The University of British Columbia

Computer Science 304

Midterm Examination

October 3, 2013

Time: 75 minutes Total marks: 44

Instructor: Rachel Pottinger

Name ANSWER KEY Student No

(PRINT) (Last) (First)

Signature

This examination has 3 double-sided pages.

Check that you have a complete paper.

This is a closed book, closed notes exam. No books or other material may be used.

Answer all the questions on this paper.

Give very **short but precise** answers.

State any assumptions you make

Work fast and do the easy questions first. Leave some time to review your exam at the end.

Good Luck

Question	Mark	Out of
1		10
2		15
3		9
4		10
Total		44

1. { 10 marks}

a.	In ER diagrams, a relationship set can exist between an entity set and itself. *True. This is the case when we need to have "roles" for the entities.*	*TRUE* *FALSE*
b.	In relational algebra, two tuples for the relation R(a, <u>b</u>) can have the same value for a. *True. There cannot, however, be duplicate values for the entire relations.*	*TRUE* *FALSE*
c.	We use ER diagrams to logically model our data. *False. ER diagrams conceptually model our data. Relational models do the logical modeling.*	*TRUE* *FALSE*
d.	It is possible for a table in a relational schema to have more than one key. *True. It is not, however, possible for a table to have more than one* primary *key.*	*TRUE* *FALSE*
e.	We can check a database instance to verify that an integrity constraint holds. *True. It is not, however, possible to tell the integrity constraints by looking at the instance.*	*TRUE* *FALSE*

2. {15 marks}

Consider the schema that we have used in class:

Movie(<u>MovieID</u>, Title, Year)
StarsIn(<u>MovieID</u>, <u>StarID</u>, Character)
MovieStar(<u>StarID</u>, Name, Gender)

Write relational algebra queries to answer the following. For each query, return EXACTLY (and only) what is asked for. State any assumptions that you need to make.

a. Find the titles of all the movies that have only male movie stars

An easy way to do this is to subtract off the movies that have female movie stars. Movies with female stars are (StarsIn $\bowtie(\sigma_{Gender = 'female'}$ *MovieStar)). So we can project out the movie ID, do a set difference of those we don't want, and then join with movie to get the title. Altogether it looks like this:*

π_{Title} *(Movie* $\bowtie$ *(*$\pi_{MovieID}$*Movie -* $\pi_{MovieID}$*(StarsIn* $\bowtie(\sigma_{Gender = 'female'}$ *MovieStar))))*

Note that there is more than one way to do this query; this is just one example of how it could be done correctly.

b. Find the titles of all movies where an actor plays him/herself (e.g., in "Being John Malkovich", John Malkovich plays John Malkovich).

The only thing interesting about this query is that you can't use a natural join unless you do a careful selection later. Otherwise, it's what you'd expect:

π_{Title}*((MovieStar* $\bowtie_{name = character \wedge MovieStar.StarID = StarsIn.StarID}$ *StarsIn)* $\bowtie$ *Movie)*

Note that there is more than one way to do this query; this is just one example of how it could be done correctly.

c. Find the titles of all movies that have *all* the female actors from the movie titled "The Thing" in them.

This question is most easily answered by division. First we find the female actors from the thing:

thingwomen $\leftarrow$ π_{StarID}*((*$\sigma_{Gender = 'female'}$ *MovieStar)* $\bowtie$*StarsIn* $\bowtie(\sigma_{title = 'The\ Thing'}$ *Movie)*

then we divide all the movies that have these stars, being careful to set up our division properly, and then join back to get the titles and project them back out. You could also leave the title on and avoid the last join, but this only works because MovieID is a key. So in general, being minimal about what is in the numerator is a better solution:

