Jason Lai
CS 143 204995126
February 14, 2021 Dis 1B

Homework 4

1 Problem 1

Yes, the decomposition is lossless. This is because the shared attribute A between the relations (A, B, C, F) and (A, D, E) is a key for the latter relation (A, D, E), and a decomposition is lossless if the shared attribute(s) are the key of one of the decomposed tables.

Work:
Closure of shared attribute A:

$$\begin{aligned}\{A\}+ &= \{A, B, C\} \qquad A \to BC \\ &= \{A, B, C, D\} \qquad B \to D \\ &= \{A, B, C, D, E\} \qquad CD \to E\end{aligned}$$

We have the logical implication $A \to DE$ so we have that A is a key of the relation (A, D, E).

2 Problem 2

The non-trivial functional dependencies satisfied by the following relation are as follows: $A \to B$, $C \to A$, $C \to B$.

Work:

A	B	C
a_1	b_1	c_2
a_1	b_1	c_2
a_2	b_1	c_1
a_2	b_1	c_3

From the above, we see that have $A \to B$ is satisfied by all four tuples, since tuples with matching A attributes have matching B attributes. Then $C \to B$ and $C \to A$ are also satisfied by all four tuples as all tuples with matching C attributes have matching A and B attributes. Rows 3 and 4 can have the same A and B attributes despite having different C attributes, since the definition of functional dependencies does not have any constraints on the left hand side of a dependency given the right hand side.

3 Problem 3

a) `sid`→`(dept, cnum)`, `(dept, cnum)`→`sid`

b) `sid`→`(dept, cnum)`

Work:
According to Piazza, the question is asking for functional dependencies that imply the cardinalities required in the problem.

a) The set of functional dependencies `sid`→`(dept, cnum)`, `(dept, cnum)`→`sid` gives us a one-to-one relationship as every tuple with the same `sid` must have the same `(dept, cnum)` and every tuple with the same `(dept, cnum)` must have the same `sid`.

b) The functional dependency `sid`→`(dept, cnum)` gives us a one-to-many relationship as each `sid` can only have one `(dept, cnum)`, but many different `sid`'s can have the same `(dept, cnum)`. Semantically, this means that many students can be assigned to the same class, so we have a one-to-many relationship.

4 Problem 4

a) Yes, A is a key for R, as its closure, $\{A\}+$, logically implies all attributes in R.

b) Yes, BC is a key for R, as its closure, $\{BC\}+$, logically implies all attributes in R, and its subsets: B and C are not keys, since their closures do not logically imply all attributes.

Work:

a) $\{A\}+ = \{A, B, C\} \quad A \to BC$
$= \{A, B, C, D\} \quad B \to D$
$= \{A, B, C, D, E\} \quad CD \to E$

b) $\{BC\}+ = \{B, C, D\} \quad B \to D$
$= \{B, C, D, E\} \quad CD \to E$
$= \{A, B, C, D, E\} \quad E \to A$

5 Problem 5

No, the relation R given the set of functional dependencies is not in **BCNF**. This is because none of the functional dependencies in the set are keys, since the attribute F is not implied by any of them.

The set of relations normalized from R that are in **BCNF** are as follows:

$$R1(A, B, C),\ R2(C, E),\ R3(B, D),\ R4(A, F)$$

Work:

$\{A\}+ = \{A, B, C, E, D\}$ => $R1(A, B, C, E, D),\ R2(A, F)$

$\{C\}+ = \{C, E\}$ => $R1(A, B, C, D),\ R2(C, E),\ R3(A, F)$

$\{B\}+ = \{B, D\}$ => $R1(A, B, C),\ R2(C, E),\ R3(B, D),\ R4(A, F)$

6 Problem 6

a)
```
CREATE TABLE Laptop(
    ...
    weight REAL,
    ...
    CHECK(0 < weight AND weight <= 5),
    ...
);
```

b)
```
CREATE TRIGGER weighLaptop
INSERT ON Laptop
REFERENCING NEW ROW AS NEW_L
FOR EACH ROW
WHEN(weight <= 0 OR weight > 5)
BEGIN
    UPDATE Laptop SET weight = NULL WHERE model = NEW_L.model;
END;
```

7 Problem 7

a)
```
CREATE TABLE Employee(
    eid INTEGER NOT NULL,
    name VARCHAR(30) NOT NULL,
    salary REAL NOT NULL,
    PRIMARY KEY(eid)
);
CREATE TABLE LeavingTime(
    eid INTEGER NOT NULL,
    date DATE NOT NULL,
    time TIME NOT NULL,
    FOREIGN KEY(eid) REFERENCES Employee(eid),
    PRIMARY KEY(eid, date)
);
```

b)
```
INSERT INTO LeavingTime(eid, date, time) VALUES (143, '2015-04-01', '04:00:00');
```

c) Assuming that when the employee swiped her card an hour after the first time that it is still the same day i.e. same date, then this would generate an error. The issue is that this violates the key constraints of the `LeavingTime` relation, as the `date` attribute is part of the primary key. Therefore, if the employee swipes her card a second time on the same day, then the system would try to `INSERT` a second tuple with her `eid` and that `date`, leading to two tuples with the same primary key, but different `time` values since the swipes are an hour apart, which violates the key constraints.

d)
```
WITH dupes AS
( SELECT ROW_NUMBER() OVER(PARTITION BY eid, date ORDER BY time ASC)
  AS row_num FROM LeavingTime )
DELETE FROM dupes
WHERE row_num > 1;
```

8 Problem 8

After the update statement is executed, all the tuples in the table `R` are `{(1, 27), (100, 0), (100, 0)}`.

Work:
Initially, the table `R` only has the tuple `(1, 0)`.

After the `UPDATE` statement, the tuple is `(1, 3)`.

On the `TRIGGER`, the tuple becomes `(1, 9)`.

On the second `TRIGGER`, the tuple becomes `(1, 27)`.

The `TRIGGER` then inserts another tuple `(100, 0)`.

The first `TRIGGER` finally inserts the third tuple `(100, 0)`.

`{(1, 27), (100, 0), (100, 0)}`