```
Set Error Capture [ On ]
Allow User Abort [ Off ]
Set Variable [ $$er; Value:"" ]
If [ Get(ScriptParameter) = "close" ]
    Close Window [ Current Window ]
    Exit Script [  ]
End If
If [ Get(ScriptParameter) = "again" ]
Else
    If [ Get(ScriptParameter) = "item" ]
        New Window [  ]
        Go to Layout [ “Quote Entry Item Find” (quotes) ]
        Perform Script [ “window resize [centered, full]”; Parameter: "centered" ]
    Else
        Go to Layout [ “Quote Entry” (quotes) ]
    End If
End If
Enter Find Mode [  ]
Loop
    #enter find mode
    Enter Find Mode [  ]
    Go to Object [ Object Name: "field.description" ]
    Pause/Resume Script [ Indefinitely ]
    Perform Find [  ]
    Exit Loop If [ Case(
        $$er = 1000;1;
         (Let($$er =Get(LastError) ; $$er = 0  or $$er = 400  ));1;
        ""
        ) ]
    #try switching name to company
    Modify Last Find
    Cut [ quotes::Quote company name ]
        [ Select ]
    Paste [ quotes::Quote contact name ]
        [ Select ]
    Perform Find [  ]
    Exit Loop If [ (Let($$er =Get(LastError) ; $$er = 0  or $$er = 400  )) ]
    #try mixing contact and company
    Modify Last Find
    Set Field [ quotes::Quote contact name; LeftWords(quotes::Quote contact name;  1) ]
    Set Field [ quotes::Quote company name; RightWords(quotes::Quote contact name; WordCount(quotes::Quote contact name) -
        1) ]
    Perform Find [  ]
    Perform Find/Replace [  ]
    Exit Loop If [ (Let($$er =Get(LastError) ; $$er = 0  or $$er = 400  )) ]
    #find failed, try again
    Beep
End Loop
If [ $$er =0  //Find worked ]
    Perform Script [ “Sort by Date” ]
    Go to Record/Request/Page
```

```
        [ First ]
Else
    Close Window [ Current Window ]
End If
#
Exit Script [  ]
#
#put before each find for initials filter, make new script if needed
Go to Record/Request/Page
    [ First ]
If [
    PatternCount(Get(CurrentPrivilegeSetName);"superuser")

     or
    contacts.primary::<Field Missing> =  "all" ]
Else
    New Record/Request
    Set Field [ contacts.primary::xxIDL_Owner; "<" & contacts.primary::<Field Missing> ]
    Omit Record
    New Record/Request
    Set Field [ contacts.primary::xxIDL_Owner; ">" & contacts.primary::<Field Missing> ]
    Omit Record
End If
```

```
Freeze Window
If [ Get(ScriptParameter) = "previous" ]
    Go to Record/Request/Page
        [ Previous ]
Else
    Go to Record/Request/Page
        [ Next ]
End If
```

Delete quote

Allow User Abort [Off]

Show Custom Dialog [Title: "Message"; Message: "Delete quote?"; Buttons: “Cancel”, “OK”]

If [Get(LastMessageChoice) = 2]

    Set Variable [$PC; Value:quotes::xxIDL_LinkPrimaryItem]

    Set Field [window::xSys_NewItem; "Trash"]

    Set Field [window::xSys_NewItem[2]; quotes::xxIDL_IDNumberInternal]

    Perform Script [“Move/Copy/ClearToAnyFolder [$$drop.ID [1], $$DragList [2] cleans, g(sp) = from folders] ” from file: “customers”]

    Omit Record

End If

Freeze Window

Go to Layout [“Quote Entry” (quotes)]

Set Variable [$PC; Value:quotes::xxIDL_LinkPrimaryItem]

New Record/Request

Set Field [revision::System rev quote notes print; ""]

Set Field [revision::System rev Quote date; Get(CurrentDate)]

Set Field [quotes::xxIDL_LinkPrimaryItem; $PC]

Set Field [window::xSys_NewItem; $PC]

Set Field [window::xSys_NewItem[2]; quotes::xxIDL_IDNumberInternal]

Commit Records/Requests
[Skip data entry validation; No dialog]

Perform Script [“LinkItems [$$drop.id[1], $$draglist[2]] / NewLinks [$$drop.id [1]]” from file: “customers”]

Set Field [window::xSys_NewItem; "Inbox"]

Set Field [window::xSys_NewItem[2]; quotes::xxIDL_IDNumberInternal]

Perform Script [“Move/Copy/ClearToAnyFolder [$$drop.ID [1], $$DragList [2] cleans, g(sp) = from folders] ” from file: “customers”]

Exit Script [ ]

Show Custom Dialog [Message: $$drop.id & "¶" & $$draglist; Buttons: “OK”, “Cancel”]

Freeze Window

Go to Layout [“Quote Entry” (quotes)]

Set Field [revision::System rev quote notes print; ""]

Exit Script [ ]

Show Custom Dialog [Message: $$drop.id & "¶" & $$draglist; Buttons: “OK”, “Cancel”]

Copy Quote

Set Error Capture [On]

If [Get(WindowMode) = 0]

Freeze Window

Go to Layout [“Quote Entry” (quotes)]

Set Field [quotes::sys.rev number; quotes::sys.rev number max]

Set Variable [$note; Value:revision::System rev quote notes print]

#duplicate line items,note: create ID number in variable before duplicating line items, set to field after duplicating line items

Set Variable [$b; Value:Let([
$$newID[1] = "Q" &
Right("00" & Day(Get(CurrentDate));2) &
Right("00" & Month(Get(CurrentDate));2) &
Right(Year(Get(CurrentDate));2) &
Right("0000" & GetAsText(Get(RecordID));4) &
Right("0000" & GetAsText(Int(9999*Random));4) ;

$$newID[2] = "1";
$$newID[3] = ""
];"")]

Go to Related Record [From table: “quote data”; Using layout: “data.quote data” (quote data)]
[Show only related records]

If [Get(LastError) = 0]

Perform Script [“duplicate multiple line items ($$newID[1]=ac, [2]=rev#,[3]=""/end)”]

End If

Go to Layout [original layout]

Commit Records/Requests
[Skip data entry validation; No dialog]

Duplicate Record/Request

#set new quote fileds to new values

Set Field [quotes::xxIDL_IDNumberInternal; $$newID[1]]

Set Field [quotes::sys.rev number; 1]

Set Field [revision::System rev quote notes print; $note]

Set Field [revision::System rev Quote date; Get(CurrentDate)]

Set Field [quotes::Quote notes system; ""]

Set Field [quotes::Quote number; Max(Quotes default for max quote number::Quote number) + 1]

Set Field [quotes::Quote close month; If(Day(Get(CurrentDate)) < 15; Left(MonthName(Get(CurrentDate)); 3); Left (MonthName(Get(CurrentDate)+ 30); 3))]

Set Field [quotes::Quote close year; Right(Year(Get(CurrentDate)) ; 2)]

Set Field [quotes::Quote status; "Open"]

Set Variable [$PC; Value:quotes::xxIDL_LinkPrimaryItem]

Set Field [window::xSys_NewItem; $PC]

Set Field [window::xSys_NewItem[2]; quotes::xxIDL_IDNumberInternal]

Commit Records/Requests
[Skip data entry validation; No dialog]

Perform Script [“LinkItems [$$drop.id[1], $$draglist[2]] / NewLinks [$$drop.id [1]]” from file: “customers”]

Set Field [window::xSys_NewItem; "Inbox"]

Set Field [window::xSys_NewItem[2]; quotes::xxIDL_IDNumberInternal]

Perform Script [“Move/Copy/ClearToAnyFolder [$$drop.ID [1], $$DragList [2] cleans, g(sp) = from folders] ” from file: “customers”]

Commit Records/Requests
[Skip data entry validation; No dialog]

End If

Commit Records/Requests

[Skip data entry validation; No dialog]

Commit Records/Requests
[Skip data entry validation; No dialog]

New Window [Name: "Import"; Height: 848; Width: 1090; Top: 91; Left: 187]

Go to Layout [“Import” (quotes)]

Commit Records/Requests
[Skip data entry validation; No dialog]

Freeze Window

Commit Records/Requests
[Skip data entry validation; No dialog]

Paste [quotes::sys.importer.paste]
[Select; No style]

Set Variable [$load; Value:quotes::sys.importer.paste]

Set Field [quotes::sys.importer.paste; ""]

Commit Records/Requests
[Skip data entry validation; No dialog]

If [PatternCount ($load ; Char(9)) > 3]

Set Variable [$ac; Value:Right(GetAsText(Random) ; 5) & Right(GetAsText(Random) ; 5) & Right(GetAsText(Random) ; 5) & Right(GetAsText(Random) ; 5)]

Set Field [quotes::ID importer; $ac]

Set Variable [$ll; Value:Filter (
TextFormatRemove (
Substitute (
$load ; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"]; ["¶¶" ; "¶"]; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"]
)
) ;
"
~!@#$%^&*()_+`1234567890-=QWERTYUIOP{}lqwertyuiop[]\ASDFGHJKL:\"asdfghjkl;'ZXCVBNM<>?zxcvbnm,./°•
¶"
)]

Loop

Exit Loop If [Case(
$count < 1 ; Let([
$count = 1 ;
$end = PatternCount($ll ; "¶") + 1
] ; "") ;
$count ≥ $end or $end < 1; Let($count = 0 ; 1) ;
Let($count = $count + 1 ; "")
)]

Set Variable [$line; Value:Trim(Substitute(MiddleValues($ll ; $count ; 1) ; ["¶" ; ""] ; [Char(9) ; "¶"]))]

If [PatternCount ($line ; "¶") > 1]

Set Variable [$id; Value:Right(GetAsText(Random) ; 5) & Right(GetAsText(Random) ; 5) & Right(GetAsText (Random) ; 5) & Right(GetAsText(Random) ; 5)]

Set Field [quotes::ID importer.item; $ID]

Set Field [importer.item::ID number internal importer; $ID]

Set Field [importer.item::ID importer; $ac]

Loop

Exit Loop If [Case(
$count2 < 1 ; Let([
$count2 = 1 ;
$end2 = PatternCount($line ; "¶") + 1 ;
$end2 = If($end2 > 30 ; 30 ; $end2)
] ; "") ;
$count2 ≥ $end2 or $end2 < 1; Let($count2 = 0 ; 1) ;
Let($count2 = $count2 + 1 ; "")
)]

Set Field [quote data::sys.importer.movecolumns[$count2]; Case($count = 1 ; "" ; importer::sys.importer. movecolumns[$count2])]

Set Field [importer.item::sys.importer.fields[$count2]; Trim (Substitute(MiddleValues ($line ; $count2; 1) ; ["¶" ; ""]))]

Set Variable [$value; Value:If(Length(Trim (Substitute(MiddleValues ($line ; $count2; 1) ; ["¶" ; ""]))) > 0 ; 1 ; 0)]

Set Variable [$aa; Value:$aa + $value]

Set Field [quote data::sys.importer.movecolumns[$count2]; Case($value ; $value ; importer::sys.importer. movecolumns[$count2])]

