CS203 Database System

Wednesday, June 24, 2020

Course Instructor

Dr. Shahzad Sarfaraz, Mr. Hafiz Tayyeb Javed, Miss. Faryal Saud, Mr. Mazhar Hussain, and Mr. Suffiyan Nizami

Serial No:

Final Term Exam
Spring Semester 2020
Max Time: 3 Hour
(Attempt Time), 20 Minutes
(Submission Time)
Max Marks: 100
Exam Weight (Out of 100). 45

Roll No _____ Section _____

Guidelines for Submission:

1. You have to solve this Exam on 'A4' papers.
2. Each question is to be solved on separate papers by clearly mentioning question number and Roll No at the top of the page.
3. After completing a hand written exam, you have to take clear images of your papers and paste them in order in a single WORD document in ascending order.
4. Save the word document in PDF format and submit this single Document.
5. You should submit only one PDF document named 'RollNo_Name'.
6. You must submit your solution before due time via Slate only.
7. You must submit your solution within the due time to get a good letter or S/U grade.
8. You have **3 hours and 20 Minutes** to submit the Exam
    a. 3 hours --- working time
    b. 20 Minutes --- for preparing and submitting the document on Slate.
9. If you don't finish every part of a question, don't worry! You can still submit what you've done to get marks based on your efforts.
10. In case of copied or plagiarized solutions in exam Or If you provide help to another student during exam both will be awarded "F" grade and it will affect your CGPA.
11. Viva of this exam may be conducted in a day or two.
12. This document should be submitted through LMS (Slate). But in the worst case only, you can email it within the deadline.

Question	1	2	3	4	5	6	7	8	Total
Total Marks	15	20	10	10	10	20	05	10	100
Obtained Marks									

Database (Carefully Observation required for Question 1 and 2)

1. Events

Event_Name	Event Type	Event Coordinator
Basket Ball	Sports	Mr. Adeel
Futsal	Sports	Mr. Adeel
Rob maze	EE	Mr. Adeel
Badminton	Sports	Mr. Adeel
Speed Programming	CS	Mr. Adeel
Taken Tag	Gaming	Mr. Adeel
BUISNESS IDEAS	Management	Mr. Adeel

2. Timeline

Time	Day	Location
08:00-09:45	Day 1	Front ground, 101
05:00-06:30	Day 2	Cricket Ground
04:00-05:53	Day 3	Futsall Ground, 102
03:45-09:00	Day 1	Main Lobby, 101

3. Society

Society_id	Society Name	Organizer_id
So-001	FAS	O-001
So-004	FEGS	O-005
So-002	Rang	O-003
So-007	Arts	O-003
So-003	Business	O-006
So-009	FCAP	O-004

4. Event Organizer

Organizer_id	Organizer Name	Organizer-Office-Address	Organizer_Dept
O-001	Mr. Zeeshan	CS-11	CS
O-004	Mr. Amir	F-208	MG
O-003	Mr. Usman	CS-09	CS
O-006	Mr. Habib	CS-08	CS
O-005	Mr. Adeel	CS-14	CS

5. Media/ Sponsorships

Sponsor-Id	Sponser-Name	sponsored-amount
S0019	HBL	4,000
FD00120000	City FM 93	5,999
XMQ213212	Oklays Bristo	10,000
XMN009	Roof Top Rest	98,000
S000563	Fork n knife	1,245
S00125634	Bahria Town	19, 000
S001	Gawadar Golf City	2,1053

7. Participants

P-Name	P-Affiliation Type 'Not Null'	Event_Name
Ali	UCP	Basket Ball
Adeel	GC-FSD	Futsal
Faheem	FAST-CFD	Rob maze
Azeem	FAST-ISB	Badminton
Asif	LUMS	Speed Programming
Zain	NUST	Taken Tag
Zainab	Arid University	BUISNESS IDEAS

Question No. 1 Answer the following. **(Marks: 15)**

Triggers

i) In the given database, consider one more attribute 'sponsored-amount' in the table Media/Sponsorships. Write PL/SQL statements to create a **trigger** which will validate the value of sponsored-amount that is in the range of 5000 to 90,000, if not then trigger will generate an exception informing to user either its value is low or higher than the range. Also use SQL insert and update statements to test it. **(8 Marks)**

