CSC343 Winter 2023 Assignment #3: Design and normalization

Due Thursday April 6 by 5:00pm

Shreyaansh Dadoo – Student No: 1006904101 – shreyaansh.dadoo@mail.utoronto.ca
Rishistav Ray – Student No: 1006884301 – rishi.ray@mail.utoronto.ca

Part 2: functional dependencies, normal forms, decompositions

1. Relation R_1 has attributes: D E F G H I J K and functional dependencies S_1:

$$S_1 = \{D \rightarrow F\ G,\ E \rightarrow H\ K,\ F \rightarrow E\ I\ J,\ F \rightarrow K\}$$

(a) Which of the dependencies violate BCNF?

D+ = DEFGHIJK (All attributes present, satisfies BCNF)
E+ = EHK (Does not satisfy BCNF)
F+ = FEHIJK (Does not satisfy BCNF)

Therefore, the following FDs don't satisfy BCNF:

- **E → H K**
- **F → E I J**
- **F → K**

(b) Use the BCNF decomposition algorithm to produce a lossless and redundancy-preventing decomposition of R_1 into a set of relations that are in BCNF. Make it clear to the reader which relations are in the final decomposition. There may well be more than one correct answer, depending on which FD you use to make a choice at each step. List your final relations so that (i) the attributes are in alphabetical order from left to right and (ii) the relations are in alphabetical order from top to bottom.

D	E	F	G	H	I	J	K	Closure	BCNF Test
✓								D^+ = DFGEIJK	All attributes are present hence BCNF satisfied
	✓							E^+ = EHK	Attributes missing, BCNF not satisfied

We can now decompose R_1:

R_2: **E** H K
R_3: D **E** F G I J

The LHS of our starting FD is E hence it will be the common attribute in both relations.

We will now check if R_2 satisfies BCNF:

E	H	K	Closure	BCNF Test
✓			$E^+ = EHK$	All attributes of R_2 are present hence BCNF satisfied.

We can stop checking other attributes since neither H nor K is present in the LHS of the FDs. Furthermore, EH and EK will yield weaker variations of the same superkey:

$E \rightarrow HK$, $EH \rightarrow K$, and $EK \rightarrow H$ are all equivalent superkeys that all satisfy BCNF

Next we will check if R_3 is in BCNF and decompose it further if necessary:

D	E	F	G	I	J	Closure	BCNF Test
✓						$D^+ = DEFGIJ$	All attributes are present hence BCNF satisfied
	✓					$E^+ = E$	-
		✓				$F^+ = EFIJ$	Missing attributes hence BCNF not satisfied

We can now decompose R_3:

R_4: E **F** I J
R_5: D **F** G

The LHS of our starting FD is F hence it will be the common attribute in both relations.

We will now check if R_4 satisfies BCNF:

E	F	I	J	Closure	BCNF Test
✓				$E^+ = E$	-
	✓			$F^+ = EFIJ$	BCNF satisfied.

We can ignore the closure of I and J as they do not appear in the LHS of any FD. Furthermore, because $E^+ = E$, any subsets of EIJ will only have EIJ in its closure and lastly any subsets containing F will yield weaker superkeys that satisfy BCNF.

Hence we can conclude R_4 satisfies BCNF.

We will now check if R_5 satisfies BCNF:

D	F	G	Closure	BCNF Test
✓			$D^+ = DFG$	All attributes of R_5 are present hence BCNF is satisfied.
	✓		$F^+ = F$	-

We don't need to check for G or subsets of G as its not present in the LHS of any FD. Hence it will yield weaker FDs. Since closure of F is just F, combinations of D and F will only yield weaker superkeys.

We can conclude that R_5 satisfies BCNF.

So the final decomposition is:

R_4: D F G
R_3: E F I J
R_2: E H K

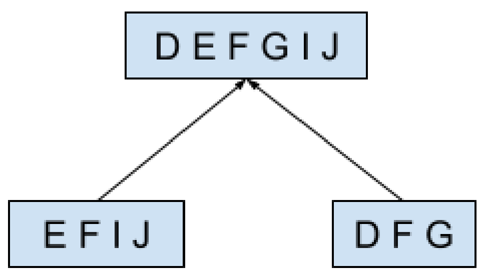
(c) Does your solution preserve dependencies? Explain how you know whether or not it does.

Yes, it does preserve all dependencies. We can look at the decomposed relations and their corresponding FDs:

R4(D F G): D → F G
R3(E F I J): F → E I J
R2(E H K): E → H K

We can see the FD: F→ K is missing.
However, if we compute the closure of F^+=EIFH**K** we can infer F→ K therefore all 4 FD are preserved.