End Loop

Set Field [importer.item::sys.importer.item number; $count]

```
        Set Field [ importer.item::sys.importer.active; Case(
            $aa > 2 ; 1 ; "" ) ]
        Set Variable [ $aa; Value:0 ]
        Commit Records/Requests
            [ Skip data entry validation; No dialog ]
    End If
End Loop
#
Set Variable [ $c2; Value:Count( importer::ID number internal importer ) ]
Loop
    Exit Loop If [ Case(
    $count < 1 ; Let( [
        $count = 1 ;
        $end = 30
    ] ; "" ) ;
    $count  ≥ $end or $end < 1;    Let( $count = 0 ; 1) ;
    Let( $count = $count + 1 ; "" )
    ) ]
    Set Variable [ $nexte; Value:Case(
    importer::sys.importer.movecolumns[ 1 ] < 1 ; 1 ;
    importer::sys.importer.movecolumns[ 2 ] < 1 ; 2 ;
    importer::sys.importer.movecolumns[ 3 ] < 1 ; 3 ;
    importer::sys.importer.movecolumns[ 4 ] < 1 ; 4 ;
    importer::sys.importer.movecolumns[ 5 ] < 1 ; 5 ;
    importer::sys.importer.movecolumns[ 6 ] < 1 ; 6 ;
    importer::sys.importer.movecolumns[ 7 ] < 1 ; 7 ;
    importer::sys.importer.movecolumns[ 8 ] < 1 ; 8 ;
    importer::sys.importer.movecolumns[ 9 ] < 1 ; 9 ;
    importer::sys.importer.movecolumns[ 10 ] < 1 ; 10 ;
    importer::sys.importer.movecolumns[ 11 ] < 1 ; 11 ;
    importer::sys.importer.movecolumns[ 12 ] < 1 ; 12 ;
    importer::sys.importer.movecolumns[ 13 ] < 1 ; 13 ;
    importer::sys.importer.movecolumns[ 14 ] < 1 ; 14 ;
    importer::sys.importer.movecolumns[ 15 ] < 1 ; 15 ;
    importer::sys.importer.movecolumns[ 16 ] < 1 ; 16 ;
    importer::sys.importer.movecolumns[ 17 ] < 1 ; 17 ;
    importer::sys.importer.movecolumns[ 18 ] < 1 ; 18 ;
    importer::sys.importer.movecolumns[ 19 ] < 1 ; 19 ;
    importer::sys.importer.movecolumns[ 20 ] < 1 ; 20 ;
    importer::sys.importer.movecolumns[ 21 ] < 1 ; 21 ;
    importer::sys.importer.movecolumns[ 22 ] < 1 ; 22 ;
    importer::sys.importer.movecolumns[ 23 ] < 1 ; 23 ;
    importer::sys.importer.movecolumns[ 24 ] < 1 ; 24 ;
    importer::sys.importer.movecolumns[ 25 ] < 1 ; 25 ;
    importer::sys.importer.movecolumns[ 26 ] < 1 ; 26 ;
    importer::sys.importer.movecolumns[ 27 ] < 1 ; 27 ;
    importer::sys.importer.movecolumns[ 28 ] < 1 ; 28 ;
    importer::sys.importer.movecolumns[ 29 ] < 1 ; 29 ;
    importer::sys.importer.movecolumns[ 30 ] < 1 ; 30 ;
    32
    ) ]
    If [ importer::sys.importer.movecolumns[ $count ] < 1 ]
    Else If [ $nexte < $count ]
        Loop
            Exit Loop If [ Case(
            $count2 < 1 ; Let( [
               $count2 = 1 ;
               $end2 = $c2
            ] ; "" ) ;
            $count2  ≥ $end2 or $end2 < 1;          Let( $count2 = 0 ; 1) ;
            Let( $count2 = $count2 + 1 ; "" )
            ) ]
            Set Field [ quotes::ID importer.item; GetNthRecord ( importer::ID number internal importer ; $count2 ) ]
            Set Field [ importer.item::sys.importer.fields[$nexte]; importer.item::sys.importer.fields[$count ] ]
            Set Field [ importer.item::sys.importer.fields[$count]; "" ]
```

```
        End Loop
            Set Field [ quote data::sys.importer.movecolumns[$count]; "" ]
            Set Field [ quote data::sys.importer.movecolumns[$nexte]; 1 ]
            Commit Records/Requests
                [ Skip data entry validation; No dialog ]
        End If
    End Loop
Else
    Show Custom Dialog [ Message: "There does not appear to be a valid quote in the clipboard."; Buttons: “OK” ]
End If
Set Field [ quotes::ID importer.item; "" ]
Exit Script [  ]
Show Custom Dialog [ Message:
    "nexte"  &  $nexte & "¶" &
    "$count "  & $count & "¶"; Buttons: “OK”, “Cancel” ]
Show Custom Dialog [ Message: "val "  &  $value  & "¶" &
    "importer::sys.importer.movecolumns "  & importer::sys.importer.movecolumns[$count2] & "¶" &
    "$count2 "  & $count2 & "¶" &
    "nexte"  &  $nexte & "¶" &
    "$count "  & $count & "¶" &
    ""  & "¶" &
    ""  & "¶" &
    ""  & "¶"; Buttons: “OK”, “Cancel” ]
```

Freeze Window

Commit Records/Requests
[Skip data entry validation; No dialog]

If [IsEmpty(quotes::sys.importer.selection[1]) and
IsEmpty(quotes::sys.importer.selection[2]) and
IsEmpty(quotes::sys.importer.selection[3]) and
IsEmpty(quotes::sys.importer.selection[4]) and
IsEmpty(quotes::sys.importer.selection[5]) and
IsEmpty(quotes::sys.importer.selection[6]) and
IsEmpty(quotes::sys.importer.selection[7]) and
IsEmpty(quotes::sys.importer.selection[8]) and
IsEmpty(quotes::sys.importer.selection[9]) and
IsEmpty(quotes::sys.importer.selection[10]) and
IsEmpty(quotes::sys.importer.selection[11]) and
IsEmpty(quotes::sys.importer.selection[12])]

Show Custom Dialog [Message: "Please map fields"; Buttons: “OK”]

Exit Script [ ]

End If

Set Variable [$LL; Value:List(importer::ID number internal importer)]

Set Variable [$c; Value:WordCount($LL)]

If [$c > 0 and Count(importer::sys.importer.active) > 0]

Set Variable [$in; Value:Int(Max(quote data::Item number))]

Set Variable [$z; Value:Let([
$rpart = 31 ;
$rsku = 31 ;
$rdesc1 = 31 ;
$rdesc2 = 31 ;
$rqty = 31 ;
$rlist = 31 ;
$rcost = 31
] ; "")

/*
importer.item::sys.importer.fields [$count2]
quote data.item::Item list price manual
quote data.item::Item description
quote data.item::Item number
quote data.item::Item part number
quote data.item::Item part number SKU
quote data.item::Item quantity
rpart
rsku
rdesc1
rdesc2
rqty
rlist
rcost

Part#
SKU#
Desc1
Desc2
Qty
List
Cost
Filter (Trim (Substitute(importer.item::sys.importer.fields [$count2] ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789")
*/]

Loop

Exit Loop If [Case(
$count < 1 ; Let([
$count = 1 ;
$end = 12
] ; "") ;
$count ≥ $end or $end < 1; Let($count = 0 ; 1) ;
Let($count = $count + 1 ; "")
)]

Set Variable [$rpart; Value:Case(quotes::sys.importer.selection [$count] = "Part#" ; $count ; $rpart)

/*
importer.item::sys.importer.fields [$count2]
quote data.item::Item list price manual
quote data.item::Item description
quote data.item::Item number
quote data.item::Item part number
quote data.item::Item part number SKU
quote data.item::Item quantity
rpart
rsku
rdesc1
rdesc2
rqty
rlist
rcost

Part#
SKU#
Desc1
Desc2
Qty
List
Cost
Filter (Trim (Substitute(importer.item::sys.importer.fields [$count2] ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789")
*/]

Set Variable [$rsku; Value:Case(quotes::sys.importer.selection [$count] = "SKU#" ; $count ; $rsku)

/*
importer.item::sys.importer.fields [$count2]
quote data.item::Item list price manual
quote data.item::Item description
quote data.item::Item number
quote data.item::Item part number
quote data.item::Item part number SKU
quote data.item::Item quantity
rpart
rsku
rdesc1
rdesc2
rqty
rlist
rcost

Part#
SKU#
Desc1
Desc2
Qty
List
Cost
Filter (Trim (Substitute(importer.item::sys.importer.fields [$count2] ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789")
*/]

Set Variable [$rdesc1; Value:Case(quotes::sys.importer.selection [$count] = "desc1" ; $count ; $rdesc1)

/*
importer.item::sys.importer.fields [$count2]
quote data.item::Item list price manual
quote data.item::Item description
quote data.item::Item number
quote data.item::Item part number
quote data.item::Item part number SKU
quote data.item::Item quantity
rpart
rsku
rdesc1
rdesc2
rqty
rlist
rcost

Part#
SKU#
Desc1

Desc2
Qty
List
Cost
Filter (Trim (Substitute(importer.item::sys.importer.fields [$count2] ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789")
*/]

Set Variable [$rDesc2; Value:Case(quotes::sys.importer.selection [$count] = "Desc2" ; $count ; $rDesc2)

/*
importer.item::sys.importer.fields [$count2]
quote data.item::Item list price manual
quote data.item::Item description
quote data.item::Item number
quote data.item::Item part number
quote data.item::Item part number SKU
quote data.item::Item quantity
rpart
rsku
rdesc1
rdesc2
rqty
rlist
rcost

Part#
SKU#
Desc1
Desc2
Qty
List
Cost
Filter (Trim (Substitute(importer.item::sys.importer.fields [$count2] ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789")
*/]

Set Variable [$rQty; Value:Case(quotes::sys.importer.selection [$count] = "Qty" ; $count ; $rQty)

/*
importer.item::sys.importer.fields [$count2]
quote data.item::Item list price manual
quote data.item::Item description
quote data.item::Item number
quote data.item::Item part number
quote data.item::Item part number SKU
quote data.item::Item quantity
rpart
rsku
rdesc1
rdesc2
rqty
rlist
rcost

Part#
SKU#
Desc1
Desc2
Qty
List
Cost
Filter (Trim (Substitute(importer.item::sys.importer.fields [$count2] ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789")
*/]

Set Variable [$rlist; Value:Case(quotes::sys.importer.selection [$count] = "List" ; $count ; $rlist)

/*
importer.item::sys.importer.fields [$count2]
quote data.item::Item list price manual
quote data.item::Item description
quote data.item::Item number
quote data.item::Item part number
quote data.item::Item part number SKU
quote data.item::Item quantity
rpart
rsku
rdesc1

rdesc2
rqty
rlist
rcost

Part#
SKU#
Desc1
Desc2
Qty
List
Cost
Filter (Trim (Substitute(importer.item::sys.importer.fields [$count2] ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789")
*/]

Set Variable [$rcost; Value:Case(quotes::sys.importer.selection [$count] = "Cost" ; $count ; $rcost)

/*
importer.item::sys.importer.fields [$count2]
quote data.item::Item list price manual
quote data.item::Item description
quote data.item::Item number
quote data.item::Item part number
quote data.item::Item part number SKU
quote data.item::Item quantity
rpart
rsku
rdesc1
rdesc2
rqty
rlist
rcost

Part#
SKU#
Desc1
Desc2
Qty
List
Cost
Filter (Trim (Substitute(importer.item::sys.importer.fields [$count2] ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789")
*/]

End Loop

Loop

Exit Loop If [Case(
$count < 1 ; Let([
$count = 1 ;
$end = $c
] ; "") ;
$count ≥ $end or $end < 1; Let($count = 0 ; 1) ;
Let($count = $count + 1 ; "")
)]

Set Variable [$dc; Value:$c -$count + 1]

Set Field [quotes::ID importer.item; MiddleWords($LL; $dc ; 1)]

If [importer.item::sys.importer.active = 1]

Set Field [quotes::ID quote data.item; Right(Random ; 7) &
Right(Random ;6)]

Set Field [quote data.item::ID number internal; quotes::ID quote data.item]

Set Field [quote data.item::ID quote number w Rev; quotes::xxIDL_IDNumberInternalWithRev]

Set Field [quote data.item::Item number; $in + $dc]

End If

Commit Records/Requests
[Skip data entry validation; No dialog]

