CMPT 354
Assignment 3 Key

Total marks: 60 — 2000-1
Due: March 15, 2000 by 20:30 — Instructor: G. Louie

1. Relational Database Design.

   Given the relation schema R = (A, B, C, D, E, F, G, H) and the following set of functional dependencies:

   $$F = \{ \begin{array}{l} A \rightarrow B \\ ABCD \rightarrow E \\ EF \rightarrow G \\ EF \rightarrow H \\ ACDF \rightarrow EG \end{array} \}$$

a) (4 marks) Compute the canonical cover for F. (Note: If this question looks familiar to you, you may be experiencing a case of déjà vu) Show your steps *clearly* to get full marks!

   Note that this canonical cover question was used as a step-by-step example in class, so it is only worth 4 marks! However, it is required to do the remaining parts of the question. Start w. 4 marks and take away 1 mark for incorrect or missing steps. Use the algorithm outlined on page 209 of the text:

   1. Use the union rule to replace $EF \rightarrow G$ and $EF \rightarrow H$ with $EF \rightarrow GH$.

      $$F = \{ \begin{array}{l} A \rightarrow B \\ ABCD \rightarrow E \\ EF \rightarrow GH \\ ACDF \rightarrow EG \end{array} \}$$

   1. B is extraneous in $ABCD \rightarrow E$ because $B \in ABCD$ and $\{A \rightarrow B, ABCD \rightarrow E, EF \rightarrow GH, ACDF \rightarrow EG\}$ logically implies $\{A \rightarrow B, ACD \rightarrow E, EF \rightarrow GH, ACDF \rightarrow EG\}$.

      This is because every FD in the 1st set is found in the 2nd set except for $ACD \rightarrow E$. This FD can be derived using Armstrong's Axioms from $A \rightarrow B$ and $ABCD \rightarrow E$ via the pseudotransitivity rule ($\alpha = A$, $\beta = B$, $\gamma = ACD$, and $\delta = E$). So remove B from $ABCD \rightarrow E$.

      $$F = \{ \begin{array}{l} A \rightarrow B \\ ACD \rightarrow E \end{array}$$

EF $\rightarrow$ GH
ACDF $\rightarrow$ EG }

2. E is extraneous in ACDF $\rightarrow$ EG because E $\in$ EG and {A $\rightarrow$ B, ACD $\rightarrow$ E, EF $\rightarrow$ GH, ACDF $\rightarrow$ G} logically implies {A $\rightarrow$ B, ACD $\rightarrow$ E, EF $\rightarrow$ GH, ACDF $\rightarrow$ EG}.
   This is true because:

1.	A $\rightarrow$ B	given
2.	ACD $\rightarrow$ E	given
3.	EF $\rightarrow$ GH	given
4.	ACDF $\rightarrow$ EF	augment 2 w. F
5.	ACDF $\rightarrow$ E	decompose 4
6.	ACDF $\rightarrow$ G	given
7.	ACDF $\rightarrow$ EG	union 5 & 6

   So remove E from ACDF $\rightarrow$ EG
   F = { A $\rightarrow$ B
   ACD $\rightarrow$ E
   EF $\rightarrow$ GH
   ACDF $\rightarrow$ G }

3. G is extraneous in ACDF $\rightarrow$ G. Note that ACDF $\rightarrow$ G is already implied by ACD $\rightarrow$ E and EF $\rightarrow$ GH in F because of the following:

1.	ACD $\rightarrow$ E	given
2.	EF $\rightarrow$ GH	given
3.	ACDF $\rightarrow$ EF	augment 1 w. F
4.	ACDF $\rightarrow$ GH	transitivity of 2 & 3
5.	ACDF $\rightarrow$ G	decomposition of 4

   So we can remove ACDF $\rightarrow$ G from F since it is derived.

4. None of the remaining FD's in F have extraneous attributes so the canonical cover is:
   F_c = { A $\rightarrow$ B
   ACD $\rightarrow$ E
   EF $\rightarrow$ GH }

b) (6 marks) Decompose R into 3rd Normal Form.

