POSSIBLE QUESTIONS-THERE WILL BE ALSO SQL QUERIES

What is "functional" about functional dependencies? [4 pts]

A1, A2, …An -> B is a **functional** dependency because in principle there is a function that takes a list of values, one for each of the attributes A1, A2, …An and produces a **unique** value (or no value at all) for B. This function is only "computed" by look-up in the relation. The functional dependency is a property of the semantics or meaning of the attributes.

2) List the steps of 3NF synthesis algorithm as described in class. [4 pts]

Synthesis algorithm for computing the decomposition on the basis of F:

Step1: Determine a canonical cover F_c for F.

Step2: For each FD $A \rightarrow B \in F_c$:

- create a relation schema Ra := $A \cup B$
- assign the FDs $F_A = \{C \rightarrow D \in F_c \mid C \cup D \subseteq R_A\}$ to R_A

Step3: If all schemas R_A created in Step2 do not contain a candidate key of the original schema *R*,

additionally create a relation with the schema $R_K = K$ and $F_K = Ø$ where K is candidiate key of *R*.
Step4: Eliminate schemas R_A that are contained in another schema RA.

3) What is meant by the closure of a set of functional dependencies? [4 pts]

Closure of a set F of FDs is the set F of all FDs that can be inferred from F
Closure of a set of attributes X with respect to F is the set X^+ of all attributes that are functionally determined by X
X^+ can be calculated by repeatedly applying IR1, IR2, IR3 using the FDs in F.
[Refer section 14.2.2 in textbook]

4) When are two sets of functional dependencies equivalent? How can we determine their equivalence?
[4 pts]

Two sets of FDs F and G are equivalent if:
- every FD in F can be inferred from G, and
- every FD in G can be inferred from F

Hence, F and G are equivalent if $F^+ = G^+$
Definition: F covers G if every FD in G can be inferred from F (i.e., if G^+ subset-of F^+)
F and G are equivalent if F covers G and G covers F. Refer text for the algorithm to check equivalence

of sets of FDs.

5). Multiple Choice Questions: [2pts each]

1. Consider relation R(A;B;C;D) with FD's A → D, B → D, and D → BC.
Which of the following is true about the decomposition of R into relations with schemas AB
and BCD? Explain your answer.

(a) The decomposition is neither lossless nor dependency-preserving.
(b) The decomposition is lossless, but not dependency-preserving.
(c) The decomposition is dependency-preserving, but not lossless.
(d) The decomposition is both lossless and dependency-preserving.

Ans:

The decomposition is lossless because the intersection of the two schemas, that is B,
functionally determines one of the two schemas, namely $B \to BCD$. To check, compute B^+,

which is BCD.

To check dependency preservation, note that $A^+ = ABCD$, thus, when we project dependencies
onto AB, we get $A \to B$. The projection onto BCD surely gives us the second and third of the
given FD's: $B \to D$ and $D \to BC$. We claim that the three dependencies $A \to D$, $B \to D$, and $D \to BC$
are equivalent to the three dependencies that we can preserve in the projection: $A \to B$, $B \to D$,
and $D \to BC$. The last two in each set are the same. Also, $A \to B$ is easily seen to follow from the
first three, and $A \to D$ follows from the last three. If we can preserve an equivalent set of FD's in
the projections, then surely we can preserve the given set.

2. Suppose we have a relation R(A;B;C;D;E) and the FD's A → DE, D → B, and E → C.
If we project R (and therefore its FD's) onto schema ABC, what is true about the key(s) for ABC? Explain why.
(a) Only ABC is a key
(b) Only A is a key
(c) Only DE is a key
(d) A, B, and C are each keys.
Ans: (b)
First, note that $A^+ = ABCDE$. Thus, when we project the FD's onto *ABC*, *A* is certainly a key. However, there cannot be any other keys, because neither *B* nor *C* have anything else in their closures, because they are not on the left sides of any FD's. Thus, *A* is the only key.

