Database Management Systems

(COP 5725)

Fall 2019

Instructor: Dr. Markus Schneider

TAs: Kyuseo Park

Homework 4

Name:	Weibin Sun
UFID:	59935801
Email Address:	Weibin.sun@ufl.edu

Pledge (Must be signed according to UF Honor Code)

On my honor, I have neither given nor received unauthorized aid in doing this assignment.

Weibin Sun

Signature

For scoring use only:

	Maximum	Received
Exercise 1	35	
Exercise 2	20	
Exercise 3	35	
Exercise 4	10	
Total	100	

Exercise 1 [35 points]

1. [5 points] Consider the relation schema *R* = (A, B, C, D, E, F) with the functional dependencies *FD* = {A→B, D→E, A→C}. Which of the following sets of attributes functionally determine E and which sets are the candidate key? If no candidate key found, compute it. Show each step.
   - AD：yes, no:
     AD+=ABCDE, which includes E, so E is functionally determined.
     It is not candidate key because AD+ doesn't include every attribute in R.

   - BCD: yes, no
     BCD+=BCDE, which includes E, so E is functionally determined.
     It is not candidate key because BCD+ doesn't include every attribute in R.

   - AC: no, no
     AC+=ABC, which doesn't include E, so E is not functionally determined.
     It is not candidate key because AC+ doesn't include every attribute in R.

   - CD: yes, no:
     CD+=CDE, which includes E, so E is functionally determined.
     It is not candidate key because CD+ doesn't include every attribute in R.

   - AF: no, no
     AF+=ABCF, which doesn't include E, so E is not functionally determined.
     It is not candidate key because AF+ doesn't include every attribute in R.

2. [5 points] Consider a relation schema R(X, Y, Z) with the functional dependencies XY→Z and Z→X. Can we conclude that Y→XZ holds? If yes, please argue why. If no, please argue why not by giving a counter example.

   No: the reason show as below:

X	Y	Z
A1	B1	Z1
A2	B1	Z2

   The diagram above shows that XY→Z and Z→X but doesn't satisfy Y→XZ.

3. [5 points] Consider the relation schema *R* = (A, B, C, D, E, F, G, H) with functional dependencies *FD* = {A→B, CH→A, B→E, BD→C, EG→H, DE→F}. Which of the following *FDs* is also guaranteed to be satisfied by *R*? Show each step.
   - ADG→CH: ADG+=ABCDEFGH which includes CH and is guaranteed to be satisfied by *R*
   - CGH→BF: CGH+=ACGH which doesn't include BF and is not guaranteed to be satisfied by *R*
   - BFG→AE: BFG+=BEFGH which doesn't include AE and is not guaranteed to be satisfied by *R*
   - ADE→CH: ADE+=ABCDEF which doesn't include CH and is not guaranteed to be satisfied by *R*

4. [5 points] Consider the relation schema R = (A, B, C, D, E, F, G, H, I, J) with functional dependencies FD = {B→E, E→FH, BCD→G, CD→A, A→J, I→BCDE, H→I}. Determine if B→J holds and list every candidate key. Show each step.

   Using the algorithm CalculateAttributeCloisure and set B+:B. In the loop we set Old_B+:=B and check all FDs whether they can contribute to B+.

   First we take B→E and check whether B⊆B+ holds. This is the case. Therefore we set B+:=B+∪E=BE(due to transitivity).

   Next we take E→FH, and using the same augment as before, we obtain B+:=B+∪FH=BEFH.

   H→I, so B+:=BEFHI.

   I→BCDE, so B+:=BDFHICD.

   CD→A, so B+:=BDFHICDA.

   BCD→G, so B+:=BDFHICDAG.

   A→J, so B+:=BDFHICDAGJ.

   Thus B→R, since J⊂R, so B→J holds.

   And B is minimality so that it is a candidate key.

   Same method as above, we can get:

   E+=ABCDEFGHIJ, which is candidate key

   BCD+=ABCDEFGHIJ, which is not smallest set, so it is not a candidate key.

   CD+=ACDJ, which is not candidate key

   C+=C, which is not candidate key

   A+=AJ, which is not candidate key

   I+=ABCDEFGHIJ, which is candidate key

   H+=ABCDEFGH, which is candidate key

   F+=F, which is not candidate key

   J+=J, which is not candidate key

   Candidate keys: B E I H

5. [15 points] We have a set of functional dependencies given as F = {A→B, B→C} for four attributes A, B, C, and D in a relation schema R. Write down all the functional dependencies in the closure F^+ of F and count them.