Use the algorithm outlined on page 230 of the text:

1. For A $\rightarrow$ B, R1 = (A, B)
2. For ACD $\rightarrow$ E, R2 = (A, C, D, E)

3. For EF $\rightarrow$ GH, R3 = (E, F, G, H)
4. However, a candidate key computed for the universal relation R is ACDF.

1.	A $\rightarrow$ B	in F_c (see part a)
2.	ACD $\rightarrow$ BCD	augment 1 w. CD
3.	ACD $\rightarrow$ E	in F_c
4.	ACD $\rightarrow$ BCDE	union of 2 & 3
5.	ACDF $\rightarrow$ ABCDEF	augment 4 w. AF
6.	ACDF $\rightarrow$ EF	decompose 5
7.	EF $\rightarrow$ GH	in F_c
8.	ACDF $\rightarrow$ GH	transitivity of 6 & 7
9.	ACDF $\rightarrow$ ABCDEFGH	union of 5 & 8

Since none of the decomposed relations contain a candidate key for R, we have to add an additional R4 = (A, C, D, F). 3 marks

5. We end up with the following decomposition:
R1 = (A, B),
R2 = (A, C, D, E),
R3 = (E, F, G, H)
and R4 = (A, C, D, F)

Note that for efficiency, we can combine R2 and R4 into a single relation.

3 marks

c) (5 marks) Prove that your decomposition in part b) is a lossless join. Note: No marks will be given for stating that the algorithm used gives a lossless-join, dependency-preserving decomposition!

Use the definition from page 222 of the text:

A decomposition is a lossless join if, for all relations r on schema R that are legal under the given set of functional dependency constraints,

$r = \Pi_{R1}(r) \bowtie \Pi_{R2}(r) \bowtie \Pi_{R3}(r) \bowtie \Pi_{R4}(r)$

Note that the universal relation r is first decomposed into two smaller relations r_A and r_B. If the relation r_A is then further decomposed to r_C

and r_D and we can show that r_C and r_D is a lossless-join, we can recover the relation r_A and show that it is a lossless join. Then if we can show that r_B and r_A also form a lossless-join, then we can recover the universal relation r and the entire decomposition is a lossless join. Additional decompositions are shown to be lossless joins in the same manner.

To show that two relations r_A and r_B form a lossless join, we must show one of the following:

$r_A \cap r_B \rightarrow r_A$
$r_A \cap r_B \rightarrow r_B$

We first decompose the universal relation R into r_A = (EFGH) and r_B = (ABCDEF). $r_A \cap r_B$ is then EF. Since EF $\rightarrow$ GH is given, augmenting each side with EF gives EF $\rightarrow$ EFGH and therefore this decomposition is lossless.

Next we decompose ABCDEF into r_C = (ACDF) and r_D = (ABCDE). $r_C \cap r_D$ is then ACD. Since ACD $\rightarrow$ E, A $\rightarrow$ B is given, we can show ACD $\rightarrow$ ABCDE and therefore this decomposition is lossless.

Then we decompose ABCDE into r_E = (AB) and r_F = (ACDE). $r_E \cap r_F$ is A. Since A $\rightarrow$ B is given, we can show A $\rightarrow$ AB and therefore this decomposition is also lossless.

By showing that each individual decomposition is lossless, we show that the entire decomposition is lossless.

d) (5 marks) Show that your decomposition in part b) is dependency preserving. Note that you are not asked to formally *prove* why, just to show that it is so.

Based on page 223 of the text, one can indicate that all FD's in F_c can be tested in at least one relation in the decomposition (2 marks). So,
A $\rightarrow$ B can be tested in R4,
ACD $\rightarrow$ E can be tested in R2,
EF $\rightarrow$ GH can be tested in R3. 1 mark each

Thus, the decomposition is dependency preserving.

2. (10 marks) Give a lossless join decomposition into Fourth Normal Form for the relation S = (F, G, H, I, J) if the following set of multivalued dependencies hold:

$F \twoheadrightarrow GH$

$G \twoheadrightarrow HI$

$J \twoheadrightarrow FI$

Note the only superkey for S is (FGHIJ) because there are no FD's given, so the decomposition must contain only trivial multivalued dependencies (*i.e.* a multivalued dependency $\alpha \twoheadrightarrow \beta$ is trivial if $\beta \subseteq \alpha$ or $\alpha \cup \beta = R$).

