Name	NetID

CS411: Database Systems **Spring 2018**

Midterm, March 5

- READ THESE INSTRUCTIONS CAREFULLY BEFORE YOU START. DO NOT turn this page UNTIL we instruct you to.
- First: write your name and NetID at the top of this sheet.
- The exam lasts for 75 minutes, i.e., from 8–9.15am.
- We will not answer any questions during the exam. If you need to make any assumptions for any of the questions, please feel free to do so and then clarify the assumption in your answer.
- All questions are compulsory.
- For each question, please answer in the space provided; if you need more space, feel free to use the back side of each page. We will not provide additional space.
- The maximum score you can obtain is $18 + 20 + 12 + 24 + 26 = 100$.
- You must stop writing when time is called.
- **Cheating: No.**

Question	1	2	3	4	5	Total
Points						

1 Short Questions - 18 points

1. [3] True/False questions: answer "true" or "false" for each statement. No explanation necessary.

   (a) The relationship from a weak entity set E to its supporting entity set F must be a binary, many-one relationship with a rounded arrow on F.
   [**A**nswer: True. (one-one relationship is a special case of many-one relationship.)
   Rubric. no mark deduction.]

   (b) When converting subclasses into relations using the Object-Oriented (OO) approach, each entity only exists in one relation.
   [**A**nswer: True.
   Rubric. -1 : wrong answer.]

   (c) Consider three entity sets: Flights, Customers, and Agencies. There is a multi-way relationship Reservations connecting the three entity sets. We can encode the requirement that a customer may use only one agency for all of their flight reservations by placing appropriate arrows in the E/R diagram.
   [**A**nswer: False.
   Rubric. -1 : wrong answer.]

2. [5] Given a relation $R(A)$, express $\min(A)$ using set relational algebra, i.e., the minimum value of A in R. You can only use the following operators: projection π, selection σ, rename ρ, join $\bowtie$ and set difference $-$. (Hint: you may not need all of them.) No explanation necessary. Only provide the relational algebra expression.

   [**A**nswer: $\min(A) = R - (\pi_A(\rho_{T1(B)}(R) \bowtie_{B<A} R))$
   Rubric. -3: not returning min.
   -1: no projection after theta join.
   -1: set difference is not with the same attribute name.
   -1: wrong notation, e.g., mistakenly use σ for π.
   ]

3. [10] There are three distinct properties we would like a decomposition (or normalization) to have:

   (a) Elimination of Anomalies, i.e., redundancy, update anomalies and delete anomalies.

   (b) Recoverability of Information, i.e., lossless decomposition.

   (c) Preservation of Dependencies, i.e., the original functional dependencies are satisfied after table reconstruction.

   Which of the above three properties are always satisfied in Boyce-Codd Normal Form (BCNF) and Third Normal Form (3NF), respectively? Give a succinct counter example if the property is not guaranteed to be true. If the property is true, just say so – no explanation is needed in this case.
   [**A**nswer: BCNF satisfies (a) and (b), but not (c).

Rubric. -1: each wrong answer. In total 3 pts.
Counter example: consider a relation $R(A, B, C)$ with functional dependency $AB \rightarrow C$ and $C \rightarrow A$. After BCNF decomposition, we have $R1(B, C)$ and $R2(A, C)$. However, $R1$ and $R2$ can not guarantee FD $AB \rightarrow C$.
Rubric. -2: wrong example or no example.
3NF satisfies (b) and (c), but not (a).
Rubric. -1: each wrong answer. In total 3 pts.
Counter example: continue the above example, $R(A, B, C)$ is in 3NF. However, when we want to update the value of attribute A for a particular value of attribute C, we may incur the update anomaly.
Rubric. -2: wrong example or no example. A common mistake: "there is redundancy in relations after 3NF decomposition. E.g., $R(A, B, C, D)$ with FD $AB \rightarrow C$, (A, B, C) and (A, B, D) after 3NF decomposition share attribute AB. Hence, redundancy." This is a wrong claim. First, "(a) Elimination of Anomalies" is in terms of the relation before decomposition. Second, BCNF would also result in (A, B, C) and (A, B, D), but BCNF can eliminate anomalies.
]

2 Database Design - 20 points

1. [10] Draw an ER diagram of a database model for a blogging platform, capturing the following information:

   - A "Blog" has a unique "URL", along with a "title".
   - A "User" has a unique "uID", along with a "name".
   - Any user can opt in to become a "Writer", which has an additional attribute "bio".
   - Any user can opt in to become a "Commenter", which has an additional attribute "karma".
   - An "Article" has an "aNo" (article number) and a "date". Note, however, that articles from different blogs can have the same "aNo". Hint: you can use the "Blog" to help uniquely identify an article.
   - A "Comment" has a unique "cID", along with a "date".
   - Each article or comment is written by exactly one user.
   - Each comment refers to exactly one article.

