- Cascadless Schedule

eg:

T1	T2	T3
R(a)		
R(b)		
W(a) →		
	→ R(a)	
	W(a) →	
		→ R(a)

In the above example T2 is reading a value written by T1 that means T2 is dependend on T1, T3 is reading a value written by T2 that means T3 is dependend on T2

Now suppose at point T1 fails then T1 must rollback, T2 is depending on T1, T2 has to rollback. and similarly T3 has to rollback.

This concept in which single transaction failure results in a series of Transaction rollback is called cascading rollback

If there is no cascading rollback that schedule is called as cascadless schedule

Imp Condition 1:

In any schedule transaction Ti is performing inital read on same data then it is another schedule same transaction Ti should perform inital head on same data.

Condition 2:

If ~~any~~ in any schedule transaction Ti is performing final write on same data then in another schedule the final schedule should be perfom by same transaction Ti on same data.

Condition 3.

If any schedule transaction Ti is reading a value written by Transaction Tj, then in another schedule also it must read the value by same transaction (~~itor~~ intermediate read also same)

If all three conditions satisfy by the schedule then we can say that they are view equivalence.

- Recoverable Schedule

If there are two transactions T_1 and T_2 and if T_2 is reading a value written by T_1 then commit operation of T_2 should appear after commit operation of T_1 this type of schedule is called as recoverable schedule

eg:

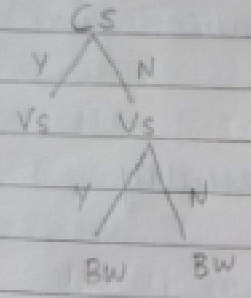
T_1	T_2
R(a)	
W(b)	
	R(b)
	C

- View Serializabillty.

1. If the schedule is conflict serializable then for sure it is view serializable
2. But if a schedule is view serializable then it is not neccasary that the schedule is conflict serializable.
3. If the schedule is not conflict serializable but view serializable then it should have atleast one blind write.

- Blind write

without reading a value if transaction writes it then it is called as blind write. If their is no blind write then it is not view serializable



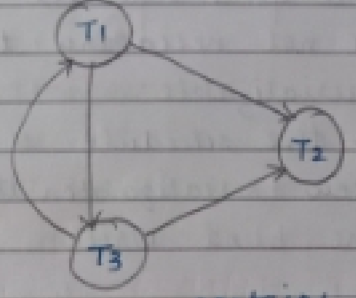
Pure graph mai ek bhi cycle aae to its not a schedule

Two instruction from diff transaction

Imp Q

T₁	T₂	T₃
R(x)		
		R(z)
		W(z)
	R(y)	
R(y)		
	W(y)	
		W(x)
	W(z)	
W(x)		

Check given schedule is conflict serializeable or not



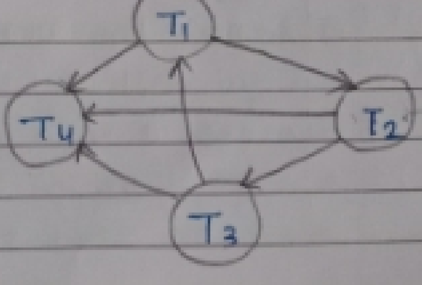
So the graph ~~contents~~ contains ~~as~~. cycle so it is not a serializeable schedule.

