Gretel Rajamoney
rajamong@oregonstate.edu
933188305

Kaavya Subramanian
subramka@oregonstate.edu
933291513

1: Sort-Merge Join

a. Source code submitted separately through Canvas!

2: Query Optimization

a.

Query	Output Size	Cost	Plan
R X_B S	60K	8750	R X_B S
R X_C S	40K	8750	R X_C S
R X S	12M	3500	R X S
R X_B W	40K	7500	R X_B W
R X W	8M	3000	R X W
R X_A U	40K	6250	R X_A U
R X U	3M	2500	R X U
S X_B W	60K	6250	S X_B W
S X U	3M	2000 (no sorting)	S X U
W X_D U	20K	3750	W X_D U
W X U	2M	1500	W X U
R X_B S X_B W	600K	(R X_B S)X_B W = 41250	(R X_B W)X_B S

		(R X_B W)X_BS = 31250 (W X_BS)X_BR = 41250	
R X S X U	20M	(R X_cS) X U = 29250 (R X_A U) X S = 26750 (S X U) X R = 6002000	(R X_A U) X S
R X W X U	40M	(R X_BW) X U = 28000 (R X_A U) X W = 27750 (U X W) X R > 27250	(R X_A U) X W = 27750
S X W X U	60M	(S X_BW) X U = 36750 (W X_DU) X S = 1300 (S X U) X W > 1300	(W X_DU) X S
R X S X W X U	600M	R X (S X W X U) = R X ((W X_DU) X S)= 120M S X (R X W X U) = S X ((R X_A U) X W) = 80M W X (R X S X U) = W X ((R	U X ((R X_BW)X_BS)

		X_A U) X S) = 40M U X (R X S X W) = U X ((R X_B W)X_B S) = 1.2M	

The most efficient join order is U X ((R X_B W)X_B S).

3: Serializability and 2PL

a.

1. T1:R(X), T2:R(Y), T3:W(X), T2:R(X), T1:R(Y)
   It is serializable because it doesn't contain blind-writes or repeated reads. It does not avoid cascading aborts, and it is not strict either. Finally, we can not determine whether it is recoverable or not because the abort/commit sequence of both of these transactions have not been specified.
2. T1:R(X), T1:R(Y), T1:W(X), T2:R(Y), T3:W(Y), T1:W(X), T2:R(Y)
   It is not serializable because it contains a repeated read. It does not avoid cascading aborts, and it is not strict either. Finally, we can not determine whether it is recoverable or not because the abort/commit sequence of these transactions have not been specified.
3. T1:W(X), T2:R(X), T1:W(X)
   It is serializable because it does not contain blind-writes or repeated reads. It does not avoid cascading aborts, and it is not strict either. Finally, we can not determine whether it is recoverable or not because the abort/commit sequence of these transactions have not been specified.
4. T1:R(X), T2:W(X), T1:W(X), T3:R(X)
   It is not serializable because it contains a blind-write. It does not avoid cascading aborts, and it is not strict either. Finally, we can not determine whether it is recoverable or not because the abort/commit sequence of these transactions have not been specified.

4: Degrees of Consistency

a. In 'Degree 0', the transaction does not overwrite dirty data of other xacts, essentially it does not contain any blind-writes. In 'Degree 1', the transaction does not commit any writes until the end of the transaction. In 'Degree 2', the transaction does not read dirty data from other xacts. In 'Degree 3', other xacts do not dirty any data read by the transaction before the transaction completes. In the schedule shown in Table 1, T1 will

read the transaction as well as write, but it will not be saved since it does not commit. Meanwhile, T2 will read the transaction and it will write and save it by committing it. Also in T1, it will again read Y, write Y, and it will also be committed. Since T2 is reading what T1 is writing, before T1 is able to commit it, it is performing a dirty read. Therefore it cannot be in degree 2. Since both transactions T1 and T2 commit their writes and the end of the transaction, the maximum degrees of consistency for T1 and T2 that makes this schedule possible is degree 1.