**CSC 355 Database Systems**

**Fall Quarter 2018**

**Assignment 4 (10/3)**

**Due Sunday, October 14th @ 11:59pm.**

**Reading:** Chapter 15 ( Sec 1: Database Design Rules; Sec 2: Functional Dependencies; Sec 3 & 4: 1NF, 2NF, and 3NF; Sec 5: BCNF; (skip 4NF and 5NF),

Chapter 16 (Sec 1: Canonical Cover, Sec 2 & 3: Testing for Lossless Join Property and Dependency Preservation),

Chapter 21: Serializability and Transactions

Please note that there is a total of 6 questions.

**1. Canonical Cover Algorithm (11 pts).**

Given R(ABCD),

A. Find a canonical cover for the following set F of functional dependencies.

A🡪BC

AB🡪D

C🡪AD

D🡪B

A 🡪 BCD

C 🡪 AD

D 🡪 B

B. Find all candidate keys.

Candidate Key: A

We can determine the whole relation ABCD.

**2. Dependency Preservation (15 pts).**

For the universal relation R(w,x,y,z), consider the decomposition D consisting of R1(w,y,z) and R2(x,y), and the set F of functional dependencies { y🡪xz ; yz🡪w ; x🡪w }.

Recall that the projection of set of functional dependences G on relation Rx consists of every functional dependency in (G)+ that contains only attributes from Rx.

**a.** Compute the projection of F on R1.

YZ 🡪 W

Y 🡪 Z (because Y🡪XZ)

**b.** Compute the projection of F on R2.

Y 🡪 X

**c.** Does the decomposition D preserve the set of dependencies F? Why or why not?

No because the union of both projections is missing the dependence X 🡪 W

**3. Nonadditive (Lossless) Join Property (24 pts).**

R = {ABCDEFGHIJ}

F = { AB -> C

A -> DE

B -> F

F -> GH

D -> IJ

}

Determine whether each decomposition into R1 and R2 has the lossless join property. Be sure to state why or why not.

**a.** R1(ABC) and R2(ABDEFGHIJ)

Lossless. All dependencies were preserved and no new were added.

**b.** R1(ADE) and R2(EBCFGHIJ)

Not lossless. Lost:

AB 🡪 C

D 🡪 IJ

**c.** R1(BFDIJ) and R2(ABCDEFGH)

Lossless. All dependencies were preserved and no new were added.

**d.** R1(BDFGHIJ) and R2(ABCDE)

Lossless. All dependencies were preserved and no new were added.

**4. Normalization and Decomposition (20 pts).**

R(ABCDEG)

FDs {AB=>C, AC=>B, AD=>E, B=>D, BC=>A, E=>G}

**a.** Identify candidate keys of R. Hint: There are three candidate keys.

AB

AC

BC

**b.** Construct a decomposition of R into relations in 2NF.

R1(A B C G)

R2(B D)

R3(A D E)

**c.** Construct a decomposition of R into relations in 3NF.

R1(A B C G)

R2(B D)

R3(A D E)

R4(D E G)

**d.** Are all of the relations in your decomposition in BCNF? Either explain why they are, or identify one that is not and explain why it is not. (Note that for a relation to be in BCNF, the determinants of all functional dependencies in the relation must be superkeys *of that relation* – not superkeys of the original universal relation.)

No because Relation3 under 3NF contains D as a determinant, but D is not a superkey.

**5. Transactions and Serialization (15 pts).**

Consider the three transactions T1, T2, and T3, and the schedules S1 and S2 given below. Draw the serializability (precedence) graphs for S1 and S2, and state whether each schedule is serializable or not. If a schedule is serializable, write down the equivalent serial schedule(s).

T1: R1(W), R1(Z), W1(X)

T2: R2(Z), R2(Y), W2(Z), W2(Y)

T3: R3(X), R3(Y), W3(Y)

S1: R1(X), R2(Z), R1(Z), R3(X), R3(Y), W1(X), W3(Y), R2(Y), W2(Z), W2(Y)

S2: R1(X), R2(Z), R3(X), R1(Z), R2(Y), R3(Y), W1(X), W2(Z), W3(Y), W2(Y)

S1:

|  |  |  |
| --- | --- | --- |
| R1(X) |  |  |
|  | R2(Z) |  |
| R1(Z) |  |  |
|  |  | R3(X) |
|  |  | R3(Y) |
| W1(X) |  |  |
|  |  | W3(Y) |
|  | R2(Y) |  |
|  | W2(Z) |  |
|  | W2(Y) |  |

Equivalent S1: T1, T3, T2

S2 :

|  |  |  |
| --- | --- | --- |
| R1(X) |  |  |
|  | R2(Z) |  |
|  |  | R3(X) |
| R1(Z) |  |  |
|  | R2(Y) |  |
|  |  | R3(Y) |
| W1(X) |  |  |
|  | W2(Z) |  |
|  |  | W3(Y) |
|  | W2(Y) |  |

No equivalent schedule. Cycle exists between T1 🡪 T3 🡪 T2 🡪 T1

**6. Regular Expressions/SQL (15 pts).**

Write SQL for the following problems on the university database.

**A.** Find students whose name is properly capitalized (both first and last name start from an upper-case letter, rest of the name is lower-case). Hint: you can use [A-Z] to represent any upper case letter in regular expressions.

SELECT \* FROM RegEx

WHERE REGEXP\_LIKE( TEXT, ‘^[A-Z][a-z]+’ );

**B.** Find course titles that consist of exactly 3 words (e.g., “Theory of Computation”).

SELECT \* FROM RegEx

WHERE REGEXP LIKE (TEXT, ‘ [\w\s] {3}’);

**C.** Find students whose SSN numbers have a pattern of 321 repeated twice (sequentially).

SELECT \* FROM RegEx

WHERE REGEXP LIKE (TEXT, ‘ [321]{2}’);