projekt kivitendo - Fehler #355

Kontoauszug verbuchen -> Eine Bankbewegung mit zwei Skonto Rechnungen verknüpfen geht nicht

02.06.2018 22:08 - Jan Büren

Status:	Erledigt	**Beginn:**	02.06.2018
Priorität:	Niedrig	**Abgabedatum:**	
Zugewiesen an:		**% erledigt:**	0%
Kategorie:		**Geschätzter Aufwand:**	0.00 Stunde
Zielversion:		**Aufgewendete Zeit:**	0.00 Stunde

Beschreibung

Der Kunde bezahlt zwei Rechnungen mit Skonto.

Zuweisen kann ich diese nicht auf einmal.

Möglich ist dies, wenn ich die DATEV-Prüfung ausschalte und nicht mehr überprüfe, ob die eine Rechnung vom Betrag kleiner als die Bankbewegung ist:

SL/Controller/BankTransaction.pm

```
       # pay invoice or go to the next bank transaction if the amount is not sufficiently high
-      if ($invoice->open_amount <= $amount_of_transaction && $n_invoices < $max_invoices) {
+      if ( $n_invoices < $max_invoices) {
```

Es wäre vielleicht etwas besser die Fälle die nicht sauber verbuchbar sind, schon vor der Verbuchung der einzelnen Rechnungen anzumahnen.

POD:
save_single_bank_transaction
Assings one bank transaction to one or more invoice.
The invoices may have optional payment terms.
If the sum of the bank transaction is lower than the sum of the selected invoices, the user´s last selected invoice will only be partly paid.
If there is more than one invoice which has to be partly paid, nothing will be commited and a error message is thrown.

Historie

#1 - 12.06.2018 14:48 - Jan Büren

Folgende Anpassungen:

```
       # pay invoice or go to the next bank transaction if the amount is not sufficiently high
-      if ($invoice->open_amount <= $amount_of_transaction && $n_invoices < $max_invoices) {
+      if ($n_invoices < $max_invoices) {
         my $open_amount = ($payment_type eq 'with_skonto_pt'?$invoice->amount_less_skonto:$invoice->open_amou
nt);
         # first calculate new bank transaction amount ...
         if ($invoice->is_sales) {
@@ -658,6 +686,7 @@ sub save_single_bank_transaction {
                               transdate    => $bank_transaction->transdate->to_kivitendo);
       } else {
         # use the whole amount of the bank transaction for the invoice, overpay the invoice if necessary
+        # but also check for the skonto case

         # this catches credit_notes and negative sales invoices
         if ( $invoice->is_sales && $invoice->amount < 0 ) {
@@ -673,6 +702,8 @@ sub save_single_bank_transaction {
           # but amount of transaction is for both positive
           $amount_of_transaction *= -1 if $invoice->open_amount == - $amount_of_transaction;
         }
+        # if we have a skonto case - the last invoice needs
+        $amount_of_transaction = $invoice->amount_less_skonto if ($payment_type eq 'with_skonto_pt');
```

Damit wird das Skonto auch berehchnet wenn wir bei der letzten Zuweisung sind.

Prinzipiell vertrau ich dem Algorithmus nicht.
Im POD wird behauptet, dass die Routine transaktionssicher ist, ein Rollback passiert aber in dem Fall Mehrere Rechnung für eine Bankverbuchung nicht.
D.h. einige Rechnungen werden verbucht und nur die Rechnungen die fehlerhaft sind werden davon ausgeklammert.

Im Odyn sieht es an dieser Stelle nicht besser aus, sondern eher leider nur komplexer.
Ich würde auch einen rewrite scheuen, aber die Zustände dieser Funktion sollte schon mal vorher geprüft werden, bevor die Verbucherei losrennt:

```
+    my $max_invoices          = scalar(@{ $params{invoice_ids} });

+    # safety check first. save_single bank transaction is not transaction safe
+    # the algorithms also falsely assumed that the last invoice is reached
+    # to avoid side effects we will do the checks before committing anything
+    # 1. algorithm works well for 1 to 1 relationship
+    # 2. if we have n invoices maybe with different payment types we calculate
+    #    the skonto amount based on the payment types and check if we can assign with some rounding mistakes
+    # 3. We generally disallow overpaying invoices if we have multiple invoices (which invoice was overpaid?)

+    my ($has_skonto, $payment_sum);
+    my @payment_types = @{ $::form->{invoice_skontos}->{"$bt_id"} };
     foreach my $invoice_id (@{ $params{invoice_ids} }) {
       my $invoice = SL::DB::Manager::Invoice->find_by(id => $invoice_id) || SL::DB::Manager::PurchaseInvoice-
>find_by(id => $invoice_id);
       if (!$invoice) {
@@ -578,6 +588,24 @@ sub save_single_bank_transaction {
         };
       }
       push @{ $data{invoices} }, $invoice;
+
+      my $payment_type;
+      $payment_type = shift(@payment_types) if @payment_types;
+      if ($payment_type eq 'with_skonto_pt') {
+        $payment_sum += $invoice->amount_less_skonto;
+        $has_skonto = 1;
+      } else {
+        $payment_sum += $invoice->open_amount;
+      }
+    }
+    if ($max_invoices > 1 && abs($amount_of_transaction - $payment_sum) > 0.05) {
+      return {
+        %data,
+        result => 'error',
+        message => $has_skonto ? $::locale->text("Cannot post bank transaction because the calculated amounts
, #1 " .
+                                  " (less skonto) differ from bank transaction amount #2", $payment_sum, $amou
nt_of_transaction)
+                               : $::locale->text("Cannot overpay multiple invoices. Please select only a sing
le one")
+      };
```

#2 - 20.06.2018 13:16 - Jan Büren

- Priorität wurde von Normal zu Niedrig geändert

Ich hab einen Testfall dafür #b75c6cbb82023b0d626ffc9996e

Ich kann das Verhalten mit exakten Werten im Testfall nicht reproduzieren, da müssen sich sehr wahrscheinlich noch andere Parameter beeinflussen oder ich hab falsch geprüft.

#3 - 14.11.2018 11:18 - Jan Büren

- Status wurde von Neu zu Erledigt geändert

Nicht mehr nachvollziehbar