|  |
| --- |
| **Database Systems (CS2005)** |
| Date: May 21st 2025 |
| **Course Instructor(s)** |
| IR, ZA, MN, AA, HI, MM, SA |

|  |  |
| --- | --- |
| **Final Exam** | |
| **Total Time (Hrs.):** | **3** |
| **Total Marks:** | **80** |
| **Total Questions:** | **8** |

|  |
| --- |
| **SOLUTION**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Roll No Section Student Signature |

**Do not write below this line.**

**Note:** **Please ensure that you attempt all questions and their respective parts in the given order.**

**Consider a detailed DB schema for the various CS competitions held in our university under the society SOFTEC every year.**

The participant table stores details about each participant who may take part in competitions. Participants can have mentors, but we only record mentors who are also participants or have been in the past. Teams participate in competitions. A team can have multiple participants and a Team lead (who must be a participant). The Competition table stores information about each competition organized by SOFTEC, including which team won.

|  |  |
| --- | --- |
| **CompetitionTeam** | |
| **CID** | **TeamID** |
| C01 | T01 |
| C01 | T02 |
| C01 | T03 |
| C02 | T01 |
| C02 | T02 |
| C02 | T03 |
| C03 | T05 |
| C03 | T04 |
| C04 | T02 |
| C04 | T03 |
| C05 | T04 |
| C05 | T05 |

|  |  |
| --- | --- |
| **TeamMember** | |
| **TeamID** | **PID** |
| T01 | P01 |
| T01 | P02 |
| T01 | P03 |
| T02 | P04 |
| T02 | P05 |
| T03 | P06 |
| T03 | P08 |
| T03 | P02 |
| T04 | P03 |
| T05 | P09 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Participant** | |  |  |
| **PID** | **Name** | **Gender** | **MentorID** |
| P01 | Areeba | F | NULL |
| P02 | Zain | M | P01 |
| P03 | Sara | F | P01 |
| P04 | Usman | M | P02 |
| P05 | Ali | M | NULL |
| P06 | Hamza | M | P04 |
| P07 | Sana | F | P03 |
| P08 | Bilal | M | P01 |
| P09 | Mehak | F | P10 |
| P10 | Noor | F | NULL |

|  |  |  |
| --- | --- | --- |
| **Team** |  |  |
| **TeamID** | **Name** | **TeamLeadID** |
| T01 | Alpha Coders | P01 |
| T02 | Binary Blasters | P04 |
| T03 | Code Ninjas | P06 |
| T04 | Data Wizards | P03 |
| T05 | Solo Master | P09 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Competition** | |  |  |
| **CID** | **Cname** | **Venue** | **Date** |
| C01 | Coding Challenge | SHall A | 4/27/2025 |
| C02 | Hackathon Sprint | CS Lab 3 | 4/27/2025 |
| C03 | Query Master | CS Lab 4 | 4/26/2025 |
| C04 | Data Analysis | Data Lab | 4/26/2025 |
| C05 | Debug Solo | Online | 4/24/2025 |

***CLO # 5: Author queries using relational algebra and SQL.***

**Q. No 1:** Consider the database state given above. Give the output for each of the following queries. Also show the intermediate tables. Explain in one sentence what these queries are doing. [9]

1. R1 ← σGender='F' (Participant)

Result ← πTeamID (σR1.PID = ϕ (TeamMember ⟕ TeamMember.PID = R1.PID R1)

**Ans: Teams with male members T1, T2, T3...; a team with all female members is not included.**

1. SELECT DISTINCT ct.CID FROM CompetitionTeam ct

WHERE EXISTS (SELECT \* FROM CompetitionTeam ct1 JOIN TeamMember tm1 ON ct1.TeamID = tm1.TeamID

WHERE ct1.CID = ct.CID

GROUP BY ct1.TeamID

HAVING COUNT(tm1.PID) <> ALL (

SELECT COUNT(tm2.PID)