Q If the schedule is conflict serilizeable give order of serializebilty

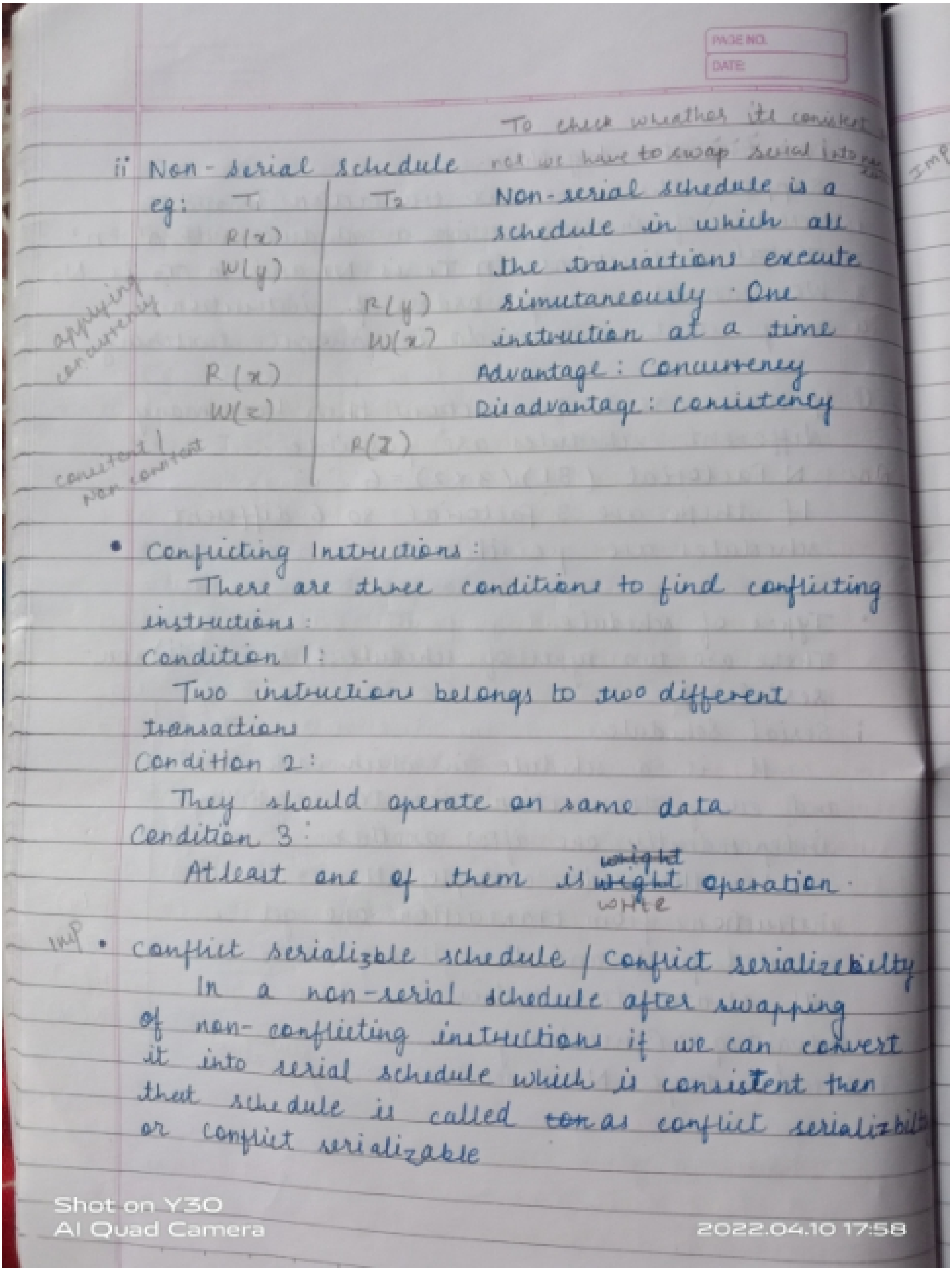
Ans T₁, T₃, T₂ : Order of serializebilty

Q.

T₁	T₂	T₃	T₄
	R(x)		
		W(x)	
W(x)			
	W(y)		
	R(z)		
			R(x)
			R(y)



Order: T₂, T₃, T₁, T₄



PAGE NO.
DATE:

To check wheather its consistent or not we have to swap serial into non ser

ii Non-serial schedule

eg:

T1	T2
R(x)	
W(y)	
	R(y)
	W(x)
R(x)	
W(z)	
	R(z)

applying concurrency

consistent / non constent

Non-serial schedule is a schedule in which all the transactions execute simutaneously. One instruction at a time

Advantage: Concurrency

Disadvantage: consistency

- Conflicting Instructions:

There are three conditions to find conflicting instructions:

Condition 1:

Two instructions belongs to two different transactions

Condition 2:

