USN										

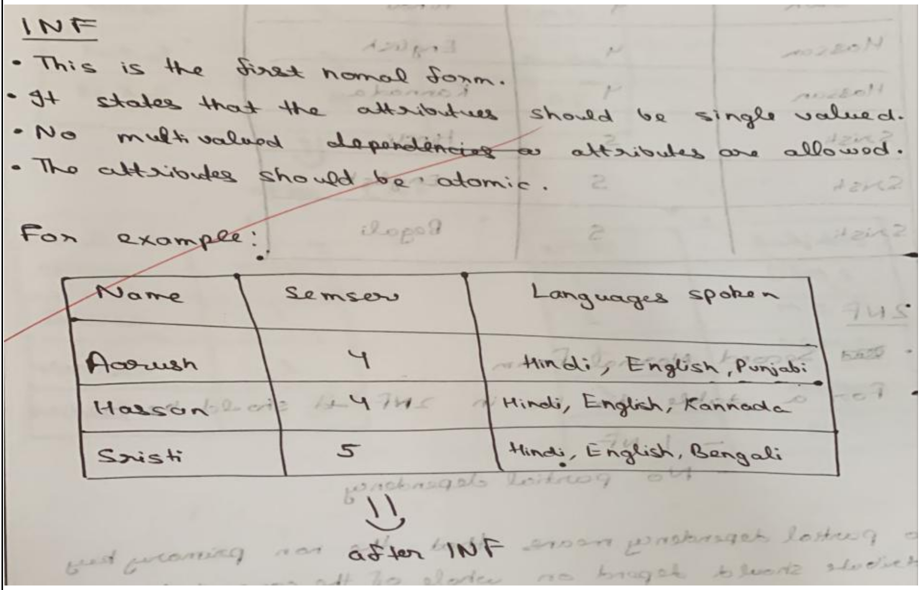
Internal Assessment Test 2 – May 2025

Sub:	Database Management System					Sub Code:	BCS 403		Branch:	AINDS / CS (DS	
Date:	24/05/2025	Duration:	90 minutes	Max Marks:	50	Sem	IV			OBE	
Answer any FIVE Questions									MARKS	CO	RBT

1

What is Normalization and explain different types

Normalization: Database Design Theory – Introduction to Normalization using Functional and Multivalued Dependencies: Informal design guidelines for relation schema, Functional Dependencies, Normal Forms based on Primary Keys, Second and Third Normal Forms, Boyce-Codd Normal Form, Multivalued Dependency and Fourth Normal Form, Join Dependencies and Fifth Normal Form.



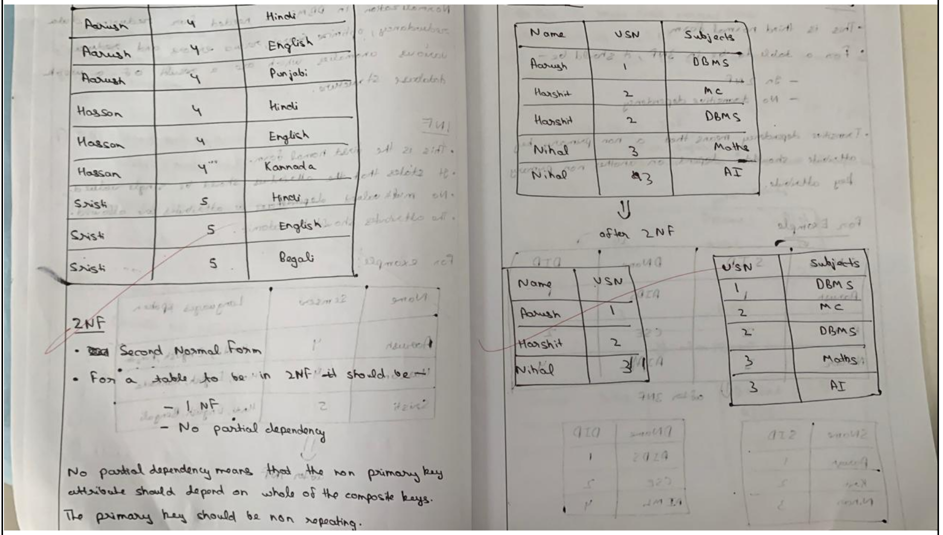
1NF

- This is the first normal form.
- It states that the attributes should be single valued.
- No multivalued ~~dependencies~~ or attributes are allowed.
- The attributes should be atomic.

