Final Examination
"Course 1-02-322: Database Systems"
Exam A Answers

Kinneret College School of Engineering

February 1, 2010 9:00am-12:00pm

- Answer the following questions in English or Hebrew.
- You may bring two pages of notes to the exam with notes on both sides.
- The number of points for each question is listed next to each one to indicate its weight.
- There are a total of **70** points on the test. You must answer all of the questions.
- Write all of your answers in the test booklet which you received.
- Marks made on the test sheets will not be counted or graded.
- You must return the test questions sheet at the end of the exam.

1 Short Answers (9 points / 3 points each)

Briefly explain the following terms as they relate to databases:

1. Isolation
   - The guarantee that any transaction executing will not be aware of any other transactions executing concurrently.
2. Durability
   - The guarantee that any transaction which has committed will not be lost
3. Candidate Key
   - A column or set of columns which uniquely identifies each row in every valid instance of the relation

2 Transactions (12 points / 3 points each)

Consider the following schedule $S1$:

T1	T2
	W(A)
R(A)	
R(B)	
	R(B)
W(D)	
	W(D)
W(D)	
Commit	
	Commit

Answer the following questions about $S1$:

(a) Is $S1$ (view) serializable? If yes, what serial schedule is it the same as? If not, explain why and list all conflicts.
   - It is serializable to T2;T1

(b) Is $S1$ permitted by Strict 2PL? If not, explain why and list all conflicts.
   - It is forbidden. T1 reads A while T2 has an exclusive lock on it. T2 writes D when T1 has an exclusive lock on it.

(c) Does $S1$ avoid cascading aborts? Why?
   - No, after T1 reads A, T1 would need to be aborted if T2 would abort.

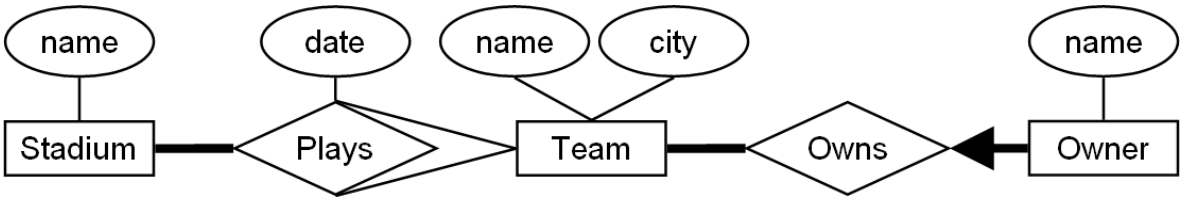
(d) Is $S1$ recoverable? Why?
   - No, If T2 would abort after T1 has committed, we could not recover.

3 Entity Relationship Diagrams (6 points)

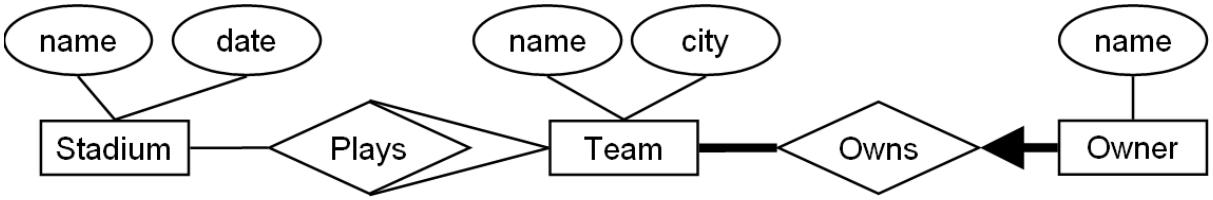
Indicate which Entity Relationship Diagram (if any) **exactly** matches the description below (no extra constraints, no missing constraints).

> Football teams have a *name*, a *city*, and one or more *owners*. Every owner (in the database) (identified by *name*) owns exactly one football team. Two football teams play games against each other in *stadiums* (identified by name) on particular dates. Teams may play zero or more games. Stadiums have zero or more games.

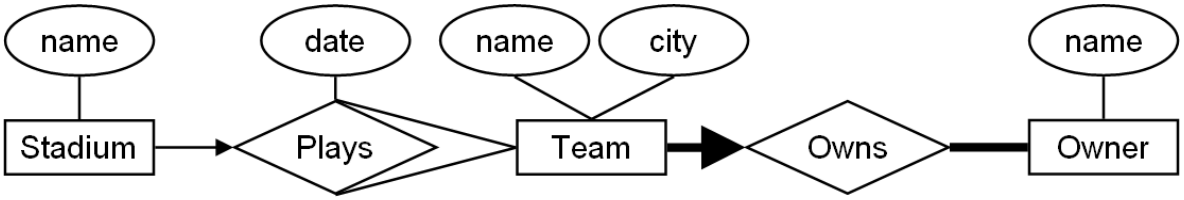
(a)