They should operate on same data

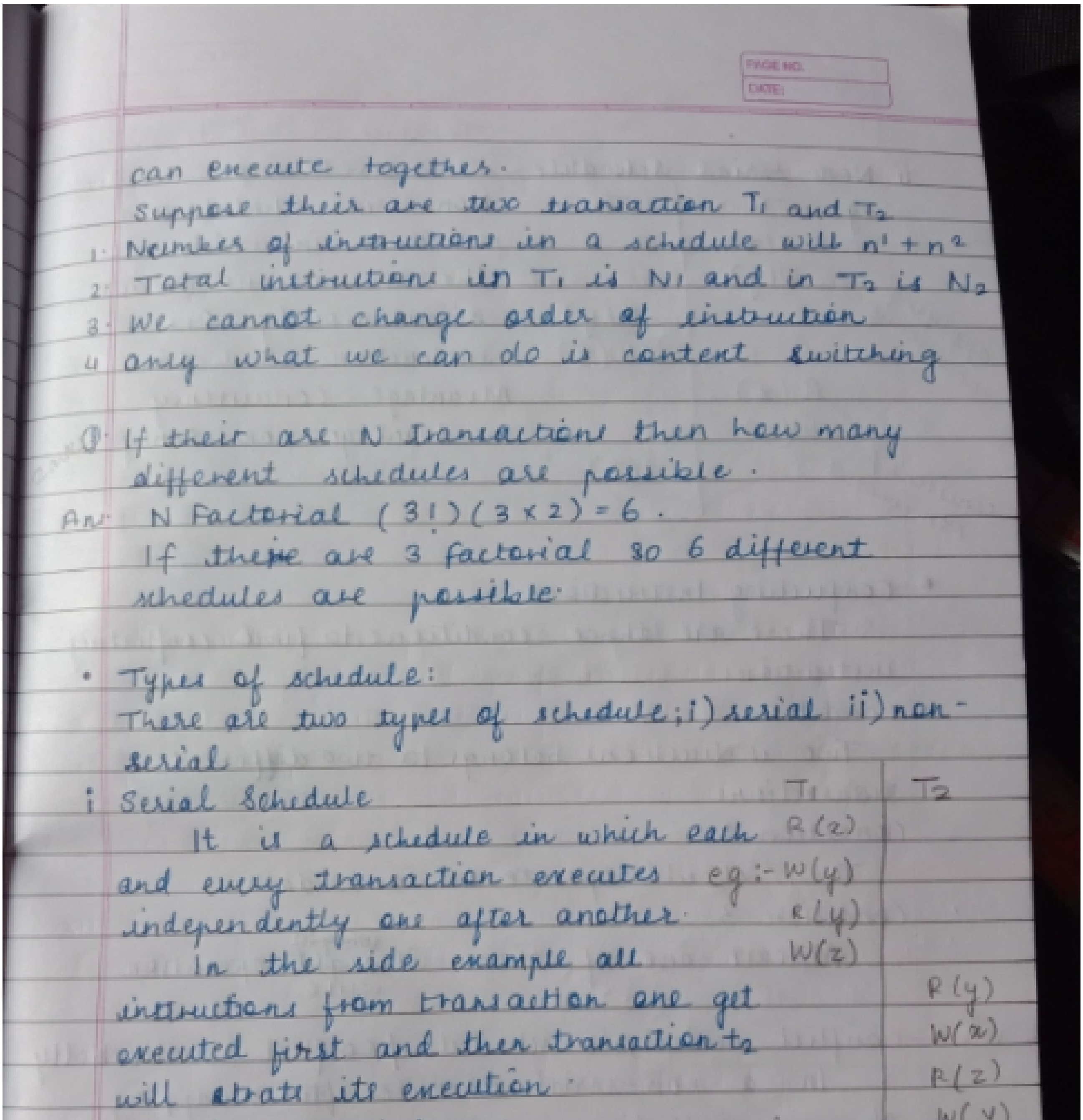
Condition 3:

Atleast one of them is wright ~~weight~~ write operation.

Imp

- conflict serializble schedule / Conflict serializebilty

In a non-serial schedule after swapping of non-conflicting instructions if we can convert it into serial schedule which is consistent then that schedule is called ~~ton~~ as conflict serializbilty or conflict serializable



PAGE NO.
DATE:

can execute together.

Suppose their are two transaction T_1 and T_2

1. Number of instructions in a schedule will $n^1 + n^2$
2. Total instructions in T_1 is N_1 and in T_2 is N_2
3. We cannot change order of instruction
4. only what we can do is content switching

Q. If their are N transactions then how many different schedules are possible.

Ans. N Factorial (3!) (3 × 2) = 6.
If ~~these~~ are 3 factorial so 6 different schedules are possible.

- Types of schedule:

There are two types of schedule; i) serial ii) non-serial

i Serial Schedule

It is a schedule in which each and every transaction executes independently one after another.

In the side example all instructions from transaction one get executed first and then transaction t_2 will strate its execution.

Advantage: Consistency

Disadvantage: No concurrency.

always consistent

eg:-

T_1	T_2
R(z)	
W(y)	
R(y)	
W(z)	
	R(y)
	W(x)
	R(z)
	W(Y)

3. Resource utilization will improve (hardware, software).
4. ~~Effecientiony~~ Entir system will work effeciently.

- Problems which occur due to concurrency.

