--COMP 9311 Assgnment3 Written by <<Xinjie Lu>> Student ID: <<5101488>>

1.[16 marks] Consider a relation R (A, B, C, D, E, F). For each of the following sets of functional dependencies (i.e., i. to iv.), assuming that those are the only dependencies that hold for R, do the following:

a. List all of the candidate keys for R.

b. What are the BCNF violations, if any?

c. Decompose the relation, as necessary, into collections of BCNF relations?

i. AD → B, C → D, BC → A, B → D

a. candidate keys: ACEF, BCEF

b. The FD AD → B violates BCNF (AD doesn’t contain a key)
The FD C → D violates BCNF (C doesn’t contain a key)
The FD BC → A violates BCNF (BC doesn’t contain a key)
The FD B → D violates BCNF (B doesn’t contain a key)

c. We start from a schema: ABCDEF
The FD C → D violates BCNF.
To fix, we need to decompose into tables: CD and ABCEF.
FDs for CD are {C → D}, therefore key is C, therefore BCNF.
FDs for ABCEF are {BC → A}.
Key for ABCEF is BCEF, and FD BC → A violates BCNF.
To fix, we need to decompose into tables: ABC and BCEF.
FDs for ABC are {BC → A}, therefore key is BC, therefore BCNF.
FDs for BCEF are { }, so key is BCEF and table is BCNF
Final schema: **C**D, A**BC**, **BCEF**.

ii. BC → E, C → AB, AF → CD

a. candidate keys: AF, CF

b. The FD BC → E violates BCNF (BC doesn’t contain a key)
The FD C → AB violates BCNF (C doesn’t contain a key)

c. We start from a schema: ABCDEF
The FD BC → E violates BCNF.
To fix, we need to decompose into tables: BCE and ABCDF.
FDs for BCE are {BC → E}, therefore key is BC, therefore BCNF.
FDs for ABCDF are {C → AB, AF → CD}
Key for ABCDF is AF, and FD C → AB violates BCNF
To fix, we need to decompose into tables: ABC and CDF.
FDs for ABC are {C → AB}, therefore key is C, therefore BCNF.
FDs for CDF are { }, so key is CDF and table is BCNF.
Final schema: **BC**E, AB**C**, **CDF**

iii. ABF → D, CD → E, BD → A

a. candidate keys: ABCF, BCDF

b. The FD ABF → D violates BCNF (ABF doesn’t contain a key)
The FD CD → E violates BCNF (CD doesn’t contain a key)
The FD BD → A violates BCNF (BD doesn’t contain a key)

c. We start from a schema: ABCDEF

The FD ABF → D violates BCNF.

To fix, we need to decompose into tables: ABFD and ABFCE.

FDs for ABFD are {ABF → D}, therefore key is ABF, therefore BCNF.

FDs for ABCEF are { }, so key is ABCEF and table is BCNF.

Final schema: **AB**D**F**, **ABCEF**

iv. AB → D, BCD → EF, B → C

a. candidate keys: AB

b. The FD BCD → EF violates BCNF (BCD doesn't contain a key)

   The FD B → C violates BCNF (B doesn't contain a key)

c. We start from a schema: ABCDEF.

   The FD BCD → EF violates BCNF.

   To fix, we need to decompose into tables: BCDEF and ABCD.

   FDs for BCDEF are {BCD → EF}, therefore key is BCD, therefore BCNF.

   FDs for ABCD are {AB → D, B → C}.

   Key for ABCD is AB, and FD B → C violates BCNF.

   To fix, we need to decompose into tables: BC and ABD.

   FDs for BC are {B → C}, therefore key is B, therefore BCNF.

   FDs for ABD are {AB → D}, therefore key is AB, therefore BCNF.

   Final schema: **BCD**EF, **B**C, **AB**D

2. [12 marks] Assuming the schema from assignment 2 (i.e., the ASX database), give the following queries in relational algebra.

i. List all the company names that are in the sector of "Technology".

SectorCode = Proj[code](Sel[sector = 'Technology'](category))

Answer = Proj[name](SectorCode join Company)

ii. List all the company codes that have more than five executive members on record (i.e., at least six).

