University of Toronto Mississauga
CSC 343 Fall 2019
Group Assignment 3 – **SOLUTIONS**
Due: Wednesday December 4th, 2019 by 11:59pm

Michael Liut

December 8, 2019

I. Database Design (25 marks)

Question 1 (10 marks)

From the `Schedule` relation, you are required to represent the following additional information:

a) No professor can be assigned to teach two (or more) courses on the same day and time.
ANSWER: professor,day,time → course

b) There is at most one teaching assistant per course.
ANSWER: course → teaching assistant

c) No two (or more) courses can be assigned to the same location on the same day and time.
ANSWER: location,day,time → course

d) No two (or more) professors can be assigned to the same location on the same day and time.
ANSWER: location,day,time → professor

e) The combination of course, day, and time will uniquely determine what professor is teaching.
ANSWER: course,day,time → professor

Answer the following questions:

1. Given the additional information, `a` – `e`, list all of the functional dependencies that can be inferred. [5 marks]

   ANSWERS ARE INLINE ABOVE

2. Dr. Strange will only be satisfied if your design is a good one (i.e. the schema satisfies either the 3NF or the BCNF). Is the design of your schema with the functional dependencies from part (1), above, a good one? Justify your answer. If the design is not a good

one, provide a better one, using one of the decomposition algorithms discussed in class. [5 marks]

ANSWER: This is a BCNF Decomposition. The relation Schedule is not currently in BCNF as course → teaching assistant directly violates BCNF. The decomposition should look like: $R_1(teaching\, assistant, course)$ and $R_2(course, professor, location, day, time)$ along with the projected functional dependencies.
Please Note: Students are required to justify their answer (i.e. WHY is it in BCNF). **2 marks of 5** are allocated to justification.

Question 2 (5 marks)

1. List all of the functional dependencies that this Relation Table satisfies. [3 marks]

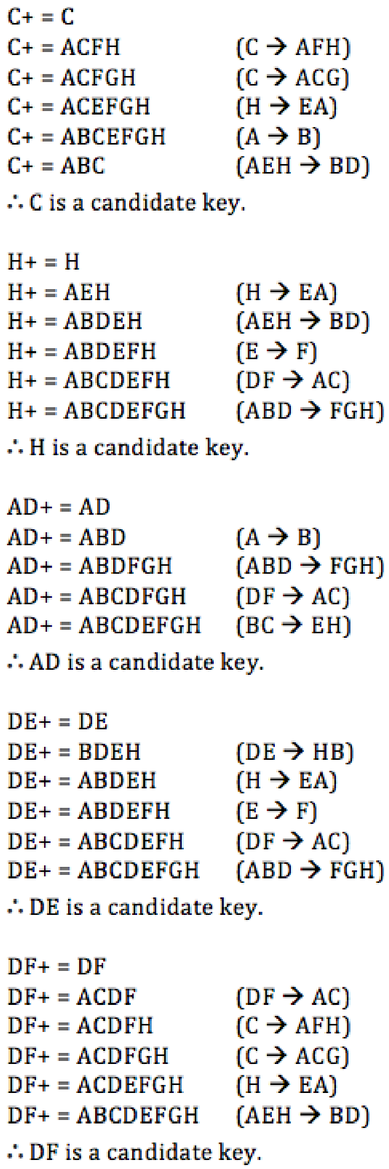
   ANSWER: $C \to B$, $A \to B$, and $AC \to B$.

2. Now let's modify attribute **C**'s last record from c_3 to c_2. What functional dependencies, if any, from question (2) part (1), above, have been changed? List them. Explain your reasoning in no more than two sentences. [2 marks]

   ANSWER: The FDs are identical to above. The set is unchanged.

Question 3 (10 marks)

(a) Candidate Keys: `{{C}, {H}, {A, D}, {D, E}, {D, F}}`. We can prove this by looking at the closure of each key:



C+ = C
C+ = ACFH (C → AFH)
C+ = ACFGH (C → ACG)
C+ = ACEFGH (H → EA)
C+ = ABCEFGH (A → B)
C+ = ABC (AEH → BD)
∴ C is a candidate key.

H+ = H
H+ = AEH (H → EA)
H+ = ABDEH (AEH → BD)
H+ = ABDEFH (E → F)
H+ = ABCDEFH (DF → AC)
H+ = ABCDEFGH (ABD → FGH)
∴ H is a candidate key.

