Homework 3
Name: Jigeesha Jain
B-Number: B00928112

1. The following are some of the relations transformed from the ER diagram for the Student Registration System. Note that some changes have been made due to the creation of a single-attribute key for Classes.

Students(sid, firstname, lastname, status, gpa, email)
Courses(dept_code, course#, title, credits, deptname)
Classes (classid, dept_code, course#, sect#, year, semester, start_time, end_time, limit, size, classroom, capacity, fid) /* note: classid is added to serve as a single-attribute key */
Faculty(fid, firstname, lastname, rank, office, email, deptname)
Enrollments(sid, classid, lgrade, ngrade)

Do the following for each relation schema:

(a) [20%] Identify all non-trivial functional dependencies. Don't make unrealistic assumptions about the data. Use the union rule to combine the functional dependencies as much as possible. Furthermore, if a functional dependency is redundant (i.e., it can be derived from the ones you keep), don't include it.

Answer:

- **Students (sid, firstname, lastname, status, gpa, email)**
  Functional dependencies of Student are:
  sid -> Firstname lastname status gpa email
  email -> sid

- **Courses (dept_code, course#, title, credits, deptname)**
  Functional dependencies of Courses are:
  dept_code course# -> title
  course# -> credits
  dept_code -> deptname

- **Classes (classid, dept_code, course#, sect#, year, semester, start_time, end_time, limit, size, classroom, capacity, fid)**
  Functional dependencies of Classes are:
  classid -> dept_code course# sect# year semester start_time end_time limit size classroom capacity fid
  dept_code course# sect# year semester -> classid
  classroom -> capacity

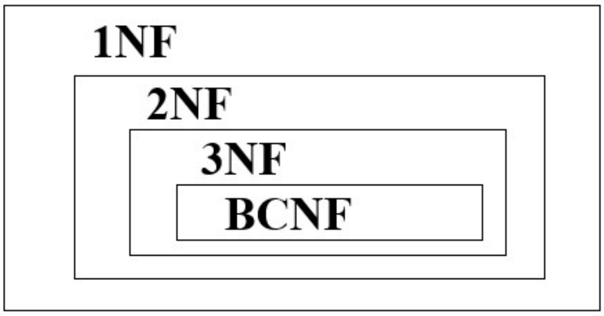
- **Faculty (fid, firstname, lastname, rank, office, email, deptname)**
  Functional dependencies of Faculty are:
  fid-> firstname lastname rank office email deptname
  email->fid
  office->deptname

- **Enrollments (sid, classid, lgrade, ngrade)**
  Functional dependencies of Enrollments are:
  sid classid -> lgrade
  ngrade->lgrade
  lgrade ->ngrade

(b) [20%] Determine whether or not each table is in 3NF or in BCNF. Justify your conclusion.

Answer:

According to the lecture slide no.62 of Normalization if a relation is in BCNF it is also in 3NF, but the other way it is not true. In order words, if we prove that a relation is in BCNF we do not have to prove that it is in 3NF as well. The below image is taken from lecture slides for better clarity of the above-mentioned point.



Now to find whether each table is in BCNF we need to find the candidate key as per the flowchart in lecture slide.

- Students (sid, firstname, lastname, status, gpa, email)

Candidate key: sid, email
Here in the first functional dependency sid is the superkey hence first relation is in 3NF and the second functional dependency, email is also a superkey as per the SRS requirement document different students have different mail ids. Hence the relation Students is in BCNF.

- Courses (dept_code, course#, title, credits, deptname)

Candidate key: dept_code, course#
In this relation, 1st and 3rd functional dependency are in 3NF but the 2nd functional dependency is not as course# is not a superkey. So this schema is not in 3NF.

•Classes (classid, dept_code, course#, sect#, year, semester, start_time, end_time, limit, size, classroom, capacity, fid)
Candidate key: classid, dept_code course# sec# year semester
Here both the functional dependencies 1st and 2nd are BCNF as the condition of BCNF i.e. a superkey is satisfied. According to lecture slide, R is in BCNF if for every non-trivial FD, the left side is a superkey and as classid and dept_code course# sect# year semester are superkeys so it is in BCNF. But the third FD i.e. classroom -> capacity is not in 3NF as classroom is not a superkey and capacity is not a prime i.e. it is not a part of candidate key so it is not in 3NF

• Faculty (fid, firstname, lastname, rank, office, email, deptname)
Candidate key: fid, email, offices
Here it is in BCNF as all 3 FD which are mentioned above have their left side (i.e. fid, email and office) are superkeys so they are in BCNF and thus in 3NF.

• Enrollments (sid, classid, lgrade, ngrade)
Candidate key: sid, classid
This schema enrollments is also not in 3NF form as in both the Functional dependencies ngrade->lgrade, lgrade ->ngrade the attributes ngrade and l grade are not a prime attribute and neither are the attributes lgrade and ngrade superkeys. Hence they do not satisfy the condition of 3NF so they are not in 3NF.

