SPRING END SEMESTER EXAMINATION-2019

4th Semester B.Tech & B.Tech Dual Degree

DATABASE MANAGEMENT SYSTEM

CS-2004/CS-402

(For 2017 & Previous Admitted Batches)

Time: 3 Hours Full Marks: 50/60

Answer any SIX questions.
Question paper consists of four sections-A, B, C, D.
Section A is compulsory.
Attempt minimum one question each from Sections B, C, D.
The figures in the margin indicate full marks.
Candidates are required to give their answers in their own words as far as practicable and all parts of a question should be answered at one place only.

SECTION-A

1. [1]

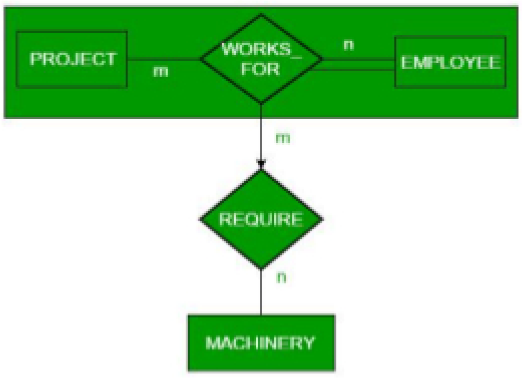
(a) State the difference between data, database and database management systems.
Ans:
Data is any sort of raw or unprocessed fact. Data itself is not of any use as it is in raw form. When processed, data provides information. Information is useful in real world applications.
Database is a computer based organized collection of inter-related data. More specifically, database is an electronic system that allows data to be stored, easily accessed, manipulated, and updated.
DBMS is a general purpose software system that facilitates the process of defining, constructing, manipulating, and sharing DB among various users and applications.
Hence, database consists of data and in order to perform several data related operations on the database, we need a software which is known as DBMS.

(b) Differentiate between schema and instance.
Ans:
Schema is the physical description of the database. On the other hand, Instance is the set of information currently stored in the database at a specific time.
Schema does not changes too often whereas, instance changes very frequently in the database

(c) What is the use of data dictionary in DBMS?
Ans:
Data dictionary describes every data element in the database. It enables all users to share a common view of the data source.

(d) What is aggregation? How is it represented using ER diagram? Give example to support your answer.
Ans:
An ER diagram is not capable of representing relationship between an entity and a relationship which may be required in some scenarios. In those cases, a relationship

with its corresponding entities is aggregated into a higher level entity. For Example, Employee working for a project may require some machinery. So, REQUIRE relationship is needed between relationship WORKS_FOR and entity MACHINERY. Using aggregation, WORKS_FOR relationship with its entities EMPLOYEE and PROJECT is aggregated into single entity and relationship REQUIRE is created between aggregated entity and MACHINERY.



Aggregation

(e) What are the DBMS keys? Explain in brief.
Ans:
Key plays an important role in relational database; it is used for identifying unique rows from table. In DBMS, following types of key exist:
Super key is a set of one or more attributes that allow us to identify uniquely an entity in an entity set or a tuple in a relation i.e. table.
Candidate key is a minimal set of super key which cannot have any columns removed from it without losing the unique identification property. This property is sometimes known as minimality or (better) irreducibility. Every candidate key is super key, but the reverse is not always true i.e. every super key is not a candidate key.
Primary key is a candidate key that is chosen by the database designer as the principal means of identifying tuples in a table.
Alternate key: Out of all candidate keys, only one gets selected as primary key, remaining keys are known as alternate keys.
Foreign keys are the columns of a table that points to the primary key of another table. They act as a cross-reference between tables.

(f) Compute the outer join of the following r(R) and s(S).
Ans:

Left Outer Join

A	**B**	**C**	**D**
10	42	22	44
10	42	22	66
30	52	33	NULL

Right Outer Join

A	**B**	**C**	**D**
10	42	22	44
10	42	22	66
NULL	NULL	11	55

Full Outer Join

A	B	C	D
10	42	22	44
10	42	22	66
30	52	33	NULL
NULL	NULL	11	55

(g) What is trivial functional dependency?

Ans:

A functional dependency X→Y is a trivial functional dependency, if Y is a subset or equal of X.

For example, {Name, Course}→Course. Here, the dependent is the subset of determinant. Hence, this dependency is trivial in nature.

(h) Write the ACID properties of transactions with citing one example of "A" property.