Set Variable [$qty; Value:Case(
IsEmpty($rqty) ; "1" ;
Filter (Trim (Substitute(importer.item::sys.importer.fields [$rqty] ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789")
)]

```
If [ importer.item::sys.importer.active = 1 ]

Set Variable [ $warn; Value:Case(
    $cost = 0 and $list = 0 ; $warn  ;
    $qty > 1 ; 1 + $warn ;
    $warn
    ) ]

Set Variable [ $qty; Value:Case(
    $cost = 0 and $list = 0 ; $qty ; 1 ) ]

Set Field [ quote data.item::Item cost manual; Filter ( Trim ( Substitute( importer.item::sys.importer.fields [ $rcost ] ;
[ "¶" ; "" ] ; [ ".00" ; "" ] ) ) ; ".-0123456789" )
      +  $cost

    /*
    quote data.item::Item list price manual
    quote data.item::Item description
    quote data.item::Item number
    quote data.item::Item part number
    quote data.item::Item part number SKU
    quote data.item::Item quantity

    Part#
    SKU#
    Desc1
    Desc2
    Qty
    List
    Cost

    Filter ( Trim ( Substitute( importer.item::sys.importer.fields [ $count2 ] ; [ "¶" ; "" ] ; [ ".00" ; "" ] ) ) ; "0123456789" )
    */ ]

Set Field [ quote data.item::Item list price manual; Filter ( Trim ( Substitute( importer.item::sys.importer.fields [ $rlist ] ;
[ "¶" ; "" ] ; [ ".00" ; "" ] ) ) ; ".-0123456789" )
      + $list

    /*
    quote data.item::Item list price manual
    quote data.item::Item description
    quote data.item::Item number
    quote data.item::Item part number
    quote data.item::Item part number SKU
    quote data.item::Item quantity

    Part#
    SKU#
    Desc1
    Desc2
    Qty
    List
    Cost

    Filter ( Trim ( Substitute( importer.item::sys.importer.fields [ $count2 ] ; [ "¶" ; "" ] ; [ ".00" ; "" ] ) ) ; "0123456789" )
    */ ]

Set Field [ quote data.item::Item description; Case(
    IsEmpty( importer.item::sys.importer.fields [ $rdesc1 ] ) and IsEmpty( importer.item::sys.importer.fields
    [ $rdesc2 ] ) ; "" ;
    IsEmpty( importer.item::sys.importer.fields [ $rdesc2 ] ) ; importer.item::sys.importer.fields [ $rdesc1 ] ;
    IsEmpty( importer.item::sys.importer.fields [ $rdesc1 ] ) ; importer.item::sys.importer.fields [ $rdesc2 ] ;
    importer.item::sys.importer.fields [ $rdesc1 ] & ", " & importer.item::sys.importer.fields [ $rdesc2 ]
    ) &
    Case(
    IsEmpty( $desc1 ) and IsEmpty( $desc2) ; "" ;
    IsEmpty( $desc2) ; "¶"  & $desc1 ;
    IsEmpty( $desc1) ; "¶"  & $desc2 ;
    "¶"  & $desc1 & ", "  & $desc2
    )

    /*
    importer.item::sys.importer.fields [ $count2 ]

    quote data.item::Item list price manual
    quote data.item::Item description
    quote data.item::Item number
```

quote data.item::Item part number
quote data.item::Item part number SKU
quote data.item::Item quantity

Part#
SKU#
Desc1
Desc2
Qty
List
Cost

Filter (Trim (Substitute(importer.item::sys.importer.fields [$count2] ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789")
*/]

Set Field [quote data.item::Item part number; importer.item::sys.importer.fields [$rpart]

/*
importer.item::sys.importer.fields [$count2]

quote data.item::Item list price manual
quote data.item::Item description
quote data.item::Item number
quote data.item::Item part number
quote data.item::Item part number SKU
quote data.item::Item quantity

Part#
SKU#
Desc1
Desc2
Qty
List
Cost

Filter (Trim (Substitute(importer.item::sys.importer.fields [$count2] ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789")
*/]

Set Field [quote data.item::Item part number SKU; importer.item::sys.importer.fields [$rsku]

/*
importer.item::sys.importer.fields [$count2]

quote data.item::Item list price manual
quote data.item::Item description
quote data.item::Item number
quote data.item::Item part number
quote data.item::Item part number SKU
quote data.item::Item quantity

Part#
SKU#
Desc1
Desc2
Qty
List
Cost

Filter (Trim (Substitute(importer.item::sys.importer.fields [$count2] ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789")
*/]

Set Field [quote data.item::Item quantity; $qty

/*
importer.item::sys.importer.fields [$count2]

quote data.item::Item list price manual
quote data.item::Item description
quote data.item::Item number
quote data.item::Item part number
quote data.item::Item part number SKU
quote data.item::Item quantity

Part#
SKU#
Desc1

```
            Desc2
            Qty
            List
            Cost

            Filter ( Trim ( Substitute( importer.item::sys.importer.fields [ $count2 ] ; [ "¶" ; "" ] ; [ ".00" ; "" ] ) ) ; "0123456789" )
            */ ]
        Set Variable [ $cost; Value:"" ]
        Set Variable [ $list; Value:"" ]
        Set Variable [ $desc1; Value:"" ]
        Set Variable [ $desc2; Value:"" ]
    Else
        Set Variable [ $cost; Value:Case(
            IsEmpty( importer.item::sys.importer.bundle cost ) and IsEmpty (importer.item::sys.importer.bundle desc ) ; $cost ;
            ($qty * Filter ( Trim ( Substitute( importer.item::sys.importer.fields [ $rcost ] ; [ "¶" ; "" ] ; [ ".00" ; "" ] ) ) ) ;
            "-.0123456789" ) )+ $cost
            ) ]
        Set Variable [ $list; Value:Case(
            IsEmpty( importer.item::sys.importer.bundle cost ) and IsEmpty (importer.item::sys.importer.bundle desc ) ; $list ;
            ($qty *  Filter ( Trim ( Substitute( importer.item::sys.importer.fields [ $rlist ] ; [ "¶" ; "" ] ; [ ".00" ; "" ] ) ) ) ;
            "-.0123456789" ) )  + $list
            ) ]
        Set Variable [ $desc1; Value:Case(
            IsEmpty (importer.item::sys.importer.bundle desc ) ; $desc1 ;
            Case( $qty > 1 ; "(" & $qty & ") " ; "" ) & importer.item::sys.importer.fields [ $rdesc1 ]   & If( IsEmpty( $desc1 ) ; "" ;
            "¶"  & $desc1 )
            ) ]
        Set Variable [ $desc2; Value:Case(
            IsEmpty (importer.item::sys.importer.bundle desc ) ; $desc2 ;
            importer.item::sys.importer.fields [ $rdesc2 ]  & If( IsEmpty( $desc2 ) ; "" ; "¶"  & $desc2 )
            ) ]
    End If
    Commit Records/Requests
        [ Skip data entry validation; No dialog ]
End Loop
If [ Case(
    not IsEmpty( $cost ) ; 1 ;
    not IsEmpty( $list ) ; 1 ;
    not IsEmpty( $desc1 ) ; 1 ;
    not IsEmpty( $desc2 ) ; 1 ;
    ""
    ) ]
    Set Field [ quotes::ID quote data.item; Right( Random ; 7 )  &
        Right( Random ;6) ]
    Set Field [ quote data.item::ID number internal; quotes::ID quote data.item ]
    Set Field [ quote data.item::Item number; $in + .5 ]
    Commit Records/Requests
        [ Skip data entry validation; No dialog ]
    Set Field [ quote data.item::Item cost manual; $cost ]
    Set Field [ quote data.item::Item list price manual; $list ]
    Set Field [ quote data.item::Item description; $desc1  & Case( IsEmpty( $desc2) ; "" ; ", " & $desc2) ]
    Commit Records/Requests
        [ Skip data entry validation; No dialog ]
End If
Close Window [ Current Window ]
If [ $warn = 1 ]
    Show Custom Dialog [ Title: "Warning"; Message: Case(
        $warn > 1 ; "Due to bundling, " & $warn & " items were reduced to quantity 1." ;
        "Due to bundling, " & $warn & " item was reduced to quantity 1."
        ); Buttons: “OK” ]
```

```
End If
    If [ 1 = 2 ]
        Refresh Window
            [ Flush cached join results; Flush cached external data ]
    End If
End If
Commit Records/Requests
    [ Skip data entry validation; No dialog ]
Exit Script [  ]
Show Custom Dialog [ Message: $c & "¶" &
    $dc & "¶" &
    quotes::ID importer.item & "¶" &
    importer.item::sys.importer.active  & "¶" &
    "val "  &  importer.item::sys.importer.fields[$count2] & "¶" &
    "$desc1"  & $desc1 & "¶" &
    "$desc1"  & $desc2 & "¶" &
    "qd desc"  & quote data.item::Item description & "¶" &
    ""  & "¶" &
    ""  & "¶" &
    ""  & "¶"; Buttons: “OK”, “Cancel” ]
Show Custom Dialog [ Message: $c & "¶" &
    $dc & "¶" &
    quotes::ID importer.item & "¶" &
    importer.item::sys.importer.active  & "¶" &
    "des " &  importer.item::sys.importer.bundle desc& "¶" &
    "val "  &  importer.item::sys.importer.fields[$count2] & "¶" &
    "des$"  & $desc1 & "¶" &
    ""  & "¶" &
    "qs"  & quote data::Item description & "¶" &
    ""  & "¶" &
    ""  & "¶" &
    ""  & "¶"; Buttons: “OK”, “Cancel” ]
```

Importer: Checkboxes

```
Set Variable [ $gs; Value:Get(ScriptParameter) ]
If [ $gs = "active" ]
    Set Field [ importer::sys.importer.bundle cost; "" ]
    Set Field [ importer::sys.importer.bundle desc; "" ]
End If
If [ $gs = "d" ]
    Set Field [ importer::sys.importer.bundle cost; "" ]
    Set Field [ importer::sys.importer.active; "" ]
End If
If [ $gs = "c" ]
    Set Field [ importer::sys.importer.bundle desc; "" ]
    Set Field [ importer::sys.importer.active; "" ]
End If
Commit Records/Requests
    [ Skip data entry validation; No dialog ]
Refresh Window
```

Importer: Clear selections

Set Field [quotes::sys.importer.selection; ""]
Set Field [quotes::sys.importer.selection[2]; ""]
Set Field [quotes::sys.importer.selection[3]; ""]
Set Field [quotes::sys.importer.selection[4]; ""]
Set Field [quotes::sys.importer.selection[5]; ""]
Set Field [quotes::sys.importer.selection[6]; ""]
Set Field [quotes::sys.importer.selection[7]; ""]
Set Field [quotes::sys.importer.selection[8]; ""]
Set Field [quotes::sys.importer.selection[9]; ""]
Set Field [quotes::sys.importer.selection[10]; ""]
Set Field [quotes::sys.importer.selection[11]; ""]
Set Field [quotes::sys.importer.selection[12]; ""]
Commit Records/Requests
[Skip data entry validation; No dialog]

Commit Records/Requests
[Skip data entry validation; No dialog]

Close Window [Current Window]

Line items: New line items from scratch field

Set Variable [$gs; Value:If(IsEmpty(Get (ScriptParameter)) ; 1; Get(ScriptParameter))]

Set Variable [$ll; Value:Filter (
TextFormatRemove (
Substitute (
quotes::Quote scratch field[$gs] ; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"]; ["¶¶" ; "¶"]; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"]
)
) ;
" ~!@#$%^&*()_+`1234567890-=QWERTYUIOP{}lqwertyuiop[]\ASDFGHJKL:\"asdfghjkl;'ZXCVBNM<>?zxcvbnm,./°• ¶"
)]

Set Variable [$ac; Value:quotes::xxIDL_IDNumberInternalWithRev]

New Window [Name: "~temp~"; Height: 1; Width: 1; Top: -300]

Go to Layout [“data.quote data” (quote data)]

Loop

Exit Loop If [Case(
$count < 1 ; Let([
$count = 1 ;
$end = PatternCount($ll ; "¶")
] ; "") ;
$count ≥ $end or $end < 1; Let($count = 0 ; 1) ;
Let($count = $count + 1 ; "")
)]

Set Variable [$line; Value:Trim(Substitute(MiddleValues ($ll ; $count ; 1) ; ["¶" ; ""] ; [" " ; "¶"] ; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"]))]

If [PatternCount ($line ; "¶") > 1]

New Record/Request

Set Field [quote data::ID quote number w Rev; $ac]

