Database Management Systems (COP 5725)

Spring 2017

Instructor: Dr. Markus Schneider
TAs: Keke Zhai, Man Mohan Devineni

Homework 4 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:

<table>
<thead>
<tr>
<th></th>
<th>Maximum</th>
<th>Received</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exercise 1</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Exercise 2</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Exercise 3</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>
Exercise 1 (Knowledge Questions) [40 points]

1. Is there any problem in the schema studentTranscript(sId, sname, sphone, saddr, courseId, courseGrade)? If yes, list two aspects of the problem. If no, argue, why the schema is fine. [4 points]

1) redundancy: for each course the student selected, the student information such as student name, student address has to be stored;
2) update anomalies: the student's phone number can be changed in one of its tuple but remains unchanged in another tuple (inconsistency);
3) insertion anomalies: a student's information cannot be inserted without course information;

(Any 2 of the above)

2. Please define the term “functional dependency”. [4 points]

Let R be the relation schema of a relation R, and let A, B ⊆ R. B is functionally dependent on A, written A -> B if, and only if, to each value in A exactly one value in B belongs:

A->B ⇔ ∀ t1, t2∈ R: t1[A]=t2[A]⇒ t1[B]=t2[B] for all possible relations R over R.

3. Please describe the 6 inference rules of the Armstrong axioms. What are the two criteria that we value these rules? [8 points]

1) reflexivity rule: Let B ⊆ A. Then always A -> B (special case: A -> A) holds.
2) augmentation rule: If A -> B holds, then also A ∪ C -> B ∪ C holds.
3) transitivity rule: If A -> B and B -> C holds, then also A -> C holds.
4) union rule: If A → B and A → C holds, then also A → B ∪ C holds.
5) decomposition rule: If A → B ∪ C holds, then also A → B and A → C holds.
6) pseudotransitivity rule: If A → B and B ∪ C → D holds, then also A ∪ C → D holds.

The two criteria are: soundness and completeness

4. Please give the algorithm of calculating the attribute closure A^+ given a set F of FDs. [4 points]

See lecture slides.
5. What are the steps of computing the canonical cover. Given the set \( F = \{ A \rightarrow B, B \rightarrow C, AC \rightarrow D \} \), use the steps learned to compute the canonical cover. List each step in detail and explain it. [8 points]

Algorithm: refer to the lecture slides

Step 1: AC \( \rightarrow \) D is replaced by A \( \rightarrow \) D, because C on the left side is extraneous (C is already functionally dependent from A by the first two FDs).

Step 2: Nothing to be done

Step 3: Nothing to be done

Step 4: A \( \rightarrow \) B and A \( \rightarrow \) D is replaced by A \( \rightarrow \) BD

We obtained \( F_c = \{ A \rightarrow BD, B \rightarrow C \} \)

6. Describe the two fundamental correctness criteria for the normalization. [4 points]

losslessness (lossless join decomposition): An arbitrary instance r(R) must be reconstructable from the instances r1(R1), ..., rn(Rn).

dependency preservation: All FDs which hold for schema R must be transferable to the schemas R1, ..., Rn and must be efficiently checkable.

7. What kind of functional dependency does 2NF eliminate from 1NF? What kind of functional dependency does 3NF eliminate from 2NF? [4 points]

The 2NF eliminates the partial FDs between key and non-key attributes. The 3NF eliminates the dependencies from non-key attributes.

8. Describe the synthesis algorithm for computing the decomposition on the basis of \( F \). [4 points]

Step 1: Determine a canonical cover \( F_c \) for \( F \) (i.e., left reduction of the FDs, right reduction of the remaining FDs, removal of FDs of the form \( A \rightarrow \emptyset \), union rule for identical left sides)

Step 2: for each FD \( A \rightarrow B \in F_c \):

+ create a relation schema \( R_A := A \cup B \)

+ assign the FDs \( F_A = \{ C \rightarrow D \in F_c \mid C \cup D \subseteq R_A \} \) to \( R_A \)

Step 3: If all schemas \( R_A \) created in step 2 do not contain a candidate key of the original schema R, additionally create a relation with the schema \( R_K = K \) and \( F_K = \emptyset \).
where $K$ is a candidate key of $R$.

