

Oracle SaaS Redwood SCM-Procurement pages.

Export 10000+ records to csv from any Redwood page.

PROBLEM NOT FULLY SOLVED YET.

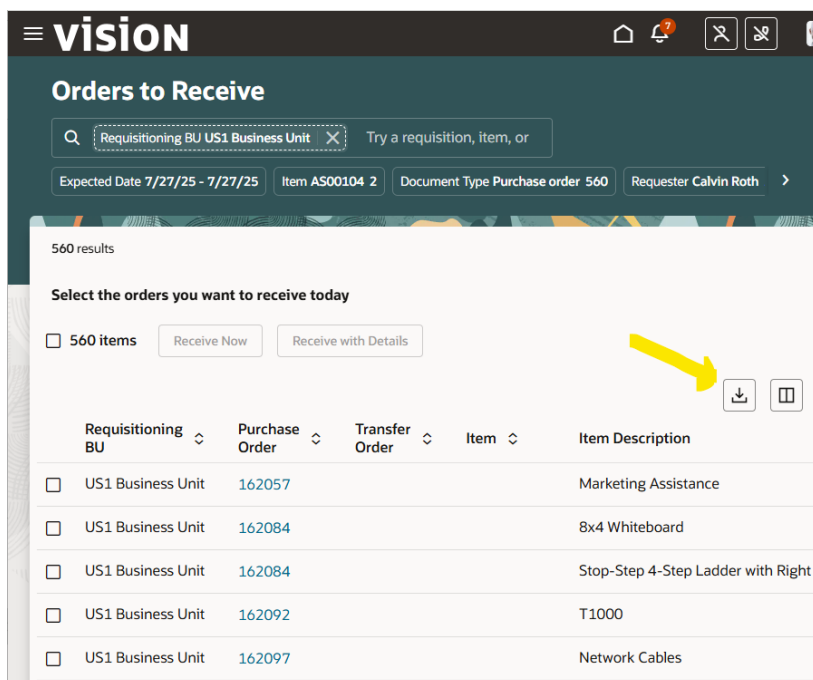
Work in progress...keep reading.

Note: to Visual Builders experts in JavaScript, you will be welcome to take the code and improve performance. Please, post here!

Introduction/Goal.

In Oracle Customer connect we have seen quite a few requirements in several modules (Redwood approach), to export 10000+ records to csv from the following type of pages.

Receiving.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
10		US1 Busin	162034		Closed	Lee Supplies	0.00 USD	Calvin Roth				US1 Business Unit	Calvin Roth	
11		US1 Busin	162036			Printer M32240	2	0 Ea				Lee Supplies	#####	
12		US1 Busin	162281		CM50009	Vision E3-1290 4 Cor	100	0 Ea				Midtown Computer	#####	
13		US1 Busin	162303			Stop-Step	203682	1	0 Ea			JGA	#####	
14		US1 Busin	162304			Stop-Step	203683	1	0 Ea			JGA	#####	
15		US1 Busin	162306			Color Pho	203685	1	0 Ea			Lee Supplies	#####	
16		US1 Busin	162518			Color Pho	203696	2	1 Ea			Lee Supplies	#####	
17		US1 Busin	162719			Color Printer		2	0 Ea			Advanced Corp	#####	
18		US1 Busin	162802			Stop-Step	203728	1	0 Ea			JGA	#####	
19		US1 Busin	162867			Kensington	203738	1	0 Ea			CDW	#####	
20		US1 Busin	162890			Stop-Step	203740	1	0 Ea			JGA	#####	
21		US1 Busin	162891			Installatio	203741	2500	0	USD		Lee Supplies	#####	
22		US1 Busin	162897		CM665505	32 GB Plat	203747	440	0 Ea			Midtown Computer	#####	
23		US1 Busin	162900			Stop-Step	203750	1	0 Ea			JGA	#####	
24		US1 Busin	162978			Legal Servi	203752	200	0	USD		JGA	#####	
25		US1 Busin	162980			Kensington	203752	1	0 Ea			CDW	#####	
26		US1 Busin	163149		AS00104	Blue Hat Enterprise L		3	0 Ea			Lee Supplies	#####	
27														
28														
29														

Maximum number of records=10.000

Purchase Orders.