Ans:

In transaction processing system, ACID refers to the properties named as Atomicity, Consistency, Isolation, and Durability.

Atomicity: Either all operations of the transaction are reflected properly in the database, or no operation is reflected.

Atomicity requires that all operations of a transaction be completed; if not, the transaction is aborted by rolling back all the updates done during the transaction. For example: In an online money transfer, one cannot deduct money from one account if the money is not deposited in other account. i.e. the transaction must have both deduct(money) from account A and deposit(money) to account B.

(i) What do you mean by serializability?

Ans:

Serializability is a concept that helps to identify which non-serial schedules are correct and will maintain the consistency of the database. A serializable schedule behaves exactly like serial schedule. A concurrent schedule is serializable if it is equivalent to a serial schedule.

(j) Define normalization and de-normalization. When can we prefer to de-normalization?

Ans:

Normalization is the process of decomposing or breaking a relational schema R into fragments (i.e. smaller schemas) such that the following conditions hold:

Lossless decomposition: The fragments should contain the same information as the original relation.

Dependency preservation: All the functional dependencies should be preserved within each fragment Ri.

Good form: Each fragment Ri should be free from any type of redundancy.

Denormalization is the opposite of Normalization. It is the process of increasing redundancy in the database either for convenience or to improve performance. It can be preferred in following scenarios:

Maintaining history database and its related operations.

Improving query performance

Speeding up reporting

Computing commonly-needed values up front.

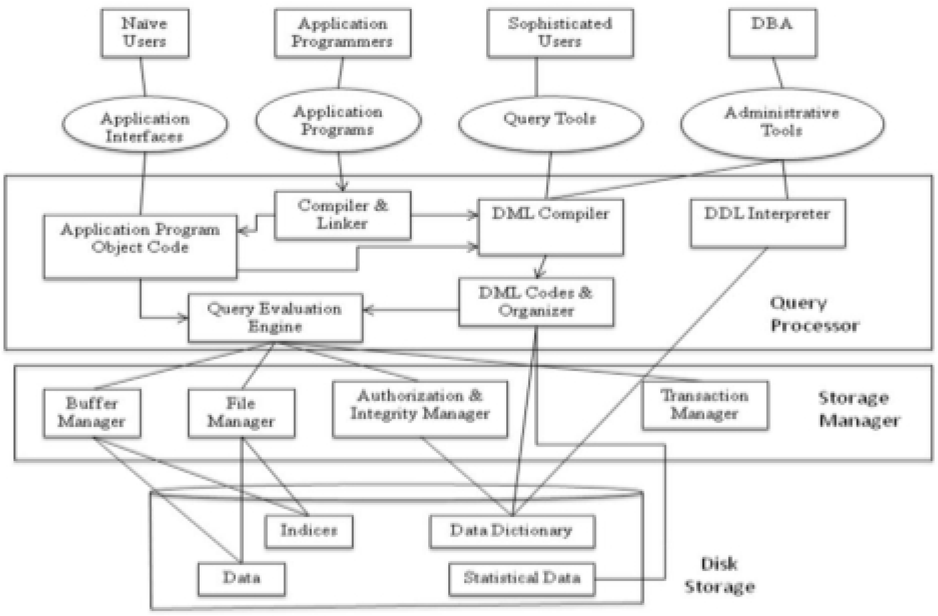
SECTION-B

2. (a) Draw and explain each block of the database management system structure. [4]

Ans:

Diagram [2]

Explanation [2]



