Database Management Systems (COP 5725)

(Spring 2019)

Instructor: Dr. Markus Schneider

TA: Kyuseo Park

Exam 3 Solutions

Name:	
UFID:	
Email Address:	

Pledge (Must be signed according to UF Honor Code)

On my honor, I have neither given nor received unauthorized aid in doing this assignment.

____

Signature

For scoring use only:

	Maximum	Received
Question 1	40	
Question 2	35	
Question 3	25	
Total	100	

Question 1 (Functional Dependencies) [40 points]

Consider the following table which is used to store a list of baseball players.

ID	Player_name	Position	Team	Head coach name
001	P1	First baseman	Red hacks	H1
002	P2	Third baseman	didgers	H2
003	P3	Outfielder	Red hacks	H1
004	P4	Pitcher	Red hacks	H1
005	P5	Catcher	Hubs	H3

1. Explain which anomalies can occur with the table above, why they happen, and how to solve the problem. [5 points]

   Insertion anomaly: Information of a head coach H4 of a new team cannot be inserted without the information of a player.

   Deletion anomaly: If the player P5 is deleted, Team and head coach information is also removed from the database.

   Update anomaly: Every tuple has to be updated with the update of the head coach of Red hacks.

   It happens because of the transitive functional dependency. ID→Team, Team→Head coach name, then ID→Head coach name

   Remove the transitive functional dependency by splitting the schema into the two 3NF schemas.

ID	Player_name	Position	Team
001	P1	First baseman	Red hacks
002	P2	Third baseman	didgers
003	P3	Outfielder	Red hacks
004	P4	Pitcher	Red hacks
005	P5	Catcher	Hubs

Team	Head coach name
Red hacks	H1
didgers	H2
Hubs	H3

2. Consider the relational schema R(A, B, C, D, E, F, G) and the set F = {AB→CF, A→G, BC→E, C→D} of functional dependencies. Find all candidate keys of R by using the tableau method. [5 points]

I(isolated)	L(left)	B(both)	R(right)
-	A, B	C	D, E, F, G

Union of I and L: AB
AB^+ = ABCDEFG

Since AB^+ includes every attribute of R, AB is the only candidate key, and the algorithm terminates.

3. Suppose relation R(A, B, C) currently has only the tuple (0, 0, 0), and it satisfies the functional dependencies F = {A→B, B→C}. Which of the following tuples can be inserted next into R? Explain the reason if the tuple cannot be inserted. [5 points]
   - (0, 1, 1)
   - (1, 0, 0)
   - (0, 0, 1)
   - (1, 2, 3)
   - (1, 1, 0)
   - (1, 0, 2)
   - (1, 0, 1)

Tuples (1, 0, 0), (1, 2, 3), and (1, 1, 0) can be inserted.

(0, 1, 1) violates A→B

(1, 0, 2), (1, 0, 1), and (0, 0, 1) violate B→C

4. Consider the relational schema R(A, B, C, D, E) and the set F = {A→B, A→C, CD→E, B→D, E→A} of functional dependencies. Which of the following functional dependencies can be implied by F? Justify your answer. [5 points]
   - CD→AC
   - BD→CD
   - BC→CD
   - AC→BC

CD→AC, BC→CD, and AC→BC can be implied because $CD^+ = BC^+ = AC^+$ = ABCDE

BD^+ = BD, hence it is not.

An alternative answer could be based on Armstrong's axioms.

5. For a relational schema R(A, B, C), write an SQL assertion command to ensure that the functional dependency AB→C holds. [5 points]

```
CREATE ASSERTION FDCHECK CHECK
(NOT EXISTS (SELECT A, B
               FROM R
               GROUP BY A, B
               HAVING COUNT(distinct C) > 1))
```

6. For the relational schema R(A, B, C, D, E), determine if F = {A→C, AC→D, D→C, D→E} and G = {A→BC, E→B, D→ACE} are equivalent. [5 points]

F has the left-hand sides: A, AC, D. We calculate A^+, AC^+, and D^+ with respect to G:

A^+ = ABC

AC^+ = ABC

D^+ = ABCDE

These attribute closures do not cover all right-hand sides of FDs in F.

G has the left-hand sides: A, D and E. We calculate A^+, D^+, and E^+ with respect to F:

A^+ = ABC

D^+ = ABCDE

E^+ = BE

These attribute closures cover all right-hand sides of FDs in G. However, F and G are not equivalent.

