Recognizing Normal forms

There are 4 normal forms that we are concerned with in this class, 1NF, 2NF, 3NF, and BCNF. There are others – but we will save that for a future class.

BCNF.

For a relation to be in BCNF with respect to F (a set of Functional Dependencies), for each FD in F where X -> A, one of the two must be true.

- It is a trivial functional dependency of the form A -> A
- X is a superkey of R.

So let's look at a schema R = (ABCDE) and F = {AB -> CDE, DE-> ABC}.
First we compute the attribute closure based on F.
A+ = A
B+ = B
C+ = C
D+ = D
E+ = E
AB+ = ABCDE
DE+ = ABCDE

Given this, we know the candidate keys are AB and DE. Now we inspect each of the FDs in F to see if it violates BCNF.

The first FD is AB -> CDE. Is AB a key for R? Yes, so this does not violate BCNF.
The second FD is DE -> ABC. Is DE a key for R? Yes, so this does not violate BCNF.

The relation is in BCNF.

3NF

3NF has the same requirements as BCNF, with one additional

- It is a trivial functional dependency of the form A -> A or
- X is a superkey of R. or
- A is a part of a key for R

Let's look at R = (ABCDE) and F = {AB -> CDE, C -> B}
First we compute the attribute closure based on F.
A+ = A
B+ = B
C+ = BC
D+ = D
E+ = E
AB+=ABCDE
AC+=ABCDE

Now we look at the FDs:
AB -> CDE – AB is a key – check (Still in BCNF)
C -> B, B is a part of a key for R – check (Not in BCNF, but in 3NF)

The relation is in 3NF.

2NF

For 2NF, we go back to our phrase – every non-key attribute depends on the whole key.

The point here being that if you have a key (AB) and a functional dependency B -> C, the relation is not in 2NF. Every non-key value must depend on a whole key.

1NF

For 1NF, we go back to our phrase – every non-key attribute depends on the key. This says there must be a key for the relation – which says that every attribute in the relation is atomic (i.e. not multivalued)