FROM CompetitionTeam ct2 JOIN TeamMember tm2 ON ct2.TeamID = tm2.TeamID

WHERE ct2.CID = ct.CID

GROUP BY ct2.TeamID

)

);

**Ans: *The competitions that allow teams of different numbers of members*, *null***

1. SELECT DISTINCT p1.PID, p1.Name

FROM Participant p1 JOIN Participant p2 ON p1.PID = p2.MentorID

LEFT JOIN TeamMember tm ON p1.PID = tm.PID

LEFT JOIN CompetitionTeam ct ON tm.TeamID = ct.TeamID

WHERE ct.TeamID IS NULL;

**Ans: Find participants who are mentors but did not participate in any competition: P10 Noor**

***CLO # 5: Author queries using relational algebra and SQL.***

**Q. No 2:** Consider the above database. Write **SQL and RA** statements for each of the following problems. [16]

1. List the name of the participant who participated in all the competitions held in 2025.

**Ans:**

**SELECT p.Name**

**FROM Participant p**

**JOIN TeamMember tm ON p.PID = tm.PID**

**JOIN CompetitionTeam ct ON tm.TeamID = ct.TeamID**

**JOIN Competition c ON ct.CID = c.CID**

**WHERE YEAR(c.Date) = 2025**

**GROUP BY p.PID, p.Name**

**HAVING COUNT(DISTINCT ct.CID) = (**

**SELECT COUNT(\*)**

**FROM Competition**

**WHERE YEAR(Date) = 2025**

**);**

1. For each team with more than three members, list the name of the Team along with the name of the Team Lead.

**Ans:** **SELECT t.Name AS TeamName, p.Name AS TeamLeadName**

**FROM Team t**

**JOIN Participant p ON t.TeamLeadID = p.PID**

**JOIN (**

**SELECT TeamID**

**FROM TeamMember**

**GROUP BY TeamID**

**HAVING COUNT(PID) > 3**

**) tm\_count ON t.TeamID = tm\_count.TeamID;**

***CLO # 5: Author queries using relational algebra and SQL.***

**Q. No 3:** Consider the above database. Create a view that finds and lists the ***pairs of Teams*** that have participated in the same set of competitions. [5]

Hint: For the above data, the answer to your query should be

|  |  |
| --- | --- |
| **Team1** | **Team2** |
| T04 | T05 |
| T02 | T03 |

**Ans:**

**CREATE VIEW SameCompetitionTeams AS**

**SELECT DISTINCT ct1.TeamID AS Team1, ct2.TeamID AS Team2**

**FROM CompetitionTeam ct1 JOIN CompetitionTeam ct2 ON ct1.CID = ct2.CID**

**WHERE ct1.TeamID < ct2.TeamID**

**GROUP BY ct1.TeamID, ct2.TeamID**

**HAVING COUNT(\*) = (SELECT COUNT(\*) FROM CompetitionTeam c1 WHERE c1.TeamID = ct1.TeamID)**

**AND COUNT(\*) = (SELECT COUNT(\*) FROM CompetitionTeam c2 WHERE c2.TeamID = ct2.TeamID);**

***CLO # 3: Develop a normalized relational design to remove anomalies in a set of relations.***

**Q. No 4:** Consider the relation schema R (A, B, C, D, E), with FDs

F= {A → BC, AB → D, C → D, A → E, EC → B}. Find a minimal cover of *F* (i.e. Fc). [5]

**Ans: Fc= {A → CE, C → D, CE → B}**

***CLO # 3: Develop a normalized relational design to remove anomalies in a set of relations.***

**Q. No 5:** Identify the best normal form of the following relations. Justify your answer. If the given relation is not in BCNF, decompose it into a set of BCNF relations. Please note all parts are independent. [15]

1. Consider the relation R(A, B, C, D), with FDs F= {A → BC, B → D, CD → A}.
2. Consider the relation R(A, B, C, D, E), with FDs F= {AB → C, C → D, D → E, A → B}.
3. Consider the relation R(A, B, C, D), with FDs F= {AB → C, C → D, D → B}.

