ST. XAVIER’S COLLEGE

(Affiliated to Tribhuvan University)

Maitighar, Kathmandu



**DBMS**

**THEORY ASSIGNMENT # 10**

Submitted by:

Anlisha Maharjan  
013BSCCSIT006

Submitted to:

|  |  |
| --- | --- |
| Er. Sanjay K. Yadav  Lecturer Department of Computer Science |  |

Date of Submission: October 8, 2015

**FUNCTIONAL DEPENDENCIES**

**BASIC CONCEPTS:**

Functional dependency is a relationship that exists when one attribute uniquely determines another attribute.

If R is a relation with attributes X and Y, a functional dependency between the attributes is represented as X->Y, which specifies Y is functionally dependent on X. Here X is a determinant set and Y is a dependent attribute. Each value of X is associated precisely with one Y value.

A functional dependency occurs when one attribute in a relation uniquely determines another attribute. This can be written A -> B which would be the same as stating "B is functionally dependent upon A." In relational database theory, a functional dependency is a constraint between two sets of attributes in a relation from a database.

Given a relation R, a set of attributes X in R is said to functionally determine another set of attributes Y, also in R, (written X → Y) if, and only if, each X value is associated with precisely one Y value; R is then said to satisfy the functional dependency X → Y. Equivalently, the projection \pi\_{X,Y}R is a function, i.e. Y is a function of X.[1][2] In simple words, if the values for the X attributes are known (say they are x), then the values for the Y attributes corresponding to x can be determined by looking them up in any tuple of R containing x. Customarily X is called the determinant set and Y the dependent set. A functional dependency FD: X → Y is called trivial if Y is a subset of X.

Functional dependency in a database serves as a constraint between two sets of attributes. Defining functional dependency is an important part of relational database design and contributes to aspect normalization.

**CLOSURE OF A SET OF FUNCTIONAL DEPENDENCIES:**

We need to consider all functional dependencies that hold. Given a set F of functional dependencies, we can prove that certain other ones also hold. We say these ones are logically implied by F.

* 1. Suppose we are given a relation scheme R=(A,B,C,G,H,I), and the set of functional dependencies:

A tex2html_wrap_inline1090 B

A tex2html_wrap_inline1090 C

CG tex2html_wrap_inline1090 H

CG tex2html_wrap_inline1090 I

B tex2html_wrap_inline1090 H

Then the functional dependency tex2html_wrap_inline1194 is logically implied.

* 1. To see why, let t1 and t2 be tuples such that

tex2html_wrap_inline1200

As we are given A tex2html_wrap_inline1090 B , it follows that we must also have

tex2html_wrap_inline1204

Further, since we also have B tex2html_wrap_inline1090 H , we must also have

tex2html_wrap_inline1208

Thus, whenever two tuples have the same value on A, they must also have the same value on H, and we can say that A tex2html_wrap_inline1090 H .

* 1. The closure of a set F of functional dependencies is the set of all functional dependencies logically implied by F.
  2. We denote the closure of F by tex2html_wrap_inline1222 .
  3. To compute tex2html_wrap_inline1222 , we can use some rules of inference called Armstrong's

Axioms:

* + - Reflexivity rule: if tex2html_wrap_inline958 is a set of attributes and tex2html_wrap_inline1158 , then tex2html_wrap_inline1058 holds.
    - Augmentation rule: if tex2html_wrap_inline1058 holds, and tex2html_wrap_inline1234 is a set of attributes, then tex2html_wrap_inline1236 holds.
    - Transitivity rule: if tex2html_wrap_inline1058 holds, and tex2html_wrap_inline1240 holds, then tex2html_wrap_inline1242 holds.

**CLOSURE OF ATTRIBUTE SETS:**

1. To test whether a set of attributes tex2html_wrap_inline958 is a superkey, we need to find the set of attributes functionally determined by tex2html_wrap_inline958 .
2. Let tex2html_wrap_inline958 be a set of attributes. We call the set of attributes determined by tex2html_wrap_inline958 under a set F of functional dependencies the closure of tex2html_wrap_inline958 under F, denoted tex2html_wrap_inline1292 .
3. The following algorithm computes tex2html_wrap_inline1292 :

result := tex2html_wrap_inline958

tex2html_wrap_inline1240 while (changes to result) do

for each functional dependency

in F do

tex2html_wrap_inline1302 begin

if result

then result := result tex2html_wrap_inline1304 ;

end

1. If we use this algorithm on our example to calculate tex2html_wrap_inline1306 then we find:

* We start with result = AG.
* A tex2html_wrap_inline1090 B causes us to include B in result.
* A tex2html_wrap_inline1090 C causes result to become ABCG.
* CG tex2html_wrap_inline1090 H causes result to become ABCGH.
* CG tex2html_wrap_inline1090 I causes result to become ABCGHI.
* The next time we execute the while loop, no new attributes are added, and the algorithm terminates.

1. This algorithm has worst case behavior quadratic in the size of F. There is a linear algorithm that is more complicated.

**DECOMPOSITION**

**LOSSLESS-JOIN DECOMPOSITION**

Lossless Join Decomposition can also be called Non-additive. If you decompose a relation R into relations R\_1 and R\_2 you will guarantee a Lossless-Join if R\_1⋈R\_2 = R.

If R is split into R1 and R2, for the decomposition to be lossless then at least one of the two should hold true. Projecting on R1 and R2, and joining back, results in the relation you started with. Let R be a relation schema. Let F be a set of functional dependencies on R. Let R\_1 and R\_2 form a decomposition of R. The decomposition is a lossless-join decomposition of R if at least one of the following functional dependencies are in F+ (where F+ stands for the closure for every attribute or attribute sets in F).

**DEPENDENCY PRESERVATION**

Functional dependencies in a restriction can be tested in one relation, as they involve attributes in one relation schema. The set of restrictions tex2html\_wrap\_inline1660 is the set of dependencies that can be checked efficiently.

If this is so, then every functional dependency in F is implied by F', and if F' is satisfied, then F must also be satisfied.

A decomposition having the property that tex2html\_wrap\_inline1670 is a dependency-preserving decomposition.