Solution:

```
create or replace trigger amount_trigger
 before update of sponsored-amount or insert on Media/sponsorships
for each row
when((new. sponsored-amount < 5000) or (new. sponsored-amount > 90000))
begin
if :new. sponsored-amount < 5000 then
     RAISE_APPLICATION_ERROR(-2005, 'The amount cannot be less than 5000.');
end if;
 if :new. sponsored-amount > 90000 then
     RAISE_APPLICATION_ERROR(-2006, 'The amount cannot exceed 90000..');
end if;
end;
/

UPDATE Media/Sponsorships
 SET  sponsored-amount = 4000
WHERE Sponsor-Id = 'S0019';
```

View

ii) Write an SQL query to create a **view** that selects every P-Affiliation in the "Participants" table with a participation count higher than the average participation count. **(7 Marks)**

Solution:

```
CREATE VIEW [Max-Participation] AS
SELECT P-Affiliation, count
FROM Participants
WHERE count > (SELECT AVG(count) FROM Participants);
```

Question # 02. **(Marks: 20)**

Normalization Anomalies & Functional Dependencies

I. Identify the Update Anomaly in the above database, discuss how it can be resolved. **(2 Marks)**

Update anomaly may occur in table Society because updating society name automatically needs to update the organizer id of respective society

II. What will happen on anomalies (Insert, Delete, Update **(Anyone)**), if we change the type of attribute 'P-Affiliation' may Null of table '**Participants**' **(2 Marks)**

The attribute will be null able entries however, may not effect on any anomaly

III. Identify the Delete Anomaly in the above database, discuss how it can be resolved. **(2 Marks)**

Deleting the last record in first table may delete the information of event coordinator.

IV. Identify Insert Anomaly in the above database **(2 Marks)**
Adding organizer id in event organizer need to add its address and department.

V. In case of deleting record of **'O-003'** in **'Event Organizer'** table, what other event will be happened in the above database. **(1 Marks)**
The information of society name 'businesses' will lost in the data table.

VI. Identify which table(s) in the above database is not is First Normal Form. (1NF). Convert above database in (1NF) if identified **(2 Marks)**
Timeline table is not in 1st normal form, it contains multiple values in its third attribute with no atomic values for conversion split tables into two

VII. Identify which table(s) in the above database is not is Second Normal Form. (2NF). Convert above database in (2NF) if identified **(3 Marks)**
Event organizer contains functional dependency. Splitting in two different tables can resolve functional dependency.

VIII. Identify which table(s) in the above database is not is Third Normal Form. (3NF). And define by giving example if the above database is currently in (3NF) **(3 Marks)**
Event organizer table remove transitive dependency, removing such dependency in the table by splitting into two table may resolve it.

IX. Identify those tables which are not linked in the above database. Suggest attributes for Linking of these tables. **(3 Marks)**
Media and sponsorship and Timeline tables are not linked with each other. May establish the link by deciding about key attributes.

Question # 03

The DB-Pizza Store

Consider the following relational design used in our friendly local DB-Pizza store in Blacksburg:

Customer (cid, name, phonenumber, ccn, neighborhood, age)
Pizza (pid, pname, size, price)
Order (cid, pid, ordertime, orderyear, ordermonth, orderday, quantity, slices)
Supplies (sid, sname, amountleft, unitprice)
Ingredient (pid, sid, amount)

In the **Pizza** table, every pizza has a name (e.g., "the works") and particular size (e.g., 7 inches) and price and is assigned a unique pid. Note that different pizzas may have the same name, but different sizes.

The **Order** table includes the records about which customer ordered which pizza, quantity of pizzas, slices, orderyear (e.g., 2012), ordermonth (e.g., 12), orderday (e.g., 27), and the order time (e.g., "6:13pm"). Note that every customer can order one or more pizzas.

The **Customer** table maintains the personal information, such as a unique id, name, phone number, credit card number (ccn), neighborhood (e.g., Foxridge and Terrace View), and age. It is possible that two different people have the same name.

The **Supplies** table includes the information of the various groceries used by the store: the name, unit-price, and the amount-left in the store (e.g. the store might have 3kgs of mozzarella left with a unit price of 5$ per kg).

The **Ingredient** table keeps the records about the amount of ingredients used by each pizza (so "The works 7inch" might use only 10gms of mozzarella, while "Four cheese 12inch" might use 40gms).

