter the value: 20

Do you want to project specfic columns?? 1

Enter column numbers:
Enter 0:1
Enter 1:3
Enter 2:0
Enter 3:0
Reethikka o+
SriGanesh o+
Sheetal o

Menu
1.Insert donor record
2.Project and select operations
3.Exit
3

2. Develop an implementation package that would contribute to a normalization setup by generating the Candidate key(s) and Super key(s) in a Relation given the Functional Dependencies. Your code should work for any given FD's, not just for the given sample below.
Example: Given R(X Y Z W) and FD = { XYZ → W, XY → ZW and X → YZW} Candidate key: {X}; Super keys: {X, XY, XZ, XW, XYZ, XYW, XZW, XYZW}
Given R(X Y Z W) and FD = {X→Y, Y→Z, Z→X} Candidate keys: {WX, WY, WZ}; Super keys: {WXY, WXZ, WYZ, WXYZ}

Ans:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_ATTR 20
#define MAX_FD 20
#define MAX_LEN 100

char attributes[MAX_ATTR][MAX_LEN];  // List of attributes
char fds[MAX_FD][MAX_LEN];           // List of functional dependencies
```

```
int numAttrs, numFDs;

void readInput() {
    printf("Enter number of attributes: ");
    scanf("%d", &numAttrs);
    printf("Enter the attributes (space-separated): ");
    for (int i = 0; i < numAttrs; i++) {
        scanf("%s", attributes[i]);
    }

    printf("Enter number of functional dependencies: ");
    scanf("%d", &numFDs);
    printf("Enter the functional dependencies (in the form A,B->C,D):\n");
    for (int i = 0; i < numFDs; i++) {
        scanf("%s", fds[i]);
    }
}

int isSubset(const char *subset, const char *set) {
    for (int i = 0; subset[i] != '\0'; i++) {
        if (strchr(set, subset[i]) == NULL) {
            return 0;
        }
    }
    return 1;
}


void closure(const char *attributesSet, char *result) {
    strcpy(result, attributesSet);
    int changed;

    do {
        changed = 0;
        for (int i = 0; i < numFDs; i++) {
            char lhs[MAX_LEN], rhs[MAX_LEN];
            sscanf(fds[i], "%[^->]->%s", lhs, rhs);

            // Check if LHS is a subset of current closure
            if (isSubset(lhs, result)) {
                for (int j = 0; rhs[j] != '\0'; j++) {
                    if (rhs[j] != ',' && !strchr(result, rhs[j])) { // Ignore commas
                        strncat(result, &rhs[j], 1); // Append new attributes
                        changed = 1;
```

```
                }
            }
        }
    }
  } while (changed);
}

void generateSuperKeys(char * c_key) {
  int totalKeys = 1 << numAttrs; // 2^numAttrs
  printf("\nSuper Keys:\n");
  for (int i = 1; i < totalKeys; i++) {
    char key[MAX_LEN] = "";
    for (int j = 0; j < numAttrs; j++) {
      if (i & (1 << j)) {
        strncat(key, attributes[j], 1);
      }
    }

    char closureSet[MAX_LEN] = "";
    if (isSubset(c_key,key)) {
      printf("%s\n", key);
    }
  }
}


int isCandidateKey(const char *key) {
  char closureSet[MAX_LEN] = "";
  closure(key, closureSet);
  return strlen(closureSet) == numAttrs;
}

void generateCandidateKeys() {
  int totalKeys = 1 << numAttrs; // 2^numAttrs
  for (int i = 1; i < totalKeys; i++) {
    char key[MAX_LEN] = "";
    for (int j = 0; j < numAttrs; j++) {
      if (i & (1 << j)) {
        strncat(key, attributes[j], 1);
      }
    }

    if (isCandidateKey(key)) {
      // Ensure minimality: check if the key can be reduced
```

```
            int isMinimal = 1;
            for (int k = 0; k < strlen(key); k++) {
                char subkey[MAX_LEN];
                strncpy(subkey, key, k);
                subkey[k] = '\0';
                strncat(subkey, key + k + 1, strlen(key) - k - 1);

                if (isCandidateKey(subkey)) {
                    isMinimal = 0; // Found a smaller candidate key
                    break;
                }
            }

            if (isMinimal) {
                printf("\nCandidate Key:\n");
                printf("%s\n", key);
                generateSuperKeys(key);
            }
        }
    }
}


int main() {
    readInput();
    generateCandidateKeys();
    //generateSuperKeys();
    return 0;
}
```

Output:

```
reethi@DESKTOP-8744EFO:~/dir1/dbms$ g++ test1.c
reethi@DESKTOP-8744EFO:~/dir1/dbms$ ./a.out
Enter number of attributes: 4
Enter the attributes (space-separated): w x y z
Enter number of functional dependencies: 3
Enter the functional dependencies (in the form A,B->C,D):
x,y,z->w
x,y->z,w
x->y,z,w

Candidate Key:
x
```

Super Keys:
x
wx
xy
wxy
xz
wxz
xyz
wxyz
reethi@DESKTOP-8744EFO:~/dir1/dbms$ g++ test1.c
reethi@DESKTOP-8744EFO:~/dir1/dbms$ ./a.out
Enter number of attributes: 4
Enter the attributes (space-separated): x y z w
Enter number of functional dependencies: 3
Enter the functional dependencies (in the form A,B->C,D):
x->y
y->z
z->x

Candidate Key:
xw

Super Keys:
xw
xyw
xzw
xyzw

Candidate Key:
yw

Super Keys:
yw
xyw
yzw
xyzw

Candidate Key:
zw

Super Keys:
zw
xzw

yzw
xyzw