

# CS 564 Midterm Exam

## Spring 2020

### Answers

#### A: LET'S WRITE SOME QUERIES! [30%]

---

Consider the following database schema about movies:

**Movie** (mid, title, year, language, duration)

**Actor** (aid, firstname, lastname, gender)

**ActedIn** (mid, aid, character)

**Genre** (mid, genre)

ActedIn.mid is a foreign key referring to Movie.mid, ActedIn.aid is a foreign key referring to Actor.aid, and MovieGenre.mid is a foreign key referring to Movie.mid.

1. [10%] Express the following query in SQL: *for each movie genre, count the number of 2019 movies having that genre.*

```
SELECT g.genre, COUNT(m.mid)
FROM Movie m, Genre g
WHERE m.mid = g.mid AND m.year = 2019
GROUP BY g.genre ;
```

2. [10%] Express the following query in SQL: *output the first and last name of the actors who have played two different characters in the same movie.*

```
SELECT DISTINCT a.firstname, a.lastname
FROM Actor a, ActedIn c
WHERE c.aid = a.aid
GROUP BY a.aid, c.mid, a.firstname, a.lastname
HAVING COUNT(*) > 1;
```

3. [10%] Write a SQL query equivalent to the following query in Relational Algebra:

$$\pi_{mid,title}(Movie) - \pi_{mid,title}(\sigma_{gender= male}(Actor) \bowtie ActedIn \bowtie Movie)$$

BONUS [2%]: write the query without using EXCEPT.

```
SELECT DISTINCT mid, title
FROM Movie
WHERE mid NOT IN (
    SELECT m.mid
    FROM Movie m, Actor a, ActedIn c
    WHERE m.mid = c.mid AND a.aid = c.aid AND a.gender = "male") ;
```

## B: MORE SQL [15%]

---

For the following questions, circle **exactly one** correct option.

1. [5%] Consider the schema from Part A. The following SQL queries will always give the same result (assume no NULLs):

```
SELECT language FROM Movie GROUP BY language ;
```

```
SELECT language FROM Movie ;
```

**FALSE**

2. [5%] Consider the schema from Part A, and the following schema for Genre.

```
CREATE TABLE Genre (
    mid      INTEGER,
    genre    CHAR(20),
    FOREIGN KEY (mid) REFERENCES Movie(mid)
) ;
```

If we attempt to delete a row from the table `Genre`, the query will be rejected.

**FALSE**

3. [5%] The following SQL queries will always give the same result (assume no NULLs):

```
SELECT A FROM T ;
```

```
SELECT A FROM T
UNION
SELECT A FROM T ;
```

**FALSE**

## C: NORMALIZATION AND FDs [39%]

Consider the relation  $R(A, B, C, D, E)$  with the following set  $F$  of functional dependencies:

$$A \rightarrow B \quad A \rightarrow C \quad B \rightarrow C \quad C \rightarrow A \quad D \rightarrow E \quad E \rightarrow D$$

For the following questions, circle the right option(s).

**#1 There can be more than one correct options for every question!**

#2 You can get partial credit by explaining how you came up with your answer.

1. [10%] The number of keys in relation  $R$  is:



**ANSWER: 6**

2. [10%]  $F$  is a minimal basis for  $F$ :



**FALSE:** The FD  $A \rightarrow C$  is redundant.

3. [10%] Relation  $R$  is in:



**ANSWER:** 3NF but not BCNF

4. [9%] The following decompositions of  $R$  are dependency-preserving:



**ANSWER: (i) and (iii)**

## D: BUFFER MANAGEMENT [16%]

Consider a buffer pool with 7 frames, and a file that consists of 10 pages. Consider the following sequence of page requests: 1, 2, 3, . . . , 10, 10, 9, 8, . . . , 1. Each page request is followed directly by a page release (before the next page request). Initially, all

buffer frames are free, and none of the pages to be accessed are in RAM. We say that a page request is a **page hit** if the page is already in the buffer pool.

For the following questions, circle the right option.

**#1 You can get partial credit by explaining how you came up with your answer.**

1. [10%] If the buffer manager uses the **LRU** replacement policy, how many of the 20 page requests will be page hits?

(i) 0      (ii) 7      (iii) 10      (iv) 13

**ANSWER: 7**

2. [6%] Suppose that the above sequence changes such that every page is **modified** before being released (so page 1 is requested, then modified, then released, etc.). How many I/Os (both writes and reads) occur using the **LRU** replacement policy?

(i) 13      (ii) 16      (iii) 19      (iv) 26

**ANSWER: 19**