**Higher Normal Forms**

**BCNF**

In most cases, a relation that is 3NF is also BCNF – but this is not always the case. Here is an example of a relation that is 3NF, but not BCNF. This example is a slight modification of one that was on the stackoverflow.com site at <http://stackoverflow.com/questions/8437957/difference-between-3nf-and-bcnf-in-simple-terms-must-be-able-to-explain-to-an-8> on January 20, 2013.

Suppose we have a table to store pizza orders. A pizza must have exactly three topping types:

* one type of cheese
* one type of meat
* one type of vegetable

For each pizza ordered, we will store the order id, the topping selected, and the topping type. Since each pizza has exactly one of each topping type, we will use the order id and the topping type as a composite primary key.

Assume two pizzas have been ordered, resulting in the following values being stored in the **PizzaOrder** table:

**PizzaOrder**

**OrderID Topping ToppingType**

**-------- ---------- -------------**

**1 mozzarella cheese**

**1 pepperoni meat**

**1 olives vegetable**

**2 mozzarella meat**

**2 sausage cheese**

**2 peppers vegetable**

But wait a second, mozzarella can't be both a cheese and a meat! And sausage isn't a cheese! This table is 3NF, but it still has problems. We need to prevent these sorts of mistakes, to make mozzarella always be cheese and to be sure that sausage is always listed as a meat. Let’s take a closer look at this table.

The table is in third normal form. We selected**(OrderID, ToppingType)** as the primary key, and that is a reasonable and correct choice. However, we could just as well have chosen **(OrderID, Topping)** as the primary key. In other words, we have two *candidate keys* for this table, and we arbitrarily selected one of them as the primary key. So we have two candidate keys, both are composite, and they have overlapping attributes (both contain the attribute **OrderID**).

Here are the functional dependencies for this relation:

**OrderID, ToppingType 🡪 Topping**

**OrderID, Topping 🡪 ToppingType**

**Topping 🡪 ToppingType**

The functional dependency **Topping 🡪 ToppingType** does not violate the definition of 2NF or 3NF, because **Topping** is a component of the candidate key **(OrderID, Topping)**, so it is not a non-key attribute. However, it does violate the definition of BCNF, because the left side of the functional dependency **Topping 🡪 ToppingType** is not a candidate key.

In BCNF, only candidate keys can appear on the left side of a functional dependency.

To fix this problem, we split the original table into two tables, in much the same way that we would for non-key attributes.

**OrderID Topping**

**-------- ----------**

**1 mozzarella**

**1 pepperoni**

**1 olives**

**2 mozzarella**

**2 sausage**

**2 peppers**

**Topping ToppingType**

**------- -------------**

**mozzarella cheese**

**pepperoni meat**

**olives vegetable**

**sausage meat**

**peppers vegetable**

The primary key (and the only candidate key) for the first table is **(OrderID, Topping)**. The primary key (and the only candidate key) for the second table is **(Topping)**. Both tables are in 3NF and also in BCNF.

**4NF and 5NF**

See <http://www.bkent.net/Doc/simple5.htm> for more information about 4NF and 5NF.