CS411: Database Systems **Spring 2016**

Midterm - Solutions

Objective-Type Questions - 20 points

Please **circle** the right answer, and provide a short 1–2 line description justifying your choice.

1. [2] It is possible for weak entity sets to have non-supporting relationships with other entity sets.

   $\boxed{\text{True}}$ False

2. [2] A weak entity set can never be part of a many-to-many relationship.

   True $\boxed{\text{False}}$

3. [2] For a relation $R(A, B)$ with two tuples, at least one among i) $A \to B$ or ii) $B \to A$ must always hold.

   $\boxed{\text{True}}$ False

4. [2] The projection operation is faster in set semantics compared to bag semantics.

   True $\boxed{\text{False}}$

5. [2] The following formula is valid under bag semantics: $(R \cup S) - T = (R - T) \cup (S - T)$. If yes, prove it and if no, provide a counter-example.

   True $\boxed{\text{False}}$

6. [2] If X^+ contains all the attributes of the relation, then X must be a superkey.

   $\boxed{\text{True}}$ False

7. [2] Any table with two attributes must be in BCNF.

   $\boxed{\text{True}}$ False

8. [2] 3NF is dependency-preserving, lossless, and minimizes redundancy when compared to BCNF.

   True $\boxed{\text{False}}$

9. [2] An E-R diagram with m entities and n relationships will always translate to at least

$m + n$ tables in the relational model.

True $\boxed{\text{False}}$

10. [2] An attribute declared as UNIQUE can have NULL as its value.

$\boxed{\text{True}}$ False

Database Design - 20 points

1. [10] You are to design a database for an insurance company. The data will include:

   - Information about customers (SSN, name, address and phone number)
   - Information about insured cars (model, vehicleID and insurance rate)
   - Information about claims made on insured cars (claimID, claim date, settlement date and amount of settlement)
   - An insured car can have multiple insurance claims
   - You may assume that all insured cars are owned by a single customer, but you should allow a customer to own several cars

   Specify an E/R design for your database. Please state any additional assumptions you make in your design. Don't forget to underline key attributes for entity sets and include arrowheads indicating the multiplicity of relationship sets. If there are weak entity sets or "is-a" relationships, make sure to notate them appropriately.

   Answer: in Figure 1.

2. [10] Convert the E/R diagram from question 1 to a relational schema. Merge relations where appropriate. Please show the relational schema before and after the merge, if any. Make sure to underline key/s in each relation.

   Answer:

   Before Merge:

   Customers (<u>SSN</u>, name, address, phone number)
   Insured Cars (<u>vehicleId</u>, model, insurance rate)
   Claims (<u>claimId</u>, claim date, settlement date, settlement amount)
   Owns (<u>SSN</u>, <u>vehicleId</u>)
   Makes (<u>vehicleId</u>, <u>claimId</u>)

   After Merge: Merge many to one relationships

   Customers (<u>SSN</u>, name, address, phone number)
   Insured Cars (<u>vehicleId</u>, model, insurance rate, <u>SSN</u>)
   Claims (<u>claimId</u>, claim date, settlement date, settlement amount, <u>vehicleID</u>)