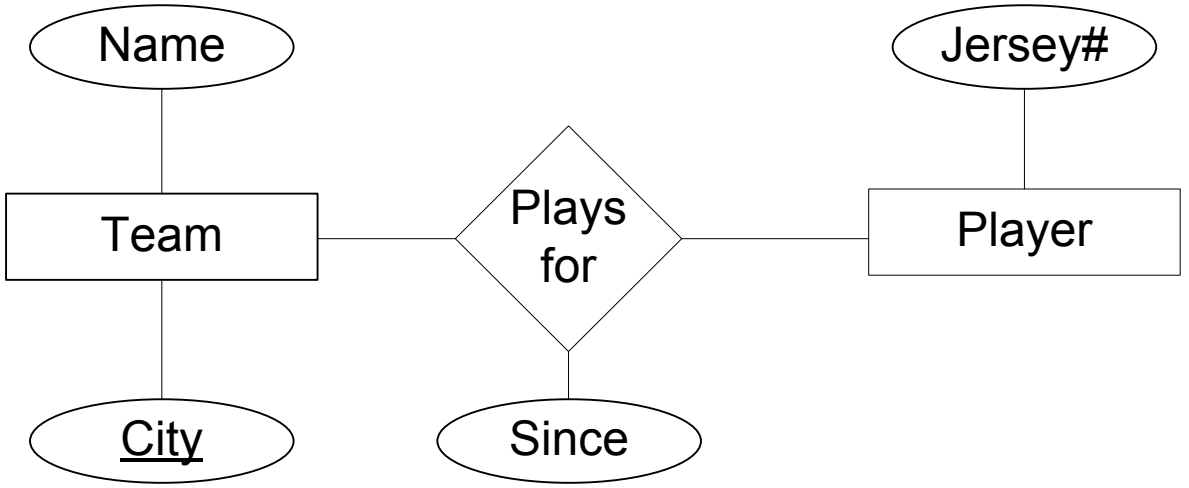
π_{title}*((*$\pi_{MovieID,\ StarID}$*(StarsIn))/thingwomen)* $\bowtie$ *Movie)*

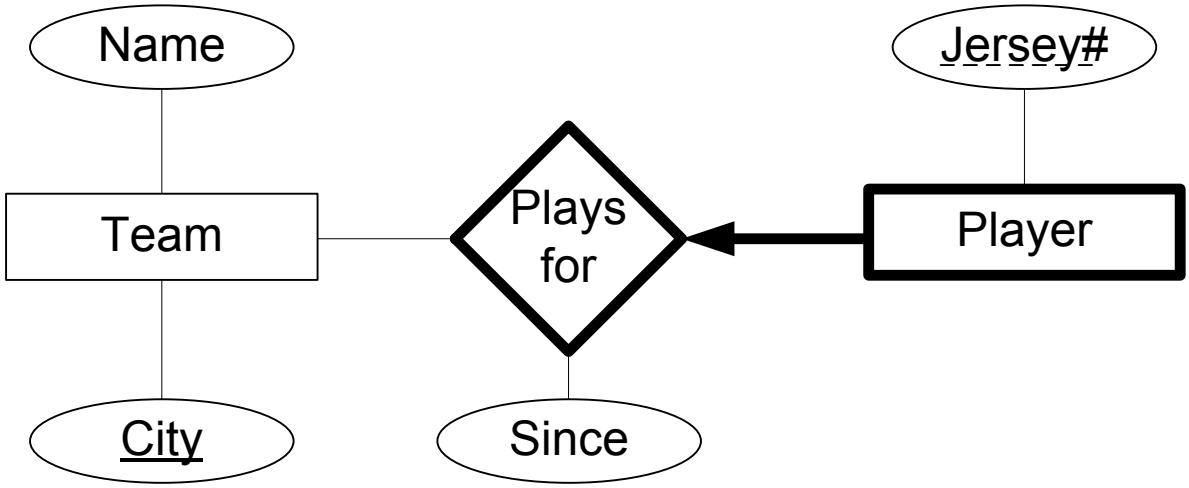
Note that there is more than one way to do this query; this is just one example of how it could be done correctly.

3. {9 marks}

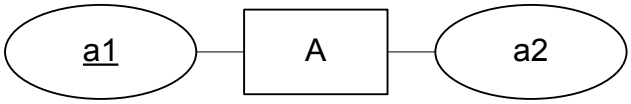
For each part below, annotate the related diagram so that it provides the additional requested functionality– ***do not add any additional items or constraints beyond what is required***. If nothing needs to be done to the diagram or it is impossible to add that constraint in our version of ER diagram, state why. State any assumptions.

a. The entity player depends on Team for its existence. The key of Player is the team's city and the Player's Jersey #