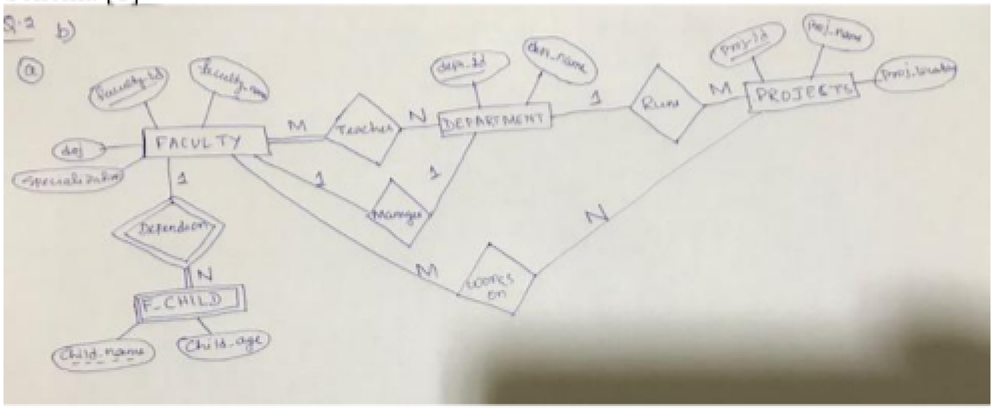
(b) An educational institute database needs to store information about faculty members (identified by faculty-id, with faculty-name, doj, and specialization as attributes); departments (identified by dept-id, with dept-name as attributes); projects (identified by proj-id, with proj-name, proj-location as attributes) and children of faculty members (with child-name and child-age as attributes). A department can have many faculty members and faculty member can teach in more than one department. Faculty members can work on different projects. A department can have many projects and a project can belong to at most one department. Each department is managed by one HOD, who is a faculty member. A child must be identified uniquely by name when the parent (who is a faculty member; assume that only one parent works for institute) is known. We are not interested in information about a child one the parent leaves the institute.

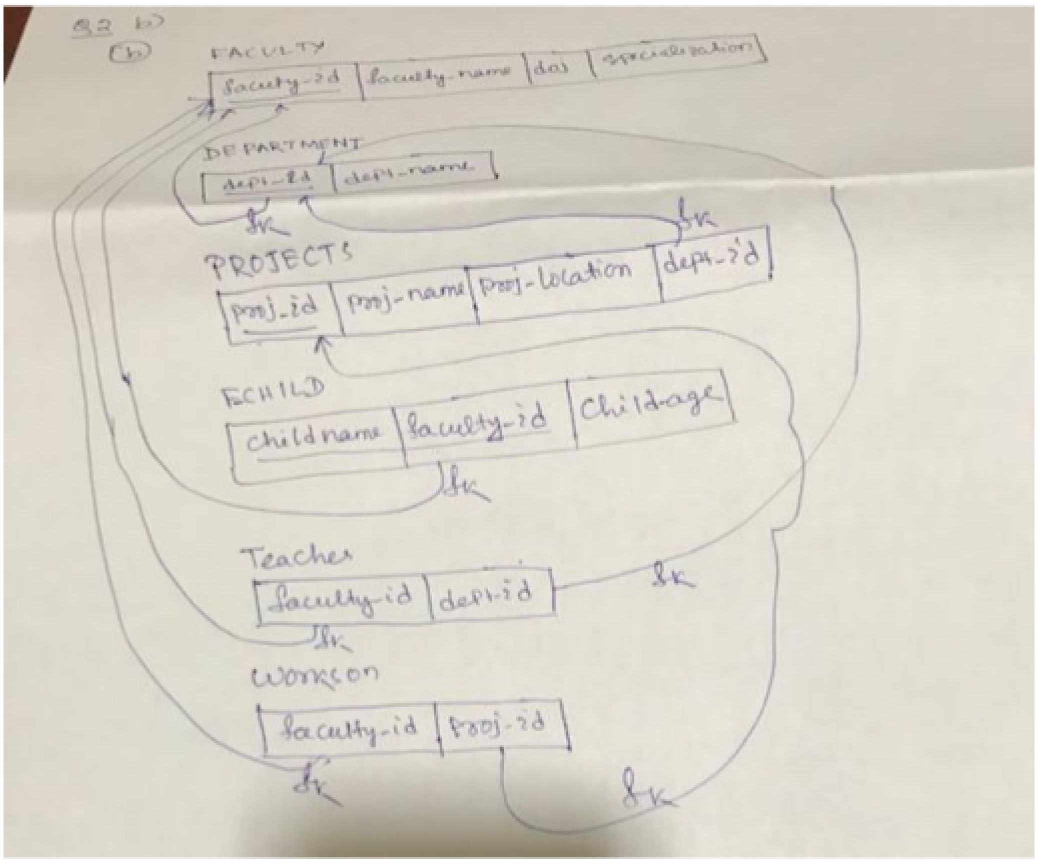
Answer the following questions

i) Draw the ER diagram that captures the above information.

ii) Translate the ER diagram into relations. Also identify the primary key and foreign keys.

Ans:
Diagram [3]
Schema [1]





3. (a) What do you mean by integrity constraints? Explain each with proper example. [4]

Ans:

Integrity constraints ensure that changes (update deletion, insertion) made to the database by authorized users do not result in a loss of data consistency.