Exercise 2 : Functional Dependencies and Normalization [80 points]

1) Consider the relation STUDENT (SNO, SNAME, CNO, CNAME, ADDRESS) where the following FDs hold:
SNO → SNAME
CNO → CNAME
SNO → ADDRESS
Let attribute set (SNO, CNO) be denoted by A. Compute the closure of A, i.e. A^+ [5pts]
Answer:

A^+ = (SNO, CNO, SNAME, CNAME, ADDRESS)

2) Use your own words to explain how the closure of attribute set A, i.e. A^+, can be used to determine the containment of a FD in a closure F^+. [5pts]

Answer:

Assume we have a FD A→B and a set of FDs, which is denoted by F. First, we generate the closure of A, which is denoted by A^+, according to F. If B $\subseteq$ A^+, then we can conclude that FD A→B is in the closure F^+.

3) Consider the relation R (CLASS, MEET_DAY, STUDENT, GRADE, COMPLEX, MANAGER) with the meaning:

- A STUDENT takes a CLASS that meets on several day or days every week (given by the attribute MEET_DAY).
- A STUDENT can take multiple classes.
- The STUDENT gets a GRADE in the CLASS.
- Each STUDENT lives in only one COMPLEX
- Each COMPLEX has only one MANAGER, but each manager can manage one or more COMPLEXes.

(a) Find all the non-trivial FDs (Functional Dependencies) that hold in R. [5pts]

Answer:
(STUDENT, CLASS) → GRADE
STUDENT → COMPLEX
COMPLEX → MANAGER
(STUDENT, CLASS) → COMPLEX
(STUDENT, CLASS) → (GRADE, COMPLEX)
STUDENT → MANAGER
(STUDENT, CLASS) → (GRADE, COMPLEX, MANAGER)
(STUDENT, CLASS) → MANAGER
(STUDENT, CLASS) → (GRADE, MANAGER)
STUDENT → (COMPLEX, MANAGER)

(b) What kind of problems does relation R have? Give an explanation for each problem you found.
[5pts]

Answer:
Redundancies
Update anomalies
Insertion anomalies
Deletion anomalies

---

4) Consider a relation R with four attributes, XYZW, and a set of functional dependencies F

F: XY → Z, XY → W, Z → X, W → Y

a. Identify the candidate key(s) for R.
b. Identify the best normal form that R satisfies (1NF, 2NF, 3NF, or BCNF).
c. Is R in BCNF? If *R* is not in BCNF, decompose it into a set of BCNF relations that preserve the dependencies.
[10 points]

a. Candidate keys: XY, ZW, ZY, XW
b. R is in 3NF but not BCNF due to the FDs: Z → X and W → Y. Neither W nor Z is a candidate key.
c. No.

One possible decomposition that is in BCNF is XZ, WZ and WY. But this decomposition does not
preserve the dependencies XY → Z and XY → W. If we include a relation such as XYZ then we are
left with the violating FD: Z → X. Similarly for XYW. So, there is no dependency-preserving
decomposition into BCNF unless we do a join.

5) Consider the relation
R1 (CLASS, MEET_DAY, STUDENT, GRADE, TRANSFER_CLASS, CREDITS_AWARDED)
with the following FDs:
{CLASS, STUDENT} → {GRADE}
{STUDENT, TRANSFER_CLASS} → {CREDITS_AWARDED}
Please decompose the relation *R1* into BCNF and list each step of your work. [10 points]

Answer:
Canonical cover:
FD1: {CLASS, STUDENT} → {GRADE}
FD2: {STUDENT, TRANSFER_CLASS} → {CREDITS_AWARDED}
Candidate key: {CLASS, STUDENT, MEET_DAY, TRANSFER_CLASS}
Decomposition:
From FD1: R1a = {CLASS, STUDENT, GRADE}
From FD2: R1b = {STUDENT, TRANSFER_CLASS, CREDITS_AWARDED}
Since no relations contain candidate key, additionally create a relation
R1c = {CLASS, STUDENT, MEET_DAY, TRANSFER_CLASS}
Now R1a, R1b and R1c are in the 3NF. Also it is in BCNF.

