Linearizability

- atomic consistency
- the basic idea is to make a system appear as if there were only one copy of the data, and all operations on it are atomic
- it is a recency guarantee on reads and writes on an individual object
- it does NOT group operations together into transactions, so it does not prevent problems like write skew, unless you take additional measures like materializing conflicts

Serializability

- it is an isolation property of transactions, where every transaction may read and write multipl objects
- it guarantees that transactions behave the same as if they had executed in some serial order, i.e. each transaction runs to its end before a subsequent one starts
- implementations of serializability based on 2PL or actual serial execution are typically linearizable
- however, Serializable Snapshot Isolation is not linearizable: by design, it makes reads from a consistent snapshot to avoid lock contention between readers and writers
- the whole point of a consistent snapshot is that it does not include writes that are more recent than the snapshot, so reads from the snapshot are not linearizable

Race conditions

Dirty Reads

- one client reads another client's writes before they have been committed
- Read committed isolation solves this

Dirty Writes

- one client overwrites data that another client has written but not yet committed
- Almost all txn implementations prevent dirty writes

Read skew

- a client sees different parts of the db at different points in time
- snapshot isolation prevents this -- usually implemented with MVCC

Lost updates

- two clients concurrently perform read-modify-write
- one overwrites the other's writes without incorporating changes from the other
- use snapshot isolation or locks

Write skew

- a txn reads something, makes a decision based on what it saw, and writes the decision fo the db, However, by the time the write is made, the premise of the decisoin is no longer true
- Only serializable isolation helps

Phantom reads

- a txn reads objects that match some search condition
- another client makes a write that affects the results of that search
- snapshot isolation prevents straightforward phantom reads
- but you may need index-range locks in the context of write skew