	Eg: T_1	T_2	
$x = 10$	$W(x)$	$R(x)$	(both the transaction are working on same it
$x = 15$	$W(x$	$R(x$	will occur problem)

1. Unrepeatable Read Problem means transaction itself cannot repeat its read opreation.

2.

	T_1	T_2	
	$W(x)$		
$x = 10$	$R(x)$	$R(x)$	Dirty Read Problem

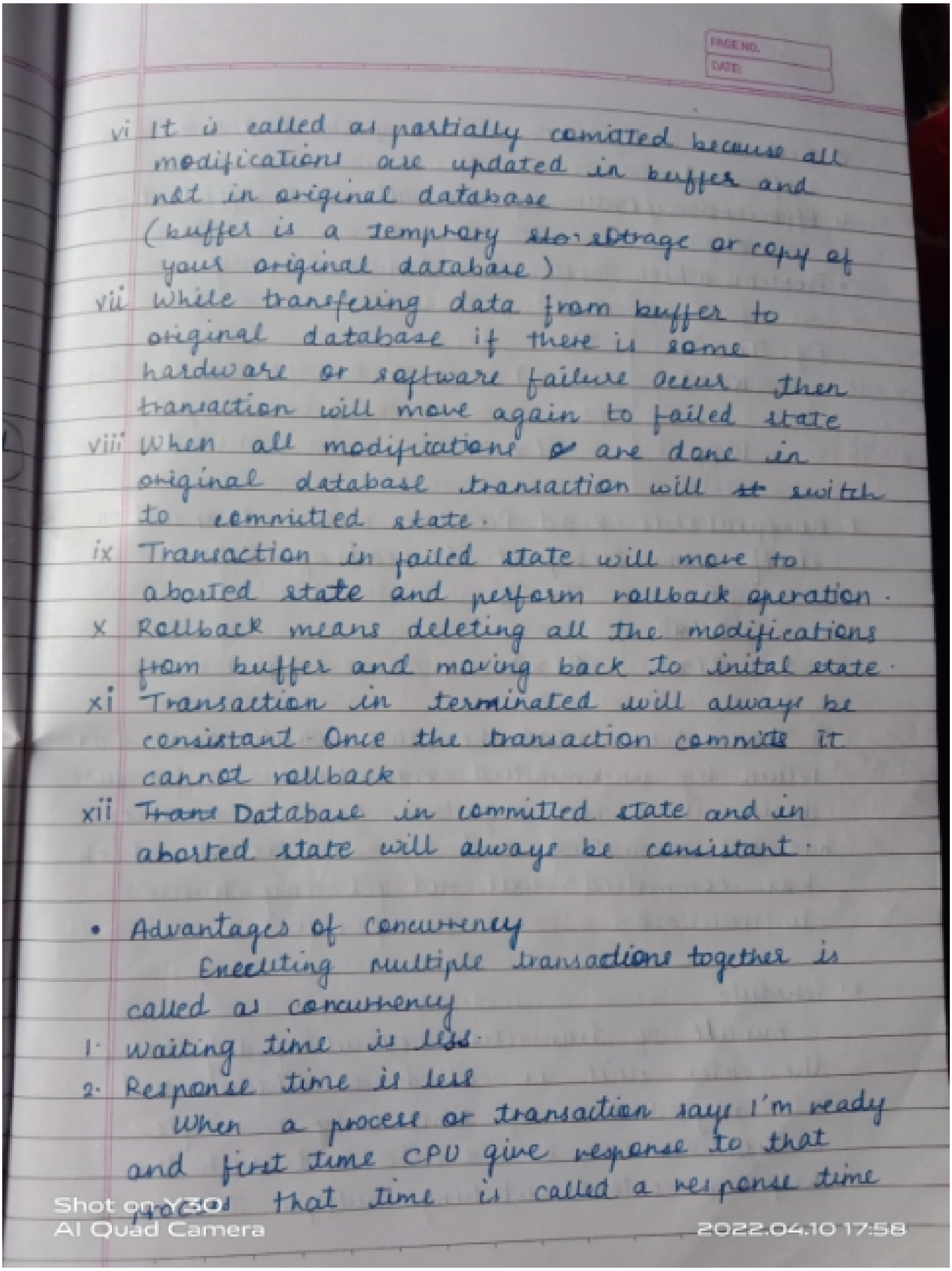
$x = 15$ failure rollback

Jab commit nhi kiye usko chance h rollback karne ka aur jisne kiya usne nahi kar skta

Transaction read the value which ~~cause~~ was return by uncommitted transaction so for sure their is a risk a failure. As the transaction has not committed and the transaction which has committed will not get any chance to rollback.

