Part 1: Schedules and Anomalies (10 points)

Consider a database with objects X, Y, and Z and assume that there are two transactions T1 and T2 that attempt the following operations.

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

T2: R(X), R(Y), W(Y), R(X), R(Y), W(X), R(Z), W(Z)

A) Write an example schedule that interleaves operations between T1 and T2, that is NOT conflict serializable.
To make schedule that is NOT conflict serializable it needs to be cyclic, in our case there needs to be a cycle between the two operations T1 and T2.
R1(X) → R2(X) → R2(Y) → W2(Y) → R(Y) → R1(Y) → R2(X) → R2(Y) → W2(X) → W1(X) → R2(Z) → W2(Z) → C1 → C2

B) If T1 is instead just "R(X)", this corresponds to T1 just being a single query like

```
SELECT * FROM Flights WHERE id=1024;
```

Do we need a transaction for a single query statement like this? Why or why not?
We do not need a transaction for a single query statement like above. One reason is that every single SQL statement will have implicit transaction statements, therefore there is not a need to make a transaction for a single query like above. Another reason is that when running a transaction, it will "begin transaction/commit," locking its functionality due to the atomic requirement in ACID. However, this is not very beneficial for one statement since it can't have its statement "interrupted" by another query. This of course would be beneficial for a transaction with multiple queries.