REG.NO.:

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
CONTINUOUS ASSESSMENT TEST - II
WINTER SEMESTER 2024-2025

SLOT: A1

Programme Name & Branch :
Course Code : CSI1001
Course Name : Principles of Database Systems
Faculty Name(s) : Saravanakumar K / Nagaraja Rao A / Gopichand G / Saraswathi Priyadharshini A / Mohankumar P
Class Number(s) : 1927 / 1930 / 1938 / 1941 / 1946
Date of Examination : 16/03/2025
Exam Duration : 90 minutes **Maximum Marks: 50**

General instruction(s):

- **Answer all the questions**

Q. No	Question	Marks	CO	BL
1.	Suppose that you are given a relation Patient_Appointment with the schema ***Patient_Appointment(PNo, PName, appNo, Time, Doctor)*** and set of functional dependencies F = {PNo, appNo → Time, Doctor; PNo → PName; Time → appNo}. Is Patient_Appointment in BCNF? If not, normalize the given relation to BCNF relation(s) with detailed procedure. Also, mention the set of functional dependencies hold by each of the resulting relations. **Answer:** Not in BCNF The **candidate key** for this relation is **(PNo, appNo)** We have two violations of BCNF: • **PNo → PName** (Violation 1) • **Time → appNo** (Violation 2) We will decompose the relation into smaller relations to eliminate these violations. **Decompose based on PNo → PName:** We will create two relations: 1. **R1(PNo, PName)** with the functional dependency **PNo → PName** 2. **R2(PNo, appNo, Time, Doctor)** with the functional dependencies **PNo, appNo → Time, Doctor** and **Time → appNo** R1 is in BCNF R2 is NOT. **Decompose R2 based on Time → appNo:** We decompose **R2** into two relations: 1. **R3(Time, appNo)** with the functional dependency **Time → appNo** **R4(PNo, appNo, Doctor)** with the functional dependency **PNo, appNo → Doctor** **Final relations:**	10	3	3

	After decomposition, we have the following relations that are all in BCNF: 1. **R1(PNo, PName)** with the functional dependency **PNo → PName** 2. **R3(Time, appNo)** with the functional dependency **Time → appNo** 3. **R4(PNo, appNo, Doctor)** with the functional dependency **PNo, appNo → Doctor**			
2.	Use the table EmployeeDetails and the sample data (Table 1) to answer the following questions; [2+2+3+3] a) Identify all the functional dependencies that are true for the table. (Consider the real-world scenarios while choosing a functional dependency) b) Find all the candidate keys for the table using attribute closure algorithm. c) Check whether the table is in 2NF or not. If not, decompose the table into 2NF relations. d) Verify whether the decomposition is Lossy or lossless. **Answer:** a) EID → EName, SSN, DOB, Gender, JTitle, Salary SSN → EID, EName, DOB, Gender, JTitle, Salary DeptID → Dept, MgrID b) Candidate keys of EmployeeDetails – {EID, DeptID} and {SSN, DeptID} c) Not in 2NF because of violation of few FDs. 2NF relations are R1(EID, EName, SSN, DOB, Gender, JTitle, Salary), R2(DeptID, Dept, MgrID), R3(EID, DeptID) d) The decomposition is lossless	10	3	3
3.	This question is based on the following relational schemas. ***Movie (title, year, length, release_date, budget, studioID)*** ***Studio (StudioID, Studioname, address, contact)*** Write queries in ***Relational Algebra*** to achieve the following; [2+2+3+3] a) Find the title and length of all movies with the budget of minimum 2 crore and maximum 10 crores. $\pi_{\text{title,length}}\left(\sigma_{\text{budget}\geq 2\text{ crores}\wedge\text{budget}\leq 10\text{ crores}}(\text{Movie})\right)$ b) Insert any one record into the relation Studio. Studio <-- Studio U {(101,'DreamWorks','123DreamSt.','1234567890')} c) Find the title, length and 10 percentage of budget of all the movies that are released by 'Universal' studio. $\pi_{\text{title,length},0.1\times\text{budget}}\left(\sigma_{\text{studioName}='Universal'}(\text{Studio}) \bowtie_{\text{StudioID}} \text{Movie}\right)$	10	5	3

	d) Find the title, budget, and address of all movies that are produced by the studio with the contact number 7788990011. [Use Cartesian product.] $\pi_{\text{title,budget,address}}(\sigma_{\text{Movie.studioID=Studio.studioID}}(\text{Movie} \times \sigma_{\text{contact=7788990011}}(\text{Studio})))$			
4.	Use the database schema given below to answer the following questions; [2+3+5] ***Emps (<u>empid</u>, name, gender, dept_ID)*** ***Depts (<u>deptid</u>, dept_name, mgrID)*** ***Mgrs (mgrID, mgr_name, age)*** a) Convert the SQL query "SELECT empid, name FROM emps INNER JOIN depts ON dept_ID = deptid INNER JOIN mgrs ON depts.mgrID = mgrs.mgrID WHERE gender = 'm' AND dept_name = 'chemistry' AND age>40;" into its equivalent relational algebra expression. b) Draw a query tree for the relational algebra expression that you wrote as an answer for 4(a). c) Construct an optimized query tree from the initial query tree (4(b) answer). Also, you need to discuss with the necessary information regarding the equivalence rules used and change in ordering of relational algebra operations. **<u>Answer:</u>** a) Equivalent RA could be the one given below or anything equialent; $\pi_{\text{empid,name}}(\sigma_{\text{gender='m'} \wedge \text{dept\_name='chemistry'} \wedge \text{age>40}}((\text{emps} \bowtie_{\text{dept\_ID=deptid}} \text{depts}) \bowtie_{\text{depts.mgrID=mgrs.mgrID}} \text{mgrs}))$ b) π(empid, name) | σ(gender = 'm' ∧ dept_name = 'chemistry' ∧ age > 40) | ⋈(depts.mgrID = mgrs.mgrID) / \\ ⋈(emps.dept_ID = depts.deptid) \\ | | \\ emps depts mgrs c) Draw a tree where the following heuristics were applied; The primary optimization strategies are: 1. **Push selections down**: Apply selections as early as possible to reduce the size of intermediate results. 2. **Reorder joins**: Perform joins in an order that minimizes the number of rows involved in each step. 3. **Use indexed attributes**: If possible, perform selections based on indexed attributes before joining to reduce the size of intermediate relations.	10	5	6