6) Let *AB* be a shortcut for {*A*, *B*}. Imagine that we have the relation *R2*(*ABCD*) with FDs
$AB \rightarrow C$, $C \rightarrow D$, $D \rightarrow A$. Explain why this relation is not in BCNF, but is in 3NF. Why would it be
problematic to decompose *R2* into BCNF? [5 points]

Answer:
The candidate key of R2 is {AB} {BC} or {BD}. So the right side of all the FDs is part of a key. That
is to say, R2 is in 3NF. But the left side of FD $C \rightarrow D$, $D \rightarrow A$ is not a superkey, which violates the
BCNF. It would be a bad idea to decompose this relation, however, because if we did so it would mean
that there is some FD that is not present in any relation.

7) Now consider the relation *R3* (*ABCDEFGH*), with FDs $BC \rightarrow AD$, $E \rightarrow FH$, $F \rightarrow GH$. Please
decompose *R3* into BCNF and list each step of the process. [10 points]

Answer:
Canonical cover:
FD1: $BC \rightarrow AD$

FD2: $E \rightarrow F$
FD3: $F \rightarrow GH$
Candidate key: {B, C, E}
Decomposition:
From FD1: R3a = {B, C, A, D}
From FD2: R3b = {E, F}
From FD3: R3c = {F, G, H}
Since no relation contains the candidate key, additionally create a relation
R3d = {B, C, E}
Now R3a, R3b, R3c and R3d are in 3NF and BCNF.

8) Given the following relational schema R
R(A,B,C,D,E,F,G,H,I,J)
and functional dependencies:
B → E ; E → F,H ; B,C,D → G ; C,D → A ; A → J
I → B,C,D,E ; H → I
Answer these questions: [10 points]
(a) Does the functional dependency B → J hold?

(b) List the candidate keys of R.
(c) Normalize R into BCNF. Make sure to underline primary key fields.
Answer:
(a) The functional dependency B → J holds as J is in the closure of B = B+ = {A,B,C,D,E,F,G,H,I,J}.
(b) The candidate keys of R are {B}, {E}, {H}, {I}.
(c) Since there are so many keys of R, there is not much normalization to be done.
The only two FDs that violate BCNF are: C,D → A and A → J.
Final normalized relations are:
R1 (A,J)
R2 (C,D,A)
R3 (B,C,D,E,F,G,H,I)
Note that could chose any of the candidate keys as the primary key.
9) Suppose functional dependency B → C holds in relation R(A;B;C;D).
For every additional functional dependency, state if it makes R to be in 3NF, BCNF or both. [5 points]
(a) D → AB
(b) AC → D
(c) CD → B
(d) AD → B
Answer:
In (a), *D* is the only key, so *B→C* is both a 3NF and BCNF violation.
In (b), *AB* is the only key, so both FD's are 3NF and BCNF violations.
In (c), we can check that the keys are *ACD* and *ABD*. Both FD's violate BCNF, but all the
attributes are prime, so there can be no 3NF violation.
In (d), *AD* is the only key, so *B→C* violates both normal forms.
10) Exercise 14.33 from textbook. [10 points]
Consider the relation for published books:

BOOK (Book_title, Authorname, Book_type, Listprice, Author_affil, Publisher)
Author_affil refers to the affliation of author. Suppose the following dependencies exist:

Book-title → Publisher, Book_type
Book_type → Listprice
Authorname → Author_affil
a. What normal form is the relation in? Explain you answer.
b. Apply normalization until you cannot decompose the relations further. State the reasons behind each
decomposition.
Answer:
a. The key for this relation is Book_title,Authorname. This relation is in 1NF and not in 2NF as no
attributes are FFD (Fully Functionally Dependent) on the key. It is also not in 3NF.
b. 2NF decomposition:
Book0(Book_title, Authorname)
Book1(Book_title, Publisher, Book_type, Listprice)
Book2(Authorname, Author_affil)
This decomposition eliminates the partial dependencies.
3NF decomposition:
Book0(Book_title, Authorname)
Book1-1(Book_title, Publisher, Book_type)
Book1-2(Book_type, Listprice)
Book2(Authorname, Author_affil)
This decomposition eliminates the transitive dependency of Listprice.
***