For example:

Name	Semser	Languages spoken
Aarush	4	Hindi, English, Punjabi
Hassan	4	Hindi, English, Kannada
Sristi	5	Hindi, English, Bengali

after 1NF



Aarush	4	Hindi
Aarush	4	English
Aarush	4	Punjabi
Hasson	4	Hindi
Hassan	4	English
Hassan	4	Kannada
Sristi	5	Hindi
Sristi	5	English
Sristi	5	Begali

2NF

- Second Normal Form
- For a table to be in 2NF it should be:–
  - 1 NF
  - No partial dependency

No partial dependency means that the non primary key attribute should depend on whole of the composite keys. The primary key should be non repeating.

Name	USN	Subjects
Aarush	1	DBMS
Harshit	2	MC
Harshit	2	DBMS
Nihal	3	Maths
Nihal	3	AI

after 2NF

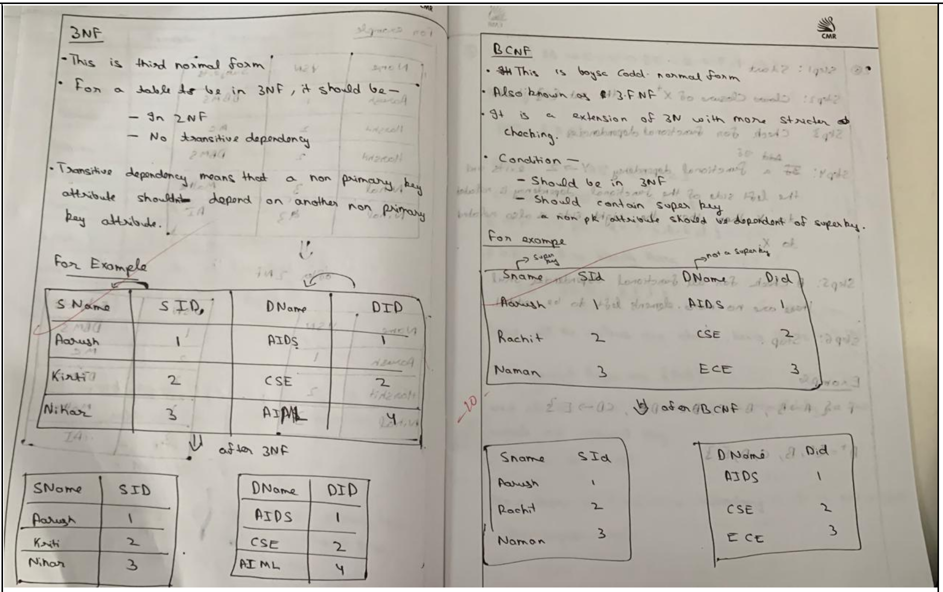
Name	USN
Aarush	1
Harshit	2
Nihal	3

USN	Subjects
1	DBMS
2	MC
2	DBMS
3	Maths
3	AI

10

CO 1

L1

3NF

- This is third normal form
- For a table ~~to~~ be in 3NF, it should be—
  - In 2NF
  - No transitive dependency
- Transitive dependency means that a non primary key attribute ~~shouldn't~~ depend on another non primary key attribute.

For Example

S Name	S ID,	D Name	D ID
Aarush	1	AIDS	1
Kirti	2	CSE	2
Nikar	3	AIML	4

⇓ after 3NF

SName	SID
Aarush	1
Kriti	2
Nihar	3

DName	DID
AIDS	1
CSE	2
AIML	4

—10

BCNF

- ~~It~~ This is boyce Codd normal form
- Also known as 3.F NF
- It is a extension of 3N with more stricter checking.
- Condition —
  - Should be in 3NF
  - Should contain super key
  - a non pk attribute should be dependent of super key.

