Q1. A schedule is the representations of execution sequence for all the instructions of the transactions. They are categorized in two types: 1) Serial Schedule.
2) Concurrent Schedule

A schedule is said to be serial if and only if all the instructions of all the transactions get executed non-preemptively as an unit.
Thus for a set of n transactions, there exists n! different valid serial schedules.
Eg for 3 transaction there are 3! = 6 serial schedules.

T1 → T2 → T3.
T1 → T3 → T2.
T2 → T1 → T3
T2 → T3 → T1
T3 → T1 → T2
T3 → T2 → T1.

Also serial schedules gives guarantee for data consistency.

A schedule is said to be concurrent in case the instructions of the transactions get executed preemptively.

When the database system executes several transactions concurrently the corresponding schedule no longer needs to be serial.

Say for eg if two transactions are running concurrently, the OS may execute one transaction for a little while, then perform a context switch, execute the second transaction for some time, and the switch back, so on and so forth. With multiple transactions, the CPU time is shared among all the transactions.

In general it is not possible to predict how many instructions of a transaction will be executed before CPU switches to another transaction i.e. it is much larger than $n!$

Concurrent schedules might get affected with conflicting operations and hence does not give guaranty for data consistency.

Conflicting operations: a pair of operations is said to be conflicting iff :→ they belong to diff. transaction.
- → both operat" are accessing same data item.
- → Atleast one operation is write operation.

Advantages of concurrent schedule : 1.) less waiting time.
2.) Improved response time & throughput.

Disadvantage→ (1) Possibly data inconsistency (2) some time too much context switching

Q2. Functional dependency determines the relation of one attribute to another attribute in a database system. It helps you maintain the quality of data in the database. A functional dependency is denoted by an arrow →.

The Armstrong's axioms are the basic inference rule which are used to conclude functional dependencies on a relational database. They can apply on a functional dependency (FD) to deriude other FD

There are 6 rules:

1.) Reflexive Rule (IR_1): In the reflexive rule, if Y is a subset of X, the X determines Y.

If $Y \subseteq X$ then $X \rightarrow Y$.

2.) Augumentation Rule (IR_2): It is also known as partial dependency if X determines Y, then XZ determines YZ for any Z.

If $X \rightarrow Y$ then $XZ \rightarrow YZ$

3.) Transitive Rule (IR_3): If X determines Y and Y determines Z then X must also determine Z.

If $X \rightarrow Y$ and $Y \rightarrow Z$ then $X \rightarrow Z$.

4.) Union Rule (IR_4):
If X determines Y and X determines Z the X must also determine Y and Z.

Proof: $X \to Y$ and $X \to Z$ then $X \to YZ$

$X \to Y$
$X \to Z$
$X \to XY$ (using augmentation)
$XY \to YZ$ (Using IR_2)
$X \to YZ$ (Using IR_3 on above 2).

5.) Decomposition Rule (IR_5): Also known as project rule, it is reverse of union rule.

If $X \to YZ$ then $X \to Y$ and $X \to Z$

Proof: $X \to YZ$.
$YZ \to Y$ (using IR_1)
$X \to Y$. (Using IR_3 on above 2)

6.) Pseudo transitive Rule (IR_6): If $X \to Y$ and $YZ \to W$ then $XZ \to W$.

proof: $X \to Y$.
$WZ \to Z$.
$WX \to WY$ (Using IR_2)
$WX \to Z$ (using IR_3 on above 2)

Q3 $R = (A, B, C, D, E)$

$FD = (A \to BC, CD \to E, B \to D, E \to A)$

Now,

$A \to BC$, $B \to D$ so it means $A \to D \Rightarrow A \to DC \to E$

$\therefore A \to ABCDE$.

$E \to A$, $A \to ABCDE$, so $E \to ABCDE$

$CD \to E$, so $CD \to ABCDE$.

$B \to D$, $BC \to CD$, so $BC \to ABCDE$

∴ the candidate keys are A, E, CD and BC and any combination of attributes that includes those is a superkey.

~~To check~~ Now, prime attributes → A, C, D, E.

non prime " → B.

R is in first normal form → for each FD, LHS must be CK or SK. or RHS must be prime attribute.

$A \to BC$ ✓ valid. (CK)

$CD \to E$ ✓ valid (CK)

$B \to D$ ✓ valid. (prime attribute)

$E \to A$ ✓ valid. (CK)

⇒ R is In 1st normal form (at last)

Q4 Deadlock refers to a specific condition when two or more processes are each waiting for another to release a resource, or more than two processes are waiting for resources in a circular chain. It is common in multiprocessing where many processes share a specific type of mutually exclusive resource known as a software, lock computers intended for the time sharing and/or real time markets are often equipped with a hardware lock. deadlock are troubling because there is no general solution to avoid deadlocks.

These are the following condition for deadlock:

1.) Mutual exclusion condition: A resource cannot be used by more than one process at a time.

2.) Hold and wait condition: processes already holding resources may request new resources.

3.) No preemption condition: only a process holding a resource may release it.

4.) Circular Wait condition: Two or more processes form a circular chain where each process waits for a resource that the next process in chain holds.

Q5 Serializability is a concept that helps us to check which schedules are serializable. It is the major correctness criterion for concurrent transactions executions. It is considered the highest level of isolation b/w transactions and plays an essential role in concurrency control. Its theory provides the formal framework to reason about and analyze serializability & its techniques. As such it is supported in all general purpose database systems.

	Conflict serializability	View Serializability.
1.)	It is easy to be achieved.	1.) It is hard to be achieved.
2.)	Every conflict serializable is view serializable	2.) not vice versa.
3.)	It is easy to test and cheap	3.) It is not easy & expensive to test
4.)	Most of concurrency control schemes used in practice are based on conflict serializability	4.) not so for view serializability.

(1BEJICS169) Vaibhav Saran

contd
Q3

☆ Check for II normal form!

for each FD, LHS must be proper subset of CK and RHS must be NPA.

A → BC. ×	CD → E ×	B → D ×	E → A
T F	T F	F F	F F

∴ the highest normal form of given functional dependency is 2nd normal form.