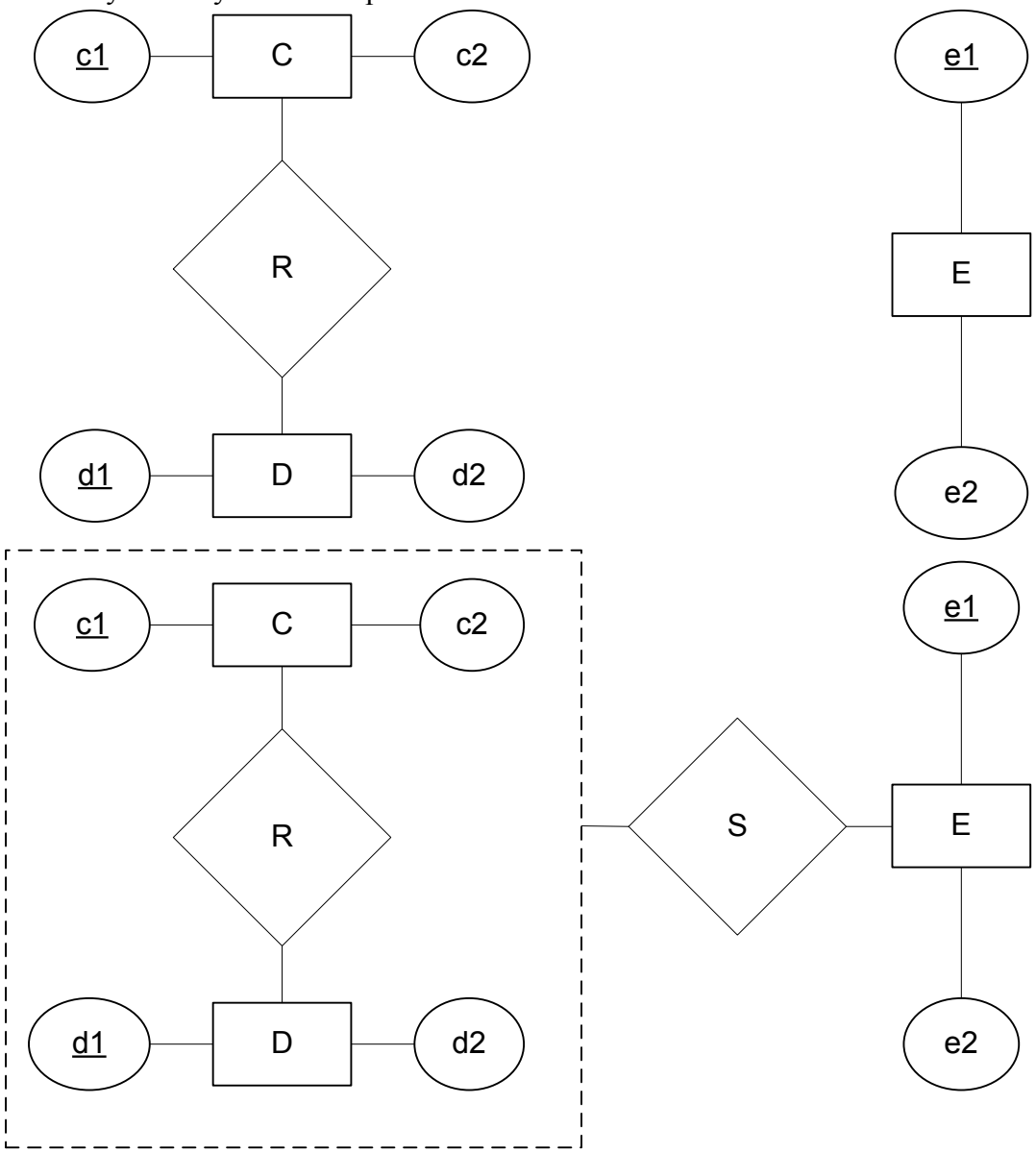


b. a2 is another key of A



Cannot be done. In an ER diagram, one can only illustrate exactly one key (the primary key).

c. E is in a many to many relationship with R



4. {10 marks}

Suppose that we have a ternary relationship T between entity sets D, E, and F such that D has a key constraint and E has a key constraint and total participation; these are the only constraints. D has attributes d1 and d2, with d1 being the key; E has attributes e1 and e2, with e1 being the key; and F has attributes f1 and f2, with f1 being the key. T has no descriptive attributes. All attributes are integers. Write SQL statements that create tables corresponding to this information so as to capture as many of the constraints as possible. If you cannot capture some constraint, explain why.

This is isomorphic to problem 3.11 from the book.and problem 5 from Practice test 7

Answer: The following SQL statements create the corresponding relations.

```
CREATE TABLE E_T (
e1 int,
e2 int,
d1 int NOT NULL,
f1 int NOT NULL,
PRIMARY KEY (e1),
UNIQUE (d1),
FOREIGN KEY (d1) REFERENCES D,
FOREIGN KEY (f1) REFERENCES F )
CREATE TABLE D (
d1 int,
d2 int,
PRIMARY KEY (d1) )

CREATE TABLE F (
 f1 int,
f2 int,
      PRIMARY KEY (f1) )
```

The first SQL statement folds the relationship T into table E. Adding not null constraints to d1 and f1 guarantees participation of E in the relationship, T.