The major types of integrity constraints are

Domain Constraints: All the values that appear in a column of a relation must be taken from the same domain. This constraint can be applied by specifying a particular data type to a column. [1]

e.g age attribute in a relation can't be character type as the data type can either be number or integer.

Entity Integrity: The entity integrity rule is designed to assure that every relation has a primary key, and that the data values for that primary key are all valid. Usually, the primary key of each relation is the first column. Entity integrity guarantees that every primary key attribute is NOT NULL. Primary key performs the unique identification function in a relational model. [1]

e.g In student relation Stud_roll can be taken as primary key because each student must have a roll number.

Referential Integrity: In relational data model, associations between tables are defined by using foreign keys. A referential integrity constraint is a rule that maintains consistency among the rows of two relations.The rule states that if there is a foreign key in one relation, either each foreign key value must match a primary key value in the other table or else the foreign key value must be NULL.A foreign key that references its own relation is known as recursive foreign key. The linking between the foreign key and primary key allows a set of relations to form an integrated database. [1]

e.g Student=(Stud_roll, name, age, Dept_id)
Department(Dept_id, dept_name)
here Dept_id is a primary key in department relation and referred as foreign key in student relation.

Operational Constraints: These are the constraints enforced in the database by the business rules or real world limitations.

(b) Explain the Armstrong's Axioms and additional inference rules with example. [4]

Ans:
Fundamentals rule [2]
Additional rule [2]

Armstrong's Inference Axioms

The inference axioms or rules allow users to infer the FDs that are satisfied by a relation

Let R(X, Y, Z, W) where X, Y, Z, and W are arbitrary subsets of the set of attributes of a universal relation schema R

The three fundamental inference rules are:

Reflexivity Rule: If Y is a subset of X, then X→Y (Trivial FDs).
Ex:{ Name, Course}→Course

Augmentation Rule: If X→Y, then {X, Z}→{Y, Z}. Ex: as Prof→Grade, therefore {Prof, Major}→{Grade, Major}

Transitivity Rule: If X→Y and Y→Z, then X→Z.
Ex: Course→Name and Name→Phone_no functional dependencies are present, therefore Course→Phone_no.

Additional inference rules are:

Union or Additive Rule: If X→Y and X→Z, then X→{Y, Z}.
Ex: Prof→Grade and Prof→Course FDs are present; therefore, Prof→{Grade, Course}

Decomposition Rule: If X→{Y, Z}, then X→Y and X→Z. Ex: if Prof→{Grade, Course}, then this FD can be decomposed as Prof→Grade and Prof→Course

Composition Rule: If X→Y and Z→W, then {X, Z}→{Y, W}.
Ex: if Prof→Grade and Name→Phone_no, then the FDs can be composed as {Prof, Name}→{Grade, Phone_no}

Pseudo transitivity Rule: If X→Y and {Y, W}→Z, then {X, W}→Z.
Ex: if Prof→Grade and {Grade, Major}→Course, then the FD {Prof, Major}→Course is valid Database Design

SECTION-C

4. Consider the following relations: [4]
PERSON(P_id, F_name, Occupation, Salary, City)
ORDER(O_id, P_id, Item, Quantity, Price, Order_date)
The primary keys are P_id and O_id respectively. Express the following queries in SQL and relational algebra.

(a) Find the person's name and city whose name starts with S.

Ans:

SQL: SELECT f_name||l_name, city FROM Person WHERE f_name LIKE 'S%'; [1]

ALGEBRA: $\pi_{f\_name||l\_name,\ city}\left(\sigma_{f_{name} LIKE 'S\%'}(Person)\right)$ [1]

(b) Find the person with highest salary.

Ans:

SQL: SELECT * FROM Person WHERE salary = (SELECT MAX(salary) FROM Person); [1]

ALGEBRA: $temp \leftarrow g_{MAX(salary)}(Person)$

$\pi_{all}\left(\sigma_{salary=temp}(Person)\right)$ [1]

(c) Find the name of person(s) who have ordered on the same date.

