COMP9311: Assignment 3

Changfeng Li (z5137858)

UNSW
School of Computer Science and Engineering
Sydney, Australia

6/10/2017

Question 1

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
ii. BC->E, C->AB, AF->CD
iii. ABF->D, CD->E, BD->A
iv. AB->D, BCD->EF, B->C

Q1(i)

Solution:
(a) 1.

$F = \{AD- > B, C- > D, BC- > A, B- > D\}$
$set0 = (AC),\ C- > D \Rightarrow set1 = (AC)\bigcup(D) = (ACD)$
$set1 = (ACD),\ AD- > B \Rightarrow set2 = (ACD)\bigcup(B) = (ABCD)$
$set3 = (ABCD)\bigcup(EF)$

2.
$set0 = (BC),\ BC- > A \Rightarrow set1 = (BC)\bigcup(A) = (ABC)$
$set1 = (ABC),\ C- > D \Rightarrow set2 = (ABC)\bigcup(D) = (ABCD)$
$set3 = (ABCD)\bigcup(EF)$
so there are 2 candidate keys: ACEF,BCEF

(b) Because $AD+ = ABD, C+ = CD, BC+ = ABC, B+ = BD$, and none of them violates to be a key, and because AD, C, BC, B does not contain a key, all of them does not satisfy, so not BCNF

(c) 1.Get key of ABCDEF = ACEF.
1.1.Function dependency AD -> B not satisfy BCNF.
1.2 So decomposite FD to ABD and ABCDEF to ACDEF.
2.Get key of ABD is AD.

2.1 Function dependency B->D not satisfy BCNF.

2.2 So decomposite FD to AB and other to BD.

3.For BD and AB, Function dependency are B->D and AB->B, these are both BCNFs.

4.Get key of ACDEF = ACEF.

4.1 Functional Dependency C->D not satisfy BCNF.

4.2 So decomposite FD to ACEF and other to CD.

5.ACEF->ACEF, so satisfy BCNF.

6.C->C, so satisfy BCNF.

Result = [AB,BD,ACEF,CD],key for each of them are [AB,B,ACEF,C].

Q1(ii)

Solution:

(a) 1.

$F = \{BC- > E, C- > AB, AF- > CD\}$

$set0 = (CF),\ C- > AB \Rightarrow set1 = (CF)\bigcup(AB) = (ABCF)$

$set1 = (ABCF),\ BC- > E \Rightarrow set2 = (ABCF)\bigcup(E) = (ABCEF)$

$set2 = (ABCEF),\ AF- > D \Rightarrow set3 = (ABCEF)\bigcup(D) = (ABCDEF)$

2.

$set0 = (AF),\ AF- > CD \Rightarrow set1 = (AF)\bigcup(CD) = (ACDF)$

$set1 = (ACDF),\ C- > AB \Rightarrow set2 = (ACDF)\bigcup(AB) = (ABCDF)$

$set2 = (ABCDF),\ BC- > E \Rightarrow set3 = (ABCDF)\bigcup(E) = (ABCDEF)$

so there are 2 candidate keys: CF,AF

(b) Because $BC+ = BCE, C+ = ABC, AF+ = ACDF$, and 2/3 of them violates to be a key, and because BC, C does not contain a key, so not BCNF

(c) 1.Get key of ABCDEF = AF.

1.1.Function dependency BC -> E not satisfy BCNF.

1.2 So decomposite FD to BCE and ABCDEF to ABCDF.

2.For BCE, Function dependency are BC->E, It is BCNF.

3.Get key of ACDEF = AF.

3.1 Function dependency C->AB not satisfy BCNF.

3.2 So decomposite FD to ABC and other to CDF.

4.For CDF and ABC, Function dependency are CDF->CDF and C->AB, these are both BCNFs.

Result = [BCE,ABC,CDF],key for each of them are [BC,C,CDF].

Q1(iii)

Solution:
(a) 1.

$F = \{ABF- > D, CD- > E, BD- > A\}$
$set0 = (BCD),\ CD- > E \Rightarrow set1 = (BCD) \bigcup (E) = (BCDE)$
$set1 = (BCDE),\ BD- > A \Rightarrow set2 = (BCDE) \bigcup (A) = (ABCDE)$
$set2 = (ABCDE),\ BC- > E \Rightarrow set3 = (ABCF) \bigcup (E) = (ABCEF)$
$set4 = (ABCDE) \bigcup (F) = (ABCDEF)$

2.
$set0 = (ABCF),\ ABF- > D \Rightarrow set1 = (ABCF) \bigcup (D) = (ABCDF)$
$set1 = (ABCDF),\ CD- > E \Rightarrow set2 = (ABCDF) \bigcup (E) = (ABCDEF)$
so there are 2 candidate keys: ABCF,ACDF

(b) Because $ABF+ = ABDF, CD+ = CDE, BD+ = ABD$, and none of them violates to be a key, and because ABF, CD, BD does not contain a key, so not BCNF

(c) 1.Get key of ABCDEF = ABCF.
1.1 Function dependency ABF -> D not satisfy BCNF.
1.2 So decomposite FD to ABDF and ABCDEF to ABCEF.
2.For ABDF, key is ABF.
2.1 Function dependency are BD->A, It is not BCNF.
2.2 So decomposite FD to ABD and other to BDF.
3.For ABD,BDF,ABCEF, Function Depedency are BD->A, BDF->BDF, ABCF->E, Each of them are BCNFs.
Result = [BDA,BDF,ABCEF],key for each of them are [BD,BDF,ABCEF].

Q1(iv)

Solution:

(a)

$F = \{AB- > D, BCD- > EF, B- > C\}$

$set0 = (AB),\ AB- > D \Rightarrow set1 = (AB)\bigcup(D) = (ABD)$

$set1 = (ABD),\ B- > C \Rightarrow set2 = (ABD)\bigcup(C) = (ABCD)$

$set2 = (ABCD),\ BCD- > EF \Rightarrow set3 = (ABCD)\bigcup(EF) = (ABCDEF)$

so there are 1 candidate key: AB

(b) Because $AB+ = ABD, BCD+ = BCDEF, B+ = BC$, and 2/3 of them violates to be a key, and because ABF, CD, BD does not contain a key, so not BCNF

(c) 1.Get key of ABCDEF = AB.

1.1 Function dependency BCD -> EF not satisfy BCNF.

1.2 So decomposite FD to BCDEF and ABCDEF to ABCD.

2.For ABCD, key is AB.

2.1 Function dependency B->C not satisfy BCNF.

2.2 So decomposite FD to BC and other to ABD.

3 For BC,ABD,BCDEF, Function Depedency are B->C, AB->D, BCD->EF, Each of them are BCNFs.

Result = [BC,ABD,BCDEF],key for each of them are [B,AB,BCD].

Question 2

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".

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

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

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.

Q2(i)

Solution:

$Sel\_Name = Proj[name]$

$C1 = Company$

$C2 = Category$

$Result = Sel\_Name(Sel[sector = \ 'Technology'](C1\ Join\ C2))$

Q2(ii)

Solution:

$E = Executive$

$T1 = Groupby[code, Count[person]](E)$

$T2 = Rename[1- > code, 2- > numP(T1)]$

$Result = Proj[code](Sel[numP > 5](T2))$

Q2(iii)

Solution:

$E = Executive$

$T1 = Groupby[person, Count[Code]](E)$

$T2 = Rename[1- > person, 2- > numC](T1)$

$Result = Proj[person](Sel[numC >= 2](T2))$

Q2(iv)

Solution:

$C1 = Category$

$T1 = Groupby[industry, Count[code]](C1)$

$T2 = Rename[1- > industry, 2- > numC]$

$Result = Proj[T2.code, T2.industry](Sel[numC = 2](T2 Join C1))$

Question 3

Suppose that the 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).

ii.$SEL_{[c]}$ (R × S), for some condition c.

iii.$PROJ_{[a]}$ (R) - $PROJ_[a]$ (R JOIN S), for some list of attributes a.

Q3(i)

Solution:

For $T_1 = S$ INTERSECT T,

Max: Assumpt that S and T are compatible, under this case S $\subset T$ *or* $T \subset S$, max value = S or T

Min: Obviously that when S and T are not intersect $S \cap T = \Phi$, min value = 0

For $T_2 = R$ UNION (T_1)

Max: Assumpt that S and T are compatible, under this case $R \cap T = \Phi$ or $R \cap S = \Phi$, max value = R + T | R + S

Min: Obviously that when S and T are not intersect $S \cap T = \Phi$, min value = 0

Q3(ii)

Solution:

For $T_1 = S \times T$,

Max: max value = S * R

Min: the same, min value = S * R

For $T_2 = SEL_{[c]}(T_1)$

Max: Attribute c belongs to R_1, if all matches, max value = S * R

Min: if no matches exist, min value = 0

Q3(iii)

Solution:

For $T_1 = R$ JOIN S,

Max: Easy to get max value = S or R

Min: Assumpt that R and S have totally different attribute, min value = 0

For $T_2 = PROJ_{[a]}(T_1)$

Max: According to for some list of contribute a to form R_1, so if all matches, max value = S or R

Min: if no matches exist, min value = 0

For $T_3 = PROJ_{[a]}(R)$

Max: For R, if all matches, max value = R

Min: if no matches exist, min value = 0

For $T_4 = T_3 - T_2$

Max: max value = R

Min: min value = 0

Question 4

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)

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)

Q4(i)

Solution:

$T1$	$T2$
$R(x)$	
	$W(x)$
$W(x)$	
	$R(x)$
	$R(y)$
$R(y)$	
$W(y)$	
	$W(x)$

Table 1: Table for Q4(i)

Obviously, There are several conflicts:
T(1):W(x), T(2):R(x) conflict gives T1->T2
T(2):R(y), T(1):R(y) conflict gives T2->T1

So cycle exist in the precedence graph. Therefore it is not serialisable

Q4(ii)

Solution:

$T1$	$T2$	$T3$	$T4$
		$R(x)$	
			$W(y)$
			$W(z)$
$W(y)$			
	$R(y)$		
		$R(d)$	
	$W(x)$		
$R(x)$			

Table 2: Table for Q4(ii)

Obviously, There are several conflicts:
T(3):R(x), T(2):W(x) conflict gives T3->T2
T(4):W(y), T(2):R(y) conflict gives T4->T2
T(4):W(y), T(1):W(y) conflict gives T4->T1
T(2):W(x), T(1):R(x) conflict gives T2->T1
T(1):W(y), T(2):R(y) conflict gives T1->T2

So cycle exist in the precedence graph. Therefore it is not serialisable