For exampe

Sname (super key)	SId	DName (not a super key)	Did
Aarush	1	AIDS	1
Rachit	2	CSE	2
Naman	3	ECE	3

⇓ after BCNF

Sname	SId
Aarush	1
Rachit	2
Naman	3

DName	Did
AIDS	1
CSE	2
ECE	3

2	a	Write an algorithm to find the closure of functional dependency 'F' ? To find the closure of a set of functional dependencies F, we need to determine the set of all attributes that can be functionally determined by a given set of attributes. This is typically denoted as X+ for a set of attributes X. ● Algorithm to Find the Closure of Functional Dependency F ● Input: A set of attributes X A set of functional dependencies F ● Output: The closure of X (denoted as X+) Steps: ● Initialize the closure: ○ Start with the closure X+ containing all attributes in X. ● Iterate through functional dependencies: ○ For each functional dependency A → B in F, if A is a subset of X+, add all attributes in B to X+. ● Repeat until no new attributes are added: ○ Continue the process until X+ no longer changes. ● Return the closure: ○ The final set X+ is the closure of X.	6	CO 2	L1

b

R(ABCDEF) Check the highest Normal Form using closure Algorithm ?
FD { AB->C,C->DE,E->F,F->A}.

Step 1: Candidate Key

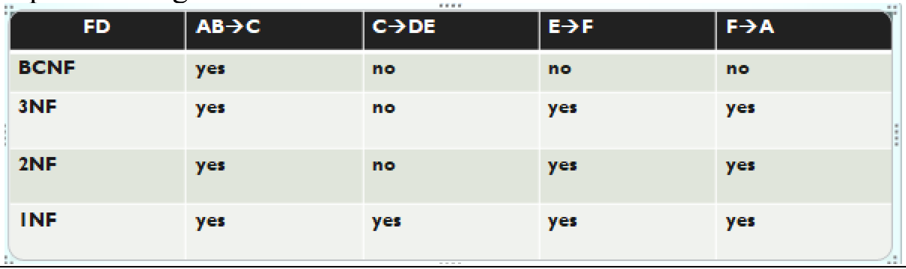
First, we need to determine the candidate key(s) for the relation. We can do this by finding the closure of different attribute sets to see which ones determine all attributes in the relation.

Find Closure of AB:

- Start with AB: {A,B}
- AB→C:{A,B,C}
- C→DE:{A,B,C,D,E}
- E→F:{A,B,C,D,E,F}
- Closure of AB is {A,B,C,D,E,F}. Hence, AB is a candidate key.

- Since AB is a candidate key and it alone determines all attributes, no other subsets are candidate keys.
- C.K□{AB
  AB+=ABCDEF
  FB
FB+=FBACDE
  EB
EB+=EBFACD
  CB}
CB+=CBDEFA
- Step 2: Write All Prime Attributes
  {A,B,C,E,F}
- Step 3: Write All Non-Prime Attributes
  { D }

Step 4:-Finding FD's



FD	AB→C	C→DE	E→F	F→A
BCNF	yes	no	no	no
3NF	yes	no	yes	yes
2NF	yes	no	yes	yes
1NF	yes	yes	yes	yes

CO 4

L3

3	a	Explain the basic data types available for attributes in SQL? . Numeric Data Types Used to store numbers (both integers and floating-point numbers). ● INT / INTEGER: Whole numbers (e.g., 1, 100, -5). ● SMALLINT: Smaller range of integers, consumes less storage. ● BIGINT: Larger range of integers, for very large numbers. ● DECIMAL(p, s) / NUMERIC(p, s): Fixed-point numbers with precision p (total digits) and scale s (digits after the decimal). ● FLOAT / REAL / DOUBLE PRECISION: Approximate, floating-point numbers. Used for scientific or imprecise calculations. 2. Character/String Data Types Used to store text. ● CHAR(n): Fixed-length character string. Always stores exactly n characters (padded with spaces if shorter). ● VARCHAR(n): Variable-length character string with a maximum of n characters. More flexible than CHAR. ● TEXT (or CLOB): Large blocks of text. Used when the size of the string may exceed VARCHAR lim 3. Date and Time Data Types Used to store temporal data. ● DATE: Stores date values (year, month, day). ● TIME: Stores time of day (hour, minute, second). ● TIMESTAMP: Stores both date and time.	5	CO 3	L