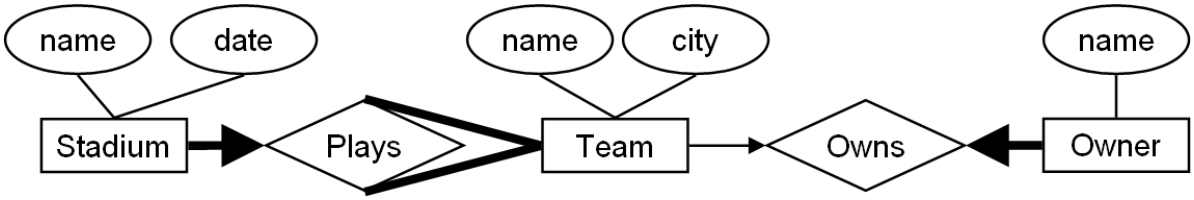
(b)



(c)



(d)



(e) None of the above.

- Correct.

4 Relational Algebra (14 points)

Consider the following relational schema:

Teams (tname:CHAR(20), city:CHAR(20), averagePoints:real)
Players (tname:CHAR(20), pname:CHAR(20), age:int)
Games (tname:CHAR(20), gameDate:datetime, stadium:string, score:int)

Write Relational Algebra expressions to evaluate the following queries:

(a) (4 points) Show the *pnames* of all players who are on a team from the city 'Tiberias'

$$\pi_{pname}(\sigma_{(city='Tiberias')}(Teams) \bowtie Players)$$

(b) (4 points) Show the *tname* and *city* of each team which has played zero games.

$$\pi_{(tname,city)}(Teams \bowtie (\pi_{tname}(Teams) - \pi_{tname}(Games)))$$

(c) (6 points) Show the *tname* and *city* of each team which has not lost or tied any games (*i.e.* its score is higher than the other team in every game it has played, if it has played any games).

$$\rho(G1(1 \to tname1, 4 \to score1), Games)$$

$$\rho(G2(1 \to tname2, 4 \to score2), Games)$$

$$\rho(Loser(tname1 \to tname), \pi_{tname1}(\sigma_{(score1<score2)}(G1 \bowtie G2)))$$

$$\pi_{(tname,city)}(Teams \bowtie (\pi_{tname}(Teams) - \pi_{tname}(Loser)))$$

Note: Many students wrote the following **wrong** formulation:

$$\rho(G1(1 \to tname1, 4 \to score1), Games)$$

$$\rho(G2(1 \to tname2, 4 \to score2), Games)$$

$$\rho(Temp(tname1 \to tname), \pi_{tname1}(\sigma_{(tname1<>tname2 \wedge score1>score2)}(G1 \bowtie G2)))$$

$$\pi_{(tname,city)}(Teams \bowtie Temp)$$

This is **wrong** because Temp will contain any team which won **at least one game**, not the requested query - a team which has never lost or tied a game.

5 SQL Queries (20 points)

Consider the following relational schema:

Teams (tname:CHAR(20), city:CHAR(20), averagePoints:real)
Players (tname:CHAR(20), pname:CHAR(20), age:int)
Games (tname:CHAR(20), gameDate:datetime, stadium:CHAR(20), score:int)

Write SQL expressions for the following queries. Ensure that there are no duplicates in any results:

(a) (4 points) Show the *pname*s of all players who are on a team from the city 'Tiberias'

```
SELECT DISTINCT P.pname FROM Players P, Teams T WHERE P.tname = T.tname AND t.city = 'Tiberias'
```

(b) (6 points) Show the *tname* and *city* of each team which has played zero games.

```
SELECT DISTINCT T.tname, T.city FROM Teams T WHERE T.tname NOT IN (SELECT G.tname FROM Games
G)
```

(c) (6 points) Show the *tname* and *city* of each team which has not lost or tied any games (*i.e.* its score is higher than the other team in every game it has played, if it has played any games).

```
SELECT DISTINCT T1.tname, T1.city
FROM Teams T1
WHERE NOT EXISTS
(SELECT *
FROM Games G1, Games G2
WHERE G1.tname = T1.tname
AND G1.gameDate = G2.gameDate
AND G1.stadium = G2.stadium
AND G1.score <= G2.score)
```

Note: Many students provided the following **wrong** answer which parallels the wrong answer for the Relational Algebra query above. For example:

```
SELECT DISTINCT T1.tname, T1.city
FROM Teams T1, Games G1, Games G2
WHERE T1.tname = G1.tname
AND T1.tname <> G2.tname
AND G1.gameDate = G2.gameDate
AND G1.score > G2.score
```

As in the relational algebra question, the above query just returns teams which have won at least one game, not what the question asked for.