7. Compute a minimum cover in *standard form* for the relation R(A, B, C, D, E, F) and the functional dependencies F = {A→BC, C→BD, B→AC, E→BF}. [10 points]

- F_c = F
  - F_c = {A→BC, C→BD, B→AC, E→BF}
- LHS reduction
  - Nothing has to be done since all left sides are singleton attribute sets.
- RHS reduction
  - Checking A→BC:
    - (not considering A→B) A^+ = ABCD. Since it includes B, B can be removed.
    - F_c = {A→C, C→BD, B→AC, E→BF}
    - (not considering A→C) A^+ = A. Since it does not include C, C cannot be removed.
  - Checking C→BD:
    - (not considering C→B) C^+ = CD. Since it does not include B, B cannot be removed.
    - (not considering C→D) C^+ = ABC. Since it does not include D, D cannot be removed.
    - F_c = {A→C, C→BD, B→AC, E→BF}
  - Checking B→AC:
    - (not considering B→A) B^+ = BCD. Since it does not include A, A cannot be removed.
    - (not considering B→C) B^+ = ABCD. Since it includes C, C can be removed.
    - F_c = {A→C, C→BD, B→A, E→BF}
  - Checking E→BF:
    - (not considering E→B) E^+ = EF. Since it does not include B, B cannot be removed.
    - (not considering E→F) E^+ = ABCDE. Since it does not include F, F cannot be removed.
    - F_c = {A→C, C→BD, B→A, E→BF}
- Hence, we obtain F_c in *standard form* = {A→C, C→B, C→D, B→A, E→B, E→F}

Question 2 (Normalization) [35 points]

1. Explain concisely in your own words what, starting from the UNF, the normal forms 1NF, 2NF, 3NF, and BCNF achieve to improve the quality of a database schema. [5 points]

- Unnormalized Form (UNF)
- 1NF: UNF + Remove structures in attribute values
- 2NF: 1NF + Remove partial FDs
- 3NF: 2NF + Remove transitive FDs
- BCNF: 3NF + Remove FDs with no candidate key on the left-hand side

2. For the relation schema R = ABCDEGHI and the set of functional dependencies F = {A→B, ABCD→E, EG→HI, ACDG→EH}, determine a lossless join and dependency preserving 3NF decomposition of R. Provide all the steps in detail. [10 points]

- Minimal Cover
  - Initialization: Fc = F = {A→B, ABCD→E, EG→HI, ACDG→EH}
  - LHS reduction:
    - Checking ABCD→E:
      - BCD^+ = BCD, therefore A cannot be removed.
      - ACD^+ = ABCDE, therefore, B can be removed.
      - ABD^+ = ABD, therefore, C cannot be removed.
      - ABC^+ = ABC, therefore, D cannot be removed.
    - Checking EG→HI:
      - E^+ = E, therefore, G cannot be removed.
      - G^+ = G, therefore, G cannot be removed.
    - Checking ACDG→EH:
      - CDG^+ = CDG, therefore, A cannot be removed.
      - ADG^+ = ABDG, therefore, C cannot be removed.
      - ACG^+ = ABCG, therefore, D cannot be removed.
      - ACD^+ = ABCDE, therefore, G cannot be removed.
    - Fc = {A→B, ACD→E, EG→HI, ACDG→EH}
  - RHS reduction:
    - Checking A→B:
      - A^+ = A, therefore, B cannot be removed.
    - Checking ACD→E:
      - ACD^+= ABCD, therefore, E cannot be removed.
    - Checking EG→H
      - EG^+ = EGI, therefore, H cannot be removed.

        - Checking $EG \to I$:
            - $EG^+ = EGH$, therefore, I cannot be removed.
        - Checking $ACDG \to E$:
            - $ACDG^+ = ABCDEGHI$, therefore, E can be removed.
        - Checking $ACDG \to H$:
            - $ACDG^+ = ABCDEGHI$, therefore, H can be removed.
        - $Fc = \{A \to B, ACD \to E, EG \to HI, ACDG \to \emptyset \}$
    - Removing redundant FDs:
        - $Fc = \{A \to B, ACD \to E, EG \to HI\}$
- Reconstruct Schema:
    - R1 (AB); F1 $\{A \to B\}$
    - R2 (ACDE); F2 $\{ACD \to E\}$
    - R3 (EGHI); F3 $\{EG \to HI\}$
- Candidate key:
    - $A^+ = AB$
    - $ACD^+ = ABCDE$
    - $ACDG^+ = ABCDEGHI$

  Candidate key is ACDG. Therefore, we add another relation R4 (ACDG).
- No relation is contained in other relations.
- Decomposition:
    - R1 (AB); F1 $\{A \to B\}$
    - R2 (ACDE); F2 $\{ACD \to E\}$
    - R3 (EGHI); F3 $\{EG \to HI\}$
    - R4 (ACDG)

3. Consider the relation schema R = ABCDEG with the set F = {A→BC, D→E, CD→G, B→D, G→A} of functional dependencies.

(a) Provide a BCNF decomposition of R with respect to F. [5 points]

Solution 1: BCNF Decomposition algorithm

R is not in BCNF and the first violating FD is D→E. D^+ = DE, therefore, we split R into R1 = DE and R2 = ABCDG.

Compute restrictions F1 and F2: F1 = {D→E} and F2 = {A→BCDG, CD→ABG, B→D, G→ABCD}.

R2 is not in BCNF since we have B→D and B is not a superkey. B^+ = BD, therefore, we split R2 into R21 = BD and R22 = ABCG.

Compute restrictions F21 and F22: F21 = {B→D} and F22 = {A→BCG, G→ABC}. Both R21 and R22 are in BCNF.

Therefore, the final decomposition D of R is D = {R1, R21, R22} = {DE, BD, ABCG}.