Ans:

SQL: SELECT f_name||l_name FROM Person, Order O1 WHERE Person.p_id=O1.p_id AND O1.order_date = (SELECT O2.order_date FROM Order O2 WHERE O1.p_id ≠ O2.p_id); [1]

ALGEBRA:

$\pi_{f\_name||l_{name}}\left(\sigma_{P.p_{id}=O1.p_{id}}\left[(\rho_P(Person) X \rho_{O1}(Order))\ \overline{\bowtie}\ [\pi_{f\_name||l\_name}(\sigma_{p_{id}=O1.p\_id}(Ord\right.\right.$

[1]

(d) Find the name of person(s) who have not ordered any item.

Ans:

SQL:SELECT f_name||l_name FROM Person MINUS
SELECT f_name||l_name FROM Person, Order WHERE
Person.p_id=Order.p_id; [1]

ALGEBRA: $\pi_{f\_name||l_{name}}(Person) - \pi_{f\_name||l_{name}}(Person \bowtie Order)$ [1]

5. (a) Consider the following relation R(A, B, C, D, E, F, G) with a set of functional dependencies F = {A → BC, B → CD, D → EF, BC → AG, ABG→DF} and find the canonical cover / minimal cover. [4]

Ans:

F=F={A→BC, B→CD, D→EF, BC→AG, ABG→DF}
={A→B, A→C, B→C, B→D, D→E, D→F, B→AG, AG→DF}
={A→B, B→C, B→D, D→E, D→F, B→A, B→G, AG→D, AG→F}
={A→B, B→C, B→D, B→A, B→G, D→E, D→F, AG→D, AG→F}
= {A→B, B→C, B→D, B→A, B→G, D→E, D→F, A→D, A→F}
={A→B, B→C, B→D, B→A, B→G, D→E, D→F}
⇨ Fc={A→B, B→CDAG, D→EF}

(b) Given R{A, B, C, D, E, F, G, H} with FDs {A→BCDEFGH, BCD→AEFGH, BCE→ADEFGH, CE→H, CD→H}. Identify the best normal form that R satisfies (2NF/3NF/BCNF). [4]

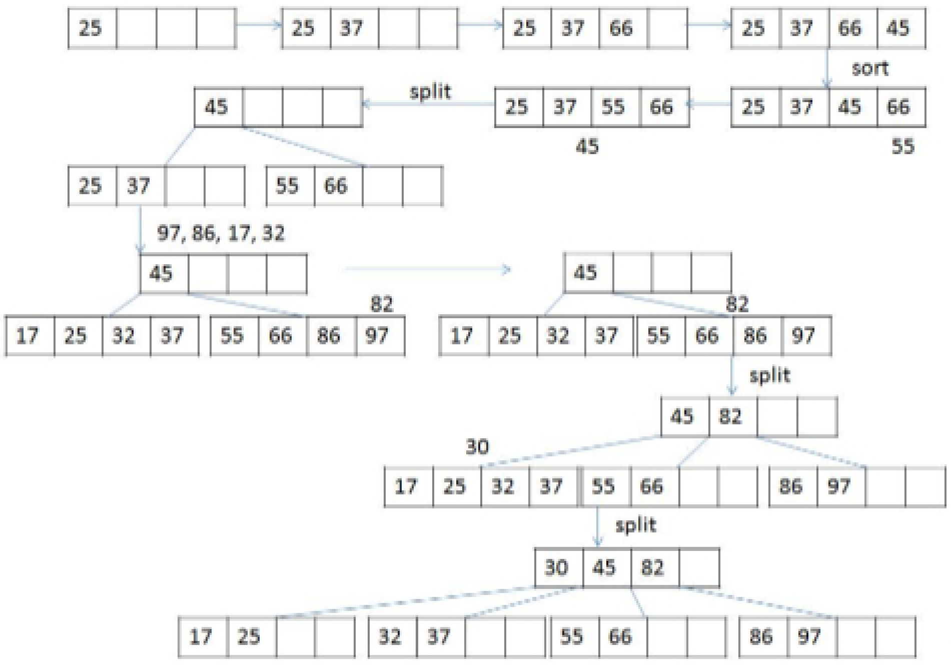
Ans:

Since the relational schema is provided, the relation is in 1NF.
A^+={A,B,C,D,E,F,G,H}
So, A is the Primary key.
Since the primary key contains only one attribute, there is no chance for partial FD. Thus, the relation is in 2NF.

Since the non-key attributes are depending on the key transitively, this relation is not present in 3NF.
The best normal form for this relation is 2NF.