Step 4: Eliminate schemas $R_A$ that are contained in another schema $R_{A'}$.

Exercise 2 (Functional Dependencies and Normal Forms) [30 points]

1. [5 points] Consider a relation schema $R(XYZ)$ with functional dependencies $XY \rightarrow Z$ and $Z \rightarrow X$. Can we conclude that $Y \rightarrow XZ$? If yes, please proof it. If no, please give a counter example.

No.

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Y</td>
<td>Z</td>
</tr>
<tr>
<td>A1</td>
<td>B1</td>
<td>C1</td>
</tr>
<tr>
<td>A2</td>
<td>B1</td>
<td>C2</td>
</tr>
</tbody>
</table>

For the table above, it satisfies the FDs that $XY \rightarrow Z$ and $Z \rightarrow X$. But it doesn’t satisfy the FD $Y \rightarrow XZ$ since $B_1$ is the same, but $A_1 C_1$ and $A_2 C_2$ are not the same.

2. [8 points] Given the relation schema $R = (A, B, C, D, E)$ and the canonical cover of its set of functional dependencies $F_c = \{ A \rightarrow BC \, CD \rightarrow E \, B \rightarrow D \, E \rightarrow A \}$. Compute a lossless join decomposition in Boyce-Codd Normal Form for $R$. Show your steps clearly.

1) $(A, B, C, D, E)$ is not in BCNF because $B \rightarrow D$ is not a trivial dependency and it is not a superkey for $(A, B, C, D, E)$.

2) $(A, B, C, D, E)$ decomposes by $B \rightarrow D$ into $(A, B, C, E)$ and $(B, D)$

3) We determine that $(B, D)$ is in BCNF because the nontrivial functional dependency $B \rightarrow D$ is given, so $B$ is a superkey for schema $(B, D)$.

4) We determine that $(A, B, C, E)$ is in BCNF because for the nontrivial functional dependencies given, $A \rightarrow BC$ and $E \rightarrow A$, both $A$ and $E$ are superkeys for the schema $(A, B, C, E)$, since $A \rightarrow ABCDE$ and $E \rightarrow ABCDE$.

5) Final lossless-join decomposition: $(A,B,C,E)$ and $(B,D)$
3. [4 points] Is this decomposition dependency-preserving? Why or why not?

No. For FD CD \(\rightarrow\) E, C E and D are not in the same table in the decomposition.

4. [6 points] Compute the Canonical Cover for \(F = \{A \rightarrow B, ABCD \rightarrow E, EF \rightarrow GH, ACDF \rightarrow EG\}\). List the steps in detail.

1) Left reduction: \(\{A \rightarrow B, ACD \rightarrow E, EF \rightarrow GH, ACDF \rightarrow EG\}\) since \(A \rightarrow B\)

2) Right reduction: \(\{A \rightarrow B, ACD \rightarrow E, EF \rightarrow GH, ACDF \rightarrow \emptyset\}\) since \(ACDF \rightarrow EG\) can be get from the first three FDs.

3) Remove the form \(A \rightarrow \emptyset\) : \(\{A \rightarrow B, ACD \rightarrow E, EF \rightarrow GH\}\)

5. [3 points] List all attribute sets from below that are candidate keys (if any) based on the above question.

\[
\begin{align*}
\text{AB} & \quad \text{ACDE} & \quad \text{ACDF} & \quad \text{CDG} \\
\end{align*}
\]

Answer: ACDF

6. [4 points] Is the answer you computed from Question 4 in 3NF? Is it in 2 NF? Why or why not?