**Ans:**

**a. Keys are A, BC, and CD. HNF is 3NF as FD2: B→D violates BCNF.**

**BCNF Schema: R1(A, B, C), R2(B, D); FD3: CD→A is lost.**

**b. Key is A. HNF is 2NF as FD2: C→D & FD3: D→E violate 3NF.**

**3NF Schema: R1(A, B, C), R2(C, D, E); FD3: D→E violate BCNF.**

**BCNF Schema: R1(A, B, C), R21(C, D), R22(D, E); All FDs are preserved.**

**c. Keys are AB, AC, and AD. HNF is 3NF as FD2: C→D & FD3: D→B violate BCNF.**

**BCNF Schema: R1(A, C), R2(C, D), R3(D, B); FD1: AB→C is lost.**

***CLO # 3: Develop a normalized relational design to remove anomalies in a set of relations.***

**Q. No 6:** Consider the following schedule: [5]

**S:** r1(A); w1(A); r2(A); w3(A); w4(D); r4(D); r3(B); w5(B); w4(C); r5(C).

Draw the serializability (precedence) graph for this schedule. State whether this schedule is (conflict) serializable or not. If the schedule is serializable, write down the equivalent serial schedule(s) otherwise explain why it is not.

**Ans: It is conflict-serializable and view-serializable. Equivalent serial schedules are**

**T1🡪T2🡪T3🡪T4🡪T5, T4🡪T1🡪T2🡪T3🡪T5, & more …**

A

A

B

A

C

***CLO # 2: Design a conceptual model using ER Model for an enterprise.***

**Q. No 7:** Map the following ER/EER Diagram into a relational model and specify all the constraints including primary key, foreign key, not null, and unique. [10]

1

1ç√√

1

1

Request

User

Passenger

Driver

Trip

TripCancellation

Receives

Add

Accepts

Initiates

ResultsIn

Car

has

Mç

1

1

1

Mç

1

1

N

1

***CLO # 2: Design a conceptual model using ER Model for an enterprise.***

**Q. No 8: Draw an ER/EER diagram** (using notation discussed in lectures) for the following requirements of a Deep Space Mission Control System to manage interstellar expeditions, spacecraft, astronauts, and planetary research. Specify all constraints that should hold on to the database and state any assumptions you make. [15]

The system must track **spacecraft** (uniquely identified by Ship\_ID) with attributes including model, manufacturer, fuel capacity, backup crew member and engine specifications (thrust level, propellant type, and last maintenance date). Astronauts, identified by their unique Astronaut\_ID, operate and control these spacecrafts. Each astronaut's record must include their rank, training certifications, and specialized role. The system categorizes astronauts into two distinct subtypes: Pilot\_Astronauts who have tracking flight hours and license number and other is Science\_Astronauts who is documenting research fields and publications. An astronaut cannot be both a pilot and scientist. Astronauts can belong to other types as well.

The system regularly conducts **Mission\_Experiments** like soil analysis or atmospheric tests, which depend on their parent **Mission** identified by Mission\_ID. Each experiment has a partial key Exp\_ID and attributes like objective and status. A mission must involve at least one spacecraft and multiple astronauts, while astronauts can participate in multiple missions with different roles. Additionally, spacecraft visits **celestial bodies** (planets, moons), where each celestial body has attributes including *body\_ID*, name, mass. These visit records are maintained through an Expedition Log, whenever spacecrafts visit any celestial body/bodies, where celestial bodies may have zero or more record. To address security threats, the system also incorporates **missile defense protocols**. A spacecraft generates multiple defense logs during attacks, and Every defense log requires a spacecraft. Defense\_Log is identified by Log\_ID which depends upon spacecraft.

A paper with red marker on it

AI-generated content may be incorrect.