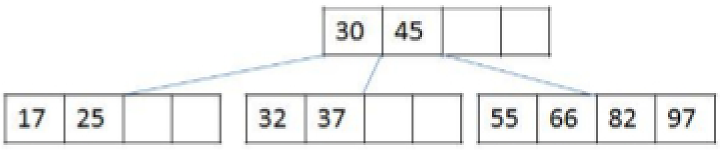
6.

(a) Construct a B-Tree of order 5 for the data items: 25, 37, 66, 45, 55, 97, 86, 17, 32, 82, 30. Redraw the tree after deleting 86 from the original B-Tree.
Ans:



[3]

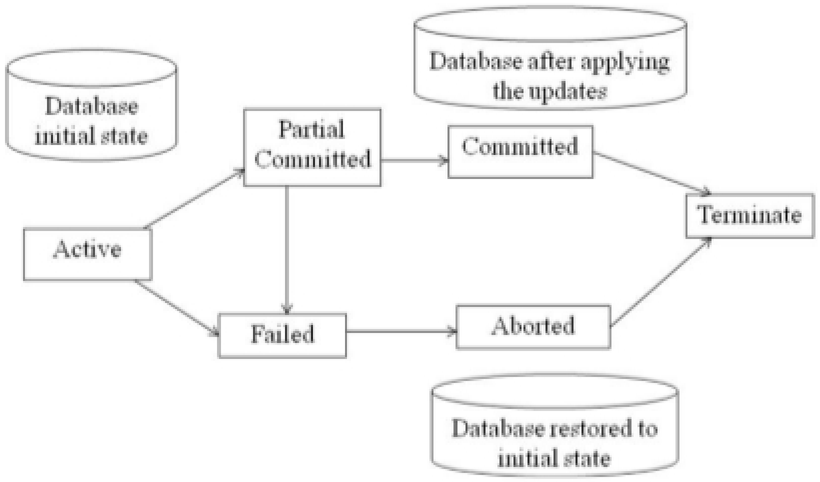
After removal of 86



[1]

(b) What are the different states of a transaction? Explain with the suitable example.
Ans:
Active state [0.5]
Partial Committed state [0.5]
Committed state [0.5]
Failed state [0.5]
Aborted state [0.5]
Terminate state [0.5]

Diagram [1]



