PROPAGATION_REQUIRED = 0; If DataSourceTransactionObject T1 is already started for Method M1.If for another Method M2 Transaction object is required ,no new Transaction object is created .Same object T1 is used for M2

PROPAGATION_MANDATORY = 2; method must run within a transaction. If no existing transaction is in progress, an exception will be thrown

PROPAGATION_REQUIRES_NEW = 3; If DataSourceTransactionObject T1 is already started for Method M1 and it is in progress(executing method M1) .If another method M2 start executing then T1 is suspended for the duration of method M2 with new DataSourceTransactionObject T2 for M2.M2 run within its own transaction context

PROPAGATION_NOT_SUPPORTED = 4; If DataSourceTransactionObject T1 is already started for Method M1.If another method M2 is run concurrently .Then M2 should not run within transaction context. T1 is suspended till M2 is finished.

PROPAGATION_NEVER = 5; None of the methods run in transaction context.

An isolation level: It is about how much a transaction may be impacted by the activities of other concurrent transactions.It a supports consistency leaving the data across many tables in a consistent state. It involves locking rows and/or tables in a database.

The problem with multiple transaction

Scenario 1.If T1 transaction reads data from table A1 that was written by another concurrent transaction T2.If on the way T2 is rollback,the data obtained by T1 is invalid one.E.g a=2 is original data .If T1 read a=1 that was written by T2.If T2 rollback then a=1 will be rollback to a=2 in DB.But,Now ,T1 has a=1 but in DB table it is changed to a=2.

Scenario2.If T1 transaction reads data from table A1.If another concurrent transaction(T2) update data on table A1.Then the data that T1 has read is different from table A1.Because T2 has updated the data on table A1.E.g if T1 read a=1 and T2 updated a=2.Then a!=b.

Scenario 3.If T1 transaction reads data from table A1 with certain number of rows. If another concurrent transaction(T2) inserts more rows on table A1.The number of rows read by T1 is different from rows on table A1

Scenario 1 is called **Dirty reads**

Scenario 2 is called **Nonrepeatable reads**

Scenario 3 is called **Phantom reads .**

So,isolation level is the extend to which **Scenario 1 ,Scenario 2 ,Scenario 3** can be prevented. You can obtained complete isolation level by implementing locking.That is preventing concurrent reads and writes to the same data from occurring.But it affects performance .The level of isolation depends upon application to application how much isolation is required.

ISOLATION_READ_UNCOMMITTED :Allows to read changes that haven't yet been committed.It suffer from Scenario 1 ,Scenario 2 ,Scenario 3

ISOLATION_READ_COMMITTED:Allows reads from concurrent transactions that have been com- mitted.It may suffer from Scenario 2 ,Scenario 3 . Because other transactions may be updating the data.

ISOLATION_REPEATABLE_READ:Multiple reads of the same field will yield the same results untill it is changed by itself.It may suffer from Scenario 3.Because other transactions may be inserting the data

ISOLATION_SERIALIZABLE: Scenario 1,Scenario 2,Scenario 3

never happens.It is complete isolation.It involves full locking.It affets performace because of locking.