Answer the following **(Marks: 2+2+3+3=10)**

i. **Specify the candidate keys for "Customer" and "Order table"**
Customer: id, phone number(if unique and not null), ccn
Order: (cid,pid) (ordertime, orderyear, ordermonth, orderday)

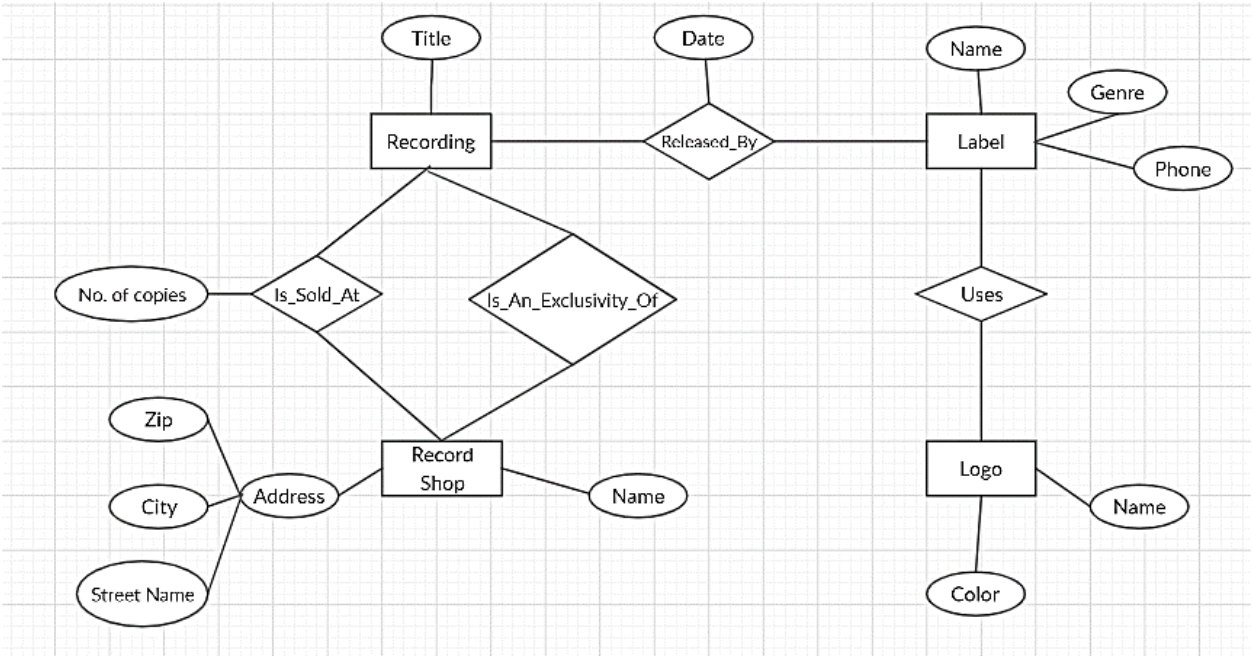
ii. **What are good primary keys for the five tables?**
Customer: cid
Pizza: pid,
Order: (cid, pid)
Supplies: sid
Ingredient: (pid, sid)

iii. **Specify the foreign keys for this schema with references.**
Order: cid references customer, pid references Pizza
Ingredient: pid references Pizza, sid references Supplies

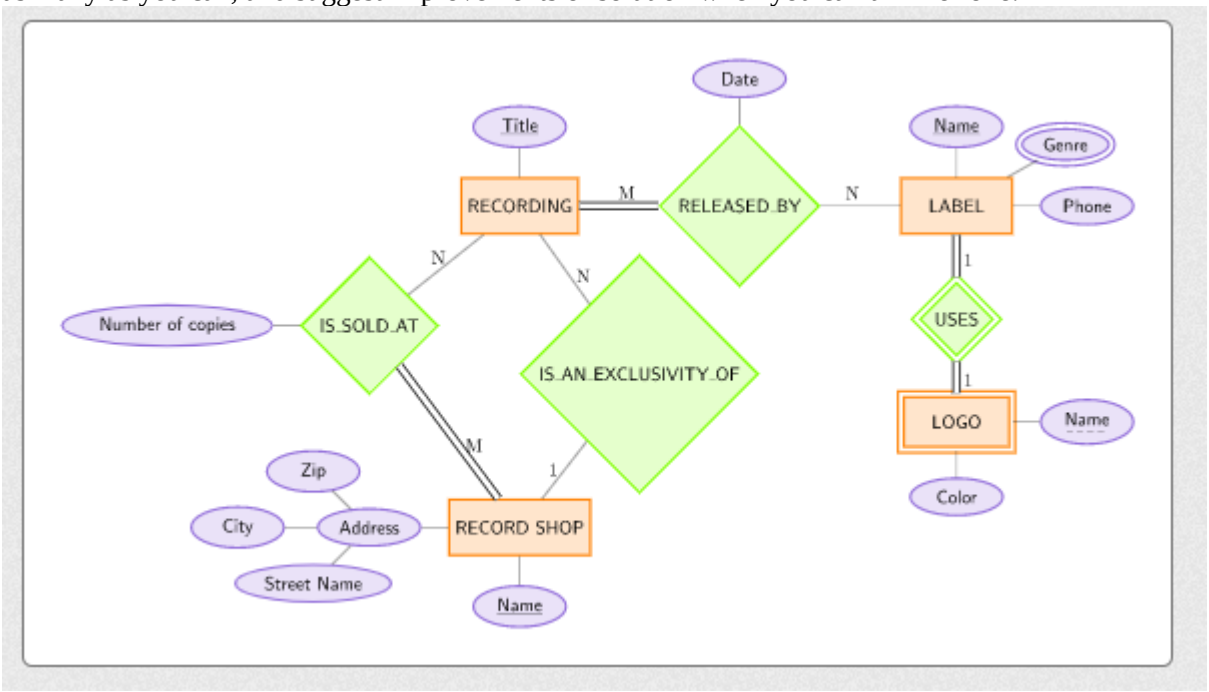
iv. **Specify additional constraints on each table other than keys.**
Partial constraints:
Customer: name → not null, phonenumber, ccn → not null
Pizza: pname → not null, size, price range??
Order: (cid, pid) → not null
Supplies: check amountleft >0
Ingredient: (pid, sid) → not null
More constraints can also be applied on it.