- Schedule

Bundde of transaction executing together, the entir unit is called as schedule, but at any instant of time only a single instruction can ~~be~~ execute and with the help of content switching multiple transcation



PAGE NO.

DATE:

vi It is called as partially comitted because all modifications are updated in buffer and not in original database
(buffer is a temprary ~~sto~~ storage or copy of your original database)

vii While transfering data from buffer to original database if there is some hardware or software failure Occur then transaction will move again to failed state

viii When all modifications ~~o~~ are done in original database transaction will ~~st~~ switch to committed state.

ix Transaction in failed state will move to aborted state and perform rollback operation.

x Rollback means deleting all the modifications from buffer and moving back to inital state.

xi Transaction in terminated will always be consistant Once the transaction commits it cannot rollback

xii ~~Trans~~ Database in committed state and in aborted state will always be consistant.

- Advantages of concurrency

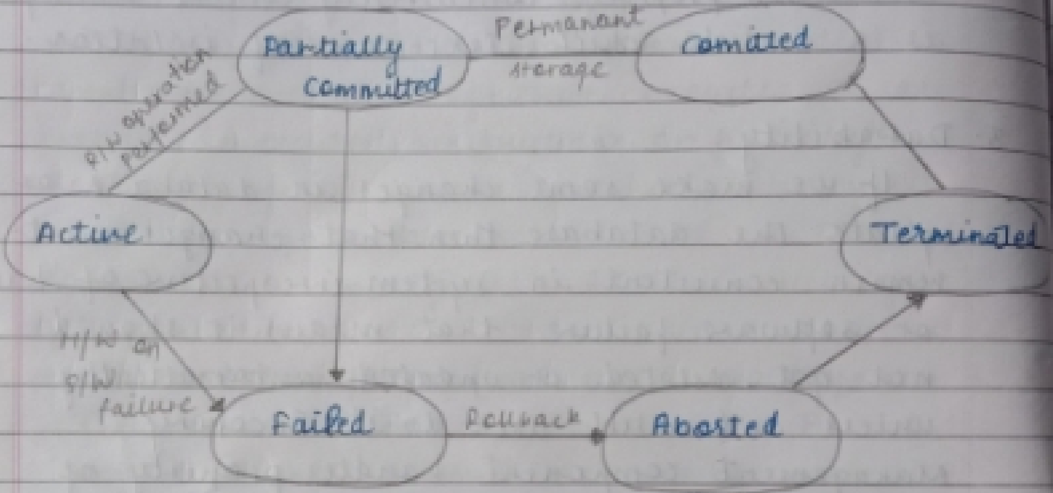
Executing multiple transactions together is called as concurrency

1. waiting time is less.
2. Response time is less

When a process or transaction says I'm ready and first time CPU give response to that process that time is called a response time

- States of Transaction:

Imp 5 mks Q. With neat label diagram explain life cycle of transaction.