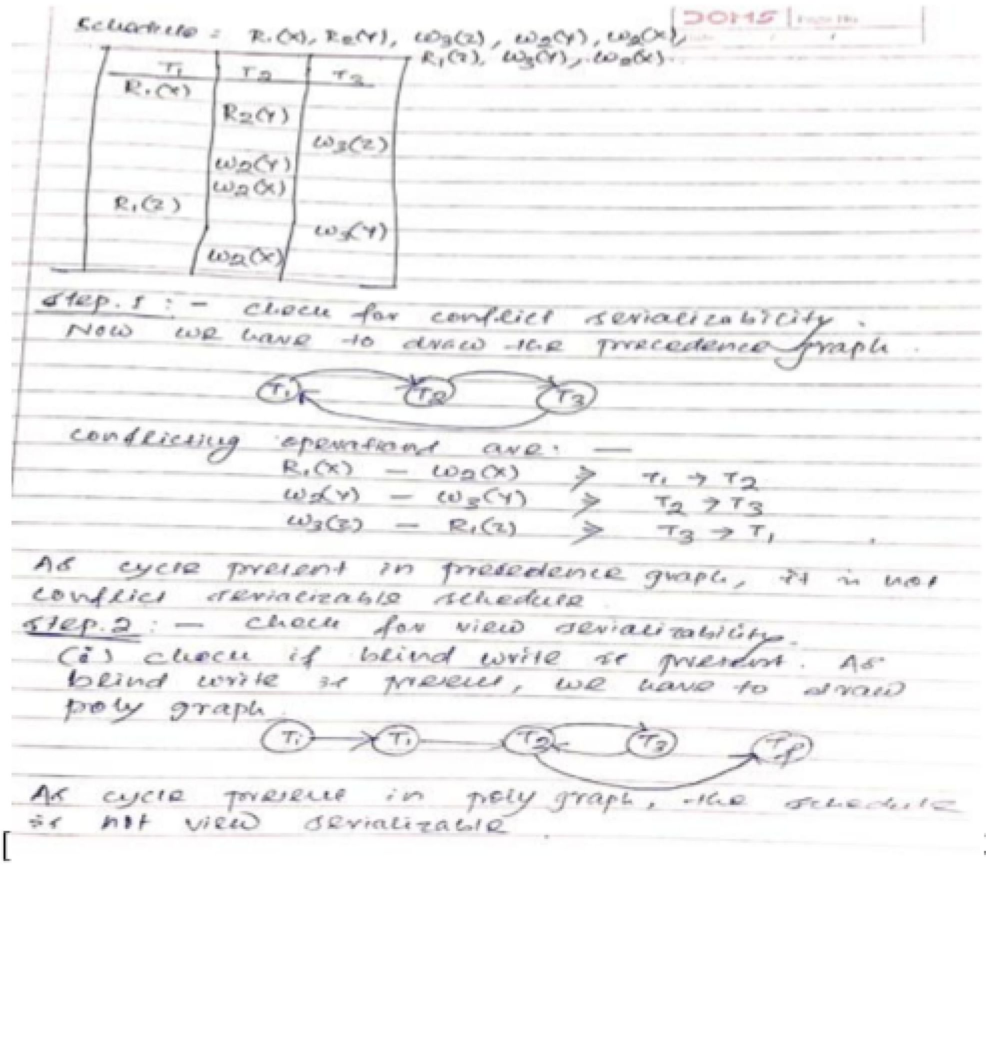
SECTION-D

7. (a) What is conflict and view serializability? Consider the following non serial schedule: R1(X), R2(Y), W3(Z), W2(Y), W2(X), R1(Z), W3(Y), W2(X) and check for conflict and view serializability. [4]

Ans:

Conflict Serializability: A concurrent schedule S is conflict serializable if it is conflict equivalent to one of its serial schedule. [0.5]

View Serializability: A concurrent schedule S is view serializable if it is view equivalent to one of its serial schedule. [0.5]



Schedule = $R_1(X), R_2(Y), W_3(Z), W_2(Y), W_2(X), R_1(Z), W_3(Y), W_2(X)$.

T_1	T_2	T_3
$R_1(X)$		
	$R_2(Y)$	
		$W_3(Z)$
	$W_2(Y)$	
	$W_2(X)$	
$R_1(Z)$		
		$W_3(Y)$
	$W_2(X)$	

Step. 1 :— check for conflict serializability. Now we have to draw the precedence graph.

$$\begin{tikzcd} T_1 \arrow[r, bend left] & T_2 \arrow[r, bend left] & T_3 \arrow[ll, bend left] \end{tikzcd}$$

conflicting operations are:—

$R_1(X) - W_2(X) \Rightarrow T_1 \rightarrow T_2$

$W_2(Y) - W_3(Y) \Rightarrow T_2 \rightarrow T_3$

$W_3(Z) - R_1(Z) \Rightarrow T_3 \rightarrow T_1$.

As cycle present in precedence graph, it is not conflict serializable schedule

Step. 2 :— check for view serializability.

(i) check if blind write is present. As blind write is present, we have to draw poly graph.

$$\begin{tikzcd} T_1 \arrow[r] & T_1 \arrow[r, dash] & T_2 \arrow[r, bend left] \arrow[rr, bend right] & T_3 \arrow[l] & T_f \end{tikzcd}$$

As cycle present in poly graph, the schedule is not view serializable

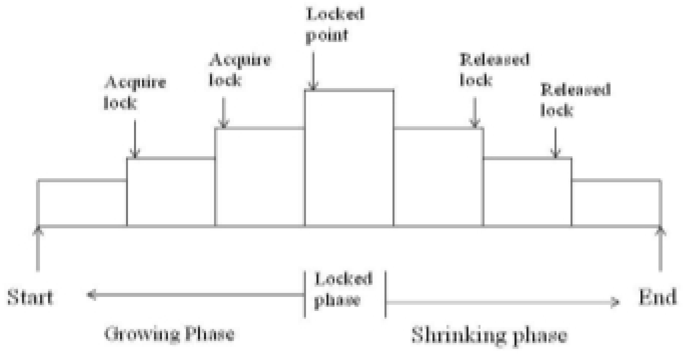
[3]

(b) Explain two-phase locking protocol with example. [4]

Ans:

Two phases [1]
Diagram [1]
Explanation [2]
Two-phase locking protocol is a protocol which ensures serializability.
This protocol requires that each transaction issues lock and unlock requests in two phases. The two phases are:

- ✓ **Growing phase**: Here, a transaction acquires all required locks without unlocking any data, i.e. the transaction may not release any lock
- ✓ **Shrinking phase:** Here, a transaction releases all locks and cannot obtain any new lock
  - ➢ The point in the schedule where the transaction has obtained its final lock is called the **lock point** of the transaction.
  - ➢ Transactions can be ordered according to their lock points.