Purchase Orders

Create Purchase Order

Invoice Holds

1

Overdue

131

Draft and Rejected

78

Processing Errors

16

Try an order number, supplier, or item

Purchase Orders All

Requisitioning BU

Order Status

Buyer

Supplier

Filters

3082 items

Edit

Duplicate

Communicate

More Actions

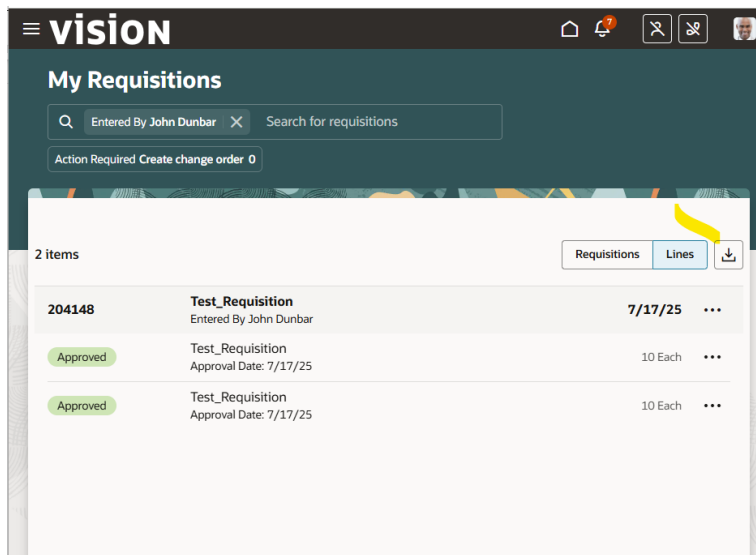
Context Orders

Ord	Order Status	Order Description	Supplier	Total	Buyer
<input type="checkbox"/> US16517	Incomplete		GE	10.95 USD	Calvi
<input type="checkbox"/> US16517	Incomplete		Midtown Computer Supplies	0.00 USD	Calvi
<input type="checkbox"/> US16517	Incomplete		Midtown Computer Supplies	1,000.00 USD	Calvi
<input type="checkbox"/> US16516	Incomplete		Midtown Computer Supplies	0.00 USD	Calvi

This one takes a while to complete-download.

3073	162034	Closed		Lee Supplies	0.00 USD	Calvin Roth		US1 Business Unit	Calvin Roth
3076	162033	Canceled		Lee Supplies	0.00 USD	Calvin Roth		US1 Business Unit	Calvin Roth
3077	162032	Closed for Receivir		Midtown Comput	85,136.25 USD	Calvin Roth		US1 Business Unit	Calvin Roth
3078	162031	Closed for Receivir		Lee Supplies	349.26 USD	Calvin Roth	Multiple	US1 Business Unit	Calvin Roth
3079	162030	Closed		JGA	219.00 USD	Calvin Roth	203642	US1 Business Unit	Calvin Roth
3080	162029	Closed for Receivir		Dell Inc.	1,335.11 USD	Calvin Roth	203638	US1 Business Unit	Calvin Roth
3081	162028	Closed for Receivir		Lee Supplies	2,897.37 USD	Calvin Roth		US1 Business Unit	Calvin Roth
3082	162027	Closed		Lee Supplies	163.16 USD	Calvin Roth	203637	US1 Business Unit	Calvin Roth
3083	162026	Closed for Receivir		JGA	305.51 USD	Calvin Roth	203636	US1 Business Unit	Calvin Roth

Purchase requisitions.