Set Field [quote data::Item part number; Trim (Substitute(MiddleValues ($line ; 1; 1) ; ["¶" ; ""]))]

Set Field [quote data::Item description; Trim (Substitute(MiddleValues ($line ; 2; 1) ; ["¶" ; ""]))]

Set Field [quote data::Item quantity; Trim (Substitute(MiddleValues ($line ; 3; 1) ; ["¶" ; ""]))]

Set Field [quote data::Item cost manual; Filter (Trim (Substitute(MiddleValues ($line ; 4; 1) ; ["¶" ; ""] ; [".00" ; ""])) ;
"0123456789")]

Set Field [quote data::Item list price manual; Let([
a = Filter (Trim (Substitute(MiddleValues ($line ; 5; 1) ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789") ;
b = Filter (Trim (Substitute(MiddleValues ($line ; 5; 1) ; ["¶" ; ""] ; [".00" ; ""])) ; "0123456789")
] ;
Case (
IsEmpty(a) ; b ;
a
)
)]

End If

End Loop

Commit Records/Requests
[Skip data entry validation; No dialog]

Close Window [Current Window]

Line items: New line items from scratch field List

```
Set Variable [ $gs; Value:If( IsEmpty( Get (ScriptParameter ) ) ; 1; Get( ScriptParameter ) ) ]
Set Variable [ $ll; Value:Filter (
    TextFormatRemove (
     Substitute (
      quotes::Quote scratch field[ $gs ] ;  [ "¶¶" ; "¶" ] ; [ "¶¶" ; "¶" ] ; [ "¶¶" ; "¶" ]; [ "¶¶" ; "¶" ]; [ "¶¶" ; "¶" ] ; [ "¶¶" ; "¶" ] ; [ "¶¶" ; "¶" ]
     )
    ) ;
"     ~!@#$%^&*()_+`1234567890-=QWERTYUIOP{}lqwertyuiop[]\ASDFGHJKL:\"asdfghjkl;'ZXCVBNM<>?zxcvbnm,./°• ¶"
) ]
Set Variable [ $ac; Value:quotes::xxIDL_IDNumberInternalWithRev ]
New Window [ Name: "~temp~"; Height: 1; Width: 1; Top: -300 ]
Go to Layout [ “data.quote data” (quote data) ]
Loop
    Exit Loop If [ Case(
        $count < 1 ; Let( [
                $count = 1 ;
                $end = PatternCount( $ll ; "¶" )
        ] ; "" ) ;
        $count  ≥ $end or $end < 1;          Let( $count = 0 ; 1) ;
        Let( $count = $count + 1 ; "" )
        ) ]
    Set Variable [ $line; Value:Trim( Substitute( MiddleValues ( $ll ; $count ; 1 ) ; [ "¶" ; "" ] ; [ "       " ; "¶" ] ) ) ]
    If [ PatternCount ( $line ; "¶" ) > 1 ]
        New Record/Request
        Set Field [ quote data::ID quote number w Rev; $ac ]
        Set Field [ quote data::Item part number; Trim ( Substitute( MiddleValues ( $line ; 1; 1 ) ; [ "¶" ; "" ] ) ) ]
        Set Field [ quote data::Item description; Trim ( Substitute( MiddleValues ( $line ; 2; 1 ) ; [ "¶" ; "" ] ) ) ]
        Set Field [ quote data::Item quantity; Trim ( Substitute( MiddleValues ( $line ; 3; 1 ) ; [ "¶" ; "" ] ) ) ]
        Set Field [ quote data::Item cost manual; Filter ( Trim ( Substitute( MiddleValues ( $line ; 4; 1 ) ; [ "¶" ; "" ] ; [ ".00" ; "" ] ) ) ;
            "0123456789" ) ]
        Set Field [ quote data::Item list price manual; Filter ( Trim ( Substitute( MiddleValues ( $line ; 5; 1 ) ; [ "¶" ; "" ] ; [ ".00" ;
            "" ] ) ) ; "0123456789" ) ]
    End If
End Loop
Close Window [ Current Window ]
```

Line items: Clean up scratch field

Set Variable [$gs; Value:If(IsEmpty(Get (ScriptParameter)) ; 1; Get(ScriptParameter))]

Set Field [quotes::Quote scratch field[$gs]; Substitute(
Filter (
TextFormatRemove (
Substitute (
quotes::Quote scratch field[$gs] ; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"]; ["¶¶" ; "¶"]; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"] ; ["¶¶" ; "¶"]
)
) ;
" ~!@#$%^&*()_+`1234567890-=QWERTYUIOP{}lqwertyuiop[]\ASDFGHJKL:\"asdfghjkl;'ZXCVBNM<>?zxcvbnm,./°• ¶"
)
; [Char(9) & Char(9) & Char(9) ; Char(9)] ;[Char(9) & Char(9) & Char(9) ; Char(9)] ;[Char(9) & Char(9) ; Char(9)] ;[Char(9) & Char(9) ; Char(9)] ;[Char(9) & Char(9) ; Char(9)] ;[Char(9) & Char(9) ; Char(9)] ;[Char(9) & Char(9) ; Char(9)] ;[Char(9) & Char(9) ; Char(9)]
)]

Set Error Capture [On]

Freeze Window

Show Custom Dialog [Title: "Delete Line Item"; Message: "Delete line item..."; Buttons: “Continue”, “Cancel”]

If [Get(LastMessageChoice) = 2]

    Exit Script [ ]

End If

Delete Portal Row
    [No dialog]

Commit Records/Requests
    [Skip data entry validation; No dialog]

Select Window [Current Window]

Freeze Window

Set Field [quote data::sys.line item total print choice; If(quote data::sys.line item total print choice = "Total Here";"";"Total Here")]

Commit Records/Requests

[Skip data entry validation; No dialog]

Line items: paste copied line items

Freeze Window

Set Error Capture [On]

Set Variable [$b; Value:Let([
$$newID[1] = quotes::xxIDL_IDNumberInternal ;
$$newID[2] = Right(quotes::xxIDL_IDNumberInternalWithRev;3);
$$newID[3] = "end"
];"")]

Go to Related Record [From table: “quote data.copy mark”; Using layout: “data.quote data” (quote data)]
[Show only related records]

If [Get(LastError) = 0]

Perform Script [“duplicate multiple line items ($$newID[1]=ac, [2]=rev#,[3]=""/end)”]

End If

Go to Layout [original layout]

Commit Records/Requests
[Skip data entry validation; No dialog]

Freeze Window

Set Error Capture [On]

If [Get(WindowMode) = 0]

    Go to Related Record [From table: “quote data”; Using layout: “data.quote data” (quote data)]
        [Show only related records]

    If [Get(LastError) = 0]

        Go to Layout [“data.quote data” (quote data)]

        Sort Records [Specified Sort Order: quote data::Item number; ascending]
            [Restore; No dialog]

        Go to Record/Request/Page
            [First]

        Replace Field Contents [quote data::Item number; Replace with serial numbers: Custom values; Initial value: 1; Increment value: 1]
            [No dialog]

    End If

    Go to Layout [original layout]

End If

Exit Script [ ]

Commit Records/Requests

Line items: copy down [renumber, quantity, cost discount, price discount, template cost, cost plus]

```
If [ Get( WindowMode)  ≠  0 ]
    Exit Script [  ]
End If
Set Variable [ $pr; Value:Get(ActivePortalRowNumber) ]
Allow User Abort [ Off ]
Set Error Capture [ On ]
Freeze Window
If [ Get(ScriptParameter) = "cost plus" ]
    Commit Records/Requests
        [ Skip data entry validation; No dialog ]
    Go to Object [ Object Name: "field.cost plus" ]
    Go to Portal Row [ $pr ]
        [ No dialog ]
    Set Field [ quote data::Item price manual; quote data::Item desired cost plus price ]
    Go to Object [ Object Name: "field.cost plus" ]
    Select All
    Exit Script [  ]
End If
If [ Get(ScriptParameter) = "renumber" ]
    If [ Get( WindowMode) = 0 ]
        Go to Related Record [ From table: “quote data”; Using layout: “data.quote data” (quote data) ]
            [ Show only related records ]
        If [ Get(LastError) = 0 ]
            Go to Layout [ “data.quote data” (quote data) ]
            Sort Records [ Specified Sort Order: quote data::Item number; ascending ]
                [ Restore; No dialog ]
            Go to Record/Request/Page
                [ First ]
            Replace Field Contents [ quote data::Item number; Replace with serial numbers: Custom values; Initial value: 1;
                Increment value: 1 ]
                [ No dialog ]
            Replace Field Contents [ quote data::Item header text; Replace with calculation: If( quote data::Item header text =
                "none"; ""; quote data::Item header text)  //clean up field ]
                [ No dialog ]
        End If
        Go to Layout [ original layout ]
    End If
    Commit Records/Requests
    Exit Script [  ]
End If
If [ Get(ScriptParameter) = "template cost" ]
    Set Variable [ $v; Value:Template.list view::Item cost discount manual ]
    Commit Records/Requests
        [ Skip data entry validation; No dialog ]
    Show Custom Dialog [ Title: "Copy"; Message: "Copy the selected (or first if none selected) cost discount to all line items.¶ALSO
        clear all of the MANUALLY entered costs?¶(the costs will then be calculated from the discount)"; Buttons: “Copy”,
        “Copy/Clear”, “Cancel” ]
    If [ Get(LastMessageChoice) = 3 ]
        Exit Script [  ]
    End If
    Go to Related Record [ From table: “Template.list view”; Using layout: “data.templates” (Template item) ]
        [ Show only related records; New window ]
```

```
        If [ Get(LastError)  = 0 ]
        Replace Field Contents [ Template item::Item cost discount manual; Replace with calculation: $v ]
            [ No dialog ]
        If [ Get(LastMessageChoice) = 2 ]
            Replace Field Contents [ Template item::Item cost manual; Replace with calculation: "" ]
                [ No dialog ]
        End If
        Close Window [ Current Window ]
    End If
    Exit Script [  ]
End If
Set Variable [ $pr; Value:Get(ActivePortalRowNumber) ]
Set Variable [ $b; Value:Let([ $s = Get(ScriptParameter);
    $v = Case(
      $s = "quantity"; quote data::Item quantity;
      $s =  "price discount"; quote data::Item price discount manual;
      $s =  "cost discount" ; quote data::Item cost discount manual
    )
    ]; "") ]
Commit Records/Requests
    [ Skip data entry validation; No dialog ]
If [ $s = "quantity" ]
    Show Custom Dialog [ Title: "Copy Quantity"; Message: "Copy the selected (or first if none selected) quantity to all line items?";
        Buttons: “Copy”, “Cancel” ]
Else If [ $s = "price discount" ]
    Show Custom Dialog [ Title: "Copy"; Message: "Copy the selected (or first if none selected) price discount to all line items.¶ALSO
        clear all of the MANUALLY entered prices?¶(the prices will then be calculated from the discount)"; Buttons: “Copy”,
        “Copy/Clear”, “Cancel” ]
Else
    Show Custom Dialog [ Title: "Copy"; Message: "Copy the selected (or first if none selected) cost discount to all line items.¶ALSO
        clear all of the MANUALLY entered costs?¶(the costs will then be calculated from the discount)"; Buttons: “Copy”,
        “Copy/Clear”, “Cancel” ]
End If
If [ Get(LastMessageChoice) = 3 ]
    Exit Script [  ]
End If
Go to Related Record [ From table: “quote data”; Using layout: “data.quote data” (quote data) ]
    [ Show only related records; New window ]
If [ Get(LastError)  = 0 ]
    Replace Field Contents [ quote data::Item quantity; Replace with calculation: If($s = "quantity";$v;quote data::Item quantity) ]
        [ No dialog ]
    Replace Field Contents [ quote data::Item price discount manual; Replace with calculation: If($s = "price discount";$v;quote
        data::Item price discount manual) ]
        [ No dialog ]
    Replace Field Contents [ quote data::Item cost discount manual; Replace with calculation: If($s = "cost discount";$v;quote data::
        Item cost discount manual) ]
        [ No dialog ]
    If [ Get(LastMessageChoice) = 2 ]
        Replace Field Contents [ quote data::Item cost manual; Replace with calculation: If($s = "cost discount";"";quote data::Item
            cost manual) ]
            [ No dialog ]
        Replace Field Contents [ quote data::Item price manual; Replace with calculation: If($s = "price discount";"";quote data::Item
            price manual) ]
            [ No dialog ]
    End If
    Close Window [ Current Window ]
```

