Transactions

Location Update Conflict

TRANSACTION T1

```
START TRANSACTION;

UPDATE car
SET location = 'Govind Puri'
WHERE id = (SELECT RC FROM driver WHERE status = 'Active');

COMMIT;
```

TRANSACTION T2

```
START TRANSACTION;

UPDATE car
SET location = 'Kashmere Gate'
WHERE id = (SELECT RC FROM driver WHERE status = 'Active');

COMMIT;
```

NON-CONFLICT SERIALIZABLE SCHEDULE:-

T1	T2
Shared Lock(Driver)	
Read(Driver)	
	Shared Lock(Driver)
	Read(Driver)
	Exclusive Lock(Car)

	Write(Car)
	COMMIT
	Release Shared Lock(Driver)
	Release Exclusive Lock(Car)
Exclusive Lock(Car)	
Write(Car)	
COMMIT	
Release Shared Lock(Driver)	
Release Exclusive Lock(Car)	

CONFLICT SERIALIZABLE SCHEDULE:-

T1	T2
Shared Lock(Driver)	
Read(Driver)	
	Shared Lock(Driver)
	Read(Driver)
Exclusive Lock(Car)	
Write(Car)	
COMMIT	
Release Shared Lock(Driver)	
Release Exclusive Lock(Car)	
	Exclusive Lock(Car)
	Write(Car)

	COMMIT
	Release Shared Lock(Driver)
	Release Exclusive Lock(Car)

EXPLANATION:- When these transactions are executing concurrently and assume there is only 1 available car, then if T1 comes first and updates the car location, then before it commits changes, the transaction manager switches to the T2 and starts executing it, and T2 also updates the location of the same car and commits after that T1 commits then changes made my T2 will become irrecoverable hence this lead to a conflict between two transactions.