Virginia Tech. **CS 4604 – Introduction to DBMS**
Computer Science **Spring 2014, Prakash**

Homework 6: FDs and NFs

(due April 15, 2014, 3:30pm, hard-copy in-class please)

Reminders:

a. Out of 100 points. Contains 4 pages.
b. Rough time-estimates: ~2-4 hours.
c. Please type your answers. Illegible handwriting may get no points, at the discretion of the grader. Only drawings may be hand-drawn, as long as they are neat and legible.
d. There could be more than one correct answer. We shall accept them all.
e. Whenever you are making an assumption, please state is clearly.
f. Lead TA for this homework: Qianzhou Du.

Q1. FDs Definition [10 points]

Consider the situation in Exercise 19.6 in your textbook. We repeat it here for your convenience. Suppose that we have the following three tuples in a legal *instance* of relation schema S with three attributes ABC:

A	B	C
1	2	3
4	2	3
5	3	3

Q1.1 (5 points) Which of the following dependencies can you infer does not hold over schema S?

(a) $A \rightarrow B$, (b) $BC \rightarrow A$, (c) $B \rightarrow C$

Q1.2 (5 points) Can you identify any functional dependencies that *may* hold over S?

Q2. Inferring FDs [27 points]

Consider the following relations and sets of functional dependencies that hold in those relations. For each sub-part, it is enough for you to list only completely non-trivial FDs with a single attribute on the right hand side.

Note that 'candidate key' means just 'key' (i.e. both words are interchangeable). A candidate key (or simply key) should imply the entire relation and should be minimal. On the other hand, a 'superkey' is any super-set of a candidate key.

Q2.1 R1(A, B, C, D) with FDs $A \rightarrow B$, $B \rightarrow C$, and $C \rightarrow D$.

Q2.1.1 (3 points) What are all the non-trivial FDs that follow from the given FDs?

Q2.1.2 (3 points) What are all the keys (i.e. candidate keys) of R1?

Q2.1.3 (3 points) What are all the superkeys for R1?

Q2.2 R2(A, B, C, D) with FDs AB → C, C → A, and C → D.

Q2.2.1 (3 points) What are all the non-trivial FDs that follow from the given FDs?

Q2.2.2 (3 points) What are all the keys (i.e candidate keys) of R2?

Q2.2.3 (3 points) What are all the superkeys for R2?

Q2.3 R3(A, B, C, D, E) with FDs A → B, B → D, BC → E, and ED → A.

Q2.3.1 (3 points) What are all the non-trivial FDs that follow from the given FDs?

Q2.3.2 (3 points) What are all the keys (i.e. candidate keys) of R3?

Q2.3.3 (3 points) What are all the superkeys for R3?

Q3. Projection of FDs [22 points]

Consider the situation in Exercise 19.8 in your textbook. We repeat it here for your convenience. Consider the relation R = ABCDEFGH and the FD set F = {AB → C, AC → B, AD → E, B → D, BC → A, E → G}.

Q3.1 (4x3=12 points) Compute the set of FDs that hold over each attribute set (think of each attribute set as a relation derived from R and that you need to project F onto each). Only write down a minimal cover.

A) Attribute set = ABCD
B) Attribute set = ABCEG
C) Attribute set = DCEGH
D) Attribute set = ACEH

Q3.2 (10 points) Which of the following decompositions of R = ABCDEG, with the same set of dependencies F above, is (a) dependency-preserving? (b) lossless-join? Show your steps.

A) (5 points) {ABCDE, EG}

B) (5 points) {ABC, ACDE, ADG}
Hint: First figure out if {ABCDE, ADG} is a lossless and dependency preserving decomposition of R. Then check if {ABC, ACDE} is a lossless and dependency preserving decomposition of R1 = ABCDE.

Q4. 3NF Decomposition [15 points]

Consider a relation **R**(A, B, C, D, E, F) and let the set of FDs for **R** be: A → DF, DE → B, DE → C, F → B, A → B, and B → C.

Q4.1 (3 points) List all the candidate keys (i.e. keys) for R.

Q4.2 (10 points) Is R in 3NF? If not, use the 3NF synthesis algorithm to find a lossless-join, dependency-preserving decomposition of R into a set of 3NF relations. Show your steps.

Q4.3 (2 points) Are any of the relations (that you get in Q4.2 above) not in BCNF?

Q5. FDs in English language [26 points]

Consider the following table:
flightOrderDB (CustomerID, CustomerName, Age, Gender, Address, OrderTime, FlightNO, From, To, Price, Departs, Arrives, Distance, AircraftID, Manufacturer, Model)

Now consider the following constraints in English:

Statement 1 Every customer has a unique customer id, but multiple customers may have the same names, ages, gender, and address. At a particular time, distance, one customer can order only one flight.

Statement 2 Every flight has a from city, a to city, a flight price, departure time, arrival time, flight distance, and a unique flight number.

Statement 3 Every aircraft has a manufacturer, model type, and a unique aircraft id. And each model type belongs to only one manufacturer.

Statement 4 Every flight uses a unique aircraft.

Statement 5 Every aircraft has a manufacturer, model type, and a unique aircraft id. And each model type belongs to only one manufacturer.

Statement 6 Between the same city pair of A and city B, the flight distance should the same and there are probably **multiple** different optional flights existing. For the same departure time, distance, the flights **must** have the same price. For example, all flights, which departs at

6:00 pm and flight distance is 700 miles, should have the same price.

Q5.1 (2 points) Is this a good relational design or not? Explain your answer, without going into any of the normal forms, just list the different types of anomalies this relation has.

Q5.2 (7 points) List the functional dependencies for this relation that you can construct using the English description and also specifically mention the corresponding statement number(s) you used to formulate each FD. (Note: a minimal cover is enough and some 'Statement's may give rise to multiple FDs).
Hint: You should get 7 FDs in total.

Q5.3 (4 points) Rigorously prove (using Armstrong's axioms) that you can derive the FD: From To Departs → Price from the set of the FDs you found in Q5.2. Show your steps.

Q5.4 (9 points) Is the above relation in BCNF using the FDs you found in Q5.2? If not, decompose it into a set of BCNF relations.

Q5.5 (4 points) Is your result of decomposition in Q5.4 dependency-preserving? If it is not dependency-preserving, explain your answer, and then decompose the original relation into a set of 3NF relations instead. Show your steps.