AD+ = AD
AD+ = ABD (A → B)
AD+ = ABDFGH (ABD → FGH)
AD+ = ABCDFGH (DF → AC)
AD+ = ABCDEFGH (BC → EH)
∴ AD is a candidate key.

DE+ = DE
DE+ = BDEH (DE → HB)
DE+ = ABDEH (H → EA)
DE+ = ABDEFH (E → F)
DE+ = ABCDEFH (DF → AC)
DE+ = ABCDEFGH (ABD → FGH)
∴ DE is a candidate key.

DF+ = DF
DF+ = ACDF (DF → AC)
DF+ = ACDFH (C → AFH)
DF+ = ACDFGH (C → ACG)
DF+ = ACDEFGH (H → EA)
DF+ = ABCDEFGH (AEH → BD)
∴ DF is a candidate key.

Marking: **1 mark** per key = 5 marks, **2 marks** for correct use of axioms. If only the keys are found without use of axioms, you may award **0.5 marks** per key (max 2.5/7).

(b) i. FALSE – 3NF decomp is different from closure.
ii. FALSE – there are two 3NF decomps for this example

Marking: **1 mark** for correct FALSE response and **2 marks** for a sufficient justification/proof.

Given Functional Dependencies:

DF → AC AEH → BD C → AFH C → ACG E → F
BC → EH A → B DE → HB ABD → FGH H → EA

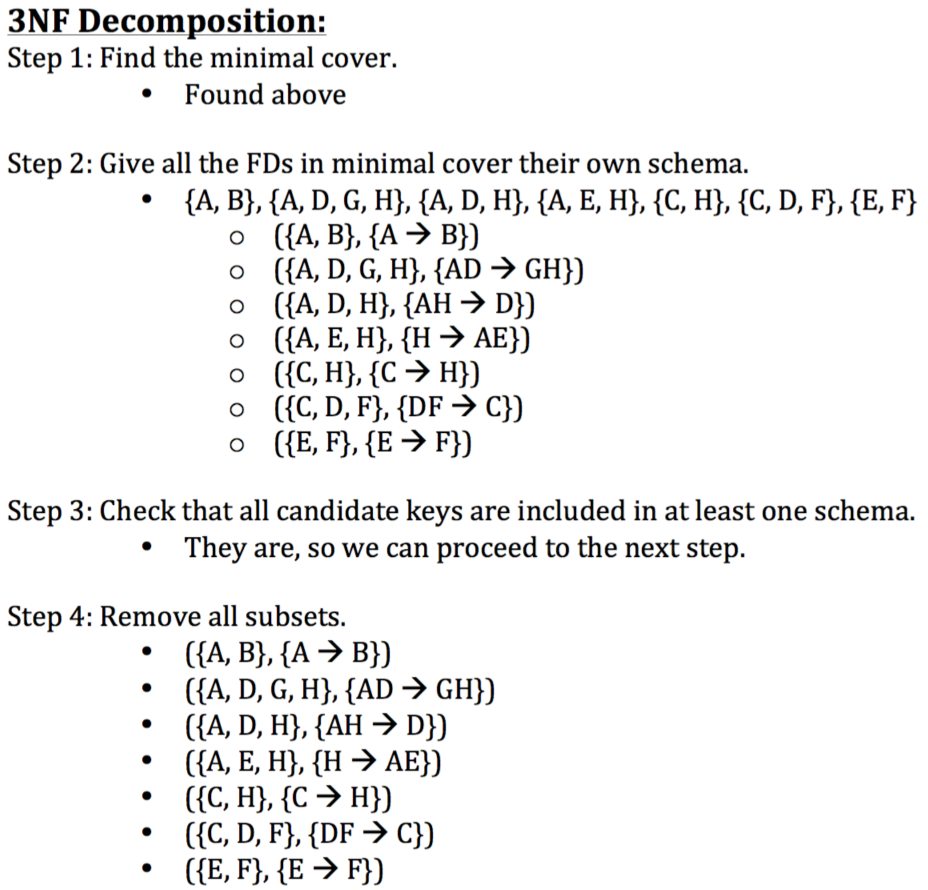
Redundant	**Minimal Cover**	**Minimal Redundant**
DF → C	DF → C	AEH → D (E is redundant)
AEH → B	E → F	ABD → G (B is redundant)
		ABD → H (B is redundant)
C → A	A → B	
C → F		
C → H	H → A	
	H → E	
C → A		
C → C		
C → G		
BC → E		
BC → H		
DE → B		
DE → H		
ABD → F		

