Subankamo 300471606

Question 1. Functional Dependencies and Normal Forms [4 marks]

Consider a relation schema N (R, F) where $R = \{A, B, C\}$. Suppose we find the following two tuples in an instance of this relation schema.

A	B	C
a	b	3
a	d	3

Which of the following functional dependencies does definitely **not** hold over the relation schema N? Justify your answer.

1) $A \rightarrow B$
2) $B \rightarrow A$
3) $C \rightarrow B$
4) $A \rightarrow C$

Answer:

1. **A → B:** functional dependency does not hold because a A can be associated with more than one B
   >In the given tuples, we have A = 'a' with different values of B ('b' and 'd')

2. **B→A:** functional dependency holds because B can be associated with only one A
   >In the given tuples, we have B = 'b' with A = 'a' and B = 'd' with A = 'a'.
3. **C→B:** functional dependency does not hold because C can be associated with more than one B
   >In the given tuples, we have C = 3 with different values of B ('b' and 'd')
4. **A→C:** functional dependency holds because A can be associated with only one C
   >In the given tuples, we have A = 'a' with the same value of C = 3

Question 2. Normal Forms

Consider a relation schema N (R, F) where $R = \{A, B, C, D\}$. For each of the following sets F of functional dependencies, determine which normal form (1NF, 2NF, 3NF, BCNF) the relation schema N is in. Justify your answer.

Hint: Note that in the first three cases, BC is the only key for N. In the fourth case, BC and BA are the only keys.

1) $F = \{BC \rightarrow A, A \rightarrow D\}$
2) $F = \{BC \rightarrow D, B \rightarrow A\}$
3) $F = \{BC \rightarrow A, BC \rightarrow D\}$
4) $F = \{BC \rightarrow AD, A \rightarrow C\}$

1. **F = {BC→A, A→D} :** 2NF

Justification:

- 1NF: A relation is in 1NF if it has an ordered list of attributes, each attribute has atomic values, and each tuple is unique. This is a basic requirement for a relation to be a valid relation, and our scenario meets this requirement.
- 2NF: A relation is in 2NF if it is in 1NF and no non-prime attribute is partially dependent on any candidate key of the relation. A non-prime attribute of a relation is an attribute that is not a part of any candidate key of the relation. In our case, the

relation schema N is in 2NF because every non-prime attribute (A and D) is fully functionally dependent on the keys (BC and AD). There is no partial dependency.

- 3NF: A relation is in 3NF if it is in 2NF and no non-prime attribute of the relation schema is transitively dependent on the primary key. Transitive dependency means you can determine a non-key attribute from another non-key attribute. In our case, D is dependent on A, which is not a candidate key in the BC→A dependency. This is a transitive dependency, hence violating the 3NF rule so the highest normal form that the relation schema N can be in is 2NF.

2. **F = {BC→D, B→A} :** 1NF

<u>Justification:</u>

- 1NF: The relation schema N is in 1NF if it has an ordered list of attributes, each attribute has atomic values, and each tuple is unique. Our scenario meets these requirements, so it is in 1NF.
- 2NF: A relation schema is in 2NF if it is in 1NF and no non-prime attribute (an attribute that is not part of any candidate key) is partially dependent on any candidate key of the relation. In our case, the relation schema N violates 2NF because A, which is a non-prime attribute, is partially dependent on B, which is part of the candidate key BC. This partial dependency violates the rules of 2NF, so the highest normal form that the relation schema N can be in is 1NF.

