Lock

We use locks to protect data integrity and atomicity in concurrent applications where a record could get read/write requests.

Imagine a case where multiple users want to purchase an item where there's only 1 left in the stock. The company advertised that item on facebook/twitter/TV/etc. and instantly 2000 users want to purchase that item. Of course, you have to sell this item to only one user since only there's 1 left in the stock. To manage situations like this we employ locks. If we don't manage these situations this may cause that single item to be sold to all of those 2000 users.

Pessimistic Lock

Pessimistic Lock is where you assume that all the users are trying to access the same record and it literally locks the record exclusively for the first started transaction until it is completed successfully or failed. Then the lock is released and the next transaction on the record is handled in the same way.

1. In our case, if we apply Pessimistic Lock, the first user to come and buy the last item in the stock will click on "Purchase".
2. This will lock the object until the payment is completed or failed.
3. Let's say User A was able to pay for it and the stock value for that item is set to 0 now.
4. All the other users have to wait during this process.
5. Now all the other users will see that the item went out of stocks, and cannot do anything with the item.

Optimistic Lock

Optimistic Lock is where you manage your data by checking a special value in the database — it is often a version number, timestamp, date, etc. If the data is stale, the transaction is not completed successfully and an error is thrown to indicate that. Something like: *"The record you attempted to edit was modified by another user after you got the original value"*.

1. In our case, if we apply Optimistic Lock, the first user to come and buy the last item in the stock will click on "Purchase".

2. Let's say User A was able to pay for it and before the payment step the stock value is checked before committing to change it from 1 to 0.

3. If the version numbers match the operation is committed and the item goes out of stock.

4. Now all the other users that try to purchase that the item will be warned about that the item is no longer available, right at the moment they try to buy, pay or add it to their baskets.