Eliminate Redundant FDs	**Eliminate Unnecessary Attributes**
DF → A **(Redundant)**	AEH → BD
DF → DF (Reflexivity)	AEH+ = {ABEFH}
DF → C (DF → C)	
DF → A (C → A)	AH → BD **(Min Cover)**
	AH → AH (Reflexivity)
AH → B **(Redundant)**	AH → ABH (A → B)
AH → AH (Reflexivity)	AH → ABEH (H → E)
AH → AEH (H → E)	AH → ABEFH (E → F)
AH → DE (AH → D)	AH+ = {ABEFH}
AH → HB (DE → HB)	∴ E can be eliminated
AH → B (Decomposition)	
	ABD → GH (B is redundant so remove it)
C → A **(Redundant)**	AD → GH **(Min Cover)**
C → C (Reflexivity)	AD → AD (Reflexivity)
C → H (C → H)	AD → ABD (A → B)
C → A (Transitivity: H → A)	AD → FGH (ABD → FGH)
	∴ B can be eliminated as we reached GH without the aid of B
C → F **(Redundant)**	
C → C (Reflexivity)	

C → E (C → E) C → F (Transitivity: E → F) C → G **(Redundant)** C → C (Reflexivity) C → H (C → H) C → AEH (Transitivity: H → A, H → E) C → ABD (AEH → B, AEH → D) C → G (ABD → G) BC → E **(Redundant)** BC → BC (Reflexivity) BC → H (BC → H) BC → E (Transitivity: H → E) BC → H **(Redundant)** BC → BC (Reflexivity) BC → BH (Transitivity: C → H) BC → H (Augmentation) DE → B **(Redundant)** DE → DE (Reflexivity) DE → H (DE → H) DE → A (H → A) DE → B (A → B) DE → H **(Redundant)** DE → DE (Reflexivity) DE → DF (E → F) DE → C (DF → C) DE → H (C → H) AD → F **(Redundant)** AD → AD (Reflexivity) AD → E (AD → E) AD → F (Transitivity: E → F)	

∴ the minimal cover are:

A → B   AD → GH   AH → D   C → H
DF → C   E → F   H → AE



3NF Decomposition:

Step 1: Find the minimal cover.

- Found above

Step 2: Give all the FDs in minimal cover their own schema.

- {A, B}, {A, D, G, H}, {A, D, H}, {A, E, H}, {C, H}, {C, D, F}, {E, F}
    - ({A, B}, {A → B})
    - ({A, D, G, H}, {AD → GH})
    - ({A, D, H}, {AH → D})
    - ({A, E, H}, {H → AE})
    - ({C, H}, {C → H})
    - ({C, D, F}, {DF → C})
    - ({E, F}, {E → F})

Step 3: Check that all candidate keys are included in at least one schema.

- They are, so we can proceed to the next step.

Step 4: Remove all subsets.

- ({A, B}, {A → B})
- ({A, D, G, H}, {AD → GH})
- ({A, D, H}, {AH → D})
- ({A, E, H}, {H → AE})
- ({C, H}, {C → H})
- ({C, D, F}, {DF → C})
- ({E, F}, {E → F})

The keys from the above 3NF decomposition are: A, AD, AH, H, C, DF, and E.

Another minimal cover and 3NF decomposition solution is:

```
DF -> C  R1[DFC] key DF
H -> ADE  R2[HADE] key H
C -> H  R3[CH] key C
E -> F  R4[EF] key E
A -> B  R5[AB] key A
AD -> GH  R6[ADGH] key AD
```

Question 4 (10 marks)

I forgot to add the column for “strict” in the Assignment itself, this column is not required. It will NOT count towards the grades for this section.

Property Question	Serializable	Conflict-Serializabile	View-Serializabile	Recoverable	ACA	Strict
1.	✓	✓	✓	?	?	X
2.	✓	✓	✓	✓	✓	X
3.	✓	✓	✓	✓	✓	✓
4.	✓	✓	✓	✓	✓	✓
5.	X	X	X	✓	✓	✓

Marking: **2 marks** per question. “?” must have associated justifications. For example, in Q1, we cannot decide whether it’s recoverable or not, since the abort/commit sequence of these two transactions are not specified.