End If

Exit Script [ ]

```
Set Error Capture [ On ]
Freeze Window
Go to Related Record [ From table: “quote data”; Using layout: “data.quote data” (quote data) ]
    [ Show only related records ]
If [ Get(LastError) = 0 ]
    Set Variable [ $IN; Value:quote data::Item number ]
    Duplicate Record/Request
    Set Variable [ $intID; Value:Right( "00000000"  & Left(Int(Random*9999999);7);7 )  &
        Right("000000" & Get(RecordID);6) ]
    #
    #duplicate calc items
    If [ quote data::ID number internal = calc.item::ID calc quote data number ]
        Go to Related Record [ From table: “calc.item”; Using layout: “data.quote data” (quote data) ]
            [ Show only related records; New window ]
        If [ Get(LastError) = 0 ]
            Unsort Records
            Go to Record/Request/Page
                [ Last ]
            Loop
                Duplicate Record/Request
                Set Field [ quote data::ID calc quote data number; $intID ]
                Omit Record
                Omit Record
                Exit Loop If [ Get( FoundCount) = 0 ]
            End Loop
        End If
        Close Window [ Current Window ]
    End If
    #
    Set Field [ quote data::ID number internal; $intID ]
    Set Field [ quote data::Item number; $IN +  .1 ]
End If
Go to Layout [ original layout ]
```

Line items: duplicate multiple line items ($$newID[1]=ac, [2]=rev#,[3]=""/end)

```
Set Error Capture [ On ]
If [ Get( FoundCount) > 0 ]
    Sort Records [ Specified Sort Order: quote data::Item number; descending ]
        [ Restore; No dialog ]
    Go to Record/Request/Page
        [ Last ]
    Loop
        Set Variable [ $IN; Value:quote data::Item number ]
        Duplicate Record/Request
        Set Variable [ $intID; Value:Right( "00000000"  & Left(Int(Random*9999999);7);7 )  &
            Right("000000" & Get(RecordID);6) ]
        #duplicate calc items
        If [  Count(calc.item::ID number internal ) > 0 ]
            Go to Related Record [ From table: “calc.item”; Using layout: “data.quote data” (quote data) ]
                [ Show only related records; New window ]
            If [ Get(LastError) = 0 ]
                Unsort Records
                Go to Record/Request/Page
                    [ Last ]
                Loop
                    Duplicate Record/Request
                    Set Field [ quote data::ID calc quote data number; $intID ]
                    Omit Record
                    Omit Record
                    Exit Loop If [ Get( FoundCount) = 0 ]
                End Loop
            End If
            Close Window [ Current Window ]
        End If
        #
        Set Field [ quote data::ID quote number w Rev; $$newID[1] & "R"  & Right( "00"  &  GetAsText( $$newID[2] );2) ]
        Set Field [ quote data::ID number internal; $intID ]
        Set Field [ quote data::Item number; If($$newID[3] = "end";quote data::Item number;$IN) ]
        Omit Record
        Omit Record
        Exit Loop If [ Get( FoundCount) = 0 ]
    End Loop
End If
```

New Window [ ]
Go to Layout [“Quote Entry” (quotes)]
Perform Script [“window resize [centered, full]”; Parameter: "full"]
Halt Script

Set Variable [$sp; Value:If(Get(LayoutName)= "Quote entry"; "full" ; Get(ScriptParameter))]

Show/Hide Status Area
[Hide]

Show/Hide Text Ruler
[Hide]

Set Zoom Level
[100%]

Move/Resize Window [Current Window; Height: If($sp = "full"; Get (WindowDesktopHeight) - 180 ;
""); Width: If(Get(WindowWidth) > 1130; 979;""); Top: If(Get(ScriptParameter)= "centered" ; ((Get (WindowDesktopHeight) - Get (WindowContentHeight)) / 2) ;

Let($n = PatternCount(WindowNames ; "¶");
Case(
IsEmpty(WindowNames) ; 60;
$n < 8 ; 60 + (PatternCount(WindowNames ; "¶")) * 15 ;
$n < 16 ; 60 + (PatternCount(WindowNames ; "¶") -8) * 15 ;
60 + (PatternCount(WindowNames ; "¶") -16) * 15
))); Left: If(Get(ScriptParameter)= "centered" ; ((Get (WindowDesktopWidth) - Get (WindowWidth)) / 2) - 160;

30 + If(IsEmpty(WindowNames) ; ""; (PatternCount(WindowNames ; "¶")) * 20)

)]

Adjust Window
[Resize to Fit]

Move/Resize Window [Current Window; Height: If($sp = "full"; Get (WindowDesktopHeight) - 180 ;
""); Width: If(Get(WindowWidth) > 1130; 979;""); Top: If(Get(ScriptParameter)= "centered" ; ((Get (WindowDesktopHeight) - Get (WindowContentHeight)) / 2) ;

Let($n = PatternCount(WindowNames ; "¶");
Case(
IsEmpty(WindowNames) ; 60;
$n < 8 ; 60 + (PatternCount(WindowNames ; "¶")) * 15 ;
$n < 16 ; 60 + (PatternCount(WindowNames ; "¶") -8) * 15 ;
60 + (PatternCount(WindowNames ; "¶") -16) * 15
))); Left: If(Get(ScriptParameter)= "centered" ; ((Get (WindowDesktopWidth) - Get (WindowWidth)) / 2) - 160;

30 + If(IsEmpty(WindowNames) ; ""; (PatternCount(WindowNames ; "¶")) * 20)

)]

Open price lists/save [save]

If [Get(ScriptParameter) = "save"]

    Set Field [preferences::user.sys.global note position; Get(WindowHeight)]

    Set Field [preferences::user.sys.global note position[2]; Get(WindowWidth)]

    Set Field [preferences::user.sys.global note position[3]; Get(WindowTop)]

    Set Field [preferences::user.sys.global note position[4]; Get(WindowLeft)]

    Exit Script [ ]

End If

New Window [Name: "Price Lists"; Height: preferences::user.sys.global note position[1]; Width: preferences::user.sys.global note position[2]; Top: preferences::user.sys.global note position[3]; Left: preferences::user.sys.global note position[4]]

Go to Layout [<unknown>]

Freeze Window

Commit Records/Requests
[Skip data entry validation; No dialog]

Enter Browse Mode

Set Field [print::print.preview image; ""]

New Window [Name: "Print"; Height: 668; Width: 806; Top: 180; Left: 285]

Go to Layout [“Print” (quotes)]

Perform Script [“window resize [centered, full]”; Parameter: "centered"]

Freeze Window

Set Field [print::sys.selected; Case(
not IsEmpty(quotes::Quote print layout) ; quotes::Quote print layout;
not IsEmpty(print::sys.selected); print::sys.selected;
print::print.layout exact name
)]

Perform Script [“Rebuild list”; Parameter: "refresh picture"]

Perform Script [“Select print layout/refresh picture”; Parameter: "refresh picture"]

```
Set Error Capture [ On ]
Freeze Window
Set Variable [ $count; Value:Let($layout = print::sys.selected;1) ]
New Window [ Height: 1; Width: 1; Top: -400 ]
Go to Layout [ “data.quotes” (quotes) ]
Copy [ quotes::Quote save to file name ]
    [ Select ]
Set Variable [ $file; Value:"file:../Quotes Orders/"  &
    quotes::Quote save to file name  & ".pdf" ]
Go to Layout [ original layout ]
#get quote data line items if printing a quote sheet
If [ Left(print::sys.selected;3) = "QSH" ]
    Go to Related Record [ From table: “quote data”; Using layout: “data.quote data” (quote data) ]
        [ Show only related records ]
    Perform Script [ “Prep quote items for print” ]
End If
#goto the layout to print
Loop
    Go to Layout [ $count ]
    Exit Loop If [ Get(LayoutName) = $layout or $count > 500 ]
    Set Variable [ $count; Value:$count + 1 ]
End Loop
If [ Get(LayoutName)  ≠  $layout ]
    Show Custom Dialog [ Title: "Error"; Message: "Layout does not appear to exist"; Buttons: “Cancel” ]
    Exit Script [  ]
End If
If [ print::sys.setting.all one  =  "Entire Set" or Left(Get(LayoutName);3)  =  "QSH" ]
    If [ Get(ScriptParameter) = "pdf" ]
        Save Records as PDF [ Records being browsed ]
        [ Document -  Title: "Quote " & quotes::Quote number Print; Subject: quotes::Quote summary; Author: preferences::user.
            company & ", " & preferences::user.name; Keywords: quotes::Quote summary; Compatibility: Acrobat 5 and later ]
        [ Pages -  Number Pages From: 1; Include: All pages ]
        [ Security -  Printing: High Resolution; Editing: Any except extracting pages; Enable copying; Enable Screen Reader ]
        [ Initial View -  Show: Page Only; Page Layout: Single Page; Magnification: 100% ]
            [ Restore; No dialog ]
        Set Variable [ $error; Value:Get ( LastError ) ]
    Else If [ Get(ScriptParameter) = "quick pdf" ]
        Save Records as PDF [ File Name: “$file”; Records being browsed ]
        [ Document -  Title: "Quote " & quotes::Quote number Print; Subject: quotes::Quote summary; Author: preferences::user.
            company & ", " & preferences::user.name; Keywords: quotes::Quote summary; Compatibility: Acrobat 5 and later ]
        [ Pages -  Number Pages From: 1; Include: All pages ]
        [ Security -  Printing: High Resolution; Editing: Any except extracting pages; Enable copying; Enable Screen Reader ]
        [ Initial View -  Show: Page Only; Page Layout: Single Page; Magnification: 100% ]
            [ Restore; No dialog ]
        Set Variable [ $error; Value:Get ( LastError ) ]
    Else
        Print [ Records being browsed; All Pages; Orientation: Portrait; Paper size: 8.5" x 11" ]
            [ Restore ]
        Set Variable [ $error; Value:Get ( LastError ) ]
    End If
Else
    If [ Get(ScriptParameter) = "pdf" ]
        Save Records as PDF [ Current record ]
```

[*Document* - Title: "Quote " & quotes::Quote number Print; Subject: quotes::Quote summary; Author: preferences::user.company & ", " & preferences::user.name; Keywords: quotes::Quote summary; Compatibility: Acrobat 5 and later]
[*Pages* - Number Pages From: 1; Include: All pages]
[*Security* - Printing: High Resolution; Editing: Any except extracting pages; Enable copying; Enable Screen Reader]
[*Initial View* - Show: Page Only; Page Layout: Single Page; Magnification: 100%]
[Restore; No dialog]

Set Variable [$error; Value:Get (LastError)]

Else If [Get(ScriptParameter) = "quick pdf"]

Save Records as PDF [File Name: “$file”; Records being browsed]
[*Document* - Title: "Quote " & quotes::Quote number Print; Subject: quotes::Quote summary; Author: preferences::user.company & ", " & preferences::user.name; Keywords: quotes::Quote summary; Compatibility: Acrobat 5 and later]
[*Pages* - Number Pages From: 1; Include: All pages]
[*Security* - Printing: High Resolution; Editing: Any except extracting pages; Enable copying; Enable Screen Reader]
[*Initial View* - Show: Page Only; Page Layout: Single Page; Magnification: 100%]
[Restore; No dialog]

Set Variable [$error; Value:Get (LastError)]

Else

Show All Records

Omit Record

Show Omitted Only

Print [Current record; All Pages; Orientation: Portrait; Paper size: 8.5" x 11"]
[Restore]

Set Variable [$error; Value:Get (LastError)]

End If

End If

Close Window [Current Window]

Set Field [quotes::Quote print layout; print::sys.selected]

#record note and history

If [$error = 0 and quotes::Quote date printed ≠ Get(CurrentDate)]