5.	Write short notes on the following classes of schedules; conflict serializable, view serializable, recoverable, and avoid cascading aborts. Also, determine which of these classes of schedules are applicable and not applicable for each of the following schedules. [5+5] Schedule S1:4. T1:R(X), T1:R(Y), T1:W(X), T2:R(Y), T3:W(Y), T1:W(X), T2:R(Y) Schedule S2:6. T1:R(X), T2:W(X), T1:W(X), T2:Commit, T1:Commit Schedule S3:7. T1:W(X), T2:R(X), T1:W(X), T2:Abort, T1:Commit Schedule S4:8. T1:W(X), T2:R(X), T1:W(X), T2:Commit, T1:Commit You are expected to fill your answers in the following table. If your answer is applicable fill YES otherwise NO.	10	5	5

Schedule	Conflict serializable	View serializable	Recoverable	Avoid cascading aborts

Answer:

Conflict serializable: A **schedule** is conflict serializable if it is possible to transform it into a **serial schedule** by swapping non-conflicting operations. **Conflict serializability** ensures that a schedule avoids conflicts that might lead to inconsistent results or violations of serializability.

View serializable: A **schedule** is view serializable if, for each transaction, the operations in the schedule result in the same "view" of the data as a serial schedule. **View serializability** is a more relaxed criterion than conflict serializability because it allows certain types of non-conflicting operations that still yield a serializable result.

Recoverable: A **schedule** is recoverable if, whenever a transaction **T1** reads a value written by another transaction **T2**, the commit of **T1** occurs after the commit of **T2**. This ensures that no transaction commits based on uncommitted data, which could lead to inconsistencies if **T2** fails and is rolled back. A recoverable schedule prevents **cascading aborts**, where rolling back one transaction would require rolling back several others due to interdependencies on uncommitted data.

Avoid cascading abort: A **cascading abort** occurs when a transaction has to be rolled back because of the failure of another transaction that it depends on. This can lead to a situation where multiple transactions need to be aborted due to one failure, creating a cascade of rollbacks.

To **avoid cascading aborts**, we use a **strict schedule** or enforce the rule that a transaction can only read data written by a transaction that has already committed. This way, if one transaction fails, others that rely on it do not need to be rolled back.

Schedule	Conflict serializable	View serializable	Recoverable	Avoid cascading aborts
S1	no	no	Cannot be decided	Cannot be decided
S2	no	yes	yes	Yes
S3	no	no	no	No
S4	no	no	no	no

S1: It is NOT conflict-serializable, NOT view-serializable;
It is NOT avoid cascading aborts;
We can not decide whether it's recoverable or not, since the abort/commit sequence of these transactions are not specified.

S2: It is view-serializable, not conflict-serializable;
It is recoverable and avoid cascading aborts;

S3: It is not view-serializable, not conflict-serializable;
It is not recoverable, therefore not avoid cascading aborts.

S4: It is not view-serializable, not conflict-serializable;
It is not recoverable, therefore not avoid cascading aborts

Table 1: EmployeeDetails

EID	EName	SSN	DOB	Gender	JTitle	Salary	DeptID	Dept	MgrID
1	John Doe	123-45-6789	1990-05-14	Male	Mgr	65000	3	HR	1
2	Emily Davis	444-55-6666	1992-03-30	Female	Clerk	42000	3	HR	1
3	Mark	111-22-3333	1980-02-12	Male	Mgr	75000	1	Finance	3
2	Emily Davis	444-55-6666	1992-03-30	Female	Clerk	42000	4	Marketing	12
5	Smith	987-61-4341	1985-08-22	Male	SWE	52000	2	Manufact	11
6	Mark	333-44-5555	1990-05-14	Female	Asst Mgr	62000	5	Manufact	13

REG.NO.:

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
CONTINUOUS ASSESSMENT TEST - II
WINTER SEMESTER 2024-2025

SLOT: A1

NOTE*: Please refer below to the BL – Bloom's Taxonomy Levels and report the respective level in the questions for moderation. After the moderation from the school office, kindly remove the COs and BLs in the final copies, which are supposed to be distributed to the students.

Bloom's Taxonomy Levels	Category
BL1	Remembering
BL2	Understanding
BL3	Applying
BL4	Analysing
BL5	Evaluating
BL6	Creating

****