A+=ABC [7FDs]	AB+=ABC [7FDs]	ABC+=ABC [7FDs]	ABCD+=ABCD[15FDs]
B+=BC [3FDs]	AC+=ABC [7FDs]	ABD+=ABCD[15FDs]	
C+=C [1FD]	AD+=ABCD[15FDs]	ACD+=ABCD[15FDs]	
D+=D [1FD]	BC+=BC [3FDs]	BCD+=BCD [7FDs]	
	BD+=BCD [7FDs]		
	CD+=CD [3FDs]		

A+ = ABC: A→A, A→B, A→C, A→AB, A→AC, A→BC, A→ABC
B+ = BC: B→B, B→C, B→BC
C+ = C: C→C
D+ = D: D→D
AB+ = ABC: AB→A, AB→B, AB→C, AB→AB, AB→AC, AB→BC, AB→ABC
AC+ = ABC : AC→A, AC→B, AC→C, AC→AB, AC→AC, AC→BC, AC→ABC
AD+ = ABCD: AD→A, AD→B, AD→C, AD→D, AD→AB, AD→AC, AD→AD, AD→BC,AD→BD, AD→CD, AD→ABC, AD→ABD, AD→ACD, AD→BCD, AD→ABCD
BC+ = BC: BC→B, BC→C, BC→BC
BD+ = BCD: BD→B, BD→C, BD→D, BD→BC, BD→BD, BD→CD, BD→BCD
CD+ = CD: CD→C, CD→D, CD→CD
ABC+ = ABC: ABC→A, ABC→B, ABC→C, ABC→AB, ABC→AC, ABC→BC, ABC→ABC
ABD^+ = ABCD: ABD→A, ABD→B, ABD→C, ABD→D, ABD→AB, ABD→AC, ABD→AD, ABD→BC, ABD→BD, ABD→CD, ABD→ABC, ABD→ABD, ABD→ACD, ABD→BCD, ABD→ABCD
ACD^+ = ABCD : ACD→A, ACD→B, ACD→C, ACD→D, ACD→AB, ACD→AC, ACD→AD, ACD→BC, ACD→BD, ACD→CD, ACD→ABC, ACD→ABD, ACD→ACD, ACD→BCD, ACD→ABCD
BCD^+ = BCD: BCD→B, BCD→C, BCD→D, BCD→BC, BCD→BD, BCD→CD, BCD→BCD
$ABCD^+$ = ABCD: ABCD→A, ABCD→B, ABCD→C, ABCD→D, ABCD→AB, ABCD→AC, ABCD→AD, ABCD→BC, ABCD→BD, ABCD→CD, ABCD→ABC, ABCD→ABD, ABCD→ACD, ABCD→BCD, ABCD→ABCD

Exercise 2 [20 points]

1. [5 points] Consider the relation schema R = (A, B, C, D, E, F, G, H) with functional dependencies F = {A→C, AC→E, D→EH, F→G} and G = {A→BCE, AD→CFG, D→A, DE→GH, F→D}. Are the two sets F and G **equivalent**? Show each step.

   No

   F left-hand A, AC, D, F

To G, calculate follows:	check right-hand F:	
A+=ABCE	A→C, C⊆A+	holds
AC+=ABCE	AC→E, E⊆AC+	holds
D+=ABCDEFGH	D→EH, EH⊆D+	holds
F+=ABCDEFGH	F→G, G⊆F+	holds

   G left-hand A, AD, D, DE, F

To F, calculate follows:	check right-hand G	
A+=ACE	A→BCE, B⊈A+	not holds
AD+=ACDEH	AD→CFG, FG⊈AD+	not holds
D+=EH	D→A, A⊈D+	not holds
DE+=DEH	DE→GH, G⊈DE+	not holds
F+FG	F→D, D⊈F+	not holds

   So they are not equivalent.

2. [2.5 points each] Use the Armstrong axioms to prove the following deductions.
   (1) {X→Y, X∪Y→Z} ⇒ {X→Z}
   (2) {X→Z, Y→W} ⇒ {X∪Y→Z∪W}

   (1) Applying augmentation rule: X→Y=>X→XY
       Applying transitivity rule: X→XY∧XY→Z => X→Z
   (2) Applying augmentation rule: X→Y=>X∪Y→Z∪Y,
       Y→W=>X∪Y→X∪W
       Applying union rule: X∪Y→Z∪Y∪X∪W
       Applying decomposition rule: X∪Y→Z∪W

3. [5 points] Consider the relation schema R = (A, B, C, D, E) with the set of functional dependencies F = {A→BC, CD→E, B→D, E→A}. List all candidate keys of R by using the Armstrong's Axioms. Show each step.