We have analyzed the challenge to cover this gap, and we have “failed”, well, **we have succeeded** but the resulting performance is not good.

Sure, there will be other approaches to accomplish the goal, but not this one for now.

Anyway, we have learned so much in the process that we consider it is worth sharing the code and ideas.

The most important features shown would be:

- Differences between ADP and SDP in Visual Builder.
- Some JavaScript code to manage ADP variables.
- Export component provided by Visual Builder.
- REST API offset feature.
- JavaScript function internal to Action Chain.

Preparatory steps

Connection.

We will use the Receivables transactions REST API for our testing.

vbredwoodapp																											
sn_ar_trx x																											
Overview	Servers	Endpoints	Headers Source																								
<input type="text" value="Filter Endpoints"/> + Endpoint <input type="button" value="Q"/>																											
<div> <div>receivablesInvoices</div> <div> <div>/receivablesInvoices</div> <table> <tr> <td>GET</td> <td>Get Many</td> <td>Get all</td> <td>getall_receivablesinvoices</td> </tr> <tr> <td>POST</td> <td>Create</td> <td>Create</td> <td>create_receivablesinvoices</td> </tr> </table> <div>/receivablesInvoices/{receivablesInvoices_id}</div> <table> <tr> <td>GET</td> <td>Get One</td> <td>Get</td> <td>get_receivablesinvoices</td> </tr> <tr> <td>PATCH</td> <td>Update</td> <td>Update</td> <td>update_receivablesinvoices</td> </tr> <tr> <td>DELETE</td> <td>Delete</td> <td>Delete</td> <td>delete_receivablesinvoices</td> </tr> </table> <div>/receivablesInvoices/{receivablesInvoices_id}/action/splitinstallments</div> <table> <tr> <td>POST</td> <td>Create</td> <td>splitinstallments</td> <td>do_splitinstallments_receivablesinvoices</td> </tr> </table> </div> </div>				GET	Get Many	Get all	getall_receivablesinvoices	POST	Create	Create	create_receivablesinvoices	GET	Get One	Get	get_receivablesinvoices	PATCH	Update	Update	update_receivablesinvoices	DELETE	Delete	Delete	delete_receivablesinvoices	POST	Create	splitinstallments	do_splitinstallments_receivablesinvoices
GET	Get Many	Get all	getall_receivablesinvoices																								
POST	Create	Create	create_receivablesinvoices																								
GET	Get One	Get	get_receivablesinvoices																								
PATCH	Update	Update	update_receivablesinvoices																								
DELETE	Delete	Delete	delete_receivablesinvoices																								
POST	Create	splitinstallments	do_splitinstallments_receivablesinvoices																								
< attachments																											

Steps

- Create app. using “Welcome page template”.
- Create these variables.

vbredwoodapp sn_ar_trx main-start x

Page Designer Action Chains (2) Event Listeners (1) Events Types (3) **Variables (20)** JavaScript JSON Settings

Filter

Constants

No constants defined.

Variables

- backgroundcolor
- illustrationBackground
- illustrationForeground
- imageStretch
- mobileImage
- SDP receivablesInvoicesListSDP
- ADP var_trx_ADP
- varGoOn
- varHasMore
- varLimit
- varLoop
- varMaxExportADP
- varMaxExportSDP
- varMaxLoops
- varOffset
- varQuery
- varTotalRows
- varTrxArray_1
- varTrxArray_2
- varTrxArrayForAdp

- Create these input components.

ADP ready for export

Load ADP Export

REST Limit 500	Max Loops 4	Max Rows Export ADP 12000	Current Loop 4	Offset 2000
-------------------	----------------	------------------------------	-------------------	----------------

BillToCustomerName	BusinessUnit	TransactionDate	TransactionNumber	CustomerTransactionId
Conifer International	US1 Business Unit	2013-12-18	4	1
ABC Application Software	US1 Business Unit	2013-12-18	1	2

- Create these table components.

a) For SDP.