		● INTERVAL: Represents a duration (e.g., 5 days, 3 hours). 4. Boolean Data Type Used to store truth values. ● BOOLEAN: Stores TRUE, FALSE, or NULL. 5. Binary Data Types Used to store binary data like images, files, or multimedia. ● BINARY(n): Fixed-length binary data. ● VARBINARY(n): Variable-length binary data. ● BLOB: Binary Large Object, used for large binary files. 6. Other Data Types (Database-Specific) Some databases offer additional types: ● ENUM: A predefined set of string values (e.g., MySQL). ● UUID: Universally Unique Identifier. ● JSON / XML: For storing semi-structured data.			

b

In SQL and PL/SQL, a cursor is a pointer that allows you to retrieve, manipulate, and traverse through rows returned by a query one at a time. Cursors are essential when you need to process individual rows of a query result, especially in procedural operations.

Types of Cursors in PL/SQL

1. Implicit Cursor

- Automatically created by Oracle/SQL engine for single SQL statements such as SELECT INTO, INSERT, UPDATE, DELETE.
- You don't need to declare or open/close it.
- Managed internally.

◆ Syntax & Example (Implicit Cursor):

```
DECLARE
   emp_name employees.first_name%TYPE;
   emp_id   employees.employee_id%TYPE := 101;
BEGIN
   SELECT first_name INTO emp_name
   FROM employees
   WHERE employee_id = emp_id;

   DBMS_OUTPUT.PUT_LINE('Employee Name: ' || emp_name);
END;
```

- This uses an implicit cursor for the SELECT INTO statement.

◆ Cursor Attributes (for implicit cursors):

- %FOUND: Returns TRUE if DML affected one or more rows.
- %NOTFOUND: Returns TRUE if DML affected no rows.
- %ROWCOUNT: Returns the number of rows affected.
- %ISOPEN: Always FALSE for implicit cursors.

CO 3

L1

2. Explicit Cursor

- Declared by the programmer when a query returns multiple rows.
- Allows row-by-row processing using OPEN, FETCH, CLOSE.

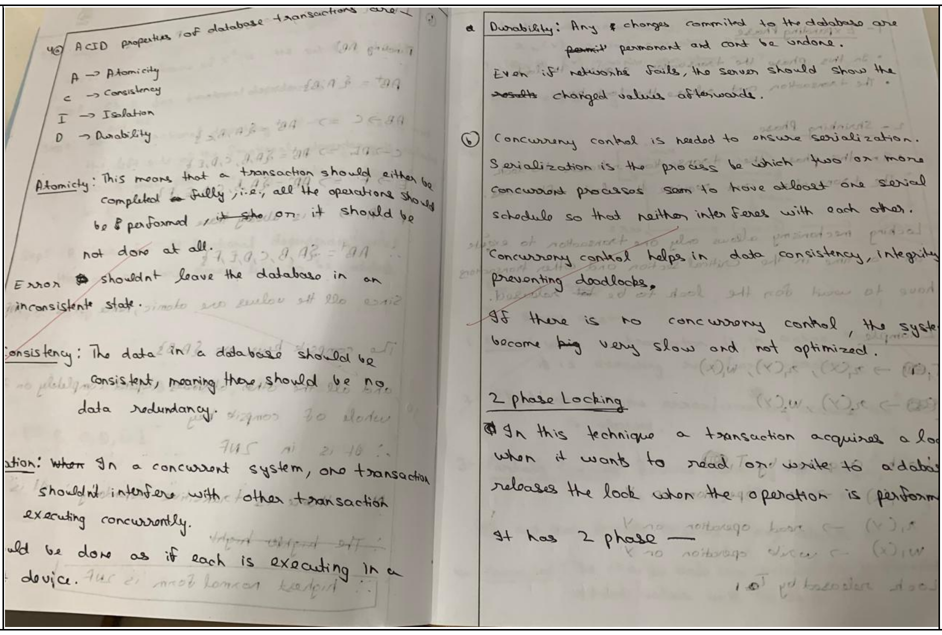
🔹 Syntax:

```
DECLARE
   CURSOR emp_cursor IS
      SELECT first_name, salary FROM employees WHERE
department_id = 10;

   v_name   employees.first_name%TYPE;
   v_salary employees.salary%TYPE;
BEGIN
   OPEN emp_cursor;
   LOOP
      FETCH emp_cursor INTO v_name, v_salary;
      EXIT WHEN emp_cursor%NOTFOUND;
      DBMS_OUTPUT.PUT_LINE('Name: ' || v_name || ', Salary: ' ||
v_salary);
   END LOOP;
   CLOSE emp_cursor;
END;
```