Question 4: **(Marks: 10)**

Consider the following ER diagram:



This diagram, to your expert eyes, has multiple flaws, missing constraints, cardinality ratios and has some inconsistencies with their requirements. **Re-draw** the **ER diagram** after correcting the mistakes as many as you can, and suggest improvements or solution when you can think of one.



Marks Distribution of Q4

2 marks for constraints
4 marks for cardinality ratios
2 marks for attributes
2 marks for identifying weak entity and identifying relationship

Question 5 **(Marks: 10)**

Enhanced Entity Relationship Diagram

Consider a database system for a baseball organization such as the major leagues. The data requirements are summarized as follows:

- The personnel involved in the league include players, coaches, managers, and umpires. Each is identified by a unique personnel id. They are also described by their first and last names along with the date and place of birth. (2)

In this part you have to make Personnel a Super Entity and players, coaches, managers, and umpires as Sub Entity, followed by the attributes of Personnel Entity.

- Players are further described by other attributes such as their batting orientation (left, right, or switch) and have a lifetime batting average (BA). (1)

In this you have to make batting orientation and bating average as attributes of Players Entity, batting orientation is composite attribute and further decomposed into left, right, or switch.

- Within the players group is a subset of players called pitchers. Pitchers have a lifetime ERA (earned run average) associated with them. (1)

Pitcher is further a subtype of Players.

- Teams are uniquely identified by their names. Teams are also described by the city in which they are located and the division and league in which they play (such as Central division of the American League). (2)

In this you have to make attributes of Teams, as names, city, division and league. Names is the primary key.

- Teams have one manager, a number of coaches, and a number of players. (2)

Cardinality ratios needs to be defined in this part.

- Games are played between two teams with one designated as the home team and the other the visiting team on a particular date. The score (runs, hits, and errors) are recorded for each team. The team with the most runs is declared the winner of the game. (1)

There is a relation between Teams and Games and score will be the attribute of the relationship defined.

- With each finished game, a winning pitcher and a losing pitcher are recorded. In case there is a save awarded, the save pitcher is also recorded.
- With each finished game, the number of hits (singles, doubles, triples, and home runs) obtained by each player is also recorded. (1)

You can make Hits as Entity that will be connected with players or you can connect players and game and number of hits will be the attribute of relationship defined.

Design an Enhanced Entity-Relationship diagram for the BASEBALL database.

Marks Distribution of Q5 is given in front of each part in Q5

Question 6 **(Marks: 20)**

Joins & Indexing

(i). Enlist students first name and last name in single column (use alias "Top_3_postions") along with total score for top three students who have the highest total score across all subjects obtained in the exams (Hint: Use join). (8 marks)

```
SELECT TOP 3 s.first_name+' '+s.last_name as student_name, sum(g.scores) as
Total_Score
        FROM graded g, exams e, students s
        WHERE g.student_ID = s.student_ID
        AND g.exam_id = e.exam_id
        GROUP BY g.student_ID,s.first_name,s.last_name
        ORDER BY sum(g.scores) desc;
```

Exam Table (Sample Data)

exam_name	exam_id	date
mathematics	1	2018-06-01
linear algebra	2	2018-06-06
chemistry	3	2018-06-11
physics	4	2018-06-15
english language	5	2018-06-20

