COMP3311
Connor McLeod
z5058240

Assignment 2 Q2-3

Q2

i) For R(A,B,C,D,E,G,H) with FDs {A,B → C,D; E → D; A,B,C → D,E; E → A,B; D → A,G; ACD → B,E}, find F_m:

Reducing RHS and LHS and removing redundant FDs gives:

D	**E**	**A,B**	**A,B,C**	**A,C,D**
D → A	~~E → A~~	A,B → C	~~A,B,C → D~~	~~A,C,D, → B~~
D → G	E → B	~~A,B → D~~	A,B,~~C~~ → E	~~A~~,C,D → E
	E → D			

.: F_m = {D → A,G; E → B,D; A,B → C,E; C,D → E}

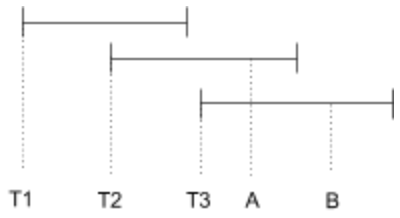
ii) 3NF Decomposition (dependency preserving and lossless):

From F_m above, take the LHS term and check resulting schema is not a key of R (From Q1: {E,H}, {A,B,H}, {B,D,H} or {C,D,H}) . Since key of R is not found, add key {E,H} as R5.

R1 = (A, D, G) having FDs {D → A; D → G}
R2 = (E, B, D) having FDs {E → B; E → D}
R3 = (A, B, C, E) having FDs {E → B; A,B → C; A,B → E}
R4 = (C, D, E) having FDs {E → D; C,D → E}
R5 = (E, H) having no FDs

Q3

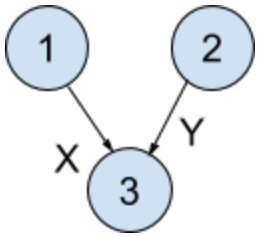
i) The system crashes at “B” because there are two values for Y. Hence, T3 and T2 should be undone by rolling back, or restore from the backup.



ii) Checkpoint “A” would allow the three transactions should be undone to return to the checkpoint “A”

iii) Yes, it is conflict serialisable.

By mapping the schedule graph, and determining that the graph is acyclic, we conclude that it is conflict serialisable.



iv) Reordering and selecting some of the transactions, as well as using read/write locks on the schedule, we can reach a deadlock situation:

$$WL_1(X),\ R_1(X),\ WL_2(Y),\ R_2(Y),\ WL_1(Y),\ WL_2(X)$$

This results in T1 waiting for Y and T2 waiting for X, thereby causing deadlock for the schedule