   First step, check the single attributes
   For A:
   Applying the reflexivity rule: A→A
   Applying the decomposition rule: A→BC A→B, A→C
   Applying the transitivity rule: A→B B→D A→D
   Applying the union rule: A→C, A→D A→CD
   Applying the transitivity rule: A→CD CD→E A→E
   So A is a candidate key.

   Fro B:
   Applying the reflexivity rule: B→B
   Applying the union rule: B→B, B→D B→BD

   For C:
   Applying the reflexivity rule: C→C

   For D:

Applying the reflexivity rule: D→D
For E :
Applying the transitivity rule: E→A∧A→ABCDE=>E→ABCDE
So E is a candidate key.

Second step, check the set which have two attributes
Since A and E are candidate keys, then there is no need to check AB, AC, AD, AE, BE, CE, DE.

For BC:
Applying the decomposition rule: BC→BC=>BC→B, BC→C
Applying the transitivity rule: BC→B∧B→D=>BC→D
Applying the union rule: BC→C, BC→D=>BC→CD
Applying the transitivity rule: BC→CD∧CD→E=>BC→E
Applying the transitivity rule: BC→E∧E→ABCDE=>BC→ABCDE
So BC is a candidate key.

For BD:
Applying the reflexivity rule: BD→BD

For CD:
Applying the transitivity rule: CD→E∧E→ABCDE=>CD→ABCDE
So CD is a candidate key.

Third step: check the sets that have three, four and five attributes. Since A, E, BC, CD are candidate keys and all other sets we still need to check contain them. So, there is no need to check other setw.

Hence, the candidate keys are : A, E, BC, CD

4. [5 points] For a relation scheme R = (A, B, C, D, E, F) and a set of functional dependencies given as F = {A→B, A→C, CD→E, CD→F, B→E}, use Armstrong's Axioms rules to find one candidate key for R. Show each step.

   Step 1: check the attributes are not in the right-hand sides of F: AD.
   So the candidate keys must contain AD.
   Step 2: check the set AD
   Applying the reflexivity rule: AD→AD
   Applying the decomposition rule: AD→AD=> AD→A, AD→D
   Applying the transitivity rule: AD→A∧A→B=>AD→B
   Applying the transitivity rule: AD→A∧A→C=>AD→C
   Applying the union rule: AD→D, AD→C=>AD→CD
   Applying the transitivity rule: AD→CD∧CD→E=>AD→E
   Applying the transitivity rule: AD→CD∧CD→F=>AD→F
   Applying the union rule: AD→ABCDEF
   So AD is a candidate key.
   Hence, one candidate key of R is AD.

Exercise 3 [35 points]

1. [15 points] Find a minimal cover for the relation R = (A, B, C, D, E, F, G, H) with the set F = {A→B, ABCD→E, EF→GH, ACDF→EG} of functional dependencies. Show each step.
   Step 1: Fc:={A→B, ABCD→E, EF→GH, ACDF→EG}
   Step 2:
   ABCD→E, EF→GH and ACDF→EG have more than one attributes on their left-hand sides.

   For ABCD→E:
   To check whether A can be removed, we compute whether
   E⊆CalculateAttribuetClosure(Fc, BCD) holds
   This is not the case, since BCD+=BCD and E⊈BCD
   To check whether B can be removed, we compute whether
   E⊆CalculateAttribuetClosure(Fc, ACD) holds
   This is the case, since ACD+=ABCDE and E⊆ABCDE
   Hence, B can be removed, and we obtain Fc:={A→B, ACD→E, EF→GH, ACDF→EG}
   To check whether C can be removed, we compute whether
   E⊆CalculateAttribuetClosure(Fc, AD) holds
   This is not the case, since AD+=ABD and E⊈ABD
   To check whether D can be removed, we computer whether
   E⊆CalculateAttribuetClosure(Fc, AC) holds
   This is not the case since AC+=ABC and E⊈ABC
   For EF→GH:
   To check whether E can be removed, we compute whether
   GH⊆CalculateAttribuetClosure(Fc, F) holds
   This is not the case since F+=F and GH⊈F
   To check whether F can be removed, we computer whether
   GH⊆CalculateAttribuetClosure(Fc, E) holds
   This is not the case since E+=E and GH⊈E