The screenshot shows the Page Designer interface with a component palette on the left. A yellow arrow points to the 'Table' component under the 'Data' category. The main canvas displays two sections:

Direct query of Receivables transactions

Bill-to Customer Name	Business Unit	TransactionDate	Transaction Number	CustomerTra
Conifer International	US1 Business Unit	2013-12-18	4	1
ABC Application Software	US1 Business Unit	2013-12-18	1	2
Business World	US1 Business Unit	2013-12-18	3	3

ADP ready for export

Buttons: Load ADP, Export

REST Limit 500	Max Loops 24	Max Rows Export ADP 12000
Current Loop 0	Offset 0	

Fields: BillToCustomerName, BusinessUnit, TransactionDate, TransactionNumber, CustomerTra

No data to display.

b) For ADP.

The screenshot shows the Page Designer interface with the 'Table' component selected in the component palette. The main canvas displays the same two sections as the previous screenshot:

Direct query of Receivables transactions

Bill-to Customer Name	Business Unit	TransactionDate	Transaction Number	CustomerTra
Conifer International	US1 Business Unit	2013-12-18	4	1
ABC Application Software	US1 Business Unit	2013-12-18	1	2
Business World	US1 Business Unit	2013-12-18	3	3

ADP ready for export

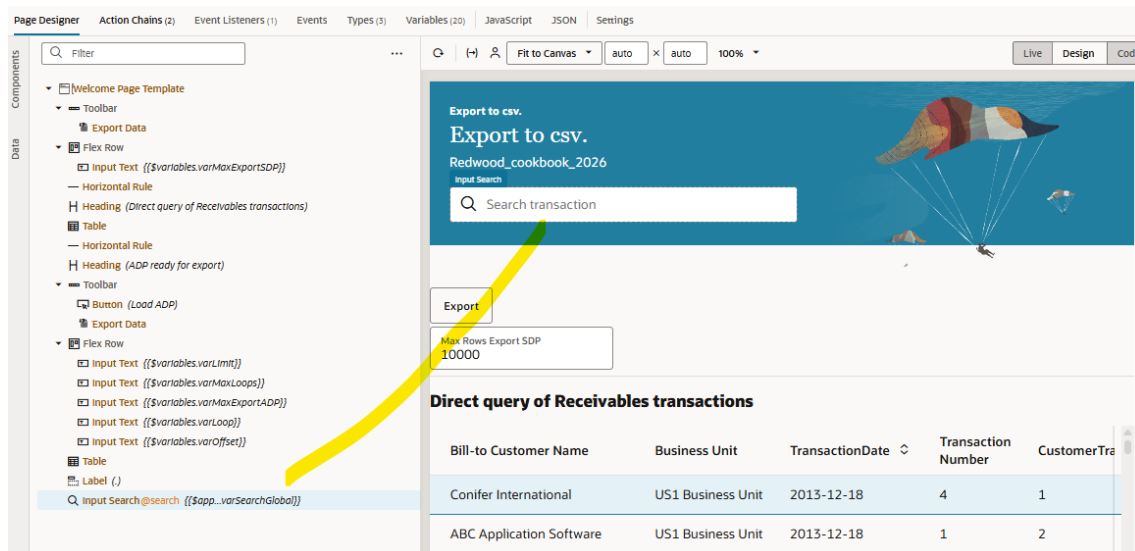
Buttons: Load ADP, Export

REST Limit 500	Max Loops 24	Max Rows Export ADP 12000
Current Loop 0	Offset 0	

Fields: BillToCustomerName, BusinessUnit, TransactionDate, TransactionNumber, CustomerTra

