DM505 uge 3

## Exercise 3.1.1 side 67

Functional dependencies for given attributes:

Social security 🡪 street address, city, state, Zip code, area code and phone number  
Area code, phone number 🡪 Social security number, street address, city, state, zip code

Keys: Social security number, {Area code, phone number}

## Exercise 3.2.1 side 79



1. All the non-trivial functional dependencies

AB🡪D  
C🡪A

1. All the keys of R

AB🡪ABCD  
CB🡪ABCD  
DB🡪ABCD

Skal indeholde b da ingen FD giver B.

1. Alle superkeys

En superkey skal kunne have alle attributterne som closure men behøver ikke at indeholde minimal mængde attributter selv. Altså er superkeys der ikke er keys viderebygninger af svaret fra b).

ABC  
ABD  
BCD  
ABCD

## Exercise 3.1 side 88

1. Venstre siden af alle FD skal være superkey

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Opgave | relation | FD’s | Key | Violation | Decomposition |
| A | R(A,B,C,D) | AB🡪C, C🡪D, D🡪A | BA, BC, BD | C🡪D, D🡪A, C🡪A, CD🡪A, AC🡪D  Keys skal indeholde B+en anden. | S(ABCD)  Vælg: C🡪D som har closure C🡪DA Så trækkes højresiden fra s og laves som en for sig selv:  S(BC, ACD)  ACD violater stadig med: D🡪A som er sin egen closure: S(CD, BC, DA) |
| B | R(A,B,C,D) | B🡪C, B🡪D | AB | B🡪C, B🡪D | S(ABCD)  B🡪C som er sin egen closure:  S(ABD, BC) B🡪D som er sin egen closure:  S(AB, BC, BD) |
| C | R(A,B,C,D) | AB🡪C, BC🡪D, CD🡪A, AD🡪B | AB, BC, CD, AD | Intet | Allerede I BCNF |
| D | R(A,B,C,D) | A🡪B, B🡪C, C🡪D, D🡪A | A, B, C, D | Intet | Allerede I BCNF |
| E | R(A,B,C,D, E) | AB🡪C, DE🡪C, B🡪D | ABE | AB🡪C, DE🡪C, B🡪D, AB🡪D, CB🡪D, BE🡪D | S(ABCDE)  AB🡪C med closure n – AB🡪CD  S(ABE, ABCD)  Violation på ABCD  B🡪D med sig selv som closure:  S(ABE, ABC, BD) |
| F | R(A,B,C,D, E) | AB🡪C, C🡪D, D🡪B, D🡪E | AB | C🡪D, D🡪B, D🡪E, C🡪B, C🡪E | S(ABCDE)  C🡪D med closure  C🡪BDE  S(AC, BCDE)  BCDE har violation D🡪B med closure  D🡪BE  S(AC, CD, DBE) |

## Exercise 3.2 side 89

Ja vi får samme resultat fordi uanset om vi vælger at decompose på A🡪C eller A🡪B vil vi få samme closure some r A🡪BC

S(ABCD)  
A🡪B  
S(ACD, AB)  
A🡪C  
S(AD, AB, AC)

S(ABCD)  
A🡪BC  
S(AD, ABC)

## Exercise 3.4.1 page 139

Først skriver vi op hvad vi ved I et skema

1. B🡪E

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | d1 | e1 |
| a2 | b | c | d | e2 |
| a | b3 | c | d3 | e |

Først indsættes B🡪E, hvor de tupler der er enige om b også må være enige om E:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | d1 | E1 |
| a2 | b | c | d | E1 |
| a | b3 | c | d3 | E |

Dernest bruges CE🡪A, hvor de tupler der er enige om C og E skal være enige om A:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | d1 | e1 |
| a | b | c | d | E1 |
| a | b3 | c | d3 | e |

Da ingen række bliver fuldendt uden unsubscripted variabler er dekompositionen ikke lossless.

1. AC🡪E og BC🡪D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | d1 | e1 |
| a2 | b | c | d | e2 |
| a | b3 | c | d3 | e |

AC🡪E

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | d1 | e |
| a2 | b | c | d | e2 |
| a | b3 | c | d3 | E |

BC🡪D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | d | E |
| a2 | b | c | d | e2 |
| a | b3 | c | d3 | E |

Her er øverste række uden subscripts angiver at der er sket et lossless join på R. Altås når vi rejoiner de seperate set fås hvad der var i R.

1. A🡪D, D🡪E B🡪D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | d1 | e1 |
| a2 | b | c | d | e2 |
| a | b3 | c | d3 | e |

A🡪D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | d1 | e1 |
| a2 | b | c | d | e2 |
| a | b3 | c | D1 | E |

B🡪D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | D | e1 |
| a2 | b | c | D | e2 |
| a | b3 | c | D | E |

D🡪E

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | D | e |
| a2 | b | c | D | E |
| a | b3 | c | d | E |

Øverste række uden subscripts og R er et lossless join

1. A🡪D, CD🡪E, E🡪D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | d1 | e1 |
| a2 | b | c | d | e2 |
| a | b3 | c | d3 | e |

A🡪D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | d1 | e1 |
| a2 | b | c | d | e2 |
| a | b3 | c | D1 | e |

CD🡪E

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | d1 | e |
| a2 | b | c | d | e2 |
| a | b3 | c | D1 | e |

E🡪D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | E |
| a | b | c | d1 | e |
| a2 | b | c | d | e2 |
| a | b3 | c | D1 | e |

Der er ikke sket et lossless join.