  - ➢ Two transactions cannot have conflicting locks.
  - ➢ No unlock operation can precede a lock operation in the same transaction.
  - ➢ No data are affected until all locks are obtained, i.e. until the transaction is in its locked point.
  - ➢ Two-phase locking does not ensure freedom from deadlock. Along with the serializability property, the schedules should be cascade-less in nature.

8. Write short notes on the following

(a) File system Vs DBMS [4]
Ans:
8 Differences [0.5X8=4]

File System	DBMS
File System is a general, easy-to-use system to store general files which require less security and constraints.	Database management system is used when security constraints are high.
Data Redundancy is more in file management system.	Data Redundancy is less in database management system.
Data Inconsistency is more in file system.	Data Inconsistency is less in database management system.
Centralisation is hard to get when it comes to File Management System.	Centralisation is achieved in Database Management System.
User locates the physical address of the files to access data in File Management System.	In Database Management System, user is unaware of physical address where data is stored.
Security is low in File Management System.	Security is high in Database Management System.
File Management System stores unstructured data as isolated data files/entities.	Database Management System stores structured data which have well defined constraints and interrelation.
File system doesn't have a crash mechanism, i.e., if the system crashes while entering some data, then the content of the file will lost.	DBMS provides a crash recovery mechanism, i.e., DBMS protects the user from the system failure.
In the File system, concurrent access has many problems like redirecting the file while other deleting some information or updating some information.	DBMS takes care of Concurrent access of data using some form of locking

(b) 4th Normal Form & 5th Normal Form [4]

Ans:

4NF [2]

5NF [2]

Fourth Normal form (4NF)

- A relation will be in 4NF if it is in Boyce Codd normal form and has no multi-valued dependency.
- For a dependency A → B, if for a single value of A, multiple values of B exists, then the relation will be a multi-valued dependency.

Example

STUDENT

STU_ID	COURSE	HOBBY
21	Computer	Dancing
21	Math	Singing
34	Chemistry	Dancing
74	Biology	Cricket
59	Physics	Hockey

The given STUDENT table is in 3NF, but the COURSE and HOBBY are two independent entity. Hence, there is no relationship between COURSE and

HOBBY.

In the STUDENT relation, a student with STU_ID, **21** contains two courses, **Computer** and **Math** and two hobbies, **Dancing** and **Singing**. So there is a Multi-valued dependency on STU_ID, which leads to unnecessary repetition of data.

So to make the above table into 4NF, we can decompose it into two tables:

STUDENT_COURSE

STU_ID	COURSE
21	Computer
21	Math
34	Chemistry
74	Biology
59	Physics

STUDENT_HOBBY

STU_ID	HOBBY
21	Dancing
21	Singing
34	Dancing
74	Cricket
59	Hockey

Fifth normal form (5NF)

- A relation is in 5NF if it is in 4NF and not contains any join dependency and joining should be lossless.
- 5NF is satisfied when all the tables are broken into as many tables as possible in order to avoid redundancy.
- 5NF is also known as Project-join normal form (PJ/NF).

Example

SUBJECT	LECTURER	SEMESTER
Computer	Anshika	Semester 1
Computer	John	Semester 1
Math	John	Semester 1
Math	Akash	Semester 2
Chemistry	Praveen	Semester 1

In the above table, John takes both Computer and Math class for Semester 1 but he doesn't take Math class for Semester 2. In this case, combination of all these fields required to identify a valid data.
Suppose we add a new Semester as Semester 3 but do not know about the subject and who will be taking that subject so we leave Lecturer and Subject as NULL. But all three columns together acts as a primary key, so we can't leave other two columns blank.
So to make the above table into 5NF, we can decompose it into three relations P1, P2 & P3:

P1

SEMESTER	SUBJECT
Semester 1	Computer
Semester 1	Math
Semester 1	Chemistry
Semester 2	Math

P2

SUBJECT	LECTURER
Computer	Anshika
Computer	John
Math	John
Math	Akash
Chemistry	Praveen

P3

SEMSTER	LECTURER
Semester 1	Anshika
Semester 1	John
Semester 1	John
Semester 2	Akash
Semester 1	Praveen

*****