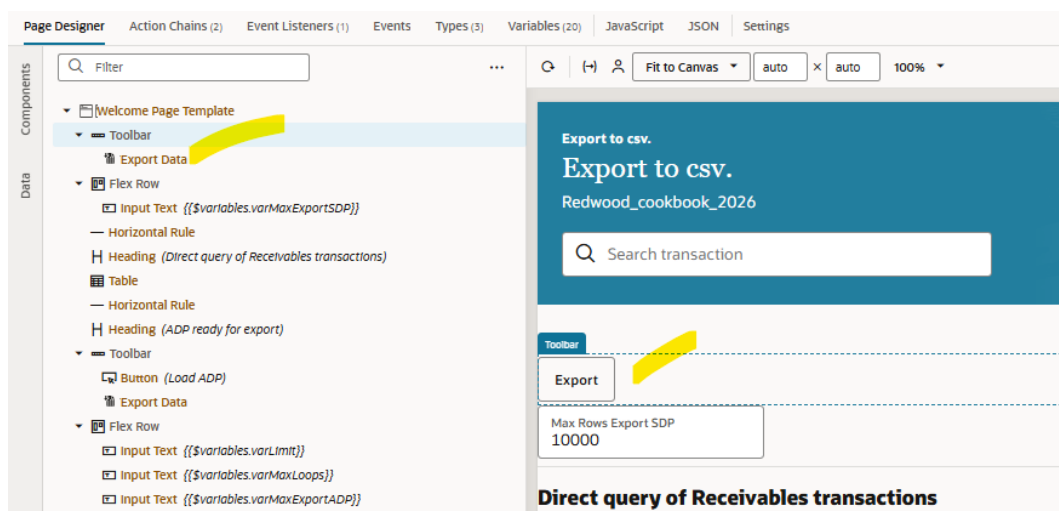
No data to display.

- Create this basic filter.



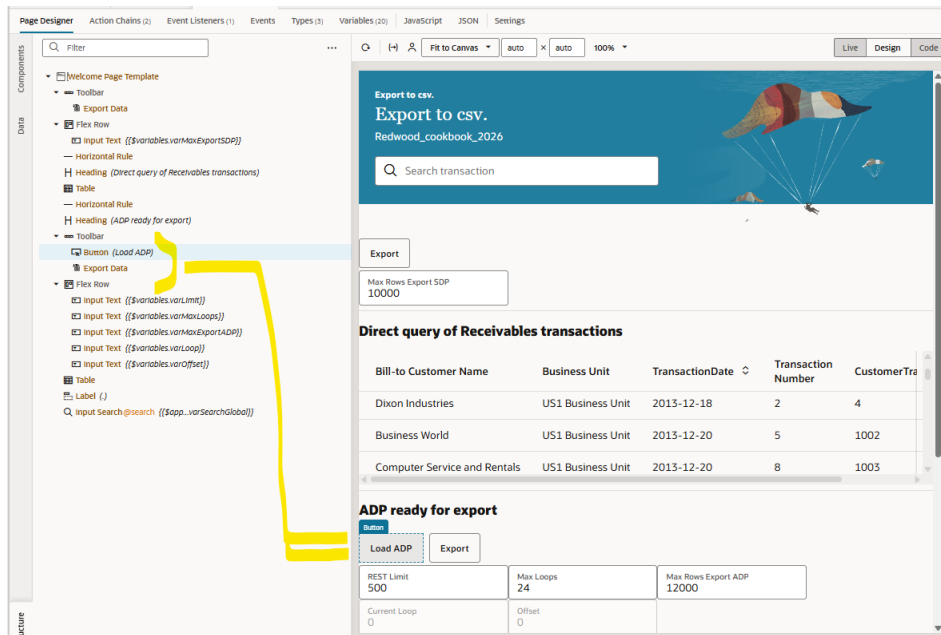
- Create this toolbar 1.

For exporting SDP variable.



