EXPERIMENT-6

AIM: Implementation & attribute donne algorithm.

THEORY: In database management systems, attribute closure is an important concept related to functional dependencies. The attribute closure of a set of attributes is the set of all attributes that can be determined by those attributes through a set of functional dependencies

A functional dependency is a contraint between two sure of attributes in a relation. It states that if two tupler have the some value for the first set of attributes, they must also have. The same value for the second set of attributes. For example, if we have a relation R (A,B,C) with the functional dependency A > B, it means that if two tuples have the same value for B. They must also have the same value for B.

The attribute closure of a set of attributes is important because it can help us determine the superkeys and condidate keys of a relation. A superkey is a set of attributes that can uniquely identify a tupel in a relation, while a condidate key is minimal superkey (i.e., a superkey that centains

no unnecessary attributer). The attribute desure of a set of attributes can help us determine if that set of attributes is a superky or a condidate key.

To find the attribute closure of a set of attributes, we need to apply the functional dependencies to that set of attributes repeateadly until we can no too lenger add any new attributes. For example, if we have a relation R (A,B,C,D) with the functional dependencies A > B and B > C, the attribute closure of A would be (A,B,C).

In summary, the attribute closure of a set of attributes is the few of all attributes that can be determined by those attributes through a set of functional dependencies. It is an important concept in database management systems because it can help us determine the superkeys and condidate peys of a relation.

ALGORITHM:

Step 1: Initialize the closure of A as A+ = A

Step 2: Repeat until no more attributes can be added

to A+;

a. For each functional dependency X-y:nF; i. 9/ X is a subset of A+, Then add y to A+3

Stcp3: Return A+

distribution of the second	
	Example: He are given the relation R (A,B,C,D,E).
	This means that the table R has fire columns.
	A.B, C, D and E. We are also given the set of functional
	A.B, C, D and E. We are also given the set of functional dependencies: (A + B, B + C, C + D, D -> E 3.
	To find attribute of A or find A+.
	· First, we add A to (A)+
	· We have A -B, so we can determine B. Therefore
	At is now (A,B)
5 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	· We have B->c , so we can determine c. Therefore
	IAIT is now (A.B.c).
	· Nour, we have A,B,C,D and D->E, so we can
	was I to (A)
	· Not , we have usedall of the clumps in R and we
	han all used all functional dependencies. [A] += [A,B,C,D,E].
<u> </u>	
211	
)	
ac with	

Attribute Closure Code (Java):

```
import java.util.*;
 class prac6aashish{
    public static void main(String[] args) {
        // char[] R={'A','B','C','D','E'}; //Relations
        String[][]
FD={{"A","BC"},{"CD","E"},{"B","D"},{"E","A"}}; //functional dependencies
        String[] closure={"A","B","E","C","D","BC","BD"};
        int i,j,k,l,count=0,flag=-1;
        for(i=0;i<closure.length;i++)</pre>
            Set<Character> AC=new HashSet<Character>();
            for(char c: closure[i].toCharArray())
                 AC.add(c);
            for(j=0;j<FD.length;j++)</pre>
                 ArrayList<Character> A=new ArrayList<Character>(AC);
                for(k=0;k<FD[j][0].length();k++)</pre>
                 {
                     for(int m=0;m<A.size();m++)</pre>
                         if(FD[j][0].charAt(k)==A.get(m))
                         {
                             flag=0;
                             break;
                         else if(m==A.size()-1)
                         {
                             flag=1;
                             break;
                         }
                     if(flag==1)
                     break;
                 if(flag==0)
                 {
                     int a=A.size();
                     for(l=0;1<FD[j][1].length();1++)</pre>
                         AC.add(FD[j][1].charAt(1));
                     int b=AC.size();
                     if(a==b)
                     {
                         count++;
                     }
                 }
```

Output:

```
PS D:\GCOEN SEM 4\Database Management System L
Attribute closure of A is [A, B, C, D, E]

Attribute closure of B is [B, D]

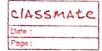
Attribute closure of E is [A, B, C, D, E]

Attribute closure of C is [C]

Attribute closure of D is [D]

Attribute closure of BC is [A, B, C, D, E]

Attribute closure of BD is [B, D]
```



	Page:
	Conclusion: Here we have used a relation R= [ABC DE] 4 [+: 1] 60
	R= {A,B,C,D,E} and 4 functional dependencies FD, Where FD= {A→BC, CD→E, B→D, E→A} and
	and BD and successfully implemented the algorithm
	of attribute desure.
A state of the	
800 74	
*	
Sa Film	