   Make sure to underline the key for each entity set. Pay attention to the relationship multiplicity specified in the model design, and clearly state additional assumptions if necessary.

   Answer: See Figure 1.

   Rubric: 1 point for each of the 6 entity sets; 1 point for each of the 4 relationships (excluding "isa"). (Minimal) keys and multiplicity constraints must be correct.

2. [10] Construct a relational database schema corresponding to the ER Diagram that you have drawn. Make sure to underline the key for each relation. If you need to translate subclass entity sets, use the ER approach. Do not create redundant relations!

   Answer:

   Blog(<u>URL</u>, title)
   User(<u>uID</u>, name)
   Writer(<u>uID</u>, bio)
   Commenter(<u>uID</u>, karma)
   Article(<u>blogURL</u>, <u>aNo</u>, date, uID)
   Comment(<u>cID</u>, date, uID, blogURL, aNo)

   Note since each article or comment is written by exactly one user, there is no need to create relations for the "articleBy" or "commentBy" relationships. Similarly, since each comment refers to exactly one article, the "commentOn" relationship can also be captured within the relation for Comment alone (but be aware that Article is a weak entity set).

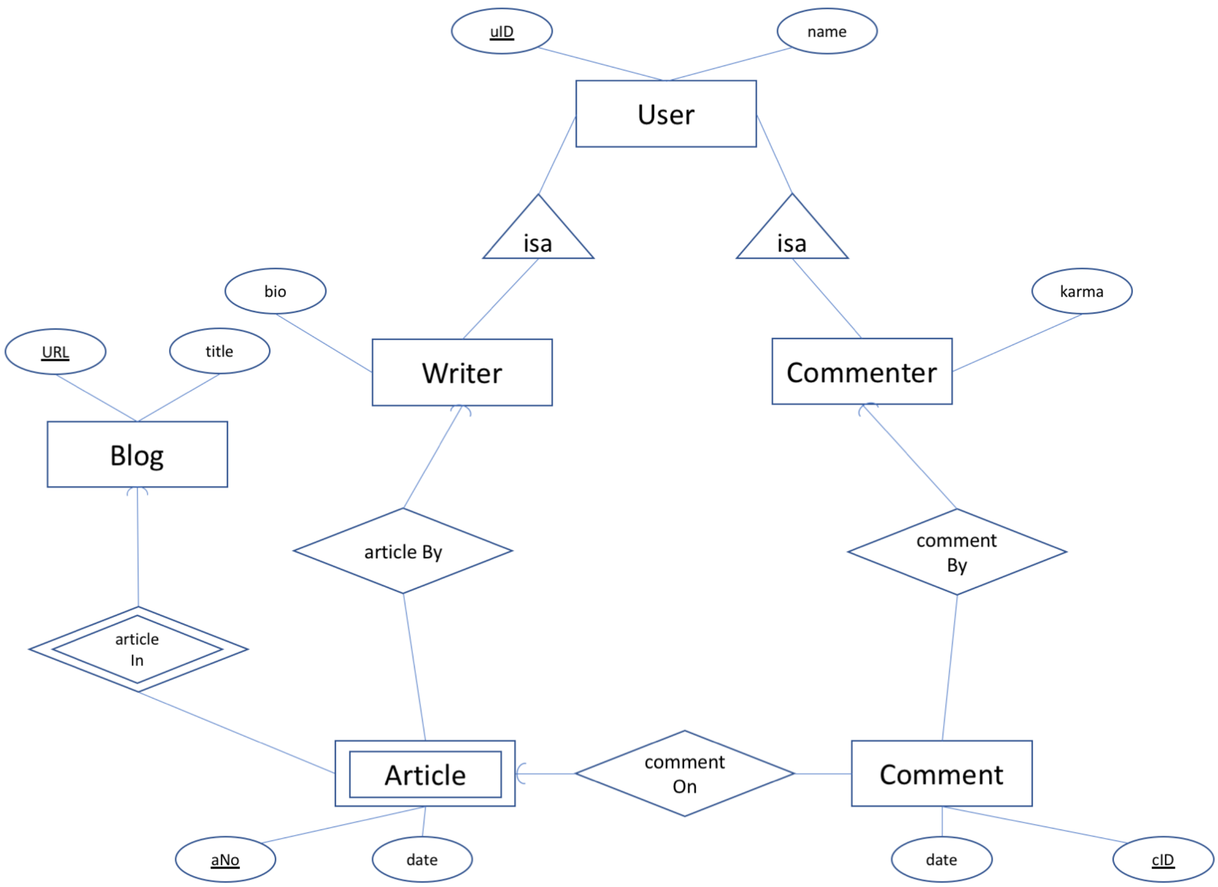

Figure 1: ER Diagram for Database Design

Rubric: 1 point for each of "Blog", "User", "Writer", and "Commenter". 3 points for each of "Article" and "Comment". (Minimal) keys must be correct. Must not create unnecessary relations for any of the 4 relationships.

3 Functional Dependencies - 12 points

1. Consider the following rules, where "$\Longrightarrow$" indicates that the FD on the right hand side can be derived from the ones on the left hand side.

   - $A \to B,\ BC \to D \Longrightarrow AC \to D$
   - $AB \to C \Longrightarrow A \to C$
   - $A \to B_1B_2B_3, B_2 \to C \Longrightarrow A \to C$
   - $A \to C,\ B \to C,\ ABC \to D \Longrightarrow A \to D$

   Are these rules correct? For each of them, prove using the Armstrong axioms (indicate the name of the rule after each derivation), or, justify otherwise using the simplest example relation instance you can come up with where the rule does not hold.
   Hint: None of the proofs require more than 2 steps.
   [**A**nswer: **Answer:**
   For each rule, 1 pt for correct/incorrect.