Temp1 = GroupBy[code, count[person]](executive)

Temp2 = Rename[1->code,2->count](Temp1)

Temp3 = Sel[count > 5](Temp2)

Answer = Proj[code](Temp3)

iii. Output the person names of the executives that are affiliated with more than one company.

Temp1 = GroupBy[person, count[code]](executive)

Temp2 = Rename[1->code,2->count](Temp1)

Temp3 = Sel[count > 1(Temp2)

Answer = Proj[person](Temp3)

iv. List all the companies (by their Code) that are the only one in their Industry (i.e., no competitors). Same as Assignment 2, please include both Code and Industry in the output.

Temp1 = GroupBy[industry,count[code]](category)

Temp2 = Rename[1->Industry,2->count](Temp1)

Temp3 = Sel[count = 1](Temp2)

Temp4 = Proj [industry](Temp3)

Answer=Proj[code, industry](category join Temp4)

3.[9 marks] Suppose relations R, S and T have r tuples, s tuples and t tuples, respectively. Derive the minimum and maximum numbers of tuples that the results of the following expressions can have.

i. R UNION (S INTERSECT T).

Temp = S INTERSECT T

Min size: 0 (when $S \cap T = \varnothing$)

Max size: s (when s < t and $S \subset T$) or t (when t< s and $T \subset S$)

R UNION Temp

Min size: r (when $S \cap T = \varnothing$ or Temp $\subset$ R)

Max size: r + s (when R $\cap$ Temp=$\varnothing$ and s < t and $S \subset T$) or
r + t (when R $\cap$ Temp=$\varnothing$ and t< s and $T \subset S$)

ii. SEL[c](R × S), for some condition c.

Temp = R × S

Min size: r * s

Max size: r * s

SEL[c]Temp

Min size:0 (no matches)

Max size: r * s (all match)

iii. R - PROJ[a](R JOIN S), for some list of attributes a.

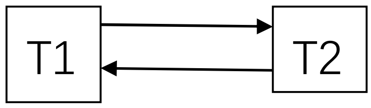
Min size: 0 (when R$\cap$ PROJ[a](R JOIN S)= PROJ[a](R JOIN S))

Max size: r (when R$\cap$ PROJ[a](R JOIN S)= $\varnothing$)

4. [8 marks]

i. For the following execution schedule, construct its precedence graph. Is this schedule serialisable? Explain your answer.

T1:R(X) T2:R(X) T1:W(X) T2:W(X) T2:R(Y) T1:R(Y) T1:W(Y) T2:W(X)



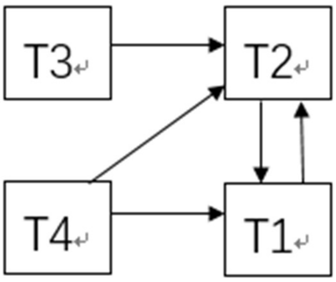
The precedence graph has an edge, from T1 to T2, because of the conflict between T1:R(X) and T2:W(X) and the conflict between T1:W(X) and T2:W(X) .

It also has an edge, from T2 to T1, because of the conflict between T2:R(X) and T1:W(X) and the conflict between T2:R(Y) and T1:W(Y).

The graph has a cycle, so the schedule is not serializable.

ii. For the following execution schedule, construct its precedence graph. Is this schedule serialisable? Explain your answer.

T3:R(X) T4:W(Y) T4:W(Z) T1:W(Y) T2:R(Y) T3:R(D) T2:W(X) T1:R(X)



The precedence graph has an edge, from T3 to T2, because of the conflict between T3:R(X) and T2:W(X).

It also has an edge, from T4 to T1, because of the conflict between T4:W(Y) and T1:W(Y).

It also has an edge, from T4 to T2, because of the conflict between T4:W(Y) and T2:R(Y).

It also has an edge, from T1 to T2, because of the conflict between T1:W(Y) and T2:R(Y).

It also has an edge, from T2 to T1, because of the conflict between T2:W(X) and T2:R(X).

The graph has a cycle, so the schedule is not serializable.