3. **F = {BC→A, BC→D} :** BCNF

<u>Justification:</u>

- 1NF: The relation schema N is in 1NF if it has an ordered list of attributes, each attribute has atomic values, and each tuple is unique. This is a fundamental requirement for any relation, and our scenario fulfills this.
- 2NF: A relation is in 2NF if it is in 1NF and there is no partial dependency of any non-prime attribute on any candidate key. A non-prime attribute is an attribute that is not part of any candidate key. In our case, the relation schema N is in 2NF because each non-prime attribute (A and D) is fully functionally dependent on the candidate key (BC). There is no partial dependency, which would violate 2NF.
- 3NF: A relation is in 3NF if it is in 2NF and there is no transitive dependency of any non-prime attribute on any candidate key. Transitive dependency means that you can determine a non-key attribute from another non-key attribute. In this case, there are no such dependencies. Neither A nor D can be determined from any attribute other than BC. Therefore, this relation schema is also in 3NF.
- BCNF: A relation is in BCNF if for every one of its dependencies X → Y, at least one of the following conditions holds:
    - X is a super key for schema R,
    - Each attribute in Y - X is a prime attribute (an attribute that is part of some candidate key).
- In our case, for both functional dependencies (BC→A and BC→D), the left side (BC) is a superkey. Hence, the relation is in BCNF.
  So, given the functional dependencies F = {BC→A, BC→D}, the relation schema N is in BCNF, which is the highest normal form in this context.

4. **F = {BC→AD, A→C}:** 2NF

<u>Justification:</u>

- 1NF: The relation schema N is in 1NF if it has an ordered list of attributes, each attribute has atomic values, and each tuple is unique. This is a fundamental requirement for any relation, and our scenario fulfills this.
- 2NF: A relation is in 2NF if it is in 1NF and there is no partial dependency of any non-prime attribute on any candidate key. A non-prime attribute is an attribute that is not part of any candidate key. In our case, the relation schema N is in 2NF because each non-prime attribute (A, D and C) is fully functionally dependent on the candidate keys (BC and BA). There is no partial dependency, which would violate 2NF.
- 3NF: A relation is in 3NF if it is in 2NF and there is no transitive dependency of any non-prime attribute on any candidate key. Transitive dependency means that you can determine a non-key attribute from another non-key attribute. In this case, C can be determined from A (via the functional dependency A→C), which is part of another candidate key (BA). This transitive dependency violates 3NF. Therefore, this relation schema is not in 3NF so the highest normal form that the relation schema N can be in is 2NF.

Question 3. Functional Dependency [7 marks]

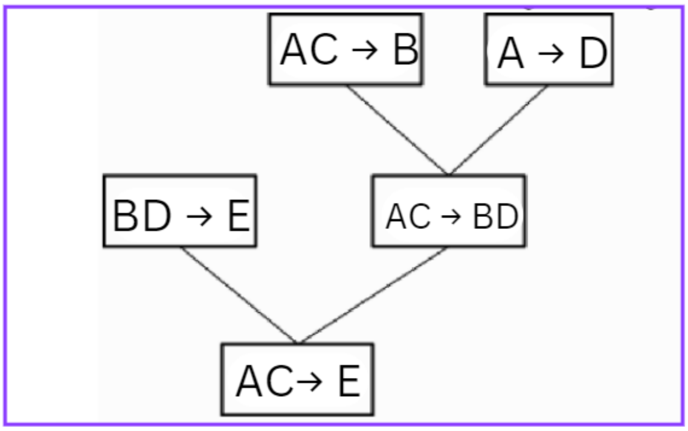
Consider a relation schema $N(R, F)$ where $R=\{A, B, C, D, E\}$ with the set of functional dependencies

$$F = \{AC \rightarrow B, BD \rightarrow E, A \rightarrow D\}$$

Show that $AC \rightarrow E$ can be inferred from F using Armstrong's inference rules.

1. Reflexivity Rule: If Y ⊆ X, then X → Y.
2. Augmentation Rule: If X → Y, then XZ → YZ for any Z.
3. Transitivity Rule: If X → Y and Y → Z, then X → Z.
4. Union Rule: If X → Y and X → Z, then X → YZ.
5. Decomposition Rule: If X → YZ, then X → Y and X → Z.
6. Pseudo Transitivity Rule: If X → Y and WY → Z, then WX → Z.