## Exercise 3.5.1 page 101

For at en relation er i 3NF kræver det at alle non-trivieller FD’s X🡪Y skal have X som superkey eller Y som et medlem af mindst en key.

Decomposing sker ved at lave en relation for hver FD(Fra minimal basis) X🡪Y og til sidst tilføje en ekstra der er en key for R(hvis en given relation ikke allerede er det).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Opgave | relation | FD’s | Key | Violation | Decomposition |
| A | R(A,B,C,D) | AB🡪C, C🡪D, D🡪A | BA, BC, BD | Ikke noget | Allerede I 3NF |
| B | R(A,B,C,D) | B🡪C, B🡪D | AB | B🡪C, B🡪D | R1=(BC) R2=(BD) ingen af disse er key så der laves: R3=(AB) |
| C | R(A,B,C,D) | AB🡪C, BC🡪D, CD🡪A, AD🡪B | AB, BC, CD, AD | Intet | Allerede I BCNF |
| D | R(A,B,C,D) | A🡪B, B🡪C, C🡪D, D🡪A | A, B, C, D | Intet | Allerede I BCNF |
| E | R(A,B,C,D, E) | AB🡪C, DE🡪C, B🡪D | ABE | AB🡪C, DE🡪C, B🡪D | R1(ABC)  R2(DEC)  R3(BD)  Ingen er key:  R4(ABE) |
| F | R(A,B,C,D, E) | AB🡪C, C🡪D, D🡪B, D🡪E | AB | C🡪D, D🡪E | R1(ABC)  R2(CD)  R3(DB)  R4(DE)  R1 er superkey |

## Exercise 5.2 page 101

1. What are the keys for Courses

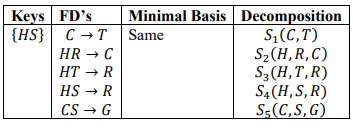
H og S optræder ikke på højresiden af nogen FD’s og skal derfor være medlem af alle keys. Closure på HS🡪RCTG

HS er altså key.

1. Er disse FD’s minimal basis?

Ja, jeg har ikke kunne finde nogle overflødige FD’s

1. Lav en tabel for hver regel og se om der er en superkey ellers tilføj en.



R1(CT)  
R2(HRC)  
R3(HTR)  
R4(HSR)  
R5(CSG)  
R4 er superkey, og de er alle i BCNF.

## Exercise 3.2.2

|  |  |  |  |
| --- | --- | --- | --- |
|  | I | II | III |
| A | A🡪C, A🡪D | AB🡪D, BC🡪A, CD 🡪B | A🡪C, A🡪D, B🡪D, B🡪A, C🡪A, C🡪B, D🡪B, D🡪C |
| B | A | AB, BC, CD, AD | A, B, C, D |
| c | Alt med A | Alle versioner større end dem |  |

## Exercise 3.1.2

Surely ID is a key by itself. However, we think that the attributes x, y, and z together form another key. The reason is that at no time can two molecules occupy the same point.  
ID → x y z  
ID → vx, vy, vz  
x y z → vx, vy, vz

## Exercise 2.4 page 80

Vis at nedenstående ikke er korrekt.

1. If A🡪B then B🡪A

Hvis vi har en persons CPR som A og personens højde som B kan vi lave højde som en functional dependency for A, men når vi har hans højde er det ikke nødvendigvis det navn.

R(A,B) A🡪B, men ikke B🡪A

1. If AB🡪C and A🡪C , then B🡪

R(A,B,C) A=CPR, B=navn(ikke unikt) og C=højde.  
med CPR og navn kan vi finde højde. Med kun CPR kan vi finde højde men da navn ikke er unikt kan vi ikke finden den specifikke højde med kun det.

1. If AB🡪C, then A🡪C or B🡪C

R(A,B,C) A=fornavn, B=efternavn, C=højde. Sammensætningen af navn og efternavn siges at være unikt altså fås højden ud fra det. Individuelt er navn og efternavn ikke unikt og man kan ikek konkluderer højde ud fra dem.

## Exercise 3.3.2 page 130

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Opgave | relation | FD’s | Key | Violation | Decomposition |
| A | R(A,B,C,D) | A🡪B  A🡪C | AD | A🡪B  A🡪C | S(ABCD)  Vælg: A🡪B Så trækkes højresiden fra s og laves som en for sig selv:  S(ACD, AB)  ACD violater stadig med: A🡪C S(AD, AB, AC) |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Opgave | relation | FD’s | Key | Violation | Decomposition |
| A | R(A,B,C,D) | A🡪B  A🡪C | AD | A🡪B  A🡪C | S(ABCD)  Vælg: A🡪BC Så trækkes højresiden fra s og laves som en for sig selv:  S(AD, ABC) |

If we decompose on a closure we handle all the violations of a specific left side in one decomposition where as if we only handle each violation 1 at a time we might have to decompose multiple times for the same left side.

## Exercise 3.5.3 page 142

|  |  |  |  |
| --- | --- | --- | --- |
| Keys | FD’s | Minimal basis | Decomposition |
| BIS | S🡪D  I🡪B  IS🡪Q  B🡪O | same | S1(SD)  S2(IB)  S1(ISQ)  S1(BO)  S1(BIS) |