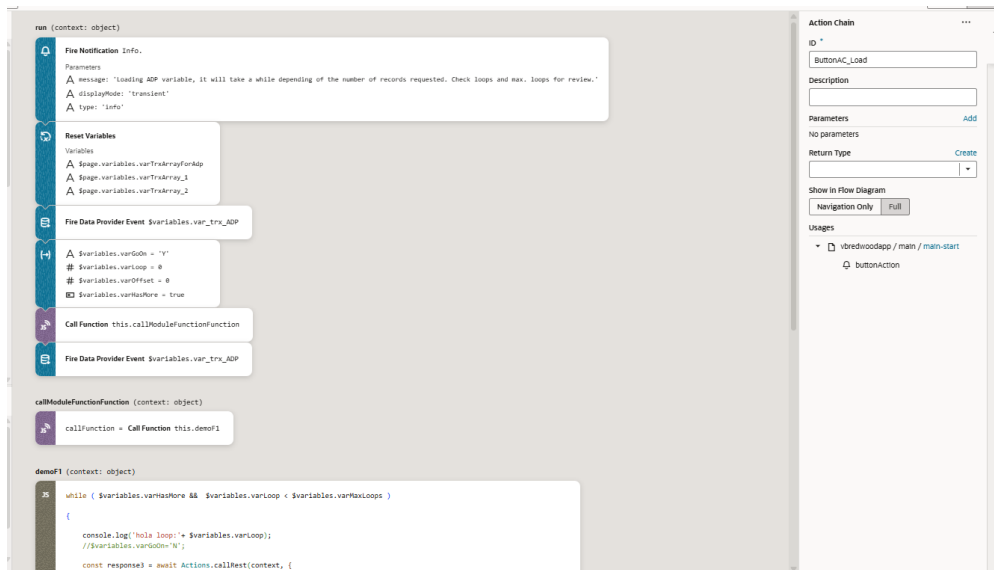
- Create this toolbar 2.

For creating ADP and exporting ADP with JavaScript.



Button Load ADP.

a) Action chain Overview



b) Action chain code.

```

define([
  'vb/action/actionChain',
  'vb/action/actions',
  'vb/action/actionUtils',
], (
  ActionChain,
  Actions,
  ActionUtils
) => {
  'use strict';

  class ButtonAC_Load extends ActionChain {

    /**
     * @param {Object} context
     */

```



```

async run(context) {
  const { $page, $flow, $application, $constants, $variables, $functions } = context;

  await Actions.fireNotificationEvent(context, {
    summary: 'Info.',
    message: 'Loading ADP variable, it will take a while depending of the number of records requested. Check loops and max.
loops for review.',
    displayMode: 'transient',
    type: 'info',
  });

  await Actions.resetVariables(context, {
    variables: [
      '$page.variables.varTrxArrayForAdp',
      '$page.variables.varTrxArray_1',
      '$page.variables.varTrxArray_2',
    ],
  });

  await Actions.fireDataProviderEvent(context, {
    target: $variables.var_trx_ADP,
    refresh: null,
  });

  $variables.varGoOn = 'Y';
  $variables.varLoop = 0 ;
  $variables.varOffset = 0 ;
  $variables.varHasMore = true ;

  await this.callModuleFunctionFunction(context);

  await Actions.fireDataProviderEvent(context, {
    target: $variables.var_trx_ADP,
    refresh: null,
  });
}

/**
 * @param {Object} context
 */
async callModuleFunctionFunction(context) {
  const { $page, $flow, $application, $constants, $variables } = context;

  const callFunction = await this.demoF1(context);
}

/**
 * @param {Object} context
 */
async demoF1(context) {
  const { $page, $flow, $application, $constants, $variables } = context;

  while ( $variables.varHasMore && $variables.varLoop < $variables.varMaxLoops )

  {

    console.log('demo loop:'+ $variables.varLoop);
    //$variables.varGoOn='N';

    const response3 = await Actions.callRest(context, {
      endpoint: 'sn_ar_trx/getall_receivablesInvoices',
      responseBodyFormat: 'json',
      responseType: 'getAllReceivablesInvoicesResponse',
      uriParams: {
        limit: $variables.varLimit,

        orderBy: 'CustomerTransactionId',
        onlyData: true,
        offset: $variables.varOffset,
      },
    },