Set Field [quotes::Quote notes system; If(quotes::Quote date printed = Get(CurrentDate);quotes::Quote notes system; Substitute(Substitute(GetAsText(Get(CurrentDate)) ;"200";"0") ;"201";"1") & ", " & If(IsEmpty(preferences::sys.initials.my initials);"SYS"; preferences::sys.initials.my initials) & ", Printed quote Rev. " & revision::system rev letter)]

Set Field [window::xSys_NewItem; "new print"]

Set Field [window::xSys_NewItem[2]; quotes::xxIDL_IDNumberInternal & "¶" & quotes::xxIDL_LinkPrimaryItem]

Set Field [window::xSys_NewItem[3]; quotes::xxIDL_LinkPrimaryItem]

Set Field [window::xSys_NewItem[4]; "external"]

Set Field [window::xSys_NewItem[5]; "Printed quote " & quotes::Quote number Print & "; " & quotes::Quote first line item]

Set Field [window::xSys_NewItem[6]; quotes::xxIDL_IDNumberInternal]

Perform Script [“New [[1] type, [2] link to, [3] primaryID, [4] Portal, [5] Note, [6] Ref ID]” from file: “customers”]

End If

Set Field [quotes::Quote date printed; Get(CurrentDate)]

If [Get (LayoutName) = "Print"]

Close Window [Current Window]

End If

Exit Script [ ]

Show Custom Dialog [Message: Get (DesktopPath) & "¶¶" & Get (DocumentsPath) & "¶¶" & Get (FileMakerPath) & "¶¶" & Get (FilePath) & "¶¶" & Get (PreferencesPath) & "¶¶"; Buttons: “OK”, “Cancel”]

```
Freeze Window
Set Variable [ $count; Value:Let([$max = Get ( LayoutCount ); $temp = quotes::sys.print.key choice]; 1) ]
Loop
    Go to Layout [ $count ]
    Set Variable [ $layout; Value: Get(LayoutName) ]
    If [ Left($layout;3) = "QSH" or
        Left($layout;3) = "QLS" or
        Left($layout;3) = "QSY" ]
        Set Field [ contacts.primary::xSys_PrintKeyChoice; $layout ]
        Go to Layout [ “Quote Entry” (quotes) ]
        If [ $layout  ≠  print::print.layout exact name ]
            Go to Layout [ “data.print” (print) ]
            New Record/Request
            Set Field [ print::print.layout exact name; $layout ]
        End If
        Go to Layout [ “data.print” (print) ]
        Set Field [ print::print.mark for delete; "" ]
    End If
    Exit Loop If [ Let( $count = $count + 1; $count) > $max
         or $count > 1000 ]
End Loop
Set Error Capture [ On ]
#delete old layout records
Go to Layout [ “data.print” (print) ]
Enter Find Mode [  ]
Set Field [ print::print.mark for delete; 1 ]
Perform Find [  ]
If [ Get(FoundCount) > 0 ]
    Delete All Records
        [ No dialog ]
End If
Show All Records
Replace Field Contents [ print::print.mark for delete; Replace with calculation: 1 ]
    [ No dialog ]
Go to Layout [ original layout ]
Set Field [ quotes::sys.print.key choice; $temp ]
```

Print: Select print layout/refresh picture

```
Set Error Capture [ On ]
If [ Get ( ScriptParameter ) ≠ "refresh picture" ]
    Set Field [ print::sys.setting.all one; Case(
        Left(print::print.layout exact name;3) = "QSH"; "Current Record";
        "Entire Set"
            ) ]
    Set Field [ print::sys.selected; print::print.layout exact name ]
End If
New Window [ Height: 1; Width: 1; Top: -400 ]
#get quote data line items if printing a quote sheet
If [ Left(print::sys.selected;3) = "QSH" ]
    Go to Related Record [ From table: “quote data”; Using layout: “data.quote data” (quote data) ]
        [ Show only related records ]
    If [ Get(LastError) ≠ 0 ]
        Go to Layout [ “data.quote data” (quote data) ]
        Show All Records
        Omit Multiple Records [ 999999999 ]
            [ No dialog ]
    Else
        Perform Script [ “Prep quote items for print” ]
    End If
End If
#goto the layout to print
Set Variable [ $count; Value:1 ]
Loop
    Go to Layout [ $count ]
    Set Variable [ $first; Value:Case(
        $first > 0 ; $first ;
        Left(Get(LayoutName);3) = "QSH" ; $count ;
        $first
        ) ]
    Exit Loop If [ Get(LayoutName) = print::sys.selected or $count > 200 ]
    Set Variable [ $count; Value:$count + 1 ]
End Loop
If [ Get(LayoutName) ≠ print::sys.selected ]
    Go to Layout [ $first ]
    Set Field [ print::sys.selected; Get(LayoutName) ]
End If
If [ print::sys.setting.all one = "Entire Set" or Left(Get(LayoutName);3) = "QSH" ]
Else
    Show All Records
    Omit Record
    Show Omitted Only
End If
Enter Preview Mode
Copy [ ]
    [ Select ]
Close Window [ Current Window ]
If [ Left( print::sys.selected ; 3 ) = "QSH" ]
    Set Field [ quotes::Quote print layout; print::sys.selected ]
End If
Freeze Window
```

Print: Select print layout/refresh picture

Paste [print::print.preview image]
[Select]

Go to Object [Object Name: If(print::sys.setting.portrait landscapre = "Landscape"; "tab.landscape";"tab.portrait")]

Commit Records/Requests
[Skip data entry validation; No dialog]

Print: Prep quote items for print

```
Set Error Capture [ On ]
If [ Get(FoundCount) > 0 and Get(FoundCount) < 400 ]
    Go to Layout [ “data.quote data” (quote data) ]
    Set Variable [ $t; Value:Let([
        $to[1] = "" ;  //list price total
        $to[2] = "" ; //price total
        $to[3] = "" ; // cost total
        $b = ""
        ]; 1 ) ]
    Sort Records [ Specified Sort Order: quote data::Item number; ascending ]
        [ Restore; No dialog ]
    Go to Record/Request/Page
        [ First ]
    Loop
        Set Variable [ $b; Value:Let([
            $to[1] = $to[1] +  quote data::Item list price extended;  //list price total
            $to[2] = $to[2] + quote data::Item price extended  ; //price total
            $to[3] = $to[3] + quote data::Item cost extended  // cost total
            ]; "" ) ]
        #clean up fields
        Set Field [ quote data::Item header text; If( quote data::Item header text = "none"; ""; quote data::Item header text)  //clean up
            field ]
        Set Field [ quote data::sys.line item total print choice; If(quote data::sys.line item total print choice = "none" ;"";quote data::
            sys.line item total print choice)   //clean up field ]
        Set Field [ quote data::sys.line item totals label; "" ]
        Set Field [ quote data::sys.print.quote total amount list; "" ]
        Set Field [ quote data::sys.print.quote total amount; "" ]
        Set Field [ quote data::sys.print.quote total amount cost; "" ]
        If [
            quote data::sys.line item total print choice = "Total Here" ]
            Set Field [ quote data::sys.print.quote total amount list; $to[1] ]
            Set Field [ quote data::sys.print.quote total amount; $to[2] ]
            Set Field [ quote data::sys.print.quote total amount cost; $to[3] ]
            Set Field [ quote data::sys.line item totals label; "Total:" ]
            Set Variable [ $b; Value:Let([
                $to[1] = "" ;  //list price total
                $to[2] = "" ; //price total
                $to[3] = ""  // cost total
                ]; 1 ) ]
        End If
        Go to Record/Request/Page
            [ Next; Exit after last ]
    End Loop
    Go to Record/Request/Page
        [ First ]
    Replace Field Contents [ quote data::Item number print; Replace with serial numbers: Custom values; Initial value: 1; Increment
        value: 1 ]
        [ No dialog ]
    Go to Layout [ original layout ]
End If
```

#

Exit Script [ ]

Go to Layout [<unknown>]

New Record/Request

Set Field [<Table Missing>; /*GetRepetition (notes::sys.pass variable;1)*/]

Set Field [<Table Missing>; /*GetRepetition (notes::sys.pass variable;2)*/]

Set Field [<Table Missing>; /*GetRepetition (notes::sys.pass variable;3)*/]

Set Field [<Table Missing>; Get(CurrentDate)]

Set Field [<Table Missing>; preferences::sys.initials.my initials]

If [/*notes::ID account number = notes to self::ID account number*/]

    Delete Record/Request
        [No dialog]

End If

#

Exit Script [ ]

Set Field [contacts.primary::<Field Missing>; /*preferences::sys.default.notepad entry & customers::<Field Missing> & "¶" & customers::System history*/]

Set Variable [$count; Value:/*Let($max = ValueCount(GetRepetition (notes::sys.pass variable ; 1)); 1)*/]

sort: Sort by Quote Number

Sort Records [Specified Sort Order: <Table Missing>; descending
<Table Missing>; descending
quotes::<Field Missing>; descending]
[Restore; No dialog]

Go to Record/Request/Page
[First]

sort: Sort by Date

Sort Records [Specified Sort Order: quotes::sys.date created; descending
quotes::sys.created time; descending]
[Restore; No dialog]

Go to Record/Request/Page
[First]

sort: Sort by Company

Sort Records [Specified Sort Order: quotes::Quote company name; ascending
quotes::sys.date created; descending
quotes::sys.created time; descending]
[Restore; No dialog]

Go to Record/Request/Page
[First]

sort: Sort by Status

Sort Records [Specified Sort Order: quotes::Quote status; based on value list: “Quote Status List”
quotes::sys.date created; descending
quotes::sys.created time; descending]
[Restore; No dialog]

Go to Record/Request/Page
[First]

sort: Sort by Total Ammount

Sort Records [Specified Sort Order: quotes::Total forecast price; descending
quotes::sys.date created; descending
quotes::sys.created time; descending]
[Restore; No dialog]

Go to Record/Request/Page
[First]

sort: Sort by Commision

Sort Records [Specified Sort Order: quotes::Total forecast commission; descending
quotes::sys.date created; descending
quotes::sys.created time; descending]
[Restore; No dialog]

Go to Record/Request/Page
[First]

Sort Records [Specified Sort Order: quotes::Quote summary; descending
quotes::sys.date created; descending
quotes::sys.created time; descending]
[Restore; No dialog]

Go to Record/Request/Page
[First]

sort: Sort by Close date

Sort Records [Specified Sort Order: quotes::Quote close year; based on value list: “year”
quotes::Quote close month; based on value list: “month”]
[Restore; No dialog]

Go to Record/Request/Page
[First]

sort: Sort by Probability

Sort Records [Specified Sort Order: quotes::Quote probability; descending
quotes::Quote close year; based on value list: “year”
quotes::Quote close month; based on value list: “month”]
[Restore; No dialog]