   For correct rules, 0.5 pt for each derivation (1 pt total), 0.5 pt for naming each rule (1 pt total). If you do not remember the rule's name but tried your best (e.g "addition" for "augmentation"), for the first one is okay, but if more than one, -0.5 for the whole question.
   For incorrect rules, 2 pts for counterexample.

   1 Correct.
   $A \to B \Longrightarrow AC \to BC$ (augmentation rule)
   $AC \to BC,\ BC \to D \Longrightarrow AC \to D$ (transitivity rule)

   - Incorrect.
   $R(A, B, C)$ with $R = \{(1, 2, 3), (1, 4, 5)\}$
   - Correct.
   $B_2 \to C \Longrightarrow B_1B_2B_3 \to C$ (augmentation rule)
   $A \to B_1B_2B_3, B_1B_2B_3 \to C \Longrightarrow A \to C$ (transitivity rule)
   - Incorrect.
   $R(A, B, C, D)$ with $R = \{(1, 2, 3, 4), (1, 5, 3, 6)\}$

   ]

4 Normal Forms - 24 points

Consider a relation R(A,B,C,D,E) with the following functional dependencies:

- AB $\rightarrow$ C
- BC $\rightarrow$ D
- CD $\rightarrow$ E
- DE $\rightarrow$ A

1 [6] Specify all minimal keys for R.

[**A**nswer: AB, BC, BDE]

Rubric:

-2 for each missing minimal key

-1 for each extra, incorrect minimal key. (max: -3)

2 [6] Which of the given functional dependencies (FDs) are Boyce-Codd Normal Form (BCNF) violations?

[**A**nswer: CD $\rightarrow$ E, DE $\rightarrow$ A]

Rubric:

-3 for each missing violated FD

3 [6] Using the algorithm covered in the lecture for BCNF decomposition, give a decomposition of R into BCNF based on the FDs.

[**A**nswer: (A,D,E), (C,D,E) (B,C,D)

Consider CD $\rightarrow$ E, $CD^+ = \{CDEA\}$. We now get R1(C, D, E, A), R2(C, D, B). DE $\rightarrow$ A violates R1. Decomposing R1, we finally get R3(D, E, A), R4(C, D, E). R2, R3, R4 constitutes the final decomposition.]

Rubric:

-2 for each incorrect/missing decomposition

-3 for not fully expanding each violated FD. For example, R1(C,D, E), R2(C,D, A, B) or R1(C,D, E), R2(A, B, C), R3 (A, B, D). We said "using the algorithm covered in the lecture for BCNF decomposition" (See algorithm detail on p30, 2/21 lecture slide)

4 [6] Can the algorithm produce a different decomposition into BCNF? If so, give it. Otherwise write "No different decomposition". [**A**nswer: No different decomposition]

Rubric:

-2 for each incorrect/missing decomposition

5 Relational Algebra and SQL - 26 points

Welcome to Jokemon world! You have the following relational database schema for a Jokemon game.

