**HOMEWORK 5**

**1.Is the corona table from BarBeerDrinker database in BCNF?**

**Yes/No? Either way you have to prove your answer (BCNF or not) using non-trivial functional dependencies. If you find one dependency f which violates BCNF, you do not have to find other functional dependencies. But you have to prove that f violates BCNF. Since there are so many attributes, you can group attributes, say into country attributes and corona stats and use notation X->Y where Y is a set of attributes.**

No, the corona table from BarBeerDrinker database is not in BCNF.

Proof:

Country Attributes (C) – iso\_code, continent, location

Country Properties (P) – population, population\_density, median\_age, aged\_65\_older, aged\_70\_older, cardiovasc\_death\_rate, diabetes\_prevalence, female\_smokers, male\_smokers, life\_expectancy, human\_development\_index, extreme\_poverty, gdp\_per\_capital, Hospital\_beds\_per\_thousand, handwashing\_facilities, total\_cases, new\_cases, new\_cases\_smoothed, total\_deaths, new\_deaths, new\_deaths\_smoothed, total\_cases\_per\_million, new\_cases\_per\_million, new\_cases\_smoothed\_per\_million, total\_deaths\_per\_million, new\_deaths\_per\_million, new\_deaths\_smoothed\_per\_million, stringency\_index, New\_tests, total\_tests, total\_tests\_per\_thousand, new\_tests\_per\_thousand, new\_tests\_smoothed, new\_tests\_smoothed\_per\_thousand, tests\_per\_case, positive\_rate, tests\_units

Dates (D) – date

I am assuming that attributes such as gdp\_per\_capital, population, and attributes in P, E, and M are not going to change by date since most of these numbers are recorded periodically. So, there will not be a relation b/w P, E, and M with D. But since the covid statistics does change by date and are thus reliant on what day in which it was recorded it will have a relation with the date attribute.

Functional Dependencies: {C->P, CD->V}

C+ = CP – not a super key

CD+ = CDVP – super key

Therefore, the corona table from BarBeerDrinker database is not in BCNF since attribute C is not a super key.

**2. Is original TripAdvisor table that we used in class in BCNF?**

**Yes/No? Either way you have to prove your answer (BCNF or not) using non-trivial functional dependencies. In this case, since there are fewer attributes than for corona data set, you should find as many functional dependencies as you can (as long as they are not implied by these you found already).This is requested even if you find a functional dependency which violates BCNF.**

No, the TripAdvisor table is not in BCNF

Restaurant (A), Rank (B), Score (C), User\_Name (D), Review\_Stars (E), Review\_Date (F), User\_Reviews (G), User\_Restaurant\_Reviews (H), User\_Helpful\_Votes (I)

I am assuming that for every Restaurant it has a unique Rank and that User\_Names are unique (i.e. there can’t be two users with the same User\_Name). The last assumption is because there are anomalies within TripAdvisor with the attribute User\_Names and its user data like the amount of User\_Reviews.

Functional Dependencies: {A->B, A->C, B->A, D->G, D->H, D->I, DF->E}

A+ = ABC – not a super key

B+ = ABC – not a super key

D+ = DGHI – not a super key

DF+ = DFEGHI – not a super key

Therefore, the TripAdvisor table is not in BCNF since attributes A, B, D, and DF are not super keys.

**3) Write SQL query which will check if functional dependency f= User -> User\_reviews is satisfied by the table TripAdvisor. The query should return “Yes” if f is satisfied, and “No” if f is violated in our Tripadvisor table instance.**

Select 'No' as Satisfies From tripadvisor

where EXISTS

( Select distinct ta1.User\_Name from tripadvisor ta1 where

(Select COUNT(\*) From tripadvisor ta2 where ta1.User\_Name = ta2.User\_Name and ta1.User\_reviews = ta2.User\_reviews)

<>

(Select COUNT(\*) From tripadvisor ta3 where ta1.User\_Name = ta3.User\_Name) )

UNION

Select 'Yes' as Satisfies From tripadvisor

where NOT EXISTS

( Select distinct ta1.User\_Name From tripadvisor ta1where

(Select COUNT(\*) From tripadvisor ta2 where ta1.User\_Name = ta2.User\_Name and ta1.User\_reviews = ta2.User\_reviews)