🔹 Cursor Attributes (for explicit cursors):

- cursor_name%FOUND
- cursor_name%NOTFOUND
- cursor_name%ROWCOUNT
- cursor_name%ISOPEN

4			10	CO 1	L2

4(a) ACID properties of database transactions are -

A → Atomicity
C → Consistency
I → Isolation
D → Durability

Atomicty: This means that a transaction should either be completed fully, i.e., all the operations should be performed, ~~it sho~~ or it should be not done at all.

Error shouldnt leave the database in an inconsistente state.

onsistency: The data in a database should be consistent, meaning there should be no data redundancy.

ation: ~~When~~ In a concurrent system, one transaction shouldn't interfere with other transaction executing concurrently.

uld be done as if each is executing in a device.

Durability: Any changes commited to the database are ~~permit~~ permanant and cant be undone.

Even if networks fails, the server should show the ~~results~~ changed values afterwards.

(b) Concurreny control is needed to ensure serialization. Serialization is the process be which two or more concurrent processes seem to have atleast one serial schedule so that neither interferes with each other.

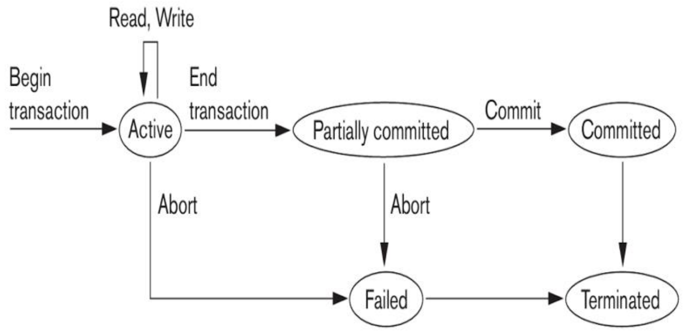
Concurrency control helps in data consistency, integrity preventing deadlocks.

If there is no concurreny control, the syste become ~~big~~ very slow and not optimized.

