Tianyu Zhang

UID: 805332081

1.SuppliersBasic(SUPPLIER) <- part\_cost(\_,SUPPLIER,\_,\_), ~Missingpart(SUPPLIER).

Missingpart(SUPPLIER) <- Part\_cost(BASIC\_PART,\_,\_,\_), ~Haspart(BASIC\_PART,SUPPLIER).

Haspart(BASIC\_PART,SUPPLIER)<- part\_cost(BASIC\_PART,SUPPLIER,\_,\_).

2. Student\_Satisfy(Name) <- took(Name,Course1,\_),

took(Name,Course2,\_),Course1~=Course2,~NotHighest(Name).

NotHighest(Name)<- took(Name1,Course,G1),took(Name2,Course,G2),Name1~=Name2, G1<G2.

5.

a. To show that the power does not change, we need to prove that the intersection can still be accomplished with other operations. The intersection of the two sets basically means the elements that have in common. What we can do is that, we find out the the difference of A and B first. Name it A-B, then we know that A-B means the elements that are in A but not in B. Reversely, we will know that except A-B, others are in A and also in B. Therefore, A-(A-B) will work like intersection.

b. 1) set union 2) set intersection 3)Cartesian Product 4)selection 5)projection

c. To show that the power loss if we remove set difference, we need to know that set difference is not a monotonic operator and all other operations are monotonic. Since we know the power will not change if and only if the operation could be replaced by other operations, the set difference can not be replaced

7.

a)first we need to take a look at the body of the rule, the variable X is contained in b1 and b2. b1 and b2 are positive base predicate, the body is safe, then the rule is safe.

Convert the rule to relational algebra

Original ->

Bodyr = σ$1=$2,$3=a,$4>$1(b1xb2) ->

S = π$4,$4Bodyr

b)Similarly, since the variables are put in safe rules p as proved before, since variable is safe, then the body is safe, then the whole rule is safe.

Original->

Bodyr = σ$2=$3(pxp)->

S = π$1,$4Bodyr

c) since the variable is contained in a negative goal, then the whole rule is not safe.

DeAL

HW1.fac

%Facts

%-----

% \* Facts corresponds to tuples in the database.

% \* Variables start with capital letters.

% \* Constants that begin with a capital letter are enclosed in single quotes.

% \* city('Houston', 'Texas', 1630000) corresponds to the row

% ('Houston', 'Texas', 1630000) in the city table.

% city(Name:string, State:string, Population:integer)

city('Houston', 'Texas', 3000000).

city('Dallas', 'Texas', 2000000).

city('Huntsville', 'Texas', 150000).

city('Austin', 'Texas', 750000).

city('Corsicana', 'Texas', 60000).

city('Shreveport', 'Louisiana', 90000).

city('Bastrop', 'Texas', 6000).

city('San Antonio', 'Texas', 1500000).

% distance(City1:string, City2:string, Distance:float)

distance('Houston', 'Bastrop', 130.0).

distance('Houston', 'Huntsville', 60.0).

distance('Huntsville', 'Dallas', 100.0).

distance('Austin', 'Waco', 110.0).

distance('Waco', 'Dallas', 100.0).

distance('Dallas', 'Shreveport', 200.0).

distance('Austin', 'Bastrop', 30.0).

distance('Austin', 'San Antonio', 80.0).

distance('San Antonio', 'Houston', 190.0).

HW1.Deal

%schema

database( {city(Name:string, State:string, Population:integer), distance(City1:string, City2:string,

Distance:float) }).

%Q1

findcity(City,Distance) <- distance("Austin",City,Distance).

% Q2

findfurthest(City,Distance) <- findcity(City,Distance), ~closer(City,Distance).

closer(City2,Distance2) <- distance("Austin", City1, Distance1), distance("Austin", City2,

Distance2), Distance2 < Distance1.

% Q3

texas\_pop(sum<Pop>) <- city( \_ , "Texas", Pop).

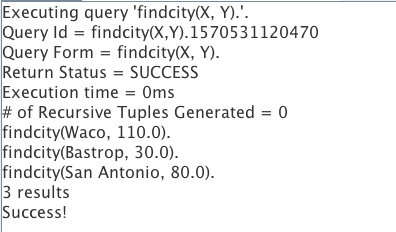
export findcity(X,Y).

export findfurthest(X,Y).

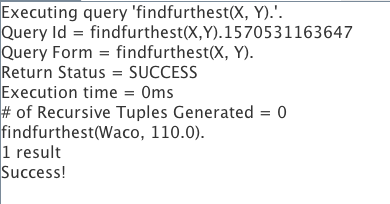
export texas\_pop(X).

Output:

Q1:



Q2:



Q3:

