COMP9311 - Assignment 3 - Hao Chen - 5102446

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. $AD \rightarrow B,\ C \rightarrow D,\ BC \rightarrow A,\ B \rightarrow D$

a. Candidate key: *ACEF, BCEF*

b. Not BCNF, none of LHS FDs contain a key.

c.

- We start from a schema: *ABCDEF*, with key *ACEF*.
- The FD $AD \rightarrow B$ violates BCNF (FD with non-key on LHS).
- To fix, we need to decompose into tables: *ABD* and *ACDEF*.
- Key for *ABD* is *AD*, and FD $B \rightarrow D$ violates BCNF (FD with non-key on LHS).
- To fix, we need to decompose into tables: *AB* and *BD*.
- FDs for *AB* are $\{AB \rightarrow B\}$, therefore key is *AB*, therefore BCNF.
- FDs for *BD* are $\{B \rightarrow D\}$, so key is *B* and table is BCNF
- Key for *ACDEF* is *ACEF*, and FD $C \rightarrow D$ violates BCNF (FD with non-key on LHS).
- To fix, we need to decompose into tables: *ACEF* and *CD*.
- FDs for *ACEF* are $\{ACEF \rightarrow ACEF\}$, therefore key is *ACEF*, therefore BCNF.
- FDs for *CD* are $\{C \rightarrow D\}$, so key is *C* and table is BCNF
- Final schema (with keys bold): ***AB***, ***BD***, ***ACEF***, ***CD***.

ii. $BC \rightarrow E,\ C \rightarrow AB,\ AF \rightarrow CD$

a. Candidate key: *AF, CF*

b. Not BCNF, in $BC \rightarrow E$, *BC* does not contain a key, in $C \rightarrow AB$, *C* does not contain a key.

c.

- We start from a schema: *ABCDEF*, with key *AF*.
- The FD $BC \rightarrow E$ violates BCNF (FD with non-key on LHS).
- To fix, we need to decompose into tables: *BCE* and *ABCDF*.
- FDs for *BCE* are $\{BC \rightarrow E\}$, so key is *BC* and table is BCNF
- Key for *ACDEF* is *AF*, and FD $C \rightarrow AB$ violates BCNF (FD with non-key on LHS).
- To fix, we need to decompose into tables: *ABC* and *CDF*.
- FDs for *ABC* are $\{C \rightarrow AB\}$, therefore key is *C*, therefore BCNF.
- FDs for *CDEF* are $\{CDF \rightarrow CDF\}$, so key is *CF* and table is BCNF
- Final schema (with keys bold): ***BCE***, *AB**C***, ***CDF***.

iii. $ABF \rightarrow D, \quad CD \rightarrow E, \quad BD \rightarrow A$

a. Candidate keys: *ABCF, BCDF.*

b. Not BCNF, none of LHS of the FDs contain a key

c.

- We start from a schema: *ABCDEF*, with key *ABCF.*
- The FD $ABF \rightarrow D$ violates BCNF (FD with non-key on LHS).
- To fix, we need to decompose into tables: *ABDF* and *ABCEF.*
- Key for *ABDF* is *ABF*, and FD $BD \rightarrow A$ violates BCNF (FD with non-key on LHS).
- To fix, we need to decompose into tables: *ABD* and *BDF.*
- FDs for *ABD* are $\{BD \rightarrow A\}$, therefore key is *BD*, therefore BCNF.
- FDs for *BDF* are $\{BDF \rightarrow BDF\}$, so key is *BDF* and table is BCNF.
- FDs for *ABCEF* are $\{ABCF \rightarrow E\}$, therefore key is *ABCF*, therefore BCNF.
- Final schema (with keys bold): ***BD****A*, ***BDF, ABCFE***

iv. $AB \rightarrow D, \quad BCD \rightarrow EF, \quad B \rightarrow C$

a. Candidate key: *AB*

b. Not BCNF, in $BCD \rightarrow EF$, *BCD* does not contain a key, also in $B \rightarrow C$, *B* does not contains a key.

c.

- We start from a schema: *ABCDEF*, with key *AB.*
- The FD $BCD \rightarrow EF$ violates BCNF (FD with non-key on LHS).
- To fix, we need to decompose into tables: *BCDEF* and *ABCD.*
- FDs for *BCDEF* are $\{BCD \rightarrow EF\}$, therefore key is *BCD*, therefore BCNF.
- Key for *ABCD* is *AB*, and FD $B \rightarrow C$ violates BCNF (FD with non-key on LHS).
- To fix, we need to decompose into tables: *BC* and *ABD.*
- FDs for *BC* are $\{B \rightarrow C\}$, therefore key is *BD*, therefore BCNF.
- FDs for *ABD* are $\{AB \rightarrow D\}$, so key is *AB* and table is BCNF.
- Final schema (with keys bold): ***BCD****EF*, ***BC, AB****D*

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".