Graded Table (Sample Data)

student_ID	exam_id	scores
1001	1	61
1002	1	5
1001	2	98
1002	2	71
1001	3	60
1002	3	62
1001	4	36
1002	4	85
1001	5	90
1002	5	21

Student Table (Sample Data)

student_ID	first_name	last_name
1001	Dillon	Neitzel
1002	Bridgette	Viruet
1003	Lean	Wessel
1004	Corey	Mogan
1005	Amberly	Schneiderman

(ii). Write a nested/correlated query to retrieve student ID, exam ID and exam score of those students having exam score less than the average score for that particular exam. (7 marks)

```
SELECT student_ID, exam_id, scores FROM graded g1 WHERE g1.scores < (SELECT
AVG(scores)
FROM graded g2  WHERE g1.exam_id = g2.exam_id);
```

(iii). For both part 1 and 2 assuming the same tasks, create an index on the suitable attribute's for the better performance also give solid reasoning/justification (with query) why have you chosen the specific attribute's and the type of indexing. (5 Marks)

Query: 3marks

```
create index XYZ
on Graded (student_ID, exam_id,scores)
```

Reasoning: 2marks

The table which has primary-key attributes of other tables that is Graded. Creating an index on such a table saves time.
(similar explanations and index created on other tables with a primary key is also acceptable).

Question 7: **(Marks: 5)**

Transaction Processing

Develop a transaction scenario in the form of SQL language, in which you have to transfer payment from one bank account to another. Make sure that the payment comes out of one account, and into the other, at exactly the same time. Transaction should be designed in a way that it must not be lost from both accounts or be duplicated in both accounts.

Assume account IDs as numeric such as Account ID 6512 and 6257 with some initial balance in each account. Transfer amount should be last four digits of your Roll No.

```
CREATE TABLE accounts (account_id NUMBER(?), balance NUMBER (?));
INSERT INTO accounts VALUES (ID1 e.g. 6512, Bal in ID1 e.g. 1500);
INSERT INTO accounts VALUES (ID2 e.g. 6257, Bal in ID2 e.g. 2000);
DECLARE
transfer NUMBER (?): = 500/Roll_NO;
       BEGIN
       UPDATE accounts SET balance = balance - transfer WHERE account_id =
       6512;
       UPDATE accounts SET balance = balance + transfer WHERE account_id =
       6257;
       COMMIT COMMENT 'Transfer From 7715 to 7720' WRITE IMMEDIATE NOWAIT;
       END;
```

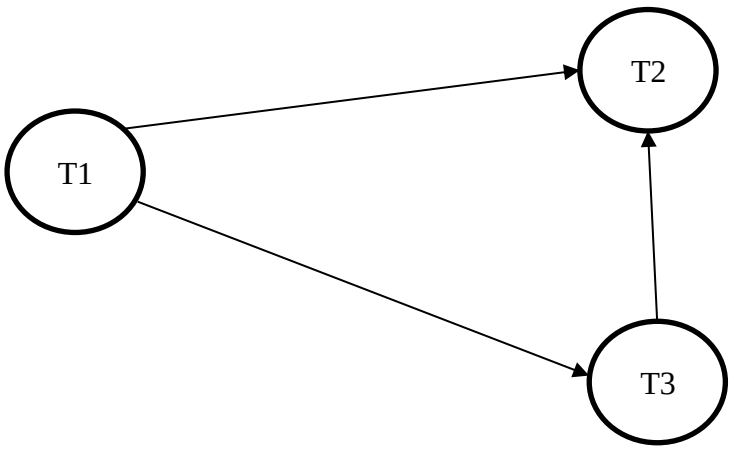
Question 8. **(Marks: 10)**

Database Transaction Scheduling

Draw a precedence graph for the following three transactions and write down the conflict equivalent schedule.

Time	T_1	T_2	T_3
t_0	Read (A)		
t_1	Read (B)		
t_2		Read (A)	
t_3		Read (C)	
t_4	Write (B)		
t_5	Commit		
t_6			Read (B)
t_7			Read (C)
t_8			Write (B)
t_9		Write (A)	Commit
t_{10}		Write (C)	
t_{11}		Commit	

END



Time	T_1	T_2	T_3
t_0	Read (A)		
t_1	Read (B)		
t_2	Write (B)		
t_3	Commit		
t_4			Read (B)
t_5			Read (C)
t_6			Write (B)
t_7			Commit
t_8		Read (A)	
t_9		Read (C)	
t_{10}		Write (A)	
t_{11}		Write (C)	
t_{12}		Commit	