

Ubercart 2.0 – Fractional Quantities

By Rob DiNardo

In some cases, it is necessary to sell products in non-integer quantities. Examples: a fabric store or butcher shop should be able to sell in 2.4 m of fabric or 3.75 kg of meat. The current Ubercart (v 2.0) does not allow decimals for their quantities.

If you search the Ubercart forums for “quantity as decimal”, “fractions for quantity”, and “decimal quantities” you get some hits. This article is an effort to outline some changes that may be made to your installed Ubercart system to allow for “fractional quantities”.

Thank you to Lyle and his post reply that helped me get started on this article:
http://www.ubercart.org/forum/ideas_and_suggestions/2510/quantity_decimal

Add To Ubercart Core?

I hope that the Ubercart developers can find a way to implement the ability for “fractional quantities” to Ubercart Core. I hope that this document / article will help make that happen!

Database Changes

Changing Ubercart to accept fractional quantities means the datatype of some table columns must be changed from INTEGER to FLOAT(M,D). The FLOAT data type allows for the decimal to be stored. Here is a description from

<http://dev.mysql.com/doc/refman/5.0/en/numeric-types.html>

Here, “(M,D)” means that values can be stored with up to M digits in total, of which D digits may be after the decimal point. For example, a column defined as FLOAT(7,4) will look like -999.9999 when displayed. MySQL performs rounding when storing values, so if you insert 999.00009 into a FLOAT(7,4) column, the approximate result is 999.0001.

Below are the changes to the tables to be made to allow 2 decimal places and 6 total digits. Applying the following database table changes **should not** affect existing data, unless you have quantities greater than 6 digits – in which case you could increase the M value.

UC_CART_PRODUCTS

The UC_CART_PRODUCTS table intersects the UC_CARTS table and the UC_PRODUCTS table. The column “qty” is the one to adjust. Here is the MySQL statement that will modify the column to a FLOAT data type:

```
ALTER TABLE `uc_cart_products` MODIFY COLUMN `qty` FLOAT(6,2) UNSIGNED NOT NULL DEFAULT 0;
```

UC_ORDERS

The UC_ORDERS table holds all the orders created. The column to adjust is “product_count”.

```
ALTER TABLE `uc_orders` MODIFY COLUMN `product_count` FLOAT(6,2) UNSIGNED NOT NULL DEFAULT 0;
```

UC_PRODUCTS

The UC_PRODUCTS table contains a default_qty field. This value gets inserted into both the product edit page and the product view page (for the customer). I think it would be a good idea to show the customer how many decimal places one may use.

```
ALTER TABLE `uc_products` MODIFY COLUMN `default_qty` FLOAT(6,2) UNSIGNED NOT NULL DEFAULT 1.00;
```

UC_ORDER_PRODUCTS

The UC_ORDER_PRODUCTS table intersects the UC_ORDERS table and the UC_PRODUCTS table. The column to adjust is the "qty" column. Here is the MySQL statement that will modify the column to a FLOAT data type:

```
ALTER TABLE `uc_order_products` MODIFY COLUMN `qty` FLOAT(6,2) UNSIGNED NOT NULL DEFAULT 0;
```

Code Changes

Making **changes to core Ubercart code is not recommend**. It is recommended to create modules (with separate database tables) to add custom functionality. These code changes may not be complete and may disrupt your system. If you decide to make these changes... **Please proceed with caution**. I suggest that the changes

uc_cart.module (line 1445)

```
db_query("UPDATE {uc_cart_products} SET qty = %d, changed = UNIX_TIMESTAMP(), data = '%s' WHERE cart_item_id = %d",
```

to...

```
db_query("UPDATE {uc_cart_products} SET qty = %f, changed = UNIX_TIMESTAMP(), data = '%s' WHERE cart_item_id = %d",
```

uc_cart.module (line 1509)

```
db_query("INSERT INTO {uc_cart_products} (cart_id, nid, qty, changed, data) VALUES ('%s', %d, %d, %d, '%s')", $cid, $node->nid, $qty, time(), serialize($data));
```

to...

```
db_query("INSERT INTO {uc_cart_products} (cart_id, nid, qty, changed, data) VALUES ('%s', %d, %f, %d, '%s')", $cid, $node->nid, $qty, time(), serialize($data));
```

uc_order.module (line 1043)

```
db_query("UPDATE {uc_orders} SET uid = %d, order_status = '%s', order_total = %f, product_count = %d, primary_email = '%s', "
```

to...

```
db_query("UPDATE {uc_orders} SET uid = %d, order_status = '%s', order_total = %f, product_count = %f, primary_email = '%s', "
```

uc_order.module (line 1143)

```
db_query("UPDATE {uc_orders} SET product_count = %d WHERE order_id = %d",  
$count, $order->order_id);
```

to...

```
db_query("UPDATE {uc_orders} SET product_count = %f WHERE order_id = %d",  
$count, $order->order_id);
```

uc_order.install (replace lines 48 to 51)

```
'type' => 'int',  
'unsigned' => TRUE,  
'not null' => TRUE,  
'default' => 0,
```

with...

```
'type' => 'float',  
'precision' => '6',  
'scale' => '2',  
'unsigned' => TRUE,  
'not null' => TRUE,  
'default' => 1.00,
```

uc_product.module (line 1207)

```
db_query("UPDATE {uc_cart_products} SET qty = %d, changed = %d WHERE nid = %d  
AND cart_id = '%s' AND data = '%s'", $qty, time(), $nid, $cid,  
serialize($data));
```

to...

```
db_query("UPDATE {uc_cart_products} SET qty = %f, changed = %d WHERE nid = %d  
AND cart_id = '%s' AND data = '%s'", $qty, time(), $nid, $cid,  
serialize($data));
```

Thoughts - Additional Changes

It would probably be a good idea to add some functionality to allow a choice as to whether or not a certain product type / class is allowed to accept “fractional quantities”. Maybe the addition of a boolean column (like “allow_frac_qty”) in the **uc_product_classes** table or the **uc_products** table. Then of course, there would be more code additions / changes to allow for this.

Also, the “pkg_qty” column in the **uc_products** table may also need to be changed!

Other Forum Posts

http://www.ubercart.org/forum/support/4651/use_fractions_quantity_15_yards

http://www.ubercart.org/forum/support/6074/decimal_quantities_items

http://www.ubercart.org/issue/6044/ability_have_decimal_quantities

http://www.ubercart.org/forum/ideas_and_suggestions/3283/comma_values_quantity_field

-- end, last updated: 2009-11-13