   For ACDF→EG:
   To check whether A can be removed, we compute whether
   EG⊆CalculateAttribuetClosure(Fc, CDF) holds
   This is not the case since CDF+=CDF and EG⊈CDF
   To check whether C can be removed, we compute whether
   EG⊆CalculateAttribuetClosure(Fc, ADF) holds
   This is not the case since ADF+=ABDF and EG⊈ABDF
   To check whether D can be removed, we compute whether
   EG⊆CalculateAttribuetClosure(Fc, ACF) holds
   This is not the case since ACF+=ABCF and EG⊈ABCF
   To check whether F can be removed, we compute whether
   EG⊆CalculateAttribuetClosure(Fc, ACD) holds
   This is not the case since ACD+=ABCDE and EG⊈ABCDE

   Step 3:

   For A→B
   To check whether B can be removed, we check whether
   B⊆CalculateAttribuetClosure({A→∅, ACD→E,EF→GH,ACDF→EG},A) holds
   This is not the case since A+=A and B⊈A

For ACD→E
To check whether B can be removed, we check whether
E⊆CalculateAttribuetClosure({A→B,ACD→∅,EF→GH,ACDF→EG},ACD)
holds
This is not the case since ACD+=ABCD and E⊈ABCD

For EF→GH
To check whether G can be removed, we check whether
G⊆CalculateAttribuetClosure({A→B, ACD→E, EF→H, ACDF→EG},EF) holds
This is not the case since EF+=EFH and G⊈EFH
To check whether H can be removed, we check whether
H⊆CalculateAttribuetClosure({A→B, ACD→E, EF→G, ACDF→EG},EF) holds
This is not the case since EF+=EFH and H⊈EFH

For ACDF→EG
To check whether E can be removed, we check whether
E⊆CalculateAttribuetClosure({A→B,ACD→E,EF→GH,ACDF→G},ACDF)
holds
This is the case since ACDF+=ABCDEFGH and E⊆ABCDEFGH
Hence, E can be removed, and we obtain
Fc:={A→B, ACD→E, EF→GH, ACDF→G}

To check whether G can be removed, we check whether
G⊆CalculateAttribuetClosure({A→B,ACD→E,EF→GH,ACDF→∅},ACDF)
holds
This is the case since ACDF+=ABCDEFGH and G⊆ABCDEFGH
Hence, G can be removed, and we obtain
Fc:={A→B, ACD→E, EF→GH, ACDF→∅}

Step 4: we obtain Fc:={A→B, ACD→E, EF→GH}
Step 5: for standard form Fc:={A→B, ACD→E, EF→G,EF→H}
or nonstandard form Fc:={A→B, ACD→E, EF→GH}

2. [10 points] Find a minimal cover for the relation *R* = (A, B, C, D, E) with the set *F* = {A→BC, CD→E, B→D, E→A} of functional dependencies. Show each step.

Step 1: Fc:={A→BC, CD→E, B→D, E→A}
Step 2:
Only CD→E has more than one attributes on its left-hand side
To check whether C can be removed, we compute whether
E⊆CalculateAttribuetClosure({Fc, D) holds
This is not the case since D+=D and E⊈D

To check whether D can be removed, we compute whether
E⊆CalculateAttribuetClosure({Fc, C) holds
This is not the case since C+=C and E⊈C

Step 3:
For A→BC:
To check whether B can be removed, we check whether
B⊆CalculateAttribuetClosure({A→C,CD→E,B→D,E→A},A) holds

This is not the case since A+=AC and B⊈AC
To check whether C can be removed, we check whether
C⊆CalculateAttribuetClosure({A→B,CD→E,B→D,E→A},A) holds
This is not the case since A+=ABD and C⊈ABD

For CD→E
To check whether E can be removed, we check whether
B⊆CalculateAttribuetClosure({A→BC,CD→,B→D,E→A},CD) holds
This is not the case since CD+=CD and E⊈CD

For B→D
To check whether D can be removed, we check whether
D⊆CalculateAttribuetClosure({A→BC,CD→E,B→,E→A},A) holds
This is not the case since B+=B and D⊈B

For E→A
To check whether A can be removed, we check whether
A⊆CalculateAttribuetClosure({A→BC,CD→E,B→D,E→},A) holds
This is not the case since E+=E and A⊈E

Step 4: Nothing to do since there is no FD with an ∅ on its right-hand side
Step 5: For standard form Fc:={A→B, A→C,CD→E,B→D,E→A}
Or nonstandard form Fc:={A→BC, CD→E, B→D, E→A}

3. [10 points] Find a minimal cover for the relation *R* = (A, B, C, D, E, F) with the set *F* = {A→D, AC→DE, B→F, D→CE} of functional dependencies. Show each step.