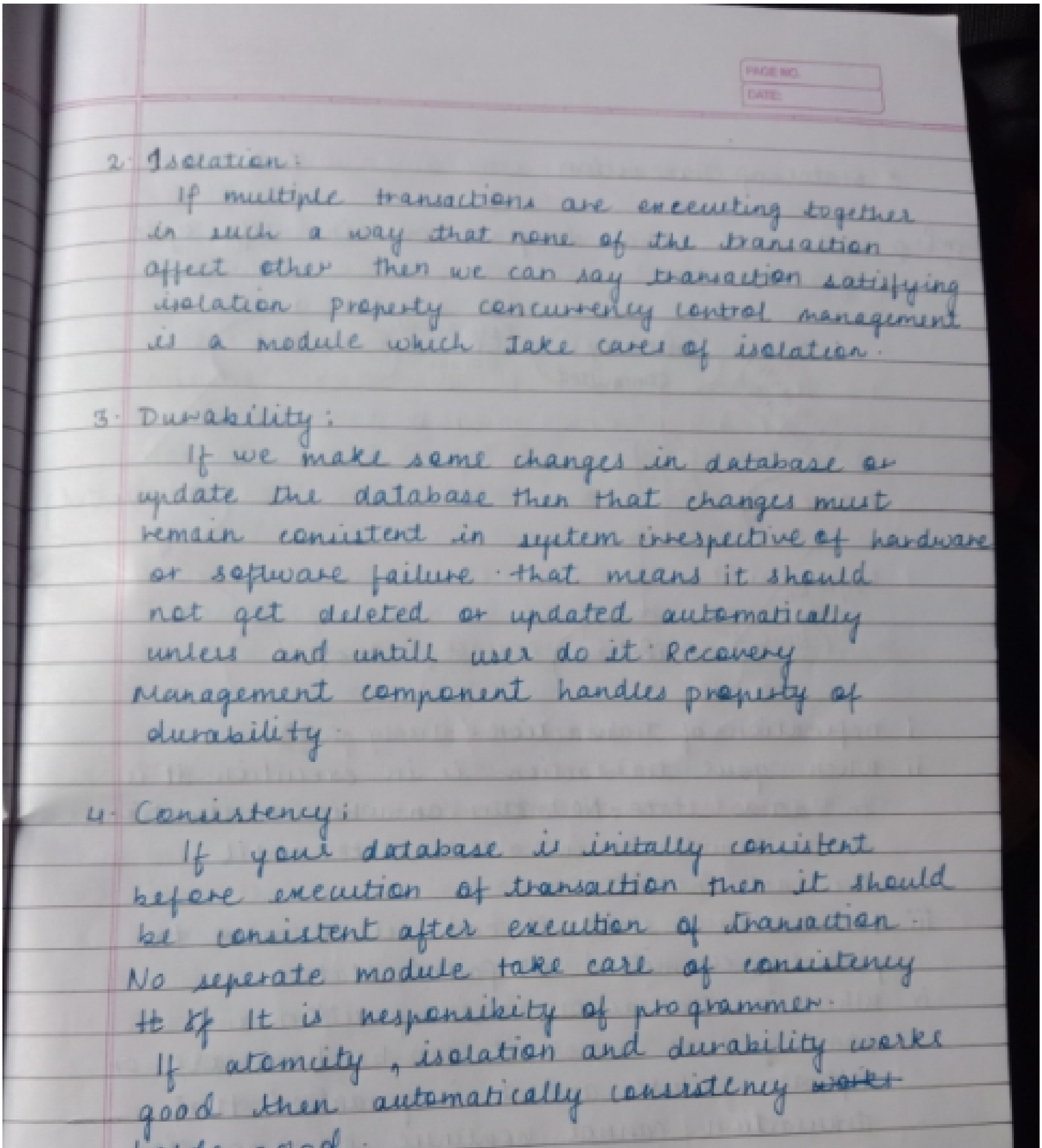
i Defination of Transaction ~~s~~(alrealy given)

ii When your transaction is in execution it is in active state; No matter on which instruction it is, it will be in active state until the last instruction.

iii From active state it can either move to partially committed or failed state.

iv When a transaction is in execution and some failure occur it can be hardware or software failure and it is guranteed that transaction cannot continue its execution again, then it will move to failed state.

v Once all read write operations ~~to~~ perform completely it will switch to pariatlly committed state ~~it is ea~~



2. Isolation:

If multiple transactions are executing together in such a way that none of the transaction affect other then we can say transaction satisfying isolation property concurrency control management is a module which Take cares of isolation.

3. Durability:

If we make some changes in database or update the database then that changes must remain consistent in system irrespective of hardware or software failure. that means it should not get deleted or updated automatically unless and untill user do it. Recovery Management component handles property of durability.

4. Consistency:

If your database is initally consistent before execution of transaction then it should be consistent after execution of transaction. No seperate module take care of consistency ~~It If~~ It is responsibity of programmer.

If atomcity, isolation and durability works good then automatically consistency ~~works~~ holds good.

Unit : 2

Transaction concepts and concurrency Control.

- Transaction is a set of instructions to perform some logical task. This task is atomic in nature i.e either all instructions executes completely or none of them executes at all. Partially executed instructions or partially executed programs instructions or partially executed program are not allowed in DBMS, because it is meanginless to have such type of transactions.

  eg: Person A transfer amount of Rs 100 to person B. The amount is deducted from person A but the amount is not ~~reduced~~ recieved by person B.

IMP • ACID properties of transaction:

There are 4 properties of transaction:

Atomicity ; Consistency ; Isolation ; Durability.

1. Atomcity :

   All instructions must execute completely or none of them executes at all.

Q. which component of databases takes care of atomicity.

Ans Transaction Management Component. It is a small module inside your database.