<>

(Select COUNT(\*) From tripadvisor ta3 where ta1.User\_Name = ta3.User\_Name) )

**In case the functional dependency f is violated, write an SQL query which will return at least two tuples in Trip Advisor table which violate f– in other words, counterexample (it takes two tuples to demonstrate that fd is violated)**

Username ‘Micha M’ has 4 different tuples with different number of user reviews

Select distinct User\_Name, User\_reviews from tripadvisor ta1

where User\_Name = 'Micha M'

This SQL query lists all usernames that violate the function dependency:

select distinct ta1.User\_Name from tripadvisor ta1

where (Select COUNT(\*) from tripadvisor ta2 where ta1.User\_Name = ta2.User\_Name and ta1.User\_reviews = ta2.User\_reviews) <>

(Select COUNT(\*) from tripadvisor ta3 where ta1.User\_Name = ta3.User\_Name)

**4\*. Write SQL query which will verify if decomposition of table Sells from BarBeerDrinker database into two tables Sells1[Bar, Beer] and Sells2[Beer, Price] is lossless.**

**Sells1 and Sells2 are simple projections of Sell table onto their respective attribute sets. The query should return “Lossless” in case the decomposition is in fact lossless and “Lossy” if it is not. This check is performed against specific instance of Sells table which we have in our practice db.**

**\*) Extra credit**

**THEORY**

**5) Let R=ABCDEGHK and F= {ABK→C, A→DG, B→K, K→ADH, H→GE}.**

**Is it in BCNF? Prove your answer.**

A relational Schema R is in BCNF if for every functional dependency X->A, X is a super key of R.

To determine if X is a super key, we will use the Closure Test. If the closure covers all attributes it implies that X is a super key.

ABK+ = ABCDEGK – super key

A+ = ADG – not a super key

B+ = ABCDEGK – super key

K+ = ADEGHK – not a super key

H+ = EGH – not a super key

Therefore, it is not in BCNF since A, K, and H are not super keys.

**6) Let R(ABCDEFG) be a relation and F = {A -> C, A-> D, B->F, E->F, F->G}**

1. **Show example of lossless join decomposition of R into three tables R1, R2 and R3 and demonstrate that this decomposition is lossless using chase algorithm**

F = {A -> CD, B -> F, E ->FG, F->G}

{A->C, A->D} => {A->CD}

A->CD violation of BCNF

R1 = ACD, F1 = {A->CD} – in BCNF

R2 = ABEFG, F2 = {F->G, E->FG, B->F} – violation pick (B -> F)

Decompose R2

R1 = BFG, F1 = {B->F, F->G} – violation pick (F->G)

R2 = ABE, F2 = {} – there are only trivial relations therefore it is in BCNF

Decompose R1

R1 = FG, F1 = {F->G} – in BCNF

R2 = BF, F2 = {B->F} – in BCNF

R1 = {A, C, D}, R2 = {A, B, E}, R3 = {F, G}, R4 = {B, F}

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G |
| R1(ACD) | A | B1 | C | D | E1 | F1 | G1 |
| R2(ABE) | A | B | C2 | D2 | E | F2 | G2 |
| R3(FG) | A3 | B3 | C3 | D3 | E3 | F | G |
| R4(BF) | A4 | B | C4 | D4 | E4 | F | G4 |

Apply A->CD

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G |
| R1(ACD) | A | B1 | C | D | E1 | F1 | G1 |
| R2(ABE) | A | B | ~~C2~~ C | ~~D2~~ D | E | F2 | G2 |
| R3(FG) | A3 | B3 | C3 | D3 | E3 | F | G |
| R4(BF) | A4 | B | C4 | D4 | E4 | F | G4 |

Apply B->F

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G |
| R1(ACD) | A | B1 | C | D | E1 | F1 | G1 |
| R2(ABE) | A | B | C | D | E | ~~F2~~ F | G2 |
| R3(FG) | A3 | B3 | C3 | D3 | E3 | F | G |
| R4(BF) | A4 | B | C4 | D4 | E4 | F | G4 |

Apply E->F

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G |
| R1(ACD) | A | B1 | C | D | E1 | F1 | G1 |
| R2(ABE) | A | B | C | D | E | F | G2 |
| R3(FG) | A3 | B3 | C3 | D3 | E3 | F | G |
| R4(BF) | A4 | B | C4 | D4 | E4 | F | G4 |