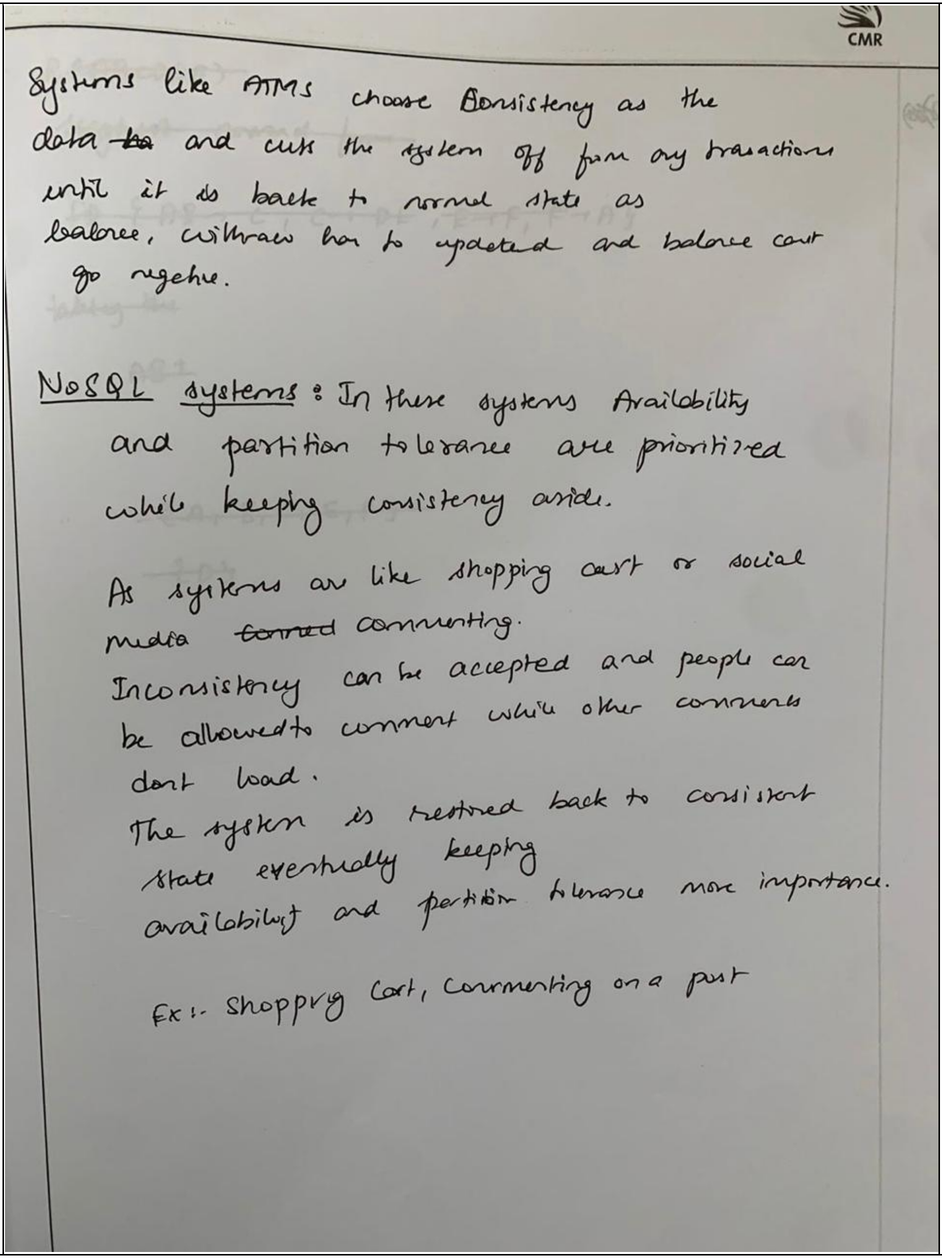
2 phase Locking

In this technique a transaction acquires a lo when it wants to read or write to a databa releases the lock when the operation is perform

It has 2 phase —

5	a	With a neat diagram, Explain the various states of a transaction execution.  Read, Write Begin transaction Active End transaction Partially committed Commit Committed Abort Abort Failed Terminated **Figure 20.4** State transition diagram illustrating the states for transaction execution. • BEGIN_TRANSACTION:-This marks the beginning of transaction execution. • READ or WRITE.:-These specify read or write operations on the database items that are executed as part of a transaction. • END_TRANSACTION. This specifies that READ and WRITE transaction operations have ended and marks the end of transaction execution. • However, at this point it may be necessary to check whether the changes introduced by the transaction can be permanently applied to the database (committed) or whether the transaction has to be aborted because it violates Serializability or for some other reason. • COMMIT_TRANSACTION. This signals a successful end of the transaction so that any changes (updates) executed by the transaction can be safely committed to the database and will not be undone. • ROLLBACK (or ABORT). This signals that the transaction has ended unsuccessfully, so that any changes or effects that the transaction may have applied to the database must be undone. • Active State: • Description: Transaction begins execution, able to perform READ and WRITE operations on the database. • Transition: Moves to the next state after starting execution. Partially Committed State: • Description: Transaction prepares to commit changes. Additional checks may be performed by concurrency control protocols. • Commit Point: Ensures changes can be permanently recorded, often by logging them in the system log.	10	CO 5	L2

		• Transition: Moves to committed state if checks are successful. Committed State: • Description: Transaction successfully completes execution. • Outcome: All changes made by the transaction are permanently recorded in the database, even in case of system failure. Failed State: • Description: Transaction fails due to unsuccessful checks or being aborted during execution. • Rollback: Requires undoing WRITE operations on the database to maintain consistency. • Potential Restart: Failed transactions may be restarted later as new transactions. Terminated State: • Description: Transaction leaves the system after completion. • Cleanup: Information maintained in system tables during transaction execution is removed upon ter			



Systems like ATMS choose Consistency as the data ~~ka~~ and cuts the system off from any transactions until it is back to normal state as balance, withraw has to updated and balance cant go negative.