Co = Company

Ca = Category

Answer = Proj[name](Sel[sector = 'Technology'](Co Join Ca))

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

Ex = Executive

R1 = GroupBy[code, Count[person]](Ex)

R2 = Rename[1->person, 2->nperson](R1)

Answer = Proj[code](Sel[nperson >= 6](R2))

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

Ex = Executive

R1 = GroupBy[person, Count[code]](Ex)

R2 = Rename[1->person, 2->ncode](R1)

Answer = Proj[person](Sel[ncode >= 2](R2))

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.

Ca = Category

R1 = GroupBy[industry, Count[code]](Ca)

R2 = Rename[1->industry, 2->ncode](R1)

Answer = Proj[R2.code, R2.industry](Sel[ncode 1= 2](R2 Join Ca))

3. 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)

Expression	Assumptions	Max	Min
S INTERSECT T	S and T are union-compatible	s or t when $s \subset t$ or $t \subset s$	0 when $s \cap t = \varnothing$
R UNION (S INTERSECT T)	S and T are union-compatible	r + s or r +t when $r \cap s = \varnothing$ or $r \cap t = \varnothing$	0 when r = ∅

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

Expression	Assumptions	Max	Min
R x S	None	r * s	r * s
Sel[c](R x S)	R x S has an attribute c	r * s all match	0 no matchs

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

Expression	Assumptions	Max	Min
R JOIN S	None	r or s	0 when r and s have no same attribute
Pro[a](R JOIN S)	None	r or s all matchs	0 no match
R - Pro[a](R JOIN S)	None	r	0 or r-s

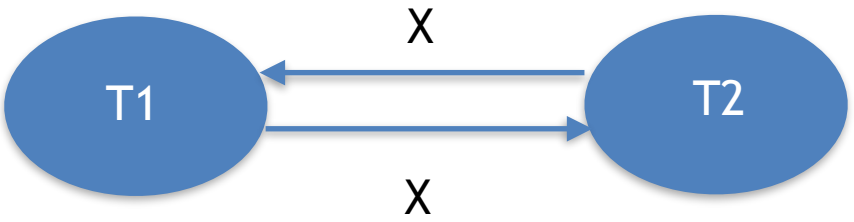
4. For the following execution schedule, construct its precedence graph. Is this schedule serialisable?

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

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

T2: R(X), T1: W(X) conflict gives T2 → T1

T1: W(X), T2: W(X) conflict gives T1 → T2



Graph has cycle => not serialisable

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

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

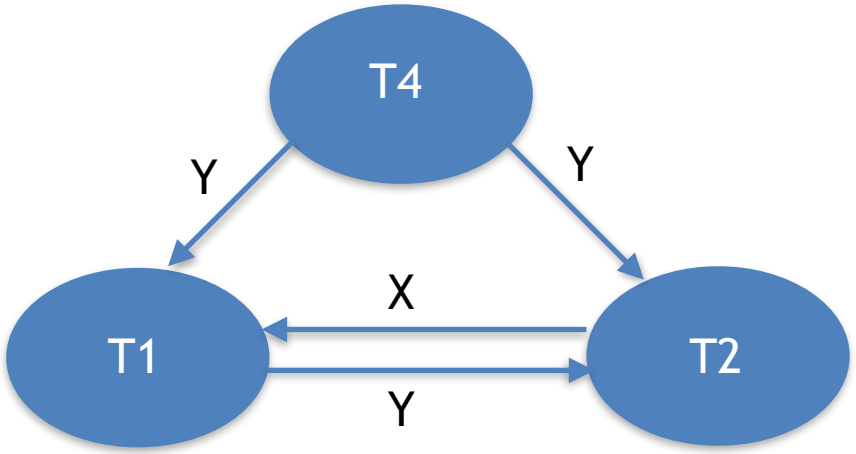
T1				W(Y)				R(X)
T2					R(Y)		W(X)	
T3	R(X)					R(D)		
T4		W(Y)	W(Z)					

T4: W(Y), T1: W(Y) conflict gives T4 → T1

T1: W(Y), T2: R(Y) conflict gives T1 → T2

T4: W(Y), T2: R(Y) conflict gives T4 → T2

T2: W(X), T1: R(X) conflict gives T2 → T1



Graph has cycle => not serialisable