Solution 2: 3NF Synthesis algorithm first, then check whether 3NF schemas are also in BCNF. If yes, that is fine. If not, decompose those schemas that violate BCNF into BCNF.

Minimal cover = {
f1: A → BC;
f2: B → D;
f3: D → E;
f4: G → A;
f5: CD → G
}
Candidate keys are A, G, BC, CD
Generation of relation schemas from the FDs f1 to f5.
R1 = {A, B, C}, F1 = {f1}
R2 = {B, D}, F2 = {f2}
R3 = {D, E}, F3 = {f3}
R4 = {A, G}, F4 = {f4}
R5 = {C, D, G}, F5 = {f5}

R1, …, R5 with the corresponding F_i are also in BCNF.

(b) We decompose R into schemas R1(ABC) and R2(ADEG). Use the Nonadditive Join Test for Binary Decompositions to determine if this is a lossless decomposition. Also, show whether it is dependency preserving. [5 points]

R1 ∩ R2 = A and A+ = ABCDEG = R.

R1 – R2 = BC and BC ⊆ R. Therefore, the decomposition is lossless.

However, B→D is not in R1 or R2, and therefore, it is not dependency preserving.

5. Assume that we have a relation R = ABCDE with the set of functional dependencies F = {AC→DE, B→E, D→B, E→A}. R is decomposed into relations R1 = ABC, R2 = BD and R3 = CDE. Use the Chase test to show if this is a lossless decomposition. [10 points]

Initial tableau:

	A	B	C	D	E
ABC	a	b	c	d_1	e_1
BD	a_2	b	c_2	d	e_2
CDE	a_3	b_3	c	d	e

B→E:

	A	B	C	D	E
ABC	a	b	c	d_1	e_1
BD	a_2	b	c_2	d	e_1
CDE	a_3	b_3	c	d	e

D→B:

	A	B	C	D	E
ABC	a	b	c	d_1	e_1
BD	a_2	b	c_2	d	e_1
CDE	a_3	b	c	d	e

E→A:

	A	B	C	D	E
ABC	a	b	c	d_1	e_1
BD	a	b	c_2	d	e_1
CDE	a_3	b	c	d	e

B→E:

	A	B	C	D	E
ABC	a	b	c	d_1	e
BD	a	b	c_2	d	e
CDE	a_3	b	c	d	e

E→A:

	A	B	C	D	E
ABC	a	b	c	d_1	e
BD	a	b	c_2	d	e
CDE	a	b	c	d	e

The decomposition is lossless.

Question 3 (Database Integrity) [25 points]

Consider the following relational schema:

```
Faculty (fid, name, tenured, dname, salary)
Department (dname, phone_number)
Course (fid, dname, semester, cname)
```

The type of the attribute 'tenured' in the Faculty relation is Boolean.

1. Write an assertion check for the following condition: `The number of tenured faculty members cannot be greater than the number of untenured faculty members.` [5 points]

```
CREATE ASSERTION Q1
CHECK ((SELECT COUNT(*) FROM faculty WHERE tenured = 'T')
        <
       (SELECT COUNT(*) FROM faculty WHERE tenured = 'F'));
```

2. Write an assertion check for the following condition: `CISE department cannot have more than 30 courses in a Semester.` [5 points]

```
CREATE ASSERTION Q2
CHECK (NOT EXISTS (SELECT c.did
                   FROM department d, course c
                   WHERE d.dname = c.dname
                         And d.dname = 'cise'
                   GROUP BY c.dname, c.semester
                   HAVING COUNT(cname) > 30)
```

3. Write a trigger (one trigger command) for the following: `When a new record is inserted into Faculty, if the department does not exist in the Department table, then insert a record into Department with the same department name, phone number = null. Additionally, when a record is deleted from Faculty, if the faculty member is tenured, then arise an error (use RAISE_APPLICATION_ERROR(-10000,'error')).` [5 points]

```
CREATE TRIGGER Q3
BEFORE INSERT OR DELETE ON Faculty
FOR EACH ROW
BEGIN
      IF INSERTING THEN
      WHEN (NOT EXISTS (SELECT * FROM Department WHERE dname = :new.dname))
            INSERT INTO Department VALUES (:new.dname, NULL)
      END IF

      IF DELETING THEN
            IF :OLD.TENURED = 'TRUE' THEN
```

```
                  RAISE_APPLICATION_ERROR(-10000,'error')
            END IF
      END IF

END
```

4. Write a trigger for the following: `When a department name is updated on Department, also update the corresponding record on Faculty with the same department name.` [5 points]

```
CREATE TRIGGER Q4
AFTER UPDATE ON DEPARTMENT
FOR EACH ROW
BEGIN
      UPDATE FACULTY SET DNAME = :NEW.DNAME WHERE DNAME = :OLD.DNAME
END
```

5. Write a trigger for the following: `When a record is updated on Faculty, if the salary increases more than double then do not increase the salary.` [5 points]

```
CREATE TRIGGER Q5
BEFORE UPDATE ON FACULTY
FOR EACH ROW WHEN (:NEW.SALARY > (2 * :OLD.SALARY))
BEGIN
      :NEW.SALARY = :OLD.SALARY
END;
```