Go to Record/Request/Page
[First]

```
Set Error Capture [ On ]
Freeze Window
If [ Get( WindowMode) = 0 ]
    If [ Get(ScriptParameter) = "new" ]
        Commit Records/Requests
            [ Skip data entry validation; No dialog ]
        Set Field [ quotes::sys.rev number; quotes::sys.rev number max ]
        Set Variable [ $note; Value:revision::System rev quote notes print ]
        #duplicate line items,note: create ID number in variable before duplicating line items, set to field after duplicating line items
        Set Variable [ $b; Value:Let([
            $$newID[1] = quotes::xxIDL_IDNumberInternal ;

            $$newID[2] = quotes::sys.rev number max + 1   ;
            $$newID[3] = ""
            ];"") ]
        Go to Related Record [ From table: “quote data”; Using layout: “data.quote data” (quote data) ]
            [ Show only related records ]
        If [ Get(LastError) = 0 ]
            Perform Script [ “duplicate multiple line items ($$newID[1]=ac, [2]=rev#,[3]=""/end)” ]
        End If
        Go to Layout [ original layout ]
        #set new quote fileds to new values
        Set Field [ quotes::sys.rev number; quotes::sys.rev number max + 1 ]
        Set Field [ revision::System rev quote notes print; $note ]
        Set Field [ revision::System rev Quote date; Get(CurrentDate) ]
    End If
    If [ Get(ScriptParameter) = "select" ]
        Set Field [ quotes::sys.rev number; revision all revs::System rev number ]
    End If
End If
Commit Records/Requests
    [ Skip data entry validation; No dialog ]
```

```
Allow User Abort [ Off ]
Set Error Capture [ On ]
Commit Records/Requests
    [ Skip data entry validation; No dialog ]
If [ Get( WindowMode) = 0 ]
    If [ Count(revision all revs::ID quote number w Rev) < 2 ]
        Exit Script [  ]
    End If
    Show Custom Dialog [ Title: "Message"; Message: "Delete revision?"; Buttons: “Cancel”, “OK” ]
    If [ Get(LastMessageChoice) = 1 ]
        Exit Script [  ]
    End If
    If [ Count(quote data::ID number internal) > 0 ]
        Go to Related Record [ From table: “quote data”; Using layout: “data.quote data” (quote data) ]
            [ Show only related records ]
        If [ Get(LastError) = 0 and Get(LayoutName) = "data.quote data" and Get(FoundCount) < 100 ]
            Delete All Records
                [ No dialog ]
        End If
    End If
    Go to Layout [ original layout ]
    Go to Related Record [ From table: “revision”; Using layout: “data.revision” (revision) ]
        [ Show only related records ]
    If [ Get(LastError) = 0 and Get(LayoutName) = "data.revision" ]
        Delete Record/Request
            [ No dialog ]
    End If
    Go to Layout [ original layout ]
    Commit Records/Requests
        [ Skip data entry validation; No dialog ]
    Set Field [ quotes::sys.rev number; Max(revision all revs::System rev number) ]
End If
```

Freeze Window

Set Variable [$pr; Value:Get(ActivePortalRowNumber)]

Commit Records/Requests

If [Get(ScriptParameter) = "close"]

    If [Count(calc.item::ID number internal) = 1 and IsEmpty(calc.item::ID calc builder choice) and IsEmpty(calc.item::sys.builder.item 3 specific)]

        Go to Portal Row
            [Select; First]

        Delete Portal Row
            [No dialog]

    End If

    Close Window [Current Window]

    Exit Script [ ]

End If

Go to Object [Object Name: "portal.quote data"]

Go to Portal Row [$pr]
    [Select; No dialog]

Set Field [quote data::Item part number; quote data::Item part number]

Commit Records/Requests

Go to Related Record [From table: “quote data”; Using layout: “Item Builder” (quote data)]
    [Show only related records; New window]

If [Get(LastError) ≠ 0]

    Exit Script [ ]

End If

Go to Record/Request/Page [$pr]
    [No dialog]

Perform Script [“window resize [centered, full]”; Parameter: "centered"]

Set Field [quotes::sys.template.choice.category; If(IsEmpty(quote data::sys.calc.category choice); quotes::sys.template.choice.category ; quote data::sys.calc.category choice)]

If [Count(calc.item::ID number internal) = 0]

    Perform Script [“Calc items [new, update text, update numbers, copy temp prices, select]”; Parameter: "new"]

End If

Calc items [new, update text, update numbers, copy temp prices, select]

```
Freeze Window
Set Error Capture [ On ]
Set Field [ quote data::sys.calc.category choice;   quotes::sys.template.choice.category ]
If [ Get(ScriptParameter)  = "new" ]
    Set Variable [ $pr; Value:If(Get(ActivePortalRowNumber) = 0;0; calc.item::Item number) ]
    Set Variable [ $ac; Value:quote data::ID number internal ]
    Commit Records/Requests
        [ Skip data entry validation; No dialog ]
    Go to Related Record [ From table: “calc.item”; Using layout: “data.quote data” (quote data) ]
        [ Show only related records; New window ]
    If [ Get(LastError) = 0 ]
        New Record/Request
        Set Field [ quote data::ID calc quote data number; $ac ]
        Set Field [ quote data::Item number; $pr ]
        Sort Records [ Specified Sort Order: quote data::Item number; ascending
            quote data::sys.date time stamp created; ascending ]
            [ Restore; No dialog ]
        Replace Field Contents [ quote data::Item number; Replace with serial numbers: Custom values; Initial value: 1; Increment
            value: 1 ]
            [ No dialog ]
    Else
        New Window [ Height: 1; Width: 1; Top: -350 ]
        New Record/Request
        Set Field [ quote data::ID calc quote data number; $ac ]
        Set Field [ quote data::Item number; 1 ]
    End If
    Close Window [ Current Window ]
End If
#
If [ Get(ScriptParameter)  = "select" ]
    Set Field [ calc.item::Item cost manual; calc.templates::Item cost manual ]
    Set Field [ calc.item::Item list price manual; calc.templates::Item list price manual ]
    Set Field [ calc.item::Item cost discount manual; calc.templates::Item cost discount manual ]
End If
#
If [ Get(ScriptParameter)  = "update text" ]
    Go to Related Record [ From table: “calc.item”; Using layout: “data.quote data” (quote data) ]
        [ Show only related records; New window ]
    If [ Get(LastError)  ≠ 0 ]
        Exit Script [  ]
    End If
    Go to Record/Request/Page
        [ First ]
    Set Variable [ $d; Value:"" ]
    Set Variable [ $pn; Value:"" ]
    Loop
        Set Variable [ $d; Value:$d &
            Let( a =
            Substitute(
            Substitute(  quote data::sys.builder.item description  ;
            [  "pp"; If( ( ( PatternCount (quote data::sys.builder.item description;"!!+1") * quote data::Item quantity) + quote data::Item
            quantity) > 1 ; "s" ;"" ) ]  ;
            [ "!!+1";  GetAsText( quote data::Item quantity + 1 ) ] ;
```

```
            [ "!!";   quote data::Item quantity  ]
            )
            & "¶"; [ "--¶"; "" ];["--";""]
            )
            ;
            If (a = "¶"; ""; a )
            ) ]
        Set Variable [ $pn; Value:$pn &
            Substitute(  quote data::sys.builder.item part number  ;
            ["!!+1";  (quote data::Item quantity + 1) ] ;
            ["!!";   quote data::Item quantity ];
            [  "pp"; "" ]  ;
            [ "--¶"; "" ];
            ["--";""]
            ) ]
        Go to Record/Request/Page
            [ Next; Exit after last ]
    End Loop
    Close Window [ Current Window ]
    Commit Records/Requests
        [ Skip data entry validation; No dialog ]
    Set Field [ quote data::Item description; If(Right ($d;1) = "¶"; Left($d;Length($d) - 1);$d) ]
    Set Field [ quote data::Item part number; Substitute(  $pn; "¶" ; "") ]
    Set Field [ quote data::Item picture; calc.templates::Item picture ]
End If
#
If [ Get(ScriptParameter)  = "update numbers" ]
    Commit Records/Requests
        [ Skip data entry validation; No dialog ]
    Set Field [ quote data::Item cost manual; Case(
        Count(calc.item::Item cost manual) = 0 ; "" ;
        quote data::sys.builder.total cost
        ) ]
    Set Field [ quote data::Item cost discount manual; Case(
        Count(calc.item::Item cost discount manual) = 0 ; "" ;
        quote data::sys.builder.total cost margin
        ) ]
    Set Field [ quote data::Item price manual; Case(
        Count(calc.item::Item price manual) = 0 ; "" ;
        quote data::sys.builder.total price
        ) ]
    Set Field [ quote data::Item price discount manual; Round( Case(
        Count(calc.item::Item price discount manual) = 0 ; "" ;
        quote data::sys.builder.total price margin
        ); 1) ]
    Set Field [ quote data::Item list price manual; quote data::sys.builder.total list price ]
End If
#
If [ Get(ScriptParameter)  = "copy temp prices" ]
    Set Field [ calc.templates::Item list price manual; calc.item::Item list price manual ]
    Set Field [ calc.templates::Item cost manual; calc.item::Item cost manual ]
    Set Field [ calc.templates::Item cost discount manual; calc.item::Item cost discount manual ]
End If
Commit Records/Requests
    [ Skip data entry validation; No dialog ]
```

Use new customer from list (fix)

Show Custom Dialog [Title: "Message"; Message: "Replace contact information?¶¶(this will be applied immediately)"; Buttons: “Yes”, “Cancel”]

If [Get(LastMessageChoice) > 1]

    Exit Script [ ]

End If

Set Field [quotes::xxIDL_LinkList; spotlight.customer::xxIDL_IDNumberInternal]

Commit Records/Requests

Copy [contacts.primary::Email]
[Select]

Commit Records/Requests
[Skip data entry validation; No dialog]

Perform AppleScript [Native AppleScript: tell application "Finder"
activate
open application file "get emails.app" of folder "filemaker" of folder "Scripts" of folder "Library" of folder "jim" of folder "Users" of startup disk
end tell
]

Allow User Abort [Off]

Go to Related Record [From table: <unknown>; External; Using layout: <Current Layout>]

Select Window [Current Window]

Set Error Capture [On]

If [not IsEmpty(preferences::<Field Missing>)]

    Perform AppleScript [Calculated AppleScript: <Table Missing>::<Field Missing>]

End If

Set Error Capture [Off]

Perform Script [<unknown> from file: “” (file not open)]

Select Window [Current Window]

Set Field [revision::<Field Missing>; preferences::<Field Missing> & "Emailed Quote¶" & quotes::Quote notes system]

Set Field [contacts.primary::<Field Missing>; preferences::<Field Missing> & "Emailed Quote #" &quotes::Quote number & " " & quotes::Quote summary & "¶" & contacts.primary::<Field Missing>]

If [preferences::<Field Missing> = 1]

    Set Field [contacts.primary::<Field Missing>; preferences::<Field Missing> & "Emailed Quote #" &quotes::Quote number & " " & quotes::Quote summary & "¶" & contacts.primary::<Field Missing>]

End If

If [not IsEmpty(preferences::<Field Missing>)]

    Set Error Capture [On]

    Show Custom Dialog [Title: "Message"; Message: "Continue..."; Buttons: “Continue”]

    Perform AppleScript [Calculated AppleScript: <Table Missing>::<Field Missing>]

End If

Refresh Window

open template list/item2 [item2, item calc]

Set Error Capture [On]

Enter Browse Mode

If [Get(ScriptParameter) = "item2"]

    Go to Related Record [From table: “Template.list view 2”; Using layout: “Template Item” (Template item)]
        [Show only related records; New window]

    Perform Script [“window resize [centered, full]”; Parameter: "centered"]

    Exit Script [ ]

End If

If [Get(ScriptParameter) = "item calc"]

    Set Field [quotes::sys.template.choice.category; quote data::sys.calc.category choice]

    Set Field [quotes::sys.template.choice.subcategory; "All"]

    Go to Related Record [From table: “calc.templates”; Using layout: “Template Item” (Template item)]
        [Show only related records; New window]

    Perform Script [“window resize [centered, full]”; Parameter: "centered"]

    Exit Script [ ]

End If

Commit Records/Requests
    [Skip data entry validation; No dialog]

New Window [Name: "Template List"; Height: 612; Width: 667; Top: 208; Left: 354]

Go to Layout [“Template List” (quotes)]

Show All Records

Set Field [quotes::sys.template.choice.subcategory; If(IsEmpty(quotes::sys.template.choice.category) ; ""; "All")]

Perform Script [“window resize [centered, full]”; Parameter: "centered"]

template list [delete, rename, select, select nw, new, new temp, clear]

Set Error Capture [On]

Freeze Window

If [Get(ScriptParameter) = "delete"]

Show Custom Dialog [Title: "Delete Category"; Message: "Delete entire category and all listed items? " & "¶" & "(items that share more than one category will not be deleted)"; Buttons: “Cancel”, “Continue”]

If [Get(LastMessageChoice) ≠ 2]

Exit Script [ ]

End If

Show Custom Dialog [Title: "Delete Category"; Message: "Confirm, delete entire category and all listed items? " & "¶" & "(items that share more than one category will not be deleted)"; Buttons: “Cancel”, “Continue”]

