**Transaction**

ACID – Atomic, Consistent, Isolated & Durable

**Atomic** – either it all works or it all fails

**Consistent** – whether it works (commits) or fails (roll back), data should stay consistent with the business logic reality.

**Isolated** – transaction should be protected from one another, the effect of one transaction should not corrupt the state of another transaction, very similar to thread synchronization

**Durable** – Once the transaction commits, the changes made by the transaction must become permanent!

CMT’s and BMT’s

Code it or declare it

<transaction-type>bean or container

Bean – UserTransactions

**Transaction Attributes**

Required, RequiresNew, Mandatory, Supports, NotSupported, Never

**Transaction Isolation levels**

**Dirty read**: You're permitted to read uncommitted, or dirty, data.

**Nonrepeatable read**: This simply means that if you read a row at time T1 and try to reread that row at time T2, the row may have changed. It may have disappeared (**deleted)**, it may have been **updated**, and so on.

**Phantom read**: This means that if you execute a query at time T1 and re-execute it at time T2, additional rows may have been added to the database, which may affect your results. This differs from a nonrepeatable read in that with a phantom read, data you already read hasn't been changed, but instead, **more data (addition) satisfies** your query criteria than before.

The **READ UNCOMMITTED** isolation level allows dirty reads. Oracle Database doesn't use dirty reads, nor does it even allow them. The basic goal of a READ UNCOMMITTED isolation level is to provide a standards-based definition that allows for nonblocking reads.

The **READ COMMITTED** isolation level states that a transaction may read only data that has been committed in the database.

**REPEATABLE READ**. The goal of REPEATABLE READ is to provide an isolation level that gives consistent, correct answers and prevents lost updates.

**SERIALIZABLE**. This is generally considered the most restrictive level of transaction isolation, but it provides the highest degree of isolation. A SERIALIZABLE transaction operates in an environment that makes it appear as if there are no other users modifying data in the database. Any row you read is assured to be the same upon a reread, and any query you execute is guaranteed to return the same results for the life of a transaction.

1. Transaction\_read\_uncommitted: Allows a method to read uncommitted data from a DB (fast but not wise).

2. Transaction\_read\_committed: Guarantees that the data you are getting has been committed.

3. Transaction\_repeatable\_read: Guarantees that all reads of the database will be the same during the transaction (good for read and update operations).

4. Transaction\_serializable: All the transactions for resource are performed serial.