****

**Group Project 4**

**Alex Salazar-Almaraz, Jordan Nguyen, Frederick Smeltzer, Danny Ta**

****

CS/DSA-4513 – Database management systems

instructor:

Dr. Le Gruenwald

**Relational Schema:**

R (A, B, C, D, E, F, G)

**Functional Dependencies:**

SetOfFDs = {A -> B, B -> AC, F -> ACDE, ADE -> FG}

**Problem 1:**

1. Candidate keys (Highlighted):

A+ = ABC

B+ = ABC

C+ = C

D+ = D

E+ = E

F+ = ABCDEFG

G+ = G

AB+ = ABC

AC+ = ACB

AD+ = ADBC

AE+ = AEBC

BD+ = BDAC

BE+ = BEAC

DE+ = DE

ABD+ = ABDC

ABE+ = ABEC

BDE+ = ABCDEFG

ADE+ = ABCDEFG

ABDE+ = ABCDEFG

Our non-prime attributes are those that are not present in our candidate keys. The only two not present are C and G.

1. Normal Forms:
   1. 1NF – Each of the attributes is atomic, meaning that each is not divisible, so it will be in 1NF and is satisfied by the functional dependencies.
   2. 2NF – To be in second normal form, it must already be in 1NF but also must have no non-prime attributes with partial dependencies on the primary key.

Candidate keys – F, BDE, ADE

Non-prime attributes – C, G

G:

G is fully dependent on each of the candidate keys. We don’t see G in any subset of the candidate keys

BDE+ = ABCDEFG

B+ = ABC

D+ = D

E+ = E

BD+ = BDAC

BE+ = BEAC

ADE+ = ABCDEFG

A+ = ABC

D+ = D

E+ = E

AD+ = ADBC

AE+ = AEBC

F+ = ABCDEFG

C:

C is partially dependent, thus breaking the 2NF. We can see that subsets of each of the candidate keys consist of the attribute C

BDE+ = ABCDEFG

B+ = ABC

D+ = D

E+ = E

BD+ = BDAC

BE+ = BEAC

ADE+ = ABCDEFG

A+ = ABC

D+ = D

E+ = E

AD+ = ADBC

AE+ = AEBC

* 1. 3NF – To be in third normal form, it must already be in 2NF but also must have no attributes with transitive dependencies on the primary key. This is not satisfied because we do not have 2NF.
  2. BNCF – To be in Boyce-Codd Normal Form it must be in 3NF but has stricter terms. For any non-trivial functional dependency, X -> A, X must be a super-key. This is not satisfied because we do not have 2NF or 3NF.

1. Decomposition Algorithm:

R (A, B, C, D, E, F, G)

F = {A -> B,

B -> AC,

F -> ACDE,

ADE -> FG}

ADE and F are candidate keys. We need to decompose A and B where

Result := (result- Ri) U (Ri-B) U (a, B)

Result = {R}

R1 = (ABDEFG)

R2 = (A,C)

R = { } U { R1 = (ABDEFG)} U { R2 = (A,C)}

We have R1 and R2

We now need to check to see if any subset of R1 or R2 violates the 2NF. This algorithm guarantees that R1 is in 2NF, but what about R2? Because we know that none of our candidate keys changed and that G is fully functionally dependent, there are no other violations to 2NF.

1. Lossless:

Using the decomposition algorithm, we know that we are guaranteed a lossless join. If one of the following dependencies is in F+, R1 ∩ R2 → R1 or R1 ∩ R2 → R2 then we know that there was a lossless join. We can see that this is true by combining the attributes from both R’s: (ABDEFG) + (AC) = R

1. Dependency:

Using the decomposition algorithm, we are not guaranteed dependency, so we must check to see if it has been preserved.

R1 = (ABDEFG)

R2 = (AC)

R1 ∩ R2 = {A} and A -> AB in F+, i.e., R1 ∩ R2 -> R1 in F+

This is not dependency preserving as we cannot check:

B -> AC

F -> ACDE

Without computing R1 |x| R2

**Problem 2:**

**Problem 3:**

A.

