1)
After the serial schedule execution, A = 11 and B = 12. In order for the schedule to be serializable, it needs to have the same output.

a. Serializable - end up with A = 11 and B = 12 at the end
b. Non-serializable - end up with A = 10 and B = 12 at the end
c. Serializable - end up with A = 11 and B = 12 at the end

2)
S1.
Strict: NO, T1 reads A before T2 has committed after T2 wrote on A
Serial: NO, T1 reads A before T2 has committed
Recoverable: NO, T1 writes C and T2 writes C but T2 commits before T1
S2.
Strict: NO, T2 write on A before T1 has committed the read on A
Serial: NO, T2 writes A before T1 has committed
Recoverable: YES, T1 reads and writes on C and commits before T2 reads and writes C
S3.
Strict: YES, the schedule is serial, meaning that it is also strict
Serial: YES, T1 commits before T2 does any actions
Recoverable: YES, there is no conflict between T1 and T2
S4.
Strict: NO, T2 writes on A before T1 has committed the read on A
Serial: NO, T2 writes on A before T1 has committed
Recoverable: YES, there is no read after writing C

3)

Data	Lock	Owner	Waiting	Explanation
A	S	T2		S(A) is granted to T2
A	S	T2		T2 does R(A)
				T2 commits and S(A) is released
A	S	T1		S(A) is granted to T1
A	S	T1		T1 does R(A)
A	X	T1		T1 does W(A)
				T1 commits and X(A) is released

Data	Lock	Owner	Waiting	Explanation
A	S	T1		S(A) is granted to T1
A	S	T1		T1 does R(A)
A	S	T2, T1		S(A) is also granted to T2 so now T1 and T2 have S(A)
A	S	T2, T1		T2 does R(A)
A	S	T2, T1	T1 waiting for X(A)	T1 is waiting for X(A) because T2 has S(A)
A	S	T1	T1 waiting for X(A)	S(A) is removed from T2 because of the commit
A	X	T1		S(A) is upgraded to X(A) because S(A) was removed from T2 and T1 does W(A)
				X(A) is removed from T1 because of the commit

4)

a. needs: IS lock on db (granted), IS lock on f1 (granted), IS lock on p1 (granted), S lock on r3 (not granted)
b. Needs: IX lock on db (granted), IX lock on f2 (granted), SIX lock (granted), X lock on r4 (granted)