Apply F->G

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G |
| R1(ACD) | A | B1 | C | D | E1 | F1 | G1 |
| R2(ABE) | A | B | C | D | E | F | ~~G2~~ G |
| R3(FG) | A3 | B3 | C3 | D3 | E3 | F | G |
| R4(BF) | A4 | B | C4 | D4 | E4 | F | ~~G4~~ G |

Therefore, the decomposition R1 = {A, C, D}, R2 = {A, B, E}, R3 = {F, G}, R4 = {B, F} is a lossless join.

1. **Find one key for this scheme and prove that it is a key using closure algorithm and definition of a key.**

A+ = ACD

B+ = BFG

E+ = EFG

F+ = FG

AB+ = ABCDFG

AE+ = ACDEFG

AF+ = ACDFG

BE+ = BEFG

BF+ = BFG

EF+ = EFG

ABE+ = ABCDEFG – Super key

ABF+ = ABCDFG

ACF+ = ACDFG

AEF+ = ACDEFG

BEF+ … = All the following closures do not have attributes A thus are not a Super key therefore we do not need to check.

Further combinations will result in a higher number of attributes in a closure set therefore ABE is a minimal super key thus it is a key, and any superset of ABE is a super key.

ABE is a key for R

ABE+ = ABCDEFG – ABE is a super key since it covers all attributes of R but it is also a key since ABE is a minimal set of attributes that covers all attributes of R or a minimal super key thus all supersets of ABE are super keys:

ABE+ = ABCDEFG – Super key (key)

ABEF+ = ABCDEFG – Super key

ABEG+ = ABCDEFG – Super key

ABEFG+ = ABCDEFG – Super key

**7) Consider R = ABCDEG, with the set of dependencies F={AB → D, AB → C, AC → E, B → D, BE → A, E → G}. Suppose we have decomposed it into relations with set of attributes R1={ABD}, R2={ACE}, R3={ADEG}. Show using chase algorithm if this decomposition has lossless join property.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1(ABD) | A | B | C1 | D | E1 | G1 |
| R2(ACE) | A | B2 | C | D2 | E | G2 |
| R3(ADEG) | A | B3 | C3 | D | E | G |

Apply AB->D, AB->C, AC->E, B->D, BE->A

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1(ABD) | A | B | C1 | D | E1 | G1 |
| R2(ACE) | A | B2 | C | D2 | E | G2 |
| R3(ADEG) | A | B3 | C3 | D | E | G |

Apply E->G

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | G |
| R1(ABD) | A | B | C1 | D | E1 | G1 |
| R2(ACE) | A | B2 | C | D2 | E | ~~G2~~ G |
| R3(ADEG) | A | B3 | C3 | D | E | G |

Therefore, it is not a lossless join.

**8) Let R(ABCD) be a relation and F={A->B, C->D, BC->A}. Apply chase algorithm to test if decomposition of R onto R1(AB), R2(AC), R3(BCD) is lossless.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A | B | C | D |
| R1(AB) | a | b | c1 | d1 |
| R2(AC) | a | b2 | c | d2 |
| R3(BCD) | a3 | b | c | d |

Apply A->B

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A | B | C | D |
| R1(AB) | a | b | c1 | d1 |
| R2(AC) | a | ~~b2~~ b | c | d2 |
| R3(BCD) | a3 | b | c | d |

Apply C->D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A | B | C | D |
| R1(AB) | a | b | c1 | d1 |
| R2(AC) | a | b | c | ~~d2~~ d |
| R3(BCD) | a3 | b | c | d |

Apply BC-A

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A | B | C | D |
| R1(AB) | a | b | c1 | d1 |
| R2(AC) | a | b | c | d |
| R3(BCD) | ~~a3~~ a | b | c | d |

Therefore, it is a lossless join.