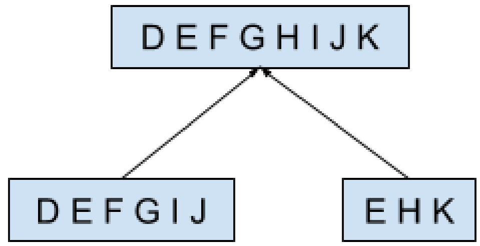
(d) Although a lossless join is guaranteed by using the BCNF algorithm to decompose the original relation, prove this is true using the Chase Test.



We can see that F has the same value in both tuples so we must enforce E, I and J to also have the same values as the other tuple.

D	E	F	G	I	J
d	e (enforced by F→EIJ)	f	g	i (enforced by F→EIJ)	j (enforced by F→EIJ)
4	e	f	5	i	j

The highlighted row contains all values d, e, f, g, i, j hence it passes Chase Test.



We can see that E has the same value in both tuples so we must enforce H and K to also have the same values as the other tuple.

D	E	F	G	H	I	J	K
d	e	f	g	h (enforced by E→HK)	i	j	k (enforced by E→HK)
3	e	4	5	h	6	7	k

The highlighted row contains all values d, e, f, g, h, i, j, k hence it passes Chase Test can we can lossless join the 3 smaller tables to the original table.

2. Relation R_2 has attributes JKLMNOPQ with functional dependencies S_2:

$$S_2 \ \{JLM \rightarrow N, K \rightarrow LM, KN \rightarrow JLO, M \rightarrow JKO, N \rightarrow JL\}$$

(a) Find a minimal basis for R_2. In your answer, put the FDs in order by making sure that:

i. attributes on each LHS and RHS are in alphabetical order

ii. your list of FDs is in alphabetical order by LHS; break ties by alphabetical order on their RHS

We split the RHS of all FDs and then check if the RHS can be determined from a subset of LHS using the closure of its subsets. Here are some closures that helped:

$K^+ = KLMJON$

$M^+ = MJKOLN$

Index	FD	Minimal FD
1	K→L	Yes
2	K→M	Yes
3	KN→J	No, K→J from K^+
(repeated)	KN→L	No, K→L from K^+
4	KN→O	No, K→O from K^+
5	JLM→N	No, M→N from M^+
6	M→J	Yes
7	M→K	Yes
8	M→O	Yes
9	N→J	Yes
10	N→L	Yes

We will now check if we have any redundant FDs and eliminate them:

Index	FD	Closure computed excluding Index FD	Status
1	K→L	K^+= KMJON**L**	Remove
2	K→M	K^+= KJO	Keep
3	K→J	K^+= KMON**J**K	Remove
4	K→O	K^+= KMNJKO**L**	Remove
5	M→N	M^+= JKM	Keep
6	M→J	M^+= MNKO**J**L	Remove
7	M→K	M^+= NO	Keep
8	M→O	M^+= KNJL	Keep
9	N→J	N^+= J	Keep
10	N→L	N^+=L	Keep

Minimal Basis: S' = { K→M, M→K, M→N, M→O, N→J, N→L }

(b) use your minimal basis from the previous part to find all keys for R_2.

Appears in LHS	Appears in RHS	Attributes
✓	✗	
✗	✓	J, L, O (In no keys)
✓	✓	K, M, N (Check)
✗	✗	P, Q (In every key)

Let’s check if K, M, and N are in keys

K	M	N	Closure (PQ is in every key)	Justification
✓			KPQ^+= KPQMNOJL	Key, because it's a minimal superkey (K cannot be a key by itself because PQ is in every key and PQ is not it any FDs so it cannot be a superkey.
	✓		MPQ^+= MPQKNOJL	Key, for the above reasons.
		✓	NPQ^+= NPQJL	Can’t be a key as attributes missing.
✓	✓		No need to check, its not a key	MPQ^+ is minimal to $KMPQ^+$
✓		✓	No need to check, its not a key	KPQ^+ is minimal to $KNPQ^+$
	✓	✓	No need to check, its not a key	MPQ^+ is minimal to $MNPQ^+$
✓	✓	✓	No need to check, its not a key	KPQ^+, MPQ^+ are minimal to $KMNPQ^+$

KPQ and MPQ are all the keys

(c) Use the 3NF synthesis algorithm to produce a lossless and dependency-preserving decomposition of R_2 into a collection of relations that are in 3NF. Be sure to combine all FDs with the same LHS to create a single relation. If you have a relation whose attributes are a subset of another relation, remove the relation with fewer attributes.

Combining the FDs in the Minimal Basis we get S'' where

S'' = { $K \rightarrow M$, $M \rightarrow NKO$, $N \rightarrow JL$}

We will make a relation from each FD:

$R_1(KM)$, $R_2(KMNO)$, $R_3(JLN)$, $R_4(KPQ)$

We can remove R_1 since its a subset of attributes of R_2.

Hence the final Relations are:

R_2(KMNO), R_3(JLN), R_4(KPQ)