Spring boot - @Transaction Part3 (Isolation Level)

Isolation Level

Isolation level:
It tells, how the changes made by one transaction are visible to other transactions running in parallel.

```java
@Transactional(propagation = Propagation.REQUIRED, isolation = Isolation.READ_COMMITTED)
public void updateUser() {

    //some operations here
}
```

Isolation Level	Dirty Read Possible	Non-Repeatable Read Possible	Phantom Read Possible
READ_UNCOMMITTED	Yes	Yes	Yes
READ_COMMITTED	No	Yes	Yes
REPEATABLE_COMMITTED	No	No	Yes
SERIALIZABLE	No	No	No

↑ Concurrency High

Concurrency Low

Default isolation level, depends upon the DB which we are using.
Like Most relational Databases uses READ_COMMITTED as default isolation, but again it depends upon DB to DB.

Dirty Read Problem

Transaction A reads the un-committed data of other transaction.
and
If other transaction get ROLLED BACK, the un-committed data which is read by Transaction A is known as Dirty Read.

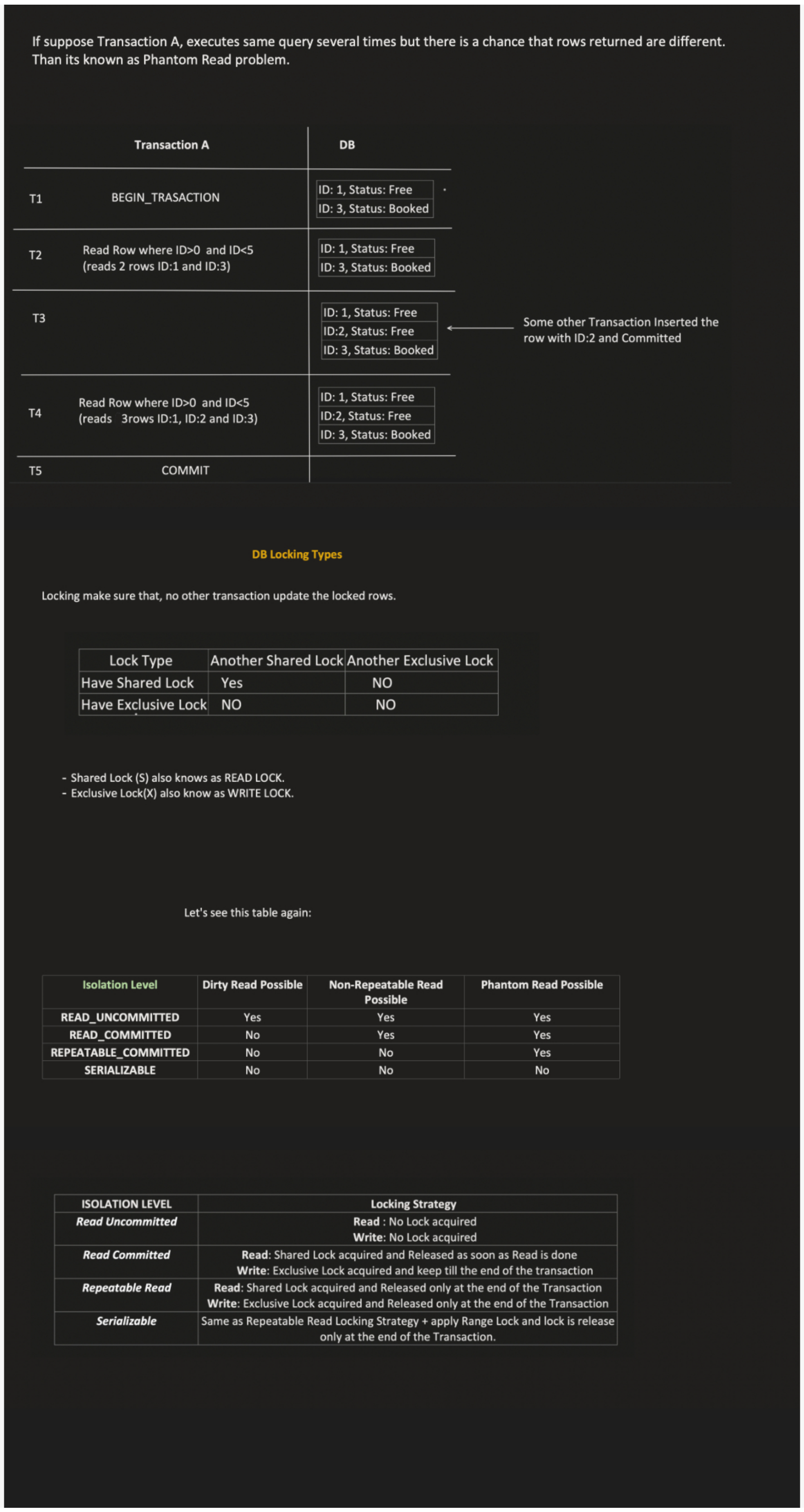
Time	Transaction A	Transaction B	DB Status
T1	BEGIN_TRANSACTION	BEGIN_TRANSACTION	Id: 123 Status: free
T2		Update Row id:123 Status = booked	Id: 123 Status: booked (Not Committed by Transaction B yet)
T3	Read Row id:123 (Got status = booked)		Id: 123 Status: booked (Not Committed by Transaction B yet)
T4		Rollback	Id: 123 Status: Free (Un-committed changes of Txn B got Rolled Back)