NoSQL systems: In these systems Availability and partition tolerance are prioritized while keeping consistency aside.

As systems are like shopping cart or social media ~~correct~~ commenting.
Inconsistency can be accepted and people can be allowed to comment while other comments dont load.
The system is restored back to consistent state eventually keeping availability and partition tolerance more importance.

Ex:- Shopping Cart, Commenting on a post

eCheck whether the below schedule is conflict serializable or not
{b2, r2(X), b1, r1(X), w1(X), r1(Y), w1(Y), w2(X), e1, c1, e2, c2}
To determine whether the given schedule is conflict serializable, we need to:

1. Understand the operations and transactions involved.
2. Build a precedence graph (also known as a serializability graph).
3. Check for cycles — if the graph has no cycles, the schedule is conflict serializable.

Step 1: Parse the schedule

Schedule:
{b2, r2(X), b1, r1(X), w1(X), r1(Y), w1(Y), w2(X), e1, c1, e2, c2}

Let's break it down by transaction:

- T1: b1, r1(X), w1(X), r1(Y), w1(Y), e1, c1
- T2: b2, r2(X), w2(X), e2, c2

Step 2: Identify conflicting operations

6

The CAP Theorem, proposed by Eric Brewer, states that in a distributed database system, it is impossible to simultaneously guarantee all three of the following properties:

1. Consistency (C):
   Every read receives the most recent write or an error.
   ➤ *Same data across all nodes at any time.*

2. Availability (A):
   Every request (read/write) gets a response (success or failure), even if some nodes are down.
   ➤ *System remains responsive.*

3. Partition Tolerance (P):
   The system continues to operate even if there is a communication breakdown between nodes in the network.
   ➤ *Handles network failures gracefully.*

```
 Consistency
       /\
      /  \
     /    \
Availability ---- Partition Tolerance
```

Combinations Allowed by CAP:

Combination	Description
CP (Consistency + Partition Tolerance)	System remains consistent and tolerates network failures, but may become unavailable during partition.
AP (Availability + Partition Tolerance)	System is always available and partition-tolerant, but may not always show the most recent data (eventual consistency).
CA (Consistency + Availability)	System is consistent and available only if there is no network partition, which is unrealistic in distributed systems.

Most Important in NoSQL

10

CO 3

L3

NoSQL systems generally prioritize:

Availability (A)

- NoSQL databases aim for high availability, especially in web-scale systems like social networks and e-commerce.

Partition Tolerance (P)

- In distributed environments (e.g., cloud, data centers), network failures (partitions) are common, so tolerance is a must. Consistency is often relaxed (i.e., eventual consistency is used).

Examples of NoSQL Systems and CAP Choices:

NoSQL System	CAP Model
MongoDB	AP (eventual consistency)
Cassandra	AP
HBase	CP
CouchDB	AP