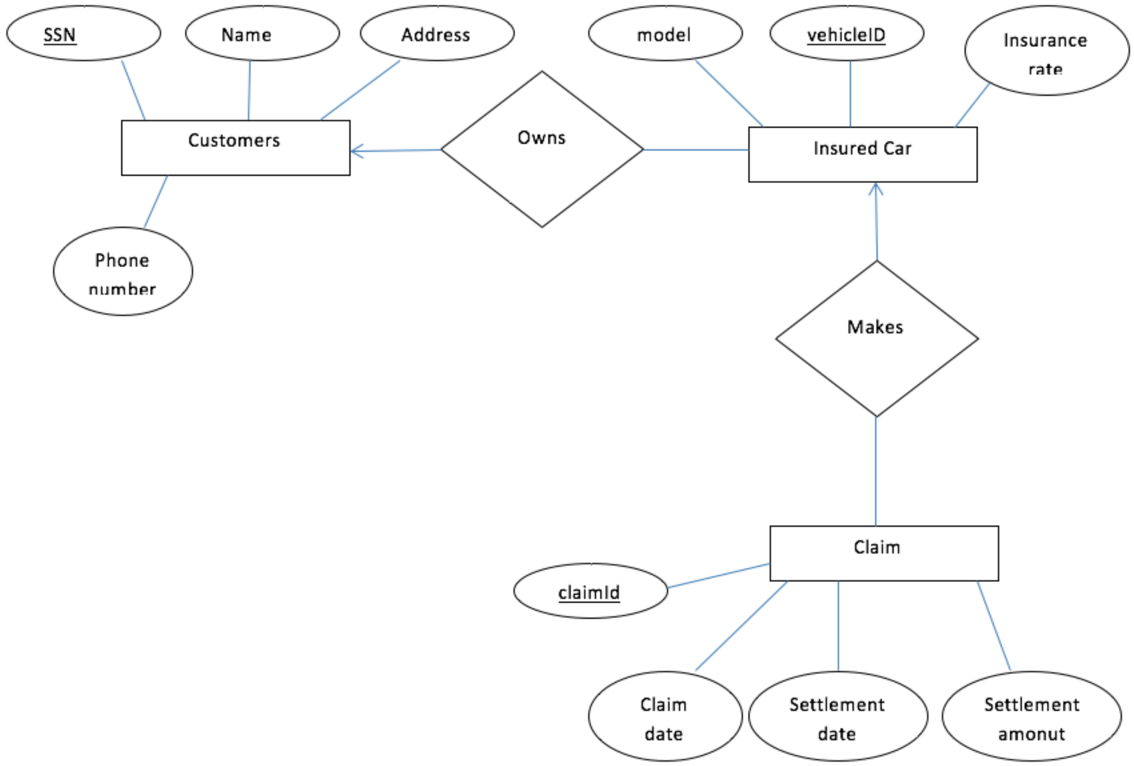

Figure 1: ER Diagram for the insurance company database

Armstrong's Axioms - 10 points

Derive the following rules using Armstrong's axioms:

1. [5] Combining rule: $X \to Y, X \to Z \implies X \to YZ$

   Answer:

$$X \to Y \implies XZ \to YZ \quad \text{(augmentation rule)}$$
$$X \to Z \implies X \to XZ \quad \text{(augmentation rule)}$$
$$\therefore X \to YZ \quad \text{(transitivity rule)}$$

2. [5] Splitting rule: $X \to YZ \implies X \to Y$

   Answer:

$$X \to YZ$$
$$YZ \to Y \quad \text{(reflexivity rule)}$$
$$\therefore X \to Y \quad \text{(transitivity rule)}$$

Functional Dependencies and Normal Forms - 24 points

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

$AB \to C$

$BC \to D$

$CD \to E$

$DE \to A$

1. [6] Specify all minimal keys for R among the following: $\{AB, BC, CD, ED, ABD, BDE, BCE\}$.

   Answer: AB, BC, BDE

2. [4] Which of the given functional dependencies (FDs) are Boyce-Codd Normal Form (BCNF) violations?

   Answer: $CD \to E, DE \to A$

3. [7] Using our algorithm for BCNF decomposition, give a decomposition of R into BCNF based on the FDs.

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

4. [7] Can the algorithm produce a different decomposition into BCNF? If so, provide the corresponding decomposition. Otherwise write "no different decomposition".

   Answer: No different decomposition. Considering $DE \to A$ first results in the same decomposition as above.

Relational Algebra and SQL - 26 points

Consider the following relational database schema:

lives (name, street, city)

works (name, company, salary)

located (company, city)

manages (managee-name, manager-name)

You may make the following assumptions for this problem:

(a) The first attribute is a key for each relation (i.e., lives.name, works.name, located.company, and manages.managee-name are keys).

(b) Every person in the works relation also appears in the lives relation, but not necessarily vice-versa (i.e., no one works without living, but people may live without working).

Specify a relational algebra expression for each of the following queries:

1. [5] Find the names, streets, and cities of all people who work for IBM and earn more than 30,000.

   Answer: $\Pi_{name,street,city}\sigma_{company=“IBM”,salary>30000}(lives \bowtie works)$

2. [5] Find the names of all managers who work in Seattle.

   Answer: $\Pi_{name}\sigma_{city=“Seattle”}(manages \bowtie_{name=manager-name} (located \bowtie works))$

Specify the following queries using SQL:

1. [8] Find the names of all people who work for IBM, and do not manage anyone.

   Answer:

```
(SELECT name FROM works WHERE company = "IBM")
 EXCEPT
(SELECT name FROM works, manages
 WHERE name = manager-name AND company = "IBM")
```

2. [8] Find the names of all people who live and work in different cities.

   Answer:

```
SELECT name
FROM lives, works, located
WHERE works.name = lives.name
AND works.company = located.company
AND located.city <> lives.city
```