Factors to consider in gathering data from the web and social media include verification of what is often error-prone data. It is necessary to know if there are multiple sources and accounts on a particular piece of information before one can deem that it is, in fact, true. Additionally, the person or organization we obtain the data from is very important -- in the face of social media, one is more likely to get accurate information from BBC News' Facebook page over Sean Murray's Twitter feed. BBC News generally has a better reputation over anyone's posts that are limited to 120 characters. While BBC News cannot be the only page to see on the field of social media, it is thus imperative that fabricated information is identified.

Because the CDC is a well-reputed government facility, users will easily trust the information given out by the CDC and CDC-related software, regardless of their age and background. If misinformation began spreading to these users because the database system picked up false or fabricated information about Zika, the health of these users (mostly unborn children in the wombs of pregnant users) could potentially be at risk (and die in childbirth). The media would be quick to cover at how wrong information was sent by the CDC, and CDC's reputation would receive a deadly blow, resulting in a change of employees or even disbanding entirely so that the government can save face. As for society in general, they would begin a distrust towards the government and the lies it has begun spreading, if these people haven't already been crippled by their incapability of giving birth to children with normal-sized heads.

B.

Suppose we began with an initial table, VictimLocation(vicname, country, numSymptoms, isdead) where vicname is the victim's name, country is the location where the victim resided at, and numSymptoms is the number of symptoms the victim had over the course of being infected with zika. We can split the table into two, namely, VictimLocationB(vicname, country, isdead) and CountrySymptom(country, numSymptoms). The first table is suffixed with a B to differentiate it from the original table (though, it is not necessary in practice).

An example of the contents would be as follows (in Markdown format):

**#### VictimLocation**

**| vicname | country | numSymptoms | isdead |**

**|=========|=========|=============|========|**

**| George | France | 5 | True |**

**| Bob | India | 3 | True |**

**| Sarah | UK | 1 | True |**

**| Nerven | Asgard | 9 | False |**

**| Vicki | USA | 4 | False |**

**| Vestus | Kenya | 2 | False |**

**#### VictimLocationB**

**| vicname | country | isdead |**

**|=========|=========|========|**

**| George | France | True |**

**| Bob | India | True |**

**| Sarah | USA | True |**

**| Nerven | Asgard | False |**

**| Vicki | USA | False |**

**| Vestus | Kenya | False |**

**#### CountrySymptom**

**| country | numSymptoms |**

**|=========|=============|**

**| France | 5 |**

**| India | 3 |**

**| USA | 1 |**

**| Asgard | 9 |**

**| USA | 4 |**

**| Kenya | 2 |**

With these tables and their lossy join decomposition, we are forced to lose the original data from the first table. When using natural join on these two latter tables, we would get misleading information. The following table results from a natural join.

**| vicname | country | numSymptoms | isdead |**

**|=========|=========|=============|========|**

**| George | France | 5 | True |**

**| Bob | India | 3 | True |**

**| Sarah | USA | 1 | True |**

**| Nerven | Asgard | 9 | False |**

**| Sarah | USA | 4 | True |**

**| Vicki | USA | 1 | False |**

**| Vestus | Kenya | 2 | False |**

**| Vicki | USA | 4 | False |**

From the natural join, we appear to have extra tuples -- and these extra tuples have the wrong information. How can a single person have two different numbers of symptoms? How truly odd.

When users see this database, they will mostly be confused, but they can easily point out a flaw. A lossy decomposition does not directly affect users around the world, but they will become skeptical of the integrity of CDC's systems. The CDC, however, is affected more, because now they have no idea if Vicki has either 1 or 4 symptoms -- and if it will lead to a correlation in being dead in a certain country like USA. This indirectly affects users as the CDC will be unable to come up with a proper statement of where countries prone to zika might be found. In influencing society, people will be uncertain of where the danger might be, and the world then risks infecting millions of people simply because of a mistake made by the CDC.