Identifying 1st, 2nd, and 3rd Normal Forms

For each relation:
Every non-key attribute
depends on the key (1st normal form)
the whole key (2nd normal form)
and nothing but the key (3rd normal form)
so help me Codd.

Following up:

A relation in first normal form says that every table has a primary key whose value functionally determines the single value of every other attribute in the table. Every table must meet the definition of a true relation - primary key and no multi-valued attributes.

Every non-key attribute depends on the key.

A relation is in second normal form if it is in 1NF and every non-key attribute is fully functionally dependent on the primary key. (i.e. 2NF = 1NF + no partial dependencies).

The whole key.

A relation is in third normal form if it is in 2NF and there are no dependencies between non-key attributes. (i.e. 2NF + no transitive dependencies).

And nothing but the key.

So let's work through a few of these.

Assuming multiple stores can carry the same item, is the following relation in 1NF?

(<u>item #</u>, store #, quantity, store loc, store mgr)

You are correct - it is no. The fact that having item # as the primary key allows a multi-valued set of attributes (your 1 to many from item to store) breaks one of the reqirements for 1NF.

Think of the phrase - every non-key attribute depends on the key. In this example, store is a non-key attribute and can have multiple values for each value of the key.

This can be made 1NF by making store part of the key. Now the combination of item # and store is enough to identify a single value for each attribute in the relation.

Next example:

(<u>Store id</u>, Store address, Store mgr)

In this relation, Store id is a composite field made up of :

Store # - a unique store identifier
Store city code - an identifier for which city the store is in.

The relation is in 1NF because it passes our first test (assuming every store has one and only one manager) :

Every non-key attribute depends on the key.

Is the relation in 2NF form?

I would guess that it's not in 2NF because Store #, by itself, is enough to uniquely identify each store. As a result, the non-key attributes don't depend on "the whole key."

Correct. Because Store # is - all by itself - enough to identify Store address, it does not fit the criteria of 2NF.

And finally (and by now you can guess the answer), is the following in 3NF form?

SUPPLIER (supplier_no, status, city)

Functional Dependencies:

supplier_no -> status

supplier_no -> city

city -> status

And since I am shutting down in a few minutes, I will go ahead and answer - no, it is not.

Since status can be determined from city as well as supplier_no, it violates the phrase

and nothing but the key.

Hope these examples have helped...