<u>Answer:</u>
Step 1: AC → B (Given)
Step 2: A → D (Given)
Step 3: AC → B and A → D then AC → BD (Using Union Rule)
Step 4: BD → E (Given)
Step 5: AC → BD and BD → E then AC → E (Using Pseudo Transitivity Rule)



Question 4. Minimal Cover of a set of Functional Dependencies [18 marks]

Consider the set of functional dependencies $F = \{C \to A, B \to AC, D \to A, AB \to D\}$. Compute a minimal cover of F. Show your process.

Working:
Step 1: Right Hand Side (RHS) of all FDs should be single attribute.
F = {C → A, B → AC, D → A, AB→ D}
F’= {C → A, B → A, B→C, D → A, AB→ D}

Step 2: Remove extraneous attributes.
Closure:
A+={A}
B+={A,B,C}
C+={A,C}
D+={A,D}
Therefore A is extraneous in AB→ D and is removed resulting in {B->D}

Selection:
C → A = redundant, covered by B → A
B → A = selected
B→ C = unique
D → A = redundant, covered by B → A
B→ D = unique

Answer:
F” = {B → A, B→C, B→ D} = {B → ACD}

Question 5. 3NF Normalization [25 marks]

Consider a relation schema $N(R, F)$ where $R = \{A, B, C, D\}$ and $F = \{C \to A, B \to D\}$. Perform the following tasks. Justify your answers.

1) [5 marks] Identify all keys for N. Show your process.
2) [5 marks] Identify the highest normal form (1NF, 2NF, 3NF, BCNF) that N satisfies.
3) [10 marks] If N is not in 3NF, compute a lossless transformation into a set of 3NF relation schemas using the Synthesis algorithm. Show your process.
4) [5 marks] Verify explicitly that your result has the lossless property, satisfies 3NF, and that all functional dependencies are preserved.

1) [5 marks] Identify all keys for N. Show your process.
Working:
A candidate key is a minimal set of attributes that can uniquely identify each tuple in the relation.

- C→A implies that {C} is a superkey since it can determine {A}
- B→D implies that {B} is a superkey since it can determine {D}

Neither {C} nor {B} is a minimal set of attributes, so we need to check if any combination of {C} and {B} is a candidate key.

If we combine {C} and {B}, we get {C, B}, and we can see that:
- {C, B}→{A, D} using the transitive rule, since C→A and B→D, and therefore {C, B} can determine both {A} and {D}
- {C, B} is minimal since neither {C} nor {B} can be removed without losing the ability to uniquely identify tuples in the relation

Therefore, {C, B} is the only candidate key for relation schema N.

<u>Answer:</u>
{C, B}

2) [5 marks] Identify the highest normal form (1NF, 2NF, 3NF, BCNF) that N satisfies
<u>Justification:</u>
- 1NF: The relation schema N is in 1NF if it has an ordered list of attributes, each attribute has atomic values, and each tuple is unique. Our scenario meets these requirements, so it is in 1NF.
- 2NF: A relation schema is in 2NF if it is in 1NF and no non-prime attribute (an attribute that is not part of any candidate key) is partially dependent on any candidate key of the relation. In our case, the relation schema N violates 2NF because A and D, which is a non-prime attribute, is partially dependent on C and B, which is part of the candidate key BC. This partial dependency violates the rules of 2NF, so the highest normal form that the relation schema N can be in is 1NF.

<u>Answer:</u>
1NF

3) [10 marks] If N is not in 3NF, compute a lossless transformation into a set of 3NF relation schemas using the Synthesis algorithm. Show your process
Working:
Step 1: Minimal cover
F = {C→A, B→D}

Step 2: Group FDs according to LHS and make schema for each group
B+={B,D}
C+={A,C}
S={({C,A},{C}), ({B,D},{B})}

Step 3: if none of relation scheme in S contains a key of (U,F), create scheme in S that contains key of (U,F)
S={({C,A},{C}), ({B,D},{B}), ({B,C},{BC})}

<u>Answer:</u>
S={({C,A},{C}), ({B,D},{B}), ({B,C},{BC})}

4) [5 marks] Verify explicitly that your result has the lossless property, satisfies 3NF, and that all functional dependencies are preserved.

