National University of Computer and Emerging Sciences

Lahore Campus

# Database Systems (CS2005) Final Exam

Date: Thu, 23 May 2024

**Course Instructor(s)**

ZA, MN, SF, AA, HI, MM, IR

| | |
|---|---|
| **Total Time (Hrs.):** | **3** |
| **Total Marks:** | **70** |
| **Total Questions:** | **9** |

_____________ ____________ __________________________

Roll No Section Student Signature

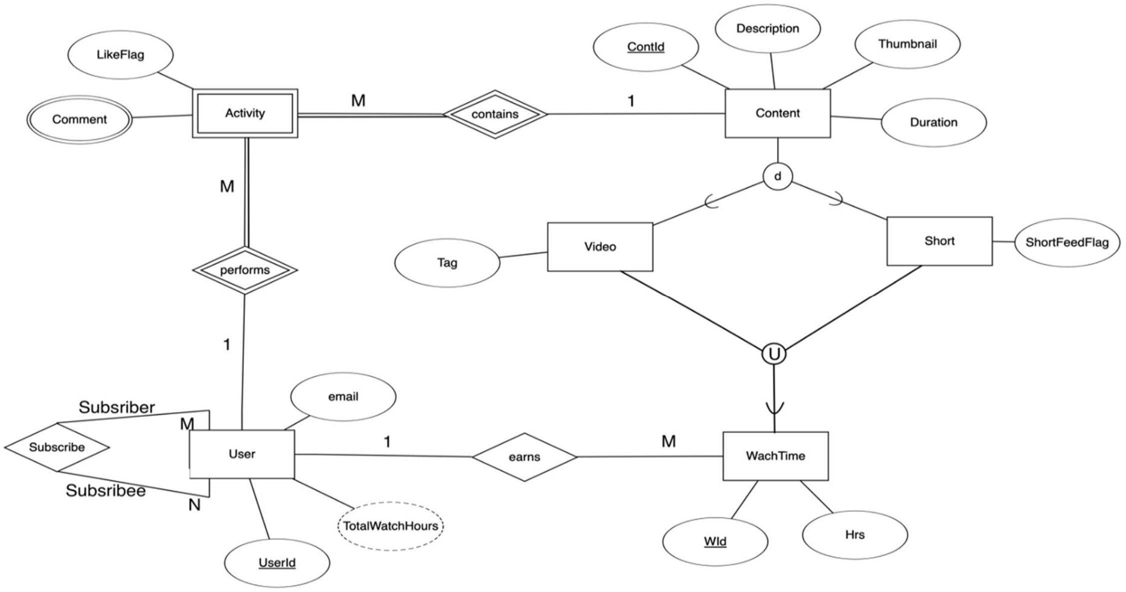
**Do not write below this line.**

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

**SOLUTION**

***CLO # 2: Design conceptual, logical and physical database schemas using different data models.***

**Q. No 1:** 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]



***CLO # 2: Design conceptual, logical and physical database schemas using different data models.***

**Q. No 2: Draw an ER/EER diagram** (using notation discussed in lectures) for the following requirements of a simple database of National Hockey League (NHL). Specify all constraints that should hold on to the database and state any assumptions you make. [10]

The NHL has many teams, each team has a name, a city, a coach, a captain, and a set of players, each player belongs to only one team, each player has a name (first, last, middle), a position (such as left wing or goalie), and a skill level. There are two types of players, beginners, and experts. For beginners the number of matches per day for training is more than the experts one. The trainers are just required for beginners and 90% attendance is required in training sessions to remain in the team. Team captain is also a player, a game is played between two teams (referred to as host_team and guest_team) and has a date (such as May 11th, 1999) and a score (such as 4 to 2). A player can also get an injury during a game. A description or record of injury is also maintained for each player.

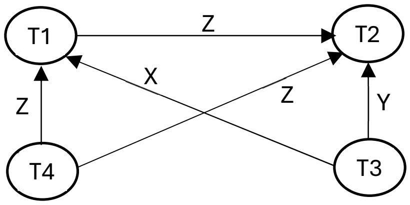
***CLO # 3: Identify FDs and resolve database anomalies by normalizing database tables.***

**Q. No 3:** Consider the following schedule of four transactions T1, T2, T3, and T4.

**S:** r1(X), r4 (Z), w1(Z), r3 (X), r3 (Y), w1 (X), w3 (Y), r2 (Y), w2 (Z), w2 (Y).

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. [5]

**Ans: It is conflict-serializable and equivalent serial schedules are T4→T3→T1→T2 and T3→T4→T1→T2.**
**It is also view-serializable.**