- Jokedex (jokemon_name, type, weak_against)
- Skill (name, required_type, required_level)
- Trainer (name, gender, level)
- HasJokemon (trainer_name, jokemon_name, catch_time, level, cp)

The Jokedex is a dictionary of all jokemon. It contains Jokemon's name, its type, and the type of jokemon it is weak against. For simplicity, we assume that each jokemon has a single type itself, and a single type that it is weak against.

The Skill table has information about skills: for each skill, it contains the name of the skill and the requirement for any jokemon to learn that particular skill. To learn a given skill, the jokemon has to have the same type as the required_type AND has to have a level greater than or equal to the required_level. Both requirements have to be satisfied to make the jokemon learn the skill.

The Trainer table has information about trainers: for each trainer, it contains the name, gender, and the trainer level of the trainer.

The HasJokemon table has information about which jokemon were captured by which trainers at what point. When a trainer encounters a jokemon in the jokemon world, they have the ability to capture them: each jokemon encountered has a certain level, and a certain combat potential, or cp. This table contains the following information, one for each jokemon captured by each trainer: the trainer name, the jokemon name, the time the jokemon was caught, and the level and cp of the jokemon, both in integer type.

You are one of the trainers in this game, your **first name** is used in column "name" in Trainer table. All primary keys are underlined.

Specify the following queries using both Relational Algebra and SQL:

1. (12 points) You encountered a Level 10 enemy jokemon that has the "Fire" type. Let's quickly find out the names of all your jokemon that EITHER have a level that is greater than or equal to 10, OR are NOT weak against the enemy's type.

   [**A**nswer:

   Relational Algebra:

   $\Pi_{jokemon\_name}\sigma_{trainer\_name=\text{Litian}\wedge(level\geq 10\vee weak\_to\neq Fire)}(\text{Jokedex} \bowtie \text{HasJokemon})$

   SQL:

   SELECT jokemon_name from Jokedex, HasJokemon WHERE trainer_name=Litian AND (level >= 10 OR weak_to <> Fire) AND Jokedex.jokemon_name=HasJokemon.jokemon_name

   ]

   Rubric:

(a) Relational Algebra (6 points):
   - Join the two correct tables (2 points).
   - Correctly using sigma (1 point).
   - Condition of sigma (2 point).
   - Correctly select column (π) (1 point).

(b) SQL (6 points):
   - SELECT FROM two correct tables (2 points).
   - In where clause:
     - Condition to join the two tables (1 points).
     - Condition for two requirements (2 points).
     - Condition for trainer's name (1 point).

(c) -0.5 for students only forgot to involve his/her trainer's name in.

2. (14 points) Find all the names of your jokemon that can learn skill "Mega Thunderbolt". (Recall that the jokemon has to have the same type as the required type AND has to have level greater than or equal to the required level.)

[**A**nswer:

Relational Algebra:

$\Pi_{jokemon\_name}\sigma_{type=required\_type \wedge level \geq required\_level}$
$(\sigma_{trainer\_name=\text{Litian}}(\text{Jokedex} \bowtie \text{HasJokemon}) \times \sigma_{name=\text{Mega Thunderbolt}}\text{Skill})$

SQL:

Example answer1:

```
SELECT jokemon_name FROM Jokedex, HasJokemon, Skill
WHERE Jokedex.jokemon_name=HasJokemon.jokemon_name
AND trainer_name=Litian
AND type=required_type
AND level >= required_level
AND skill.name="Mega Thunderbolt"
```

Example answer2:

```
SELECT jokemon_name FROM
(SELECT * FROM Jokedex, HasJokemon
WHERE trainer_name=Litian AND Jokedex.jokemon_name=HasJokemon.jokemon_name),
(SELECT * FROM Skill WHERE name="Mega Thunderbolt")
WHERE type=required_type AND level >= required_level
```

]

Rubric:

(a) Relational Algebra (7 points):

- Jokedex JOIN HasJokemon (2 points).
- Using sigma to get "Mega Thunderbolt" in Skill (1 points).
- Join the Skill table in (1 point).
- Condition of sigma for skill learning requirement (2 points).
- Select the correct column (1 point).

(b) SQL (7 points):

- SELECT the correct column (1 point)
- SELECT FROM three correct tables (2 points)
- In where clause:
  - Condition to join the two tables (1 point).
  - Other conditions for skill requirements (3 points)

(c) -0.5 for students only forgot to involve his/her trainer's name in.