Subankamo 300471606

Step 1: Compute the union of the projection of all relation schemas in S

- R1 = {C, A}
- R2 = {B, D}
- R3 = {B, C}

R = R1 ∪ R2 ∪ R3 = {B, C, A, D}

Step 2: Compute the natural join of all relation schemas in S
T = R1 ⋈ R2 ⋈ R3

Step 3: Verify if T is equal to R, if so, the decomposition is lossless.
T = {B, C, A, D}
R = {B, C, A, D}
Since T = R, the decomposition is lossless.

Step 3: verify that our result satisfies 3NF and that all functional dependencies are preserved
R1 = ({C,A},{C})

- C→A is preserved, and all attributes are functionally dependent on the key {C}, so it is in 3NF.

R2 = ({B,D},{B})

- B→D is preserved, and all attributes are functionally dependent on the key {B}, so it is in 3NF.

R3 = ({B,C},{BC})

- C→A is preserved since {C} is a subset of the key {B,C}.
- B→D is not preserved, but it is not relevant since {B,C} is a key and {D} is functionally dependent on {B} in R2.

All attributes are functionally dependent on the key {B,C}, so it is in 3NF.
Therefore, our result satisfies 3NF and preserves all functional dependencies.

Question 6. BCNF Normalization [30 marks]

Consider a relation schema N (R, F), where $R = \{A, B, C, D\}$ and $F = \{B \rightarrow D, A \rightarrow C, CD \rightarrow B, CD \rightarrow A\}$. Perform the following tasks. Justify your answers.

1) [5 marks] Identify all keys for N. Show process.
2) [4 marks] Identify the highest normal form (1NF, 2NF, 3NF, BCNF) that N satisfies.
3) [16 marks] If N is not in BCNF, compute a lossless decomposition into a set of BCNF relation schemas using the BCNF decomposition algorithm. Show your process.
4) [5 marks] Verify explicitly whether your result satisfies BCNF, and all functional dependencies are preserved.

1) [5 marks] Identify all keys for N. Show your process.
Working:
Closure:
A+ = {A,C}
B+={B,D}
C+={C}
D+={D}

CD+={CDAB}
AB+ = {A, B, C, D} (using A→C and B→D)
AC+ = {A, C, B, D} (using A→C and CD→B and CD→A)
AD+ = {A, D, C, B} (using A→C and CD→B and CD→A)
BC+ = {B, C, D, A} (using B→D and CD→B and CD→A)
BD+ = {B, D, A, C} (using B→D and A→C)
CD+ = {C, D, B, A} (using CD→B and CD→A)
From the list above, the closures of AB, AC, AD, BC, BD, and CD include all the attributes of the schema, so they all are keys.

<u>Superkey:</u>
superkey such that no proper subset of the superkey is itself a superkey

- AB can't be reduced
- AC can be reduced to A or C, but neither of these are superkeys
- AD can be reduced to A or D, but neither of these are superkeys
- BC can be reduced to B or C, but neither of these are superkeys
- BD can be reduced to B or D, but neither of these are superkeys
- CD can't be reduced

<u>Answer:</u>
AB and CD

2) [4 marks] Identify the highest normal form (1NF, 2NF, 3NF, BCNF) that N satisfies.
<u>Justification</u>