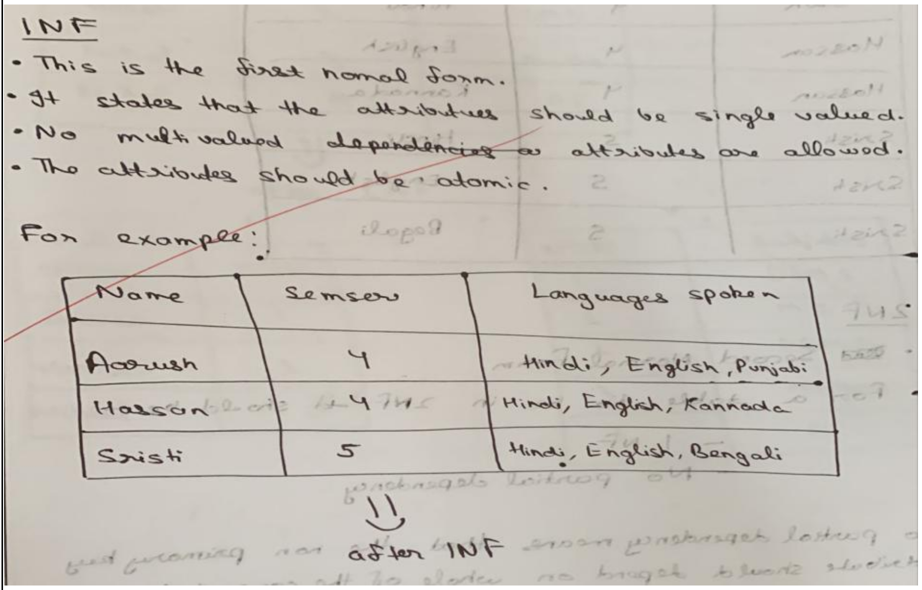
USN | | | | | | | | | | |

Internal Assessment Test 2 – May 2025

Sub:	Database Management System	Sub Code:	BCS 403	Branch:	AINDS / CS (DS

Date:	24/05 /2025	Duration:	90 minutes	Max Marks:	50	Sem	IV	OBE	

Answer any FIVE Questions			MARKS	CO	RBT
1		**What is Normalization and explain different types** **Normalization:** Database Design Theory – Introduction to Normalization using Functional and Multivalued Dependencies: Informal design guidelines for relation schema, Functional Dependencies, Normal Forms based on Primary Keys, Second and Third Normal Forms, Boyce-Codd Normal Form, Multivalued Dependency and Fourth Normal Form, Join Dependencies and Fifth Normal Form.	10	CO 1	L1



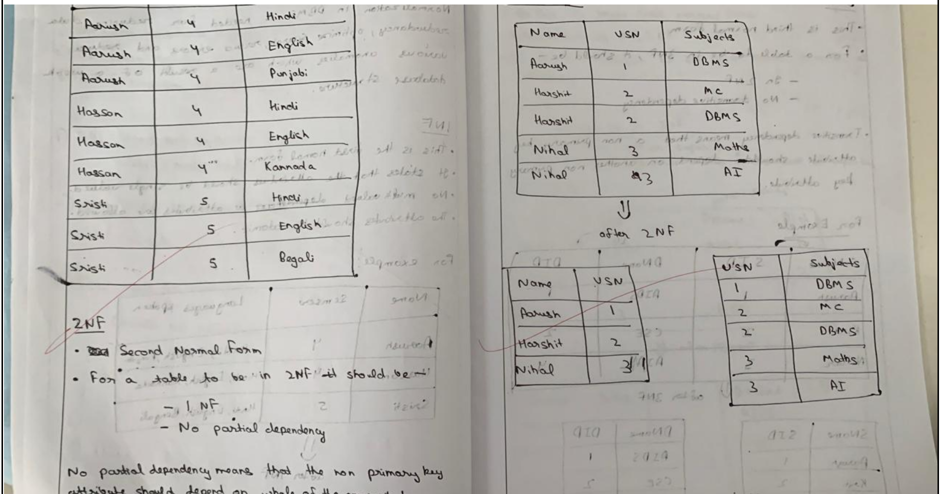
1NF

- This is the first normal form.
- It states that the attributes should be single valued.
- No multi valued dependencies ~~a~~ attributes are allowed.
- The attributes should be atomic.

For example:

Name	Semser	Languages spoken
Aarush	4	Hindi, English, Punjabi
Hasson	4	Hindi, English, Kannada
Sristi	5	Hindi, English, Bengali

after 1NF



Aarush	4	Hindi
Aarush	4	English
Aarush	4	Punjabi
Hasson	4	Hindi
Hasson	4	English
Hassan	4	Kannada
Sristi	5	Hindi
Sristi	5	English
Sristi	5	Begali

2NF

- Second Normal Form
- For a table to be in 2NF it should be –
  - 1 NF
  - No partial dependency

No partial dependency means that the non primary key attribute should depend on whole of the

Name	USN	Subjects
Aarush	1	DBMS
Harshit	2	MC
Harshit	2	DBMS
Nihal	3	Maths
Nihal	3	AI

after 2NF

Name	USN
Aarush	1
Harshit	2
Nihal	3

USN	Subjects
1	DBMS
2	MC
2	DBMS
3	Maths
3	AI

CI	CCI	HOD