```

```

    }, { id: 'fn00003' });

    $variables.varTrxArray_2 = response3.body.items;

    const addAdp = await $page.functions.add_adp($variables.varTrxArray_1, $variables.varTrxArray_2);
    $variables.varTrxArray_1 = addAdp;

    console.log('demo zz0: ');

    //$variables.varTotalRows = response3.body.totalResults;

    $variables.varHasMore = response3.body.hasMore;
    console.log('demo 0: '+' hasmore: '+ $variables.varHasMore );
    console.log('demo 2: '+' xx: '+ $variables.varTrxArray_1.length );
    console.log('demo 2: '+' xx: '+ $variables.varTrxArray_2.length );
    console.log('demo 2: '+' xx: '+ $variables.varTrxArrayForAdp.length );

    $variables.varOffset=Number($variables.varOffset)+Number($variables.varLimit);

    $variables.varLoop = $variables.varLoop + 1 ;

}

$variables.varTrxArrayForAdp = $variables.varTrxArray_1;

}

}

return ButtonAC_Load;

});

```

c) JavaScript code.

```

define([], () => {
    'use strict';

    class PageModule {
    }

    PageModule.prototype.add_adp = function (totalAdp, newAdp) {

        let varTotalAdp = totalAdp;

        for (let i = 0; i < newAdp.length; i++) {

            let ChargeComponentObj = {};
            ChargeComponentObj.BillToCustomerName = newAdp[i].BillToCustomerName;
            ChargeComponentObj.BusinessUnit = newAdp[i].BusinessUnit;
            ChargeComponentObj.CustomerTransactionId = newAdp[i].CustomerTransactionId;
            ChargeComponentObj.TransactionDate = newAdp[i].TransactionDate;
            ChargeComponentObj.TransactionNumber = newAdp[i].TransactionNumber;

            varTotalAdp.push(ChargeComponentObj);

        }

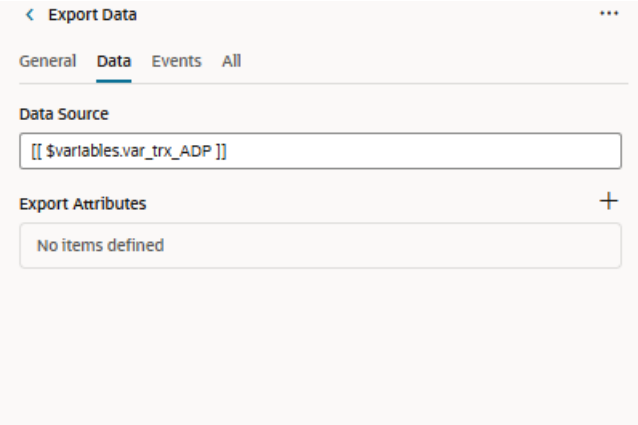
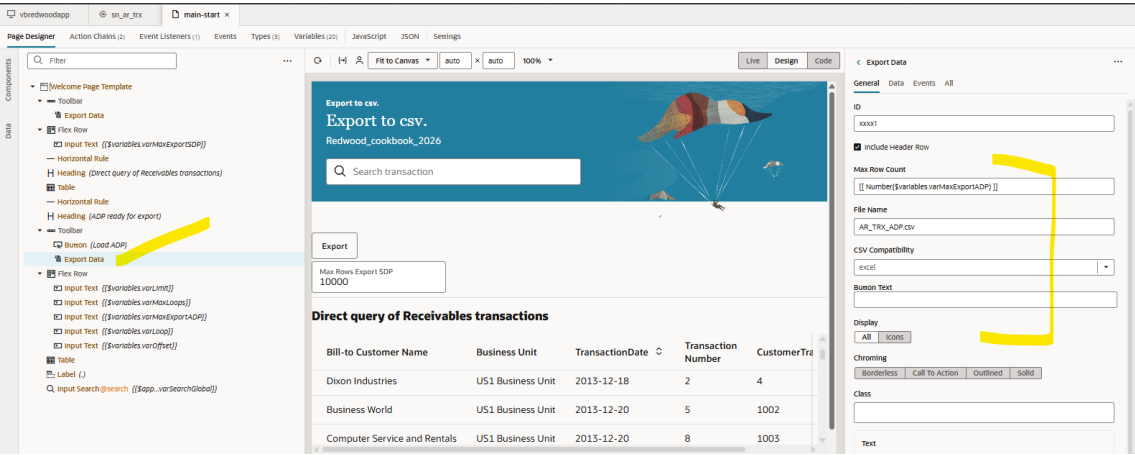
        return varTotalAdp;

    };

    return PageModule;
});