The definition of 4NF states that a relational schema R is in 4NF with respect to a set D of functional and multivalued dependencies if for all dependencies in D^+ of the form $\alpha \twoheadrightarrow \beta$, where $\alpha \subseteq R$ and $\beta \subseteq R$, at least one of the following holds:

- $\alpha \twoheadrightarrow \beta$ is trivial (*i.e.* $\beta \subseteq \alpha$ or $\alpha \cup \beta = R$)
- α is a superkey for R (2 marks for definition)

Following the algorithm in Figure 7.12 of the text,

1. R is not in 4NF because neither of the two conditions are true.
2. $F \twoheadrightarrow GH$: $F \rightarrow FGHIJ$ is not in D+ and $F \cap GH = \emptyset$, so decompose R (1 mark)
3. R1 = (F, G, H) (1 mark)
   R2 = (F, I, J) (1 mark)
4. R1 is in 4NF because $F \twoheadrightarrow GH$ is a trivial MVD, (1 mark)
   R2 is in 4NF because $F \twoheadrightarrow IJ$ is a trivial MVD, so the decomposition ends. (2 marks)
5. We get S = {(F, G, H), (F, I, J)} (2 marks)

3. (15 marks) 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 to get full marks!

Using the algorithm to decompose a relation to BCNF from figure 7.6 in text:

1. result = {(A, B, C, D, E)}; done = false; (1 mark)

2. Note that we are given the canonical cover F_c in the question. This means that we can avoid computing the closure of F and just use F_c and Armstrong's axioms to determine if a given functional dependency is in F^+.

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

$A \rightarrow BC$	given
$A \rightarrow B$, $A \rightarrow C$	decomposition
$B \rightarrow D$, so $A \rightarrow D$	given, transitive
$A \rightarrow CD$	union
$CD \rightarrow E$, so $A \rightarrow E$	transitive
$A \rightarrow ABCDE$	union of above steps
$E \rightarrow A$, so $E \rightarrow ABCDE$	given, transitive
$CD \rightarrow E$, so $CD \rightarrow ABCDE$	transitive
$B \rightarrow D$, so $BC \rightarrow CD$	augmentation
$BC \rightarrow ABCDE$	transitive

Since BC is a candidate key, B cannot be a superkey. As soon as we find one functional dependency that does not meet the criteria for BCNF, the schema is not in BCNF. (3 marks for explanation and application of rules)

4. B $\rightarrow$ D holds on (A, B, C, D, E), (1 mark)
   B $\rightarrow$ ABCDE is not in F^+ (*i.e.* can't be computed using Armstrong's Axioms from the canonical cover F_c) (1 mark) and
   B $\cap$ D = empty set, so: (1 mark)

   result = {(A, B, C, D, E) – (A, B, C, D, E)} $\cup$ {(A, B, C, D, E) – D} $\cup$ (B, D)
   result = {empty set} $\cup$ (A, B, C, E) $\cup$ (B, D)
   result = {(A, B, C, E), (B, D)} (2 marks)

5. 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). (2 marks)

6. 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 from step 3. (2 marks).

4. (15 marks) Use the axioms for functional and multivalued dependencies to show the soundness of the difference rule.

   If $\alpha \twoheadrightarrow \beta$ holds and $\alpha \twoheadrightarrow \gamma$ holds, then $\alpha \twoheadrightarrow \beta - \gamma$ holds and $\alpha \twoheadrightarrow \gamma - \beta$ holds.