- 1NF: A relation is in 1NF if it has atomic values for each attribute in the relation schema. Since we're given only the schema and not any data, we can typically assume that the relation is in 1NF, as we don't have any information indicating otherwise.
- 2NF: A relation is in 2NF if it is in 1NF and each non-prime attribute is fully functionally dependent on each key of the relation. A non-prime attribute is an attribute not part of any candidate key. In this case, there are no non-prime attributes because every attribute is part of at least one candidate key (AB, CD). Therefore, N is in 2NF.
- 3NF: A relation is in 3NF if it is in 2NF and every non-key attribute is non-transitively dependent on every key of the relation. As per our scenario, since there are no non-prime attributes, the relation is also in 3NF.
- BCNF (Boyce-Codd Normal Form): A relation is in BCNF if it is in 3NF and for every non-trivial functional dependency X → Y, X is a superkey. Looking at the given set of functional dependencies:
    - B→D: B is not a superkey, so this violates BCNF.
    - A→C: A is not a superkey, so this violates BCNF.
    - CD→B: CD is a superkey.
    - CD→A: CD is a superkey.

  Therefore, the relation is not in BCNFso the highest normal form that the relation schema N can be in is 3NF.

<u>Answer:</u>

Subankamo 300471606

3NF

3) [16 marks] If N is not in BCNF, compute a lossless decomposition into a set of BCNF relation schemas using the BCNF decomposition algorithm. Show your process.
R = {A, B, C, D}
F = {B→D, A→C, CD→B, CD→A}
R is not in BCNF because there are FDS, B→D and A→C, of which the LHS is not a super key.

Working:
FD that violates BCNF:
B→D
A→C

Decompose the original relation:
starting with B→D, B→D violates the BCNF rule because A isn't a superkey.

- R1 includes all attributes from X union Y (in this case, B and D), and
- R2 includes attributes from the original relation N minus the attributes in Y that are not part of any key. So R2 will include all attributes from R minus D (because D isn't part of any key), i.e., {A, B, C}.

R1 = {B, D} with functional dependency B→D
R2 = {A, B, C} with functional dependencies B→D, A→C, and CD→B

>R1 in BCNF as B is super key for R1
>R2 A→C violates the BCNF

Decompose R2:
Here, A→C violates the BCNF rule because A isn't a superkey.

- R3 includes all attributes from X union Y (in this case, A and C), and
- R4 includes attributes from the original relation R2 minus the attributes in Y that are not part of any key. So R4 will include all attributes from R2 minus C (because C isn't part of any key), i.e., {A, B}.

R3 = {A, C} with functional dependency A→C
R4 = {A, B} with functional dependency B→D

>R3 in BCNF as A is super key for R3
>R4 does not contain functional dependencies that violate BCNF so BCNF

Therefore, we've achieved a lossless decomposition of N into a set of BCNF relation schemas: R1 = {B, D}, R3 = {A, C}, and R4 = {A, B}.

4) [5 marks] Verify explicitly whether your result satisfies BCNF, and all functional dependencies are preserved.

Subankamo 300471606

<u>BCNF</u>

BCNF requires that for each non-trivial functional dependency X → Y, X should be a superkey.

R1 = {B, D} with functional dependency B→D

- B is a superkey in R1, so R1 is in BCNF.

R3 = {A, C} with functional dependency A→C

- A is a superkey in R3, so R3 is in BCNF.

R4 = {A, B}

- R4 doesn't have any non-trivial functional dependencies, which is allowed in BCNF. Thus, R4 is in BCNF.

So, all decomposed relations are in BCNF.

<u>Functional Dependency Preservation</u>

The original set of functional dependencies F in relation N was {B→D, A→C, CD→B, CD→A}.

We need to check if these dependencies are still held in the decomposed relations.

- B→D is held in R1.
- A→C is held in R3.
- CD→B and CD→A are not directly represented in any single relation, but if we take the union of R3 and R4 (i.e., {A, B, C}), we can get CD→B and CD→A.

So, all functional dependencies from the original set are preserved in the decomposed relations.

Therefore, the result satisfies BCNF, and all functional dependencies are preserved.