It is not in 3NF. For \(A \rightarrow B\), it is not trivial. A is not superkey of F. B is not (part of) some candidate key of F.

It is in 2NF since all FDs is left reduced.

Exercise 3 (Functional Dependencies and Normal Forms) [30 points]

Consider the relation schema \(R(ABCDEFG)\) with the functional dependencies \(A \rightarrow B, BD \rightarrow CG\) and \(A \rightarrow F\) G\(\rightarrow E\).

1. [5 points] Using Armstrong’s axioms (6 inference rules), show that the given FD’s (functional dependencies) imply that \(ADF \rightarrow E\). For each step, indicate which axiom and other FD’s you’re using.

\[
\begin{align*}
(1) \text{AD}\rightarrow\text{BD} & \quad \text{Augmenting A}\rightarrow\text{B by D} \\
(2) \text{AD}\rightarrow\text{CG} & \quad \text{Transitivity of (1) and BD}\rightarrow\text{CG} \\
(3) \text{ADF}\rightarrow\text{CFG} & \quad \text{Augmenting (2) by F} \\
(4) \text{ADF}\rightarrow\text{ACDFG} & \quad \text{Augmenting (3) by AD} \\
(5) \text{ADF}\rightarrow\text{AFG} & \quad \text{Decomposition of (4)} \\
(6) \text{ADF}\rightarrow\text{E} & \quad \text{Transitivity of (5) and AFG}\rightarrow\text{E} \\
\end{align*}
\]
2. [5 points] List all candidate keys of R. Show each step in detail.

<table>
<thead>
<tr>
<th>Left</th>
<th>Both</th>
<th>Right</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADF</td>
<td>BG</td>
<td>CE</td>
</tr>
</tbody>
</table>

1) $ADF' = ABCDEFG$
So, ADF is the only candidate key.

3. [5 points] Suppose we project R onto $S(ADF)$. Give a minimal cover of FD’s that hold in S.

$ADF \rightarrow G$

4. [10 points] Let $R = ABCDE$, $R1 = AD$, $R2 = AB$, $R3 = BE$, $R4 = CDE$, and $R5 = AE$. Let the functional dependencies be: $A \rightarrow C$, $B \rightarrow D$, $C \rightarrow D$, $DE \rightarrow C$, $CE \rightarrow A$. Use Chase Test to determine if this is Lossless Join Decomposition.

$$A \rightarrow C$$

$$B \rightarrow D$$

$$C \rightarrow D$$

$$DE \rightarrow C$$
The third row has become equal to t. So it’s Lossless Join Decomposition.

5. [5 points] Is the FDs of the previous question equivalent to FDs \{A \rightarrow CD, \ DE \rightarrow A, \ BE \rightarrow C, \ B \rightarrow D\}?

1) \{A \rightarrow C, \ B \rightarrow D, \ C \rightarrow D, \ DE \rightarrow C, \ CE \rightarrow A\} can derive \{A \rightarrow CD, \ DE \rightarrow A, \ BE \rightarrow C, \ B \rightarrow D\} since:
- \(A \rightarrow CD\) is got from \(A \rightarrow C\) and \(C \rightarrow D\)
- \(DE \rightarrow A\) is got from \(DE \rightarrow C\) and \(CE \rightarrow A\), which \(DE^* = DECA\)
- \(BE \rightarrow C\) is got from \(B \rightarrow D\) and \(DE \rightarrow C\), which \(BE^* = BEDC\)
- \(B \rightarrow D\) is already got.

2) \{A \rightarrow CD, \ DE \rightarrow A, \ BE \rightarrow C, \ B \rightarrow D\} cannot derive \{A \rightarrow C, \ B \rightarrow D, \ C \rightarrow D, \ DE \rightarrow C, \ CE \rightarrow A\} since:
- \(C \rightarrow D\) cannot got from any of the FDs \{A \rightarrow CD, \ DE \rightarrow A, \ BE \rightarrow C, \ B \rightarrow D\}. 