Step 1: Fc:={A→D, AC→DE, B→F, D→CE}
Step 2:
Only AC→DE has more than one attribute on its left-hand side
To check whether A can be removed, we compute whether
DE ⊆CalculateAttribuetClosure(Fc, C) holds
This is not the case since C+=C and DE⊈C

To check whether C can be removed, we compute whether
DE ⊆CalculateAttribuetClosure(Fc, A) holds
This is the case since A+=ACDE and DE⊈ACDE
Hence C can be removed, and we obtain Fc:={A→D, A→DE, B→F, D→CE}

Step 3:
For A→D:
To check whether D can be removed, we compute whether
D ⊆CalculateAttribuetClosure({A→∅, A→DE, B→F, D→CE}, A) holds
This is the case since A+=ACDE and D⊆ACDE
Hence, D can be removed, and we obtain Fc:={A→∅, A→DE, B→F, D→CE}

For A→DE
To check whether D can be removed, we compute whether
D ⊆CalculateAttribuetClosure({A→∅, A→E, B→F, D→CE}, A) holds
This is not the case since A+=AE and D⊈AE

To check whether E can be removed, we compute whether
E ⊆CalculateAttribuetClosure({A→∅, A→D, B→F, D→CE}, B) holds

This is the case since A+=ACDE and E⊆ACDE
Hence E can be removed, and we obtain Fc:={A→∅, A→D, B→F, D→CE}

For B→F:
To check whether F can be removed, we compute whether
F ⊆CalculateAttribuetClosure({A→∅, A→D, B→∅, D→CE}, D) holds
This is not the case since B+=B and F⊈B

For D→CE:
To check whether C can be removed, we compute whether
C ⊆CalculateAttribuetClosure({A→∅, A→D, B→F, D→E}, D) holds
This is not the case since D+=DE and C⊈DE

To check whether E can be removed, we compute whether
E ⊆CalculateAttribuetClosure({A→∅, A→D, B→F, D→C}, D) holds
This is not the case since D+=CD and E⊈CD

Step 4: we obtain Fc:={ A→D, B→F, D→CE}
Step 5: For standard form: Fc:={A→D, B→F, D→C, D→E}
Or nonstandard form: Fc:={ A→D, B→F, D→CE}

Exercise 4 [10 points]

1. [5 points] Consider the relation schema *R* = (A, B, C, D, E, F) with a set of functional dependencies *F* = {CF→D, AE→F, D→A, AB→C}. List all candidate keys of *R* in a systematic manner (do not use the Armstrong's Axioms) and explain how you determine them. Show each step.

   Step 1:
   Check the attributes which are not in the right-hand side of F: BE
   So the candidate keys must contain BE
   Step 2:
   To check whether BE is a candidate key, we can compute BE+=BE.
   So BE is not a candidate key.
   Step 3:
   Check the subset which contain BE. Starting with the subsets with three attributes: ABE, BCE, BDE, BEF.
   For ABE+=ABCDEF=R, so ABE is a candidate key.
   For BCE+=BCE, so BCE is not a candidate key.
   For BDE+=ABCDEF, so BDE is a candidate key.
   For BEF+=BEF, so BEF is not a candidate key.

   So the candidate keys are: ABE, BDE
   Step 4:
   Check the subset with four attributes but does not contain ABE and BDE as subset: BCEF
   For BCEF: BCEF+=ABCDEF=R, so BCEF is a candidate key.
   Now we have candidate keys: ABE, BDE, BCEF
   Step 5:
   Check the subset with five and six attributes, they all have ABE, BDE and BCEF as subset. So there is no need to check them.

   All in all, the candidate keys are: ABE, BDE, BCEF

2. [5 points] Consider the relation schema *R*(A, B, C, D, E, F) with the functional dependencies *FD* = {D→C, CE→A, D→A, AE→D}. Determine all candidate keys of R in a systematic manner (do not use the Armstrong's Axioms) and explain how you determine them.

   Step1:
   Check the attributes that are not in the right-hand sides of F: BEF
   So the candidate keys must contain BEF
   Step2:
   To check whether BEF is a candidate key, we compute BEF+=BEF. So BEF is not a candidate key.
   Step3:
   Check the subset which contains BEF. Starting with the subset which has four attributes: ABEF, BCEF, BDEF.
   For ABEF+=ABCDEF=R, so ABEF is a candidate key.
   For BCEF+=ABCDEF=R, so BCEF is a candidate key.
   For BDEF+=ABCDEF=R, so BDEF is a candidate key.
   Now we have candidate keys: ABEF, BCEF, BDEF
   Step 4:
   For subset with five and six attributes, they all have ABEF, BCEF, BDEF as subsets.
   So there is no need to check them.
   So all the candidates of R are: ABEF, BCEF, BDEF.