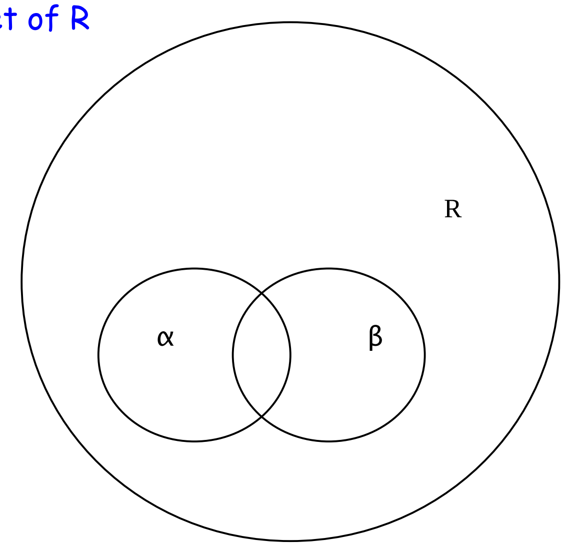
1.	$\alpha \twoheadrightarrow \beta$	given
2.	$\alpha \twoheadrightarrow R - \beta - \alpha$	complementation rule
3.	$\alpha \twoheadrightarrow R - \beta$	augment with $(\alpha - \beta)$*
4.	$\alpha \twoheadrightarrow \gamma$	given
5.	$\alpha \twoheadrightarrow (R - \beta)\gamma$	multivalued union rule
6.	$\alpha \twoheadrightarrow R - (\beta - \gamma)$	set theory
7.	$\alpha \twoheadrightarrow \beta - \gamma$	complement

8. $\alpha \rightarrow \gamma$ given
9. $\alpha \rightarrow R - \gamma - \alpha$ complementation rule
10. $\alpha \rightarrow R - \gamma$ augment with $(\gamma - \beta)$
11. $\alpha \rightarrow \beta$ given
12. $\alpha \rightarrow (R - \gamma)\beta$ multivalued union rule
13. $\alpha \rightarrow R - (\gamma - \beta)$ set theory
14. $\alpha \rightarrow \gamma - \beta$ complementation rule

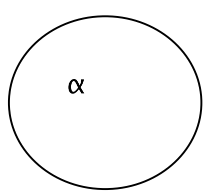
Students should have both parts of the proof. If only one side is correctly given and the other is left out, subtract 5 marks. Subtract 1 mark for each step along the way that is incorrect *i.e.* if proof is only correct up to step 3, then 4 marks are subtracted (7-3).

* Many students may find step 3 of this answer difficult to follow. The best way to envision the result is with a Venn diagram:

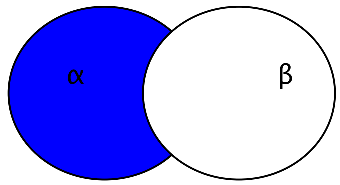
R is the set of all attributes
α is a subset of R
β is a subset of R



1. To augment in set theory means to add the members to the set if they are not members of the set. Nothing happens if they are already members. α looks like this:

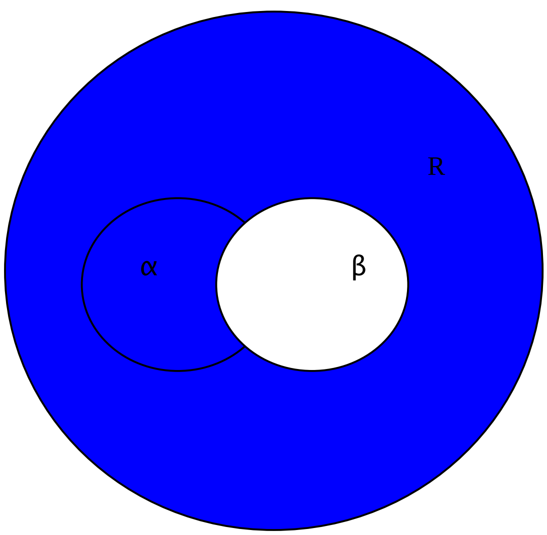


2. α - β looks like so (the filled in part):



3. Thus, augmenting α with α - β just ends up with α.

4. The second part of this is to augment (R - β - α) with (α - β). This basically adds the blue part from step 2 back to the original Venn diagram. If (R - β - α) were shaded on the Venn diagram, everything inside R and outside of (α - β) would be filled in. Thus, augmenting that picture with (α - β) from step 2 gives the resulting diagram:



5. Looking at the Venn diagram from 4, it appears that augmenting (R - β - α) with (α - β) just gives us (R - β). Combining both sides of the multivalued dependency, we get α ↠ (R - β).