Problem 1: Suppose that we have the following tuples in a relation S with three attributes ABC: (1,2,3), (4,2,3), (5,3,3), (5,3,4). Which of the following (→) and multivalued (→→) dependencies can you infer does not hold over relation S?

1. A → B
2. A →→ B
3. BC → A
4. BC →→ A
5. B → C
6. B →→ C

Problem 2: Consider a relation R with five attributes ABCDE.

1. For each of the following instances of R, state whether it violates (a) the FD BC → D and b) the MVD BC →→ D:

a) { } (i.e, empty relation)

BC → D is not violated
BC →→ D is not violated

b) {(a,2,3,4,5), (2,a,3,5,5)}

BC → D is not violated
BC →→ D is not violated

c) {(a,2,3,4,5), (2,a,3,5,5), (a,2,3,4,6)}

BC → D is being violated as there are two tuples with the same BC values but different D values
BC →→ D is not violated

d) {(a,2,3,4,5), (2,a,3,4,5), (a,2,3,6,5)}

BC → D is being violated as there are two tuples with the same BC values but different D values
BC →→ D is not violated

e) {(a,2,3,4,5), (2,a,3,7,5), (a,2,3,4,6)}

BC → D is being violated as there are two tuples with the same BC values but different D values
BC →→ D is violated for the same reason

f) {(a,2,3,4,5), (2,a,3,4,5), (a, 2,3,6,5), (a,2,3,6,6)}

BC⇒D is being violated as there are two tuples with the same BC values but different D values

BC↠D is violated for the same reason

g) {(a,2,3,4,5), (a,2,3,6,5), (a, 2,3,6,6), (a,2,3,4,6)}

BC⇒D is being violated as there are two tuples with the same BC values but different D values

BC↠D is violated for the same reason

2. If each instance for R listed above is legal, what can you say about the FD A⇒B?

This implies that for every pair of tuples in R, if they have the same A value, they must also have the same B value

Problem 3: Consider the following actions taken by transaction T_1 on database objects X and Y:

R(X), W(X), R(Y), W(Y)

1. Give an example of another transaction T_2 that, if run concurrently on transaction T w/o some form of concurrency control, could interfere with T_1.

If the transaction T_2 performed W(Y) before T_1 performed R(Y), and then T_2 aborted, the value read by T_1 would be invalid and the abort would be cascaded to T_1.

2. Explain how the use of Repeatable Read Isolation Level would prevent interference between the two transactions.

Under the Repeatable Read Isolation Level, locks are placed on the data items that have been read by a transaction, ensuring that other transactions cannot modify those items while the transaction is in progress. This eliminates the risk of interference between the two transactions.

Problem 4: Consider a database with objects X and Y and assume that there are two transactions T_1 and T_2. Transaction T_1 reads objects X and Y and then writes object X. Transaction T_2 reads objects X and Y and then writes objects X and Y.

1. Give an example schedule with actions of transactions T_1 and T_2 on objects X and Y that results in a write-read conflict

$T_1: R(X), T_1: R(Y), T_1: W(X), T_2: R(X)$ – Dirty Read

2. Give an example schedule with actions of transactions T_1 and T_2 on objects X and Y that results in a read-write conflict

$T_1: R(X), T_1: R(Y), T_2: R(X), T_1: W(X), T_2$ [R,Y] will get unrepeatable read

3. Give an example schedule with actions of transactions T_1 and T_2 on objects X and Y that results in a write-write conflict

$T_1: R(X), T_1: R(Y), T_2: R(X), T_1: W(X), T_2: W(X)$

4. For each of the three schedules show that the use of Repeatable Read Isolation Level disallows the schedule.

write-read: T_2 will get a shared lock on X until T_1 commits
read-write: T_1 will not get exclusive lock on X until T_2 commits
write-write: T_1 will not get exclusive lock on X until T_2 commits