**9) Let R(ABCDE) be a relation and F={A→B, BC→E, and ED →A}. Decompose R into BCNF using BCNF decomposition algorithm. Remember that you need to compute projections of F to check if the decomposed tables are in BCNF.**

F = {A->B, BC->E, ED->AB}

ED -> A violation of BCNF

R1 = EDA, F1 = {ED->A} – in BCNF

R2 = BCDE, F2 = {BC->E, ED->B} – violation pick (BC->E)

Decompose R2

R1 = BCE, F1 = {BC->E} – in BCNF

R2 = BCD, F2 = {} – there are only trivial relations therefore it is in BCNF

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E |
| R1(EDA) | A | B1 | C1 | D | E |
| R2(BCE) | A2 | B | C | D2 | E |
| R3(BCD) | A3 | B | C | D | E3 |

Apply A->B

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E |
| R1(EDA) | A | B1 | C1 | D | E |
| R2(BCE) | A2 | B | C | D2 | E |
| R3(BCD) | A3 | B | C | D | E3 |

Apply BC->E

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E |
| R1(EDA) | A | B1 | C1 | D | E |
| R2(BCE) | A2 | B | C | D2 | E |
| R3(BCD) | A3 | B | C | D | ~~E3~~ E |

Apply ED->A

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E |
| R1(EDA) | A | B1 | C1 | D | E |
| R2(BCE) | A2 | B | C | D2 | E |
| R3(BCD) | ~~A3~~ A | B | C | D | E |

**10) Let R(ABCDEFGH) be a relation and F= {AB→E, C→D, D→E, FG→A}. Decompose R in BCNF using BCNF decomposition algorithm. Remember that you need to compute projections of F to check if the decomposed tables are in BCNF. Using Chase algorithm demonstrate if the decomposition you obtained is in fact lossless.**

F = {AB->E, C->DE, D->E, FG->A}

AB->E violation of BCNF

R1 = ABE, F1 = {AB->E} – in BCNF

R2 = ABCDFGH, F2 = {C->D, FG->A} – violation pick (FG->A)

Decompose R2

R1 = FGA, F1 = {FG->A} – in BCNF

R2 = BCDFGH, F2 = {C->D} – violation pick (C->D)

Decompose R2

R1 = CD, F1 = {C->D} – in BCNF

R2 = BCFGH, F2 = {} – there are only trivial relations therefore it is in BCNF

R1 = {ABE} R2 = {AFG} R3 = {CD} R4 = {BCFGH}

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H |
| R1(ABE) | A | B | C1 | D1 | E | F1 | G1 | H1 |
| R2(AFG) | A | B2 | C2 | D2 | E2 | F | G | H2 |
| R3(CD) | A3 | B3 | C | D | E3 | F3 | G3 | H3 |
| R4(BCFGH) | A4 | B | C | D4 | E4 | F | G | H |

Apply AB->E

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H |
| R1(ABE) | A | B | C1 | D1 | E | F1 | G1 | H1 |
| R2(AFG) | A | B2 | C2 | D2 | E2 | F | G | H2 |
| R3(CD) | A3 | B3 | C | D | E3 | F3 | G3 | H3 |
| R4(BCFGH) | A4 | B | C | D4 | E4 | F | G | H |

Apply C->D

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H |
| R1(ABE) | A | B | C1 | D1 | E | F1 | G1 | H1 |
| R2(AFG) | A | B2 | C2 | D2 | E2 | F | G | H2 |
| R3(CD) | A3 | B3 | C | D | E3 | F3 | G3 | H3 |
| R4(BCFGH) | A4 | B | C | ~~D4~~ D | E4 | F | G | H |

Apply D->E

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H |
| R1(ABE) | A | B | C1 | D1 | E | F1 | G1 | H1 |
| R2(AFG) | A | B2 | C2 | D2 | E2 | F | G | H2 |
| R3(CD) | A3 | B3 | C | D | ~~E3~~ E | F3 | G3 | H3 |
| R4(BCFGH) | A4 | B | C | D | ~~E4~~ E | F | G | H |

Apply FG->A

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H |
| R1(ABE) | A | B | C1 | D1 | E | F1 | G1 | H1 |
| R2(AFG) | A | B2 | C2 | D2 | E2 | F | G | H2 |
| R3(CD) | A3 | B3 | C | D | E | F3 | G3 | H3 |
| R4(BCFGH) | ~~A4~~ A | B | C | D | E | F | G | H |

Therefore, the decomposition R1 = {ABE} R2 = {AFG} R3 = {CD} R4 = {BCFGH} is a lossless join.