```

Button Export.



Testing.

Play with these 3 fields for your testing.

The screenshot shows a web application interface for exporting transactions. At the top, there's a header with the title 'Redwood_cookbook_2026' and a search bar labeled 'Search transaction'. Below the header, there's a section for 'Export SDP' with a button and a field for 'Max Rows Export SDP' set to 1000. The main section is titled 'Direct query of Receivables transactions' and contains a table with the following data:

Bill-to Customer Name	Business Unit	TransactionDate	Transaction Number	CustomerTransactionId
Conifer International	US1 Business Unit	2013-12-18	4	1
ABC Application Software	US1 Business Unit	2013-12-18	1	2
Business World	US1 Business Unit	2013-12-18	3	3

Below this table, there's a section titled 'ADP ready for export' with buttons for 'Load ADP' and 'Export ADP'. It also has input fields for 'REST Limit' (500), 'Max Loops' (1), 'Max Rows Export ADP' (12000), 'Current Loop' (1), and 'Offset' (500). A yellow arrow points from the 'REST Limit' field to the 'BillToCustomerName' column header of the table below. This table has the same columns as the one above and contains four rows of data:

BillToCustomerName	BusinessUnit	TransactionDate	TransactionNumber	CustomerTransactionId
Conifer International	US1 Business Unit	2013-12-18	4	1
ABC Application Software	US1 Business Unit	2013-12-18	1	2
Business World	US1 Business Unit	2013-12-18	3	3
Dixon Industries	US1 Business Unit	2013-12-18	2	4

REST Limit means the number of records retrieved in each REST API call.

Max loops means the number of loops you want.

Max Rows Export ADP is the number of records you want really exported. It could be a number smaller than the 2 previous multiplied. Example: $500 \times 3 = 1500$ retrieved, but you could export only 600.

Conclusion

We have achieved the goal of exporting 10.000+ records from Redwood page (BUT...)

Export to csv.
Export to csv.
Redwood_cookbook_2026

Q Search transaction

Export

Max Rows Export SDP
10000

Direct query of Receivables transactions

Bill-to Customer Name	Business Unit	TransactionDate ↕	Transaction Number	CustomerTransactionId ↕
Conifer International	US1 Business Unit	2013-12-18	4	1
ABC Application Software	US1 Business Unit	2013-12-18	1	2
Business World	US1 Business Unit	2013-12-18	3	3

ADP ready for export

Load ADP

Export

REST Limit
500

Max Loops
24

Max Rows Export ADP
12000

Current Loop
0

Offset
0

BillToCustomerName ↕

BusinessUnit ↕

TransactionDate ↕

TransactionNumber ↕

CustomerTransactionId ↕

No data to display.

But the performance is poor when 10.000 records are requested, so perhaps we should not use this approach.

We have learned a lot in the process.

Technical

Code example	Comments
XX_AR_TRX_20K_EXPORT-1.0.3.zip	Demo code in exercise.