(c) [20%] For each table that is not in 3NF, decompose it into 3NF schemas using Algorithm LLJD-DPD-3NF. Show the result after each step of the algorithm. Are the decomposed schemas in BCNF? Justify your answer.

Answer:

We can see that the tables Courses, Classes and Enrollments are not in 3NF hence we will decompose it into 3NF schemas using Algorithm LLJD-DPD-3NF.

- Courses (dept_code, course#, title, credits, deptname)

  1.Candidate keys are: dept_code course#

  2.Fmin: dept_code course# -> title, course#->credits, dept_code->deptname, dept_name->dept_code

  3.Form Relations:

  R1 = (dept_code course #,title)
  R2= (course# credits)
  R3= (dept_code deptname)

4. The above decomposed schema is in BCNF because as per the lecture slide, R is in BCNF if for every non-trivial FD, the left side(i.e. dept_code course#, course#, dept_code) is a superkey which is true for the above decomposed schema.

- Classes (classid, dept_code, course#, sect#, year, semester, start_time, end_time, limit, size, classroom, capacity, fid)

  1. Candidate keys are : classid, dept_code course# sec# year semester

  2.Fmin: classid -> dept_code, classid->course#, classid->sec#, classid->year, classid->semester, classid -> semester, classid -> start_time, classid->end_time, classid->limit, classid->size, classid->fid, classroom->capacity, course#sec#year semester->classid

  3.Form relations: R1=(classid,dept_code course# sec# year semester start_time end_time limit size classroom fid)

  R2= (classroom, capacity)

4. The above decomposed schema is in BCNF because as per the lecture slide, R is in BCNF if for every non-trivial FD, the left side (i.e. classid, classroom) is a superkey which is true for the above decomposed schema.

- Enrollments (sid, classid, lgrade, ngrade)
  1.Candidate keys are: sid, classid
  2.Fmin: sid classid -> lgrade, ngrade -> lgrade, lgrade -> ngrade
  3.Form Relations:
  R1= (sid classid lgrade)
  R2 = (lgrade ngrade)

4. The above decomposed schema is in BCNF because asper the lecture slide, R is in BCNF if for every non-trivial FD, the left side (i.e. sid classid, lgrade) is a superkey which is true for the above decomposed schema.

2. Prove or disprove the following rules:

   (a) [10%] {B → CD, AB → E, E → C} |= {AE → CD}
   (b) [10%] {B → CD, AD → E} |= {AB → E}
   (c) [10%] {AB → AC, B → A} |= {B → C}
   (d) [10%] {AB → C, BC → A} |= {B → C}

   When proving a rule, **you can use all the six inference rules**, IR1 – IR6, in the lecture slides (i.e., reflexivity rule, augmentation rule, transitivity rule, decomposition rule, union rule and pseudotransitivity rule). To disprove a rule, just give **one counter example** by constructing a relation with appropriate attributes and tuples such that the tuples of the relation satisfy the functional dependencies on the left of the rule but do not satisfy the functional dependency on the right of the rule.

Answer:

a. {B → CD, AB → E, E → C} |= {AE → CD}

A	B	C	D	E
A1	B1	C1	D1	E1
A2	B1	C1	D1	E2
A2	B2	C1	D2	E2
A2	B3	C1	D1	E2

The above rule cannot be proved. It is **disproved** as in the above table we see that for all the tuples, functional dependencies B → CD, AB → E and E → C are satisfied. As mentioned in the question, as the rule is disproved, the above table gives the counter example which disproved the rule. As all the functional dependencies given are fulfilled except AE → CD in tuples 2,3 and 4

b. {B → CD, AD → E} |= {AB → E}

Given: B → CD, AD → E
To prove: AB → E
Proof: As it is given B → CD,
By IR4, B→ C and B→D
B→D and AD→E,
By IR6, AB→ E

Hence proved

c. {AB → AC, B → A} |= {B → C}

Given: AB → AC, B→A

To prove: B→ C

Proof: B→A … given

By IR2, BB→ AB,

Now, BB→AB and AB→AC,

By IR3, BB→AC,

By IR4, B→A and B → C

Hence proved.

d. $\{AB \rightarrow C, BC \rightarrow A\} \models \{B \rightarrow C\}$

A	B	C
A1	B1	C1
A1	B2	C1
A2	B2	C2
A3	B3	C3

This rule is **disproved** as per the above table, the given $AB \rightarrow C$, $BC \rightarrow A$ are both true whereas $B \rightarrow C$ is not applicable. When we check for the tuple 2 and 3 we can clearly see that $B \rightarrow C$ is not possible even though these rules $AB \rightarrow C$, $BC \rightarrow A$ are true.