If [Get(LastMessageChoice) ≠ 2]

Exit Script [ ]

End If

Set Variable [$c; Value:quotes::sys.template.choice.category]

Go to Related Record [From table: “Template.category”; Using layout: “data.templates” (Template item)]
[Show only related records; New window]

If [Get(LastError) = 0]

Replace Field Contents [Template item::sys.builder.item 1 category; Replace with calculation: Substitute(Trim (Substitute(

Substitute("¶" & Template item::sys.builder.item 1 category & "¶"; ["¶" & $c & "¶" ; "¶"])

; ["¶¶"; "¶"]; [" "; "$$"] ;["¶";" "])) ; [" "; "¶"] ; ["$$" ; " "]; ["¶¶"; "¶"])]
[No dialog]

Go to Record/Request/Page
[First]

Loop

If [IsEmpty(Template item::sys.builder.item 1 category)]

Delete Record/Request
[No dialog]

Else

Omit Record

End If

Exit Loop If [Get(FoundCount) = 0]

End Loop

Close Window [Current Window]

Set Field [quotes::sys.template.choice.category; ""]

Set Field [quotes::sys.template.choice.subcategory; ""]

Commit Records/Requests
[Skip data entry validation; No dialog]

End If

End If

#

If [Get(ScriptParameter) = "rename"]

Set Variable [$c; Value:quotes::sys.template.choice.category]

Show Custom Dialog [Title: "Rename Category"; Message: "Specify the new category name..."; Buttons: “Continue”, “Cancel”; Input #1: quotes::sys.template.choice.category]

Set Variable [$nc; Value:Substitute (quotes::sys.template.choice.category; "¶";" ")]

Set Field [quotes::sys.template.choice.category; $c]

If [IsEmpty($nc) or Get (LastMessageChoice) = 2]

Exit Script [ ]

End If

Go to Related Record [From table: “Template.category”; Using layout: “data.templates” (Template item)]

```
        [ Show only related records; New window ]
    If [ Get(LastError) = 0 ]
        Replace Field Contents [ Template item::sys.builder.item 1 category; Replace with calculation: Substitute(  Trim (Substitute(
            Substitute("¶" & Template item::sys.builder.item 1 category & "¶"; ["¶" & $c & "¶" ;"¶"  & $nc & "¶"] )
            ;  [ "¶¶"; "¶"]; [" "; "$$"] ;["¶";" "] )  )  ; [" "; "¶"] ; [ "$$" ; " " ]; [ "¶¶"; "¶"] ) ) ]
            [ No dialog ]
        Close Window [ Current Window ]
        Set Field [ quotes::sys.template.choice.category; $nc ]
    End If
End If
If [ Get(ScriptParameter) = "select nw" ]
    Go to Related Record [ From table: “Template.list view”; Using layout: “Template Item” (Template item) ]
        [ Show only related records; New window ]
    Perform Script [ “window resize [centered, full]”; Parameter: "centered" ]
End If
If [ Get(ScriptParameter) = "select" ]
    Go to Related Record [ From table: “Template.list view”; Using layout: <Current Layout> ]
        [ Show only related records ]
End If
If [ Get(ScriptParameter) = "new" ]
    Set Variable [ $tmp; Value:quotes::sys.template.choice.category ]
    Set Field [ quotes::sys.template.choice.category; "" ]
    Show Custom Dialog [ Title: "Rename Category"; Message: "Specify the new category name..." & "¶" & "(At least one template
        item must be assigned to this category.  A new template item will be created.)"; Buttons: “Continue”, “Cancel”; Input #1:
        quotes::sys.template.choice.category ]
    If [ IsEmpty(quotes::sys.template.choice.category) or Get ( LastMessageChoice ) = 2 ]
        Set Field [ quotes::sys.template.choice.category; $tmp ]
        Exit Script [  ]
    End If
    Set Field [ quotes::sys.template.choice.subcategory; "All" ]
    New Window [ Name: "Template Item"; Height: 1; Width: 1; Top: -80 ]
    Go to Layout [ “Template Item” (Template item) ]
    Perform Script [ “new template item” ]
    Set Field [ Template item::sys.builder.item 3 specific; "<new>" ]
    Set Field [ Template item::Item part number; "<new>" ]
    Close Window [ Current Window ]
    Commit Records/Requests
        [ Skip data entry validation; No dialog ]
End If
If [ Get(ScriptParameter) = "new temp" ]
    New Window [ Name: "Template Item"; Height: 421; Width: 574; Top: 304; Left: 401 ]
    Go to Layout [ “Template Item” (Template item) ]
    Perform Script [ “window resize [centered, full]” ]
    Perform Script [ “new template item” ]
End If
If [ Get(ScriptParameter) = "clear" ]
    Show Custom Dialog [ Title: "Delete Category"; Message: "Clear all dates in the current list?"; Buttons: “Continue”, “Cancel” ]
    If [ Get( LastMessageChoice)  =  2 ]
        Exit Script [  ]
    End If
```

End If

Go to Related Record [From table: “Template.list view”; Using layout: “data.templates” (Template item)]
[Show only related records; New window]

If [Get(LastError) = 0]

Replace Field Contents [Template item::sys.date price modified; Replace with calculation: ""]
[No dialog]

Close Window [Current Window]

Commit Records/Requests
[Skip data entry validation; No dialog]

End If

End If

New Record/Request

Set Field [Template item::sys.builder.item 1 category; quotes::sys.template.choice.category]

Set Field [Template item::sys.builder.item 2 subcategory; If(quotes::sys.template.choice.subcategory= "all" or quotes::sys.template.choice.subcategory = "none";""; quotes::sys.template.choice.subcategory)]

template items [close, copy, delete]

Set Error Capture [On]

Freeze Window

If [Get(ScriptParameter) = "close"]

Commit Records/Requests
[Skip data entry validation; No dialog]

Close Window [Current Window]

Commit Records/Requests
[Skip data entry validation; No dialog]

End If

If [Get(ScriptParameter) = "copy"]

Set Variable [$d; Value:Template item::sys.date price modified]

Duplicate Record/Request

Set Field [Template item::ID number internal; Right("00000000" & Left(Int(Random*9999999);7);7) &
Right("000000" & Get(RecordID);6)]

Set Field [Template item::sys.builder.item 3 specific; If(IsEmpty(Template item::sys.builder.item 3 specific) and IsEmpty
(Template item::sys.builder.item description) and IsEmpty(Template item::sys.builder.item part number);"";Template item::
sys.builder.item 3 specific & " <copy>")]

Set Field [Template item::Item part number; Template item::Item part number & " <copy>"]

Set Field [Template item::sys.date price modified; $d]

Commit Records/Requests
[Skip data entry validation; No dialog]

End If

If [Get(ScriptParameter) = "delete"]

Show Custom Dialog [Title: "Delete template Item"; Message: "Delete template item..."; Buttons: “Continue”, “Cancel”]

If [Get(LastMessageChoice) =2]

Exit Script [ ]

End If

Delete Record/Request
[No dialog]

Commit Records/Requests
[Skip data entry validation; No dialog]

Go to Related Record [From table: “Template.list view”; Using layout: <Current Layout>]
[Show only related records]

End If

new line item from template list/open temp.

Set Error Capture [On]

Freeze Window

Set Variable [$ac; Value:quotes::xxIDL_IDNumberInternalWithRev]

Set Variable [$pr; Value:Get(ActivePortalRowNumber)]

Commit Records/Requests
[Skip data entry validation; No dialog]

Go to Related Record [From table: “Template.list view 2”; Using layout: “data.templates” (Template item)]
[Show only related records; New window]

If [Get(LastError) = 0]

Go to Record/Request/Page [$pr]
[No dialog]

Set Variable [$b; Value:Let([
$t[1] = Template item::Item cost discount manual;
$t[2] = Template item::Item cost manual;
$t[3] = Template item::Item list price manual;
$t[4] = Template item::Item description;
$t[5] = Template item::Item part number
];"")]

Set Field [quotes::sys.picture copy; Template item::Item picture]

Go to Layout [“data.quote data” (quote data)]

New Record/Request

Set Field [quote data::ID quote number w Rev; $ac]

Set Field [quote data::Item cost discount manual; $t[1]

/*
et([
$t[1] = Template item::Item cost discount manual;
$t[2] = Template item::Item cost manual;
$t[3] = Template item::Item list price manual;
$t[4] = Template item::Item description;
$t[5] = Template item::Item part number
];"")
*/]

Set Field [quote data::Item cost manual;
$t[2]

/*
et([
$t[1] = Template item::Item cost discount manual;
$t[2] = Template item::Item cost manual;
$t[3] = Template item::Item list price manual;
$t[4] = Template item::Item description;
$t[5] = Template item::Item part number
];"")
*/]

Set Field [quote data::Item list price manual;
$t[3]

/*
et([
$t[1] = Template item::Item cost discount manual;
$t[2] = Template item::Item cost manual;
$t[3] = Template item::Item list price manual;
$t[4] = Template item::Item description;
$t[5] = Template item::Item part number
];"")
*/]

Set Field [quote data::Item description;
$t[4]

/*
et([
$t[1] = Template item::Item cost discount manual;
$t[2] = Template item::Item cost manual;

```
        $t[3] = Template item::Item list price manual;
        $t[4] = Template item::Item description;
        $t[5] = Template item::Item part number
        ];"")
        */ ]
    Set Field [ quote data::Item part number;
        $t[5]

        /*
        et([
        $t[1] = Template item::Item cost discount manual;
        $t[2] = Template item::Item cost manual;
        $t[3] = Template item::Item list price manual;
        $t[4] = Template item::Item description;
        $t[5] = Template item::Item part number
        ];"")
        */ ]
    Set Field [ quote data::Item picture; quotes::sys.picture copy ]
    Close Window [ Current Window ]
    Commit Records/Requests
        [ Skip data entry validation; No dialog ]
End If
```

Set Field [quotes::System transfer order number; <Table Missing>::<Field Missing>]

copy from order to entry (fix)

Allow User Abort [Off]

Set Field [quotes::<Field Missing>; Get(ActivePortalRowNumber)]

Set Field [Order transfer data::<Field Missing>; "default"]

Set Field [default::<Field Missing>; quotes::<Field Missing>]

Perform Script [<unknown> from file: “” (file not open)]

Select Window [Current Window]

Go to Field [Order transfer data::Item cost]
[Select/perform]

Go to Portal Row [quotes::<Field Missing>]
[Select; No dialog]

Perform Script [“<unknown>”]
Set Field [quote data::Item price manual; order data all::Item price]
Set Field [quote data::Item quantity; order data all::Item quantity]
If [IsEmpty(order data all::Item part number)]
    Set Field [quote data::Item part number; order data all::Item vendor part number]
Else
    Set Field [quote data::Item part number; order data all::Item part number]
End If
If [/*not IsEmpty(order data::<Field Missing>)*/]
    Set Field [quote data::<Field Missing>; /*order data::<Field Missing>*/]
    Set Field [quote data::<Field Missing>; /*order data::<Field Missing>*/]
End If
Set Field [quote data::Item description; order data all::Item description]
Set Field [quote data::Item cost manual; order data all::Item cost]
Set Field [order data all::<Field Missing>; ""]

Show Custom Dialog [Message: "height: " & Get (WindowHeight) & "¶" &
"width: " & Get (WindowWidth) & "¶" &
"top: " & Get (WindowTop) & "¶" &
"left: " & Get (WindowLeft) & "¶"; Buttons: “OK”, “Cancel”]

show pictures [copy]

If [Get(ScriptParameter) = "copy"]

Copy [picture::Item picture]
    [Select]

Commit Records/Requests
    [Skip data entry validation; No dialog]

Else

New Window [ ]

Go to Layout [“Pictures” (quotes)]

Perform Script [“window resize [centered, full]”; Parameter: "centered"]

End If

Go to Object [Object Name: Case(
GetLayoutObjectAttribute("tab.noscratch";"isFrontTabPanel") ; "tab.scratch" ;
"tab.noscratch"
)]

Commit Records/Requests
[Skip data entry validation; No dialog]