Question 5 (15 marks)

Answers:

Question 2:
Auto-commit is off in Session A and thus the command "commit;"
must be run to appear in Session B.

```
mysql> SELECT * FROM Accounts;
+----------+-----------------+---------+
| username | name            | balance |
+----------+-----------------+---------+
| capt     | Captain America | 1250.00 |
| hulk     | The Hulk        | 4750.00 |
| iron     | Iron Man        | 6000.00 |
| nat      | Black Widow     |  550.00 |
| thor     | Thor            | 4250.00 |
+----------+-----------------+---------+
5 rows in set (0.00 sec)
```

Question 6:
Withdraw $750 from Thor’s Account:

```
UPDATE Accounts SET balance=balance-750 WHERE username="thor";
```

Deposit $750 into Hulk’s Account:

```
UPDATE Accounts SET balance=balance+750 WHERE username="hulk";
```

```
Session B: select * from Accounts;
+----------+-----------------+---------+
| username | name            | balance |
+----------+-----------------+---------+
| capt     | Captain America | 1250.00 |
| hulk     | The Hulk        | 4750.00 |
| iron     | Iron Man        | 6000.00 |
| nat      | Black Widow     |  550.00 |
| thor     | Thor            | 4250.00 |
+----------+-----------------+---------+
5 rows in set (0.01 sec)
```

Withdraw $300 from Black Widow’s Account:

```
UPDATE Accounts SET balance=balance-300 WHERE username="nat";
```

Deposit $300 into the Hulk’s Account:

```
UPDATE Accounts SET balance=balance+300 WHERE username="hulk";
```

What happens and why?

- Session B hangs as it is pending on the completed transaction (commit) from Session A.

- After you commit Session A, Session B completed it’s attempt to transfer giving the following message:

```
Query OK, 1 row affected (47.33 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Question 8:

The Hulk’s balance is $5800.00.

Question 10:

Transfer 80% of the Hulk’s account balance to Captain America’s Account. Commit the transaction.

```
UPDATE Accounts SET balance=balance*0.2 WHERE username="hulk";
```

- The Hulk has $5,800. 80% is $4,640.

```
UPDATE Accounts SET balance=balance+4640 WHERE username="capt";
```

Question 11:

Transfer 50% of Iron Man’s funds to Captain America’s Account

```
UPDATE Accounts SET balance=balance*0.5 WHERE username="iron";
```

- Iron Man has $6,000. 50% is $3,000.

```
UPDATE Accounts SET balance=balance+3000 WHERE username="capt";
```

WITHOUT COMMITTING!! Session B’s Balances are:

```
mysql> select * from Accounts;
+----------+-----------------+---------+
| username | name            | balance |
+----------+-----------------+---------+
| capt     | Captain America | 5890.00 |
| hulk     | The Hulk        | 1160.00 |
| iron     | Iron Man        | 6000.00 |
| nat      | Black Widow     |  250.00 |
| thor     | Thor            | 3500.00 |
+----------+-----------------+---------+
5 rows in set (0.00 sec)
```

NOTICE that it DOES NOT reflect the latest transfer from Iron Man.

For reference only, Session A looks like this prior to commiting:

```
mysql> select * from Accounts;
+----------+-----------------+---------+
| username | name            | balance |
+----------+-----------------+---------+
| capt     | Captain America | 8890.00 |
| hulk     | The Hulk        | 1160.00 |
| iron     | Iron Man        | 3000.00 |
| nat      | Black Widow     |  250.00 |
| thor     | Thor            | 3500.00 |
+----------+-----------------+---------+
5 rows in set (0.00 sec)
```

Question 12:

If I implement the rollback, then I will have:

SESSION A:

```
mysql> rollback;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from Accounts;
+----------+-----------------+---------+
| username | name            | balance |
+----------+-----------------+---------+
| capt     | Captain America | 5890.00 |
| hulk     | The Hulk        | 1160.00 |
| iron     | Iron Man        | 6000.00 |
| nat      | Black Widow     |  250.00 |
| thor     | Thor            | 3500.00 |
+----------+-----------------+---------+
5 rows in set (0.00 sec)
```

SESSION B:

```
mysql> select * from Accounts;
+----------+-----------------+---------+
| username | name            | balance |
+----------+-----------------+---------+
| capt     | Captain America | 5890.00 |
| hulk     | The Hulk        | 1160.00 |
| iron     | Iron Man        | 6000.00 |
| nat      | Black Widow     |  250.00 |
| thor     | Thor            | 3500.00 |
+----------+-----------------+---------+
5 rows in set (0.00 sec)
```