***CLO # 3: Identify FDs and resolve database anomalies by normalizing database tables.***

**Q. No 4:** Consider a relation R (A, B, C, D, E), with the set of FDs F= {C → D, B → ACE, AD → B}. Find all possible keys (i.e. candidate keys) of this relation? Prove it. [5]

**Ans: Keys are Ans: B, AC, and AD.**

***CLO # 3: Identify FDs and resolve database anomalies by normalizing database tables.***

**Q. No 5:** Consider the relation schema R (A, B, C, D, E), with FDs F= {ABC → D, AD→ B, E → C, A→ B, D→ A, BC → A}. Find a minimal cover of *F* (i.e. $F_c$). [5]

**Ans: $F_c$= {~~A~~BC → D, ~~AD→ B,~~ E → C, A→ B, D→ A, ~~BC → A~~}**
**i.e. $F_c$= {BC → D, E → C, A→ B, D→ A}.**

***CLO # 3: Identify FDs and resolve database anomalies by normalizing database tables.***

**Q. No 6:** Consider the relation R (A, B. C, D, E), with FDs {A→ B, BC → D, E → C, D→ A}. State which of the following decompositions of R relation are lossless decomposition. Prove it. [5]

**a.** R1(C, E), R2(A, B), and R3(A, C, D).
**b.** R1(C, E), R2(A, B), and R3(A, D, E).

**Ans: Keys of R are AE, BE, and DE.**
**a. Lossy decomposition.** R1(C, E), R2(A, B), and R3(A, C, D); Only one cond. is true i.e. R2∩R3→R2.
**b. Lossless decomposition.** R1(C, E), R2(A, B), and R3(A, D, E); R1∩R3→R1 & R2∩R3→R2.

***CLO # 3: Identify FDs and resolve database anomalies by normalizing database tables.***

**Q. No 7:** Consider the relation schema R (A, B, C, D, E, G), with FDs F= {AB → C, CD → EG, EG → A, BC → D}. Keys of this relation are AB, BC, and BEG. Identify the best normal form that R satisfies (1NF, 2NF, 3NF, or BCNF). Justify your answer. If R is not in BCNF, decompose it into a set of BCNF relations and show your steps. Indicate which dependencies if any are not preserved by the BCNF decomposition. [5]

**Ans: HNF=3NF as FD2/FD3 violates BCNF.**
**BCNF Schema is R1(B, C, D), R2(C, D, E, G), R3(E, G, A). FD1: AB→C is lost.**

**Consider the following part of Library database for the next questions:**
The NUCES library keeps records of the different reports submitted by students to the department, such as FYP Reports, MS thesis, and PhD thesis. Recently, the library added a new module, ***Report Rating***, that allows library users to rate and review the reports in the library database. This feature is beneficial for students looking for quality reports on a subject.

The Rating module includes Users, Reports, and Ratings. Users are identified by their unique ID, and have username, and gender. Each Report has a title, unique identifier (ReportID), year (publication year), and Subject (which can be a Database, Operating system, Algorithms, etc).
The authors of a Report are stored in the Report_Author table, as a report (such as an FYP report) can have multiple authors. **AuthorID is a foreign key and refers to UserID in the User table.** The Rating Table links Users and Reports and stores the user ratings and reviews on different Reports. The user gives a rating on a scale of 1-5.

**USER**

| *UserID* | *UserName* | Gender |
|---|---|---|
| **1** | Ali | male |
| **2** | Ahmed | male |
| **3** | Hamza | male |
| **4** | kiran | female |
| **5** | Maria | female |

**REPORT**

| *ReportID* | *Title* | Subject | Year |
|---|---|---|---|
| 1 | Distributed DB for big data | Database | 2010 |
| 2 | Fraud detection using AI | Machine Learning | 2023 |
| 3 | Advanced leftist trees | Data Structure | 2001 |
| 4 | Applications of Neural Networks | Machine Learning | 1994 |
| 5 | AVL Trees | Data Structure | 2000 |

**REPORT_AUTHOR**

| ReportID | AuthorID |
|---|---|
| 1 | **1** |
| 1 | **2** |
| 2 | **3** |
| 3 | **4** |
| 4 | **5** |
| 5 | **1** |

**RATING**

| ReportID | *UserID* | UserRating | UserReview |
|---|---|---|---|
| 1 | **1** | 4 | Excellent |
| 1 | **2** | 5 | Highly recommended |
| 3 | **2** | 3 | Good read |
| 4 | **4** | 4 | Classic literature |
| 5 | **5** | 2 | Not my cup of tea |

***CLO # 4: Use SQL for database definition and manipulation in any DBMS.***

**Q. No 8:** Consider the database state given above and for each of the following queries, give the output for the database state given above. [10]

