A5: FDs and BCNF

Andrew Choi

December 6, 2022

THIS WORK WAS MY OWN WORK. IT WAS WRITTEN WITHOUT CONSULTING WORK WRITTEN BY OTHER STUDENTS OR COPIED FROM ONLINE RESOURCES.
Andrew Choi

1 Functional Dependencies and Boyce-Codd Normal Form

Relation R:

```
R(saleID, saleTime, gameTitle, gamePublisher, publisherCutPercent,
    quantity, price, customerID, address, creditCardNo)
```

Functional Dependencies S:

- gameTitle → price
- gameTitle → gamePublisher
- gamePublisher → publisherCutPercent
- customerID → address
- customerID → creditcardNo
- saleID → saleTime, gameTitle, quantity, price, customerID

We will also use the following symbols to simplify the work:
A = gameTitle, B = price, C = gamePublisher, D = publisherCutPercent, E = customerID, F = address, G = creditCardNo, H = saleID, I = saleTime, J = quantity

Therefore, the simplified FD's in S are:

- $A \rightarrow B$

- A → C
- C → D
- E → F
- E → G
- H → I, A, J, B, E

1. Which of the functional dependencies in S violate BCNF? Justify your answer.

   In order to see which functional dependencies in S violate BCNF, we need to find the closure of the LHS attribute of each FD in S:
   A+ = ABCD, so A is not a superkey and A → B violates BCNF.
   Similarly, A → C also violates BCNF.
   C+ = CD, so C is not a superkey and C → D violates BCNF.
   E+ = EFG, so E is not a superkey and E → F violates BCNF.
   Similarly, E → G also violates BCNF.
   Finally, H+ = HIAJBECDFG, so H is a superkey and H → I, A, J, B, E does not violate BCNF.

   Answer: Therefore, every functional dependency, except saleID → saleTime, gameTitle, quantity, price, customerID violates BCNF.

2. Employ the BCNF decomposition algorithm to obtain a lossless decomposition of R into a collection of relations that are in BCNF. Make sure it is clear which relations are in the final decomposition and project the dependencies onto each relation in that final decomposition. Because there are choice points in the algorithm, there may be more than one correct answer.

   For the purposes of simplifying the work, we will be using the symbols when performing the BCNF decomposition.
   First, let's consider FD A → B. Decomposing R with A → B, we get A+ = ABCD, so R1 = ABCD and R2 = AEFGHIJ.
   Now project the FDs onto R1 = ABCD:

A	B	C	D	closure	FDs
✓				A+ = ABCD	A → BCD
	✓			B+ = B	nothing
		✓		C+ = CD	C → D; violates BCNF, therefore decompose R1 further.

Next, decompose R1 using FD C → D. This yields two relations: R3 = CD and R4 = ABC. Project FDs onto R3 = CD:

C	D	closure	FDs
✓		C+ = CD	C → D
	✓	D+ = D	nothing
✓	✓	CD+ = CD	nothing

R3 = (gamePublisher, publisherCutPercent) satisfies BCNF. Next, project FDs onto R4 = ABC:

A	B	C	closure	FDs
✓			A+ = ABCD	A → BC
	✓		B+ = B	nothing
		✓	C+ = CD	nothing
✓	✓		AB+ = ABCD	weaker version of FD A → BC
	✓	✓	BC+ = BCD	nothing
✓		✓	AC+ = ABCD	weaker version of FD A → BC

Therefore, R4 = (gameTitle, price, gamePublisher) satisfies BCNF. Now, let's return to R2 = AEFGHIJ.

A	E	F	G	H	I	J	closure	FDs
✓							A+ = ABCD	nothing
	✓						E+ = EFG	E → FG; violates BCNF, therefore decompose further.

Decomposing R2 using E → FG, we get R5 = EFG and R6 = AEHIJ.
Next, project FDs onto R5:

E	F	G	closure	FDs
✓			E+ = EFG	E → FG
	✓		F+ = F	nothing
		✓	G+ = G	nothing
✓	✓		EF+ = EFG	weaker version of FD E → FG
	✓	✓	FG+ = FG	nothing
✓		✓	EG+ = EFG	weaker version of FD E → FG

Therefore, R5 = (customerID, address, creditCardNo) satisfies BCNF. Next, project FDs onto R6 = AEHIJ:

A	E	H	I	J	closure	FDs
✓					A+ = ABCD	nothing
	✓				E+ = EFG	nothing
		✓			H+ = ABCDEFGHIJ	H → AEIJ
			✓		I+ = I	nothing
				✓	J+ = J	nothing
✓	✓				AE+ = ABCDE	nothing
✓			✓		AI+ = ABCDI	nothing
✓				✓	AJ+ = ABCDJ	nothing
	✓		✓		EI+ = EI	nothing
	✓			✓	EJ+ = EJ	nothing
			✓	✓	IJ+ = IJ	nothing
supersets of H					irrelevant	can only generate weaker FDs than already given

Therefore, R6 = (gameTitle, customerID, saleID, saleTime, quantity) also satisfies BCNF.

Therefore the final decomposition is:

(a) R3 = (gamePublisher, publisherCutPercent) with FD gamePublisher → publisherCutPercent

(b) R4 = (gameTitle, price, gamePublisher) with FD gameTitle → price, gamePublisher

(c) R5 = (customerID, address, creditCardNo) with FD customerID → address, creditcardNo

(d) R6 = (gameTitle, customerID, saleID, saleTime, quantity) with FD saleID → gameTitle, customerID, saleTime, quantity

2 FDs and Candidate Keys

(a) Given the above instance of this relation, which of the following FDs may hold in this relation? If a FD cannot hold, explain why by specifying exactly the tuples that cause the violation:

Tuple	A	B	C
1	10	b1	c1
2	10	b2	c2
3	11	b4	c1
4	12	b3	c4
5	13	b1	c1
6	14	b3	c4

1. A → B: Tuples 1 and 2 conflict, as the value 10 in A leads to both b1 and b2 in B.
2. B → C: ✓ may hold
3. C → B: Tuples 1 and 3 conflict, as c1 leads to both b1 and b4.
4. B → A: Tuples 1 and 5 conflict, as b1 leads to both 10 and 13.
5. C → A: Tuples 1,3, and 5 conflict, as c1 leads to 10, 11, and 13.

Answer: Only item 2, FD B → C holds.

(b) Does the above relation have a potential candidate key? If yes, what is it? If it does not, why not?

Answer: No, as there is not a single attribute or combination of attributes that would serve as a superkey in this relation with the FDs given.

3 A decomposition that fails to perserve dependencies.

Create small instances of R1 and R2 that satisfy their own FDs, but when natural-joined together, violate one of the original FDs. You can optionally use the empty tables created in the sample Latex template file 'A5LatexSamples.tex' to answer this question.

Consider two relations, R1(theatre, city) and R2(theatre, movie), where:

theatre	city
AMC	Dallas
Cinemark	Dallas

theatre	movie
AMC	Black Panther
Cinemark	Black Panther

Then, if R1 ⋈ R2:

movie	theatre	city
Black Panther	AMC	Dallas
Black Panther	Cinemark	Dallas

However, since Dallas and Black Panther both lead to AMC and Cinemark, this violates the original functional dependency of movie, city → theatre. In their separate tables, they satisfy the FD's separately, and therefore, this is one example of a decomposition that fails to preserve dependencies.