Hi Abdul,

Here is what we need for the product pricing.

We'll do this in a similar way to the cost pricing with a table of prices, and another containing a record for each day as a log.

This time there is need to differentiate between simple and bundled products as the sales prices are all set manually. There are two prices for each item the "regular_price" and "special_price" (when the item is reduced). When there is no reduction, special_price is set to 0.

Two new tables to import are attached:

product_sale_prices

product_price_history

For the history, we will need a cron just like cost price to loop through the products where active > 0, with each currency – this will produce three entries in product_price_history table , one for EUR, GBP and USD, each with the regular price, and the special price (or 0)

For now temporarily inhibit USD as we won't have a price list for this until January 2018.

I think we need an insert/update query (like we have on the fx_exchange_rates cron) here as it is possible the cron will be re-run in a day, or just for an individual product and we only want one record per day for each product.

Here is the calculation in two parts – I can't see how to do in one query, but maybe you can.

By way of explanation, "prevailing prices" are prices for current days history, "In date" is where effective >= current date and expiry is <= current date, or 0, which means it has no expiry and will be current until a new price is loaded.

First, find the lowest special price that is "in date" and greater than 0. Zero means there is no special price and regular price is only one available. If a result returned, that (and regular price from the same row) are the prevailing prices.

```
SELECT *
FROM product_sale_prices
WHERE product_id = ?
AND CURDATE() >= effective
AND (CURDATE() <= expiry OR expiry = 0)
AND special_price > 0
ORDER BY special_price ASC
LIMIT 1
```

If no result returned, find the lowest regular price that is in date and set the special price to 0.

```
SELECT *
FROM product_sale_prices
WHERE product_id = ?
AND CURDATE() >= effective
AND (CURDATE() <= expiry OR expiry = 0)
AND special_price = 0
ORDER BY regular_price ASC
LIMIT 1
```

If nothing found on second search we need to generate an email notice to the administrator. Maybe do this at the end of the cron job with each product_id listed on line, like

Following items were not priced:

100004 – Falk 24cm Frying Pan - EUR

To find the current price for a product retrieve the most recent record with matching product_id and currency from product_price_history.