Non-Repeatable Read Problem

If suppose Transaction A, reads the same row several times and there is a chance that it get different value, then its known as Non-Repeatable Read problem.

	Transaction A	DB	
T1	BEGIN_TRASACTION	ID: 1 Status: Free	
T2	Read Row ID:1 (reads status: **Free**)	ID: 1 Status: Free	
T3		ID: 1 Status: Booked	← Some other Transaction changed and committed the changes.
T4	Read Row ID:1 (reads status: **Booked**)	ID: 1 Status: Booked	
T5	COMMIT		

Phantom Read Problem



If suppose Transaction A, executes same query several times but there is a chance that rows returned are different. Than its known as Phantom Read problem.

	Transaction A	DB	
T1	BEGIN_TRASACTION	ID: 1, Status: Free ID: 3, Status: Booked	
T2	Read Row where ID>0 and ID<5 (reads 2 rows ID:1 and ID:3)	ID: 1, Status: Free ID: 3, Status: Booked	
T3		ID: 1, Status: Free ID:2, Status: Free ID: 3, Status: Booked	← Some other Transaction Inserted the row with ID:2 and Committed
T4	Read Row where ID>0 and ID<5 (reads 3rows ID:1, ID:2 and ID:3)	ID: 1, Status: Free ID:2, Status: Free ID: 3, Status: Booked	
T5	COMMIT		

DB Locking Types

Locking make sure that, no other transaction update the locked rows.

Lock Type	Another Shared Lock	Another Exclusive Lock
Have Shared Lock	Yes	NO
Have Exclusive Lock	NO	NO

- Shared Lock (S) also knows as READ LOCK.
- Exclusive Lock(X) also know as WRITE LOCK.

Let's see this table again:

Isolation Level	Dirty Read Possible	Non-Repeatable Read Possible	Phantom Read Possible
READ_UNCOMMITTED	Yes	Yes	Yes
READ_COMMITTED	No	Yes	Yes
REPEATABLE_COMMITTED	No	No	Yes
SERIALIZABLE	No	No	No

ISOLATION LEVEL	Locking Strategy
Read Uncommitted	**Read** : No Lock acquired **Write**: No Lock acquired
Read Committed	**Read**: Shared Lock acquired and Released as soon as Read is done **Write**: Exclusive Lock acquired and keep till the end of the transaction
Repeatable Read	**Read**: Shared Lock acquired and Released only at the end of the Transaction **Write**: Exclusive Lock acquired and Released only at the end of the Transaction
Serializable	Same as Repeatable Read Locking Strategy + apply Range Lock and lock is release only at the end of the Transaction.