**a.**
```
SELECT u.username   FROM User AS u
WHERE u.UserID IN (SELECT a.AuthorID  FROM Report_Author AS a
                          WHERE a.AuthorID IN (SELECT b.UserID   FROM Rating AS b
                                                 WHERE a.ReportID = b.ReportID));
```

Answer:

| Username |
|---|
| Ali |
| Ahmed |

**b.**
```
SELECT b.ReportID, b.Title , COUNT(a.AuthorID) AS C
FROM Report b LEFT OUTER JOIN Report_Author a ON b.ReportID = a.ReportID
GROUP BY b.ReportID, b.Title;
```

Answer:

| | | |
|---|---|---|
| 1 | Distributed DB for big data | 2 |
| 2 | Fraud detection using AI | 1 |
| 3 | Advanced leftist trees | 1 |
| 4 | Applications of Neural Networks | 1 |
| 5 | AVL Trees | 1 |

**c.** $\pi_{\text{UserID, UserName}}(\sigma_{\text{UserRating}<3}(\text{RATING} \bowtie_{\textit{Rating.userID=user.userID}} \text{USER}))$

| userID | UserName |
|---|---|
| 5 | Maria |

**d.** $\text{Subject}\ \Im_{\text{COUNT(Subject)}}\ (\text{RATING} * \text{REPORT})$

| Subject | C |
|---|---|
| Database | 2 |
| Data Structure | 2 |
| Machiene Learning | 1 |

***CLO # 4: Use SQL for database definition and manipulation in any DBMS.***

**Q. No 9:** Consider the above database for the following problems. [15]

**a.** Write **SQL and RA** statements to list the names and IDs of the Authors who have **only** written a Report on the Subject "Database".

```
SELECT A.AuthorID, U.name
FROM ReportAuthor A JOIN Report R ON A.ReportID = R.ReportID JOIN User U ON A.AuthorID = U.userID
WHERE R.Subject = 'Database' AND A.AuthorID NOT IN (
                          SELECT A1.AuthorID
                          FROM ReportAuthor A1 JOIN Report R1 ON A1.ReportID = R1.ReportID
                          WHERE R1.Subject <> 'Database' );
```

- $\text{NonDBAuthors} = \pi_{\{\text{AuthorID}\}} (\text{ReportAuthor} * \{\sigma_{\{\text{Subject<>'Database'}\}} (\text{Report})\})$
- $\text{DBAuthors} = \pi_{\{\text{AuthorID}\}} (\text{ReportAuthor} * \sigma_{\text{Subject='Database'}}(\text{Report}))$
- $\text{OnlyDBAuthors} = \text{DBAuthors} - \text{NonDBAuthors}$
- $\text{AuthorNames} = \pi_{\{\text{AuthorID, name}\}} (\text{User} \bowtie \text{OnlyDBAuthors})$

**b.** Write **SQL and RA** statements to list the names of the male users who have reviewed ***all*** the Reports rated 3 or above by the user "Damam Shah" with UserID=101.

```
SELECT   UserNAME
FROM     USER U
WHERE    Gender = 'Male' and
NOT EXISTS (
           (SELECT  ReportID   FROM Rating r
           WHERE    rating>=3 and r.UserID=101)
                  EXCEPT
      (SELECT      ReportID
      FROM         Rating R
      WHERE        U.UserID= R.USerID)      )
```

- $U1 \leftarrow \pi_{\text{ReportID}} (\sigma_{\text{rating>=3 and r.UserID=101}} \text{Rating})$
- $U2 \leftarrow \pi_{\text{UserID, ReportID}} (\text{Rating})$
- $U3 \leftarrow \mathbf{(U2 \div U1)}$
- **Result** $\leftarrow \pi_{\text{UserName}} \{U3 * \sigma_{\text{gender='Male'}} \text{ User}\}$

**c.** Create a ***View*** that lists the ReportID, Title, MaximumRating, and the number of reviews received for the Report, **but** only for the Reports with a minimum rating above 3 and written by 2 or more authors.

```
CREATE VIEW HighRatedReports AS
SELECT R.ReportID, R.Title, MAX(Rating.UserRating) AS MaximumRating,
COUNT(Rating.ReportID) AS NumReviews
FROM Report R JOIN ReportAuthor RA ON R.ReportID = RA.ReportID JOIN Rating ON
R.ReportID = Rating.ReportID
GROUP BY R.ReportID, R.Title
HAVING MIN(Rating.UserRating) > 3 AND COUNT(DISTINCT RA.AuthorID) >= 2;
```