(d) (4 points) For each *gameDate*, show the *date* and the highest *score* of all games played on that date, and the number of *teams* which played on that data.

```
SELECT DISTINCT G.date, MAX(score), count(tname) FROM Games G GROUP BY G.date
```

6 Triggers (9 points / 3 points each)

Consider the following relational schema:

Teams (tname:CHAR(20), city:CHAR(20), averagePoints:real)
Players (tname:CHAR(20), pname:CHAR(20), age:int)
Games (tname:CHAR(20), gameDate:datetime, stadium:CHAR(20), score:int)

Consider the following trigger written in MS SQL Server 2005 syntax:

```
CREATE TRIGGER t1 ON Games AFTER INSERT, UPDATE AS
DECLARE @var1 CHAR(20)
DECLARE @var2 REAL
SELECT @var1 = (SELECT DISTINCT tname FROM inserted I)
SELECT @var2 = (SELECT AVG(score) FROM Games G WHERE G.tname = @var1)
UPDATE Teams SET averagePoints = @var2 WHERE tname = @var1
```

Assume the tables Teams and Games have been initialized as follows:

Teams:

tname	city	averagePoints
team1	Tiberias	20
team2	Bet Shean	21
team3	Tzefat	20

Games:

tname	gameDate	stadium	score
team1	2010-01-01	Beit Bendel	20
team2	2010-01-01	Beit Bendel	21
team1	2010-02-02	Hula	20
team3	2010-02-02	Hula	20

For each of the following commands, indicate what the resulting state of the Teams and Games tables will be. Consider each command separately, **without respect to the previous commands**:

6.1 UPDATE Games SET score = 30 WHERE tname = 'team2'

A. Teams:

tname	city	averagePoints
team1	Tiberias	20
team2	Bet Shean	30
team3	Tzefat	20

Games:

tname	gameDate	stadium	score
team1	2010-01-01	Beit Bendel	20
team2	2010-01-01	Beit Bendel	30
team1	2010-02-02	Hula	20
team3	2010-02-02	Hula	20

Correct

B. Teams:

tname	city	averagePoints
team1	Tiberias	20
team2	Bet Shean	21
team3	Tzefat	20

Games:

tname	gameDate	stadium	score
team1	2010-01-01	Beit Bendel	20
team2	2010-01-01	Beit Bendel	30
team1	2010-02-02	Hula	20
team3	2010-02-02	Hula	20

C. Teams:

tname	city	averagePoints
team1	Tiberias	20
team2	Bet Shean	25
team3	Tzefat	20

Games:

tname	gameDate	stadium	score
team1	2010-01-01	Beit Bendel	20
team2	2010-01-01	Beit Bendel	30
team1	2010-02-02	Hula	20
team3	2010-02-02	Hula	20

D. The command will cause an error and the tables will be unchanged.

6.2 DELETE FROM Games WHERE stadium = 'Beit Bendel'

A. Teams:

tname	city	averagePoints
team1	Tiberias	20
team2	Bet Shean	21
team3	Tzefat	20

Games:

tname	gameDate	stadium	score
team1	2010-02-02	Hula	20
team3	2010-02-02	Hula	20

Correct

B. Teams:

tname	city	averagePoints
team1	Tiberias	20
team2	Bet Shean	0
team3	Tzefat	20

Games:

tname	gameDate	stadium	score
team1	2010-02-02	Hula	20
team3	2010-02-02	Hula	20

C. Teams:

tname	city	averagePoints
team1	Tiberias	20
team2	Bet Shean	0
team3	Tzefat	20

Games:

tname	gameDate	stadium	score
team1	2010-01-01	Beit Bendel	20
team2	2010-01-01	Beit Bendel	21
team1	2010-02-02	Hula	20
team3	2010-02-02	Hula	20

D. The command will cause an error and the tables will be unchanged.

6.3 UPDATE Games SET score = score + 1 WHERE stadium = 'Hula'

A. Teams:

tname	city	averagePoints
team1	Tiberias	20
team2	Bet Shean	21
team3	Tzefat	20

Games:

tname	gameDate	stadium	score
team1	2010-01-01	Beit Bendel	20
team2	2010-01-01	Beit Bendel	21
team1	2010-02-02	Hula	21
team3	2010-02-02	Hula	21

B. Teams:

tname	city	averagePoints
team1	Tiberias	20.5
team2	Bet Shean	21
team3	Tzefat	21

Games:

tname	gameDate	stadium	score
team1	2010-01-01	Beit Bendel	20
team2	2010-01-01	Beit Bendel	21
team1	2010-02-02	Hula	21
team3	2010-02-02	Hula	21

C. Teams:

tname	city	averagePoints
team1	Tiberias	20.5
team2	Bet Shean	21
team3	Tzefat	21

Games:

tname	gameDate	stadium	score
team1	2010-01-01	Beit Bendel	20
team2	2010-01-01	Beit Bendel	21
team1	2010-02-02	Hula	21
team3	2010-02-02	Hula	21

D. The command will cause an error and the tables will be unchanged. **Correct**