<http://www.andrewrollins.com/2009/08/11/database-normalization-first-second-and-third-normal-forms/>

**Database Normalization: First, Second, and Third Normal Forms**

Posted on [August 11, 2009](http://www.andrewrollins.com/2009/08/11/database-normalization-first-second-and-third-normal-forms/) by [Andrew](http://www.andrewrollins.com/author/admin/)

I read a great explanation of [first, second, and third normal form](http://phlonx.com/resources/nf3/) a few weeks ago. For those that know what database normalization is but haven’t seen the “forms”, the different forms are essentially rules for having a well normalized relation DB. Keeping them in mind when doing DB design is key to keeping a great database. I’d like to make an attempt at condensing the linked tutorial into its essentials.

First Normal Form (1NF): No repeating elements or groups of elements

Don’t repeat your columns. Avoid this:

|  |  |  |  |
| --- | --- | --- | --- |
| OrderId | ItemId1 | ItemId2 | … |
| 1 | 100 | 101 |  |

ItemId1, 2, … should be split out into relational tables.

Second Normal Form (2NF): No partial dependencies on a concatenated key

This is a complex way of saying that if a column isn’t intrinsically related to the entire primary key, then you should break out the primary key into different tables.

Example:

|  |  |  |  |
| --- | --- | --- | --- |
| OrderId (PK) | ItemId (PK) | OrderDate | … |
| 1 | 100 | 2009-01-01 |  |
| 1 | 101 | 2009-01-01 |  |

The primary key is (OrderId, ItemId).

Consider OrderDate. It is conceptually part of an order. An order always occurs at some time. But is an OrderDate related to an Item? Not really.

You may be saying, “but items are part of an order!”, and you would be right. But that’s not what I’m getting at. OrderDate is independent of the item itself.

Look at another way: in the table above the OrderDate will always be the same for a given OrderId regardless of the value of the ItemId column. This means data duplication, which is denormalization.

Here’s how we correct the problem:

|  |  |  |
| --- | --- | --- |
| **Orders** |  |  |
| OrderId (PK) | OrderDate | … |
| 1 | 2009-01-01 |  |

|  |  |  |
| --- | --- | --- |
| **Order\_Items** |  |  |
| OrderId (PK) | ItemId (PK) | … |
| 1 | 100 |  |
| 1 | 101 |  |

Here is an excellent line from the article, “All we are trying to establish here is whether a particular order on a particular date relies on a particular item.”

Third Normal Form (3NF): No dependencies on non-key attributes

2NF covers the case of multi-column primary keys. 3NF is meant to cover single column keys. Simply stated, pull out columns that don’t directly relate to the subject of the row (the primary key), and put them in their own table.

Example:

|  |  |  |  |
| --- | --- | --- | --- |
| **Orders** |  |  |  |
| OrderId (PK) | OrderDate | CustomerName | CustomerCity |
| 1 | 2009-01-01 | John Smith | Chicago |

Customer information could be the subject of its own table. Pull out customer name and other customer fields into another table, and then put a Customer foreign key into Orders.

Wikipedia has a [great quote from Bill Kent](http://en.wikipedia.org/wiki/Third_normal_form#cite_note-Kent-4): “every non-key attribute ‘must provide a fact about the key, the whole key, and nothing but the key’.”

This entry was posted in [Databases](http://www.andrewrollins.com/category/databases/) and tagged [Databases](http://www.andrewrollins.com/tag/databases/). Bookmark the [permalink](http://www.andrewrollins.com/2009/08/11/database-normalization-first-second-and-third-normal-forms/).

[← MySQL Join Performance](http://www.andrewrollins.com/2009/06/21/mysql-join-performance/)