Last Name: wang First Name: yiting Student ID: 13853462

**1. [20 pts]**

Suppose a coworker (who sadly never took CS122A) came up with their own design for the Tweet table (shown below). The table has tweet_id, tweeter_id, and tweet_text. Your coworker proposed that the primary key for the table Tweet is both tweet_id and tweeter_id. Your boss asked you to review and revise the design by providing answers to the following questions.

```
CREATE TABLE Tweet (
    tweet_id VARCHAR(20),
    tweeter_id VARCHAR(20) NOT NULL,
    tweet_text VARCHAR(300) NOT NULL,
    PRIMARY KEY (tweet_id, tweeter_id),
);
```

(a) [5 pts] What non-trivial functional dependencies does your coworker's Tweet table have, if any? List them all:

`(tweet_id, tweeter_id)→ tweet_text`

(b) [5 pts] What normal form is your coworker's Tweet table currently in? Briefly show your reasoning.

BCNF. Since everything depend on the key, the whole key, and nothing but the key.

(c) [5 pts] Given your coworker's Tweet table, list **all** 1) superkey(s), 2) prime attribute(s), and 3) non-prime attribute(s). If there exists one or more superkey(s), explain why they are superkeys.

1. Superkey: `(tweet_id, tweeter_id) (tweet_id, tweeter_id, tweet_text) because they contain the candidate key`

2. prime: tweet_id, tweeter_id

3. non-prime: tweet_text

(d) [5 pts] Given what you know about tweets, from previous homeworks in this class, what change(s) would you propose for your coworker's design? List all the functional dependencies of your revised design (or again of your coworker's design if you recommended no changes). *(Note: Please stick to just adjusting your coworker's current design, even though it might be missing a few fields and foreign keys that you would have expected from past homeworks. You should* ***NOT*** *add those to the revised design here).*

CREATE TABLE tweet_id (

tweet_id VARCHAR(20),

tweet_text VARCHAR(300) NOT NULL,

PRIMARY KEY (tweet_id),

);

**2. [20 pts]**

Your CheckedTweets.org business analysts have a copy of the database that gets refreshed periodically. Suppose that they have created a new Tweet table called TweetAlpha, containing a new field: hash_tag_alpha. This represents the result of some mythical deterministic function that accepts a Tweet's hashtags and outputs some numeric value. Additionally, you can safely assume that all hashtags found for a Tweet will exist in the Tweet text itself.

```
CREATE TABLE TweetAlpha (
    tweet_id VARCHAR(20),
    tweet_text VARCHAR(300) NOT NULL,
    hash_tag_alpha INTEGER NOT NULL,
    PRIMARY KEY (tweet_id),
    FOREIGN KEY tweet_id REFERENCES RawTweet (tweet_id)
);
```

(a) [5 pts] What non-trivial functional dependencies does this modified Tweet table have, if any? List them here:

tweet_id → tweet_text

tweet_text→ hash_tag_alpha

(b) [5 pts] Does this current table satisfy 2NF [Yes/No]? Give a short reasoning as to why (<= 2 sentences).

yes, since there's no non-prime attribute has partial dependency to the candidate key.

(c) [5 pts] Does this current table satisfy 3NF [Yes/No]? Give a short reasoning as to why (<= 2 sentences).

no, since there is transitive dependency to non-prime attribute.

(d) [5 pts] Decompose TweetAlpha into multiple tables to produce a BCNF design if the current design isn't already there. If the design is already in BCNF, write "no change needed".

Need change

TweetAlpha(tweet_id, tweet_text) candidate key: tweet_id

TweetAlpha(tweet_text, hash_tag_alpha) candidate key: tweet_text

**3. [20 pts]**
Consider the following relation:

| H | J | G |
|---|---|---|
| h_2 | j_2 | g_1 |
| h_5 | j_5 | g_9 |
| h_7 | j_5 | g_8 |
| h_0 | j_5 | g_5 |
| h_0 | j_2 | g_5 |
| h_8 | j_5 | g_2 |

(a) [12 pts] Given the current state of the database, for each one of the following functional dependencies answer a) Does this functional dependency hold in the above relation instance [Yes/No]? b) If your answer to the previous question was no, explain why by listing a tuple that causes a violation.

i. G → H : yes

ii. H → J : NO (h_0, j_5, g_5),(h_0, j_2, g_5)

*iii.* J → H: No, (h_2, j_2, g_1), (h_0, j_2, g_5)

*iv.* G → J: No, (h_0, j_5, g_5),(h_0, j_2, g_5)

(b) [3 pts] List all ***potential*** candidate keys (if there are any) for the above relation.

GJ

(c) [3 pts] List all ***definite*** candidate keys (if there are any) for the above relation.

GJ

Normalizing a schema with a set of FDs can be done by hand or automatically by computers. Complete questions 4 and 5 with the help of the normalization tool provided by Griffith University - *but try each part by hand first*! Use the problems to cement your understanding of FDs and normal forms and then use the tool to check your answers.

**4. [20 pts]**
R(A, B, C, D, E, F)
(All attributes contain only atomic values.)
FD1: A → BCD
FD2: A → F
FD3: BC → E
FD4: D → F

(a) [5 pts] Compute A+, the attribute closure of attribute A. Show your work as well as the final result.

1. A→BC(FD1, decomposition)
2. A→E(1, FD3, through transitivity)
3. A→ABCDEF(FD1,FD2,2, union)
   so A+={ABCDEF}

(b) [5 pts] List the candidate keys and the minimal cover of R.

candidate key: A

minimal cover of R: A→B, A→C, A→D, BC→ E, D→F

(c) [5 pts] What's the highest normal form that R satisfies and why?
2NF. Since it violates 3NF because it is non-trivial, whose LHS is not superkey, and RHS contains a non-key attribute.

(d) [5 pts] If R is not already at least in 3NF, then normalize R into 3NF and show the resulting relation(s) ***and*** specify their candidate keys. Make sure that your 3NF decomposition is both lossless-join and dependency-preserving. Note: If R was already in 3NF, then just list the candidate keys of R. What is the highest normal form that your answer now satisfies?

R0(A,B,C,D) candidate key: A

R1(B,C,E). candidate key: (B,C)

R2(D,F). candidate key: D
BCNF

**5. [20 pts]**
R(A, B, C, D, E, F, G)
(All attributes contain only atomic values.)
FD1: B → G
FD2: AB → AFD
FD3: BC → E
FD4: G → C

(a) [5 pts] Compute B+, the attribute closure of attribute B. Show your work and final result.
1.B→C(FD1, FD4, transitivity)

2. B→BG (FD1)

3. BG→BC(FD4, argumentation)

4. B→BC(2,3, transitivity)

5. B→E(4,FD3, transitivity)

6. B→CGE(1,FD1,5, union )
so $B^{+}$={BCGE}

(b) [5 pts] What is the minimal cover for the given set of FDs?
B→G

AB→F

AB→D

B→E

G→C

(c) [5 pts] Normalize R into BCNF and show the resulting relation(s) and their candidate keys.
R0(A,B,D,F) candidate: AB
R1(G,C) candidate: G

R2(B,G,E) candidate: B

(d) [5 pts] Is the decomposition in part (c) dependency-preserving? Why or why not? (Be specific when answering, referring to the initial functional dependencies by name, i.e., FD1-FD4, as needed.)

Yes, The closure of the set of all FDs hold on the decomposed relations(FD1,2,4) is the same as the closure of the original FDs {FD1, FD2, FD4}+={FD1, FD2, FD3, FD4}+